```
!
! note AMIN1, AMAX1 was changed to DMIN1,DMAX1 for CRAY and NECSX5
!*
!*
!*         *****************************
!*         *                           *
!*         *    UTILITY SUBROUTINES    *
!*         *  FROM STANDARD LIBRARIES  *
!*         *  FOR USE WITH PW PROGRAM  *
!*         *                           *
!*         *****************************
!*
!*
      subroutine diagc(ei,mtxd,neig,                                   &
     & hamkr,hamki,zvecr,zveci,                                        &
     & irow,ind,tau,d,sd,sd2,rv1,rv2,rv3,rv4,rv5,                      &
     & mxddim)
!*
!     call diagc(ei,nn,nbandi,
!    1   qdr,qdi,vecr,veci,
!    2   iwrk31,iwrk32,wrk11,wrk13,wrk14,wrk15,
!    3   wrk21,wrk22,wrk23,wrk24,wrk25,mxdiis)
!*
!
!     subroutine diagonalizes the hermitian hamiltonian
!     stored in lower triangular form.
!     adapted from sverre froyen plane wave program
!
!
!     input:
!     mtxd        dimension of the hamiltonian.
!     neig        number of eigenvectors required
!     hamkr(j)    real part of the hamiltonian. (Rydberg)
!     hamki(j)    imaginary part of the hamiltonian. (Rydberg)
!     mxddim      array dimension of hamiltonian rows
!
!     output:
!     ei(i)       eigenvalue no. i. (Rydberg)
!     zvecr(i)    real part of the components of the eigenvectors
!                 written successively
!     zveci(i)    imaginary part of the components of the eigenvectors
!                 written successively
!
!     work arrays: irow,tau,d,sd,sd2,ind,rv1,rv2,rv3,rv4,rv5
!
!     implicit double precision (a-h,o-z)
!
      implicit real*8 (a-h,o-z)
      parameter (zero = 0.0d0, um = 1.0d0)
!
      dimension ei(mtxd)
      dimension hamkr((mtxd*mtxd+mtxd)/2),hamki((mtxd*mtxd+mtxd)/2)
      dimension zvecr(mtxd*mtxd),zveci(mtxd*mtxd)
!
!
      dimension irow(mxddim),tau(2,mxddim),                            &
     & d(mxddim),sd(mxddim),sd2(mxddim),ind(mxddim),                   &
     & rv1(mxddim),rv2(mxddim),rv3(mxddim),rv4(mxddim),rv5(mxddim)
!
!     call diagonalization routines
!
!
       if(mtxd .gt. mxddim)  then
       write(6,1234) mxddim,mtxd
                             stop
                             endif
 1234  format('mxddim is too small, mxddim=,mtxd= ',2i5)
!
      iemerg=0
```



```
!
      do 10 i=1,mtxd
        irow(i) = (i*i - i)/2
   10 continue
!
      call htridi(mtxd,irow,hamkr,hamki,d,sd,sd2,tau)
!
!     diagonalizes tridiagonal matrix
!
      if(neig .lt. mtxd/4) then
   15   continue
        eps1 = -um
        call tridib(mtxd,eps1,d,sd,sd2,bl,bu,1,neig,ei,ind,            &
     &  ierr,rv4,rv5)
!
!       check for missing degenerate eigenvalue
!       if so increase mtxd1 and recompute
!
        if (ierr .ne. 0 .and. mtxd .ne. neig) then
          neig = neig + 1
          goto 15
        endif
      else
        call imtqlv(mtxd,d,sd,sd2,ei,ind,ierr,rv5)
      endif
      if (ierr .ne. 0) then
      iemerg=1
      write(6,*) ' warning in diagc 220 '
!     call warnd(220,xdum,idum)
                       endif
!
!     compute eigenvector(s)
!
      call tinvit(mtxd,mtxd,d,sd,sd2,neig,ei,ind,                      &
     & zvecr,ierr,rv1,rv2,rv3,rv4,rv5)
!
      if (ierr .ne. 0) then
      iemerg=1
      write(6,*) ' warning in diagc 221 '
!     call warnd(221,xdum,ierr)
                       endif
!
      call htribk(mtxd,irow,hamkr,hamki,tau,neig,zvecr,zveci)
      if(iemerg .eq. 1) irow(1)=-1
      return
     END
!*
!*
!*
     SUBROUTINE TQL2(NM,N,D,E,Z,IERR)
!
     IMPLICIT REAL*8 (A-H,O-Z)
     INTEGER I,J,K,L,M,N,II,L1,L2,NM,MML,IERR
     DIMENSION D(N),E(N),Z(NM,N)
!!   REAL C,C2,C3,DL1,EL1,F,G,H,P,R,S,S2,TST1,TST2,PYTHAG
!
!    THIS SUBROUTINE IS A TRANSLATION OF THE ALGOL PROCEDURE TQL2,
!    NUM. MATH. 11, 293-306(1968) BY BOWDLER, MARTIN, REINSCH, AND
!    WILKINSON.
!    HANDBOOK FOR AUTO. COMP., VOL.II-LINEAR ALGEBRA, 227-240(1971).
!
!    THIS SUBROUTINE FINDS THE EIGENVALUES AND EIGENVECTORS
!    OF A SYMMETRIC TRIDIAGONAL MATRIX BY THE QL METHOD.
!    THE EIGENVECTORS OF A FULL SYMMETRIC MATRIX CAN ALSO
!    BE FOUND IF  TRED2  HAS BEEN USED TO REDUCE THIS
!    FULL MATRIX TO TRIDIAGONAL FORM.
!
!    ON INPUT
!
```

```
!         NM MUST BE SET TO THE ROW DIMENSION OF TWO-DIMENSIONAL
!           ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
!           DIMENSION STATEMENT.
!
!         N IS THE ORDER OF THE MATRIX.
!
!         D CONTAINS THE DIAGONAL ELEMENTS OF THE INPUT MATRIX.
!
!         E CONTAINS THE SUBDIAGONAL ELEMENTS OF THE INPUT MATRIX
!           IN ITS LAST N-1 POSITIONS.  E(1) IS ARBITRARY.
!
!         Z CONTAINS THE TRANSFORMATION MATRIX PRODUCED IN THE
!           REDUCTION BY  TRED2, IF PERFORMED.  IF THE EIGENVECTORS
!           OF THE TRIDIAGONAL MATRIX ARE DESIRED, Z MUST CONTAIN
!           THE IDENTITY MATRIX.
!
!       ON OUTPUT
!
!         D CONTAINS THE EIGENVALUES IN ASCENDING ORDER.  IF AN
!           ERROR EXIT IS MADE, THE EIGENVALUES ARE CORRECT BUT
!           UNORDERED FOR INDICES 1,2,...,IERR-1.
!
!         E HAS BEEN DESTROYED.
!
!         Z CONTAINS ORTHONORMAL EIGENVECTORS OF THE SYMMETRIC
!           TRIDIAGONAL (OR FULL) MATRIX.  IF AN ERROR EXIT IS MADE,
!           Z CONTAINS THE EIGENVECTORS ASSOCIATED WITH THE STORED
!           EIGENVALUES.
!
!         IERR IS SET TO
!           ZERO       FOR NORMAL RETURN,
!           J          IF THE J-TH EIGENVALUE HAS NOT BEEN
!                      DETERMINED AFTER 30 ITERATIONS.
!
!      CALLS PYTHAG FOR  DSQRT(A*A + B*B) .
!
!      QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO BURTON S. GARBOW,
!      MATHEMATICS AND COMPUTER SCIENCE DIV, ARGONNE NATIONAL LABORATORY
!
!      THIS VERSION DATED AUGUST 1983.
!
!      ------------------------------------------------------------------
!
      IERR = 0
      IF (N .EQ. 1) GO TO 1001
!
      DO 100 I = 2, N
  100 E(I-1) = E(I)
!
      F = 0.0E0
      TST1 = 0.0E0
      E(N) = 0.0E0
!
      DO 240 L = 1, N
         J = 0
         H = ABS(D(L)) + ABS(E(L))
         IF (TST1 .LT. H) TST1 = H
!     .......... LOOK FOR SMALL SUB-DIAGONAL ELEMENT ..........
         DO 110 M = L, N
            TST2 = TST1 + ABS(E(M))
            IF (TST2 .EQ. TST1) GO TO 120
!     .......... E(N) IS ALWAYS ZERO, SO THERE IS NO EXIT
!                THROUGH THE BOTTOM OF THE LOOP ..........
  110    CONTINUE
!
  120    IF (M .EQ. L) GO TO 220
  130    IF (J .EQ. 30) GO TO 1000
         J = J + 1
!     .......... FORM SHIFT ..........
```



```
         L1 = L + 1
         L2 = L1 + 1
         G = D(L)
         P = (D(L1) - G) / (2.0E0 * E(L))
         R = 1.0E0
         R = PYTHAG(P,R)
         D(L) = E(L) / (P + SIGN(R,P))
         D(L1) = E(L) * (P + SIGN(R,P))
         DL1 = D(L1)
         H = G - D(L)
         IF (L2 .GT. N) GO TO 145
!
         DO 140 I = L2, N
  140    D(I) = D(I) - H
!
  145    F = F + H
!     .......... QL TRANSFORMATION ..........
         P = D(M)
         C = 1.0E0
         C2 = C
         EL1 = E(L1)
         S = 0.0E0
         MML = M - L
!     .......... FOR I=M-1 STEP -1 UNTIL L DO -- ..........
         DO 200 II = 1, MML
            C3 = C2
            C2 = C
            S2 = S
            I = M - II
            G = C * E(I)
            H = C * P
            R = PYTHAG(P,E(I))
            E(I+1) = S * R
            S = E(I) / R
            C = P / R
            P = C * D(I) - S * G
            D(I+1) = H + S * (C * G + S * D(I))
!     .......... FORM VECTOR ..........
            DO 180 K = 1, N
               H = Z(K,I+1)
               Z(K,I+1) = S * Z(K,I) + C * H
               Z(K,I) = C * Z(K,I) - S * H
  180       CONTINUE
!
  200    CONTINUE
!
         P = -S * S2 * C3 * EL1 * E(L) / DL1
         E(L) = S * P
         D(L) = C * P
         TST2 = TST1 + ABS(E(L))
         IF (TST2 .GT. TST1) GO TO 130
  220    D(L) = D(L) + F
  240 END DO
!     .......... ORDER EIGENVALUES AND EIGENVECTORS ..........
      DO 300 II = 2, N
         I = II - 1
         K = I
         P = D(I)
!
         DO 260 J = II, N
            IF (D(J) .GE. P) GO TO 260
            K = J
            P = D(J)
  260    CONTINUE
!
         IF (K .EQ. I) GO TO 300
         D(K) = D(I)
         D(I) = P
!
```

```
        DO 280 J = 1, N
           P = Z(J,I)
           Z(J,I) = Z(J,K)
           Z(J,K) = P
  280   CONTINUE
!
  300 END DO
!
      GO TO 1001
!     .......... SET ERROR -- NO CONVERGENCE TO AN
!                EIGENVALUE AFTER 30 ITERATIONS .........
 1000 IERR = L
 1001 RETURN
      END
!*
      SUBROUTINE TRIDIB(N,EPS1,D,E,E2,XLB,UB,M11,M,W,IND,IERR,RV4,RV5)
      IMPLICIT REAL*8 (A-H,O-Z)
!     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION D(N),E(N),E2(N),W(M),RV4(N),RV5(N)
!!    REAL ABS,MAX,MIN,DBLE
      DIMENSION IND(M)
!
!     THIS SUBROUTINE IS A TRANSLATION OF THE ALGOL PROCEDURE BISECT,
!     NUM. MATH. 9, 386-393(1967) BY BARTH, MARTIN, AND WILKINSON.
!     HANDBOOK FOR AUTO. COMP., VOL.II-LINEAR ALGEBRA, 249-256(1971).
!
!     THIS SUBROUTINE FINDS THOSE EIGENVALUES OF A TRIDIAGONAL
!     SYMMETRIC MATRIX BETWEEN SPECIFIED BOUNDARY INDICES,
!     USING BISECTION.
!
!     ON INPUT-
!
!        N IS THE ORDER OF THE MATRIX,
!
!        EPS1 IS AN ABSOLUTE ERROR TOLERANCE FOR THE COMPUTED
!          EIGENVALUES.  IF THE INPUT EPS1 IS NON-POSITIVE,
!          IT IS RESET FOR EACH SUBMATRIX TO A DEFAULT VALUE,
!          NAMELY, MINUS THE PRODUCT OF THE RELATIVE MACHINE
!          PRECISION AND THE 1-NORM OF THE SUBMATRIX,
!
!        D CONTAINS THE DIAGONAL ELEMENTS OF THE INPUT MATRIX,
!
!        E CONTAINS THE SUBDIAGONAL ELEMENTS OF THE INPUT MATRIX
!          IN ITS LAST N-1 POSITIONS.  E(1) IS ARBITRARY,
!
!        E2 CONTAINS THE SQUARES OF THE CORRESPONDING ELEMENTS OF E.
!          E2(1) IS ARBITRARY,
!
!        M11 SPECIFIES THE LOWER BOUNDARY INDEX FOR THE DESIRED
!          EIGENVALUES,
!
!        M SPECIFIES THE NUMBER OF EIGENVALUES DESIRED.  THE UPPER
!          BOUNDARY INDEX M22 IS THEN OBTAINED AS M22=M11+M-1.
!
!     ON OUTPUT-
!
!        EPS1 IS UNALTERED UNLESS IT HAS BEEN RESET TO ITS
!          (LAST) DEFAULT VALUE,
!
!        D AND E ARE UNALTERED,
!
!        ELEMENTS OF E2, CORRESPONDING TO ELEMENTS OF E REGARDED
!          AS NEGLIGIBLE, HAVE BEEN REPLACED BY ZERO CAUSING THE
!          MATRIX TO SPLIT INTO A DIRECT SUM OF SUBMATRICES.
!          E2(1) IS ALSO SET TO ZERO,
!
!        XLB AND UB DEFINE AN INTERVAL CONTAINING EXACTLY THE DESIRED
!          EIGENVALUES,
!
```



```
!        W CONTAINS, IN ITS FIRST M POSITIONS, THE EIGENVALUES
!          BETWEEN INDICES M11 AND M22 IN ASCENDING ORDER,
!
!        IND CONTAINS IN ITS FIRST M POSITIONS THE SUBMATRIX INDICES
!          ASSOCIATED WITH THE CORRESPONDING EIGENVALUES IN W --
!          1 FOR EIGENVALUES BELONGING TO THE FIRST SUBMATRIX FROM
!          THE TOP, 2 FOR THOSE BELONGING TO THE SECOND SUBMATRIX, ETC.,
!
!        IERR IS SET TO
!          ZERO       FOR NORMAL RETURN,
!          3*N+1      IF MULTIPLE EIGENVALUES AT INDEX M11 MAKE
!                     UNIQUE SELECTION IMPOSSIBLE,
!          3*N+2      IF MULTIPLE EIGENVALUES AT INDEX M22 MAKE
!                     UNIQUE SELECTION IMPOSSIBLE,
!
!        RV4 AND RV5 ARE TEMPORARY STORAGE ARRAYS.
!
!     NOTE THAT SUBROUTINE TQL1, IMTQL1, OR TQLRAT IS GENERALLY FASTER
!     THAN TRIDIB, IF MORE THAN N/4 EIGENVALUES ARE TO BE FOUND.
!
!     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
!     APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
!
!     ------------------------------------------------------------------
!
!     ********** XMACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING
!                THE RELATIVE PRECISION OF FLOATING POINT ARITHMETIC.
!
!                **********
      XMACHEP = 2.E0**(-47)
!
      IERR = 0
      JTAG = 0
      XU = D(1)
      X0 = D(1)
      U = 0.E0
!     ********** LOOK FOR SMALL SUB-DIAGONAL ENTRIES AND DETERMINE AN
!                INTERVAL CONTAINING ALL THE EIGENVALUES **********
      DO 40 I = 1, N
         X1 = U
         U = 0.E0
         IF (I .NE. N) U = ABS(E(I+1))
         XU = DMIN1(D(I)-(X1+U),XU)
         X0 = DMAX1(D(I)+(X1+U),X0)
         IF (I .EQ. 1) GO TO 20
         IF (ABS(E(I)) .GT. XMACHEP * (ABS(D(I)) + ABS(D(I-1))))        &
     &      GO TO 40
   20    E2(I) = 0.E0
   40 END DO
!
      X1 = DMAX1(ABS(XU),ABS(X0)) * XMACHEP * FLOAT(N)
      XU = XU - X1
      T1 = XU
      X0 = X0 + X1
      T2 = X0
!     ********** DETERMINE AN INTERVAL CONTAINING EXACTLY
!                THE DESIRED EIGENVALUES **********
      JP = 1
      JQ = N
      M1 = M11 - 1
      IF (M1 .EQ. 0) GO TO 75
      ISTURM = 1
   50 V = X1
      X1 = XU + (X0 - XU) * 0.5E0
      IF (X1 .EQ. V) GO TO 980
      GO TO 320
   60 IF (JS - M1) 65, 73, 70
   65 XU = X1
      GO TO 50
```

```
   70 X0 = X1
      GO TO 50
   73 XU = X1
      T1 = X1
   75 M22 = M1 + M
      IF (M22 .EQ. N) GO TO 90
      X0 = T2
      ISTURM = 2
      GO TO 50
   80 IF (JS - M22) 65, 85, 70
   85 T2 = X1
   90 JQ = 0
      JR = 0
!     ********** ESTABLISH AND PROCESS NEXT SUBMATRIX, REFINING
!                INTERVAL BY THE GERSCHGORIN BOUNDS **********
  100 IF (JR .EQ. M) GO TO 1001
      JTAG = JTAG + 1
      JP = JQ + 1
      XU = D(JP)
      X0 = D(JP)
      U = 0.E0
!
      DO 120 JQ = JP, N
         X1 = U
         U = 0.E0
         V = 0.E0
         IF (JQ .EQ. N) GO TO 110
         U = ABS(E(JQ+1))
         V = E2(JQ+1)
  110    XU = DMIN1(D(JQ)-(X1+U),XU)
         X0 = DMAX1(D(JQ)+(X1+U),X0)
         IF (V .EQ. 0.E0) GO TO 140
  120 END DO
!
  140 X1 = MAX(ABS(XU),ABS(X0)) * XMACHEP
      IF (EPS1 .LE. 0.E0) EPS1 = -X1
      IF (JP .NE. JQ) GO TO 180
!     ********** CHECK FOR ISOLATED ROOT WITHIN INTERVAL **********
      IF (T1 .GT. D(JP) .OR. D(JP) .GE. T2) GO TO 940
      M1 = JP
      M2 = JP
      RV5(JP) = D(JP)
      GO TO 900
  180 X1 = X1 * FLOAT(JQ-JP+1)
      XLB = DMAX1(T1,XU-X1)
      UB = DMIN1(T2,X0+X1)
      X1 = XLB
      ISTURM = 3
      GO TO 320
  200 M1 = JS + 1
      X1 = UB
      ISTURM = 4
      GO TO 320
  220 M2 = JS
      IF (M1 .GT. M2) GO TO 940
!     ********** FIND ROOTS BY BISECTION **********
      X0 = UB
      ISTURM = 5
!
      DO 240 I = M1, M2
         RV5(I) = UB
         RV4(I) = XLB
  240 END DO
!     ********** LOOP FOR K-TH EIGENVALUE
!                FOR K=M2 STEP -1 UNTIL M1 DO --
!                (-DO- NOT USED TO LEGALIZE -COMPUTED GO TO-) **********
      K = M2
  250    XU = XLB
!     ********** FOR I=K STEP -1 UNTIL M1 DO -- **********
```



```
         DO 260 II = M1, K
            I = M1 + K - II
            IF (XU .GE. RV4(I)) GO TO 260
            XU = RV4(I)
            GO TO 280
  260    CONTINUE
!
  280    IF (X0 .GT. RV5(K)) X0 = RV5(K)
!     ********** NEXT BISECTION STEP **********
  300    X1 = (XU + X0) * 0.5E0
         IF ((X0 - XU) .LE. (2.E0 * XMACHEP *                          &
     &      (ABS(XU) + ABS(X0)) + ABS(EPS1))) GO TO 420
!     ********** IN-LINE PROCEDURE FOR STURM SEQUENCE **********
  320    JS = JP - 1
         U = 1.E0
!
         DO 340 I = JP, JQ
            IF (U .NE. 0.E0) GO TO 325
            V = ABS(E(I)) / XMACHEP
            IF (E2(I) .EQ. 0.E0) V = 0.E0
            GO TO 330
  325       V = E2(I) / U
  330       U = D(I) - X1 - V
            IF (U .LT. 0.E0) JS = JS + 1
  340    CONTINUE
!
         GO TO (60,80,200,220,360), ISTURM
!     ********** REFINE INTERVALS **********
  360    IF (JS .GE. K) GO TO 400
         XU = X1
         IF (JS .GE. M1) GO TO 380
         RV4(M1) = X1
         GO TO 300
  380    RV4(JS+1) = X1
         IF (RV5(JS) .GT. X1) RV5(JS) = X1
         GO TO 300
  400    X0 = X1
         GO TO 300
!     ********** K-TH EIGENVALUE FOUND **********
  420    RV5(K) = X1
      K = K - 1
      IF (K .GE. M1) GO TO 250
!     ********** ORDER EIGENVALUES TAGGED WITH THEIR
!                SUBMATRIX ASSOCIATIONS **********
  900 JS = JR
      JR = JR + M2 - M1 + 1
      J = 1
      K = M1
!
      DO 920 L = 1, JR
         IF (J .GT. JS) GO TO 910
         IF (K .GT. M2) GO TO 940
         IF (RV5(K) .GE. W(L)) GO TO 915
!
         DO 905 II = J, JS
            I = L + JS - II
            W(I+1) = W(I)
            IND(I+1) = IND(I)
  905    CONTINUE
!
  910    W(L) = RV5(K)
         IND(L) = JTAG
         K = K + 1
         GO TO 920
  915    J = J + 1
  920 END DO
!
  940 IF (JQ .LT. N) GO TO 100
      GO TO 1001
```

```
!     ********** SET ERROR -- INTERVAL CANNOT BE FOUND CONTAINING
!                EXACTLY THE DESIRED EIGENVALUES **********
  980 IERR = 3 * N + ISTURM
 1001 XLB = T1
      UB = T2
      RETURN
!     ********** LAST CARD OF TRIDIB **********
      END
!*
      SUBROUTINE TINVIT(NM,N,D,E,E2,M,W,IND,Z,                          &
     &                   IERR,RV1,RV2,RV3,RV4,RV6)
      IMPLICIT REAL*8 (A-H,O-Z)
!     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
!
      DIMENSION D(N),E(N),E2(N),W(M),Z(NM,M)
      DIMENSION RV1(N),RV2(N),RV3(N),RV4(N),RV6(N)
!!    REAL SQRT,ABS,DBLE
      DIMENSION IND(M)
!     LEVEL 2, Z
!
!     THIS SUBROUTINE IS A TRANSLATION OF THE INVERSE ITERATION TECH-
!     NIQUE IN THE ALGOL PROCEDURE TRISTURM BY PETERS AND WILKINSON.
!     HANDBOOK FOR AUTO. COMP., VOL.II-LINEAR ALGEBRA, 418-439(1971).
!
!     THIS SUBROUTINE FINDS THOSE EIGENVECTORS OF A TRIDIAGONAL
!     SYMMETRIC MATRIX CORRESPONDING TO SPECIFIED EIGENVALUES,
!     USING INVERSE ITERATION.
!
!     ON INPUT-
!
!        NM MUST BE SET TO THE ROW DIMENSION OF TWO-DIMENSIONAL
!          ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
!          DIMENSION STATEMENT,
!
!        N IS THE ORDER OF THE MATRIX,
!
!        D CONTAINS THE DIAGONAL ELEMENTS OF THE INPUT MATRIX,
!
!        E CONTAINS THE SUBDIAGONAL ELEMENTS OF THE INPUT MATRIX
!          IN ITS LAST N-1 POSITIONS.  E(1) IS ARBITRARY,
!
!        E2 CONTAINS THE SQUARES OF THE CORRESPONDING ELEMENTS OF E,
!          WITH ZEROS CORRESPONDING TO NEGLIGIBLE ELEMENTS OF E.
!          E(I) IS CONSIDERED NEGLIGIBLE IF IT IS NOT LARGER THAN
!          THE PRODUCT OF THE RELATIVE MACHINE PRECISION AND THE SUM
!          OF THE MAGNITUDES OF D(I) AND D(I-1).  E2(1) MUST CONTAIN
!          0.0 IF THE EIGENVALUES ARE IN ASCENDING ORDER, OR 2.0
!          IF THE EIGENVALUES ARE IN DESCENDING ORDER.  IF  BISECT,
!          TRIDIB, OR  IMTQLV  HAS BEEN USED TO FIND THE EIGENVALUES,
!          THEIR OUTPUT E2 ARRAY IS EXACTLY WHAT IS EXPECTED HERE,
!
!        M IS THE NUMBER OF SPECIFIED EIGENVALUES,
!
!        W CONTAINS THE M EIGENVALUES IN ASCENDING OR DESCENDING ORDER,
!
!        IND CONTAINS IN ITS FIRST M POSITIONS THE SUBMATRIX INDICES
!          ASSOCIATED WITH THE CORRESPONDING EIGENVALUES IN W --
!          1 FOR EIGENVALUES BELONGING TO THE FIRST SUBMATRIX FROM
!          THE TOP, 2 FOR THOSE BELONGING TO THE SECOND SUBMATRIX, ETC.
!
!     ON OUTPUT-
!
!        ALL INPUT ARRAYS ARE UNALTERED,
!
!        Z CONTAINS THE ASSOCIATED SET OF ORTHONORMAL EIGENVECTORS.
!          ANY VECTOR WHICH FAILS TO CONVERGE IS SET TO ZERO,
!
!        IERR IS SET TO
!          ZERO       FOR NORMAL RETURN,
```



```
!          -R         IF THE EIGENVECTOR CORRESPONDING TO THE R-TH
!                     EIGENVALUE FAILS TO CONVERGE IN 5 ITERATIONS,
!
!        RV1, RV2, RV3, RV4, AND RV6 ARE TEMPORARY STORAGE ARRAYS.
!
!     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
!     APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
!
!     ------------------------------------------------------------------
!
!     ********** XMACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING
!                THE RELATIVE PRECISION OF FLOATING POINT ARITHMETIC.
!
!                **********
      XMACHEP = 2.E0**(-47)
!
      IERR = 0
      IF (M .EQ. 0) GO TO 1001
      JTAG = 0
      ORDER = 1.E0 - E2(1)
      JQ = 0
!     ********** ESTABLISH AND PROCESS NEXT SUBMATRIX **********
  100 JP = JQ + 1
!
      DO 120 JQ = JP, N
         IF (JQ .EQ. N) GO TO 140
         IF (E2(JQ+1) .EQ. 0.E0) GO TO 140
  120 END DO
!     ********** FIND VECTORS BY INVERSE ITERATION **********
  140 JTAG = JTAG + 1
      JS = 0
!
      DO 920 JR = 1, M
         IF (IND(JR) .NE. JTAG) GO TO 920
         ITS = 1
         X1 = W(JR)
         IF (JS .NE. 0) GO TO 510
!     ********** CHECK FOR ISOLATED ROOT **********
         XU = 1.E0
         IF (JP .NE. JQ) GO TO 490
         RV6(JP) = 1.E0
         GO TO 870
  490    XNORM = ABS(D(JP))
         IP = JP + 1
!
         DO 500 I = IP, JQ
  500    XNORM = XNORM + ABS(D(I)) + ABS(E(I))
!     ********** EPS2 IS THE CRITERION FOR GROUPING,
!                EPS3 REPLACES ZERO PIVOTS AND EQUAL
!                ROOTS ARE MODIFIED BY EPS3,
!                EPS4 IS TAKEN VERY SMALL TO AVOID OVERFLOW **********
         EPS2 = 1.0E-3 * XNORM
         EPS3 = XMACHEP * XNORM
         UK = FLOAT(JQ-JP+1)
         EPS4 = UK * EPS3
         UK = EPS4 / SQRT(UK)
         JS = JP
  505    JGROUP = 0
         GO TO 520
!     ********** LOOK FOR CLOSE OR COINCIDENT ROOTS **********
  510    IF (ABS(X1-X0) .GE. EPS2) GO TO 505
         JGROUP = JGROUP + 1
         IF (ORDER * (X1 - X0) .LE. 0.E0) X1 = X0 + ORDER * EPS3
!     ********** ELIMINATION WITH INTERCHANGES AND
!                INITIALIZATION OF VECTOR **********
  520    V = 0.E0
!
         DO 580 I = JP, JQ
            RV6(I) = UK
```

```
         IF (I .EQ. JP) GO TO 560
         IF (ABS(E(I)) .LT. ABS(U)) GO TO 540
!     ********** WARNING -- A DIVIDE CHECK MAY OCCUR HERE IF
!                E2 ARRAY HAS NOT BEEN SPECIFIED CORRECTLY **********
         XU = U / E(I)
         RV4(I) = XU
         RV1(I-1) = E(I)
         RV2(I-1) = D(I) - X1
         RV3(I-1) = 0.E0
         IF (I .NE. JQ) RV3(I-1) = E(I+1)
         U = V - XU * RV2(I-1)
         V = -XU * RV3(I-1)
         GO TO 580
  540    XU = E(I) / U
         RV4(I) = XU
         RV1(I-1) = U
         RV2(I-1) = V
         RV3(I-1) = 0.E0
  560    U = D(I) - X1 - XU * V
         IF (I .NE. JQ) V = E(I+1)
  580 CONTINUE
!
      IF (U .EQ. 0.E0) U = EPS3
      RV1(JQ) = U
      RV2(JQ) = 0.E0
      RV3(JQ) = 0.E0
!     ********** BACK SUBSTITUTION
!                FOR I=JQ STEP -1 UNTIL P DO -- **********
  600 DO 620 II = JP, JQ
         I = JP + JQ - II
         RV6(I) = (RV6(I) - U * RV2(I) - V * RV3(I)) / RV1(I)
         V = U
         U = RV6(I)
  620 CONTINUE
!     ********** ORTHOGONALIZE WITH RESPECT TO PREVIOUS
!                MEMBERS OF GROUP **********
      IF (JGROUP .EQ. 0) GO TO 700
      J = JR
!
      DO 680 JJ = 1, JGROUP
  630    J = J - 1
         IF (IND(J) .NE. JTAG) GO TO 630
         XU = 0.E0
!
         DO 640 I = JP, JQ
  640    XU = XU + RV6(I) * Z(I,J)
!
         DO 660 I = JP, JQ
  660    RV6(I) = RV6(I) - XU * Z(I,J)
!
  680 CONTINUE
!
  700 XNORM = 0.E0
!
      DO 720 I = JP, JQ
  720 XNORM = XNORM + ABS(RV6(I))
!
      IF (XNORM .GE. 1.E0) GO TO 840
!     ********** FORWARD SUBSTITUTION **********
      IF (ITS .EQ. 5) GO TO 830
      IF (XNORM .NE. 0.E0) GO TO 740
      RV6(JS) = EPS4
      JS = JS + 1
      IF (JS .GT. JQ) JS = JP
      GO TO 780
  740 XU = EPS4 / XNORM
!
      DO 760 I = JP, JQ
  760 RV6(I) = RV6(I) * XU
```



```
!     ********** ELIMINATION OPERATIONS ON NEXT VECTOR
!                ITERATE **********
  780 DO 820 I = IP, JQ
         U = RV6(I)
!     ********** IF RV1(I-1) .EQ. E(I), A ROW INTERCHANGE
!                WAS PERFORMED EARLIER IN THE
!                TRIANGULARIZATION PROCESS **********
         IF (RV1(I-1) .NE. E(I)) GO TO 800
         U = RV6(I-1)
         RV6(I-1) = RV6(I)
  800    RV6(I) = U - RV4(I) * RV6(I-1)
  820 CONTINUE
!
      ITS = ITS + 1
      GO TO 600
!     ********** SET ERROR -- NON-CONVERGED EIGENVECTOR **********
  830 IERR = -JR
      XU = 0.E0
      GO TO 870
!     ********** NORMALIZE SO THAT SUM OF SQUARES IS
!                1 AND EXPAND TO FULL ORDER **********
  840 U = 0.E0
!
      DO 860 I = JP, JQ
  860 U = U + RV6(I)**2
!
      XU = 1.E0 / SQRT(U)
!
  870 DO 880 I = 1, N
  880 Z(I,JR) = 0.E0
!
      DO 900 I = JP, JQ
  900 Z(I,JR) = RV6(I) * XU
!
      X0 = X1
  920 END DO
!
     IF (JQ .LT. N) GO TO 100
 1001 RETURN
!     ********** LAST CARD OF TINVIT **********
     END
!*
     SUBROUTINE TRED3(N,NV,A,D,E,E2)
!
     IMPLICIT REAL*8 (A-H,O-Z)
     INTEGER I,J,K,L,N,II,IZ,JK,NV,JM1
     DIMENSION A(NV),D(N),E(N),E2(N)
!!   REAL F,G,H,HH,SCALE
!
!    THIS SUBROUTINE IS A TRANSLATION OF THE ALGOL PROCEDURE TRED3,
!    NUM. MATH. 11, 181-195(1968) BY MARTIN, REINSCH, AND WILKINSON.
!    HANDBOOK FOR AUTO. COMP., VOL.II-LINEAR ALGEBRA, 212-226(1971).
!
!    THIS SUBROUTINE REDUCES A REAL SYMMETRIC MATRIX, STORED AS
!    A ONE-DIMENSIONAL ARRAY, TO A SYMMETRIC TRIDIAGONAL MATRIX
!    USING ORTHOGONAL SIMILARITY TRANSFORMATIONS.
!
!    ON INPUT
!
!       N IS THE ORDER OF THE MATRIX.
!
!       NV MUST BE SET TO THE DIMENSION OF THE ARRAY PARAMETER A
!         AS DECLARED IN THE CALLING PROGRAM DIMENSION STATEMENT.
!
!       A CONTAINS THE LOWER TRIANGLE OF THE REAL SYMMETRIC
!         INPUT MATRIX, STORED ROW-WISE AS A ONE-DIMENSIONAL
!         ARRAY, IN ITS FIRST N*(N+1)/2 POSITIONS.
!
!    ON OUTPUT
```

```
!
!        A CONTAINS INFORMATION ABOUT THE ORTHOGONAL
!          TRANSFORMATIONS USED IN THE REDUCTION.
!
!        D CONTAINS THE DIAGONAL ELEMENTS OF THE TRIDIAGONAL MATRIX.
!
!        E CONTAINS THE SUBDIAGONAL ELEMENTS OF THE TRIDIAGONAL
!          MATRIX IN ITS LAST N-1 POSITIONS.  E(1) IS SET TO ZERO.
!
!        E2 CONTAINS THE SQUARES OF THE CORRESPONDING ELEMENTS OF E.
!          E2 MAY COINCIDE WITH E IF THE SQUARES ARE NOT NEEDED.
!
!     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO BURTON S. GARBOW,
!     MATHEMATICS AND COMPUTER SCIENCE DIV, ARGONNE NATIONAL LABORATORY
!
!     THIS VERSION DATED AUGUST 1983.
!
!     ------------------------------------------------------------------
!
!     .......... FOR I=N STEP -1 UNTIL 1 DO -- ..........
      DO 300 II = 1, N
         I = N + 1 - II
         L = I - 1
         IZ = (I * L) / 2
         H = 0.0E0
         SCALE = 0.0E0
         IF (L .LT. 1) GO TO 130
!     .......... SCALE ROW (ALGOL TOL THEN NOT NEEDED) ..........
         DO 120 K = 1, L
            IZ = IZ + 1
            D(K) = A(IZ)
            SCALE = SCALE + ABS(D(K))
  120    CONTINUE
!
         IF (SCALE .NE. 0.0E0) GO TO 140
  130    E(I) = 0.0E0
         E2(I) = 0.0E0
         GO TO 290
!
  140    DO 150 K = 1, L
            D(K) = D(K) / SCALE
            H = H + D(K) * D(K)
  150    CONTINUE
!
         E2(I) = SCALE * SCALE * H
         F = D(L)
         G = -SIGN(SQRT(H),F)
         E(I) = SCALE * G
         H = H - F * G
         D(L) = F - G
         A(IZ) = SCALE * D(L)
         IF (L .EQ. 1) GO TO 290
         JK = 1
!
         DO 240 J = 1, L
            F = D(J)
            G = 0.0E0
            JM1 = J - 1
            IF (JM1 .LT. 1) GO TO 220
!
            DO 200 K = 1, JM1
               G = G + A(JK) * D(K)
               E(K) = E(K) + A(JK) * F
               JK = JK + 1
  200       CONTINUE
!
  220       E(J) = G + A(JK) * F
            JK = JK + 1
  240    CONTINUE
```



```
!     .......... FORM P ..........
         F = 0.0E0
!
         DO 245 J = 1, L
            E(J) = E(J) / H
            F = F + E(J) * D(J)
  245    CONTINUE
!
         HH = F / (H + H)
!     .......... FORM Q ..........
         DO 250 J = 1, L
  250    E(J) = E(J) - HH * D(J)
!
         JK = 1
!     .......... FORM REDUCED A ..........
         DO 280 J = 1, L
            F = D(J)
            G = E(J)
!
            DO 260 K = 1, J
               A(JK) = A(JK) - F * E(K) - G * D(K)
               JK = JK + 1
  260       CONTINUE
!
  280    CONTINUE
!
  290    D(I) = A(IZ+1)
         A(IZ+1) = SCALE * SQRT(H)
  300 END DO
!
      RETURN
      END
!*
      SUBROUTINE TRBAK3(NM,N,NV,A,M,Z)
!
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER I,J,K,L,M,N,IK,IZ,NM,NV
      DIMENSION A(NV),Z(NM,M)
!!    REAL H,S
!
!     THIS SUBROUTINE IS A TRANSLATION OF THE ALGOL PROCEDURE TRBAK3,
!     NUM. MATH. 11, 181-195(1968) BY MARTIN, REINSCH, AND WILKINSON.
!     HANDBOOK FOR AUTO. COMP., VOL.II-LINEAR ALGEBRA, 212-226(1971).
!
!     THIS SUBROUTINE FORMS THE EIGENVECTORS OF A REAL SYMMETRIC
!     MATRIX BY BACK TRANSFORMING THOSE OF THE CORRESPONDING
!     SYMMETRIC TRIDIAGONAL MATRIX DETERMINED BY  TRED3.
!
!     ON INPUT
!
!        NM MUST BE SET TO THE ROW DIMENSION OF TWO-DIMENSIONAL
!          ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
!          DIMENSION STATEMENT.
!
!        N IS THE ORDER OF THE MATRIX.
!
!        NV MUST BE SET TO THE DIMENSION OF THE ARRAY PARAMETER A
!          AS DECLARED IN THE CALLING PROGRAM DIMENSION STATEMENT.
!
!        A CONTAINS INFORMATION ABOUT THE ORTHOGONAL TRANSFORMATIONS
!          USED IN THE REDUCTION BY  TRED3  IN ITS FIRST
!          N*(N+1)/2 POSITIONS.
!
!        M IS THE NUMBER OF EIGENVECTORS TO BE BACK TRANSFORMED.
!
!        Z CONTAINS THE EIGENVECTORS TO BE BACK TRANSFORMED
!          IN ITS FIRST M COLUMNS.
!
!     ON OUTPUT
```

```
!
!        Z CONTAINS THE TRANSFORMED EIGENVECTORS
!          IN ITS FIRST M COLUMNS.
!
!     NOTE THAT TRBAK3 PRESERVES VECTOR EUCLIDEAN NORMS.
!
!     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO BURTON S. GARBOW,
!     MATHEMATICS AND COMPUTER SCIENCE DIV, ARGONNE NATIONAL LABORATORY
!
!     THIS VERSION DATED AUGUST 1983.
!
!     ------------------------------------------------------------------
!
      IF (M .EQ. 0) GO TO 200
      IF (N .EQ. 1) GO TO 200
!
      DO 140 I = 2, N
         L = I - 1
         IZ = (I * L) / 2
         IK = IZ + I
         H = A(IK)
         IF (H .EQ. 0.0E0) GO TO 140
!
         DO 130 J = 1, M
            S = 0.0E0
            IK = IZ
!
            DO 110 K = 1, L
               IK = IK + 1
               S = S + A(IK) * Z(K,J)
  110       CONTINUE
!     .......... DOUBLE DIVISION AVOIDS POSSIBLE UNDERFLOW ..........
            S = (S / H) / H
            IK = IZ
!
            DO 120 K = 1, L
               IK = IK + 1
               Z(K,J) = Z(K,J) - S * A(IK)
  120       CONTINUE
!
  130    CONTINUE
!
  140 END DO
!
  200 RETURN
      END
!*
      SUBROUTINE HTRIDI(N,IROW,AR,AI,D,E,E2,TAU)
!
      IMPLICIT REAL*8 (A-H,O-Z)
!     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION IROW(N)
      DIMENSION AR((N*N+N)/2),AI((N*N+N)/2),D(N),E(N),E2(N),TAU(2,N)
!!    REAL SQRT,CABS,ABS
!
!     THIS SUBROUTINE IS A TRANSLATION OF A COMPLEX ANALOGUE OF
!     THE ALGOL PROCEDURE TRED1, NUM. MATH. 11, 181-195(1968)
!     BY MARTIN, REINSCH, AND WILKINSON.
!     HANDBOOK FOR AUTO. COMP., VOL.II-LINEAR ALGEBRA, 212-226(1971).
!
!     THIS SUBROUTINE REDUCES A COMPLEX HERMITIAN MATRIX
!     TO A REAL SYMMETRIC TRIDIAGONAL MATRIX USING
!     UNITARY SIMILARITY TRANSFORMATIONS.
!
!     ON INPUT-
!
!        N IS THE ORDER OF THE MATRIX,
!
!        IROW CONTAINS THE INDEX OF THE FIRST ELEMENT IN ROW I.
```



```
!
!        AR AND AI CONTAIN THE REAL AND IMAGINARY PARTS,
!          RESPECTIVELY, OF THE COMPLEX HERMITIAN INPUT MATRIX.
!          ONLY THE LOWER TRIANGLE OF THE MATRIX NEED BE SUPPLIED.
!          THEY ARE STORED IN PACKED FORM? A11,A21,A22,A31...
!
!     ON OUTPUT-
!
!        AR AND AI CONTAIN INFORMATION ABOUT THE UNITARY TRANS-
!          FORMATIONS USED IN THE REDUCTION IN THEIR FULL LOWER
!          TRIANGLES.  THEIR STRICT UPPER TRIANGLES AND THE
!          DIAGONAL OF AR ARE UNALTERED,
!
!        D CONTAINS THE DIAGONAL ELEMENTS OF THE THE TRIDIAGONAL MATRIX,
!
!        E CONTAINS THE SUBDIAGONAL ELEMENTS OF THE TRIDIAGONAL
!          MATRIX IN ITS LAST N-1 POSITIONS.  E(1) IS SET TO ZERO,
!
!        E2 CONTAINS THE SQUARES OF THE CORRESPONDING ELEMENTS OF E.
!          E2 MAY COINCIDE WITH E IF THE SQUARES ARE NOT NEEDED,
!
!        TAU CONTAINS FURTHER INFORMATION ABOUT THE TRANSFORMATIONS.
!
!     ARITHMETIC IS REAL EXCEPT FOR THE USE OF THE SUBROUTINES
!     ABS AND DCMPLX IN COMPUTING COMPLEX ABSOLUTE VALUES.
!
!     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
!     APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
!
!     ------------------------------------------------------------------
!
      TAU(1,N) = 1.0
      TAU(2,N) = 0.0
!
      DO 100 I = 1, N
      II = IROW(I) + I
  100 D(I) = AR(II)
!     ********** FOR I=N STEP -1 UNTIL 1 DO -- **********
      DO 300 III = 1, N
         I = N + 1 - III
         L = I - 1
         H = 0.0
         SCALE = 0.0
         IF (L .LT. 1) GO TO 130
!     ********** SCALE ROW (ALGOL TOL THEN NOT NEEDED) **********
         DO 120 K = 1, L
         IK = IROW(I) + K
  120    SCALE = SCALE + ABS(AR(IK)) + ABS(AI(IK))
!
         IF (SCALE .NE. 0.0) GO TO 140
         TAU(1,L) = 1.0
         TAU(2,L) = 0.0
  130    E(I) = 0.0
         E2(I) = 0.0
         GO TO 290
!
  140    DO 150 K = 1, L
            IK = IROW(I) + K
            AR(IK) = AR(IK) / SCALE
            AI(IK) = AI(IK) / SCALE
            H = H + AR(IK) * AR(IK) + AI(IK) * AI(IK)
  150    CONTINUE
!
         E2(I) = SCALE * SCALE * H
         G = SQRT(H)
         E(I) = SCALE * G
         IL = IROW(I) + L
         F = PYTHAG(AR(IL),AI(IL))
!     ********** FORM NEXT DIAGONAL ELEMENT OF MATRIX T **********
```

```
         IF (F .EQ. 0.0) GO TO 160
         TAU(1,L) = (AI(IL) * TAU(2,I) - AR(IL) * TAU(1,I)) / F
         SI = (AR(IL) * TAU(2,I) + AI(IL) * TAU(1,I)) / F
         H = H + F * G
         G = 1.0 + G / F
         AR(IL) = G * AR(IL)
         AI(IL) = G * AI(IL)
         IF (L .EQ. 1) GO TO 270
         GO TO 170
  160    TAU(1,L) = -TAU(1,I)
         SI = TAU(2,I)
         AR(IL) = G
  170    F = 0.0
!
         DO 240 J = 1, L
            G = 0.0
            GI = 0.0
!     ********** FORM ELEMENT OF A*U **********
            DO 180 K = 1, J
               JK = IROW(J) + K
               IK = IROW(I) + K
               G = G + AR(JK) * AR(IK) + AI(JK) * AI(IK)
               GI = GI - AR(JK) * AI(IK) + AI(JK) * AR(IK)
  180       CONTINUE
!
            JP1 = J + 1
            IF (L .LT. JP1) GO TO 220
!
            DO 200 K = JP1, L
               KJ = IROW(K) + J
               IK = IROW(I) + K
               G = G + AR(KJ) * AR(IK) - AI(KJ) * AI(IK)
               GI = GI - AR(KJ) * AI(IK) - AI(KJ) * AR(IK)
  200       CONTINUE
!     ********** FORM ELEMENT OF P **********
  220       E(J) = G / H
            TAU(2,J) = GI / H
            IJ = IROW(I) + J
            F = F + E(J) * AR(IJ) - TAU(2,J) * AI(IJ)
  240    CONTINUE
!
         HH = F / (H + H)
!     ********** FORM REDUCED A **********
         DO 260 J = 1, L
            IJ = IROW(I) + J
            F = AR(IJ)
            G = E(J) - HH * F
            E(J) = G
            FI = -AI(IJ)
            GI = TAU(2,J) - HH * FI
            TAU(2,J) = -GI
!
            DO 260 K = 1, J
               JK = IROW(J) + K
               IK = IROW(I) + K
               AR(JK) = AR(JK) - F * E(K) - G * AR(IK)                 &
     &                         + FI * TAU(2,K) + GI * AI(IK)
               AI(JK) = AI(JK) - F * TAU(2,K) - G * AI(IK)             &
     &                         - FI * E(K) - GI * AR(IK)
  260    CONTINUE
!
  270    DO 280 K = 1, L
            IK = IROW(I) + K
            AR(IK) = SCALE * AR(IK)
            AI(IK) = SCALE * AI(IK)
  280    CONTINUE
!
         TAU(2,L) = -SI
  290    HH = D(I)
```



```
            II = IROW(I) + I
            D(I) = AR(II)
            AR(II) = HH
            AI(II) = SCALE * SQRT(H)
  300    END DO
!
         RETURN
!        ********** LAST CARD OF HTRIDI **********
         END
!*
         SUBROUTINE HTRIBK(N,IROW,AR,AI,TAU,M,ZR,ZI)
!
         IMPLICIT REAL*8 (A-H,O-Z)
!        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
         DIMENSION IROW(N)
         DIMENSION AR((N*N+N)/2),AI((N*N+N)/2),TAU(2,N),ZR(N*N),ZI(N*N)
!
!        THIS SUBROUTINE IS A TRANSLATION OF A COMPLEX ANALOGUE OF
!        THE ALGOL PROCEDURE TRBAK1, NUM. MATH. 11, 181-195(1968)
!        BY MARTIN, REINSCH, AND WILKINSON.
!        HANDBOOK FOR AUTO. COMP., VOL.II-LINEAR ALGEBRA, 212-226(1971).
!
!        THIS SUBROUTINE FORMS THE EIGENVECTORS OF A COMPLEX HERMITIAN
!        MATRIX BY BACK TRANSFORMING THOSE OF THE CORRESPONDING
!        REAL SYMMETRIC TRIDIAGONAL MATRIX DETERMINED BY  HTRIDI.
!
!        ON INPUT-
!
!           N IS THE ORDER OF THE MATRIX,
!
!           IROW CONTAINS THE INDEX OF THE FIRST ELEMENT IN ROW I,
!
!           AR AND AI CONTAIN INFORMATION ABOUT THE UNITARY TRANS-
!             FORMATIONS USED IN THE REDUCTION BY  HTRIDI  IN THEIR
!             FULL LOWER TRIANGLES EXCEPT FOR THE DIAGONAL OF AR,
!             THEY ARE STORED IN PACKED FORM ? A11,A12,A22,A31...
!
!           TAU CONTAINS FURTHER INFORMATION ABOUT THE TRANSFORMATIONS,
!
!           M IS THE NUMBER OF EIGENVECTORS TO BE BACK TRANSFORMED,
!
!           ZR CONTAINS THE EIGENVECTORS TO BE BACK TRANSFORMED
!             IN ITS FIRST M*N POSITIONS.
!
!        ON OUTPUT-
!
!           ZR AND ZI CONTAIN THE REAL AND IMAGINARY PARTS,
!             RESPECTIVELY, OF THE TRANSFORMED EIGENVECTORS
!             IN THEIR FIRST M*N POSITIONS.
!
!        NOTE THAT THE LAST COMPONENT OF EACH RETURNED VECTOR
!        IS REAL AND THAT VECTOR EUCLIDEAN NORMS ARE PRESERVED.
!
!        QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
!        APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
!
!        ------------------------------------------------------------------
!
!
         IF (M .EQ. 0) GO TO 200
!        ********** TRANSFORM THE EIGENVECTORS OF THE REAL SYMMETRIC
!                   TRIDIAGONAL MATRIX TO THOSE OF THE HERMITIAN
!                   TRIDIAGONAL MATRIX. **********
!
         DO 50 J = 1, M
            DO 50 K = 1, N
               KJ = K + (J - 1) * N
               ZI(KJ) = -ZR(KJ) * TAU(2,K)
               ZR(KJ) = ZR(KJ) * TAU(1,K)
```

```
   50 CONTINUE
!
      IF (N .EQ. 1) GO TO 200
!     ********** RECOVER AND APPLY THE HOUSEHOLDER MATRICES **********
      DO 140 I = 2, N
         L = I - 1
         II = IROW(I) + I
         H = AI(II)
         IF (H .EQ. 0.0) GO TO 140
!
         DO 130 J = 1, M
            S = 0.0
            SI = 0.0
            JN = (J - 1) * N
!
            DO 110 K = 1, L
               IK = IROW(I) + K
               KJ = K + JN
               S = S + AR(IK) * ZR(KJ) - AI(IK) * ZI(KJ)
               SI = SI + AR(IK) * ZI(KJ) + AI(IK) * ZR(KJ)
  110       CONTINUE
!     ********** DOUBLE DIVISIONS AVOID POSSIBLE UNDERFLOW **********
            S = (S / H) / H
            SI = (SI / H) / H
!
            DO 120 K = 1, L
               IK = IROW(I) + K
               KJ = K + JN
               ZR(KJ) = ZR(KJ) - S * AR(IK) - SI * AI(IK)
               ZI(KJ) = ZI(KJ) - SI * AR(IK) + S * AI(IK)
  120       CONTINUE
!
  130    CONTINUE
!
  140 END DO
!
  200 RETURN
!     ********** LAST CARD OF HTRIBK **********
      END
!*
      SUBROUTINE IMTQLV(N,D,E,E2,W,IND,IERR,RV1)
!
      IMPLICIT REAL*8 (A-H,O-Z)
!     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION D(N),E(N),E2(N),W(N),RV1(N)
      DIMENSION IND(N)
!
!     THIS SUBROUTINE IS A VARIANT OF  IMTQL1  WHICH IS A TRANSLATION OF
!     ALGOL PROCEDURE IMTQL1, NUM. MATH. 12, 377-383(1968) BY MARTIN AND
!     WILKINSON, AS MODIFIED IN NUM. MATH. 15, 450(1970) BY DUBRULLE.
!     HANDBOOK FOR AUTO. COMP., VOL.II-LINEAR ALGEBRA, 241-248(1971).
!
!     THIS SUBROUTINE FINDS THE EIGENVALUES OF A SYMMETRIC TRIDIAGONAL
!     MATRIX BY THE IMPLICIT QL METHOD AND ASSOCIATES WITH THEM
!     THEIR CORRESPONDING SUBMATRIX INDICES.
!
!     ON INPUT
!
!        N IS THE ORDER OF THE MATRIX.
!
!        D CONTAINS THE DIAGONAL ELEMENTS OF THE INPUT MATRIX.
!
!        E CONTAINS THE SUBDIAGONAL ELEMENTS OF THE INPUT MATRIX
!          IN ITS LAST N-1 POSITIONS.  E(1) IS ARBITRARY.
!
!        E2 CONTAINS THE SQUARES OF THE CORRESPONDING ELEMENTS OF E.
!          E2(1) IS ARBITRARY.
!
!     ON OUTPUT
```



```
!
!        D AND E ARE UNALTERED.
!
!        ELEMENTS OF E2, CORRESPONDING TO ELEMENTS OF E REGARDED
!          AS NEGLIGIBLE, HAVE BEEN REPLACED BY ZERO CAUSING THE
!          MATRIX TO SPLIT INTO A DIRECT SUM OF SUBMATRICES.
!          E2(1) IS ALSO SET TO ZERO.
!
!        W CONTAINS THE EIGENVALUES IN ASCENDING ORDER.  IF AN
!          ERROR EXIT IS MADE, THE EIGENVALUES ARE CORRECT AND
!          ORDERED FOR INDICES 1,2,...IERR-1, BUT MAY NOT BE
!          THE SMALLEST EIGENVALUES.
!
!        IND CONTAINS THE SUBMATRIX INDICES ASSOCIATED WITH THE
!          CORRESPONDING EIGENVALUES IN W -- 1 FOR EIGENVALUES
!          BELONGING TO THE FIRST SUBMATRIX FROM THE TOP,
!          2 FOR THOSE BELONGING TO THE SECOND SUBMATRIX, ETC..
!
!        IERR IS SET TO
!          ZERO       FOR NORMAL RETURN,
!          J          IF THE J-TH EIGENVALUE HAS NOT BEEN
!                     DETERMINED AFTER 30 ITERATIONS.
!
!        RV1 IS A TEMPORARY STORAGE ARRAY.
!
!     CALLS PYTHAG FOR  DSQRT(A*A + B*B) .
!
!     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO BURTON S. GARBOW,
!     MATHEMATICS AND COMPUTER SCIENCE DIV, ARGONNE NATIONAL LABORATORY
!
!     THIS VERSION DATED AUGUST 1983.
!
!     ------------------------------------------------------------------
!
      IERR = 0
      K = 0
      JTAG = 0
!
      DO 100 I = 1, N
         W(I) = D(I)
         IF (I .NE. 1) RV1(I-1) = E(I)
  100 END DO
!
      E2(1) = 0.0E0
      RV1(N) = 0.0E0
!
      DO 290 L = 1, N
         J = 0
!     .......... LOOK FOR SMALL SUB-DIAGONAL ELEMENT ..........
  105    DO 110 M = L, N
            IF (M .EQ. N) GO TO 120
            TST1 = ABS(W(M)) + ABS(W(M+1))
            TST2 = TST1 + ABS(RV1(M))
            IF (TST2 .EQ. TST1) GO TO 120
!     .......... GUARD AGAINST UNDERFLOWED ELEMENT OF E2 ..........
            IF (E2(M+1) .EQ. 0.0E0) GO TO 125
  110    CONTINUE
!
  120    IF (M .LE. K) GO TO 130
         IF (M .NE. N) E2(M+1) = 0.0E0
  125    K = M
         JTAG = JTAG + 1
  130    P = W(L)
         IF (M .EQ. L) GO TO 215
         IF (J .EQ. 30) GO TO 1000
         J = J + 1
!     .......... FORM SHIFT ..........
         G = (W(L+1) - P) / (2.0E0 * RV1(L))
         R = 1.0E0
```

```
        R = PYTHAG(G,R)
        G = W(M) - P + RV1(L) / (G + SIGN(R,G))
        S = 1.0E0
        C = 1.0E0
        P = 0.0E0
        MML = M - L
!     .......... FOR I=M-1 STEP -1 UNTIL L DO -- ..........
        DO 200 II = 1, MML
           I = M - II
           F = S * RV1(I)
           B = C * RV1(I)
           R = PYTHAG(F,G)
           RV1(I+1) = R
           IF (R .EQ. 0.0E0) GO TO 210
           S = F / R
           C = G / R
           G = W(I+1) - P
           R = (W(I) - G) * S + 2.0E0 * C * B
           P = S * R
           W(I+1) = G + P
           G = C * R - B
  200   CONTINUE
!
        W(L) = W(L) - P
        RV1(L) = G
        RV1(M) = 0.0E0
        GO TO 105
!     .......... RECOVER FROM UNDERFLOW ..........
  210   W(I+1) = W(I+1) - P
        RV1(M) = 0.0E0
        GO TO 105
!     .......... ORDER EIGENVALUES ..........
  215   IF (L .EQ. 1) GO TO 250
!     .......... FOR I=L STEP -1 UNTIL 2 DO -- ..........
        DO 230 II = 2, L
           I = L + 2 - II
           IF (P .GE. W(I-1)) GO TO 270
           W(I) = W(I-1)
           IND(I) = IND(I-1)
  230   CONTINUE
!
  250   I = 1
  270   W(I) = P
        IND(I) = JTAG
  290 END DO
!
      GO TO 1001
!     .......... SET ERROR -- NO CONVERGENCE TO AN
!                EIGENVALUE AFTER 30 ITERATIONS ..........
 1000 IERR = L
 1001 RETURN
      END
!*
      FUNCTION PYTHAG(A,B)
      IMPLICIT REAL*8 (A-H,O-Z)
!!    REAL A,B
!
!     FINDS SQRT(A**2+B**2) WITHOUT OVERFLOW OR DESTRUCTIVE UNDERFLOW
!
!!    REAL P,R,S,T,U
      P = DMAX1(ABS(A),ABS(B))
      IF (P .EQ. 0.0E0) GO TO 20
      R = (DMIN1(ABS(A),ABS(B))/P)**2
   10 CONTINUE
        T = 4.0E0 + R
        IF (T .EQ. 4.0E0) GO TO 20
        S = R/T
        U = 1.0E0 + 2.0E0*S
        P = U*P
```



```
         R = (S/U)**2 * R
      GO TO 10
   20 PYTHAG = P
      RETURN
      END
!*
!*
       SUBROUTINE DOTC(AR,AI,BR,BI,ZR,ZI,N)
!*     CALCULATES THE DOT PRODUCT OF TWO COMPLEX VECTORS <A|B>
      IMPLICIT REAL*8 (A-H,O-Z)
!      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       PARAMETER ( ZERO = 0.0D0)
       DIMENSION AR(N),AI(N),BR(N),BI(N)
       ZR = ZERO
       ZI = ZERO
       DO 10 I=1,N
         ZR = ZR + AR(I)*BR(I) + AI(I)*BI(I)
         ZI = ZI + AR(I)*BI(I) - AI(I)*BR(I)
   10  CONTINUE
!      DO 11 I=1,N
!        ZR = ZR + AI(I)*BI(I)
!11    CONTINUE
!      DO 12 I=1,N
!        ZI = ZI + AR(I)*BI(I) - AI(I)*BR(I)
!12    CONTINUE
!      DO 13 I=1,N
!        ZI = ZI - AI(I)*BR(I)
!13    CONTINUE
       RETURN
      END
!234567890
       SUBROUTINE NORMC(AR,AI,ZR,N)
!*     CALCULATES THE NORM OF COMPLEX VECTOR <A|A>
      IMPLICIT REAL*8 (A-H,O-Z)
!      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       PARAMETER ( ZERO = 0.0D0)
       DIMENSION AR(N),AI(N)
       ZR = ZERO
       DO 10 I=1,N
!        ZR = ZR + AR(I)**2 + AI(I)**2
         ZR = ZR + AR(I)*AR(I) + AI(I)*AI(I)
   10  CONTINUE
!      DO 11 I=1,N
!         ZR = ZR + AI(I)*AI(I)
!  11  CONTINUE
       RETURN
      END
!234567890
       SUBROUTINE DOTR(AR,AI,BR,BI,ZR,N)
!*     CALCULATES THE DOT PRODUCT OF TWO COMPLEX VECTORS <A|B>
      IMPLICIT REAL*8 (A-H,O-Z)
!      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       PARAMETER ( ZERO = 0.0D0)
       DIMENSION AR(N),AI(N),BR(N),BI(N)
       ZR = ZERO
       DO 10 I=1,N
         ZR = ZR + AR(I)*BR(I) + AI(I)*BI(I)
   10  CONTINUE
!      DO 11 I=1,N
!        ZR = ZR + AI(I)*BI(I)
!11    CONTINUE
       RETURN
      END
!234567890
       SUBROUTINE DOTI(AR,AI,BR,BI,ZI,N)
!*     CALCULATES THE DOT PRODUCT OF TWO COMPLEX VECTORS <A|B>
      IMPLICIT REAL*8 (A-H,O-Z)
!      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       PARAMETER ( ZERO = 0.0D0)
```

```
      DIMENSION AR(N),AI(N),BR(N),BI(N)
      ZI = ZERO
      DO 12 I=1,N
      ZI = ZI + AR(I)*BI(I) - AI(I)*BR(I)
   12 CONTINUE
      RETURN
     END
!
!
!
!
```