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 = S0*P*A*P' + SHIFT*I WHERE THE SCALE S0 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        BLE01060
C         FOR COMPUTING THE Q ALGEBRAICALLY-LARGEST EIGENVALUES AND     BLE01070
C         A CORRESPONDING EIGENSPACE OF LARGE, SPARSE REAL SYMMETRIC    BLE01080
C         MATRICES, PROCEEDINGS OF THE 1974 IEEE CONFERENCE ON          BLE01090
C         DECISION AND CONTROL, PHOENIX, ARIZONA, PP.505-509, NOVEMBER  BLE01100
C         1974.                                                         BLE01110
C                                                                       BLE01120
C     3.  JANE CULLUM,  AN ACCELERATED 'BLOCK' LANCZOS ALGORITHM        BLE01130
C         FOR A FEW EXTREME EIGENVALUES OF A LARGE, SPARSE REAL         BLE01140
C         SYMMETRIC MATRIX.  IBM REPORT 1983.  PRESENTED AT THE         BLE01150
C         SPARSE MATRIX CONFERENCE, FAIRFIELD GLADE, TENNESSEE,         BLE01160
C         OCTOBER 1982.                                                 BLE01170
C                                                                       BLE01180
C                                                                       BLE01190
C-----PORTABILITY-------------------------------------------------------BLE01200
C                                                                       BLE01210
C     PROGRAMS WERE TESTED FOR PORTABILITY USING THE PFORT VERIFIER.    BLE01220
C     FOR DETAILS OF THE VERIFIER SEE FOR EXAMPLE, B. G. RYDER AND      BLE01230
C     A. D. HALL, "THE PFORT VERIFIER", COMPUTING SCIENCE TECHNICAL     BLE01240
C     REPORT 12, BELL LABORATORIES, MURRAY HILL, NEW JERSEY 07974,      BLE01250
C     (REVISED), JANUARY 1981.                                          BLE01260
C                                                                       BLE01270
C     EXCEPT FOR THE FOLLOWING CONSTRUCTIONS WHICH CAN BE EASILY        BLE01280
C     MODIFIED BY THE USER TO MATCH THE PARTICULAR COMPUTER BEING       BLE01290
C     USED, THE PROGRAM STATEMENTS ARE PORTABLE.                        BLE01300
C                                                                       BLE01310
C    NONPORTABLE STATEMENTS.                                            BLE01320
C                                                                       BLE01330
C    IN BLEVAL, BLIEVAL (MAIN PROGRAMS)                                 BLE01340
C         1.  DATA/MACHEP STATEMENT                                     BLE01350
C         2.  ALL READ(5,*) STATEMENTS (FREE FORMAT)                    BLE01360
C         3.  FORMAT(20A4) USED FOR THE EXPLANATORY HEADER ARRAY, EXPLANBLE01370
C         4.  FORMAT(4Z20) WHICH CAN BE USED TO WRITE LARGE VECTOR      BLE01380
C             FILES                                                     BLE01390
C         5.  THE COMMON BLOCK: LOOPS.                                  BLE01400
C    IN BLMULT, BLIMULT                                                 BLE01410
C         1.  IN BMATV, BLSOLV, AND USPEC, THE ENTRIES WHICH            BLE01420
C             PASS THE STORAGE LOCATIONS OF THE ARRAYS DEFINING         BLE01430
C             THE USER-SPECIFIED MATRIX OR FACTORIZATION.               BLE01440
C    IN BLSUB                                                           BLE01450
C         1.  ALL STATEMENTS ARE PORTABLE EXCEPT THE ENTRY TO           BLE01460
C             SUBROUTINE LPERM WHICH PASSES THE PERMUTATION USED        BLE01470
C             TO OBTAIN THE B-MATRIX FROM SUBROUTINE USPEC.             BLE01480
C             SUBROUTINE LPERM IS USED ONLY IN CASE (2).                BLE01490
C                                                                       BLE01500
C                                                                       BLE01510
C-----MATRIX SPECIFICATION----------------------------------------------BLE01520
C                                                                       BLE01530
C     SUBROUTINE USPEC IS USED TO SPECIFY THE MATRIX WHICH THE BLOCK    BLE01540
C     LANCZOS PROCEDURE WILL USE.  IN CASE (1) THIS IS THE USER-        BLE01550
C     SPECIFIED A-MATRIX.   IN CASE (2) THE FACTORIZATION OF THE        BLE01560
C     ASSOCIATED B-MATRIX IS SPECIFIED.  SUBROUTINE USPEC HAS THE       BLE01570
C     CALLING SEQUENCE                                                  BLE01580
C                                                                       BLE01590
C           CALL USPEC(N,MATNO,NNZ,AVER)                                BLE01600
```

```
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.
C
C     MASK = MASKS OVERFLOW AND UNDERFLOW.  OPTIONAL.
C            USER MUST SUPPLY OR COMMENT OUT CALL.
C
C    USPEC = DIMENSIONS AND INITIALIZES ARRAYS NEEDED TO SPECIFY
C            MATRIX USED BY LANCZS SUBROUTINE.  SEE MATRIX
C            SPECIFICATION SECTION.
C
C    BMATV = CASE (1) ONLY: COMPUTES MATRIX-VECTOR MULTIPLY FOR
C            USER-SUPPLIED A-MATRIX. SEE MATRIX SPECIFICATION SECTION.
C
C   BLSOLV = CASE (2) ONLY:  FOR GIVEN VECTOR V, COMPUTES U SUCH
C            B*U = V, GIVEN THE SPARSE FACTORIZTION OF THE B-MATRIX.
C
C
C-----PARAMETER CONTROLS------------------------------------------------
C
C
C     PARAMETER CONTROLS ARE INTRODUCED TO CONTROL VARIOUS
C     ASPECTS OF THESE PROGRAMS.
C
C     THE FLAG EFLAG SPECIFIES THE NUMBER OF COMPUTATIONAL PHASES.
C
C     EFLAG  = (0,1)  MEANS
C
C              (0) PROGRAM TERMINATES AFTER COMPLETING PHASE 1
C                  COMPUTATIONS.
C
C              (1) PROGRAM COMPLETES BOTH PHASE 1 AND PHASE 2 OF
C                  THE COMPUTATIONS.
C
C     THE FLAG OFLAG CONTROLS THE ORTHOGONALITY CHECKS BETWEEN THE
C     JTH Q-BLOCK GENERATED AND THAT VECTOR IN THE 1ST Q-BLOCK THAT
C     IS GENERATING DESCENDANTS.  FOR SAFETY, OFLAG SHOULD BE 1.
C
C     OFLAG = (0,1)  MEANS
C
C              (0) NO ORTHOGONALITY CHECKS ARE MADE ON PHASE
C                  1 PORTION OF THE COMPUTATIONS.  ORTHOGONALITY
C                  CHECKS ARE ALWAYS MADE ON PHASE 2 PORTION.
C
C              (1) PROGRAM CHECKS ORTHOGONALITY OF GENERATED
C                  Q-BLOCKS W.R.T. THAT VECTOR IN THE 1ST Q-BLOCK
C                  THAT IS GENERATING DESCENDANTS IN BOTH PHASE
C                  1 AND PHASE 2 OF THE COMPUTATIONS.
C
C     THE FLAG IWRITE DETERMINES THE AMOUNT OF OUTPUT TO FILE 6
C     DURING THE COMPUTATIONS
C
C     IWRITE = (0,1)   MEANS
C
C              (0) ABBREVIATED OUTPUT TO FILE 6.
C
C              (1) ADDITIONAL COMMENTARY ON THE COMPUTATIONS IS
```

```
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
C            IS MONITORED AND USED AS A MEASURE OF THE RATE OF
C            CONVERGENCE OF THE BLOCK PROCEDURE.
C
C     FRACT = DOUBLE PRECISION SCALAR.  EXPECTED OR HOPED FOR
C            FRACTIONAL CHANGE IN THE KM-TH RESIDUAL OVER THE PAST
C            BLOCK LANCZOS ITERATIONS USED TO TEST FOR STAGNATION
C            OF CONVERGENCE.
C
C     NNZ   = DOUBLE PRECISION SCALAR.  AVERAGE NUMBER OF NONZERO
C            ENTRIES PER ROW IN THE MATRIX USED IN THE LANCZOS
C            PROCEDURE.
C
C
C     AVER  = DOUBLE PRECISION SCALAR.  AVERAGE SIZE OF THE NONZERO
C            ENTRIES IN THE MATRIX USED IN THE LANCZOS PROCEDURE.
C
C     CASE (2) ONLY:
C
C     S0, SHIFT = DOUBLE PRECISION SCALARS.  MATRIX USED BY LANCZS
C                SUBROUTINE IS B = S0*P*A*P' + SHIFT*I  WHERE P
C                DENOTES A PERMUTATION MATRIX SELECTED TO PRESERVE
C                THE SPARSITY OF A IN THE FACTORIZATION OF B.
C                S0 AND SHIFT ARE CHOSEN BY THE USER SO THAT THE
C                DESIRED EIGENVALUES BECOME THE EXTREME EIGENVALUES
C                OF B-INVERSE.
C
C
C-----CONVERGENCE TEST-------------------------------------------------
C
C
C     THE CONVERGENCE TEST INCORPORATED IN THIS PROGRAM IS
C     BASED UPON THE FOLLOWING FACT:  GIVEN A REAL SYMMETRIC
C     MATRIX A, A VECTOR X OF NORM 1, AND A SCALAR EVAL
C     THEN THERE EXISTS AN EIGENVALUE AEVAL OF A SUCH THAT
C     DABS(AEVAL - EVAL) .LE. NORM(A*X - EVAL*X).  WITHIN
C     EACH ITERATION OF THE BLOCK LANCZOS PROCESS THESE TYPES
C     OF NORMS ARE COMPUTED IN THE PROCESS OF COMPUTING THE
C     2ND Q-BLOCK.
C
C
C-----ARRAYS REQUIRED---------------------------------------------------
C
C
C     Q(J)     = DOUBLE PRECISON ARRAY.  ITS DIMENSION MUST BE AT
C               LEAST AS LARGE AS KMAX*N, WHERE N IS THE ORDER OF
C               THE GIVEN MATRIX, AND MXBLK = KMAX - 1 IS THE
C               MAXIMUM SIZE T-MATRIX ALLOWED ON ANY GIVEN
C               ITERATION.  THE COLUMNS OF Q HOLD THE LANCZOS
C               VECTORS GENERATED ON EACH ITERATION OF BLOCK
C               LANCZOS PLUS THERE MUST BE AN ADDITIONAL COLUMN
C               AVAILABLE FOR WORK SPACE.  THE FIRST KACT COLUMNS
C               OF Q CONTAIN THE CURRENT APPROXIMATING EIGENSPACE.
C
C     E(J)    = DOUBLE PRECISION ARRAY.  ITS DIMENSION MUST BE AT
```

```
C                LEAST MXBLK = KMAX - 1.  ON EACH ITERATION CONTAINS
C                THE COMPUTED EIGENVALUES OF THE LANCZOS T-MATRIX.
C
C     TM(J)   = DOUBLE PRECISION ARRAY.  ITS DIMENSION MUST BE AT
C                LEAST MXBLK**2 WHERE MXBLK = KMAX - 1.  CONTAINS
C                THE LANCZOS T-MATRIX GENERATED ON EACH ITERATION
C                AND THEN THE COMPUTED EIGENVECTORS OF THIS MATRIX.
C                EISPACK SUBROUTINES ARE USED FOR THE SMALL
C                EIGENELEMENT COMPUTATIONS.  EISPACK SUBROUTINE
C                TRED2 IS USED TO REDUCE THE GIVEN T-MATRIX TO
C                TRIDIAGONAL FORM.  THE EIGENELEMENT PROBLEM FOR THE
C                TRIDIAGONAL MATRIX IS THEN SOLVED USING THE EISPACK
C                SUBROUTINE IMTQL2.
C
C     EXPLAN(J) = REAL ARRAY.  ITS DIMENSION IS 20.  THIS ARRAY IS
C                  USED TO ALLOW EXPLANATORY COMMENTS IN THE INPUT FILES.
C
C     G(J) = REAL ARRAY.  ITS DIMENSION MUST BE >= N.  IT IS USED
C            FOR HOLDING THE PSEUDO-RANDOM NUMBERS USED TO GENERATE
C            ANY STARTING VECTORS NOT SUPPLIED BY THE USER.
C
C     RESIDL(J),  = DOUBLE PRECISION ARRAYS.  DIMENSION >= MAXIMUM
C     RESIDK(J),    NUMBER OF ITERATIONS ALLOWED.  MAXIMUM IS
C                   CURRENTLY SET TO 100. USED TO MONITOR THE
C                   RATE OF CONVERGENCE.
C
C     TD(J), TOD(J), = DOUBLE PRECISION ARRAYS.  DIMENSION >= MXBLK.
C         SM(J)        WORK SPACES.
C
C     DESC(J), XLFT(J), = INTEGER ARRAYS.  DIMENSION >= MXBLK.
C         LEFT(J)         WORK SPACES.
C
C     DIR(2,J)  = 2-DIMENSIONAL INTEGER ARRAY.  COLUMN DIMENSION >=
C                  MXBLK, ROW DIMENSION 2.  KEEPS TRACK OF NUMBER
C                  OF VECTORS IN EACH QBLOCK.
C
C     CASE (2) ONLY:
C
C     IPR(J), IPT(J) = INTEGER ARRAYS.  EACH OF DIMENSION AT LEAST N.
C                      USED TO STORE THE REORDERING (IF ANY) OF
C                      THE GIVEN MATRIX.
C
C     OTHER ARRAYS
C
C     THE USER IN THE SUBROUTINE USPEC MUST SPECIFY WHATEVER ARRAYS
C     ARE REQUIRED TO DEFINE THE MATRIX BEING USED BY LANCZS.
C
C
C-----SUBROUTINES INCLUDED----------------------------------------------
C
C
C     LANCZS = CONTAINS MAJOR LOOP FOR BLOCK LANCZOS PROCEDURES.
C              CALLED FROM MAIN PROGRAM, CALLS SUBROUTINE LANCI1
C              TO GENERATE WITHIN A GIVEN ITERATION THE Q-BLOCKS
C              AND CORRESPONDING LANCZOS T-MATRICES.  THEN CALLS
```

```
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
C
C     READ USER-SPECIFIED DIMENSIONS OF Q-ARRAY (MDIMQ) AND OF THE
C     TM-ARRAY (MDIMTM).  READ MAXIMUM NUMBER (MAXIT) OF MATRIX-VECTOR
C     MULTIPLIES ALLOWED IN PHASE 1.
      READ(5,20) EXPLAN
      READ(5,*) MDIMQ, MDIMTM, MAXIT
C
C     READ FLAGS:   EFLAG = (0,1).  EFLAG = 0, MEANS PROGRAM STOPS
C     AFTER COMPLETING PHASE 1 PORTION OF BLOCK LANCZOS PROCEDURE.
C     EFLAG = 1, MEANS PROGRAM COMPLETES BOTH PHASES BEFORE
C     TERMINATING.
C     OFLAG = (0,1).  OFLAG = 0, MEANS THAT IN PHASE 1 PORTION
C     OF THE COMPUTATION, THE PROGRAM DOES NO ORTHOGONALITY CHECKS
C     ON THE Q-BLOCKS GENERATED.  OFLAG = 1 MEANS THAT IN THE
C     PHASE 1 PORTION AND IN THE PHASE 2 PORTIONS OF THE COMPUTATIONS
C     THE PROGRAM CHECKS THE ORTHOGONALITY OF THE Q-BLOCKS GENERATED
C     W.R.T. THAT VECTOR IN THE FIRST BLOCK THAT IS GENERATING
C     DESCENDANTS.  NOTE THAT IN PHASE 2, THE PROGRAM ALWAYS MAKES
C     THIS CHECK OF ORTHOGONALITY REGARDLESS OF THE VALUE OF OFLAG.
C     FOR SAFETY, OFLAG SHOULD ALWAYS BE SET TO 1, ALTHOUGH IN MANY
C     PROBLEMS THIS IS NOT NECESSARY.
      READ(5,20) EXPLAN
      READ(5,*) EFLAG,OFLAG
C
C     READ SEED USED BY SUBROUTINE GENRAN TO OBTAIN THOSE STARTING
C     VECTORS WHICH ARE GENERATED RANDOMLY.
      READ(5,20) EXPLAN
      READ(5,*) SEED
C
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 ON THE SIZE OF KACT.
      READ(5,20) EXPLAN
      READ(5,*) KMAX,KACT,KSET
C
C     SPECIFY NUMBER OF EXTREME EIGENVALUES AND EIGENVECTORS TO BE
C     COMPUTED (KM).  USER CAN SPECIFY THAT THE ALGEBRAICALLY-
C     SMALLEST EIGENVALUES ARE BEING COMPUTED BY SETTING KM < 0.
C     PROGRAM THEN ASSUMES THAT THE MATRIX-VECTOR MULTIPLY
C     SUBROUTINE WHICH THE USER HAS PROVIDED IS COMPUTING -A*X
C     INSTEAD OF A*X AND INTERNALLY IT COMPUTES THE |KM|
C     ALGEBRAICALLY-LARGEST EIGENVALUES OF -A.
      READ(5,20) EXPLAN
      READ(5,*) KM
      IF(KM.EQ.0) GO TO 490
      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.  IN THE TESTS FRACT WAS SET TO .01.
      READ(5,20) EXPLAN
```

```
      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'/     BLE02090
     1I15,E26.4/4X,'AVERAGE SIZE OF NONZERO ENTRIES',5X,'MATRIX ID'/    BLE02100
     1E25.4,I21/)                                                       BLE02110
C                                                                       BLE02120
      WRITE(6,90) MDIMQ, MDIMTM                                         BLE02130
   90 FORMAT(/18X,'USER-SPECIFIED'/2X,'MAX. DIMENSION Q-ARRAY',4X,'MAX. BLE02140
     1DIMENSION TM-ARRAY'/I16,I26/)                                     BLE02150
C                                                                       BLE02160
      WRITE(6,100)  OFLAG, EFLAG                                        BLE02170
  100 FORMAT(/4X,'OFLAG',4X,'EFLAG'/I8,I9/)                             BLE02180
C                                                                       BLE02190
      IF(EFLAG.EQ.1)  WRITE(6,110) MAXIT,RELTOL,MAXIT2                  BLE02200
  110 FORMAT(/4X,' MAXIT ',8X,'  RELTOL  ',6X,' MAXIT2 '/I10,E20.6,I12/)BLE02210
      IF(EFLAG.EQ.0)  WRITE(6,120) MAXIT                                BLE02220
  120 FORMAT(/4X,' MAXIT '/I10/)                                        BLE02230
C                                                                       BLE02240
      WRITE(6,130)  SEED                                                BLE02250
  130 FORMAT(/' SEED FOR RANDOM NUMBER GENERATOR'/I24/)                 BLE02260
C                                                                       BLE02270
      IF(KM.GT.0)  WRITE(6,140) KML                                     BLE02280
  140 FORMAT(/' COMPUTE THE',I3,' ALGEBRAICALLY-LARGEST EIGENVALUES AND BLE02290
     1CORRESPONDING VECTORS'/)                                          BLE02300
      IF(KM.LT.0)  WRITE(6,150) KML                                     BLE02310
  150 FORMAT(/' COMPUTE THE',I3,' ALGEBRAICALLY-SMALLEST EIGENVALUES ANDBLE02320
     1 CORRESPONDING VECTORS'/' PROGRAM ASSUMES THAT USER IS PROVIDING -BLE02330
     1A*X INSTEAD OF A*X'/' AND COMPUTES THE ALGEBRAICALLY-LARGEST EIGENBLE02340
     1VALUES OF -A.'/' HOWEVER ON EXIT, FILE 15 CONTAINS THE ALGEBRAICALBLE02350
     1LY-SMALLEST EIGENVALUES OF'/' THE ORIGINAL A-MATRIX AND CORRESPONDBLE02360
     1ING EIGENVECTORS.'/)                                              BLE02370
      IF(KM.LT.0) KM = - KM                                             BLE02380
C                                                                       BLE02390
C     COMPUTE PHASE 1 CONVERGENCE TOLERANCE                             BLE02400
      IF(AVER.GE.1.)                                                    BLE02410
     1ERRMAX = 2.D0*DFLOAT(N+1000)*NNZ*AVER*MACHEP                      BLE02420
      IF(AVER.LT.1.)                                                    BLE02430
     1ERRMAX = 2.D0*DFLOAT(N+1000)*NNZ*AVER**2*MACHEP                   BLE02440
C                                                                       BLE02450
      WRITE(6,160) KACT,MXBLK,KSET                                      BLE02460
  160 FORMAT(/' ON INITIAL ITERATIONS, THE FIRST BLOCK CONTAINS ',I3,' VBLE02470
     1ECTORS'/' HOWEVER THE SIZE OF THE FIRST BLOCK MAY CHANGE AS THE ITBLE02480
     1ERATIONS PROCEED'/' THE MAXIMUM SIZE T-MATRIX THAT CAN BE GENERATEBLE02490
     1D IS ',I4/' THE USER SUPPLIED ',I3,' STARTING VECTORS'/)          BLE02500
C                                                                       BLE02510
      WRITE(6,170)                                                      BLE02520
  170 FORMAT(/' ITERATIVE PROCEDURE'/' PROCEDURE MONITORS THE SIZES OF TBLE02530
     1HE NORM(GRADIENTS)**2 ON EACH'/' ITERATION.  CONVERGENCE IS SAID  BLE02540
     1TO HAVE OCCURRED WHEN ALL'/' RELEVANT (NORMS)**2 ARE LESS THAN ERRBLE02550
     1MAX',E10.3/' TYPICALLY, PHASE 1 ERRMAX YIELDS SOMEWHAT LESS THAN'/BLE02560
     1' SINGLE PRECISION ACCURACY.  PHASE 2 REFINES THE VECTORS OBTAINEDBLE02570
     1'/' ON PHASE 1, ACCORDING TO THE ACCURACY SPECIFIED BY THE USER'/)BLE02580
C                                                                       BLE02590
      WRITE(6,180) ERRMAX                                               BLE02600
  180 FORMAT(//' PHASE 1 CONVERGENCE CRITERION, ERRMAX '/E22.3/)        BLE02610
C                                                                       BLE02620
C-----------------------------------------------------------------------BLE02630
```

```
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-----------------------------------------------------------------------BLS00440
      DOUBLE PRECISION  E(1),Q(1),ERRMAX,SPREC,RESN,FRACT,RKM,SUM       BLS00450
      DOUBLE PRECISION  TM(1),SM(1),TD(*),TOD(1),RESIDL(*),RESIDK(1)    BLS00460
      REAL  G(1)                                                        BLS00470
      INTEGER  EFLAG,OFLAG,DIR(2,*),DESC(1),LEFT(1),XLFT(*)             BLS00480
      DOUBLE PRECISION FINPRO                                           BLS00490
      COMMON /LOOPS/MAXIT,ITER                                          BLS00500
      COMMON/FLAGS/EFLAG,OFLAG                                          BLS00510
      EXTERNAL MATVEC                                                   BLS00520
C-----------------------------------------------------------------------BLS00530
      GO TO 3                                                           BLS00540
C-----------------------------------------------------------------------BLS00550
C     ENTRY RECEIVES STORAGE LOCATIONS OF SEVERAL OF THE ARRAYS         BLS00560
C     USED BY THE LANCZS SUBROUTINE.  THIS ALLOWS USER TO SPECIFY       BLS00570
C     THE DIMENSIONS OF THESE ARRAYS IN THE MAIN PROGRAM.               BLS00580
C                                                                       BLS00590
      ENTRY LANZP(DIR,DESC,SM,TM,TOD,TD,G,XLFT,LEFT,SPREC)              BLS00600
      GO TO 4                                                           BLS00610
C-----------------------------------------------------------------------BLS00620
C                                                                       BLS00630
    3 CONTINUE                                                          BLS00640
C                                                                       BLS00650
      KM = KML                                                          BLS00660
      MMT = MXBLK*MXBLK                                                 BLS00670
      MPMIN = -1000                                                     BLS00680
      IKACT = KACT + 10                                                 BLS00690
      FRACT = RESIDL(1)                                                 BLS00700
      NSTAG = RESIDL(2)                                                 BLS00710
      IORTHO = 0                                                        BLS00720
C                                                                       BLS00730
C     CONSTRUCT STARTING VECTORS                                        BLS00740
      IF(KSET.EQ.0) GO TO 10                                            BLS00750
C-----------------------------------------------------------------------BLS00760
      CALL ORTHOG(1,KSET,N,Q)                                           BLS00770
   10 CALL START (KSET+1,KACT,N,Q,G,ERRMAX)                             BLS00780
C-----------------------------------------------------------------------BLS00790
   20 CONTINUE                                                          BLS00800
C     INITIALIZE THE LANCZOS T-MATRIX.                                  BLS00810
      DO 30 J=1,MMT                                                     BLS00820
   30 TM(J)=0.D0                                                        BLS00830
C                                                                       BLS00840
C     INITIALIZE THE Q-BLOCK DIRECTORY                                  BLS00850
      DIR(1,1)=1                                                        BLS00860
      DIR(2,1)=KACT                                                     BLS00870
C                                                                       BLS00880
C     ORTHOGONALIZE THE STARTING VECTORS                                BLS00890
      IF(NITER.EQ.0) GO TO 40                                           BLS00900
C-----------------------------------------------------------------------BLS00910
      CALL ORTHOG(1,KACT,N,Q)                                           BLS00920
C-----------------------------------------------------------------------BLS00930
   40 CONTINUE                                                          BLS00940
C                                                                       BLS00950
C     GENERATE THE QSUBBLOCKS USED ON ITERATION NITER AND STORE IN      BLS00960
C     THE Q-ARRAY                                                       BLS00970
C                                                                       BLS00980
```

```
      DO 90 I=1,MXBLK                                                   BLS00990
C                                                                       BLS01000
C-----------------------------------------------------------------------BLS01010
      CALL LANCI1(MATVEC,MXBLK,NITER,I,N,Q,KACT,KML,ERRMAX,RESN,RKM,    BLS01020
     1 IND,KACTN,IWRITE)                                                BLS01030
C-----------------------------------------------------------------------BLS01040
C                                                                       BLS01050
C     HAS CONVERGENCE OCCURRED?                                         BLS01060
      II = I+1                                                          BLS01070
      IF (I.EQ.1.AND.DIR(2,I).EQ.DIR(2,II)) GO TO 140                   BLS01080
C                                                                       BLS01090
C     WAS THERE ROOM FOR ANOTHER Q-BLOCK?                               BLS01100
      IF (DIR(2,II).LT.DIR(1,II))  GO TO 100                            BLS01110
C                                                                       BLS01120
C     IF OFLAG = 1 OR IPHASE = 2, CHECK THE ORTHOGONALITY OF            BLS01130
C     THE Q-SUBBLOCKS GENERATED WITH RESPECT TO THAT VECTOR             BLS01140
C     IN THE 1ST Q-BLOCK WHICH IS GENERATING DESCENDANTS.               BLS01150
C     IN PHASE 2 LOSSES IN ORTHOGONALITY ARE USED TO                    BLS01160
C     DETERMINE WHEN THE LIMITS ON THE ACHIEVABLE ACCURACY HAVE         BLS01170
C     BEEN REACHED.                                                     BLS01180
C                                                                       BLS01190
      IF(OFLAG.EQ.0.AND.IPHASE.EQ.1) GO TO 90                           BLS01200
C                                                                       BLS01210
      L1=DIR(1,II)                                                      BLS01220
      LL1 = (L1-1)*N + 1                                                BLS01230
      IND1 = (IND-1)*N + 1                                              BLS01240
C-----------------------------------------------------------------------BLS01250
      SUM = FINPRO(N,Q(IND1),1,Q(LL1),1)                                BLS01260
C-----------------------------------------------------------------------BLS01270
C                                                                       BLS01280
      IF(DABS(SUM).LT.SPREC)GO  TO 80                                   BLS01290
C                                                                       BLS01300
      IF(IWRITE.EQ.1) WRITE(6,50)  IND,L1,SUM,I                         BLS01310
   50 FORMAT(/' INNER PRODUCT OF VECTORS ',I3,' AND ',I3,' = ',E13.3/   BLS01320
     1' THIS VIOLATES ORTHOGONALITY TEST.  TERMINATE BLOCK GENERATION'  BLS01330
     1/' WITH ',I3,'TH BLOCK '/)                                        BLS01340
C                                                                       BLS01350
C     ORTHOGONALITY TEST VIOLATED, TERMINATE BLOCK GENERATION           BLS01360
C     FOR THIS ITERATION.  IN PHASE 2 KEEP TRACK OF NUMBER OF           BLS01370
C     SUCH VIOLATIONS THAT LIMIT THE NUMBER OF BLOCKS TO < 10.          BLS01380
C     TERMINATE AFTER 3 SUCH VIOLATIONS IN PHASE 2.                     BLS01390
      IF(IPHASE.NE.1.AND.I.LT.IKACT)  IORTHO = IORTHO + 1               BLS01400
      IF(IORTHO.LT.3.AND.II.NE.2)  GO TO 70                             BLS01410
      WRITE(6,60)                                                       BLS01420
   60 FORMAT(/' THE ORTHOGONALITY TEST HAS FAILED THREE TIMES'/         BLS01430
     1' TERMINATE THE BLOCK PROCEDURE'/)                                BLS01440
      IPHASE = -1000                                                    BLS01450
C     BEFORE TERMINATING WRITE THE CURRENT EIGENVECTOR/EIGENVALUE       BLS01460
C     APPROXIMATIONS TO FILE 15                                         BLS01470
      GO TO 160                                                         BLS01480
C                                                                       BLS01490
C     TERMINATE THE Q-BLOCK GENERATION ON THIS ITERATION                BLS01500
   70 DIR(2,II)=DIR(2,I)                                                BLS01510
      GO TO 100                                                         BLS01520
C                                                                       BLS01530
```

```
   80 CONTINUE                                                          BLS01540
C                                                                       BLS01550
C     END OF ORTHOGONALITY TESTS                                        BLS01560
C                                                                       BLS01570
   90 CONTINUE                                                          BLS01580
C                                                                       BLS01590
C     END OF RECURSIVE Q-BLOCK GENERATION                               BLS01600
C                                                                       BLS01610
  100 CONTINUE                                                          BLS01620
      MM = DIR(2,II)                                                    BLS01630
      IF(IWRITE.EQ.1)  WRITE (6,110)  MM,I                              BLS01640
  110 FORMAT(' T-MATRIX IS OF ORDER ',I3, ' NUMBER OF BLOCKS = ',I3)    BLS01650
C                                                                       BLS01660
C-----------------------------------------------------------------------BLS01670
C     DIAGONALIZE THE PROJECTION MATRIX TM.  ON RETURN THE              BLS01680
C     UPDATED APPROXIMATIONS TO THE DESIRED EIGENVECTORS ARE IN THE     BLS01690
C     FIRST KACT COLUMNS OF THE Q-ARRAY.                                BLS01700
C     UPDATED EIGENVALUE APPROXIMATIONS ARE IN E.                       BLS01710
      TD(1) = RKM                                                       BLS01720
      TD(2) = FRACT                                                     BLS01730
      IERR = NSTAG                                                      BLS01740
C                                                                       BLS01750
      CALL DIAGOM(MXBLK,MM,TM,KACT,N,Q,E,RESIDL,RESIDK,                 BLS01760
     1 RESN,IND,KACTN,KM,TD,TOD,NITER,IERR,IWRITE)                      BLS01770
C-----------------------------------------------------------------------BLS01780
C                                                                       BLS01790
C     INCREMENT COUNTER FOR NUMBER OF BLOCK LANCZOS ITERATIONS          BLS01800
      NITER = NITER + 1                                                 BLS01810
C     IWRITE = MPMIN MEANS BLOCK LANCZOS PROCEDURE TERMINATED ABNORMALLYBLS01820
      IF(IWRITE.EQ.MPMIN)  GO TO 160                                    BLS01830
C     IERR .NE. 0 MEANS EISPACK SUBROUTINE DEFAULTED                    BLS01840
      IF(IERR.EQ.0) GO TO 130                                           BLS01850
      WRITE(6,120)                                                      BLS01860
  120 FORMAT(/' EISPACK SIGNALS TROUBLE IN SMALL IMTQL2 EIGENVALUE SUBROBLS01870
     1UTINE,'/' SO BLOCK LANCZOS PROGRAM TERMINATES'/)                  BLS01880
      ITER = -ITER                                                      BLS01890
C                                                                       BLS01900
      RETURN                                                            BLS01910
C                                                                       BLS01920
  130 IF (ITER.GE.MAXIT) GO TO 160                                      BLS01930
C                                                                       BLS01940
C     UPDATED APPROXIMATIONS WERE OBTAINED WITHOUT EXCEEDING            BLS01950
C     MAXIMUM NUMBER OF MATRIX-VECTOR MULTIPLIES SET BY THE USER.       BLS01960
C     CONTINUE BLOCK LANCZOS LOOP ITERATIONS                            BLS01970
C                                                                       BLS01980
      GO TO 20                                                          BLS01990
C                                                                       BLS02000
  140 WRITE(6,150)                                                      BLS02010
  150 FORMAT(//' BLOCK LANCZOS PROCEDURE CONVERGED'//)                  BLS02020
C                                                                       BLS02030
C     BLOCK LANCZOS PROCEDURE HAS CONVERGED.                            BLS02040
C     ATTEMPT TO IMPROVE THE APPROXIMATE EIGENVECTORS BY DIAGONALIZING  BLS02050
C     THE SMALL PROJECTION MATRIX OBTAINED BY USING ONLY THE            BLS02060
C     FIRST BLOCK IN Q-ARRAY.                                           BLS02070
C                                                                       BLS02080
```

```
  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                                                                       BLS03740
C     COMPUTE ASSOCIATED CORRECTION TERMS IN TM-MATRIX.                 BLS03750
      IF(XLFT(I).EQ.0) GO TO 50                                         BLS03760
      LUP = XLFT(I)                                                     BLS03770
      DO 40 JJ = 1,LUP                                                  BLS03780
      L= LEFT(JJ)                                                       BLS03790
      JL1 = (L-1)*N + 1                                                 BLS03800
C-----------------------------------------------------------------------BLS03810
      SUM = FINPRO(N,Q(JI31),1,Q(JL1),1)                                BLS03820
C-----------------------------------------------------------------------BLS03830
      KK = (L-1)*MXBLK + K                                              BLS03840
      TM(KK) = SUM + TM(KK)                                             BLS03850
   40 CONTINUE                                                          BLS03860
C                                                                       BLS03870
   50 CONTINUE                                                          BLS03880
C                                                                       BLS03890
   60 CONTINUE                                                          BLS03900
C                                                                       BLS03910
      RETURN                                                            BLS03920
C                                                                       BLS03930
C     ON EVERY BLOCK PASS THROUGH HERE TO GENERATE THE ITH-BLOCK        BLS03940
C     DIAGONAL ENTRY A(I) OF THE TM-MATRIX, EXCEPT THE LAST DIAGONAL    BLS03950
C     BLOCK WHICH IS GENERATED ABOVE                                    BLS03960
C                                                                       BLS03970
   70 CONTINUE                                                          BLS03980
C     COMPUTE (A-MATRIX)*(ITH-Q-BLOCK)                                  BLS03990
      KA=I2                                                             BLS04000
      DO 80 K=I1,I2                                                     BLS04010
      KA=KA+1                                                           BLS04020
      JKA1 = (KA-1)*N + 1                                               BLS04030
      JK1 = (K-1)*N + 1                                                 BLS04040
C-----------------------------------------------------------------------BLS04050
      CALL MATVEC(Q(JK1),Q(JKA1))                                       BLS04060
C-----------------------------------------------------------------------BLS04070
      DESC(K)=KA                                                        BLS04080
   80 DESC(KA)=K                                                        BLS04090
C                                                                       BLS04100
C     COMPUTE (A-MATRIX)*(ITH-Q-BLOCK) - ((I-1)TH-Q-BLOCK)*B(I)-TRANS   BLS04110
C     WHERE B(I) DENOTES THE ITH SUBDIAGONAL BLOCK                      BLS04120
C                                                                       BLS04130
      IF(I.EQ.1) GO TO 110                                              BLS04140
      J1 = DIR(1,I-1)                                                   BLS04150
      J2 = DIR(2,I-1)                                                   BLS04160
      DO 100 K=I1,I2                                                    BLS04170
      KD=DESC(K)                                                        BLS04180
      JKD0 = (KD-1)*N                                                   BLS04190
      KK = (J1-2)*MXBLK + K                                             BLS04200
      DO 90 L=J1,J2                                                     BLS04210
      JL = (L-1)*N                                                      BLS04220
      KK = KK + MXBLK                                                   BLS04230
      S=TM(KK)                                                          BLS04240
      JKD = JKD0                                                        BLS04250
      DO 90 J=1,N                                                       BLS04260
      JKD = JKD + 1                                                     BLS04270
      JL = JL + 1                                                       BLS04280
```

```
   90 Q(JKD) = Q(JKD) - S*Q(JL)                                          BLS04290
  100 CONTINUE                                                           BLS04300
      LINT = (KD-1)*N + 1                                                BLS04310
      LFIN = KD*N                                                        BLS04320
C                                                                        BLS04330
C     COMPUTE A(I)                                                       BLS04340
C                                                                        BLS04350
  110 DO 130 K=I1,I2                                                     BLS04360
      KKMX = (K-1)*MXBLK                                                 BLS04370
      KD=DESC(K)                                                         BLS04380
      JKD1 = (KD-1)*N+ 1                                                 BLS04390
      JL1 = (K-2)*N + 1                                                  BLS04400
      DO 120 L=K,I2                                                      BLS04410
      JL1 = JL1 + N                                                      BLS04420
      KK = KKMX + L                                                      BLS04430
C-----------------------------------------------------------------------BLS04440
      TM(KK) = FINPRO(N,Q(JL1),1,Q(JKD1),1)                              BLS04450
C-----------------------------------------------------------------------BLS04460
  120 CONTINUE                                                           BLS04470
  130 CONTINUE                                                           BLS04480
C                                                                        BLS04490
C     COMPUTE P(I) = P(I) - (ITH-Q-BLOCK)*A(I)                           BLS04500
C                                                                        BLS04510
      DO 170 K=I1,I2                                                     BLS04520
      KKMX = (K-1)*MXBLK                                                 BLS04530
      KD=DESC(K)                                                         BLS04540
      JKD0 = (KD-1)*N                                                    BLS04550
      JL = (I1-1)*N                                                      BLS04560
      DO 140 L=I1,I2                                                     BLS04570
      KK = KKMX + L                                                      BLS04580
      IF(L.LT.K) KK=(L-1)*MXBLK + K                                      BLS04590
      S=TM(KK)                                                           BLS04600
      JKD = JKD0                                                         BLS04610
      DO 140 J=1,N                                                       BLS04620
      JL = JL + 1                                                        BLS04630
      JKD = JKD + 1                                                      BLS04640
  140 Q(JKD) = Q(JKD) - S*Q(JL)                                          BLS04650
C                                                                        BLS04660
C     REORTHOGONALIZE THE BLOCK P(I) WITH RESPECT TO ALL VECTORS         BLS04670
C     IN THE 1ST QBLOCK THAT ARE NOT CURRENTLY GENERATING ANY            BLS04680
C     DESCENDANTS.  NOTE THAT 2ND Q-BLOCK IS REORTHOGONALIZED            BLS04690
C     ELSEWHERE.                                                         BLS04700
      IF(XLFT(I).EQ.0) GO TO 170                                         BLS04710
      LUP = XLFT(I)                                                      BLS04720
      DO 160 JJ = 1,LUP                                                  BLS04730
      L= LEFT(JJ)                                                        BLS04740
      JL0 = (L-1)*N                                                      BLS04750
      LLMX = (L-1)*MXBLK                                                 BLS04760
      JL1 = JL0 + 1                                                      BLS04770
      JKD1 = JKD0 + 1                                                    BLS04780
C-----------------------------------------------------------------------BLS04790
      SUM = FINPRO(N,Q(JL1),1,Q(JKD1),1)                                 BLS04800
C-----------------------------------------------------------------------BLS04810
      JKD = JKD0                                                         BLS04820
      JL = JL0                                                           BLS04830
```

```
      DO 150 J=1,N                                                      BLS04840
      JKD = JKD + 1                                                     BLS04850
      JL = JL + 1                                                       BLS04860
  150 Q(JKD) = Q(JKD) - SUM* Q(JL)                                      BLS04870
      KK = LLMX + K                                                     BLS04880
      TM(KK) = SUM + TM(KK)                                             BLS04890
C                                                                       BLS04900
  160 CONTINUE                                                          BLS04910
  170 CONTINUE                                                          BLS04920
C                                                                       BLS04930
C                                                                       BLS04940
C   GENERATE B(I+1)                                                     BLS04950
C                                                                       BLS04960
      K1=DESC(I1)                                                       BLS04970
      K2=DESC(I2)                                                       BLS04980
      IFLAG=0                                                           BLS04990
C                                                                       BLS05000
C     COMPUTE NORMS                                                     BLS05010
C                                                                       BLS05020
  180 CONTINUE                                                          BLS05030
      JK1 = (K1-2)*N + 1                                                BLS05040
      DO 190 K=K1,K2                                                    BLS05050
      JK1 = JK1 + N                                                     BLS05060
C-----------------------------------------------------------------------BLS05070
      SM(K) = FINPRO(N,Q(JK1),1,Q(JK1),1)                               BLS05080
C-----------------------------------------------------------------------BLS05090
  190 CONTINUE                                                          BLS05100
C                                                                       BLS05110
        IF(I.EQ.1.AND.K1.EQ.I2+1)  WRITE(6,200) NITER,                  BLS05120
     1 (K,SM(K), K =K1,K2)                                              BLS05130
  200 FORMAT(//' ON ITERATION', I4,' NORM(GRADIENTS)**2 OF 1ST BLOCK = 'BLS05140
     1/5(I4,E12.3))                                                     BLS05150
C                                                                       BLS05160
C     TEST FOR CONVERGENCE OF BLOCK LANCZOS                             BLS05170
C                                                                       BLS05180
      IF(I.GT.1.OR.K1.GT.I2+1)  GO TO 250                               BLS05190
C                                                                       BLS05200
C     TEST THE FIRST KM OF THE EIGENVALUES FOR CONVERGENCE              BLS05210
      K2L = K1 + KML - 1                                                BLS05220
      RKM = SM(K2L)                                                     BLS05230
      DO 210 K = K1,K2L                                                 BLS05240
      IF(SM(K).GT.ERRMAX )  GO TO 220                                   BLS05250
  210 CONTINUE                                                          BLS05260
      GO TO 430                                                         BLS05270
C                                                                       BLS05280
C     CAN WE REDUCE KACT?  IF A SMALL RESIDUAL (GRADIENT) IS IDENTIFIED,BLS05290
C     SIZE OF 1ST BLOCK MAY BE REDUCED.                                 BLS05300
  220 IF(KML.EQ.KACT) GO TO 250                                         BLS05310
      DO 230 K = K2L,K2                                                 BLS05320
      IF(SM(K).GT.ERRMAX) GO TO 230                                     BLS05330
      KSAV = K                                                          BLS05340
      KACTN = KSAV - KACT                                               BLS05350
      GO TO 240                                                         BLS05360
C                                                                       BLS05370
  230 CONTINUE                                                          BLS05380
```

```
      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-----------------------------------------------------------------------BLS07040
      S = FINPRO(N,Q(LL),1,Q(MM),1)                                     BLS07050
C-----------------------------------------------------------------------BLS07060
C                                                                       BLS07070
      IF (M.EQ.L) GO TO 20                                              BLS07080
C                                                                       BLS07090
      MM = MM0                                                          BLS07100
      LL = LL0                                                          BLS07110
      DO 10 I=1,N                                                       BLS07120
      LL = LL + 1                                                       BLS07130
      MM = MM + 1                                                       BLS07140
   10 Q(MM) = Q(MM) - S*Q(LL)                                           BLS07150
      GO TO 40                                                          BLS07160
C                                                                       BLS07170
   20 S = DSQRT(S)                                                      BLS07180
      MM = MM0                                                          BLS07190
      DO 30 I=1,N                                                       BLS07200
      MM = MM + 1                                                       BLS07210
   30 Q(MM) = Q(MM)/S                                                   BLS07220
C                                                                       BLS07230
   40 CONTINUE                                                          BLS07240
   50 CONTINUE                                                          BLS07250
C                                                                       BLS07260
      RETURN                                                            BLS07270
C-----END OF ORTHOG-----------------------------------------------------BLS07280
      END                                                               BLS07290
C                                                                       BLS07300
C-----START-------------------------------------------------------------BLS07310
C     GENERATES PSEUDO-RANDOM STARTING VECTORS.                         BLS07320
C                                                                       BLS07330
      SUBROUTINE START(KA,KB,N,Q,G,ERRMAX)                              BLS07340
C                                                                       BLS07350
C-----------------------------------------------------------------------BLS07360
      DOUBLE PRECISION  Q(1), ERRMAX, S                                 BLS07370
      REAL  G(1)                                                        BLS07380
      COMMON/RANDOM/IIX                                                 BLS07390
      DOUBLE PRECISION FINPRO, DSQRT                                    BLS07400
C-----------------------------------------------------------------------BLS07410
      IF(KA.GT.KB) RETURN                                               BLS07420
C                                                                       BLS07430
      IIL = IIX                                                         BLS07440
      DO 110 K = KA,KB                                                  BLS07450
      KK0 = (K-1)*N                                                     BLS07460
C                                                                       BLS07470
C-----------------------------------------------------------------------BLS07480
      CALL GENRAN(IIL,G,N)                                              BLS07490
C-----------------------------------------------------------------------BLS07500
C                                                                       BLS07510
      KK = KK0                                                          BLS07520
      DO 10 I = 1,N                                                     BLS07530
      KK = KK + 1                                                       BLS07540
   10 Q(KK) = G(I)                                                      BLS07550
      LL0 = -N                                                          BLS07560
   20 DO 70 L=1,K                                                       BLS07570
      LL0 = LL0 + N                                                     BLS07580
```

```
      LL = LL0 + 1
      KK = KK0 + 1
C-----------------------------------------------------------------------
      S = FINPRO(N,Q(LL),1,Q(KK),1)
C-----------------------------------------------------------------------
C
      IF (K.EQ.L) GO TO 40
C
      LL = LL0
      KK = KK0
      DO 30 I=1,N
      LL = LL + 1
      KK = KK + 1
   30 Q(KK) = Q(KK) - S*Q(LL)
      GO TO 70
C
   40 S = DSQRT(S)
      IF(S.LE.ERRMAX) GO TO 80
      KK = KK0
      DO 50 I=1,N
      KK = KK + 1
   50 Q(KK) = Q(KK)/S
C
      WRITE(6,60) K
   60 FORMAT(I6,' TH STARTING VECTOR IS GENERATED RANDOMLY')
C
   70 CONTINUE
      GO TO 110
C
C-----------------------------------------------------------------------
   80 CALL GENRAN(IIX,G,N)
C-----------------------------------------------------------------------
C
      WRITE(6,90) K
   90 FORMAT(/I6,' TH RANDOM VECTOR REJECTED, GENERATE ANOTHER'/)
C
      KK = KK0
      DO 100 I = 1,N
      KK = KK + 1
  100 Q(KK) = G(I)
      GO TO 20
C
  110 CONTINUE
      RETURN
C-----END OF START------------------------------------------------------
      END
C
C-----START OF DIAGOM----------------------------------------------------
C     DIAGOM CALLS THE EISPACK SUBROUTINES TRED2 AND IMTQL2 TO
C     DIAGONALIZE THE SMALL SYMMETRIC MATRICES GENERATED AT EACH
C     ITERATION OF BLOCK LANCZOS.
C
      SUBROUTINE DIAGOM(MXBLK,MM,TM,KACT,N,Q,E,RESID,RESK,RESN,IND,
     1 KACTN,KM,TD,TOD,NITER,IERR,IWRITE)
C
```

```
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.
      INDEXP = IND
      ICOUNT = 0
      KACT = KACT + 1
      KWANT = KACT
C     CHECK THAT THERE IS ENOUGH ROOM TO ENLARGE THE 1ST QBLOCK
      IF(2*KACT.GT.MXBLK) GO TO 230
      GO TO 80
C
   60 RESID(ICOUNT) = RESN
      INDEXP = IND
      GO TO 80
C
   70 ICOUNT = 1
      RESID(1) = RESN
      INDEXP = IND
C
C-----------------------------------------------------------------------
C     USE EISPACK SUBROUTINES TO DIAGONALIZE THE SMALL TM-MATRIX.
C
   80 CALL TRED2(MXBLK,MM,TM,TD,TOD,TM)
      CALL IMTQL2(MXBLK,MM,TD,TOD,TM,IERR)
C-----------------------------------------------------------------------
      IF(IERR.EQ.0) GO TO 90
      RETURN
C
C    SELECT RELEVANT EIGENVALUES AND EIGENVECTORS OF THE T-MATRIX.
   90 CONTINUE
C
C    IMTQL2 RETURNS EIGENVALUES (AND CORRESPONDING EIGENVECTORS) IN
C    ALGEBRAICALLY-ASCENDING ORDER.  REARRANGE TO DESCENDING ORDER.
C
      DO 100 L=1,MM
      MML = MM-L+1
  100 E(L) = TD(MML)
C
  110 WRITE(6,120) KACT, (E(J), J=1,KACT)
  120 FORMAT(' COMPUTED',I4,' ALGEBRAICALLY-LARGEST EIGENVALUES'/(4E20.1
     12))
C
C     COMPUTE ESTIMATE MAXIMUM EIGENVALUE AND OF SPREAD
      IF(NITER.GT.1) GO TO 140
      EMAX = DMAX1(DABS(E(1)),DABS(E(MM)))
      SPREAD = DABS(E(1) - E(MM))
      EGAP = SPREAD/DFLOAT(N)
      IF(NITER.EQ.1)  WRITE(6,130) EMAX,SPREAD,EGAP
  130 FORMAT(/4X,'ESTIMATED NORM OF MATRIX',4X,'ESTIMATED SPREAD',6X,'SP
     1READ*(SIZE)*(-1)'/E28.4,E20.4,E24.3)
  140 CONTINUE
C
C     COMPUTE RITZ VECTORS
      DO 180 I=1,N
      DO 150 KK=1,KWANT
      TOD(KK)=0.D0
      K = MM - KK + 1
```

```
      IL = - N + I                                                      BLS09240
      DO 150 L = 1,MM                                                   BLS09250
      IL = IL + N                                                       BLS09260
  150 TOD(KK) = TOD(KK) + TM(L,K)*Q(IL)                                 BLS09270
      IKK = -N + I                                                      BLS09280
  160 DO 170 KK=1,KACT                                                  BLS09290
      IKK = IKK + N                                                     BLS09300
  170 Q(IKK)=TOD(KK)                                                    BLS09310
  180 CONTINUE                                                          BLS09320
C                                                                       BLS09330
C     ON FILE 13 SAVE ANY EXTRA VECTORS NO LONGER NEEDED IN 1ST Q-BLOCK BLS09340
      IF(KWANT.EQ.KACT) GO TO 290                                       BLS09350
      K1 = KACT + 1                                                     BLS09360
      K2 = KWANT                                                        BLS09370
      DUMMY = 100.                                                      BLS09380
      DO 190 K = K1,K2                                                  BLS09390
      LINT = (K-1)*N + 1                                                BLS09400
      LFIN = K*N                                                        BLS09410
      WRITE(13,210) E(K),DUMMY,K                                        BLS09420
      WRITE(13,220) (Q(L), L=LINT,LFIN)                                 BLS09430
  190 CONTINUE                                                          BLS09440
      KDELTA = KWANT - KACT                                             BLS09450
      WRITE(13,200) KDELTA                                              BLS09460
  200 FORMAT(/' ABOVE ARE ',I3,' VECTORS STRIPPED FROM A 1ST Q-BLOCK'/  BLS09470
     1' DURING A BLOCK LANZCOS RUN WHICH COULD BE USED AS STARTING VECTOBLS09480
     1RS'/' IN A LATER RUN IF THE USER DECIDES THAT THESE EIGENVALUES SHBLS09490
     1OULD'/' BE COMPUTED AFTER ALL.  FORMAT USED IN THE SAME AS WAS USEBLS09500
     1D'/' IN THE CORRESPONDING BLSTARTV FILE'/)                        BLS09510
  210 FORMAT(/E20.12,E13.4,I6,' = EVAL,DUMMY,EVAL NUMBER,EVEC='/)       BLS09520
  220 FORMAT(4E20.12)                                                   BLS09530
      GO TO 290                                                         BLS09540
C                                                                       BLS09550
C     DEFAULT, SIZE OF 1ST Q-BLOCK TOO LARGE FOR MXBLK                  BLS09560
  230 IWRITE = -1000                                                    BLS09570
      WRITE(6,240) KACT,MXBLK                                           BLS09580
      WRITE(15,240) KACT,MXBLK                                          BLS09590
  240 FORMAT(//' BLOCK LANCZOS PROCEDURE TRIED TO INCREASE THE SIZE OF 1BLS09600
     1ST QBLOCK'/' TO ',I3,' BUT THIS IS NOT FEASIBLE BECAUSE TWICE THISBLS09610
     1 SIZE'/' IS G.T. MXBLK WHICH EQUALS ',I4/' USER CAN RERUN PROGRAM BLS09620
     1WITH LARGER MXBLK'/)                                              BLS09630
      GO TO 290                                                         BLS09640
C                                                                       BLS09650
C     DEFAULT, CONVERGENCE RATE IS TOO SLOW                             BLS09660
  250 IWRITE = -1000                                                    BLS09670
      WRITE(6,260) NITER,RATIO,FRACT                                    BLS09680
      WRITE(15,260) NITER,RATIO,FRACT                                   BLS09690
  260 FORMAT(//' ON ITERATION ',I3,' CONVERGENCE APPEARS TO BE STAGNATEDBLS09700
     1'/' RATIO OF SQUARE OF CURRENT KM-TH RESIDUAL TO CORRESPONDING SQUBLS09710
     1ARE'/' 10 ITERATIONS EARLIER IS ',E10.3,' COMPARED TO '/          BLS09720
     1' USER-SPECIFIED RATIO ',E10.3,'. THEREFORE, PROGRAM TERMINATES'/'BLS09730
     1 USER SHOULD LOOK AT THE OUTPUT. IF CONVERGENCE HAS STAGNATED, USEBLS09740
     1R'/' CAN EITHER INCREASE KACT OR KMAX OR RESET THE STAGNATION PARABLS09750
     1METERS'/' NSTAG AND FRACT, AND RESTART THE BLOCK PROCEDURE USING TBLS09760
     1HE'/' CURRENT EIGENVECTOR APPROXIMATIONS AS STARTING VECTORS'/)   BLS09770
      GO TO 290                                                         BLS09780
```

```
C                                                                       BLS09790
  270 IWRITE = -1000                                                    BLS09800
      WRITE(6,280)                                                      BLS09810
      WRITE(15,280)                                                     BLS09820
  280 FORMAT(//' SOMETHING IS SERIOUSLY WRONG.  NUMBER OF ITERATIONS IS BLS09830
     1EXCESSIVE'/' PROGRAM TERMINATES FOR USER TO DECIDE WHAT TO DO'/'  BLS09840
     1ALTERNATIVES INCLUDE INCREASING KACT OR KMAX OR BOTH, AND RESTARTIBLS09850
     1NG'/' USING THE CURRENT APPROXIMATIONS AS STARTING VECTORS'//)    BLS09860
C                                                                       BLS09870
  290 CONTINUE                                                          BLS09880
      RETURN                                                            BLS09890
C-----END OF DIAGOM-----------------------------------------------------BLS09900
      END                                                               BLS09910
C-----LPERM PERMUTES VECTORS-----------------------------------------   BLS09920
C                                                                       BLS09930
      SUBROUTINE LPERM(W,U,IPERM)                                       BLS09940
C                                                                       BLS09950
C-----------------------------------------------------------------------BLS09960
      DOUBLE PRECISION  U(1),W(1)                                       BLS09970
      INTEGER  IPR(1),IPT(1)                                            BLS09980
C-----------------------------------------------------------------------BLS09990
C     SUBROUTINE HAS 2 BRANCHES:  IPERM = 1,  CALCULATES                BLS10000
C     U = P*W  WHERE  P IS THE PERMUTATION REPRESENTED BY IPR           BLS10010
C     LET J = IPR(K) THEN U(K) = W(J), K = 1,N. WE SET W(K)=U(K), K=1,N BLS10020
C     IPERM = 2, USING THE PERMUTATION IPT (P-TRANSPOSE) U = P'*W, W=U  BLS10030
C     LET J = IPT(K) THEN U(K) = W(J), K=1,N. WE SET W(K) = U(K), K=1,N BLS10040
C-----------------------------------------------------------------------BLS10050
C                                                                       BLS10060
      GO TO 3                                                           BLS10070
C-----------------------------------------------------------------------BLS10080
      ENTRY LPERME(IPR,IPT,N)                                           BLS10090
      GO TO 4                                                           BLS10100
C-----------------------------------------------------------------------BLS10110
C                                                                       BLS10120
    3 CONTINUE                                                          BLS10130
      IF(IPERM.EQ.2) GO TO 10                                           BLS10140
C     IPERM = 1                                                         BLS10150
      DO 20 K = 1,N                                                     BLS10160
      J = IPR(K)                                                        BLS10170
   20 U(K) = W(J)                                                       BLS10180
      DO 30 K = 1,N                                                     BLS10190
   30 W(K) = U(K)                                                       BLS10200
      GO TO 60                                                          BLS10210
C     IPERM = 2                                                         BLS10220
   10 DO 40  K = 1,N                                                    BLS10230
      J = IPT(K)                                                        BLS10240
   40 U(K) = W(J)                                                       BLS10250
      DO 50 K = 1,N                                                     BLS10260
   50 W(K) = U(K)                                                       BLS10270
   60 CONTINUE                                                          BLS10280
C                                                                       BLS10290
C                                                                       BLS10300
C-----END OF LPERM-------------------------------------------------     BLS10310
    4 RETURN                                                            BLS10320
      END                                                               BLS10330
```

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
C     EIGENVECTORS.  THE MATRIX B = S0*P*A*P' + SHIFT*I WHERE THE
C     SCALE S0 AND SHIFT ARE READ IN THIS PROGRAM, AND THE
C     PERMUTATION P IS DEFINED IN THE CORRESPONDING USPEC SUBROUTINE.
C     THE PROGRAM ASSUMES THAT THE FACTORIZATION READ IN USPEC
C     CORRESPONDS TO THE S0, SHIFT AND PERMUTATION READ IN.  THE S0
C     AND SHIFT ARE CHOSEN SO THAT THE DESIRED EIGENVALUES ARE AT
C     THE EXTREME OF THE SPECTRUM OF B-INVERSE.
C
C     THIS IS AN ITERATIVE 'BLOCK' LANCZOS PROCEDURE FOR WHICH ON
C     EVERY ITERATION, THE 2ND AND SUCCEEDING BLOCKS CONTAIN ONLY ONE
C     VECTOR WHICH IS SELECTED ON THE BASIS OF ITS EXPECTED INFLUENCE
C     ON THE CONVERGENCE.   Q-BLOCKS GENERATED ON A GIVEN ITERATION
C     ARE REORTHOGONALIZED ONLY W.R.T. THOSE VECTORS IN THE FIRST
C     Q-BLOCK WHICH ARE NOT ALLOWED TO GENERATE DESCENDANTS ON
C     THAT ITERATION.
C
C     PFORT VERIFIER IDENTIFIED THE FOLLOWING NONPORTABLE CONSTRUCTIONS:
C     1.  DATA MACHEP DEFINITION
C     2.  FORMAT (20A4) USED FOR READING EXPLANATORY COMMENTS.
C     3.  FREE FORMAT (5,*), USED FOR PARAMETER INPUT FROM FILE 5.
C     4.  COMMON/LOOPS/ AS CONSTRUCTED IS NOT PORTABLE
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION Q(44000),E(50),TM(2500),TOD(50),TD(50),EPSM,NNZ
      DOUBLE PRECISION SM(100),ERRMAX,SPREC,MACHEP,AVER,RELTOL,ERRMAN
      DOUBLE PRECISION  EVAL, RESIDL(100), RESIDK(100), RESID, FRACT
      DOUBLE PRECISION S0,SHIFT
      REAL EXPLAN(20),G(2000)
      INTEGER   DIR(2,100),DESC(100),LEFT(100),XLFT(100)
      INTEGER SEED,OFLAG,EFLAG
      COMMON/LOOPS/MAXIT,ITER
      COMMON /RANDOM/SEED
      COMMON/FLAGS/EFLAG,OFLAG
      DOUBLE PRECISION  DABS, DFLOAT
C-----------------------------------------------------------------------
      EXTERNAL BLSOLV
      DATA MACHEP/Z3410000000000000/
C-----------------------------------------------------------------------
C
C     ARRAYS MUST DIMENSIONED AS FOLLOWS:
C
C     1.  Q:   >=  KMAX*N
C     2.  G:   >= N
C     3.  E:   >=  MXBLK
C     4.  TM:   >=  MXBLK**2
C     5.  TOD, TD, SM, DESC, LEFT, XLFT:   >= MXBLK
C     6.  DIR:  ROW DIMENSION = 2;   COLUMN DIMENSION >= MXBLK
C     7.  RESIDL, RESIDK:  >= MAXIMUM NUMBER OF ITERATIONS ALLOWED.
C         PROGRAM CURRENTLY TERMINATES IF MORE THAN 100 ITERATIONS
C         ARE REQUESTED.  USED TO MONITOR CONVERGENCE. SEE SUBROUTINE
C         DIAGOM.
C     8.  EXPLAN:  DIMENSION = 20.
C
C-----------------------------------------------------------------------
```

```
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          BLI01550
C     STARTING BLOCK (KACT);  NUMBER OF STARTING VECTORS SUPPLIED (KSET)BLI01560
C     SEE BLOCK LANCZOS HEADER FOR COMMENTS REGARDING THE SIZE OF KACT. BLI01570
      READ(5,20) EXPLAN                                                 BLI01580
      READ(5,*) KMAX,KACT,KSET                                          BLI01590
C                                                                       BLI01600
C     SPECIFY NUMBER (KM) OF EXTREME EIGENVALUES AND EIGENVECTORS       BLI01610
C     OF B-INVERSE TO BE COMPUTED.  THE BLOCK PROCEDURE WORKS WITH THE  BLI01620
C     INVERSE OF THE MATRIX B = S0*P*A*P' + SHIFT*I, USING A            BLI01630
C     FACTORIZATION OF B.  TO INDICATE THAT THE ALGEBRAICALLY-          BLI01640
C     SMALLEST EIGENVALUES OF B-INVERSE ARE BEING COMPUTED SET KM < 0.  BLI01650
C     IF KM < 0, THE PROGRAM ASSUMES THAT BLSOLV SUBROUTINE WHICH       BLI01660
C     THE USER HAS PROVIDED IS COMPUTING -(B-INVERSE)*X                 BLI01670
C     INSTEAD OF (B-INVERSE)*X AND INTERNALLY IT COMPUTES THE |KM|      BLI01680
C     ALGEBRAICALLY-LARGEST EIGENVALUES OF -(B-INVERSE).                BLI01690
      READ(5,20) EXPLAN                                                 BLI01700
      READ(5,*) KM                                                      BLI01710
      IF(KM.EQ.0) GO TO 540                                             BLI01720
      KML = IABS(KM)                                                    BLI01730
C                                                                       BLI01740
C     STAGNATION OF CONVERGENCE OF THE KM-TH EIGENVALUE WILL BE         BLI01750
C     TESTED AFTER NSTAG ITERATIONS.  CONVERGENCE WILL BE SAID TO       BLI01760
C     HAVE STAGNATED IF THE RATIO OF THE SQUARE OF THE CURRENT KM-TH    BLI01770
C     RESIDUAL TO THE SQUARE OF THE CORRESPONDING RESIDUAL OBTAINED     BLI01780
C     10 ITERATIONS EARLIER IS GREATER THAN FRACT.  NSTAG SHOULD BE     BLI01790
C     >= 25.  FRACT WAS SET EQUAL TO .01 IN THE TESTS.                  BLI01800
      READ(5,20) EXPLAN                                                 BLI01810
      READ(5,*) NSTAG, FRACT                                            BLI01820
C                                                                       BLI01830
C     READ IN THE RELATIVE TOLERANCE (RELTOL) USED TO DETERMINE A       BLI01840
C     CONVERGENCE CRITERION FOR PHASE 2, AND THE MAXIMUM NUMBER (MAXIT2)BLI01850
C     OF CALLS TO SUBROUTINE BLSOLV ALLOWED IN PHASE 2.                 BLI01860
      READ(5,20) EXPLAN                                                 BLI01870
      IF(EFLAG.EQ.1)  READ(5,*)  RELTOL, MAXIT2                         BLI01880
C                                                                       BLI01890
C     CONSISTENCY CHECKS                                                BLI01900
C     PROCEDURE REQUIRES ENOUGH ROOM IN THE Q-ARRAY FOR AT LEAST 2      BLI01910
C     BLOCKS OF SIZE KACT PLUS A WORKING VECTOR OF LENGTH N.            BLI01920
      MXBLK = KMAX -1                                                   BLI01930
      MXBLK2 = MXBLK*MXBLK                                              BLI01940
      IF(MDIMTM.LT.MXBLK2) GO TO 520                                    BLI01950
      NKMAX = N*KMAX                                                    BLI01960
      IF(MDIMQ.LT.NKMAX)   GO TO 560                                    BLI01970
      IF(KML.GT.KACT) GO TO 420                                         BLI01980
      IF(MXBLK.GT.N)  GO TO 440                                         BLI01990
      IF(2*KACT.GT.MXBLK)  GO TO 500                                    BLI02000
C                                                                       BLI02010
C-----------------------------------------------------------------------BLI02020
C     DEFINE AND INITIALIZE THE ARRAYS NEEDED TO DEFINE THE             BLI02030
C     FACTORIZATION OF THE B-MATRIX.  PASS THE STORAGE LOCATIONS        BLI02040
C     OF THESE ARRAYS TO THE SUBROUTINE BLSOLV.                         BLI02050
C                                                                       BLI02060
      CALL USPEC(N,MATNO,NNZ,AVER)                                      BLI02070
C                                                                       BLI02080
C-----------------------------------------------------------------------BLI02090
```

```
C     MASK OVERFLOW AND UNDERFLOW
      CALL MASK
C
C-----------------------------------------------------------------------
C     ARE THERE STARTING VECTORS TO READ IN FROM FILE 10 (KSET.NE.0) ?
      IF(KSET.EQ.0) GO TO 70
C
      READ(10,30) NOLD,KACT
   30 FORMAT(I6,I4)
      IF(NOLD.NE.N.OR.KSET.GT.KACT) GO TO 460
      DO 50 J=1,KSET
      READ(10,20) EXPLAN
      READ(10,40) EVAL,RESID
   40 FORMAT(E20.12,E13.4)
      READ(10,20) EXPLAN
      LINT= (J-1)*N + 1
      LFIN = J*N
   50 READ(10,60) (Q(JL), JL = LINT,LFIN)
   60 FORMAT(4E20.12)
C
   70 CONTINUE
C
C     WRITE TO A SUMMARY OF THE PARAMETERS FOR THIS RUN TO FILE 6
C
      MXBLK = KMAX - 1
      WRITE(6,80)  N, NNZ, AVER, MATNO
   80 FORMAT(/4X,'ORDER OF B-MATRIX ',5X,'AVERAGE NUMBER NONZEROES PER R
     1OW IN FACTOR'/
     1I15,E47.4/3X,'CRUDE ESTIMATE OF SIZE NONZERO ENTRIES',5X,'MATRIX I
     1D'/E31.4,I21/)
C
      WRITE(6,90)  S0, SHIFT
   90 FORMAT(/4X,'SCALE USED ON A-MATRIX',5X,'SHIFT USED ON A-MATRIX'/
     1E26.4,E27.4/)
C
      WRITE(6,100) MDIMQ, MDIMTM
  100 FORMAT(/18X,'USER-SPECIFIED'/2X,'MAX. DIMENSION Q-ARRAY',4X,'MAX.
     1DIMENSION TM-ARRAY'/I16,I26/)
C
      WRITE(6,110)  OFLAG, EFLAG
  110 FORMAT(/4X,'OFLAG',4X,'EFLAG'/I8,I9/)
C
      IF(OFLAG.EQ.1)  WRITE(6,120)  SPREC
  120 FORMAT(/4X,'ORTHOGONALITY TEST TOLERANCE'/E25.2)
C
      IF(EFLAG.EQ.1)  WRITE(6,130) MAXIT,RELTOL,MAXIT2
  130 FORMAT(/4X,' MAXIT ',8X,'  RELTOL  ',6X,' MAXIT2 '/I10,E20.6,I12/)
      IF(EFLAG.EQ.0)  WRITE(6,140) MAXIT
  140 FORMAT(/4X,' MAXIT '/I10/)
C
      WRITE(6,150)  SEED
  150 FORMAT(/' SEED FOR RANDOM NUMBER GENERATOR'/I24/)
C
      IF(KM.GT.0)  WRITE(6,160) KML
  160 FORMAT(/' COMPUTE THE',I3,' ALGEBRAICALLY-LARGEST EIGENVALUES AND
```

```
     1CORRESPONDING VECTORS'/' OF THE INVERSE OF B =  (S0*P*A*P-TRANS + BLI02650
     1HIFT*I)'/)                                                        BLI02660
      IF(KM.LT.0)  WRITE(6,170) KML                                     BLI02670
  170 FORMAT(/' COMPUTE THE',I3,' ALGEBRAICALLY-SMALLEST EIGENVALUES ANDBLI02680
     1 CORRESPONDING VECTORS'/' OF THE INVERSE OF THE MATRIX B = (S0*P*ABLI02690
     1*P-TRANS + SHIFT*I).'/' PROGRAM ASSUMES THAT USER IS PROVIDING -(BBLI02700
     1-INVERSE)*X INSTEAD OF (B-INVERSE)*X'/' AND COMPUTES THE ALGEBRAICBLI02710
     1ALLY-LARGEST EIGENVALUES OF -(B-INVERSE).'/' HOWEVER ON EXIT, FILEBLI02720
     1 15 CONTAINS THE ALGEBRAICALLY-SMALLEST EIGENVALUES'/' OF B-INVERSBLI02730
     1E, THE CORRESPONDING EIGENVALUES OF THE ORIGINAL A-MATRIX'/' AND TBLI02740
     1HE CORRESPONDING EIGENVECTORS OF A.'/)                            BLI02750
C                                                                       BLI02760
C     NOTE THAT THE ESTIMATE FOR AVER IN THE INVERSE CASE IS VERY CRUDE BLI02770
C     COMPUTE PHASE 1 CONVERGENCE TOLERANCE                             BLI02780
      IF(AVER.GE.1.)                                                    BLI02790
     1ERRMAX = 2.D0*DFLOAT(N+1000)*NNZ*AVER*MACHEP                      BLI02800
      IF(AVER.LT.1.)                                                    BLI02810
     1ERRMAX = 2.D0*DFLOAT(N+1000)*NNZ*AVER**2*MACHEP                   BLI02820
C                                                                       BLI02830
      WRITE(6,180) KACT,MXBLK,KSET                                      BLI02840
  180 FORMAT(/' ON INITIAL ITERATIONS, THE FIRST BLOCK CONTAINS ',I3,' VBLI02850
     1ECTORS'/' HOWEVER THE SIZE OF THE FIRST BLOCK MAY CHANGE AS THE ITBLI02860
     1ERATIONS PROCEED'/' THE MAXIMUM SIZE T-MATRIX THAT CAN BE GENERATEBLI02870
     1D IS ',I4/' THE USER SUPPLIED ',I3,' STARTING VECTORS'/)          BLI02880
C                                                                       BLI02890
      WRITE(6,190)                                                      BLI02900
  190 FORMAT(/' ITERATIVE PROCEDURE'/' PROCEDURE MONITORS THE SIZES OF TBLI02910
     1HE NORM(GRADIENTS)**2 ON EACH'/' ITERATION.  CONVERGENCE IS SAID  BLI02920
     1TO HAVE OCCURRED WHEN ALL'/' RELEVANT (NORMS)**2 ARE LESS THAN ERRBLI02930
     1MAX',E10.3/' PHASE 1 ERRMAX MAY YIELD SOMEWHAT LESS THAN SINGLE PRBLI02940
     1ECISION ACCURACY.'/'  PHASE 2 REFINES THE VECTORS OBTAINED ON PHASBLI02950
     1E 1, ACCORDING TO'/' THE ACCURACY SPECIFIED BY THE USER'/)        BLI02960
C                                                                       BLI02970
      WRITE(6,200) ERRMAX                                               BLI02980
  200 FORMAT(//' PHASE 1 CONVERGENCE CRITERION, ERRMAX '/E22.3/)        BLI02990
C                                                                       BLI03000
C-----------------------------------------------------------------------BLI03010
C     PASS STORAGE LOCATIONS OF VARIOUS ARRAYS TO LANCZS AND LANCI1     BLI03020
C     SUBROUTINES                                                       BLI03030
C                                                                       BLI03040
      CALL LANZP(DIR,DESC,SM,TM,TOD,TD,G,XLFT,LEFT,SPREC)               BLI03050
      CALL LANCP1(DIR,DESC,TM,SM,XLFT,LEFT)                             BLI03060
C                                                                       BLI03070
C-----------------------------------------------------------------------BLI03080
C                                                                       BLI03090
C     ENTER PHASE 1 OF BLOCK LANCZOS PROCEDURE.  BLOCK PROCEDURE        BLI03100
C     HAS 2 POSSIBLE PHASES.  USER SPECIFIES PHASE 1 ONLY OR PHASE 1    BLI03110
C     AND PHASE 2 BY SETTING EFLAG = 0 OR 1, RESPECTIVELY. PHASE 1      BLI03120
C     COMPUTES VECTORS THAT ARE USUALLY ACCURATE TO SINGLE PRECISION.   BLI03130
C     PHASE 2 TAKES THE VECTORS OBTAINED IN PHASE 1 AND REFINES THEM.   BLI03140
C     THE USER SPECIFIES THE DEGREE OF REFINEMENT DESIRED BY SELECTING  BLI03150
C     THE VALUE OF RELTOL AND MAXIT2.  BOTH PHASES SHOULD BE USED.      BLI03160
      IPHASE = 1                                                        BLI03170
      NITER = 0                                                         BLI03180
  210 ITER = 0                                                          BLI03190
```

```
      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.
C     USER CONTROLS THE SIZE OF THE ERROR TOLERANCE BY SPECIFYING
C     THE PARAMETER RELTOL.
C
      IPHASE = 2
      MAXIT = MAXIT2
      KSET = KACT
C
C     ERROR TOLERANCE USES THE CONVERGED EIGENVALUE LARGEST IN
C     MAGNITUDE.
      TD(1) = DABS(E(1))
      IF(KML.EQ.1) GO TO 400
      DO 390 J = 2,KML
  390 IF(DABS(E(J)).GT.TD(1))    TD(1) = DABS(E(J))
  400 TD(1) = DMAX1(TD(1),1.D0)
      ERRMAN = RELTOL**2 * TD(1)**2
      IF(ERRMAN.GE.ERRMAX) GO TO 480
      ERRMAX = ERRMAN
C
      WRITE(6,410) ERRMAX, MAXIT2
  410 FORMAT(//' ENTER PHASE 2 OF COMPUTATION'/' CONVERGENCE CRITERION I
     1S REDUCED TO ',E13.4/' NO MORE THAN ',I5,' CALLS TO SUBROUTINE BLS
     1OLV WILL BE ALLOWED.'/' PROGRAM WILL TERMINATE IF BLOCK ORTHGONALI
     1TY PROBLEMS MATERIALIZE'/)
C
      GO TO 210
C
C     INCONSISTENCIES IN THE DATA
C
  420 WRITE(6,430) KM,KACT
  430 FORMAT(/' PROGRAM TERMINATES BECAUSE THE NUMBER OF EIGENELEMENTS
     1REQUESTED, KM =',I3/'  IS LARGER THAN THE SIZE OF THE FIRST Q BLOC
     1K, KACT =',I3,' SPECIFIED'/'  USER MUST RESET KM OR KACT'/)
      GO TO 580
C
  440 WRITE(6,450) KMAX,N
  450 FORMAT(/' PROGRAM TERMINATES BECAUSE KMAX = ',I5,' IS TOO LARGE FO
     1R THE SIZE, N = ',I5,', OF THE GIVEN MATRIX'/'  USER MUST DECREASE
     1THE SIZE OF KMAX.'/)
      GO TO 580
C
  460 WRITE(6,470) NOLD,N,KACT,KSET
  470 FORMAT(/' PROGRAM TERMINATES BECAUSE FAULT OCCURRED IN READING IN
     1THE EIGENVECTOR APPROXIMATIONS'/' EITHER THE SIZE MATRIX SPECIFIED
     1ON THE EIGENVECTOR FILE' ,I6/'  DID NOT MATCH THE SIZE SPECIFIED '
     1,I5,' IN THE PROGRAM OR THE NUMBER'/'    OF VECTORS IN FILE 10 = '
     1,I4,' IS LESS THAN THE NUMBER ',I3/'   USER SAID WERE THERE'/)
      GO TO 580
C
  480 WRITE(6,490)  ERRMAN, ERRMAX
  490 FORMAT(/' COMPUTED PHASE 2 CONVERGENCE CRITERION ',E13.4/'  IS LAR
     1GER THAN PHASE 1 CRITERION ',E13.4/'  SO PROGRAM TERMINATES'/)
      GO TO 580
C
  500 WRITE(6,510)  KACT,MXBLK
```

```
  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 = 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).
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
C-----------------------------------------------------------------------
    3 CONTINUE
      ITER = ITER + 1
      ZERO = 0.0D0
      ONE  = 1.0D0
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 10 K = 1,N
   10 U(K) = V(K)
      DO 30 K = 1,N
      TEMP = U(K)/BD(K)
      U(K) = TEMP
      IF (KCOL(K).EQ.0.OR.K.EQ.N) GO TO 30
      KF = KL + 1
      KL = KL + KCOL(K)
      DO 20 KK = KF,KL
      KR = KROW(KK)
   20 U(KR) = U(KR) - TEMP*BSD(KK)
   30 CONTINUE
      NP1 = N+1
      KF = NZT + 1
      DO 50 K = 1,N
      L = NP1 - K
      TEMP = U(L)
      IF (KCOL(L).EQ.0.OR.L.EQ.N) GO TO 50
      KL = KF - 1
      KF = KF - KCOL(L)
      DO 40 LL = KF,KL
      LR = KROW(LL)
   40 TEMP = TEMP - BSD(LL)*U(LR)
   50 U(L) = TEMP/BD(L)
   60 CONTINUE
C
    4 RETURN
C
C-----END OF BLSOLV-----------------------------------------------------
      END
C
C-----SUBROUTINES FOR DIAGONAL TEST MATRICES----------------------------
C     BLSOLV AND USPEC SUBROUTINES FOR DIAGONAL TEST MATRICES
C
C-------BLSOLV DIAGONAL TEST MATRIX-------------------------------------
C
C     SUBROUTINE DBSOLV(V,U)
      SUBROUTINE BLSOLV(V,U)
C
C-----------------------------------------------------------------------
      DOUBLE PRECISION  V(1),U(1),D(1)
      COMMON/LOOPS/MAXIT,ITER
C-----------------------------------------------------------------------
      GO TO 3
C-----------------------------------------------------------------------
C     BELOW ENTRY IS FOR A DIAGONAL TEST MATRIX
```

```
      ENTRY DSOLVE(D,N)                                                 BLI02110
      GO TO 4                                                           BLI02120
C-----------------------------------------------------------------------BLI02130
    3 CONTINUE                                                          BLI02140
      ITER = ITER + 1                                                   BLI02150
   10 DO 20 I=1,N                                                       BLI02160
   20 U(I)= V(I)/D(I)                                                   BLI02170
C  20 U(I)= -V(I)/D(I)                                                  BLI02180
C                                                                       BLI02190
   30 CONTINUE                                                          BLI02200
    4 RETURN                                                            BLI02210
C-----END OF 'DIAGONAL' TEST MATRIX  BLSOLV-----------------------------BLI02220
      END                                                               BLI02230
C                                                                       BLI02240
C-----START OF USPEC FOR DIAGONAL TEST MATRIX---------------------------BLI02250
C                                                                       BLI02260
      SUBROUTINE USPEC(N,MATNO,NNZ,AVER)                                BLI02270
C     SUBROUTINE DUSPEC(N,MATNO,NNZ,AVER)                               BLI02280
C                                                                       BLI02290
C-----------------------------------------------------------------------BLI02300
      DOUBLE PRECISION  D(1000),DI(1000),SHIFT,SPACE,NNZ,AVER           BLI02310
      DOUBLE PRECISION  DABS, DFLOAT                                    BLI02320
      REAL  EXPLAN(20)                                                  BLI02330
C-----------------------------------------------------------------------BLI02340
C                                                                       BLI02350
      READ(7,10) EXPLAN                                                 BLI02360
   10 FORMAT(20A4)                                                      BLI02370
      READ(7,*) NOLD,NUNIF,SPACE,D(1),SHIFT                             BLI02380
      NNUNIF = NOLD - NUNIF                                             BLI02390
      WRITE(6,20) NOLD,SPACE,NNUNIF,D(1),SHIFT                          BLI02400
   20 FORMAT(/' DIAGONAL TEST MATRIX, SIZE = ',I4/' IS THE INVERSE OF MABLI02410
     1TRIX WITH MOST ENTRIES',E10.3/' UNITS APART AND WITH ',I3,' ENTRIEBLI02420
     1S IRREGULARLY SPACED'/' FIRST ENTRY WAS ',E13.4,' SHIFT = ',E10.3 BLI02430
     1/)                                                                BLI02440
C                                                                       BLI02450
      IF(N.NE.NOLD) GO TO 120                                           BLI02460
C     COMPUTE THE UNIFORM PORTION OF THE SPECTRUM                       BLI02470
      DO 30 J=2,NUNIF                                                   BLI02480
   30 D(J) = D(1) - DFLOAT(J-1)*SPACE                                   BLI02490
      NUNIF1=NUNIF + 1                                                  BLI02500
      READ(7,10)  EXPLAN                                                BLI02510
      DO 40 J=NUNIF1,N                                                  BLI02520
   40 READ(7,*) D(J)                                                    BLI02530
      NB = NUNIF - 2                                                    BLI02540
C                                                                       BLI02550
      IF(SHIFT.EQ.0.) GO TO 60                                          BLI02560
      DO 50 J=1,N                                                       BLI02570
   50 D(J) = D(J) + SHIFT                                               BLI02580
C                                                                       BLI02590
C     COMPUTE EIGENVALUES OF INVERSE FOR PRINTOUT ONLY                  BLI02600
   60 DO 70 J = 1,N                                                     BLI02610
   70 DI(J) = 1.D0/D(J)                                                 BLI02620
      WRITE(6,80) (J,DI(J), J=1,N )                                     BLI02630
   80 FORMAT(/' INVERSE BLOCK LANCZOS TEST, LANCZS USES INVERSE OF GIVENBLI02640
     1MATRIX'/' ENTRIES OF INVERSE OF DIAGONAL TEST MATRIX = '/(I4,E20.1BLI02650
```

```
     12,I4,E20.12,I4,E20.12))                                           BLI02660
C                                                                       BLI02670
C     DIAGONAL GENERATION COMPLETE                                      BLI02680
C                                                                       BLI02690
C     COMPUTE NNZ AND AVER                                              BLI02700
      NNZ = 1.D0                                                        BLI02710
      AVER = 0.D0                                                       BLI02720
      DO 90 K = 1,N                                                     BLI02730
   90 AVER = AVER + DABS(DI(K))                                         BLI02740
      AVER = AVER/DFLOAT(N)                                             BLI02750
      AVER = 1.D0/AVER                                                  BLI02760
C                                                                       BLI02770
C     COMPUTE THE GAPS                                                  BLI02780
      N1 = N-1                                                          BLI02790
      DO 100 K = 1,N1                                                   BLI02800
  100 DI(K) = DI(K+1) - DI(K)                                           BLI02810
      WRITE(6,110) (K,DI(K), K=1,N1)                                    BLI02820
  110 FORMAT(/' GAPS BETWEEN EIGENVALUES'/(I4,E13.4,I4,E13.4,I4,E13.4,I4BLI02830
     1,E13.4))                                                          BLI02840
C                                                                       BLI02850
C-----------------------------------------------------------------------BLI02860
C     PASS STORAGE LOCATIONS OF D AND N TO DSOLV SUBROUTINE             BLI02870
      CALL DSOLVE(D,N)                                                  BLI02880
C-----------------------------------------------------------------------BLI02890
C                                                                       BLI02900
      RETURN                                                            BLI02910
  120 WRITE(6,130) NOLD,N                                               BLI02920
  130 FORMAT(' PROGRAM TERMINATES BECAUSE NOLD = ',I5,'DOES NOT EQUAL N BLI02930
     1 =',I5)                                                           BLI02940
C-----END OF USPEC SUBROUTINE FOR 'DIAGONAL' TEST MATRICES--------------BLI02950
      STOP                                                              BLI02960
      END                                                               BLI02970
```

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
-----------------------------------------------------------------------
```