Volume 2

Documentation and Listings
Original Lanczos Codes

Lanczos Algorithms for Large Symmetric Eigenvalue Computations

Jane K. Cullum

Ralph A. Willoughby

Originally Published by Birkhäuser-Boston in 1985

- **Volume 1: Theory**

  Volume 1: Theory has been republished verbatim in 2002 as Volume 41 in the SIAM Book Series, Classics in Applied Mathematics. SIAM Publications, Philadelphia, PA. ISBN 0-89871-523-7. (A short Errata for Volume 1 is included in this web version of Volume 2 as Chapter 10.)

- **Uni-Processor and Parallel/Fortran90 Versions**

  Currently, Leonard Hoffnung (Math. Dept, U. Kentucky), Spencer Shellman, (Comp. Sc. Dept, U. Utah) and Jane Cullum (*cullumj@lanl.gov*) are working on uni-processor and on MPI parallel Fortran90 versions of the codes contained in Volume 2. The Hoffnung and Shellman contributions are supported currently by a U.S. Department of Energy, Office of Science, MICS, Los Alamos AMS Program grant. The resulting codes will be made available via the Netlib software repository.

- **Matrix Size**

  This book was published 17 years ago. Computers of today are orders of magnitude faster and have orders of magnitude more memory and storage than those which were availabe when this book was written. Seventeen years ago, a matrix of size $10,000$ was considered very large. Since 1985 some of the algorithms which are included in this book have been used on problems of size a million or more. The requirements are *accurate* matrix computations and *sufficient* computer arithmetic precision.

- **Thanks to Leonard Hoffnung for his help in converting ancient 'script' files for Volume 2.**

Contents

LANCZOS ALGORITHMS

Chapter 1

Lanczos procedures

1.1 Introduction

The FORTRAN codes contained in this volume are designed for computing eigenvalues and eigenvectors or singular values and singular vectors of large, sparse matrices. Large means of order several hundred to perhaps 10,000. The largest matrix which we tested was real symmetric and had order 4900. This book is divided into 9 chapters. In this first chapter we give a brief description of Lanczos eigenelement procedures and then make some comments about what the Lanczos codes in this book can and cannot be expected to compute. Detailed analyses of the ideas used in these procedures are contained in Volume 1 of this book.

Chapters 2 through 7 contain procedures which are based upon the single-vector Lanczos recursion with no reorthogonalization of any kind. Six different classes of problems are addressed in these 6 chapters: Eigenelement computations for

1. Real symmetric matrices (Chapter 2)
2. Hermitian matrices (Chapter 3)
3. Factored inverses of real symmetric matrices (Chapter 4)
4. Real symmetric, generalized problems (Chapter 5)
5. Nondefective, complex symmetric matrices (Chapter 7)
6. Singular value and vector computations for real, rectangular matrices (Chapter 6).

Chapters 8 and 9 contain Lanczos procedures which are based upon 'block' versions of the Lanczos recursions. These iterative block procedures include some reorthogonalization within each iteration, but this reorthogonalization is limited to reorthogonalizations w.r.t. certain vectors in each first Lanczos block.

The single-vector procedures can be used to compute anywhere from a very few to very many eigenvalues (singular values). These eigenvalues (singular values) need not be at the extremes of the spectrum. For some matrices it is even possible to compute all of the eigenvalues. The iterative block procedures can only be used to compute a few extreme eigenvalues of the specified matrix. The single vector codes consist of two phases. First eigenvalues or singular values are computed and then corresponding eigenvectors or singular vectors are computed. The iterative 'block' codes compute eigenvalues and corresponding eigenvector approximations simultaneously. Block codes for computing singular values are not included in this book. See for example, Golub, Luk, and Overton [13] for an example of such a block algorithm.

With three exceptions which are given below, each Chapter 2 through 9 contains the following types of information for the particular class of problems considered in that chapter: documentation; main program(s); LANCZS subroutine for computing Lanczos matrices; sample matrix-vector multiply and/or solve subroutines; other subroutines needed by the codes in that chapter; and definitions of the files used by the programs together with sample input files. Because of the similarities between the variables, flags, etc., the documentation for the codes contained in Chapters 2, 3, 4, and 5 was combined and is contained in Section 2.2 of Chapter 2. The codes in Chapters 2, 3, 4, and 5 use essentially (with 2 exceptions) the same set of 'other or additional subroutines' so these subroutines were combined and are given only in Chapter 2, Section 2.6. Similarly, the block codes in Chapters 8 and 9 use the same set of additional subroutines and these are given only in Section 8.5. Some additional optional, preprocessing codes are also provided, and again each of these is included in only one of the chapters and not in each of the ones where it might be useful.

Each set of codes contains many write statements. These write statements serve two major functions: to provide consistency checks on the information supplied by the user, and to provide running commentary on the progress of the computations. Much of the code has been modularized to help make the program logic more transparent to the user. These codes are not designed as efficiently as they could be. Many internal comments have been included. Numerous consistency checks have been used to verify that the user has set up the procedure properly. Basically, we have compromised some efficiency for safety and robustness.

Each LANCZS subroutine together with the corresponding sample matrix-vector multiply and solve subroutines are in files labelled as *MULT. For example in Chapter 2 where real symmetric matrices are discussed this file is labelled LEMULT. The user should note that within a given *MULT file, each sample USPEC* and *MATV subroutine has been given two names so that these subroutines can co-exist with similar subroutines for other test matrices. However, two different *MULT files cannot co-exist because subroutine names are reused in going from one category of matrices to another category. In particular for the codes in Chapters 2, 3, and 7, the matrix-vector multiply subroutine is called CMATV. Moreover, in all of the chapters, the matrix specification subroutines are called USPEC. This reuse of names makes it easier for the user to pass from one set of codes to another. Furthermore, from category to category, subroutines with similar function were typically given the same name. For example, all of the subroutines which generate families of Lanczos matrices are named LANCZS. There are two BISEC bisection subroutines for computing eigenvalues of real symmetric tridiagonal matrices, one for Chapters 2, 3, 4, and 5 and the other one is for Chapter 6. If these sets of codes had to co-exist in one computer file, then it would be necessary for the user to devise a scheme for renaming those subroutines which have the same names.

With respect to portability, each of these programs and subroutines has been individually checked for portability by the PFORT Verifier [22], but the communications between these subroutines have not been checked. Obvious problems with portability like non-Fortran items in the format statements have all been removed. However, certain nonportable constructions have been retained because they make the programs somewhat easier to use. The header of each of the programs contains a list of those constructions in that program which were identified by the PFORT verifier as being nonportable. These headers can be used to locate the nonportable items so that if necessary they can be modified. A list of most of the nonportable items and the reasons for retaining them are given in Table 1.1.

The single vector Lanczos codes in Chapters 2 through 7 are essentially self-contained. The user must provide the matrix-vector multiply and/or solve subroutines which are required by these codes, together with a matrix specification subroutine which defines, dimensions and initializes the matrix which will be used by the Lanczos procedure. The sample matrix-specification subroutines and sample matrix-vector multiply and solve subroutines contained with these codes can be modified and used if appropriate or they can be replaced completely. All of these procedures require a random number generator subroutine, inner product subroutines, and a subroutine to mask underflow. These procedures assume that each time the random number generator is called that the seed for this generator is automatically reset to a different value.

The iterative 'block' Lanczos codes in Chapters 8 and 9 require matrix specification and matrix-vector

Nonportable Construction	Where Used	Why Used
Entry	Passes storage locations of arrays and parameters needed to define user-specified matrix from subroutine USPEC where arrays are dimensioned and initialized to the corresponding matrix-vector multiply or solve subroutine.	Codes do not need to 'see' the user-specified matrix. Codes need only output from matrix-vector multiply or solve subroutines for the matrix being used. User does not have to alter the calling sequences to these subroutines every time the number or kind of arrays needed to define the given matrix is changed.
Formats (20A4) and (4Z20)	(20A4) is used to read and write explanatory comments within the main programs and in sample USPEC subroutines. Machine format (4Z20) is used to read in and write out the Lanczos tridiagonal matrices generated and other quantities for which conversion errors could cause numerical problems.	Allows the user to easily modify headers describing the matrix and code being used. Prevents format conversion errors incurred in input/output conversions.
Free Format Read (5,*)	Used in main program and in sample USPEC subroutines on read-ins of user-specified parameters from input file 5.	Ease of input. User does not have to have the input values properly aligned in the input file.
Complex*16 Variables	Used only in the Hermitian and in the complex symmetric Lanczos codes.	Computations require double precision complex arithmetic.
Specification of Machine Epsilon	Used in main programs	Required to define tolerances used at various points in the computations.

Table 1.1: Nonportable Constructions Used in the Codes

multiply and solve subroutines very similar to those used in the single vector codes, plus the same type of random number generating subroutine, inner product subroutine, and mask subroutine. However, as implemented here the block codes are not self-contained. These codes call two subroutines from the EISPACK Library [23, 8], TRED2 and IMTQL2, which are used repeatedly to compute the eigenvalues and eigenvectors of the small Lanczos matrices generated on each iteration of the block procedures. The user can of course replace these calls by calls to subroutines which perform similar functions, if the EISPACK Library is not available.

The optional preprocessing programs in Sections 2.7, 4.5, 6.7, and 7.7 are stand-alone (if one includes the programs which must be supplied by the user), except for the subroutine PERMUT given in Section 4.5. PERMUT can be used in conjunction with the procedures in Chapters 4, 5, and 9. It calls the SPARSPAK Library [9] (A. George, J. Liu, E. Ng, U. Waterloo) to try to determine a reordering of the given sparse matrix for which the sparsity of the given matrix translates into a sparse factorization of the reordered matrix.

1.2 What are Lanczos procedures?

Lanczos procedures for computing eigenvalues and eigenvectors of real symmetric matrices are based upon one or more variants of the basic single-vector Lanczos recursion for tridiagonalizing a real symmetric matrix A. Given a starting vector v_1 which is typically-generated randomly, the Lanczos recursion implements a Gram-Schmidt orthogonalization of the matrix-vector products Av_i corresponding to the Lanczos vectors v_i generated by the recursion. See for example Bjorck [1]. Specifically, we have that for $i = 2, \ldots, m$,

$$\beta_{i+1} v_{i+1} = Av_i - \alpha_i v_i - \beta_i v_{i-1} \tag{1.2.1}$$

where $\alpha_i \equiv v_i^T A v_i$ and $\beta_{i+1} \equiv v_{i+1}^T A v_i$. By definition $\alpha_i v_i$ and $\beta_i v_{i-1}$ are the projections of Av_i onto the two most recently-generated Lanczos vectors v_i and v_{i-1}. In practice to improve the numerical stability of this recursion, the above formulas are replaced by the following ones.

$$\alpha_i \equiv v_i^T (Av_i - \beta_i v_{i-1}) \text{ and } \beta_{i+1} \equiv \|Av_i - \alpha_i v_i - \beta_i v_{i-1}\|. \tag{1.2.2}$$

The α_i as defined in Eqn(1.2.2) correspond to a modified Gram-Schmidt orthogonalization procedure. The formula for β_{i+1} given in Eqn(1.2.2) is theoretically equivalent to the one given with Eqn(1.2.1). However, it is superior numerically because this choice directly controls the sizes of the Lanczos vectors. See Paige [19].

Rewriting Eqn(1.2.1) in matrix form, we obtain

$$AV_j = V_j T_j + \beta_{j+1} v_{j+1} e_j^T \tag{1.2.3}$$

where T_j denotes the real symmetric tridiagonal Lanczos matrix of order j whose diagonal entries are the scalars α_i, $1 \le i \le j$, and whose subdiagonal (superdiagonal) entries are the scalars β_{i+1}, $1 \le i \le j-1$, generated by the Lanczos recursion. In Eqn(1.2.3), $V_j = (v_1, v_2, \ldots, v_j)$, the matrix whose columns are the Lanczos vectors generated by the recursion, and e_j is the coordinate vector whose j-th component is 1 and whose other components are 0.

It is easy to demonstrate by induction that in exact arithmetic each set of vectors V_j generated by the recursion in Eqns(1.2.1) and (1.2.2) is an orthonormal set. Therefore for any A-matrix with n distinct eigenvalues and any starting vector v_1 which has a projection on every eigenspace of A, we have that for each $j \le n$,

$$T_j = V_j^T A V_j. \tag{1.2.4}$$

Thus the symmetric tridiagonal matrices T_j are representations of the projections of the given matrix A onto the subspaces spanned by the corresponding sets of Lanczos vectors V_j. The eigenvalues of these matrices are the eigenvalues of the A-matrix restricted to these subspaces. Since the Lanczos vectors are obtained by orthogonalizing vectors of the form $\{v_1, Av_1, A^2v_1, \ldots,\}$, we expect the eigenvalues of the T_j to provide good approximations to some of the eigenvalues of A, if j is sufficiently large. Clearly, at least theoretically, if we extend the recursion to $j = n$, then the eigenvalues of T_n will be the eigenvalues of A. T_n is simply an orthogonal transformation of A and must therefore have the same eigenvalues as A. Moreover, any Ritz vector $V_j u$ obtained from an eigenvector u of some T_j is an approximation to a corresponding eigenvector of A.

Basic steps in any Lanczos procedure for computing eigenvalues and eigenvectors of 'symmetric' matrices are the following.

1. Use a variant of the Lanczos recursion to transform the given 'symmetric' matrix A into a family of 'symmetric' tridiagonal matrices of varying sizes.
2. Compute eigenvalues and eigenvectors of certain members of this family. Because of the real symmetric tridiagonal structure this is a much simpler problem than computing the eigenvalues and eigenvectors of A directly.
3. Take some or all of these eigenvalues as approximations to eigenvalues of A and map the corresponding eigenvectors of the tridiagonal matrix into Ritz vectors for the matrix A.
4. Use these Ritz vectors as approximations to the eigenvectors of A.

The Lanczos recursion in Eqn(1.2.1) has several properties which make it particularly attractive for dealing with large but sparse matrices. First the given matrix enters the recursion only through the matrix-vector multiply terms Av_i. Thus contrary to what is done in the standard methods for solving small or medium size eigenvalue problems, see for example EISPACK [23, 8], the given matrix is not explicitly modified. The user must provide only a subroutine which computes Ax for any given vector x. If the matrix A is sparse, this computation can be done using an amount of storage that is only linear in the size of the matrix instead of quadratic. Second, the recursion uses only the two most recently-generated Lanczos vectors. The Gram-Schmidt orthogonalization of an arbitrary set of vectors would require that at any given stage in the process that all of the vectors which have already been orthogonalized be available for orthogonalizing each additional vector as it is considered. Thus, the storage requirements for implementing the basic Lanczos recursion are minimal. If we use Eqns(1.2.1) and (1.2.2) then only 2 n-vectors are needed for the two most recently-generated Lanczos vectors plus storage for the α and β arrays.

There are however numerical problems if only a simple direct implementation of this recursion is programmed. In general such an implementation yields Lanczos matrices which have extra eigenvalues in addition to the 'good' eigenvalues which are approximations to eigenvalues of A. These extraneous or 'spurious' eigenvalues are caused by the losses in the orthogonality of the Lanczos vectors which in turn are caused by the combination of the roundoff errors resulting from the finite computer arithmetic and the convergence (as j is increased) of eigenvalues of the Lanczos matrices to eigenvalues of the original matrix A. This interaction between the computer arithmetic and the convergence of eigenvalues is discussed in Paige [17, 20].

During the past $5-10$ years many different types of Lanczos eigenelement algorithms have been proposed. See Volume 1, Chapter 2 of this book for a brief survey of the literature. Most of these procedures incorporate modifications to the basic Lanczos recursion in Eqns(1.2.1) and (1.2.2) which force the Lanczos vectors to stay nearly orthonormal. These approaches require either the repeated computation of Ritz vectors or the repeated reorthogonalization of the Lanczos vectors as they are generated or some combination of these two computations. In either case as the size of the Lanczos matrix generated is increased to be able to compute more eigenvalues, the associated Ritz vectors or the Lanczos vectors needed for the reorthogonalizations require more and more storage. These modifications often work well but destroy much of the simplicity of the basic procedure, and because of the added storage requirements resulting from the reorthogonalizations they limit the number of eigenelements which can be computed.

The approach which we have chosen and which is implemented in the enclosed FORTRAN programs in Chapters 2 through 7 is not to force the orthogonality of the Lanczos vectors by reorthogonalizing, but to work directly with the basic Lanczos recursion, accepting the losses in orthogonality, and then unraveling the effects of these losses. This approach allows us to retain the basic simplicity of the Lanczos recursion, to minimize the storage requirements, and to therefore maximize the number of eigenvalues of A which can be computed. In our approach in the single-vector algorithms in Chapters 2 through 7, Ritz vectors are not computed until after the eigenvalues have been computed accurately. Consequently, the basic storage requirements for our eigenvalue (singular value) algorithms are only a small multiple of the size of the largest Lanczos matrix used in the computations. Thus, we can compute many eigenvalues of very large but sparse matrices. Depending upon what is to be computed and upon the eigenvalue distribution in the given matrix A, the sizes of the Lanczos matrices used in these computations may be much smaller or considerably larger than the original A-matrix. However the Lanczos matrices generated by the procedures in Chapters 2 through 6 are real symmetric and tridiagonal so that these matrices can be very large and still not present insurmountable computational problems. Eigenvalue and eigenvector computations for such matrices require minimal amounts of storage and fairly reasonable numbers of arithmetic operations.

The computational problems which arise from not maintaining near orthogonality of the Lanczos vectors and which we must address in our single-vector codes are of two types. First and most importantly, we must deal with the question of sorting the eigenvalues of the Lanczos matrices into 2 classes, one corresponding to the 'good' eigenvalues which are approximations to the eigenvalues of A and the other corresponding to the extra or 'spurious' eigenvalues caused by the losses in orthogonality. The identification test used for doing this is discussed in Volume 1, Chapter 4, Section 4.5. For the procedures discussed in Chapters 2 through 6, this identification test is an integral and inexpensive part of the eigenvalue (singular value) computations. For the complex symmetric procedure discussed in Chapter 7 this test is handled in a considerably less eloquent manner and is expensive.

The second but much less serious difficulty we must address is the question of false multiplicities. The multiplicity of a particular 'good' eigenvalue as an eigenvalue of the Lanczos matrices is not related to the multiplicity of that eigenvalue as an eigenvalue of the A-matrix. 'Good' eigenvalues may replicate many times as eigenvalues of a Lanczos matrix, but be only simple eigenvalues of the original A-matrix. Thus, these single-vector procedures cannot directly determine the true multiplicities of the computed 'good' eigenvalues. Of course, this latter comment is also applicable to any single-vector Lanczos procedure not just to our procedures. Theoretically, at most one eigenvector for each distinct eigenvalue of the A-matrix can be obtained using the single-vector Lanczos recursion given in Eqns(1.2.1) and (1.2.2). (This of course is not true for iterative block Lanczos procedures.) It is interesting to note however that if the Lanczos recursion is used without any reorthogonalization, then it can yield sets of linearly independent eigenvectors for eigenvalues which are multiple in the A-matrix. The amount of work required to compute these additional eigenvectors depends upon the particular matrix in question and upon the particular eigenvalue. The codes provided in Chapters 2 through 7 of this book do not however incorporate this capability.

The iterative 'block' Lanczos procedures for real symmetric matrices given in Chapters 8 through 9 are based upon a block version of the Lanczos recursion

$$Q_{j+1}B_{j+1} = AQ_j - Q_jA_j - Q_{j-1}B_j^T \tag{1.2.5}$$

for $j = 1, 2, \ldots, s$ where Q_1 is $n \times q$ and the coefficient matrices A_j and B_{j+1} are block analogs of the scalar coefficients in the single-vector Lanczos recursion in Eqns(1.2.1) and (1.2.2). The number of blocks s used on each iteration is chosen such that $qs \ll n$, where n is the order of the given A-matrix and q is chosen such that $q \geq q'$, the number of eigenvalues and eigenvectors desired. The Lanczos matrices are real symmetric, block tridiagonal matrices. In Eqn(1.2.5) we used Q_j instead of V_j because in our block Lanczos procedures we maintain near-orthogonality of the blocks generated within each iteration by incorporating reorthogonalization of the blocks of Lanczos vectors with respect to certain vectors in the first Lanczos block.

The 'block' procedures provided in Chapters 8 and 9 are really hybrid algorithms, something between a true block Lanczos procedure, see for example, Cullum and Donath [4, 3] and Chapter 7 in [5], and the single-vector Lanczos procedures given in Chapters 2 through 7. The sequence of 'blocks' generated on each iteration of this hybrid method has the property that the first Q-block contains at least as many vectors as the user is trying to compute, but the second and succeeding blocks each contain only one vector. The corresponding resulting Lanczos matrices are not block tridiagonal. Each Lanczos matrix has a border of blocks in the first q rows and columns and is tridiagonal below this border.

At the beginning of each chapter, a brief description is given of the particular variant of the Lanczos recursion used in the Lanczos codes included in that chapter, along with some additional comments relevant to the particular types of problems being considered in that chapter.

1.3 Comments and disclaimers

The single-vector Lanczos procedures contained in Chapters 2 through 7 do not behave like standard eigenelement procedures. Their behavior is both non-classical and somewhat unorthodox. If one of these codes were run on two different kinds of computers but with the same original matrix and the same initial specifications, the computed results could be quite different. A primary cause for such differences can of course be a difference in the starting vector caused by a difference in the random number generators. However even if the same starting vector were read in, the results would almost surely differ due to the differences in the computer arithmetic. In practice, the Lanczos matrices generated on two different kinds of computers may agree for a certain number of Lanczos steps but will begin to diverge upon the convergence of one or more of the eigenvalues of these Lanczos matrices to eigenvalues of the A-matrix. If after a reasonable number of steps in the Lanczos recursion we were to compare the entries in the Lanczos matrices generated by the two different computers, the values would probably be very different.

Furthermore, if we were to compute the eigenvalues of the two sets of Lanczos matrices for various sizes and 'spurious' eigenvalues were present, then these spurious eigenvalues would be different and even appear in different portions of the spectrum. In fact, prior to the convergence of a particular 'good' eigenvalue, the values of that good eigenvalue, in terms of how accurate it is at any given stage in the computations, may differ. However once a 'good' eigenvalue in either set has converged, that 'good' eigenvalue will agree with a true eigenvalue of the original user-specified matrix to as many digits as can be expected.

Therefore, if the user carries out the sample eigenvalue computation provided in Chapter 2, he/she should not be alarmed or surprised if the output from the computer being used does not agree with what is shown in the sample, as long as the converged 'good' eigenvalues agree. Actually one may observe different rates of convergence on different kinds of computers, depending upon the computer arithmetic. With increased arithmetic precision in all of the computations, these procedures may converge more rapidly. With decreased precision, they will converge less rapidly. All of our codes use double precision arithmetic (for an IBM 3083) and any precision less than that is not recommended.

Each of these procedures requires the user to supply either a matrix-vector multiply subroutine or a matrix-vector solve subroutine. (Both types of subroutines are required for the codes in Chapter 5 .) Such subroutines should perform the required computations rapidly and accurately, taking advantage of any special properties or structure in the given matrix. Our Lanczos programs see the original matrix as the outputs of these subroutines. The codes provided include sample matrix-vector multiply subroutines for a general sparse 'symmetric' matrix given in a particular sparse format. These are available for the user to use or modify as desired. Note that similar programs are also provided for the singular value/vector computations. Accuracy is important in these subroutines because consistency must be maintained in the information being provided to the LANCZS subroutine which is generating the Lanczos matrices. There is no built-in mechanism for preserving symmetry. Therefore, the matrix-vector multiply and solve subroutines must be coded with care. Without such consistency the Lanczos codes will not function properly.

The convergence characteristics of the two types of Lanczos procedures considered are quite different. These differences are discussed in Chapters 4 and 7 of Volume 1 of this book. However, in both cases, the degree of difficulty in computing the desired eigenvalues depends upon the eigenvalue gaps. For the single-vector procedures the primary factor in determining whether or not it is feasible to compute either large numbers of eigenvalues or the eigenvalues with the smallest gaps, is the gap ratio, the ratio of the largest gap between two neighboring eigenvalues to the smallest such gap. The smaller this ratio, the easier it is to compute all of the eigenvalues of the given matrix. The larger this ratio, the harder it is to compute those eigenvalues with the smallest gaps. The locations of the desired eigenvalues in the spectrum of the given matrix also play a significant role in the rate of convergence of individual eigenvalues. Both types of Lanczos procedures favor extreme eigenvalues. The iterative block codes, in fact, can only compute a few extreme eigenvalues. However for the single-vector codes, it is possible for interior eigenvalues which have gaps which are significantly larger than the gaps for some of the extreme eigenvalues to converge prior to the convergence of those extreme eigenvalues. Examples of the convergence achievable are given in Volume 1, Chapter 4 of this book.

The convergence of the iterative block procedures depends primarily upon the gaps between the eigenvalues being computed and the closest eigenvalue not being approximated, the spread of the matrix, and the overall eigenvalue distribution. The block procedures discussed in Chapters 8 and 9 are iterative and the codes track the rate of convergence. If the observed rate is too slow (as specified by the user), these block procedures will terminate without achieving convergence. The user then has the option of restarting the block procedure with a different choice of parameters and using the current approximation to the basis for the desired eigenspace as the starting vectors.

Thus, the amount of work required for a particular eigenelement computation for a given matrix using a particular method depends directly upon the eigenvalue distribution in that matrix and upon which portion of the spectrum is being computed. Some problems are 'easy', others are hard. Therefore failure can occur, in the sense that these procedures may not be able to compute the information desired by the user within the computational bounds specified by the user. However the single-vector Lanczos procedures, even in 'failure', provide a great deal of information about the eigenvalue spectrum of the given matrix.

In deciding which procedure to use on a given problem, our preference is a single-vector procedure, although the iterative block procedures can often quickly provide simultaneously the desired eigenvalues and eigenvectors. If the user wants extreme eigenvalues and the user knows or suspects that one or more of these is multiple, then the block procedure is probably preferable. More details about the Lanczos procedures contained in this book can be found in Volume 1. Any questions about these programs including the question of obtaining copies of these codes or of problems with these codes, should be addressed directly to the authors. We hope that these codes will prove useful in many different applications in the engineering and scientific community.

Chapter 2

Real Symmetric Matrices

2.1 Introduction

The FORTRAN codes in this chapter address the question of computing distinct eigenvalues and corresponding eigenvectors of real symmetric matrices, using a single-vector Lanczos procedure. For a given real symmetric matrix A, these codes compute real scalars λ and corresponding real vectors $x \neq 0$, such that

$$Ax = \lambda x. \tag{2.1.1}$$

Definition 1 *The real $n \times n$ matrix $A \equiv (a_{ij})$, $1 \leq i, j \leq n$, is a real symmetric matrix if and only if for every i, j, $a_{ij} = a_{ji}$.*

Real symmetric matrices are discussed in detail in Stewart [24]. Properties which we use are:

1. Real symmetric matrices have complete eigensystems. That is, the dimension of the eigenspace corresponding to any given eigenvalue of the given matrix A is the same as the multiplicity of that eigenvalue as a root of the characteristic polynomial of A.

2. For any two distinct eigenvalues of A, λ and μ , and corresponding eigenvectors x and y, $x^T y = 0$. Thus, eigenvectors corresponding to different eigenvalues are orthogonal, and we can construct an eigenvector basis which is orthonormal. Vectors are orthonormal if they are orthogonal and each has a Euclidean norm of 1. (The Euclidean norm of a vector is just the square root of the sum of the squares of its components.)

3. Small perturbations in the matrix cause only small perturbations in the eigenvalues. Of the classes of matrices which we consider, the class of real symmetric matrices is the most well-behaved and thus the 'easiest'.

The Lanczos codes contained in this chapter correspond to the most straight-forward implementation of the Lanczos recursion included in this book. These codes can be used to compute either a very few or very many of the distinct eigenvalues of the given real symmetric matrix. As the documentation in the next section indicates, the A-multiplicity of a given computed 'good' Lanczos eigenvalue can be obtained only with additional computation, and the modifications required to do this additional computation are not included in these versions of the codes. This implementation uses the basic Lanczos recursion given in Eqns (1.2.1) and (1.2.2) in Section 1.2 of Chapter 1 to generate a family of real symmetric, tridiagonal matrices (T-matrices) whose sizes are specified by the user. There is no reorthogonalization of the Lanczos vectors at any stage in any of the computations.

LEVAL, the main program for the real symmetric eigenvalue computations, calls the subroutine BISEC to compute eigenvalues of the user-specified Lanczos tridiagonal matrices on the user-specified intervals. BISEC simultaneously computes these T-eigenvalues with their T-multiplicities and sorts the computed T-eigenvalues into two classes, the 'good' T-eigenvalues and the 'spurious' T-eigenvalues. The 'good' T-eigenvalues are accepted as approximations to eigenvalues of the user-specified matrix A. The accuracy of these 'good' T-eigenvalues as eigenvalues of A is then estimated using error estimates computed by subroutine INVERR. Error estimates are computed only for isolated 'good' T-eigenvalues. All other 'good' T-eigenvalues are assumed to have converged. If convergence has not yet occurred and a larger Lanczos matrix has been specified by the user, the program will continue on to a larger Lanczos matrix, repeating the above procedure on this larger matrix.

Once the eigenvalues have been computed accurately enough, the user can select a subset of the 'converged' eigenvalues for which eigenvectors are to be computed. The main program LEVEC, for computing eigenvectors of real symmetric matrices, is then used to compute these desired eigenvectors.

All computations are in double precision real arithmetic. The user must supply a subroutine USPEC which defines and initializes the user-specified matrix A and a subroutine CMATV which computes matrix-vector multiplies Ax for any given vector x. These subroutines must be constructed in such a way as to take advantage of the sparsity (and/or structure) of the user-supplied A-matrix and such that these computations are done accurately. More details about these real symmetric single-vector Lanczos procedures are given in Chapter 4 of Volume 1 of this book.

2.2 Documentation for the Codes in Chapters 2, 3, 4, 5

```
C-----LEVALHED----------------------------------------------------------LEV00010
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)             LEV00020
C            Los Alamos National Laboratory                             LEV00030
C            Los Alamos, New Mexico 87544                               LEV00040
C                                                                       LEV00050
C            E-mail:  cullumj@lanl.gov                                  LEV00060
C                                                                       LEV00070
C  These codes are copyrighted by the authors.  These codes             LEV00080
C  and modifications of them or portions of them are NOT to be          LEV00090
C  incorporated into any commercial codes or used for any other         LEV00100
C  commercial purposes such as consulting for other companies,          LEV00110
C  without legal agreements with the authors of these Codes.            LEV00120
C  If these Codes or portions of them are used in other scientific or   LEV00130
C  engineering research works the names of the authors of these codes   LEV00140
C  and appropriate references to their written work are to be           LEV00150
C  incorporated in the derivative works.                                LEV00160
C                                                                       LEV00170
C  This header is not to be removed from these codes.                   LEV00180
C                                                                       LEV00190
C        REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4             LEV00200
C        Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsLEV00210
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in    LEV00220
C        Applied Mathematics, 2002. SIAM Publications,                  LEV00230
C        Philadelphia, PA. USA                                          LEV00240
C                                                                       LEV00250
C                                                                       LEV00260
C-----DOCUMENTATION FOR SINGLE-VECTOR-----------------------------------LEV00270
C                                                                       LEV00280
C     LANCZOS EIGENVALUE/EIGENVECTOR PROGRAMS FOR                       LEV00290
C     (1) REAL SYMMETRIC MATRICES                                       LEV00300
C     (2) HERMITIAN MATRICES                                            LEV00310
C     (3) FACTORED INVERSES OF REAL SYMMETRIC MATRICES                  LEV00320
C     (4) REAL SYMMETRIC, GENERALIZED PROBLEMS WHERE ONE OF THE         LEV00330
C         MATRICES IS POSITIVE DEFINITE AND ITS CHOLESKY FACTORS ARE    LEV00340
C         AVAILABLE                                                     LEV00350
C                                                                       LEV00360
C                                                                       LEV00370
C                                                                       LEV00380
C-----------------------------------------------------------------------LEV00390
C                                                                       LEV00400
C     REAL SYMMETRIC MATRICES:                                          LEV00410
C                                                                       LEV00420
C     GIVEN A REAL SYMMETRIC MATRIX A OF ORDER N THE THREE SETS OF      LEV00430
C     FORTRAN FILES LABELLED LEVAL, LESUB, AND LEMULT CAN BE USED TO    LEV00440
C     COMPUTE DISTINCT EIGENVALUES OF THE USER-SPECIFIED MATRIX         LEV00450
C     IN USER-SPECIFIED INTERVALS.                                      LEV00460
C                                                                       LEV00470
C     CORRESPONDING EIGENVECTORS FOR SELECTED, COMPUTED EIGENVALUES CAN LEV00480
C     BE COMPUTED USING THE SETS OF FILES LABELLED LEVEC, LESUB, AND    LEV00490
C     LEMULT.                                                           LEV00500
C                                                                       LEV00510
```

```
C                                                                       LEV00520
C     HERMITIAN MATRICES:                                               LEV00530
C                                                                       LEV00540
C     GIVEN A HERMITIAN MATRIX A OF ORDER N THE THREE SETS OF           LEV00550
C     FORTRAN FILES LABELLED HLEVAL, LESUB, AND HLEMULT CAN BE USED     LEV00560
C     TO COMPUTE DISTINCT EIGENVALUES IN USER-SPECIFIED INTERVALS.      LEV00570
C                                                                       LEV00580
C     CORRESPONDING EIGENVECTORS FOR SELECTED, COMPUTED EIGENVALUES     LEV00590
C     CAN BE COMPUTED USING THE SETS OF PROGRAMS LABELLED HLEVEC,       LEV00600
C     LESUB, AND HLEMULT.                                               LEV00610
C                                                                       LEV00620
C                                                                       LEV00630
C     FACTORED INVERSES OF REAL SYMMETRIC MATRICES:                     LEV00640
C                                                                       LEV00650
C     GIVEN A REAL SYMMETRIC MATRIX A, THE LANCZOS RECURSION IS         LEV00660
C     APPLIED TO THE INVERSE OF A, USING A FACTORIZATION                LEV00670
C     OF A.  THE SETS OF FILES LIVAL, LESUB, AND LIMULT                 LEV00680
C     CAN BE USED TO COMPUTE THE DISTINCT EIGENVALUES OF THE            LEV00690
C     INVERSE OF THE A-MATRIX AND OF A IN  USER-SPECIFIED               LEV00700
C     INTERVALS.  THE PROGRAMS ACTUALLY ALLOW ONE TO WORK WITH          LEV00710
C     ANY MATRIX B = PCP' WHERE  C = SO*A + SHIFT*I, WHERE              LEV00720
C     SO AND SHIFT ARE SCALARS CHOSEN BY THE USER AND P IS A            LEV00730
C     PERMUTATION MATRIX CHOSEN SUCH THAT THE FACTORIZATION             LEV00740
C     OF THE B-MATRIX RETAINS SPARSITY.  IN THE                         LEV00750
C     SAMPLE LIMULT SUBROUTINES PROVIDED, SO AND SHIFT MUST BE          LEV00760
C     CHOSEN SO THAT THE RESULTING B-MATRIX IS POSITIVE DEFINITE,       LEV00770
C     AND THE CHOLESKY FACTORS ARE USED TO SOLVE B*U = V.               LEV00780
C     HOWEVER, THE USER CAN EASILY REPLACE THE SAMPLE USPEC AND         LEV00790
C     BSOLV SUBROUTINES PROVIDED BY SUBROUTINES THAT ALLOW THE          LEV00800
C     GENERAL FACTORIZATION L*D*(L-TRANSPOSE).  THESE LANCZOS           LEV00810
C     PROGRAMS APPLY THE LANCZOS RECURSION TO B-INVERSE, USING          LEV00820
C     THE FACTORIZATION PROVIDED.  OPTIONAL PREPROCESSING PROGRAMS      LEV00830
C     PERMUT, LORDER, LFACT, AND LTEST ARE PROVIDED FOR SET-UP PURPOSES.LEV00840
C     PERMUT USES THE SPARSPAK PACKAGE OF A. GEORGE, J. LIU AND         LEV00850
C     E. NG TO OBTAIN A REORDERING OF THE GIVEN MATRIX THAT             LEV00860
C     PRESERVES SPARSENESS ON SUBEQUENT FACTORIZATION.  LORDER          LEV00870
C     CAN BE USED TO REORDER A GIVEN MATRIX, USING A GIVEN              LEV00880
C     PERMUTATION.  LFACT CAN BE USED TO COMPUTE THE CHOLESKY           LEV00890
C     FACTORS OF A GIVEN POSITIVE DEFINITE B-MATRIX.  LTEST CAN         LEV00900
C     BE USED TO ESTIMATE THE NUMERICAL CONDITION OF THE                LEV00910
C     B-MATRIX.                                                         LEV00920
C                                                                       LEV00930
C     CORRESPONDING EIGENVECTORS FOR SELECTED, COMPUTED                 LEV00940
C     EIGENVALUES CAN BE COMPUTED USING THE SETS OF FILES               LEV00950
C     LABELLED LIVEC, LESUB, AND LIMULT.                                LEV00960
C                                                                       LEV00970
C     GENERALIZED REAL SYMMETRIC PROBLEMS:                              LEV00980
C                                                                       LEV00990
C     GIVEN 2 REAL SYMMETRIC MATRICES A AND B WHERE IN ADDITION B IS    LEV01000
C     POSITIVE DEFINITE AND ITS CHOLESKY FACTORS ARE AVAILABLE,         LEV01010
C     THE SETS OF FILES LGVAL, LGMULT, AND LESUB CAN BE USED            LEV01020
C     TO COMPUTE THE DISTINCT EIGENVALUES OF THE GENERALIZED            LEV01030
C     PROBLEM A*X = EVAL*B*X.                                           LEV01040
C                                                                       LEV01050
C     CORRESPONDING EIGENVECTORS CAN BE COMPUTED USING THE PROGRAMS     LEV01060
```

```
C     LGVEC, LGMULT, AND LESUB.  NOTE THAT THE PREPROCESSING PROGRAMS
C     AVAILABLE FOR USE IN CASE (3) (PERMUT, LORDER, LFACT, AND LTEST)
C     CAN ALSO BE USED IN THIS CASE TO OBTAIN A SUITABLE PERMUTATION,
C     AND A FACTORIZATION OF THE RESULTING B-MATRIX.  THE A-MATRIX
C     CAN THEN BE PERMUTED USING LORDER.
C
C
C     THESE PROGRAMS ALL USE LANCZOS TRIDIAGONALIZATION WITHOUT
C     REORTHOGONALIZATION TO GENERATE REAL SYMMETRIC TRIDIAGONAL
C     MATRICES, T(1,MEV), OF ORDER MEV.  SUBSETS OF THE EIGENVALUES OF
C     THESE T-MATRICES, LABELLED AS THE 'GOOD EIGENVALUES', YIELD
C     APPROXIMATIONS TO THE DESIRED EIGENVALUES.  CORRESPONDING
C     RITZ VECTORS ARE APPROXIMATIONS TO THE DESIRED EIGENVECTORS.
C     NOTE THAT FOR CASE (4) THE GENERALIZED LANCZOS RECURSION
C     B*V(I+1)*BETA(I+1) = A*V(I) - B*V(I)*ALPHA(I) - B*V(I-1)*BETA(I)
C     IS USED, ALONG WITH THE B-NORM.
C
C     THE IDEAS USED IN THESE PROGRAMS ARE DISCUSSED IN THE FOLLOWING
C     REFERENCES.
C
C     1.  JANE CULLUM AND RALPH A. WILLOUGHBY, LANCZOS ALGORITHMS
C         FOR LARGE SYMMETRIC MATRICES, VOLUME ?, PROGRESS IN
C         SCIENTIFIC COMPUTING, EDITORS, G. GOLUB, H.O. KREISS,
C         S. ARBARBANEL, AND R. GLOWINSKI,  BIRKHAUSER BOSTON INC.,
C         CAMBRIDGE, MASSACHUSETTS, 1983.
C
C     2.  JANE CULLUM AND RALPH A. WILLOUGHBY, COMPUTING EIGENVECTORS
C         (AND EIGENVALUES) OF LARGE, SYMMETRIC MATRICES USING
C         LANCZOS TRIDIAGONALIZATION, LECTURE NOTES IN MATHEMATICS,
C         773, NUMERICAL ANALYSIS PROCEEDINGS, DUNDEE 1979, EDITED BY
C         G. A. WATSON, SPRINGER-VERLAG, (1980), BERLIN, PP.46-63.
C
C     3. IBID, LANCZOS AND THE COMPUTATION IN SPECIFIED INTERVALS OF
C        THE SPECTRUM OF LARGE SPARSE, REAL SYMMETRIC MATRICES, SPARSE
C        MATRIX PROCEEDINGS 1978, ED. I.S. DUFF AND G. W. STEWART,
C        SIAM, PHILADELPHIA, PP.220-255, 1979.
C
C     4. IBID, COMPUTING EIGENVALUES OF VERY LARGE SYMMETRIC MATRICES-
C        AN IMPLEMENTATION OF A LANCZOS ALGORITHM WITHOUT
C        REORTHOGONALIZATION, J. COMPUT. PHYS. 44(1981), 329-358.
C
C
C-----PORTABILITY-----------------------------------------------------
C
C
C     PROGRAMS WERE TESTED FOR PORTABILITY USING THE PFORT VERIFIER.
C     FOR DETAILS OF THE VERIFIER SEE FOR EXAMPLE, B. G. RYDER AND
C     A. D. HALL, "THE PFORT VERIFIER", COMPUTING SCIENCE TECHNICAL
C     REPORT 12, BELL LABORATORIES, MURRAY HILL, NEW JERSEY 07974,
C     (REVISED), JANUARY 1981.
C
C     WITH THE EXCEPTION OF THE PROGRAMS FOR HERMITIAN MATRICES WHICH
C     ARE NOT PORTABLE BECAUSE OF THEIR USE OF COMPLEX*16 VARIABLES,
C     THE OTHER PROGRAMS INCLUDED ARE PORTABLE EXCEPT FOR A FEW
C     CONSTRUCTIONS WHICH, IF NECESSARY, WILL HAVE TO BE MODIFIED
```

```
C     BY THE USER FOR THE PARTICULAR COMPUTER BEING USED.               LEV01620
C                                                                       LEV01630
C     NONPORTABLE CONSTRUCTIONS:                                        LEV01640
C                                                                       LEV01650
C     REAL SYMMETRIC MATRICES:                                          LEV01660
C     IN LEVAL AND IN LEVEC                                             LEV01670
C         1.  DATA/MACHEP STATEMENT                                     LEV01680
C         2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                    LEV01690
C         3.  FORMAT(20A4) USED FOR THE EXPLANATORY HEADER ARRAY, EXPLANLEV01700
C         4.  FORMAT(4Z20) USED TO READ AND WRITE ALPHA/BETA FILES.     LEV01710
C     IN LEMULT                                                         LEV01720
C         1.  IN CMATV AND  USPEC THE ENTRY THAT PASSES THE STORAGE     LEV01730
C             LOCATIONS OF THE ARRAYS DEFINING THE USER-SPECIFIED       LEV01740
C             MATRIX.                                                   LEV01750
C         2.  IN THE SAMPLE USPEC PROVIDED:  FREE FORMAT (8,*),         LEV01760
C             THE FORMAT (20A4), AND DATA/MACHEP STATEMENT.             LEV01770
C                                                                       LEV01780
C     HERMITIAN MATRICES:                                               LEV01790
C     IN HLEVAL AND IN HLEVEC                                           LEV01800
C         1.  DATA/MACHEP STATEMENT                                     LEV01810
C         2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                    LEV01820
C         3.  FORMAT(20A4) USED FOR THE EXPLANATORY HEADER ARRAY, EXPLANLEV01830
C         4.  COMPLEX*16 VARIABLES AND FUNCTIONS SUCH AS DCMPLX.        LEV01840
C         5.  FORMAT (4Z20) USED TO READ AND WRITE ALPHA/BETA FILES.    LEV01850
C    IN HLEMULT                                                         LEV01860
C         1.  IN CMATV AND  USPEC THE ENTRY THAT PASSES THE STORAGE     LEV01870
C             LOCATIONS OF THE ARRAYS DEFINING THE USER-SPECIFIED       LEV01880
C             MATRIX.                                                   LEV01890
C         2.  COMPLEX*16 VARIABLES AND FUNCTIONS SUCH AS DCMPLX.        LEV01900
C         3.  IN THE SAMPLE USPEC PROVIDED:  FREE FORMAT (8,*),         LEV01910
C             THE FORMAT (20A4), AND DATA/MACHEP STATEMENT.             LEV01920
C                                                                       LEV01930
C    FACTORED INVERSES OF REAL SYMMETRIC MATRICES:                      LEV01940
C    IN LIVAL AND IN LIVEC                                              LEV01950
C         1.  DATA/MACHEP STATEMENT                                     LEV01960
C         2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                    LEV01970
C         3.  FORMAT(20A4) USED FOR THE EXPLANATORY HEADER ARRAY, EXPLANLEV01980
C         4.  FORMAT(4Z20) USED TO READ AND WRITE ALPHA/BETA FILES.     LEV01990
C    IN LIMULT                                                          LEV02000
C         1.  IN USPEC AND BSOLV, THE ENTRIES THAT PASS                 LEV02010
C             THE STORAGE LOCATIONS OF THE ARRAYS DEFINING THE          LEV02020
C             USER-SPECIFIED MATRIX.                                    LEV02030
C         2.  IN THE SAMPLE USPEC SUBROUTINES PROVIDED:                 LEV02040
C             FORMATS (20A4) AND (4Z20), FREE FORMAT (8,*), AND         LEV02050
C             DATA/MACHEP STATEMENTS.                                   LEV02060
C                                                                       LEV02070
C                                                                       LEV02080
C    GENERALIZED SYMMETRIC PROBLEM, B-MATRIX POSITIVE                   LEV02090
C    DEFINITE AND CHOLESKY FACTORS AVAILABLE:                           LEV02100
C    IN LGVAL AND IN LGVEC                                              LEV02110
C         1.  DATA/MACHEP STATEMENT                                     LEV02120
C         2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                    LEV02130
C         3.  FORMAT(20A4) USED FOR THE EXPLANATORY HEADER ARRAY, EXPLANLEV02140
C         4.  FORMAT(4Z20) USED TO READ AND WRITE ALPHA/BETA FILES.     LEV02150
C    IN LGMULT                                                          LEV02160
```

```
C        1.  IN USPECA, USPECB, AMATV AND LSOLV THE ENTRIES            LEV02170
C            THAT PASS THE STORAGE LOCATIONS OF THE ARRAYS DEFINING    LEV02180
C            THE USER-SPECIFIED MATRICES.                              LEV02190
C        2.  IN THE SAMPLE USPECA AND USPECB SUBROUTINES PROVIDED:     LEV02200
C            FORMATS (20A4) AND (4Z20), FREE FORMAT (8,*), AND         LEV02210
C            DATA/MACHEP STATEMENTS.                                   LEV02220
C                                                                      LEV02230
C     ALL 4 CASES USE THE FORTRAN FILE LESUB:                          LEV02240
C     IN LESUB ALL STATEMENTS ARE PORTABLE EXCEPT FOR:                 LEV02250
C         (1)  THE ENTRY IN SUBROUTINE LPERM THAT PASSES THE           LEV02260
C              PERMUTATION FROM THE USPEC SUBROUTINE TO LPERM.         LEV02270
C              (THIS IS USED ONLY IN CASES (3) AND (4)).               LEV02280
C         (2)  THE COMPLEX*16 VARIABLES AND FUNCTIONS USED IN          LEV02290
C              SUBROUTINE CINPRD. (THIS IS USED ONLY IN CASE (2)).     LEV02300
C                                                                      LEV02310
C     IN THE COMMENTS BELOW:                                           LEV02320
C                                                                      LEV02330
C     COMPLEX*16 = COMPLEX VARIABLE, 16 BYTES OF STORAGE               LEV02340
C     REAL*8 = REAL VARIABLE, 8 BYTES OF STORAGE                       LEV02350
C     REAL*4 = REAL VARIABLE, 4 BYTES OF STORAGE                       LEV02360
C     INTEGER*4 = INTEGER VARIABLE, 4 BYTES                            LEV02370
C                                                                      LEV02380
C                                                                      LEV02390
C-----MATRIX SPECIFICATION---------------------------------------------LEV02400
C                                                                      LEV02410
C                                                                      LEV02420
C     IN CASES (1) AND (2), SUBROUTINE USPEC IS USED TO SPECIFY THE    LEV02430
C     USER-SUPPLIED A-MATRIX.  SIMILARLY, IN CASE (4) SUBROUTINES      LEV02440
C     USPECA AND USPECB DEFINE THE USER-SUPPLIED A-MATRIX AND B-MATRIX. LEV02450
C     IN CASE (3) ((4)), SUBROUTINE USPECB DEFINES THE FACTORIZATION   LEV02460
C     OF THE MATRIX (B-MATRIX) USED BY THE LANCZOS PROCEDURE.          LEV02470
C     (IN CASE (3) THE A-MATRIX IS NOT USED DIRECTLY.)                 LEV02480
C                                                                      LEV02490
C     IN CASES (1) AND (2), SUBROUTINE CMATV IS A CORRESPONDING        LEV02500
C     MATRIX-VECTOR MULTIPLY SUBROUTINE WHICH SHOULD BE DESIGNED       LEV02510
C     TO TAKE ADVANTAGE OF ANY SPECIAL PROPERTIES OF THE GIVEN         LEV02520
C     MATRIX.  IN CASE (4) THIS SUBROUTINE IS NEEDED FOR THE           LEV02530
C     A-MATRIX AND THUS IS CALLED AMATV.  IN CASES (3) AND (4)         LEV02540
C     SUBROUTINES THAT CAN SOLVE B*U = V, USING A SPARSE               LEV02550
C     FACTORIZATION OF B ARE NEEDED.  THESE SUBROUTINES ARE            LEV02560
C     CALLED RESPECTIVELY, BSOLV AND LSOLV.  IN ALL CASES,             LEV02570
C     ANY MATRIX-VECTOR MULTIPLY AND SOLVE SUBROUTINES USED            LEV02580
C     MUST BE DESIGNED TO COMPUTE RAPIDLY AND ACCURATELY.              LEV02590
C                                                                      LEV02600
C     IN ALL CASES:                                                    LEV02610
C     SUBROUTINE USPEC(A OR B) HAS THE CALLING SEQUENCE                LEV02620
C                                                                      LEV02630
C            CALL USPEC(N,MATNO)                                       LEV02640
C                                                                      LEV02650
C     WHERE N IS THE ORDER OF THE USER-SUPPLIED MATRIX A, AND          LEV02660
C     MATNO IS A <= 8 DIGIT INTEGER USED AS A MATRIX AND               LEV02670
C     TEST IDENTIFICATION NUMBER.  IN ALL CASES THIS (THESE)           LEV02680
C     SUBROUTINE(S) DEFINES (DIMENSIONS) THE ARRAYS REQUIRED           LEV02690
C     TO SPECIFY THE MATRIX (MATRICES IN CASE (4)) THAT WILL BE        LEV02700
C     USED BY THE LANCZS SUBROUTINE.  IN CASES (1) AND (2)             LEV02710
```

```
C     THIS IS THE A-MATRIX; IN CASE (3) THIS IS THE FACTORIZATION
C     OF A SCALED, SHIFTED AND PERMUTED VERSION OF THE
C     USER-SPECIFIED A-MATRIX.  IN CASE (4) THE A-MATRIX
C     IS SPECIFIED AS WELL AS THE FACTORIZATION OF THE
C     B-MATRIX.  THIS SUBROUTINE ALSO INITIALIZES THE ARRAYS
C     AND ANY OTHER PARAMETERS NEEDED TO DEFINE THE MATRIX
C     (MATRICES).  THE STORAGE LOCATIONS OF THESE PARAMETERS
C     AND ARRAYS ARE THEN PASSED TO THE MATRIX-VECTOR MULTIPLY
C     SUBROUTINE CMATV IN CASES (1) AND (2), TO THE SUBROUTINE
C     BSOLV IN CASE (3), AND TO THE SUBROUTINES AMATV
C     AND LSOLV IN CASE (4) VIA ENTRY CALLS.  IN CASES (3) AND (4)
C     WHENEVER A MATRIX HAS BEEN PERMUTED, THERE IS ALSO AN
C     ENTRY INTO THE SUBROUTINE LPERM TO PASS THE LOCATIONS OF
C     THE PERMUTATIONS IPR AND IPRT USED.  SAMPLE USPECS, CMATV,
C     AMATV, BSOLV AND LSOLV SUBROUTINES ARE INCLUDED
C     IN THE RELEVANT FILES.  THESE SAMPLE PROGRAMS ASSUME THAT
C     THE USER-SUPPLIED A-MATRIX IS STORED ON FILE 8 IN CASES (1),
C     (2), AND (4), AND THAT THE FACTORIZATION OF THE B-MATRIX
C     IS ON FILE 7 IN CASES (3) AND (4).  THE USER SHOULD SEE
C     THE INDIVIDUAL SAMPLE SUBROUTINES FOR MORE DETAILS.
C
C     IN CASES (1) AND (2):
C     SUBROUTINE CMATV HAS THE CALLING SEQUENCE
C
C            CALL CMATV(W,U,SUM)
C
C     IN THE REAL SYMMETRIC CASE, U AND W ARE REAL*8 VECTORS
C     AND SUM IS A REAL*8 SCALAR.  IN THE HERMITIAN CASE, U
C     AND W ARE COMPLEX*16 VECTORS AND SUM IS A REAL*8 SCALAR.
C     CMATV CALCULATES U = A*W - SUM*U FOR THE USER-SPECIFIED
C     MATRIX A. ONE OF THE SAMPLE CMATV SUBROUTINES INCLUDED
C     COMPUTES MATRIX-VECTOR MULTIPLIES FOR AN ARBITRARY SPARSE,
C     SYMMETRIC MATRIX STORED IN THE SPARSE FORMAT SPECIFIED IN THE
C     CORRESPONDING SAMPLE USPEC SUBROUTINE.  FOR CASES (1) AND
C     (2) CMATV IS THE SUBROUTINE USED BY THE LANCZS SUBROUTINE
C     THAT GENERATES THE T-MATRICES.  IN CASE (4) SUBROUTINE
C     AMATV HAS THE SAME CALLING SEQUENCE AS CMATV IN CASE (1).
C
C     IN CASES (3) AND (4):
C     ALPHA/BETA HISTORY IS GENERATED USING SPARSE MATRIX INVERSION.
C     IN CASE (3), AT EACH ITERATION OF THE LANCZOS RECURSION
C     GIVEN A FACTORIZATION OF THE MATRIX BEING USED, THE
C     SUBROUTINE BSOLV FOR A GIVEN V, COMPUTES U SUCH THAT B*U = V.
C     THE CALLING SEQUENCE OF BSOLV IS
C
C                   CALL BSOLV(V,U,IBSOLV)
C
C     WHEN IBSOLV = 2,  U = (B-INVERSE)*V IS RETURNED.  IN CASE (4),
C     AT EACH ITERATION OF THE GENERALIZED LANCZOS RECURSION BOTH THE
C     SUBROUTINE AMATV AND THE SUBROUTINE LSOLV ARE USED.  THE
C     CALLING SEQUENCE OF LSOLV IS
C
C                   CALL LSOLV(V,U,ISOLV)
C
C     WHERE U AND V ARE REAL*8 VECTORS.  LSOLV PERFORMS 4 FUNCTIONS.
```

```
C     LET L DENOTE THE CHOLESKY FACTOR OF THE B-MATRIX USED IN LANCZS.
C     WHEN ISOLV = 1, LSOLV COMPUTES U = L*V.  WHEN ISOLV = 2,
C     LSOLV COMPUTES U = (L-TRANSPOSE)*V.  WHEN ISOLV = 3, LSOLV
C     COMPUTES U = (L-INVERSE)*V.  WHEN ISOLV = 4, LSOLV
C     COMPUTES  U = ((L-TRANSPOSE)-INVERSE)*V.
C
C     SAMPLE PROGRAMS ASSUME THAT THE A-MATRIX (CASES (1),(2),(4))
C     IS ON FILE 8 AND STORED IN THE FOLLOWING SPARSE FORMAT:
C     ICOL(K), K = 1,NZL, NUMBER OF SUBDIAGONAL NONZEROS IN COLUMN K.
C     IROW(K), K = 1,NZS, ROW INDEX OF ASD(K).
C     AD(K), K=1,N, CONTAINS THE DIAGONAL ELEMENTS OF THE A-MATRIX.
C     ASD(K), K=1,NZS  CONTAINS THE SUBDIAGONAL ELEMENTS OF A BY COLUMN.
C     NZS = NUMBER OF NONZERO ELEMENTS BELOW THE DIAGONAL OF A
C     NZL = INDEX OF LAST COLUMN WITH NONZERO SUBDIAGONAL ENTRIES
C     N = ORDER OF THE A-MATRIX.
C
C     NOTE THAT THE OPTIONAL PREPROCESSING PROGRAMS PERMUT AND
C     LORDER ASSUME THAT THE GIVEN MATRIX IS ON FILE 8.  CASES (3)
C     AND (4) ASSUME THAT THE SPARSE FACTORIZATION OF B IS STORED ON
C     FILE 7.  THE SAMPLE BSOLV SUBROUTINE SUPPLIED ASSUMES
C     THAT THE B-MATRIX IS POSITIVE DEFINITE AND THAT ITS CHOLESKY
C     FACTOR IS PROVIDED ON FILE 7, STORED IN SPARSE FORMAT IN
C     ARRAYS BD AND BSD.  THE USER CAN EASILY REPLACE THIS SAMPLE
C     BSOLV SUBROUTINE AND THE CORRESPONDING SAMPLE USPEC
C     SUBROUTINE BY SUBROUTINES THAT DEFINE AND USE A GENERAL
C     FACTORIZATION L*D*(L-TRANSPOSE).
C
C     THE SAMPLE USPEC, CMATV (CASES (1) AND (2)), AMATV (CASE (4)),
C     BSOLV (CASE (3)), AND LSOLV (CASE(4)) MUST BE MODIFIED BY
C     THE USER TO ACCOMODATE THE USER-SPECIFIED MATRIX OR MATRICES.
C
C
C-----MACHEP-------------------------------------------------------------
C
C
C     MACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING THE RELATIVE
C     PRECISION OF THE FLOATING POINT ARITHMETIC USED.
C     MACHEP = 2.2 * 10**-16 FOR DOUBLE PRECISION ARITHMETIC ON
C     IBM 370-3081.
C
C     THE USER WILL HAVE TO RESET THIS PARAMETER TO
C     THE CORRESPONDING VALUE FOR THE MACHINE BEING USED.  NOTE THAT
C     IF A MACHINE WITH A MACHINE EPSILON THAT IS MUCH LARGER THAN THE
C     VALUE GIVEN HERE IS BEING USED, THEN THERE COULD BE
C     PROBLEMS WITH THE TOLERANCES.
C
C
C-----SUBROUTINES AND FUNCTIONS USER MUST SUPPLY------------------------
C
C
C     GENRAN,  FINPRO,  MASK,  USPEC,  AND
C     CASES (1) AND (2),  CMATV:  CASE (3), BSOLV:
C     CASE (4),  AMATV AND LSOLV.
C
C     GENRAN = COMPUTES K PSEUDO-RANDOM NUMBERS AND STORES THEM IN
```

```
C              THE REAL*4 ARRAY, G.  THIS SUBROUTINE IS USED TO         LEV03820
C              GENERATE A STARTING VECTOR FOR THE LANCZOS PROCEDURE     LEV03830
C              IN THE SUBROUTINE LANCZS AND A STARTING RIGHT-HAND SIDE  LEV03840
C              FOR INVERSE ITERATION IN THE SUBROUTINE INVERR.          LEV03850
C                                                                       LEV03860
C              TESTS REPORTED IN THE REFERENCES USED EITHER GGL1 OR     LEV03870
C              GGL2 FROM THE IBM LIBRARY  SLMATH.                       LEV03880
C              THE EXISTING CALLING SEQUENCE IS:                        LEV03890
C                                                                       LEV03900
C                      CALL GENRAN(IIX,G,K).                            LEV03910
C                                                                       LEV03920
C              WHERE IIX =INTEGER SEED, G = REAL*4 ARRAY WHOSE          LEV03930
C              DIMENSION MUST BE >= K.  K RANDOM NUMBERS ARE GENERATED  LEV03940
C              AND PLACED IN G.                                         LEV03950
C                                                                       LEV03960
C     FINPRO = DOUBLE PRECISION FUNCTION WHICH COMPUTES THE INNER       LEV03970
C              PRODUCT OF 2 DOUBLE PRECISION VECTORS OF DIMENSION  N.   LEV03980
C              TESTS REPORTED IN THE REFERENCES USED THE HARWELL        LEV03990
C              LIBRARY SUBROUTINE FM02AD.                               LEV04000
C              EXISTING CALLING SEQUENCE IS                             LEV04010
C                                                                       LEV04020
C                      CALL FINPRO(N,V,J,W,K).                          LEV04030
C                                                                       LEV04040
C              COMPUTES THE INNER PRODUCT OF DIMENSION N OF THE VECTORS LEV04050
C              V AND W.  SUCCESSIVE COMPONENTS OF V AND OF W ARE STORED LEV04060
C              AT LOCATIONS THAT ARE ,RESPECTIVELY, J AND K UNITS APART.LEV04070
C                                                                       LEV04080
C     MASK = MASKS OVERFLOW AND UNDERFLOW.                              LEV04090
C            USER MUST SUPPLY OR COMMENT OUT CALL.                      LEV04100
C                                                                       LEV04110
C    USPEC = DIMENSIONS AND INITIALIZES ARRAYS NEEDED TO SPECIFY        LEV04120
C            MATRIX THAT WILL BE USED BY LANCZS SUBROUTINE.             LEV04130
C            IN CASE (4) A-MATRIX AND B-MATRIX MUST BOTH BE SPECIFIED.  LEV04140
C                                                                       LEV04150
C    CMATV = MATRIX-VECTOR MULTIPLY FOR USER-SUPPLIED MATRIX.           LEV04160
C            CASES (1) AND (2).  SEE MATRIX SPECIFICATION SECTION.      LEV04170
C                                                                       LEV04180
C    AMATV = MATRIX-VECTOR MULTIPLY FOR USER-SUPPLIED A-MATRIX.         LEV04190
C            CASES (4) ONLY.  SEE MATRIX SPECIFICATION SECTION.         LEV04200
C                                                                       LEV04210
C    BSOLV = GIVEN A VECTOR V COMPUTES U SUCH THAT B*U = V,              LEV04220
C            USING THE FACTORIZATION OF B.  USED IN CASE (3) ONLY.      LEV04230
C            SEE MATRIX SPECIFICATION SECTION.                          LEV04240
C                                                                       LEV04250
C    LSOLV = PERFORMS 4 FUNCTIONS.  GIVEN A VECTOR V COMPUTES           LEV04260
C            U = L*V, U = (L-TRANSPOSE)*V, U = (L-INVERSE)*V OR         LEV04270
C            U = (L-TRANSPOSE)-INVERSE*V, USING THE CHOLESKY            LEV04280
C            FACTORS OF B.  USED ONLY IN CASE (4).  SEE MATRIX          LEV04290
C            SPECIFICATION SECTION.                                     LEV04300
C                                                                       LEV04310
C                                                                       LEV04320
C-----------------------------------------------------------------------LEV04330
C                                                                       LEV04340
C     COMMENTS FOR EIGENVALUE COMPUTATIONS                              LEV04350
C                                                                       LEV04360
```

```
C-----------------------------------------------------------------------
C
C
C-----PARAMETER CONTROLS FOR EIGENVALUE PROGRAMS------------------------
C
C
C     PARAMETER CONTROLS ARE INTRODUCED TO ALLOW SEGMENTATION OF THE
C     EIGENVALUE COMPUTATIONS AND TO ALLOW VARIOUS COMBINATIONS OF
C     READ/WRITES.
C
C     THE FLAG ISTART CONTROLS THE T-MATRIX (ALPHA/BETA HISTORY)
C     GENERATION.
C
C     ISTART = (0,1)  MEANS
C
C              (0) THERE IS NO EXISTING ALPHA/BETA HISTORY AND ONE
C                  MUST BE GENERATED.
C
C              (1) THERE IS AN EXISTING ALPHA/BETA HISTORY AND IT IS
C                  TO BE READ IN FROM FILE 2 AND EXTENDED IF NECESSARY.
C
C     THE FLAG ISTOP CAN BE USED IN CONJUNCTION WITH THE FLAG ISTART TO
C     ALLOW SEGMENTATION OF THE EIGENVALUE COMPUTATIONS.
C
C     ISTOP  = (0,1)  MEANS
C
C              (0) PROGRAM COMPUTES ONLY THE REQUESTED ALPHAS/BETAS,
C                  STORES THEM AND THE LAST 2 LANCZOS VECTORS GENERATED
C                  IN FILE 1 AND THEN TERMINATES.  IN CASE (4) THERE
C                  ARE ACTUALLY 3 VECTORS TO BE SAVED.
C
C              (1) PROGRAM COMPUTES REQUESTED ALPHAS/BETAS AND THEN
C                  USES THE BISEC SUBROUTINE TO CALCULATE EIGENVALUES
C                  OF THE TRIDIAGONAL MATRICES GENERATED FOR THE ORDERS
C                  SPECIFIED BY THE USER AND ON THE USER-SPECIFIED
C                  INTERVALS.  PROGRAM THEN USES THE SUBROUTINE INVERR
C                  TO COMPUTE ERROR ESTIMATES FOR THE ISOLATED GOOD
C                  T-EIGENVALUES WHICH ARE USED TO CHECK THE
C                  CONVERGENCE OF THESE T-EIGENVALUES.
C
C     CONTROL PARAMETERS FOR WRITES
C
C     IHIS  = (0,1)  MEANS
C
C             (0) IF ISTOP .GT. 0 THEN ALPHA/BETAS ARE NOT SAVED ON
C                 FILE 1.
C
C             (1) PROGRAM WRITES ALPHAS/BETAS AND LAST 2 LANCZOS
C                 VECTORS TO FILE 1 SO THAT THE T-MATRIX GENERATION
C                 MAY BE REUSED OR CONTINUED LATER IF NECESSARY.
C                 TYPICALLY ONE WOULD ALWAYS DO THIS ON ANY RUN WHERE
C                 A HISTORY FILE IS BEING GENERATED.  HISTORY MUST BE
C                 SAVED IN MACHINE FORMAT ((4Z20) FOR IBM 3081) SO
C                 THAT NO ERRORS ARE INTRODUCED BY FORMAT CONVERSIONS.
C
```

```
C     IDIST  = (0,1)  MEANS                                             LEV04920
C                                                                       LEV04930
C              (0) DISTINCT EIGENVALUES OF T-MATRICES ARE NOT SAVED.    LEV04940
C                                                                       LEV04950
C              (1) PROGRAM WRITES COMPUTED DISTINCT EIGENVALUES OF      LEV04960
C                  T-MATRICES ALONG WITH THEIR T-MULTIPLICITIES         LEV04970
C                  TO FILE 11.                                          LEV04980
C                                                                       LEV04990
C     IWRITE = (0,1)   MEANS                                            LEV05000
C                                                                       LEV05010
C              (0) NO EXTENDED OUTPUT FROM SUBROUTINES BISEC AND INVERR LEV05020
C                  IS SENT TO FILE 6.                                   LEV05030
C                                                                       LEV05040
C              (1) INDIVIDUAL COMPUTED T-EIGENVALUES AND CORRESPONDING  LEV05050
C                  ERROR ESTIMATES FROM THE SUBROUTINES BISEC AND INVERRLEV05060
C                  ARE PRINTED OUT TO FILE 6 AS THEY ARE COMPUTED.      LEV05070
C                                                                       LEV05080
C     THE PROGRAM ALWAYS MAKES A SEPARATE LIST OF THE COMPUTED GOOD     LEV05090
C     T-EIGENVALUES ALONG WITH THEIR MINIMAL GAPS AND WRITES THEM OUT   LEV05100
C     TO FILE 3.  CORRESPONDING ERROR ESTIMATES FOR ANY ISOLATED        LEV05110
C     GOOD T-EIGENVALUES ARE ALWAYS WRITTEN TO FILE 4.                  LEV05120
C                                                                       LEV05130
C                                                                       LEV05140
C-----INPUT/OUTPUT FILES FOR EIGENVALUE PROGRAMS------------------------LEV05150
C                                                                       LEV05160
C     ANY INPUT DATA OTHER THAN THE ALPHA/BETA HISTORY SHOULD BE STORED LEV05170
C     ON FILE 5. SEE SAMPLE INPUT/OUTPUT FROM TYPICAL RUN.              LEV05180
C     THE READ STATEMENTS IN THE GIVEN FORTRAN PROGRAM ASSUME THAT      LEV05190
C     THE DATA STORED ON FILE 5 IS IN FREE FORMAT.  USER SHOULD NOTE    LEV05200
C     THAT 'FREE FORMAT' IS NOT CLASSIFIED AS PORTABLE BY PFORT SO THAT LEV05210
C     THE USER MAY HAVE TO MODIFY THE READ STATEMENTS FROM FILE 5 TO    LEV05220
C     CONFORM TO WHAT IS PERMISSIBLE ON THE MACHINE BEING USED.         LEV05230
C                                                                       LEV05240
C     FILE 6 WAS USED AS THE INTERACTIVE TERMINAL OUTPUT FILE.          LEV05250
C     THIS FILE PROVIDES A RUNNING ACCOUNT OF THE PROGRESS OF THE       LEV05260
C     COMPUTATIONS.  THE AMOUNT OF INFORMATION PRINTED OUT IS           LEV05270
C     CONTROLLED BY THE PARAMETER IWRITE.                               LEV05280
C                                                                       LEV05290
C DESCRIPTION OF OTHER I/O FILES                                        LEV05300
C                                                                       LEV05310
C FILE (K)     CONTAINS:                                                LEV05320
C                                                                       LEV05330
C      (1)     OUTPUT FILE:                                             LEV05340
C              HISTORY FILE OF NEWLY-GENERATED T-MATRIX (ALPHA AND      LEV05350
C              BETA VECTORS) AND LAST 2 LANCZOS VECTORS USED            LEV05360
C              IN THE T-MATRIX GENERATION.  NOTE THAT IN CASE (4)       LEV05370
C              THREE 'LANCZOS' VECTORS ARE WRITTEN TO FILE 1.           LEV05380
C              IF IHIS = 0 AND ISTOP = 1, FILE 1 IS NOT WRITTEN.        LEV05390
C                                                                       LEV05400
C      (2)     INPUT FILE:                                              LEV05410
C              SAME AS FILE 1 EXCEPT THAT IT CONTAINS A                 LEV05420
C              PREVIOUSLY-GENERATED T-MATRIX (IF ANY).  IF ISTART = 1,  LEV05430
C              PROGRAM ASSUMES THAT THERE IS A HISTORY FILE OF ALPHAS   LEV05440
C              AND BETAS ON FILE 2.  THESE ALPHAS AND BETAS ARE         LEV05450
C              READ IN ALONG WITH THE LAST 2 LANCZOS VECTORS THAT       LEV05460
```

```
C            WERE GENERATED.  IN CASE (4) THREE 'LANCZOS' VECTORS
C            ARE READ IN FROM FILE 2.
C
C     (3)    OUTPUT FILE:
C            COMPUTED GOOD EIGENVALUES OF THE T-MATRICES USED. ALSO
C            CONTAINS T-MULTIPLICITIES OF THESE EIGENVALUES AS
C            EIGENVALUES OF THE T-MATRIX, AND THEIR GAPS AS
C            EIGENVALUES IN THE A-MATRIX AND IN THE T-MATRIX.
C            FILE 3 IS ALWAYS WRITTEN.  IN CASE (3) THIS OUTPUT
C            CONTAINS THE EIGENVALUES OF THE B-INVERSE MATRIX
C            SINCE IN THIS CASE THE T-MATRICES CORRESPOND TO
C            THE B-INVERSE MATRIX AND NOT TO THE A-MATRIX.  IN
C            THIS CASE, 3 SETS OF GAPS ARE GIVEN, THOSE IN
C            THE T-MATRIX, IN THE B-INVERSE MATRIX AND THOSE
C            FOR THE CORRESPONDING EIGENVALUES IN THE A-MATRIX.
C
C     (4)    OUTPUT FILE:
C            ERROR ESTIMATES FOR THE ISOLATED GOOD T-EIGENVALUES
C            WHICH ARE OBTAINED USING THE SUBROUTINE INVERR. THESE
C            ESITMATES USE THE LAST COMPONENTS OF THE ASSOCIATED
C            T-EIGENVECTORS WHICH ARE COMPUTED USING INVERSE
C            ITERATION.  FILE 4 IS ALWAYS WRITTEN.
C
C
C     (7)    INPUT FILE:
C            USED ONLY IN CASES (3) AND (4), FACTORED INVERSES
C            OF REAL SYMMETRIC MATRICES AND GENERALIZED EIGENVALUE
C            PROBLEM.  CONTAINS THE REQUIRED FACTORIZATION OF THE
C            B-MATRIX.
C
C     (8)    INPUT FILE:
C            SAMPLE USPEC SUBROUTINE ASSUMES THAT THE ARRAYS
C            REQUIRED TO SPECIFY THE USER'S-MATRIX ARE STORED ON
C            FILE 8.  USERS MUST MAKE WHATEVER DEFINITIONS ARE
C            APPROPRIATE FOR THEIR MATRICES.  NOTE THAT IN CASE
C            (3) THE LANCZS SUBROUTINE DOES NOT USE THE MATRIX
C            ON FILE 8 IN THE T-MATRIX GENERATION, RATHER IT
C            USES THE FACTORIZATION OF AN ASSOCIATED
C            B-MATRIX WHICH IS STORED ON FILE 7.  IN CASE (4),
C            THE INFORMATION STORED ON BOTH FILES 7 AND 8 IS USED.
C
C     (9)    INPUT AND OUTPUT FILE:
C            CAN BE USED TO STORE THE TRUE EIGENVALUES OF THE
C            GIVEN PROBLEM,  WHEN THE PROCEDURE
C            IS BEING EXERCISED ON A TEST MATRIX.
C
C    (11)    OUTPUT FILE:
C            COMPUTED DISTINCT EIGENVALUES OF T-MATRICES USED.
C            ALSO CONTAINS THEIR T-MULTIPLICITIES AND T-GAPS TO
C            NEAREST DISTINCT EIGENVALUES, AND THE T-MULTIPLICITY
C            PATTERN OF THE GOOD AND THE SPURIOUS T-EIGENVALUES.
C            FILE 11 IS WRITTEN ONLY IF IDIST = 1.
C
C
C-----PARAMETERS SET BY THE EIGENVALUE PROGRAMS-----------------------
```

```
C
C
C     THESE PARAMETERS ARE SET INTERNALLY IN THE PROGRAM
C
C     SCALEK     K = 1,2,3,4
C
C                THE SCALING FACTORS SCALEK HAVE BEEN INTRODUCED IN AN
C                ATTEMPT TO MAKE THE TOLERANCES USED IN THE
C                T-MULTIPLICITY, SPURIOUS, ISOLATION AND PRTESTS ADJUST
C                TO THE SCALE OF THE GIVEN MATRIX.  THESE FACTORS MUST
C                NOT BE MODIFIED.  THESE TOLERANCES OCCUR IN LUMP,
C                ISOEV, AND PRTEST.
C
C     NOTE:    THE USER SHOULD NOTE THAT IF THE MATRIX BEING
C     PROCESSED IS VERY STIFF, THAT IS THE RATIO OF THE LARGEST
C     EIGENVALUE IN MAGNITUDE TO THE SMALLEST IN MAGNITUDE IS VERY
C     LARGE, THEN THE TOLERANCES BEING USED IN BISEC, LUMP, ISOEV
C     AND PRTEST MAY NOT TREAT THE SMALL END (SMALL IN MAGNITUDE)
C     VERY WELL.  IN SOME SUCH CASES A USER-INTRODUCED REDUCTION
C     IN THE SIZE OF TKMAX AND THE SUBSEQUENT RECOMPUTATION OF
C     THE T-MATRIX EIGENVALUES IN (ONLY) THE LOWER END OF THE
C     SPECTRUM WITH THIS TKMAX MAY RESULT IN IMPROVED COMPUTATIONS
C     AT THE LOW END.
C
C     THE LUMP, ISOEV, AND PRTEST TOLERANCES THAT WERE USED
C     MOST IN  THE TESTING OF THESE ALGORITHMS WERE NOT
C     SCALE INVARIANT BUT SEEMED TO WORK WELL ON MATRICES THAT
C     HAD EIGENVALUES WITH MAGNITUDES BOTH GREATER THAN AND LESS
C     THAN 1.  THESE TOLERANCES ARE ALSO INCLUDED IN THESE THREE
C     SUBROUTINES BUT AS COMMENTED OUT STATEMENTS.  THEY CAN BE
C     REVIVED BY COMMENTING OUT THE CORRESPONDING TOLERANCES
C     SPECIFIED IN THE STATEMENT ABOVE EACH OF THESE.
C
C     IMPORTANT TOLERANCES OR SCALES THAT ARE USED REPEATEDLY
C     THROUGHOUT THE LANCZOS EIGENVALUE PROGRAMS ARE THE FOLLOWING:
C     SCALED MACHINE EPSILON:  TTOL = TKMAX*EPSM WHERE
C     EPSM = 2*MACHINE EPSILON AND
C     TKMAX = MAX(|ALPHA(J)|,BETA(J), J = 1,MEV)
C     BISEC CONVERGENCE TOLERANCE:  BISTOL = DSQRT(1000+MEV)*TTOL
C     BISEC T-MULTIPLICITY TOLERANCE:  MULTOL = (1000+MEV)*TTOL
C     LANCZOS CONVERGENCE TOLERANCE:   CONTOL = BETA(MEV+1)*1.D-10
C
C
C     BTOL = RELATIVE TOLERANCE USED TO ESTIMATE ANY LOSS OF LOCAL
C            ORTHOGONALITY OF THE LANCZOS VECTORS AFTER THE T-MATRIX
C            HAS BEEN GENERATED.  THE LANCZOS PROCEDURE WORKS WELL
C            ONLY IF LOCAL ORTHOGONALITY BETWEEN SUCCESSIVE LANCZOS
C            VECTORS IS MAINTAINED.  THE TNORM SUBROUTINE TESTS
C            WHETHER OR NOT
C
C                 MINIMUM  |BETA(I)|/||A|| > BTOL.
C                 I=2,KMAX
C
C            IF THIS TEST IS VIOLATED BY SOME BETA AND A T-MATRIX THAT
C            WOULD INCLUDE SUCH A BETA IS REQUESTED, THEN THE LANCZOS
```

```
C           PROCEDURE WILL TERMINATE FOR THE USER TO DECIDE WHAT TO
C           DO.  THE USER CAN OVER-RIDE THIS TEST BY SIMPLY DECREASING
C           THE SIZE OF BTOL, BUT THEN CONVERGENCE IS NOT AS CERTAIN.
C           THE PROGRAM SETS BTOL = 1.D-8 WHICH IS A VERY CONSERVATIVE
C           CHOICE. THE || A || IS ESTIMATED BY USING AN ESTIMATE
C           OF THE NORM OF THE T-MATRIX, T(1,KMAX).
C
C     GAPTOL = RELATIVE TOLERANCE USED IN THE SUBROUTINE ISOEV
C             TO DETERMINE WHICH OF THE GOOD T-EIGENVALUES NEED
C             ERROR ESTIMATES.  THE PROGRAM SETS GAPTOL = 1.D-8.
C             IF FOR A GIVEN 'GOOD' T-EIGENVALUE THE COMPUTED GAP
C             IS TOO SMALL AND IS DUE TO A 'SPURIOUS' T-EIGENVALUE
C             THEN THE 'GOOD' T-EIGENVALUE IS ASSUMED TO HAVE CONVERGED
C             AND NO ERROR ESTIMATES ARE COMPUTED.
C
C
C-----USER-SPECIFIED PARAMETERS FOR EIGENVALUE PROGRAMS-----------------
C
C
C     RELTOL = RELATIVE TOLERANCE USED IN 'COMBINING' COMPUTED
C             EIGENVALUES OF T(1,MEV) PRIOR TO COMPUTING ERROR
C             ESTIMATES.
C
C     THE LUMPING OF T-EIGENVALUES OCCURS IN SUBROUTINE LUMP.
C     LUMPING IS NECESSARY BECAUSE IT IS IMPOSSIBLE TO ACCURATELY
C     PREDICT THE ACCURACY OF THE BISEC SUBROUTINE.  LUMP 'COMBINES'
C     T-EIGENVALUES THAT HAVE SLIPPED BY THE TOLERANCE THAT WAS USED
C     IN THE T-MULTIPLICITY TESTS.  IN PARTICULAR IF FOR SOME J,
C
C   |EVALUE(J)-EVALUE(J-1)| < DMAX1(RELTOL*|EVALUE(J)|,SCALE2*MULTOL)
C
C     THEN THESE T-EIGENVALUES ARE 'COMBINED'.  MULTOL IS THE TOLERANCE
C     THAT WAS USED IN THE T-MULTIPLICITY TEST IN BISEC.  SEE THE HEADER
C     ON THE LUMP SUBROUTINE FOR MORE DETAILS.
C
C     RELTOL IS SET TO 1.D-10.
C
C     MXINIT = MAXIMUM NUMBER OF INVERSE ITERATIONS ALLOWED IN
C             SUBROUTINE INVERR FOR EACH ISOLATED GOOD T-EIGENVALUE.
C             TYPICALLY ONLY ONE ITERATION IS REQUIRED.
C
C     SEEDS FOR RANDOM NUMBER GENERATORS = INTEGER*4 SCALARS.
C
C              (1) SVSEED = SEED FOR STARTING VECTOR USED IN
C                  T-MATRIX GENERATION IN LANCZS SUBROUTINE
C
C              (2) RHSEED = SEED FOR RIGHT-HAND SIDE USED IN
C                  INVERSE ITERATION COMPUTATIONS IN INVERR.
C
C     BISEC DATA
C
C     (1) NINT  =  NUMBER OF SUBINTERVALS ON WHICH EIGENVALUES ARE
C                 TO BE COMPUTED.
C
C     (2) LB(J) = (J = 1,NINT)  =  LEFT END POINTS OF THESE INTERVALS.
```

```
C              MUST BE PROVIDED IN INCREASING ORDER.  THAT IS,
C              LB(J) < LB(J+1) FOR J = 1,NINT.
C
C     (3) UB(J) = (J = 1,NINT) =  RIGHT END POINTS OF THESE INTERVALS.
C              MUST BE PROVIDED IN INCREASING ORDER.  THAT IS,
C              UB(J) < UB(J+1) FOR J = 1,NINT.
C
C     (4) MXSTUR  =  MAXIMUM NUMBER OF STURM ITERATIONS ALLOWED FOR
C                    ENTIRE SET OF EIGENVALUE CALCULATIONS OVER ALL
C                    SPECIFIED SIZE T-MATRICES.  PROGRAM WILL
C                    TERMINATE IF THIS LIMIT IS EXCEEDED.
C
C     T-MATRICES
C
C     SIZES OF T-MATRICES
C
C             (1) KMAX= MAXIMUM ORDER FOR T-MATRIX THAT USER IS WILLING
C                       TO CONSIDER.
C
C             (2) NMEVS = MAXIMUM NUMBER OF T-MATRICES THAT WILL BE
C                         CONSIDERED.
C
C             (3) NMEV(J)  (J=1,NMEVS)  = SIZES OF T-MATRIX TO BE
C                                         CONSIDERED SEQUENTIALLY.
C
C     T-MATRIX-GENERATION
C
C     USER SHOULD NOTE THAT THIS PROGRAM FIRST COMPUTES A T-MATRIX
C     OF ORDER KMAX AND THEN CYCLES THROUGH THE T-MATRICES SPECIFIED
C     A PRIORI BY THE USER, USING THE SUBROUTINE BISEC TO COMPUTE THE
C     EIGENVALUES OF THE T-MATRICES ON THE INTERVALS SPECIFIED BY
C     THE USER.
C
C     IDEALLY, ONE WOULD COMPUTE THE EIGENVALUE APPROXIMATIONS AT A
C     REASONABLE SIZE T-MATRIX, LOOK AT THE ACCURACY OF THE COMPUTED
C     RESULTS AND USE THAT TO DETERMINE AN APPROPRIATE
C     INCREMENT FOR THE SIZE OF THE T-MATRIX BASED UPON WHAT
C     HAS ALREADY CONVERGED AND UPON THE SIZES OF THE ERROR ESTIMATES
C     ON THOSE EIGENVALUES THAT ARE DESIRED BUT THAT HAVE NOT YET
C     CONVERGED. HOWEVER, IN THE INTERESTS OF GENERALITY AND
C     SIMPLICITY WE DID NOT DO THAT HERE.
C
C
C-----CONVERGENCE TESTS FOR THE EIGENVALUE PROGRAMS--------------------
C
C
C     THE CONVERGENCE TEST INCORPORATED IN THIS PROGRAM IS
C     BASED UPON THE ASSUMPTION THAT THOSE T-EIGENVALUES AND THEIR
C     ASSOCIATED T-EIGENVECTORS WHICH CORRESPOND TO THE
C     EIGENVALUES AND RITZVECTORS WHICH WE WISH TO COMPUTE
C     CONVERGE AS THE T-SIZE IS INCREASED.
C
C     AS CURRENTLY PROGRAMMED, CONVERGENCE IS CHECKED BY EXAMINING
C     THE SIZES OF ALL OF THE COMPUTED ERROR ESTIMATES ON ALL OF THE
C     INTERVALS SPECIFIED BY THE USER.  IDEALLY CONVERGENCE SHOULD
```

```
C     BE CHECKED ONLY ON THOSE EIGENVALUES OF INTEREST AND              LEV07670
C     ONCE THE EIGENVALUES ON SUB-INTERVALS OF THESE INTERVALS HAVE     LEV07680
C     CONVERGED, ANY SUBSEQUENT EIGENVALUE COMPUTATIONS SHOULD BE       LEV07690
C     MADE ONLY ON THE UNCONVERGED PORTIONS.  OBVIOUSLY, IT WOULD BE    LEV07700
C     DIFFICULT TO INCORPORATE CODE TO DO THE ABOVE WITHOUT KNOWING     LEV07710
C     A PRIORI PRECISELY WHAT THE USER IS TRYING TO COMPUTE.            LEV07720
C     THEREFORE, WE DID NOT ATTEMPT TO DO THIS.  IF ONE WISHES TO       LEV07730
C     MAKE SUCH A MODIFICATION THEN ONE MUST ALSO MODIFY THE PROGRAM    LEV07740
C     SO THAT IT CREATES AN OVERALL LIST OF THE CONVERGED 'GOOD'        LEV07750
C     T-EIGENVALUES AS THEY ARE COMPUTED, SINCE CONVERGED 'GOOD'        LEV07760
C     T-EIGENVALUES WHICH WERE COMPUTED AT A PARTICULAR VALUE OF MEV    LEV07770
C     WOULD NO LONGER BE RECOMPUTED AT LARGER VALUES OF MEV.            LEV07780
C                                                                       LEV07790
C     IF ONLY A FEW EIGENVALUES ARE TO BE COMPUTED THEN SUCH CHANGES    LEV07800
C     WOULD NOT MAKE MUCH DIFFERENCE IN THE RUNNING TIME.               LEV07810
C                                                                       LEV07820
C                                                                       LEV07830
C-----ARRAYS REQUIRED BY THE EIGENVALUE PROGRAMS------------------------LEV07840
C                                                                       LEV07850
C                                                                       LEV07860
C     ALL 4 CASES                                                       LEV07870
C                                                                       LEV07880
C     ALPHA(J) = REAL*8 ARRAY. ITS DIMENSION MUST BE AT LEAST KMAX,     LEV07890
C                THE LENGTH OF THE LARGEST T-MATRIX ALLOWED.  THIS      LEV07900
C                ARRAY CONTAINS THE DIAGONAL ENTRIES OF THE T-MATRICES. LEV07910
C                                                                       LEV07920
C     BETA(J) = REAL*8 ARRAY.  ITS DIMENSION MUST BE AT LEAST KMAX+1.   LEV07930
C               THIS ARRAY CONTAINS THE SUBDIAGONAL ENTRIES OF THE      LEV07940
C               T-MATRICES.                                             LEV07950
C                                                                       LEV07960
C               THE ALPHA AND BETA VECTORS ARE NOT ALTERED              LEV07970
C               DURING THE CALCULATIONS.                                LEV07980
C                                                                       LEV07990
C     V1(J),V2(J),VS(J) = REAL*8 ARRAYS IN REAL SYMMETRIC CASES.        LEV08000
C                         V1 AND V2 ARE COMPLEX*16 IN HERMITIAN CASE.   LEV08010
C                         IN CASES (1) AND (2) VS MUST BE OF            LEV08020
C                         DIMENSION AT LEAST KMAX.  IN CASES (3) AND    LEV08030
C                         (4) VS MUST BE OF DIMENSION AT LEAST          LEV08040
C                         MAX(N,KMAX).  IN REAL SYMMETRIC CASES         LEV08050
C                         V1 MUST BE OF DIMENSION AT LEAST              LEV08060
C                         MAX(KMAX+1,N) AND V2 MUST BE OF DIMENSION     LEV08070
C                         MAX(KMAX,N).  IN HERMITIAN CASES V1           LEV08080
C                         MUST BE OF DIMENSION MAX(N,(KMAX+1)/2)        LEV08090
C                         AND V2 OF DIMENSION AT LEAST MAX(N,KMAX/2).   LEV08100
C                         IN ALL CASES HOWEVER, THE ABOVE DIMENSIONS    LEV08110
C                         FOR V2 ARE VALID ONLY IF NO MORE              LEV08120
C                         THAN KMAX/2 EIGENVALUES OF THE GIVEN          LEV08130
C                         T-MATRICES ARE TO BE COMPUTED IN ANY GIVEN    LEV08140
C                         SUBINTERVAL. V2 IS USED IN THE SUBROUTINE     LEV08150
C                         BISEC TO HOLD THE UPPER AND LOWER             LEV08160
C                         ENDPOINTS OF THE SUBINTERVALS GENERATED       LEV08170
C                         DURING THE BISECTIONS.  THEREFORE, ITS        LEV08180
C                         REAL*8 DIMENSION MUST ALWAYS BE AT LEAST      LEV08190
C                         2*Q WHERE Q IS THE MAXIMUM NUMBER OF          LEV08200
C                         EIGENVALUES OF THE SPECIFIED T-MATRIX IN ANY  LEV08210
```

```
C                        ONE OF THE SPECIFIED INTERVALS.              LEV08220
C                        NOTE THAT IN THE HERMITIAN CASE, V1 AND V2   LEV08230
C                        ARE DEFINED AS COMPLEX*16 IN THE MAIN PROGRAM LEV08240
C                        AND IN THE LANCZS SUBROUTINE BUT ARE         LEV08250
C                        REDEFINED AS REAL*8 IN OTHER SUBROUTINES.    LEV08260
C                                                                     LEV08270
C     LB(J),UB(J) = REAL*8 ARRAYS. EACH MUST BE OF DIMENSION AT LEAST LEV08280
C                  NINT, THE NUMBER OF SUBINTERVALS TO BE CONSIDERED. LEV08290
C                  LB CONTAINS THE LEFT-END POINTS OF THE INTERVALS   LEV08300
C                  ON WHICH EIGENVALUES ARE TO BE COMPUTED.  UB       LEV08310
C                  CONTAINS THE RIGHT-END POINTS.                     LEV08320
C                                                                     LEV08330
C     EXPLAN(J) = REAL*4 ARRAY.  ITS DIMENSION IS 20.  THIS ARRAY IS  LEV08340
C                USED TO ALLOW EXPLANATORY COMMENTS IN THE INPUT FILES.LEV08350
C                                                                     LEV08360
C     G(J) = REAL*4 ARRAY.  ITS DIMENSION MUST BE >= MAX(2*KMAX,N)    LEV08370
C           IT IS USED FOR HOLDING THE RANDOM VECTORS GENERATED,      LEV08380
C           HOLDING THE COMPUTED ERROR ESTIMATES AND THE COMPUTED     LEV08390
C           MINIMAL GAPS FOR THE GOOD T-EIGENVALUES.                  LEV08400
C                                                                     LEV08410
C     MP(J) = INTEGER*4 ARRAY.  ITS DIMENSION MUST BE AT LEAST KMAX,  LEV08420
C            THE MAXIMUM SIZE OF THE T-MATRICES ALLOWED.  IT CONTAINS LEV08430
C            THE T-MULTIPLICITIES OF THE COMPUTED EIGENVALUES. NOTE   LEV08440
C            THAT 'SPURIOUS' T-EIGENVALUES ARE DENOTED BY A           LEV08450
C            T-MULTIPLICITY OF 0.  T-EIGENVALUES THAT THE SUBROUTINE  LEV08460
C            PRTEST HAS IDENTIFIED AS 'GOOD' BUT HIDDEN ARE IDENTIFIED LEV08470
C            BY A T-MULTIPLICITY OF -10.                              LEV08480
C                                                                     LEV08490
C     NMEV(J) = INTEGER*4 ARRAY.  ITS DIMENSION MUST BE AT LEAST THE  LEV08500
C              NUMBER OF T-MATRICES ALLOWED.  IT CONTAINS THE ORDERS  LEV08510
C              OF THE T-MATRICES TO BE CONSIDERED.                    LEV08520
C                                                                     LEV08530
C                                                                     LEV08540
C     FOR CASE (3) ONLY:                                              LEV08550
C     GR(J),GC(J) = REAL*8 ARRAYS. USED ONLY IN THE HERMITIAN CASE.   LEV08560
C                  GR AND GC MUST EACH BE OF DIMENSION AT LEAST N.    LEV08570
C                  BOTH ARE USED IN THE RANDOM VECTOR GENERATION.     LEV08580
C                  GR IS ALSO USED TO STORE MINIMAL GAPS BETWEEN      LEV08590
C                  'GOOD' T-EIGENVALUES.                              LEV08600
C                                                                     LEV08610
C     FOR CASES (3) AND (4) FOR THE PERMUTATION:                      LEV08620
C                                                                     LEV08630
C     IPR(J), IPT(J) = INTEGER*4 ARRAYS.  EACH OF DIMENSION AT LEAST N. LEV08640
C                     USED TO STORE THE REORDERING OF THE GIVEN MATRIX LEV08650
C                     OR MATRICES.                                    LEV08660
C                                                                     LEV08670
C                                                                     LEV08680
C     OTHER ARRAYS                                                    LEV08690
C                                                                     LEV08700
C     THE USER MUST SPECIFY IN THE SUBROUTINE USPEC (A OR B) WHATEVER LEV08710
C     ARRAYS ARE REQUIRED TO DEFINE THE MATRIX OR MATRICES BEING USED. LEV08720
C     ALSO IN CASES (3) AND (4) ONLY, WHEN WORKING WITH INVERSES      LEV08730
C     OF SPARSE SYMMETRIC MATRICES, IN THE OPTIONAL, PREPROCESSING    LEV08740
C     PROGRAMS PERMUT, LFACT, LORDER, AND LTEST IT IS NECESSARY TO    LEV08750
C     SPECIFY ADDITIONAL ARRAYS JUST FOR THESE COMPUTATIONS.  THE USER LEV08760
```

```
C     IS REFERRED TO THOSE PROGRAMS FOR DETAILS.
C
C
C-----SUBROUTINES INCLUDED-------------------------------------------
C
C
C     LANCZS = COMPUTES THE ALPHA/BETA HISTORY. IN CASES (1) AND (2)
C              REAL SYMMETRIC AND HERMITIAN MATRICES, USES SUBROUTINES
C              FINPRO, GENRAN AND CMATV.  IN CASE (3), INVERSES OF
C              REAL SYMMETRIC MATRICES, USES SUBROUTINES FINPRO,
C              GENRAN AND BSOLV.  IN CASE (4), GENERALIZED EIGENVALUE
C              PROBLEM, USES SUBROUTINES FINPRO, GENRAN, AMATV AND
C              LSOLV.
C
C     BISEC = COMPUTES EIGENVALUES OF THE SPECIFIED T-MATRIX
C             USING STURM SEQUENCING, ON SEQUENCE OF INTERVALS
C             SPECIFIED BY THE USER.  EACH SUBINTERVAL IS TREATED
C             AS OPEN ON THE LEFT AND CLOSED ON THE RIGHT.
C             EIGENVALUES ARE COMPUTED WITH SIMULTANEOUS DETERMINATION
C             OF THE T-MULTIPLICITIES AND OF SPURIOUS T-EIGENVALUES.
C
C     INVERR = USES INVERSE ITERATION ON T-MATRICES TO COMPUTE ERROR
C              ESTIMATES ON COMPUTED GOOD T-EIGENVALUES. (USES GENRAN)
C
C     LUMP = 'COMBINES' EIGENVALUES OF T-MATRIX USING THE RELATIVE
C             TOLERANCE RELTOL.
C
C     ISOEV = CALCULATES GAPS BETWEEN DISTINCT EIGENVALUES OF T-MATRIX
C             AND THEN USES THESE GAPS TO LABEL THOSE 'GOOD'
C             T-EIGENVALUES FOR WHICH ERROR ESTIMATES ARE NOT COMPUTED.
C
C     TNORM = COMPUTES THE SCALE TKMAX USED IN DETERMINING THE
C             TOLERANCES FOR THE SPURIOUS, T-MULTIPLICITY AND PRTESTS.
C             IT ALSO CHECKS FOR LOCAL ORTHOGONALITY OF THE LANCZOS
C             VECTORS BY TESTING THE RELATIVE SIZE OF THE BETAS USING
C             THE RELATIVE TOLERANCE BTOL.
C
C     PRTEST = LOOKS FOR GOOD T-EIGENVALUES THAT HAVE BEEN MISLABELLED
C              BY THE SPURIOUS TEST BECAUSE THEY HAD 'TOO SMALL' A
C              PROJECTION ON THE STARTING LANCZOS VECTOR.
C              (LESS THAN SINGLE PRECISION)
C              TESTS INDICATE THAT SUCH EIGENVALUES ARE RARE.
C              PRTEST SHOULD BE CALLED ONLY AFTER CONVERGENCE
C              HAS BEEN ESTABLISHED.
C
C     INVERM = USED TO COMPUTE ERROR ESTIMATES FOR ANY T-EIGENVALUES
C              WHICH PRTEST INDICATES MAY HAVE BEEN MISLABELLED.
C              SUCH T-EIGENVALUES ARE RELABELLED ONLY IF THEIR ERROR
C              ESTIMATES ARE SUFFICIENTLY SMALL.  PRIMARY USE OF
C              INVERM IS IN THE CORRESPONDING EIGENVECTOR COMPUTATIONS.
C
C     CASES (3) AND (4) ONLY, FACTORED INVERSES:
C
C     FOR OPTIONAL, PRELIMINARY PROCESSING:
C     PERMUT  (PROGRAM CALLS SPARSPAK PACKAGE) :
```

```
C     USES THE NONZERO STRUCTURE OF A GIVEN MATRIX A.                   LEV09320
C     CAN BE USED TO OBTAIN A REORDERING OF A THAT PRESERVES            LEV09330
C     SPARSITY UNDER FACTORIZATION.  PERMUT CALLS                       LEV09340
C     THE SPARSPAK PROGRAMS, (A. GEORGE, J. LIU, E.NG,                  LEV09350
C     U. WATERLOO).  PERMUT ALSO TAKES THE USER-SPECIFIED MATRIX,       LEV09360
C     APPLIES THE SCALE S0 AND THE SHIFT TO IT, AND THEN WRITES         LEV09370
C     OUT THE CORRESPONDING SPARSE MATRIX DATA FILE FOR THE             LEV09380
C     RESULTING MATRIX C = S0*A + SHIFT*I.  SEE THE PERMUT FORTRAN      LEV09390
C     CODE FOR DETAILS.                                                 LEV09400
C                                                                       LEV09410
C     LORDER (STAND-ALONE PROGRAM):                                     LEV09420
C     GIVEN A MATRIX C IN SPARSE FORMAT AND A PERMUTATION P,            LEV09430
C     COMPUTES THE REORDERED MATRIX B = P*C*P' AND WRITES IT            LEV09440
C     TO FILE 9 IN SPARSE FORMAT.  SEE THE LORDER FORTRAN CODE          LEV09450
C     FOR DETAILS.                                                      LEV09460
C                                                                       LEV09470
C     LFACT (STAND-ALONE PROGRAM) :                                     LEV09480
C     GIVEN A POSITIVE DEFINITE MATRIX B IN SPARSE FORMAT               LEV09490
C     COMPUTES THE SPARSE CHOLESKY FACTOR L OF B AND WRITES IT          LEV09500
C     TO FILE 7 IN SPARSE FORMAT.  THUS, B = L*L'.                      LEV09510
C     SEE THE LFACT FORTRAN CODE FOR DETAILS.                           LEV09520
C                                                                       LEV09530
C     LTEST (CALLS 3 USER-SUPPLIED PROGRAMS CMATV, CMATS, AND BSOLV):   LEV09540
C     GIVEN THE FACTORIZATION OF A MATRIX B, LTEST COMPUTES             LEV09550
C     THE SOLUTION OF THE EQUATION B*U = B*V1 FOR A SPECIFIC RANDOMLY-  LEV09560
C     GENERATED V1, WITH AND WITHOUT ITERATIVE REFINEMENT, TO           LEV09570
C     OBTAIN A ROUGH CHECK ON THE NUMERICAL CONDITION OF THE MATRIX B.  LEV09580
C     THIS PROGRAM USES 3 SUBROUTINES CMATV, CMATS, AND BLSOLV.         LEV09590
C     SEE THE LTEST FORTRAN PROGRAM FOR DETAILS.                        LEV09600
C                                                                       LEV09610
C                                                                       LEV09620
C-----OTHER PROGRAMS PROVIDED-------------------------------------------LEV09630
C                                                                       LEV09640
C     LECOMPAC (STAND ALONE PROGRAM):                                   LEV09650
C                  TRANSLATES A REAL SYMMETRIC MATRIX PROVIDED IN THE   LEV09660
C                  FORMAT  I, J, A(I,J) INTO THE SPARSE MATRIX          LEV09670
C                  FORMAT USED IN THE SAMPLE USPEC, CMATV, BSOLV AND    LEV09680
C                  LSOLV SUBROUTINES PROVIDED. IT ASSUMES THAT THE      LEV09690
C                  MATRIX ENTRIES ARE GIVEN EITHER COLUMN BY COLUMN OR  LEV09700
C                  ROW BY ROW.  THE DATA SET CREATED IS WRITTEN TO      LEV09710
C                  FILE 8.                                              LEV09720
C                                                                       LEV09730
C                                                                       LEV09740
C-----COMMENTS ON THE STORAGE REQUIRED FOR EIGENVALUE PROGRAMS----------LEV09750
C                                                                       LEV09760
C                                                                       LEV09770
C     CASE (1), REAL SYMMETRIC MATRICES:                                LEV09780
C                                                                       LEV09790
C     THE ARRAYS IN THE REAL SYMMETRIC EIGENVALUE PROGRAM REQUIRE       LEV09800
C     APPROXIMATELY THE EQUIVALENT OF ONE REAL*8 ARRAY OF DIMENSION     LEV09810
C                                                                       LEV09820
C         3.5*KMAX + 2*MAX(KMAX,N) + .5* MAX(2*KMAX,N)                  LEV09830
C                                                                       LEV09840
C     PLUS WHATEVER IS NEEDED TO GENERATE A*X FOR THE GIVEN MATRIX A.   LEV09850
C     THE ARRAYS ALPHA, BETA, VS AND MP CONSUME 3.5*KMAX*8 BYTES.       LEV09860
```

```
C     THE ARRAYS V1 AND V2 CONSUME 2*MAXIMUM(KMAX,N)*8 BYTES, WITH THE
C     QUALIFICATION STATED ABOVE WHERE V2 IS DEFINED.  THE G-ARRAY
C     CONSUMES .5*MAX(2*KMAX,N)*8 BYTES.
C
C     CASE (2), HERMITIAN MATRICES:
C
C     THE ARRAYS IN THE HERMITIAN EIGENVALUE PROGRAMS REQUIRE
C     THE EQUIVALENT OF ONE REAL*8 ARRAY OF DIMENSION
C
C         3.5*KMAX + 4*MAX(KMAX/2,N) + .5*MAX(2*KMAX,N) + 2*N
C
C     PLUS WHATEVER IS NEEDED TO GENERATE A*X FOR THE GIVEN MATRIX A.
C     THE ARRAYS ALPHA, BETA, VS, AND MP CONSUME 3.5*KMAX*8 BYTES.
C     THE ARRAYS V1 AND V2 CONSUME 4*MAXIMUM(KMAX/2,N)*8 BYTES, WITH THE
C     QUALIFICATION STATED ABOVE WHERE V2 IS DEFINED.  THE G-ARRAY
C     CONSUMES .5*MAX(2*KMAX,N)*8 BYTES.  GR REQUIRES
C     AND GC REQUIRE 2*N*8BYTES.
C
C
C     CASE (3), INVERSES OF REAL SYMMETRIC MATRICES:
C
C     THE ARRAYS IN THE EIGENVALUE PROGRAMS DESIGNED FOR
C     CASE (3), INVERSES OF REAL SYMMETRIC MATRICES USING
C     REORDERING AND FACTORIZATION, REQUIRE
C     THE EQUIVALENT OF ONE REAL*8 ARRAY OF DIMENSION
C
C         3*KMAX + 3*MAX(KMAX,N) + .5*MAX(2*KMAX,N)
C
C     PLUS WHATEVER IS NEEDED TO GENERATE B(INVERSE)*X  FOR THE
C     SCALED, SHIFTED AND PERMUTED VERSION OF A WHICH WE DENOTE
C     BY B.  THE ARRAYS ALPHA, BETA, MP, AND MP2 CONSUME 3*KMAX*8
C     BYTES.  THE ARRAYS V1, V2, AND VS CONSUME 3*MAX(KMAX,N)*8 BYTES,
C     WITH THE QUALIFICATION STATED ABOVE WHERE V2 IS DEFINED.
C     THE G ARRAY CONSUMES .5*MAX(2*KMAX,N)*8 BYTES. THESE NUMBERS
C     DO NOT INCLUDE THE STORAGE REQUIRED BY THE PREPROCESSING PROGRAMS
C     PERMUT, LORDER, LFACT, AND LTEST.
C
C
C     A SYMMETRIC, SPARSE MATRIX OF ORDER N WITH NZS NONZERO ELEMENTS
C     BELOW THE MAIN DIAGONAL WOULD REQUIRE THE EQUIVALENT OF ONE
C     REAL*8 ARRAY OF DIMENSION 1.5*(NZS + N) IF THE POINTERS USED
C     ARE INTEGER*4.
C
C     SOME OF THE ARRAY STORAGE IS NOT ESSENTIAL AND COULD BE
C     ELIMINATED IF STORAGE IS A PROBLEM.
C     THE FOLLOWING COMMENTS APPLY DIRECTLY ONLY TO CASE (1),
C     THE PROGRAMS FOR REAL SYMMETRIC MATRICES, HOWEVER, SIMILAR
C     STATEMENTS COULD BE MADE ABOUT THE OTHER CASES.
C
C     CASE (1), REAL SYMMETRIC PROGRAMS:
C     THE G ARRAY COULD BE REMOVED IF THE USER IS WILLING TO
C
C              (1) REGENERATE THE RANDOM STARTING VECTOR IN INVERR
C                  FOR EACH ERROR ESTIMATE
C              (2) WRITE OUT THE ERROR ESTIMATES AND VARIOUS GAPS AS
```

```
C                THEY ARE GENERATED RATHER THAN STORING THEM IN G FOR
C                LATER PRINTOUT
C            (3) CHECK CONVERGENCE WITHIN INVERR
C
C     CLEARLY THE INDEX VECTOR MP COULD BE AN INTEGER*2 ARRAY AS COULD
C     THE POINTERS USED TO DEFINE THE USER'S MATRIX.
C
C     THE USER SHOULD NOTE THAT WITH AN EIGENVALUE SUBROUTINE THAT
C     USES BISECTION (LIKE BISEC) IF MORE THAN 25% OF THE
C     EIGENVALUES ARE TO BE COMPUTED, THEN IT MAY BE MORE
C     ECONOMICAL TO USE THE EISPACK SUBROUTINE IMTQL1.
C     (SEE MATRIX EIGENSYTEM ROUTINES-EISPACK GUIDE (2ND EDITION)
C     B.T. SMITH ET AL, SPRINGER-VERLAG, NEW YORK, 1976, P213.).
C     HOWEVER, IF THE SUBROUTINE IMTQL1 IS TO BE USED IN PLACE
C     OF BISEC, THEN NONTRIVIAL CHANGES IN THE LANCZOS CODE MUST BE
C     MADE. FOR DETAILS OF ONE SUCH IMPLEMENTATION SEE
C     IBM RESEARCH REPORT 8298, COMPUTING
C     EIGENVALUES OF LARGE SYMMETRIC MATRICES - AN IMPLEMENTATION OF A
C     LANCZOS ALGORITHM WITH NO REORTHOGONALIZATION.  PART II. COMPUTER
C     PROGRAMS., DECEMBER 1980, WHICH CONTAINS A GENERAL
C     LANCZOS CODE WHICH INCLUDES AN IMTQL1 OPTION OR
C     PREFERABLY CONTACT THE AUTHORS.
C
C     THE BISEC SUBROUTINE WHICH IS INCLUDED IS A MODIFIED FORM OF
C     THE BISECT SUBROUTINE IN EISPACK. BISEC ASSUMES THAT THE
C     VECTOR V2 IS LONG ENOUGH TO HOLD BOTH THE UPPER AND THE
C     LOWER BOUNDS ON THE BISECTION INTERVALS USED TO COMPUTE
C     THE EIGENVALUES OF THE T-MATRICES.  THEREFORE, IF THE
C     LENGTH OF V2 IS ONLY KMAX, BISEC CAN COMPUTE ONLY AT MOST
C     KMAX/2 EIGENVALUES OF THE GIVEN T-MATRIX IN ANY GIVEN
C     SUBINTERVAL.
C
C     AS PROGRAMMED BISEC USES THE ARRAYS ALPHA,BETA,V1,V2,VS AND MP.
C     HOWEVER, V1 IS USED ONLY TO STORE BETA(J)**2 SO THAT THEY DO NOT
C     HAVE TO BE REGENERATED ON EACH STURM.  IF THE USER IS WILLING TO
C     COMPUTE THE BETA(J)**2 AS NEEDED, THEN V1 COULD BE ELIMINATED
C     FROM BISEC. BISEC STORAGE THEN BECOMES A REAL*8 ARRAY OF DIMENSION
C     4.25*KMAX IF WE ALSO REDUCE MP TO INTEGER*2.  FURTHERMORE,
C     IF ONE KNEW THAT ONLY Q*MEV EIGENVALUES OF T(1,MEV) WERE TO BE
C     COMPUTED AT EACH STAGE FOR SOME Q<.5 THEN FURTHER REDUCTIONS IN
C     STORAGE COULD BE MADE IN BISEC.
C
C     AS PROGRAMMED INVERR USES ALPHA, BETA,V1,V2,VS,G AND MP.
C     VS CONTAINS THE COMPUTED EIGENVALUES OF T(1,MEV). MP GIVES
C     THEIR T-MULTIPLICITIES AND FLAGS WHICH EIGENVALUES ARE TO HAVE
C     ERROR ESTIMATES COMPUTED.  V2 IS USED FOR THE SOLUTION
C     VECTOR IN THE INVERSE ITERATION AND V1 FOR THE FACTORIZATION.
C     G CONTAINS THE RANDOMLY-GENERATED STARTING VECTOR FOR THE
C     INVERSE ITERATION. THE BASIC STORAGE FOR INVERR IS THEREFORE
C     A REAL*8 ARRAY OF DIMENSION 4*KMAX PLUS THE STORAGE NEEDED FOR
C     THE COMPUTED T-EIGENVALUES AND THEIR T-MULTIPLICITIES.
C
C     VS COULD BE USED TO STORE ONLY THOSE COMPUTED EIGENVALUES OF
C     T(1,MEV) THAT ARE OF INTEREST. IN THAT CASE THE DIMENSIONS OF VS
C     AND OF MP NEED NOT BE ANY LONGER THAN THE NUMBER OF SUCH
```

```
C     EIGENVALUES. AS PROGRAMMED, ALL THE COMPUTED DISTINCT EIGENVALUES
C     OF T(1,MEV) ARE STORED IN VS.  THEREFORE TO TAKE ADVANTAGE OF
C     SUCH A REDUCTION IN STORAGE THE USER WOULD HAVE TO MODIFY THAT
C     PART OF THE PROGRAM AND ALSO COMMENT OUT THE CALL TO THE
C     SUBROUTINE PRTEST.
C
C
C-----------------------------------------------------------------------
C
C     COMMENTS FOR EIGENVECTOR COMPUTATIONS
C
C-----------------------------------------------------------------------
C
C
C     THE EIGENVALUES WHOSE EIGENVECTORS ARE TO BE COMPUTED MUST
C     HAVE BEEN COMPUTED USING THE CORRESPONDING LANCZOS EIGENVALUE
C     PROGRAMS BECAUSE THE EIGENVECTOR PROGRAMS WILL USE THE SAME
C     FAMILY OF LANCZOS TRIDIAGONAL MATRICES THAT WAS USED IN THE
C     CORRESPONDING EIGENVALUE COMPUTATIONS.
C
C     THESE PROGRAMS ASSUME THAT THE EIGENVALUES SUPPLIED TO IT
C     HAVE BEEN COMPUTED ACCURATELY, AS MEASURED BY THE
C     ERROR ESTIMATES COMPUTED IN THE CORRESPONDING LANCZOS
C     EIGENVALUE COMPUTATIONS, ALTHOUGH THESE ESTIMATES ARE
C     TYPICALLY CONSERVATIVE.  IN CASES (1), (2) AND (4), THE
C     EIGENVALUES OF INTEREST ARE STORED IN THE ARRAY GOODEV(J),
C     J=1,NGOOD.  IN CASE (3) THE PROGRAM WORKS WITH THE
C     EIGENVALUES OF B(INVERSE) WHICH ARE STORED IN THE ARRAY
C     GOODBI(J), J=1,NGOOD.  THE CORRESPONDING EIGENVALUES
C     OF A ARE STORED IN GOODA(J), J=1,NGOOD.
C
C     FOR EACH GOODEV(J), THE SUBROUTINE STURMI COMPUTES THE
C     SMALLEST SIZE LANCZOS TRIDIAGONAL MATRIX, T(1,M1(J)), FOR
C     WHICH  GOODEV(J) IS AN EIGENVALUE TO WITHIN A SPECIFIED
C     TOLERANCE.  IT ALSO ATTEMPTS TO COMPUTE THE SIZE, M2(J),
C     BY WHICH THE GIVEN EIGENVALUE BECOMES A DOUBLE EIGENVALUE
C     TO WITHIN THE GIVEN TOLERANCE.  THESE VALUES ARE USED
C     TO DETERMINE 1ST GUESSES AT SIZES FOR THE T-EIGENVECTORS
C     THAT WILL BE USED IN THE RITZ VECTOR COMPUTATIONS.
C     SUBROUTINE INVERM SUCCESSIVELY COMPUTES CORRESPONDING
C     EIGENVECTORS OF ENLARGED T-MATRICES UNTIL A SUITABLE
C     SIZE T-MATRIX IS DETERMINED FOR EACH J.  UP TO 10 SUCH
C     EIGENVECTOR COMPUTATIONS ARE ALLOWED FOR EACH EIGENVALUE.
C
C     AFTER APPROPRIATE T-EIGENVECTORS HAVE BEEN COMPUTED,
C     RITZ VECTOR CORRESPONDING TO THESE T-EIGENVECTORS ARE THEN
C     COMPUTED AND TAKEN AS APPROXIMATE EIGENVECTORS OF A FOR THE
C     GIVEN EIGENVALUES, GOODEV(J), J = 1, ..., NGOOD.
C
C     THIS IMPLEMENTATION FIRST COMPUTES ALL OF THE RELEVANT
C     EIGENVECTORS OF THE SYMMETRIC TRIDIAGONAL MATRICES
C     IN THE VECTOR, TVEC.
C
C     THEN, AS EACH OF THE LANCZOS VECTORS IS REGENERATED, ALL
C     OF THE RITZ VECTORS CORRESPONDING TO THESE
```

```
C     T-EIGENVECTORS ARE UPDATED USING THE CURRENTLY-GENERATED          LEV11520
C     LANCZOS VECTOR.  LANCZOS VECTORS ARE GENERATED (NOTE              LEV11530
C     THAT THEY ARE NOT BEING KEPT), UNTIL ENOUGH HAVE                  LEV11540
C     BEEN GENERATED TO MAP THE LONGEST T-EIGENVECTOR INTO ITS          LEV11550
C     CORRESPONDING RITZ VECTOR.  THE ARRAY RITVEC CONTAINS THE         LEV11560
C     SUCCESSIVE RITZ VECTORS WHICH ARE THE APPROXIMATE                 LEV11570
C     EIGENVECTORS OF A.                                                LEV11580
C                                                                       LEV11590
C                                                                       LEV11600
C-----PARAMETER CONTROLS FOR EIGENVECTOR PROGRAMS-----------------------LEV11610
C                                                                       LEV11620
C                                                                       LEV11630
C     IN CASES (3) AND (4) WHERE A SPARSE FACTORIZATION OF A            LEV11640
C     SPECIFIED MATRIX IS USED, THE USER SPECIFIES USING THE FLAG       LEV11650
C     JPERM WHETHER OR NOT THE FACTORIZATION SUPPLIED CORRESPONDS       LEV11660
C     TO THE ORIGINAL MATRIX OR TO A PERMUTATION OF THE ORIGINAL        LEV11670
C     MATRIX.                                                           LEV11680
C                                                                       LEV11690
C     JPERM = (0,1) MEANS                                               LEV11700
C              0    NO PERMUTATION                                      LEV11710
C              1    MATRIX HAS BEEN PERMUTED.  NOTE THAT IN             LEV11720
C                   CASE (4) THE PROGRAM WILL ASSUME THAT THE           LEV11730
C                   DATA SUPPLIED FOR THE A-MATRIX CORRESPONDS TO THE   LEV11740
C                   CORRESPONDING PERMUTATION OF THE ORIGINAL A-MATRIX. LEV11750
C                   IN BOTH CASES THE LANCZS CODES WILL WORK WITH THE   LEV11760
C                   PERMUTED MATRICES AND THE PERMUTATION WILL BE       LEV11770
C                   UNDONE ONLY IN THE EIGENVECTOR PROGRAM AFTER        LEV11780
C                   THE RITZ VECTORS FOR THE PERMUTED PROBLEM HAVE      LEV11790
C                   BEEN COMPUTED.                                      LEV11800
C                                                                       LEV11810
C     OTHER PARAMETER CONTROLS ARE INTRODUCED TO ALLOW SEGMENTATION     LEV11820
C     OF THE EIGENVECTOR COMPUTATIONS AND TO ALLOW VARIOUS COMBINATIONS LEV11830
C     OF READ/WRITES.                                                   LEV11840
C                                                                       LEV11850
C     THE FLAG MBOUND ALLOWS THE USER TO DETERMINE A FIRST GUESS ON THE LEV11860
C     STORAGE THAT WILL BE REQUIRED BY THE T-EIGENVECTORS FOR THE       LEV11870
C     EIGENVALUES WHOSE EIGENVECTORS ARE TO BE COMPUTED.                LEV11880
C     THIS CAN BE USED TO ESTIMATE THE REQUIRED SIZE OF THE TVEC ARRAY. LEV11890
C                                                                       LEV11900
C     MBOUND = (0,1) MEANS                                              LEV11910
C                                                                       LEV11920
C              (0)  PROGRAM COMPUTES FIRST GUESSES AT THE SIZES         LEV11930
C                   OF THE T-MATRICES REQUIRED BY EACH OF THE           LEV11940
C                   EIGENVALUES SUPPLIED AND THEN CONTINUES WITH        LEV11950
C                   THE CORRESPONDING T-EIGENVECTOR COMPUTATIONS.       LEV11960
C                                                                       LEV11970
C              (1)  PROGRAM COMPUTES FIRST GUESSES AT THE SIZES         LEV11980
C                   OF THE T-MATRICES REQUIRED BY EACH OF THE           LEV11990
C                   EIGENVALUES SUPPLIED, STORES THESE IN FILE 10       LEV12000
C                   AND THEN TERMINATES.  THE USER CAN USE THESE        LEV12010
C                   SIZES TO ESTIMATE THE SIZE TVEC ARRAY NEEDED        LEV12020
C                   FOR THE DESIRED T-EIGENVECTOR COMPUTATIONS.         LEV12030
C                                                                       LEV12040
C     THE FLAGS NTVCON, TVSTOP, LVCONT, AND ERCONT CONTROL THE STOPPING LEV12050
C     CRITERIA FOR INTERMEDIATE POINTS IN THE LANCZOS PROCEDURE.  THEY  LEV12060
```

```
C     TERMINATE THE PROCEDURE IF VARIOUS QUANTITIES COULD NOT BE        LEV12070
C     COMPUTED AS DESIRED.                                              LEV12080
C                                                                       LEV12090
C     NTVCON = (0,1) MEANS                                              LEV12100
C                                                                       LEV12110
C               (0)  IF THE ESTIMATED STORAGE FOR THE T-EIGENVECTORS    LEV12120
C                    EXCEEDS THE USER-SPECIFIED DIMENSION OF THE        LEV12130
C                    TVEC ARRAY PROGRAM DOES NOT CONTINUE WITH THE      LEV12140
C                    T-EIGENVECTOR COMPUTATIONS.  TERMINATION OCCURS.   LEV12150
C                                                                       LEV12160
C               (1)  CONTINUE WITH THE T-EIGENVECTOR COMPUTATIONS       LEV12170
C                    EVEN IF THE ESTIMATED STORAGE FOR TVEC EXCEEDS     LEV12180
C                    THE USER-SPECIFIED DIMENSION OF THE TVEC ARRAY.    LEV12190
C                    IN THIS SITUATION THE PROGRAM COMPUTES AS MANY     LEV12200
C                    T-EIGENVECTORS AS IT HAS ROOM FOR, IN THE SAME     LEV12210
C                    ORDER IN WHICH THE EIGENVALUES ARE PROVIDED.       LEV12220
C                                                                       LEV12230
C      SVTVEC = (0,1) MEANS                                             LEV12240
C                                                                       LEV12250
C               (0)  DO NOT STORE THE COMPUTED T-EIGENVECTORS ON        LEV12260
C                    FILE 11 UNLESS ALSO HAVE THE FLAG TVSTOP = 1,      LEV12270
C                    IN WHICH CASE THE T-EIGENVECTORS ARE ALWAYS        LEV12280
C                    WRITTEN TO FILE 11.                                LEV12290
C                                                                       LEV12300
C               (1)  STORE THE COMPUTED T-EIGENVECTORS ON FILE 11.      LEV12310
C                                                                       LEV12320
C      TVSTOP = (0,1) MEANS                                             LEV12330
C                                                                       LEV12340
C               (0)  ATTEMPT TO CONTINUE ON TO THE COMPUTATION          LEV12350
C                    OF THE RITZVECTORS AFTER COMPLETING THE            LEV12360
C                    COMPUTATION OF THE T-EIGENVECTORS.                 LEV12370
C                                                                       LEV12380
C               (1)  TERMINATE AFTER COMPUTING THE                      LEV12390
C                    T-EIGENVECTORS AND STORING THEM ON FILE 11.        LEV12400
C                                                                       LEV12410
C      LVCONT = (0,1) MEANS                                             LEV12420
C                                                                       LEV12430
C               (0)  IF SOME OF THE T-EIGENVECTORS THAT WERE            LEV12440
C                    REQUESTED WERE NOT COMPUTED, EXIT                  LEV12450
C                    FROM THE PROGRAM WITHOUT COMPUTING THE             LEV12460
C                    CORRESPONDING RITZ VECTORS.                        LEV12470
C                                                                       LEV12480
C               (1)  CONTINUE ON TO THE RITZ VECTOR COMPUTATIONS        LEV12490
C                    EVEN IF NOT ALL OF THE T-EIGENVECTORS              LEV12500
C                    REQUESTED WERE COMPUTED.                           LEV12510
C                                                                       LEV12520
C      ERCONT = (0,1) MEANS                                             LEV12530
C                                                                       LEV12540
C               (0)  PROCEDURE WILL NOT COMPUTE A RITZ VECTOR FOR       LEV12550
C                    ANY EIGENVALUE FOR WHICH NO T-EIGENVECTOR WHICH    LEV12560
C                    SATISFIES THE ERROR ESTIMATE TEST (ERTOL) HAS      LEV12570
C                    BEEN IDENTIFIED.                                   LEV12580
C                                                                       LEV12590
C               (1)  A RITZ VECTOR WILL BE COMPUTED FOR EVERY           LEV12600
C                    EIGENVALUE FOR WHICH A T-EIGENVECTOR HAS BEEN      LEV12610
```

```
C                    COMPUTED REGARDLESS OF WHETHER OR NOT THAT         LEV12620
C                    T-EIGENVECTOR SATISFIED THE ERROR ESTIMATE TEST.   LEV12630
C                                                                       LEV12640
C                                                                       LEV12650
C-----INPUT/OUTPUT FILES FOR THE EIGENVECTOR COMPUTATIONS---------------LEV12660
C                                                                       LEV12670
C                                                                       LEV12680
C     ANY INPUT DATA OTHER THAN THE T-MATRIX HISTORY FILE AND THE       LEV12690
C     PREVIOUSLY COMPUTED EIGENVALUES AND CORRESPONDING ERROR           LEV12700
C     ESTIMATES SHOULD BE STORED ON FILE 5 IN FREE FORMAT.              LEV12710
C     SEE SAMPLE INPUT/OUTPUT FOR TYPICAL INPUT FILE.                   LEV12720
C                                                                       LEV12730
C     FILE 6 WAS USED AS THE INTERACTIVE TERMINAL OUTPUT FILE.          LEV12740
C     THIS FILE PROVIDES A RUNNING ACCOUNT OF THE PROGRESS OF THE       LEV12750
C     COMPUTATIONS.  ADDITIONAL PRINTOUT IS GENERATED WHEN              LEV12760
C     THE FLAG IWRITE = 1.                                              LEV12770
C                                                                       LEV12780
C                                                                       LEV12790
C DESCRIPTION OF OTHER I/O FILES                                        LEV12800
C                                                                       LEV12810
C FILE (K)     CONTAINS:                                                LEV12820
C                                                                       LEV12830
C     (2)      INPUT FILE:                                              LEV12840
C              PREVIOUSLY-GENERATED T-MATRICES (ALPHA/BETA ARRAYS)      LEV12850
C              AND THE FINAL TWO LANCZOS VECTORS USED ON THAT           LEV12860
C              COMPUTATION.  THIS PROGRAM ALLOWS ENLARGEMENT            LEV12870
C              OF ANY T-MATRICES PROVIDED ON FILE 2.  NOTE THAT         LEV12880
C              IN CASE (4), THREE 'LANCZOS' VECTORS ARE ON FILE 2.      LEV12890
C                                                                       LEV12900
C     (3)      INPUT FILE:                                              LEV12910
C              THE GOOD EIGENVALUES OF THE T-MATRIX  T(1,MEV)           LEV12920
C              FOR WHICH RITZ VECTORS ARE REQUESTED.                    LEV12930
C              FILE 3 ALSO CONTAINS THE T-MULTIPLICITIES OF THESE       LEV12940
C              EIGENVALUES AND THEIR COMPUTED GAPS IN THE               LEV12950
C              T-MATRICES AND IN THE USER-SUPPLIED MATRIX. IN           LEV12960
C              CASE (3) FILE 3 CONTAINS THE EIGENVALUES OF THE          LEV12970
C              B-INVERSE MATRIX AND 3 SETS OF CORRESPONDING GAPS.       LEV12980
C              THIS FILE IS CREATED IN THE LANCZOS EIGENVALUE           LEV12990
C              COMPUTATIONS.                                            LEV13000
C                                                                       LEV13010
C     (4)      INPUT FILE:                                              LEV13020
C              ERROR ESTIMATES FOR THE ISOLATED GOOD T-EIGENVALUES      LEV13030
C              ON FILE 3.  THIS FILE IS CREATED DURING THE LANCZOS      LEV13040
C              EIGENVALUE COMPUTATIONS.                                 LEV13050
C                                                                       LEV13060
C     (7)      INPUT FILE:                                              LEV13070
C              IN CASE (3) ((4)),                                       LEV13080
C              CONTAINS SPARSE MATRIX REPRESENTATION OF FACTORIZATION   LEV13090
C              OF MATRIX (B-MATRIX) USED BY LANCZS SUBROUTINE.          LEV13100
C                                                                       LEV13110
C     (8)      INPUT FILE:                                              LEV13120
C              IN CASES (1),(2) AND (4), USPEC SUBROUTINE ASSUMES       LEV13130
C              THAT USER-SUPPLIED A-MATRIX IS ON FILE 8.  IN CASE (3)   LEV13140
C              A-MATRIX CAN BE STORED ON FILE 8, BUT IT IS NOT          LEV13150
C              USED BY THE LANCZOS PROGRAMS.                            LEV13160
```

```
C
C     (9)     OUTPUT FILE:
C             ERROR ESTIMATES FOR THE COMPUTED RITZ VECTORS CONSIDERED
C             AS EIGENVECTORS OF THE MATRIX USED BY THE LANCZS
C             SUBROUTINE.  THESE ESTIMATES ARE OF THE FORM
C                   AERROR = || A*RITVEC - EVAL*RITVEC ||
C             WHERE A DENOTES THE MATRIX USED BY LANCZS, EVAL DENOTES
C             THE EIGENVALUE BEING CONSIDERED AND RITVEC DENOTES
C             THE COMPUTED RITZ VECTOR.
C
C     (10)    OUTPUT FILE:
C             GUESSES AT APPROPRIATE SIZE T-MATRICES FOR THE
C             T-EIGENVECTORS FOR EACH SUPPLIED EIGENVALUE, GOODEV(J).
C
C     (11)    OUTPUT FILE:
C             COMPUTED T-EIGENVECTORS CORRESPONDING TO EIGENVALUES
C             IN THE GOODEV ARRAY.  NOTE THAT IT IS POSSIBLE IN
C             CERTAIN SITUATIONS THAT FOR SOME EIGENVALUES IN THE
C             GOODEV ARRAY A T-EIGENVECTOR WILL NOT BE COMPUTED.
C
C     (12)    OUTPUT FILE:
C             CONTAINS COMPUTED RITZ VECTORS CORRESPONDING TO
C             THE T-EIGENVECTORS ON FILE 11. NOTE THAT IN
C             SOME SITUATIONS THAT FOR SOME EIGENVALUES IN
C             THE GOODEV ARRAY FOR WHICH T-EIGENVECTORS HAVE
C             BEEN COMPUTED NO RITZ VECTOR WILL HAVE BEEN
C             COMPUTED.
C
C     (13)    OUTPUT FILE:
C             ADDITIONAL INFORMATION ABOUT THE BOUNDS AND ERROR
C             ESTIMATES OBTAINED.
C
C
C-----SEEDS FOR EIGENVECTOR PROGRAMS-----------------------------------
C
C     SEEDS FOR RANDOM NUMBER GENERATOR GENRAN
C               (1) SVSEED = INTEGER*4 SCALAR USED IN THE SUBROUTINE
C                            GENRAN TO GENERATE THE STARTING VECTOR FOR
C                            THE REGENERATION OF THE LANCZOS VECTORS.
C
C               (2) RHSEED = INTEGER*4 SCALAR USED IN THE SUBROUTINE
C                            GENRAN TO GENERATE A RANDOM VECTOR FOR
C                            USE IN SUBROUTINE INVERM.
C
C     USER SHOULD NOTE THAT SVSEED MUST BE THE SAME SEED THAT
C     WAS USED TO GENERATE THE T-MATRICES THAT WERE USED TO
C     COMPUTE THE EIGENVALUES WHOSE EIGENVECTORS ARE TO BE COMPUTED.
C     SVSEED IS READ IN FROM FILE 3.
C
C
C-----USER-SPECIFIED PARAMETERS FOR THE EIGENVECTOR PROGRAMS------------
C
C
C     NGOOD   = NUMBER OF EIGENVALUES READ INTO THE GOODEV ARRAY
C                READ FROM FILE 3.
```

```
C
C     N       = SIZE OF THE USER-SUPPLIED MATRIX.
C
C     MEV     = SIZE OF THE T-MATRIX THAT WAS USED TO COMPUTE
C                 THE EIGENVALUES WHOSE EIGENVECTORS ARE REQUESTED.
C                 MEV IS READ IN FROM FILE 3.
C
C     KMAX    =  SIZE OF THE T-MATRIX PROVIDED ON FILE 2.
C
C     MDIMTV  = MAXIMUM CUMULATIVE SIZE OF THE TVEC ARRAY ALLOWED
C                 FOR ALL OF THE T-EIGENVECTORS REQUIRED.  MDIMTV
C                 MUST NOT EXCEED THE USER-SPECIFIED DIMENSION OF
C                 THE TVEC ARRAY.  PROGRAM CAN BE RUN WITH THE FLAG
C                 MBOUND = 1 TO DETERMINE AN EDUCATED GUESS ON AN
C                 APPROPRIATE DIMENSION FOR THE TVEC ARRAY.
C
C     MDIMRV = MAXIMUM CUMULATIVE SIZE OF THE RITVEC ARRAY ALLOWED
C                FOR ALL OF THE RITZ VECTORS TO BE COMPUTED. MDIMRV
C                MUST NOT EXCEED THE USER-SPECIFIED DIMENSION OF
C                THE RITVEC ARRAY.  MUST BE SELECTED SO THAT
C                THERE IS ENOUGH ROOM FOR A RITZ VECTOR FOR EVERY
C                GOODEV(J) READ INTO PROGRAM.  (>= NGOOD*N)
C
C
C-----ARRAYS REQUIRED BY THE EIGENVECTOR PROGRAMS---------------------
C
C
C     ALL 4 CASES
C
C     ALPHA(J) = REAL*8 ARRAY.  ITS DIMENSION MUST BE AT LEAST
C                  KMAXN, THE LARGEST SIZE T-MATRIX CONSIDERED BY
C                  THE PROGRAM.  NOTE THAT KMAXN IS THE LARGER OF
C                  THE SIZE OF THE ALPHA, BETA HISTORY PROVIDED
C                  ON FILE 2 (IF ANY ) AND THE SIZE WHICH THE PROGRAM
C                  SPECIFIES INTERNALLY, THIS LATTER IS ALWAYS
C                  < = 11*MEV / 8  +  12, WHERE MEV IS THE SIZE
C                  T-MATRIX THAT WAS USED IN THE CORRESPONDING EIGENVALUE
C                  COMPUTATIONS.  ALPHA CONTAINS THE DIAGONAL ENTRIES
C                  OF THE LANCZOS T-MATRICES.  ALPHA IS NOT DESTROYED
C                  IN THE COMPUTATIONS.
C
C     BETA(J) = REAL*8 ARRAY.  ITS DIMENSION MUST BE AT LEAST 1
C                 MORE THAN THAT OF ALPHA.  DIMENSION COMMENTS ABOVE
C                 ABOUT ALPHA APPLY ALSO TO THE BETA ARRAY.  BETA
C                 CONTAINS THE SUBDIAGONAL ENTRIES OF THE T-MATRICES.
C                 BETA IS NOT DESTROYED IN THE COMPUTATIONS.
C
C     RITVEC(J) = REAL*8 ARRAY IN REAL SYMMETRIC AND INVERSE OF
C                   REAL SYMMETRIC CASES.  COMPLEX*16 IN CASE (2),
C                   HERMITIAN MATRICES.  IN EACH CASE ITS DIMENSION >=
C                   NGOOD*N WHERE N IS THE ORDER OF THE USER-SUPPLIED
C                   MATRIX AND NGOOD IS THE NUMBER OF EIGENVALUES WHOSE
C                   EIGENVECTORS ARE TO BE COMPUTED.  IT CONTAINS THE
C                   COMPUTED APPROXIMATE EIGENVECTORS OF A.  THESE
C                   COMPUTED RITZ VECTORS ARE STORED ON FILE 12.
```

```
C                                                                       LEV14270
C     TVEC(J)   = REAL*8 ARRAY.  ITS DIMENSION MUST BE AT LEAST         LEV14280
C                 MTOL = |MA(1)| + |MA(2)| + ... + |MA(NGOOD)|          LEV14290
C                 WHERE NGOOD IS THE NUMBER OF EIGENVALUES BEING        LEV14300
C                 CONSIDERED AND |MA(J)| IS THE SIZE OF THE             LEV14310
C                 T-MATRIX BEING USED FOR THE RITZ VECTOR               LEV14320
C                 COMPUTATION FOR GOODEV(J).  THESE SIZES               LEV14330
C                 ARE COMPUTED BY THE PROGRAM.  AN ESTIMATE OF          LEV14340
C                 MTOL CAN BE OBTAINED BY SETTING MBOUND = 1,           LEV14350
C                 RUNNING THE PROGRAM, AND MULTIPLYING THE RESULTING    LEV14360
C                 TOTAL OF THE T-SIZES SPECIFIED BY 5/4.  THE ARRAY     LEV14370
C                 TVEC CONTAINS THE COMPUTED T-EIGENVECTORS.  IF THE    LEV14380
C                 FLAG SVTVEC = 1 OR THE FLAG TVSTOP = 1, THEN          LEV14390
C                 THESE VECTORS ARE SAVED ON FILE 11.                   LEV14400
C                                                                       LEV14410
C     V1(J)     = REAL*8 ARRAY IN REAL SYMMETRIC AND INVERSE OF         LEV14420
C                 REAL SYMMETRIC CASES.  COMPLEX*16 IN CASE (2),        LEV14430
C                 HERMITIAN MATRICES.  IN THE REAL CASES ITS            LEV14440
C                 DIMENSION MUST BE THE MAXIMUM OF KMAX AND N.          LEV14450
C                 IN THE HERMITIAN CASE ITS DIMENSION MUST BE           LEV14460
C                 THE MAXIMUM OF KMAX/2 AND N WHERE KMAX IS THE         LEV14470
C                 LARGEST SIZE T-MATRIX THAT IS TO BE CONSIDERED        LEV14480
C                 IN THE T-EIGENVECTOR COMPUTATIONS.  V1 IS USED        LEV14490
C                 IN THE SUBROUTINE INVERM AND IN THE REGENERATION      LEV14500
C                 OF THE LANCZOS VECTORS.                               LEV14510
C                                                                       LEV14520
C     V2(J)     = REAL*8 ARRAY IN THE REAL SYMMETRIC AND INVERSE        LEV14530
C                 OF REAL SYMMETRIC CASES.  COMPLEX*16 IN CASE (2),     LEV14540
C                 HERMITIAN MATRICES.  IN CASES (1),(3) AND (4), ITS    LEV14550
C                 DIMENSION MUST BE > = MAX(KMAX,N); IN CASE (2)        LEV14560
C                 > = MAX(KMAX/2,N).  IT IS USED IN THE REGENERATION    LEV14570
C                 OF THE LANCZOS VECTORS AND IN SUBROUTINE INVERM.      LEV14580
C                                                                       LEV14590
C     GOODEV(J),  = REAL*8 ARRAYS EACH OF DIMENSION AT LEAST NGOOD.     LEV14600
C     EVNEW(J)      CONTAIN THE EIGENVALUES FOR WHICH EIGENVECTORS      LEV14610
C                   ARE REQUESTED.  EIGENVALUES IN GOODEV ARE READ      LEV14620
C                   IN FROM FILE 3.  IN CASE (3) GOODEV IS REPLACED     LEV14630
C                   BY GOODA AND GOODBI ARRAYS, SEE BELOW.              LEV14640
C                                                                       LEV14650
C     AMINGP(J), = REAL*4 ARRAYS OF DIMENSION AT LEAST NGOOD.           LEV14660
C     TMINGP(J)    CONTAIN, RESPECTIVELY, THE MINIMAL GAPS FOR          LEV14670
C                  CORRESPONDING EIGENVALUES IN GOODEV ARRAY IN         LEV14680
C                  A-MATRIX AND IN T-MATRIX.                            LEV14690
C                                                                       LEV14700
C     TERR(J), ERR(J),     = REAL*4 ARRAYS (EXCEPT TLAST WHICH IS       LEV14710
C     ERRDGP(J), TLAST(J)    REAL*8).  EACH MUST BE OF DIMENSION        LEV14720
C     RNORM(J), TBETA(J)     AT LEAST NGOOD.  USED TO STORE QUANTITIES  LEV14730
C                            GENERATED DURING THE COMPUTATIONS FOR      LEV14740
C                            LATER PRINTOUT.                            LEV14750
C                                                                       LEV14760
C     G(J)     = REAL*4 ARRAY WHOSE DIMENSION MUST BE AT LEAST          LEV14770
C                MAX(KMAX,N).  USED IN SUBROUTINE GENRAN TO HOLD        LEV14780
C                RANDOM NUMBERS NEEDED FOR THE LANCZOS VECTOR           LEV14790
C                REGENERATION AND FOR THE INVERSE ITERATION             LEV14800
C                COMPUTATIONS IN THE SUBROUTINE INVERM.                 LEV14810
```

```
C                                                                       LEV14820
C     MP(J) = INTEGER*4 ARRAY WHOSE DIMENSION IS AT LEAST NGOOD.        LEV14830
C             INITIALLY CONTAINS THE MULTIPLICITY OF THE EIGENVALUE     LEV14840
C             GOODEV(J) AS AN EIGENVALUE OF THE T-MATRIX T(1,MEV).      LEV14850
C             USED TO FLAG EIGENVALUES FOR WHICH NO T-EIGENVECTOR       LEV14860
C             OR NO RITZ VECTOR IS TO BE COMPUTED.                      LEV14870
C                                                                       LEV14880
C     MA(J)   = INTEGER*4 ARRAYS EACH OF WHOSE DIMENSIONS               LEV14890
C               IS AT LEAST NGOOD.  USED IN DETERMINING                 LEV14900
C               AN APPROPRIATE T-MATRIX FOR EACH EIGENVALUE             LEV14910
C               IN GOODEV ARRAY.                                        LEV14920
C                                                                       LEV14930
C     MINT(J),MFIN(J) = INTEGER*4 ARRAYS WHOSE DIMENSIONS MUST BE AT    LEV14940
C                       LEAST NGOOD.  USED TO POINT TO THE BEGINNINGS   LEV14950
C                       AND THE ENDS OF THE COMPUTED EIGENVECTOR        LEV14960
C                       OF THE T-MATRIX, T(1,|MA(J)|).                  LEV14970
C                                                                       LEV14980
C     IDELTA(J)  = INTEGER*4 ARRAY WHOSE DIMENSION MUST BE AT           LEV14990
C                  LEAST NGOOD.  CONTAINS INCREMENTS USED IN LOOPS      LEV15000
C                  ON APPROPRIATE SIZE T-MATRIX FOR THE T-EIGENVECTOR   LEV15010
C                  COMPUTATIONS.                                        LEV15020
C                                                                       LEV15030
C                                                                       LEV15040
C     CASE (2) ONLY, HERMITIAN MATRICES:                                LEV15050
C                                                                       LEV15060
C     GR(J),GC(J)     = REAL*8 ARRAYS USED ONLY IN CASE (2),            LEV15070
C                       HERMITIAN MATRICES.  EACH MUST BE AT            LEV15080
C                       LEAST MAX(N,KMAX).  USED TO HOLD                LEV15090
C                       STARTING VECTORS FOR LANCZS                     LEV15100
C                       COMPUTATIONS AND FOR INVERM SUBROUTINES.        LEV15110
C                                                                       LEV15120
C     CASES (3) AND (4) ONLY, FACTORED INVERSES OF REAL SYMMETRIC       LEV15130
C     MATRICES AND GENERALIZED EIGENVALUE PROBLEMS:                     LEV15140
C                                                                       LEV15150
C     VS(J) =  REAL*8 ARRAY WHOSE DIMENSION MUST BE AT LEAST N.         LEV15160
C              USED IN REGENERATION OF THE LANCZOS VECTORS.             LEV15170
C                                                                       LEV15180
C     IPR(J), IPT(J) = INTEGER*4 ARRAYS.  EACH MUST BE OF DIMENSION     LEV15190
C                      AT LEAST N, THE ORDER OF A. USED TO STORE        LEV15200
C                      THE REORDERING OF THE GIVEN MATRIX.              LEV15210
C                                                                       LEV15220
C     CASE (3) ONLY, INVERSES OF REAL SYMMETRIC MATRICES:               LEV15230
C                                                                       LEV15240
C     GOODA(J), GOODBI(J) = REAL*8 ARRAYS.  EACH MUST BE OF DIMENSION   LEV15250
C                           AT LEAST NGOOD, THE NUMBER OF EIGENVALUES   LEV15260
C                           BEING CONSIDERED.  GOODA CONTAINS THE       LEV15270
C                           EIGENVALUES OF A AND GOODBI CONTAINS THE    LEV15280
C                           EIGENVALUES OF B(INVERSE).  THE PROGRAM     LEV15290
C                           WORKS DIRECTLY WITH THE GOODBI ARRAY.       LEV15300
C                                                                       LEV15310
C                                                                       LEV15320
C-----SUBROUTINES INCLUDED FOR THE EIGENVECTOR COMPUTATIONS-------------LEV15330
C                                                                       LEV15340
C                                                                       LEV15350
C     STURMI = FOR EACH GIVEN EIGENVALUE GOODEV(J) DETERMINES           LEV15360
```

```
C            THE SMALLEST SIZE T-MATRIX FOR WHICH GOODEV(J) IS        LEV15370
C            AN EIGENVALUE (TO WITHIN A GIVEN TOLERANCE) AND IF       LEV15380
C            POSSIBLE THE SMALLEST SIZE T-MATRIX FOR WHICH            LEV15390
C            IT IS A DOUBLE EIGENVALUE (TO WITHIN THE SAME            LEV15400
C            TOLERANCE).  THE SIZE T-MATRIX USED IN THE               LEV15410
C            EIGENVECTOR COMPUTATIONS IS THEN DETERMINED BY           LEV15420
C            STARTING WITH AN INITIAL GUESS BASED UPON THE            LEV15430
C            INFORMATION FROM STURMI, AND LOOPING ON THE SIZE         LEV15440
C            OF THE T-EIGENVECTOR COMPUTATIONS.                       LEV15450
C                                                                     LEV15460
C     LBISEC = RECOMPUTES THE VALUE OF THE GIVEN EIGENVALUE AT THE    LEV15470
C            T-SIZE SPECIFIED FOR THE T-EIGENVECTOR COMPUTATION.      LEV15480
C            LBISEC IS A SIMPLIFICATION OF THE BISEC SUBROUTINE       LEV15490
C            USED IN THE LANCZOS EIGENVALUE COMPUTATIONS.             LEV15500
C                                                                     LEV15510
C     INVERM = FOR THE T-SIZES CONSIDERED BY THE PROGRAM COMPUTES     LEV15520
C            THE CORRESPONDING EIGENVECTORS OF THESE T-MATRICES       LEV15530
C            CORRESPONDING TO THE USER-SUPPLIED EIGENVALUES IN        LEV15540
C            THE GOODEV ARRAY.                                        LEV15550
C                                                                     LEV15560
C     THE LANCZS, TNORM , AND CINPRD (CASE (2) ONLY) SUBROUTINES      LEV15570
C     ARE USED HERE AS WELL AS IN THE EIGENVALUE COMPUTATIONS.        LEV15580
C                                                                     LEV15590
C     IN CASES (3) AND (4) ONLY AND THEN ONLY IF THE ORIGINAL MATRIX  LEV15600
C     (MATRICES) HAS (HAVE) BEEN PERMUTED:                            LEV15610
C                                                                     LEV15620
C     LPERM  = (USED IN CASE (3) AND (4) ONLY). GIVEN A B-MATRIX AND  LEV15630
C            A PERMUTATION P DEFINED IN THE VECTORS IPR AND IPT,      LEV15640
C            AND A VECTOR X COMPUTE EITHER (P-TRANSPOSE)*X OR PX.     LEV15650
C                                                                     LEV15660
C-----------------------------------------------------------------------LEV15670
```

2.3 LEVAL: Main Program, Eigenvalue Computations

```
C-----LEVAL  (EIGENVALUES OF REAL SYMMETRIC MATRICES)-------------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C         REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4
C         Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C         Applied Mathematics, 2002. SIAM Publications,
C         Philadelphia, PA. USA
C
C
C     CONTAINS MAIN PROGRAM FOR COMPUTING DISTINCT EIGENVALUES OF
C     A REAL SYMMETRIC MATRIX USING LANCZOS TRIDIAGONALIZATION
C     WITHOUT REORTHOGONALIZATION.
C
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE
C     CONSTRUCTIONS
C
C     1.  DATA/MACHEP/ STATEMENT
C     2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)
C     3.  FORMAT(20A4) USED WITH EXPLANATORY HEADER EXPLAN.
C     4.  HEXADECIMAL FORMAT (4Z20) USED IN ALPHA/BETA FILES 1 AND 2.
C
C-----------------------------------------------------------------------
C
      DOUBLE PRECISION  ALPHA(5000),BETA(5001)
      DOUBLE PRECISION  V1(5001),V2(5000),VS(5000)
      DOUBLE PRECISION  LB(20),UB(20)
      DOUBLE PRECISION  BTOL,GAPTOL,TTOL,MACHEP,EPSM,RELTOL
      DOUBLE PRECISION  SCALE1,SCALE2,SCALE3,SCALE4,BISTOL,CONTOL,MULTOL
      DOUBLE PRECISION  ONE,ZERO,TEMP,TKMAX,BETAM,BKMIN,T0,T1
      REAL  G(10000),EXPLAN(20)
      INTEGER  MP(5000),NMEV(20)
      INTEGER  SVSEED,RHSEED,SVSOLD
      INTEGER IABS
      REAL  ABS
      DOUBLE PRECISION  DABS, DSQRT, DFLOAT
```

```
      EXTERNAL CMATV
C
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.0D0*MACHEP
C-----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     DIMENSION OF V2 ASSUMES THAT NO MORE THAN KMAX/2 EIGENVALUES
C     OF THE T-MATRICES ARE BEING COMPUTED IN ANY ONE OF THE
C     SUB-INTERVALS BEING CONSIDERED.  V2 CONTAINS THE UPPER AND LOWER
C     BOUNDS FOR EACH T-EIGENVALUE BEING COMPUTED BY BISEC IN ANY ONE
C     GIVEN INTERVAL.
C
C     1.  ALPHA: >= KMAX,   BETA: >= (KMAX+1)
C     2.  V1:  >= MAX(N,KMAX+1)
C     3.  V2:   >= MAX(N,KMAX)
C     4.  VS:  >= KMAX
C     5.  G:  >= MAX(N,2*KMAX)
C     6.  MP:    >= KMAX
C     7.  LB,UB:  >= NUMBER OF SUBINTERVALS SUPPLIED TO BISEC.
C     8.  NMEV:  >= NUMBER OF T-MATRICES ALLOWED.
C     9.  EXPLAN:  DIMENSION IS 20.
C
C
C     IMPORTANT TOLERANCES OR SCALES THAT ARE USED REPEATEDLY
C     THROUGHOUT THIS PROGRAM ARE THE FOLLOWING:
C     SCALED MACHINE EPSILON:  TTOL = TKMAX*EPSM WHERE
C     EPSM = 2*MACHINE EPSILON AND
C     TKMAX = MAX(|ALPHA(J)|,BETA(J), J = 1,MEV)
C     BISEC CONVERGENCE TOLERANCE:  BISTOL = DSQRT(1000+MEV)*TTOL
C     BISEC T-MULTIPLICITY TOLERANCE:  MULTOL = (1000+MEV)*TTOL
C     LANCZOS CONVERGENCE TOLERANCE:   CONTOL = BETA(MEV+1)*1.D-10
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LANCZOS PROCEDURE FOR REAL SYMMETRIC MATRICES'/)
C
C     SET PROGRAM PARAMETERS
C     SCALEK ARE USED IN TOLERANCES NEEDED IN SUBROUTINES LUMP,
C     ISOEV AND PRTEST.  USER MUST NOT MODIFY THESE SCALES.
      SCALE1 = 5.0D2
      SCALE2 = 5.0D0
      SCALE3 = 5.0D0
      SCALE4 = 1.0D4
      ONE  = 1.0D0
      ZERO = 0.0D0
      BTOL = 1.0D-8
C     BTOL = EPSM
      GAPTOL = 1.0D-8
      ICONV = 0
      MOLD = 0
      MOLD1 = 1
      ICT = 0
      MMB = 0
```

```
      IPROJ = 0                                                         LEV01020
C-----------------------------------------------------------------------LEV01030
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 5 (FREE FORMAT)    LEV01040
C                                                                       LEV01050
C     READ USER-PROVIDED HEADER FOR RUN                                 LEV01060
      READ(5,20) EXPLAN                                                 LEV01070
      WRITE(6,20) EXPLAN                                                LEV01080
      READ(5,20) EXPLAN                                                 LEV01090
      WRITE(6,20) EXPLAN                                                LEV01100
   20 FORMAT(20A4)                                                      LEV01110
C                                                                       LEV01120
C     READ ORDER OF MATRICES (N) , MAXIMUM ORDER OF T-MATRIX (KMAX),    LEV01130
C     NUMBER OF T-MATRICES ALLOWED (NMEVS), AND MATRIX IDENTIFICATION   LEV01140
C     NUMBERS (MATNO)                                                   LEV01150
      READ(5,20) EXPLAN                                                 LEV01160
      READ(5,*) N,KMAX,NMEVS,MATNO                                      LEV01170
C                                                                       LEV01180
C     READ SEEDS FOR LANCZS AND INVERR SUBROUTINES (SVSEED AND RHSEED)  LEV01190
C     READ MAXIMUM NUMBER OF ITERATIONS ALLOWED FOR EACH INVERSE        LEV01200
C     ITERATION (MXINIT) AND MAXIMUM NUMBER OF STURM SEQUENCES          LEV01210
C     ALLOWED (MXSTUR)                                                  LEV01220
      READ(5,20) EXPLAN                                                 LEV01230
      READ(5,*) SVSEED,RHSEED,MXINIT,MXSTUR                             LEV01240
C                                                                       LEV01250
C     ISTART = (0,1):  ISTART = 0 MEANS ALPHA/BETA FILE IS NOT          LEV01260
C     AVAILABLE.  ISTART = 1 MEANS ALPHA/BETA FILE IS AVAILABLE ON      LEV01270
C     FILE 2.                                                           LEV01280
C     ISTOP = (0,1):  ISTOP = 0 MEANS PROCEDURE GENERATES ALPHA/BETA    LEV01290
C     FILE AND THEN TERMINATES.  ISTOP = 1 MEANS PROCEDURE GENERATES    LEV01300
C     ALPHAS/BETAS IF NEEDED AND THEN COMPUTES EIGENVALUES AND ERROR    LEV01310
C     ESTIMATES AND THEN TERMINATES.                                    LEV01320
      READ(5,20) EXPLAN                                                 LEV01330
      READ(5,*) ISTART,ISTOP                                            LEV01340
C                                                                       LEV01350
C     IHIS = (0,1):  IHIS = 0 MEANS ALPHA/BETA FILE IS NOT WRITTEN      LEV01360
C     TO FILE 1.  IHIS = 1 MEANS ALPHA/BETA FILE IS WRITTEN TO FILE 1.  LEV01370
C     IDIST = (0,1):  IDIST = 0 MEANS DISTINCT T-EIGENVALUES            LEV01380
C     ARE NOT WRITTEN TO FILE 11.  IDIST = 1 MEANS DISTINCT             LEV01390
C     T-EIGENVALUES ARE WRITTEN TO FILE 11.                             LEV01400
C     IWRITE = (0,1):  IWRITE = 0 MEANS NO INTERMEDIATE OUTPUT          LEV01410
C     FROM THE COMPUTATIONS IS WRITTEN TO FILE 6.  IWRITE = 1 MEANS     LEV01420
C     T-EIGENVALUES AND ERROR ESTIMATES ARE WRITTEN TO FILE 6           LEV01430
C     AS THEY ARE COMPUTED.                                             LEV01440
      READ(5,20) EXPLAN                                                 LEV01450
      READ(5,*) IHIS,IDIST,IWRITE                                       LEV01460
C                                                                       LEV01470
C     READ IN THE RELATIVE TOLERANCE (RELTOL) FOR USE IN THE            LEV01480
C     SPURIOUS, T-MULTIPLICITY, AND PRTESTS.                            LEV01490
      READ(5,20) EXPLAN                                                 LEV01500
      READ(5,*) RELTOL                                                  LEV01510
C                                                                       LEV01520
C     READ IN THE SIZES OF THE T-MATRICES TO BE CONSIDERED.             LEV01530
      READ(5,20) EXPLAN                                                 LEV01540
      READ(5,*) (NMEV(J), J=1,NMEVS)                                    LEV01550
C                                                                       LEV01560
```

```
C     READ IN THE NUMBER OF SUBINTERVALS TO BE CONSIDERED.
      READ(5,20) EXPLAN
      READ(5,*) NINT
C
C     READ IN THE LEFT-END POINTS OF THE SUBINTERVALS TO BE CONSIDERED.
C     THESE MUST BE IN ALGEBRAICALLY-INCREASING ORDER
      READ(5,20) EXPLAN
      READ(5,*) (LB(J), J=1,NINT)
C
C     READ IN THE RIGHT-END POINTS OF THE SUBINTERVALS TO BE CONSIDERED.
C     THESE MUST BE IN ALGEBRAICALLY-INCREASING ORDER
      READ(5,20) EXPLAN
      READ(5,*) (UB(J), J=1,NINT)
C
C-----------------------------------------------------------------------
C     INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED MATRIX
C     AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE
C     MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV.
C
      CALL USPEC(N,MATNO)
C
C-----------------------------------------------------------------------
C     MASK UNDERFLOW AND OVERFLOW
C
      CALL MASK
C
C-----------------------------------------------------------------------
C
C     WRITE TO FILE 6, A SUMMARY OF THE PARAMETERS FOR THIS RUN
C
      WRITE(6,30) MATNO,N,KMAX
   30 FORMAT(/3X,'MATRIX ID',4X,'ORDER OF A',4X,'MAX ORDER OF T'/
     1 I12,I14,I18/)
C
      WRITE(6,40) ISTART,ISTOP
   40 FORMAT(/2X,'ISTART',3X,'ISTOP'/2I8/)
C
      WRITE(6,50) IHIS,IDIST,IWRITE
   50 FORMAT(/4X,'IHIS',3X,'IDIST',2X,'IWRITE'/3I8/)
C
      WRITE(6,60) SVSEED,RHSEED
   60 FORMAT(/' SEEDS FOR RANDOM NUMBER GENERATOR'//
     1 4X,'LANCZS SEED',4X,'INVERR SEED'/2I15/)
C
      WRITE(6,70) (NMEV(J), J=1,NMEVS)
   70 FORMAT(/' SIZES OF T-MATRICES TO BE CONSIDERED'/(6I12))
C
      WRITE(6,80) RELTOL,GAPTOL,BTOL
   80 FORMAT(/' RELATIVE TOLERANCE USED TO COMBINE COMPUTED T-EIGENVALUE
     1S'/E15.3/' RELATIVE GAP TOLERANCES USED IN INVERSE ITERATION'/
     1E15.3/' RELATIVE TOLERANCE FOR CHECK ON SIZE OF BETAS'/E15.3/)
C
      WRITE(6,90) (J,LB(J),UB(J), J=1,NINT)
   90 FORMAT(/' BISEC WILL BE USED ON THE FOLLOWING INTERVALS'/
     1 (I6,2E20.6))
```

```
C
      IF (ISTART.EQ.0) GO TO 140
C
C     READ IN ALPHA BETA HISTORY
C
      READ(2,100)MOLD,NOLD,SVSOLD,MATOLD
  100 FORMAT(2I6,I12,I8)
C
      IF (KMAX.LT.MOLD) KMAX = MOLD
      KMAX1 = KMAX + 1
C
C     CHECK THAT ORDER N, MATRIX ID MATNO, AND RANDOM SEED SVSEED
C     AGREE WITH THOSE IN THE HISTORY FILE.  IF NOT PROCEDURE STOPS.
C
      ITEMP = (NOLD-N)**2+(MATNO-MATOLD)**2+(SVSEED-SVSOLD)**2
C
      IF (ITEMP.EQ.0) GO TO 120
C
      WRITE(6,110)
  110 FORMAT(' PROGRAM TERMINATES'/    ' READ FROM FILE 2 CORRESPONDS TO
     1 DIFFERENT MATRIX THAN MATRIX SPECIFIED'/)
      GO TO 640
C
  120 CONTINUE
      MOLD1 = MOLD+1
C
      READ(2,130)(ALPHA(J), J=1,MOLD)
      READ(2,130)(BETA(J), J=1,MOLD1)
  130 FORMAT(4Z20)
C
      IF (KMAX.EQ.MOLD) GO TO 160
C
      READ(2,130)(V1(J), J=1,N)
      READ(2,130)(V2(J), J=1,N)
C
  140 CONTINUE
      IIX = SVSEED
C
C-----------------------------------------------------------------------
C
      CALL LANCZS(CMATV,ALPHA,BETA,V1,V2,G,KMAX,MOLD1,N,IIX)
C
C-----------------------------------------------------------------------
C
      KMAX1 = KMAX + 1
C
      IF (IHIS.EQ.0.AND.ISTOP.GT.0) GO TO 160
C
      WRITE(1,150) KMAX,N,SVSEED,MATNO
  150 FORMAT(2I6,I12,I8,' = KMAX,N,SVSEED,MATNO')
C
      WRITE(1,130)(ALPHA(I), I=1,KMAX)
      WRITE(1,130)(BETA(I), I=1,KMAX1)
C
      WRITE(1,130)(V1(I), I=1,N)
```

```
      WRITE(1,130)(V2(I), I=1,N)
C
      IF (ISTOP.EQ.0) GO TO 540
C
  160 CONTINUE
      BKMIN = BTOL
C
      WRITE(6,170)
  170 FORMAT(/' T-MATRICES (ALPHA AND BETA) ARE NOW AVAILABLE'/)
C
C-----------------------------------------------------------------------
C     SUBROUTINE TNORM CHECKS MIN(BETA)/(ESTIMATED NORM(A)) > BTOL .
C     IF THIS IS VIOLATED IB IS SET EQUAL TO THE NEGATIVE OF THE INDEX
C     OF THE MINIMAL BETA.  IF(IB < 0) THEN SUBROUTINE TNORM IS
C     CALLED FOR EACH VALUE OF MEV TO DETERMINE WHETHER OR NOT THERE
C     IS A BETA IN THE T-MATRIX SPECIFIED THAT VIOLATES THIS TEST.
C     IF THERE IS SUCH A BETA THE PROGRAM TERMINATES FOR THE USER
C     TO DECIDE WHAT TO DO.  THIS TEST CAN BE OVER-RIDDEN BY
C     SIMPLY MAKING BTOL SMALLER, BUT THEN THERE IS THE POSSIBILITY
C     THAT LOSSES IN THE LOCAL ORTHOGONALITY MAY HURT THE COMPUTATIONS.
C     BTOL = 1.D-8 IS HOWEVER A CONSERVATIVE CHOICE FOR BTOL.
C
C     TNORM ALSO COMPUTES TKMAX = MAX(|ALPHA(K)|,BETA(K), K=1,KMAX).
C     TKMAX IS USED TO SCALE THE TOLERANCES USED IN THE
C     T-MULTIPLICITY AND SPURIOUS TESTS IN BISEC. TKMAX IS ALSO USED IN
C     THE PROJECTION TEST FOR HIDDEN EIGENVALUES THAT HAD 'TOO SMALL'
C     A PROJECTION ON THE STARTING VECTOR.
C
      CALL TNORM(ALPHA,BETA,BKMIN,TKMAX,KMAX,IB)
C
C-----------------------------------------------------------------------
C
      TTOL = EPSM*TKMAX
C
C     LOOP ON THE SIZE OF THE T-MATRIX
C
  180 CONTINUE
      MMB = MMB + 1
      MEV = NMEV(MMB)
C     IS MEV TOO LARGE ?
      IF(MEV.LE.KMAX) GO TO 200
      WRITE(6,190) MMB, MEV, KMAX
  190 FORMAT(/' TERMINATE PRIOR TO CONSIDERING THE',I6,'TH T-MATRIX'/
     1' BECAUSE THE SIZE REQUESTED',I6,' IS GREATER THAN THE MAXIMUM SIZ
     1E ALLOWED',I6/)
      GO TO 540
C
  200 MP1 = MEV + 1
      BETAM = BETA(MP1)
C
      IF (IB.GE.0) GO TO 210
C
      T0 = BTOL
C
C-----------------------------------------------------------------------
```

```
C
      CALL TNORM(ALPHA,BETA,T0,T1,MEV,IBMEV)
C
C-----------------------------------------------------------------------
C
      TEMP = T0/TKMAX
      IBMEV = IABS(IBMEV)
      IF (TEMP.GE.BTOL) GO TO 210
      IBMEV = -IBMEV
      GO TO 600
C
  210 CONTINUE
      IC = MXSTUR-ICT
C
C-----------------------------------------------------------------------
C     BISEC LOOP. THE SUBROUTINE BISEC INCORPORATES DIRECTLY THE
C     T-MULTIPLICITY AND SPURIOUS TESTS. T-EIGENVALUES WILL BE
C     CALCULATED BY BISEC SEQUENTIALLY ON INTERVALS
C     (LB(J),UB(J)), J = 1,NINT).
C
C     ON RETURN FROM BISEC
C     NDIS = NUMBER OF DISTINCT EIGENVALUES OF T(1,MEV) ON UNION
C            OF THE (LB,UB) INTERVALS
C     VS = DISTINCT T-EIGENVALUES IN ALGEBRAICALLY INCREASING ORDER
C     MP = MULTIPLICITIES OF THE T-EIGENVALUES IN VS
C     MP(I) = (0,1,MI), MI>1, I=1,NDIS  MEANS:
C        (0)  VS(I) IS SPURIOUS
C        (1)  VS(I) IS T-SIMPLE AND GOOD
C        (MI) VS(I) IS MULTIPLE AND IS THEREFORE NOT ONLY GOOD BUT
C             ALSO A CONVERGED GOOD T-EIGENVALUE.
C
C
      CALL BISEC(ALPHA,BETA,V1,V2,VS,LB,UB,EPSM,TTOL,MP,NINT,
     1 MEV,NDIS,IC,IWRITE)
C
C-----------------------------------------------------------------------
C
      IF (NDIS.EQ.0) GO TO 620
C
C     COMPUTE THE TOTAL NUMBER OF STURM SEQUENCES USED TO DATE
C     COMPUTE THE BISEC CONVERGENCE AND T-MULTIPLICITY TOLERANCES USED.
C     COMPUTE THE CONVERGENCE TOLERANCE FOR EIGENVALUES OF A.
      ICT = ICT + IC
      TEMP = DFLOAT(MEV+1000)
      MULTOL = TEMP*TTOL
      TEMP = DSQRT(TEMP)
      BISTOL = TTOL*TEMP
      CONTOL = BETAM*1.D-10
C
C-----------------------------------------------------------------------
C    SUBROUTINE LUMP 'COMBINES' T-EIGENVALUES THAT ARE 'TOO CLOSE'.
C    NOTE HOWEVER THAT CLOSE SPURIOUS T-EIGENVALUES ARE NOT AVERAGED
C    WITH GOOD ONES. HOWEVER, THEY MAY BE USED TO INCREASE THE
C    MULTIPLICITY OF A GOOD T-EIGENVALUE.
C
```

```
      LOOP = NDIS
      CALL LUMP(VS,RELTOL,MULTOL,SCALE2,MP,LOOP)
C
C-----------------------------------------------------------------------
C
      IF(NDIS.EQ.LOOP) GO TO 230
C
      WRITE(6,220) NDIS, MEV, LOOP
  220 FORMAT(/I6,' DISTINCT T-EIGENVALUES WERE COMPUTED IN BISEC AT MEV
     1',I6/ 2X,' LUMP SUBROUTINE REDUCES NUMBER OF DISTINCT EIGENVALUES
     1O',I6)
C
  230 CONTINUE
      NDIS = LOOP
      BETA(MP1) = BETAM
C
C-----------------------------------------------------------------------
C     THE SUBROUTINE ISOEV LABELS THOSE SIMPLE EIGENVALUES OF T(1,MEV)
C     WITH VERY SMALL GAPS BETWEEN NEIGHBORING EIGENVALUES OF T(1,MEV)
C     TO AVOID COMPUTING ERROR ESTIMATES FOR ANY SIMPLE GOOD
C     T-EIGENVALUE THAT IS TOO CLOSE TO A SPURIOUS EIGENVALUE.
C     ON RETURN FROM ISOEV, G CONTAINS CODED MINIMAL GAPS
C     BETWEEN THE DISTINCT EIGENVALUES OF T(1,MEV). (G IS REAL).
C     G(I) < 0 MEANS MINGAP IS DUE TO LEFT GAP G(I) > 0 MEANS DUE TO
C     RIGHT GAP. MP(I) = -1 MEANS THAT THE GOOD T-EIGENVALUE IS SIMPLE
C     AND HAS A VERY SMALL MINGAP IN T(1,MEV) DUE TO A SPURIOUS
C     EIGENVALUE.  NG = NUMBER OF GOOD T-EIGENVALUES.
C     NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES.
C
      CALL ISOEV(VS,GAPTOL,MULTOL,SCALE1,G,MP,NDIS,NG,NISO)
C
C-----------------------------------------------------------------------
C
      WRITE(6,240)NG,NISO,NDIS
  240 FORMAT(/I6,' GOOD T-EIGENVALUES HAVE BEEN COMPUTED'/
     1 I6,' OF THESE ARE T-ISOLATED'/
     2 I6,' = NUMBER OF DISTINCT T-EIGENVALUES COMPUTED'/)
C
C     DO WE WRITE DISTINCT EIGENVALUES OF T-MATRIX TO FILE 11?
      IF (IDIST.EQ.0) GO TO 280
C
      WRITE(11,250) NDIS,NISO,MEV,N,SVSEED,MATNO
  250 FORMAT(/4I6,I12,I8,' = NDIS,NISO,MEV,N,SVSEED,MATNO'/)
C
      WRITE(11,260) (MP(I),VS(I),G(I), I=1,NDIS)
  260 FORMAT(2(I3,E25.16,E12.3))
C
      WRITE(11,270) NDIS, (MP(I), I=1,NDIS)
  270 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/(20I4))
C
  280 CONTINUE
C
      IF (NISO.NE.0) GO TO 310
C
      WRITE(4,290) MEV
```

```
  290 FORMAT(/' AT MEV = ',I6,' THERE ARE NO ISOLATED T-EIGENVALUES'/
     1' SO NO ERROR ESTIMATES WERE COMPUTED/')
C
      WRITE(6,300)
  300 FORMAT(/' ALL COMPUTED GOOD T-EIGENVALUES ARE MULTIPLE'/
     1 ' THEREFORE ALL SUCH EIGENVALUES ARE ASSUMED TO HAVE CONVERGED')
C
      ICONV = 1
      GO TO 350
C
  310 CONTINUE
C
C-----------------------------------------------------------------------
C     SUBROUTINE INVERR COMPUTES ERROR ESTIMATES FOR ISOLATED GOOD
C     T-EIGENVALUES USING INVERSE ITERATION ON T(1,MEV). ON RETURN
C     G(J) = MINIMUM GAP IN T(1,MEV) FOR EACH VS(J), J=1,NDIS
C     G(MEV+I) = BETAM*|U(MEV)| = ERROR ESTIMATE FOR ISOLATED GOOD
C              T-EIGENVALUES, WHERE I = 1, NISO AND  BETAM = BETA(MEV+1)
C              U(MEV) IS MEVTH COMPONENT OF THE UNIT EIGENVECTOR OF T
C              CORRESPONDING TO THE ITH ISOLATED GOOD T-EIGENVALUE.
C     A NEGATIVE ERROR ESTIMATE MEANS THAT FOR THAT PARTICULAR
C     EIGENVALUE THE INVERSE ITERATION DID NOT CONVERGE IN <= MXINIT
C     STEPS AND THAT THE CORRESPONDING ERROR ESTIMATE IS QUESTIONABLE.
C
C     V2 CONTAINS THE ISOLATED GOOD T-EIGENVALUES
C     V1 CONTAINS THE MINGAPS TO THE NEAREST DISTINCT  EIGENVALUE
C        OF T(1,MEV) FOR EACH ISOLATED GOOD T-EIGENVALUE IN V2.
C     VS CONTAINS THE NDIS DISTINCT EIGENVALUES OF T(1,MEV)
C     MP CONTAINS THE CORRESPONDING CODED T-MULTIPLICITIES
C
      IT = MXINIT
      CALL INVERR(ALPHA,BETA,V1,V2,VS,EPSM,G,MP,MEV,MMB,NDIS,NISO,N,
     1    RHSEED,IT,IWRITE)
C
C-----------------------------------------------------------------------
C
C     SIMPLE CHECK FOR CONVERGENCE. CHECKS TO SEE IF ALL OF THE ERROR
C     ESTIMATES ARE SMALLER THAN CONTOL = BETAM*1.D-10.
C     IF THIS TEST IS SATISFIED, THEN CONVERGENCE FLAG, ICONV IS SET
C     TO 1.  TYPICALLY ERROR ESTIMATES ARE VERY CONSERVATIVE.
C
      WRITE(6,320) CONTOL
  320 FORMAT(/' CONVERGENCE IS TESTED USING THE CONVERGENCE TOLERANCE',
     1E13.4/)
C
      II = MEV +1
      IF = MEV+NISO
      DO 330 I = II,IF
      IF (ABS(G(I)).GT.CONTOL) GO TO 350
  330 CONTINUE
      ICONV = 1
      MMB = NMEVS
C
      WRITE(6,340) CONTOL
  340 FORMAT(' ALL COMPUTED ERROR ESTIMATES WERE LESS THAN',E15.4/
```

```
     1 ' THEREFORE PROCEDURE TERMINATES'/)                              LEV04870
C                                                                       LEV04880
  350 CONTINUE                                                          LEV04890
C                                                                       LEV04900
C     IF CONVERGENCE IS INDICATED, THAT IS ICONV = 1 ,THEN              LEV04910
C     THE SUBROUTINE PRTEST IS CALLED TO CHECK FOR ANY CONVERGED        LEV04920
C     T-EIGENVALUES THAT HAVE BEEN MISLABELLED AS SPURIOUS BECAUSE      LEV04930
C     THE PROJECTION OF THEIR EIGENVECTOR(S) ON THE STARTING            LEV04940
C     VECTOR WERE TOO SMALL.                                            LEV04950
C     NUMERICAL TESTS INDICATE THAT SUCH EIGENVALUES ARE RARE.          LEV04960
C     IF FOR SOME REASON MANY OF THESE HIDDEN EIGENVALUES APPEAR        LEV04970
C     ON SOME RUN, YOU CAN BE CERTAIN THAT SOMETHING IS FOULED UP.      LEV04980
C                                                                       LEV04990
      IF (ICONV.EQ.0) GO TO 480                                         LEV05000
C                                                                       LEV05010
C-----------------------------------------------------------------------LEV05020
C                                                                       LEV05030
      CALL PRTEST (ALPHA,BETA,VS,TKMAX,EPSM,RELTOL,SCALE3,SCALE4,       LEV05040
     1 MP,NDIS,MEV,IPROJ)                                               LEV05050
C                                                                       LEV05060
C-----------------------------------------------------------------------LEV05070
C                                                                       LEV05080
      IF(IPROJ.EQ.0) GO TO 470                                          LEV05090
C                                                                       LEV05100
      IF(IDIST.EQ.1)  WRITE(11,360) IPROJ                               LEV05110
  360 FORMAT(' SUBROUTINE PRTEST WANTS TO RELABEL',I6,' SPURIOUS T-EIGENLEV05120
     1VALUES'/' WE ACCEPT RELABELLING ONLY IF LAST COMPONENT OF T-EIGENVLEV05130
     1ECTOR IS L.T. 1.D-10'/)                                           LEV05140
C                                                                       LEV05150
      IIX = RHSEED                                                      LEV05160
C                                                                       LEV05170
C-----------------------------------------------------------------------LEV05180
C                                                                       LEV05190
      CALL GENRAN(IIX,G,MEV)                                            LEV05200
C                                                                       LEV05210
C-----------------------------------------------------------------------LEV05220
C                                                                       LEV05230
      ITEN = -10                                                        LEV05240
      NISOM = NISO + MEV                                                LEV05250
      IWRITO = IWRITE                                                   LEV05260
      IWRITE = 0                                                        LEV05270
C                                                                       LEV05280
      DO 390 J = 1,NDIS                                                 LEV05290
      IF(MP(J).NE.ITEN) GO TO 390                                       LEV05300
      T0 = VS(J)                                                        LEV05310
C                                                                       LEV05320
C-----------------------------------------------------------------------LEV05330
C                                                                       LEV05340
      IT = MXINIT                                                       LEV05350
      CALL INVERM(ALPHA,BETA,V1,V2,T0,TEMP,T1,EPSM,G,MEV,IT,IWRITE)     LEV05360
C                                                                       LEV05370
C-----------------------------------------------------------------------LEV05380
C                                                                       LEV05390
      IF(TEMP.LE.1.D-10) GO TO 380                                      LEV05400
C     ERROR ESTIMATE WAS NOT SMALL REJECT RELABELLING OF THIS EIGENVALUELEV05410
```

```
      IF(IDIST.EQ.1)  WRITE(11,370) J,T0,TEMP                           LEV05420
  370 FORMAT(/' LAST COMPONENT FOR',I6,'TH EIGENVALUE',E20.12/' IS TOO LLEV05430
     1ARGE = ',E15.6,' SO DO NOT ACCEPT PRTEST RELABELLING'/)           LEV05440
      MP(J) = 0                                                         LEV05450
      IPROJ = IPROJ - 1                                                 LEV05460
      GO TO 390                                                         LEV05470
C     RELABELLING ACCEPTED                                              LEV05480
  380 NISOM = NISOM + 1                                                 LEV05490
      G(NISOM) = BETAM*TEMP                                             LEV05500
  390 CONTINUE                                                          LEV05510
      IWRITE = IWRITO                                                   LEV05520
C                                                                       LEV05530
      IF(IPROJ.EQ.0) GO TO 430                                          LEV05540
      WRITE(6,400) IPROJ                                                LEV05550
  400 FORMAT(/I6,' T-EIGENVALUES WERE RECLASSIFIED AS GOOD.'/           LEV05560
     1' THESE ARE IDENTIFIED IN FILE 3 BY A T-MULTIPLICITY OF -10'/' USELEV05570
     2R SHOULD INSPECT EACH TO MAKE SURE NEIGHBORS HAVE CONVERGED'/)    LEV05580
C                                                                       LEV05590
      IF(IDIST.EQ.1)  WRITE(11,410) IPROJ                               LEV05600
  410 FORMAT(/I6,' T-EIGENVALUES WERE RELABELLED AS GOOD'/              LEV05610
     1' BELOW IS CORRECTED T-MULTIPLICITY PATTERN'/)                    LEV05620
C                                                                       LEV05630
      WRITE(6,420) NDIS, (MP(I), I=1,NDIS)                              LEV05640
      IF(IDIST.EQ.1)  WRITE(11,420) NDIS, (MP(I), I=1,NDIS)             LEV05650
  420 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/       LEV05660
     1 6X, ' (-10) MEANS SPURIOUS T-EIGENVALUE RELABELLED AS GOOD'/(20I4LEV05670
     1))                                                                LEV05680
C                                                                       LEV05690
C     RECALCULATE MINGAPS FOR DISTINCT T(1,MEV) EIGENVALUES.            LEV05700
  430 NM1 = NDIS - 1                                                    LEV05710
      G(NDIS) = VS(NM1)-VS(NDIS)                                        LEV05720
      G(1) = VS(2)-VS(1)                                                LEV05730
C                                                                       LEV05740
      DO 440 J = 2,NM1                                                  LEV05750
      T0 = VS(J)-VS(J-1)                                                LEV05760
      T1 = VS(J+1)-VS(J)                                                LEV05770
      G(J) = T1                                                         LEV05780
      IF (T0.LT.T1) G(J) = -T0                                          LEV05790
  440 CONTINUE                                                          LEV05800
      IF(IPROJ.EQ.0) GO TO 470                                          LEV05810
C     WRITE TO FILE 4 ERROR ESTIMATES FOR THOSE T-EIGENVALUES RELABELLEDLEV05820
      NGOOD = 0                                                         LEV05830
      DO 450 J = 1,NDIS                                                 LEV05840
      IF(MP(J).EQ.0) GO TO 450                                          LEV05850
      NGOOD = NGOOD + 1                                                 LEV05860
      IF(MP(J).NE.ITEN) GO TO 450                                       LEV05870
      T0 = VS(J)                                                        LEV05880
      NISO = NISO + 1                                                   LEV05890
      NISOM = MEV + NISO                                                LEV05900
      WRITE(4,460) NGOOD,T0,G(NISOM),G(J)                               LEV05910
  450 CONTINUE                                                          LEV05920
  460 FORMAT(I10,E25.16,2E14.3)                                         LEV05930
C                                                                       LEV05940
  470 CONTINUE                                                          LEV05950
C                                                                       LEV05960
```

```
C     WRITE THE GOOD T-EIGENVALUES TO FILE 3.  FIRST TRANSFER THEM      LEV05970
C     TO V2 AND THEIR T-MULTIPLICITIES TO THE CORRESPONDING POSITIONS   LEV05980
C     IN MP AND COMPUTE THE A-MINGAPS, THE MINIMAL GAPS BETWEEN THE     LEV05990
C     GOOD T-EIGENVALUES.  THESE GAPS WILL BE PUT IN THE ARRAY G.       LEV06000
C     SINCE G CURRENTLY CONTAINS THE MINIMAL GAPS BETWEEN THE DISTINCT  LEV06010
C     EIGENVALUES OF THE T-MATRIX, THESE GAPS WILL FIRST BE             LEV06020
C     TRANSFERRED TO V1.  NOTE THAT V1<0 MEANS THAT THAT MINIMAL GAP    LEV06030
C     IN THE T-MATRIX IS DUE TO A SPURIOUS T-EIGENVALUE.                LEV06040
C     ALL THIS INFORMATION IS PRINTED TO FILE 3                         LEV06050
C                                                                       LEV06060
  480 CONTINUE                                                          LEV06070
C                                                                       LEV06080
      NG = 0                                                            LEV06090
      DO 490 I = 1,NDIS                                                 LEV06100
      IF (MP(I).EQ.0) GO TO 490                                         LEV06110
      NG = NG+1                                                         LEV06120
      MP(NG) = MP(I)                                                    LEV06130
      V2(NG) = VS(I)                                                    LEV06140
      TEMP = G(I)                                                       LEV06150
      TEMP = DABS(TEMP)                                                 LEV06160
      J = I+1                                                           LEV06170
      IF (G(I).LT.ZERO) J = I-1                                         LEV06180
      IF (MP(J).EQ.0) TEMP = -TEMP                                      LEV06190
      V1(NG) = TEMP                                                     LEV06200
  490 CONTINUE                                                          LEV06210
C                                                                       LEV06220
      WRITE(6,500)MEV                                                   LEV06230
  500 FORMAT(//' T-EIGENVALUE CALCULATION AT MEV = ',I6,'    IS COMPLETELEV06240
     1')                                                                LEV06250
C                                                                       LEV06260
C     NG = NUMBER OF COMPUTED DISTINCT GOOD T-EIGENVALUES.  NEXT        LEV06270
C     GENERATE GAPS BETWEEN GOOD T-EIGENVALUES (AMINGAPS) AND PUT THEM  LEV06280
C     IN G.  G(J) < 0 MEANS THE AMINGAP IS DUE TO THE LEFT-HAND GAP.    LEV06290
C                                                                       LEV06300
      NGM1 = NG - 1                                                     LEV06310
      G(NG) = V2(NGM1)-V2(NG)                                           LEV06320
      G(1) = V2(2)-V2(1)                                                LEV06330
C                                                                       LEV06340
      DO 510 J = 2,NGM1                                                 LEV06350
      T0 = V2(J)-V2(J-1)                                                LEV06360
      T1 = V2(J+1)-V2(J)                                                LEV06370
      G(J) = T1                                                         LEV06380
      IF (T0.LT.T1) G(J) = -T0                                          LEV06390
  510 CONTINUE                                                          LEV06400
C                                                                       LEV06410
C     WRITE GOOD T-EIGENVALUES OUT TO FILE 3.                           LEV06420
C                                                                       LEV06430
      WRITE(3,520)NG,NDIS,MEV,N,SVSEED,MATNO,MULTOL,IB,BTOL             LEV06440
  520 FORMAT(4I6,I12,I8,' = NG,NDIS,MEV,N,SVEED,MATNO'/                 LEV06450
     1 E20.12,I6,E13.4,' = MUTOL,INDEX MINIMAL BETA,BTOL'/              LEV06460
     1' EV NO',1X,'TMULT',10X,'GOOD EIGENVALUE',7X,'TMINGAP',7X,'AMINGAPLEV06470
     1')                                                                LEV06480
C                                                                       LEV06490
      WRITE(3,530)(I,MP(I),V2(I),V1(I),G(I), I=1,NG)                    LEV06500
  530 FORMAT(2I6,E25.16,2E14.3)                                         LEV06510
```

```
C
C     IF CONVERGENCE FLAG ICONV.NE.1 AND NUMBER OF T-MATRICES
C     CONSIDERED TO DATE IS LESS THAN NUMBER ALLOWED, INCREMENT MEV.
C     AND LOOP BACK TO 210 TO REPEAT COMPUTATIONS.  RESTORE BETA(MEV+1).
C
      BETA(MP1) = BETAM
C
      IF (MMB.LT.NMEVS.AND.ICONV.NE.1) GO TO 180
C
C     END OF LOOP ON DIFFERENT SIZE T-MATRICES ALLOWED.
C
  540 CONTINUE
C
      IF(ISTOP.EQ.0)  WRITE(6,550)
  550 FORMAT(/' T-MATRICES (ALPHA AND BETA) ARE NOW AVAILABLE, TERMINATE
     1')
      IF (IHIS.EQ.1.AND.KMAX.NE.MOLD) WRITE(1,560)
  560 FORMAT(/' ABOVE ARE THE FOLLOWING VECTORS '/
     1 '  ALPHA(I), I = 1,KMAX'/
     2 '  BETA(I), I = 1,KMAX+1'/
     3 ' FINAL TWO LANCZOS VECTORS OF ORDER N FOR I = KMAX,KMAX+1'/
     4 ' ALL VECTORS IN THIS FILE HAVE HEX FORMAT 4Z20 '/
     5 ' ----- END OF FILE 1 NEW ALPHA, BETA HISTORY---------------'///)
C
      IF (ISTOP.EQ.0) GO TO 640
C
      WRITE(3,570)
  570 FORMAT(/' ABOVE ARE COMPUTED GOOD T-EIGENVALUES'/
     1 ' NG = NUMBER OF GOOD T-EIGENVALUES COMPUTED'/
     2 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)'/
     3 ' N = ORDER OF A,  MATNO = MATRIX IDENT'/
     4 ' MULTOL = T-MULTIPLICITY TOLERANCE FOR T-EIGENVALUES IN BISEC'/
     4 ' TMULT IS THE T-MULTIPLICITY OF GOOD T-EIGENVALUE'/
     5 ' TMULT = -1 MEANS SPURIOUS T-EIGENVALUE TOO CLOSE'/
     6 ' DO NOT COMPUTE ERROR ESTIMATES FOR SUCH EIGENVALUES'/
     7 ' AMINGAP = MINIMAL GAP BETWEEN THE COMPUTED A-EIGENVALUES'/
     8 ' AMINGAP .LT. 0. MEANS MINIMAL GAP IS DUE TO LEFT-HAND GAP'/
     9 ' TMINGAP= MINIMAL GAP W.R.T.  DISTINCT EIGENVALUES IN T(1,MEV)'/
     1 ' TMINGAP .LT. 0. MEANS MINGAP IS DUE TO SPURIOUS T-EIGENVALUE'/
     2 ' ----- END OF FILE 3 GOODEIGENVALUES-----------------------'///)
C
      IF (IDIST.EQ.1) WRITE(11,580)
  580 FORMAT(/' ABOVE ARE THE DISTINCT EIGENVALUES OF T(1,MEV).'/
     2 ' THE FORMAT IS      T-MULTIPLICITY    T-EIGENVALUE   TMINGAP'/
     3 '        THIS FORMAT IS REPEATED TWICE ON EACH LINE.'/
     4 ' T-MULTIPLICITY = -1 MEANS THAT THE SUBROUTINE ISOEV HAS TAGGED'
     5/'   THIS SIMPLE T-EIGENVALUE AS HAVING A VERY CLOSE SPURIOUS'/
     6 '   T-EIGENVALUE SO THAT NO ERROR ESTIMATE WILL BE COMPUTED'/
     7 '   FOR THAT EIGENVALUE IN SUBROUTINE INVERR.'/
     8 ' TMINGAP .LT. 0, TMINGAP IS DUE TO LEFT GAP .GT. 0, RIGHT GAP.'/
     9 ' EACH OF THE DISTINCT T-EIGENVALUE TABLES IS FOLLOWED'/
     9 ' BY THE T-MULTIPLICITY PATTERN.'/
     1 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV).'/
     2 ' NG = NUMBER OF GOOD T-EIGENVALUES. '/
     3 ' NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES. '/
```

```
     4 ' NISO ALSO IS THE COUNT OF +1 ENTRIES IN T-MULTIPLICITY PATTERN.
     5 '/' ----- END OF FILE 11 DISTINCT T-EIGENVALUES--------------'///
     6 )
C
      IF(NISO.NE.0)  WRITE(4,590)
  590 FORMAT(/' ABOVE ARE THE ERROR ESTIMATES OBTAINED FOR THE ISOLATED
     1GOOD T-EIGENVALUES'/
     1' OBTAINED VIA INVERSE ITERATION IN THE SUBROUTINE INVERR.'/
     1' ALL OTHER GOOD T-EIGENVALUES HAVE CONVERGED.'/
     2' ERROR ESTIMATE = BETAM*ABS(UM)'/
     2' WHERE BETAM = BETA(MEV+1) AND UM = U(MEV).'/
     3' U = UNIT EIGENVECTOR OF T WHERE T*U = EV*U AND EV = ISOLATED GOO
     3D T-EIGENVALUE.'/
     4' TMINGAP = GAP TO NEAREST DISTINCT EIGENVALUE OF T(1,MEV).'/
     5' TMINGAP .LT. 0. MEANS MINGAP IS DUE TO A LEFT NEIGHBOR.'/
     6' ERROR ESTIMATE L.T. 0 MEANS INVERSE ITERATION DID NOT CONVERGE'/
     7' ------ END OF FILE 4 ERRINV -------------------------------'//)
      GO TO 640
C
  600 CONTINUE
C
      IBB = IABS(IBMEV)
      IF (IBMEV.LT.0) WRITE(6,610) MEV,IBB,BETA(IBB)
  610 FORMAT(/' PROGRAM TERMINATES BECAUSE MEV REQUESTED = ',I6,' IS .GT
     1',I6/' AT WHICH AN ABNORMALLY SMALL BETA = ' , E13.4,' OCCURRED'/)
      GO TO 640
C
  620 IF (NDIS.EQ.0.AND.ISTOP.GT.0) WRITE(6,630)
  630 FORMAT(/' INTERVALS SPECIFIED FOR BISECT DID NOT CONTAIN ANY T-EIG
     1ENVALUES'/' PROGRAM TERMINATES')
C
  640 CONTINUE
C
      STOP
C-----END OF MAIN PROGRAM FOR LANCZOS EIGENVALUE COMPUTATIONS-----------
      END
```

2.4 LEVEC: Main Program, Eigenvector Computations

```
C-----LEVEC (EIGENVECTORS OF REAL SYMMETRIC MATRICES)-------------------LEV00010
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)              LEV00020
C            Los Alamos National Laboratory                             LEV00030
C            Los Alamos, New Mexico 87544                               LEV00040
C                                                                       LEV00050
C            E-mail:  cullumj@lanl.gov                                  LEV00060
C                                                                       LEV00070
C  These codes are copyrighted by the authors.  These codes             LEV00080
C  and modifications of them or portions of them are NOT to be          LEV00090
C  incorporated into any commercial codes or used for any other         LEV00100
C  commercial purposes such as consulting for other companies,          LEV00110
C  without legal agreements with the authors of these Codes.            LEV00120
C  If these Codes or portions of them are used in other scientific or   LEV00130
C  engineering research works the names of the authors of these codes   LEV00140
C  and appropriate references to their written work are to be          LEV00150
C  incorporated in the derivative works.                                LEV00160
C                                                                       LEV00170
C  This header is not to be removed from these codes.                   LEV00180
C                                                                       LEV00190
C         REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4            LEV00191
C         Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsLEV00192
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   LEV00193
C         Applied Mathematics, 2002. SIAM Publications,                 LEV00194
C         Philadelphia, PA. USA                                         LEV00195
C                                                                       LEV00196
C                                                                       LEV00197
C                                                                       LEV00200
C     CONTAINS MAIN PROGRAM FOR COMPUTING AN EIGENVECTOR CORRESPONDING  LEV00210
C     TO EACH OF A SET OF EIGENVALUES WHICH HAVE BEEN COMPUTED          LEV00220
C     ACCURATELY BY THE CORRESPONDING LANCZOS EIGENVALUE PROGRAM        LEV00230
C     (LEVAL) FOR REAL SYMMETRIC MATRICES.  THIS PROGRAM COULD BE       LEV00240
C     MODIFIED TO COMPUTE ADDITIONAL EIGENVECTORS FOR ANY EIGENVALUE    LEV00250
C     WHICH IS A MULTIPLE EIGENVALUE OF THE GIVEN A-MATRIX.  THE        LEV00260
C     AMOUNT OF ADDITIONAL COMPUTATION REQUIRED WOULD DEPEND UPON       LEV00270
C     THE GIVEN A-MATRIX AND UPON WHAT PART OF THE SPECTRUM OF          LEV00280
C     A IS INVOLVED.                                                    LEV00290
C                                                                       LEV00300
C     THE LANCZOS EIGENVECTOR COMPUTATIONS ASSUME THAT EACH             LEV00310
C     EIGENVALUE THAT IS BEING CONSIDERED HAS CONVERGED AS AN           LEV00320
C     EIGENVALUE OF THE LANCZOS TRIDIAGONAL MATRICES.                   LEV00330
C                                                                       LEV00340
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE               LEV00350
C     CONSTRUCTIONS                                                     LEV00360
C                                                                       LEV00370
C     1.  DATA/MACHEP/ STATEMENT                                        LEV00380
C     2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                        LEV00390
C     3.  FORMAT(20A4) USED WITH THE EXPLANATORY HEADER, EXPLAN         LEV00400
C     4.  HEXADECIMAL FORMAT (4Z20) USED IN ALPHA/BETA FILES 1 AND 2.   LEV00410
C                                                                       LEV00420
C     IMPORTANT NOTE:  THIS PROGRAM ALLOWS ENLARGEMENT OF THE ALPHA,    LEV00430
C     BETA ARRAYS.  IN PARTICULAR, IF ANY ONE OF THE EIGENVALUES        LEV00440
C     SUPPLIED IS T-SIMPLE AND NOT CLOSE TO A SPURIOUS EIGENVALUE,      LEV00450
```

```
C     THE PROGRAM REQUIRES THAT KMAX BE AT LEAST 11*MEV/8 + 12.  IF
C     KMAX IS NOT THIS LARGE, THEN THE PROGRAM RESETS KMAX TO THIS
C     SIZE AND EXTENDS THE ALPHA, BETA HISTORY IF REQUIRED.
C     THUS, THE DIMENSIONS OF THE ALPHA AND BETA ARRAYS MUST BE
C     LARGE ENOUGH TO ALLOW FOR THIS POSSIBILITY.
C     REMEMBER THAT THE BETA ARRAY, BETA(J), IS SUCH THAT
C     J = 1,..., KMAX+1.  SO IF THE KMAX USED BY THE PROGRAM
C     IS TO BE 3000, THEN BETA MUST BE OF LENGTH AT LEAST 3001.
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  ALPHA(5000),BETA(5001)
      DOUBLE PRECISION  V1(5000),V2(5000)
      DOUBLE PRECISION  RITVEC(30000),TVEC(30000),GOODEV(50),EVNEW(50)
      DOUBLE PRECISION  EVAL,EVALN,TOLN,TTOL,ERTOL,ALFA,BATA
      DOUBLE PRECISION  MULTOL,SCALE0,STUTOL,BTOL,LB,UB
      DOUBLE PRECISION  ONE,ZERO,MACHEP,EPSM,TEMP,SUM,ERRMIN,BKMIN
      DOUBLE PRECISION  RELTOL,ERROR,TERROR,TLAST(50)
      REAL  G(10000),AMINGP(50),TMINGP(50),EXPLAN(20)
      REAL  TERR(50),ERR(50),ERRDGP(50),RNORM(50),TBETA(50)
      INTEGER  MP(50),M1(50),M2(50),MA(50),ML(50),MINT(50),MFIN(50)
      INTEGER  SVSEED,SVSOLD,RHSEED,IDELTA(50)
      INTEGER  MBOUND,NTVCON,SVTVEC,TVSTOP,LVCONT,ERCONT,TFLAG
      DOUBLE PRECISION  FINPRO
      DOUBLE PRECISION  DABS, DMAX1, DSQRT, DFLOAT
      REAL ABS
      INTEGER  IABS
C-----------------------------------------------------------------------
      EXTERNAL CMATV
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.D0*MACHEP
C-----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     1.  ALPHA:  >= KMAXN,  BETA: >= (KMAXN+1) WHERE KMAXN, THE
C                 LARGEST SIZE T-MATRIX CONSIDERED BY THE PROGRAM,
C                 IS THE LARGER OF THE SIZE OF THE ALPHA, BETA HISTORY
C                 PROVIDED ON FILE 2 (IF ANY ) AND THE SIZE WHICH THE
C                 PROGRAM SPECIFIES INTERNALLY, THIS LATTER IS ALWAYS
C                 < = 11*MEV / 8  +  12, WHERE MEV IS THE SIZE
C                 T-MATRIX THAT WAS USED IN THE CORRESPONDING EIGENVALUE
C                 COMPUTATIONS.
C     2.  V1:  >= MAX(N,KMAX)
C     3.  V2:   >= N
C     4.  G:  >= MAX(N,KMAX)
C     5.  RITVEC:  >= N*NGOOD, WHERE NGOOD IS NUMBER OF EIGENVALUES
C                   SUPPLIED TO THIS PROGRAM.
C     6.  TVEC:  >= CUMULATIVE LENGTH OF ALL THE T-EIGENVECTORS
C         NEEDED TO GENERATE THE DESIRED RITZ VECTORS.  AN EDUCATED
C         GUESS AT AN APPROPRIATE LENGTH CAN BE OBTAINED BY RUNNING THE
C         PROGRAM WITH THE FLAG MBOUND = 1 AND MULTIPLYING THE
C         RESULTING SIZE BY 5/4.
C     7.  GOODEV, AMINGP, TMINGP, TERR, ERR, ERRGDP, RNORM, TBETA,
C         TLAST, EVNEW, MP, MA, M1, M2, MINT, MFIN AND IDELTA ALL MUST
C         BE >= NGOOD.
C
```

```
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LANCZOS EIGENVECTOR PROCEDURE FOR REAL SYMMETRIC MATRICE
     1S'/)
C
C     SET PROGRAM PARAMETERS
C     USER MUST NOT MODIFY SCALE0
      SCALE0 = 5.0D0
      ZERO = 0.0D0
      ONE = 1.0D0
      MPMIN = -1000
C     SET CONVERGENCE CRITERION FOR T-EIGENVECTORS.
      ERTOL = 1.D-10
C
C     READ USER-SPECIFIED PARAMETER FROM INPUT FILE 5 (FREE FORMAT)
C
C     READ USER-PROVIDED HEADER FOR RUN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C     READ IN THE MAXIMUM PERMISSIBLE DIMENSIONS FOR THE TVEC ARRAY
C     (MDIMTV), FOR THE RITVEC ARRAY (MDIMRV), AND FOR THE BETA
C     ARRAY (MBETA).
      READ(5,20) EXPLAN
      READ(5,*) MDIMTV, MDIMRV, MBETA
C
C     READ IN RELATIVE TOLERANCE (RELTOL) USED IN DETERMINING
C     APPROPRIATE SIZES FOR THE T-MATRICES TO BE USED IN THE RITZ
C     VECTOR COMPUTATIONS.
      READ(5,20) EXPLAN
      READ(5,*) RELTOL
C
C     SET FLAGS TO 0 OR 1:
C     MBOUND = 1:  PROGRAM TERMINATES AFTER COMPUTING 1ST GUESSES
C                  ON APPROPRIATE T-SIZES FOR USE IN THE RITZ VECTOR
C                  COMPUTATIONS
C     NTVCON = 0:  PROGRAM TERMINATES IF THE TVEC ARRAY IS NOT
C                  LARGE ENOUGH TO HOLD ALL THE T-EIGENVECTORS REQUIRED.
C     SVTVEC = 0:  THE T-EIGENVECTORS ARE NOT WRITTEN TO FILE 11
C                  UNLESS TVSTOP = 1
C     SVTVEC = 1:  WRITE THE T-EIGENVECTORS TO FILE 11.
C     TVSTOP = 1:  PROGRAM TERMINATES AFTER COMPUTING THE
C                  T-EIGENVECTORS
C     LVCONT = 0:  PROGRAM TERMINATES IF THE NUMBER OF T-EIGENVECTORS
C                  COMPUTED IS NOT EQUAL TO THE NUMBER OF RITZ
C                  VECTORS REQUESTED.
C     ERCONT = 0:  MEANS FOR ANY GIVEN EIGENVALUE, A RITZ VECTOR
C                  WILL NOT BE COMPUTED FOR THAT EIGENVALUE UNLESS
C                  A T-EIGENVECTOR HAS BEEN IDENTIFIED WITH A LAST
C                  COMPONENT WHICH SATISFIES THE SPECIFIED
C                  CONVERGENCE CRITERION.
C     ERCONT = 1:  MEANS FOR ANY GIVEN EIGENVALUE, A RITZ VECTOR
C                  WILL BE COMPUTED.  IF A T-EIGENVECTOR CANNOT
```

```
C                 BE IDENTIFIED WHICH SATISFIES THE LAST
C                 COMPONENT CRITERION, THEN THE PROGRAM WILL
C                 USE THE T-VECTOR THAT CAME CLOSEST TO
C                 SATISFYING THE CRITERION.
C     IWRITE = 1: EXTENDED OUTPUT OF INTERMEDIATE COMPUTATIONS
C                 IS WRITTEN TO FILE 6
C     IREAD = 0:  ALPHA/BETA FILE IS REGENERATED.
C     IREAD = 1:  ALPHA/BETA FILE USED IN EIGENVALUE COMPUTATIONS
C                 IS READ IN AND EXTENDED IF NECESSARY.  IN BOTH
C                 CASES IREAD = 0 OR 1, THE LANCZOS VECTORS ARE
C                 ALWAYS REGENERATED FOR THE RITZ VECTOR
C                 COMPUTATIONS
C
      READ(5,20) EXPLAN
      READ(5,*) MBOUND,NTVCON,SVTVEC,IREAD
C
      READ(5,20) EXPLAN
      READ(5,*) TVSTOP,LVCONT,ERCONT,IWRITE
      IF (TVSTOP.EQ.1) SVTVEC = 1
C
C     READ IN SEED (RHSEED) FOR GENERATING RANDOM STARTING VECTOR
C     FOR INVERSE ITERATION ON THE T-MATRICES.
      READ(5,20) EXPLAN
      READ(5,*) RHSEED
C
C     READ IN MATNO = MATRIX/RUN IDENTIFICATION NUMBER AND
C     N = ORDER OF A-MATRIX
      READ(5,20) EXPLAN
      READ(5,*) MATNO,N
C
C-----------------------------------------------------------------------
C     INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED MATRIX
C     AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE
C     MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV.
C
      CALL USPEC(N,MATNO)
C
C-----------------------------------------------------------------------
C     MASK UNDERFLOW AND OVERFLOW
      CALL MASK
C
C-----------------------------------------------------------------------
C     WRITE RUN PARAMETERS OUT TO FILE 6
C
      WRITE(6,30) MATNO,N
   30 FORMAT(/' MATRIX IDENTIFICATION NO. = ',I10,' ORDER OF A = ',I5)
C
      WRITE(6,40) MBOUND,NTVCON,SVTVEC,IREAD
   40 FORMAT(/3X,'MBOUND',3X,'NTVCON',3X,'SVTVEC',3X,'IREAD'/3I9,I8)
C
      WRITE(6,50) TVSTOP,LVCONT,ERCONT,IWRITE
   50 FORMAT(/3X,'TVSTOP',3X,'LVCONT',3X,'ERCONT',3X,'IWRITE'/4I9)
C
      WRITE(6,60) MDIMTV,MDIMRV,MBETA
   60 FORMAT(/3X,'MDIMTV',3X,'MDIMRV',3X,'MBETA'/2I9,I8)
```

```
C                                                                       LEV02110
      WRITE(6,70) RELTOL,RHSEED                                         LEV02120
   70 FORMAT(/7X,'RELTOL',3X,'RHSEED'/E13.4,I9)                         LEV02130
C                                                                       LEV02140
C                                                                       LEV02150
C     FROM FILE 3 READ IN THE NUMBER OF EIGENVALUES (NGOOD) FOR WHICH   LEV02160
C     EIGENVECTORS ARE REQUESTED, THE ORDER (MEV) OF THE LANCZOS        LEV02170
C     TRIDIAGONAL MATRIX USED IN COMPUTING THESE EIGENVALUES, THE       LEV02180
C     ORDER (NOLD) OF THE USER-SPECIFIED MATRIX USED IN THE EIGENVALUE  LEV02190
C     COMPUTATIONS, THE SEED (SVSEED) USED FOR GENERATING THE STARTING  LEV02200
C     VECTOR THAT WAS USED IN THOSE LANCZOS EIGENVALUE COMPUTATIONS,    LEV02210
C     AND THE MATRIX/RUN IDENTIFICATION NUMBER (MATOLD) USED IN THOSE   LEV02220
C     COMPUTATIONS.  ALSO READ IN THE NUMBER (NDIS) OF DISTINCT         LEV02230
C     EIGENVALUES OF T(1,MEV) THAT WERE COMPUTED BUT THIS VALUE IS      LEV02240
C     NOT USED IN THE EIGENVECTOR COMPUTATIONS.                         LEV02250
C                                                                       LEV02260
      READ(3,80) NGOOD,NDIS,MEV,NOLD,SVSEED,MATOLD                      LEV02270
   80 FORMAT(4I6,I12,I8)                                                LEV02280
C                                                                       LEV02290
C     READ IN THE T-MULTIPLICITY TOLERANCE USED IN THE BISEC SUBROUTINE LEV02300
C     DURING THE COMPUTATION OF THE GIVEN EIGENVALUES.                  LEV02310
C     ALSO READ IN THE FLAG IB.  IF IB < 0, THEN SOME BETA(I) IN THE    LEV02320
C     T-MATRIX FILE PROVIDED ON FILE 2 FAILED THE ORTHOGONALITY         LEV02330
C     TEST IN THE TNORM SUBROUTINE.  USER SHOULD NOTE THAT THIS VECTOR  LEV02340
C     PROGRAM PROCEEDS INDEPENDENTLY OF THE SIZE OF THE BETA USED.      LEV02350
C                                                                       LEV02360
      READ(3,90) MULTOL,IB,BTOL                                         LEV02370
   90 FORMAT(E20.12,I6,E13.4)                                           LEV02380
C                                                                       LEV02390
      TEMP = DFLOAT(MEV+1000)                                           LEV02400
      TTOL = MULTOL/TEMP                                                LEV02410
      WRITE(6,100) MULTOL,TTOL                                          LEV02420
  100 FORMAT(/' T-MULTIPLICITY TOLERANCE USED IN THE EIGENVALUE COMPUTATLEV02430
     1IONS WAS',E13.4/' SCALED MACHINE EPSILON IS',E13.4)               LEV02440
C                                                                       LEV02450
C     CONTINUE WRITE TO FILE 6 OF THE PARAMETERS FOR THIS RUN           LEV02460
C                                                                       LEV02470
      WRITE(6,110)NGOOD,NDIS,MEV,NOLD,MATOLD,SVSEED,MULTOL,IB,BTOL      LEV02480
  110 FORMAT(/' EIGENVALUES SUPPLIED ARE READ IN FROM FILE 3'/' FILE 3  LEV02490
     1HEADER IS'/4X,'NG',2X,'NDIS',3X,'MEV',2X,'NOLD',2X,'MATOLD',4X,   LEV02500
     1'SVSEED',6X,'MULTOL',6X,'IB',9X,'BTOL'/4I6,I8,I10,E12.3,I8,E13.4/)LEV02510
C                                                                       LEV02520
C     IS THE ARRAY RITVEC LONG ENOUGH TO HOLD ALL OF THE DESIRED        LEV02530
C     RITZ VECTORS (APPROXIMATE EIGENVECTORS)?                          LEV02540
      NMAX = NGOOD*N                                                    LEV02550
      IF(MBOUND.NE.0) GO TO 120                                         LEV02560
      IF(TVSTOP.NE.1.AND.NMAX.GT.MDIMRV) GO TO 1310                     LEV02570
C                                                                       LEV02580
C     CHECK THAT THE ORDER N AND THE MATRIX IDENTIFICATION NUMBER       LEV02590
C     MATNO SPECIFIED BY THE USER AGREE WITH THOSE READ IN FROM         LEV02600
C     FILE 3.                                                           LEV02610
  120 ITEMP = (NOLD-N)**2+(MATOLD-MATNO)**2                             LEV02620
      IF (ITEMP.NE.0) GO TO 1330                                        LEV02630
C                                                                       LEV02640
C     READ IN FROM FILE 3, THE T-MULTIPLICITIES OF THE EIGENVALUES      LEV02650
```

```
C     WHOSE EIGENVECTORS ARE TO BE COMPUTED, THE VALUES OF THESE
C     EIGENVALUES AND THEIR MINIMAL GAPS AS EIGENVALUES OF THE
C     USER-SPECIFIED MATRIX AND AS EIGENVALUES OF THE T-MATRIX.
C
      READ(3,20) EXPLAN
      READ(3,130) (MP(J),GOODEV(J),TMINGP(J),AMINGP(J), J=1,NGOOD)
  130 FORMAT(6X,I6,E25.16,2E14.3)
C
      WRITE(6,140) (J,GOODEV(J),MP(J),TMINGP(J),AMINGP(J), J=1,NGOOD)
  140 FORMAT(/' EIGENVALUES READ IN, T-MULTIPLICITIES, T-GAPS AND A-GAPS
     1 '/4X,' J ',5X,'GOOD EIGENVALUE',5X,'MULT',4X,'  TMINGAP ',4X,
     1'  AMINGAP '/(I6,E25.16,I4,2E15.4))
C
C     READ IN ERROR ESTIMATES
      WRITE(6,150) MEV,SVSEED
  150 FORMAT(/' THESE EIGENVALUES WERE COMPUTED USING A T-MATRIX OF
     1ORDER ',I5/' AND SEED FOR RANDOM NUMBER GENERATOR =',I12)
C     CHECK WHETHER OR NOT THERE ARE ANY T-ISOLATED EIGENVALUES IN
C     THE EIGENVALUES PROVIDED
      DO 160 J=1,NGOOD
      IF(MP(J).EQ.1) GO TO 170
  160 CONTINUE
      GO TO 190
  170 READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,180) NISO
  180 FORMAT(18X,I6)
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
  190 DO 220 J=1,NGOOD
      ERR(J) = 0.D0
      IF(MP(J).NE.1) GO TO 220
      READ(4,200) EVAL, ERR(J)
  200 FORMAT(10X,E25.16,E14.3)
      IF(DABS(EVAL - GOODEV(J)).LT.1.D-10) GO TO 220
      WRITE(6,210) EVAL,GOODEV(J)
  210 FORMAT(' PROBLEM WITH READ IN OF ERROR ESTIMATES'/' EIGENVALUE REA
     1D IN',E20.12,' DOES NOT MATCH GOODEV(J) ='/E20.12)
      GO TO 1550
C
  220 CONTINUE
C
      WRITE(6,230) (J,GOODEV(J),ERR(J), J=1,NGOOD)
  230 FORMAT(' ERROR ESTMATES ='/4X,' J',5X,'EIGENVALUE',10X,' ESTIMATE
     1'/(I6,E20.12,E14.3))
C
      IF(IREAD.EQ.0)  GO TO 330
C
C     READ IN THE SIZE OF THE T-MATRIX PROVIDED ON FILE 2.  READ IN
C     THE ORDER OF THE USER-SPECIFIED MATRIX , THE SEED FOR THE
C     RANDOM NUMBER GENERATOR, AND THE MATRIX/TEST IDENTIFICATION
C     NUMBER THAT WERE USED IN THE LANCZOS EIGENVALUE COMPUTATIONS.
C     THESE ARE USED IN A CONSISTENCY CHECK
```

```
C     IF FLAG IREAD = 0 REGENERATE ALPHA, BETA
C
      READ(2,240) KMAX,NOLD,SVSOLD,MATOLD
  240 FORMAT(2I6,I12,I8)
C
      WRITE(6,250) KMAX,NOLD,SVSOLD,MATOLD
  250 FORMAT(/' READ IN THE T-MATRICES STORED ON FILE 2'/' FILE 2 HEADER
     1 IS'/2X,'KMAX',2X,'NOLD',6X,'SVSOLD',2X,'MATOLD'/2I6,I12,I8/)
C
C     CHECK THAT THE ORDER, THE MATRIX/TEST IDENTIFICATION NUMBER
C     AND THE SEED FOR THE RANDOM NUMBER GENERATOR USED IN THE
C     LANCZOS COMPUTATIONS THAT GENERATED THE ALPHA,BETA FILE
C     BEING USED AGREE WITH WHAT THE USER HAS SPECIFIED.
      IF (NOLD.NE.N.OR.MATOLD.NE.MATNO.OR.SVSOLD.NE.SVSEED) GO TO 1350
C
      KMAX1 = KMAX + 1
C
C     READ IN THE T-MATRICES FROM FILE 2.  THESE ARE USED TO GENERATE
C     THE T-EIGENVECTORS THAT WILL BE USED IN THE RITZ VECTOR
C     COMPUTATIONS.  HISTORY MUST BE STORED IN MACHINE FORMAT
C     ((4Z20) FOR IBM/3081)
C
      READ(2,260) (ALPHA(J), J=1,KMAX)
      READ(2,260) (BETA(J), J=1,KMAX1)
  260 FORMAT(4Z20)
C
      READ(2,260) (V1(J), J=1,N)
      READ(2,260) (V2(J), J=1,N)
C
C     KMAX MAY BE ENLARGED IF THE SIZE AT WHICH THE EIGENVALUE
C     COMPUTATIONS WERE PERFORMED IS ESSENTIALLY KMAX AND
C     THERE IS AT LEAST ONE EIGENVALUE THAT IS T-SIMPLE AND
C     T-ISOLATED, IN THE SENSE THAT IF ITS NEAREST NEIGHBOR IS TOO
C     CLOSE THAT NEIGHBOR IS A 'GOOD' T-EIGENVALUE.
      DO 270 J = 1,NGOOD
      IF(MP(J).EQ.1) GO TO 290
  270 CONTINUE
      WRITE(6,280)
  280 FORMAT(/' ALL EIGENVALUES USED ARE T-MULTIPLE OR CLOSE TO SPURIOUS
     1 T-EIGENVALUES'/' SO KMAX IS NOT INCREASED')
      IF(KMAX.LT.MEV) GO TO 1370
      GO TO 310
C
  290 KMAXN= 11*MEV/8 + 12
      IF(MBETA.LE.KMAXN) GO TO 1530
      IF(KMAX.GE.KMAXN )  GO TO 310
      WRITE(6,300) KMAX, KMAXN
  300 FORMAT(' ENLARGE KMAX FROM ',I6,' TO ',I6)
      MOLD1 = KMAX + 1
      KMAX = KMAXN
      GO TO 380
C
  310 WRITE(6,320) KMAX
  320 FORMAT(/' T-MATRICES HAVE BEEN READ IN FROM FILE 2'/' THE LARGEST
     1SIZE T-MATRIX ALLOWED IS',I6/)
```

```
C
      IF(IREAD.EQ.1) GO TO 400
C
C     REGENERATE THE ALPHA AND BETA
C
  330 MOLD1 = 1
C
      DO 340 J = 1,NGOOD
      IF(MP(J).EQ.1) GO TO 360
  340 CONTINUE
      KMAX = MEV + 12
      WRITE(6,350) KMAX
  350 FORMAT(/' ALL EIGENVALUES FOR WHICH EIGENVECTORS ARE TO BE COMPUTE
     1D ARE EITHER T-MULTIPLE OR CLOSE TO'/' A SPURIOUS T-EIGENVALUE. TH
     1EREFORE SET KMAX = MEV + 12 = ',I7)
      GO TO 380
C
  360 KMAXN = 11*MEV/8 + 12
      IF(MBETA.LE.KMAXN) GO TO 1530
      WRITE(6,370) KMAXN
  370 FORMAT(' SET KMAX EQUAL TO ',I6)
      KMAX = KMAXN
C
  380 WRITE(6,390) MOLD1,KMAX
  390 FORMAT(/' LANCZS SUBROUTINE GENERATES ALPHA(J), BETA(J+1), J =',
     1 I6,' TO ', I6/)
C
C-----------------------------------------------------------------------
C
      IIX = SVSEED
      CALL LANCZS(CMATV,ALPHA,BETA,V1,V2,G,KMAX,MOLD1,N,IIX)
C
C-----------------------------------------------------------------------
C
  400 CONTINUE
C
C     THE SUBROUTINE STURMI DETERMINES THE SMALLEST SIZE T-MATRIX FOR
C     WHICH THE EIGENVALUE IN QUESTION IS A T-EIGENVALUE (TO WITHIN A
C     GIVEN TOLERANCE) AND IF POSSIBLE THE SMALLEST SIZE T-MATRIX
C     FOR WHICH IT IS A DOUBLE T-EIGENVALUE (TO WITHIN THE SAME
C     TOLERANCE).  THE SIZE T-MATRIX USED IN THE RITZ VECTOR
C     COMPUTATIONS IS THEN DETERMINED BY LOOPING ON SIZE OF THE
C     T-EIGENVECTORS STARTING WITH A T-SIZE DETERMINED FROM THE
C     OUTPUT FROM STURMI.
C
C
      STUTOL = SCALE0*MULTOL
      IF(IWRITE.EQ.1) WRITE(6,410)
  410 FORMAT(' FROM STURMI')
      DO 450 J = 1,NGOOD
      EVAL = GOODEV(J)
C     COMPUTE THE TOLERANCES USED BY STURMI TO DETERMINE AN INTERVAL
C     CONTAINING THE EIGENVALUE EVAL.
      TEMP = DABS(EVAL)*RELTOL
      TOLN = DMAX1(TEMP,STUTOL)
```

```
C
C-----------------------------------------------------------------------
C
      CALL STURMI(ALPHA,BETA,EVAL,TOLN,EPSM,KMAX,MK1,MK2,IC,IWRITE)
C
C-----------------------------------------------------------------------
C
C     STORE THE COMPUTED ORDERS OF T-MATRICES FOR LATER PRINTOUT
      M1(J) = MK1
      M2(J) = MK2
      ML(J) = (MK1 + 3*MK2)/4
      IF(MK2.EQ.KMAX)  ML(J) = KMAX
C
      IF(IC.GT.0) GO TO 430
C     IC = 0 MEANS THERE WAS NO EIGENVALUE IN THE DESIGNATED INTERVAL
C     BY T-SIZE KMAX.  THIS MEANS THAT THE EIGENVALUE PROVIDED HAS
C     NOT YET CONVERGED SO ITS EIGENVECTOR WILL NOT BE COMPUTED.
      WRITE(6,420) J,GOODEV(J),MK1,MK2
  420 FORMAT(I6,'TH EIGENVALUE',E20.12,' HAS NOT CONVERGED '/
     1' SO DO NOT COMPUTE ANY T-EIGENVECTOR OR RITZ VECTOR FOR IT'
     1/' MK1 AND MK2 FOR THIS EIGENVALUE WERE',2I6)
      MP(J) = MPMIN
      MA(J) = -2*KMAX
      GO TO 450
C     COMPUTE AN APPROPRIATE SIZE T-MATRIX FOR THE GIVEN EIGENVALUE.
  430 IF(M2(J).EQ.KMAX) GO TO 440
C     M1 AND M2 WERE BOTH DETERMINED
      MA(J) = (3*M1(J) + M2(J))/4  + 1
      GO TO 450
C     M2 NOT DETERMINED
  440 MA(J) = (5*M1(J))/4  + 1
C
  450 CONTINUE
C
      IF (IWRITE.EQ.1) WRITE(6,460) (MA(JJ), JJ=1,NGOOD)
  460 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZE T-MATRICES'/
     1 ' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/(13I6))
C
C     PRINT OUT TO FILE 10 1ST GUESSES AT SIZES OF THE T-MATRICES TO
C     BE USED IN THE EIGENVECTOR COMPUTATIONS.
C     PROGRAM LOOPS ON T-SIZE TO DETERMINE APPROPRIATE SIZE T-MATRIX.
      WRITE(10,470) N,KMAX
  470 FORMAT(2I8,' = ORDER OF USER MATRIX AND MAX ORDER OF T(1,MEV)')
C
      WRITE(10,480)
  480 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZE T-MATRICES'/
     1 ' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/)
C
      WRITE(10,490)
  490 FORMAT(4X,'J',4X,'A-EIGENVALUE',4X,'M1(J)',1X,'M2(J)',1X,'MA(J)')
C
      WRITE(10,500) (J,GOODEV(J),M1(J),M2(J), MA(J), J=1,NGOOD)
  500 FORMAT(I5,E19.12,3I6)
C
      IF(MBOUND.EQ.1) WRITE(10,510)
```

```
  510 FORMAT(/'  EV = GOODEV(J) IS A GOOD EIGENVALUE OF T(1,MEV)'/
     1' M1 = SMALLEST VALUE OF M SUCH THAT T(1,M) HAS AT LEAST'/
     1 '     ONE EIGENVALUE IN THE INTERVAL (EV-TOLN,EV+TOLN)'/
     1 '     TOLN(J) = DMAX1(GOODEV(J)*RELTOL, SCALE0*MULTOL)'/
     1 ' M2 = SMALLEST M (IF ANY) SUCH THAT IN THE ABOVE INTERVAL'/
     1 '      T(1,M) HAS AT LEAST TWO EIGENVALUES '/
     1 ' IABS(MA(J)) = APPROPRIATE SIZE T-MATRIX FOR GOODEV(J)'/
     1 ' INITIAL VALUE OF MA(J) IS CHOSEN HEURISTICALLY'/
     1 ' PROGRAM LOOPS ON SIZE OF T-MATRIX TO GET BETTER SIZE'/
     1 ' END OF SIZES OF T-MATRICES FILE 10'///)
C
C     TERMINATE AFTER COMPUTING 1ST GUESSES AT SIZES OF THE
C     T-MATRICES REQUIRED FOR THE GIVEN EIGENVALUES?
      IF(MBOUND.EQ.1) GO TO 1390
C
C     IS THERE ROOM FOR ALL OF THE REQUESTED T-EIGENVECTORS?
      MTOL = 0
      DO 520 J = 1,NGOOD
      IF(MP(J).EQ.MPMIN) GO TO 520
      MTOL = MTOL + IABS(MA(J))
  520 CONTINUE
      MTOL = (5*MTOL)/4
      IF(MTOL.GT.MDIMTV.AND.NTVCON.EQ.0) GO TO 1410
C
C-----------------------------------------------------------------------
C     GENERATE A RANDOM VECTOR TO BE USED REPEATEDLY BY
C     SUBROUTINE INVERM
C
      CALL GENRAN(RHSEED,G,KMAX)
C
C----------------------------------------------------------------------
C
C     LOOP ON GIVEN EIGENVALUES TO COMPUTE THE CORRESPONDING
C     T-EIGENVECTOR.
C
      MTOL = 0
      NTVEC = 0
      ILBIS = 0
      DO 710 J = 1,NGOOD
      ICOUNT = 0
      ERRMIN = 10.D0
      MABEST = MPMIN
      IF(MP(J).EQ.MPMIN) GO TO 710
      TFLAG = 0
      EVAL = GOODEV(J)
      TEMP = DABS(EVAL)*RELTOL
      UB = EVAL + DMAX1(STUTOL,TEMP)
      LB = EVAL - DMAX1(STUTOL,TEMP)
  530 KMAXU = IABS(MA(J))
C
C     SELECT A SUITABLE INCREMENT FOR THE ORDERS OF THE T-MATRICES
C     TO BE CONSIDERED IN DETERMINING APPROPRIATE SIZES FOR THE RITZ
C     VECTOR COMPUTATIONS.
      IF(ICOUNT.GT.0) GO TO 550
C     SELECT IDELTA(J) BASED UPON THE T-MULTIPLICITY OBTAINED
```

```
      IF(M2(J).EQ.KMAX) GO TO 540
C     M2 DETERMINED
      IDELTA(J) = ((3*M1(J) + 5*M2(J))/8  +  1 - IABS(MA(J)))/10  +  1
      GO TO 550
C     M2 NOT DETERMINED
  540 MAMAX = MIN0((11*MEV)/8 + 12, (13*M1(J))/8 + 1)
      IDELTA(J) = (MAMAX - IABS(MA(J)))/10  +  1
  550 ICOUNT = ICOUNT + 1
C
C-----------------------------------------------------------------------
C     TO MIMIMIZE THE EFFECT OF THE ONE-SIDED ACCEPTANCE TEST FOR
C     EIGENVALUES IN THE BISEC SUBROUTINE, RECOMPUTE THE GIVEN
C     EIGENVALUE AT THE SPECIFIED KMAXU
C
      CALL LBISEC(ALPHA,BETA,EPSM,EVAL,EVALN,LB,UB,TTOL,KMAXU,NEVT)
C
C-----------------------------------------------------------------------
C
C     CHECK WHETHER OR NOT GIVEN T-MATRIX HAS AN EIGENVALUE IN THE
C     SPECIFIED INTERVAL AND IF SO WHAT ITS T-MULTIPLICITY IS.
C
      IF(NEVT.EQ.1) GO TO 590
      IF(NEVT.NE.0) GO TO 570
      ILBIS = 1
      WRITE(6,560) EVAL,KMAXU
  560 FORMAT(/' PROBLEM ENCOUNTERED IN RECOMPUTATION OF USER-SUPPLIED EI
     1GENVALUE',E20.12/' THE SIZE T-MATRIX SPECIFIED',I6,' DOES NOT
     1HAVE AN EIGENVALUE IN THE INTERVAL SPECIFIED'/' THEREFORE NO EIGEN
     1VECTOR WILL BE COMPUTED FOR THIS PARTICULAR EIGENVALUE'/)
      GO TO 610
C
  570 IF(NEVT.GT.1)  WRITE(6,580) EVAL,KMAXU
  580 FORMAT(/' PROBLEM ENCOUNTERED IN RECOMPUTATION OF USER-SUPPLIED
     1EIGENVALUE',E20.12/' FOR THE SIZE T-MATRIX SPECIFIED =',I6,' THE
     1GIVEN EIGENVALUE IS T-MULTIPLE IN THE INTERVAL SPECIFIED'/' SOMETH
     1ING IS WRONG, THEREFORE NO EIGENVECTOR WILL BE COMPUTED FOR THIS E
     1IGENVALUE'/)
C
      MP(J) = MPMIN
      MA(J) = -2*KMAX
      GO TO 710
C
  590 CONTINUE
      ILBIS = 0
C
      EVNEW(J) = EVALN
      EVAL = EVALN
      MTOL = MTOL+KMAXU
C
C     IS THERE ROOM IN TVEC ARRAY FOR THE NEXT T-EIGENVECTOR?
C     IF NOT, SKIP TO RITZ VECTOR COMPUTATIONS.
      IF (MTOL.GT.MDIMTV) GO TO 720
C
      IT = 3
      KINT = MTOL - KMAXU +1
```

```
C
C     RECORD THE BEGINNING AND END OF THE T-EIGENVECTOR BEING COMPUTED
      MINT(J) = KINT
      MFIN(J) = MTOL
C
C-----------------------------------------------------------------------
C     SUBROUTINE INVERM DOES INVERSE ITERATION, I.E. SOLVES
C     (T(1,KMAXU) - EVAL)*U = RHS  FOR EACH EIGENVALUE TO OBTAIN THE
C     DESIRED T-EIGENVECTOR.
C
      IF(IWRITE.EQ.1)  WRITE(6,600) J
  600 FORMAT(/I6,'TH EIGENVALUE')
C
      CALL INVERM(ALPHA,BETA,V1,TVEC(KINT),EVAL,ERROR,TERROR,EPSM,
     1 G,KMAXU,IT,IWRITE)
C
C-----------------------------------------------------------------------
C
      TERR(J) = TERROR
      TLAST(J) = ERROR
      KMAXU1 = KMAXU + 1
      TBETA(J) = BETA(KMAXU1)*ERROR
C
C     AFTER EACH OF THE T-EIGENVECTORS IS COMPUTED, THE
C     SIZE OF THE ERROR ESTIMATE, ERROR IS CHECKED.
C     IF THIS ESTIMATE IS NOT AS SMALL AS DESIRED AND
C     |MA(J)| < ML(J), PROGRAM ATTEMPTS TO INCREASE THE SIZE OF |MA(J)|
C     AND REPEAT THE T-EIGENVECTOR COMPUTATIONS.
C
      IF(ERROR.LT.ERTOL.OR.TFLAG.EQ.1)  GO TO 700
C
      IF(ERROR.GE.ERRMIN) GO TO 610
C     LAST COMPONENT IS LESS THAN MINIMAL TO DATE
      ERRMIN = ERROR
      MABEST = MA(J)
  610 CONTINUE
C
      IF(MA(J).GT.0)  ITEST = MA(J) + IDELTA(J)
      IF(MA(J).LT.0)  ITEST = -(IABS(MA(J)) + IDELTA(J))
      IF(IABS(ITEST).LE.ML(J).AND.ICOUNT.LE.10) GO TO 630
C     NEW MA(J) IS GREATER THAN MAXIMUM ALLOWED.
      IF(ERCONT.EQ.0.OR.MABEST.EQ.MPMIN) GO TO 650
      TFLAG = 1
      MA(J) = MABEST
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU
      WRITE(6,620) MA(J)
  620 FORMAT(' 10 ORDERS WERE CONSIDERED.  NONE SATISFIED THE ERROR TEST
     1'/' THEREFORE USE THE BEST ORDER OBTAINED FOR THE EIGENVECTORS'
     1,I6)
      GO TO 530
C
  630 MA(J) = ITEST
C
      MT = IABS(MA(J))
      IF(IWRITE.EQ.1) WRITE(6,640)  MT
```

```
  640 FORMAT(/' CHANGE SIZE OF T-MATRIX TO ',I6,' RECOMPUTE T-EIGENVECTO
     1R')
C
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU
C
      GO TO 530
C
C     APPROPRIATE SIZE T-MATRIX WAS NOT OBTAINED
  650 CONTINUE
      WRITE(10,660) J,EVAL,MP(J)
  660 FORMAT(/' ON 10 INCREMENTS NOT ABLE TO IDENTIFY APPROPRIATE SIZE
     1T-MATRIX FOR'/
     1' EIGENVALUE(',I4,') = ',E20.12,' T-MULTIPLICITY =',I4/)
      IF(M2(J).EQ.KMAX) WRITE(10,670)
      IF(M2(J).LT.KMAX) WRITE(10,680)
  670 FORMAT(/' ORDERS TESTED RANGED FROM 5*M1(J)/4 TO APPROXIMATELY
     1 '/'  MIN(11*MEV/8,13*M1(J)/8)'/)
  680 FORMAT(/' ORDERS TESTED RANGED FROM (3*M1(J)+M2(J))/4 TO APPROXIMA
     1TELY'/'  (3*M1(J) + 5*M2(J))/8.'/)
      WRITE(10,690)
  690 FORMAT(' ALLOWING LARGER ORDERS FOR THE T-MATRICES MAY RESULT IN
     1 SUCCESS'/' BUT PROBABLY WILL NOT.  PROBLEM IS PROBABLY DUE TO'
     1 /' LACK OF CONVERGENCE OF GIVEN EIGENVALUE, CHECK THE ERROR ESTIM
     1ATE'/)
      MP(J) = MPMIN
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU
      GO TO 710
  700 NTVEC = NTVEC + 1
C
  710 CONTINUE
      NGOODC = NGOOD
      GO TO 740
C
C     COME HERE IF THERE IS NOT ENOUGH ROOM FOR ALL OF T-EIGENVECTORS
  720 NGOODC = J-1
      WRITE(6,730)  J, MTOL, MDIMTV
  730 FORMAT(/'  NOT ENOUGH ROOM IN TVEC FOR ',I4,'TH T-VECTOR'/'  T-DIM
     1ENSION REQUESTED = ',I6,' BUT TVEC HAS DIMENSION = ',I6/)
      IF(NGOODC.EQ.0)  GO TO 1430
      MTOL = MTOL-KMAXU
C
  740 CONTINUE
C
C     THE LOOP ON T-EIGENVECTOR COMPUTATIONS IS COMPLETE.
C     WRITE OUT THE SIZE T-MATRICES THAT WILL BE USED FOR
C     THE RITZ VECTOR COMPUTATIONS.
C
      WRITE(10,750)
  750 FORMAT(/' SIZES OF T-MATRICES THAT WILL BE USED IN THE RITZ COMPUT
     1ATIONS'/5X,'J',16X,'GOODEV(J)',1X,'MA(J)')
C
      WRITE(10,760)   (J,GOODEV(J),MA(J), J=1,NGOOD)
  760 FORMAT(I6,E25.14,I6)
      WRITE(10,510)
C
```

```
      WRITE(6,770) MTOL                                                 LEV07060
  770 FORMAT(/' THE CUMULATIVE LENGTH OF THE T-EIGENVECTORS IS',I18)    LEV07070
C                                                                       LEV07080
      WRITE(6,780) NTVEC,NGOOD                                          LEV07090
  780 FORMAT(/I6,' T-EIGENVECTORS OUT OF',I6,' REQUESTED WERE COMPUTED')LEV07100
C                                                                       LEV07110
C     SAVE THE T-EIGENVECTORS ON FILE 11?                               LEV07120
      IF(TVSTOP.NE.1.AND.SVTVEC.EQ.0) GO TO 840                         LEV07130
C                                                                       LEV07140
      WRITE(11,790) NTVEC,MTOL,MATNO,SVSEED                             LEV07150
  790 FORMAT(I6,3I12,' = NTVEC,MTOL,MATNO,SVSEED')                      LEV07160
C                                                                       LEV07170
      DO 820 J=1,NGOODC                                                 LEV07180
C     IF MP(J) = MPMIN THEN NO SUITABLE T-EIGENVECTOR IS AVAILABLE      LEV07190
C     FOR THAT EIGENVALUE.                                              LEV07200
      IF(MP(J).EQ.MPMIN) WRITE(11,800) J,MA(J),GOODEV(J),MP(J)          LEV07210
  800 FORMAT(2I6,E20.12,I6/' TH EIGVAL,T-SIZE,EVALUE,FLAG,NO EIGVEC')   LEV07220
      IF(MP(J).NE.MPMIN) WRITE(11,810) J,MA(J),GOODEV(J),MP(J)          LEV07230
  810 FORMAT(I6,I6,E20.12,I6/' T-EIGVEC,SIZE T,EVALUE OF A,MP(J)')      LEV07240
      IF(MP(J).EQ.MPMIN) GO TO 820                                      LEV07250
      KI = MINT(J)                                                      LEV07260
      KF = MFIN(J)                                                      LEV07270
C                                                                       LEV07280
      WRITE(11,260) (TVEC(K), K=KI,KF)                                  LEV07290
C                                                                       LEV07300
  820 CONTINUE                                                          LEV07310
C                                                                       LEV07320
      IF(TVSTOP.NE.1) GO TO 840                                         LEV07330
C                                                                       LEV07340
      WRITE(6,830) TVSTOP, NTVEC,NGOOD                                  LEV07350
  830 FORMAT(/' USER SET TVSTOP = ',I1/                                 LEV07360
     1' THEREFORE PROGRAM TERMINATES AFTER T-EIGENVECTOR COMPUTATIONS'/ LEV07370
     1' T-EIGENVECTORS THAT WERE COMPUTED ARE SAVED ON FILE 11'/        LEV07380
     1I8,' T-EIGENVECTORS WERE COMPUTED OUT OF',I7,' REQUESTED'/)       LEV07390
C                                                                       LEV07400
      GO TO 1550                                                        LEV07410
C                                                                       LEV07420
  840 CONTINUE                                                          LEV07430
C     IF NOT ABLE TO COMPUTE ALL THE REQUESTED T-EIGENVECTORS,          LEV07440
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS ANYWAY?             LEV07450
      IF(NTVEC.NE.NGOOD.AND.LVCONT.EQ.0) GO TO 1450                     LEV07460
C                                                                       LEV07470
C     COMPUTE THE MAXIMUM SIZE OF THE T-MATRIX USED FOR THOSE           LEV07480
C     EIGENVALUES WITH GOOD ERROR ESTIMATES.                            LEV07490
C                                                                       LEV07500
      KMAXU = 0                                                         LEV07510
      DO 850 J = 1,NGOODC                                               LEV07520
      MT = IABS(MA(J))                                                  LEV07530
      IF(MT.LT.KMAXU.OR.MP(J).EQ.MPMIN) GO TO 850                       LEV07540
      KMAXU = MT                                                        LEV07550
  850 CONTINUE                                                          LEV07560
C                                                                       LEV07570
      IF(KMAXU.EQ.0) GO TO 1490                                         LEV07580
C                                                                       LEV07590
      WRITE(6,860) KMAXU                                                LEV07600
```

```
  860 FORMAT(/I6,' = LARGEST SIZE T-MATRIX TO BE USED IN THE RITZ VECTORLEV07610
     1 COMPUTATIONS')                                                   LEV07620
C                                                                       LEV07630
C     COUNT THE NUMBER OF RITZ VECTORS NOT BEING COMPUTED               LEV07640
      MREJEC = 0                                                        LEV07650
      DO 870 J=1,NGOODC                                                 LEV07660
  870 IF(MP(J).EQ.MPMIN)  MREJEC = MREJEC + 1                           LEV07670
      MREJET = MREJEC + (NGOOD-NGOODC)                                  LEV07680
      IF(MREJET.NE.0) WRITE(6,880) MREJET                               LEV07690
  880 FORMAT(/' RITZ VECTORS ARE NOT COMPUTED FOR',I6,' OF THE EIGENVALULEV07700
     1ES'/)                                                             LEV07710
      NACT = NGOODC - MREJEC                                            LEV07720
      WRITE(6,890) NGOOD,NTVEC,NACT                                     LEV07730
  890 FORMAT(/I6,' RITZ VECTORS WERE REQUESTED'/I6,' T-EIGENVECTORS WERELEV07740
     1 COMPUTED'/I6,' RITZ VECTORS WILL BE COMPUTED'/)                  LEV07750
C     CHECK IF THERE ARE ANY RITZ VECTORS TO COMPUTE                    LEV07760
      IF(MREJEC.EQ.NGOODC) GO TO 1470                                   LEV07770
C                                                                       LEV07780
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS?                    LEV07790
      IF(LVCONT.EQ.0.AND.MREJEC.NE.0) GO TO 1450                        LEV07800
C                                                                       LEV07810
C     NOW COMPUTE THE RITZ VECTORS.  REGENERATE THE                     LEV07820
C     LANCZOS VECTORS.                                                  LEV07830
C                                                                       LEV07840
      DO 900 I = 1,NMAX                                                 LEV07850
  900 RITVEC(I) = ZERO                                                  LEV07860
C                                                                       LEV07870
C-----------------------------------------------------------------------LEV07880
C     REGENERATE THE STARTING VECTOR. THIS MUST BE GENERATED AND        LEV07890
C     NORMALIZED PRECISELY THE WAY IT WAS DONE IN THE EIGENVALUE        LEV07900
C     COMPUTATIONS, OTHERWISE THERE WILL BE A MISMATCH BETWEEN          LEV07910
C     THE T-EIGENVECTORS THAT HAVE BEEN COMPUTED FROM THE T-MATRICES    LEV07920
C     READ IN FROM FILE 2 AND THE LANCZOS VECTORS THAT ARE              LEV07930
C     BEING REGENERATED.                                                LEV07940
C                                                                       LEV07950
      IIL = SVSEED                                                      LEV07960
      CALL GENRAN(IIL,G,N)                                              LEV07970
C                                                                       LEV07980
C-----------------------------------------------------------------------LEV07990
C                                                                       LEV08000
      DO 910 J = 1,N                                                    LEV08010
  910 V2(J) = G(J)                                                      LEV08020
C                                                                       LEV08030
      SUM = FINPRO(N,V2(1),1,V2(1),1)                                   LEV08040
      SUM = ONE/DSQRT(SUM)                                              LEV08050
C                                                                       LEV08060
      DO 920 J = 1,N                                                    LEV08070
      V1(J) = ZERO                                                      LEV08080
  920 V2(J) = V2(J)*SUM                                                 LEV08090
C                                                                       LEV08100
C     LOOP FOR GENERATING RITZ VECTORS  (IVEC = 1,KMAXU)                LEV08110
      IVEC = 1                                                          LEV08120
      BATA = ZERO                                                       LEV08130
C                                                                       LEV08140
      GO TO 980                                                         LEV08150
```

```
C                                                                       LEV08160
  930 CONTINUE                                                          LEV08170
C                                                                       LEV08180
C     COMPUTE V1 = A*V2 - BATA*V1                                       LEV08190
C                                                                       LEV08200
C-----------------------------------------------------------------------LEV08210
C                                                                       LEV08220
      CALL CMATV(V2,V1,BATA)                                            LEV08230
C                                                                       LEV08240
C-----------------------------------------------------------------------LEV08250
C                                                                       LEV08260
      ALFA = FINPRO(N,V1(1),1,V2(1),1)                                  LEV08270
C                                                                       LEV08280
      DO 940 J = 1,N                                                    LEV08290
  940 V1(J) = V1(J)-ALFA*V2(J)                                          LEV08300
C                                                                       LEV08310
      BATA = FINPRO(N,V1(1),1,V1(1),1)                                  LEV08320
      BATA = DSQRT(BATA)                                                LEV08330
      SUM = ONE/BATA                                                    LEV08340
C                                                                       LEV08350
      TEMP = BETA(IVEC)                                                 LEV08360
      TEMP = DABS(BATA - TEMP)/TEMP                                     LEV08370
      IF (TEMP.LT.1.0D-10)GO TO 960                                     LEV08380
C                                                                       LEV08390
C     THE BETA BEING REGENERATED DO NOT MATCH THE BETA IN FILE 2.       LEV08400
C     SOMETHING IS WRONG IN THE LANCZOS VECTOR GENERATION.              LEV08410
C     PROGRAM TERMINATES FOR USER TO CORRECT THE PROBLEM                LEV08420
C     WHICH MUST BE IN THE STARTING VECTOR GENERATION OR IN             LEV08430
C     THE MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV SUPPLIED.             LEV08440
C     THIS SUBROUTINE MUST BE THE SAME ONE USED IN THE                  LEV08450
C     EIGENVALUE COMPUTATIONS OR A MISMATCH WILL ENSUE.                 LEV08460
C                                                                       LEV08470
      WRITE(6,950) IVEC,BATA,BETA(IVEC),TEMP                            LEV08480
  950 FORMAT(/2X,'IVEC',16X,'BATA',10X,'BETA(IVEC)',14X,'RELDIF'/I6,    LEV08490
     13E20.12/' IN LANCZOS VECTOR REGENERATION THE ENTRIES OF THE TRIDIALEV08500
     1GONAL MATRICES BEING'/' GENERATED ARE NOT THE SAME AS THOSE IN THELEV08510
     1 MATRIX SUPPLIED ON FILE 2.'/' THEREFORE SOMETHING IS BEING INITIALEV08520
     1LIZED OR COMPUTED DIFFERENTLY FROM THE WAY'/' IT WAS COMPUTED IN TLEV08530
     1HE EIGENVALUE COMPUTATIONS'/' THE PROGRAM TERMINATES FOR THE USER LEV08540
     1TO DETERMINE WHAT THE PROBLEM IS'/)                               LEV08550
      GO TO 1550                                                        LEV08560
C                                                                       LEV08570
C                                                                       LEV08580
  960 CONTINUE                                                          LEV08590
      DO 970 J = 1,N                                                    LEV08600
      TEMP = SUM*V1(J)                                                  LEV08610
      V1(J) = V2(J)                                                     LEV08620
  970 V2(J) = TEMP                                                      LEV08630
C                                                                       LEV08640
  980 CONTINUE                                                          LEV08650
C                                                                       LEV08660
      LFIN = 0                                                          LEV08670
      DO 1000 J = 1,NGOODC                                              LEV08680
      LL = LFIN                                                         LEV08690
      LFIN = LFIN + N                                                   LEV08700
```

```
C
      IF(IABS(MA(J)).LT.IVEC.OR.MP(J).EQ.MPMIN) GO TO 1000
      II = IVEC + MINT(J) - 1
      TEMP = TVEC(II)
C     II IS THE (IVEC)TH COMPONENT OF THE T-EIGENVECTOR CONTAINED
C     IN TVEC(MINT(J)).
C
      DO 990 K = 1,N
      LL = LL + 1
  990 RITVEC(LL) = TEMP*V2(K) + RITVEC(LL)
C
 1000 CONTINUE
C
      IVEC = IVEC + 1
      IF (IVEC.LE.KMAXU) GO TO 930
C
C
C     RITZVECTOR GENERATION IS COMPLETE. NORMALIZE EACH RITZVECTOR.
C     NOTE THAT IF CERTAIN RITZ VECTORS WERE NOT COMPUTED THEN THAT
C     PORTION OF THE RITVEC ARRAY WAS NOT UTILIZED.
C
      LFIN = 0
      DO 1050 J = 1,NGOODC
C
      KK = LFIN
      LFIN = LFIN + N
      IF(MP(J).EQ.MPMIN) GO TO 1050
C
      DO 1010 K = 1,N
      KK = KK + 1
 1010 V2(K) = RITVEC(KK)
C
      SUM = FINPRO(N,V2(1),1,V2(1),1)
      SUM = DSQRT(SUM)
      RNORM(J) = SUM
      TEMP = DABS(ONE-SUM)
      SUM = ONE/SUM
C
      KK = LFIN - N
      DO 1020 K = 1,N
      KK = KK + 1
      V2(K) = SUM*V2(K)
 1020 RITVEC(KK) = V2(K)
C
      IF (IWRITE.NE.0) WRITE(6,1030) J,GOODEV(J)
 1030 FORMAT(/I5,' TH EIGENVALUE CONSIDERED = ',E20.12/)
C
      IF (IWRITE.NE.0) WRITE(6,1040) TERR(J),TBETA(J),TEMP
 1040 FORMAT(' NORM OF ERROR IN T-EIGENVECTOR = ',E14.3/
     1 ' BETA(MA(J)+1)*U(MA(J)) = ',E14.3/
     1 ' ABS(NORM(RITVEC) - 1.0)  = ',E14.3/)
C
      LINT = LFIN - N + 1
      EVAL = EVNEW(J)
C
```

```
C-----------------------------------------------------------------------
C
      CALL CMATV(RITVEC(LINT),V2,EVAL)
C
C-----------------------------------------------------------------------
C
C     COMPUTE ERROR IN RITZ VECTOR CONSIDERED AS A EIGENVECTOR OF A.
C     V2 = A*RITVEC - EVAL*RITVEC
C
      SUM = FINPRO(N,V2(1),1,V2(1),1)
      SUM = DSQRT(SUM)
      ERR(J) = SUM
      GAP = ABS(AMINGP(J))
      ERRDGP(J) = SUM/GAP
C
 1050 CONTINUE
C
C
C     RITZVECTORS ARE NORMALIZED AND ERROR ESTIMATES ARE IN ERR ARRAY
C     AND IN ERRDGP ARRAY. STORE EVERYTHING
C
C
      WRITE(9,1060)
 1060 FORMAT(3X,'A-EIGENVALUE',2X,'MA(J)',3X,'A-MINGAP',6X,'AERROR',2X,
     1 'AERROR/GAP',6X,'TERROR')
C
      WRITE(13,1070)
 1070 FORMAT(16X,'GOODEV(J)',5X,'RITZNORM',6X,'AMINGAP',5X,
     1 'TBETA(J)',5X,'TLAST(J)')
C
      DO 1100 J=1,NGOODC
C
      IF(MP(J).EQ.MPMIN) GO TO 1100
C
      WRITE(9,1080)EVNEW(J),MA(J),AMINGP(J),ERR(J),ERRDGP(J),TERR(J)
 1080 FORMAT(E15.8,I6,4E12.4)
C
      WRITE(13,1090) EVNEW(J),RNORM(J),AMINGP(J),TBETA(J),TLAST(J)
 1090 FORMAT(E25.14,4E13.5)
C
 1100 CONTINUE
C
      IF(MREJEC.EQ.0) GO TO 1180
      WRITE(9,1110)
 1110 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING EIGENVA
     1LUES'/' EITHER BECAUSE THEY HAD NOT CONVERGED OR BECAUSE THE ERROR
     1 ESTIMATE'/'  WAS NOT AS SMALL AS DESIRED'/)
C
      DO 1170 J = 1,NGOODC
      IF(MP(J).NE.MPMIN) GO TO 1170
C     WRITE OUT MESSAGE FOR EACH EIGENVALUE FOR WHICH NO EIGENVECTOR
C     WAS COMPUTED.
C
      WRITE(9,1120)
 1120 FORMAT(6X,'GOODEV(J)',3X,'MA(J)',5X,'AMINGP(J)',6X,'TLAST(J)',3X,
```

```
     1'MP(J)')
      WRITE(9,1130) GOODEV(J),MA(J),AMINGP(J),TBETA(J),MP(J)
 1130 FORMAT(E15.8,I8,2E14.4,I8)
C
      WRITE(13,1140)
 1140 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING EIGENVA
     1LUES'/' BECAUSE THEY HAD NOT CONVERGED'/)
C
      WRITE(13,1150)
 1150 FORMAT(6X,'GOODEV(J)',3X,'MA(J)',3X,'M1(J)',3X,'M2(J)',3X,'MP(J)'
     1/)
      WRITE(13,1160) GOODEV(J),MA(J),M1(J),M2(J),MP(J)
 1160 FORMAT(E15.8,4I8)
C
 1170 CONTINUE
 1180 CONTINUE
C
      WRITE(9,1190)
 1190 FORMAT(/' ABOVE ARE ERROR ESTIMATES FOR THE A AND T EIGENVECTORS'/
     1 ' ASSOCIATED WITH THE GOODEV LISTED IN COLUMN 1'/
     1 ' AERROR = NORM(A*X - EV*X)  TERROR = NORM(T*Y - EV*Y) '/
     1 ' WHERE T = T(1,MA(J))    X = RITZ VECTOR = V*Y  V = SUCCESSIVE'/
     1 ' LANCZOS VECTORS. AMINGAP = GAP TO NEAREST A-EIGENVALUE'//)
C
      WRITE(13,1200)
 1200 FORMAT(/' ABOVE ARE ERROR ESTIMATES ASSOCIATED WITH THE GOODEV'/
     1 ' RITZNORM = NORM(COMPUTED RITZ VECTOR)'/
     1 ' TBETA(J) = BETA(MA(J)+1)*Y(MA(J)),  T*Y = EVAL*Y'/
     1 ' TLAST(J) = Y(MA(J))'/
     1 ' AMINGAP = GAP TO NEAREST A-EIGENVALUE'/)
C
C     NUMBER OF RITZ VECTORS COMPUTED
      NCOMPU = NGOODC - MREJEC
      WRITE(12,1210) N,NCOMPU,NGOODC,MATNO
 1210 FORMAT(3I6,I12,' SIZE A, NO.RITZVECS, NO.EVALUES,MATNO')
C
      LFIN = 0
      DO 1270 J = 1,NGOODC
      LINT = LFIN + 1
      LFIN = LFIN + N
C
      IF(MP(J).EQ.MPMIN) GO TO 1250
C     RITZ VECTOR WAS COMPUTED
      WRITE(12,1220) J, GOODEV(J), MP(J)
 1220 FORMAT(I6,4X,E20.12,I6,' J, EIGENVAL, MP(J)')
C
      WRITE(12,1230) ERR(J),ERRDGP(J)
 1230 FORMAT(2E15.5,' = NORM(A*Z-EVAL*Z) AND  NORM(A*Z-EVAL*Z)/MINGAP')
C
      WRITE(12,1240) (RITVEC(LL), LL=LINT,LFIN)
 1240 FORMAT(4E20.12)
      GO TO 1270
C     NO RITZ VECTOR WAS COMPUTED FOR THIS EIGENVALUE
 1250 WRITE(12,1260) J,GOODEV(J),MP(J)
 1260 FORMAT(I6,4X,E20.12,I6,' J,EIGVALUE,NO RITZ VECTOR COMPUTED')
```

```
C
 1270 CONTINUE
C
C     DID ANY T-MATRICES INCLUDE OFF-DIAGONAL ENTRIES SMALLER THAN
C     DESIRED, AS SPECIFIED BY BTOL?
C
      IF(IB.GT.0) GO TO 1300
C
      WRITE(6,1280) KMAXU
 1280 FORMAT(/' FOR LARGEST T-MATRIX CONSIDERED',I7,' CHECK THE SIZE OF
     1BETAS')
C
C-----------------------------------------------------------------------
C
      CALL TNORM(ALPHA,BETA,BKMIN,TEMP,KMAXU,IBMT)
C
C-----------------------------------------------------------------------
C
      IF(IBMT.LT.0) WRITE (6,1290)
 1290 FORMAT(/' WARNING THE T-MATRICES FOR ONE OR MORE OF THE EIGENVALUE
     1S CONSIDERED'/' HAD AN OFF-DIAGONAL ENTRY THAT WAS SMALLER THAN TH
     1E BETA TOLERANCE THAT WAS SPECIFIED'/)
 1300 CONTINUE
C
       GO TO 1550
C
 1310 WRITE(6,1320) NGOOD,NMAX,MDIMRV
 1320 FORMAT(/I4,' RITZ VECTORS WERE REQUESTED BUT THE REQUIRED DIMENSIO
     1N',I6/' IS LARGER THAN THE USER-SPECIFIED DIMENSION OF RITVEC',I6
     1/' THEREFORE, THE EIGENVECTOR PROCEDURE TERMINATES FOR THE USER TO
     1 INTERVENE')
C
      GO TO 1550
C
 1330 WRITE(6,1340) NOLD,N,MATOLD,MATNO
 1340 FORMAT(//' PARAMETERS READ FROM FILE 3 DO NOT AGREE WITH THOSE SPE
     1CIFIED BY THE USER'/' N,NOLD,MATOLD,MATNO = ',2I6,2I12/' PROGRAM T
     1ERMINATES FOR USER TO RESOLVE PROBLEM'/)
C
      GO TO 1550
C
 1350 WRITE(6,1360)
 1360 FORMAT(//' PARAMETERS IN THE ALPHA,BETA FILE HEADER DO NOT AGREE W
     1ITH PARAMTERS'/' SPECIFIED BY THE USER.  THEREFORE THE PROGRAM TER
     1MINATES FOR THE USER'/' TO RESOLVE THE PROBLEM'/)
C
      GO TO 1550
C
 1370 WRITE(6,1380) KMAX,MEV
 1380 FORMAT(/' ALPHA,BETA FILE HEADER GIVES KMAX =',I6/
     1' BUT EIGENVALUES WERE COMPUTED AT MEV = ',I6,' PROGRAM STOPS'/)
C
      GO TO 1550
C
 1390 WRITE(6,1400)
```

```
 1400 FORMAT(//' PROGRAM COMPUTED 1ST GUESSES AT T-MATRIX SIZES AND READLEV10910
     1 THEM TO'/' FILE 10, THEN TERMINATED AS REQUESTED.')              LEV10920
      GO TO 1550                                                        LEV10930
C                                                                       LEV10940
 1410 WRITE(6,1420) MTOL, MDIMTV                                        LEV10950
 1420 FORMAT(/' PROGRAM TERMINATES BECAUSE THE TVEC DIMENSION ANTICIPATELEV10960
     1D',I7/'  IS LARGER THAN THE TVEC DIMENSION',I7,' SPECIFIED BY THE LEV10970
     1USER.'/'  USER MAY RESET THE TVEC DIMENSION AND RESTART THE PROGRALEV10980
     1M')                                                               LEV10990
      GO TO 1550                                                        LEV11000
C                                                                       LEV11010
 1430 WRITE(6,1440)                                                     LEV11020
 1440 FORMAT(/' PROGRAM TERMINATES BECAUSE NO SUITABLE T-EIGENVECTORS WELEV11030
     1RE IDENTIFIED'/'  FOR ANY OF THE EIGENVALUES SUPPLIED.  PROBLEM COLEV11040
     1ULD BE CAUSED'/'  BY TOO SMALL A TVEC DIMENSION OR SIMPLY THAT SUILEV11050
     1TABLE T-VECTORS COULD'/'  NOT BE IDENTIFIED.  USER SHOULD CHECK OULEV11060
     1TPUT'/)                                                           LEV11070
      GO TO 1550                                                        LEV11080
C                                                                       LEV11090
 1450 WRITE(6,1460) LVCONT,NTVEC,NGOOD                                  LEV11100
 1460 FORMAT(/' LVCONT FLAG =',I2,' AND NUMBER ',I5,' OF T-EIGENVECTORS LEV11110
     1 COMPUTED N.E.'/' NUMBER',I5,' REQUESTED SO PROGRAM TERMINATES'/) LEV11120
      GO TO 1550                                                        LEV11130
C                                                                       LEV11140
 1470 WRITE(6,1480)                                                     LEV11150
 1480 FORMAT(//' PROGRAM TERMINATES WITHOUT COMPUTING RITZ VECTORS'/    LEV11160
     1' BECAUSE ALL T-EIGENVECTORS WERE REJECTED AS NOT SUITABLE FOR THELEV11170
     1 RITZ VECTOR'/' COMPUTATIONS.  PROBABLE CAUSE IS LACK OF CONVERGENLEV11180
     1CE OF THE EIGENVALUES SUPPLIED'/)                                 LEV11190
       GO TO 1550                                                       LEV11200
C                                                                       LEV11210
 1490 WRITE(6,1500)                                                     LEV11220
 1500 FORMAT(/' PROGRAM INDICATES THAT IT IS NOT POSSIBLE TO COMPUTE ANYLEV11230
     1 OF THE'/' REQUESTED EIGENVECTORS. THEREFORE PROGRAM TERMINATES') LEV11240
      DO 1510 J=1,NGOODC                                                LEV11250
 1510 WRITE(6,1520)  J,GOODEV(J),MP(J)                                  LEV11260
 1520 FORMAT(/4X,' J',11X,'GOODEV(J)',4X,'MP(J)'/I6,E20.12,I9)          LEV11270
      GO TO 1550                                                        LEV11280
C                                                                       LEV11290
 1530 WRITE(6,1540) MBETA,KMAXN                                         LEV11300
 1540 FORMAT(/' PROGRAM TERMINATES BECAUSE THE STORAGE ALLOTTED FOR THE LEV11310
     1BETA ARRAY',I8/' IS NOT SUFFICIENT FOR THE ENLARGED KMAX =',I8,' TLEV11320
     1HAT THE PROGRAM WANTS'/' USER CAN ENLARGE THE DIMENSIONS OF THE ALLEV11330
     1PHA AND BETA ARRAYS'/' AND RERUN THE PROGRAM'/)                   LEV11340
C                                                                       LEV11350
 1550 CONTINUE                                                          LEV11360
C                                                                       LEV11370
      STOP                                                              LEV11380
C-----END OF MAIN PROGRAM FOR LANCZOS EIGENVECTORS--------------------- LEV11390
      END                                                               LEV11400
```

2.5 LEMULT: LANCZS and Sample Matrix-Vector Multiply Subroutines

```
C-----LEMULT-----------------------------------------------------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C        REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4
C        Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C        Applied Mathematics, 2002. SIAM Publications,
C        Philadelphia, PA. USA
C
C
C     CONTAINS SUBROUTINES  LANCZS, USPEC, AND CMATV
C     TO BE USED WITH THE REAL SYMMETRIC VERSION OF THE LANCZOS
C     EIGENVALUE/EIGENVECTOR PROCEDURES.
C     ALSO CONTAINS SUBROUTINES FOR POISSON TEST MATRICES THAT ALLOW
C     COMPUTATION OF TRUE ERRORS IN COMPUTED EIGENVALUES AND
C     IN CORRESPONDING EIGENVECTORS.
C
C     NONPORTABLE CONSTRUCTIONS:
C     1.  THE ENTRY MECHANISM USED TO PASS THE STORAGE
C         LOCATIONS OF THE USER-SPECIFIED MATRIX FROM THE
C         SUBROUTINE USPEC TO THE MATRIX-VECTOR SUBROUTINE
C         CMATV.
C     2.  IN THE SAMPLE USPEC AND CMATV FOR DIAGONAL TEST MATRICES:
C         FREE FORMAT (8,*) AND THE FORMAT (20A4).
C     3.  IN THE POISSON SUBROUTINES PROVIDED, THE DATA MACHEP
C         DEFINITION AND MANY OF THE INDICES FOR ARRAYS ARE NOT
C         IN A PORTABLE CONSTRUCTION.  THESE PROGRAMS SHOULD BE
C         REMOVED FROM THE LEMULT FILE IF THE USER IS NOT USING THEM.
C
C-----LANCZS-COMPUTE THE LANCZOS TRIDIAGONAL MATRICES-------------------
C
      SUBROUTINE LANCZS(MATVEC,ALPHA,BETA,V1,V2,G,KMAX,MOLD1,N,IIX)
C
C----------------------------------------------------------------------
```

```
      DOUBLE PRECISION  ALPHA(1),BETA(1),V1(1),V2(1),SUM,TEMP,ONE,ZERO
      REAL  G(1)
      DOUBLE PRECISION FINPRO, DSQRT
      EXTERNAL MATVEC
C-----------------------------------------------------------------------
C
      ZERO = 0.D0
      ONE = 1.D0
C
      IF(MOLD1.GT.1)GO TO 30
C
C    ALPHA/BETA GENERATION STARTS AT I = 1
C    MOLD1 = 1 SET V1 = 0. AND V2 = RANDOM UNIT VECTOR
      BETA(1) = ZERO
      IIL=IIX
C
C-----------------------------------------------------------------------
      CALL GENRAN(IIL,G,N)
C-----------------------------------------------------------------------
C
      DO 10 I = 1,N
   10 V2(I) = G(I)
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V2(1),1,V2(1),1)
C-----------------------------------------------------------------------
C
      SUM = ONE/DSQRT(SUM)
      DO 20 I = 1,N
      V1(I) = ZERO
   20 V2(I) = V2(I)*SUM
C
C    ALPHA BETA GENERATION LOOP
   30 CONTINUE
C
      DO 60 I=MOLD1,KMAX
      SUM = BETA(I)
C    MATVEC(V2,V1,SUM) CALCULATES  V1 = A*V2 - SUM*V1
C
C-----------------------------------------------------------------------
      CALL MATVEC(V2,V1,SUM)
C-----------------------------------------------------------------------
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V1(1),1,V2(1),1)
C-----------------------------------------------------------------------
C
      ALPHA(I) = SUM
      DO 40 J=1,N
   40 V1(J) = V1(J)-SUM*V2(J)
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V1(1),1,V1(1),1)
C-----------------------------------------------------------------------
C
```

```
      IN = I+1
      BETA(IN) = DSQRT(SUM)
      SUM = ONE/BETA(IN)
      DO 50 J=1,N
      TEMP = SUM*V1(J)
      V1(J) = V2(J)
   50 V2(J) = TEMP
   60 CONTINUE
C
C     END ALPHA, BETA GENERATION LOOP
C
      RETURN
C-----END OF LANCZS-----------------------------------------------------
      END
C
C-----USPEC (GENERAL SYMMETRIC SPARSE MATRICES)-------------------------
C
C     SUBROUTINE USPEC(N,MATNO)
      SUBROUTINE GUSPEC(N,MATNO)
C
C----------------------------------------------------------------------
      DOUBLE PRECISION  A(10000),AD(5010)
      INTEGER  IROW(10000),ICOL(5010)
C----------------------------------------------------------------------
C    USPEC DIMENSIONS AND INITIALIZES THE ARRAYS NEEDED TO DEFINE
C    THE USER-SPECIFIED MATRIX AND THEN PASSES THE STORAGE LOCATIONS
C    OF THESE ARRAYS TO THE MULTIPLY SUBROUTINE CMATV.
C
C    MATRIX IS STORED IN FOLLOWING SPARSE MATRIX FORMAT:
C    N = ORDER OF A-MATRIX,
C    NZS = NUMBER OF NONZERO SUBDIAGONAL ENTRIES,
C    NZL = INDEX OF LAST COLUMN CONTAINING NONZERO SUBDIAGONAL ENTRIES,
C    ICOL(J), J=1,NZL IS THE NUMBER OF NONZERO SUBDIAGONAL ELEMENTS
C             IN COLUMN J.
C    IROW(K), K = 1,NZS IS THE CORRESPONDING ROW INDEX FOR A(K).
C    AD(I), I=1,N CONTAINS DIAGONAL ENTRIES (INCLUDING ANY 0
C           DIAGONAL ENTRIES).
C    A(K), K=1,NZS CONTAINS NONZERO SUBDIAGONAL ENTRIES, BY COLUMN
C    FOR J > NZL THERE ARE NO NONZERO SUBDIAGONAL ELEMENTS IN COLUMN J.
C    ICOL(J) = 0 IS ALLOWED
C
C----------------------------------------------------------------------
C     ARRAYS THAT DEFINE THE MATRIX ARE READ IN FROM FILE 8
C
      READ(8,10) NZS,NOLD,NZL,MATOLD
   10 FORMAT(I10,2I6,I8)
C
      WRITE(6,20) NZS,NOLD,NZL,MATOLD
   20 FORMAT(I10,2I6,I8,' = NZS,NOLD,NZL,MATOLD'/)
C
C     TEST OF PARAMETER CORRECTNESS
      ITEMP = (NOLD-N)**2 + (MATNO-MATOLD)**2
C
      IF(ITEMP.EQ.0) GO TO 40
C
```

```
      WRITE(6,30)                                                       LEM01550
   30 FORMAT(' PROGRAM TERMINATES BECAUSE EITHER ORDERS OF OR LABELS FORLEM01560
     1 MATRIX DISAGREE')                                                LEM01570
      GO TO 70                                                          LEM01580
C                                                                       LEM01590
   40 CONTINUE                                                          LEM01600
C                                                                       LEM01610
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS READ      LEM01620
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS READ      LEM01630
      READ(8,50) (ICOL(K), K=1,NZL)                                     LEM01640
      READ(8,50) (IROW(K), K=1,NZS)                                     LEM01650
   50 FORMAT(13I6)                                                      LEM01660
C                                                                       LEM01670
C     DIAGONAL IS READ FIRST, THEN NONZERO BELOW DIAGONAL ENTRIES       LEM01680
      READ(8,60) (AD(K), K=1,N)                                         LEM01690
      READ(8,60) (A(K), K=1,NZS)                                        LEM01700
   60 FORMAT(4E19.10)                                                   LEM01710
C                                                                       LEM01720
C-----------------------------------------------------------------------LEM01730
C     PASS STORAGE LOCATIONS OF ARRAYS THAT DEFINE THE MATRIX TO        LEM01740
C     THE MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV                       LEM01750
C                                                                       LEM01760
      CALL CMATVE(A,AD,ICOL,IROW,N,NZL)                                 LEM01770
C                                                                       LEM01780
C-----------------------------------------------------------------------LEM01790
C                                                                       LEM01800
      RETURN                                                            LEM01810
   70 STOP                                                              LEM01820
C-----END OF USPEC------------------------------------------------------LEM01830
      END                                                               LEM01840
C                                                                       LEM01850
C-----MATRIX-VECTOR MULTIPLY FOR REAL SPARSE SYMMETRIC MATRICES---------LEM01860
C                                                                       LEM01870
C     SUBROUTINE CMATV(W,U,SUM)                                         LEM01880
      SUBROUTINE GCMATV(W,U,SUM)                                        LEM01890
C                                                                       LEM01900
C-----------------------------------------------------------------------LEM01910
       DOUBLE PRECISION   U(1),W(1),A(1),AD(1),SUM                      LEM01920
       INTEGER  IROW(1),ICOL(1)                                         LEM01930
C-----------------------------------------------------------------------LEM01940
C    SPARSE MATRIX-VECTOR MULTIPLY FOR LANCZS  U = A*W - SUM*U          LEM01950
C    SEE USPEC SUBROUTINE FOR DESCRIPTION OF THE ARRAYS THAT DEFINE     LEM01960
C    THE MATRIX                                                         LEM01970
C-----------------------------------------------------------------------LEM01980
C                                                                       LEM01990
      GO TO 3                                                           LEM02000
C     STORAGE LOCATIONS OF ARRAYS ARE PASSED TO CMATV FROM USPEC        LEM02010
      ENTRY CMATVE(A,AD,ICOL,IROW,N,NZL)                                LEM02020
      GO TO 4                                                           LEM02030
C-----------------------------------------------------------------------LEM02040
C     COMPUTE THE DIAGONAL TERMS                                        LEM02050
    3 DO 10 I = 1,N                                                     LEM02060
   10 U(I) = AD(I)*W(I)-SUM*U(I)                                        LEM02070
C                                                                       LEM02080
C     COMPUTE BY COLUMN                                                 LEM02090
```

```
      LLAST = 0
      DO 30 J = 1,NZL
C
      IF (ICOL(J).EQ.0) GO TO 30
      LFIRST = LLAST + 1
      LLAST = LLAST + ICOL(J)
C
      DO 20 L = LFIRST,LLAST
      I = IROW(L)
C
      U(I) = U(I) + A(L)*W(J)
      U(J) = U(J) + A(L)*W(I)
C
   20 CONTINUE
C
   30 CONTINUE
C
    4 RETURN
C
C-----END OF CMATV------------------------------------------------------
      END
C
C-----MATRIX-VECTOR MULTIPLY FOR DIAGONAL TEST MATRICES-----------------
C
C     SUBROUTINE CMATV(W,U,SUM)
      SUBROUTINE DCMATV(W,U,SUM)
C
C     CMATV COMPUTES  U = (DIAGONAL MATRIX) * W - SUM * U
C-----------------------------------------------------------------------
      DOUBLE PRECISION  W(1),U(1),SUM
      DOUBLE PRECISION  D(1)
C-----------------------------------------------------------------------
      GO TO 3
      ENTRY MVDIAE(D,N)
      GO TO 4
C-----------------------------------------------------------------------
C
    3 DO 10 I=1,N
   10 U(I)= D(I)*W(I) - SUM*U(I)
    4 RETURN
C
C-----END OF DIAGONAL TEST MATRIX  MULTIPLY-----------------------------
      END
C
C
C-----START OF USPEC FOR DIAGONAL TEST MATRIX---------------------------
C
C     SUBROUTINE USPEC(N,MATNO)
      SUBROUTINE DUSPEC(N,MATNO)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  D(1000), SHIFT, SPACE
      DOUBLE PRECISION  DABS, DFLOAT
      REAL  EXPLAN(20)
C-----------------------------------------------------------------------
```

```
C                                                                       LEM02650
      READ(8,10) EXPLAN                                                 LEM02660
   10 FORMAT(20A4)                                                      LEM02670
      READ(8,*) NOLD,NUNIF,SPACE,D(1),SHIFT                             LEM02680
      NNUNIF = NOLD - NUNIF                                             LEM02690
      WRITE(6,20) NOLD,SPACE,NNUNIF,D(1),SHIFT                          LEM02700
   20 FORMAT(/' DIAGONAL TEST MATRIX, SIZE = ',I4/' MOST ENTRIES ARE ', LEM02710
     1E10.3,' UNITS APART.',I3,' ENTRIES'/' ARE IRREGULARLY SPACED. FIRSLEM02720
     1T ENTRY IS ',E10.3,' SHIFT = ',E10.3/)                            LEM02730
C                                                                       LEM02740
      IF(N.NE.NOLD) GO TO 90                                            LEM02750
C     COMPUTE THE UNIFORM PORTION OF THE SPECTRUM                       LEM02760
      DO 30 J=2,NUNIF                                                   LEM02770
   30 D(J) = D(1) - DFLOAT(J-1)*SPACE                                   LEM02780
      NUNIF1=NUNIF + 1                                                  LEM02790
      READ(8,10)  EXPLAN                                                LEM02800
      DO 40 J=NUNIF1,N                                                  LEM02810
   40 READ(8,*) D(J)                                                    LEM02820
      NB = NUNIF - 2                                                    LEM02830
C                                                                       LEM02840
      IF(SHIFT.EQ.0.) GO TO 60                                          LEM02850
      DO 50 J=1,N                                                       LEM02860
   50 D(J) = D(J) + SHIFT                                               LEM02870
C                                                                       LEM02880
C     PRINT OUT THE EIGENVALUES OF INTEREST                             LEM02890
   60 WRITE(6,70) (D(I), I=1,10 )                                       LEM02900
      WRITE(6,80) (D(I), I = NB,N)                                      LEM02910
   70 FORMAT(/' REAL SYMMETRIC LANCZOS TEST, 1ST 10 ENTRIES OF DIAGONAL LEM02920
     1TEST MATRIX = '/(3E22.14))                                        LEM02930
   80 FORMAT(/' MIDDLE UNIFORM PORTION OF MATRIX IS NOT PRINTED OUT'/   LEM02940
     1' END OF UNIFORM PLUS NONUNIFORM SECTION = '/(3E25.16))           LEM02950
C                                                                       LEM02960
C     DIAGONAL GENERATION COMPLETE                                      LEM02970
C                                                                       LEM02980
C    CALL ENTRY TO MATRIX-VECTOR MULTIPLY SUBROUTINE  TO PASS           LEM02990
C    STORAGE LOCATION OF D-ARRAY AND ORDER OF A-MATRIX.                 LEM03000
       CALL MVDIAE(D,N)                                                 LEM03010
C                                                                       LEM03020
      RETURN                                                            LEM03030
   90 WRITE(6,100) NOLD,N                                               LEM03040
  100 FORMAT(' PROGRAM TERMINATES BECAUSE NOLD = ',I5,'DOES NOT EQUAL N LEM03050
     1 =',I5)                                                           LEM03060
C-----END OF USPEC SUBROUTINE FOR DIAGONAL TEST MATRICES----------------LEM03070
      STOP                                                              LEM03080
      END                                                               LEM03090
C                                                                       LEM03100
C-----POISSON TEST MATRICES---------------------------------------------LEM03110
C                                                                       LEM03120
C     CONTAINS SUBROUTINES USPEC, CMATV, EXEVG, EXERR AND  EXVEC        LEM03130
C                                                                       LEM03140
C-----START OF USPEC----------------------------------------------------LEM03150
C                                                                       LEM03160
      SUBROUTINE USPEC(N,MATNO)                                         LEM03170
C     SUBROUTINE PUSPEC(N,MATNO)                                        LEM03180
C                                                                       LEM03190
```

```
C-----------------------------------------------------------------------
      DOUBLE PRECISION  C0,C1,C2,HALF,ONE
      REAL  EXPLAN(20)
C--------------------------------------------------------------------
      HALF = 0.5D0
      ONE  = 1.0D0
C
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 8 (FREE FORMAT)
C
      READ(8,10) EXPLAN
      WRITE(6,10) EXPLAN
   10 FORMAT(20A4)
C
      READ(8,10) EXPLAN
      READ(8,*) KX,KY,C0
      N = KX*KY
      C1 = HALF-C0
      C2 = ONE
C
      WRITE(6,20) N,KX,KY,C2,C0,C1
   20 FORMAT(/5X,'N',4X,'KX',4X,'KY',7X,'DIAGONAL',3X,'X-CODIAGONAL',
     1 3X,'Y-CODIAGONAL'/3I6,3E15.8/)
C
C-----------------------------------------------------------------------
      CALL PMATVE(C0,C1,C2,KX,KY)
      CALL EXEVE(C0,C1,C2,KX,KY)
      CALL EXERRP(C0,C1,C2,KX,KY)
C     CALL EXVECP(C0,C1,C2,KX,KY)
C-----------------------------------------------------------------------
C
      RETURN
C-----END OF USPEC------------------------------------------------------
      END
C
C-----START OF CMATV----------------------------------------------------
C     CALCULATE U = A*W - SUM*U FOR REAL POISSON MATRICES
C
      SUBROUTINE CMATV(W,U,SUM)
C     SUBROUTINE PMATV(W,U,SUM)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  U(1),W(1)
      DOUBLE PRECISION  C0,C1,C2,CC0,CC1,SUM
C-----------------------------------------------------------------------
      GO TO 3
      ENTRY  PMATVE(CC0,CC1,C2,KX,KY)
C-----------------------------------------------------------------------
C
      C0 = -CC0
      C1 = -CC1
      GO TO 4
C
    3 N = KX*KY
      KX1 = KX-1
      KY1 = KY-1
```

```
C
      KK = 1
      U(KK) = (C2*W(KK)+C0*W(KK+1)+C1*W(KK+KX)) - SUM*U(KK)
      KK = KX
      U(KK) = (C2*W(KK)+C0*W(KK-1)+C1*W(KK+KX)) - SUM*U(KK)
      KK = N - KX + 1
      U(KK) = (C2*W(KK)+C0*W(KK+1)+C1*W(KK-KX)) - SUM*U(KK)
      KK = N
      U(KK) = (C2*W(KK)+C0*W(KK-1)+C1*W(KK-KX)) - SUM*U(KK)
C
      DO 10 J = 2,KX1
      KK = J
      U(KK) = (C2*W(KK)+C0*W(KK-1)+C0*W(KK+1)+C1*W(KK+KX)) - SUM*U(KK)
      KK = J+N-KX
      U(KK) = (C2*W(KK)+C0*W(KK-1)+C0*W(KK+1)+C1*W(KK-KX))-SUM*U(KK)
   10 CONTINUE
C
      DO 30 J = 2,KY1
      KK = (J-1)*KX + 1
      U(KK) = (C2*W(KK)+C0*W(KK+1)+C1*W(KK-KX)+C1*W(KK+KX)) - SUM*U(KK)
      DO 20 I = 2,KX1
      KK = KK + 1
      U(KK) = (C2*W(KK)+C0*W(KK-1)+C0*W(KK+1)+C1*W(KK-KX)
     1 +C1*W(KK+KX)) - SUM*U(KK)
   20 CONTINUE
      KK = KK + 1
      U(KK) = (C2*W(KK)+C0*W(KK-1)+C1*W(KK-KX)+C1*W(KK+KX)) - SUM*U(KK)
   30 CONTINUE
C
    4 RETURN
C
C-----END OF CMATV------------------------------ ----------------------
      END
C
C-----START OF EXEVG--------------------------------------------------
C
C     COMPUTES TRUE EIGENVALUES OF POISSON MATRIX, GAPS BETWEEN
C     TRUE EIGENVALUES, AND MULTIPLICITIES OF TRUE EIGENVALUES
C     AND STORE THESE VALUES, RESPECTIVELY, IN U, G, AND MP.
C     THESE QUANTITIES ARE WRITTEN OUT TO FILE 9
C
      SUBROUTINE EXEVG(U,G,MP)
C
C--------------------------------------------------------------------
      DOUBLE PRECISION  U(*)
      DOUBLE PRECISION MACHEP,EPSM,C0,C1,C2,T0,T1,PIK,PIL,ONE,TWO
      DOUBLE PRECISION ATOLN,EE
      REAL G(1)
      INTEGER MP(1)
C--------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.0D0*MACHEP
C--------------------------------------------------------------------
      GO TO 3
      ENTRY EXEVE(C0,C1,C2,KX,KY)
```

```
      GO TO 4
C-----------------------------------------------------------------------
    3 N = KX*KY
      ONE  = 1.0D0
      TWO  = 2.0D0
      T0 = DARCOS(-ONE)
      T1 = DFLOAT(KX+1)
      PIK = T0/T1
      T1 = DFLOAT(KY+1)
      PIL = T0/T1
C     GENERATE TRUE EIGENVALUES
      KP = 0
      DO 20 J = 1,KY
      T1 = PIL*DFLOAT(J)
      T0 = C2 - TWO*C1*DCOS(T1)
      DO 10 I = 1,KX
      KP = KP+1
      T1 = PIK*DFLOAT(I)
   10 U(KP) = T0 - TWO*C0*DCOS(T1)
   20 CONTINUE
C
C     ORDER U VECTOR BY INCREASING ALGEBRAIC SIZE
      DO 40 K = 2,N
      KM1 = K-1
      DO 30 L = 1,KM1
      JJ = K-L
      IF (U(JJ+1).GE.U(JJ)) GO TO 40
      T0 = U(JJ)
      U(JJ) = U(JJ+1)
   30 U(JJ+1) = T0
   40 CONTINUE
      ATOLN = DMAX1(DABS(U(1)),DABS(U(N)))*EPSM
C
      WRITE(9,50)
   50 FORMAT(' TRUE EIGENVALUES FOR POISSON'/)
C
      WRITE(9,60)N,KX,KY,C2,C0,C1,ATOLN
      WRITE(6,60) N,KX,KY,C2,C0,C1,ATOLN
   60 FORMAT(1X,'A-SIZE',2X,'X-DIM',2X,'Y-DIM'/3I7/
     1 5X,'A-DIAGONAL',3X,'X-CODIAGONAL',3X,'Y-CODIAGONAL',10X,'ATOLN'/
     2 4E15.8)
C
C     DETERMINE MULTIPLICITIES FOR TRUE EIGENVALUES
      I = 1
      IDEX = 1
      J = 1
      NEXACT = 0
   70 J = J+1
      IF (J.GT.N) GO TO 80
      EE = DABS(U(J)-U(I))
      IF (EE.GT.ATOLN) GO TO 80
      IDEX = IDEX+1
      GO TO 70
   80 NEXACT = NEXACT+1
      U(NEXACT) = U(I)
```

```
      MP(NEXACT) = IDEX
C     MP(K) = MULTIPLICITY OF KTH EIGENVALUE CLUSTER FOR A
      IDEX = 1
      I = J
      IF (I.GT.N) GO TO 90
      GO TO 70
   90 CONTINUE
C
C     MULTIPLICITIES HAVE BEEN DETERMINED
C     NEXACT = NUMBER OF DISTINCT A-EIGENVALUES
C
      WRITE(9,100)NEXACT
      WRITE(6,100)NEXACT
  100 FORMAT(I6,' = NUMBER OF TRUE A-EIGENVALUES WHICH ARE DISTINCT'/)
C
C     MINGAP CALCULATION FOR DISTINCT A-EIGENVALUES
      NM1 = NEXACT - 1
      G(NEXACT) = U(NM1)-U(NEXACT)
      G(1) = U(2)-U(1)
C
      DO 110 J = 2,NM1
      T0 = U(J)-U(J-1)
      T1 = U(J+1)-U(J)
      G(J) = T1
      IF (T0.LT.T1) G(J) = -T0
  110 CONTINUE
C
C     NEXACT DISTINCT A-EIGENVALUES ARE IN U IN ASCENDING ORDER
C     MP = MULTIPLICITIES OF THE DISTINCT EIGENVALUES OF A
C     G = TRUE MINIMUM GAP IN A FOR EACH OF THESE EIGENVALUES
C     G < 0 INDICATES THE LEFT-HAND GAP WAS MINIMAL.
C     OUTPUT MULTIPLICITIES, DISTINCT EVS, AND MINGAPS TO FILE 9
C
      WRITE(9,120)
  120 FORMAT(5X,'I',1X,'AMULT',5X,'TRUE A-EIGENVALUE(I)',
     1 3X,'A-MINGAP(I)')
C
      WRITE(9,130)(J,MP(J),U(J),G(J), J=1,NEXACT)
  130 FORMAT(2I6,E25.16,E14.3)
C
      WRITE(9,140)
  140 FORMAT(' NEXACT DISTINCT A-EIGENVALUES ARE IN ASCENDING ORDER'/
     1 ' AMULT = MULTIPLICITIES OF THE DISTINCT EIGENVALUES OF A.'/
     2 ' A-MINGAP(I) = TRUE MINIMUM GAP IN A FOR EACH EIGENVALUE.'/
     3 ' A-MINGAP(I) LT 0 INDICATES THE LEFT-HAND GAP WAS MINIMAL.'//)
C
C     WE ORDER U VECTOR BY INCREASING SIZE OF THE GAPS
C
      DO 150 K = 1,NEXACT
  150 MP(K) = K
C
      DO 170 K = 2,NEXACT
      KM1 = K-1
C
      DO 160 L = 1,KM1
```

```
      JJ = K - L                                                        LEM05400
      IF (ABS(G(JJ+1)).GE.ABS(G(JJ))) GO TO 170                         LEM05410
      EE = U(JJ)                                                        LEM05420
      U(JJ) = U(JJ+1)                                                   LEM05430
      U(JJ+1) = EE                                                      LEM05440
      GG = G(JJ)                                                        LEM05450
      G(JJ) = G(JJ+1)                                                   LEM05460
      G(JJ+1) = GG                                                      LEM05470
      IEE = MP(JJ)                                                      LEM05480
      MP(JJ) = MP(JJ+1)                                                 LEM05490
  160 MP(JJ+1) = IEE                                                    LEM05500
C                                                                       LEM05510
  170 CONTINUE                                                          LEM05520
C                                                                       LEM05530
      WRITE(9,180)                                                      LEM05540
  180 FORMAT(5X,'K',6X,'A-MINGAP',5X,'TRUE A-EIGENVALUE(I)',2X,'A-EVNO')LEM05550
C                                                                       LEM05560
      WRITE(9,190)(J,G(J),U(J),MP(J), J=1,NEXACT)                       LEM05570
  190 FORMAT(I6,E14.3,E25.16,I8)                                        LEM05580
C                                                                       LEM05590
      WRITE(9,200)                                                      LEM05600
  200 FORMAT(' NEXACT DISTINCT A-EIGENVALUES. GAPS IN ASCENDING ORDER'/ LEM05610
     2 ' A-MINGAP(I) = TRUE MINIMUM GAP IN A FOR EACH EIGENVALUE.'/     LEM05620
     3 ' A-MINGAP(I) LT 0 INDICATES THE LEFT-HAND GAP WAS MINIMAL.'/    LEM05630
     3 ' A-MATRIX IS BLOCK TRIDIAGONAL AND EACH DIAGONAL BLOCK IS OF ORDLEM05640
     3ER NX.'/                                                          LEM05650
     4 ' NX = NUMBER OF POINTS ON EACH X-LINE. THERE ARE NY DIAGONAL BLOLEM05660
     4CKS.'/                                                            LEM05670
     5 ' NY = NUMBER OF POINTS ON EACH Y-LINE.'/                        LEM05680
     5 ' A-DIAGONAL   = A(K,K)'/                                        LEM05690
     6 ' X-CODIAGONAL = A(I,I+1)'/                                      LEM05700
     7 ' Y-CODIAGONAL = A(I,I+NX)'/                                     LEM05710
     8 ' ----- END OF FILE 9 TRUEEV------------------------------'//)   LEM05720
C                                                                       LEM05730
    4 RETURN                                                            LEM05740
C                                                                       LEM05750
C-----END OF EXEVG------------------------------------------------------LEM05760
      END                                                               LEM05770
C                                                                       LEM05780
C-----START OF EXERR----------------------------------------------------LEM05790
C                                                                       LEM05800
C     FOR GIVEN COMPUTED EIGENVALUES, V(I), I=1,2,...,NG                LEM05810
C     COMPUTES THE CLOSEST TRUE EIGENVALUES AND THE ERROR IN THE        LEM05820
C     COMPUTED EIGENVALUES, AND STORES THESE RESPECTIVELY               LEM05830
C     IN U(I) AND IN G(MEV+I).  THESE QUANTITIES ARE WRITTEN            LEM05840
C     TO FILE 10.                                                       LEM05850
C                                                                       LEM05860
      SUBROUTINE EXERR(V,U,G,MP,MEV,NG,NEXACT,IWRITE)                   LEM05870
C                                                                       LEM05880
C-----------------------------------------------------------------------LEM05890
      DOUBLE PRECISION U(1),V(1)                                        LEM05900
      DOUBLE PRECISION EV,EE,T0,T1,C0,C1,C2,PIK,PIL                     LEM05910
      DOUBLE PRECISION ATOLN,EPSM,MACHEP,ZERO,ONE,TWO                   LEM05920
      REAL G(1)                                                         LEM05930
      INTEGER MP(1)                                                     LEM05940
```

```
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.0D0*MACHEP
C-----------------------------------------------------------------------
C     ON ENTRY V CONTAINS NG GOOD EIGENVALUES OF T(1,MEV)
C     MP CONTAINS THE MULTIPLICITIES OF THESE EIGENVALUES.
C     U(I) = GAP TO NEAREST DISTINCT TMEV I=1,NG
C     U < 0. MEANS GAP IS DUE TO SPURIOUS EV
C
C     ON EXIT G(MEV+I) = ERROR FOR V(I) < 0. IF MULT EV > 1
C     K = MP(I) MEANS |V(I) - U(K)| = MIN
C     MP < 0 MEANS MORE THAN ONE I USES SAME K
C
C     T0 = C2 - 2*C1*COS(PIL*J)
C     U(KP) = T0 - 2*C0*COS(PIK*I)
C     KP = (J-1)*KX + I
C     C2 = ONE
C     C0 = X-CODIAGONAL = INPUT
C     C1 = Y-CODIAGONAL = HALF - C0
C-----------------------------------------------------------------------
      GO TO 3
      ENTRY EXERRP(C0,C1,C2,KX,KY)
      GO TO 4
C-----------------------------------------------------------------------
    3 N = KX*KY
      ZERO = 0.0D0
      ONE  = 1.0D0
      TWO  = 2.0D0
C
C     SET G(I) = GAP FROM GOOD T(MEV) TO NEAREST DISTINCT TMEV I=1,NG
      DO 10 I = 1,NG
      G(I) = U(I)
   10 CONTINUE
C
C     REGENERATE A-EIGENVALUES
      T0 = DARCOS(-ONE)
      T1 = DFLOAT(KX+1)
      PIK = T0/T1
      T1 = DFLOAT(KY+1)
      PIL = T0/T1
      KP = 0
C
      DO 30 J = 1,KY
      T1 = PIL*DFLOAT(J)
      T0 = C2 - TWO*C1*DCOS(T1)
      DO 20 I = 1,KX
      KP = KP+1
      T1 = PIK*DFLOAT(I)
   20  U(KP) = T0 - TWO*C0*DCOS(T1)
   30 CONTINUE
C
C     ORDER U VECTOR BY INCREASING ALGEBRAIC SIZE
      DO 50 K = 2,N
      KM1 = K-1
      DO 40 L = 1,KM1
```

```
      JJ = K-L
      IF (U(JJ+1).GE.U(JJ)) GO TO 50
      T0 = U(JJ)
      U(JJ) = U(JJ+1)
   40 U(JJ+1) = T0
   50 CONTINUE
C
      ATOLN = DMAX1(DABS(U(1)),DABS(U(N)))*EPSM
C
C     DETERMINE MULTIPLICITIES FOR TRUE EIGENVALUES
      I = 1
      J = 1
      NEXACT = 0
   60 J = J+1
      IF (J.GT.N) GO TO 70
      EE = DABS(U(J)-U(I))
      IF (EE.GT.ATOLN) GO TO 70
      IDEX = IDEX+1
      GO TO 60
   70 NEXACT = NEXACT+1
      U(NEXACT) = U(I)
      I = J
      IF (I.GT.N) GO TO 80
      GO TO 60
   80 CONTINUE
C
C     NEXACT = NUMBER OF DISTINCT A-EIGENVALUES
C     U CONTAINS TRUE DISTINCT A-EV ORDERED BY INCREASING SIZE
C
      IF ( IWRITE.EQ.1) WRITE(6,90)MEV,NG,NEXACT
   90 FORMAT(/3I6,' = MEV, NG, NEXACT, POISZ CASE'/
     1 ' TRUE ERRORS FOR GOOD EIGENVALUES'/)
C
C      WRITE(6,61) (K,U(K), K=1,NEXACT)
C  61 FORMAT(4(I5,E15.8))
C
C     CALCULATION OF THE TRUE ERRORS.
      KL = 1
      DO 110 ITEV = 1,NG
      EV = V(ITEV)
      K = KL
      T1 = DABS(EV - U(KL))
C
      DO 100 KP = KL,NEXACT
      T0 = DABS(EV - U(KP))
      IF (T0.GE.T1) GO TO 100
      K = KP
      T1 = T0
  100 CONTINUE
C
      IF (K.EQ.KL.AND.ITEV.GT.1) T1 = -T1
      KL = K
      MP(ITEV) = K
      G(MEV+ITEV) = T1
  110 CONTINUE
```

```
C                                                                       LEM07050
C     TRUE ERRORS HAVE BEEN COMPUTED OUTPUT THEM TO FILE 10             LEM07060
C     FORM HEADER FOR ERREXACT FILE 10                                  LEM07070
      WRITE(10,120)N,KX,KY,C2,C0,C1                                     LEM07080
  120 FORMAT(' POISSONZ TRUE ERROR FOR GOOD EIGENVALUES'/               LEM07090
     1 5X,'N',4X,'NX',4X,'NY'/3I6//                                     LEM07100
     2 5X,'A-DIAGONAL',3X,'X-CODIAGONAL',3X,'Y-CODIAGONAL'/3E15.8//)    LEM07110
C                                                                       LEM07120
      WRITE(10,130)MEV,NG,NEXACT                                        LEM07130
  130 FORMAT(/3I6,' = MEV,NG,NEXACT'/1X,'T-EV NO',1X,'A-EV NO',         LEM07140
     1 10X,'GOOD EIGENVALUE',5X,'TRUEERROR',7X,'TMINGAP')               LEM07150
C                                                                       LEM07160
      WRITE(10,140)(I,MP(I),V(I),G(MEV+I),G(I), I=1,NG)                 LEM07170
  140 FORMAT(2I8,E25.16,2E14.3)                                         LEM07180
C                                                                       LEM07190
      WRITE(10,150)                                                     LEM07200
  150 FORMAT(' ABOVE ARE THE TRUE ERRORS FOR POISSON GOODEV'/           LEM07210
     1 ' IF A-EV NO LT 0 THEN GOODEV HAS MULTIPLICITY GT 1'/            LEM07220
     1 ' IF TRUE ERROR LT 0 THEN MORE THAN ONE GOODEV APPROXIMATES'/    LEM07230
     1 ' THE SAME TRUE POISSON EIGENVALUE'/                             LEM07240
     1 ' IF TMINGAP LT 0 THE MINGAP IS DUE TO SPURIOUS EIGENVALUE'//)   LEM07250
C                                                                       LEM07260
    4 RETURN                                                            LEM07270
C                                                                       LEM07280
C-----END OF EXERR------------------------------------------------------LEM07290
      END                                                               LEM07300
C                                                                       LEM07310
C-----START OF EXVEC----------------------------------------------------LEM07320
C                                                                       LEM07330
C     (JVEC = 1): FOR A GIVEN RITZ VECTOR V AND EIGENVALUE X1, COMPUTES LEM07340
C     THE CLOSEST EIGENVALUE Y1 AND CORRESPONDING TRUE EIGENVECTOR U,   LEM07350
C     AND THEN CALCULATES THE NORM OF THE DIFFERENCE BETWEEN            LEM07360
C     V AND U AND THE MAXIMAL DIFFERENCE BETWEEN THE COMPONENTS.        LEM07370
C     THESE QUANTITIES ARE WRITTEN TO FILE 6.                           LEM07380
C                                                                       LEM07390
C     (JVEC = 2):  COMPUTES THE PROJECTION OF EACH                      LEM07400
C     OF THE TRUE EIGENVECTORS ON THE LANCZOS STARTING VECTOR           LEM07410
C     USED BY THE LANCZS SUBROUTINE AND WRITES THEM TO FILE 12.         LEM07420
C                                                                       LEM07430
      SUBROUTINE EXVEC(U,V,X1,Y1,G,MP,IIX,JVEC,ICOUNT)                  LEM07440
C                                                                       LEM07450
C-----------------------------------------------------------------------LEM07460
      DOUBLE PRECISION U(*),V(1)                                        LEM07470
      DOUBLE PRECISION WI(110),WJ(110),WII(110)                         LEM07480
      DOUBLE PRECISION X1,Y1,EV,EE,WS,PIK,PIL,SUM,PROJ,TEMP,S           LEM07490
      DOUBLE PRECISION ATOLN,EPSM,MACHEP,ZERO,HALF,ONE,TWO              LEM07500
      DOUBLE PRECISION C0,C1,C2,T0,T1,T2                                LEM07510
      REAL G(1),GG                                                      LEM07520
      INTEGER MP(1)                                                     LEM07530
      DOUBLE PRECISION FINPRO                                           LEM07540
C-----------------------------------------------------------------------LEM07550
C     THIS PROGRAM CALCULATES THE TRUE EIGENVALUES AND EIGENVECTORS     LEM07560
C     OF THE POISSON MATRIX A OF ORDER  N = KX*KY                       LEM07570
C     A CONSISTS OF KY TRIDIAGONAL BLOCKS OF ORDER KX                   LEM07580
C     KX = X-DIMENSION     KY = Y-DIMENSION.                            LEM07590
```

```
C
C     IIX = SEED FOR RANDOM NUMBER GENERATOR USED TO CALCULATE
C           STARTING LANCZOS VECTOR IN LANCZS
C     V = RANDOM UNIT STARTING VECTOR FOR LANCZS
C     A*U = EV*U   ||U|| = ONE
C
C     C2 = DIAGONAL OF KX BY KX MATRIX
C    -C0 = CO-DIAGONAL OF THE KX BY KX MATRIX.
C    -C1 = Y-CODIAGONAL.
C
C     NOTE THAT THE VECTORS WI,WJ,WII ARE DIMENSIONED INTERNALLY
C     THEY ARE USED JUST TO KEEP FROM REGENERATING INFORMATION.
C     WI,WII = REAL*8 ARRAYS OF DIMENSION AT LEAST KX
C     WJ     = REAL*8 ARRAY  OF DIMENSION AT LEAST KY.
C
C     NOTATION USED IN PROGRAM
C
C     PIK = ARCOS(-1)/(KX+1)    PIL = ARCOS(-1)/(KY+1)
C     WI(I) = PIK*I        WJ(J) = PIL*J
C
C     U(K) IS A-EV ORDERED BY INCREASING SIZE, K = 1,N
C     LATER U IS USED TO STORE THE TRUE EIGENVECTOR
C     T0 = C2 - 2*C1*COS(PIL*J)    EV(I,J) = T0 - 2*C0*COS(PIK*I)
C     I = 1,KX    J = 1,KY    KP = (J-1)*KX + I
C
C     U(KV) = SIN(PIK*I*IK)*SIN(PIL*J*JK)
C     IK = 1,KX    JK = 1,KY    KV = (JK-1)*KX + IK
C     U IS UNSCALED EIGENVECTOR FOR EV(I,J) = Y1
C     WS = 1/||U||: ||U|| = .5*DSQRT(T2*T3)  T2 = KX+1  T3 = KY+1
C
C    JVEC  = (1,2) FLAGS COMPUTATIONS TO BE PERFORMED.
C
C          = (1) MEANS GIVEN X1 FIND Y1 AND KVEC SUCH THAT
C                       Y1 = EV(KVEC) AND |X1-Y1| = MIN
C                ALSO GIVEN UNIT RITZ VECTOR ASSOCIATED WITH X1
C                     CALCULATE UNIT EIGENVECTOR U, A*U = Y1*U
C                     T2 = ||V-U||  T1 = MAX(|V(K)-U(K)|, K= 1,N)
C                     MAX OCCURS FIRST AT K = KK
C
C          = (2) MEANS CALCULATION OF THE PROJECTION OF THE STARTING
C                     LANCZOS VECTOR ON EACH EIGENVECTOR OF A.
C
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.0D0*MACHEP
C-----------------------------------------------------------------------
      GO TO 3
      ENTRY EXVECP(C0,C1,C2,KX,KY)
      GO TO 4
C-----------------------------------------------------------------------
C     SPECIFY PARAMETERS
    3 N = KX*KY
      ZERO = 0.0D0
      HALF = 0.5D0
      ONE  = 1.0D0
```

```
      TWO  = 2.0D0                                                      LEM08150
      T0 = DARCOS(-ONE)                                                 LEM08160
      T1 = DFLOAT(KX+1)                                                 LEM08170
      PIK = T0/T1                                                       LEM08180
      T2 = DFLOAT(KY+1)                                                 LEM08190
      PIL = T0/T2                                                       LEM08200
      WS = TWO/DSQRT(T1*T2)                                             LEM08210
C                                                                       LEM08220
C     GENERATE WI WJ VECTORS                                            LEM08230
      KP = 0                                                            LEM08240
      DO 20 J = 1,KY                                                    LEM08250
      T1 = PIL*DFLOAT(J)                                                LEM08260
      WJ(J) = T1                                                        LEM08270
      T0 = C2 - TWO*C1*DCOS(T1)                                         LEM08280
      DO 10 I = 1,KX                                                    LEM08290
      KP = KP+1                                                         LEM08300
      T1 = PIK*DFLOAT(I)                                                LEM08310
      WI(I) = T1                                                        LEM08320
   10 U(KP) = T0 - TWO*C0*DCOS(T1)                                      LEM08330
   20 CONTINUE                                                          LEM08340
C     U(KP) = EV(I,J) = C2 - 2*C1*COS(PIL*J) - 2*C0*COS(PIK*I)          LEM08350
C                                                                       LEM08360
C     INITIALIZE MP VECTOR                                              LEM08370
      DO 30 K = 1,N                                                     LEM08380
   30 MP(K) = K                                                         LEM08390
C                                                                       LEM08400
C     WE ORDER U VECTOR BY INCREASING SIZE OF THE EVS                   LEM08410
      DO 50 K = 2,N                                                     LEM08420
      KM1 = K-1                                                         LEM08430
C                                                                       LEM08440
      DO 40 L = 1,KM1                                                   LEM08450
      JJ = K - L                                                        LEM08460
      IF (U(JJ+1).GE.U(JJ)) GO TO 50                                    LEM08470
      EE = U(JJ)                                                        LEM08480
      U(JJ) = U(JJ+1)                                                   LEM08490
      U(JJ+1) = EE                                                      LEM08500
      IEE = MP(JJ)                                                      LEM08510
      MP(JJ) = MP(JJ+1)                                                 LEM08520
   40 MP(JJ+1) = IEE                                                    LEM08530
C                                                                       LEM08540
   50 CONTINUE                                                          LEM08550
C                                                                       LEM08560
      ATOLN = DMAX1(DABS(U(1)),DABS(U(N)))*EPSM                         LEM08570
C                                                                       LEM08580
      IF (ICOUNT.EQ.1) WRITE(6,60) N,KX,KY,JVEC,C2,C0,C1,ATOLN          LEM08590
   60 FORMAT(/' TRUE ERRORS FOR CONVERGED GOODEV'/                      LEM08600
     1 4I6,' = N KX KY JVEC'//                                          LEM08610
     1 4E12.5,' = C2 C0 C1 ATOLN'//)                                    LEM08620
C                                                                       LEM08630
C     KP = MP(K) MEANS EIGENVALUE U(K) CORRESPONDS TO EIGENVECTOR W(KP) LEM08640
C     COMPUTE TOLERANCE USED IN COMPUTING TRUE MULTIPLICITIES           LEM08650
C                                                                       LEM08660
      IF (JVEC.EQ.1) GO TO 180                                          LEM08670
C                                                                       LEM08680
C     JVEC = 2 SO CALCULATE PROJECTIONS AND WRITE IN FILE 12            LEM08690
```

```
C
      WRITE(12,70)
   70 FORMAT(' PROJECTIONS OF LANCZOS STARTING VECTOR ON A-EIGENVECS')
C
      WRITE(12,80)N,KX,KY,IIX,C2,C0,C1,ATOLN
   80 FORMAT(1X,'A-SIZE',2X,'X-DIM',2X,'Y-DIM',6X,'SVSEED'/3I7,I12/
     1 5X,'A-DIAGONAL',3X,'X-CODIAGONAL',3X,'Y-CODIAGONAL',5X,'ATOLN'/
     2 3E15.8,E10.3)
C
      WRITE(12,90)
   90 FORMAT(5X,'PROJECTION',8X,'TRUE A-EIGENVALUE',1X,'EV NO'
     1,2X,'VEC NO')
C
C     GENERATE SAME RANDOM UNIT VECTOR USED IN THE LANCZS RECURSIONS.
      IIL=IIX
C
C-----------------------------------------------------------------------
      CALL GENRAN(IIL,G,N)
C-----------------------------------------------------------------------
C
      DO 100 I = 1,N
  100 V(I) = G(I)
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V(1),1,V(1),1)
C-----------------------------------------------------------------------
C
      SUM = 1.D0/DSQRT(SUM)
C
      DO 110 I = 1,N
  110 V(I) = V(I)*SUM
C
C     DETERMINE UNIT EIGENVECTOR W ASSOCIATED WITH EACH EV(I,J) = Y1
C     AND CALCULATE THE PROJECTION G(K) OF U ON THE STARTING VECTOR V
C     A*U = EV*U       WS = 1/||WU||: WU = UNSCALED EIGENVECTOR
C
      DO 160 K =1,N
C     DETERMINE I J FROM K: MP(K) = KP = (J-1)*KX+I
      KP = MP(K)
      I = MOD(KP,KX)
      IF (I.EQ.0) I = KX
      T1 = WI(I)
      J = 1 + (KP-1)/KX
      T0 = WJ(J)
      T0 = WJ(J)
C
      Y1 = C2 - TWO*C1*DCOS(WJ(J)) - TWO*C0*DCOS(WI(I))
C     Y1 = EV(I,J)
C
      DO 120 II = 1,KX
      T2 = T1*DFLOAT(II)
  120 WII(II) = WS*DSIN(T2)
C
      KV = 0
      DO 140 JJ = 1,KY
```

```
      T2 = T0*DFLOAT(JJ)
      T2 = DSIN(T2)
C
      DO 130 II = 1,KX
      KV = KV + 1
  130 U(KV) = T2*WII(II)
C
  140 CONTINUE
C
C     U IS UNIT EIGENVECTOR OF A ASSOCIATED WITH EV(I,J) = Y1
C     G(K) IS THE PROJECTION OF U ON V  FOR Y1
C
C-----------------------------------------------------------------------
      PROJ = FINPRO(N,U(1),1,V(1),1)
C-----------------------------------------------------------------------
C
      TEMP = DABS(PROJ)
      G(K) = TEMP
C
C     DESIRED PROJECTION HAS BEEN COMPUTED OUTPUT IT TO FILE 12.
      WRITE(12,150) G(K),Y1,K,MP(K)
  150 FORMAT(E15.8,E25.16,I6,I8)
C
  160 CONTINUE
C
      WRITE(12,170)
  170 FORMAT(' ----- END OF FILE 12 PROJECT -----------------------'//)
C
      GO TO 310
C
C     JVEC = 1
C
C     X1 IS AN INPUT PARAMETER. WE CALCULATE TRUE
C     A-EIGENVALUE WHICH IS CLOSEST TO X1, LABEL IT Y1 AND CALCULATE
C     UNIT EIGENVECTOR OF A ASSOCIATED WITH Y1. A*U = Y1*U, ||U|| = 1.
C     Y1 = EV(I,J). EIGENVALUES OF A ARE ORDERED BY INCREASING SIZE.
C     V = RITZ VECTOR ASSOCIATED WITH GOODEV X1
C
  180 CONTINUE
      KX1 = 0
      IF (X1.LE.U(1)) KX1 = 1
      IF (X1.GE.U(N)) KX1 = N
      NM1 = N-1
      IF (KX1.NE.0) GO TO 200
C
      DO 190 KVEC = 2,N
      IF (X1.GE.U(KVEC)) GO TO 190
C     U(KVEC-1).LE.X1.LT.U(KVEC)
      T1 = X1 - U(KVEC-1)
      T2 = U(KVEC) - X1
      KX1 = KVEC - 1
      IF (T1.GT.T2) KX1 = KVEC
      GO TO 200
  190 CONTINUE
C
```

```
  200 Y1 = U(KX1)                                                       LEM09800
C                                                                       LEM09810
      IF (KX1.EQ.1) EE = U(2) - U(1)                                    LEM09820
      IF (KX1.EQ.N) EE = U(N) - U(NM1)                                  LEM09830
      IF (KX1.EQ.1.OR.KX1.EQ.N) GO TO 210                               LEM09840
      EE = DMIN1(U(KX1+1)-U(KX1),U(KX1)-U(KX1-1))                       LEM09850
  210 CONTINUE                                                          LEM09860
C                                                                       LEM09870
      T0 = DABS(ONE - X1/Y1)                                            LEM09880
C                                                                       LEM09890
      WRITE(6,220) N,KX1,ICOUNT,Y1,X1,T0,EE                             LEM09900
  220 FORMAT(3I8,' = N, A-EV NUMBER,RITZ NUMBER'//                      LEM09910
     1 18X,' TRUEEV',19X,'GOODEV',4X,'RELERROR',4X,'A-MINGAP'/          LEM09920
     1 2E25.16,2E12.3/)                                                 LEM09930
C                                                                       LEM09940
      IF (EE.GT.ATOLN) GO TO 240                                        LEM09950
C                                                                       LEM09960
      WRITE(6,230)                                                      LEM09970
  230 FORMAT(' Y1 IS A MULTIPLE EIGENVALUE OF A SO WE EXIT'/)           LEM09980
C                                                                       LEM09990
      GO TO 310                                                         LEM10000
C                                                                       LEM10010
C     Y1 IS TOEPLITZ EIGENVALUE CLOSEST TO X1.                          LEM10020
C     CALCULATION OF EIGENVECTOR ASSOCIATED WITH EIGENVALUE Y1          LEM10030
C     A*U = Y1*U                                                        LEM10040
C     DETERMINE I J FROM K: MP(K) = KP = (J-1)*KX+I                     LEM10050
  240 CONTINUE                                                          LEM10060
      K = KX1                                                           LEM10070
      KP = MP(K)                                                        LEM10080
      I = MOD(KP,KX)                                                    LEM10090
      IF (I.EQ.0) I = KX                                                LEM10100
      T1 = WI(I)                                                        LEM10110
      J = 1 + (KP-1)/KX                                                 LEM10120
      T2 = WJ(J)                                                        LEM10130
C                                                                       LEM10140
      DO 250 II = 1,KX                                                  LEM10150
      T0 = T1*DFLOAT(II)                                                LEM10160
  250 WII(II) = WS*DSIN(T0)                                             LEM10170
C                                                                       LEM10180
      KV = 0                                                            LEM10190
      DO 270 JJ = 1,KY                                                  LEM10200
      T0 = T2*DFLOAT(JJ)                                                LEM10210
      T0 = DSIN(T0)                                                     LEM10220
C                                                                       LEM10230
      DO 260 II = 1,KX                                                  LEM10240
      KV = KV + 1                                                       LEM10250
  260 U(KV) = T0*WII(II)                                                LEM10260
C                                                                       LEM10270
  270 CONTINUE                                                          LEM10280
C                                                                       LEM10290
C     U IS UNIT TRUE EIGENVECTOR OF A ASSOCIATED WITH Y1                LEM10300
C     V IS UNIT RITZVECTOR OF A ASSOCIATED WITH X1                      LEM10310
C                                                                       LEM10320
      KK = 0                                                            LEM10330
      S = ONE                                                           LEM10340
```

```
      T1 = ZERO                                                         LEM10350
C                                                                       LEM10360
      DO 280 K = 1,N                                                    LEM10370
      IF (DABS(U(K)).LE.T1) GO TO 280                                   LEM10380
      T1 = DABS(U(K))                                                   LEM10390
      KK = K                                                            LEM10400
  280 CONTINUE                                                          LEM10410
      IF (U(KK)*V(KK).LT.ZERO) S = - ONE                                LEM10420
C                                                                       LEM10430
      KK = 0                                                            LEM10440
      T1 = ZERO                                                         LEM10450
      T2 = ZERO                                                         LEM10460
      DO 290 K = 1,N                                                    LEM10470
      TEMP = DABS(S*U(K) - V(K))                                        LEM10480
      T2 = T2 + TEMP**2                                                 LEM10490
      IF (TEMP.LE.T1) GO TO 290                                         LEM10500
      KK = K                                                            LEM10510
      T1 = TEMP                                                         LEM10520
  290 CONTINUE                                                          LEM10530
C                                                                       LEM10540
      T2 = DSQRT(T2)                                                    LEM10550
      WRITE(6,300) KK,T1,T2                                             LEM10560
  300 FORMAT(' EIGENVECTOR ERROR. MAX ERROR AT COMPONENT = ',I6/        LEM10570
     1 ' MAX DABS(TRUEVEC(K)-RITZVEC(K)) = ',E12.5/                     LEM10580
     1 ' NORM(TRUEVEC-RITZVEC)  = ',E12.5/)                             LEM10590
C                                                                       LEM10600
  310 CONTINUE                                                          LEM10610
C                                                                       LEM10620
    4 RETURN                                                            LEM10630
C                                                                       LEM10640
C-----END OF EXVEC------------------------------------------------------LEM10650
      END                                                               LEM10660
```

2.6 LESUB: Other Subroutines used by the Codes in Chapters 2, 3, 4, 5

```
C-------------------------------------------------------------------    LES00010
C                                                                       LES00020
C     LESUB                                                             LES00030
C                                                                       LES00040
C-----------------------------------------------------------------------LES00050
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)             LES00060
C            Los Alamos National Laboratory                             LES00070
C            Los Alamos, New Mexico 87544                               LES00080
C                                                                       LES00090
C            E-mail:  cullumj@lanl.gov                                  LES00100
C                                                                       LES00110
C  These codes are copyrighted by the authors.  These codes             LES00120
C  and modifications of them or portions of them are NOT to be          LES00130
C  incorporated into any commercial codes or used for any other         LES00140
C  commercial purposes such as consulting for other companies,          LES00150
C  without legal agreements with the authors of these Codes.            LES00160
C  If these Codes or portions of them are used in other scientific or   LES00170
C  engineering research works the names of the authors of these codes   LES00180
C  and appropriate references to their written work are to be           LES00190
C  incorporated in the derivative works.                                LES00200
C                                                                       LES00210
C  This header is not to be removed from these codes.                   LES00220
C                                                                       LES00230
C         REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4            LES00231
C         Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsLES00232
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   LES00233
C         Applied Mathematics, 2002. SIAM Publications,                 LES00234
C         Philadelphia, PA. USA                                         LES00235
C                                                                       LES00236
C                                                                       LES00237
C            (1) REAL SYMMETRIC                                         LES00240
C            (2) HERMITIAN MATRICES                                     LES00250
C            (3) FACTORED INVERSES OF REAL SYMMETRIC MATRICES AND       LES00260
C            (4) REAL SYMMETRIC GENERALIZED, A*X = EVAL*B*X WHERE       LES00270
C                B IS POSITIVE DEFINITE, CHOLESKY FACTOR AVAILABLE      LES00280
C                                                                       LES00290
C     ACCORDING TO PFORT THESE SUBROUTINES ARE PORTABLE EXCEPT FOR:     LES00300
C     (1) THE COMPLEX*16 VARIABLES AND THE CORRESPONDING FUNCTIONS      LES00310
C         FOR COMPLEX VARIABLES, DCMPLX, DREAL AND DCONJG USED IN       LES00320
C         THE SUBROUTINE CINPRD (USED ONLY IN CASE (2), HERMITIAN)      LES00330
C     (2) THE ENTRY IN THE SUBROUTINE LPERM USED TO PASS THE            LES00340
C         PERMUTATION FROM THE UPSEC SUBROUTINE TO LPERM. (USED         LES00350
C         ONLY IN CASES (3) AND (4), INVERSE AND GENERALIZED).          LES00360
C                                                                       LES00370
C     SUBROUTINES    BISEC, INVERR, TNORM, LUMP, ISOEV, PRTEST, AND     LES00380
C                    INVERM ARE USED WITH THE LANCZOS EIGENVALUE        LES00390
C                    PROGRAMS LEVAL, HLEVAL, LIVAL AND LGVAL. STURMI,   LES00400
C                    INVERM, LBISEC, AND TNORM ARE USED WITH THE        LES00410
C                    EIGENVECTOR PROGRAMS LEVEC, HLEVEC, LIVEC AND      LES00420
C                    LGVEC.  LPERM IS USED WITH LIVEC AND LGVEC.        LES00430
```

```
C                    IN THE HERMITIAN CASE, THE SUBROUTINE CINPRD
C                    IS ALSO USED.
C
C-----COMPUTE T-EIGENVALUES BY BISECTION--------------------------------
C
      SUBROUTINE BISEC(ALPHA,BETA,BETA2,VB,VS,LBD,UBD,EPS,TTOL,MP,
     1 NINT,MEV,NDIS,IC,IWRITE)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  ALPHA(1),BETA(1),BETA2(1),VB(1),VS(1)
      DOUBLE PRECISION  LBD(1),UBD(1),EPS,EPT,EP0,EP1,TEMP,TTOL
      DOUBLE PRECISION  ZERO,ONE,HALF,YU,YV,LB,UB,XL,XU,X1,X0,XS,BETAM
      INTEGER  MP(1),IDEF(100)
      DOUBLE PRECISION  DABS, DSQRT, DMAX1, DMIN1, DFLOAT
C-----------------------------------------------------------------------
C     COMPUTES EIGENVALUES OF T(1,MEV) BY LOOPING INTERNALLY ON THE
C     USER-SPECIFIED INTERVALS,  (LB(J),UB(J)), J = 1,NINT.  INTERVALS
C     ARE TREATED AS OPEN ON THE LEFT AND CLOSED ON THE RIGHT.
C     THE BISEC SUBROUTINE SIMULTANEOUSLY LABELS SPURIOUS T-EIGENVALUES
C     AND DETERMINES THE T-MULTIPLICITIES OF EACH GOOD T-EIGENVALUE.
C     SPURIOUS T-EIGENVALUES ARE LABELLED BY A T-MULTIPLICITY = 0.
C     ANY T-EIGENVALUE WITH A T-MULTIPLICITY >= 1 IS 'GOOD'.
C
C     IF IWRITE = 0 THEN MOST OF THE WRITES TO FILE 6 ARE NOT
C     ACTIVATED.
C
C     NOTE THAT PROGRAM ASSUMES THAT NO MORE THAN MMAX/2 EIGENVALUES
C     OF T(1,MEV) ARE TO BE COMPUTED IN ANY ONE OF THE SUBINTERVALS
C     CONSIDERED, WHERE MMAX = DIMENSION OF VB SPECIFIED BY THE USER
C     IN THE MAIN PROGRAM LEVAL.
C
C     ON ENTRY
C     BETA2(J) IS SET = BETA(J)*BETA(J).  THE STORAGE FOR BETA2 COULD
C     BE ELIMINATED BY RECOMPUTING THE BETA(J)**2 FOR EACH STURM
C     SEQUENCE.
C
C     EPS = 2*MACHEP =  4.4 * 10**-16 ON IBM 3081.
C     TTOL = EPS*TKMAX WHERE
C     TKMAX = MAX(|ALPHA(K)|,BETA(K), K=1,KMAX)
C
C     ON EXIT
C     NDIS = TOTAL NUMBER OF COMPUTED DISTINCT EIGENVALUES OF
C            T(1,MEV) ON THE UNION OF THE (LB,UB) INTERVALS.
C     VS = COMPUTED DISTINCT EIGENVALUES OF T(1,MEV) IN ALGEBRAICALLY-
C          INCREASING ORDER
C     MP = CORRESPONDING T-MULTIPLICITIES OF THESE EIGENVALUES
C     MP(I) = (0,1,MI), MI>1, I=1,NDIS  MEANS:
C        (0)  V(I) IS SPURIOUS
C        (1)  V(I) IS ISOLATED AND GOOD
C        (MI) V(I) IS MULTIPLE AND HENCE A CONVERGED GOOD T-EIGENVALUE.
C     IC = TOTAL NUMBER OF STURMS USED
C
C     DEFAULTS
C     ISKIP = 0 INITIALLY. IF DEFAULT OCCURS ON J-TH SUB-INTERVAL, SET
C             ISKIP=ISKIP+1 AND IDEF(ISKIP) = J
```

```
C            DEFAULTS OCCUR IF THERE ARE NO T-EIGENVALUES  IN THE
C            SUBINTERVAL SPECIFIED OR IF THE NUMBER
C            OF STURMS SEQUENCES REQUIRED EXCEEDS MXSTUR.
C            WHEN A DEFAULT OCCURS THE PROGRAM
C            SKIPS THE INTERVAL INVOLVED AND GOES ON TO THE NEXT
C            INTERVAL.
C
C-----------------------------------------------------------------------
C     SPECIFY PARAMETERS
      ZERO = 0.0D0
      ONE  = 1.0D0
      HALF = 0.5D0
      MXSTUR = IC
      NDIS = 0
      IC = 0
      ISKIP = 0
      MP1 = MEV+1
C     SAVE THEN SET BETA(MEV+1) = 0. GENERATE BETA**2
      BETAM = BETA(MP1)
      BETA(MP1) = ZERO
C
      DO 10 I = 1,MP1
   10 BETA2(I) = BETA(I)*BETA(I)
C
C     EP0 IS USED IN T-MULTIPLICITY AND SPURIOUS TESTS
C     EP1 AND EPS ARE USED IN THE BISEC CONVERGENCE TEST
C
      TEMP = DFLOAT(MEV+1000)
      EP0  = TEMP*TTOL
      EP1  = DSQRT(TEMP)*TTOL
C
      WRITE(6,20)MEV,NINT
   20 FORMAT(/' BISEC CALCULATION'/' ORDER OF T IS',I6/
     1' NUMBER OF INTERVALS IS',I6/)
C
      WRITE(6,30) EP0,EP1
   30 FORMAT(/' MULTOL, TOLERANCE USED IN T-MULTIPLICITY AND SPURIOUS TE
     1STS = ',E10.3/' BISTOL, TOLERANCE USED IN BISEC CONVERGENCE TEST =
     1 ',E10.3/)
C
C     LOOP ON THE NINT INTERVALS  (LB(J),UB(J)), J=1,NINT
      DO 430 JIND = 1,NINT
      LB = LBD(JIND)
      UB = UBD(JIND)
C
      WRITE(6,40)JIND,LB,UB
   40 FORMAT(//1X,'BISEC INTERVAL NO',2X,'LOWER BOUND',2X,'UPPER BOUND'/
     1I18,2E13.5/)
C
C     INITIALIZATION AND PARAMETER SPECIFICATION
C     ICT IS TOTAL STURM COUNT ON (LB,UB)
C
      NA = 0
      MD = 0
      NG = 0
```

```
      ICT = 0                                                           LES01540
C                                                                       LES01550
C     START OF T-EIGENVALUE CALCULATIONS                                LES01560
      X1 = UB                                                           LES01570
      ISTURM = 1                                                        LES01580
      GO TO 330                                                         LES01590
C     FORWARD STURM CALCULATION TO DETERMINE NA = NO. T-EIGENVALUES > UBLES01600
   50 NA = NEV                                                          LES01610
C                                                                       LES01620
      X1 = LB                                                           LES01630
      ISTURM = 2                                                        LES01640
      GO TO 330                                                         LES01650
C     FORWARD STURM CALC TO DETERMINE MT = NO. T-EIGENVALUES ON (LB,UB) LES01660
   60 CONTINUE                                                          LES01670
      MT=NEV                                                            LES01680
      ICT = ICT +2                                                      LES01690
C                                                                       LES01700
      WRITE(6,70)MT,NA                                                  LES01710
   70 FORMAT(/2I6,' = NO. TMEV ON (LB,UB) AND NO. .GT. UB'/)            LES01720
C                                                                       LES01730
C     DEFAULT TEST: IS ESTIMATED NUMBER OF STURMS > MXSTUR?             LES01740
      IEST = 30*MT                                                      LES01750
      IF (IEST.LT.MXSTUR) GO TO 90                                      LES01760
C                                                                       LES01770
      WRITE(6,80)                                                       LES01780
   80 FORMAT(//' ESTIMATED NUMBER OF STURMS REQUIRED EXCEEDS USER LIMIT'LES01790
     1/' SKIP THIS SUBINTERVAL')                                        LES01800
      GO TO 110                                                         LES01810
C                                                                       LES01820
   90 CONTINUE                                                          LES01830
C                                                                       LES01840
      IF (MT.GE.1) GO TO 120                                            LES01850
C                                                                       LES01860
      WRITE(6,100)                                                      LES01870
  100 FORMAT(//' THERE ARE NO T-EIGENVALUES ON THIS INTERVAL)'/)        LES01880
C                                                                       LES01890
  110 ISKIP = ISKIP+1                                                   LES01900
      IDEF(ISKIP) = JIND                                                LES01910
      GO TO 430                                                         LES01920
C                                                                       LES01930
C     REGULAR CASE.                                                     LES01940
  120 CONTINUE                                                          LES01950
C                                                                       LES01960
      IF (IWRITE.NE.0) WRITE(6,130)                                     LES01970
  130 FORMAT(/' DISTINCT T-EIGENVALUES COMPUTED USING BISEC'/           LES01980
     1 13X,'T-EIGENVALUE',2X,'TMULT',3X,'MD',4X,'NG')                   LES01990
C                                                                       LES02000
C     SET UP INITIAL UPPER AND LOWER BOUNDS FOR T-EIGENVALUES           LES02010
      DO 140 I=1,MT                                                     LES02020
      VB(I) = LB                                                        LES02030
      MTI = MT + I                                                      LES02040
  140 VB(MTI) = UB                                                      LES02050
C                                                                       LES02060
C     CALCULATE T-EIGENVALUES FROM LB UP TO UB  K = MT,...,1            LES02070
C     MAIN LOOP FOR FINDING KTH T-EIGENVALUE                            LES02080
```

```
C
      K = MT
  150 CONTINUE
      ICO = 0
      XL = VB(K)
      MTK = MT+K
      XU = VB(MTK)
C
      ISTURM = 3
      X1 = XU
      ICO = ICO + 1
      GO TO 330
C     FORWARD STURM CALCULATION AT XU
  160 NU=NEV
C
C     BISECTION LOOP FOR KTH T-EIGENVALUE. TEST  X1=MIDPOINT OF (XL,XU)
      ISTURM = 4
  170 CONTINUE
      X1 = (XL+XU)*HALF
      XS = DABS(XL)+DABS(XU)
      X0 = XU-XL
      EPT = EPS*XS+EP1
C
C     EPT IS CONVERGENCE TOLERANCE FOR KTH T-EIGENVALUE
C
      IF (X0.LE.EPT) GO TO 230
C
C     T-EIGENVALUE HAS NOT YET CONVERGED
C
      ICO = ICO + 1
      GO TO 330
C     FORWARD STURM CALCULATION AT CURRENT T-EIGENVALUE APPROXIMATION.
  180 CONTINUE
C
C     UPDATE T-EIGENVALUE INTERVAL (XL,XU)
C
      IF (NEV.LT.K) GO TO 190
C
C     NUMBER OF T-EIGENVALUES NEV = K
      XL = X1
      GO TO 170
  190 CONTINUE
C     NUMBER OF T-EIGENVALUES NEV<K
      XU = X1
      NU = NEV
C
C     UPDATE OF T-EIGENVALUE BOUNDS
C
      IF (NEV.EQ.0) GO TO 210
C
      DO 200 I = 1,NEV
  200 VB(I) = DMAX1(X1,VB(I))
C
  210 NEV1 = NEV+1
C
```

```
      DO 220 II = NEV1,K                                                LES02640
      I = MT+II                                                         LES02650
  220 VB(I) = DMIN1(X1,VB(I))                                           LES02660
C                                                                       LES02670
      GO TO 170                                                         LES02680
C                                                                       LES02690
C     END (XL,XU) BISECTION LOOP FOR KTH T-EIGENVALUE ON (LB,UB)        LES02700
C     TEST FOR T-MULTIPLICITY AND IF SIMPLE THEN TEST FOR SPURIOUSNESS  LES02710
C                                                                       LES02720
  230 CONTINUE                                                          LES02730
      NDIS = NDIS+1                                                     LES02740
      MD = MD+1                                                         LES02750
      VS(NDIS) = X1                                                     LES02760
C                                                                       LES02770
      JSTURM = 1                                                        LES02780
      X1 = XL-EP0                                                       LES02790
      GO TO 370                                                         LES02800
C     BACKWARD STURM CALCULATION                                        LES02810
  240 KL = KEV                                                          LES02820
      JL = JEV                                                          LES02830
C                                                                       LES02840
      JSTURM = 2                                                        LES02850
      IC0 = IC0 + 2                                                     LES02860
      X1 = XU+EP0                                                       LES02870
      GO TO 370                                                         LES02880
C     BACKWARD STURM CALCULATION                                        LES02890
  250 JU = JEV                                                          LES02900
      KU = KEV                                                          LES02910
C                                                                       LES02920
C     FOR T(1,MEV)                                                      LES02930
C     NU - KU = NO. T-EIGENVALUES ON (XU, XU + EP0)                     LES02940
C     KL - KU = NO. T-EIGENVALUES ON (XL - EP0, XU + EP0)               LES02950
C                                                                       LES02960
C     FOR T(2,MEV)                                                      LES02970
C     JL -JU = NO. T-EIGENVALUES ON (XL - EP0, XU + EP0)                LES02980
C                                                                       LES02990
C     IS THIS A SIMPLE T-EIGENVALUE?                                    LES03000
C                                                                       LES03010
      IF (KL-KU-1.EQ.0) GO TO 290                                       LES03020
C                                                                       LES03030
C     VS(NDIS) = KTH-T-EIGENVALUE OF (LB,UB) IS MULTIPLE AND HENCE GOOD LES03040
      IF (KU.EQ.NU) GO TO 280                                           LES03050
C     CONTINUE TO CHECK FOR T-MULTIPLICITY                              LES03060
  260 CONTINUE                                                          LES03070
      ISTURM = 5                                                        LES03080
      X1 = X1+EP0                                                       LES03090
      IC0 = IC0 + 1                                                     LES03100
      GO TO 330                                                         LES03110
C     FORWARD STURM CALCULATION                                         LES03120
  270 KNE = KU-NEV                                                      LES03130
      KU = NEV                                                          LES03140
      IF (KNE.NE.0) GO TO 260                                           LES03150
C     SPECIFY T-MULTIPLICITY = MP(NDIS)                                 LES03160
  280 MPEV = KL-KU                                                      LES03170
      KNEW = KU                                                         LES03180
```

```
      GO TO 300
C     END MULTIPLE CASE
C
C     T-EIGENVALUE IS SIMPLE   CHECK IF IT IS SPURIOUS
  290 CONTINUE
      MPEV = 1
      IF (JU.LT.JL) MPEV=0
      KNEW = K-1
C
C     X1 >= XU+EP0
C     SPURIOUS TEST AND T-SIMPLE CASE COMPLETED
C     START OF NEXT T-EIGENVALUE COMPUTATION
C
  300 K = KNEW
      MP(NDIS) = MPEV
      IF (MPEV.GE.1) NG = NG + 1
C
      IF (IWRITE.NE.0) WRITE(6,310) VS(NDIS),MPEV,MD,NG
  310 FORMAT(E25.16,3I6)
C
C     UPDATE STURM COUNT. IC0 = STURM COUNT FOR KTH T-EIGENVALUE
      ICT = ICT + IC0
C
C     EXIT TEST FOR K DO LOOP
C
      IF (K.LE.0) GO TO 410
C
C     UPDATE LOWER BOUNDS
      DO 320 I=1,KNEW
  320 VB(I) = DMAX1(X1,VB(I))
C
      GO TO 150
C     END OF BISECTION LOOP FOR KTH T-EIGENVALUE
C
C     FORWARD STURM CALCULATION
  330 NEV = -NA
      YU  = ONE
C
      DO 360 I = 1,MEV
      IF (YU.NE.ZERO) GO TO 340
      YV = BETA(I)/EPS
      GO TO 350
  340 YV = BETA2(I)/YU
  350 YU = X1 - ALPHA(I) - YV
      IF (YU.GE.ZERO) GO TO 360
      NEV = NEV + 1
  360 CONTINUE
C     NEV = NUMBER OF T-EIGENVALUES ON (X1,UB)
C
      GO TO (50,60,160,180,270), ISTURM
C
C     BACKWARD STURM CALCULATION FOR T(1,MEV) AND T(2,MEV)
  370 KEV = -NA
      YU  = ONE
C
```

```
      DO 400 II = 1,MEV                                                 LES03740
      I = MP1-II                                                        LES03750
      IF (YU.NE.ZERO) GO TO 380                                         LES03760
      YV = BETA(I+1)/EPS                                                LES03770
      GO TO 390                                                         LES03780
  380 YV = BETA2(I+1)/YU                                                LES03790
  390 YU = X1-ALPHA(I)-YV                                               LES03800
      JEV = 0                                                           LES03810
      IF (YU.GE.ZERO) GO TO 400                                         LES03820
      KEV = KEV+1                                                       LES03830
      JEV = 1                                                           LES03840
  400 CONTINUE                                                          LES03850
      JEV = KEV-JEV                                                     LES03860
C                                                                       LES03870
      GO TO (240,250), JSTURM                                           LES03880
C                                                                       LES03890
C     KEV = -NA + (NUMBER OF T(1,MEV) EIGENVALUES) > X1                 LES03900
C     JEV = -NA + (NUMBER OF T(2,MEV) EIGENVALUES) > X1                 LES03910
C     SET PARAMETERS FOR NEXT INTERVAL                                  LES03920
  410 CONTINUE                                                          LES03930
      IC = ICT+IC                                                       LES03940
      MXSTUR = MXSTUR-ICT                                               LES03950
C                                                                       LES03960
      WRITE(6,420) JIND,NG,MD                                           LES03970
  420 FORMAT(/' T-EIGENVALUE CALCULATION ON INTERVAL',I6,'  IS COMPLETE'LES03980
     1 /3X,'NO. GOOD',3X,'NO. DISTINCT'/I10,I13)                        LES03990
C                                                                       LES04000
  430 CONTINUE                                                          LES04010
C                                                                       LES04020
C     END LOOP ON THE SUBINTERVALS (LB(J),UB(J)), J=1,NINT              LES04030
C     ISKIP OUTPUT                                                      LES04040
C                                                                       LES04050
      IF (ISKIP.GT.0) WRITE(6,440)ISKIP                                 LES04060
  440 FORMAT(' BISEC DEFAULTED ON',I3,3X,'INTERVALS'/                   LES04070
     1 ' DEFAULTS OCCUR IF AN INTERVAL HAS NO T-EIGENVALUES'/           LES04080
     2 ' OR THE STURM ESTIMATE EXCEEDS THE USER-SPECIFIED LIMIT'/)      LES04090
C                                                                       LES04100
      IF (ISKIP.GT.0) WRITE(6,450)(IDEF(I), I=1,ISKIP)                  LES04110
  450 FORMAT(' BISEC DEFAULTED ON INTERVALS'/(10I8))                    LES04120
C                                                                       LES04130
C     RESET BETA AT I = MP1                                             LES04140
      BETA(MP1) = BETAM                                                 LES04150
C-----END OF BISEC------------------------------------------------------LES04160
      RETURN                                                            LES04170
      END                                                               LES04180
C                                                                       LES04190
C-----INVERSE ITERATION ON T(1,MEV)-------------------------------------LES04200
C                                                                       LES04210
      SUBROUTINE INVERR(ALPHA,BETA,V1,V2,VS,EPS,G,MP,MEV,MMB,NDIS,NISO, LES04220
     1 N,IKL,IT,IWRITE)                                                 LES04230
C                                                                       LES04240
C-----------------------------------------------------------------------LES04250
      DOUBLE PRECISION ALPHA(1),BETA(*),V1(1),V2(*),VS(*)               LES04260
      DOUBLE PRECISION X1,U,Z,EST,TEMP,T0,T1,RATIO,SUM,XU,NORM,TSUM     LES04270
      DOUBLE PRECISION  BETAM,EPS,EPS3,EPS4,ZERO,ONE                    LES04280
```

```
      REAL G(1)                                                         LES04290
      INTEGER  MP(1)                                                    LES04300
      DOUBLE PRECISION FINPRO                                           LES04310
      REAL  ABS                                                         LES04320
      DOUBLE PRECISION  DABS, DMIN1, DSQRT, DFLOAT                      LES04330
C-----------------------------------------------------------------------LES04340
C     COMPUTES ERROR ESTIMATES FOR COMPUTED ISOLATED GOOD T-EIGENVALUES LES04350
C     IN VS AND WRITES THESE T-EIGENVALUES AND ESTIMATES TO FILE 4.     LES04360
C     BY DEFINITION A GOOD T-EIGENVALUE IS ISOLATED IF ITS              LES04370
C     CLOSEST T-NEIGHBOR IS ALSO GOOD, OR ITS CLOSEST NEIGHBOR IS       LES04380
C     SPURIOUS, BUT THAT NEIGHBOR IS FAR ENOUGH AWAY.  SO               LES04390
C     IN PARTICULAR, WE COMPUTE ESTIMATES FOR GOOD T-EIGENVALUES        LES04400
C     THAT ARE IN CLUSTERS OF GOOD T-EIGENVALUES.                       LES04410
C                                                                       LES04420
C     USES INVERSE ITERATION ON T(1,MEV) SOLVING THE EQUATION           LES04430
C     (T - X1*I)V2 = RIGHT-HAND SIDE (RANDOMLY-GENERATED)               LES04440
C     FOR EACH SUCH GOOD T-EIGENVALUE X1.                               LES04450
C                                                                       LES04460
C     PROGRAM REFACTORS T-X1*I ON EACH ITERATION OF INVERSE ITERATION.  LES04470
C     TYPICALLY ONLY ONE ITERATION IS NEEDED PER EIGENVALUE X1.         LES04480
C                                                                       LES04490
C     POSSIBLE STORAGE COMPRESSION                                      LES04500
C     G STORAGE COULD BE ELIMINATED BY REGENERATING THE RANDOM          LES04510
C     RIGHT-HAND SIDE ON EACH ITERATION AND PRINTING OUT THE            LES04520
C     ERROR ESTIMATES AS THEY ARE GENERATED.                            LES04530
C                                                                       LES04540
C     ON ENTRY AND EXIT                                                 LES04550
C     MEV = ORDER OF T                                                  LES04560
C     ALPHA, BETA CONTAIN THE NONZERO ENTRIES OF THE T-MATRIX           LES04570
C     VS = COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)                    LES04580
C     MP = T-MULTIPLICITY OF EACH T-EIGENVALUE IN VS. MP(I) = -1 MEANS  LES04590
C          VS(I) IS A GOOD T-EIGENVALUE BUT THAT IT IS SITTING CLOSE TO LES04600
C          A SPURIOUS T-EIGENVALUE.  MP(I) = 0 MEANS VS(I) IS SPURIOUS. LES04610
C          ESTIMATES ARE COMPUTED ONLY FOR THOSE T-EIGENVALUES          LES04620
C          WITH MP(I) = 1. FLAGGING WAS DONE IN SUBROUTINE ISOEV        LES04630
C          PRIOR TO ENTERING INVERR.                                    LES04640
C     NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES CONTAINED IN VS      LES04650
C     NDIS =  NUMBER OF DISTINCT T-EIGENVALUES IN VS                    LES04660
C     IKL = SEED FOR RANDOM NUMBER GENERATOR                            LES04670
C     EPS = 2. * MACHINE EPSILON                                        LES04680
C                                                                       LES04690
C     IN PROGRAM:                                                       LES04700
C     ITER = MAXIMUM NUMBER OF INVERSE ITERATION STEPS ALLOWED FOR EACH LES04710
C            X1.  ITER = IT ON ENTRY.                                   LES04720
C     G = ARRAY OF DIMENSION AT LEAST MEV + NISO.  USED TO STORE        LES04730
C         RANDOMLY-GENERATED RIGHT-HAND SIDE.  THIS IS NOT              LES04740
C         REGENERATED FOR EACH X1. G IS ALSO USED TO STORE ERROR        LES04750
C         ESTIMATES AS THEY ARE COMPUTED FOR LATER PRINTOUT.            LES04760
C     V1,V2 = WORK SPACES USED IN THE FACTORIZATION OF T(1,MEV).        LES04770
C     AT THE END OF THE INVERSE ITERATION COMPUTATION FOR X1, V2        LES04780
C     CONTAINS THE UNIT EIGENVECTOR OF T(1,MEV) CORRESPONDING TO X1.    LES04790
C     V1 AND V2 MUST BE OF DIMENSION AT LEAST MEV.                      LES04800
C                                                                       LES04810
C     ON EXIT                                                           LES04820
C     G(J) = MINIMUM GAP IN T(1,MEV) FOR EACH VS(J), J=1,NDIS           LES04830
```

```
C     G(MEV+I) = BETAM*|V2(MEV)| = ERROR ESTIMATE FOR ISOLATED GOOD
C              T-EIGENVALUES, WHERE I = 1,NISO  AND  BETAM = BETA(MEV+1)
C              V2(MEV) IS LAST COMPONENT OF THE UNIT EIGENVECTOR OF
C              T(1,MEV) CORRESPONDING TO ITH ISOLATED GOOD T-EIGENVALUE.
C
C     IF FOR SOME X1 IT.GT.ITER THEN THE ERROR ESTIMATE IN G IS MARKED
C     WITH A - SIGN.
C
C     V2 = ISOLATED GOOD T-EIGENVALUES
C     V1 = MINIMAL T-GAPS FOR THE T-EIGENVALUES IN V2.
C     THESE ARE CONSTRUCTED FOR WRITE-OUT PURPOSES ONLY AND NOT
C     NEEDED ELSEWHERE IN THE PROGRAM.
C-----------------------------------------------------------------------
C
C     LABEL OUTPUT FILE  4
      IF (MMB.EQ.1) WRITE(4,10)
   10 FORMAT(' INVERSE ITERATION ERROR ESTIMATES'/)
C
C     FILE 6 (TERMINAL) OUTPUT OF ERROR ESTIMATES
      IF (IWRITE.NE.0.AND.NISO.NE.0) WRITE(6,20)
   20 FORMAT(/' INVERSE ITERATION ERROR ESTIMATES'/'  JISO',' JDIST',8X
     1,'GOOD T-EIGENVALUE',4X,'BETAM*UM',5X,'TMINGAP')
C
C     INITIALIZATION AND PARAMETER SPECIFICATION
      ZERO = 0.0D0
      ONE = 1.0D0
      NG = 0
      NISO = 0
      ITER = IT
      MP1 = MEV+1
      MM1 = MEV-1
      BETAM = BETA(MP1)
      BETA(MP1) = ZERO
C
C     CALCULATE SCALE AND TOLERANCES
      TSUM = DABS(ALPHA(1))
      DO 30 I = 2,MEV
   30 TSUM = TSUM + DABS(ALPHA(I)) + BETA(I)
C
      EPS3 = EPS*TSUM
      EPS4 = DFLOAT(MEV)*EPS3
C
C     GENERATE SCALED RANDOM RIGHT-HAND SIDE
      ILL = IKL
C
C-----------------------------------------------------------------------
      CALL GENRAN(ILL,G,MEV)
C-----------------------------------------------------------------------
C
      GSUM = ZERO
      DO 40 I = 1,MEV
   40 GSUM = GSUM+ABS(G(I))
      GSUM = EPS4/GSUM
C
      DO 50 I = 1,MEV
```

```
   50 G(I) = GSUM*G(I)
C
C     LOOP ON ISOLATED GOOD T-EIGENVALUES IN VS (MP(I) = 1) TO
C     CALCULATE CORRESPONDING UNIT EIGENVECTOR OF T(1,MEV)
C
      DO 180 JEV = 1,NDIS
C
      IF (MP(JEV).EQ.0) GO TO 180
      NG = NG + 1
      IF (MP(JEV).NE.1) GO TO 180
C
      IT = 1
      NISO = NISO + 1
      X1 = VS(JEV)
C
C     INITIALIZE RIGHT HAND SIDE FOR INVERSE ITERATION
      DO 60 I = 1,MEV
   60 V2(I) = G(I)
C
C     TRIANGULAR FACTORIZATION WITH NEAREST NEIGHBOR PIVOT
C     STRATEGY. INTERCHANGES ARE LABELLED BY SETTING BETA < 0.
C
   70 CONTINUE
      U = ALPHA(1)-X1
      Z = BETA(2)
C
      DO 90 I = 2,MEV
      IF (BETA(I).GT.DABS(U)) GO TO 80
C     NO INTERCHANGE
      V1(I-1) = Z/U
      V2(I-1) = V2(I-1)/U
      V2(I) = V2(I)-BETA(I)*V2(I-1)
      RATIO = BETA(I)/U
      U = ALPHA(I)-X1-Z*RATIO
      Z = BETA(I+1)
      GO TO 90
   80 CONTINUE
C     INTERCHANGE CASE
      RATIO = U/BETA(I)
      BETA(I) = -BETA(I)
      V1(I-1) = ALPHA(I)-X1
      U = Z-RATIO*V1(I-1)
      Z = -RATIO*BETA(I+1)
      TEMP = V2(I-1)
      V2(I-1) = V2(I)
      V2(I) = TEMP-RATIO*V2(I)
   90 CONTINUE
      IF (U.EQ.ZERO) U = EPS3
C
C     SMALLNESS TEST AND DEFAULT VALUE FOR LAST COMPONENT
C     PIVOT(I-1) = |BETA(I)| FOR INTERCHANGE CASE
C     (I-1,I+1) ELEMENT IN RIGHT FACTOR = BETA(I+1)
C     END OF FACTORIZATION AND FORWARD SUBSTITUTION
C
C     BACK SUBSTITUTION
```

```
      V2(MEV) = V2(MEV)/U
      DO 110 II = 1,MM1
      I = MEV-II
      IF (BETA(I+1).LT.ZERO) GO TO 100
C     NO INTERCHANGE
      V2(I) = V2(I)-V1(I)*V2(I+1)
      GO TO 110
C     INTERCHANGE CASE
  100 BETA(I+1) = -BETA(I+1)
      V2(I) = (V2(I)-V1(I)*V2(I+1)-BETA(I+2)*V2(I+2))/BETA(I+1)
  110 CONTINUE
C
C     TESTS FOR CONVERGENCE OF INVERSE ITERATION
C     IF SUM |V2| COMPS. LE. 1 AND IT. LE. ITER DO ANOTHER INVIT STEP
C
      NORM = DABS(V2(MEV))
      DO 120 II = 1,MM1
      I = MEV-II
  120 NORM = NORM+DABS(V2(I))
C
      IF (NORM.GE.ONE) GO TO 140
      IT = IT+1
      IF (IT.GT.ITER) GO TO 140
      XU = EPS4/NORM
C
      DO 130 I = 1,MEV
  130 V2(I) = V2(I)*XU
C
      GO TO 70
C     ANOTHER INVERSE ITERATION STEP
C
C     INVERSE ITERATION FINISHED
C     NORMALIZE COMPUTED T-EIGENVECTOR : V2 = V2/||V2||
  140 CONTINUE
      SUM = FINPRO(MEV,V2(1),1,V2(1),1)
      SUM = ONE/DSQRT(SUM)
C
      DO 150 II = 1,MEV
  150 V2(II) = SUM*V2(II)
C
C     SAVE ERROR ESTIMATE FOR LATER OUTPUT
      EST = BETAM*DABS(V2(MEV))
      IF (IT.GT.ITER) EST = -EST
      MEVPNI = MEV + NISO
      G(MEVPNI) = EST
      IF (IWRITE.EQ.0) GO TO 180
C
C     FILE 6 (TERMINAL) OUTPUT OF ERROR ESTIMATES.
      IF (JEV.EQ.1) GAP = VS(2) - VS(1)
      IF (JEV.EQ.MEV) GAP = VS(MEV) - VS(MEV-1)
      IF (JEV.EQ.MEV.OR.JEV.EQ.1) GO TO 160
      TEMP = DMIN1(VS(JEV+1)-VS(JEV),VS(JEV)-VS(JEV-1))
      GAP = TEMP
  160 CONTINUE
C
```

```
      WRITE(6,170) NISO,JEV,X1,EST,GAP
  170 FORMAT(2I6,E25.16,2E12.3)
C
  180 CONTINUE
C
C     END ERROR ESTIMATE LOOP ON ISOLATED GOOD T-EIGENVALUES.
C     GENERATE DISTINCT MINGAPS FOR T(1,MEV).  THIS IS USEFUL AS AN
C     INDICATOR OF THE GOODNESS OF THE INVERSE ITERATION ESTIMATES.
C     TRANSFER ISOLATED GOOD T-EIGENVALUES AND CORRESPONDING TMINGAPS
C     TO V2 AND V1 FOR OUTPUT PURPOSES ONLY.
C
      NM1 = NDIS - 1
      G(NDIS) = VS(NM1)-VS(NDIS)
      G(1) = VS(2)-VS(1)
C
      DO 190 J = 2,NM1
      T0 = VS(J)-VS(J-1)
      T1 = VS(J+1)-VS(J)
      G(J) = T1
      IF (T0.LT.T1) G(J)=-T0
  190 CONTINUE
      ISO = 0
      DO 200 J = 1,NDIS
      IF (MP(J).NE.1) GO TO 200
      ISO = ISO+1
      V1(ISO) = G(J)
      V2(ISO) = VS(J)
  200 CONTINUE
C
      IF(NISO.EQ.0) GO TO 250
C
C     ERROR ESTIMATES ARE WRITTEN TO FILE 4
      WRITE(4,210)MEV,NDIS,NG,NISO,N,IKL,ITER,BETAM
  210 FORMAT(1X,'TSIZE',2X,'NDIS',1X,'NGOOD',2X,'NISO',1X,'ASIZE'/5I6/
     1 4X,'RHSEED',2X,'MXINIT',5X,'BETAM'/I10,I8,E10.3/
     2 2X,'GOODEVNO',8X,'GOOD T-EIGENVALUE',6X,'BETAM*UM',7X,'TMINGAP')
C
      ISPUR = 0
      I = 0
      DO 240 J = 1,NDIS
      IF(MP(J).NE.0) GO TO 220
      ISPUR = ISPUR + 1
      GO TO 240
  220 IF(MP(J).NE.1) GO TO 240
      I = I + 1
      MEVI = MEV + I
      IGOOD = J - ISPUR
      WRITE(4,230) IGOOD,V2(I),G(MEVI),V1(I)
  230 FORMAT(I10,E25.16,2E14.3)
  240 CONTINUE
      GO TO 270
C
  250 WRITE(4,260)
  260 FORMAT(/' THERE ARE NO ISOLATED T-EIGENVALUES SO NO ERROR ESTIMATE
     1S WERE COMPUTED')
```

```
C
C     RESTORE BETA(MEV+1) = BETAM
  270 BETA(MP1) = BETAM
C-----END OF INVERR-----------------------------------------------------
      RETURN
      END
C
C-----START OF TNORM----------------------------------------------------
C
      SUBROUTINE TNORM(ALPHA,BETA,BMIN,TMAX,MEV,IB)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  ALPHA(1),BETA(*)
      DOUBLE PRECISION  TMAX,BMIN,BMAX,BSIZE,BTOL
      DOUBLE PRECISION  DABS, DMAX1
C-----------------------------------------------------------------------
C     COMPUTE SCALING FACTOR USED IN THE T-MULTIPLICITY, SPURIOUS AND
C     PRTESTS.  CHECK RELATIVE SIZE OF THE BETA(K), K=1,MEV
C     AS A TEST ON THE LOCAL ORTHOGONALITY OF THE LANCZOS VECTORS.
C
C             TMAX = MAX (|ALPHA(I)|, BETA(I),  I=1,MEV)
C             BMIN = MIN (BETA(I) I=2,MEV)
C             BSIZE = BMIN/TMAX
C             |IB| = INDEX OF MINIMAL(BETA)
C             IB < 0 IF BMIN/TMAX < BTOL
C-----------------------------------------------------------------------
C     SPECIFY PARAMETERS
      IB = 2
      BTOL = BMIN
      BMIN = BETA(2)
      BMAX = BETA(2)
      TMAX = DABS(ALPHA(1))
C
      DO 20 I = 2,MEV
      IF (BETA(I).GE.BMIN) GO TO 10
      IB = I
      BMIN = BETA(I)
   10 TMAX = DMAX1(TMAX,DABS(ALPHA(I)))
      BMAX = DMAX1(BETA(I),BMAX)
   20 CONTINUE
      TMAX = DMAX1(BMAX,TMAX)
C
C     TEST OF LOCAL ORTHOGONALITY USING SCALED BETAS
      BSIZE = BMIN/TMAX
      IF (BSIZE.GE.BTOL) GO TO 40
C
C     DEFAULT.  BSIZE IS SMALLER THAN TOLERANCE BTOL SPECIFIED IN MAIN
C     PROGRAM.  PROGRAM TERMINATES FOR USER TO DECIDE WHAT TO DO
C     BECAUSE LOCAL ORTHOGONALITY OF THE LANCZOS VECTORS COULD BE
C     LOST.
C
      IB = -IB
      WRITE(6,30) MEV
   30 FORMAT(/' BETA TEST INDICATES POSSIBLE LOSS OF LOCAL ORTHOGONALITY
     1OVER 1ST',I6,' LANCZOS VECTORS'/)
```

```
C
   40 CONTINUE
C
      WRITE(6,50) IB
   50 FORMAT(/' MINIMUM BETA RATIO OCCURS AT',I6,' TH BETA'/)
C
      WRITE(6,60) MEV,BMIN,TMAX,BSIZE
   60 FORMAT(/1X,'TSIZE',6X,'MIN BETA',5X,'TKMAX',6X,'MIN RATIO'/
     1 I6,E14.3,E10.3,E15.3/)
C
C-----END OF TNORM-------------------------------------------------------
      RETURN
      END
C
C
C-----START OF LUMP------------------------------------------------------
C
      SUBROUTINE LUMP(V1,RELTOL,MULTOL,SCALE2,LINDEX,LOOP)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  V1(1),SUM,RELTOL,MULTOL,THOLD,ZERO,SCALE2
      INTEGER  LINDEX(1)
      DOUBLE PRECISION  DABS, DFLOAT, DMAX1
C-----------------------------------------------------------------------
C     LINDEX(J) = T-MULTIPLICITY OF JTH DISTINCT T-EIGENVALUE
C     LOOP = NUMBER OF DISTINCT T-EIGENVALUES
C     LUMP 'COMBINES' COMPUTED 'GOOD' T-EIGENVALUES THAT ARE
C     'TOO CLOSE'.
C     VALUE OF RELTOL IS 1.D-10.
C
C     IF IN A SET OF T-EIGENVALUES TO BE COMBINED THERE IS AN EIGENVALUE
C     WITH LINDEX=1, THEN THE VALUE OF THE COMBINED EIGENVALUES IS SET
C     EQUAL TO THE VALUE OF THAT EIGENVALUE.  NOTE THAT IF A SPURIOUS
C     T-EIGENVALUE IS TO BE 'COMBINED' WITH A GOOD T-EIGENVALUE, THEN
C     THIS IS DONE ONLY BY INCREASING THE INDEX, LINDEX, FOR THAT
C     T-EIGENVALUE.  NUMERICAL VALUES OF SPURIOUS EIGENVALUES ARE NEVER
C     COMBINE WITH THOSE OF GOOD T-EIGENVALUES.
C-----------------------------------------------------------------------
      ZERO = 0.0D0
      NLOOP = 0
      J = 0
      ICOUNT = 1
      JI = 1
      THOLD = DMAX1(RELTOL*DABS(V1(1)),SCALE2*MULTOL)
C     THOLD = DMAX1(RELTOL*DABS(V1(1)),RELTOL)
C
   10 J = J+1
      IF (J.EQ.LOOP) GO TO 20
      SUM = DABS(V1(J)-V1(J+1))
      IF (SUM.LT.THOLD) GO TO 60
   20 JF = JI + ICOUNT - 1
      INDSUM = 0
      ISPUR = 0
C
      DO 30 KK = JI,JF
```

```
      IF (LINDEX(KK).NE.0) GO TO 30
      ISPUR = ISPUR + 1
      INDSUM = INDSUM + 1
   30 INDSUM = INDSUM + LINDEX(KK)
C
C     IF (JF-JI.GE.1) WRITE(6,40) (V1(KKK), KKK=JI,JF)
   40 FORMAT(/' LUMP LUMPS THE T-EIGENVALUES'/(4E20.13))
C
C     COMPUTE THE 'COMBINED' T-EIGENVALUE AND THE RESULTING
C     T-MULTIPLICITY
      K = JI - 1
   50 K = K+1
      IF (K.GT.JF) GO TO 70
      IF (LINDEX(K) .NE.1) GO TO 50
      NLOOP = NLOOP + 1
      V1(NLOOP) = V1(K)
      GO TO 100
   60 ICOUNT = ICOUNT + 1
      GO TO 10
C
C     ALL INDICES WERE 0 OR >1
   70 NLOOP = NLOOP + 1
      IDIF = INDSUM - ISPUR
      IF (IDIF.EQ.0) GO TO 90
C
      SUM = ZERO
      DO 80 KK = JI,JF
   80 SUM = SUM + V1(KK) * DFLOAT(LINDEX(KK))
C
      V1(NLOOP) = SUM/DFLOAT(IDIF)
      GO TO 100
   90 V1(NLOOP) = V1(JI)
  100 LINDEX(NLOOP) = INDSUM
      IDIF = INDSUM - ISPUR
      IF (IDIF.EQ.0.AND.ISPUR.EQ.1) LINDEX(NLOOP) = 0
      IF (J.EQ.LOOP) GO TO 110
      ICOUNT = 1
      JI= J+1
      THOLD = DMAX1(RELTOL*DABS(V1(JI)),SCALE2*MULTOL)
C     THOLD = DMAX1(RELTOL*DABS(V1(JI)),RELTOL)
      IF (JI.LT.LOOP) GO TO 10
      NLOOP = NLOOP + 1
      V1(NLOOP)= V1(JI)
      LINDEX(NLOOP) = LINDEX(JI)
  110 CONTINUE
C
C     ON RETURN V1 CONTAINS THE DISTINCT T-EIGENVALUES
C     LINDEX CONTAINS THE CORRESPONDING T-MULTIPLICITIES
C
      LOOP = NLOOP
      RETURN
C-----END OF LUMP-------------------------------------------------------
      END
C
C
```

```
C-----START OF ISOEV-----------------------------------------------------
C
      SUBROUTINE ISOEV(VS,GAPTOL,MULTOL,SCALE1,G,MP,NDIS,NG,NISO)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  VS(*),T0,T1,MULTOL,GAPTOL,SCALE1,TEMP
      REAL   G(1),GAP
      INTEGER  MP(1)
      REAL  ABS
      DOUBLE PRECISION  DABS, DMAX1
C-----------------------------------------------------------------------
C     GENERATE DISTINCT TMINGAPS AND USE THEM TO LABEL THE ISOLATED
C     GOOD T-EIGENVALUES THAT ARE VERY CLOSE TO SPURIOUS ONES.
C     ERROR ESTIMATES WILL NOT BE COMPUTED FOR THESE T-EIGENVALUES.
C
C     ON ENTRY AND EXIT
C     VS CONTAINS THE COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)
C     MP CONTAINS THE CORRESPONDING T-MULTIPLICITIES
C     NDIS = NUMBER OF DISTINCT EIGENVALUES
C     GAPTOL = RELATIVE GAP TOLERANCE SET IN MAIN
C
C     ON EXIT
C     G CONTAINS THE TMINGAPS.
C     G(I) < 0 MEANS MINGAP IS DUE TO LEFT GAP
C     MP(I) IS NOT CHANGED EXCEPT THAT  MP(I)=-1, IF MP(I)=1,
C     TMINGAP WAS TOO SMALL AND DUE TO A SPURIOUS T-EIGENVALUE.
C
C     IF MP(I)=-1 THAT SIMPLE GOOD T-EIGENVALUE WILL BE SKIPPED
C     IN THE SUBSEQUENT ERROR ESTIMATE COMPUTATIONS IN INVERR
C     THAT IS, WE COMPUTE ERROR ESTIMATES ONLY FOR THOSE GOOD
C     T-EIGENVALUES WITH MP(I)=1.
C-----------------------------------------------------------------------
C     CALCULATE MINGAPS FOR DISTINCT T(1,MEV) EIGENVALUES.
      NM1 = NDIS - 1
      G(NDIS) = VS(NM1)-VS(NDIS)
      G(1) = VS(2)-VS(1)
C
      DO 10 J = 2,NM1
      T0 = VS(J)-VS(J-1)
      T1 = VS(J+1)-VS(J)
      G(J) = T1
      IF (T0.LT.T1) G(J) = -T0
   10 CONTINUE
C
C     SET MP(I)=-1 FOR SIMPLE GOOD T-EIGENVALUES WHOSE MINGAPS  ARE
C     'TOO SMALL' AND DUE TO SPURIOUS T-EIGENVALUES.
C
      NISO = 0
      NG = 0
      DO 20 J = 1,NDIS
      IF (MP(J).EQ.0) GO TO 20
      NG = NG+1
      IF (MP(J).NE.1) GO TO 20
C     VS(J) IS NEXT SIMPLE GOOD T-EIGENVALUE
      NISO = NISO + 1
```

```
      I = J+1
      IF (G(J).LT.0.0) I = J-1
      IF (MP(I).NE.0) GO TO 20
      GAP = ABS(G(J))
      T0 = DMAX1(SCALE1*MULTOL,GAPTOL*DABS(VS(J)))
C     T0 = DMAX1(GAPTOL,GAPTOL*DABS(VS(J)))
      TEMP = T0
      IF (GAP.GT.TEMP) GO TO 20
      MP(J) = -MP(J)
      NISO = NISO-1
   20 CONTINUE
C
C-----END OF ISOEV----------------------------------------------------
      RETURN
      END
C
C-----START OF PRTEST-------------------------------------------------
C
      SUBROUTINE PRTEST(ALPHA,BETA,TEIG,TKMAX,EPSM,RELTOL,SCALE3,SCALE4,
     1 TMULT,NDIST,MEV,IPROJ)
C
C--------------------------------------------------------------------
      DOUBLE PRECISION  ALPHA(1), BETA(1),TEIG(*),SIGMA(10)
      DOUBLE PRECISION  EPSM,RELTOL,PRTOL,TKMAX,LRATIO,URATIO
      DOUBLE PRECISION  EPS,EPS1,BETAM,LBD,UBD,SIG,YU,YV,LRATS,URATS
      DOUBLE PRECISION ZERO,ONE,TEN,BISTOL,SCALE3,SCALE4,AEV,TEMP
      INTEGER  TMULT(*),ISIGMA(10)
      DOUBLE PRECISION  DABS, DMAX1, DSQRT, DFLOAT
C--------------------------------------------------------------------
C     AFTER CONVERGENCE HAS BEEN ESTABLISHED, SUBROUTINE PRTEST
C     TESTS COMPUTED EIGENVALUES OF T(1,MEV) THAT HAVE BEEN LABELLED
C     SPURIOUS TO DETERMINE IF ANY EIGENVALUES OF A HAVE BEEN
C     MISSED BY LANCZOS PROCEDURE.  AN EIGENVALUE WITH A VERY SMALL
C     PROJECTION ON THE STARTING VECTOR (< SINGLE PRECISION)
C     CAN BE MISSED BECAUSE IT IS ALSO AN EIGENVALUE OF T(2,MEV) TO
C     WITHIN THE SQUARE OF THIS ORIGINAL PROJECTION.
C     OUR EXPERIENCE IS THAT SUCH SMALL PROJECTIONS OCCUR ONLY
C     VERY INFREQUENTLY.
C
C     THIS SUBROUTINE IS CALLED ONLY AFTER CONVERGENCE HAS BEEN
C     ESTABLISHED. ONCE CONVERGENCE HAS BEEN OBSERVED ON THE
C     OTHER EIGENVALUES THEN ONE CAN EXPECT TO ALSO HAVE CONVERGENCE
C     ON ANY SUCH HIDDEN EIGENVALUES.(IF THERE ARE ANY).  THIS
C     PROCEDURE CONSIDERS ONLY SPURIOUS T-EIGENVALUES AND ONLY THOSE
C     SPURIOUS T-EIGENVALUES THAT ARE ISOLATED FROM GOOD T-EIGENVALUES.
C     FOR EACH SUCH T-EIGENVALUE IT DOES 2 STURM SEQUENCES
C     AND A FEW SCALAR MULTIPLICATIONS.  UPON RETURN TO MAIN
C     PROGRAM ERROR ESTIMATES WILL BE COMPUTED FOR ANY EIGENVALUES
C     THAT HAVE BEEN LABELLED AS 'HIDDEN'.  SUCH T-EIGENVALUES
C     WILL BE RELABELLED AS 'GOOD' ONLY IF THESE ERROR ESTIMATES
C     ARE SUFFICIENTLY SMALL.
C--------------------------------------------------------------------
      ZERO = 0.0D0
      ONE  = 1.0D0
      TEN  = 10.0D0
```

```
      PRTOL = 1.D-6
      TEMP = DFLOAT(MEV+1000)
      TEMP = DSQRT(TEMP)
      BISTOL = TKMAX*EPSM*TEMP
      NSIGMA = 4
      SIGMA(1) = TEN*TKMAX
C
      DO 10 J = 2,NSIGMA
   10 SIGMA(J) = TEN*SIGMA(J-1)
C
      IFIN = 0
      MF = 1
      ML = MEV
      BETAM = BETA(MF)
      BETA(MF) = ZERO
      IPROJ = 0
      J = 1
C
      IF (TMULT(1).NE.0) GO TO 110
C
      AEV = DABS(TEIG(1))
      TEMP = PRTOL*AEV
      EPS1 = DMAX1(TEMP,SCALE4*BISTOL)
C     EPS1 = DMAX1(TEMP,PRTOL)
      TEMP = RELTOL*AEV
      EPS  = DMAX1(TEMP,SCALE3*BISTOL)
C     EPS  = DMAX1(TEMP,RELTOL)
C
      IF (TEIG(2)-TEIG(1).LT.EPS1.AND.TMULT(2).NE.0) GO TO 110
C
   20 LBD = TEIG(J) - EPS
      UBD = TEIG(J) + EPS
      MEVL = 0
      IL = 0
      YU = ONE
C
      DO 50 I=MF,ML
      IF (YU.NE.ZERO) GO TO 30
      YV = BETA(I)/EPSM
      GO TO 40
   30 YV = BETA(I)*BETA(I)/YU
   40 YU = ALPHA(I)-LBD-YV
      IF (YU.GE.ZERO) GO TO 50
C     MEVL INCREMENTED
      MEVL = MEVL + 1
      IL = I
   50 CONTINUE
C
      LRATIO = YU
      MEV1L = MEVL
      IF (IL.EQ.ML) MEV1L=MEVL-1
C
C     MEVL = NUMBER OF EVS OF T(1,MEV) WHICH ARE < LBD
C     MEV1L = NUMBER OF EVS OF T(1,MEV-1) WHICH ARE < LBD
C     LRATIO = DET(T(1,MEV)-LBD)/DET(T(1,MEV-1)-LBD):
```

```
C
      MEVU = 0
      IL = 0
      YU = ONE
C
      DO 80 I=MF,ML
      IF (YU.NE.ZERO) GO TO 60
      YV = BETA(I)/EPSM
      GO TO 70
   60 YV = BETA(I)*BETA(I)/YU
   70 YU = ALPHA(I)-UBD-YV
      IF (YU.GE.ZERO) GO TO 80
C     MEVU INCREMENTED
      MEVU = MEVU + 1
      IL = I
   80 CONTINUE
C
      URATIO = YU
      MEV1U = MEVU
      IF (IL.EQ.ML) MEV1U=MEVU-1
C
C     MEVU = NUMBER OF EVS OF T(MEV) WHICH ARE < UBD
C     MEV1U = NUMBER OF EVS OF T(MEV-1) WHICH ARE < UBD
C     URATIO = DET(TM-UBD)/DET(T(M-1)-UBD): TM=T(MF,ML)
C
      NEV1 = MEV1U-MEV1L
C
      DO 90 K=1,NSIGMA
      SIG = SIGMA(K)
      LRATS = LRATIO-SIG
      URATS = URATIO-SIG
C     NOTE THE INCREMENT IS ON NUMBER OF EVALUES OF T(M-1)
      MEVLS = MEV1L
      IF (LRATS.LT.0.) MEVLS=MEV1L+1
      MEVUS = MEV1U
      IF (URATS.LT.0.) MEVUS=MEV1U+1
      ISIGMA(K) = MEVUS - MEVLS
   90 CONTINUE
C
      ICOUNT = 0
      DO 100 K=1,NSIGMA
  100 IF (ISIGMA(K).EQ.1) ICOUNT=ICOUNT + 1
C
      IF (ICOUNT.LT.2.OR.NEV1.EQ.0) GO TO 110
      TMULT(J) = -10
      IPROJ=IPROJ+1
C
  110 J=J+1
C
      IF (J.GE.NDIST) GO TO 120
      IF (TMULT(J).NE.0) GO TO 110
C
      AEV = DABS(TEIG(J))
      TEMP = PRTOL*AEV
      EPS1 = DMAX1(TEMP,SCALE4*BISTOL)
```

```
C     EPS1 = DMAX1(TEMP,PRTOL)
      TEMP = RELTOL*AEV
      EPS  = DMAX1(TEMP,SCALE3*BISTOL)
C     EPS  = DMAX1(TEMP,RELTOL)
C
      IF (TEIG(J)-TEIG(J-1).LT.EPS1.AND.TMULT(J-1).NE.0) GO TO 110
      IF (TEIG(J+1)-TEIG(J).LT.EPS1.AND.TMULT(J+1).NE.0) GO TO 110
C
      GO TO 20
C
  120 IF (IFIN.EQ.1) GO TO 130
      IF (TMULT(NDIST).NE.0) GO TO 130
C
      AEV = DABS(TEIG(NDIST))
      TEMP = PRTOL*AEV
      EPS1 = DMAX1(TEMP,SCALE4*BISTOL)
C     EPS1 = DMAX1(TEMP,PRTOL)
      TEMP = RELTOL*AEV
      EPS  = DMAX1(TEMP,SCALE3*BISTOL)
C     EPS  = DMAX1(TEMP,RELTOL)
C
      NDIST1=NDIST -1
      TEMP = TEIG(NDIST)-TEIG(NDIST1)
      IF (TEMP.LT.EPS1.AND.TMULT(NDIST1).NE.0) GO TO 130
      IFIN = 1
C
      GO TO 20
C
  130 BETA(MF) = BETAM
C
C-----END OF PRTEST------------------------------------------------------
      RETURN
      END
C
C------START OF STURMI---------------------------------------------------
C
      SUBROUTINE STURMI(ALPHA,BETA,X1,TOLN,EPSM,MMAX,MK1,MK2,IC,IWRITE)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  ALPHA(1),BETA(1)
      DOUBLE PRECISION  EPSM,X1,TOLN,EVL,EVU,BETA2
      DOUBLE PRECISION  U1,U2,V1,V2,ZERO,ONE
      INTEGER I,IC,ICD,IC0,IC1,IC2,MK1,MK2,MMAX
C-----------------------------------------------------------------------
C
C     FOR ANY EIGENVALUE OF A THAT HAS CONVERGED AS AN EIGENVALUE
C     OF THE T-MATRICES THIS SUBROUTINE CALCULATES
C     THE SMALLEST SIZE OF THE T-MATRIX, T(1,MK1) DEFINED
C     BY THE ALPHA AND BETA ARRAYS SUCH THAT MK1.LE.MMAX
C     AND THE INTERVAL (X1-TOLN,X1+TOLN) CONTAINS AT LEAST ONE
C     EIGENVALUE OF T(1,MK1). IT ALSO CALCULATES MK2 <= MMAX
C     AS THE SMALLEST SIZE T-MATRIX (IF ANY) SUCH THAT THIS INTERVAL
C     CONTAINS AT LEAST TWO EIGENVALUES OF T(1,MK2).
C     IF NO T-MATRIX OF ORDER < MMAX SATISFIES THIS REQUIREMENT
C     THEN MK2 IS SET EQUAL TO MMAX.  THE EIGENVECTOR PROGRAM
```

```
C     USES THESE VALUES TO DETERMINE AN APPROPRIATE 1ST GUESS AT        LES11440
C     AN APPROPRIATE SIZE T-MATRIX FOR THE EIGENVALUE X1.               LES11450
C                                                                       LES11460
C     ON EXIT IC = NUMBER OF EIGENVALUES OF T(1,MK2) IN THIS INTERVAL   LES11470
C                                                                       LES11480
C     STURMI REGENERATES THE QUANTITIES BETA(I)**2 EACH TIME IT IS      LES11490
C     CALLED, OBVIOUSLY FOR THE PRICE OF ANOTHER VECTOR OF LENGTH       LES11500
C     MMAX THIS GENERATION COULD BE DONE ONCE IN THE MAIN               LES11510
C     PROGRAM BEFORE THE LOOP ON THE CALLS TO SUBROUTINE STURMI.        LES11520
C                                                                       LES11530
C     IF ANY OF THE EIGENVALUES BEING CONSIDERED WERE MULTIPLE          LES11540
C     AS EIGENVALUES OF THE USER-SPECIFIED MATRIX, THEN                 LES11550
C     THIS SUBROUTINE COULD BE MODIFIED TO COMPUTE ADDITIONAL           LES11560
C     SIZES MKJ, J = 3, ...  WHICH COULD THEN BE USED IN THE            LES11570
C     MAIN LANCZOS EIGENVECTOR PROGRAM TO COMPUTE ADDITIONAL            LES11580
C     EIGENVECTORS CORRESPONDING TO THESE MULTIPLE EIGENVALUES.         LES11590
C     THE MAIN PROGRAM PROVIDED DOES NOT INCLUDE THIS OPTION.           LES11600
C                                                                       LES11610
C-----------------------------------------------------------------------LES11620
C     INITIALIZATION OF PARAMETERS                                      LES11630
      MK1 = 0                                                           LES11640
      MK2 = 0                                                           LES11650
      ZERO = 0.0D0                                                      LES11660
      ONE  = 1.0D0                                                      LES11670
      BETA(1) = ZERO                                                    LES11680
      EVL = X1-TOLN                                                     LES11690
      EVU = X1+TOLN                                                     LES11700
      U1 = ONE                                                          LES11710
      U2 = ONE                                                          LES11720
      IC0 = 0                                                           LES11730
      IC1 = 0                                                           LES11740
      IC2 = 0                                                           LES11750
C                                                                       LES11760
C     MAIN LOOP FOR CALCULATING THE SIZES MK1,MK2                       LES11770
      DO 60 I = 1,MMAX                                                  LES11780
      BETA2 = BETA(I)*BETA(I)                                           LES11790
      IF (U1.NE.ZERO) GO TO 10                                          LES11800
      V1 = BETA(I)/EPSM                                                 LES11810
      GO TO 20                                                          LES11820
   10 V1 = BETA2/U1                                                     LES11830
   20 U1 = EVL - ALPHA(I) - V1                                          LES11840
      IF (U1.LT.ZERO) IC1 = IC1+1                                       LES11850
      IF (U2.NE.ZERO) GO TO 30                                          LES11860
      V2 = BETA(I)/EPSM                                                 LES11870
      GO TO 40                                                          LES11880
   30 V2 = BETA2/U2                                                     LES11890
   40 U2 = EVU - ALPHA(I) - V2                                          LES11900
      IF (U2.LT.ZERO) IC2 = IC2+1                                       LES11910
C     TEST FOR CHANGE IN NUMBER OF T-EIGENVALUES ON (EVL,EVU)           LES11920
      ICD = IC1-IC2                                                     LES11930
      IC = ICD-IC0                                                      LES11940
      IF (IC.GE.1) GO TO 50                                             LES11950
      GO TO 60                                                          LES11960
   50 CONTINUE                                                          LES11970
      IF (IC0.EQ.0) MK1 = I                                             LES11980
```

```
      ICO = ICO+1
      IF (ICO.GT.1) GO TO 70
   60 CONTINUE
C
      I = I-1
      IF (ICO.EQ.0) MK1 = MMAX
   70 MK2 = I
      IC = ICD
C
      IF (IWRITE.EQ.1) WRITE(6,80) X1,MK1,MK2,IC
   80 FORMAT(' EVAL =',E20.12,' MK1 =',I6,' MK2 =',I6,' IC =',I3/)
C
      RETURN
C-----END OF STURMI---------------------------------------------------
      END
C
C
C-----START OF INVERM-------------------------------------------------
C
      SUBROUTINE INVERM(ALPHA,BETA,V1,V2,X1,ERROR,ERRORV,EPS,G,MEV,IT,
     1 IWRITE)
C
C--------------------------------------------------------------------
      DOUBLE PRECISION  ALPHA(1),BETA(*),V1(1),V2(*)
      DOUBLE PRECISION  X1,U,Z,TEMP,RATIO,SUM,XU,NORM,TSUM,BETAM
      DOUBLE PRECISION  EPS,EPS3,EPS4,ERROR,ERRORV,ZERO,ONE
      REAL  G(1)
      DOUBLE PRECISION  DABS, DSQRT, DFLOAT
      DOUBLE PRECISION FINPRO
      REAL  ABS
C--------------------------------------------------------------------
C
C     COMPUTES T-EIGENVECTORS FOR ISOLATED GOOD T-EIGENVALUES X1
C     USING INVERSE ITERATION ON T(1,MEV(X1)) SOLVING EQUATION
C     (T - X1*I)V2 = RIGHT-HAND SIDE (RANDOMLY-GENERATED) .
C     PROGRAM REFACTORS T- X1*I ON EACH ITERATION OF INVERSE ITERATION.
C     TYPICALLY ONLY ONE ITERATION IS NEEDED PER T-EIGENVALUE X1.
C
C     IF IWRITE = 1 THEN THERE ARE EXTENDED WRITES TO FILE 6 (TERMINAL)
C
C     ON ENTRY G CONTAINS A REAL*4 RANDOM VECTOR WHICH WAS GENERATED
C     IN MAIN PROGRAM.
C
C     ON ENTRY AND EXIT
C     MEV = ORDER OF T
C     ALPHA, BETA CONTAIN THE DIAGONAL AND OFFDIAGONAL ENTRIES OF T.
C     EPS = 2. * MACHINE EPSILON
C
C     IN PROGRAM:
C     ITER = MAXIMUM NUMBER STEPS ALLOWED FOR INVERSE ITERATION
C     ITER = IT ON ENTRY.
C     V1,V2 = WORK SPACES USED IN THE FACTORIZATION OF T(1,MEV).
C     V1 AND V2 MUST BE OF DIMENSION AT LEAST MEV.
C
C     ON EXIT
```

```
C     V2 = THE UNIT EIGENVECTOR OF T(1,MEV) CORRESPONDING TO X1.
C     ERROR = |V2(MEV)| = ERROR ESTIMATE FOR CORRESPONDING
C             RITZ VECTOR FOR X1.
C
C     ERRORV = || T*V2 - X1*V2 || = ERROR ESTIMATE ON T-EIGENVECTOR.
C     IF IT.GT.ITER THEN ERRORV = -ERRORV
C     IT = NUMBER OF ITERATIONS ACTUALLY REQUIRED
C-----------------------------------------------------------------------
C     INITIALIZATION AND PARAMETER SPECIFICATION
      ONE  = 1.0D0
      ZERO = 0.0D0
      ITER = IT
      MP1 = MEV+1
      MM1 = MEV-1
      BETAM = BETA(MP1)
      BETA(MP1) = ZERO
C
C     CALCULATE SCALE AND TOLERANCES
      TSUM = DABS(ALPHA(1))
      DO 10 I = 2,MEV
   10 TSUM = TSUM + DABS(ALPHA(I)) + BETA(I)
C
      EPS3 = EPS*TSUM
      EPS4 = DFLOAT(MEV)*EPS3
C
C     GENERATE SCALED RANDOM RIGHT-HAND SIDE
      GSUM = ZERO
      DO 20 I = 1,MEV
   20 GSUM = GSUM+ABS(G(I))
      GSUM = EPS4/GSUM
C
C     INITIALIZE RIGHT HAND SIDE FOR INVERSE ITERATION
      DO 30 I = 1,MEV
   30 V2(I) = GSUM*G(I)
      IT = 1
C
C     CALCULATE UNIT EIGENVECTOR OF T(1,MEV) FOR ISOLATED GOOD
C     T-EIGENVALUE X1.
C
C     TRIANGULAR FACTORIZATION WITH NEAREST NEIGHBOR PIVOT
C     STRATEGY. INTERCHANGES ARE LABELLED BY SETTING BETA < 0.
C
   40 CONTINUE
      U = ALPHA(1)-X1
      Z = BETA(2)
C
      DO 60 I=2,MEV
      IF (BETA(I).GT.DABS(U)) GO TO 50
C     NO PIVOT INTERCHANGE
      V1(I-1) = Z/U
      V2(I-1) = V2(I-1)/U
      V2(I) = V2(I)-BETA(I)*V2(I-1)
      RATIO = BETA(I)/U
      U = ALPHA(I)-X1-Z*RATIO
      Z = BETA(I+1)
```

```
      GO TO 60                                                          LES13090
C     PIVOT INTERCHANGE                                                 LES13100
   50 CONTINUE                                                          LES13110
      RATIO = U/BETA(I)                                                 LES13120
      BETA(I) = -BETA(I)                                                LES13130
      V1(I-1) = ALPHA(I)-X1                                             LES13140
      U = Z-RATIO*V1(I-1)                                               LES13150
      Z = -RATIO*BETA(I+1)                                              LES13160
      TEMP = V2(I-1)                                                    LES13170
      V2(I-1) = V2(I)                                                   LES13180
      V2(I) = TEMP-RATIO*V2(I)                                          LES13190
   60 CONTINUE                                                          LES13200
C                                                                       LES13210
      IF (U.EQ.ZERO) U=EPS3                                             LES13220
C                                                                       LES13230
C     SMALLNESS TEST AND DEFAULT VALUE FOR LAST COMPONENT               LES13240
C     PIVOT(I-1) = |BETA(I)| FOR INTERCHANGE CASE                       LES13250
C     (I-1,I+1) ELEMENT IN RIGHT FACTOR = BETA(I+1)                     LES13260
C     END OF FACTORIZATION AND FORWARD SUBSTITUTION                     LES13270
C                                                                       LES13280
C     BACK SUBSTITUTION                                                 LES13290
      V2(MEV) = V2(MEV)/U                                               LES13300
      DO 80 II = 1,MM1                                                  LES13310
      I = MEV-II                                                        LES13320
      IF (BETA(I+1).LT.ZERO) GO TO 70                                   LES13330
C     NO PIVOT INTERCHANGE                                              LES13340
      V2(I) = V2(I)-V1(I)*V2(I+1)                                       LES13350
      GO TO 80                                                          LES13360
C     PIVOT INTERCHANGE                                                 LES13370
   70 BETA(I+1) = -BETA(I+1)                                            LES13380
      V2(I) = (V2(I)-V1(I)*V2(I+1)-BETA(I+2)*V2(I+2))/BETA(I+1)         LES13390
   80 CONTINUE                                                          LES13400
C                                                                       LES13410
C                                                                       LES13420
C     TESTS FOR CONVERGENCE OF INVERSE ITERATION                        LES13430
C     IF SUM |V2| COMPS. LE. 1 AND IT. LE. ITER DO ANOTHER INVIT STEP   LES13440
C                                                                       LES13450
      NORM = DABS(V2(MEV))                                              LES13460
      DO 90 II = 1,MM1                                                  LES13470
      I = MEV-II                                                        LES13480
   90 NORM = NORM+DABS(V2(I))                                           LES13490
C                                                                       LES13500
C     IS DESIRED GROWTH IN VECTOR ACHIEVED ?                            LES13510
C     IF NOT, DO ANOTHER INVERSE ITERATION STEP UNLESS NUMBER ALLOWED ISLES13520
C     EXCEEDED.                                                         LES13530
      IF (NORM.GE.ONE) GO TO 110                                        LES13540
C                                                                       LES13550
      IT=IT+1                                                           LES13560
      IF (IT.GT.ITER) GO TO 110                                         LES13570
C                                                                       LES13580
      XU = EPS4/NORM                                                    LES13590
      DO 100 I=1,MEV                                                    LES13600
  100 V2(I) = V2(I)*XU                                                  LES13610
C                                                                       LES13620
      GO TO 40                                                          LES13630
```

```
C                                                                       LES13640
C     NORMALIZE COMPUTED T-EIGENVECTOR : V2 = V2/||V2||                 LES13650
C                                                                       LES13660
  110 CONTINUE                                                          LES13670
C                                                                       LES13680
      SUM = FINPRO(MEV,V2(1),1,V2(1),1)                                 LES13690
      SUM = ONE/DSQRT(SUM)                                              LES13700
      DO 120 II = 1,MEV                                                 LES13710
  120 V2(II) = SUM*V2(II)                                               LES13720
C                                                                       LES13730
C     SAVE ERROR ESTIMATE FOR LATER OUTPUT                              LES13740
      ERROR = DABS(V2(MEV))                                             LES13750
C                                                                       LES13760
C     GENERATE ERRORV = ||T*V2 - X1*V2||.                               LES13770
      V1(MEV) = ALPHA(MEV)*V2(MEV)+BETA(MEV)*V2(MEV-1)-X1*V2(MEV)       LES13780
      DO 130 J = 2,MM1                                                  LES13790
      JM = MP1 - J                                                      LES13800
      V1(JM) = ALPHA(JM)*V2(JM) + BETA(JM)*V2(JM-1) + BETA(JM+1)*V2(JM+1LES13810
     1) - X1*V2(JM)                                                     LES13820
  130 CONTINUE                                                          LES13830
C                                                                       LES13840
      V1(1) = ALPHA(1)*V2(1) + BETA(2)*V2(2) - X1*V2(1)                 LES13850
      ERRORV = FINPRO(MEV,V1(1),1,V1(1),1)                              LES13860
      ERRORV = DSQRT(ERRORV)                                            LES13870
      IF (IT.GT.ITER) ERRORV = -ERRORV                                  LES13880
      IF (IWRITE.EQ.0) GO TO 150                                        LES13890
C                                                                       LES13900
C     FILE 6 (TERMINAL) OUTPUT OF ERROR ESTIMATES.                      LES13910
      WRITE(6,140) MEV,X1,ERROR,ERRORV                                  LES13920
  140 FORMAT(2X,'TSIZE',15X,'EIGENVALUE',11X,'U(M)',9X,'ERRORV'/        LES13930
     1 I6,E25.16,2E15.5)                                                LES13940
C                                                                       LES13950
C     RESTORE BETA(MEV+1) = BETAM                                       LES13960
  150 CONTINUE                                                          LES13970
      BETA(MP1) = BETAM                                                 LES13980
C-----END OF INVERM-----------------------------------------------------LES13990
      RETURN                                                            LES14000
      END                                                               LES14010
C                                                                       LES14020
C-----START OF LBISEC---------------------------------------------------LES14030
C                                                                       LES14040
      SUBROUTINE LBISEC(ALPHA,BETA,EPSM,EVAL,EVALN,LB,UB,TTOL,M,NEVT)   LES14050
C                                                                       LES14060
C-----------------------------------------------------------------------LES14070
      DOUBLE PRECISION  ALPHA(1),BETA(1),X0,X1,XL,XU,YU,YV,LB,UB        LES14080
      DOUBLE PRECISION  EPSM,EP1,EVAL,EVALN,EVD,EPT                     LES14090
      DOUBLE PRECISION  ZERO,ONE,HALF,TTOL,TEMP                         LES14100
      DOUBLE PRECISION  DABS,DSQRT,DFLOAT                               LES14110
C-----------------------------------------------------------------------LES14120
C     SPECIFY PARAMETERS                                                LES14130
      ZERO = 0.0D0                                                      LES14140
      HALF = 0.5D0                                                      LES14150
      ONE  = 1.0D0                                                      LES14160
      XL = LB                                                           LES14170
      XU = UB                                                           LES14180
```

```
C
C     EP1 = DSQRT(1000+M)*TTOL     TTOL = EPSM*TKMAX
C     TKMAX = MAX(|ALPHA(K)|,BETA(K), K= 1,KMAX)
C
      TEMP = DFLOAT(1000+M)
      EP1 = DSQRT(TEMP)*TTOL
C
      NA = 0
      X1 = XU
      JSTURM = 1
      GO TO 60
C     FORWARD STURM CALCULATION
   10 NA = NEV
      X1 = XL
      JSTURM = 2
      GO TO 60
C     FORWARD STURM CALCULATION
   20 NEVT = NEV
C
C     WRITE(6,30) M,EVAL,NEVT,EP1
   30 FORMAT(/3X,'TSIZE',23X,'EV',9X/I8,E25.16/
     1 I6,' = NUMBER OF T(1,M) EIGENVALUES ON TEST INTERVAL'/
     1 E12.3,' =  CONVERGENCE TOLERANCE'/)
C
      IF (NEVT.NE.1) GO TO 120
C
C     BISECTION LOOP
      JSTURM = 3
   40 X1 = HALF*(XL+XU)
      X0 = XU-XL
      EPT = EPSM*(DABS(XL) + DABS(XU)) + EP1
C     CONVERGENCE TEST
      IF (X0.LE.EP1) GO TO 100
      GO TO 60
C     FORWARD STURM CALCULATION
   50 CONTINUE
      IF(NEV.EQ.0) XU = X1
      IF(NEV.EQ.1) XL = X1
      GO TO 40
C     NEV = NUMBER OF T-EIGENVALUES OF T(1,M) ON (X1,XU)
C     THERE IS EXACTLY ONE T-EIGENVALUE OF T(1,M) ON (XL,XU)
C
C     FORWARD STURM CALCULATION
   60 NEV = -NA
      YU = ONE
      DO 90 I = 1,M
      IF (YU.NE.ZERO) GO TO 70
      YV = BETA(I)/EPSM
      GO TO 80
   70 YV = BETA(I)*BETA(I)/YU
   80 YU = X1 - ALPHA(I) - YV
      IF (YU.GE.ZERO) GO TO 90
      NEV = NEV+1
   90 CONTINUE
      GO TO (10,20,50), JSTURM
```

```
C
  100 CONTINUE
C
      EVALN = X1
      EVD = DABS(EVALN-EVAL)
C     WRITE(6,110) EVALN,EVAL,EVD
  110 FORMAT(/20X,'EVALN',21X,'EVAL',6X,'CHANGE'/2E25.16,E12.3/)
C
  120 CONTINUE
      RETURN
C-----END OF LBISEC-----------------------------------------------------
      END
C-----START OF COMPLEX INNER PRODUCT------------------------------------
C
C     COMPLEX INNER PRODUCT
C
      SUBROUTINE CINPRD(V2,V1,SUM,N)
C-----------------------------------------------------------------------
      DOUBLE PRECISION  ZERO,SUM
      COMPLEX*16 V2(1),V1(1),SUMC
C-----------------------------------------------------------------------
C
C     NOTE THAT THE ORDER MATTERS HERE
C     COMPUTES THE INNER PRODUCT OF THE CONJUGATE OF V2 WITH V1.
      ZERO = 0.D0
      SUMC = DCMPLX(ZERO,ZERO)
      DO 10 J=1,N
   10 SUMC = SUMC + DCONJG(V2(J))*V1(J)
      SUM = DREAL(SUMC)
C
      RETURN
C-----END OF COMPLEX INNER PRODUCT SUBROUTINE---------------------------
      END
C
C-----LPERM-PERMUTES VECTORS--------------------------------------------
C
      SUBROUTINE LPERM(W,U,IPERM)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  U(1),W(1)
      INTEGER  IPR(1),IPT(1)
C-----------------------------------------------------------------------
C     SUBROUTINE HAS 2 BRANCHES:  IPERM = 1,  CALCULATES
C     U = P*W
C     LET J = IPR(K) THEN U(K) = W(J), K = 1,N.
C     IPERM = 2, CALCULATES
C     U = P'*W  LET J = IPT(K) THEN U(K) = W(J), K=1,N.
C-----------------------------------------------------------------------
      GO TO 3
      ENTRY LPERME(IPR,IPT,N)
      GO TO 4
C-----------------------------------------------------------------------
C
    3 IF(IPERM.EQ.2) GO TO 30
C
```

```
C-----------------------------------------------------------------------
C     IPERM = 1
      DO 10 K = 1,N
      J = IPR(K)
   10 U(K) = W(J)
      GO TO 60
C----------------------------------------------------------------------
C     IPERM = 2
   30 DO 40 K = 1,N
      J = IPT(K)
   40 U(K) = W(J)
C----------------------------------------------------------------------
   60 CONTINUE
      DO 50 K = 1,N
   50 W(K) = U(K)
C
    4 RETURN
C
C-----END OF LPERM--------------------------------------------------
      END
```

2.7 LECOMPAC: Optional Preprocessing Program

```
C-----LECOMPAC-(STAND-ALONE PROGRAM)-----------------------------------
C  AUTHOR:   RALPH A. WILLOUGHBY (Deceased)
C
C
C
C          E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C
C     THIS PROGRAM TRANSLATES A SPARSE SYMMETRIC N X N MATRIX A,
C     GIVEN AS I, J, A(I,J), INTO THE SPARSE MATRIX FORMAT
C     REQUIRED BY THE SAMPLE USPEC AND CMATV PROGRAMS PROVIDED
C     FOR USE WITH THE LANCZOS EIGENVALUE/EIGENVECTOR PROCEDURES.
C     THIS PROGRAM ASSUMES THAT THE MATRIX ENTRIES ARE PROVIDED
C     EITHER COLUMN BY COLUMN OR ROW BY ROW.
C     NOTE THAT THIS PROGRAM DOES NOT DIRECTLY APPLY TO THE
C     HERMITIAN CASE BECAUSE FOR HERMITIAN MATRICES THE DIAGONALS
C     ARE REAL AND THE OFF-DIAGONAL ENTRIES ARE COMPLEX VARIABLES.
C
C     NONPORTABLE STATEMENTS:  PFORT VERIFIER INDICATES THAT THIS
C                              IS PORTABLE.
C
C----------------------------------------------------------------------
      DOUBLE PRECISION A(15000), AD(2000)
      DOUBLE PRECISION ZERO
      INTEGER IROW(15000),ICOL(15000)
C----------------------------------------------------------------------
C    INPUT FILE 7 CONTAINS THE SPARSE SYMMETRIC NXN MATRIX STORED AS:
C
C               NZ,M,N,MATNO
C               I(K)  J(K)  A(K)  K = 1,NZ
C
C     WHERE NZ IS THE TOTAL NUMBER OF NONZEROS IN THE MATRIX A,
C     N IS THE ROW AND COLUMN DIMENSION OF A,
C     AND A(K) ARE THE NONZERO ENTRIES STORED ROW BY ROW OR
C     COLUMN BY COLUMN.  PROGRAM READS THIS IN AS IROW(K) = I(K),
C     ICOL(K) = J(K), AND A(K) = A(K).
C
C     OUTPUT FILE = 8 CONTAINS THE A-MATRIX IN SPARSE FORMAT
C
C             NZS,N,NZL,MATNO
```

```
C              ICOL(K)    K = 1,NZL
C              IROW(K)    K = 1,NZS
C                AD(K)    K = 1,N
C                 A(K)    K = 1,NZS
C
C      WHERE N IS THE ORDER OF THE INPUT MATRIX A,
C      NZ EQUALS THE NUMBER OF NONZERO ELEMENTS IN A WHICH ARE ON
C      OR BELOW THE MAIN DIAGONAL.  NZL EQUALS THE NUMBER OF THE
C      LAST COLUMN HAVING NONZEROES BELOW THE DIAGONAL IN A.
C      NZS EQUALS THE NUMBER OF NONZERO ELEMENTS BELOW THE MAIN
C      DIAGONAL.   AD(K), K=1,N, CONTAINS THE DIAGONAL ELEMENTS OF A.
C      A(K), K=1,NZS, CONTAINS THE KTH NONZERO SUB-DIAGONAL ELEMENT
C      OF THE INPUT MATRIX.  A IS STORED COLUMN BY COLUMN.
C      IROW(K), K=1,NZS, CONTAINS THE ROW INDEX OF THE  NONZERO
C      STRICTLY LOWER TRIANGULAR ELEMENT A(K).
C      ICOL(K), K=1,NZL, EQUALS THE NUMBER OF STRICTLY LOWER
C      TRIANGULAR NONZEROES IN COLUMN K OF THE INPUT MATRIX.
C
C-----------------------------------------------------------------------
      ZERO = 0.0D0
C
      READ(7,10) NZ,N,MATNO,IIROW
   10 FORMAT(2I6,I8,I4)
C
      WRITE(6,20) NZ,N,MATNO,IIROW
   20 FORMAT(I10,I6,I10,' = NO. NONZERO AIJ J.GE.I, ORDER OF A, MATNO'/
     1 I6,' = IIROW IF IIROW=0 ORDERING IS BY COLS IIROW=1 BY ROWS'/)
C
      DO 30 K = 1,N
   30 AD(K) = ZERO
C
      IF (IIROW.EQ.0) READ(7,40) (IROW(K),ICOL(K),A(K), K=1,NZ)
C
      IF (IIROW.EQ.1) READ(7,40) (ICOL(K),IROW(K),A(K), K=1,NZ)
   40 FORMAT(2I5,E14.7)
C
      LCOUNT = 0
      K = 1
C
C     START OF A NEW COLUMN
   50 CONTINUE
      J = ICOL(K)
      ICOL(J) = 0
   60 CONTINUE
C
      IF (J.NE.IROW(K)) GO TO 70
C
C     DIAGONAL CASE
      AD(J) = A(K)
      GO TO 80
C
C     SUB-DIAGONAL NONZERO
   70 CONTINUE
      NZL = J
      LCOUNT = LCOUNT + 1
```

```
      A(LCOUNT) = A(K)
      IROW(LCOUNT) = IROW(K)
      ICOL(J) = ICOL(J) + 1
C
   80 CONTINUE
      K = K+1
C
      IF(K.GT.NZ) GO TO 90
C
      IF(ICOL(K).GT.J) GO TO 50
C
      GO TO 60
C
   90 CONTINUE
      NZS = LCOUNT
C
      WRITE(8,100) NZS,N,NZL,MATNO
      WRITE(6,100) NZS,N,NZL,MATNO
  100 FORMAT(I10,2I6,I8,' = NZS N NZL MATNO')
C
      WRITE(8,110) (ICOL(I), I=1,NZL)
      WRITE(8,110) (IROW(K), K=1,NZS)
  110 FORMAT(13I6)
C
      WRITE(8,120) (AD(K), K=1,N)
      WRITE(8,120) (A(K), K=1,NZS)
  120 FORMAT(4E19.10)
C
C-----END LECOMPAC------------------------------------------------------
      STOP
      END
```

2.8 LEVAL: LEVEC: File Definitions, Sample Input Files

Below is a listing of the input/output filew which are accessed by the real symmetric Lanczos eigenvalue program, LEVAL. Included also is a sample of the input file which LEVAL requires on file 5. The parameters are supplied in free format. LEVAL computes eigenvalues of real symmetric matrices A on user-specified intervals which must be supplied in ascending order. File 8 is assumed to contain the data which defines the real symmetric nxn matrix A.

```
Sample Specifications of the Input/Output Files for LEVAL
-----------------------------------------------------------------
 LEVAL EXEC LANCZOS EIGENVALUE CALCULATION REAL SYMMETRIC MATRICES
FI 06 TERM
FILEDEF  1 DISK &1        NHISTORY  A (RECFM F LRECL 80 BLOCK 80
FILEDEF  2 DISK &1        HISTORY   A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1        GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1        ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK LEVAL     INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1        INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1        DISTINCT  A (RECFM F LRECL 80 BLOCK 80
LOAD   LEVAL   LESUB   LEMULT
-----------------------------------------------------------------
```

```
Sample Input File for LEVAL
-----------------------------------------------------------
 LANCZOS EIGENVALUE COMPUTATIONS, NO REORTHOGONALIZATION
 TEST MATRIX
LINE 1     N    KMAX    NMEVS     MATNO
         143     429        1    706830
LINE 2   SVSEED    RHSEED      MXINIT    MXSTUR
        7892713    147935           5    100000
LINE 3   ISTART     ISTOP
              0         1
LINE 4       IHIS      IDIST  IWRITE
                1        0       1
LINE 5   RELTOL (RELATIVE TOLERANCE IN 'COMBINING' GOODEV)
    .0000000001
LINE 6   MB(1)   MB(2)   MB(3)   MB(4) (ORDERS OF T(1,MEV))
           190
LINE 7    NINT     (NUMBER OF SUB-INTERVALS FOR BISEC)
             1
LINE 8    LB(1)    LB(2)   LB(3)  (INTERVAL LOWER BOUNDS)
            0.0
LINE 9    UB(1)    UB(2)   UB(3)  (INTERVAL UPPER BOUNDS)
          1.001
-----------------------------------------------------------
```

Below is a listing of the input/output filew which are accessed by the real symmetric Lanczos eigenvector program, LEVEC. Included also is a sample of the input file which LEVEC requires on file 5. The parameters are supplied in free format. LEVEC computes eigenvectors for each of a user-specified subset of the eigenvalues computed by the companion program LEVAL. Eigenvector approximation are computed only for eigenvalue approximations which have 'converged'.

```
Sample Specifications of the Input/Output Files for LEVEC
----------------------------------------------------------------------
 LEVEC EXEC TO RUN LANCZOS EIGENVECTOR PROGRAM, REAL SYMMETRIC MATRICES
FI  06 TERM
FILEDEF  2 DISK &1       HISTORY   A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1       GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1       ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK LEVEC    INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1       INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  9 DISK &1       ERREST    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 10 DISK &1       BOUNDS    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1       TEIGVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 12 DISK &1       RITZVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 13 DISK &1       PAIGE     A (RECFM F LRECL 80 BLOCK 80
LOAD  LEVEC  LESUB  LEMULT
----------------------------------------------------------------------
```

```
Sample Input File for LEVEC
----------------------------------------------------------------------
 LEVEC REAL SYMMETRIC EIGENVECTOR COMPUTATIONS, NO REORTHOGONALIZATION
LINE 1  MDIMTV    MDIMRV  MBETA(MAX.DIMENSIONS, TVEC, RITVEC AND BETA
         10000     10000   2000
LINE 2       RELTOL
        .0000000001
LINE 3  MBOUND   NTVCON SVTVEC IREAD (FLAGS
            0        1      0     1
LINE 4  TVSTOP   LVCONT ERCONT  IWRITE (FLAGS
            0        1      1      1
LINE 5    RHSEED  (RANDOM GENERATOR SEED FOR STARTING VECTOR IN INVERM
        45329517
LINE 6  MATNO      N
          100    100
----------------------------------------------------------------------
```

Chapter 3

Hermitian Matrices

3.1 Introduction

The FORTRAN codes in this chapter address the question of computing distinct eigenvalues and corresponding eigenvectors of Hermitian matrices, using a single-vector Lanczos procedure. For a given Hermitian matrix A, these codes compute real scalars λ and corresponding complex vectors $x \neq 0$ such that

$$Ax = \lambda x. \tag{3.1.1}$$

Definition 2 *A complex nxn matrix A, $A \equiv (a_{ij}), 1 \leq i, j \leq n$, is a Hermitian matrix if and only if for every i and j, $a_{ij} = \overline{a_{ji}}$, where the overbar denotes the complex conjugate of the complex-valued entry a_{ij}.*

It is straight-forward to demonstrate from Definition 2 that for any Hermitian matrix $A = B + Ci$, where B and C are real matrices and $i = \sqrt{-1}$, that B must be a real symmetric matrix and C must be a skew symmetric matrix. That is, $B^T = B$ and $C^T = -C$. Futhermore, it is not difficult to see that Hermitian matrices must have real diagonal entries and real eigenvalues. However, the eigenvectors are complex-valued. Any Hermitian matrix can be transformed into a real symmetric tridiagonal matrix for the purposes of computing the eigenvalues of the Hermitian matrix, Stewart [24]. In fact, the Lanczos recursion which we use in the codes in this chapter transforms the given Hermitian matrix A into a family of real symmetric tridiagonal matrices rather than into a family of Hermitian tridiagonal matrices.

Hermitian matrices possess the 'same' properties as real symmetric matrices do, except that these properties are defined with respect to the complex or Hermitian norm, rather than with respect to the Euclidean norm, see Stewart [24]. The Hermitian norm of a given complex-valued vector $x \equiv (x(i)), 1 \leq i \leq n$, is defined as $\|x\|_C^2 \equiv \sum_{i=1}^n \overline{x(i)} x(i)$. Three properties which we use are:

1. Hermitian matrices have complete eigensystems. That is, the dimension of the eigenspace corresponding to any given eigenvalue of a Hermitian matrix is the same as the multiplicity of that eigenvalue as a root of the characteristic polynomial of that matrix.

2. For any two distinct eigenvalues λ, μ and corresponding eigenvectors x, y, $x^H y = 0$, where the superscript H denotes the complex conjugate transpose of the vector x. The complex conjugate transpose of a column vector x is the row vector whose i^{th} component is $\overline{x(i)}$. There is a complete set of eigenvectors $X_n \equiv (x_1, \ldots, x_n)$ such that X is a unitary matrix.

3. Small Hermitian perturbations in a Hermitian matrix cause only small perturbations in the eigenvalues.

The single-vector Lanczos codes in this chapter can be used to compute either a very few or very many of the distinct eigenvalues of the given Hermitian matrix. The documentation for these codes is contained in Chapter 2, Section 2.2. As in the real symmetric case, the A-multiplicity of a given computed 'good' Lanczos eigenvalue can be obtained only with additional computation, and the modifications required to do this additional computation are not included in these versions of the codes. This implementation uses a Hermitian analog of the basic Lanczos recursion contained in Eqns (1.2.1) and (1.2.2) to generate a family of real symmetric tridiagonal matrices whose sizes are specified by the user. There is no reorthogonalization of the Lanczos vectors at any stage in any of the computations.

The Hermitian version of the Lanczos recursion which we use is given below. For $i = 1, 2, ..., m$ and a randomly-generated complex starting vector v_1 with $\|v_1\|_C = 1$, generate Lanczos vectors v_i using the following recursion.

$$\beta_{i+1} v_{i+1} = A v_i - \alpha_i v_i - \beta_i v_{i-1}, \tag{3.1.2}$$

where

$$\alpha_i \equiv v_i^H A v_i, \quad \beta_{i+1} = \|A v_i - \alpha_i v_i - \beta_i v_{i-1}\|_C \tag{3.1.3}$$

We see from Eqns(3.1.3) that the Hermitian inner product is used. This is the 'natural' inner product for Hermitian matrices. Gram-Schmidt orthogonalization is used, unlike the real symmetric case where a modified Gram-Schmidt orthogonalization was used. This change in the local orthogonalization procedure increases the storage requirements for the implementation of the Lanczos recursion by one additional complex vector of length equal to the order of the original A-matrix. Modified Gram-Schmidt orthogonalization cannot be used in the Hermitian case because corrections to the α_i defined by this modification are complex-valued not real, and it would not be legitimate to accept the real portions of these corrections and simply ignore the complex portions.

It is easy to demonstrate that as we stated earlier, each Lanczos matrix (T-matrix) generated by this Hermitian recursion is a real symmetric tridiagonal matrix. In particular, we see from the formulas in Eqn(3.1.3) that the diagonal entries of each of these matrices are Rayleigh quotients of the given Hermitian matrix A, and therefore must all be real-valued. Furthermore by construction, the nonzero off-diagonal entries β_{i+1} are all real-valued. This use of real-valued β_i requires some justification. This justification is given in Section 4.9 of Chapter 4 of Volume 1 of this book.

HLEVAL, the main program for the Hermitian eigenvalue computations, calls the subroutine BISEC to compute eigenvalues of the specified tridiagonal Lanczos matrices on the user-specified intervals. BISEC simultaneously computes these T-eigenvalues with their T-multiplicities and sorts the computed T-eigenvalues into two classes, the 'good' T-eigenvalues and the 'spurious' T-eigenvalues. The 'good' T-eigenvalues are accepted as approximations to eigenvalues of the user-specified matrix A. The accuracy of these 'good' T-eigenvalues as eigenvalues of A is then estimated using error estimates computed by subroutine INVERR. Error estimates are computed only for isolated 'good' T-eigenvalues. All other 'good' T-eigenvalues are assumed to have converged. Convergence is then checked. If convergence has not yet occurred and a larger T-matrix has been specified by the user, the program will continue on to the larger T-matrix, repeating the above procedure on this larger matrix.

Once the eigenvalues have been computed accurately enough, the user can select a subset of the 'converged' eigenvalues for which eigenvectors are to be computed. The main program HLEVEC, for computing eigenvectors of Hermitian matrices, is then used to compute these desired eigenvectors.

The computations in the Lanczos recursion are a mixture of double precision real arithmetic and of double precision complex arithmetic. Once the Lanczos matrices have been computed, the remaining computations are all done in double precision real arithmetic, using the same subroutines that are used in the real symmetric case. In addition to the programs and subroutines provided here, the user must supply a subroutine USPEC which defines and initializes the user-specified matrix A and a subroutine CMATV which computes matrix-vector multiplies Ax for any given vector x. These subroutines must be constructed in such a way as to take advantage of the sparsity (and/or structure) of the user-supplied A-matrix and such that these computations are done accurately.

3.2 HLEVAL: Main Program, Eigenvalue Computations

```
C-----HLEVAL  (EIGENVALUES OF HERMITIAN MATRICES)----------------------HHL00010
C  Authors:  Jane Cullum and Ralph A. Willoughby (deceased)            HHL00020
C            Los Alamos National Laboratory                            HHL00030
C            Los Alamos, New Mexico 87544                              HHL00040
C            cullumj@lanl.gov                                          HHL00045
c                                                                      HHL00050
c  These codes are copyrighted by the authors.  These codes            HHL00060
c  and modifications of them or portions of them are NOT to be         HHL00070
c  incorporated into any commercial codes without legal agreements     HHL00080
c  with the authors.  If these codes or portions of them               HHL00090
c  are used in other scientific or engineering research works          HHL00100
c  the names of the authors of these codes and appropriate             HHL00110
c  references to their written work are to be incorporated in the      HHL00120
c  derivative works.                                                   HHL00130
c                                                                      HHL00140
c  This header is not to be removed from these codes.                  HHL00150
C                                                                      HHL00155
C         REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4           HHL00160
C         Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsHHL00165
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in  HHL00166
C         Applied Mathematics, 2002. SIAM Publications,                HHL00167
C         Philadelphia, PA. USA                                        HHL00168
C                                                                      HHL00169
C     CONTAINS MAIN PROGRAM FOR COMPUTING DISTINCT EIGENVALUES OF      HHL00170
C     A HERMITIAN MATRIX USING LANCZOS TRIDIAGONALIZATION WITHOUT      HHL00180
C     REORTHOGONALIZATION                                              HHL00190
C                                                                      HHL00200
C     PORTABILITY:                                                     HHL00210
C     THIS PROGRAM IS NOT PORTABLE DUE TO THE USE OF COMPLEX*16        HHL00220
C     VARIABLES.  MOREOVER, THE PFORT VERIFIER IDENTIFIED THE          HHL00230
C     FOLLOWING ADDITIONAL NONPORTABLE CONSTRUCTIONS:                  HHL00240
C                                                                      HHL00250
C     1.  DATA/MACHEP/ STATEMENT                                       HHL00260
C     2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                       HHL00270
C     3.  FORMAT(20A4) USED WITH EXPLANATORY HEADER EXPLAN.            HHL00280
C     4.  HEXADECIMAL FORMAT (4Z20) USED IN ALPHA/BETA FILES 1 AND 2.  HHL00290
C                                                                      HHL00300
C----------------------------------------------------------------------HHL00310
C     SPECIFY DIMENSIONS OF ARRAYS NEEDED BY LANCZOS ROUTINES          HHL00320
C                                                                      HHL00330
C     USER SPECIFIES THE FOLLOWING:                                    HHL00340
C     OTHER ARRAY DIMENSIONS ARE COMPUTED IN PARAMETER STATEMENTS      HHL00350
C     N = DIMENSION OF THE MATRIX EIGENVALUE PROBLEM                   HHL00360
C     KMAX =  MAXIMUM SIZE OF LANCZOS MATRICES TO BE USED              HHL00370
C     NSINT >= NUMBER OF SUBINTERVALS SPECIFIED IN INPUT FILE 5        HHL00380
C     NTMATS >= NUMBER OF LANCZOS MATRICES SPECIFIED IN INPUT FILE 5   HHL00390
C     BELOW WE ASSUME THAT NO MORE THAN KMAX/2 EIGENVALUES              HHL00400
C     ARE COMPUTED IN ANY ONE OF THE SUBINTERVALS (LB(J),UB(J))        HHL00410
C     SUPPLIED BY THE USER.  V2 WILL BE USED FOR BOTH UPPER AND        HHL00420
C     LOWER BOUNDS ON THE EIGENVALUES AS THEY ARE COMPUTED SO          HHL00430
C     IF MORE THAN KMAX/2 EIGENVALUES ARE TO BE COMPUTED IN ANY        HHL00440
```

```
C     ONE SUBINTERVAL, THE DIMENSION OF V2 MUST BE ADJUSTED             HHL00450
C     ACCORDINGLY.  FOR EXAMPLE IF THE USER WANTS ALL THE EIGENVALUES   HHL00460
C     OF THE LANCZOS MATRIX THEN KV2 MUST BE > MAX(KMAX,N)              HHL00470
C     BECAUSE OF THE INTEGER ARITHMETIC IT IS NECESSARY TO ADD AN       HHL00480
C     EXTRA 1 TO THE EXPRESSIONS.                                       HHL00490
C                                                                       HHL00500
C     TO AVOID USING MAX(I,J) IN THE PARAMETER LISTING WE HAVE USED     HHL00510
C     THE FOLLOWING EQUIVALENT RELATIONSHIP                             HHL00520
C                                                                       HHL00530
C     MAX(I,J) = ( 2*I/(I+J))*I  + (2*J/(I+J))*J                        HHL00540
C                                                                       HHL00550
C     PARAMETER ( N = 81, KMAX = 100, NSINT = 20, NTMATS = 20)          HHL00560
      PARAMETER ( N = 625, KMAX = 1500, NSINT = 20, NTMATS = 20)        HHL00570
C                                                                       HHL00580
      PARAMETER ( KMAX1 = KMAX+1, KMAX2 = 2*KMAX, NKMAX = N+KMAX )      HHL00590
      PARAMETER ( KMAXP2 = KMAX + 2)                                    HHL00600
      PARAMETER ( N2 = 2*N, N2KMAX = N2+KMAX, NKMAX2=N+KMAX2)           HHL00610
      PARAMETER ( KMAXPO2 = KMAXP2/2, KMAX1O2 = KMAX1/2 )               HHL00620
      PARAMETER ( NKMAX12 = N+KMAX1O2, NKMAXPO = N+KMAXPO2)             HHL00630
      PARAMETER (KVS = ((2*N2)/N2KMAX)*N2 + ((2*KMAX)/N2KMAX)*KMAX )    HHL00640
      PARAMETER (KV1 =((2*N)/NKMAXPO)*N+((2*KMAXPO2)/NKMAXPO)*KMAXPO2)  HHL00650
C     PARAMETER (KV2 = ((2*N)/NKMAX12)*N +((2*KMAX1O2)/NKMAX12)*KMAX1O2)HHL00660
C  BELOW GOES WITH COMPUTING ALL EIGENVALUES OF LANCZOS MATRIX          HHL00670
      PARAMETER (KV2 = ((2*N)/NKMAX)*N + ((2*KMAX)/NKMAX)*KMAX)         HHL00680
      PARAMETER (KG = ((2*KMAX2)/NKMAX2)*KMAX2 +((2*N)/NKMAX2)*N )      HHL00690
C                                                                       HHL00700
C-----------------------------------------------------------------------HHL00710
C                                                                       HHL00720
      DOUBLE PRECISION  ALPHA(KMAX),BETA(KMAX1),VS(KVS)                 HHL00730
      COMPLEX*16  V1(KV1),V2(KV2)                                       HHL00740
      DOUBLE PRECISION  GR(N),GC(N),LB(NSINT),UB(NSINT)                 HHL00750
      DOUBLE PRECISION  BTOL,GAPTOL,TTOL,MACHEP,EPSM,RELTOL             HHL00760
      DOUBLE PRECISION  SCALE1,SCALE2,SCALE3,SCALE4,BISTOL,CONTOL,MULTOLHHL00770
      DOUBLE PRECISION  ONE,ZERO,TEMP,TKMAX,BETAM,BKMIN,T0,T1           HHL00780
      REAL  G(KG),EXPLAN(20)                                            HHL00790
      INTEGER  MP(KMAX),NMEV(NTMATS)                                    HHL00800
      INTEGER  SVSEED,RHSEED,SVSOLD                                     HHL00810
      INTEGER IABS                                                      HHL00820
      REAL  ABS                                                         HHL00830
      DOUBLE PRECISION  DABS, DSQRT, DFLOAT                             HHL00840
      EXTERNAL CMATV                                                    HHL00850
C                                                                       HHL00860
C-----------------------------------------------------------------------HHL00870
      DATA MACHEP/Z3410000000000000/                                    HHL00880
      EPSM = 2.0D0*MACHEP                                               HHL00890
C-----------------------------------------------------------------------HHL00900
      WRITE(6,1) N,KMAX,NSINT,NTMATS                                    HHL00910
    1 FORMAT(' N,KMAX,NSINT,NTMATS =’/4I10)                             HHL00920
      WRITE(6,2) KMAX1,KMAX2,N2,N2KMAX,NKMAX2                           HHL00930
    2 FORMAT(' KMAX1,KMAX2,N2,N2KMAX,NKMAX2 =’/5I10)                    HHL00940
      WRITE(6,3) KMAXPO2,KMAX1O2,NKMAXPO,NKMAX12                        HHL00950
    3 FORMAT(' KMAXPO2,KMAX1O2,NKMAXPO,NKMAX12 =’/4I10)                 HHL00960
      WRITE(6,4) KVS,KV1,KV2,KG                                         HHL00970
    4 FORMAT(' KVS,KV1,KV2,KG =’/4I10)                                  HHL00980
C                                                                       HHL00990
```

```
C     THE ARRAYS V1 AND V2 ARE DEFINED AS COMPLEX*16 IN THE MAIN PROGRAMHHL01000
C     AND IN THE SUBROUTINE LANCZS.  HOWEVER, IN THE OTHER SUBROUTINES  HHL01010
C     THEY ARE DECLARED AS DOUBLE PRECISION ARRAYS. NOTE THAT THE       HHL01020
C     DIMENSION OF V2 ASSUMES THAT NO MORE THAN KMAX/2 EIGENVALUES OF   HHL01030
C     THE T-MATRICES ARE BEING COMPUTED IN ANY ONE OF THE SUB-INTERVALS HHL01040
C     BEING CONSIDERED.  V2 MUST CONTAIN UPPER AND LOWER BOUNDS         HHL01050
C     ON EACH T-EIGENVALUE COMPUTED BY BISEC IN ANY ONE GIVEN INTERVAL. HHL01060
C                                                                       HHL01070
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:                            HHL01080
C          1. ALPHA:   >= KMAX.  BETA:  >= (KMAX+1)                     HHL01090
C          2. V1:  >= MAX(N,(KMAX+1)/2).   V2:  >= MAX(N,KMAX/2)        HHL01100
C          3. VS:  >= MAX(2*N,KMAX).                                    HHL01110
C          4. GR,GC:  >=  N                                             HHL01120
C          5. G:  >= MAX(2*KMAX,N)                                      HHL01130
C          6. MP:   >= KMAX                                             HHL01140
C          7. LB,UB:  >= NUMBER OF SUB-INTERVALS SPECIFIED              HHL01150
C          8. NMEV:  >= NUMBER OF T-MATRICES SPECIFIED                  HHL01160
C          9. EXPLAN:  DIMENSION IS 20.                                 HHL01170
C                                                                       HHL01180
C                                                                       HHL01190
C     IMPORTANT TOLERANCES OR SCALES THAT ARE USED REPEATEDLY           HHL01200
C     THROUGHOUT THE PROGRAM ARE THE FOLLOWING:                         HHL01210
C     SCALED MACHINE EPSILON:  TTOL = TKMAX*EPSM WHERE                  HHL01220
C     EPSM = 2*MACHINE EPSILON AND                                      HHL01230
C     TKMAX = MAX(|ALPHA(J)|,BETA(J), J = 1,MEV)                        HHL01240
C     BISEC CONVERGENCE TOLERANCE:  BISTOL = DSQRT(1000+MEV)*TTOL       HHL01250
C     BISEC MULTIPLICITY TOLERANCE:  MULTOL = (1000+MEV)*TTOL           HHL01260
C     LANCZOS CONVERGENCE TOLERANCE:   CONTOL = BETA(MEV+1)*1.D-10      HHL01270
C-----------------------------------------------------------------------HHL01280
C     OUTPUT HEADER                                                     HHL01290
      WRITE(6,10)                                                       HHL01300
   10 FORMAT(/' LANCZOS PROCEDURE FOR HERMITIAN MATRICES'/)             HHL01310
C                                                                       HHL01320
C     SET PROGRAM PARAMETERS                                            HHL01330
C     SCALEK ARE USED IN TOLERANCES NEEDED IN SUBROUTINES LUMP,         HHL01340
C     ISOEV AND PRTEST.  USER MUST NOT MODIFY THESE SCALES.             HHL01350
      SCALE1 = 5.0D2                                                    HHL01360
      SCALE2 = 5.0D0                                                    HHL01370
      SCALE3 = 5.0D0                                                    HHL01380
      SCALE4 = 1.0D4                                                    HHL01390
      ONE  = 1.0D0                                                      HHL01400
      ZERO = 0.0D0                                                      HHL01410
      BTOL = EPSM                                                       HHL01420
C     BTOL = 1.0D-8                                                     HHL01430
      GAPTOL = 1.0D-8                                                   HHL01440
      ICONV = 0                                                         HHL01450
      MOLD = 0                                                          HHL01460
      MOLD1 = 1                                                         HHL01470
      ICT = 0                                                           HHL01480
      MMB = 0                                                           HHL01490
      IPROJ = 0                                                         HHL01500
C                                                                       HHL01510
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 5 (FREE FORMAT)    HHL01520
C                                                                       HHL01530
C     READ USER-PROVIDED HEADER FOR RUN                                 HHL01540
```

```
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C  MODIFIED 4/16/93, N AND KMAX SET IN PARAMETER LIST.
C XXXXREAD ORDER OF MATRICES (N) , MAXIMUM ORDER OF T-MATRIX (KMAX),
C     NUMBER OF T-MATRICES ALLOWED (NMEVS), AND MATRIX IDENTIFICATION
C     NUMBERS (MATNO)
      READ(5,20) EXPLAN
      READ(5,*) NMEVS,MATNO
C     READ(5,*) N,KMAX,NMEVS,MATNO
C
C     READ SEEDS FOR LANCZS AND INVERR SUBROUTINES (SVSEED AND RHSEED)
C     READ MAXIMUM NUMBER OF ITERATIONS ALLOWED FOR EACH INVERSE
C     ITERATION (MXINIT) AND MAXIMUM NUMBER OF STURM SEQUENCES
C     ALLOWED (MXSTUR)
      READ(5,20) EXPLAN
      READ(5,*) SVSEED,RHSEED,MXINIT,MXSTUR
C
C     ISTART = (0,1):  ISTART = 0 MEANS ALPHA/BETA FILE IS NOT
C     AVAILABLE.  ISTART = 1 MEANS ALPHA/BETA FILE IS AVAILABLE ON
C     FILE 2.
C     ISTOP = (0,1):  ISTOP = 0 MEANS PROCEDURE GENERATES ALPHA/BETA
C     FILE AND THEN TERMINATES.  ISTOP = 1 MEANS PROCEDURE GENERATES
C     ALPHAS/BETAS IF NEEDED AND THEN COMPUTES EIGENVALUES AND ERROR
C     ESTIMATES AND THEN TERMINATES.
      READ(5,20) EXPLAN
      READ(5,*) ISTART,ISTOP
C
C     IHIS = (0,1):  IHIS = 0 MEANS ALPHA/BETA FILE IS NOT WRITTEN
C     TO FILE 1.  IHIS = 1 MEANS ALPHA/BETA FILE IS WRITTEN TO FILE 1.
C     IDIST = (0,1):  IDIST = 0 MEANS DISTINCT T-EIGENVALUES
C     ARE NOT WRITTEN TO FILE 11.  IDIST = 1 MEANS DISTINCT
C     T-EIGENVALUES ARE WRITTEN TO FILE 11.
C     IWRITE = (0,1):  IWRITE = 0 MEANS NO INTERMEDIATE OUTPUT
C     FROM THE COMPUTATIONS IS WRITTEN TO FILE 6.  IWRITE = 1 MEANS
C     T-EIGENVALUES AND ERROR ESTIMATES ARE WRITTEN TO FILE 6
C     AS THEY ARE COMPUTED.
      READ(5,20) EXPLAN
      READ(5,*) IHIS,IDIST,IWRITE
C
C     READ IN THE RELATIVE TOLERANCE (RELTOL) FOR USE IN THE
C     SPURIOUS, T-MULTIPLICITY, AND PRTESTS.
      READ(5,20) EXPLAN
      READ(5,*) RELTOL
C
C     READ IN THE SIZES OF THE T-MATRICES TO BE CONSIDERED.
      READ(5,20) EXPLAN
      READ(5,*) (NMEV(J), J=1,NMEVS)
C
C     READ IN THE NUMBER OF SUBINTERVALS TO BE CONSIDERED.
      READ(5,20) EXPLAN
      READ(5,*) NINT
```

```
C                                                                       HHL02100
C     READ IN THE LEFT-END POINTS OF THE SUBINTERVALS TO BE CONSIDERED. HHL02110
C     THESE MUST BE IN ALGEBRAICALLY-INCREASING ORDER                   HHL02120
      READ(5,20) EXPLAN                                                 HHL02130
      READ(5,*) (LB(J), J=1,NINT)                                       HHL02140
C                                                                       HHL02150
C     READ IN THE RIGHT-END POINTS OF THE SUBINTERVALS TO BE CONSIDERED.HHL02160
C     THESE MUST BE IN ALGEBRAICALLY-INCREASING ORDER                   HHL02170
      READ(5,20) EXPLAN                                                 HHL02180
      READ(5,*) (UB(J), J=1,NINT)                                       HHL02190
C                                                                       HHL02200
C-----------------------------------------------------------------------HHL02210
C                                                                       HHL02220
C     INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED MATRIX               HHL02230
C     AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE             HHL02240
C     MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV.                          HHL02250
C                                                                       HHL02260
      CALL USPEC(N,MATNO)                                               HHL02270
C                                                                       HHL02280
C-----------------------------------------------------------------------HHL02290
C                                                                       HHL02300
C     MASK UNDERFLOW AND OVERFLOW                                       HHL02310
C                                                                       HHL02320
      CALL MASK                                                         HHL02330
C                                                                       HHL02340
C-----------------------------------------------------------------------HHL02350
C                                                                       HHL02360
C     WRITE TO FILE 6, A SUMMARY OF THE PARAMETERS FOR THIS RUN         HHL02370
C                                                                       HHL02380
      WRITE(6,30) MATNO,N,KMAX                                          HHL02390
   30 FORMAT(/3X,'MATRIX ID',4X,'ORDER OF A',4X,'MAX ORDER OF T'/       HHL02400
     1 I12,I14,I18/)                                                    HHL02410
C                                                                       HHL02420
      WRITE(6,40) ISTART,ISTOP                                          HHL02430
   40 FORMAT(/2X,'ISTART',3X,'ISTOP'/2I8/)                              HHL02440
C                                                                       HHL02450
      WRITE(6,50) IHIS,IDIST,IWRITE                                     HHL02460
   50 FORMAT(/4X,'IHIS',3X,'IDIST',2X,'IWRITE'/3I8/)                    HHL02470
C                                                                       HHL02480
      WRITE(6,60) SVSEED,RHSEED                                         HHL02490
   60 FORMAT(/' SEEDS FOR RANDOM NUMBER GENERATOR'//                    HHL02500
     1 4X,'LANCZS SEED',4X,'INVERR SEED'/2I15/)                         HHL02510
C                                                                       HHL02520
      WRITE(6,70) (NMEV(J), J=1,NMEVS)                                  HHL02530
   70 FORMAT(/' SIZES OF T-MATRICES TO BE CONSIDERED'/(6I12))           HHL02540
C                                                                       HHL02550
      WRITE(6,80) RELTOL,GAPTOL,BTOL                                    HHL02560
   80 FORMAT(/' RELATIVE TOLERANCE USED TO COMBINE COMPUTED T-EIGENVALUEHHL02570
     1S'/E15.3/' RELATIVE GAP TOLERANCES USED IN INVERSE ITERATION'/    HHL02580
     1E15.3/' RELATIVE TOLERANCE FOR CHECK ON SIZE OF BETAS'/E15.3/)    HHL02590
C                                                                       HHL02600
      WRITE(6,90) (J,LB(J),UB(J), J=1,NINT)                             HHL02610
   90 FORMAT(/' BISEC WILL BE USED ON THE FOLLOWING INTERVALS'/         HHL02620
     1 (I6,2E20.6))                                                     HHL02630
C                                                                       HHL02640
```

```
      IF (ISTART.EQ.0) GO TO 140                                        HHL02650
C                                                                       HHL02660
C     READ IN ALPHA BETA HISTORY                                        HHL02670
C                                                                       HHL02680
      READ(2,100)MOLD,NOLD,SVSOLD,MATOLD                                HHL02690
  100 FORMAT(2I6,I12,I8)                                                HHL02700
C                                                                       HHL02710
C  CHANGED KMAX TO PARAMETER VARIABLE SO BELOW NO LONGER ALLOWED        HHL02720
C  SO DEFAULT TO TERMINATE IF HISTORY FILE IS NOT LONG ENOUGH           HHL02730
C     IF (KMAX.LT.MOLD) KMAX = MOLD                                     HHL02740
C     KMAX1 = KMAX + 1                                                  HHL02750
C                                                                       HHL02760
      IF (KMAX.LT.MOLD) WRITE(6,115) KMAX,MOLD                          HHL02770
      IF (KMAX.LT.MOLD) GO TO 640                                       HHL02780
  115 FORMAT(/' PROGRAM TERMINATES FOR USER TO RESET KMAX.  CURRENT VALUHHL02790
     1E',I6/' IS LARGER THAN THE SIZE',I6,' OF THE TRIDIAGONAL MATRIX ONHHL02800
     1FILE 2'/)                                                         HHL02810
C                                                                       HHL02820
C     CHECK THAT ORDER N, MATRIX ID MATNO, AND RANDOM SEED SVSEED       HHL02830
C     AGREE WITH THOSE IN THE HISTORY FILE.  IF NOT PROCEDURE STOPS.    HHL02840
C                                                                       HHL02850
      ITEMP = (NOLD-N)**2+(MATNO-MATOLD)**2+(SVSEED-SVSOLD)**2          HHL02860
C                                                                       HHL02870
      IF (ITEMP.EQ.0) GO TO 120                                         HHL02880
C                                                                       HHL02890
      WRITE(6,110)                                                      HHL02900
  110 FORMAT(' PROGRAM TERMINATES'/    ' READ FROM FILE 2 CORRESPONDS TOHHL02910
     1 DIFFERENT MATRIX THAN MATRIX SPECIFIED'/)                        HHL02920
      GO TO 640                                                         HHL02930
C                                                                       HHL02940
  120 CONTINUE                                                          HHL02950
      MOLD1 = MOLD+1                                                    HHL02960
C                                                                       HHL02970
      READ(2,130)(ALPHA(J), J=1,MOLD)                                   HHL02980
      READ(2,130)(BETA(J), J=1,MOLD1)                                   HHL02990
  130 FORMAT(4Z20)                                                      HHL03000
C                                                                       HHL03010
      IF (KMAX.EQ.MOLD) GO TO 160                                       HHL03020
C                                                                       HHL03030
      READ(2,130)(V1(J), J=1,N)                                         HHL03040
      READ(2,130)(V2(J), J=1,N)                                         HHL03050
C                                                                       HHL03060
  140 CONTINUE                                                          HHL03070
      IIX = SVSEED                                                      HHL03080
C                                                                       HHL03090
C-----------------------------------------------------------------------HHL03100
C                                                                       HHL03110
      CALL LANCZS(CMATV,V1,V2,VS,ALPHA,BETA,GR,GC,G,KMAX,MOLD1,N,IIX)   HHL03120
C                                                                       HHL03130
C-----------------------------------------------------------------------HHL03140
C                                                                       HHL03150
C  COMMENTED OUT BELOW BECAUSE KMAX1 IS NOW SET IN PARAMETER LIST       HHL03160
C     KMAX1 = KMAX + 1                                                  HHL03170
C                                                                       HHL03180
      IF (IHIS.EQ.0.AND.ISTOP.GT.0) GO TO 160                           HHL03190
```

```
C
      WRITE(1,150) KMAX,N,SVSEED,MATNO
  150 FORMAT(2I6,I12,I8,' = KMAX,N,SVSEED,MATNO')
C
C     TO AVOID PERTURBATIONS CAUSED BY HEX TO DECIMAL AND DECIMAL TO HEX
C     CONVERSIONS, THE ALPHA AND BETA MUST BE WRITTEN OUT IN HEX.
      WRITE(1,130)(ALPHA(I), I=1,KMAX)
      WRITE(1,130)(BETA(I), I=1,KMAX1)
C     WRITE(1,135)(ALPHA(I), I=1,N)
C     WRITE(1,135)(BETA(I), I=1,N)
  135 FORMAT(4E20.12)
C
C     WRITE(1,130)(V1(I), I=1,N)
C     WRITE(1,130)(V2(I), I=1,N)
C
      IF (ISTOP.EQ.0) GO TO 540
C
  160 CONTINUE
      BKMIN = BTOL
      WRITE(6,170)
  170 FORMAT(/' T-MATRICES (ALPHA AND BETA) ARE NOW AVAILABLE'/)
C
C-----------------------------------------------------------------------
C     SUBROUTINE TNORM CHECKS MIN(BETA)/(ESTIMATED NORM(A)) > BTOL .
C     IF THIS IS VIOLATED IB IS SET EQUAL TO THE NEGATIVE OF THE INDEX
C     OF THE MINIMAL BETA.  IF(IB < 0) THEN SUBROUTINE TNORM IS
C     CALLED FOR EACH VALUE OF MEV TO DETERMINE WHETHER OR NOT THERE
C     IS A BETA IN THE T-MATRIX SPECIFIED THAT VIOLATES THIS TEST.
C     IF THERE IS SUCH A BETA THE PROGRAM TERMINATES FOR THE USER
C     TO DECIDE WHAT TO DO.  THIS TEST CAN BE OVER-RIDDEN BY
C     SIMPLY MAKING BTOL SMALLER, BUT THEN THERE IS THE POSSIBILITY
C     THAT LOSSES IN THE LOCAL ORTHOGONALITY MAY HURT THE COMPUTATIONS.
C     BTOL = 1.D-8 IS HOWEVER A CONSERVATIVE CHOICE FOR BTOL.
C
C     TNORM ALSO COMPUTES TKMAX = MAX(|ALPHA(K)|,BETA(K), K=1,KMAX).
C     TKMAX IS USED TO SCALE THE TOLERANCES USED IN THE
C     T-MULTIPLICITY AND SPURIOUS TESTS IN BISEC. TKMAX IS ALSO USED IN
C     THE PROJECTION TEST FOR HIDDEN EIGENVALUES THAT HAD 'TOO SMALL'
C     A PROJECTION ON THE STARTING VECTOR.
C
      CALL TNORM(ALPHA,BETA,BKMIN,TKMAX,KMAX,IB)
C
C-----------------------------------------------------------------------
C
      TTOL = EPSM*TKMAX
C
C     LOOP ON THE SIZE OF THE T-MATRIX
C
  180 CONTINUE
      MMB = MMB + 1
      MEV = NMEV(MMB)
C     IS MEV TOO LARGE ?
      IF(MEV.LE.KMAX) GO TO 200
      WRITE(6,190) MMB, MEV, KMAX
  190 FORMAT(/' TERMINATE PRIOR TO CONSIDERING THE',I6,'TH T-MATRIX'/
```

```
     1' BECAUSE THE SIZE REQUESTED',I6,' IS GREATER THAN THE MAXIMUM SIZHHL03750
     1E ALLOWED',I6/)                                                   HHL03760
      GO TO 540                                                         HHL03770
C                                                                       HHL03780
  200 MP1 = MEV + 1                                                     HHL03790
      BETAM = BETA(MP1)                                                 HHL03800
C                                                                       HHL03810
      IF (IB.GE.0) GO TO 210                                            HHL03820
C                                                                       HHL03830
      T0 = BTOL                                                         HHL03840
C                                                                       HHL03850
C-----------------------------------------------------------------------HHL03860
C                                                                       HHL03870
      CALL TNORM(ALPHA,BETA,T0,T1,MEV,IBMEV)                            HHL03880
C                                                                       HHL03890
C-----------------------------------------------------------------------HHL03900
C                                                                       HHL03910
      TEMP = T0/TKMAX                                                   HHL03920
      IBMEV = IABS(IBMEV)                                               HHL03930
      IF (TEMP.GE.BTOL) GO TO 210                                       HHL03940
      IBMEV = -IBMEV                                                    HHL03950
      GO TO 600                                                         HHL03960
C                                                                       HHL03970
  210 CONTINUE                                                          HHL03980
      IC = MXSTUR-ICT                                                   HHL03990
C                                                                       HHL04000
C-----------------------------------------------------------------------HHL04010
C     BISEC LOOP. THE SUBROUTINE BISEC INCORPORATES DIRECTLY THE        HHL04020
C     T-MULTIPLICITY AND SPURIOUS TESTS. T-EIGENVALUES WILL BE          HHL04030
C     CALCULATED BY BISEC SEQUENTIALLY ON INTERVALS                     HHL04040
C     (LB(J),UB(J)), J = 1,NINT).                                       HHL04050
C                                                                       HHL04060
C     ON RETURN FROM BISEC                                              HHL04070
C     NDIS = NUMBER OF DISTINCT EIGENVALUES OF T(1,MEV) ON UNION        HHL04080
C            OF THE (LB,UB) INTERVALS                                   HHL04090
C     VS = DISTINCT T-EIGENVALUES IN ALGEBRAICALLY INCREASING ORDER     HHL04100
C     MP = MULTIPLICITIES OF THE T-EIGENVALUES IN VS                    HHL04110
C     MP(I) = (0,1,MI), MI>1, I=1,NDIS  MEANS:                          HHL04120
C        (0)  VS(I) IS SPURIOUS                                         HHL04130
C        (1)  VS(I) IS T-SIMPLE AND GOOD                                HHL04140
C        (MI) VS(I) IS MULTIPLE AND IS THEREFORE NOT ONLY GOOD BUT      HHL04150
C             ALSO A CONVERGED GOOD T-EIGENVALUE.                       HHL04160
C     WITHIN BISEC V1 AND V2 ARE DEFINED AS DOUBLE PRECISION ARRAYS     HHL04170
C                                                                       HHL04180
C                                                                       HHL04190
      CALL BISEC(ALPHA,BETA,V1,V2,VS,LB,UB,EPSM,TTOL,MP,NINT,           HHL04200
     1 MEV,NDIS,IC,IWRITE)                                              HHL04210
C                                                                       HHL04220
C-----------------------------------------------------------------------HHL04230
C                                                                       HHL04240
      IF (NDIS.EQ.0) GO TO 620                                          HHL04250
C                                                                       HHL04260
C     COMPUTE THE TOTAL NUMBER OF STURM SEQUENCES USED TO DATE          HHL04270
C     COMPUTE THE BISEC CONVERGENCE AND T-MULTIPLICITY TOLERANCES USED. HHL04280
C     COMPUTE THE CONVERGENCE TOLERANCE FOR EIGENVALUES OF A.           HHL04290
```

```
      ICT = ICT + IC
      TEMP = DFLOAT(MEV+1000)
      MULTOL = TEMP*TTOL
      TEMP = DSQRT(TEMP)
      BISTOL = TTOL*TEMP
      CONTOL = BETAM*1.D-10
C
C-----------------------------------------------------------------------
C    SUBROUTINE LUMP 'COMBINES' T-EIGENVALUES THAT ARE 'TOO CLOSE'.
C    NOTE HOWEVER THAT CLOSE SPURIOUS T-EIGENVALUES ARE NOT AVERAGED
C    WITH GOOD ONES. HOWEVER, THEY MAY BE USED TO INCREASE THE
C    MULTIPLICITY OF A GOOD T-EIGENVALUE.
C
      LOOP = NDIS
      CALL LUMP(VS,RELTOL,MULTOL,SCALE2,MP,LOOP)
C
C-----------------------------------------------------------------------
C
      IF(NDIS.EQ.LOOP) GO TO 230
C
      WRITE(6,220) NDIS, MEV, LOOP
  220 FORMAT(/I6,' DISTINCT T-EIGENVALUES WERE COMPUTED IN BISEC AT MEV
     1',I6/ 2X,' LUMP SUBROUTINE REDUCES NUMBER OF DISTINCT EIGENVALUES
     1TO',I6)
C
  230 CONTINUE
      NDIS = LOOP
      BETA(MP1) = BETAM
C
C-----------------------------------------------------------------------
C     THE SUBROUTINE ISOEV LABELS THOSE SIMPLE EIGENVALUES OF T(1,MEV)
C     WITH VERY SMALL GAPS BETWEEN NEIGHBORING EIGENVALUES OF T(1,MEV)
C     TO AVOID COMPUTING ERROR ESTIMATES FOR ANY SIMPLE GOOD
C     T-EIGENVALUE THAT IS TOO CLOSE TO A SPURIOUS T-EIGENVALUE.
C     ON RETURN FROM ISOEV, G CONTAINS CODED MINIMAL GAPS
C     BETWEEN THE DISTINCT EIGENVALUES OF T(1,MEV). (G IS REAL).
C     G(I) < 0 MEANS MINGAP IS DUE TO LEFT GAP G(I) > 0 MEANS DUE TO
C     RIGHT GAP. MP(I) = -1 MEANS THAT THE GOOD T-EIGENVALUE IS SIMPLE
C     AND HAS A VERY SMALL MINGAP IN T(1,MEV) DUE TO A SPURIOUS
C     T-EIGENVALUE.   NG = NUMBER OF GOOD EIGENVALUES.
C     NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES.
C
      CALL ISOEV(VS,GAPTOL,MULTOL,SCALE1,G,MP,NDIS,NG,NISO)
C
C-----------------------------------------------------------------------
C
      WRITE(6,240)NG,NISO,NDIS
  240 FORMAT(/I6,' GOOD T-EIGENVALUES HAVE BEEN COMPUTED'/
     1 I6,' OF THESE ARE T-ISOLATED'/
     2 I6,' = NUMBER OF DISTINCT T-EIGENVALUES COMPUTED'/)
C
C     DO WE WRITE DISTINCT EIGENVALUES OF T-MATRIX TO FILE 4?
      IF (IDIST.EQ.0) GO TO 280
C
      WRITE(11,250) NDIS,NISO,MEV,N,SVSEED,MATNO
```

```
  250 FORMAT(/4I6,I12,I8,' = NDIS,NISO,MEV,N,SVSEED,MATNO'/)
C
      WRITE(11,260) (MP(I),VS(I),G(I), I=1,NDIS)
  260 FORMAT(2(I3,E25.16,E12.3))
C
      WRITE(11,270) NDIS, (MP(I), I=1,NDIS)
  270 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/(20I4))
C
  280 CONTINUE
C
      IF (NISO.NE.0) GO TO 310
C
      WRITE(4,290) MEV
  290 FORMAT(/' AT MEV = ',I6,' THERE ARE NO ISOLATED T-EIGENVALUES'/
     1' SO NO ERROR ESTIMATES WERE COMPUTED/')
C
      WRITE(6,300)
  300 FORMAT(/' ALL COMPUTED GOOD T-EIGENVALUES ARE MULTIPLE'/
     1 ' THEREFORE ALL SUCH EIGENVALUES ARE ASSUMED TO HAVE CONVERGED')
C
      ICONV = 1
      GO TO 350
C
  310 CONTINUE
C
C-----------------------------------------------------------------------
C     SUBROUTINE INVERR COMPUTES ERROR ESTIMATES FOR ISOLATED GOOD
C     T-EIGENVALUES USING INVERSE ITERATION ON T(1,MEV). ON RETURN
C     G(J) = MINIMUM GAP IN T(1,MEV) FOR EACH VS(J), J=1,NDIS
C     G(MEV+I) = BETAM*|U(MEV)| = ERROR ESTIMATE FOR ISOLATED GOOD
C              T-EIGENVALUES, WHERE I = 1, NISO AND  BETAM = BETA(MEV+1)
C              U(MEV) IS MEVTH COMPONENT OF THE UNIT EIGENVECTOR OF T
C              CORRESPONDING TO THE ITH ISOLATED GOOD T-EIGENVALUE.
C     A NEGATIVE ERROR ESTIMATE MEANS THAT FOR THAT PARTICULAR
C     EIGENVALUE THE INVERSE ITERATION DID NOT CONVERGE IN <= MXINIT
C     STEPS AND THAT THE CORRESPONDING ERROR ESTIMATE IS QUESTIONABLE.
C
C     V2 CONTAINS THE ISOLATED GOOD T-EIGENVALUES
C     V1 CONTAINS THE MINGAPS TO THE NEAREST DISTINCT  EIGENVALUE
C        OF T(1,MEV) FOR EACH ISOLATED GOOD EIGENVALUE IN V2.
C     VS CONTAINS THE NDIS DISTINCT EIGENVALUES OF T(1,MEV)
C     MP CONTAINS THE CORRESPONDING CODED T-MULTIPLICITIES
C     WITHIN INVERR V1 AND V2 ARE DOUBLE PRECISION ARRAYS
C
      IT = MXINIT
      CALL INVERR(ALPHA,BETA,V1,V2,VS,EPSM,G,MP,MEV,MMB,NDIS,NISO,N,
     1 RHSEED,IT,IWRITE)
C
C-----------------------------------------------------------------------
C
C     SIMPLE CHECK FOR CONVERGENCE. CHECKS TO SEE IF ALL OF THE ERROR
C     ESTIMATES ARE SMALLER THAN CONTOL.
C     IF THIS TEST IS SATISFIED, THEN CONVERGENCE FLAG, ICONV IS SET
C     TO 1.  TYPICALLY ERROR ESTIMATES ARE VERY CONSERVATIVE.
C
```

```
      WRITE(6,320) CONTOL
  320 FORMAT(/' CONVERGENCE IS TESTED USING THE CONVERGENCE TOLERANCE',
     1E13.4/)
C
      II = MEV +1
      IF = MEV+NISO
      DO 330 I = II,IF
      IF (ABS(G(I)).GT.CONTOL) GO TO 350
  330 CONTINUE
      ICONV = 1
      MMB = NMEVS
C
      WRITE(6,340) CONTOL
  340 FORMAT(' ALL COMPUTED ERROR ESTIMATES WERE LESS THAN',E15.4/
     1 ' THEREFORE PROCEDURE TERMINATES'/)
C
  350 CONTINUE
C
C     IF CONVERGENCE IS INDICATED, THAT IS ICONV = 1 ,THEN
C     THE SUBROUTINE PRTEST IS CALLED TO CHECK FOR ANY CONVERGED
C     EIGENVALUES THAT HAVE BEEN MISLABELLED AS SPURIOUS BECAUSE
C     THE PROJECTION OF THEIR EIGENVECTOR(S) ON THE STARTING
C     VECTOR WERE TOO SMALL.
C     NUMERICAL TESTS INDICATE THAT SUCH EIGENVALUES ARE RARE.
C     IF FOR SOME REASON MANY OF THESE HIDDEN EIGENVALUES APPEAR
C     ON SOME RUN, YOU CAN BE CERTAIN THAT SOMETHING IS FOULED UP.
C
      IF (ICONV.EQ.0) GO TO 480
C
C-----------------------------------------------------------------------
C
      CALL PRTEST (ALPHA,BETA,VS,TKMAX,EPSM,RELTOL,SCALE3,SCALE4,
     1 MP,NDIS,MEV,IPROJ)
C
C-----------------------------------------------------------------------
C
      IF(IPROJ.EQ.0) GO TO 470
C
      IF(IDIST.EQ.1)  WRITE(11,360) IPROJ
  360 FORMAT(' SUBROUTINE PRTEST WANTS TO RELABEL',I6,' SPURIOUS EIGENVA
     1LUES'/' WE ACCEPT RELABELLING ONLY IF LAST COMPONENT OF T-EIGENVEC
     1TOR IS L.T. 1.D-10'/)
C
      IIX = RHSEED
C
C-----------------------------------------------------------------------
C
      CALL GENRAN(IIX,G,MEV)
C
C-----------------------------------------------------------------------
C
      ITEN = -10
      NISOM = NISO + MEV
      IWRITO = IWRITE
      IWRITE = 0
```

```
C
      DO 390 J = 1,NDIS
      IF(MP(J).NE.ITEN) GO TO 390
      T0 = VS(J)
C
C-----------------------------------------------------------------------
C
      IT = MXINIT
      CALL INVERM(ALPHA,BETA,V1,V2,T0,TEMP,T1,EPSM,G,MEV,IT,IWRITE)
C
C-----------------------------------------------------------------------
C
      IF(TEMP.LE.1.D-10) GO TO 380
C     ERROR ESTIMATE WAS NOT SMALL REJECT RELABELLING OF THIS EIGENVALUE
      IF(IDIST.EQ.1)  WRITE(11,370) J,T0,TEMP
  370 FORMAT(/' LAST COMPONENT FOR',I6,'TH EIGENVALUE',E20.12/' IS TOO L
     1ARGE = ',E15.6,' SO DO NOT ACCEPT PRTEST RELABELLING'/)
      MP(J) = 0
      IPROJ = IPROJ - 1
      GO TO 390
C     RELABELLING ACCEPTED
  380 NISOM = NISOM + 1
      G(NISOM) = BETAM*TEMP
  390 CONTINUE
      IWRITE = IWRITO
C
      IF(IPROJ.EQ.0) GO TO 430
      WRITE(6,400) IPROJ
  400 FORMAT(/I6,' T-EIGENVALUES WERE RECLASSIFIED AS GOOD.'/
     1' THESE ARE IDENTIFIED IN FILE 3 BY A T-MULTIPLICITY OF -10'/' USE
     2R SHOULD INSPECT EACH TO MAKE SURE NEIGHBORS HAVE CONVERGED'/)
C
      IF(IDIST.EQ.1)  WRITE(11,410) IPROJ
  410 FORMAT(/I6,' T-EIGENVALUES WERE RELABELLED AS GOOD'/
     1' BELOW IS CORRECTED T-MULTIPLICITY PATTERN'/)
C
      WRITE(6,420) NDIS, (MP(I), I=1,NDIS)
      IF(IDIST.EQ.1)  WRITE(11,420) NDIS, (MP(I), I=1,NDIS)
  420 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/
     1 6X, ' (-10) MEANS SPURIOUS T-EIGENVALUE RELABELLED AS GOOD'/(20I4
     1))
C
C     RECALCULATE MINGAPS FOR DISTINCT T(1,MEV) EIGENVALUES.
  430 NM1 = NDIS - 1
      G(NDIS) = VS(NM1)-VS(NDIS)
      G(1) = VS(2)-VS(1)
C
      DO 440 J = 2,NM1
      T0 = VS(J)-VS(J-1)
      T1 = VS(J+1)-VS(J)
      G(J) = T1
      IF (T0.LT.T1) G(J) = -T0
  440 CONTINUE
      IF(IPROJ.EQ.0) GO TO 470
C     WRITE TO FILE 4 ERROR ESTIMATES FOR THOSE T-EIGENVALUES RELABELLED
```

```
      NGOOD = 0                                                         HHL06500
      DO 450 J = 1,NDIS                                                 HHL06510
      IF(MP(J).EQ.0) GO TO 450                                          HHL06520
      NGOOD = NGOOD + 1                                                 HHL06530
      IF(MP(J).NE.ITEN) GO TO 450                                       HHL06540
      T0 = VS(J)                                                        HHL06550
      NISO = NISO + 1                                                   HHL06560
      NISOM = MEV + NISO                                                HHL06570
      WRITE(4,460) NGOOD,T0,G(NISOM),G(J)                               HHL06580
  450 CONTINUE                                                          HHL06590
  460 FORMAT(I10,E25.16,2E14.3)                                         HHL06600
C                                                                       HHL06610
  470 CONTINUE                                                          HHL06620
C                                                                       HHL06630
C     WRITE THE GOOD T-EIGENVALUES TO FILE 3.  FIRST TRANSFER THEM      HHL06640
C     TO V2 AND THEIR T-MULTIPLICITIES TO THE CORRESPONDING POSITIONS   HHL06650
C     IN MP AND COMPUTE THE A-MINGAPS, THE MINIMAL GAPS BETWEEN THE     HHL06660
C     GOOD T-EIGENVALUES.  THESE GAPS WILL BE PUT IN THE ARRAY G.       HHL06670
C     SINCE G CURRENTLY CONTAINS THE MINIMAL GAPS BETWEEN THE DISTINCT  HHL06680
C     EIGENVALUES OF THE T-MATRIX, THESE GAPS WILL FIRST BE             HHL06690
C     TRANSFERRED TO GC.  NOTE THAT GC<0 MEANS THAT THAT MINIMAL GAP    HHL06700
C     IN THE T-MATRIX IS DUE TO A SPURIOUS T-EIGENVALUE.                HHL06710
C     ALL THIS INFORMATION IS PRINTED TO FILE 3                         HHL06720
C                                                                       HHL06730
  480 CONTINUE                                                          HHL06740
C                                                                       HHL06750
      NG = 0                                                            HHL06760
      DO 490 I = 1,NDIS                                                 HHL06770
      IF (MP(I).EQ.0) GO TO 490                                         HHL06780
      NG = NG+1                                                         HHL06790
      MP(NG) = MP(I)                                                    HHL06800
      GR(NG) = VS(I)                                                    HHL06810
      TEMP = G(I)                                                       HHL06820
      TEMP = DABS(TEMP)                                                 HHL06830
      J = I+1                                                           HHL06840
      IF (G(I).LT.ZERO) J = I-1                                         HHL06850
      IF (MP(J).EQ.0) TEMP = -TEMP                                      HHL06860
      GC(NG) = TEMP                                                     HHL06870
  490 CONTINUE                                                          HHL06880
C                                                                       HHL06890
      WRITE(6,500)MEV                                                   HHL06900
  500 FORMAT(//' T-EIGENVALUE CALCULATION AT MEV = ',I6,'    IS COMPLETEHHL06910
     1')                                                                HHL06920
C                                                                       HHL06930
C     NG = NUMBER OF COMPUTED DISTINCT GOOD T-EIGENVALUES.  NEXT        HHL06940
C     GENERATE GAPS BETWEEN GOOD T-EIGENVALUES (AMINGAPS) AND PUT THEM  HHL06950
C     IN G.  G(J) < 0 MEANS THE AMINGAP IS DUE TO THE LEFT-HAND GAP.    HHL06960
C                                                                       HHL06970
      NGM1 = NG - 1                                                     HHL06980
      G(NG) = GR(NGM1)-GR(NG)                                           HHL06990
      G(1) = GR(2)-GR(1)                                                HHL07000
C                                                                       HHL07010
      DO 510 J = 2,NGM1                                                 HHL07020
      T0 = GR(J)-GR(J-1)                                                HHL07030
      T1 = GR(J+1)-GR(J)                                                HHL07040
```

```
      G(J) = T1                                                         HHL07050
      IF (T0.LT.T1) G(J) = -T0                                          HHL07060
  510 CONTINUE                                                          HHL07070
C                                                                       HHL07080
C     WRITE GOOD T-EIGENVALUES OUT TO FILE 3.                           HHL07090
C                                                                       HHL07100
      WRITE(3,520)NG,NDIS,MEV,N,SVSEED,MATNO,MULTOL,IB,BTOL             HHL07110
  520 FORMAT(4I6,I12,I8,' = NG,NDIS,MEV,N,SVEED,MATNO'/                 HHL07120
     1 E20.12,I6,E13.4,' = MUTOL,INDEX MINIMAL BETA,BTOL'/              HHL07130
     1' EV NO',2X,'TMULT',7X,'GOOD T-EIGENVALUE',7X,'TMINGAP',7X,'AMINGAHHL07140
     1P')                                                               HHL07150
C                                                                       HHL07160
      WRITE(3,530)(I,MP(I),GR(I),GC(I),G(I), I=1,NG)                    HHL07170
  530 FORMAT(2I6,E25.16,2E14.3)                                         HHL07180
C                                                                       HHL07190
C     IF CONVERGENCE FLAG ICONV.NE.1 AND NUMBER OF T-MATRICES           HHL07200
C     CONSIDERED TO DATE IS LESS THAN NUMBER ALLOWED, INCREMENT MEV.    HHL07210
C     AND LOOP BACK TO 210 TO REPEAT COMPUTATIONS.  RESTORE BETA(MEV+1).HHL07220
C                                                                       HHL07230
      BETA(MP1) = BETAM                                                 HHL07240
C                                                                       HHL07250
      IF (MMB.LT.NMEVS.AND.ICONV.NE.1) GO TO 180                        HHL07260
C                                                                       HHL07270
C     END OF LOOP ON DIFFERENT SIZE T-MATRICES ALLOWED.                 HHL07280
C                                                                       HHL07290
  540 CONTINUE                                                          HHL07300
C                                                                       HHL07310
      IF(ISTOP.EQ.0)  WRITE(6,550)                                      HHL07320
  550 FORMAT(/' T-MATRICES (ALPHA AND BETA) ARE NOW AVAILABLE, TERMINATEHHL07330
     1')                                                                HHL07340
      IF (IHIS.EQ.1.AND.KMAX.NE.MOLD) WRITE(1,560)                      HHL07350
  560 FORMAT(/' ABOVE ARE THE FOLLOWING VECTORS '/                      HHL07360
     1 '  ALPHA(I), I = 1,KMAX'/                                        HHL07370
     2 '  BETA(I), I = 1,KMAX+1'/                                       HHL07380
     3 ' FINAL TWO LANCZOS VECTORS OF ORDER N FOR I = KMAX,KMAX+1'/     HHL07390
     4 ' ALL ENTRIES IN THIS FILE HAVE FORMAT 4Z20 '/                   HHL07400
     5 ' ----- END OF FILE 1 NEW ALPHA, BETA HISTORY---------------'///)HHL07410
C                                                                       HHL07420
      IF (ISTOP.EQ.0) GO TO 640                                         HHL07430
C                                                                       HHL07440
      WRITE(3,570)                                                      HHL07450
  570 FORMAT(/' ABOVE ARE COMPUTED GOOD T-EIGENVALUES'/                 HHL07460
     1 ' NG = NUMBER OF GOOD T-EIGENVALUES COMPUTED'/                   HHL07470
     2 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)'/   HHL07480
     3 ' N = ORDER OF A,  MATNO = MATRIX IDENT'/                        HHL07490
     4 ' MULTOL = MULTIPLICITY TOLERANCE FOR T-EIGENVALUES IN BISEC'/   HHL07500
     4 ' TMULT IS THE T-MULTIPLICITY OF GOOD T-EIGENVALUE'/             HHL07510
     5 ' TMULT = -1 MEANS SPURIOUS T-EIGENVALUE TOO CLOSE'/             HHL07520
     6 ' DO NOT COMPUTE ERROR ESTIMATES FOR SUCH EIGENVALUES'/          HHL07530
     7 ' AMINGAP = MINIMAL GAP BETWEEN THE COMPUTED A-EIGENVALUES'/     HHL07540
     8 ' AMINGAP .LT. 0. MEANS MINIMAL GAP IS DUE TO LEFT-HAND GAP'/    HHL07550
     9 ' TMINGAP= MINIMAL GAP W.R.T.  DISTINCT EIGENVALUES IN T(1,MEV)'/HHL07560
     1 ' TMINGAP .LT. 0. MEANS MINGAP IS DUE TO SPURIOUS EIGENVALUE'/   HHL07570
     2 ' ----- END OF FILE 3 GOODEIGENVALUES-----------------------'///)HHL07580
C                                                                       HHL07590
```

```
      IF (IDIST.EQ.1) WRITE(11,580)                                     HHL07600
  580 FORMAT(/' ABOVE ARE THE DISTINCT EIGENVALUES OF T(1,MEV).'/       HHL07610
     2 ' THE FORMAT IS      T-MULTIPLICITY    EIGENVALUE   TMINGAP'/    HHL07620
     3 '        THIS FORMAT IS REPEATED TWICE ON EACH LINE.'/           HHL07630
     4 ' T-MULTIPLICITY = -1 MEANS THAT THE SUBROUTINE ISOEV HAS TAGGED'HHL07640
     5 /'   THIS SIMPLE T-EIGENVALUE AS HAVING A VERY CLOSE SPURIOUS'/  HHL07650
     6 '   T-EIGENVALUE SO THAT NO ERROR ESTIMATE WILL BE COMPUTED'/    HHL07660
     7 '   FOR THAT EIGENVALUE IN SUBROUTINE INVERR.'/                  HHL07670
     8 ' TMINGAP .LT. 0, TMINGAP IS DUE TO LEFT GAP .GT. 0, RIGHT GAP.'/HHL07680
     9 ' EACH OF THE DISTINCT T-EIGENVALUE TABLES IS FOLLOWED'/         HHL07690
     9 ' BY THE T-MULTIPLICITY PATTERN.'/                               HHL07700
     1 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV).'/  HHL07710
     2 ' NG = NUMBER OF GOOD T-EIGENVALUES. '/                          HHL07720
     3 ' NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES. '/               HHL07730
     4 ' NISO ALSO IS THE COUNT OF +1 ENTRIES IN T-MULTIPLICITY PATTERN.HHL07740
     5 '/' ----- END OF FILE 4 DISTINCT T-EIGENVALUES----------------'//HHL07750
     6 /)                                                               HHL07760
C                                                                       HHL07770
      IF(NISO.NE.0)  WRITE(4,590)                                       HHL07780
  590 FORMAT(/' ABOVE ARE THE ERROR ESTIMATES OBTAINED FOR THE ISOLATED HHL07790
     1GOOD EIGENVALUES'/                                                HHL07800
     1' OBTAINED VIA INVERSE ITERATION IN THE SUBROUTINE INVERR.'/      HHL07810
     1' ALL OTHER GOOD EIGENVALUES HAVE CONVERGED.'/                    HHL07820
     2' ERROR ESTIMATE = BETAM*ABS(UM)'/                                HHL07830
     2' WHERE BETAM = BETA(MEV+1) AND UM = U(MEV).'/                    HHL07840
     3' U = UNIT EIGENVECTOR OF T WHERE T*U = EV*U AND EV = ISOLATED GOOHHL07850
     3D EIGENVALUE.'/                                                   HHL07860
     4' TMINGAP = GAP TO NEAREST DISTINCT EIGENVALUE OF T(1,MEV).'/     HHL07870
     5' TMINGAP .LT. 0. MEANS MINGAP IS DUE TO LEFT NEIGHBOR.'/         HHL07880
     6' ERROR ESTIMATE L.T. 0 MEANS INVERSE ITERATION DID NOT CONVERGE'/HHL07890
     7' ------ END OF FILE 7 ERRINV -------------------------------'//) HHL07900
      GO TO 640                                                         HHL07910
C                                                                       HHL07920
  600 CONTINUE                                                          HHL07930
C                                                                       HHL07940
      IBB = IABS(IBMEV)                                                 HHL07950
      IF (IBMEV.LT.0) WRITE(6,610) MEV,IBB,BETA(IBB)                    HHL07960
  610 FORMAT(/' PROGRAM TERMINATES BECAUSE MEV REQUESTED = ',I6,' IS .GTHHL07970
     1',I6/' AT WHICH AN ABNORMALLY SMALL BETA = ' , E13.4,' OCCURRED'/)HHL07980
      GO TO 640                                                         HHL07990
C                                                                       HHL08000
  620 IF (NDIS.EQ.0.AND.ISTOP.GT.0) WRITE(6,630)                        HHL08010
  630 FORMAT(/' INTERVALS SPECIFIED FOR BISECT DID NOT CONTAIN ANY EIGENHHL08020
     1VALUES'/' PROGRAM TERMINATES')                                    HHL08030
C                                                                       HHL08040
  640 CONTINUE                                                          HHL08050
C                                                                       HHL08060
      STOP                                                              HHL08070
C-----END OF MAIN PROGRAM FOR LANCZOS HERMITIAN EIGENVALUE COMPUTATIONS-HHL08080
      END                                                               HHL08090
```

3.3 HLEVEC: Main Program, Eigenvector Computations

```
C     MODIFIED 8/16/83 ( and 4/27/93 to change array dimensioning)      HHL00010
C                                                                       HHL00020
C-----HLEVEC (EIGENVECTORS OF HERMITIAN MATRICES)-----------------------HHL00030
C                                                                       HHL00040
C  Authors:  Jane Cullum and Ralph A. Willoughby (deceased)             HHL00050
C            Los Alamos National Laboratory                             HHL00060
C            Los Alamos, New Mexico 87544                               HHL00070
C            E-Mail: cullumj@lanl.gov                                   HHL00075
c                                                                       HHL00080
c  These codes are copyrighted by the authors.  These codes             HHL00090
c  and modifications of them or portions of them are NOT to be          HHL00100
c  incorporated into any commercial codes without legal agreements      HHL00110
c  with the authors.  If these codes or portions of them                HHL00120
c  are used in other scientific or engineering research works           HHL00130
c  the names of the authors of these codes and appropriate              HHL00140
c  references to their written work are to be incorporated in the      HHL00150
c  derivative works.                                                    HHL00160
c                                                                       HHL00170
c  This header is not to be removed from these codes.                   HHL00180
C                                                                       HHL00181
C        REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4             HHL00182
C        Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsHHL00183
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in    HHL00184
C        Applied Mathematics, 2002. SIAM Publications,                  HHL00185
C        Philadelphia, PA. USA                                          HHL00186
C                                                                       HHL00187
c                                                                       HHL00190
C     CONTAINS MAIN PROGRAM FOR COMPUTING AN EIGENVECTOR CORRESPONDING  HHL00200
C     TO EACH OF A SET OF EIGENVALUES THAT HAVE BEEN COMPUTED           HHL00210
C     ACCURATELY BY THE CORRESPONDING LANCZOS EIGENVALUE PROGRAM        HHL00220
C     (HLEVAL) FOR HERMITIAN MATRICES.  THIS PROGRAM COULD BE           HHL00230
C     MODIFIED TO COMPUTE ADDITIONAL EIGENVECTORS FOR ANY               HHL00240
C     MULTIPLE EIGENVALUE OF THE GIVEN A-MATRIX.  THE AMOUNT OF         HHL00250
C     ADDITIONAL COMPUTATION REQUIRED BY SUCH A MODIFICATION WOULD      HHL00260
C     DEPEND UPON THE GIVEN MATRIX AND UPON WHICH PART OF THE           HHL00270
C     SPECTRUM WAS INVOLVED.                                            HHL00280
C                                                                       HHL00290
C     THE LANCZOS EIGENVECTOR COMPUTATIONS ASSUME THAT EACH             HHL00300
C     EIGENVALUE THAT IS BEING CONSIDERED HAS CONVERGED AS AN           HHL00310
C     EIGENVALUE OF THE LANCZOS TRIDIAGONAL MATRICES.                   HHL00320
C                                                                       HHL00330
C     PORTABILITY:                                                      HHL00340
C     THIS PROGRAM IS NOT PORTABLE BECAUSE OF THE USE OF COMPLEX*16     HHL00350
C     VARIABLES.  MOREOVER, THE PFORT VERIFIER IDENTIFIED THE           HHL00360
C     FOLLOWING ADDITIONAL NONPORTABLE CONSTRUCTIONS:                    HHL00370
C                                                                       HHL00380
C     1.  DATA/MACHEP/ STATEMENT                                        HHL00390
C     2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                        HHL00400
C     3.  FORMAT(20A4) USED WITH THE EXPLANATORY HEADER, EXPLAN         HHL00410
C     4.  HEXADECIMAL FORMAT (4Z20) USED IN ALPHA/BETA FILES 1 AND 2.   HHL00420
C                                                                       HHL00430
```

```
C     IMPORTANT NOTE:  PROGRAM ALLOWS ENLARGEMENT OF THE ALPHA, BETA    HHL00440
C     ARRAYS.  IN PARTICULAR, IF ANY ONE OF THE EIGENVALUES SUPPLIED    HHL00450
C     IS T-SIMPLE AND NOT CLOSE TO A SPURIOUS T-EIGENVALUE, THE PROGRAM HHL00460
C     REQUIRES THAT KMAX BE AT LEAST 11*MEV/8 + 12.  IF KMAX IS NOT     HHL00470
C     THIS LARGE, THEN THE PROGRAM WILL RESET KMAX TO THIS SIZE         HHL00480
C     AND EXTEND THE ALPHA, BETA HISTORY IF REQUIRED.                   HHL00490
C     THUS, THE DIMENSIONS OF THE ALPHA AND BETA ARRAYS MUST BE         HHL00500
C     LARGE ENOUGH TO ALLOW FOR THIS POSSIBILITY.                       HHL00510
C     REMEMBER THAT THE BETA ARRAY, BETA(J), IS SUCH THAT               HHL00520
C     J = 1,..., KMAX+1.  SO IF THE KMAX USED BY THE PROGRAM            HHL00530
C     IS TO BE 3000, THEN BETA MUST BE OF LENGTH AT LEAST 3001.         HHL00540
C                                                                       HHL00550
C     TO AVOID USING MAX(I,J) IN THE PARAMETER LISTING WE HAVE USED     HHL00560
C     THE FOLLOWING EQUIVALENT RELATIONSHIP                             HHL00570
C                                                                       HHL00580
C     MAX(I,J) = ( 2*I/(I+J))*I  + (2*J/(I+J))*J                        HHL00590
C                                                                       HHL00600
      parameter (n=625,mev=1500,ngood= 3,nngood=n*ngood)                HHL00610
      parameter (kmaxn = (3*mev)/2 + 12, kmaxn1=kmaxn+1)                HHL00620
      parameter(nkmaxn = ngood*kmaxn)                                   HHL00630
      PARAMETER ( KMAXn1 = KMAXn+1, KMAXn12 = KMAXn1/2 )                HHL00640
      PARAMETER ( NKMAXn2 = N+KMAXn12, NPKMAXn = N+KMAXn)               HHL00650
      PARAMETER (KVS = ((2*N)/NPKMAXn)*N + ((2*KMAXn)/NPKMAXn)*KMAXn )  HHL00660
      PARAMETER (KV2 = ((2*N)/NKMAXn2)*N +((2*KMAXn12)/NKMAXn2)*KMAXn12)HHL00670
C-----------------------------------------------------------------------HHL00680
      COMPLEX*16  V1(kv2),V2(n),VS(n),RITVEC(nngood),ZEROC,TEMPC        HHL00690
      DOUBLE PRECISION  ALPHA(kmaxn),BETA(kmaxn1),GR(n),GC(n)           HHL00700
      DOUBLE PRECISION  TVEC(nkmaxm),GOODEV(ngood),EVNEW(ngood)         HHL00710
      DOUBLE PRECISION  EVAL,EVALN,TOLN,TTOL,ERTOL,ALFA,BATA            HHL00720
      DOUBLE PRECISION  MULTOL,SCALE0,STUTOL,BTOL,LB,UB                 HHL00730
      DOUBLE PRECISION  ONE,ZERO,MACHEP,EPSM,TEMP,SUM,ERRMIN,BKMIN      HHL00740
      DOUBLE PRECISION  RELTOL,ERROR,TERROR,TLAST(ngood)                HHL00750
      REAL  G(kvs),AMINGP(ngood),TMINGP(ngood),EXPLAN(20)               HHL00760
      REAL  TERR(ngood),ERR(ngood),ERRDGP(ngood),RNORM(ngood)           HHL00770
      real TBETA(ngood)                                                 HHL00780
      INTEGER  MP(ngood),M1(ngood),M2(ngood),MA(ngood)                  HHL00790
      integer ML(ngood),MINT(ngood),MFIN(ngood)                         HHL00800
      INTEGER  SVSEED,SVSOLD,RHSEED,IDELTA(ngood),MULEVA(ngood)         HHL00810
      INTEGER  MBOUND,NTVCON,SVTVEC,TVSTOP,LVCONT,ERCONT,TFLAG          HHL00820
      DOUBLE PRECISION  DABS, DMAX1, DSQRT, DFLOAT                      HHL00830
      REAL ABS                                                          HHL00840
      INTEGER  IABS                                                     HHL00850
C-----------------------------------------------------------------------HHL00860
      EXTERNAL CMATV                                                    HHL00870
      DATA MACHEP/Z3410000000000000/                                    HHL00880
      EPSM = 2.D0*MACHEP                                                HHL00890
C-----------------------------------------------------------------------HHL00900
C                                                                       HHL00910
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:                            HHL00920
C     1.  ALPHA:  >= KMAXN,  BETA: >= (KMAXN+1) WHERE KMAXN, THE        HHL00930
C               LARGEST SIZE T-MATRIX CONSIDERED BY THE PROGRAM,        HHL00940
C               IS THE LARGER OF THE SIZE OF THE ALPHA, BETA HISTORY    HHL00950
C               PROVIDED ON FILE 2 (IF ANY ) AND THE SIZE WHICH THE     HHL00960
C               PROGRAM SPECIFIES INTERNALLY, THIS LATTER IS ALWAYS     HHL00970
C               < = 11*MEV / 8  +  12, WHERE MEV IS THE SIZE            HHL00980
```

```
C                T-MATRIX THAT WAS USED IN THE CORRESPONDING EIGENVALUE
C                COMPUTATIONS.
C     2.  V1:  >= MAX(N,KMAX/2)
C     3.  V2, VS:   >= N
C     4.  G:  >= MAX(N,KMAX).   GR, GC:  >= N
C     5.  RITVEC:  >= N*NGOOD, WHERE NGOOD IS NUMBER OF EIGENVALUES
C                  SUPPLIED TO THIS PROGRAM.
C     6.  TVEC:  >= CUMULATIVE LENGTH OF ALL THE T-EIGENVECTORS
C         NEEDED TO GENERATE THE DESIRED RITZ VECTORS.  AN EDUCATED
C         GUESS AT AN APPROPRIATE LENGTH CAN BE OBTAINED BY RUNNING THE
C         PROGRAM WITH THE FLAG MBOUND = 1 AND MULTIPLYING THE
C         RESULTING SIZE BY 5/4.
C     7.  GOODEV, EVNEW, AMINGP, TMINGP, TERR, ERR, ERRGDP, RNORM, TBETA
C         TLAST, MP, MA, M1, M2, MINT, MFIN, MULEVA, AND IDELTA ALL
C         MUST BE AT LEAST NGOOD.
C
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LANCZOS EIGENVECTOR PROCEDURE FOR HERMITIAN MATRICES'/)
C
C     SET PROGRAM PARAMETERS
C     USER MUST NOT MODIFY SCALE0
      SCALE0 = 5.0D0
      ZERO = 0.0D0
      ZEROC = DCMPLX(ZERO,ZERO)
      ONE = 1.0D0
      MPMIN = -1000
C     CONVERGENCE TOLERANCE FOR T-EIGENVECTORS FOR RITZ VECTORS
      ERTOL = 1.D-10
      ISREAL = 0
C
C     READ USER-SPECIFIED PARAMETER FROM INPUT FILE 5 (FREE FORMAT)
C
C     READ USER-PROVIDED HEADER FOR RUN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C     READ IN THE MAXIMUM PERMISSIBLE DIMENSIONS FOR THE TVEC ARRAY
C     (MDIMTV), FOR THE RITVEC ARRAY (MDIMRV), AND FOR THE BETA
C     ARRAY (MBETA).
C
      READ(5,20) EXPLAN
      READ(5,*) MDIMTV, MDIMRV, MBETA
C
C     READ IN RELATIVE TOLERANCE (RELTOL) USED IN DETERMINING
C     APPROPRIATE SIZES FOR THE T-MATRICES USED IN THE EIGENVECTOR
C     COMPUTATIONS
C
      READ(5,20) EXPLAN
      READ(5,*) RELTOL
C
C
C     SET FLAGS TO 0 OR 1:
```

```
C     MBOUND = 1:  PROGRAM TERMINATES AFTER COMPUTING 1ST GUESSES
C                  ON APPROPRIATE T-SIZES FOR USE IN THE RITZ VECTOR
C                  COMPUTATIONS
C     NTVCON = 0:  PROGRAM TERMINATES IF THE TVEC ARRAY IS NOT
C                  LARGE ENOUGH TO HOLD ALL THE T-EIGENVECTORS REQUIRED.
C     SVTVEC = 0:  THE T-EIGENVECTORS ARE NOT WRITTEN TO FILE 11
C                  UNLESS TVSTOP = 1
C     SVTVEC = 1:  WRITE THE T-EIGENVECTORS TO FILE 11.
C     TVSTOP = 1:  PROGRAM TERMINATES AFTER COMPUTING THE
C                  T-EIGENVECTORS
C     LVCONT = 0:  PROGRAM TERMINATES IF THE NUMBER OF T-EIGENVECTORS
C                  COMPUTED IS NOT EQUAL TO THE NUMBER OF RITZ
C                  VECTORS REQUESTED.
C     ERCONT = 0:  MEANS FOR ANY GIVEN EIGENVALUE, A RITZ VECTOR
C                  WILL NOT BE COMPUTED FOR THAT EIGENVALUE UNLESS
C                  A T-EIGENVECTOR HAS BEEN IDENTIFIED WITH A LAST
C                  COMPONENT WHICH SATISFIES THE SPECIFIED
C                  CONVERGENCE CRITERION.
C     ERCONT = 1:  MEANS FOR ANY GIVEN EIGENVALUE, A RITZ VECTOR
C                  WILL BE COMPUTED.  IF A T-EIGENVECTOR CANNOT
C                  BE IDENTIFIED WHICH SATISFIES THE LAST
C                  COMPONENT CRITERION, THEN THE PROGRAM WILL
C                  USE THE T-VECTOR THAT CAME CLOSEST TO
C                  SATISFYING THE CRITERION
C     IWRITE = 1:  EXTENDED OUTPUT OF INTERMEDIATE COMPUTATIONS
C                  IS WRITTEN TO FILE 6
C     IREAD = 0:   ALPHA/BETA FILE IS REGENERATED
C     IREAD = 1:   ALPHA/BETA FILE USED IN EIGENVALUE COMPUTATIONS
C                  IS READ IN AND EXTENDED IF NECESSARY.  IN BOTH
C                  CASES IREAD = 0 OR 1, THE LANCZOS VECTORS ARE
C                  ALWAYS REGENERATED FOR THE RITZ VECTOR
C                  COMPUTATIONS
C
      READ(5,20) EXPLAN
      READ(5,*) MBOUND,NTVCON,SVTVEC,IREAD
C
      READ(5,20) EXPLAN
      READ(5,*) TVSTOP,LVCONT,ERCONT,IWRITE
      IF (TVSTOP.EQ.1) SVTVEC = 1
C
C     READ IN SEED FOR GENERATING RANDOM STARTING VECTOR FOR THE
C     INVERSE ITERATION ON THE T-MATRICES.
C
      READ(5,20) EXPLAN
      READ(5,*) RHSEED
C
C     READ IN MATNO = MATRIX/RUN IDENTIFICATION NUMBER AND
C     N = ORDER OF A-MATRIX
C
      READ(5,20) EXPLAN
      READ(5,*) MATNO,N
C     IF MATNO < 0, THEN MATRIX SUPPLIED IS REAL AND USER WANTS TO
C     CHECK ON THE T-MULTIPLICITY OF THE EIGENVALUES OF GIVEN MATRIX
      IF(MATNO.GT.0) GO TO 30
      ISREAL = 1
```

```
      MATNO = - MATNO                                                   HHL02090
   30 CONTINUE                                                          HHL02100
C                                                                       HHL02110
C-----------------------------------------------------------------------HHL02120
C     INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED MATRIX               HHL02130
C     AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE             HHL02140
C     MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV.                          HHL02150
C                                                                       HHL02160
      CALL USPEC(N,MATNO)                                               HHL02170
C                                                                       HHL02180
C-----------------------------------------------------------------------HHL02190
C     MASK UNDERFLOW AND OVERFLOW                                       HHL02200
      CALL MASK                                                         HHL02210
C                                                                       HHL02220
C-----------------------------------------------------------------------HHL02230
C                                                                       HHL02240
C     WRITE RUN PARAMETERS OUT TO FILE 6                                HHL02250
C                                                                       HHL02260
      WRITE(6,40) MATNO,N                                               HHL02270
   40 FORMAT(/' MATRIX IDENTIFICATION NO. = ',I10,' ORDER OF A = ',I5)  HHL02280
C                                                                       HHL02290
      WRITE(6,50) MBOUND,NTVCON,SVTVEC,IREAD                            HHL02300
   50 FORMAT(/3X,'MBOUND',3X,'NTVCON',3X,'SVTVEC',3X,'IREAD'/3I9,I8)    HHL02310
C                                                                       HHL02320
      WRITE(6,60) TVSTOP,LVCONT,ERCONT,IWRITE                           HHL02330
   60 FORMAT(/3X,'TVSTOP',3X,'LVCONT',3X,'ERCONT',3X,'IWRITE'/4I9)      HHL02340
C                                                                       HHL02350
      WRITE(6,70) MDIMTV,MDIMRV,MBETA                                   HHL02360
   70 FORMAT(/3X,'MDIMTV',3X,'MDIMRV',3X,'MBETA'/2I9,I8)                HHL02370
C                                                                       HHL02380
      WRITE(6,80) RELTOL,RHSEED                                         HHL02390
   80 FORMAT(/7X,'RELTOL',3X,'RHSEED'/E13.4,I9)                         HHL02400
C                                                                       HHL02410
C                                                                       HHL02420
C     FROM FILE 3 READ IN THE NUMBER OF EIGENVALUES (NGOOD) FOR WHICH   HHL02430
C     EIGENVECTORS ARE REQUESTED, THE ORDER (MEV) OF THE LANCZOS        HHL02440
C     TRIDIAGONAL MATRIX USED IN COMPUTING THESE EIGENVALUES, THE       HHL02450
C     ORDER (NOLD) OF THE USER-SPECIFIED MATRIX USED IN THE EIGENVALUE  HHL02460
C     COMPUTATIONS, THE SEED (SVSEED) USED FOR GENERATING THE STARTING  HHL02470
C     VECTOR THAT WAS USED IN THOSE LANCZOS EIGENVALUE COMPUTATIONS,    HHL02480
C     AND THE MATRIX/RUN IDENTIFICATION NUMBER (MATOLD) USED IN THOSE   HHL02490
C     COMPUTATIONS.  ALSO READ IN THE NUMBER (NDIS) OF DISTINCT         HHL02500
C     EIGENVALUES OF T(1,MEV) THAT WERE COMPUTED BUT THIS VALUE IS      HHL02510
C     NOT USED IN THE EIGENVECTOR COMPUTATIONS.                         HHL02520
C                                                                       HHL02530
      READ(3,90) NGOOD,NDIS,MEV,NOLD,SVSEED,MATOLD                      HHL02540
   90 FORMAT(4I6,I12,I8)                                                HHL02550
C                                                                       HHL02560
C     READ IN THE T-MULTIPLICITY TOLERANCE USED IN THE BISEC SUBROUTINE HHL02570
C     DURING THE COMPUTATION OF THE GIVEN EIGENVALUES.                  HHL02580
C     ALSO READ IN THE FLAG IB.  IF IB < 0, THEN SOME BETA(I) IN THE    HHL02590
C     T-MATRIX FILE PROVIDED ON FILE 2 FAILED THE ORTHOGONALITY         HHL02600
C     TEST IN THE TNORM SUBROUTINE.  USER SHOULD NOTE THAT THIS         HHL02610
C     VECTOR PROGRAM PROCEEDS INDEPENDENTLY OF THE SIZE OF THE BETA.    HHL02620
C                                                                       HHL02630
```

```
      READ(3,100) MULTOL,IB,BTOL                                        HHL02640
  100 FORMAT(E20.12,I6,E13.4)                                           HHL02650
C                                                                       HHL02660
      TEMP = DFLOAT(MEV+1000)                                           HHL02670
      TTOL = MULTOL/TEMP                                                HHL02680
      WRITE(6,110) MULTOL,TTOL                                          HHL02690
  110 FORMAT(/' T-MULTIPLICITY TOLERANCE USED IN THE EIGENVALUE COMPUTATHHL02700
     1IONS WAS',E13.4/' SCALED MACHINE EPSILON IS',E13.4)               HHL02710
C                                                                       HHL02720
C     CONTINUE WRITE TO FILE 6 OF THE PARAMETERS FOR THIS RUN           HHL02730
C                                                                       HHL02740
      WRITE(6,120)NGOOD,NDIS,MEV,NOLD,MATOLD,SVSEED,MULTOL,IB,BTOL      HHL02750
  120 FORMAT(/' EIGENVALUES SUPPLIED ARE READ IN FROM FILE 3'/' FILE 3  HHL02760
     1HEADER IS'/4X,'NG',2X,'NDIS',3X,'MEV',2X,'NOLD',2X,'MATOLD',4X,   HHL02770
     1'SVSEED',6X,'MULTOL',6X,'IB',9X,'BTOL'/4I6,I8,I10,E12.3,I8,E13.4/)HHL02780
C                                                                       HHL02790
C     IS THE ARRAY RITVEC LONG ENOUGH TO HOLD ALL OF THE DESIRED        HHL02800
C     RITZ VECTORS (APPROXIMATE EIGENVECTORS)?                          HHL02810
      NMAX = NGOOD*N                                                    HHL02820
      IF(MBOUND.NE.0) GO TO 130                                         HHL02830
      IF(TVSTOP.NE.1.AND.NMAX.GT.MDIMRV) GO TO 1390                     HHL02840
C                                                                       HHL02850
C     CHECK THAT THE ORDER N AND THE MATRIX IDENTIFICATION NUMBER       HHL02860
C     MATNO SPECIFIED BY THE USER AGREE WITH THOSE READ IN FROM         HHL02870
C     FILE 3.                                                           HHL02880
  130 ITEMP = (NOLD-N)**2+(MATOLD-MATNO)**2                             HHL02890
      IF (ITEMP.NE.0) GO TO 1410                                        HHL02900
C                                                                       HHL02910
C                                                                       HHL02920
C     THE LANCZOS EIGENVECTOR COMPUTATIONS ASSUME THAT EACH             HHL02930
C     EIGENVALUE THAT IS BEING CONSIDERED HAS CONVERGED AS AN           HHL02940
C     EIGENVALUE OF THE LANCZOS TRIDIAGONAL MATRICES.                   HHL02950
C                                                                       HHL02960
C     READ IN FROM FILE 3, THE T-MULTIPLICITIES OF THE EIGENVALUES      HHL02970
C     WHOSE EIGENVECTORS ARE TO BE COMPUTED, THE VALUES OF THESE        HHL02980
C     EIGENVALUES AND THEIR MINIMAL GAPS AS EIGENVALUES OF THE          HHL02990
C     USER-SPECIFIED MATRIX AND AS EIGENVALUES OF THE T-MATRIX.         HHL03000
C                                                                       HHL03010
      READ(3,20) EXPLAN                                                 HHL03020
      READ(3,140) (MP(J),GOODEV(J),TMINGP(J),AMINGP(J), J=1,NGOOD)      HHL03030
  140 FORMAT(6X,I6,E25.16,2E14.3)                                       HHL03040
C                                                                       HHL03050
      WRITE(6,150) (J,GOODEV(J),MP(J),TMINGP(J),AMINGP(J), J=1,NGOOD)   HHL03060
  150 FORMAT(/' EIGENVALUES READ IN, T-MULTIPLICITIES, T-GAPS AND A-GAPSHHL03070
     1 '/4X,' J ',5X,'GOOD EIGENVALUE',5X,'MULT',4X,'  TMINGAP ',4X,    HHL03080
     1'  AMINGAP '/(I6,E25.16,I4,2E15.4))                               HHL03090
C                                                                       HHL03100
C     READ IN ERROR ESTIMATES                                           HHL03110
      WRITE(6,180) MEV,SVSEED                                           HHL03120
C     CHECK WHETHER OR NOT THERE ARE ANY ISOLATED T-EIGENVALUES IN      HHL03130
C     THE EIGENVALUES PROVIDED                                          HHL03140
      DO 160 J=1,NGOOD                                                  HHL03150
      IF(MP(J).EQ.1) GO TO 170                                          HHL03160
  160 CONTINUE                                                          HHL03170
      GO TO 200                                                         HHL03180
```

```
  170 READ(4,20) EXPLAN                                                 HHL03190
      READ(4,20) EXPLAN                                                 HHL03200
      READ(4,20) EXPLAN                                                 HHL03210
  180 FORMAT(/' THESE EIGENVALUES WERE COMPUTED USING A T-MATRIX OF     HHL03220
     1ORDER ',I5/' AND SEED FOR RANDOM NUMBER GENERATOR =',I12)         HHL03230
      READ(4,190) NISO                                                  HHL03240
  190 FORMAT(18X,I6)                                                    HHL03250
      READ(4,20) EXPLAN                                                 HHL03260
      READ(4,20) EXPLAN                                                 HHL03270
      READ(4,20) EXPLAN                                                 HHL03280
  200 DO 230 J=1,NGOOD                                                  HHL03290
      ERR(J) = 0.D0                                                     HHL03300
      IF(MP(J).NE.1) GO TO 230                                          HHL03310
      READ(4,210) EVAL, ERR(J)                                          HHL03320
  210 FORMAT(10X,E25.16,E14.3)                                          HHL03330
      IF(DABS(EVAL - GOODEV(J)).LT.1.D-10) GO TO 230                    HHL03340
      WRITE(6,220) EVAL,GOODEV(J)                                       HHL03350
  220 FORMAT(' PROBLEM WITH READ IN OF ERROR ESTIMATES'/' EIGENVALUE REAHHL03360
     1D IN',E20.12,' DOES NOT MATCH GOODEV(J) ='/E20.12)                HHL03370
      GO TO 1630                                                        HHL03380
C                                                                       HHL03390
  230 CONTINUE                                                          HHL03400
C                                                                       HHL03410
      WRITE(6,240) (J,GOODEV(J),ERR(J), J=1,NGOOD)                      HHL03420
  240 FORMAT(' ERROR ESTIMATES ='/4X,' J',5X,'EIGENVALUE',10X,'ESTIMATE'HHL03430
     1 /(I6,E20.12,E14.3))                                              HHL03440
C                                                                       HHL03450
      IF(IREAD.EQ.0) GO TO 340                                          HHL03460
C                                                                       HHL03470
C     READ IN THE SIZE OF THE T-MATRIX PROVIDED ON FILE 2.  READ IN     HHL03480
C     THE ORDER OF THE USER-SPECIFIED MATRIX , THE SEED FOR THE         HHL03490
C     RANDOM NUMBER GENERATOR, AND THE MATRIX/TEST IDENTIFICATION       HHL03500
C     NUMBER THAT WERE USED IN THE LANCZOS EIGENVALUE COMPUTATIONS.     HHL03510
C     IF FLAG IREAD = 0, REGENERATE HISTORY.  HISTORY MUST BE           HHL03520
C     STORED IN HEXADECIMAL FORMAT TO AVOID ERRORS INCURRED IN          HHL03530
C     INPUT/OUTPUT CONVERSIONS.                                         HHL03540
C                                                                       HHL03550
      READ(2,250) KMAX,NOLD,SVSOLD,MATOLD                               HHL03560
  250 FORMAT(2I6,I12,I8)                                                HHL03570
C                                                                       HHL03580
      WRITE(6,260) KMAX,NOLD,SVSOLD,MATOLD                              HHL03590
  260 FORMAT(/' READ IN THE T-MATRICES STORED ON FILE 2'/' FILE 2 HEADERHHL03600
     1 IS'/2X,'KMAX',2X,'NOLD',6X,'SVSOLD',2X,'MATOLD'/2I6,I12,I8/)     HHL03610
C                                                                       HHL03620
C     CHECK THAT THE ORDER, THE MATRIX/TEST IDENTIFICATION NUMBER       HHL03630
C     AND THE SEED FOR THE RANDOM NUMBER GENERATOR USED IN THE          HHL03640
C     LANCZOS COMPUTATIONS THAT GENERATED THE HISTORY FILE              HHL03650
C     BEING USED AGREE WITH WHAT THE USER HAS SPECIFIED.                HHL03660
      IF (NOLD.NE.N.OR.MATOLD.NE.MATNO.OR.SVSOLD.NE.SVSEED) GO TO 1430  HHL03670
C                                                                       HHL03680
      KMAX1 = KMAX + 1                                                  HHL03690
C                                                                       HHL03700
C     READ IN THE T-MATRICES FROM FILE 2.  THESE ARE USED TO GENERATE   HHL03710
C     THE T-EIGENVECTORS THAT WILL BE USED IN THE RITZ VECTOR           HHL03720
C     COMPUTATIONS.  ALPHA/BETA HISTORY MUST BE STORED IN               HHL03730
```

```
C     MACHINE FORMAT, ((4Z20) FOR IBM/3081).                            HHL03740
C                                                                       HHL03750
      READ(2,270) (ALPHA(J), J=1,KMAX)                                  HHL03760
      READ(2,270) (BETA(J), J=1,KMAX1)                                  HHL03770
  270 FORMAT(4Z20)                                                      HHL03780
C                                                                       HHL03790
      READ(2,270) (V1(J), J=1,N)                                        HHL03800
      READ(2,270) (V2(J), J=1,N)                                        HHL03810
C                                                                       HHL03820
C     ENLARGE KMAX IF THE SIZE AT WHICH THE EIGENVALUE                  HHL03830
C     COMPUTATIONS WERE PERFORMED IS ESSENTIALLY KMAX AND               HHL03840
C     THERE IS AT LEAST ONE EIGENVALUE THAT IS T-SIMPLE AND             HHL03850
C     T-ISOLATED, IN THE SENSE THAT IF ITS NEAREST T-NEIGHBOR IS TOO    HHL03860
C     CLOSE THAT NEIGHBOR IS A 'GOOD' T-EIGENVALUE.                     HHL03870
      DO 280 J = 1,NGOOD                                                HHL03880
      IF(MP(J).EQ.1) GO TO 300                                          HHL03890
  280 CONTINUE                                                          HHL03900
      WRITE(6,290)                                                      HHL03910
  290 FORMAT(/' ALL EIGENVALUES USED ARE T-MULTIPLE OR CLOSE TO SPURIOUSHHL03920
     1 T-EIGENVALUES'/' SO DO NOT CHANGE KMAX')                         HHL03930
      IF(KMAX.LT.MEV) GO TO 1450                                        HHL03940
      GO TO 320                                                         HHL03950
C                                                                       HHL03960
  300 KMAXN= 11*MEV/8 + 12                                              HHL03970
      IF(MBETA.LE.KMAXN) GO TO 1610                                     HHL03980
      IF(KMAX.GE.KMAXN )  GO TO 320                                     HHL03990
      WRITE(6,310) KMAX, KMAXN                                          HHL04000
  310 FORMAT(' ENLARGE KMAX FROM ',I6,' TO ',I6)                        HHL04010
      MOLD1 = KMAX + 1                                                  HHL04020
      KMAX = KMAXN                                                      HHL04030
      GO TO 390                                                         HHL04040
C                                                                       HHL04050
  320 WRITE(6,330) KMAX                                                 HHL04060
  330 FORMAT(/' T-MATRICES HAVE BEEN READ IN FROM FILE 2'/' THE LARGEST HHL04070
     1SIZE T-MATRIX ALLOWED IS',I6/)                                    HHL04080
C                                                                       HHL04090
      IF(IREAD.EQ.1) GO TO 410                                          HHL04100
C                                                                       HHL04110
C     REGENERATE THE ALPHA AND BETA                                     HHL04120
C                                                                       HHL04130
  340 MOLD1 = 1                                                         HHL04140
C                                                                       HHL04150
      DO 350 J = 1,NGOOD                                                HHL04160
      IF(MP(J).EQ.1) GO TO 370                                          HHL04170
  350 CONTINUE                                                          HHL04180
      KMAX = MEV + 12                                                   HHL04190
      WRITE(6,360) KMAX                                                 HHL04200
  360 FORMAT(/' ALL EIGENVALUES FOR WHICH EIGENVECTORS ARE TO BE COMPUTEHHL04210
     1D ARE EITHER T-MULTIPLE OR CLOSE TO'/' A SPURIOUS EIGENVALUE. THERHHL04220
     1EFORE SET KMAX = MEV + 12 = ',I7)                                 HHL04230
      GO TO 390                                                         HHL04240
C                                                                       HHL04250
  370 KMAXN = 11*MEV/8 + 12                                             HHL04260
      IF(MBETA.LE.KMAXN) GO TO 1610                                     HHL04270
      WRITE(6,380) KMAXN                                                HHL04280
```

```
  380 FORMAT(' SET KMAX EQUAL TO ',I6)                                  HHL04290
      KMAX = KMAXN                                                      HHL04300
C                                                                       HHL04310
  390 WRITE(6,400) MOLD1,KMAX                                           HHL04320
  400 FORMAT(/' LANCZS SUBROUTINE GENERATES ALPHA(J), BETA(J+1), J =',  HHL04330
     1 I6,' TO ', I6/)                                                  HHL04340
C                                                                       HHL04350
C-----------------------------------------------------------------------HHL04360
C                                                                       HHL04370
      CALL LANCZS(CMATV,V1,V2,VS,ALPHA,BETA,GR,GC,G,KMAX,MOLD1,N,SVSEED)HHL04380
C                                                                       HHL04390
C-----------------------------------------------------------------------HHL04400
C                                                                       HHL04410
  410 CONTINUE                                                          HHL04420
C                                                                       HHL04430
C     THE SUBROUTINE STURMI DETERMINES THE SMALLEST SIZE T-MATRIX FOR   HHL04440
C     WHICH THE EIGENVALUE IN QUESTION IS AN EIGENVALUE (TO WITHIN A    HHL04450
C     GIVEN TOLERANCE) AND IF POSSIBLE THE SMALLEST SIZE T-MATRIX       HHL04460
C     FOR WHICH IT IS A DOUBLE EIGENVALUE (TO WITHIN THE SAME           HHL04470
C     TOLERANCE).  THE SIZE T-MATRIX USED IN THE EIGENVECTOR            HHL04480
C     COMPUTATIONS IS THEN DETERMINED BY LOOPING ON THE SIZES OF THE    HHL04490
C     T-EIGENVECTORS, USING THE INFORMATION FROM STURMI TO OBTAIN       HHL04500
C     STARTING GUESSES AT THE T-SIZES.                                  HHL04510
C                                                                       HHL04520
C                                                                       HHL04530
      STUTOL = SCALE0*MULTOL                                            HHL04540
      IF(IWRITE.EQ.1) WRITE(6,420)                                      HHL04550
  420 FORMAT(' FROM STURMI')                                            HHL04560
      DO 460 J = 1,NGOOD                                                HHL04570
      EVAL = GOODEV(J)                                                  HHL04580
C     COMPUTE THE TOLERANCES USED BY STURMI TO DETERMINE AN INTERVAL    HHL04590
C     CONTAINING THE EIGENVALUE EVAL.                                   HHL04600
      TEMP = DABS(EVAL)*RELTOL                                          HHL04610
      TOLN = DMAX1(TEMP,STUTOL)                                         HHL04620
C                                                                       HHL04630
C-----------------------------------------------------------------------HHL04640
C                                                                       HHL04650
      CALL STURMI(ALPHA,BETA,EVAL,TOLN,EPSM,KMAX,MK1,MK2,IC,IWRITE)     HHL04660
C                                                                       HHL04670
C-----------------------------------------------------------------------HHL04680
C                                                                       HHL04690
C     STORE THE COMPUTED ORDERS OF T-MATRICES FOR LATER PRINTOUT        HHL04700
      M1(J) = MK1                                                       HHL04710
      M2(J) = MK2                                                       HHL04720
      ML(J) = (MK1 + 3*MK2)/4                                           HHL04730
      IF(MK2.EQ.KMAX)  ML(J) = KMAX                                     HHL04740
C                                                                       HHL04750
      IF(IC.GT.0) GO TO 440                                             HHL04760
C     IC = 0 MEANS THERE WAS NO T-EIGENVALUE IN THE DESIGNATED INTERVAL HHL04770
C     BY T-SIZE KMAX.  THIS MEANS THAT THE T-EIGENVALUE PROVIDED HAS    HHL04780
C     NOT YET CONVERGED AS AN EIGENVALUE OF THE TRIDIAGONAL MATRICES    HHL04790
C     SO PROGRAM SHOULD NOT COMPUTE ITS EIGENVECTOR.                    HHL04800
      WRITE(6,430) J,GOODEV(J),MK1,MK2                                  HHL04810
  430 FORMAT(I6,'TH EIGENVALUE',E20.12,' HAS NOT CONVERGED '/           HHL04820
     1' SO DO NOT COMPUTE ANY T-EIGENVECTOR OR RITZ VECTOR FOR IT'      HHL04830
```

```
     1/' MK1 AND MK2 FOR THIS EIGENVALUE WERE',2I6)
      MP(J) = MPMIN
      MA(J) = -2*KMAX
      GO TO 460
C     COMPUTE AN APPROPRIATE SIZE T-MATRIX FOR THE GIVEN EIGENVALUE.
  440 IF(M2(J).EQ.KMAX) GO TO 450
C     M1 AND M2 WERE BOTH DETERMINED
      MA(J) = (3*M1(J) + M2(J))/4  + 1
      GO TO 460
C     M2 NOT DETERMINED
  450 MA(J) = 5*M1(J)/4  + 1
C
  460 CONTINUE
C
      IF (IWRITE.EQ.1) WRITE(6,470) (MA(JJ), JJ=1,NGOOD)
  470 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZE T-MATRICES'/
     1 ' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/(13I6))
C
C     PRINT OUT TO FILE 10 1ST GUESSES AT SIZES OF T-MATRICES TO
C     BE USED IN THE EIGENVECTOR COMPUTATIONS.
C     ACTUAL SIZES MAY BE 1/4 OR MORE LARGER THAN THESE SIZES.
      WRITE(10,480) N,KMAX
  480 FORMAT(2I8,' = ORDER OF USER MATRIX AND MAX ORDER OF T(1,MEV)')
C
      WRITE(10,490)
  490 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZE T-MATRICES'/
     1 ' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/)
C
      WRITE(10,500)
  500 FORMAT(4X,'J',7X,'GOODEV(J)',4X,'M1(J)',1X,'M2(J)',1X,'MA(J)')
C
      WRITE(10,510) (J,GOODEV(J),M1(J),M2(J), MA(J), J=1,NGOOD)
  510 FORMAT(I5,E19.12,3I6)
C
      IF(MBOUND.EQ.1) WRITE(10,520)
  520 FORMAT(/' GOODEV(J) IS A GOOD EIGENVALUE OF T(1,MEV)'/
     1 ' M1 = SMALLEST VALUE OF M SUCH THAT T(1,M) HAS AT LEAST'/
     1 '      ONE EIGENVALUE IN THE INTERVAL (EV-TOLN,EV+TOLN)'/
     1 '      TOLN(J) = DMAX1(GOODEV(J)*RELTOL, SCALE0*MULTOL)'/
     1 ' M2 = SMALLEST M (IF ANY) SUCH THAT IN THE ABOVE INTERVAL'/
     1 '      T(1,M) HAS AT LEAST TWO EIGENVALUES '/
     1 ' INITIAL VALUE OF MA(J) IS CHOSEN HEURISTICALLY'/
     1 ' PROGRAM LOOPS ON SIZE OF T-MATRIX TO GET BETTER SIZE'/
     1 ' END OF SIZES OF T-MATRICES FILE 10'///)
C
C
C     TERMINATE AFTER COMPUTING 1ST GUESSES AT  SIZES OF THE
C     T-MATRICES REQUIRED FOR THE GIVEN EIGENVALUES?
      IF(MBOUND.EQ.1) GO TO 1470
C
C
C     IS THERE ROOM FOR ALL OF THE REQUESTED T-EIGENVECTORS?
      MTOL = 0
      DO 530 J = 1,NGOOD
      IF(MP(J).EQ.MPMIN) GO TO 530
```

```
      MTOL = MTOL + IABS(MA(J))                                         HHL05390
  530 CONTINUE                                                          HHL05400
      MTOL = (5*MTOL)/4                                                 HHL05410
      IF(MTOL.GT.MDIMTV.AND.NTVCON.EQ.0) GO TO 1490                     HHL05420
C                                                                       HHL05430
C-----------------------------------------------------------------------HHL05440
C     GENERATE A RANDOM VECTOR TO BE USED REPEATEDLY BY                 HHL05450
C     SUBROUTINE INVERM                                                 HHL05460
C                                                                       HHL05470
      IIL = RHSEED                                                      HHL05480
      CALL GENRAN(IIL,G,KMAX)                                           HHL05490
C                                                                       HHL05500
C---------------------------------------------------------------------- HHL05510
C                                                                       HHL05520
C     LOOP ON GIVEN EIGENVALUES TO COMPUTE THE CORRESPONDING            HHL05530
C     T-EIGENVECTOR.                                                    HHL05540
C                                                                       HHL05550
      MTOL = 0                                                          HHL05560
      NTVEC = 0                                                         HHL05570
      ILBIS = 0                                                         HHL05580
      DO 720 J = 1,NGOOD                                                HHL05590
      ICOUNT = 0                                                        HHL05600
      ERRMIN = 10.D0                                                    HHL05610
      MABEST = MPMIN                                                    HHL05620
      IF(MP(J).EQ.MPMIN) GO TO 720                                      HHL05630
      TFLAG = 0                                                         HHL05640
      EVAL = GOODEV(J)                                                  HHL05650
      TEMP = RELTOL* DABS(EVAL)                                         HHL05660
      UB = EVAL + DMAX1(STUTOL,TEMP)                                    HHL05670
      LB = EVAL - DMAX1(STUTOL,TEMP)                                    HHL05680
  540 KMAXU = IABS(MA(J))                                               HHL05690
C                                                                       HHL05700
C     SELECT A SUITABLE INCREMENT FOR THE ORDERS OF THE T-MATRICES      HHL05710
C     TO BE CONSIDERED IN DETERMINING APPROPRIATE SIZES FOR THE RITZ    HHL05720
C     VECTOR COMPUTATIONS.                                              HHL05730
      IF(ICOUNT.GT.0) GO TO 560                                         HHL05740
C     SELECT IDELTA(J) BASED UPON THE T-MULTIPLICITY OBTAINED           HHL05750
      IF(M2(J).EQ.KMAX) GO TO 550                                       HHL05760
C     M2 DETERMINED                                                     HHL05770
      IDELTA(J) = ((3*M1(J) + 5*M2(J))/8  +  1 - IABS(MA(J)))/10 + 1    HHL05780
      GO TO 560                                                         HHL05790
C     M2 NOT DETERMINED                                                 HHL05800
  550 MAMAX = MIN0((11*MEV)/8 + 12, (13*M1(J))/8 + 1)                   HHL05810
      IDELTA(J) = (MAMAX - IABS(MA(J)))/10  +  1                        HHL05820
  560 ICOUNT = ICOUNT + 1                                               HHL05830
C                                                                       HHL05840
C-----------------------------------------------------------------------HHL05850
C     TO MIMIMIZE THE EFFECT OF THE ONE-SIDED ACCEPTANCE TEST FOR       HHL05860
C     T-EIGENVALUES IN THE BISEC SUBROUTINE, RECOMPUTE THE GIVEN        HHL05870
C     EIGENVALUE AT THE SPECIFIED KMAXU                                 HHL05880
C                                                                       HHL05890
      CALL LBISEC(ALPHA,BETA,EPSM,EVAL,EVALN,LB,UB,TTOL,KMAXU,NEVT)     HHL05900
C                                                                       HHL05910
C-----------------------------------------------------------------------HHL05920
C                                                                       HHL05930
```

```
C     CHECK WHETHER OR NOT GIVEN T-MATRIX HAS AN EIGENVALUE IN THE
C     SPECIFIED INTERVAL AND IF SO WHAT ITS T-MULTIPLICITY IS.
C
      IF(NEVT.EQ.1) GO TO 600
      IF(NEVT.NE.0) GO TO 580
      ILBIS = 1
      WRITE(6,570) EVAL,KMAXU
  570 FORMAT(/' PROBLEM ENCOUNTERED IN RECOMPUTATION OF USER-SUPPLIED EI
     1GENVALUE',E20.12/' THE SIZE T-MATRIX SPECIFIED',I6,' DOES NOT
     1HAVE AN EIGENVALUE IN THE INTERVAL SPECIFIED'/' THEREFORE NO EIGEN
     1VECTOR WILL BE COMPUTED FOR THIS PARTICULAR EIGENVALUE'/)
      GO TO 620
C
  580 IF(NEVT.GT.1)  WRITE(6,590) EVAL,KMAXU
  590 FORMAT(/' PROBLEM ENCOUNTERED IN RECOMPUTATION OF USER-SUPPLIED
     1EIGENVALUE',E20.12/' FOR THE SIZE T-MATRIX SPECIFIED =',I6,' THE
     1GIVEN EIGENVALUE IS T-MULTIPLE IN THE INTERVAL SPECIFIED'/' SOMETH
     1ING IS WRONG, THEREFORE NO EIGENVECTOR WILL BE COMPUTED FOR THIS E
     1IGENVALUE'/)
C
      MP(J) = MPMIN
      MA(J) = -2*KMAX
      GO TO 720
C
  600 CONTINUE
      ILBIS = 0
C
      EVNEW(J) = EVALN
      EVAL = EVALN
      MTOL = MTOL+KMAXU
C
C     IS THERE ROOM IN TVEC ARRAY FOR THE NEXT T-EIGENVECTOR?
C     IF NOT, SKIP TO RITZ VECTOR COMPUTATIONS.
      IF (MTOL.GT.MDIMTV) GO TO 730
C
      IT = 3
      KINT = MTOL - KMAXU +1
C
C     RECORD THE BEGINNING AND END OF THE T-EIGENVECTOR BEING COMPUTED
      MINT(J) = KINT
      MFIN(J) = MTOL
C
C-----------------------------------------------------------------------
C     SUBROUTINE INVERM DOES INVERSE ITERATION, I.E. SOLVES
C     (T(1,KMAXU) - EVAL)*U = RHS  FOR EACH EIGENVALUE TO OBTAIN THE
C     DESIRED T-EIGENVECTOR.
C
      IF(IWRITE.EQ.1)  WRITE(6,610) J
  610 FORMAT(/I6,'TH EIGENVALUE')
C
      CALL INVERM(ALPHA,BETA,V1,TVEC(KINT),EVAL,ERROR,TERROR,EPSM,
     1 G,KMAXU,IT,IWRITE)
C
C-----------------------------------------------------------------------
C
```

```
      TERR(J) = TERROR
      TLAST(J) = ERROR
      KMAXU1 = KMAXU + 1
      TBETA(J) = BETA(KMAXU1)*ERROR
C
C     AFTER COMPUTING EACH OF THE T-EIGENVECTORS,
C     CHECK THE SIZE OF THE ERROR ESTIMATE, ERROR.
C     IF THIS ESTIMATE IS NOT AS SMALL AS DESIRED AND
C     |MA(J)| < ML(J), ATTEMPT TO INCREASE THE SIZE OF |MA(J)|
C     AND REPEAT THE T-EIGENVECTOR COMPUTATIONS.
C
      IF(ERROR.LT.ERTOL.OR.TFLAG.EQ.1)  GO TO 710
C
      IF(ERROR.GE.ERRMIN) GO TO 620
C     LAST COMPONENT IS LESS THAN MINIMAL TO DATE
      ERRMIN = ERROR
      MABEST = MA(J)
  620 CONTINUE
C
      IF(MA(J).GT.0)  ITEST = MA(J) + IDELTA(J)
      IF(MA(J).LT.0)  ITEST = -(IABS(MA(J)) + IDELTA(J))
      IF(IABS(ITEST).LE.ML(J).AND.ICOUNT.LE.10) GO TO 640
C     NEW MA(J) IS GREATER THAN MAXIMUM ALLOWED.
      IF(ERCONT.EQ.0.OR.MABEST.EQ.MPMIN) GO TO 660
      TFLAG = 1
      MA(J) = MABEST
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU
      WRITE(6,630) MA(J)
  630 FORMAT(' 10 ORDERS WERE CONSIDERED.  NONE SATISFIED THE ERROR TEST
     1'/' THEREFORE USE THE BEST ORDER OBTAINED FOR THE EIGENVECTORS'
     1,I6)
      GO TO 540
C
  640 MA(J) = ITEST
C
      MT = IABS(MA(J))
      IF(IWRITE.EQ.1) WRITE(6,650)  MT
  650 FORMAT(/' CHANGE SIZE OF T-MATRIX TO ',I6,' RECOMPUTE T-EIGENVECTO
     1R')
C
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU
C
      GO TO 540
C
C       APPROPRIATE SIZE T-MATRIX WAS NOT OBTAINED
  660 CONTINUE
      WRITE(10,670) J,EVAL,MP(J)
  670 FORMAT(/' ON 10 INCREMENTS NOT ABLE TO IDENTIFY APPROPRIATE SIZE
     1T-MATRIX FOR'/
     1' EIGENVALUE(',I4,') = ',E20.12,' T-MULTIPLICITY =',I4/)
      IF(M2(J).EQ.KMAX) WRITE(10,680)
      IF(M2(J).LT.KMAX) WRITE(10,690)
  680 FORMAT(/' ORDERS TESTED RANGED FROM 5*M1(J)/4 TO APPROXIMATELY'/
     1'   MIN(11*MEV/8, 13*M1(J)/8)'/)
  690 FORMAT(/' ORDERS TESTED RANGED FROM (3*M1(J)+M2(J)/4 TO APPROXIMAT
```

```
     1ELY'/'  (3*M1(J) + 5*M2(J))/8'/)                                  HHL07040
      WRITE(10,700)                                                     HHL07050
  700 FORMAT(' ALLOWING LARGER ORDERS FOR THE T-MATRICES MAY RESULT IN  HHL07060
     1 SUCCESS'/' BUT PROBABLY WILL NOT.  PROBLEM IS PROBABLY DUE TO'   HHL07070
     1 /' LACK OF CONVERGENCE OF GIVEN EIGENVALUE, CHECK THE ERROR ESTIMHHL07080
     1ATE')                                                             HHL07090
      MP(J) = MPMIN                                                     HHL07100
      IF(ILBIS.EQ.0) MTOL = MTOL - KMAXU                                HHL07110
      GO TO 720                                                         HHL07120
  710 NTVEC = NTVEC + 1                                                 HHL07130
C                                                                       HHL07140
  720 CONTINUE                                                          HHL07150
      NGOODC = NGOOD                                                    HHL07160
      GO TO 750                                                         HHL07170
C                                                                       HHL07180
C     COME HERE IF THERE IS NOT ENOUGH ROOM FOR ALL OF T-EIGENVECTORS   HHL07190
  730 NGOODC = J-1                                                      HHL07200
      WRITE(6,740) J,MTOL,MDIMTV                                        HHL07210
  740 FORMAT(/' NOT ENOUGH ROOM IN TVEC ARRAY FOR ',I4,'TH T-EIGENVECTORHHL07220
     1'/'  TVEC DIMENSION REQUESTED = ',I6,' BUT TVEC HAS DIMENSION ',I6HHL07230
     1/)                                                                HHL07240
      IF(NGOODC.EQ.0)  GO TO 1510                                       HHL07250
      MTOL = MTOL-KMAXU                                                 HHL07260
C                                                                       HHL07270
  750 CONTINUE                                                          HHL07280
C                                                                       HHL07290
C     THE LOOP ON T-EIGENVECTOR COMPUTATIONS IS COMPLETE.               HHL07300
C     WRITE OUT THE SIZE T-MATRICES THAT WILL BE USED FOR               HHL07310
C     THE RITZ VECTOR COMPUTATIONS.                                     HHL07320
C                                                                       HHL07330
      WRITE(10,760)                                                     HHL07340
  760 FORMAT(/' SIZES OF T-MATRICES THAT WILL BE USED IN THE RITZ COMPUTHHL07350
     1ATIONS'/5X,'J',16X,'GOODEV(J)',1X,'MA(J)')                        HHL07360
C                                                                       HHL07370
      WRITE(10,770)   (J,GOODEV(J),MA(J), J=1,NGOOD)                    HHL07380
  770 FORMAT(I6,E25.14,I6)                                              HHL07390
      WRITE(10,520)                                                     HHL07400
C                                                                       HHL07410
      WRITE(6,780) MTOL                                                 HHL07420
  780 FORMAT(/' THE CUMULATIVE LENGTH OF THE T-EIGENVECTORS IS',I18)    HHL07430
C                                                                       HHL07440
      WRITE(6,790) NTVEC,NGOOD                                          HHL07450
  790 FORMAT(/I6,' T-EIGENVECTORS OUT OF',I6,' REQUESTED WERE COMPUTED')HHL07460
C                                                                       HHL07470
C     SAVE THE T-EIGENVECTORS ON FILE 11?                               HHL07480
      IF(TVSTOP.NE.1.AND.SVTVEC.EQ.0) GO TO 850                         HHL07490
C                                                                       HHL07500
      WRITE(11,800) NTVEC,MTOL,MATNO,SVSEED                             HHL07510
  800 FORMAT(I6,3I12,' = NTVEC,MTOL,MATNO,SVSEED')                      HHL07520
C                                                                       HHL07530
      DO 830 J=1,NGOODC                                                 HHL07540
C     IF MP(J) = MPMIN THEN NO SUITABLE T-EIGENVECTOR IS AVAILABLE      HHL07550
C     FOR THAT EIGENVALUE.                                              HHL07560
      IF(MP(J).EQ.MPMIN) WRITE(11,810) J,MA(J),GOODEV(J),MP(J)          HHL07570
  810 FORMAT(2I6,E20.12,I6/' TH EIGVAL,T-SIZE,EVALUE,FLAG,NO EIGVEC')   HHL07580
```

```
      IF(MP(J).NE.MPMIN) WRITE(11,820) J,MA(J),GOODEV(J),MP(J)
  820 FORMAT(I6,I6,E20.12,I6/' T-EIGVEC,SIZE T,EVALUE OF A,MP(J)')
      IF(MP(J).EQ.MPMIN) GO TO 830
      KI = MINT(J)
      KF = MFIN(J)
C
      WRITE(11,270) (TVEC(K), K=KI,KF)
C
  830 CONTINUE
C
      IF(TVSTOP.NE.1) GO TO 850
C
      WRITE(6,840) TVSTOP, NTVEC,NGOOD
  840 FORMAT(/' USER SET TVSTOP = ',I1/
     1' THEREFORE PROGRAM TERMINATES AFTER T-EIGENVECTOR COMPUTATIONS'/
     1' T-EIGENVECTORS THAT WERE COMPUTED ARE SAVED ON FILE 11'/
     1I8,' T-EIGENVECTORS WERE COMPUTED OUT OF',I7,' REQUESTED'/)
C
      GO TO 1630
C
  850 CONTINUE
C     IF NOT ABLE TO COMPUTE ALL THE REQUESTED T-EIGENVECTORS
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS ANYWAY?
      IF(NTVEC.NE.NGOOD.AND.LVCONT.EQ.0) GO TO 1530
C
C     COMPUTE THE MAXIMUM SIZE OF THE T-MATRIX USED FOR THOSE
C     EIGENVALUES WITH GOOD ERROR ESTIMATES.
C
      KMAXU = 0
      DO 860 J = 1,NGOODC
      MT = IABS(MA(J))
      IF(MT.LT.KMAXU.OR.MP(J).EQ.MPMIN) GO TO 860
      KMAXU = MT
  860 CONTINUE
C
      IF(KMAXU.EQ.0) GO TO 1570
C
      WRITE(6,870) KMAXU
  870 FORMAT(/I6,' = LARGEST SIZE T-MATRIX TO BE USED IN THE RITZ VECTOR
     1 COMPUTATIONS')
C
C     COUNT THE NUMBER OF RITZ VECTORS NOT BEING COMPUTED
      MREJEC = 0
      DO 880 J=1,NGOODC
  880 IF(MP(J).EQ.MPMIN)  MREJEC = MREJEC + 1
      MREJET = MREJEC + (NGOOD-NGOODC)
      IF(MREJET.NE.0) WRITE(6,890) MREJET
  890 FORMAT(/' RITZ VECTORS ARE NOT COMPUTED FOR',I6,' OF THE EIGENVALU
     1ES'/)
      NACT = NGOODC - MREJEC
      WRITE(6,900) NGOOD,NTVEC,NACT
  900 FORMAT(/I6,' RITZ VECTORS WERE REQUESTED'/I6,' T-EIGENVECTORS WERE
     1 COMPUTED'/I6,' RITZ VECTORS WILL BE COMPUTED'/)
C     CHECK IF THERE ARE ANY RITZ VECTORS TO COMPUTE
      IF(MREJEC.EQ.NGOODC) GO TO 1550
```

```
C                                                                       HHL08140
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS?                    HHL08150
      IF(LVCONT.EQ.0.AND.MREJEC.NE.0) GO TO 1530                        HHL08160
C                                                                       HHL08170
C     NOW COMPUTE THE RITZ VECTORS.  REGENERATE THE                     HHL08180
C     LANCZOS VECTORS.                                                  HHL08190
C                                                                       HHL08200
      DO 910 I = 1,NMAX                                                 HHL08210
  910 RITVEC(I) = ZEROC                                                 HHL08220
C                                                                       HHL08230
C     REGENERATE THE STARTING VECTOR. THIS MUST BE GENERATED AND        HHL08240
C     NORMALIZED PRECISELY THE WAY IT WAS DONE IN THE EIGENVALUE        HHL08250
C     COMPUTATIONS, OTHERWISE THERE WILL BE A MISMATCH BETWEEN          HHL08260
C     THE T-EIGENVECTORS THAT HAVE BEEN COMPUTED FROM THE T-MATRICES    HHL08270
C     READ IN FROM FILE 2 AND THE LANCZOS VECTORS THAT ARE              HHL08280
C     BEING REGENERATED.                                                HHL08290
C                                                                       HHL08300
C-----------------------------------------------------------------------HHL08310
C                                                                       HHL08320
      IIL = SVSEED                                                      HHL08330
      CALL GENRAN(IIL,G,N)                                              HHL08340
C                                                                       HHL08350
C-----------------------------------------------------------------------HHL08360
C                                                                       HHL08370
      DO 920 I = 1,N                                                    HHL08380
  920 GR(I) = G(I)                                                      HHL08390
C                                                                       HHL08400
C-----------------------------------------------------------------------HHL08410
C                                                                       HHL08420
      CALL GENRAN(IIL,G,N)                                              HHL08430
C                                                                       HHL08440
C-----------------------------------------------------------------------HHL08450
C                                                                       HHL08460
      DO 930 I = 1,N                                                    HHL08470
  930 GC(I) = G(I)                                                      HHL08480
C                                                                       HHL08490
      DO 940 I = 1,N                                                    HHL08500
  940 V2(I) = DCMPLX(GR(I),GC(I))                                       HHL08510
C                                                                       HHL08520
C-----------------------------------------------------------------------HHL08530
      CALL CINPRD(V2,V2,SUM,N)                                          HHL08540
C-----------------------------------------------------------------------HHL08550
C                                                                       HHL08560
      SUM = ONE/DSQRT(SUM)                                              HHL08570
      DO 950 I = 1,N                                                    HHL08580
      V1(I) = ZEROC                                                     HHL08590
  950 V2(I) = V2(I)*SUM                                                 HHL08600
C                                                                       HHL08610
C     LOOP FOR GENERATING REQUIRED RITZ VECTORS (IVEC = 1,KMAXU)        HHL08620
C     USES GRAM-SCHMIDT ORTHOGONALIZATION WITHOUT MODIFICATION          HHL08630
C                                                                       HHL08640
      IVEC = 1                                                          HHL08650
      BATA = ZERO                                                       HHL08660
C                                                                       HHL08670
      GO TO 1010                                                        HHL08680
```

```
C                                                                       HHL08690
  960 CONTINUE                                                          HHL08700
C                                                                       HHL08710
C-----------------------------------------------------------------------HHL08720
C     CMATV(V2,VS,SUM) CALCULATES  VS = A*V2 - SUM*VS                   HHL08730
      SUM = ZERO                                                        HHL08740
      CALL CMATV(V2,VS,SUM)                                             HHL08750
      CALL CINPRD(V2,VS,ALFA,N)                                         HHL08760
C                                                                       HHL08770
C-----------------------------------------------------------------------HHL08780
C                                                                       HHL08790
      DO 970 J=1,N                                                      HHL08800
  970 V1(J) = (VS(J) - BATA*V1(J)) - ALFA*V2(J)                         HHL08810
C                                                                       HHL08820
C-----------------------------------------------------------------------HHL08830
      CALL CINPRD(V1,V1,BATA,N)                                         HHL08840
C-----------------------------------------------------------------------HHL08850
C                                                                       HHL08860
      BATA = DSQRT(BATA)                                                HHL08870
      SUM = ONE/BATA                                                    HHL08880
C                                                                       HHL08890
      TEMP = BETA(IVEC)                                                 HHL08900
      TEMP = DABS(BATA - TEMP)/TEMP                                     HHL08910
      IF (TEMP.LT.1.0D-10)GO TO 990                                     HHL08920
C                                                                       HHL08930
C     THE BETA BEING REGENERATED DO NOT MATCH THE HISTORY FILE          HHL08940
C     SOMETHING IS WRONG IN THE LANCZOS VECTOR GENERATION               HHL08950
C     PROGRAM TERMINATES FOR USER TO CORRECT THE PROBLEM                HHL08960
C     WHICH MUST BE IN THE STARTING VECTOR GENERATION OR IN             HHL08970
C     THE MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV SUPPLIED.             HHL08980
C     THIS SUBROUTINE MUST BE THE SAME ONE USED IN THE                  HHL08990
C     EIGENVALUE COMPUTATIONS OR A MISMATCH WILL ENSUE.                 HHL09000
C                                                                       HHL09010
      WRITE(6,980) IVEC,BATA,BETA(IVEC),TEMP                            HHL09020
  980 FORMAT(/2X,'IVEC',16X,'BATA',10X,'BETA(IVEC)',14X,'RELDIF'/I6,    HHL09030
     13E20.12/' IN LANCZOS VECTOR REGENERATION THE ENTRIES OF THE TRIDIAHHL09040
     1GONAL MATRICES BEING'/' GENERATED ARE NOT THE SAME AS THOSE IN THEHHL09050
     1 MATRIX SUPPLIED ON FILE 2.'/' THEREFORE SOMETHING IS BEING INITIAHHL09060
     1LIZED OR COMPUTED DIFFERENTLY FROM THE WAY'/' IT WAS COMPUTED IN THHL09070
     1HE EIGENVALUE COMPUTATIONS'/' THE PROGRAM TERMINATES FOR THE USER HHL09080
     1TO DETERMINE WHAT THE PROBLEM IS'/)                               HHL09090
      GO TO 1630                                                        HHL09100
C                                                                       HHL09110
C                                                                       HHL09120
  990 CONTINUE                                                          HHL09130
      DO 1000 J = 1,N                                                   HHL09140
      TEMPC = SUM*V1(J)                                                 HHL09150
      V1(J) = V2(J)                                                     HHL09160
 1000 V2(J) = TEMPC                                                     HHL09170
C                                                                       HHL09180
 1010 CONTINUE                                                          HHL09190
C                                                                       HHL09200
      LFIN = 0                                                          HHL09210
      DO 1030 J = 1,NGOODC                                              HHL09220
      LL = LFIN                                                         HHL09230
```

```
      LFIN = LFIN + N
C
      IF(IABS(MA(J)).LT.IVEC.OR.MP(J).EQ.MPMIN) GO TO 1030
      II = IVEC + MINT(J) - 1
      TEMP = TVEC(II)
C     II IS THE (IVEC)TH COMPONENT OF THE T-EIGENVECTOR CONTAINED
C     IN TVEC(MINT(J)).
C
      DO 1020 K = 1,N
      LL = LL + 1
 1020 RITVEC(LL) = TEMP*V2(K) + RITVEC(LL)
C
 1030 CONTINUE
C
      IVEC = IVEC + 1
      IF (IVEC.LE.KMAXU) GO TO 960
C
C
C     RITZVECTOR GENERATION IS COMPLETE. NORMALIZE EACH RITZVECTOR.
C     NOTE THAT IF CERTAIN RITZ VECTORS WERE NOT COMPUTED THEN THAT
C     PORTION OF THE RITVEC ARRAY WAS NOT UTILIZED.
C
      LFIN = 0
      DO 1130 J = 1,NGOODC
C
      KK = LFIN
      LFIN = LFIN + N
      IF(MP(J).EQ.MPMIN) GO TO 1130
C
      DO 1040 K = 1,N
      KK = KK + 1
 1040 V2(K) = RITVEC(KK)
C
C-----------------------------------------------------------------------
      CALL CINPRD(V2,V2,SUM,N)
C-----------------------------------------------------------------------
C
      SUM = DSQRT(SUM)
      RNORM(J) = SUM
      TEMP = DABS(ONE-SUM)
      SUM = ONE/SUM
C
      KK = LFIN - N
      DO 1050 K = 1,N
      KK = KK + 1
      V2(K) = SUM*V2(K)
 1050 RITVEC(KK) = V2(K)
C
C    ONLY ENTER NEXT PORTION IF GIVEN MATRIX IS REAL.
      IF(ISREAL.NE.1) GO TO 1100
C
C    AT THIS POINT RITZ VECTOR IS IN V2.
C    THIS PROGRAM CAN BE USED ON REAL MATRICES TO DETERMINE
C    WHICH IF ANY EIGENVALUES ARE A-MULTIPLE AND IF SO TO COMPUTE
C    TWO EIGENVECTORS FOR THOSE EIGENVALUES THAT ARE MULTIPLE AND ONE
```

```
C    FOR THOSE THAT ARE NOT MULTIPLE. HERE ONLY IDENTIFIES WHETHER
C    EIGENVALUE IS AT LEAST DOUBLE.  THIS IS DONE BY CHECKING THE
C    RATIOS OF SUCCEEDING REAL AND IMAGINARY PARTS OF THE COMPUTED
C    RITZ VECTORS.
C
      SUM = DIMAG(V2(1))/DREAL(V2(1))
      DO 1060 K=2,N
      TEMP = DREAL(V2(K))
      IF(DABS(TEMP).LT.1.D-9) GO TO 1060
      TEMP = DIMAG(V2(K))/DREAL(V2(K))
      IF(DABS(TEMP - SUM).LE.1.D-6) GO TO 1060
      MULEVA(J) = 2
      GO TO 1070
 1060 CONTINUE
      MULEVA(J) = 1
 1070 IF(MULEVA(J).EQ.2) WRITE(6,1090) J,GOODEV(J)
      IF(MULEVA(J).EQ.1) WRITE(6,1080) J,GOODEV(J)
 1080 FORMAT(I6,'TH EIGENVALUE CONSIDERED =',E20.12,' IS SIMPLE')
 1090 FORMAT(I6,'TH EIGENVALUE CONSIDERED =',E20.12,' IS MULTIPLE')
C
 1100 CONTINUE
C
      IF (IWRITE.NE.0) WRITE(6,1110) J,GOODEV(J)
 1110 FORMAT(/I5,' TH EIGENVALUE CONSIDERED = ',E20.12/)
C
      IF (IWRITE.NE.0) WRITE(6,1120) TERR(J),TBETA(J),TEMP
 1120 FORMAT(' NORM OF ERROR IN T-EIGENVECTOR = ',E14.3/
     1 ' BETA(MA(J)+1)*U(MA(J))  = ',E14.3/
     1 ' ABS(NORM(RITVEC) - 1.0)  = ',E14.3/)
C
      LINT = LFIN - N + 1
      EVAL = EVNEW(J)
C
C-----------------------------------------------------------------------
C
      CALL CMATV(RITVEC(LINT),V2,EVAL)
C
C-----------------------------------------------------------------------
C
C     COMPUTE ERROR IN RITZ VECTOR CONSIDERED AS A EIGENVECTOR OF A.
C     V2 = A*RITVEC - EVAL*RITVEC
C
C-----------------------------------------------------------------------
      CALL CINPRD(V2,V2,SUM,N)
C-----------------------------------------------------------------------
C
      SUM = DSQRT(SUM)
      ERR(J) = SUM
      GAP = ABS(AMINGP(J))
      ERRDGP(J) = SUM/GAP
C
 1130 CONTINUE
C
C
C     RITZVECTORS ARE NORMALIZED AND ERROR ESTIMATES ARE IN ERR ARRAY
```

```
C     AND IN ERRDGP ARRAY. STORE EVERYTHING
C
C
      WRITE(9,1140)
 1140 FORMAT(6X,'GOODEV(J)',1X,'MA(J)',4X,'A MINGAP',6X,'AERROR',2X,
     1 'AERROR/GAP',6X,'TERROR')
C
      WRITE(13,1150)
 1150 FORMAT(16X,'GOODEV(J)',5X,'RITZNORM',6X,'AMINGAP',5X,
     1 'TBETA(J)',5X,'TLAST(J)')
C
      DO 1180 J=1,NGOODC
C
      IF(MP(J).EQ.MPMIN) GO TO 1180
C
      WRITE(9,1160)EVNEW(J),MA(J),AMINGP(J),ERR(J),ERRDGP(J),TERR(J)
 1160 FORMAT(E15.8,I6,4E12.4)
C
      WRITE(13,1170) EVNEW(J),RNORM(J),AMINGP(J),TBETA(J),TLAST(J)
 1170 FORMAT(E25.14,4E13.5)
C
 1180 CONTINUE
C
      IF(MREJEC.EQ.0) GO TO 1260
      WRITE(9,1190)
 1190 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING EIGENVA
     1LUES'/' EITHER BECAUSE THEY HAD NOT CONVERGED OR BECAUSE THE ERROR
     1 ESTIMATE'/'  WAS NOT AS SMALL AS DESIRED'/)
C
      DO 1250 J = 1,NGOODC
      IF(MP(J).NE.MPMIN) GO TO 1250
C     WRITE OUT MESSAGE FOR EACH EIGENVALUE FOR WHICH NO EIGENVECTOR
C     WAS COMPUTED.
C
      WRITE(9,1200)
 1200 FORMAT(6X,'GOODEV(J)',3X,'MA(J)',5X,'AMINGP(J)',6X,'TLAST(J)',3X,
     1'MP(J)')
      WRITE(9,1210) GOODEV(J),MA(J),AMINGP(J),TBETA(J),MP(J)
 1210 FORMAT(E15.8,I8,2E14.4,I8)
C
      WRITE(13,1220)
 1220 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING EIGENVA
     1LUES'/' EITHER BECAUSE THEY HAD NOT CONVERGED OR BECAUSE'/' THE ER
     1ROR ESTIMATE WAS NOT AS SMALL AS DESIRED'/)
C
      WRITE(13,1230)
 1230 FORMAT(6X,'GOODEV(J)',3X,'MA(J)',3X,'M1(J)',3X,'M2(J)',3X,'MP(J)'
     1/)
      WRITE(13,1240) GOODEV(J),MA(J),M1(J),M2(J),MP(J)
 1240 FORMAT(E15.8,4I8)
C
 1250 CONTINUE
 1260 CONTINUE
C
      WRITE(9,1270)
```

```
 1270 FORMAT(/' ABOVE ARE ERROR ESTIMATES FOR THE A AND T EIGENVECTORS'/
     1 ' ASSOCIATED WITH THE GOODEV LISTED IN COLUMN 1'/
     1 ' AERROR = NORM(A*X - EV*X)  TERROR = NORM(T*Y - EV*Y) '/
     1 ' WHERE T = T(1,MA(J))    X = RITZ VECTOR = V*Y  V = SUCCESSIVE'/
     1 ' LANCZOS VECTORS. A MINGAP = GAP TO NEAREST A-EIGENVALUE'//)
C
      WRITE(13,1280)
 1280 FORMAT(/' ABOVE ARE ERROR ESTIMATES ASSOCIATED WITH THE GOODEV'/
     1 ' RITZNORM = NORM(RITZ VECTOR)'/
     1 ' TBETA(J) = CDABS(BETA(MA(J)+1)*Y(MA(J))),  T*Y = GOODEV*Y'/
     1 ' TLAST(J) = CDABS(Y(MA(J)))'/
     1 ' AMINGAP = DISTANCE TO CLOSEST COMPUTED GOOD T-EIGENVALUE'/)
C
C     NUMBER OF RITZ VECTORS COMPUTED
      NCOMPU = NGOODC - MREJEC
      WRITE(12,1290) N,NCOMPU,NGOODC,MATNO
 1290 FORMAT(3I6,I12,' SIZE A, NO.RITZVECS, NO.EVALUES,MATNO')
C
      LFIN = 0
      DO 1350 J = 1,NGOODC
      LINT = LFIN + 1
      LFIN = LFIN + N
C
      IF(MP(J).EQ.MPMIN) GO TO 1330
C     RITZ VECTOR WAS COMPUTED
      WRITE(12,1300) J, GOODEV(J), MP(J)
 1300 FORMAT(I6,4X,E20.12,I6,' J, EIGENVAL, MP(J)')
C
      WRITE(12,1310) ERR(J),ERRDGP(J)
 1310 FORMAT(2E15.5,' = NORM(A*Z-EVAL*Z) AND  NORM(A*Z-EVAL*Z)/MINGAP')
C
      WRITE(12,1320) (RITVEC(LL), LL=LINT,LFIN)
 1320 FORMAT(4E20.12)
      GO TO 1350
C     NO RITZ VECTOR WAS COMPUTED FOR THIS EIGENVALUE
 1330 WRITE(12,1340) J,GOODEV(J),MP(J)
 1340 FORMAT(I6,4X,E20.12,I6,' J,EIGVALUE,NO RITZ VECTOR COMPUTED')
C
 1350 CONTINUE
C
C     DID ANY T-MATRICES INCLUDE OFF-DIAGONAL ENTRIES SMALLER THAN
C     DESIRED, AS SPECIFIED BY BTOL?
C
      IF(IB.GT.0) GO TO 1380
      WRITE(6,1360) KMAXU
 1360 FORMAT(/' FOR LARGEST T-MATRIX CONSIDERED',I7,' CHECK THE SIZE OF
     1BETAS')
C
C-----------------------------------------------------------------------
C
      CALL TNORM(ALPHA,BETA,BKMIN,TEMP,KMAXU,IBMT)
C
C-----------------------------------------------------------------------
C
      IF(IBMT.LT.0) WRITE (6,1370)
```

```
 1370 FORMAT(/' WARNING THE T-MATRICES FOR ONE OR MORE OF THE EIGENVALUEHHL11440
     1S CONSIDERED'/' HAD AN OFF-DIAGONAL ENTRY THAT WAS SMALLER THAN THHHL11450
     1E BETA TOLERANCE THAT WAS SPECIFIED'/)                            HHL11460
 1380 CONTINUE                                                          HHL11470
C                                                                       HHL11480
       GO TO 1630                                                       HHL11490
C                                                                       HHL11500
 1390 WRITE(6,1400) NGOOD,NMAX,MDIMRV                                   HHL11510
 1400 FORMAT(/I4,' RITZ VECTORS WERE REQUESTED BUT THE REQUIRED DIMENSIOHHL11520
     1N',I6/' IS LARGER THAN THE USER-SPECIFIED DIMENSION OF RITVEC',I6 HHL11530
     1/' THEREFORE, THE EIGENVECTOR PROCEDURE TERMINATES FOR THE USER TOHHL11540
     1 INTERVENE')                                                      HHL11550
C                                                                       HHL11560
      GO TO 1630                                                        HHL11570
C                                                                       HHL11580
 1410 WRITE(6,1420) NOLD,N,MATOLD,MATNO                                 HHL11590
 1420 FORMAT(/' PARAMETERS READ FROM FILE 3 DO NOT AGREE WITH THOSE SPECHHL11600
     1 IFIED'/' BY THE USER.   NOLD,N,MATOLD,MATNO = '/2I6,2I12/        HHL11610
     1' THEREFORE, PROGRAM TERMINATES FOR USER TO RESOLVE THE DIFFERENCEHHL11620
     1S'/)                                                              HHL11630
C                                                                       HHL11640
      GO TO 1630                                                        HHL11650
C                                                                       HHL11660
 1430 WRITE(6,1440)                                                     HHL11670
 1440 FORMAT(/' PARAMETERS IN ALPHA,BETA FILE READ IN DO NOT AGREE WITH HHL11680
     1 THOSE'/' SPECIFIED BY THE USER.  THEREFORE, THE PROCEDURE TERMINAHHL11690
     1TES'/' FOR THE USER TO RESOLVE THE DIFFERENCES.'/)                HHL11700
C                                                                       HHL11710
      GO TO 1630                                                        HHL11720
C                                                                       HHL11730
 1450 WRITE(6,1460) KMAX,MEV                                            HHL11740
 1460 FORMAT(/' ON ALPHA,BETA HEADER KMAX = ',I6/                       HHL11750
     1' BUT EIGENVALUES WERE COMPUTED AT MEV = ',I6,' PROGRAM STOPS'/)  HHL11760
C                                                                       HHL11770
      GO TO 1630                                                        HHL11780
C                                                                       HHL11790
 1470 WRITE(6,1480)                                                     HHL11800
 1480 FORMAT(/' PROGRAM COMPUTED 1ST GUESSES ON T-MATRIX SIZES, READ THEHHL11810
     1M TO FILE 10'/' THEN TERMINATED AS REQUESTED.')                   HHL11820
      GO TO 1630                                                        HHL11830
C                                                                       HHL11840
 1490 WRITE(6,1500) MTOL, MDIMTV                                        HHL11850
 1500 FORMAT(/' PROGRAM TERMINATES BECAUSE THE TVEC DIMENSION ANTICIPATEHHL11860
     1D',I7/'  IS LARGER THAN THE TVEC DIMENSION',I7,' SPECIFIED BY THE HHL11870
     1USER.'/'  USER MAY RESET THE TVEC DIMENSION AND RESTART THE PROGRAHHL11880
     1M')                                                               HHL11890
      GO TO 1630                                                        HHL11900
C                                                                       HHL11910
 1510 WRITE(6,1520)                                                     HHL11920
 1520 FORMAT(/' PROGRAM TERMINATES BECAUSE NO SUITABLE T-EIGENVECTORS WEHHL11930
     1RE IDENTIFIED'/'  FOR ANY OF THE EIGENVALUES SUPPLIED.  PROBLEM COHHL11940
     1ULD BE CAUSED'/'  BY TOO SMALL A TVEC DIMENSION OR SIMPLY BE THAT HHL11950
     1IT WAS NOT POSSIBLE'/'  TO IDENTIFY T-VECTORS.  USER SHOULD CHECK HHL11960
     1OUTPUT'/)                                                         HHL11970
      GO TO 1630                                                        HHL11980
```

```
C                                                                       HHL11990
 1530 WRITE(6,1540) LVCONT,NTVEC,NGOOD                                  HHL12000
 1540 FORMAT(/' LVCONT FLAG =',I2,' AND NUMBER ',I5,' OF T-EIGENVECTORS HHL12010
     1 COMPUTED N.E.'/' NUMBER',I5,' REQUESTED SO PROGRAM TERMINATES'/) HHL12020
      GO TO 1630                                                        HHL12030
 1550 WRITE(6,1560)                                                     HHL12040
 1560 FORMAT(/' PROGRAM TERMINATES WITHOUT COMPUTING ANY RITZ VECTORS'/ HHL12050
     1 ' BECAUSE ALL T-EIGENVECTORS WERE REJECTED AS NOT SUITABLE'/     HHL12060
     1 ' PROBABLE CAUSE IS LACK OF CONVERGENCE OF THE EIGENVALUES'/)    HHL12070
       GO TO 1630                                                       HHL12080
C                                                                       HHL12090
 1570 WRITE(6,1580)                                                     HHL12100
 1580 FORMAT(/' PROGRAM INDICATES THAT IT IS NOT POSSIBLE TO COMPUTE ANYHHL12110
     1 OF THE'/' REQUESTED EIGENVECTORS. THEREFORE PROGRAM TERMINATES') HHL12120
      DO 1590 J=1,NGOODC                                                HHL12130
 1590 WRITE(6,1600)  J,GOODEV(J),MP(J)                                  HHL12140
 1600 FORMAT(/4X,' J',11X,'GOODEV(J)',4X,'MP(J)'/I6,E20.12,I9)          HHL12150
      GO TO 1630                                                        HHL12160
C                                                                       HHL12170
 1610 WRITE(6,1620) MBETA,KMAXN                                         HHL12180
 1620 FORMAT(/' PROGRAM TERMINATES BECAUSE THE STORAGE ALLOTTED FOR THE HHL12190
     1BETA ARRAY',I8/' IS NOT SUFFICIENT FOR THE ENLARGED KMAX =',I8,' THHL12200
     1HAT THE PROGRAM WANTS'/' USER CAN ENLARGE THE ALPHA AND BETA ARRAYHHL12210
     1S AND RERUN THE PROGRAM.'/)                                       HHL12220
C                                                                       HHL12230
 1630 CONTINUE                                                          HHL12240
C                                                                       HHL12250
      STOP                                                              HHL12260
C-----END OF MAIN PROGRAM FOR LANCZOS HERMITIAN EIGENVECTOR COMPUTATIONSHHL12270
      END                                                               HHL12280
```

3.4 HLEMULT: LANCZS and Sample Matrix-Vector Multiply Subroutines

```
C-----HLEMULT----HERMITIAN MATRICES------------------------------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C            cullumj@lanl.gov
c
c  These codes are copyrighted by the authors.  These codes
c  and modifications of them or portions of them are NOT to be
c  incorporated into any commercial codes without legal agreements
c  with the authors.  If these codes or portions of them
c  are used in other scientific or engineering research works
c  the names of the authors of these codes and appropriate
c  references to their written work are to be incorporated in the
c  derivative works.
c
c  This header is not to be removed from these codes.
C
C        REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4
C        Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C        Applied Mathematics, 2002. SIAM Publications,
C        Philadelphia, PA. USA
C
C     CONTAINS SUBROUTINE LANCZS AND SAMPLE USPEC, CMATV
C     USED BY THE HERMITIAN VERSION OF THE LANCZOS ALGORITHMS
C
C     PORTABILITY:
C     THESE PROGRAMS ARE NOT PORTABLE DUE TO THE USE OF COMPLEX*16
C     VARIABLES.  MOREOVER, THE PFORT VERIFIER IDENTIFIED THE
C     FOLLOWING ADDITIONAL NONPORTABLE CONSTRUCTIONS:
C     1.  THE ENTRY MECHANISM USED TO PASS THE STORAGE
C         LOCATIONS OF THE USER-SPECIFIED MATRIX FROM THE
C         SUBROUTINE USPEC TO THE MATRIX-VECTOR SUBROUTINE CMATV.
C     2.  IN THE PROGRAMS PROVIDED FOR 'HERMITIAN POISSON' TEST MATRICES
C         USPEC CONTAINS FREE FORMAT (8,*), AND FORMAT (20A4); AND
C         EXACT ERROR SUBROUTINE CONTAINS DATA/MACHEP DEFINITION.
C
C
C-----LANCZS-COMPUTE THE LANCZOS TRIDIAGONAL MATRICES-------------------
C
C     GRAM-SCHMIDT ORTHOGONALIZATION WITHOUT MODIFICATION
C     REQUIRES EXTRA VECTOR VS IN LANCZS.  MODIFICATION IS NOT
C     PERMISSIBLE IN THE HERMITIAN CASE BECAUSE COMPLEX PORTION
C     OF THE MODIFICATION COULD NOT BE INCORPORATED.
C
      SUBROUTINE LANCZS(MATVEC,V1,V2,VS,ALPHA,BETA,GR,GC,G,KMAX,MOLD1,N,
     1 IIX)
C
C----------------------------------------------------------------------
      COMPLEX*16 V1(1), V2(1), VS(1), ZEROC, TEMP
```

```
      DOUBLE PRECISION  ALPHA(1), BETA(1), BATA, SUM, ONE, ZERO
      DOUBLE PRECISION GR(1),GC(1)
      REAL  G(1)
      EXTERNAL MATVEC
      DOUBLE PRECISION  DSQRT
C-----------------------------------------------------------------------
C
      ZERO = 0.D0
      ONE = 1.D0
      ZEROC = DCMPLX(ZERO,ZERO)
C
      IF(MOLD1.GT.1)GO TO 50
C
C     ALPHA/BETA GENERATION STARTS AT I = 1
C     MOLD1 = 1 SET V1 = 0. AND V2 = RANDOM UNIT VECTOR
      IIL=IIX
C
C-----------------------------------------------------------------------
      CALL GENRAN(IIL,G,N)
C-----------------------------------------------------------------------
C
      DO 10 I = 1,N
   10 GR(I) = G(I)
C
C-----------------------------------------------------------------------
      CALL GENRAN(IIL,G,N)
C-----------------------------------------------------------------------
C
      DO 20 I = 1,N
   20 GC(I) = G(I)
C
      DO 30 I = 1,N
   30 V2(I) = DCMPLX(GR(I),GC(I))
C
C-----------------------------------------------------------------------
      CALL CINPRD(V2,V2,SUM,N)
C-----------------------------------------------------------------------
C
      SUM = ONE/DSQRT(SUM)
      DO 40 I = 1,N
      V1(I) = ZEROC
   40 V2(I) = V2(I)*SUM
      BETA(1) = ZERO
C
C     ALPHA BETA GENERATION LOOP
   50 CONTINUE
C
      DO 80 I=MOLD1,KMAX
      SUM = ZERO
C
C-----------------------------------------------------------------------
C     MATVEC(V2,VS,SUM) CALCULATES  VS = A*V2 - SUM*VS
      CALL MATVEC(V2,VS,SUM)
      CALL CINPRD(V2,VS,SUM,N)
C-----------------------------------------------------------------------
```

```
C                                                                       HHL00850
      ALPHA(I) = SUM                                                    HHL00860
      BATA = BETA(I)                                                    HHL00870
      DO 60 J=1,N                                                       HHL00880
   60 V1(J) = (VS(J)-BATA*V1(J)) - SUM*V2(J)                            HHL00890
C                                                                       HHL00900
C-----------------------------------------------------------------------HHL00910
      CALL CINPRD(V1,V1,SUM,N)                                          HHL00920
C-----------------------------------------------------------------------HHL00930
C                                                                       HHL00940
      IN = I+1                                                          HHL00950
      BETA(IN) = DSQRT(SUM)                                             HHL00960
      SUM = ONE/BETA(IN)                                                HHL00970
      DO 70 J=1,N                                                       HHL00980
      TEMP = SUM*V1(J)                                                  HHL00990
      V1(J) = V2(J)                                                     HHL01000
   70 V2(J) = TEMP                                                      HHL01010
   80 CONTINUE                                                          HHL01020
C     END ALPHA, BETA GENERATION LOOP                                   HHL01030
C                                                                       HHL01040
C-----END OF LANCZS-----------------------------------------------------HHL01050
C                                                                       HHL01060
      RETURN                                                            HHL01070
      END                                                               HHL01080
C                                                                       HHL01090
C-----USPEC-GENERAL SPARSE, HERMITIAN MATRIX----------------------------HHL01100
C                                                                       HHL01110
C     SUBROUTINE USPEC(N,MATNO)                                         HHL01120
      SUBROUTINE GUSPEC(N,MATNO)                                        HHL01130
C                                                                       HHL01140
C-----------------------------------------------------------------------HHL01150
      COMPLEX*16 A(3000)                                                HHL01160
      DOUBLE PRECISION  AD(1000)                                        HHL01170
      INTEGER  IROW(3000),ICOL(1000)                                    HHL01180
C-----------------------------------------------------------------------HHL01190
C    DIMENSION ARRAYS NEEDED TO DEFINE MATRIX, READ IN VALUES FOR       HHL01200
C    ARRAYS AND THEN PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO      HHL01210
C    THE MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV.                       HHL01220
C                                                                       HHL01230
C    USER-SUPPLIED MATRIX IS STORED IN FOLLOWING SPARSE FORMAT:         HHL01240
C    N = ORDER OF A-MATRIX                                              HHL01250
C    NZS = NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN A                   HHL01260
C    NZL = INDEX OF LAST COLUMN CONTAINING NONZERO SUBDIAGONAL ENTRIES  HHL01270
C    ICOL(J), J=1,NZL IS THE NUMBER OF NONZERO SUBDIAGONAL ELEMENTS     HHL01280
C            IN COLUMN J.                                               HHL01290
C    IROW(K), K = 1,NZS, IS THE ROW INDEX FOR CORRESPONDING A(K).       HHL01300
C    AD(I), I=1,N ARE DIAGONAL ENTRIES (INCLUDING ANY 0 DIAGONAL        HHL01310
C          ENTRIES)                                                     HHL01320
C    A(K), K=1,NZS ARE NONZERO SUBDIAGONAL ENTRIES, LISTED BY COLUMN.   HHL01330
C    FOR J > NZL THERE ARE NO NONZERO SUBDIAGONAL ELEMENTS IN COLUMN J. HHL01340
C    ICOL(J) = 0 IS ALLOWED                                             HHL01350
C                                                                       HHL01360
C-----------------------------------------------------------------------HHL01370
C     IN THIS SAMPLE SUBROUTINE THE ARRAYS ARE READ IN FROM FILE 8      HHL01380
C                                                                       HHL01390
```

```
      READ(8,10) NZS,NOLD,NZL,MATOLD                                    HHL01400
   10 FORMAT(I10,2I6,I8)                                                HHL01410
C                                                                       HHL01420
      WRITE(6,20) NZS,NOLD,NZL,MATOLD                                   HHL01430
   20 FORMAT(I10,2I6,I8,' = NZS,NOLD,NZL,MATOLD'/)                      HHL01440
C                                                                       HHL01450
C     TEST OF PARAMETER CORRECTNESS                                     HHL01460
      ITEMP = (NOLD-N)**2 + (MATNO-MATOLD)**2                           HHL01470
C                                                                       HHL01480
      IF(ITEMP.EQ.0) GO TO 40                                           HHL01490
C                                                                       HHL01500
      WRITE(6,30)                                                       HHL01510
   30 FORMAT(/' PROGRAM TERMINATES BECAUSE EITHER ORDERS OF OR LABELS FOHHL01520
     1R MATRIX DISAGREE'/)                                              HHL01530
      GO TO 80                                                          HHL01540
C                                                                       HHL01550
   40 CONTINUE                                                          HHL01560
C                                                                       HHL01570
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS READ      HHL01580
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS READ      HHL01590
      READ(8,50) (ICOL(K), K=1,NZL)                                     HHL01600
      READ(8,50) (IROW(K), K=1,NZS)                                     HHL01610
   50 FORMAT(13I6)                                                      HHL01620
C     DIAGONAL IS READ FIRST, THEN NONZERO BELOW DIAGONAL ENTRIES       HHL01630
      READ(8,60) (AD(K), K=1,N)                                         HHL01640
   60 FORMAT(4E20.12)                                                   HHL01650
      READ(8,70) (A(K), K=1,NZS)                                        HHL01660
C  50 FORMAT(4Z20)                                                      HHL01670
   70 FORMAT(4E20.12)                                                   HHL01680
C                                                                       HHL01690
C-----------------------------------------------------------------------HHL01700
C     PASS STORAGE LOCATIONS OF ARRAYS THAT DEFINE THE MATRIX TO        HHL01710
C     THE MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV                       HHL01720
      CALL CMATVE(A,AD,ICOL,IROW,N,NZL)                                 HHL01730
C-----------------------------------------------------------------------HHL01740
C                                                                       HHL01750
      RETURN                                                            HHL01760
   80 STOP                                                              HHL01770
C                                                                       HHL01780
C-----END OF USPEC FOR GENERAL, SPARSE HERMITIAN MATRICES---------------HHL01790
      END                                                               HHL01800
C                                                                       HHL01810
C-----START OF MATRIX-VECTOR MULTIPLY-GENERAL SPARSE HERMITIAN--------- HHL01820
C                                                                       HHL01830
C     SUBROUTINE CMATV(W,U,SUM)                                         HHL01840
      SUBROUTINE GCMATV(W,U,SUM)                                        HHL01850
C                                                                       HHL01860
C-----------------------------------------------------------------------HHL01870
      COMPLEX*16  U(1),W(1),A(1)                                        HHL01880
      DOUBLE PRECISION  AD(1),SUM                                       HHL01890
      INTEGER  IROW(1),ICOL(1)                                          HHL01900
C-----------------------------------------------------------------------HHL01910
C    SPARSE MATRIX-VECTOR MULTIPLY FOR LANCZS  U = A*W - SUM*U          HHL01920
C    SEE USPEC SUBROUTINE FOR DESCRIPTION OF THE ARRAYS THAT DEFINE     HHL01930
C    THE MATRIX                                                         HHL01940
```

```
C                                                                       HHL01950
      GO TO 3                                                           HHL01960
C                                                                       HHL01970
C-----------------------------------------------------------------------HHL01980
C     STORAGE LOCATIONS OF ARRAYS ARE PASSED TO CMATV FROM USPEC        HHL01990
      ENTRY CMATVE(A,AD,ICOL,IROW,N,NZL)                                HHL02000
C-----------------------------------------------------------------------HHL02010
C                                                                       HHL02020
      GO TO 4                                                           HHL02030
    3 CONTINUE                                                          HHL02040
C                                                                       HHL02050
C     COMPUTE THE DIAGONAL TERMS                                        HHL02060
      DO 10 I = 1,N                                                     HHL02070
   10 U(I) = AD(I)*W(I)-SUM*U(I)                                        HHL02080
C                                                                       HHL02090
C     COMPUTE BY COLUMN                                                 HHL02100
      LLAST = 0                                                         HHL02110
      DO 30 J = 1,NZL                                                   HHL02120
C                                                                       HHL02130
      IF (ICOL(J).EQ.0) GO TO 30                                        HHL02140
      LFIRST = LLAST + 1                                                HHL02150
      LLAST = LLAST + ICOL(J)                                           HHL02160
C                                                                       HHL02170
      DO 20 L = LFIRST,LLAST                                            HHL02180
      I = IROW(L)                                                       HHL02190
C                                                                       HHL02200
      U(I) = U(I) + A(L)*W(J)                                           HHL02210
      U(J) = U(J) + DCONJG(A(L))*W(I)                                   HHL02220
C                                                                       HHL02230
   20 CONTINUE                                                          HHL02240
C                                                                       HHL02250
   30 CONTINUE                                                          HHL02260
C                                                                       HHL02270
    4 RETURN                                                            HHL02280
C                                                                       HHL02290
C-----END OF CMATV-GENERAL, SPARSE, HERMITIAN MATRICES -----------------HHL02300
      END                                                               HHL02310
C                                                                       HHL02320
C-----USPEC, CMATV, EXEVG, AND HEXVEC FOR HERMITIAN 'POISSON' MATRICES--HHL02330
C                                                                       HHL02340
C-----USPEC (HERMITIAN POISSON MATRICES)--------------------------------HHL02350
C                                                                       HHL02360
      SUBROUTINE HUSPEC(N,MATNO)                                        HHL02370
C     SUBROUTINE USPEC(N,MATNO)                                         HHL02380
C                                                                       HHL02390
C-----------------------------------------------------------------------HHL02400
      DOUBLE PRECISION  C0,C1,C2,HALF,ONE,SCR,SCI,ANGLE,TEMP            HHL02410
      COMPLEX*16 SC,TC,CL0,CL1,CL3,CL4                                  HHL02420
      REAL EXPLAN(20)                                                   HHL02430
      DOUBLE PRECISION EIGVAL(1000)                                     HHL02440
      REAL GAPS(1000)                                                   HHL02450
      INTEGER MULTS(1000)                                               HHL02460
C-----------------------------------------------------------------------HHL02470
      HALF = 0.5D0                                                      HHL02480
      ONE  = 1.0D0                                                      HHL02490
```

```
C
C     READ IN PARAMETERS TO DEFINE MATRIX
C     MATRIX IS COMPLEX DIAGONAL SIMILITARY TRANSFORM OF REAL SYMMETRIC
C     POISSON MATRIX WHICH HAS SYMMETRIC TOEPLITZ BLOCKS ALONG
C     THE DIAGONAL, EACH ONE OF WHICH HAS THE PARAMETER C2 ALONG THE
C     DIAGONAL AND -C0 ABOVE AND BELOW THE DIAGONAL, AND OFF-DIAGONAL
C     BLOCKS THAT ARE DIAGONAL WITH DIAGONAL ENTRIES -C1.  EACH BLOCK
C     IS KX*KX AND THERE ARE KY BLOCKS.  THE HERMITIAN VERSION IS
C     OBTAINED BY APPLYING A DIAGONAL SIMILARITY TRANSFORM TO THE
C     REAL MATRIX WHERE THIS TRANSFORMATION IS SUCH THAT ITS
C     DIAGONAL ENTRIES ARE (SC)**(K-1), K = 1,...,N, WHERE SC
C     HAS MODULUS 1.
C
      READ(8,10) EXPLAN
      WRITE(6,10) EXPLAN
      READ(8,10) EXPLAN
   10 FORMAT(20A4)
C     IF MTYPE = 0 WE HAVE ZERO BOUNDARY CONDITIONS
C     IF MTYPE = 1 WE HAVE NORMAL DERIVATIVE BOUNDARY CONDITIONS
C     NOTE THAT SUBROUTINES EXEVG AND HEXVEC ARE VALID ONLY FOR
C     MTYPE = 0.
      READ(8,*) NOLD,MATOLD,IVEC,MTYPE
      WRITE(6,20) NOLD,MATOLD
   20 FORMAT(' ORDER OF MATRIX READ FROM FILE =',I6/' MATRIX NUMBER =',
     1I8/)
      IF(MTYPE.EQ.0) WRITE(6,30)
   30 FORMAT(/' HERMITIAN POISSON CORRESPONDING TO ZERO BOUNDARY CONDITI
     1ONS'/)
      IF(MTYPE.EQ.1) WRITE(6,40)
   40 FORMAT(/' HERMITIAN POISSON CORRESPONDING TO NORMAL DERIVATIVE BOU
     1NDARY CONDITIONS'/)
      IF(IVEC.NE.0.AND.MTYPE.EQ.0) WRITE(6,50)
   50 FORMAT(' COMPUTE THE TRUE EIGENVALUES AND PUT IN FN TRUEEVAL'/)
C
C     TEST OF PARAMETER CORRECTNESS
      ITEMP = (NOLD-N)**2 + (MATNO-MATOLD)**2
C
      IF(ITEMP.EQ.0) GO TO 70
C
      WRITE(6,60)
   60 FORMAT(' PROGRAM TERMINATES BECAUSE EITHER ORDERS OF OR LABELS FOR
     1 MATRIX DISAGREE')
      GO TO 150
C
   70 CONTINUE
C
      READ(8,10) EXPLAN
      READ(8,*)  C0,KX,KY
      IF (KX.GT.4.AND.KY.GT.4) GO TO 90
      WRITE(6,80) KX,KY
   80 FORMAT(2I6,' = KX KY  ONE OR BOTH OF KX KY TOO SMALL SO STOP'/)
      GO TO 150
   90 CONTINUE
      READ(8,10) EXPLAN
C     BELOW SC = COS(ANGLE) + I SIN(ANGLE)
```

```
C     READ IN DESIRED COSINE, COMPUTE ANGLE, THEN SINE
      READ(8,*) SCR
      ANGLE = DACOS(SCR)
      SCI = DSIN(ANGLE)
      SC = DCMPLX(SCR,SCI)
      WRITE(6,100) SC
C     IF (IVEC.NE.0.AND.MTYPE.EQ.0) WRITE(9,7) SC
  100 FORMAT(' GENERATOR OF DIAGONAL TRANSFORMATION ='/2E20.12)
C
      TC = SC
      DO 110 J=2,KX
  110 TC = SC*TC
      WRITE(6,120) TC
  120 FORMAT(' TC = ',2E20.12)
C
      N = KX*KY
      C2 = ONE
      C1 = HALF-C0
      TEMP = DSQRT(2.0D0)
      IF (MTYPE.EQ.0) TEMP = ONE
      CL0 = -SC*C0
      CL1 = -TC*C1
      CL3 = -SC*C0*TEMP
      CL4 = -TC*C1*TEMP
C
      WRITE(6,130) N,MTYPE,KX,KY,C2,C0,C1
  130 FORMAT(/5X,'N',1X,'MTYPE',4X,'KX',4X,'KY',7X,'DIAGONAL',
     1 3X,'X-CODIAGONAL',3X,'Y-CODIAGONAL'/4I6,3E15.8/)
C
C-----------------------------------------------------------------------
       CALL HMATVE(C2,CL0,CL1,CL3,CL4,KX,KY)
C-----------------------------------------------------------------------
C
      IF(IVEC.EQ.0.OR.MTYPE.NE.0) GO TO 140
C
C     COMPUTE THE EXACT EIGENVALUES
C
C-----------------------------------------------------------------------
      CALL EXEVG(EIGVAL,C0,C1,C2,GAPS,MULTS,KX,KY)
C-----------------------------------------------------------------------
C
      IF(IVEC.LT.0) GO TO 150
C
  140 CONTINUE
      RETURN
C
C-----END OF USPEC------------------------------------------------------
  150 STOP
      END
C
C-----START OF CMATV FOR HERMITIAN POISSON MATRICES---------------------
C
      SUBROUTINE HMATV(W,U,SUM)
C     SUBROUTINE CMATV(W,U,SUM)
C
```

```
C-----------------------------------------------------------------------
      DOUBLE PRECISION  C2,SUM
      COMPLEX*16 U(1),W(1)
      COMPLEX*16 CL0,CL1,CL3,CL4,CR0,CR1,CR3,CR4
C-----------------------------------------------------------------------
C     CALCULATES U = A*W - SUM*U
C
      GO TO 3
C
      ENTRY HMATVE(C2,CL0,CL1,CL3,CL4,KK,LL)
C
      GO TO 4
C
    3 CONTINUE
C
      N = KK*LL
      CR0 = DCONJG(CL0)
      CR1 = DCONJG(CL1)
      CR3 = DCONJG(CL3)
      CR4 = DCONJG(CL4)
C
C-----------------------------------------------------------------------
C    FIRST AND LAST BLOCKS
      J = 1
      U(J)=(C2*W(J)+CR3*W(J+1)+CR1*W(J+KK)) - SUM*U(J)
      J = 2
      U(J)=(C2*W(J)+CL3*W(J-1)+CR0*W(J+1)+CR1*W(J+KK))-SUM*U(J)
      J = KK
      U(J)=(C2*W(J)+CL3*W(J-1)+CR1*W(J+KK))-SUM*U(J)
      J = KK - 1
      U(J)=(C2*W(J)+CR3*W(J+1)+CL0*W(J-1)+CR1*W(J+KK))-SUM*U(J)
      J = N - KK + 1
      U(J)=(C2*W(J)+CR3*W(J+1)+CL4*W(J-KK))-SUM*U(J)
      J = N - KK + 2
      U(J)=(C2*W(J)+CL3*W(J-1)+CR0*W(J+1)+CL4*W(J-KK))-SUM*U(J)
      J = N
      U(J)=(C2*W(J)+CL3*W(J-1)+CL4*W(J-KK))-SUM*U(J)
      J = N - 1
      U(J)=(C2*W(J)+CL0*W(J-1)+CR3*W(J+1)+CL4*W(J-KK))-SUM*U(J)
C
      KK2 = KK - 2
      DO 10 JJ = 3,KK2
      J = JJ
      U(J)=(C2*W(J)+CL0*W(J-1)+CR0*W(J+1)+CR1*W(J+KK))-SUM*U(J)
      J = N - KK + JJ
   10 U(J)=(C2*W(J)+CL0*W(J-1)+CR0*W(J+1)+CL4*W(J-KK))-SUM*U(J)
C
C    START BLOCKS 2 AND LL-1
      J = KK + 1
      U(J)=(C2*W(J)+CR3*W(J+1)+CL1*W(J-KK)+CR1*W(J+KK))-SUM*U(J)
      J = KK + 2
      U(J)=(C2*W(J)+CL3*W(J-1)+CR0*W(J+1)+CL1*W(J-KK)+CR1*W(J+KK))
     1 -SUM*U(J)
      J = KK + KK
      U(J)=(C2*W(J)+CL3*W(J-1)+CL1*W(J-KK)+CR1*W(J+KK))-SUM*U(J)
```

```
      J = KK + KK - 1
      U(J)=(C2*W(J)+CR3*W(J+1)+CL0*W(J-1)+CL1*W(J-KK)+CR1*W(J+KK))
     1  -SUM*U(J)
      J = N - 2*KK + 1
      U(J)=(C2*W(J)+CR3*W(J+1)+CR4*W(J+KK)+CL1*W(J-KK))
     1 -SUM*U(J)
      J = N - 2*KK + 2
      U(J)=(C2*W(J)+CL3*W(J-1)+CR0*W(J+1)+CR4*W(J+KK)+CL1*W(J-KK))
     1 -SUM*U(J)
      J = N - KK
      U(J)=(C2*W(J)+CL3*W(J-1)+CR4*W(J+KK)+CL1*W(J-KK))-SUM*U(J)
      J = N - KK - 1
      U(J)=(C2*W(J)+CR3*W(J+1)+CL0*W(J-1)+CR4*W(J+KK)+CL1*W(J-KK))
     1 -SUM*U(J)
C
      DO 20 JJ = 3,KK2
      J = KK + JJ
      U(J)=(C2*W(J)+CL0*W(J-1)+CR0*W(J+1)+CL1*W(J-KK)+CR1*W(J+KK))
     1 -SUM*U(J)
      J = N - 2*KK + JJ
      U(J)=(C2*W(J)+CL0*W(J-1)+CR0*W(J+1)+CR4*W(J+KK)+CL1*W(J-KK))
     1 -SUM*U(J)
   20 CONTINUE
C
C    MIDDLE BLOCKS
      LL2 = LL - 2
      JP = KK
      DO 40 JJ = 3,LL2
      JP = JP + KK
C     JP = (JJ-1)*KK
      J = JP + 1
      U(J)=(C2*W(J)+CR3*W(J+1)+CL1*W(J-KK)+CR1*W(J+KK))-SUM*U(J)
      J = J + 1
      U(J)=(C2*W(J)+CL3*W(J-1)+CR0*W(J+1)+CL1*W(J-KK)+
     1 CR1*W(J+KK))-SUM*U(J)
      J = J + KK - 2
      U(J) = (C2*W(J)+CL3*W(J-1)+CL1*W(J-KK)+CR1*W(J+KK))-SUM*U(J)
      J = J - 1
      U(J)=(C2*W(J)+CR3*W(J+1)+CL0*W(J-1)+CL1*W(J-KK)+
     1 CR1*W(J+KK))-SUM*U(J)
C
      DO 30 II = 3,KK2
      J = JP + II
      U(J)=(C2*W(J)+CL0*W(J-1)+CR0*W(J+1)+CL1*W(J-KK)+CR1*W(J+KK))
     1 -SUM*U(J)
   30 CONTINUE
C
   40 CONTINUE
C
    4 RETURN
C
C-----END OF HMATV-----------------------------------------------------
      END
C
C-----START OF EXEVG---------------------------------------------------
```

```
C
C     FOR MTYPE = 0, ZERO BOUNDARY CONDITIONS:
C     COMPUTES EXACT EIGENVALUES OF HERMITIAN POISSON MATRIX,
C     THEIR MULTIPLICITIES, AND THE GAPS BETWEEN THE EIGENVALUES AND
C     PUTS THEM RESPECTIVELY INTO VECTORS U, MP, AND G.  THESE
C     QUANTITIES ARE ALL WRITTEN TO FILE 9.
C
      SUBROUTINE EXEVG(U,C0,C1,C2,G,MP,KX,KY)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  U(*),MACHEP
      DOUBLE PRECISION  EPSM,C0,C1,C2,T0,T1,PIK,PIL,ONE,TWO,ATOLN,EE
      REAL G(1)
      INTEGER MP(1)
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.0D0*MACHEP
C-----------------------------------------------------------------------
      N = KX*KY
      ONE  = 1.0D0
      TWO  = 2.0D0
      T0 = DACOS(-ONE)
      T1 = DFLOAT(KX+1)
      PIK = T0/T1
      T1 = DFLOAT(KY+1)
      PIL = T0/T1
C     GENERATE EXACT EIGENVALUES
      KP = 0
      DO 20 J = 1,KY
      T1 = PIL*DFLOAT(J)
      T0 = C2 - TWO*C1*DCOS(T1)
      DO 10 I = 1,KX
      KP = KP+1
      T1 = PIK*DFLOAT(I)
   10 U(KP) = T0 - TWO*C0*DCOS(T1)
   20 CONTINUE
C
C     ORDER U VECTOR BY INCREASING ALGEBRAIC SIZE
      DO 40 K = 2,N
      KM1 = K-1
      DO 30 L = 1,KM1
      JJ = K-L
      IF (U(JJ+1).GE.U(JJ)) GO TO 40
      T0 = U(JJ)
      U(JJ) = U(JJ+1)
   30 U(JJ+1) = T0
   40 CONTINUE
      ATOLN = DMAX1(DABS(U(1)),DABS(U(N)))*EPSM
C
      WRITE(9,50)
   50 FORMAT(' TRUE EIGENVALUES FOR HERMITIAN POISSON')
C
      WRITE(9,60)N,KX,KY,C2,C0,C1,ATOLN
      WRITE(6,60) N,KX,KY,C2,C0,C1,ATOLN
   60 FORMAT(1X,'A-SIZE',2X,'X-DIM',2X,'Y-DIM'/3I7/
```

```
     1 5X,'A-DIAGONAL',3X,'X-CODIAGONAL',3X,'Y-CODIAGONAL',10X,'ATOLN'/ HHL05250
     2 4E15.8)                                                          HHL05260
C                                                                       HHL05270
C     DETERMINE TRUE MULTIPLICITIES FOR EXACT EIGENVALUES               HHL05280
      I = 1                                                             HHL05290
      IDEX = 1                                                          HHL05300
      J = 1                                                             HHL05310
      NEXACT = 0                                                        HHL05320
   70 J = J+1                                                           HHL05330
      IF (J.GT.N) GO TO 80                                              HHL05340
      EE = DABS(U(J)-U(I))                                              HHL05350
      IF (EE.GT.ATOLN) GO TO 80                                         HHL05360
      IDEX = IDEX+1                                                     HHL05370
      GO TO 70                                                          HHL05380
   80 NEXACT = NEXACT+1                                                 HHL05390
      U(NEXACT) = U(I)                                                  HHL05400
      MP(NEXACT) = IDEX                                                 HHL05410
C     MP(K) = MULTIPLICITY OF KTH EIGENVALUE CLUSTER FOR A              HHL05420
      IDEX = 1                                                          HHL05430
      I = J                                                             HHL05440
      IF (I.GT.N) GO TO 90                                              HHL05450
      GO TO 70                                                          HHL05460
   90 CONTINUE                                                          HHL05470
C                                                                       HHL05480
C     MULTIPLICITIES HAVE BEEN DETERMINED                               HHL05490
C     NEXACT = NUMBER OF DISTINCT A-EIGENVALUES                         HHL05500
C                                                                       HHL05510
      WRITE(9,100)NEXACT                                                HHL05520
      WRITE(6,100)NEXACT                                                HHL05530
  100 FORMAT(I6,' = NUMBER OF TRUE A-EIGENVALUES WHICH ARE DISTINCT'/)  HHL05540
C                                                                       HHL05550
C     MINGAP CALCULATION FOR DISTINCT A-EIGENVALUES                     HHL05560
      NM1 = NEXACT - 1                                                  HHL05570
      G(NEXACT) = U(NM1)-U(NEXACT)                                      HHL05580
      G(1) = U(2)-U(1)                                                  HHL05590
C                                                                       HHL05600
      DO 110 J = 2,NM1                                                  HHL05610
      T0 = U(J)-U(J-1)                                                  HHL05620
      T1 = U(J+1)-U(J)                                                  HHL05630
      G(J) = T1                                                         HHL05640
      IF (T0.LT.T1) G(J) = -T0                                          HHL05650
  110 CONTINUE                                                          HHL05660
C                                                                       HHL05670
C     NEXACT DISTINCT A-EIGENVALUES ARE IN U IN ASCENDING ORDER         HHL05680
C     MP = MULTIPLICITIES OF THE DISTINCT EIGENVALUES OF A              HHL05690
C     G = TRUE MINIMUM GAP IN A FOR EACH OF THESE EIGENVALUES           HHL05700
C     G < 0 INDICATES THE LEFT-HAND GAP WAS MINIMAL.                    HHL05710
C     OUTPUT MULTIPLICITIES, DISTINCT EVS, AND MINGAPS TO FILE 11       HHL05720
C                                                                       HHL05730
      WRITE(9,120)                                                      HHL05740
  120 FORMAT(5X,'I',1X,'AMULT',5X,'TRUE A-EIGENVALUE(I)',               HHL05750
     1 3X,'A-MINGAP(I)')                                                HHL05760
C                                                                       HHL05770
      WRITE(9,130)(J,MP(J),U(J),G(J), J=1,NEXACT)                       HHL05780
  130 FORMAT(2I6,E25.16,E14.3)                                          HHL05790
```

```
C
      WRITE(9,140)
  140 FORMAT(' NEXACT DISTINCT A-EIGENVALUES ARE IN ASCENDING ORDER'/
     1 ' AMULT = MULTIPLICITIES OF THE DISTINCT EIGENVALUES OF A.'/
     2 ' A-MINGAP(I) = TRUE MINIMUM GAP IN A FOR EACH EIGENVALUE.'/
     3 ' A-MINGAP(I) LT 0 INDICATES THE LEFT-HAND GAP WAS MINIMAL.'//)
C
C     WE ORDER U VECTOR BY INCREASING SIZE OF THE GAPS
C
      DO 150 K = 1,N
  150 MP(K) = K
C
      DO 170 K = 2,N
      KM1 = K-1
C
      DO 160 L = 1,KM1
      JJ = K - L
      IF (ABS(G(JJ+1)).GE.ABS(G(JJ))) GO TO 170
      EE = U(JJ)
      U(JJ) = U(JJ+1)
      U(JJ+1) = EE
      GG = G(JJ)
      G(JJ) = G(JJ+1)
      G(JJ+1) = GG
      IEE = MP(JJ)
      MP(JJ) = MP(JJ+1)
  160 MP(JJ+1) = IEE
C
  170 CONTINUE
C
      WRITE(9,180)
  180 FORMAT(5X,'K',6X,'A-MINGAP',5X,'TRUE A-EIGENVALUE(I)',2X,'A-EVNO')
C
      WRITE(9,190)(J,G(J),U(J),MP(J), J=1,NEXACT)
  190 FORMAT(I6,E14.3,E25.16,I8)
C
      WRITE(9,200)
  200 FORMAT(' NEXACT DISTINCT A-EIGENVALUES. GAPS IN ASCENDING ORDER'/
     2 ' A-MINGAP(I) = TRUE MINIMUM GAP IN A FOR EACH EIGENVALUE.'/
     3 ' A-MINGAP(I) LT 0 INDICATES THE LEFT-HAND GAP WAS MINIMAL.'/
     3 ' A-MATRIX IS BLOCK TRIDIAGONAL AND EACH DIAGONAL BLOCK IS OF ORD
     3ER NX.'/
     4 ' NX = NUMBER OF POINTS ON EACH X-LINE. THERE ARE NY DIAGONAL BLO
     4CKS.'/
     5 ' NY = NUMBER OF POINTS ON EACH Y-LINE.'/
     5 ' A-DIAGONAL   = A(K,K)'/
     6 ' X-CODIAGONAL = A(I,I+1)'/
     7 ' Y-CODIAGONAL = A(I,I+NX)'/
     8 ' ----- END OF FILE 9 EXACTEV-----------------------------'//)
C
C-----END OF EXEVG-------------------------------------------------------
C
      RETURN
      END
C
```

```
C-----START OF HEXVEC-------------------------------------------------
C
C     FOR THE HERMITIAN POISSON TEST CASES WITH MTYPE = 0 ONLY:
C     FOR A GIVEN RITZ VECTOR V AND EIGENVALUE X1, COMPUTES
C     THE CLOSEST TRUE EIGENVALUE Y1 AND CORRESPONDING TRUE
C     EIGENVECTOR Z, CALCULATES THE NORM OF V-Z AND THE MAXIMAL
C     DIFFERENCE OF THE COMPONENTS.  USER WOULD HAVE TO
C     INCORPORATE ENTRY AND CALL TO THIS SUBROUTINE INTO
C     HLEVEC PROGRAM IF THESE QUANTITIES ARE DESIRED.
C     U CONTAINS THE COMPUTED TRUE EIGENVALUES.
C     W CONTAINS THE TRUE EIGENVECTOR FOR THE REAL POISSON MATRIX
C
      SUBROUTINE HEXVEC(Z,V,U,W,X1,Y1,MP,JNUM)
C
C----------------------------------------------------------------------
      DOUBLE PRECISION  U(*),W(*)
      DOUBLE PRECISION WI(110),WJ(110),WII(110)
      DOUBLE PRECISION X1,Y1,EV,EE,WS,PIK,PIL,SUM,TEMP
      DOUBLE PRECISION ATOLN,EPSM,ZERO,HALF,ONE,TWO,MACHEP
      DOUBLE PRECISION C0,C1,C2,T0,T1,T2
      COMPLEX*16 CONE,S,SB,STEMP,V(1),Z(1)
      INTEGER MP(1)
C----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.0D0*MACHEP
C----------------------------------------------------------------------
C     THIS PROGRAM CALCULATES THE EXACT EIGENVALUES AND EIGENVECTORS
C     OF THE HERMITIAN POISSON MATRIX A OF ORDER  N = KX BY KY
C     A CONSISTS OF KY TRIDIAGONAL BLOCKS OF ORDER KX
C     KX = X-DIMENSION     KY = Y-DIMENSION.
C
C     C2 = DIAGONAL OF KX BY KX MATRIX
C    -C0 = CO-DIAGONAL OF THE KX BY KX MATRIX.
C    -C1 = Y-CODIAGONAL.
C
C     NOTE THAT THE VECTORS WI,WJ,WII ARE DIMENSIONED INTERNALLY
C     THEY ARE USED JUST TO KEEP FROM REGENERATING INFORMATION.
C     WI,WII = REAL*8 ARRAYS OF DIMENSION AT LEAST KX
C     WJ     = REAL*8 ARRAY  OF DIMENSION AT LEAST KY.
C
C     NOTATION USED IN PROGRAM
C
C     PIK = ARCOS(-1)/(KX+1)    PIL = ARCOS(-1)/(KY+1)
C     WI(I) = PIK*I        WJ(J) = PIL*J
C
C     T0 = C2 - 2*C1*COS(PIL*J)    EV(I,J) = T0 - 2*C0*COS(PIK*I)
C     I = 1,KX    J = 1,KY    KP = (J-1)*KX + I
C
C     W(KV) = SIN(PIK*I*IK)*SIN(PIL*J*JK)
C     IK = 1,KX    JK = 1,KY    KV = (JK-1)*KX + IK
C     W IS UNSCALED EIGENVECTOR FOR EV(I,J)
C     WS = 1/||W||: ||W|| = .5*DSQRT(T2*T3)  T2 = KX+1  T3 = KY+1
C     U(K) IS A-EV ORDERED BY INCREASING SIZE, K = 1,N
C
C     GIVEN X1 FIND Y1 AND KVEC SUCH THAT
```

```
C           Y1 = EV(KVEC) AND |X1-Y1| = MIN
C           ALSO GIVEN UNIT RITZ VECTOR ASSOCIATED WITH X1
C           CALCULATE UNIT EIGENVECTOR W, A*W = Y1*W
C           T2 = ||V-W||  T1 = MAX(|V(K)-W(K)|, K= 1,N)
C           MAX OCCURS FIRST AT K = KK
C
C-----------------------------------------------------------------------
C     C2 = A(K,K)
C     C0 = A(K,K+1) = A(K+1,K)
C     C1 = A(K,K+KX) = A(K+KX,K)
C     C0 + C1 = HALF
C
      GO TO 3
C
C-----------------------------------------------------------------------
      ENTRY EXVECP(SB,C0,C1,C2,KX,KY)
C-----------------------------------------------------------------------
      GO TO 4
C
    3 CONTINUE
C
C     SPECIFY PARAMETERS
      N = KX*KY
      ZERO = 0.0D0
      HALF = 0.5D0
      ONE  = 1.0D0
      TWO  = 2.0D0
      T0 = DACOS(-ONE)
      T1 = DFLOAT(KX+1)
      PIK = T0/T1
      T2 = DFLOAT(KY+1)
      PIL = T0/T2
      WS = TWO/DSQRT(T1*T2)
C
C     GENERATE WI WJ VECTORS
      KP = 0
      DO 20 J = 1,KY
      T1 = PIL*DFLOAT(J)
      WJ(J) = T1
      T0 = C2 - TWO*C1*DCOS(T1)
      DO 10 I = 1,KX
      KP = KP+1
      T1 = PIK*DFLOAT(I)
      WI(I) = T1
   10 U(KP) = T0 - TWO*C0*DCOS(T1)
   20 CONTINUE
C     U(KP) = EV(I,J) = C2 - 2*C1*COS(PIL*J) - 2*C0*COS(PIK*I)
C
C     INITIALIZE MP VECTOR
      DO 30 K = 1,N
   30 MP(K) = K
C
C     WE ORDER U VECTOR BY INCREASING SIZE OF THE EVS
      DO 50 K = 2,N
      KM1 = K-1
```

```
C
      DO 40 L = 1,KM1
      JJ = K - L
      IF (U(JJ+1).GE.U(JJ)) GO TO 50
      EE = U(JJ)
      U(JJ) = U(JJ+1)
      U(JJ+1) = EE
      IEE = MP(JJ)
      MP(JJ) = MP(JJ+1)
   40 MP(JJ+1) = IEE
C
   50 CONTINUE
C
      ATOLN = DMAX1(DABS(U(1)),DABS(U(N)))*EPSM
C
      WRITE(6,60) N,KX,KY,C2,C0,C1,ATOLN
   60 FORMAT(/' EXACT ERRORS FOR CONVERGED GOODEV'/
     1 4I6,' = N KX KY'//
     1 4E12.5,' = C2 C0 C1 ATOLN'//)
C
C     KP = MP(K) MEANS EIGENVALUE U(K) CORRESPONDS TO EIGENVECTOR W(KP)
C     COMPUTE TOLERANCE USED IN COMPUTING TRUE MULTIPLICITIES
C
C     X1 IS AN INPUT PARAMETER. WE CALCULATE EXACT
C     A-EIGENVALUE WHICH IS CLOSEST TO X1, LABEL IT Y1 AND CALCULATE
C     UNIT EIGENVECTOR OF A ASSOCIATED WITH Y1. A*W = Y1*W, ||W|| = 1.
C     Y1 = U(KEV). EIGENVALUES OF A ARE ORDERED BY INCREASING SIZE.
C     V = COMPLEX RITZ VECTOR ASSOCIATED WITH GOODEV X1
C     WE SHOULD HAVE V = D*W WHERE D = DIAG(D(1),D(2),..,D(N))
C     D(1) = ONE, D(K+1)/D(K) = SB, |SB| = ONE
C
      KX1 = 0
      IF (X1.LE.U(1)) KX1 = 1
      IF (X1.GE.U(N)) KX1 = N
      NM1 = N-1
      IF (KX1.NE.0) GO TO 80
C
      DO 70 KVEC = 2,N
      IF (X1.GE.U(KVEC)) GO TO 70
C     U(KVEC-1).LE.X1.LT.U(KVEC)
      T1 = X1 - U(KVEC-1)
      T2 = U(KVEC) - X1
      KX1 = KVEC - 1
      IF (T1.GT.T2) KX1 = KVEC
      GO TO 80
   70 CONTINUE
C
   80 Y1 = U(KX1)
C
      IF (KX1.EQ.1) EE = U(2) - U(1)
      IF (KX1.EQ.N) EE = U(N) - U(NM1)
      IF (KX1.EQ.1.OR.KX1.EQ.N) GO TO 90
      EE = DMIN1(U(KX1+1)-U(KX1),U(KX1)-U(KX1-1))
   90 CONTINUE
C
```

```
      T0 = DABS(ONE - X1/Y1)
C
      WRITE(6,100) N,KX1,JNUM,Y1,X1,T0,EE
  100 FORMAT(3I8,' = N, A-EV NUMBER,GOODEV NO'//
     1 18X,'EXACTEV',19X,'GOODEV',4X,'RELERROR',4X,'A-MINGAP'/
     1 2E25.16,2E12.3/)
C
      IF (EE.GT.ATOLN) GO TO 120
C
      WRITE(6,110)
  110 FORMAT(' Y1 IS A MULTIPLE EIGENVALUE OF A SO WE EXIT'/)
C
      GO TO 200
C
C     Y1 IS TOEPLITZ EIGENVALUE CLOSEST TO X1.
C     CALCULATION OF EIGENVECTOR ASSOCIATED WITH EIGENVALUE Y1
C     A*W = Y1*W
C
C     DETERMINE I J FROM K: MP(K) = KP = (J-1)*KX+I
  120 CONTINUE
      K = KX1
      KP = MP(K)
      I = MOD(KP,KX)
      IF (I.EQ.0) I = KX
      T1 = WI(I)
      J = 1 + (KP-1)/KX
      T2 = WJ(J)
C
      DO 130 II = 1,KX
      T0 = T1*DFLOAT(II)
  130 WII(II) = WS*DSIN(T0)
C
      KV = 0
      DO 150 JJ = 1,KY
      T0 = T2*DFLOAT(JJ)
      T0 = DSIN(T0)
C
      DO 140 II = 1,KX
      KV = KV + 1
  140 W(KV) = T0*WII(II)
C
  150 CONTINUE
C
C     W IS UNIT EXACT EIGENVECTOR OF A ASSOCIATED WITH Y1
C     V IS UNIT COMPLEX RITZVECTOR OF B ASSOCIATED WITH X1
C
      CONE = DCMPLX(ONE,ZERO)
      STEMP = CONE
      DO 160 K = 1,N
      Z(K) = STEMP*W(K)
  160 STEMP = STEMP*SB
C
      T1 = ZERO
      S = ONE
      KK = 0
```

```
      DO 170 K = 1,N
      IF (CDABS(Z(K)).LE.T1) GO TO 170
      T1 = CDABS(Z(K))
      KK = K
  170 CONTINUE
C
      S = V(KK)/Z(KK)
C
      KK = 0
      T1 = ZERO
      T2 = ZERO
      DO 180 K = 1,N
      TEMP = CDABS(S*Z(K) - V(K))
      T2 = T2 + TEMP**2
      IF (TEMP.LE.T1) GO TO 180
      KK = K
      T1 = TEMP
  180 CONTINUE
C
      T2 = DSQRT(T2)
      WRITE(6,190) KK,T1,T2
  190 FORMAT(' EIGENVECTOR ERROR. MAX ERROR AT COMPONENT = ',I6/
     1 ' MAX CDABS(EXACTVEC(K)-RITZVEC(K)) = ',E12.5/
     1 ' NORM(EXACTVEC-RITZVEC) = ',E12.5/)
C
  200 CONTINUE
C
C-----END OF HEXVEC---------------------------------------------------
    4 RETURN
      END
C
C-----USPEC (TRIDIAGONAL HERMITIAN MATRICES)----------------------------
C
C     SUBROUTINE USPEC(N,MATNO)
      SUBROUTINE TSPEC(N,MATNO)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  D(100), DAR(100),DAI(100), PI, EIGVAL(100)
      DOUBLE PRECISION SPACE
      COMPLEX*16   DA(100),DB(100)
      REAL EXPLAN(20)
C-----------------------------------------------------------------------
C    DIMENSION ARRAYS NEEDED TO DEFINE MATRIX.  THEN
C    PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE MATRIX-VECTOR
C    MULTIPLY SUBROUTINE CMATV.
C
C    DIAGONAL ENTRY = D, ABOVE DIAGONAL ENTRY = DA, BELOW DIAGONAL = DB.
C
      READ(8,10) EXPLAN
   10 FORMAT(20A4)
      READ(8,*) NOLD,MATOLD
C
      WRITE(6,20) N,MATOLD
   20 FORMAT(I10,2I6,I8,' = N,MATOLD'/)
C
```

```
C     TEST OF PARAMETER CORRECTNESS                                     HHL09100
      ITEMP = (NOLD-N)**2 + (MATNO-MATOLD)**2                           HHL09110
C                                                                       HHL09120
      IF(ITEMP.EQ.0) GO TO 40                                           HHL09130
C                                                                       HHL09140
      WRITE(6,30)                                                       HHL09150
   30 FORMAT(' PROGRAM TERMINATES BECAUSE EITHER ORDERS OF OR LABELS FORHHL09160
     1 MATRIX DISAGREE')                                                HHL09170
      GO TO 250                                                         HHL09180
C                                                                       HHL09190
   40 CONTINUE                                                          HHL09200
C                                                                       HHL09210
C     IF ITOEP = 1 THEN MATRIX IS TOEPLITZ AND WE PRINT OUT TRUE        HHL09220
C     EIGENVALUES                                                       HHL09230
      READ(8,10) EXPLAN                                                 HHL09240
      READ(8,*) ITOEP                                                   HHL09250
      READ(8,10) EXPLAN                                                 HHL09260
C                                                                       HHL09270
      IF(ITOEP.EQ.1) WRITE(6,50)                                        HHL09280
   50 FORMAT(/' TEST MATRIX IS HERMITIAN TOEPLITZ'/)                    HHL09290
      IF(ITOEP.NE.1) GO TO 110                                          HHL09300
C                                                                       HHL09310
      READ(8,*) DAR(1),DAI(1),D(1)                                      HHL09320
      DA(1) = DCMPLX(DAR(1),DAI(1))                                     HHL09330
      DB(1) = DCONJG(DA(1))                                             HHL09340
      DO 60 J=2,N                                                       HHL09350
      D(J) = D(1)                                                       HHL09360
      DA(J) = DA(1)                                                     HHL09370
   60 DB(J) = DB(1)                                                     HHL09380
      WRITE(6,70) DB(1),D(1),DA(1)                                      HHL09390
      WRITE(9,70) DB(1),D(1),DA(1)                                      HHL09400
   70 FORMAT(' HERMITIAN TOEPLITZ MATRIX IS USED.'/' BELOW DIAGONAL ENTRHHL09410
     1Y = ',2E12.3/' DIAGONAL ENTRY = ',E12.3/' ABOVE DIAGONAL ENTRY =' HHL09420
     1,2E12.3)                                                          HHL09430
C                                                                       HHL09440
C     COMPUTE THE TRUE EIGENVALUES.  FORMULA IS CORRECT ONLY FOR THOSE  HHL09450
C     MATRICES WHOSE DIAGONAL = 2., ABOVE DIAGONAL = A, BELOW DIAGONAL  HHL09460
C     = A-CONJUGATE, AND A HAS NORM 1.                                  HHL09470
C                                                                       HHL09480
      PI = DACOS(-1.D0)                                                 HHL09490
      DO 80 J=1,N                                                       HHL09500
   80 EIGVAL(J) = 2.D0 * (1.D0 -DCOS(PI*DFLOAT(J)/DFLOAT(N+1)))         HHL09510
      WRITE(9,90) N                                                     HHL09520
   90 FORMAT(I6, ' = ORDER OF MATRIX'/' TRUE EIGENVALUES ARE'/)         HHL09530
      WRITE(9,100) (J, EIGVAL(J), J=1,N)                                HHL09540
  100 FORMAT(I5,4X,E25.16,6X,I5,4X,E25.16)                              HHL09550
      GO TO 240                                                         HHL09560
C                                                                       HHL09570
C     NONTOEPLITZ HERMITIAN.  DIAGONAL ENTRIES ARE EQUALLY-SPACED.      HHL09580
C     ABOVE DIAGONAL ENTRIES ARE GENERATED BY GENERATING EQUALLY-SPACED HHL09590
C     REAL PARTS, AND EQUALLY-SPACED IMAGINARY PARTS.  THE BELOW        HHL09600
C     DIAGONAL ENTRIES ARE THEN OBTAINED BY TAKING THE COMPLEX CONJUGATEHHL09610
C     OF THE ABOVE DIAGONAL ENTRIES                                     HHL09620
C                                                                       HHL09630
  110 READ(8,*) D(1), SPACE                                             HHL09640
```

```
      WRITE(6,120) D(1),SPACE
  120 FORMAT(' 1ST DIAGONAL ENTRY =',E20.12,' SPACING =',E20.12)
      DO 130 J=2,N
  130 D(J) = D(J-1) + SPACE
      WRITE(6,140) (D(J), J=1,3)
  140 FORMAT(' 1ST THREE DIAGONAL ENTRIES ='/(2E20.12))
      READ(8,10) EXPLAN
      READ(8,*) DAR(1), SPACE
      WRITE(6,150) DAR(1),SPACE
  150 FORMAT(' REAL PART OF 1ST ABOVE DIAGONAL ENTRY =',E20.12,/
     1' SPACING = ',E20.12)
      DO 160 J=2,N
  160 DAR(J) = DAR(J-1) + SPACE
      WRITE(6,170) (DAR(J), J=1,3)
  170 FORMAT(' REAL PARTS OF 1ST THREE ABOVE DIAGONAL ENTRIES ='/
     1(2E20.12))
      READ(8,10) EXPLAN
      READ(8,*) DAI(1), SPACE
      WRITE(6,180) DAI(1),SPACE
  180 FORMAT(' IMAGINARY PART OF 1ST ABOVE =',E20.12,/' SPACING =',
     1 E20.12)
      DO 190 J=2,N
  190 DAI(J) = DAI(J-1) + SPACE
      WRITE(6,200) (DAI(J), J = 1,3)
  200 FORMAT(' IMAGINARY PARTS OF 1ST THREE ABOVE DIAGONAL ENTRIES ='/
     1 (2E20.12))
      DO 210 J=1,N
      DA(J) = DCMPLX(DAR(J),DAI(J))
  210 DB(J) = DCONJG(DA(J))
C
      WRITE(9,220) (D(J), J=1,N)
  220 FORMAT(' DIAGONAL ENTRIES ='/(4E20.12))
      WRITE(9,230) (DA(J), J=1,N)
  230 FORMAT(' ABOVE DIAGONAL ENTRIES'/(4E20.12))
C
C     PASS STORAGE LOCATIONS OF ARRAYS THAT DEFINE THE MATRIX TO
C     THE MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV
C
  240 CONTINUE
C
C-----------------------------------------------------------------------
      CALL TMATVE(DA,DB,D,N)
C-----------------------------------------------------------------------
C
      RETURN
  250 STOP
C
C-----END OF USPEC------------------------------------------------------
      END
C
C-----START OF MATRIX-VECTOR MULTIPLY (HERMITIAN TRIDIAGONAL)-----------
C
C     SUBROUTINE CMATV(W,U,SUM)
      SUBROUTINE TMATV(W,U,SUM)
C
```

```
C-----------------------------------------------------------------------
      COMPLEX*16   U(1),W(*),DA(1),DB(1)
      DOUBLE PRECISION  D(1),SUM
C-----------------------------------------------------------------------
C    HERMITIAN MATRIX-VECTOR MULTIPLY FOR LANCZS  U = A*W - SUM*U
C    MATRIX IS TRIDIAGONAL HERMITIAN TOEPLITZ
C-----------------------------------------------------------------------
C
C     COMPUTE A*W - SUM*U
C
      GO TO 3
C-----------------------------------------------------------------------
C    STORAGE LOCATIONS ARE PASSED TO CMATV FROM USPEC
      ENTRY TMATVE(DA,DB,D,N)
      GO TO 4
C-----------------------------------------------------------------------
    3 CONTINUE
C
      U(1) = D(1)*W(1) + DA(1)*W(2) - SUM*U(1)
      N1 = N-1
      DO 10 I = 2,N1
   10 U(I) = DB(I-1)*W(I-1)+D(I)*W(I) + DA(I)*W(I+1) -SUM*U(I)
      U(N) = DB(N-1)*W(N-1) + D(N)*W(N) - SUM*U(N)
C
    4 RETURN
C
C-----END OF CMATV------------------------------------------------------
      END
C-----DUMMY USPEC DOES NOTHING------------------------------------------
C
      SUBROUTINE USPEC(N,MATNO)
C     SUBROUTINE CUSPEC(N,MATNO)
C
C-----------------------------------------------------------------------
      RETURN
      END
```

3.5 HLEVAL: HLEVEC: File Definitions, Sample Input Files

Below is a listing of the input/output files definitions which are accessed by the Hermitian Lanczos eigenvalue program, HLEVAL. Included also is a sample of the input file which HLEVAL requires on file 5. The parameters are supplied in free format. HLEVAL computes eigenvalues of Hermitian matrices A on user-specified intervals which must be supplied in ascending order. File 8 is assumed to contain the data which defines the Hermitian nxn matrix A.

```
Sample Specifications of the input/output files for HLEVAL
----------------------------------------------------------------
 HLEVAL EXEC HERMITIAN EIGENVALUE CALCULATION
FI 06 TERM
FILEDEF  1 DISK &1        NHISTORY  A (RECFM F LRECL 80 BLOCK 80
FILEDEF  2 DISK &1        HISTORY   A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1        GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1        ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK HLEVAL    INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1        INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1        DISTINCT  A (RECFM F LRECL 80 BLOCK 80
LOAD   HLEVAL   LESUB   HLEMULT
----------------------------------------------------------------
```

```
Sample Input File for HLEVAL
-----------------------------------------------------------------
 HLEVAL INPUT EIGENVALUE COMPUTATION, NO REORTHOGONALIZATION
 HERMITIAN TEST MATRIX
LINE 1      N    KMAX    NMEVS     MATNO
         528    1600        3    721830
LINE 2   SVSEED    RHSEED      MXINIT    MXSTUR
       49302312   5731029           5    100000
LINE 3   ISTART     ISTOP
              0         1
LINE 4     IHIS     IDIST  IWRITE
              1         0       1
LINE 5   RELTOL (RELATIVE TOLERANCE IN 'COMBINING' GOODEV)
    .0000000001
LINE 6   MB(1)   MB(2)   MB(3)   MB(4)    (ORDERS OF T(1,MEV) )
           528    1056    1584
LINE 7    NINT     (NUMBER OF SUB-INTERVALS FOR BISEC)
             1
LINE 8    LB(1)    LB(2)   LB(3)   LB(4)  (INTERVAL LOWER BOUNDS)
           1.0
LINE 9    UB(1)    UB(2)   UB(3)   UB(4)  (INTERVAL UPPER BOUNDS)
           2.0
-----------------------------------------------------------------
```

Below is a listing of the input/output files definitions which are accessed by the Hermitian Lanczos eigenvector program, HLEVEC. Included also is a sample of the input file which HLEVEC requires on file 5. The parameters are supplied in free format. HLEVEC computes eigenvectors for each of a user-specified subset of the eigenvalues computed by the companion code HLEVEC. Eigenvector approximations will be computed only for eigenvalue approximations which have converged.

```
Sample Specifications of the Input/Output Files for HLEVEC
-------------------------------------------------------------------
 HLEVEC EXEC TO RUN LANCZOS EIGENVECTOR PROGRAM, HERMITIAN MATRICES
FI 06 TERM
FILEDEF  2 DISK &1       HISTORY   A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1       GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1       ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK HLEVEC   INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1       INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  9 DISK &1       ERREST    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 10 DISK &1       BOUNDS    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1       TEIGVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 12 DISK &1       RITZVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 13 DISK &1       PAIGE     A (RECFM F LRECL 80 BLOCK 80
LOAD  HLEVEC  LESUB  HLEMULT
-------------------------------------------------------------------
```

```
Sample Input File for HLEVEC
---------------------------------------------------------------------
 HLEVEC EIGENVECTORS OF HERMITIAN MATRIX, NO REORTHOGONALIZATION
LINE 1  MDIMTV    MDIMRV  MBETA(MAX.DIMENSIONS, TVEC, RITVEC AND BETA
         10000     10000   2000
LINE 2       RELTOL
        .0000000001
LINE 3  MBOUND   NTVCON SVTVEC IREAD (FLAGS
             0        1      0     1
LINE 4  TVSTOP   LVCONT ERCONT  IWRITE (FLAGS
             0        1      1       1
LINE 5    RHSEED  (RANDOM GENERATOR SEED FOR STARTING VECTOR IN INVERM
        45329517
LINE 6  MATNO      N
          100    100
----------------------------------------------------------------------
```

Chapter 4

Factored Inverses of Real Symmetric Matrices

4.1 Introduction

The FORTRAN codes in this chapter address the question of computing distinct eigenvalues and corresponding eigenvectors of a real symmetric matrix by applying a single-vector Lanczos procedure to the inverse of an associated matrix $B \equiv PCP^T$ where $C = (SCALE) * A + (SHIFT) * I$. The scalars $SCALE$ and $SHIFT$ are specified by the user, selected in such a way that the resulting matrix C (or B) has a reasonable numerical condition. The permutation matrix P is chosen so that for a sparse matrix A, the resulting factorization of B is also sparse.

For a given real symmetric matrix A, these codes compute real scalars λ and corresponding real-valued vectors $x \neq 0$ such that

$$B^{-1}x = \lambda x, \tag{4.1.1}$$

where B is as defined above. Note that the eigenvectors of B^{-1} are simple permutations of the eigenvectors of A. The eigenvalues of A are obtained from those of B by a simple scalar modification, which is incorporated in the codes. These codes do not require the matrix A. The Lanczos computations use only the user-supplied factorization of the associated matrix B, the scalars $SCALE$ and $SHIFT$, and the permutation P (if any).

Real symmetric matrices and factorizations of such matrices are discussed in Stewart [24]. See also Bunch and Kaufman [2] and George and Liu [10]. Chapter 2, Section 2.1, contains a brief summary of the properties of real symmetric matrices which we use in these codes.

Given a real symmetric matrix A, the user may decide to use the codes in this chapter rather than those in Chapter 2 if the eigenvalues to be computed are 'small' with 'small' gaps between them and the required factorization can be obtained with a reasonable amount of computation and storage. The user should note however that this type of transformation of the given matrix may not yield an eigenvalue distribution which is better for these Lanczos codes. Such a transformation will accelerate the Lanczos computations only if the desired eigenvalues either become larger in size relative to the other eigenvalues and/or the gaps between the desired eigenvalues become larger relative to the gaps between the other eigenvalues. This type of transformation can be very effective in compressing the big end of the spectrum of a given matrix and enhancing the small end of the spectrum. The Lanczos procedure, however, does not require large gaps between the desired eigenvalues, all it really requires is a reasonable overall gap ratio. That is, the ratio of the largest gap between two neighboring eigenvalues to the smallest such gap must be a reasonable size.

The single-vector Lanczos codes in this chapter can be used to compute either a very few or very many of the distinct eigenvalues of the given real symmetric matrix. The documentation for these codes is contained in Chapter 2, Section 2.2. As in the direct real symmetric case (Chapter 2, Section 2.1), the A-multiplicity of a given computed eigenvalue can be obtained only with additional computation, and the modifications required to do this additional computation are not included in these versions of the codes. This implementation uses the basic Lanczos recursion contained in Eqns (1.2.1) and (1.2.2) to generate a family of real symmetric tridiagonal matrices (T-matrices) for the matrix B^{-1} , whose sizes are specified by the user. Specifically, for $i = 1, 2, ..., m$ and a randomly-generated starting vector v_1 with $\|v_1\| = 1$, generate Lanczos vectors v_i using the following recursion and Eqn(1.2.2) applied to the matrix B^{-1} .

$$\beta_{i+1} v_{i+1} = B^{-1} v_i - \alpha_i v_i - \beta_i v_{i-1}. \tag{4.1.2}$$

B is the matrix defined above in terms of the scalars $SCALE$ and $SHIFT$, and the permutation P, and each $B^{-1}v_i$ is evaluated by solving the system of equations $Bz = v_i$.

LIVAL, the main program for the factored inverse computations, calls the subroutine BISEC to compute eigenvalues of the specified Lanczos tridiagonal matrices on the user-specified intervals. BISEC simultaneously computes these T-eigenvalues with their T-multiplicities and sorts the computed T-eigenvalues into two classes, the 'good' T-eigenvalues and the 'spurious' T-eigenvalues. The 'good' T-eigenvalues are accepted as approximations to eigenvalues of the B^{-1} matrix associated with the user-specified matrix A, scalars $SCALE$ and $SHIFT$, and the permutation matrix P (if any). The accuracy of these 'good' T-eigenvalues as eigenvalues of B^{-1} is then estimated using error estimates computed by subroutine INVERR. Error estimates are computed only for isolated 'good' T-eigenvalues. All other 'good' T-eigenvalues are assumed to have converged.

Convergence is then checked. If convergence has not yet occurred and a larger T-matrix has been specified by the user, the program will continue on to the larger T-matrix, repeating the above procedure on this larger matrix. After each T-matrix eigenvalue computation, the corresponding approximations to the eigenvalues of the user-specified matrix A are computed and included in the output.

Once the eigenvalues of B^{-1} have been computed accurately enough, the user can select a subset of the 'converged' eigenvalues for which eigenvectors are to be computed. The main program LIVEC, for computing eigenvectors of the inverse of a real symmetric matrix, given a factorization, is used to compute the desired eigenvectors. If the matrix B is a permutation of the matrix C, then LIVEC unwinds the permutation to obtain the corresponding eigenvectors of the user-supplied A-matrix.

All of the computations are done in double precision real arithmetic. Once the Lanczos T-matrices have been computed, the remaining computations use the same subroutines that are used in the real symmetric case discussed in Chapter 2. In addition to the programs and subroutines provided here, the user must supply a subroutine USPEC which defines and initializes the factorization of the scaled, shifted, and permuted version B of the original matrix A, and a subroutine BSOLV which computes matrix-vector multiplies $B^{-1}x$ for any given vector x. These subroutines must be constructed in such a way as to take advantage of the sparsity (and/or structure) of the user-supplied A-matrix and such that these computations are done accurately.

The sample subroutines USPEC and BSOLV provided assume that the associated matrix B is positive definite and that its Cholesky factorization

$$B = LL^T, \tag{4.1.3}$$

where L is a lower triangular matrix, is used to compute $B^{-1}y$, for any given y. Thus, the sample USPEC subroutine provided for this chapter defines and initializes arrays which define the Cholesky factor L of the associated matrix B. The sample BSOLV subroutine provided computes the required matrix-vector multiplies $u = B^{-1}y$ by solving sequentially the two equations $Lz = y$ and $L^T u = z$. These two equations are very easy to solve since L is a triangular matrix. The main portions of these Lanczos codes do not however require that the B-matrix be positive definite, only that a factorization be available. Therefore, the user could replace the sample USPEC and BSOLV subroutines by subroutines which use a more general factorization of B, for example $B = LDL^T$, where D is a diagonal matrix. All that is necessary is that the BSOLV subroutine provide the matrix-vector products $B^{-1}x$, rapidly and accurately. The information supplied to the Lanczos procedures about the matrix being processed must be consistent.

Several optional preprocessing programs are provided, PERMUT, LORDER, LFACT, and LTEST. PERMUT calls the SPARSPAK Library [9] to attempt to identify a reordering or permutation P of the given matrix A for which sparseness will be preserved under factorization of the permuted matrix. LORDER takes a given matrix C and permutation P and computes the sparse matrix format for the permuted matrix, $B \equiv PCP^T$. LFACT computes the Cholesky factors of a given positive definite matrix. LTEST performs a very crude check on the numerical condition of the matrix supplied to it, by solving a system of equations with and without iterative refinement LINPACK [7].

4.2 LIVAL: Main Program, Eigenvalue Computations

```
C-----LIVAL---(EIGENVALUES OF INVERSES OF REAL SYMMETRIC MATRICES)------LIV00010
C  Authors:  Jane Cullum and Ralph A. Willoughby (deceased)             LIV00020
C            Los Alamos National Laboratory                             LIV00030
C            Los Alamos, New Mexico 87544                               LIV00040
C                                                                       LIV00050
C            E-mail:  cullumj@lanl.gov                                  LIV00060
C                                                                       LIV00070
C  These codes are copyrighted by the authors.  These codes             LIV00080
C  and modifications of them or portions of them are NOT to be          LIV00090
C  incorporated into any commercial codes or used for any other         LIV00100
C  commercial purposes such as consulting for other companies,          LIV00110
C  without legal agreements with the authors of these Codes.            LIV00120
C  If these Codes or portions of them are used in other scientific or   LIV00130
C  engineering research works the names of the authors of these codes   LIV00140
C  and appropriate references to their written work are to be           LIV00150
C  incorporated in the derivative works.                                LIV00160
C                                                                       LIV00170
C  This header is not to be removed from these codes.                   LIV00180
C                                                                       LIV00190
C         REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4            LIV00191
C         Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsLIV00192
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   LIV00193
C         Applied Mathematics, 2002. SIAM Publications,                 LIV00194
C         Philadelphia, PA. USA                                         LIV00195
C                                                                       LIV00200
C      CONTAINS MAIN PROGRAM FOR COMPUTING DISTINCT EIGENVALUES OF      LIV00210
C      INVERSES OF REAL SYMMETRIC MATRICES USING REORDERING             LIV00220
C      AND SPARSE FACTORIZATION.  THE LANCZOS RECURSION IS APPLIED      LIV00230
C      TO A SCALED, SHIFTED, AND REORDERED VERSION B OF THE             LIV00240
C      ORIGINAL A-MATRIX.  THE PROCEDURE USES LANCZOS                   LIV00250
C      TRIDIAGONALIZATION WITHOUT REORTHOGONALIZATION                   LIV00260
C                                                                       LIV00270
C      PFORT VERIFIER IDENITIFIED THE FOLLOWING NONPORTABLE             LIV00280
C      CONSTRUCTIONS                                                    LIV00290
C                                                                       LIV00300
C      1.  DATA/MACHEP/ STATEMENT                                       LIV00310
C      2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                       LIV00320
C      3.  FORMAT(20A4) USED WITH EXPLANATORY HEADER EXPLAN.            LIV00330
C      4.  HEXADECIMAL FORMAT (4Z20) USED IN ALPHA/BETA FILES 1 AND 2.  LIV00340
C                                                                       LIV00350
C-----------------------------------------------------------------------LIV00360
C                                                                       LIV00370
      DOUBLE PRECISION  ALPHA(3000),BETA(3001)                          LIV00380
      DOUBLE PRECISION  V1(3001),V2(3000),VS(3000)                      LIV00390
      DOUBLE PRECISION  LB(20),UB(20)                                   LIV00400
      DOUBLE PRECISION  BTOL,GAPTOL,TTOL,MACHEP,EPSM,SHIFT,SHIFTO,RELTOLLIV00410
      DOUBLE PRECISION  SCALE1,SCALE2,SCALE3,SCALE4,BISTOL,CONTOL,MULTOLLIV00420
      DOUBLE PRECISION  ONE,ZERO,TEMP,TKMAX,BETAM,BKMIN,T0,T1,S0        LIV00430
      REAL  G(3000),GG(3000),EXPLAN(20)                                 LIV00440
      INTEGER  MP(3000),NMEV(20)                                        LIV00450
      INTEGER  SVSEED,RHSEED,SVSOLD                                     LIV00460
```

```
      INTEGER IABS
      REAL  ABS
      DOUBLE PRECISION  DABS, DSQRT, DFLOAT
      EXTERNAL BSOLV
C
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.0D0*MACHEP
C-----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     1.  ALPHA: >= KMAX,   BETA: >= (KMAX+1) WHERE KMAX MAY
C         IS THE LARGEST SIZE T-MATRIX TO BE CONSIDERED.
C     2.  V1:  >= MAX(N,KMAX+1)
C     3.  V2,VS:   >= MAX(N,KMAX)
C     4.  GG:  >=  KMAX
C     5.  G:  >= MAX(N,2*KMAX)
C     6.  MP:  >= KMAX
C     7.  LB,UB:  >= NUMBER OF SUBINTERVALS SUPPLIED TO BISEC.
C     8.  NMEV:  >= NUMBER OF T-MATRICES ALLOWED.
C     9.  EXPLAN:  DIMENSION IS 20.
C
C
C     IMPORTANT TOLERANCES OR SCALES THAT ARE USED REPEATEDLY
C     THROUGHOUT THE PROGRAM ARE THE FOLLOWING:
C     SCALED MACHINE EPSILON:  TTOL = TKMAX*EPSM WHERE
C     EPSM = 2*MACHINE EPSILON AND
C     TKMAX = MAX(|ALPHA(J)|,BETA(J), J = 1,MEV)
C     BISEC CONVERGENCE TOLERANCE:  BISTOL  = DSQRT(1000+MEV)*TTOL
C     BISEC MULTIPLICITY TOLERANCE:  MULTOL = (1000+MEV)*TTOL
C     LANCZOS CONVERGENCE TOLERANCE: CONTOL = BETA(MEV+1)*1.D-10
C
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LANCZOS PROCEDURE FOR FACTORED INVERSES OF REAL SYMMETRI
     1C MATRICES')
C
C     SET PROGRAM PARAMETERS
C     SCALEK ARE USED IN TOLERANCES NEEDED IN SUBROUTINES LUMP,
C     ISOEV AND PRTEST.  USER MUST NOT MODIFY THESE SCALES.
      SCALE1 = 5.0D2
      SCALE2 = 5.0D0
      SCALE3 = 5.0D0
      SCALE4 = 1.0D4
      ONE  = 1.0D0
      ZERO = 0.0D0
C     BTOL = 1.0D-8
      BTOL = EPSM
      GAPTOL = 1.0D-8
      ICONV = 0
      MOLD = 0
      MOLD1 = 1
      ICT = 0
      MMB = 0
```

```
      IPROJ = 0
C-----------------------------------------------------------------------
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 5 (FREE FORMAT)
C
C     READ USER-PROVIDED HEADER FOR RUN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C     READ ORDER OF MATRICES (N) , MAXIMUM ORDER OF T-MATRIX (KMAX),
C     NUMBER OF T-MATRICES ALLOWED (NMEVS), AND MATRIX IDENTIFICATION
C     NUMBERS (MATNO), SHIFT APPLIED TO MATRIX (SHIFT) AND
C     SCALE (S0).
      READ(5,20) EXPLAN
      READ(5,*) N,KMAX,NMEVS,MATNO,S0,SHIFT
C
C     READ SEEDS FOR LANCZS AND INVERR SUBROUTINES (SVSEED AND RHSEED)
C     READ MAXIMUM NUMBER OF ITERATIONS ALLOWED FOR EACH INVERSE
C     ITERATION (MXINIT) AND MAXIMUM NUMBER OF STURM SEQUENCES
C     ALLOWED (MXSTUR)
      READ(5,20) EXPLAN
      READ(5,*) SVSEED,RHSEED,MXINIT,MXSTUR
C
C     ISTART = (0,1):  ISTART = 0 MEANS ALPHA/BETA FILE IS NOT
C     AVAILABLE.  ISTART = 1 MEANS ALPHA/BETA FILE IS AVAILABLE ON
C     FILE 2.
C     ISTOP = (0,1):  ISTOP = 0 MEANS PROCEDURE GENERATES ALPHA/BETA
C     FILE AND THEN TERMINATES.  ISTOP = 1 MEANS PROCEDURE GENERATES
C     ALPHAS/BETAS IF NEEDED AND THEN COMPUTES EIGENVALUES AND ERROR
C     ESTIMATES AND THEN TERMINATES.
      READ(5,20) EXPLAN
      READ(5,*) ISTART,ISTOP
C
C     IHIS = (0,1):  IHIS = 0 MEANS ALPHA/BETA FILE IS NOT WRITTEN
C     TO FILE 1.  IHIS = 1 MEANS ALPHA/BETA FILE IS WRITTEN TO FILE 1.
C     IDIST = (0,1):  IDIST = 0 MEANS DISTINCT T-EIGENVALUES
C     ARE NOT WRITTEN TO FILE 11.  IDIST = 1 MEANS DISTINCT
C     T-EIGENVALUES ARE WRITTEN TO FILE 11.
C     IWRITE = (0,1):  IWRITE = 0 MEANS NO INTERMEDIATE OUTPUT
C     FROM THE COMPUTATIONS IS WRITTEN TO FILE 6.  IWRITE = 1 MEANS
C     T-EIGENVALUES AND ERROR ESTIMATES ARE WRITTEN TO FILE 6
C     AS THEY ARE COMPUTED.
      READ(5,20) EXPLAN
      READ(5,*) IHIS,IDIST,IWRITE
C
C     READ IN THE RELATIVE TOLERANCE (RELTOL) FOR USE IN THE
C     SPURIOUS, T-MULTIPLICITY, AND PRTESTS.
      READ(5,20) EXPLAN
      READ(5,*) RELTOL
C
C     READ IN THE SIZES OF THE T-MATRICES TO BE CONSIDERED.
      READ(5,20) EXPLAN
      READ(5,*) (NMEV(J), J=1,NMEVS)
```

```
C                                                                       LIV01570
C     READ IN THE NUMBER OF SUBINTERVALS TO BE CONSIDERED.              LIV01580
      READ(5,20) EXPLAN                                                 LIV01590
      READ(5,*) NINT                                                    LIV01600
C                                                                       LIV01610
C     READ IN THE LEFT-END POINTS OF THE SUBINTERVALS TO BE CONSIDERED. LIV01620
C     THESE MUST BE IN ALGEBRAICALLY-INCREASING ORDER                   LIV01630
      READ(5,20) EXPLAN                                                 LIV01640
      READ(5,*) (LB(J), J=1,NINT)                                       LIV01650
C                                                                       LIV01660
C     READ IN THE RIGHT-END POINTS OF THE SUBINTERVALS TO BE CONSIDERED.LIV01670
C     THESE MUST BE IN ALGEBRAICALLY-INCREASING ORDER                   LIV01680
      READ(5,20) EXPLAN                                                 LIV01690
      READ(5,*) (UB(J), J=1,NINT)                                       LIV01700
C                                                                       LIV01710
C-----------------------------------------------------------------------LIV01720
C     INITIALIZE THE ARRAYS FOR THE FACTORIZATION OF THE ASSOCIATED     LIV01730
C     SCALED, SHIFTED AND PERMUTED VERSION OF THE A-MATRIX.             LIV01740
C     THE STORAGE LOCATIONS OF THESE ARRAYS ARE PASSED TO THE BSOLV     LIV01750
C     SUBROUTINE WHICH WILL BE CALLED FROM LANCZS FOR THE T-MATRIX      LIV01760
C     GENERATION.                                                       LIV01770
C                                                                       LIV01780
      CALL USPEC(N,MATNO)                                               LIV01790
C                                                                       LIV01800
C-----------------------------------------------------------------------LIV01810
C                                                                       LIV01820
C     MASKS UNDERFLOW AND OVERFLOW, USER MUST SUPPLY OR COMMENT OUT.    LIV01830
      CALL MASK                                                         LIV01840
C                                                                       LIV01850
C-----------------------------------------------------------------------LIV01860
C                                                                       LIV01870
C     WRITE TO FILE 6, A SUMMARY OF THE PARAMETERS FOR THIS RUN         LIV01880
C                                                                       LIV01890
      WRITE(6,30) MATNO,N,KMAX,SHIFT,S0                                 LIV01900
   30 FORMAT(/3X,'MATRIX ID',4X,'ORDER OF A',4X,'MAX ORDER OF T'//      LIV01910
     1 I12,I14,I18//8X,' SHIFT',8X,'SCALE'/2E15.6//                     LIV01920
     1 ' C = SCALE*A + SHIFT*I '/                                       LIV01930
     1 ' B = P*C*P-TRANSPOSE WHERE P IS A REORDERING OF C'/             LIV01940
     1 ' LANCZOS PROCEDURE USES THE FACTORIZATION OF B'/)               LIV01950
C                                                                       LIV01960
      WRITE(6,40) ISTART,ISTOP                                          LIV01970
   40 FORMAT(/2X,'ISTART',3X,'ISTOP'/2I8/)                              LIV01980
C                                                                       LIV01990
      WRITE(6,50) IHIS,IDIST,IWRITE                                     LIV02000
   50 FORMAT(/4X,'IHIS',3X,'IDIST',2X,'IWRITE'/3I8/)                    LIV02010
C                                                                       LIV02020
      WRITE(6,60) SVSEED,RHSEED                                         LIV02030
   60 FORMAT(/' SEEDS FOR RANDOM NUMBER GENERATOR'//                    LIV02040
     1 4X,'LANCZS SEED',4X,'INVERR SEED'/2I15/)                         LIV02050
C                                                                       LIV02060
      WRITE(6,70) (NMEV(J), J=1,NMEVS)                                  LIV02070
   70 FORMAT(/' SIZES OF T-MATRICES TO BE CONSIDERED'/(6I12))           LIV02080
C                                                                       LIV02090
      WRITE(6,80) RELTOL,GAPTOL,BTOL                                    LIV02100
   80 FORMAT(/' RELATIVE TOLERANCE USED TO COMBINE COMPUTED T-EIGENVALUELIV02110
```

```
     1S'/E15.3/' RELATIVE GAP TOLERANCES USED IN INVERSE ITERATION'/
     1E15.3/' RELATIVE TOLERANCE FOR CHECK ON SIZE OF BETAS'/E15.3/)
C
      WRITE(6,90) (J,LB(J),UB(J), J=1,NINT)
   90 FORMAT(/' BISEC WILL BE USED ON THE FOLLOWING INTERVALS'/
     1 (I6,2E20.6))
C
      IF (ISTART.EQ.0) GO TO 140
C
C     READ IN ALPHA BETA HISTORY
C
      READ(2,100)MOLD,NOLD,SVSOLD,MATOLD,SHIFTO
  100 FORMAT(2I6,I12,I8,E13.4)
C
      IF (KMAX.LT.MOLD) KMAX = MOLD
      KMAX1 = KMAX + 1
C
C     CHECK THAT ORDER N, MATRIX ID MATNO, AND RANDOM SEED SVSEED
C     AGREE WITH THOSE IN THE HISTORY FILE.  IF NOT PROCEDURE STOPS.
C
      ITEMP = (NOLD-N)**2+(MATNO-MATOLD)**2+(SVSEED-SVSOLD)**2
C
      IF (ITEMP.EQ.0.AND.SHIFT.EQ.SHIFTO) GO TO 120
C
      WRITE(6,110)
  110 FORMAT(' PROGRAM TERMINATES'/    ' READ FROM FILE 2 CORRESPONDS TO
     1 DIFFERENT MATRIX THAN MATRIX SPECIFIED'/)
      GO TO 700
C
  120 CONTINUE
      MOLD1 = MOLD+1
C
      READ(2,130)(ALPHA(J), J=1,MOLD)
      READ(2,130)(BETA(J), J=1,MOLD1)
  130 FORMAT(4Z20)
C
      IF (KMAX.EQ.MOLD) GO TO 170
C
      READ(2,130)(V1(J), J=1,N)
      READ(2,130)(V2(J), J=1,N)
C
  140 CONTINUE
      IIX = SVSEED
C
      WRITE(6,150)
  150 FORMAT(' ENTERING LANCZS'/)
C
C-----------------------------------------------------------------------
C
      CALL LANCZS(BSOLV,ALPHA,BETA,V1,V2,VS,G,KMAX,MOLD1,N,IIX)
C
C-----------------------------------------------------------------------
C
C     ALPHA BETA WRITE
      KMAX1 = KMAX + 1
```

```
C
      IF(IHIS.EQ.0.AND.ISTOP.GT.0) GO TO 170
C
      WRITE(1,160) KMAX,N,SVSEED,MATNO,SHIFT
  160 FORMAT(2I6,I12,I8,E13.4,' = KMAX,N,SVSEED,MATNO,SHIFT')
C
      WRITE(1,130)(ALPHA(I), I=1,KMAX)
      WRITE(1,130)(BETA(I), I=1,KMAX1)
C
      WRITE(1,130)(V1(I), I=1,N)
      WRITE(1,130)(V2(I), I=1,N)
C
      IF (ISTOP.EQ.0) GO TO 600
C
  170 CONTINUE
      KMAX1 = KMAX + 1
      BKMIN = BTOL
C
      WRITE(6,180)
  180 FORMAT(/' T-MATRICES (ALPHA AND BETA) ARE NOW AVAILABLE'/)
C
C-----------------------------------------------------------------------
C     SUBROUTINE TNORM CHECKS MIN(BETA)/(ESTIMATED NORM(A)) > BTOL .
C     IF THIS IS VIOLATED IB IS SET EQUAL TO THE NEGATIVE OF THE INDEX
C     OF THE MINIMAL BETA.  IF(IB < 0) THEN SUBROUTINE TNORM IS
C     CALLED FOR EACH VALUE OF MEV TO DETERMINE WHETHER OR NOT THERE
C     IS A BETA IN THE T-MATRIX SPECIFIED THAT VIOLATES THIS TEST.
C     IF THERE IS SUCH A BETA THE PROGRAM TERMINATES FOR THE USER
C     TO DECIDE WHAT TO DO.  THIS TEST CAN BE OVER-RIDDEN BY
C     SIMPLY MAKING BTOL SMALLER, BUT THEN THERE IS THE POSSIBILITY
C     THAT LOSSES IN THE LOCAL ORTHOGONALITY MAY HURT THE COMPUTATIONS.
C     BTOL = 1.D-8 IS HOWEVER A CONSERVATIVE CHOICE FOR BTOL.
C
C     TNORM ALSO COMPUTES TKMAX = MAX(|ALPHA(K)|,BETA(K), K=1,KMAX).
C     TKMAX IS USED TO SCALE THE TOLERANCES USED IN THE
C     T-MULTIPLICITY AND SPURIOUS TESTS IN BISEC. TKMAX IS ALSO USED IN
C     THE PROJECTION TEST FOR HIDDEN EIGENVALUES THAT HAD 'TOO SMALL'
C     A PROJECTION ON THE STARTING VECTOR.
C
      CALL TNORM(ALPHA,BETA,BKMIN,TKMAX,KMAX,IB)
C
C-----------------------------------------------------------------------
      TTOL = EPSM*TKMAX
C
C     LOOP ON THE SIZE OF THE T-MATRIX
  190 CONTINUE
      MMB = MMB + 1
      MEV = NMEV(MMB)
C     IS MEV TOO LARGE ?
      IF(MEV.LE.KMAX) GO TO 210
C
      WRITE(6,200) MMB, MEV, KMAX
  200 FORMAT(/' TERMINATE PRIOR TO CONSIDERING THE',I6,'TH T-MATRIX'/
     1' BECAUSE THE SIZE REQUESTED',I6,' IS GREATER THAN THE MAXIMUM SIZ
     1E ALLOWED',I6/)
```

```
      GO TO 600
C
  210 MP1 = MEV + 1
      BETAM = BETA(MP1)
      WRITE(6,220) MEV,MEV,BETA(MEV),MEV,BETAM
  220 FORMAT(/' AT T-SIZE = ',I6,' BETA(',I4,') = ',E13.4/' BETA(',I4,'+
     11) =',E13.4)
      IF (IB.GE.0) GO TO 230
      T0 = BTOL
C-----------------------------------------------------------------------
C
      CALL TNORM(ALPHA,BETA,T0,T1,MEV,IBMEV)
C
C-----------------------------------------------------------------------
      TEMP = T0/TKMAX
      IBMEV = IABS(IBMEV)
      IF (TEMP.GE.BTOL) GO TO 230
      IBMEV = -IBMEV
      GO TO 660
  230 CONTINUE
      IC = MXSTUR-ICT
C
C-----------------------------------------------------------------------
C     BISEC LOOP. THE SUBROUTINE BISEC INCORPORATES DIRECTLY THE
C     T-MULTIPLICITY AND SPURIOUS TESTS. T-EIGENVALUES WILL BE
C     CALCULATED BY BISEC SEQUENTIALLY ON INTERVALS
C     (LB(J),UB(J)), J = 1,NINT).
C
C     ON RETURN FROM BISEC
C     NDIS = NUMBER OF DISTINCT EIGENVALUES OF T(1,MEV) ON UNION
C            OF THE (LB,UB) INTERVALS
C     VS = DISTINCT T-EIGENVALUES IN ALGEBRAICALLY INCREASING ORDER
C     MP = T-MULTIPLICITIES OF THE T-EIGENVALUES IN VS
C     MP(I) = (0,1,MI), MI>1, I=1,NDIS  MEANS:
C        (0)  VS(I) IS SPURIOUS
C        (1)  VS(I) IS T-SIMPLE AND GOOD
C        (MI) VS(I) IS MULTIPLE AND IS THEREFORE NOT ONLY GOOD BUT
C             ALSO A CONVERGED GOOD T-EIGENVALUE.
C
      CALL BISEC(ALPHA,BETA,V1,V2,VS,LB,UB,EPSM,TTOL,MP,NINT,
     1 MEV,NDIS,IC,IWRITE)
C
C-----------------------------------------------------------------------
      IF (NDIS.EQ.0) GO TO 680
C
C     COMPUTE THE TOTAL NUMBER OF STURM SEQUENCES USED TO DATE
C     COMPUTE THE BISEC CONVERGENCE AND T-MULTIPLICITY TOLERANCES USED.
C     COMPUTE THE CONVERGENCE TOLERANCE FOR EIGENVALUES OF A.
      ICT = ICT + IC
      TEMP = DFLOAT(MEV+1000)
      MULTOL = TEMP*TTOL
      TEMP = DSQRT(TEMP)
      BISTOL = TTOL*TEMP
      CONTOL = BETAM*1.D-10
C
```

```
C-----------------------------------------------------------------------LIV03770
C    SUBROUTINE LUMP 'COMBINES' T-EIGENVALUES THAT ARE 'TOO CLOSE'.     LIV03780
C    NOTE HOWEVER THAT CLOSE SPURIOUS T-EIGENVALUES ARE NOT AVERAGED    LIV03790
C    WITH GOOD ONES. HOWEVER, THEY MAY BE USED TO INCREASE THE          LIV03800
C    T-MULTIPLICITY OF A GOOD T-EIGENVALUE.                             LIV03810
C                                                                       LIV03820
      LOOP = NDIS                                                       LIV03830
      CALL LUMP(VS,RELTOL,MULTOL,SCALE2,MP,LOOP)                        LIV03840
C                                                                       LIV03850
C-----------------------------------------------------------------------LIV03860
      IF(NDIS.EQ.LOOP) GO TO 250                                        LIV03870
C                                                                       LIV03880
      WRITE(6,240) NDIS, MEV, LOOP                                      LIV03890
  240 FORMAT(/I6,' DISTINCT T-EIGENVALUES WERE COMPUTED IN BISEC AT MEV LIV03900
     1',I6/ 2X,' LUMP SUBROUTINE REDUCES NUMBER OF DISTINCT T-EIGENVALUELIV03910
     1S TO',I6)                                                         LIV03920
C                                                                       LIV03930
  250 CONTINUE                                                          LIV03940
      NDIS = LOOP                                                       LIV03950
      BETA(MP1) = BETAM                                                 LIV03960
C-----------------------------------------------------------------------LIV03970
C     THE SUBROUTINE ISOEV LABELS THOSE SIMPLE EIGENVALUES OF T(1,MEV)  LIV03980
C     WITH VERY SMALL GAPS BETWEEN NEIGHBORING EIGENVALUES OF T(1,MEV)  LIV03990
C     TO AVOID COMPUTING ERROR ESTIMATES FOR ANY SIMPLE GOOD            LIV04000
C     T-EIGENVALUE THAT IS TOO CLOSE TO A SPURIOUS EIGENVALUE.          LIV04010
C     ON RETURN FROM ISOEV, G CONTAINS CODED MINIMAL GAPS               LIV04020
C     BETWEEN THE DISTINCT EIGENVALUES OF T(1,MEV). (G IS REAL).        LIV04030
C     G(I) < 0 MEANS MINGAP IS DUE TO LEFT GAP G(I) > 0 MEANS DUE TO    LIV04040
C     RIGHT GAP. MP(I) = -1 MEANS THAT THE GOOD T-EIGENVALUE IS SIMPLE  LIV04050
C     AND HAS A VERY SMALL MINGAP IN T(1,MEV) DUE TO A SPURIOUS         LIV04060
C     T-EIGENVALUE.  NG = NUMBER OF GOOD T-EIGENVALUES.                 LIV04070
C     NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES.                     LIV04080
C                                                                       LIV04090
      CALL ISOEV(VS,GAPTOL,MULTOL,SCALE1,G,MP,NDIS,NG,NISO)             LIV04100
C                                                                       LIV04110
C-----------------------------------------------------------------------LIV04120
C                                                                       LIV04130
      WRITE(6,260)NG,NISO,NDIS                                          LIV04140
  260 FORMAT(/I6,' GOOD T-EIGENVALUES HAVE BEEN COMPUTED'/              LIV04150
     1 I6,' OF THESE ARE T-ISOLATED'/                                   LIV04160
     2 I6,' = NUMBER OF DISTINCT T-EIGENVALUES COMPUTED'/)              LIV04170
C                                                                       LIV04180
C     DO WE WRITE DISTINCT EIGENVALUES OF T-MATRIX TO FILE 11?          LIV04190
      IF (IDIST.EQ.0) GO TO 310                                         LIV04200
C                                                                       LIV04210
      WRITE(11,270) NDIS,NISO,MEV,N,SVSEED,MATNO                        LIV04220
  270 FORMAT(/4I6,I12,I8,' = NDIS,NISO,MEV,N,SVSEED,MATNO'/)            LIV04230
C                                                                       LIV04240
      WRITE(11,280)                                                     LIV04250
  280 FORMAT(/1X,'MP',21X,'EVBI',5X,'TMINGAP',1X,'MP',21X,'EVBI',5X,    LIV04260
     1'TMINGAP'/)                                                       LIV04270
C                                                                       LIV04280
      WRITE(11,290) (MP(I),VS(I),G(I), I=1,NDIS)                        LIV04290
  290 FORMAT(2(I3,E25.16,E12.3))                                        LIV04300
C                                                                       LIV04310
```

```
      WRITE(11,300) NDIS, (MP(I), I=1,NDIS)                             LIV04320
  300 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/(20I4))LIV04330
C                                                                       LIV04340
  310 CONTINUE                                                          LIV04350
      IF (NISO.NE.0) GO TO 340                                          LIV04360
C                                                                       LIV04370
      WRITE(4,320) MEV                                                  LIV04380
  320 FORMAT(/' AT MEV = ',I6,' THERE ARE NO ISOLATED T-EIGENVALUES'/   LIV04390
     1' SO NO ERROR ESTIMATES WERE COMPUTED/')                          LIV04400
C                                                                       LIV04410
      WRITE(6,330)                                                      LIV04420
  330 FORMAT(/' ALL COMPUTED GOOD T-EIGENVALUES ARE MULTIPLE'/          LIV04430
     1 ' THEREFORE ALL SUCH EIGENVALUES ARE ASSUMED TO HAVE CONVERGED') LIV04440
C                                                                       LIV04450
      ICONV = 1                                                         LIV04460
      GO TO 380                                                         LIV04470
  340 CONTINUE                                                          LIV04480
C-----------------------------------------------------------------------LIV04490
C     SUBROUTINE INVERR COMPUTES ERROR ESTIMATES FOR ISOLATED GOOD      LIV04500
C     T-EIGENVALUES USING INVERSE ITERATION ON T(1,MEV). ON RETURN      LIV04510
C     G(J) = MINIMUM GAP IN T(1,MEV) FOR EACH VS(J), J=1,NDIS           LIV04520
C     G(MEV+I) = BETAM*|U(MEV)| = ERROR ESTIMATE FOR ISOLATED GOOD      LIV04530
C              T-EIGENVALUES, WHERE I = 1, NISO AND  BETAM = BETA(MEV+1)LIV04540
C              U(MEV) IS MEVTH COMPONENT OF THE UNIT EIGENVECTOR OF T   LIV04550
C              CORRESPONDING TO THE ITH ISOLATED GOOD T-EIGENVALUE.     LIV04560
C     A NEGATIVE ERROR ESTIMATE MEANS THAT FOR THAT PARTICULAR          LIV04570
C     EIGENVALUE THE INVERSE ITERATION DID NOT CONVERGE IN <= MXINIT    LIV04580
C     STEPS AND THAT THE CORRESPONDING ERROR ESTIMATE IS QUESTIONABLE.  LIV04590
C                                                                       LIV04600
C     V2 CONTAINS THE ISOLATED GOOD T-EIGENVALUES                       LIV04610
C     V1 CONTAINS THE MINGAPS TO THE NEAREST DISTINCT  EIGENVALUE       LIV04620
C        OF T(1,MEV) FOR EACH ISOLATED GOOD T-EIGENVALUE IN V2.         LIV04630
C     VS CONTAINS THE NDIS DISTINCT EIGENVALUES OF T(1,MEV)             LIV04640
C     MP CONTAINS THE CORRESPONDING CODED T-MULTIPLICITIES              LIV04650
C                                                                       LIV04660
      IT = MXINIT                                                       LIV04670
      CALL INVERR(ALPHA,BETA,V1,V2,VS,EPSM,G,MP,MEV,MMB,NDIS,NISO,N,    LIV04680
     1 RHSEED,IT,IWRITE)                                                LIV04690
C                                                                       LIV04700
C-----------------------------------------------------------------------LIV04710
C     SIMPLE CHECK FOR CONVERGENCE. CHECKS TO SEE IF ALL OF THE         LIV04720
C     LAST COMPONENTS OF EIGENVECTORS ARE L.T. CONTOL.                  LIV04730
C     IF THIS TEST IS SATISFIED, THEN CONVERGENCE FLAG, ICONV IS SET    LIV04740
C     TO 1.  TYPICALLY ERROR ESTIMATES ARE VERY CONSERVATIVE.           LIV04750
C                                                                       LIV04760
      WRITE(6,350) CONTOL                                               LIV04770
  350 FORMAT(/' CONVERGENCE IS TESTED USING THE CONVERGENCE TOLERANCE', LIV04780
     1E13.4/)                                                           LIV04790
C                                                                       LIV04800
      II = MEV +1                                                       LIV04810
      IF = MEV+NISO                                                     LIV04820
      DO 360 I = II,IF                                                  LIV04830
      IF (ABS(G(I)).GT.CONTOL) GO TO 380                                LIV04840
  360 CONTINUE                                                          LIV04850
      ICONV = 1                                                         LIV04860
```

```
      MMB = NMEVS
C
      WRITE(6,370) CONTOL
  370 FORMAT(' ALL COMPUTED ERROR ESTIMATES WERE LESS THAN',E15.4/
     1 ' THEREFORE PROCEDURE TERMINATES'/)
C
  380 CONTINUE
C
      IF (ICONV.EQ.0) GO TO 510
C
C-----------------------------------------------------------------------
C     IF CONVERGENCE IS INDICATED, THAT IS ICONV = 1 ,THEN
C     THE SUBROUTINE PRTEST IS CALLED TO CHECK FOR ANY CONVERGED
C     T-EIGENVALUES THAT HAVE BEEN MISLABELLED AS SPURIOUS BECAUSE
C     THE PROJECTION OF THEIR EIGENVECTOR(S) ON THE STARTING
C     VECTOR WAS(WERE) TOO SMALL.
C     NUMERICAL TESTS INDICATE THAT SUCH EIGENVALUES ARE RARE.
C     IF FOR SOME REASON MANY OF THESE HIDDEN EIGENVALUES APPEAR
C     ON SOME RUN, YOU CAN BE CERTAIN THAT SOMETHING IS FOULED UP.
C
      CALL PRTEST(ALPHA,BETA,VS,TKMAX,EPSM,RELTOL,SCALE3,SCALE4,
     1 MP,NDIS,MEV,IPROJ)
C
C------------------------------------------------------------------------
C
      IF(IPROJ.EQ.0) GO TO 500
C
      IF(IDIST.EQ.1)  WRITE(11,390) IPROJ
  390 FORMAT(' SUBROUTINE PRTEST WANTS TO RELABEL',I6,' SPURIOUS T-EIGEN
     1VALUES'/' WE ACCEPT RELABELLING ONLY IF LAST COMPONENT OF T-EIGENV
     1ECTOR IS L.T. 1.D-10'/)
C
      IIX = RHSEED
C
C------------------------------------------------------------------------
C
      CALL GENRAN(IIX,G,MEV)
C
C------------------------------------------------------------------------
C
      ITEN = -10
      NISOM = NISO + MEV
      IWRITO = IWRITE
      IWRITE = 0
C
      DO 420 J = 1,NDIS
      IF(MP(J).NE.ITEN) GO TO 420
      T0 = VS(J)
C
C------------------------------------------------------------------------
C
      IT = MXINIT
      CALL INVERM(ALPHA,BETA,V1,V2,T0,TEMP,T1,EPSM,G,MEV,IT,IWRITE)
C
C------------------------------------------------------------------------
```

```
C                                                                       LIV05420
      IF(TEMP.LE.1.D-10) GO TO 410                                      LIV05430
C     ERROR ESTIMATE WAS NOT SMALL REJECT RELABELLING OF THIS           LIV05440
C     T-EIGENVALUE                                                      LIV05450
      IF(IDIST.EQ.1)  WRITE(11,400) J,T0,TEMP                           LIV05460
  400 FORMAT(/' LAST COMPONENT FOR',I6,'TH T-EIGENVALUE',E20.12/' IS TOOLIV05470
     1 LARGE = ',E15.6,' SO DO NOT ACCEPT PRTEST RELABELLING'/)         LIV05480
      MP(J) = 0                                                         LIV05490
      IPROJ = IPROJ - 1                                                 LIV05500
      GO TO 420                                                         LIV05510
C     RELABELLING ACCEPTED                                              LIV05520
  410 NISOM = NISOM + 1                                                 LIV05530
      G(NISOM) = BETAM*TEMP                                             LIV05540
  420 CONTINUE                                                          LIV05550
      IWRITE = IWRITO                                                   LIV05560
C                                                                       LIV05570
      IF(IPROJ.EQ.0) GO TO 460                                          LIV05580
      WRITE(6,430) IPROJ                                                LIV05590
  430 FORMAT(/I6,' T-EIGENVALUES WERE RECLASSIFIED AS GOOD.'/           LIV05600
     1' THESE ARE IDENTIFIED IN FILE 3 BY A T-MULTIPLICITY OF -10'/' USELIV05610
     2R SHOULD INSPECT EACH TO MAKE SURE NEIGHBORS HAVE CONVERGED'/)    LIV05620
C                                                                       LIV05630
      IF(IDIST.EQ.1)  WRITE(11,440) IPROJ                               LIV05640
  440 FORMAT(/I6,' T-EIGENVALUES WERE RELABELLED AS GOOD'/              LIV05650
     1' BELOW IS CORRECTED T-MULTIPLICITY PATTERN'/)                    LIV05660
C                                                                       LIV05670
      WRITE(6,450) NDIS, (MP(I), I=1,NDIS)                              LIV05680
      IF(IDIST.EQ.1)  WRITE(11,450) NDIS, (MP(I), I=1,NDIS)             LIV05690
  450 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/       LIV05700
     1 6X, ' (-10) MEANS SPURIOUS T-EIGENVALUE RELABELLED AS GOOD'/(20I4LIV05710
     1))                                                                LIV05720
C                                                                       LIV05730
C     RECALCULATE MINGAPS FOR DISTINCT T(1,MEV) EIGENVALUES.            LIV05740
  460 NM1 = NDIS - 1                                                    LIV05750
      G(NDIS) = VS(NM1)-VS(NDIS)                                        LIV05760
      G(1) = VS(2)-VS(1)                                                LIV05770
C                                                                       LIV05780
      DO 470 J = 2,NM1                                                  LIV05790
      T0 = VS(J)-VS(J-1)                                                LIV05800
      T1 = VS(J+1)-VS(J)                                                LIV05810
      G(J) = T1                                                         LIV05820
      IF (T0.LT.T1) G(J) = -T0                                          LIV05830
  470 CONTINUE                                                          LIV05840
      IF(IPROJ.EQ.0) GO TO 500                                          LIV05850
C     WRITE TO FILE 4 ERROR ESTIMATES FOR THOSE T-EIGENVALUES RELABELLEDLIV05860
      NGOOD = 0                                                         LIV05870
      DO 480 J = 1,NDIS                                                 LIV05880
      IF(MP(J).EQ.0) GO TO 480                                          LIV05890
      NGOOD = NGOOD + 1                                                 LIV05900
      IF(MP(J).NE.ITEN) GO TO 480                                       LIV05910
      T0 = VS(J)                                                        LIV05920
      NISO = NISO + 1                                                   LIV05930
      NISOM = MEV + NISO                                                LIV05940
      WRITE(4,490) NGOOD,T0,G(NISOM),G(J)                               LIV05950
  480 CONTINUE                                                          LIV05960
```

```
  490 FORMAT(I10,E25.16,2E14.3)
C
  500 CONTINUE
C
C     WRITE THE GOOD T-EIGENVALUES TO FILE 3.  FIRST TRANSFER THEM
C     TO V2 AND THEIR T-MULTIPLICITIES TO THE CORRESPONDING POSITIONS
C     IN MP AND COMPUTE THE A-MINGAPS, THE MINIMAL GAPS BETWEEN THE
C     GOOD T-EIGENVALUES.  THESE GAPS WILL BE PUT IN THE ARRAY G.
C     SINCE G CURRENTLY CONTAINS THE MINIMAL GAPS BETWEEN THE DISTINCT
C     EIGENVALUES OF THE T-MATRIX, THESE GAPS WILL FIRST BE
C     TRANSFERRED TO V1.  NOTE THAT V1<0 MEANS THAT THAT MINIMAL GAP
C     IN THE T-MATRIX IS DUE TO A SPURIOUS T-EIGENVALUE.
C     ALL THIS INFORMATION IS PRINTED TO FILE 3
C
  510 CONTINUE
      NG = 0
      DO 520 I = 1,NDIS
      IF (MP(I).EQ.0) GO TO 520
      NG = NG+1
      MP(NG) = MP(I)
      V2(NG) = VS(I)
      TEMP = G(I)
      TEMP = DABS(TEMP)
      J = I+1
      IF (G(I).LT.ZERO) J = I-1
      IF (MP(J).EQ.0) TEMP = -TEMP
      V1(NG) = TEMP
  520 CONTINUE
C
      WRITE(6,530)MEV
  530 FORMAT(//' T-EIGENVALUE CALCULATION AT MEV = ',I6,'  IS COMPLETE'/
     1)
C
C     NG = NUMBER OF COMPUTED DISTINCT GOOD T-EIGENVALUES.  NEXT
C     GENERATE GAPS BETWEEN GOOD T-EIGENVALUES (BIMINGAPS) AND PUT THEM
C     G.  G(J) < 0 MEANS THE MINIMAL GAP IS DUE TO THE LEFT-HAND GAP.
C
C     GG(J) = BIMINGAP FOR EIGENVALUES OF B-INVERSE MATRIX.
      NGM1 = NG - 1
      GG(NG) = V2(NGM1)-V2(NG)
      GG(1) = V2(2)-V2(1)
C
      DO 540 J = 2,NGM1
      T0 = V2(J)-V2(J-1)
      T1 = V2(J+1)-V2(J)
      GG(J) = T1
      IF (T0.LT.T1) GG(J) = -T0
  540 CONTINUE
C
C     WRITE GOOD BI EIGENVALUES TO FILE 3.
      WRITE(3,550)NG,NDIS,MEV,N,SVSEED,MATNO,MULTOL,IB,BTOL,SHIFT
  550 FORMAT(4I6,I12,I8, ' = NG,NDIS,MEV,N,SVSEED,MATNO'/
     1 E20.12,I6,2E10.3,' = MULTOL,I(MINBETA),BTOL,SHIFT')
C
C     CALCULATE EIGENVALUES OF ORIGINAL INPUT MATRIX CORRESPONDING
```

```
C     TO COMPUTED GOOD T-EIGENVALUES.
      TEMP = -ONE/S0
      DO 560 K = 1,NG
      VS(K) = (SHIFT - (ONE/V2(K)))*TEMP
  560 CONTINUE
C
      NGM1 = NG - 1
      G(NG) = DABS(VS(NGM1)-VS(NG))
      G(1) = DABS(VS(2)-VS(1))
C
      DO 570 J = 2,NGM1
      T0 = DABS(VS(J)-VS(J-1))
      T1 = DABS(VS(J+1)-VS(J))
      G(J) = T1
      IF (T0.LT.T1) G(J)=-T0
  570 CONTINUE
C
      WRITE(3,580)
  580 FORMAT(' EVNO',1X,'TMULT',20X,'EVBI',5X,'BIGAP',6X,'AGAP',6X,
     1'TGAP',12X,'EVA')
C
      WRITE(3,590)(I,MP(I),V2(I),GG(I),G(I),V1(I),VS(I),  I=1,NG)
  590 FORMAT(2I5,E25.16,3E10.3,E15.8)
C
C     IF CONVERGENCE FLAG ICONV.NE.1 AND NUMBER OF T-MATRICES
C     CONSIDERED TO DATE IS LESS THAN NUMBER ALLOWED, INCREMENT MEV.
C     AND LOOP BACK TO 210 TO REPEAT COMPUTATIONS.  RESTORE BETA(MEV+1).
C
      BETA(MP1) = BETAM
      IF (MMB.LT.NMEVS.AND.ICONV.NE.1) GO TO 190
C     END OF LOOP ON DIFFERENT SIZE T-MATRICES ALLOWED.
  600 CONTINUE
C
      IF(ISTOP.EQ.0)  WRITE(6,610)
  610 FORMAT(/' T-MATRICES (ALPHA AND BETA) ARE NOW AVAILABLE, TERMINATE
     1')
      IF (IHIS.EQ.1.AND.KMAX.NE.MOLD) WRITE(1,620)
  620 FORMAT(/' ABOVE ARE THE FOLLOWING VECTORS '/
     1 '  ALPHA(I), I = 1,KMAX'/
     2 '  BETA(I), I = 1,KMAX+1'/
     3 ' FINAL TWO LANCZOS VECTORS OF ORDER N FOR I = KMAX,KMAX+1'/
     4 ' ALL VECTORS IN THIS FILE HAVE HEX FORMAT 4Z20'/
     5 ' ----- END OF FILE 1 NEW ALPHA, BETA HISTORY---------------'///)
C
      IF (ISTOP.EQ.0) GO TO 700
C
      WRITE(3,630)
  630 FORMAT(/' ABOVE ARE COMPUTED GOOD T-EIGENVALUES'/
     1 ' NG = NUMBER OF GOOD T-EIGENVALUES COMPUTED'/
     2 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)'/
     3 ' N = ORDER OF A,  MATNO = MATRIX IDENT'/
     3 ' THERE ARE TWO SETS OF EIGENVALUES, THOSE FOR A AND THOSE FOR'/
     3 ' B-INVERSE WHERE C=S0*A + SHIFT*I, B = P*C*P-TRANS = L*L-TRANS'/
     3 ' THE LANCZOS RECURSIONS ARE APPLIED TO B-INVERSE, USING L'/
     3 ' IF EVBI IS A GOOD EIGENVALUE OF B-INVERSE, THEN EVA IS A'/
```

```
     3 ' GOOD EIGENVALUE OF A WHERE EVA = (SHIFT-ONE/EVBI)(-ONE/SO)'/   LIV07070
     4 ' MULTOL = T-MULTIPLICITY TOLERANCE FOR T-EIGENVALUES IN BISEC'/ LIV07080
     4 ' TMULT IS THE T-MULTIPLICITY OF GOOD T-EIGENVALUE'/             LIV07090
     5 ' TMULT = -1 MEANS SPURIOUS T-EIGENVALUE TOO CLOSE'/             LIV07100
     6 ' DO NOT COMPUTE ERROR ESTIMATES FOR SUCH EIGENVALUES'/          LIV07110
     7 ' AMINGAP = MINIMAL GAP BETWEEN THE COMPUTED A-EIGENVALUES'/     LIV07120
     8 ' AMINGAP .LT. 0. MEANS MINIMAL GAP IS DUE TO LEFT-HAND GAP'/    LIV07130
     9 ' TMINGAP= MINIMAL GAP W.R.T.  DISTINCT EIGENVALUES IN T(1,MEV)'/LIV07140
     1 ' TMINGAP .LT. 0. MEANS MINGAP IS DUE TO SPURIOUS T-EIGENVALUE'/ LIV07150
     2 ' ----- END OF FILE 3 GOODEIGENVALUES-----------------------'///)LIV07160
C                                                                       LIV07170
      IF (IDIST.EQ.1) WRITE(11,640)                                     LIV07180
  640 FORMAT(/' ABOVE ARE THE DISTINCT EIGENVALUES OF T(1,MEV).'/       LIV07190
     2 ' THE FORMAT IS      T-MULTIPLICITY    T-EIGENVALUE   TMINGAP'/  LIV07200
     3 '        THIS FORMAT IS REPEATED TWICE ON EACH LINE.'/           LIV07210
     4 ' T-MULTIPLICITY = -1 MEANS THAT THE SUBROUTINE ISOEV HAS TAGGED'LIV07220
     5/'   THIS SIMPLE T-EIGENVALUE AS HAVING A VERY CLOSE SPURIOUS'/   LIV07230
     6 '   T-EIGENVALUE SO THAT NO ERROR ESTIMATE WILL BE COMPUTED'/    LIV07240
     7 '   FOR THAT EIGENVALUE IN SUBROUTINE INVERR.'/                  LIV07250
     8 ' TMINGAP .LT. 0, TMINGAP IS DUE TO LEFT GAP .GT. 0, RIGHT GAP.'/LIV07260
     9 ' EACH OF THE DISTINCT T-EIGENVALUE TABLES IS FOLLOWED'/         LIV07270
     9 ' BY THE T-MULTIPLICITY PATTERN.'/                               LIV07280
     1 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV).'/  LIV07290
     2 ' NG = NUMBER OF GOOD T-EIGENVALUES. '/                          LIV07300
     3 ' NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES. '/               LIV07310
     4 ' NISO ALSO IS THE COUNT OF +1 ENTRIES IN T-MULTIPLICITY PATTERN.LIV07320
     5 '/' ----- END OF FILE 11 DISTINCT T-EIGENVALUES--------------'///LIV07330
     6)                                                                 LIV07340
C                                                                       LIV07350
      IF(NIOS.NE.0)  WRITE(4,650)                                       LIV07360
  650 FORMAT(/' ABOVE ARE THE ERROR ESTIMATES OBTAINED FOR THE ISOLATED LIV07370
     1GOOD T-EIGENVALUES'/                                              LIV07380
     1' OBTAINED VIA INVERSE ITERATION IN THE SUBROUTINE INVERR.'/      LIV07390
     1' ALL OTHER GOOD T-EIGENVALUES HAVE CONVERGED.'/                  LIV07400
     2' ERROR ESTIMATE = BETAM*ABS(UM)'/                                LIV07410
     2' WHERE BETAM = BETA(MEV+1) AND UM = U(MEV).'/                    LIV07420
     3' U = UNIT EIGENVECTOR OF T WHERE T*U = EV*U AND EV = ISOLATED GOOLIV07430
     3D T-EIGENVALUE.'/                                                 LIV07440
     4' TMINGAP = GAP TO NEAREST DISTINCT EIGENVALUE OF T(1,MEV).'/     LIV07450
     5' TMINGAP .LT. 0. MEANS MINGAP IS DUE TO LEFT NEIGHBOR'/          LIV07460
     6' ERROR ESTIMATE L.T. 0 MEANS INVERSE ITERATION DID NOT CONVERGE'/LIV07470
     7' ------ END OF FILE 4 ERRINV -------------------------------'//) LIV07480
      GO TO 700                                                         LIV07490
C                                                                       LIV07500
  660 CONTINUE                                                          LIV07510
C                                                                       LIV07520
      IBB = IABS(IBMEV)                                                 LIV07530
      IF (IBMEV.LT.0) WRITE(6,670) MEV,IBB,BETA(IBB)                    LIV07540
  670 FORMAT(/' PROGRAM TERMINATES BECAUSE MEV REQUESTED = ',I6,' IS .GTLIV07550
     1',I6/' AT WHICH AN ABNORMALLY SMALL BETA = ' , E13.4,' OCCURRED'/)LIV07560
      GO TO 700                                                         LIV07570
C                                                                       LIV07580
  680 IF (NDIS.EQ.0.AND.ISTOP.GT.0) WRITE(6,690)                        LIV07590
  690 FORMAT(/' INTERVALS SPECIFIED FOR BISECT DID NOT CONTAIN ANY T-EIGLIV07600
     1ENVALUES'/' PROGRAM TERMINATES')                                  LIV07610
```

```
C                                                                       LIV07620
  700 CONTINUE                                                          LIV07630
C                                                                       LIV07640
      STOP                                                              LIV07650
C-----END OF LIVAL (INVERSES OF REAL SYMMETRIC MATRICES)--------------- LIV07660
      END                                                               LIV07670
```

4.3 LIVEC: Main Program, Eigenvector Computations

```
C-----LIVEC (EIGENVECTORS OF INVERSES OF REAL SYMMETRIC MATRICES)-------
C  Authors:  Jane Cullum and Ralph A. Willoughby (deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C         REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4
C         Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C         Applied Mathematics, 2002. SIAM Publications,
C         Philadelphia, PA. USA
C
C     CONTAINS MAIN PROGRAM FOR COMPUTING AN EIGENVECTOR CORRESPONDING
C     TO EACH OF A SET OF EIGENVALUES WHICH HAVE BEEN COMPUTED
C     ACCURATELY BY THE CORRESPONDING LANCZOS EIGENVALUE PROGRAM
C     (LIVAL) FOR FACTORED INVERSES OF REAL, SYMMETRIC MATRICES.
C     THIS PROGRAM COULD BE MODIFIED TO COMPUTE ADDITIONAL EIGENVECTORS
C     FOR ANY EIGENVALUES WHICH ARE MULTIPLE EIGENVALUES OF THE
C     A-MATRIX.  THE AMOUNT OF ADDITIONAL COMPUTATION REQUIRED BY
C     SUCH A MODIFICATION DEPENDS UPON THE GIVEN A-MATRIX AND UPON
C     WHICH PORTION OF THE SPECTRUM IS INVOLVED.
C
C     THESE LANCZOS EIGENVECTOR COMPUTATIONS ASSUME THAT EACH
C     EIGENVALUE THAT IS BEING CONSIDERED HAS CONVERGED AS AN
C     EIGENVALUE OF THE LANCZOS TRIDIAGONAL MATRICES.
C
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE
C     CONSTRUCTIONS
C
C     1.  DATA/MACHEP/ STATEMENT
C     2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)
C     3.  FORMAT(20A4) USED WITH THE EXPLANATORY HEADER, EXPLAN
C     4.  HEXADECIMAL FORMAT (4Z20) USED FOR ALPHA/BETA FILES 1 AND 2.
C
C     IMPORTANT NOTE:  PROGRAM ALLOWS ENLARGEMENT OF THE ALPHA, BETA
C     ARRAYS.  IN PARTICULAR, IF ANY ONE OF THE EIGENVALUES SUPPLIED
C     IS T-SIMPLE AND NOT CLOSE TO A SPURIOUS EIGENVALUE, THE PROGRAM
C     REQUIRES THAT KMAX BE AT LEAST 11*MEV/8 + 12.  IF KMAX IS NOT
```

```
C     THIS LARGE, THEN THE PROGRAM RESETS KMAX TO THIS SIZE
C     AND EXTENDS THE ALPHA, BETA HISTORY IF REQUIRED.
C     THUS, THE DIMENSIONS OF THE ALPHA AND BETA ARRAYS MUST BE
C     LARGE ENOUGH TO ALLOW FOR THIS POSSIBILITY.
C     REMEMBER THAT THE BETA ARRAY, BETA(J), IS SUCH THAT
C     J = 1,..., KMAX+1,  SO IF THE KMAX USED BY THE PROGRAM
C     IS TO BE 3000, THEN BETA MUST BE OF LENGTH AT LEAST 3001.
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  ALPHA(1000),BETA(1001)
      DOUBLE PRECISION  V1(2200),V2(2200),VS(2200)
      DOUBLE PRECISION  RITVEC(40000),TVEC(5000)
      DOUBLE PRECISION  GOODA(50),GOODBI(50),EVNEW(50),TLAST(50)
      DOUBLE PRECISION  EVAL,EVALN,TOLN,TTOL,ERTOL,ALFA,BATA
      DOUBLE PRECISION  MULTOL,SCALE0,STUTOL,BTOL,LB,UB,S0,RNORME
      DOUBLE PRECISION  ONE,ZERO,MACHEP,EPSM,TEMP,SUM,SHIFT,SHIFTO
      DOUBLE PRECISION  RELTOL,ERROR,TERROR,BKMIN,ERRMIN
      REAL G(5000),AMINGP(50),TMINGP(50),BIERR(50),BIEVER(50),BIERRG(50)
      REAL TERR(50),RNORM(50),TBETA(50),BIMING(50)
      REAL EXPLAN(20)
      INTEGER  MP(50),IDELTA(50)
      INTEGER  M1(50),M2(50),MA(50),ML(50),MINT(50),MFIN(50)
      INTEGER  SVSEED,SVSOLD,RHSEED
      INTEGER  MBOUND,NTVCON,SVTVEC,TVSTOP,LVCONT,ERCONT,TFLAG
      DOUBLE PRECISION  FINPRO
      DOUBLE PRECISION  DABS, DMAX1, DSQRT, DFLOAT
      REAL ABS
      INTEGER  IABS
      EXTERNAL BSOLV
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.D0*MACHEP
C-----------------------------------------------------------------------
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     1.  ALPHA:  >= KMAXN,  BETA: >= (KMAXN+1) WHERE KMAXN, THE
C              LARGEST SIZE T-MATRIX CONSIDERED BY THE PROGRAM,
C              IS THE LARGER OF THE SIZE OF THE ALPHA, BETA HISTORY
C              PROVIDED ON FILE 2 (IF ANY ) AND THE SIZE WHICH THE
C              PROGRAM SPECIFIES INTERNALLY, THIS LATTER IS ALWAYS
C              < = 11*MEV / 8  +  12, WHERE MEV IS THE SIZE
C              T-MATRIX THAT WAS USED IN THE CORRESPONDING EIGENVALUE
C              COMPUTATIONS.
C     2.  V1:  >= MAX(N,KMAX)
C     3.  V2, VS:   >= N
C     4.  G:  >= MAX(N,KMAX)
C     5.  RITVEC:  >= N*NGOOD, WHERE NGOOD IS NUMBER OF EIGENVALUES
C                SUPPLIED TO THIS PROGRAM.
C     6.  TVEC:  >= CUMULATIVE LENGTH OF ALL THE T-EIGENVECTORS
C         NEEDED TO GENERATE THE DESIRED RITZ VECTORS.  AN EDUCATED
C         GUESS AT AN APPROPRIATE LENGTH CAN BE OBTAINED BY RUNNING THE
C         PROGRAM WITH THE FLAG MBOUND = 1 AND MULTIPLYING THE
C         RESULTING SIZE BY 5/4.
C     7.  GOODA, GOODBI, EVNEW, AMINGP, TMINGP, TERR, RNORM,
C         TBETA, TLAST, BIERR, BIERRG, MP, MA, M1, M2, MINT,
C         MFIN AND IDELTA MUST BE OF DIMENSION AT LEAST NGOOD.
```

```
C
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LANCZOS PROCEDURE FOR FACTORED INVERSES OF REAL SYMMETRI
     1C MATRICES'/'      COMPUTE EIGENVECTORS'/)
C
C     SET PROGRAM PARAMETERS
C     USER MUST NOT MODIFY SCALE0
      SCALE0 = 5.0D0
      ZERO = 0.0D0
      ONE = 1.0D0
      MPMIN = -1000
C     CONVERGENCE TOLERANCE FOR T-EIGENVECTORS FOR RITZ COMPUTATIONS
      ERTOL = 1.D-10
C
C     READ USER-SPECIFIED PARAMETER FROM INPUT FILE 5 (FREE FORMAT)
C
C     READ USER-PROVIDED HEADER FOR RUN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C     READ IN MATNO = MATRIX/RUN IDENTIFICATION NUMBER AND
C     N = ORDER OF A-MATRIX
C     READ IN SCALE (S0) AND SHIFT (SHIFT) APPLIED TO GIVEN
C     MATRIX AND FLAG JPERM.  JPERM = (0,1):
C     JPERM = 1 MEANS THAT A-MATRIX HAS BEEN PERMUTED.
C
      READ(5,20) EXPLAN
      READ(5,*) MATNO,N,S0,SHIFT,JPERM
C
C     READ IN THE MAXIMUM PERMISSIBLE DIMENSIONS FOR THE TVEC ARRAY
C     (MDIMTV), FOR THE RITVEC ARRAY (MDIMRV), AND FOR THE BETA
C     ARRAY (MBETA).
C
      READ(5,20) EXPLAN
      READ(5,*) MDIMTV, MDIMRV, MBETA
C
C     READ IN RELATIVE TOLERANCE (RELTOL) USED IN DETERMINING
C     APPROPRIATE SIZES FOR THE T-MATRICES USED IN THE EIGENVECTOR
C     COMPUTATIONS
C
      READ(5,20) EXPLAN
      READ(5,*) RELTOL
C
C     SET FLAGS TO 0 OR 1:
C     MBOUND = 1:  PROGRAM TERMINATES AFTER COMPUTING 1ST GUESSES
C                  ON APPROPRIATE T-SIZES FOR USE IN THE RITZ VECTOR
C                  COMPUTATIONS
C     NTVCON = 0:  PROGRAM TERMINATES IF THE TVEC ARRAY IS NOT
C                  LARGE ENOUGH TO HOLD ALL THE T-EIGENVECTORS REQUIRED.
C     SVTVEC = 0:  THE T-EIGENVECTORS ARE NOT WRITTEN TO FILE 11
C                  UNLESS TVSTOP = 1
C     SVTVEC = 1:  WRITE THE T-EIGENVECTORS TO FILE 11.
```

```
C     TVSTOP = 1:  PROGRAM TERMINATES AFTER COMPUTING THE
C                  T-EIGENVECTORS
C     LVCONT = 0:  PROGRAM TERMINATES IF THE NUMBER OF T-EIGENVECTORS
C                  COMPUTED IS NOT EQUAL TO THE NUMBER OF RITZ
C                  VECTORS REQUESTED.
C     ERCONT = 0:  MEANS FOR ANY GIVEN EIGENVALUE, A RITZ VECTOR
C                  WILL NOT BE COMPUTED FOR THAT EIGENVALUE UNLESS
C                  A T-EIGENVECTOR HAS BEEN IDENTIFIED WITH A LAST
C                  COMPONENT WHICH SATISFIES THE SPECIFIED
C                  CONVERGENCE CRITERION.
C     ERCONT = 1:  MEANS FOR ANY GIVEN EIGENVALUE, A RITZ VECTOR
C                  WILL BE COMPUTED.  IF A T-EIGENVECTOR CANNOT
C                  BE IDENTIFIED WHICH SATISFIES THE LAST
C                  COMPONENT CRITERION, THEN THE PROGRAM WILL
C                  USE THE T-VECTOR THAT CAME CLOSEST TO
C                  SATISFYING THE CRITERION
C     IWRITE = 1:  EXTENDED OUTPUT OF INTERMEDIATE COMPUTATIONS
C                  IS WRITTEN TO FILE 6
C     IREAD = 0:   ALPHA/BETA FILE IS REGENERATED.
C     IREAD = 1:   ALPHA/BETA FILE USED IN EIGENVALUE COMPUTATIONS
C                  IS READ IN AND EXTENDED IF NECESSARY.  IN BOTH
C                  CASES IREAD = 0 OR 1, THE LANCZOS VECTORS ARE
C                  ALWAYS REGENERATED FOR THE RITZ VECTOR
C                  COMPUTATIONS
C
      READ(5,20) EXPLAN
      READ(5,*) MBOUND,NTVCON,SVTVEC,IREAD
C
      READ(5,20) EXPLAN
      READ(5,*) TVSTOP,LVCONT,ERCONT,IWRITE
      IF (TVSTOP.EQ.1) SVTVEC = 1
C
C     READ IN SEED (RHSEED) FOR GENERATING RANDOM STARTING VECTOR
C     FOR THE INVERSE ITERATION ON THE T-MATRICES.
C
      READ(5,20) EXPLAN
      READ(5,*) RHSEED
C
C-----------------------------------------------------------------------
C     INITIALIZE THE ARRAYS THAT DEFINE THE FACTORIZATION OF
C     THE B-MATRIX AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS
C     TO THE SUBROUTINE BSOLV.
C
      CALL USPEC(N,MATNO)
C-----------------------------------------------------------------------
C     MASK UNDERFLOW AND OVERFLOW
C
      CALL MASK
C-----------------------------------------------------------------------
C     WRITE RUN PARAMETERS OUT TO FILE 6
C
      WRITE(6,30) MATNO,N,JPERM
   30 FORMAT(/4X,'MATRIX IDENTIFICATION NO.',4X,'SIZE OF A-MATRIX',4X,
     1'JPERM'/I29,I21,I9)
C
```

```
      WRITE(6,40) S0,SHIFT                                              LIV02120
   40 FORMAT(/4X,'SCALE APPLIED TO MATRIX',4X,'SHIFT APPLIED TO MATRIX'/LIV02130
     1E27.4,E27.4)                                                      LIV02140
C                                                                       LIV02150
      WRITE(6,50) MBOUND,NTVCON,SVTVEC,IREAD                            LIV02160
   50 FORMAT(/3X,'MBOUND',3X,'NTVCON',3X,'SVTVEC',3X,'IREAD'/3I9,I8/)   LIV02170
C                                                                       LIV02180
      WRITE(6,60) TVSTOP,LVCONT,ERCONT,IWRITE                           LIV02190
   60 FORMAT(3X,'TVSTOP',3X,'LVCONT',3X,'ERCONT',3X,'IWRITE'/4I9)       LIV02200
C                                                                       LIV02210
      WRITE(6,70) MDIMTV,MDIMRV,MBETA                                   LIV02220
   70 FORMAT(/3X,'MDIMTV',3X,'MDIMRV',3X,'MBETA'/2I9,I8)                LIV02230
C                                                                       LIV02240
      WRITE(6,80) RELTOL,RHSEED                                         LIV02250
   80 FORMAT(/7X,'RELTOL',3X,'RHSEED'/E13.4,I9)                         LIV02260
C                                                                       LIV02270
C     FROM FILE 3 READ IN THE NUMBER OF EIGENVALUES (NGOOD) FOR WHICH   LIV02280
C     EIGENVECTORS ARE REQUESTED, THE ORDER (MEV) OF THE LANCZOS        LIV02290
C     TRIDIAGONAL MATRIX USED IN COMPUTING THESE EIGENVALUES, THE       LIV02300
C     ORDER (NOLD) OF THE USER-SPECIFIED MATRIX USED IN THE EIGENVALUE  LIV02310
C     COMPUTATIONS, THE SEED (SVSEED) USED FOR GENERATING THE STARTING  LIV02320
C     VECTOR THAT WAS USED IN THOSE LANCZOS EIGENVALUE COMPUTATIONS,    LIV02330
C     AND THE MATRIX/RUN IDENTIFICATION NUMBER (MATOLD) USED IN THOSE   LIV02340
C     COMPUTATIONS.  ALSO READ IN THE NUMBER (NDIS) OF DISTINCT         LIV02350
C     EIGENVALUES OF T(1,MEV) THAT WERE COMPUTED BUT THIS VALUE IS      LIV02360
C     NOT USED IN THE EIGENVECTOR COMPUTATIONS.                         LIV02370
C                                                                       LIV02380
      READ(3,90) NGOOD,NDIS,MEV,NOLD,SVSEED,MATOLD                      LIV02390
   90 FORMAT(4I6,I12,I8)                                                LIV02400
C                                                                       LIV02410
C     READ IN THE MULTIPLICITY TOLERANCE USED IN THE BISEC SUBROUTINE   LIV02420
C     DURING THE COMPUTATION OF THE GIVEN EIGENVALUES.                  LIV02430
C     ALSO READ IN THE FLAG IB.  IF IB < 0, THEN SOME BETA(I) IN THE    LIV02440
C     T-MATRIX FILE PROVIDED ON FILE 2 FAILED THE ORTHOGONALITY         LIV02450
C     TEST IN THE TNORM SUBROUTINE.  USER SHOULD NOTE THAT THIS VECTOR  LIV02460
C     PROGRAM PROCEEDS INDEPENDENTLY OF THE SIZE OF THE BETA USED.      LIV02470
C                                                                       LIV02480
      READ(3,100) MULTOL,IB,BTOL,SHIFTO                                 LIV02490
  100 FORMAT(E20.12,I6,2E10.3)                                          LIV02500
C                                                                       LIV02510
      TEMP = DFLOAT(MEV+1000)                                           LIV02520
      TTOL = MULTOL/TEMP                                                LIV02530
C                                                                       LIV02540
      WRITE(6,110) MULTOL,TTOL                                          LIV02550
  110 FORMAT(/' T-MULTIPLICITY TOLERANCE USED IN THE EIGENVALUE COMPUTATLIV02560
     1IONS WAS',E13.4/' SCALED MACHINE EPSILON TTOL IS',E13.4)          LIV02570
C                                                                       LIV02580
C     CONTINUE WRITE TO FILE 6 OF THE PARAMETERS FOR THIS RUN           LIV02590
C                                                                       LIV02600
      NG = NGOOD                                                        LIV02610
      WRITE(6,120)NG,NDIS,MEV,NOLD,MATOLD,SVSEED,IB,MULTOL,BTOL,SHIFTO  LIV02620
  120 FORMAT(/' EIGENVALUES ARE READ IN FROM FILE 3.  THE HEADER IS'/   LIV02630
     1 4X,'NG',2X,'NDIS',3X,'MEV',2X,'NOLD',2X,'MATOLD',6X,'SVSEED'/    LIV02640
     1 4I6,I8,I12/                                                      LIV02650
     1 6X,'IB',6X,'MULTOL',8X,'BTOL',6X,'SHIFTO'/                       LIV02660
```

```
     1 I8,E12.3,E12.3,E12.3/)                                           LIV02670
C                                                                       LIV02680
C     IS THE ARRAY RITVEC LONG ENOUGH TO HOLD ALL OF THE DESIRED        LIV02690
C     RITZ VECTORS (APPROXIMATE EIGENVECTORS)?                          LIV02700
C                                                                       LIV02710
      NMAX = NGOOD*N                                                    LIV02720
      IF(MBOUND.EQ.1) GO TO 130                                         LIV02730
      IF(TVSTOP.NE.1.AND.NMAX.GT.MDIMRV) GO TO 1430                     LIV02740
C                                                                       LIV02750
C     CHECK THAT THE ORDER N AND THE MATRIX IDENTIFICATION NUMBER       LIV02760
C     MATNO SPECIFIED BY THE USER AGREE WITH THOSE READ IN FROM FILE 3. LIV02770
C                                                                       LIV02780
  130 ITEMP = (NOLD-N)**2+(MATOLD-MATNO)**2                             LIV02790
      IF (ITEMP.NE.0.OR.SHIFTO.NE.SHIFT) GO TO 1450                     LIV02800
C                                                                       LIV02810
C     READ IN FROM FILE 3, THE T-MULTIPLICITIES OF THE EIGENVALUES      LIV02820
C     WHOSE EIGENVECTORS ARE TO BE COMPUTED, THE VALUES OF THESE        LIV02830
C     EIGENVALUES AND THEIR MINIMAL GAPS AS EIGENVALUES OF THE          LIV02840
C     USER-SPECIFIED MATRIX AND AS EIGENVALUES OF THE T-MATRIX.         LIV02850
C                                                                       LIV02860
      READ(3,20) EXPLAN                                                 LIV02870
      READ(3,140) (MP(J),GOODBI(J),BIMING(J),AMINGP(J),TMINGP(J),       LIV02880
     1 J = 1,NGOOD)                                                     LIV02890
  140 FORMAT(5X,I5,E25.16,3E10.3)                                       LIV02900
C                                                                       LIV02910
C                                                                       LIV02920
      DO 150 J=1,NGOOD                                                  LIV02930
  150 GOODA(J) = (ONE/GOODBI(J) - SHIFT)/S0                             LIV02940
C                                                                       LIV02950
      WRITE(6,160) (J,GOODA(J),MP(J),GOODBI(J), J=1,NGOOD)              LIV02960
  160 FORMAT(/' EIGENVALUES READ IN, T-MULTIPLICITIES'/                 LIV02970
     1 4X,' J ',5X,' A-EIGENVALUE',6X,'TMULT',3X,'B-INVERSE EIGENVALUE'/LIV02980
     1(I6,E25.16,I4,E25.16))                                            LIV02990
      WRITE(6,170) (J,GOODBI(J),TMINGP(J),BIMING(J), J=1,NGOOD)         LIV03000
  170 FORMAT(/' B(INVERSE) EIGENVALUES READ IN, T-GAPS AND B(INVERSE)-GALIV03010
     1PS'/4X,' J ',3X,'B-INVERSE EIGENVALUE',6X,'  TMINGAP ',6X,        LIV03020
     1' BIMINGAP '/(I6,E25.16,2E15.4))                                  LIV03030
      WRITE(6,180) (J,GOODA(J),AMINGP(J), J=1,NGOOD)                    LIV03040
  180 FORMAT(/' A-EIGENVALUES READ IN AND A-GAPS'/                      LIV03050
     1 4X,' J ',5X,'A-EIGENVALUE',10X,'  AMINGAP '                      LIV03060
     1/(I6,E25.16,E15.4))                                               LIV03070
C                                                                       LIV03080
C     READ IN ERROR ESTIMATES                                           LIV03090
      WRITE(6,210) MEV,SVSEED                                           LIV03100
C     CHECK WHETHER OR NOT THERE ARE ANY T-ISOLATED EIGENVALUES IN      LIV03110
C     THE EIGENVALUES PROVIDED                                          LIV03120
      DO 190 J=1,NGOOD                                                  LIV03130
      IF(MP(J).EQ.1) GO TO 200                                          LIV03140
  190 CONTINUE                                                          LIV03150
      GO TO 230                                                         LIV03160
  200 READ(4,20) EXPLAN                                                 LIV03170
      READ(4,20) EXPLAN                                                 LIV03180
      READ(4,20) EXPLAN                                                 LIV03190
  210 FORMAT(/' THESE EIGENVALUES WERE COMPUTED USING A T-MATRIX OF     LIV03200
     1ORDER ',I5/' AND SEED FOR RANDOM NUMBER GENERATOR =',I12)         LIV03210
```

```
      READ(4,220) NISO                                                  LIV03220
  220 FORMAT(18X,I6)                                                    LIV03230
      READ(4,20) EXPLAN                                                 LIV03240
      READ(4,20) EXPLAN                                                 LIV03250
      READ(4,20) EXPLAN                                                 LIV03260
  230 DO 260 J=1,NGOOD                                                  LIV03270
      BIERR(J) = 0.D0                                                   LIV03280
      IF(MP(J).NE.1) GO TO 260                                          LIV03290
      READ(4,240) EVAL, BIERR(J)                                        LIV03300
  240 FORMAT(10X,E25.16,E14.3)                                          LIV03310
      IF(DABS(EVAL - GOODBI(J)).LT.1.D-10) GO TO 260                    LIV03320
      WRITE(6,250) EVAL,GOODBI(J)                                       LIV03330
  250 FORMAT(' PROBLEM WITH READ IN OF ERROR ESTIMATES'/' EIGENVALUE REALLIV03340
     1D IN',E20.12,' DOES NOT MATCH GOODBI(J) ='/E20.12)                LIV03350
      GO TO 1670                                                        LIV03360
C                                                                       LIV03370
  260 CONTINUE                                                          LIV03380
C                                                                       LIV03390
      WRITE(6,270) (J,GOODBI(J),BIERR(J), J=1,NGOOD)                    LIV03400
  270 FORMAT(' B(INVERSE) ERROR ESTIMATES '/4X,' J',5X,'EIGENVALUE',10X LIV03410
     1,'ESTIMATE'/(I6,E20.12,E14.3))                                    LIV03420
C                                                                       LIV03430
      IF(IREAD.EQ.0)  GO TO 370                                         LIV03440
C                                                                       LIV03450
C     READ IN THE SIZE OF THE T-MATRIX PROVIDED ON FILE 2.  READ IN     LIV03460
C     THE ORDER OF THE USER-SPECIFIED MATRIX , THE SEED FOR THE         LIV03470
C     RANDOM NUMBER GENERATOR, AND THE MATRIX/TEST IDENTIFICATION       LIV03480
C     NUMBER THAT WERE USED IN THE LANCZOS EIGENVALUE COMPUTATIONS.     LIV03490
C     IF FLAG IREAD = 0, REGENERATE ALPHA,BETA ARRAYS                   LIV03500
C                                                                       LIV03510
      READ(2,280) KMAX,NOLD,SVSOLD,MATOLD,SHIFTO                        LIV03520
  280 FORMAT(2I6,I12,I8,E13.4)                                          LIV03530
C                                                                       LIV03540
      WRITE(6,290) KMAX,NOLD,SVSOLD,MATOLD,SHIFTO                       LIV03550
  290 FORMAT(/' READ IN THE T-MATRICES STORED ON FILE 2'/' FILE 2 HEADERLIV03560
     1 IS'/2X,'KMAX',2X,'NOLD',6X,'SVSOLD',2X,'MATOLD',4X,'SHIFTO'/     LIV03570
     1 2I6,I12,I8,E10.3/)                                               LIV03580
C                                                                       LIV03590
C     CHECK THAT THE ORDER, THE MATRIX/TEST IDENTIFICATION NUMBER       LIV03600
C     AND THE SEED FOR THE RANDOM NUMBER GENERATOR USED IN THE          LIV03610
C     LANCZOS COMPUTATIONS THAT GENERATED THE ALPHA,BETA FILE           LIV03620
C     BEING USED AGREE WITH WHAT THE USER HAS SPECIFIED.                LIV03630
      IF (NOLD.NE.N.OR.MATOLD.NE.MATNO.OR.SVSOLD.NE.SVSEED) GO TO 1470  LIV03640
C                                                                       LIV03650
      KMAX1 = KMAX + 1                                                  LIV03660
C                                                                       LIV03670
C     READ IN THE T-MATRICES FROM FILE 2.  THESE ARE USED TO GENERATE   LIV03680
C     THE T-EIGENVECTORS THAT WILL BE USED IN THE RITZ VECTOR           LIV03690
C     COMPUTATIONS.  ALPHA,BETA MUST BE STORED IN MACHINE FORMAT        LIV03700
C     ((4Z20) ON IBM/3081)                                              LIV03710
C                                                                       LIV03720
      READ(2,300) (ALPHA(J), J=1,KMAX)                                  LIV03730
      READ(2,300) (BETA(J), J=1,KMAX1)                                  LIV03740
  300 FORMAT(4Z20)                                                      LIV03750
C                                                                       LIV03760
```

```
      READ(2,300) (V1(J), J=1,N)
      READ(2,300) (V2(J), J=1,N)
C
C     ENLARGE KMAX IF THE SIZE AT WHICH THE EIGENVALUE
C     COMPUTATIONS WERE PERFORMED IS ESSENTIALLY KMAX AND
C     THERE IS AT LEAST ONE EIGENVALUE THAT IS T-SIMPLE AND
C     T-ISOLATED IN THE SENSE THAT IF ITS NEAREST NEIGHBOR IS
C     TOO CLOSE THEN THAT NEIGHBOR IS A GOOD T-EIGENVALUE.
      DO 310 J = 1,NGOOD
      IF(MP(J).EQ.1) GO TO 330
  310 CONTINUE
      WRITE(6,320)
  320 FORMAT(/' ALL EIGENVALUES USED ARE T-MULTIPLE OR CLOSE TO SPURIOUS
     1 T-EIGENVALUES'/' SO DO NOT CHANGE KMAX')
      IF(KMAX.LT.MEV) GO TO 1490
      GO TO 350
C
  330 KMAXN= 11*MEV/8 + 12
      IF(MBETA.LE.KMAXN) GO TO 1650
      IF(KMAX.GE.KMAXN )  GO TO 350
      WRITE(6,340) KMAX, KMAXN
  340 FORMAT(' ENLARGE KMAX FROM ',I6,' TO ',I6)
      MOLD1 = KMAX + 1
      KMAX = KMAXN
      GO TO 420
C
  350 WRITE(6,360) KMAX
  360 FORMAT(/' T-MATRICES HAVE BEEN READ IN FROM FILE 2'/' THE LARGEST
     1SIZE T-MATRIX ALLOWED IS',I6/)
C
      IF(IREAD.EQ.1) GO TO 440
C
C     REGENERATE THE ALPHA AND BETA
C
  370 MOLD1 = 1
C
      DO 380 J = 1,NGOOD
      IF(MP(J).EQ.1) GO TO 400
  380 CONTINUE
      KMAX = MEV + 12
      WRITE(6,390) KMAX
  390 FORMAT(/' ALL EIGENVALUES FOR WHICH EIGENVECTORS ARE TO BE COMPUTE
     1D ARE EITHER T-MULTIPLE OR CLOSE TO'/' A SPURIOUS T-EIGENVALUE. TH
     1EREFORE SET KMAX = MEV + 12 = ',I7)
      GO TO 420
C
  400 KMAXN = 11*MEV/8 + 12
      IF(MBETA.LE.KMAXN) GO TO 1650
      WRITE(6,410) KMAXN
  410 FORMAT(' SET KMAX EQUAL TO ',I6)
      KMAX = KMAXN
C
  420 WRITE(6,430) MOLD1,KMAX
  430 FORMAT(/' LANCZS SUBROUTINE GENERATES ALPHA(J), BETA(J+1), J =',
     1 I6,' TO ', I6/)
```

```
C
C-----------------------------------------------------------------------
C
      CALL LANCZS(BSOLV,ALPHA,BETA,V1,V2,VS,G,KMAX,MOLD1,N,SVSEED)
C
C-----------------------------------------------------------------------
C
  440 CONTINUE
C
C     THE SUBROUTINE STURMI DETERMINES THE SMALLEST SIZE T-MATRIX FOR
C     WHICH THE EIGENVALUE IN QUESTION IS AN EIGENVALUE (TO WITHIN A
C     GIVEN TOLERANCE) AND IF POSSIBLE THE SMALLEST SIZE T-MATRIX
C     FOR WHICH IT IS A DOUBLE EIGENVALUE (TO WITHIN THE SAME
C     TOLERANCE).  THE SIZE T-MATRIX USED IN THE EIGENVECTOR
C     COMPUTATIONS IS THEN DETERMINED BY LOOPING ON SIZE OF THE
C     T-EIGENVECTORS, USING THE VALUES FROM STURMI TO DETERMINE
C     FIRST GUESSES AT THE APPROPRIATE T-SIZES.
C
C
      STUTOL = SCALE0*MULTOL
      IF(IWRITE.EQ.1) WRITE(6,450)
  450 FORMAT(' FROM STURMI')
      DO 490 J = 1,NGOOD
      EVAL = GOODBI(J)
C     COMPUTE THE TOLERANCES USED BY STURMI TO DETERMINE AN INTERVAL
C     CONTAINING THE EIGENVALUE EVAL.
      TEMP = DABS(EVAL)*RELTOL
      TOLN = DMAX1(TEMP,STUTOL)
C
C-----------------------------------------------------------------------
C
      CALL STURMI(ALPHA,BETA,EVAL,TOLN,EPSM,KMAX,MK1,MK2,IC,IWRITE)
C
C-----------------------------------------------------------------------
C
C     STORE THE COMPUTED ORDERS OF T-MATRICES FOR LATER PRINTOUT
      M1(J) = MK1
      M2(J) = MK2
      ML(J) = (MK1 + 3*MK2)/4
      IF(MK2.EQ.KMAX)  ML(J) = KMAX
C
      IF(IC.GT.0) GO TO 470
C     IC = 0 MEANS THERE WAS NO T-EIGENVALUE IN THE DESIGNATED INTERVAL
C     BY T-SIZE KMAX.  THIS MEANS THAT THE EIGENVALUE PROVIDED HAS
C     NOT YET CONVERGED SO ITS EIGENVECTOR SHOULD NOT BE COMPUTED.
      WRITE(6,460) J,GOODBI(J),MK1,MK2
  460 FORMAT(I6,'TH EIGENVALUE',E20.12,' HAS NOT CONVERGED '/
     1' SO DO NOT COMPUTE ANY T-EIGENVECTOR OR RITZ VECTOR FOR IT'
     1/' MK1 AND MK2 FOR THIS EIGENVALUE WERE',2I6)
      MP(J) = MPMIN
      MA(J) = -2*KMAX
      GO TO 490
C     COMPUTE AN APPROPRIATE SIZE T-MATRIX FOR THE GIVEN EIGENVALUE.
  470 IF(M2(J).EQ.KMAX) GO TO 480
C     M1 AND M2 WERE BOTH DETERMINED
```

```
      MA(J) = (3*M1(J) + M2(J))/4 + 1
      GO TO 490
C     M2 NOT DETERMINED
  480 MA(J) = (5*M1(J))/4 + 1
C
  490 CONTINUE
C
      IF (IWRITE.EQ.1) WRITE(6,500) (MA(JJ), JJ=1,NGOOD)
  500 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZE T-MATRICES'/
     1 ' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/(13I6))
C
C     PRINT OUT TO FILE 10 1ST GUESSES AT SIZES OF THE T-MATRICES TO
C     BE USED IN THE EIGENVECTOR COMPUTATIONS.
C     ACTUAL VALUES USED MAY BE 1/4 OR MORE LARGER THAN THESE VALUES.
      WRITE(10,510) N,KMAX
  510 FORMAT(2I8,' = ORDER OF USER MATRIX AND MAX ORDER OF T(1,MEV)')
C
      WRITE(10,520)
  520 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZE T-MATRICES'/
     1 ' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/)
C
      WRITE(10,530)
  530 FORMAT(4X,'J',7X,'GOODBI(J)',4X,'M1(J)',1X,'M2(J)',1X,'MA(J)')
C
      WRITE(10,540) (J,GOODBI(J),M1(J),M2(J), MA(J), J=1,NGOOD)
  540 FORMAT(I5,E19.12,3I6)
C
      IF(MBOUND.EQ.1) WRITE(10,550)
  550 FORMAT(/' EV = GOODBI(J) IS A GOOD EIGENVALUE OF T(1,MEV)'/
     1 ' M1 = SMALLEST VALUE OF M SUCH THAT T(1,M) HAS AT LEAST'/
     1 '      ONE EIGENVALUE IN THE INTERVAL (EV-TOLN,EV+TOLN)'/
     1 ' TOLN(J) = DMAX1(GOODBI(J)*RELTOL, SCALE0*MULTOL)'/
     1 ' M2 = SMALLEST M (IF ANY) SUCH THAT IN THE ABOVE INTERVAL'/
     1 '      T(1,M) HAS AT LEAST TWO EIGENVALUES '/
     1 ' INITIAL VALUE OF MA(J) IS CHOSEN HEURISTICALLY'/
     1 ' PROGRAM LOOPS ON SIZE OF T-MATRIX TO GET APPROPRIATE SIZE'/
     1 ' END OF SIZES OF T-MATRICES FILE 10'///)
C
C
C     TERMINATE AFTER COMPUTING 1ST GUESSES AT  SIZES OF THE
C     T-MATRICES REQUIRED FOR THE GIVEN EIGENVALUES?
      IF(MBOUND.EQ.1) GO TO 1510
C
C
C     WILL THERE BE ROOM FOR ALL OF THE REQUESTED T-EIGENVECTORS?
      MTOL = 0
      DO 560 J = 1,NGOOD
      IF(MP(J).EQ.MPMIN) GO TO 560
      MTOL = MTOL + IABS(MA(J))
  560 CONTINUE
      MTOL = (5*MTOL)/4
      IF(MTOL.GT.MDIMTV.AND.NTVCON.EQ.0) GO TO 1530
C
C-----------------------------------------------------------------------
C     GENERATE A RANDOM VECTOR TO BE USED REPEATEDLY BY
```

```
C     SUBROUTINE INVERM
C
      IIL = RHSEED
      CALL GENRAN(IIL,G,KMAX)
C
C-----------------------------------------------------------------------
C
C     FOR EACH EIGENVALUE LOOP ON T-EIGENVECTOR COMPUTATIONS TO
C     COMPUTE AN APPROPRIATE T-EIGENVECTOR TO USE IN THE RITZ
C     VECTOR COMPUTATIONS.
C
      MTOL = 0
      NTVEC = 0
      ILBIS = 0
      DO 750 J = 1,NGOOD
      ICOUNT = 0
      ERRMIN = 10.D0
      MABEST = MPMIN
      IF(MP(J).EQ.MPMIN) GO TO 750
      TFLAG = 0
      EVAL = GOODBI(J)
      TEMP = RELTOL*DABS(EVAL)
      UB = EVAL + DMAX1(STUTOL,TEMP)
      LB = EVAL - DMAX1(STUTOL,TEMP)
  570 KMAXU = IABS(MA(J))
C
C     SELECT A SUITABLE INCREMENT FOR THE ORDERS OF THE T-MATRICES
C     TO BE CONSIDERED IN DETERMINING APPROPRIATE SIZES FOR THE RITZ
C     VECTOR COMPUTATIONS.
      IF(ICOUNT.GT.0) GO TO 590
C     SELECT IDELTA(J) BASED UPON THE T-MULTIPLICITY OBTAINED
      IF(M2(J).EQ.KMAX) GO TO 580
C     M2 DETERMINED
      IDELTA(J) = ((3*M1(J) + 5*M2(J))/8  +  1 - IABS(MA(J)))/10 +  1
      GO TO 590
C     M2 NOT DETERMINED
  580 MAMAX = MIN0((11*MEV)/8 + 12, (13*M1(J))/8 + 1)
      IDELTA(J) = (MAMAX - IABS(MA(J)))/10 +  1
  590 ICOUNT = ICOUNT + 1
C
C-----------------------------------------------------------------------
C     TO MIMIMIZE THE EFFECT OF THE ONE-SIDED ACCEPTANCE TEST FOR
C     EIGENVALUES IN THE BISEC SUBROUTINE, RECOMPUTE THE GIVEN
C     EIGENVALUE AT THE SPECIFIED KMAXU
C
      CALL LBISEC(ALPHA,BETA,EPSM,EVAL,EVALN,LB,UB,TTOL,KMAXU,NEVT)
C
C-----------------------------------------------------------------------
C
C     CHECK WHETHER OR NOT GIVEN T-MATRIX HAS AN EIGENVALUE IN THE
C     SPECIFIED INTERVAL AND IF SO WHAT ITS T-MULTIPLICITY IS.
C
      IF(NEVT.EQ.1) GO TO 630
      IF(NEVT.NE.0) GO TO 610
      ILBIS = 1
```

```
      WRITE(6,600) EVAL,KMAXU                                           LIV05970
  600 FORMAT(/' PROBLEM ENCOUNTERED IN RECOMPUTATION OF USER-SUPPLIED EILIV05980
     1GENVALUE',E20.12/' THE SIZE T-MATRIX SPECIFIED',I6,' DOES NOT     LIV05990
     1HAVE AN EIGENVALUE IN THE INTERVAL SPECIFIED'/' THEREFORE NO EIGENLIV06000
     1VECTOR WILL BE COMPUTED FOR THIS PARTICULAR EIGENVALUE'/)         LIV06010
      GO TO 650                                                         LIV06020
C                                                                       LIV06030
  610 IF(NEVT.GT.1)  WRITE(6,620) EVAL,KMAXU                            LIV06040
  620 FORMAT(/' PROBLEM ENCOUNTERED IN RECOMPUTATION OF USER-SUPPLIED   LIV06050
     1EIGENVALUE',E20.12/' FOR THE SIZE T-MATRIX SPECIFIED =',I6,' THE  LIV06060
     1GIVEN EIGENVALUE IS MULTIPLE IN THE INTERVAL SPECIFIED'/' SOMETHINLIV06070
     1G IS WRONG, THEREFORE NO EIGENVECTOR WILL BE COMPUTED FOR THIS EIGLIV06080
     1NVALUE'/)                                                         LIV06090
C                                                                       LIV06100
      MP(J) = MPMIN                                                     LIV06110
      MA(J) = -2*KMAX                                                   LIV06120
      GO TO 750                                                         LIV06130
C                                                                       LIV06140
  630 CONTINUE                                                          LIV06150
      ILBIS = 0                                                         LIV06160
C                                                                       LIV06170
      EVNEW(J) = EVALN                                                  LIV06180
      EVAL = EVALN                                                      LIV06190
      MTOL = MTOL+KMAXU                                                 LIV06200
C                                                                       LIV06210
C     IS THERE ROOM IN TVEC ARRAY FOR THE NEXT T-EIGENVECTOR?           LIV06220
C     IF NOT, SKIP TO RITZ VECTOR COMPUTATIONS.                         LIV06230
      IF (MTOL.GT.MDIMTV) GO TO 760                                     LIV06240
C                                                                       LIV06250
      IT = 3                                                            LIV06260
      KINT = MTOL - KMAXU +1                                            LIV06270
C                                                                       LIV06280
C     RECORD THE BEGINNING AND END OF THE T-EIGENVECTOR BEING COMPUTED  LIV06290
      MINT(J) = KINT                                                    LIV06300
      MFIN(J) = MTOL                                                    LIV06310
C                                                                       LIV06320
C-----------------------------------------------------------------------LIV06330
C     SUBROUTINE INVERM DOES INVERSE ITERATION, I.E. SOLVES             LIV06340
C     (T(1,KMAXU) - EVAL)*U = RHS  FOR EACH EIGENVALUE TO OBTAIN        LIV06350
C     THE DESIRED T-EIGENVECTOR.                                        LIV06360
C                                                                       LIV06370
      IF(IWRITE.EQ.1)   WRITE(6,640) J                                  LIV06380
  640 FORMAT(/I6,'TH EIGENVALUE')                                       LIV06390
C                                                                       LIV06400
      CALL INVERM(ALPHA,BETA,V1,TVEC(KINT),EVAL,ERROR,TERROR,EPSM,      LIV06410
     1 G,KMAXU,IT,IWRITE)                                               LIV06420
C                                                                       LIV06430
C-----------------------------------------------------------------------LIV06440
C                                                                       LIV06450
      TERR(J) = TERROR                                                  LIV06460
      TLAST(J) = ERROR                                                  LIV06470
      KMAXU1 = KMAXU + 1                                                LIV06480
      TBETA(J) = BETA(KMAXU1)*ERROR                                     LIV06490
C                                                                       LIV06500
C     AFTER COMPUTING EACH OF THE T-EIGENVECTORS,                       LIV06510
```

```
C     CHECK THE SIZE OF THE ERROR ESTIMATE, ERROR.
C     IF THIS ESTIMATE IS NOT AS SMALL AS DESIRED AND
C     |MA(J)| < ML(J), ATTEMPT TO INCREASE THE SIZE OF |MA(J)|
C     AND REPEAT THE T-EIGENVECTOR COMPUTATIONS.
C
      IF(ERROR.LT.ERTOL.OR.TFLAG.EQ.1)  GO TO 740
C
      IF(ERROR.GE.ERRMIN) GO TO 650
C     LAST COMPONENT IS LESS THAN MINIMAL TO DATE
      ERRMIN = ERROR
      MABEST = MA(J)
  650 CONTINUE
C
      IF(MA(J).GT.0)  ITEST = MA(J) + IDELTA(J)
      IF(MA(J).LT.0)  ITEST = -(IABS(MA(J)) + IDELTA(J))
      IF(IABS(ITEST).LE.ML(J).AND.ICOUNT.LE.10) GO TO 670
C     NEW MA(J) IS GREATER THAN MAXIMUM ALLOWED.
      IF(ERCONT.EQ.0.OR.MABEST.EQ.MPMIN) GO TO 690
      TFLAG = 1
      MA(J) = MABEST
      IF(ILBIS.EQ.0) MTOL = MTOL - KMAXU
      WRITE(6,660) MA(J)
  660 FORMAT(' 10 ORDERS WERE CONSIDERED.  NONE SATISFIED THE ERROR TEST
     1'/' THEREFORE USE THE BEST ORDER OBTAINED FOR THE EIGENVECTORS'
     1,I6)
      GO TO 570
C
  670 MA(J) = ITEST
C
      MT = IABS(MA(J))
      IF(IWRITE.EQ.1) WRITE(6,680)  MT
  680 FORMAT(/' CHANGE SIZE OF T-MATRIX TO ',I6,' RECOMPUTE T-EIGENVECTO
     1R')
C
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU
C
      GO TO 570
C
C     APPROPRIATE SIZE T-MATRIX WAS NOT OBTAINED
  690 CONTINUE
      WRITE(10,700) J,EVAL,MP(J)
  700 FORMAT(/' ON 10 INCREMENTS NOT ABLE TO IDENTIFY APPROPRIATE SIZE
     1T-MATRIX FOR'/
     1' EIGENVALUE(',I4,') = ',E20.12,' T-MULTIPLICITY =',I4/)
      IF(M2(J).EQ.KMAX) WRITE(10,710)
      IF(M2(J).LT.KMAX) WRITE(10,720)
  710 FORMAT(' ORDERS TESTED RANGED FROM 5*M1(J)/4 TO APPROXIMATELY
     1 MIN(11*MEV/8,13*M1(J)/8)')
  720 FORMAT(' ORDERS TESTED RANGED FROM APPROX. (3*M1(J)+M2(J))/4 TO (3
     1*M1(J)+5*M2(J))/8')
      WRITE(10,730)
  730 FORMAT(' ALLOWING LARGER ORDERS FOR THE T-MATRICES MAY RESULT IN
     1 SUCCESS'/' BUT PROBABLY WILL NOT.  PROBLEM IS PROBABLY DUE TO'
     1 /' LACK OF CONVERGENCE OF GIVEN EIGENVALUE, CHECK THE ERROR ESTIM
     1ATE')
```

```
      MP(J) = MPMIN                                                     LIV07070
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU                               LIV07080
      GO TO 750                                                         LIV07090
  740 NTVEC = NTVEC + 1                                                 LIV07100
C                                                                       LIV07110
  750 CONTINUE                                                          LIV07120
      NGOODC = NGOOD                                                    LIV07130
      GO TO 780                                                         LIV07140
C                                                                       LIV07150
C     COME HERE IF THERE IS NOT ENOUGH ROOM FOR ALL OF T-EIGENVECTORS   LIV07160
  760 NGOODC = J-1                                                      LIV07170
      WRITE(6,770)  J,MTOL,MDIMTV                                       LIV07180
  770 FORMAT(/' NOT ENOUGH ROOM IN TVEC ARRAY FOR ',I4,'TH T-EIGENVECTORLIV07190
     1'/'  TVEC DIMENSION REQUESTED = ',I6,' BUT TVEC HAS DIMENSION ',I6LIV07200
     1/)                                                                LIV07210
      IF(NGOODC.EQ.0)  GO TO 1550                                       LIV07220
      MTOL = MTOL-KMAXU                                                 LIV07230
C                                                                       LIV07240
  780 CONTINUE                                                          LIV07250
C                                                                       LIV07260
C     THE LOOP ON T-EIGENVECTOR COMPUTATIONS IS COMPLETE.               LIV07270
C     WRITE OUT THE SIZE T-MATRICES THAT WILL BE USED FOR               LIV07280
C     THE RITZ VECTOR COMPUTATIONS.                                     LIV07290
C                                                                       LIV07300
      WRITE(10,790)                                                     LIV07310
  790 FORMAT(/' SIZES OF T-MATRICES THAT WILL BE USED IN THE RITZ COMPUTLIV07320
     1ATIONS'/5X,'J',8X,'  GOODBI(J) ',13X,'  GOODA(J) ',7X,'MA(J)')    LIV07330
C                                                                       LIV07340
      WRITE(10,800)   (J,GOODBI(J),GOODA(J),MA(J), J=1,NGOOD)           LIV07350
  800 FORMAT(I6,2E25.14,I6)                                             LIV07360
      WRITE(10,550)                                                     LIV07370
C                                                                       LIV07380
      WRITE(6,810) MTOL                                                 LIV07390
  810 FORMAT(/' THE CUMULATIVE LENGTH OF THE T-EIGENVECTORS IS',I18)    LIV07400
C                                                                       LIV07410
      WRITE(6,820) NTVEC,NGOOD                                          LIV07420
  820 FORMAT(/I6,' T-EIGENVECTORS OUT OF',I6,' REQUESTED WERE COMPUTED')LIV07430
C                                                                       LIV07440
C     SAVE THE T-EIGENVECTORS ON FILE 11?                                LIV07450
      IF(TVSTOP.NE.1.AND.SVTVEC.EQ.0) GO TO 880                         LIV07460
C                                                                       LIV07470
      WRITE(11,830) NTVEC,MTOL,MATNO,SVSEED                             LIV07480
  830 FORMAT(I6,3I12,' = NTVEC,MTOL,MATNO,SVSEED')                      LIV07490
C                                                                       LIV07500
      DO 860 J=1,NGOODC                                                 LIV07510
C     IF MP(J) = MPMIN THEN NO SUITABLE T-EIGENVECTOR IS AVAILABLE      LIV07520
C     FOR THAT EIGENVALUE.                                              LIV07530
      IF(MP(J).EQ.MPMIN) WRITE(11,840) J,MA(J),GOODBI(J),MP(J)          LIV07540
  840 FORMAT(2I6,E20.12,I6/' TH EIGVAL,T-SIZE,EVALUE,FLAG,NO EIGVEC')   LIV07550
      IF(MP(J).NE.MPMIN) WRITE(11,850) J,MA(J),GOODBI(J),MP(J)          LIV07560
  850 FORMAT(I6,I6,E20.12,I6/'  T-EIGENVECTOR, T-SIZE , BI-EIGENVALUE, TLIV07570
     1-MULTIPLICITY')                                                   LIV07580
      IF(MP(J).EQ.MPMIN) GO TO 860                                      LIV07590
      KI = MINT(J)                                                      LIV07600
      KF = MFIN(J)                                                      LIV07610
```

```
C
      WRITE(11,300) (TVEC(K), K=KI,KF)
C
  860 CONTINUE
C
      IF(TVSTOP.NE.1) GO TO 880
C
      WRITE(6,870) TVSTOP, NTVEC,NGOOD
  870 FORMAT(/' USER SET TVSTOP = ',I1/
     1' THEREFORE PROGRAM TERMINATES AFTER T-EIGENVECTOR COMPUTATIONS'/
     1' T-EIGENVECTORS THAT WERE COMPUTED ARE SAVED ON FILE 11'/
     1I8,' T-EIGENVECTORS WERE COMPUTED OUT OF',I7,' REQUESTED'/)
C
      GO TO 1670
C
  880 CONTINUE
C     IF NOT ABLE TO COMPUTE ALL THE REQUESTED T-EIGENVECTORS
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS ANYWAY?
      IF(NTVEC.NE.NGOOD.AND.LVCONT.EQ.0) GO TO 1570
C
C     COMPUTE THE MAXIMUM SIZE OF THE T-MATRIX USED FOR THOSE
C     EIGENVALUES WITH GOOD ERROR ESTIMATES.
C
      KMAXU = 0
      DO 890 J = 1,NGOODC
      MT = IABS(MA(J))
      IF(MT.LT.KMAXU.OR.MP(J).EQ.MPMIN) GO TO 890
      KMAXU = MT
  890 CONTINUE
C
      IF(KMAXU.EQ.0) GO TO 1610
C
      WRITE(6,900) KMAXU
  900 FORMAT(/I6,' = LARGEST SIZE T-MATRIX TO BE USED IN THE RITZ VECTOR
     1 COMPUTATIONS')
C
C     COUNT THE NUMBER OF RITZ VECTORS NOT BEING COMPUTED
      MREJEC = 0
      DO 910 J=1,NGOODC
  910 IF(MP(J).EQ.MPMIN)  MREJEC = MREJEC + 1
      MREJET = MREJEC + (NGOOD-NGOODC)
      IF(MREJET.NE.0) WRITE(6,920) MREJET
  920 FORMAT(/' RITZ VECTORS ARE NOT COMPUTED FOR',I6,' OF THE EIGNEVALU
     1ES'/)
      NACT = NGOODC - MREJEC
      WRITE(6,930) NGOOD,NTVEC,NACT
  930 FORMAT(/I6,' RITZ VECTORS WERE REQUESTED'/I6,' T-EIGENVECTORS WERE
     1 COMPUTED'/I6,' RITZ VECTORS WILL BE COMPUTED'/)
C     CHECK IF THERE ARE ANY RITZ VECTORS TO COMPUTE
      IF(MREJEC.EQ.NGOODC) GO TO 1590
C
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS?
      IF(LVCONT.EQ.0.AND.MREJEC.NE.0) GO TO 1570
C
C     NOW COMPUTE THE RITZ VECTORS.  REGENERATE THE
```

```
C     LANCZOS VECTORS.
C
      DO 940 I = 1,NMAX
  940 RITVEC(I) = ZERO
C
C-----------------------------------------------------------------------
C     REGENERATE THE STARTING VECTOR. THIS MUST BE GENERATED AND
C     NORMALIZED PRECISELY THE WAY IT WAS DONE IN THE EIGENVALUE
C     COMPUTATIONS, OTHERWISE THERE WILL BE A MISMATCH BETWEEN
C     THE T-EIGENVECTORS THAT HAVE BEEN COMPUTED FROM THE T-MATRICES
C     READ IN FROM FILE 2 AND THE LANCZOS VECTORS THAT ARE
C     BEING REGENERATED.
C
      CALL GENRAN(SVSEED,G,N)
C
C-----------------------------------------------------------------------
C
      DO 950 J = 1,N
  950 V2(J) = G(J)
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V2(1),1,V2(1),1)
C-----------------------------------------------------------------------
C
      SUM = ONE/DSQRT(SUM)
C
      DO 960 I = 1,N
      V1(I) = ZERO
  960 V2(I) = V2(I)*SUM
C
      WRITE(6,970)
  970 FORMAT(' STARTING LANCZOS VECTOR HAS BEEN CALCULATED'/)
C
C     LOOP FOR GENERATING RITZ VECTORS  (IVEC = 1,KMAXU)
      IVEC = 1
      BATA = ZERO
C
      GO TO 1050
C
  980 CONTINUE
C
C     SOLVE B*VS = V2 FOR VS
      DO 990 K = 1,N
  990 VS(K) = V2(K)
C
C-----------------------------------------------------------------------
      JBSOLV = 2
      CALL BSOLV(VS,VS,JBSOLV)
C-----------------------------------------------------------------------
C
C     VS = BI*V2   BI = B(INVERSE)
C     COMPUTE V1 = BI*V2 - BATA*V1
      DO 1000 K = 1,N
 1000 V1(K) = VS(K) - BATA*V1(K)
C
```

```
C-----------------------------------------------------------------------
      ALFA = FINPRO(N,V1(1),1,V2(1),1)
C-----------------------------------------------------------------------
C
      DO 1010 J = 1,N
 1010 V1(J) = V1(J)-ALFA*V2(J)
C
C-----------------------------------------------------------------------
      BATA = FINPRO(N,V1(1),1,V1(1),1)
C-----------------------------------------------------------------------
C
      BATA = DSQRT(BATA)
      SUM = ONE/BATA
C
      TEMP = BETA(IVEC)
      TEMP = DABS(BATA - TEMP)/TEMP
      IF (TEMP.LT.1.0D-10)GO TO 1030
C
C     THE BETA BEING REGENERATED DO NOT MATCH THE HISTORY FILE
C     SOMETHING IS WRONG IN THE LANCZOS VECTOR GENERATION
C     PROGRAM TERMINATES FOR USER TO CORRECT THE PROBLEM
C     WHICH MUST BE IN THE STARTING VECTOR GENERATION OR IN
C     THE MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV SUPPLIED.
C     THIS SUBROUTINE MUST BE THE SAME ONE USED IN THE
C     EIGENVALUE COMPUTATIONS OR AGAIN A MISMATCH WILL ENSUE.
C
      WRITE(6,1020) IVEC,BATA,BETA(IVEC),TEMP
 1020 FORMAT(/2X,'IVEC',16X,'BATA',10X,'BETA(IVEC)',14X,'RELDIF'/I6,
     13E20.12/' IN LANCZOS VECTOR REGENERATION THE ENTRIES OF THE TRIDIA
     1GONAL MATRICES BEING'/' GENERATED ARE NOT THE SAME AS THOSE IN THE
     1 MATRIX SUPPLIED ON FILE 2.'/' THEREFORE SOMETHING IS BEING INITIA
     1LIZED OR COMPUTED DIFFERENTLY FROM THE WAY'/' IT WAS COMPUTED IN T
     1HE EIGENVALUE COMPUTATIONS'/' THE PROGRAM TERMINATES FOR THE USER
     1TO DETERMINE WHAT THE PROBLEM IS'/)
      GO TO 1670
C
 1030 CONTINUE
      DO 1040 J = 1,N
      TEMP = SUM*V1(J)
      V1(J) = V2(J)
 1040 V2(J) = TEMP
C
 1050 CONTINUE
C
      LFIN = 0
      DO 1070 J = 1,NGOODC
      LL = LFIN
      LFIN = LFIN + N
C
      IF(IABS(MA(J)).LT.IVEC.OR.MP(J).EQ.MPMIN) GO TO 1070
      II = IVEC + MINT(J) - 1
      TEMP = TVEC(II)
C     II IS THE (IVEC)TH COMPONENT OF THE T-EIGENVECTOR CONTAINED
C     IN TVEC(MINT(J)).
C
```

```
      DO 1060 K = 1,N
      LL = LL + 1
 1060 RITVEC(LL) = TEMP*V2(K) + RITVEC(LL)
C
 1070 CONTINUE
C
      IVEC = IVEC + 1
      IF (IVEC.LE.KMAXU) GO TO 980
C
C     RITZVECTOR GENERATION IS COMPLETE. NORMALIZE EACH RITZVECTOR.
C     NOTE THAT IF CERTAIN RITZ VECTORS WERE NOT COMPUTED THEN THAT
C     PORTION OF THE RITVEC ARRAY WAS NOT UTILIZED.
C
      LFIN = 0
      DO 1140 J = 1,NGOODC
C
      KK = LFIN
      LFIN = LFIN + N
      IF(MP(J).EQ.MPMIN) GO TO 1140
C
      DO 1080 K = 1,N
      KK = KK + 1
      V1(K) = RITVEC(KK)
 1080 VS(K) = V1(K)
C
      IF(JPERM.EQ.0)  GO TO 1090
C
C-----------------------------------------------------------------------
C     V2 = V1 = (L-TRANSPOSE)*V1
      IPERM = 2
      CALL LPERM(V1,V2,IPERM)
C-----------------------------------------------------------------------
C
C     V2 CONTAINS RITZ VECTOR FOR A, VS CONTAINS THE RITZ VECTOR FOR B
C
 1090 CONTINUE
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V1(1),1,V1(1),1)
C-----------------------------------------------------------------------
C
      SUM = DSQRT(SUM)
      RNORM(J) = SUM
      RNORME = DABS(ONE-SUM)
      SUM = ONE/SUM
C
      KK = LFIN - N
      DO 1100 K = 1,N
      KK = KK + 1
      VS(K) = SUM*VS(K)
 1100 RITVEC(KK) = SUM*V1(K)
C
C     VS IS RITZ VECTOR FOR BI:  RITVEC IS RITZ VECTOR FOR A-MATRIX
C     B = SO*P*A*P' + SHIFT*I
C     BIERR = ||BI*VS - GOODBI(J)*VS||
```

```
C     BIEVER = |(VS-TRANS)*BI*VS - GOODBI(J)|
C
C-----------------------------------------------------------------------
C     V1 = (B-INVERSE)*VS
      JBSOLV = 2
      CALL BSOLV(VS,V1,JBSOLV)
C-----------------------------------------------------------------------
C
      EVALN = EVNEW(J)
C
C-----------------------------------------------------------------------
      TEMP = FINPRO(N,V1(1),1,VS(1),1)
C-----------------------------------------------------------------------
C
      TEMP = DABS(TEMP - EVALN)
      BIEVER(J) = TEMP
      DO 1110 K = 1,N
 1110 V1(K) = V1(K) - EVALN*VS(K)
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V1(1),1,V1(1),1)
C-----------------------------------------------------------------------
C
      SUM = DSQRT(SUM)
      BIERR(J) = SUM
      BIERRG(J) = SUM/ABS(BIMING(J))
C
      LINT = LFIN - N + 1
      EVAL = (ONE/EVALN - SHIFT)/S0
      GOODA(J) = EVAL
      TEMP = BIEVER(J)
C
      IF(IWRITE.EQ.0)  GO TO 1140
      WRITE(6,1120) J,GOODBI(J)
 1120 FORMAT(/I5,' TH B-INVERSE EIGENVALUE COMPUTED = ',E20.12/)
C
      WRITE(6,1130)  TERR(J),TBETA(J),RNORME
 1130 FORMAT(' NORM OF ERROR IN T-EIGENVECTOR = ',E14.3/
     1' BETA(MA(J)+1)*U(MA(J)) = ',E14.3/
     1' ABS(NORM(RITVEC) - 1.0) = ',E14.3/)
C
 1140 CONTINUE
C
C     RITZVECTORS ARE NORMALIZED AND ERROR ESTIMATES ARE IN BIERR
C     AND BIERRG ARRAYS. STORE EVERYTHING
C
      WRITE(13,1150)
 1150 FORMAT(6X,'BIEIGENVALUE',6X,'RITZNORM',7X,'TBETA',7X,'TLAST',5X,
     1 'BIERROR',6X,'BIEVER')
C
      WRITE(9,1160)
 1160 FORMAT(5X,'BIEIGENVALUE',4X,'MA(J)',4X,'BIMINGAP',5X,'BIERROR',3X
     1 ,'BIERR/GAP',6X,'TERROR')
C
      DO 1190 J=1,NGOODC
```

```
C
      IF(MP(J).EQ.MPMIN) GO TO 1190
C
      WRITE(9,1170) GOODBI(J),MA(J),BIMING(J),BIERR(J),BIERRG(J),TERR(J)
 1170 FORMAT(E20.12,I6,4E12.4)
C
      WRITE(13,1180) EVNEW(J),RNORM(J),TBETA(J),TLAST(J),BIERR(J),
     1 BIEVER(J)
 1180 FORMAT(E20.12,5E12.4)
C
 1190 CONTINUE
C
      WRITE(9,1200)
 1200 FORMAT(/5X, 'J',7X,'AEIGENVALUE',3X,'MA(J)',5X,'AMINGAP')
C
      DO 1210 J = 1,NGOOD
      IF(MP(J).EQ.MPMIN) GO TO 1210
      WRITE(9,1220) J,GOODA(J),MA(J),AMINGP(J)
 1210 CONTINUE
 1220 FORMAT(I6,E20.12,I6,E12.4)
C
      IF (MREJEC.EQ.0) GO TO 1300
C
      WRITE(9,1230)
 1230 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING EIGENVAL
     1LUES'/' EITHER BECAUSE THEY HAD NOT CONVERGED OR BECAUSE THE ERROR
     1 ESTIMATE'/'  WAS NOT AS SMALL AS DESIRED'/)
C
      WRITE(9,1240)
 1240 FORMAT(6X,'GOODBI(J)',3X,'MA(J)',5X,'BIMING(J)',6X,'TBETA(J)',3X,
     1'MP(J)')
C
      WRITE(13,1250)
 1250 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING EIGENVAL
     1LUES'/' EITHER BECAUSE THEY HAD NOT CONVERGED OR BECAUSE'/' THE ER
     1ROR ESTIMATE WAS NOT AS SMALL AS DESIRED'/)
C
      WRITE(13,1260)
 1260 FORMAT(3X,'BIEIGENVALUE',3X,'MA(J)',3X,'M1(J)',3X,'M2(J)',3X,'MP(J
     1)')
C
      DO 1290 J = 1,NGOODC
C
      IF(MP(J).NE.MPMIN) GO TO 1290
C
C     WRITE OUT MESSAGE FOR EACH EIGENVALUE FOR WHICH NO EIGENVECTOR
C     WAS COMPUTED.
C
      WRITE(9,1270) GOODBI(J),MA(J),BIMING(J),TBETA(J),MP(J)
 1270 FORMAT(E15.8,I8,2E14.4,I8)
C
      WRITE(13,1280) GOODBI(J),MA(J),M1(J),M2(J),MP(J)
 1280 FORMAT(E15.8,4I8)
C
 1290 CONTINUE
```

```
C
 1300 CONTINUE
C
      WRITE(9,1310)
 1310 FORMAT(/' ABOVE ARE ERROR ESTIMATES FOR THE BI AND T EIGENVECTORS'
     1/ ' ASSOCIATED WITH THE GOODBI LISTED, DENOTED BY EV '/
     1 ' BIERROR = NORM(BI*X-EV*X),  TERROR = NORM(T*Y - EV*Y)'/
     1 ' WHERE T = T(1,MA(J)),  P*X = RITZVEC = V*Y, T*Y = GOODBI*Y'/
     1 ' BIMINGAP = GAP TO NEAREST BI-EIGENVALUE'/)
C
      WRITE(13,1320)
 1320 FORMAT(/' ABOVE ARE ERROR ESTIMATES FOR THE EIGENVECTORS'/
     1 ' ASSOCIATED WITH THE BI-EIGENVALUES'/
     1 ' RITZNORM = NORM(COMPUTED RITZ VECTOR FOR B-INVERSE)'/
     1 ' TBETA(J) = BETA(MA(J)+1)*Y(MA(J)),  T*Y = BIEVAL*Y'/
     1 ' TLAST(J) = DABS(Y(MA(J)))'/
     1 ' BIERROR = NORM(BI*X - BIEVAL*X) WHERE X = V*Y'/
     1 ' BIEVER = DABS(BIEIGENVALUE - (X-TRANSPOSE*BINVERSE*X))'/)
C
C     NUMBER OF RITZ VECTORS COMPUTED
      NCOMPU = NGOODC - MREJEC
      WRITE(12,1330) N,NCOMPU,NGOODC,MATNO
 1330 FORMAT(3I6,I8,' = SIZE A, NO.RITZVECS, NO.GOODEVALUES,MATNO')
C
      LFIN = 0
      DO 1390 J = 1,NGOODC
      LINT = LFIN + 1
      LFIN = LFIN + N
C
      IF(MP(J).EQ.MPMIN) GO TO 1370
C     RITZ VECTOR WAS COMPUTED
      WRITE(12,1340) J, EVNEW(J), GOODA(J),MP(J)
 1340 FORMAT(I6,4X,2E20.12,I6,' J,GOODBI,GOODA,MP(J)')
C
      WRITE(12,1350) BIERR(J), BIERRG(J), BIMING(J),AMINGP(J)
 1350 FORMAT(4X,' BIRESIDUAL  ',2X,'BIRESIDUAL/GAP',
     12X,'BIMINGAP',3X,' AMINGAP'/
     1 E15.5,E16.5,2E11.3)
C
      WRITE(12,1360) (RITVEC(LL), LL=LINT,LFIN)
 1360 FORMAT(4E20.12)
      GO TO 1390
C     NO RITZ VECTOR WAS COMPUTED FOR THIS EIGENVALUE
 1370 CONTINUE
      WRITE(12,1380) J,GOODBI(J),GOODA(J),MP(J)
 1380 FORMAT(/I5,E20.12,E20.12,I6,' = J,GOODBI,GOODA,MP'/' NO RITZ VECTO
     1R WAS COMPUTED FOR THIS EIGENVALUE'/)
C
 1390 CONTINUE
C
C     DID ANY T-MATRICES INCLUDE OFF-DIAGONAL ENTRIES SMALLER THAN
C     DESIRED, AS SPECIFIED BY BTOL?
C
      IF(IB.GT.0) GO TO 1420
      WRITE(6,1400) KMAXU
```

```
 1400 FORMAT(/' FOR LARGEST T-MATRIX CONSIDERED',I7,' CHECK THE SIZE OF
     1BETAS')
C
C-----------------------------------------------------------------------
C
      CALL TNORM(ALPHA,BETA,BKMIN,TEMP,KMAXU,IBMT)
C
C-----------------------------------------------------------------------
C
      IF(IBMT.LT.0) WRITE(6,1410)
 1410 FORMAT(/' WARNING THE T-MATRICES FOR ONE OR MORE OF THE EIGENVALUE
     1S CONSIDERED'/' HAD AN OFF DIAGONAL ENTRY THAT WAS SMALLER THAN TH
     1E BETA TOLERANCE THAT WAS SPECIFIED'/)
 1420 CONTINUE
C
       GO TO 1670
C
 1430 WRITE(6,1440) NGOOD,NMAX,MDIMRV
 1440 FORMAT(/I4,' RITZ VECTORS WERE REQUESTED BUT THE REQUIRED DIMENSIO
     1N',I6/' IS LARGER THAN USER-SPECIFIED DIMENSION OF RITVEC',I6/
     1' THEREFORE, THE EIGENVECTOR PROCEDURE TERMINATES FOR THE USER TO
     1 INTERVENE'/)
C
      GO TO 1670
C
 1450 WRITE(6,1460) NOLD,N,MATOLD,MATNO,SHIFTO,SHIFT
 1460 FORMAT(/' PARAMETERS READ FROM FILE 3 DO NOT AGREE WITH WHAT USER
     1SPECIFIED'/ ' NOLD,N,MATOLD,MATNO,SHIFTO,SHIFT = '/2I6,2I8,2E10.3
     1/' THEREFORE PROGRAM TERMINATES FOR USER TO RESOLVE THE DIFFERENCE
     1S'/)
C
      GO TO 1670
C
 1470 WRITE(6,1480)
 1480 FORMAT(/' PARAMETERS READ FROM ALPHA,BETA FILE DO NOT AGREE WITH W
     1HAT USER SPECIFIED'/' PROGRAM TERMINATES FOR USER TO RESOLVE THE D
     1IFFERENCES'/)
C
      GO TO 1670
C
 1490 WRITE(6,1500) KMAX,MEV
 1500 FORMAT(/' IN ALPHA, BETA FILE KMAX = ',I6/
     1' BUT EIGENVALUES WERE COMPUTED AT MEV =',I6,' PROGRAM STOPS'/)
C
      GO TO 1670
C
 1510 WRITE(6,1520)
 1520 FORMAT(/' PROGRAM COMPUTED 1ST GUESSES ON T-MATRIX SIZES AND READ
     1THEM TO FILE 10'/' THEN TERMINATED AS REQUESTED.'/)
      GO TO 1670
C
 1530 WRITE(6,1540) MTOL, MDIMTV
 1540 FORMAT(/' PROGRAM TERMINATES BECAUSE THE TVEC DIMENSION ANTICIPATE
     1D',I7/'  IS LARGER THAN THE TVEC DIMENSION',I7,' SPECIFIED BY THE
     1USER.'/'  USER MAY RESET THE TVEC DIMENSION AND RESTART THE PROGRA
```

```
     1M')
      GO TO 1670
C
 1550 WRITE(6,1560)
 1560 FORMAT(/' PROGRAM TERMINATES BECAUSE NO SUITABLE T-EIGENVECTORS WE
     1RE IDENTIFIED'/'  FOR ANY OF THE EIGENVALUES SUPPLIED.  PROBLEM CO
     1ULD BE CAUSED'/'  BY TOO SMALL A TVEC DIMENSION OR SIMPLY THAT SUI
     1TABLE T-VECTORS COULD'/' NOT BE IDENTIFIED.  USER SHOULD EXAMINE O
     1UTPUT'/)
      GO TO 1670
C
 1570 WRITE(6,1580) LVCONT,NTVEC,NGOOD
 1580 FORMAT(/' LVCONT FLAG =',I2,' AND NUMBER ',I5,' OF T-EIGENVECTORS
     1 COMPUTED N.E.'/' NUMBER',I5,' REQUESTED SO PROGRAM TERMINATES'/)
      GO TO 1670
C
 1590 WRITE(6,1600)
 1600 FORMAT(/' PROGRAM TERMINATES WITHOUT COMPUTING RITZ VECTORS'/
     1' BECAUSE ALL T-EIGENVECTORS WERE REJECTED AS NOT SUITABLE FOR THE
     1RITZ VECTOR'/' COMPUTATIONS.  PROBABLE CAUSE IS LACK OF CONVERGENC
     1E OF EIGENVALUES SUPPLIED'/)
       GO TO 1670
C
 1610 WRITE(6,1620)
 1620 FORMAT(/' PROGRAM INDICATES THAT IT IS NOT POSSIBLE TO COMPUTE ANY
     1 OF THE REQUESTED EIGENVECTORS.'/' THEREFORE PROGRAM TERMINATES')
      DO 1630 J=1,NGOODC
 1630 WRITE(6,1640)  J,GOODBI(J),MP(J)
 1640 FORMAT(/4X,' J',11X,'GOODBI(J)',4X,'MP(J)'/I6,E20.12,I9/)
      GO TO 1670
C
 1650 WRITE(6,1660) MBETA,KMAXN
 1660 FORMAT(/' PROGRAM TERMINATES BECAUSE THE STORAGE ALLOTTED FOR THE
     1BETA ARRAY',I8,/' IS NOT SUFFICIENT FOR THE ENLARGED KMAX =',I8,'
     1THAT THE PROGRAM WANTS.'/' USER CAN ENLARGE THE ALPHA,BETA ARRAYS
     1 AND RERUN THE PROGRAM'/)
C
 1670 CONTINUE
C
      STOP
C-----END EIGENVECTOR COMPUTATIONS FOR INVERSES OF REAL SYMMETRIC-------
      END
```

4.4 LIMULT: LANCZS and Sample Matrix-Vector Multiply Subroutines

```
C---LIMULT-(INVERSES OF REAL SYMMETRIC MATRICES)-----------------------LIM00010
C  Authors:  Jane Cullum and Ralph A. Willoughby (deceased)            LIM00020
C            Los Alamos National Laboratory                            LIM00030
C            Los Alamos, New Mexico 87544                              LIM00040
C                                                                      LIM00050
C            E-mail:  cullumj@lanl.gov                                 LIM00060
C                                                                      LIM00070
C  These codes are copyrighted by the authors.  These codes            LIM00080
C  and modifications of them or portions of them are NOT to be         LIM00090
C  incorporated into any commercial codes or used for any other        LIM00100
C  commercial purposes such as consulting for other companies,         LIM00110
C  without legal agreements with the authors of these Codes.           LIM00120
C  If these Codes or portions of them are used in other scientific or  LIM00130
C  engineering research works the names of the authors of these codes  LIM00140
C  and appropriate references to their written work are to be          LIM00150
C  incorporated in the derivative works.                               LIM00160
C                                                                      LIM00170
C  This header is not to be removed from these codes.                  LIM00180
C                                                                      LIM00190
C         REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4           LIM00191
C         Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsLIM00192
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in  LIM00193
C         Applied Mathematics, 2002. SIAM Publications,                LIM00194
C         Philadelphia, PA. USA                                        LIM00195
C                                                                      LIM00196
C                                                                      LIM00200
C     CONTAINS SUBROUTINE LANCZS AND SAMPLE USPEC AND BSOLV            LIM00210
C     USED BY THE VERSION OF THE LANCZOS ALGORITHMS FOR                LIM00220
C     FACTORED INVERSES OF REAL SYMMETRIC MATRICES, LIVAL AND LIVEC.   LIM00230
C                                                                      LIM00240
C     NONPORTABLE CONSTRUCTIONS:                                       LIM00250
C     1.  THE ENTRY MECHANISM USED TO PASS THE STORAGE LOCATIONS       LIM00260
C         OF THE FACTORIZATION OF THE MATRIX TO BE USED BY             LIM00270
C         LANCZS TO THE SOLVE SUBROUTINE BSOLV.                        LIM00280
C     2.  IN THE SAMPLE USPEC SUBROUTINES PROVIDED:                    LIM00290
C         THE FREE FORMAT (7,*) AND FORMATS (20A4) AND  (4Z20)         LIM00300
C         USED IN DEFINING THE MATRICES.                               LIM00310
C                                                                      LIM00320
C-----LANCZS-COMPUTE LANCZOS TRIDIAGONAL MATRICES-----------------------LIM00330
C                                                                      LIM00340
      SUBROUTINE LANCZS(MATVEC,ALPHA,BETA,V1,V2,VS,G,KMAX,MOLD1,N,IIX)  LIM00350
C                                                                      LIM00360
C----------------------------------------------------------------------LIM00370
      DOUBLE PRECISION  ALPHA(1), BETA(1), V1(1), V2(1), VS(1)         LIM00380
      DOUBLE PRECISION SUM, ONE, ZERO, TEMP                            LIM00390
      REAL  G(1)                                                       LIM00400
      EXTERNAL MATVEC                                                  LIM00410
      DOUBLE PRECISION  FINPRO, DSQRT                                  LIM00420
C----------------------------------------------------------------------LIM00430
C     ALPHA, BETA, LANCZOS VECTOR GENERATION                           LIM00440
C     ALPHA BETA GENERATION STARTS WITH IVEC = 1, BETA(1) = ZERO       LIM00450
```

```
C     V2 =  RANDOM UNIT VECTOR AND V1 = ZERO, OR EXTENDS
C     AN EXISTING ALPHA/BETA FILE.
C
      ZERO = 0.0D0
      ONE  = 1.0D0
      IF (MOLD1.GT.1) GO TO 30
      BETA(1) = ZERO
      IIL = IIX
C
C-----------------------------------------------------------------------
      CALL GENRAN(IIL,G,N)
C-----------------------------------------------------------------------
C
      DO 10 K = 1,N
   10 V2(K) = G(K)
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V2(1),1,V2(1),1)
C-----------------------------------------------------------------------
C
      SUM = ONE/DSQRT(SUM)
C
      DO 20 K = 1,N
      V1(K) = ZERO
   20 V2(K) = SUM*V2(K)
C
   30 CONTINUE
C
      DO 80 IVEC = MOLD1,KMAX
C
      DO 40 K = 1,N
   40 VS(K) = V2(K)
C
C-----------------------------------------------------------------------
      JBSOLV = 2
      CALL MATVEC(VS,VS,JBSOLV)
C-----------------------------------------------------------------------
C
C     VS = B(INVERSE)*V2
C
      SUM = BETA(IVEC)
C
      DO 50 K = 1,N
   50 V1(K) = VS(K)-SUM*V1(K)
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V1(1),1,V2(1),1)
C-----------------------------------------------------------------------
C
      ALPHA(IVEC) = SUM
C
      DO 60 K = 1,N
   60 V1(K) = V1(K)-SUM*V2(K)
C
C-----------------------------------------------------------------------
```

```
      SUM = FINPRO(N,V1(1),1,V1(1),1)
C-----------------------------------------------------------------------
C
      IN = IVEC+1
C
      BETA(IN) = DSQRT(SUM)
      SUM = ONE/BETA(IN)
C
      DO 70 K = 1,N
      TEMP = SUM*V1(K)
      V1(K) = V2(K)
   70 V2(K) = TEMP
C
   80 CONTINUE
C
      RETURN
C-----END LANCZS--------------------------------------------------------
      END
C
C-----USPEC FOR FACTORED INVERSES OF REAL SYMMETRIC MATRICES------------
C
      SUBROUTINE CUSPEC(N,MATNO)
C     SUBROUTINE USPEC(N,MATNO)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  BD(2200),BSD(10000)
      REAL EXPLAN(20)
      INTEGER  KCOL(2200),KROW(10000),IPR(2200),IPT(2200)
C-----------------------------------------------------------------------
C     NOTE THAT THIS SUBROUTINE ASSUMES THAT B IS POSITIVE DEFINITE.
C     USER COULD REPLACE THIS SUBROUTINE AND CORRESPONDING SAMPLE
C     USPEC SUBROUTINE BY ONE THAT WORKS WITH GENERAL FACTORIZATION.
C
C     DIMENSIONS ARRAYS NEEDED TO DEFINE CHOLESKY FACTOR OF B-MATRIX,
C     READS CHOLESKY FACTOR FROM FILE 7, AND THEN PASSES STORAGE
C     LOCATIONS OF THESE ARRAYS TO THE B-MATRIX SOLVE SUBROUTINE BSOLV.
C
C     HERE WE HAVE B = P*C*P' = L*L'  WHERE  C = SO*A + SHIFT*I.
C     P IS A PERMUTATION MATRIX DEFINED BY THE VECTOR MAPS IPR AND IPT.
C     THE ITH ROW OF B CORRESPONDS TO THE JTH ROW OF C (A) WHERE
C     J = IPR(I) AND I = IPT(J).   A IS THE ORIGINAL MATRIX.
C
C     THE B-CHOLESKY FACTOR IS STORED IN THE FOLLOWING SPARSE FORMAT:
C     N = ORDER OF THE B-MATRIX.
C     NZT = NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN THE CHOLESKY
C           FACTOR, L.
C     KCOL(J), J=1,N IS THE NUMBER OF NONZERO SUBDIAGONAL ELEMENTS IN
C              COLUMN J OF L.
C     KROW(K), K=1,NZT IS THE ROW INDEX FOR CORRESPONDING ENTRY BSD(K).
C     BD(J), J = 1,N CONTAINS THE DIAGONAL ENTRIES OF L.
C     BSD(K), K =1,NZT CONTAINS THE NONZERO SUBDIAGONAL ENTRIES OF L
C             BY COLUMN.
C     JPERM = (0,1):  1 MEANS CHOLEKSY FACTOR CORRESPONDS TO
C                     PERMUTED C.  0 MEANS NO PERMUTATION WAS USED.
C-----------------------------------------------------------------------
```

```
C     READ CHOLESKY FACTOR FROM FILE 7.  MUST BE STORED
C     IN SPARSE MATRIX FORMAT.
C
      READ(7,5) EXPLAN
    5 FORMAT(20A4)
C
      READ(7,10) NZT,NOLD,NZL,MATOLD,JPERM
   10 FORMAT(I10,2I6,I8,I6)
C
      WRITE(6,20) NZT,NZL,N,NOLD,MATOLD,JPERM
   20 FORMAT(' HEADER, CHOLESKY FACTOR FILE'/
     1 3X,'NZT',3X,'NZL',5X,'N',2X,'NOLD',2X,'MATOLD',1X,'JPERM'/
     1 4I6,I8,I6/)
C
      IF (N.NE.NOLD.OR.MATNO.NE.MATOLD) GO TO 70
C
      READ(7,5) EXPLAN
C
      READ(7,30) (KCOL(K), K = 1,NZL)
   30 FORMAT(13I6)
C
      READ(7,5) EXPLAN
C
      READ(7,30) (KROW(K), K = 1,NZT)
C
      READ(7,5) EXPLAN
C
      READ(7,40) (BD(K), K = 1,N)
   40 FORMAT(4Z20)
C
      READ(7,5) EXPLAN
C
      READ(7,40) (BSD(K), K = 1,NZT)
C
C     DOES CHOLESKY FACTOR CORRESPOND TO PERMUTED B?
      IF(JPERM.EQ.0)  GO TO 60
C
      READ(7,5) EXPLAN
C
      READ(7,30) (IPR(K), K = 1,N)
C
      DO 50 K = 1,N
      J = IPR(K)
   50 IPT(J) = K
C
C-----------------------------------------------------------------------
      CALL LPERME(IPR,IPT,N)
C-----------------------------------------------------------------------
C
   60 CONTINUE
C
C-----------------------------------------------------------------------
C     PASS STORAGE LOCATIONS OF FACTORS TO INVERSION SUBROUTINE BSOLV
      CALL BSOLVE(BSD,BD,KCOL,KROW,N,NZT,NZL)
C-----------------------------------------------------------------------
```

```
C
      GO TO 90
C
   70 CONTINUE
C     DEFAULT EXIT
      WRITE(6,80)
   80 FORMAT(' TERMINATE.  PARAMETERS IN CHOLESKY FACTOR FILE'/
     1' DO NOT AGREE WITH THOSE SPECIFIED BY THE USER'/)
      STOP
C
   90 CONTINUE
C-----END OF USPEC------------------------------------------------------
      RETURN
      END
C
C-----BSOLV-(FACTORED INVERSE OR L*L-TRANS MULTIPLY)--------------------
C           (FOR POSITIVE DEFINITE SYMMETRIC SPARSE MATRICES)
C
C     SUBROUTINE BSOLV(V,U,JBSOLV)
      SUBROUTINE CBSOLV(V,U,JBSOLV)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  BD(1),BSD(1),U(1),V(1),TEMP,ZERO,ONE
      INTEGER  KCOL(1),KROW(1)
C-----------------------------------------------------------------------
      GO TO 3
      ENTRY BSOLVE(BSD,BD,KCOL,KROW,N,NZT,NZL)
      GO TO 4
C-----------------------------------------------------------------------
C     JBSOLV = 2 MEANS SOLVE B*U = V
C     JBSOLV = 1 MEANS COMPUTE U = B*V:  NOTE THAT IN THIS CASE V IS
C     DESTROYED.  LANCZOS PROGRAMS AS WRITTEN DO NOT USE JBSOLV = 1
C     PATH.
    3 ZERO = 0.0D0
      ONE  = 1.0D0
      IF (JBSOLV .EQ.2) GO TO 60
C     U = B*V WHERE  B = L*L'
      KL = 0
      DO 20 J = 1,N
      TEMP = V(J)*BD(J)
      IF (KCOL(J).EQ.0.OR.J.EQ.N) GO TO 20
      KF = KL + 1
      KL  = KL + KCOL(J)
      DO 10 K = KF,KL
      IK = KROW(K)
   10 TEMP = BSD(K)*V(IK) + TEMP
   20 V(J) = TEMP
C     V = L'*V
      DO 30 K = 1,N
   30 U(K) = V(K)*BD(K)
      KL = 0
      DO 50 K = 1,N
      TEMP = V(K)
      IF (KCOL(K).EQ.0.OR.K.EQ.N) GO TO 50
      KF = KL + 1
```

```
      KL = KL + KCOL(K)
      DO 40 KK = KF,KL
      KR = KROW(KK)
   40 U(KR) = U(KR) + TEMP*BSD(KK)
   50 CONTINUE
      GO TO 120
C     U = B*V
C-----------------------------------------------------------------------
   60 CONTINUE
C     SOLVE B*U = V FOR U WHERE  B = L*L'
C     SET U = V. FIRST SOLVE L*U = U FOR U, THEN SOLVE L'*U = U FOR U
      KL = 0
      DO 70 K = 1,N
   70 U(K) = V(K)
      DO 90 K = 1,N
      TEMP = U(K)/BD(K)
      U(K) = TEMP
      IF (KCOL(K).EQ.0.OR.K.EQ.N) GO TO 90
      KF = KL + 1
      KL = KL + KCOL(K)
      DO 80 KK = KF,KL
      KR = KROW(KK)
   80 U(KR) = U(KR) - TEMP*BSD(KK)
   90 CONTINUE
      NP1 = N+1
      KF = NZT + 1
      DO 110 K = 1,N
      L = NP1 - K
      TEMP = U(L)
      IF (KCOL(L).EQ.0.OR.L.EQ.N) GO TO 110
      KL = KF - 1
      KF = KF - KCOL(L)
      DO 100 LL = KF,KL
      LR = KROW(LL)
  100 TEMP = TEMP - BSD(LL)*U(LR)
  110 U(L) = TEMP/BD(L)
  120 CONTINUE
C
    4 RETURN
C
C-----END OF BSOLV-------------------------------------------------------
      END
C
C-----SUBROUTINES FOR DIAGONAL TEST MATRICES----------------------------
C
C     BSOLV AND USPEC SUBROUTINES FOR DIAGONAL TEST MATRICES
C
C-----BSOLV DIAGONAL TEST MATRIX----------------------------------------
C
C     SUBROUTINE DBSOLV(V,U,JBSOLV)
      SUBROUTINE BSOLV(V,U,JBSOLV)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  V(1),U(1),D(1)
C-----------------------------------------------------------------------
```

```
      GO TO 3
C     BELOW ENTRY IS FOR A DIAGONAL TEST MATRIX
      ENTRY DSOLVE(D,N)
      GO TO 4
C-----------------------------------------------------------------------
C     JBSOLV = 1, COMPUTE U = D*V. (NOTE THIS IS NOT USED)
C     JBSOLV = 2, COMPUTE U = (D-INVERSE)*V
    3 IF(JBSOLV.EQ.2) GO TO 20
      DO 10 I=1,N
   10 U(I) = D(I)*V(I)
      GO TO 40
C
   20 DO 30 I=1,N
   30 U(I)= V(I)/D(I)
C
   40 CONTINUE
    4 RETURN
C
C-----END OF BSOLV FOR DIAGONAL TEST MATRIX ----------------------------
      END
C
C-----START OF USPEC FOR DIAGONAL TEST MATRIX---------------------------
C
      SUBROUTINE USPEC(N,MATNO)
C     SUBROUTINE DUSPEC(N,MATNO)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  D(1000), DI(1000), SHIFT, SPACE
      DOUBLE PRECISION  DABS, DFLOAT
      REAL  EXPLAN(20)
C-----------------------------------------------------------------------
C
      READ(7,10) EXPLAN
   10 FORMAT(20A4)
      READ(7,*) NOLD,NUNIF,SPACE,D(1),SHIFT
      NNUNIF = NOLD - NUNIF
      WRITE(6,20) NOLD,SPACE,NNUNIF,D(1),SHIFT
   20 FORMAT(/' DIAGONAL TEST MATRIX, SIZE = ',I4/' IS THE INVERSE OF MA
     1TRIX WITH MOST ENTRIES',E10.3/' UNITS APART AND WITH ',I3,' ENTRIE
     1S IRREGULARLY SPACED'/' FIRST ENTRY WAS ',E13.4,' SHIFT = ',E10.3
     1/)
C
      IF(N.NE.NOLD) GO TO 100
C     COMPUTE THE UNIFORM PORTION OF THE SPECTRUM
      DO 30 J=2,NUNIF
   30 D(J) = D(1) - DFLOAT(J-1)*SPACE
      NUNIF1=NUNIF + 1
      READ(7,10)  EXPLAN
      DO 40 J=NUNIF1,N
   40 READ(7,*) D(J)
      NB = NUNIF - 2
C
      IF(SHIFT.EQ.0.) GO TO 60
      DO 50 J=1,N
   50 D(J) = D(J) + SHIFT
```

```
C                                                                       LIM03760
C     COMPUTE EIGENVALUES OF INVERSE FOR PRINTOUT ONLY                  LIM03770
   60 DO 70 J = 1,N                                                     LIM03780
   70 DI(J) = 1.D0/D(J)                                                 LIM03790
      WRITE(6,80) (DI(I), I=1,10 )                                      LIM03800
      WRITE(6,90) (DI(I), I = NB,N)                                     LIM03810
   80 FORMAT(/' INVERSE LANCZOS TEST, LANCZS USES INVERSE OF GIVEN MATRILIM03820
     1X'/' 1ST 10 ENTRIES OF INVERSE OF DIAGONAL TEST MATRIX = '/(3E22.1LIM03830
     14))                                                               LIM03840
   90 FORMAT(/' MIDDLE (ORIGINALLY UNIFORM) PORTION OF MATRIX IS NOT PRILIM03850
     1NTED OUT'/' END OF (UNIFORM) PLUS NONUNIFORM SECTION = '/(3E25.16)LIM03860
     1)                                                                 LIM03870
C                                                                       LIM03880
C     DIAGONAL GENERATION COMPLETE                                      LIM03890
C                                                                       LIM03900
C-----------------------------------------------------------------------LIM03910
C    PASS STORAGE LOCATIONS OF D AND N TO DSOLV SUBROUTINE              LIM03920
       CALL DSOLVE(D,N)                                                 LIM03930
C-----------------------------------------------------------------------LIM03940
C                                                                       LIM03950
      RETURN                                                            LIM03960
  100 WRITE(6,110) NOLD,N                                               LIM03970
  110 FORMAT(' PROGRAM TERMINATES BECAUSE NOLD = ',I5,'DOES NOT EQUAL N LIM03980
     1 =',I5)                                                           LIM03990
C-----END OF USPEC SUBROUTINE FOR DIAGONAL TEST MATRICES----------------LIM04000
      STOP                                                              LIM04010
      END                                                               LIM04020
```

4.5 PERMUT: LORDER: LFACT: LTEST: Optional Routines for Chapters 4, 5, 9

```
C-----PERMUT-(USES SPARSPAK PACKAGE)-----------------------------------
C  AUTHORS:  RALPH A. WILLOUGHBY (DECEASED)
C
C
C
C
C          E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C
C     OPTIONAL PREPROCESSING PROGRAM FOR USE WITH LANCZOS CODES.
C     GIVEN A REAL SYMMETRIC A-MATRIX IN SPARSE MATRIX FORMAT, PERMUT
C     CALLS THE SPARSPAK PACKAGE (A. GEORGE, J. LIU, E. NG, U. WATERLOO)
C     TO DETERMINE A REORDERING OF A, THAT IS A PERMUTATION MATRIX
C     P, SUCH THAT SPARSITY IS PRESERVED IN THE FACTORIZATION OF
C     THE PERMUTED MATRIX.  PERMUT ALSO MODIFIES THE GIVEN A-MATRIX
C     TO FORM THE MATRIX C = S0*A + SHIFT*I, WHERE S0 AND SHIFT
C     ARE SCALARS PROVIDED BY THE USER, AND THEN WRITES THIS
C     C-MATRIX OUT TO FILE 9 ALONG WITH THE PERMUTATION P WHICH
C     IS DEFINED BY THE VECTOR IPR.  IPR IS ALSO WRITTEN SEPARATELY
C     TO FILE 14.
C
C     NONPORTABLE CONSTRUCTIONS:
C     1.  INTEGER*2 VARIABLE NPERM.  NOTE THAT THIS VARIABLE CANNOT
C         BE CHANGED TO INTEGER*4.
C     2.  FREE FORMAT (5,*) AND THE FORMAT (20A4).
C     3.  TO AVOID COMPOUNDING FORMAT CONVERSION ERRORS, THE MATRIX
C         ENTRIES SHOULD BE STORED IN MACHINE FORMAT, ((4Z20) FOR
C         IBM/3081)
C
C-----------------------------------------------------------------------
C     SYMMETRIC A-MATRIX IS READ FROM FILE 8.  MATRIX IS STORED
C     IN FOLLOWING SPARSE FORMAT:
C
C     NZL = INDEX OF LAST COLUMN CONTAINING NONZEROS BELOW THE DIAGONAL.
C     NZS = NUMBER OF NONZERO SUBDIAGONAL ENTRIES
C     ICOL(K), K=1,NZL CONTAINS THE NUMBER OF NONZERO SUBDIAGONAL
C              ENTRIES IN COLUMN K.
C     IROW(K), K=1,NZS CONTAINS ROW INDEX OF KTH NONZERO SUBDIAGONAL
C              ENTRY, ENTRIES STORED COLUMN BY COLUMN.
```

```
C     AD(K), K=1,N CONTAINS THE DIAGONAL ENTRIES OF A, INCLUDING ANY    PER00520
C            ZERO ENTRIES.                                              PER00530
C     ASD(K), K=1,NZS CONTAINS THE NONZERO SUBDIAGONAL ENTRIES OF A,    PER00540
C             COLUMN BY COLUMN.                                         PER00550
C                                                                       PER00560
C-----INPUT/OUTPUT FILES -----------------------------------------------PER00570
C                                                                       PER00580
C    INPUT FILES:                                                       PER00590
C    FILE 5      CONTAINS THE PROGRAM PARAMETERS SET BY USER            PER00600
C    FILE 8      CONTAINS THE SPARSE A-MATRIX                           PER00610
C                                                                       PER00620
C    OUTPUT FILES:                                                      PER00630
C    FILE 6      INTERACTIVE TERMINAL FILE                              PER00640
C    FILE 9      CONTAINS THE SPARSE DATA FOR C = S0*A + SHIFT*I.       PER00650
C    FILE 14     CONTAINS PERMUTATION IPR DEFINING THE REORDERING.      PER00660
C                IN PARTICULAR J = IPR(I) MEANS ROW(COL) I OF           PER00670
C                B = P*C*(P-TRANSPOSE) CORRESPONDS TO ROW(COL) J        PER00680
C                OF THE A-MATRIX.                                       PER00690
C                                                                       PER00700
C-----SPARSPAK----------------------------------------------------------PER00710
C     ARRAYS AND PARAMETERS THAT ARE REQUIRED BY SPARSPAK.              PER00720
C     NOTE THAT THE CALL FOR SPARSPAK IS SPRSPK.  SUBROUTINES           PER00730
C     IJBEGN, INIJ, IJEND, ORDRB5, AND PSTATS ARE SPARSPAK              PER00740
C     SUBROUTINES.                                                      PER00750
C                                                                       PER00760
C     S = VECTOR WHOSE ACTUAL DIMENSION IS DETERMINED BY SPARSPAK       PER00770
C         WHEN THE REORDERING IS OBTAINED.  USER SPECIFIES MAXIMUM      PER00780
C         DIMENSION MAXS ALLOWED; SPARSPAK DEFAULTS IF THIS MAXIMUM     PER00790
C         IS EXCEEDED.  SPARSPAK IS DESIGNED FOR SOLVING SYSTEMS        PER00800
C         OF EQUATIONS, THUS THE VECTOR S IS DESIGNED TO CONTAIN        PER00810
C         THE SOLUTION VECTOR IF THERE IS ONE, FOLLOWED BY THE          PER00820
C         PERMUTATION VECTOR IPR, FOLLOWED BY OTHER INFORMATION         PER00830
C         GENERATED BY SPARSPAK.  A CORRECT SIZE FOR MAXS CAN BE        PER00840
C         DETERMINED ONLY AFTER THE FACT.  AS A FIRST GUESS ONE         PER00850
C         CAN SET MAXS = K*N WHERE K >= 10.                             PER00860
C                                                                       PER00870
C     MSGLVL = CONTROL FOR WRITES TO FILE 6                             PER00880
C     NEQNS = ORDER OF A, THIS IS COMPUTED BY SPARSPAK                  PER00890
C     IERR = CONTROLS WRITING OF ERROR MESSAGES BY SPARSPAK.            PER00900
C     MAXS = USER-SPECIFIED MAXIMUM ALLOWED DIMENSION OF S-ARRAY.       PER00910
C                                                                       PER00920
C                                                                       PER00930
C----------------------------------------------------------------------PER00940
      DOUBLE PRECISION AD(3000),ASD(10000),S0,SHIFT                     PER00950
      DOUBLE PRECISION S(30000),STEMP                                   PER00960
      REAL EXPLAN(20)                                                   PER00970
      INTEGER ICOL(3000),IROW(10000),IPR(3000)                          PER00980
      INTEGER*2 NPERM(4)                                                PER00990
      COMMON /SPKUSR/ MSGLVL,IERR,MAXS,NEQNS                            PER01000
      EQUIVALENCE (STEMP,NPERM(1))                                      PER01010
C---------------------------------------------------------------------- PER01020
C                                                                       PER01030
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:                            PER01040
C     1.  AD:  >= N, THE ORDER OF A-MATRIX.                             PER01050
C     2.  ASD: >= NZS, THE NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN A.  PER01060
```

```
C     4.  ICOL:   >= N
C     5.  IROW:   >= NZS
C     6.  IPR:    >= N + 4
C     7.  S:   >= MAXS
C
C-----------------------------------------------------------------------
C
      WRITE(6,10)
   10 FORMAT(/' CALL SPARSPAK TO FIND REORDERING OF THE GIVEN MATRIX'/
     1' THAT PRESERVES SPARSITY IN THE FACTORIZATION'/)
C
C     READ IN USER-SPECIFIED PARAMETERS
C     SPECIFY THE MAXIMUM DIMENSION (MAXS) ALLOWED FOR ARRAY S,
C     AND WHETHER OR NOT A-MATRIX IS BEING SCALED BY S0 OR SHIFTED
C     ISCALE = 0, THEN NO SCALING OR SHIFTING
      READ (5,20) EXPLAN
      WRITE (6,20) EXPLAN
      READ (5,20) EXPLAN
   20 FORMAT(20A4)
      READ (5,*) MAXS,ISCALE,S0,SHIFT
C
C     READ IN INDICES FOR A-MATRIX STORED IN SPARSE FORMAT ON FILE 8.
C     ONLY THE NONZERO STRUCTURE IS NEEDED TO OBTAIN THE PERMUTATION.
C
      READ(8,30) NZS,N,NZL,MATNO
   30 FORMAT(I10,2I6,I8)
C
      WRITE(6,40) NZS,N,NZL,MATNO,MAXS,ISCALE,SHIFT,S0
   40 FORMAT(/I10,2I6,I10,I10,' = NZS,N,NZL,MATNO,MAXS'/
     1 I6,2E12.5,' = ISCALE,SHIFT,S0'/)
C
      READ(8,50) (ICOL(K), K=1,NZL)
      READ(8,50) (IROW(K), K=1,NZS)
   50 FORMAT(13I6)
C
C     DIAGONAL IS READ (INCLUDING ANY ZERO ENTRIES), THEN NONZERO
C     BELOW DIAGONAL ENTRIES ARE READ IN
      READ(8,60) (AD(K), K=1,N)
      READ(8,60) (ASD(K), K=1,NZS)
   60 FORMAT(4E19.10)
C
      IF (ISCALE.EQ.0) GO TO 90
C
C     CALCULATE C = S0*A + SHIFT*I AND PUT IN A-ARRAY
      DO 70 K = 1,N
   70 AD(K) = S0*AD(K) + SHIFT
      DO 80 K = 1,NZS
   80 ASD(K) = S0*ASD(K)
   90 CONTINUE
C
C-----------------------------------------------------------------------
C     INPUT THE SPARSENESS STRUCTURE OF GIVEN A-MATRIX TO SPARSPAK
      CALL SPRSPK
C-----------------------------------------------------------------------
C
```

```
      MSGLVL = 4
C
C-----------------------------------------------------------------------
      CALL IJBEGN
C-----------------------------------------------------------------------
C
      LLAST = 0
      DO 110 J = 1,NZL
      IF (ICOL(J).EQ.0) GO TO 110
      JJ = J
      LFIRST = LLAST + 1
      LLAST = LLAST + ICOL(J)
      DO 100 L = LFIRST,LLAST
      II = IROW(L)
C
C-----------------------------------------------------------------------
      CALL INIJ(II,JJ,S)
C-----------------------------------------------------------------------
C
  100 CONTINUE
C
  110 CONTINUE
C
C     SPARSENESS STRUCTURE HAS BEEN INPUTED TO SPARSPAK.
C
C-----------------------------------------------------------------------
      CALL IJEND(S)
C-----------------------------------------------------------------------
C
      WRITE(6,120) N,NEQNS
  120 FORMAT(/2I6,' = N,NEQNS'/)
      IF (N.NE.NEQNS) GO TO 230
C
C-----------------------------------------------------------------------
C     USE SPARSPAK TO GENERATE REORDERING OF A THAT PRESERVES
C     SPARSITY.  CORRESPONDING FACTORIZATION CAN BE COMPUTED BY
C     PREPROCESSING PROGRAM LFACT WHEN C = S0*A + SHIFT*I IS POSITIVE
C     DEFINITE. BELOW CALLS THE MINIMUM DEGREE ALGORITHM PROVIDED
C     IN SPARSPAK.
      CALL ORDRB5(S)
      CALL PSTATS
C-----------------------------------------------------------------------
C
C     EXTRACT THE REORDERING FROM SPARSPAK S VECTOR AND STORE IN FILE 14
      L = 1
      KNUM = N
      DO 130 K = 1,N
      KNUM = KNUM + 1
      STEMP = S(KNUM)
      IPR(L) = NPERM(1)
      IPR(L+1) = NPERM(2)
      IPR(L+2) = NPERM(3)
      IPR(L+3) = NPERM(4)
      L = L+4
      IF (L.GT.N) GO TO 140
```

```
  130 CONTINUE                                                          PER02170
  140 CONTINUE                                                          PER02180
C                                                                       PER02190
      WRITE(14,150) N,MATNO                                             PER02200
  150 FORMAT(I6,I8,' = N MATNO  K IPR(K)   A-MATRIX PERMUTATION')       PER02210
      WRITE(14,160) (K,IPR(K), K = 1,N)                                 PER02220
  160 FORMAT(6(1X,2I6))                                                 PER02230
C                                                                       PER02240
C                                                                       PER02250
C     WRITE C = S0*A + SHIFT*I WITH THE PERMUTATION IPR TO FILE 9.      PER02260
C                                                                       PER02270
      JPERM = 1                                                         PER02280
      WRITE(9,170) NZS,N,NZL,MATNO,JPERM                                PER02290
  170 FORMAT(I10,2I6,I8,I6,' = NZS,N,NZL,MATNO,JPERM. ACOMPAC')         PER02300
C                                                                       PER02310
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS WRITTEN   PER02320
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS WRITTEN   PER02330
      WRITE(9,180) (ICOL(K), K=1,NZL)                                   PER02340
      WRITE(9,180) (IROW(K), K=1,NZS)                                   PER02350
  180 FORMAT(13I6)                                                      PER02360
C     DIAGONAL IS WRITTEN FIRST, THEN NONZERO BELOW DIAGONAL ENTRIES    PER02370
      WRITE(9,190) (AD(K), K=1,N)                                       PER02380
      WRITE(9,190) (ASD(K), K=1,NZS)                                    PER02390
  190 FORMAT(4E19.10)                                                   PER02400
      WRITE(9,180) (IPR(K), K=1,N)                                      PER02410
C                                                                       PER02420
      IF(ISCALE.NE.0) GO TO 200                                         PER02430
C     ISCALE = 0, SET DEFAULT VALUES OF S0 AND SHIFT                    PER02440
      S0 = 1.D0                                                         PER02450
      SHIFT = 0.D0                                                      PER02460
  200  WRITE(9,210) S0,SHIFT                                            PER02470
  210 FORMAT(2E12.5,' = S0 SHIFT'/                                      PER02480
     1 ' ABOVE IS SPARSE DATA FOLLOWED BY PERMUTATION IPR'/             PER02490
     1 ' FOR THE MATRIX C = S0*A+SHIFT*I '/                             PER02500
     1 ' B = P*C*PTRANS CAN BE GENERATED IN SUBROUTINE LORDER'/         PER02510
     1 ' ROW(COL) I OF B CORRESPONDS TO ROW(COL) J OF C, J = IPR(I)'/   PER02520
     1 ' NZS = TOTAL NUMBER OF SUBDIAGONAL NONZEROS IN C'/              PER02530
     1 ' KCOL(K) = NUMBER OF SUBDIAGONAL NONZEROS IN COL K OF C'/       PER02540
     1 ' KROW(K) = ROW INDEX OF SUBDIAGONAL NONZERO'/                   PER02550
     1 ' SUBDIAGONAL NONZEROS IN C ARE STORED COLUMN BY COLUMN'/        PER02560
     1 ' AD(K) = THE KTH DIAGONAL ELEMENT OF C'/                        PER02570
     1 ' ASD(K) =  KTH SUBDIAGONAL NONZERO IN C'/)                      PER02580
C                                                                       PER02590
      WRITE(6,220)                                                      PER02600
  220 FORMAT(/' PERMUT IS FINISHED MATRIX IS ON FILE 9'/)               PER02610
C                                                                       PER02620
  230 CONTINUE                                                          PER02630
C                                                                       PER02640
C-----END PERMUT--------------------------------------------------------PER02650
      STOP                                                              PER02660
      END                                                               PER02670
```

```
C-----LORDER-(STAND ALONE PROGRAM)-------------------------------------
C  AUTHORS:  RALPH A. WILLOUGHBY (DECEASED)
C
C
C
C           E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C
C     ACCORDING TO THE PFORT VERIFIER THIS PROGRAM IS PORTABLE.
C     HOWEVER TO AVOID COMPOUNDING FORMAT CONVERSION ERRORS,
C     MATRIX ENTRIES SHOULD BE STORED IN MACHINE FORMAT, ((4Z20)
C     FOR IBM/3081).
C
C     LORDER TAKES A SPARSE MATRIX C AND A PERMUTATION P GIVEN BY
C     THE VECTOR IPR AND COMPUTES THE PERMUTED MATRIX B = P*C*P' ,
C     AND THEN WRITES B TO FILE 9 ALONG WITH IPR AND ANY SCALE S0
C     AND SHIFT THAT WERE USED TO OBTAIN THE INPUT MATRIX C. (HERE
C     ROW(COL) I OF B CORRESPONDS TO ROW(COL) J OF A WHERE J = IPR(I),
C     AND INPUT MATRIX C = S0*A + SHIFT*I.
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION ASD(10000),AD(3000),BSD(10000),BD(3000)
      DOUBLE PRECISION SHIFT,S0
      INTEGER  IPR(3000),IPT(3000)
      INTEGER  IROW(10000),INUM(10000),ICOL(3000)
      INTEGER  KROW(10000),KNUM(10000),KCOL(3000)
C-----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     1.  AD, BD:  >= N, THE ORDER OF C-MATRIX.
C     2.  ASD: >= NZS, THE NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN C.
C     3.  BSD:   >= NZS, THE NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN
C                  B = P*C*P-TRANSPOSE
C     4.  IPR, IPT:   >= N
C     5.  ICOL, KCOL:   >= N
C     6.  IROW, KROW, INUM, KNUM:   >=  NZ = 2*NZS + N
C
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LORDER PROGRAM, COMPUTE B = P*C*(P-TRANSPOSE), STORE ON
     1FILE 9'/)
C
```

```
C     READ NUMBER OF NONZERO SUBDIAGONAL ENTRIES (NZS), ORDER OF MATRIX
C     (N),INDEX OF LAST COLUMN CONTAINING NONZERO ENTRIES BELOW THE
C     DIAGONAL (NZL), MATRIX IDENTIFICATION NUMBER (MATNO), PERMUTATION
C     FLAG (JPERM).
      READ(8,20) NZS,N,NZL,MATNO,JPERM
   20 FORMAT(I10,2I6,I8,I6)
C
      WRITE(6,30) NZS,N,NZL,MATNO,JPERM
   30 FORMAT(/I10,2I6,I8,I3,' = NZS,N,NZL,MATNO,JPERM'/)
C
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS READ
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS READ
      READ(8,40) (ICOL(K), K=1,NZL)
      READ(8,40) (IROW(K), K=1,NZS)
   40 FORMAT(13I6)
C
      NZL1 = NZL + 1
      DO 50 K = NZL1,N
   50 ICOL(K) = 0
C
C     DIAGONAL OF C-MATRIX IS READ (INCLUDING ANY ZERO ENTRIES), THEN
C     NONZERO SUBDIAGONAL ENTRIES ARE READ IN
      READ(8,60) (AD(K), K=1,N)
      READ(8,60) (ASD(K), K=1,NZS)
   60 FORMAT(4E19.10)
C
      IF(JPERM.EQ.0) GO TO 390
C     READ PERMUTATION
      READ(8,40) (IPR(K), K = 1,N)
C
      DO 70 K = 1,N
      J = IPR(K)
   70 IPT(J) = K
C
      READ(8,80) S0,SHIFT
   80 FORMAT(2E12.5)
C
      WRITE(6,90)
   90 FORMAT(/' MATRIX HAS BEEN READ IN FROM FILE 8'/
     1 ' PERMUTATION IPR HAS BEEN READ IN'/)
C
C     EXPAND IROW AND ICOL TO INCLUDE DIAGONAL AND SUPER DIAGONAL
      KCOL(1) = 1 + ICOL(1)
      KNUM(1) = -1
      KROW(1) = 1
      IF (ICOL(1).EQ.0) GO TO 110
      KL = ICOL(1)
      DO 100 K = 1,KL
      KP1 = K+1
      KROW(KP1) = IROW(K)
  100 KNUM(KP1) = K
  110 KCOUNT = KCOL(1)
C
      DO 160 K = 2,N
      K1 = MIN(K-1,NZL)
```

```
      JL = 0
      JCOUNT = 0
      DO 140 J = 1,K1
      IF (ICOL(J).EQ.0) GO TO 140
      JF = JL + 1
      JL = JL + ICOL(J)
      DO 130 JJ = JF,JL
      IF (IROW(JJ)-K) 130,120,140
  120 KCOUNT = KCOUNT + 1
      JCOUNT = JCOUNT + 1
      KROW(KCOUNT) = J
      KNUM(KCOUNT) = JJ
      GO TO 140
  130 CONTINUE
  140 CONTINUE
      KCOUNT = KCOUNT + 1
      KROW(KCOUNT) = K
      KNUM(KCOUNT) = -K
      ITEMP = 0
      IF (K.LE.NZL) ITEMP = ICOL(K)
      KCOL(K) = JCOUNT + 1 + ITEMP
      IF (K.GT.NZL.OR.ICOL(K).EQ.0) GO TO 160
      KF = 1 + KL
      KL = KL + ICOL(K)
      DO 150 J = KF,KL
      KCOUNT = KCOUNT + 1
      KROW(KCOUNT) = IROW(J)
  150 KNUM(KCOUNT) = J
  160 CONTINUE
C     NTOTAL = N + 2*NZS
C     A-MATRIX INDEX LISTS HAVE BEEN EXPANDED
C
      WRITE(6,170)
  170 FORMAT(/' EXPANSION OF INDEX LISTS FOR C-MATRIX IS COMPLETED'/)
C
C     DETERMINE STRUCTURE OF B = P*C*P-TRANSPOSE
      IL = 0
      KCOUNT = 0
      DO 180 K = 1,N
  180 ICOL(K) = 0
      DO 270 K = 1,N
      J = IPR(K)
      JL = 0
      IF (J.EQ.1) GO TO 200
      JM1 = J - 1
      DO 190 JJ = 1,JM1
  190 JL = JL + KCOL(JJ)
  200 CONTINUE
      JF = JL + 1
      JL = JL + KCOL(J)
      ICOL(K) = KCOL(J)
      IF = IL + 1
      IL = IL + ICOL(K)
C
      DO 210 JJ = JF,JL
```

```
      KCOUNT = KCOUNT + 1                                               LOR01660
      JR = KROW(JJ)                                                     LOR01670
      JK = IPT(JR)                                                      LOR01680
      INUM(KCOUNT) = KNUM(JJ)                                           LOR01690
  210 IROW(KCOUNT) = JK                                                 LOR01700
C                                                                       LOR01710
C     ORDER IROW VECTOR BY INCREASING SIZE                              LOR01720
      IF (IF.EQ.IL) GO TO 240                                           LOR01730
      IF1 = IF + 1                                                      LOR01740
      DO 230 I = IF1,IL                                                 LOR01750
      IM1 = I-1                                                         LOR01760
      IMF = IM1 + IF                                                    LOR01770
      DO 220 L = IF,IM1                                                 LOR01780
      II = IMF - L                                                      LOR01790
      IF (IROW(II+1).GE.IROW(II)) GO TO 230                             LOR01800
      I0 = IROW(II)                                                     LOR01810
      IROW(II) = IROW(II+1)                                             LOR01820
      IROW(II+1) = I0                                                   LOR01830
      I0 = INUM(II)                                                     LOR01840
      INUM(II) = INUM(II+1)                                             LOR01850
      INUM(II+1) = I0                                                   LOR01860
  220 CONTINUE                                                          LOR01870
  230 CONTINUE                                                          LOR01880
  240 CONTINUE                                                          LOR01890
C                                                                       LOR01900
      DO 250 I = IF,IL                                                  LOR01910
      IF (INUM(I).LT.0) GO TO 260                                       LOR01920
  250 CONTINUE                                                          LOR01930
  260 INUM(I) = -J                                                      LOR01940
  270 CONTINUE                                                          LOR01950
C                                                                       LOR01960
C     GENERATE SPARSE MATRIX REPRESENTATION OF B-MATRIX                 LOR01970
      KCOUNT = 0                                                        LOR01980
      DO 280 K = 1,N                                                    LOR01990
  280 KCOL(K) = 0                                                       LOR02000
      DO 320 K = 1,N                                                    LOR02010
      KL = 0                                                            LOR02020
      DO 290 KK = 1,K                                                   LOR02030
  290 KL = KL + ICOL(KK)                                                LOR02040
      KK = KL+ 1                                                        LOR02050
  300 KK = KK - 1                                                       LOR02060
      IF (INUM(KK).GE.0) GO TO 300                                      LOR02070
      KCOL(K) = KL - KK                                                 LOR02080
      J = IPR(K)                                                        LOR02090
      BD(K) = AD(J)                                                     LOR02100
      KF = KK + 1                                                       LOR02110
      IF (KCOL(K).EQ.0) GO TO 320                                       LOR02120
      DO 310 JJ = KF,KL                                                 LOR02130
      KCOUNT = KCOUNT + 1                                               LOR02140
      KROW(KCOUNT) = IROW(JJ)                                           LOR02150
      KK = INUM(JJ)                                                     LOR02160
  310 BSD(KCOUNT) = ASD(KK)                                             LOR02170
  320 CONTINUE                                                          LOR02180
      NZL = 0                                                           LOR02190
      DO 330 K = 1,N                                                    LOR02200
```

```
      IF (KCOL(K).NE.0) NZL = K                                         LOR02210
  330 CONTINUE                                                          LOR02220
C     WE NOW HAVE B = P*A*P-TRANSPOSE IN SPARSE MATRIX FORMAT, WRITE TO LOR02230
C     FILE 9                                                            LOR02240
C                                                                       LOR02250
      JPERM = 1                                                         LOR02260
      WRITE(9,340) NZS,N,NZL,MATNO,JPERM                                LOR02270
  340 FORMAT(I10,2I6,I8,I6,' = NZS,N,NZL,MATNO,JPERM. BCOMPAC')         LOR02280
C                                                                       LOR02290
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS WRITTEN   LOR02300
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS WRITTEN   LOR02310
      WRITE(9,350) (KCOL(K), K=1,NZL)                                   LOR02320
      WRITE(9,350) (KROW(K), K=1,NZS)                                   LOR02330
  350 FORMAT(13I6)                                                      LOR02340
C     DIAGONAL IS WRITTEN FIRST, THEN NONZERO BELOW DIAGONAL ENTRIES    LOR02350
      WRITE(9,360) (BD(K), K=1,N)                                       LOR02360
      WRITE(9,360) (BSD(K), K=1,NZS)                                    LOR02370
  360 FORMAT(4E19.10)                                                   LOR02380
C                                                                       LOR02390
C     WRITE PERMUTATION                                                 LOR02400
      WRITE(9,350) (IPR(K), K=1,N)                                      LOR02410
C                                                                       LOR02420
      WRITE(9,370) S0,SHIFT                                             LOR02430
  370 FORMAT(2E12.5,' = S0 SHIFT'/                                      LOR02440
     1 ' ABOVE IS REORDERED MATRIX, B'/                                 LOR02450
     1 ' INPUT MATRIX SUPPLIED WAS C = S0*A + SHIFT*I'/                 LOR02460
     1 ' B = P*C*(P-TRANSPOSE),  B IS STORED IN SPARSE MATRIX FORMAT'/  LOR02470
     1 ' ROW(COL) I OF B CORRESPONDS TO ROW(COL) J OF C, J = IPR(I)'/   LOR02480
     1 ' NZS = TOTAL NUMBER OF SUBDIAGONAL NONZEROS IN B-MATRIX'/       LOR02490
     1 ' KCOL(K) = NUMBER OF SUBDIAGONAL NONZEROS IN COL K OF B'/       LOR02500
     1 ' KROW(K) = ROW INDEX OF SUBDIAGONAL NONZERO'/                   LOR02510
     1 ' SUBDIAGONAL NONZEROS IN B ARE STORED COLUMN BY COLUMN'/        LOR02520
     1 ' BD(K) = THE KTH DIAGONAL ELEMENT OF B'/                        LOR02530
     1 ' BSD(K) = NUMERICAL VALUE OF KTH SUBDIAGONAL NONZERO IN B'/     LOR02540
     1 ' IPR(K) = J MEANS THAT ROW J OF C CORRESPONDS TO ROW K OF B'/)  LOR02550
C                                                                       LOR02560
      WRITE(6,380)                                                      LOR02570
  380 FORMAT(' SPARSE FORMAT FOR B-MATRIX HAS BEEN WRITTEN TO FILE 9'/) LOR02580
      GO TO 410                                                         LOR02590
C                                                                       LOR02600
  390 WRITE(6,400)                                                      LOR02610
  400 FORMAT(/' LORDER PROGRAM TERMINATES BECAUSE MATRIX FILE SUPPLIED DLOR02620
     1ID NOT'/' CONTAIN A PERMUTATION'/)                                LOR02630
C                                                                       LOR02640
  410 CONTINUE                                                          LOR02650
C                                                                       LOR02660
C-----END OF LORDER---------------------------------------------------- LOR02670
      STOP                                                              LOR02680

      END                                                               LOR02690
```

```
C-----LFACT------------------------------------------------------------
C
C     NONPORTABLE CONSTRUCTIONS:
C     1.  FORMAT (4Z20).  TO AVOID COMPOUNDING FORMAT CONVERSION
C         ERRORS, THE MATRIX ENTRIES SHOULD BE IN MACHINE FORMAT,
C         (4Z20) FOR IBM/3081.
C
C     LFACT COMPUTES THE CHOLESKY FACTOR L FOR THE MATRIX B AND STORES
C     THIS FACTOR ON FILE 7.  B MUST BE A POSITIVE DEFINITE MATRIX.
C     THE PERMUTATION P (IN IPR), THE SCALE S0 AND THE SHIFT (IF ANY)
C     USED TO OBTAIN B FROM THE ORIGINAL MATRIX A ARE STORED AT THE END
C     OF FILE 7.   THAT IS,  B = S0*P*A*P' + SHIFT*I.  THE PROGRAM
C     ASSUMES THAT THE DATA READ FROM FILE 9 IS FOR THE B-MATRIX.
C
C-----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     1.  AD:  >= N, THE ORDER OF A-MATRIX.
C     3.  ASD:   >= NZT, THE NUMBER OF NONZERO SUBDIAGONAL ENTRIES
C                IN THE CHOLESKY FACTOR OF B.
C     4.  ICOL,IPR:   >= N
C     5.  IROW:   >=  NZT
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION ASD(10000),AD(3000)
      DOUBLE PRECISION ZERO,ONE,TEMP,S0,SHIFT
      INTEGER IROW(10000),ICOL(3000),IPR(3000)
      DOUBLE PRECISION DSQRT
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,5)
    5 FORMAT(/' LFACT PROGRAM, COMPUTE CHOLESKY FACTOR FOR POSITIVED DEF
     1INITE B-MATRIX'/' AND STORE THE FACTOR ON FILE 7'/)
C
C     SET PROGRAM PARAMETERS
      ONE = 1.0D0
      ZERO = 0.0D0
C
C     READ NUMBER OF NONZERO BELOW DIAGONAL ENTRIES, ORDER OF MATRIX,
C     INDEX OF LAST COLUMN CONTAINING NONZERO ENTRIES BELOW THE
C     DIAGONAL, MATRIX IDENTIFICATION NUMBER
      READ(9,15) NZS,N,NZL,MATNO,JPERM
   15 FORMAT(I10,2I6,I8,I6)
C
      WRITE(6,20) NZS,N,NZL,JPERM,MATNO
   20 FORMAT(I10,3I6,I8,' = NZS,N,NZL,JPERM,MATNO'/)
C
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS READ
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS READ
      READ(9,30) (ICOL(K), K=1,NZL)
      READ(9,30) (IROW(K), K=1,NZS)
   30 FORMAT(13I6)
C
C
      NZL1 = NZL + 1
```

```
      DO 40 K = NZL1,N
   40 ICOL(K) = 0
C
C     DIAGONAL IS READ (INCLUDING ANY ZERO ENTRIES), THEN NONZERO
C     BELOW DIAGONAL ENTRIES ARE READ IN
      READ(9,50) (AD(K), K=1,N)
      READ(9,50) (ASD(K), K=1,NZS)
   50 FORMAT(4E19.10)
C  50 FORMAT(4Z20)
C
      IF (JPERM.NE.0) READ(9,30) (IPR(K), K = 1,N)
C
      READ(9,55) S0,SHIFT
   55 FORMAT(2E12.5)
C
      WRITE(6,60)
   60 FORMAT(/' B-MATRIX HAS BEEN READ IN FROM FILE 9'/)
C
      IF (JPERM.NE.0) WRITE(6,65)
   65 FORMAT( ' PERMUTATION IPR HAS BEEN READ IN'/)
C
C     CALCULATE CHOLESKY FACTOR,  B = BL*(BL-TRANSPOSE)
      NZT = NZS
      NZL = N-1
      KL = 0
      DO 70 K = 1,N
C     CALCULATE KTH PIVOT FOR BL
      TEMP = AD(K)
C
      IF (AD(K).GT.ZERO) GO TO 80
C
      WRITE(6,90) K,AD(K)
   90 FORMAT(/I6,E15.8,' = K,AD(K)'/
     1' PIVOT IS NEGATIVE SO B-MATRIX IS NOT POSITIVE DEFINITE'/
     1' THEREFORE COMPUTATION OF CHOLESKY FACTOR TERMINATES'/)
      GO TO 240
C
   80 CONTINUE
      TEMP = DSQRT(TEMP)
      AD(K) = TEMP
      TEMP = ONE/TEMP
      IF(K.EQ.N.OR.ICOL(K).EQ.0) GO TO 70
      KF = KL + 1
      KL = KL + ICOL(K)
      DO 100 KK = KF,KL
      KR = IROW(KK)
      ASD(KK) = TEMP*ASD(KK)
  100 AD(KR) = AD(KR) - ASD(KK)**2
      IF (KF.EQ.KL) GO TO 70
      K1 = K+1
      DO 110 KK = KF,KL
      KR = IROW(KK)
      IF (KK.EQ.KL) GO TO 110
      KE = KL
      DO 120 KC = K1,KR
```

```
  120 KE= KE + ICOL(KC)
      KB = KE - ICOL(KR) + 1
      KK1 = KK + 1
      L = KB
      DO 130 LL = KK1,KL
      LR = IROW(LL)
      IF (ICOL(KR).EQ.0.OR.L.GT.KE) GO TO 140
  150 LC = IROW(L)
      IF (LC - LR) 160,170,140
  160 L = L + 1
      IF (L.LE.KE) GO TO 150
C     NEW NONZERO IN CHOLESKY FACTOR L
  140 NZT = NZT + 1
      L1 = L + 1
      NT = NZT + L1
      DO 180 KM = L1,NZT
      MK = NT - KM
      ASD(MK) = ASD(MK-1)
  180 IROW(MK) = IROW(MK-1)
      ICOL(KR) = ICOL(KR) + 1
      KE = KE + 1
      ASD(L) = -ASD(KK)*ASD(LL)
      IROW(L) = LR
      GO TO 130
C     UPDATE EXISTING ELEMENT
  170 ASD(L) = ASD(L) - ASD(KK)*ASD(LL)
  130 L = L + 1
  110 CONTINUE
   70 CONTINUE
C
C
C     FACTOR L HAS BEEN COMPUTED, STORE IN SPARSE FORMAT ON FILE 7
C
      WRITE(7,190) NZT,N,NZL,MATNO,JPERM
  190 FORMAT(I10,2I6,I8,I6,' = NZT,N,NZL,MATNO,JPERM. LCOMPAC')
C
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS WRITTEN
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS WRITTEN
      WRITE(7,200) (ICOL(K), K=1,NZL)
      WRITE(7,200) (IROW(K), K=1,NZT)
  200 FORMAT(13I6)
C     DIAGONAL IS WRITTEN FIRST, THEN NONZERO BELOW DIAGONAL ENTRIES
      WRITE(7,210) (AD(K), K=1,N)
      WRITE(7,210) (ASD(K), K=1,NZT)
  210 FORMAT(4Z20)
C 210 FORMAT(3E25.16)
      IF (JPERM.NE.0) WRITE(7,200) (IPR(K), K=1,N)
C
      WRITE(7,220) S0,SHIFT
  220 FORMAT(2E12.5,' = S0 SHIFT'/
     1 ' ABOVE IS CHOLESKY FACTOR FOR B-MATRIX'/
     1 ' IF JPERM = 0, THEN P = I.  C = S0*A * SHIFT*I'/
     1 ' B = P*C*P-TRANS = L*L-TRANS,  L IS STORED IN SPARSE FORMAT'/
     1 ' ROW(COL) I OF B CORRESPONDS TO ROW(COL) J OF C, J = IPR(I)'/
     1 ' NZT = TOTAL NUMBER OF SUBDIAGONAL NONZEROS IN L'/
```

```
     1 ' ICOL(K) = NUMBER OF SUBDIAGONAL NONZEROS IN COL K OF L'/       LFA01660
     1 ' IROW(K) = ROW INDEX OF SUBDIAGONAL NONZERO'/                   LFA01670
     1 ' SUBDIAGONAL NONZEROS IN L ARE STORED COLUMN BY COLUMN'/        LFA01680
     1 ' AD(K) = KTH DIAGONAL ELEMENT OF L'/                            LFA01690
     1 ' ASD(K) = KTH SUBDIAGONAL NONZERO IN L'/)                       LFA01700
C                                                                       LFA01710
      WRITE(6,230)                                                      LFA01720
  230 FORMAT(' CHOLESKY FACTOR HAS BEEN WRITTEN TO FILE 7 '/)           LFA01730
C                                                                       LFA01740
  240 CONTINUE                                                          LFA01750
C                                                                       LFA01760
C-----END OF LFACT---------------------------------------------------   LFA01770
      STOP                                                              LFA01780
      END                                                               LFA01790
```

```
C-----LTEST-----------------------------------------------------------
C
C     CONTAINS MAIN PROGRAM LTEST AND SAMPLE CMATS, CMATV, BSOLV
C     LTEST ALSO REQUIRES A RANDOM NUMBER GENERATOR.
C
C     LTEST GIVES A ROUGH CHECK ON THE CONDITION OF A MATRIX B BY
C     SOLVING B*X = B*V1 FOR X WHERE V1 IS A KNOWN, RANDOMLY-GENERATED
C     VECTOR.  SOLVING IS DONE, WITH AND WITHOUT ITERATIVE REFINEMENT.
C     IN BOTH CASES, X IS COMPARED WITH V1 AND THE ERRORS ARE
C     WRITTEN TO FILE 6.
C
C     VECTORS V0, V1, V2, VS, AND G ARE USED IN THE COMPUTATIONS.
C     NOTE THAT THE SUBROUTINE CMATS USED TO COMPUTE THE RESIDUAL
C     IN EXTENDED PRECISION FOR THE ITERATIVE REFINEMENT CALCULATION
C     REQUIRES AN EXTRA LONG V0 VECTOR OF LENGTH TWICE THE SIZE OF B.
C
C     NONPORTABLE CONSTRUCTIONS:
C     1.  THE ENTRY MECHANISM WHICH PASSES THE STORAGE LOCATIONS OF
C         ARRAYS AND PARAMETERS THAT DEFINE THE B-MATRIX TO THE
C         SUBROUTINES CMATV, CMATS, AND BSOLV.
C     2.  FORMATS (20A4) AND (4Z20).  TO AVOID COMPOUNDING FORMAT
C         CONVERSION ERRORS, MATRIX ENTRIES SHOULD BE STORED IN
C         MACHINE FORMAT, ((4Z20) FOR IBM/3081). ALSO FREE FORMAT
C         (5,*).
C     3.  REAL*16 VARIABLES IN CMATS SUBROUTINE.
C
C
C----------------------------------------------------------------------
      DOUBLE PRECISION ASD(10000),AD(3000),BSD(20000),BD(3000)
      DOUBLE PRECISION V0(6000),V1(3000),V2(3000),VS(3000)
      DOUBLE PRECISION ZERO,ONE,TEMP,SUM
      DOUBLE PRECISION ERROR0,ERROR1,ENORM0,ENORM1
      REAL EXPLAN(20),G(3000)
      INTEGER  IROW(20000),ICOL(3000),KROW(30000),KCOL(3000),SVSEED
      DOUBLE PRECISION FINPRO
      DOUBLE PRECISION DABS, DMAX1, DSQRT
C----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     1.  AD, BD:  >= N, THE ORDER OF A-MATRIX.
C     2.  ASD:   >= NZS, THE NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN B.
C     3.  BSD:   >= NZT, THE NUMBER OF NONZERO SUBDIAGONAL ENTRIES
C                IN THE CHOLESKY FACTOR OF B.
C     5.  ICOL, KCOL:   >= N
C     6.  KROW:   >=  NZS
C     7.  IROW:   >=  NZT
C     8.  V1,V2,VS:   >=   N
C     9.  V0:   >=   2*N
C
C----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LTEST PROGRAM, ROUGH CHECK ON NUMERICAL CONDITION OF GIV
     1EN MATRIX'/)
C
```

```
C     SET PROGRAM PARAMETERS
      ONE = 1.0D0
      ZERO = 0.0D0
C
C     READ INPUT HEADER
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C     READ IN IN FREE FORMAT USER-SPECIFIED PARAMETERS FROM FILE 5
      READ(5,20) EXPLAN
      READ(5,*) SVSEED
C
C     READ NUMBER OF NONZERO BELOW DIAGONAL ENTRIES, ORDER OF MATRIX,
C     INDEX OF LAST COLUMN CONTAINING NONZERO ENTRIES BELOW THE
C     DIAGONAL, MATRIX IDENTIFICATION NUMBER
      READ(9,30) NZS,N,NZL,MATNO,JPERM
   30 FORMAT(I10,2I6,I8,I6)
C
      WRITE(6,40) NZS,N,NZL,JPERM,MATNO,SVSEED
   40 FORMAT(I10,3I6,' = NZS,N,NZL,JPERM'/
     1 I8,I12,' = MATNO,SVSEED'/)
C
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS READ
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS READ
      READ(9,50) (KCOL(K), K=1,NZL)
      READ(9,50) (KROW(K), K=1,NZS)
   50 FORMAT(13I6)
C
C
      NZL1 = NZL + 1
      DO 60 K = NZL1,N
   60 KCOL(K) = 0
C
C     DIAGONAL IS READ (INCLUDING ANY ZERO ENTRIES), THEN NONZERO
C     BELOW DIAGONAL ENTRIES ARE READ IN
      READ(9,70) (AD(K), K=1,N)
      READ(9,70) (ASD(K), K=1,NZS)
   70 FORMAT(4E19.10)
C
      WRITE(6,80)
   80 FORMAT(/' B-MATRIX HAS BEEN READ IN FROM FILE 9'/)
C
C-----------------------------------------------------------------------
C     ENTRIES TO CMATS AND CMATV SUBROUTINES
      CALL CMATSE(ASD,AD,KCOL,KROW,N,NZL)
      CALL CMATVE(ASD,AD,KCOL,KROW,N,NZL)
C-----------------------------------------------------------------------
C
C     READ CHOLESKY FACTOR FROM FILE 7
C
      READ(7,90) NZT,N,NZL,MATNO,JPERM
   90 FORMAT(I10,2I6,I8,I6)
C
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS READ
```

```
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS READ
      READ(7,100) (ICOL(K), K=1,NZL)
      READ(7,100) (IROW(K), K=1,NZT)
  100 FORMAT(13I6)
C     DIAGONAL IS READ FIRST, THEN NONZERO BELOW DIAGONAL ENTRIES
      READ(7,110) (BD(K), K=1,N)
      READ(7,110) (BSD(K), K=1,NZT)
  110 FORMAT(4Z20)
C  90 FORMAT(3E25.16)
C
C-----------------------------------------------------------------------
C     ENTRY TO BSOLV SUBROUTINE, PASS FACTOR OF B
      CALL BSOLVE(BSD,BD,ICOL,IROW,N,NZT,NZL)
C-----------------------------------------------------------------------
C
C     SOLVE B*X = B*V1 WITH AND WITHOUT ITERATIVE REFINEMENT, COMPARE
C     ERRORS IN SOLVING AS A ROUGH CHECK ON THE CONDITION OF THE
C     MATRIX B.
C
      IIX = SVSEED
C
C-----------------------------------------------------------------------
C     COMPUTES RANDOM VECTOR FOR USE IN RIGHT-HAND SIDE
      CALL GENRAN(IIX,G,N)
C-----------------------------------------------------------------------
C
      DO 120 K = 1,N
  120 V1(K) = G(K)
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V1(1),1,V1(1),1)
C-----------------------------------------------------------------------
      SUM = ONE/DSQRT(SUM)
C
      DO 130 K = 1,N
  130 V1(K) = V1(K)*SUM
C
      SUM = ZERO
C
C---------------------------------------------------------------------
C     COMPUTE V2 = RHS = B*V1  C = SO*A + SHIFT*I  B = P*C*P'
C     VS = B(INVERSE)*V2
      CALL CMATV(V1,V2,SUM)
      CALL BSOLV(VS,V2)
C---------------------------------------------------------------------
C
      SUM = ZERO
      ERROR0 = ZERO
      DO 140 K = 1,N
      TEMP = DABS(V1(K) - VS(K))
      SUM = SUM + TEMP*TEMP
  140 ERROR0 = DMAX1(ERROR0,TEMP)
      ENORM0 = DSQRT(SUM)
C
      WRITE(6,150) ENORM0,ERROR0
```

```
  150 FORMAT(6X,'ENORM0',6X,'ERROR0'/2E12.4/
     1 ' ENORM0  = NORM (V1 - VS),     VS = BI*(B*V1)'/
     1 ' ERROR0 = MAX DABS(V1(K) - VS(K)),  K = 1,N'/)
C
      SUM = ONE
C
C-----------------------------------------------------------------------
C     CALCULATE RESIDUAL IN EXTENDED PRECISION V2 = B*VS - V2
C     THEN DO ITERATIVE REFINEMENT
      CALL CMATS(VS,V2,V0,SUM)
      CALL BSOLV(V2,V2)
C-----------------------------------------------------------------------
C
      DO 160 K = 1,N
  160 VS(K) = VS(K) - V2(K)
C
      SUM = ZERO
      ERROR1 = ZERO
      DO 170 K = 1,N
      TEMP = DABS(V1(K) - VS(K))
      SUM = SUM + TEMP*TEMP
  170 ERROR1 = DMAX1(ERROR1,TEMP)
      ENORM1 = DSQRT(SUM)
C
      WRITE(6,180) ENORM1,ERROR1
  180 FORMAT(6X,'ENORM1',6X,'ERROR1'/2E12.4/
     1 ' ERROR AFTER ITERATIVE REFINEMENT'/
     1 ' ENORM1  = NORM (V1 - VS),     VS = BI*(B*V1)'/
     1 ' ERROR1 = MAX DABS(V1(K) - VS(K)),  K = 1,N'/)
C
      STOP
C-----END OF LTEST------------------------------------------------------
      END
C
C----CMATS--------------------------------------------------------------
C
C     REAL, SYMMETRIC, SPARSE MATRIX-VECTOR MULTIPLY USING EXTENDED
C     PRECISION.  CALCULATES U = B*W - SUM*U FOR USE IN ITERATIVE
C     REFINEMENT.  MATRIX B STORED IN SPARSE FORMAT.
C
      SUBROUTINE CMATS(W,U,Z,SUM)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  U(1),W(1),BSD(1),BD(1),SUM
      REAL*16 Z(1),T0,T1,T2,S0
      INTEGER  IROW(1),ICOL(1)
C-----------------------------------------------------------------------
      S0 = SUM
C
      DO 10 I = 1,N
      T0 = BD(I)
      T1 = W(I)
      T2 = U(I)
   10 Z(I) = T0*T1-S0*T2
C
```

```
      LLAST = 0
C
      DO 30 J = 1,NZL
C
      IF (ICOL(J).EQ.0) GO TO 30
C
      LFIRST = LLAST + 1
      LLAST = LLAST + ICOL(J)
C
      DO 20 L = LFIRST,LLAST
      I = IROW(L)
      T0 = BSD(L)
      T1 = W(J)
      T2 = W(I)
C
      Z(I) = Z(I) + T0*T1
      Z(J) = Z(J) + T0*T2
C
   20 CONTINUE
C
   30 CONTINUE
C
      DO 40 I =1,N
   40 U(I) = Z(I)
C
      RETURN
C
C-----------------------------------------------------------------------
      ENTRY CMATSE(BSD,BD,ICOL,IROW,N,NZL)
C-----------------------------------------------------------------------
C
      RETURN
C-----END OF CMATS------------------------------------------------------
      END
C
C-----CMATV-------------------------------------------------------------
C
C     SYMMETRIC, SPARSE MATRIX-VECTOR MULTIPLY, B MATRIX STORED
C     IN SPARSE FORMAT.  CMATV CALCULATES U = B*W - SUM*U
C
      SUBROUTINE CMATV(W,U,SUM)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  U(1),W(1),BSD(1),BD(1),SUM
      INTEGER  KROW(1),KCOL(1)
C-----------------------------------------------------------------------
C
      DO 10 I = 1,N
   10 U(I) = BD(I)*W(I) - SUM*U(I)
C
      LLAST = 0
C
      DO 30 J = 1,NZL
C
      IF (KCOL(J).EQ.0) GO TO 30
```

```
C
      LFIRST = LLAST + 1
      LLAST = LLAST + KCOL(J)
C
      DO 20 L = LFIRST,LLAST
      I = KROW(L)
C
      U(I) = U(I) + BSD(L)*W(J)
      U(J) = U(J) + BSD(L)*W(I)
C
   20 CONTINUE
C
   30 CONTINUE
C
      RETURN
C
C-----------------------------------------------------------------------
      ENTRY CMATVE(BSD,BD,KCOL,KROW,N,NZL)
C-----------------------------------------------------------------------
C
      RETURN
C-----END OF CMATV------------------------------------------------------
      END
C
C-----BSOLV-----------------------------------------------------------
C
C     SOLVES B*U = V WHERE  B = L*L'.
C     FIRST SOLVES L*U = V FOR U, THEN SOLVES L'*U = U FOR U
C
      SUBROUTINE BSOLV(U,V)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION   AD(1),ASD(1),U(1),V(1),TEMP
      INTEGER  ICOL(1),IROW(1)
C-----------------------------------------------------------------------
      KL = 0
      DO 10 K = 1,N
   10 U(K) = V(K)
      DO 30 K = 1,N
      TEMP = U(K)/AD(K)
      U(K) = TEMP
      IF (ICOL(K).EQ.0.OR.K.EQ.N) GO TO 30
      KF = KL + 1
      KL = KL + ICOL(K)
      DO 20 KK = KF,KL
      KR = IROW(KK)
   20 U(KR) = U(KR) - TEMP*ASD(KK)
   30 CONTINUE
C
      NP1 = N+1
      KF = NZT + 1
      DO 50 K = 1,N
      L = NP1 - K
      TEMP = U(L)
      IF (ICOL(L).EQ.0.OR.L.EQ.N) GO TO 50
```

```
      KL = KF - 1                                                       LTE03310
      KF = KF - ICOL(L)                                                 LTE03320
      DO 40 LL = KF,KL                                                  LTE03330
      LR = IROW(LL)                                                     LTE03340
   40 TEMP = TEMP - ASD(LL)*U(LR)                                       LTE03350
   50 U(L) = TEMP/AD(L)                                                 LTE03360
C                                                                       LTE03370
      RETURN                                                            LTE03380
C                                                                       LTE03390
C-----------------------------------------------------------------------LTE03400
      ENTRY BSOLVE(ASD,AD,ICOL,IROW,N,NZT,NZL)                          LTE03410
C-----------------------------------------------------------------------LTE03420
C                                                                       LTE03430
C-----END OF BSOLV------------------------------------------------------LTE03440
      RETURN                                                            LTE03450
      END                                                               LTE03460
```

4.6 LIVAL: LIVEC: File Definitions, Sample Input Files

Below is a listing of the input/output files which are accessed by the real symmetric Lanczos eigenvalue program, LIVAL. Included also is a sample of the input file which LIVAL requires on file 5. The parameters in this file are supplied in free format. LIVAL computes eigenvalues of real symmetric matrices B^{-1} on user-specified intervals where $B = PCP^T$ with $C = (SCALE) * A + (SHIFT) * I$ where $SCALE$ and $SHIFT$ are scalars. The sample codes assume that C is positive definite and has a reasonable condition number. The permutation matrix P Is used to preserve the sparseness of the given matrix in the Cholesky factorization, $B = LL^T$. The user could replace the BSOLVE subroutine provided here by another more general factorization subroutine.

```
Sample Specification of the Input/Output Files for LIVAL
-----------------------------------------------------------------
 LIVAL EXEC LANCZOS EIGENVALUE CALCULATION USING FACTORIZATION
FI 06 TERM
FILEDEF  1 DISK &1        NHISTORY  A (RECFM F LRECL 80 BLOCK 80
FILEDEF  2 DISK &1        HISTORY   A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1        GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1        ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK LIVAL     INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  7 DISK &1        LDATA     A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1        DISTINCT  A (RECFM F LRECL 80 BLOCK 80
LOAD   LIVAL   LESUB   LIMULT
-----------------------------------------------------------------
```

```
Sample Input File for LIVAL
--------------------------------------------------------------------
 LIVAL EIGENVALUE COMPUTATION, NO REORTHOGONALIZATION
 USING INVERSE OF REAL SYMMETRIC MATRIX VIA FACTORIZATION
LINE 1      N    KMAX    NMEVS     MATNO     S0    SHIFT
         528    2640        2    721830    1.0       0.
LINE 2   SVSEED    RHSEED      MXINIT    MXSTUR
       49302312   5731029           5    100000
LINE 3   ISTART     ISTOP
              0         1
LINE 4     IHIS     IDIST  IWRITE
             1         0       1
LINE 5   RELTOL (RELATIVE TOLERANCE IN 'COMBINING' GOODEV)
    .0000000001
LINE 6   MB(1)   MB(2)   MB(3)   MB(4)    (ORDERS OF T(1,MEV) )
           100     125
LINE 7    NINT     (NUMBER OF SUB-INTERVALS FOR BISEC)
             1
LINE 8    LB(1)    LB(2)   LB(3)   LB(4)  (INTERVAL LOWER BOUNDS)
            1.0
LINE 9     UB(1)    UB(2)   UB(3)   UB(4)  (INTERVAL UPPER BOUNDS)
           100.0
----------------------------------------------------------------------
```

Below is a listing of the input/output files which are accessed by the real symmetric Lanczos eigenvector program, LIVEC. Included also is a sample of the input file which LIVEC requires on file 5. The parameters in this file are supplied in free format. LIVEC computes eigenvectors for each of a user-specified subset of the eigenvalues computed by the companion program LIVAL. The matrix used in the eigenvector computation is a scaled, shifted and inverted version of a given matrix. Inversion is accomplished via matrix factorization.

```
Sample Specifications of the Input/Output Files for LIVEC
----------------------------------------------------------------
 LIVEC EXEC, EIGENVECTORS FOR INVERSE OF REAL SYMMETRIC MATRIX
FI 06 TERM
FILEDEF  2 DISK &1       HISTORY   A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1       GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1       ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK LIVEC    INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  7 DISK &1       LDATA     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  9 DISK &1       ERREST    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 10 DISK &1       BOUNDS    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1       TEIGVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 12 DISK &1       RITZVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 13 DISK &1       PAIGE     A (RECFM F LRECL 80 BLOCK 80
LOAD  LIVEC  LESUB  LIMULT
----------------------------------------------------------------
```

```
Sample Input File for LIVEC
----------------------------------------------------------------------
 LIVEC INPUT LANCZOS EIGENVECTOR COMPUTATIONS, NO REORTHOGONALIZATION
LINE 1  MATNO     N    S0   SHIFT  JPERM (ID,SIZE,SCALE,SHIFT,PERMUT?
           20  2161   -1.0     0.01     0
LINE 2  MDIMTV    MDIMRV  MBETA (MAX.DIMENSIONS,TVEC,RITVEC AND BETA
         10000     10000   2000
LINE 3      RELTOL
        .0000000001
LINE 4  MBOUND   NTVCON SVTVEC IREAD (FLAGS
           0        1      0     1
LINE 5  TVSTOP   LVCONT ERCONT  IWRITE (FLAGS
           0        1      1       1
LINE 6    RHSEED  (RANDOM GENERATOR SEED FOR STARTING VECTOR IN INVERM)
        45329517
-------------------------------------------------------------------------
```

Chapter 5

Real Symmetric Generalized Problems

5.1 Introduction

The FORTRAN codes in this Chapter address the question of computing distinct eigenvalues and corresponding eigenvectors of a real symmetric generalized eigenvalue problem. Given two real symmetric matrices A and B, where B is positive definite and its Cholesky factors are available, these codes compute real scalars λ and corresponding real-valued vectors $x \neq 0$ such that

$$Ax = \lambda Bx. \tag{5.1.1}$$

Given a real symmetric positive definite matrix B, the Cholesky decomposition of B has the form

$$B = LL^T, \tag{5.1.2}$$

where L is a lower triangular matrix. Real symmetric matrices and Cholesky factorizations are discussed in detail in Stewart [24]. See Section 2.1 for a brief summary of the properties of real symmetric matrices which we use.

Theoretically, this type of real symmetric generalized problem is equivalent to the following real symmetric problem:

$$L^{-1}AL^{-T}y = \lambda y, \quad y = L^T x. \tag{5.1.3}$$

Therefore, we could solve this type of generalized problem by applying the real symmetric Lanczos procedure given in Chapter 2 directly to the composite matrix $C \equiv L^{-1}AL^{-T}$ given in Eqn(5.1.3). However, we prefer to work directly with the generalized problem. In this setting the role of the B-matrix in the single-vector Lanczos computations is clearly displayed.

The single-vector Lanczos codes in this chapter can be used to compute either a very few or very many of the distinct eigenvalues of the given real symmetric generalized problem. The documentation for these codes is contained in Section 2.2. As in the real symmetric case, the AB-multiplicity of a given computed 'good' Lanczos eigenvalue can be obtained only with additional computation, and the modifications required to do this additional computation are not included in the enclosed versions of these codes.

We use the following 'generalized' Lanczos recursion. For $i = 1, 2, \ldots m$ and a randomly-generated starting vector v_1 with $\|v_1\|_B = 1$, generate Lanczos vectors v_i using the following recursion.

$$\beta_{i+1} B v_{i+1} = A v_i - \alpha_i B v_i - \beta_i B v_{i-1} \tag{5.1.4}$$

where

$$\begin{aligned} \alpha_i &\equiv v_i^T (A v_i - \beta_i B v_{i-1}) \\ \beta_{i+1} &\equiv \|L^{-1}(A v_i - \alpha_i B v_i - \beta_i B v_{i-1})\| \end{aligned} \tag{5.1.5}$$

By construction, the B-norm of each Lanczos vector is one. That is, for all i, $\|v_i\|_B \equiv \left(v_i^T B v_i\right)^{1/2} = 1$.

The B-norm is used because it is the 'natural' norm for real symmetric generalized problems when the B-matrix is positive definite. Given any two distinct eigenvalues λ and μ of Eqn(5.1.1), and corresponding eigenvectors x and y, we have that $x^T B y = 0$. That is, the eigenvectors are orthogonal w.r.t. the B-norm, and the eigenvectors form a complete set of vectors. The positive definiteness of B is essential. The closer B is to being singular or indefinite, the less stable these computations will be. The generalized Lanczos recursion in Eqns (5.1.4) and (5.1.5) generates a family of real symmetric tridiagonal matrices (T-matrices) whose sizes are specified by the user.

LGVAL, the main program for the real symmetric generalized computations, calls the subroutine BISEC to compute eigenvalues of the specified tridiagonal T-matrices on the user-specified intervals. BISEC simultaneously computes these T-eigenvalues with their T-multiplicities and sorts the computed T-eigenvalues into two classes, the 'good' T-eigenvalues and the 'spurious' T-eigenvalues. The 'good' T-eigenvalues are accepted as approximations to eigenvalues of the generalized problem. The accuracy of these 'good' T-eigenvalues as eigenvalues of the generalized problem is then estimated using error estimates computed by the subroutine INVERR. Error estimates are computed only for isolated 'good' T-eigenvalues. All other 'good' T-eigenvalues are assumed to have converged. Convergence is then checked. If convergence has not yet occurred and a larger T-matrix has been specified by the user, the program will continue on to the larger T-matrix, repeating the above procedure on this larger matrix. After each T-matrix eigenvalue computation the corresponding approximations to the eigenvalues of the user-specified matrix A are computed and included in the output.

Once the eigenvalues have been computed accurately enough, the user can select a subset of the 'converged' eigenvalues for which eigenvectors are to be computed. The main program LGVEC, for computing eigenvectors of the real symmetric generalized problem using a factorization of B, is used to compute the desired eigenvectors.

All of the computations are done in double precision arithmetic. Once the Lanczos matrices have been computed, the remaining computations use the same subroutines which are used in the real symmetric case discussed in Chapter 2. In addition to the programs and subroutines provided here, the user must supply a subroutine USPECA which defines and initializes the A-matrix and a subroutine USPECB which defines and initializes the factors of the B-matrix. A subroutine AMATV which computes matrix-vector multiplies Ax for the A-matrix, and a subroutine BSOLV which solves the system of equations $Bz = v$ must also be supplied. These subroutines must be constructed in such a way as to take advantage of the sparsity (and/or structure) of the two user-supplied matrices A and B and such that they are accurate.

The optional preprocessing programs PERMUT, LORDER, LFACT, and LTEST listed in Chapter 4 can also be used with the codes in this chapter. PERMUT calls the SPARSPAK Library [9] to attempt to identify a reordering or permutation P of the given matrix B for which the sparseness of B is preserved under the factorization of the permuted matrix. LORDER takes a given matrix C and permutation P and computes the sparse format for the permuted matrix, PCP^T . LFACT computes the Cholesky factors of a given positive definite matrix. LTEST performs a very crude check on the numerical condition of the matrix supplied to it, by solving a system of equations with and without iterative refinement, LINPACK [7]. Obviously, if the B-matrix is permuted then the A-matrix must be subjected to the same permutation. These codes assume that the Cholesky factor supplied in the subroutine USPECB corresponds to the permuted B-matrix and that the AMATV subroutine supplied corresponds to the corresponding permuted A-matrix. Thus, the Lanczos codes compute the eigenvalues and eigenvectors of the permuted problem. The permutation (if any) is then unwrapped in the eigenvector program LGVEC.

5.2 LGVAL: Main Program, Eigenvalue Computations

```
C-----LGVAL  (EIGENVALUES, GENERALIZED SYMMETRIC PROBLEM)---------------LGV00010
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)             LGV00020
C            Los Alamos National Laboratory                             LGV00030
C            Los Alamos, New Mexico 87544                               LGV00040
C                                                                       LGV00050
C            E-mail:  cullumj@lanl.gov                                  LGV00060
C                                                                       LGV00070
C  These codes are copyrighted by the authors.  These codes             LGV00080
C  and modifications of them or portions of them are NOT to be          LGV00090
C  incorporated into any commercial codes or used for any other         LGV00100
C  commercial purposes such as consulting for other companies,          LGV00110
C  without legal agreements with the authors of these Codes.            LGV00120
C  If these Codes or portions of them are used in other scientific or   LGV00130
C  engineering research works the names of the authors of these codes   LGV00140
C  and appropriate references to their written work are to be           LGV00150
C  incorporated in the derivative works.                                LGV00160
C                                                                       LGV00170
C  This header is not to be removed from these codes.                   LGV00180
C                                                                       LGV00190
C         REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4            LGV00191
C         Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsLGV00192
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   LGV00193
C         Applied Mathematics, 2002. SIAM Publications,                 LGV00194
C         Philadelphia, PA. USA                                         LGV00195
C                                                                       LGV00196
C                                                                       LGV00200
C     CONTAINS MAIN PROGRAM FOR COMPUTING DISTINCT EIGENVALUES OF       LGV00210
C     A*X = EVAL*B*X WHERE A AND B ARE REAL SYMMETRIC MATRICES,         LGV00220
C     B IS POSITIVE DEFINITE, AND THE CHOLESKY FACTORS OF B             LGV00230
C     ARE AVAILABLE FOR USE IN THE PROCEDURE.  PROCEDURE USES           LGV00240
C     GENERALIZATION OF LANCZOS TRIDIAGONALIZATION WITHOUT ANY          LGV00250
C     REORTHGONALIZATION.                                               LGV00260
C                                                                       LGV00270
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE               LGV00280
C     CONSTRUCTIONS                                                     LGV00290
C                                                                       LGV00300
C     1.  DATA/MACHEP/ STATEMENT                                        LGV00310
C     2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                        LGV00320
C     3.  FORMAT(20A4) USED WITH EXPLANATORY HEADER EXPLAN.             LGV00330
C     4.  HEXADECIMAL FORMAT (4Z20) USED IN ALPHA/BETA FILES 1 AND 2.   LGV00340
C                                                                       LGV00350
C-----------------------------------------------------------------------LGV00360
C                                                                       LGV00370
      DOUBLE PRECISION  ALPHA(5000),BETA(5001)                          LGV00380
      DOUBLE PRECISION  V1(5000),V2(5000),VS(5000)                      LGV00390
      DOUBLE PRECISION  LB(20),UB(20)                                   LGV00400
      DOUBLE PRECISION  BTOL,GAPTOL,TTOL,MACHEP,EPSM,RELTOL             LGV00410
      DOUBLE PRECISION  SCALE1,SCALE2,SCALE3,SCALE4,BISTOL,CONTOL,MULTOLLGV00420
      DOUBLE PRECISION  ONE,ZERO,TEMP,TKMAX,BETAM,BKMIN,T0,T1           LGV00430
      REAL  G(5000),EXPLAN(20)                                          LGV00440
      INTEGER  MP(5000),NMEV(20)                                        LGV00450
```

```
      INTEGER  SVSEED,RHSEED,SVSOLD
      INTEGER IABS
      REAL  ABS
      DOUBLE PRECISION  DABS, DSQRT, DFLOAT
      EXTERNAL LSOLV, AMATV
C
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.0D0*MACHEP
C-----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     DIMENSION OF V2 ASSUMES THAT NO MORE THAN KMAX/2 EIGENVALUES
C     OF THE LANCZOS T-MATRICES ARE BEING COMPUTED IN ANY ONE OF THE
C     SUB-INTERVALS BEING CONSIDERED.  V2 CONTAINS THE UPPER AND LOWER
C     BOUNDS FOR EACH T-EIGENVALUE BEING COMPUTED BY BISEC IN ANY ONE
C     GIVEN INTERVAL.
C
C     1.  ALPHA: >= KMAX,   BETA: >= (KMAX+1)
C     2.  V1:  >= MAX(N,KMAX+1)
C     3.  V2,VS:   >= MAX(N,KMAX)
C     4.  G:  >= MAX(N,2*KMAX)
C     5.  MP:    >= KMAX
C     6.  LB,UB:  >= NUMBER OF SUBINTERVALS SUPPLIED TO BISEC.
C     7.  NMEV:  >= NUMBER OF T-MATRICES ALLOWED.
C     8.  EXPLAN:  DIMENSION IS 20.
C
C
C     IMPORTANT TOLERANCES OR SCALES THAT ARE USED REPEATEDLY
C     THROUGHOUT THE PROGRAM ARE THE FOLLOWING:
C     SCALED MACHINE EPSILON:  TTOL = TKMAX*EPSM WHERE
C     EPSM = 2*MACHINE EPSILON AND
C     TKMAX = MAX(|ALPHA(J)|,BETA(J), J = 1,MEV)
C     BISEC CONVERGENCE TOLERANCE:  BISTOL = DSQRT(1000+MEV)*TTOL
C     BISEC T-MULTIPLICITY TOLERANCE:  MULTOL = (1000+MEV)*TTOL
C     LANCZOS CONVERGENCE TOLERANCE:   CONTOL = BETA(MEV+1)*1.D-10
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LANCZOS EIGENVALUE PROCEDURE FOR REAL SYMMETRIC GENERALI
     1ZED PROBLEMS,'/' A*X = EVAL*B*X, B POSITIVE DEFINITE WITH CHOLESKY
     1 FACTORS AVAILABLE'/)
C
C     SET PROGRAM PARAMETERS
C     SCALEK ARE USED IN TOLERANCES NEEDED IN SUBROUTINES LUMP,
C     ISOEV AND PRTEST.  USER MUST NOT MODIFY THEM.
      SCALE1 = 5.0D2
      SCALE2 = 5.0D0
      SCALE3 = 5.0D0
      SCALE4 = 1.0D4
      ONE  = 1.0D0
      ZERO = 0.0D0
      BTOL = 1.0D-8
C     BTOL = EPSM
      GAPTOL = 1.0D-8
```

```
      ICONV = 0
      MOLD = 0
      MOLD1 = 1
      ICT = 0
      MMB = 0
      IPROJ = 0
C-----------------------------------------------------------------------
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 5 (FREE FORMAT)
C
C     READ USER-PROVIDED HEADER FOR RUN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C     READ ORDER OF MATRICES (N) , MAXIMUM ORDER OF T-MATRIX (KMAX),
C     NUMBER OF T-MATRICES ALLOWED (NMEVS), AND MATRIX IDENTIFICATION
C     NUMBERS (MATNOA AND MATNOB)
      READ(5,20) EXPLAN
      READ(5,*) N,KMAX,NMEVS,MATNOA,MATNOB
C
C     READ SEEDS FOR LANCZS AND INVERR SUBROUTINES (SVSEED AND RHSEED)
C     READ MAXIMUM NUMBER OF ITERATIONS ALLOWED FOR EACH INVERSE
C     ITERATION (MXINIT) AND MAXIMUM NUMBER OF STURM SEQUENCES
C     ALLOWED (MXSTUR)
      READ(5,20) EXPLAN
      READ(5,*) SVSEED,RHSEED,MXINIT,MXSTUR
C
C     ISTART = (0,1):  ISTART = 0 MEANS ALPHA/BETA FILE IS NOT
C     AVAILABLE.  ISTART = 1 MEANS ALPHA/BETA FILE IS AVAILABLE ON
C     FILE 2.
C     ISTOP = (0,1):  ISTOP = 0 MEANS PROCEDURE GENERATES ALPHA/BETA
C     FILE AND THEN TERMINATES.  ISTOP = 1 MEANS PROCEDURE GENERATES
C     ALPHAS/BETAS IF NEEDED AND THEN COMPUTES EIGENVALUES AND ERROR
C     ESTIMATES AND THEN TERMINATES.
      READ(5,20) EXPLAN
      READ(5,*) ISTART,ISTOP
C
C     IHIS = (0,1):  IHIS = 0 MEANS ALPHA/BETA FILE IS NOT WRITTEN
C     TO FILE 1.  IHIS = 1 MEANS ALPHA/BETA FILE IS WRITTEN TO FILE 1.
C     IDIST = (0,1):  IDIST = 0 MEANS DISTINCT EIGENVALUES OF
C     ARE NOT WRITTEN TO FILE 11.  IDIST = 1 MEANS DISTINCT
C     EIGENVALUES ARE WRITTEN TO FILE 11.
C     IWRITE = (0,1):  IWRITE = 0 MEANS NO INTERMEDIATE OUTPUT
C     FROM THE COMPUTATIONS IS WRITTEN TO FILE 6.  IWRITE = 1 MEANS
C     EIGENVALUES AND ERROR ESTIMATES ARE WRITTEN TO FILE 6
C     AS THEY ARE COMPUTED.
      READ(5,20) EXPLAN
      READ(5,*) IHIS,IDIST,IWRITE
C
C     READ IN THE RELATIVE TOLERANCE (RELTOL) FOR USE IN THE
C     SPURIOUS, T-MULTIPLICITY, AND PRTESTS.
      READ(5,20) EXPLAN
      READ(5,*) RELTOL
```

```
C
C     READ IN THE SIZES OF THE T-MATRICES TO BE CONSIDERED.
      READ(5,20) EXPLAN
      READ(5,*) (NMEV(J), J=1,NMEVS)
C
C     READ IN THE NUMBER OF SUBINTERVALS TO BE CONSIDERED.
      READ(5,20) EXPLAN
      READ(5,*) NINT
C
C     READ IN THE LEFT-END POINTS OF THE SUBINTERVALS TO BE CONSIDERED.
C     THESE MUST BE IN ALGEBRAICALLY-INCREASING ORDER
      READ(5,20) EXPLAN
      READ(5,*) (LB(J), J=1,NINT)
C
C     READ IN THE RIGHT-END POINTS OF THE SUBINTERVALS TO BE CONSIDERED.
C     THESE MUST BE IN ALGEBRAICALLY-INCREASING ORDER
      READ(5,20) EXPLAN
      READ(5,*) (UB(J), J=1,NINT)
C
C-----------------------------------------------------------------------
C     INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED MATRICES
C     AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE
C     MATRIX-VECTOR MULTIPLY SUBROUTINE AMATV AND THE SOLVE
C     SUBROUTINE LSOLV.
C
      CALL USPECA(N,MATNOA)
      CALL USPECB(N,MATNOB)
C
C-----------------------------------------------------------------------
C
C     MASK UNDERFLOW AND OVERFLOW
      CALL MASK
C
C-----------------------------------------------------------------------
C
C     WRITE TO FILE 6, A SUMMARY OF THE PARAMETERS FOR THIS RUN
C
      WRITE(6,30) MATNOA,MATNOB,N,KMAX
   30 FORMAT(/3X,'A-MATRIX ID',3X,'B-MATRIX ID',4X,'ORDER OF A',4X,
     1'MAX ORDER OF T'/I14,I14,I14,I18/)
C
      WRITE(6,40) ISTART,ISTOP
   40 FORMAT(/2X,'ISTART',3X,'ISTOP'/2I8/)
C
      WRITE(6,50) IHIS,IDIST,IWRITE
   50 FORMAT(/4X,'IHIS',3X,'IDIST',2X,'IWRITE'/3I8/)
C
      WRITE(6,60) SVSEED,RHSEED
   60 FORMAT(/' SEEDS FOR RANDOM NUMBER GENERATOR'//
     1 4X,'LANCZS SEED',4X,'INVERR SEED'/2I15/)
C
      WRITE(6,70) (NMEV(J), J=1,NMEVS)
   70 FORMAT(/' SIZES OF T-MATRICES TO BE CONSIDERED'/(6I12))
C
      WRITE(6,80) RELTOL,GAPTOL,BTOL
```

```
   80 FORMAT(/' RELATIVE TOLERANCE USED TO COMBINE COMPUTED T-EIGENVALUE
     1S'/E15.3/' RELATIVE GAP TOLERANCES USED IN INVERSE ITERATION'/
     1E15.3/' RELATIVE TOLERANCE FOR CHECK ON SIZE OF BETAS'/E15.3/)
C
      WRITE(6,90) (J,LB(J),UB(J), J=1,NINT)
   90 FORMAT(/' BISEC WILL BE USED ON THE FOLLOWING INTERVALS'/
     1 (I6,2E20.6))
C
      IF (ISTART.EQ.0) GO TO 140
C
C     READ IN ALPHA BETA HISTORY
C
      READ(2,100)MOLD,NOLD,SVSOLD,MATAO,MATBO
  100 FORMAT(2I6,I12,2I8)
C
      IF (KMAX.LT.MOLD) KMAX = MOLD
      KMAX1 = KMAX + 1
C
C     CHECK THAT ORDER N, MATRIX IDS (MATNOA AND MATNOB), AND RANDOM
C     SEED (SVSEED) AGREE WITH THOSE IN THE HISTORY FILE.  IF NOT
C     PROCEDURE STOPS.
C
      ITEMP = (NOLD-N)**2 + (MATNOA-MATAO)**2 + (SVSEED-SVSOLD)**2
     1 + (MATNOB-MATBO)**2
C
      IF (ITEMP.EQ.0) GO TO 120
C
      WRITE(6,110)
  110 FORMAT(' PROGRAM TERMINATES'/    ' READ FROM FILE 2 CORRESPONDS TO
     1 DIFFERENT MATRIX THAN MATRIX SPECIFIED'/)
      GO TO 640
C
  120 CONTINUE
      MOLD1 = MOLD+1
C
      READ(2,130)(ALPHA(J), J=1,MOLD)
      READ(2,130)(BETA(J), J=1,MOLD1)
  130 FORMAT(4Z20)
C
      IF (KMAX.EQ.MOLD) GO TO 160
C
C     SAVE V1 = B*V(KMAX), VS = B*V(KMAX+1), V2 = V(KMAX+1)
      READ(2,130) (V1(J), J=1,N)
      READ(2,130) (VS(J), J=1,N)
      READ(2,130) (V2(J), J=1,N)
C
  140 CONTINUE
      IIX = SVSEED
C
C-----------------------------------------------------------------------
C
      CALL LANCZS(LSOLV,AMATV,ALPHA,BETA,V1,V2,VS,G,KMAX,MOLD1,N,IIX)
C
C-----------------------------------------------------------------------
C
```

```
      KMAX1 = KMAX + 1
C
      IF (IHIS.EQ.0.AND.ISTOP.GT.0) GO TO 160
C
      WRITE(1,150) KMAX,N,SVSEED,MATNOA,MATNOB
  150 FORMAT(2I6,I12,2I8,' = KMAX,N,SVSEED,MATNOA,MATNOB')
C
      WRITE(1,130)(ALPHA(I), I=1,KMAX)
      WRITE(1,130)(BETA(I), I=1,KMAX1)
C
C     SAVE V1 = B*V(KMAX), VS = B*V(KMAX+1), V2 = V(KMAX+1)
      WRITE(1,130) (V1(I), I=1,N)
      WRITE(1,130) (VS(I), I=1,N)
      WRITE(1,130) (V2(I), I=1,N)
C
      IF (ISTOP.EQ.0) GO TO 540
C
  160 CONTINUE
      BKMIN = BTOL
      WRITE(6,170)
  170 FORMAT(/' T-MATRICES (ALPHA AND BETA) ARE NOW AVAILABLE'/)
C
C-----------------------------------------------------------------------
C     SUBROUTINE TNORM CHECKS MIN(BETA)/(ESTIMATED NORM(A)) > BTOL .
C     IF THIS IS VIOLATED IB IS SET EQUAL TO THE NEGATIVE OF THE INDEX
C     OF THE MINIMAL BETA.  IF(IB < 0) THEN SUBROUTINE TNORM IS
C     CALLED FOR EACH VALUE OF MEV TO DETERMINE WHETHER OR NOT THERE
C     IS A BETA IN THE T-MATRIX SPECIFIED THAT VIOLATES THIS TEST.
C     IF THERE IS SUCH A BETA THE PROGRAM TERMINATES FOR THE USER
C     TO DECIDE WHAT TO DO.  THIS TEST CAN BE OVER-RIDDEN BY
C     SIMPLY MAKING BTOL SMALLER, BUT THEN THERE IS THE POSSIBILITY
C     THAT LOSSES IN THE LOCAL ORTHOGONALITY MAY HURT THE COMPUTATIONS.
C     BTOL = 1.D-8 IS HOWEVER A CONSERVATIVE CHOICE FOR BTOL.
C
C     TNORM ALSO COMPUTES TKMAX = MAX(|ALPHA(K)|,BETA(K), K=1,KMAX).
C     TKMAX IS USED TO SCALE THE TOLERANCES USED IN THE
C     T-MULTIPLICITY AND SPURIOUS TESTS IN BISEC. TKMAX IS ALSO USED IN
C     THE PROJECTION TEST FOR HIDDEN EIGENVALUES THAT HAD 'TOO SMALL'
C     A PROJECTION ON THE STARTING VECTOR.
C
      CALL TNORM(ALPHA,BETA,BKMIN,TKMAX,KMAX,IB)
C
C-----------------------------------------------------------------------
C
      TTOL = EPSM*TKMAX
C
C     LOOP ON THE SIZE OF THE T-MATRIX
C
  180 CONTINUE
      MMB = MMB + 1
      MEV = NMEV(MMB)
C     IS MEV TOO LARGE ?
      IF(MEV.LE.KMAX) GO TO 200
      WRITE(6,190) MMB, MEV, KMAX
  190 FORMAT(/' TERMINATE PRIOR TO CONSIDERING THE',I6,'TH T-MATRIX'/
```

```
     1' BECAUSE THE SIZE REQUESTED',I6,' IS GREATER THAN THE MAXIMUM SIZ
     1E ALLOWED',I6/)
      GO TO 540
C
  200 MP1 = MEV + 1
      BETAM = BETA(MP1)
C
      IF (IB.GE.0) GO TO 210
C
      T0 = BTOL
C
C-----------------------------------------------------------------------
C
      CALL TNORM(ALPHA,BETA,T0,T1,MEV,IBMEV)
C
C-----------------------------------------------------------------------
C
      TEMP = T0/TKMAX
      IBMEV = IABS(IBMEV)
      IF (TEMP.GE.BTOL) GO TO 210
      IBMEV = -IBMEV
      GO TO 600
C
  210 CONTINUE
      IC = MXSTUR-ICT
C
C-----------------------------------------------------------------------
C     BISEC LOOP. THE SUBROUTINE BISEC INCORPORATES DIRECTLY THE
C     T-MULTIPLICITY AND SPURIOUS TESTS. T-EIGENVALUES WILL BE
C     CALCULATED BY BISEC SEQUENTIALLY ON INTERVALS
C     (LB(J),UB(J)), J = 1,NINT).
C
C     ON RETURN FROM BISEC
C     NDIS = NUMBER OF DISTINCT EIGENVALUES OF T(1,MEV) ON UNION
C            OF THE (LB,UB) INTERVALS
C     VS = DISTINCT T-EIGENVALUES IN ALGEBRAICALLY INCREASING ORDER
C     MP = MULTIPLICITIES OF THE T-EIGENVALUES IN VS
C     MP(I) = (0,1,MI), MI>1, I=1,NDIS  MEANS:
C        (0)  VS(I) IS SPURIOUS
C        (1)  VS(I) IS T-SIMPLE AND GOOD
C        (MI) VS(I) IS MULTIPLE AND IS THEREFORE NOT ONLY GOOD BUT
C             ALSO A CONVERGED GOOD T-EIGENVALUE.
C
C
      CALL BISEC(ALPHA,BETA,V1,V2,VS,LB,UB,EPSM,TTOL,MP,NINT,
     1 MEV,NDIS,IC,IWRITE)
C
C-----------------------------------------------------------------------
C
      IF (NDIS.EQ.0) GO TO 620
C
C     COMPUTE THE TOTAL NUMBER OF STURM SEQUENCES USED TO DATE
C     COMPUTE THE BISEC CONVERGENCE AND T-MULTIPLICITY TOLERANCES USED.
C     COMPUTE THE CONVERGENCE TOLERANCE FOR EIGENVALUES OF A.
      ICT = ICT + IC
```

```
      TEMP = DFLOAT(MEV+1000)
      MULTOL = TEMP*TTOL
      TEMP = DSQRT(TEMP)
      BISTOL = TTOL*TEMP
      CONTOL = BETAM*1.D-10
C
C-----------------------------------------------------------------------
C    SUBROUTINE LUMP 'COMBINES' T-EIGENVALUES THAT ARE 'TOO CLOSE'.
C    NOTE HOWEVER THAT CLOSE SPURIOUS T-EIGENVALUES ARE NOT AVERAGED
C    WITH GOOD ONES. HOWEVER, THEY MAY BE USED TO INCREASE THE
C    MULTIPLICITY OF A GOOD T-EIGENVALUE.
C
      LOOP = NDIS
      CALL LUMP(VS,RELTOL,MULTOL,SCALE2,MP,LOOP)
C
C-----------------------------------------------------------------------
C
      IF(NDIS.EQ.LOOP) GO TO 230
C
      WRITE(6,220) NDIS, MEV, LOOP
  220 FORMAT(/I6,' DISTINCT T-EIGENVALUES WERE COMPUTED IN BISEC AT MEV
     1',I6/ 2X,' LUMP SUBROUTINE REDUCES NUMBER OF DISTINCT EIGENVALUES
     1O',I6)
C
  230 CONTINUE
      NDIS = LOOP
      BETA(MP1) = BETAM
C
C-----------------------------------------------------------------------
C     THE SUBROUTINE ISOEV LABELS THOSE SIMPLE EIGENVALUES OF T(1,MEV)
C     WITH VERY SMALL GAPS BETWEEN NEIGHBORING EIGENVALUES OF T(1,MEV)
C     TO AVOID COMPUTING ERROR ESTIMATES FOR ANY SIMPLE GOOD
C     T-EIGENVALUE THAT IS TOO CLOSE TO A SPURIOUS EIGENVALUE.
C     ON RETURN FROM ISOEV, G CONTAINS CODED MINIMAL GAPS
C     BETWEEN THE DISTINCT EIGENVALUES OF T(1,MEV). (G IS REAL).
C     G(I) < 0 MEANS MINGAP IS DUE TO LEFT GAP G(I) > 0 MEANS DUE TO
C     RIGHT GAP. MP(I) = -1 MEANS THAT THE GOOD T-EIGENVALUE IS SIMPLE
C     AND HAS A VERY SMALL MINGAP IN T(1,MEV) DUE TO A SPURIOUS
C     EIGENVALUE.  NG = NUMBER OF GOOD T-EIGENVALUES.
C     NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES.
C
      CALL ISOEV(VS,GAPTOL,MULTOL,SCALE1,G,MP,NDIS,NG,NISO)
C
C-----------------------------------------------------------------------
C
      WRITE(6,240)NG,NISO,NDIS
  240 FORMAT(/I6,' GOOD T-EIGENVALUES HAVE BEEN COMPUTED'/
     1 I6,' OF THESE ARE T-ISOLATED'/
     2 I6,' = NUMBER OF DISTINCT T-EIGENVALUES COMPUTED'/)
C
C     DO WE WRITE DISTINCT EIGENVALUES OF T-MATRIX TO FILE 11?
      IF (IDIST.EQ.0) GO TO 280
C
      WRITE(11,250) NDIS,NISO,MEV,N,SVSEED,MATNOA,MATNOB
  250 FORMAT(/4I6,I12,I8,'=ND,NIS,MEV,N,SEED,MNA,MNB')
```

```
C
      WRITE(11,260) (MP(I),VS(I),G(I), I=1,NDIS)
  260 FORMAT(2(I3,E25.16,E12.3))
C
      WRITE(11,270) NDIS, (MP(I), I=1,NDIS)
  270 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/(20I4))
C
  280 CONTINUE
C
      IF (NISO.NE.0) GO TO 310
C
      WRITE(4,290) MEV
  290 FORMAT(/' AT MEV = ',I6,' THERE ARE NO ISOLATED T-EIGENVALUES'/
     1' SO NO ERROR ESTIMATES WERE COMPUTED/')
C
      WRITE(6,300)
  300 FORMAT(/' ALL COMPUTED GOOD T-EIGENVALUES ARE MULTIPLE'/
     1 ' THEREFORE ALL SUCH EIGENVALUES ARE ASSUMED TO HAVE CONVERGED')
C
      ICONV = 1
      GO TO 350
C
  310 CONTINUE
C
C-----------------------------------------------------------------------
C     SUBROUTINE INVERR COMPUTES ERROR ESTIMATES FOR ISOLATED GOOD
C     T-EIGENVALUES USING INVERSE ITERATION ON T(1,MEV). ON RETURN
C     G(J) = MINIMUM GAP IN T(1,MEV) FOR EACH VS(J), J=1,NDIS
C     G(MEV+I) = BETAM*|U(MEV)| = ERROR ESTIMATE FOR ISOLATED GOOD
C              T-EIGENVALUES, WHERE I = 1, NISO AND  BETAM = BETA(MEV+1)
C              U(MEV) IS MEVTH COMPONENT OF THE UNIT EIGENVECTOR OF T
C              CORRESPONDING TO THE ITH ISOLATED GOOD T-EIGENVALUE.
C     A NEGATIVE ERROR ESTIMATE MEANS THAT FOR THAT PARTICULAR
C     EIGENVALUE THE INVERSE ITERATION DID NOT CONVERGE IN <= MXINIT
C     STEPS AND THAT THE CORRESPONDING ERROR ESTIMATE IS QUESTIONABLE.
C
C     V2 CONTAINS THE ISOLATED GOOD T-EIGENVALUES
C     V1 CONTAINS THE MINGAPS TO THE NEAREST DISTINCT  EIGENVALUE
C        OF T(1,MEV) FOR EACH ISOLATED GOOD T-EIGENVALUE IN V2.
C     VS CONTAINS THE NDIS DISTINCT EIGENVALUES OF T(1,MEV)
C     MP CONTAINS THE CORRESPONDING CODED T-MULTIPLICITIES
C
      IT = MXINIT
      CALL INVERR(ALPHA,BETA,V1,V2,VS,EPSM,G,MP,MEV,MMB,NDIS,NISO,N,
     1 RHSEED,IT,IWRITE)
C
C-----------------------------------------------------------------------
C
C     SIMPLE CHECK FOR CONVERGENCE. CHECKS TO SEE IF ALL OF THE ERROR
C     ESTIMATES ARE SMALLER THAN CONTOL = BETAM*1.D-10.
C     IF THIS TEST IS SATISFIED, THEN CONVERGENCE FLAG, ICONV IS SET
C     TO 1.  TYPICALLY ERROR ESTIMATES ARE VERY CONSERVATIVE.
C
      WRITE(6,320) CONTOL
  320 FORMAT(/' CONVERGENCE IS TESTED USING THE CONVERGENCE TOLERANCE',
```

```
     1E13.4/)                                                           LGV04860
C                                                                       LGV04870
      II = MEV +1                                                       LGV04880
      IF = MEV+NISO                                                     LGV04890
      DO 330 I = II,IF                                                  LGV04900
      IF (ABS(G(I)).GT.CONTOL) GO TO 350                                LGV04910
  330 CONTINUE                                                          LGV04920
      ICONV = 1                                                         LGV04930
      MMB = NMEVS                                                       LGV04940
C                                                                       LGV04950
      WRITE(6,340) CONTOL                                               LGV04960
  340 FORMAT(' ALL COMPUTED ERROR ESTIMATES WERE LESS THAN',E15.4/      LGV04970
     1 ' THEREFORE PROCEDURE TERMINATES'/)                              LGV04980
C                                                                       LGV04990
  350 CONTINUE                                                          LGV05000
C                                                                       LGV05010
C     IF CONVERGENCE IS INDICATED, THAT IS ICONV = 1 ,THEN              LGV05020
C     THE SUBROUTINE PRTEST IS CALLED TO CHECK FOR ANY CONVERGED        LGV05030
C     T-EIGENVALUES THAT HAVE BEEN MISLABELLED AS SPURIOUS BECAUSE      LGV05040
C     THE PROJECTION OF THEIR EIGENVECTOR(S) ON THE STARTING            LGV05050
C     VECTOR WERE TOO SMALL.                                            LGV05060
C     NUMERICAL TESTS INDICATE THAT SUCH EIGENVALUES ARE RARE.          LGV05070
C     IF FOR SOME REASON MANY OF THESE HIDDEN EIGENVALUES APPEAR        LGV05080
C     ON SOME RUN, YOU CAN BE CERTAIN THAT SOMETHING IS FOULED UP.      LGV05090
C                                                                       LGV05100
      IF (ICONV.EQ.0) GO TO 480                                         LGV05110
C                                                                       LGV05120
C-----------------------------------------------------------------------LGV05130
C                                                                       LGV05140
      CALL PRTEST (ALPHA,BETA,VS,TKMAX,EPSM,RELTOL,SCALE3,SCALE4,       LGV05150
     1 MP,NDIS,MEV,IPROJ)                                               LGV05160
C                                                                       LGV05170
C-----------------------------------------------------------------------LGV05180
C                                                                       LGV05190
      IF(IPROJ.EQ.0) GO TO 470                                          LGV05200
C                                                                       LGV05210
      IF(IDIST.EQ.1)  WRITE(11,360) IPROJ                               LGV05220
  360 FORMAT(' SUBROUTINE PRTEST WANTS TO RELABEL',I6,' SPURIOUS T-EIGENLGV05230
     1VALUES'/' WE ACCEPT RELABELLING ONLY IF LAST COMPONENT OF T-EIGENVLGV05240
     1ECTOR IS L.T. 1.D-10')                                            LGV05250
C                                                                       LGV05260
      IIX = RHSEED                                                      LGV05270
C                                                                       LGV05280
C-----------------------------------------------------------------------LGV05290
C                                                                       LGV05300
      CALL GENRAN(IIX,G,MEV)                                            LGV05310
C                                                                       LGV05320
C-----------------------------------------------------------------------LGV05330
C                                                                       LGV05340
      ITEN = -10                                                        LGV05350
      NISOM = NISO + MEV                                                LGV05360
      IWRITO = IWRITE                                                   LGV05370
      IWRITE = 0                                                        LGV05380
C                                                                       LGV05390
      DO 390 J = 1,NDIS                                                 LGV05400
```

```
      IF(MP(J).NE.ITEN) GO TO 390
      T0 = VS(J)
C
C-----------------------------------------------------------------------
C
      IT = MXINIT
      CALL INVERM(ALPHA,BETA,V1,V2,T0,TEMP,T1,EPSM,G,MEV,IT,IWRITE)
C
C-----------------------------------------------------------------------
C
      IF(TEMP.LE.1.D-10) GO TO 380
C     ERROR ESTIMATE WAS NOT SMALL REJECT RELABELLING OF THIS EIGENVALUE
      IF(IDIST.EQ.1)  WRITE(11,370) J,T0,TEMP
  370 FORMAT(/' LAST COMPONENT FOR',I6,'TH EIGENVALUE',E20.12/' IS TOO L
     1ARGE = ',E15.6,' SO DO NOT ACCEPT PRTEST RELABELLING'/)
      MP(J) = 0
      IPROJ = IPROJ - 1
      GO TO 390
C     RELABELLING ACCEPTED
  380 NISOM = NISOM + 1
      G(NISOM) = BETAM*TEMP
  390 CONTINUE
      IWRITE = IWRITO
C
      IF(IPROJ.EQ.0) GO TO 430
      WRITE(6,400) IPROJ
  400 FORMAT(/I6,' T-EIGENVALUES WERE RECLASSIFIED AS GOOD.'/
     1' THESE ARE IDENTIFIED IN FILE 3 BY A T-MULTIPLICITY OF -10'/' USE
     2R SHOULD INSPECT EACH TO MAKE SURE NEIGHBORS HAVE CONVERGED'/)
C
      IF(IDIST.EQ.1)  WRITE(11,410) IPROJ
  410 FORMAT(/I6,' T-EIGENVALUES WERE RELABELLED AS GOOD'/
     1' BELOW IS CORRECTED T-MULTIPLICITY PATTERN'/)
C
      WRITE(6,420) NDIS, (MP(I), I=1,NDIS)
      IF(IDIST.EQ.1)  WRITE(11,420) NDIS, (MP(I), I=1,NDIS)
  420 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/
     1 6X, ' (-10) MEANS SPURIOUS T-EIGENVALUE RELABELLED AS GOOD'/(20I4
     1))
C
C     RECALCULATE MINGAPS FOR DISTINCT T(1,MEV) EIGENVALUES.
  430 NM1 = NDIS - 1
      G(NDIS) = VS(NM1)-VS(NDIS)
      G(1) = VS(2)-VS(1)
C
      DO 440 J = 2,NM1
      T0 = VS(J)-VS(J-1)
      T1 = VS(J+1)-VS(J)
      G(J) = T1
      IF (T0.LT.T1) G(J) = -T0
  440 CONTINUE
      IF(IPROJ.EQ.0) GO TO 470
C     WRITE TO FILE 4 ERROR ESTIMATES FOR THOSE T-EIGENVALUES RELABELLED
      NGOOD = 0
      DO 450 J = 1,NDIS
```

```
      IF(MP(J).EQ.0) GO TO 450
      NGOOD = NGOOD + 1
      IF(MP(J).NE.ITEN) GO TO 450
      T0 = VS(J)
      NISO = NISO + 1
      NISOM = MEV + NISO
      WRITE(4,460) NGOOD,T0,G(NISOM),G(J)
  450 CONTINUE
  460 FORMAT(I10,E25.16,2E14.3)
C
  470 CONTINUE
C
C     WRITE THE GOOD T-EIGENVALUES TO FILE 3.  FIRST TRANSFER THEM
C     TO V2 AND THEIR T-MULTIPLICITIES TO THE CORRESPONDING POSITIONS
C     IN MP AND COMPUTE THE AB-MINGAPS, THE MINIMAL GAPS BETWEEN THE
C     GOOD T-EIGENVALUES.  THESE GAPS WILL BE PUT IN THE ARRAY G.
C     SINCE G CURRENTLY CONTAINS THE MINIMAL GAPS BETWEEN THE DISTINCT
C     EIGENVALUES OF THE T-MATRIX, THESE GAPS WILL FIRST BE
C     TRANSFERRED TO V1.  NOTE THAT V1<0 MEANS THAT THAT MINIMAL GAP
C     IN THE T-MATRIX IS DUE TO A SPURIOUS T-EIGENVALUE.
C     ALL THIS INFORMATION IS PRINTED TO FILE 3
C
  480 CONTINUE
C
      NG = 0
      DO 490 I = 1,NDIS
      IF (MP(I).EQ.0) GO TO 490
      NG = NG+1
      MP(NG) = MP(I)
      V2(NG) = VS(I)
      TEMP = G(I)
      TEMP = DABS(TEMP)
      J = I+1
      IF (G(I).LT.ZERO) J = I-1
      IF (MP(J).EQ.0) TEMP = -TEMP
      V1(NG) = TEMP
  490 CONTINUE
C
      WRITE(6,500)MEV
  500 FORMAT(//' T-EIGENVALUE CALCULATION AT MEV = ',I6,'    IS COMPLETE
     1')
C
C     NG = NUMBER OF COMPUTED DISTINCT GOOD T-EIGENVALUES.  NEXT
C     GENERATE GAPS BETWEEN GOOD T-EIGENVALUES (ABMINGAPS) AND PUT THEM
C     IN G.  G(J) < 0 MEANS THE ABMINGAP IS DUE TO THE LEFT-HAND GAP.
C
      NGM1 = NG - 1
      G(NG) = V2(NGM1)-V2(NG)
      G(1) = V2(2)-V2(1)
C
      DO 510 J = 2,NGM1
      T0 = V2(J)-V2(J-1)
      T1 = V2(J+1)-V2(J)
      G(J) = T1
      IF (T0.LT.T1) G(J) = -T0
```

```
  510 CONTINUE
C
C     WRITE GOOD T-EIGENVALUES OUT TO FILE 3.
C
      WRITE(3,520)NG,NDIS,MEV,N,SVSEED,MATNOA,MATNOB,MULTOL,IB,BTOL
  520 FORMAT(4I6,I12,2I8,'=NG,ND,MEV,N,SEED,MNA,MNB'/
     1 E20.12,I6,E13.4,' = MUTOL,INDEX MINIMAL BETA,BTOL'/
     1' EV NO',1X,'TMULT',10X,'GOOD EIGENVALUE',7X,'TMINGAP',6X,'ABMINGA
     1P')
C
      WRITE(3,530)(I,MP(I),V2(I),V1(I),G(I), I=1,NG)
  530 FORMAT(2I6,E25.16,2E14.3)
C
C     IF CONVERGENCE FLAG ICONV.NE.1 AND NUMBER OF T-MATRICES
C     CONSIDERED TO DATE IS LESS THAN NUMBER ALLOWED, INCREMENT MEV.
C     AND LOOP BACK TO 210 TO REPEAT COMPUTATIONS.  RESTORE BETA(MEV+1).
C
      BETA(MP1) = BETAM
C
      IF (MMB.LT.NMEVS.AND.ICONV.NE.1) GO TO 180
C
C     END OF LOOP ON DIFFERENT SIZE T-MATRICES ALLOWED.
C
  540 CONTINUE
C
      IF(ISTOP.EQ.0)  WRITE(6,550)
  550 FORMAT(/' T-MATRICES (ALPHA AND BETA) ARE NOW AVAILABLE, TERMINATE
     1')
      IF (IHIS.EQ.1.AND.KMAX.NE.MOLD) WRITE(1,560)
  560 FORMAT(/' ABOVE ARE THE FOLLOWING VECTORS '/
     1 '  ALPHA(I), I = 1,KMAX'/
     2 '  BETA(I), I = 1,KMAX+1'/
     3 ' FINAL THREE VECTORS USED IN LANCZS SUBROUTINE'/
     3 ' V1 = B*V(KMAX), VS = B*V(KMAX+1), V2 = V(KMAX+1)'/
     4 ' ALL VECTORS IN THIS FILE HAVE HEX FORMAT 4Z20'/
     5 ' ----- END OF FILE 1 NEW ALPHA, BETA HISTORY---------------'///)
C
      IF (ISTOP.EQ.0) GO TO 640
C
      WRITE(3,570)
  570 FORMAT(/' ABOVE ARE COMPUTED GOOD T-EIGENVALUES'/
     1 ' NG = NUMBER OF GOOD T-EIGENVALUES COMPUTED'/
     2 ' ND = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)'/
     3 ' N = ORDER OF A AND B-MATRIX,  MNA, MNB = MATRIX IDENTS'/
     4 ' MULTOL = T-MULTIPLICITY TOLERANCE FOR T-EIGENVALUES IN BISEC'/
     4 ' TMULT IS THE T-MULTIPLICITY OF GOOD T-EIGENVALUE'/
     5 ' TMULT = -1 MEANS SPURIOUS T-EIGENVALUE TOO CLOSE'/
     6 ' DO NOT COMPUTE ERROR ESTIMATES FOR SUCH EIGENVALUES'/
     7 ' ABMINGAP = MINIMAL GAP BETWEEN THE COMPUTED EIGENVALUES'/
     8 ' ABMINGAP .LT. 0. MEANS MINIMAL GAP IS DUE TO LEFT-HAND GAP'/
     9 ' TMINGAP= MINIMAL GAP W.R.T.  DISTINCT EIGENVALUES IN T(1,MEV)'/
     1 ' TMINGAP .LT. 0. MEANS MINGAP IS DUE TO SPURIOUS T-EIGENVALUE'/
     2 ' ----- END OF FILE 3 GOODEIGENVALUES-----------------------'///)
C
      IF (IDIST.EQ.1) WRITE(11,580)
```

```
  580 FORMAT(/' ABOVE ARE THE DISTINCT EIGENVALUES OF T(1,MEV).'/       LGV07060
     2 ' THE FORMAT IS      T-MULTIPLICITY   T-EIGENVALUE   TMINGAP'/  LGV07070
     3 '       THIS FORMAT IS REPEATED TWICE ON EACH LINE.'/            LGV07080
     4 ' T-MULTIPLICITY = -1 MEANS THAT THE SUBROUTINE ISOEV HAS TAGGED'LGV07090
     5/'   THIS SIMPLE T-EIGENVALUE AS HAVING A VERY CLOSE SPURIOUS'/   LGV07100
     6 '   T-EIGENVALUE SO THAT NO ERROR ESTIMATE WILL BE COMPUTED'/    LGV07110
     7 '   FOR THAT EIGENVALUE IN SUBROUTINE INVERR.'/                  LGV07120
     8 ' TMINGAP .LT. 0, TMINGAP IS DUE TO LEFT GAP .GT. 0, RIGHT GAP.'/LGV07130
     9 ' EACH OF THE DISTINCT T-EIGENVALUE TABLES IS FOLLOWED'/         LGV07140
     9 ' BY THE T-MULTIPLICITY PATTERN.'/                               LGV07150
     1 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV).'/  LGV07160
     2 ' NG = NUMBER OF GOOD T-EIGENVALUES. '/                          LGV07170
     3 ' NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES. '/               LGV07180
     4 ' NISO ALSO IS THE COUNT OF +1 ENTRIES IN T-MULTIPLICITY PATTERN.LGV07190
     5 '/' ----- END OF FILE 11 DISTINCT T-EIGENVALUES--------------'///LGV07200
     6 )                                                                LGV07210
C                                                                       LGV07220
      IF(NISO.NE.0)  WRITE(4,590)                                       LGV07230
  590 FORMAT(/' ABOVE ARE THE ERROR ESTIMATES OBTAINED FOR THE ISOLATED LGV07240
     1GOOD T-EIGENVALUES'/                                              LGV07250
     1' OBTAINED VIA INVERSE ITERATION IN THE SUBROUTINE INVERR.'/      LGV07260
     1' ALL OTHER GOOD T-EIGENVALUES HAVE CONVERGED.'/                  LGV07270
     2' ERROR ESTIMATE = BETAM*ABS(UM)'/                                LGV07280
     2' WHERE BETAM = BETA(MEV+1) AND UM = U(MEV).'/                    LGV07290
     3' U = UNIT EIGENVECTOR OF T WHERE T*U = EV*U AND EV = ISOLATED GOOLGV07300
     3D T-EIGENVALUE.'/                                                 LGV07310
     4' TMINGAP = GAP TO NEAREST DISTINCT EIGENVALUE OF T(1,MEV).'/     LGV07320
     5' TMINGAP .LT. 0. MEANS MINGAP IS DUE TO A LEFT NEIGHBOR.'/       LGV07330
     6' ERROR ESTIMATE L.T. 0 MEANS INVERSE ITERATION DID NOT CONVERGE'/LGV07340
     7' ------ END OF FILE 4 ERRINV ------------------------------'//) LGV07350
      GO TO 640                                                         LGV07360
C                                                                       LGV07370
  600 CONTINUE                                                          LGV07380
C                                                                       LGV07390
      IBB = IABS(IBMEV)                                                 LGV07400
      IF (IBMEV.LT.0) WRITE(6,610) MEV,IBB,BETA(IBB)                    LGV07410
  610 FORMAT(/' PROGRAM TERMINATES BECAUSE MEV REQUESTED = ',I6,' IS .GTLGV07420
     1',I6/' AT WHICH AN ABNORMALLY SMALL BETA = ' , E13.4,' OCCURRED'/)LGV07430
      GO TO 640                                                         LGV07440
C                                                                       LGV07450
  620 IF (NDIS.EQ.0.AND.ISTOP.GT.0) WRITE(6,630)                        LGV07460
  630 FORMAT(/' INTERVALS SPECIFIED FOR BISECT DID NOT CONTAIN ANY T-EIGLGV07470
     1ENVALUES'/' PROGRAM TERMINATES')                                  LGV07480
C                                                                       LGV07490
  640 CONTINUE                                                          LGV07500
C                                                                       LGV07510
      STOP                                                              LGV07520
C-----END OF MAIN PROGRAM FOR LANCZOS EIGENVALUE COMPUTATIONS-----------LGV07530
      END                                                               LGV07540
```

5.3 LGVEC: Main Program, Eigenvector Computations

```
C-----LGVEC (EIGENVECTORS OF A*X = EVAL*B*X)---------------------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C        REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4
C        Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C        Applied Mathematics, 2002. SIAM Publications,
C        Philadelphia, PA. USA
C
C
C     CONTAINS MAIN PROGRAM FOR COMPUTING AN EIGENVECTOR CORRESPONDING
C     TO EACH OF A SET OF EIGENVALUES WHICH HAVE BEEN COMPUTED
C     ACCURATELY BY THE CORRESPONDING LANCZOS EIGENVALUE PROGRAM
C     (LGVAL) FOR THE SYMMETRIC, GENERALIZED PROBLEM A*X = EVAL*B*X.
C     LGVAL AND LGVEC ASSUME THAT B IS POSITIVE DEFINITE AND THAT THE
C     CHOLESKY FACTORS OF B (OR OF A PERMUTATION OF B) ARE AVAILABLE
C     FOR USE IN THE LANCZOS PROCEDURES.  IF B HAS BEEN PERMUTED,
C     THEN THESE PROCEDURES ASSUME THAT THE DATA PRESENTED FOR THE
C     A-MATRIX HAS BEEN SUBJECTED TO THE SAME PERMUTATION.  THAT
C     PERMUTATION WILL THEN BE USED AFTER THE RITZ VECTORS FOR THE
C     PERMUTED VERSION OF THE ORIGINAL PROBLEM HAVE BEEN COMPUTED,
C     TO OBTAIN THE ASSOCIATED RITZ VECTORS FOR THE ORIGINAL PROBLEM.
C     NOTE THAT THIS PROGRAM COULD BE MODIFIED TO COMPUTE ADDITIONAL
C     EIGENVECTORS FOR ANY COMPUTED EIGENVALUE WHICH IS A MULTIPLE
C     EIGENVALUE OF THE GIVEN A-MATRIX.  THE AMOUNT OF ADDITIONAL
C     COMPUTATION REQUIRED WOULD DEPEND UPON THE PARTICULAR
C     A-MATRIX AND B-MATRIX USED AND UPON WHAT PART OF THE
C     SPECTRUM OF EIGENVALUES IS BEING CONSIDERED.
C
C     THE LANCZOS EIGENVECTOR COMPUTATIONS ASSUME THAT EACH
C     EIGENVALUE THAT IS BEING CONSIDERED HAS CONVERGED AS AN
C     EIGENVALUE OF THE ASSOCIATED LANCZOS TRIDIAGONAL MATRICES.
C
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE
C     CONSTRUCTIONS
```

```
C
C     1.  DATA/MACHEP/ STATEMENT
C     2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)
C     3.  FORMAT(20A4) USED WITH THE EXPLANATORY HEADER, EXPLAN
C     4.  HEXADECIMAL FORMAT (4Z20) USED IN ALPHA/BETA FILES 1 AND 2.
C
C     IMPORTANT NOTE:  PROGRAM ALLOWS ENLARGEMENT OF THE ALPHA, BETA
C     ARRAYS.  IN PARTICULAR, IF ANY ONE OF THE EIGENVALUES SUPPLIED
C     IS T-SIMPLE AND NOT CLOSE TO A SPURIOUS EIGENVALUE, THE PROGRAM
C     REQUIRES THAT KMAX BE AT LEAST 11*MEV/8 + 12.  IF KMAX IS NOT
C     THIS LARGE, THEN THE PROGRAM RESETS KMAX TO THIS SIZE
C     AND EXTENDS THE ALPHA, BETA HISTORY IF REQUIRED.
C     THUS, THE DIMENSIONS OF THE ALPHA AND BETA ARRAYS MUST BE
C     LARGE ENOUGH TO ALLOW FOR THIS POSSIBILITY.
C     REMEMBER THAT THE BETA ARRAY, BETA(J), IS SUCH THAT
C     J = 1,..., KMAX+1.  SO IF THE KMAX USED BY THE PROGRAM
C     IS TO BE 3000, THEN BETA MUST BE OF LENGTH AT LEAST 3001.
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  ALPHA(5000),BETA(5001)
      DOUBLE PRECISION  V1(5000),V2(5000),VS(5000)
      DOUBLE PRECISION  RITVEC(30000),TVEC(30000),GOODEV(50),EVNEW(50)
      DOUBLE PRECISION  EVAL,EVALN,TOLN,TTOL,ERTOL,ALFA,BATA
      DOUBLE PRECISION  MULTOL,SCALE0,STUTOL,BTOL,LB,UB
      DOUBLE PRECISION  ONE,ZERO,MACHEP,EPSM,TEMP,SUM,ERRMIN,BKMIN
      DOUBLE PRECISION  RELTOL,ERROR,TERROR,TLAST(50)
      REAL  G(10000),AMINGP(50),TMINGP(50),EXPLAN(20)
      REAL  TERR(50),ERR(50),ERRDGP(50),RNORM(50),TBETA(50)
      INTEGER  MP(50),M1(50),M2(50),MA(50),ML(50),MINT(50),MFIN(50)
      INTEGER  SVSEED,SVSOLD,RHSEED,IDELTA(50)
      INTEGER  MBOUND,NTVCON,SVTVEC,TVSTOP,LVCONT,ERCONT,TFLAG
      DOUBLE PRECISION  FINPRO
      DOUBLE PRECISION  DABS, DMAX1, DSQRT, DFLOAT
      REAL ABS
      INTEGER  IABS
      EXTERNAL LSOLV, AMATV
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.D0*MACHEP
C-----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     1.  ALPHA:  >= KMAXN,  BETA: >= (KMAXN+1) WHERE KMAXN, THE
C               LARGEST SIZE T-MATRIX CONSIDERED BY THE PROGRAM,
C               IS THE LARGER OF THE SIZE OF THE ALPHA, BETA HISTORY
C               PROVIDED ON FILE 2 (IF ANY ) AND THE SIZE WHICH THE
C               PROGRAM SPECIFIES INTERNALLY, THIS LATTER IS ALWAYS
C               < = 11*MEV / 8  +  12, WHERE MEV IS THE SIZE
C               T-MATRIX THAT WAS USED IN THE CORRESPONDING EIGENVALUE
C               COMPUTATIONS.
C     2.  V1:  >= MAX(N,KMAX)
C     3.  V2,VS:   >= N
C     4.  G:  >= MAX(N,KMAX)
C     5.  RITVEC:  >= N*NGOOD, WHERE NGOOD IS NUMBER OF EIGENVALUES
C                 SUPPLIED TO THIS PROGRAM.
```

```
C     6.  TVEC:  >= CUMULATIVE LENGTH OF ALL THE T-EIGENVECTORS
C         NEEDED TO GENERATE THE DESIRED RITZ VECTORS.  AN EDUCATED
C         GUESS AT AN APPROPRIATE LENGTH CAN BE OBTAINED BY RUNNING THE
C         PROGRAM WITH THE FLAG MBOUND = 1 AND MULTIPLYING THE
C         RESULTING SIZE BY 5/4.
C     7.  GOODEV, AMINGP, TMINGP, TERR, ERR, ERRGDP, RNORM, TBETA,
C         TLAST, EVNEW, MP, MA, M1, M2, MINT, MFIN AND IDELTA ALL MUST
C         BE >= NGOOD.
C
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LANCZOS EIGENVECTOR PROCEDURE FOR REAL SYMMETRIC MATRICE
     1S'/)
C
C     SET PROGRAM PARAMETERS
C     USER MUST NOT MODIFY SCALE0
      SCALE0 = 5.0D0
      ZERO = 0.0D0
      ONE = 1.0D0
      MPMIN = -1000
C     SET CONVERGENCE CRITERION FOR T-EIGENVECTORS.
      ERTOL = 1.D-10
C
C     READ USER-SPECIFIED PARAMETER FROM INPUT FILE 5 (FREE FORMAT)
C
C     READ USER-PROVIDED HEADER FOR RUN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C     READ IN THE MAXIMUM PERMISSIBLE DIMENSIONS FOR THE TVEC ARRAY
C     (MDIMTV), FOR THE RITVEC ARRAY (MDIMRV), AND FOR THE BETA
C     ARRAY (MBETA).
      READ(5,20) EXPLAN
      READ(5,*) MDIMTV, MDIMRV, MBETA
C
C     READ IN RELATIVE TOLERANCE (RELTOL) USED IN DETERMINING
C     APPROPRIATE SIZES FOR THE T-MATRICES TO BE USED IN THE RITZ
C     VECTOR COMPUTATIONS.
      READ(5,20) EXPLAN
      READ(5,*) RELTOL
C
C     SET FLAGS TO 0 OR 1:
C     MBOUND = 1:  PROGRAM TERMINATES AFTER COMPUTING 1ST GUESSES
C                  ON APPROPRIATE T-SIZES FOR USE IN THE RITZ VECTOR
C                  COMPUTATIONS
C     NTVCON = 0:  PROGRAM TERMINATES IF THE TVEC ARRAY IS NOT
C                  LARGE ENOUGH TO HOLD ALL THE T-EIGENVECTORS REQUIRED.
C     SVTVEC = 0:  THE T-EIGENVECTORS ARE NOT WRITTEN TO FILE 11
C                  UNLESS TVSTOP = 1
C     SVTVEC = 1:  WRITE THE T-EIGENVECTORS TO FILE 11.
C     TVSTOP = 1:  PROGRAM TERMINATES AFTER COMPUTING THE
C                  T-EIGENVECTORS
C     LVCONT = 0:  PROGRAM TERMINATES IF THE NUMBER OF T-EIGENVECTORS
```

```
C               COMPUTED IS NOT EQUAL TO THE NUMBER OF RITZ
C               VECTORS REQUESTED.
C     ERCONT = 0:  MEANS FOR ANY GIVEN EIGENVALUE, A RITZ VECTOR
C               WILL NOT BE COMPUTED FOR THAT EIGENVALUE UNLESS
C               A T-EIGENVECTOR HAS BEEN IDENTIFIED WITH A LAST
C               COMPONENT WHICH SATISFIES THE SPECIFIED
C               CONVERGENCE CRITERION.
C     ERCONT = 1:  MEANS FOR ANY GIVEN EIGENVALUE, A RITZ VECTOR
C               WILL BE COMPUTED.  IF A T-EIGENVECTOR CANNOT
C               BE IDENTIFIED WHICH SATISFIES THE LAST
C               COMPONENT CRITERION, THEN THE PROGRAM WILL
C               USE THE T-VECTOR THAT CAME CLOSEST TO
C               SATISFYING THE CRITERION
C     IWRITE = 1:  EXTENDED OUTPUT OF INTERMEDIATE COMPUTATIONS
C               IS WRITTEN TO FILE 6
C     IREAD = 0:   ALPHA/BETA FILE IS REGENERATED.
C     IREAD = 1:   ALPHA/BETA FILE USED IN EIGENVALUE COMPUTATIONS
C               IS READ IN AND EXTENDED IF NECESSARY.  IN BOTH
C               CASES IREAD = 0 OR 1, THE LANCZOS VECTORS ARE
C               ALWAYS REGENERATED FOR THE RITZ VECTOR
C               COMPUTATIONS
C
      READ(5,20) EXPLAN
      READ(5,*) MBOUND,NTVCON,SVTVEC,IREAD
C
      READ(5,20) EXPLAN
      READ(5,*) TVSTOP,LVCONT,ERCONT,IWRITE
      IF (TVSTOP.EQ.1) SVTVEC = 1
C
C     READ IN SEED (RHSEED) FOR GENERATING RANDOM STARTING VECTOR
C     FOR INVERSE ITERATION ON THE T-MATRICES.
      READ(5,20) EXPLAN
      READ(5,*) RHSEED
C
C     READ IN MATNOA, MATNOB = MATRIX/RUN IDENTIFICATION NUMBERS,
C     N = ORDER OF A-MATRIX AND B-MATRIX AND FLAG, JPERM.
C     JPERM = (0,1):  1 MEANS PERMUTED A AND B ARE BEING USED, 0
C     MEANS A AND B HAVE NOT BEEN PERMUTED.
      READ(5,20) EXPLAN
      READ(5,*) MATNOA,MATNOB,N,JPERM
C
C-----------------------------------------------------------------------
C     INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED MATRICES
C     AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE
C     MATRIX-VECTOR MULTIPLY SUBROUTINE AMATV AND THE SOLVE
C     SUBROUTINE LSOLV.
      CALL USPECA(N,MATNOA)
      CALL USPECB(N,MATNOB)
C
C-----------------------------------------------------------------------
C
C     MASK UNDERFLOW AND OVERFLOW
      CALL MASK
C
C-----------------------------------------------------------------------
```

```
C     WRITE RUN PARAMETERS OUT TO FILE 6
C
      WRITE(6,30) MATNOA,MATNOB,N,JPERM
   30 FORMAT(/4X,'A-MATRIX ID',4X,'B-MATRIX ID',4X,'SIZES OF MATRICES',
     14X,'JPERM'/I15,I15,I21,I9)
C
      WRITE(6,40) MBOUND,NTVCON,SVTVEC,IREAD
   40 FORMAT(/3X,'MBOUND',3X,'NTVCON',3X,'SVTVEC',3X,'IREAD'/3I9,I8)
C
      WRITE(6,50) TVSTOP,LVCONT,ERCONT,IWRITE
   50 FORMAT(/3X,'TVSTOP',3X,'LVCONT',3X,'ERCONT',3X,'IWRITE'/4I9)
C
      WRITE(6,60) MDIMTV,MDIMRV,MBETA
   60 FORMAT(/3X,'MDIMTV',3X,'MDIMRV',3X,'MBETA'/2I9,I8)
C
      WRITE(6,70) RELTOL,RHSEED
   70 FORMAT(/7X,'RELTOL',3X,'RHSEED'/E13.4,I9)
C
C
C     FROM FILE 3 READ IN THE NUMBER OF EIGENVALUES (NGOOD) FOR WHICH
C     EIGENVECTORS ARE REQUESTED, THE ORDER (MEV) OF THE LANCZOS
C     TRIDIAGONAL MATRIX USED IN COMPUTING THESE EIGENVALUES, THE
C     ORDER (NOLD) OF THE USER-SPECIFIED MATRIX USED IN THE EIGENVALUE
C     COMPUTATIONS, THE SEED (SVSEED) USED FOR GENERATING THE STARTING
C     VECTOR THAT WAS USED IN THOSE LANCZOS EIGENVALUE COMPUTATIONS,
C     AND THE MATRIX/RUN IDENTIFICATION NUMBERS (MATA, MATB) USED IN
C     THOSE COMPUTATIONS.  ALSO READ IN THE NUMBER (NDIS) OF DISTINCT
C     EIGENVALUES OF T(1,MEV) THAT WERE COMPUTED BUT THIS VALUE IS
C     NOT USED IN THE EIGENVECTOR COMPUTATIONS.
C
      READ(3,80) NGOOD,NDIS,MEV,NOLD,SVSEED,MATA,MATB
   80 FORMAT(4I6,I12,2I8)
C
C     READ IN THE T-MULTIPLICITY TOLERANCE USED IN THE BISEC SUBROUTINE
C     DURING THE COMPUTATION OF THE GIVEN EIGENVALUES.
C     ALSO READ IN THE FLAG IB.  IF IB < 0, THEN SOME BETA(I) IN THE
C     T-MATRIX FILE PROVIDED ON FILE 2 FAILED THE ORTHOGONALITY
C     TEST IN THE TNORM SUBROUTINE.  USER SHOULD NOTE THAT THIS VECTOR
C     PROGRAM PROCEEDS INDEPENDENTLY OF THE SIZE OF THE BETA USED.
C
      READ(3,90) MULTOL,IB,BTOL
   90 FORMAT(E20.12,I6,E13.4)
C
      TEMP = DFLOAT(MEV+1000)
      TTOL = MULTOL/TEMP
      WRITE(6,100) MULTOL,TTOL
  100 FORMAT(/' T-MULTIPLICITY TOLERANCE USED IN THE EIGENVALUE COMPUTAT
     1IONS WAS',E13.4/' SCALED MACHINE EPSILON IS',E13.4)
C
C     CONTINUE WRITE TO FILE 6 OF THE PARAMETERS FOR THIS RUN
C
      WRITE(6,110)NGOOD,NDIS,MEV,NOLD,MATA,MATB,SVSEED,MULTOL,IB,BTOL
  110 FORMAT(/' EIGENVALUES SUPPLIED ARE READ IN FROM FILE 3'/' FILE 3
     1HEADER IS'/4X,'NG',2X,'NDIS',3X,'MEV',2X,'NOLD',2X,'MATNOA',2X,
     1'MATNOB'/(4I6,2I8)/4X,'SVSEED',6X,'MULTOL',6X,'IB',9X,'BTOL'/
```

```
     1I12,E12.3,I8,E13.4/)
C
C     IS THE ARRAY RITVEC LONG ENOUGH TO HOLD ALL OF THE DESIRED
C     RITZ VECTORS (APPROXIMATE EIGENVECTORS)?
      NMAX = NGOOD*N
      IF(MBOUND.NE.0) GO TO 120
      IF(TVSTOP.NE.1.AND.NMAX.GT.MDIMRV) GO TO 1350
C
C     CHECK THAT THE ORDER N AND THE MATRIX IDENTIFICATION NUMBERS
C     MATNOA AND MATNOB SPECIFIED BY THE USER AGREE WITH THOSE READ
C     IN FROM FILE 3.
  120 ITEMP = (NOLD-N)**2 + (MATA-MATNOA)**2 + (MATB-MATNOB)**2
      IF (ITEMP.NE.0) GO TO 1370
C
C     READ IN FROM FILE 3, THE T-MULTIPLICITIES OF THE EIGENVALUES
C     WHOSE EIGENVECTORS ARE TO BE COMPUTED, THE VALUES OF THESE
C     EIGENVALUES AND THEIR MINIMAL GAPS AS EIGENVALUES OF THE
C     USER-SPECIFIED MATRIX AND AS EIGENVALUES OF THE T-MATRIX.
C
      READ(3,20) EXPLAN
      READ(3,130) (MP(J),GOODEV(J),TMINGP(J),AMINGP(J), J=1,NGOOD)
  130 FORMAT(6X,I6,E25.16,2E14.3)
C
      WRITE(6,140) (J,GOODEV(J),MP(J),TMINGP(J),AMINGP(J), J=1,NGOOD)
  140 FORMAT(/' EIGENVALUES READ IN, T-MULTIPLICITIES, T-GAPS AND A-GAPS
     1 '/4X,' J ',5X,'GOOD EIGENVALUE',5X,'MULT',4X,'  TMINGAP ',4X,
     1'  ABMINGAP '/(I6,E25.16,I4,2E15.4))
C
C     READ IN ERROR ESTIMATES
      WRITE(6,150) MEV,SVSEED
  150 FORMAT(/' THESE EIGENVALUES WERE COMPUTED USING A T-MATRIX OF
     1ORDER ',I5/' AND SEED FOR RANDOM NUMBER GENERATOR =',I12)
C     CHECK WHETHER OR NOT THERE ARE ANY T-ISOLATED EIGENVALUES IN
C     THE EIGENVALUES PROVIDED
      DO 160 J=1,NGOOD
      IF(MP(J).EQ.1) GO TO 170
  160 CONTINUE
      GO TO 190
  170 READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,180) NISO
  180 FORMAT(18X,I6)
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
  190 DO 220 J=1,NGOOD
      ERR(J) = 0.D0
      IF(MP(J).NE.1) GO TO 220
      READ(4,200) EVAL, ERR(J)
  200 FORMAT(10X,E25.16,E14.3)
      IF(DABS(EVAL - GOODEV(J)).LT.1.D-10) GO TO 220
      WRITE(6,210) EVAL,GOODEV(J)
  210 FORMAT(' PROBLEM WITH READ IN OF ERROR ESTIMATES'/' EIGENVALUE REAL
     1D IN',E20.12,' DOES NOT MATCH GOODEV(J) ='/E20.12)
```

```
      GO TO 1590
C
  220 CONTINUE
C
      WRITE(6,230) (J,GOODEV(J),ERR(J), J=1,NGOOD)
  230 FORMAT(' ERROR ESTMATES ='/4X,' J',5X,'EIGENVALUE',10X,' ESTIMATE
     1'/(I6,E20.12,E14.3))
C
      IF(IREAD.EQ.0)  GO TO 330
C
C     READ IN THE SIZE OF THE T-MATRIX PROVIDED ON FILE 2.  READ IN
C     THE ORDER OF THE USER-SPECIFIED MATRIX , THE SEED FOR THE
C     RANDOM NUMBER GENERATOR, AND THE MATRIX/TEST IDENTIFICATION
C     NUMBERS THAT WERE USED IN THE LANCZOS EIGENVALUE COMPUTATIONS.
C     THESE ARE USED IN A CONSISTENCY CHECK
C     IF FLAG IREAD = 0 REGENERATE ALPHA, BETA
C
      READ(2,240) KMAX,NOLD,SVSOLD,MATA,MATB
  240 FORMAT(2I6,I12,2I8)
C
      WRITE(6,250) KMAX,NOLD,SVSOLD,MATA,MATB
  250 FORMAT(/' READ IN THE T-MATRICES STORED ON FILE 2'/' FILE 2 HEADER
     1 IS'/2X,'KMAX',2X,'NOLD',6X,'SVSOLD',2X,'MATNOA',2X,'MATNOB'/
     1 2I6,I12,2I8/)
C
C     CHECK THAT THE ORDER, THE MATRIX/TEST IDENTIFICATION NUMBERS
C     AND THE SEED FOR THE RANDOM NUMBER GENERATOR USED IN THE
C     LANCZOS COMPUTATIONS THAT GENERATED THE HISTORY FILE
C     BEING USED AGREE WITH WHAT THE USER HAS SPECIFIED.
      IF (NOLD.NE.N.OR.MATA.NE.MATNOA.OR.MATNOB.NE.MATB.OR.SVSOLD.NE.
     1 SVSEED) GO TO 1390
C
      KMAX1 = KMAX + 1
C
C     READ IN THE T-MATRICES FROM FILE 2.  THESE ARE USED TO GENERATE
C     THE T-EIGENVECTORS THAT WILL BE USED IN THE RITZ VECTOR
C     COMPUTATIONS.  HISTORY MUST BE STORED IN MACHINE FORMAT
C     ((4Z20) FOR IBM/3081).
C
      READ(2,260) (ALPHA(J), J=1,KMAX)
      READ(2,260) (BETA(J), J=1,KMAX1)
  260 FORMAT(4Z20)
C
      READ(2,260) (V1(J), J=1,N)
      READ(2,260) (VS(J), J=1,N)
      READ(2,260) (V2(J), J=1,N)
C
C     KMAX MAY BE ENLARGED IF THE SIZE AT WHICH THE EIGENVALUE
C     COMPUTATIONS WERE PERFORMED IS ESSENTIALLY KMAX AND
C     THERE IS AT LEAST ONE EIGENVALUE THAT IS T-SIMPLE AND
C     T-ISOLATED, IN THE SENSE THAT IF ITS NEAREST NEIGHBOR IS TOO
C     CLOSE THAT NEIGHBOR IS A 'GOOD' T-EIGENVALUE.
      DO 270 J = 1,NGOOD
      IF(MP(J).EQ.1) GO TO 290
  270 CONTINUE
```

```
      WRITE(6,280)
  280 FORMAT(/' ALL EIGENVALUES USED ARE T-MULTIPLE OR CLOSE TO SPURIOUS
     1 T-EIGENVALUES'/' SO DO NOT CHANGE KMAX')
      IF(KMAX.LT.MEV) GO TO 1410
      GO TO 310
C
  290 KMAXN= 11*MEV/8 + 12
      IF(MBETA.LE.KMAXN) GO TO 1570
      IF(KMAX.GE.KMAXN )  GO TO 310
      WRITE(6,300) KMAX, KMAXN
  300 FORMAT(' ENLARGE KMAX FROM ',I6,' TO ',I6)
      MOLD1 = KMAX + 1
      KMAX = KMAXN
      GO TO 380
C
  310 WRITE(6,320) KMAX
  320 FORMAT(/' T-MATRICES HAVE BEEN READ IN FROM FILE 2'/' THE LARGEST
     1SIZE T-MATRIX ALLOWED IS',I6/)
C
      IF(IREAD.EQ.1) GO TO 400
C
C     REGENERATE THE ALPHA AND BETA
C
  330 MOLD1 = 1
C
      DO 340 J = 1,NGOOD
      IF(MP(J).EQ.1) GO TO 360
  340 CONTINUE
      KMAX = MEV + 12
      WRITE(6,350) KMAX
  350 FORMAT(/' ALL EIGENVALUES FOR WHICH EIGENVECTORS ARE TO BE COMPUTE
     1D ARE EITHER T-MULTIPLE OR CLOSE TO'/' A SPURIOUS T-EIGENVALUE. TH
     1EREFORE SET KMAX = MEV + 12 = ',I7)
      GO TO 380
C
  360 KMAXN = 11*MEV/8 + 12
      IF(MBETA.LE.KMAXN) GO TO 1570
      WRITE(6,370) KMAXN
  370 FORMAT(' SET KMAX EQUAL TO ',I6)
      KMAX = KMAXN
C
  380 WRITE(6,390) MOLD1,KMAX
  390 FORMAT(/' LANCZS SUBROUTINE GENERATES ALPHA(J), BETA(J+1), J =',
     1 I6,' TO ', I6/)
C
C-----------------------------------------------------------------------
C
      IIX = SVSEED
      CALL LANCZS(LSOLV,AMATV,ALPHA,BETA,V1,V2,VS,G,KMAX,MOLD1,N,IIX)
C
C-----------------------------------------------------------------------
C
  400 CONTINUE
C
C     THE SUBROUTINE STURMI DETERMINES THE SMALLEST SIZE T-MATRIX FOR
```

```
C     WHICH THE EIGENVALUE IN QUESTION IS A T-EIGENVALUE (TO WITHIN A
C     GIVEN TOLERANCE) AND IF POSSIBLE THE SMALLEST SIZE T-MATRIX
C     FOR WHICH IT IS A DOUBLE T-EIGENVALUE (TO WITHIN THE SAME
C     TOLERANCE).  THE SIZE T-MATRIX USED IN THE RITZ VECTOR
C     COMPUTATIONS IS THEN DETERMINED BY LOOPING ON SIZE OF THE
C     T-EIGENVECTORS, STARTING WITH A T-SIZE DETERMINED BY STURMI.
C
C
      STUTOL = SCALE0*MULTOL
      IF(IWRITE.EQ.1) WRITE(6,410)
  410 FORMAT(' FROM STURMI')
      DO 450 J = 1,NGOOD
      EVAL = GOODEV(J)
C     COMPUTE THE TOLERANCES USED BY STURMI TO DETERMINE AN INTERVAL
C     CONTAINING THE EIGENVALUE EVAL.
      TEMP = DABS(EVAL)*RELTOL
      TOLN = DMAX1(TEMP,STUTOL)
C
C-----------------------------------------------------------------------
C
      CALL STURMI(ALPHA,BETA,EVAL,TOLN,EPSM,KMAX,MK1,MK2,IC,IWRITE)
C
C-----------------------------------------------------------------------
C
C     STORE THE COMPUTED ORDERS OF T-MATRICES FOR LATER PRINTOUT
      M1(J) = MK1
      M2(J) = MK2
      ML(J) = (MK1 + 3*MK2)/4
      IF(MK2.EQ.KMAX)  ML(J) = KMAX
C
      IF(IC.GT.0) GO TO 430
C     IC = 0 MEANS THERE WAS NO EIGENVALUE IN THE DESIGNATED INTERVAL
C     BY T-SIZE KMAX.  THIS MEANS THAT THE EIGENVALUE PROVIDED HAS
C     NOT YET CONVERGED SO ITS EIGENVECTOR IS NOT COMPUTED.
      WRITE(6,420) J,GOODEV(J),MK1,MK2
  420 FORMAT(I6,'TH EIGENVALUE',E20.12,' HAS NOT CONVERGED '/
     1' SO DO NOT COMPUTE ANY T-EIGENVECTOR OR RITZ VECTOR FOR IT'
     1/' MK1 AND MK2 FOR THIS EIGENVALUE WERE',2I6)
      MP(J) = MPMIN
      MA(J) = -2*KMAX
      GO TO 450
C     COMPUTE AN APPROPRIATE SIZE T-MATRIX FOR THE GIVEN EIGENVALUE.
  430 IF(M2(J).EQ.KMAX) GO TO 440
C     M1 AND M2 WERE BOTH DETERMINED
      MA(J) = (3*M1(J) + M2(J))/4  + 1
      GO TO 450
C     M2 NOT DETERMINED
  440 MA(J) = (5*M1(J))/4  + 1
C
  450 CONTINUE
C
      IF (IWRITE.EQ.1) WRITE(6,460) (MA(JJ), JJ=1,NGOOD)
  460 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZE T-MATRICES'/
     1 ' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/(13I6))
C
```

```
C     PRINT OUT TO FILE 10 1ST GUESSES AT SIZES OF THE T-MATRICES TO
C     BE USED IN THE EIGENVECTOR COMPUTATIONS.
C     PROGRAM LOOPS ON T-SIZE TO DETERMINE APPROPRIATE SIZE T-MATRIX.
      WRITE(10,470) N,KMAX
  470 FORMAT(2I8,' = ORDER OF USER MATRIX AND MAX ORDER OF T(1,MEV)')
C
      WRITE(10,480)
  480 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZE T-MATRICES'/
     1 ' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/)
C
      WRITE(10,490)
  490 FORMAT(4X,'J',3X,'AB-EIGENVALUE',4X,'M1(J)',1X,'M2(J)',1X,'MA(J)')
C
      WRITE(10,500) (J,GOODEV(J),M1(J),M2(J), MA(J), J=1,NGOOD)
  500 FORMAT(I5,E19.12,3I6)
C
      IF(MBOUND.EQ.1) WRITE(10,510)
  510 FORMAT(/'  EV = AB-EIGENVALUE IS A GOOD EIGENVALUE OF T(1,MEV)'/
     1' M1 = SMALLEST VALUE OF M SUCH THAT T(1,M) HAS AT LEAST'/
     1 '     ONE EIGENVALUE IN THE INTERVAL (EV-TOLN,EV+TOLN)'/
     1 '     TOLN(J) = DMAX1(EV(J)*RELTOL, SCALE0*MULTOL)'/
     1 ' M2 = SMALLEST M (IF ANY) SUCH THAT IN THE ABOVE INTERVAL'/
     1 '      T(1,M) HAS AT LEAST TWO EIGENVALUES '/
     1 ' IABS(MA(J)) = APPROPRIATE SIZE T-MATRIX FOR EV(J)'/
     1 ' INITIAL VALUE OF MA(J) IS CHOSEN HEURISTICALLY'/
     1 ' PROGRAM LOOPS ON SIZE OF T-MATRIX TO GET BETTER SIZE'/
     1 ' END OF SIZES OF T-MATRICES FILE 10'///)
C
C
C      TERMINATE AFTER COMPUTING 1ST GUESSES AT  SIZES OF THE
C     T-MATRICES REQUIRED FOR THE GIVEN EIGENVALUES?
      IF(MBOUND.EQ.1) GO TO 1430
C
C
C     IS THERE ROOM FOR ALL OF THE REQUESTED T-EIGENVECTORS?
      MTOL = 0
      DO 520 J = 1,NGOOD
      IF(MP(J).EQ.MPMIN) GO TO 520
      MTOL = MTOL + IABS(MA(J))
  520 CONTINUE
      MTOL = (5*MTOL)/4
      IF(MTOL.GT.MDIMTV.AND.NTVCON.EQ.0) GO TO 1450
C
C-----------------------------------------------------------------------
C     GENERATE A RANDOM VECTOR TO BE USED REPEATEDLY BY
C     SUBROUTINE INVERM
C
      IIL = RHSEED
      CALL GENRAN(IIL,G,KMAX)
C
C----------------------------------------------------------------------
C
C     LOOP ON GIVEN EIGENVALUES TO COMPUTE THE CORRESPONDING
C     T-EIGENVECTOR.
C
```

```
      MTOL = 0                                                          LGV05410
      NTVEC = 0                                                         LGV05420
      ILBIS = 0                                                         LGV05430
      DO 710 J = 1,NGOOD                                                LGV05440
      ICOUNT = 0                                                        LGV05450
      ERRMIN = 10.D0                                                    LGV05460
      MABEST = MPMIN                                                    LGV05470
      IF(MP(J).EQ.MPMIN) GO TO 710                                      LGV05480
      TFLAG = 0                                                         LGV05490
      EVAL = GOODEV(J)                                                  LGV05500
      TEMP = DABS(EVAL)*RELTOL                                          LGV05510
      UB = EVAL + DMAX1(STUTOL,TEMP)                                    LGV05520
      LB = EVAL - DMAX1(STUTOL,TEMP)                                    LGV05530
  530 KMAXU = IABS(MA(J))                                               LGV05540
C                                                                       LGV05550
C     SELECT A SUITABLE INCREMENT FOR THE ORDERS OF THE T-MATRICES      LGV05560
C     TO BE CONSIDERED IN DETERMINING APPROPRIATE SIZES FOR THE RITZ    LGV05570
C     VECTOR COMPUTATIONS.                                              LGV05580
      IF(ICOUNT.GT.0) GO TO 550                                         LGV05590
C     SELECT IDELTA(J) BASED UPON THE T-MULTIPLICITY OBTAINED           LGV05600
      IF(M2(J).EQ.KMAX) GO TO 540                                       LGV05610
C     M2 DETERMINED                                                     LGV05620
      IDELTA(J) = ((3*M1(J) + 5*M2(J))/8  +  1 - IABS(MA(J)))/10  +  1  LGV05630
      GO TO 550                                                         LGV05640
C     M2 NOT DETERMINED                                                 LGV05650
  540 MAMAX = MIN0((11*MEV)/8 + 12, (13*M1(J))/8 + 1)                   LGV05660
      IDELTA(J) = (MAMAX - IABS(MA(J)))/10  +  1                        LGV05670
  550 ICOUNT = ICOUNT + 1                                               LGV05680
C                                                                       LGV05690
C-----------------------------------------------------------------------LGV05700
C     TO MIMIMIZE THE EFFECT OF THE ONE-SIDED ACCEPTANCE TEST FOR       LGV05710
C     EIGENVALUES IN THE BISEC SUBROUTINE, RECOMPUTE THE GIVEN          LGV05720
C     EIGENVALUE AT THE SPECIFIED KMAXU                                 LGV05730
C                                                                       LGV05740
      CALL LBISEC(ALPHA,BETA,EPSM,EVAL,EVALN,LB,UB,TTOL,KMAXU,NEVT)     LGV05750
C                                                                       LGV05760
C-----------------------------------------------------------------------LGV05770
C                                                                       LGV05780
C     CHECK WHETHER OR NOT GIVEN T-MATRIX HAS AN EIGENVALUE IN THE      LGV05790
C     SPECIFIED INTERVAL AND IF SO WHAT ITS T-MULTIPLICITY IS.          LGV05800
C                                                                       LGV05810
      IF(NEVT.EQ.1) GO TO 590                                           LGV05820
      IF(NEVT.NE.0) GO TO 570                                           LGV05830
      ILBIS = 1                                                         LGV05840
      WRITE(6,560) EVAL,KMAXU                                           LGV05850
  560 FORMAT(/' PROBLEM ENCOUNTERED IN RECOMPUTATION OF USER-SUPPLIED EILGV05860
     1GENVALUE',E20.12/' THE SIZE T-MATRIX SPECIFIED',I6,' DOES NOT     LGV05870
     1HAVE AN EIGENVALUE IN THE INTERVAL SPECIFIED'/' THEREFORE NO EIGENLGV05880
     1VECTOR WILL BE COMPUTED FOR THIS PARTICULAR EIGENVALUE'/)         LGV05890
      GO TO 610                                                         LGV05900
C                                                                       LGV05910
  570 IF(NEVT.GT.1)  WRITE(6,580) EVAL,KMAXU                            LGV05920
  580 FORMAT(/' PROBLEM ENCOUNTERED IN RECOMPUTATION OF USER-SUPPLIED   LGV05930
     1EIGENVALUE',E20.12/' FOR THE SIZE T-MATRIX SPECIFIED =',I6,' THE  LGV05940
     1GIVEN EIGENVALUE IS T-MULTIPLE IN THE INTERVAL SPECIFIED'/' SOMETHLGV05950
```

```
     1ING IS WRONG, THEREFORE NO EIGENVECTOR WILL BE COMPUTED FOR THIS ELGV05960
     1IGENVALUE')/)                                                     LGV05970
C                                                                       LGV05980
      MP(J) = MPMIN                                                     LGV05990
      MA(J) = -2*KMAX                                                   LGV06000
      GO TO 710                                                         LGV06010
C                                                                       LGV06020
  590 CONTINUE                                                          LGV06030
      ILBIS = 0                                                         LGV06040
C                                                                       LGV06050
      EVNEW(J) = EVALN                                                  LGV06060
      EVAL = EVALN                                                      LGV06070
      MTOL = MTOL+KMAXU                                                 LGV06080
C                                                                       LGV06090
C     IS THERE ROOM IN TVEC ARRAY FOR THE NEXT T-EIGENVECTOR?           LGV06100
C     IF NOT, SKIP TO RITZ VECTOR COMPUTATIONS.                         LGV06110
      IF (MTOL.GT.MDIMTV) GO TO 720                                     LGV06120
C                                                                       LGV06130
      IT = 3                                                            LGV06140
      KINT = MTOL - KMAXU +1                                            LGV06150
C                                                                       LGV06160
C     RECORD THE BEGINNING AND END OF THE T-EIGENVECTOR BEING COMPUTED  LGV06170
      MINT(J) = KINT                                                    LGV06180
      MFIN(J) = MTOL                                                    LGV06190
C                                                                       LGV06200
C-----------------------------------------------------------------------LGV06210
C     SUBROUTINE INVERM DOES INVERSE ITERATION, I.E. SOLVES             LGV06220
C     (T(1,KMAXU) - EVAL)*U = RHS  FOR EACH EIGENVALUE TO OBTAIN THE    LGV06230
C     DESIRED T-EIGENVECTOR.                                            LGV06240
C                                                                       LGV06250
      IF(IWRITE.EQ.1)  WRITE(6,600) J                                   LGV06260
  600 FORMAT(/I6,'TH EIGENVALUE')                                       LGV06270
C                                                                       LGV06280
      CALL INVERM(ALPHA,BETA,V1,TVEC(KINT),EVAL,ERROR,TERROR,EPSM,      LGV06290
     1 G,KMAXU,IT,IWRITE)                                               LGV06300
C                                                                       LGV06310
C-----------------------------------------------------------------------LGV06320
C                                                                       LGV06330
      TERR(J) = TERROR                                                  LGV06340
      TLAST(J) = ERROR                                                  LGV06350
      KMAXU1 = KMAXU + 1                                                LGV06360
      TBETA(J) = BETA(KMAXU1)*ERROR                                     LGV06370
C                                                                       LGV06380
C     AFTER COMPUTING EACH OF THE T-EIGENVECTORS,                       LGV06390
C     CHECK THE SIZE OF THE ERROR ESTIMATE, ERROR.                      LGV06400
C     IF THIS ESTIMATE IS NOT AS SMALL AS DESIRED AND                   LGV06410
C     |MA(J)| < ML(J), ATTEMPT TO INCREASE THE SIZE OF |MA(J)|          LGV06420
C     AND REPEAT THE T-EIGENVECTOR COMPUTATIONS.                        LGV06430
C                                                                       LGV06440
      IF(ERROR.LT.ERTOL.OR.TFLAG.EQ.1)  GO TO 700                       LGV06450
C                                                                       LGV06460
      IF(ERROR.GE.ERRMIN) GO TO 610                                     LGV06470
C     LAST COMPONENT IS LESS THAN MINIMAL TO DATE                       LGV06480
      ERRMIN = ERROR                                                    LGV06490
      MABEST = MA(J)                                                    LGV06500
```

```
  610 CONTINUE                                                          LGV06510
C                                                                       LGV06520
      IF(MA(J).GT.0)  ITEST = MA(J) + IDELTA(J)                         LGV06530
      IF(MA(J).LT.0)  ITEST = -(IABS(MA(J)) + IDELTA(J))                LGV06540
      IF(IABS(ITEST).LE.ML(J).AND.ICOUNT.LE.10) GO TO 630               LGV06550
C     NEW MA(J) IS GREATER THAN MAXIMUM ALLOWED.                        LGV06560
      IF(ERCONT.EQ.0.OR.MABEST.EQ.MPMIN) GO TO 650                      LGV06570
      TFLAG = 1                                                         LGV06580
      MA(J) = MABEST                                                    LGV06590
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU                               LGV06600
      WRITE(6,620) MA(J)                                                LGV06610
  620 FORMAT(' 10 ORDERS WERE CONSIDERED.  NONE SATISFIED THE ERROR TESTLGV06620
     1'/' THEREFORE USE THE BEST ORDER OBTAINED FOR THE EIGENVECTORS'   LGV06630
     1,I6)                                                              LGV06640
      GO TO 530                                                         LGV06650
C                                                                       LGV06660
  630 MA(J) = ITEST                                                     LGV06670
C                                                                       LGV06680
      MT = IABS(MA(J))                                                  LGV06690
      IF(IWRITE.EQ.1) WRITE(6,640)  MT                                  LGV06700
  640 FORMAT(/' CHANGE SIZE OF T-MATRIX TO ',I6,' RECOMPUTE T-EIGENVECTOLGV06710
     1R')                                                               LGV06720
C                                                                       LGV06730
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU                               LGV06740
C                                                                       LGV06750
      GO TO 530                                                         LGV06760
C                                                                       LGV06770
C     APPROPRIATE SIZE T-MATRIX WAS NOT OBTAINED                        LGV06780
  650 CONTINUE                                                          LGV06790
      WRITE(10,660) J,EVAL,MP(J)                                        LGV06800
  660 FORMAT(/' ON 10 INCREMENTS NOT ABLE TO IDENTIFY APPROPRIATE SIZE  LGV06810
     1T-MATRIX FOR'/                                                    LGV06820
     1' EIGENVALUE(',I4,') = ',E20.12,' T-MULTIPLICITY =',I4/)          LGV06830
      IF(M2(J).EQ.KMAX) WRITE(10,670)                                   LGV06840
      IF(M2(J).LT.KMAX) WRITE(10,680)                                   LGV06850
  670 FORMAT(/' ORDERS TESTED RANGED FROM 5*M1(J)/4 TO APPROXIMATELY    LGV06860
     1 '/'  MIN(11*MEV/8,13*M1(J)/8)'/)                                 LGV06870
  680 FORMAT(/' ORDERS TESTED RANGED FROM (3*M1(J)+M2(J))/4 TO APPROXIMALGV06880
     1TELY'/'  (3*M1(J) + 5*M2(J))/8.'/)                                LGV06890
      WRITE(10,690)                                                     LGV06900
  690 FORMAT(' ALLOWING LARGER ORDERS FOR THE T-MATRICES MAY RESULT IN  LGV06910
     1 SUCCESS'/' BUT PROBABLY WILL NOT.  PROBLEM IS PROBABLY DUE TO'   LGV06920
     1 /' LACK OF CONVERGENCE OF GIVEN EIGENVALUE, CHECK THE ERROR ESTIMLGV06930
     1ATE'/)                                                            LGV06940
      MP(J) = MPMIN                                                     LGV06950
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU                               LGV06960
      GO TO 710                                                         LGV06970
  700 NTVEC = NTVEC + 1                                                 LGV06980
C                                                                       LGV06990
  710 CONTINUE                                                          LGV07000
      NGOODC = NGOOD                                                    LGV07010
      GO TO 740                                                         LGV07020
C                                                                       LGV07030
C     COME HERE IF THERE IS NOT ENOUGH ROOM FOR ALL OF T-EIGENVECTORS   LGV07040
  720 NGOODC = J-1                                                      LGV07050
```

```
      WRITE(6,730)  J, MTOL, MDIMTV
  730 FORMAT(/'  NOT ENOUGH ROOM IN TVEC FOR ',I4,'TH T-VECTOR'/'  T-DIM
     1ENSION REQUESTED = ',I6,' BUT TVEC HAS DIMENSION = ',I6/)
      IF(NGOODC.EQ.0)  GO TO 1470
      MTOL = MTOL-KMAXU
C
  740 CONTINUE
C
C     THE LOOP ON T-EIGENVECTOR COMPUTATIONS IS COMPLETE.
C     WRITE OUT THE SIZE T-MATRICES THAT WILL BE USED FOR
C     THE RITZ VECTOR COMPUTATIONS.
C
      WRITE(10,750)
  750 FORMAT(/' SIZES OF T-MATRICES THAT WILL BE USED IN THE RITZ COMPUT
     1ATIONS'/5X,'J',16X,'GOODEV(J)',1X,'MA(J)')
C
      WRITE(10,760)   (J,GOODEV(J),MA(J), J=1,NGOOD)
  760 FORMAT(I6,E25.14,I6)
      WRITE(10,510)
C
      WRITE(6,770) MTOL
  770 FORMAT(/' THE CUMULATIVE LENGTH OF THE T-EIGENVECTORS IS',I18)
C
      WRITE(6,780) NTVEC,NGOOD
  780 FORMAT(/I6,' T-EIGENVECTORS OUT OF',I6,' REQUESTED WERE COMPUTED')
C
C     SAVE THE T-EIGENVECTORS ON FILE 11?
      IF(TVSTOP.NE.1.AND.SVTVEC.EQ.0) GO TO 840
C
      WRITE(11,790) NTVEC,MTOL,MATNOA,MATNOB,SVSEED
  790 FORMAT(I6,3I8,I12,' = NTVEC,MTOL,MATNOA,MATNOB,SVSEED')
C
      DO 820 J=1,NGOODC
C     IF MP(J) = MPMIN THEN NO SUITABLE T-EIGENVECTOR IS AVAILABLE
C     FOR THAT EIGENVALUE.
      IF(MP(J).EQ.MPMIN) WRITE(11,800) J,MA(J),GOODEV(J),MP(J)
  800 FORMAT(2I6,E20.12,I6/' TH EIGVAL,T-SIZE,EVALUE,FLAG,NO EIGVEC')
      IF(MP(J).NE.MPMIN) WRITE(11,810) J,MA(J),GOODEV(J),MP(J)
  810 FORMAT(I6,I6,E20.12,I6/' T-EIGVEC,SIZE T,EVALUE OF A,MP(J)')
      IF(MP(J).EQ.MPMIN) GO TO 820
      KI = MINT(J)
      KF = MFIN(J)
C
      WRITE(11,260) (TVEC(K), K=KI,KF)
C
  820 CONTINUE
C
      IF(TVSTOP.NE.1) GO TO 840
C
      WRITE(6,830) TVSTOP, NTVEC,NGOOD
  830 FORMAT(/' USER SET TVSTOP = ',I1/
     1' THEREFORE PROGRAM TERMINATES AFTER T-EIGENVECTOR COMPUTATIONS'/
     1' T-EIGENVECTORS THAT WERE COMPUTED ARE SAVED ON FILE 11'/
     1I8,' T-EIGENVECTORS WERE COMPUTED OUT OF',I7,' REQUESTED'/)
C
```

```
      GO TO 1590                                                        LGV07610
C                                                                       LGV07620
  840 CONTINUE                                                          LGV07630
C     IF NOT ABLE TO COMPUTE ALL THE REQUESTED T-EIGENVECTORS,          LGV07640
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS ANYWAY?             LGV07650
      IF(NTVEC.NE.NGOOD.AND.LVCONT.EQ.0) GO TO 1490                     LGV07660
C                                                                       LGV07670
C     COMPUTE THE MAXIMUM SIZE OF THE T-MATRIX USED FOR THOSE           LGV07680
C     EIGENVALUES WITH GOOD ERROR ESTIMATES.                            LGV07690
C                                                                       LGV07700
      KMAXU = 0                                                         LGV07710
      DO 850 J = 1,NGOODC                                               LGV07720
      MT = IABS(MA(J))                                                  LGV07730
      IF(MT.LT.KMAXU.OR.MP(J).EQ.MPMIN) GO TO 850                       LGV07740
      KMAXU = MT                                                        LGV07750
  850 CONTINUE                                                          LGV07760
C                                                                       LGV07770
      IF(KMAXU.EQ.0) GO TO 1530                                         LGV07780
C                                                                       LGV07790
      WRITE(6,860) KMAXU                                                LGV07800
  860 FORMAT(/I6,' = LARGEST SIZE T-MATRIX TO BE USED IN THE RITZ VECTORLGV07810
     1 COMPUTATIONS')                                                   LGV07820
C                                                                       LGV07830
C     COUNT THE NUMBER OF RITZ VECTORS NOT BEING COMPUTED               LGV07840
      MREJEC = 0                                                        LGV07850
      DO 870 J=1,NGOODC                                                 LGV07860
  870 IF(MP(J).EQ.MPMIN)  MREJEC = MREJEC + 1                           LGV07870
      MREJET = MREJEC + (NGOOD-NGOODC)                                  LGV07880
      IF(MREJET.NE.0) WRITE(6,880) MREJET                               LGV07890
  880 FORMAT(/' RITZ VECTORS ARE NOT COMPUTED FOR',I6,' OF THE EIGENVALULGV07900
     1ES'/)                                                             LGV07910
      NACT = NGOODC - MREJEC                                            LGV07920
      WRITE(6,890) NGOOD,NTVEC,NACT                                     LGV07930
  890 FORMAT(/I6,' RITZ VECTORS WERE REQUESTED'/I6,' T-EIGENVECTORS WERELGV07940
     1 COMPUTED'/I6,' RITZ VECTORS WILL BE COMPUTED'/)                  LGV07950
C     CHECK IF THERE ARE ANY RITZ VECTORS TO COMPUTE                    LGV07960
      IF(MREJEC.EQ.NGOODC) GO TO 1510                                   LGV07970
C                                                                       LGV07980
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS?                    LGV07990
      IF(LVCONT.EQ.0.AND.MREJEC.NE.0) GO TO 1490                        LGV08000
C                                                                       LGV08010
C     NOW COMPUTE THE RITZ VECTORS.  REGENERATE THE                     LGV08020
C     LANCZOS VECTORS.                                                  LGV08030
C                                                                       LGV08040
      DO 900 I = 1,NMAX                                                 LGV08050
  900 RITVEC(I) = ZERO                                                  LGV08060
C                                                                       LGV08070
C-----------------------------------------------------------------------LGV08080
C     REGENERATE THE STARTING VECTOR. THIS MUST BE GENERATED AND        LGV08090
C     NORMALIZED PRECISELY THE WAY IT WAS DONE IN THE EIGENVALUE        LGV08100
C     COMPUTATIONS, OTHERWISE THERE WILL BE A MISMATCH BETWEEN          LGV08110
C     THE T-EIGENVECTORS THAT HAVE BEEN COMPUTED FROM THE T-MATRICES    LGV08120
C     READ IN FROM FILE 2 AND THE LANCZOS VECTORS THAT ARE              LGV08130
C     BEING REGENERATED.                                                LGV08140
C                                                                       LGV08150
```

```
      IIL = SVSEED
      CALL GENRAN(IIL,G,N)
C
C-----------------------------------------------------------------------
C
      DO 910 K = 1,N
  910 V2(K) = G(K)
C
C-----------------------------------------------------------------------
C     COMPUTE L-TRANSPOSE*V2 AND ITS NORM
      ISOLV = 2
      CALL LSOLV(V2,VS,ISOLV)
      SUM = FINPRO(N,VS(1),1,VS(1),1)
C-----------------------------------------------------------------------
C
C     NORMALIZE STARTING VECTORS:  (V2-TRANSPOSE*B*V2) = 1
      SUM = ONE/DSQRT(SUM)
      DO 920 K = 1,N
      VS(K) = SUM*VS(K)
  920 V2(K) = SUM*V2(K)
C
C-----------------------------------------------------------------------
C     INITIALIZE V1 = B*V2 = L*VS
      ISOLV = 1
      CALL LSOLV(VS,V1,ISOLV)
C-----------------------------------------------------------------------
C
      DO 930 K = 1,N
      VS(K) = V1(K)
  930 V1(K) = ZERO
C
      IVEC = 1
      BATA = ZERO
C
      GO TO 1000
C
C     VS = B*V(I),  V1 = B*V(I-1),  V2 = V(I)
  940 CONTINUE
      SUM = BATA
C
C-----------------------------------------------------------------------
C     COMPUTE V1 = A*V2 - SUM*V1
      CALL MATVEC(V2,V1,SUM)
C     COMPUTE ALFA
      ALFA = FINPRO(N,V1(1),1,V2(1),1)
C-----------------------------------------------------------------------
C
      DO 950 K = 1,N
  950 V1(K) = V1(K)-ALFA*VS(K)
C
C     SET V1 = B*V(IVEC) AND VS = (NEW BATA)*B*V(IVEC+1)
      DO 960 K = 1,N
      TEMP = V1(K)
      V1(K) = VS(K)
  960 VS(K) = TEMP
```

```
C                                                                       LGV08710
C-----------------------------------------------------------------------LGV08720
C     COMPUTE V2 = (L-INVERSE)*VS                                       LGV08730
      ISOLV = 3                                                         LGV08740
      CALL LSOLV(VS,V2,ISOLV)                                           LGV08750
C     COMPUTE NEXT BATA                                                 LGV08760
      SUM = FINPRO(N,V2(1),1,V2(1),1)                                   LGV08770
C-----------------------------------------------------------------------LGV08780
C                                                                       LGV08790
      BATA = DSQRT(SUM)                                                 LGV08800
      TEMP = BETA(IVEC)                                                 LGV08810
      TEMP = DABS(BATA - TEMP)/TEMP                                     LGV08820
      IF (TEMP.LT.1.0D-10)GO TO 980                                     LGV08830
C                                                                       LGV08840
C     THE BETA BEING REGENERATED DO NOT MATCH THE BETA IN FILE 2.       LGV08850
C     SOMETHING IS WRONG IN THE LANCZOS VECTOR GENERATION.              LGV08860
C     PROGRAM TERMINATES FOR USER TO CORRECT THE PROBLEM                LGV08870
C     WHICH MUST BE IN THE STARTING VECTOR GENERATION OR IN             LGV08880
C     THE SUBROUTINES AMATV AND LSOLV SUPPLIED.                         LGV08890
C     THESE SUBROUTINES MUST BE THE SAME ONES USED IN THE               LGV08900
C     EIGENVALUE COMPUTATIONS OR A MISMATCH WILL ENSUE.                 LGV08910
C                                                                       LGV08920
      WRITE(6,970) IVEC,BATA,BETA(IVEC),TEMP                            LGV08930
  970 FORMAT(/2X,'IVEC',16X,'BATA',10X,'BETA(IVEC)',14X,'RELDIF'/I6,    LGV08940
     13E20.12/' IN LANCZOS VECTOR REGENERATION THE ENTRIES OF THE TRIDIALGV08950
     1GONAL MATRICES BEING'/' GENERATED ARE NOT THE SAME AS THOSE IN THELGV08960
     1 MATRIX SUPPLIED ON FILE 2.'/' THEREFORE SOMETHING IS BEING INITIALGV08970
     1LIZED OR COMPUTED DIFFERENTLY FROM THE WAY'/' IT WAS COMPUTED IN TLGV08980
     1HE EIGENVALUE COMPUTATIONS'/' THE PROGRAM TERMINATES FOR THE USER LGV08990
     1TO DETERMINE WHAT THE PROBLEM IS'/)                               LGV09000
      GO TO 1590                                                        LGV09010
  980 CONTINUE                                                          LGV09020
C                                                                       LGV09030
C-----------------------------------------------------------------------LGV09040
      ISOLV = 4                                                         LGV09050
      CALL LSOLV(V2,V2,ISOLV)                                           LGV09060
C-----------------------------------------------------------------------LGV09070
C                                                                       LGV09080
      SUM = ONE/BATA                                                    LGV09090
      DO 990 K = 1,N                                                    LGV09100
      V2(K) = SUM*V2(K)                                                 LGV09110
  990 VS(K) = SUM*VS(K)                                                 LGV09120
C                                                                       LGV09130
 1000 CONTINUE                                                          LGV09140
C                                                                       LGV09150
      LFIN = 0                                                          LGV09160
      DO 1020 J = 1,NGOODC                                              LGV09170
      LL = LFIN                                                         LGV09180
      LFIN = LFIN + N                                                   LGV09190
C                                                                       LGV09200
      IF(IABS(MA(J)).LT.IVEC.OR.MP(J).EQ.MPMIN) GO TO 1020              LGV09210
      II = IVEC + MINT(J) - 1                                           LGV09220
      TEMP = TVEC(II)                                                   LGV09230
C     II IS THE (IVEC)TH COMPONENT OF THE T-EIGENVECTOR CONTAINED       LGV09240
C     IN TVEC(MINT(J)).                                                 LGV09250
```

```
C                                                                       LGV09260
      DO 1010 K = 1,N                                                   LGV09270
      LL = LL + 1                                                       LGV09280
 1010 RITVEC(LL) = TEMP*V2(K) + RITVEC(LL)                              LGV09290
C                                                                       LGV09300
 1020 CONTINUE                                                          LGV09310
C                                                                       LGV09320
      IVEC = IVEC + 1                                                   LGV09330
      IF (IVEC.LE.KMAXU) GO TO 940                                      LGV09340
C                                                                       LGV09350
C     RITZVECTOR GENERATION IS COMPLETE. B-NORMALIZE EACH RITZVECTOR.   LGV09360
C     NOTE THAT IF CERTAIN RITZ VECTORS WERE NOT COMPUTED THEN THAT     LGV09370
C     PORTION OF THE RITVEC ARRAY WAS NOT UTILIZED.                     LGV09380
C                                                                       LGV09390
      LFIN = 0                                                          LGV09400
      DO 1090 J = 1,NGOODC                                              LGV09410
C                                                                       LGV09420
      KK = LFIN                                                         LGV09430
      LFIN = LFIN + N                                                   LGV09440
      IF(MP(J).EQ.MPMIN) GO TO 1090                                     LGV09450
C                                                                       LGV09460
      DO 1030 K = 1,N                                                   LGV09470
      KK = KK + 1                                                       LGV09480
 1030 V2(K) = RITVEC(KK)                                                LGV09490
C                                                                       LGV09500
C-----------------------------------------------------------------------LGV09510
      ISOLV = 2                                                         LGV09520
      CALL LSOLV(V2,VS,ISOLV)                                           LGV09530
      SUM = FINPRO(N,VS(1),1,VS(1),1)                                   LGV09540
C-----------------------------------------------------------------------LGV09550
C                                                                       LGV09560
      SUM = DSQRT(SUM)                                                  LGV09570
      RNORM(J) = SUM                                                    LGV09580
      TEMP = DABS(ONE-SUM)                                              LGV09590
      SUM = ONE/SUM                                                     LGV09600
C                                                                       LGV09610
      DO 1040 K = 1,N                                                   LGV09620
      VS(K) = SUM*VS(K)                                                 LGV09630
      V2(K) = SUM*V2(K)                                                 LGV09640
 1040 CONTINUE                                                          LGV09650
C                                                                       LGV09660
C-----------------------------------------------------------------------LGV09670
      ISOLV = 1                                                         LGV09680
      CALL LSOLV(VS,V1,ISOLV)                                           LGV09690
C-----------------------------------------------------------------------LGV09700
C                                                                       LGV09710
C     V1 = B*V2                                                         LGV09720
      EVAL = EVNEW(J)                                                   LGV09730
C                                                                       LGV09740
C     COMPUTE ERROR IN RITZ VECTOR CONSIDERED AS A EIGENVECTOR OF A.    LGV09750
C     V1 = A*RITVEC - EVAL*B*RITVEC                                     LGV09760
C                                                                       LGV09770
C-----------------------------------------------------------------------LGV09780
      CALL AMATV(V2,V1,EVAL)                                            LGV09790
      SUM = FINPRO(N,V1(1),1,V1(1),1)                                   LGV09800
```

```
C-----------------------------------------------------------------------
C
      SUM = DSQRT(SUM)
      ERR(J) = SUM
      GAP = ABS(AMINGP(J))
      ERRDGP(J) = SUM/GAP
C
C
      IF (JPERM.EQ.0) GO TO 1050
C
C-----------------------------------------------------------------------
C     ON RETURN V2 = P(TRANSPOSE)*V2
      IPERM = 2
      CALL LPERM(V2,V1,IPERM)
C-----------------------------------------------------------------------
C
 1050 CONTINUE
      KK = LFIN - N
      DO 1060 K = 1,N
      KK = KK + 1
 1060 RITVEC(KK) = V2(K)
C
      IF (IWRITE.NE.0) WRITE(6,1070) J,GOODEV(J)
 1070 FORMAT(/I5,' TH EIGENVALUE CONSIDERED = ',E20.12/)
C
      IF (IWRITE.NE.0) WRITE(6,1080) TERR(J),TBETA(J),TEMP
 1080 FORMAT(' NORM OF ERROR IN T-EIGENVECTOR = ',E14.3/
     1 ' BETA(MA(J)+1)*U(MA(J)) = ',E14.3/
     1 ' ABS(NORM(RITVEC) - 1.0)  = ',E14.3/)
C
 1090 CONTINUE
C
C     RITZVECTORS ARE NORMALIZED AND ERROR ESTIMATES ARE IN ERR ARRAY
C     AND IN ERRDGP ARRAY. STORE EVERYTHING
C
C
      WRITE(9,1100)
 1100 FORMAT(2X,'AB-EIGENVALUE',2X,'MA(J)',2X,'AB-MINGAP',5X,'ABERROR',1
     1X, 'ABERROR/GAP',6X,'TERROR')
C
      WRITE(13,1110)
 1110 FORMAT(12X,'AB-EIGENVALUE',5X,'RITZNORM',5X,'ABMINGAP',5X,
     1 'TBETA(J)',5X,'TLAST(J)')
C
      DO 1140 J=1,NGOODC
C
      IF(MP(J).EQ.MPMIN) GO TO 1140
C
      WRITE(9,1120)EVNEW(J),MA(J),AMINGP(J),ERR(J),ERRDGP(J),TERR(J)
 1120 FORMAT(E15.8,I6,4E12.4)
C
      WRITE(13,1130) EVNEW(J),RNORM(J),AMINGP(J),TBETA(J),TLAST(J)
 1130 FORMAT(E25.14,4E13.5)
C
 1140 CONTINUE
```

```
C
      IF(MREJEC.EQ.0) GO TO 1220
      WRITE(9,1150)
 1150 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING EIGENVA
     1LUES'/' EITHER BECAUSE THEY HAD NOT CONVERGED OR BECAUSE THE ERROR
     1 ESTIMATE'/'  WAS NOT AS SMALL AS DESIRED'/)
C
      DO 1210 J = 1,NGOODC
      IF(MP(J).NE.MPMIN) GO TO 1210
C     WRITE OUT MESSAGE FOR EACH EIGENVALUE FOR WHICH NO EIGENVECTOR
C     WAS COMPUTED.
C
      WRITE(9,1160)
 1160 FORMAT(2X,'AB-EIGENVALUE',3X,'MA(J)',5X,'AMINGP(J)',6X,'TLAST(J)',
     13X,'MP(J)')
      WRITE(9,1170) GOODEV(J),MA(J),AMINGP(J),TBETA(J),MP(J)
 1170 FORMAT(E15.8,I8,2E14.4,I8)
C
      WRITE(13,1180)
 1180 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING EIGENVA
     1LUES'/' BECAUSE THEY HAD NOT CONVERGED'/)
C
      WRITE(13,1190)
 1190 FORMAT(2X,'AB-EIGENVALUE',3X,'MA(J)',3X,'M1(J)',3X,'M2(J)',3X,'MP(
     1J)'/)
      WRITE(13,1200) GOODEV(J),MA(J),M1(J),M2(J),MP(J)
 1200 FORMAT(E15.8,4I8)
C
 1210 CONTINUE
 1220 CONTINUE
C
      WRITE(9,1230)
 1230 FORMAT(/' ABOVE ARE ERROR ESTIMATES FOR THE AB AND T EIGENVECTORS'
     1 /' ASSOCIATED WITH THE AB-EIGENVALUES LISTED IN COLUMN 1'/
     1 ' ABERROR = NORM(A*X - EV*B*X)  TERROR = NORM(T*Y - EV*Y)
     1 '/' WHERE T = T(1,MA(J))    X = RITZ VECTOR = V*Y  V = SUCCESSIVE
     1 '/' LANCZOS VECTORS. ABMINGAP = GAP TO NEAREST AB-EIGENVALUE'//)
C
      WRITE(13,1240)
 1240 FORMAT(/' ABOVE ARE ERROR ESTIMATES ASSOCIATED WITH THE AB-EIGVALS
     1 '/' RITZNORM = NORM(COMPUTED RITZ VECTOR)'/
     1 ' TBETA(J) = BETA(MA(J)+1)*Y(MA(J)),  T*Y = EVAL*Y'/
     1 ' TLAST(J) = Y(MA(J))'/
     1 ' ABMINGAP = GAP TO NEAREST AB-EIGENVALUE'/)
C
C     NUMBER OF RITZ VECTORS COMPUTED
      NCOMPU = NGOODC - MREJEC
      WRITE(12,1250) N,NCOMPU,NGOODC,MATNOA,MATNOB
 1250 FORMAT(3I6,2I8,' SIZE A, NO.RITZVECS, NO.EVALS,MATNOA,MATNOB')
C
      LFIN = 0
      DO 1310 J = 1,NGOODC
      LINT = LFIN + 1
      LFIN = LFIN + N
C
```

```
      IF(MP(J).EQ.MPMIN) GO TO 1290
C     RITZ VECTOR WAS COMPUTED
      WRITE(12,1260) J, GOODEV(J), MP(J)
 1260 FORMAT(I6,4X,E20.12,I6,' J, AB-EIGENVAL, MP(J)')
C
      WRITE(12,1270) ERR(J),ERRDGP(J)
 1270 FORMAT(2E15.5,'= NORM(A*Z-EVAL*B*Z), NORM(A*Z-EVAL*B*Z)/ABMINGAP')
C
      WRITE(12,1280) (RITVEC(LL), LL=LINT,LFIN)
 1280 FORMAT(4E20.12)
      GO TO 1310
C     NO RITZ VECTOR WAS COMPUTED FOR THIS EIGENVALUE
 1290 WRITE(12,1300) J,GOODEV(J),MP(J)
 1300 FORMAT(I6,4X,E20.12,I6,' J,AB-EIGVALUE,NO RITZ VECTOR COMPUTED')
C
 1310 CONTINUE
C
C     DID ANY T-MATRICES INCLUDE OFF-DIAGONAL ENTRIES SMALLER THAN
C     DESIRED, AS SPECIFIED BY BTOL?
C
      IF(IB.GT.0) GO TO 1340
C
      WRITE(6,1320) KMAXU
 1320 FORMAT(/' FOR LARGEST T-MATRIX CONSIDERED',I7,' CHECK THE SIZE OF
     1BETAS')
C
C-----------------------------------------------------------------------
C
      CALL TNORM(ALPHA,BETA,BKMIN,TEMP,KMAXU,IBMT)
C
C-----------------------------------------------------------------------
C
      IF(IBMT.LT.0) WRITE (6,1330)
 1330 FORMAT(/' WARNING THE T-MATRICES FOR ONE OR MORE OF THE EIGENVALUE
     1S CONSIDERED'/' HAD AN OFF-DIAGONAL ENTRY THAT WAS SMALLER THAN TH
     1E BETA TOLERANCE THAT WAS SPECIFIED'/)
 1340 CONTINUE
C
       GO TO 1590
C
 1350 WRITE(6,1360) NGOOD,NMAX,MDIMRV
 1360 FORMAT(/I4,' RITZ VECTORS WERE REQUESTED BUT THE REQUIRED DIMENSIO
     1N',I6/' IS LARGER THAN THE USER-SPECIFIED DIMENSION OF RITVEC',I6
     1/' THEREFORE, THE EIGENVECTOR PROCEDURE TERMINATES FOR THE USER TO
     1 INTERVENE')
C
      GO TO 1590
C
 1370 WRITE(6,1380) NOLD,N,MATA,MATNOA,MATB,MATNOB
 1380 FORMAT(/' PARAMETERS READ FROM FILE 3 DO NOT AGREE WITH USER-SPECI
     1FIED VALUES'/' NOLD,N,MATA,MATNOA,MATB,MATNOB = '/2I6,4I12/
     1' THEREFORE PROGRAM TERMINATES FOR USER TO RESOLVE DIFFERENCES'/)
C
      GO TO 1590
C
```

```
 1390 WRITE(6,1400)                                                     LGV11460
 1400 FORMAT(/' PARAMETERS IN ALPHA,BETA FILE READ IN DO NOT AGREE WITH LGV11470
     1THOSE'/' SPECIFIED BY THE USER.  THEREFORE PROGRAM TERMINATES FOR'LGV11480
     1/' USER TO RESOLVE DIFFERENCES'/)                                 LGV11490
C                                                                       LGV11500
      GO TO 1590                                                        LGV11510
C                                                                       LGV11520
 1410 WRITE(6,1420) KMAX,MEV                                            LGV11530
 1420 FORMAT(/' ALPHA,BETA HEADER HAS KMAX = ',I6/                      LGV11540
     1' BUT EIGENVALUES WERE COMPUTED AT MEV = ',I6,' PROGRAM STOPS'/)  LGV11550
C                                                                       LGV11560
      GO TO 1590                                                        LGV11570
C                                                                       LGV11580
 1430 WRITE(6,1440)                                                     LGV11590
 1440 FORMAT(/' PROGRAM COMPUTED 1ST GUESSES AT T-MATRIX SIZES AND READ LGV11600
     1THEM TO FILE 10'/' THEN TERMINATED AS REQUESTED.')                LGV11610
      GO TO 1590                                                        LGV11620
C                                                                       LGV11630
 1450 WRITE(6,1460) MTOL, MDIMTV                                        LGV11640
 1460 FORMAT(/' PROGRAM TERMINATES BECAUSE THE TVEC DIMENSION ANTICIPATELGV11650
     1D',I7/'  IS LARGER THAN THE TVEC DIMENSION',I7,' SPECIFIED BY THE LGV11660
     1USER.'/'  USER MAY RESET THE TVEC DIMENSION AND RESTART THE PROGRALGV11670
     1M')                                                               LGV11680
      GO TO 1590                                                        LGV11690
C                                                                       LGV11700
 1470 WRITE(6,1480)                                                     LGV11710
 1480 FORMAT(/' PROGRAM TERMINATES BECAUSE NO SUITABLE T-EIGENVECTORS WELGV11720
     1RE IDENTIFIED'/'  FOR ANY OF THE EIGENVALUES SUPPLIED.  PROBLEM COLGV11730
     1ULD BE CAUSED'/'  BY TOO SMALL A TVEC DIMENSION OR SIMPLY THAT SUILGV11740
     1TABLE T-VECTORS COULD'/'  NOT BE IDENTIFIED.  USER SHOULD CHECK OULGV11750
     1TPUT'/)                                                           LGV11760
      GO TO 1590                                                        LGV11770
C                                                                       LGV11780
 1490 WRITE(6,1500) LVCONT,NTVEC,NGOOD                                  LGV11790
 1500 FORMAT(/' LVCONT FLAG =',I2,' AND NUMBER ',I5,' OF T-EIGENVECTORS LGV11800
     1 COMPUTED N.E.'/' NUMBER',I5,' REQUESTED SO PROGRAM TERMINATES'/) LGV11810
      GO TO 1590                                                        LGV11820
C                                                                       LGV11830
 1510 WRITE(6,1520)                                                     LGV11840
 1520 FORMAT(/' PROGRAM TERMINATES WITHOUT COMPUTING RITZ VECTORS'/     LGV11850
     1' BECAUSE ALL T-EIGENVECTORS WERE REJECTED AS NOT SUITABLE FOR THELGV11860
     1 RITZ VECTOR'/' COMPUTATIONS.  PROBABLE CAUSE IS LACK OF CONVERGENLGV11870
     1CE OF THE EIGENVALUES SUPPLIED'/)                                 LGV11880
       GO TO 1590                                                       LGV11890
C                                                                       LGV11900
 1530 WRITE(6,1540)                                                     LGV11910
 1540 FORMAT(/' PROGRAM INDICATES THAT IT IS NOT POSSIBLE TO COMPUTE ANYLGV11920
     1 OF THE'/' REQUESTED EIGENVECTORS. THEREFORE PROGRAM TERMINATES') LGV11930
      DO 1550 J=1,NGOODC                                                LGV11940
 1550 WRITE(6,1560)  J,GOODEV(J),MP(J)                                  LGV11950
 1560 FORMAT(/4X,' J',9X,'AB-EIGENVALUE',4X,'MP(J)'/I6,E20.12,I9)       LGV11960
      GO TO 1590                                                        LGV11970
C                                                                       LGV11980
 1570 WRITE(6,1580) MBETA,KMAXN                                         LGV11990
 1580 FORMAT(/' PROGRAM TERMINATES BECAUSE THE STORAGE ALLOTTED FOR THE LGV12000
```

```
     1BETA ARRAY',I8/' IS NOT SUFFICIENT FOR THE ENLARGED KMAX =',I8,' TLGV12010
     1HAT THE PROGRAM WANTS'/' USER CAN ENLARGE THE ALPHA AND BETA ARRAYLGV12020
     1S AND RERUN THE PROGRAM'/)                                       LGV12030
C                                                                       LGV12040
 1590 CONTINUE                                                          LGV12050
C                                                                       LGV12060
      STOP                                                              LGV12070
C-----END OF MAIN PROGRAM FOR LANCZOS EIGENVECTOR COMPUTATIONS----------LGV12080
      END                                                               LGV12090
```

5.4 LGMULT: LANCZS and Sample Matrix-Vector Multiply Subnroutines

```
C-----LGMULT------------------------------------------------------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C         REFERENCE: Cullum and Willoughby, Chapters 1,2,3,4
C         Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C         Applied Mathematics, 2002. SIAM Publications,
C         Philadelphia, PA. USA
C
C
C     CONTAINS SUBROUTINES  LANCZS, USPECA, USPECB, AMATV, AND LSOLV.
C     TO BE USED WITH THE LANCZOS CODES FOR THE GENERALIZED EIGENVALUE
C     PROBLEM, A*X = EVAL*B*X, WHERE A AND B ARE REAL SYMMETRIC, AND
C     B IS POSITIVE DEFINITE WITH ITS CHOLESKY FACTORS AVAILABLE.
C
C     NONPORTABLE CONSTRUCTIONS:
C     1.  THE ENTRY MECHANISM USED TO PASS THE STORAGE
C         LOCATIONS OF THE USER-SPECIFIED MATRICES FROM THE
C         SUBROUTINES USPECA AND USPECB TO THE MATRIX-VECTOR
C         SUBROUTINE, AMATV AND TO THE SOLVE SUBROUTINE, LSOLV.
C     2.  IN SAMPLE USPECA AND USPECB:  FREE FORMAT (8,*); FORMAT
C         (20A4), AND FORMAT (4Z20).
C
C-----LANCZS-COMPUTE LANCZOS TRIDIAGONAL MATRICES-----------------------
C
      SUBROUTINE LANCZS(LSOLV,MATVEC,ALPHA,BETA,V1,V2,VS,G,KMAX,MOLD1,N,
     1 IIX)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  ALPHA(1), BETA(1), V1(1), V2(1), VS(1)
      DOUBLE PRECISION SUM, ONE, ZERO, TEMP
      REAL  G(1)
      DOUBLE PRECISION  FINPRO,DSQRT
      EXTERNAL MATVEC, LSOLV
```

```
C-----------------------------------------------------------------------
C     ALPHA, BETA, AND LANCZOS VECTOR GENERATION
C     ALPHA BETA GENERATION STARTS WITH IVEC = 1, BETA(1) = ZERO,
C     V2 =  RANDOM VECTOR WITH UNIT B-NORM, VS = B*V2, AND V1 = 0.;
C     OR STARTS WITH AN EXISTING ALPHA/BETA FILE AND THE MOST
C     RECENTLY GENERATED V2, VS, AND V1.
C
      ZERO = 0.0D0
      ONE  = 1.0D0
      IF (MOLD1.GT.1) GO TO 40
      BETA(1) = ZERO
      IIL = IIX
C
C-----------------------------------------------------------------------
      CALL GENRAN(IIL,G,N)
C-----------------------------------------------------------------------
C
      DO 10 K = 1,N
   10 V2(K) = G(K)
C
C-----------------------------------------------------------------------
C     COMPUTE L-TRANSPOSE*V2 AND ITS NORM
      ISOLV = 2
      CALL LSOLV(V2,VS,ISOLV)
      SUM = FINPRO(N,VS(1),1,VS(1),1)
C-----------------------------------------------------------------------
C
C     NORMALIZE STARTING VECTORS:  (V2-TRANSPOSE*B*V2) = 1
      SUM = ONE/DSQRT(SUM)
      DO 20 K = 1,N
      VS(K) = SUM*VS(K)
   20 V2(K) = SUM*V2(K)
C
C-----------------------------------------------------------------------
C     INITIALIZE V1 = B*V2 = L*VS
      ISOLV = 1
      CALL LSOLV(VS,V1,ISOLV)
C-----------------------------------------------------------------------
C
      DO 30 K = 1,N
      VS(K) = V1(K)
   30 V1(K) = 0.D0
   40 CONTINUE
C
C     INITIALIZATIONS ARE:  VS = B*V(I),  V1 = B*V(I-1),  V2 = V(I)
C
      DO 80 IVEC = MOLD1,KMAX
      SUM = BETA(IVEC)
C
C-----------------------------------------------------------------------
C     COMPUTE V1 = A*V2 - SUM*V1
      CALL MATVEC(V2,V1,SUM)
C     COMPUTE ALPHA(I)
      SUM = FINPRO(N,V1(1),1,V2(1),1)
C-----------------------------------------------------------------------
```

```
C
      ALPHA(IVEC) = SUM
      DO 50 K = 1,N
   50 V1(K) = V1(K)-SUM*VS(K)
C
C     SET V1 = B*V(IVEC) AND VS = BETA(IVEC+1)*B*V(IVEC+1)
      DO 60 K = 1,N
      TEMP = V1(K)
      V1(K) = VS(K)
   60 VS(K) = TEMP
C
C-----------------------------------------------------------------------
C     COMPUTE V2 = (L-INVERSE)*VS
      ISOLV = 3
      CALL LSOLV(VS,V2,ISOLV)
C     COMPUTE BETA(IVEC+1)
      SUM = FINPRO(N,V2(1),1,V2(1),1)
C-----------------------------------------------------------------------
C
      IN = IVEC+1
      BETA(IN) = DSQRT(SUM)
C
C-----------------------------------------------------------------------
      ISOLV = 4
      CALL LSOLV(V2,V2,ISOLV)
C-----------------------------------------------------------------------
C
      SUM = ONE/BETA(IN)
      DO 70 K = 1,N
      V2(K) = SUM*V2(K)
   70 VS(K) = SUM*VS(K)
C
   80 CONTINUE
C
      RETURN
C-----END LANCZS--------------------------------------------------------
      END
C
C-----USPEC (GENERAL SYMMETRIC SPARSE MATRICES)-------------------------
C
C     SUBROUTINE USPECA(N,MATNOA)
      SUBROUTINE GUSPEC(N,MATNOA)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  ASD(10000),AD(5010)
      INTEGER  IROW(10000),ICOL(5010)
C-----------------------------------------------------------------------
C    USPEC DIMENSIONS AND INITIALIZES THE ARRAYS NEEDED TO DEFINE
C    THE USER-SPECIFIED A-MATRIX AND THEN PASSES THE STORAGE LOCATIONS
C    OF THESE ARRAYS TO THE MULTIPLY SUBROUTINE AMATV.
C
C    MATRIX IS STORED IN FOLLOWING SPARSE MATRIX FORMAT:
C    N = ORDER OF A-MATRIX,
C    NZS = NUMBER OF NONZERO SUBDIAGONAL ENTRIES,
C    NZL = INDEX OF LAST COLUMN CONTAINING NONZERO SUBDIAGONAL ENTRIES,
```

```
C    ICOL(J), J=1,NZL IS THE NUMBER OF NONZERO SUBDIAGONAL ELEMENTS
C             IN COLUMN J.
C    IROW(K), K = 1,NZS IS THE CORRESPONDING ROW INDEX FOR ASD(K).
C    AD(I), I=1,N CONTAINS DIAGONAL ENTRIES (INCLUDING ANY 0
C           DIAGONAL ENTRIES).
C    ASD(K), K=1,NZS CONTAINS NONZERO SUBDIAGONAL ENTRIES, BY COLUMN
C    FOR J > NZL THERE ARE NO NONZERO SUBDIAGONAL ELEMENTS IN COLUMN J.
C    ICOL(J) = 0 IS ALLOWED
C
C-----------------------------------------------------------------------
C     ARRAYS THAT DEFINE THE A-MATRIX ARE READ IN FROM FILE 8.  NOTE
C     THAT IF THE B-MATRIX IS PERMUTED, THEN LANCZOS PROGRAM ASSUMES
C     THAT THE DATA ON FILE 8 CORRESPONDS TO THE CORRESPONDING
C     PERMUTED A-MATRIX.  LANCZOS PROCEDURE WORKS DIRECTLY WITH THE
C     PERMUTED MATRICES.  EIGENVECTOR CODE, LGVEC, THEN PERMUTES THE
C     COMPUTED EIGENVECTORS TO GET THOSE CORRESPONDING TO THE ORIGINAL
C     MATRICES.
C
      READ(8,10) NZS,NOLD,NZL,MATOLD
   10 FORMAT(I10,2I6,I8)
C
      WRITE(6,20) NZS,NOLD,NZL,MATOLD
   20 FORMAT(I10,2I6,I8,' = NZS,NOLD,NZL,MATOLD'/)
C
C     TEST OF PARAMETER CORRECTNESS
      ITEMP = (NOLD-N)**2 + (MATNOA-MATOLD)**2
C
      IF(ITEMP.EQ.0) GO TO 40
C
      WRITE(6,30)
   30 FORMAT(' PROGRAM TERMINATES BECAUSE EITHER ORDERS OF OR LABELS FOR
     1 MATRIX DISAGREE')
      GO TO 70
C
   40 CONTINUE
C
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS READ
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS READ
      READ(8,50) (ICOL(K), K=1,NZL)
      READ(8,50) (IROW(K), K=1,NZS)
   50 FORMAT(13I6)
C
C     DIAGONAL IS READ FIRST, THEN NONZERO BELOW DIAGONAL ENTRIES
      READ(8,60) (AD(K), K=1,N)
      READ(8,60) (ASD(K), K=1,NZS)
   60 FORMAT(4E19.10)
C
C-----------------------------------------------------------------------
C     PASS STORAGE LOCATIONS OF ARRAYS THAT DEFINE THE A-MATRIX TO
C     THE MATRIX-VECTOR MULTIPLY SUBROUTINE AMATV
      CALL AMATVE(ASD,AD,ICOL,IROW,N,NZL)
C-----------------------------------------------------------------------
C
      RETURN
   70 STOP
```

```
C-----END OF USPECA-----------------------------------------------------LGM02100
      END                                                               LGM02110
C                                                                       LGM02120
C-----USPECB FOR CHOLESKY FACTORS OF GENERAL SPARSE SYMMETRIC MATRIX----LGM02130
C                                                                       LGM02140
C     SUBROUTINE USPECB(N,MATNOB)                                       LGM02150
      SUBROUTINE CUSPEC(N,MATNOB)                                       LGM02160
C                                                                       LGM02170
C-----------------------------------------------------------------------LGM02180
      DOUBLE PRECISION  BD(2200),BSD(10000)                             LGM02190
      INTEGER  KCOL(2200),KROW(10000),IPR(2200),IPT(2200)               LGM02200
C-----------------------------------------------------------------------LGM02210
C     DIMENSIONS ARRAYS NEEDED TO DEFINE CHOLESKY FACTOR OF B-MATRIX,   LGM02220
C     READS CHOLESKY FACTOR FROM FILE 7, AND THEN PASSES STORAGE        LGM02230
C     LOCATIONS OF THESE ARRAYS TO THE MATRIX SOLVE SUBROUTINE LSOLV    LGM02240
C                                                                       LGM02250
C     THE LANCZOS PROCEDURE LGVAL WILL USE THE CHOLESKY FACTORS ON      LGM02260
C     FILE 7.  THESE FACTORS MAY CORRESPOND TO A PERMUTED VERSION OF    LGM02270
C     THE GIVEN B-MATRIX IN WHICH CASE THIS PERMUTATION WILL BE STORED  LGM02280
C     IN IPR. THE ITH ROW OF THE PERMUTED B WILL CORRESPOND TO THE      LGM02290
C     JTH ROW OF B WHERE J = IPR(I) AND I = IPT(J).  IF B IS            LGM02300
C     PERMUTED, THE LANCZOS PROCEDURE ASSUMES THAT THE USER-PROVIDED    LGM02310
C     A-MATRIX IS IN FACT, THE CORRESPONDING PERMUTED VERSION OF THE    LGM02320
C     ORIGINAL A-MATRIX.                                                LGM02330
C                                                                       LGM02340
C     THE CHOLESKY FACTOR IS STORED IN THE FOLLOWING SPARSE FORMAT:     LGM02350
C     N = ORDER OF THE B-MATRIX.                                        LGM02360
C     NZT = NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN THE CHOLESKY       LGM02370
C           FACTOR, L.                                                  LGM02380
C     KCOL(J), J=1,N IS THE NUMBER OF NONZERO SUBDIAGONAL ELEMENTS IN   LGM02390
C              COLUMN J OF L.                                           LGM02400
C     KROW(K), K=1,NZT IS THE ROW INDEX FOR CORRESPONDING ENTRY BSD(K). LGM02410
C     BD(J), J = 1,N CONTAINS THE DIAGONAL ENTRIES OF L.                LGM02420
C     BSD(K), K =1,NZT CONTAINS THE NONZERO SUBDIAGONAL ENTRIES OF L    LGM02430
C             BY COLUMN.                                                LGM02440
C-----------------------------------------------------------------------LGM02450
C                                                                       LGM02460
C     READ CHOLESKY FACTOR FROM FILE 7.  MUST BE STORED                 LGM02470
C     IN SPARSE MATRIX FORMAT.                                          LGM02480
      READ(7,10) NZT,NOLD,NZL,MATOLD,JPERM                              LGM02490
   10 FORMAT(I10,2I6,I8,I6)                                             LGM02500
C                                                                       LGM02510
      WRITE(6,20) NZT,NZL,N,NOLD,MATOLD,JPERM                           LGM02520
   20 FORMAT(' HEADER, CHOLESKY FACTOR FILE'/                           LGM02530
     1 3X,'NZT',3X,'NZL',5X,'N',2X,'NOLD',2X,'MATOLD',1X,'JPERM'/       LGM02540
     1 4I6,I8,I6/)                                                      LGM02550
C                                                                       LGM02560
      IF (N.NE.NOLD.OR.MATNOB.NE.MATOLD) GO TO 70                       LGM02570
C                                                                       LGM02580
      READ(7,30) (KCOL(K), K = 1,NZL)                                   LGM02590
      READ(7,30) (KROW(K), K = 1,NZT)                                   LGM02600
   30 FORMAT(13I6)                                                      LGM02610
      READ(7,40) (BD(K), K = 1,N)                                       LGM02620
      READ(7,40) (BSD(K), K = 1,NZT)                                    LGM02630
   40 FORMAT(4Z20)                                                      LGM02640
```

```
C  20 FORMAT(3E25.16)
C
      IF(JPERM.EQ.0)  GO TO 60
C
      READ(7,30) (IPR(K), K = 1,N)
      DO 50 K = 1,N
      J = IPR(K)
   50 IPT(J) = K
C
C-----------------------------------------------------------------------
      CALL LPERME(IPR,IPT,N)
C-----------------------------------------------------------------------
C
   60 CONTINUE
C
C-----------------------------------------------------------------------
C     PASS STORAGE LOCATIONS OF FACTORS TO SUBROUTINE LSOLV
      CALL LSOLVE(BSD,BD,KCOL,KROW,N,NZT,NZL)
C-----------------------------------------------------------------------
C
      GO TO 90
C
   70 CONTINUE
C     DEFAULT EXIT
      WRITE(6,80) MATNOB,MATOLD
   80 FORMAT(' TERMINATE.  PARAMETERS IN CHOLESKY FACTOR FILE'/
     1' DO NOT AGREE WITH THOSE SPECIFIED BY THE USER'/
     1' MATNOB = ',I8,' MATOLD = ',I8/)
      STOP
C
   90 CONTINUE
C-----END OF USPECB----------------------------------------------------
      RETURN
      END
C
C-----MATRIX-VECTOR MULTIPLY FOR REAL SPARSE SYMMETRIC MATRICES---------
C
C     SUBROUTINE AMATV(W,U,SUM)
      SUBROUTINE GCMATV(W,U,SUM)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION   U(1),W(1),ASD(1),AD(1),SUM
      INTEGER  IROW(1),ICOL(1)
C-----------------------------------------------------------------------
C    SPARSE MATRIX-VECTOR MULTIPLY FOR LANCZS  U = A*W - SUM*U
C    SEE USPECA SUBROUTINE FOR DESCRIPTION OF THE ARRAYS
C    THAT DEFINE THE A-MATRIX
C-----------------------------------------------------------------------
      GO TO 3
C     STORAGE LOCATIONS OF ARRAYS ARE PASSED TO AMATV FROM USPECA
      ENTRY AMATVE(ASD,AD,ICOL,IROW,N,NZL)
      GO TO 4
C-----------------------------------------------------------------------
C
C     COMPUTE THE DIAGONAL TERMS
```

```
    3 DO 10 I = 1,N                                                     LGM03200
   10 U(I) = AD(I)*W(I)-SUM*U(I)                                        LGM03210
C                                                                       LGM03220
C     COMPUTE BY COLUMN                                                 LGM03230
      LLAST = 0                                                         LGM03240
      DO 30 J = 1,NZL                                                   LGM03250
C                                                                       LGM03260
      IF (ICOL(J).EQ.0) GO TO 30                                        LGM03270
      LFIRST = LLAST + 1                                                LGM03280
      LLAST = LLAST + ICOL(J)                                           LGM03290
C                                                                       LGM03300
      DO 20 L = LFIRST,LLAST                                            LGM03310
      I = IROW(L)                                                       LGM03320
C                                                                       LGM03330
      U(I) = U(I) + ASD(L)*W(J)                                         LGM03340
      U(J) = U(J) + ASD(L)*W(I)                                         LGM03350
C                                                                       LGM03360
   20 CONTINUE                                                          LGM03370
C                                                                       LGM03380
   30 CONTINUE                                                          LGM03390
C                                                                       LGM03400
    4 RETURN                                                            LGM03410
C                                                                       LGM03420
C-----END OF AMATV----------------------------------------------------- LGM03430
      END                                                               LGM03440
C                                                                       LGM03450
C-----LSOLV-GENERAL SPARSE, POSITIVE DEFINITE B-MATRIX------------------LGM03460
C     (USES THE CHOLESKY FACTORS OF B, B = L*(L-TRANSPOSE))             LGM03470
C                                                                       LGM03480
      SUBROUTINE TLSOLV(W,U,ISOLV)                                      LGM03490
C     SUBROUTINE LSOLV(W,U,ISOLV)                                       LGM03500
C                                                                       LGM03510
C-----------------------------------------------------------------------LGM03520
      DOUBLE PRECISION  U(1),W(1),BD(1),BSD(1), TEMP                    LGM03530
      INTEGER  KCOL(1),KROW(1)                                          LGM03540
C-----------------------------------------------------------------------LGM03550
C     SUBROUTINE HAS 4 BRANCHES:  ISOLV = (1,2,3,4)  CALCULATES         LGM03560
C     ISOLV = 1   U = L*W                                               LGM03570
C     ISOLV = 2   U = L'*W                                              LGM03580
C     ISOLV = 3   SOLVE FOR U IN  L*U = W                               LGM03590
C     ISOLV = 4   SOLVE FOR U IN  L'*U = W                              LGM03600
C-----------------------------------------------------------------------LGM03610
      GO TO 3                                                           LGM03620
      ENTRY LSOLVE(BSD,BD,KCOL,KROW,N,NZT,NZL)                          LGM03630
      GO TO 4                                                           LGM03640
C-----------------------------------------------------------------------LGM03650
    3 GO TO (10,50,80,120), ISOLV                                       LGM03660
C                                                                       LGM03670
C     ISOLV = 1,  U=L*W                                                 LGM03680
   10 CONTINUE                                                          LGM03690
      KL = 0                                                            LGM03700
      DO 20 K = 1,N                                                     LGM03710
   20 U(K) = W(K)*BD(K)                                                 LGM03720
      DO 40 K = 1,N                                                     LGM03730
      TEMP = W(K)                                                       LGM03740
```

```
      IF (KCOL(K).EQ.0.OR.K.EQ.N) GO TO 40
      KF = KL + 1
      KL = KL + KCOL(K)
      DO 30 KK = KF,KL
      KR = KROW(KK)
   30 U(KR) = U(KR) + TEMP*BSD(KK)
   40 CONTINUE
      GO TO 150
C
C     ISOLV = 2,  U = (L-TRANSPOSE)*W
   50 CONTINUE
      KL = 0
      DO 70 J = 1,N
      TEMP = W(J)*BD(J)
      IF (KCOL(J).EQ.0.OR.J.EQ.N) GO TO 70
      KF = KL + 1
      KL  = KL + KCOL(J)
      DO 60 K = KF,KL
      IK = KROW(K)
   60 TEMP = BSD(K)*W(IK) + TEMP
   70 U(J) = TEMP
      GO TO 150
C
C     ISOLV = 3,  U = (L-INVERSE)*W
   80 CONTINUE
      DO 90 K = 1,N
   90 U(K) = W(K)
      KL = 0
      DO 110 K = 1,N
      TEMP = U(K)/BD(K)
      U(K) = TEMP
      IF (KCOL(K).EQ.0.OR.K.EQ.N) GO TO 110
      KF = KL + 1
      KL = KL + KCOL(K)
      DO 100 KK = KF,KL
      KR = KROW(KK)
  100 U(KR) = U(KR) - TEMP*BSD(KK)
  110 CONTINUE
      GO TO 150
C
C     ISOLV = 4,  U = (L-TRANSPOSE)-INVERSE*W
  120 CONTINUE
      NP1 = N+1
      KF = NZT + 1
      DO 140 K = 1,N
      L = NP1 - K
      TEMP = W(L)
      IF (KCOL(L).EQ.0.OR.L.EQ.N) GO TO 140
      KL = KF - 1
      KF = KF - KCOL(L)
      DO 130 LL = KF,KL
      LR = KROW(LL)
  130 TEMP = TEMP - BSD(LL)*U(LR)
  140 U(L) = TEMP/BD(L)
      GO TO 150
```

```
  150 CONTINUE
C
    4 RETURN
C
C-----END OF LSOLV-------------------------------------------------
      END
C
C-----START OF USPEC FOR DIAGONAL TEST A-MATRIX-------------------------
C
      SUBROUTINE USPECA(N,MATNO)
C     SUBROUTINE DUSPEC(N,MATNO)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  D(1000), SPACE, SHIFT
      DOUBLE PRECISION  DABS, DFLOAT
      REAL  EXPLAN(20)
C-----------------------------------------------------------------------
C
      READ(8,10) EXPLAN
   10 FORMAT(20A4)
      READ(8,*) NOLD,NUNIF,SPACE,D(1),SHIFT
      NNUNIF = NOLD - NUNIF
      WRITE(6,20) NOLD,SPACE,NNUNIF,D(1),SHIFT
   20 FORMAT(/' DIAGONAL TEST A-MATRIX, SIZE = ',I4/' MOST ENTRIES ARE '
     1,E10.3,' UNITS APART.',I3,' ENTRIES'/' ARE IRREGULARLY SPACED. FIR
     1ST ENTRY IS ',E10.3,' SHIFT = ',E10.3/)
C
      IF(N.NE.NOLD) GO TO 90
C     COMPUTE THE UNIFORM PORTION OF THE SPECTRUM
      DO 30 J=2,NUNIF
   30 D(J) = D(1) - DFLOAT(J-1)*SPACE
      NUNIF1=NUNIF + 1
      READ(8,10)  EXPLAN
      DO 40 J=NUNIF1,N
   40 READ(8,*) D(J)
      NB = NUNIF - 2
C
      IF(SHIFT.EQ.0.) GO TO 60
      DO 50 J=1,N
   50 D(J) = D(J) + SHIFT
C
C     PRINT OUT A-MATRIX
   60 WRITE(6,70) (D(I), I=1,10 )
      WRITE(6,80) (D(I), I = NB,N)
   70 FORMAT(/' GENERALIZED LANCZOS TEST, 1ST 10 ENTRIES OF DIAGONAL A-M
     1ATRIX = '/(3E22.14))
   80 FORMAT(/' MIDDLE UNIFORM PORTION OF MATRIX IS NOT PRINTED OUT'/
     1' END OF UNIFORM PLUS NONUNIFORM SECTION = '/(3E25.16))
C
C     DIAGONAL GENERATION COMPLETE
C
C-----------------------------------------------------------------------
C    CALL ENTRY TO MATRIX-VECTOR MULTIPLY SUBROUTINE  TO PASS
C    STORAGE LOCATION OF D-ARRAY AND ORDER OF A-MATRIX.
       CALL MVDIAE(D,N)
```

```
C-----------------------------------------------------------------------
C
      RETURN
   90 WRITE(6,100) NOLD,N
  100 FORMAT(' PROGRAM TERMINATES BECAUSE NOLD = ',I5,'DOES NOT EQUAL N
     1 =',I5)
C-----END OF USPECA SUBROUTINE FOR DIAGONAL TEST MATRICES---------------
      STOP
      END
C
C-----USPECB--DIAGONAL TEST B-MATRIX------------------------------------
C
      SUBROUTINE USPECB(N,MATNO)
C     SUBROUTINE USPECD(N,MATNO)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  D(1000), DS(1000), SPACE, SHIFT
      DOUBLE PRECISION  DFLOAT, DSQRT
      REAL  EXPLAN(20)
C-----------------------------------------------------------------------
C
      READ(7,10) EXPLAN
   10 FORMAT(20A4)
      READ(7,*) NOLD,NUNIF,SPACE,D(1),SHIFT
      NNUNIF = NOLD - NUNIF
      WRITE(6,20) NOLD,SPACE,NNUNIF,D(1),SHIFT
   20 FORMAT(/' DIAGONAL TEST B-MATRIX, SIZE = ',I4/' MOST ENTRIES ARE '
     1,E10.3,' UNITS APART.',I3,' ENTRIES'/' ARE IRREGULARLY SPACED. FIR
     1ST ENTRY IS ',E10.3,' SHIFT = ',E10.3/)
C
      IF(N.NE.NOLD) GO TO 100
C     COMPUTE THE UNIFORM PORTION OF THE SPECTRUM
      DO 30 J=2,NUNIF
   30 D(J) = D(1) - DFLOAT(J-1)*SPACE
      NUNIF1=NUNIF + 1
      READ(7,10)  EXPLAN
      DO 40 J=NUNIF1,N
   40 READ(7,*) D(J)
      NB = NUNIF - 2
C
      IF(SHIFT.EQ.0.) GO TO 60
      DO 50 J=1,N
   50 D(J) = D(J) + SHIFT
C
C     PRINT OUT B-MATRIX
   60 WRITE(6,70) (D(I), I=1,10 )
      WRITE(6,80) (D(I), I = NB,N)
   70 FORMAT(/' GENERALIZED LANCZOS TEST, 1ST 10 ENTRIES OF DIAGONAL B-M
     1ATRIX = '/(3E22.14))
   80 FORMAT(/' MIDDLE UNIFORM PORTION OF MATRIX IS NOT PRINTED OUT'/
     1' END OF UNIFORM PLUS NONUNIFORM SECTION = '/(3E25.16))
C
C     DIAGONAL GENERATION COMPLETE
C
      DO 90 K = 1,N
```

```
   90 DS(K) = DSQRT(D(K))
C
C-----------------------------------------------------------------------
C    PASS STORAGE LOCATION OF THE L-FACTOR (THE DS-ARRAY) AND ORDER OF
C    B-MATRIX TO LSOLV SUBROUTINE.
        CALL DSOLVE(DS,N)
C-----------------------------------------------------------------------
C
      RETURN
  100 WRITE(6,110) NOLD,N
  110 FORMAT(' PROGRAM TERMINATES BECAUSE NOLD = ',I5,'DOES NOT EQUAL N
     1 =',I5)
C-----END OF USPECB SUBROUTINE FOR DIAGONAL TEST MATRICES---------------
      STOP
      END
C
C-----MATRIX-VECTOR MULTIPLY FOR DIAGONAL TEST MATRICES-----------------
C
      SUBROUTINE AMATV(W,U,SUM)
C     SUBROUTINE DCMATV(W,U,SUM)
C
C     AMATV COMPUTES  U = (DIAGONAL MATRIX) * W - SUM * U
C-----------------------------------------------------------------------
      DOUBLE PRECISION  W(1),U(1),D(1),SUM
C-----------------------------------------------------------------------
      GO TO 3
      ENTRY MVDIAE(D,N)
      GO TO 4
C-----------------------------------------------------------------------
C
    3 DO 10 I=1,N
   10 U(I)= D(I)*W(I) - SUM*U(I)
C
    4 RETURN
C
C-----END OF DIAGONAL TEST MATRIX  MULTIPLY-----------------------------
      END
C
C-----LSOLV FOR DIAGONAL MATRIX-----------------------------------------
C
      SUBROUTINE LSOLV(W,U,ISOLV)
C     SUBROUTINE DSOLV(W,U,ISOLV)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION U(1), W(1),  DS(1)
C-----------------------------------------------------------------------
      GO TO 3
      ENTRY DSOLVE(DS,N)
      GO TO 4
C-----------------------------------------------------------------------
    3 GO TO (10,30,50,70), ISOLV
C
C     ISOLV = 1
   10 CONTINUE
      DO 20 K = 1,N
```

```
   20 U(K) = DS(K)*W(K)
      GO TO 90
C
C      ISOLV = 2
   30 CONTINUE
      DO 40 K = 1,N
   40 U(K) = DS(K)*W(K)
      GO TO 90
C
C      ISOLV = 3
   50 CONTINUE
      DO 60 K = 1,N
   60 U(K) = W(K)/DS(K)
      GO TO 90
C
C      ISOLV = 4
   70 CONTINUE
      DO 80 K = 1,N
   80 U(K) = W(K)/DS(K)
C
   90 CONTINUE
C
    4 RETURN
C
C-----END OF DSOLV------------------------------------------------------
      END
```

5.5 LGVAL: LGVEC: File Definitions, Sample Input Files

Below is a listing of the input/output files which are accessed by the Lanczos eigenvalue program LGVAL for real symmetric generalized problems where one of the two matrices is positive definite. Included also is a sample of the input file which LGVAL requires on file 5. The parameters in this file are supplied in free format. LGVAL computes eigenvalues of the matrix eigenvalue problem $Ax = \lambda Bx$ on user-specified intervals. It is assumed that A and B are real symmetric matrices and that B is positive definite. The program uses Cholesky Factor L of $B = LL^T$.

```
Sample Specification of Input/Output Files for LGVAL
----------------------------------------------------------------
 LGVAL EXEC LANCZOS EIGENVALUE CALCULATION AX = EV*BX CASE
FI 06 TERM
FILEDEF  1 DISK &1        NHISTORY  A (RECFM F LRECL 80 BLOCK 80
FILEDEF  2 DISK &1        HISTORY   A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1        GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1        ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK LGVAL     INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  7 DISK &1        LDATA     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1        ADATA     A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1        DISTINCT  A (RECFM F LRECL 80 BLOCK 80
LOAD  LGVAL   LESUB  LGMULT
----------------------------------------------------------------
```

```
Sample Input File for LGVAL
-------------------------------------------------------------------
 LGVAL INPUT LANCZOS EIGENVALUE COMPUTATION, NO REORTHOGONALIZATION
 AX = EV*BX GENERALIZED EIGENVALUE PROBLEM
LINE 1      N    KMAX    NMEVS     MATNOA    MATNOB
         100     300        1        100       100
LINE 2   SVSEED    RHSEED      MXINIT    MXSTUR
       49302312   5731029           5    100000
LINE 3   ISTART     ISTOP
              0         1
LINE 4     IHIS     IDIST  IWRITE
              1         0       1
LINE 5   RELTOL (RELATIVE TOLERANCE IN 'COMBINING' GOODEV)
    .0000000001
LINE 6   MB(1)   MB(2)   MB(3)   MB(4)    (ORDERS OF T(1,MEV) )
           300
LINE 7    NINT     (NUMBER OF SUB-INTERVALS FOR BISEC)
             1
LINE 8    LB(1)    LB(2)   LB(3)   LB(4)  (INTERVAL LOWER BOUNDS)
            1.5
LINE 9    UB(1)    UB(2)   UB(3)   UB(4)  (INTERVAL UPPER BOUNDS)
          2100.
-------------------------------------------------------------------
```

Below is a listing of the input/output files which are accessed by the Lanczos eigenvector program for real symmetric generalized problems, LGVEC. Also included below is a sample of the input file which LGVEC requires on file 5. The parameters in this file are supplied in free format. LGVEC computes eigenvectors for each of a user-specified subset of the eigenvalues computed by the companion program LGVAL.

```
Sample Specifications for the Input/Output Files for LGVEC
-----------------------------------------------------------------------
 LGVEC EXEC TO RUN LANCZOS EIGENVECTOR PROGRAM, REAL SYMMETRIC MATRICES
FI 06 TERM
FILEDEF  2 DISK &1       HISTORY   A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1       GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1       ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK LGVEC    INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  7 DISK &1       LDATA     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1       ADATA     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  9 DISK &1       ERREST    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 10 DISK &1       BOUNDS    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1       TEIGVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 12 DISK &1       RITZVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 13 DISK &1       PAIGE     A (RECFM F LRECL 80 BLOCK 80
LOAD  LGVEC  LESUB  LGMULT
-----------------------------------------------------------------------
```

```
Sample Input File for LGVEC
-----------------------------------------------------------------------
 LGVEC EIGENVECTOR COMPUTATIONS AX = EV*BX NO REORTHOGONALIZATION
LINE 1  MDIMTV    MDIMRV  MBETA (MAX.DIMENSIONS, TVEC, RITVEC AND BETA
         10000     10000   2000
LINE 2       RELTOL
        .0000000001
LINE 3  MBOUND   NTVCON SVTVEC IREAD (FLAGS
            0        1      0     1
LINE 4  TVSTOP   LVCONT ERCONT  IWRITE (FLAGS
            0        1      1       1
LINE 5    RHSEED  (RANDOM GENERATOR SEED FOR STARTING VECTOR IN INVERM)
        45329517
LINE 6  MATNOA     MATNOB      N    JPERM
          100        100    100        0
------------------------------------------------------------------------
```

Chapter 6

Real Rectangular Matrices, Singular Values and Vectors

6.1 Introduction

The FORTRAN codes in this Chapter address the question of computing distinct singular values and corresponding left and right singular vectors of real rectangular matrices, using a single-vector Lanczos procedure. For a given real rectangular mxn matrix A, these codes compute nonnegative scalars σ and corresponding real vectors $x \neq 0$ and $y \neq 0$ such that

$$\begin{aligned} Ax &= \sigma y \\ A^T y &= \sigma x. \end{aligned} \tag{6.1.1}$$

Every real rectangular mxn, $m \geq n$, matrix has a singular value decomposition,

$$A = Y \Sigma X^T, \quad X^T X = I, \quad Y^T Y = I, \quad \Sigma = \begin{bmatrix} \Sigma_1 \\ 0 \end{bmatrix} \tag{6.1.2}$$

where Σ is $m \times n$, and $\Sigma_1 = \mathrm{diag}\{\sigma_1, ..., \sigma_n\}$ with $\sigma_i, 1 \leq i \leq n$, the singular values of A. X is a $n \times n$ orthogonal matrix, Y is a $m \times m$ orthogonal matrix, and the columns of X and of Y are respectively, right and left singular vectors of A. There are many applications for this type of decomposition. Singular values and vectors are discussed in detail for example in Stewart [24].

Using Eqn(6.1.1), it is not difficult to demonstrate that the singular values of a given real matrix A are just the nonnegative square roots of the eigenvalues of the associated real symmetric matrix $A^T A$. Thus from the perturbation theorems for real symmetric matrices, we have that a small perturbation in the given matrix A causes small perturbations in the singular values. The same arguments demonstrate that the right singular vectors of a matrix A are eigenvectors of the matrix $A^T A$, and the left singular vectors are eigenvectors of the matrix AA^T. Therefore, we also have that the perturbation theorems for eigenvectors of real symmetric matrices apply to the singular vectors.

The Lanczos recursion as presented in Eqns(1.2.1) and (1.2.2) is only applicable to real symmetric matrices. Therefore we ask the question: How do we construct a real symmetric matrix which will give us the desired singular values? Obviously, we could just apply the real symmetric Lanczos recursion to $A^T A$. However in general, these matrices are not suitable because of the effects that squaring a matrix can have on the eigenvalues. Small singular values of A which are close together correspond to eigenvalues of $A^T A$ which are smaller and even closer together. Large singular values of A which are far apart correspond to eigenvalues of $A^T A$ which are larger and further apart. When a matrix A has both small and large singular values, dealing numerically with the square of that matrix is difficult. Lanczos [15] suggested the use of an alternative real symmetric matrix. He proposed that the following larger but real symmetric $[m+n] \times [m+n]$ matrix be used.

$$B = \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}. \tag{6.1.3}$$

The relationships between the eigenvalues and the eigenvectors of B and the singular values and singular vectors of A are discussed in detail in Section 5.4 of Chapter 5 in Volume 1.

We could apply the real symmetric version of the Lanczos recursion directly to the matrix B in Eqn(6.1.3). However, because this matrix is considerably larger than the A-matrix, we use a modification of the real symmetric Lanczos recursion which incorporates the following choice of starting vector suggested by Golub and Kahan [11]. We choose a starting vector either of the form $(0, u^T)^T$ or of the form $(v^T, 0)^T$ where u is of length n, the column order of the A-matrix, and v is of length m , the row order of the A-matrix. If we use such a starting vector in the basic Lanczos recursion in Eqns (1.2.1) and (1.2.2), we obtain a version of the Lanczos recursion designed specifically for the B-matrix in Eqn(6.1.3). The Lanczos vectors generated by this recursion alternate in form from either $(0, u^T)^T$ to $(v^T, 0)^T$ or vice-versa, as the iterations proceed. Furthermore, on each iteration of this recursion it is only necessary to either compute Au_i or $A^T v_i$. Therefore, the amount of work per iteration of this recursion is no more than applying the real symmetric Lanczos recursion to a real symmetric matrix of order $\max m, n$. For details on the corresponding Lanczos recursion see Section 5.4 of Chapter 5 in Volume 1.

These codes can compute either a very few or very many of the distinct singular values of a given real rectangular matrix. As the documentation in Section 6.2 indicates, the A-multiplicity of a computed singular value can be obtained only with additional computation, and the modifications required to do this additional computation are not included in these versions of the codes.

The Lanczos recursions which we use generate a family of real symmetric, tridiagonal matrices (T-matrices). The diagonal entries of each of these T-matrices are all 0. The eigenvalues of any even-ordered T-matrix occur in $\pm$ pairs. This latter property is inherited from the B-matrix whose eigenvalues are just $\pm\sigma_i$, the $\pm$ pairs of singular values plus $m - 2n$ additional zero eigenvalues if $m \geq n$. Only even-ordered T-matrices may be used in the Lanczos computations. There is no reorthogonalization of the Lanczos vectors at any stage in any of the computations.

LSVAL, the main program for the single-vector, Lanczos singular value computations, calls the subroutine BISEC to compute eigenvalues of those Lanczos tridiagonal matrices specified by the user and on those subintervals specified by the user. The BISEC subroutine used in this chapter is a modification of the BISEC subroutine given in LESUB in Chapter 2 which assumes that the diagonal entries of the T-matrices supplied to it are all 0. BISEC simultaneously computes the T-eigenvalues and T-multiplicities and then sorts the computed T-eigenvalues into two categories, the 'good' T-eigenvalues and the 'spurious' T-eigenvalues. The 'good' T-eigenvalues are accepted as approximations to singular values of the user-specified matrix A. The accuracy of these 'good' T-eigenvalues as singular values of A is then estimated using error estimates computed by subroutine INVERR. The subroutine INVERR in this chapter is a modification of the INVERR subroutine in Chapter 2 which assumes the diagonal entries of the tridiagonal matrices supplied to it are all 0. Error estimates are computed only for isolated 'good' T-eigenvalues. All other 'good' T-eigenvalues are assumed to have converged. If convergence has not yet occurred and a larger Lanczos matrix has been specified by the user, these programs will continue on repeating the above procedure on a larger Lanczos matrix.

Once the singular values have been computed accurately enough, the user can select a subset of the 'converged' singular values for which singular vectors are to be computed. The main program LSVEC, for computing singular vectors of real rectangular matrices, is then used to compute these desired singular vectors. These singular vectors are obtained by computing Ritz vectors for the B-matrix and then splitting each of these $(m+n)$-dimensional Ritz vectors into approximate left and right singular vectors of A. The user should note that if the singular value being considered is very small, then LSVEC is not able to accurately compute both a left and a right singular vector approximation simultaneously. In this situation one of the two singular vectors will be more accurate than the other one is. If the starting vector is of the form $(0, u^T)^T$, then the right singular vector will be more accurate than the corresponding left vector. Similarly, if we use a starting vector of the form $(v^T, 0)^T$, then the left vector will be more accurate than the right vector will be. This loss in accuracy in one of the two vectors increases as the size of the singular value is decreased, and in the limit for a zero singular value, one of the two computed singular vectors will have no accuracy at all. See Section 5.4 of Chapter 5 in Volume 1.

All computations are in double precision real arithmetic. The user must supply a subroutine USPEC which defines and initializes the user-specified matrix A, and subroutines SVMAT and STRAN which compute respectively, matrix-vector multiplies Ax and A^Ty for any given vectors x and y. These subroutines must be constructed in such a way as to take advantage of the sparsity (and/or structure) of the user-supplied A-matrix and such that these computations are done accurately. More details about these real rectangular, single-vector Lanczos procedures are given in Section 5.4 of Chapter 5 in Volume 1.

6.2 Documentation for the Codes in Chapters 6

```
C-----LSVALHED------------------------------------------------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C
C
C         REFERENCE: Cullum and Willoughby, Chapter 5
C         Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C         Applied Mathematics, 2002. SIAM Publications,
C         Philadelphia, PA. USA
C
C
C
C     DOCUMENTATION FOR THE SINGLE-VECTOR
C     LANCZOS SINGULAR VALUE/VECTOR PROGRAMS
C     FOR REAL, RECTANGULAR MATRICES
C
C-------------------------------------------------------------------
C
C     GIVEN A REAL RECTANGULAR MATRIX A OF ORDER M X N THE THREE
C     SETS OF FORTRAN FILES LABELLED LSVAL, LSSUB, AND LSMULT
C     CAN BE USED TO COMPUTE DISTINCT SINGULAR VALUES OF A IN
C     USER-SPECIFIED INTERVALS.
C
C     CORRESPONDING SINGULAR VECTORS FOR SELECTED, COMPUTED
C     SINGULAR VALUES CAN BE COMPUTED USING THE SETS OF FILES
C     LABELLED LSVEC, LSSUB AND LSMULT.
C
C     THESE PROGRAMS USE LANCZOS TRIDIAGONALIZATION WITHOUT
C     REORTHOGONALIZATION ON THE ASSOCIATED REAL SYMMETRIC MATRIX
C
C                                    ----                 ----
C                                   |  0                    A  |
C                      B  =         |                          |
C                                   |  A-TRANSPOSE          0  |
C                                    ----                 ----
```

```
C
C     OF ORDER M + N  TO GENERATE REAL SYMMETRIC TRIDIAGONAL
C     MATRICES, T(1,MEV), OF ORDER MEV.  SUBSETS OF THE EIGENVALUES OF
C     THESE T-MATRICES, LABELLED AS THE 'GOOD EIGENVALUES' OF T(1,MEV),
C     ARE APPROXIMATIONS TO THE DESIRED SINGULAR VALUES OF A.
C     CORRESPONDING RITZ VECTORS FOR B ARE APPROXIMATIONS TO
C     EIGENVECTORS OF B WHICH IN TURN CONTAIN APPROXIMATIONS TO
C     THE DESIRED LEFT AND RIGHT SINGULAR VECTORS OF A.  THIS
C     PROCEDURE USES A SPECIAL STARTING VECTOR SUGGESTED BY GOLUB
C     AND KAHAN.  THUS, THE STARTING LANCZOS VECTOR IS EITHER OF
C     THE FORM (V1,0) OR (0,V2) WHERE V1 IS MX1 AND V2 IS NX1 AND
C     ALL SUCCEEDING LANCZOS VECTORS GENERATED ALTERNATE BETWEEN
C     THESE 2 FORMS.  THIS SPECIAL CHOICE OF STARTING VECTOR RESULTS
C     IN SIGNIFICANT GAINS IN STORAGE AND OPERATION COUNTS AND
C     ALSO IN CONVERGENCE RELATIVE TO A 'BRUTE FORCE' APPLICATION
C     OF THE REAL SYMMETRIC LANCZOS PROCEDURE DIRECTLY TO THE
C     MATRIX B ABOVE.  FOR MORE DETAILS SEE REFERENCE 1 BELOW.
C     IN THE DISCUSSIONS T(1,MEV) DENOTES THE LANCZOS T-MATRIX
C     OF SIZE MEV.
C
C     THE IDEAS USED IN THESE PROGRAMS ARE DISCUSSED IN THE FOLLOWING
C     REFERENCES.
C
C     1.  JANE CULLUM, RALPH A. WILLOUGHBY AND MARK LAKE, A LANCZOS
C         ALGORITHM FOR COMPUTING SINGULAR VALUES AND VECTORS OF LARGE
C         MATRICES, SIAM J. SCIENTIFIC AND STATISTICAL COMPUTING,
C         VOL. 4, JUNE 1983, PP. 197-215.
C
C     2.  JANE CULLUM AND RALPH A. WILLOUGHBY, LANCZOS ALGORITHMS
C         FOR LARGE SYMMETRIC MATRICES,  PROGRESS IN
C         SCIENTIFIC COMPUTING, EDITORS, G. GOLUB, H.O. KREISS,
C         S. ARBARBANEL, AND R. GLOWINSKI,  BIRKHAUSER BOSTON INC.,
C         CAMBRIDGE, MASSACHUSETTS, 1984.
C
C     3.  JANE CULLUM AND RALPH A. WILLOUGHBY, COMPUTING EIGENVECTORS
C         (AND EIGENVALUES) OF LARGE, SYMMETRIC MATRICES USING
C         LANCZOS TRIDIAGONALIZATION, LECTURE NOTES IN MATHEMATICS,
C         773, NUMERICAL ANALYSIS PROCEEDINGS, DUNDEE 1979, EDITED BY
C         G. A. WATSON, SPRINGER-VERLAG, (1980), BERLIN, PP.46-63.
C
C     4. IBID, LANCZOS AND THE COMPUTATION IN SPECIFIED INTERVALS OF
C        THE SPECTRUM OF LARGE SPARSE, REAL SYMMETRIC MATRICES, SPARSE
C        MATRIX PROCEEDINGS 1978, ED. I.S. DUFF AND G. W. STEWART,
C        SIAM, PHILADELPHIA, PP.220-255, 1979.
C
C     5. IBID, COMPUTING EIGENVALUES OF VERY LARGE SYMMETRIC MATRICES-
C        AN IMPLEMENTATION OF A LANCZOS ALGORITHM WITHOUT
C        REORTHOGONALIZATION, J. COMPUT. PHYS. 44(1981), 329-358.
C
C
C-----PORTABILITY-------------------------------------------------------
C
C
C     PROGRAMS WERE TESTED FOR PORTABILITY USING THE PFORT VERIFIER.
C     FOR DETAILS OF THE VERIFIER SEE FOR EXAMPLE, B. G. RYDER AND
```

```
C     A. D. HALL, "THE PFORT VERIFIER", COMPUTING SCIENCE TECHNICAL     LSV01080
C     REPORT 12, BELL LABORATORIES, MURRAY HILL, NEW JERSEY 07974,      LSV01090
C     (REVISED), JANUARY 1981.                                          LSV01100
C                                                                       LSV01110
C     EXCEPT FOR THE FOLLOWING CONSTRUCTIONS WHICH CAN BE EASILY        LSV01120
C     MODIFIED BY THE USER TO MATCH THE PARTICULAR COMPUTER BEING       LSV01130
C     USED, THE PROGRAM STATEMENTS ARE PORTABLE.                        LSV01140
C                                                                       LSV01150
C    NONPORTABLE CONSTRUCTIONS.                                         LSV01160
C                                                                       LSV01170
C    IN LSVAL AND IN LSVEC                                              LSV01180
C         1.  DATA/MACHEP STATEMENT                                     LSV01190
C         2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                    LSV01200
C         3.  FORMAT(20A4) USED FOR THE EXPLANATORY HEADER ARRAY, EXPLANLSV01210
C         4.  FORMAT(4Z20) USED TO READ AND WRITE BETA FILES 1 AND 2.   LSV01220
C    IN LSMULT                                                          LSV01230
C         1.  IN SVMAT, STRAN, AND  USPEC THE ENTRY THAT PASSES THE     LSV01240
C             STORAGE LOCATIONS OF THE ARRAYS DEFINING THE              LSV01250
C             USER-SPECIFIED MATRIX.                                    LSV01260
C         2.  IN SAMPLE USPEC FOR 'DIAGONAL' MATRICES:  THE FREE        LSV01270
C             FORMAT (8,*) AND THE FORMAT (20A4).                       LSV01280
C    IN LSSUB                                                           LSV01290
C         1.  ALL STATEMENTS ARE PORTABLE.                              LSV01300
C                                                                       LSV01310
C                                                                       LSV01320
C     IN THE COMMENTS BELOW:                                            LSV01330
C     COMPLEX*16 = COMPLEX VARIABLE, 16 BYTES OF STORAGE                LSV01340
C     REAL*8 = REAL VARIABLE, 8 BYTES OF STORAGE                        LSV01350
C     REAL*4 = REAL VARIABLE, 4 BYTES OF STORAGE                        LSV01360
C     INTEGER*4 = INTEGER VARIABLE, 4 BYTES                             LSV01370
C                                                                       LSV01380
C                                                                       LSV01390
C-----A-MATRIX SPECIFICATION--------------------------------------------LSV01400
C                                                                       LSV01410
C                                                                       LSV01420
C     SUBROUTINE USPEC IS USED TO SPECIFY THE USER-SUPPLIED A-MATRIX.   LSV01430
C     SUBROUTINES SVMAT AND STRAN ARE, RESPECTIVELY, CORRESPONDING      LSV01440
C     MATRIX-VECTOR MULTIPLE SUBROUTINES FOR A AND FOR A-TRANSPOSE.     LSV01450
C     THESE SUBROUTINES SHOULD BE DESIGNED TO TAKE ADVANTAGE OF         LSV01460
C     ANY SPECIAL PROPERTIES OF THE USER-SUPPLIED MATRIX.  THE          LSV01470
C     MATRIX-VECTOR MULTIPLIES REQUIRED BY THE LANCZOS PROCEDURES       LSV01480
C     MUST BE COMPUTED RAPIDLY AND ACCURATELY.                          LSV01490
C                                                                       LSV01500
C     SUBROUTINE USPEC HAS THE CALLING SEQUENCE                         LSV01510
C                                                                       LSV01520
C            CALL USPEC(M,N,MATNO)                                      LSV01530
C                                                                       LSV01540
C     WHERE M IS THE NUMBER OF ROWS IN THE USER-SPECIFIED               LSV01550
C     A-MATRIX AND N IS THE NUMBER OF COLUMNS. MATNO IS A               LSV01560
C     <= 8 DIGIT INTEGER USED AS A MATRIX AND TEST IDENTIFICATION       LSV01570
C     NUMBER.  THIS SUBROUTINE DEFINES (DIMENSIONS) THE ARRAYS          LSV01580
C     REQUIRED TO SPECIFY THE A-MATRIX.  THIS SUBROUTINE ALSO           LSV01590
C     INITIALIZES THESE ARRAYS AND ANY OTHER PARAMETERS NEEDED TO       LSV01600
C     DEFINE THE MATRIX.  THE STORAGE LOCATIONS OF THESE PARAMETERS     LSV01610
C     AND ARRAYS ARE THEN PASSED TO THE MATRIX-VECTOR MULTIPLY          LSV01620
```

```
C     SUBROUTINES SVMAT AND STRAN VIA ENTRIES.  SAMPLE SUBROUTINES
C     ARE INCLUDED IN THE FORTRAN FILE LSMULT.
C
C     IMPORTANT NOTE:
C     THE SAMPLE MATRIX-VECTOR MULTIPLY SUBROUTINES IN LSMULT
C     ASSUME THAT M >= N.  THEY ALSO ASSUME THAT THE USER-SUPPLIED
C     INFORMATION ABOUT THE GIVEN MATRIX IS STORED ON FILE 8.
C     THE USER SHOULD SEE THE LSMULT PROGRAMS FOR MORE DETAILS.
C
C     SUBROUTINE SVMAT HAS THE CALLING SEQUENCE
C
C            CALL SVMAT(W,U,SUM)
C
C     WHERE U AND W ARE REAL*8 VECTORS AND SUM IS A REAL*8
C     SCALAR.  SVMAT CALCULATES U = A*W - SUM*U FOR THE
C     USER-SPECIFIED A-MATRIX.  SUBROUTINE STRAN HAS THE
C     CALLING SEQUENCE
C
C            CALL STRAN(W,U,SUM)
C
C     STRAN CALCULATES U = (A-TRANSPOSE)*W - SUM*U FOR THE
C     TRANSPOSE OF THE USER-SUPPLIED A-MATRIX.  THE ARRAY AND PARAMETER
C     INFORMATION NEEDED TO PERFORM THE MATRIX-VECTOR MULTIPLIES
C     IS PASSED TO THE SVMAT AND THE STRAN SUBROUTINES FROM THE
C     USPEC SUBROUTINE VIA ENTRIES.  ONE SET OF THE SAMPLE SVMAT
C     AND STRAN SUBROUTINES INCLUDED IN LSMULT COMPUTES
C     MATRIX-VECTOR MULTIPLIES FOR AN ARBITRARY SPARSE,
C     RECTANGULAR MATRIX STORED IN THE SPARSE FORMAT SPECIFIED
C     IN THE CORRESPONDING SAMPLE USPEC SUBROUTINE.  THE LANCZS
C     SUBROUTINE CALLS SVMAT AND STRAN IN THE GENERATION OF
C     THE LANCZOS T-MATRICES FOR THE B MATRIX.
C
C     THE DATA FOR THE A-MATRIX IS ASSUMED TO BE ON FILE 8 AND
C     IN THE FOLLOWING SPARSE FORMAT:
C     NZ = NUMBER OF NONZERO ELEMENTS OF A
C     ICOL(K), K = 1,N, NUMBER OF NONZEROS OF A IN COLUMN K.
C     IROW(K), K = 1,NZ, ROW INDEX OF A(K).
C     A(K), K=1,NZ  CONTAINS THE ELEMENTS OF A BY COLUMN.
C
C
C     SVMATV AND STRAN ARE CALLED FROM THE SUBROUTINE LANCZS
C     WHICH GENERATES THE LANCZOS TRIDIAGONAL MATRICES, THE
C     BETA HISTORY.   SIMILARLY, THSE SUBROUTINES ARE CALLED FROM
C     THE CORRESPONDING SINGULAR VECTOR PROGRAM, LSVEC.
C     SVMAT AND STRAN ARE DECLARED AS EXTERNAL VARIABLES.
C     EACH IS AN ARGUMENT FOR THE LANCZS SUBROUTINE.
C
C     USPEC, SVMAT, AND STRAN SUBROUTINES SUITABLE FOR THE
C     USER-SPECIFIED MATRIX MUST BE SUPPLIED BY THE USER.
C
C     THE MAIN PROGRAMS FOR THE SINGULAR VALUE AND SINGULAR VECTOR
C     CALCULATIONS ASSUME THAT INPUT FILE 5 CONTAINS THE ROW ORDER
C     M AND THE COLUMN ORDER N OF THE GIVEN A-MATRIX AND MATNO,
C     AN IDENTIFICATION NUMBER OF <= 8 DIGITS FOR THE GIVEN MATRIX.
C
```

```
C
C-----MACHEP-----------------------------------------------------------
C
C
C     MACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING THE RELATIVE
C     PRECISION OF THE FLOATING POINT ARITHMETIC USED.
C     MACHEP = 2.2 * 10**-16 FOR DOUBLE PRECISION ARITHMETIC ON
C     IBM 370-3081.
C
C     THE USER WILL HAVE TO RESET THIS PARAMETER TO
C     THE CORRESPONDING VALUE FOR THE MACHINE BEING USED.  NOTE THAT
C     IF A MACHINE WITH A MACHINE EPSILON THAT IS MUCH LARGER THAN THE
C     VALUE GIVEN HERE IS BEING USED, THEN THERE COULD BE
C     PROBLEMS WITH THE TOLERANCES.
C
C
C-----SUBROUTINES AND FUNCTIONS USER MUST SUPPLY------------------------
C
C
C     GENRAN,  FINPRO,  MASK,  USPEC,  SVMAT AND  STRAN
C
C     GENRAN = COMPUTES K PSEUDO-RANDOM NUMBERS AND STORES THEM IN
C              THE REAL ARRAY, G.  THIS SUBROUTINE IS USED TO
C              GENERATE A STARTING VECTOR FOR THE LANCZOS PROCEDURE
C              IN THE SUBROUTINE LANCZS AND A STARTING RIGHT-HAND SIDE
C              FOR INVERSE ITERATION IN THE SUBROUTINE INVERR.
C
C              TESTS REPORTED IN THE REFERENCES USED EITHER GGL1 OR
C              GGL2 FROM THE IBM LIBRARY  SLMATH.
C              THE EXISTING CALLING SEQUENCE IS:
C
C                      CALL GENRAN(IIX,G,K).
C
C              WHERE IIX =INTEGER SEED, G = REAL*4 ARRAY WHOSE
C              DIMENSION MUST BE >= K.  K RANDOM NUMBERS ARE GENERATED
C              AND PLACED IN G.
C
C     FINPRO = DOUBLE PRECISION FUNCTION WHICH COMPUTES THE INNER
C              PRODUCT OF 2 DOUBLE PRECISION VECTORS OF DIMENSION  K.
C              TESTS REPORTED IN THE REFERENCES USED THE HARWELL
C              LIBRARY SUBROUTINE FM02AD.
C              EXISTING CALLING SEQUENCE IS
C
C                      CALL FINPRO(N,V,J,W,K).
C
C              COMPUTES THE INNER PRODUCT OF DIMENSION N OF THE VECTORS
C              V AND W.  SUCCESSIVE COMPONENTS OF V AND OF W ARE STORED
C              AT LOCATIONS THAT ARE ,RESPECTIVELY, J AND K UNITS APART.
C
C     MASK = MASKS OVERFLOW AND UNDERFLOW.
C            USER MUST SUPPLY OR COMMENT OUT CALL.
C
C    USPEC = DIMENSIONS AND INITIALIZES ARRAYS NEEDED TO SPECIFY
C            USER-SUPPLIED A-MATRIX.  SEE A-MATRIX SPECIFICATION SECTION
C
```

```
C    SVMAT = MATRIX-VECTOR MULTIPLY FOR USER-SUPPLIED A-MATRIX.
C            SEE A-MATRIX SPECIFICATION SECTION.
C
C    STRAN = MATRIX-VECTOR MULTIPLY FOR TRANSPOSE OF USER-SUPPLIED
C            A-MATRIX.  SEE A-MATRIX SPECIFICATION SECTION.
C
C
C-----------------------------------------------------------------------
C
C     COMMENTS FOR SINGULAR VALUE COMPUTATIONS
C
C-----------------------------------------------------------------------
C
C
C-----PARAMETER CONTROLS FOR SINGULAR VALUE PROGRAMS--------------------
C
C
C     PARAMETER CONTROLS ARE INTRODUCED TO ALLOW SEGMENTATION OF THE
C     SINGULAR VALUE COMPUTATIONS AND TO ALLOW VARIOUS COMBINATIONS
C     OF READ/WRITES.
C
C     THE FLAG ISTART CONTROLS THE T-MATRIX (BETA HISTORY)
C     GENERATION.
C
C     ISTART = (0,1)  MEANS
C
C              (0) THERE IS NO EXISTING BETA HISTORY AND ONE
C                  MUST BE GENERATED.
C
C              (1) THERE IS AN EXISTING BETA HISTORY AND IT IS
C                  TO BE READ IN FROM FILE 2 AND EXTENDED IF NECESSARY.
C
C     THE FLAG ISTOP CAN BE USED IN CONJUNCTION WITH THE FLAG ISTART TO
C     ALLOW SEGMENTATION OF THE SINGULAR VALUE COMPUTATIONS.
C
C     ISTOP  = (0,1)  MEANS
C
C              (0) PROGRAM COMPUTES ONLY THE REQUESTED BETAS,
C                  STORES THEM AND THE LAST 2 LANCZOS VECTORS GENERATED
C                  IN FILE 1 AND THEN TERMINATES.
C
C              (1) PROGRAM COMPUTES REQUESTED BETAS AND THEN
C                  USES THE BISEC SUBROUTINE TO CALCULATE EIGENVALUES
C                  OF THE TRIDIAGONAL MATRICES GENERATED FOR THE ORDERS
C                  SPECIFIED BY THE USER AND ON THE USER-SPECIFIED
C                  INTERVALS.  PROGRAM THEN USES THE SUBROUTINE INVERR
C                  TO COMPUTE ERROR ESTIMATES FOR THE ISOLATED GOOD
C                  T-EIGENVALUES WHICH ARE USED TO CHECK THE
C                  CONVERGENCE OF THESE T-EIGENVALUES.
C
C     CONTROL PARAMETERS FOR WRITES
C
C     IHIS  = (0,1)  MEANS
C
C             (0) IF ISTOP .GT. 0 THEN BETAS ARE NOT SAVED ON FILE 1.
```

```
C
C             (1) PROGRAM WRITES BETAS AND LAST 2 LANCZOS
C                 VECTORS TO FILE 1 SO THAT THE T-MATRIX GENERATION
C                 MAY BE REUSED OR CONTINUED LATER IF NECESSARY.
C                 TYPICALLY ONE WOULD ALWAYS DO THIS ON ANY RUN WHERE
C                 A HISTORY FILE IS BEING GENERATED.  HISTORY MUST BE
C                 SAVED IN MACHINE FORMAT ((4Z20) FOR IBM/3081) SO
C                 THAT NO ERRORS DUE TO FORMAT CONVERSIONS OCCUR.
C
C     IDIST  = (0,1)  MEANS
C
C              (0) DISTINCT EIGENVALUES OF T-MATRICES ARE NOT SAVED.
C
C              (1) PROGRAM WRITES COMPUTED DISTINCT EIGENVALUES OF
C                  T-MATRICES ALONG WITH THEIR T-MULTIPLICITIES
C                  TO FILE 11.
C
C     IWRITE = (0,1)   MEANS
C
C              (0) NO EXTENDED OUTPUT FROM SUBROUTINES BISEC AND INVERR
C                  IS SENT TO FILE 6.
C
C              (1) INDIVIDUAL COMPUTED T-EIGENVALUES AND CORRESPONDING
C                  ERROR ESTIMATES FROM THE SUBROUTINES BISEC AND INVERR
C                  ARE PRINTED OUT TO FILE 6 AS THEY ARE COMPUTED.
C
C     THE PROGRAM ALWAYS MAKES A SEPARATE LIST OF THE COMPUTED GOOD
C     EIGENVALUES OF THE LANCZOS MATRICES T(1,MEV) CONSIDERED,
C     THESE ARE THE APPROXIMATIONS TO THE DESIRED SINGULAR VALUES,
C     ALONG WITH THEIR MINIMAL GAPS AS SINGULAR VALUES OF A AND
C     WRITES THEM TO FILE 3.  CORRESPONDING ERROR ESTIMATES FOR ANY
C     ISOLATED COMPUTED GOOD T-EIGENVALUES (SINGULAR VALUES OF A)
C     ARE ALWAYS WRITTEN TO FILE 4.
C
C
C-----INPUT/OUTPUT FILES FOR SINGULAR VALUE PROGRAMS--------------------
C
C     ANY INPUT DATA OTHER THAN THE BETA HISTORY SHOULD BE STORED
C     ON FILE 5. SEE SAMPLE INPUT/OUTPUT FROM TYPICAL RUN.
C     THE READ STATEMENTS IN THE GIVEN FORTRAN PROGRAM ASSUME THAT
C     THE DATA STORED ON FILE 5 IS IN FREE FORMAT.  USER SHOULD NOTE
C     THAT 'FREE FORMAT' IS NOT CLASSIFIED AS PORTABLE BY PFORT SO THAT
C     THE USER MAY HAVE TO MODIFY THE READ STATEMENTS FROM FILE 5 TO
C     CONFORM TO WHAT IS PERMISSIBLE ON THE MACHINE BEING USED.
C
C     FILE 6 WAS USED AS THE INTERACTIVE TERMINAL OUTPUT FILE.
C     THIS FILE PROVIDES A RUNNING ACCOUNT OF THE PROGRESS OF THE
C     COMPUTATIONS.  THE AMOUNT OF INFORMATION PRINTED OUT IS
C     CONTROLLED BY THE PARAMETER IWRITE.
C
C DESCRIPTION OF OTHER I/O FILES
C
C FILE (K)     CONTAINS:
C
C      (1)     OUTPUT FILE:
```

```
C            HISTORY FILE OF NEWLY-GENERATED T-MATRIX                  LSV03830
C            (BETA VECTOR) AND LAST 2 LANCZOS VECTORS USED             LSV03840
C            IN THE T-MATRIX GENERATION.                               LSV03850
C            IF IHIS = 0 AND ISTOP = 1, FILE 1 IS NOT WRITTEN.         LSV03860
C                                                                      LSV03870
C     (2)    INPUT FILE:                                               LSV03880
C            SAME AS FILE 1 EXCEPT THAT IT CONTAINS A                  LSV03890
C            PREVIOUSLY-GENERATED T-MATRIX (IF ANY). IF ISTART = 1,    LSV03900
C            PROGRAM ASSUMES THAT THERE IS A HISTORY FILE OF           LSV03910
C            BETAS ON FILE 2.  THESE BETAS AND THE LAST TWO LANCZOS    LSV03920
C            VECTORS USED IN THE T-MATRIX GENERATION ARE READ IN.      LSV03930
C                                                                      LSV03940
C     (3)    OUTPUT FILE:                                              LSV03950
C            COMPUTED GOOD EIGENVALUES OF THE T-MATRICES CONSIDERED.   LSV03960
C            ALSO CONTAINS T-MULTIPLICITIES OF THESE T-EIGENVALUES AS  LSV03970
C            EIGENVALUES OF THE T-MATRIX, AND THEIR GAPS AS            LSV03980
C            EIGENVALUES IN THE B MATRIX AND IN THE T-MATRIX.          LSV03990
C            NOTE THAT THESE GOOD T-EIGENVALUES ARE THE COMPUTED       LSV04000
C            SINGULAR VALUES OF THE A-MATRIX AND THAT THE GAPS         LSV04010
C            OF THESE EIGENVALUES AS EIGENVALUES OF THE B-MATRIX       LSV04020
C            ARE EQUAL TO THEIR GAPS AS SINGULAR VALUES OF A.  FILE    LSV04030
C            3 IS ALWAYS WRITTEN.                                      LSV04040
C                                                                      LSV04050
C     (4)    OUTPUT FILE:                                              LSV04060
C            ERROR ESTIMATES FOR THE ISOLATED COMPUTED SINGULAR        LSV04070
C            SINGULAR VALUES (ISOLATED GOOD EIGENVALUES OF T(1,MEV))   LSV04080
C            THESE ARE OBTAINED USING THE SUBROUTINE INVERR. THESE     LSV04090
C            ESTIMATES USE THE LAST COMPONENTS OF THE ASSOCIATED       LSV04100
C            T-EIGENVECTORS WHICH ARE COMPUTED USING INVERSE           LSV04110
C            ITERATION.  FILE 4 IS ALWAYS WRITTEN.                     LSV04120
C                                                                      LSV04130
C                                                                      LSV04140
C     (8)    INPUT FILE:                                               LSV04150
C            SAMPLE USPEC SUBROUTINE ASSUMES THAT THE ARRAYS           LSV04160
C            REQUIRED TO SPECIFY THE USER'S MATRIX ARE STORED ON       LSV04170
C            FILE 8.  USERS MUST MAKE WHATEVER DEFINITIONS ARE         LSV04180
C            APPROPRIATE FOR THEIR MATRICES.                           LSV04190
C                                                                      LSV04200
C     (9)    OUTPUT FILE:  OPTIONAL                                    LSV04210
C            CAN BE USED TO STORE THE TRUE SINGULAR VALUES OF          LSV04220
C            A GIVEN TEST MATRIX, WHEN THE SINGULAR VALUE PROCEDURE    LSV04230
C            IS BEING EXERCISED ON A TEST MATRIX.                      LSV04240
C                                                                      LSV04250
C    (11)    OUTPUT FILE:                                              LSV04260
C            COMPUTED DISTINCT EIGENVALUES OF T-MATRICES USED.         LSV04270
C            ALSO CONTAINS THEIR T-MULTIPLICITIES AND T-GAPS TO        LSV04280
C            NEAREST DISTINCT T-EIGENVALUES, AND THE T-MULTIPLICITY    LSV04290
C            PATTERN OF THE GOOD AND THE SPURIOUS T-EIGENVALUES.       LSV04300
C            FILE 11 IS WRITTEN ONLY IF IDIST = 1.                     LSV04310
C                                                                      LSV04320
C                                                                      LSV04330
C-----PARAMETERS SET BY THE SINGULAR VALUE PROGRAMS---------------------LSV04340
C                                                                      LSV04350
C                                                                      LSV04360
C     THESE PARAMETERS ARE SET INTERNALLY IN THE PROGRAM               LSV04370
```

```
C
C     SCALEK     K = 1,2,3,4
C
C                THE SCALING FACTORS SCALEK HAVE BEEN INTRODUCED IN AN
C                ATTEMPT TO MAKE THE TOLERANCES USED IN THE
C                T-MULTIPLICITY, SPURIOUS, ISOLATION AND PRTESTS ADJUST
C                TO THE SCALE OF THE GIVEN MATRIX.  THESE FACTORS MUST
C                NOT BE MODIFIED.
C
C     NOTE:    THE USER SHOULD NOTE THAT IF THE MATRIX BEING
C     PROCESSED IS VERY STIFF, THAT IS THE RATIO OF THE LARGEST
C     SINGULAR VALUE TO THE SMALLEST SINGULAR VALUE IS VERY
C     LARGE, THEN THE TOLERANCES BEING USED IN BISEC, LUMP, ISOEV
C     AND PRTEST MAY NOT TREAT THE SMALLEST SINGULAR VALUES
C     VERY WELL.  IN SOME SUCH CASES A USER-INTRODUCED REDUCTION
C     IN THE SIZE OF TKMAX AND THE SUBSEQUENT RECOMPUTATION OF
C     THE T-MATRIX EIGENVALUES CORRESPONDING TO THE SMALLEST
C     SINGULAR VALUES USING THIS TKMAX MAY RESULT IN IMPROVED
C     COMPUTATIONS AT THE LOW END.
C
C     THE LUMP, ISOEV, AND PRTEST TOLERANCES THAT WERE USED
C     MOST IN  THE TESTING OF THIS ALGORITHM WERE NOT
C     SCALE INVARIANT BUT SEEMED TO WORK WELL ON MATRICES THAT
C     HAD SINGULAR VALUES BOTH GREATER THAN AND LESS
C     THAN 1.  THESE TOLERANCES ARE ALSO INCLUDED IN THESE THREE
C     SUBROUTINES BUT AS COMMENTED OUT STATEMENTS.  THEY CAN BE
C     REVIVED BY COMMENTING OUT THE CORRESPONDING TOLERANCES
C     SPECIFIED IN THE STATEMENT ABOVE EACH OF THESE.
C
C     IMPORTANT TOLERANCES OR SCALES THAT ARE USED REPEATEDLY
C     THROUGHOUT THIS PROGRAM ARE THE FOLLOWING:
C     SCALED MACHINE EPSILON:  TTOL = TKMAX*EPSM WHERE
C     EPSM = 2*MACHINE EPSILON AND
C     TKMAX = MAX(BETA(J), J = 1,MEV)
C     BISEC CONVERGENCE TOLERANCE:  BISTOL = DSQRT(1000+MEV)*TTOL
C     BISEC T-MULTIPLICITY TOLERANCE:  MULTOL = (1000+MEV)*TTOL
C     LANCZOS CONVERGENCE TOLERANCE:   CONTOL = BETA(MEV+1)*1.D-10
C
C
C     BTOL = RELATIVE TOLERANCE USED TO ESTIMATE ANY LOSS OF LOCAL
C            ORTHOGONALITY OF THE LANCZOS VECTORS AFTER THE T-MATRIX
C            HAS BEEN GENERATED.  THE LANCZOS PROCEDURE WORKS WELL
C            ONLY IF LOCAL ORTHOGONALITY BETWEEN SUCCESSIVE LANCZOS
C            VECTORS IS MAINTAINED.  THE TNORM SUBROUTINE TESTS
C            WHETHER OR NOT
C
C                 MINIMUM  |BETA(I)|/||A|| > BTOL.
C                 I=2,KMAX
C
C            IF THIS TEST IS VIOLATED BY SOME BETA AND A T-MATRIX THAT
C            WOULD INCLUDE SUCH A BETA IS REQUESTED, THEN THE LANCZOS
C            PROCEDURE WILL TERMINATE FOR THE USER TO DECIDE WHAT TO
C            DO.  THE USER CAN OVER-RIDE THIS TEST BY SIMPLY DECREASING
C            THE SIZE OF BTOL, BUT THEN CONVERGENCE IS NOT AS CERTAIN.
C            THE PROGRAM SETS BTOL = 1.D-8 WHICH IS A VERY CONSERVATIVE
```

```
C           CHOICE. THE || A || IS ESTIMATED BY USING AN ESTIMATE      LSV04930
C           OF THE NORM OF THE T-MATRIX, T(1,KMAX).                    LSV04940
C                                                                      LSV04950
C     GAPTOL = RELATIVE TOLERANCE USED IN THE SUBROUTINE ISOEV         LSV04960
C             TO DETERMINE FOR WHICH OF THE GOOD T-EIGENVALUES,        LSV04970
C             THE COMPUTED SINGULAR VALUES, ERROR ESTIMATES SHOULD     LSV04980
C             BE COMPUTED.  THE PROGRAM SETS GAPTOL = 1.D-8.           LSV04990
C             IF FOR A GIVEN 'GOOD' T-EIGENVALUE OF THE GIVEN          LSV05000
C             T-MATRIX  THE COMPUTED GAP IN THE T-MATRIX IS TOO        LSV05010
C             SMALL AND IS DUE TO A 'SPURIOUS' EIGENVALUE OF           LSV05020
C             THE T-MATRIX, THEN THE 'GOOD' T-EIGENVALUE IS ASSUMED    LSV05030
C             TO HAVE CONVERGED AND AN ERROR ESTIMATE IS NOT           LSV05040
C             COMPUTED.                                                LSV05050
C                                                                      LSV05060
C                                                                      LSV05070
C-----USER-SPECIFIED PARAMETERS FOR SINGULAR VALUE PROGRAMS-------------LSV05080
C                                                                      LSV05090
C                                                                      LSV05100
C     RELTOL = RELATIVE TOLERANCE USED IN 'COMBINING' COMPUTED         LSV05110
C             EIGENVALUES OF T(1,MEV) PRIOR TO COMPUTING ERROR         LSV05120
C             ESTIMATES.                                               LSV05130
C                                                                      LSV05140
C     THE LUMPING OF T-EIGENVALUES OCCURS IN SUBROUTINE LUMP.          LSV05150
C     LUMPING IS NECESSARY BECAUSE IT IS IMPOSSIBLE TO ACCURATELY      LSV05160
C     PREDICT THE ACCURACY OF THE BISEC SUBROUTINE.  LUMP 'COMBINES'   LSV05170
C     T-EIGENVALUES THAT HAVE SLIPPED BY THE TOLERANCE THAT WAS USED   LSV05180
C     IN THE T-MULTIPLICITY TESTS.  IN PARTICULAR IF FOR SOME J,       LSV05190
C                                                                      LSV05200
C   |EVALUE(J)-EVALUE(J-1)| < DMAX1(RELTOL*|EVALUE(J)|,SCALE2*MULTOL)   LSV05210
C                                                                      LSV05220
C     THEN THESE T-EIGENVALUES ARE 'COMBINED'.  MULTOL IS THE TOLERANCE LSV05230
C     THAT WAS USED IN THE T-MULTIPLICITY TEST IN BISEC.  SEE THE HEADERLSV05240
C     ON THE LUMP SUBROUTINE FOR MORE DETAILS.                         LSV05250
C                                                                      LSV05260
C     RELTOL IS SET TO 1.D-10.                                         LSV05270
C                                                                      LSV05280
C     MXINIT = MAXIMUM NUMBER OF INVERSE ITERATIONS ALLOWED IN         LSV05290
C             SUBROUTINE INVERR FOR EACH ISOLATED GOOD T-EIGENVALUE    LSV05300
C             CONSIDERED.  TYPICALLY ONLY ONE IS REQUIRED.             LSV05310
C                                                                      LSV05320
C     SEEDS FOR RANDOM NUMBER GENERATORS = INTEGER*4 SCALARS.          LSV05330
C                                                                      LSV05340
C              (1) SVSEED = SEED FOR STARTING VECTOR USED IN           LSV05350
C                  T-MATRIX GENERATION IN LANCZS SUBROUTINE            LSV05360
C                                                                      LSV05370
C              (2) RHSEED = SEED FOR RIGHT-HAND SIDE USED IN           LSV05380
C                  INVERSE ITERATION COMPUTATIONS IN INVERR.           LSV05390
C                                                                      LSV05400
C     BISEC DATA                                                       LSV05410
C                                                                      LSV05420
C     (1) NINT  =  NUMBER OF SUBINTERVALS ON WHICH SINGULAR VALUES     LSV05430
C                 ARE TO BE COMPUTED.                                  LSV05440
C                                                                      LSV05450
C     (2) LB(J) = (J = 1,NINT)  =  LEFT END POINTS OF THESE INTERVALS.  LSV05460
C                MUST BE PROVIDED IN INCREASING ORDER.  THAT IS,       LSV05470
```

```
C              LB(J) < LB(J+1) FOR J = 1,NINT.
C
C     (3) UB(J) = (J = 1,NINT) =  RIGHT END POINTS OF THESE INTERVALS.
C              MUST BE PROVIDED IN INCREASING ORDER.  THAT IS,
C              UB(J) < UB(J+1) FOR J = 1,NINT.
C
C     (4) MXSTUR  =  MAXIMUM NUMBER OF STURM ITERATIONS ALLOWED FOR
C                 ENTIRE SET OF SINGULAR VALUE CALCULATIONS OVER
C                 ALL SPECIFIED SIZE T-MATRICES.  PROGRAM WILL
C                 TERMINATE IF THIS LIMIT IS EXCEEDED.
C
C     T-MATRICES
C
C     SIZES OF T-MATRICES
C
C             (1) KMAX= MAXIMUM ORDER FOR T-MATRIX THAT USER IS WILLING
C                     TO CONSIDER.
C
C             (2) NMEVS = MAXIMUM NUMBER OF T-MATRICES THAT WILL BE
C                       CONSIDERED.
C
C             (3) NMEV(J)  (J=1,NMEVS)  = SIZES OF T-MATRIX TO BE
C                                       CONSIDERED SEQUENTIALLY.
C
C     T-MATRIX-GENERATION
C
C     IPAR   = (1,2)   MEANS
C
C              (1) STARTING VECTOR IS OF FORM (0,V2) WHERE V2 IS
C                  NX1.  USE WHEN M > N .
C
C              (2) STARTING VECTOR IF OF FORM (V1,0) WHERE V1 IS
C                  MX1.  USE WHEN M < N .
C
C     USER SHOULD NOTE THAT THIS PROGRAM FIRST COMPUTES A T-MATRIX
C     OF ORDER KMAX AND THEN CYCLES THROUGH THE T-MATRICES SPECIFIED
C     A PRIORI BY THE USER, USING THE SUBROUTINE BISEC TO COMPUTE
C     EIGENVALUES OF THE T-MATRICES ON THE INTERVALS SPECIFIED BY
C     THE USER.  SUBSETS OF THESE T-EIGENVALUES ARE THEN SELECTED
C     AS APPROXIMATIONS TO THE DESIRED SINGULAR VALUES.
C
C     IDEALLY, ONE WOULD COMPUTE THE SINGULAR VALUE APPROXIMATIONS
C     AT A REASONABLE SIZE T-MATRIX, LOOK AT THE ACCURACY OF THE
C     COMPUTED RESULTS AND USE THAT TO DETERMINE AN APPROPRIATE
C     INCREMENT FOR THE SIZE OF THE T-MATRIX BASED UPON WHAT
C     HAS ALREADY CONVERGED AND UPON THE SIZES OF THE ERROR ESTIMATES
C     ON THOSE SINGULAR VALUES THAT ARE DESIRED BUT THAT HAVE NOT
C     YET CONVERGED.  HOWEVER, IN THE INTERESTS OF GENERALITY AND
C     SIMPLICITY WE CHOSE NOT TO DO THAT HERE.
C
C
C-----CONVERGENCE TESTS FOR THE SINGULAR VALUE PROGRAMS-----------------
C
C
C     THE CONVERGENCE TEST INCORPORATED IN THIS PROGRAM IS
```

```
C     BASED UPON THE ASSUMPTION THAT THOSE T-EIGENVALUES AND           LSV06030
C     THEIR ASSOCIATED T-EIGENVECTORS THAT CORRESPOND TO               LSV06040
C     THE SINGULAR VALUES AND VECTORS WHICH WE WISH TO COMPUTE         LSV06050
C     CONVERGE AS THE T-SIZE IS INCREASED.                             LSV06060
C                                                                      LSV06070
C     AS CURRENTLY PROGRAMMED, CONVERGENCE IS CHECKED BY EXAMINING     LSV06080
C     THE SIZES OF ALL OF THE COMPUTED ERROR ESTIMATES ON ALL OF THE   LSV06090
C     INTERVALS SPECIFIED BY THE USER.  IDEALLY CONVERGENCE SHOULD     LSV06100
C     BE CHECKED ONLY ON THOSE SINGULAR VALUES OF INTEREST AND         LSV06110
C     ONCE THE SINGULAR VALUES ON SUB-INTERVALS OF THESE INTERVALS     LSV06120
C     HAVE CONVERGED, ANY SUBSEQUENT SINGULAR VALUE COMPUTATIONS       LSV06130
C     SHOULD BE MADE ONLY ON THE UNCONVERGED PORTIONS.  OBVIOUSLY,     LSV06140
C     IT WOULD BE DIFFICULT TO INCORPORATE CODE TO DO THE ABOVE        LSV06150
C     WITHOUT KNOWING A PRIORI PRECISELY WHAT THE USER IS TRYING       LSV06160
C     TO COMPUTE.  THEREFORE, WE DID NOT ATTEMPT TO DO THIS.  IF       LSV06170
C     ONE WISHES TO MAKE SUCH A MODIFICATION THEN ONE MUST ALSO        LSV06180
C     MODIFY THE PROGRAM SO THAT IT CREATES AN OVERALL LIST OF THE     LSV06190
C     CONVERGED SINGULAR VALUES AS THEY ARE COMPUTED, SINCE            LSV06200
C     CONVERGED SINGULAR VALUES OBTAINED AT A PARTICULAR VALUE OF      LSV06210
C     MEV WOULD NO LONGER BE RECOMPUTED AT LARGER VALUES OF MEV.       LSV06220
C                                                                      LSV06230
C     IF ONLY A FEW SINGULAR VALUES ARE TO BE COMPUTED THEN SUCH       LSV06240
C     CHANGES WOULD NOT MAKE MUCH DIFFERENCE IN THE RUNNING TIME.      LSV06250
C                                                                      LSV06260
C                                                                      LSV06270
C-----ARRAYS REQUIRED BY THE SINGULAR VALUE PROGRAMS--------------------LSV06280
C                                                                      LSV06290
C                                                                      LSV06300
C     BETA(J) = REAL*8 ARRAY.  ITS DIMENSION MUST BE AT LEAST KMAX+1.  LSV06310
C               THE LENGTH OF THE LARGEST T-MATRIX ALLOWED.  THIS      LSV06320
C               ARRAY CONTAINS THE SUBDIAGONAL ENTRIES OF THE          LSV06330
C               T-MATRICES.  THE DIAGONAL ENTRIES ARE ALL ZERO.        LSV06340
C                                                                      LSV06350
C               THE BETA VECTOR IS NOT ALTERED DURING THE              LSV06360
C               CALCULATIONS.  IMPORTANT NOTE:  ONLY EVEN ORDER        LSV06370
C               T-MATRICES ARE PERMISSIBLE.                            LSV06380
C                                                                      LSV06390
C     V1(J),V2(J),VS(J) = REAL*8 ARRAYS.   VS MUST BE OF               LSV06400
C                         DIMENSION AT LEAST KMAX.  V1 MUST BE         LSV06410
C                         OF DIMENSION AT LEAST MAX(M,KMAX+1).         LSV06420
C                         V2 MUST BE OF DIMENSION AT LEAST             LSV06430
C                         MAX(N,KMAX). M IS THE ROW DIMENSION OF       LSV06440
C                         A, AND N IS THE COLUMN DIMENSION.            LSV06450
C                         HOWEVER, THE DIMENSION                       LSV06460
C                         FOR V2 IS VALID ONLY IF NO MORE              LSV06470
C                         THAN KMAX/2 EIGENVALUES OF THE GIVEN         LSV06480
C                         T-MATRICES ARE TO BE COMPUTED IN ANY GIVEN   LSV06490
C                         SUBINTERVAL. V2 IS USED IN THE SUBROUTINE    LSV06500
C                         BISEC TO HOLD THE UPPER AND LOWER            LSV06510
C                         ENDPOINTS OF THE SUBINTERVALS GENERATED      LSV06520
C                         DURING THE BISECTIONS.  THEREFORE, ITS       LSV06530
C                         DIMENSION MUST ALWAYS BE AT LEAST 2*Q        LSV06540
C                         WHERE Q IS THE MAXIMUM NUMBER OF             LSV06550
C                         EIGENVALUES OF THE SPECIFIED T-MATRIX IN ANY LSV06560
C                         ONE OF THE SPECIFIED INTERVALS.              LSV06570
```

```
C
C     LB(J),UB(J) = REAL*8 ARRAYS. EACH MUST BE OF DIMENSION AT LEAST
C                   NINT, THE NUMBER OF SUBINTERVALS TO BE CONSIDERED.
C                   LB CONTAINS THE LEFT-END POINTS OF THE INTERVALS
C                   ON WHICH SINGULAR VALUES ARE TO BE COMPUTED.
C                   UB CONTAINS THE RIGHT-END POINTS.
C
C     EXPLAN(J) = REAL*4 ARRAY.  ITS DIMENSION IS 20.  THIS ARRAY IS
C                 USED TO ALLOW EXPLANATORY COMMENTS IN THE INPUT FILES.
C
C     G(J) = REAL*4 ARRAY.  ITS DIMENSION MUST BE >= MAX(2*KMAX,M,N)
C            IT IS USED FOR HOLDING THE RANDOM VECTORS GENERATED,
C            HOLDING THE COMPUTED ERROR ESTIMATES AND THE COMPUTED
C            MINIMAL GAPS FOR THE SINGULAR VALUES.
C
C     MP(J) = INTEGER*4 ARRAY.  ITS DIMENSION MUST BE AT LEAST KMAX,
C             THE MAXIMUM SIZE OF THE T-MATRICES ALLOWED.  IT CONTAINS
C             THE T-MULTIPLICITIES OF THE COMPUTED T-EIGENVALUES OF
C             THE T-MATRICES.  NOTE THAT 'SPURIOUS' EIGENVALUES
C             OF THE T-MATRICES ARE DENOTED BY A T-MULTIPLICITY OF
C             0.  T-EIGENVALUES THAT THE SUBROUTINE PRTEST HAS
C             IDENTIFIED AS 'GOOD' BUT HIDDEN ARE IDENTIFIED BY A
C             T-MULTIPLICITY OF -10 AND SUBSEQUENTLY ADDED TO THE LIST
C             OF COMPUTED SINGULAR VALUES.
C
C     NMEV(J) = INTEGER*4 ARRAY.  ITS DIMENSION MUST BE AT LEAST THE
C               NUMBER OF T-MATRICES ALLOWED.  IT CONTAINS THE ORDERS
C               OF THE T-MATRICES TO BE CONSIDERED.
C
C
C     OTHER ARRAYS
C
C     THE USER MUST SPECIFY IN THE SUBROUTINE USPEC WHATEVER ARRAYS
C     ARE REQUIRED TO DEFINE THE MATRIX BEING USED.
C
C
C-----SUBROUTINES INCLUDED----------------------------------------------
C
C
C     LANCZS = COMPUTES THE BETA HISTORY. USES SUBROUTINES
C              FINPRO, GENRAN, SVMAT AND STRAN.
C
C     BISEC = COMPUTES EIGENVALUES OF THE SPECIFIED T-MATRIX USING
C             STURM SEQUENCING, ON SEQUENCE OF INTERVALS SPECIFIED
C             BY THE USER.  EACH SUBINTERVAL IS TREATED AS OPEN
C             ON THE LEFT AND CLOSED ON THE RIGHT.  EIGENVALUES
C             ARE COMPUTED WITH SIMULTANEOUS DETERMINATION OF THE
C             T-MULTIPLICITIES AND OF WHICH T-EIGENVALUES ARE SPURIOUS.
C
C     INVERR = USES INVERSE ITERATION ON T-MATRICES TO COMPUTE ERROR
C              ESTIMATES ON COMPUTED SINGULAR VALUES. (USES GENRAN)
C
C     LUMP = 'COMBINES' EIGENVALUES OF T-MATRIX USING THE RELATIVE
C             TOLERANCE RELTOL.
C
```

```
C     ISOEV = CALCULATES GAPS BETWEEN DISTINCT EIGENVALUES OF T-MATRIX
C             AND THEN USES THESE GAPS TO LABEL THOSE 'GOOD'
C             T-EIGENVALUES FOR WHICH ERROR ESTIMATES ARE NOT COMPUTED.
C
C     TNORM = COMPUTES THE SCALE TKMAX USED IN DETERMINING THE
C             TOLERANCES FOR THE SPURIOUS, T-MULTIPLICITY AND PRTESTS.
C             IT ALSO CHECKS FOR LOCAL ORTHOGONALITY OF THE LANCZOS
C             VECTORS BY TESTING THE RELATIVE SIZE OF THE BETAS USING
C             THE RELATIVE TOLERANCE BTOL.
C
C     PRTEST = LOOKS FOR 'GOOD' T-EIGENVALUES THAT HAVE BEEN MISLABELLED
C              BY THE SPURIOUS TEST BECAUSE THEY HAD 'TOO SMALL' A
C              PROJECTION ON THE STARTING LANCZOS VECTOR.
C              (LESS THAN SINGLE PRECISION)
C              TESTS INDICATE THAT SUCH T-EIGENVALUES ARE RARE.
C              PRTEST SHOULD BE CALLED ONLY AFTER CONVERGENCE
C              HAS BEEN ESTABLISHED.
C
C     INVERM = USED TO COMPUTE ERROR ESTIMATES FOR ANY T-EIGENVALUES
C              WHICH PRTEST INDICATES MAY HAVE BEEN MISLABELLED.
C              SUCH T-EIGENVALUES ARE RELABELLED ONLY IF THEIR ERROR
C              ESTIMATES ARE SUFFICIENTLY SMALL.  PRIMARY USE OF
C              INVERM IS IN THE CORRESPONDING SINGULAR VECTOR PROGRAM.
C
C     SAMPLE USPEC, SVMAT AND STRAN SUBROUTINES ARE INCLUDED.
C
C     ALSO INCLUDED IS A STAND-ALONE PROGRAM, LSCOMPAC, THAT
C     TRANSLATES A MATRIX GIVEN IN THE I,J, A(I,J) FORMAT INTO
C     THE PARTICULAR SPARSE MATRIX FORMAT USED IN THE SAMPLE USPEC,
C     SVMAT AND STRAN SUBROUTINES PROVIDED.
C
C
C-----OTHER PROGRAMS PROVIDED-------------------------------------------
C
C
C     LSCOMPAC = STAND-ALONE PROGRAM THAT TRANSLATES A SPARSE
C                RECTANGULAR M X N MATRIX A, GIVEN AS I, J, A(I,J),
C                INTO THE SPARSE MATRIX FORMAT REQUIRED BY THE SAMPLE
C                USPEC, STRAN AND SVMAT SUBROUTINES PROVIDED FOR USE
C                IN THE SINGULAR VALUE/VECTOR PROGRAMS.
C                THIS PROGRAM ASSUMES THAT THE MATRIX ENTRIES ARE
C                GIVEN EITHER COLUMN BY COLUMN OR ROW BY ROW.  IT
C                CANNOT HANDLE ANY OTHER ORDERINGS.  IN FACT IF
C                THE ENTRIES ARE GIVEN ROW BY ROW, THE DATA SET
C                CREATED ON FILE 8 CORRESPONDS TO A-TRANSPOSE AND
C                NOT TO A.  THUS, IN THIS SITUATION, IN ANY
C                SUBSEQUENT USE OF THE LANCZOS SINGULAR VALUE/VECTOR
C                PROGRAMS THE USER WILL HAVE TO INTERCHANGE THE
C                ROLES OF M AND OF N.
C
C
C-----COMMENTS ON THE STORAGE REQUIRED FOR SINGULAR VALUE PROGRAMS------
C
C
C     THE ARRAYS IN THE REAL SINGULAR VALUE PROGRAM REQUIRE
```

```
C     APPROXIMATELY THE EQUIVALENT OF ONE REAL*8 ARRAY OF DIMENSION     LSV07680
C                                                                       LSV07690
C        2.5*KMAX + MAX(KMAX,M) + MAX(KMAX,N) + .5* MAX(2*KMAX,M,N)     LSV07700
C                                                                       LSV07710
C     PLUS WHATEVER IS NEEDED TO GENERATE A*X FOR THE GIVEN MATRIX A.   LSV07720
C     THE ARRAYS BETA, VS AND MP CONSUME 2.5*KMAX*8 BYTES.              LSV07730
C     THE ARRAY V1 CONSUMES MAXIMUM(KMAX+1,M)*8 BYTES,  THE             LSV07740
C     ARRAY V2 CONSUMES MAXIMUM(KMAX,N)*8 BYTES, WITH THE               LSV07750
C     QUALIFICATION STATED ABOVE WHERE V2 IS DEFINED.  THE G-ARRAY      LSV07760
C     CONSUMES .5*MAX(2*KMAX,M,N)*8 BYTES.                              LSV07770
C                                                                       LSV07780
C                                                                       LSV07790
C-----------------------------------------------------------------------LSV07800
C                                                                       LSV07810
C     COMMENTS FOR SINGULAR VECTOR COMPUTATIONS                         LSV07820
C                                                                       LSV07830
C-----------------------------------------------------------------------LSV07840
C                                                                       LSV07850
C                                                                       LSV07860
C     THE SINGULAR VALUES WHOSE SINGULAR VECTORS ARE TO BE COMPUTED     LSV07870
C     MUST HAVE BEEN COMPUTED USING THE CORRESPONDING LANCZOS           LSV07880
C     SINGULAR VALUE PROGRAMS FOR REAL RECTANGULAR MATRICES BECAUSE     LSV07890
C     THESE SINGULAR VECTOR PROGRAMS USE THE SAME FAMILY OF LANCZOS     LSV07900
C     TRIDIAGONAL MATRICES THAT WAS USED IN THE CORRESPONDING           LSV07910
C     SINGULAR VALUE COMPUTATIONS.                                      LSV07920
C                                                                       LSV07930
C     THESE PROGRAMS ASSUME THAT THE SINGULAR VALUES SUPPLIED TO IT     LSV07940
C     HAVE BEEN COMPUTED ACCURATELY, AS MEASURED BY THE                 LSV07950
C     ERROR ESTIMATES COMPUTED IN THE CORRESPONDING LANCZOS             LSV07960
C     SINGULAR VALUE COMPUTATIONS, ALTHOUGH THESE ESTIMATES             LSV07970
C     ARE TYPICALLY CONSERVATIVE.   THE SINGULAR VALUES SUPPLIED        LSV07980
C     ARE STORED IN THE ARRAY GOODSV(J), J=1,NGOOD.                     LSV07990
C                                                                       LSV08000
C     FOR EACH GOODSV(J), THE SUBROUTINE STURMI COMPUTES THE            LSV08010
C     SMALLEST SIZE LANCZOS TRIDIAGONAL MATRIX, T(1,M1(J)), FOR         LSV08020
C     WHICH  GOODSV(J) IS A T-EIGENVALUE TO WITHIN A SPECIFIED          LSV08030
C     TOLERANCE.  IT ALSO ATTEMPTS TO COMPUTE THE SIZE, M2(J),          LSV08040
C     BY WHICH THE GIVEN SINGULAR VALUE BECOMES A DOUBLE                LSV08050
C     T-EIGENVALUE TO WITHIN THE GIVEN TOLERANCE.  THESE SIZES ARE      LSV08060
C     USED TO DETERMINE 1ST GUESSES AT SIZES FOR THE T-EIGENVECTORS     LSV08070
C     THAT WILL BE USED IN THE SINGULAR VECTOR COMPUTATIONS.            LSV08080
C     SUBROUTINE INVERM SUCCESSIVELY COMPUTES CORRESPONDING             LSV08090
C     T-EIGENVECTORS OF ENLARGED T-MATRICES UNTIL A SUITABLE            LSV08100
C     SIZE T-MATRIX IS DETERMINED FOR EACH J.  UP TO 10 SUCH            LSV08110
C     T-EIGENVECTOR COMPUTATIONS ARE ALLOWED FOR EACH SINGULAR          LSV08120
C     VALUE SUPPLIED.                                                   LSV08130
C                                                                       LSV08140
C     AFTER APPROPRIATE T-EIGENVECTORS HAVE BEEN COMPUTED,              LSV08150
C     RITZ VECTORS FOR THE MATRIX B CORRESPONDING TO THESE              LSV08160
C     T-EIGENVECTORS ARE THEN COMPUTED.  SECTIONS OF THESE              LSV08170
C     RITZ VECTORS ARE THEN TAKEN AS APPROXIMATE LEFT AND               LSV08180
C     RIGHT SINGULAR VECTORS CORRESPONDING TO THE GIVEN                 LSV08190
C     SINGULAR VALUES GOODSV(J),  J = 1,...,NGOOD.                      LSV08200
C                                                                       LSV08210
C     THIS IMPLEMENTATION FIRST COMPUTES ALL OF THE RELEVANT            LSV08220
```

```
C     T-EIGENVECTORS OF THE SYMMETRIC TRIDIAGONAL MATRICES
C     IN THE VECTOR, TVEC.
C
C     THEN, AS EACH OF THE LANCZOS VECTORS IS REGENERATED, ALL
C     OF THE B-MATRIX RITZ VECTORS CORRESPONDING TO THESE
C     T-EIGENVECTORS ARE UPDATED USING THE CURRENTLY-GENERATED
C     LANCZOS VECTOR.  LANCZOS VECTORS ARE GENERATED (NOTE
C     THAT THEY ARE NOT BEING KEPT), UNTIL ENOUGH HAVE
C     BEEN GENERATED TO MAP THE LONGEST T-EIGENVECTOR INTO ITS
C     CORRESPONDING B-MATRIX  RITZ VECTOR.  THE ARRAY RITVEC
C     CONTAINS THE SUCCESSIVE RITZ VECTORS WHICH ARE THEN
C     SPLIT INTO APPROXIMATIONS TO THE LEFT AND RIGHT SINGULAR
C     VECTORS OF THE USER-SUPPLIED MATRIX A.
C
C
C-----PARAMETER CONTROLS FOR SINGULAR VECTOR PROGRAMS-------------------
C
C
C     PARAMETER CONTROLS ARE INTRODUCED TO ALLOW SEGMENTATION OF THE
C     SINGULAR VECTOR COMPUTATIONS AND TO ALLOW VARIOUS COMBINATIONS
C     OF READ/WRITES.
C
C     THE FLAG MBOUND ALLOWS THE USER TO DETERMINE A FIRST GUESS ON THE
C     STORAGE THAT WILL BE REQUIRED BY THE T-EIGENVECTORS FOR THE
C     SINGULAR VALUES WHOSE SINGULAR VECTORS ARE TO BE COMPUTED.
C     THIS CAN BE USED TO ESTIMATE THE REQUIRED SIZE OF THE TVEC ARRAY.
C
C     MBOUND = (0,1) MEANS
C
C              (0)  PROGRAM COMPUTES FIRST GUESSES AT THE SIZES
C                   OF THE T-MATRICES REQUIRED BY EACH OF THE
C                   SINGULAR VALUES SUPPLIED AND THEN CONTINUES
C                   WITH THE CORRESPONDING T-EIGENVECTOR
C                   COMPUTATIONS.
C
C              (1)  PROGRAM COMPUTES FIRST GUESSES AT THE SIZES
C                   OF THE T-MATRICES REQUIRED BY EACH OF THE
C                   SINGULAR VALUES SUPPLIED, STORES THESE IN FILE
C                   10 AND THEN TERMINATES.  THE USER CAN USE THESE
C                   SIZES TO ESTIMATE THE SIZE TVEC ARRAY NEEDED
C                   FOR THE DESIRED T-EIGENVECTOR COMPUTATIONS.
C
C     THE FLAGS NTVCON, TVSTOP, LVCONT, AND ERCONT CONTROL THE STOPPING
C     CRITERIA FOR INTERMEDIATE POINTS IN THE LANCZOS PROCEDURE.  THEY
C     TERMINATE THE PROCEDURE IF VARIOUS SPECIFIED QUANTITIES COULD
C     NOT BE COMPUTED AS DESIRED.
C
C     NTVCON = (0,1) MEANS
C
C               (0)  IF THE ESTIMATED STORAGE FOR THE T-EIGENVECTORS
C                    EXCEEDS THE USER-SPECIFIED DIMENSION OF THE
C                    TVEC ARRAY PROGRAM DOES NOT CONTINUE WITH THE
C                    T-EIGENVECTOR COMPUTATIONS. TERMINATION OCCURS.
C
C               (1)  CONTINUE WITH THE T-EIGENVECTOR COMPUTATIONS
```

```
C                   EVEN IF THE ESTIMATED STORAGE FOR TVEC EXCEEDS
C                   THE USER-SPECIFIED DIMENSION OF THE TVEC ARRAY.
C                   IN THIS SITUATION THE PROGRAM COMPUTES AS MANY
C                   T-EIGENVECTORS AS IT HAS ROOM FOR, IN THE SAME
C                   ORDER IN WHICH THE SINGULAR VALUES ARE SUPPLIED.
C
C      SVTVEC = (0,1) MEANS
C
C              (0)  DO NOT STORE THE COMPUTED T-EIGENVECTORS ON
C                   FILE 11 UNLESS ALSO HAVE THE FLAG TVSTOP = 1,
C                   IN WHICH CASE THE T-EIGENVECTORS ARE ALWAYS
C                   WRITTEN TO FILE 11.
C
C              (1)  STORE THE COMPUTED T-EIGENVECTORS ON FILE 11.
C
C      TVSTOP = (0,1) MEANS
C
C              (0)  ATTEMPT TO CONTINUE ON TO THE COMPUTATION
C                   OF THE B-MATRIX RITZVECTORS AFTER COMPLETING THE
C                   COMPUTATION OF THE T-EIGENVECTORS.
C
C              (1)  TERMINATE AFTER COMPUTING THE
C                   T-EIGENVECTORS AND STORING THEM ON FILE 11.
C
C      LVCONT = (0,1) MEANS
C
C              (0)  IF SOME OF THE T-EIGENVECTORS THAT WERE
C                   REQUESTED WERE NOT COMPUTED, EXIT
C                   FROM THE PROGRAM WITHOUT COMPUTING THE
C                   CORRESPONDING RITZ VECTORS.
C
C              (1)  CONTINUE ON TO THE RITZ VECTOR COMPUTATIONS
C                   EVEN IF NOT ALL OF THE T-EIGENVECTORS THAT
C                   WERE REQUESTED WERE COMPUTED.
C
C      ERCONT = (0,1) MEANS
C
C              (0)  PROGRAM WILL NOT COMPUTE THE RITZ
C                   VECTOR FOR ANY SINGULAR VALUE FOR WHICH NO
C                   T-EIGENVECTOR WHICH SATISFIES THE ERROR
C                   ESTIMATE TEST (ERTOL) HAS BEEN IDENTIFIED.
C
C              (1)  A RITZ VECTOR WILL BE COMPUTED FOR EVERY
C                   SINGULAR VALUE FOR WHICH A T-EIGENVECTOR HAS BEEN
C                   COMPUTED REGARDLESS OF WHETHER OR NOT THAT
C                   T-EIGENVECTOR SATISFIES THE ERROR ESTIMATE TEST.
C
C
C-----INPUT/OUTPUT FILES FOR THE SINGULAR VECTOR COMPUTATIONS-----------
C
C
C     ANY INPUT DATA OTHER THAN THE T-MATRIX HISTORY FILE AND THE
C     PREVIOUSLY COMPUTED SINGULAR VALUES AND ERROR ESTIMATES
C     SHOULD BE STORED ON FILE 5 IN FREE FORMAT.  SEE SAMPLE
C     INPUT/OUTPUT FOR TYPICAL INPUT FILE.
```

```
C
C     FILE 6 WAS USED AS THE INTERACTIVE TERMINAL OUTPUT FILE.
C     THIS FILE PROVIDES A RUNNING ACCOUNT OF THE PROGRESS OF THE
C     COMPUTATIONS.  ADDITIONAL PRINTOUT IS GENERATED WHEN
C     THE FLAG IWRITE = 1.
C
C
C DESCRIPTION OF OTHER I/O FILES
C
C FILE (K)     CONTAINS:
C
C      (2)     INPUT FILE:
C              PREVIOUSLY-GENERATED T-MATRICES (BETA ARRAY)
C              AND THE FINAL TWO LANCZOS VECTORS USED ON THAT
C              COMPUTATION.  THIS PROGRAM ALLOWS ENLARGEMENT
C              OF ANY T-MATRICES PROVIDED ON FILE 2.
C
C      (3)     INPUT FILE:
C              THE SINGULAR VALUES FOR WHICH CORRESPONDING
C              SINGULAR VECTORS ARE REQUESTED.  FILE 3 ALSO
C              CONTAINS THE T-MULTIPLICITIES OF THESE SINGULAR
C              VALUES (AS T-EIGENVALUES) AND THEIR COMPUTED GAPS
C              BOTH THE T-MATRICES AND IN THE USER-SUPPLIED MATRIX.
C              THIS FILE IS CREATED IN THE LANCZOS SINGULAR
C              VALUE COMPUTATIONS.
C
C      (4)     INPUT FILE:
C              ERROR ESTIMATES FOR THE ISOLATED SINGULAR VALUES
C              OF FILE 3.  THIS FILE IS CREATED DURING THE LANCZOS
C              SINGULAR VALUE COMPUTATIONS.
C
C      (8)     INPUT FILE:
C              USPEC SUBROUTINE ASSUMES THAT THE USER-
C              SUPPLIED MATRIX IS ON FILE 8.
C
C      (9)     OUTPUT FILE:
C              ERROR ESTIMATES FOR THE COMPUTED RITZ VECTORS CONSIDERED
C              AS EIGENVECTORS OF THE B-MATRIX. THESE ESTIMATES
C              ARE OF THE FORM
C                    BERROR = || B*RITVEC - SVAL*RITVEC ||
C              WHERE B DENOTES THE M+N ORDER SYMMETRIC MATRIX
C              ASSOCIATED WITH THE USER-SUPPLIED MATRIX A, SVAL
C              DENOTES THE SINGULAR VALUE BEING CONSIDERED AND
C              RITVEC DENOTES THE ASSOCIATED COMPUTED RITZ VECTOR.
C
C      (10)    OUTPUT FILE:
C              GUESSES AT APPROPRIATE SIZE T-MATRICES FOR THE
C              T-EIGENVECTORS FOR EACH SUPPLIED SINGULAR VALUE
C              IN THE ARRAY GOODSV(J),  J = 1,...,NGOOD.
C
C      (11)    OUTPUT FILE:
C              COMPUTED T-EIGENVECTORS CORRESPONDING TO SINGULAR
C              VALUES IN THE GOODSV ARRAY.  NOTE THAT IT IS POSSIBLE
C              IN CERTAIN SITUATIONS THAT FOR SOME SINGULAR VALUES
C              SUPPLIED IN THE GOODSV ARRAY A T-EIGENVECTOR WILL
```

```
C              NOT BE COMPUTED.
C
C      (12)    OUTPUT FILE:
C              CONTAINS COMPUTED RITZ VECTORS CORRESPONDING TO
C              THE T-EIGENVECTORS ON FILE 11. NOTE THAT IN
C              SOME SITUATIONS THAT FOR SOME SINGULAR VALUES IN
C              THE GOODSV ARRAY FOR WHICH T-EIGENVECTORS HAVE
C              BEEN COMPUTED NO CORRESPONDING RITZ VECTOR WILL
C              HAVE BEEN COMPUTED.
C
C      (13)    OUTPUT FILE:
C              ADDITIONAL INFORMATION ABOUT THE BOUNDS AND ERROR
C              ESTIMATES OBTAINED.
C
C
C-----SEEDS FOR SINGULAR VECTOR PROGRAMS--------------------------------
C
C     SEEDS FOR RANDOM NUMBER GENERATOR GENRAN
C                (1) SVSEED = INTEGER*4 SCALAR USED IN THE SUBROUTINE
C                             GENRAN TO GENERATE THE STARTING VECTOR FOR
C                             THE REGENERATION OF THE LANCZOS VECTORS.
C
C                (2) RHSEED = INTEGER*4 SCALAR USED IN THE SUBROUTINE
C                             GENRAN TO GENERATE A RANDOM VECTOR FOR
C                             USE IN SUBROUTINE INVERM.
C
C     USER SHOULD NOTE THAT SVSEED MUST BE THE SAME SEED THAT
C     WAS USED TO GENERATE THE T-MATRICES THAT WERE USED TO
C     COMPUTE THE SINGULAR VALUES WHOSE SINGULAR VECTORS ARE TO BE
C     COMPUTED.  SVSEED IS READ IN FROM FILE 3.
C
C
C-----USER-SPECIFIED PARAMETERS FOR THE SINGULAR VECTOR PROGRAMS--------
C
C
C     NGOOD   = NUMBER OF SINGULAR VALUES READ INTO THE GOODSV ARRAY
C               READ FROM FILE 3.
C
C     M       = ROW ORDER OF THE USER-SUPPLIED MATRIX.
C
C     N       = COLUMN ORDER OF THE USER-SUPPLIED MATRIX.
C
C     MEV     = SIZE OF THE T-MATRIX THAT WAS USED TO COMPUTE
C               THE SINGULAR VALUES WHOSE SINGULAR VECTORS ARE
C               REQUESTED.  MEV IS READ IN FROM FILE 3.
C
C     KMAX    =  SIZE OF THE T-MATRIX PROVIDED ON FILE 2.
C
C     MDIMTV  = MAXIMUM CUMULATIVE SIZE OF THE TVEC ARRAY ALLOWED
C               FOR ALL OF THE T-EIGENVECTORS REQUIRED.  MDIMTV
C               MUST NOT EXCEED THE USER-SPECIFIED DIMENSION OF
C               THE TVEC ARRAY.  PROGRAM CAN BE RUN WITH THE FLAG
C               MBOUND = 1 TO DETERMINE AN EDUCATED GUESS ON AN
C               APPROPRIATE DIMENSION FOR THE TVEC ARRAY.
C
```

```
C     MDIMRV = MAXIMUM CUMULATIVE SIZE OF THE RITVEC ARRAY ALLOWED
C              FOR ALL OF THE RITZ VECTORS TO BE COMPUTED. MDIMRV
C              MUST NOT EXCEED THE USER-SPECIFIED DIMENSION OF
C              THE RITVEC ARRAY.  MUST BE SELECTED SO THAT
C              THERE IS ENOUGH ROOM FOR A RITZ VECTOR FOR EVERY
C              GOODEV(J) READ INTO PROGRAM.  (>= NGOOD*(M+N))
C
C
C-----ARRAYS REQUIRED BY THE SINGULAR VECTOR PROGRAMS-------------------
C
C
C     BETA(J) = REAL*8 ARRAY.  ITS DIMENSION MUST BE AT LEAST
C                KMAXN+1,  WHERE KMAXN IS THE LARGEST SIZE T-MATRIX
C                CONSIDERED BY THE PROGRAM.  NOTE THAT KMAXN IS THE
C                LARGER OF THE SIZE OF THE BETA HISTORY PROVIDED
C                ON FILE 2 (IF ANY ) AND THE SIZE WHICH THE PROGRAM
C                SPECIFIES INTERNALLY, THIS LATTER IS ALWAYS
C                < = 11*MEV / 8  +  12, WHERE MEV IS THE SIZE
C                T-MATRIX THAT WAS USED IN THE CORRESPONDING
C                SINGULAR VALUE COMPUTATIONS.  BETA CONTAINS THE
C                NONZERO ENTRIES OF THE LANCZOS T-MATRICES.
C                BETA IS NOT DESTROYED IN THE COMPUTATIONS.
C                THE DIAGONAL ENTRIES OF THE T-MATRICES ARE ALL ZERO.
C
C     RITVEC(J) = REAL*8 ARRAY. IT DIMENSION MUST BE > = NGOOD*(M+N)
C                 WHERE THE USER-SUPPLIED MATRIX IS MXN
C                 AND NGOOD IS THE NUMBER OF SINGULAR VALUES WHOSE
C                 SINGULAR VECTORS ARE TO BE COMPUTED.  IT CONTAINS
C                 THE COMPUTED APPROXIMATE SINGULAR VECTORS OF A.
C                 THESE COMPUTED RITZ VECTORS ARE STORED ON FILE 12.
C
C     TVEC(J)   = REAL*8 ARRAY.  ITS DIMENSION MUST BE AT LEAST
C                 MTOL = |MA(1)| + |MA(2)| + ... + |MA(NGOOD)|
C                 WHERE NGOOD IS THE NUMBER OF SINGULAR VALUES BEING
C                 CONSIDERED AND |MA(J)| IS THE SIZE OF THE
C                 T-MATRIX BEING USED FOR THE B-MATRIX RITZ VECTOR
C                 COMPUTATION FOR GOODSV(J).  THESE SIZES
C                 ARE COMPUTED BY THE PROGRAM.  AN ESTIMATE OF
C                 MTOL CAN BE OBTAINED BY SETTING MBOUND = 1,
C                 RUNNING THE PROGRAM, AND THEN MULTIPLYING THE
C                 RESULTING TOTAL T-SIZE SPECIFIED BY 5/4.  THE TVEC
C                 ARRAY CONTAINS THE COMPUTED T-EIGENVECTORS.  IF
C                 THE FLAG SVTVEC = 1 OR THE FLAG TVSTOP = 1, THEN
C                 THESE VECTORS ARE SAVED ON FILE 11.
C
C     V1(J)     = REAL*8 ARRAY.  ITS DIMENSION MUST BE GREATER
C                 THAN THE MAXIMUM OF KMAX AND M, WHERE M IS
C                 THE ROW ORDER OF THE GIVEN MATRIX.  V1 IS USED
C                 IN THE SUBROUTINE INVERM AND IN THE REGENERATION
C                 OF THE LANCZOS VECTORS.
C
C     V2(J)     = REAL*8 ARRAY.  ITS DIMENSION MUST BE GREATER
C                 THAN MAX(KMAX,N), WHERE N IS THE COLUMN ORDER OF
C                 THE GIVEN MATRIX.  IT IS USED IN THE REGENERATION
C                 OF THE LANCZOS VECTORS AND IN SUBROUTINE INVERM.
```

```
C                                                                       LSV10980
C     GOODSV(J),  = REAL*8 ARRAYS EACH OF DIMENSION AT LEAST NGOOD.     LSV10990
C     SVNEW(J)      CONTAIN THE SINGULAR VALUES FOR WHICH               LSV11000
C                   SINGULAR VECTORS ARE REQUESTED.  SINGULAR VALUES    LSV11010
C                   IN GOODSV ARE READ IN FROM FILE 3.                  LSV11020
C                                                                       LSV11030
C     BMINGP(J), = REAL*4 ARRAYS OF DIMENSION AT LEAST NGOOD.           LSV11040
C     TMINGP(J)    CONTAIN, RESPECTIVELY, THE MINIMAL GAPS FOR          LSV11050
C                  CORRESPONDING SINGULAR VALUES IN GOODSV ARRAY IN     LSV11060
C                  B-MATRIX AND IN T-MATRIX.                            LSV11070
C                                                                       LSV11080
C     TERR(J), ERR(J),     = REAL*4 ARRAYS (EXCEPT TLAST WHICH IS       LSV11090
C     ERRDGP(J), TLAST(J)    REAL*8).  EACH MUST BE OF DIMENSION        LSV11100
C     RNORM(J), TBETA(J)     AT LEAST NGOOD.  USED TO STORE QUANTITIES  LSV11110
C                            GENERATED DURING THE COMPUTATIONS FOR      LSV11120
C                            LATER PRINTOUT.                            LSV11130
C                                                                       LSV11140
C     G(J)     = REAL*4 ARRAY WHOSE DIMENSION MUST BE AT LEAST          LSV11150
C                MAX(KMAX,M,N).  USED IN SUBROUTINE GENRAN TO HOLD      LSV11160
C                RANDOM NUMBERS NEEDED FOR THE LANCZOS VECTOR           LSV11170
C                REGENERATION AND FOR THE INVERSE ITERATION             LSV11180
C                COMPUTATIONS IN THE SUBROUTINE INVERM.                 LSV11190
C                                                                       LSV11200
C     MP(J) = INTEGER*4 ARRAY WHOSE DIMENSION IS AT LEAST NGOOD.        LSV11210
C             INITIALLY CONTAINS THE T-MULTIPLICITY OF THE SINGULAR     LSV11220
C             VALUE GOODSV(J) AS AN EIGENVALUE OF THE T-MATRIX.         LSV11230
C             USED TO FLAG SINGULAR VALUES FOR WHICH NO T-EIGENVECTOR   LSV11240
C             OR NO RITZ VECTOR IS TO BE COMPUTED.                      LSV11250
C                                                                       LSV11260
C     MA(J)   = INTEGER*4 ARRAYS EACH OF WHOSE DIMENSIONS               LSV11270
C               IS AT LEAST NGOOD.  USED IN DETERMINING                 LSV11280
C               AN APPROPRIATE T-MATRIX FOR EACH SINGULAR VALUE         LSV11290
C               IN GOODSV ARRAY.                                        LSV11300
C                                                                       LSV11310
C     MINT(J),MFIN(J) = INTEGER*4 ARRAYS WHOSE DIMENSIONS MUST BE AT    LSV11320
C                       LEAST NGOOD.  USED TO POINT TO THE BEGINNINGS   LSV11330
C                       AND THE ENDS OF THE COMPUTED EIGENVECTOR        LSV11340
C                       OF THE T-MATRIX, T(1,|MA(J)|).                  LSV11350
C                                                                       LSV11360
C     IDELTA(J)  = INTEGER*4 ARRAY WHOSE DIMENSION MUST BE AT           LSV11370
C                  LEAST NGOOD.  CONTAINS INCREMENTS USED IN LOOPS      LSV11380
C                  ON APPROPRIATE SIZE T-MATRIX FOR THE T-EIGENVECTOR   LSV11390
C                  COMPUTATIONS.                                        LSV11400
C                                                                       LSV11410
C                                                                       LSV11420
C-----SUBROUTINES INCLUDED FOR THE SINGULAR VECTOR COMPUTATIONS---------LSV11430
C                                                                       LSV11440
C                                                                       LSV11450
C     STURMI = FOR EACH GIVEN SINGULAR VALUE GOODSV(J) DETERMINES       LSV11460
C              THE SMALLEST SIZE T-MATRIX FOR WHICH GOODSV(J) IS        LSV11470
C              A T-EIGENVALUE (TO WITHIN A GIVEN TOLERANCE) AND IF      LSV11480
C              POSSIBLE THE SMALLEST SIZE T-MATRIX FOR WHICH            LSV11490
C              IT IS A DOUBLE T-EIGENVALUE (TO WITHIN THE SAME          LSV11500
C              TOLERANCE).  THE SIZE T-MATRIX USED IN THE               LSV11510
C              T-EIGENVECTOR COMPUTATIONS IS THEN DETERMINED BY         LSV11520
```

```
C           STARTING WITH AN INITIAL GUESS BASED ON THE            LSV11530
C           INFORMATION FROM STURMI, AND THEN LOOPING ON THE       LSV11540
C           SIZE OF THE T-EIGENVECTOR COMPUTATIONS.                LSV11550
C                                                                  LSV11560
C     LBISEC = RECOMPUTES THE VALUE OF THE GIVEN SINGULAR VALUE    LSV11570
C           AT THE T-SIZE SPECIFIED FOR THE T-EIGENVECTOR          LSV11580
C           COMPUTATION.  LBISEC IS A SIMPLIFICATION OF THE        LSV11590
C           BISEC SUBROUTINE USED IN THE LANCZOS SINGULAR          LSV11600
C           VALUE COMPUTATIONS.                                    LSV11610
C                                                                  LSV11620
C     INVERM = FOR THE T-SIZES CONSIDERED BY THE PROGRAM COMPUTES  LSV11630
C           THE CORRESPONDING EIGENVECTORS OF THESE T-MATRICES     LSV11640
C           CORRESPONDING TO THE USER-SUPPLIED SINGULAR VALUES     LSV11650
C           IN THE GOODSV ARRAY.                                   LSV11660
C                                                                  LSV11670
C     LANCZS AND TNORM  SUBROUTINES ARE ALSO USED HERE AS WELL AS  LSV11680
C     IN THE CORRESPONDING SINGULAR VALUE COMPUTATIONS.            LSV11690
C                                                                  LSV11700
C                                                                  LSV11710
C-----------------------------------------------------------------------LSV11720
```

6.3 LSVAL: Main Program, Eigenvalue Computations

```
C-----LSVAL  (SINGULAR VALUES OF REAL, RECTANGULAR MATRICES-------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C        REFERENCE: Cullum and Willoughby, Chapter 5
C        Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C        Applied Mathematics, 2002. SIAM Publications,
C        Philadelphia, PA. USA
C
C
C
C     CONTAINS MAIN PROGRAM FOR COMPUTING DISTINCT SINGULAR VALUES OF
C     A REAL M X N MATRIX USING LANCZOS TRIDIAGONALIZATION WITHOUT
C     REORTHOGONALIZATION AND WITH SPECIAL STARTING VECTORS.
C
C     FOR A GIVEN REAL MATRIX A OF ORDER M X N THE LANCZOS RECURSION
C     IS APPLIED TO THE ASSOCIATED REAL SYMMETRIC MATRIX B OF ORDER
C     MN = M + N
C
C                               ----                ----
C                              |  0                  A  |
C                     B  =     |                        |
C                              |  A-TRANSPOSE        0  |
C                               ----                ----
C
C     USING SPECIAL STARTING VECTORS.  PLEASE NOTE: ONLY EVEN ORDER
C     LANCZOS TRIDIAGONAL MATRICES AND ONLY NONNEGATIVE SUBINTERVALS
C     ARE PERMISSIBLE.
C
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE
C     CONSTRUCTIONS
C
C     1.  DATA/MACHEP/ STATEMENT
C     2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)
C     3.  FORMAT(20A4) USED WITH EXPLANATORY HEADER EXPLAN.
```

```
C     4.  HEXADECIMAL FORMAT (4Z20) USED IN BETA FILES.
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  BETA(5001),V1(5000),V2(5000),VS(5000)
      DOUBLE PRECISION  LB(20),UB(20)
      DOUBLE PRECISION  BTOL,GAPTOL,TTOL,MACHEP,EPSM,RELTOL
      DOUBLE PRECISION  SCALE1,SCALE2,SCALE3,SCALE4,BISTOL,CONTOL,MULTOL
      DOUBLE PRECISION  ONE,ZERO,TEMP,TKMAX,BETAM,BKMIN,T0,T1
      REAL  G(5000),EXPLAN(20)
      INTEGER  MP(5000),NMEV(20)
      INTEGER  SVSEED,RHSEED,SVSOLD
      INTEGER IABS
      REAL  ABS
      DOUBLE PRECISION  DABS, DSQRT, DFLOAT
      EXTERNAL SVMAT,STRAN
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.0D0*MACHEP
C-----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     1.  BETA: >= (KMAX+1) WHERE KMAX IS READ IN AND IS
C         THE SIZE OF THE LARGEST T-MATRIX THAT CAN BE CONSIDERED.
C     2.  V1:  >= MAX(M,KMAX+1)
C     3.  V2:  >= MAX(N,KMAX)
C     4.  VS:  >= KMAX
C     5.  G:  >= MAX(2*KMAX,M,N)
C     6.  MP:  >= KMAX
C     7.  LB,UB:  >= NUMBER OF SUBINTERVALS SUPPLIED TO BISEC.
C     8.  NMEV:  >= NUMBER OF T-MATRICES ALLOWED.
C     9.  EXPLAN:  DIMENSION IS 20.
C
C
C     IMPORTANT TOLERANCES OR SCALES THAT ARE USED REPEATEDLY
C     THROUGHOUT THIS PROGRAM ARE THE FOLLOWING:
C     SCALED MACHINE EPSILON:  TTOL = TKMAX*EPSM WHERE
C     EPSM = 2*MACHINE EPSILON AND
C     TKMAX = MAX(BETA(J), J = 1,MEV)
C     BISEC CONVERGENCE TOLERANCE:  BISTOL = DSQRT(1000+MEV)*TTOL
C     BISEC MULTIPLICITY TOLERANCE:  MULTOL = (1000+MEV)*TTOL
C     LANCZOS CONVERGENCE TOLERANCE:   CONTOL = BETA(MEV+1)*1.D-10
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LANCZOS PROCEDURE FOR REAL, RECTANGULAR MATRICES'/)
C
C     SET PROGRAM PARAMETERS
C     SCALEK ARE USED IN TOLERANCES NEEDED IN SUBROUTINES LUMP,
C     ISOEV AND PRTEST.  USER MUST NOT MODIFY THESE SCALES.
      SCALE1 = 5.0D2
      SCALE2 = 5.0D0
      SCALE3 = 5.0D0
      SCALE4 = 1.0D4
      ONE  = 1.0D0
      ZERO = 0.0D0
```

```
      BTOL = 1.0D-8
C     BTOL = EPSM
      GAPTOL = 1.0D-8
      ICONV = 0
      MOLD = 0
      MOLD1 = 1
      ICT = 0
      MMB = 0
      IPROJ = 0
C
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 5 (FREE FORMAT)
C
C     READ USER-PROVIDED HEADERS FOR RUN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C     READ THE ROW ORDER M OF THE MATRIX AND THE COLUMN ORDER N.
C     READ THE MAXIMUM ORDER OF THE T-MATRICES ALLOWED (KMAX),
C     THE NUMBER OF T-MATRICES ALLOWED (NMEVS), AND A
C     MATRIX IDENTIFICATION NUMBER (MATNO).
      READ(5,20) EXPLAN
      READ(5,*) M,N,KMAX,NMEVS,MATNO
      NM = M + N
C
C     READ SEEDS FOR LANCZS AND INVERR SUBROUTINES (SVSEED AND RHSEED)
C     READ MAXIMUM NUMBER OF ITERATIONS ALLOWED FOR EACH INVERSE
C     ITERATION (MXINIT) AND MAXIMUM NUMBER OF STURM SEQUENCES
C     ALLOWED (MXSTUR)
      READ(5,20) EXPLAN
      READ(5,*) SVSEED,RHSEED,MXINIT,MXSTUR
C
C     ISTART = (0,1):  ISTART = 0 MEANS BETA FILE IS NOT
C     AVAILABLE.  ISTART = 1 MEANS BETA FILE IS AVAILABLE ON
C     FILE 2.
C     ISTOP = (0,1):  ISTOP = 0 MEANS PROCEDURE GENERATES BETA
C     FILE AND THEN TERMINATES.  ISTOP = 1 MEANS PROCEDURE GENERATES
C     BETAS IF NEEDED AND THEN COMPUTES SINGULAR VALUES AND
C     ERROR ESTIMATES AND THEN TERMINATES.
      READ(5,20) EXPLAN
      READ(5,*) ISTART,ISTOP
C
C     IHIS = (0,1):  IHIS = 0 MEANS BETA FILE IS NOT WRITTEN
C     TO FILE 1.  IHIS = 1 MEANS BETA FILE IS WRITTEN TO FILE 1.
C     IDIST = (0,1):  IDIST = 0 MEANS DISTINCT T-EIGENVALUES
C     ARE NOT WRITTEN TO FILE 11.  IDIST = 1 MEANS DISTINCT
C     T-EIGENVALUES ARE WRITTEN TO FILE 11.
C     IWRITE = (0,1):  IWRITE = 0 MEANS NO INTERMEDIATE OUTPUT
C     FROM THE COMPUTATIONS IS WRITTEN TO FILE 6.  IWRITE = 1 MEANS
C     T-EIGENVALUES AND ERROR ESTIMATES ARE WRITTEN TO FILE 6
C     AS THEY ARE COMPUTED.  SPECIFY THE PARITY (IPAR) OF THE
C     LANCZOS STARTING VECTOR.  IF M > N, THEN IPAR = 1,
C     IF M < N, THEN IPAR = 2.
```

```
      READ(5,20) EXPLAN
      READ(5,*) IHIS,IDIST,IWRITE,IPAR
      IF(M.GT.N) IPAR = 1
      IF(M.LT.N) IPAR = 2
      IPARO = IPAR
C
C     READ IN THE RELATIVE TOLERANCE (RELTOL) FOR USE IN THE
C     SPURIOUS, T-MULTIPLICITY, AND PRTEST TESTS.
      READ(5,20) EXPLAN
      READ(5,*) RELTOL
C
C     READ IN THE SIZES OF THE T-MATRICES TO BE CONSIDERED.
C     NOTE THAT ONLY EVEN ORDER T-SIZES ARE PERMISSIBLE.
      READ(5,20) EXPLAN
      READ(5,*) (NMEV(J), J=1,NMEVS)
C
C     CHECK TO SEE THAT ALL T-SIZES PROVIDED ARE EVEN ORDERED.
C     TERMINATE IF THAT IS NOT THE CASE.
      DO 30 I = 1,NMEVS
      NMEV2 = NMEV(I)/2
      IF(2*NMEV2.NE.NMEV(I)) GO TO 670
   30 CONTINUE
C
C     READ IN THE NUMBER OF SUBINTERVALS TO BE CONSIDERED.
      READ(5,20) EXPLAN
      READ(5,*) NINT
C
C     READ IN THE LEFT-END POINTS OF THE SUBINTERVALS TO BE CONSIDERED.
C     THESE MUST BE IN ALGEBRAICALLY INCREASING ORDER
      READ(5,20) EXPLAN
      READ(5,*) (LB(J), J=1,NINT)
C
C     READ IN THE RIGHT-END POINTS OF THE SUBINTERVALS TO BE CONSIDERED.
C     THESE MUST BE IN ALGEBRAICALLY INCREASING ORDER
      READ(5,20) EXPLAN
      READ(5,*) (UB(J), J=1,NINT)
C
C-----------------------------------------------------------------------
C     INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED MATRIX
C     AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE
C     MATRIX-VECTOR MULTIPLY SUBROUTINES SVMAT AND STRAN.
C
      CALL USPEC(M,N,MATNO)
C
C-----------------------------------------------------------------------
C     MASK UNDERFLOW AND OVERFLOW
C
      CALL MASK
C
C-----------------------------------------------------------------------
C
C     WRITE TO FILE 6, A SUMMARY OF THE PARAMETERS FOR THIS RUN
C
      WRITE(6,40) MATNO,M,N,KMAX
   40 FORMAT(/3X,'MATRIX ID',5X,'M',5X,'N',4X,'MAX ORDER OF T'/
```

```
     1 I12,2I6,I18/)
C
      WRITE(6,50) ISTART,ISTOP
   50 FORMAT(/2X,'ISTART',3X,'ISTOP'/2I8/)
C
      WRITE(6,60) IHIS,IDIST,IWRITE,IPAR
   60 FORMAT(/4X,'IHIS',3X,'IDIST',2X,'IWRITE',4X,'IPAR'/4I8/)
C
      WRITE(6,70) SVSEED,RHSEED
   70 FORMAT(/' SEEDS FOR RANDOM NUMBER GENERATOR'//
     1 4X,'LANCZS SEED',4X,'INVERR SEED'/2I15/)
C
      WRITE(6,80) (NMEV(J), J=1,NMEVS)
   80 FORMAT(/' SIZES OF T-MATRICES TO BE CONSIDERED'/(6I12))
C
      WRITE(6,90) RELTOL,GAPTOL,BTOL
   90 FORMAT(/' RELATIVE TOLERANCE USED TO COMBINE COMPUTED T-EIGENVALUE
     1S'/E15.3/' RELATIVE GAP TOLERANCES USED IN INVERSE ITERATION'/
     1E15.3/' RELATIVE TOLERANCE FOR CHECK ON SIZE OF BETAS'/E15.3/)
C
      WRITE(6,100) (J,LB(J),UB(J), J=1,NINT)
  100 FORMAT(/' BISEC WILL BE USED ON THE FOLLOWING INTERVALS'/
     1 (I6,2E20.6)/)
C
      IF (ISTART.EQ.0.AND.IPAR.EQ.1) WRITE(6,110)
      IF (ISTART.EQ.0.AND.IPAR.EQ.2) WRITE(6,120)
  110 FORMAT(/' STARTING VECTOR IS OF FORM (0,V2)'/)
  120 FORMAT(/' STARTING VECTOR IS OF FORM (V1,0)'/)
C
      IF (ISTART.EQ.0) GO TO 170
C
C     READ IN BETA HISTORY FROM FILE 2
C
      READ(2,130)MOLD,MO,NO,IPARO,IPAR,SVSOLD,MATOLD
  130 FORMAT(3I6,2I3,I12,I8)
C
      IF (KMAX.LT.MOLD) KMAX = MOLD
      KMAX1 = KMAX + 1
C
C     CHECK THAT M, N, MATRIX ID MATNO, AND RANDOM SEED SVSEED
C     AGREE WITH THOSE IN THE HISTORY FILE.  IF NOT PROCEDURE STOPS.
C
      ITEMP = (MO-M)**2+(NO-N)**2+(MATNO-MATOLD)**2+(SVSEED-SVSOLD)**2
C
      IF (ITEMP.EQ.0) GO TO 150
C
      WRITE(6,140)
  140 FORMAT(' PROGRAM TERMINATES'/    ' READ FROM FILE 2 CORRESPONDS TO
     1 DIFFERENT MATRIX THAN MATRIX SPECIFIED'/)
      GO TO 690
C
  150 CONTINUE
      MOLD1 = MOLD+1
C
      READ(2,160)(BETA(J), J=1,MOLD1)
```

```
  160 FORMAT(4Z20)                                                      LSV02650
C                                                                       LSV02660
      IF (KMAX.EQ.MOLD) GO TO 190                                       LSV02670
C                                                                       LSV02680
      READ(2,160)(V1(J), J=1,M)                                         LSV02690
      READ(2,160)(V2(J), J=1,N)                                         LSV02700
C                                                                       LSV02710
  170 CONTINUE                                                          LSV02720
      IIX = SVSEED                                                      LSV02730
C                                                                       LSV02740
C-----------------------------------------------------------------------LSV02750
C                                                                       LSV02760
      CALL LANCZS(SVMAT,STRAN,BETA,V1,V2,G,KMAX,MOLD1,M,N,IPAR,IIX)     LSV02770
C                                                                       LSV02780
C-----------------------------------------------------------------------LSV02790
C                                                                       LSV02800
      KMAX1 = KMAX + 1                                                  LSV02810
C                                                                       LSV02820
      IF (IHIS.EQ.0.AND.ISTOP.GT.0) GO TO 190                           LSV02830
C                                                                       LSV02840
      WRITE(1,180) KMAX,M,N,IPARO,IPAR,SVSEED,MATNO                     LSV02850
  180 FORMAT(3I6,2I3,I12,I8,' = KMAX,M,N,IPARO,IPAR,SVSEED,MATNO')      LSV02860
C                                                                       LSV02870
      WRITE(1,160)(BETA(I), I=1,KMAX1)                                  LSV02880
C                                                                       LSV02890
      WRITE(1,160)(V1(I), I=1,M)                                        LSV02900
      WRITE(1,160)(V2(I), I=1,N)                                        LSV02910
C                                                                       LSV02920
      IF (ISTOP.EQ.0) GO TO 570                                         LSV02930
C                                                                       LSV02940
  190 CONTINUE                                                          LSV02950
      BKMIN = BTOL                                                      LSV02960
      WRITE(6,200)                                                      LSV02970
  200 FORMAT(/' T-MATRICES (BETA) ARE NOW AVAILABLE'/)                  LSV02980
C                                                                       LSV02990
C-----------------------------------------------------------------------LSV03000
C     SUBROUTINE TNORM CHECKS MIN(BETA)/(ESTIMATED NORM(A)) > BTOL .    LSV03010
C     IF THIS IS VIOLATED IB IS SET EQUAL TO THE NEGATIVE OF THE INDEX  LSV03020
C     OF THE MINIMAL BETA.  IF(IB < 0) THEN SUBROUTINE TNORM IS         LSV03030
C     CALLED FOR EACH VALUE OF MEV TO DETERMINE WHETHER OR NOT THERE    LSV03040
C     IS A BETA IN THE T-MATRIX SPECIFIED THAT VIOLATES THIS TEST.      LSV03050
C     IF THERE IS SUCH A BETA THE PROGRAM TERMINATES FOR THE USER       LSV03060
C     TO DECIDE WHAT TO DO.  THIS TEST CAN BE OVER-RIDDEN BY            LSV03070
C     SIMPLY MAKING BTOL SMALLER, BUT THEN THERE IS THE POSSIBILITY     LSV03080
C     THAT LOSSES IN THE LOCAL ORTHOGONALITY MAY HURT THE COMPUTATIONS. LSV03090
C     BTOL = 1.D-8 IS HOWEVER A CONSERVATIVE CHOICE FOR BTOL.           LSV03100
C                                                                       LSV03110
C     TNORM ALSO COMPUTES TKMAX = MAX(BETA(K), K=1,KMAX).               LSV03120
C     TKMAX IS USED TO SCALE THE TOLERANCES USED IN THE                 LSV03130
C     T-MULTIPLICITY AND SPURIOUS TESTS IN BISEC. TKMAX IS ALSO USED IN LSV03140
C     THE PROJECTION TEST FOR HIDDEN T-EIGENVALUES THAT HAD 'TOO SMALL' LSV03150
C     A PROJECTION ON THE STARTING VECTOR.                              LSV03160
C                                                                       LSV03170
      CALL TNORM(BETA,BKMIN,TKMAX,KMAX,IB)                              LSV03180
C                                                                       LSV03190
```

```
C-----------------------------------------------------------------------
C
      TTOL = EPSM*TKMAX
C
C     LOOP ON THE SIZE OF THE T-MATRIX
C
  210 CONTINUE
      MMB = MMB + 1
C     NOTE THAT ONLY EVEN ORDER T-SIZES ARE PERMISSIBLE.
      MEV = NMEV(MMB)
C     IS MEV TOO LARGE ?
      IF(MEV.LE.KMAX) GO TO 230
      WRITE(6,220) MMB, MEV, KMAX
  220 FORMAT(/' TERMINATE PRIOR TO CONSIDERING THE',I6,'TH T-MATRIX'/
     1' BECAUSE THE SIZE REQUESTED',I6,' IS GREATER THAN THE MAXIMUM SIZ
     1E ALLOWED',I6/)
      GO TO 570
C
  230 MP1 = MEV + 1
      BETAM = BETA(MP1)
C
      IF (IB.GE.0) GO TO 240
C
      T0 = BTOL
C
C-----------------------------------------------------------------------
C
      CALL TNORM(BETA,T0,T1,MEV,IBMEV)
C
C-----------------------------------------------------------------------
C
      TEMP = T0/TKMAX
      IBMEV = IABS(IBMEV)
      IF (TEMP.GE.BTOL) GO TO 240
      IBMEV = -IBMEV
      GO TO 630
C
  240 CONTINUE
      IC = MXSTUR-ICT
C
C-----------------------------------------------------------------------
C     BISEC LOOP. THE SUBROUTINE BISEC INCORPORATES DIRECTLY THE
C     T-MULTIPLICITY AND SPURIOUS TESTS. T-EIGENVALUES WILL BE
C     CALCULATED BY BISEC SEQUENTIALLY ON INTERVALS
C     (LB(J),UB(J)), J = 1,NINT).
C
C     ON RETURN FROM BISEC
C     NDIS = NUMBER OF DISTINCT EIGENVALUES OF T(1,MEV) ON UNION
C            OF THE (LB,UB) INTERVALS
C     VS = DISTINCT T-EIGENVALUES IN ALGEBRAICALLY INCREASING ORDER
C     MP = T-MULTIPLICITIES OF THE T-EIGENVALUES STORED IN VS
C     MP(I) = (0,1,MI), MI>1, I=1,NDIS  MEANS:
C        (0)  VS(I) IS SPURIOUS
C        (1)  VS(I) IS T-SIMPLE AND GOOD
C        (MI) VS(I) IS T-MULTIPLE AND IS THEREFORE NOT ONLY GOOD BUT
```

```
C           ALSO A CONVERGED GOOD T-EIGENVALUE.
C
C
      CALL BISEC(BETA,V1,V2,VS,LB,UB,EPSM,TTOL,MP,NINT,
     1 MEV,NDIS,IC,IWRITE)
C
C-----------------------------------------------------------------------
C
      IF (NDIS.EQ.0) GO TO 650
C
C     COMPUTE THE TOTAL NUMBER OF STURM SEQUENCES USED TO DATE
C     COMPUTE THE BISEC CONVERGENCE AND T-MULTIPLICITY TOLERANCES USED.
C     COMPUTE THE CONVERGENCE TOLERANCE FOR T-EIGENVALUES.
      ICT = ICT + IC
      TEMP = DFLOAT(MEV+1000)
      MULTOL = TEMP*TTOL
      TEMP = DSQRT(TEMP)
      BISTOL = TTOL*TEMP
      CONTOL = BETAM*1.D-10
C
C-----------------------------------------------------------------------
C    SUBROUTINE LUMP 'COMBINES' T-EIGENVALUES THAT ARE 'TOO CLOSE'.
C    NOTE HOWEVER THAT CLOSE SPURIOUS T-EIGENVALUES ARE NOT AVERAGED
C    WITH GOOD ONES. HOWEVER, THEY MAY BE USED TO INCREASE THE
C    T-MULTIPLICITY OF A GOOD T-EIGENVALUE.
C
      LOOP = NDIS
      CALL LUMP(VS,RELTOL,MULTOL,SCALE2,MP,LOOP)
C
C-----------------------------------------------------------------------
C
      IF(NDIS.EQ.LOOP) GO TO 260
C
      WRITE(6,250) NDIS, MEV, LOOP
  250 FORMAT(/I6,' DISTINCT T-EIGENVALUES WERE COMPUTED IN BISEC AT MEV
     1=',I6/ 2X,' LUMP SUBROUTINE REDUCES NUMBER OF DISTINCT T-EIGENVALU
     1ES TO',I6)
C
  260 CONTINUE
      NDIS = LOOP
      BETA(MP1) = BETAM
C
C-----------------------------------------------------------------------
C     THE SUBROUTINE ISOEV LABELS THOSE SIMPLE T-EIGENVALUES OF T(1,MEV)
C     WITH VERY SMALL GAPS BETWEEN NEIGHBORING T-EIGENVALUES OF T(1,MEV)
C     TO AVOID COMPUTING ERROR ESTIMATES FOR ANY SIMPLE GOOD
C     T-EIGENVALUE THAT IS TOO CLOSE TO A SPURIOUS T-EIGENVALUE.
C     ON RETURN FROM ISOEV, G CONTAINS CODED MINIMAL GAPS
C     BETWEEN THE DISTINCT EIGENVALUES OF T(1,MEV). (G IS REAL).
C     G(I) < 0 MEANS MINGAP IS DUE TO LEFT GAP G(I) > 0 MEANS DUE TO
C     RIGHT GAP. MP(I) = -1 MEANS THAT THE GOOD T-EIGENVALUE IS SIMPLE
C     AND HAS A VERY SMALL MINGAP IN T(1,MEV) DUE TO A SPURIOUS
C     T-EIGENVALUE.
C     NG = NUMBER OF GOOD T-EIGENVALUES.
C     NISO = NUMBER OF ISOLATED, GOOD T-EIGENVALUES.
```

```
C
      CALL ISOEV(VS,GAPTOL,MULTOL,SCALE1,G,MP,NDIS,NG,NISO)
C
C-----------------------------------------------------------------------
C
      WRITE(6,270)NG,NISO,NDIS
  270 FORMAT(/I6,' SINGULAR VALUES HAVE BEEN COMPUTED'/
     1 I6,' OF THESE ARE ISOLATED'/
     2 I6,' = NUMBER OF DISTINCT T-EIGENVALUES COMPUTED'/)
C
C     DO WE WRITE DISTINCT T-EIGENVALUES TO FILE 11?
      IF (IDIST.EQ.0) GO TO 310
C
      WRITE(11,280) NDIS,NISO,MEV,M,N,SVSEED,MATNO
  280 FORMAT(5I5,I12,I8,' = NDIS,NISO,MEV,M,N,SVSEED,MATNO'/)
C
      WRITE(11,290) (MP(I),VS(I),G(I), I=1,NDIS)
  290 FORMAT(2(I3,E25.16,E12.3))
C
      WRITE(11,300) NDIS, (MP(I), I=1,NDIS)
  300 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/(20I4))
C
  310 CONTINUE
C
      IF (NISO.NE.0) GO TO 340
C
      WRITE(4,320) MEV
  320 FORMAT(/' AT MEV = ',I6,' THERE ARE NO ISOLATED T-EIGENVALUES'/
     1' SO NO ERROR ESTIMATES WERE COMPUTED/')
C
      WRITE(6,330)
  330 FORMAT(/' ALL COMPUTED SINGULAR VALUES ARE T-MULTIPLE'/
     1 ' THEREFORE ALL COMPUTED SINGULAR VALUES ARE ASSUMED TO HAVE CONV
     1ERGED'/)
C
      ICONV = 1
      GO TO 380
C
  340 CONTINUE
C
C-----------------------------------------------------------------------
C     SUBROUTINE INVERR COMPUTES ERROR ESTIMATES FOR ISOLATED GOOD
C     T-EIGENVALUES USING INVERSE ITERATION ON T(1,MEV). ON RETURN
C     G(J) = MINIMUM GAP IN T(1,MEV) FOR EACH VS(J), J=1,NDIS
C     G(MEV+I) = BETAM*|U(MEV)| = ERROR ESTIMATE FOR ISOLATED GOOD
C              T-EIGENVALUES, WHERE I = 1, NISO AND  BETAM = BETA(MEV+1)
C              U(MEV) IS MEVTH COMPONENT OF THE UNIT EIGENVECTOR OF T
C              CORRESPONDING TO THE ITH ISOLATED GOOD T-EIGENVALUE.
C     A NEGATIVE ERROR ESTIMATE MEANS THAT FOR THAT PARTICULAR
C     T-EIGENVALUE THE INVERSE ITERATION DID NOT CONVERGE IN <= MXINIT
C     STEPS AND THAT THE CORRESPONDING ERROR ESTIMATE IS QUESTIONABLE.
C
C     V2 CONTAINS THE ISOLATED GOOD T-EIGENVALUES
C     V1 CONTAINS THE MINGAPS TO THE NEAREST DISTINCT  EIGENVALUE
C        OF T(1,MEV) FOR EACH ISOLATED GOOD T-EIGENVALUE IN V2.
```

```
C     VS CONTAINS THE NDIS DISTINCT EIGENVALUES OF T(1,MEV)
C     MP CONTAINS THE CORRESPONDING CODED T-MULTIPLICITIES
C
      IT = MXINIT
      CALL INVERR(BETA,V1,V2,VS,EPSM,G,MP,MEV,MMB,NDIS,NISO,NM,
     1 RHSEED,IT,IWRITE)
C
C-----------------------------------------------------------------------
C
C     SIMPLE CHECK FOR CONVERGENCE. CHECKS TO SEE IF ALL OF THE ERROR
C     ESTIMATES ARE SMALLER THAN CONTOL.
C     IF THIS TEST IS SATISFIED, THEN CONVERGENCE FLAG, ICONV IS SET
C     TO 1.  TYPICALLY ERROR ESTIMATES ARE VERY CONSERVATIVE.
C
      WRITE(6,350) CONTOL
  350 FORMAT(/' CONVERGENCE IS TESTED USING THE CONVERGENCE TOLERANCE',
     1E13.4/)
C
      II = MEV +1
      IF = MEV+NISO
      DO 360 I = II,IF
      IF (ABS(G(I)).GT.CONTOL) GO TO 380
  360 CONTINUE
      ICONV = 1
      MMB = NMEVS
C
      WRITE(6,370) CONTOL
  370 FORMAT(' ALL COMPUTED ERROR ESTIMATES WERE LESS THAN',E15.4/
     1 ' THEREFORE PROCEDURE TERMINATES'/)
C
  380 CONTINUE
C
C     IF CONVERGENCE IS INDICATED, THAT IS ICONV = 1 ,THEN
C     THE SUBROUTINE PRTEST IS CALLED TO CHECK FOR ANY CONVERGED
C     T-EIGENVALUES THAT HAVE BEEN MISLABELLED AS SPURIOUS BECAUSE
C     THE PROJECTION OF THEIR SINGULAR VECTOR ON THE STARTING
C     VECTOR WAS TOO SMALL.  NUMERICAL TESTS INDICATE THAT
C     SUCH SINGULAR VALUES ARE RARE.  THEREFORE, IF MANY OF
C     THESE HIDDEN SINGULAR VALUES APPEAR ON SOME RUN, THE USER
C     CAN BE CERTAIN THAT SOMETHING IS FOULED UP.
C
      IF (ICONV.EQ.0) GO TO 510
C
C-----------------------------------------------------------------------
C
      CALL PRTEST (BETA,VS,TKMAX,EPSM,RELTOL,SCALE3,SCALE4,
     1 MP,NDIS,MEV,IPROJ)
C
C-----------------------------------------------------------------------
C
      IF(IPROJ.EQ.0) GO TO 500
C
      IF(IDIST.EQ.1)  WRITE(11,390) IPROJ
  390 FORMAT(' SUBROUTINE PRTEST WANTS TO RELABEL',I6,' SPURIOUS T-EIGEN
     1VALUES'/' WE ACCEPT RELABELLING ONLY IF LAST COMPONENT OF T-EIGENV
```

```
     1ECTOR IS L.T. 1.D-10')
C
      IIX = RHSEED
C
C-----------------------------------------------------------------------
C
      CALL GENRAN(IIX,G,MEV)
C
C-----------------------------------------------------------------------
C
      ITEN = -10
      NISOM = NISO + MEV
      IWRITO = IWRITE
      IWRITE = 0
C
      DO 420 J = 1,NDIS
      IF(MP(J).NE.ITEN) GO TO 420
      T0 = VS(J)
C
C-----------------------------------------------------------------------
C
      IT = MXINIT
      CALL INVERM(BETA,V1,V2,T0,TEMP,T1,EPSM,G,MEV,IT,IWRITE)
C
C-----------------------------------------------------------------------
C
      IF(TEMP.LE.1.D-10) GO TO 410
C     ERROR ESTIMATE WAS NOT SMALL REJECT RELABELLING OF THIS
C     T-EIGENVALUE.
      IF(IDIST.EQ.1) WRITE(11,400) J,T0,TEMP
  400 FORMAT(/' LAST COMPONENT FOR',I6,'TH T-EIGENVALUE',E20.12/' IS TOO
     1 LARGE = ',E15.6,' SO DO NOT ACCEPT PRTEST RELABELLING'/)
      MP(J) = 0
      IPROJ = IPROJ - 1
      GO TO 420
C     RELABELLING ACCEPTED
  410 NISOM = NISOM + 1
      G(NISOM) = BETAM*TEMP
  420 CONTINUE
      IWRITE = IWRITO
C
      IF(IPROJ.EQ.0) GO TO 460
      WRITE(6,430) IPROJ
  430 FORMAT(/I6,' T-EIGENVALUES WERE RECLASSIFIED AS GOOD.'/
     1' THESE ARE IDENTIFIED IN FILE 3 BY A T-MULTIPLICITY OF -10'/' USE
     2R SHOULD INSPECT EACH TO MAKE SURE NEIGHBORS HAVE CONVERGED'/)
C
      IF(IDIST.EQ.1)  WRITE(11,440) IPROJ
  440 FORMAT(/I6,' T-EIGENVALUES WERE RELABELLED AS GOOD'/
     1' BELOW IS CORRECTED T-MULTIPLICITY PATTERN'/)
C
      WRITE(6,450) NDIS, (MP(I), I=1,NDIS)
      IF(IDIST.EQ.1) WRITE(11,450) NDIS, (MP(I), I=1,NDIS)
  450 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/
     1 6X, ' (-10) MEANS SPURIOUS T-EIGENVALUE RELABELLED AS GOOD'/(20I4
```

```
     1))
C
C     RECALCULATE MINGAPS FOR DISTINCT T(1,MEV) EIGENVALUES.
  460 NDIS1 = NDIS - 1
      G(NDIS) = VS(NDIS1)-VS(NDIS)
      G(1) = VS(2)-VS(1)
C
      DO 470 J = 2,NDIS1
      T0 = VS(J)-VS(J-1)
      T1 = VS(J+1)-VS(J)
      G(J) = T1
      IF (T0.LT.T1) G(J) = -T0
  470 CONTINUE
      IF(IPROJ.EQ.0) GO TO 500
C     WRITE TO FILE 4 ERROR ESTIMATES FOR THOSE T-EIGENVALUES RELABELLED
      NGOOD = 0
      DO 480 J = 1,NDIS
      IF(MP(J).EQ.0) GO TO 480
      NGOOD = NGOOD + 1
      IF(MP(J).NE.ITEN) GO TO 480
      T0 = VS(J)
      NISO = NISO + 1
      NISOM = MEV + NISO
      WRITE(4,490) NGOOD,T0,G(NISOM),G(J)
  480 CONTINUE
  490 FORMAT(I10,E25.16,2E14.3)
C
  500 CONTINUE
C
C     WRITE THE COMPUTED SINGULAR VALUES TO FILE 3.  FIRST TRANSFER THEM
C     TO V2 AND THEIR T-MULTIPLICITIES TO THE CORRESPONDING POSITIONS
C     IN MP AND COMPUTE THE B-MINGAPS, THE MINIMAL GAPS BETWEEN THE
C     SINGULAR VALUES CONSIDERED AS EIGENVALUES OF THE B-MATRIX.
C     THESE GAPS WILL BE PUT IN THE ARRAY G.
C     SINCE G CURRENTLY CONTAINS THE MINIMAL GAPS BETWEEN THE DISTINCT
C     EIGENVALUES OF THE T-MATRIX, THESE GAPS WILL FIRST BE
C     TRANSFERRED TO V1.  NOTE THAT V1<0 MEANS THAT THAT MINIMAL GAP
C     IN THE T-MATRIX IS DUE TO A SPURIOUS T-EIGENVALUE.
C     ALL THIS INFORMATION IS PRINTED TO FILE 3
C
  510 CONTINUE
C
      NG = 0
      DO 520 I = 1,NDIS
      IF (MP(I).EQ.0) GO TO 520
      NG = NG+1
      MP(NG) = MP(I)
      V2(NG) = VS(I)
      TEMP = G(I)
      TEMP = DABS(TEMP)
      J = I+1
      IF (G(I).LT.ZERO) J = I-1
      IF (MP(J).EQ.0) TEMP = -TEMP
      V1(NG) = TEMP
  520 CONTINUE
```

```
C
      WRITE(6,530)MEV
  530 FORMAT(//' SINGULAR VALUE CALCULATION AT MEV = ',I6,'    IS COMPLE
     1TE'//)
C
C     NG = NUMBER OF COMPUTED DISTINCT GOOD T-EIGENVALUES.  NEXT
C     GENERATE GAPS BETWEEN GOOD T-EIGENVALUES (BMINGAPS) AND PUT THEM
C     IN G.  G(J) < 0 MEANS THE BMINGAP IS DUE TO THE LEFT-HAND GAP.
C
      NGM1 = NG - 1
      G(NG) = V2(NGM1)-V2(NG)
      G(1) = V2(2)-V2(1)
C
      DO 540 J = 2,NGM1
      T0 = V2(J)-V2(J-1)
      T1 = V2(J+1)-V2(J)
      G(J) = T1
      IF (T0.LT.T1) G(J) = -T0
  540 CONTINUE
C
C     WRITE GOOD T-EIGENVALUES (COMPUTED SINGULAR VALUES) OUT TO FILE 3.
C
      WRITE(3,550)NG,NDIS,MEV,M,N,SVSEED,MATNO,IPARO,MULTOL,IB,BTOL
  550 FORMAT(5I6,I12,I8,I2,'=NG,ND,MEV,M,N,SEED,MN,IPARO'/
     1 E20.12,I6,E13.4,' = MUTOL,INDEX MINIMAL BETA,BTOL'/
     1' SV NO',2X,'T-MULT',10X,'SINGULAR VALUE',7X,'BMINGAP',7X,'TMINGAP
     1')
C
      WRITE(3,560)(I,MP(I),V2(I),G(I),V1(I),  I=1,NG)
  560 FORMAT(I6,I8,E25.16,2E14.3)
C
C     IF CONVERGENCE FLAG ICONV.NE.1 AND NUMBER OF T-MATRICES
C     CONSIDERED TO DATE IS LESS THAN NUMBER ALLOWED, INCREMENT MEV.
C     AND LOOP BACK TO 210 TO REPEAT COMPUTATIONS.  RESTORE BETA(MEV+1).
C
      BETA(MP1) = BETAM
C
      IF (MMB.LT.NMEVS.AND.ICONV.NE.1) GO TO 210
C
C     END OF LOOP ON DIFFERENT SIZE T-MATRICES ALLOWED.
C
  570 CONTINUE
C
      IF(ISTOP.EQ.0)  WRITE(6,580)
  580 FORMAT(/' T-MATRICES (BETA) ARE NOW AVAILABLE, TERMINATE'/)
      IF (IHIS.EQ.1.AND.KMAX.NE.MOLD) WRITE(1,590)
  590 FORMAT(/' ABOVE ARE THE FOLLOWING VECTORS '/
     2 '  BETA(I), I = 1,KMAX+1'/
     3 ' FINAL TWO LANCZOS VECTORS OF ORDERS M,N FOR I = KMAX,KMAX+1'/
     4 ' ALL VECTORS IN THIS FILE HAVE FORMAT 4Z20'/
     5 ' ----- END OF FILE 1 NEW BETA HISTORY---------------'///)
C
      IF (ISTOP.EQ.0) GO TO 690
C
      WRITE(3,600)
```

```
  600 FORMAT(/' ABOVE ARE COMPUTED SINGULAR VALUES'/                    LSV07050
     1 ' NG = NUMBER OF SINGULAR VALUES COMPUTED'/                      LSV07060
     2 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)'/   LSV07070
     3 ' M = ROW ORDER OF A N = COLUMN ORDER,  MATNO = MATRIX IDENT'/   LSV07080
     4 ' MULTOL = T-MULTIPLICITY TOLERANCE FOR T-EIGENVALUES IN BISEC'/ LSV07090
     4 ' T-MULT IS THE T-MULTIPLICITY OF SINGULAR VALUE'/               LSV07100
     5 ' T-MULT = -1 MEANS SPURIOUS T-EIGENVALUE TOO CLOSE'/            LSV07110
     6 ' DO NOT COMPUTE ERROR ESTIMATES FOR SUCH T-EIGENVALUES'/        LSV07120
     7 ' BMINGAP = MINIMAL GAP BETWEEN THE COMPUTED SINGULAR VALUES'/   LSV07130
     8 ' BMINGAP .LT. 0. MEANS MINIMAL GAP IS DUE TO LEFT-HAND GAP'/    LSV07140
     9 ' TMINGAP= MINIMAL GAP W.R.T.  DISTINCT EIGENVALUES IN T(1,MEV)'/LSV07150
     1 ' TMINGAP .LT. 0. MEANS MINGAP IS DUE TO SPURIOUS T-EIGENVALUE'/ LSV07160
     2 ' ----- END OF FILE 3 SINGULAR VALUES------------------------'//)LSV07170
C                                                                       LSV07180
      IF (IDIST.EQ.1) WRITE(11,610)                                     LSV07190
  610 FORMAT(/' ABOVE ARE THE DISTINCT EIGENVALUES OF T(1,MEV).'/       LSV07200
     2 ' THE FORMAT IS      T-MULTIPLICITY   T-EIGENVALUE   TMINGAP'/   LSV07210
     3 '        THIS FORMAT IS REPEATED TWICE ON EACH LINE.'/           LSV07220
     4 ' T-MULTIPLICITY = -1 MEANS THAT THE SUBROUTINE ISOEV HAS TAGGED'LSV07230
     5 /'   THIS COMPUTED SINGULAR VALUE AS HAVING A VERY CLOSE SPURIOUSLSV07240
     6 '/'   T-EIGENVALUE SO THAT NO ERROR ESTIMATE WILL BE COMPUTED'/  LSV07250
     7 '   FOR THAT SINGULAR VALUE IN SUBROUTINE INVERR.'/              LSV07260
     8 ' TMINGAP .LT. 0, TMINGAP IS DUE TO LEFT GAP .GT. 0, RIGHT GAP.'/LSV07270
     9 ' EACH OF THE DISTINCT T-EIGENVALUE TABLES IS FOLLOWED'/         LSV07280
     9 ' BY THE T-MULTIPLICITY PATTERN.'/                               LSV07290
     1 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV).'/  LSV07300
     2 ' NG = NUMBER OF COMPUTED SINGULAR VALUES. '/                    LSV07310
     3 ' NISO = NUMBER OF ISOLATED (IN T-MATRIX) SINGULAR VALUES. '/    LSV07320
     4 ' NISO ALSO IS THE COUNT OF +1 ENTRIES IN T-MULTIPLICITY PATTERN.LSV07330
     5 '/' ----- END OF FILE 11 DISTINCT T-EIGENVALUES--------------'//)LSV07340
C                                                                       LSV07350
      IF(NISO.NE.0)  WRITE(4,620)                                       LSV07360
  620 FORMAT(/' ABOVE ARE THE ERROR ESTIMATES OBTAINED FOR THE ISOLATED LSV07370
     1GOOD T-EIGENVALUES'/                                              LSV07380
     1' OBTAINED VIA INVERSE ITERATION IN THE SUBROUTINE INVERR.'/      LSV07390
     1' ALL OTHER GOOD T-EIGENVALUES HAVE CONVERGED.'/                  LSV07400
     2' ERROR ESTIMATE = BETAM*ABS(UM)'/                                LSV07410
     2' WHERE BETAM = BETA(MEV+1) AND UM = U(MEV).'/                    LSV07420
     3' U = UNIT EIGENVECTOR OF T WHERE T*U = SV*U AND SV = ISOLATED GOOLSV07430
     3D T-EIGENVALUE.'/                                                 LSV07440
     4' TMINGAP = GAP TO NEAREST DISTINCT EIGENVALUE OF T(1,MEV).'/     LSV07450
     5' TMINGAP .LT. 0. MEANS MINGAP IS DUE TO A SPURIOUS T-EIGENVALUE.'LSV07460
     6/' ------ END OF FILE 4 ERRINV -------------------------------'//)LSV07470
      GO TO 690                                                         LSV07480
C                                                                       LSV07490
  630 CONTINUE                                                          LSV07500
C                                                                       LSV07510
      IBB = IABS(IBMEV)                                                 LSV07520
      IF (IBMEV.LT.0) WRITE(6,640) MEV,IBB,BETA(IBB)                    LSV07530
  640 FORMAT(/' PROGRAM TERMINATES BECAUSE MEV REQUESTED = ',I6,' IS .GTLSV07540
     1',I6/' AT WHICH AN ABNORMALLY SMALL BETA = ' , E13.4,' OCCURRED'/)LSV07550
      GO TO 690                                                         LSV07560
C                                                                       LSV07570
  650 IF (NDIS.EQ.0.AND.ISTOP.GT.0) WRITE(6,660)                        LSV07580
  660 FORMAT(/' INTERVALS SPECIFIED FOR BISECT DID NOT CONTAIN ANY T-EIGLSV07590
```

```
     1ENVALUES'/' PROGRAM TERMINATES')
      GO TO 690
C
  670 WRITE(6,680) I, NMEV(I)
  680 FORMAT(//I6,'TH T-SIZE REQUESTED ',I6,' IS ODD'/
     1' BUT ONLY EVEN T-SIZES ARE PERMISSIBLE.  PROGRAM TERMINATES FOR U
     1SER TO FIX'//)
      GO TO 690
C
  690 CONTINUE
C
      STOP
C-----END OF MAIN PROGRAM FOR LANCZOS SINGULAR VALUE COMPUTATIONS-------
      END
```

6.4 LSVEC: Main Program, Eigenvector Computations

```
C-----LSVEC (SINGULAR VECTORS OF REAL RECTANGULAR MATRICES)-------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C         REFERENCE: Cullum and Willoughby, Chapter 5
C         Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C         Applied Mathematics, 2002. SIAM Publications,
C         Philadelphia, PA. USA
C
C
C
C     CONTAINS MAIN PROGRAM FOR COMPUTING A LEFT AND A
C     RIGHT SINGULAR VECTOR CORRESPONDING TO EACH OF A SET
C     OF SINGULAR VALUES WHICH HAVE BEEN COMPUTED ACCURATELY BY THE
C     CORRESPONDING LANCZOS SINGULAR VALUE PROGRAM (LSVAL)
C     FOR REAL RECTANGULAR MATRICES.  THIS PROGRAM COULD BE
C     MODIFIED TO COMPUTE ADDITIONAL SINGULAR VECTORS FOR ANY
C     SINGULAR VALUE THAT IS A MULTIPLE SINGULAR VALUE OF A.
C     THE AMOUNT OF ADDITIONAL COMPUTATION REQUIRED BY SUCH A
C     MODIFICATION DEPENDS UPON THE GIVEN A-MATRIX AND UPON
C     THE PART OF THE SPECTRUM INVOLVED.
C
C     FOR A GIVEN REAL MATRIX A OF ORDER M X N THE LANCZOS RECURSION
C     IS APPLIED TO THE ASSOCIATED REAL SYMMETRIC MATRIX B OF ORDER
C     MN = M+N
C
C                                 ----                ----
C                                |  O                   A  |
C                     B  =       |                         |
C                                |  A-TRANSPOSE         O  |
C                                 ----                ----
C     USING SPECIAL STARTING VECTORS.
C
C     THESE SINGULAR VECTOR COMPUTATIONS ASSUME THAT EACH
C     SINGULAR VALUE THAT IS BEING CONSIDERED HAS CONVERGED AS
```

```
C     AN EIGENVALUE OF THE LANCZOS TRIDIAGONAL MATRICES GENERATED.
C
C     THE EIGENVALUES OF EACH EVEN-ORDERED LANCZOS MATRIX OCCUR
C     IN + AND - PAIRS, AND THE RITZ VECTOR COMPUTATION RESTS ON
C     AN INVERSE ITERATION COMPUTATION FOR A LANCZOS MATRIX.
C     THIS CAUSES AN ANOMALY IN THE SINGULAR VECTOR COMPUTATIONS
C     FOR VERY SMALL SINGULAR VALUES.  IN PRACTICE WE SEE THAT
C     FOR ANY SUCH SINGULAR VALUE THAT ONE MEMBER OF EACH PAIR OF
C     APPROXIMATE SINGULAR VECTORS WILL BE MORE ACCURATE THAN THE
C     OTHER MEMBER OF THAT PAIR IS.  IF IPAR = 1 (STARTING LANCZOS
C     VECTOR IS OF FORM (0,V2) WHERE V2 IS NX1) THEN THE RIGHT
C     SINGULAR VECTOR WILL BE OBTAINED MORE ACCURATELY THAN THE
C     LEFT SINGULAR VECTOR.  IF IPAR = 2 (STARTING LANCZOS VECTOR
C     IS OF FORM (V1,0) WHERE V1 IS MX1) THEN THE LEFT SINGULAR
C     VECTOR WILL BE MORE ACCURATE THAN THE RIGHT SINGULAR VECTOR.
C     PRIOR TO NORMALIZATION THE SIZES OF THESE INACCURATE VECTORS
C     WILL BE THE SAME AS THE SIZE OF THE ASSOCIATED VERY SMALL
C     SINGULAR VALUE.  IN FACT IN THE LIMIT, FOR A ZERO SINGULAR VALUE
C     AND IPAR = 1, THE VECTOR COMPUTED AS THE APPROXIMATION TO THE
C     LEFT SINGULAR VECTOR WILL BE THE 0 VECTOR.  (IF IPAR = 2 THEN
C     THIS WOULD BE THE RIGHT SINGULAR VECTOR).  THE CORRESPONDING
C     ERROR ESTIMATES WILL REFLECT THE INACCURACY OF THE ONE MEMBER
C     OF EACH SUCH PAIR, SINCE THESE ESTIMATES ARE A SUM OF ESTIMATES
C     FOR THE INDIVIDUAL MEMBERS OF THE PAIR.  THEREFORE, FOR ANY VERY
C     SMALL SINGULAR VALUE A CORRESPONDING SINGULAR VECTOR WILL BE
C     COMPUTED ONLY IF THE USER HAS SET THE FLAG ERCONT TO 1.
C
C-----------------------------------------------------------------------
C
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE
C     CONSTRUCTIONS
C
C     1.  DATA/MACHEP/ STATEMENT
C     2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)
C     3.  FORMAT(20A4) USED WITH THE EXPLANATORY HEADER, EXPLAN
C     4.  HEXADECIMAL FORMAT (4Z20) USED FOR BETA HISTORY.
C
C     IMPORTANT NOTE:  THIS PROGRAM ALLOWS ENLARGEMENT OF THE
C     BETA ARRAY.  IN PARTICULAR, IF ANY ONE OF THE SINGULAR VALUES
C     SUPPLIED IS T-SIMPLE AND AS AN EIGENVALUE OF THE ASSOCIATED
C     LANCZOS TRIDIAGONAL MATRIX IS NOT CLOSE TO A SPURIOUS
C     EIGENVALUE OF THAT MATRIX, THIS PROGRAM WILL REQUIRE
C     THAT KMAX BE AT LEAST THE LARGEST EVEN NUMBER LESS
C     THAN OR EQUAL TO  (11*MEV)/8 + 13.  IF KMAX IS NOT THAT
C     LARGE, THEN THIS PROGRAM WILL RESET KMAX TO THIS SIZE
C     AND EXTEND THE BETA HISTORY IF REQUIRED.
C     THUS, THE DIMENSION OF THE BETA ARRAY MUST BE
C     LARGE ENOUGH TO ALLOW FOR THIS POSSIBILITY.
C     REMEMBER THAT THE BETA ARRAY, BETA(J), IS SUCH THAT
C     J = 1,..., KMAX+1.  SO IF THE KMAX USED BY THE PROGRAM
C     IS TO BE 3000, THEN BETA MUST BE OF LENGTH AT LEAST 3001.
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  BETA(5001),V1(5000),V2(5000),RITVEC(30000)
      DOUBLE PRECISION  TVEC(30000),GOODSV(50),SVNEW(50),TLAST(50)
```

```
      DOUBLE PRECISION  SVAL,SVALN,TOLN,TTOL,ERTOL,BATA
      DOUBLE PRECISION  MULTOL,SCALE0,STUTOL,BTOL,LB,UB
      DOUBLE PRECISION  ONE,ZERO,MACHEP,EPSM,TEMP,SUM
      DOUBLE PRECISION  RELTOL,ERROR,TERROR,ERRMIN,BKMIN
      REAL     G(10000),BMINGP(50),TMINGP(50),EXPLAN(20)
      REAL     TERR(50),BERR(50),BERRGP(50),RNORM(50),TBETA(50)
      INTEGER  MP(50),M1(50),M2(50),MA(50),ML(50),MINT(50),MFIN(50)
      INTEGER  SVSEED,SVSOLD,RHSEED,IDELTA(50)
      INTEGER  MBOUND,NTVCON,SVTVEC,TVSTOP,LVCONT,ERCONT,TFLAG
      DOUBLE PRECISION  FINPRO
      DOUBLE PRECISION  DABS, DMAX1, DSQRT, DFLOAT
      REAL ABS
      INTEGER  IABS
C-----------------------------------------------------------------------
      EXTERNAL SVMAT, STRAN
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.D0*MACHEP
C-----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     1.  BETA: >= (KMAX+1) WHERE KMAX, THE LARGEST SIZE
C               T-MATRIX CONSIDERED BY THE PROGRAM, IS THE
C               LARGER OF THE SIZE OF THE BETA HISTORY PROVIDED
C               ON FILE 2 (IF ANY ) AND THE SIZE WHICH THE PROGRAM
C               SPECIFIES INTERNALLY, THIS LATTER IS ALWAYS
C               < = (11*MEV)/8 + 13, WHERE MEV IS THE SIZE
C               T-MATRIX THAT WAS USED IN THE CORRESPONDING
C               SINGULAR VALUE COMPUTATIONS.  NOTE THAT ALL
C               T-MATRICES CONSIDERED MUST HAVE EVEN ORDER.
C     2.  V1:  >= MAX(M,KMAX)
C     3.  V2:   >= N
C     4.  G:  >= MAX(M,N,KMAX)
C     5.  RITVEC:  >= (N+M)*NGOOD, WHERE NGOOD IS THE NUMBER OF
C                 SINGULAR VALUES SUPPLIED TO THIS PROGRAM.
C     6.  TVEC:  >= CUMULATIVE LENGTH OF ALL THE T-EIGENVECTORS
C               NEEDED TO GENERATE THE DESIRED RITZ VECTORS.  AN
C               EDUCATED GUESS AT AN APPROPRIATE LENGTH CAN BE
C               OBTAINED BY RUNNING THE PROGRAM WITH THE FLAG
C               MBOUND = 1 AND MULTIPLYING THE RESULTING SIZE BY 5/4.
C     7.  GOODSV, TMINGP, BMINGP,TERR, BERR, BERRGP, RNORM,
C         TBETA, TLAST, SVNEW, MP, MA, M1, M2, MINT, MFIN AND
C         IDELTA MUST ALL BE >= NGOOD.
C
C-----------------------------------------------------------------------
C     OUTPUT HEADER
      WRITE(6,10)
   10 FORMAT(/' LANCZOS PROCEDURE FOR REAL, RECTANGULAR MATRICES'/
     1'      COMPUTE SINGULAR VECTORS'/)
C
C     SET PROGRAM PARAMETERS
C     USER MUST NOT MODIFY SCALE0
      SCALE0 = 5.0D0
      ZERO = 0.0D0
      ONE = 1.0D0
      MPMIN = -1000
```

```
C     CONVERGENCE TOLERANCE FOR T-EIGENVECTORS FOR RITZ COMPUTATIONS
      ERTOL = 1.D-10
C
C     READ USER-SPECIFIED PARAMETER FROM INPUT FILE 5 (FREE FORMAT)
C
C     READ USER-PROVIDED HEADER FOR RUN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C     READ IN MATNO = MATRIX/RUN IDENTIFICATION NUMBER, 8 DIGITS OR LESS
C     AND THE ORDER OF THE MATRIX M X N .
C
      READ(5,20) EXPLAN
      READ(5,*) MATNO, M, N
      MN = M + N
C
C     READ IN THE MAXIMUM PERMISSIBLE DIMENSIONS FOR THE TVEC ARRAY
C     (MDIMTV), FOR THE RITVEC ARRAY (MDIMRV), AND FOR THE BETA
C     ARRAY (MBETA).
C
      READ(5,20) EXPLAN
      READ(5,*) MDIMTV, MDIMRV, MBETA
C
C     READ IN RELATIVE TOLERANCE USED IN DETERMINING APPROPRIATE
C     SIZES FOR THE T-MATRICES USED IN THE SINGULAR VECTOR COMPUTATIONS.
C
      READ(5,20) EXPLAN
      READ(5,*) RELTOL
C
C     SET FLAGS TO 0 OR 1:
C     MBOUND = 1:  PROGRAM TERMINATES AFTER COMPUTING 1ST GUESSES
C                  ON APPROPRIATE T-SIZES FOR USE IN THE RITZ VECTOR
C                  COMPUTATIONS
C     NTVCON = 0:  PROGRAM TERMINATES IF THE TVEC ARRAY IS NOT
C                  LARGE ENOUGH TO HOLD ALL THE T-EIGENVECTORS REQUIRED.
C     SVTVEC = 0:  THE T-EIGENVECTORS ARE NOT WRITTEN TO FILE 11
C                  UNLESS TVSTOP = 1
C     SVTVEC = 1:  WRITE THE T-EIGENVECTORS TO FILE 11.
C     TVSTOP = 1:  PROGRAM TERMINATES AFTER COMPUTING THE
C                  T-EIGENVECTORS
C     LVCONT = 0:  PROGRAM TERMINATES IF THE NUMBER OF T-EIGENVECTORS
C                  COMPUTED IS NOT EQUAL TO THE NUMBER OF RITZ
C                  VECTORS (SINGULAR VECTORS) REQUESTED.
C     ERCONT = 0:  MEANS FOR ANY GIVEN SINGULAR VALUE, A RITZ VECTOR
C                  WILL NOT BE COMPUTED FOR THAT SINGULAR VALUE UNLESS
C                  A T-EIGENVECTOR HAS BEEN IDENTIFIED WITH A LAST
C                  COMPONENT WHICH SATISFIES THE SPECIFIED
C                  CONVERGENCE CRITERION.
C     ERCONT = 1:  MEANS FOR ANY GIVEN SINGULAR VALUE, A RITZ VECTOR
C                  WILL BE COMPUTED.  IF A T-EIGENVECTOR CANNOT
C                  BE IDENTIFIED WHICH SATISFIES THE LAST
C                  COMPONENT CRITERION, THEN THE PROGRAM WILL
C                  USE THE T-VECTOR THAT CAME CLOSEST TO
C                  SATISFYING THE CRITERION
```

```
C     IWRITE = 1:  EXTENDED OUTPUT OF INTERMEDIATE COMPUTATIONS         LSV02100
C                  IS WRITTEN TO FILE 6                                 LSV02110
C     IREAD = 0:   BETA FILE IS REGENERATED.                            LSV02120
C     IREAD = 1:   BETA FILE USED IN SINGULAR VALUE COMPUTATIONS        LSV02130
C                  IS READ IN AND EXTENDED IF NECESSARY.  IN BOTH       LSV02140
C                  CASES IREAD = 0 OR 1, THE LANCZOS VECTORS ARE        LSV02150
C                  ALWAYS REGENERATED FOR THE RITZ VECTOR               LSV02160
C                  COMPUTATIONS                                         LSV02170
C                                                                       LSV02180
      READ(5,20) EXPLAN                                                 LSV02190
      READ(5,*) MBOUND,NTVCON,SVTVEC,IREAD                              LSV02200
C                                                                       LSV02210
      READ(5,20) EXPLAN                                                 LSV02220
      READ(5,*) TVSTOP,LVCONT,ERCONT,IWRITE                             LSV02230
      IF (TVSTOP.EQ.1) SVTVEC = 1                                       LSV02240
C                                                                       LSV02250
C     READ IN SEED (RHSEED) FOR GENERATING RANDOM STARTING VECTOR       LSV02260
C     FOR THE INVERSE ITERATION ON THE T-MATRICES.                      LSV02270
C                                                                       LSV02280
      READ(5,20) EXPLAN                                                 LSV02290
      READ(5,*) RHSEED                                                  LSV02300
C                                                                       LSV02310
C-----------------------------------------------------------------------LSV02320
C     INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED MATRIX AND           LSV02330
C     PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE MATRIX-VECTOR   LSV02340
C     MULTIPLY SUBROUTINES SVMAT AND STRAN.                             LSV02350
C                                                                       LSV02360
      CALL USPEC(M,N,MATNO)                                             LSV02370
C                                                                       LSV02380
C-----------------------------------------------------------------------LSV02390
C     MASK UNDERFLOW AND OVERFLOW                                       LSV02400
      CALL MASK                                                         LSV02410
C                                                                       LSV02420
C-----------------------------------------------------------------------LSV02430
C     WRITE RUN PARAMETERS OUT TO FILE 6                                LSV02440
C                                                                       LSV02450
      WRITE(6,30) M,N,MATNO                                             LSV02460
   30 FORMAT(/' MATRIX ORDER =',I5,' BY ',I5/                           LSV02470
     1 ' A-MATRIX AND CASE IDENTIFIER = ',I10/)                         LSV02480
C                                                                       LSV02490
      WRITE(6,40) MBOUND,NTVCON,SVTVEC,IREAD                            LSV02500
   40 FORMAT(/3X,'MBOUND',3X,'NTVCON',3X,'SVTVEC',3X,'IREAD'/3I9,I8/)   LSV02510
C                                                                       LSV02520
      WRITE(6,50) TVSTOP,LVCONT,ERCONT,IWRITE                           LSV02530
   50 FORMAT(/3X,'TVSTOP',3X,'LVCONT',3X,'ERCONT',3X,'IWRITE'/4I9)      LSV02540
C                                                                       LSV02550
      WRITE(6,60) MDIMTV,MDIMRV,MBETA                                   LSV02560
   60 FORMAT(/3X,'MDIMTV',3X,'MDIMRV',3X,'MBETA'/2I9,I8)                LSV02570
C                                                                       LSV02580
      WRITE(6,70) RELTOL,RHSEED                                         LSV02590
   70 FORMAT(/7X,'RELTOL',3X,'RHSEED'/E13.4,I9)                         LSV02600
C                                                                       LSV02610
C     FROM FILE 3 READ IN THE NUMBER OF SINGULAR VALUES (NGOOD)         LSV02620
C     FOR WHICH SINGULAR VECTORS ARE REQUESTED, THE ORDER (MEV) OF      LSV02630
C     THE LANCZOS TRIDIAGONAL MATRIX USED IN COMPUTING THESE            LSV02640
```

```
C     SINGULAR VALUES, THE ORDER MOLD X NOLD OF THE USER-SPECIFIED
C     MATRIX USED IN THOSE COMPUTATIONS, THE SEED (SVSEED) USED FOR
C     GENERATING THE STARTING VECTOR THAT WAS USED IN THOSE
C     COMPUTATIONS, AND THE MATRIX/RUN IDENTIFICATION NUMBER (MATOLD)
C     USED IN THOSE COMPUTATIONS.  ALSO READ IN THE NUMBER (NDIS) OF
C     DISTINCT EIGENVALUES OF THE MATRIX T(1,MEV) THAT WERE COMPUTED
C     BUT THIS VALUE IS NOT USED IN THE SINGULAR VECTOR
C     COMPUTATIONS.
C
      READ(3,80) NGOOD,NDIS,MEV,MOLD,NOLD,SVSEED,MATOLD,IPARO
   80 FORMAT(5I6,I12,I8,I2)
C
C     READ IN THE T-MULTIPLICITY TOLERANCE USED IN THE BISEC SUBROUTINE
C     DURING THE COMPUTATION OF THE GIVEN SINGULAR VALUES.
C     ALSO READ IN THE FLAG IB.  IF IB < 0, THEN SOME BETA(I) IN THE
C     T-MATRIX FILE PROVIDED ON FILE 2 FAILED THE ORTHOGONALITY
C     TEST IN THE TNORM SUBROUTINE.  USER SHOULD NOTE THAT THIS
C     PROGRAM PROCEEDS INDEPENDENTLY OF THE SIZE OF THE BETA USED.
C
      READ(3,90) MULTOL,IB,BTOL
   90 FORMAT(E20.12,I6,E13.4)
C
      TEMP = DFLOAT(MEV+1000)
      TTOL = MULTOL/TEMP
      WRITE(6,100) MULTOL,TTOL
  100 FORMAT(/' T-MULTIPLICITY TOLERANCE USED IN THE SINGULAR VALUE COMP
     1UTATIONS WAS',E13.4/' SCALED MACHINE EPSILON IS',E13.4)
C
C      CONTINUE WRITE TO FILE 6 OF THE PARAMETERS FOR THIS RUN
C
      WRITE(6,110)NGOOD,NDIS,MEV,MOLD,NOLD,MATOLD,SVSEED,MULTOL,IB,
     1BTOL,IPARO
  110 FORMAT(/' SINGULAR VALUES SUPPLIED ARE READ IN FROM FILE 3'/
     1 6X,'NG',2X,'NDIS',3X,'MEV',2X,'MOLD',2X,'NOLD',2X,'MATOLD',4X/
     1I8,4I6,I8//6X,'SVSEED',6X,'MULTOL',9X,'IB',8X,'BTOL',4X,'IPARO'/
     1I12,E12.3,I11,E12.4,I9/)
C
C      IS THE ARRAY RITVEC LONG ENOUGH TO HOLD ALL OF THE DESIRED
C     RITZ VECTORS (APPROXIMATE EIGENVECTORS OF B)?
      MNMAX = NGOOD*MN
      IF(MBOUND.EQ.1) GO TO 120
      IF(TVSTOP.NE.1.AND.MNMAX.GT.MDIMRV) GO TO 1600
C
C     CHECK THAT THE ORDERS M,N AND THE MATRIX IDENTIFICATION NUMBER
C     MATNO SPECIFIED BY THE USER AGREE WITH THOSE READ IN FROM
C     FILE 3.
  120 ITEMP = (MOLD-M)**2+(NOLD-N)**2+(MATOLD-MATNO)**2
      IF (ITEMP.NE.0) GO TO 1620
C
C     READ IN FROM FILE 3, THE T-MULTIPLICITIES OF THE SINGULAR VALUES
C     WHOSE SINGULAR VECTORS ARE TO BE COMPUTED, THE VALUES OF THESE
C     SINGULAR VALUES AND THEIR MINIMAL GAPS AS SINGULAR VALUES OF THE
C     USER-SPECIFIED MATRIX AND OF THE RELATED T-MATRIX.
C
      READ(3,20) EXPLAN
```

```
      READ(3,130) (MP(J),GOODSV(J),BMINGP(J),TMINGP(J), J=1,NGOOD)
  130 FORMAT(6X,I8,E25.16,2E14.3)
C
      WRITE(6,140) (J,GOODSV(J),MP(J),BMINGP(J),  J=1,NGOOD)
  140 FORMAT(/' SINGULAR VALUES READ IN FROM FILE 3 AND THEIR T-MULTIPLI
     1CITIES'/4X,' J ',4X,' SINGULAR VALUE',5X,'TMULT',4X,'BMINGP'/
     1(I6,E20.12,I6,E13.4))
C
      WRITE(6,150) MEV,SVSEED
  150 FORMAT(/' THESE SINGULAR VALUES WERE COMPUTED USING A T-MATRIX OF
     1ORDER ',I5/' AND SEED FOR RANDOM NUMBER GENERATOR =',I12)
C
C     READ IN THE ERROR ESTIMATES
C
C     CHECK WHETHER OR NOT THERE ARE ANY ISOLATED T-EIGENVALUES IN
C     THE T-EIGENVALUES PROVIDED (HERE THE SINGULAR VALUES ARE
C     CONSIDERED AS EIGENVALUES OF THE ASSOCIATED LANCZOS TRIDIAGONAL
C     MATRICES.)
      DO 160 J=1,NGOOD
      IF(MP(J).EQ.1) GO TO 170
  160 CONTINUE
      GO TO 190
  170 READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,180) NISO
  180 FORMAT(18X,I6)
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
  190 DO 220 J=1,NGOOD
      BERR(J) = 0.D0
      IF(MP(J).NE.1) GO TO 220
      READ(4,200) SVAL, BERR(J)
  200 FORMAT(10X,E25.16,E14.3)
      IF(DABS(SVAL - GOODSV(J)).LT.1.D-10) GO TO 220
      WRITE(6,210) SVAL,GOODSV(J)
  210 FORMAT(' PROBLEM WITH READ IN OF ERROR ESTIMATES'/' SINGULAR VALUE
     1READ IN',E20.12,' DOES NOT MATCH GOODSV(J) ='/E20.12)
      GO TO 1860
C
  220 CONTINUE
C
      WRITE(6,230) (J,GOODSV(J),BERR(J), J=1,NGOOD)
  230 FORMAT(' ERROR ESTIMATES ='/4X,' J',3X,'SINGULAR VALUE',8X,
     1'ESTIMATE'/(I6,E20.12,E14.3))
C
      IF(IREAD.EQ.0) IPAR = IPARO
      IF(IREAD.EQ.0)  GO TO 350
C
C     READ IN THE SIZE OF THE T-MATRIX PROVIDED ON FILE 2.  READ IN
C     THE ORDER OF THE USER-SPECIFIED MATRIX , THE FLAGS IPARO
C     AND IPAR WHICH INDICATE RESPECTIVELY THE PARITY OF THE
C     STARTING VECTOR USED IN THE GENERATION OF THE EXISTING
C     BETA AND THE PARITY OF THE NEXT LANCZOS VECTOR THAT
```

```
C     HAS TO BE GENERATED IF THE BETA HISTORY IS EXTENDED,
C     THE SEED USED BY THE RANDOM NUMBER GENERATOR WHEN
C     GENERATING THE STARTING VECTOR THAT WAS USED, AND THE
C     MATRIX/TEST IDENTIFICATION NUMBER THAT WERE USED IN
C     THE LANCZOS SINGULAR VALUE COMPUTATIONS.  IF THE FLAG
C     IREAD = 0, REGENERATE HISTORY AND DO NOT READ ANYTHING
C     FROM FILE 2.  HISTORY MUST BE STORED IN MACHINE FORMAT,
C     ((4Z20) FOR IBM 3081).
C
      READ(2,240) KMAX,MOLD,NOLD,IPARO,IPAR,SVSOLD,MATOLD
  240 FORMAT(3I6,2I3,I12,I8)
C
      WRITE(6,250) KMAX,MOLD,NOLD,IPARO,IPAR,SVSOLD,MATOLD
  250 FORMAT(/' READ IN HEADER FROM BETA FILE 2'/
     1 2X,'KMAX',2X,'MOLD',2X,'NOLD',2X,'IPARO',2X,'IPAR',6X,'SVSOLD
     1 ',2X,'MATOLD'/3I6,I7,I6,I12,I12)
C
C     CHECK THAT THE PARAMETERS READ IN AGREE WITH WHAT THE USER
C     HAS SPECIFIED
      IF(MOLD.NE.M.OR.NOLD.NE.N.OR.MATOLD.NE.MATNO.OR.SVSOLD.NE.SVSEED)
     1 GO TO 1640
C
      IF(IPARO.EQ.1) WRITE(6,260)
      IF(IPARO.EQ.2) WRITE(6,270)
  260 FORMAT(/' STARTING VECTOR USED IN EXISTING SINGULAR VALUE HISTORY
     1WAS'/' OF THE FORM (0,V2)')
  270 FORMAT(/' STARTING VECTOR USED IN EXISTING SINGULAR VALUE HISTORY
     1WAS'/' OF THE FORM (V1,0)')
C
      KMAX1 = KMAX + 1
C
C     READ IN THE T-MATRICES FROM FILE 2.  THESE ARE USED TO GENERATE
C     THE T-EIGENVECTORS THAT WILL BE USED IN THE RITZ VECTOR
C     COMPUTATIONS.  HISTORY MUST BE STORED IN 4Z20 FORMAT.
C
      READ(2,280) (BETA(J), J=1,KMAX1)
  280 FORMAT(4Z20)
C
      READ(2,280) (V1(J), J=1,M)
      READ(2,280) (V2(J), J=1,N)
C
C     KMAX MAY BE ENLARGED IF THE SIZE AT WHICH THE SINGULAR VALUE
C     COMPUTATIONS WERE PERFORMED IS ESSENTIALLY KMAX AND
C     THERE IS AT LEAST ONE SINGULAR VALUE THAT IS SIMPLE AS AN
C     EIGENVALUE OF T(1,MEV), AND IF ITS NEAREST NEIGHBOR IN THE
C     T-MATRIX IS TOO CLOSE, THAT NEIGHBOR IS A 'GOOD' T-EIGENVALUE.
      DO 290 J = 1,NGOOD
      IF(MP(J).EQ.1) GO TO 310
  290 CONTINUE
      WRITE(6,300)
  300 FORMAT(/' ALL SINGULAR VALUES USED ARE T-MULTIPLE OR CLOSE TO SPUR
     1IOUS EIGENVALUES'/' (AS EIGENVALUES OF T(1,MEV)) SO KMAX IS NOT CH
     1ANGED'/)
      IF(KMAX.LT.MEV) GO TO 1660
      GO TO 330
```

```
C                                                                       LSV04300
  310 KMAXN= (11*MEV)/8 + 12                                            LSV04310
      IF((KMAXN/2)*2.NE.KMAXN) KMAXN = KMAXN + 1                        LSV04320
      IF(MBETA.LE.KMAXN) GO TO 1840                                     LSV04330
      IF(KMAX.GE.KMAXN )  GO TO 330                                     LSV04340
      WRITE(6,320) KMAX, KMAXN                                          LSV04350
  320 FORMAT(' ENLARGE KMAX FROM ',I6,' TO ',I6)                        LSV04360
      MOLD1 = KMAX + 1                                                  LSV04370
      KMAX = KMAXN                                                      LSV04380
      GO TO 420                                                         LSV04390
C                                                                       LSV04400
  330 WRITE(6,340) KMAX                                                 LSV04410
  340 FORMAT(/' T-MATRICES HAVE BEEN READ IN FROM FILE 2'/' THE LARGEST LSV04420
     1SIZE T-MATRIX ALLOWED IS',I6/)                                    LSV04430
C                                                                       LSV04440
      IF(IREAD.EQ.1) GO TO 460                                          LSV04450
C                                                                       LSV04460
C     REGENERATE THE BETA                                               LSV04470
C                                                                       LSV04480
  350 MOLD1 = 1                                                         LSV04490
C                                                                       LSV04500
      IF(IPAR.EQ.1) WRITE(6,360)                                        LSV04510
      IF(IPAR.EQ.2) WRITE(6,370)                                        LSV04520
  360 FORMAT(/' STARTING VECTOR USED IN HISTORY REGENERATION IS OF THE  LSV04530
     1FORM (0,V2)')                                                     LSV04540
  370 FORMAT(/' STARTING VECTOR USED IN HISTORY REGENERATION IS OF THE  LSV04550
     1FORM (V1,0)')                                                     LSV04560
C                                                                       LSV04570
      DO 380 J = 1,NGOOD                                                LSV04580
      IF(MP(J).EQ.1) GO TO 400                                          LSV04590
  380 CONTINUE                                                          LSV04600
      KMAX = MEV + 12                                                   LSV04610
      IF((KMAX/2)*2.NE.KMAX) GO TO 1680                                 LSV04620
      WRITE(6,390) KMAX                                                 LSV04630
  390 FORMAT(/' ALL SINGULAR VALUES FOR WHICH SINGULAR VECTORS ARE TO BELSV04640
     1COMPUTED ARE EITHER T-MULTIPLE OR CLOSE TO'/' A SPURIOUS T-EIGENVALSV04650
     1LUE THEREFORE SET KMAX = MEV + 12 = ',I7)                         LSV04660
      GO TO 420                                                         LSV04670
C                                                                       LSV04680
  400 KMAXN = (11*MEV)/8 + 12                                           LSV04690
      IF((KMAXN/2)*2.NE.KMAXN) KMAXN = KMAXN + 1                        LSV04700
      IF(MBETA.LE.KMAXN) GO TO 1840                                     LSV04710
      WRITE(6,410) KMAXN                                                LSV04720
  410 FORMAT(' SET KMAX EQUAL TO ',I6)                                  LSV04730
      KMAX = KMAXN                                                      LSV04740
C                                                                       LSV04750
  420 KMAX1 = KMAX + 1                                                  LSV04760
      WRITE(6,430) MOLD1,KMAX1                                          LSV04770
  430 FORMAT(/' LANCZS SUBROUTINE GENERATES BETA(J+1), J =',            LSV04780
     1 I6,' TO ', I6/)                                                  LSV04790
      IF(IREAD.EQ.1.AND.IPAR.EQ.1) WRITE(6,440)                         LSV04800
      IF(IREAD.EQ.1.AND.IPAR.EQ.2) WRITE(6,450)                         LSV04810
  440 FORMAT(/' FIRST LANCZOS VECTOR IN HISTORY EXTENSION IF OF THE FORMLSV04820
     1 (0,V2)')                                                         LSV04830
  450 FORMAT(/' FIRST LANCZOS VECTOR IN HISTORY EXTENSION IF OF THE FORMLSV04840
```

```
     1 (V1,0)')
C
C-----------------------------------------------------------------------
C
      CALL LANCZS(SVMAT,STRAN,BETA,V1,V2,G,KMAX,MOLD1,M,N,IPAR,SVSEED)
C
C-----------------------------------------------------------------------
C
  460 CONTINUE
C
C     THE SUBROUTINE STURMI DETERMINES THE SMALLEST SIZE T-MATRIX FOR
C     WHICH THE SINGULAR VALUE IN QUESTION IS AN EIGENVALUE (TO
C     WITHIN A SPECIFIED TOLERANCE) AND IF POSSIBLE THE SMALLEST
C     SIZE T-MATRIX FOR WHICH THE SINGULAR VALUE IS A DOUBLE
C     EIGENVALUE (TO WITHIN THE SAME TOLERANCE).  THE SIZE
C     T-MATRIX THAT WILL BE USED IN EACH OF THE RITZ VECTOR COMPUTATIONS
C     IS THEN DETERMINED BY LOOPING ON THE SIZE OF THE T-EIGENVECTOR
C     COMPUTATIONS, STARTING WITH A SIZE DETERMINED FROM THE
C     INFORMATION OBTAINED FROM STURMI.
C
      STUTOL = SCALE0*MULTOL
      IF(IWRITE.EQ.1) WRITE(6,470)
  470 FORMAT(' FROM STURMI')
      DO 510 J = 1,NGOOD
      SVAL = GOODSV(J)
C     COMPUTE THE TOLERANCES USED BY STURMI TO DETERMINE AN INTERVAL
C     CONTAINING THE SINGULAR VALUE SVAL.
      TEMP = DABS(SVAL)*RELTOL
      TOLN = DMAX1(TEMP,STUTOL)
C
C-----------------------------------------------------------------------
C
      CALL STURMI(BETA,SVAL,TOLN,EPSM,KMAX,MK1,MK2,IC,IWRITE)
C
C-----------------------------------------------------------------------
C
C     STORE THE COMPUTED ORDERS OF T-MATRICES FOR LATER PRINTOUT
      IF(MK1.GT.1) GO TO 475
C     SVAL IS VERY SMALL SINGULAR VALUE, RESET MK1 TO CORRECT VALUE
      MK1 = MK2
      MK2 = MIN0(2*MK1,KMAX)
      M1(J) = MK1
      M2(J) = MK2
      ML(J) = MK2
      GO TO 476
  475 M1(J) = MK1
      M2(J) = MK2
      ML(J) = (MK1 + 3*MK2)/4
      IF(MK2.EQ.KMAX)  ML(J) = KMAX
C
  476 IF(IC.GT.0) GO TO 490
C     IC = 0 MEANS THERE WAS NO T-EIGENVALUE IN THE DESIGNATED INTERVAL
C     EVEN BY T-SIZE KMAX.  THIS MEANS THAT THE SINGULAR VALUE
C     PROVIDED HAS NOT YET CONVERGED SO PROGRAM DOES NOT COMPUTE
C     A SINGULAR VECTOR FOR IT.
```

```
      WRITE(6,480) J,GOODSV(J),MK1,MK2
  480 FORMAT(I6,'TH SINGULAR VALUE',E20.12,' HAS NOT CONVERGED '/
     1' SO DO NOT COMPUTE ANY T-EIGENVECTOR OR RITZ VECTOR FOR IT'
     1/' MK1 AND MK2 FOR THIS SINGULAR VALUE WERE',2I6)
      MP(J) = MPMIN
      MA(J) = -2*KMAX
      GO TO 510
C     COMPUTE AN APPROPRIATE SIZE T-MATRIX FOR THE GIVEN SINGULAR
C     VALUE.
  490 IF(M2(J).EQ.KMAX) GO TO 500
C     M1 AND M2 WERE BOTH DETERMINED
      MAJ = (3*M1(J) + M2(J))/4  + 1
      IF((MAJ/2)*2.NE.MAJ)  MAJ = MAJ + 1
      MA(J) = MAJ
      GO TO 510
C     M2 NOT DETERMINED
  500 MAJ = (5*M1(J))/4  + 1
      IF((MAJ/2)*2.NE.MAJ)  MAJ = MAJ + 1
      MA(J) = MAJ
C
  510 CONTINUE
C
      IF (IWRITE.EQ.1) WRITE(6,520) (MA(JJ), JJ=1,NGOOD)
  520 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZE T-MATRICES'/
     1 ' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/(13I6))
C
C     PRINT OUT TO FILE 10 1ST GUESSES AT SIZES OF THE T-MATRICES TO
C     BE USED IN THE SINGULAR VECTOR COMPUTATIONS.
C     PROGRAM LOOPS ON T-SIZE TO DETERMINE APPROPRIATE SIZE T-MATRIX.
      WRITE(10,530) N,KMAX
  530 FORMAT(2I8,' = ORDER OF USER MATRIX AND MAX ORDER OF T(1,MEV)')
C
      WRITE(10,540)
  540 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZE T-MATRICES'/
     1 ' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/)
C
      WRITE(10,550)
  550 FORMAT(4X,'J',7X,'GOODSV(J)',4X,'M1(J)',1X,'M2(J)',1X,'MA(J)')
C
      WRITE(10,560) (J,GOODSV(J),M1(J),M2(J), MA(J), J=1,NGOOD)
  560 FORMAT(I5,E19.12,3I6)
C
      IF(MBOUND.EQ.1) WRITE(10,570)
  570 FORMAT(/' GOODSV(J) IS A GOOD EIGENVALUE OF T(1,MEV)'/
     1 ' M1 = SMALLEST VALUE OF M SUCH THAT T(1,M) HAS AT LEAST'/
     1 '      ONE EIGENVALUE IN THE INTERVAL (SV-TOLN,SV+TOLN)'/
     1 '      TOLN(J) = DMAX1(GOODSV(J)*RELTOL, SCALE0*MULTOL)'/
     1 ' M2 = SMALLEST M (IF ANY) SUCH THAT IN THE ABOVE INTERVAL'/
     1 '      T(1,M) HAS AT LEAST TWO EIGENVALUES '/
     1 ' INITIAL VALUE OF MA(J) IS CHOSEN HEURISTICALLY'/
     1 ' PROGRAM LOOPS ON SIZE OF T-MATRIX TO GET APPROPRIATE SIZE'/
     1 ' END OF SIZES OF T-MATRICES FILE 10'///)
C
C
C     TERMINATE AFTER COMPUTING 1ST GUESSES AT  SIZES OF THE
```

```
C     T-MATRICES REQUIRED FOR THE GIVEN SINGULAR VALUES?
      IF(MBOUND.EQ.1) GO TO 1700
C
C
C     IS THERE ROOM FOR ALL OF THE REQUESTED T-EIGENVECTORS?
      MTOL = 0
      DO 580 J = 1,NGOOD
      IF(MP(J).EQ.MPMIN) GO TO 580
      MTOL = MTOL + IABS(MA(J))
  580 CONTINUE
      MTOL = (5*MTOL)/4
      IF(MTOL.GT.MDIMTV.AND.NTVCON.EQ.0) GO TO 1720
C
C-----------------------------------------------------------------------
C     GENERATE A RANDOM VECTOR TO BE USED REPEATEDLY BY
C     SUBROUTINE INVERM
C
      IIL = RHSEED
      CALL GENRAN(IIL,G,KMAX)
C
C-----------------------------------------------------------------------
C
C     FOR EACH SINGULAR VALUE LOOP ON T-EIGENVECTOR COMPUTATIONS
C     TO COMPUTE AN APPROPRIATE T-EIGENVECTOR TO USE IN THE
C     RITZ VECTOR COMPUTATIONS.
C
      MTOL = 0
      NTVEC = 0
      ILBIS = 0
      DO 770 J = 1,NGOOD
      ICOUNT = 0
      ERRMIN = 10.D0
      MABEST = MPMIN
      IF(MP(J).EQ.MPMIN) GO TO 770
      TFLAG = 0
      SVAL = GOODSV(J)
      TEMP = RELTOL*DABS(SVAL)
      UB = SVAL + DMAX1(STUTOL,TEMP)
      LB = SVAL - DMAX1(STUTOL,TEMP)
      LB = DMAX1(LB,ZERO)
  590 KMAXU = IABS(MA(J))
C
C     SELECT A SUITABLE INCREMENT FOR THE ORDERS OF THE T-MATRICES
C     TO BE CONSIDERED IN DETERMINING APPROPRIATE SIZES FOR THE RITZ
C     VECTOR COMPUTATIONS.  ALL ORDERS CONSIDERED MUST BE EVEN.
      IF(ICOUNT.GT.0) GO TO 610
C     SELECT IDELTA(J) BASED UPON THE T-MULTIPLICITY OBTAINED
      IF(M2(J).EQ.KMAX) GO TO 600
C     M2 DETERMINED
      IDEL = ((3*M1(J) + 5*M2(J))/8  +  1 - IABS(MA(J)))/10  +  1
      IF((IDEL/2)*2.NE.IDEL) IDEL = IDEL + 1
      IDELTA(J) = IDEL
      GO TO 610
C     M2 NOT DETERMINED
  600 MAMAX = MIN0((11*MEV)/8 + 12, (13*M1(J))/8 + 1)
```

```
      IDEL = (MAMAX - IABS(MA(J)))/10  +  1
      IF((IDEL/2)*2.NE.IDEL)  IDEL = IDEL + 1
      IDELTA(J) = IDEL
  610 ICOUNT = ICOUNT + 1
C
C-----------------------------------------------------------------------
C     TO MIMIMIZE THE EFFECT OF THE ONE-SIDED ACCEPTANCE TEST FOR
C     EIGENVALUES IN THE BISEC SUBROUTINE, RECOMPUTE THE GIVEN
C     SINGULAR VALUE AT THE SPECIFIED KMAXU
C
      CALL LBISEC(BETA,EPSM,SVAL,SVALN,LB,UB,TTOL,KMAXU,NEVT)
C
C-----------------------------------------------------------------------
C
C     CHECK WHETHER OR NOT GIVEN T-MATRIX HAS AN EIGENVALUE IN THE
C     SPECIFIED INTERVAL AND IF SO WHAT ITS T-MULTIPLICITY IS.
C
      IF(NEVT.EQ.1) GO TO 650
      IF(NEVT.NE.0)  GO TO 630
      ILBIS = 1
      WRITE(6,620) SVAL,KMAXU
  620 FORMAT(/' PROBLEM ENCOUNTERED IN RECOMPUTATION OF USER-SUPPLIED SI
     1NGULAR VALUE',E20.12/' THE SIZE T-MATRIX SPECIFIED',I6,' DOES NOT
     1HAVE A SINGULAR VALUE IN THE INTERVAL SPECIFIED'/' INCREASE SIZE A
     1ND TRY AGAIN'/)
      GO TO 670
C
  630 IF(NEVT.GT.1)  WRITE(6,640) SVAL,KMAXU
  640 FORMAT(/' PROBLEM ENCOUNTERED IN RECOMPUTATION OF USER-SUPPLIED
     1SINGULAR VALUE',E20.12/' FOR THE SIZE T-MATRIX SPECIFIED =',I6,' T
     1HE GIVEN SINGULAR VALUE IS T-MULTIPLE IN THE INTERVAL SPECIFIED'/'
     1SOMETHING IS WRONG, THEREFORE NO SINGULAR VECTORS WILL BE COMPUTED
     1 FOR THIS SINGULAR VALUE'/)
C
      MP(J) = MPMIN
      MA(J) = -2*KMAX
      GO TO 770
C
  650 CONTINUE
      ILBIS = 0
C
C
      SVNEW(J) = SVALN
      SVAL = SVALN
      MTOL = MTOL+KMAXU
C
C     IS THERE ROOM IN TVEC ARRAY FOR THE NEXT T-EIGENVECTOR?
C     IF NOT, SKIP TO RITZ VECTOR COMPUTATIONS.
      IF (MTOL.GT.MDIMTV) GO TO 780
C
      IT = 3
      KINT = MTOL - KMAXU +1
C
C     RECORD THE BEGINNING AND END OF THE T-EIGENVECTOR BEING COMPUTED
      MINT(J) = KINT
```

```
      MFIN(J) = MTOL
C
C-----------------------------------------------------------------------
C     SUBROUTINE INVERM DOES INVERSE ITERATION, I.E. SOLVES
C     (T(1,KMAXU) - SVAL)*U = RHS  FOR EACH SINGULAR VALUE TO
C     OBTAIN THE DESIRED T-EIGENVECTOR.
C
      IF(IWRITE.EQ.1)  WRITE(6,660) J
  660 FORMAT(/I6,'TH SINGULAR VALUE ')
C
      CALL INVERM(BETA,V1,TVEC(KINT),SVAL,ERROR,TERROR,EPSM,G,KMAXU,
     1 IT,IWRITE)
C
C-----------------------------------------------------------------------
C
      TERR(J) = TERROR
      TLAST(J) = ERROR
      KMAXU1 = KMAXU + 1
      TBETA(J) = BETA(KMAXU1)*ERROR
C
C     AFTER COMPUTING EACH OF THE T-EIGENVECTORS,
C     CHECK THE SIZE OF THE ERROR ESTIMATE, ERROR.
C     IF THIS ESTIMATE IS NOT AS SMALL AS DESIRED AND
C     |MA(J)| < ML(J), ATTEMPT TO INCREASE THE SIZE OF |MA(J)|
C     AND REPEAT THE T-EIGENVECTOR COMPUTATIONS.
C
      IF(ERROR.LT.ERTOL.OR.TFLAG.EQ.1)  GO TO 760
C
      IF(ERROR.GE.ERRMIN) GO TO 670
C     LAST COMPONENT IS LESS THAN MINIMAL TO DATE
      ERRMIN = ERROR
      MABEST = MA(J)
  670 CONTINUE
C
      IF(MA(J).GT.0)  ITEST = MA(J) + IDELTA(J)
      IF(MA(J).LT.0)  ITEST = -(IABS(MA(J)) + IDELTA(J))
      IF(IABS(ITEST).LE.ML(J).AND.ICOUNT.LE.10) GO TO 690
C     NEW MA(J) IS GREATER THAN MAXIMUM ALLOWED.
      IF(ERCONT.EQ.0.OR.MABEST.EQ.MPMIN) GO TO 710
      TFLAG = 1
      MA(J) = MABEST
      IF(ILBIS.EQ.0)   MTOL = MTOL - KMAXU
      WRITE(6,680) MA(J)
  680 FORMAT(' 10 ORDERS WERE CONSIDERED.  NONE SATISFIED THE ERROR TEST
     1'/' THEREFORE USE THE BEST ORDER OBTAINED FOR THE T-EIGENVECTORS'
     1,I6)
      GO TO 590
C
  690 MA(J) = ITEST
C
      MT = IABS(MA(J))
      IF(IWRITE.EQ.1.AND.ILBIS.EQ.0) WRITE(6,700) MT
  700 FORMAT(/' CHANGE SIZE OF T-MATRIX TO ',I6,' RECOMPUTE T-EIGENVECTO
     1R')
C
```

```
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU
C
      GO TO 590
C
C     APPROPRIATE SIZE T-MATRIX WAS NOT OBTAINED
  710 CONTINUE
      WRITE(10,720) J,SVAL,MP(J)
  720 FORMAT(/' ON 10 INCREMENTS NOT ABLE TO IDENTIFY APPROPRIATE SIZE
     1T-MATRIX FOR'/
     1I4,' TH SINGULAR VALUE  = ',E20.12,' T-MULTIPLICITY =',I4/)
      IF(M2(J).EQ.KMAX) WRITE(10,730)
      IF(M2(J).LT.KMAX) WRITE(10,740)
  730 FORMAT(/' ORDERS TESTED RANGED FROM 5*M1(J)/4 TO APPROXIMATELY'/
     1 '  MIN(11*MEV/8, 13*M1(J)/8)'/)
  740 FORMAT(/' ORDERS TESTED RANGED FROM (3*M1(J)+M2(J))/4  TO APPROXIM
     1ATELY'/'  (3*M1(J)+5*M2(J))/8'/)
      WRITE(10,750)
  750 FORMAT(' ALLOWING LARGER ORDERS FOR THE T-MATRICES MAY RESULT IN
     1 SUCCESS'/' BUT PROBABLY WILL NOT.  PROBLEM IS PROBABLY DUE TO'
     1 /' LACK OF CONVERGENCE OF GIVEN SINGULAR VALUE, CHECK THE ERROR E
     1STIMATE')
      MP(J) = MPMIN
      IF(ILBIS.EQ.0)  MTOL = MTOL - KMAXU
      GO TO 770
  760 NTVEC = NTVEC + 1
C
  770 CONTINUE
      NGOODC = NGOOD
      GO TO 800
C
C     COME HERE IF THERE IS NOT ENOUGH ROOM FOR ALL OF T-EIGENVECTORS
  780 NGOODC = J-1
      WRITE(6,790) J,MTOL,MDIMTV
  790 FORMAT(/' NOT ENOUGH ROOM IN TVEC ARRAY FOR ',I4,'TH T-EIGENVECTOR
     1'/'  TVEC DIMENSION REQUESTED = ',I6,' BUT TVEC HAS DIMENSION ',I6
     1/)
      IF(NGOODC.EQ.0) GO TO 1740
      MTOL = MTOL-KMAXU
C
  800 CONTINUE
C
C     THE LOOP ON T-EIGENVECTOR COMPUTATIONS IS COMPLETE.
C     WRITE OUT THE SIZE T-MATRICES THAT WILL BE USED FOR
C     THE RITZ VECTOR COMPUTATIONS.
C
      WRITE(10,810)
  810 FORMAT(/' SIZES OF T-MATRICES THAT WILL BE USED IN THE RITZ COMPUT
     1ATIONS'/5X,'J',8X,' SINGULAR VALUE  ',1X,'MA(J)')
C
      WRITE(10,820)   (J,GOODSV(J),MA(J),  J=1,NGOOD)
  820 FORMAT(I6,E25.14,I6)
      WRITE(10,570)
C
      WRITE(6,830) MTOL
  830 FORMAT(/' THE CUMULATIVE LENGTH OF THE T-EIGENVECTORS IS',I18)
```

```
C
      WRITE(6,840) NTVEC,NGOOD
  840 FORMAT(/I6,' T-EIGENVECTORS OUT OF',I6,' REQUESTED WERE COMPUTED')
C
C     SAVE THE T-EIGENVECTORS ON FILE 11?
      IF(TVSTOP.NE.1.AND.SVTVEC.EQ.0) GO TO 900
C
      WRITE(11,850) NTVEC,MTOL,MATNO,SVSEED
  850 FORMAT(I6,3I12,' = NTVEC,MTOL,MATNO,SVSEED')
C
      DO 880 J=1,NGOODC
C     IF MP(J) = MPMIN THEN NO SUITABLE T-EIGENVECTOR IS AVAILABLE
C     FOR THAT SINGULAR VALUE.
      IF(MP(J).EQ.MPMIN) WRITE(11,860) J,MA(J),GOODSV(J),MP(J)
  860 FORMAT(2I6,E20.12,I6/' TH SINGVAL,T-SIZE,SVALUE,FLAG,NO EIGVEC')
      IF(MP(J).NE.MPMIN) WRITE(11,870) J,MA(J),GOODSV(J),MP(J)
  870 FORMAT(I6,I6,E20.12,I6/' T-EIGVEC,SIZE T,SVALUE OF A,MP(J)')
      IF(MP(J).EQ.MPMIN) GO TO 880
      KI = MINT(J)
      KF = MFIN(J)
C
      WRITE(11,280) (TVEC(K), K=KI,KF)
C
  880 CONTINUE
C
      IF(TVSTOP.NE.1) GO TO 900
C
      WRITE(6,890) TVSTOP, NTVEC,NGOOD
  890 FORMAT(/' USER SET TVSTOP = ',I1/
     1' THEREFORE PROGRAM TERMINATES AFTER T-EIGENVECTOR COMPUTATIONS'/
     1' T-EIGENVECTORS THAT WERE COMPUTED ARE SAVED ON FILE 11'/
     1I8,' T-EIGENVECTORS WERE COMPUTED OUT OF',I7,' REQUESTED'/)
C
      GO TO 1860
C
  900 CONTINUE
C     IF NOT ALL OF THE REQUESTED T-EIGENVECTORS WERE COMPUTED,
C     ARE THE LANCZOS SINGULAR VECTOR COMPUTATIONS CONTINUED?
C
      IF(NTVEC.NE.NGOOD.AND.LVCONT.EQ.0) GO TO 1760
C
C     COMPUTE THE MAXIMUM SIZE OF THE T-MATRIX USED FOR THOSE
C     SINGULAR VALUES WITH GOOD ERROR ESTIMATES.
C
      KMAXU = 0
      DO 910 J = 1,NGOODC
      MT = IABS(MA(J))
      IF(MT.LT.KMAXU.OR.MP(J).EQ.MPMIN) GO TO 910
      KMAXU = MT
  910 CONTINUE
C
      IF(KMAXU.EQ.0) GO TO 1800
C
      WRITE(6,920) KMAXU
  920 FORMAT(/I6,' = LARGEST SIZE T-MATRIX TO BE USED IN THE RITZ VECTOR
```

```
     1 COMPUTATIONS')                                                   LSV08700
C                                                                       LSV08710
C     COUNT THE NUMBER OF RITZ VECTORS NOT BEING COMPUTED               LSV08720
      MREJEC = 0                                                        LSV08730
      DO 930 J=1,NGOODC                                                 LSV08740
  930 IF(MP(J).EQ.MPMIN)  MREJEC = MREJEC + 1                           LSV08750
      MREJET = MREJEC + (NGOOD-NGOODC)                                  LSV08760
      IF(MREJET.NE.0) WRITE(6,940) MREJET                               LSV08770
  940 FORMAT(/' RITZ VECTORS ARE NOT COMPUTED FOR',I6,' OF THE SINGULAR LSV08780
     1VALUES'/)                                                         LSV08790
      NACT = NGOODC - MREJEC                                            LSV08800
      WRITE(6,950) NGOOD,NTVEC,NACT                                     LSV08810
  950 FORMAT(/I6,' RITZ VECTORS WERE REQUESTED'/I6,' T-EIGENVECTORS WERELSV08820
     1 COMPUTED'/I6,' RITZ VECTORS WILL BE COMPUTED'/)                  LSV08830
C     CHECK IF THERE ARE ANY RITZ VECTORS TO COMPUTE                    LSV08840
      IF(MREJEC.EQ.NGOODC) GO TO 1780                                   LSV08850
C                                                                       LSV08860
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS?                    LSV08870
      IF(LVCONT.EQ.0.AND.MREJEC.NE.0) GO TO 1760                        LSV08880
C                                                                       LSV08890
C     NOW COMPUTE THE RITZ VECTORS.  REGENERATE THE                     LSV08900
C     LANCZOS VECTORS.                                                  LSV08910
C                                                                       LSV08920
      DO 960 I = 1,MNMAX                                                LSV08930
  960 RITVEC(I) = ZERO                                                  LSV08940
C                                                                       LSV08950
C     REGENERATE THE STARTING VECTOR. THIS MUST BE GENERATED AND        LSV08960
C     NORMALIZED PRECISELY THE WAY IT WAS DONE IN THE CORRESPONDING     LSV08970
C     SINGULAR VALUE COMPUTATIONS, OTHERWISE THERE WILL BE A            LSV08980
C     MISMATCH BETWEEN THE T-EIGENVECTORS THAT HAVE BEEN COMPUTED       LSV08990
C     FROM THE T-MATRICES READ IN FROM FILE 2 (IF THEY WERE READ IN)    LSV09000
C     AND THE LANCZOS TRIDIAGONAL MATRICES  THAT ARE BEING REGENERATED. LSV09010
C                                                                       LSV09020
C     STARTING VECTORS ARE OF THE FORM (V1,0) OR (0,V2) WHERE V1 IS     LSV09030
C     OF LENGTH M AND V2 IS OF LENGTH N.  SUCCEEDING LANCZOS VECTORS    LSV09040
C     ALTERNATE BETWEEN THESE TWO FORMS AND THE DIAGONAL ENTRIES OF THE LSV09050
C     T-MATRICES ALL VANISH.  THE PARAMETER IPARO DETERMINES THE SHAPE  LSV09060
C     OF THE STARTING VECTOR.  IF IPARO=1, THEN STARTING VECTOR WAS     LSV09070
C     OF THE FORM (0,V2).   IF IPARO=2, THEN STARTING VECTOR WAS OF     LSV09080
C     THE FORM (V1,0).                                                  LSV09090
C     REGENERATE STARTING VECTOR                                        LSV09100
      BATA = ZERO                                                       LSV09110
      IPAR = IPARO                                                      LSV09120
      ITNUM = 1                                                         LSV09130
      IF (IPAR.EQ.2) GO TO 1020                                         LSV09140
C                                                                       LSV09150
C-----------------------------------------------------------------------LSV09160
C     IPAR = 1 SO SET V2 TO RANDOM UNIT VECTOR AND SET V1 = 0.          LSV09170
C                                                                       LSV09180
      IIL = SVSEED                                                      LSV09190
      CALL GENRAN(IIL,G,N)                                              LSV09200
C                                                                       LSV09210
C-----------------------------------------------------------------------LSV09220
C                                                                       LSV09230
      DO 970 J = 1,N                                                    LSV09240
```

```
  970 V2(J) = G(J)
C-----------------------------------------------------------------------
      SUM = ONE/DSQRT(FINPRO(N,V2,1,V2,1))
C-----------------------------------------------------------------------
C
      DO 980 J = 1,M
  980 V1(J) = ZERO
C
      DO 990 J = 1,N
  990 V2(J) = V2(J)*SUM
C
C     INITIALIZE RITZ VECTORS
      DO 1010 J = 1,NGOODC
      IF (MP(J).EQ.MPMIN) GO TO 1010
      LL = MN*J - N
      II = MINT(J)
      TEMP = TVEC(II)
C
      DO 1000 K = 1,N
      LL = LL + 1
 1000 RITVEC(LL) = TEMP*V2(K)
C
 1010 CONTINUE
C
      GO TO 1150
C
 1020 CONTINUE
C
C-----------------------------------------------------------------------
C     IPAR = 2 SO SET V1 TO RANDOM UNIT VECTOR AND SET V2 = 0.
C
      CALL GENRAN(SVSEED,G,M)
C
C-----------------------------------------------------------------------
C
      DO 1030 J = 1,M
 1030 V1(J) = G(J)
C-----------------------------------------------------------------------
      SUM = ONE/DSQRT(FINPRO(M,V1,1,V1,1))
C-----------------------------------------------------------------------
C
      DO 1040 J = 1,N
 1040 V2(J) = ZERO
C
      DO 1050 J = 1,M
 1050 V1(J) = V1(J)*SUM
C
C     INITIALIZE RITZ VECTORS
      DO 1070 J = 1,NGOODC
      IF (MP(J).EQ.MPMIN) GO TO 1070
      LL = MN*(J-1)
      II = MINT(J)
      TEMP = TVEC(II)
C
      DO 1060 K = 1,M
```

```
      LL = LL + 1
 1060 RITVEC(LL) = TEMP*V1(K)
C
 1070 CONTINUE
C
 1080 CONTINUE
C
C  DO ONE ITERATION OF LANCZOS WHERE NEW LANCZOS VECTOR WILL HAVE THE
C  FORM  (0,V2).
C
C-----------------------------------------------------------------------
C
      CALL STRAN(V1,V2,BATA)
C
C-----------------------------------------------------------------------
C
C-----------------------------------------------------------------------
      BATA = DSQRT(FINPRO(N,V2,1,V2,1))
C-----------------------------------------------------------------------
      SUM = ONE/BATA
      ITNUM = ITNUM + 1
      IPAR = 2
C
      TEMP = BETA(ITNUM)
      TEMP = DABS(BATA - TEMP)/TEMP
      IF (TEMP.LT.1.0D-10) GO TO 1110
C
C     HISTORY MISMATCH ON REGENERATION THUS DEFAULT
 1090 WRITE(6,1100) ITNUM,IPAR,BATA,BETA(ITNUM),TEMP
 1100 FORMAT(1X,'ITNUM',2X,'IPAR',16X,'BATA',16X,'BETA',14X,'RELERR'/
     1 2I6,3E20.12/' BATA AND BETA DO NOT AGREE SO PROGRAM STOPS'/)
      GO TO 1860
C
 1110 CONTINUE
C     NORMALIZE LANCZOS VECTOR
      DO 1120 J = 1,N
 1120 V2(J) = V2(J)*SUM
C
C  UPDATE RITZ VECTORS
      DO 1140 J = 1,NGOODC
      IF (IABS(MA(J)).LT.ITNUM.OR.MP(J).EQ.MPMIN) GO TO 1140
      LL = MN*J - N
      II = MINT(J) + ITNUM - 1
      TEMP = TVEC(II)
C
      DO 1130 K = 1,N
      LL = LL + 1
 1130 RITVEC(LL) = TEMP*V2(K) + RITVEC(LL)
C
 1140 CONTINUE
C     HAVE ALL REQUIRED LANCZOS VECTORS BEEN REGENERATED ?
C
      IF(ITNUM.EQ.KMAXU) GO TO 1190
C
 1150 CONTINUE
```

```
C
C     DO ONE ITERATION OF LANCZOS WHERE NEW LANCZOS VECTOR WILL HAVE
C     THE FORM  (V1,0).
C
C-----------------------------------------------------------------------
C
      CALL SVMAT(V2,V1,BATA)
C
C-----------------------------------------------------------------------
C
C-----------------------------------------------------------------------
      BATA = DSQRT(FINPRO(M,V1,1,V1,1))
C-----------------------------------------------------------------------
      SUM = ONE/BATA
      ITNUM = ITNUM + 1
      IPAR = 1
C
      TEMP = BETA(ITNUM)
      TEMP = DABS(BATA - TEMP)/TEMP
      IF (TEMP.GE.1.0D-10) GO TO 1090
C
C     NORMALIZE LANCZOS VECTOR
      DO 1160 J = 1,M
 1160 V1(J) = V1(J)*SUM
C
C     UPDATE RITZ VECTORS
      DO 1180 J = 1,NGOODC
      IF (IABS(MA(J)).LT.ITNUM.OR.MP(J).EQ.MPMIN) GO TO 1180
      LL = MN*(J-1)
      II = MINT(J) + ITNUM - 1
      TEMP = TVEC(II)
C
      DO 1170 K = 1,M
      LL = LL + 1
 1170 RITVEC(LL) = TEMP*V1(K) + RITVEC(LL)
C
 1180 CONTINUE
C  HAVE ALL REQUIRED LANCZOS VECTORS BEEN COMPUTED ?
      IF (ITNUM.LT.KMAXU) GO TO 1080
C
 1190 CONTINUE
C
C     RITZVECTOR GENERATION IS COMPLETE. NORMALIZE EACH RITZVECTOR
C     AS AN EIGENVECTOR OF THE ASSOCIATED SYMMETRIC MATRIX B.
C     THEN COMPUTE THE ERRORS IN THESE VECTORS AS EIGENVECTORS
C     OF B AND WRITE THESE OUT TO FILE 9.  THEN INDIVIDUALLY
C     NORMALIZE THE FIRST M AND THE LAST N COMPONENTS OF EACH OF
C     THESE RITZ VECTORS AND TAKE THESE NORMALIZED VECTORS AS
C     RESPECTIVELY APPROXIMATIONS TO THE LEFT AND TO THE RIGHT
C     SINGULAR VECTORS OF THE CORRESPONDING SINGULAR VALUE OF
C     THE ORIGINAL MATRIX.
C
C
C     NORMALIZE THE RITZ VECTORS AS EIGENVECTORS OF B
      DO 1280 J = 1,NGOODC
```

```
      IF (MP(J).EQ.MPMIN) GO TO 1280
      LINT = MN*(J-1) + 1
      LFIN = MN*J
      SUM = ZERO
      SVAL = SVNEW(J)
C
      DO 1200 K = LINT,LFIN
 1200 SUM = SUM + RITVEC(K)*RITVEC(K)
C
      SUM = DSQRT(SUM)
      RNORM(J) = SUM
      TEMP = ONE - SUM
      SUM = ONE/SUM
C
      DO 1210 K = LINT,LFIN
 1210 RITVEC(K) = RITVEC(K)*SUM
C
C     COMPUTE ERROR IN RITZ VECTOR CONSIDERED AS AN EIGENVECTOR OF B.
      LINTM = LINT + M
      L = LINT - 1
      DO 1220 K = 1,M
      L = L + 1
 1220 V1(K) = RITVEC(L)
      DO 1230 K = 1,N
      L = L + 1
 1230 V2(K) = RITVEC(L)
C
C-----------------------------------------------------------------------
C
      CALL SVMAT(RITVEC(LINTM),V1,SVAL)
      CALL STRAN(RITVEC(LINT),V2,SVAL)
C
C-----------------------------------------------------------------------
C
      SUM = ZERO
      DO 1240 JJ = 1,M
 1240 SUM = SUM + V1(JJ)*V1(JJ)
C
      DO 1250 JJ = 1,N
 1250 SUM = SUM + V2(JJ)*V2(JJ)
C
      IF(IWRITE.NE.0)  WRITE(6,1260) J,GOODSV(J)
 1260 FORMAT(/I5,'TH SINGULAR VALUE CONSIDERED =',E20.12/)
C
      IF(IWRITE.NE.0) WRITE(6,1270) TERR(J), TBETA(J), RNORM(J)
 1270 FORMAT(' RESIDUAL FOR T-EIGENVECTOR = ',E14.3/
     1' DABS(BETA(MA(J)+1)*U(MA(J)) = ' ,E14.3/
     1' NORM(RITZVEC) = ', E14.3/)
C
      SUM = DSQRT(SUM)
      BERR(J) = SUM
      BERRGP(J) = SUM/ABS(BMINGP(J))
 1280 CONTINUE
C
C     RITZVECTORS ARE NORMALIZED AND B-MATRIX ESTIMATES ARE IN BERR
```

```
C     AND IN BERRGP ARRAYS.   STORE THESE ESTIMATES BUT NOT THE
C     VECTORS.
C
      WRITE(9,1290)
 1290 FORMAT(11X,'GOODSV(J)',3X,'MA(J)',2X,'  BMINGAP',6X,' BERROR',2X,
     1 'BERROR/BGAP',4X,' TERROR')
C
      WRITE(13,1300)
 1300 FORMAT(11X,'GOODSV(J)',5X,'RITZNORM',5X,' BMINGAP',5X,'TBETA(J)',
     1 5X,'TLAST(J)')
C
      DO 1330 J=1,NGOODC
C
      IF(MP(J).EQ.MPMIN) GO TO 1330
C
      WRITE(9,1310)SVNEW(J),MA(J),BMINGP(J),BERR(J),BERRGP(J),TERR(J)
 1310 FORMAT(E20.12,I6,4E13.5)
C
      WRITE(13,1320) SVNEW(J),RNORM(J),BMINGP(J),TBETA(J),TLAST(J)
 1320 FORMAT(E20.12,4E13.5)
C
 1330 CONTINUE
C
      IF (MREJEC.EQ.0) GO TO 1410
C
      WRITE(9,1340)
 1340 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING SINGULA
     1R VALUES'/' EITHER BECAUSE THEY HAD NOT CONVERGED OR BECAUSE THE E
     1RROR ESTIMATE'/'  WAS NOT AS SMALL AS DESIRED'/)
C
      WRITE(13,1350)
 1350 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING SINGULA
     1R VALUES'/' EITHER BECAUSE THEY HAD NOT CONVERGED OR BECAUSE THE E
     1ERROR ESTIMATE'/' WAS NOT AS SMALL AS DESIRED'/)
C
      DO 1400 J = 1,NGOODC
      IF(MP(J).NE.MPMIN) GO TO 1400
C     EACH SINGULAR VALUE FOR WHICH NO SINGULAR VECTOR WAS CALCULATED
C     HAS INFORMATION OUTPUTTED TO FILES 4 AND 13
C
      WRITE(9,1360)
 1360 FORMAT(6X,'GOODSV(J)',3X,'MA(J)',5X,'BMINGP(J)',6X,'TLAST(J)',
     1 6X,'TBETA(J)',3X,'MP(J)')
      WRITE(9,1370) GOODSV(J),MA(J),BMINGP(J),TLAST(J),TBETA(J),MP(J)
 1370 FORMAT(E15.8,I8,3E14.4,I8)
C
      WRITE(13,1380)
 1380 FORMAT(6X,'GOODSV(J)',3X,'MA(J)',3X,'M1(J)',3X,'M2(J)',3X,'MP(J)'
     1/)
      WRITE(13,1390) GOODSV(J),MA(J),M1(J),M2(J),MP(J)
 1390 FORMAT(E15.8,4I8)
C
 1400 CONTINUE
C
 1410 CONTINUE
```

```
C
      WRITE(9,1420)
 1420 FORMAT(/' ABOVE ARE ERROR ESTIMATES FOR THE B AND T EIGENVECTORS'/
     1 ' ASSOCIATED WITH THE GOODSV LISTED IN COLUMN 1'/
     1 ' BERROR = NORM(B*X - SV*X)  TERROR = NORM(T*Y - SV*Y) '/
     1 ' WHERE T = T(1,MA(J))    X = RITZ VECTOR = V*Y  V = SUCCESSIVE'/
     1 ' LANCZOS VECTORS. BMINGAP = GAP TO NEAREST B-EIGENVALUE'//)
C
      WRITE(13,1430)
 1430 FORMAT(/' ABOVE ARE ERROR ESTIMATES FOR THE GOODSVS'/
     1 ' RITZNORM = NORM(COMPUTED RITZ VECTOR FOR B-MATRIX'/
     1 ' TBETA(J) = BETA(MA(J)+1)*Y(MA(J)),  T*Y = SV*Y '/
     1 ' TLAST(J) = DABS(Y(MA(J)) '/)
C
C     NUMBER OF RITZ VECTORS COMPUTED
      NCOMPU = NGOODC - MREJEC
      WRITE(12,1440) N,NCOMPU,NGOODC,MATNO
 1440 FORMAT(3I6,I12,' SIZE A, NO.RITZVECS, NO.SVALUES,MATNO')
C
C     INDIVIDUALLY NORMALIZE THE FIRST M AND THE LAST N COMPONENTS OF
C     EACH RITZ VECTOR.
C
      LFIN = 0
      DO 1560 J = 1,NGOODC
C
      IF(MP(J).EQ.MPMIN) GO TO 1540
C
C     RITZ VECTOR WAS COMPUTED
      LINT = MN*(J-1) + 1
      LFIN = MN*J
      LFIN1 = LINT + M - 1
      LINT1 = LFIN1 + 1
C
      SUM = 0.D0
      TEMP = 0.D0
      DO 1450 I = LINT,LFIN1
 1450 SUM = SUM + RITVEC(I)*RITVEC(I)
      SUM = ONE/DSQRT(SUM)
      DO 1460 I = LINT,LFIN1
 1460 RITVEC(I) = SUM*RITVEC(I)
      DO 1470 I = LINT1,LFIN
 1470 TEMP = TEMP + RITVEC(I)*RITVEC(I)
      TEMP = ONE/DSQRT(TEMP)
      DO 1480 I = LINT1,LFIN
 1480 RITVEC(I) = TEMP*RITVEC(I)
C
      WRITE(12,1490) J, GOODSV(J), MP(J)
 1490 FORMAT(/I6,4X,E20.12,I6,' J, SINGULAR VALUE, MP(J)')
C
      WRITE(12,1500) BERR(J),BERRGP(J)
 1500 FORMAT(2E15.5,' = NORM(B*Z-SVAL*Z) AND  NORM(B*Z-SVAL*Z)/BMINGAP')
C
      WRITE(12,1510) J
 1510 FORMAT(/I6,'TH LEFT SINGULAR VECTOR'/)
C     WRITE(12,170) (RITVEC(LL), LL=LINT,LFIN1)
```

```
      WRITE(12,1520) (RITVEC(LL), LL=LINT,LFIN1)
 1520 FORMAT(4E20.12)
C
      WRITE(12,1530) J
 1530 FORMAT(/I6,'TH RIGHT SINGULAR VECTOR'/)
C     WRITE(12,170) (RITVEC(LL), LL=LINT1,LFIN)
      WRITE(12,1520) (RITVEC(LL), LL=LINT1,LFIN)
C
      GO TO 1560
C
C     NO RITZ VECTOR WAS COMPUTED FOR THIS SINGULAR VALUE
 1540 WRITE(12,1550) J,GOODSV(J),MP(J)
 1550 FORMAT(I6,4X,E20.12,I6,' J,SINGVALUE,MP(J),NO RITZ VECTOR COMPUTED
     1')
C
 1560 CONTINUE
C
C     DID ANY T-MATRICES INCLUDE OFF-DIAGONAL ENTRIES SMALLER THAN
C     DESIRED, AS SPECIFIED BY BTOL?
C
      IF(IB.GT.0) GO TO 1590
      WRITE(6,1570) KMAXU
 1570 FORMAT(/' FOR LARGEST T-MATRIX CONSIDERED',I7,' CHECK THE SIZE OF
     1BETAS')
C
C-----------------------------------------------------------------------
C
      CALL TNORM(BETA,BKMIN,TEMP,KMAXU,IBMT)
C
C-----------------------------------------------------------------------
C
      IF(IBMT.LT.0) WRITE (6,1580)
 1580 FORMAT(/' WARNING THE T-MATRICES FOR ONE OR MORE OF THE SINGULAR V
     1ALUES CONSIDERED'/' HAD AN OFF-DIAGONAL ENTRY THAT WAS SMALLER THA
     1N THE BETA TOLERANCE THAT WAS SPECIFIED'/)
 1590 CONTINUE
C
       GO TO 1860
C
 1600 WRITE(6,1610) NGOOD,MNMAX,MDIMRV
 1610 FORMAT(/I4,' RITZ VECTORS WERE REQUESTED BUT THE REQUIRED DIMENSIO
     1N',I6/' IS LARGER THAN THE USER-SPECIFIED DIMENSION OF RITVEC',I6
     1/' THEREFORE, THE SINGULAR VECTOR PROCEDURE TERMINATES FOR THE USE
     1R TO INTERVENE')
C
      GO TO 1860
C
 1620 WRITE(6,1630)  MOLD,M,NOLD,N,MATOLD,MATNO
 1630 FORMAT(/' GOODSV PARAMETERS READ FROM FILE 3 DO NOT AGREE WITH THO
     1SE'/' SPECIFIED BY THE USER. MOLD,M,NOLD,N,MATOLD,MATNO ='/
     14I6, 2I12/' THEREFORE PROGRAM TERMINATES FOR USER TO RESOLVE DIFFE
     1RENCES'/)
C
      GO TO 1860
C
```

```
 1640 WRITE(6,1650)                                                     LSV13100
 1650 FORMAT(/' PARAMETERS IN BETA FILE DO NOT AGREE WITH THOSE SPECIFIELSV13110
     1D BY THE USER.'/' THEREFORE, THE PROGRAM TERMINATES FOR THE USER TLSV13120
     1O RESOLVE THE DIFFERENCES'/)                                      LSV13130
C                                                                       LSV13140
      GO TO 1860                                                        LSV13150
C                                                                       LSV13160
 1660 WRITE(6,1670) KMAX,MEV                                            LSV13170
 1670 FORMAT(/' IN BETA HISTORY HEADER KMAX =',I6/                      LSV13180
     1' BUT SINGULAR VALUES WERE COMPUTED AT MEV = ',I6,' PROGRAM STOPS'LSV13190
     1)                                                                 LSV13200
C                                                                       LSV13210
      GO TO 1860                                                        LSV13220
C                                                                       LSV13230
 1680 WRITE(6,1690) MEV                                                 LSV13240
 1690 FORMAT(/' SOMETHING IS WRONG.'/' HEADER SAYS THAT SIZE T-MATRIX USLSV13250
     1ED IN THE SINGULAR VALUE COMPUTATIONS WAS = ',I6/' BUT THIS IS AN LSV13260
     1ODD ORDER AND THAT IS NOT ALLOWED.  PROGRAM STOPS'/)              LSV13270
C                                                                       LSV13280
      GO TO 1860                                                        LSV13290
C                                                                       LSV13300
 1700 WRITE(6,1710)                                                     LSV13310
 1710 FORMAT(/' PROGRAM COMPUTED 1ST GUESSES AT T-MATRIX SIZES, READ THELSV13320
     1M TO FILE 10'/' THEN TERMINATED AS REQUESTED.')                   LSV13330
      GO TO 1860                                                        LSV13340
C                                                                       LSV13350
 1720 WRITE(6,1730) MTOL, MDIMTV                                        LSV13360
 1730 FORMAT(/' PROGRAM TERMINATES BECAUSE THE TVEC DIMENSION ANTICIPATELSV13370
     1D',I7/'  IS LARGER THAN THE TVEC DIMENSION',I7,' SPECIFIED BY THE LSV13380
     1USER.'/'  USER MAY RESET THE TVEC DIMENSION AND RESTART THE PROGRALSV13390
     1M'/)                                                              LSV13400
      GO TO 1860                                                        LSV13410
C                                                                       LSV13420
 1740 WRITE(6,1750)                                                     LSV13430
 1750 FORMAT(/' PROGRAM TERMINATES BECAUSE NO SUITABLE T-EIGENVECTORS WELSV13440
     1RE IDENTIFIED'/' FOR ANY OF THE SINGULAR VALUES SUPPLIED.  PROBLEMLSV13450
     1 COULD BE CAUSED BY'/' TOO SMALL A TVEC DIMENSION OR SIMPLY BE THALSV13460
     1T NO SUITABLE T-VECTORS'/' WERE IDENTIFIED.  USER SHOULD CHECK OUTLSV13470
     1PUT'/)                                                            LSV13480
      GO TO 1860                                                        LSV13490
C                                                                       LSV13500
 1760 WRITE(6,1770) LVCONT,NTVEC,NGOOD                                  LSV13510
 1770 FORMAT(/' LVCONT FLAG =',I2,' AND NUMBER ',I5,' OF T-EIGENVECTORS LSV13520
     1 COMPUTED N.E.'/' NUMBER',I5,' REQUESTED SO PROGRAM TERMINATES'/) LSV13530
      GO TO 1860                                                        LSV13540
 1780 WRITE(6,1790)                                                     LSV13550
 1790 FORMAT(/' PROGRAM TERMINATES WITHOUT COMPUTING ANY RITZ VECTORS'/ LSV13560
     1/' BECAUSE ALL OF THE T-EIGENVECTORS WERE REJECTED AS NOT SUITABLELSV13570
     1 FOR'/' THE RITZ VECTOR COMPUTATIONS.  PROBABLE CAUSE WAS LACK OF LSV13580
     1CONVERGENCE'/' OF THE SINGULAR VALUES'/)                          LSV13590
       GO TO 1860                                                       LSV13600
C                                                                       LSV13610
 1800 WRITE(6,1810)                                                     LSV13620
 1810 FORMAT(/' PROGRAM INDICATES THAT IT IS NOT POSSIBLE TO COMPUTE ANYLSV13630
     1 OF THE'/' REQUESTED EIGENVECTORS. THEREFORE PROGRAM TERMINATES') LSV13640
```

```
      DO 1820 J=1,NGOODC
 1820 WRITE(6,1830)  J,GOODSV(J),MP(J)
 1830 FORMAT(/4X,' J',11X,'GOODSV(J)',4X,'MP(J)'/I6,E20.12,I9)
      GO TO 1860
C
 1840 WRITE(6,1850) MBETA,KMAXN
 1850 FORMAT(/' PROGRAM TERMINATES BECAUSE THE STORAGE ALLOTTED FOR THE
     1BETA ARRAY',I8/' IS NOT SUFFICIENT FOR THE ENLARGED KMAX =',I8,' T
     1HAT THE PROGRAM WANTS'/' USER CAN ENLARGE THE BETA ARRAY AND RERUN
     1 THE PROGRAM'/)
C
 1860 CONTINUE
C
      STOP
C-----END OF MAIN PROGRAM FOR LANCZOS SINGULAR VECTOR COMPUTATIONS------
      END
```

6.5 LSMULT: LANCZS and Sample Matrix-Vector Multiply Subroutines

```
C-----LSMULT------------------------------------------------------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C         REFERENCE: Cullum and Willoughby, Chapter 5
C         Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C         Applied Mathematics, 2002. SIAM Publications,
C         Philadelphia, PA. USA
C
C
C
C     CONTAINS SUBROUTINES LANCZS, USPECS, STRAN, AND SVMAT
C     FOR USE WITH THE LANCZOS SINGULAR VALUE/VECTOR PROGRAMS
C
C     NONPORTABLE CONSTRUCTIONS:
C     1.  THE ENTRY MECHANISM USED TO PASS THE STORAGE LOCATIONS
C         OF THE USER-SPECIFIED MATRIX FROM THE SUBROUTINE USPEC
C         TO THE MATRIX-VECTOR MULTIPLY SUBROUTINES SVMAT AND
C         STRAN.
C     2.  IN THE SAMPLE USPEC PROVIDED:  THE FREE FORMAT (8,*),
C         AND THE FORMAT (20A4).
C
C-----START OF LANCZS---------------------------------------------------
C
      SUBROUTINE LANCZS(MATVEC,MTRAN,BETA,V1,V2,G,KMAX,MOLD1,
     1                  M,N,IPAR,IIX)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION BETA(1),V1(1),V2(1),SUM,TEMP,ONE,ZERO
      REAL G(1)
      DOUBLE PRECISION FINPRO
      INTEGER IPAR
      EXTERNAL MATVEC,MTRAN
C-----------------------------------------------------------------------
```

```
C     COMPUTE T(1,MEV) FOR SYMMETRIZED VERSION OF GIVEN A-MATRIX.
C
C                              ----                ----
C                             |  0                  A  |
C                B  =         |                        |
C                             |  A-TRANSPOSE        0  |
C                              ----                ----
C
C     WHERE A IS AN M BY N REAL SPARSE MATRIX, USING STARTING
C     VECTORS OF THE FORM (V1,0) WHEN THE FLAG IPAR = 2 AND
C     OF THE FORM (0,V2) WHEN THE FLAG IPAR = 1.  V1 IS OF
C     DIMENSION M, THE ROW DIMENSION OF A, AND V2 IS OF DIMENSION
C     N, THE COLUMN DIMENSION OF A.
C
C     WITH STARTING VECTORS OF THESE FORMS, THE LANCZOS VECTORS
C     GENERATED ALTERNATE BETWEEN THESE 2 FORMS AND ALL OF THE
C     DIAGONAL ENTRIES OF THE LANCZOS TRIDIAGONAL MATRICES T(1,MEV)
C     GENERATED ARE 0.
C
C     LANCZS USES 2 USER-SUPPLIED SUBROUTINES MATVEC AND MTRAN.
C     MAIN PROGRAM CALLS THESE SVMAT AND STRAN, RESPECTIVELY.
C     CALLING SEQUENCES ARE
C
C           CALL MATVEC(V2,V1,SUM)
C           CALL MTRAN(V1,V2,SUM)
C
C     MATVEC  COMPUTES V1 = A*V2 - SUM*V1.
C     MTRAN COMPUTES V2 = (A-TRANSPOSE)*V1 - SUM*V2.
C
C     ON EXIT V1 AND V2 CONTAIN THE NONZERO PARTS OF THE
C     LAST TWO LANCZOS VECTORS.
C
C     IF MOLD1 = 1 THEN T(1,KMAX) IS GENERATED FROM SCRATCH.
C     IF MOLD1 > 1 THEN A PREVIOUSLY-GENERATED T-MATRIX OF SIZE
C     (MOLD1-1) IS EXTENDED TO ONE OF SIZE KMAX. SINGULAR VALUE
C     PRGORAMS CAN ONLY UTILIZE T-MATRICES OF EVEN ORDER.
C     BETA(KMAX+1) IS ALSO COMPUTED FOR USE IN THE ERROR ESTIMATES.
C
C-----------------------------------------------------------------------
      ONE  = 1.0D0
      ZERO = 0.0D0
      ITNUM = MOLD1
C
      IF (ITNUM .GT. 1) GO TO (80,100), IPAR
C
C     NO PREVIOUS BETA HISTORY
      BETA(1) = ZERO
      IIL = IIX
      IF (IPAR .EQ. 2) GO TO 40
C
C-----------------------------------------------------------------------
C     IPAR = 1 SO SET V2 EQUAL TO A UNIT RANDOM VECTOR AND SET V1 = 0.
      CALL GENRAN(IIL,G,N)
C-----------------------------------------------------------------------
C
```

```
      DO 10 J = 1,N
   10 V2(J) = G(J)
C
C-----------------------------------------------------------------------
      TEMP = FINPRO(N,V2(1),1,V2(1),1)
C-----------------------------------------------------------------------
C
      SUM = ONE/DSQRT(TEMP)
      DO 20 J = 1,M
   20 V1(J) = ZERO
C
      DO 30 J = 1,N
   30 V2(J) = V2(J)*SUM
      GO TO 100
C
   40 CONTINUE
C
C-----------------------------------------------------------------------
C     IPAR = 2 SO SET V1 EQUAL TO A UNIT RANDOM VECTOR AND SET V2 = 0.
      CALL GENRAN(IIL,G,M)
C-----------------------------------------------------------------------
C
      DO 50 J=1,M
   50 V1(J) = G(J)
C
C-----------------------------------------------------------------------
      TEMP = FINPRO(M,V1(1),1,V1(1),1)
C-----------------------------------------------------------------------
C
      SUM = ONE/DSQRT(TEMP)
      DO 60 J = 1,N
   60 V2(J) = ZERO
      DO 70 J = 1,M
   70 V1(J) = V1(J)*SUM
C
C     BELOW IS START FOR MOLD1 > 1 AND IPAR = 1
C     DO ONE ITERATION OF LANCZOS TO OBTAIN (0,V2)
C
   80 CONTINUE
      SUM = BETA(ITNUM)
C
C-----------------------------------------------------------------------
      CALL MTRAN(V1,V2,SUM)
C-----------------------------------------------------------------------
C
C-----------------------------------------------------------------------
      SUM = FINPRO(N,V2(1),1,V2(1),1)
C-----------------------------------------------------------------------
C
      ITNUM = ITNUM + 1
      BETA(ITNUM) = DSQRT(SUM)
      SUM = ONE/BETA(ITNUM)
C
      DO 90 J = 1,N
   90 V2(J) = V2(J)*SUM
```

```
C                                                                       LSM01540
      IPAR = 2                                                          LSM01550
      IF (ITNUM .GT. KMAX) GO TO 120                                    LSM01560
C                                                                       LSM01570
C     BELOW IS START FOR MOLD1 > 1 AND IPAR = 2                         LSM01580
C     DO ONE ITERATION OF LANCZOS TO OBTAIN (V1,0)                      LSM01590
C                                                                       LSM01600
  100 CONTINUE                                                          LSM01610
      SUM = BETA(ITNUM)                                                 LSM01620
C                                                                       LSM01630
C-----------------------------------------------------------------------LSM01640
      CALL MATVEC(V2,V1,SUM)                                            LSM01650
C-----------------------------------------------------------------------LSM01660
C                                                                       LSM01670
C-----------------------------------------------------------------------LSM01680
      SUM = FINPRO(M,V1(1),1,V1(1),1)                                   LSM01690
C-----------------------------------------------------------------------LSM01700
C                                                                       LSM01710
      ITNUM = ITNUM + 1                                                 LSM01720
      BETA(ITNUM) = DSQRT(SUM)                                          LSM01730
      SUM = ONE/BETA(ITNUM)                                             LSM01740
C                                                                       LSM01750
      DO 110 J = 1,M                                                    LSM01760
  110 V1(J)= V1(J) * SUM                                                LSM01770
C                                                                       LSM01780
      IPAR = 1                                                          LSM01790
      IF (ITNUM .GT. KMAX) GO TO 120                                    LSM01800
      GO TO 80                                                          LSM01810
C                                                                       LSM01820
  120 CONTINUE                                                          LSM01830
C                                                                       LSM01840
      RETURN                                                            LSM01850
C-----END OF LANCZS-----------------------------------------------------LSM01860
      END                                                               LSM01870
C                                                                       LSM01880
C-----START OF USPEC (GENERAL SPARSE, RECTANGULAR MATRIX)---------------LSM01890
C                                                                       LSM01900
C     SUBROUTINE USPEC(M,N,MATNO)                                       LSM01910
      SUBROUTINE SUSPEC(M,N,MATNO)                                      LSM01920
C                                                                       LSM01930
C--------------------------------------------------------------------   LSM01940
      DOUBLE PRECISION A(10000)                                         LSM01950
      INTEGER IROW(10000),ICOL(3010)                                    LSM01960
C--------------------------------------------------------------------   LSM01970
C    DIMENSIONS ARRAYS NEEDED TO DEFINE THE USER-SUPPLIED               LSM01980
C    M X N RECTANGULAR A-MATRIX, READS IN VALUES OF THESE               LSM01990
C    ARRAYS AND THEN PASSES THE STORAGE LOCATIONS OF THESE              LSM02000
C    ARRAYS TO THE CORRESPONDING MATRIX-VECTOR MULTIPLY                 LSM02010
C    SUBROUTINES SVMAT AND STRAN.                                       LSM02020
C                                                                       LSM02030
C    THE A-MATRIX IS STORED IN THE FOLLOWING SPARSE FORMAT:             LSM02040
C    M = NUMBER OF ROWS IN A.                                           LSM02050
C    N = NUMBER OF COLUMNS IN A.                                        LSM02060
C    NZ = NUMBER OF NONZERO ENTRIES IN A-MATRIX.                        LSM02070
C    ICOL(J), J=1,N IS NUMBER OF NONZERO ENTRIES IN COLUMN J.           LSM02080
```

```
C    IROW(K), K = 1,NZ IS THE ROW INDEX FOR CORRESPONDING A(K).         LSM02090
C    A(K), K=1,NZ IS NONZERO ENTRIES IN A, COLUMN BY COLUMN.            LSM02100
C    IT IS ASSUMED THAT ICOL(J) > 0 FOR ALL J                           LSM02110
C                                                                       LSM02120
C     NOTE:  ASSOCIATED SUBROUTINES SVMAT AND STRAN ASSUME THAT         LSM02130
C             M >= N.                                                   LSM02140
C                                                                       LSM02150
C-----------------------------------------------------------------------LSM02160
C     READ IN  MATRIX FROM  FILE 8                                      LSM02170
C                                                                       LSM02180
      READ(8,10) NZ,MOLD,NOLD,MATOLD                                    LSM02190
   10 FORMAT(I10,2I6,I8)                                                LSM02200
C                                                                       LSM02210
      WRITE(6,20) NZ,MOLD,NOLD,MATOLD                                   LSM02220
   20 FORMAT(6X,'NZ',4X,'MOLD',4X,'NOLD',4X,'MATOLD'/I10,2I6,I10/)      LSM02230
C                                                                       LSM02240
C     TEST OF PARAMETER CORRECTNESS                                     LSM02250
      ITEMP = (MOLD-M)**2 + (NOLD-N)**2 + (MATOLD-MATNO)**2             LSM02260
C                                                                       LSM02270
      IF (ITEMP.EQ.0) GO TO 40                                          LSM02280
C                                                                       LSM02290
      WRITE(6,30)                                                       LSM02300
   30 FORMAT(' PROGRAM TERMINATES BECAUSE EITHER ORDERS OF OR LABELS FORLSM02310
     1 MATRIX DISAGREE')                                                LSM02320
      GO TO 70                                                          LSM02330
C                                                                       LSM02340
   40 CONTINUE                                                          LSM02350
C                                                                       LSM02360
C     NUMBER OF NONZERO ENTRIES IN EACH COLUMN IS READ IN               LSM02370
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS READ      LSM02380
      READ(8,50) (ICOL(K), K=1,N)                                       LSM02390
      READ(8,50) (IROW(K), K=1,NZ)                                      LSM02400
   50 FORMAT(13I6)                                                      LSM02410
C                                                                       LSM02420
C     READ IN THE NONZERO ENTRIES IN THE MATRIX                         LSM02430
      READ(8,60) (A(K), K=1,NZ)                                         LSM02440
   60 FORMAT(3E25.16)                                                   LSM02450
C  50 FORMAT(4E19.10)                                                   LSM02460
C                                                                       LSM02470
C-----------------------------------------------------------------------LSM02480
C     PASS STORAGE LOCATIONS OF ARRAYS THAT DEFINE THE MATRIX TO        LSM02490
C     THE MATRIX-VECTOR MULTIPLY SUBROUTINES SVMAT AND STRAN            LSM02500
      CALL SMATVE(A,ICOL,IROW,M,N)                                      LSM02510
      CALL STRANE(A,ICOL,IROW,M,N)                                      LSM02520
C-----------------------------------------------------------------------LSM02530
C                                                                       LSM02540
C-----END OF USPEC------------------------------------------------------LSM02550
      RETURN                                                            LSM02560
   70 STOP                                                              LSM02570
      END                                                               LSM02580
C                                                                       LSM02590
C-----STRAN (GENERAL SPARSE MATRIX)-------------------------------------LSM02600
C                                                                       LSM02610
C     SUBROUTINE STRAN(W,U,SUM)                                         LSM02620
      SUBROUTINE SSTRAN(W,U,SUM)                                        LSM02630
```

```
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION W(1),U(1),A(1),SUM,TEMP
      INTEGER IROW(1),ICOL(1)
C-----------------------------------------------------------------------
C     SUBROUTINE TO COMPUTE U = (A-TRANSPOSE)*W - SUM*U WHERE A IS
C     A GENERAL, SPARSE M X N MATRIX WITH M >= N.
C
C     ASSUMES MATRIX IS STORED IN SPARSE FORMAT GIVEN IN
C     CORRESPONDING USPEC SUBROUTINE.
C-----------------------------------------------------------------------
      JLAST = 0
      DO 20 J = 1,N
      JFIRST = JLAST + 1
      JLAST  = JLAST + ICOL(J)
      TEMP   = -SUM*U(J)
C
      DO 10 K = JFIRST,JLAST
      IK = IROW(K)
   10 TEMP = A(K)*W(IK) + TEMP
C
   20 U(J) = TEMP
C
      RETURN
C
C-----------------------------------------------------------------------
      ENTRY STRANE(A,ICOL,IROW,M,N)
C-----------------------------------------------------------------------
C
C-----END OF STRAN FOR GENERAL SPARSE MATRIX----------------------------
      RETURN
      END
C
C-----SVMAT (GENERAL SPARSE MATRIX)-------------------------------------
C
C     SUBROUTINE SVMAT(W,U,SUM)
      SUBROUTINE SSVMAT(W,U,SUM)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  W(1),U(1),A(1),SUM,TEMP
      INTEGER IROW(1),ICOL(1)
C-----------------------------------------------------------------------
C     SUBROUTINE TO COMPUTE U = A*W - SUM*U WHERE A IS A
C     GENERAL, SPARSE M X N MATRIX WITH M >= N.
C
C     ASSUMES THAT THE MATRIX IS STORED IN THE SPARSE FORMAT
C     GIVEN IN THE CORRESPONDING USPEC SUBROUTINE.
C-----------------------------------------------------------------------
      DO 10 I = 1,M
   10 U(I) = -SUM*U(I)
C
C  MAIN LOOP.  PROCESSING PROCEEDS COL BY COL.  JFIRST AND JLAST ARE
C  POINTERS TO THE FIRST AND LAST NONZEROS IN COLUMN J.
C
      JLAST = 0
```

```
      DO 30 J = 1,N                                                     LSM03190
      JFIRST = JLAST + 1                                                LSM03200
      JLAST = JLAST + ICOL(J)                                           LSM03210
      TEMP = W(J)                                                       LSM03220
C                                                                       LSM03230
      DO 20 K = JFIRST,JLAST                                            LSM03240
      IK = IROW(K)                                                      LSM03250
   20 U(IK) = U(IK) + A(K)*TEMP                                         LSM03260
C                                                                       LSM03270
   30 CONTINUE                                                          LSM03280
C                                                                       LSM03290
      RETURN                                                            LSM03300
C                                                                       LSM03310
C-----------------------------------------------------------------------LSM03320
      ENTRY SMATVE(A,ICOL,IROW,M,N)                                     LSM03330
C-----------------------------------------------------------------------LSM03340
C                                                                       LSM03350
C----END OF SVMAT FOR GENERAL SPARSE MATRICES---------------------------LSM03360
      RETURN                                                            LSM03370
      END                                                               LSM03380
C                                                                       LSM03390
C-----ROUTINES FOR 'DIAGONAL' TEST MATRICES-----------------------------LSM03400
C     DMATV,DMTRAN,DIAGSP SUBROUTINES ARE FOR RECTANGULAR DIAGONAL      LSM03410
C     TEST MATRICES.                                                    LSM03420
C                                                                       LSM03430
C-----START OF USPEC FOR 'DIAGONAL' TEST MATRIX-------------------------LSM03440
C                                                                       LSM03450
      SUBROUTINE USPEC(M,N,MATNO)                                       LSM03460
C     SUBROUTINE DIAGSP(M,N,MATNO)                                      LSM03470
C                                                                       LSM03480
C     DEFINES 'DIAGONAL' MATRIX OF FOLLOWING FORM                       LSM03490
C                                                                       LSM03500
C                         -----         -----                           LSM03510
C                        | 0       0       D |                          LSM03520
C              A =       | 0       0       0 |                          LSM03530
C                        |D-TRANS  0       0 |                          LSM03540
C                         -----         -----                           LSM03550
C                                                                       LSM03560
C     WHERE D IS DIAGONAL MATRIX OF ORDER N, AND IN THE                 LSM03570
C     MIDDLE THERE ARE (M-N) ROWS OF ZEROES.                            LSM03580
C     CALLS ENTRY TO MATRIX-VECTOR MULTIPLY SUBROUTINE TO PASS          LSM03590
C     STORAGE LOCATION OF THE D-ARRAY AND THE ORDERS M AND N.           LSM03600
C                                                                       LSM03610
C     NOTE:  ASSOCIATED MATRIX-VECTOR SUBROUTINES ASSUME THAT           LSM03620
C            M >= N.                                                    LSM03630
C-----------------------------------------------------------------------LSM03640
      DOUBLE PRECISION  D(1000), SPACE                                  LSM03650
      REAL  EXPLAN(20)                                                  LSM03660
C-----------------------------------------------------------------------LSM03670
C                                                                       LSM03680
      READ(8,10) EXPLAN                                                 LSM03690
   10 FORMAT(20A4)                                                      LSM03700
      READ(8,*) MOLD,NOLD,NUNIF,SPACE,D(1)                              LSM03710
C                                                                       LSM03720
      IF(N.NE.NOLD.OR.M.NE.MOLD) GO TO 80                               LSM03730
```

```
C     COMPUTE THE UNIFORM PORTION OF THE SPECTRUM
      DO 20 J=2,NUNIF
   20 D(J) = D(1) - DFLOAT(J-1)*SPACE
      NUNIF1=NUNIF + 1
      READ(8,10)  EXPLAN
      DO 30 J=NUNIF1,N
   30 READ(8,*) D(J)
      NNUNIF = NOLD - NUNIF
      WRITE(6,40) NOLD,SPACE,NNUNIF,D(1)
   40 FORMAT(/' DIAGONAL TEST MATRIX, SIZE = ',I4/' MOST ENTRIES ARE ',
     1E10.3,' UNITS APART.',I3,' ENTRIES'/' ARE IRREGULARLY SPACED. FIRS
     1T ENTRY IS ',E10.3/)
      NB = NUNIF - 2
C
C
C     PRINT OUT DIAGONAL PORTION OF A-MATRIX
      WRITE(6,50) (D(I), I=1,10 )
      WRITE(6,60) (D(I), I = NB,N)
      MNDIF = MOLD - NOLD
      IF(MNDIF.NE.0) WRITE(6,70) MNDIF
   50 FORMAT(/' SINGULAR VALUE LANCZOS TEST, 1ST 10 ENTRIES OF DIAGONAL
     1A-MATRIX = '/(3E22.14))
   60 FORMAT(/' MIDDLE UNIFORM PORTION OF MATRIX IS NOT PRINTED OUT'/
     1' END OF UNIFORM PLUS NONUNIFORM SECTION = '/(3E22.14))
   70 FORMAT(I4,' ZERO ROWS ARE ADDED TO THE DIAGONAL TO MAKE IT RECTANG
     1ULAR'/)
C
C     DIAGONAL GENERATION COMPLETE
C
C-----------------------------------------------------------------------
C     CALL ENTRY TO MATRIX-VECTOR MULTIPLY SUBROUTINES  TO PASS
C     STORAGE LOCATION OF D-ARRAY AND ORDER OF A-MATRIX.
      CALL DMATVE(D,M,N)
      CALL DMTRAE(D,M,N)
C-----------------------------------------------------------------------
C
      RETURN
   80 WRITE(6,90) MOLD,NOLD,M,N
   90 FORMAT(' PROGRAM TERMINATES MOLD=',I5,' N.E. M=',I5,' OR NOLD=',
     1I5,' N.E. N=',I5)
C-----END OF USPEC SUBROUTINE FOR 'DIAGONAL' TEST MATRICES--------------
      STOP
      END
C
C-----DSVMAT ('DIAGONAL' TEST MATRICES)---------------------------------
C
C     SUBROUTINE DSVMAT(Z,W,SUM)
      SUBROUTINE SVMAT(Z,W,SUM)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  A(1),Z(1),W(1),SUM
C-----------------------------------------------------------------------
C
C     COMPUTES W = A*Z - SUM*W .  ASSUMES THAT M >= N.
      DO 10 I = 1,N
```

```
   10 W(I) = A(I)*Z(I) - SUM *W(I)
      IF(M.EQ.N) RETURN
      N1 = N+1
      DO 20 I = N1,M
   20 W(I) = -SUM*W(I)
      RETURN
C
C-----------------------------------------------------------------------
C     STORAGE LOCATIONS OF THE A-ARRAY
C     AND THE ORDER OF THE A-MATRIX ARE PASSED TO THE MATVEC SUBROUTINE.
C     ENTRY MATVE(A,M,N)
      ENTRY DMATVE(A,M,N)
C-----------------------------------------------------------------------
C
C-----END OF MATRIX -VECTOR MULTIPLY 'DIAGONAL' TEST PROBLEMS-----------
      RETURN
      END
C
C-----MATRIX-VECTOR MULTIPLY FOR 'DIAGONAL' TEST MATRICES---------------
C
      SUBROUTINE STRAN(Z,W,SUM)
C     SUBROUTINE DSTRAN(Z,W,SUM)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  A(1),Z(1),W(1),SUM
C-----------------------------------------------------------------------
C
C     COMPUTES W = A-TRANSPOSE*Z - SUM*W .  ASSUMES M >= N.
      DO 10 I = 1,N
   10 W(I) = A(I)*Z(I)- SUM*W(I)
      RETURN
C
C-----------------------------------------------------------------------
C     STORAGE LOCATIONS OF THE A-ARRAY AND THE ORDER
C     OF THE A-MATRIX ARE OBTAINED FROM USPEC SUBROUTINE.
C     ENTRY MTRANE(A,M,N)
      ENTRY DMTRAE(A,M,N)
C-----------------------------------------------------------------------
C
C-----END OF SPARSE SYMMETRIC MATRIX-VECTOR MULTIPLY--------------------
      RETURN
      END
```

6.6 LSSUB: Other Subroutines used by the Codes in Chapter 6

```
C-----LSSUB-------(SINGULAR VALUES AND VECTORS)------------------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them are used in other scientific or
C  engineering research works the names of the authors of these codes
C  and appropriate references to their written work are to be
C  incorporated in the derivative works.
C
C  This header is not to be removed from these codes.
C
C        REFERENCE: Cullum and Willoughby, Chapter 5
C        Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C        Applied Mathematics, 2002. SIAM Publications,
C        Philadelphia, PA. USA
C
C
C     ACCORDING TO PFORT THESE SUBROUTINES ARE PORTABLE
C
C
C     SUBROUTINES    BISEC, INVERR, TNORM, LUMP, ISOEV, PRTEST, AND
C                    INVERM ARE USED WITH LANCZOS SINGULAR VALUE
C                    PROGRAM LSVAL.  STURMI, INVERM, LBISEC, TNORM
C                    ARE USED WITH THE LANCZOS SINGULAR VECTOR
C                    PROGRAM LSVEC.
C
C
C-----COMPUTE T-EIGENVALUES BY BISECTION--------------------------------
C
      SUBROUTINE BISEC(BETA,BETA2,VB,VS,LBD,UBD,EPS,TTOL,MP,
     1 NINT,MEV,NDIS,IC,IWRITE)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  BETA(1),BETA2(1),VB(1),VS(1)
      DOUBLE PRECISION  LBD(1),UBD(1),EPS,EPT,EP0,EP1,TEMP,TTOL
      DOUBLE PRECISION  ZERO,ONE,HALF,YU,YV,LB,UB,XL,XU,X1,X0,XS,BETAM
      INTEGER  MP(1),IDEF(10)
      DOUBLE PRECISION  DABS, DSQRT, DMAX1, DMIN1, DFLOAT
C-----------------------------------------------------------------------
C     COMPUTES EIGENVALUES OF T(1,MEV) BY LOOPING INTERNALLY ON THE
C     USER-SPECIFIED INTERVALS,  (LB(J),UB(J)), J = 1,NINT.  INTERVALS
```

```
C     ARE TREATED AS OPEN ON THE LEFT AND CLOSED ON THE RIGHT.          LSS00440
C     THE BISEC SUBROUTINE SIMULTANEOUSLY LABELS SPURIOUS T-EIGENVALUES LSS00450
C     AND DETERMINES THE T-MULTIPLICITIES OF EACH GOOD T-EIGENVALUE.    LSS00460
C     SPURIOUS T-EIGENVALUES ARE LABELLED BY A T-MULTIPLICITY = 0.      LSS00470
C     ANY T-EIGENVALUE WITH A T-MULTIPLICITY >= 1 IS 'GOOD'.            LSS00480
C                                                                       LSS00490
C     IF IWRITE = 0 THEN MOST OF THE WRITES TO FILE 6 ARE NOT           LSS00500
C     ACTIVATED.                                                        LSS00510
C                                                                       LSS00520
C     NOTE THAT PROGRAM ASSUMES THAT NO MORE THAN MMAX/2 T-EIGENVALUES  LSS00530
C     OF T(1,MEV) ARE TO BE COMPUTED IN ANY ONE OF THE SUBINTERVALS     LSS00540
C     CONSIDERED, WHERE MMAX = DIMENSION OF VB SPECIFIED BY THE USER    LSS00550
C     IN THE MAIN PROGRAM LEVAL.                                        LSS00560
C                                                                       LSS00570
C     ON ENTRY                                                          LSS00580
C     BETA2(J) IS SET = BETA(J)*BETA(J).  THE STORAGE FOR BETA2 COULD   LSS00590
C     BE ELIMINATED BY RECOMPUTING THE BETA(J)**2 FOR EACH STURM        LSS00600
C     SEQUENCE.                                                         LSS00610
C                                                                       LSS00620
C     EPS = 2*MACHEP =  4.4 * 10**-16 ON IBM 3081.                      LSS00630
C     TTOL = EPS*TKMAX WHERE                                            LSS00640
C     TKMAX = MAX(BETA(K), K=1,KMAX)                                    LSS00650
C                                                                       LSS00660
C     ON EXIT                                                           LSS00670
C     NDIS = TOTAL NUMBER OF COMPUTED DISTINCT T-EIGENVALUES OF         LSS00680
C            T(1,MEV) ON THE UNION OF THE (LB,UB) INTERVALS.            LSS00690
C     VS = COMPUTED DISTINCT T-EIGENVALUES OF T(1,MEV) IN ALGEBRAICALLY-LSS00700
C          INCREASING ORDER                                             LSS00710
C     MP = CORRESPONDING T-MULTIPLICITIES OF THESE T-EIGENVALUES        LSS00720
C     MP(I) = (0,1,MI), MI>1, I=1,NDIS  MEANS:                          LSS00730
C        (0)  V(I) IS SPURIOUS                                          LSS00740
C        (1)  V(I) IS T-ISOLATED AND GOOD                               LSS00750
C        (MI) V(I) IS T-MULTIPLE AND HENCE A CONVERGED GOOD T-EIGENVALUELSS00760
C     IC = TOTAL NUMBER OF STURMS USED                                  LSS00770
C                                                                       LSS00780
C     DEFAULTS                                                          LSS00790
C     ISKIP = 0 INITIALLY. IF DEFAULT OCCURS ON J-TH SUB-INTERVAL, SET  LSS00800
C             ISKIP=ISKIP+1 AND IDEF(ISKIP) = J                         LSS00810
C             DEFAULTS OCCUR IF THERE ARE NO T-EIGENVALUES  IN THE      LSS00820
C             SUBINTERVAL SPECIFIED OR IF THE NUMBER                    LSS00830
C             OF STURMS SEQUENCES REQUIRED EXCEEDS MXSTUR.              LSS00840
C             WHEN A DEFAULT OCCURS THE PROGRAM                         LSS00850
C             SKIPS THE INTERVAL INVOLVED AND GOES ON TO THE NEXT       LSS00860
C             INTERVAL.                                                 LSS00870
C                                                                       LSS00880
C-----------------------------------------------------------------------LSS00890
C     SPECIFY PARAMETERS                                                LSS00900
      ZERO = 0.0D0                                                      LSS00910
      ONE  = 1.0D0                                                      LSS00920
      HALF = 0.5D0                                                      LSS00930
      MXSTUR = IC                                                       LSS00940
      NDIS = 0                                                          LSS00950
      IC = 0                                                            LSS00960
      ISKIP = 0                                                         LSS00970
      MP1 = MEV+1                                                       LSS00980
```

```
C     SAVE THEN SET BETA(MEV+1) = 0. GENERATE BETA**2
      BETAM = BETA(MP1)
      BETA(MP1) = ZERO
C
      DO 10 I = 1,MP1
   10 BETA2(I) = BETA(I)*BETA(I)
C
C     EP0 IS USED IN T-MULTIPLICITY AND SPURIOUS TESTS
C     EP1 AND EPS ARE USED IN THE BISEC CONVERGENCE TEST
C
      TEMP = DFLOAT(MEV+1000)
      EP0  = TEMP*TTOL
      EP1  = DSQRT(TEMP)*TTOL
C
      WRITE(6,20)MEV,NINT
   20 FORMAT(/' BISEC CALCULATION'/' ORDER OF T IS',I6/
     1' NUMBER OF INTERVALS IS',I6/)
C
      WRITE(6,30) EP0,EP1
   30 FORMAT(/' MULTOL, TOLERANCE USED IN T-MULTIPLICITY AND SPURIOUS TE
     1STS = ',E10.3/' BISTOL, TOLERANCE USED IN BISEC CONVERGENCE TEST =
     1',E10.3/)
C
C     LOOP ON THE NINT INTERVALS  (LB(J),UB(J)), J=1,NINT
      DO 430 JIND = 1,NINT
      LB = LBD(JIND)
      UB = UBD(JIND)
C
      WRITE(6,40)JIND,LB,UB
   40 FORMAT(//1X,'BISEC INTERVAL NO',2X,'LOWER BOUND',2X,'UPPER BOUND'/
     1I18,2E13.5/)
C
C     INITIALIZATION AND PARAMETER SPECIFICATION
C     ICT IS TOTAL STURM COUNT ON (LB,UB)
C
      NA = 0
      MD = 0
      NG = 0
      ICT = 0
C
C     START OF T-EIGENVALUE CALCULATIONS
      X1 = UB
      ISTURM = 1
      GO TO 330
C     FORWARD STURM CALCULATION TO DETERMINE NA = NO. T-EIGENVALUES > UB
   50 NA = NEV
C
      X1 = LB
      ISTURM = 2
      GO TO 330
C     FORWARD STURM CALC TO DETERMINE MT = NO. T-EIGENVALUES ON (LB,UB)
   60 CONTINUE
      MT=NEV
      ICT = ICT +2
C
```

```
      WRITE(6,70)MT,NA
   70 FORMAT(/2I6,' = NO. TMEV ON (LB,UB) AND NO. .GT. UB'/)
C
C     DEFAULT TEST: IS ESTIMATED NUMBER OF STURMS > MXSTUR?
      IEST = 30*MT
      IF (IEST.LT.MXSTUR) GO TO 90
C
      WRITE(6,80)
   80 FORMAT(//' ESTIMATED NUMBER OF STURMS REQUIRED EXCEEDS USER LIMIT'
     1/' SKIP THIS SUBINTERVAL')
      GO TO 110
C
   90 CONTINUE
C
      IF (MT.GE.1) GO TO 120
C
      WRITE(6,100)
  100 FORMAT(//' THERE ARE NO T-EIGENVALUES ON THIS INTERVAL)'/)
C
  110 ISKIP = ISKIP+1
      IDEF(ISKIP) = JIND
      GO TO 430
C
C     REGULAR CASE.
  120 CONTINUE
C
      IF (IWRITE.NE.0) WRITE(6,130)
  130 FORMAT(/' DISTINCT T-EIGENVALUES COMPUTED USING BISEC'/
     1 13X,'T-EIGENVALUE',2X,'TMULT',3X,'MD',4X,'NG')
C
C     SET UP INITIAL UPPER AND LOWER BOUNDS FOR T-EIGENVALUES
      DO 140 I=1,MT
      VB(I) = LB
      MTI = MT + I
  140 VB(MTI) = UB
C
C     CALCULATE T-EIGENVALUES FROM LB UP TO UB  K = MT,...,1
C     MAIN LOOP FOR FINDING KTH T-EIGENVALUE
C
      K = MT
  150 CONTINUE
      IC0 = 0
      XL = VB(K)
      MTK = MT+K
      XU = VB(MTK)
C
      ISTURM = 3
      X1 = XU
      IC0 = IC0 + 1
      GO TO 330
C     FORWARD STURM CALCULATION AT XU
  160 NU=NEV
C
C     BISECTION LOOP FOR KTH T-EIGENVALUE. TEST X1=MIDPOINT OF (XL,XU)
      ISTURM = 4
```

```
  170 CONTINUE                                                          LSS02090
      X1 = (XL+XU)*HALF                                                 LSS02100
      XS = DABS(XL)+DABS(XU)                                            LSS02110
      X0 = XU-XL                                                        LSS02120
      EPT = EPS*XS+EP1                                                  LSS02130
C                                                                       LSS02140
C     EPT IS CONVERGENCE TOLERANCE FOR KTH T-EIGENVALUE                 LSS02150
C                                                                       LSS02160
      IF (X0.LE.EPT) GO TO 230                                          LSS02170
C                                                                       LSS02180
C     T-EIGENVALUE HAS NOT YET CONVERGED                                LSS02190
C                                                                       LSS02200
      IC0 = IC0 + 1                                                     LSS02210
      GO TO 330                                                         LSS02220
C     FORWARD STURM CALCULATION AT CURRENT T-EIGENVALUE APPROXIMATION.  LSS02230
  180 CONTINUE                                                          LSS02240
C                                                                       LSS02250
C     UPDATE T-EIGENVALUE INTERVAL (XL,XU)                              LSS02260
C                                                                       LSS02270
      IF (NEV.LT.K) GO TO 190                                           LSS02280
C                                                                       LSS02290
C     NUMBER OF T-EIGENVALUES NEV = K                                   LSS02300
      XL = X1                                                           LSS02310
      GO TO 170                                                         LSS02320
  190 CONTINUE                                                          LSS02330
C     NUMBER OF T-EIGENVALUES NEV<K                                     LSS02340
      XU = X1                                                           LSS02350
      NU = NEV                                                          LSS02360
C                                                                       LSS02370
C     UPDATE OF T-EIGENVALUE BOUNDS                                     LSS02380
C                                                                       LSS02390
      IF (NEV.EQ.0) GO TO 210                                           LSS02400
C                                                                       LSS02410
      DO 200 I = 1,NEV                                                  LSS02420
  200 VB(I) = DMAX1(X1,VB(I))                                           LSS02430
C                                                                       LSS02440
  210 NEV1 = NEV+1                                                      LSS02450
C                                                                       LSS02460
      DO 220 II = NEV1,K                                                LSS02470
      I = MT+II                                                         LSS02480
  220 VB(I) = DMIN1(X1,VB(I))                                           LSS02490
C                                                                       LSS02500
      GO TO 170                                                         LSS02510
C                                                                       LSS02520
C     END (XL,XU) BISECTION LOOP FOR KTH T-EIGENVALUE ON (LB,UB)        LSS02530
C     TEST FOR T-MULTIPLICITY AND IF SIMPLE THEN TEST FOR SPURIOUSNESS  LSS02540
C                                                                       LSS02550
  230 CONTINUE                                                          LSS02560
      NDIS = NDIS+1                                                     LSS02570
      MD = MD+1                                                         LSS02580
      VS(NDIS) = X1                                                     LSS02590
C                                                                       LSS02600
      JSTURM = 1                                                        LSS02610
      X1 = XL-EP0                                                       LSS02620
      GO TO 370                                                         LSS02630
```

```
C     BACKWARD STURM CALCULATION
  240 KL = KEV
      JL = JEV
C
      JSTURM = 2
      IC0 = IC0 + 2
      X1 = XU+EP0
      GO TO 370
C     BACKWARD STURM CALCULATION
  250 JU = JEV
      KU = KEV
C
C     FOR T(1,MEV)
C     NU - KU = NO. T-EIGENVALUES ON (XU, XU + EP0)
C     KL - KU = NO. T-EIGENVALUES ON (XL - EP0, XU + EP0)
C
C     FOR T(2,MEV)
C     JL -JU = NO. T-EIGENVALUES ON (XL - EP0, XU + EP0)
C
C     IS THIS A SIMPLE T-EIGENVALUE?
C
      IF (KL-KU-1.EQ.0) GO TO 290
C
C     VS(NDIS) = KTH-T-EIGENVALUE OF (LB,UB) IS T-MULTIPLE AND HENCE
C     GOOD
      IF (KU.EQ.NU) GO TO 280
C     CONTINUE TO CHECK FOR T-MULTIPLICITY
  260 CONTINUE
      ISTURM = 5
      X1 = X1+EP0
      IC0 = IC0 + 1
      GO TO 330
C     FORWARD STURM CALCULATION
  270 KNE = KU-NEV
      KU = NEV
      IF (KNE.NE.0) GO TO 260
C     SPECIFY T-MULTIPLICITY = MP(NDIS)
  280 MPEV = KL-KU
      KNEW = KU
      GO TO 300
C     END T-MULTIPLE CASE
C
C     T-EIGENVALUE IS SIMPLE   CHECK IF IT IS SPURIOUS
  290 CONTINUE
      MPEV = 1
      IF (JU.LT.JL) MPEV=0
      KNEW = K-1
C
C     X1 >= XU+EP0
C     SPURIOUS TEST AND SIMPLE CASE COMPLETED
C     START OF NEXT T-EIGENVALUE COMPUTATION
C
  300 K = KNEW
      MP(NDIS) = MPEV
      IF (MPEV.GE.1) NG = NG + 1
```

```
C
      IF (IWRITE.NE.0) WRITE(6,310) VS(NDIS),MPEV,MD,NG
  310 FORMAT(E25.16,3I6)
C
C     UPDATE STURM COUNT. IC0 = STURM COUNT FOR KTH T-EIGENVALUE
      ICT = ICT + IC0
C
C     EXIT TEST FOR K DO LOOP
C
      IF (K.LE.0) GO TO 410
C
C     UPDATE LOWER BOUNDS
      DO 320 I=1,KNEW
  320 VB(I) = DMAX1(X1,VB(I))
C
      GO TO 150
C     END OF BISECTION LOOP FOR KTH EIGENVALUE
C
C     FORWARD STURM CALCULATION
  330 NEV = -NA
      YU  = ONE
C
      DO 360 I = 1,MEV
      IF (YU.NE.ZERO) GO TO 340
      YV = BETA(I)/EPS
      GO TO 350
  340 YV = BETA2(I)/YU
  350 YU = X1 - YV
      IF (YU.GE.ZERO) GO TO 360
      NEV = NEV + 1
  360 CONTINUE
C     NEV = NUMBER OF T-EIGENVALUES ON (X1,UB)
C
      GO TO (50,60,160,180,270), ISTURM
C
C     BACKWARD STURM CALCULATION FOR T(1,MEV) AND T(2,MEV)
  370 KEV = -NA
      YU  = ONE
C
      DO 400 II = 1,MEV
      I = MP1-II
      IF (YU.NE.ZERO) GO TO 380
      YV = BETA(I+1)/EPS
      GO TO 390
  380 YV = BETA2(I+1)/YU
  390 YU = X1-YV
      JEV = 0
      IF (YU.GE.ZERO) GO TO 400
      KEV = KEV+1
      JEV = 1
  400 CONTINUE
      JEV = KEV-JEV
C
      GO TO (240,250), JSTURM
C
```

```
C     KEV = -NA + (NUMBER OF T(1,MEV) T-EIGENVALUES) > X1
C     JEV = -NA + (NUMBER OF T(2,MEV) T-EIGENVALUES) > X1
C
C     SET PARAMETERS FOR NEXT INTERVAL
  410 CONTINUE
      IC = ICT+IC
      MXSTUR = MXSTUR-ICT
C
      WRITE(6,420) JIND,NG,MD,ICT
  420 FORMAT(/' T-EIGENVALUE CALCULATION ON INTERVAL',I6,'  IS COMPLETE'
     1 /3X,'NO. GOOD',3X,'NO. DISTINCT',4X,'STURMS'/I10,I13,I10)
C
  430 CONTINUE
C
C     END LOOP ON THE SUBINTERVALS (LB(J),UB(J)), J=1,NINT
C     ISKIP OUTPUT
C
      IF (ISKIP.GT.0) WRITE(6,440)ISKIP
  440 FORMAT(' BISEC DEFAULTED ON',I3,3X,'INTERVALS'/
     1 ' DEFAULTS OCCUR IF AN INTERVAL HAS NO T-EIGENVALUES'/
     2 ' OR THE STURM ESTIMATE EXCEEDS THE USER-SPECIFIED LIMIT'/)
C
      IF (ISKIP.GT.0) WRITE(6,450)(IDEF(I), I=1,ISKIP)
  450 FORMAT(' BISEC DEFAULTED ON INTERVALS'/(10I8))
C
C     RESET BETA AT I = MP1
      BETA(MP1) = BETAM
C-----END OF BISEC------------------------------------------------------
      RETURN
      END
C
C-----INVERSE ITERATION ON T(1,MEV)-------------------------------------
C
      SUBROUTINE INVERR(BETA,V1,V2,VS,EPS,G,MP,MEV,MMB,NDIS,NISO,
     1 NM,IKL,IT,IWRITE)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION BETA(1),V1(1),V2(1),VS(1)
      DOUBLE PRECISION X1,U,Z,EST,TEMP,T0,T1,RATIO,SUM,XU,NORM,TSUM
      DOUBLE PRECISION  BETAM,EPS,EPS3,EPS4,ZERO,ONE
      REAL G(1)
      INTEGER  MP(1)
C-----------------------------------------------------------------------
      DOUBLE PRECISION FINPRO
      REAL  ABS
      DOUBLE PRECISION  DABS, DMIN1, DSQRT, DFLOAT
C-----------------------------------------------------------------------
C     COMPUTES ERROR ESTIMATES FOR COMPUTED ISOLATED GOOD T-EIGENVALUES
C     IN VS AND WRITES THESE T-EIGENVALUES AND ESTIMATES TO FILE 4.
C     BY DEFINITION A GOOD T-EIGENVALUE IS ISOLATED IF ITS
C     CLOSEST NEIGHBOR IS ALSO GOOD, OR IF ONE OF ITS NEIGHBORS IS
C     SPURIOUS BUT THAT NEIGHBOR IS FAR ENOUGH AWAY.  SO
C     IN PARTICULAR, WE COMPUTE ESTIMATES FOR GOOD T-EIGENVALUES
C     THAT ARE IN CLUSTERS OF GOOD T-EIGENVALUES.
C
```

```
C     USES INVERSE ITERATION ON T(1,MEV) SOLVING THE EQUATION           LSS04290
C     (T - X1*I)V2 = RIGHT-HAND SIDE (RANDOMLY-GENERATED)               LSS04300
C     FOR EACH SUCH GOOD T-EIGENVALUE X1.                               LSS04310
C                                                                       LSS04320
C     PROGRAM REFACTORS T-X1*I ON EACH ITERATION OF INVERSE ITERATION.  LSS04330
C     TYPICALLY ONLY ONE ITERATION IS NEEDED PER T-EIGENVALUE X1.       LSS04340
C                                                                       LSS04350
C     POSSIBLE STORAGE COMPRESSION                                      LSS04360
C     G STORAGE COULD BE ELIMINATED BY REGENERATING THE RANDOM          LSS04370
C     RIGHT-HAND SIDE ON EACH ITERATION AND PRINTING OUT THE            LSS04380
C     ERROR ESTIMATES AS THEY ARE GENERATED.                            LSS04390
C                                                                       LSS04400
C     ON ENTRY AND EXIT                                                 LSS04410
C     MEV = ORDER OF T                                                  LSS04420
C     BETA CONTAINS THE NONZERO ENTRIES OF THE T-MATRIX                 LSS04430
C     VS = COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)                    LSS04440
C     MP = T-MULTIPLICITY OF EACH EIGENVALUE IN VS. MP(I) = -1 MEANS    LSS04450
C          VS(I) IS A GOOD T-EIGENVALUE BUT THAT IT IS SITTING CLOSE TO LSS04460
C          A SPURIOUS T-EIGENVALUE.  MP(I) = 0 MEANS VS(I) IS SPURIOUS. LSS04470
C          ESTIMATES ARE COMPUTED ONLY FOR THOSE T-EIGENVALUES          LSS04480
C          WITH MP(I) = 1. FLAGGING WAS DONE IN SUBROUTINE ISOEV        LSS04490
C          PRIOR TO ENTERING INVERR.                                    LSS04500
C     NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES CONTAINED IN VS      LSS04510
C     NDIS =  NUMBER OF DISTINCT T-EIGENVALUES IN VS                    LSS04520
C     IKL = SEED FOR RANDOM NUMBER GENERATOR                            LSS04530
C     EPS = 2. * MACHINE EPSILON                                        LSS04540
C                                                                       LSS04550
C     IN PROGRAM:                                                       LSS04560
C     ITER = MAXIMUM NUMBER OF INVERSE ITERATION STEPS ALLOWED FOR EACH LSS04570
C            X1.  ITER = IT ON ENTRY.                                   LSS04580
C     G = ARRAY OF DIMENSION AT LEAST MEV + NISO.  USED TO STORE        LSS04590
C         RANDOMLY-GENERATED RIGHT-HAND SIDE.  THIS IS NOT              LSS04600
C         REGENERATED FOR EACH X1. G IS ALSO USED TO STORE ERROR        LSS04610
C         ESTIMATES AS THEY ARE COMPUTED FOR LATER PRINTOUT.            LSS04620
C     V1,V2 = WORK SPACES USED IN THE FACTORIZATION OF T(1,MEV).        LSS04630
C     AT THE END OF THE INVERSE ITERATION COMPUTATION FOR X1, V2        LSS04640
C     CONTAINS THE UNIT EIGENVECTOR OF T(1,MEV) CORRESPONDING TO X1.    LSS04650
C     V1 AND V2 MUST BE OF DIMENSION AT LEAST MEV.                      LSS04660
C                                                                       LSS04670
C     ON EXIT                                                           LSS04680
C     G(J) = MINIMUM GAP IN T(1,MEV) FOR EACH VS(J), J=1,NDIS           LSS04690
C     G(MEV+I) = BETAM*|V2(MEV)| = ERROR ESTIMATE FOR ISOLATED GOOD     LSS04700
C              T-EIGENVALUES, WHERE I = 1,NISO  AND  BETAM = BETA(MEV+1)LSS04710
C              V2(MEV) IS LAST COMPONENT OF THE UNIT T-EIGENVECTOR OF   LSS04720
C              T(1,MEV) CORRESPONDING TO ITH ISOLATED GOOD T-EIGENVALUE.LSS04730
C                                                                       LSS04740
C     IF FOR SOME X1 IT.GT.ITER THEN THE ERROR ESTIMATE IN G IS MARKED  LSS04750
C     WITH A - SIGN.                                                    LSS04760
C                                                                       LSS04770
C     V2 = ISOLATED GOOD T-EIGENVALUES                                  LSS04780
C     V1 = MINIMAL T-GAPS FOR THE EIGENVALUES IN V2.                    LSS04790
C     THESE ARE CONSTRUCTED FOR WRITE-OUT PURPOSES ONLY AND NOT         LSS04800
C     NEEDED ELSEWHERE IN THE PROGRAM.                                  LSS04810
C-----------------------------------------------------------------------LSS04820
C                                                                       LSS04830
```

```
C     LABEL OUTPUT FILE  4
      IF (MMB.EQ.1) WRITE(4,10)
   10 FORMAT(' INVERSE ITERATION ERROR ESTIMATES'/)
C
C     FILE 6 (TERMINAL) OUTPUT OF ERROR ESTIMATES
      IF (IWRITE.NE.0.AND.NISO.NE.0) WRITE(6,20)
   20 FORMAT(/' INVERSE ITERATION ERROR ESTIMATES'/'  JISO',' JDIST',8X
     1,'GOOD T-EIGENVALUE',4X,'BETAM*UM',5X,'TMINGAP')
C
C     INITIALIZATION AND PARAMETER SPECIFICATION
      ZERO = 0.0D0
      ONE = 1.0D0
      NG = 0
      NISO = 0
      ITER = IT
      MP1 = MEV+1
      MM1 = MEV-1
      BETAM = BETA(MP1)
      BETA(MP1) = ZERO
C
C     CALCULATE SCALE AND TOLERANCES
      TSUM = ZERO
      DO 30 I = 2,MEV
   30 TSUM = TSUM + BETA(I)
C
      EPS3 = EPS*TSUM
      EPS4 = DFLOAT(MEV)*EPS3
C
C     GENERATE SCALED RANDOM RIGHT-HAND SIDE
      ILL = IKL
C
C-----------------------------------------------------------------------
      CALL GENRAN(ILL,G,MEV)
C-----------------------------------------------------------------------
C
      GSUM = ZERO
      DO 40 I = 1,MEV
   40 GSUM = GSUM+ABS(G(I))
      GSUM = EPS4/GSUM
C
      DO 50 I = 1,MEV
   50 G(I) = GSUM*G(I)
C
C     LOOP ON ISOLATED GOOD T-EIGENVALUES IN VS (MP(I) = 1) TO
C     CALCULATE CORRESPONDING UNIT T-EIGENVECTOR OF T(1,MEV)
C
      DO 180 JEV = 1,NDIS
      IF (MP(JEV).EQ.0) GO TO 180
      NG = NG + 1
      IF (MP(JEV).NE.1) GO TO 180
      IT = 1
      NISO = NISO + 1
      X1 = VS(JEV)
C
C     INITIALIZE RIGHT HAND SIDE FOR INVERSE ITERATION
```

```
      DO 60 I = 1,MEV                                                   LSS05390
   60 V2(I) = G(I)                                                      LSS05400
C                                                                       LSS05410
C     TRIANGULAR FACTORIZATION WITH NEAREST NEIGHBOR PIVOT              LSS05420
C     STRATEGY. INTERCHANGES ARE LABELLED BY SETTING BETA < 0.          LSS05430
C                                                                       LSS05440
   70 CONTINUE                                                          LSS05450
      U = -X1                                                           LSS05460
      Z = BETA(2)                                                       LSS05470
C                                                                       LSS05480
      DO 90 I = 2,MEV                                                   LSS05490
      IF (BETA(I).GT.DABS(U)) GO TO 80                                  LSS05500
C     NO INTERCHANGE                                                    LSS05510
      V1(I-1) = Z/U                                                     LSS05520
      V2(I-1) = V2(I-1)/U                                               LSS05530
      V2(I) = V2(I)-BETA(I)*V2(I-1)                                     LSS05540
      RATIO = BETA(I)/U                                                 LSS05550
      U = -X1-Z*RATIO                                                   LSS05560
      Z = BETA(I+1)                                                     LSS05570
      GO TO 90                                                          LSS05580
   80 CONTINUE                                                          LSS05590
C     INTERCHANGE CASE                                                  LSS05600
      RATIO = U/BETA(I)                                                 LSS05610
      BETA(I) = -BETA(I)                                                LSS05620
      V1(I-1) = -X1                                                     LSS05630
      U = Z-RATIO*V1(I-1)                                               LSS05640
      Z = -RATIO*BETA(I+1)                                              LSS05650
      TEMP = V2(I-1)                                                    LSS05660
      V2(I-1) = V2(I)                                                   LSS05670
      V2(I) = TEMP-RATIO*V2(I)                                          LSS05680
   90 CONTINUE                                                          LSS05690
      IF (U.EQ.ZERO) U = EPS3                                           LSS05700
C                                                                       LSS05710
C     SMALLNESS TEST AND DEFAULT VALUE FOR LAST COMPONENT               LSS05720
C     PIVOT(I-1) = |BETA(I)| FOR INTERCHANGE CASE                       LSS05730
C     (I-1,I+1) ELEMENT IN RIGHT FACTOR = BETA(I+1)                     LSS05740
C     END OF FACTORIZATION AND FORWARD SUBSTITUTION                     LSS05750
C                                                                       LSS05760
C     BACK SUBSTITUTION                                                 LSS05770
      V2(MEV) = V2(MEV)/U                                               LSS05780
      DO 110 II = 1,MM1                                                 LSS05790
      I = MEV-II                                                        LSS05800
      IF (BETA(I+1).LT.ZERO) GO TO 100                                  LSS05810
C     NO INTERCHANGE                                                    LSS05820
      V2(I) = V2(I)-V1(I)*V2(I+1)                                       LSS05830
      GO TO 110                                                         LSS05840
C     INTERCHANGE CASE                                                  LSS05850
  100 BETA(I+1) = -BETA(I+1)                                            LSS05860
      V2(I) = (V2(I)-V1(I)*V2(I+1)-BETA(I+2)*V2(I+2))/BETA(I+1)         LSS05870
  110 CONTINUE                                                          LSS05880
C                                                                       LSS05890
C     TESTS FOR CONVERGENCE OF INVERSE ITERATION                        LSS05900
C     IF SUM |V2| COMPS. LE. 1 AND IT. LE. ITER DO ANOTHER INVIT STEP   LSS05910
C                                                                       LSS05920
      NORM = DABS(V2(MEV))                                              LSS05930
```

```
      DO 120 II = 1,MM1
      I = MEV-II
  120 NORM = NORM+DABS(V2(I))
C
      IF (NORM.GE.ONE) GO TO 140
      IT = IT+1
      IF (IT.GT.ITER) GO TO 140
      XU = EPS4/NORM
C
      DO 130 I = 1,MEV
  130 V2(I) = V2(I)*XU
C
      GO TO 70
C     ANOTHER INVERSE ITERATION STEP
C
C     INVERSE ITERATION FINISHED
C     NORMALIZE COMPUTED T-EIGENVECTOR : V2 = V2/||V2||
  140 CONTINUE
      SUM = FINPRO(MEV,V2(1),1,V2(1),1)
      SUM = ONE/DSQRT(SUM)
C
      DO 150 II = 1,MEV
  150 V2(II) = SUM*V2(II)
C
C     SAVE ERROR ESTIMATE FOR LATER OUTPUT
      EST = BETAM*DABS(V2(MEV))
      IF (IT.GT.ITER) EST = -EST
      MEVPNI = MEV + NISO
      G(MEVPNI) = EST
      IF (IWRITE.EQ.0) GO TO 180
C
C     FILE 6 (TERMINAL) OUTPUT OF ERROR ESTIMATES.
      IF (JEV.EQ.1) GAP = VS(2) - VS(1)
      IF (JEV.EQ.MEV) GAP = VS(MEV) - VS(MEV-1)
      IF (JEV.EQ.MEV.OR.JEV.EQ.1) GO TO 160
      TEMP = DMIN1(VS(JEV+1)-VS(JEV),VS(JEV)-VS(JEV-1))
      GAP = TEMP
  160 CONTINUE
C
      WRITE(6,170) NISO,JEV,X1,EST,GAP
  170 FORMAT(2I6,E25.16,2E12.3)
C
  180 CONTINUE
C
C     END ERROR ESTIMATE LOOP ON ISOLATED GOOD T-EIGENVALUES.
C     GENERATE DISTINCT MINGAPS FOR T(1,MEV).  THIS IS USEFUL AS AN
C     INDICATOR OF THE GOODNESS OF THE INVERSE ITERATION ESTIMATES.
C     TRANSFER ISOLATED GOOD T-EIGENVALUES AND CORRESPONDING TMINGAPS
C     TO V2 AND V1 FOR OUTPUT PURPOSES ONLY.
C
      NM1 = NDIS - 1
      G(NDIS) = VS(NM1)-VS(NDIS)
      G(1) = VS(2)-VS(1)
C
      DO 190 J = 2,NM1
```

```
      T0 = VS(J)-VS(J-1)
      T1 = VS(J+1)-VS(J)
      G(J) = T1
      IF (T0.LT.T1) G(J)=-T0
  190 CONTINUE
      ISO = 0
      DO 200 J = 1,NDIS
      IF (MP(J).NE.1) GO TO 200
      ISO = ISO+1
      V1(ISO) = G(J)
      V2(ISO) = VS(J)
  200 CONTINUE
C
      IF(NISO.EQ.0) GO TO 250
C
C     ERROR ESTIMATES ARE WRITTEN TO FILE 4
      WRITE(4,210)MEV,NDIS,NG,NISO,NM,IKL,ITER,BETAM
  210 FORMAT(1X,'TSIZE',2X,'NDIS',1X,'NGOOD',2X,'NISO',3X,'M+N'/5I6/
     1 4X,'RHSEED',2X,'MXINIT',5X,'BETAM'/I10,I8,E10.3/
     2 2X,'GOODEVNO',8X,'GOOD T-EIGENVALUE',6X,'BETAM*UM',7X,'TMINGAP')
C
      ISPUR = 0
      I = 0
      DO 240 J = 1,NDIS
      IF(MP(J).NE.0) GO TO 220
      ISPUR = ISPUR + 1
      GO TO 240
  220 IF(MP(J).NE.1) GO TO 240
      I = I + 1
      MEVI = MEV + I
      IGOOD = J - ISPUR
      WRITE(4,230) IGOOD,V2(I),G(MEVI),V1(I)
  230 FORMAT(I10,E25.16,2E14.3)
  240 CONTINUE
      GO TO 270
C
  250 WRITE(4,260)
  260 FORMAT(/' THERE ARE NO ISOLATED T-EIGENVALUES SO NO ERROR ESTIMATE
     1S WERE COMPUTED')
C     RESTORE BETA(MEV+1) = BETAM
  270 BETA(MP1) = BETAM
C-----END OF INVERR----------------------------------------------------
      RETURN
      END
C
C-----START OF TNORM---------------------------------------------------
C
      SUBROUTINE TNORM(BETA,BMIN,TMAX,MEV,IB)
C
C---------------------------------------------------------------------
      DOUBLE PRECISION  BETA(1)
      DOUBLE PRECISION  TMAX,BMIN,BSIZE,BTOL
      DOUBLE PRECISION  DABS, DMAX1
C---------------------------------------------------------------------
C     COMPUTE SCALING FACTOR USED IN THE T-MULTIPLICITY, SPURIOUS AND
```

```
C     PRTESTS.   CHECK RELATIVE SIZE OF THE BETA(K), K=1,MEV
C     AS A TEST ON THE LOCAL ORTHOGONALITY OF THE LANCZOS VECTORS.
C
C             TMAX = MAX (BETA(I),  I=1,MEV)
C             BMIN = MIN (BETA(I) I=2,MEV)
C             BSIZE = BMIN/TMAX
C             |IB| = INDEX OF MINIMAL(BETA)
C             IB < 0 IF BMIN/TMAX < BTOL
C-----------------------------------------------------------------------
C     SPECIFY PARAMETERS
      IB = 2
      BTOL = BMIN
      BMIN = BETA(2)
      TMAX = BETA(2)
C
      DO 20 I = 2,MEV
      IF (BETA(I).GE.BMIN) GO TO 10
      IB = I
      BMIN = BETA(I)
   10 TMAX = DMAX1(TMAX,BETA(I))
   20 CONTINUE
C
C     TEST OF LOCAL ORTHOGONALITY USING SCALED BETAS
      BSIZE = BMIN/TMAX
      IF (BSIZE.GE.BTOL) GO TO 40
C
C     DEFAULT.  BSIZE IS SMALLER THAN TOLERANCE BTOL SPECIFIED IN MAIN
C     PROGRAM.  PROGRAM TERMINATES FOR USER TO DECIDE WHAT TO DO
C     BECAUSE LOCAL ORTHOGONALITY OF THE LANCZOS VECTORS COULD BE
C     LOST.
C
      IB = -IB
      WRITE(6,30) MEV
   30 FORMAT(/' BETA TEST INDICATES POSSIBLE LOSS OF LOCAL ORTHOGONALITY
     1OVER 1ST',I6,' LANCZOS VECTORS'/)
C
   40 CONTINUE
C
      WRITE(6,50) IB
   50 FORMAT(/' MINIMUM BETA RATIO OCCURS AT',I6,' TH BETA'/)
C
      WRITE(6,60) MEV,BMIN,TMAX,BSIZE
   60 FORMAT(/1X,'TSIZE',6X,'MIN BETA',5X,'TKMAX',6X,'MIN RATIO'/
     1 I6,E14.3,E10.3,E15.3/)
C
C-----END OF TNORM------------------------------------------------------
      RETURN
      END
C
C
C-----START OF LUMP-----------------------------------------------------
C
      SUBROUTINE LUMP(V1,RELTOL,MULTOL,SCALE2,LINDEX,LOOP)
C
C-----------------------------------------------------------------------
```

```
      DOUBLE PRECISION  V1(1),SUM,RELTOL,MULTOL,THOLD,ZERO,SCALE2
      INTEGER  LINDEX(1)
      DOUBLE PRECISION  DABS, DFLOAT, DMAX1
C-----------------------------------------------------------------------
C     LINDEX(J) = T-MULTIPLICITY OF JTH DISTINCT T-EIGENVALUE
C     LOOP = NUMBER OF DISTINCT T-EIGENVALUES
C     LUMP 'COMBINES' COMPUTED 'GOOD' T-EIGENVALUES THAT ARE
C     'TOO CLOSE'.
C     VALUE FOR RELTOL IS 1.D-10.
C
C     IF IN A SET OF T-EIGENVALUES TO BE COMBINED THERE IS AN EIGENVALUE
C     WITH LINDEX=1, THEN THE VALUE OF THE COMBINED T-EIGENVALUES IS SET
C     EQUAL TO THE VALUE OF THAT T-EIGENVALUE.  NOTE THAT IF A SPURIOUS
C     T-EIGENVALUE IS TO BE 'COMBINED' WITH A GOOD T-EIGENVALUE, THEN
C     THIS IS DONE ONLY BY INCREASING THE INDEX, LINDEX, FOR THAT
C     T-EIGENVALUE. NUMERICAL VALUES OF SPURIOUS T-EIGENVALUES ARE
C     NEVER COMBINED WITH THOSE OF GOOD T-EIGENVALUES.
C-----------------------------------------------------------------------
      ZERO = 0.0D0
      NLOOP = 0
      J = 0
      ICOUNT = 1
      JI = 1
      THOLD = DMAX1(RELTOL*DABS(V1(1)),SCALE2*MULTOL)
C     THOLD = DMAX1(RELTOL*DABS(V1(1)),RELTOL)
C
   10 J = J+1
      IF (J.EQ.LOOP) GO TO 20
      SUM = DABS(V1(J)-V1(J+1))
      IF (SUM.LT.THOLD) GO TO 60
   20 JF = JI + ICOUNT - 1
      INDSUM = 0
      ISPUR = 0
C
      DO 30 KK = JI,JF
      IF (LINDEX(KK).NE.0) GO TO 30
      ISPUR = ISPUR + 1
      INDSUM = INDSUM + 1
   30 INDSUM = INDSUM + LINDEX(KK)
C
C     IF (JF-JI.GE.1) WRITE(6,40) (V1(KKK), KKK=JI,JF)
   40 FORMAT(/' LUMP LUMPS THE T-EIGENVALUES'/(4E20.13))
C
C     COMPUTE THE 'COMBINED' T-EIGENVALUE AND THE RESULTING
C     T-MULTIPLICITY
      K = JI - 1
   50 K = K+1
      IF (K.GT.JF) GO TO 70
      IF (LINDEX(K) .NE.1) GO TO 50
      NLOOP = NLOOP + 1
      V1(NLOOP) = V1(K)
      GO TO 100
   60 ICOUNT = ICOUNT + 1
      GO TO 10
C
```

```
C     ALL INDICES WERE 0 OR >1                                          LSS08140
   70 NLOOP = NLOOP + 1                                                 LSS08150
      IDIF = INDSUM - ISPUR                                             LSS08160
      IF (IDIF.EQ.0) GO TO 90                                           LSS08170
C                                                                       LSS08180
      SUM = ZERO                                                        LSS08190
      DO 80 KK = JI,JF                                                  LSS08200
   80 SUM = SUM + V1(KK) * DFLOAT(LINDEX(KK))                           LSS08210
C                                                                       LSS08220
      V1(NLOOP) = SUM/DFLOAT(IDIF)                                      LSS08230
      GO TO 100                                                         LSS08240
   90 V1(NLOOP) = V1(JI)                                                LSS08250
  100 LINDEX(NLOOP) = INDSUM                                            LSS08260
      IDIF = INDSUM - ISPUR                                             LSS08270
      IF (IDIF.EQ.0.AND.ISPUR.EQ.1) LINDEX(NLOOP) = 0                   LSS08280
      IF (J.EQ.LOOP) GO TO 110                                          LSS08290
      ICOUNT = 1                                                        LSS08300
      JI= J+1                                                           LSS08310
      THOLD = DMAX1(RELTOL*DABS(V1(JI)),SCALE2*MULTOL)                  LSS08320
C     THOLD = DMAX1(RELTOL*DABS(V1(JI)),RELTOL)                         LSS08330
      IF (JI.LT.LOOP) GO TO 10                                          LSS08340
      NLOOP = NLOOP + 1                                                 LSS08350
      V1(NLOOP)= V1(JI)                                                 LSS08360
      LINDEX(NLOOP) = LINDEX(JI)                                        LSS08370
  110 CONTINUE                                                          LSS08380
C                                                                       LSS08390
C     ON RETURN V1 CONTAINS THE DISTINCT T-EIGENVALUES                  LSS08400
C     LINDEX CONTAINS THE CORRESPONDING T-MULTIPLICITIES                LSS08410
C                                                                       LSS08420
      LOOP = NLOOP                                                      LSS08430
      RETURN                                                            LSS08440
C-----END OF LUMP-------------------------------------------------------LSS08450
      END                                                               LSS08460
C                                                                       LSS08470
C                                                                       LSS08480
C-----START OF ISOEV-----------------------------------------------------LSS08490
C                                                                       LSS08500
      SUBROUTINE ISOEV(VS,GAPTOL,MULTOL,SCALE1,G,MP,NDIS,NG,NISO)       LSS08510
C                                                                       LSS08520
C-----------------------------------------------------------------------LSS08530
      DOUBLE PRECISION  VS(1),T0,T1,MULTOL,GAPTOL,SCALE1,TEMP           LSS08540
      REAL   G(1),GAP                                                   LSS08550
      INTEGER  MP(1)                                                    LSS08560
      REAL  ABS                                                         LSS08570
      DOUBLE PRECISION  DABS, DMAX1                                     LSS08580
C-----------------------------------------------------------------------LSS08590
C     GENERATE DISTINCT TMINGAPS AND USE THEM TO LABEL THE ISOLATED     LSS08600
C     GOOD T-EIGENVALUES THAT ARE VERY CLOSE TO SPURIOUS ONES.          LSS08610
C     ERROR ESTIMATES WILL NOT BE COMPUTED FOR THESE T-EIGENVALUES.     LSS08620
C                                                                       LSS08630
C     ON ENTRY AND EXIT                                                 LSS08640
C     VS CONTAINS THE COMPUTED DISTINCT T-EIGENVALUES OF T(1,MEV)       LSS08650
C     MP CONTAINS THE CORRESPONDING T-MULTIPLICITIES                    LSS08660
C     NDIS = NUMBER OF DISTINCT T-EIGENVALUES                           LSS08670
C     GAPTOL = RELATIVE GAP TOLERANCE SET IN MAIN                       LSS08680
```

```
C
C     ON EXIT
C     G CONTAINS THE TMINGAPS.
C     G(I) < 0 MEANS MINGAP IS DUE TO LEFT GAP
C     MP(I) IS NOT CHANGED EXCEPT THAT  MP(I)=-1, IF MP(I)=1,
C     TMINGAP WAS TOO SMALL AND DUE TO A SPURIOUS T-EIGENVALUE.
C
C     IF MP(I)=-1 THAT SIMPLE GOOD T-EIGENVALUE WILL BE SKIPPED
C     IN THE SUBSEQUENT ERROR ESTIMATE COMPUTATIONS IN INVERR
C     THAT IS, WE COMPUTE ERROR ESTIMATES ONLY FOR THOSE GOOD
C     T-EIGENVALUES WITH MP(I)=1.
C-----------------------------------------------------------------------
C     CALCULATE MINGAPS FOR DISTINCT T(1,MEV) EIGENVALUES.
      NM1 = NDIS - 1
      G(NDIS) = VS(NM1)-VS(NDIS)
      G(1) = VS(2)-VS(1)
C
      DO 10 J = 2,NM1
      T0 = VS(J)-VS(J-1)
      T1 = VS(J+1)-VS(J)
      G(J) = T1
      IF (T0.LT.T1) G(J) = -T0
   10 CONTINUE
C
C     SET MP(I)=-1 FOR SIMPLE GOOD T-EIGENVALUES WHOSE MINGAPS  ARE
C     'TOO SMALL' AND DUE TO SPURIOUS T-EIGENVALUES.
C
      NISO = 0
      NG = 0
      DO 20 J = 1,NDIS
      IF (MP(J).EQ.0) GO TO 20
      NG = NG+1
      IF (MP(J).NE.1) GO TO 20
C     VS(J) IS NEXT TO SIMPLE GOOD T-EIGENVALUE
      NISO = NISO + 1
      I = J+1
      IF (G(J).LT.0.0) I = J-1
      IF (MP(I).NE.0) GO TO 20
      GAP = ABS(G(J))
      T0 = DMAX1(SCALE1*MULTOL,GAPTOL*DABS(VS(J)))
C     T0 = DMAX1(GAPTOL,GAPTOL*DABS(VS(J)))
      TEMP = T0
      IF (GAP.GT.TEMP) GO TO 20
      MP(J) = -MP(J)
      NISO = NISO-1
   20 CONTINUE
C
C-----END OF ISOEV-------------------------------------------------------
      RETURN
      END
C
C-----START OF PRTEST----------------------------------------------------
C
      SUBROUTINE PRTEST(BETA,TEIG,TKMAX,EPSM,RELTOL,SCALE3,SCALE4,
     1 TMULT,NDIST,MEV,IPROJ)
```

```
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  BETA(1),TEIG(1),SIGMA(4)
      DOUBLE PRECISION  EPSM,RELTOL,PRTOL,TKMAX,LRATIO,URATIO
      DOUBLE PRECISION  EPS,EPS1,BETAM,LBD,UBD,SIG,YU,YV,LRATS,URATS
      DOUBLE PRECISION  ZERO,ONE,TEN,BISTOL,SCALE3,SCALE4,AEV,TEMP
      INTEGER  TMULT(1),ISIGMA(4)
      DOUBLE PRECISION  DABS, DMAX1, DSQRT, DFLOAT
C-----------------------------------------------------------------------
C     AFTER CONVERGENCE HAS BEEN ESTABLISHED, SUBROUTINE PRTEST
C     TESTS COMPUTED EIGENVALUES OF T(1,MEV) THAT HAVE BEEN LABELLED
C     SPURIOUS TO DETERMINE IF ANY SINGULAR VALUES OF A HAVE BEEN
C     MISSED BY LANCZOS PROCEDURE.  A SINGULAR VALUE WHOSE
C     SINGULAR VECTOR(S) HAS A VERY SMALL PROJECTION ON THE
C     STARTING VECTOR (< SINGLE PRECISION) CAN BE MISSED BECAUSE
C     IT WILL THEN ALSO BE AN EIGENVALUE OF T(2,MEV) TO WITHIN
C     THE SQUARE OF THIS ORIGINAL PROJECTION.  HOWEVER,
C     OUR EXPERIENCE IS THAT SUCH SMALL PROJECTIONS OCCUR ONLY
C     VERY INFREQUENTLY.
C
C     THIS SUBROUTINE IS CALLED ONLY AFTER CONVERGENCE HAS BEEN
C     ESTABLISHED. ONCE CONVERGENCE HAS BEEN OBSERVED ON THE
C     OTHER SINGULAR VALUES, THEN ONE CAN EXPECT TO ALSO HAVE
C     CONVERGENCE ON ANY SUCH 'HIDDEN' SINGULAR VALUES. (IF THERE
C     ARE ANY).  PROCEDURE CONSIDERS ONLY SPURIOUS T-EIGENVALUES AND
C     ONLY THOSE SPURIOUS T-EIGENVALUES THAT ARE ISOLATED FROM GOOD
C     T-EIGENVALUES.  FOR EACH SUCH T-EIGENVALUE IT DOES 2 STURM
C     SEQUENCES AND A FEW SCALAR MULTIPLICATIONS.  UPON RETURN TO MAIN
C     PROGRAM ERROR ESTIMATES WILL BE COMPUTED FOR ANY T-EIGENVALUES
C     THAT HAVE BEEN LABELLED AS 'HIDDEN'.  SUCH T-EIGENVALUES
C     WILL BE RELABELLED AS 'GOOD' ONLY IF THESE ERROR ESTIMATES
C     ARE SUFFICIENTLY SMALL.
C-----------------------------------------------------------------------
      ZERO = 0.0D0
      ONE  = 1.0D0
      TEN  = 10.0D0
      PRTOL = 1.D-6
      TEMP = DFLOAT(MEV+1000)
      TEMP = DSQRT(TEMP)
      BISTOL = TKMAX*EPSM*TEMP
      NSIGMA = 4
      SIGMA(1) = TEN*TKMAX
C
      DO 10 J = 2,NSIGMA
   10 SIGMA(J) = TEN*SIGMA(J-1)
C
      IFIN = 0
      MF = 1
      ML = MEV
      BETAM = BETA(MF)
      BETA(MF) = ZERO
      IPROJ = 0
      J = 1
C
      IF (TMULT(1).NE.0) GO TO 110
```

```
C                                                                       LSS09790
      AEV = DABS(TEIG(1))                                               LSS09800
      TEMP = PRTOL*AEV                                                  LSS09810
      EPS1 = DMAX1(TEMP,SCALE4*BISTOL)                                  LSS09820
C     EPS1 = DMAX1(TEMP,PRTOL)                                          LSS09830
      TEMP = RELTOL*AEV                                                 LSS09840
      EPS  = DMAX1(TEMP,SCALE3*BISTOL)                                  LSS09850
C     EPS  = DMAX1(TEMP,RELTOL)                                         LSS09860
C                                                                       LSS09870
      IF (TEIG(2)-TEIG(1).LT.EPS1.AND.TMULT(2).NE.0) GO TO 110          LSS09880
C                                                                       LSS09890
   20 LBD = TEIG(J) - EPS                                               LSS09900
      UBD = TEIG(J) + EPS                                               LSS09910
      MEVL = 0                                                          LSS09920
      IL = 0                                                            LSS09930
      YU = ONE                                                          LSS09940
C                                                                       LSS09950
      DO 50 I=MF,ML                                                     LSS09960
      IF (YU.NE.ZERO) GO TO 30                                          LSS09970
      YV = BETA(I)/EPSM                                                 LSS09980
      GO TO 40                                                          LSS09990
   30 YV = BETA(I)*BETA(I)/YU                                           LSS10000
   40 YU = -LBD-YV                                                      LSS10010
      IF (YU.GE.ZERO) GO TO 50                                          LSS10020
C     MEVL INCREMENTED                                                  LSS10030
      MEVL = MEVL + 1                                                   LSS10040
      IL = I                                                            LSS10050
   50 CONTINUE                                                          LSS10060
C                                                                       LSS10070
      LRATIO = YU                                                       LSS10080
      MEV1L = MEVL                                                      LSS10090
      IF (IL.EQ.ML) MEV1L=MEVL-1                                        LSS10100
C                                                                       LSS10110
C     MEVL = NUMBER OF EVS OF T(1,MEV) WHICH ARE < LBD                  LSS10120
C     MEV1L = NUMBER OF EVS OF T(1,MEV-1) WHICH ARE < LBD               LSS10130
C     LRATIO = DET(T(1,MEV)-LBD)/DET(T(1,MEV-1)-LBD):                   LSS10140
C                                                                       LSS10150
      MEVU = 0                                                          LSS10160
      IL = 0                                                            LSS10170
      YU = ONE                                                          LSS10180
C                                                                       LSS10190
      DO 80 I=MF,ML                                                     LSS10200
      IF (YU.NE.ZERO) GO TO 60                                          LSS10210
      YV = BETA(I)/EPSM                                                 LSS10220
      GO TO 70                                                          LSS10230
   60 YV = BETA(I)*BETA(I)/YU                                           LSS10240
   70 YU = -UBD-YV                                                      LSS10250
      IF (YU.GE.ZERO) GO TO 80                                          LSS10260
C     MEVU INCREMENTED                                                  LSS10270
      MEVU = MEVU + 1                                                   LSS10280
      IL = I                                                            LSS10290
   80 CONTINUE                                                          LSS10300
C                                                                       LSS10310
      URATIO = YU                                                       LSS10320
      MEV1U = MEVU                                                      LSS10330
```

```
      IF (IL.EQ.ML) MEV1U=MEVU-1
C
C     MEVU = NUMBER OF EVS OF T(MEV) WHICH ARE < UBD
C     MEV1U = NUMBER OF EVS OF T(MEV-1) WHICH ARE < UBD
C     URATIO = DET(TM-UBD)/DET(T(M-1)-UBD): TM=T(MF,ML)
C
      NEV1 = MEV1U-MEV1L
C
      DO 90 K=1,NSIGMA
      SIG = SIGMA(K)
      LRATS = LRATIO-SIG
      URATS = URATIO-SIG
C     NOTE THE INCREMENT IS ON NUMBER OF EVALUES OF T(M-1)
      MEVLS = MEV1L
      IF (LRATS.LT.0.) MEVLS=MEV1L+1
      MEVUS = MEV1U
      IF (URATS.LT.0.) MEVUS=MEV1U+1
      ISIGMA(K) = MEVUS - MEVLS
   90 CONTINUE
C
      ICOUNT = 0
      DO 100 K=1,NSIGMA
  100 IF (ISIGMA(K).EQ.1) ICOUNT=ICOUNT + 1
C
      IF (ICOUNT.LT.2.OR.NEV1.EQ.0) GO TO 110
      TMULT(J) = -10
      IPROJ=IPROJ+1
C
  110 J=J+1
C
      IF (J.GE.NDIST) GO TO 120
      IF (TMULT(J).NE.0) GO TO 110
C
      AEV = DABS(TEIG(J))
      TEMP = PRTOL*AEV
      EPS1 = DMAX1(TEMP,SCALE4*BISTOL)
C     EPS1 = DMAX1(TEMP,PRTOL)
      TEMP = RELTOL*AEV
      EPS  = DMAX1(TEMP,SCALE3*BISTOL)
C     EPS  = DMAX1(TEMP,RELTOL)
C
      IF (TEIG(J)-TEIG(J-1).LT.EPS1.AND.TMULT(J-1).NE.0) GO TO 110
      IF (TEIG(J+1)-TEIG(J).LT.EPS1.AND.TMULT(J+1).NE.0) GO TO 110
C
      GO TO 20
C
  120 IF (IFIN.EQ.1) GO TO 130
      IF (TMULT(NDIST).NE.0) GO TO 130
C
      AEV = DABS(TEIG(NDIST))
      TEMP = PRTOL*AEV
      EPS1 = DMAX1(TEMP,SCALE4*BISTOL)
C     EPS1 = DMAX1(TEMP,PRTOL)
      TEMP = RELTOL*AEV
      EPS  = DMAX1(TEMP,SCALE3*BISTOL)
```

```
C     EPS  = DMAX1(TEMP,RELTOL)
C
      NDIST1=NDIST -1
      TEMP = TEIG(NDIST)-TEIG(NDIST1)
      IF (TEMP.LT.EPS1.AND.TMULT(NDIST1).NE.0) GO TO 130
      IFIN = 1
C
      GO TO 20
C
  130 BETA(MF) = BETAM
C
C-----END OF PRTEST----------------------------------------------------
      RETURN
      END
C
C------START OF STURMI-------------------------------------------------
C
      SUBROUTINE STURMI(BETA,X1,TOLN,EPSM,MMAX,MK1,MK2,IC,IWRITE)
C
C---------------------------------------------------------------------
      DOUBLE PRECISION  BETA(1)
      DOUBLE PRECISION  EPSM,X1,TOLN,EVL,EVU,BETA2
      DOUBLE PRECISION  U1,U2,V1,V2,ZERO,ONE
      INTEGER I,IC,ICD,IC0,IC1,IC2,MK1,MK2,MMAX
C---------------------------------------------------------------------
C
C     FOR ANY GOOD T-EIGENVALUE THAT HAS CONVERGED AS AN EIGENVALUE
C     OF THE T-MATRICES THIS SUBROUTINE CALCULATES
C     THE SMALLEST SIZE OF THE T-MATRIX, T(1,MK1) DEFINED
C     BY THE BETA ARRAY SUCH THAT MK1.LE.MMAX
C     AND THE INTERVAL (X1-TOLN,X1+TOLN) CONTAINS AT LEAST ONE
C     EIGENVALUE OF T(1,MK1). IT ALSO CALCULATES MK2 <= MMAX
C     AS THE SMALLEST SIZE T-MATRIX (IF ANY) SUCH THAT THIS INTERVAL
C     CONTAINS AT LEAST TWO EIGENVALUES OF T(1,MK2).
C     IF NO T-MATRIX OF ORDER < MMAX SATISFIES THIS REQUIREMENT
C     THEN MK2 IS SET EQUAL TO MMAX.  THE SINGULAR VECTOR PROGRAM
C     USES THESE VALUES TO DETERMINE A 1ST GUESS AT AN APPROPRIATE
C     SIZE T-MATRIX FOR THE SINGULAR VALUE  X1.
C
C     ON EXIT IC = NUMBER OF EIGENVALUES OF T(1,MK2) IN THIS INTERVAL
C
C     STURMI REGENERATES THE QUANTITIES BETA(I)**2 EACH TIME IT IS
C     CALLED, OBVIOUSLY FOR THE PRICE OF ANOTHER VECTOR OF LENGTH
C     MMAX THIS GENERATION COULD BE DONE ONCE IN THE MAIN
C     PROGRAM BEFORE THE LOOP ON THE CALLS TO SUBROUTINE STURMI.
C
C     IF ANY OF THE GOOD T-EIGENVALUES BEING CONSIDERED WERE MULTIPLE
C     AS SINGULAR VALUES OF THE USER-SPECIFIED MATRIX, THEN
C     THIS SUBROUTINE COULD BE MODIFIED TO COMPUTE ADDITIONAL
C     SIZES MKJ, J = 3, ...  WHICH COULD THEN BE USED IN THE
C     MAIN LANCZOS SINGULAR VECTOR PROGRAM TO COMPUTE ADDITIONAL
C     SINGULAR VECTORS CORRESPONDING TO THESE MULTIPLE SINGULAR
C     VALUES. THE MAIN PROGRAM LSVEC PROVIDED DOES NOT INCLUDE
C     THIS OPTION.
C
```

```
C-----------------------------------------------------------------------
C     INITIALIZATION OF PARAMETERS
      MK1 = 0
      MK2 = 0
      ZERO = 0.0D0
      ONE  = 1.0D0
      BETA(1) = ZERO
      EVL = X1-TOLN
      EVU = X1+TOLN
      U1 = ONE
      U2 = ONE
      IC0 = 0
      IC1 = 0
      IC2 = 0
C
C     MAIN LOOP FOR CALCULATING THE SIZES MK1,MK2
      DO 60 I = 1,MMAX
      BETA2 = BETA(I)*BETA(I)
      IF (U1.NE.ZERO) GO TO 10
      V1 = BETA(I)/EPSM
      GO TO 20
   10 V1 = BETA2/U1
   20 U1 = EVL - V1
      IF (U1.LT.ZERO) IC1 = IC1+1
      IF (U2.NE.ZERO) GO TO 30
      V2 = BETA(I)/EPSM
      GO TO 40
   30 V2 = BETA2/U2
   40 U2 = EVU - V2
      IF (U2.LT.ZERO) IC2 = IC2+1
C     TEST FOR CHANGE IN NUMBER OF T-EIGENVALUES ON (EVL,EVU)
      ICD = IC1-IC2
      IC = ICD-IC0
      IF (IC.GE.1) GO TO 50
      GO TO 60
   50 CONTINUE
      IF (IC0.EQ.0) MK1 = I
      IC0 = IC0+1
      IF (IC0.GT.1) GO TO 70
   60 CONTINUE
C
      I = I-1
      IF (IC0.EQ.0) MK1 = MMAX
   70 MK2 = I
      IC = ICD
C
      IF (IWRITE.EQ.1) WRITE(6,80) X1,MK1,MK2,IC
   80 FORMAT(' EVAL =',E20.12,' MK1 =',I6,' MK2 =',I6,' IC =',I3/)
C
      RETURN
C-----END OF STURMI-----------------------------------------------------
      END
C
C
C-----START OF INVERM---------------------------------------------------
```

```
C
      SUBROUTINE INVERM(BETA,V1,V2,X1,ERROR,ERRORV,EPS,G,MEV,IT,
     1 IWRITE)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  BETA(1),V1(1),V2(1)
      DOUBLE PRECISION  X1,U,Z,TEMP,RATIO,SUM,XU,NORM,TSUM,BETAM
      DOUBLE PRECISION  EPS,EPS3,EPS4,ERROR,ERRORV,ZERO,ONE
      REAL  G(1)
      DOUBLE PRECISION  DABS, DSQRT, DFLOAT
      DOUBLE PRECISION FINPRO
      REAL  ABS
C-----------------------------------------------------------------------
C
C     COMPUTES T-EIGENVECTORS FOR ISOLATED GOOD T-EIGENVALUES X1
C     USING INVERSE ITERATION ON T(1,MEV(X1)) SOLVING EQUATION
C     (T - X1*I)V2 = RIGHT-HAND SIDE (RANDOMLY-GENERATED) .
C     PROGRAM REFACTORS T- X1*I ON EACH ITERATION OF INVERSE ITERATION.
C     TYPICALLY ONLY ONE ITERATION IS NEEDED PER T-EIGENVALUE X1.
C
C     IF IWRITE = 1 THEN THERE ARE EXTENDED WRITES TO FILE 6 (TERMINAL)
C
C     ON ENTRY G CONTAINS A REAL*4 RANDOM VECTOR WHICH WAS GENERATED
C     IN MAIN PROGRAM.
C
C     ON ENTRY AND EXIT
C     MEV = ORDER OF T
C     BETA CONTAINS THE OFFDIAGONAL ENTRIES OF T.
C     EPS = 2. * MACHINE EPSILON
C
C     IN PROGRAM:
C     ITER = MAXIMUM NUMBER STEPS ALLOWED FOR INVERSE ITERATION
C     ITER = IT ON ENTRY.
C     V1,V2 = WORK SPACES USED IN THE FACTORIZATION OF T(1,MEV).
C     V1 AND V2 MUST BE OF DIMENSION AT LEAST MEV.
C
C     ON EXIT
C     V2 = THE UNIT EIGENVECTOR OF T(1,MEV) CORRESPONDING TO X1.
C     ERROR = |V2(MEV)| = ERROR ESTIMATE FOR CORRESPONDING
C             RITZ VECTOR FOR X1.
C
C     ERRORV = || T*V2 - X1*V2 || = ERROR ESTIMATE ON T-EIGENVECTOR.
C     IF IT.GT.ITER THEN ERRORV = -ERRORV
C     IT = NUMBER OF ITERATIONS ACTUALLY REQUIRED
C-----------------------------------------------------------------------
C     INITIALIZATION AND PARAMETER SPECIFICATION
      ONE  = 1.0D0
      ZERO = 0.0D0
      ITER = IT
      MP1 = MEV+1
      MM1 = MEV-1
      BETAM = BETA(MP1)
      BETA(MP1) = ZERO
C
C     CALCULATE SCALE AND TOLERANCES
```

```
      TSUM = ZERO
      DO 10 I = 2,MEV
   10 TSUM = TSUM + BETA(I)
C
      EPS3 = EPS*TSUM
      EPS4 = DFLOAT(MEV)*EPS3
C
C     GENERATE SCALED RANDOM RIGHT-HAND SIDE
      GSUM = ZERO
      DO 20 I = 1,MEV
   20 GSUM = GSUM+ABS(G(I))
      GSUM = EPS4/GSUM
C
C     INITIALIZE RIGHT HAND SIDE FOR INVERSE ITERATION
      DO 30 I = 1,MEV
   30 V2(I) = GSUM*G(I)
      IT = 1
C
C     CALCULATE UNIT EIGENVECTOR OF T(1,MEV) FOR ISOLATED GOOD
C     T-EIGENVALUE X1.
C
C     TRIANGULAR FACTORIZATION WITH NEAREST NEIGHBOR PIVOT
C     STRATEGY. INTERCHANGES ARE LABELLED BY SETTING BETA < 0.
C
   40 CONTINUE
      U = -X1
      Z = BETA(2)
C
      DO 60 I=2,MEV
      IF (BETA(I).GT.DABS(U)) GO TO 50
C     NO PIVOT INTERCHANGE
      V1(I-1) = Z/U
      V2(I-1) = V2(I-1)/U
      V2(I) = V2(I)-BETA(I)*V2(I-1)
      RATIO = BETA(I)/U
      U = -X1-Z*RATIO
      Z = BETA(I+1)
      GO TO 60
C     PIVOT INTERCHANGE
   50 CONTINUE
      RATIO = U/BETA(I)
      BETA(I) = -BETA(I)
      V1(I-1) = -X1
      U = Z-RATIO*V1(I-1)
      Z = -RATIO*BETA(I+1)
      TEMP = V2(I-1)
      V2(I-1) = V2(I)
      V2(I) = TEMP-RATIO*V2(I)
   60 CONTINUE
C
      IF (U.EQ.ZERO) U=EPS3
C
C     SMALLNESS TEST AND DEFAULT VALUE FOR LAST COMPONENT
C     PIVOT(I-1) = |BETA(I)| FOR INTERCHANGE CASE
C     (I-1,I+1) ELEMENT IN RIGHT FACTOR = BETA(I+1)
```

```
C     END OF FACTORIZATION AND FORWARD SUBSTITUTION
C
C     BACK SUBSTITUTION
      V2(MEV) = V2(MEV)/U
      DO 80 II = 1,MM1
      I = MEV-II
      IF (BETA(I+1).LT.ZERO) GO TO 70
C     NO PIVOT INTERCHANGE
      V2(I) = V2(I)-V1(I)*V2(I+1)
      GO TO 80
C     PIVOT INTERCHANGE
   70 BETA(I+1) = -BETA(I+1)
      V2(I) = (V2(I)-V1(I)*V2(I+1)-BETA(I+2)*V2(I+2))/BETA(I+1)
   80 CONTINUE
C
C
C     TESTS FOR CONVERGENCE OF INVERSE ITERATION
C     IF SUM |V2| COMPS. LE. 1 AND IT. LE. ITER DO ANOTHER INVIT STEP
C
      NORM = DABS(V2(MEV))
      DO 90 II = 1,MM1
      I = MEV-II
   90 NORM = NORM+DABS(V2(I))
C
C     IS DESIRED GROWTH IN VECTOR ACHIEVED ?
C     IF NOT, DO ANOTHER INVERSE ITERATION STEP UNLESS NUMBER ALLOWED IS
C     EXCEEDED.
      IF (NORM.GE.ONE) GO TO 110
C
      IT=IT+1
      IF (IT.GT.ITER) GO TO 110
C
      XU = EPS4/NORM
      DO 100 I=1,MEV
  100 V2(I) = V2(I)*XU
C
      GO TO 40
C
C     NORMALIZE COMPUTED T-EIGENVECTOR : V2 = V2/||V2||
C
  110 CONTINUE
C
      SUM = FINPRO(MEV,V2(1),1,V2(1),1)
      SUM = ONE/DSQRT(SUM)
      DO 120 II = 1,MEV
  120 V2(II) = SUM*V2(II)
C
C     SAVE ERROR ESTIMATE FOR LATER OUTPUT
      ERROR = DABS(V2(MEV))
C
C     GENERATE ERRORV = ||T*V2 - X1*V2||.
      V1(MEV) = BETA(MEV)*V2(MEV-1)-X1*V2(MEV)
      DO 130 J = 2,MM1
      JM = MP1 - J
      V1(JM) = BETA(JM)*V2(JM-1) + BETA(JM+1)*V2(JM+1
```

```
     1) - X1*V2(JM)
  130 CONTINUE
C
      V1(1) = BETA(2)*V2(2) - X1*V2(1)
      ERRORV = FINPRO(MEV,V1(1),1,V1(1),1)
      ERRORV = DSQRT(ERRORV)
      IF (IT.GT.ITER) ERRORV = -ERRORV
      IF (IWRITE.EQ.0) GO TO 150
C
C     FILE 6 (TERMINAL) OUTPUT OF ERROR ESTIMATES.
      WRITE(6,140) MEV,X1,ERROR,ERRORV
  140 FORMAT(' INVERSE ITERATION OUTPUT'/
     1 2X,'TSIZE',13X,'T-EIGENVALUE',11X,'U(M)',9X,'ERRORV'/
     1 I6,E25.16,2E15.5)
C
C     RESTORE BETA(MEV+1) = BETAM
  150 CONTINUE
      BETA(MP1) = BETAM
C-----END OF INVERM------------------------------------------------------
      RETURN
      END
C
C-----START OF LBISEC----------------------------------------------------
C
      SUBROUTINE LBISEC(BETA,EPSM,EVAL,EVALN,LB,UB,TTOL,M,NEVT)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  BETA(1),X0,X1,XL,XU,YU,YV,LB,UB
      DOUBLE PRECISION  EPSM,EP1,EVAL,EVALN,EVD,EPT
      DOUBLE PRECISION  ZERO,ONE,HALF,TTOL,TEMP
      DOUBLE PRECISION  DABS,DSQRT,DFLOAT
C-----------------------------------------------------------------------
C     SPECIFY PARAMETERS
      ZERO = 0.0D0
      HALF = 0.5D0
      ONE  = 1.0D0
      XL = LB
      XU = UB
C
C     EP1 = DSQRT(1000+M)*TTOL     TTOL = EPSM*TKMAX
C     TKMAX = MAX(BETA(K), K= 1,KMAX)
C
      TEMP = DFLOAT(1000+M)
      EP1 = DSQRT(TEMP)*TTOL
C
      NA = 0
      X1 = XU
      JSTURM = 1
      GO TO 60
C     FORWARD STURM CALCULATION
   10 NA = NEV
      X1 = XL
      JSTURM = 2
      GO TO 60
C     FORWARD STURM CALCULATION
```

```
   20 NEVT = NEV
C
C     WRITE(6,30) M,EVAL,NEVT,EP1
   30 FORMAT(/3X,'TSIZE',23X,'EV',9X/I8,E25.16/
     1 I6,' = NUMBER OF T(1,M) EIGENVALUES ON TEST INTERVAL'/
     1 E12.3,' = CONVERGENCE TOLERANCE'/)
C
      IF (NEVT.NE.1) GO TO 120
C
C     BISECTION LOOP
      JSTURM = 3
   40 X1 = HALF*(XL+XU)
      X0 = XU-XL
      EPT = EPSM*(DABS(XL) + DABS(XU)) + EP1
C     CONVERGENCE TEST
      IF (X0.LE.EPT) GO TO 100
      GO TO 60
C     FORWARD STURM CALCULATION
   50 CONTINUE
      IF(NEV.EQ.0) XU = X1
      IF(NEV.EQ.1) XL = X1
      GO TO 40
C     NEV = NUMBER OF EIGENVALUES OF T(1,M) ON (X1,XU)
C     THERE IS EXACTLY ONE EIGENVALUE OF T(1,M) ON (XL,XU)
C
C     FORWARD STURM CALCULATION
   60 NEV = -NA
      YU = ONE
      DO 90 I = 1,M
      IF (YU.NE.ZERO) GO TO 70
      YV = BETA(I)/EPSM
      GO TO 80
   70 YV = BETA(I)*BETA(I)/YU
   80 YU = X1 - YV
      IF (YU.GE.ZERO) GO TO 90
      NEV = NEV+1
   90 CONTINUE
      GO TO (10,20,50), JSTURM
C
  100 CONTINUE
C
      EVALN = X1
      EVD = DABS(EVALN-EVAL)
C     WRITE(6,110) EVALN,EVAL,EVD
  110 FORMAT(/20X,'EVALN',21X,'EVAL',6X,'CHANGE'/2E25.16,E12.3/)
C
  120 CONTINUE
      RETURN
C-----END OF LBISEC-----------------------------------------------------
      END
```

6.7 LSVAL: LSVEC: File Definitions, Sample Input Files

Below is a listing of the input/output files which are accessed by the Lanczos program LSVAL for computing singular values of real rectangular matrices on user-specified intervals. Included also is a sample of the input file which LSVAL requires on file 5. The parameters in this file are supplied in free format. File 8 contains the data for the rectangular mxn matrix A.

```
Sample Specifications for Input/Output Files for LSVAL
----------------------------------------------------------------
 LSVAL EXEC FOR LANCZOS SINGULAR VALUE CALCULATIONS
FI 06 TERM
FILEDEF  1 DISK &1        NSHISTOR  A (RECFM F LRECL 80 BLOCK 80
FILEDEF  2 DISK &1        SVHISTOR  A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1        GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1        ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK LSVAL     INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1        INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1        DISTINCT  A (RECFM F LRECL 80 BLOCK 80
LOAD   LSVAL   LSSUB    LSMULT
----------------------------------------------------------------
```

```
Sample Input File for LSVAL
----------------------------------------------------------------------
 LANCZOS SINGULAR VALUE PROCEDURE,
 WITHOUT REORTHOGONALIZATION BUT WITH BIDIAGONALIZATION.
LINE 1      M    N    KMAX   NMEVS    MATNO
         100   100    300       1      2220
LINE 2   SVSEED    RHSEED    MXINIT    MXSTUR
       49302312   7549309         5    100000
LINE 3     ISTART    ISTOP
               0        1
LINE 4     IHIS    IDIST    IWRITE    IPAR
             1        0         1       2
LINE 5  RELTOL(RELATIVE TOLERANCE USED IN 'COMBINING'  GOOD EVALS
   .0000000001
LINE 6  MB(1)   MB(2)   MB(3)   MB(4)  (SIZE OF T(1,MEV) MUST BE EVEN)
          280
LINE 7   NINT    (NUMBER OF BISEC INTERVALS)
           1
LINE 8    LB(1)    LB(2)   LB(3)   LB(4)  (LOWER BOUNDS INTERVALS)
            0.0
LINE 9    UB(1)    UB(2)   UB(3)   UB(4)  (UPPER BOUNDS INTERVALS)
           1.0
----------------------------------------------------------------------
```

Below is a listing of the input/output files which are accessed by the Lanczos program for computing singular vectors, LSVEC. Included also is a sample of the input file which LSVEC requires on file 5. The parameters in this file are supplied in free format.

File 8 contains the data for the rectangular mxn matrix A. LSVEC computes singular vectors for each of a user-specified subset of the singular values computed by the companion program LSVAL.

```
Sample Specifications of the Input/Output Files for LSVEC
---------------------------------------------------------------
 LSVEC EXEC TO RUN LANCZOS SINGULAR VECTOR PROGRAM
FI 06 TERM
FILEDEF  2 DISK &1       SVHISTOR  A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1       GOODSV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1       ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK LSVEC    INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1       INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  9 DISK &1       ERREST    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 10 DISK &1       BOUNDS    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1       TEIGVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 12 DISK &1       RITZVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 13 DISK &1       PAIGE     A (RECFM F LRECL 80 BLOCK 80
LOAD   LSVEC   LSSUB   LSMULT
---------------------------------------------------------------
```

```
Sample Input File for LEVEC
-----------------------------------------------------------------------
 LSVEC SINGULAR VECTORS, NO REORTHOGONALIZATION BUT BIDIAGONALIZATION
LINE 1  MATNO      M     N
          100    100    80
LINE 2  MDIMTV    MDIMRV  MBETA (MAX.DIMENSIONS,TVEC,RITVEC AND BETA
         10000     10000   2000
LINE 3       RELTOL
        .0000000001
LINE 4  MBOUND   NTVCON SVTVEC IREAD (FLAGS
            0        1      0     1
LINE 5  TVSTOP   LVCONT ERCONT  IWRITE (FLAGS
            0        1      1       1
LINE 6    RHSEED  (RANDOM GENERATOR SEED FOR STARTING VECTOR IN INVERM)
        45329517
-----------------------------------------------------------------------
```

Chapter 7

Nondefective Complex Symmetric Matrices

7.1 Introduction

The FORTRAN codes in this chapter address the question of computing distinct eigenvalues and eigenvectors of a nondefective, complex symmetric matrix, using a single-vector Lanczos procedure. For a given nondefective, complex symmetric matrix A, these codes compute complex scalars λ and corresponding complex vectors $x \neq 0$ such that

$$Ax = \lambda x. \tag{7.1.1}$$

Definition 3 . *A complex nxn matrix* $A \equiv (a_{ij})$, $1 \leq i, j \leq n$, *is complex symmetric if and only if for every* i *and* j, $a_{ij} = a_{ji}$. *A complex symmetric matrix is nondefective if and only if it has a complete set of eigenvectors.*

It is straight-forward to show from Definition 3 that if $A = B + iC$, where A and B are real matrices and $i = \sqrt{-1}$, is a complex symmetric matrix then B and C are real symmetric matrices. It is also easy to prove that if λ and μ are two distinct eigenvalues of A and x and y are corresponding eigenvectors of A, then the Euclidean inner product applied to the complex vectors x and y satisfies

$$x^T y = 0. \tag{7.1.2}$$

In Eqn(7.1.2) the superscript T denotes transpose. Thus, although the eigenvectors of a complex symmetric matrix are not orthogonal with respect to the complex norm, $\|x\|_C^2 = \sum_{i=1}^n \overline{x(i)} x(i)$, they are real orthogonal in the sense specified in Eqn(7.1.2). Therefore, when we consider generalizing the Lanczos recursion to the complex symmetric case we are led to consider an 'inner product' which is a mixture of real and complex quantities. In fact the Euclidean inner product, which of course is not an inner product for complex vectors, is the natural 'inner product' to use in the complex symmetric case.

Complex symmetric matrices are not 'easy' like real symmetric matrices. They bear little resemblance to real symmetric matrices. Complex symmetric matrices need not have complete sets of eigenvectors. Even if a complete set of eigenvectors exists, eigenvectors corresponding to different eigenvalues are only real orthogonal in the sense of Eqn(7.1.2). If a small perturbation is applied to a complex symmetric matrix, then large perturbations in the eigenvalues may result. See Wilkinson [25] for a discussion of the properties of complex symmetric matrices.

The Lanczos recursion as presented in Eqns(1.2.1), (1.2.2) is only applicable to real symmetric matrices so we ask the question: How do we construct a complex symmetric version of the basic Lanczos recursion which will give us the desired eigenvalues? We have used what has been suggested elsewhere, Moro and Freed [16]. In particular, we use the recursion in Eqn(1.2.1) with the formulas for the scalars α_i and β_{i+1} given in Eqn(1.2.2), except that the quantities involved are now complex-valued, but the real Euclidean inner product is used. See Section 6.3 in Chapter 6 in Volume 1.

There are some fundamental differences between the amount of computation required by the complex symmetric codes versus that required by the real symmetric codes. First, all of the complex symmetric computations are done in double precision complex arithmetic. All the vectors used are complex vectors. Each of the Lanczos matrices generated is a complex symmetric tridiagonal matrix. Unfortunately, there is no simple analog of the bisection procedure used in the real symmetric case which would allow us to compute the eigenvalues of a given complex symmetric tridiagonal matrix on only some small portion of the spectrum. We are therefore forced to do a complete eigenvalue computation on each complex symmetric tridiagonal matrix which we consider. Actually in the complex symmetric case we are forced to do two complete eigenvalue computations for each Lanczos tridiagonal matrix which we consider. Two are required because the identification test for categorizing the eigenvalues of the Lanczos T-matrices into 'good' and 'spurious' ones uses the eigenvalues of the corresponding tridiagonal matrix obtained from the Lanczos T-matrix by crossing out the first row and column of that matrix. This is the same identification test as that used in the procedures for real symmetric problems. However, in the real symmetric cases this test is directly incorporated into the BISEC subroutine which is used to compute the eigenvalues of the Lanczos matrices, and the resulting cost of this test is negligible for those types of problems.

These codes can be used to compute either a very few or very many of the distinct eigenvalues of a nondefective, complex symmetric matrix. As the documentation in the next section indicates, the A-multiplicity of a given computed eigenvalue can be obtained only with additional computation, and the modifications required to do this additional computation are not included in these versions of the codes.

The Lanczos recursions used generate a family of complex symmetric, tridiagonal matrices. A real orthogonal analog of the EISPACK [23, 8] subroutine IMTQL1 which we call CMTQL1 was developed to compute the eigenvalues of the complex symmetric, tridiagonal Lanczos matrices generated. There is no reorthogonalization of the Lanczos vectors at any stage in any of the computations.

CSLEVAL, the main program for the complex symmetric eigenvalue computations, calls the subroutines COMPEV and CMTQL1 to compute the eigenvalues of the Lanczos T-matrices specified by the user. The eigenvalues of the related complex symmetric tridiagonal matrices obtained by deleting the first row and first column from the given Lanczos T-matrix are also computed. COMPEV then determines the T-multiplicities of the T-eigenvalues and sorts the computed T-eigenvalues into two classes, the 'good' T-eigenvalues and the 'spurious' T-eigenvalues. The 'good' T-eigenvalues are accepted as approximations to eigenvalues of the user-specified matrix A. The accuracy of these 'good' T-eigenvalues as eigenvalues of A is then estimated using error estimates computed by a complex version of the subroutine INVERR. Error estimates are computed only for isolated 'good' T-eigenvalues. All other 'good' T-eigenvalues are assumed to have converged. Convergence is then checked. If convergence has not yet occurred and a larger Lanczos matrix has been specified by the user, the program will continue on to the larger T-matrix, repeating the above procedure on this larger matrix.

Once the eigenvalues been computed accurately enough, the user can select a subset of the 'converged' eigenvalues for which eigenvectors are to be computed. The main program CSLEVEC, for computing eigenvectors of complex symmetric matrices, is then used to compute these desired eigenvectors.

As stated earlier, all computations are in double precision complex arithmetic. The user must supply a subroutine USPEC which defines and initializes the user-specified matrix A and a subroutine CMATV which computes matrix-vector multiplies Ax for any given vector x. These subroutines must be constructed in such a way as to take advantage of the sparsity (and/or structure) of the user-supplied A-matrix and such that these computations are done accurately.

The user should note that the complex symmetric computations are considerably more expensive than the corresponding real symmetric ones. Two complete T-matrix eigenvalue computations must be done for each T-size. Moreover, the accuracy of these computations is noticeably less than that achievable in the real symmetric case. This is to be expected from the perturbation analysis for the complex symmetric case. Therefore we reduced the anticipated accuracy of the computed eigenvalues and used larger tolerances in our multiplicity and spuriousness tests. These larger tolerances decrease the resolution capabilities of these codes. However, these tolerances are realistic. Moreover, these complex symmetric codes cannot be expected to handle stiff problems effectively. More details about these complex symmetric, single-vector Lanczos procedures are included in Chapter 6 of Volume 1.

7.2 Documentation for the Codes in Chapter 7

```
C-----CSLEVALD-----------------------------------------------------
C
C     DOCUMENTATION FOR SINGLE-VECTOR
C     LANCZOS EIGENVALUE/EIGENVECTOR PROGRAMS FOR
C     NONDEFECTIVE COMPLEX SYMMETRIC MATRICES
C
C-----------------------------------------------------------------------
C         REFERENCE:  Cullum and Willoughby, Chapter 6,
C         Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C         Applied Mathematics, 2002. SIAM Publications,
C         Philadelphia, PA. USA
C
C
C-----------------------------------------------------------------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them
C  are used in other scientific or engineering research works
C  the names of the authors of these codes and appropriate
C  references to their written work are to be incorporated in the
C  derivative works.
C
C  This header is not to be removed from these codes.
C
C     GIVEN A NONDEFECTIVE COMPLEX SYMMETRIC MATRIX A OF ORDER N
C     THE THREE SETS OF FORTRAN FILES LABELLED CSLEVAL, CSLESUB,
C     AND CSLEMULT CAN BE USED TO COMPUTE DISTINCT EIGENVALUES OF
C     A.  NOTE THAT THESE PROGRAMS DIFFER FROM THE REAL SYMMETRIC
C     AND HERMITIAN PROGRAMS IN THAT IT IS NOT POSSIBLE TO
C     COMPUTE THE EIGENVALUES OF THE LANCZOS TRIDIAGONAL MATRICES
C     ONLY IN SPECIFIED INTERVALS.  THUS, ON ANY GIVEN
C     ITERATION ALL OF THE EIGENVALUES OF THESE TRIDIAGONAL MATRICES
C     MUST BE COMPUTED.  IN FACT TWO COMPLETE TRIDIAGONAL EIGENVALUE
C     COMPUTATIONS ARE USED.
C
C     CORRESPONDING EIGENVECTORS FOR SELECTED, COMPUTED EIGENVALUES CAN
C     BE COMPUTED USING THE CORRESPONDING SETS OF FILES LABELLED
C     CSLEVEC, CSLESUB AND CSLEMULT.
C
C     THESE PROGRAMS ALL USE A GENERALIZATION OF LANCZOS
C     TRIDIAGONALIZATION TO COMPLEX SYMMETRIC MATRICES TO
```

```
C     GENERATE COMPLEX SYMMETRIC TRIDIAGONAL MATRICES, T(1,MEV)         CSL00520
C     OF ORDER MEV.  NO REORTHOGONALIZATION IS USED.  SUBSETS OF        CSL00530
C     THE EIGENVALUES OF THESE T-MATRICES, LABELLED AS THE              CSL00540
C     'GOOD EIGENVALUES', YIELD APPROXIMATIONS TO THE DESIRED           CSL00550
C     EIGENVALUES OF A.  CORRESPONDING RITZ VECTORS ARE APPROXIMATIONS  CSL00560
C     TO THE DESIRED EIGENVECTORS OF A.  NOTE THAT IN THE DISCUSSION    CSL00570
C     T(1,MEV) DENOTES THE LANCZOS MATRIX OF ORDER MEV AND T(2,MEV)     CSL00580
C     DENOTES THE MATRIX OF SIZE MEV-1 OBTAINED FROM T(1,MEV) BY        CSL00590
C     DELETING THE FIRST ROW AND COLUMN OF T(1,MEV).                    CSL00600
C                                                                       CSL00610
C     THE IDEAS USED IN THESE PROGRAMS ARE DISCUSSED IN THE FOLLOWING   CSL00620
C     REFERENCES.                                                       CSL00630
C                                                                       CSL00640
C     1.  JANE CULLUM AND RALPH A. WILLOUGHBY, LANCZOS ALGORITHMS       CSL00650
C         FOR LARGE SYMMETRIC MATRICES,  PROGRESS IN                    CSL00660
C         SCIENTIFIC COMPUTING, EDITORS, G. GOLUB, H.O. KREISS,         CSL00670
C         S. ARBARBANEL, AND R. GLOWINSKI,  BIRKHAUSER BOSTON INC.,     CSL00680
C         CAMBRIDGE, MASSACHUSETTS, 1985.                               CSL00690
C                                                                       CSL00700
C     2.  JANE CULLUM AND RALPH A. WILLOUGHBY, COMPUTING EIGENVECTORS   CSL00710
C         (AND EIGENVALUES) OF LARGE, SYMMETRIC MATRICES USING          CSL00720
C         LANCZOS TRIDIAGONALIZATION, LECTURE NOTES IN MATHEMATICS,     CSL00730
C         773, NUMERICAL ANALYSIS PROCEEDINGS, DUNDEE 1979, EDITED BY   CSL00740
C         G. A. WATSON, SPRINGER-VERLAG, (1980), BERLIN, PP.46-63.      CSL00750
C                                                                       CSL00760
C     3. IBID, LANCZOS AND THE COMPUTATION IN SPECIFIED INTERVALS OF    CSL00770
C        THE SPECTRUM OF LARGE SPARSE, REAL SYMMETRIC MATRICES, SPARSE  CSL00780
C        MATRIX PROCEEDINGS 1978, ED. I.S. DUFF AND G. W. STEWART,      CSL00790
C        SIAM, PHILADELPHIA, PP.220-255, 1979.                          CSL00800
C                                                                       CSL00810
C     4. IBID, COMPUTING EIGENVALUES OF VERY LARGE SYMMETRIC MATRICES-  CSL00820
C        AN IMPLEMENTATION OF A LANCZOS ALGORITHM WITHOUT               CSL00830
C        REORTHOGONALIZATION, J. COMPUT. PHYS. 44(1981), 329-358.       CSL00840
C                                                                       CSL00850
C     5. IBID, A LANCZOS ALGORITHM FOR NONDEFECTIVE COMPLEX SYMMETRIC   CSL00860
C        MATRICES, IBM RESEARCH REPORT, 1984.                           CSL00870
C                                                                       CSL00880
C-----PORTABILITY-------------------------------------------------------CSL00890
C                                                                       CSL00900
C     PROGRAMS WERE TESTED FOR PORTABILITY USING THE PFORT VERIFIER.    CSL00910
C     FOR DETAILS OF THE VERIFIER SEE FOR EXAMPLE, B. G. RYDER AND      CSL00920
C     A. D. HALL, 'THE PFORT VERIFIER', COMPUTING SCIENCE TECHNICAL     CSL00930
C     REPORT 12, BELL LABORATORIES, MURRAY HILL, NEW JERSEY 07974,      CSL00940
C     (REVISED), JANUARY 1981.                                          CSL00950
C                                                                       CSL00960
C     PORTABILITY:                                                      CSL00970
C     THESE PROGRAMS ARE NOT PORTABLE DUE TO THE USE OF COMPLEX*16      CSL00980
C     VARIABLES AND CORRESPONDING COMPLEX FUNCTIONS.  IN ADDITION, THE  CSL00990
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE               CSL01000
C     CONSTRUCTIONS.                                                    CSL01010
C     IN CSLEVAL AND IN CSLEVEC                                         CSL01020
C         1.  DATA/MACHEP STATEMENT                                     CSL01030
C         2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                    CSL01040
C         3.  FORMAT(20A4) USED FOR THE EXPLANATORY HEADER ARRAY, EXPLANCSL01050
C         4.  HEXADECIMAL FORMAT (4Z20) FOR ALPHA/BETA FILES 1 AND 2.   CSL01060
```

```
C     IN CSLEMULT
C         1.  IN CMATV AND  USPEC THE ENTRY THAT PASSES THE STORAGE
C             LOCATIONS OF THE ARRAYS DEFINING THE USER-SPECIFIED
C             MATRIX.
C         2.  IN SAMPLE USPEC PROVIDED : FREE FORMAT (8,*), THE
C             FORMAT (20A4), AND THE DATA/MACHEP STATEMENT.
C
C     IN THE COMMENTS BELOW :
C     REAL*16 = COMPLEX VARIABLE, 16 BYTES OF STORAGE
C     REAL*8 = REAL VARIABLE, 8 BYTES OF STORAGE
C     REAL*4 = REAL VARIABLE, 4 BYTES OF STORAGE
C     INTEGER*4 = INTEGER VARIABLE, 4 BYTES OF STORAGE
C
C-----A-MATRIX SPECIFICATION-------------------------------------------
C
C     SUBROUTINE USPEC IS USED TO SPECIFY THE USER-SUPPLIED MATRIX.
C     SUBROUTINE CMATV IS A CORRESPONDING MATRIX-VECTOR MULTIPLY
C     SUBROUTINE WHICH SHOULD BE DESIGNED TO TAKE ADVANTAGE OF
C     ANY SPECIAL PROPERTIES OF THE USER-SUPPLIED MATRIX.  THE
C     MATRIX-VECTOR MULTIPLIES REQUIRED BY THE LANCZOS PROCEDURES
C     MUST BE COMPUTED RAPIDLY AND ACCURATELY.
C
C     SUBROUTINE USPEC HAS THE CALLING SEQUENCE
C
C            CALL USPEC(N,MATNO)
C
C     WHERE N IS THE ORDER OF THE USER-SUPPLIED MATRIX A AND
C     MATNO IS A <= 8 DIGIT INTEGER USED AS A MATRIX AND
C     TEST IDENTIFICATION NUMBER.  THIS SUBROUTINE DEFINES (DIMENSIONS)
C     THE ARRAYS REQUIRED TO SPECIFY THE USER-SUPPLIED MATRIX AND
C     INITIALIZES THESE ARRAYS AND ANY OTHER PARAMETERS NEEDED TO
C     DEFINE THE MATRIX.  THE STORAGE LOCATIONS OF THESE PARAMETERS
C     AND ARRAYS ARE THEN PASSED TO THE MATRIX-VECTOR MULTIPLY
C     SUBROUTINE CMATV VIA AN ENTRY.  A SAMPLE USPEC SUBROUTINE
C     IS INCLUDED.  THIS SAMPLE SUBROUTINE ASSUMES THAT THE MATRIX
C     IS STORED ON FILE 8 IN A TYPICAL SPARSE MATRIX FORMAT.
C     SEE THE HEADER ON THE SUBROUTINE USPEC FOR DETAILS ON THIS
C     PARTICULAR STORAGE FORMAT.
C
C     SUBROUTINE CMATV HAS THE CALLING SEQUENCE
C
C            CALL CMATV(W,U,SUM)
C
C     IN THE COMPLEX SYMMETRIC CASE, U AND W ARE
C     COMPLEX*16 VECTORS AND SUM IS A COMPLEX*16
C     SCALAR.   CMATV CALCULATES U = A*W - SUM*U FOR THE
C     USER-SPECIFIED MATRIX A.  THE ARRAY AND PARAMETER INFORMATION
C     NEEDED TO PERFORM THE MATRIX-VECTOR MULTIPLIES IS PASSED TO
C     THE CMATV SUBROUTINE FROM THE USPEC SUBROUTINE VIA THE CMATVE
C     ENTRY IN CMATV.  A SAMPLE CMATV SUBROUTINE IS INCLUDED WHICH
C     COMPUTES MATRIX-VECTOR MULTIPLIES FOR AN ARBITRARY SPARSE,
C     COMPLEX SYMMETRIC MATRIX STORED IN THE SPARSE FORMAT
C     SPECIFIED IN THE SAMPLE USPEC SUBROUTINE.
C
C     CMATV IS CALLED FROM THE SUBROUTINE LANCZS WHICH GENERATES
```

```
C     THE T-MATRICES IN THE ALPHA, BETA ARRAYS.  IT IS ALSO CALLED      CSL01620
C     FROM THE MAIN PROGRAM CSLEVEC FOR THE EIGENVECTOR COMPUTATIONS.   CSL01630
C     CMATV IS DECLARED AS AN EXTERNAL VARIABLE AND IS AN ARGUMENT      CSL01640
C     FOR THE SUBROUTINE LANCZS.                                        CSL01650
C                                                                       CSL01660
C     THE USPEC AND CMATV SUBROUTINES MUST BE MODIFIED BY THE USER      CSL01670
C     TO ACCOMODATE THE USER'S SPECIFIED MATRIX.                        CSL01680
C                                                                       CSL01690
C     THE MAIN PROGRAMS FOR THE EIGENVALUE AND EIGENVECTOR              CSL01700
C     CALCULATIONS ASSUME THAT INPUT FILE 5 CONTAINS N = ORDER OF       CSL01710
C     THE MATRIX AND MATNO = AN IDENTIFICATION NUMBER OF <= 8 DIGITS    CSL01720
C     FOR THE MATRIX AND THE RUN.                                       CSL01730
C                                                                       CSL01740
C                                                                       CSL01750
C-----MACHEP------------------------------------------------------------CSL01760
C                                                                       CSL01770
C                                                                       CSL01780
C     MACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING THE RELATIVE   CSL01790
C     PRECISION OF THE FLOATING POINT ARITHMETIC USED.                  CSL01800
C     MACHEP = 2.2 * 10**-16 FOR DOUBLE PRECISION ARITHMETIC ON         CSL01810
C     IBM 370-3081.                                                     CSL01820
C                                                                       CSL01830
C     THE USER WILL HAVE TO RESET THIS PARAMETER TO                     CSL01840
C     THE CORRESPONDING VALUE FOR THE MACHINE BEING USED.  NOTE THAT    CSL01850
C     IF A MACHINE WITH A MACHINE EPSILON THAT IS MUCH LARGER THAN THE  CSL01860
C     VALUE GIVEN HERE IS BEING USED, THEN THERE COULD BE               CSL01870
C     PROBLEMS WITH THE TOLERANCES.                                     CSL01880
C                                                                       CSL01890
C                                                                       CSL01900
C-----SUBROUTINES AND FUNCTIONS USER MUST SUPPLY------------------------CSL01910
C                                                                       CSL01920
C     GENRAN,  MASK,  USPEC,  AND  CMATV                                CSL01930
C                                                                       CSL01940
C     GENRAN = COMPUTES K PSEUDO-RANDOM NUMBERS AND STORES THEM IN      CSL01950
C              THE REAL ARRAY, G.  THIS SUBROUTINE IS USED TO           CSL01960
C              GENERATE A STARTING VECTOR FOR THE LANCZOS PROCEDURE     CSL01970
C              IN THE SUBROUTINE LANCZS AND A STARTING RIGHT-HAND SIDE  CSL01980
C              FOR INVERSE ITERATION IN THE SUBROUTINE INVERR.          CSL01990
C                                                                       CSL02000
C              TESTS REPORTED IN THE REFERENCES USED EITHER GGL1 OR     CSL02010
C              GGL2 FROM THE IBM LIBRARY  SLMATH.                       CSL02020
C              THE EXISTING CALLING SEQUENCE IS:                        CSL02030
C                                                                       CSL02040
C                      CALL GENRAN(IIX,G,K).                            CSL02050
C                                                                       CSL02060
C              WHERE IIX =INTEGER SEED, G = REAL*4 ARRAY WHOSE          CSL02070
C              DIMENSION MUST BE >= K.  K RANDOM NUMBERS ARE GENERATED  CSL02080
C              AND PLACED IN G.                                         CSL02090
C                                                                       CSL02100
C     MASK = MASKS OVERFLOW AND UNDERFLOW.                              CSL02110
C            USER MUST SUPPLY OR COMMENT OUT CALL.                      CSL02120
C                                                                       CSL02130
C     USPEC = DIMENSIONS AND INITIALIZES ARRAYS NEEDED TO SPECIFY       CSL02140
C             USER-SUPPLIED MATRIX.  SEE A-MATRIX SPECIFICATION SECTION.CSL02150
C                                                                       CSL02160
```

```
C     CMATV = MATRIX-VECTOR MULTIPLY FOR USER-SUPPLIED MATRIX.
C             SEE A-MATRIX SPECIFICATION SECTION.
C
C
C-----------------------------------------------------------------------
C
C     COMMENTS FOR EIGENVALUE COMPUTATIONS
C
C-----------------------------------------------------------------------
C
C
C-----PARAMETER CONTROLS FOR EIGENVALUE PROGRAMS------------------------
C
C     PARAMETER CONTROLS ARE INTRODUCED TO ALLOW SEGMENTATION OF THE
C     EIGENVALUE COMPUTATIONS AND TO ALLOW VARIOUS COMBINATIONS OF
C     READ/WRITES.
C
C     THE FLAG ISTART CONTROLS THE T-MATRIX (ALPHA/BETA HISTORY)
C     GENERATION.
C
C     ISTART = (0,1)  MEANS
C
C              (0) THERE IS NO EXISTING ALPHA/BETA HISTORY AND ONE
C                  MUST BE GENERATED.
C
C              (1) THERE IS AN EXISTING ALPHA/BETA HISTORY AND IT IS
C                  TO BE READ IN FROM FILE 2 AND EXTENDED IF NECESSARY.
C
C     THE FLAG ISTOP CAN BE USED IN CONJUNCTION WITH THE FLAG ISTART TO
C     ALLOW SEGMENTATION OF THE EIGENVALUE COMPUTATIONS.
C
C     ISTOP  = (0,1)  MEANS
C
C              (0) PROGRAM COMPUTES ONLY THE REQUESTED ALPHAS/BETAS,
C                  STORES THEM AND THE LAST 2 LANCZOS VECTORS GENERATED
C                  IN FILE 1 AND THEN TERMINATES.
C
C              (1) PROGRAM COMPUTES REQUESTED ALPHAS/BETAS AND THEN
C                  USES THE CMTQL1 SUBROUTINE TO CALCULATE EIGENVALUES
C                  OF THE TRIDIAGONAL MATRICES GENERATED FOR THE ORDERS
C                  SPECIFIED BY THE USER.  PROGRAM THEN USES THE
C                  SUBROUTINE INVERR TO COMPUTE ERROR ESTIMATES FOR
C                  THE ISOLATED GOOD T-EIGENVALUES WHICH ARE USED TO
C                  CHECK THE CONVERGENCE OF THESE GOOD T-EIGENVALUES.
C
C     CONTROL PARAMETERS FOR WRITES
C
C     IHIS  = (0,1)  MEANS
C
C             (0) IF ISTOP .GT. 0 THEN ALPHAS/BETAS ARE NOT SAVED
C                 ON FILE 1.
C
C             (1) PROGRAM WRITES ALPHAS/BETAS AND LAST 2 LANCZOS
C                 VECTORS TO FILE 1 SO THAT THE T-MATRIX GENERATION
C                 MAY BE REUSED OR CONTINUED LATER IF NECESSARY.
```

```
C                 TYPICALLY ONE WOULD ALWAYS DO THIS ON ANY RUN WHERE
C                 A HISTORY FILE IS BEING GENERATED.  HISTORY MUST
C                 BE SAVED IN MACHINE FORMAT ((4Z20) FOR IBM/3081)
C                 SO THAT NO ERRORS ARE INTRODUCED DUE TO FORMAT
C                 CONVERSIONS.
C
C     IDIST  = (0,1)  MEANS
C
C              (0) DISTINCT EIGENVALUES OF T-MATRICES ARE NOT SAVED.
C
C              (1) PROGRAM WRITES COMPUTED DISTINCT EIGENVALUES OF
C                  T-MATRICES ALONG WITH THEIR T-MULTIPLICITIES
C                  TO FILE 11.
C
C     IWRITE = (0,1)   MEANS
C
C              (0) NO EXTENDED OUTPUT FROM SUBROUTINES COMPEV AND INVERR
C                  IS SENT TO FILE 6.
C
C              (1) INDIVIDUAL COMPUTED EIGENVALUES AND CORRESPONDING
C                  ERROR ESTIMATES FROM THE SUBROUTINES COMPEV AND
C                  INVERR ARE PRINTED OUT TO FILE 6 AS THEY ARE COMPUTED
C
C     SAVTEV = (-1,0,1)   MEANS
C
C              (-1) NO T-EIGENVALUE COMPUTATIONS.  PREVIOUSLY-COMPUTED
C                   EIGENVALUES OF T(1,MEV) AND T(2,MEV) ARE TO
C                   BE READ IN FROM FILE 10.
C
C              (0) COMPUTED EIGENVALUES OF T(1,MEV) AND OF T(2,MEV)
C                  ARE NOT TO BE SAVED ON FILE 10.  THIS IS NOT
C                  RECOMMENDED IF THE T-MATRICES BEING USED ARE VERY
C                  LARGE BECAUSE IN THAT CASE THE TRIDIAGONAL
C                  EIGENVALUE COMPUTATIONS ARE VERY EXPENSIVE.
C
C              (1) COMPUTED EIGENVALUES OF T(1,MEV) AND OF T(2,MEV)
C                  WILL BE SAVED ON FILE 10.  THIS IS RECOMMENDED
C                  BECAUSE ONCE THESE T-EIGENVALUES ARE COMPUTED THE
C                  LATTER PORTION OF THE EIGENVALUE PROGRAM IS EASILY
C                  RESTARTED FROM THE POINT OF THESE EIGENVALUE
C                  COMPUTATIONS.
C
C     THE PROGRAM ALWAYS MAKES A SEPARATE LIST OF THE COMPUTED GOOD
C     T-EIGENVALUES ALONG WITH THEIR MINIMAL GAPS AND WRITES THEM OUT
C     TO FILE 3.  CORRESPONDING ERROR ESTIMATES FOR ANY ISOLATED
C     GOOD T-EIGENVALUES ARE ALWAYS WRITTEN TO FILE 4.
C
C
C-----INPUT/OUTPUT FILES FOR EIGENVALUE PROGRAMS------------------------
C
C     ANY INPUT DATA OTHER THAN THE ALPHA/BETA HISTORY OR PREVIOUSLY-
C     COMPUTED EIGENVALUES OF T(1,MEV) AND T(2,MEV) SHOULD BE STORED
C     ON FILE 5. SEE SAMPLE INPUT/OUTPUT FROM TYPICAL RUN.
C     THE READ STATEMENTS IN THE GIVEN FORTRAN PROGRAM ASSUME THAT
C     THE DATA STORED ON FILE 5 IS IN FREE FORMAT.  USER SHOULD NOTE
```

```
C     THAT 'FREE FORMAT' IS NOT CLASSIFIED AS PORTABLE BY PFORT SO THAT CSL03270
C     THE USER MAY HAVE TO MODIFY THE READ STATEMENTS FROM FILE 5 TO    CSL03280
C     CONFORM TO WHAT IS PERMISSIBLE ON THE MACHINE BEING USED.         CSL03290
C                                                                       CSL03300
C     FILE 6 WAS USED AS THE INTERACTIVE TERMINAL OUTPUT FILE.          CSL03310
C     THIS FILE PROVIDES A RUNNING ACCOUNT OF THE PROGRESS OF THE       CSL03320
C     COMPUTATIONS.  THE AMOUNT OF INFORMATION PRINTED OUT IS           CSL03330
C     CONTROLLED BY THE PARAMETER IWRITE.                               CSL03340
C                                                                       CSL03350
C DESCRIPTION OF OTHER I/O FILES                                        CSL03360
C                                                                       CSL03370
C FILE (K)      CONTAINS:                                               CSL03380
C                                                                       CSL03390
C      (1)      OUTPUT FILE:                                            CSL03400
C               HISTORY FILE OF NEWLY-GENERATED T-MATRIX (ALPHA AND     CSL03410
C               BETA VECTORS) AND LAST 2 LANCZOS VECTORS USED           CSL03420
C               IN THE T-MATRIX GENERATION.                             CSL03430
C               IF IHIS = 0 AND ISTOP = 1, FILE 1 IS NOT WRITTEN.       CSL03440
C                                                                       CSL03450
C      (2)      INPUT FILE:                                             CSL03460
C               SAME AS FILE 1 EXCEPT THAT IT CONTAINS A                CSL03470
C               PREVIOUSLY-GENERATED T-MATRIX (IF ANY).  IF ISTART = 1, CSL03480
C               PROGRAM ASSUMES THAT THERE IS A HISTORY FILE OF ALPHAS  CSL03490
C               AND BETAS ON FILE 2.  THESE ALPHAS AND BETAS ARE        CSL03500
C               READ IN ALONG WITH THE LAST TWO LANCZOS VECTORS         CSL03510
C               USED IN THE T-MATRIX GENERATION.                        CSL03520
C                                                                       CSL03530
C      (3)      OUTPUT FILE:                                            CSL03540
C               COMPUTED GOOD EIGENVALUES OF THE T-MATRICES USED. ALSO  CSL03550
C               CONTAINS T-MULTIPLICITIES OF THESE EIGENVALUES AS       CSL03560
C               EIGENVALUES OF THE T-MATRIX, AND THEIR GAPS AS          CSL03570
C               EIGENVALUES IN THE A MATRIX AND IN THE T-MATRIX.        CSL03580
C               FILE 3 IS ALWAYS WRITTEN.                               CSL03590
C                                                                       CSL03600
C      (4)      OUTPUT FILE:                                            CSL03610
C               ERROR ESTIMATES FOR THE ISOLATED GOOD T-EIGENVALUES WHICHCSL03620
C               ARE OBTAINED USING THE SUBROUTINE INVERR. THESE         CSL03630
C               ESITMATES USE THE LAST COMPONENTS OF THE ASSOCIATED     CSL03640
C               T-EIGENVECTORS WHICH ARE COMPUTED USING INVERSE         CSL03650
C               ITERATION.  FILE 4 IS ALWAYS WRITTEN.                   CSL03660
C                                                                       CSL03670
C      (8)      INPUT FILE:                                             CSL03680
C               SAMPLE USPEC SUBROUTINE ASSUMES THAT THE ARRAYS         CSL03690
C               REQUIRED TO SPECIFY THE USER'S-MATRIX ARE STORED ON     CSL03700
C               FILE 8.  USERS MUST MAKE WHATEVER DEFINITIONS ARE       CSL03710
C               APPROPRIATE FOR THEIR MATRICES.                         CSL03720
C                                                                       CSL03730
C     (10)      OUTPUT OR INPUT FILE DEPENDING UPON VALUE OF SAVTEV:    CSL03740
C               COMPUTED EIGENVALUES OF EACH T(1,MEV) FOLLOWED          CSL03750
C               BY THE COMPUTED EIGENVALUES OF THE CORRESPONDING        CSL03760
C               T(2,MEV).                                               CSL03770
C                                                                       CSL03780
C     (11)      OUTPUT FILE:                                            CSL03790
C               COMPUTED DISTINCT EIGENVALUES OF T-MATRICES USED.       CSL03800
C               ALSO CONTAINS THEIR T-MULTIPLICITIES AND T-GAPS TO      CSL03810
```

```
C            NEAREST DISTINCT EIGENVALUES, AND THE T-MULTIPLICITY
C            PATTERN OF THE GOOD AND THE SPURIOUS T-EIGENVALUES.
C            FILE 11 IS WRITTEN ONLY IF IDIST = 1.
C
C
C-----PARAMETERS SET BY THE EIGENVALUE PROGRAM--------------------------
C
C     THESE PARAMETERS ARE SET INTERNALLY IN THE PROGRAM
C
C     SCALEK     K = 1,2,3,4
C
C              THE SCALING FACTORS SCALEK HAVE BEEN INTRODUCED IN AN
C              ATTEMPT TO MAKE THE TOLERANCES USED IN THE
C              T-MULTIPLICITY, SPURIOUS, AND ISOLATION TESTS ADJUST
C              TO THE SCALE OF THE GIVEN MATRIX.  THESE FACTORS MUST
C              NOT BE MODIFIED.
C
C     BTOL = RELATIVE TOLERANCE USED TO ESTIMATE ANY LOSS OF LOCAL
C          ORTHOGONALITY OF THE LANCZOS VECTORS AFTER THE T-MATRIX
C          HAS BEEN GENERATED.  THE LANCZOS PROCEDURE WORKS WELL
C          ONLY IF LOCAL ORTHOGONALITY BETWEEN SUCCESSIVE LANCZOS
C          VECTORS IS MAINTAINED.  THE TNORM SUBROUTINE TESTS
C          WHETHER OR NOT
C
C               MINIMUM  |BETA(I)|/||A|| > BTOL.
C               I=2,KMAX
C
C          IF THIS TEST IS VIOLATED BY SOME BETA AND A T-MATRIX THAT
C          WOULD INCLUDE SUCH A BETA IS REQUESTED, THEN THE LANCZOS
C          PROCEDURE WILL TERMINATE FOR THE USER TO DECIDE WHAT TO
C          DO.  THE USER CAN OVER-RIDE THIS TEST BY SIMPLY DECREASING
C          THE SIZE OF BTOL, BUT THEN CONVERGENCE IS NOT AS CERTAIN.
C          THE PROGRAM SETS BTOL = 1.D-8 WHICH IS A VERY CONSERVATIVE
C          CHOICE. THE || A || IS ESTIMATED BY USING
C          AN ESTIMATE OF THE NORM OF THE T-MATRIX, T(1,KMAX).
C
C     GAPTOL = RELATIVE TOLERANCE USED IN THE SUBROUTINE ISOEV
C            TO DETERMINE WHICH OF THE GOOD T-EIGENVALUES NEED
C            ERROR ESTIMATES.  THE PROGRAM SETS GAPTOL = 1.D-7.
C            IF FOR A GIVEN 'GOOD' T-EIGENVALUE THE COMPUTED GAP
C            IS TOO SMALL AND IS DUE TO A 'SPURIOUS' T-EIGENVALUE
C            THEN THE 'GOOD' T-EIGENVALUE IS ASSUMED TO HAVE CONVERGED
C            AND NO ERROR ESTIMATES ARE COMPUTED.
C
C-----USER-SPECIFIED PARAMETERS FOR EIGENVALUE PROGRAMS-----------------
C
C     RELTOL = RELATIVE TOLERANCE USED IN 'COMBINING' COMPUTED
C            EIGENVALUES OF T(1,MEV) PRIOR TO COMPUTING ERROR
C            ESTIMATES.
C
C     THE LUMPING OF T-EIGENVALUES OCCURS IN SUBROUTINE LUMP.
C     LUMPING IS NECESSARY BECAUSE IT IS IMPOSSIBLE TO ACCURATELY
C     PREDICT THE ACCURACY OF THE CMTQL1 SUBROUTINE.  LUMP 'COMBINES'
C     T-EIGENVALUES THAT HAVE SLIPPED BY THE TOLERANCE THAT WAS USED
C     IN THE T-MULTIPLICITY TESTS.  IN PARTICULAR IF FOR SOME J,
```

```
C
C   |EVALUE(J)-EVALUE(J-1)| < DMAX1(RELTOL*|EVALUE(J)|,SCALE2*MULTOL)
C
C     THEN THESE T-EIGENVALUES ARE 'COMBINED'.  MULTOL IS THE TOLERANCE
C     THAT WAS USED IN THE T-MULTIPLICITY TEST IN COMPEV.  SEE THE
C     HEADER ON THE LUMP SUBROUTINE FOR MORE DETAILS.
C
C     THE RECOMMENDED VALUE OF RELTOL (ONLY IN THE COMPLEX SYMMETRIC
C     CASE) IS 1.D-8 BECAUSE THE OBSERVED ACCURACY OF THE
C     COMPUTED EIGENVALUES OF THE T-MATRICES IS SEVERAL DIGITS
C     LESS THAN THAT OBSERVED IN THE REAL SYMMETRIC CASE.
C     THUS, THE OBSERVED RESOLUTION OF THE COMPLEX SYMMETRIC
C     VERSION IS LESS THAN THAT OBTAINABLE IN THE REAL SYMMETRIC CASE.
C
C     MXINIT = MAXIMUM NUMBER OF INVERSE ITERATIONS ALLOWED IN
C              SUBROUTINE INVERR FOR EACH ISOLATED GOOD T-EIGENVALUE.
C              TYPICALLY ONLY ONE ITERATION IS REQUIRED.
C
C     SEEDS FOR RANDOM NUMBER GENERATORS = INTEGER*4 SCALARS.
C
C               (1) SVSEED = SEED FOR STARTING VECTOR USED IN
C                   T-MATRIX GENERATION IN LANCZS SUBROUTINE
C
C               (2) RHSEED = SEED FOR RIGHT-HAND SIDE USED IN
C                   INVERSE ITERATION COMPUTATIONS IN INVERR.
C
C
C     T-MATRICES
C
C     SIZES OF T-MATRICES
C
C             (1) KMAX= MAXIMUM ORDER FOR T-MATRIX THAT USER IS WILLING
C                       TO CONSIDER.
C
C             (2) NMEVS = MAXIMUM NUMBER OF T-MATRICES THAT WILL BE
C                         CONSIDERED.
C
C             (3) NMEV(J)  (J=1,NMEVS)  = SIZES OF T-MATRIX TO BE
C                                         CONSIDERED SEQUENTIALLY.
C
C     T-MATRIX-GENERATION
C
C     USER SHOULD NOTE THAT THIS PROGRAM FIRST COMPUTES A T-MATRIX
C     OF ORDER KMAX AND THEN CYCLES THROUGH THE T-MATRICES SPECIFIED
C     A PRIORI BY THE USER, USING THE SUBROUTINE CMTQL1 TO COMPUTE THE
C     EIGENVALUES OF THE T-MATRICES.  THE EIGENVALUE COMPUTATION
C     FOR THE COMPLEX SYMMETRIC CASE WILL BE
C     CONSIDERABLY MORE EXPENSIVE THAN FOR THE REAL SYMMETRIC OR
C     HERMITIAN CASES BECAUSE WE DO NOT HAVE AN ANALOG OF
C     THE BISECTION SUBROUTINE FOR THE COMPLEX SYMMETRIC CASE.
C     THUS, ANY RECYCLING AND SUBSEQUENT ENLARGEMENT OF THE T-MATRIX
C     REQUIRES THE RECOMPUTATION OF ALL OF THE EIGENVALUES OF
C     THE RESULTING T-MATRIX.  WE CANNOT GO IN AND COMPUTE ONLY THOSE
C     T-EIGENVALUES ON SOME SUBINTERVAL OF THE SPECTRUM OF THE
C     T-MATRIX AS WE DID IN THE REAL SYMMETRIC AND HERMITIAN CASES.
```

```
C     OF COURSE, IF THE T-MATRICES BEING CONSIDERED ARE NOT
C     VERY LARGE, THEN THIS IS NOT REALLY A PROBLEM.  HOWEVER, IF THEY
C     ARE VERY LARGE, THEN THE USER SHOULD PROBABLY DO ONE EIGENVALUE
C     COMPUTATION OF A LARGE T-MATRIX RATHER THAN START WITH
C     A SMALLER T-MATRIX AND WORK UP TO A BIG ONE.
C
C-----CONVERGENCE TESTS FOR THE EIGENVALUE PROGRAMS---------------------
C
C     THE CONVERGENCE TEST INCORPORATED IN THIS PROGRAM IS
C     BASED UPON THE ASSUMPTION THAT THOSE T-EIGENVALUES AND
C     THEIR ASSOCIATED T-EIGENVECTORS WHICH CORRESPOND TO THE
C     EIGENVALUES AND RITZVECTORS WHICH ARE TO BE COMPUTED
C     CONVERGE AS THE T-SIZE IS INCREASED.
C
C-----ARRAYS REQUIRED BY THE EIGENVALUE PROGRAM-------------------------
C
C     ALPHA(J) = COMPLEX*16  ARRAY. ITS DIMENSION MUST BE AT LEAST
C                KMAX, THE LENGTH OF THE LARGEST T-MATRIX ALLOWED.
C                THIS ARRAY CONTAINS THE DIAGONAL ENTRIES OF THE
C                T-MATRICES GENERATED.
C
C     BETA(J) = COMPLEX*16 ARRAY.  ITS DIMENSION MUST BE AT LEAST
C               KMAX+1. THIS ARRAY CONTAINS THE SUBDIAGONAL ENTRIES OF
C               THE T-MATRICES.
C
C               THE ALPHA AND BETA VECTORS ARE NOT ALTERED
C               DURING THE CALCULATIONS.
C
C     V1(J),V2(J),VS(J) = COMPLEX*16 ARRAYS.  V1 AND V2
C                         MUST BE OF DIMENSION AT LEAST MAX(KMAX,N).
C                         VS MUST BE OF DIMENSION AT LEAST KMAX.
C
C     GR(J),GC(J) = REAL*8 ARRAYS. USED FOR RANDOM VECTOR GENERATION.
C                   EACH MUST BE OF DIMENSION AT LEAST MAX(KMAX,N).
C
C     EXPLAN(J) = REAL*4 ARRAY.  ITS DIMENSION IS 20.  THIS ARRAY IS
C                 USED TO ALLOW EXPLANATORY COMMENTS IN THE INPUT FILES.
C
C     G(J),GG(J) = REAL*4 ARRAYS.  G MUST BE OF DIMENSION AT LEAST
C                  MAX(N,KMAX).  GG MUST BE OF DIMENSION AT LEAST
C                  KMAX.  G AND GG ARE USED IN RANDOM VECTOR GENERATIONS
C                  AND TO STORE GAPS IN T-MATRIX, GAPS IN A-MATRIX,
C                  AND ERROR ESTIMATES.
C
C     MP(J),MP2(J) = INTEGER*4 ARRAYS.  EACH MUST HAVE DIMENSION
C                    AT LEAST KMAX, THE MAXIMUM SIZE OF THE T-MATRICES.
C                    MP CONTAINS THE T-MULTIPLICITIES OF THE COMPUTED
C                    T-EIGENVALUES. 'SPURIOUS' T-EIGENVALUES ARE DENOTED
C                    BY A T-MULTIPLICITY OF 0.  NOTE THAT WE DO NOT HAVE
C                    AN ANALOG OF THE SUBROUTINE PRTEST FOR THE
C                    COMPLEX SYMMETRIC CASE, SO NO RELABELLING OF
C                    MP OCCURS.  MP2 IS USED TO KEEP TRACK OF WHICH
C                    EIGENVALUES OF T(1,MEV) HAVE BEEN USED IN THE
C                    T-MULTIPLICITY TEST AND WHICH EIGENVALUES OF
C                    T(2,MEV) HAVE BEEN USED IN THE SPURIOUS TEST.
```

```
C
C     NMEV(J) = INTEGER*4 ARRAY.  ITS DIMENSION MUST BE AT LEAST THE
C              NUMBER OF T-MATRICES ALLOWED.  IT CONTAINS THE ORDERS
C              OF THE T-MATRICES TO BE CONSIDERED.
C
C     OTHER ARRAYS
C
C     THE USER MUST SPECIFY IN THE SUBROUTINE USPEC WHATEVER ARRAYS
C     ARE REQUIRED TO DEFINE THE MATRIX BEING USED.
C
C
C-----SUBROUTINES INCLUDED FOR EIGENVALUE COMPUTATIONS------------------
C
C     LANCZS = COMPUTES THE ALPHA/BETA HISTORY. USES SUBROUTINES
C             CINPRD, INPRDC, GENRAN, AND CMATV.
C
C     COMPEV = CALLS CMTQL1 TO COMPUTE THE EIGENVALUES OF T(1,MEV)
C             AND OF T(2,MEV), THEN DETERMINES T-MULTIPLE AND
C             SPURIOUS T-EIGENVALUES.
C
C     COMGAP = COMPUTES MINIMAL GAPS BETWEEN T-EIGENVALUES
C             SUPPLIED.
C
C     CMTQL1 = COMPUTES EIGENVALUES OF THE SPECIFIED T-MATRIX USING
C             A REAL ORTHOGONAL ANALOG OF THE QL ALGORITHM IMTQL1
C             IN EISPACK.
C
C     INVERR = USES INVERSE ITERATION ON T-MATRICES TO COMPUTE ERROR
C             ESTIMATES ON COMPUTED T-EIGENVALUES. (USES GENRAN)
C
C     LUMP = 'COMBINES' EIGENVALUES OF T-MATRIX USING THE RELATIVE
C            TOLERANCE RELTOL.
C
C     ISOEV = CALCULATES GAPS BETWEEN DISTINCT EIGENVALUES OF T-MATRIX
C            AND THEN USES THESE GAPS TO LABEL THOSE 'GOOD'
C            T-EIGENVALUES FOR WHICH ERROR ESTIMATES ARE NOT COMPUTED.
C
C     TNORM = COMPUTES THE SCALE TKMAX USED IN CHECKING
C            FOR LOCAL ORTHOGONALITY OF THE LANCZOS VECTORS
C            BY TESTING THE RELATIVE SIZE OF THE BETAS USING
C            THE RELATIVE TOLERANCE BTOL.
C
C    CINPRD = COMPUTES THE HERMITIAN INNER PRODUCT OF TWO
C            COMPLEX*16 VECTORS, USED IN SUBROUTINE INVERR
C            AND IN THE MAIN PROGRAM.
C
C    INPRDC = COMPUTES THE EUCLIDEAN INNER PRODUCT OF TWO
C            COMPLEX*16 VECTORS. USED IN SUBROUTINE LANCZS.
C
C
C-----OTHER PROGRAMS SUPPLIED-------------------------------------------
C
C
C     LCCOMPAC = PROGRAM TO TRANSLATE A SPARSE, COMPLEX SYMMETRIC
C               MATRIX GIVEN AS I, J, A(I,J), INTO THE SPARSE MATRIX
```

```
C              FORMAT USED IN THE SAMPLE USPEC AND CMATV SUBROUTINES  CSL06020
C              PROVIDED.  PROGRAM ASSUMES THAT THE MATRIX ENTRIES     CSL06030
C              ARE GIVEN EITHER COLUMN BY COLUMN OR ROW BY ROW.       CSL06040
C                                                                     CSL06050
C                                                                     CSL06060
C-----COMMENTS ON THE STORAGE REQUIRED FOR EIGENVALUE COMPUTATIONS------CSL06070
C                                                                     CSL06080
C     THE ARRAYS USED IN THIS EIGENVALUE PROGRAM USE THE EQUIVALENT OF  CSL06090
C     ONE REAL*8 ARRAY OF DIMENSION                                   CSL06100
C                                                                     CSL06110
C        8*KMAX + 4*MAX(KMAX,N)                                       CSL06120
C                                                                     CSL06130
C     PLUS WHATEVER IS NEEDED TO GENERATE A*X FOR THE GIVEN MATRIX A.   CSL06140
C     THE ARRAYS ALPHA, BETA, VS, G, GG, MP, AND MP2 CONSUME          CSL06150
C     8*KMAX*8 BYTES.  THE ARRAYS V1 AND V2 CONSUME                   CSL06160
C     4*MAXIMUM(KMAX,N)*8 BYTES.                                      CSL06170
C                                                                     CSL06180
C                                                                     CSL06190
C-----------------------------------------------------------------------CSL06200
C                                                                     CSL06210
C     COMMENTS FOR EIGENVECTOR COMPUTATIONS                           CSL06220
C                                                                     CSL06230
C-----------------------------------------------------------------------CSL06240
C                                                                     CSL06250
C                                                                     CSL06260
C     THE EIGENVALUES WHOSE EIGENVECTORS ARE TO BE COMPUTED MUST       CSL06270
C     HAVE BEEN COMPUTED USING THE CORRESPONDING LANCZOS EIGENVALUE    CSL06280
C     FILES:  CSLEVAL + CSLESUB + CSLEMULT, FOR COMPLEX SYMMETRIC      CSL06290
C     MATRICES BECAUSE THE EIGENVECTOR PROGRAMS WILL USE THE SAME      CSL06300
C     FAMILY OF LANCZOS TRIDIAGONAL MATRICES AND LANCZOS VECTORS       CSL06310
C     THAT WAS USED IN THE EIGENVALUE COMPUTATIONS.                   CSL06320
C                                                                     CSL06330
C     THESE PROGRAMS ASSUME THAT THE EIGENVALUES SUPPLIED TO IT        CSL06340
C     HAVE BEEN COMPUTED ACCURATELY, AS MEASURED BY THE               CSL06350
C     ERROR ESTIMATES COMPUTED IN THE CORRESPONDING LANCZOS           CSL06360
C     EIGENVALUE COMPUTATIONS, ALTHOUGH THESE ESTIMATES ARE           CSL06370
C     TYPICALLY CONSERVATIVE.  THE EIGENVALUES OF INTEREST            CSL06380
C     ARE IN THE ARRAY GOODEV(J), J=1,NGOOD.                          CSL06390
C                                                                     CSL06400
C     FOR EACH GOODEV(J), AN INITIAL ESTIMATE IS MADE OF AN           CSL06410
C     APPROPRIATE ORDER, MA(J), J=1,NGOOD, FOR A LANCZOS TRIDIAGONAL   CSL06420
C     FOR THE JTH EIGENVECTOR COMPUTATION.  THEN FOR EACH J,          CSL06430
C     SUBROUTINE INVERM SUCCESSIVELY COMPUTES CORRESPONDING           CSL06440
C     EIGENVECTORS OF ENLARGED T-MATRICES UNTIL A SUITABLE            CSL06450
C     SIZE T-MATRIX IS DETERMINED FOR EACH J.  UP TO 10 SUCH          CSL06460
C     EIGENVECTOR COMPUTATIONS ARE ALLOWED FOR EACH EIGENVALUE.       CSL06470
C                                                                     CSL06480
C     ONCE SUITABLE T-EIGENVECTORS HAVE BEEN OBTAINED THEN THE        CSL06490
C     RITZ VECTOR CORRESPONDING TO THESE T-EIGENVECTORS ARE           CSL06500
C     COMPUTED AND TAKEN AS APPROXIMATE EIGENVECTORS OF A FOR THE     CSL06510
C     GIVEN EIGENVALUES, GOODEV(J), J = 1, ..., NGOOD.                CSL06520
C                                                                     CSL06530
C     THIS IMPLEMENTATION FIRST COMPUTES ALL OF THE RELEVANT          CSL06540
C     EIGENVECTORS OF THE COMPLEX SYMMETRIC TRIDIAGONAL MATRICES      CSL06550
C     IN THE VECTOR, TVEC.                                            CSL06560
```

```
C
C     THEN, AS EACH OF THE LANCZOS VECTORS IS REGENERATED, ALL
C     OF THE RITZ VECTORS CORRESPONDING TO THESE
C     T-EIGENVECTORS ARE UPDATED USING THE CURRENTLY-GENERATED
C     LANCZOS VECTOR.  LANCZOS VECTORS ARE GENERATED (NOTE
C     THAT THEY ARE NOT BEING KEPT), UNTIL ENOUGH HAVE
C     BEEN GENERATED TO MAP THE LONGEST T-EIGENVECTOR INTO ITS
C     CORRESPONDING RITZ VECTOR.  THE ARRAY RITVEC CONTAINS THE
C     SUCCESSIVE RITZ VECTORS WHICH ARE THE APPROXIMATE
C     EIGENVECTORS OF A.
C
C
C-----PARAMETER CONTROLS FOR EIGENVECTOR PROGRAMS-----------------------
C
C
C     PARAMETER CONTROLS ARE INTRODUCED TO ALLOW SEGMENTATION OF THE
C     EIGENVECTOR COMPUTATIONS AND TO ALLOW VARIOUS COMBINATIONS OF
C     READ/WRITES.
C
C     THE FLAG MBOUND ALLOWS THE USER TO DETERMINE A FIRST GUESS ON THE
C     STORAGE THAT WILL BE REQUIRED BY THE T-EIGENVECTORS FOR THE
C     EIGENVALUES WHOSE EIGENVECTORS ARE TO BE COMPUTED.
C     THIS CAN BE USED TO ESTIMATE THE REQUIRED SIZE OF THE TVEC ARRAY.
C
C     MBOUND = (0,1) MEANS
C
C              (0)  PROGRAM COMPUTES FIRST GUESSES AT THE SIZES
C                   OF THE T-MATRICES REQUIRED BY EACH OF THE
C                   EIGENVALUES SUPPLIED AND THEN CONTINUES WITH
C                   THE CORRESPONDING T-EIGENVECTOR COMPUTATIONS.
C
C              (1)  PROGRAM COMPUTES FIRST GUESSES AT THE SIZES
C                   OF THE T-MATRICES REQUIRED BY EACH OF THE
C                   EIGENVALUES SUPPLIED, STORES THESE IN FILE 10
C                   AND THEN TERMINATES.  THE USER CAN USE THESE
C                   SIZES TO ESTIMATE THE SIZE TVEC ARRAY NEEDED
C                   FOR THE DESIRED T-EIGENVECTOR COMPUTATIONS.
C
C     THE FLAGS NTVCON, TVSTOP, LVCONT, AND ERCONT CONTROL THE STOPPING
C     CRITIERIA FOR INTERMEDIATE POINTS IN THE LANCZOS PROCEDURE.
C     THEY CAUSE TERMINATION OF THE LANCZOS PROCEDURE IF VARIOUS
C     QUANTITIES CANNOT BE COMPUTED AS DESIRED.
C
C     NTVCON = (0,1) MEANS
C
C              (0)  IF THE ESTIMATED STORAGE FOR THE T-EIGENVECTORS
C                   EXCEEDS THE USER-SPECIFIED DIMENSION OF THE
C                   TVEC ARRAY PROGRAM DOES NOT CONTINUE WITH THE
C                   T-EIGENVECTOR COMPUTATIONS. TERMINATION OCCURS.
C
C              (1)  CONTINUE WITH THE T-EIGENVECTOR COMPUTATIONS
C                   EVEN IF THE ESTIMATED STORAGE FOR TVEC EXCEEDS
C                   THE USER-SPECIFIED DIMENSION OF THE TVEC ARRAY.
C                   IN THIS SITUATION THE PROGRAM COMPUTES AS MANY
C                   T-EIGENVECTORS AS IT HAS ROOM FOR, IN THE SAME
```

```
C                   ORDER IN WHICH THE EIGENVALUES ARE PROVIDED.
C
C      SVTVEC = (0,1) MEANS
C
C               (0)  DO NOT STORE THE COMPUTED T-EIGENVECTORS ON
C                    FILE 11 UNLESS ALSO HAVE THE FLAG TVSTOP = 1,
C                    IN WHICH CASE THE T-EIGENVECTORS ARE ALWAYS
C                    WRITTEN TO FILE 11.
C
C               (1)  STORE THE COMPUTED T-EIGENVECTORS ON FILE 11.
C
C      TVSTOP = (0,1) MEANS
C
C               (0)  ATTEMPT TO CONTINUE ON TO THE COMPUTATION
C                    OF THE RITZVECTORS AFTER COMPLETING THE
C                    COMPUTATION OF THE T-EIGENVECTORS.
C
C               (1)  TERMINATE AFTER COMPUTING THE
C                    T-EIGENVECTORS AND STORING THEM ON FILE 11.
C
C      LVCONT = (0,1) MEANS
C
C               (0)  IF SOME OF THE T-EIGENVECTORS THAT WERE
C                    REQUIRED WERE NOT COMPUTED, EXIT
C                    FROM THE PROGRAM WITHOUT COMPUTING THE
C                    CORRESPONDING RITZ VECTORS.
C
C               (1)  CONTINUE ON TO THE RITZ VECTOR COMPUTATIONS
C                    EVEN IF NOT ALL OF THE T-EIGENVECTORS THAT
C                    WERE REQUESTED WERE COMPUTED.
C
C      ERCONT = (0,1) MEANS
C
C               (0)  PROGRAM WILL NOT COMPUTE THE RITZ
C                    VECTOR FOR ANY EIGENVALUE FOR WHICH NO
C                    T-EIGENVECTOR WHICH SATISFIES THE ERROR ESTIMATE
C                    TEST (ERTOL) HAS BEEN IDENTIFIED.
C
C               (1)  A RITZ VECTOR WILL BE COMPUTED FOR EVERY
C                    EIGENVALUE FOR WHICH A T-EIGENVECTOR HAS BEEN
C                    COMPUTED REGARDLESS OF WHETHER OR NOT THAT
C                    T-EIGENVECTOR SATISFIED THE ERROR ESTIMATE TEST.
C
C
C-----INPUT/OUTPUT FILES FOR THE EIGENVECTOR COMPUTATIONS---------------
C
C
C     INPUT DATA OTHER THAN THE T-MATRIX HISTORY FILE AND THE
C     EIGENVALUES AND ERROR ESTIMATES SUPPLIED SHOULD BE STORED ON
C     FILE 5 IN FREE FORMAT.  SEE SAMPLE INPUT/OUTPUT FOR TYPICAL
C     INPUT/OUTPUT FILE.
C
C     FILE 6 WAS USED AS THE INTERACTIVE TERMINAL OUTPUT FILE.
C     THIS FILE PROVIDES A RUNNING ACCOUNT OF THE PROGRESS OF THE
C     COMPUTATIONS.  ADDITIONAL PRINTOUT IS GENERATED WHEN
```

```
C     THE FLAG IWRITE = 1.
C
C
C DESCRIPTION OF OTHER I/O FILES
C
C FILE (K)     CONTAINS:
C
C      (2)     INPUT FILE:
C              PREVIOUSLY-GENERATED T-MATRICES (ALPHA/BETA ARRAYS)
C              AND THE FINAL TWO LANCZOS VECTORS USED ON THAT
C              COMPUTATION.  THIS PROGRAM ALLOWS ENLARGEMENT
C              OF ANY T-MATRICES PROVIDED ON FILE 2.
C
C      (3)     INPUT FILE:
C              THE GOOD EIGENVALUES OF THE T-MATRIX  T(1,MEV)
C              FOR WHICH EIGENVECTORS ARE REQUESTED.
C              FILE 3 ALSO CONTAINS THE T-MULTIPLICITIES OF THESE
C              EIGENVALUES AND THEIR COMPUTED GAPS IN THE
C              T-MATRICES AND IN THE USER-SUPPLIED MATRIX. THIS
C              FILE IS CREATED IN THE LANCZOS EIGENVALUE COMPUTATIONS.
C
C      (4)     INPUT FILE:
C              ERROR ESTIMATES FOR THE ISOLATED GOOD T-EIGENVALUES
C              IN FILE 3.  THIS FILE IS CREATED DURING THE LANCZOS
C              EIGENVALUE COMPUTATIONS.
C
C      (8)     INPUT FILE:
C              SAMPLE USPEC SUBROUTINE ASSUMES THAT THE ARRAYS
C              REQUIRED TO SPECIFY THE USER'S-MATRIX ARE STORED ON
C              FILE 8.  USERS MUST MAKE WHATEVER DEFINITIONS ARE
C              APPROPRIATE FOR THEIR MATRICES.
C
C      (9)     OUTPUT FILE:
C              ERROR ESTIMATES FOR THE COMPUTED RITZ VECTORS CONSIDERED
C              AS EIGENVECTORS OF THE ORIGINAL MATRIX. THESE ESTIMATES
C              ARE OF THE FORM
C                    AERROR = || A*RITVEC - EVAL*RITVEC ||
C              WHERE A DENOTES THE USER-SUPPLIED MATRIX, EVAL DENOTES
C              THE EIGENVALUE BEING CONSIDERED AND RITVEC DENOTES
C              THE COMPUTED RITZ VECTOR.
C
C      (10)    OUTPUT FILE:
C              GUESSES AT APPROPRIATE SIZE T-MATRICES FOR THE
C              T-EIGENVECTORS FOR EACH SUPPLIED EIGENVALUE GOODEV(J).
C
C      (11)    OUTPUT FILE:
C              COMPUTED T-EIGENVECTORS CORRESPONDING TO EIGENVALUES
C              IN THE GOODEV ARRAY.  NOTE THAT IT IS POSSIBLE IN
C              CERTAIN SITUATIONS THAT FOR SOME EIGENVALUES IN THE
C              GOODEV ARRAY A T-EIGENVECTOR WILL NOT BE COMPUTED.
C              (WRITTEN ONLY IF FLAG SVTVEC = 1).
C
C      (12)    OUTPUT FILE:
C              CONTAINS COMPUTED RITZ VECTORS CORRESPONDING TO
C              THE T-EIGENVECTORS ON FILE 11. NOTE THAT IN
```

```
C            SOME SITUATIONS THAT FOR SOME EIGENVALUES IN
C            THE GOODEV ARRAY FOR WHICH T-EIGENVECTORS HAVE
C            BEEN COMPUTED NO RITZ VECTOR WILL HAVE BEEN
C            COMPUTED.
C
C     (13)   OUTPUT FILE:
C            ADDITIONAL INFORMATION ABOUT THE BOUNDS AND ERROR
C            ESTIMATES OBTAINED.
C
C
C-----SEEDS FOR EIGENVECTOR PROGRAMS-----------------------------------
C
C     SEEDS FOR RANDOM NUMBER GENERATOR GENRAN
C              (1) SVSEED = INTEGER*4 SCALAR USED IN THE SUBROUTINE
C                           GENRAN TO GENERATE THE STARTING VECTOR FOR
C                           THE REGENERATION OF THE LANCZOS VECTORS.
C
C              (2) RHSEED = INTEGER*4 SCALAR USED IN THE SUBROUTINE
C                           GENRAN TO GENERATE A RANDOM VECTOR FOR
C                           USE IN SUBROUTINE INVERM.
C
C     USER SHOULD NOTE THAT SVSEED MUST BE THE SAME SEED THAT
C     WAS USED TO GENERATE THE T-MATRICES THAT WERE USED TO
C     COMPUTE THE EIGENVALUES WHOSE EIGENVECTORS ARE TO BE COMPUTED.
C     SVSEED IS READ IN FROM FILE 3.
C
C
C-----USER-SPECIFIED PARAMETERS FOR THE EIGENVECTOR PROGRAMS------------
C
C     NGOOD   = NUMBER OF EIGENVALUES READ INTO THE GOODEV ARRAY
C               READ FROM FILE 3.
C
C     N       = SIZE OF THE USER-SUPPLIED MATRIX.
C
C     MEV     = SIZE OF THE T-MATRIX THAT WAS USED TO COMPUTE
C               THE EIGENVALUES WHOSE EIGENVECTORS ARE REQUESTED.
C               MEV IS READ IN FROM FILE 3.
C
C     KMAX    =  SIZE OF THE T-MATRIX PROVIDED ON FILE 2.
C
C     MDIMTV  = MAXIMUM CUMULATIVE SIZE OF THE TVEC ARRAY ALLOWED
C               FOR ALL OF THE T-EIGENVECTORS REQUIRED.  MDIMTV
C               MUST NOT EXCEED THE USER-SPECIFIED DIMENSION OF
C               THE TVEC ARRAY.  PROGRAM CAN BE RUN WITH THE FLAG
C               MBOUND = 1 TO DETERMINE AN EDUCATED GUESS ON AN
C               APPROPRIATE DIMENSION FOR THE TVEC ARRAY.
C
C     MDIMRV = MAXIMUM CUMULATIVE SIZE OF THE RITVEC ARRAY ALLOWED
C              FOR ALL OF THE RITZ VECTORS TO BE COMPUTED. MDIMRV
C              MUST NOT EXCEED THE USER-SPECIFIED DIMENSION OF
C              THE RITVEC ARRAY.  MUST BE SELECTED SO THAT
C              THERE IS ENOUGH ROOM FOR A RITZ VECTOR FOR EVERY
C              GOODEV(J) READ INTO PROGRAM.  (>= NGOOD*N)
C
C
```

```
C-----ARRAYS REQUIRED BY THE EIGENVECTOR PROGRAMS-----------------------CSL08770
C                                                                       CSL08780
C                                                                       CSL08790
C     ALPHA(J) = COMPLEX*16 ARRAY WHOSE DIMENSION MUST BE AT LEAST      CSL08800
C                KMAXN, THE LARGEST SIZE T-MATRIX CONSIDERED BY         CSL08810
C                THE PROGRAM.  NOTE THAT KMAXN IS THE LARGER OF         CSL08820
C                THE SIZE OF THE ALPHA, BETA HISTORY PROVIDED           CSL08830
C                ON FILE 2 (IF ANY ) AND THE SIZE WHICH THE PROGRAM     CSL08840
C                SPECIFIES INTERNALLY, THIS LATTER IS ALWAYS            CSL08850
C                < = 11*MEV / 8  +  12, WHERE MEV IS THE SIZE           CSL08860
C                T-MATRIX THAT WAS USED IN THE CORRESPONDING EIGENVALUE CSL08870
C                COMPUTATIONS.  ALPHA CONTAINS THE DIAGONAL ENTRIES     CSL08880
C                OF THE LANCZOS T-MATRICES.  ALPHA IS NOT DESTROYED     CSL08890
C                IN THE COMPUTATIONS.                                   CSL08900
C                                                                       CSL08910
C     BETA(J) = COMPLEX*16 ARRAY WHOSE DIMENSION MUST BE AT LEAST 1     CSL08920
C               MORE THAN THAT OF ALPHA.  DIMENSION COMMENTS ABOVE      CSL08930
C               ABOUT ALPHA APPLY ALSO TO THE BETA ARRAY.  BETA         CSL08940
C               CONTAINS THE SUBDIAGONAL ENTRIES OF THE T-MATRICES.     CSL08950
C               BETA IS NOT DESTROYED IN THE COMPUTATIONS.              CSL08960
C                                                                       CSL08970
C     RITVEC(J) = COMPLEX*16 ARRAY WHOSE DIMENSION MUST BE AT LEAST     CSL08980
C                 NGOOD*N WHERE N IS THE ORDER OF THE USER-SUPPLIED     CSL08990
C                 MATRIX AND NGOOD IS THE NUMBER OF EIGENVALUES         CSL09000
C                 WHOSE EIGENVECTORS ARE TO BE COMPUTED.  IT CONTAINS   CSL09010
C                 THE COMPUTED RITZ VECTORS  (THE APPROXIMATE           CSL09020
C                 EIGENVECTORS OF A).  THESE VECTORS ARE STORED         CSL09030
C                 ON FILE 12.                                           CSL09040
C                                                                       CSL09050
C     TVEC(J)   = COMPLEX*16 ARRAY WHOSE DIMENSION MUST BE AT LEAST     CSL09060
C                 MTOL = |MA(1)| + |MA(2)| + ... + |MA(NGOOD)|          CSL09070
C                 WHERE NGOOD IS THE NUMBER OF EIGENVALUES BEING        CSL09080
C                 CONSIDERED AND |MA(J)| IS THE SIZE OF THE             CSL09090
C                 T-MATRIX BEING USED IN THE RITZ VECTOR COMPUTATIONS   CSL09100
C                 FOR GOODEV(J).  THESE SIZES ARE DETERMINED BY THE     CSL09110
C                 PROGRAM.  AN ESTIMATE OF MTOL CAN BE OBTAINED BY      CSL09120
C                 SETTING MBOUND = 1, RUNNING THE PROGRAM, AND          CSL09130
C                 MULTIPLYING THE RESULTING TOTAL T-SIZES BY 5/4.       CSL09140
C                 THE ARRAY TVEC IS USED TO HOLD THE COMPUTED           CSL09150
C                 T-EIGENVECTORS.  IF THE FLAG SVTVEC = 1 OR THE        CSL09160
C                 FLAG TVSTOP = 1, THESE VECTORS ARE SAVED ON FILE 11.  CSL09170
C                                                                       CSL09180
C     V1(J)     = COMPLEX*16 ARRAY WHOSE DIMENSION MUST BE AT LEAST     CSL09190
C                 MAX(KMAX,N)  WHERE KMAX IS THE                        CSL09200
C                 LARGEST SIZE T-MATRIX THAT CAN BE CONSIDERED          CSL09210
C                 IN THE T-EIGENVECTOR COMPUTATIONS.  V1 IS USED        CSL09220
C                 IN THE SUBROUTINE INVERM AND IN THE REGENERATION      CSL09230
C                 OF THE LANCZOS VECTORS.                               CSL09240
C                                                                       CSL09250
C     V2(J)     = COMPLEX*16 ARRAY WHOSE DIMENSION MUST BE AT LEAST     CSL09260
C                 MAX(KMAX,N). IT IS USED IN THE REGENERATION OF        CSL09270
C                 THE LANCZOS VECTORS AND IN THE SUBROUTINE INVERM.     CSL09280
C                                                                       CSL09290
C     GOODEV(J) = COMPLEX*16 ARRAY OF DIMENSION AT LEAST NGOOD.         CSL09300
C                 CONTAINS THE EIGENVALUES FOR WHICH EIGENVECTORS       CSL09310
```

```
C                 ARE REQUESTED. THESE EIGENVALUES ARE READ IN          CSL09320
C                 FROM FILE 3.                                          CSL09330
C                                                                       CSL09340
C     GR(J),GC(J)     = REAL*8 ARRAYS WHOSE DIMENSION MUST BE AT        CSL09350
C                       LEAST MAX(N,KMAX).  USED TO HOLD RANDOMLY-      CSL09360
C                       GENERATED STARTING VECTORS FOR LANCZS           CSL09370
C                       COMPUTATIONS AND FOR THE INVERM SUBROUTINE.     CSL09380
C                                                                       CSL09390
C     AMINGP(J), = REAL*4 ARRAYS OF DIMENSION AT LEAST NGOOD.           CSL09400
C     TMINGP(J)    CONTAIN, RESPECTIVELY, THE MINIMAL GAPS FOR          CSL09410
C                  CORRESPONDING EIGENVALUES IN GOODEV ARRAY IN         CSL09420
C                  A-MATRIX AND IN T-MATRIX.                            CSL09430
C                                                                       CSL09440
C     TERR(J), ERR(J),     = REAL*4 ARRAYS (EXCEPT TLAST WHICH IS       CSL09450
C     ERRDGP(J), TLAST(J)    REAL*8) EACH OF WHOSE DIMENSIONS MUST BE   CSL09460
C     RNORM(J), TBETA(J)     AT LEAST NGOOD.  USED TO STORE QUANTITIES  CSL09470
C                            GENERATED DURING THE COMPUTATIONS FOR      CSL09480
C                            LATER PRINTOUT.                            CSL09490
C                                                                       CSL09500
C     G(J)     = REAL*4 ARRAY WHOSE DIMENSION MUST BE AT LEAST          CSL09510
C                MAX(KMAX,N).  USED IN SUBROUTINE GENRAN TO HOLD        CSL09520
C                RANDOM NUMBERS NEEDED FOR THE LANCZOS VECTORS          CSL09530
C                REGENERATION AND FOR THE INVERSE ITERATION             CSL09540
C                COMPUTATIONS IN THE SUBROUTINE INVERM.                 CSL09550
C                                                                       CSL09560
C     MP(J) = INTEGER*4 ARRAY WHOSE DIMENSION IS AT LEAST NGOOD.        CSL09570
C             INITIALLY CONTAINS THE T-MULTIPLICITY OF THE EIGENVALUE   CSL09580
C             GOODEV(J) AS AN EIGENVALUE OF THE T-MATRIX T(1,MEV).      CSL09590
C             USED TO FLAG EIGENVALUES FOR WHICH NO T-EIGENVECTOR       CSL09600
C             OR NO RITZ VECTOR IS TO BE COMPUTED.                      CSL09610
C                                                                       CSL09620
C     MA(J)   = INTEGER*4 ARRAYS EACH OF WHOSE DIMENSIONS               CSL09630
C               IS AT LEAST NGOOD.  USED IN DETERMINING                 CSL09640
C               AN APPROPRIATE T-MATRIX FOR EACH EIGENVALUE             CSL09650
C               IN GOODEV ARRAY.                                        CSL09660
C                                                                       CSL09670
C     MINT(J),MFIN(J) = INTEGER*4 ARRAYS WHOSE DIMENSIONS MUST BE AT    CSL09680
C                       LEAST NGOOD.  USED TO POINT TO THE BEGINNINGS   CSL09690
C                       AND THE ENDS OF THE COMPUTED EIGENVECTOR        CSL09700
C                       OF THE T-MATRIX, T(1,|MA(J)|).                  CSL09710
C                                                                       CSL09720
C     IDELTA(J)  = INTEGER*4 ARRAY WHOSE DIMENSION MUST BE AT           CSL09730
C                  LEAST NGOOD.  CONTAINS INCREMENTS USED IN LOOPS      CSL09740
C                  ON APPROPRIATE SIZE T-MATRIX FOR THE T-EIGENVECTOR   CSL09750
C                  COMPUTATIONS.                                        CSL09760
C                                                                       CSL09770
C                                                                       CSL09780
C     INTERC(J)  = INTEGER*4 ARRAY WHOSE DIMENSION MUST BE AT           CSL09790
C                  LEAST KMAX.  WORK SPACE USED IN INVERM.              CSL09800
C                                                                       CSL09810
C-----SUBROUTINES INCLUDED FOR THE EIGENVECTOR COMPUTATIONS-------------CSL09820
C                                                                       CSL09830
C                                                                       CSL09840
C     INVERM = FOR THE T-SIZES CONSIDERED BY THE PROGRAM COMPUTES       CSL09850
C              THE CORRESPONDING EIGENVECTORS OF THESE T-MATRICES       CSL09860
```

```
C             CORRESPONDING TO THE USER-SUPPLIED EIGENVALUES IN        CSL09870
C             THE GOODEV ARRAY.                                        CSL09880
C                                                                      CSL09890
C     LANCZS, TNORM , CINPRD, INPRDC, CMATV AND GENRAN ARE USED        CSL09900
C     HERE AS WELL AS IN THE EIGENVALUE COMPUTATIONS.                  CSL09910
C                                                                      CSL09920
C                                                                      CSL09930
C-----------------------------------------------------------------------CSL09940
```

7.3 CSLEVAL: Main Program, Eigenvalue Computations

```
C-----CSLEVAL (EIGENVALUES OF COMPLEX SYMMETRIC MATRICES)---------------CSL00010
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)             CSL00020
C            Los Alamos National Laboratory                             CSL00030
C            Los Alamos, New Mexico 87544                               CSL00040
C                                                                       CSL00050
C            E-mail:  cullumj@lanl.gov                                  CSL00060
C                                                                       CSL00070
C  These codes are copyrighted by the authors.  These codes             CSL00080
C  and modifications of them or portions of them are NOT to be          CSL00090
C  incorporated into any commercial codes or used for any other         CSL00100
C  commercial purposes such as consulting for other companies,          CSL00110
C  without legal agreements with the authors of these Codes.            CSL00120
C  If these Codes or portions of them                                   CSL00130
C  are used in other scientific or engineering research works           CSL00140
C  the names of the authors of these codes and appropriate              CSL00150
C  references to their written work are to be incorporated in the       CSL00160
C  derivative works.                                                    CSL00170
C                                                                       CSL00180
C  This header is not to be removed from these codes.                   CSL00190
C                                                                       CSL00200
C        REFERENCE:  Cullum and Willoughby, Chapter 6,                  CSL00201
C        Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsCSL00202
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in    CSL00203
C        Applied Mathematics, 2002. SIAM Publications,                  CSL00204
C        Philadelphia, PA. USA                                          CSL00205
C                                                                       CSL00206
C                                                                       CSL00207
C     CONTAINS MAIN PROGRAM FOR COMPUTING DISTINCT EIGENVALUES OF       CSL00210
C     A NONDEFECTIVE COMPLEX SYMMETRIC MATRIX USING LANCZOS             CSL00220
C     TRIDIAGONALIZATION WITHOUT REORTHOGONALIZATION                    CSL00230
C                                                                       CSL00240
C     PORTABILITY:                                                      CSL00250
C     THESE PROGRAMS ARE NOT PORTABLE DUE TO THE USE OF COMPLEX*16      CSL00260
C     VARIABLES AND CORRESPONDING COMPLEX FUNCTIONS SUCH AS DCMPLX      CSL00270
C     AND CDABS.  FURTHERMORE, OTHER NONPORTABLE CONSTRUCTIONS          CSL00280
C     IDENTIFIED BY THE PFORT VERIFIER ARE THE FOLLOWING:               CSL00290
C                                                                       CSL00300
C     1.  DATA/MACHEP/ STATEMENT THAT DEFINES MACHINE EPSILON           CSL00310
C     2.  ALL READ(5,*) INPUT STATEMENTS IN FREE FORMAT                 CSL00320
C     3.  FORMAT(20A4) USED WITH EXPLANATORY HEADER EXPLAN.             CSL00330
C     4.  HEXADECIMAL FORMAT (4Z20) USED WITH ALPHA/BETA FILES 1 AND 2. CSL00340
C                                                                       CSL00350
C-----------------------------------------------------------------------CSL00360
C                                                                       CSL00370
      COMPLEX*16  ALPHA(3000),BETA(3000),VS(3000)                       CSL00380
      COMPLEX*16  V1(3000),V2(3000),ZEROC,BETAM,Z                       CSL00390
      DOUBLE PRECISION  GR(3000),GC(3000)                               CSL00400
      DOUBLE PRECISION  BTOL,GAPTOL,TTOL,MACHEP,EPSM,RELTOL             CSL00410
      DOUBLE PRECISION  SCALE1,SCALE2,SPUTOL,CONTOL,MULTOL,EVMAX        CSL00420
      DOUBLE PRECISION  ONE,ZERO,TEMP,TKMAX,BKMIN,T0,T1                 CSL00430
      REAL  G(3000),GG(3000),EXPLAN(20),GTEMP                           CSL00440
```

```
      INTEGER  MP(3000),MP2(3000),NMEV(20)                              CSL00450
      INTEGER  SVSEED,RHSEED,SVSOLD,SAVTEV                              CSL00460
      INTEGER IABS                                                      CSL00470
      REAL  ABS                                                         CSL00480
      DOUBLE PRECISION  DABS, DFLOAT                                    CSL00490
      EXTERNAL CMATV                                                    CSL00500
C                                                                       CSL00510
C-----------------------------------------------------------------------CSL00520
      DATA MACHEP/Z3410000000000000/                                    CSL00530
      EPSM = 2.0D0*MACHEP                                               CSL00540
C-----------------------------------------------------------------------CSL00550
C                                                                       CSL00560
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:                            CSL00570
C          1. ALPHA AND VS:  >= KMAX.   BETA:  >= (KMAX+1)              CSL00580
C          2. V1, V2, GR, GC:  >= MAX(N,KMAX)                           CSL00590
C          3. G:  >= MAX(N,KMAX).   GG:  >= KMAX.                       CSL00600
C          4. MP, MP2:   >= KMAX                                        CSL00610
C          5. NMEV:  >= NUMBER OF T-MATRICES ALLOWED                    CSL00620
C          6. EXPLAN:  DIMENSION IS 20.                                 CSL00630
C                                                                       CSL00640
C     NOTE:   THE OBSERVED ACHIEVABLE ACCURACY FOR THE COMPLEX          CSL00650
C     SYMMETRIC MATRICES TESTED WAS SIGNIFICANTLY LESS THAN THAT        CSL00660
C     OBTAINED WITH THE REAL SYMMETRIC AND HERMITIAN VERSIONS           CSL00670
C     OF THESE LANCZOS CODES AND IT IS DOUBTFUL THAT THIS CODE          CSL00680
C     CAN HANDLE VERY STIFF COMPLEX SYMMETRIC MATRICES.                 CSL00690
C                                                                       CSL00700
C     IMPORTANT TOLERANCES OR SCALES THAT ARE USED REPEATEDLY           CSL00710
C     THROUGHOUT THE PROGRAM ARE THE FOLLOWING:                         CSL00720
C     SCALED MACHINE EPSILON:  TTOL = EVMAX*EPSM WHERE                  CSL00730
C     EPSM = 2*MACHINE EPSILON AND                                      CSL00740
C     EVMAX = MAX(|LAMBDA(J)|), J =1,MEV OF EIGENVALUES OF T(1,MEV).    CSL00750
C     TOLERANCE: T-MULTIPLICITY TESTS:  MULTOL = 500*(1000+MEV)*TTOL    CSL00760
C     TOLERANCE: SPURIOUS TESTS   SPUTOL = MULTOL                       CSL00770
C     NOTE THAT IN THE MAIN PROGRAM THESE TOLERANCES ARE INITIALIZED    CSL00780
C     TO QUANTITIES THAT ARE NOT A FUNCTION OF THE SIZE OF THE          CSL00790
C     T-EIGENVALUES AND THEN THE SIZES OF THE T-EIGENVALUES ARE         CSL00800
C     INTRODUCED IN THE SUBROUTINE COMPEV.                              CSL00810
C                                                                       CSL00820
C     LANCZOS CONVERGENCE TOLERANCE: CONTOL = CDABS(BETA(MEV+1)*1.D-10  CSL00830
C-----------------------------------------------------------------------CSL00840
C     OUTPUT HEADER                                                     CSL00850
      WRITE(6,10)                                                       CSL00860
   10 FORMAT(/' LANCZOS EIGENVALUE PROCEDURE FOR COMPLEX SYMMETRIC MATRICSL00870
     1CES'/)                                                            CSL00880
C                                                                       CSL00890
C     SET PROGRAM PARAMETERS                                            CSL00900
C     SCALEK ARE USED IN TOLERANCES NEEDED IN SUBROUTINES LUMP          CSL00910
C     AND ISOEV.  USER MUST NOT MODIFY THESE SCALES.                    CSL00920
      SCALE1 = 5.0D2                                                    CSL00930
      SCALE2 = 5.0D0                                                    CSL00940
      ONE  = 1.0D0                                                      CSL00950
      ZERO = 0.0D0                                                      CSL00960
      ZEROC = DCMPLX(ZERO,ZERO)                                         CSL00970
      BTOL = 1.0D-8                                                     CSL00980
C     BTOL = MACHEP                                                     CSL00990
```

```
      GAPTOL = 1.0D-7
      ICONV = 0
      MOLD = 0
      MOLD1 = 1
      MMB = 0
C
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 5 (FREE FORMAT)
C
C     READ USER-PROVIDED HEADER FOR RUN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
      READ(5,20) EXPLAN
      WRITE(6,20) EXPLAN
   20 FORMAT(20A4)
C
C     READ ORDER OF MATRICES (N) , MAXIMUM ORDER OF T-MATRIX (KMAX),
C     NUMBER OF T-MATRICES ALLOWED (NMEVS), AND MATRIX IDENTIFICATION
C     NUMBERS (MATNO)
      READ(5,20) EXPLAN
      READ(5,*) N,KMAX,NMEVS,MATNO
C
C     READ SEEDS FOR LANCZS AND INVERR SUBROUTINES (SVSEED AND RHSEED)
C     READ MAXIMUM NUMBER OF ITERATIONS ALLOWED FOR EACH INVERSE
C     ITERATION (MXINIT).
      READ(5,20) EXPLAN
      READ(5,*) SVSEED,RHSEED,MXINIT
C
C     ISTART = (0,1):  ISTART = 0 MEANS ALPHA/BETA FILE IS NOT
C     AVAILABLE.  ISTART = 1 MEANS ALPHA/BETA FILE IS AVAILABLE ON
C     FILE 2.  COMPLEX SYMMETRIC HISTORIES MUST BE STORED
C     IN HEX FORMAT (4Z20).
C     ISTOP = (0,1):  ISTOP = 0 MEANS PROCEDURE GENERATES ALPHA/BETA
C     FILE AND THEN TERMINATES.  ISTOP = 1 MEANS PROCEDURE GENERATES
C     ALPHAS/BETAS IF NEEDED AND THEN COMPUTES EIGENVALUES AND ERROR
C     ESTIMATES AND THEN TERMINATES.
      READ(5,20) EXPLAN
      READ(5,*) ISTART,ISTOP
C
C     IHIS = (0,1):  IHIS = 0 MEANS ALPHA/BETA FILE IS NOT WRITTEN
C     TO FILE 1.  IHIS = 1 MEANS ALPHA/BETA FILE IS WRITTEN TO FILE 1.
C     IDIST = (0,1):  IDIST = 0 MEANS DISTINCT T(1,MEV)-EIGENVALUES
C     ARE NOT WRITTEN TO FILE 11.  IDIST = 1 MEANS DISTINCT
C     T(1,MEV)-EIGENVALUES ARE WRITTEN TO FILE 11.
C     SAVTEV = (-1,0,1):  SAVTEV = - 1 MEANS T(1,MEV) AND T(2,MEV)
C     EIGENVALUES ARE AVAILABLE ON FILE 10 FROM AN EARLIER RUN.
C     IN THIS CASE, ALPHA/BETA FILE FROM THAT RUN MUST BE
C     AVAILABLE ON FILE 2.
C     SAVTEV = 0 MEANS WE WILL NOT SAVE THE T(1,MEV) AND T(2,MEV)
C     EIGENVALUES.  SAVTEV = 1 MEANS WE WRITE THE T(1,MEV) AND
C     T(2,MEV) EIGENVALUES TO FILE 10.
C     IWRITE = (0,1):  IWRITE = 0 MEANS NO INTERMEDIATE OUTPUT
C     FROM THE COMPUTATIONS IS WRITTEN TO FILE 6.  IWRITE = 1 MEANS
C     EIGENVALUES AND ERROR ESTIMATES ARE WRITTEN TO FILE 6
C     AS THEY ARE COMPUTED.
      READ(5,20) EXPLAN
```

```
      READ(5,*) IHIS,IDIST,SAVTEV,IWRITE
C
      IF(SAVTEV.GE.0) GO TO 30
      NMEVS = 1
      IF(ISTART.EQ.0) GO TO 610
C
   30 CONTINUE
C     READ IN THE RELATIVE TOLERANCE (RELTOL) FOR USE IN THE LUMP
C     SUBROUTINE
      READ(5,20) EXPLAN
      READ(5,*) RELTOL
C
C     READ IN THE SIZES OF THE T(1,MEV) MATRICES TO BE CONSIDERED.
      READ(5,20) EXPLAN
      READ(5,*) (NMEV(J), J=1,NMEVS)
C
C-----------------------------------------------------------------------
C     INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED MATRIX
C     AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE
C     MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV.
C
      CALL USPEC(N,MATNO)
C
C-----------------------------------------------------------------------
C     MASK UNDERFLOW AND OVERFLOW
C
      CALL MASK
C
C-----------------------------------------------------------------------
C
C     WRITE TO FILE 6, A SUMMARY OF THE PARAMETERS FOR THIS RUN
C
      WRITE(6,40) MATNO,N,KMAX
   40 FORMAT(/3X,'MATRIX ID',4X,'ORDER OF A',4X,'MAX ORDER OF T'/
     1 I12,I14,I18/)
C
      WRITE(6,50) ISTART,ISTOP
   50 FORMAT(/2X,'ISTART',3X,'ISTOP'/2I8/)
C
      WRITE(6,60) IHIS,IDIST,SAVTEV,IWRITE
   60 FORMAT(/4X,'IHIS',3X,'IDIST',3X,'SAVTEV',2X,'IWRITE'/2I8,I9,I8/)
C
      WRITE(6,70) SVSEED,RHSEED
   70 FORMAT(/' SEEDS FOR RANDOM NUMBER GENERATOR'//
     1 4X,'LANCZS SEED',4X,'INVERR SEED'/2I15/)
C
      WRITE(6,80) (NMEV(J), J=1,NMEVS)
   80 FORMAT(/' SIZES OF T-MATRICES TO BE CONSIDERED'/(6I12))
C
      WRITE(6,90) RELTOL,GAPTOL,BTOL
   90 FORMAT(/' RELATIVE TOLERANCE USED TO COMBINE COMPUTED T-EIGENVALUE
     1S'/E15.3/' RELATIVE GAP TOLERANCES USED IN INVERSE ITERATION'/
     1E15.3/' RELATIVE TOLERANCE FOR CHECK ON SIZE OF BETAS'/E15.3/)
C
      IF (ISTART.EQ.0) GO TO 140
```

```
C
C     READ IN ALPHA BETA HISTORY
C     HISTORY MUST BE STORED IN MACHINE FORMAT TO PREVENT
C     ERRORS CAUSED BY INPUT/OUTPUT CONVERSIONS.
C
      READ(2,100)MOLD,NOLD,SVSOLD,MATOLD
  100 FORMAT(2I6,I12,I8)
C
      IF (KMAX.LT.MOLD) KMAX = MOLD
      KMAX1 = KMAX + 1
C
C     CHECK THAT ORDER N, MATRIX ID MATNO, AND RANDOM SEED SVSEED
C     AGREE WITH THOSE IN THE HISTORY FILE.  IF NOT PROCEDURE STOPS.
C
      ITEMP = (NOLD-N)**2+(MATNO-MATOLD)**2+(SVSEED-SVSOLD)**2
C
      IF (ITEMP.EQ.0) GO TO 120
C
      WRITE(6,110)
  110 FORMAT(' PROGRAM TERMINATES'/    ' READ FROM FILE 2 CORRESPONDS TO
     1 DIFFERENT MATRIX THAN MATRIX SPECIFIED'/)
      GO TO 650
C
  120 CONTINUE
      MOLD1 = MOLD+1
C
      READ(2,130)(ALPHA(J), J=1,MOLD)
      READ(2,130)(BETA(J), J=1,MOLD1)
  130 FORMAT(4Z20)
C
      IF (KMAX.EQ.MOLD) GO TO 160
C
      READ(2,130)(V1(J), J=1,N)
      READ(2,130)(V2(J), J=1,N)
C
  140 CONTINUE
      IIX = SVSEED
C
C-----------------------------------------------------------------------
C
      CALL LANCZS(CMATV,V1,V2,ALPHA,BETA,GR,GC,G,KMAX,MOLD1,N,IIX)
C
C-----------------------------------------------------------------------
C
      KMAX1 = KMAX + 1
C
      IF (IHIS.EQ.0.AND.ISTOP.GT.0) GO TO 160
C
      WRITE(1,150) KMAX,N,SVSEED,MATNO
  150 FORMAT(2I6,I12,I8,' = KMAX,N,SVSEED,MATNO')
C
      WRITE(1,130)(ALPHA(I), I=1,KMAX)
      WRITE(1,130)(BETA(I), I=1,KMAX1)
C
      WRITE(1,130)(V1(I), I=1,N)
```

```
      WRITE(1,130)(V2(I), I=1,N)
C
      IF (ISTOP.EQ.0) GO TO 520
C
  160 CONTINUE
      BKMIN = BTOL
      WRITE(6,170)
  170 FORMAT(/' T-MATRICES (ALPHA AND BETA) ARE NOW AVAILABLE'/)
C
C-----------------------------------------------------------------------
C     SUBROUTINE TNORM CHECKS MIN|BETA|/(ESTIMATED NORM(A)) > BTOL .
C     IF THIS IS VIOLATED IB IS SET EQUAL TO THE NEGATIVE OF THE INDEX
C     OF THE MINIMAL BETA.  IF(IB < 0) THEN SUBROUTINE TNORM IS
C     CALLED FOR EACH VALUE OF MEV TO DETERMINE WHETHER OR NOT THERE
C     IS A BETA IN THE T-MATRIX SPECIFIED THAT VIOLATES THIS TEST.
C     IF THERE IS SUCH A BETA THE PROGRAM TERMINATES FOR THE USER
C     TO DECIDE WHAT TO DO.  THIS TEST CAN BE OVER-RIDDEN BY
C     SIMPLY MAKING BTOL SMALLER, BUT THEN THERE IS THE POSSIBILITY
C     THAT LOSSES IN THE LOCAL ORTHOGONALITY MAY HURT THE COMPUTATIONS.
C
C     TNORM ALSO COMPUTES TKMAX = MAX(|ALPHA(K)|,|BETA(K)|, K=1,KMAX).
C     HOWEVER, IN THE COMPLEX SYMMETRIC CASE SINCE ALL OF THE
C     EIGENVALUES OF T(1,MEV) ARE COMPUTED, TKMAX IS NOT USED TO SCALE
C     THE T-MULTIPLICITY AND SPURIOUS TOLERANCES.  THE COMPUTED
C     T-EIGENVALUE LARGEST IN MAGNITUDE IS USED INSTEAD.
C
      CALL TNORM(ALPHA,BETA,BKMIN,TKMAX,KMAX,IB)
C
C-----------------------------------------------------------------------
C
C     LOOP ON THE SIZE OF THE T-MATRIX
C
  180 CONTINUE
      MMB = MMB + 1
      MEV = NMEV(MMB)
C     IS MEV TOO LARGE ?
      IF(MEV.LE.KMAX) GO TO 200
      WRITE(6,190) MMB, MEV, KMAX
  190 FORMAT(/' TERMINATE PRIOR TO CONSIDERING THE',I6,'TH T-MATRIX'/
     1' BECAUSE THE SIZE REQUESTED',I6,' IS GREATER THAN THE MAXIMUM SIZ
     1E ALLOWED',I6/)
      GO TO 520
C
  200 MP1 = MEV + 1
      BETAM = BETA(MP1)
C
      IF (IB.GE.0) GO TO 220
C
      T0 = BTOL
C
C-----------------------------------------------------------------------
C
      CALL TNORM(ALPHA,BETA,T0,T1,MEV,IBMEV)
C
C-----------------------------------------------------------------------
```

```
C
  210 TEMP = T0/TKMAX
      IBMEV = IABS(IBMEV)
      IF (TEMP.GE.BTOL) GO TO 220
      IBMEV = -IBMEV
      GO TO 590
  220 CONTINUE
C
C-----------------------------------------------------------------------
C     SUBROUTINE COMPEV CALLS SUBROUTINE CMTQL1 TO COMPUTE THE
C     T-EIGENVALUES.  COMPEV THEN APPLIES THE T-MULTIPLICITY AND
C     SPURIOUS TESTS TO THE COMPUTED T-EIGENVALUES.  HERE INITIALIZE
C     THE TOLERANCES USED IN THE T-MULTIPLICITY AND THE SPURIOUS
C     TESTS.  THE MAX(|LAMBDA(T(1,MEV)|) WILL BE INCORPORATED
C     INSIDE THE SUBROUTINE COMPEV.  NOTE THAT THE OBSERVED ACCURACY
C     OF THE COMPUTED T-EIGENVALUES FOR THE COMPLEX SYMMETRIC CASE
C     IS APPROXIMATELY 3 DIGITS LESS THAN THAT ACHIEVED IN THE REAL
C     CASE. THUS, A FACTOR OF 500 HAS BEEN INTRODUCED.  THIS HOWEVER
C     MEANS THAT THIS TEST IS NOT AS SHARP AS IT WAS IN THE
C     REAL SYMMETRIC AND HERMITIAN CASES.  THUS, IT HAS LOWER
C     RESOLUTION AND CAN OCCASIONALLY MAKE A MISTAKE.
C
      MULTOL = 500.D0 * DFLOAT(MEV+1000) * EPSM
      SPUTOL =  MULTOL
C
C     ON RETURN FROM COMPEV
C     NDIS = NUMBER OF DISTINCT EIGENVALUES OF T(1,MEV)
C     VS = DISTINCT T-EIGENVALUES IN INCREASING ORDER OF MAGNITUDE
C     GR(K) = |VS(K)|, K = 1,NDIS,  GR(K).LE.GR(K+1)
C     MP = T-MULTIPLICITIES OF THE T-EIGENVALUES IN VS
C     MP(I) = (0,1,MI), MI>1, I=1,NDIS  MEANS:
C        (0)  VS(I) IS SPURIOUS
C        (1)  VS(I) IS SIMPLE AND GOOD
C        (MI) VS(I) IS T-MULTIPLE AND IS THEREFORE NOT ONLY GOOD BUT
C             ALSO A CONVERGED GOOD T-EIGENVALUE.
C
C
      CALL COMPEV(ALPHA,BETA,V1,V2,VS,GR,MULTOL,SPUTOL,MP,MP2,
     1MEV,NDIS,SAVTEV)
C
C-----------------------------------------------------------------------
C
      IF (NDIS.EQ.0) GO TO 630
C
C     ON EXIT FROM COMPEV MULTOL AND SPUTOL SHOULD BE SCALED
C     BY THE SIZES OF THE T-EIGENVALUES
      EVMAX = GR(NDIS)
      LOOP = NDIS
C
C-----------------------------------------------------------------------
C
      CALL LUMP(VS,V1,GR,RELTOL,SPUTOL,SCALE2,MP,MP2,LOOP)
C
C-----------------------------------------------------------------------
C
```

```
      IF (LOOP.LT.0) GO TO 650                                          CSL03750
C                                                                       CSL03760
      IF (NDIS.EQ.LOOP) GO TO 240                                       CSL03770
C                                                                       CSL03780
      WRITE(6,230) NDIS,LOOP,MEV                                        CSL03790
  230 FORMAT(/' AFTER LUMP NDIS,LOOP,MEV = ',3I6/)                      CSL03800
C                                                                       CSL03810
  240 CONTINUE                                                          CSL03820
      NDIS = LOOP                                                       CSL03830
C                                                                       CSL03840
C-----------------------------------------------------------------------CSL03850
C     CALCULATE MINGAPS FOR DISTINCT T(1,MEV) EIGENVALUES.              CSL03860
C     ON EXIT |GG(K)| = MIN(J.NE.K,|VS(K)-VS(J)|), MP2(K)=J INDEX       CSL03870
C     FOR MINIMUM.  GG(K)< 0 MEANS NEAREST NEIGHBOR IS SPURIOUS.        CSL03880
      IGAP = 0                                                          CSL03890
      ITAG = 1                                                          CSL03900
C                                                                       CSL03910
      CALL COMGAP(VS,GR,GG,MP,MP2,NDIS,IGAP,ITAG)                       CSL03920
C                                                                       CSL03930
C-----------------------------------------------------------------------CSL03940
C                                                                       CSL03950
C     SET CONVERGENCE CRITIERION                                        CSL03960
      TTOL = EPSM * EVMAX                                               CSL03970
      CONTOL = CDABS(BETAM)*1.D-10                                      CSL03980
C                                                                       CSL03990
  250 CONTINUE                                                          CSL04000
      BETA(MP1) = BETAM                                                 CSL04010
C                                                                       CSL04020
C-----------------------------------------------------------------------CSL04030
C     THE SUBROUTINE ISOEV LABELS THOSE SIMPLE EIGENVALUES OF T(1,MEV)  CSL04040
C     WITH VERY SMALL GAPS BETWEEN NEIGHBORING EIGENVALUES OF T(1,MEV)  CSL04050
C     TO AVOID COMPUTING ERROR ESTIMATES FOR ANY SIMPLE GOOD            CSL04060
C     T-EIGENVALUE THAT IS TOO CLOSE TO A SPURIOUS T-EIGENVALUE.        CSL04070
C     MP(I) = -1 MEANS THAT THE GOOD T-EIGENVALUE IS SIMPLE AND         CSL04080
C     IS TOO CLOSE TO A SPURIOUS T-EIGENVALUE.                          CSL04090
C                                                                       CSL04100
C     NG = NUMBER OF GOOD T-EIGENVALUES.                                CSL04110
C     NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES.                     CSL04120
C     GG = MINIMAL GAPS IN T(1,MEV)                                     CSL04130
C     GR(K) = |VS(K)|, K=1,NDIS                                         CSL04140
C                                                                       CSL04150
      CALL ISOEV(VS,GR,GG,GAPTOL,SPUTOL,SCALE1,MP,NDIS,NG,NISO)         CSL04160
C                                                                       CSL04170
C-----------------------------------------------------------------------CSL04180
C                                                                       CSL04190
      WRITE(6,260)NG,NISO,NDIS                                          CSL04200
  260 FORMAT(/I6,' GOOD T-EIGENVALUES HAVE BEEN COMPUTED'/              CSL04210
     1 I6,' OF THESE ARE ISOLATED'/                                     CSL04220
     2 I6,' = NUMBER OF DISTINCT T-EIGENVALUES COMPUTED'/)              CSL04230
C                                                                       CSL04240
C     DO WE WRITE DISTINCT EIGENVALUES OF T-MATRIX TO FILE 11?          CSL04250
      IF (IDIST.EQ.0) GO TO 300                                         CSL04260
C                                                                       CSL04270
      WRITE(11,270) NDIS,NISO,MEV,N,SVSEED,MATNO                        CSL04280
  270 FORMAT(/4I6,I12,I8,' = NDIS,NISO,MEV,N,SVSEED,MATNO'/)            CSL04290
```

```
C
      WRITE(11,280) (I,MP(I),VS(I),GG(I),MP2(I), I=1,NDIS)
  280 FORMAT(I4,I4,2E20.12,E12.3,I6)
C
      WRITE(11,290) NDIS, (MP(I), I=1,NDIS)
  290 FORMAT(/I6,' = NDIS, T-MULTIPLICITIES (0 MEANS  SPURIOUS)'/(20I4))
C
  300 CONTINUE
C
      IF (NISO.NE.0) GO TO 330
C
      WRITE(4,310) MEV
  310 FORMAT(/' AT MEV = ',I6,' THERE ARE NO ISOLATED T-EIGENVALUES'/
     1' SO NO ERROR ESTIMATES WERE COMPUTED/')
C
      WRITE(6,320)
  320 FORMAT(/' ALL COMPUTED GOOD T-EIGENVALUES ARE T-MULTIPLE'/
     1 ' THEREFORE THESE EIGENVALUES ARE ASSUMED TO HAVE CONVERGED')
C
      ICONV = 1
      GO TO 370
C
  330 CONTINUE
C
C-----------------------------------------------------------------------
C     SUBROUTINE INVERR COMPUTES ERROR ESTIMATES FOR ISOLATED GOOD
C     T-EIGENVALUES USING INVERSE ITERATION ON T(1,MEV). ON RETURN
C     GG(J) = MINIMUM GAP IN T(1,MEV) FOR EACH VS(J), J=1,NDIS
C     G(I) = |BETAM|*|U(MEV)| = ERROR ESTIMATE FOR ISOLATED GOOD
C              T-EIGENVALUES, WHERE I = 1, NISO AND  BETAM = BETA(MEV+1)
C              U(MEV) IS MEVTH COMPONENT OF THE UNIT EIGENVECTOR OF T
C              CORRESPONDING TO THE ITH ISOLATED GOOD T-EIGENVALUE.
C     A NEGATIVE ERROR ESTIMATE MEANS THAT FOR THAT PARTICULAR
C     T-EIGENVALUE THE INVERSE ITERATION DID NOT CONVERGE IN <= MXINIT
C     STEPS AND THAT THE CORRESPONDING ERROR ESTIMATE IS QUESTIONABLE.
C
C     ON EXIT
C     V2 CONTAINS THE ISOLATED GOOD T-EIGENVALUES
C     GR CONTAINS THE MINGAPS TO THE NEAREST DISTINCT EIGENVALUE
C        OF T(1,MEV) FOR EACH ISOLATED GOOD T-EIGENVALUE IN V2.
C     VS CONTAINS THE NDIS DISTINCT EIGENVALUES OF T(1,MEV)
C     MP CONTAINS THE CORRESPONDING CODED T-MULTIPLICITIES
C
      IT = MXINIT
C
      CALL INVERR(ALPHA,BETA,V1,V2,VS,EPSM,GR,GC,G,GG,MP,MP2,MEV,MMB,
     1NDIS,NISO,N,RHSEED,IT,IWRITE)
C
C-----------------------------------------------------------------------
C
C     SIMPLE CHECK FOR CONVERGENCE. CHECKS TO SEE IF ALL OF THE ERROR
C     ESTIMATES ARE SMALLER THAN CONTOL = CDABS(BETA(MEV+1)*1.D-10
C     IF THIS TEST IS SATISFIED, THEN CONVERGENCE FLAG, ICONV IS SET
C     TO 1.  TYPICALLY ERROR ESTIMATES ARE VERY CONSERVATIVE.
C
```

```
      WRITE(6,340) CONTOL
  340 FORMAT(/' CONVERGENCE IS TESTED USING THE CONVERGENCE TOLERANCE',
     1E13.4/)
C
      DO 350 I = 1,NISO
      IF (ABS(G(I)).GT.CONTOL) GO TO 370
  350 CONTINUE
      ICONV = 1
      MMB = NMEVS
C
      WRITE(6,360) CONTOL
  360 FORMAT(' ALL COMPUTED ERROR ESTIMATES WERE LESS THAN',E15.4/
     1 ' THEREFORE PROCEDURE TERMINATES'/)
C
  370 CONTINUE
C
C     IN REAL SYMMETRIC AND HERMITIAN LANCZOS PROGRAMS
C     AT THIS CORRESPONDING POINT THE SUBROUTINE PRTEST IS CALLED
C     TO IDENTIFY ANY T-EIGENVALUES THAT MAY HAVE BEEN MISLABELLED
C     AS SPURIOUS BECAUSE THEIR PROJECTIONS ON THE STARTING VECTOR
C     WERE TOO SMALL.  THIS CHECK WAS MADE ONLY AFTER CONVERGENCE
C     HAD OCCURRED.  HOWEVER, THE PRTEST SUBROUTINE IS BASED UPON
C     STURM SEQUENCING AND THAT IS NOT VALID FOR COMPLEX SYMMETRIC
C     MATRICES.  PERHAPS THERE IS SOME RECTANGLE ANALOG OF THE
C     PRTEST BUT WE HAVE NOT ATTEMPTED TO IDENTIFY AND INCLUDE
C     SUCH A TEST BECAUSE WE EXPECT, AS IN THE REAL SYMMETRIC AND
C     HERMITIAN CASES THAT HIDDEN EIGENVALUES WILL BE RARE.
C
C     WRITE THE GOOD T-EIGENVALUES TO FILE 3.  FIRST TRANSFER THEM
C     TO V2 AND THEIR T-MULTIPLICITIES TO THE CORRESPONDING POSITIONS
C     IN MP AND COMPUTE THE A-MINGAPS, THE MINIMAL GAPS BETWEEN THE
C     GOOD T-EIGENVALUES.  THESE GAPS WILL BE PUT IN THE ARRAY GG.
C     NOTE THAT AFTER THE SECOND CALL TO COMGAP THE ARRAY GC
C     WILL CONTAIN THE CORRESPONDING MINIMAL GAPS IN THE
C     T-MATRIX, T(1,MEV).
C
  380 CONTINUE
C
      NG = 0
      DO 390 I = 1,NDIS
      IF (MP(I).EQ.0) GO TO 390
      NG = NG+1
      MP(NG) = MP(I)
      V2(NG) = VS(I)
      GC(NG) = GG(I)
  390 CONTINUE
C
      DO 400 I = 1,NG
  400 GR(I) = CDABS(V2(I))
C
C-----------------------------------------------------------------------
C     CALCULATE MINGAPS FOR GOODEV
C     ON EXIT GG(K) = MIN(J.NE.K,|V2(K)-V2(J)|), MP2(K)=J INDEX FOR MIN
C     NG = NUMBER OF COMPUTED DISTINCT GOOD T-EIGENVALUES.
      IGAP = 0
```

```
      ITAG = 0                                                          CSL05400
C                                                                       CSL05410
      CALL COMGAP(V2,GR,GG,MP,MP2,NG,IGAP,ITAG)                         CSL05420
C                                                                       CSL05430
C-----------------------------------------------------------------------CSL05440
C                                                                       CSL05450
C     WRITE GOOD T-EIGENVALUES OUT TO FILE 3.                           CSL05460
C                                                                       CSL05470
      WRITE(6,410)MEV                                                   CSL05480
  410 FORMAT(//' EIGENVALUE CALCULATION AT MEV = ',I6,'  IS COMPLETE'/) CSL05490
C                                                                       CSL05500
      WRITE(3,420)NG,NDIS,MEV,N,SVSEED,MATNO,MULTOL,SPUTOL,IB,BTOL      CSL05510
  420 FORMAT(4I6,I12,I8,' = NG,NDIS,MEV,N,SVEED,MATNO'/                 CSL05520
     1 2E15.5,I6,E13.4,' = MULTOL,SPUTOL,IB,BTOL'/                      CSL05530
     1' EVNO',1X,'MULT',13X,'R(GOODEV)',13X,'I(GOODEV)',                CSL05540
     1 3X,'TMINGAP',3X,'AMINGAP',1X,'NEIGH')                            CSL05550
C                                                                       CSL05560
      WRITE(3,430)(I,MP(I),V2(I),GC(I),GG(I),MP2(I), I=1,NG)            CSL05570
  430 FORMAT(2I5,2E22.14,2E10.3,I6)                                     CSL05580
C                                                                       CSL05590
C     ORDER GOODEV BY INCREASING GAP SIZE                               CSL05600
      DO 440 I = 1,NG                                                   CSL05610
      MP(I) = I                                                         CSL05620
      V1(I) = V2(I)                                                     CSL05630
      G(I) = GG(I)                                                      CSL05640
  440 CONTINUE                                                          CSL05650
C                                                                       CSL05660
C     WRITE(12,436)                                                     CSL05670
  450 FORMAT(' MINGAPS FOR GOOD T-EIGENVALUES'/                         CSL05680
     1 1X,'EVNUM',1X,'NEIGH',15X,'R(EV)',15X,'I(EV)',4X,'MINGAP')       CSL05690
C                                                                       CSL05700
C     WRITE(12,439) (K,MP2(K),V2(K),G(K), K = 1,NG)                     CSL05710
  460 FORMAT(2I6,2E20.12,E10.3)                                         CSL05720
C                                                                       CSL05730
      DO 480 K = 2,NG                                                   CSL05740
      KM1 = K-1                                                         CSL05750
      DO 470 L = 1,KM1                                                  CSL05760
      KK = K-L                                                          CSL05770
      KP1 = KK+1                                                        CSL05780
      IF (G(KP1).GE.G(KK)) GO TO 480                                    CSL05790
      Z = V1(KK)                                                        CSL05800
      V1(KK) = V1(KP1)                                                  CSL05810
      V1(KP1) = Z                                                       CSL05820
      GTEMP = G(KK)                                                     CSL05830
      G(KK) = G(KP1)                                                    CSL05840
      G(KP1) = GTEMP                                                    CSL05850
      ITEMP = MP(KK)                                                    CSL05860
      MP(KK) = MP(KP1)                                                  CSL05870
      MP(KP1) = ITEMP                                                   CSL05880
  470 CONTINUE                                                          CSL05890
  480 CONTINUE                                                          CSL05900
C                                                                       CSL05910
C     WRITE(12,441)                                                     CSL05920
      WRITE(3,490)                                                      CSL05930
  490 FORMAT(' T-EIGENVALUES ORDERED BY INCREASING MINGAP'/             CSL05940
```

```
     1 1X,'GAPNUM',1X,'EVNUM',15X,'R(EV)',15X,'I(EV)',4X,'MINGAP')      CSL05950
C                                                                       CSL05960
C     WRITE(12,442) (K,MP(K),V1(K),G(K), K = 1,NG)                      CSL05970
      WRITE(3,500) (K,MP(K),V1(K),G(K), K = 1,NG)                       CSL05980
  500 FORMAT(I7,I6,2E20.12,E10.3)                                       CSL05990
C                                                                       CSL06000
  510 CONTINUE                                                          CSL06010
C                                                                       CSL06020
C     IF CONVERGENCE FLAG ICONV.NE.1 AND NUMBER OF T-MATRICES           CSL06030
C     CONSIDERED TO DATE IS LESS THAN NUMBER ALLOWED, INCREMENT MEV.    CSL06040
C     AND LOOP BACK TO 210 TO REPEAT COMPUTATIONS.  RESTORE BETA(MEV+1).CSL06050
C                                                                       CSL06060
      BETA(MP1) = BETAM                                                 CSL06070
C                                                                       CSL06080
      IF (MMB.LT.NMEVS.AND.ICONV.NE.1) GO TO 180                        CSL06090
C                                                                       CSL06100
C     END OF LOOP ON DIFFERENT SIZE T-MATRICES ALLOWED.                 CSL06110
C                                                                       CSL06120
  520 CONTINUE                                                          CSL06130
C                                                                       CSL06140
      IF(ISTOP.EQ.0)  WRITE(6,530)                                      CSL06150
  530 FORMAT(/' T-MATRICES (ALPHA AND BETA) ARE NOW AVAILABLE, TERMINATECSL06160
     1')                                                                CSL06170
      IF (ISTOP.EQ.0.AND.KMAX.NE.MOLD) WRITE(1,540)                     CSL06180
      IF (IHIS.EQ.1.AND.KMAX.NE.MOLD) WRITE(1,540)                      CSL06190
  540 FORMAT(/' ABOVE ARE THE FOLLOWING VECTORS '/                      CSL06200
     1 '  ALPHA(I), I = 1,KMAX'/                                        CSL06210
     2 '  BETA(I), I = 1,KMAX+1'/                                       CSL06220
     3 ' FINAL TWO LANCZOS VECTORS OF ORDER N FOR I = KMAX,KMAX+1'/     CSL06230
     4 ' ALPHA BETA ARE IN HEX FORMAT 4Z20 '/                           CSL06240
     4 ' LANCZOS VECTORS ARE IN HEX FORMAT 4Z20 '/                      CSL06250
     5 ' ----- END OF FILE 1 NEW ALPHA, BETA HISTORY---------------'///)CSL06260
C                                                                       CSL06270
      IF (ISTOP.EQ.0) GO TO 650                                         CSL06280
C                                                                       CSL06290
      WRITE(3,550)                                                      CSL06300
  550 FORMAT(/' ABOVE ARE COMPUTED GOOD T-EIGENVALUES'/                 CSL06310
     1 ' NG = NUMBER OF GOOD T-EIGENVALUES COMPUTED'/                   CSL06320
     2 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)'/   CSL06330
     3 ' N = ORDER OF A,  MATNO = MATRIX IDENT'/                        CSL06340
     4 ' MULTOL = T-MULTIPLICITY TOLERANCE FOR T-EIGENVALUES'/          CSL06350
     4 ' SPUTOL = SPURIOUS TOLERANCE FOR T-EIGENVALUES'/                CSL06360
     4 ' MULT IS THE T-MULTIPLICITY OF GOOD T-EIGENVALUE'/              CSL06370
     5 ' MULT = -1 MEANS SPURIOUS T-EIGENVALUE TOO CLOSE'/              CSL06380
     6 ' DO NOT COMPUTE ERROR ESTIMATES FOR SUCH T-EIGENVALUES'/        CSL06390
     7 ' AMINGAP = MINIMAL GAP BETWEEN THE COMPUTED A-EIGENVALUES'/     CSL06400
     9 ' TMINGAP= MINIMAL GAP W.R.T.  DISTINCT EIGENVALUES IN T(1,MEV)'/CSL06410
     2 ' ----- END OF FILE 3 GOOD T-EIGENVALUES---------------------'///CSL06420
     3 )                                                                CSL06430
C                                                                       CSL06440
      IF (IDIST.NE.0) WRITE(11,560)                                     CSL06450
  560 FORMAT(/' ABOVE ARE THE DISTINCT EIGENVALUES OF T(1,MEV).'/       CSL06460
     2 ' THE FORMAT IS      T-MULTIPLICITY    T-EIGENVALUE   TMINGAP'/  CSL06470
     4 ' T-MULTIPLICITY = -1 MEANS THAT THE SUBROUTINE ISOEV HAS TAGGED'CSL06480
     5 /'   THIS SIMPLE T-EIGENVALUE AS HAVING A VERY CLOSE SPURIOUS'/  CSL06490
```

```
     6 '   T-EIGENVALUE SO THAT NO ERROR ESTIMATE WILL BE COMPUTED'/    CSL06500
     7 '   FOR THAT EIGENVALUE IN SUBROUTINE INVERR.'/                  CSL06510
     9 ' EACH OF THE DISTINCT T-EIGENVALUE TABLES IS FOLLOWED'/         CSL06520
     9 ' BY THE T-MULTIPLICITY PATTERN.'/                               CSL06530
     1 ' NDIS = NUMBER OF COMPUTED DISTINCT EIGENVALUES OF T(1,MEV).'/  CSL06540
     2 ' NG = NUMBER OF GOOD T-EIGENVALUES. '/                          CSL06550
     3 ' NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES. '/               CSL06560
     4 ' NISO ALSO IS THE COUNT OF +1 ENTRIES IN MULTIPLICITY PATTERN.'/CSL06570
     5 ' -----END OF FILE 11 DISTINCT T-EIGENVALUES----------------'///)CSL06580
C                                                                       CSL06590
      WRITE(4,570)                                                      CSL06600
  570 FORMAT(/' ABOVE ARE THE ERROR ESTIMATES OBTAINED FOR THE ISOLATED CSL06610
     1GOOD T-EIGENVALUES'/                                              CSL06620
     1' OBTAINED VIA INVERSE ITERATION IN THE SUBROUTINE INVERR.'/      CSL06630
     1' ALL OTHER GOOD T-EIGENVALUES HAVE CONVERGED.'/                  CSL06640
     2' ERROR ESTIMATE = CDABS(BETAM*(UM))'/                            CSL06650
     2' WHERE BETAM = BETA(MEV+1) AND UM = U(MEV).'/                    CSL06660
     3' U = UNIT EIGENVECTOR OF T WHERE T*U = EV*U AND EV = ISOLATED GOOCSL06670
     3D T-EIGENVALUE.'/                                                 CSL06680
     4' TMINGAP = GAP TO NEAREST DISTINCT EIGENVALUE OF T(1,MEV).'/     CSL06690
     6' ------ END OF FILE 4 ERRINV -------------------------------'///)CSL06700
C                                                                       CSL06710
      IF(SAVTEV.LT.0) GO TO 650                                         CSL06720
      WRITE(10,580)                                                     CSL06730
  580 FORMAT(//' ABOVE ARE THE T(1,MEV) EIGENVALUES FOLLOWED BY THE'/   CSL06740
     1 ' T(2,MEV) EIGENVALUES FOR MEV = NMEV(J), J = 1,NMEVS'/          CSL06750
     1 ' ------END OF FILE 10 T-T2EVAL----------------------'///)       CSL06760
C                                                                       CSL06770
      GO TO 650                                                         CSL06780
C                                                                       CSL06790
  590 CONTINUE                                                          CSL06800
C                                                                       CSL06810
      IBB = IABS(IBMEV)                                                 CSL06820
      TEMP = CDABS(BETA(IBB))                                           CSL06830
      IF (IBMEV.LT.0) WRITE(6,600) MEV,IBB,TEMP                         CSL06840
  600 FORMAT(/' PROGRAM TERMINATES BECAUSE MEV REQUESTED = ',I6,' IS .GTCSL06850
     1',I6/' AT WHICH AN ABNORMALLY SMALL BETA = ' ,E13.4,' OCCURRED'/) CSL06860
      GO TO 650                                                         CSL06870
C                                                                       CSL06880
  610 WRITE(6,620) SAVTEV,ISTART                                        CSL06890
  620 FORMAT(2I6,' = SAVTEV,ISTART'/' WHEN SAVTEV = -1, WE MUST HAVE ISTCSL06900
     1ART = 1'/)                                                        CSL06910
      GO TO 650                                                         CSL06920
C                                                                       CSL06930
  630 IF (NDIS.EQ.0.AND.ISTOP.GT.0) WRITE(6,640)                        CSL06940
  640 FORMAT(/' INTERVALS SPECIFIED FOR BISECT DID NOT CONTAIN ANY T-EIGCSL06950
     1ENVALUES'/' PROGRAM TERMINATES')                                  CSL06960
C                                                                       CSL06970
  650 CONTINUE                                                          CSL06980
C                                                                       CSL06990
      STOP                                                              CSL07000
C-----END OF MAIN PROGRAM FOR COMPLEX SYMMETRIC EIGENVALUE COMPUTATIONS-CSL07010
      END                                                               CSL07020
```

7.4 CSLEVEC: Main Program, Eigenvector Computations

```
C-----CSLEVEC (EIGENVECTORS OF COMPLEX SYMMETRIC MATRICES)--------------CSL00010
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)             CSL00020
C            Los Alamos National Laboratory                             CSL00030
C            Los Alamos, New Mexico 87544                               CSL00040
C                                                                       CSL00050
C            E-mail:  cullumj@lanl.gov                                  CSL00060
C                                                                       CSL00070
C  These codes are copyrighted by the authors.  These codes             CSL00080
C  and modifications of them or portions of them are NOT to be          CSL00090
C  incorporated into any commercial codes or used for any other         CSL00100
C  commercial purposes such as consulting for other companies,          CSL00110
C  without legal agreements with the authors of these Codes.            CSL00120
C  If these Codes or portions of them                                   CSL00130
C  are used in other scientific or engineering research works           CSL00140
C  the names of the authors of these codes and appropriate              CSL00150
C  references to their written work are to be incorporated in the      CSL00160
C  derivative works.                                                    CSL00170
C                                                                       CSL00180
C  This header is not to be removed from these codes.                   CSL00190
C                                                                       CSL00200
C         REFERENCE:  Cullum and Willoughby, Chapter 6,                 CSL00201
C         Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsCSL00202
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   CSL00203
C         Applied Mathematics, 2002. SIAM Publications,                 CSL00204
C         Philadelphia, PA. USA                                         CSL00205
C                                                                       CSL00206
C                                                                       CSL00207
C     CONTAINS MAIN PROGRAM FOR COMPUTING AN EIGENVECTOR CORRESPONDING  CSL00210
C     TO EACH OF A SET OF EIGENVALUES THAT HAVE BEEN COMPUTED           CSL00220
C     ACCURATELY BY THE CORRESPONDING LANCZOS EIGENVALUE PROGRAM        CSL00230
C     (CSLEVAL) FOR NONDEFECTIVE COMPLEX SYMMETRIC MATRICES.            CSL00240
C     THIS PROGRAM COULD BE MODIFIED TO COMPUTE ADDITIONAL              CSL00250
C     EIGENVECTORS FOR THOSE EIGENVALUES WHICH ARE MULTIPLE EIGENVALUES CSL00260
C     OF THE GIVEN A-MATRIX.  THE AMOUNT OF ADDITIONAL COMPUTATION      CSL00270
C     REQUIRED WOULD DEPEND UPON THE GIVEN A-MATRIX AND UPON WHAT       CSL00280
C     PART OF THE SPECTRUM OF A IS INVOLVED.                            CSL00290
C                                                                       CSL00300
C     THESE LANCZOS EIGENVECTOR COMPUTATIONS ASSUME THAT EACH           CSL00310
C     EIGENVALUE THAT IS BEING CONSIDERED HAS CONVERGED AS AN           CSL00320
C     EIGENVALUE OF THE CORRESPONDING LANCZOS TRIDIAGONAL MATRICES.     CSL00330
C                                                                       CSL00340
C     PORTABILITY:                                                      CSL00350
C     THIS PROGRAM IS NOT PORTABLE DUE TO THE USE OF THE COMPLEX*16     CSL00360
C     VARIABLES AND CORRESPONDING COMPLEX FUNCTIONS.  MOREOVER, PFORT   CSL00370
C     IDENTIFIED THE FOLLOWING ADDITIONAL NONPORTABLE CONSTRUCTIONS:    CSL00380
C                                                                       CSL00390
C     1.  DATA/MACHEP/ STATEMENT                                        CSL00400
C     2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                        CSL00410
C     3.  FORMAT(20A4) USED WITH THE EXPLANATORY HEADER, EXPLAN         CSL00420
C     4.  FORMAT (4Z20) USED FOR ALPHA/ BETA FILE 2.                    CSL00430
C                                                                       CSL00440
```

```
C     IMPORTANT NOTE:  PROGRAM ALLOWS ENLARGEMENT OF THE ALPHA,BETA
C     ARRAYS.  IN PARTICULAR, IF ANY ONE OF THE EIGENVALUES SUPPLIED
C     IS T-SIMPLE AND NOT CLOSE TO A SPURIOUS T-EIGENVALUE, THE PROGRAM
C     REQUIRES THAT KMAX BE AT LEAST 11*MEV/8 + 12.  IF KMAX IS NOT
C     THIS LARGE, THEN THE PROGRAM WILL RESET KMAX TO THIS SIZE
C     AND EXTEND THE ALPHA, BETA HISTORY IF REQUIRED.
C     THUS, THE DIMENSIONS OF THE ALPHA AND BETA ARRAYS MUST BE
C     LARGE ENOUGH TO ALLOW FOR THIS POSSIBILITY.
C     REMEMBER THAT THE BETA ARRAY, BETA(J), IS SUCH THAT
C     J = 1,..., KMAX+1.  SO IF THE KMAX USED BY THE PROGRAM
C     IS TO BE 3000, THEN BETA MUST BE OF LENGTH AT LEAST 3001.
C
C-----------------------------------------------------------------------
      COMPLEX*16  V1(1600),V2(1600),RITVEC(10000),ZEROC,TEMPC
      COMPLEX*16  ALPHA(1600),BETA(1601),GOODEV(50),TVEC(20000)
      COMPLEX*16  EVAL,ALFA,BATA,SUMC
      DOUBLE PRECISION  GR(1600),GC(1600)
      DOUBLE PRECISION  ERTOL,SUM,TEMP,BKMIN
      DOUBLE PRECISION  MULTOL,SPUTOL,SCALE0,BTOL
      DOUBLE PRECISION  ONE,ZERO,MACHEP,EPSM
      DOUBLE PRECISION  RELTOL,ERROR,ERRMIN,TERROR,TLAST(50)
      REAL  G(1600),AMINGP(50),TMINGP(50),EXPLAN(20)
      REAL  TERR(50),ERR(50),ERRDGP(50),RNORM(50),TBETA(50)
      INTEGER  MP(50),MA(50),ML(50),MINT(50),MFIN(50),IDELTA(50)
      INTEGER  SVSEED,SVSOLD,RHSEED
      INTEGER INTERC(1600)
      INTEGER  MBOUND,NTVCON,SVTVEC,TVSTOP,LVCONT,ERCONT,TFLAG
      DOUBLE PRECISION  DABS, DMAX1, DSQRT
      REAL ABS
      INTEGER  IABS
C-----------------------------------------------------------------------
      EXTERNAL CMATV
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.D0*MACHEP
C-----------------------------------------------------------------------
C
C     ARRAYS MUST BE DIMENSIONED AS FOLLOWS:
C     1.  ALPHA:  >= KMAXN,  BETA: >= (KMAXN+1) WHERE KMAXN, THE
C               LARGEST SIZE T-MATRIX CONSIDERED BY THE PROGRAM,
C               IS THE LARGER OF THE SIZE OF THE ALPHA, BETA HISTORY
C               PROVIDED ON FILE 2 (IF ANY ) AND THE SIZE WHICH THE
C               PROGRAM SPECIFIES INTERNALLY, THIS LATTER IS ALWAYS
C               < = 11*MEV / 8  +  12, WHERE MEV IS THE SIZE
C               T-MATRIX THAT WAS USED IN THE CORRESPONDING EIGENVALUE
C               COMPUTATIONS.
C     2.  V1:  >= MAX(N,KMAX)
C     3.  V2:  >= N
C     4.  G, GR, GC:  >= MAX(N,KMAX)
C     5.  RITVEC:  >= N*NGOOD, WHERE NGOOD IS THE NUMBER OF EIGENVALUES
C                 SUPPLIED TO THIS PROGRAM.
C     6.  TVEC:  >= CUMULATIVE LENGTH OF ALL THE T-EIGENVECTORS NEEDED
C                  TO GENERATE THE DESIRED RITZ VECTORS.  AN EDUCATED
C                  GUESS AT AN APPROPRIATE LENGTH CAN BE OBTAINED
C                  BY RUNNING THE PROGRAM WITH THE FLAG MBOUND = 1
C                  AND MULTIPLYING THE RESULTING SIZE BY 5/4.
```

```
C     7.  INTERC:   >= KMAX                                             CSL01000
C     8.  GOODEV, AMINGP, TMINGP, TERR, ERR, ERRDGP, RNORM, TBETA,      CSL01010
C         TLAST, MP, MA, MINT, MFIN, AND IDELTA :    >= NUMBER OF       CSL01020
C         EIGENVALUES SUPPLIED.                                         CSL01030
C                                                                       CSL01040
C     OUTPUT HEADER                                                     CSL01050
      WRITE(6,10)                                                       CSL01060
   10 FORMAT(/' LANCZOS EIGENVECTOR PROCEDURE FOR COMPLEX SYMMETRIC MATRCSL01070
     1ICES'/)                                                           CSL01080
C                                                                       CSL01090
C     SET PROGRAM PARAMETERS                                            CSL01100
C     USER MUST NOT MODIFY SCALE0                                       CSL01110
      SCALE0 = 5.0D0                                                    CSL01120
      ZERO = 0.0D0                                                      CSL01130
      ZEROC = DCMPLX(ZERO,ZERO)                                         CSL01140
      ONE = 1.0D0                                                       CSL01150
      MPMIN = -1000                                                     CSL01160
      MONE = -1                                                         CSL01170
C     CONVERGENCE TOLERANCE FOR T-EIGENVECTORS FOR RITZ COMPUTATIONS    CSL01180
      ERTOL = 1.D-10                                                    CSL01190
C-----------------------------------------------------------------------CSL01200
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 5 (FREE FORMAT)    CSL01210
C                                                                       CSL01220
C     READ USER-PROVIDED HEADER FOR RUN                                 CSL01230
      READ(5,20) EXPLAN                                                 CSL01240
      WRITE(6,20) EXPLAN                                                CSL01250
   20 FORMAT(20A4)                                                      CSL01260
C                                                                       CSL01270
C     READ IN THE MAXIMUM PERMISSIBLE DIMENSIONS FOR THE TVEC ARRAY     CSL01280
C     (MDIMTV), FOR THE RITVEC ARRAY (MDIMRV), AND FOR THE BETA         CSL01290
C     ARRAY (MBETA).                                                    CSL01300
C                                                                       CSL01310
      READ(5,20) EXPLAN                                                 CSL01320
      READ(5,*) MDIMTV, MDIMRV, MBETA                                   CSL01330
C                                                                       CSL01340
C     READ IN RELATIVE TOLERANCE (RELTOL) USED IN DETERMINING           CSL01350
C     APPROPRIATE SIZES FOR THE T-MATRICES USED IN THE RITZ             CSL01360
C     VECTOR COMPUTATIONS                                               CSL01370
C                                                                       CSL01380
      READ(5,20) EXPLAN                                                 CSL01390
      READ(5,*) RELTOL                                                  CSL01400
C                                                                       CSL01410
C     SET FLAGS TO 0 OR 1:                                              CSL01420
C     MBOUND = 1:  PROGRAM TERMINATES AFTER COMPUTING 1ST GUESSES       CSL01430
C                  ON APPROPRIATE T-SIZES FOR USE IN THE RITZ VECTOR    CSL01440
C                  COMPUTATIONS                                         CSL01450
C     NTVCON = 0:  PROGRAM TERMINATES IF THE TVEC ARRAY IS NOT          CSL01460
C                  LARGE ENOUGH TO HOLD ALL THE T-EIGENVECTORS REQUIRED.CSL01470
C     SVTVEC = 0:  THE T-EIGENVECTORS ARE NOT WRITTEN TO FILE 11        CSL01480
C                  UNLESS TVSTOP = 1                                    CSL01490
C     SVTVEC = 1:  WRITE THE T-EIGENVECTORS TO FILE 11.                 CSL01500
C     TVSTOP = 1:  PROGRAM TERMINATES AFTER COMPUTING THE               CSL01510
C                  T-EIGENVECTORS                                       CSL01520
C     LVCONT = 0:  PROGRAM TERMINATES IF THE NUMBER OF T-EIGENVECTORS   CSL01530
C                  COMPUTED IS NOT EQUAL TO THE NUMBER OF RITZ          CSL01540
```

```
C                VECTORS REQUESTED.
C     ERCONT = 0:  MEANS FOR ANY GIVEN EIGENVALUE, A RITZ VECTOR
C                  WILL NOT BE COMPUTED FOR THAT EIGENVALUE UNLESS
C                  A T-EIGENVECTOR HAS BEEN IDENTIFIED WITH A LAST
C                  COMPONENT WHICH SATISFIES THE SPECIFIED
C                  CONVERGENCE CRITERION.
C     ERCONT = 1:  MEANS FOR ANY GIVEN EIGENVALUE, A RITZ VECTOR
C                  WILL BE COMPUTED.  IF A T-EIGENVECTOR CANNOT
C                  BE IDENTIFIED WHICH SATISFIES THE LAST
C                  COMPONENT CRITERION, THEN THE PROGRAM WILL
C                  USE THE T-VECTOR THAT CAME CLOSEST TO
C                  SATISFYING THE CRITERION
C     IWRITE = 1:  EXTENDED OUTPUT OF INTERMEDIATE COMPUTATIONS
C                  IS WRITTEN TO FILE 6
C     IREAD = 0:   ALPHA/BETA FILE IS REGENERATED.
C     IREAD = 1:   ALPHA/BETA FILE USED IN EIGENVALUE COMPUTATIONS
C                  IS READ IN AND EXTENDED IF NECESSARY.  IN BOTH
C                  CASES IREAD = 0 OR 1, THE LANCZOS VECTORS ARE
C                  ALWAYS REGENERATED FOR THE RITZ VECTOR
C                  COMPUTATIONS
C
      READ(5,20) EXPLAN
      READ(5,*) MBOUND,NTVCON,SVTVEC,IREAD
C
      READ(5,20) EXPLAN
      READ(5,*) TVSTOP,LVCONT,ERCONT,IWRITE
      IF (TVSTOP.EQ.1) SVTVEC = 1
C
C     READ IN SEED (RHSEED) FOR GENERATING RANDOM STARTING VECTOR
C     FOR INVERSE ITERATION ON THE T-MATRICES.
C
      READ(5,20) EXPLAN
      READ(5,*) RHSEED
C
C     READ IN MATNO = MATRIX/RUN IDENTIFICATION NUMBER AND
C     N = ORDER OF A-MATRIX
C
      READ(5,20) EXPLAN
      READ(5,*) MATNO,N
C
C-----------------------------------------------------------------------
C     INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED MATRIX
C     AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS TO THE
C     MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV.
C
      CALL USPEC(N,MATNO)
C
C-----------------------------------------------------------------------
C
C     MASK UNDERFLOW AND OVERFLOW
      CALL MASK
C
C-----------------------------------------------------------------------
C     WRITE RUN PARAMETERS OUT TO FILE 6
C
```

```
      WRITE(6,30) MATNO,N
   30 FORMAT(/' MATRIX IDENTIFICATION NO. = ',I10,' ORDER OF A = ',I5)
C
      WRITE(6,40) MBOUND,NTVCON,SVTVEC,IREAD
   40 FORMAT(/3X,'MBOUND',3X,'NTVCON',3X,'SVTVEC',3X,'IREAD'/3I9,I8)
C
      WRITE(6,50) TVSTOP,LVCONT,ERCONT,IWRITE
   50 FORMAT(/3X,'TVSTOP',3X,'LVCONT',3X,'ERCONT',3X,'IWRITE'/4I9)
C
      WRITE(6,60) MDIMTV,MDIMRV,MBETA
   60 FORMAT(/3X,'MDIMTV',3X,'MDIMRV',3X,'MBETA'/2I9,I8)
C
      WRITE(6,70) RELTOL,RHSEED
   70 FORMAT(/7X,'RELTOL',3X,'RHSEED'/E13.4,I9)
C
C
C     FROM FILE 3 READ IN THE NUMBER OF EIGENVALUES (NGOOD) FOR WHICH
C     EIGENVECTORS ARE REQUESTED, THE ORDER (MEV) OF THE LANCZOS
C     TRIDIAGONAL MATRIX USED IN COMPUTING THESE EIGENVALUES, THE
C     ORDER (NOLD) OF THE USER-SPECIFIED MATRIX USED IN THE EIGENVALUE
C     COMPUTATIONS, THE SEED (SVSEED) USED FOR GENERATING THE STARTING
C     VECTOR THAT WAS USED IN THOSE LANCZOS EIGENVALUE COMPUTATIONS,
C     AND THE MATRIX/RUN IDENTIFICATION NUMBER (MATOLD) USED IN THOSE
C     COMPUTATIONS.  ALSO READ IN THE NUMBER (NDIS) OF DISTINCT
C     EIGENVALUES OF T(1,MEV) THAT WERE COMPUTED BUT THIS VALUE IS
C     NOT USED IN THE EIGENVECTOR COMPUTATIONS.
C
      READ(3,80) NGOOD,NDIS,MEV,NOLD,SVSEED,MATOLD
   80 FORMAT(4I6,I12,I8)
C
C     READ IN THE TOLERANCES USED IN THE T-MULTIPLICITY AND SPURIOUS
C     TESTS DURING THE EIGENVALUE COMPUTATIONS.
C     ALSO READ IN THE FLAG IB.  IF IB < 0, THEN SOME BETA(I) IN THE
C     T-MATRIX FILE PROVIDED ON FILE 2 FAILED THE ORTHOGONALITY
C     TEST IN THE TNORM SUBROUTINE.  USER SHOULD NOTE THAT THIS
C     PROGRAM PROCEEDS INDEPENDENTLY OF THE SIZES OF THE BETA USED.
C
      READ(3,90) MULTOL,SPUTOL,IB,BTOL
   90 FORMAT(2E15.5,I6,E13.4)
C
      WRITE(6,100) MULTOL,SPUTOL
  100 FORMAT(/' MULTIPLICITY TOLERANCE USED IN THE T-EIGENVALUE COMPUTAT
     1IONS WAS',E13.4/' TOLERANCE USED IN SPURIOUS CHECK',E13.4)
C
C     CONTINUE WRITE TO FILE 6 OF THE PARAMETERS FOR THIS RUN
C
      WRITE(6,110)NGOOD,NDIS,MEV,NOLD,MATOLD,SVSEED,MULTOL,SPUTOL,IB,
     1BTOL
  110 FORMAT(/' EIGENVALUES SUPPLIED ARE READ IN FROM FILE 3'/' FILE 3
     1HEADER IS'/4X,'NG',2X,'NDIS',3X,'MEV',2X,'NOLD',2X,'MATOLD',4X,
     1'SVSEED'/4I6,I8,I10/7X,'MULTOL',7X,'SPUTOL',6X,'IB',9X,'BTOL'/
     12E13.4,I8,E13.4)
C
C     IS THE ARRAY RITVEC LONG ENOUGH TO HOLD ALL OF THE DESIRED
C     RITZ VECTORS (APPROXIMATE EIGENVECTORS)?
```

```
      NMAX = NGOOD*N
      IF(MBOUND.EQ.1) GO TO 120
      IF(TVSTOP.NE.1.AND.NMAX.GT.MDIMRV) GO TO 1310
C
C     CHECK THAT THE ORDER N AND THE MATRIX IDENTIFICATION NUMBER
C     MATNO SPECIFIED BY THE USER AGREE WITH THOSE READ IN FROM
C     FILE 3.
  120 ITEMP = (NOLD-N)**2+(MATOLD-MATNO)**2
      IF (ITEMP.NE.0) GO TO 1330
C
C     READ IN FROM FILE 3, THE T(1,MEV)-MULTIPLICITIES OF THE
C     EIGENVALUES WHOSE EIGENVECTORS ARE TO BE COMPUTED, THE VALUES
C     OF THESE EIGENVALUES AND THEIR MINIMAL GAPS AS EIGENVALUES
C     OF THE USER-SPECIFIED MATRIX AND AS EIGENVALUES OF THE T-MATRIX.
C
      READ(3,20) EXPLAN
      READ(3,130) (MP(J),GOODEV(J),TMINGP(J),AMINGP(J), J=1,NGOOD)
  130 FORMAT(5X,I5,2E22.14,2E10.3)
C
      WRITE(6,140) (J,GOODEV(J),MP(J),TMINGP(J),AMINGP(J), J=1,NGOOD)
  140 FORMAT(/' EIGENVALUES READ IN, T-MULTIPLICITIES, T-GAPS AND A-GAPS
     1 '/4X,' J ',15X,' EIGENVALUE',14X,'TMULT',4X,'  TMINGAP ',4X,
     1'  AMINGAP '/(I6,2E20.12,I4,2E15.4))
C
C     READ IN ERROR ESTIMATES
      WRITE(6,170) MEV,SVSEED
C     CHECK WHETHER OR NOT THERE ARE ANY T-ISOLATED EIGENVALUES IN
C     THE EIGENVALUES PROVIDED
      DO 150 J=1,NGOOD
      IF(MP(J).EQ.1) GO TO 160
  150 CONTINUE
      GO TO 190
  160 READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
  170 FORMAT(/' THESE EIGENVALUES WERE COMPUTED USING A T-MATRIX OF
     1ORDER ',I5/' AND SEED FOR RANDOM NUMBER GENERATOR =',I12)
      READ(4,180) NISO
  180 FORMAT(18X,I6)
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
      READ(4,20) EXPLAN
  190 DO 220 J=1,NGOOD
      ERR(J) = 0.D0
      IF(MP(J).NE.1) GO TO 220
      READ(4,200) EVAL, ERR(J)
  200 FORMAT(10X,2E20.12,E14.3)
      IF(CDABS(EVAL - GOODEV(J)).LT.1.D-10) GO TO 220
      WRITE(6,210) EVAL,GOODEV(J)
  210 FORMAT(' PROBLEM WITH READ IN OF ERROR ESTIMATES'/' EIGENVALUE REA
     1D IN',2E20.12,' DOES NOT MATCH GOODEV(J) ='/2E20.12)
      GO TO 1550
C
  220 CONTINUE
C
```

```
      WRITE(6,230) (J,GOODEV(J),ERR(J), J=1,NGOOD)
  230 FORMAT(' ERROR ESTIMATES ='/4X,' J',15X,'EIGENVALUE',20X,'ESTIMATE
     1'/(I6,2E20.12,E14.3))
C
C     READ IN THE SIZE OF THE T-MATRIX PROVIDED ON FILE 2.  READ IN
C     THE ORDER OF THE USER-SPECIFIED MATRIX , THE SEED FOR THE
C     RANDOM NUMBER GENERATOR, AND THE MATRIX/TEST IDENTIFICATION
C     NUMBER THAT WERE USED IN THE LANCZOS EIGENVALUE COMPUTATIONS.
C     IF FLAG IREAD = 0, REGENERATE HISTORY FROM SCRATCH
C     HISTORY MUST BE STORED IN MACHINE FORMAT, ((4Z20) FOR
C     IBM/3081)
C
      IF(IREAD.EQ.0)  GO TO 330
C
      READ(2,240) KMAX,NOLD,SVSOLD,MATOLD
  240 FORMAT(2I6,I12,I8)
C
      WRITE(6,250) KMAX,NOLD,SVSOLD,MATOLD
  250 FORMAT(/' READ IN HEADER FOR T-MATRICES'/' FILE 2 HEADER IS'/
     1 2X,'KMAX',2X,'NOLD',6X,'SVSOLD',2X,'MATOLD'/2I6,I12,I8/)
C
C     CHECK THAT THE ORDER, THE MATRIX/TEST IDENTIFICATION NUMBER
C     AND THE SEED FOR THE RANDOM NUMBER GENERATOR USED IN THE
C     LANCZOS COMPUTATIONS THAT GENERATED THE HISTORY FILE
C     BEING USED AGREE WITH WHAT THE USER HAS SPECIFIED.
      IF (NOLD.NE.N.OR.MATOLD.NE.MATNO.OR.SVSOLD.NE.SVSEED) GO TO 1350
C
      KMAX1 = KMAX + 1
C
C     READ IN THE T-MATRICES FROM FILE 2.  THESE ARE USED TO GENERATE
C     THE T-EIGENVECTORS THAT WILL BE USED IN THE RITZ VECTOR
C     COMPUTATIONS.  HISTORY MUST BE IN MACHINE FORMAT.
C
      READ(2,260) (ALPHA(J), J=1,KMAX)
      READ(2,260) (BETA(J), J=1,KMAX1)
  260 FORMAT(4Z20)
C
      READ(2,260) (V1(J), J=1,N)
      READ(2,260) (V2(J), J=1,N)
C
C     KMAX MAY BE ENLARGED IF THE SIZE AT WHICH THE EIGENVALUE
C     COMPUTATIONS WERE PERFORMED IS ESSENTIALLY KMAX AND
C     THERE IS AT LEAST ONE EIGENVALUE THAT IS T-SIMPLE AND
C     T-ISOLATED, IN THE SENSE THAT IF ITS CLOSEST NEIGHBOR IS TOO
C     CLOSE THAT NEIGHBOR IS A 'GOOD' T-EIGENVALUE.
      DO 270 J = 1,NGOOD
      IF(MP(J).EQ.1) GO TO 290
  270 CONTINUE
      WRITE(6,280)
  280 FORMAT(/' ALL EIGENVALUES USED ARE T-MULTIPLE OR CLOSE TO SPURIOUS
     1 T-EIGENVALUES'/' SO DO NOT CHANGE KMAX')
      IF(KMAX.LT.MEV) GO TO 1370
      GO TO 310
C
  290 KMAXN= 11*MEV/8 + 12
```

```
      IF(MBETA.LE.KMAXN) GO TO 1530                                     CSL03750
      IF(KMAX.GE.KMAXN )  GO TO 310                                     CSL03760
      WRITE(6,300) KMAX, KMAXN                                          CSL03770
  300 FORMAT(' ENLARGE KMAX FROM ',I6,' TO ',I6)                        CSL03780
      MOLD1 = KMAX + 1                                                  CSL03790
      KMAX = KMAXN                                                      CSL03800
      GO TO 380                                                         CSL03810
C                                                                       CSL03820
  310 WRITE(6,320) KMAX                                                 CSL03830
  320 FORMAT(/' T-MATRICES HAVE BEEN READ IN FROM FILE 2'/' THE LARGEST CSL03840
     1SIZE T-MATRIX ALLOWED IS',I6/)                                    CSL03850
C                                                                       CSL03860
      IF(IREAD.EQ.1) GO TO 400                                          CSL03870
C                                                                       CSL03880
C     REGENERATE THE ALPHA AND BETA                                     CSL03890
C                                                                       CSL03900
  330 MOLD1 = 1                                                         CSL03910
C                                                                       CSL03920
C     SET KMAX                                                          CSL03930
      DO 340 J = 1,NGOOD                                                CSL03940
      IF(MP(J).EQ.1) GO TO 360                                          CSL03950
  340 CONTINUE                                                          CSL03960
      KMAX = MEV + 12                                                   CSL03970
      WRITE(6,350) KMAX                                                 CSL03980
  350 FORMAT(/' ALL EIGENVALUES FOR WHICH EIGENVECTORS ARE TO BE COMPUTECSL03990
     1D ARE EITHER T-MULTIPLE OR CLOSE TO'/' A SPURIOUS EIGENVALUE. THERCSL04000
     1EFORE SET KMAX = MEV + 12 = ',I7)                                 CSL04010
      GO TO 380                                                         CSL04020
C                                                                       CSL04030
  360 KMAXN = 11*MEV/8 + 12                                             CSL04040
      IF(MBETA.LE.KMAXN) GO TO 1530                                     CSL04050
      WRITE(6,370) KMAXN                                                CSL04060
  370 FORMAT(' SET KMAX EQUAL TO ',I6)                                  CSL04070
      KMAX = KMAXN                                                      CSL04080
C                                                                       CSL04090
  380 WRITE(6,390) MOLD1,KMAX                                           CSL04100
  390 FORMAT(/' LANCZS SUBROUTINE GENRATES ALPHA(J), BETA(J+1), J =',   CSL04110
     1 I6,' TO ', I6/)                                                  CSL04120
C                                                                       CSL04130
C-----------------------------------------------------------------------CSL04140
C                                                                       CSL04150
      CALL LANCZS(CMATV,V1,V2,ALPHA,BETA,GR,GC,G,KMAX,MOLD1,N,SVSEED)   CSL04160
C                                                                       CSL04170
C-----------------------------------------------------------------------CSL04180
C                                                                       CSL04190
  400 CONTINUE                                                          CSL04200
C                                                                       CSL04210
C     SIMPLE STURM SEQUENCING IS NOT VALID FOR COMPLEX SYMMETRIC        CSL04220
C     MATRICES.  THUS, THE STRATEGY USED HERE FOR SELECTING             CSL04230
C     APPROPRIATE SIZE T-MATRICES FOR THE EIGENVECTOR COMPUTATIONS      CSL04240
C     MUST BE DIFFERENT FROM THAT USED IN THE REAL SYMMETRIC,           CSL04250
C     HERMITIAN, AND SINGULAR VALUE CASES.  AS IN THOSE CASES,          CSL04260
C     FOR EACH EIGENVALUE, A FIRST GUESS IS SELECTED AND THEN           CSL04270
C     LOOPING ON THE SIZE OF THE T-EIGENVECTOR COMPUTATIONS             CSL04280
C     DETERMINES APPROPRIATE SIZES FOR THE EIGENVECTOR COMPUTATIONS.    CSL04290
```

```
C     FIRST GUESSES AT APPROPRIATE SIZES ARE SPECIFIED BELOW.
C
      DO 430 J = 1,NGOOD
      EVAL = GOODEV(J)
C     COMPUTE A FIRST GUESS ON AN APPROPRIATE SIZE T-MATRIX EACH
C     EIGENVALUE.
      IF(MP(J).GT.1) GO TO 410
C     EIGENVALUE IS T-SIMPLE
      IF(MP(J).EQ.MONE) GO TO 420
C     EIGENVALUE IS T-SIMPLE AND T-ISOLATED
      MA(J) = (8*MEV)/9  +  1
      ML(J) = ((11*MEV)/8 + 12)
      GO TO 430
C     EIGENVALUE IS T-MULTIPLE
  410 MA(J) = (5*MEV)/(4*MP(J))  +  1
      ML(J) = (7*MEV)/(4*MP(J)) + 1
      GO TO 430
C     EIGENVALUE IS T-SIMPLE AND NOT T-ISOLATED
  420 MA(J) = (5*MEV)/8 + 1
      ML(J) = MEV
  430 CONTINUE
C
      IF (IWRITE.EQ.1) WRITE(6,440) (MA(JJ), JJ=1,NGOOD)
  440 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZES FOR T-MATRICES '/
     1' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/(13I6))
C
      WRITE(10,450) N,KMAX
  450 FORMAT(2I8,' = ORDER OF USER MATRIX AND MAX ORDER OF T(1,MEV)')
C
      WRITE(10,460)
  460 FORMAT(/' 1ST GUESS AT APPROPRIATE SIZES FOR T-MATRICES '/
     1' ACTUAL VALUES WILL PROBABLY BE 1/4 AGAIN AS MUCH'/)
      WRITE(10,470)
  470 FORMAT(5X,'J',8X,'REAL(GOODEV)',8X,'IMAG(GOODEV)',7X,'MA(J)',
     17X,'MP(J)')
C
      WRITE(10,480) (J,GOODEV(J), MA(J), MP(J), J=1,NGOOD)
  480 FORMAT(I6,2E20.12,I12,I12)
C
      IF(MBOUND.EQ.1) WRITE(10,490)
  490 FORMAT(/' GOODEV(J) IS A GOOD EIGENVALUE OF T(1,MEV)'/
     1 ' IABS(MA(J)) = APPROPRIATE SIZE T-MATRIX FOR GOODEV(J)'/
     1 ' INITIAL VALUE OF MA(J) IS CHOSEN HEURISTICALLY'/
     1 ' PROGRAM LOOPS ON SIZE OF T-MATRIX TO GET BETTER SIZE'/
     1 ' END OF SIZES OF T-MATRICES FILE 10'///)
C
C
C     TERMINATE AFTER COMPUTING 1ST GUESSES ON SIZES OF T-MATRICES
C     REQUIRED FOR THE GIVEN EIGENVALUES?
      IF(MBOUND.EQ.1) GO TO 1390
C
C
C     IS THERE ROOM FOR ALL OF THE REQUESTED T-EIGENVECTORS?
      MTOL = 0
      DO 500 J = 1,NGOOD
```

```
      MTOL = MTOL + IABS(MA(J))
  500 CONTINUE
      MTOL = (5*MTOL)/4
      IF(MTOL.GT.MDIMTV.AND.NTVCON.EQ.0) GO TO 1410
C
C-----------------------------------------------------------------------
C     GENERATE A RANDOM VECTOR TO BE USED REPEATEDLY BY
C     SUBROUTINE INVERM
C
      ILL = RHSEED
      CALL GENRAN(ILL,G,KMAX)
C
C-----------------------------------------------------------------------
C
      DO 510 I = 1,KMAX
  510 GR(I) = G(I)
C
C-----------------------------------------------------------------------
C
      CALL GENRAN(ILL,G,KMAX)
C
C-----------------------------------------------------------------------
C
      DO 520 I = 1,KMAX
  520 GC(I) = G(I)
C
C     FOR EACH EIGENVALUE LOOP ON T-EIGENVECTOR COMPUTATIONS TO
C     COMPUTE AN APPROPRIATE T-EIGENVECTOR TO USE IN THE RITZ
C     VECTOR COMPUTATIONS.
C
      MTOL = 0
      NTVEC = 0
      DO 690 J = 1,NGOOD
      ICOUNT = 0
      TFLAG = 0
      ERRMIN = 10.D0
      MABEST = MPMIN
      IF(MP(J).EQ.MPMIN) GO TO 690
      EVAL = GOODEV(J)
  530 KMAXU = IABS(MA(J))
C     SELECT A SUITABLE INCREMENT FOR THE ORDERS OF T-MATRICES
C     TO BE CONSIDERED IN DETERMINING APPROPRIATE SIZES FOR THE RITZ
C     VECTOR COMPUTATIONS
      IF(ICOUNT.GT.0) GO TO 560
C     SELECT IDELTA(J) BASED UPON THE MULTIPLICITY IN T(1,MEV)
      IF(MP(J).GT.1) GO TO 540
      IF(MP(J).LT.0) GO TO 550
C     MP(J) = 1, INITIAL MA(J) = 8*MEV/9 + 1
      IDELTA(J) = (ML(J) - IABS(MA(J)))/10  + 1
      GO TO 560
C     MULTIPLE T-EIGENVALUE: INITIAL MA(J) = 5*MEV/4*MP + 1
  540 IDELTA(J) = (ML(J) - IABS(MA(J)))/10  +  1
      GO TO 560
C     T-SIMPLE EVALUE, NEAR SPURIOUS ONE, INITIAL MA(J) = 5*MEV/8 + 1
  550 IDELTA(J) = (ML(J) - IABS(MA(J)))/10  +  1
```

```
  560 ICOUNT = ICOUNT + 1                                               CSL05400
      MTOL = MTOL+KMAXU                                                 CSL05410
C                                                                       CSL05420
C     IS THERE ROOM IN TVEC ARRAY FOR THE NEXT T-EIGENVECTOR?           CSL05430
C     IF NOT, SKIP TO RITZ VECTOR COMPUTATIONS.                         CSL05440
      IF (MTOL.GT.MDIMTV) GO TO 700                                     CSL05450
C                                                                       CSL05460
      IT = 3                                                            CSL05470
      KINT = MTOL - KMAXU +1                                            CSL05480
C                                                                       CSL05490
C     RECORD THE BEGINNING AND END OF THE T-EIGENVECTOR BEING COMPUTED  CSL05500
      MINT(J) = KINT                                                    CSL05510
      MFIN(J) = MTOL                                                    CSL05520
C                                                                       CSL05530
C-----------------------------------------------------------------------CSL05540
C     SUBROUTINE INVERM DOES INVERSE ITERATION, I.E. SOLVES             CSL05550
C     (T(1,KMAXU) - EVAL)*U = RHS  FOR EACH EIGENVALUE TO OBTAIN THE    CSL05560
C     DESIRED T-EIGENVECTOR.                                            CSL05570
C                                                                       CSL05580
      IF(IWRITE.EQ.1)  WRITE(6,570) J                                   CSL05590
  570 FORMAT(/I6,'TH EIGENVALUE')                                       CSL05600
C                                                                       CSL05610
      CALL INVERM(ALPHA,BETA,V1,TVEC(KINT),EVAL,ERROR,TERROR,EPSM,      CSL05620
     1    GR,GC,INTERC,KMAXU,IT,IWRITE)                                 CSL05630
C                                                                       CSL05640
C-----------------------------------------------------------------------CSL05650
C                                                                       CSL05660
      TERR(J) = TERROR                                                  CSL05670
      TLAST(J) = ERROR                                                  CSL05680
      KMAXU1 = KMAXU + 1                                                CSL05690
      TBETA(J) = CDABS(BETA(KMAXU1))*ERROR                              CSL05700
C                                                                       CSL05710
C     AFTER COMPUTING EACH OF THE T-EIGENVECTORS,                       CSL05720
C     CHECK THE SIZE OF THE ERROR ESTIMATE, ERROR.                      CSL05730
C     IF THIS ESTIMATE IS NOT AS SMALL AS DESIRED AND                   CSL05740
C     |MA(J)| < ML(J), ATTEMPT TO INCREASE THE SIZE OF |MA(J)|          CSL05750
C     AND REPEAT THE T-EIGENVECTOR COMPUTATIONS.                        CSL05760
C                                                                       CSL05770
      IF(ERROR.LT.ERTOL.OR.TFLAG.EQ.1)  GO TO 680                       CSL05780
C                                                                       CSL05790
      IF(ERROR.GE.ERRMIN) GO TO 580                                     CSL05800
C     LAST COMPONENT IS LESS THAN MINIMAL TO DATE                       CSL05810
      ERRMIN = ERROR                                                    CSL05820
      MABEST = MA(J)                                                    CSL05830
  580 CONTINUE                                                          CSL05840
C                                                                       CSL05850
      IF(MA(J).GT.0)  ITEST = MA(J) + IDELTA(J)                         CSL05860
      IF(MA(J).LT.0)  ITEST = -(IABS(MA(J)) + IDELTA(J))                CSL05870
      IF(IABS(ITEST).LE.ML(J).AND.ICOUNT.LE.10) GO TO 600               CSL05880
C     NEW MA(J) IS GREATER THAN MAXIMUM ALLOWED.                        CSL05890
      IF(ERCONT.EQ.0.OR.MABEST.EQ.MPMIN) GO TO 620                      CSL05900
      TFLAG = 1                                                         CSL05910
      MA(J) = MABEST                                                    CSL05920
      MTOL = MTOL - KMAXU                                               CSL05930
      WRITE(6,590) MA(J)                                                CSL05940
```

```
  590 FORMAT(' 10 ORDERS WERE CONSIDERED.  NONE SATISFIED THE ERROR TEST
     1'/' THEREFORE USE THE BEST ORDER OBTAINED FOR THE EIGENVECTORS'
     1,I6)
      GO TO 530
C
  600 MA(J) = ITEST
C
      MT = IABS(MA(J))
      IF(IWRITE.EQ.1) WRITE(6,610) MT
  610 FORMAT(/' CHANGE SIZE OF T-MATRIX TO',I6,' RECOMPUTE T-EIGENVECTOR
     1')
C
      MTOL = MTOL - KMAXU
C
      GO TO 530
C
C        APPROPRIATE SIZE T-MATRIX WAS NOT OBTAINED
  620 CONTINUE
      WRITE(10,630) J,EVAL,MP(J)
  630 FORMAT(/' ON 10 INCREMENTS NOT ABLE TO IDENTIFY APPROPRIATE SIZE
     1T-MATRIX FOR'/
     1' EIGENVALUE(',I4,') =',2E20.12,' T-MULTIPLICITY =',I4/)
      IF(MP(J).GT.1) WRITE(10,640)
      IF(MP(J).LT.0)  WRITE(10,650)
      IF(MP(J).EQ.1) WRITE(10,660)
  640 FORMAT(/' ORDERS TESTED RANGED FROM (5*MEV/4*MP(J)) TO APPROXIMATE
     1LY'/'  (7*MEV)/(4*MP(J)'/)
  650 FORMAT(/' ORDERS TESTED RANGED FROM (5*MEV/8)  TO MEV'/)
  660 FORMAT(/' ORDERS TESTED RANGED FROM 8*MEV/9 TO APPROXIMATELY 11*ME
     1V/8'/)
      WRITE(10,670)
  670 FORMAT(' ALLOWING LARGER ORDERS FOR THE T-MATRICES MAY RESULT IN
     1 SUCCESS'/' BUT PROBABLY WILL NOT.  PROBLEM IS PROBABLY DUE TO'
     1 /' LACK OF CONVERGENCE OF GIVEN EIGENVALUE, CHECK THE ERROR ESTIM
     1ATE')
      MP(J) = MPMIN
      MTOL = MTOL - KMAXU
      GO TO 690
  680 NTVEC = NTVEC + 1
C
  690 CONTINUE
      NGOODC = NGOOD
      GO TO 720
C
C     COME HERE IF THERE IS NOT ENOUGH ROOM FOR ALL OF T-EIGENVECTORS
  700 NGOODC = J-1
      WRITE(6,710) J,MTOL,MDIMTV
  710 FORMAT(/' NOT ENOUGH ROOM IN TVEC ARRAY FOR ',I4,'TH T-EIGENVECTOR
     1'/' TVEC-DIMENSION REQUESTED = ',I6,' BUT TVEC HAS DIMENSION ',I6/
     1)
      IF(NGOODC.EQ.0) GO TO 1430
      MTOL = MTOL-KMAXU
C
  720 CONTINUE
C
```

```
C     THE LOOP ON T-EIGENVECTOR COMPUTATIONS IS COMPLETE.               CSL06500
C     WRITE OUT THE SIZE T-MATRICES THAT WILL BE USED FOR               CSL06510
C     THE RITZ VECTOR COMPUTATIONS.                                     CSL06520
C                                                                       CSL06530
      WRITE(10,730)                                                     CSL06540
  730 FORMAT(/' SIZES OF T-MATRICES THAT WILL BE USED IN THE RITZ COMPUTCSL06550
     1ATIONS'/5X,'J',13X,'REAL(GOODEV)',13X,'IMAG(GOODEV)',1X,'MA(J)')  CSL06560
C                                                                       CSL06570
      WRITE(10,740)   (J,GOODEV(J),MA(J), J=1,NGOOD)                    CSL06580
  740 FORMAT(I6,2E25.14,I6)                                             CSL06590
      WRITE(10,490)                                                     CSL06600
C                                                                       CSL06610
      WRITE(6,750) MTOL                                                 CSL06620
  750 FORMAT(/' THE CUMULATIVE LENGTH OF THE T-EIGENVECTORS IS',I18)    CSL06630
C                                                                       CSL06640
      WRITE(6,760) NTVEC,NGOOD                                          CSL06650
  760 FORMAT(/I6,' T-EIGENVECTORS OUT OF',I6,' REQUESTED WERE COMPUTED')CSL06660
C                                                                       CSL06670
C     SAVE THE T-EIGENVECTORS ON FILE 11?                               CSL06680
      IF(TVSTOP.NE.1.AND.SVTVEC.EQ.0) GO TO 820                         CSL06690
C                                                                       CSL06700
      WRITE(11,770) NTVEC,MTOL,MATNO,SVSEED                             CSL06710
  770 FORMAT(I6,3I12,' = NTVEC,MTOL,MATNO,SVSEED')                      CSL06720
C                                                                       CSL06730
      DO 800 J=1,NGOODC                                                 CSL06740
C     IF MP(J) = MPMIN THEN NO SUITABLE T-EIGENVECTOR IS AVAILABLE      CSL06750
C     FOR THAT EIGENVALUE.                                              CSL06760
      IF(MP(J).EQ.MPMIN) WRITE(11,780) J,MA(J),GOODEV(J),MP(J)          CSL06770
  780 FORMAT(2I6,2E20.12,I6/' TH EIGVAL,T-SIZE,EVALUE,FLAG,NO EIGVEC')  CSL06780
      IF(MP(J).NE.MPMIN) WRITE(11,790) J,MA(J),GOODEV(J),MP(J)          CSL06790
  790 FORMAT(I6,I6,2E20.12,I6/' T-EIGVEC,SIZE T,EVALUE OF A,MP(J)')     CSL06800
      IF(MP(J).EQ.MPMIN) GO TO 800                                      CSL06810
      KI = MINT(J)                                                      CSL06820
      KF = MFIN(J)                                                      CSL06830
C                                                                       CSL06840
      WRITE(11,260) (TVEC(K), K=KI,KF)                                  CSL06850
C                                                                       CSL06860
  800 CONTINUE                                                          CSL06870
C                                                                       CSL06880
      IF(TVSTOP.NE.1) GO TO 820                                         CSL06890
C                                                                       CSL06900
      WRITE(6,810) TVSTOP, NTVEC,NGOOD                                  CSL06910
  810 FORMAT(/' USER SET TVSTOP = ',I1/                                 CSL06920
     1' THEREFORE PROGRAM TERMINATES AFTER T-EIGENVECTOR COMPUTATIONS'/ CSL06930
     1' T-EIGENVECTORS THAT WERE COMPUTED ARE SAVED ON FILE 11'/        CSL06940
     1I8,' T-EIGENVECTORS WERE COMPUTED OUT OF',I7,' REQUESTED'/)       CSL06950
C                                                                       CSL06960
      GO TO 1550                                                        CSL06970
C                                                                       CSL06980
  820 CONTINUE                                                          CSL06990
C     IF NOT ABLE TO COMPUTE ALL THE REQUESTED T-EIGENVECTORS           CSL07000
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS ANYWAY?             CSL07010
C                                                                       CSL07020
      IF(NTVEC.NE.NGOOD.AND.LVCONT.EQ.0) GO TO 1450                     CSL07030
C                                                                       CSL07040
```

```
C     COMPUTE THE MAXIMUM SIZE OF THE T-MATRIX USED FOR THOSE
C     EIGENVALUES WITH GOOD ERROR ESTIMATES.
C
      KMAXU = 0
      DO 830 J = 1,NGOODC
      MT = IABS(MA(J))
      IF(MT.LT.KMAXU.OR.MP(J).EQ.MPMIN) GO TO 830
      KMAXU = MT
  830 CONTINUE
C
      IF(KMAXU.EQ.0) GO TO 1490
C
      WRITE(6,840) KMAXU
  840 FORMAT(/I6,' = LARGEST SIZE T-MATRIX TO BE USED IN THE RITZ VECTOR
     1 COMPUTATIONS')
C
C     COUNT THE NUMBER OF RITZ VECTORS NOT BEING COMPUTED
      MREJEC = 0
      DO 850 J=1,NGOODC
  850 IF(MP(J).EQ.MPMIN)  MREJEC = MREJEC + 1
      MREJET = MREJEC + (NGOOD-NGOODC)
      IF(MREJET.NE.0) WRITE(6,860) MREJET
  860 FORMAT(/' RITZ VECTORS ARE NOT COMPUTED FOR',I6,' OF THE EIGENVALU
     1ES'/)
      NACT = NGOODC - MREJEC
      WRITE(6,870) NGOOD,NTVEC,NACT
  870 FORMAT(/I6,' RITZ VECTORS WERE REQUESTED'/I6,' T-EIGENVECTORS WERE
     1COMPUTED'/I6,' RITZ VECTORS WILL BE COMPUTED'/)
C     CHECK IF THERE ARE ANY RITZ VECTORS TO COMPUTE
      IF(MREJEC.EQ.NGOODC) GO TO 1470
C
C     CONTINUE WITH THE LANCZOS VECTOR COMPUTATIONS?
      IF(LVCONT.EQ.0.AND.MREJEC.NE.0) GO TO 1450
C
C     NOW COMPUTE THE RITZ VECTORS.  REGENERATE THE
C     LANCZOS VECTORS.
C
      DO 880 I = 1,NMAX
  880 RITVEC(I) = ZEROC
C
C-----------------------------------------------------------------------
C     REGENERATE THE STARTING VECTOR. THIS MUST BE GENERATED AND
C     NORMALIZED PRECISELY THE WAY IT WAS DONE IN THE EIGENVALUE
C     COMPUTATIONS, OTHERWISE THERE WILL BE A MISMATCH BETWEEN
C     THE T-EIGENVECTORS THAT HAVE BEEN COMPUTED FROM THE T-MATRICES
C     READ IN FROM FILE 2 AND THE LANCZOS VECTORS THAT ARE
C     BEING REGENERATED.
C
      IIL = SVSEED
      CALL GENRAN(IIL,G,N)
C
C-----------------------------------------------------------------------
C
      DO 890 I = 1,N
  890 GR(I) = G(I)
```

```
C
C-----------------------------------------------------------------------
C
      CALL GENRAN(IIL,G,N)
C
C-----------------------------------------------------------------------
C
      DO 900 I = 1,N
  900 GC(I) = G(I)
C
      DO 910 I = 1,N
  910 V2(I) = DCMPLX(GR(I),GC(I))
C
C-----------------------------------------------------------------------
      CALL INPRDC(V2,V2,SUMC,N)
C-----------------------------------------------------------------------
C
      SUMC = ONE/CDSQRT(SUMC)
      DO 920 I = 1,N
      V1(I) = ZEROC
  920 V2(I) = V2(I)*SUMC
C
C     LOOP FOR GENERATING REQUIRED RITZ VECTORS (IVEC = 1,KMAXU)
C
      IVEC = 1
      BATA = ZEROC
C
      GO TO 980
C
  930 CONTINUE
C
C-----------------------------------------------------------------------
C
C    CMATV(V2,V1,BATA) CALCULATES  V1 = A*V2 - BATA*V1
      CALL CMATV(V2,V1,BATA)
      CALL INPRDC(V2,V1,ALFA,N)
C
C-----------------------------------------------------------------------
C
      DO 940 J=1,N
  940 V1(J) = V1(J)-ALFA*V2(J)
C
C-----------------------------------------------------------------------
      CALL INPRDC(V1,V1,BATA,N)
C-----------------------------------------------------------------------
C
      BATA = CDSQRT(BATA)
      SUMC = ONE/BATA
C
      TEMPC = BETA(IVEC)
      TEMP = CDABS(BATA - TEMPC)/CDABS(TEMPC)
      IF (TEMP.LT.1.0D-10)GO TO 960
C
C     IF THE BETA BEING REGENERATED DO NOT MATCH THE HISTORY FILE
C     THEN SOMETHING IS WRONG IN THE LANCZOS VECTOR GENERATION
```

```
C     AND PROGRAM TERMINATES FOR USER TO CORRECT THE PROBLEM            CSL08150
C     WHICH MUST BE IN THE STARTING VECTOR GENERATION OR IN             CSL08160
C     THE MATRIX-VECTOR MULTIPLY SUBROUTINE CMATV SUPPLIED.             CSL08170
C     THIS PART OF THE COMPUTATIONS MUST BE IDENTICAL TO THE            CSL08180
C     CORRESPONDING PART IN THE EIGENVALUE COMPUTATIONS.                CSL08190
C                                                                       CSL08200
      WRITE(6,950) IVEC,BATA,BETA(IVEC),TEMP                            CSL08210
  950 FORMAT(/2X,'IVEC',16X,'BATA',10X,'BETA(IVEC)',14X,'RELDIF'/I6,    CSL08220
     13E20.12/' IN LANCZOS VECTOR REGENERATION THE ENTRIES OF THE TRIDIACSL08230
     1GONAL MATRICES BEING'/' GENERATED ARE NOT THE SAME AS THOSE IN THECSL08240
     1 MATRIX SUPPLIED ON FILE 2.'/' THEREFORE SOMETHING IS BEING INITIACSL08250
     1LIZED OR COMPUTED DIFFERENTLY FROM THE WAY'/' IT WAS COMPUTED IN TCSL08260
     1HE EIGENVALUE COMPUTATIONS'/' THE PROGRAM TERMINATES FOR THE USER CSL08270
     1TO DETERMINE WHAT THE PROBLEM IS'/)                               CSL08280
      GO TO 1550                                                        CSL08290
C                                                                       CSL08300
C                                                                       CSL08310
  960 CONTINUE                                                          CSL08320
      DO 970 J = 1,N                                                    CSL08330
      TEMPC = SUMC*V1(J)                                                CSL08340
      V1(J) = V2(J)                                                     CSL08350
  970 V2(J) = TEMPC                                                     CSL08360
C                                                                       CSL08370
  980 CONTINUE                                                          CSL08380
C                                                                       CSL08390
      LFIN = 0                                                          CSL08400
      DO 1000 J = 1,NGOODC                                              CSL08410
      LL = LFIN                                                         CSL08420
      LFIN = LFIN + N                                                   CSL08430
C                                                                       CSL08440
      IF(IABS(MA(J)).LT.IVEC.OR.MP(J).EQ.MPMIN) GO TO 1000              CSL08450
      II = IVEC + MINT(J) - 1                                           CSL08460
      TEMPC = TVEC(II)                                                  CSL08470
C     II IS THE (IVEC)TH COMPONENT OF THE T-EIGENVECTOR CONTAINED       CSL08480
C     IN TVEC(MINT(J)).                                                 CSL08490
C                                                                       CSL08500
      DO 990 K = 1,N                                                    CSL08510
      LL = LL + 1                                                       CSL08520
  990 RITVEC(LL) = TEMPC*V2(K) + RITVEC(LL)                             CSL08530
C                                                                       CSL08540
 1000 CONTINUE                                                          CSL08550
C                                                                       CSL08560
      IVEC = IVEC + 1                                                   CSL08570
      IF (IVEC.LE.KMAXU) GO TO 930                                      CSL08580
C                                                                       CSL08590
C                                                                       CSL08600
C     RITZVECTOR GENERATION IS COMPLETE. NORMALIZE EACH RITZVECTOR.     CSL08610
C     NOTE THAT IF CERTAIN RITZ VECTORS WERE NOT COMPUTED THEN THE      CSL08620
C     CORRESPONDING PORTION OF THE RITVEC ARRAY WAS NOT UTILIZED.       CSL08630
C                                                                       CSL08640
      LFIN = 0                                                          CSL08650
      DO 1050 J = 1,NGOODC                                              CSL08660
C                                                                       CSL08670
      KK = LFIN                                                         CSL08680
      LFIN = LFIN + N                                                   CSL08690
```

```
      IF(MP(J).EQ.MPMIN) GO TO 1050                                     CSL08700
C                                                                       CSL08710
      DO 1010 K = 1,N                                                   CSL08720
      KK = KK + 1                                                       CSL08730
 1010 V2(K) = RITVEC(KK)                                                CSL08740
C                                                                       CSL08750
C-----------------------------------------------------------------------CSL08760
      CALL INPRDC(V2,V2,SUMC,N)                                         CSL08770
C-----------------------------------------------------------------------CSL08780
C                                                                       CSL08790
      SUMC = CDSQRT(SUMC)                                               CSL08800
      RNORM(J) = CDABS(SUMC)                                            CSL08810
      TEMP = DABS(ONE-RNORM(J))                                         CSL08820
      SUMC = DCMPLX(ONE,ZERO)/SUMC                                      CSL08830
C                                                                       CSL08840
      KK = LFIN - N                                                     CSL08850
      DO 1020 K = 1,N                                                   CSL08860
      KK = KK + 1                                                       CSL08870
      V2(K) = SUMC*V2(K)                                                CSL08880
 1020 RITVEC(KK) = V2(K)                                                CSL08890
C                                                                       CSL08900
C     COMPUTE THE 'REAL' NORM                                           CSL08910
C                                                                       CSL08920
C-----------------------------------------------------------------------CSL08930
      CALL CINPRD(V2,V2,SUM,N)                                          CSL08940
C-----------------------------------------------------------------------CSL08950
C                                                                       CSL08960
      IF (IWRITE.NE.0) WRITE(6,1030) J,GOODEV(J)                        CSL08970
 1030 FORMAT(/I5,' TH EIGENVALUE CONSIDERED = ',2E20.12/)               CSL08980
C                                                                       CSL08990
      IF (IWRITE.NE.0) WRITE(6,1040) TERR(J),TBETA(J),RNORM(J),SUM      CSL09000
 1040 FORMAT(' NORM OF ERROR IN T-EIGENVECTOR = ',E14.3/                CSL09010
     1 ' CDABS(BETA(MA(J)+1)*U(MA(J))) ',E14.3/                         CSL09020
     1 ' CDABS(EUCLIDEAN-NORM(RITVEC))  = ',E14.3/                      CSL09030
     1 ' HERMITIAN-NORM(RITVEC)**2  = ',E14.3/)                         CSL09040
C                                                                       CSL09050
      LINT = LFIN - N + 1                                               CSL09060
      EVAL = GOODEV(J)                                                  CSL09070
C                                                                       CSL09080
C-----------------------------------------------------------------------CSL09090
C                                                                       CSL09100
      CALL CMATV(RITVEC(LINT),V2,EVAL)                                  CSL09110
C                                                                       CSL09120
C-----------------------------------------------------------------------CSL09130
C                                                                       CSL09140
C     COMPUTE ERROR IN RITZ VECTOR CONSIDERED AS A EIGENVECTOR OF A.    CSL09150
C     V2 = A*RITVEC - EVAL*RITVEC                                       CSL09160
C                                                                       CSL09170
C-----------------------------------------------------------------------CSL09180
      CALL CINPRD(V2,V2,SUM,N)                                          CSL09190
C-----------------------------------------------------------------------CSL09200
C                                                                       CSL09210
      SUM = DSQRT(SUM)                                                  CSL09220
      ERR(J) = SUM                                                      CSL09230
      GAP = ABS(AMINGP(J))                                              CSL09240
```

```
      ERRDGP(J) = SUM/GAP
C
 1050 CONTINUE
C
C
C     RITZVECTORS ARE NORMALIZED AND ERROR ESTIMATES ARE IN ERR ARRAY
C     AND IN ERRDGP ARRAY. STORE EVERYTHING
C
C
      WRITE(9,1060)
 1060 FORMAT(3X,'REAL(GOODEV)',3X,'IMAG(GOODEV)',1X,'MA(J)',7X,'AMINGAP'
     1 ,4X,'AERROR',2X,'AERR/GAP',4X,'TERROR')
C
      WRITE(13,1070)
 1070 FORMAT(8X,'REAL(GOODEV)',8X,'IMAG(GOODEV)',2X,'RITZNORM',3X,'AMING
     1AP',2X,'TBETA(J)',2X,'TLAST(J)')
C
      DO 1100 J=1,NGOODC
C
      IF(MP(J).EQ.MPMIN) GO TO 1100
C
      WRITE(9,1080)GOODEV(J),MA(J),AMINGP(J),ERR(J),ERRDGP(J),TERR(J)
 1080 FORMAT(2E15.8,I6,E14.6,3E10.3)
C
      WRITE(13,1090) GOODEV(J),RNORM(J),AMINGP(J),TBETA(J),TLAST(J)
 1090 FORMAT(2E20.12,4E10.3)
C
 1100 CONTINUE
C
      IF(MREJEC.EQ.0) GO TO 1180
      WRITE(9,1110)
 1110 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING EIGENVA
     1LUES'/' EITHER BECAUSE THEY HAD NOT CONVERGED OR BECAUSE THE ERROR
     1 ESTIMATE'/'  WAS NOT AS SMALL AS DESIRED'/)
C
      WRITE(13,1120)
 1120 FORMAT(/' RITZ VECTORS WERE NOT COMPUTED FOR THE FOLLOWING EIGENVA
     1LUES'/' EITHER BECAUSE THEY HAD NOT CONVERGED OR BECAUSE'/' THE ER
     1ROR ESTIMATE WAS NOT AS SMALL AS DESIRED'/)
C
      DO 1170 J = 1,NGOODC
      IF(MP(J).NE.MPMIN) GO TO 1170
C     WRITE OUT MESSAGE FOR EACH EIGENVALUE FOR WHICH NO EIGENVECTOR
C     WAS COMPUTED.
C
      WRITE(9,1130)
 1130 FORMAT(6X,'GOODEV(J)',3X,'MA(J)',5X,'AMINGP(J)',6X,'TLAST(J)',3X,
     1'MP(J)')
      WRITE(9,1140) GOODEV(J),MA(J),AMINGP(J),TBETA(J),MP(J)
 1140 FORMAT(2E15.8,I8,2E14.4,I8)
C
      WRITE(13,1150)
 1150 FORMAT(6X,'REAL(GOODEV(J))',6X,'IMAG(GOODEV(J))',4X,'MA(J)',3X,
     1'MP(J)')
      WRITE(13,1160) GOODEV(J),MA(J),MP(J)
```

```
 1160 FORMAT(2E15.8,2I8)
C
 1170 CONTINUE
 1180 CONTINUE
C
      WRITE(9,1190)
 1190 FORMAT(/' ABOVE ARE ERROR ESTIMATES FOR THE A AND T EIGENVECTORS'/
     1 ' ASSOCIATED WITH THE GOODEV LISTED IN COLUMN 1'/
     1 ' AERROR = NORM(A*X - EV*X)  TERROR = NORM(T*Y - EV*Y) '/
     1 ' WHERE T = T(1,MA(J))    X = RITZ VECTOR = V*Y  V = SUCCESSIVE'/
     1 ' LANCZOS VECTORS. A MINGAP = GAP TO NEAREST A-EIGENVALUE'//)
C
      WRITE(13,1200)
 1200 FORMAT(/' ABOVE ARE ERROR ESTIMATES FOR THE A AND T EIGENVECTORS'/
     1 ' ASSOCIATED WITH THE GOODEV LISTED IN COLUMN 1'/
     1 ' AERROR = NORM(A*X-EV*X)  TERROR = NORM(T*Y-EV*Y) WHERE'
     1 /' T = T(1,MA(J))    X = RITZ VECTOR = V*Y  V = SUCCESSIVE '/
     1 ' LANCZOS VECTORS. A MINGAP = GAP TO NEAREST A-EIGENVALUE'/
     1 ' AERROR AND TERROR ARE GIVEN IN FILE 9. RNORM = NORM(X)'/
     1 ' BETA(M+1)*ABS(Y(M)) IS AN ESTIMATOR OF NORM(A*X-EV*X)'//)
C
C     NUMBER OF RITZ VECTORS COMPUTED
      NCOMPU = NGOODC - MREJEC
      WRITE(12,1210) N,NCOMPU,NGOODC,MATNO
 1210 FORMAT(3I6,I12,' SIZE A, NO.RITZVECS, NO.EVALUES,MATNO')
C
      LFIN = 0
      DO 1270 J = 1,NGOODC
      LINT = LFIN + 1
      LFIN = LFIN + N
C
      IF(MP(J).EQ.MPMIN) GO TO 1250
C     RITZ VECTOR WAS COMPUTED
      WRITE(12,1220) J, GOODEV(J), MP(J)
 1220 FORMAT(I6,4X,2E20.12,I6,' J, EIGENVAL, MP(J)')
C
      WRITE(12,1230) ERR(J),ERRDGP(J)
 1230 FORMAT(2E15.5,' = NORM(A*Z-EVAL*Z) AND  NORM(A*Z-EVAL*Z)/MINGAP')
C
      WRITE(12,1240) (RITVEC(LL), LL=LINT,LFIN)
C1240 FORMAT(4Z20)
 1240 FORMAT(2(2E20.12))
      GO TO 1270
C     NO RITZ VECTOR WAS COMPUTED FOR THIS EIGENVALUE
 1250 WRITE(12,1260) J,GOODEV(J),MP(J)
 1260 FORMAT(I6,4X,E20.12,I6,' J,EIGVALUE,NO RITZ VECTOR COMPUTED')
C
 1270 CONTINUE
C
C     DID ANY T-MATRICES INCLUDE OFF-DIAGONAL ENTRIES SMALLER THAN
C     DESIRED, AS SPECIFIED BY BTOL?
C
      IF(IB.GT.0) GO TO 1300
      WRITE(6,1280) KMAXU
 1280 FORMAT(/' FOR LARGEST T-MATRIX CONSIDERED',I7,' CHECK THE SIZE OF
```

```
     1BETAS')                                                           CSL10350
C                                                                       CSL10360
C-----------------------------------------------------------------------CSL10370
C                                                                       CSL10380
      CALL TNORM(ALPHA,BETA,BKMIN,TEMP,KMAXU,IBMT)                      CSL10390
C                                                                       CSL10400
C-----------------------------------------------------------------------CSL10410
C                                                                       CSL10420
      IF(IBMT.LT.0) WRITE (6,1290)                                      CSL10430
 1290 FORMAT(/' WARNING THE T-MATRICES FOR ONE OR MORE OF THE EIGENVALUECSL10440
     1S CONSIDERED'/' HAD AN OFF-DIAGONAL ENTRY THAT WAS SMALLER THAN THCSL10450
     1E BETA TOLERANCE THAT WAS SPECIFIED'/)                            CSL10460
 1300 CONTINUE                                                          CSL10470
C                                                                       CSL10480
       GO TO 1550                                                       CSL10490
C                                                                       CSL10500
 1310 WRITE(6,1320) NGOOD,NMAX,MDIMRV                                   CSL10510
 1320 FORMAT(/I4,' RITZ VECTORS WERE REQUESTED BUT THE REQUIRED DIMENSIOCSL10520
     1N',I6/' IS LARGER THAN THE USER-SPECIFIED DIMENSION OF RITVEC',I6 CSL10530
     1/' THEREFORE, THE EIGENVECTOR PROCEDURE TERMINATES FOR THE USER TOCSL10540
     1 INTERVENE')                                                      CSL10550
C                                                                       CSL10560
      GO TO 1550                                                        CSL10570
C                                                                       CSL10580
 1330 WRITE(6,1340) NOLD,N,MATOLD,MATNO                                 CSL10590
 1340 FORMAT(/' PARAMETERS READ FROM FILE 3 DO NOT AGREE WITH USER-SPECICSL10600
     1FIED'/' PARAMETERS, NOLD,N,MATOLD,MATNO = '/2I6,2I12/             CSL10610
     1' THEREFORE PROGRAM TERMINATES FOR USER TO RESOLVE DIFFERENCES'/) CSL10620
C                                                                       CSL10630
      GO TO 1550                                                        CSL10640
C                                                                       CSL10650
 1350 WRITE(6,1360)                                                     CSL10660
 1360 FORMAT(/' PARAMETERS IN ALPHA,BETA FILE READ IN DO NOT AGREE WITH CSL10670
     1 THOSE'/' SPECIFIED BY THE USER.  THEREFORE, THE PROGRAM TERMINATECSL10680
     1S FOR'/' THE USER TO RESOLVE THE DIFFERENCES'/)                   CSL10690
C                                                                       CSL10700
      GO TO 1550                                                        CSL10710
C                                                                       CSL10720
 1370 WRITE(6,1380) KMAX,MEV                                            CSL10730
 1380 FORMAT(/' IN ALPHA, BETA HISTORY HEADER KMAX =',I6/               CSL10740
     1' BUT EIGENVALUES WERE COMPUTED AT MEV = ',I6,' PROGRAM STOPS'/)  CSL10750
C                                                                       CSL10760
      GO TO 1550                                                        CSL10770
C                                                                       CSL10780
 1390 WRITE(6,1400)                                                     CSL10790
 1400 FORMAT(/' PROGRAM COMPUTED 1ST GUESSES AT T-MATRIX SIZES'/' READ TCSL10800
     1HEM TO FILE 10, THEN TERMINATED AS REQUESTED.')                   CSL10810
      GO TO 1550                                                        CSL10820
C                                                                       CSL10830
 1410 WRITE(6,1420) MTOL, MDIMTV                                        CSL10840
 1420 FORMAT(/' PROGRAM TERMINATES BECAUSE THE MINIMAL TVEC DIMENSION ANCSL10850
     1TICIPATED',I7/'  IS LARGER THAN THE TVEC DIMENSION',I7,' SPECIFIEDCSL10860
     1 BY THE USER.'/'  USER MAY RESET THE TVEC DIMENSION AND RESTART THCSL10870
     1E PROGRAM')                                                       CSL10880
      GO TO 1550                                                        CSL10890
```

```
C                                                                       CSL10900
 1430 WRITE(6,1440)                                                     CSL10910
 1440 FORMAT(/' PROGRAM TERMINATES BECAUSE NO SUITABLE T-EIGENVECTORS WECSL10920
     1RE IDENTIFIED'/' FOR ANY OF THE EIGENVALUES SUPPLIED.  PROBLEM COUCSL10930
     1LD BE CAUSED BY'/'  TOO SMALL A TVEC DIMENSION OR SIMPLY BE THAT TCSL10940
     1-EIGENVECTORS COULD'/'  NOT BE IDENTIFIED.  USER SHOULD CHECK OUTPCSL10950
     1UT'/)                                                             CSL10960
      GO TO 1550                                                        CSL10970
C                                                                       CSL10980
 1450 WRITE(6,1460) LVCONT,NTVEC,NGOOD                                  CSL10990
 1460 FORMAT(/' LVCONT FLAG =',I2,' AND NUMBER ',I5,' OF T-EIGENVECTORS CSL11000
     1 COMPUTED N.E.'/' NUMBER',I5,' REQUESTED SO PROGRAM TERMINATES'/) CSL11010
      GO TO 1550                                                        CSL11020
 1470 WRITE(6,1480)                                                     CSL11030
 1480 FORMAT(/' PROGRAM TERMINATES WITHOUT COMPUTING ANY RITZ VECTORS'/ CSL11040
     1' BECAUSE ALL T-EIGENVECTORS COMPUTED WERE REJECTED AS NOT SUITABLCSL11050
     1E'/' FOR THE RITZ VECTOR COMPUTATIONS.  PROBABLE CAUSE IS LACK OF CSL11060
     1'/' CONVERGENCE OF THE EIGENVALUES'/)                             CSL11070
       GO TO 1550                                                       CSL11080
C                                                                       CSL11090
 1490 WRITE(6,1500)                                                     CSL11100
 1500 FORMAT(/' PROGRAM INDICATES THAT IT IS NOT POSSIBLE TO COMPUTE ANYCSL11110
     1 OF THE'/' REQUESTED EIGENVECTORS. THEREFORE PROGRAM TERMINATES') CSL11120
      DO 1510 J=1,NGOODC                                                CSL11130
 1510 WRITE(6,1520)  J,GOODEV(J),MP(J)                                  CSL11140
 1520 FORMAT(/4X,' J',11X,'GOODEV(J)',4X,'MP(J)'/I6,2E20.12,I9)         CSL11150
      GO TO 1550                                                        CSL11160
C                                                                       CSL11170
 1530 WRITE(6,1540) MBETA,KMAXN                                         CSL11180
 1540 FORMAT(/' PROGRAM TERMINATES BECAUSE THE STORAGE ALLOTTED FOR THE CSL11190
     1BETA ARRAY',I8/' IS NOT SUFFICIENT FOR THE ENLARGED KMAX =',I8,' TCSL11200
     1HAT THE PROGRAM WANTS'/' USER CAN ENLARGE THE ALPHA AND BETA ARRAYCSL11210
     1S AND RERUN THE PROGRAM'/)                                        CSL11220
C                                                                       CSL11230
 1550 CONTINUE                                                          CSL11240
C                                                                       CSL11250
      STOP                                                              CSL11260
C-----END OF MAIN PROGRAM FOR COMPLEX SYMMETRIC EIGENVECTORS------------CSL11270
      END                                                               CSL11280
```

7.5 CSLEMULT: LANCZS and Sample Matrix-Vector Multiply Subroutines

```
C-----CSLEMULT--(COMPLEX SYMMETRIC MATRICES)---------------------------
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)
C            Los Alamos National Laboratory
C            Los Alamos, New Mexico 87544
C
C            E-mail:  cullumj@lanl.gov
C
C  These codes are copyrighted by the authors.  These codes
C  and modifications of them or portions of them are NOT to be
C  incorporated into any commercial codes or used for any other
C  commercial purposes such as consulting for other companies,
C  without legal agreements with the authors of these Codes.
C  If these Codes or portions of them
C  are used in other scientific or engineering research works
C  the names of the authors of these codes and appropriate
C  references to their written work are to be incorporated in the
C  derivative works.
C
C  This header is not to be removed from these codes.
C
C         REFERENCE:  Cullum and Willoughby, Chapter 6,
C         Lanczos Algorithms for Large Symmetric Eigenvalue Computations
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in
C         Applied Mathematics, 2002. SIAM Publications,
C         Philadelphia, PA. USA
C
C
C
C     CONTAINS SUBROUTINE LANCZS USED IN THE COMPLEX SYMMETRIC
C     VERSION OF THE LANCZOS PROCEDURES PLUS SAMPLE USPEC AND
C     CMATV SUBROUTINES.
C
C     PORTABILITY:
C     THESE PROGRAMS ARE NOT PORTABLE DUE TO THE USE OF COMPLEX*16
C     VARIABLES AND CORRESPONDING FUNCTIONS.  MOREOVER, THE PFORT
C     VERIFIER IDENTIFIED THE FOLLOWING ADDITIONAL NONPORTABLE
C     CONSTRUCTIONS:
C
C     1.  ENTRIES USED TO PASS THE STORAGE LOCATIONS OF THE
C         ARRAYS AND PARAMETERS NEEDED TO SPECIFY THE GIVEN MATRIX
C         FROM THE USPEC SUBROUTINE TO THE MATRIX-VECTOR MULTIPLY
C         SUBROUTINE CMATV.
C     2.  IN THE SAMPLE USPEC SUBROUTINES PROVIDED: THE FREE FORMAT
C         READ(8,*) AND THE FORMAT (20A4).  IN THE SAMPLE CMATV:
C         THE COMPUTATION OF INDICES:  IN THE AUXILIARY SUBROUTINE
C         USED FOR COMPUTING THE KNOWN EIGENVALUES OF TEST CLASS 2
C         MATRICES, THE DATA/MACHEP DEFINITION.
C
C-----LANCZS-COMPUTE THE LANCZOS TRIDIAGONAL MATRICES-------------------
C
```

```
      SUBROUTINE LANCZS(MATVEC,V1,V2,ALPHA,BETA,
     1GR,GC,G,KMAX,MOLD1,N,IIX)
C
C-----------------------------------------------------------------------
      COMPLEX*16 V1(1), V2(1), BATA, ZEROC, TEMP, SUMC
      COMPLEX*16  ALPHA(1), BETA(1)
      DOUBLE PRECISION SUM, ONE, ZERO, GR(1), GC(1)
      REAL  G(1)
      EXTERNAL MATVEC
C     COMPLEX*16  CDSQRT, DCMPLX
C-----------------------------------------------------------------------
C
      ZERO = 0.D0
      ONE = 1.D0
      ZEROC = DCMPLX(ZERO,ZERO)
C
      IF(MOLD1.GT.1)GO TO 50
C
C    ALPHA/BETA GENERATION STARTS AT I = 1
C    MOLD1 = 1 SET V1 = 0. AND V2 = RANDOM UNIT VECTOR
      IIL=IIX
C
C-----------------------------------------------------------------------
      CALL GENRAN(IIL,G,N)
C-----------------------------------------------------------------------
C
      DO 10 I = 1,N
   10 GR(I) = G(I)
C
C-----------------------------------------------------------------------
      CALL GENRAN(IIL,G,N)
C-----------------------------------------------------------------------
C
      DO 20 I = 1,N
   20 GC(I) = G(I)
C
      DO 30 I = 1,N
   30 V2(I) = DCMPLX(GR(I),GC(I))
C
C-----------------------------------------------------------------------
      CALL INPRDC(V2,V2,SUMC,N)
C-----------------------------------------------------------------------
C
      SUMC = ONE/CDSQRT(SUMC)
      DO 40 I = 1,N
      V1(I) = ZEROC
   40 V2(I) = V2(I)*SUMC
      BETA(1) = ZEROC
C
C    ALPHA BETA GENERATION LOOP
   50 CONTINUE
C
      DO 80 I=MOLD1,KMAX
      SUMC = BETA(I)
C
```

```
C-----------------------------------------------------------------------
C    MATVEC(V2,V1,SUMC) CALCULATES  V1 = A*V2 - SUMC*V1
      CALL MATVEC(V2,V1,SUMC)
      CALL INPRDC(V2,V1,SUMC,N)
C-----------------------------------------------------------------------
C
      ALPHA(I) = SUMC
      DO 60 J=1,N
   60 V1(J) = V1(J)-SUMC*V2(J)
C
C-----------------------------------------------------------------------
      CALL INPRDC(V1,V1,SUMC,N)
C-----------------------------------------------------------------------
C
      IN = I+1
      BATA = CDSQRT(SUMC)
      BETA(IN) = BATA
      SUMC = ONE/BATA
      DO 70 J=1,N
      TEMP = SUMC*V1(J)
      V1(J) = V2(J)
   70 V2(J) = TEMP
   80 CONTINUE
C     END ALPHA, BETA GENERATION LOOP
C
C-----END OF LANCZS-----------------------------------------------------
C
      RETURN
      END
C
C-----USPEC, AND CMATV FOR COMPLEX SYMMETRIC TEST MATRICES 1------------
C
C-----START OF USPEC-(COMPLEX SYMMETRIC TEST MATRICES 1)----------------
C
C     SUBROUTINE CSPEC(N,MATNO)
      SUBROUTINE USPEC(N,MATNO)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  C0,C1,C2,HALF,ONE,SCR,SCI,ANGLE
      COMPLEX*16 SC,TC,CL0,CL1
      REAL EXPLAN(20)
      DOUBLE PRECISION DARCOS
C     COMPLEX*16 DCMPLX
C-----------------------------------------------------------------------
      HALF = 0.5D0
      ONE  = 1.0D0
C
C     READ IN PARAMETERS TO DEFINE MATRIX
C     MATRIX IS COMPLEX DIAGONAL SIMILITARY TRANSFORM OF THE BLOCK
C     TOEPLITZ POISSON MATRICES USED TO TEST REAL SYMMETRIC MATRICES.
C     THE REAL POISSON MATRIX HAS SYMMETRIC TOEPLITZ BLOCKS ALONG THE
C     DIAGONAL.  EACH ONE OF THESE HAS THE PARAMETER C2 ALONG THE
C     DIAGONAL AND -C0 ABOVE AND BELOW THE DIAGONAL.  THE OFF-DIAGONAL
C     BLOCKS ARE DIAGONAL WITH DIAGONAL ENTRIES -C1.  EACH BLOCK IS
C     KX*KX AND THERE ARE KY BLOCKS.  A HERMITIAN VERSION IS OBTAINED
```

```
C     BY APPLYING A DIAGONAL SIMILARITY TRANSFORM TO THE ABOVE
C     MATRIX WHERE THE DIAGONAL MATRIX IS SUCH THAT ITS
C     DIAGONAL ENTRIES ARE (SC)**(K-1), K=1,...,N-1.
C     THIS HERMITIAN VERSION IS TURNED INTO A COMPLEX SYMMETRIC ONE
C     IN THE MATRIX VECTOR MULTIPLY BY TREATING THE BELOW DIAGONAL
C     ENTRIES AS BEING EQUAL TO THE ABOVE DIAGONAL ENTRIES RATHER
C     THAN THEIR COMPLEX CONJUGATES.
C
      READ(8,10) EXPLAN
   10 FORMAT(20A4)
      READ(8,*) NOLD,MATOLD
      WRITE(6,20) NOLD,MATOLD
   20 FORMAT(' ORDER OF MATRIX READ FROM FILE =',I6/' MATRIX NUMBER =',
     1I8)
C
C     TEST OF PARAMETER CORRECTNESS
      ITEMP = (NOLD-N)**2 + (MATNO-MATOLD)**2
C
      IF(ITEMP.EQ.0) GO TO 40
C
      WRITE(6,30)
   30 FORMAT(' PROGRAM TERMINATES BECAUSE EITHER ORDERS OF OR LABELS FOR
     1 MATRIX DISAGREE')
      GO TO 100
C
   40 CONTINUE
C
      READ(8,10) EXPLAN
      READ(8,*)  C0,KX,KY
      READ(8,10) EXPLAN
      READ(8,*) SCR
      ANGLE = DARCOS(SCR)
      SCI = DSIN(ANGLE)
      SC = DCMPLX(SCR,SCI)
      WRITE(6,50) SC
      WRITE(9,50) SC
   50 FORMAT(' GENERATOR OF DIAGONAL TRANSFORMATION ='/2E20.12)
C
      TC = SC
      DO 60 J=2,KX
   60 TC = SC*TC
      WRITE(6,70) TC
   70 FORMAT(' TC = ',2E20.12)
C
      N = KX*KY
      C2 = ONE
      C1 = HALF-C0
      CL0 = -SC*C0
      CL1 = -TC*C1
C
      WRITE(6,80) N,KX,KY,C2,C0,C1
   80 FORMAT(/5X,'N',4X,'KX',4X,'KY',7X,'DIAGONAL',3X,'X-CODIAGONAL',
     1 3X,'Y-CODIAGONAL'/3I6,3E15.8/)
C
C-----------------------------------------------------------------------
```

```
       CALL HMATVE(C2,CL0,CL1,KX,KY)
C-----------------------------------------------------------------------
C
   90 CONTINUE
      RETURN
C
C-----END OF USPEC------------------------------------------------------
  100 STOP
      END
C
C-----START OF CSMATV (FOR TEST MATRICES 1)-----------------------------
C     CALCULATE U = A*W - SUMC*U FOR COMPLEX SYMMETRIC MATRICES
C     HERE WE HAVE TAKEN A HERMITIAN VERSION OF POISSON MATRICES
C     AND TURNED IT INTO A COMPLEX SYMMETRIC TEST PROBLEM (WHOSE
C     EIGENVALUES WE DO NOT KNOW)
C
C     SUBROUTINE CSMATV(W,U,CSUM)
      SUBROUTINE CMATV(W,U,CSUM)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  C2
      COMPLEX*16 U(1),W(1)
      COMPLEX*16 CL0,CL1,CR0,CR1,CSUM
C-----------------------------------------------------------------------
C
      N = KX*KY
      KX1 = KX-1
      KY1 = KY-1
      CR0 = CL0
      CR1 = CL1
C
      KK = 1
      U(KK)=(C2*W(KK)+CR0*W(KK+1)+CR1*W(KK+KX)) - CSUM*U(KK)
      KK = KX
      U(KK)=(C2*W(KK)+CL0*W(KK-1)+CR1*W(KK+KX)) - CSUM*U(KK)
      KK = N - KX + 1
      U(KK)=(C2*W(KK)+CR0*W(KK+1)+CL1*W(KK-KX)) - CSUM*U(KK)
      KK = N
      U(KK)=(C2*W(KK)+CL0*W(KK-1)+CL1*W(KK-KX)) - CSUM*U(KK)
C
      DO 10 J = 2,KX1
      KK = J
      U(KK)=(C2*W(KK)+CL0*W(KK-1)+CR0*W(KK+1)+CR1*W(KK+KX))-CSUM*U(KK)
      KK = J+N-KX
      U(KK)=(C2*W(KK)+CL0*W(KK-1)+CR0*W(KK+1)+CL1*W(KK-KX))-CSUM*U(KK)
   10 CONTINUE
C
      DO 30 J = 2,KY1
      KK = (J-1)*KX + 1
      U(KK)=(C2*W(KK)+CR0*W(KK+1)+CL1*W(KK-KX)+CR1*W(KK+KX))-CSUM*U(KK)
      KK = J*KX
      U(KK)=(C2*W(KK)+CL0*W(KK-1)+CL1*W(KK-KX)+CR1*W(KK+KX))-CSUM*U(KK)
      DO 20 I = 2,KX1
      KK = (J-1)*KX + I
      U(KK)=(C2*W(KK)+CL0*W(KK-1)+CR0*W(KK+1)+CL1*W(KK-KX)
```

```
     1 +CR1*W(KK+KX)) - CSUM*U(KK)
   20 CONTINUE
   30 CONTINUE
C
      RETURN
C
C-----------------------------------------------------------------------
      ENTRY HMATVE(C2,CL0,CL1,KX,KY)
C-----------------------------------------------------------------------
C
C-----END OF CSMATV-----------------------------------------------------
      RETURN
      END
C
C     BELOW IS USPEC AND CMATV FOR TEST MATRICES 2.  IN THIS CASE
C     THE EIGENVALUES ARE KNOWN AND WE COMPUTE THEM TO CHECK
C     VALUES OBTAINED FROM THE LANCZOS PROGRAMS.
C
C     USES 3 SUBROUTINES BELOW, USPEC CMATV EXEVG
C
C-----START OF USPEC (TEST MATRICES 2)----------------------------------
C
C     SUBROUTINE USPEC(N,MATNO)
      SUBROUTINE CSPEC(N,MATNO)
C
C-----------------------------------------------------------------------
      COMPLEX*16 CPAR,CC0,CC1,CC2
      DOUBLE PRECISION   C0,C1,C2,HALF,ONE
      REAL  EXPLAN(20)
C     COMPLEX*16  DCMPLX
C-----------------------------------------------------------------------
C     IVEC   = (0,-1,1) MEANS
C              (0) ONLY SET ENTRY FOR CMATV
C              (-1) CALCULATE EXACTEV AND MINGAPS AND STOP.
C              (1) CALCULATE EXACTEV AND MINGAPS AND THEN CONTINUE.
C-----------------------------------------------------------------------
      HALF = 0.5D0
      ONE  = 1.0D0
      CPAR = DCMPLX(ONE,ONE)
C-----------------------------------------------------------------------
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 8 (FREE FORMAT)
C
      READ(8,10) EXPLAN
      WRITE(6,10) EXPLAN
   10 FORMAT(20A4)
C
      READ(8,10) EXPLAN
      READ(8,*) KX,KY,IVEC,C0
      N = KX*KY
      C1 = HALF-C0
      C2 = ONE
      CC0 = CPAR*C0
      CC1 = CPAR*C1
      CC2 = CPAR*C2
C
```

```
      WRITE(6,20) N,KX,KY,C2,C0,C1,CPAR                                 CSL03180
   20 FORMAT(/5X,'N',4X,'KX',4X,'KY',7X,'DIAGONAL',3X,'X-CODIAGONAL',   CSL03190
     1 3X,'Y-CODIAGONAL'/3I6,3E15.8/7X,' COMPLEX SCALAR MULTIPLIER'/    CSL03200
     13X,2E15.4)                                                        CSL03210
C                                                                       CSL03220
C-----------------------------------------------------------------------CSL03230
      CALL CMATVE(CC0,CC1,CC2,KX,KY)                                    CSL03240
C-----------------------------------------------------------------------CSL03250
C                                                                       CSL03260
      IF (IVEC.EQ.0) GO TO 30                                           CSL03270
C                                                                       CSL03280
C-----------------------------------------------------------------------CSL03290
C     COMPUTE TRUE EIGENVALUES FOR CORRESPONDING REAL POISSON MATRIX    CSL03300
      CALL EXEVG(C0,C1,C2,KX,KY)                                        CSL03310
C-----------------------------------------------------------------------CSL03320
C                                                                       CSL03330
      IF (IVEC.LT.0) STOP                                               CSL03340
C                                                                       CSL03350
   30 CONTINUE                                                          CSL03360
C                                                                       CSL03370
C-----END OF USPEC------------------------------------------------------CSL03380
      RETURN                                                            CSL03390
      END                                                               CSL03400
C                                                                       CSL03410
C-----START OF CMATV (USES TEST MATRICES 2)-----------------------------CSL03420
C     CALCULATE U = A*W - SUM*U                                         CSL03430
C                                                                       CSL03440
C     SUBROUTINE CMATV(W,U,CSUM)                                        CSL03450
      SUBROUTINE CSRMAT(W,U,CSUM)                                       CSL03460
C                                                                       CSL03470
C-----------------------------------------------------------------------CSL03480
      COMPLEX*16 U(1),W(1)                                              CSL03490
      COMPLEX*16 CC0,CC1,CC2,CL0,CL1,CR0,CR1,CSUM                       CSL03500
C-----------------------------------------------------------------------CSL03510
C                                                                       CSL03520
      N = KX*KY                                                         CSL03530
      KX1 = KX-1                                                        CSL03540
      KY1 = KY-1                                                        CSL03550
      CR0 = CC0                                                         CSL03560
      CR1 = CC1                                                         CSL03570
      CL0 = CC0                                                         CSL03580
      CL1 = CC1                                                         CSL03590
C                                                                       CSL03600
      KK = 1                                                            CSL03610
      U(KK)=(CC2*W(KK)+CR0*W(KK+1)+CR1*W(KK+KX)) - CSUM*U(KK)           CSL03620
      KK = KX                                                           CSL03630
      U(KK)=(CC2*W(KK)+CL0*W(KK-1)+CR1*W(KK+KX)) - CSUM*U(KK)           CSL03640
      KK = N - KX + 1                                                   CSL03650
      U(KK)=(CC2*W(KK)+CR0*W(KK+1)+CL1*W(KK-KX)) - CSUM*U(KK)           CSL03660
      KK = N                                                            CSL03670
      U(KK)=(CC2*W(KK)+CL0*W(KK-1)+CL1*W(KK-KX)) - CSUM*U(KK)           CSL03680
C                                                                       CSL03690
      DO 10 J = 2,KX1                                                   CSL03700
      KK = J                                                            CSL03710
      U(KK)=(CC2*W(KK)+CL0*W(KK-1)+CR0*W(KK+1)+CR1*W(KK+KX))-CSUM*U(KK) CSL03720
```

```
      KK = J+N-KX
      U(KK)=(CC2*W(KK)+CL0*W(KK-1)+CR0*W(KK+1)+CL1*W(KK-KX))-CSUM*U(KK)
   10 CONTINUE
C
      DO 30 J = 2,KY1
      KK = (J-1)*KX + 1
      U(KK)=(CC2*W(KK)+CR0*W(KK+1)+CL1*W(KK-KX)+CR1*W(KK+KX))-CSUM*U(KK)
      DO 20 I = 2,KX1
      KK = KK + 1
      U(KK)=(CC2*W(KK)+CL0*W(KK-1)+CR0*W(KK+1)+CL1*W(KK-KX)
     1 +CR1*W(KK+KX)) - CSUM*U(KK)
   20 CONTINUE
      KK = KK + 1
      U(KK)=(CC2*W(KK)+CL0*W(KK-1)+CL1*W(KK-KX)+CR1*W(KK+KX))-CSUM*U(KK)
   30 CONTINUE
C
      RETURN
C
C-----------------------------------------------------------------------
      ENTRY CMATVE(CC0,CC1,CC2,KX,KY)
C-----------------------------------------------------------------------
C
C-----END OF CMATV------------------------------------------------------
      RETURN
      END
C
C-----START OF EXEVG (COMPUTES EXACT EIGENVALUES FOR TEST MATRICES 2)---
C
      SUBROUTINE EXEVG(C0,C1,C2,KX,KY)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION U(2000),MACHEP
      DOUBLE PRECISION EPSM,C0,C1,C2,T0,T1,PIK,PIL,ONE,TWO,ATOLN,EE
      REAL G(2000)
      INTEGER MP(2000)
      REAL ABS
      DOUBLE PRECISION DABS, DARCOS, DFLOAT, DCOS, DMAX1
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPSM = 2.0D0*MACHEP
C-----------------------------------------------------------------------
      N = KX*KY
      ONE  = 1.0D0
      TWO  = 2.0D0
      T0 = DARCOS(-ONE)
      T1 = DFLOAT(KX+1)
      PIK = T0/T1
      T1 = DFLOAT(KY+1)
      PIL = T0/T1
C     GENERATE EXACT EIGENVALUES
      KP = 0
      DO 20 J = 1,KY
      T1 = PIL*DFLOAT(J)
      T0 = C2 - TWO*C1*DCOS(T1)
      DO 10 I = 1,KX
```

```
      KP = KP+1                                                         CSL04280
      T1 = PIK*DFLOAT(I)                                                CSL04290
   10 U(KP) = T0 - TWO*C0*DCOS(T1)                                      CSL04300
   20 CONTINUE                                                          CSL04310
C                                                                       CSL04320
C     ORDER U VECTOR BY INCREASING ALGEBRAIC SIZE                       CSL04330
      DO 40 K = 2,N                                                     CSL04340
      KM1 = K-1                                                         CSL04350
      DO 30 L = 1,KM1                                                   CSL04360
      JJ = K-L                                                          CSL04370
      IF (U(JJ+1).GE.U(JJ)) GO TO 40                                    CSL04380
      T0 = U(JJ)                                                        CSL04390
      U(JJ) = U(JJ+1)                                                   CSL04400
   30 U(JJ+1) = T0                                                      CSL04410
   40 CONTINUE                                                          CSL04420
      ATOLN = DMAX1(DABS(U(1)),DABS(U(N)))*EPSM                         CSL04430
C                                                                       CSL04440
      WRITE(9,50)                                                       CSL04450
   50 FORMAT(' TRUE EIGENVALUES FOR POISSON'/)                          CSL04460
C                                                                       CSL04470
      WRITE(9,60)N,KX,KY,C2,C0,C1,ATOLN                                 CSL04480
      WRITE(6,60) N,KX,KY,C2,C0,C1,ATOLN                                CSL04490
   60 FORMAT(1X,'A-SIZE',2X,'X-DIM',2X,'Y-DIM'/3I7/                     CSL04500
     1 5X,'A-DIAGONAL',3X,'X-CODIAGONAL',3X,'Y-CODIAGONAL',10X,'ATOLN'/ CSL04510
     2 4E15.8)                                                          CSL04520
C                                                                       CSL04530
C     DETERMINE MULTIPLICITIES FOR EXACT EIGENVALUES                    CSL04540
      I = 1                                                             CSL04550
      IDEX = 1                                                          CSL04560
      J = 1                                                             CSL04570
      NEXACT = 0                                                        CSL04580
   70 J = J+1                                                           CSL04590
      IF (J.GT.N) GO TO 80                                              CSL04600
      EE = DABS(U(J)-U(I))                                              CSL04610
      IF (EE.GT.ATOLN) GO TO 80                                         CSL04620
      IDEX = IDEX+1                                                     CSL04630
      GO TO 70                                                          CSL04640
   80 NEXACT = NEXACT+1                                                 CSL04650
      U(NEXACT) = U(I)                                                  CSL04660
      MP(NEXACT) = IDEX                                                 CSL04670
C     MP(K) = MULTIPLICITY OF KTH EIGENVALUE CLUSTER FOR A              CSL04680
      IDEX = 1                                                          CSL04690
      I = J                                                             CSL04700
      IF (I.GT.N) GO TO 90                                              CSL04710
      GO TO 70                                                          CSL04720
   90 CONTINUE                                                          CSL04730
C                                                                       CSL04740
C     MULTIPLICITIES HAVE BEEN DETERMINED                               CSL04750
C     NEXACT = NUMBER OF DISTINCT A-EIGENVALUES                         CSL04760
C                                                                       CSL04770
C                                                                       CSL04780
      WRITE(9,100)NEXACT                                                CSL04790
      WRITE(6,100)NEXACT                                                CSL04800
  100 FORMAT(I6,' = NUMBER OF TRUE A-EIGENVALUES WHICH ARE DISTINCT'/)  CSL04810
C                                                                       CSL04820
```

```
C     MINGAP CALCULATION FOR DISTINCT A-EIGENVALUES
      NM1 = NEXACT - 1
      G(NEXACT) = U(NM1)-U(NEXACT)
      G(1) = U(2)-U(1)
C
      DO 110 J = 2,NM1
      T0 = U(J)-U(J-1)
      T1 = U(J+1)-U(J)
      G(J) = T1
      IF (T0.LT.T1) G(J) = -T0
  110 CONTINUE
C
C     NEXACT DISTINCT A-EIGENVALUES ARE IN U IN ASCENDING ORDER
C     MP = MULTIPLICITIES OF THE DISTINCT EIGENVALUES OF A
C     G = TRUE MINIMUM GAP IN A FOR EACH OF THESE EIGENVALUES
C     G < 0 INDICATES THE LEFT-HAND GAP WAS MINIMAL.
C     OUTPUT MULTIPLICITIES, DISTINCT EVS, AND MINGAPS TO FILE 9
C
      WRITE(9,120)
  120 FORMAT(5X,'I',1X,'AMULT',5X,'TRUE A-EIGENVALUE(I)',
     1 3X,'A-MINGAP(I)')
C
      WRITE(9,130)(J,MP(J),U(J),G(J), J=1,NEXACT)
  130 FORMAT(2I6,E25.16,E14.3)
C
      WRITE(9,140)
  140 FORMAT(' NEXACT DISTINCT A-EIGENVALUES ARE IN ASCENDING ORDER'/
     1 ' AMULT = MULTIPLICITIES OF THE DISTINCT EIGENVALUES OF A.'/
     2 ' A-MINGAP(I) = TRUE MINIMUM GAP IN A FOR EACH EIGENVALUE.'/
     3 ' A-MINGAP(I).LT.0 INDICATES THE LEFT-HAND GAP WAS MINIMAL.'//)
C
C     WE ORDER U VECTOR BY INCREASING SIZE OF THE GAPS
C
      DO 150 K = 1,N
  150 MP(K) = K
C
      DO 170 K = 2,N
      KM1 = K-1
C
      DO 160 L = 1,KM1
      JJ = K - L
      IF (ABS(G(JJ+1)).GE.ABS(G(JJ))) GO TO 170
      EE = U(JJ)
      U(JJ) = U(JJ+1)
      U(JJ+1) = EE
      GG = G(JJ)
      G(JJ) = G(JJ+1)
      G(JJ+1) = GG
      IEE = MP(JJ)
      MP(JJ) = MP(JJ+1)
  160 MP(JJ+1) = IEE
C
  170 CONTINUE
C
      WRITE(9,180)
```

```
  180 FORMAT(5X,'K',6X,'A-MINGAP',5X,'TRUE A-EIGENVALUE(I)',2X,'A-EVNO')CSL05380
C                                                                       CSL05390
      WRITE(9,190)(J,G(J),U(J),MP(J), J=1,NEXACT)                       CSL05400
  190 FORMAT(I6,E14.3,E25.16,I8)                                        CSL05410
C                                                                       CSL05420
      WRITE(9,200)                                                      CSL05430
  200 FORMAT(' NEXACT DISTINCT A-EIGENVALUES. GAPS IN ASCENDING ORDER'/ CSL05440
     2 ' A-MINGAP(I) = TRUE MINIMUM GAP IN A FOR EACH EIGENVALUE.'/     CSL05450
     3 ' A-MINGAP(I).LT.0 INDICATES THE LEFT-HAND GAP WAS MINIMAL.'/    CSL05460
     3 ' A-MATRIX IS BLOCK TRIDIAGONAL AND EACH DIAGONAL BLOCK IS OF ORDCSL05470
     3ER NX.'/                                                          CSL05480
     4 ' NX = NUMBER OF POINTS ON EACH X-LINE. THERE ARE NY DIAGONAL BLOCSL05490
     4CKS.'/                                                            CSL05500
     5 ' NY = NUMBER OF POINTS ON EACH Y-LINE.'/                        CSL05510
     5 ' A-DIAGONAL   = A(K,K)'/                                        CSL05520
     6 ' X-CODIAGONAL = A(I,I+1)'/                                      CSL05530
     7 ' Y-CODIAGONAL = A(I,I+NX)'/                                     CSL05540
     8 ' ----- END OF FILE 9 EXACTEV----------------------------'//)    CSL05550
C                                                                       CSL05560
C-----END OF EXEVG------------------------------------------------------CSL05570
C                                                                       CSL05580
      RETURN                                                            CSL05590
      END                                                               CSL05600
```

7.6 CSLESUB: Other Subroutines used by the Codes in Chapter 7

```
C-----CSLESUB-(NONDEFECTIVE COMPLEX SYMMETRIC MATRICES)-----------------CSL00010
C  Authors:  Jane Cullum and Ralph A. Willoughby (Deceased)             CSL00020
C            Los Alamos National Laboratory                             CSL00030
C            Los Alamos, New Mexico 87544                               CSL00040
C                                                                       CSL00050
C            E-mail:  cullumj@lanl.gov                                  CSL00060
C                                                                       CSL00070
C  These codes are copyrighted by the authors.  These codes             CSL00080
C  and modifications of them or portions of them are NOT to be          CSL00090
C  incorporated into any commercial codes or used for any other         CSL00100
C  commercial purposes such as consulting for other companies,          CSL00110
C  without legal agreements with the authors of these Codes.            CSL00120
C  If these Codes or portions of them                                   CSL00130
C  are used in other scientific or engineering research works           CSL00140
C  the names of the authors of these codes and appropriate              CSL00150
C  references to their written work are to be incorporated in the      CSL00160
C  derivative works.                                                    CSL00170
C                                                                       CSL00180
C  This header is not to be removed from these codes.                   CSL00190
C                                                                       CSL00200
C         REFERENCE:  Cullum and Willoughby, Chapter 6,                 CSL00201
C         Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsCSL00202
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   CSL00203
C         Applied Mathematics, 2002. SIAM Publications,                 CSL00204
C         Philadelphia, PA. USA                                         CSL00205
C                                                                       CSL00206
C                                                                       CSL00207
C                                                                       CSL00210
C     NONPORTABLE CONSTRUCTIONS:                                        CSL00220
C     THESE SUBROUTINES ARE NOT PORTABLE DUE TO THE USE OF THE          CSL00230
C     COMPLEX*16 VARIABLES AND THE CORRESPONDING COMPLEX FUNCTIONS,     CSL00240
C     CDABS, DCMPLX, DREAL, DIMAG.  MOREOVER, IN SUBROUTINE             CSL00250
C     COMPEV THE NONPORTABLE FORMATS (4Z20) AND (20A4) ARE USED,        CSL00260
C     AND IN SUBROUTINE CMTQL1 THE MACHINE EPSILON IS INTRODUCED        CSL00270
C     VIA A NONPORTABLE DATA DEFINITION.                                CSL00280
C                                                                       CSL00290
C     CONTAINS SUBROUTINES USED BY THE COMPLEX SYMMETRIC VERSION OF     CSL00300
C     THE LANCZOS EIGENVALUE/EIGENVECTOR CODES.                         CSL00310
C                                                                       CSL00320
C     SUBROUTINES      COMPEV, CMTQL1, INVERR, TNORM, LUMP, ISOEV AND   CSL00330
C                      COMGAP ARE USED WITH THE LANCZOS EIGENVALUE      CSL00340
C                      PROGRAM CSLEVAL.  INVERM IS USED                 CSL00350
C                      IN THE EIGENVECTOR PROGRAM CSLEVEC.  THE INNER   CSL00360
C                      PRODUCT SUBROUTINES CINPRD AND INPRDC ARE USED   CSL00370
C                      BY BOTH PROGRAMS.                                CSL00380
C                                                                       CSL00390
C-----INVERSE ITERATION ON COMPLEX SYMMETRIC T(1,MEV)-------------------CSL00400
C                                                                       CSL00410
      SUBROUTINE INVERR(ALPHA,BETA,V1,V2,VS,EPS,GR,GC,G,GG,MP,INTERC,   CSL00420
     1MEV,MMB,NDIS,NISO,N,IKL,IT,IWRITE)                                CSL00430
```

```
C                                                                       CSL00440
C-----------------------------------------------------------------------CSL00450
      COMPLEX*16  ALPHA(1),BETA(1),V1(1),V2(1),VS(1)                    CSL00460
      COMPLEX*16  U,Z,X1,RATIO,BETAM,TEMP,ZEROC                         CSL00470
      DOUBLE PRECISION EST,ESTR,ESTC,SUM,XU,NORM,TSUM,GSUM              CSL00480
      DOUBLE PRECISION  EPS,EPS3,EPS4,ZERO,ONE,GR(1),GC(1),GAP          CSL00490
      REAL G(1),GG(1)                                                   CSL00500
      INTEGER  MP(1), INTERC(1)                                         CSL00510
      REAL  ABS                                                         CSL00520
      DOUBLE PRECISION  DABS, DMIN1, DSQRT, DFLOAT, CDABS, DIMAG, DREAL CSL00530
C     COMPLEX*16 DCMPLX                                                 CSL00540
C-----------------------------------------------------------------------CSL00550
C                                                                       CSL00560
C     COMPUTES ERROR ESTIMATES FOR COMPUTED ISOLATED GOOD T-EIGENVALUES CSL00570
C     IN VS AND WRITES THESE EIGENVALUES AND ESTIMATES TO FILE 4.       CSL00580
C     BY DEFINITION A GOOD T-EIGENVALUE IS ISOLATED IF ITS CLOSEST      CSL00590
C     NEIGHBOR IS ALSO GOOD, OR IF ITS CLOSEST NEIGHBOR IS              CSL00600
C     SPURIOUS BUT THAT NEIGHBOR IS FAR ENOUGH AWAY.  SO                CSL00610
C     IN PARTICULAR, WE WILL COMPUTE ESTIMATES FOR ANY GOOD             CSL00620
C     T-EIGENVALUE THAT IS IN A CLUSTER OF GOOD T-EIGENVALUES.          CSL00630
C                                                                       CSL00640
C     USES INVERSE ITERATION ON T(1,MEV) SOLVING THE EQUATION           CSL00650
C     (T - X1*I)V2 = RIGHT-HAND SIDE (RANDOMLY-GENERATED)               CSL00660
C     FOR EACH SUCH GOOD T-EIGENVALUE X1.                               CSL00670
C                                                                       CSL00680
C     PROGRAM REFACTORS T-X1*I ON EACH ITERATION OF INVERSE ITERATION.  CSL00690
C     TYPICALLY ONLY ONE ITERATION IS NEEDED PER T-EIGENVALUE X1.       CSL00700
C                                                                       CSL00710
C     ON ENTRY AND EXIT                                                 CSL00720
C     MEV = ORDER OF T :  N = ORDER OF ORIGINAL MATRIX A                CSL00730
C     ALPHA, BETA CONTAIN THE NONZERO ENTRIES OF THE T-MATRIX           CSL00740
C     VS = COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)                    CSL00750
C     MP = T-MULTIPLICITY OF EACH T-EIGENVALUE IN VS. MP(I) = -1 MEANS  CSL00760
C          VS(I) IS A GOOD T-EIGENVALUE BUT THAT IT IS SITTING CLOSE TO CSL00770
C          A SPURIOUS T-EIGENVALUE.  MP(I) = 0 MEANS VS(I) IS SPURIOUS. CSL00780
C          ESTIMATES ARE COMPUTED ONLY FOR THOSE T-EIGENVALUES          CSL00790
C          WITH MP(I) = 1. FLAGGING WAS DONE IN SUBROUTINE ISOEV        CSL00800
C          PRIOR TO ENTERING INVERR.                                    CSL00810
C     NISO = NUMBER OF ISOLATED GOOD T-EIGENVALUES CONTAINED IN VS      CSL00820
C     NDIS =  NUMBER OF DISTINCT T-EIGENVALUES IN VS                    CSL00830
C     IKL = SEED FOR RANDOM NUMBER GENERATOR                            CSL00840
C     EPS = 2. * MACHINE EPSILON                                        CSL00850
C                                                                       CSL00860
C     IN PROGRAM:                                                       CSL00870
C     ITER = MAXIMUM NUMBER OF INVERSE ITERATION STEPS ALLOWED FOR EACH CSL00880
C            X1.  ITER = IT ON ENTRY.                                   CSL00890
C     GR,GC = ARRAYS OF DIMENSION AT LEAST MEV + NISO.  USED TO STORE   CSL00900
C         RANDOMLY-GENERATED RIGHT-HAND SIDE.  THIS IS NOT              CSL00910
C         REGENERATED FOR EACH X1. G IS ALSO USED TO STORE ERROR        CSL00920
C         ESTIMATES AS THEY ARE COMPUTED FOR LATER PRINTOUT.            CSL00930
C     V1,V2 = WORK SPACES USED IN THE FACTORIZATION OF T(1,MEV).        CSL00940
C     AT THE END OF THE INVERSE ITERATION COMPUTATION FOR X1, V2        CSL00950
C     CONTAINS THE UNIT EIGENVECTOR OF T(1,MEV) CORRESPONDING TO X1.    CSL00960
C     V1 AND V2 MUST BE OF DIMENSION AT LEAST MEV.                      CSL00970
C                                                                       CSL00980
```

```
C     ON EXIT                                                           CSL00990
C     GG(J) = MINIMUM GAP IN T(1,MEV) FOR EACH VS(J), J=1,NDIS          CSL01000
C     G(I) = |BETAM|*|V2(MEV)| = ERROR ESTIMATE FOR ISOLATED GOOD       CSL01010
C              T-EIGENVALUES, WHERE I = 1,NISO  AND  BETAM = BETA(MEV+1)CSL01020
C              T(1,MEV) CORRESPONDING TO ITH ISOLATED GOOD T-EIGENVALUE.CSL01030
C                                                                       CSL01040
C     IF FOR SOME X1 IT.GT.ITER THEN THE ERROR ESTIMATE IN G IS MARKED  CSL01050
C     WITH A - SIGN.                                                    CSL01060
C                                                                       CSL01070
C     V2 = ISOLATED GOOD T-EIGENVALUES                                  CSL01080
C     V1 = MINIMAL T-GAPS FOR THE T-EIGENVALUES IN V2.                  CSL01090
C     THESE ARE CONSTRUCTED FOR WRITE-OUT PURPOSES ONLY AND NOT         CSL01100
C     NEEDED ELSEWHERE IN THE PROGRAM.                                  CSL01110
C-----------------------------------------------------------------------CSL01120
C                                                                       CSL01130
C     LABEL OUTPUT FILE  4                                              CSL01140
      IF (MMB.EQ.1) WRITE(4,10)                                         CSL01150
   10 FORMAT(' INVERSE ITERATION ERROR ESTIMATES'/)                     CSL01160
C                                                                       CSL01170
C     FILE 6 (TERMINAL) OUTPUT OF ERROR ESTIMATES                       CSL01180
      IF (IWRITE.NE.0.AND.NISO.NE.0) WRITE(6,20)                        CSL01190
   20 FORMAT(/' INVERSE ITERATION ERROR ESTIMATES'/'  JISO',' JDIST',8X CSL01200
     1,'GOOD T-EIGENVALUE',4X,'BETAM*UM',5X,'TMINGAP')                  CSL01210
C                                                                       CSL01220
C     INITIALIZATION AND PARAMETER SPECIFICATION                        CSL01230
      ZERO = 0.0D0                                                      CSL01240
      ONE = 1.0D0                                                       CSL01250
      ZEROC = DCMPLX(ZERO,ZERO)                                         CSL01260
      NG = 0                                                            CSL01270
      NISO = 0                                                          CSL01280
      ITER = IT                                                         CSL01290
      MP1 = MEV+1                                                       CSL01300
      MM1 = MEV-1                                                       CSL01310
      BETAM = BETA(MP1)                                                 CSL01320
      BETA(MP1) = ZEROC                                                 CSL01330
C                                                                       CSL01340
C     CALCULATE SCALE AND TOLERANCES                                    CSL01350
      TSUM = CDABS(ALPHA(1))                                            CSL01360
      DO 30 I = 2,MEV                                                   CSL01370
   30 TSUM = TSUM + CDABS(ALPHA(I)) + CDABS(BETA(I))                    CSL01380
C                                                                       CSL01390
      EPS3 = EPS*TSUM                                                   CSL01400
      EPS4 = DFLOAT(MEV)*EPS3                                           CSL01410
C                                                                       CSL01420
C     GENERATE SCALED RANDOM RIGHT-HAND SIDE                            CSL01430
      ILL = IKL                                                         CSL01440
C                                                                       CSL01450
C-----------------------------------------------------------------------CSL01460
      CALL GENRAN(ILL,G,MEV)                                            CSL01470
C-----------------------------------------------------------------------CSL01480
C                                                                       CSL01490
      DO 40 I = 1,MEV                                                   CSL01500
   40 GR(I) = G(I)                                                      CSL01510
C                                                                       CSL01520
C-----------------------------------------------------------------------CSL01530
```

```
      CALL GENRAN(ILL,G,MEV)
C-----------------------------------------------------------------------
C
      DO 50 I = 1,MEV
   50 GC(I) = G(I)
C
      GSUM = ZERO
      DO 60 I = 1,MEV
   60 GSUM = GSUM + DABS(GR(I)) + DABS(GC(I))
      GSUM = EPS4/GSUM
C
      DO 70 I = 1,MEV
      GR(I) = GSUM*GR(I)
   70 GC(I) = GSUM*GC(I)
C
C     LOOP ON ISOLATED GOOD T-EIGENVALUES IN VS (MP(I) = 1) TO
C     CALCULATE CORRESPONDING UNIT EIGENVECTOR OF T(1,MEV)
C
      DO 200 JEV = 1,NDIS
      IF (MP(JEV).EQ.0) GO TO 200
      NG = NG + 1
      IF (MP(JEV).NE.1) GO TO 200
      IT = 1
      NISO = NISO + 1
      X1 = VS(JEV)
C
C     INITIALIZE RIGHT HAND SIDE FOR INVERSE ITERATION
C     AND THE FLAG ON WHICH ROWS ARE INTERCHANGED
      DO 80 I = 1,MEV
      INTERC(I) = 0
   80 V2(I) = DCMPLX(GR(I),GC(I))
C
C     TRIANGULAR FACTORIZATION WITH NEAREST NEIGHBOR PIVOT
C     STRATEGY. INTERCHANGES ARE LABELLED BY SETTING INTERC = 1.
C
   90 CONTINUE
      U = ALPHA(1)-X1
      Z = BETA(2)
C
      DO 110 I = 2,MEV
      IF (CDABS(BETA(I)).GT.CDABS(U)) GO TO 100
C     NO INTERCHANGE
      V1(I-1) = Z/U
      V2(I-1) = V2(I-1)/U
      V2(I) = V2(I)-BETA(I)*V2(I-1)
      RATIO = BETA(I)/U
      U = ALPHA(I)-X1-Z*RATIO
      Z = BETA(I+1)
      GO TO 110
  100 CONTINUE
C     INTERCHANGE CASE
      RATIO = U/BETA(I)
      INTERC(I) = 1
      V1(I-1) = ALPHA(I)-X1
      U = Z-RATIO*V1(I-1)
```

```
      Z = -RATIO*BETA(I+1)                                              CSL02090
      TEMP = V2(I-1)                                                    CSL02100
      V2(I-1) = V2(I)                                                   CSL02110
      V2(I) = TEMP-RATIO*V2(I)                                          CSL02120
  110 CONTINUE                                                          CSL02130
      IF (CDABS(U).EQ.ZERO) U = DCMPLX(EPS3,EPS3)                       CSL02140
C                                                                       CSL02150
C     SMALLNESS TEST AND DEFAULT VALUE FOR LAST COMPONENT               CSL02160
C     PIVOT(I-1) = BETA(I) FOR INTERCHANGE CASE                         CSL02170
C     (I-1,I+1) ELEMENT IN RIGHT FACTOR = BETA(I+1)                     CSL02180
C     END OF FACTORIZATION AND FORWARD SUBSTITUTION                     CSL02190
C                                                                       CSL02200
C     BACK SUBSTITUTION                                                 CSL02210
      V2(MEV) = V2(MEV)/U                                               CSL02220
      DO 130 II = 1,MM1                                                 CSL02230
      I = MEV-II                                                        CSL02240
      IF (INTERC(I+1).EQ.1) GO TO 120                                   CSL02250
C     NO INTERCHANGE                                                    CSL02260
      V2(I) = V2(I)-V1(I)*V2(I+1)                                       CSL02270
      GO TO 130                                                         CSL02280
C     INTERCHANGE CASE                                                  CSL02290
  120 CONTINUE                                                          CSL02300
      V2(I) = (V2(I)-V1(I)*V2(I+1)-BETA(I+2)*V2(I+2))/BETA(I+1)         CSL02310
  130 CONTINUE                                                          CSL02320
C                                                                       CSL02330
C     TESTS FOR CONVERGENCE OF INVERSE ITERATION                        CSL02340
C     IF SUM |V2| COMPS. LE. 1 AND IT. LE. ITER DO ANOTHER INVIT STEP   CSL02350
C                                                                       CSL02360
      NORM = CDABS(V2(MEV))                                             CSL02370
      DO 140 II = 1,MM1                                                 CSL02380
      I = MEV-II                                                        CSL02390
  140 NORM = NORM + CDABS(V2(I))                                        CSL02400
C                                                                       CSL02410
      IF (NORM.GE.ONE) GO TO 160                                        CSL02420
      IT = IT+1                                                         CSL02430
      IF (IT.GT.ITER) GO TO 160                                         CSL02440
      XU = EPS4/NORM                                                    CSL02450
C                                                                       CSL02460
      DO 150 I = 1,MEV                                                  CSL02470
  150 V2(I) = V2(I)*XU                                                  CSL02480
C                                                                       CSL02490
      GO TO 90                                                          CSL02500
C     ANOTHER INVERSE ITERATION STEP                                    CSL02510
C                                                                       CSL02520
C     INVERSE ITERATION FINISHED                                        CSL02530
C     NORMALIZE COMPUTED T-EIGENVECTOR : V2 = V2/||V2||                  CSL02540
  160 CONTINUE                                                          CSL02550
C                                                                       CSL02560
C-----------------------------------------------------------------------CSL02570
      CALL CINPRD(V2,V2,SUM,MEV)                                        CSL02580
C-----------------------------------------------------------------------CSL02590
C                                                                       CSL02600
      SUM = ONE/DSQRT(SUM)                                              CSL02610
C                                                                       CSL02620
      DO 170 II = 1,MEV                                                 CSL02630
```

```
  170 V2(II) = SUM*V2(II)
C
C     SAVE ERROR ESTIMATE FOR LATER OUTPUT
      EST = CDABS(BETAM)*CDABS(V2(MEV))
      ESTR = DABS(DREAL(V2(MEV)))
      ESTC = DABS(DIMAG(V2(MEV)))
      GSUM = CDABS(BETAM)
      IF (IT.GT.ITER) EST = -EST
      G(NISO) = EST
      IF (IWRITE.EQ.0) GO TO 200
C
C     FILE 6 (TERMINAL) OUTPUT OF ERROR ESTIMATES.
      GAP = GG(JEV)
      WRITE(6,180) NISO,JEV,X1,EST,GAP
  180 FORMAT(2I6,2E20.12,2E12.3)
      WRITE(6,190) JEV, X1, EST,ESTR,ESTC
  190 FORMAT(I6,2E20.12,3E11.3)
C
  200 CONTINUE
C
C     END ERROR ESTIMATE LOOP ON ISOLATED GOOD T-EIGENVALUES.
C     GENERATE DISTINCT MINGAPS FOR T(1,MEV).  THIS IS USEFUL AS AN
C     INDICATOR OF THE GOODNESS OF THE INVERSE ITERATION ESTIMATES.
C     TRANSFER ISOLATED GOOD T-EIGENVALUES AND CORRESPONDING TMINGAPS
C     TO V2 AND V1 FOR OUTPUT PURPOSES ONLY.
C
      ISO = 0
      DO 210 J = 1,NDIS
      IF (MP(J).NE.1) GO TO 210
      ISO = ISO+1
      GR(ISO) = GG(J)
      V2(ISO) = VS(J)
  210 CONTINUE
      IF(NISO.EQ.0) GO TO 270
C
C     ERROR ESTIMATES ARE WRITTEN TO FILE 4
      WRITE(4,220)MEV,NDIS,NG,NISO,N,IKL,ITER,GSUM
  220 FORMAT(1X,'TSIZE',2X,'NDIS',1X,'NGOOD',2X,'NISO',1X,'ASIZE'/5I6/
     1 4X,'RHSEED',2X,'MXINIT',5X,'BETAM'/I10,I8,E10.3)
C
      WRITE(4,230)
  230 FORMAT(2X,'GOODEVNO',11X,'R(GOODEV)',11X,'I(GOODEV)',
     1 6X,'BETAM*UM',7X,'TMINGAP')
C
      ISPUR = 0
      I = 0
      DO 260 J = 1,NDIS
      IF(MP(J).NE.0) GO TO 240
      ISPUR = ISPUR + 1
      GO TO 260
  240 IF(MP(J).NE.1) GO TO 260
      I = I + 1
      IGOOD = J - ISPUR
      WRITE(4,250) IGOOD,V2(I),G(I),GR(I)
  250 FORMAT(I10,2E20.12,2E14.3)
```

```
  260 CONTINUE                                                          CSL03190
      GO TO 290                                                         CSL03200
C                                                                       CSL03210
  270 WRITE(4,280)                                                      CSL03220
  280 FORMAT(/' THERE ARE NO ISOLATED T-EIGENVALUES SO NO ERROR ESTIMATECSL03230
     1S WERE COMPUTED')                                                 CSL03240
C     RESTORE BETA(MEV+1) = BETAM                                       CSL03250
  290 BETA(MP1) = BETAM                                                 CSL03260
C-----END OF INVERR-----------------------------------------------------CSL03270
      RETURN                                                            CSL03280
      END                                                               CSL03290
C-----START OF TNORM----------------------------------------------------CSL03300
C                                                                       CSL03310
      SUBROUTINE TNORM(ALPHA,BETA,BMIN,TMAX,MEV,IB)                     CSL03320
C                                                                       CSL03330
C-----------------------------------------------------------------------CSL03340
      COMPLEX*16  ALPHA(1),BETA(1)                                      CSL03350
      DOUBLE PRECISION  TMAX,BMIN,BMAX,BSIZE,BTOL,ABATA,AALFA           CSL03360
      DOUBLE PRECISION  DMAX1, CDABS                                    CSL03370
C     COMPLEX*16 DCMPLX                                                 CSL03380
C-----------------------------------------------------------------------CSL03390
C     IN REAL SYMMETRIC AND HERMITIAN VERSIONS TMAX IS USED             CSL03400
C     TO DETERMINE THE TOLERANCES USED IN THE T-MULTIPLICITY AND IN     CSL03410
C     THE SPURIOUS TESTS.  FOR THE COMPLEX SYMMETRIC CASE WE            CSL03420
C     HAVE TO COMPUTE ALL OF THE T-EIGENVALUES SO WE USE THEM INSTEAD   CSL03430
C     OF TMAX TO DETERMINE THESE TOLERANCES.  WE USE TMAX TO            CSL03440
C     CHECK THE RELATIVE SIZES OF THE BETA(K), K=1,...,MEV AS A         CSL03450
C     TEST ON THE LOCAL ORTHOGONALITY OF THE LANCZOS VECTORS.           CSL03460
C                                                                       CSL03470
C             TMAX = MAX (|ALPHA(I)|, |BETA(I)|,  I=1,MEV)              CSL03480
C             BMIN = MIN (|BETA(I)|, I=2,MEV)                           CSL03490
C             BSIZE = BMIN/TMAX                                         CSL03500
C             |IB| = INDEX OF MINIMAL(BETA)                             CSL03510
C             IB < 0 IF BMIN/TMAX < BTOL                                CSL03520
C-----------------------------------------------------------------------CSL03530
C     SPECIFY PARAMETERS                                                CSL03540
      IB = 2                                                            CSL03550
      BTOL = BMIN                                                       CSL03560
      BMIN = CDABS(BETA(2))                                             CSL03570
      BMAX = BMIN                                                       CSL03580
      TMAX = CDABS(ALPHA(1))                                            CSL03590
C                                                                       CSL03600
      DO 20 I = 2,MEV                                                   CSL03610
      ABATA = CDABS(BETA(I))                                            CSL03620
      IF (ABATA.GE.BMIN) GO TO 10                                       CSL03630
      IB = I                                                            CSL03640
      BMIN = ABATA                                                      CSL03650
   10 AALFA = CDABS(ALPHA(I))                                           CSL03660
      TMAX = DMAX1(TMAX,AALFA)                                          CSL03670
      BMAX = DMAX1(ABATA,BMAX)                                          CSL03680
   20 CONTINUE                                                          CSL03690
      TMAX = DMAX1(BMAX,TMAX)                                           CSL03700
C                                                                       CSL03710
C     TEST OF LOCAL ORTHOGONALITY USING SCALED BETAS                    CSL03720
      BSIZE = BMIN/TMAX                                                 CSL03730
```

```
      IF (BSIZE.GE.BTOL) GO TO 40
C
C     DEFAULT.  BSIZE IS SMALLER THAN TOLERANCE BTOL SPECIFIED IN MAIN
C     PROGRAM.  PROGRAM TERMINATES FOR USER TO DECIDE WHAT TO DO
C     BECAUSE LOCAL ORTHOGONALITY OF THE LANCZOS VECTORS COULD BE
C     LOST.
C
      IB = -IB
      WRITE(6,30) MEV
   30 FORMAT(/' BETA TEST INDICATES POSSIBLE LOSS OF LOCAL ORTHOGONALITY
     1 OVER 1ST',I6,' LANCZOS VECTORS'/)
C
   40 CONTINUE
C
      WRITE(6,50) IB
   50 FORMAT(/' MINIMUM BETA RATIO OCCURS AT',I6,' TH BETA'/)
C
      WRITE(6,60) MEV,BMIN,TMAX,BSIZE
   60 FORMAT(/1X,'TSIZE',6X,'MIN BETA',5X,'TKMAX',6X,'MIN RATIO'/
     1 I6,E14.3,E10.3,E15.3/)
C
C-----END OF TNORM-----------------------------------------------------
      RETURN
      END
C
C-----START OF LUMP----------------------------------------------------
C
      SUBROUTINE LUMP(VC,V1,VA,RELTOL,SPUTOL,SCALE2,LINDEX,TFLAG,LOOP)
C
C---------------------------------------------------------------------
      COMPLEX*16 VC(1),V1(1),ZEROC,SUMC
      DOUBLE PRECISION  VA(1),RELTOL,SPUTOL,SCALE2
      DOUBLE PRECISION  THOLD,TH1,TH2,DGAP,ZERO,ONE
      INTEGER  LINDEX(1),TFLAG(1)
      DOUBLE PRECISION  DFLOAT, DMAX1, CDABS
C     COMPLEX*16 DCMPLX
C---------------------------------------------------------------------
C     VC(J) = JTH DISTINCT T-EIGENVALUE,  VA(J) = |VC(J)|, IN ORDER
C     OF INCREASING MAGNITUDE.
C     LINDEX(J) = T-MULTIPLICITY OF JTH DISTINCT T-EIGENVALUE
C     LOOP = NUMBER OF DISTINCT T-EIGENVALUES
C     LUMP 'COMBINES' COMPUTED 'GOOD' T-EIGENVALUES THAT ARE 'TOO CLOSE'
C     VALUE OF RELTOL IS 1.D-8.
C
C     IF IN A SET OF T-EIGENVALUES TO BE COMBINED THERE IS AN EIGENVALUE
C     WITH LINDEX=1, THEN THE VALUE OF THE COMBINED T-EIGENVALUES IS SET
C     EQUAL TO THE VALUE OF THAT EIGENVALUE.  NOTE THAT IF A SPURIOUS
C     T-EIGENVALUE IS TO BE 'COMBINED' WITH A GOOD EIGENVALUE, THEN THIS
C     IS DONE ONLY BY INCREASING THE INDEX, LINDEX, FOR THAT EIGENVALUE
C     NUMERICAL VALUES OF SPURIOUS T-EIGENVALUES ARE NEVER COMBINED WITH
C     THOSE OF GOOD T-EIGENVALUES.
C---------------------------------------------------------------------
      ZERO = 0.0D0
      ONE = 1.D0
      ZEROC = DCMPLX(ZERO,ZERO)
```

```
      TH2 = SCALE2*SPUTOL
      DO 10 K = 1,LOOP
   10 TFLAG(K) = 0
      NLOOP = 0
      J = 0
   20 J = J+1
      IF (J.GT.LOOP) GO TO 130
      IF (TFLAG(J).EQ.1) GO TO 20
      NLOOP = NLOOP + 1
      TFLAG(J) = 1
      V1(1) = VC(J)
      ICOUNT = 1
      JN = LINDEX(J)
      TH1 = RELTOL*VA(J)
      THOLD = DMAX1(TH1,TH2)
C     THOLD = RELTOL*DMAX1(ONE,VA(J))
      IF (JN.EQ.0) GO TO 30
      INDSUM = JN
      ISPUR = 0
      SUMC = DFLOAT(JN)*VC(J)
      GO TO 40
   30 INDSUM = 1
      ISPUR = 1
      SUMC = ZEROC
   40 IF (J.EQ.LOOP) GO TO 70
      I = J
   50 I = I + 1
      IF (I.GT.LOOP) GO TO 70
      IF (TFLAG(I).EQ.1) GO TO 50
      DGAP = VA(I) - VA(J)
      IF (DGAP.GE.THOLD) GO TO 70
      DGAP = CDABS(VC(I)-VC(J))
      IF (DGAP.GE.THOLD) GO TO 50
C     LUMP VC(I) WITH VC(J)
      ICOUNT = ICOUNT + 1
      TFLAG(I) = 1
      V1(ICOUNT) = VC(I)
      IN = LINDEX(I)
      IF (IN.NE.0) GO TO 60
      ISPUR = ISPUR + 1
      INDSUM = INDSUM + 1
      GO TO 50
   60 INDSUM = INDSUM + IN
      SUMC = SUMC + DFLOAT(IN)*VC(I)
      GO TO 50
C     COMPUTE THE 'COMBINED' T-EIGENVALUE AND THE RESULTING
C     T-MULTIPLICITY
   70 CONTINUE
C
C     IF (ICOUNT.GT.1) WRITE(6,80) (K,V1(K), K = 1,ICOUNT)
   80 FORMAT(/' T-EIGENVALUES ARE LUMPED '/
     1 5X,'J',12X,'REAL(EV)',12X,'IMAG(EV)'/(I6,2E20.12))
C
      IF (ICOUNT.EQ.1) INDSUM = JN
      IDIF = INDSUM - ISPUR
```

```
      IF (IDIF.EQ.0.AND.ICOUNT.GT.1) GO TO 90
      IF (ICOUNT.EQ.1) GO TO 90
C     ICOUNT.GT.1 AND IDIF.GT.0
      SUMC = SUMC/DFLOAT(IDIF)
      VC(NLOOP) = SUMC
      VA(NLOOP) = CDABS(SUMC)
      GO TO 100
   90 VC(NLOOP) = VC(J)
      VA(NLOOP) = VA(J)
  100 LINDEX(NLOOP) = INDSUM
      GO TO 20
C     INDEX J IS FINISHED
C
C     ON RETURN VC CONTAINS THE DISTINCT T-EIGENVALUES  VA = |VC|
C     LINDEX CONTAINS THE CORRESPONDING T-MULTIPLICITIES
C
  130 CONTINUE
      LOOP = NLOOP
      RETURN
C-----END OF LUMP-----------------------------------------------------
      END
C
C-----START OF ISOEV--------------------------------------------------
C
      SUBROUTINE ISOEV(VS,GR,GG,GAPTOL,SPUTOL,SCALE1,MP,NDIS,NG,NISO)
C
C--------------------------------------------------------------------
      COMPLEX*16 VS(1),T0
      DOUBLE PRECISION  GR(1),SPUTOL,GAPTOL,SCALE1,TEMP,TOL,TJ,DGAP,ONE
      REAL GG(1)
      INTEGER  MP(1)
      REAL  ABS
      DOUBLE PRECISION  DMAX1, CDABS
C--------------------------------------------------------------------
C     USE TMINGAPS TO LABEL THE ISOLATED GOOD T-EIGENVALUES
C     THAT ARE VERY CLOSE TO SPURIOUS ONES.  ERROR ESTIMATES
C     WILL NOT BE COMPUTED FOR THESE T-EIGENVALUES.
C
C     ON ENTRY AND EXIT
C     VS CONTAINS THE COMPUTED DISTINCT EIGENVALUES OF T(1,MEV)
C     GR(K) = |VS(K)|, K = 1,NDIS, GR(K).LE.GR(K+1)
C     GG(K) = MIN(J.NE.K,|VS(K)-VS(J)|)   MINGAP
C     MP CONTAINS THE CORRESPONDING T-MULTIPLICITIES
C     NDIS = NUMBER OF DISTINCT T-EIGENVALUES
C     GAPTOL = RELATIVE GAP TOLERANCE SET IN MAIN
C
C     ON EXIT
C     MP(J) IS NOT CHANGED EXCEPT THAT  MP(J)=-1, IF MP(J)=1,
C     AND A SPURIOUS T-EIGENVALUE IS TOO CLOSE.
C
C     IF MP(I)=-1 THAT SIMPLE GOOD T-EIGENVALUE WILL BE SKIPPED
C     IN THE SUBSEQUENT ERROR ESTIMATE COMPUTATIONS IN INVERR
C     THAT IS, WE COMPUTE ERROR ESTIMATES ONLY FOR THOSE GOOD
C     T-EIGENVALUES WITH MP(J)=1.
C--------------------------------------------------------------------
```

```
      ONE = 1.0D0                                                       CSL05390
      DGAP = SCALE1*SPUTOL                                              CSL05400
      NISO = 0                                                          CSL05410
      NG = 0                                                            CSL05420
      DO 40 J = 1,NDIS                                                  CSL05430
      IF (MP(J).EQ.0) GO TO 40                                          CSL05440
      NG = NG+1                                                         CSL05450
      IF (MP(J).NE.1) GO TO 40                                          CSL05460
      TJ = GR(J)                                                        CSL05470
      T0 = VS(J)                                                        CSL05480
      TOL = DMAX1(DGAP,GAPTOL*TJ)                                       CSL05490
C     TOL = DMAX1(ONE,TJ)*GAPTOL                                        CSL05500
C     VS(J) IS NEXT SIMPLE GOOD T-EIGENVALUE                            CSL05510
      NISO = NISO + 1                                                   CSL05520
      IF (ABS(GG(J)).GT.TOL) GO TO 40                                   CSL05530
      I = J                                                             CSL05540
   10 I = I-1                                                           CSL05550
      IF (I.LT.1) GO TO 20                                              CSL05560
      IF (TJ-GR(I).GT.TOL) GO TO 20                                     CSL05570
      IF (MP(I).NE.0) GO TO 10                                          CSL05580
      TEMP = CDABS(T0-VS(I))                                            CSL05590
      IF (TEMP.GT.TOL) GO TO 10                                         CSL05600
      MP(J) = -MP(J)                                                    CSL05610
      NISO = NISO-1                                                     CSL05620
      GO TO 40                                                          CSL05630
   20 I = J                                                             CSL05640
   30 I = I+1                                                           CSL05650
      IF (I.GT.NDIS) GO TO 40                                           CSL05660
      IF (GR(I)-TJ.GT.TOL) GO TO 40                                     CSL05670
      IF (MP(I).NE.0) GO TO 30                                          CSL05680
      TEMP = CDABS(T0-VS(I))                                            CSL05690
      IF (TEMP.GT.TOL) GO TO 30                                         CSL05700
      MP(J) = -MP(J)                                                    CSL05710
      NISO = NISO-1                                                     CSL05720
   40 CONTINUE                                                          CSL05730
C                                                                       CSL05740
C-----END OF ISOEV------------------------------------------------------CSL05750
      RETURN                                                            CSL05760
      END                                                               CSL05770
C---COMPEV------------------------------------------------------------- CSL05780
C                                                                       CSL05790
      SUBROUTINE COMPEV(ALPHA,BETA,V1,V2,VS,EVMAG,MULTOL,SPUTOL,        CSL05800
     1MP,T2FLAG,MEV,NDIS,SAVTEV)                                        CSL05810
C                                                                       CSL05820
C     USES COMPLEX SYMMETRIC VERSION OF IMTQL1, CMTQL1, TO              CSL05830
C     COMPUTE EIGENVALUES OF THE T-MATRIX T(1,MEV).                     CSL05840
C                                                                       CSL05850
C-----------------------------------------------------------------------CSL05860
      COMPLEX*16 ALPHA(1),BETA(1),VS(1),V1(1),V2(1),EVAL,CTEMP          CSL05870
      DOUBLE PRECISION EVMAG(1)                                         CSL05880
      DOUBLE PRECISION TEMP,DGAP,TOL,DELMIN                             CSL05890
      DOUBLE PRECISION MULTOL,SPUTOL,EVALR,EVALC                        CSL05900
      INTEGER MP(1),T2FLAG(1),SAVTEV                                    CSL05910
      DOUBLE PRECISION CDABS, DFLOAT                                    CSL05920
C-----------------------------------------------------------------------CSL05930
```

```
C
      MEV1 = MEV - 1
C
      IF (SAVTEV.GE.0) GO TO 40
C
      READ(10,10) MEV
   10 FORMAT(I6)
   20 FORMAT(20A4)
      MEV1 = MEV - 1
      READ(10,30) (VS(K), K = 1,MEV)
   30 FORMAT(4Z20)
      READ(10,20) EXPLAN
      READ(10,20) EXPLAN
      READ(10,30) (V2(K), K = 1,MEV1)
      GO TO 90
C
   40 CONTINUE
C
      DO 50 J = 1,MEV
      VS(J) = ALPHA(J)
   50 V1(J) = BETA(J)
C
      WRITE(6,60) MEV
   60 FORMAT(/' COMPUTE EIGENVALUES OF T(1,',I4,') USING CMTQL1'/)
C
C-----------------------------------------------------------------------
      CALL CMTQL1(MEV,VS,V1,IERR)
C-----------------------------------------------------------------------
C
C     WRITE(6,70) IERR
   70 FORMAT(' T-EIGENVALUES VIA CMTQL1'/' IERR = ',I6/)
C
      IF (IERR.EQ.0) GO TO 90
C
      WRITE(6,80)
   80 FORMAT(' ON RETURN FROM CMTQL1 ERROR FLAG WAS NOT ZERO'/)
      GO TO 410
C
   90 CONTINUE
C
C     T-EIGENVALUES ARE IN VS IN INCREASING ORDER OF MAGNITUDE
      DO 100 J = 1,MEV
  100 EVMAG(J) = CDABS(VS(J))
C
C     THE MAGNITUDES OF THE T-EIGENVALUES ARE IN EVMAG, IN ORDER OF
C     INCREASING MAGNITUDE
C     WRITE(13,105) (EVMAG(J), J = 1,MEV)
C 105 FORMAT(' MAGNITUDES OF T-EIGENVALUES'/(4E20.12))
C
      IF(SAVTEV.NE.1) GO TO 130
      WRITE(10,110) MEV
  110 FORMAT(I6,' = ORDER OF T-MATRIX, T-EIGVALS =')
      WRITE(10,120) (VS(J), J = 1,MEV)
C 120 FORMAT(4Z20)
  120 FORMAT(4E20.12)
```

```
C
C
  130 CONTINUE
      MULTOL = MULTOL*EVMAG(MEV)
      SPUTOL = SPUTOL*EVMAG(MEV)
      TOL = 1000.0D0*SPUTOL
      WRITE(6,140) MULTOL,SPUTOL
  140 FORMAT(/' TOLERANCES USED IN T-MULTIPLICITY AND SPURIOUS TESTS ='
     1 ,2E10.3/)
C
C     T-MULTIPLICITY DETERMINATION
      J = 0
      NDIS = 0
      DO 150 I = 1,MEV
  150 T2FLAG(I) = 0
C
  160 J = J+1
      IF (J.GT.MEV) GO TO 190
      IF (T2FLAG(J).EQ.1) GO TO 160
      CTEMP = VS(J)
      EVAL = CTEMP
      TEMP = EVMAG(J)
      NDIS = NDIS + 1
      INDEX = 1
      T2FLAG(J) = 1
      I = J
  170 I = I+1
      IF (I.GT.MEV) GO TO 180
      IF (T2FLAG(I).EQ.1) GO TO 170
      DGAP = EVMAG(I)-TEMP
      IF (DGAP.GT.MULTOL) GO TO 180
      DGAP = CDABS(EVAL-VS(I))
      IF (DGAP.GT.MULTOL) GO TO 170
C     T-MULTIPLICITY INCREASES
      INDEX = INDEX + 1
      CTEMP = CTEMP + VS(I)
      T2FLAG(I) = 1
      GO TO 170
C     T-MULTIPLICITY FOR VS(NDIS) HAS BEEN DETERMINED
  180 VS(NDIS) = CTEMP/DFLOAT(INDEX)
      MP(NDIS) = INDEX
      GO TO 160
  190 CONTINUE
C     T-MULTIPLICITY CALCULATION IS COMPLETE
C
C     T(2,MEV) EIGENVALUE CALCULATION AND SPURIOUS TESTS
C
      IF (SAVTEV.LT.0) GO TO 240
C
      WRITE(6,200) MEV1
  200 FORMAT(/' COMPUTE T(2,',I4,') EIGENVALUES'/)
C
      DO 210 J = 1,MEV1
      JP1 = J+1
      V2(J) = ALPHA(JP1)
```

```
  210 V1(J) = BETA(JP1)
C
C-----------------------------------------------------------------------
      CALL CMTQL1(MEV1,V2,V1,IERR)
C-----------------------------------------------------------------------
C
C     WRITE(6,220) IERR
  220 FORMAT(' T2-HAT EIGENVALUES VIA CMTQL1'/' IERR = ',I6/)
C
      IF (IERR.EQ.0) GO TO 240
C
      WRITE(6,230)
  230 FORMAT(' ON RETURN FROM CMTQL1 ERROR FLAG WAS NOT ZERO'/)
      GO TO 410
C
  240 CONTINUE
C
      DO 250 J = 1,MEV1
  250 EVMAG(J) = CDABS(V2(J))
C
C     WRITE(13,255) (EVMAG(J), J = 1,MEV)
C 255 FORMAT(/' MAGNITUDES OF T2 EIGENVALUES'/(4E20.12))
C
      IF(SAVTEV.NE.1) GO TO 270
      WRITE(10,260) MEV1
  260 FORMAT(/I6,' = ORDER OF T2-HAT, T2EIGVALS = ')
      WRITE(10,120) (V2(J), J = 1,MEV1)
  270 CONTINUE
C
C     SPURIOUS TESTS
      DO 280 I = 1,MEV1
  280 T2FLAG(I) = 0
C
C     GO THROUGH THE EIGENVALUES OF T2-HAT.  FIND THE CLOSEST EIGENVALUE
C     OF T(1,MEV).  IF IT IS T-MULTIPLE GO ON.  IF IT IS SIMPLE DECLARE
C     SPURIOUS WHENEVER DELMIN < SPUTOL BY SETTING MP(I) = 0
      J = 0
  290 J = J+1
      IF (J.GT.MEV1) GO TO 390
C
C     WRITE(14,300) J,V2(J)
  300 FORMAT('EIGENVALUE T2-HAT =', I6,2E22.14)
C
      TEMP = EVMAG(J)
      EVAL = V2(J)
      EVALR = TEMP + SPUTOL
      EVALC = TEMP - SPUTOL
      DELMIN = 2.D0*CDABS(VS(MEV))
      IMIN = 0
C     BACKWARD SEARCH
      I = J + 1
  310 I = I - 1
      IF(I.LT.1) GO TO 320
      IF(I.GT.NDIS) I = NDIS
C
```

```
      TEMP = CDABS(VS(I))
      IF (TEMP.LT.EVALC) GO TO 320
      IF(MP(I).EQ.0) GO TO 310
      DGAP = CDABS(VS(I) - EVAL)
      IF (DGAP.GE.DELMIN) GO TO 310
      DELMIN = DGAP
      IMIN = I
C
      GO TO 310
C     FORWARD SEARCH
  320 I = J
  330 I = I + 1
      IF(I.GT.NDIS) GO TO 340
C
      TEMP = CDABS(VS(I))
      IF (TEMP.GT.EVALR) GO TO 340
      IF(MP(I).EQ.0) GO TO 330
      DGAP = CDABS(VS(I) - EVAL)
      IF (DGAP.GE.DELMIN) GO TO 330
      DELMIN = DGAP
      IMIN = I
C
      GO TO 330
C
  340 CONTINUE
      IF(IMIN.EQ.0) GO TO 370
C     WRITE(14,350) IMIN, MP(IMIN),VS(IMIN),DELMIN,J
  350 FORMAT(/I6,' TH EVALUE, MP =',I3,' EVALUE =',2E22.13/
     1' MINDEL = ',E14.3,' OCCURS FOR',I6,' TH T2-HAT EVALUE')
      IF(DELMIN.GT.SPUTOL) GO TO 290
      IF(MP(IMIN).GT.1)  GO TO 290
      MP(IMIN) = 0
C     WRITE(14,360)
  360 FORMAT(' ABOVE T-EIGENVALUE IS SPURIOUS')
      GO TO 290
  370 CONTINUE
      GO TO 290
  390 CONTINUE
C     END OF SPURIOUS TESTS
C
      DO 400 J = 1,NDIS
  400 EVMAG(J) = CDABS(VS(J))
C
      RETURN
C-----END OF COMPEV-----------------------------------------------------
  410 STOP
      END
C-----CMTQL1 (EIGENVALUES OF COMPLEX SYMMETRIC TRIDIAGONAL)-------------
C
      SUBROUTINE CMTQL1(N,D,E,IERR)
C
C-----------------------------------------------------------------------
      INTEGER I,J,L,M,N,II,MML,IERR
      COMPLEX*16 D(1),E(1),B,C,F,G,P,R,S,W,CZERO,CONE
      COMPLEX*16 CDSQRT,DCMPLX
```

```
      DOUBLE PRECISION MACHEP,EPS,TEMP,T0,T1,ZERO,HALF,ONE,TWO
      DOUBLE PRECISION CDABS,DSQRT
C-----------------------------------------------------------------------
      DATA MACHEP/Z3410000000000000/
      EPS = 100.D0*MACHEP
C-----------------------------------------------------------------------
      ZERO = 0.0D0
      HALF = 0.5D0
      ONE  = 1.0D0
      TWO  = 2.0D0
      CZERO = DCMPLX(ZERO,ZERO)
      CONE  = DCMPLX(ONE,ZERO)
      IERR = 0
      IF (N.EQ.1) GO TO 160
C
      DO 10 I = 2,N
   10 E(I-1) = E(I)
      E(N) = CZERO
C
      DO 140 L = 1,N
      J = 0
C
C     DETERMINE FIRST NEGLIGIBLE SUBDIAGONAL ELEMENT IF ANY
   20 DO 30 M = L,N
      IF (M.EQ.N) GO TO 40
      TEMP = CDABS(D(M)) + CDABS(D(M+1))
      IF (CDABS(E(M)).LE.TEMP*MACHEP) GO TO 40
   30 CONTINUE
C
   40 P = D(L)
C
      IF (M.EQ.L) GO TO 100
      IF (J.EQ.100) GO TO 150
      J = J+1
C
C     FORM SHIFT AS EIGENVALUE OF (L,L+1) 2X2 CLOSEST TO D(L)
      G = (D(L+1) - P)*HALF
      T0 = CDABS(G)
      T1 = CDABS(E(L))
      IF (T0.GT.T1) GO TO 50
      W = G/E(L)
      R = CDSQRT(CONE + W**2)
      T0 = CDABS(W + R)
      T1 = CDABS(W - R)
      TEMP = ONE
      IF (T1.GT.T0) TEMP = -ONE
      G = D(M) - P + E(L)/(W + TEMP*R)
      GO TO 60
   50 CONTINUE
      W = E(L)/G
      R = CDSQRT(CONE + W**2)
      T0 = CDABS(CONE + R)
      T1 = CDABS(CONE - R)
      TEMP = ONE
      IF (T1.GT.T0) TEMP = -ONE
```

```
      G = D(M) - P + W*E(L)/(CONE + TEMP*R)
   60 CONTINUE
C
C     G IS SHIFTED D(M)
C     SPECIFY PARAMETERS FOR I = M-1 CASE, I = M-1,M-2,...,L
C
      S = CONE
      C = -CONE
      P = CZERO
      MML = M - L
C
      DO 90 II = 1,MML
      I = M - II
C
C     FOR I<M-1 F=T(I+2,I), B=NEW E(I), AIM OF (I,I+1) TRANSFORMATION
C     IS TO ZERO OUT F
C
      F =  S*E(I)
      B = -C*E(I)
      T0 = CDABS(G)
      T1 = CDABS(F)
      IF (T1.GT.T0) GO TO 70
C     |G| >= |F|
      W = F/G
      R = CDSQRT(CONE + W**2)
      E(I+1) = G*R
      C = CONE/R
      S = W*C
      GO TO 80
C     |F| > |G|
   70 CONTINUE
      W = G/F
      R = CDSQRT(CONE + W**2)
      E(I+1) = F*R
      S = CONE/R
      C = W*S
   80 CONTINUE
      TEMP = CDABS(W)**2 + ONE
      T0 = DSQRT(TEMP)
      T1 = CDABS(R)
      IERR = -L
      IF (T1.LE.EPS*T0) GO TO 160
      IERR = 0
C
C     C**2 + S**2 = CONE, -Q(I,I) = Q(I+1,I+1) = C, Q(I,I+1) = S
C     Q = Q-TRANSPOSE = Q-INVERSE  RR = CDSQRT(G**2 +F**2)
C     G = D(I+1) AFTER PREVIOUS TRANSFORMATION THEN G = NEW E(I)
C     NEW D(I) = D(I) - S*RR,   NEW D(I+1) = D(I+1) + S*RR
C     NEW E(I) = E(I) - C*RR,   NEW E(I+1) = RR,   P = S*RR
C
      G = D(I+1) - P
      R = (D(I) - G)*S + TWO*C*B
      P = S*R
      D(I+1) = G + P
      G = B - C*R
```

```
   90 CONTINUE                                                          CSL09240
C     END OF I LOOP                                                     CSL09250
C                                                                       CSL09260
C     UPDATE PARAMETERS FOR I = L CASE                                  CSL09270
      D(L) = D(L) - P                                                   CSL09280
      E(L) = G                                                          CSL09290
      E(M) = CZERO                                                      CSL09300
      GO TO 20                                                          CSL09310
C                                                                       CSL09320
C     ORDER EIGENVALUES   P = D(L)                                      CSL09330
  100 IF (L.EQ.1) GO TO 120                                             CSL09340
      DO 110 II = 2,L                                                   CSL09350
      I = L+2-II                                                        CSL09360
      IF (CDABS(P).GE.CDABS(D(I-1))) GO TO 130                          CSL09370
      D(I) = D(I-1)                                                     CSL09380
  110 CONTINUE                                                          CSL09390
C                                                                       CSL09400
  120 I = 1                                                             CSL09410
C                                                                       CSL09420
  130 D(I) = P                                                          CSL09430
C                                                                       CSL09440
  140 CONTINUE                                                          CSL09450
      GO TO 160                                                         CSL09460
C                                                                       CSL09470
  150 IERR = L                                                          CSL09480
C-----END OF CMTQL1-----------------------------------------------------CSL09490
  160 RETURN                                                            CSL09500
      END                                                               CSL09510
C                                                                       CSL09520
C-----COMGAP------------------------------------------------------------CSL09530
C                                                                       CSL09540
      SUBROUTINE COMGAP(VC,VA,GG,MP,IND,M,IGAP,ITAG)                    CSL09550
C                                                                       CSL09560
C-----------------------------------------------------------------------CSL09570
      COMPLEX*16 VC(1),Z                                                CSL09580
      DOUBLE PRECISION  VA(1),T0,T1,TU,TK                               CSL09590
      REAL  GG(1),GTEMP                                                 CSL09600
      INTEGER  MP(1),IND(1)                                             CSL09610
      REAL ABS                                                          CSL09620
      DOUBLE PRECISION CDABS                                            CSL09630
C-----------------------------------------------------------------------CSL09640
C     IF IGAP = 0 WE DO NOT ORDER EIGENVALUES BY INCREASING GAP SIZE    CSL09650
C     AND WE DO NOT WRITE GAP OUTPUT TO FILE 12                         CSL09660
C                                                                       CSL09670
C     VA(K) = |VC(K)|  VA(K) <= VA(K+1)                                 CSL09680
C     GG(K) = MIN |VC(K)-VC(J)|  J .NE. K.                              CSL09690
C-----------------------------------------------------------------------CSL09700
      TU = VA(M) + VA(M)                                                CSL09710
      K = 0                                                             CSL09720
   10 K = K+1                                                           CSL09730
      IF (K.GT.M) GO TO 60                                              CSL09740
      INDEX = 0                                                         CSL09750
      T1 = TU                                                           CSL09760
      TK = VA(K)                                                        CSL09770
      Z = VC(K)                                                         CSL09780
```

```
      J = K
C     BACKWARDS
   20 J = J-1
      IF (J.LT.1) GO TO 30
      T0 = TK - VA(J)
      IF (T0.GT.T1) GO TO 30
      T0 = CDABS(Z - VC(J))
      IF (T1.LE.T0) GO TO 20
      T1 = T0
      INDEX = J
      GO TO 20
C     FORWARDS
   30 J = K
   40 J = J+1
      IF (J.GT.M) GO TO 50
      T0 = VA(J) - TK
      IF (T0.GT.T1) GO TO 50
      T0 = CDABS(Z - VC(J))
      IF (T1.LE.T0) GO TO 40
      T1 = T0
      INDEX = J
      GO TO 40
   50 IND(K) = INDEX
      GG(K) = T1
      IF(ITAG.EQ.0) GO TO 10
      IF(MP(INDEX).EQ.0)  GG(K) = -GG(K)
      GO TO 10
C
   60 CONTINUE
      IF (IGAP.EQ.0) GO TO 140
C
C     WRITE(12,70)
   70 FORMAT(' MINGAPS FOR GOOD T-EIGENVALUES'/
     1 1X,'EVNUM',1X,'NEIGH',15X,'R(EV)',15X,'I(EV)',4X,'MINGAP')
C     WRITE(12,80) (K,IND(K),VC(K),GG(K), K = 1,M)
   80 FORMAT(2I6,2E20.12,E10.3)
C
C     ORDER VC G BY INCREASING MINGAP SIZE
      DO 90 J = 1,M
      IND(J) = J
   90 CONTINUE
C
      DO 110 K = 2,M
      KM1 = K-1
      DO 100 L = 1,KM1
      KK = K-L
      KP1 = KK+1
      IF (ABS(GG(KP1)).GE.ABS(GG(KK))) GO TO 110
      Z = VC(KK)
      VC(KK) = VC(KP1)
      VC(KP1) = Z
      GTEMP = GG(KK)
      GG(KK) = GG(KP1)
      GG(KP1) = GTEMP
      ITEMP = IND(KK)
```

```
      IND(KK) = IND(KP1)
      IND(KP1) = ITEMP
  100 CONTINUE
  110 CONTINUE
C
C     WRITE(12,120)
  120 FORMAT(' T-EIGENVALUES ORDERED BY INCREASING MINGAP'/
     1 1X,'GAPNUM',1X,'EVNUM',15X,'R(EV)',15X,'I(EV)',4X,'MINGAP')
C
C     WRITE(12,130) (K,IND(K),VC(K),GG(K), K = 1,M)
  130 FORMAT(I7,I6,2E20.12,E10.3)
C
  140 CONTINUE
C-----END OF COMGAP----------------------------------------------------
      RETURN
      END
C
C-----START OF INVERM FOR TRIDIAGONAL COMPLEX SYMMETRIC MATRICES--------
C
      SUBROUTINE INVERM(ALPHA,BETA,V1,V2,X1,ERROR,ERRORV,EPS,GR,GC,
     1INTERC,MEV,IT,IWRITE)
C
C----------------------------------------------------------------------
      COMPLEX*16  ALPHA(1),BETA(1),V1(1),V2(1)
      COMPLEX*16  X1,U,Z,TEMP,RATIO,BETAM,ZEROC
      DOUBLE PRECISION  SUM,XU,NORM,TSUM,GSUM
      DOUBLE PRECISION  EPS,EPS3,EPS4,ERROR,ERRORV,ZERO,ONE
      DOUBLE PRECISION GR(1),GC(1)
      INTEGER INTERC(1)
      DOUBLE PRECISION  DABS, DSQRT, DFLOAT, CDABS
C     COMPLEX*16 DCMPLX
C----------------------------------------------------------------------
C
C     COMPUTES T-EIGENVECTORS FOR ISOLATED GOOD T-EIGENVALUES X1
C     USING INVERSE ITERATION ON T(1,MEV(X1)) SOLVING EQUATION
C     (T - X1*I)V2 = RIGHT-HAND SIDE (RANDOMLY-GENERATED) .
C     PROGRAM REFACTORS T- X1*I ON EACH ITERATION OF INVERSE ITERATION.
C     TYPICALLY ONLY ONE ITERATION IS NEEDED PER T-EIGENVALUE X1.
C
C     IF IWRITE = 1 THEN THERE ARE EXTENDED WRITES TO FILE 6 (TERMINAL)
C
C     ON ENTRY G CONTAINS A REAL*4 RANDOM VECTOR WHICH WAS GENERATED
C     IN MAIN PROGRAM.
C
C     ON ENTRY AND EXIT
C     MEV = ORDER OF T
C     ALPHA, BETA CONTAIN THE DIAGONAL AND OFFDIAGONAL ENTRIES OF T.
C     EPS = 2. * MACHINE EPSILON
C
C     IN PROGRAM:
C     ITER = MAXIMUM NUMBER STEPS ALLOWED FOR INVERSE ITERATION
C     ITER = IT ON ENTRY.
C     V1,V2 = WORK SPACES USED IN THE FACTORIZATION OF T(1,MEV).
C     V1 AND V2 MUST BE OF DIMENSION AT LEAST MEV.
C
```

```
C     ON EXIT                                                           CSL10890
C     V2 = THE UNIT EIGENVECTOR OF T(1,MEV) CORRESPONDING TO X1.        CSL10900
C     ERROR = |V2(MEV)| = ERROR ESTIMATE FOR CORRESPONDING              CSL10910
C             RITZ VECTOR FOR X1.                                       CSL10920
C                                                                       CSL10930
C     ERRORV = || T*V2 - X1*V2 || = ERROR ESTIMATE ON T-EIGENVECTOR.    CSL10940
C     IF IT.GT.ITER THEN ERRORV = -ERRORV                               CSL10950
C     IT = NUMBER OF ITERATIONS ACTUALLY REQUIRED                       CSL10960
C-----------------------------------------------------------------------CSL10970
C     INITIALIZATION AND PARAMETER SPECIFICATION                        CSL10980
      ONE  = 1.0D0                                                      CSL10990
      ZERO = 0.0D0                                                      CSL11000
      ZEROC = DCMPLX(ZERO,ZERO)                                         CSL11010
      ITER = IT                                                         CSL11020
      MP1 = MEV+1                                                       CSL11030
      MM1 = MEV-1                                                       CSL11040
      BETAM = BETA(MP1)                                                 CSL11050
      BETA(MP1) = ZEROC                                                 CSL11060
C                                                                       CSL11070
C     CALCULATE SCALE AND TOLERANCES                                    CSL11080
      TSUM = CDABS(ALPHA(1))                                            CSL11090
      DO 10 I = 2,MEV                                                   CSL11100
   10 TSUM = TSUM + CDABS(ALPHA(I)) + CDABS(BETA(I))                    CSL11110
C                                                                       CSL11120
      EPS3 = EPS*TSUM                                                   CSL11130
      EPS4 = DFLOAT(MEV)*EPS3                                           CSL11140
C                                                                       CSL11150
C     GENERATE SCALED RANDOM RIGHT-HAND SIDE                            CSL11160
      GSUM = ZERO                                                       CSL11170
      DO 20 I = 1,MEV                                                   CSL11180
   20 GSUM = GSUM + DABS(GR(I)) + DABS(GC(I))                           CSL11190
      GSUM = EPS4/GSUM                                                  CSL11200
C                                                                       CSL11210
C     INITIALIZE RIGHT HAND SIDE FOR INVERSE ITERATION                  CSL11220
      DO 30 I = 1,MEV                                                   CSL11230
      INTERC(I) = 0                                                     CSL11240
   30 V2(I) = GSUM*DCMPLX(GR(I),GC(I))                                  CSL11250
      IT = 1                                                            CSL11260
C                                                                       CSL11270
C     CALCULATE UNIT EIGENVECTOR OF T(1,MEV) FOR ISOLATED GOOD          CSL11280
C     T-EIGENVALUE X1.                                                  CSL11290
C                                                                       CSL11300
C     TRIANGULAR FACTORIZATION WITH NEAREST NEIGHBOR PIVOT              CSL11310
C     STRATEGY. INTERCHANGES ARE LABELLED BY SETTING INTERC(I)=0        CSL11320
C                                                                       CSL11330
   40 CONTINUE                                                          CSL11340
      U = ALPHA(1)-X1                                                   CSL11350
      Z = BETA(2)                                                       CSL11360
C                                                                       CSL11370
      DO 60 I=2,MEV                                                     CSL11380
      IF (CDABS(BETA(I)).GT.CDABS(U)) GO TO 50                          CSL11390
C     NO PIVOT INTERCHANGE                                              CSL11400
      V1(I-1) = Z/U                                                     CSL11410
      V2(I-1) = V2(I-1)/U                                               CSL11420
      V2(I) = V2(I)-BETA(I)*V2(I-1)                                     CSL11430
```

```
      RATIO = BETA(I)/U
      U = ALPHA(I)-X1-Z*RATIO
      Z = BETA(I+1)
      GO TO 60
C     PIVOT INTERCHANGE
   50 CONTINUE
      RATIO = U/BETA(I)
      INTERC(I) = 1
      V1(I-1) = ALPHA(I)-X1
      U = Z-RATIO*V1(I-1)
      Z = -RATIO*BETA(I+1)
      TEMP = V2(I-1)
      V2(I-1) = V2(I)
      V2(I) = TEMP-RATIO*V2(I)
   60 CONTINUE
C
      IF (CDABS(U).EQ.ZERO) U= DCMPLX(EPS3,EPS3)
C
C     SMALLNESS TEST AND DEFAULT VALUE FOR LAST COMPONENT
C     PIVOT(I-1) = |BETA(I)| FOR INTERCHANGE CASE
C     (I-1,I+1) ELEMENT IN RIGHT FACTOR = BETA(I+1)
C     END OF FACTORIZATION AND FORWARD SUBSTITUTION
C
C     BACK SUBSTITUTION
      V2(MEV) = V2(MEV)/U
      DO 80 II = 1,MM1
      I = MEV-II
      IF (INTERC(I+1).EQ.1) GO TO 70
C     NO PIVOT INTERCHANGE
      V2(I) = V2(I)-V1(I)*V2(I+1)
      GO TO 80
C     PIVOT INTERCHANGE
   70 V2(I) = (V2(I)-V1(I)*V2(I+1)-BETA(I+2)*V2(I+2))/BETA(I+1)
   80 CONTINUE
C
C
C     TESTS FOR CONVERGENCE OF INVERSE ITERATION
C     IF SUM |V2| COMPS. LE. 1 AND IT. LE. ITER DO ANOTHER INVIT STEP
C
      NORM = CDABS(V2(MEV))
      DO 90 II = 1,MM1
      I = MEV-II
   90 NORM = NORM+CDABS(V2(I))
C
C     IS DESIRED GROWTH IN VECTOR ACHIEVED ?
C     IF NOT, DO ANOTHER INVERSE ITERATION STEP UNLESS NUMBER ALLOWED IS
C     EXCEEDED.
      IF (NORM.GE.ONE) GO TO 110
C
      IT=IT+1
      IF (IT.GT.ITER) GO TO 110
C
      XU = EPS4/NORM
      DO 100 I=1,MEV
      INTERC(I) = 0
```

```
  100 V2(I) = V2(I)*XU
C
      GO TO 40
C
C     NORMALIZE COMPUTED T-EIGENVECTOR : V2 = V2/||V2||
C
  110 CONTINUE
C
C-----------------------------------------------------------------------
      CALL CINPRD(V2,V2,SUM,MEV)
C-----------------------------------------------------------------------
C
      SUM = ONE/DSQRT(SUM)
      DO 120 II = 1,MEV
  120 V2(II) = SUM*V2(II)
C
C     SAVE ERROR ESTIMATE FOR LATER OUTPUT
      ERROR = CDABS(V2(MEV))
C
C     GENERATE ERRORV = ||T*V2 - X1*V2||.
C     LOOP IS BOTTOM UP BECAUSE LAST COMPONENTS MAY BE VERY SMALL
      V1(MEV) = ALPHA(MEV)*V2(MEV)+BETA(MEV)*V2(MEV-1)-X1*V2(MEV)
      DO 130 J = 2,MM1
      JM = MP1 - J
      V1(JM) = ALPHA(JM)*V2(JM) + BETA(JM)*V2(JM-1) + BETA(JM+1)*V2(JM+1
     1) - X1*V2(JM)
  130 CONTINUE
C
      V1(1) = ALPHA(1)*V2(1) + BETA(2)*V2(2) - X1*V2(1)
C
C-----------------------------------------------------------------------
      CALL CINPRD(V1,V1,ERRORV,MEV)
C-----------------------------------------------------------------------
C
      ERRORV = DSQRT(ERRORV)
      IF (IT.GT.ITER) ERRORV = -ERRORV
      IF (IWRITE.EQ.0) GO TO 150
C
C     FILE 6 (TERMINAL) OUTPUT OF ERROR ESTIMATES.
      WRITE(6,140) MEV,X1,ERROR,ERRORV
  140 FORMAT(1X,'TSIZE',10X,'RE(GOODEV)',10X,'IM(GOODEV)',11X,'U(M)',
     1 9X,'TERROR'/I6,2E20.12,2E15.5)
C
C     RESTORE BETA(MEV+1) = BETAM
  150 CONTINUE
      BETA(MP1) = BETAM
C-----END OF INVERM-----------------------------------------------------
      RETURN
      END
C
C-----START OF INNER PRODUCT ROUTINE------------------------------------
C
C     COMPUTES EUCLIDEAN INNER PRODUCT OF 2 COMPLEX VECTORS
C     SUMC = (V2-TRANSPOSE)*V1
C
```

```
      SUBROUTINE INPRDC(V2,V1,SUMC,N)                                   CSL12540
C                                                                       CSL12550
C-----------------------------------------------------------------------CSL12560
      DOUBLE PRECISION  ZERO                                            CSL12570
      COMPLEX*16 V2(1),V1(1),SUMC                                       CSL12580
C-----------------------------------------------------------------------CSL12590
C                                                                       CSL12600
      ZERO = 0.D0                                                       CSL12610
      SUMC = DCMPLX(ZERO,ZERO)                                          CSL12620
      DO 10 J=1,N                                                       CSL12630
   10 SUMC = SUMC + V2(J)*V1(J)                                         CSL12640
C                                                                       CSL12650
      RETURN                                                            CSL12660
C-----END OF EUCLIDEAN INNER PRODUCT SUBROUTINE-------------------------CSL12670
      END                                                               CSL12680
C                                                                       CSL12690
C-----START OF HERMITIAN INNER PRODUCT ROUTINE--------------------------CSL12700
C     COMPLEX INNER PRODUCT                                             CSL12710
C                                                                       CSL12720
      SUBROUTINE CINPRD(V2,V1,SUM,N)                                    CSL12730
C-----------------------------------------------------------------------CSL12740
      DOUBLE PRECISION  ZERO,SUM                                        CSL12750
      COMPLEX*16 V2(1),V1(1),SUMC                                       CSL12760
C-----------------------------------------------------------------------CSL12770
C     COMPUTES THE INNER PRODUCT OF THE CONJUGATE OF V2 WITH V1.        CSL12780
      ZERO = 0.D0                                                       CSL12790
      SUMC = DCMPLX(ZERO,ZERO)                                          CSL12800
      DO 10 J=1,N                                                       CSL12810
   10 SUMC = SUMC + DCONJG(V2(J))*V1(J)                                 CSL12820
      SUM = DREAL(SUMC)                                                 CSL12830
C                                                                       CSL12840
      RETURN                                                            CSL12850
C-----END OF COMPLEX INNER PRODUCT SUBROUTINE---------------------------CSL12860
      END                                                               CSL12870
```

7.7 CSLEVAL: CSLEVEC: File Definitions, Sample Input Files

Below is a listing of the input/output files which are accessed by the complex symmetric Lanczos eigenvalue program, CSLEVAL. Included also is a sample of the input file which CSLEVAL requires on file 5. The parameters in this file are supplied in free format.File 8 contains the data for the nxn complex symmetric matrix A.

CSLEVAL computes eigenvalues of diagonalizable complex symmetric matrices.

```
Sample Specifications of Input/Output Files for CSLEVAL
------------------------------------------------------------------
 CSLEVAL EXEC LANCZOS EIGENVALUE CALCULATION COMPLEX SYMMETRIC CASE
FI 06 TERM
FILEDEF  1 DISK &1        NHISTORY  A (RECFM F LRECL 80 BLOCK 80
FILEDEF  2 DISK &1        HISTORY   A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1        GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1        ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK CSLEVAL   INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1        INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF 10 DISK &1        T-T2EVAL  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1        DISTINCT  A (RECFM F LRECL 80 BLOCK 80
LOAD   CSLEVAL   CSLESUB   CSLEMULT
------------------------------------------------------------------
```

```
Sample Input File for CSLEVAL
--------------------------------------------------------------------
 CSLEVAL INPUT LANCZOS EIGENVALUE COMPUTATION, NO REORTHOGONALIZATION
 OF A NONDEFECTIVE COMPLEX SYMMETRIC MATRIX.
LINE 1     N    KMAX    NMEVS     MATNO
         528     792        1       528
LINE 2   SVSEED     RHSEED      MXINIT
       49302312    5731029           5
LINE 3   ISTART     ISTOP
              0         1
LINE 4  IHIS  IDIST  SAVTEV  IWRITE (SAVE HIST.,DISTINCT EV,TEV,WRITE
           1      0       1       1
LINE 5   RELTOL (RELATIVE TOLERANCE IN 'COMBINING' GOODEV)
    .0000000001
LINE 6   MB(1)   MB(2)   MB(3)   MB(4)    (ORDERS OF T(1,MEV) )
           528
C     NOTE THAT WHEN READING IN PREVIOUSLY COMPUTED EIGENVALUES
C     THE VALUE OF MB(1) MUST BE EQUAL TO THE SIZE AT WHICH
C     THOSE EIGENVALUES WERE COMPUTED AND KMAX MUST BE LISTED AS
C     LARGER THAN MB(1).
--------------------------------------------------------------------
```

Below is a listing of the input/output files which are accessed by the complex symmetric Lanczos eigenvector program, CSLEVEC. Included also is a sample of the input file which CSLEVEC requires on file 5. The parameters in this file are supplied in free format.

File 8 contains the data for the nxn complex symmetric matrix A. CSLEVEC computes eigenvectors for each of a user-specified subset of the eigenvalues computed by the companion program CSLEVAL.

```
Sample Specifications of the Input/Output Files for CSLEVEC
----------------------------------------------------------------
 CSLEVEC EXEC LANCZOS EIGENVECTOR PROGRAM COMPLEX SYMMETRIC CASE
FI 06 TERM
FILEDEF  2 DISK &1       HISTORY   A (RECFM F LRECL 80 BLOCK 80
FILEDEF  3 DISK &1       GOODEV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  4 DISK &1       ERRINV    A (RECFM F LRECL 80 BLOCK 80
FILEDEF  5 DISK CSLEVEC  INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1       INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  9 DISK &1       ERREST    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 10 DISK &1       BOUNDS    A (RECFM F LRECL 80 BLOCK 80
FILEDEF 11 DISK &1       TEIGVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 12 DISK &1       RITZVECS  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 13 DISK &1       PAIGE     A (RECFM F LRECL 80 BLOCK 80
LOAD  CSLEVEC  CSLESUB  CSLEMULT
----------------------------------------------------------------
```

```
Sample Input File for CSLEVEC
-----------------------------------------------------------------------
 CSLEVEC EIGENVECTORS COMPLEX SYMMETRIC CASE NO REORTHOGONALIZATION
LINE 1  MDIMTV    MDIMRV  MBETA (MAX.DIMENSIONS,TVEC,RITVEC AND BETA
         10000     10000   2000
LINE 2       RELTOL
        .0000000001
LINE 3  MBOUND   NTVCON SVTVEC IREAD (FLAGS
             0        1      0     1
LINE 4  TVSTOP   LVCONT ERCONT  IWRITE (FLAGS
             0        1      1       1
LINE 5    RHSEED  (RANDOM GENERATOR SEED FOR STARTING VECTOR IN INVERM)
        45329517
LINE 6  MATNO      N
          100    100
-----------------------------------------------------------------------
```

Chapter 8

Real Symmetric Matrices, Block Lanczos Code

8.1 Introduction

The FORTRAN codes in this chapter address the question of using an iterative 'block' Lanczos procedure to compute a 'few' extreme eigenvalues and a basis for the corresponding invariant subspace of a given real symmetric matrix A. An eigenvalue is extreme if it is one of the algebraically-smallest or the algebraically-largest eigenvalues.

For a given real symmetric matrix A, these codes compute the q algebraically-largest eigenvalues, $\lambda_i, 1 \leq i \leq q$, of A and corresponding orthonormal real vectors $X_q \equiv (x_1, \ldots, x_q)$ such that

$$AX_q = X_q A_q, \quad A_q \equiv X_q^T A X_q. \tag{8.1.1}$$

Typically, $A_q = \Lambda_q$, a diagonal matrix whose nonzero entries are the eigenvalues λ_i . The number q is small and specified by the user.

Real symmetric matrices are discussed in detail in Stewart [24]. See Section 2.1 for a brief summary of the properties of real symmetric matrices which we use. The Lanczos procedure included in this chapter is not a true block Lanczos procedure. It is a hybrid Lanczos algorithm which combines ideas from the iterative block Lanczos procedures such as the one in Cullum and Donath [4, 3] and from the single-vector Lanczos procedure given in Chapter 2.

Several differences between the single-vector Lanczos codes in Chapters 2 through Chapter 7 and the iterative 'block' Lanczos codes should be stated explicitly. The single-vector Lanczos codes do not have the capability of directly computing the A-multiplicities of the computed eigenvalues. The 'block' procedures however, will determine the true A-multiplicity of a given computed eigenvalue and compute a complete invariant subspace for such an eigenvalue, as long as the number of Lanczos vectors in the first block is large enough. In order to determine A-multiplicities the single-vector codes have to do additional computation. In some cases these multiplicities and a basis for the required eigenspace can be determined without too much additional computation. This is true for example, whenever the desired eigenvalues replicate readily during the single-vector Lanczos computations.

The single-vector Lanczos procedures in Chapters 2 through Chapter 7 function in two stages. First the eigenvalues of the matrix being considered are computed, and then a separate program is used to compute the corresponding desired eigenvectors. The iterative 'block' Lanczos codes obtain approximations to the eigenvalues and to the eigenvectors simultaneously. Both types of codes are restartable from pre-existing computations. However, restarting has a different meaning for the two different types of codes. In the single-vector codes, restarting means computing a larger Lanczos T-matrix, starting from a pre-existing smaller one. The eigenvalue and eigenvector computations are then repeated on the larger T-matrix. In the iterative block procedures, restarting means using the current approximations to the eigenvectors (or more correctly to a basis for the desired eigenspace), to initiate another iteration of the 'block' Lanczos procedure.

The single-vector Lanczos procedures in Chapters 2 through 7 are iterative only in the sense that one may consider several Lanczos T-matrices of different sizes before achieving the desired convergence. However, the 'block' procedure presented here is genuinely iterative. On each iteration a block version of the Lanczos recursion is used to generate a sequence of blocks of Lanczos vectors, simultaneously generating a 'small' real symmetric Lanczos T-matrix. The eigenvalues and eigenvectors of this small Lanczos matrix are computed and mapped into approximating eigenvectors for the given matrix using the Lanczos vectors. These approximate eigenvectors then become the starting block of Lanczos vectors for the next iteration of the block Lanczos procedure. This 'block' procedure is described in detail in Section 7.5 of Chapter 7 in Volume 1.

As we said earlier, the 'block' procedure included here is a hybrid of the single-vector and of the basic iterative block Lanczos procedures. This procedure is based upon a modification of the following basic block version of the Lanczos recursion

$$Q_{j+1}B_{j+1} = AQ_j - Q_jA_j - Q_{j-1}B_j^T \equiv P_j \tag{8.1.2}$$

for $j = 1, 2, \ldots, s$ where the coefficient matrices A_j and B_{j+1} are block analogs of the scalar coefficients in the single vector Lanczos recursion. In the standard block procedure,

$$A_j \equiv Q_j^T(AQ_j - Q_{j-1}B_j^T) \tag{8.1.3}$$

and each B_{j+1} is obtained by the Gram-Schmidt orthogonalization of the columns of P_j and $s \ll n$, the order of the given A-matrix. Our single-vector Lanczos procedures do not use any reorthogonalization at any point in the computations. However, in our block procedures we require near-orthogonality of the Q-blocks. This orthogonality is maintained by incorporating reorthogonalization of the blocks generated within a given iteration, with respect to certain vectors in the first Lanczos block.

The sequence of 'blocks' generated on each iteration of this hybrid procedure has the property that the first Q-block, Q_1 , contains at least as many vectors as the user is trying to compute. However, the second and succeeding blocks contain exactly one vector. The corresponding Lanczos T-matrices are not block tridiagonal. Each has a border of blocks occupying the first q rows and columns and is tridiagonal below these rows and columns.

The convergence of these procedures is monitored by the subroutine DIAGOM. Convergence requires reasonable gaps between the eigenvalues requested and the eigenvalues not being approximated by the block procedure. Typically, it is the ratio of these gaps to the spread, and the distribution of the A-eigenvalues over the A-spread which controls the rate of convergence. In particular, an iterative block Lanczos procedure may have difficulty with a matrix with evenly-distributed eigenvalues. Heuristics are incorporated which allow the number of vectors used in the first Lanczos block to vary. If the convergence stagnates the procedure will terminate to allow the user to intervene and reset the program parameters if desired.

BLEVAL, the main 'block' program for these real symmetric eigenelement computations, calls the subroutine LANCZS which on each iteration then calls the subroutine LANCI1 to generate a sequence of Q-blocks for that iteration. Subroutine LANCZS then calls the subroutine DIAGOM to diagonalize the

Lanczos T-matrix generated on that iteration and to compute the updated approximations to the desired eigenspace. Convergence is checked and if it has not occurred, another iteration of the block Lanczos procedure is carried out.

In this 'block' procedure there is no identification or 'spurious' test for the eigenvalues of the Lanczos T-matrix. Since near-orthogonality of the Lanczos blocks is maintained, the q algebraically-largest eigenvalues of the T-matrices are approximations to the q algebraically-largest eigenvalues of the A-matrix being used in the recursions. This statement however, is not true for the other eigenvalues of these T-matrices because the orthogonality maintained is only with respect to the eigenspace which goes with the first q eigenvalues. The accuracy of the computed eigenvalues and eigenvectors is estimated on each iteration as part of the process of computing the second block of Lanczos vectors.

All computations are in double precision real arithmetic. The user must supply a subroutine USPEC which defines and initializes the A-matrix and a subroutine BMATV which computes Ax for any specified vector x. The small T-matrix eigenelement computations use two subroutines from the EISPACK Library [23, 8], TRED2 and IMTQL2. If the q algebraically-smallest eigenvalues are required, then the user must supply the programs with a subroutine which computes $-Ax$ rather that Ax. The user should refer to Chapter 7 in Volume 1 for more details on iterative block Lanczos procedures.

8.2 Documentation for the Codes in Chapters 8 and 9

```
C-----BLEVALHD----------------------------------------------------------BLE00010
C  Authors:  Jane Cullum* and Bill Donath**                             BLE00020
C          **IBM Research, T.J. Watson Research Center                  BLE00030
C          **Yorktown Heights, N.Y. 10598                               BLE00040
C          * Los Alamos National Laboratory                             BLE00050
C          * Los Alamos, New Mexico 87544                               BLE00060
C            E-mail:  cullumj@lanl.gov                                  BLE00065
C                                                                       BLE00070
C  These codes are copyrighted by the authors.  These codes             BLE00080
C  and modifications of them or portions of them are NOT to be          BLE00090
C  incorporated into any commercial codes or used for any other         BLE00100
C  commercial purposes such as consulting for other companies,          BLE00110
C  without legal agreements with the authors of these Codes.            BLE00120
C  If these Codes or portions of them are used in other scientific or   BLE00130
C  engineering research works the names of the authors of these codes   BLE00140
C  and appropriate references to their written work are to be          BLE00150
C  incorporated in the derivative works.                                BLE00160
C                                                                       BLE00170
C  This header is not to be removed from these codes.                   BLE00180
C                                                                       BLE00190
C                                                                       BLE00200
C     DOCUMENTATION BLOCK LANCZOS EIGENVALUE/EIGENVECTOR PROGRAMS       BLE00210
C     (1) REAL SYMMETRIC MATRICES                                       BLE00220
C     (2) FACTORED INVERSES OF REAL SYMMETRIC MATRICES                  BLE00230
C                                                                       BLE00240
C-----------------------------------------------------------------------BLE00250
C         REFERENCE: Cullum and Willoughby, Chapter 7,                  BLE00260
C         Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsBLE00270
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   BLE00280
C         Applied Mathematics, 2002. SIAM Publications,                 BLE00290
C         Philadelphia, PA. USA                                         BLE00290
C                                                                       BLE00300
C                                                                       BLE00310
C                                                                       BLE00320
C-----------------------------------------------------------------------BLE00330
C                                                                       BLE00340
C     REAL SYMMETRIC MATRICES:                                          BLE00350
C                                                                       BLE00360
C     GIVEN A REAL SYMMETRIC MATRIX A THE FILES BLEVAL, BLSUB AND       BLE00370
C     BLMULT CAN BE USED TO COMPUTE A FEW EXTREME EIGENVALUES           BLE00380
C     OF A, THAT IS THE ALGEBRAICALLY-LARGEST OR THE ALGEBRAICALLY-     BLE00390
C     SMALLEST EIGENVALUES, AND A BASIS FOR THE CORRESPONDING           BLE00400
C     EIGENSPACE.                                                       BLE00410
C                                                                       BLE00420
C     FACTORED INVERSES OF REAL SYMMETRIC MATRICES:                     BLE00430
C                                                                       BLE00440
C     GIVEN A REAL SYMMETRIC MATRIX A, THE BLOCK PROCEDURE              BLE00450
C     CAN BE APPLIED TO AN ASSOCIATED B-MATRIX WHICH IS A               BLE00460
C     SCALED, SHIFTED AND PERMUTED VERSION OF A.  THAT IS,              BLE00470
C     B = SO*P*A*P' + SHIFT*I WHERE THE SCALE SO AND THE SHIFT          BLE00480
C     ARE CHOSEN BY THE USER TO PLACE THE DESIRED EIGENVALUES           BLE00490
C     AT THE EXTREME OF THE SPECTRUM OF B-INVERSE, AND THE              BLE00500
```

```
C     PERMUTATION P IS CHOSEN SO THAT THE SPARSITY OF THE A-MATRIX      BLE00510
C     IS PRESERVED IN THE SPARSITY OF THE FACTORIZATION OF B.           BLE00520
C     THE INVERSE BLOCK PROCEDURE REQUIRES A SUBROUTINE BLSOLV          BLE00530
C     THAT FOR A GIVEN VECTOR U, COMPUTES THE VECTOR V SUCH THAT        BLE00540
C     B*V = U, USING THE FACTORIZATION OF B.  THE SAMPLE BLSOLV         BLE00550
C     SUBROUTINE PROVIDED ASSUMES THAT THE B-MATRIX IS POSITIVE         BLE00560
C     DEFINITE AND THAT THE CHOLESKY FACTORS OF B ARE SUPPLIED          BLE00570
C     ON FILE 7.  HOWEVER, THE USER MAY REPLACE THIS SUBROUTINE         BLE00580
C     BY ONE THAT COMPUTES A MORE GENERAL FACTORIZATION                 BLE00590
C     L*D*(L-TRANSPOSE) FOR AN INDEFINITE SYMMETRIC MATRIX.             BLE00600
C     THE BLOCK PROCEDURE USED IN THIS FASHION USES THE FILES           BLE00610
C     BLIEVAL, BLIMULT AND BLSUB.                                       BLE00620
C                                                                       BLE00630
C                                                                       BLE00640
C     ALGORITHM:                                                        BLE00650
C     THESE PROGRAMS USE A BLOCK FORM OF LANCZOS TRIDIAGONALIZATION     BLE00660
C     WITH REORTHOGONALIZATION ONLY WITH RESPECT TO VECTORS             BLE00670
C     IN THE 1ST Q-BLOCK.  THE PROCEDURES ARE ITERATIVE, GENERATING     BLE00680
C     ON EACH ITERATION A SMALL SYMMETRIC LANCZOS MATRIX, T.            BLE00690
C     THE EIGENVALUES AND EIGENVECTORS OF THE SMALL MATRIX ARE          BLE00700
C     COMPUTED USING SUBROUTINES FROM THE EISPACK LIBRARY.              BLE00710
C     THE RELEVANT SUBSET OF THE T-EIGENVECTORS IS THEN MAPPED          BLE00720
C     INTO THE LARGE N-SPACE CORRESPONDING TO THE MATRIX BEING          BLE00730
C     USED BY THE LANCZS SUBROUTINE, CONVERGENCE IS CHECKED,            BLE00740
C     AND IF CONVERGENCE OF THE DESIRED EIGENVALUES AND                 BLE00750
C     EIGENVECTORS HAS NOT YET OCCURRED, THEN THE CURRENT               BLE00760
C     APPROXIMATIONS TO THE DESIRED EIGENSPACE ARE USED AS              BLE00770
C     STARTING VECTORS FOR THE NEXT ITERATION OF BLOCK LANCZOS.         BLE00780
C                                                                       BLE00790
C     USERS SHOULD NOTE THAT TYPICALLY IN THE BLOCK LANCZOS             BLE00800
C     PROCEDURES, IT IS THE RATIO OF THE GAPS TO THE SPREAD THAT        BLE00810
C     CONTROLS THE CONVERGENCE ALONG WITH HOW THE EIGENVALUES           BLE00820
C     ARE DISTRIBUTED OVER THAT SPREAD.   THE BIGGER THE GAPS           BLE00830
C     BETWEEN THE ONES BEING COMPUTED AND THE CLOSEST ONES NOT          BLE00840
C     BEING COMPUTED AND THE WEAKER THE SPREAD, THE FASTER THE          BLE00850
C     CONVERGENCE WILL BE.  WITHOUT DECENT GAPS THIS PROCEDURE          BLE00860
C     WILL NOT CONVERGE.  THE PROGRAMS CONTAIN CHECKS ON                BLE00870
C     THE ACTUAL RATE OF CONVERGENCE WHICH WILL CAUSE THE               BLE00880
C     PROCEDURE TO TERMINATE IF CONVERGENCE IS NOT OCCURRING            BLE00890
C     SUFFICIENTLY RAPIDLY.  THE USER MAY THEN CHANGE EITHER OR         BLE00900
C     BOTH THE MAXIMUM SIZE T-MATRIX ALLOWED AND THE NUMBER             BLE00910
C     OF VECTORS IN THE FIRST Q-BLOCK AND RERUN THE PROCEDURE           BLE00920
C     WITH THE CURRENT APPROXIMATION TO THE DESIRED EIGENSPACE          BLE00930
C     AS THE STARTING BLOCK OF VECTORS.                                 BLE00940
C                                                                       BLE00950
C                                                                       BLE00960
C     THE IDEAS USED IN THESE PROGRAMS ARE DISCUSSED IN THE FOLLOWING   BLE00970
C     REFERENCES.                                                       BLE00980
C                                                                       BLE00990
C     1.  JANE CULLUM AND RALPH A. WILLOUGHBY, LANCZOS ALGORITHMS       BLE01000
C         FOR LARGE SYMMETRIC MATRICES,  PROGRESS IN                    BLE01010
C         SCIENTIFIC COMPUTING, EDITORS, G. GOLUB, H.O. KREISS,         BLE01020
C         S. ARBARBANEL, AND R. GLOWINSKI,  BIRKHAUSER BOSTON INC.,     BLE01030
C         CAMBRIDGE, MASSACHUSETTS, 1984.                               BLE01040
C                                                                       BLE01050
```

```
C     2.  JANE CULLUM AND W.E. DONATH, A BLOCK LANCZOS ALGORITHM
C         FOR COMPUTING THE Q ALGEBRAICALLY-LARGEST EIGENVALUES AND
C         A CORRESPONDING EIGENSPACE OF LARGE, SPARSE REAL SYMMETRIC
C         MATRICES, PROCEEDINGS OF THE 1974 IEEE CONFERENCE ON
C         DECISION AND CONTROL, PHOENIX, ARIZONA, PP.505-509, NOVEMBER
C         1974.
C
C     3.  JANE CULLUM,  AN ACCELERATED 'BLOCK' LANCZOS ALGORITHM
C         FOR A FEW EXTREME EIGENVALUES OF A LARGE, SPARSE REAL
C         SYMMETRIC MATRIX.  IBM REPORT 1983.  PRESENTED AT THE
C         SPARSE MATRIX CONFERENCE, FAIRFIELD GLADE, TENNESSEE,
C         OCTOBER 1982.
C
C
C-----PORTABILITY------------------------------------------------------
C
C     PROGRAMS WERE TESTED FOR PORTABILITY USING THE PFORT VERIFIER.
C     FOR DETAILS OF THE VERIFIER SEE FOR EXAMPLE, B. G. RYDER AND
C     A. D. HALL, "THE PFORT VERIFIER", COMPUTING SCIENCE TECHNICAL
C     REPORT 12, BELL LABORATORIES, MURRAY HILL, NEW JERSEY 07974,
C     (REVISED), JANUARY 1981.
C
C     EXCEPT FOR THE FOLLOWING CONSTRUCTIONS WHICH CAN BE EASILY
C     MODIFIED BY THE USER TO MATCH THE PARTICULAR COMPUTER BEING
C     USED, THE PROGRAM STATEMENTS ARE PORTABLE.
C
C    NONPORTABLE STATEMENTS.
C
C    IN BLEVAL, BLIEVAL (MAIN PROGRAMS)
C         1.  DATA/MACHEP STATEMENT
C         2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)
C         3.  FORMAT(20A4) USED FOR THE EXPLANATORY HEADER ARRAY, EXPLAN
C         4.  FORMAT(4Z20) WHICH CAN BE USED TO WRITE LARGE VECTOR
C             FILES
C         5.  THE COMMON BLOCK: LOOPS.
C    IN BLMULT, BLIMULT
C         1.  IN BMATV, BLSOLV, AND USPEC, THE ENTRIES WHICH
C             PASS THE STORAGE LOCATIONS OF THE ARRAYS DEFINING
C             THE USER-SPECIFIED MATRIX OR FACTORIZATION.
C    IN BLSUB
C         1.  ALL STATEMENTS ARE PORTABLE EXCEPT THE ENTRY TO
C             SUBROUTINE LPERM WHICH PASSES THE PERMUTATION USED
C             TO OBTAIN THE B-MATRIX FROM SUBROUTINE USPEC.
C             SUBROUTINE LPERM IS USED ONLY IN CASE (2).
C
C
C-----MATRIX SPECIFICATION---------------------------------------------
C
C     SUBROUTINE USPEC IS USED TO SPECIFY THE MATRIX WHICH THE BLOCK
C     LANCZOS PROCEDURE WILL USE.  IN CASE (1) THIS IS THE USER-
C     SPECIFIED A-MATRIX.   IN CASE (2) THE FACTORIZATION OF THE
C     ASSOCIATED B-MATRIX IS SPECIFIED.  SUBROUTINE USPEC HAS THE
C     CALLING SEQUENCE
C
C           CALL USPEC(N,MATNO,NNZ,AVER)
```

```
C
C     WHERE N IS THE ORDER OF THE USER-SUPPLIED MATRIX A,
C     MATNO IS AN <= 8 DIGIT INTEGER USED AS A MATRIX AND
C     TEST IDENTIFICATION NUMBER, NNZ IS THE AVERAGE NUMBER
C     OF NONZERO ENTRIES IN EACH COLUMN, AND AVER IS THE
C     AVERAGE SIZE OF THE NONZERO ENTRIES IN THE MATRIX USED
C     BY LANCZS.  NOTE THAT NNZ AND AVER ARE DEFINED AS DOUBLE
C     PRECISION SCALARS.  THE MAIN PROGRAMS ASSUME THAT THEY
C     ARE COMPUTED IN USPEC.  THE USPEC SUBROUTINE
C     DEFINES AND DIMENSIONS THE ARRAYS REQUIRED TO
C     SPECIFY THE MATRIX THAT WILL BE USED BY THE LANCZS
C     SUBROUTINE AND INITIALIZES THESE ARRAYS.  THE STORAGE
C     LOCATIONS OF THESE ARRAYS ARE THEN PASSED TO THE
C     SUBROUTINE BMATV IN CASE (1) AND TO THE SUBROUTINE BSOLV
C     IN CASE (2).  SAMPLE SUBROUTINES ARE INCLUDED FOR EACH
C     CASE.  CASE (1) ASSUMES THAT THE A-MATRIX IS STORED ON
C     FILE 8.  CASE (2) ASSUMES THAT THE FACTORIZATION OF THE
C     B-MATRIX IS STORED ON FILE 7.
C
C     IN CASE (1) :
C     BMATV IS THE SUBROUTINE USED BY THE LANCZS SUBROUTINE
C     THAT GENERATES THE LANCZOS T-MATRICES.  SUBROUTINE
C     BMATV HAS THE CALLING SEQUENCE
C
C            CALL BMATV(W,U)
C
C     WHERE U AND W ARE DOUBLE PRECISION VECTORS.  FOR A GIVEN
C     W, BMATV CALCULATES U = A*W  FOR THE USER-SPECIFIED MATRIX A.
C     A SAMPLE BMATV IS INCLUDED FOR AN ARBITRARY SPARSE,
C     SYMMETRIC A-MATRIX STORED IN THE SPARSE FORMAT SPECIFIED
C     IN THE CORRESPONDING SAMPLE USPEC SUBROUTINE.
C
C     IN CASE (2):
C     THE LANCZOS T-MATRICES ARE GENERATED USING SPARSE MATRIX
C     INVERSION, USING THE SUBROUTINE BLSOLV.  THE CALLING
C     SEQUENCE OF BLSOLV IS
C
C                   CALL BLSOLV(U,V)
C
C     WHERE U AND V ARE DOUBLE PRECISION VECTORS.  FOR A GIVEN V,
C     BLSOLV COMPUTES U = (B-INVERSE)*V USING A SPARSE
C     FACTORIZATION OF THE B-MATRIX ASSOCIATED WITH THE USER-
C     SPECIFIED A-MATRIX.
C
C     THE FOLLOWING SPARSE MATRIX FORMAT IS USED TO STORE THE
C     MATRICES IN THE SAMPLE PROGRAMS:
C     ICOL(K), K = 1,NZL, NUMBER OF SUBDIAGONAL NONZEROS IN COLUMN K.
C     IROW(K), K = 1,NZS, ROW INDEX OF ASD(K).
C     AD(K), K=1,N, CONTAINS THE DIAGONAL ELEMENTS OF THE A-MATRIX.
C     ASD(K), K=1,NZS  CONTAINS THE SUBDIAGONAL ELEMENTS OF A BY COLUMN.
C     NZS = NUMBER OF NONZERO ELEMENTS BELOW THE DIAGONAL OF A
C     NZL = INDEX OF LAST COLUMN WITH NONZERO SUBDIAGONAL ENTRIES
C     N = ORDER OF THE A-MATRIX.
C
C     IN CASE (1) THE A-MATRIX IS STORED IN THIS FORMAT ON FILE 8.
```

```
C     IN CASE (2), IN THE SAMPLE USPEC PROVIDED WHICH IS ONLY           BLE02160
C     FOR POSITIVE DEFINITE B-MATRICES, THE SPARSE CHOLESKY FACTOR      BLE02170
C     OF B, L, IS STORED ON FILE 7 IN THE ABOVE SPARSE FORMAT           BLE02180
C     USING ARRAYS BD AND BSD.  IN CASE (2) THE OPTIONAL AUXILIARY      BLE02190
C     PROGRAMS PERMUT AND LORDER ALSO REQUIRE THE A-MATRIX;             BLE02200
C     HOWEVER, THE BLOCK LANCZOS PROCEDURE ONLY USES THE                BLE02210
C     FACTORIZATION OF THE B-MATRIX.                                    BLE02220
C                                                                       BLE02230
C                                                                       BLE02240
C-----MACHEP------------------------------------------------------------BLE02250
C                                                                       BLE02260
C                                                                       BLE02270
C     MACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING THE RELATIVE   BLE02280
C     PRECISION OF THE FLOATING POINT ARITHMETIC USED.                  BLE02290
C     MACHEP = 2.2 * 10**-16 FOR DOUBLE PRECISION ARITHMETIC ON         BLE02300
C     IBM 370-3081.                                                     BLE02310
C                                                                       BLE02320
C     THE USER WILL HAVE TO RESET THIS PARAMETER TO                     BLE02330
C     THE CORRESPONDING VALUE FOR THE MACHINE BEING USED.  NOTE THAT    BLE02340
C     IF A MACHINE WITH A MACHINE EPSILON THAT IS MUCH LARGER THAN THE  BLE02350
C     VALUE GIVEN HERE IS BEING USED, THEN THERE COULD BE               BLE02360
C     PROBLEMS WITH THE TOLERANCES.                                     BLE02370
C                                                                       BLE02380
C                                                                       BLE02390
C-----SUBROUTINES AND FUNCTIONS USER MUST SUPPLY------------------------BLE02400
C                                                                       BLE02410
C                                                                       BLE02420
C     GENRAN,  FINPRO,  MASK,  USPEC,  AND                              BLE02430
C     CASE (1)  BMATV:  CASE (2)  BLSOLV :                              BLE02440
C                                                                       BLE02450
C     GENRAN = COMPUTES K PSEUDO-RANDOM NUMBERS AND STORES THEM IN      BLE02460
C              THE REAL ARRAY, G.  THIS SUBROUTINE IS USED TO           BLE02470
C              GENERATE STARTING VECTORS FOR THE BLOCK LANCZOS          BLE02480
C              PROCEDURE.  CALLED FROM LANCZS SUBROUTINE.               BLE02490
C              USER CAN SUPPLY STARTING VECTORS FOR THE BLOCK           BLE02500
C              PROCEDURES.  ANY ADDITIONAL VECTORS REQUIRED ARE         BLE02510
C              GENERATED RANDOMLY BY GENRAN.  VECTORS SUPPLIED MUST     BLE02520
C              BE STORED ON FILE 10.  THE NUMBER OF SUCH VECTORS TO     BLE02530
C              BE READ IN IS SPECIFIED BY THE PARAMETER KSET.  THE      BLE02540
C              EXISTING CALLING SEQUENCE IS                             BLE02550
C                                                                       BLE02560
C                      CALL GENRAN(IIX,G,K).                            BLE02570
C                                                                       BLE02580
C              WHERE IIX =INTEGER SEED, G = REAL ARRAY WHOSE DIMENSION  BLE02590
C              MUST BE >= K.  K PSEUDO-RANDOM NUMBERS ARE GENERATED     BLE02600
C              AND PLACED IN G.                                         BLE02610
C                                                                       BLE02620
C     FINPRO = DOUBLE PRECISION FUNCTION WHICH COMPUTES THE INNER       BLE02630
C              PRODUCT OF 2 DOUBLE PRECISION VECTORS OF DIMENSION N.    BLE02640
C              EXISTING CALLING SEQUENCE IS                             BLE02650
C                                                                       BLE02660
C                      CALL FINPRO(N,V,J,W,K).                          BLE02670
C                                                                       BLE02680
C              COMPUTES THE INNER PRODUCT OF DIMENSION N OF THE VECTORS BLE02690
C              V AND W.  SUCCESSIVE COMPONENTS OF V AND OF W ARE STORED BLE02700
```

```
C             AT LOCATIONS THAT ARE ,RESPECTIVELY, J AND K UNITS APART.BLE02710
C                                                                      BLE02720
C      MASK = MASKS OVERFLOW AND UNDERFLOW.  OPTIONAL.                 BLE02730
C             USER MUST SUPPLY OR COMMENT OUT CALL.                    BLE02740
C                                                                      BLE02750
C     USPEC = DIMENSIONS AND INITIALIZES ARRAYS NEEDED TO SPECIFY      BLE02760
C             MATRIX USED BY LANCZS SUBROUTINE.  SEE MATRIX            BLE02770
C             SPECIFICATION SECTION.                                   BLE02780
C                                                                      BLE02790
C     BMATV = CASE (1) ONLY: COMPUTES MATRIX-VECTOR MULTIPLY FOR       BLE02800
C             USER-SUPPLIED A-MATRIX. SEE MATRIX SPECIFICATION SECTION.  BLE02810
C                                                                      BLE02820
C     BLSOLV = CASE (2) ONLY:  FOR GIVEN VECTOR V, COMPUTES U SUCH     BLE02830
C             B*U = V, GIVEN THE SPARSE FACTORIZTION OF THE B-MATRIX.   BLE02840
C                                                                      BLE02850
C                                                                      BLE02860
C-----PARAMETER CONTROLS-----------------------------------------------BLE02870
C                                                                      BLE02880
C                                                                      BLE02890
C      PARAMETER CONTROLS ARE INTRODUCED TO CONTROL VARIOUS            BLE02900
C      ASPECTS OF THESE PROGRAMS.                                      BLE02910
C                                                                      BLE02920
C      THE FLAG EFLAG SPECIFIES THE NUMBER OF COMPUTATIONAL PHASES.    BLE02930
C                                                                      BLE02940
C      EFLAG  = (0,1)  MEANS                                           BLE02950
C                                                                      BLE02960
C               (0) PROGRAM TERMINATES AFTER COMPLETING PHASE 1        BLE02970
C                   COMPUTATIONS.                                      BLE02980
C                                                                      BLE02990
C               (1) PROGRAM COMPLETES BOTH PHASE 1 AND PHASE 2 OF      BLE03000
C                   THE COMPUTATIONS.                                  BLE03010
C                                                                      BLE03020
C      THE FLAG OFLAG CONTROLS THE ORTHOGONALITY CHECKS BETWEEN THE    BLE03030
C      JTH Q-BLOCK GENERATED AND THAT VECTOR IN THE 1ST Q-BLOCK THAT   BLE03040
C      IS GENERATING DESCENDANTS.  FOR SAFETY, OFLAG SHOULD BE 1.      BLE03050
C                                                                      BLE03060
C      OFLAG = (0,1)  MEANS                                            BLE03070
C                                                                      BLE03080
C               (0) NO ORTHOGONALITY CHECKS ARE MADE ON PHASE          BLE03090
C                   1 PORTION OF THE COMPUTATIONS.  ORTHOGONALITY      BLE03100
C                   CHECKS ARE ALWAYS MADE ON PHASE 2 PORTION.         BLE03110
C                                                                      BLE03120
C               (1) PROGRAM CHECKS ORTHOGONALITY OF GENERATED          BLE03130
C                   Q-BLOCKS W.R.T. THAT VECTOR IN THE 1ST Q-BLOCK     BLE03140
C                   THAT IS GENERATING DESCENDANTS IN BOTH PHASE       BLE03150
C                   1 AND PHASE 2 OF THE COMPUTATIONS.                 BLE03160
C                                                                      BLE03170
C      THE FLAG IWRITE DETERMINES THE AMOUNT OF OUTPUT TO FILE 6       BLE03180
C      DURING THE COMPUTATIONS                                         BLE03190
C                                                                      BLE03200
C      IWRITE = (0,1)   MEANS                                          BLE03210
C                                                                      BLE03220
C               (0) ABBREVIATED OUTPUT TO FILE 6.                      BLE03230
C                                                                      BLE03240
C               (1) ADDITIONAL COMMENTARY ON THE COMPUTATIONS IS       BLE03250
```

```
C                 PRINTED TO FILE 6.                                    BLE03260
C                                                                       BLE03270
C     THE PROGRAM ALWAYS WRITES A LIST OF THE COMPUTED EIGENVALUES      BLE03280
C     AND THE BASIS FOR THE CORRESPONDING EIGENSPACE TO FILE 15,        BLE03290
C     ALONG WITH ESTIMATES OF THE ERRORS IN THESE COMPUTED VALUES.      BLE03300
C                                                                       BLE03310
C-----INPUT/OUTPUT FILES------------------------------------------------BLE03320
C                                                                       BLE03330
C     ANY INPUT DATA OTHER THAN THE A-MATRIX, THE FACTORIZATION         BLE03340
C     OF THE B-MATRIX OR USER-SPECIFIED STARTING VECTORS SHOULD         BLE03350
C     BE STORED ON FILE 5.  SEE SAMPLE INPUT/OUTPUT FROM TYPICAL RUN.   BLE03360
C     THE READ STATEMENTS IN THE GIVEN FORTRAN PROGRAM ASSUME THAT      BLE03370
C     THE DATA STORED ON FILE 5 IS IN FREE FORMAT.  USER SHOULD NOTE    BLE03380
C     THAT 'FREE FORMAT' IS NOT CLASSIFIED AS PORTABLE BY PFORT SO THAT BLE03390
C     THE USER MAY HAVE TO MODIFY THE READ STATEMENTS FROM FILE 5 TO    BLE03400
C     CONFORM TO WHAT IS PERMISSIBLE ON THE COMPUTER BEING USED.        BLE03410
C                                                                       BLE03420
C     FILE 6 WAS USED AS THE INTERACTIVE TERMINAL OUTPUT FILE.          BLE03430
C     THIS FILE PROVIDES A RUNNING ACCOUNT OF THE PROGRESS OF THE       BLE03440
C     COMPUTATIONS.  THE AMOUNT OF INFORMATION PRINTED OUT IS           BLE03450
C     CONTROLLED BY THE PARAMETER IWRITE.                               BLE03460
C                                                                       BLE03470
C DESCRIPTION OF OTHER I/O FILES                                        BLE03480
C                                                                       BLE03490
C FILE (K)     CONTAINS:                                                BLE03500
C                                                                       BLE03510
C      (7)     INPUT FILE:                                              BLE03520
C              USED IN CASE (2).  CONTAINS THE FACTORIZATION            BLE03530
C              OF THE B-MATRIX.                                         BLE03540
C                                                                       BLE03550
C      (8)     INPUT FILE:                                              BLE03560
C              USED IN CASE (1).  CONTAINS THE ARRAYS REQUIRED          BLE03570
C              TO SPECIFY THE A-MATRIX.                                 BLE03580
C                                                                       BLE03590
C     (10)     INPUT FILE:                                              BLE03600
C              CONTAINS USER-SUPPLIED STARTING VECTORS, IF ANY.         BLE03610
C              TYPICALLY, THESE WOULD BE 1 OR MORE EIGENVECTOR          BLE03620
C              APPROXIMATIONS OBTAINED DURING AN EARLIER RUN.           BLE03630
C                                                                       BLE03640
C     (13)     OUTPUT FILE:                                             BLE03650
C              CONTAINS EXTRA EIGENVECTOR APPROXIMATIONS THAT           BLE03660
C              WOULD OTHERWISE BE LOST UPON ANY REDUCTION IN THE        BLE03670
C              SIZE OF THE 1ST Q-BLOCK.  IF AT ANY STAGE IN THE         BLE03680
C              BLOCK PROCEDURE, THE SIZE OF THE 1ST Q-BLOCK IS          BLE03690
C              REDUCED FROM KACT TO KACTN, THE Q-VECTORS FROM           BLE03700
C              K = KACTN+1,KACT ARE WRITTEN TO FILE 13 FOR POSSIBLE     BLE03710
C              USE AS STARTING VECTORS IN A LATER RUN OF THE            BLE03720
C              BLOCK LANCZOS PROCEDURE.                                 BLE03730
C                                                                       BLE03740
C     (15)     OUTPUT FILE:                                             BLE03750
C              CONTAINS COMPUTED EIGENVALUES AND CORRESPONDING          BLE03760
C              COMPUTED EIGENSPACE AVAILABLE AT THE TIME OF             BLE03770
C              TERMINATION OF THE BLOCK LANCZOS PROCEDURE.              BLE03780
C                                                                       BLE03790
C-----PARAMETERS SET BY THE BLOCK PROGRAMS------------------------------BLE03800
```

```
C
C
C   SPREC = TOLERANCE USED IN CHECKING ORTHOGONALITY BETWEEN
C           COMPUTED Q-BLOCKS AND THAT VECTOR IN THE FIRST
C           Q-BLOCK THAT IS GENERATING DESCENDANTS.  SEE COMMENTS
C           ON OFLAG.
C
C-----USER-SPECIFIED PARAMETERS ----------------------------------------
C
C
C    FOR BOTH CASES:
C
C    N, MATNO = INTEGERS. SIZE OF USER-SPECIFIED MATRIX AND MATRIX
C               IDENTIFICATION NUMBER OF 8 OR FEWER DIGITS.
C
C    MDIMQ, MDIMTM = INTEGERS.  USER-SPECIFIED DIMENSIONS OF THE
C                    Q-ARRAY AND OF THE TM-ARRAY.  MDIMQ >= N*KMAX
C                    AND MDIMTM >= MXBLK**2.
C
C    MAXIT,MAXIT2  = INTEGERS.  MAXIMUM NUMBER OF CALLS TO BMATV
C                    (CASE(1)) OR TO BLSOLV (CASE (2)) ALLOWED
C                    RESPECTIVELY, IN PHASE 1 AND IN PHASE 2.
C
C    RELTOL = DOUBLE PRECISION SCALAR.  RELATIVE TOLERANCE USED
C             TO COMPUTE CONVERGENCE CRITERION FOR PHASE 2 OF
C             THE BLOCK PROCEDURE.
C
C    SEED   = INTEGER.  SEED FOR RANDOM NUMBER GENERATOR.
C             USED IN GENERATION OF STARTING VECTORS FOR
C             THE BLOCK PROCEDURES.
C
C    KMAX  = INTEGER.  MXBLK = (KMAX - 1) IS MAXIMUM ALLOWED SIZE
C            FOR THE SMALL LANCZOS T-MATRICES.
C
C    KM    = INTEGER.  NUMBER OF EIGENVALUES AND EIGENVECTORS
C            TO BE COMPUTED.
C
C    KACT  = INTEGER.  INITIAL NUMBER OF VECTORS IN THE 1ST Q-BLOCK.
C            IF THERE IS ANY POSSIBILITY THAT THE KM-TH DESIRED
C            EIGENVALUE IS MULTIPLE, AND THE USER NEEDS TO KNOW
C            THIS, THEN THE USER SHOULD SET KACT > KM.  OTHERWISE,
C            THIS PROGRAM WILL NOT BE ABLE TO DETERMINE THAT THAT
C            EIGENVALUE IS MULTIPLE UNLESS THE (KM-1)-TH AND KM-TH
C            HAPPEN TO BE MULTIPLE.  IF IN FACT, THE KM-TH
C            EIGENVALUE IS MULTIPLE AND THE USER NEEDS A BASIS FOR
C            THE CORRESPONDING EIGENSPACE, THEN THE PROCEDURE SHOULD
C            BE RERUN WITH THE EXISTING EIGENVECTORS APPROXIMATIONS
C            AS STARTING VECTORS AND A LARGER KACT TO GUARANTEE THAT
C            A COMPLETE BASIS FOR THAT EIGENSPACE HAS BEEN OBTAINED.
C
C    KSET =  INTEGER.  NUMBER OF STARTING VECTORS SUPPLIED BY THE
C            THE USER.  THESE VECTORS SHOULD BE ON FILE 10.
C
C
C    NSTAG = INTEGER.  NUMBER OF THE ITERATION BEYOND WHICH THE
```

```
C             CHANGE IN THE KM-TH RESIDUAL OVER THE PAST 10 ITERATIONS  BLE04360
C             IS MONITORED AND USED AS A MEASURE OF THE RATE OF         BLE04370
C             CONVERGENCE OF THE BLOCK PROCEDURE.                       BLE04380
C                                                                       BLE04390
C     FRACT = DOUBLE PRECISION SCALAR.  EXPECTED OR HOPED FOR           BLE04400
C             FRACTIONAL CHANGE IN THE KM-TH RESIDUAL OVER THE PAST     BLE04410
C             BLOCK LANCZOS ITERATIONS USED TO TEST FOR STAGNATION      BLE04420
C             OF CONVERGENCE.                                           BLE04430
C                                                                       BLE04440
C     NNZ   = DOUBLE PRECISION SCALAR.  AVERAGE NUMBER OF NONZERO       BLE04450
C             ENTRIES PER ROW IN THE MATRIX USED IN THE LANCZOS         BLE04460
C             PROCEDURE.                                                BLE04470
C                                                                       BLE04480
C                                                                       BLE04490
C     AVER  = DOUBLE PRECISION SCALAR.  AVERAGE SIZE OF THE NONZERO     BLE04500
C             ENTRIES IN THE MATRIX USED IN THE LANCZOS PROCEDURE.      BLE04510
C                                                                       BLE04520
C     CASE (2) ONLY:                                                    BLE04530
C                                                                       BLE04540
C     S0, SHIFT = DOUBLE PRECISION SCALARS.  MATRIX USED BY LANCZS      BLE04550
C                 SUBROUTINE IS B = S0*P*A*P' + SHIFT*I  WHERE P        BLE04560
C                 DENOTES A PERMUTATION MATRIX SELECTED TO PRESERVE     BLE04570
C                 THE SPARSITY OF A IN THE FACTORIZATION OF B.          BLE04580
C                 S0 AND SHIFT ARE CHOSEN BY THE USER SO THAT THE       BLE04590
C                 DESIRED EIGENVALUES BECOME THE EXTREME EIGENVALUES    BLE04600
C                 OF B-INVERSE.                                         BLE04610
C                                                                       BLE04620
C                                                                       BLE04630
C-----CONVERGENCE TEST--------------------------------------------------BLE04640
C                                                                       BLE04650
C                                                                       BLE04660
C     THE CONVERGENCE TEST INCORPORATED IN THIS PROGRAM IS              BLE04670
C     BASED UPON THE FOLLOWING FACT:  GIVEN A REAL SYMMETRIC            BLE04680
C     MATRIX A, A VECTOR X OF NORM 1, AND A SCALAR EVAL                 BLE04690
C     THEN THERE EXISTS AN EIGENVALUE AEVAL OF A SUCH THAT              BLE04700
C     DABS(AEVAL - EVAL) .LE. NORM(A*X - EVAL*X).  WITHIN               BLE04710
C     EACH ITERATION OF THE BLOCK LANCZOS PROCESS THESE TYPES           BLE04720
C     OF NORMS ARE COMPUTED IN THE PROCESS OF COMPUTING THE             BLE04730
C     2ND Q-BLOCK.                                                      BLE04740
C                                                                       BLE04750
C                                                                       BLE04760
C-----ARRAYS REQUIRED---------------------------------------------------BLE04770
C                                                                       BLE04780
C                                                                       BLE04790
C     Q(J)     = DOUBLE PRECISON ARRAY.  ITS DIMENSION MUST BE AT       BLE04800
C                LEAST AS LARGE AS KMAX*N, WHERE N IS THE ORDER OF      BLE04810
C                THE GIVEN MATRIX, AND MXBLK = KMAX - 1 IS THE          BLE04820
C                MAXIMUM SIZE T-MATRIX ALLOWED ON ANY GIVEN             BLE04830
C                ITERATION.  THE COLUMNS OF Q HOLD THE LANCZOS          BLE04840
C                VECTORS GENERATED ON EACH ITERATION OF BLOCK           BLE04850
C                LANCZOS PLUS THERE MUST BE AN ADDITIONAL COLUMN        BLE04860
C                AVAILABLE FOR WORK SPACE.  THE FIRST KACT COLUMNS      BLE04870
C                OF Q CONTAIN THE CURRENT APPROXIMATING EIGENSPACE.     BLE04880
C                                                                       BLE04890
C     E(J)    = DOUBLE PRECISION ARRAY.  ITS DIMENSION MUST BE AT       BLE04900
```

```
C            LEAST MXBLK = KMAX - 1.  ON EACH ITERATION CONTAINS     BLE04910
C            THE COMPUTED EIGENVALUES OF THE LANCZOS T-MATRIX.       BLE04920
C                                                                    BLE04930
C     TM(J)   = DOUBLE PRECISION ARRAY.  ITS DIMENSION MUST BE AT    BLE04940
C            LEAST MXBLK**2 WHERE MXBLK = KMAX - 1.  CONTAINS        BLE04950
C            THE LANCZOS T-MATRIX GENERATED ON EACH ITERATION        BLE04960
C            AND THEN THE COMPUTED EIGENVECTORS OF THIS MATRIX.      BLE04970
C            EISPACK SUBROUTINES ARE USED FOR THE SMALL              BLE04980
C            EIGENELEMENT COMPUTATIONS.  EISPACK SUBROUTINE          BLE04990
C            TRED2 IS USED TO REDUCE THE GIVEN T-MATRIX TO           BLE05000
C            TRIDIAGONAL FORM.  THE EIGENELEMENT PROBLEM FOR THE     BLE05010
C            TRIDIAGONAL MATRIX IS THEN SOLVED USING THE EISPACK     BLE05020
C            SUBROUTINE IMTQL2.                                      BLE05030
C                                                                    BLE05040
C     EXPLAN(J) = REAL ARRAY.  ITS DIMENSION IS 20.  THIS ARRAY IS   BLE05050
C              USED TO ALLOW EXPLANATORY COMMENTS IN THE INPUT FILES.BLE05060
C                                                                    BLE05070
C     G(J) = REAL ARRAY.  ITS DIMENSION MUST BE >= N.  IT IS USED    BLE05080
C          FOR HOLDING THE PSEUDO-RANDOM NUMBERS USED TO GENERATE    BLE05090
C          ANY STARTING VECTORS NOT SUPPLIED BY THE USER.            BLE05100
C                                                                    BLE05110
C     RESIDL(J),  = DOUBLE PRECISION ARRAYS.  DIMENSION >= MAXIMUM   BLE05120
C     RESIDK(J),    NUMBER OF ITERATIONS ALLOWED.  MAXIMUM IS        BLE05130
C                   CURRENTLY SET TO 100. USED TO MONITOR THE        BLE05140
C                   RATE OF CONVERGENCE.                             BLE05150
C                                                                    BLE05160
C     TD(J), TOD(J), = DOUBLE PRECISION ARRAYS.  DIMENSION >= MXBLK. BLE05170
C         SM(J)        WORK SPACES.                                  BLE05180
C                                                                    BLE05190
C     DESC(J), XLFT(J), = INTEGER ARRAYS.  DIMENSION >= MXBLK.       BLE05200
C         LEFT(J)         WORK SPACES.                               BLE05210
C                                                                    BLE05220
C     DIR(2,J)  = 2-DIMENSIONAL INTEGER ARRAY.  COLUMN DIMENSION >=  BLE05230
C              MXBLK, ROW DIMENSION 2.  KEEPS TRACK OF NUMBER        BLE05240
C              OF VECTORS IN EACH QBLOCK.                            BLE05250
C                                                                    BLE05260
C     CASE (2) ONLY:                                                 BLE05270
C                                                                    BLE05280
C     IPR(J), IPT(J) = INTEGER ARRAYS.  EACH OF DIMENSION AT LEAST N.  BLE05290
C                      USED TO STORE THE REORDERING (IF ANY) OF      BLE05300
C                      THE GIVEN MATRIX.                             BLE05310
C                                                                    BLE05320
C     OTHER ARRAYS                                                   BLE05330
C                                                                    BLE05340
C     THE USER IN THE SUBROUTINE USPEC MUST SPECIFY WHATEVER ARRAYS  BLE05350
C     ARE REQUIRED TO DEFINE THE MATRIX BEING USED BY LANCZS.        BLE05360
C                                                                    BLE05370
C                                                                    BLE05380
C-----SUBROUTINES INCLUDED----------------------------------------------BLE05390
C                                                                    BLE05400
C                                                                    BLE05410
C     LANCZS = CONTAINS MAJOR LOOP FOR BLOCK LANCZOS PROCEDURES.     BLE05420
C             CALLED FROM MAIN PROGRAM, CALLS SUBROUTINE LANCI1      BLE05430
C             TO GENERATE WITHIN A GIVEN ITERATION THE Q-BLOCKS      BLE05440
C             AND CORRESPONDING LANCZOS T-MATRICES.  THEN CALLS      BLE05450
```

```
C              SUBROUTINE DIAGOM TO COMPUTE THE EIGENELEMENTS          BLE05460
C              OF THE LANCZOS T-MATRIX AND TO MAP THE RELEVANT         BLE05470
C              T-EIGENVECTORS INTO RITZ VECTORS FOR THE A-MATRIX.      BLE05480
C                                                                      BLE05490
C     LANCI1 =  ON EACH ITERATION OF BLOCK LANCZOS COMPUTES            BLE05500
C               Q-SUBBLOCKS.                                           BLE05510
C                                                                      BLE05520
C     DIAGOM = CALLS EISPACK SUBROUTINES TO COMPUTE THE                BLE05530
C              EIGENELEMENTS OF THE SMALL LANCZOS T-MATRICES           BLE05540
C              GENERATED ON EACH ITERATION OF BLOCK LANCZOS.           BLE05550
C              COMPUTES CORRESPONDING RITZ VECTORS FOR A-MATRIX.       BLE05560
C              MONITORS CONVERGENCE OF BLOCK LANCZOS PROCEDURE.        BLE05570
C                                                                      BLE05580
C     START  = GENERATES ANY REQUIRED STARTING VECTORS FOR 1ST         BLE05590
C              Q-BLOCK FOR FIRST ITERATION OF BLOCK LANCZOS.           BLE05600
C                                                                      BLE05610
C     ORTHOG = GIVEN A SET OF Q-VECTORS, Q(J), J = MA,MB,              BLE05620
C              ORTHOGONALIZES THESE VECTORS W.R.T. THE Q-VECTORS       BLE05630
C              Q(J), J = 1,MA-1.                                       BLE05640
C                                                                      BLE05650
C     LPERM  = (USED IN CASE (2) ONLY) GIVEN A MATRIX B AND A          BLE05660
C              PERMUTATION P DEFINED IN THE VECTORS IPR AND IPT,       BLE05670
C              AND A VECTOR X COMPUTE EITHER (P-TRANSPOSE)*X OR PX.    BLE05680
C                                                                      BLE05690
C     CASE (2) ONLY:                                                   BLE05700
C     FOR OPTIONAL PRELIMINARY PROCESSING:                             BLE05710
C                                                                      BLE05720
C     PERMUT  (STAND-ALONE PROGRAM):                                   BLE05730
C     USES THE NONZERO STRUCTURE OF A GIVEN MATRIX A.                  BLE05740
C     CAN BE USED TO OBTAIN A REORDERING OF A THAT WILL PRESERVE       BLE05750
C     THE SPARSENESS OF A UNDER FACTORIZATION.  PERMUT CALLS           BLE05760
C     CALLS THE SPARSPAK PACKAGE, (A. GEORGE, J. LIU, E. NG,           BLE05770
C     U. WATERLOO).  SEE THE PERMUT FORTRAN CODE FOR DETAILS.          BLE05780
C                                                                      BLE05790
C     LORDER (STAND-ALONE PROGRAM) :                                   BLE05800
C     GIVEN A MATRIX C IN SPARSE FORMAT AND A PERMUTATION P,           BLE05810
C     COMPUTES THE REORDERED MATRIX B = P*C*P' AND WRITES IT           BLE05820
C     TO FILE 9 IN SPARSE FORMAT.  SEE THE LORDER FORTRAN CODE         BLE05830
C     FOR DETAILS.                                                     BLE05840
C                                                                      BLE05850
C     LFACT (STAND-ALONE PROGRAM) :                                    BLE05860
C     GIVEN A POSITIVE DEFINITE MATRIX B IN SPARSE FORMAT,             BLE05870
C     COMPUTES THE SPARSE CHOLESKY FACTOR L OF B AND WRITES IT         BLE05880
C     TO FILE 7 IN SPARSE FORMAT.  THUS, B = L*L'.                     BLE05890
C     SEE THE LFACT FORTRAN CODE FOR DETAILS.                          BLE05900
C                                                                      BLE05910
C     LTEST (STAND-ALONE MAIN PROGRAM ) :                              BLE05920
C           (USER MUST PROVIDE 3 SUBROUTINES)                          BLE05930
C     GIVEN THE FACTORIZATION OF A SPARSE MATRIX B, COMPUTES           BLE05940
C     THE SOLUTION OF THE EQUATION B*U = B*V1 FOR A KNOWN BUT          BLE05950
C     RANDOMLY-GENERATED VECTOR V1, SOLVING WITH AND WITHOUT ITERATIVE BLE05960
C     REFINEMENT TO OBTAIN A ROUGH CHECK ON THE NUMERICAL CONDITION    BLE05970
C     OF THE B-MATRIX.  THIS PROGRAM USES 3 USER-SUPPLIED SUBROUTINES  BLE05980
C     CMATV, CMATS AND BLSOLV.  SEE THE LTEST FORTRAN CODE FOR DETAILS. BLE05990
C                                                                      BLE06000
```

```
C                                                                       BLE06010
C-----OTHER PROGRAMS PROVIDED-------------------------------------------BLE06020
C                                                                       BLE06030
C                                                                       BLE06040
C     LECOMPAC  =  TRANSLATES A REAL SYMMETRIC MATRIX PROVIDED          BLE06050
C                  IN THE FORMAT  I, J, A(I,J) INTO THE SPARSE          BLE06060
C                  MATRIX FORMAT USED IN THE SAMPLE SUBROUTINES         BLE06070
C                  PROVIDED. IT ASSUMES THAT THE MATRIX                 BLE06080
C                  ENTRIES ARE GIVEN EITHER COLUMN BY COLUMN OR         BLE06090
C                  ROW BY ROW.  THE DATA SET CREATED IS WRITTEN TO      BLE06100
C                  FILE 8.                                              BLE06110
C                                                                       BLE06120
C                                                                       BLE06130
C-----------------------------------------------------------------------BLE06140
```

8.3 BLEVAL: Main Program, Eigenvalue and Eigenvector Computations

```
C-----BLEVAL (FEW EXTREME EIGENVALUES AND EIGENVECTORS)-----------------BLE00010
C                      (REAL SYMMETRIC MATRICES)                        BLE00020
C  Authors:  Jane Cullum* and Bill Donath**                             BLE00030
C         **IBM Research, T.J. Watson Research Center                   BLE00040
C         **Yorktown Heights, N.Y. 10598                                BLE00050
C         * Los Alamos National Laboratory                              BLE00060
C         * Los Alamos, New Mexico 87544                                BLE00065
C           E-mail:  cullumj@lanl.gov                                   BLE00070
C                                                                       BLE00080
C  These codes are copyrighted by the authors.  These codes             BLE00090
C  and modifications of them or portions of them are NOT to be          BLE00100
C  incorporated into any commercial codes or used for any other         BLE00110
C  commercial purposes such as consulting for other companies,          BLE00120
C  without legal agreements with the authors of these Codes.            BLE00130
C  If these Codes or portions of them are used in other scientific or   BLE00140
C  engineering research works the names of the authors of these codes   BLE00150
C  and appropriate references to their written work are to be           BLE00160
C  incorporated in the derivative works.                                BLE00170
C                                                                       BLE00180
C  This header is not to be removed from these codes.                   BLE00190
C                                                                       BLE00200
C        REFERENCE: Cullum and Willoughby, Chapter 7,                   BLE00201
C        Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsBLE00202
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in    BLE00203
C        Applied Mathematics, 2002. SIAM Publications,                  BLE00204
C        Philadelphia, PA. USA                                          BLE00205
C                                                                       BLE00206
C                                                                       BLE00210
C     CONTAINS MAIN PROGRAM FOR COMPUTING A FEW OF THE ALGEBRAICALLY-   BLE00220
C     LARGEST EIGENVALUES AND CORRESPONDING EIGENVECTORS OF A REAL      BLE00230
C     SYMMETRIC MATRIX, USING A BLOCK FORM OF LANCZOS TRIDIAGONALIZATIONBLE00240
C     WITH LIMITED REORTHOGONALIZATION.  PROCEDURE IS ITERATIVE.        BLE00250
C     PROCEDURE CAN BE USED TO COMPUTE THE ALGEBRAICALLY-SMALLEST       BLE00260
C     EIGENVALUES BY THE USER SUPPLYING -A*X RATHER THAN A*X, IN        BLE00270
C     WHICH CASE IT COMPUTES THE CORRESPONDING ALGEBRAICALLY-LARGEST    BLE00280
C     EIGENVALUES OF -A.  IN THIS CASE THE SIGNS OF THE COMPUTED        BLE00290
C     EIGENVALUES ARE CHANGED PRIOR TO WRITING TO FILE 15 SO THAT       BLE00300
C     ON EXIT, FILE 15 CONTAINS THE ALGEBRAICALLY-SMALLEST EIGENVALUES  BLE00310
C     OF A ALONG WITH THE CORRESPONDING EIGENVECTORS.                   BLE00320
C                                                                       BLE00330
C     ITERATIVE 'BLOCK' LANCZOS PROCEDURE FOR WHICH ON EVERY            BLE00340
C     ITERATION, THE 2ND AND SUCCEEDING BLOCKS CONTAIN ONLY ONE         BLE00350
C     VECTOR WHICH IS SELECTED ON THE BASIS OF ITS EXPECTED INFLUENCE   BLE00360
C     ON THE CONVERGENCE.   Q-BLOCKS GENERATED ON A GIVEN ITERATION     BLE00370
C     ARE REORTHOGONALIZED ONLY W.R.T. THOSE VECTORS IN THE FIRST       BLE00380
C     Q-BLOCK THAT ARE NOT GENERATING DESCENDANTS ON THAT               BLE00390
C     ITERATION.                                                        BLE00400
C                                                                       BLE00410
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE CONSTRUCTIONS:BLE00420
C     1.  DATA MACHEP DEFINITION                                        BLE00430
```

```
C     2.  FORMAT (20A4) USED FOR READING EXPLANATORY COMMENTS.          BLE00440
C     3.  FREE FORMAT (5,*), USED FOR PARAMETER INPUT FROM FILE 5.      BLE00450
C     4.  COMMON/LOOPS/ AS CONSTRUCTED IS NOT PORTABLE                  BLE00460
C                                                                       BLE00470
C-----------------------------------------------------------------------BLE00480
      DOUBLE PRECISION Q(44000),E(50),TM(2500),TOD(50),TD(50),EPSM,NNZ  BLE00490
      DOUBLE PRECISION SM(100),ERRMAX,SPREC,MACHEP,AVER,RELTOL,ERRMAN   BLE00500
      DOUBLE PRECISION  EVAL, RESIDL(100), RESIDK(100), RESID, FRACT    BLE00510
      REAL EXPLAN(20),G(2000)                                           BLE00520
      INTEGER   DIR(2,100),DESC(100),LEFT(100),XLFT(100)                BLE00530
      INTEGER SEED,OFLAG,EFLAG                                          BLE00540
      COMMON/LOOPS/MAXIT,ITER                                           BLE00550
      COMMON /RANDOM/SEED                                               BLE00560
      COMMON/FLAGS/EFLAG,OFLAG                                          BLE00570
      DOUBLE PRECISION  DABS, DFLOAT                                    BLE00580
C-----------------------------------------------------------------------BLE00590
      EXTERNAL BMATV                                                    BLE00600
      DATA MACHEP/Z3410000000000000/                                    BLE00610
C-----------------------------------------------------------------------BLE00620
C                                                                       BLE00630
C     ARRAYS MUST DIMENSIONED AS FOLLOWS:                               BLE00640
C                                                                       BLE00650
C     1.  Q:   >=  KMAX*N                                               BLE00660
C     2.  G:   >= N                                                     BLE00670
C     3.  E:   >=  MXBLK                                                BLE00680
C     4.  TM:   >=  MXBLK**2                                            BLE00690
C     5.  TOD, TD, SM, DESC, LEFT, XLFT:   >= MXBLK                     BLE00700
C     6.  DIR:  ROW DIMENSION = 2;   COLUMN DIMENSION >= MXBLK          BLE00710
C     7.  RESIDL, RESIDK:  >= MAXIMUM NUMBER OF ITERATIONS ALLOWED.     BLE00720
C         PROGRAM CURRENTLY TERMINATES IF MORE THAN 100 ITERATIONS      BLE00730
C         ARE REQUESTED.  USED TO MONITOR CONVERGENCE.                  BLE00740
C     8.  EXPLAN:  DIMENSION = 20.                                      BLE00750
C                                                                       BLE00760
C-----------------------------------------------------------------------BLE00770
C     OUTPUT HEADER                                                     BLE00780
      WRITE(6,10)                                                       BLE00790
   10 FORMAT(/' BLOCK LANCZOS PROCEDURE FOR REAL SYMMETRIC MATRICES'    BLE00800
     1 /' 2ND AND SUCCEEDING BLOCKS CONTAIN ONLY ONE VECTOR'//)         BLE00810
C                                                                       BLE00820
C     SET PROGRAM PARAMETERS                                            BLE00830
      EPSM = 2.D0*MACHEP                                                BLE00840
      SPREC = 1.D-5                                                     BLE00850
      MPMIN = -1000                                                     BLE00860
C                                                                       BLE00870
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 5 (FREE FORMAT)    BLE00880
C                                                                       BLE00890
C     SELECT THE AMOUNT OF INTERMEDIATE OUTPUT DESIRED (IWRITE =0,1).   BLE00900
C     IWRITE = 1 INCREASES THE AMOUNT OF INTERMEDIATE OUTPUT WRITTEN    BLE00910
C     TO FILE 6 ON EACH ITERATION OF THE BLOCK LANCZOS PROCEDURE.       BLE00920
      READ(5,20) EXPLAN                                                 BLE00930
   20 FORMAT(20A4)                                                      BLE00940
      READ(5,*)  IWRITE                                                 BLE00950
C                                                                       BLE00960
C     READ ORDER (N) OF MATRIX AND MATRIX IDENTIFICATION NUMBER (MATNO) BLE00970
      READ(5,20) EXPLAN                                                 BLE00980
```

```
      READ(5,*) N,MATNO                                                 BLE00990
C                                                                       BLE01000
C     READ USER-SPECIFIED DIMENSIONS OF Q-ARRAY (MDIMQ) AND OF THE      BLE01010
C     TM-ARRAY (MDIMTM).  READ MAXIMUM NUMBER (MAXIT) OF MATRIX-VECTOR  BLE01020
C     MULTIPLIES ALLOWED IN PHASE 1.                                    BLE01030
      READ(5,20) EXPLAN                                                 BLE01040
      READ(5,*) MDIMQ, MDIMTM, MAXIT                                    BLE01050
C                                                                       BLE01060
C     READ FLAGS:   EFLAG = (0,1).  EFLAG = 0, MEANS PROGRAM STOPS      BLE01070
C     AFTER COMPLETING PHASE 1 PORTION OF BLOCK LANCZOS PROCEDURE.      BLE01080
C     EFLAG = 1, MEANS PROGRAM COMPLETES BOTH PHASES BEFORE             BLE01090
C     TERMINATING.                                                      BLE01100
C     OFLAG = (0,1).  OFLAG = 0, MEANS THAT IN PHASE 1 PORTION          BLE01110
C     OF THE COMPUTATION, THE PROGRAM DOES NO ORTHOGONALITY CHECKS      BLE01120
C     ON THE Q-BLOCKS GENERATED.  OFLAG = 1 MEANS THAT IN THE           BLE01130
C     PHASE 1 PORTION AND IN THE PHASE 2 PORTIONS OF THE COMPUTATIONS   BLE01140
C     THE PROGRAM CHECKS THE ORTHOGONALITY OF THE Q-BLOCKS GENERATED    BLE01150
C     W.R.T. THAT VECTOR IN THE FIRST BLOCK THAT IS GENERATING          BLE01160
C     DESCENDANTS.  NOTE THAT IN PHASE 2, THE PROGRAM ALWAYS MAKES      BLE01170
C     THIS CHECK OF ORTHOGONALITY REGARDLESS OF THE VALUE OF OFLAG.     BLE01180
C     FOR SAFETY, OFLAG SHOULD ALWAYS BE SET TO 1, ALTHOUGH IN MANY     BLE01190
C     PROBLEMS THIS IS NOT NECESSARY.                                   BLE01200
      READ(5,20) EXPLAN                                                 BLE01210
      READ(5,*) EFLAG,OFLAG                                             BLE01220
C                                                                       BLE01230
C     READ SEED USED BY SUBROUTINE GENRAN TO OBTAIN THOSE STARTING      BLE01240
C     VECTORS WHICH ARE GENERATED RANDOMLY.                             BLE01250
      READ(5,20) EXPLAN                                                 BLE01260
      READ(5,*) SEED                                                    BLE01270
C                                                                       BLE01280
C     SPECIFY MAXIMUM T-SIZE ALLOWED (KMAX-1); INITIAL SIZE OF          BLE01290
C     STARTING BLOCK (KACT);  NUMBER OF STARTING VECTORS SUPPLIED (KSET)BLE01300
C     SEE BLOCK LANCZOS HEADER FOR COMMENTS ON THE SIZE OF KACT.        BLE01310
      READ(5,20) EXPLAN                                                 BLE01320
      READ(5,*) KMAX,KACT,KSET                                          BLE01330
C                                                                       BLE01340
C     SPECIFY NUMBER OF EXTREME EIGENVALUES AND EIGENVECTORS TO BE      BLE01350
C     COMPUTED (KM).  USER CAN SPECIFY THAT THE ALGEBRAICALLY-          BLE01360
C     SMALLEST EIGENVALUES ARE BEING COMPUTED BY SETTING KM < 0.        BLE01370
C     PROGRAM THEN ASSUMES THAT THE MATRIX-VECTOR MULTIPLY              BLE01380
C     SUBROUTINE WHICH THE USER HAS PROVIDED IS COMPUTING -A*X          BLE01390
C     INSTEAD OF A*X AND INTERNALLY IT COMPUTES THE |KM|                BLE01400
C     ALGEBRAICALLY-LARGEST EIGENVALUES OF -A.                          BLE01410
      READ(5,20) EXPLAN                                                 BLE01420
      READ(5,*) KM                                                      BLE01430
      IF(KM.EQ.0) GO TO 490                                             BLE01440
      KML = IABS(KM)                                                    BLE01450
C                                                                       BLE01460
C     STAGNATION OF CONVERGENCE OF THE KM-TH EIGENVALUE WILL BE         BLE01470
C     TESTED AFTER NSTAG ITERATIONS.  CONVERGENCE WILL BE SAID TO       BLE01480
C     HAVE STAGNATED IF THE RATIO OF THE SQUARE OF THE CURRENT KM-TH    BLE01490
C     RESIDUAL TO THE SQUARE OF THE CORRESPONDING RESIDUAL OBTAINED     BLE01500
C     10 ITERATIONS EARLIER IS GREATER THAN FRACT.  NSTAG SHOULD BE     BLE01510
C     >= 25.  IN THE TESTS FRACT WAS SET TO .01.                        BLE01520
      READ(5,20) EXPLAN                                                 BLE01530
```

```
      READ(5,*) NSTAG, FRACT                                            BLE01540
C                                                                       BLE01550
C     READ IN THE RELATIVE TOLERANCE (RELTOL) USED TO DETERMINE A       BLE01560
C     CONVERGENCE CRITERION FOR PHASE 2, AND THE MAXIMUM NUMBER (MAXIT2)BLE01570
C     OF MATRIX-VECTOR MULTIPLIES ALLOWED IN PHASE 2.                   BLE01580
      READ(5,20) EXPLAN                                                 BLE01590
      IF(EFLAG.EQ.1)  READ(5,*)  RELTOL, MAXIT2                         BLE01600
C                                                                       BLE01610
C     CONSISTENCY CHECKS                                                BLE01620
C     PROCEDURE REQUIRES ENOUGH ROOM IN Q-ARRAY FOR AT LEAST 2          BLE01630
C     BLOCKS OF SIZE KACT PLUS A WORKING VECTOR OF LENGTH N.            BLE01640
      MXBLK = KMAX -1                                                   BLE01650
      MXBLK2 = MXBLK*MXBLK                                              BLE01660
      IF(MDIMTM.LT.MXBLK2) GO TO 470                                    BLE01670
      NKMAX = N*KMAX                                                    BLE01680
      IF(MDIMQ.LT.NKMAX)   GO TO 510                                    BLE01690
      IF(KML.GT.KACT) GO TO 370                                         BLE01700
      IF(MXBLK.GT.N)  GO TO 390                                         BLE01710
      IF(2*KACT.GT.MXBLK)  GO TO 450                                    BLE01720
C                                                                       BLE01730
C-----------------------------------------------------------------------BLE01740
C     DEFINE AND INITIALIZE THE ARRAYS FOR THE USER-SPECIFIED           BLE01750
C     A-MATRIX AND PASS THE STORAGE LOCATIONS OF THESE ARRAYS AND       BLE01760
C     OF ANY OTHER PARAMTERS NEEDED TO DEFINE THE MATRIX TO THE         BLE01770
C     MATRIX-VECTOR MULTIPLY SUBROUTINE BMATV.                          BLE01780
C                                                                       BLE01790
      CALL USPEC(N,MATNO,NNZ,AVER)                                      BLE01800
C                                                                       BLE01810
C-----------------------------------------------------------------------BLE01820
C     MASK OVERFLOW AND UNDERFLOW                                       BLE01830
      CALL MASK                                                         BLE01840
C                                                                       BLE01850
C-----------------------------------------------------------------------BLE01860
C     ARE THERE STARTING VECTORS TO READ IN FROM FILE 10 (KSET.NE.0) ?  BLE01870
      IF(KSET.EQ.0) GO TO 70                                            BLE01880
C                                                                       BLE01890
      READ(10,30) NOLD,KACT                                             BLE01900
   30 FORMAT(I6,I4)                                                     BLE01910
      IF(NOLD.NE.N.OR.KSET.GT.KACT) GO TO 410                           BLE01920
      DO 50 J=1,KSET                                                    BLE01930
      READ(10,20) EXPLAN                                                BLE01940
      READ(10,40) EVAL,RESID                                            BLE01950
   40 FORMAT(E20.12,E13.4)                                              BLE01960
      READ(10,20) EXPLAN                                                BLE01970
      LINT= (J-1)*N + 1                                                 BLE01980
      LFIN = J*N                                                        BLE01990
   50 READ(10,60) (Q(JL), JL = LINT,LFIN)                               BLE02000
   60 FORMAT(4E20.12)                                                   BLE02010
C                                                                       BLE02020
   70 CONTINUE                                                          BLE02030
C                                                                       BLE02040
C     WRITE TO A SUMMARY OF THE PARAMETERS FOR THIS RUN TO FILE 6       BLE02050
C                                                                       BLE02060
      MXBLK = KMAX - 1                                                  BLE02070
      WRITE(6,80)  N, NNZ, AVER, MATNO                                  BLE02080
```

```
   80 FORMAT(/6X,'ORDER OF MATRIX ',5X,'AVERAGE NONZEROES PER ROW'/
     1I15,E26.4/4X,'AVERAGE SIZE OF NONZERO ENTRIES',5X,'MATRIX ID'/
     1E25.4,I21/)
C
      WRITE(6,90) MDIMQ, MDIMTM
   90 FORMAT(/18X,'USER-SPECIFIED'/2X,'MAX. DIMENSION Q-ARRAY',4X,'MAX.
     1DIMENSION TM-ARRAY'/I16,I26/)
C
      WRITE(6,100)  OFLAG, EFLAG
  100 FORMAT(/4X,'OFLAG',4X,'EFLAG'/I8,I9/)
C
      IF(EFLAG.EQ.1)  WRITE(6,110) MAXIT,RELTOL,MAXIT2
  110 FORMAT(/4X,' MAXIT ',8X,'  RELTOL  ',6X,' MAXIT2 '/I10,E20.6,I12/)
      IF(EFLAG.EQ.0)  WRITE(6,120) MAXIT
  120 FORMAT(/4X,' MAXIT '/I10/)
C
      WRITE(6,130)  SEED
  130 FORMAT(/' SEED FOR RANDOM NUMBER GENERATOR'/I24/)
C
      IF(KM.GT.0)  WRITE(6,140) KML
  140 FORMAT(/' COMPUTE THE',I3,' ALGEBRAICALLY-LARGEST EIGENVALUES AND
     1CORRESPONDING VECTORS'/)
      IF(KM.LT.0)  WRITE(6,150) KML
  150 FORMAT(/' COMPUTE THE',I3,' ALGEBRAICALLY-SMALLEST EIGENVALUES AND
     1 CORRESPONDING VECTORS'/' PROGRAM ASSUMES THAT USER IS PROVIDING -
     1A*X INSTEAD OF A*X'/' AND COMPUTES THE ALGEBRAICALLY-LARGEST EIGEN
     1VALUES OF -A.'/' HOWEVER ON EXIT, FILE 15 CONTAINS THE ALGEBRAICAL
     1LY-SMALLEST EIGENVALUES OF'/' THE ORIGINAL A-MATRIX AND CORRESPOND
     1ING EIGENVECTORS.'/)
      IF(KM.LT.0) KM = - KM
C
C     COMPUTE PHASE 1 CONVERGENCE TOLERANCE
      IF(AVER.GE.1.)
     1ERRMAX = 2.D0*DFLOAT(N+1000)*NNZ*AVER*MACHEP
      IF(AVER.LT.1.)
     1ERRMAX = 2.D0*DFLOAT(N+1000)*NNZ*AVER**2*MACHEP
C
      WRITE(6,160) KACT,MXBLK,KSET
  160 FORMAT(/' ON INITIAL ITERATIONS, THE FIRST BLOCK CONTAINS ',I3,' V
     1ECTORS'/' HOWEVER THE SIZE OF THE FIRST BLOCK MAY CHANGE AS THE IT
     1ERATIONS PROCEED'/' THE MAXIMUM SIZE T-MATRIX THAT CAN BE GENERATE
     1D IS ',I4/' THE USER SUPPLIED ',I3,' STARTING VECTORS'/)
C
      WRITE(6,170)
  170 FORMAT(/' ITERATIVE PROCEDURE'/' PROCEDURE MONITORS THE SIZES OF T
     1HE NORM(GRADIENTS)**2 ON EACH'/' ITERATION.  CONVERGENCE IS SAID
     1TO HAVE OCCURRED WHEN ALL'/' RELEVANT (NORMS)**2 ARE LESS THAN ERR
     1MAX',E10.3/' TYPICALLY, PHASE 1 ERRMAX YIELDS SOMEWHAT LESS THAN'/
     1' SINGLE PRECISION ACCURACY.  PHASE 2 REFINES THE VECTORS OBTAINED
     1'/' ON PHASE 1, ACCORDING TO THE ACCURACY SPECIFIED BY THE USER'/)
C
      WRITE(6,180) ERRMAX
  180 FORMAT(//' PHASE 1 CONVERGENCE CRITERION, ERRMAX '/E22.3/)
C
C-----------------------------------------------------------------------
```

```
C     PASS STORAGE LOCATIONS OF VARIOUS ARRAYS TO LANCZS AND LANCI1
C     SUBROUTINES
C
      CALL LANZP(DIR,DESC,SM,TM,TOD,TD,G,XLFT,LEFT,SPREC)
      CALL LANCP1(DIR,DESC,TM,SM,XLFT,LEFT)
C
C-----------------------------------------------------------------------
C
C     ENTER PHASE 1 OF BLOCK LANCZOS PROCEDURE.  BLOCK PROCEDURE
C     HAS 2 POSSIBLE PHASES.  USER SPECIFIES PHASE 1 ONLY OR PHASE 1
C     AND PHASE 2 BY SETTING EFLAG = 0 OR 1, RESPECTIVELY. PHASE 1
C     COMPUTES VECTORS THAT MAY BE SOMEWHAT LESS ACCURATE THAN SINGLE
C     PRECISION.  PHASE 2 TAKES THE VECTORS OBTAINED IN PHASE 1
C     AND ATTEMPTS TO REFINE THEM.  THE USER SPECIFIES THE DEGREE
C     OF REFINEMENT DESIRED BY SETTING THE VALUES OF RELTOL AND MAXIT2.
C     BOTH PHASES SHOULD BE USED.
      IPHASE = 1
      NITER = 0
  190 ITER = 0
      RESIDL(1) = FRACT
      RESIDL(2) = NSTAG
C
C-----------------------------------------------------------------------
C     CALL INITIATES THE BLOCK LANCZOS PROCEDURE.
C     ON RETURN EIGENVALUE APPROXIMATIONS ARE IN E(I), I=1,KACT
C     IN ALGEBRAICALLY DECREASING ORDER.   EIGENVECTOR APPROXIMATIONS
C     ARE IN FIRST N*KACT LOCATIONS IN THE Q-ARRAY.
C
      CALL LANCZS(BMATV,KML,KSET,KACT,MXBLK,N,Q,E,RESIDL,RESIDK,ERRMAX,
     1 IPHASE,NITER,IWRITE)
C
C-----------------------------------------------------------------------
C
      IF(IPHASE.EQ.MPMIN) WRITE(15,200) N,KACT
  200 FORMAT(2I10,' PHASE 2 TERMINATED '/' PROGRAM INDICATES ACCURACY SP
     1ECIFIED BY USER IS NOT ACHIEVABLE'/)
C
      ITERA = IABS(ITER)
      IF(IWRITE.NE.MPMIN.AND.ITER.GT.0)  WRITE(6,210) IPHASE,ITERA
  210 FORMAT(/1X,'PHASE COMPLETED',5X,' NUMBER MATRIX-VECTOR MULTIPLIES
     1USED'/I10,I30)
C
      IF(IWRITE.EQ.MPMIN.OR.ITER.LT.0)  WRITE(6,220) IPHASE,ITERA
  220 FORMAT(/1X,'PHASE TERMINATED',5X,' NUMBER MATRIX-VECTOR MULTIPLIES
     1 USED'/I10,I30)
C
      IF(ITER.GT.0.AND.IWRITE.NE.MPMIN)  GO TO 250
C
      IF(ITER.LT.0)  WRITE(6,230)
  230 FORMAT(//' SMALL EIGENVALUE SUBROUTINE DEFAULTED'/' BLOCK LANCZOS
     1 PROCEDURE STOPS AFTER SAVING CURRENT EIGENVECTOR APPROXIMATIONS'/
     1/)
C
      WRITE(15,240)
      WRITE(6,240)
```

```
  240 FORMAT(//' BLOCK LANCZOS PROCEDURE TERMINATES WITHOUT CONVERGENCE BLE03190
     1'/' USER SHOULD EXAMINE OUTPUT TO DETERMINE REASONS FOR TERMINATIOBLE03200
     1N'//)                                                             BLE03210
C                                                                       BLE03220
C     WRITE EIGENVALUE AND EIGENVECTOR APPROXIMATIONS CONTAINED IN      BLE03230
C     THE FIRST Q-BLOCK TO FILE 15                                      BLE03240
C                                                                       BLE03250
  250 IF(IPHASE.EQ.1)  WRITE(15,260) N,KACT,SEED                        BLE03260
  260 FORMAT(I6,I4,I12,' PHASE 1, ORDER A-MATRIX, SIZE OF Q(1), SEED')  BLE03270
      IF(IPHASE.EQ.2)  WRITE(15,270) N,KACT,SEED                        BLE03280
  270 FORMAT(I6,I4,I12,' PHASE 2, ORDER A-MATRIX, SIZE OF Q(1), SEED')  BLE03290
C                                                                       BLE03300
      JJ=KACT                                                           BLE03310
      LINT = -N+1                                                       BLE03320
      LFIN = 0                                                          BLE03330
      DO 290 J=1,KACT                                                   BLE03340
      LINT = LINT + N                                                   BLE03350
      LFIN = LFIN + N                                                   BLE03360
      JJ=JJ+1                                                           BLE03370
C                                                                       BLE03380
C     NOTE THAT RESIDUAL PRINTED OUT CORRESPONDS TO VALUE OBTAINED      BLE03390
C     PRIOR TO FINAL PROJECTION Q(1)-TRANSPOSE*AQ(1) DONE BEFORE        BLE03400
C     TERMINATION                                                       BLE03410
C                                                                       BLE03420
      IF(KM.LT.0)   E(J) = -E(J)                                        BLE03430
      WRITE(15,280) E(J), SM(JJ)                                        BLE03440
  280 FORMAT(/E20.12,E13.4,'= EIGENVALUE, NORM(ERROR)**2,EIGENVECTOR='/)BLE03450
  290 WRITE(15,300) (Q(L), L=LINT,LFIN)                                 BLE03460
      WRITE(15,310)                                                     BLE03470
  300 FORMAT(4E20.12)                                                   BLE03480
  310 FORMAT(/' ABOVE ARE COMPUTED APPROXIMATE EIGENVECTORS'/)          BLE03490
C                                                                       BLE03500
      IF(ITER.GT.MAXIT) WRITE(15,320) ITER,MAXIT                        BLE03510
  320 FORMAT(//' PROCEDURE TERMINATED BECAUSE NUMBER OF MATRIX-VECTOR MUBLE03520
     1LTIPLIES ',I6/' EXCEEDED MAXIMUM NUMBER ',I6,' ALLOWED'//)        BLE03530
C                                                                       BLE03540
      IF(ITER.LT.0) WRITE(15,330)                                       BLE03550
  330 FORMAT(//' USER BEWARE.  EIGENELEMENT COMPUTATIONS DEFAULTED BECAUBLE03560
     1SE'/'   EISPACK SUBROUTINE DEFAULTED.   EIGENVALUE AND EIGENVECTORBLE03570
     1 APPROXIMATIONS'/'   ABOVE WERE THOSE AVAILABLE AT THE TIME OF DEFBLE03580
     1AULT'/' SOMETHING IS SERIOUSLY WRONG.'//)                         BLE03590
C                                                                       BLE03600
C     CHECK FOR TERMINATION AFTER PHASE 1                               BLE03610
C     ITER < 0 MEANS EISPACK SUBROUTINE DEFAULTED                       BLE03620
C     IPHASE = MPMIN MEANS THAT PHASE 2 TERMINATED DUE TO ORTHOGONALITY BLE03630
C     IWRITE = MPMIN MEANS THAT CONVERGENCE APPEARS TO HAVE STAGNATED   BLE03640
C     ITER > MAXIT MEANS MAXIMUM NUMBER OF MATRIX-VECTOR MULTIPLIES     BLE03650
C            ALLOWED BY USER WAS EXCEEDED                               BLE03660
      IF(ITER.LT.0.OR.ITER.GT.MAXIT) GO TO 530                          BLE03670
      IF(IPHASE.EQ.MPMIN.OR.IWRITE.EQ.MPMIN) GO TO 530                  BLE03680
      IF(EFLAG.NE.1.OR.IPHASE.EQ.2) GO TO 530                           BLE03690
C                                                                       BLE03700
C     ENTER 2ND PHASE OF COMPUTATION TO ATTEMPT TO OBTAIN MORE          BLE03710
C     ACCURATE EIGENVECTOR APPROXIMATIONS.                              BLE03720
C     USER CONTROLS THE SIZE OF THE ERROR TOLERANCE BY SPECIFYING       BLE03730
```

```
C     THE PARAMETER RELTOL.                                             BLE03740
C                                                                       BLE03750
      IPHASE = 2                                                        BLE03760
      MAXIT = MAXIT2                                                    BLE03770
      KSET = KACT                                                       BLE03780
C                                                                       BLE03790
C     ERROR TOLERANCE USES THE CONVERGED EIGENVALUE LARGEST IN          BLE03800
C     MAGNITUDE.                                                        BLE03810
      TD(1) = DABS(E(1))                                                BLE03820
      IF(KML.EQ.1) GO TO 350                                            BLE03830
      DO 340 J = 2,KML                                                  BLE03840
  340 IF(DABS(E(J)).GT.TD(1))    TD(1) = DABS(E(J))                     BLE03850
  350 TD(1) = DMAX1(TD(1),1.D0)                                         BLE03860
      ERRMAN = RELTOL**2 * TD(1)**2                                     BLE03870
      IF(ERRMAN.GE.ERRMAX) GO TO 430                                    BLE03880
      ERRMAX = ERRMAN                                                   BLE03890
C                                                                       BLE03900
      WRITE(6,360) ERRMAX, MAXIT2                                       BLE03910
  360 FORMAT(//' ENTER PHASE 2 OF COMPUTATION'/' CONVERGENCE CRITERION IBLE03920
     1S REDUCED TO ',E13.4/' NO MORE THAN ',I5,' MATRIX VECTOR MULTIPLIEBLE03930
     1S WILL BE ALLOWED.'/' PROGRAM WILL TERMINATE IF BLOCK ORTHGONALITYBLE03940
     1 PROBLEMS MATERIALIZE')                                           BLE03950
C                                                                       BLE03960
      GO TO 190                                                         BLE03970
C                                                                       BLE03980
C     INCONSISTENCIES IN THE DATA                                       BLE03990
C                                                                       BLE04000
  370 WRITE(6,380) KM,KACT                                              BLE04010
  380 FORMAT(/' PROGRAM TERMINATES BECAUSE THE NUMBER OF EIGENELEMENTS  BLE04020
     1REQUESTED, KM =',I3/'  IS LARGER THAN THE SIZE OF THE FIRST Q BLOCBLE04030
     1K, KACT =',I3,' SPECIFIED'/'  USER MUST RESET KM OR KACT')        BLE04040
       GO TO 530                                                        BLE04050
C                                                                       BLE04060
  390 WRITE(6,400) KMAX,N                                               BLE04070
  400 FORMAT(/' PROGRAM TERMINATES BECAUSE KMAX = ',I5,' IS TOO LARGE FOBLE04080
     1R THE SIZE, N = ',I5,', OF THE GIVEN MATRIX'/'  USER MUST DECREASEBLE04090
     1THE SIZE OF KMAX.'/)                                              BLE04100
      GO TO 530                                                         BLE04110
C                                                                       BLE04120
  410 WRITE(6,420) NOLD,N,KACT,KSET                                     BLE04130
  420 FORMAT(/' PROGRAM TERMINATES BECAUSE FAULT OCCURRED IN READING IN BLE04140
     1THE EIGENVECTOR APPROXIMATIONS'/' EITHER THE SIZE MATRIX SPECIFIEDBLE04150
     1ON THE EIGENVECTOR FILE' ,I6/'  DID NOT MATCH THE SIZE SPECIFIED 'BLE04160
     1,I5,' IN THE PROGRAM OR THE NUMBER'/'    OF VECTORS IN FILE 10 = 'BLE04170
     1,I4,' IS LESS THAN THE NUMBER ',I3/'   USER SAID WERE THERE'/)    BLE04180
      GO TO 530                                                         BLE04190
C                                                                       BLE04200
  430 WRITE(6,440)  ERRMAN, ERRMAX                                      BLE04210
  440 FORMAT(/' COMPUTED PHASE 2 CONVERGENCE CRITERION ',E13.4/'  IS LARBLE04220
     1GER THAN PHASE 1 CRITERION ',E13.4/'  SO PROGRAM TERMINATES'/)    BLE04230
      GO TO 530                                                         BLE04240
C                                                                       BLE04250
  450 WRITE(6,460)  KACT,MXBLK                                          BLE04260
  460 FORMAT(/' PROGRAM TERMINATES BECAUSE THERE IS NOT ENOUGH ROOM TO  BLE04270
     1GENERATE 2 BLOCKS',' BECAUSE KACT = ',I3,' AND MXBLK = ', I4/)    BLE04280
```

```
      GO TO 530                                                         BLE04290
C                                                                       BLE04300
C                                                                       BLE04310
  470 WRITE(6,480)  MDIMTM, MXBLK                                       BLE04320
  480 FORMAT(/' PROGRAM TERMINATES BECAUSE THE DIMENSION ',I6,' OF THE TBLE04330
     1M ARRAY'/'   IS TOO SMALL FOR THE LARGEST T-MATRIX ALLOWED ',I4)  BLE04340
      GO TO 530                                                         BLE04350
C                                                                       BLE04360
  490 WRITE(6,500)                                                      BLE04370
  500 FORMAT(/' USER SPECIFIED NUMBER OF EIGENVALUES OF INTEREST AS 0'/'BLE04380
     1 PROGRAM TERMINATES FOR USER TO RESET KM TO DESIRED NONZERO VALUE'BLE04390
     1/)                                                                BLE04400
      GO TO 530                                                         BLE04410
C                                                                       BLE04420
  510 WRITE(6,520)  MDIMQ, KMAX,N                                       BLE04430
  520 FORMAT(/' PROGRAM TERMINATES BECAUSE THE DIMENSION ',I6,' OF THE QBLE04440
     1-ARRAY'/'   IS TOO SMALL TO HOLD ',I5, ' VECTORS OF LENGTH ',I4)  BLE04450
      GO TO 530                                                         BLE04460
C                                                                       BLE04470
  530 CONTINUE                                                          BLE04480
C                                                                       BLE04490
      STOP                                                              BLE04500
C-----END OF MAIN PROGRAM FOR BLOCK LANCZOS PROCEDURE-------------------BLE04510
      END                                                               BLE04520
```

8.4 BLMULT: Sample Matrix-Vector Multiply Subroutines

```
C-----BLMULT-----------------------------------------------------------BLM00010
C  Authors:  Jane Cullum* and Bill Donath**                            BLM00020
C         **IBM Research, T.J. Watson Research Center                  BLM00030
C         **Yorktown Heights, N.Y. 10598                               BLM00040
C         * Los Alamos National Laboratory                             BLM00045
C         * Los Alamos, New Mexico 87544                               BLM00050
C           E-mail:  cullumj@lanl.gov                                  BLM00060
C                                                                      BLM00070
C  These codes are copyrighted by the authors.  These codes            BLM00080
C  and modifications of them or portions of them are NOT to be         BLM00090
C  incorporated into any commercial codes or used for any other        BLM00100
C  commercial purposes such as consulting for other companies,         BLM00110
C  without legal agreements with the authors of these Codes.           BLM00120
C  If these Codes or portions of them are used in other scientific or   BLM00130
C  engineering research works the names of the authors of these codes   BLM00140
C  and appropriate references to their written work are to be          BLM00150
C  incorporated in the derivative works.                               BLM00160
C                                                                      BLM00170
C  This header is not to be removed from these codes.                  BLM00180
C                                                                      BLM00190
C        REFERENCE: Cullum and Willoughby, Chapter 7,                  BLM00191
C        Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsBLM00192
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   BLM00193
C        Applied Mathematics, 2002. SIAM Publications,                 BLM00194
C        Philadelphia, PA. USA                                         BLM00195
C                                                                      BLM00196
C                                                                      BLM00200
C     CONTAINS SAMPLE USPEC AND BMATV SUBROUTINES FOR USE WITH          BLM00210
C     THE BLOCK LANCZOS PROCEDURE FOR REAL SYMMETRIC MATRICES.          BLM00220
C     PROGRAMS ARE USED WITH BLEVAL AND BLSUB FILES.                    BLM00230
C                                                                      BLM00240
C     NONPORTABLE CONSTRUCTIONS:                                       BLM00250
C     1.  THE ENTRY MECHANISM USED TO PASS THE STORAGE                 BLM00260
C         LOCATIONS OF THE USER-SPECIFIED MATRIX FROM THE              BLM00270
C         SUBROUTINE USPEC TO THE MATRIX-VECTOR SUBROUTINE             BLM00280
C         BMATV.                                                       BLM00290
C     2.  IN THE SAMPLE USPEC AND BMATV SUBROUTINES FOR DIAGONAL        BLM00300
C         TEST MATRICES: FREE FORMAT (8,*) AND THE FORMAT (20A4).       BLM00310
C                                                                      BLM00320
C-----USPEC (GENERAL SYMMETRIC SPARSE MATRICES)-------------------------BLM00330
C                                                                      BLM00340
C     SUBROUTINE USPEC(N,MATNO,NNZ,AVER)                               BLM00350
      SUBROUTINE GUSPEC(N,MATNO,NNZ,AVER)                              BLM00360
C                                                                      BLM00370
C----------------------------------------------------------------------BLM00380
      DOUBLE PRECISION  ASD(10000),AD(5010),AVER,NNZ                   BLM00390
      INTEGER  IROW(10000),ICOL(5010)                                  BLM00400
C----------------------------------------------------------------------BLM00410
C    USPEC DIMENSIONS AND INITIALIZES THE ARRAYS NEEDED TO DEFINE       BLM00420
C    THE USER-SPECIFIED MATRIX AND THEN PASSES THE STORAGE LOCATIONS    BLM00430
C    OF THESE ARRAYS TO THE MULTIPLY SUBROUTINE BMATV.                 BLM00440
```

```
C
C    MATRIX IS STORED IN FOLLOWING SPARSE MATRIX FORMAT:
C    N = ORDER OF A-MATRIX,
C    NZS = NUMBER OF NONZERO SUBDIAGONAL ENTRIES,
C    NZL = INDEX OF LAST COLUMN CONTAINING NONZERO SUBDIAGONAL ENTRIES,
C    ICOL(J), J=1,NZL IS THE NUMBER OF NONZERO SUBDIAGONAL ELEMENTS
C             IN COLUMN J.
C    IROW(K), K = 1,NZS IS THE CORRESPONDING ROW INDEX FOR ASD(K).
C    AD(I), I=1,N CONTAINS DIAGONAL ENTRIES (INCLUDING ANY 0
C           DIAGONAL ENTRIES).
C    ASD(K), K=1,NZS CONTAINS NONZERO SUBDIAGONAL ENTRIES, BY COLUMN
C    FOR J > NZL THERE ARE NO NONZERO SUBDIAGONAL ELEMENTS IN COLUMN J.
C    ICOL(J) = 0 IS ALLOWED
C
C-----------------------------------------------------------------------
C     ARRAYS THAT DEFINE THE MATRIX ARE READ IN FROM FILE 8
C
      READ(8,10) NZS,NOLD,NZL,MATOLD
   10 FORMAT(I10,2I6,I8)
C
      WRITE(6,20) NZS,NOLD,NZL,MATOLD
   20 FORMAT(I10,2I6,I8,' = NZS,NOLD,NZL,MATOLD'/)
C
C     TEST OF PARAMETER CORRECTNESS
      ITEMP = (NOLD-N)**2 + (MATNO-MATOLD)**2
C
      IF(ITEMP.EQ.0) GO TO 40
C
      WRITE(6,30) NOLD,N,MATOLD,MATNO
   30 FORMAT(/' PROGRAM TERMINATES BECAUSE EITHER THE SIZE ',I4,' OF THE
     1 MATRIX'/' READ FROM FILE 8 DIFFERS FROM THE SIZE ',I4,' SPECIFIED
     1 BY'/' THE USER OR THE MATNO ',I8,' READ IN DIFFERS FROM THE MATNO
     1 '/ I8,' SPECIFIED BY THE USER'/)
      GO TO 100
C
   40 CONTINUE
C
C     NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN EACH COLUMN IS READ
C     THEN THE CORRESPONDING ROW INDEX FOR EACH SUCH ENTRY IS READ
      READ(8,50) (ICOL(K), K=1,NZL)
      READ(8,50) (IROW(K), K=1,NZS)
   50 FORMAT(13I6)
C
C     DIAGONAL IS READ FIRST, THEN NONZERO BELOW DIAGONAL ENTRIES
      READ(8,60) (AD(K), K=1,N)
      READ(8,60) (ASD(K), K=1,NZS)
   60 FORMAT(4E19.10)
C
C     COMPUTE NNZ, THE AVERAGE NUMBER OF NONZEROS PER COLUMN, AND
C     AVER, THE AVERAGE SIZE OF NONZERO ENTRIES.
      ITCOL = 0
      AVER = 0.D0
      DO 70 K = 1,N
      IF(DABS(AD(K)).EQ.0.D0)  GO TO 70
      ITCOL = ITCOL + 1
```

```
      AVER = AVER + DABS(AD(K))                                         BLM01000
   70 CONTINUE                                                          BLM01010
      NTCOL = ITCOL                                                     BLM01020
      DO 80 K = 1,N                                                     BLM01030
   80 ITCOL = ITCOL + 2*ICOL(K)                                         BLM01040
      NNZ = DFLOAT(ITCOL)/DFLOAT(N)                                     BLM01050
      DO 90 K = 1,NZS                                                   BLM01060
   90 AVER = AVER + DABS(ASD(K))                                        BLM01070
      AVER = AVER/DFLOAT(NZS + NTCOL)                                   BLM01080
C                                                                       BLM01090
C-----------------------------------------------------------------------BLM01100
C     PASS STORAGE LOCATIONS OF ARRAYS THAT DEFINE THE MATRIX TO        BLM01110
C     THE MATRIX-VECTOR MULTIPLY SUBROUTINE BMATV                       BLM01120
C                                                                       BLM01130
      CALL BMATVE(ASD,AD,ICOL,IROW,N,NZL)                               BLM01140
C-----------------------------------------------------------------------BLM01150
C                                                                       BLM01160
      RETURN                                                            BLM01170
  100 STOP                                                              BLM01180
C-----END OF USPEC------------------------------------------------------BLM01190
      END                                                               BLM01200
C                                                                       BLM01210
C-----MATRIX-VECTOR MULTIPLY FOR REAL SPARSE SYMMETRIC MATRICES---------BLM01220
C                                                                       BLM01230
C     SUBROUTINE BMATV(W,U)                                             BLM01240
      SUBROUTINE GBMATV(W,U)                                            BLM01250
C                                                                       BLM01260
C-----------------------------------------------------------------------BLM01270
      DOUBLE PRECISION   U(1),W(1),ASD(1),AD(1)                         BLM01280
      INTEGER  IROW(1),ICOL(1)                                          BLM01290
      COMMON/LOOPS/MAXIT,ITER                                           BLM01300
C-----------------------------------------------------------------------BLM01310
C    SPARSE MATRIX-VECTOR MULTIPLY FOR LANCZS  U = A*W                  BLM01320
C    SEE USPEC SUBROUTINE FOR DESCRIPTION OF THE ARRAYS THAT DEFINE     BLM01330
C    THE A-MATRIX                                                       BLM01340
C-----------------------------------------------------------------------BLM01350
C                                                                       BLM01360
      GO TO 3                                                           BLM01370
C                                                                       BLM01380
C-----------------------------------------------------------------------BLM01390
C    STORAGE LOCATIONS OF ARRAYS ARE PASSED TO BMATV FROM USPEC         BLM01400
C                                                                       BLM01410
      ENTRY BMATVE(ASD,AD,ICOL,IROW,N,NZL)                              BLM01420
C-----------------------------------------------------------------------BLM01430
C                                                                       BLM01440
      GO TO 4                                                           BLM01450
C                                                                       BLM01460
    3 CONTINUE                                                          BLM01470
C     INCREMENT THE A*W COUNTER                                         BLM01480
      ITER = ITER + 1                                                   BLM01490
C     COMPUTE THE DIAGONAL TERMS                                        BLM01500
      DO 10 I = 1,N                                                     BLM01510
   10 U(I) = AD(I)*W(I)                                                 BLM01520
C                                                                       BLM01530
C     COMPUTE BY COLUMN                                                 BLM01540
```

```
      LLAST = 0
      DO 30 J = 1,NZL
C
      IF (ICOL(J).EQ.0) GO TO 30
      LFIRST = LLAST + 1
      LLAST = LLAST + ICOL(J)
C
      DO 20 L = LFIRST,LLAST
      I = IROW(L)
C
      U(I) = U(I) + ASD(L)*W(J)
      U(J) = U(J) + ASD(L)*W(I)
C
   20 CONTINUE
C
   30 CONTINUE
C
    4 RETURN
C-----END OF BMATV-----------------------------------------------------
      END
C
C-----MATRIX-VECTOR MULTIPLY FOR DIAGONAL TEST MATRICES-----------------
C     BMATV COMPUTES  U = (DIAGONAL MATRIX) * W
C
      SUBROUTINE BMATV(W,U)
C     SUBROUTINE DBMATV(W,U)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  W(1),U(1),D(1)
      COMMON/LOOPS/MAXIT,ITER
C-----------------------------------------------------------------------
      GO TO 3
C-----------------------------------------------------------------------
C    STORAGE LOCATIONS OF ARRAYS ARE PASSED TO BMATV FROM USPEC
      ENTRY MVDIAE(D,N)
C-----------------------------------------------------------------------
      GO TO 4
C
    3 CONTINUE
C     INCREMENT THE LOOP COUNTER
      ITER = ITER + 1
C
      DO 10 I=1,N
   10 U(I)= D(I)*W(I)
C  10 U(I)= -D(I)*W(I)
C
    4 RETURN
C
C-----END OF DIAGONAL TEST MATRIX  MULTIPLY-----------------------------
      END
C
C-----START OF USPEC FOR DIAGONAL TEST MATRIX---------------------------
C
      SUBROUTINE USPEC(N,MATNO,NNZ,AVER)
C     SUBROUTINE DUSPEC(N,MATNO,NNZ,AVER)
```

```
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  D(1000),SPACE,SHIFT,AVER,NNZ
      DOUBLE PRECISION  DABS, DFLOAT
      REAL  EXPLAN(20)
C-----------------------------------------------------------------------
C
      READ(8,10) EXPLAN
   10 FORMAT(20A4)
      READ(8,*) NOLD,NUNIF,SPACE,D(1),SHIFT
      NNUNIF = NOLD - NUNIF
      WRITE(6,20) NOLD,SPACE,NNUNIF,D(1),SHIFT
   20 FORMAT(/' DIAGONAL TEST MATRIX, SIZE = ',I4/' MOST ENTRIES ARE ',
     1E10.3,' UNITS APART.',I3,' ENTRIES'/' ARE IRREGULARLY SPACED. FIRS
     1T ENTRY IS ',E10.3,' SHIFT = ',E10.3/)
C
      IF(N.NE.NOLD) GO TO 100
C     COMPUTE THE UNIFORM PORTION OF THE SPECTRUM
      DO 30 J=2,NUNIF
   30 D(J) = D(1) - DFLOAT(J-1)*SPACE
      NUNIF1=NUNIF + 1
      READ(8,10)  EXPLAN
      DO 40 J=NUNIF1,N
   40 READ(8,*) D(J)
C
      IF(SHIFT.EQ.0.) GO TO 60
      DO 50 J=1,N
   50 D(J) = D(J) + SHIFT
C
C     PRINT OUT THE EIGENVALUES OF INTEREST
   60 WRITE(6,70) (D(I), I=1,10 )
      NB = NUNIF - 2
      WRITE(6,80) (D(I), I = NB,N)
   70 FORMAT(/' BLOCK LANCZOS TEST, 1ST 10 ENTRIES OF DIAGONAL TEST MATR
     1IX = '/(3E22.14))
   80 FORMAT(/' MIDDLE UNIFORM PORTION OF MATRIX IS NOT PRINTED OUT'/
     1' END OF UNIFORM PLUS NONUNIFORM SECTION = '/(3E25.16))
C
C     DIAGONAL GENERATION COMPLETE
C     COMPUTE NNZ AND AVER
      NNZ = 1.D0
      AVER = 0.D0
      DO 90 K = 1,N
   90 AVER = AVER + DABS(D(K))
      AVER = AVER/DFLOAT(N)
C
C-----------------------------------------------------------------------
C    CALL ENTRY TO MATRIX-VECTOR MULTIPLY SUBROUTINE  TO PASS
C    STORAGE LOCATION OF D-ARRAY AND ORDER OF A-MATRIX.
C
       CALL MVDIAE(D,N)
C-----------------------------------------------------------------------
C
      RETURN
  100 WRITE(6,110) NOLD,N
```

```
  110 FORMAT(' PROGRAM TERMINATES BECAUSE NOLD = ',I5,'DOES NOT EQUAL N BLM02650
     1 =',I5)                                                            BLM02660
C-----END OF USPEC SUBROUTINE FOR 'DIAGONAL' TEST MATRICES--------------BLM02670
      STOP                                                              BLM02680
      END                                                               BLM02690
```

8.5 BLSUB: Other Subroutines used by the Codes in Chapters 8 and 9

```
C-----BLSUB-------------------------------------------------------------BLS00010
C  Authors:  Jane Cullum* and Bill Donath**                             BLS00020
C         **IBM Research, T.J. Watson Research Center                   BLS00030
C         **Yorktown Heights, N.Y. 10598                                BLS00040
C         * Los Alamos National Laboratory                              BLS00050
C         * Los Alamos, New Mexico 87544                                BLS00060
C           E-mail:  cullumj@lanl.gov                                   BLS00065
C                                                                       BLS00070
C  These codes are copyrighted by the authors.  These codes             BLS00080
C  and modifications of them or portions of them are NOT to be          BLS00090
C  incorporated into any commercial codes or used for any other         BLS00100
C  commercial purposes such as consulting for other companies,          BLS00110
C  without legal agreements with the authors of these Codes.            BLS00120
C  If these Codes or portions of them are used in other scientific or   BLS00130
C  engineering research works the names of the authors of these codes   BLS00140
C  and appropriate references to their written work are to be          BLS00150
C  incorporated in the derivative works.                                BLS00160
C                                                                       BLS00170
C  This header is not to be removed from these codes.                   BLS00180
C                                                                       BLS00190
C        REFERENCE: Cullum and Willoughby, Chapter 7,                   BLS00191
C        Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsBLS00192
C        VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   BLS00193
C        Applied Mathematics, 2002. SIAM Publications,                  BLS00194
C        Philadelphia, PA. USA                                          BLS00195
C                                                                       BLS00196
C                                                                       BLS00200
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE               BLS00210
C     CONSTRUCTIONS:                                                    BLS00220
C     1. ENTRY MECHANISMS USED TO PASS THE STORAGE LOCATIONS OF         BLS00230
C        SEVERAL ARRAYS FROM THE MAIN PROGRAM TO THE SUBROUTINES        BLS00240
C        LANCZS AND LANCI1.                                             BLS00250
C     2. COMMON BLOCK: LOOPS: USED IN LANCZS AND LANCI1.                BLS00260
C                                                                       BLS00270
C     SUBROUTINES:    LANCZS, LANCI1, ORTHOG, START, AND DIAGOM         BLS00280
C                     ARE USED WITH THE BLOCK LANCZOS PROGRAMS          BLS00290
C                     BLEVAL AND BLIEVAL.  LPERM IS USED WITH BLIEVAL.  BLS00300
C                                                                       BLS00310
C                                                                       BLS00320
C-----LANCZS FOR BLOCK LANCZOS PROCEDURE--------------------------------BLS00330
C                                                                       BLS00340
C     ON EACH ITERATION CALLS LANCI1 SUBROUTINE TO GENERATE             BLS00350
C     THE Q-SUBBLOCKS AND THEN CALLS DIAGOM SUBROUTINE TO               BLS00360
C     DIAGONALIZE THE SMALL SYMMETRIC MATRIX WHICH IS THE PROJECTION    BLS00370
C     OF THE MATRIX BEING USED BY LANCZS ONTO THE SUBSPACE SPANNED      BLS00380
C     BY THESE Q-BLOCKS.                                                BLS00390
C                                                                       BLS00400
      SUBROUTINE LANCZS(MATVEC,KML,KSET,KACT,MXBLK,N,Q,E,RESIDL,        BLS00410
     1 RESIDK,ERRMAX,IPHASE,NITER,IWRITE)                               BLS00420
C                                                                       BLS00430
```

```
C-----------------------------------------------------------------------
      DOUBLE PRECISION  E(1),Q(1),ERRMAX,SPREC,RESN,FRACT,RKM,SUM
      DOUBLE PRECISION  TM(1),SM(1),TD(*),TOD(1),RESIDL(*),RESIDK(1)
      REAL  G(1)
      INTEGER  EFLAG,OFLAG,DIR(2,*),DESC(1),LEFT(1),XLFT(*)
      DOUBLE PRECISION FINPRO
      COMMON /LOOPS/MAXIT,ITER
      COMMON/FLAGS/EFLAG,OFLAG
      EXTERNAL MATVEC
C-----------------------------------------------------------------------
      GO TO 3
C-----------------------------------------------------------------------
C     ENTRY RECEIVES STORAGE LOCATIONS OF SEVERAL OF THE ARRAYS
C     USED BY THE LANCZS SUBROUTINE.  THIS ALLOWS USER TO SPECIFY
C     THE DIMENSIONS OF THESE ARRAYS IN THE MAIN PROGRAM.
C
      ENTRY LANZP(DIR,DESC,SM,TM,TOD,TD,G,XLFT,LEFT,SPREC)
      GO TO 4
C-----------------------------------------------------------------------
C
    3 CONTINUE
C
      KM = KML
      MMT = MXBLK*MXBLK
      MPMIN = -1000
      IKACT = KACT + 10
      FRACT = RESIDL(1)
      NSTAG = RESIDL(2)
      IORTHO = 0
C
C     CONSTRUCT STARTING VECTORS
      IF(KSET.EQ.0) GO TO 10
C-----------------------------------------------------------------------
      CALL ORTHOG(1,KSET,N,Q)
   10 CALL START (KSET+1,KACT,N,Q,G,ERRMAX)
C-----------------------------------------------------------------------
   20 CONTINUE
C     INITIALIZE THE LANCZOS T-MATRIX.
      DO 30 J=1,MMT
   30 TM(J)=0.D0
C
C     INITIALIZE THE Q-BLOCK DIRECTORY
      DIR(1,1)=1
      DIR(2,1)=KACT
C
C     ORTHOGONALIZE THE STARTING VECTORS
      IF(NITER.EQ.0) GO TO 40
C-----------------------------------------------------------------------
      CALL ORTHOG(1,KACT,N,Q)
C-----------------------------------------------------------------------
   40 CONTINUE
C
C     GENERATE THE QSUBBLOCKS USED ON ITERATION NITER AND STORE IN
C     THE Q-ARRAY
C
```

```
      DO 90 I=1,MXBLK
C
C-----------------------------------------------------------------------
      CALL LANCI1(MATVEC,MXBLK,NITER,I,N,Q,KACT,KML,ERRMAX,RESN,RKM,
     1 IND,KACTN,IWRITE)
C-----------------------------------------------------------------------
C
C     HAS CONVERGENCE OCCURRED?
      II = I+1
      IF (I.EQ.1.AND.DIR(2,I).EQ.DIR(2,II)) GO TO 140
C
C     WAS THERE ROOM FOR ANOTHER Q-BLOCK?
      IF (DIR(2,II).LT.DIR(1,II))  GO TO 100
C
C     IF OFLAG = 1 OR IPHASE = 2, CHECK THE ORTHOGONALITY OF
C     THE Q-SUBBLOCKS GENERATED WITH RESPECT TO THAT VECTOR
C     IN THE 1ST Q-BLOCK WHICH IS GENERATING DESCENDANTS.
C     IN PHASE 2 LOSSES IN ORTHOGONALITY ARE USED TO
C     DETERMINE WHEN THE LIMITS ON THE ACHIEVABLE ACCURACY HAVE
C     BEEN REACHED.
C
      IF(OFLAG.EQ.0.AND.IPHASE.EQ.1) GO TO 90
C
      L1=DIR(1,II)
      LL1 = (L1-1)*N + 1
      IND1 = (IND-1)*N + 1
C-----------------------------------------------------------------------
      SUM = FINPRO(N,Q(IND1),1,Q(LL1),1)
C-----------------------------------------------------------------------
C
      IF(DABS(SUM).LT.SPREC)GO  TO 80
C
      IF(IWRITE.EQ.1) WRITE(6,50)  IND,L1,SUM,I
   50 FORMAT(/' INNER PRODUCT OF VECTORS ',I3,' AND ',I3,' = ',E13.3/
     1' THIS VIOLATES ORTHOGONALITY TEST.  TERMINATE BLOCK GENERATION'
     1/' WITH ',I3,'TH BLOCK '/)
C
C     ORTHOGONALITY TEST VIOLATED, TERMINATE BLOCK GENERATION
C     FOR THIS ITERATION.  IN PHASE 2 KEEP TRACK OF NUMBER OF
C     SUCH VIOLATIONS THAT LIMIT THE NUMBER OF BLOCKS TO < 10.
C     TERMINATE AFTER 3 SUCH VIOLATIONS IN PHASE 2.
      IF(IPHASE.NE.1.AND.I.LT.IKACT)  IORTHO = IORTHO + 1
      IF(IORTHO.LT.3.AND.II.NE.2)  GO TO 70
      WRITE(6,60)
   60 FORMAT(/' THE ORTHOGONALITY TEST HAS FAILED THREE TIMES'/
     1' TERMINATE THE BLOCK PROCEDURE'/)
      IPHASE = -1000
C     BEFORE TERMINATING WRITE THE CURRENT EIGENVECTOR/EIGENVALUE
C     APPROXIMATIONS TO FILE 15
      GO TO 160
C
C     TERMINATE THE Q-BLOCK GENERATION ON THIS ITERATION
   70 DIR(2,II)=DIR(2,I)
      GO TO 100
C
```

```
   80 CONTINUE
C
C     END OF ORTHOGONALITY TESTS
C
   90 CONTINUE
C
C     END OF RECURSIVE Q-BLOCK GENERATION
C
  100 CONTINUE
      MM = DIR(2,II)
      IF(IWRITE.EQ.1)  WRITE (6,110)  MM,I
  110 FORMAT(' T-MATRIX IS OF ORDER ',I3, ' NUMBER OF BLOCKS = ',I3)
C
C-----------------------------------------------------------------------
C     DIAGONALIZE THE PROJECTION MATRIX TM.  ON RETURN THE
C     UPDATED APPROXIMATIONS TO THE DESIRED EIGENVECTORS ARE IN THE
C     FIRST KACT COLUMNS OF THE Q-ARRAY.
C     UPDATED EIGENVALUE APPROXIMATIONS ARE IN E.
      TD(1) = RKM
      TD(2) = FRACT
      IERR = NSTAG
C
      CALL DIAGOM(MXBLK,MM,TM,KACT,N,Q,E,RESIDL,RESIDK,
     1 RESN,IND,KACTN,KM,TD,TOD,NITER,IERR,IWRITE)
C-----------------------------------------------------------------------
C
C     INCREMENT COUNTER FOR NUMBER OF BLOCK LANCZOS ITERATIONS
      NITER = NITER + 1
C     IWRITE = MPMIN MEANS BLOCK LANCZOS PROCEDURE TERMINATED ABNORMALLY
      IF(IWRITE.EQ.MPMIN)  GO TO 160
C     IERR .NE. 0 MEANS EISPACK SUBROUTINE DEFAULTED
      IF(IERR.EQ.0) GO TO 130
      WRITE(6,120)
  120 FORMAT(/' EISPACK SIGNALS TROUBLE IN SMALL IMTQL2 EIGENVALUE SUBRO
     1UTINE,'/' SO BLOCK LANCZOS PROGRAM TERMINATES'/)
      ITER = -ITER
C
      RETURN
C
  130 IF (ITER.GE.MAXIT) GO TO 160
C
C     UPDATED APPROXIMATIONS WERE OBTAINED WITHOUT EXCEEDING
C     MAXIMUM NUMBER OF MATRIX-VECTOR MULTIPLIES SET BY THE USER.
C     CONTINUE BLOCK LANCZOS LOOP ITERATIONS
C
      GO TO 20
C
  140 WRITE(6,150)
  150 FORMAT(//' BLOCK LANCZOS PROCEDURE CONVERGED'//)
C
C     BLOCK LANCZOS PROCEDURE HAS CONVERGED.
C     ATTEMPT TO IMPROVE THE APPROXIMATE EIGENVECTORS BY DIAGONALIZING
C     THE SMALL PROJECTION MATRIX OBTAINED BY USING ONLY THE
C     FIRST BLOCK IN Q-ARRAY.
C
```

```
  160 KACT2 = KACT*MXBLK                                                BLS02090
      DO 170 KK = 1,KACT2                                               BLS02100
  170 TM(KK) = 0.D0                                                     BLS02110
C-----------------------------------------------------------------------BLS02120
      CALL ORTHOG(1,KACT,N,Q)                                           BLS02130
C-----------------------------------------------------------------------BLS02140
      KK0 = 1-N                                                         BLS02150
      KACTP1 = (KACT)*N + 1                                             BLS02160
      JJ0 = -MXBLK-1                                                    BLS02170
      DO 190 K=1,KACT                                                   BLS02180
      JJ0 = JJ0 + MXBLK + 1                                             BLS02190
      KK0 = KK0 + N                                                     BLS02200
C-----------------------------------------------------------------------BLS02210
      CALL MATVEC(Q(KK0),Q(KACTP1))                                     BLS02220
C-----------------------------------------------------------------------BLS02230
      LL0 = (K-2)*N + 1                                                 BLS02240
      JJ = JJ0                                                          BLS02250
      DO 180 L=K,KACT                                                   BLS02260
      LL0 = LL0 + N                                                     BLS02270
      JJ=JJ+1                                                           BLS02280
C-----------------------------------------------------------------------BLS02290
      TM(JJ) = FINPRO(N,Q(LL0),1,Q(KACTP1),1)                           BLS02300
C-----------------------------------------------------------------------BLS02310
  180 CONTINUE                                                          BLS02320
C                                                                       BLS02330
  190 CONTINUE                                                          BLS02340
C                                                                       BLS02350
C-----------------------------------------------------------------------BLS02360
C     USE EISPACK SUBROUTINE TRED2 TO TRIDIAGONALIZE TM-MATRIX          BLS02370
C     TM = (1ST Q-BLOCK)-TRANSPOSE*A*(1ST Q-BLOCK).                     BLS02380
C     ON RETURN DIAGONAL ELEMENTS COMPUTED ARE IN TD, OFF-DIAGONAL      BLS02390
C     ELEMENTS ARE IN TOD, TRANSFORMATIONS USED ARE IN TM.              BLS02400
C     THEN USE EISPACK SUBROUTINE IMTQL2 TO DIAGONALIZE THE T-MATRIX.   BLS02410
C     ON RETURN.  EIGENVALUES ARE IN TD IN ASCENDING ORDER.             BLS02420
C     CORRESPONDING EIGENVECTORS ARE IN TM.                             BLS02430
C                                                                       BLS02440
      CALL TRED2(MXBLK,KACT,TM,TD,TOD,TM)                               BLS02450
      CALL IMTQL2(MXBLK,KACT,TD,TOD,TM,IERR)                            BLS02460
C-----------------------------------------------------------------------BLS02470
C                                                                       BLS02480
      IF(IERR.EQ.0)  GO TO 200                                          BLS02490
      WRITE(6,120)                                                      BLS02500
      ITER = -ITER                                                      BLS02510
C                                                                       BLS02520
      RETURN                                                            BLS02530
C                                                                       BLS02540
C     COMPUTE SUCCESSIVELY THE JTH-COMPONENTS OF THE RITZ VECTORS.      BLS02550
C     REORDER THE EIGENVALUES (AND EIGENVECTORS) SO THAT THEY           BLS02560
C     ARE IN ALGEBRAICALLY DECREASING ORDER.                            BLS02570
C                                                                       BLS02580
  200 DO 220 J=1,N                                                      BLS02590
      JJ0 = - MXBLK                                                     BLS02600
      JL0 = -N + J                                                      BLS02610
      DO 210 K=1,KACT                                                   BLS02620
      TOD(K)=0.D0                                                       BLS02630
```

```
      JJ0 = JJ0 + MXBLK
      JJ= JJ0
      JL = JL0
      DO 210 L=1,KACT
      JJ=JJ+1
      JL = JL + N
  210 TOD(K)=TOD(K)+TM(JJ)*Q(JL)
      JK = JL0
      DO 220 K=1,KACT
      JK = JK + N
      KACTK = KACT - K + 1
      Q(JK)=TOD(KACTK)
  220 CONTINUE
      DO 230 K=1,KACT
      KACTK = KACT - K + 1
  230 E(K)=TD(KACTK)
C
C     HAS CONVERGENCE OCCURRED?
      IF(I.EQ.1.AND.DIR(2,I).EQ.DIR(2,I+1))  GO TO 250
C
C     CONVERGENCE HAS NOT OCCURRED, PROCEDURE TERMINATED FOR SOME
C     OTHER REASON
      WRITE(6,240)
  240 FORMAT(//' BLOCK LANCZOS PROCEDURE TERMINATES WITHOUT CONVERGENCE'
     1/' AFTER WRITING THE CURRENT EIGENVALUE AND EIGENVECTOR APPROXIMAT
     1IONS'/' TO FILE 15'/)
C
      RETURN
C
  250 IF(IPHASE.EQ.1) WRITE(6,260) (E(K), K=1,KACT)
      IF(IPHASE.EQ.2) WRITE(6,270) (E(K), K=1,KACT)
  260 FORMAT(/' AT END OF PHASE 1, COMPUTED EIGENVALUES ='/(4E20.12))
  270 FORMAT(/' AT END OF PHASE 2, COMPUTED EIGENVALUES ='/(4E20.12))
C
C
C-----END OF LANCZS------------------------------------------------------
    4 RETURN
      END
C
C-----START OF LANCI1----------------------------------------------------
C     GENERATES THE Q-SUBBLOCKS ON EACH ITERATION OF THE BLOCK LANCZOS
C     PROCEDURE.
C
      SUBROUTINE LANCI1(MATVEC,MXBLK,NITER,I,N,Q,KACT,KML,ERRMAX,
     1RESN,RKM,IND,KACTN,IWRITE)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  Q(1),TM(1),S,SM(1),T,ERRMAX,SUM,RESN,RKM
      INTEGER  DIR(2,*),DESC(1),LEFT(1),XLFT(*)
      DOUBLE PRECISION FINPRO, DSQRT
      EXTERNAL MATVEC
C-----------------------------------------------------------------------
      GO TO 3
C-----------------------------------------------------------------------
C     ALLOWS PASSAGE OF LOCATIONS OF SOME OF THE ARRAYS USED BY LANCI1
```

```
C     SO THAT THESE ARRAYS CAN BE DIMENSIONED IN THE MAIN PROGRAM
C
      ENTRY LANCP1(DIR,DESC,TM,SM,XLFT,LEFT)
      GO TO 4
C-----------------------------------------------------------------------
    3 CONTINUE
C
C     SIZE OF FIRST BLOCK CAN CHANGE.
      IF(I.EQ.1) KACTN = KACT
C
C     XLFT(I+2) IS CUMULATIVE TOTAL OF VECTORS IN 1ST QBLOCK NOT
C     GENERATING DESCENDANTS.
C
      IF(I.GT.1) GO TO 10
      XLFT(1) = 0
      XLFT(2) = 0
   10 XLFT(I+2) = XLFT(I+1)
C
C     INITIALIZE THE DIRECTORY FOR NEXT QBLOCK Q(I+1)
C
      I2=DIR(2,I)
      I1=DIR(1,I)
      DIR(1,I+1)=I2+1
      DIR(2,I+1)=I2
C
C     IS THERE ROOM FOR ANOTHER QBLOCK?
C
      MS = I2-I1+1
      IF (MS+I2.LE.MXBLK) GO TO 70
C
C     NOT ENOUGH ROOM TO GENERATE ANOTHER BLOCK
C     COMPLETE THE TM-MATRIX. NOTE THAT THE TM-MATRIX IS
C     DIMENSIONED AS (MXBLK,1) AND THE EISPACK SUBROUTINES
C     REQUIRE THE LOWER TRIANGULAR PART OF THIS MATRIX.
C
      I3=I2+1
      JI30 = (I3-1)*N
      JI31 = JI30 + 1
      JK1 = (I1-2)*N + 1
      DO 60 K=I1,I2
      JK1 = JK1 + N
C-----------------------------------------------------------------------
      CALL MATVEC(Q(JK1),Q(JI31))
C-----------------------------------------------------------------------
C     COMPUTE LAST DIAGONAL BLOCK IN TM-MATRIX FOR THIS ITERATION
C
      JL1 = (K-2)*N + 1
      KK = (K-1)*MXBLK +  K - 1
   20 DO 30 L=K,I2
      KK = KK + 1
      JL1 = JL1 + N
C-----------------------------------------------------------------------
      TM(KK) = FINPRO(N,Q(JL1),1,Q(JI31),1)
C-----------------------------------------------------------------------
   30 CONTINUE
```

```
C
C     COMPUTE ASSOCIATED CORRECTION TERMS IN TM-MATRIX.
      IF(XLFT(I).EQ.0) GO TO 50
      LUP = XLFT(I)
      DO 40 JJ = 1,LUP
      L= LEFT(JJ)
      JL1 = (L-1)*N + 1
C-----------------------------------------------------------------------
      SUM = FINPRO(N,Q(JI31),1,Q(JL1),1)
C-----------------------------------------------------------------------
      KK = (L-1)*MXBLK + K
      TM(KK) = SUM + TM(KK)
   40 CONTINUE
C
   50 CONTINUE
C
   60 CONTINUE
C
      RETURN
C
C     ON EVERY BLOCK PASS THROUGH HERE TO GENERATE THE ITH-BLOCK
C     DIAGONAL ENTRY A(I) OF THE TM-MATRIX, EXCEPT THE LAST DIAGONAL
C     BLOCK WHICH IS GENERATED ABOVE
C
   70 CONTINUE
C     COMPUTE (A-MATRIX)*(ITH-Q-BLOCK)
      KA=I2
      DO 80 K=I1,I2
      KA=KA+1
      JKA1 = (KA-1)*N + 1
      JK1 = (K-1)*N + 1
C-----------------------------------------------------------------------
      CALL MATVEC(Q(JK1),Q(JKA1))
C-----------------------------------------------------------------------
      DESC(K)=KA
   80 DESC(KA)=K
C
C     COMPUTE (A-MATRIX)*(ITH-Q-BLOCK) - ((I-1)TH-Q-BLOCK)*B(I)-TRANS
C     WHERE B(I) DENOTES THE ITH SUBDIAGONAL BLOCK
C
      IF(I.EQ.1) GO TO 110
      J1 = DIR(1,I-1)
      J2 = DIR(2,I-1)
      DO 100 K=I1,I2
      KD=DESC(K)
      JKD0 = (KD-1)*N
      KK = (J1-2)*MXBLK + K
      DO 90 L=J1,J2
      JL = (L-1)*N
      KK = KK + MXBLK
      S=TM(KK)
      JKD = JKD0
      DO 90 J=1,N
      JKD = JKD + 1
      JL = JL + 1
```

```
   90 Q(JKD) = Q(JKD) - S*Q(JL)
  100 CONTINUE
      LINT = (KD-1)*N + 1
      LFIN = KD*N
C
C     COMPUTE A(I)
C
  110 DO 130 K=I1,I2
      KKMX = (K-1)*MXBLK
      KD=DESC(K)
      JKD1 = (KD-1)*N+ 1
      JL1 = (K-2)*N + 1
      DO 120 L=K,I2
      JL1 = JL1 + N
      KK = KKMX + L
C-----------------------------------------------------------------------
      TM(KK) = FINPRO(N,Q(JL1),1,Q(JKD1),1)
C-----------------------------------------------------------------------
  120 CONTINUE
  130 CONTINUE
C
C     COMPUTE P(I) = P(I) - (ITH-Q-BLOCK)*A(I)
C
      DO 170 K=I1,I2
      KKMX = (K-1)*MXBLK
      KD=DESC(K)
      JKD0 = (KD-1)*N
      JL = (I1-1)*N
      DO 140 L=I1,I2
      KK = KKMX + L
      IF(L.LT.K) KK=(L-1)*MXBLK + K
      S=TM(KK)
      JKD = JKD0
      DO 140 J=1,N
      JL = JL + 1
      JKD = JKD + 1
  140 Q(JKD) = Q(JKD) - S*Q(JL)
C
C     REORTHOGONALIZE THE BLOCK P(I) WITH RESPECT TO ALL VECTORS
C     IN THE 1ST QBLOCK THAT ARE NOT CURRENTLY GENERATING ANY
C     DESCENDANTS.  NOTE THAT 2ND Q-BLOCK IS REORTHOGONALIZED
C     ELSEWHERE.
      IF(XLFT(I).EQ.0) GO TO 170
      LUP = XLFT(I)
      DO 160 JJ = 1,LUP
      L= LEFT(JJ)
      JL0 = (L-1)*N
      LLMX = (L-1)*MXBLK
      JL1 = JL0 + 1
      JKD1 = JKD0 + 1
C-----------------------------------------------------------------------
      SUM = FINPRO(N,Q(JL1),1,Q(JKD1),1)
C-----------------------------------------------------------------------
      JKD = JKD0
      JL = JL0
```

```
      DO 150 J=1,N
      JKD = JKD + 1
      JL = JL + 1
  150 Q(JKD) = Q(JKD) - SUM* Q(JL)
      KK = LLMX + K
      TM(KK) = SUM + TM(KK)
C
  160 CONTINUE
  170 CONTINUE
C
C
C   GENERATE B(I+1)
C
      K1=DESC(I1)
      K2=DESC(I2)
      IFLAG=0
C
C     COMPUTE NORMS
C
  180 CONTINUE
      JK1 = (K1-2)*N + 1
      DO 190 K=K1,K2
      JK1 = JK1 + N
C-----------------------------------------------------------------------
      SM(K) = FINPRO(N,Q(JK1),1,Q(JK1),1)
C-----------------------------------------------------------------------
  190 CONTINUE
C
        IF(I.EQ.1.AND.K1.EQ.I2+1)  WRITE(6,200) NITER,
     1 (K,SM(K), K =K1,K2)
  200 FORMAT(//' ON ITERATION', I4,' NORM(GRADIENTS)**2 OF 1ST BLOCK = '
     1/5(I4,E12.3))
C
C     TEST FOR CONVERGENCE OF BLOCK LANCZOS
C
      IF(I.GT.1.OR.K1.GT.I2+1)  GO TO 250
C
C     TEST THE FIRST KM OF THE EIGENVALUES FOR CONVERGENCE
      K2L = K1 + KML - 1
      RKM = SM(K2L)
      DO 210 K = K1,K2L
      IF(SM(K).GT.ERRMAX )  GO TO 220
  210 CONTINUE
      GO TO 430
C
C     CAN WE REDUCE KACT?  IF A SMALL RESIDUAL (GRADIENT) IS IDENTIFIED,
C     SIZE OF 1ST BLOCK MAY BE REDUCED.
  220 IF(KML.EQ.KACT) GO TO 250
      DO 230 K = K2L,K2
      IF(SM(K).GT.ERRMAX) GO TO 230
      KSAV = K
      KACTN = KSAV - KACT
      GO TO 240
C
  230 CONTINUE
```

```
      GO TO 250                                                         BLS05390
C                                                                       BLS05400
  240 K2 = KSAV                                                         BLS05410
C                                                                       BLS05420
C     GENERATE THE TRANSPOSE OF B(I)                                    BLS05430
C                                                                       BLS05440
  250 CONTINUE                                                          BLS05450
C                                                                       BLS05460
C     DETERMINE THE MAXIMAL NORM                                        BLS05470
      K=K1                                                              BLS05480
      S=SM(K)                                                           BLS05490
      DO 260 L=K1,K2                                                    BLS05500
      IF (SM(L).LT.S) GOTO 260                                          BLS05510
      K=L                                                               BLS05520
      S=SM(L)                                                           BLS05530
  260 CONTINUE                                                          BLS05540
C     FOR 2ND QBLOCK, SAVE INDEX AND SIZE OF MAXIMAL NORM               BLS05550
      IF(I.GT.1) GO TO 270                                              BLS05560
      IND = K - KACT                                                    BLS05570
      RESN = SM(K)                                                      BLS05580
C                                                                       BLS05590
  270 IF(S.LE.ERRMAX)GO TO 340                                          BLS05600
C                                                                       BLS05610
      IF(IFLAG.EQ.1) GO TO 340                                          BLS05620
C                                                                       BLS05630
      S=DSQRT(S)                                                        BLS05640
      JK0 = (K-1)*N                                                     BLS05650
      JK = JK0                                                          BLS05660
      DO 280 J=1,N                                                      BLS05670
      JK = JK + 1                                                       BLS05680
  280 Q(JK)=Q(JK)/S                                                     BLS05690
      JL0 = (K1-2)*N                                                    BLS05700
      DO 310 L=K1,K2                                                    BLS05710
      JL0 = JL0 + N                                                     BLS05720
      LL=(DESC(L) - 1)*MXBLK + K1                                       BLS05730
      IF (L.NE.K) GOTO 290                                              BLS05740
      TM(LL)=S                                                          BLS05750
      GO TO 310                                                         BLS05760
  290 JK1 = JK0 + 1                                                     BLS05770
      JL1 = JL0 + 1                                                     BLS05780
C-----------------------------------------------------------------------BLS05790
      T = FINPRO(N,Q(JK1),1,Q(JL1),1)                                   BLS05800
C-----------------------------------------------------------------------BLS05810
      TM(LL)=T                                                          BLS05820
      JK = JK0                                                          BLS05830
      JL = JL0                                                          BLS05840
      DO 300 J=1,N                                                      BLS05850
      JK = JK + 1                                                       BLS05860
      JL = JL + 1                                                       BLS05870
  300 Q(JL) = Q(JL) - T*Q(JK)                                           BLS05880
  310 CONTINUE                                                          BLS05890
      IF (K.EQ.K1) GOTO 330                                             BLS05900
C                                                                       BLS05910
      JK1 = (K1-1)*N                                                    BLS05920
      JK = JK0                                                          BLS05930
```

```
      DO 320 J=1,N
      JK = JK + 1
      JK1 = JK1 + 1
      T=Q(JK1)
      Q(JK1)=Q(JK)
  320 Q(JK)=T
      MA=DESC(K)
      MB=DESC(K1)
      DESC(K1)=MA
      DESC(K)=MB
      DESC(MA)=K1
      DESC(MB)=K
  330 CONTINUE
C
      DIR(2,I+1)=K1
C
      IFLAG=1
C
      K1=K1+1
      IF(I.EQ.1) GO TO 340
      IF (K1.LE.K2) GO TO 180
C     RETURN TO LANCZS
C
      RETURN
C
C    IMPLICIT VECTOR DEFLATION
C
  340 CONTINUE
      J= XLFT(I+2)
      IF(K1.GT.K2) GO TO 360
      DO 350 L= K1,K2
      J = J+1
  350 LEFT(J) = DESC(L)
  360 XLFT(I+2) = J
C
C     FORCE REORTHGONALIZATION OF 2ND AND 3RD QBLOCKS W.R.T. THOSE
C     VECTORS IN 1ST QBLOCK THAT ARE NOT GENERATING DESCENDANTS
C     ON THIS ITERATION.
      IF(I.GT.1)  GO TO 370
      XLFT(1) = XLFT(3)
      XLFT(2) = XLFT(3)
  370 IJJ = I + 2
      IJJJ= XLFT(IJJ)
C
      IF(IJJJ.EQ.0) GO TO 390
      IF(IWRITE.EQ.1)  WRITE(6,380) (LEFT(IJ),IJ= 1,IJJJ)
  380 FORMAT(' VECTORS NOT GENERATING DESCENDANTS ARE '/(10I6))
C
  390 IF(I.EQ.1.AND.KML.GT.1) GO TO 400
C
      RETURN
C
C     REORTHOGONALIZE 2ND QBLOCK W.R.T VECTORS IN 1ST BLOCK NOT
C     GENERATING DESCENDANTS
  400 IF(XLFT(I).EQ.0) RETURN
```

```
      LUP = XLFT(I)
      KD = DIR(2,I+1)
      JKD0 = (KD-1)*N
      DO 420 JJ = 1,LUP
      L = LEFT(JJ)
      JL0 = (L-1)*N
      JL1 = JL0 + 1
      JKD1 = JKD0 + 1
C-----------------------------------------------------------------------
      SUM = FINPRO(N,Q(JKD1),1,Q(JL1),1)
C-----------------------------------------------------------------------
      JL = JL0
      JKD = JKD0
      DO 410 J=1,N
      JL = JL + 1
      JKD = JKD + 1
  410 Q(JKD) = Q(JKD) - SUM *Q(JL)
  420 CONTINUE
C
      RETURN
C
C     EXIT IF CONVERGENCE OF DESIRED EIGENVECTORS IS CONFIRMED.
C
  430 CONTINUE
      DO 440 L=K1,K2
      M=DESC(L)
  440 DESC(M)=0
      DIR(2,2)=DIR(2,1)
C
      WRITE(6,450) ERRMAX
  450 FORMAT(/' CONVERGENCE OBSERVED, ALL RESIDUALS**2 .LT. ERRMAX = ',
     1 E20.12)
C
C
    4 RETURN
C-----END OF LANCI1-----------------------------------------------------
      END
C
C-----ORTHOG------------------------------------------------------------
C     ORTHOGONALIZE COLUMNS M = MA,MB OF Q-ARRAY W.R.T COLUMNS M = 1,MB
C
      SUBROUTINE ORTHOG(MA,MB,N,Q)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  Q(1), S
      DOUBLE PRECISION FINPRO, DSQRT
C-----------------------------------------------------------------------
C     MAIN LOOP
      DO 50 M = MA,MB
      MM0 = (M-1)*N
      LL0 = -N
      DO 40 L = 1,M
      LL0 = LL0 + N
      LL = LL0 + 1
      MM = MM0 + 1
```

```
C-----------------------------------------------------------------------
      S = FINPRO(N,Q(LL),1,Q(MM),1)
C-----------------------------------------------------------------------
C
      IF (M.EQ.L) GO TO 20
C
      MM = MM0
      LL = LL0
      DO 10 I=1,N
      LL = LL + 1
      MM = MM + 1
   10 Q(MM) = Q(MM) - S*Q(LL)
      GO TO 40
C
   20 S = DSQRT(S)
      MM = MM0
      DO 30 I=1,N
      MM = MM + 1
   30 Q(MM) = Q(MM)/S
C
   40 CONTINUE
   50 CONTINUE
C
      RETURN
C-----END OF ORTHOG-----------------------------------------------------
      END
C
C-----START-------------------------------------------------------------
C     GENERATES PSEUDO-RANDOM STARTING VECTORS.
C
      SUBROUTINE START(KA,KB,N,Q,G,ERRMAX)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  Q(1), ERRMAX, S
      REAL  G(1)
      COMMON/RANDOM/IIX
      DOUBLE PRECISION FINPRO, DSQRT
C-----------------------------------------------------------------------
      IF(KA.GT.KB) RETURN
C
      IIL = IIX
      DO 110 K = KA,KB
      KK0 = (K-1)*N
C
C-----------------------------------------------------------------------
      CALL GENRAN(IIL,G,N)
C-----------------------------------------------------------------------
C
      KK = KK0
      DO 10 I = 1,N
      KK = KK + 1
   10 Q(KK) = G(I)
      LL0 = -N
   20 DO 70 L=1,K
      LL0 = LL0 + N
```

```
      LL = LL0 + 1                                                      BLS07590
      KK = KK0 + 1                                                      BLS07600
C-----------------------------------------------------------------------BLS07610
      S = FINPRO(N,Q(LL),1,Q(KK),1)                                     BLS07620
C-----------------------------------------------------------------------BLS07630
C                                                                       BLS07640
      IF (K.EQ.L) GO TO 40                                              BLS07650
C                                                                       BLS07660
      LL = LL0                                                          BLS07670
      KK = KK0                                                          BLS07680
      DO 30 I=1,N                                                       BLS07690
      LL = LL + 1                                                       BLS07700
      KK = KK + 1                                                       BLS07710
   30 Q(KK) = Q(KK) - S*Q(LL)                                           BLS07720
      GO TO 70                                                          BLS07730
C                                                                       BLS07740
   40 S = DSQRT(S)                                                      BLS07750
      IF(S.LE.ERRMAX) GO TO 80                                          BLS07760
      KK = KK0                                                          BLS07770
      DO 50 I=1,N                                                       BLS07780
      KK = KK + 1                                                       BLS07790
   50 Q(KK) = Q(KK)/S                                                   BLS07800
C                                                                       BLS07810
      WRITE(6,60) K                                                     BLS07820
   60 FORMAT(I6,' TH STARTING VECTOR IS GENERATED RANDOMLY')            BLS07830
C                                                                       BLS07840
   70 CONTINUE                                                          BLS07850
      GO TO 110                                                         BLS07860
C                                                                       BLS07870
C-----------------------------------------------------------------------BLS07880
   80 CALL GENRAN(IIX,G,N)                                              BLS07890
C-----------------------------------------------------------------------BLS07900
C                                                                       BLS07910
      WRITE(6,90) K                                                     BLS07920
   90 FORMAT(/I6,' TH RANDOM VECTOR REJECTED, GENERATE ANOTHER'/)       BLS07930
C                                                                       BLS07940
      KK = KK0                                                          BLS07950
      DO 100 I = 1,N                                                    BLS07960
      KK = KK + 1                                                       BLS07970
  100 Q(KK) = G(I)                                                      BLS07980
      GO TO 20                                                          BLS07990
C                                                                       BLS08000
  110 CONTINUE                                                          BLS08010
      RETURN                                                            BLS08020
C-----END OF START------------------------------------------------------BLS08030
      END                                                               BLS08040
C                                                                       BLS08050
C-----START OF DIAGOM---------------------------------------------------BLS08060
C     DIAGOM CALLS THE EISPACK SUBROUTINES TRED2 AND IMTQL2 TO          BLS08070
C     DIAGONALIZE THE SMALL SYMMETRIC MATRICES GENERATED AT EACH        BLS08080
C     ITERATION OF BLOCK LANCZOS.                                       BLS08090
C                                                                       BLS08100
      SUBROUTINE DIAGOM(MXBLK,MM,TM,KACT,N,Q,E,RESID,RESK,RESN,IND,     BLS08110
     1 KACTN,KM,TD,TOD,NITER,IERR,IWRITE)                               BLS08120
C                                                                       BLS08130
```

```
C-----------------------------------------------------------------------
      DOUBLE PRECISION  TM(MXBLK,1),Q(1),E(1),TD(*),TOD(1),RESID(1)
      DOUBLE PRECISION  RESK(1),RESN,RATIO,FRACT,RKM,EMAX,SPREAD,EGAP
      DOUBLE PRECISION  DABS,DFLOAT,DMAX1
C-----------------------------------------------------------------------
      IF(NITER.GE.100) GO TO 270
      RKM = TD(1)
      FRACT = TD(2)
      NSTAG = IERR
      KWANT = KACT
C
C     STORE KM-TH RESIDUALS**2 FOR CHECK ON STAGNATION OF CONVERGENCE
      NITER1 = NITER + 1
      RESK(NITER1) = RKM
      IF(NITER.LE.NSTAG) GO TO 10
C     TEST FOR STAGNATION
      NITERM = NITER - 10
      RATIO = RKM / RESK(NITERM)
      IF(RATIO.GT.FRACT) GO TO 250
C
   10 CONTINUE
C
C     TEST GAPS TO DETERMINE IF SIZE OF 1ST Q-BLOCK CAN BE REDUCED
      IF(NITER.EQ.0) GO TO 40
      IF(KM.EQ.KACT.OR.NITER.LT.10) GO TO 30
      KACT1 = KACT - 1
      DO 20 K = KM,KACT1
      RATIO =  DABS(E(K+1) - E(K))
      IF(RATIO.LT.25*EGAP) GO TO 20
      KACT = K
      GO TO 40
   20 CONTINUE
C
C     IF KACT.NE.KACTN, THEN SUBROUTINE LANCI1 IDENTIFIED A VERY
C     SMALL RESIDUAL FOR SOME E(J), J>= KM.
   30 IF(KACT.EQ.KACTN)  GO TO 50
      RATIO = DABS(E(KACTN+1) - E(KACTN))
      IF(RATIO.LE.EGAP)  GO TO 50
      KACT = KACTN
   40 ICOUNT = 1
      INDEXP = IND
      RESID(1) = RESN
      GO TO 80
C
   50 CONTINUE
      IF(IND.NE.INDEXP) GO TO 70
C     INDEX OF VECTOR OF MAXIMUM NORM IS SAME AS ON PREVIOUS ITERATION
      ICOUNT = ICOUNT + 1
      IF(ICOUNT.LE.5) GO TO 60
      ITEST = ICOUNT - 4
      RATIO = RESID(ITEST)/RESN
      IF(DABS(RATIO).GT.10.D0) GO TO 60
C
C     CONVERGENCE STAGNATED, ADD NEXT RITZ VECTOR IN THE CHAIN
C     TO THE 1ST Q-BLOCK AND RESET THE FLAGS THAT KEEP TRACK OF
```

```
C     CONVERGENCE.                                                      BLS08690
      INDEXP = IND                                                      BLS08700
      ICOUNT = 0                                                        BLS08710
      KACT = KACT + 1                                                   BLS08720
      KWANT = KACT                                                      BLS08730
C     CHECK THAT THERE IS ENOUGH ROOM TO ENLARGE THE 1ST QBLOCK         BLS08740
      IF(2*KACT.GT.MXBLK) GO TO 230                                     BLS08750
      GO TO 80                                                          BLS08760
C                                                                       BLS08770
   60 RESID(ICOUNT) = RESN                                              BLS08780
      INDEXP = IND                                                      BLS08790
      GO TO 80                                                          BLS08800
C                                                                       BLS08810
   70 ICOUNT = 1                                                        BLS08820
      RESID(1) = RESN                                                   BLS08830
      INDEXP = IND                                                      BLS08840
C                                                                       BLS08850
C-----------------------------------------------------------------------BLS08860
C     USE EISPACK SUBROUTINES TO DIAGONALIZE THE SMALL TM-MATRIX.       BLS08870
C                                                                       BLS08880
   80 CALL TRED2(MXBLK,MM,TM,TD,TOD,TM)                                 BLS08890
      CALL IMTQL2(MXBLK,MM,TD,TOD,TM,IERR)                              BLS08900
C-----------------------------------------------------------------------BLS08910
      IF(IERR.EQ.0) GO TO 90                                            BLS08920
      RETURN                                                            BLS08930
C                                                                       BLS08940
C    SELECT RELEVANT EIGENVALUES AND EIGENVECTORS OF THE T-MATRIX.      BLS08950
   90 CONTINUE                                                          BLS08960
C                                                                       BLS08970
C    IMTQL2 RETURNS EIGENVALUES (AND CORRESPONDING EIGENVECTORS) IN     BLS08980
C    ALGEBRAICALLY-ASCENDING ORDER.  REARRANGE TO DESCENDING ORDER.     BLS08990
C                                                                       BLS09000
      DO 100 L=1,MM                                                     BLS09010
      MML = MM-L+1                                                      BLS09020
  100 E(L) = TD(MML)                                                    BLS09030
C                                                                       BLS09040
  110 WRITE(6,120) KACT, (E(J), J=1,KACT)                               BLS09050
  120 FORMAT(' COMPUTED',I4,' ALGEBRAICALLY-LARGEST EIGENVALUES'/(4E20.1BLS09060
     12))                                                               BLS09070
C                                                                       BLS09080
C     COMPUTE ESTIMATE MAXIMUM EIGENVALUE AND OF SPREAD                 BLS09090
      IF(NITER.GT.1) GO TO 140                                          BLS09100
      EMAX = DMAX1(DABS(E(1)),DABS(E(MM)))                              BLS09110
      SPREAD = DABS(E(1) - E(MM))                                       BLS09120
      EGAP = SPREAD/DFLOAT(N)                                           BLS09130
      IF(NITER.EQ.1)  WRITE(6,130) EMAX,SPREAD,EGAP                     BLS09140
  130 FORMAT(/4X,'ESTIMATED NORM OF MATRIX',4X,'ESTIMATED SPREAD',6X,'SPBLS09150
     1READ*(SIZE)*(-1)'/E28.4,E20.4,E24.3)                              BLS09160
  140 CONTINUE                                                          BLS09170
C                                                                       BLS09180
C     COMPUTE RITZ VECTORS                                              BLS09190
      DO 180 I=1,N                                                      BLS09200
      DO 150 KK=1,KWANT                                                 BLS09210
      TOD(KK)=0.D0                                                      BLS09220
      K = MM - KK + 1                                                   BLS09230
```

```
      IL = - N + I
      DO 150 L = 1,MM
      IL = IL + N
  150 TOD(KK) = TOD(KK) + TM(L,K)*Q(IL)
      IKK = -N + I
  160 DO 170 KK=1,KACT
      IKK = IKK + N
  170 Q(IKK)=TOD(KK)
  180 CONTINUE
C
C     ON FILE 13 SAVE ANY EXTRA VECTORS NO LONGER NEEDED IN 1ST Q-BLOCK
      IF(KWANT.EQ.KACT) GO TO 290
      K1 = KACT + 1
      K2 = KWANT
      DUMMY = 100.
      DO 190 K = K1,K2
      LINT = (K-1)*N + 1
      LFIN = K*N
      WRITE(13,210) E(K),DUMMY,K
      WRITE(13,220) (Q(L), L=LINT,LFIN)
  190 CONTINUE
      KDELTA = KWANT - KACT
      WRITE(13,200) KDELTA
  200 FORMAT(/' ABOVE ARE ',I3,' VECTORS STRIPPED FROM A 1ST Q-BLOCK'/
     1' DURING A BLOCK LANZCOS RUN WHICH COULD BE USED AS STARTING VECTO
     1RS'/' IN A LATER RUN IF THE USER DECIDES THAT THESE EIGENVALUES SH
     1OULD'/' BE COMPUTED AFTER ALL.  FORMAT USED IN THE SAME AS WAS USE
     1D'/' IN THE CORRESPONDING BLSTARTV FILE'/)
  210 FORMAT(/E20.12,E13.4,I6,' = EVAL,DUMMY,EVAL NUMBER,EVEC='/)
  220 FORMAT(4E20.12)
      GO TO 290
C
C     DEFAULT, SIZE OF 1ST Q-BLOCK TOO LARGE FOR MXBLK
  230 IWRITE = -1000
      WRITE(6,240) KACT,MXBLK
      WRITE(15,240) KACT,MXBLK
  240 FORMAT(//' BLOCK LANCZOS PROCEDURE TRIED TO INCREASE THE SIZE OF 1
     1ST QBLOCK'/' TO ',I3,' BUT THIS IS NOT FEASIBLE BECAUSE TWICE THIS
     1 SIZE'/' IS G.T. MXBLK WHICH EQUALS ',I4/' USER CAN RERUN PROGRAM
     1WITH LARGER MXBLK'/)
      GO TO 290
C
C     DEFAULT, CONVERGENCE RATE IS TOO SLOW
  250 IWRITE = -1000
      WRITE(6,260) NITER,RATIO,FRACT
      WRITE(15,260) NITER,RATIO,FRACT
  260 FORMAT(//' ON ITERATION ',I3,' CONVERGENCE APPEARS TO BE STAGNATED
     1'/' RATIO OF SQUARE OF CURRENT KM-TH RESIDUAL TO CORRESPONDING SQU
     1ARE'/' 10 ITERATIONS EARLIER IS ',E10.3,' COMPARED TO '/
     1' USER-SPECIFIED RATIO ',E10.3,'. THEREFORE, PROGRAM TERMINATES'/'
     1 USER SHOULD LOOK AT THE OUTPUT. IF CONVERGENCE HAS STAGNATED, USE
     1R'/' CAN EITHER INCREASE KACT OR KMAX OR RESET THE STAGNATION PARA
     1METERS'/' NSTAG AND FRACT, AND RESTART THE BLOCK PROCEDURE USING T
     1HE'/' CURRENT EIGENVECTOR APPROXIMATIONS AS STARTING VECTORS'/)
      GO TO 290
```

```
C
  270 IWRITE = -1000
      WRITE(6,280)
      WRITE(15,280)
  280 FORMAT(//' SOMETHING IS SERIOUSLY WRONG.  NUMBER OF ITERATIONS IS
     1EXCESSIVE'/' PROGRAM TERMINATES FOR USER TO DECIDE WHAT TO DO'/'
     1ALTERNATIVES INCLUDE INCREASING KACT OR KMAX OR BOTH, AND RESTARTI
     1NG'/' USING THE CURRENT APPROXIMATIONS AS STARTING VECTORS'//)
C
  290 CONTINUE
      RETURN
C-----END OF DIAGOM-----------------------------------------------------
      END
C-----LPERM PERMUTES VECTORS------------------------------------------
C
      SUBROUTINE LPERM(W,U,IPERM)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  U(1),W(1)
      INTEGER  IPR(1),IPT(1)
C-----------------------------------------------------------------------
C     SUBROUTINE HAS 2 BRANCHES:  IPERM = 1,  CALCULATES
C     U = P*W  WHERE  P IS THE PERMUTATION REPRESENTED BY IPR
C     LET J = IPR(K) THEN U(K) = W(J), K = 1,N. WE SET W(K)=U(K), K=1,N
C     IPERM = 2, USING THE PERMUTATION IPT (P-TRANSPOSE) U = P'*W, W=U
C     LET J = IPT(K) THEN U(K) = W(J), K=1,N. WE SET W(K) = U(K), K=1,N
C-----------------------------------------------------------------------
C
      GO TO 3
C-----------------------------------------------------------------------
      ENTRY LPERME(IPR,IPT,N)
      GO TO 4
C-----------------------------------------------------------------------
C
    3 CONTINUE
      IF(IPERM.EQ.2) GO TO 10
C     IPERM = 1
      DO 20 K = 1,N
      J = IPR(K)
   20 U(K) = W(J)
      DO 30 K = 1,N
   30 W(K) = U(K)
      GO TO 60
C     IPERM = 2
   10 DO 40  K = 1,N
      J = IPT(K)
   40 U(K) = W(J)
      DO 50 K = 1,N
   50 W(K) = U(K)
   60 CONTINUE
C
C
C-----END OF LPERM--------------------------------------------------
    4 RETURN
      END
```

8.6 BLEVAL: File Definitions, Sample Input File

Below is a listing of the input/output files which are accessed by the real symmetric block Lanczos eigenvalue/eigenvector program, BLEVAL. BLEVAL computes a few extreme eigenvalues and corresponding eigenvectors of a real symmetric matrix A. Also below is a sample of the input file which BLEVAL requires on file 5. The parameters in this file are supplied in free format. File 8 contains data for the nxn real symmetric matrix A.

```
Sample Specifications of Input/Output Files for BLEVAL
----------------------------------------------------------------
 BLEVAL EXEC
FI 06 TERM
FILEDEF  5 DISK BLEVAL    INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1        INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF 10 DISK &1        BLSTARTV  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 13 DISK &1        BLEXTRAV  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 15 DISK &1        BLEIGVEC  A (RECFM F LRECL 80 BLOCK 80
*IMTQL2 AND TRED2 ARE 2 EISPACK LIBRARY SUBROUTINES
LOAD  BLEVAL  BLSUB  BLMULT  IMTQL2  TRED2
------------------------------------------------------------------
```

```
Sample Input File for BLEVAL
----------------------------------------------------------------------
 LINE 1 IWRITE (SPECIFY MESSAGE LEVEL TO FILE 6: 1 MEANS DETAILED
             1
 LINE 2     N      MATNO  (SIZE OF A-MATRIX, MATRIX IDENT. NUMBER
          528        528
 LINE 3 MDIMQ         MDIMTM       MAXIT (DIMS. Q, TM, MAX Ax-mults
        40000           2500        1000
 LINE 4 EFLAG  OFLAG (EFLAG=(0,1) 1=2PHASES. OFLAG: 1=ORTHOG CHECK
            1      1
 LINE 5 SEED     (STARTING VECTOR SEED, RANDOM NUMBER GENERATOR
     3482736
 LINE 6 KMAX  KACT  KSET (MAX T SIZE +1,SIZE 1ST BLOCK,VECS SUPPLIED
          21     4     0
 LINE 7 KM   (NUMB. EVS FOR ALG-LARGEST, -(NUMB. EVS) FOR ALG-SMALLEST
         4
 LINE 8 NSTAG  FRACT (NO. ITNS BEFORE TEST CONVERGENCE, TEST FRACTION
           25    .01
 LINE 9 RELTOL    MAXIT2 (PHASE 2, CONVERGE. TOL. , Max Ax-mults
     .00000001      1000
----------------------------------------------------------------------
```

Chapter 9

Factored Inverses, Real Symmetric Block Lanczos Code

9.1 Introduction

The FORTRAN codes in this chapter address the question of using an iterative block Lanczos procedure to compute a 'few' eigenvalues and a basis for the corresponding eigenspace of a real symmetric matrix A by computing a few extreme eigenvalues and a corresponding basis for the inverse of a real symmetric matrix B obtained from A by scaling, shifting and permuting A. For a given real symmetric matrix A, the codes consider the inverse of a matrix B where

$$B \equiv PCP^T, \quad C \equiv (SCALE) * A + (SHIFT) * I, \tag{9.1.1}$$

$SCALE$ and $SHIFT$ are specified by the user, and the permutation matrix P is chosen so that for a sparse matrix A (or C), the resulting factorization of the associated B matrix is also sparse. An eigenvalue is 'extreme' if it is one of the algebraically-smallest or the algebraically-largest eigenvalues in the eigenvalue spectrum.

Specifically, for a given real symmetric matrix A and associated B-matrix as defined in Eqn(9.1.1), the codes in this chapter compute the q algebraically-largest eigenvalues, λ_i , $1 \leq i \leq q$, of B^{-1} and corresponding orthonormal real vectors $X_q \equiv (x_1, \ldots, x_q)$ such that

$$B^{-1}X_q = X_q A_q, \quad A_q \equiv X_q^T A X_q. \tag{9.1.2}$$

Typically, $A_q = \Lambda_q$, a diagonal matrix whose nonzero entries are the eigenvalues λ_i . The number q is small and specified by the user.

Real symmetric matrices and factorizations of real symmetric matrices are discussed in Stewart [24]. See also Bunch and Kaufman [2] and George and Liu [10]. Chapter 2, Section 2.1 contains a brief summary of the properties of real symmetric matrices which we use in these codes.

The Lanczos code contained in this chapter is a simple modification of the hybrid 'block' Lanczos procedure given in Chapter 8 to handle the factored inverse of the B-matrix given in Eqn(9.1.1). Therefore please see Chapter 8, Section 8.1, for comments about this procedure and for comments regarding the differences between iterative block Lanczos procedures and single-vector Lanczos procedures.

BLIEVAL is the main 'block' program for the factored inverse version of the 'block' Lanczos codes in

Chapter 8. BLIEVAL uses the same subroutines as the real symmetric codes in Chapter 8, with the exception of the user-supplied subroutines. The user must supply a subroutine USPEC which defines and initializes the matrix which is to be used by the LANCZS and LANCI1 subroutines. In the factored inverse case, USPEC specifies the factorization of the particular B-matrix being used. These Lanczos programs do not require the A-matrix. However, the user must supply the scalars $SCALE$ and $SHIFT$, and the permutation P (if any). The user must also supply a subroutine BLSOLV which solves the system of equations $Bu = x$ for any given vector x.

The sample USPEC and BLSOLV subroutines provided assume that the B-matrix being used is positive definite and that the Cholesky factors of B,

$$B = LL^T \tag{9.1.3}$$

where L is a lower triangular matrix, are used for the matrix-vector multiply, $B^{-1}x$, for any given vector x. However, the user may replace these subroutines by subroutines which define and use a more general factorization. These Lanczos codes only require that the BLSOLV subroutine solves the system $Bu = x$, rapidly and accurately.

All computations are in double precision real arithmetic. On each iteration, the accuracy of the computed eigenvectors is checked in the process of computing the second block of Lanczos vectors on that iteration. Note that the eigenvectors of B^{-1} are simple permutations of the eigenvectors of A. These permutations are undone prior to the termination of the block procedure. The corresponding eigenvalues of A are obtained from those of B^{-1} by a simple scalar transformation which is included in the codes. The eigenelement computations for the small Lanczos matrices use two subroutines from the EISPACK Library [23, 8], TRED2 and IMTQL2.

Several optional preprocessing programs are provided, PERMUT, LORDER, LFACT, and LTEST. Listings for these programs are given in Chapter 4. PERMUT calls the SPARSPAK Library [23, 8] to attempt to identify a reordering or permutation P of a given matrix A for which sparseness is preserved under factorization of the permuted matrix. LORDER takes a given matrix C and permutation P and computes the sparse matrix format for the permuted matrix, $B \equiv PCP^T$. LFACT computes the Cholesky factors of a given positive definite matrix. LTEST performs a very crude check on the numerical condition of the matrix supplied to it, by solving a system of equations with and without iterative refinement, LINPACK [7].

The usefulness of this code for computing a few interior eigenvalues of a given real symmetric matrix is dubious. For such an application one would have to select a shift $SHIFT$ that places the desired eigenvalues of the A-matrix on the extreme of the spectrum of the associated matrix B^{-1} and is chosen so that the B-matrix is well-conditioned numerically. This is not a trivial task. The user should refer to Chapter 7 of Volume 1 of this book for more details on iterative block Lanczos procedures.

9.2 BLIEVAL: Main Program, Eigenvalue and Eigenvector Computations

```
C-----BLIEVAL (FEW EXTREME EIGENVALUES AND EIGENVECTORS)----------------BLI00010
C           (USING FACTORED INVERSE OF A REAL SYMMETRIC MATRIX)       BLI00020
C  Authors:  Jane Cullum* and Bill Donath**                            BLI00025
C        **IBM Research, T.J. Watson Research Center                  BLI00030
C        **Yorktown Heights, N.Y. 10598                               BLI00040
C        * Los Alamos National Laboratory                             BLI00050
C        * Los Alamos, New Mexico 87544                               BLI00060
C          E-mail:  cullumj@lanl.gov                                  BLI00070
C                                                                     BLI00080
C  These codes are copyrighted by the authors.  These codes           BLI00090
C  and modifications of them or portions of them are NOT to be        BLI00100
C  incorporated into any commercial codes or used for any other       BLI00110
C  commercial purposes such as consulting for other companies,        BLI00120
C  without legal agreements with the authors of these Codes.          BLI00130
C  If these Codes or portions of them are used in other scientific or BLI00140
C  engineering research works the names of the authors of these codes BLI00150
C  and appropriate references to their written work are to be         BLI00160
C  incorporated in the derivative works.                              BLI00170
C                                                                     BLI00180
C  This header is not to be removed from these codes.                 BLI00190
C                                                                     BLI00195
C       REFERENCE: Cullum and Willoughby, Chapter 7,                  BLI00200
C       Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsBLI00205
C       VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   BLI00210
C       Applied Mathematics, 2002. SIAM Publications,                 BLI00215
C       Philadelphia, PA. USA                                         BLI00220
C                                                                     BLI00225
C    CONTAINS MAIN PROGRAM FOR COMPUTING A FEW EIGENVALUES            BLI00230
C    AND CORRESPONDING EIGENVECTORS OF A REAL SYMMETRIC MATRIX        BLI00235
C    BY COMPUTING A FEW OF THE ALGEBRAICALLY-LARGEST OR               BLI00240
C    ALGEBRAICALLY-SMALLEST EIGENVALUES OF THE INVERSE OF A SCALED,   BLI00250
C    SHIFTED, AND PERMUTED VERSION B OF THE ORIGINAL A-MATRIX         BLI00260
C    USING A BLOCK FORM OF LANCZOS TRIDIAGONALIZATION WITH LIMITED    BLI00270
C    REORTHOGONALIZATION.  THIS BLOCK PROCEDURE IS ITERATIVE AND      BLI00280
C    REQUIRES A SUBROUTINE BLSOLV THAT FOR ANY GIVEN VECTOR W         BLI00290
C    COMPUTES U SUCH THAT B*U = W.   THE SAMPLE BLSOLV SUBROUTINES    BLI00300
C    PROVIDED FOR SPARSE MATRICES ARE ONLY FOR THE CASE THAT B IS     BLI00310
C    POSITIVE DEFINITE AND USE THE CHOLESKY FACTORS OF B.  HOWEVER,   BLI00320
C    THE USER COULD REPLACE THESE BY A SUBROUTINE WHICH COMPUTES      BLI00330
C    FOR AN INDEFINITE MATRIX THE FACTORIZATION L*D*(L-TRANSPOSE).    BLI00340
C                                                                     BLI00350
C    THIS BLOCK PROCEDURE COMPUTES THE ALGEBRAICALLY-LARGEST          BLI00360
C    EIGENVALUES OF THE INVERSE OF THE B-MATRIX, UNLESS THE USER      BLI00370
C    SUPPLIES -(B-INVERSE)*X RATHER THAN (B-INVERSE)*X, IN WHICH      BLI00380
C    CASE IT COMPUTES THE CORRESPONDING ALGEBRAICALLY-SMALLEST        BLI00390
C    EIGENVALUES OF (B-INVERSE) BY COMPUTING THE ALGEBRAICALLY-       BLI00400
C    LARGEST EIGENVALUES OF -(B-INVERSE).  IN THIS CASE THE SIGNS     BLI00410
C    OF THE COMPUTED EIGENVALUES ARE CHANGED PRIOR TO WRITING TO      BLI00420
C    FILE 15 SO THAT ON EXIT, FILE 15 CONTAINS THE ALGEBRAICALLY-     BLI00430
C    SMALLEST EIGENVALUES OF B-INVERSE ALONG WITH THE CORRESPONDING   BLI00440
```

```
C     EIGENVALUES OF THE ORIGINAL A-MATRIX AND CORRESPONDING            BLI00450
C     EIGENVECTORS.  THE MATRIX B = S0*P*A*P' + SHIFT*I WHERE THE       BLI00460
C     SCALE S0 AND SHIFT ARE READ IN THIS PROGRAM, AND THE              BLI00470
C     PERMUTATION P IS DEFINED IN THE CORRESPONDING USPEC SUBROUTINE.   BLI00480
C     THE PROGRAM ASSUMES THAT THE FACTORIZATION READ IN USPEC          BLI00490
C     CORRESPONDS TO THE S0, SHIFT AND PERMUTATION READ IN.  THE S0     BLI00500
C     AND SHIFT ARE CHOSEN SO THAT THE DESIRED EIGENVALUES ARE AT       BLI00510
C     THE EXTREME OF THE SPECTRUM OF B-INVERSE.                         BLI00520
C                                                                       BLI00530
C     THIS IS AN ITERATIVE 'BLOCK' LANCZOS PROCEDURE FOR WHICH ON       BLI00540
C     EVERY ITERATION, THE 2ND AND SUCCEEDING BLOCKS CONTAIN ONLY ONE   BLI00550
C     VECTOR WHICH IS SELECTED ON THE BASIS OF ITS EXPECTED INFLUENCE   BLI00560
C     ON THE CONVERGENCE.   Q-BLOCKS GENERATED ON A GIVEN ITERATION    BLI00570
C     ARE REORTHOGONALIZED ONLY W.R.T. THOSE VECTORS IN THE FIRST       BLI00580
C     Q-BLOCK WHICH ARE NOT ALLOWED TO GENERATE DESCENDANTS ON          BLI00590
C     THAT ITERATION.                                                   BLI00600
C                                                                       BLI00610
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE CONSTRUCTIONS:BLI00620
C     1.  DATA MACHEP DEFINITION                                        BLI00630
C     2.  FORMAT (20A4) USED FOR READING EXPLANATORY COMMENTS.          BLI00640
C     3.  FREE FORMAT (5,*), USED FOR PARAMETER INPUT FROM FILE 5.      BLI00650
C     4.  COMMON/LOOPS/ AS CONSTRUCTED IS NOT PORTABLE                  BLI00660
C                                                                       BLI00670
C-----------------------------------------------------------------------BLI00680
      DOUBLE PRECISION Q(44000),E(50),TM(2500),TOD(50),TD(50),EPSM,NNZ  BLI00690
      DOUBLE PRECISION SM(100),ERRMAX,SPREC,MACHEP,AVER,RELTOL,ERRMAN   BLI00700
      DOUBLE PRECISION  EVAL, RESIDL(100), RESIDK(100), RESID, FRACT    BLI00710
      DOUBLE PRECISION S0,SHIFT                                         BLI00720
      REAL EXPLAN(20),G(2000)                                           BLI00730
      INTEGER   DIR(2,100),DESC(100),LEFT(100),XLFT(100)                BLI00740
      INTEGER SEED,OFLAG,EFLAG                                          BLI00750
      COMMON/LOOPS/MAXIT,ITER                                           BLI00760
      COMMON /RANDOM/SEED                                               BLI00770
      COMMON/FLAGS/EFLAG,OFLAG                                          BLI00780
      DOUBLE PRECISION  DABS, DFLOAT                                    BLI00790
C-----------------------------------------------------------------------BLI00800
      EXTERNAL BLSOLV                                                   BLI00810
      DATA MACHEP/Z3410000000000000/                                    BLI00820
C-----------------------------------------------------------------------BLI00830
C                                                                       BLI00840
C     ARRAYS MUST DIMENSIONED AS FOLLOWS:                               BLI00850
C                                                                       BLI00860
C     1.  Q:   >=  KMAX*N                                               BLI00870
C     2.  G:   >= N                                                     BLI00880
C     3.  E:   >=  MXBLK                                                BLI00890
C     4.  TM:   >=  MXBLK**2                                            BLI00900
C     5.  TOD, TD, SM, DESC, LEFT, XLFT:   >= MXBLK                     BLI00910
C     6.  DIR:  ROW DIMENSION = 2;   COLUMN DIMENSION >= MXBLK          BLI00920
C     7.  RESIDL, RESIDK:  >= MAXIMUM NUMBER OF ITERATIONS ALLOWED.     BLI00930
C         PROGRAM CURRENTLY TERMINATES IF MORE THAN 100 ITERATIONS      BLI00940
C         ARE REQUESTED.  USED TO MONITOR CONVERGENCE. SEE SUBROUTINE   BLI00950
C         DIAGOM.                                                       BLI00960
C     8.  EXPLAN:  DIMENSION = 20.                                      BLI00970
C                                                                       BLI00980
C-----------------------------------------------------------------------BLI00990
```

```
C     OUTPUT HEADER                                                     BLI01000
      WRITE(6,10)                                                       BLI01010
   10 FORMAT(/' BLOCK LANCZOS PROCEDURE, USES FACTORED INVERSE OF A USERBLI01020
     1-SPECIFIED MATRIX'/' 2ND AND SUCCEEDING BLOCKS GENERATED ON EACH BBLI01030
     1LOCK ITERATION '/' CONTAIN ONLY ONE VECTOR'//)                    BLI01040
C                                                                       BLI01050
C     SET PROGRAM PARAMETERS                                            BLI01060
      EPSM = 2.D0*MACHEP                                                BLI01070
      SPREC = 1.D-5                                                     BLI01080
      MPMIN = -1000                                                     BLI01090
C                                                                       BLI01100
C     READ USER-SPECIFIED PARAMETERS FROM INPUT FILE 5 (FREE FORMAT)    BLI01110
C                                                                       BLI01120
C     SELECT THE AMOUNT OF INTERMEDIATE OUTPUT DESIRED (IWRITE =0,1).   BLI01130
C     IWRITE = 1 INCREASES THE AMOUNT OF INTERMEDIATE OUTPUT WRITTEN    BLI01140
C     TO FILE 6 ON EACH ITERATION OF THE BLOCK LANCZOS PROCEDURE.       BLI01150
      READ(5,20) EXPLAN                                                 BLI01160
   20 FORMAT(20A4)                                                      BLI01170
      READ(5,*)  IWRITE                                                 BLI01180
C                                                                       BLI01190
C     READ ORDER (N) OF MATRIX AND MATRIX IDENTIFICATION NUMBER (MATNO) BLI01200
C     READ SCALE (S0) AND SHIFT (SHIFT) APPLIED TO MATRIX AND           BLI01210
C     FLAG JPERM.  JPERM = (0,1):  JPERM = 1 MEANS MATRIX HAS BEEN      BLI01220
C     PERMUTED                                                          BLI01230
      READ(5,20) EXPLAN                                                 BLI01240
      READ(5,*) N,MATNO,S0,SHIFT,JPERM                                  BLI01250
C                                                                       BLI01260
C     READ USER-SPECIFIED DIMENSIONS OF Q-ARRAY (MDIMQ) AND OF THE      BLI01270
C     TM-ARRAY (MDIMTM).  READ MAXIMUM NUMBER (MAXIT) OF CALLS TO THE   BLI01280
C     BLSOLV SUBROUTINE ALLOWED IN PHASE 1.                             BLI01290
      READ(5,20) EXPLAN                                                 BLI01300
      READ(5,*) MDIMQ, MDIMTM, MAXIT                                    BLI01310
C                                                                       BLI01320
C     READ FLAGS:   EFLAG = (0,1).  EFLAG = 0, MEANS PROGRAM STOPS      BLI01330
C     AFTER COMPLETING PHASE 1 PORTION OF BLOCK LANCZOS PROCEDURE.      BLI01340
C     EFLAG = 1, MEANS PROGRAM COMPLETES BOTH PHASES BEFORE             BLI01350
C     TERMINATING.                                                      BLI01360
C     OFLAG = (0,1).  OFLAG = 0, MEANS THAT IN PHASE 1 PORTION          BLI01370
C     OF THE COMPUTATION, THE PROGRAM DOES NO ORTHOGONALITY CHECKS      BLI01380
C     ON THE Q-BLOCKS GENERATED.  OFLAG = 1 MEANS THAT IN THE           BLI01390
C     PHASE 1 PORTION AND IN THE PHASE 2 PORTIONS OF THE COMPUTATIONS   BLI01400
C     THE PROGRAM CHECKS THE ORTHOGONALITY OF THE Q-BLOCKS GENERATED    BLI01410
C     W.R.T. THAT VECTOR IN THE FIRST BLOCK THAT IS GENERATING          BLI01420
C     DESCENDANTS.  NOTE THAT IN PHASE 2, THE PROGRAM ALWAYS MAKES      BLI01430
C     THIS CHECK OF ORTHGONALITY REGARDLESS OF THE VALUE OF OFLAG.      BLI01440
C     FOR SAFETY, OFLAG SHOULD ALWAYS BE SET TO 1, ALTHOUGH FOR MANY    BLI01450
C     PROBLEMS THIS IS NOT NECESSARY.                                   BLI01460
      READ(5,20) EXPLAN                                                 BLI01470
      READ(5,*) EFLAG,OFLAG                                             BLI01480
C                                                                       BLI01490
C     READ SEED USED BY SUBROUTINE GENRAN TO OBTAIN THOSE STARTING      BLI01500
C     VECTORS WHICH ARE GENERATED RANDOMLY.                             BLI01510
      READ(5,20) EXPLAN                                                 BLI01520
      READ(5,*) SEED                                                    BLI01530
C                                                                       BLI01540
```

```
C     SPECIFY MAXIMUM T-SIZE ALLOWED (KMAX-1); INITIAL SIZE OF
C     STARTING BLOCK (KACT);  NUMBER OF STARTING VECTORS SUPPLIED (KSET)
C     SEE BLOCK LANCZOS HEADER FOR COMMENTS REGARDING THE SIZE OF KACT.
      READ(5,20) EXPLAN
      READ(5,*) KMAX,KACT,KSET
C
C     SPECIFY NUMBER (KM) OF EXTREME EIGENVALUES AND EIGENVECTORS
C     OF B-INVERSE TO BE COMPUTED.  THE BLOCK PROCEDURE WORKS WITH THE
C     INVERSE OF THE MATRIX B = S0*P*A*P' + SHIFT*I, USING A
C     FACTORIZATION OF B.  TO INDICATE THAT THE ALGEBRAICALLY-
C     SMALLEST EIGENVALUES OF B-INVERSE ARE BEING COMPUTED SET KM < 0.
C     IF KM < 0, THE PROGRAM ASSUMES THAT BLSOLV SUBROUTINE WHICH
C     THE USER HAS PROVIDED IS COMPUTING -(B-INVERSE)*X
C     INSTEAD OF (B-INVERSE)*X AND INTERNALLY IT COMPUTES THE |KM|
C     ALGEBRAICALLY-LARGEST EIGENVALUES OF -(B-INVERSE).
      READ(5,20) EXPLAN
      READ(5,*) KM
      IF(KM.EQ.0) GO TO 540
      KML = IABS(KM)
C
C     STAGNATION OF CONVERGENCE OF THE KM-TH EIGENVALUE WILL BE
C     TESTED AFTER NSTAG ITERATIONS.  CONVERGENCE WILL BE SAID TO
C     HAVE STAGNATED IF THE RATIO OF THE SQUARE OF THE CURRENT KM-TH
C     RESIDUAL TO THE SQUARE OF THE CORRESPONDING RESIDUAL OBTAINED
C     10 ITERATIONS EARLIER IS GREATER THAN FRACT.  NSTAG SHOULD BE
C     >= 25.  FRACT WAS SET EQUAL TO .01 IN THE TESTS.
      READ(5,20) EXPLAN
      READ(5,*) NSTAG, FRACT
C
C     READ IN THE RELATIVE TOLERANCE (RELTOL) USED TO DETERMINE A
C     CONVERGENCE CRITERION FOR PHASE 2, AND THE MAXIMUM NUMBER (MAXIT2)
C     OF CALLS TO SUBROUTINE BLSOLV ALLOWED IN PHASE 2.
      READ(5,20) EXPLAN
      IF(EFLAG.EQ.1)  READ(5,*)  RELTOL, MAXIT2
C
C     CONSISTENCY CHECKS
C     PROCEDURE REQUIRES ENOUGH ROOM IN THE Q-ARRAY FOR AT LEAST 2
C     BLOCKS OF SIZE KACT PLUS A WORKING VECTOR OF LENGTH N.
      MXBLK = KMAX -1
      MXBLK2 = MXBLK*MXBLK
      IF(MDIMTM.LT.MXBLK2) GO TO 520
      NKMAX = N*KMAX
      IF(MDIMQ.LT.NKMAX)   GO TO 560
      IF(KML.GT.KACT) GO TO 420
      IF(MXBLK.GT.N)  GO TO 440
      IF(2*KACT.GT.MXBLK)  GO TO 500
C
C-----------------------------------------------------------------------
C     DEFINE AND INITIALIZE THE ARRAYS NEEDED TO DEFINE THE
C     FACTORIZATION OF THE B-MATRIX.  PASS THE STORAGE LOCATIONS
C     OF THESE ARRAYS TO THE SUBROUTINE BLSOLV.
C
      CALL USPEC(N,MATNO,NNZ,AVER)
C
C-----------------------------------------------------------------------
```

```
C     MASK OVERFLOW AND UNDERFLOW                                       BLI02100
      CALL MASK                                                         BLI02110
C                                                                       BLI02120
C-----------------------------------------------------------------------BLI02130
C     ARE THERE STARTING VECTORS TO READ IN FROM FILE 10 (KSET.NE.0) ?  BLI02140
      IF(KSET.EQ.0) GO TO 70                                            BLI02150
C                                                                       BLI02160
      READ(10,30) NOLD,KACT                                             BLI02170
   30 FORMAT(I6,I4)                                                     BLI02180
      IF(NOLD.NE.N.OR.KSET.GT.KACT) GO TO 460                           BLI02190
      DO 50 J=1,KSET                                                    BLI02200
      READ(10,20) EXPLAN                                                BLI02210
      READ(10,40) EVAL,RESID                                            BLI02220
   40 FORMAT(E20.12,E13.4)                                              BLI02230
      READ(10,20) EXPLAN                                                BLI02240
      LINT= (J-1)*N + 1                                                 BLI02250
      LFIN = J*N                                                        BLI02260
   50 READ(10,60) (Q(JL), JL = LINT,LFIN)                               BLI02270
   60 FORMAT(4E20.12)                                                   BLI02280
C                                                                       BLI02290
   70 CONTINUE                                                          BLI02300
C                                                                       BLI02310
C     WRITE TO A SUMMARY OF THE PARAMETERS FOR THIS RUN TO FILE 6       BLI02320
C                                                                       BLI02330
      MXBLK = KMAX - 1                                                  BLI02340
      WRITE(6,80)  N, NNZ, AVER, MATNO                                  BLI02350
   80 FORMAT(/4X,'ORDER OF B-MATRIX ',5X,'AVERAGE NUMBER NONZEROES PER RBLI02360
     1OW IN FACTOR'/                                                    BLI02370
     1I15,E47.4/3X,'CRUDE ESTIMATE OF SIZE NONZERO ENTRIES',5X,'MATRIX IBLI02380
     1D'/E31.4,I21/)                                                    BLI02390
C                                                                       BLI02400
      WRITE(6,90)  S0, SHIFT                                            BLI02410
   90 FORMAT(/4X,'SCALE USED ON A-MATRIX',5X,'SHIFT USED ON A-MATRIX'/  BLI02420
     1E26.4,E27.4/)                                                     BLI02430
C                                                                       BLI02440
      WRITE(6,100) MDIMQ, MDIMTM                                        BLI02450
  100 FORMAT(/18X,'USER-SPECIFIED'/2X,'MAX. DIMENSION Q-ARRAY',4X,'MAX. BLI02460
     1DIMENSION TM-ARRAY'/I16,I26/)                                     BLI02470
C                                                                       BLI02480
      WRITE(6,110)  OFLAG, EFLAG                                        BLI02490
  110 FORMAT(/4X,'OFLAG',4X,'EFLAG'/I8,I9/)                             BLI02500
C                                                                       BLI02510
      IF(OFLAG.EQ.1)  WRITE(6,120)  SPREC                               BLI02520
  120 FORMAT(/4X,'ORTHOGONALITY TEST TOLERANCE'/E25.2)                  BLI02530
C                                                                       BLI02540
      IF(EFLAG.EQ.1)  WRITE(6,130) MAXIT,RELTOL,MAXIT2                  BLI02550
  130 FORMAT(/4X,' MAXIT ',8X,'  RELTOL  ',6X,' MAXIT2 '/I10,E20.6,I12/)BLI02560
      IF(EFLAG.EQ.0)  WRITE(6,140) MAXIT                                BLI02570
  140 FORMAT(/4X,' MAXIT '/I10/)                                        BLI02580
C                                                                       BLI02590
      WRITE(6,150)  SEED                                                BLI02600
  150 FORMAT(/' SEED FOR RANDOM NUMBER GENERATOR'/I24/)                 BLI02610
C                                                                       BLI02620
      IF(KM.GT.0)  WRITE(6,160) KML                                     BLI02630
  160 FORMAT(/' COMPUTE THE',I3,' ALGEBRAICALLY-LARGEST EIGENVALUES AND BLI02640
```

```
     1CORRESPONDING VECTORS'/' OF THE INVERSE OF B =  (S0*P*A*P-TRANS +
     1HIFT*I)'/)
      IF(KM.LT.0)  WRITE(6,170) KML
  170 FORMAT(/' COMPUTE THE',I3,' ALGEBRAICALLY-SMALLEST EIGENVALUES AND
     1 CORRESPONDING VECTORS'/' OF THE INVERSE OF THE MATRIX B = (S0*P*A
     1*P-TRANS + SHIFT*I).'/' PROGRAM ASSUMES THAT USER IS PROVIDING -(B
     1-INVERSE)*X INSTEAD OF (B-INVERSE)*X'/' AND COMPUTES THE ALGEBRAIC
     1ALLY-LARGEST EIGENVALUES OF -(B-INVERSE).'/' HOWEVER ON EXIT, FILE
     1 15 CONTAINS THE ALGEBRAICALLY-SMALLEST EIGENVALUES'/' OF B-INVERS
     1E, THE CORRESPONDING EIGENVALUES OF THE ORIGINAL A-MATRIX'/' AND T
     1HE CORRESPONDING EIGENVECTORS OF A.'/)
C
C     NOTE THAT THE ESTIMATE FOR AVER IN THE INVERSE CASE IS VERY CRUDE
C     COMPUTE PHASE 1 CONVERGENCE TOLERANCE
      IF(AVER.GE.1.)
     1ERRMAX = 2.D0*DFLOAT(N+1000)*NNZ*AVER*MACHEP
      IF(AVER.LT.1.)
     1ERRMAX = 2.D0*DFLOAT(N+1000)*NNZ*AVER**2*MACHEP
C
      WRITE(6,180) KACT,MXBLK,KSET
  180 FORMAT(/' ON INITIAL ITERATIONS, THE FIRST BLOCK CONTAINS ',I3,' V
     1ECTORS'/' HOWEVER THE SIZE OF THE FIRST BLOCK MAY CHANGE AS THE IT
     1ERATIONS PROCEED'/' THE MAXIMUM SIZE T-MATRIX THAT CAN BE GENERATE
     1D IS ',I4/' THE USER SUPPLIED ',I3,' STARTING VECTORS'/)
C
      WRITE(6,190)
  190 FORMAT(/' ITERATIVE PROCEDURE'/' PROCEDURE MONITORS THE SIZES OF T
     1HE NORM(GRADIENTS)**2 ON EACH'/' ITERATION.  CONVERGENCE IS SAID
     1TO HAVE OCCURRED WHEN ALL'/' RELEVANT (NORMS)**2 ARE LESS THAN ERR
     1MAX',E10.3/' PHASE 1 ERRMAX MAY YIELD SOMEWHAT LESS THAN SINGLE PR
     1ECISION ACCURACY.'/'  PHASE 2 REFINES THE VECTORS OBTAINED ON PHAS
     1E 1, ACCORDING TO'/' THE ACCURACY SPECIFIED BY THE USER'/)
C
      WRITE(6,200) ERRMAX
  200 FORMAT(//' PHASE 1 CONVERGENCE CRITERION, ERRMAX '/E22.3/)
C
C-----------------------------------------------------------------------
C     PASS STORAGE LOCATIONS OF VARIOUS ARRAYS TO LANCZS AND LANCI1
C     SUBROUTINES
C
      CALL LANZP(DIR,DESC,SM,TM,TOD,TD,G,XLFT,LEFT,SPREC)
      CALL LANCP1(DIR,DESC,TM,SM,XLFT,LEFT)
C
C-----------------------------------------------------------------------
C
C     ENTER PHASE 1 OF BLOCK LANCZOS PROCEDURE.  BLOCK PROCEDURE
C     HAS 2 POSSIBLE PHASES.  USER SPECIFIES PHASE 1 ONLY OR PHASE 1
C     AND PHASE 2 BY SETTING EFLAG = 0 OR 1, RESPECTIVELY. PHASE 1
C     COMPUTES VECTORS THAT ARE USUALLY ACCURATE TO SINGLE PRECISION.
C     PHASE 2 TAKES THE VECTORS OBTAINED IN PHASE 1 AND REFINES THEM.
C     THE USER SPECIFIES THE DEGREE OF REFINEMENT DESIRED BY SELECTING
C     THE VALUE OF RELTOL AND MAXIT2.  BOTH PHASES SHOULD BE USED.
      IPHASE = 1
      NITER = 0
  210 ITER = 0
```

```
      RESIDL(1) = FRACT                                                 BLI03200
      RESIDL(2) = NSTAG                                                 BLI03210
C                                                                       BLI03220
C-----------------------------------------------------------------------BLI03230
C     CALL INITIATES THE BLOCK LANCZOS PROCEDURE.                       BLI03240
C     ON RETURN EIGENVALUE APPROXIMATIONS ARE IN E(I),                  BLI03250
C     I = 1,KACT, IN ALGEBRAICALLY DECREASING ORDER.   CORRESPONDING    BLI03260
C     EIGENVECTOR APPROXIMATIONS ARE IN FIRST N*KACT LOCATIONS IN       BLI03270
C     THE Q-ARRAY.                                                      BLI03280
C                                                                       BLI03290
      CALL LANCZS(BLSOLV,KML,KSET,KACT,MXBLK,N,Q,E,RESIDL,RESIDK,ERRMAX,BLI03300
     1 IPHASE,NITER,IWRITE)                                             BLI03310
C                                                                       BLI03320
C-----------------------------------------------------------------------BLI03330
C                                                                       BLI03340
      IF(IPHASE.EQ.MPMIN) WRITE(15,220) N,KACT                          BLI03350
  220 FORMAT(2I10,' PHASE 2 TERMINATED '/' PROGRAM INDICATES ACCURACY SPBLI03360
     1ECIFIED BY USER IS NOT ACHIEVABLE'/)                              BLI03370
C                                                                       BLI03380
      ITERA = IABS(ITER)                                                BLI03390
      IF(IWRITE.NE.MPMIN.AND.ITER.GT.0)  WRITE(6,230) IPHASE,ITERA      BLI03400
  230 FORMAT(/1X,'PHASE COMPLETED',5X,' NUMBER CALLS TO BLSOLV SUBROUTINBLI03410
     1E USED'/I10,I32)                                                  BLI03420
C                                                                       BLI03430
      IF(IWRITE.EQ.MPMIN.OR.ITER.LT.0)  WRITE(6,240) IPHASE,ITERA       BLI03440
  240 FORMAT(/1X,'PHASE TERMINATED',5X,' NUMBER CALLS TO BLSOLV SUBROUTIBLI03450
     1NE USED'/I10,I32)                                                 BLI03460
C                                                                       BLI03470
      IF(ITER.GT.0.AND.IWRITE.NE.MPMIN)  GO TO 270                      BLI03480
C                                                                       BLI03490
      IF(ITER.LT.0)  WRITE(6,250)                                       BLI03500
  250 FORMAT(//' SMALL EIGENVALUE SUBROUTINE DEFAULTED'/' BLOCK LANCZOS BLI03510
     1 PROCEDURE STOPS AFTER SAVING CURRENT EIGENVECTOR APPROXIMATIONS'/BLI03520
     1/)                                                                BLI03530
C                                                                       BLI03540
      WRITE(15,260)                                                     BLI03550
      WRITE(6,260)                                                      BLI03560
  260 FORMAT(//' BLOCK LANCZOS PROCEDURE TERMINATES WITHOUT CONVERGENCE BLI03570
     1'/' USER SHOULD EXAMINE OUTPUT TO DETERMINE REASONS FOR TERMINATIOBLI03580
     1N'//)                                                             BLI03590
C                                                                       BLI03600
C     WRITE EIGENVALUE AND EIGENVECTOR APPROXIMATIONS CONTAINED IN      BLI03610
C     THE FIRST Q-BLOCK TO FILE 15                                      BLI03620
C                                                                       BLI03630
  270 IF(IPHASE.EQ.1)  WRITE(15,280) N,KACT,SEED                        BLI03640
  280 FORMAT(I6,I4,I12,' PHASE 1, ORDER A-MATRIX, SIZE OF Q(1), SEED')  BLI03650
      IF(IPHASE.EQ.2)  WRITE(15,290) N,KACT,SEED                        BLI03660
  290 FORMAT(I6,I4,I12,' PHASE 2, ORDER A-MATRIX, SIZE OF Q(1), SEED')  BLI03670
C                                                                       BLI03680
C     PERMUTE THE EIGENVECTORS IF NECESSARY                             BLI03690
      IF(JPERM.EQ.0) GO TO 310                                          BLI03700
      LINT = -N + 1                                                     BLI03710
      KACT1 = KACT*N + 1                                                BLI03720
      DO 300 J = 1,KACT                                                 BLI03730
      LINT = LINT + N                                                   BLI03740
```

```
C-----------------------------------------------------------------------
      IPERM = 2
      CALL LPERM(Q(LINT),Q(KACT1),IPERM)
C-----------------------------------------------------------------------
  300 CONTINUE
C
C     COMPUTE THE EIGENVALUES OF THE A-MATRIX
  310 DO 320 J = 1,KACT
      IF(KM.LT.0)   E(J) = -E(J)
      TD(J) = 1.D0/E(J)
  320 TD(J) = (TD(J) - SHIFT)/S0
C
C     NOTE THAT RESIDUAL PRINTED OUT CORRESPONDS TO VALUE OBTAINED
C     PRIOR TO FINAL PROJECTION Q(1)-TRANSPOSE*AQ(1) DONE BEFORE
C     TERMINATION
      JJ=KACT
      LINT = -N + 1
      LFIN = 0
      DO 340 J=1,KACT
      LINT = LINT + N
      LFIN = LFIN + N
      JJ=JJ+1
C
C     NOTE THAT RESIDUAL PRINTED OUT CORRESPONDS TO VALUE OBTAINED
C     PRIOR TO FINAL PROJECTION Q(1)-TRANSPOSE*(B-INVERS)*Q(1) DONE
C     BEFORE TERMINATION
C
      WRITE(15,330) E(J), SM(JJ),TD(J)
  330 FORMAT(/E20.12,E13.4,E20.12,'BI-EVAL,ER**2,A-EVAL,A-EVEC'/)
  340 WRITE(15,350) (Q(L), L=LINT,LFIN)
      WRITE(15,360)
  350 FORMAT(4E20.12)
  360 FORMAT(/' ABOVE ARE COMPUTED APPROXIMATE EIGENVECTORS'/)
C
      IF(ITER.GT.MAXIT) WRITE(15,370) ITER,MAXIT
  370 FORMAT(//' PROCEDURE TERMINATED BECAUSE NUMBER OF CALLS TO BLSOLV
     1 SUBROUTINE',I6/' EXCEEDED MAXIMUM NUMBER ',I6,' ALLOWED'//)
C
      IF(ITER.LT.0) WRITE(15,380)
  380 FORMAT(//' USER BEWARE.  EIGENELEMENT COMPUTATIONS DEFAULTED BECAU
     1SE'/'   EISPACK SUBROUTINE DEFAULTED.   EIGENVALUE AND EIGENVECTOR
     1 APPROXIMATIONS'/'   ABOVE WERE THOSE AVAILABLE AT THE TIME OF DEF
     1AULT'/' SOMETHING IS SERIOUSLY WRONG.'//)
C
C     CHECK FOR TERMINATION AFTER PHASE 1
C     ITER < 0 MEANS EISPACK SUBROUTINE DEFAULTED
C     IPHASE = MPMIN MEANS THAT PHASE 2 TERMINATED DUE TO ORTHOGONALITY
C     IWRITE = MPMIN MEANS THAT CONVERGENCE APPEARS TO HAVE STAGNATED
C     ITER > MAXIT MEANS MAXIMUM NUMBER OF CALLS TO BLSOLV
C            ALLOWED BY USER WAS EXCEEDED
      IF(ITER.LT.0.OR.ITER.GT.MAXIT) GO TO 580
      IF(IPHASE.EQ.MPMIN.OR.IWRITE.EQ.MPMIN) GO TO 580
      IF(EFLAG.NE.1.OR.IPHASE.EQ.2) GO TO 580
C
C     ENTER 2ND PHASE OF COMPUTATION TO ATTEMPT TO OBTAIN MORE
```

```
C     ACCURATE EIGENVECTOR APPROXIMATIONS.                              BLI04300
C     USER CONTROLS THE SIZE OF THE ERROR TOLERANCE BY SPECIFYING       BLI04310
C     THE PARAMETER RELTOL.                                             BLI04320
C                                                                       BLI04330
      IPHASE = 2                                                        BLI04340
      MAXIT = MAXIT2                                                    BLI04350
      KSET = KACT                                                       BLI04360
C                                                                       BLI04370
C     ERROR TOLERANCE USES THE CONVERGED EIGENVALUE LARGEST IN          BLI04380
C     MAGNITUDE.                                                        BLI04390
      TD(1) = DABS(E(1))                                                BLI04400
      IF(KML.EQ.1) GO TO 400                                            BLI04410
      DO 390 J = 2,KML                                                  BLI04420
  390 IF(DABS(E(J)).GT.TD(1))    TD(1) = DABS(E(J))                     BLI04430
  400 TD(1) = DMAX1(TD(1),1.D0)                                         BLI04440
      ERRMAN = RELTOL**2 * TD(1)**2                                     BLI04450
      IF(ERRMAN.GE.ERRMAX) GO TO 480                                    BLI04460
      ERRMAX = ERRMAN                                                   BLI04470
C                                                                       BLI04480
      WRITE(6,410) ERRMAX, MAXIT2                                       BLI04490
  410 FORMAT(//' ENTER PHASE 2 OF COMPUTATION'/' CONVERGENCE CRITERION IBLI04500
     1S REDUCED TO ',E13.4/' NO MORE THAN ',I5,' CALLS TO SUBROUTINE BLSBLI04510
     1OLV WILL BE ALLOWED.'/' PROGRAM WILL TERMINATE IF BLOCK ORTHGONALIBLI04520
     1TY PROBLEMS MATERIALIZE'/)                                        BLI04530
C                                                                       BLI04540
      GO TO 210                                                         BLI04550
C                                                                       BLI04560
C     INCONSISTENCIES IN THE DATA                                       BLI04570
C                                                                       BLI04580
  420 WRITE(6,430) KM,KACT                                              BLI04590
  430 FORMAT(/' PROGRAM TERMINATES BECAUSE THE NUMBER OF EIGENELEMENTS  BLI04600
     1REQUESTED, KM =',I3/'  IS LARGER THAN THE SIZE OF THE FIRST Q BLOCBLI04610
     1K, KACT =',I3,' SPECIFIED'/'  USER MUST RESET KM OR KACT'/)       BLI04620
      GO TO 580                                                         BLI04630
C                                                                       BLI04640
  440 WRITE(6,450) KMAX,N                                               BLI04650
  450 FORMAT(/' PROGRAM TERMINATES BECAUSE KMAX = ',I5,' IS TOO LARGE FOBLI04660
     1R THE SIZE, N = ',I5,', OF THE GIVEN MATRIX'/'  USER MUST DECREASEBLI04670
     1THE SIZE OF KMAX.'/)                                              BLI04680
      GO TO 580                                                         BLI04690
C                                                                       BLI04700
  460 WRITE(6,470) NOLD,N,KACT,KSET                                     BLI04710
  470 FORMAT(/' PROGRAM TERMINATES BECAUSE FAULT OCCURRED IN READING IN BLI04720
     1THE EIGENVECTOR APPROXIMATIONS'/' EITHER THE SIZE MATRIX SPECIFIEDBLI04730
     1ON THE EIGENVECTOR FILE' ,I6/'  DID NOT MATCH THE SIZE SPECIFIED 'BLI04740
     1,I5,' IN THE PROGRAM OR THE NUMBER'/'    OF VECTORS IN FILE 10 = 'BLI04750
     1,I4,' IS LESS THAN THE NUMBER ',I3/'   USER SAID WERE THERE'/)    BLI04760
      GO TO 580                                                         BLI04770
C                                                                       BLI04780
  480 WRITE(6,490)  ERRMAN, ERRMAX                                      BLI04790
  490 FORMAT(/' COMPUTED PHASE 2 CONVERGENCE CRITERION ',E13.4/'  IS LARBLI04800
     1GER THAN PHASE 1 CRITERION ',E13.4/'  SO PROGRAM TERMINATES'/)    BLI04810
      GO TO 580                                                         BLI04820
C                                                                       BLI04830
  500 WRITE(6,510)  KACT,MXBLK                                          BLI04840
```

```
  510 FORMAT(/' PROGRAM TERMINATES BECAUSE THERE IS NOT ENOUGH ROOM TO  BLI04850
     1GENERATE 2 BLOCKS',' BECAUSE KACT = ',I3,' AND MXBLK = ', I4/)   BLI04860
      GO TO 580                                                         BLI04870
C                                                                       BLI04880
C                                                                       BLI04890
  520 WRITE(6,530)  MDIMTM, MXBLK                                       BLI04900
  530 FORMAT(/' PROGRAM TERMINATES BECAUSE THE DIMENSION ',I6,' OF THE TBLI04910
     1M ARRAY'/'   IS TOO SMALL FOR THE LARGEST T-MATRIX ALLOWED ',I4)  BLI04920
      GO TO 580                                                         BLI04930
C                                                                       BLI04940
  540 WRITE(6,550)                                                      BLI04950
  550 FORMAT(/' USER SPECIFIED NUMBER OF EIGENVALUES OF INTEREST AS 0'/'BLI04960
     1 PROGRAM TERMINATES FOR USER TO RESET KM TO DESIRED NONZERO VALUE'BLI04970
     1/)                                                                BLI04980
      GO TO 580                                                         BLI04990
C                                                                       BLI05000
  560 WRITE(6,570)  MDIMQ, KMAX,N                                       BLI05010
  570 FORMAT(/' PROGRAM TERMINATES BECAUSE THE DIMENSION ',I6,' OF THE QBLI05020
     1-ARRAY'/'   IS TOO SMALL TO HOLD ',I5, ' VECTORS OF LENGTH ',I4)  BLI05030
      GO TO 580                                                         BLI05040
C                                                                       BLI05050
  580 CONTINUE                                                          BLI05060
C                                                                       BLI05070
      STOP                                                              BLI05080
C-----END OF MAIN PROGRAM FOR INVERSE BLOCK LANCZOS PROCEDURE-----------BLI05090
      END                                                               BLI05100
```

9.3 BLIMULT: Sample Matrix-Vector Multiply Subroutines

```
C---BLIMULT-(INVERSES OF REAL SYMMETRIC MATRICES)----------------------BLI00010
C  Authors:  Jane Cullum* and Bill Donath**                             BLE00020
C          **IBM Research, T.J. Watson Research Center                  BLE00030
C          **Yorktown Heights, N.Y. 10598                               BLE00040
C          * Los Alamos National Laboratory                             BLE00050
C          * Los Alamos, New Mexico 87544                               BLE00060
C            E-mail:  cullumj@lanl.gov                                  BLE00065
C                                                                       BLI00070
C  These codes are copyrighted by the authors.  These codes             BLI00080
C  and modifications of them or portions of them are NOT to be          BLI00090
C  incorporated into any commercial codes or used for any other         BLI00100
C  commercial purposes such as consulting for other companies,          BLI00110
C  without legal agreements with the authors of these Codes.            BLI00120
C  If these Codes or portions of them are used in other scientific or   BLI00130
C  engineering research works the names of the authors of these codes   BLI00140
C  and appropriate references to their written work are to be           BLI00150
C  incorporated in the derivative works.                                BLI00160
C                                                                       BLI00170
C  This header is not to be removed from these codes.                   BLI00180
C                                                                       BLI00190
C         REFERENCE: Cullum and Willoughby, Chapter 7,                  BLI00191
C         Lanczos Algorithms for Large Symmetric Eigenvalue ComputationsBLI00192
C         VOL. 1 Theory. Republished as Volume 41 in SIAM CLASSICS in   BLI00193
C         Applied Mathematics, 2002. SIAM Publications,                 BLI00194
C         Philadelphia, PA. USA                                         BLI00195
C                                                                       BLI00200
C     CONTAINS SUBROUTINES  LANCZS AND SAMPLE USPEC AND BLSOLV          BLI00210
C     USED BY THE VERSION OF THE BLOCK LANCZOS ALGORITHMS FOR           BLI00220
C     FACTORED INVERSES OF REAL SYMMETRIC MATRICES, BLIVAL.             BLI00230
C     NOTE THAT SAMPLE BLSOLV FOR SPARSE MATRICES ASSUMES THAT          BLI00240
C     B-MATRIX IS POSITIVE DEFINITE AND USES CHOLESKY FACTORS.          BLI00250
C     HOWEVER, THE USER CAN DIRECTLY REPLACE THAT SUBROUTINE BY         BLI00260
C     A SUBROUTINE FOR INDEFINITE MATRICES THAT COMPUTES THE            BLI00270
C     GENERALIZED FACTORIZATION L*D*(L-TRANSPOSE).                      BLI00280
C                                                                       BLI00290
C     NONPORTABLE CONSTRUCTIONS:                                        BLI00300
C     1.  THE ENTRY MECHANISM USED TO PASS THE STORAGE LOCATIONS        BLI00310
C         OF THE FACTORIZATION OF THE MATRIX THAT WILL BE USED          BLI00320
C         BY THE LANCZS SUBROUTINE TO THE SUBROUTINE BLSOLV.            BLI00330
C     2.  IN THE SAMPLE USPEC AND BLSOLV SUBROUTINES PROVIDED:          BLI00340
C         THE FREE FORMAT (7,*), THE FORMAT (20A4) USED FOR             BLI00350
C         READING EXPLANATORY COMMENTS IN THE MATRIX SPECIFICATION      BLI00360
C         FILES, AND THE HEX FORMAT (4Z20) USED IN THE USPECS.          BLI00370
C     3.  THE COMMON BLOCK: LOOPS                                       BLI00380
C                                                                       BLI00390
C-----USPEC FOR FACTORED INVERSES OF REAL SYMMETRIC MATRICES------------BLI00400
C                                                                       BLI00410
      SUBROUTINE CUSPEC(N,MATNO,NNZ,AVER)                               BLI00420
C     SUBROUTINE USPEC(N,MATNO,NNZ,AVER)                                BLI00430
C                                                                       BLI00440
C-----------------------------------------------------------------------BLI00450
```

```
      DOUBLE PRECISION  BD(2200),BSD(10000),NNZ,AVER
      INTEGER  KCOL(2200),KROW(10000),IPR(2200),IPT(2200)
C-----------------------------------------------------------------------
C     THIS SAMPLE SUBROUTINE ASSUMES THAT B IS POSITIVE DEFINITE
C     USER COULD REPLACE BY SIMILAR SUBROUTINE FOR GENERAL FACTORIZATION
C     DIMENSIONS ARRAYS NEEDED TO DEFINE CHOLESKY FACTOR OF B-MATRIX,
C     READS CHOLESKY FACTOR FROM FILE 7, AND THEN PASSES STORAGE
C     LOCATIONS OF THESE ARRAYS TO THE B-MATRIX SOLVE SUBROUTINE BLSOLV.
C
C     HERE WE HAVE B = P*C*P' = L*L'  WHERE  C = S0*A + SHIFT*I.
C     P IS A PERMUTATION MATRIX DEFINED BY THE VECTOR MAPS IPR AND IPT.
C     THE ITH ROW OF B CORRESPONDS TO THE JTH ROW OF C (A) WHERE
C     J = IPR(I) AND I = IPT(J).
C
C     THE B-CHOLESKY FACTOR IS STORED IN THE FOLLOWING SPARSE FORMAT:
C     N = ORDER OF THE B-MATRIX.
C     NZT = NUMBER OF NONZERO SUBDIAGONAL ENTRIES IN THE CHOLESKY
C           FACTOR, L.
C     KCOL(J), J=1,N IS THE NUMBER OF NONZERO SUBDIAGONAL ELEMENTS IN
C              COLUMN J OF L.
C     KROW(K), K=1,NZT IS THE ROW INDEX FOR CORRESPONDING ENTRY BSD(K).
C     BD(J), J = 1,N CONTAINS THE DIAGONAL ENTRIES OF L.
C     BSD(K), K =1,NZT CONTAINS THE NONZERO SUBDIAGONAL ENTRIES OF L
C     JPERM = (0,1):  1 MEANS CHOLEKSY FACTOR CORRESPONDS TO
C                     PERMUTED C.  0 MEANS NO PERMUTATION WAS USED.
C-----------------------------------------------------------------------
C     READ CHOLESKY FACTOR FROM FILE 7.  MUST BE STORED
C     IN SPARSE MATRIX FORMAT.
      READ(7,10) NZT,NOLD,NZL,MATOLD,JPERM
   10 FORMAT(I10,2I6,I8,I6)
C
      WRITE(6,20) NZT,NZL,N,NOLD,MATOLD,JPERM
   20 FORMAT(' HEADER, CHOLESKY FACTOR FILE'/
     1 3X,'NZT',3X,'NZL',5X,'N',2X,'NOLD',2X,'MATOLD',1X,'JPERM'/
     1 4I6,I8,I6/)
C
      IF (N.NE.NOLD.OR.MATNO.NE.MATOLD) GO TO 100
C
      READ(7,30) (KCOL(K), K = 1,NZL)
      READ(7,30) (KROW(K), K = 1,NZT)
   30 FORMAT(13I6)
      READ(7,40) (BD(K), K = 1,N)
      READ(7,40) (BSD(K), K = 1,NZT)
   40 FORMAT(4Z20)
C  20 FORMAT(3E25.16)
C
C     DOES CHOLESKY FACTOR CORRESPOND TO PERMUTED B?
      IF(JPERM.EQ.0)  GO TO 60
      READ(7,30) (IPR(K), K = 1,N)
C
      DO 50 K = 1,N
      J = IPR(K)
   50 IPT(J) = K
C-----------------------------------------------------------------------
      CALL LPERME(IPR,IPT,N)
```

```
C-----------------------------------------------------------------------
   60 CONTINUE
C
C     COMPUTE NNZ, THE AVERAGE NUMBER OF NONZEROS PER COLUMN, AND
C     AVER, THE AVERAGE SIZE OF NONZERO ENTRIES IN THE FACTORS
C     OF THE B-MATRIX.  FROM THIS, ESTIMATE (TOO CRUDELY) THE
C     AVERAGE FOR B-INVERSE AS AVER = 1/AVER.
      ITCOL = 0
      AVER = 0.D0
      DO 70 K = 1,N
      IF(DABS(BD(K)).EQ.0.D0)  GO TO 70
      ITCOL = ITCOL + 1
      AVER = AVER + DABS(BD(K))
   70 CONTINUE
      NTCOL = ITCOL
      DO 80 K = 1,N
   80 ITCOL = ITCOL + 2*KCOL(K)
      NNZ = DFLOAT(ITCOL)/DFLOAT(N)
      DO 90 K = 1,NZS
   90 AVER = AVER + DABS(BSD(K))
      AVER = AVER/DFLOAT(NZS + NTCOL)
      AVER = 1.D0/AVER
C
C-----------------------------------------------------------------------
C     PASS STORAGE LOCATIONS OF FACTORS TO INVERSION SUBROUTINE BLSOLV
      CALL BSOLVE(BSD,BD,KCOL,KROW,N,NZT,NZL)
C-----------------------------------------------------------------------
C
      GO TO 120
C
  100 CONTINUE
C     DEFAULT EXIT
      WRITE(6,110)
  110 FORMAT(/' TERMINATE.  PARAMETERS IN CHOLESKY FACTOR FILE'/
     1' DO NOT AGREE WITH THOSE SPECIFIED BY THE USER'/)
      STOP
C
  120 CONTINUE
C-----END OF USPEC------------------------------------------------------
      RETURN
      END
C
C-----BLSOLV-(FACTORED INVERSES OF REAL SYMMETRIC MATRICES)-------------
C
C     SUBROUTINE BLSOLV(V,U)
      SUBROUTINE CBSOLV(V,U)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  BD(1),BSD(1),U(1),V(1),TEMP,ZERO,ONE
      INTEGER  KCOL(1),KROW(1)
      COMMON/LOOPS/MAXIT,ITER
C-----------------------------------------------------------------------
      GO TO 3
C-----------------------------------------------------------------------
      ENTRY BSOLVE(BSD,BD,KCOL,KROW,N,NZT,NZL)
```

```
      GO TO 4                                                           BLI01560
C-----------------------------------------------------------------------BLI01570
    3 CONTINUE                                                          BLI01580
      ITER = ITER + 1                                                   BLI01590
      ZERO = 0.0D0                                                      BLI01600
      ONE  = 1.0D0                                                      BLI01610
C     SOLVE B*U = V FOR U WHERE  B = L*L'                               BLI01620
C     SET U = V. FIRST SOLVE L*U = U FOR U, THEN SOLVE L'*U = U FOR U   BLI01630
      KL = 0                                                            BLI01640
      DO 10 K = 1,N                                                     BLI01650
   10 U(K) = V(K)                                                       BLI01660
      DO 30 K = 1,N                                                     BLI01670
      TEMP = U(K)/BD(K)                                                 BLI01680
      U(K) = TEMP                                                       BLI01690
      IF (KCOL(K).EQ.0.OR.K.EQ.N) GO TO 30                              BLI01700
      KF = KL + 1                                                       BLI01710
      KL = KL + KCOL(K)                                                 BLI01720
      DO 20 KK = KF,KL                                                  BLI01730
      KR = KROW(KK)                                                     BLI01740
   20 U(KR) = U(KR) - TEMP*BSD(KK)                                      BLI01750
   30 CONTINUE                                                          BLI01760
      NP1 = N+1                                                         BLI01770
      KF = NZT + 1                                                      BLI01780
      DO 50 K = 1,N                                                     BLI01790
      L = NP1 - K                                                       BLI01800
      TEMP = U(L)                                                       BLI01810
      IF (KCOL(L).EQ.0.OR.L.EQ.N) GO TO 50                              BLI01820
      KL = KF - 1                                                       BLI01830
      KF = KF - KCOL(L)                                                 BLI01840
      DO 40 LL = KF,KL                                                  BLI01850
      LR = KROW(LL)                                                     BLI01860
   40 TEMP = TEMP - BSD(LL)*U(LR)                                       BLI01870
   50 U(L) = TEMP/BD(L)                                                 BLI01880
   60 CONTINUE                                                          BLI01890
C                                                                       BLI01900
    4 RETURN                                                            BLI01910
C                                                                       BLI01920
C-----END OF BLSOLV-----------------------------------------------------BLI01930
      END                                                               BLI01940
C                                                                       BLI01950
C-----SUBROUTINES FOR DIAGONAL TEST MATRICES----------------------------BLI01960
C     BLSOLV AND USPEC SUBROUTINES FOR DIAGONAL TEST MATRICES           BLI01970
C                                                                       BLI01980
C-------BLSOLV DIAGONAL TEST MATRIX-------------------------------------BLI01990
C                                                                       BLI02000
C     SUBROUTINE DBSOLV(V,U)                                            BLI02010
      SUBROUTINE BLSOLV(V,U)                                            BLI02020
C                                                                       BLI02030
C-----------------------------------------------------------------------BLI02040
      DOUBLE PRECISION  V(1),U(1),D(1)                                  BLI02050
      COMMON/LOOPS/MAXIT,ITER                                           BLI02060
C-----------------------------------------------------------------------BLI02070
      GO TO 3                                                           BLI02080
C-----------------------------------------------------------------------BLI02090
C     BELOW ENTRY IS FOR A DIAGONAL TEST MATRIX                         BLI02100
```

```
      ENTRY DSOLVE(D,N)
      GO TO 4
C-----------------------------------------------------------------------
    3 CONTINUE
      ITER = ITER + 1
   10 DO 20 I=1,N
   20 U(I)= V(I)/D(I)
C  20 U(I)= -V(I)/D(I)
C
   30 CONTINUE
    4 RETURN
C-----END OF 'DIAGONAL' TEST MATRIX  BLSOLV-----------------------------
      END
C
C-----START OF USPEC FOR DIAGONAL TEST MATRIX---------------------------
C
      SUBROUTINE USPEC(N,MATNO,NNZ,AVER)
C     SUBROUTINE DUSPEC(N,MATNO,NNZ,AVER)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  D(1000),DI(1000),SHIFT,SPACE,NNZ,AVER
      DOUBLE PRECISION  DABS, DFLOAT
      REAL  EXPLAN(20)
C-----------------------------------------------------------------------
C
      READ(7,10) EXPLAN
   10 FORMAT(20A4)
      READ(7,*) NOLD,NUNIF,SPACE,D(1),SHIFT
      NNUNIF = NOLD - NUNIF
      WRITE(6,20) NOLD,SPACE,NNUNIF,D(1),SHIFT
   20 FORMAT(/' DIAGONAL TEST MATRIX, SIZE = ',I4/' IS THE INVERSE OF MA
     1TRIX WITH MOST ENTRIES',E10.3/' UNITS APART AND WITH ',I3,' ENTRIE
     1S IRREGULARLY SPACED'/' FIRST ENTRY WAS ',E13.4,' SHIFT = ',E10.3
     1/)
C
      IF(N.NE.NOLD) GO TO 120
C     COMPUTE THE UNIFORM PORTION OF THE SPECTRUM
      DO 30 J=2,NUNIF
   30 D(J) = D(1) - DFLOAT(J-1)*SPACE
      NUNIF1=NUNIF + 1
      READ(7,10)  EXPLAN
      DO 40 J=NUNIF1,N
   40 READ(7,*) D(J)
      NB = NUNIF - 2
C
      IF(SHIFT.EQ.0.) GO TO 60
      DO 50 J=1,N
   50 D(J) = D(J) + SHIFT
C
C     COMPUTE EIGENVALUES OF INVERSE FOR PRINTOUT ONLY
   60 DO 70 J = 1,N
   70 DI(J) = 1.D0/D(J)
      WRITE(6,80) (J,DI(J), J=1,N )
   80 FORMAT(/' INVERSE BLOCK LANCZOS TEST, LANCZS USES INVERSE OF GIVEN
     1MATRIX'/' ENTRIES OF INVERSE OF DIAGONAL TEST MATRIX = '/(I4,E20.1
```

```
     12,I4,E20.12,I4,E20.12))
C
C      DIAGONAL GENERATION COMPLETE
C
C      COMPUTE NNZ AND AVER
       NNZ = 1.D0
       AVER = 0.D0
       DO 90 K = 1,N
    90 AVER = AVER + DABS(DI(K))
       AVER = AVER/DFLOAT(N)
       AVER = 1.D0/AVER
C
C      COMPUTE THE GAPS
       N1 = N-1
       DO 100 K = 1,N1
   100 DI(K) = DI(K+1) - DI(K)
       WRITE(6,110) (K,DI(K), K=1,N1)
   110 FORMAT(/' GAPS BETWEEN EIGENVALUES'/(I4,E13.4,I4,E13.4,I4,E13.4,I4
     1,E13.4))
C
C-----------------------------------------------------------------------
C      PASS STORAGE LOCATIONS OF D AND N TO DSOLV SUBROUTINE
       CALL DSOLVE(D,N)
C-----------------------------------------------------------------------
C
       RETURN
   120 WRITE(6,130) NOLD,N
   130 FORMAT(' PROGRAM TERMINATES BECAUSE NOLD = ',I5,'DOES NOT EQUAL N
     1 =',I5)
C-----END OF USPEC SUBROUTINE FOR 'DIAGONAL' TEST MATRICES--------------
       STOP
       END
```

9.4 BLIEVAL: File Definitions, Sample Input File

Below is a listing of the input/output files which are accessed by the real symmetric block Lanczos eigenvalue/eigenvector program, BLIEVAL. This program calculates a few eigenvalues and corresponding eigenvectors of a real symmetric matrix A by computing a few extreme eigenvalues and corresponding eigenvectors of the inverse of a real symmetric matrix B obtained from A by scaling, shifting and permuting A.

Also below is a sample of an input file which BLIEVAL requires on file 5. The parameters in this file are supplied in free format. File 8 contains data for the nxn real symmetric matrix A.

```
Sample Definitions of Input/Output Files for BLIEVAL
----------------------------------------------------------------
 BLIEVAL EXEC
FI 06 TERM
FILEDEF  5 DISK BLIEVAL   INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF  8 DISK &1        INPUT     A (RECFM F LRECL 80 BLOCK 80
FILEDEF 10 DISK &1        BLSTARTV  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 13 DISK &1        BLEXTRAV  A (RECFM F LRECL 80 BLOCK 80
FILEDEF 15 DISK &1        BLEIGVEC  A (RECFM F LRECL 80 BLOCK 80
*IMTQL2 AND TRED2 ARE 2 EISPACK LIBRARY SUBROUTINES
LOAD  BLIEVAL  BLSUB  BLIMULT  IMTQL2  TRED2
----------------------------------------------------------------
```

```
Sample Input File for BLIEVAL
------------------------------------------------------------------------
 LINE 1 IWRITE (SPECIFY MESSAGE LEVEL TO FILE 6: 1 MEANS DETAILED
            1
 LINE 2 N    MATNO    SO    SHIFT   JPERM (SIZE,ID,SCALE,SHIFT,PERM?
      1250   1250    1.       0.       0
 LINE 3 MDIMQ         MDIMTM       MAXIT (DIMS. Q, TM, MAX Ax-Mults
       40000           2500        1000
 LINE 4 EFLAG  OFLAG ( EFLAG=(0,1) 1=2PHASES. OFLAG: 1=ORTHOG CHECK
           1      1
 LINE 5 SEED     (STARTING VECTOR SEED, RANDOM NUMBER GENERATOR
     3482736
 LINE 6 KMAX  KACT  KSET (MAX T SIZE +1,SIZE 1ST BLOCK,VECTORS SUPPLIED
          31     3     0
 LINE 7 KM (NUMB. EVS FOR ALG-LARGEST, -(NUMB. EVS) FOR ALG-SMALLEST
         3
 LINE 8 NSTAG  FRACT (NO. ITNS BEFORE TEST CONVERGENCE, TEST FRACTION
            25    .05
 LINE 9 RELTOL         MAXIT2 (PHASE 2,CONVERGE.TOL., Max Ax-Mults
     .00000001           1000
------------------------------------------------------------------------
```

Chapter 10

Errata: Volume I: Theory

1. Chapter 4: Page 162: Section: Real Symmetric Generalized Eigenvalue Problems
   (a) Line 10: $Ax = \lambda x$ should be $Ax = \lambda Bx$
   (b) In Eqn(4.9.11),
   $$|\beta_{i+1}| = \|L^{-1}(Av_i - \alpha_i v_i - \beta_i v_{i-1})\|$$
   should be
   $$|\beta_{i+1}| = \|L^{-1}(Av_i - \alpha_i Bv_i - \beta_i Bv_{i-1})\|$$

REFERENCES

[1] Bjorck,Å. (1967). *Solving linear least square problems by Gram-Schmidt orthogonalization.* BIT, **7**, 1 – 21.

[2] Bunch J. R. and Kaufman, L. (1977). *Some stable methods for calculating inertia and solving symmetric linear systems.* Math. Comp., **31**, 163 – 179.

[3] Cullum, J., and Donath, W. E. (1974). *A block generalization of the symmetric s-step Lanczos algorithm.* IBM T. J. Watson Research Center, Research Report RC 4845, Yorktown Heights, New York, USA, 10598. (May 1974).

[4] Cullum, J., and Donath, W. E. (1974). *A block Lanczos algorithms for computing the q algebraically largest eigenvalues and a corresponding eigenspace for large, sparse symmetric matrices.* In Proceedings 1974 IEEE Conference on Decision and Control, 505–509.

[5] Cullum, Jane K, and Willoughby, Ralph A. (1985, 2002). *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. I, Theory.* Birkhäuser, Basel 1985. (Republished as Volume **41**, SIAM Classics in Applied Mathematics, 2002.)

[6] Cullum, J. (1978). *The simultaneous computation of a few of the algebraically largest and smallest eigenvalues of a large, symmetric, sparse matrix.* BIT, **18**, 265-275.

[7] Dongarra, J. J., Bunch, J. R., Moler, C. B., and Stewart, G. W.(1979) *LINPACK User's Guide.* SIAM, Philadelphia, PA, USA.

[8] Garbow, B. S., Boyle, J. M., Dongarra, J. J., and Moler, C. B. (1977) *Matrix Eigensystem Routines - EISPACK Guide Extension.* Lecture Notes in Computer Science **51**, Second Edition, Springer, New York.

[9] George, A., Liu, J. W. H., and Ng, E.(1979). *SPARSPAK User Guide.* CS Dept. Tech. Report, U. Waterloo. Waterloo, Canada.

[10] George, A., and Liu, J. W. H. (1981). *Computer Solution of Large Sparse Positive Definite Systems.* Prentice-Hall, Englewood Cliffs, NJ.

[11] Golub, G. H., and Kahan, W. (1965). *Calculating the singular values and pseudoinverse of a matrix.* SIAM J. Numer. Anal. **2**, 205-224.

[12] Golub, G. H., and Underwood, R. (1977). *The block Lanczos method for computing eigenvalues.* In Mathematical Software III. Rice, J. R. (Ed.), Academic Press, New York, 361-377.

[13] Golub, G. H., Luk, F. T., and Overton, M. L. (1981). *A block Lanczos method for computing the singular values and corresponding singular vectors of a matrix.* Trans. Math. Software, **7**, 149-169.

[14] Lanczos, C. (1950). *An iterative method for the solution of the eigenvalue problem of linear differential and integral operators.*, J. Res. Nat. Bur. Standards, Sect. B, **45**, 255-282.

[15] Lanczos, C. (1961). *Linear Differential Operators.* Van Nostrand, New York.

[16] Moro, G., and Freed, J. H. (1981). *Calculation of ESR spectra and related Fokker-Planck forms by the use of the Lanczos algorithm.* J. Chem. Phys. **74**, 3757-3773.

[17] Paige, C. C. (1971). *The computation of eigenvalues and eigenvectors of very large sparse matrices.* Ph.D. Thesis, U. London.

[18] Paige, C. C. (1972). *Computational variants of the Lanczos method for the eigenproblem.* J.Inst. Math. Appl. **10**, 373-381.

[19] Paige, C. C. (1976). *Error analysis of the Lanczos algorithms for tridiagonalizing a symmetric matrix.* J. Inst. Math. Appl. **18**, 341-349.

[20] Paige, C. C. (1980). *Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem.* Linear Algebra Appl. **34**, 235-258.

[21] Ryder, B. G. (1974). *The PFORT Verifier.* Software-Practice and Experience, **4**, 359-377.

[22] Ryder, B. G., and Hall, A. D.(1981). *PFORT Verifier.* Bell Laboratory Computer Science Technical Report 12. Bell Laboratory, Murray Hill, New Jersey, USA

[23] Smith, B. T., Boyle, J. M., Garbow, B. S., Ikebe, Y., Klema, V. C., and Moler, C. B. (1976), *Matrix Eigensystem Routines - EISPACK Guide.* Lecture Notes in Computer Science, **6**, Second Edition, Springer, New York.

[24] Stewart, G. W. (1973). *Introduction to Matrix Computations.* Academic Press, New York.

[25] Wilkinson, J. H. (1965). *The Algebraic Eigenvalue Problem.* Oxford University Press, New York.

Related Work by the Authors

[1] Cullum, J. and Zhang, T. (2002). *Two-Sided Arnoldi and Nonsymmetric Lanczos Algorithms.* SIAM J. Mat. Anal. Appl., **24**, 303 – 319.

[2] Cullum, J. and Ruehli, A. (2001), *Pseudospectra analysis, nonlinear eigenvalue problems, and studying systems with time delays*, BIT, **41**, 265–281

[3] Cullum, J.K. and Ruehli. A.E. (2000), *Method for analyzing the stability and passivity of system models*, US Patent, 6058258 (May 5, 2000).

[4] Cullum, Jane (1996). *Arnoldi versus nonsymmetric Lanczos algorithms for solving matrix eigenvalue problems,* BIT **36**, 470–493.

[5] Cullum, Jane and Willoughby, Ralph A. (1996). *A QL procedure for computing the eigenvalues of complex symmetric tridiagonal matrices,* SIAM J. Matrix Anal. Appl., **17**, 83–109.

[6] Cullum, Jane and Willoughby, Ralph (1991). *Computing eigenvalues of large matrices, some Lanczos algorithms and a shift and invert strategy.* Advances in Numerical Partial Differential Equations and Optimization, Eds., S. Gomez, J.P. Hennart and R.A. Tapia, SIAM Publications Philadelphia, PA, 198–246.

[7] Cullum, J., Kerner, W. and Willoughby, R.A. (1989). *A generalized nonsymmetric Lanczos procedure.*, Computer Physics Communications, North-Holland, **53**, 19–48.

[8] Cullum, Jane and Willoughby, Ralph A., (1986). Eds., *Large Scale Eigenvalue Problems*, Mathematics Studies, **127**, North-Holland.

[9] Cullum, Jane and Willoughby, Ralph A. (1986). *A practical procedure for computing eigenvalues of large sparse nonsymmetric matrices.* In Large Scale Eigenvalue Problems, Mathematics Studies, eds., J. Cullum and R.A. Willoughby, **127**, North-Holland, 193–240.

[10] Cullum, Jane and Willoughby, Ralph A. (1984). *A Lanczos algorithm for the modal analysis of very large, nondefective, nonsymmetric matrices,* in Proceedings of the IEEE 23rd Conference on Decision and Control, Las Vegas, Nevada, Dec. 12–14, 1984, IEEE Press, 1758–1761.

[11] Chow, J.H., Cullum, Jane, and Willoughby, Ralph A. (1983). *A sparsity-based technique for identifying slow-coherent areas in large power systems*, IEEE Transactions on Power Apparatus and Systems, **103**, 463–473.

[12] Cullum, J., Willoughby, R. A., and Lake, M. (1983). *A Lanczos algorithm for computing singular values and vectors of large matrices.* SIAM J. Sci. Statist. Comput. **4**, 197–215.

[13] Cullum, J., and Willoughby, R. A. (1981). *Computing eigenvalues of very large symmetric matrices-an implementation of a Lanczos algorithm with no reorthogonalization.* J. Comput. Phys. **44**, 329–358.

[14] Cullum, Jane and Willoughby, Ralph A. (1980). *The Lanczos phenomenon: An interpretation based upon conjugate gradient optimization,* Linear Algebra and Its Applications, **29**, 63–90.

[15] Cullum, Jane and Willoughby, R. A. (1980). *Computing eigenvectors (and eigenvalues) of large, symmetric matrices using Lanczos tridiagonalization.* Lecture Notes in Mathematics No. **773**, Numerical Analysis, Proceedings Dundee, Scotland, 1979, Ed. G. A. Watson, Springer-Verlag, 46–63.

[16] Cullum, Jane (1979). *Using Lanczos tridiagonalization to compute eigenvalues and eigenvectors of large symmetric matrices,* Conjugate Gradient Methods and Similar Techniques, ed. I. S. Duff, AERE Report R-9636, Computer Science Division, A.E.R.E. Harwell, Oxfordshire, England, December 1979, 72–90.

[17] Cullum, Jane and Willoughby, R. A. (1979). *Fast modal analysis of large sparse but unstructured symmetric matrices.*. Proceedings of 17th IEEE Conference on Decision and Control, San Diego, Calif., January 1979, IEEE Press, 45–53.

[18] Cullum, J., and Willoughby, R. A. (1979). *Lanczos and the computation in specified intervals of the spectrum of large sparse real symmetric matrices.* in Proceedings of Symposium on Sparse Matrix Theory, Knoxville, Tenn., Nov. 2–3, 1978, Sparse Matrix Proceedings 1978, Eds., I. S. Duff and G. W. Stewart, SIAM, Philadelphia, 220-255.

[19] Cullum, J., Donath, W. E., and Wolfe, P. (1975). *The minimization of certain nondifferentiable sums of eigenvalues of symmetric matrices.* In Nondifferentiable Optimization, Mathematical Programming Study **3**, North-Holland, 35-55,