NUMERICAL METHODS FOR TOEPLITZ MATRICES

VOLUME II

APPENDIX - PROGRAMS AND RESULTS

VOLUME II

APPENDIX - PROGRAMS AND RESULTS

Note on machine used

The programs listed in this volume were run on an IBM 370/3033, and all floating-point variables were in IBM double-precision ("REAL*8") format. In this format, one bit is used for the sign, 7 bits represent the hexadecimal exponent in excess-64 form, and the remaining 56 bits (about 16.8 decimal digits) are used for the mantissa.

```
C  TEST PROGRAM FOR PBA1 (PIVOTED BAREISS ALG, STRATEGY 1:
C                            SIMPLE 2-CHOICE STRATEGY)

C  1. MAKES UP N X N TOEP.MX. 'A' WITH A(N,1),...,A(1,1),...,
C     ...,A(1,N) EQUAL TO T0(1),...,T0(2*N-1) RESPECTIVELY.
C  2. MAKES SPECIFIED SUB-MATRIX OF ORDER NSIZE ILL-COND.
C     BY ADJUST RELEVANT TOEP.DIAG.
C  3. CALCS. RHS. CORRESP. TO SOL. (1,1,...,1)
C  4. SOLVES SYSTEM USING BNA AND PBA1, AND PRINTS OUT ERRORS

      REAL*8 T(11),A(20,20),B(20),X(20),XERR(20),XERRSZ,EPS
      REAL*8 DELTA(2)
      DATA T/3.,2.,6.,1.,8.,4.,6.,5.,5.,3.,1./,DELTA/0.,1D-3/

C  MATRIX DIMENSIONS
      ID1=20
      N=6
      N2M1=2*N-1
      NM1=N-1

C  GEN. 'A' FROM ITS 1ST ROW AND COL, GET SPECS. FOR ILL-COND
C  SUBMATRIX, AND ADJUST TOEP.DIAG. ACCORDINGLY
      CALL TOEPMX(T,A,N,N2M1,ID1)
      WRITE(6,3)
    3 FORMAT(/' SIZE OF ILL-COND BLOCK, DIST FROM SINGULARITY')
      READ(5,*) NSIZE,EPS
      CALL ADJSMX(T,A,N,N2M1,ID1,NSIZE,1,1,EPS)

    9 WRITE(6,10)
   10 FORMAT(///' INPUT TOEPLITZ MATRIX T')
      CALL PRNTMX(A,N,N,ID1)
      WRITE(6,15)
   15 FORMAT(/////' EXACT SOL OF T*X = B IS (1,1,...,1)')

C  GENERATE RHS FOR SOL=(1,1,...,1)
      DO 20 I=1,N
      B(I)=0.
      DO 20 J=1,N
   20 B(I)=B(I)+A(I,J)

C  SOLVE SYSTEM AND PRINT DIFFERENCE VECTOR FOR BNA (WHEN
C  DELTA=0) AND PBA-1
      DO 50 IDEL=1,2
      CALL PBA1(A,B,X,DELTA(IDEL),N,ID1)
      XERRSZ=0.
         DO 30 I=1,N
         XERR(I)=X(I)-1.
         IF(DABS(XERR(I)).GT.XERRSZ) XERRSZ=DABS(XERR(I))
   30    CONTINUE
      IF(IDEL.EQ.1) WRITE (6,35)
   35 FORMAT(///' BAREISS ALG (BNA):')
      IF(IDEL.EQ.2) WRITE (6,38)
   38 FORMAT(///' PIVOTED BAREISS ALG (PBA1):')
      WRITE(6,40) XERRSZ,(XERR(I),I=1,N)
   40 FORMAT(/' ERROR NORM = ',D12.4/' ERROR VECTOR',
     1  /(1X,10D12.4))
   50 CONTINUE
```

```
      STOP
      END


      SUBROUTINE PBA1(T0,B,X,DELTA,N,LR0)
C
C  PIVOTED BAREISS ALG, STRATEGY 1 (SIMPLE 2-CHOICE STRATEGY)
C
C  T0 - I/P TOEP.MX.; B - RHS; X - SOL(O/P); DELTA - PIVOT THRESHOLD;
C  N,LR0 - ORDER,ROW-DIM OF T0
C
C  OUTLINE OF ALG.:
C  1. CARRY OUT BNA UNTIL AT STEP (-K) TM-PIVOT/TMAX.LT.DELTA,
C     OR STEP N-1 IS FINISHED, AT WHICH CASE, EXIT. IF
C     TM-PIVOT/TMAX.LT.DELTA, GO TO 2 IF THE ALTERNATE PIVOT
C     (THE ELEMENT BELOW THE ZERO-BAND) IS GREATER THAN THE
C     THE BNA PIVOT, ELSE GO BACK TO THE BEGINNING OF THIS STEP
C
C  2. WITH TM(-K-1) AS PIVOT (NOTE. A(J) DENOTES J-TH DIAG. OF
C     TOEPLITZ PART OF A) ELIMINATE TP(0).
C
C  3. WITH TP(K) AS PIVOT, ELIMINATE TM(0)
C
C  4. RESTORE TO BAREISS FORM BY USING TM(1) TO ELIMINATE TP(K)
C
C  5. REPLACE K WITH K+1
C     (RBF)
C
C  6. GO TO 1.
C
C  TEMPORARIES:
C  T - COPY OF T0
C  TM,TP - NEGATIVE-INDEX (N.I.) AND POSITIVE-INDEX (P.I.) BAREISS
C          ITERATES.
C  BM,BP - RHS'S FOR TM,TP
C  U,BU - UPR.TRIANGLE AND CORRESP RHS FOR BACK-SUBST
C  TX - TM-MAX
C  IMT,IMB - TOP,BOTTOM ROW OF TOEP.PART OF TM
C  IPT,IPB -  ..    ..   ..  ..   ..   ..   TP
C  MM,MP - MULTIPLIERS FOR BAREISS OPERATIONS

      REAL*8 T0(LR0,N),B(N),X(N),T(20,20),TM(20,20),TP(20,20),
     1  BM(20),BP(20),U(20,20),BU(20),MM,MP
      REAL*8 TX,DELTA

      LR=20
      NM1=N-1

C  INITIALIZATION
      DO 2 I=1,N
      DO 2 J=1,N
    2 T(I,J)=T0(I,J)

      CALL MOVMTX(T,TP,N,N,LR)
      CALL MOVMTX(T,TM,N,N,LR)
      CALL MOVMTX(B,BP,N,1,LR)
```

```
      CALL MOVMTX(B,BM,N,1,LR)

      CALL UTROW(TM,BM,U,BU,0,1,0,N,LR)

      IPT=1
      IMB=N

C  MAIN LOOP. K IS CYCLE NO.
      K=0
   10 K=K+1
      IF(K.EQ.N) GO TO 500

C  STEP (-K) OF BNA-CYCLE
      IMT=K+IMB-N
      IPB=N-K+IPT
      CALL BAROP(TP,BP,IPT,IPB-1,IPT,1,TM,BM,IMT+1,MM,N,LR)
      IMT=IMT+1
      CALL UTROW(TM,BM,U,BU,K,IMT,0,N,LR)

C  PIVOT IF REQUIRED
      TX=TMAX(TM,IMT,IMB,N,LR)
      IF(DABS(TM(IMT,K+1))/TX.LT.DELTA  .AND.
     1   DABS(TM(IMT+1,1)).GT.DABS(TM(IMT,K+1))) GO TO 100

C  STEP (K) OF BNA-CYCLE
      CALL BAROP(TM,BM,IMT,IMB,IMT,K+1,TP,BP,IPT,MP,N,LR)
      GO TO 10


C  PIVOTING PROCEDURE, STRATEGY 1

C  USE TM(-K-1) TO ELIM TP(0). (NOTE: A(J) DENOTES JTH
C  DIAGONAL OF TOEPLITZ PART OF A)
  100 CALL BAROP(TM,BM,IMT+1,IMB,IMT+1,1,TP,BP,IPT,MP,N,LR)
      CALL MOVMTX(TP(IPT,1),U(K+1,1),1,N,LR)
      BU(K+1)=BP(IPT)

C  USE TP(K) TO ELIM. TM(0)
      CALL BAROP(TP,BP,IPT,IPB-2,IPT,K+1,TM,BM,IMT,MM,N,LR)
      IMB=IMB-1
      CALL UTROW(TM,BM,U,BU,K+1,IMT,0,N,LR)

C  FINALLY, RESTORE TO BAREISS FORM BY USING TM(1) TO
C  ELIM. TP(K)
      CALL BAROP(TM,BM,IMT,IMB-1,IMT,K+2,TP,BP,IPT+1,MP,N,LR)
      IPT=IPT+1
      K=K+1
      GO TO 10


  500 CALL BCKSUB(U,BU,X,N,LR)
      RETURN
      END



      SUBROUTINE BAROP(TS,BS,IS1,IS2,IP,JP,TD,BD,ID1,M,N,LR)
```

```
C
C  PERFORMS A BAREISS-OPERATION (REAL*8)

C  TS,BS .... SOURCE MATRIX AND RHS
C  IS1,IS2 .. 1ST AND LAST ROWS OF SOURCE BLOCK
C  IP,JP .... INDICES OF PIVOT
C  TD,BD .... DESTINATION NATRIX AND RHS
C  ID1 ...... 1ST ROW OF DESTINATION BLOCK
C  M (O/P) .. MULTIPLIER (REAL*8)
C  N ........ ORDER OF MATRICES
C  LR ....... ROW-DIMENSION OF MATRICES

      REAL*8 M,TS(LR,N),TD(LR,N),BS(N),BD(N)

      M=TD(IP+ID1-IS1,JP)/TS(IP,JP)

      ID=ID1
      DO 20 IS=IS1,IS2
      DO 10 J=1,N
   10 TD(ID,J)=TD(ID,J) - M*TS(IS,J)
      BD(ID)=BD(ID) - M*BS(IS)
   20 ID=ID+1
C     CALL DUMP(TS,TD,N,LR)
      RETURN
      END



      DOUBLE PRECISION FUNCTION TMAX(TI,IT,IB,N,LR)
C
C  FUNC. TO FIND LGST.MAGN.OF ELEMENTS IN TOEP.PART OF TI
C
C  IT,IB - TOP,BOTTOM ROWS OF TOEP.PART OF TI
C  N,LR - ORDER,ROW DIM.OF TI

      REAL*8 TI(LR,N),TMAG

      TMAX=0.
      DO 10 J=2,N
      TMAG=DABS(TI(IT,J))
      IF(TMAG.GT.TMAX) TMAX=TMAG
   10 CONTINUE

      DO 20 I=IT,IB
      TMAG=DABS(TI(I,1))
      IF(TMAG.GT.TMAX) TMAX=TMAG
   20 CONTINUE
      RETURN
      END


      SUBROUTINE BCKSUB(U,B,X,N,LR)
C
C  S/R TO BACKSUBST THE TRIANGULAR SYSTEM U*X=B.
C
C  N,LR - ORDER, ROW-DIM OF U
```

```
      REAL*8 U(LR,N),B(N),X(N)

      I=N
      DO 20 II=1,N
      X(I)=B(I)
      IF(I.EQ.N) GO TO 15
      IP1=I+1
      DO 10 J=IP1,N
   10 X(I)=X(I)-U(I,J)*X(J)
   15 X(I)=X(I)/U(I,I)
   20 I=I-1
      RETURN
      END


      SUBROUTINE DUMP(TS,TD,N,LR)
C
C  DIAGNOSTIC DUMP OF TS AND TD. (N,LR ARE THE ORDER
C  AND ROW-DIMENSION OF TS AND TD)

      REAL*8 TS(LR,N),TD(LR,N)

      WRITE(6,10)
   10 FORMAT(/' TS')
      DO 20 I=1,N
   20 WRITE(6,30) (TS(I,J),J=1,N)
   30 FORMAT(1X,13F9.3)

      WRITE(6,40)
   40 FORMAT(/' TD')
      DO 50 I=1,N
   50 WRITE(6,30) (TD(I,J),J=1,N)
      RETURN
      END



      SUBROUTINE UTROW(TM,BM,U,BU,K,IMT,ZD,N,LR)

C  S/R TO GET NEXT ROW OF U-MATRIX (AND RHS) FOR BACK-SUBST
C
C  TM - BAREISS ITERATE(-K)
C  BM - CORRESP RHS
C  U  - (O/P) U.T. MATRIX FOR BACK-SUBST
C  B  - CORRESP RHS
C  K  - CYCLE NO.
C  IMT- TOP OF TOEP.PT.OF TM
C  ZD - (INTEGER) ZERO-BAND DISPL. - MUST BE LE.1
C  N,LR-ORDER,ROW DIM OF MATRICES

      REAL*8 TM(LR,N),BM(N),U(LR,N),BU(N)
      INTEGER ZD

      IF(ZD.GE.0) GO TO 20
      WRITE(6,10) ZD,K,IMT
   10 FORMAT(' ABORT - ZD.LT.0'/' ZD =',I3,' K =',I2,' IMT =',I2)
      STOP
```

```
   20 CALL MOVMTX(TM(IMT+ZD,1),U(K+1,1),1,N,LR)
      BU(K+1)=BM(IMT+ZD)
      RETURN
      END


      SUBROUTINE ADJSMX(T,A,N,N2M1,LR,NSIZE,NCOL,NROW,EPS)

C  S/R TO MAKE SOME ORDER-NSIZE SUBMATRICES OF AN N X N
C  TOEP.MX. ILL-COND. BY ADJUSTING TOEP.DIAGS.
C
C  T - INITIAL VALUE OF A(N,1)...A(1,1)...A(1,N)
C  A - I/P TOEP.MX.
C  N - ORDER OF A
C  N2M1 - 2*N-1
C  LR   - ROW-DIM OF A
C  NSIZE- SIZE OF ILL-COND S/MATRICES
C  NCOL - NO.OF ILL-COND S/MATRICES ALONG COL, STARTING AT TOP
C  NROW - NO.OF ILL-COND S/MATRICES ALONG ROW, STARTING AT LEFT
C  EPS  - QTY TO BE ADDED TO DIAG AFTER EACH BLOCK HAS BEEN
C         MADE SINGULAR

      REAL*8 T(N2M1),A(LR,N),EPS
      REAL*8 WKA(860)

      IF(NSIZE.EQ.0.OR.NCOL.EQ.0.OR.NROW.EQ.0) RETURN
      N2P3=2*N+3
      DO 4 I=1,NCOL
      CALL SINGSM(A,N,N,LR,I+NSIZE-1,1,NSIZE,-1,1,WKA,N2P3,IER)
      T(N+NSIZE+I-2)=A(I+NSIZE-1,1)+EPS
    4 CALL TOEPMX(T,A,N,N2M1,LR)

      IF(NROW.EQ.1) RETURN
      DO 6 J=2,NROW
      CALL SINGSM(A,N,N,LR,1,J+NSIZE-1,NSIZE,1,-1,WKA,N2P3,IER)
      IF(IER.NE.0) WRITE(6,5) IER
    5 FORMAT(' SINGSM ERROR CODE = ',I3)
      T(N-NSIZE-J+2)=A(1,J+NSIZE-1)+EPS
    6 CALL TOEPMX(T,A,N,N2M1,LR)
      RETURN
      END


      SUBROUTINE MOVMTX(A,B,M,N,LR)
C
C  MOVES D.P. MATRIX A(M X N) TO B. LR IS THE ROW-DIMENSION
C  OF A AND B.

      REAL*8 A(LR,N),B(LR,N)

      DO 10 I=1,M
      DO 10 J=1,N
   10 B(I,J)=A(I,J)
      RETURN
```

```
SIZE OF ILL-COND BLOCK, DIST FROM SINGULARITY
?
3,5d-8


INPUT TOEPLITZ MATRIX T
    4.000    8.000    1.000    6.000    2.000    3.000
    6.000    4.000    8.000    1.000    6.000    2.000
    4.733    6.000    4.000    8.000    1.000    6.000
    5.000    4.733    6.000    4.000    8.000    1.000
    3.000    5.000    4.733    6.000    4.000    8.000
    1.000    3.000    5.000    4.733    6.000    4.000



EXACT SOL OF T*X = B IS (1,1,...,1)


BAREISS ALG (BNA):

ERROR NORM =   0.6910D-01
ERROR VECTOR
  0.6910D-01 -0.3516D-01 -0.5120D-01 -0.6696D-02  0.1674D-01  0.2091D-01


PIVOTED BAREISS ALG (PBA1):

ERROR NORM =   0.1110D-14
ERROR VECTOR
 -0.8327D-15  0.1110D-14  0.8882D-15 -0.6661D-15 -0.7772D-15 -0.2637D-15
READY
```

```
C  TEST PROGRAM FOR PBA2 (PIVOTED BAREISS ALG, STRATEGY 2)

C  1. MAKES UP N X N TOEP.MX. 'A' WITH A(N,1),...,A(1,1),...,
C     ...,A(1,N) EQUAL TO T0(1),...,T0(2*N-1) RESPECTIVELY.
C  2. MAKES SPECIFIES SUB-MATRICES OF ORDER NSIZE ILL-COND.
C     BY ADJUST RELEVANT TOEP.DIAGS.
C  3. CALCS. RHS. CORRESP. TO SOL. (1,1,...,1)
C  4. SOLVES SYSTEM USING BNA AND PBA2, AND PRINTS OUT ERRORS

      REAL*8 T(25),T0(25),A(20,20),B(20),X(20),XERR(20),XERRSZ,EPS
      REAL*8 DELTA(2),K1
      DATA T0/-1.5,10.,1.,-7.,-2.,-5.,3.,7.,9.,2.,6.,-1.,5.,
     1  1.,-3.,0.,0.,-4.,-7.,-1.,2.,1.,-6.,1.,-0.5/,DELTA/0.,1D-3/,
     2  K1/100./

C  MATRIX DIMENSIONS
      ID1=20
      N=13
      N2M1=2*N-1
      NM1=N-1

C  GEN. 'A' FROM ITS 1ST ROW AND COL, GET SPECS. FOR ILL-COND
C  SUBMATRICES, AND ADJUST TOEP.DIAGS. ACCORDINGLY
    1 DO 2 I=1,N2M1
    2 T(I)=T0(I)
      CALL TOEPMX(T,A,N,N2M1,ID1)
      WRITE(6,3)
    3 FORMAT(/' SIZE OF ILL-COND BLOCK, NO.ALONG COL,ROW, DIST ',
     1        'FROM SINGULARITY')
      READ(5,*) NSIZE,NCOL,NROW,EPS
      CALL ADJSMX(T,A,N,N2M1,ID1,NSIZE,NCOL,NROW,EPS)

    9 WRITE(6,10)
   10 FORMAT(///' INPUT TOEPLITZ MATRIX T')
      CALL PRNTMX(A,N,N,ID1)
      WRITE(6,15)
   15 FORMAT(/////' EXACT SOL OF T*X = B IS (1,1,...,1)')

C  GENERATE RHS FOR SOL=(1,1,...,1)
      DO 20 I=1,N
      B(I)=0.
      DO 20 J=1,N
   20 B(I)=B(I)+A(I,J)

C  SOLVE SYSTEM AND PRINT DIFFERENCE VECTOR FOR BNA (WHEN
C  DELTA=0) AND PBA-1
      DO 50 IDEL=1,2
      CALL PBA2(A,B,X,DELTA(IDEL),K1,N,ID1)
      XERRSZ=0.
         DO 30 I=1,N
         XERR(I)=X(I)-1.
         IF(DABS(XERR(I)).GT.XERRSZ) XERRSZ=DABS(XERR(I))
   30    CONTINUE
      IF(IDEL.EQ.1) WRITE (6,35)
   35 FORMAT(///' BAREISS ALG (BNA):')
      IF(IDEL.EQ.2) WRITE (6,38)
   38 FORMAT(///' PIVOTED BAREISS ALG (PBA2):')
```

```
      WRITE(6,40) XERRSZ,(XERR(I),I=1,N)
   40 FORMAT(/' ERROR NORM = ',D12.4/' ERROR VECTOR',
     1  /(1X,10D12.4))
   50 CONTINUE
      STOP
      END



      SUBROUTINE PBA2(T0,B,X,DELTA,K1,N,LR0)
C
C  PIVOTED BAREISS ALG, STRATEGY 2 (SIMPLIFIED VERSION)
C
C  T0 - I/P TOEP.MX.; B - RHS; X - SOL(O/P); DELTA - PIVOT THRESHOLD;
C  K1 - (REAL*8) PIVOT PARAMETER (SEE BELOW); N - ORDER OF T0;
C  LR0 - ROW DIM. OF T0
C
C  N.B.: THIS IS A SLIGHTLY SIMPLIFIED VERSION OF THE PBA-2
C  DESCRIBED IN CH.5, IN THAT WE ONLY SEARCH BELOW THE ZERO-BAND
C  FOR A PIVOT (INSTEAD OF BOTH ABOVE AND BELOW). THIS PROGRAM
C  CAN EASILY BE EXTENDED TO USE PIVOTS ABOVE THE ZERO-BAND AS
C  WELL, BUT THE BACK-SUBST PROCEDURE MUST BE GENERALIZED IN
C  ACCORDANCE WITH THE REMARKS IN CH.5.
C
C  OUTLINE OF ALG.:
C  1. CARRY OUT BNA UNTIL AT STEP (-K) TM-PIVOT/TMAX.LT.DELTA,
C     OR STEP N-1 IS FINISHED, AT WHICH CASE, EXIT. IF
C     T-PIVOT/TMAX.LT.DELTA, DO:
C
C  2. FIND TM(P)  P PLACES ABOVE THE ZERO-BAND (Z.B.) AND TM(-K-1-Q)
C     Q PLACES BELOW THE Z.B. SATISFYING :
C     (A) TM(.)/TMAX.GE.DELTA
C     (B) TM(.).GE.K1*(SUM OF ELEMENTS BETW. TM(.) AND Z.B.)
C
C  3. RECOVER ITERATES +,-(K-Q-1) AND DO Q+1 C-CYCLES TO MOVE Z.B.
C     DOWN
C
C  4. DO P+Q B-CYCLES (BNA-CYCLES) TO GO ALONG BOTTOM OF BLOCK
C     OF ILL-COND SUBMATRICES
C
C  5. DO Q+1 A-CYCLES TO MOVE THE Z.B. AND RESTORE TO BAREISS FORM
C     (RBF)
C
C  6. GO TO 1.
C
C  TEMPORARIES:
C  T - COPY OF T0
C  TM,TP - NEGATIVE-INDEX (N.I.) AND POSITIVE-INDEX (P.I.) BAREISS
C          ITERATES.
C  TMSV,TPSV - MATRICES TO SAVE P.I. AND N.I. ITERATES
C  BM,BP - RHS'S FOR TM,TP
C  BMSV,BPSV - MATRICES TO SAVE TM'S AND TP'S
C  U,BU - UPR.TRIANGLE AND CORRESP RHS FOR BACK-SUBST
C  TX - TM-MAX
C  IMT,IMB - TOP,BOTTOM ROW OF TOEP.PART OF TM
C  IPT,IPB -  ..    ..   ..  ..   ..   ..   TP
C  ZD(INTEGER) - Z.B. DISPLACEMENT
```

```
C  P,Q - POSITIONS OF ACCEPTABLE PIVOTS ABOVE AND BELOW Z.B.

      INTEGER ZD,P,Q,PP1,QP1
      REAL*8 T0(LR0,N),B(N),X(N),T(20,20),TM(20,20),TP(20,20),
     1  BM(20),BP(20),TMSV(20,20),TPSV(20,20),BMSV(20,20),
     2  BPSV(20,20),U(20,20),BU(20)
      REAL*8 TX,DELTA,K1

      LR=20
      NM1=N-1

C  INITIALIZATION
      DO 2 I=1,N
      DO 2 J=1,N
    2 T(I,J)=T0(I,J)

      CALL MOVMTX(T,TP,N,N,LR)
      CALL MOVMTX(T,TM,N,N,LR)
      CALL MOVMTX(B,BP,N,1,LR)
      CALL MOVMTX(B,BM,N,1,LR)

      CALL UTROW(TM,BM,U,BU,0,1,0,N,LR)
      CALL SAVE(TM,TP,BM,BP,TMSV,TPSV,BMSV,BPSV,0,N,1,N,LR)

      IPT=1
      IMB=N
      ZD=0

C  MAIN LOOP. K IS CYCLE NO.
      K=0
   10 K=K+1
      IF(K.EQ.N) GO TO 500

C  BNA-CYCLE
      CALL BCYC(TM,BM,IMB,TP,BP,IPT,K,ZD,N,LR)
      CALL SAVE(TM,TP,BM,BP,TMSV,TPSV,BMSV,BPSV,K,IMB,IPT,N,LR)
      IMT=IMB+K-N+1
      CALL UTROW(TM,BM,U,BU,K,IMT,ZD,N,LR)

C  PIVOT IF REQUIRED
      TX=TMAX(TM,IMT,IMB,N,LR)
      IF (DABS(TM(IMT+ZD,K+1))/TX.LT.DELTA) GO TO 100
      GO TO 10


C  PIVOTING PROCEDURE, STRATEGY 2

C  FIND ACCEPTABLE PIVOTS ABOVE AND BELOW Z.B.
  100 CALL FINDPV(TM(IMT,K+1),LR*(N-K),LR,N-K,TX,DELTA,K1,PP1,IERR)
      P=PP1-1
      CALL FINDPV(TM(IMT+1,1),N-K-1,1,N-K-1,TX,DELTA,K1,QP1,IERR)
      Q=QP1-1

C  MAKE TM(-K-1-Q) AVAILABLE FOR PIVOTING BY GETTING ITERATES
C  +,-(-K-Q-1) AND PERFORMING Q+1 C-CYLES
      NCCYC=Q+1
      K=K-NCCYC
```

```
      CALL RECOV(TMSV,TPSV,BMSV,BPSV,TM,TP,BM,BP,K,IMB,IPT,N,LR)
         DO 140 I=1,NCCYC
         K=K+1
         CALL CCYC(TM,BM,IMB,TP,BP,IPT,K,ZD,N,LR)
  140    CALL UTROW(TM,BM,U,BU,K,IMB-N+K+1,ZD,N,LR)

C . NEXT DO P+Q BNA-CYCLES (B-CYCLES)
      NBCYC=P+Q
         DO 150 I=1,NBCYC
         K=K+1
         CALL BCYC(TM,BM,IMB,TP,BP,IPT,K,ZD,N,LR)
  150    CALL UTROW(TM,BM,U,BU,K,IMB-N+K+1,ZD,N,LR)

C  FINALLY DO Q+1 (=NCCYC) A-CYCLES TO RESTORE TO BAREISS FORM
         DO 160 I=1,NCCYC
         K=K+1
         CALL ACYC(TM,BM,IMB,TP,BP,IPT,K,ZD,N,LR)
  160    CALL UTROW(TM,BM,U,BU,K,IMB-N+K+1,ZD,N,LR)
         GO TO 10


  500 CALL BCKSUB(U,BU,X,N,LR)
      RETURN
      END



      SUBROUTINE MOVMTX(A,B,M,N,LR)
C
C  MOVES D.P. MATRIX A(M X N) TO B. LC IS THE ROW-DIMENSION
C  OF A AND B.

      REAL*8 A(LR,N),B(LR,N)

      DO 10 I=1,M
      DO 10 J=1,N
   10 B(I,J)=A(I,J)
      RETURN
      END



      SUBROUTINE BAROP(TS,BS,IS1,IS2,IP,JP,TD,BD,ID1,M,N,LR)
C
C  PERFORMS A BAREISS-OPERATION (REAL*8)

C  TS,BS .... SOURCE MATRIX AND RHS
C  IS1,IS2 .. 1ST AND LAST ROWS OF SOURCE BLOCK
C  IP,JP .... INDICES OF PIVOT
C  TD,BD .... DESTINATION NATRIX AND RHS
C  ID1 ...... 1ST ROW OF DESTINATION BLOCK
C  M (O/P) .. MULTIPLIER (REAL*8)
C  N ........ ORDER OF MATRICES
C  LR ....... ROW-DIMENSION OF MATRICES

      REAL*8 M,TS(LR,N),TD(LR,N),BS(N),BD(N)
```

```
      M=TD(IP+ID1-IS1,JP)/TS(IP,JP)

      ID=ID1
      DO 20 IS=IS1,IS2
      DO 10 J=1,N
   10 TD(ID,J)=TD(ID,J) - M*TS(IS,J)
      BD(ID)=BD(ID) - M*BS(IS)
   20 ID=ID+1
C     CALL DUMP(TS,TD,N,LR)
      RETURN
      END



      DOUBLE PRECISION FUNCTION TMAX(TI,IT,IB,N,LR)
C
C  FUNC. TO FIND LGST.MAGN.OF ELEMENTS IN TOEP.PART OF TI
C
C  IT,IB - TOP,BOTTOM ROWS OF TOEP.PART OF TI
C  N,LR - ORDER,ROW DIM.OF TI

      REAL*8 TI(LR,N),TMAG

      TMAX=0.
      DO 10 J=2,N
      TMAG=DABS(TI(IT,J))
      IF(TMAG.GT.TMAX) TMAX=TMAG
   10 CONTINUE

      DO 20 I=IT,IB
      TMAG=DABS(TI(I,1))
      IF(TMAG.GT.TMAX) TMAX=TMAG
   20 CONTINUE
      RETURN
      END


      SUBROUTINE BCKSUB(U,B,X,N,LR)
C
C  S/R TO BACKSUBST THE TRIANGULAR SYSTEM U*X=B.
C
C  N,LR - ORDER, ROW-DIM OF U

      REAL*8 U(LR,N),B(N),X(N)

      I=N
      DO 20 II=1,N
      X(I)=B(I)
      IF(I.EQ.N) GO TO 15
      IP1=I+1
      DO 10 J=IP1,N
   10 X(I)=X(I)-U(I,J)*X(J)
   15 X(I)=X(I)/U(I,I)
   20 I=I-1
      RETURN
      END
```

```
      SUBROUTINE DUMP(TS,TD,N,LR)
C
C  DIAGNOSTIC DUMP OF TS AND TD. (N,LR ARE THE ORDER
C  AND ROW-DIMENSION OF TS AND TD)

      REAL*8 TS(LR,N),TD(LR,N)

      WRITE(6,10)
   10 FORMAT(/' TS')
      DO 20 I=1,N
   20 WRITE(6,30) (TS(I,J),J=1,N)
   30 FORMAT(1X,13F9.3)

      WRITE(6,40)
   40 FORMAT(/' TD')
      DO 50 I=1,N
   50 WRITE(6,30) (TD(I,J),J=1,N)
      RETURN
      END


      SUBROUTINE BCYC(TM,BM,IMB,TP,BP,IPT,K,ZD,N,LR)
C
C  S/R TO DO A BAREISS CYCLE (B-CYCLE)

C  TM,BM - NEG.INDEX ITERATE AND RHS
C  IMB   - INDEX OF LAST ROW IN TOEP.PART OF TM
C  TP,TP - POS.INDEX ITERATE AND RHS
C  IPT   - INDEX OF 1ST ROW IN TOEP.PART OF TP
C  K     - CYCLE TO BE COMPUTED
C  ZD    - (INTEGER) I/P ZERO-BAND DISPLACEMENT
C  N,LR  - ORDER, ROW DIM OF MATRICES
C
C  PROCEDURE:
C  1. ELIM BELOW ZERO-BAND OF NEG.INDEX ITERATE
C  2. ELIM ABOVE ZERO-BAND OF POS.INDEX ITERATE

      INTEGER ZD
      REAL*8 MM,MP
      REAL*8 TM(LR,N),BM(N),TP(LR,N),BP(N)

      IMT=K+IMB-N
      IPB=N-K+IPT
      CALL BAROP(TP,BP,IPT,IPB-1,IPT+ZD,1,TM,BM,IMT+1,MM,N,LR)
      CALL BAROP(TM,BM,IMT+1,IMB,IMT+1,K+1-ZD,TP,BP,IPT,MP,N,LR)
      RETURN
      END


      SUBROUTINE CCYC(TM,BM,IMB,TP,BP,IPT,K,ZD,N,LR)

C  S/R TO MOVE ZERO-BAND DOWN (C-CYCLE)
C
C  PARAMETERS - AS FOR S/R 'BCYC'
```

```
C
C  PROCEDURE:
C  1. ELIM BELOW ZERO-BAND OF POS.INDEX ITERATE
C  2. ELIM BELOW ZERO-BAND OF NEG.INDEX ITERATE

      INTEGER ZD
      REAL*8 MM,MP
      REAL*8 TM(LR,N),BM(N),TP(LR,N),BP(N)

      IMT=K+IMB-N
      IPB=N-K+IPT
      CALL BAROP(TM,BM,IMT+1,IMB,IMT+ZD+1,1,TP,BP,IPT,MP,N,LR)
      CALL BAROP(TP,BP,IPT,IPB-1,IPT+ZD+1,1,TM,BM,IMT,MM,N,LR)

      ZD=ZD+1
      IMB=IMB-1
      RETURN
      END



      SUBROUTINE ACYC(TM,BM,IMB,TP,BP,IPT,K,ZD,N,LR)
C
C  S/R TO MOVE ZERO-BAND UP (A-CYCLE)
C
C  PARAMETERS - AS FOR S/R 'BCYC'
C
C  PROCEDURE:
C  1. ELIM ABOVE ZERO-BAND OF NEG.INDEX ITERATE
C  2. ELIM ABOVE ZERO-BAND OF POS.INDEX ITERATE

      INTEGER ZD
      REAL*8 MM,MP
      REAL*8 TM(LR,N),BM(N),TP(LR,N),BP(N)

      IMT=K+IMB-N
      IPB=N-K+IPT

      CALL BAROP(TP,BP,IPT,IPB-1,IPT,K+1-ZD,TM,BM,IMT+1,MM,N,LR)
      CALL BAROP(TM,BM,IMT+1,IMB,IMT+1,K+2-ZD,TP,BP,IPT+1,MP,N,LR)

      ZD=ZD-1
      IPT=IPT+1
      RETURN
      END



      SUBROUTINE SAVE(TM,TP,BM,BP,TMSV,TPSV,BMSV,BPSV,K,IMB,IPT,N,LR)

C  S/R TO SAVE TOEP.PART OF BAREISS ITERATES
C
C  1ST TOW OF TOEP PT.OF TM(ITER.-K) GOES TO TMSV(K+1,K+1)...TMSV(K+1,N)
C  1ST COL ..   ..   ..   ..   ..   ..   ..  TMSV(K+2,K+1)...TMSV(N,K+1)
C  CORRESP ENTRIES IN BM(ITER.-K) GO TO BMSV(K+1,K+1)...BMSV(N,K+1)
C  LAST ROW OF TOEP PT.OF TP(ITER.K) GOES TO TPSV(N-K,1)...TPSV(N-K,N-K)
C  LAST COL..   ..   ..   ..   ..   ..   .. TPSV(1,N-K)..TPSV(N-K-1,N-K)
```

```
C  CORRESP ENTRIES IN BP(ITER K) GO TO BPSV(1,K+1)...BPSV(N-K,K+1)

C IMB - BOTTOM OF TOEP.PT OF TM
C IPT - TOP OF TOEP.PT OF TP
C N,LR- ORDER,ROW-DIM OF MATRICES

      REAL*8 TM(LR,N),TP(LR,N),TMSV(LR,N),TPSV(LR,N),BM(N),BP(N),
     1 BMSV(LR,N),BPSV(LR,N)

      IMT=IMB+K-N+1
      CALL MOVMTX(TM(IMT,K+1),TMSV(K+1,K+1),1,N-K,LR)
      CALL MOVMTX(TM(IMT+1,1),TMSV(K+2,K+1),N-K-1,1,LR)
      CALL MOVMTX(BM(IMT),BMSV(K+1,K+1),N-K,1,LR)

      IPB=IPT+N-K-1
      CALL MOVMTX(TP(IPB,1),TPSV(N-K,1),1,N-K,LR)
      CALL MOVMTX(TP(IPT,N),TPSV(1,N-K),N-K-1,1,LR)
      CALL MOVMTX(BP(IPT),BPSV(1,K+1),N-K,1,LR)
      RETURN
      END


      SUBROUTINE RECOV(TMSV,TPSV,BMSV,BPSV,TM,TP,BM,BP,K,IMB,IPT,N,LR)
C
C  S/R TO RECOVER TOEP.PT. OF BAREISS ITERATES AS STORED IN TMSV,TPSV,
C  BMSV AND BPSV (SEE S/R 'SAVE')
C
C  TMSV,TPSV,BMSV,BPSV - 'HISTORY' OF TM,TP,BM,BP STORED AS IN S/R
C                                                          'SAVE'
C  TM,TP,BM,BP - BAREISS ITERATES (POS AND NEG INDICES) AND
C                                        RESPECTIVE RHS'S
C  K   - CYCLE NO.
C  IMB - (O/P) BOTTOM OF TOEP.PT.OF TM - SET TO N
C  IPT - (O/P) TOP OF TOEP.PT.OF TP - SET TO 1
C  N,LR- ORDER,ROW DIM OF MATRICES

      REAL*8 TM(LR,N),TP(LR,N),BM(N),BP(N),TMSV(LR,N),TPSV(LR,N),
     1  BMSV(LR,N),BPSV(LR,N)

      IMT=K+1
      IMB=N
      CALL MOVMTX(TMSV(K+1,K+1),TM(IMT,K+1),1,N-K,LR)
      CALL MOVMTX(TMSV(K+2,K+1),TM(IMT+1,1),N-K-1,1,LR)
      CALL MOVMTX(BMSV(K+1,K+1),BM(IMT),N-K,1,LR)

      IPT=1
      IPB=N-K
      CALL MOVMTX(TPSV(N-K,1),TP(IPB,1),1,N-K,LR)
      CALL MOVMTX(TPSV(1,N-K),TP(IPT,N),N-K-1,1,LR)
      CALL MOVMTX(BPSV(1,K+1),BP(IPT),N-K,1,LR)

C  FILL IN ZERO-BANDS AND TOEP.PTS.OF TM AND TP
      IF(K.EQ.0) GO TO 15
      DO 10 J=1,K
   10 TM(IMT,J)=0.
   15 IMTP1=IMT+1
```

```
      DO 20 I=IMTP1,IMB
      DO 20 J=2,N
   20 TM(I,J)=TM(I-1,J-1)

      NMKP1=N-K+1
      IF(K.EQ.0) GO TO 35
      DO 30 J=NMKP1,N
   30 TP(IPB,J)=0.
   35 IPBM1=IPB-1
      NM1=N-1
      I=IPBM1
      DO 50 II=IPT,IPBM1
      DO 40 J=1,NM1
   40 TP(I,J)=TP(I+1,J+1)
   50 I=I-1
      RETURN
      END



      SUBROUTINE UTROW(TM,BM,U,BU,K,IMT,ZD,N,LR)

C  S/R TO GET NEXT ROW OF U-MATRIX (AND RHS) FOR BACK-SUBST
C
C  TM - BAREISS ITERATE(-K)
C  BM - CORRESP RHS
C  U  - (O/P) U.T. MATRIX FOR BACK-SUBST
C  B  - CORRESP RHS
C  K  - CYCLE NO.
C  IMT- TOP OF TOEP.PT.OF TM
C  ZD - (INTEGER) ZERO-BAND DISPL. - MUST BE LE.1
C  N,LR-ORDER,ROW DIM OF MATRICES

      REAL*8 TM(LR,N),BM(N),U(LR,N),BU(N)
      INTEGER ZD

      IF(ZD.GE.0) GO TO 20
      WRITE(6,10) ZD,K,IMT
   10 FORMAT(' ABORT - ZD.LT.0'/' ZD =',I3,' K =',I2,' IMT =',I2)
      STOP

   20 CALL MOVMTX(TM(IMT+ZD,1),U(K+1,1),1,N,LR)
      BU(K+1)=BM(IMT+ZD)
      RETURN
      END



      SUBROUTINE FINDPV(TI,L,ISEP,NENTS,TX,DELTA,K1,P,IERR)

C  S/R TO FIND A PIVOT ACCORDING TO STRATEGY 2
C
C  TI   - VECTOR CONTAINING ELEMENTS TO BE SEARCHED
C  L    - LENGTH OF TI
C  ISEP - SEPARATION BETW. ELEMENTS TO BE SEARCHED
C  NENTS- NO.OF ELEMENTS TO BE SEARCHED
C  TX   - MAX ELT. OF TOEP.BLOCK BEING SEARCHED
```

```
C  DELTA- THRESHOLD: PIVOT MUST BE GE.DELTA*TX
C  K1    - (REAL*8) PIVOT PARAMETER : PIVOT MUST BE GE.K1*PREVIOUS
C                                                          ELTS.SEARCHED
C  P     - (O/P) POSITION OF PIVOT (IF FOUND)
C  IERR - 0 : PIVOT FOUND; 1 : PIVOT NOT FOUND

      INTEGER P
      REAL*8 TI(L),DELTA,K1,SUM,TX

      IERR=0
      SUM=0.
      I=1

      DO 20 P=1,NENTS
      IF(DABS(TI(I))/TX.LT.DELTA) GO TO 10
      IF(DABS(TI(I)).GE.K1*SUM) RETURN
   10 SUM=SUM+DABS(TI(I))
   20 I=I+ISEP

      IERR=1
      RETURN
      END



      SUBROUTINE ADJSMX(T,A,N,N2M1,LR,NSIZE,NCOL,NROW,EPS)

C  S/R TO MAKE SOME ORDER-NSIZE SUBMATRICES OF AN N X N
C  TOEP.MX. ILL-COND. BY ADJUSTING TOEP.DIAGS.
C
C  T - INITIAL VALUE OF A(N,1)...A(1,1)...A(1,N)
C  A - I/P TOEP.MX.
C  N - ORDER OF A
C  N2M1 - 2*N-1
C  LR    - ROW-DIM OF A
C  NSIZE- SIZE OF ILL-COND S/MATRICES
C  NCOL - NO.OF ILL-COND S/MATRICES ALONG COL, STARTING AT TOP
C  NROW - NO.OF ILL-COND S/MATRICES ALONG ROW, STARTING AT LEFT
C  EPS   - QTY TO BE ADDED TO DIAG AFTER EACH BLOCK HAS BEEN
C          MADE SINGULAR

      REAL*8 T(N2M1),A(LR,N),EPS
      REAL*8 WKA(860)

      IF(NSIZE.EQ.0.OR.NCOL.EQ.0.OR.NROW.EQ.0) RETURN
      N2P3=2*N+3
      DO 4 I=1,NCOL
      CALL SINGSM(A,N,N,LR,I+NSIZE-1,1,NSIZE,-1,1,WKA,N2P3,IER)
      T(N+NSIZE+I-2)=A(I+NSIZE-1,1)+EPS
    4 CALL TOEPMX(T,A,N,N2M1,LR)

      IF(NROW.EQ.1) RETURN
      DO 6 J=2,NROW
      CALL SINGSM(A,N,N,LR,1,J+NSIZE-1,NSIZE,1,-1,WKA,N2P3,IER)
      IF(IER.NE.0) WRITE(6,5) IER
    5 FORMAT(' SINGSM ERROR CODE = ',I3)
      T(N-NSIZE-J+2)=A(1,J+NSIZE-1)+EPS
```

```
    6 CALL TOEPMX(T,A,N,N2M1,LR)
      RETURN
      END
```

```
SIZE OF ILL-COND BLOCK, NO.ALONG COL,ROW, DIST FROM SINGULARITY
?
4,3,2,2d-3


INPUT TOEPLITZ MATRIX T
    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000   -2.000   -7.000    1.000   10.000   -1.500
    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000   -2.000   -7.000    1.000   10.000
   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000   -2.000   -7.000    1.000
   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000   -2.000   -7.000
  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000   -2.000
   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000
   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000
   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000
    2.000   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699
    1.000    2.000   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000
   -6.000    1.000    2.000   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000
    1.000   -6.000    1.000    2.000   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000
   -0.500    1.000   -6.000    1.000    2.000   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000


EXACT SOL OF T*X = B IS (1,1,...,1)


BAREISS ALG (BNA):

ERROR NORM =   0.2661D-01
ERROR VECTOR
  0.9628D-02  0.6009D-02 -0.6716D-02 -0.7315D-04  0.2661D-01  0.1394D-01 -0.2368D-01 -0.2004D-01  0.4591D-02  0.5767D-02
  0.1351D-02 -0.2150D-01  0.1127D-01


PIVOTED BAREISS ALG (PBA2):

ERROR NORM =   0.4330D-13
ERROR VECTOR
 -0.3327D-13 -0.3478D-13 -0.5551D-14  0.4663D-14 -0.2981D-13 -0.6536D-14  0.2642D-13  0.4197D-13  0.3553D-14 -0.8243D-14
  0.5107D-14  0.4330D-13 -0.5676D-14
READY
```

```
SIZE OF ILL-COND BLOCK, NO.ALONG COL,ROW, DIST FROM SINGULARITY
?
4,3,3,2d-6


INPUT TOEPLITZ MATRIX T
    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000   -2.000   -7.000    1.000   10.000   -1.500
    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000   -2.000   -7.000    1.000   10.000
   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000   -2.000   -7.000    1.000
   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000   -2.000   -7.000
  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000   -2.000
   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000
   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000
   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850
    2.000   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697
    1.000    2.000   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000
   -6.000    1.000    2.000   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000
    1.000   -6.000    1.000    2.000   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000
   -0.500    1.000   -6.000    1.000    2.000   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000



EXACT SOL OF T*X = B IS (1,1,...,1)


BAREISS ALG (BNA):

ERROR NORM =   0.1898D+00
ERROR VECTOR
  0.3581D-01  0.9108D-02  0.4447D-02  0.1255D+00  0.4220D-01 -0.1530D+00  0.8745D-01  0.9942D-02 -0.1898D+00 -0.4101D-02
  0.3750D-01 -0.6338D-01 -0.1216D+00


PIVOTED BAREISS ALG (PBA2):

ERROR NORM =   0.6040D-13
ERROR VECTOR
  0.8882D-14 -0.3536D-13 -0.3163D-13 -0.2642D-13 -0.6453D-14 -0.1586D-13  0.1443D-13  0.5884D-13  0.3020D-13 -0.4372D-14
 -0.1994D-13  0.6040D-13  0.1021D-13
READY
```

```
C  TEST PROGRAM FOR PBA3 IS THE SAME AS THAT FOR PBA2, EXCEPT THAT
C  PBA3 IS CALLED INSTEAD OF PBA2
C
      SUBROUTINE PBA3(T0,B,X,DELTA,K1,N,LR0)
C
C  PIVOTED BAREISS ALG, STRATEGY 3 (SIMPLIFIED VERSION)
C
C  T0 - I/P TOEP.MX.; B - RHS; X - SOL(O/P); DELTA - PIVOT THRESHOLD;
C  K1 - (REAL*8) PIVOT PARAMETER (SEE BELOW); N - ORDER OF T0;
C  LR0 - ROW DIM. OF T0
C
C  N.B.: THIS IS A SLIGHTLY SIMPLIFIED VERSION OF THE PBA-3
C  DESCRIBED IN CH.5, IN THAT WE ONLY SEARCH BELOW THE ZERO-BAND
C  FOR A PIVOT (INSTEAD OF BOTH ABOVE AND BELOW). THIS PROGRAM
C  CAN EASILY BE EXTENDED TO USE PIVOTS ABOVE THE ZERO-BAND AS
C  WELL, BUT THE BACK-SUBST PROCEDURE MUST BE GENERALIZED IN
C  ACCORDANCE WITH THE REMARKS IN CH.5.
C
C  OUTLINE OF ALG.:
C  1. CARRY OUT BNA UNTIL AT STEP (-K) TM-PIVOT/TMAX.LT.DELTA,
C     OR STEP N-1 IS FINISHED, AT WHICH CASE, EXIT. IF
C     T-PIVOT/TMAX.LT.DELTA, DO:
C
C  2. FIND TM(P)  P PLACES ABOVE THE ZERO-BAND (Z.B.) AND TM(-K-1-Q)
C     Q PLACES BELOW THE Z.B. SATISFYING :
C     (A) TM(.)/TMAX.GE.DELTA
C     (B) TM(.).GE.K1*(SUM OF ELEMENTS BETW. TM(.) AND Z.B.)
C
C  3. MOVE Z.B. DOWN Q+1 PLACES USING BACKTRACK PROC C (BPC).
C
C  4. DO P+Q B-CYCLES (BNA-CYCLES) TO GO ALONG BOTTOM OF BLOCK
C     OF ILL-COND SUBMATRICES
C
C  5. MOVE Z.B. UP Q+1 PLACES USING BPC TO RESTORE TO BAREISS
C     FORM (RBF).
C
C  6. GO TO 1.
C
C  TEMPORARIES:
C  T - COPY OF T0
C  TM,TP - NEGATIVE-INDEX (N.I.) AND POSITIVE-INDEX (P.I.) BAREISS
C          ITERATES.
C  TMSV,TPSV - MATRICES TO SAVE P.I. AND N.I. ITERATES
C  BM,BP - RHS'S FOR TM,TP
C  BMSV,BPSV - MATRICES TO SAVE TM'S AND TP'S
C  U,BU - UPR.TRIANGLE AND CORRESP RHS FOR BACK-SUBST
C  TX - TM-MAX
C  IMT,IMB - TOP,BOTTOM ROW OF TOEP.PART OF TM
C  IPT,IPB -  ..    ..   ..  ..   ..   ..   TP
C  ZD(INTEGER) - Z.B. DISPLACEMENT
C  P,Q - POSITIONS OF ACCEPTABLE PIVOTS ABOVE AND BELOW Z.B.
```

```
C  MM,MP -  1ST ROWS OF TOEP PTS OF MPYR MATRICES
C  MMSV,MPSV - VECTORS TO SAVE PREVIOUS MM,MP

      INTEGER ZD,P,Q,PP1,QP1,QM1
      REAL*8 TO(LR0,N),B(N),X(N),T(20,20),TM(20,20),TP(20,20),
     1  BM(20),BP(20),TMSV(20,20),TPSV(20,20),BMSV(20,20),
     2  BPSV(20,20),U(20,20),BU(20)
      REAL*8 MM(20),MP(20),MPSV(20),MMSV(20)
      REAL*8 TX,DELTA,K1

      LR=20
      NM1=N-1

C  INITIALIZATION
      DO 2 I=1,N
      DO 2 J=1,N
    2 T(I,J)=TO(I,J)

      CALL MOVMTX(T,TP,N,N,LR)
      CALL MOVMTX(T,TM,N,N,LR)
      CALL MOVMTX(B,BP,N,1,LR)
      CALL MOVMTX(B,BM,N,1,LR)

      CALL UTROW(TM,BM,U,BU,0,1,0,N,LR)
      CALL SAVE(TM,TP,BM,BP,TMSV,TPSV,BMSV,BPSV,0,N,1,N,LR)

      MP(1)=1.
      MM(1)=1.
      DO 5 I=2,N
      MP(I)=0.
    5 MM(I)=0.

      IPT=1
      IMB=N
      ZD=0

C  MAIN LOOP. K IS CYCLE NO.
      K=0
   10 K=K+1
      IF(K.EQ.N) GO TO 500

C  BNA-CYCLE
      CALL MOVMTX(MP,MPSV,N,1,LR)
      CALL MOVMTX(MM,MMSV,N,1,LR)
      CALL BCYCM(TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
      CALL SAVE(TM,TP,BM,BP,TMSV,TPSV,BMSV,BPSV,K,IMB,IPT,N,LR)
      IMT=IMB+K-N+1
      CALL UTROW(TM,BM,U,BU,K,IMT,ZD,N,LR)

C  PIVOT IF REQUIRED
      TX=TMAX(TM,IMT,IMB,N,LR)
      IF (DABS(TM(IMT+ZD,K+1))/TX.LT.DELTA) GO TO 100
      GO TO 10


C  PIVOTING PROCEDURE, STRATEGY 3
```

```
C  FIND ACCEPTABLE PIVOTS ABOVE AND BELOW Z.B.
  100 CALL FINDPV(TM(IMT,K+1),LR*(N-K),LR,N-K,TX,DELTA,K1,PP1,IERR)
      P=PP1-1
      CALL FINDPV(TM(IMT+1,1),N-K-1,1,N-K-1,TX,DELTA,K1,QP1,IERR)
      Q=QP1-1

C  MAKE TM(-K-1-Q) AVAILABLE FOR PIVOTING BY USING BPC : GET
C  ITERATES +,-(K-1), DO (R,C-CYCLE) Q TIMES, THEN ANOTHER
C  C-CYCLE.
      K=K-1
      CALL RECOV(TMSV,TPSV,BMSV,BPSV,TM,TP,BM,BP,K,IMB,IPT,N,LR)
      CALL MOVMTX(MPSV,MP,N,1,LR)
      CALL MOVMTX(MMSV,MM,N,1,LR)
         DO 140 I=1,Q
         CALL RCYCM(T,B,TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
  140    CALL CCYCM(TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
      K=K+1
      CALL CCYCM(TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
      CALL UTROW(TM,BM,U,BU,K,IMB-N+K+1,ZD,N,LR)

C  NEXT DO P+Q BNA-CYCLES (B-CYCLES)
      NBCYC=P+Q
         DO 150 I=1,NBCYC
         K=K+1
         CALL BCYCM(TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
  150    CALL UTROW(TM,BM,U,BU,K,IMB-N+K+1,ZD,N,LR)

C  FINALLY, RESTORE TO BAREISS FORM USING BPC : DO AN A-CYCLE,
C  (AN A,R-CYCLE) Q-1 TIMES, THEN ANOTHER A-CYCLE
      K=K+1
      CALL ACYCM(TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
      CALL UTROW(TM,BM,U,BU,K,IMB-N+K+1,ZD,N,LR)

      K=K+1
         QM1=Q-1
         DO 160 I=1,QM1
         CALL ACYCM(TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
  160    CALL RCYCM(T,B,TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
      CALL ACYCM(TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
      CALL UTROW(TM,BM,U,BU,K,IMB-N+K+1,ZD,N,LR)
      GO TO 10


  500 CALL BCKSUB(U,BU,X,N,LR)
      RETURN
      END



      SUBROUTINE BCYCM(TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
C
C  S/R TO DO A BAREISS CYCLE (B-CYCLE) & ACCUM MPYR MCS.

C  TM,BM - NEG.INDEX ITERATE AND RHS
C  IMB   - INDEX OF LAST ROW IN TOEP.PART OF TM
C  TP,TP - POS.INDEX ITERATE AND RHS
C  IPT   - INDEX OF 1ST ROW IN TOEP.PART OF TP
```

```
C  MM,MP - (REAL*8) 1ST ROW OF TOEP.PT. OF POS. AND NEG.
C          INDEX MPYR.MATRICES
C  K      - CYCLE TO BE COMPUTED
C  ZD     - (INTEGER) I/P ZERO-BAND DISPLACEMENT
C  N,LR   - ORDER, ROW DIM OF MATRICES
C
C  PROCEDURE:
C  1. ELIM BELOW ZERO-BAND OF NEG.INDEX ITERATE
C  2. ELIM ABOVE ZERO-BAND OF POS.INDEX ITERATE

      INTEGER ZD
      REAL*8 M
      REAL*8 TM(LR,N),BM(N),TP(LR,N),BP(N),MM(N),MP(N)

      IMT=K+IMB-N
      IPB=N-K+IPT
      CALL BAROP(TP,BP,IPT,IPB-1,IPT+ZD,1,TM,BM,IMT+1,M,N,LR)
      CALL MCALC(MP,0,MM,1,M,K+1)
      CALL BAROP(TM,BM,IMT+1,IMB,IMT+1,K+1-ZD,TP,BP,IPT,M,N,LR)
      CALL MCALC(MM,0,MP,0,M,K+1)
      RETURN
      END



      SUBROUTINE CCYCM(TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)

C  S/R TO MOVE ZERO-BAND DOWN (C-CYCLE) & ACCUM.MPYR.MCS.
C
C  PARAMETERS - AS FOR S/R 'BCYCM'
C
C  PROCEDURE:
C  1. ELIM BELOW ZERO-BAND OF POS.INDEX ITERATE
C  2. ELIM BELOW ZERO-BAND OF NEG.INDEX ITERATE

      INTEGER ZD
      REAL*8 M
      REAL*8 TM(LR,N),BM(N),TP(LR,N),BP(N),MM(N),MP(N)

      IMT=K+IMB-N
      IPB=N-K+IPT
      CALL BAROP(TM,BM,IMT+1,IMB,IMT+ZD+1,1,TP,BP,IPT,M,N,LR)
      CALL MCALC(MM,1,MP,0,M,K+1)
      CALL BAROP(TP,BP,IPT,IPB-1,IPT+ZD+1,1,TM,BM,IMT,M,N,LR)
      CALL MCALC(MP,0,MM,0,M,K+1)

      ZD=ZD+1
      IMB=IMB-1
      RETURN
      END



      SUBROUTINE ACYCM(TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)
C
C  S/R TO MOVE ZERO-BAND UP (A-CYCLE) & ACCUM.MPYR.MCS.
C
```

```
C  PARAMETERS - AS FOR S/R 'BCYCM'
C
C  PROCEDURE:
C  1. ELIM ABOVE ZERO-BAND OF NEG.INDEX ITERATE
C  2. ELIM ABOVE ZERO-BAND OF POS.INDEX ITERATE

      INTEGER ZD
      REAL*8 M
      REAL*8 TM(LR,N),BM(N),TP(LR,N),BP(N),MM(N),MP(N)

      IMT=K+IMB-N
      IPB=N-K+IPT

      CALL BAROP(TP,BP,IPT,IPB-1,IPT,K+1-ZD,TM,BM,IMT+1,M,N,LR)
      CALL MCALC(MP,0,MM,1,M,K+1)
      CALL BAROP(TM,BM,IMT+1,IMB,IMT+1,K+2-ZD,TP,BP,IPT+1,M,N,LR)
      CALL MCALC(MM,0,MP,1,M,K+1)

      ZD=ZD-1
      IPT=IPT+1
      RETURN
      END



       SUBROUTINE RCYCM(T,B,TM,BM,IMB,TP,BP,IPT,MM,MP,K,ZD,N,LR)

C  S/R TO PERFORM A REVERSE BAREISS-CYCLE
C
C  PARAMETERS - T,B : I/P TOEP.MX. AND RHS. OTHERS: AS FOR S/R
C  'BCYCM' EXCEPT THAT K IS THE I/P CYCLE (K-1) IS THE CYCLE
C  TO BE COMPUTED.
C
C  PROCEDURE:
C  1. GET PREVIOUS POS.INDEX MPYR & MPYR.MX. (1ST ROW OF
C                                              TOEP.PT.)
C  2. EXECUTE STEP K IN REVERSE TO GET STEP (K-1).
C  3. GET PREVIOUS NEG.INDEX MPYR & MPYR.MX. (1ST ROW OF
C                                              TOEP.PT.)
C  4. EXECUTE STEP (-K) IN REVERSE TO GET STEP (1-K).

      INTEGER ZD
      REAL*8 M
      REAL*8 TM(LR,N),BM(N),TP(LR,N),BP(N),MM(N),MP(N)
      REAL*8 T(LR,N),B(N)

      IMT=IMB+K+1-N
      IPB=IPT+N-K-1

      CALL MPREV(MM,MP,M,K+1)
      CALL BARREV(TM,BM,IMT,IMB,TP,BP,IPT,M,N,LR)

C  RESTORE ROW IPB+1 OF ITERATE (K-1) USING TOEPLICITY AND
C  THE MULTIPLIER IDENTITY
      DO 10 J=2,N
   10 TP(IPB+1,J)=TP(IPB,J-1)
      TP(IPB+1,1)=0.
```

```
      BP(IPB+1)=0.
      DO 20 I=1,K
      TP(IPB+1,1)=TP(IPB+1,1) + MP(I)*T(N-K+I,1)
   20 BP(IPB+1)=BP(IPB+1) + MP(I)*B(N-K+I)

      CALL MMREV(MM,MP,M,K+1)
      CALL BARREV(TP,BP,IPT,IPB,TM,BM,IMT,M,N,LR)

C  RESTORE ROW IMT-1 OF ITERATE (1-K) USING TOEPLICITY AND
C  THE MULTIPLIER IDENTITY
      DO 30 J=2,N
   30 TM(IMT-1,J-1)=TM(IMT,J)
      TM(IMT-1,N)=0.
      BM(IMT-1)=0.
      DO 40 I=1,K
      TM(IMT-1,N)=TM(IMT-1,N) + MM(I)*T(I,N)
   40 BM(IMT-1)=BM(IMT-1) + MM(I)*B(I)

      RETURN
      END



      SUBROUTINE BARREV(TS,BS,IS1,IS2,TD,BD,ID1,M,N,LR)
C
C  PERFORMS A BAREISS-OPERATION IN REVERSE (REAL*8)

C  TS,BS .... SOURCE MATRIX AND RHS
C  IS1,IS2 .. 1ST AND LAST ROWS OF SOURCE-BLOCK
C  TD,BD .... DESTINATION BLOCK AND RHS
C  ID1 ...... 1ST ROW OF DESTINATION BLOCK
C  M ........ MULTIPLIER (REAL*8)
C  N ........ ORDER OF MATRICES
C  LR ....... ROW-DIMENSION OF MATRICES

      REAL*8 M,TS(LR,N),BS(N),TD(LR,N),BD(N)

      ID=ID1
      DO 20 IS=IS1,IS2
      DO 10 J=1,N
   10 TD(ID,J)=TD(ID,J) + M*TS(IS,J)
      BD(ID)=BD(ID) + M*BS(IS)
   20 ID=ID+1
C     CALL DUMP(TS,TD,N,LR)
      RETURN
      END



      SUBROUTINE MCALC(MS,SS,MD,SD,M,L)
C
C  UPDATES MULTIPLIER VECTOR(REAL*8)
C
C  MS IS THE SOURCE VECTOR, SS IS THE NO. OF R/SHIFTS
C  FOR MS, MD IS THE DESTINATION VECTOR, SD IS THE NO.
C  OF PRELIMINARY R/SHIFTS FOR MD, M IS THE MULTIPLIER,
C  AND L IS THE LENGTH OF THE VECTORS.
```

```
C  (SEE BELOW FOR DATA TYPES)

      INTEGER SS,SD
      REAL*8 M,MS(L),MD(L),WK(20)

      CALL RSHIFT(MS,WK,SS,L)
      CALL RSHIFT(MD,MD,SD,L)
      DO 10 I=1,L
   10 MD(I)=MD(I) - M*WK(I)
      RETURN
      END


      SUBROUTINE RSHIFT(A,B,NS,L)
C
C  RIGHT-SHIFTS ROW-VECTOR A BY NS PLACES, PUTTING THE RESULT
C  IN B. L IS THE LENGTH OF THE VECTORS
C
C  A AND B ARE DOUBLE-PRECISION

      REAL*8 A(L),B(L)

      I=L
   10 IF(I.EQ.NS) GO TO 20
      B(I)=A(I-NS)
      I=I-1
      GO TO 10

   20 IF(NS.EQ.0) RETURN
      DO 30 I=1,NS
   30 B(I)=0.
      RETURN
      END



      SUBROUTINE MPREV(MM,MP,M,KP1)
C
C  S/R TO FIND MPYR & MPYR MX. (TOP OF TOEP.PT.) FOR STEP
C  K-1.
C
C  MM,MP - MPYR.MCS.(1ST ROW OF TOEP.PTS) FOR ITER.(-K),K.
C          MPYR.MX.FOR STEP (K-1) IS OVERWRITTEN ON MP.
C  M     - MPYR(O/P) FOR STEP K-1
C  KP1   - K+1

      REAL*8 MM(KP1),MP(KP1),M

      K=KP1-1
      M= -MP(KP1)/MM(KP1)
      DO 10 I=1,K
   10 MP(I)=MP(I) + M*MM(I)
      MP(KP1)=0.
      RETURN
      END
```

```
      SUBROUTINE MMREV(MM,MP,M,KP1)
C
C  S/R TO FIND MPYR & MPYR MX. (TOP OF TOEP.PT) FOR STEP
C  1-K.
C
C  MM,MP - MPYR.MCS.(1ST ROW OF TOEP.PTS) FOR ITER.(-K),K-1.
C          MPYR.MX.FOR STEP (1-K) IS OVERWRITTEN ON MM.
C  M     - MPYR(O/P) FOR STEP (1-K)
C  KP1   - K+1

      REAL*8 MM(KP1),MP(KP1),M

      K=KP1-1
      M= -MM(1)/MP(1)
      DO 10 I=1,K
   10 MM(I)=MM(I+1) + M*MP(I+1)
      MM(KP1)=0.
      RETURN
      END


C  SUBROUTINES UTROW, SAVE, RECOV, ADJSMX, BAROP, TMAX, MOVMTX AND
C  BCKSUB ARE THE SAME AS IN PBA-2.
C
C
C
```

```
SIZE OF ILL-COND BLOCK, NO.ALONG COL,ROW, DIST FROM SINGULARITY
?
4,3,2,2d-3


INPUT TOEPLITZ MATRIX T
    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000   -2.000   -7.000    1.000   10.000   -1.500
    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000   -2.000   -7.000    1.000   10.000
   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000   -2.000   -7.000    1.000
   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000   -2.000   -7.000
  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000   -2.000
   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000   -5.000
   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000    3.000
   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699    7.000
    2.000   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000    5.699
    1.000    2.000   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000    2.000
   -6.000    1.000    2.000   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000    6.000
    1.000   -6.000    1.000    2.000   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000   -1.000
   -0.500    1.000   -6.000    1.000    2.000   -1.000   -7.000   28.373  -19.661   12.757   -3.000    1.000    5.000


EXACT SOL OF T*X = B IS (1,1,...,1)


BAREISS ALG (BNA):

ERROR NORM =   0.2661D-01
ERROR VECTOR
  0.9628D-02  0.6009D-02 -0.6716D-02 -0.7315D-04  0.2661D-01  0.1394D-01 -0.2368D-01 -0.2004D-01  0.4591D-02  0.5767D-02
  0.1351D-02 -0.2150D-01  0.1127D-01


PIVOTED BAREISS ALG (PBA3):

ERROR NORM =   0.1377D-12
ERROR VECTOR
  0.1377D-12  0.1443D-13  0.1310D-13 -0.3582D-13 -0.1216D-13 -0.2537D-13 -0.2380D-13  0.4774D-13  0.7305D-13 -0.1943D-15
 -0.4677D-14  0.9770D-14  0.5218D-13
READY
```

```
SIZE OF ILL-COND BLOCK, NO.ALONG COL,ROW, DIST FROM SINGULARITY
?
4,3,3,2d-6


INPUT TOEPLITZ MATRIX T
    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000   -2.000   -7.000    1.000   10.000   -1.500
    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000   -2.000   -7.000    1.000   10.000
   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000   -2.000   -7.000    1.000
   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000   -2.000   -7.000
  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000   -2.000
   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000   -5.000
   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850    3.000
   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697    5.850
    2.000   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000    5.697
    1.000    2.000   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000    2.000
   -6.000    1.000    2.000   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000    6.000
    1.000   -6.000    1.000    2.000   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000   -1.000
   -0.500    1.000   -6.000    1.000    2.000   -1.000   -7.000   28.361  -19.656   12.755   -3.000    1.000    5.000


EXACT SOL OF T*X = B IS (1,1,...,1)


BAREISS ALG (BNA):

ERROR NORM =   0.1898D+00
ERROR VECTOR
  0.3581D-01  0.9108D-02  0.4447D-02  0.1255D+00  0.4220D-01 -0.1530D+00  0.8745D-01  0.9942D-02 -0.1898D+00 -0.4101D-02
  0.3750D-01 -0.6338D-01 -0.1216D+00


PIVOTED BAREISS ALG (PBA3):

ERROR NORM =   0.1195D-12
ERROR VECTOR
  0.1195D-12 -0.1708D-13  0.5285D-13 -0.3805D-13 -0.4725D-13 -0.2838D-13  0.7105D-14  0.6395D-13  0.2753D-13  0.1599D-13
 -0.1231D-13  0.1021D-13  0.3331D-13
READY
```

```
C  TEST PROGRAM FOR PIVOTED TOEPLITZ FACTORIZATION
C  (PBAFAC)

C  1. MAKES UP N X N TOEP.MX. 'A' WITH A(N,1),...,A(1,1),...,
C     ...,A(1,N) EQUAL TO T0(1),...,T0(2*N-1) RESPECTIVELY.
C  2. MAKES SPECIFIES SUB-MATRICES OF ORDER NSIZE ILL-COND.
C     BY ADJUST RELEVANT TOEP.DIAGS.
C  3. CALCS. FACTORS L,U USING ADAPTED BAREISS ALG (ABA) AND
C     EVALUATES L*U.
C  4. CALCS. L,U USING THE PIVOTING SEQ.RESULTING FROM PBA-2,
C     AND EVALUATES L*U.

      INTEGER ABASEQ(13),PBASEQ(13)
      REAL*8 T(25),T0(25),A(20,20),L(20,20),U(20,20),LU(20,20)
      REAL*8 EPS
      DATA T0/-1.5,10.,1.,-7.,-2.,-5.,3.,7.,9.,2.,6.,-1.,5.,
     1  1.,-3.,0.,0.,-4.,-7.,-1.,2.,1.,-6.,1.,-0.5/,
     2  ABASEQ/2,2,2,2,2,2,2,2,2,2,2,2,0/,
     3  PBASEQ/3,3,3,2,2,2,2,1,1,1,2,2,0/

C  MATRIX DIMENSIONS
      ID1=20
      N=13
      N2M1=2*N-1
      NM1=N-1

C  GEN. 'A' FROM ITS 1ST ROW AND COL, GET SPECS. FOR ILL-COND
C  SUBMATRICES, AND ADJUST TOEP.DIAGS. ACCORDINGLY
    1 DO 2 I=1,N2M1
    2 T(I)=T0(I)
      CALL TOEPMX(T,A,N,N2M1,ID1)
      WRITE(6,3)
    3 FORMAT(/' SIZE OF ILL-COND BLOCK, NO.ALONG COL,ROW, DIST ',
     1        'FROM SINGULARITY')
      READ(5,*) NSIZE,NCOL,NROW,EPS
      CALL ADJSMX(T,A,N,N2M1,ID1,NSIZE,NCOL,NROW,EPS)

    9 WRITE(6,10)
   10 FORMAT(/' INPUT TOEPLITZ MATRIX T')
      CALL PRNTMX(A,N,N,ID1)

C  FACTOR T USING ABA AND PRINT OUT L,U AND L*U
      CALL PBAFAC(A,ABASEQ,L,U,N,ID1)
      WRITE (6,30)
   30 FORMAT(/' L-FACTOR USING ABA')
      CALL PRNTMX(L,N,N,ID1)
      WRITE (6,40)
   40 FORMAT(/' U-FACTOR USING ABA')
      CALL PRNTMX(U,N,N,ID1)
      CALL MULTMX(L,U,LU,N,N,N,ID1,ID1,ID1)
      WRITE (6,45)
   45 FORMAT(/' L*U (L,U FROM ABA). SHOULD BE T')
      CALL PRNTMX(LU,N,N,ID1)

C  FACTOR T USING THE PIVOTING SEQ.FROM PBA-2 AND PRINT OUT
C  L,U AND L*U
      CALL PBAFAC(A,PBASEQ,L,U,N,ID1)
```

```
      WRITE(6,50)
   50 FORMAT(//////' L-FACTOR USING PBA-2')
      CALL PRNTMX(L,N,N,ID1)
      WRITE(6,60)
   60 FORMAT(//' U-FACTOR USING PBA-2')
      CALL PRNTMX(U,N,N,ID1)
      CALL MULTMX(L,U,LU,N,N,N,ID1,ID1,ID1)
      WRITE (6,70)
   70 FORMAT(//' L*U (L,U FROM PBA-2). SHOULD BE T')
      CALL PRNTMX(LU,N,N,ID1)

      STOP
      END



      SUBROUTINE PBAFAC(T0,PSEQ,L0,U0,N,LR0)
C
C  PIVOTED BAREISS-FACTORIZATION. IMPLEMENTS ALG.6.4.1 FOR
C  A GIVEN SEQ. OF A,B OR C-CYCLES.
C
C  T0  - I/P TOEP.MX.
C  PSEQ- (INTEGER) PIVOTING SEQ. (1:A-CYCLE,2:B-CYCLE
C                                 3:C-CYCLE)
C  L0  - O/P L-FACTOR
C  U0  - O/P U-FACTOR
C  N,LR0-ORDER,ROW DIMS.OF MATRICES

C  TEMPORARIES HAVE SAME NOTATION AS IN ALG.6.4.1
C
      INTEGER ZD,S(20),V(20),G,H,PSEQ(N)
      REAL*8 T0(LR0,N),T(20,20),TM(20,20),TP(20,20),
     1  L0(LR0,N),U0(LR0,N),L(20,20),U(20,20),SM,SP
      REAL*8 TX,DELTA

      LR=20
      NM1=N-1

      DO 2 I=1,N
      DO 2 J=1,N
    2 T(I,J)=T0(I,J)

C  INITIALIZATION
      CALL MOVMTX(T,TP,N,N,LR)
      CALL MOVMTX(T,TM,N,N,LR)


C  TOEPLITZ ROW-POINTERS AND MPYR MX SCALE FACTORS
      IPT=1
      IMB=N
      SP=1.
      SM=1.

C  INITIALIZE S(1),V(1),G,H FOR THE GIVEN PIVOTING SEQUENCE
C  PSEQ (SEE ALG.6.4.1)

      NA=0
```

```
      NC=0
      DO 5 I=1,NM1
      IF(PSEQ(I).EQ.1) NA=NA+1
      IF(PSEQ(I).EQ.3) NC=NC+1
    5 CONTINUE
      S(1)=NA+1
      V(1)=NC+1
      G=NA
      H=NC

C  GET 1ST COL AND ROW OF L AND U RESPECTIVELY
      CALL MOVMTX(TM(1+NA,1),U,1,N,LR)
        DO 8 I=1,N
    8   L(I,1)=TP(N-NC,N+1-I)/TP(N-NC,N+1-S(1))

      ZD=0

C  MAIN LOOP
      DO 40 K=1,NM1
      ICYC=PSEQ(K)
      GO TO (10,20,30),ICYC
   10 CALL BFA(TM,IMB,TP,IPT,L,U,K,ZD,SM,SP,S,V,G,H,N,LR)
      GO TO 40
   20 CALL BFB(TM,IMB,TP,IPT,L,U,K,ZD,SM,SP,S,V,G,H,N,LR)
      GO TO 40
   30 CALL BFC(TM,IMB,TP,IPT,L,U,K,ZD,SM,SP,S,V,G,H,N,LR)
   40 CONTINUE

C  COPY L,U INTO L0,U0 TO PASS TO CALLING PROG.
      DO 50 I=1,N
      DO 50 J=1,N
      L0(I,J)=L(I,J)
   50 U0(I,J)=U(I,J)

      RETURN
      END



      SUBROUTINE BFB(TM,IMB,TP,IPT,L,U,K,ZD,SM,SP,S,V,
     1  G,H,N,LR)
C
C  S/R TO DO A B-CYCLE AND GET COL OF L AND ROW OF U
C  ACCORDING TO ALG.6.4.1
C
C  TM  - NEG.INDEX BAREISS ITERATE
C  IMB - BOTTOM OF TOEP.PT.OF TM
C  TP  - POS.INDEX BAREISS ITERATE
C  IPT - TOP OF TOEP.PT.OF TP
C  L,U - (O/P,REAL*8) L,U FACTORS
C  K,ZD- BAREISS CYCLE, ZERO-BAND DISPLACEMENT
C  SM,SP-SCALE FACTORS FOR NEG.,POS.INDEX MPYR MCS.
C  S,V - PERMUTATION VECTORS (SEE ALG.6.4.1)
C  G,H - NO.OF NON-ZEROS TO RIGHT,LEFT OF ZERO-BAND OF DESIRED
C        ROW FOR USE IN FACTORS (SEE ALG.6.4.1)
C  N,LR- ORDER,ROW DIM. OF MATRICES
```

```
C  TEMPORARIES : LAMBDA,RHO - AS IN ALG.6.4.1 ; BZB,EZB -
C  BZB,EZB - BEGINNING, END OF ZERO-BAND; IMT,IPB - TOP,
C  BOTTOM OF TM,TP RESPEC.; SU - SCALE FACTOR FOR ROWS OF U.

      INTEGER ZD,S(N),V(N),G,H,RHO,BZB,EZB
      REAL*8 MM,MP,SM,SP,SU
      REAL*8 TM(LR,N),BM(20),TP(LR,N),BP(20),L(LR,N),U(LR,N)

C  DO B-CYCLE, UPDATE SCALE FACTORS AND PERM.VECTORS
      IMT=K+IMB-N
      IPB=N-K+IPT
      CALL BAROP(TP,BP,IPT,IPB-1,IPT+ZD,1,TM,BM,IMT+1,MM,N,LR)
      CALL BAROP(TM,BM,IMT+1,IMB,IMT+1,K+1-ZD,TP,BP,IPT,MP,N,LR)
      SU=SM
      SP=TP(IPT+ZD,1)*SM/TM(IMT+1,K+1-ZD)
      S(K+1)=IMAX(S,K)+1
      V(K+1)=IMAX(V,K)+1

C  GET COL,ROW OF L,U ACCORDING TO ALG.6.4.1
      LAMBDA=IMT+1+ZD+H
      DO 10 J=1,N
   10 U(K+1,J)=TM(LAMBDA,J)/SU
      RHO=IPB-1+ZD-G
      DO 20 I=1,N
   20 L(I,K+1)=TP(RHO,N+1-I)/TP(RHO,N+1-S(K+1))

C  CLEAR ELEMENTS THAT SHOULD BE ZERO IN L AND U
      BZB=H+1
      EZB=H+K
      DO 30 J=BZB,EZB
   30 U(K+1,J)=0.
      BZB=G+1
      EZB=G+K
      DO 40 I=BZB,EZB
   40 L(I,K+1)=0.
      RETURN
      END



      SUBROUTINE BFC(TM,IMB,TP,IPT,L,U,K,ZD,SM,SP,S,V,G,H,
     1  N,LR)

C  S/R TO DO A C-CYCLE AND GET COL OF L AND ROW OF U
C  ACCORDING TO ALG.6.4.1
C
C  PARAMETERS AND TEMPORARIES - AS FOR BFB

      INTEGER ZD,S(N),V(N),G,H,RHO,BZB,EZB
      REAL*8 MM,MP,SM,SP,SU
      REAL*8 TM(LR,N),BM(20),TP(LR,N),BP(20),L(LR,N),U(LR,N)

C  DO C-CYCLE, UPDATE SCALE FACTORS AND PERM.VECTORS
      IMT=K+IMB-N
      IPB=N-K+IPT
      CALL BAROP(TM,BM,IMT+1,IMB,IMT+ZD+1,1,TP,BP,IPT,MP,N,LR)
      SU=-MP*SM
```

```
      CALL BAROP(TP,BP,IPT,IPB-1,IPT+ZD+1,1,TM,BM,IMT,MM,N,LR)
      SM=TM(IMT,K-ZD)*SP/TP(IPT+ZD+1,1)
      ZD=ZD+1
      IMB=IMB-1
      S(K+1)=IMAX(S,K)+1
      V(K+1)=IMIN(V,K)-1

C  GET COL,ROW OF L,U ACCORDING TO ALG.6.4.1
      LAMBDA=IPT+ZD-1+H
      DO 10 J=1,N
   10 U(K+1,J)=TP(LAMBDA,J)/SU
      RHO=IPB-1+ZD-G
      DO 20 I=1,N
   20 L(I,K+1)=TP(RHO,N+1-I)/TP(RHO,N+1-S(K+1))

C  CLEAR ELEMENTS THAT SHOULD BE ZERO IN L AND U
      BZB=H+1
      EZB=H+K
      DO 30 J=BZB,EZB
   30 U(K+1,J)=0.
      BZB=G+1
      EZB=G+K
      DO 40 I=BZB,EZB
   40 L(I,K+1)=0.

C  UPDATE H
      H=H-1
      RETURN
      END



      SUBROUTINE BFA(TM,IMB,TP,IPT,L,U,K,ZD,SM,SP,S,V,G,H,
     1  N,LR)
C
C  S/R TO DO AN A-CYCLE AND GET COL OF L AND ROW OF U
C  ACCORDING TO ALG.6.4.1.
C
C  PARAMETERS AND TEMPORARIES - AS IN BFB.

      INTEGER ZD,S(N),V(N),G,H,RHO,BZB,EZB
      REAL*8 MM,MP,SM,SP,SU
      REAL*8 TM(LR,N),BM(20),TP(LR,N),BP(20),L(LR,N),U(LR,N)

C  DO A-CYCLE, UPDATE SCALE FACTORS AND PERM.VECTORS
      IMT=K+IMB-N
      IPB=N-K+IPT
      CALL BAROP(TP,BP,IPT,IPB-1,IPT,K+1-ZD,TM,BM,IMT+1,MM,N,LR)
      SU=-MM*SP
      CALL BAROP(TM,BM,IMT+1,IMB,IMT+1,K+2-ZD,TP,BP,IPT+1,MP,N,LR)
      SP=TP(IPT+ZD,1)*SM/TM(IMT+1,K+2-ZD)
      ZD=ZD-1
      IPT=IPT+1
      S(K+1)=IMIN(S,K)-1
      V(K+1)=IMAX(V,K)+1

C  GET COL,ROW OF L,U ACCORDING TO ALG.6.4.1
```

```
      LAMBDA=IMT+1+ZD+H
      DO 10 J=1,N
   10 U(K+1,J)=TM(LAMBDA,J)/SU
      RHO=IMB+ZD-G+1
      DO 20 I=1,N
   20 L(I,K+1)=TM(RHO,N+1-I)/TM(RHO,N+1-S(K+1))

C  CLEAR ELEMENTS THAT SHOULD BE ZERO IN L AND U
      BZB=H+1
      EZB=H+K
      DO 30 J=BZB,EZB
   30 U(K+1,J)=0.
      BZB=G+1
      EZB=G+K
      DO 40 I=BZB,EZB
   40 L(I,K+1)=0.

C  UPDATE G
      G=G-1
      RETURN
      END


      FUNCTION IMAX(KK,N)

C  FUNCTION TO FIND MAX INTEGER IN KK(N)
      DIMENSION KK(N)

      IMAX=KK(1)
      DO 10 I=1,N
      IF(KK(I).GT.IMAX) IMAX=KK(I)
   10 CONTINUE
      RETURN
      END



      FUNCTION IMIN(KK,N)

C  FUNCTION OF FIND MIN INTEGER INKK(N)
      DIMENSION KK(N)

      IMIN=KK(1)
      DO 10 I=1,N
      IF(KK(I).LT.IMIN) IMIN=KK(I)
   10 CONTINUE
      RETURN
      END




C  SUBROUTINES ADJSMX, BAROP AND MOVMTX ARE THE SAME AS IN
C  PBA2.
```

```
SIZE OF ILL-COND BLOCK, NO.ALONG COL,ROW, DIST FROM SINGULARITY
?
4,3,2,5d-3

INPUT TOEPLITZ MATRIX T
     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000    -2.000    -7.000     1.000    10.000    -1.500
     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000    -2.000    -7.000     1.000    10.000
    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000    -2.000    -7.000     1.000
    12.760    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000    -2.000    -7.000
   -19.668    12.760    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000    -2.000
    28.391   -19.668    12.760    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000
    -7.000    28.391   -19.668    12.760    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000
    -1.000    -7.000    28.391   -19.668    12.760    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702     7.000
     2.000    -1.000    -7.000    28.391   -19.668    12.760    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702
     1.000     2.000    -1.000    -7.000    28.391   -19.668    12.760    -3.000     1.000     6.000    -1.000     6.000     2.000
    -6.000     1.000     2.000    -1.000    -7.000    28.391   -19.668    12.760    -3.000     1.000     6.000    -1.000     6.000
     1.000    -6.000     1.000     2.000    -1.000    -7.000    28.391   -19.668    12.760    -3.000     1.000     6.000    -1.000
    -0.500     1.000    -6.000     1.000     2.000    -1.000    -7.000    28.391   -19.668    12.760    -3.000     1.000     6.000

L-FACTOR USING ABA
        1.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.
        0.        1.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.
       -1.        0.        1.        0.        0.        0.        0.        0.        0.        0.        0.        0.
        3.       -0.       -2.        1.        0.        0.        0.        0.        0.        0.        0.        0.
       -4.        2.        3.       -1.        1.        0.        0.        0.        0.        0.        0.        0.
        6.       -3.       -3.        2.       -1.        1.        0.        0.        0.        0.        0.        0.
       -1.        5.        0.     4189.     7172.-1292786.        1.        0.        0.        0.        0.        0.
       -0.       -1.        3.     3355.     2762.   265213.       -0.        1.        0.        0.        0.        0.
        0.       -0.       -1.    -8680.   -17248. 3717772.       -3.        1.        1.        0.        0.        0.
        0.        0.       -0.     3041.    11381.-3636350.        3.       -2.        1.        1.        0.        0.
       -1.       -0.        1.     -576.    -3148. 1128518.       -1.        1.       -2.        9.        1.        0.
        0.       -1.       -0.    -2412.    -3721.  563577.       -0.       -0.        0.      -15.       -2.        1.
       -0.        0.       -1.      -31.     1664. -739137.        1.       -1.       -0.       10.        1.        0.

U-FACTOR USING ABA
 0.60D+01-0.10D+01 0.60D+01 0.20D+01 0.57D+01 0.70D+01 0.30D+01-0.50D+01-0.20D+01-0.70D+01 0.10D+01 0.10D+02-0.15D+01
 0.0       0.52D+01-0.22D+01 0.56D+01 0.86D+00 0.43D+01 0.64D+01 0.40D+01-0.46D+01-0.60D+00-0.72D+01-0.10D+01 0.10D+02
 0.0       0.0       0.88D+01-0.23D+00 0.94D+01 0.59D+01 0.70D+01 0.37D+01 0.22D+01-0.92D+01-0.85D+00-0.92D+00-0.69D+00
 0.0       0.0       0.0      -0.32D-02-0.77D-02-0.18D+01 0.65D+01 0.25D+02 0.15D+02 0.57D+01-0.96D+01-0.29D+02-0.34D+01
 0.0       0.0       0.0       0.0      0.45D-02 0.18D+01-0.83D+01-0.18D+02 0.96D+01 0.96D+01 0.15D+02 0.20D+02-0.26D+02
 0.0       0.0       0.0       0.0      0.0      0.41D-02 0.18D+01-0.83D+01-0.18D+02 0.96D+01 0.96D+01 0.15D+02 0.20D+02
 0.0       0.0       0.0       0.0      0.0      0.0      0.23D+07-0.11D+08-0.24D+08 0.12D+08 0.12D+08 0.20D+08 0.25D+08
 0.0       0.0       0.0       0.0      0.0      0.0      0.0       0.81D+04-0.26D+05-0.11D+06-0.65D+05-0.25D+05 0.39D+05
 0.0       0.0       0.0       0.0      0.0      0.0      0.0       0.0      -0.20D+04-0.53D+04-0.29D+04-0.63D+03 0.27D+04
 0.0       0.0       0.0       0.0      0.0      0.0      0.0       0.0       0.0      0.31D+02-0.22D+03-0.21D+02-0.16D+02
 0.0       0.0       0.0       0.0      0.0      0.0      0.0       0.0       0.0       0.0      0.24D+04 0.30D+03 0.18D+03
 0.0       0.0       0.0       0.0      0.0      0.0      0.0       0.0       0.0       0.0       0.0      0.19D+02 0.49D+02
 0.0       0.0       0.0       0.0      0.0      0.0      0.0       0.0       0.0       0.0       0.0       0.0     -0.76D+02

LXU (L,U FROM ABA). SHOULD BE T
     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000    -2.000    -7.000     1.000    10.000    -1.500
     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000    -2.000    -7.000     1.000    10.000
    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000    -2.000    -7.000     1.000
    12.760    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000    -2.000    -7.000
   -19.668    12.760    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000    -2.000
    28.391   -19.668    12.760    -3.000     1.000     6.000    -1.000     6.000     2.000     5.702     7.000     3.000    -5.000
    -7.000    28.391   -19.668    12.760    -3.000     1.000     5.000    -1.000     6.000     2.000     5.702     7.000     3.000
    -1.000    -7.000    28.391   -19.668    12.760    -3.000     1.000     4.999    -1.007     6.001     2.001     5.709     7.004
     2.000    -1.000    -7.000    28.391   -19.668    12.760    -3.000     1.000     5.001    -1.007     6.001     2.000     5.709
     1.000     2.000    -1.000    -7.000    28.391   -19.668    12.761    -3.004     1.009     5.003    -1.003     5.992     2.001
    -6.000     1.000     2.000    -1.000    -7.000    28.391   -19.668    12.761    -3.021     1.011     5.004    -0.986     5.999
     1.000    -6.000     1.000     2.000    -1.000    -7.000    28.391   -19.669    12.764    -3.020     1.012     5.000    -0.986
    -0.500     1.000    -6.000     1.000     2.000    -1.000    -7.000    28.391   -19.666    12.764    -3.020     1.009     4.999
```

```
L-FACTOR USING PBA-2
   0.400   -1.750    0.687   -0.000   -0.001   -0.085    0.310    1.175   -4.716    2.738    1.000    0.0      0.0
   1.200    0.688   -0.017   -0.000   -0.000   -0.000   -0.085    0.305    0.039    1.000    0.0      0.0      0.0
  -0.200   -1.625    0.644   -0.000   -0.000   -0.000   -0.000   -0.084    1.000    0.0      0.0      0.0      0.0
   1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
   0.200    1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
  -0.600   -4.175    1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
   2.552    6.944   -1.653    1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
  -3.934  -10.101    3.385    0.800    1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
   5.678    3.962   -1.075   -2.072    0.802    1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0
  -1.400   -0.125   -0.015    0.726   -2.074    0.799    1.000    0.0      0.0      0.0      0.0      0.0      0.0
  -0.200   -0.688    0.279   -0.137    0.726   -2.070    0.805    1.000    0.0      0.0      0.0      0.0      0.0
   0.400   -0.188   -0.067   -0.576   -0.137    0.725   -2.077    0.775   -9.745    5.226    2.043    1.000    0.0
   0.200    1.938   -0.708   -0.007   -0.576   -0.137    0.727   -2.040   15.324   -6.225   -3.141    0.490    1.000

U-FACTOR USING PBA-2
  12.760   -3.000    1.000    5.000   -1.000    6.000    2.000    5.702    7.000    3.000   -5.000   -2.000   -7.000
 -22.220   13.360   -3.200    0.0      5.200   -2.200    5.600    0.860    4.302    6.400    4.000   -4.600   -0.600
 -56.722   34.311    0.0      0.0     22.110   -0.585   23.580   13.010   24.163   34.223   20.700  -17.405  -11.705
  20.953    0.0      0.0      0.0     -0.003   -0.003   -0.007   -0.013   -1.794    6.480   24.909   15.271    5.680
   0.0      0.0      0.0      0.0    -13.494    0.362  -14.387   -7.920  -10.368  -38.591  -87.271   -1.928    8.485
   0.0      0.0      0.0      0.0      0.0    -13.518    0.338  -14.469  -21.210   37.787  146.365   46.055   40.204
   0.0      0.0      0.0      0.0      0.0      0.0    -13.477    0.437    8.456 -104.393 -281.696  -49.521  -26.702
   0.0      0.0      0.0      0.0      0.0      0.0      0.0    -13.691  -48.816  187.654  583.077  139.004  106.792
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    -8.282   -0.326   39.058    8.010   15.138
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0   -75.611 -206.992  -40.707  -16.858
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0   161.240   19.857   12.044
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    18.525   48.959
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0   -76.132

L&U (L,U FROM PBA-2). SHOULD BE I
   5.000   -1.000    6.000    2.000    5.702    7.000    3.000   -5.000   -2.000   -7.000    1.000   10.000   -1.500
   1.000    5.000   -1.000    6.000    2.000    5.702    7.000    3.000   -5.000   -2.000   -7.000    1.000   10.000
  -3.000    1.000    5.000   -1.000    6.000    2.000    5.702    7.000    3.000   -5.000   -2.000   -7.000    1.000
  12.760   -3.000    1.000    5.000   -1.000    6.000    2.000    5.702    7.000    3.000   -5.000   -2.000   -7.000
 -19.668   12.760   -3.000    1.000    5.000   -1.000    6.000    2.000    5.702    7.000    3.000   -5.000   -2.000
  28.391  -19.668   12.760   -3.000    1.000    5.000   -1.000    6.000    2.000    5.702    7.000    3.000   -5.000
  -7.000   28.391  -19.668   12.760   -3.000    1.000    5.000   -1.000    6.000    2.000    5.702    7.000    3.000
  -1.000   -7.000   28.391  -19.668   12.760   -3.000    1.000    5.000   -1.000    6.000    2.000    5.702    7.000
   2.000   -1.000   -7.000   28.391  -19.668   12.760   -3.000    1.000    5.000   -1.000    6.000    2.000    5.702
   1.000    2.000   -1.000   -7.000   28.391  -19.668   12.760   -3.000    1.000    5.000   -1.000    6.000    2.000
  -6.000    1.000    2.000   -1.000   -7.000   28.391  -19.668   12.760   -3.000    1.000    5.000   -1.000    6.000
   1.000   -6.000    1.000    2.000   -1.000   -7.000   28.391  -19.668   12.760   -3.000    1.000    5.000   -1.000
  -0.500    1.000   -6.000    1.000    2.000   -1.000   -7.000   28.391  -19.668   12.760   -3.000    1.000    5.000
READY
```

```
SIZE OF ILL-COND BLOCK, NO.ALONG COL,ROW, DIST FROM SINGULARITY
?
4,3,2,4d-4

INPUT TOEPLITZ MATRIX T
    5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000    -2.000    -7.000     1.000    10.000    -1.500
    1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000    -2.000    -7.000     1.000    10.000
   -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000    -2.000    -7.000     1.000
   12.756    -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000    -2.000    -7.000
  -19.657    12.756    -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000    -2.000
   28.363   -19.657    12.756    -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000
   -7.000    28.363   -19.657    12.756    -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000
   -1.000    -7.000    28.363   -19.657    12.756    -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000
    2.000    -1.000    -7.000    28.363   -19.657    12.756    -3.000     1.000     5.000    -1.000     6.000     2.000     5.698
    1.000     2.000    -1.000    -7.000    28.363   -19.657    12.756    -3.000     1.000     5.000    -1.000     6.000     2.000
   -6.000     1.000     2.000    -1.000    -7.000    28.363   -19.657    12.756    -3.000     1.000     5.000    -1.000     6.000
    1.000    -6.000     1.000     2.000    -1.000    -7.000    28.363   -19.657    12.756    -3.000     1.000     5.000    -1.000
   -0.500     1.000    -6.000     1.000     2.000    -1.000    -7.000    28.363   -19.657    12.756    -3.000     1.000     5.000

L-FACTOR USING ABA
 0.10D+01 0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0
 0.20D+00 0.10D+01  0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0
-0.60D+00 0.77D-01  0.10D+01  0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0
 0.26D+01-0.86D-01-0.17D+01  0.10D+01  0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0
-0.39D+01  0.17D+01  0.28D+01-0.65D+00  0.10D+01  0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0
 0.57D+01-0.27D+01-0.31D+01  0.21D+01-0.65D+00  0.10D+01  0.0       0.0       0.0       0.0       0.0       0.0       0.0
-0.14D+01  0.52D+01  0.17D-01  0.52D+05  0.89D+05-0.20D+09  0.10D+01  0.0       0.0       0.0       0.0       0.0       0.0
-0.20D+00-0.14D+01  0.30D+01  0.42D+05  0.35D+05  0.40D+08-0.20D+08  0.10D+01  0.0       0.0       0.0       0.0       0.0
 0.40D+00-0.12D+00-0.11D+01-0.11D+06-0.22D+06  0.59D+09-0.29D+01  0.80D+00  0.10D+01  0.0       0.0       0.0       0.0
 0.20D+00  0.42D+00-0.14D+00  0.38D+05  0.14D+06-0.57D+09  0.28D+01-0.21D+01  0.96D+00  0.10D+01  0.0       0.0       0.0
-0.12D+01-0.38D-01  0.10D+01-0.72D+04-0.39D+05  0.18D+09-0.87D+00  0.74D+00-0.19D+01-0.38D+01  0.10D+01  0.0       0.0
 0.20D+00-0.11D+01-0.30D+00-0.30D+05-0.46D+05  0.89D+08-0.44D+00-0.14D+00  0.40D+00  0.38D+01-0.16D+01  0.10D+01  0.0
-0.10D+00  0.17D+00-0.57D+00-0.38D+03  0.21D+05-0.12D+09  0.57D+00-0.58D+00-0.63D-01-0.16D+01  0.91D+00-0.59D+00  0.10D+01

U-FACTOR USING ABA
 0.50D+01-0.10D+01  0.60D+01  0.20D+01  0.57D+01  0.70D+01  0.30D+01-0.50D+01-0.20D+01-0.70D+01  0.10D+01  0.10D+02-0.15D+01
 0.0       0.52D+01-0.22D+01  0.56D+01  0.86D+00  0.43D+01  0.64D+01  0.40D+01-0.46D+01-0.60D+00-0.72D+01-0.10D+01  0.10D+02
 0.0       0.0       0.88D+01-0.23D+00  0.94D+01  0.59D+01  0.70D+01  0.37D+01  0.22D+01-0.92D+01-0.85D+00-0.92D+00-0.69D+00
 0.0       0.0       0.0      -0.26D-03-0.61D-03-0.18D+01  0.65D+01  0.25D+02  0.15D+02  0.57D+01-0.96D+01-0.29D+02-0.34D+01
 0.0       0.0       0.0       0.0       0.36D-03  0.18D+01-0.83D+01-0.18D+02  0.96D+01  0.96D+01  0.15D+02  0.20D+02-0.26D+02
 0.0       0.0       0.0       0.0       0.0       0.33D-03  0.18D+01-0.83D+01-0.18D+02  0.96D+01  0.96D+01  0.15D+02  0.20D+02
 0.0       0.0       0.0       0.0       0.0       0.0       0.36D+09-0.17D+10-0.38D+10  0.20D+10  0.20D+10  0.31D+10  0.40D+10
 0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.94D+05-0.34D+06-0.13D+07-0.80D+06-0.30D+06  0.50D+06
 0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      -0.24D+04-0.61D+04-0.34D+04-0.66D+03  0.32D+04
 0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      -0.90D+02-0.34D+03-0.68D+02  0.39D+02
 0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      -0.63D+03-0.53D+02  0.25D+03
 0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.22D+03  0.21D+03
 0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.77D+02

L*U (L,U FROM ABA). SHOULD BE T
    5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000    -2.000    -7.000     1.000    10.000    -1.500
    1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000    -2.000    -7.000     1.000    10.000
   -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000    -2.000    -7.000     1.000
   12.756    -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000    -2.000    -7.000
  -19.657    12.756    -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000    -2.000
   28.363   -19.657    12.756    -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000    -5.000
   -7.000    28.363   -19.657    12.756    -3.000     1.000     5.000    -1.000     6.000     2.000     5.698     7.000     3.000
   -1.000    -7.000    28.363   -19.657    12.756    -3.000    12.661    -5.104   -18.804    34.268    43.731    29.548    27.695
    2.000    -1.000    -7.000    28.363   -19.657    12.756    -4.067     8.882   -14.477   -17.676    32.690    49.669    39.140
    1.000     2.000    -1.000    -7.000    28.363   -19.657    -0.028   -71.102  -154.282    16.236   -26.186   141.396   217.704
   -6.000     1.000     2.000    -1.000    -7.000    28.363     9.574   -12.378   -85.376   -93.672   114.662    11.193   161.581
    1.000    -6.000     1.000     2.000    -1.000    -7.000    18.773    15.983    -2.174  -107.122  -127.085    98.330    -1.185
   -0.500     1.000    -6.000     1.000     2.000    -1.000    -9.306    10.183    -5.259     0.598  -110.330  -113.523   120.000
```

```
L-FACTOR USING PBA-2
   0.400   -1.750    0.688   -0.000   -0.000   -0.085    0.310    1.190   -4.776    2.729    1.000    0.0      0.0
   1.200    0.688   -0.017   -0.000   -0.000   -0.000   -0.085    0.310    0.010    1.000    0.0      0.0      0.0
  -0.200   -1.625    0.645   -0.000   -0.000   -0.000   -0.000   -0.085    1.000    0.0      0.0      0.0      0.0
   1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
   0.200    1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
  -0.600   -4.174    1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
   2.561    6.949   -1.653    1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
  -3.931  -10.092    3.333    0.802    1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0
   5.673    3.990   -1.076   -2.074    0.802    1.000    0.0      0.0      0.0      0.0      0.0      0.0      0.0
  -1.400   -0.125   -0.015    0.727   -2.074    0.802    1.000    0.0      0.0      0.0      0.0      0.0      0.0
  -0.200   -0.688    0.279   -0.138    0.727   -2.074    0.802    1.000    0.0      0.0      0.0      0.0      0.0
   0.400   -0.188   -0.067   -0.577   -0.138    0.727   -2.075    0.800   -9.843    5.199    2.040    1.000    0.0
   0.200    1.938   -0.708   -0.007   -0.577   -0.138    0.727   -2.072   15.420   -6.187   -3.137    0.487    1.000


U-FACTOR USING PBA-2
  12.756   -3.000    1.000    5.000   -1.000    6.000    2.000    5.698    7.000    3.000   -5.000   -2.000   -7.000
 -22.208   13.356   -3.200    0.0      5.200   -2.200    5.600    0.860    4.298    6.400    4.000   -4.600   -0.600
 -56.672   34.283    0.0      0.0     22.103   -0.582   23.572   13.010   24.137   34.209   20.694  -17.399  -11.704
  20.905    0.0      0.0      0.0     -0.000   -0.000   -0.001   -0.001   -1.784    6.479   24.902   15.266    5.674
   0.0      0.0      0.0      0.0    -13.477    0.355  -14.373   -7.931  -10.339  -38.549  -67.268   -1.962    8.475
   0.0      0.0      0.0      0.0      0.0    -13.479    0.353  -14.379  -21.179   37.778  146.367   46.087   40.168
   0.0      0.0      0.0      0.0      0.0      0.0    -13.476    0.361    8.539 -104.427 -282.146  -49.759  -26.801
   0.0      0.0      0.0      0.0      0.0      0.0      0.0    -13.493  -48.890  187.481  583.187  139.322  106.879
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    -8.345   -0.079   39.854    8.215   15.280
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    -76.378 -208.421  -41.193  -16.910
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    160.776   19.912   11.960
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0     18.563   48.981
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    -75.936


L*U (L,U FROM PBA-2). SHOULD BE I
   5.000   -1.000    6.000    2.000    5.698    7.000    3.000   -5.000   -2.000   -7.000    1.000   10.000   -1.500
   1.000    5.000   -1.000    6.000    2.000    5.698    7.000    3.000   -5.000   -2.000   -7.000    1.000   10.000
  -3.000    1.000    5.000   -1.000    6.000    2.000    5.698    7.000    3.000   -5.000   -2.000   -7.000    1.000
  12.756   -3.000    1.000    5.000   -1.000    6.000    2.000    5.698    7.000    3.000   -5.000   -2.000   -7.000
 -19.657   12.756   -3.000    1.000    5.000   -1.000    6.000    2.000    5.698    7.000    3.000   -5.000   -2.000
  28.363  -19.657   12.756   -3.000    1.000    5.000   -1.000    6.000    2.000    5.698    7.000    3.000   -5.000
  -7.000   28.363  -19.657   12.756   -3.000    1.000    5.000   -1.000    6.000    2.000    5.698    7.000    3.000
  -1.000   -7.000   28.363  -19.657   12.756   -3.000    1.000    5.000   -1.000    6.000    2.000    5.698    7.000
   2.000   -1.000   -7.000   28.363  -19.657   12.756   -3.000    1.000    5.000   -1.000    6.000    2.000    5.698
   1.000    2.000   -1.000   -7.000   28.363  -19.657   12.756   -3.000    1.000    5.000   -1.000    6.000    2.000
  -6.000    1.000    2.000   -1.000   -7.000   28.363  -19.657   12.756   -3.000    1.000    5.000   -1.000    6.000
   1.000   -6.000    1.000    2.000   -1.000   -7.000   28.363  -19.657   12.756   -3.000    1.000    5.000   -1.000
  -0.500    1.000   -6.000    1.000    2.000   -1.000   -7.000   28.363  -19.657   12.756   -3.000    1.000    5.000
READY
```

```
C  TEST PROGRAM FOR PTZA1 (PIVOTED BAREISS ALG, STRATEGY 1:
C                            SIMPLE 2-CHOICE STRATEGY)

C  1. MAKES UP N X N TOEP.MX. 'A' WITH A(N,1),...,A(1,1),...,
C     ...,A(1,N) EQUAL TO T0(1),...,T0(2*N-1) RESPECTIVELY.
C  2. MAKES SPECIFIED SUB-MATRIX OF ORDER NSIZE ILL-COND.
C     BY ADJUST RELEVANT TOEP.DIAG.
C  3. INVERTS 'A' USING TRENCH-ZOHAR ALG (TZA) AND PIVOTED TZA,
C     STRATEGY 1 (PTZA1), AND CALCULATES ERROR MATRIX =
C     ('A'*INVERSE-'A')-IDENTITY.

      REAL*8 T(11),A(20,20),AI(20,20),ERR(20,20),ERRSZ,EPS,
     1  RSUM,MMAX(2)
      DATA T/3.,2.,6.,1.,4.,8.,4.,0.,5.,3.,1./,MMAX/1D70,1D3/

C  MATRIX DIMENSIONS
      ID1=20
      N=6
      N2M1=2*N-1
      NM1=N-1

C  GEN. 'A' FROM ITS 1ST ROW AND COL, GET SPECS. FOR ILL-COND
C  SUBMATRIX, AND ADJUST TOEP.DIAG. ACCORDINGLY
      CALL TOEPMX(T,A,N,N2M1,ID1)
      WRITE(6,3)
    3 FORMAT(/' SIZE OF ILL-COND BLOCK, DIST FROM SINGULARITY')
      READ(5,*) NSIZE,EPS
      CALL ADJSMX(T,A,N,N2M1,ID1,NSIZE,1,1,EPS)

    9 WRITE(6,10)
   10 FORMAT(///' INPUT TOEPLITZ MATRIX T')
      CALL PRNTMX(A,N,N,ID1)

C  INVERT A AND PRINT ERROR MATRIX (A*A.INV - IDENTITY) FOR
C  TZA (WHEN MMAX=1D70) AND PTZA-1
      DO 50 IMX=1,2
      CALL PTZA1(A,AI,MMAX(IMX),N,ID1)
      IF(IMX.EQ.1) WRITE(6,22)
   22 FORMAT(//' *** RESULTS FOR TRENCH-ZOHAR ALG.***')
      IF(IMX.EQ.2) WRITE(6,23)
   23 FORMAT(//' *** RESULTS FOR PIVOTED TRENCH-ZOHAR ALG., ',
     1               'STRATEGY 1 ***')
      WRITE(6,24)
   24 FORMAT(///' INVERSE OF T')
      CALL PRNTMX(AI,N,N,ID1)

      CALL MULTMX(A,AI,ERR,N,N,N,ID1,ID1,ID1)
         DO 25 I=1,N
   25    ERR(I,I)=ERR(I,I)-1.
      WRITE(6,27)
   27 FORMAT(//' ERROR MATRIX = T*T.INVERSE - IDENTITY')
         DO 28 I=1,N
   28    WRITE(6,29) (ERR(I,J),J=1,N)
   29    FORMAT(1X,6D12.3)

      ERRSZ=0.
         DO 32 I=1,N
```

```
      RSUM=0.
         DO 30 J=1,N
30       RSUM=RSUM+DABS(ERR(I,J))
      IF(RSUM.GT.ERRSZ) ERRSZ=RSUM
32    CONTINUE
   WRITE(6,35) ERRSZ
35 FORMAT(/' INFINITY-NORM OF ERROR MATRIX = ',D12.3)
50 CONTINUE
   STOP
   END



   SUBROUTINE PTZA1(T,TI,MMAX,N,LR)
C
C  S/R TO INVERT A TOEPLITZ MATRIX USING THE PIVOTED TRENCH-ZOHAR
C  ALGORITHM, STRATEGY 1.
C
C  T - I/P TOEPLITZ MATRIX; TI - (O/P) INVERSE OF T;
C  MMAX - PIVOTING THRESHOLD FOR MULTIPLIER ;
C  N,LR - ORDER, ROW-DIMENSION OF T AND TI
C
C  BRIEF DESCRIPTION OF PTZA1.
C  FOLLOWS ALG.6.9.1, WITH THE SAME NOTATION. EXECUTES
C  TRENCH-ZOHAR ALG (ACTUALLY, ALTERNATIVE BAREISS SYMMETRIC
C  ALGORITHM) UNTIL MULTIPLIER EXCEEDS MMAX. IF THE OTHER
C  MULTIPLIER IS SMALLER, DO THE SIMPLE PIVOTING PROCEDURE.
C  THIS IS EXACTLY ANALOGOUS TO THE SIMPLE PIVOTING
C  PROCEDURE IN PBA1, WITH THE FIRST ROWS OF THE TOEPLITZ
C  PART OF THE MUTIPLIER MATRICES BEING ACCUMULATED,AND ONLY
C  THOSE DIAGS OF THE BAREISS ITERATES ABOVE (TMA, NEG.INDEX,
C  TPA, POS.INDEX) AND BELOW (TMB, NEG.INDEX, TPB,
C  POS.INDEX) THE ZERO-BANDS BEING CALCULATED, USING EITHER
C  THE BAREISS RECURSION OR THE MULTIPLIER MATRICES.
C
C  OTHER TEMPORARIES:
C  K    - BAREISS CYCLE NO.
C  MM,MP- 1ST ROWS OF TOEP.PTS.OF NEG.,POS.INDEX MULTIPLIER
C         MATRICES
C  MMSV,ETC: TEMPORARY AREA FOR SAVING MM, ETC.
C
      REAL*8 T(LR,N),TI(LR,N)
      REAL*8 MMAX,TMA,TMB,TPA,TPB,MM(20),MP(20),M
      REAL*8 MMSV(20),TMASV,TMBSV,MPSV(20),TPASV,TPBSV
      REAL*8 INPROD

      NM1=N-1
      NM2=N-2

C  TZA INITIALIZATION : K,MM,MP,TM,TP
      K=0
      MP(1)=1.
      MM(1)=1.
         DO 2 I=2,N
         MP(I)=0.
    2    MM(I)=0.
      TMA=T(1,1)
```

```
      TMB=T(2,1)
      TPA=T(1,2)
      TPB=T(1,1)

C  NORMAL TZA LOOP, WITH PIVOT SELECTION TEST. K IS CYCLE NO.
C
C  STEP (-K)
    5 K=K+1
      TMASV=TMA
      TMBSV=TMB
      CALL MOVMTX(MM,MMSV,N,1,N)
      M=TMB/TPB
      CALL MCALC(MP,0,MM,1,M,N)
      TMA=TMA-M*TPA
      IF(K.NE.NM1) TMB=INPROD(MM,1,T,1,N,LR)

C  STEP (K)
      TPASV=TPA
      TPBSV=TPB
      CALL MOVMTX(MP,MPSV,N,1,N)
      M=TPA/TMASV
      CALL MCALC(MMSV,1,MP,0,M,N)
      TPB=TPB-M*TMBSV
      IF(K.EQ.NM1) GO TO 200
      TPA=INPROD(MP,0,T,K+2,N,LR)
C
C  PIVOT IF MULTIPLIER INVOLVING TPA IS TOO LARGE AND IF
C  ALTERNATE MULTIPLIER IS SMALLER, ELSE CONTINUE WITH
C  TZA
      IF(DABS(TPA/TMA).LE.MMAX.OR.DABS(TPA/TMA).LE.DABS(TPB/TMB))
     1  GO TO 5

C  *** END OF TZA LOOP ***



C  *** SIMPLE PIVOTING PROCEDURE ***

C  1. RE-DO STEP K: WITH TMB AS PIVOT, ELIMINATE TPB
      TPA=TPASV
      TPB=TPBSV
      CALL MOVMTX(MPSV,MP,N,1,LR)
      M=TPB/TMB
      CALL MCALC(MM,1,MP,0,M,N)
      TPB=INPROD(MP,1,T,1,N,LR)
C
C  2. WITH TPA AS PIVOT, ELIMINATE TMA
      M=TMA/TPA
      CALL MCALC(MP,0,MM,0,M,N)
      TMB=TMB-M*TPB
      TMA=INPROD(MM,0,T,K+2,N,LR)

C  3. RESTORE TO BAREISS FORM BY USING TMA TO ELIM. TPA
      M=TPA/TMA
      CALL MCALC(MM,0,MP,1,M,N)
      TPA=INPROD(MP,0,T,K+3,N,LR)
      K=K+1
```

```
      GO TO 5
C
C  *** END OF PIVOTING PROCEDURE ***
C
C  CALC 1ST ROW AND COL OF TI.
  200 DO 210 I=1,N
      TI(N-I+1,1)=MM(I)/TMA
  210 TI(1,I)=MP(I)/TPB
C
C  USE ZOHAR'S RECURSION TO FILL TI TO SECONDARY DIAG
      DO 215 I=1,NM2
      NMIM1=N-I-1
      DO 215 J=1,NMIM1
  215 TI(I+1,J+1)=TI(I,J) +
     1  (TI(I+1,1)*TI(1,J+1)-TI(1,N-I+1)*TI(N-J+1,1))/TI(1,1)

C  FILL IN REST OF TI USING PERSYMMETRY
      DO 220 I=2,N
      NMIP2=N-I+2
      DO 220 J=NMIP2,N
  220 TI(I,J)=TI(N-J+1,N-I+1)
      RETURN
      END



      DOUBLE PRECISION FUNCTION INPROD(V,NS,A,IC,N,LR)
C
C  FUNCTION TO FIND THE INNER PRODUCT OF V(N), DOWNSHIFTED
C  NS PLACES, WITH COLUMN IC OF MATRIX A.
C  N AND LR ARE THE ORDER AND ROW-DIMENSION OF A
C
      REAL*8 V(N),A(LR,N)

      NMNS=N-NS
      INPROD=0.
      DO 10 I=1,NMNS
      INPROD=INPROD + V(I)*A(I+NS,IC)
   10 CONTINUE
      RETURN
      END



C  SUBROUTINES ADJSMX AND MOVMTX ARE AS IN PBA2, AND
C  SUBROUTINES MCALC AND RSHIFT ARE AS IN PBA3
C
C
C
```

```
SIZE OF ILL-COND BLOCK, DIST FROM SINGULARITY
?
3,5d-13


INPUT TOEPLITZ MATRIX T
    8.000     4.000     1.000     6.000     2.000     3.000
    4.000     8.000     4.000     1.000     6.000     2.000
  -34.000     4.000     8.000     4.000     1.000     6.000
    5.000   -34.000     4.000     8.000     4.000     1.000
    3.000     5.000   -34.000     4.000     8.000     4.000
    1.000     3.000     5.000   -34.000     4.000     8.000

*** RESULTS FOR TRENCH-ZOHAR ALG.***


INVERSE OF T
    0.042    -0.010    -0.022    -0.001    -0.004     0.004
    0.011     0.018    -0.002    -0.025    -0.002    -0.004
    0.002     0.030     0.005     0.002    -0.025    -0.001
    0.034    -0.001     0.010     0.005    -0.002    -0.022
   -0.131     0.173    -0.001     0.030     0.018    -0.010
    0.214    -0.131     0.034     0.002     0.011     0.042

ERROR MATRIX = T*T.INVERSE - IDENTITY
  -0.320D-01  -0.316D-01  -0.135D-01  -0.116D-01  -0.679D-02  -0.330D-02
  -0.605D-01  -0.160D-02  -0.110D-01  -0.979D-02  -0.575D-02  -0.122D-01
  -0.935D-01   0.224D-01   0.835D-02   0.397D-03  -0.371D-03  -0.123D-01
  -0.179D+00   0.956D-02   0.754D-01   0.215D-01   0.178D-01  -0.157D-01
   0.416D-01  -0.109D+00  -0.602D-01   0.871D-01   0.185D-01   0.260D-01
   0.126D+00  -0.118D+00  -0.929D-01  -0.832D-01   0.739D-01   0.250D-01

INFINITY-NORM OF ERROR MATRIX =    0.520D+00

*** RESULTS FOR PIVOTED TRENCH-ZOHAR ALG., STRATEGY 1 ***


INVERSE OF T
    0.041    -0.008    -0.021     0.000    -0.004     0.004
    0.008     0.018     0.000    -0.025    -0.001    -0.004
    0.006     0.028     0.003     0.004    -0.025     0.000
    0.039    -0.003     0.009     0.003     0.000    -0.021
   -0.120     0.170    -0.003     0.028     0.018    -0.008
    0.213    -0.120     0.039     0.006     0.008     0.041

ERROR MATRIX = T*T.INVERSE - IDENTITY
  -0.278D-16   0.222D-15   0.0       0.162D-15   0.651D-16   0.278D-16
   0.291D-15  -0.472D-15   0.111D-15  -0.226D-16   0.128D-15   0.971D-16
   0.444D-15  -0.208D-15   0.222D-15   0.165D-16  -0.173D-16   0.971D-16
  -0.167D-15  -0.555D-15   0.141D-15   0.0         0.295D-16  -0.668D-16
  -0.301D-14   0.555D-15   0.860D-15   0.191D-15   0.444D-15  -0.278D-15
   0.178D-14  -0.197D-14  -0.888D-15   0.112D-14   0.694D-16   0.666D-15

INFINITY-NORM OF ERROR MATRIX =    0.649D-14
READY
```

```
SIZE OF ILL-COND BLOCK, DIST FROM SINGULARITY
?
3,1d-8


INPUT TOEPLITZ MATRIX T
    8.000     4.000     1.000     6.000     2.000     3.000
    4.000     8.000     4.000     1.000     6.000     2.000
  -34.000     4.000     8.000     4.000     1.000     6.000
    5.000   -34.000     4.000     8.000     4.000     1.000
    3.000     5.000   -34.000     4.000     8.000     4.000
    1.000     3.000     5.000   -34.000     4.000     8.000

*** RESULTS FOR TRENCH-ZOHAR ALG.***


INVERSE OF T
    0.041    -0.008    -0.021     0.000    -0.004     0.004
    0.008     0.018     0.000    -0.025    -0.001    -0.004
    0.006     0.028     0.003     0.004    -0.025     0.000
    0.039    -0.003     0.009     0.003     0.000    -0.021
   -0.120     0.170    -0.003     0.028     0.018    -0.008
    0.213    -0.120     0.039     0.006     0.008     0.041

ERROR MATRIX = T*T.INVERSE - IDENTITY
  -0.851D-07  -0.159D-05  -0.107D-06  -0.272D-06  -0.228D-06   0.280D-07
  -0.198D-05   0.151D-06  -0.502D-06  -0.148D-06  -0.836D-07  -0.435D-06
  -0.106D-05   0.475D-06  -0.583D-06  -0.984D-07   0.203D-07  -0.105D-06
  -0.408D-05   0.273D-06   0.224D-05  -0.513D-06   0.313D-06  -0.373D-06
  -0.135D-04   0.441D-06   0.603D-05   0.250D-05   0.850D-06  -0.981D-06
   0.103D-04  -0.104D-04  -0.768D-05   0.704D-05   0.164D-05   0.209D-05

INFINITY-NORM OF ERROR MATRIX =    0.392D-04

*** RESULTS FOR PIVOTED TRENCH-ZOHAR ALG., STRATEGY 1 ***


INVERSE OF T
    0.041    -0.008    -0.021     0.000    -0.004     0.004
    0.008     0.018     0.000    -0.025    -0.001    -0.004
    0.006     0.028     0.003     0.004    -0.025     0.000
    0.039    -0.003     0.009     0.003     0.000    -0.021
   -0.120     0.170    -0.003     0.028     0.018    -0.008
    0.213    -0.120     0.039     0.006     0.008     0.041

ERROR MATRIX = T*T.INVERSE - IDENTITY
  -0.167D-15  -0.971D-16  -0.416D-16   0.486D-16  -0.147D-16  -0.139D-16
  -0.278D-15  -0.444D-15   0.139D-16  -0.416D-16   0.538D-16  -0.278D-16
   0.222D-15  -0.153D-15   0.0         0.278D-16  -0.460D-16   0.555D-16
   0.416D-15  -0.472D-15  -0.989D-16  -0.278D-16  -0.425D-16  -0.199D-16
  -0.622D-14   0.148D-14   0.273D-14  -0.153D-15   0.444D-15  -0.652D-15
   0.200D-14  -0.334D-14  -0.126D-14   0.334D-14  -0.264D-15   0.888D-15

INFINITY-NORM OF ERROR MATRIX =    0.117D-13
READY
```

```
SIZE OF ILL-COND BLOCK, DIST FROM SINGULARITY
?
3,1d-15


INPUT TOEPLITZ MATRIX T
    8.000     4.000     1.000     6.000     2.000     3.000
    4.000     8.000     4.000     1.000     6.000     2.000
  -34.000     4.000     8.000     4.000     1.000     6.000
    5.000   -34.000     4.000     8.000     4.000     1.000
    3.000     5.000   -34.000     4.000     8.000     4.000
    1.000     3.000     5.000   -34.000     4.000     8.000

*** RESULTS FOR TRENCH-ZOHAR ALG.***


INVERSE OF T
   -0.008     0.004    -0.001    -0.004     0.003    -0.000
    0.038    -0.026     0.006     0.010    -0.012     0.003
   -0.107     0.086    -0.024     0.012     0.010    -0.004
    0.162    -0.181     0.084    -0.024     0.006    -0.001
    0.036     0.145    -0.181     0.086    -0.026     0.004
   -0.017     0.036     0.162    -0.107     0.038    -0.008


ERROR MATRIX = T*T.INVERSE - IDENTITY
  -0.244D-01  -0.672D+00   0.624D+00  -0.271D+00   0.810D-01  -0.159D-01
   0.192D+00  -0.866D-01  -0.727D+00   0.391D+00  -0.124D+00   0.132D-01
   0.140D+00   0.911D-01  -0.105D-01  -0.381D+00   0.156D+00  -0.514D-01
  -0.347D+00   0.432D+00  -0.185D+00  -0.126D+01   0.457D+00  -0.116D+00
   0.468D+01  -0.247D+01   0.372D+00  -0.202D+00  -0.142D+01   0.146D+00
  -0.593D+01   0.737D+01  -0.240D+01   0.379D+00   0.155D-01  -0.103D+01

INFINITY-NORM OF ERROR MATRIX =    0.171D+02

*** RESULTS FOR PIVOTED TRENCH-ZOHAR ALG., STRATEGY 1 ***


INVERSE OF T
    0.041    -0.008    -0.021     0.000    -0.004     0.004
    0.008     0.018     0.000    -0.025    -0.001    -0.004
    0.006     0.028     0.003     0.004    -0.025     0.000
    0.039    -0.003     0.009     0.003     0.000    -0.021
   -0.120     0.170    -0.003     0.028     0.018    -0.008
    0.213    -0.120     0.039     0.006     0.008     0.041


ERROR MATRIX = T*T.INVERSE - IDENTITY
  -0.583D-15   0.569D-15  -0.139D-16   0.181D-15   0.616D-16  -0.278D-16
   0.555D-15  -0.888D-15   0.236D-15   0.226D-16   0.114D-15   0.139D-15
   0.0        -0.180D-15   0.0         0.859D-16   0.694D-17   0.833D-16
   0.555D-15  -0.680D-15  -0.114D-15  -0.694D-16   0.156D-16   0.408D-16
  -0.665D-14   0.119D-14   0.294D-14  -0.191D-15   0.444D-15  -0.666D-15
   0.289D-14  -0.390D-14  -0.171D-14   0.359D-14  -0.361D-15   0.888D-15

INFINITY-NORM OF ERROR MATRIX =    0.133D-13
READY
```

```
C  TEST PROGRAM FOR FTO1, WHICH ORTHOGONALLY-TRIANGULARIZES A
C  TOEPLTIZ MATRIX IN O(N**2) OPERATIONS. FTO1 USES THE
C  GILL-GOLUB-MURRAY-SAUNDERS ORTHOGONAL-UPDATE ALGORITHM
C  AND THE SHIFT-INVARIANCE OF THE TOEPLITZ MATRIX.
C
C  TEST PROCEDURE: (1) GENERATE TOEP.MX. 'A' WITH 1ST ROW
C  AND COL FROM 'T'. (2) REDUCE TOEP.MX. WITH HOUSEHOLDER
C  TRANSFORMS (O/P OVERWRITES 'A') TO UPR.TRIANGULAR FORM
C  (3) CALL FTO1 TO PRODUCE UPR.TRIANGLE IN 'R'.
C
C  OTHER VARIABLES: N=MATRIX ORDER, ID1=ROW DIM.OF MATRICES
C
      REAL*8 T(19),A(10,10),R(10,10),W1(10),W2(10),W3(10),
     1  W4(10),W5(10),W6(10),W7(10),W8(10),W9(10),W10(10),W11(10),
     2  W12(10),W13(10),W14(10),W15(10),W16(10),W17(10)
      DATA T/2.2,3.7,-5.9,-1.1,-2.8,6.3,-4.9,8.0,
     1  6.2,18.6,-14.7,2.5,2.9,4.2,-8.6,-1.5,2.0,
     2  0.2,-1.5/

      ID1=10
      N=10
      N2M1=2*N-1
      NM1=N-1

      CALL TOEPMX(T,A,N,N2M1,ID1)
      WRITE(6,10)
   10 FORMAT(////////////' INPUT TOEPLITZ MATRIX T')
      CALL PRNTMX(A,N,N,ID1)

C  REDUCE TOEPLITZ MATRIX BY HOUSEHOLDER TRANSFORMS
      CALL HOUTRI(A,N,ID1)
      DO 30 I=2,N
      IM1=I-1
      DO 30 J=1,IM1
   30 A(I,J)=0.

      WRITE(6,40)
   40 FORMAT(/' REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS')
      CALL PRNTMX(A,N,N,ID1)

C  REDUCE TOEPLITZ MATRIX USING FTO1
      CALL FTO1(T,R,W1,W2,W3,W4,W5,W6,W7,W8,W9,W10,W11,W12,
     1                    W13,W14,W15,W16,W17,N,N2M1,ID1)
      DO 15 I=2,N
      IM1=I-1
      DO 15 J=1,IM1
   15 R(I,J)=0.

      WRITE(6,20)
   20 FORMAT(/' REDUCED MATRIX AS CALCULATED BY FTO1')
      CALL PRNTMX(R,N,N,ID1)
      STOP
      END



      SUBROUTINE FTO1(T,R,CPHIUP,SPHIUP,CPHIDN,SPHIDN,CTHEUP,
```

```
     1  STHEUP,CTHEDN,STHEDN,RM,RD1,RDJ,RT1,RTJ,QN,SIGD,SIGT,
     2  Z,N,N2M1,ID1)
C
C  S/R TO TRIANGULARIZE A TOEPLITZ MATRIX USING GILL ET AL'S
C  ORTHOGONAL UPDATE ALG. AND SHIFT-INVARIANCE. FOLLOWS ALG.
C  7.3.1, AND THE VARIABLE NOTATION IS THE SAME.
C
C
C  PARAMETERS
C  T.........VECTOR CONTAINING 1ST ROW AND COL OF TOEPLITZ MATRIX
C            TO BE TRIANGULARIZED. IF N IS THE ORDER OF THE MATRIX,
C            ORDER OF STORAGE IS (1,N),(1,N-1),...,(1,1),(2,1),...
C            ...,(N,1)
C  R.........CONTAINS UPPER-TRIANGULAR MATRIX PRODUCED BY THE
C            ORTHOGONALIZATION. NOTE THAT R(I,J),I.GT.J HAVE
C            NOT BEEN ZEROED
C  CPHIUP)
C  SPHIUP)   OUTPUT COSINES AND SINES OF GIVENS ROTATIONS USED
C  CPHIDN)   IN FTO1. NOTATION FOLLOWS ALG.7.3.1, WITH THE
C  SPHIDN)..FOLLOWING CONVENTION
C  CTHEUP)   1ST LETTER. C-COSINE, S-SINE
C  STHEUP)   NEXT 3 LETTERS. PHI-PHI, THE-THETA
C  CTHEDN)   LAST 2 LETTERS. UP-UP ARROW, DN-DOWN ARROW
C  STHEDN)
C  RM........TEMP. VECTOR TO STORE COL. OF R(SUPERSCRIPT N-1)
C  RD1.......O/P VECTOR CONTAINING 1ST COL OF R-DOT
C  RDJ.......TEMP. VECTOR TO STORE COL.J OF R-DOT.
C  RT1.......O/P VECTOR CONTAINING 1ST COL OF R-TILDE
C  RTJ.......TEMP. VECTOR TO STORE COL.J OF R-TILDE
C  QN........O/P VECTOR CONTAINING LAST ROW OF Q
C  SIGD......O/P VECTOR CONTAINING VALUES OF SIGMA-DOT
C  SIGT......O/P VECTOR CONTAINING VALUES OF SIGMA-TILDE
C  Z.........O/P VECTOR (= T-TRANSPOSE * (T(2,1),...,T(N,1))
C                                                  TRANSPOSE )
C  N.........ORDER OF TOEPLITZ MATRIX AND R, AND LENGTH OF
C            VECTORS CPHIUP,...,Z.
C  N2M1......2*N-1. T MUST BE AT LEAST THIS LENGTH
C  ID1.......ROW-DIMENSION OF R
C
C  GENERAL REMARKS ON CALLING FTO1. THE ONLY INPUTS REQUIRED
C  ARE T,N,N2M1 AND ID1. THE MAIN OUTPUT IS IN R. THE OTHER
C  17 VECTORS ARE BY-PRODUCTS AND TEMPORARIES OF FTO1, AND
C  NORMALLY THE USER ONLY ALLOCATES SPACE FOR THESE VECTORS
C  WITHOUT DOING ANYTHING MORE WITH THEM. HOWEVER, THE GIVENS
C  TRANSFORM VECTORS CPHIUP,...,STHEDN CAN BE USED TO GENERATE
C  THE Q-MATRIX EXPLICITLY USING ALG.7.4.1.
C
C  QUICK OUTLINE OF FTO1
C  ---------------------
C  DEFINE
C  RM = R(SUPERSCRIPT N-1) = ORTHOGONAL TRIANGULAR FACTOR
C           OF LEADING SUBMATRIX OF T.
C  QM = Q(SUPERSCRIPT N-1) = OTHOGONAL MATRIX SUCH THAT
C           LEADING SUBMATRIX OF T = QM*RM
C  R-DOT = MATRIX WITH 1ST ROW = T(1,1..N), REST OF 1ST
C           COLUMN = QM-TRANSPOSE * T(2..N,1),
C           TRAILING SUBMATRIX = LEADING SUBMATRIX OF T.
```

```
C  R-TILDE = R + Q(N,1..N)*(T(N,1..N-1),0)
C
C  R-DOT MAY BE TRANSFORMED TO R USING GILL ET AL'S RANK-1
C  ORTHOGONAL UPDATE, WITH AN UPSWEEP AND A DOWNSWEEP OF
C  GIVENS TRANSFORMS (ANGLES PHI-UP AND PHI-DOWN). SIMILARLY,
C  R-TILDE MAY BE TRANSFORMED TO AN UPPER-TRIANGULAR MATRIX
C  WITH RM AS ITS LEADING SUBMATRIX (GIVENS ANGLES THETA-UP
C  AND THETA-DOWN)
C    SUPPOSE THAT RM(.J-1), R(.J-1) AND Q(N,1..J-1) ARE KNOWN.
C  THEN R-DOT(.J) CAN BE CONSTRUCTED. PERFORM A GILL UPSWEEP
C  AND DOWNSWEEP ON R-DOT(.J), YIELDING R(.J). CALCULATE
C  Q(N,J) USING THE RELATION Q(N,1..J)*R(1..J,J)=T(N,J),
C  AND CALCULATE R-TILDE(.J) FROM THE DEFINITION. PERFORM
C  A GILL UPSWEEP AND DOWNSWEEP ON R-TILDE(.J), YIELDING
C  RM(.J). THIS COMPLETES THE BASIC RECURSION. NOTE THAT
C  THE GILL SWEEPS AND THE CALCULATION OF Q(N,J) REQUIRE
C  ONLY O(N) OPERATIONS, SO THE WHOLE ALGORITHM REQUIRES
C  ONLY O(N**2) OPERATIONS. DETAILS OF THE CALCULATION
C  OF THE GILL ROTATIONS, WHICH PROCEEDS IN TANDEM WITH
C  THE BASIC RECURSION ARE GIVEN IN ALG.7.3.1.

      REAL*8 T(N2M1),R(ID1,N),CPHIUP(N),SPHIUP(N),CPHIDN(N),SPHIDN(N)
     1  ,CTHEUP(N),STHEUP(N),CTHEDN(N),STHEDN(N),RM(N),RD1(N),RDJ(N),
     2   RT1(N),RTJ(N),SIGD(N),SIGT(N),QN(N),Z(N)
      REAL*8 EPS

C  INITIALIZATION

      M=N-1
      EPS=10.**(-5)

C  CALC R(1,1) AND RM(1,1) USING NORM-INVARIANCE
      RM(1)=0.
      DO 10 I=1,M
   10 RM(1)=RM(1)+T(I-1+N)**2
      R(1,1)=RM(1)+T(N2M1)*T(N2M1)
      R(1,1)=DSQRT(R(1,1))
      RM(1)=DSQRT(RM(1))

C  GET R-DOT(1,1),(1,2),(2,2) AND SIGMA-DOT(1),(2), WHERE SIGMA-DOT(I)
C  IS NORM(R-DOT(I..N,1))
      RD1(1)=T(N)
      RDJ(1)=T(N-1)
      RDJ(2)=RM(1)
      SIGD(1)=R(1,1)
      SIGD(2)=DSQRT(SIGD(1)**2 - RD1(1)**2)

C  OTHER INITIALIZATION - PHIDN(1),THEDN(1),Q(N,1),R-TILDE(1,1), AND
C  SIGMA-TILDE(1),(2) WHERE SIGMA-TILDE(I)=NORM(R-TILDE(I..N,1))
      CPHIDN(1)=RD1(1)/SIGD(1)
      SPHIDN(1)=SIGD(2)/SIGD(1)
      QN(1)=T(N2M1)/R(1,1)
      RT1(1)=R(1,1)-QN(1)*T(N2M1)
      SIGT(1)=RM(1)
      SIGT(2)=DSQRT(SIGT(1)**2 - RT1(1)**2)
      CTHEDN(1)=RT1(1)/SIGT(1)
      STHEDN(1)=SIGT(2)/SIGT(1)
```

```
C  Z=TRANSPOSE OF PRINC SUBMX OF T * T(2..N,1)
      DO 28 I=1,M
      Z(I)=0.
      DO 28 J=1,M
   28 Z(I)=Z(I)+T(J-I+N)*T(J+N)

C*********MAIN LOOP - CALC R(.J) AND RM(.J)*****************

      J=2
C
C  PHASE I - CALC R(.J) FROM R-DOT(.J)
C
   30 RDJ(1)=T(1-J+N)
      DO 40 I=2,J
   40 RDJ(I)=RM(I-1)
C  RM CONTAINS RM(.J-1)
C
C  BACK-SUBST FOR R-DOT(J,1) IN EQN. RM(TRANS)*R-DOT(.1)=Z
      RD1(J)=Z(J-1)
      IF(J.EQ.2) GO TO 60
      JM2=J-2
      DO 50 I=1,JM2
   50 RD1(J)=RD1(J)-RM(I)*RD1(I+1)
   60 RD1(J)=RD1(J)/RM(J-1)

C  IF J=N, CHECK THAT RD1(N) IS POSITIVE. IF NOT, RD(N.) WAS
C  REVERSED IN SIGN IN THE PREVIOUS STEP - SO RDJ(N) MUST BE
C  REPLACED BY -RDJ(N).
      IF(J.NE.N) GO TO 65
      IF(RD1(N).LT.0) RDJ(N)=-RDJ(N)
      GO TO 70

C  NEXT PHIUP
   65 SIGD(J+1)=DSQRT(SIGD(J)**2-RD1(J)**2)
      CPHIUP(J)=RD1(J)/SIGD(J)
      SPHIUP(J)=SIGD(J+1)/SIGD(J)
C  SINGULARITY CHECK
      IF(DABS(SPHIUP(J)).GE.EPS) GO TO 68
      WRITE(6,66)
   66 FORMAT(' I/P MATRIX NUMERICALLY SINGULAR - ERROR EXIT')
      RETURN

C  UP-SWEEP ON R-DOT(.J)
   68 RDJ(J+1)=0.
      CALL GIVENS(RDJ(J),RDJ(J+1),CPHIUP(J),SPHIUP(J))
   70 IF(J.EQ.2) GO TO 90
      JM1=J-1
      I=JM1
      DO 80 II=2,JM1
      CALL GIVENS(RDJ(I),RDJ(I+1),CPHIUP(I),SPHIUP(I))
   80 I=I-1

C  DOWN-SWEEP ON R-DOT(.J), EXCLUDING LAST PHIDN
   90 DO 100 I=1,JM1
      CALL GIVENS(RDJ(I),RDJ(I+1),CPHIDN(I),SPHIDN(I))
  100 R(I,J)=RDJ(I)
```

```
C  CALC AND APPLY LAST PHIDN IN DOWN-SWEEP (N/A IF J=N)
      IF(J.NE.N) GO TO 110
      R(N,N)=RDJ(N)
      RETURN
  110 R(J,J)=DSQRT(RDJ(J)**2 + RDJ(J+1)**2)
      CPHIDN(J)=RDJ(J)/R(J,J)
      SPHIDN(J)=RDJ(J+1)/R(J,J)

C  PHASE II - CALC RM(.J) FROM R-TILDE(.J)

C  CALC Q(N,J) USING Q(N,1..J)*R(1..J,J)=T(N,J)
      QN(J)=T(N-J+N)
      DO 120 I=1,JM1
  120 QN(J)=QN(J)-QN(I)*R(I,J)
      QN(J)=QN(J)/R(J,J)

C  NEXT THEUP
      RT1(J)= -T(N2M1)*QN(J)
      SIGT(J+1)=DSQRT(SIGT(J)**2 - RT1(J)**2)
      CTHEUP(J)=RT1(J)/SIGT(J)
      STHEUP(J)=SIGT(J+1)/SIGT(J)
C  SINGULARITY CHECK
      IF(DABS(STHEUP(J)).GE.EPS) GO TO 124
      WRITE(6,123)
  123 FORMAT(' LDG SUBMX NUMERICALLY SINGULAR - ERROR EXIT',
     1       /' REMEDY - AUGMENT I/P MATRIX AND RERUN')
      RETURN

C  CALC R-TILDE(1..J,J)
  124 DO 125 I=1,J
  125 RTJ(I)=R(I,J)-QN(I)*T(N-J+N)

C  UP-SWEEP ON R-TILDE(.J)
      RTJ(J+1)=SIGT(J+1)*T(N-J+N)/T(N2M1)
      CALL GIVENS(RTJ(J),RTJ(J+1),CTHEUP(J),STHEUP(J))
      IF(J.EQ.2) GO TO 140
      I=JM1
      DO 130 II=2,JM1
      CALL GIVENS(RTJ(I),RTJ(I+1),CTHEUP(I),STHEUP(I))
  130 I=I-1
C
C  DOWN-SWEEP ON R-TILDE(.J), EXCLUDING LAST THEDN
  140 DO 150 I=1,JM1
      CALL GIVENS(RTJ(I),RTJ(I+1),CTHEDN(I),STHEDN(I))
  150 RM(I)=RTJ(I)
C
C  CALC AND APPLY LAST THEDN
      RM(J)=DSQRT(RTJ(J)**2 + RTJ(J+1)**2)
      CTHEDN(J)=RTJ(J)/RM(J)
      STHEDN(J)=RTJ(J+1)/RM(J)

      J=J+1
      GO TO 30
C
C*****************END OF MAIN LOOP*************************
```

```
INPUT TOEPLITZ MATRIX T
  18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900    3.700    2.200
 -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900    3.700
   2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900
   2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100
   4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800
  -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300
  -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900
   2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000
   0.200    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200
  -1.500    0.200    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600

REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS
 -26.016    7.747   -0.780    8.378  -17.201    9.468    2.254    5.784   -9.204    3.869
   0.0    -25.554    5.698    2.817    1.874  -13.458   10.343    5.309    2.054   -7.868
   0.0      0.0   -27.275    7.360   -0.716    4.318  -12.849   11.746    2.957    2.031
   0.0      0.0     0.0    -25.782    2.909   -0.052    6.139  -10.894   10.403    2.651
   0.0      0.0     0.0      0.0   -22.936    2.543   -2.887    4.710  -11.662   10.939
   0.0      0.0     0.0      0.0     0.0    -21.736    1.355   -3.165    4.339   -6.728
   0.0      0.0     0.0      0.0     0.0      0.0    -20.661    2.342   -1.687    2.130
   0.0      0.0     0.0      0.0     0.0      0.0      0.0    -20.551    2.515   -3.749
   0.0      0.0     0.0      0.0     0.0      0.0      0.0      0.0    -20.393    0.606
   0.0      0.0     0.0      0.0     0.0      0.0      0.0      0.0      0.0     17.394

REDUCED MATRIX AS CALCULATED BY FT01
  26.016   -7.747    0.780   -8.378   17.201   -9.468   -2.254   -5.784    9.204   -3.869
   0.0     25.554   -5.698   -2.817   -1.874   13.458  -10.343   -5.309   -2.054    7.868
   0.0      0.0    27.275   -7.360    0.716   -4.318   12.849  -11.746   -2.957   -2.031
   0.0      0.0     0.0     25.782   -2.909    0.052   -6.139   10.894  -10.403   -2.651
   0.0      0.0     0.0      0.0    22.936   -2.543    2.887   -4.710   11.662  -10.939
   0.0      0.0     0.0      0.0     0.0     21.736   -1.355    3.165   -4.339    6.728
   0.0      0.0     0.0      0.0     0.0      0.0     20.661   -2.342    1.687   -2.130
   0.0      0.0     0.0      0.0     0.0      0.0      0.0     20.551   -2.515    3.749
   0.0      0.0     0.0      0.0     0.0      0.0      0.0      0.0     20.393   -0.606
   0.0      0.0     0.0      0.0     0.0      0.0      0.0      0.0      0.0    -17.394
READY
```

```
C  TEST PROGRAM FOR FTO2, WHICH FINDS THE Q-R DECOMPOSITION
C  OF A TOEPLITZ MATRIX IN O(N**2) OPERATIONS. FTO2 USES THE
C  GILL-GOLUB-MURRAY-SAUNDERS  ORTHOGONAL UPDATE ALGORITHM
C  AND THE SHIFT-INVARIANCE OF THE TOEPLITZ MATRIX TO
C  GIVE AN O(N) COLUMN-RECURSION FOR R AND AN O(N)
C  ROW-RECURSION FOR QT (Q-TRANSPOSE).
C
C
C
C  TEST PROCEDURE: (1) GENERATE TOEP.MX. 'A'  WITH 1ST ROW
C  AND COL FROM 'T'. (2) FIND Q-R DECOMP. OF 'A' USING
C  HOUSEHOLDER TRANSFORMS(R OVERWRITES 'A') (3) CALL FTO2
C  TO CALC. Q-R DECOMP.OF 'A'.
C
      REAL*8 T(19),A(10,10),R(10,10),QT(10,10),WKA(230)
      DATA T/2.2,3.7,-5.9,-1.1,-2.8,6.3,-4.9,8.0,
     1  6.2,18.6,-14.7,2.5,2.9,4.2,-8.6,-1.5,2.0,
     2  0.2,-1.5/

      EPS=10.**(-4)
      ID1=10
      N=10
      N2P3=2*N+3
      N2M1=2*N-1
      NM1=N-1

      CALL TOEPMX(T,A,N,N2M1,ID1)

      WRITE(6,10)
   10 FORMAT(///' INPUT TOEPLITZ MATRIX T')
      CALL PRNTMX(A,N,N,ID1)

C     REDUCE TOEPLITZ MATRIX BY HOUSEHOLDER TRANSFORMS
      CALL HOUQR(A,QT,N,ID1)
      DO 30 I=2,N
      IM1=I-1
      DO 30 J=1,IM1
   30 A(I,J)=0.

   35 WRITE(6,40)
   40 FORMAT(/' REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS')
      CALL PRNTMX(A,N,N,ID1)
      WRITE(6,50)
   50 FORMAT(/' QT = PRODUCT OF HOUSEHOLDER TRANSFORMS')
      CALL PRNTMX(QT,N,N,ID1)

C  REDUCE TOEPLITZ MATRIX USING FTO2
      CALL FTO2(T,QT,R,N,N2M1,ID1)
      DO 52 I=2,N
      IM1=I-1
      DO 52 J=1,IM1
   52 R(I,J)=0.

   56 WRITE(6,58)
   58 FORMAT(/' REDUCED MATRIX AS CALCULATED BY FTO2')
      CALL PRNTMX(R,N,N,ID1)
      WRITE(6,60)
```

```
   60 FORMAT(/' QT = ORTHOG MATRIX THAT REDUCES T')
      CALL PRNTMX(QT,N,N,ID1)
      STOP
      END


      SUBROUTINE FTO2(T,Y,R,N,N2M1,ID1)
C
C  S/R TO FIND THE Q-R DECOMPOSITION OF A TOEPLITZ MATRIX USING
C  GILL ET-AL'S ORTHOGONAL UPDATE ALG., SHIFT-INVARIANCE, AND A
C  A RECURSION ON THE ROWS OF Q-TRANSPOSE. FOLLOWS ALG.7.4.2.,
C  WITH THE SAME NOTATION.
C
C  PARAMETERS
C  T........VECTOR CONTAINING 1ST ROW AND COL OF TOEPLITZ MATRIX
C           TO BE TRIANGULARIZED. IF N IS THE ORDER OF THE MATRIX,
C           ORDER OF STORAGE IS (1,N),(1,N-1),...,(1,1),(2,1),...
C           ...,(N,1)
C  Y........ORTHOGONAL MATRIX SATISFYING YT=R, R UPPER-TRIANGULAR
C  R........CONTAINS UPPER-TRIANGULAR MATRIX PRODUCED BY THE
C           ORTHOGONALIZATION. NOTE THAT R(I,J),I.GT.J HAVE
C           NOT BEEN ZEROED
C  N........ORDER OF TOEPLITZ MATRIX
C  N2M1.....2*N-1. T MUST BE AT LEAST THIS LENGTH
C  ID1......ROW-DIMENSION OF R
C
C  QUICK OUTLINE OF FTO2
C  ---------------------
C  DEFINE
C  TM = T(SUPERSCRIPT N-1) = LEADING SUBMATRIX OF T
C  RM = R(SUPERSCRIPT N-1) = ORTHOGONAL TRIANGULAR FACTOR OF TM
C  YM = Y(SUPERSCRIPT N-1) = ORTHOGONAL MATRIX SUCH THAT YM*TM=RM
C  R-DOT = MATRIX WITH 1ST ROW = T(1,1..N), REST OF 1ST
C          COLUMN = YM * T(2..N,1),
C          TRAILING SUBMATRIX = LEADING SUBMATRIX OF T.
C  R-TILDE = R + Y(N.)*(T(N,1..N-1),0)
C
C  R-DOT MAY BE TRANSFORMED TO R USING GILL ET AL'S RANK-1
C  ORTHOGONAL UPDATE, WITH AN UPSWEEP AND A DOWNSWEEP OF
C  GIVENS TRANSFORMS (ANGLES PHI-UP AND PHI-DOWN). SIMILARLY,
C  R-TILDE MAY BE TRANSFORMED TO AN UPPER-TRIANGULAR MATRIX
C  WITH RM AS ITS LEADING SUBMATRIX (GIVENS ANGLES THETA-UP
C  AND THETA-DOWN)
C    SUPPOSE THAT RM(.J-1), R(.J-1), YM(J-1.) AND Y(J.) ARE
C  KNOWN. THEN R-DOT(.J) CAN BE CONSTRUCTED. PERFORM A GILL
C  UPSWEEP AND DOWNSWEEP ON R-DOT(.J), YIELDING R(.J).
C  Y(J.)  MAY BE OBTAINED FROM YM(J-1.) IN A SIMILAR MANNER,
C  RE-ARRANGING THE CALCULATIONS TO OPERATE ON ROWS, RATHER
C  THAN COLUMNS.
C    CALCULATE R-TILDE(.J) FROM THE DEFINITION, AND PERFORM
C  A GILL UPSWEEP AND DOWNSWEEP ON R-TILDE(.J), YIELDING
C  RM(.J). YM(J.) MAY BE OBTAINED FROM Y(J.) IN A SIMILAR
C  MANNER, RE-ARRANGING THE CALCULATIONS TO OPERATE ON ROWS,
C  RATHER THAN COLUMNS. THIS COMPLETES THE BASIC RECURSION.
C           NOTE THAT THE GILL SWEEPS IN THE RECURSION, WHICH
C  ACCOUNT FOR MOST OF THE COMPUTING LOAD, REQUIRE ONLY
```

```
C  O(N) OPERATIONS, SO THE WHOLE ALGORITHM REQUIRES
C  ONLY O(N**2) OPERATIONS. DETAILS OF THE CALCULATION
C  OF THE GILL ROTATIONS, WHICH PROCEEDS IN TANDEM WITH
C  THE BASIC RECURSION ARE GIVEN IN ALG.7.4.2.

C  NOTATION - WORKING STORAGE
C  --------------------------
C  CPHIUP)
C  SPHIUP)  OUTPUT COSINES AND SINES OF GIVENS ROTATIONS USED
C  CPHIDN)  IN FTO2. NOTATION FOLLOWS ALG.7.4.2, WITH THE
C  SPHIDN)..FOLLOWING CONVENTION
C  CTHEUP)  1ST LETTER. C-COSINE, S-SINE
C  STHEUP)  NEXT 3 LETTERS. PHI-PHI, THE-THETA
C  CTHEDN)  LAST 2 LETTERS. UP-UP ARROW, DN-DOWN ARROW
C  STHEDN)
C  RM.......TEMP. VECTOR TO STORE COL. OF R(SUPERSCRIPT N-1)
C  RD1......O/P VECTOR CONTAINING 1ST COL OF R-DOT
C  RDJ......TEMP. VECTOR TO STORE COL.J OF R-DOT.
C  RT1......O/P VECTOR CONTAINING 1ST COL OF R-TILDE
C  RTJ......TEMP. VECTOR TO STORE COL.J OF R-TILDE
C  SIGD.....O/P VECTOR CONTAINING VALUES OF SIGMA-DOT
C  SIGT.....O/P VECTOR CONTAINING VALUES OF SIGMA-TILDE
C  Z........O/P VECTOR (= T-TRANSPOSE * (T(2,1),...,T(N,1))
C                                              TRANSPOSE )
C  YD.......TEMP.VECTOR CONTAINING Y-DOT(J.)
C  YD1......TEMP.VECTOR CONTAINING Y-DOT-1(J.), AND OVERWRITTEN
C                                  BY Y-DOT-1(J+1.)
C  YD2......TEMP.VECTOR CONTAINING Y-DOT-2(J+1.)
C  YD3......TEMP.VECTOR CONTAINING Y-DOT-3(J.), AND OVERWRITTEN
C                                  BY Y-DOT-3(J+1.)
C  Y1 TO Y3.SIMILAR TO YD1 TO YD3, BUT REFERRING TO Y.1 TO Y.3
C  YM.......TEMP.VECTOR CONTAINING YM(J.)

      REAL*8 T(N2M1),R(ID1,N),CPHIUP(100),SPHIUP(100),CPHIDN(100),
     1  CTHEUP(100),STHEUP(100),CTHEDN(100),STHEDN(100),RM(100),RD1(100)
     2  ,RT1(100),RTJ(100),SIGD(100),SIGT(100),SPHIDN(100),RDJ(100)
      REAL*8 Y(ID1,N),YD(100),YD1(100),YD2(100),YD3(100),Y1(100),Y2(100)
     1      ,YM(100),Y3(100)
      REAL*8 EPS,TEMP

C  INITIALIZATION

      IF(N.LE.100) GO TO 5
      WRITE(6,2)
    2 FORMAT(' ** N.GT.100 - NO PROCESSING BY FTO2')
      RETURN

    5 EPS=10.**(-5)
      M=N-1

C  CALC R(1,1) AND RM(1,1) USING NORM-INVARIANCE
      RM(1)=0.
      DO 10 I=1,M
   10 RM(1)=RM(1)+T(I-1+N)**2
      R(1,1)=RM(1)+T(N2M1)*T(N2M1)
      R(1,1)=DSQRT(R(1,1))
      RM(1)=DSQRT(RM(1))
```

```
C  GET R-DOT(1,1),(1,2),(2,2) AND SIGMA-DOT(1),(2), WHERE SIGMA-DOT(I)
C  IS NORM(R-DOT(I..N,1))
      RD1(1)=T(N)
      RDJ(1)=T(N-1)
      RDJ(2)=RM(1)
      SIGD(1)=R(1,1)
      SIGD(2)=DSQRT(SIGD(1)**2 - RD1(1)**2)

C  OTHER INITLZN.FOR R-RECURSION - PHIDN(1),THEDN(1),R-TILDE(1,1), AND
C  SIGMA-TILDE(1),(2) WHERE SIGMA-TILDE(I)=NORM(R-TILDE(I..N,1))
      CPHIDN(1)=RD1(1)/SIGD(1)
      SPHIDN(1)=SIGD(2)/SIGD(1)
      Y(1,N)=T(N2M1)/R(1,1)
      RT1(1)=R(1,1)-Y(1,N)*T(N2M1)
      SIGT(1)=RM(1)
      SIGT(2)=DSQRT(SIGT(1)**2 - RT1(1)**2)
      CTHEDN(1)=RT1(1)/SIGT(1)
      STHEDN(1)=SIGT(2)/SIGT(1)

C  INITLZN.FOR Q-RECURSION - Y-DOT(1.),YM(1.),Y(1.),Y-DOT-1(2.),
C  Y-DOT-3(2.),Y.1(2.) AND Y.3(2.)
      YD(1)=1.
      DO 12 I=2,N
   12 YD(I)=0.
      DO 14 I=1,M
   14 YM(I)=T(I-1+N)/RM(1)
      YM(N)=0.
      DO 16 I=1,N
      Y(1,I)=T(I-1+N)/R(1,1)
      YD1(I)=(Y(1,I)-CPHIDN(1)*YD(I))/SPHIDN(1)
      YD3(I)=-SPHIDN(1)*YD(I) + CPHIDN(1)*YD1(I)
      Y1(I)=(YM(I) - CTHEDN(1)*Y(1,I))/STHEDN(1)
   16 Y3(I)=-STHEDN(1)*Y(1,I) + CTHEDN(1)*Y1(I)

C******MAIN LOOP - CALC R(.J),Y(J.),RM(.J) AND YM(J.)*******

      J=2
C
C  R-RECURSION PHASE I - CALC R(.J) FROM R-DOT(.J)
C
   30 RDJ(1)=T(1-J+N)
      DO 40 I=2,J
   40 RDJ(I)=RM(I-1)
C  RM CONTAINS RM(.J-1)

C  CALC  R-DOT(J,1) = YM(J-1.)*T(2..N,1)
   48 RD1(J)=0.
      DO 50 I=1,M
   50 RD1(J)=RD1(J) + YM(I)*T(I+N)

C  IF J=N, CHECK THAT RD1(N) IS POSITIVE. IF NOT, RD(N.) WAS
C  REVERSED IN SIGN IN THE PREVIOUS STEP - SO RDJ(N) MUST BE
C  REPLACED BY -RDJ(N).
      IF(J.NE.N) GO TO 65
      IF(RD1(N).LT.0) RDJ(N)=-RDJ(N)
      GO TO 70
```

```
C  NEXT PHIUP
   65 SIGD(J+1)=DSQRT(SIGD(J)**2-RD1(J)**2)
      CPHIUP(J)=RD1(J)/SIGD(J)
      SPHIUP(J)=SIGD(J+1)/SIGD(J)
C  SINGULARITY CHECK
      IF(DABS(SPHIUP(J)).GE.EPS) GO TO 68
      WRITE(6,66)
   66 FORMAT(' I/P MATRIX NUMERICALLY SINGULAR - ERROR EXIT')
      RETURN

C  APPLY PHIUP(J), THEN PHI(1..J-1) ON RDJ
   68 RDJ(J+1)=0.
      CALL GIVENS(RDJ(J),RDJ(J+1),CPHIUP(J),SPHIUP(J))
   70 CALL RSWEEP(RDJ,R(1,J),CPHIUP,SPHIUP,CPHIDN,SPHIDN,J-1,J)

C  CALC AND APPLY LAST PHIDN IN DOWN-SWEEP (N/A IF J=N)
      IF(J.NE.N) GO TO 110
      DO 105 I=1,N
  105 Y(N,I)=YD3(I)
      RETURN
  110 R(J,J)=DSQRT(RDJ(J)**2 + RDJ(J+1)**2)
      CPHIDN(J)=RDJ(J)/R(J,J)
      SPHIDN(J)=RDJ(J+1)/R(J,J)

C  Q-RECURSION PHASE I - CALC Y-DOT(J.),Y-DOT-1(J+1.),Y-DOT-2(J+1.),
C  Y(J.) AND Y-DOT-3(J+1.)
      YD(1)=0.
      DO 112 I=2,N
  112 YD(I)=YM(I-1)
      DO 115 I=1,N
      YD1(I)=(YD1(I)-CPHIUP(J)*YD(I))/SPHIUP(J)
      YD2(I)=-SPHIUP(J)*YD(I) + CPHIUP(J)*YD1(I)
      Y(J,I)=CPHIDN(J)*YD3(I) + SPHIDN(J)*YD2(I)
  115 YD3(I)=-SPHIDN(J)*YD3(I) + CPHIDN(J)*YD2(I)

C  R-RECURSION PHASE II - CALC RM(.J) FROM R-TILDE(.J)

C  CALC R-TILDE(1..J,J)
  120 DO 125 I=1,J
  125 RTJ(I)=R(I,J)-Y(I,N)*T(N-J+N)

C  NEXT THEUP
  127 RT1(J)= -T(N2M1)*Y(J,N)
      SIGT(J+1)=DSQRT(SIGT(J)**2 - RT1(J)**2)
      CTHEUP(J)=RT1(J)/SIGT(J)
      STHEUP(J)=SIGT(J+1)/SIGT(J)
C  SINGULARITY CHECK
      IF(DABS(STHEUP(J)).GE.EPS) GO TO 129
      WRITE(6,128)
  128 FORMAT(' LDG SUBMX.NUMERICALLY SINGULAR - ERROR EXIT',
     1       /' REMEDY - AUGMENT I/P MATRIX AND RERUN')
      RETURN

C  APPLY THEUP(J), THEN THETA(1..J-1) TO R-TILDE(.J)
  129 RTJ(J+1)=SIGT(J+1)*T(N-J+N)/T(N2M1)
      CALL GIVENS(RTJ(J),RTJ(J+1),CTHEUP(J),STHEUP(J))
```

```
      CALL RSWEEP(RTJ,RM,CTHEUP,STHEUP,CTHEDN,STHEDN,J-1,J)
C
C  CALC AND APPLY LAST THEDN
  975 RM(J)=DSQRT(RTJ(J)**2 + RTJ(J+1)**2)
      CTHEDN(J)=RTJ(J)/RM(J)
      STHEDN(J)=RTJ(J+1)/RM(J)

C  Q-RECURSION PHASE II - CALC Y.1(J+1.),Y.2(J+1.),YM(J.) AND Y.3(J+1.)
  155 DO 160 I=1,N
      Y1(I)=(Y1(I) - CTHEUP(J)*Y(J,I))/STHEUP(J)
      Y2(I)=-STHEUP(J)*Y(J,I) + CTHEUP(J)*Y1(I)
      YM(I)=CTHEDN(J)*Y3(I) + STHEDN(J)*Y2(I)
  160 Y3(I)=-STHEDN(J)*Y3(I) + CTHEDN(J)*Y2(I)

  170 J=J+1
      GO TO 30
C
C***************END OF MAIN LOOP*************************

      END


       SUBROUTINE RSWEEP(P,R,CUP,SUP,CDN,SDN,NROT,J)
C
C  S/R TO APPLY A GILL-TYPE UP/DOWN SWEEP TO A VECTOR
C
C  P.........I/P VECTOR
C  R.........O/P VECTOR
C  CUP,SUP..COS'S AND SIN'S OF UPSWEEP ANGLES
C  CDN,SDN..COS'S AND SIN'S OF DOWNSWEEP ANGLES
C  NROT.....NO.OF DOWNWARD ROTATIONS TO APPLY. (NO.OF
C           UPWARD ROTATIONS IS ONE LESS THAN NROT)
C  J........LENGTH OF P AND R. IF J.GT.NROT+1,
C           R(NROT+2..J)=P(NROT+2..J)
C
      REAL*8 P(J),R(J),CUP(J),SUP(J),CDN(J),SDN(J)
C
C  UPSWEEP
      IF(NROT.EQ.1) GO TO 15
      I=NROT
      DO 10 II=2,NROT
      CALL GIVENS(P(I),P(I+1),CUP(I),SUP(I))
   10 I=I-1
C
C  DOWNSWEEP
   15 DO 20 I=1,NROT
   20 CALL GIVENS(P(I),P(I+1),CDN(I),SDN(I))

C  OUTPUT
      DO 30 I=1,J
   30 R(I)=P(I)
      RETURN
      END
```

```
INPUT TOEPLITZ MATRIX T
  18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900    3.700    2.200
 -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900    3.700
   2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900
   2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100
   4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800
  -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300
  -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900
   2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000
   0.200    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200
  -1.500    0.200    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600

REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS
 -26.016    7.747   -0.780    8.378  -17.201    9.468    2.254    5.784   -9.204    3.869
   0.0    -25.554    5.698    2.817    1.874  -13.458   10.343    5.309    2.054   -7.868
   0.0      0.0    -27.275    7.360   -0.716    4.318  -12.849   11.746    2.957    2.031
   0.0      0.0      0.0    -25.782    2.909   -0.052    6.139  -10.894   10.403    2.651
   0.0      0.0      0.0      0.0    -22.936    2.543   -2.887    4.710  -11.662   10.939
   0.0      0.0      0.0      0.0      0.0    -21.736    1.355   -3.165    4.339   -6.728
   0.0      0.0      0.0      0.0      0.0      0.0    -20.661    2.342   -1.687    2.130
   0.0      0.0      0.0      0.0      0.0      0.0      0.0    -20.551    2.515   -3.749
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    -20.393    0.606
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0     17.394

QT = PRODUCT OF HOUSEHOLDER TRANSFORMS
  -0.715    0.565   -0.096   -0.111   -0.161    0.331    0.058   -0.077   -0.008    0.058
  -0.459   -0.557    0.546   -0.132   -0.162   -0.064    0.354    0.035   -0.081    0.010
  -0.389   -0.360   -0.565    0.515   -0.121   -0.129   -0.082    0.325    0.038   -0.073
  -0.198   -0.290   -0.373   -0.625    0.465   -0.033   -0.078   -0.091    0.333    0.057
   0.210   -0.281   -0.262   -0.293   -0.640    0.388   -0.131   -0.088   -0.143    0.342
   0.054    0.197   -0.297   -0.266   -0.355   -0.652    0.451   -0.116   -0.097   -0.149
  -0.110    0.108    0.216   -0.323   -0.206   -0.323   -0.641    0.496   -0.073   -0.124
  -0.111   -0.065    0.135    0.191   -0.280   -0.216   -0.372   -0.629    0.511   -0.088
  -0.173   -0.036   -0.035    0.091    0.241   -0.312   -0.257   -0.358   -0.611    0.487
   0.035    0.149    0.106    0.096   -0.033   -0.212    0.151    0.295    0.459    0.767

REDUCED MATRIX AS CALCULATED BY FT02
  26.016   -7.747    0.780   -8.378   17.201   -9.468   -2.254   -5.784    9.204   -3.869
   0.0     25.554   -5.698   -2.817   -1.874   13.458  -10.343   -5.309   -2.054    7.868
   0.0      0.0     27.275   -7.360    0.716   -4.318   12.849  -11.746   -2.957   -2.031
   0.0      0.0      0.0     25.782   -2.909    0.052   -6.139   10.894  -10.403   -2.651
   0.0      0.0      0.0      0.0     22.936   -2.543    2.887   -4.710   11.662  -10.939
   0.0      0.0      0.0      0.0      0.0     21.736   -1.355    3.165   -4.339    6.728
   0.0      0.0      0.0      0.0      0.0      0.0     20.661   -2.342    1.687   -2.130
   0.0      0.0      0.0      0.0      0.0      0.0      0.0     20.551   -2.515    3.749
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0     20.393   -0.606
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    -17.394

QT = ORTHOG MATRIX THAT REDUCES T
   0.715   -0.565    0.096    0.111    0.161   -0.331   -0.058    0.077    0.008   -0.058
   0.459    0.557   -0.546    0.132    0.162    0.064   -0.354   -0.035    0.081   -0.010
   0.369    0.360    0.565   -0.515    0.121    0.129    0.082   -0.325   -0.038    0.073
   0.198    0.290    0.373    0.625   -0.465    0.033    0.078    0.091   -0.333   -0.057
  -0.210    0.281    0.262    0.293    0.640   -0.388    0.131    0.088    0.143   -0.342
  -0.054   -0.197    0.297    0.266    0.355    0.652   -0.451    0.116    0.097    0.149
   0.110   -0.108   -0.216    0.323    0.206    0.323    0.641   -0.496    0.073    0.124
   0.111    0.065   -0.135   -0.191    0.280    0.216    0.372    0.629   -0.511    0.088
   0.173    0.036    0.035   -0.091   -0.241    0.312    0.257    0.358    0.611   -0.487
  -0.035   -0.149   -0.106   -0.096    0.033    0.212   -0.151   -0.295   -0.459   -0.767
READY
```

```
C  TEST PROGRAM FOR FTO2S, WHICH IS THE SAME AS FTO2 (Q-R DECOMP.
C  OF TOEP.MX.IN O(N**2) OPS) BUT HANDLES THE CASE WHERE THE
C  I/P MX. AND/OR ITS LDG.S/MATRIX IS SINGULAR. DOES NOT HANDLE
C  THE CASE WHERE THE I/P MATRIX HAS RANK.LT.N-1.
C
C  TEST SEQUENCE:
C  (1) GEN.TEST MATRIX 'A', REQUEST IF 'A' AND/OR ITS LDG.S/MATRIX
C      ARE TO BE MADE SINGULAR AND ADJ.ELEMENTS ACCORDINGLY
C  (2) FIND Q-R DECOMP.OF 'A' USING HOUSEHOLDER TRANSFORMS
C  (3)  ..   ..   ..   ..   ..   .. FTO2.
C
      REAL*8 T(19),A(10,10),R(10,10),QT(10,10),WKA(230)
      REAL*8 C,S,TEMP,EPS
      DATA T/2.2,3.7,-5.9,-1.1,-2.8,6.3,-4.9,8.0,
     1  6.2,18.6,-14.7,2.5,2.9,4.2,-8.6,-1.5,2.0,
     2  0.2,-1.5/

      EPS=10.**(-4)
      ID1=10
      N=10
      N2P3=2*N+3
      N2M1=2*N-1
      NM1=N-1

    1 T(N2M1-1)=0.2
      T(N2M1)=-1.5
      CALL TOEPMX(T,A,N,N2M1,ID1)
      WRITE(6,2)
    2 FORMAT(' IS T,T(N-1) SINGULAR?(1=YES,-1=EXIT)')
      READ(5,*) NSING,MSING
      IF(NSING.EQ.-1.OR.MSING.EQ.-1) STOP
      IF(MSING.EQ.1)CALL SINGSM(A,N,N,ID1,N-1,1,N-1,-1,1,WKA,N2P3,IER)
      A(N,2)=A(N-1,1)
      IF(NSING.EQ.1)CALL SINGSM(A,N,N,ID1,N,1,N,-1,1,WKA,N2P3,IER)
    5 T(2*N-2)=A(N-1,1)
      T(N2M1)=A(N,1)

      WRITE(6,10)
   10 FORMAT(/' INPUT TOEPLITZ MATRIX T')
      CALL PRNTMX(A,N,N,ID1)

C     REDUCE TOEPLITZ MATRIX BY HOUSEHOLDER TRANSFORMS
      CALL HOUQR(A,QT,N,ID1)
      DO 30 I=2,N
      IM1=I-1
      DO 30 J=1,IM1
   30 A(I,J)=0.
C  IF R(N-1,N-1)=0, APPLY ROT. TO ZERO R(N,N)
      TEMP=DSQRT(A(NM1,N)**2 + A(N,N)**2)
      IF(DABS(A(NM1,NM1))/TEMP.GE.EPS) GO TO 35
      C=A(NM1,N)/TEMP
      S=A(N,N)/TEMP
      A(NM1,N)=TEMP
      A(N,N)=0.
      DO 15 I=1,N
   15 CALL GIVENS(QT(NM1,I),QT(N,I),C,S)
```

```
   35 WRITE(6,40)
   40 FORMAT(/' REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS')
      CALL PRNTMX(A,N,N,ID1)
      WRITE(6,50)
   50 FORMAT(/' QT = PRODUCT OF HOUSEHOLDER TRANSFORMS')
      CALL PRNTMX(QT,N,N,ID1)

C  REDUCE TOEPLITZ MATRIX USING FTO2S
      CALL FTO2S(T,QT,R,N,N2M1,ID1)
      DO 52 I=2,N
      IM1=I-1
      DO 52 J=1,IM1
   52 R(I,J)=0.
C  IF R(N-1,N-1)=0, APPLY ROT. TO ZERO R(N,N)
      TEMP=DSQRT(R(NM1,N)**2 + R(N,N)**2)
      IF(DABS(R(NM1,NM1))/TEMP.GE.EPS) GO TO 56
      C=R(NM1,N)/TEMP
      S=R(N,N)/TEMP
      R(NM1,N)=TEMP
      R(N,N)=0.
      DO 54 I=1,N
   54 CALL GIVENS(QT(NM1,I),QT(N,I),C,S)

   56 WRITE(6,58)
   58 FORMAT(/' REDUCED MATRIX AS CALCULATED BY FTO2S')
      CALL PRNTMX(R,N,N,ID1)
      WRITE(6,60)
   60 FORMAT(/' QT = ORTHOG MATRIX THAT REDUCES T')
      CALL PRNTMX(QT,N,N,ID1)
      GO TO 1
      END



      SUBROUTINE FTO2S(T,Y,R,N,N2M1,ID1)
C
C  S/R TO FIND THE QR FACTORS OF A TOEPLITZ MATRIX USING GILL
C  ET AL'S RANK-1 UPDATE PROC., SHIFT-INVARIANCE AND A ROW-
C  RECURSION FOR Q-TRANSPOSE. HANDLES CASES WHERE I/P MATRIX
C  AND/OR ITS LDG PRINC.S/MATRIX IS SINGULAR, BUT NOT WHERE
C  I/P MATRIX HAS RANK.LT.N-1. PROCEDURE IS AN EXTENSION OF
C  ALG.7.4.2, AND NOTATION IS THE SAME.
C
C  PARAMETERS
C  T........VECTOR CONTAINING 1ST ROW AND COL OF TOEPLITZ MATRIX
C           TO BE TRIANGULARIZED. IF N IS THE ORDER OF THE MATRIX,
C           ORDER OF STORAGE IS (1,N),(1,N-1),...,(1,1),(2,1),...
C           ...,(N,1)
C  Y........ORTHOGONAL MATRIX SATISFYING YT=R, R UPPER-TRIANGULAR
C  R........CONTAINS UPPER-TRIANGULAR MATRIX PRODUCED BY THE
C           ORTHOGONALIZATION. NOTE THAT R(I,J),I.GT.J HAVE
C           NOT BEEN ZEROED
C  N........ORDER OF TOEPLITZ MATRIX AND R, AND LENGTH OF
C           VECTORS CPHIUP,...,Z.
C  N2M1.....2*N-1. T MUST BE AT LEAST THIS LENGTH
C  ID1......ROW-DIMENSION OF R
C
```

```
C  QUICK OUTLINE OF FTO2S
C  ----------------------
C  DEFINE
C  TM = T(SUPERSCRIPT N-1) = LEADING SUBMATRIX OF T
C  RM = R(SUPERSCRIPT N-1) = ORTHOGONAL TRIANGULAR FACTOR OF TM
C  YM = Y(SUPERSCRIPT N-1) = ORTHOGONAL MATRIX SUCH THAT YM*TM=RM
C  R-DOT = MATRIX WITH 1ST ROW = T(1,1..N), REST OF 1ST
C          COLUMN = YM * T(2..N,1),
C          TRAILING SUBMATRIX = LEADING SUBMATRIX OF T.
C  R-TILDE = R + Y(N.)*(T(N,1..N-1),0)
C
C  R-DOT MAY BE TRANSFORMED TO R USING GILL ET AL'S RANK-1
C  ORTHOGONAL UPDATE, WITH AN UPSWEEP AND A DOWNSWEEP OF
C  GIVENS TRANSFORMS (ANGLES PHI-UP AND PHI-DOWN). SIMILARLY,
C  R-TILDE MAY BE TRANSFORMED TO AN UPPER-TRIANGULAR MATRIX
C  WITH RM AS ITS LEADING SUBMATRIX (GIVENS ANGLES THETA-UP
C  AND THETA-DOWN)
C    SUPPOSE THAT RM(.J-1), R(.J-1), YM(J-1.) AND Y(J.) ARE
C  KNOWN. THEN R-DOT(.J) CAN BE CONSTRUCTED. PERFORM A GILL
C  UPSWEEP AND DOWNSWEEP ON R-DOT(.J), YIELDING R(.J).
C  Y(J.)  MAY BE OBTAINED FROM YM(J-1.) IN A SIMILAR MANNER,
C  RE-ARRANGING THE CALCULATIONS TO OPERATE ON ROWS, RATHER
C  THAN COLUMNS.
C    CALCULATE R-TILDE(.J) FROM THE DEFINITION, AND PERFORM
C  A GILL UPSWEEP AND DOWNSWEEP ON R-TILDE(.J), YIELDING
C  RM(.J). YM(J.) MAY BE OBTAINED FROM Y(J.) IN A SIMILAR
C  MANNER, RE-ARRANGING THE CALCULATIONS TO OPERATE ON ROWS,
C  RATHER THAN COLUMNS. THIS COMPLETES THE BASIC RECURSION.
C          NOTE THAT THE GILL SWEEPS IN THE RECURSION, WHICH
C  ACCOUNT FOR MOST OF THE COMPUTING LOAD, REQUIRE ONLY
C  O(N) OPERATIONS, SO THE WHOLE ALGORITHM REQUIRES
C  ONLY O(N**2) OPERATIONS. DETAILS OF THE CALCULATION
C  OF THE GILL ROTATIONS, WHICH PROCEEDS IN TANDEM WITH
C  THE BASIC RECURSION ARE GIVEN IN ALG.7.4.2.

C  NOTATION - WORKING STORAGE
C  --------------------------
C  CPHIUP)
C  SPHIUP)  OUTPUT COSINES AND SINES OF GIVENS ROTATIONS USED
C  CPHIDN)  IN FTO2S. NOTATION FOLLOWS ALG.7.4.2, WITH THE
C  SPHIDN)..FOLLOWING CONVENTION
C  CTHEUP)  1ST LETTER. C-COSINE, S-SINE
C  STHEUP)  NEXT 3 LETTERS. PHI-PHI, THE-THETA
C  CTHEDN)  LAST 2 LETTERS. UP-UP ARROW, DN-DOWN ARROW
C  STHEDN)
C  RM.......TEMP. VECTOR TO STORE COL. OF R(SUPERSCRIPT N-1)
C  RD1......O/P VECTOR CONTAINING 1ST COL OF R-DOT
C  RDJ......TEMP. VECTOR TO STORE COL.J OF R-DOT.
C  RT1......O/P VECTOR CONTAINING 1ST COL OF R-TILDE
C  RTJ......TEMP. VECTOR TO STORE COL.J OF R-TILDE
C  SIGD.....O/P VECTOR CONTAINING VALUES OF SIGMA-DOT
C  SIGT.....O/P VECTOR CONTAINING VALUES OF SIGMA-TILDE
C  Z........O/P VECTOR (= T-TRANSPOSE * (T(2,1),...,T(N,1))
C                                                TRANSPOSE )
C  YD.......TEMP.VECTOR CONTAINING Y-DOT(J.)
C  YD1......TEMP.VECTOR CONTAINING Y-DOT-1(J.), AND OVERWRITTEN
C                                BY Y-DOT-1(J+1.)
```

```
C  YD2......TEMP.VECTOR CONTAINING Y-DOT-2(J+1.)
C  YD3......TEMP.VECTOR CONTAINING Y-DOT-3(J.), AND OVERWRITTEN
C                                BY Y-DOT-3(J+1.)
C  Y1 TO Y3.SIMILAR TO YD1 TO YD3, BUT REFERRING TO Y.1 TO Y.3
C  YM.......TEMP.VECTOR CONTAINING YM(J.)

      REAL*8 T(N2M1),R(ID1,N),CPHIUP(100),SPHIUP(100),CPHIDN(100),
     1  CTHEUP(100),STHEUP(100),CTHEDN(100),STHEDN(100),RM(100),RD1(100)
     2  ,RT1(100),RTJ(100),SIGD(100),SIGT(100),SPHIDN(100),RDJ(100)
      REAL*8 Y(ID1,N),YD(100),YD1(100),YD2(100),YD3(100),Y1(100),Y2(100)
     1      ,YM(100),Y3(100)
      REAL*8 EPS,TEMP
      INTEGER PHIFLG,THEFLG

C  INITIALIZATION

      IF(N.LE.100) GO TO 5
      WRITE(6,2)
    2 FORMAT(' ** N.GT.100 - NO PROCESSING BY FTO2S')
      RETURN

    5 EPS=10.**(-4)
      M=N-1
      PHIFLG=0
      THEFLG=0

C  CALC R(1,1) AND RM(1,1) USING NORM-INVARIANCE
      RM(1)=0.
      DO 10 I=1,M
   10 RM(1)=RM(1)+T(I-1+N)**2
      R(1,1)=RM(1)+T(N2M1)*T(N2M1)
      R(1,1)=DSQRT(R(1,1))
      RM(1)=DSQRT(RM(1))

C  GET R-DOT(1,1),(1,2),(2,2) AND SIGMA-DOT(1),(2), WHERE SIGMA-DOT(I)
C  IS NORM(R-DOT(I..N,1))
      RD1(1)=T(N)
      RDJ(1)=T(N-1)
      RDJ(2)=RM(1)
      SIGD(1)=R(1,1)
      SIGD(2)=DSQRT(SIGD(1)**2 - RD1(1)**2)

C  OTHER INITLZN.FOR R-RECURSION - PHIDN(1),THEDN(1),R-TILDE(1,1), AND
C  SIGMA-TILDE(1),(2) WHERE SIGMA-TILDE(I)=NORM(R-TILDE(I..N,1))
      CPHIDN(1)=RD1(1)/SIGD(1)
      SPHIDN(1)=SIGD(2)/SIGD(1)
      Y(1,N)=T(N2M1)/R(1,1)
      RT1(1)=R(1,1)-Y(1,N)*T(N2M1)
      SIGT(1)=RM(1)
      SIGT(2)=DSQRT(SIGT(1)**2 - RT1(1)**2)
      CTHEDN(1)=RT1(1)/SIGT(1)
      STHEDN(1)=SIGT(2)/SIGT(1)

C  INITLZN.FOR Q-RECURSION - Y-DOT(1.),YM(1.),Y(1.),Y-DOT-1(2.),
C  Y-DOT-3(2.),Y.1(2.) AND Y.3(2.)
      YD(1)=1.
      DO 12 I=2,N
```

```
   12 YD(I)=0.
      DO 14 I=1,M
   14 YM(I)=T(I-1+N)/RM(1)
      YM(N)=0.
      DO 16 I=1,N
      Y(1,I)=T(I-1+N)/R(1,1)
      YD1(I)=(Y(1,I)-CPHIDN(1)*YD(I))/SPHIDN(1)
      YD3(I)=-SPHIDN(1)*YD(I) + CPHIDN(1)*YD1(I)
      Y1(I)=(YM(I) - CTHEDN(1)*Y(1,I))/STHEDN(1)
   16 Y3(I)=-STHEDN(1)*Y(1,I) + CTHEDN(1)*Y1(I)

C******MAIN LOOP - CALC R(.J),Y(J.),RM(.J) AND YM(J.)*******

      J=2
C
C  R-RECURSION PHASE I - CALC R(.J) FROM R-DOT(.J)
C
   30 RDJ(1)=T(1-J+N)
      DO 40 I=2,J
   40 RDJ(I)=RM(I-1)
C  RM CONTAINS RM(.J-1)
C
C  DO ONLY A PARTIAL SWEEP IF SOME OF THE PHI'S ARE ZERO
      IF(PHIFLG.EQ.0) GO TO 48
      RDJ(PHIFLG)=DSIGN(RDJ(PHIFLG),CPHIUP(PHIFLG))
      CALL RSWEEP(RDJ,R(1,J),CPHIUP,SPHIUP,CPHIDN,SPHIDN,PHIFLG-1,J)
      Y(J,1)=0.
      DO 45 I=2,N
   45 Y(J,I)=YM(I-1)
      IF(J.EQ.N) RETURN
      GO TO 120
C
C  CALC  R-DOT(J,1) = YM(J-1.)*T(2..N,1)
   48 RD1(J)=0.
      DO 50 I=1,M
   50 RD1(J)=RD1(J) + YM(I)*T(I+N)

C  IF J=N, CHECK THAT RD1(N) IS POSITIVE. IF NOT, RD(N.) WAS
C  REVERSED IN SIGN IN THE PREVIOUS STEP - SO RDJ(N) MUST BE
C  REPLACED BY -RDJ(N).
      IF(J.NE.N) GO TO 65
      IF(RD1(N).LT.0) RDJ(N)=-RDJ(N)
      GO TO 70

C  NEXT PHIUP
   65 TEMP=SIGD(J)**2-RD1(J)**2
      IF(TEMP.LT.0.) TEMP=0.
      SIGD(J+1)=DSQRT(TEMP)
      CPHIUP(J)=RD1(J)/SIGD(J)
      SPHIUP(J)=SIGD(J+1)/SIGD(J)
C
C  IF PHI(J)=0, USE ONLY PHI(1) TO PHI(J-1) IN THE SWEEP
      IF(DABS(SPHIUP(J)).GE.EPS) GO TO 68
      PHIFLG=J
      RDJ(J)=DSIGN(RDJ(J),CPHIUP(J))
      CALL RSWEEP(RDJ,R(1,J),CPHIUP,SPHIUP,CPHIDN,SPHIDN,PHIFLG-1,J)
      DO 66 I=1,N
```

```
   66 Y(J,I)=YD3(I)
      GO TO 120

C  APPLY PHIUP(J), THEN PHI(1..J-1) ON RDJ
   68 RDJ(J+1)=0.
      CALL GIVENS(RDJ(J),RDJ(J+1),CPHIUP(J),SPHIUP(J))
   70 CALL RSWEEP(RDJ,R(1,J),CPHIUP,SPHIUP,CPHIDN,SPHIDN,J-1,J)

C  CALC AND APPLY LAST PHIDN IN DOWN-SWEEP (N/A IF J=N)
      IF(J.NE.N) GO TO 110
      DO 105 I=1,N
  105 Y(N,I)=YD3(I)
      RETURN
  110 R(J,J)=DSQRT(RDJ(J)**2 + RDJ(J+1)**2)
      CPHIDN(J)=RDJ(J)/R(J,J)
      SPHIDN(J)=RDJ(J+1)/R(J,J)

C  Q-RECURSION PHASE I - CALC Y-DOT(J.),Y-DOT-1(J+1.),Y-DOT-2(J+1.),
C  Y(J.) AND Y-DOT-3(J+1.)
      YD(1)=0.
      DO 112 I=2,N
  112 YD(I)=YM(I-1)
      DO 115 I=1,N
      YD1(I)=(YD1(I)-CPHIUP(J)*YD(I))/SPHIUP(J)
      YD2(I)=-SPHIUP(J)*YD(I) + CPHIUP(J)*YD1(I)
      Y(J,I)=CPHIDN(J)*YD3(I) + SPHIDN(J)*YD2(I)
  115 YD3(I)=-SPHIDN(J)*YD3(I) + CPHIDN(J)*YD2(I)

C  R-RECURSION PHASE II - CALC RM(.J) FROM R-TILDE(.J)

C  CALC R-TILDE(1..J,J)
  120 DO 125 I=1,J
  125 RTJ(I)=R(I,J)-Y(I,N)*T(N-J+N)

C  IF SOME THETA'S ARE ZERO, ONLY APPLY THE NON-ZERO ONES
      IF(THEFLG.EQ.0) GO TO 127
      RTJ(THEFLG)=DSIGN(RTJ(THEFLG),CTHEUP(THEFLG))
      CALL RSWEEP(RTJ,RM,CTHEUP,STHEUP,CTHEDN,STHEDN,THEFLG-1,J)
      DO 126 I=1,N
  126 YM(I)=Y(J,I)
      GO TO 170

C  NEXT THEUP
  127 RT1(J)= -T(N2M1)*Y(J,N)
      TEMP=SIGT(J)**2 - RT1(J)**2
      IF(TEMP.LT.0.) TEMP=0.
      SIGT(J+1)=DSQRT(TEMP)
      CTHEUP(J)=RT1(J)/SIGT(J)
      STHEUP(J)=SIGT(J+1)/SIGT(J)

C  IF THETA=0, ONLY APPLY THETA(1..J-1) IN SWEEP
      IF(DABS(STHEUP(J)).GE.EPS) GO TO 129
      THEFLG=J
      RTJ(J)=DSIGN(RTJ(J),CTHEUP(J))
      CALL RSWEEP(RTJ,RM,CTHEUP,STHEUP,CTHEDN,STHEDN,THEFLG-1,J)
  947 DO 128 I=1,N
  128 YM(I)=Y3(I)
```

```
      GO TO 170

C  APPLY THEUP(J), THEN THETA(1..J-1) TO R-TILDE(.J)
  129 RTJ(J+1)=SIGT(J+1)*T(N-J+N)/T(N2M1)
      CALL GIVENS(RTJ(J),RTJ(J+1),CTHEUP(J),STHEUP(J))
      CALL RSWEEP(RTJ,RM,CTHEUP,STHEUP,CTHEDN,STHEDN,J-1,J)
C
C  CALC AND APPLY LAST THEDN
  975 RM(J)=DSQRT(RTJ(J)**2 + RTJ(J+1)**2)
      CTHEDN(J)=RTJ(J)/RM(J)
      STHEDN(J)=RTJ(J+1)/RM(J)

C  CHECK THAT YM(J,N) IS 'NUMERICALLY ZERO'. IF NOT, RM HAS RANK
C  J-1 OR LESS, AND RM(J,J)=0. THEDN(J) MUST BE RECALCULATED TO
C  TO MAKE YM(J,N) ZERO. (RM(J,J) WILL REMAIN ZERO).
      Y2(N)=-STHEUP(J)*Y(J,N) + CTHEUP(J)*
     1             (Y1(N)-CTHEUP(J)*Y(J,N))/STHEUP(J)
      YM(N)=CTHEDN(J)*Y3(N) + STHEDN(J)*Y2(N)
      IF(DABS(YM(N)).LE.EPS) GO TO 155
      RM(J)=0.
      TEMP=DSQRT(Y3(N)**2 + Y2(N)**2)
      CTHEDN(J)=Y2(N)/TEMP
      STHEDN(J)=-Y3(N)/TEMP

C  Q-RECURSION PHASE II - CALC Y.1(J+1.),Y.2(J+1.),YM(J.) AND Y.3(J+1.)
  155 DO 160 I=1,N
      Y1(I)=(Y1(I) - CTHEUP(J)*Y(J,I))/STHEUP(J)
      Y2(I)=-STHEUP(J)*Y(J,I) + CTHEUP(J)*Y1(I)
      YM(I)=CTHEDN(J)*Y3(I) + STHEDN(J)*Y2(I)
  160 Y3(I)=-STHEDN(J)*Y3(I) + CTHEDN(J)*Y2(I)

  170 J=J+1
      GO TO 30
C
C***************END OF MAIN LOOP************************

      END



       SUBROUTINE RSWEEP(P,R,CUP,SUP,CDN,SDN,NROT,J)
C
C  S/R TO APPLY A GILL-TYPE UP/DOWN SWEEP TO A VECTOR
C
C  P.........I/P VECTOR
C  R.........O/P VECTOR
C  CUP,SUP..COS'S AND SIN'S OF UPSWEEP ANGLES
C  CDN,SDN..COS'S AND SIN'S OF DOWNSWEEP ANGLES
C  NROT.....NO.OF DOWNWARD ROTATIONS TO APPLY. (NO.OF
C           UPWARD ROTATIONS IS ONE LESS THAN NROT)
C  J........LENGTH OF P AND R. IF J.GT.NROT+1,
C           R(NROT+2..J)=P(NROT+2..J)
C
      REAL*8 P(J),R(J),CUP(J),SUP(J),CDN(J),SDN(J)
C
C  UPSWEEP
      IF(NROT.EQ.1) GO TO 15
```

```
      I=NROT
      DO 10 II=2,NROT
      CALL GIVENS(P(I),P(I+1),CUP(I),SUP(I))
   10 I=I-1
C
C  DOWNSWEEP
   15 DO 20 I=1,NROT
   20 CALL GIVENS(P(I),P(I+1),CDN(I),SDN(I))

C  OUTPUT
      DO 30 I=1,J
   30 R(I)=P(I)
      RETURN
      END
```

```
IS T,T(N-1) SINGULAR?(1=YES,-1=EXIT)
?
0,1

INPUT TOEPLITZ MATRIX T
   18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100    -5.900     3.700     2.200
  -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100    -5.900     3.700
    2.500   -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100    -5.900
    2.900     2.500   -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100
    4.200     2.900     2.500   -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800
   -8.600     4.200     2.900     2.500   -14.700    18.600     6.200     8.000    -4.900     6.300
   -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600     6.200     8.000    -4.900
    2.000    -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600     6.200     8.000
  162.992     2.000    -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600     6.200
   -1.500   162.992     2.000    -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600

REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS
 -185.055     0.728     1.357     9.803    -6.854    -1.368    -2.111    15.410   -19.796    -5.505
    0.0    -165.163    -1.047     1.565     9.543    -6.672    -1.373    -1.846    15.151   -19.756
    0.0       0.0     -27.822     7.293    -2.261     7.439   -14.702    11.411     0.684     4.186
    0.0       0.0       0.0     -25.402     5.919    -1.737     2.944    -7.847     6.000    -0.441
    0.0       0.0       0.0       0.0     -25.617     6.568    -0.854     1.646    -5.604     6.366
    0.0       0.0       0.0       0.0       0.0     -24.909     4.111    -0.594     4.431    -4.503
    0.0       0.0       0.0       0.0       0.0       0.0     -22.430    -0.382    -2.194     5.312
    0.0       0.0       0.0       0.0       0.0       0.0       0.0     -18.505    -7.463    -6.293
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      4.038     2.506
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      0.0       1.126

QT = PRODUCT OF HOUSEHOLDER TRANSFORMS
   -0.113     0.089    -0.015    -0.018    -0.025     0.052     0.009    -0.012    -0.988     0.009
   -0.038    -0.112     0.089    -0.015    -0.018    -0.025     0.052     0.009    -0.016    -0.987
   -0.292    -0.214    -0.673     0.528    -0.090    -0.101    -0.152     0.388     0.006    -0.034
    0.063    -0.349    -0.438    -0.588     0.542    -0.109    -0.151    -0.081    -0.042    -0.008
   -0.190     0.064    -0.317    -0.426    -0.593     0.534    -0.102    -0.153     0.084    -0.033
   -0.013    -0.251    -0.080    -0.230    -0.464    -0.624     0.514    -0.045    -0.031     0.077
    0.866     0.170     0.096    -0.228    -0.285    -0.363    -0.655     0.441    -0.036    -0.033
    0.000     0.171    -0.127     0.196    -0.072    -0.327    -0.375    -0.813     0.005    -0.052
    0.358     0.723    -0.451    -0.039     0.110    -0.060     0.327     0.085     0.027    -0.119
    0.814    -0.405    -0.113     0.221    -0.179     0.238     0.016    -0.102    -0.113     0.000

REDUCED MATRIX AS CALCULATED BY FTO2S
  185.055    -0.728    -1.357    -9.803     6.854     1.368     2.111   -15.410    19.796     5.505
    0.0     165.163     1.047    -1.565    -9.543     6.672     1.373     1.846   -15.151    19.756
    0.0       0.0      27.822    -7.293     2.261    -7.439    14.702   -11.411    -0.684    -4.186
    0.0       0.0       0.0      25.402    -5.919     1.737    -2.944     7.847    -6.000     0.441
    0.0       0.0       0.0       0.0      25.617    -6.568     0.854    -1.646     5.604    -6.366
    0.0       0.0       0.0       0.0       0.0      24.909    -4.111     0.594    -4.431     4.503
    0.0       0.0       0.0       0.0       0.0       0.0      22.430     0.382     2.194    -5.312
    0.0       0.0       0.0       0.0       0.0       0.0       0.0      18.505     7.463     6.293
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      4.038     2.506
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      0.0      -1.126

QT = ORTHOG MATRIX THAT REDUCES T
    0.113    -0.089     0.015     0.018     0.025    -0.052    -0.009     0.012     0.988    -0.009
    0.038     0.112    -0.089     0.015     0.018     0.025    -0.052    -0.009     0.016     0.987
    0.292     0.214     0.673    -0.528     0.090     0.101     0.152    -0.388    -0.006     0.034
   -0.063     0.349     0.438     0.588    -0.542     0.109     0.151     0.081     0.042     0.008
    0.190    -0.064     0.317     0.426     0.593    -0.534     0.102     0.153    -0.084     0.033
    0.013     0.251     0.080     0.230     0.464     0.624    -0.514     0.045     0.031    -0.077
   -0.866    -0.170    -0.096     0.228     0.285     0.363     0.655    -0.441     0.036     0.033
   -0.000    -0.171     0.127    -0.196     0.072     0.327     0.375     0.813    -0.005     0.052
    0.358     0.723    -0.451    -0.039     0.110    -0.060     0.327     0.085     0.027    -0.119
   -0.814     0.405     0.113    -0.221     0.179    -0.238    -0.016     0.102     0.113    -0.000
```

```
IS T,T(N-1) SINGULAR?(1=YES,-1=EXIT)
?
1,0

INPUT TOEPLITZ MATRIX T
   18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100    -5.900     3.700     2.200
  -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100    -5.900     3.700
    2.500   -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100    -5.900
    2.900     2.500   -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100
    4.200     2.900     2.500   -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800
   -8.600     4.200     2.900     2.500   -14.700    18.600     6.200     8.000    -4.900     6.300
   -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600     6.200     8.000    -4.900
    8.000    -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600     6.200     8.000
    0.200     2.000    -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600     6.200
  368.581     0.200     2.000    -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600

REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS
 -369.496     0.345    -2.058     2.092     7.402    -3.540    -2.746    -2.097    14.075   -18.357
    0.0     -26.701     5.653     0.293     6.880   -15.673     9.210     3.376     4.818    -8.890
    0.0       0.0     -27.218     6.929    -0.733     4.428   -12.856    11.685     2.205     3.335
    0.0       0.0       0.0     -27.286     8.485    -2.047     4.131   -12.943    13.336     0.925
    0.0       0.0       0.0       0.0     -25.686     3.698     0.530     4.943    -8.133     5.228
    0.0       0.0       0.0       0.0       0.0     -21.741     0.571    -2.959     2.464    -4.079
    0.0       0.0       0.0       0.0       0.0       0.0     -21.831     2.301    -3.263     4.449
    0.0       0.0       0.0       0.0       0.0       0.0       0.0     -20.493     1.470    -2.284
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0     -17.224    -8.804
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.000

QT = PRODUCT OF HOUSEHOLDER TRANSFORMS
   -0.050     0.040    -0.007    -0.008    -0.011     0.023     0.004    -0.005    -0.001    -0.998
   -0.233    -0.696     0.550    -0.094    -0.109    -0.157     0.322     0.056    -0.075    -0.020
   -0.338    -0.375    -0.569     0.521    -0.114    -0.141    -0.088     0.328     0.040    -0.002
    0.087    -0.393    -0.366    -0.551     0.508    -0.127    -0.125    -0.070     0.324    -0.022
   -0.284    -0.103    -0.271    -0.466    -0.586     0.499    -0.049    -0.132    -0.078     0.035
    0.179     0.149    -0.298    -0.220    -0.375    -0.678     0.429    -0.104    -0.115    -0.009
    0.172    -0.022     0.196    -0.242    -0.272    -0.300    -0.679     0.485    -0.113    -0.018
   -0.068    -0.080     0.135     0.215    -0.281    -0.234    -0.371    -0.629     0.508    -0.003
   -0.176    -0.104    -0.084     0.023     0.224    -0.153    -0.298    -0.459    -0.760     0.001
   -0.802     0.410     0.119    -0.212     0.175    -0.249    -0.006     0.120     0.141     0.049

REDUCED MATRIX AS CALCULATED BY FTO2S
  369.495    -0.345     2.058    -2.092    -7.402     3.540     2.746     2.097   -14.075    18.357
    0.0      26.701    -5.653    -0.293    -6.880    15.673    -9.210    -3.376    -4.818     8.890
    0.0       0.0      27.218    -6.929     0.733    -4.428    12.856   -11.685    -2.205    -3.335
    0.0       0.0       0.0      27.286    -8.485     2.047    -4.131    12.943   -13.336    -0.925
    0.0       0.0       0.0       0.0      25.686    -3.698    -0.530    -4.943     8.133    -5.228
    0.0       0.0       0.0       0.0       0.0      21.741    -0.571     2.959    -2.464     4.079
    0.0       0.0       0.0       0.0       0.0       0.0      21.831    -2.301     3.263    -4.449
    0.0       0.0       0.0       0.0       0.0       0.0       0.0      20.493    -1.470     2.284
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      17.224     8.804
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      -0.000

QT = ORTHOG MATRIX THAT REDUCES T
    0.050    -0.040     0.007     0.008     0.011    -0.023    -0.004     0.005     0.001     0.998
    0.233     0.696    -0.550     0.094     0.109     0.157    -0.322    -0.056     0.075     0.020
    0.338     0.375     0.569    -0.521     0.114     0.141     0.088    -0.328    -0.040     0.002
   -0.087     0.393     0.366     0.551    -0.508     0.127     0.125     0.070    -0.324     0.022
    0.284     0.103     0.271     0.466     0.586    -0.499     0.049     0.132     0.078    -0.035
   -0.179    -0.149     0.298     0.220     0.375     0.678    -0.429     0.104     0.115     0.009
   -0.172     0.022    -0.196     0.242     0.272     0.300     0.679    -0.485     0.113     0.018
    0.068     0.080    -0.135    -0.215     0.281     0.234     0.371     0.629    -0.508     0.003
    0.176     0.104     0.084    -0.023    -0.224     0.153     0.298     0.459     0.760    -0.001
   -0.802     0.410     0.119    -0.212     0.175    -0.249    -0.006     0.120     0.141     0.049
```

```
IS T,T(N-1) SINGULAR?(1=YES,-1=EXIT)
?
1,1

INPUT TOEPLITZ MATRIX T
   18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100    -5.900     3.700     2.200
  -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100    -5.900     3.700
    2.500   -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100    -5.900
    2.900     2.500   -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800    -1.100
    4.200     2.900     2.500   -14.700    18.600     6.200     8.000    -4.900     6.300    -2.800
   -8.600     4.200     2.900     2.500   -14.700    18.600     6.200     8.000    -4.900     6.300
   -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600     6.200     8.000    -4.900
    8.000    -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600     6.200     8.000
  182.992     2.000    -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600     6.200
 -846.134   182.992     2.000    -1.500    -8.600     4.200     2.900     2.500   -14.700    18.600

REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS
 -296.349   134.954     2.407     4.221   -10.917     2.705     1.218    10.647   -23.160    12.288
    0.0     -95.219     1.584     8.622     1.134    -7.728    -0.639    11.769    -6.393   -16.810
    0.0       0.0     -27.725     7.757    -2.456     7.073   -14.633    12.223    -0.148     3.830
    0.0       0.0       0.0     -25.391     6.449    -2.949     3.583    -7.463     6.448    -0.796
    0.0       0.0       0.0       0.0     -25.814     6.778    -1.119     2.194    -6.596     6.129
    0.0       0.0       0.0       0.0       0.0     -24.426     3.973    -2.218     6.080    -3.729
    0.0       0.0       0.0       0.0       0.0       0.0     -22.495     0.210    -3.028     4.994
    0.0       0.0       0.0       0.0       0.0       0.0       0.0     -17.651    -8.788    -6.908
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      0.000     1.216
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      0.0       0.0

QT * PRODUCT OF HOUSEHOLDER TRANSFORMS
   -0.063     0.050    -0.008    -0.010    -0.014     0.029     0.005    -0.007    -0.550     0.831
   -0.154    -0.125     0.142    -0.040    -0.051    -0.003     0.097     0.006    -0.801    -0.535
   -0.303    -0.226    -0.663     0.527    -0.094    -0.102    -0.145     0.310    -0.039    -0.031
    0.038    -0.418    -0.400    -0.587     0.531    -0.126    -0.125    -0.070    -0.037     0.006
   -0.186     0.080    -0.337    -0.435    -0.575     0.535    -0.112    -0.156     0.029    -0.037
    0.013    -0.206    -0.083    -0.213    -0.490    -0.623     0.513    -0.050     0.075     0.069
    0.264     0.171     0.085    -0.234    -0.267    -0.364    -0.658     0.439    -0.087    -0.034
   -0.054     0.067    -0.072     0.194    -0.067    -0.315    -0.417    -0.817    -0.051    -0.035
    0.878    -0.106    -0.287     0.192    -0.104     0.184     0.114    -0.104    -0.164    -0.046
    0.000    -0.814     0.405     0.113    -0.221     0.179    -0.238    -0.016     0.102     0.113

REDUCED MATRIX AS CALCULATED BY FTQ2S
  296.349  -134.954    -2.407    -4.221    10.917    -2.705    -1.218   -10.647    23.160   -12.288
    0.0      95.219    -1.584    -8.622    -1.134     7.728     0.639   -11.769     6.393    16.810
    0.0       0.0      27.725    -7.757     2.456    -7.073    14.633   -12.223     0.148    -3.830
    0.0       0.0       0.0      25.391    -6.449     2.949    -3.583     7.463    -6.448     0.796
    0.0       0.0       0.0       0.0      25.814    -6.778     1.119    -2.194     6.596    -6.129
    0.0       0.0       0.0       0.0       0.0      24.426    -3.973     2.218    -6.080     3.729
    0.0       0.0       0.0       0.0       0.0       0.0      22.495    -0.210     3.028    -4.994
    0.0       0.0       0.0       0.0       0.0       0.0       0.0      17.651     8.788     6.908
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      -0.000     1.216
    0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0      0.0       0.0

QT * ORTHOG MATRIX THAT REDUCES T
    0.063    -0.050     0.008     0.010     0.014    -0.029    -0.005     0.007     0.550    -0.831
    0.154     0.125    -0.142     0.040     0.051    -0.003     0.097     0.006     0.801     0.535
    0.303     0.226     0.663    -0.527     0.094     0.102     0.145    -0.310     0.039     0.031
   -0.038     0.418     0.400     0.587    -0.531     0.126     0.125     0.070     0.037    -0.006
    0.186    -0.080     0.337     0.435     0.575    -0.535     0.112     0.156    -0.029     0.037
   -0.013     0.206     0.083     0.213     0.490     0.623    -0.513     0.050    -0.075    -0.069
   -0.264    -0.171    -0.085     0.234     0.267     0.364     0.658    -0.439     0.087     0.034
    0.054    -0.067     0.072    -0.194     0.067     0.315     0.417     0.817     0.051     0.035
    0.878    -0.106    -0.287     0.192    -0.104     0.184     0.114    -0.104    -0.164    -0.046
    0.0      -0.814    -0.405    -0.113     0.221    -0.179     0.238     0.016    -0.102    -0.113
```

```
C
C
C
C  TEST PROGRAM FOR FTO3 IS THE SAME AS THAT FOR FTO1,
C  EXCEPT THAT FTO3 IS CALLED INSTEAD OF FTO1.
C
C



      SUBROUTINE FTO3(T,R,CPHIUP,SPHIUP,CPHIDN,SPHIDN,SECTHE,
     1  TANTHE,RM,RD1,RDJ,SIGD,Z,N,N2M1,ID1)
C
C  S/R TO TRIANGULARIZE A TOEPLITZ MATRIX USING GILL ET AL'S
C  ORTHOG.UPDATE ALG.,INVERSE TRANSFORMS, AND SHIFT-INVARIANCE.
C  FOLLOWS ALG.8.2.2, AND THE VARIABLE NOTATION IS THE SAME.
C
C  PARAMETERS
C  T........VECTOR CONTAINING 1ST ROW AND COL OF TOEPLITZ MATRIX
C           TO BE TRIANGULARIZED. IF N IS THE ORDER OF THE MATRIX,
C           ORDER OF STORAGE IS (1,N),(1,N-1),...,(1,1),(2,1),...
C           ...,(N,1)
C  R........CONTAINS UPPER-TRIANGULAR MATRIX PRODUCED BY THE
C           ORTHOGONALIZATION. NOTE THAT R(I,J),I.GT.J HAVE
C           NOT BEEN ZEROED
C  CPHIUP)  OUTPUT COSINES,SINES (1ST LETTER 'C','S') OF THE
C  SPHIUP)  GIVENS ROTATIONS('PHIUP':PHI-UPSWEEP; 'PHIDN':
C  CPHIDN)  PHI-DOWNSWEEP) USED IN PHASE I OF FTO3.
C  SPHIDN)  NOTATION FOLLOWS ALG.8.2.2.
C
C  SECTHE)  O/P SEC(THETA),TAN(THETA), WHERE THE THETAS
C  TANTHE)  ARE THE ANGLES OF THE INVERSE TRANSFORMS USED
C        )  IN PHASE II OF FTO3. NOT'N FOLLOWS ALG.8.2.2.
C
C  RM.......TEMP. VECTOR TO STORE COL. OF R(SUPERSCRIPT N-1)
C  RD1......O/P VECTOR CONTAINING 1ST COL OF R-DOT
C  RDJ......TEMP. VECTOR TO STORE COL.J OF R-DOT.
C  SIGD.....O/P VECTOR CONTAINING VALUES OF SIGMA-DOT
C  Z........O/P VECTOR (= T-TRANSPOSE * (T(2,1),...,T(N,1))
C                                                  TRANSPOSE )
C  N........ORDER OF TOEPLITZ MATRIX AND R, AND LENGTH OF
C           VECTORS CPHIUP,...,Z.
C  N2M1.....2*N-1. T MUST BE AT LEAST THIS LENGTH
C  ID1......ROW-DIMENSION OF R
C
C  TEMPORARY VARIABLE:
C  RTNJ-----CURRENT VALUE OF R-TILDE(N,J)

C  GENERAL REMARKS ON CALLING FTO3. THE ONLY INPUTS REQUIRED
C  ARE T,N,N2M1 AND ID1. THE MAIN OUTPUT IS IN R. THE OTHER
C  11 VECTORS ARE BY-PRODUCTS AND TEMPORARIES OF FTO3, AND
C  NORMALLY THE USER ONLY ALLOCATES SPACE FOR THESE VECTORS
C  WITHOUT DOING ANYTHING MORE WITH THEM. HOWEVER, THE GIVENS
C  TRANSFORM VECTORS CPHIUP,...,TANTHE CAN BE USED TO GENERATE
C  THE Q-MATRIX EXPLICITLY USING ALG.8.3.1.
C
C  QUICK OUTLINE OF FTO3
```

```
C  --------------------
C  DEFINE
C  RM = R(SUPERSCRIPT N-1) = ORTHOGONAL TRIANGULAR FACTOR
C           OF LEADING SUBMATRIX OF T.
C  QM = Q(SUPERSCRIPT N-1) = OTHOGONAL MATRIX SUCH THAT
C           LEADING SUBMATRIX OF T = QM*RM
C  R-DOT = MATRIX WITH 1ST ROW = T(1,1..N), REST OF 1ST
C           COLUMN = QM-TRANSPOSE * T(2..N,1),
C           TRAILING SUBMATRIX = LEADING SUBMATRIX OF T.
C
C  R-DOT MAY BE TRANSFORMED TO R USING GILL ET AL'S RANK-1
C  ORTHOGONAL UPDATE, WITH AN UPSWEEP AND A DOWNSWEEP OF
C  GIVENS TRANSFORMS (ANGLES PHI-UP AND PHI-DOWN). THEN,
C  (R(1..N-1,.))  =  R-TILDE MAY BE TRANSFORMED TO AN
C  ( T(N,.)    )
C  UPPER-TRIANGULAR MATRIX WITH RM AS ITS LEADING SUBMATRIX
C  USING INVERSE TRANSFORMS.
C       SUPPOSE THAT RM(.J-1)  AND  R(.J-1) ARE KNOWN.
C  THEN R-DOT(.J) CAN BE CONSTRUCTED. PERFORM A GILL UPSWEEP
C  AND DOWNSWEEP ON R-DOT(.J), YIELDING R(.J).
C  PERFORM J INVERSE TRANSFORMS ON (R(1..N-1,J)) ,YIELDING
C                                  (  T(N,.)   )
C  RM(.J). THIS COMPLETES THE BASIC RECURSION. NOTE THAT
C  THE GILL SWEEPS AND THE INVERSE TRANSFORMS REQUIRE
C  ONLY O(N) OPERATIONS, SO THE WHOLE ALGORITHM REQUIRES
C  ONLY O(N**2) OPERATIONS. DETAILS OF THE CALCULATION
C  OF THE GILL ROTATIONS AND INVERSE TRANSFORM ANGLES,
C  WHICH PROCEEDS IN TANDEM WITH THE BASIC RECURSION,
C  ARE GIVEN IN ALG.8.2.2.

      REAL*8 T(N2M1),R(ID1,N),CPHIUP(N),SPHIUP(N),CPHIDN(N),
     1  SPHIDN(N),SECTHE(N),TANTHE(N),RM(N),RD1(N),RDJ(N),
     2  RTNJ,SIGD(N),Z(N)
      REAL*8 EPS

C  INITIALIZATION

      M=N-1
      EPS=10.**(-5)

C  CALC R(1,1) AND RM(1,1) USING NORM-INVARIANCE
      RM(1)=0.
      DO 10 I=1,M
   10 RM(1)=RM(1)+T(I-1+N)**2
      R(1,1)=RM(1)+T(N2M1)*T(N2M1)
      R(1,1)=DSQRT(R(1,1))
      RM(1)=DSQRT(RM(1))

C  GET R-DOT(1,1),(1,2),(2,2) AND SIGMA-DOT(1),(2), WHERE SIGMA-DOT(I)
C  IS NORM(R-DOT(I..N,1))
      RD1(1)=T(N)
      RDJ(1)=T(N-1)
      RDJ(2)=RM(1)
      SIGD(1)=R(1,1)
      SIGD(2)=DSQRT(SIGD(1)**2 - RD1(1)**2)

C  OTHER INITIALIZATION - PHIDN(1),THETA(1)
```

```
      CPHIDN(1)=RD1(1)/SIGD(1)
      SPHIDN(1)=SIGD(2)/SIGD(1)
      SECTHE(1)=R(1,1)/RM(1)
      TANTHE(1)=T(N2M1)/RM(1)

C  Z=TRANSPOSE OF PRINC SUBMX OF T * T(2..N,1)
      DO 28 I=1,M
      Z(I)=0.
      DO 28 J=1,M
   28 Z(I)=Z(I)+T(J-I+N)*T(J+N)

C********MAIN LOOP - CALC R(.J) AND RM(.J)*****************

      J=2
C
C  PHASE I - CALC R(.J) FROM R-DOT(.J)
C
   30 RDJ(1)=T(1-J+N)
      DO 40 I=2,J
   40 RDJ(I)=RM(I-1)
C  RM CONTAINS RM(.J-1)
C
C  BACK-SUBST FOR R-DOT(J,1) IN EQN. RM(TRANS)*R-DOT(.1)=Z
      RD1(J)=Z(J-1)
      IF(J.EQ.2) GO TO 60
      JM2=J-2
      DO 50 I=1,JM2
   50 RD1(J)=RD1(J)-RM(I)*RD1(I+1)
   60 RD1(J)=RD1(J)/RM(J-1)

C  IF J=N, CHECK THAT RD1(N) IS POSITIVE. IF NOT, RD(N.) WAS
C  REVERSED IN SIGN IN THE PREVIOUS STEP - SO RDJ(N) MUST BE
C  REPLACED BY -RDJ(N).
      IF(J.NE.N) GO TO 65
      IF(RD1(N).LT.0) RDJ(N)=-RDJ(N)
      GO TO 70

C  NEXT PHIUP
   65 SIGD(J+1)=DSQRT(SIGD(J)**2-RD1(J)**2)
      CPHIUP(J)=RD1(J)/SIGD(J)
      SPHIUP(J)=SIGD(J+1)/SIGD(J)
C  SINGULARITY CHECK
      IF(DABS(SPHIUP(J)).GE.EPS) GO TO 68
      WRITE(6,66)
   66 FORMAT(' I/P MATRIX NUMERICALLY SINGULAR - ERROR EXIT')
      RETURN

C  UP-SWEEP ON R-DOT(.J)
   68 RDJ(J+1)=0.
      CALL GIVENS(RDJ(J),RDJ(J+1),CPHIUP(J),SPHIUP(J))
   70 IF(J.EQ.2) GO TO 90
      JM1=J-1
      I=JM1
      DO 80 II=2,JM1
      CALL GIVENS(RDJ(I),RDJ(I+1),CPHIUP(I),SPHIUP(I))
   80 I=I-1
```

```
C  DOWN-SWEEP ON R-DOT(.J), EXCLUDING LAST PHIDN
   90 DO 100 I=1,JM1
      CALL GIVENS(RDJ(I),RDJ(I+1),CPHIDN(I),SPHIDN(I))
  100 R(I,J)=RDJ(I)

C  CALC AND APPLY LAST PHIDN IN DOWN-SWEEP (N/A IF J=N)
      IF(J.NE.N) GO TO 110
      R(N,N)=RDJ(N)
      RETURN
  110 R(J,J)=DSQRT(RDJ(J)**2 + RDJ(J+1)**2)
      CPHIDN(J)=RDJ(J)/R(J,J)
      SPHIDN(J)=RDJ(J+1)/R(J,J)

C  PHASE II - CALC RM(.J) FROM R-TILDE(.J)

C  INITIALIZE R-TILDE(N,J)
      RTNJ=T(2*N-J)

C  FIRST J-1 INVERSE TRANSFORMS
      JM1=J-1
         DO 120 I=1,JM1
         RM(I)=R(I,J)
  120    CALL INVXFM(RM(I),RTNJ,SECTHE(I),TANTHE(I))

C  CALC AND APPLY LAST THETA
      RM(J)=DSQRT(R(J,J)**2 - RTNJ**2)
C  SINGULARITY CHECK
      IF(RM(J)/R(J,J).GE.EPS) GO TO 140
      WRITE(6,130)
  130 FORMAT(' LDG.SUBMX.NUMERICALLY SINGULAR - ERROR EXIT',
     1  /' REMEDY - AUGMENT I/P MATRIX AND RERUN')
      RETURN

  140 SECTHE(J)=R(J,J)/RM(J)
      TANTHE(J)=RTNJ/RM(J)

      J=J+1
      GO TO 30
C
C***************END OF MAIN LOOP************************

      END


      SUBROUTINE INVXFM(X1,X2,SECA,TANA)

C  S/R TO APPLY AN INVERSE TRANSFORM
C
C  OPERATION:
C
C    (X1)      ( SECA  -TANA) (X1)
C    (  )  =   (            ) (  )
C    (X2)      (-TANA   SECA) (X2)
C

      REAL*8 X1,X2,SECA,TANA,T
```

```
T=SECA*X1 - TANA*X2
X2= -TANA*X1 + SECA*X2
X1=T
RETURN
END
```

```
INPUT TOEPLITZ MATRIX T
   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900    3.700    2.200
  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900    3.700
    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900
    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100
    4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800
   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300
   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900
    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000
    0.200    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200
   -1.500    0.200    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600

REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS
  -28.016    7.747   -0.780    8.378  -17.201    9.468    2.254    5.784   -9.204    3.869
    0.0    -25.554    5.698    2.817    1.874  -13.458   10.343    5.309    2.054   -7.868
    0.0      0.0    -27.275    7.360   -0.716    4.318  -12.849   11.746    2.957    2.031
    0.0      0.0      0.0    -25.782    2.909   -0.052    6.139  -10.894   10.403    2.651
    0.0      0.0      0.0      0.0    -22.936    2.543   -2.887    4.710  -11.662   10.939
    0.0      0.0      0.0      0.0      0.0    -21.736    1.355   -3.165    4.339   -6.728
    0.0      0.0      0.0      0.0      0.0      0.0    -20.661    2.342   -1.687    2.130
    0.0      0.0      0.0      0.0      0.0      0.0      0.0    -20.551    2.515   -3.749
    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    -20.393    0.606
    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    17.394

REDUCED MATRIX AS CALCULATED BY FT03
   28.016   -7.747    0.780   -8.378   17.201   -9.468   -2.254   -5.784    9.204   -3.869
    0.0     25.554   -5.698   -2.817   -1.874   13.458  -10.343   -5.309   -2.054    7.868
    0.0      0.0     27.275   -7.360    0.716   -4.318   12.849  -11.746   -2.957   -2.031
    0.0      0.0      0.0     25.782   -2.909    0.052   -6.139   10.894  -10.403   -2.651
    0.0      0.0      0.0      0.0     22.936   -2.543    2.887   -4.710   11.662  -10.939
    0.0      0.0      0.0      0.0      0.0     21.736   -1.355    3.165   -4.339    6.728
    0.0      0.0      0.0      0.0      0.0      0.0     20.661   -2.342    1.687   -2.130
    0.0      0.0      0.0      0.0      0.0      0.0      0.0     20.551   -2.515    3.749
    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0     20.393   -0.606
    0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0   -17.394
READY
```

```
C
C
C
C  TEST PROGRAM FOR FTO4 - SAME AS TEST PROGRAM FOR FTO2
C  EXCEPT THAT FTO4 IS CALLED INSTEAD OF FTO2.
C
C
C
C
C
C
C
      SUBROUTINE FTO4(T,Y,R,N,N2M1,ID1)
C
C  S/R TO FIND THE Q-R DECOMPOSITION OF A TOEPLITZ MATRIX USING
C  GILL ET-AL'S ORTHOGONAL UPDATE ALG.,INVERSE TRANSFORMS, AND
C  SHIFT-INVARIANCE TO GET AN O(N) RECURSION ON THE COLUMNS
C  OF R AND ROWS OF Q-TRANSPOSE. TOTAL COMPLEXITY IS THEREFORE
C  O(N**2). FOLLOWS ALG.8.3.2, WITH THE SAME NOTATION.
C
C  PARAMETERS
C  T........VECTOR CONTAINING 1ST ROW AND COL OF TOEPLITZ MATRIX
C           TO BE TRIANGULARIZED. IF N IS THE ORDER OF THE MATRIX,
C           ORDER OF STORAGE IS (1,N),(1,N-1),...,(1,1),(2,1),...
C           ...,(N,1)
C  Y........ORTHOGONAL MATRIX SATISFYING YT=R, R UPPER-TRIANGULAR
C  R........CONTAINS UPPER-TRIANGULAR MATRIX PRODUCED BY THE
C           ORTHOGONALIZATION. NOTE THAT R(I,J),I.GT.J HAVE
C           NOT BEEN ZEROED
C  N........ORDER OF TOEPLITZ MATRIX
C  N2M1.....2*N-1. T MUST BE AT LEAST THIS LENGTH
C  ID1......ROW-DIMENSION OF R
C
C  QUICK OUTLINE OF FTO4
C  ---------------------
C  DEFINE
C  TM = T(SUPERSCRIPT N-1) = LEADING SUBMATRIX OF T
C  RM = R(SUPERSCRIPT N-1) = ORTHOGONAL TRIANGULAR FACTOR OF TM
C  YM = Y(SUPERSCRIPT N-1) = ORTHOGONAL MATRIX SUCH THAT YM*TM=RM
C  R-DOT = MATRIX WITH 1ST ROW = T(1,1..N), REST OF 1ST
C          COLUMN = YM * T(2..N,1),
C          TRAILING SUBMATRIX = LEADING SUBMATRIX OF T.
C  R-TILDE = (R(1..N-1,.))
C            ( T(N,.)    )
C
C  R-DOT MAY BE TRANSFORMED TO R USING GILL ET AL'S RANK-1
C  ORTHOGONAL UPDATE, WITH AN UPSWEEP AND A DOWNSWEEP OF
C  GIVENS TRANSFORMS (ANGLES PHI-UP AND PHI-DOWN). THEN,
C  R-TILDE MAY BE TRANSFORMED TO AN UPPER-TRIANGULAR MATRIX
C  WITH RM AS ITS LEADING SUBMATRIX BY INVERSE TRANSFORMS.
C  WE CAN SIMILARLY TRANSFORM A MATRIX WITH YM AS ITS TRAILING
C  SUBMATRIX TO A MATRIX WITH YM AS ITS LEADING SUBMATRIX.
C    SUPPOSE THAT RM(.J-1), R(.J-1), YM(J-1.) AND Y(J.) ARE
C  KNOWN. THEN R-DOT(.J) CAN BE CONSTRUCTED. PERFORM A GILL
C  UPSWEEP AND DOWNSWEEP ON R-DOT(.J), YIELDING R(.J).
C  Y(J.)  MAY BE OBTAINED FROM YM(J-1.) IN A SIMILAR MANNER,
C  RE-ARRANGING THE CALCULATIONS TO OPERATE ON ROWS, RATHER
C  THAN COLUMNS.
```

```
C    CONSTRUCT R-TILDE(.J) FROM THE DEFINITION, AND PERFORM
C  J INVERSE TRANSFORMS ON R-TILDE(.J), YIELDING RM(.J).
C  YM(.J) MAY BE OBTAINING FROM Y(.J) BY AN INVERSE
C  TRANSFORM. THIS COMPLETES THE BASIC RECURSION.
C         NOTE THAT THE GILL SWEEPS AND INVERSE TRANSFORMS,
C  WHICH ACCOUNT FOR MOST OF THE COMPUTING LOAD, REQUIRE
C  ONLY O(N) OPS./RECURSION CYCLE, SO THE WHOLE ALG.
C  REQUIRES ONLY O(N**2) OPS. DETAILS OF THE CALC. OF
C  THE GILL AND INVERSE TRANSFORM ANGLES (THIS CALC.
C  PROCEEDS IN TANDEM WITH THE BASIC RECURSION) ARE GIVEN
C  IN ALG.8.3.2.

C  NOTATION - WORKING STORAGE
C  --------------------------
C  CPHIUP)  O/P COSINES,SINES (1ST LETTER 'C','S') OF THE
C  SPHIUP)..GIVENS ROTATIONS('PHIUP':PHI-UPSWEEP; 'PHIDN':
C  CPHIDN)  PHI-DOWNSWEEP) USED IN PHASE I OF Q AND R
C  SPHIDN)  RECURSIONS. NOTATION FOLLOWS ALG.8.3.2.
C
C  SECTHE)  O/P SEC(THETA),TAN(THETA), WHERE THE THETAS
C  TANTHE)..ARE THE ANGLES OF THE INVERSE TRANSFORMS USED
C        )  USED IN PHASE II OF THE Q AND R RECURSIONS.
C        )  NOTATION FOLLOWS ALG. 8.3.2.
C
C  RM.......TEMP. VECTOR TO STORE COL. OF R(SUPERSCRIPT N-1)
C  RD1......O/P VECTOR CONTAINING 1ST COL OF R-DOT
C  RDJ......TEMP. VECTOR TO STORE COL.J OF R-DOT.
C  SIGD.....O/P VECTOR CONTAINING VALUES OF SIGMA-DOT
C  YD.......TEMP.VECTOR CONTAINING Y-DOT(J.)
C  YD1......TEMP.VECTOR CONTAINING Y-DOT-1(J.), AND OVERWRITTEN
C                                  BY Y-DOT-1(J+1.)
C  YD2......TEMP.VECTOR CONTAINING Y-DOT-2(J+1.)
C  YD3......TEMP.VECTOR CONTAINING Y-DOT-3(J.), AND OVERWRITTEN
C                                  BY Y-DOT-3(J+1.)
C  YTN......TEMP.VECTOR CONTAINING CURRENT VALUE OF Y-TILDE(N.)
C  YM.......TEMP.VECTOR CONTAINING YM(J.)

      REAL*8 T(N2M1),R(ID1,N),CPHIUP(100),SPHIUP(100),CPHIDN(100),
     1  SECTHE(100),TANTHE(100),RM(100),RD1(100),SIGD(100),
     2  SPHIDN(100),RDJ(100),RTNJ
      REAL*8 Y(ID1,N),YD(100),YD1(100),YD2(100),YD3(100),YM(100),
     1  YTN(100)
      REAL*8 EPS

C  INITIALIZATION

      IF(N.LE.100) GO TO 5
      WRITE(6,2)
    2 FORMAT(' ** N.GT.100 - NO PROCESSING BY FTO4')
      RETURN

    5 EPS=10.**(-5)
      M=N-1

C  CALC R(1,1) AND RM(1,1) USING NORM-INVARIANCE
      RM(1)=0.
      DO 10 I=1,M
```

```
   10 RM(1)=RM(1)+T(I-1+N)**2
      R(1,1)=RM(1)+T(N2M1)*T(N2M1)
      R(1,1)=DSQRT(R(1,1))
      RM(1)=DSQRT(RM(1))

C  GET R-DOT(1,1),(1,2),(2,2) AND SIGMA-DOT(1),(2), WHERE SIGMA-DOT(I)
C  IS NORM(R-DOT(I..N,1))
      RD1(1)=T(N)
      RDJ(1)=T(N-1)
      RDJ(2)=RM(1)
      SIGD(1)=R(1,1)
      SIGD(2)=DSQRT(SIGD(1)**2 - RD1(1)**2)

C  OTHER INITIALIZATION FOR R-RECURSION - PHIDN(1),THETA(1)
      CPHIDN(1)=RD1(1)/SIGD(1)
      SPHIDN(1)=SIGD(2)/SIGD(1)
      SECTHE(1)=R(1,1)/RM(1)
      TANTHE(1)=T(N2M1)/RM(1)

C  INITLZN.FOR Q-RECURSION - Y-DOT(1.),YM(1.),Y(1.),
C  Y-DOT-1(2.),Y-DOT-3(2.),Y-TILDE(N.)
      YD(1)=1.
      DO 12 I=2,N
   12 YD(I)=0.
      DO 14 I=1,M
   14 YM(I)=T(I-1+N)/RM(1)
      YM(N)=0.
      DO 16 I=1,N
      Y(1,I)=T(I-1+N)/R(1,1)
      YD1(I)=(Y(1,I)-CPHIDN(1)*YD(I))/SPHIDN(1)
      YD3(I)=-SPHIDN(1)*YD(I) + CPHIDN(1)*YD1(I)
   16 YTN(I)= -TANTHE(1)*Y(1,I)
      YTN(N)= -TANTHE(1)*Y(1,N) + SECTHE(1)

C******MAIN LOOP - CALC R(.J),Y(J.),RM(.J) AND YM(J.)*******

      J=2
C
C  R-RECURSION PHASE I - CALC R(.J) FROM R-DOT(.J)
C
   30 RDJ(1)=T(1-J+N)
      DO 40 I=2,J
   40 RDJ(I)=RM(I-1)
C  RM CONTAINS RM(.J-1)

C  CALC  R-DOT(J,1) = YM(J-1.)*T(2..N,1)
   48 RD1(J)=0.
      DO 50 I=1,M
   50 RD1(J)=RD1(J) + YM(I)*T(I+N)

C  IF J=N, CHECK THAT RD1(N) IS POSITIVE. IF NOT, RD(N.) WAS
C  REVERSED IN SIGN IN THE PREVIOUS STEP - SO RDJ(N) MUST BE
C  REPLACED BY -RDJ(N).
      IF(J.NE.N) GO TO 65
      IF(RD1(N).LT.0) RDJ(N)=-RDJ(N)
      GO TO 70
```

```
C  NEXT PHIUP
   65 SIGD(J+1)=DSQRT(SIGD(J)**2-RD1(J)**2)
      CPHIUP(J)=RD1(J)/SIGD(J)
      SPHIUP(J)=SIGD(J+1)/SIGD(J)
C  SINGULARITY CHECK
      IF(DABS(SPHIUP(J)).GE.EPS) GO TO 68
      WRITE(6,66)
   66 FORMAT(' I/P MATRIX NUMERICALLY SINGULAR - ERROR EXIT')
      RETURN

C  APPLY PHIUP(J), THEN PHI(1..J-1) ON RDJ
   68 RDJ(J+1)=0.
      CALL GIVENS(RDJ(J),RDJ(J+1),CPHIUP(J),SPHIUP(J))
   70 CALL RSWEEP(RDJ,R(1,J),CPHIUP,SPHIUP,CPHIDN,SPHIDN,J-1,J)

C  CALC AND APPLY LAST PHIDN IN DOWN-SWEEP (N/A IF J=N)
      IF(J.NE.N) GO TO 110
      DO 105 I=1,N
  105 Y(N,I)=YD3(I)
      RETURN
  110 R(J,J)=DSQRT(RDJ(J)**2 + RDJ(J+1)**2)
      CPHIDN(J)=RDJ(J)/R(J,J)
      SPHIDN(J)=RDJ(J+1)/R(J,J)

C  Q-RECURSION PHASE I - CALC Y-DOT(J.),Y-DOT-1(J+1.),Y-DOT-2(J+1.),
C  Y(J.) AND Y-DOT-3(J+1.)
      YD(1)=0.
      DO 112 I=2,N
  112 YD(I)=YM(I-1)
      DO 115 I=1,N
      YD1(I)=(YD1(I)-CPHIUP(J)*YD(I))/SPHIUP(J)
      YD2(I)=-SPHIUP(J)*YD(I) + CPHIUP(J)*YD1(I)
      Y(J,I)=CPHIDN(J)*YD3(I) + SPHIDN(J)*YD2(I)
  115 YD3(I)=-SPHIDN(J)*YD3(I) + CPHIDN(J)*YD2(I)

C  R-RECURSION PHASE II - CALC RM(.J) FROM R-TILDE(.J)

C  INITIALIZE R-TILDE(N,J)
      RTNJ=T(2*N-J)

C  FIRST J-1 INVERSE TRANSFORMS
      JM1=J-1
         DO 120 I=1,JM1
         RM(I)=R(I,J)
  120    CALL INVXFM(RM(I),RTNJ,SECTHE(I),TANTHE(I))

C  CALC AND APPLY LAST THETA
      RM(J)=DSQRT(R(J,J)**2 - RTNJ**2)
C  SINGULARITY CHECK
      IF(RM(J)/R(J,J).GE.EPS) GO TO 140
      WRITE(6,130)
  130 FORMAT(' LDG.SUBMX.NUMERICALLY SINGULAR - ERROR EXIT',
     1  /' REMEDY - AUGMENT I/P MATRIX AND RERUN')
      RETURN

  140 SECTHE(J)=R(J,J)/RM(J)
      TANTHE(J)=RTNJ/RM(J)
```

```
C  Q-RECURSION PHASE II - CALC. YM(J.) AND NEW YTN
      DO 160 I=1,N
  160 YM(I)=Y(J,I)
      DO 165 I=1,N
  165 CALL INVXFM(YM(I),YTN(I),SECTHE(J),TANTHE(J))

      J=J+1
      GO TO 30
C
C***************END OF MAIN LOOP************************

      END
C
C
C
C
C  SUBROUTINES RSWEEP, INVXFM ARE THE SAME AS IN FTO2.
C
C
C
```

```
INPUT TOEPLITZ MATRIX T
  18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900    3.700    2.200
 -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900    3.700
   2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900
   2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100
   4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800
  -8.600    4.200    2.500    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300
  -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900
   8.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000
   6.200    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200
  -1.500    6.200    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600

REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS
 -28.016    7.747   -0.780    8.378  -17.201    9.468    2.254    5.784   -9.204    3.869
   0.0    -25.554    5.698    2.817    1.874  -13.458   10.343    5.309    2.054   -7.868
   0.0      0.0    -27.275    7.360   -0.716    4.318  -12.849   11.746    2.957    2.031
   0.0      0.0      0.0    -25.782    2.909   -0.052    6.139  -10.894   10.403    2.651
   0.0      0.0      0.0      0.0    -22.936    2.543   -2.887    4.710  -11.662   10.939
   0.0      0.0      0.0      0.0      0.0    -21.736    1.355   -3.165    4.339   -6.728
   0.0      0.0      0.0      0.0      0.0      0.0    -20.661    2.342   -1.687    2.130
   0.0      0.0      0.0      0.0      0.0      0.0      0.0    -20.551    2.515   -3.749
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0   -20.393    0.606
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0   17.394

QT = PRODUCT OF HOUSEHOLDER TRANSFORMS
  -0.715    0.565   -0.096   -0.111   -0.161    0.331    0.058   -0.077   -0.008    0.058
  -0.459   -0.557    0.546   -0.132   -0.162   -0.064    0.354    0.035   -0.081    0.010
  -0.369   -0.360   -0.565    0.515   -0.121   -0.129   -0.082    0.325    0.038   -0.073
  -0.198   -0.290   -0.373   -0.625    0.465   -0.033   -0.078   -0.091    0.333    0.057
   0.210   -0.281   -0.262   -0.293   -0.640    0.388   -0.131   -0.088   -0.143    0.342
   0.054    0.197   -0.297   -0.266   -0.355   -0.652    0.451   -0.116   -0.097   -0.149
  -0.110    0.108    0.216   -0.323   -0.206   -0.323   -0.641    0.496   -0.073   -0.124
  -0.111   -0.065    0.135    0.191   -0.280   -0.216   -0.372   -0.629    0.511   -0.088
  -0.173   -0.036   -0.035    0.091    0.241   -0.312   -0.257   -0.358   -0.611    0.487
   0.035    0.149    0.106    0.098   -0.033   -0.212    0.151    0.295    0.459    0.767

REDUCED MATRIX AS CALCULATED BY FT04
  26.016   -7.747    0.780   -8.378   17.201   -9.468   -2.254   -5.784    9.204   -3.869
   0.0     25.554   -5.698   -2.817   -1.874   13.458  -10.343   -5.309   -2.054    7.868
   0.0      0.0     27.275   -7.360    0.716   -4.318   12.849  -11.746   -2.957   -2.031
   0.0      0.0      0.0     25.782   -2.909    0.052   -6.139   10.894  -10.403   -2.651
   0.0      0.0      0.0      0.0     22.936   -2.543    2.887   -4.710   11.662  -10.939
   0.0      0.0      0.0      0.0      0.0     21.736   -1.355    3.165   -4.339    6.728
   0.0      0.0      0.0      0.0      0.0      0.0     20.661   -2.342    1.687   -2.130
   0.0      0.0      0.0      0.0      0.0      0.0      0.0     20.551   -2.515    3.749
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0     20.393   -0.606
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0   -17.394

QT = ORTHOG MATRIX THAT REDUCES T
   0.715   -0.565    0.096    0.111    0.161   -0.331   -0.058    0.077    0.008   -0.058
   0.459    0.557   -0.546    0.132    0.162    0.064   -0.354   -0.035    0.081   -0.010
   0.369    0.360    0.565   -0.515    0.121    0.129    0.082   -0.325   -0.038    0.073
   0.198    0.290    0.373    0.625   -0.465    0.033    0.078    0.091   -0.333   -0.057
  -0.210    0.281    0.262    0.293    0.640   -0.388    0.131    0.088    0.143   -0.342
  -0.054   -0.197    0.297    0.266    0.355    0.652   -0.451    0.116    0.097    0.149
   0.110   -0.108   -0.216    0.323    0.206    0.323    0.641   -0.496    0.073    0.124
   0.111    0.065   -0.135   -0.191    0.280    0.216    0.372    0.629   -0.511    0.088
   0.173    0.036    0.035   -0.091   -0.241    0.312    0.257    0.358    0.611   -0.487
  -0.035   -0.149   -0.106   -0.096    0.033    0.212   -0.151   -0.295   -0.459   -0.767
READY
```

```
C  TEST PROGRAM FOR FTO5, WHICH ORTHOGONALLY-TRIANGULARIZES A
C  TOEPLTIZ MATRIX IN O(N**2) OPERATIONS. FTO5 USES THE
C  GILL-GOLUB-MURRAY-SAUNDERS ORTHOGONAL-UPDATE ALGORITHM
C  WITH FAST GIVENS TRANSFORMS, FAST INVERSE TRANSFORMS,
C  AND THE SHIFT-INVARIANCE OF THE TOEPLITZ MATRIX.
C
C  TEST PROCEDURE: (1) GENERATE TOEP.MX. 'A' WITH 1ST ROW
C  AND COL FROM 'T'. (2) REDUCE TOEP.MX. WITH HOUSEHOLDER
C  TRANSFORMS (O/P OVERWRITES 'A') TO UPR.TRIANGULAR FORM
C  (3) CALL FTO5 TO PRODUCE UPR.TRIANGLE IN 'R'.
C
C  OTHER VARIABLES: N=MATRIX ORDER, ID1=ROW DIM.OF MATRICES
C
      REAL*8 T(19),A(10,10),R(10,10)
      DATA T/2.2,3.7,-5.9,-1.1,-2.8,6.3,-4.9,8.0,
     1  6.2,18.6,-14.7,2.5,2.9,4.2,-8.6,-1.5,2.0,
     2  0.2,-1.5/

      ID1=10
      N=10
      N2M1=2*N-1
      NM1=N-1

      CALL TOEPMX(T,A,N,N2M1,ID1)
      WRITE(6,10)
   10 FORMAT(////////////' INPUT TOEPLITZ MATRIX T')
      CALL PRNTMX(A,N,N,ID1)

C  REDUCE TOEPLITZ MATRIX BY HOUSEHOLDER TRANSFORMS
      CALL HOUTRI(A,N,ID1)
      DO 30 I=2,N
      IM1=I-1
      DO 30 J=1,IM1
   30 A(I,J)=0.

      WRITE(6,40)
   40 FORMAT(/' REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS')
      CALL PRNTMX(A,N,N,ID1)

C  REDUCE TOEPLITZ MATRIX USING FTO5
      CALL FTO5(T,R,N,N2M1,ID1)
      DO 15 I=2,N
      IM1=I-1
      DO 15 J=1,IM1
   15 R(I,J)=0.

      WRITE(6,20)
   20 FORMAT(/' REDUCED MATRIX AS CALCULATED BY FTO5')
      CALL PRNTMX(R,N,N,ID1)
      STOP
      END



      SUBROUTINE FTO5(T,R,N,N2M1,ID1)
C
C  S/R TO TRIANGULARIZE A TOEPLITZ MATRIX USING GILL ET AL'S
```

```
C  ORTHOGONAL UPDATE ALG. WITH FAST GIVENS TRANSFORMS, FAST
C  INVERSE TRANSFORMS, AND SHIFT-INVARIANCE. FOLLOWS ALG.
C  8.4.1, AND THE VARIABLE NOTATION IS THE SAME.
C
C
C  PARAMETERS:
C  T........VECTOR CONTAINING 1ST ROW AND COL OF TOEPLITZ MATRIX
C           TO BE TRIANGULARIZED. IF N IS THE ORDER OF THE MATRIX,
C           ORDER OF STORAGE IS (1,N),(1,N-1),...,(1,1),(2,1),...
C           ...,(N,1)
C  R........CONTAINS UPPER-TRIANGULAR MATRIX PRODUCED BY THE
C           ORTHOGONALIZATION. NOTE THAT R(I,J),I.GT.J HAVE
C           NOT BEEN ZEROED
C  N........ORDER OF TOEPLITZ MATRIX
C  N2M1.....2*N-1
C  ID1......ROW-DIMENSION OF R
C
C  TEMPORARIES:
C  TPHIUP)
C  SPHIUP)   TAN'S,COS'S,SINES(1ST LETTER=T,C,S) OF GIVENS
C  SPHIDN)..ROTATIONS USED (PHIUP:PHI-UPSWEEP,PHIDN:PHI-
C  TPHIDN)  DOWNSWEEP) IN PHASE I.
C  CPHIDN)
C
C  FGAUP1)  CO-EFFS OF FAST GIVENS TRANSFORM MATRICES USED IN
C  FGAUP2)..PHASE I. (LETTERS 4,5 : UP-UPSWEEP, DN-DOWNSWEEP;
C  FGADN1)  LAST LETTER : 1 - (1,2) ELEMENT, 2 - (2,1) ELEMENT)
C  FGADN2)
C
C  COSTHE)..COS,TAN OF ANGLES THETA USED IN FAST INVERSE
C  TANTHE)  TRANSFORMS OF PHASE II
C
C  FGB1)    1ST AND 2ND CO-EFFICIENTS OF FAST INVERSE TRANSFORMS
C  FGB2)....USED IN PHASE II. CORRESPOND TO C1,C2 IN 'FINVT'
C      )    CALL.
C
C  RM.......USED TO STORE CURRENT(JTH) COL.OF R(SUPERSCRIPT N-1)
C  WM.......SCALED VERSION OF RM
C  DM.......SCALING FACTORS FOR WM. RM(I)=DM(I)*WM(I).
C  W........SCALED VERSION OF R
C  D........SCALING FACTORS FOR W. R(I,J)=D(I)*W(I,J)
C  RD1......1ST COL OF R-DOT
C  RDJ......USED TO STORE CURRENT (J-TH) COL OF R-DOT
C           (OVERWRITTEN BY RESULTS OF FAST GIVENS TRANSFORMS)
C  WDJ......SCALED VERSION OF RDJ
C  DD.......INITIAL SCALING FACTORS OF WDJ. RDJ(I)=DD(I)*WDJ(I)
C  DD1......SCALE FACTOR OF WDJ(I) AFTER 1ST FAST GIVENS TRANSFM
C  DD2......SCALE FACTOR OF WDJ(I) AFTER 2ND FAST GIVENS TRANSFM
C  DD3......SCALE FACTOR OF WDJ(I) AFTER 3RD FAST GIVENS TRANSFM
C  RTNJ.....CURRENT VALUE OF R-TILDE(N,J)
C  WTNJ.....SCALED VERSION OF RTNJ
C  DTN......SCALE FACTOR OF WTNJ. AFTER I-TH FAST INVERSE
C           TRANSFORM, RTNJ=DTN(I)*WTNJ
C  SIGD.....CURRENT VALUE OF SIGMA-DOT (=SIGMA-DOT(J))
C  Z........ VECTOR (= T-TRANSPOSE * (T(2,1),...,T(N,1))
C                                                  TRANSPOSE )
C  S........S(I)=DM(I)*RD1(I+1)
```

```
C
C  GENERAL REMARKS ON FTO5
C  ----------------------
C  FTO5 IS THE SAME AS FTO3 EXCEPT THAT FAST GIVENS
C  TRANSFORMS AND FAST INVERSE TRANSFORMS ARE USED INSTEAD
C  OF THE STANDARD TRANSFORMS. FOR UNSCALED VARIABLES
C  OF THE TYPE R**, THE SCALED VARIABLES ARE W** AND THE
C  SCALING FACTORS ARE D**, WITH THE EXCEPTION OF DD1 TO
C  DD3, WHICH ARE THE SCALING FACTORS OF THE ROWS OF
C  RDJ AFTER THE 1ST,2ND AND 3RD FAST GIVENS TRANSFORM
C  AFFECTING THAT PARTICULAR ROW.
C    THE WHOLE ALGORITHM REQUIRES 4.5N**2 OPS. TO COMPUTE
C  THE SCALED VERSION OF R, THOUGH IT THEN CALCULATES AND
C  AND RETURNS THE ACTUAL R-MATRIX. THIS RE-SCALING IS
C  NOT USUALLY NECESSARY.

      REAL*8 T(N2M1),R(ID1,N)
      REAL*8 W(20,20),WM(20),WDJ(20),RDJ(20),
     1  RD1(20),D(20),DD(20),DD1(20),DD2(20),DD3(20),
     2  DM(20),Z(20),S(20),RM(20),SIGD(20),RTNJ,WTNJ,
     3  DTN(20)
      REAL*8 CPHIUP(20),SPHIUP(20),TPHIUP(20),CPHIDN(20),
     1  TPHIDN(20),TANTHE(20),COSTHE(20),FGAUP1(20),
     2  FGAUP2(20),FGADN1(20),FGADN2(20),
     3  FGB1(20),FGB2(20)
      REAL*8 EPS

C  INITIALIZATION

      M=N-1
      EPS=10.**(-5)

C  CALC R(1,1) AND RM(1,1) USING NORM-INVARIANCE
      RM(1)=0.
      DO 10 I=1,M
   10 RM(1)=RM(1)+T(I-1+N)**2
      R(1,1)=RM(1)+T(N2M1)*T(N2M1)
      R(1,1)=DSQRT(R(1,1))
      RM(1)=DSQRT(RM(1))

C  GET R-DOT(1,1),(1,2),(2,2) AND SIGMA-DOT(1),(2), WHERE SIGMA-DOT(I)
C  IS NORM(R-DOT(I..N,1))
      RD1(1)=T(N)
      RDJ(1)=T(N-1)
      RDJ(2)=RM(1)
      SIGD(1)=R(1,1)
      SIGD(2)=DSQRT(SIGD(1)**2 - RD1(1)**2)

C  INITIALIZE VARIABLES FOR PHASE I
      CPHIDN(1)=RD1(1)/SIGD(1)
      TPHIDN(1)=SIGD(2)/RD1(1)
      D(1)=CPHIDN(1)
      DD(1)=1
      DD3(1)=DD(1)
      W(1,1)=R(1,1)/D(1)
C  INITIALIZE VARIABLE FOR PHASE II
      RTNJ=T(N2M1)
```

```
      COSTHE(1)=RM(1)/R(1,1)
      TANTHE(1)=RTNJ/RM(1)
      DM(1)=D(1)/COSTHE(1)
      WM(1)=RM(1)/DM(1)
      FGB1(1)=TANTHE(1)/DM(1)
      FGB2(1)= -TANTHE(1)*DM(1)
      DTN(1)=COSTHE(1)

C  Z=TRANSPOSE OF PRINC SUBMX OF T * T(2..N,1)
      DO 28 I=1,M
      Z(I)=0.
      DO 28 J=1,M
   28 Z(I)=Z(I)+T(J-I+N)*T(J+N)

C********MAIN LOOP - CALC W(.J) AND WM(.J)*****************

      J=2
C
C  PHASE I - CALC W(.J) FROM W-DOT(.J)
C
   30 JM1=J-1
      JM2=J-2
      WDJ(1)=T(1-J+N)
      DO 40 I=2,J
   40 WDJ(I)=WM(I-1)
      DD(J)=DM(J-1)
      DD1(J)=DD(J)
      RDJ(J)=WDJ(J)*DD(J)
C  WM CONTAINS WM(.J-1)
C
C  BACK-SUBST FOR S(JM1) IN EQN. WM(TRANS)*S=Z. THEN
C  RD1(2..N,.)=S
      S(JM1)=Z(JM1)
      IF(J.EQ.2) GO TO 60
      DO 50 I=1,JM2
   50 S(JM1)=S(JM1)-WM(I)*S(I)
   60 S(JM1)=S(JM1)/WM(JM1)
      RD1(J)=S(JM1)/DM(JM1)

C  IF J=N, CHECK THAT RD1(N) IS POSITIVE. IF NOT, RD(N.) WAS
C  REVERSED IN SIGN IN THE PREVIOUS STEP - SO RDJ(N) MUST BE
C  REPLACED BY -RDJ(N).
      IF(J.NE.N) GO TO 65
      IF(RD1(N).LT.0) WDJ(N)=-WDJ(N)
      GO TO 70

C  NEXT PHIUP
   65 SIGD(J+1)=DSQRT(SIGD(J)**2-RD1(J)**2)
      CPHIUP(J)=RD1(J)/SIGD(J)
      SPHIUP(J)=SIGD(J+1)/SIGD(J)
      TPHIUP(J)=SIGD(J+1)/RD1(J)
C  SINGULARITY CHECK
      IF(DABS(SPHIUP(J)).GE.EPS) GO TO 68
      WRITE(6,66)
   66 FORMAT(' I/P MATRIX NUMERICALLY SINGULAR - ERROR EXIT')
      RETURN
```

```
C  FAST GGMS UPSWEEP/DOWNSWEEP ON W-DOT(.J) (PROC 8.4.2)
C  (1) UP-SWEEP ON W-DOT(.J)
   68 DD1(J)=DD(J)*CPHIUP(J)
      DD2(J)=DD1(J)
      RDJ(J+1)= -RDJ(J)*SPHIUP(J)
   70 IF(J.EQ.2) GO TO 105
C  CALC FGT MATRIX FOR PHIUP(J-1)
      FGAUP1(JM1)=TPHIUP(JM1)*DD1(J)/DD(JM1)
      FGAUP2(JM1)= -TPHIUP(JM1)*DD(JM1)/DD1(J)
      DD2(J)=DD1(J)*CPHIUP(JM1)

      I=JM1
      DO 80 II=2,JM1
      CALL FGT(WDJ(I),WDJ(I+1),FGAUP1(I),FGAUP2(I))
   80 I=I-1

C  (2) DOWN-SWEEP ON R-DOT(.J), EXCLUDING LAST TWO PHIDN'S
      DO 100 I=1,JM2
      CALL FGT(WDJ(I),WDJ(I+1),FGADN1(I),FGADN2(I))
  100 W(I,J)=WDJ(I)

C  CALC AND APPLY FGT FOR PHIDN(J-1)
  105 FGADN1(JM1)=TPHIDN(JM1)*DD2(J)/DD3(JM1)
      FGADN2(JM1)= -TPHIDN(JM1)*DD3(JM1)/DD2(J)
      DD3(J)=DD2(J)*CPHIDN(JM1)
      CALL FGT(WDJ(JM1),WDJ(J),FGADN1(JM1),FGADN2(JM1))
      W(JM1,J)=WDJ(JM1)

C  CALC AND APPLY LAST PHIDN IN DOWN-SWEEP (N/A IF J=N)
      IF(J.NE.N) GO TO 110
      W(N,N)=WDJ(N)
      D(N)=DD3(N)
      GO TO 158

  110 RDJ(J)=DD3(J)*WDJ(J)
      R(J,J)=DSQRT(RDJ(J)**2 + RDJ(J+1)**2)
      CPHIDN(J)=RDJ(J)/R(J,J)
      TPHIDN(J)=RDJ(J+1)/RDJ(J)
      D(J)=DD3(J)*CPHIDN(J)
      W(J,J)=R(J,J)/D(J)

C  PHASE II - CALC WM(.J) FROM W-TILDE(.J)

C  INITIALIZE W-TILDE(N,J)
      WTNJ=T(2*N-J)

C  1ST J-1 FAST INVERSE TRANSFORMS
        DO 120 I=1,JM1
        WM(I)=W(I,J)
  120   CALL FINVT(WM(I),WTNJ,FGB1(I),FGB2(I))
C
C  CALC AND APPLY LAST THETA
      RTNJ=WTNJ*DTN(JM1)
      RM(J)=DSQRT(R(J,J)**2 - RTNJ**2)
C  SINGULARITY CHECK
      IF(RM(J)/R(J,J).GE.EPS) GO TO 140
      WRITE(6,123)
```

```
  123 FORMAT(' LDG SUBMX NUMERICALLY SINGULAR - ERROR EXIT',
     1       /' REMEDY - AUGMENT I/P MATRIX AND RERUN')
      RETURN

  140 COSTHE(J)=RM(J)/R(J,J)
      TANTHE(J)=RTNJ/RM(J)
      DM(J)=D(J)/COSTHE(J)
      WM(J)=RM(J)/DM(J)
      FGB1(J)=TANTHE(J)*DTN(JM1)/DM(J)
      FGB2(J)= -TANTHE(J)*DM(J)/DTN(JM1)
      DTN(J)=DTN(JM1)*COSTHE(J)

      J=J+1
      GO TO 30
C
C***************END OF MAIN LOOP************************

C  RE-SCALE R
  158 DO 160 I=1,N
      DO 160 J=1,N
  160 R(I,J)=W(I,J)*D(I)
      RETURN
      END



      SUBROUTINE FGT(A,B,C1,C2)
C
C  S/R TO APPLY A FAST GIVENS TRANSFORM
C
C  OPERATION:
C
C     ( A )     ( 1   C1) ( A )
C     (   )  =  (       ) (   )
C     ( B )     ( C2   1) ( B )

      REAL*8 A,B,C1,C2,T

      T=A+C1*B
      B=C2*A+B
      A=T
      RETURN
      END



      SUBROUTINE FINVT(X,Y,C1,C2)
C
C  S/R TO APPLY A FAST INVERSE TRANSFORM
C
C  OPERATION:
C  SOLVE FOR X AND Y' IN
C
C    ( X')      ( 1   C1 ) ( X )
C    (   )  =   (        ) (   )
C    ( Y')      ( C2   1 ) ( Y )
```

```
REAL*8 X,Y,C1,C2

X=X-C1*Y
Y=C2*X+Y
RETURN
END
```

```
INPUT TOEPLITZ MATRIX T
  18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900    3.700    2.200
 -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900    3.700
   2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100   -5.900
   2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800   -1.100
   4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300   -2.800
  -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900    6.300
  -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000   -4.900
   2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200    8.000
   0.200    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600    6.200
  -1.500    0.200    2.000   -1.500   -8.600    4.200    2.900    2.500  -14.700   18.600

REDUCED MATRIX USING HOUSEHOLDER TRANSFORMS
 -26.016    7.747   -0.780    8.378  -17.201    9.468    2.254    5.784   -9.204    3.869
   0.0    -25.554    5.698    2.817    1.874  -13.458   10.343    5.309    2.054   -7.868
   0.0      0.0    -27.275    7.360   -0.716    4.318  -12.849   11.746    2.957    2.031
   0.0      0.0      0.0    -25.782    2.909   -0.052    6.139  -10.894   10.403    2.651
   0.0      0.0      0.0      0.0    -22.936    2.543   -2.887    4.710  -11.662   10.939
   0.0      0.0      0.0      0.0      0.0    -21.736    1.355   -3.165    4.339   -6.728
   0.0      0.0      0.0      0.0      0.0      0.0    -20.661    2.342   -1.687    2.130
   0.0      0.0      0.0      0.0      0.0      0.0      0.0    -20.551    2.515   -3.749
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    -20.393    0.606
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0     17.394

REDUCED MATRIX AS CALCULATED BY FT05
  26.016   -7.747    0.780   -8.378   17.201   -9.468   -2.254   -5.784    9.204   -3.869
   0.0     25.554   -5.698   -2.817   -1.874   13.458  -10.343   -5.309   -2.054    7.868
   0.0      0.0     27.275   -7.360    0.716   -4.318   12.849  -11.746   -2.957   -2.031
   0.0      0.0      0.0     25.782   -2.909    0.052   -6.139   10.894  -10.403   -2.651
   0.0      0.0      0.0      0.0     22.936   -2.543    2.887   -4.710   11.662  -10.939
   0.0      0.0      0.0      0.0      0.0     21.736   -1.355    3.165   -4.339    6.728
   0.0      0.0      0.0      0.0      0.0      0.0     20.661   -2.342    1.687   -2.130
   0.0      0.0      0.0      0.0      0.0      0.0      0.0     20.551   -2.515    3.749
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0     20.393   -0.606
   0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0      0.0    -17.394
READY
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C       MISCELLANEOUS MATRIX ROUTINES:                                C
C                                                                     C
C                   COPYMX                                            C
C                   GIVENS                                            C
C                   HOUQR                                             C
C                   HOUTRI                                            C
C                   IDMX                                              C
C                   MULTMX                                            C
C                   PRNTMX                                            C
C                   SINGSM                                            C
C                   TOEPMX                                            C
C                   TRANMX                                            C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

```
      SUBROUTINE COPYMX(A,B,M,N,ID1)
C  S/R TO COPY A(M X N) TO B.
C  (ID1 IS THE ROW-DIMENSION OF A AND B)

      REAL*8 A(ID1,N),B(ID1,N)

      DO 10 I=1,M
      DO 10 J=1,N
   10 B(I,J)=A(I,J)
      RETURN
      END
```

```
      SUBROUTINE GIVENS(A,B,COSX,SINX)
      REAL*8 A,B,T,COSX,SINX
C
C  S/R TO PERFORM GIVENS ROTATION OF ANGLE X ON A AND B.

      T=A
      A=A*COSX + B*SINX
      B=-T*SINX + B*COSX
      RETURN
      END
```

```
      SUBROUTINE HOUQR(A,QT,N,ID1)
C  S/R TO TRIANGULARIZE A(N X N) USING HOUSEHOLDER'S METHOD,
C  AND ACCUMULATE THE H/H TRANSFMS IN QT.
C  (ID1 IS THE ROW-DIMENSION OF A AND QT).

      REAL*8 A(ID1,N),QT(ID1,N),UP
      NM1=N-1
      CALL IDMX(QT,N,ID1)
      DO 10 J=1,NM1
      CALL VHS12(1,J,J+1,N,A(1,J),1,UP,A(1,J+1),1,ID1,N-J)
   10 CALL VHS12(2,J,J+1,N,A(1,J),1,UP,QT(1,1),1,ID1,N)
      RETURN
      END
```

```
      SUBROUTINE HOUTRI(A,N,ID1)
C
C  S/R TO TRIANGULARIZE A(N X N) USING HOUSEHOLDER'S METHOD.
C  ID1 IS THE ROW-DIMENSION OF A.
C  SEE IMSL WRITE-UP FOR VHS12, WHICH PERFORMS A SINGLE
C  HOUSEHOLDER TRANSFORM

      REAL*8 A(ID1,N),UP
      NM1=N-1
      DO 10 J=1,NM1
   10 CALL VHS12(1,J,J+1,N,A(1,J),1,UP,A(1,J+1),1,ID1,N-J)
      RETURN
      END
```

```
      SUBROUTINE IDMX(A,N,ID1)
C
C  S/R TO GENERATE THE IDENTITY MATRIX IN A(N X N).
C  (ID1 IS THE ROW-DIMENSION OF A).

      REAL*8 A(ID1,N)
      DO 20 I=1,N
      DO 10 J=1,N
   10 A(I,J)=0.
   20 A(I,I)=1.
      RETURN
      END
```

```
      SUBROUTINE MULTMX(A,B,C,L,M,N,ID1,ID2,ID3)

C  S/R TO MULTIPLY A(L X M) BY B(M X N), YIELDING C(L X N).
C  ID1,ID2 AND ID3 ARE THE ROW-DIMENSIONS OF A,B AND C
C  RESPECTIVELY.
C
      REAL*8 A(ID1,M),B(ID2,N),C(ID3,N)

      DO 10 I=1,L
      DO 10 J=1,N
      C(I,J)=0.
      DO 10 K=1,M
   10 C(I,J)=C(I,J)+A(I,K)*B(K,J)
      RETURN
      END
```

```
      SUBROUTINE PRNTMX(A,M,N,ID1)
C
C  S/R TO PRINT A(M X N). (ID1 IS THE ROW-
C  DIMENSION OF A). FORMAT IS AS FOLLOWS:
C  MAX MAGN .LT. 10000 : 14F9.3
C  10000.LE.MAX MAGN .LT.10**7 : 14F9.0
C  MAX MAGN .GE. 10**7 : 14D9.2

      REAL*8 A(ID1,N),MAXMAG

C  FIND MAX MAGNITUDE
      MAXMAG=0.
      DO 10 I=1,M
      DO 10 J=1,N
      IF(DABS(A(I,J)).GT.MAXMAG) MAXMAG=DABS(A(I,J))
   10 CONTINUE

      IFMT=1
      IF(MAXMAG.GE.10000.) IFMT=2
      IF(MAXMAG.GE.1D7) IFMT=3

      DO 50 I=1,M
      GO TO (20,30,40),IFMT
   20 WRITE(6,21) (A(I,J),J=1,N)
   21 FORMAT(1X,14F9.3)
      GO TO 50
   30 WRITE(6,31) (A(I,J),J=1,N)
   31 FORMAT(1X,14F9.0)
      GO TO 50
   40 WRITE(6,41) (A(I,J),J=1,N)
   41 FORMAT(1X,14D9.2)
   50 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SINGSM(A,M,N,ID1,I,J,K,LI,LJ,WKAREA,K2P3,IER)
C
C  S/R TO RE-CALCULATE A(I,J) TO MAKE A K X K SUBMATRIX WITH
C  A(I,J) ON A CORNER SINGULAR. IF LI IS +1(-1), A(I,J) IS ON A
C  TOP(BOTTOM) CORNER. IF LJ IS +1(-1), A(I,J) IS ON A LEFT
C  (RIGHT) CORNER.
C  OTHER PARAMETERS
C  M,N ..... ROW AND COLUMN ORDERS OF A
C  ID1 ..... ROW-DIMENSION OF A
C  WKAREA ... WORKING AREA OF SIZE AT LEAST K*(2*K+3)
C  K2P3 ..... 2*K+3
C             GE.K*(2K*3)
C  IER ..... ERROR PARAMETER
C             IER=0 NO ERROR
C             IER=1 TOP OF SUBMATRIX IS ABOVE TOP OF A
C             IER=2 BOTTOM OF SUBMATRIX IS BELOW BOTTOM OF A
C             IER=4 LEFT OF SUBMATRIX IS BEFORE LEFT OF A
C             IER=8 RIGHT OF SUBMATRIX IS AFTER RIGHT OF A
C             IER=16 LI.NE.-1 OR 1
C             IER=32 LJ.NE.-1 OR 1
C             IER CAN ALSO BE A COMBINATION OF THE ABOVE,
C             E.G. IER=10=8+2 MEANS THAT SUBMATRIX EXTENDS
C             BOTH BELOW AND TO THE LEFT OF A
C
C  ALGORITHM. LET B BE THE SUBMATRIX THAT IS TO BE MADE SINGULAR,
C  LET C BE THE SUBMATRIX OF B FOUND BY DELETING THE ROW AND
C  COLUMN OCCUPIED BY A(I,J), AND LET D-TRANSPOSE AND F BE
C  THE ROW AND COLUMN OF B LOCATED BY, BUT EXCLUDING A(I,J).
C  THEN, FOR B TO BE SINGULAR,
C
C                  A(I,J) = D-TRANSPOSE * C-INVERSE * F
C
      REAL*8 A(ID1,N),WKAREA(K,K2P3)

      IER=0
      IDGT=0
      KM1=K-1
C
C  CHECK FOR ERROR CONDITIONS
      IER=0
      ITOP=I+KM1*((LI-1)/2)
      IF(ITOP.LT.1) IER=IER+1
      IF(ITOP+KM1.GT.M) IER=IER+2
      LEFT=J+KM1*((LJ-1)/2)
      IF(LEFT.LT.1) IER=IER+4
      IF(LEFT+KM1.GT.N) IER=IER+8
      IF(LI.NE.-1.AND.LI.NE.1) IER=IER+16
      IF(LJ.NE.-1.AND.LJ.NE.1) IER=IER+32
      IF(IER.EQ.0) GO TO 5
      WRITE(6,2) IER
    2 FORMAT(' SINGSM ERROR COND =',I3,'  NO OPS PERFORMED')
      RETURN

C  I1F,J1F ARE THE STARTING INDICES OF F, AND I1C,J1C ARE
C  THE STARTING INDICES OF C
    5 I1F=1
      IF(LI.EQ.-1) I1F=I-KM1
```

```
      IF(LI.EQ.1) I1F=I+1
      J1F=J
      I1C=I1F
      J1C=1
      IF(LJ.EQ.-1) J1C=J-KM1
      IF(LJ.EQ.1) J1C=J+1
C
C  MOVE F AND C TO WORKING AREA AND SOLVE C * P = F
      DO 10 II=1,KM1
      WKAREA(II,K)=A(I1F+II-1,J1F)
      DO 10 JJ=1,KM1
   10 WKAREA(II,JJ)=A(I1C+II-1,J1C+JJ-1)
      CALL LEQT2F(WKAREA,1,KM1,K,WKAREA(1,K),IDGT,WKAREA(1,K+1),IER)
C
C  COMPUTE A(I,J) = D-TRANSPOSE * P. (I1D,J1D ARE THE STARTING
C  INDICES OF D)
      I1D=I
      J1D=J1C
      A(I,J)=0.
      DO 20 II=1,KM1
   20 A(I,J)=A(I,J) + A(I1D,J1D+II-1) * WKAREA(II,K)
      RETURN
      END
```

```
      SUBROUTINE TOEPMX(T,A,N,N2M1,ID1)
C
C  S/R TO MAKE UP A TOEPLITZ MATRIX FROM ITS 1ST ROW AND COL.
C
C  T - ELEMENTS (1,N),(1,N-1),...,(1,1),(2,1),...,(N,1) OF
C       TOEPLITZ MATRIX (INPUT)
C  A - TOEPLITZ MATRIX IN FULL (OUTPUT)
C  N - ORDER OF A
C  N2M1- 2*N-1. T MUST BE AT LEAST THIS LENGTH
C  ID1 - ROW DIMENSION OF A

      REAL*8 T(N2M1),A(ID1,N)

      DO 10 K=1,N
      A(K,1)=T(N+K-1)
   10 A(1,K)=T(N-K+1)

      DO 20 I=2,N
      DO 20 J=2,N
   20 A(I,J)=A(I-1,J-1)
      RETURN
      END
```

```
      SUBROUTINE TRANMX(A,N,ID1)

C  S/R TO TRANSPOSE A(N X N). ID1 IS THE ROW-DIMENSION OF A.
C
      REAL*8 A(ID1,N),TEMP
      DO 10 I=2,N
      IM1=I-1
      DO 10 J=1,IM1
      TEMP=A(I,J)
      A(I,J)=A(J,I)
   10 A(J,I)=TEMP
      RETURN
      END
```