```
      SUBROUTINE SGBCO(ABD,LDA,N,ML,MU,IPVT,RCOND,Z)
C***BEGIN PROLOGUE  SGBCO
C***DATE WRITTEN   780814   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  D2A2
C***KEYWORDS  BANDED,CONDITION,FACTOR,LINEAR ALGEBRA,LINPACK,MATRIX
C***AUTHOR  MOLER, C. B., (U. OF NEW MEXICO)
C***PURPOSE  Factors a real BAND matrix by Gaussian elimination
C            and estimates the condition number of the matrix.
C***DESCRIPTION
C
C     SBGCO factors a real band matrix by Gaussian
C     elimination and estimates the condition of the matrix.
C
C     If  RCOND  is not needed, SGBFA is slightly faster.
C     To solve  A*X = B , follow SBGCO by SGBSL.
C     To compute  INVERSE(A)*C , follow SBGCO by SGBSL.
C     To compute  DETERMINANT(A) , follow SBGCO by SGBDI.
C
C     On Entry
C
C        ABD     REAL(LDA, N)
C                contains the matrix in band storage.  The columns
C                of the matrix are stored in the columns of  ABD  and
C                the diagonals of the matrix are stored in rows
C                ML+1 through 2*ML+MU+1 of  ABD .
C                See the comments below for details.
C
C        LDA     INTEGER
C                the leading dimension of the array  ABD .
C                LDA must be .GE. 2*ML + MU + 1 .
C
C        N       INTEGER
C                the order of the original matrix.
C
C        ML      INTEGER
C                number of diagonals below the main diagonal.
C                0 .LE. ML .LT. N .
C
C        MU      INTEGER
C                number of diagonals above the main diagonal.
C                0 .LE. MU .LT. N .
C                More efficient if  ML .LE. MU .
C
C     On Return
C
C        ABD     an upper triangular matrix in band storage and
C                the multipliers which were used to obtain it.
C                The factorization can be written  A = L*U  where
C                L  is a product of permutation and unit lower
C                triangular matrices and  U  is upper triangular.
C
C        IPVT    INTEGER(N)
C                an integer vector of pivot indices.
C
C        RCOND   REAL
C                an estimate of the reciprocal condition of  A .
C                For the system  A*X = B , relative perturbations
C                in  A  and  B  of size  EPSILON  may cause
C                relative perturbations in  X  of size  EPSILON/RCOND .
C                If  RCOND  is so small that the logical expression
C                           1.0 + RCOND .EQ. 1.0
C                is true, then  A  may be singular to working
```

```
C                precision.  In particular,  RCOND  is zero  if
C                exact singularity is detected or the estimate
C                underflows.
C
C        Z       REAL(N)
C                a work vector whose contents are usually unimportant.
C                If  A  is close to a singular matrix, then  Z  is
C                an approximate null vector in the sense that
C                NORM(A*Z) = RCOND*NORM(A)*NORM(Z) .
C
C     Band Storage
C
C           If  A  is a band matrix, the following program segment
C           will set up the input.
C
C                   ML = (band width below the diagonal)
C                   MU = (band width above the diagonal)
C                   M = ML + MU + 1
C                   DO 20 J = 1, N
C                      I1 = MAX0(1, J-MU)
C                      I2 = MIN0(N, J+ML)
C                      DO 10 I = I1, I2
C                         K = I - J + M
C                         ABD(K,J) = A(I,J)
C                10    CONTINUE
C                20 CONTINUE
C
C           This uses rows  ML+1  through  2*ML+MU+1  of  ABD .
C           In addition, the first  ML  rows in  ABD  are used for
C           elements generated during the triangularization.
C           The total number of rows needed in  ABD  is  2*ML+MU+1 .
C           The  ML+MU by ML+MU  upper left triangle and the
C           ML by ML  lower right triangle are not referenced.
C
C     Example:  If the original matrix is
C
C           11 12 13  0  0  0
C           21 22 23 24  0  0
C            0 32 33 34 35  0
C            0  0 43 44 45 46
C            0  0  0 54 55 56
C            0  0  0  0 65 66
C
C      then  N = 6, ML = 1, MU = 2, LDA .GE. 5  and ABD should contain
C
C            *  *  *  +  +  +  , * = not used
C            *  * 13 24 35 46  , + = used for pivoting
C            * 12 23 34 45 56
C           11 22 33 44 55 66
C           21 32 43 54 65  *
C
C     LINPACK.  This version dated 08/14/78 .
C     Cleve Moler, University of New Mexico, Argonne National Lab.
C
C     Subroutines and Functions
C
C     LINPACK SGBFA
C     BLAS SAXPY,SDOT,SSCAL,SASUM
C     Fortran ABS,AMAX1,MAX0,MIN0,SIGN
C***REFERENCES  DONGARRA J.J., BUNCH J.R., MOLER C.B., STEWART G.W.,
C                  *LINPACK USERS  GUIDE*, SIAM, 1979.
C***ROUTINES CALLED  SASUM,SAXPY,SDOT,SGBFA,SSCAL
C***END PROLOGUE  SGBCO
      INTEGER LDA,N,ML,MU,IPVT(1)
```

```
      REAL ABD(LDA,1),Z(1)
      REAL RCOND
C
      REAL SDOT,EK,T,WK,WKM
      REAL ANORM,S,SASUM,SM,YNORM
      INTEGER IS,INFO,J,JU,K,KB,KP1,L,LA,LM,LZ,M,MM
C
C     COMPUTE 1-NORM OF A
C
C***FIRST EXECUTABLE STATEMENT  SGBCO
      ANORM = 0.0E0
      L = ML + 1
      IS = L + MU
      DO 10 J = 1, N
         ANORM = AMAX1(ANORM,SASUM(L,ABD(IS,J),1))
         IF (IS .GT. ML + 1) IS = IS - 1
         IF (J .LE. MU) L = L + 1
         IF (J .GE. N - ML) L = L - 1
   10 CONTINUE
C
C     FACTOR
C
      CALL SGBFA(ABD,LDA,N,ML,MU,IPVT,INFO)
C
C     RCOND = 1/(NORM(A)*(ESTIMATE OF NORM(INVERSE(A)))) .
C     ESTIMATE = NORM(Z)/NORM(Y) WHERE  A*Z = Y  AND  TRANS(A)*Y = E .
C     TRANS(A)  IS THE TRANSPOSE OF A .  THE COMPONENTS OF  E  ARE
C     CHOSEN TO CAUSE MAXIMUM LOCAL GROWTH IN THE ELEMENTS OF W  WHERE
C     TRANS(U)*W = E .  THE VECTORS ARE FREQUENTLY RESCALED TO AVOID
C     OVERFLOW.
C
C     SOLVE TRANS(U)*W = E
C
      EK = 1.0E0
      DO 20 J = 1, N
         Z(J) = 0.0E0
   20 CONTINUE
      M = ML + MU + 1
      JU = 0
      DO 100 K = 1, N
         IF (Z(K) .NE. 0.0E0) EK = SIGN(EK,-Z(K))
         IF (ABS(EK-Z(K)) .LE. ABS(ABD(M,K))) GO TO 30
            S = ABS(ABD(M,K))/ABS(EK-Z(K))
            CALL SSCAL(N,S,Z,1)
            EK = S*EK
   30    CONTINUE
         WK = EK - Z(K)
         WKM = -EK - Z(K)
         S = ABS(WK)
         SM = ABS(WKM)
         IF (ABD(M,K) .EQ. 0.0E0) GO TO 40
            WK = WK/ABD(M,K)
            WKM = WKM/ABD(M,K)
         GO TO 50
   40    CONTINUE
            WK = 1.0E0
            WKM = 1.0E0
   50    CONTINUE
         KP1 = K + 1
         JU = MIN0(MAX0(JU,MU+IPVT(K)),N)
         MM = M
         IF (KP1 .GT. JU) GO TO 90
            DO 60 J = KP1, JU
               MM = MM - 1
```

```
                  SM = SM + ABS(Z(J)+WKM*ABD(MM,J))
                  Z(J) = Z(J) + WK*ABD(MM,J)
                  S = S + ABS(Z(J))
   60          CONTINUE
               IF (S .GE. SM) GO TO 80
                  T = WKM - WK
                  WK = WKM
                  MM = M
                  DO 70 J = KP1, JU
                     MM = MM - 1
                     Z(J) = Z(J) + T*ABD(MM,J)
   70             CONTINUE
   80          CONTINUE
   90       CONTINUE
            Z(K) = WK
  100    CONTINUE
         S = 1.0E0/SASUM(N,Z,1)
         CALL SSCAL(N,S,Z,1)
C
C        SOLVE TRANS(L)*Y = W
C
         DO 120 KB = 1, N
            K = N + 1 - KB
            LM = MINO(ML,N-K)
            IF (K .LT. N) Z(K) = Z(K) + SDOT(LM,ABD(M+1,K),1,Z(K+1),1)
            IF (ABS(Z(K)) .LE. 1.0E0) GO TO 110
               S = 1.0E0/ABS(Z(K))
               CALL SSCAL(N,S,Z,1)
  110       CONTINUE
            L = IPVT(K)
            T = Z(L)
            Z(L) = Z(K)
            Z(K) = T
  120    CONTINUE
         S = 1.0E0/SASUM(N,Z,1)
         CALL SSCAL(N,S,Z,1)
C
         YNORM = 1.0E0
C
C        SOLVE L*V = Y
C
         DO 140 K = 1, N
            L = IPVT(K)
            T = Z(L)
            Z(L) = Z(K)
            Z(K) = T
            LM = MINO(ML,N-K)
            IF (K .LT. N) CALL SAXPY(LM,T,ABD(M+1,K),1,Z(K+1),1)
            IF (ABS(Z(K)) .LE. 1.0E0) GO TO 130
               S = 1.0E0/ABS(Z(K))
               CALL SSCAL(N,S,Z,1)
               YNORM = S*YNORM
  130       CONTINUE
  140    CONTINUE
         S = 1.0E0/SASUM(N,Z,1)
         CALL SSCAL(N,S,Z,1)
         YNORM = S*YNORM
C
C        SOLVE  U*Z = W
C
         DO 160 KB = 1, N
            K = N + 1 - KB
            IF (ABS(Z(K)) .LE. ABS(ABD(M,K))) GO TO 150
               S = ABS(ABD(M,K))/ABS(Z(K))
```

```
257               CALL SSCAL(N,S,Z,1)
258               YNORM = S*YNORM
259    150     CONTINUE
260            IF (ABD(M,K) .NE. 0.0E0) Z(K) = Z(K)/ABD(M,K)
261            IF (ABD(M,K) .EQ. 0.0E0) Z(K) = 1.0E0
262            LM = MINO(K,M) - 1
263            LA = M - LM
264            LZ = K - LM
265            T = -Z(K)
266            CALL SAXPY(LM,T,ABD(LA,K),1,Z(LZ),1)
267    160  CONTINUE
268  C      MAKE ZNORM = 1.0
269         S = 1.0E0/SASUM(N,Z,1)
270         CALL SSCAL(N,S,Z,1)
271         YNORM = S*YNORM
272  C
273         IF (ANORM .NE. 0.0E0) RCOND = YNORM/ANORM
274         IF (ANORM .EQ. 0.0E0) RCOND = 0.0E0
275         RETURN
276         END
```

```
      SUBROUTINE SGBDI(ABD,LDA,N,ML,MU,IPVT,DET)
C***BEGIN PROLOGUE  SGBDI
C***DATE WRITTEN   780814   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  D3A2
C***KEYWORDS  BANDED,DETERMINANT,FACTOR,INVERSE,LINEAR ALGEBRA,LINPACK,
C             MATRIX
C***AUTHOR  MOLER, C. B., (U. OF NEW MEXICO)
C***PURPOSE  Computes the determinant of a BAND matrix
C            using the factors computed by SGBCO or SGBFA.
C            If the inverse is needed, use SGBSL  N  times.
C***DESCRIPTION
C
C     SGBDI computes the determinant of a band matrix
C     using the factors computed by SBGCO or SGBFA.
C     If the inverse is needed, use SGBSL  N  times.
C
C     On Entry
C
C        ABD     REAL(LDA, N)
C                the output from SBGCO or SGBFA.
C
C        LDA     INTEGER
C                the leading dimension of the array  ABD .
C
C        N       INTEGER
C                the order of the original matrix.
C
C        ML      INTEGER
C                number of diagonals below the main diagonal.
C
C        MU      INTEGER
C                number of diagonals above the main diagonal.
C
C        IPVT    INTEGER(N)
C                the pivot vector from SBGCO or SGBFA.
C
C     On Return
C
C        DET     REAL(2)
C                determinant of original matrix.
C                Determinant = DET(1) * 10.0**DET(2)
C                with  1.0 .LE. ABS(DET(1)) .LT. 10.0
C                or  DET(1) = 0.0 .
C
C     LINPACK.  This version dated 08/14/78 .
C     Cleve Moler, University of New Mexico, Argonne National Lab.
C
C     Subroutines and Functions
C
C     Fortran ABS
C***REFERENCES  DONGARRA J.J., BUNCH J.R., MOLER C.B., STEWART G.W.,
C                 *LINPACK USERS  GUIDE*, SIAM, 1979.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  SGBDI
      INTEGER LDA,N,ML,MU,IPVT(1)
      REAL ABD(LDA,1),DET(2)
C
      REAL TEN
      INTEGER I,M
C
C***FIRST EXECUTABLE STATEMENT  SGBDI
      M = ML + MU + 1
```

```
      DET(1) = 1.0E0
      DET(2) = 0.0E0
      TEN = 10.0E0
      DO 50 I = 1, N
         IF (IPVT(I) .NE. I) DET(1) = -DET(1)
         DET(1) = ABD(M,I)*DET(1)
C        ...EXIT
         IF (DET(1) .EQ. 0.0E0) GO TO 60
   10    IF (ABS(DET(1)) .GE. 1.0E0) GO TO 20
            DET(1) = TEN*DET(1)
            DET(2) = DET(2) - 1.0E0
         GO TO 10
   20    CONTINUE
   30    IF (ABS(DET(1)) .LT. TEN) GO TO 40
            DET(1) = DET(1)/TEN
            DET(2) = DET(2) + 1.0E0
         GO TO 30
   40    CONTINUE
   50 CONTINUE
   60 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SGBFA(ABD,LDA,N,ML,MU,IPVT,INFO)
C***BEGIN PROLOGUE  SGBFA
C***DATE WRITTEN   780814   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  D2A2
C***KEYWORDS  BANDED,FACTOR,LINEAR ALGEBRA,LINPACK,MATRIX
C***AUTHOR  MOLER, C. B., (U. OF NEW MEXICO)
C***PURPOSE  Factors a real BAND matrix by elimination.
C***DESCRIPTION
C
C     SGBFA factors a real band matrix by elimination.
C
C     SGBFA is usually called by SBGCO, but it can be called
C     directly with a saving in time if  RCOND  is not needed.
C
C     On Entry
C
C        ABD     REAL(LDA, N)
C                contains the matrix in band storage.  The columns
C                of the matrix are stored in the columns of  ABD  and
C                the diagonals of the matrix are stored in rows
C                ML+1 through 2*ML+MU+1 of  ABD .
C                See the comments below for details.
C
C        LDA     INTEGER
C                the leading dimension of the array  ABD .
C                LDA must be .GE. 2*ML + MU + 1 .
C
C        N       INTEGER
C                the order of the original matrix.
C
C        ML      INTEGER
C                number of diagonals below the main diagonal.
C                0 .LE. ML .LT. N .
C
C        MU      INTEGER
C                number of diagonals above the main diagonal.
C                0 .LE. MU .LT. N .
C                More efficient if  ML .LE. MU .
C     On Return
C
C        ABD     an upper triangular matrix in band storage and
C                the multipliers which were used to obtain it.
C                The factorization can be written  A = L*U , where
C                L  is a product of permutation and unit lower
C                triangular matrices and  U  is upper triangular.
C
C        IPVT    INTEGER(N)
C                an integer vector of pivot indices.
C
C        INFO    INTEGER
C                = 0  normal value.
C                = K  if  U(K,K) .EQ. 0.0 .  This is not an error
C                     condition for this subroutine, but it does
C                     indicate that SGBSL will divide by zero if
C                     called.  Use  RCOND  in SBGCO for a reliable
C                     indication of singularity.
C
C     Band Storage
C
C           If  A  is a band matrix, the following program segment
C           will set up the input.
C
```

```
C                    ML = (band width below the diagonal)
C                    MU = (band width above the diagonal)
C                    M = ML + MU + 1
C                    DO 20 J = 1, N
C                       I1 = MAXO(1, J-MU)
C                       I2 = MINO(N, J+ML)
C                       DO 10 I = I1, I2
C                          K = I - J + M
C                          ABD(K,J) = A(I,J)
C                 10    CONTINUE
C                 20 CONTINUE
C
C           This uses rows  ML+1  through  2*ML+MU+1  of  ABD .
C           In addition, the first  ML  rows in  ABD  are used for
C           elements generated during the triangularization.
C           The total number of rows needed in  ABD  is  2*ML+MU+1 .
C           The  ML+MU by ML+MU  upper left triangle and the
C           ML by ML  lower right triangle are not referenced.
C
C     LINPACK.  This version dated 08/14/78 .
C     Cleve Moler, University of New Mexico, Argonne National Lab.
C
C     Subroutines and Functions
C
C     BLAS SAXPY,SSCAL,ISAMAX
C     Fortran MAXO,MINO
C***REFERENCES  DONGARRA J.J., BUNCH J.R., MOLER C.B., STEWART G.W.,
C                  *LINPACK USERS  GUIDE*, SIAM, 1979.
C***ROUTINES CALLED  ISAMAX,SAXPY,SSCAL
C***END PROLOGUE  SGBFA
      INTEGER LDA,N,ML,MU,IPVT(1),INFO
      REAL ABD(LDA,1)
C
      REAL T
      INTEGER I,ISAMAX,IO,J,JU,JZ,JO,J1,K,KP1,L,LM,M,MM,NM1
C
C***FIRST EXECUTABLE STATEMENT  SGBFA
      M = ML + MU + 1
      INFO = 0
C
C     ZERO INITIAL FILL-IN COLUMNS
C
      JO = MU + 2
      J1 = MINO(N,M) - 1
      IF (J1 .LT. JO) GO TO 30
      DO 20 JZ = JO, J1
         IO = M + 1 - JZ
         DO 10 I = IO, ML
            ABD(I,JZ) = 0.0E0
   10    CONTINUE
   20 CONTINUE
   30 CONTINUE
      JZ = J1
      JU = 0
C
C     GAUSSIAN ELIMINATION WITH PARTIAL PIVOTING
C
      NM1 = N - 1
      IF (NM1 .LT. 1) GO TO 130
      DO 120 K = 1, NM1
         KP1 = K + 1
C
C        ZERO NEXT FILL-IN COLUMN
C
```

```
489           JZ = JZ + 1
490           IF (JZ .GT. N) GO TO 50
491           IF (ML .LT. 1) GO TO 50
492              DO 40 I = 1, ML
493                 ABD(I,JZ) = 0.0E0
494     40        CONTINUE
495     50     CONTINUE
496 C
497 C           FIND L = PIVOT INDEX
498 C
499           LM = MINO(ML,N-K)
500           L = ISAMAX(LM+1,ABD(M,K),1) + M - 1
501           IPVT(K) = L + K - M
502 C
503 C           ZERO PIVOT IMPLIES THIS COLUMN ALREADY TRIANGULARIZED
504 C
505           IF (ABD(L,K) .EQ. 0.0E0) GO TO 100
506 C
507 C              INTERCHANGE IF NECESSARY
508 C
509              IF (L .EQ. M) GO TO 60
510                 T = ABD(L,K)
511                 ABD(L,K) = ABD(M,K)
512                 ABD(M,K) = T
513     60        CONTINUE
514 C
515 C              COMPUTE MULTIPLIERS
516 C
517              T = -1.0E0/ABD(M,K)
518              CALL SSCAL(LM,T,ABD(M+1,K),1)
519 C
520 C              ROW ELIMINATION WITH COLUMN INDEXING
521 C
522              JU = MINO(MAXO(JU,MU+IPVT(K)),N)
523              MM = M
524              IF (JU .LT. KP1) GO TO 90
525              DO 80 J = KP1, JU
526                 L = L - 1
527                 MM = MM - 1
528                 T = ABD(L,J)
529                 IF (L .EQ. MM) GO TO 70
530                    ABD(L,J) = ABD(MM,J)
531                    ABD(MM,J) = T
532     70          CONTINUE
533                 CALL SAXPY(LM,T,ABD(M+1,K),1,ABD(MM+1,J),1)
534     80        CONTINUE
535     90        CONTINUE
536           GO TO 110
537    100    CONTINUE
538              INFO = K
539    110    CONTINUE
540    120 CONTINUE
541    130 CONTINUE
542        IPVT(N) = N
543        IF (ABD(M,N) .EQ. 0.0E0) INFO = N
544        RETURN
545        END
```

```
546       SUBROUTINE SGBSL(ABD,LDA,N,ML,MU,IPVT,B,JOB)
547 C***BEGIN PROLOGUE  SGBSL
548 C***DATE WRITTEN   780814   (YYMMDD)
549 C***REVISION DATE  820801   (YYMMDD)
550 C***CATEGORY NO.  D2A2
551 C***KEYWORDS  BANDED,LINEAR ALGEBRA,LINPACK,MATRIX,SOLVE
552 C***AUTHOR  MOLER, C. B., (U. OF NEW MEXICO)
553 C***PURPOSE  Solves the real BAND system A*X=B or TRANS(A)*X=B
554 C            using the factors computed by SGBCO or SGBFA.
555 C***DESCRIPTION
556 C
557 C     SGBSL solves the real band system
558 C     A * X = B  or  TRANS(A) * X = B
559 C     using the factors computed by SBGCO or SGBFA.
560 C
561 C     On Entry
562 C
563 C        ABD     REAL(LDA, N)
564 C                the output from SBGCO or SGBFA.
565 C
566 C        LDA     INTEGER
567 C                the leading dimension of the array  ABD .
568 C
569 C        N       INTEGER
570 C                the order of the original matrix.
571 C
572 C        ML      INTEGER
573 C                number of diagonals below the main diagonal.
574 C
575 C        MU      INTEGER
576 C                number of diagonals above the main diagonal.
577 C
578 C        IPVT    INTEGER(N)
579 C                the pivot vector from SBGCO or SGBFA.
580 C
581 C        B       REAL(N)
582 C                the right hand side vector.
583 C
584 C        JOB     INTEGER
585 C                = 0         to solve  A*X = B ,
586 C                = nonzero   to solve  TRANS(A)*X = B , where
587 C                            TRANS(A)  is the transpose.
588 C
589 C     On Return
590 C
591 C        B       the solution vector  X .
592 C
593 C     Error Condition
594 C
595 C        A division by zero will occur if the input factor contains a
596 C        zero on the diagonal.  Technically, this indicates singularity,
597 C        but it is often caused by improper arguments or improper
598 C        setting of LDA .  It will not occur if the subroutines are
599 C        called correctly and if SBGCO has set RCOND .GT. 0.0
600 C        or SGBFA has set INFO .EQ. 0 .
601 C
602 C     To compute  INVERSE(A) * C  where  C  is a matrix
603 C     with  P  columns
604 C           CALL SBGCO(ABD,LDA,N,ML,MU,IPVT,RCOND,Z)
605 C           If (RCOND is too small) GO TO ...
606 C           DO 10 J = 1, P
607 C              CALL SGBSL(ABD,LDA,N,ML,MU,IPVT,C(1,J),0)
608 C        10 CONTINUE
```

```
609 C
610 C     LINPACK.  This version dated 08/14/78 .
611 C     Cleve Moler, University of New Mexico, Argonne National Lab.
612 C
613 C     Subroutines and Functions
614 C
615 C     BLAS SAXPY,SDOT
616 C     Fortran MINO
617 C***REFERENCES  DONGARRA J.J., BUNCH J.R., MOLER C.B., STEWART G.W.,
618 C                  *LINPACK USERS  GUIDE*, SIAM, 1979.
619 C***ROUTINES CALLED  SAXPY,SDOT
620 C***END PROLOGUE  SGBSL
621       INTEGER LDA,N,ML,MU,IPVT(1),JOB
622       REAL ABD(LDA,1),B(1)
623 C
624       REAL SDOT,T
625       INTEGER K,KB,L,LA,LB,LM,M,NM1
626 C***FIRST EXECUTABLE STATEMENT  SGBSL
627       M = MU + ML + 1
628       NM1 = N - 1
629       IF (JOB .NE. 0) GO TO 50
630 C
631 C        JOB = 0 , SOLVE  A * X = B
632 C        FIRST SOLVE L*Y = B
633 C
634          IF (ML .EQ. 0) GO TO 30
635          IF (NM1 .LT. 1) GO TO 30
636             DO 20 K = 1, NM1
637                LM = MINO(ML,N-K)
638                L = IPVT(K)
639                T = B(L)
640                IF (L .EQ. K) GO TO 10
641                   B(L) = B(K)
642                   B(K) = T
643    10          CONTINUE
644                CALL SAXPY(LM,T,ABD(M+1,K),1,B(K+1),1)
645    20       CONTINUE
646    30    CONTINUE
647 C
648 C        NOW SOLVE  U*X = Y
649 C
650          DO 40 KB = 1, N
651             K = N + 1 - KB
652             B(K) = B(K)/ABD(M,K)
653             LM = MINO(K,M) - 1
654             LA = M - LM
655             LB = K - LM
656             T = -B(K)
657             CALL SAXPY(LM,T,ABD(LA,K),1,B(LB),1)
658    40    CONTINUE
659       GO TO 100
660    50 CONTINUE
661 C
662 C        JOB = NONZERO, SOLVE  TRANS(A) * X = B
663 C        FIRST SOLVE  TRANS(U)*Y = B
664 C
665          DO 60 K = 1, N
666             LM = MINO(K,M) - 1
667             LA = M - LM
668             LB = K - LM
669             T = SDOT(LM,ABD(LA,K),1,B(LB),1)
670             B(K) = (B(K) - T)/ABD(M,K)
671    60    CONTINUE
672 C
```

```
673 C         NOW SOLVE TRANS(L)*X = Y
674 C
675           IF (ML .EQ. 0) GO TO 90
676           IF (NM1 .LT. 1) GO TO 90
677              DO 80 KB = 1, NM1
678                 K = N - KB
679                 LM = MINO(ML,N-K)
680                 B(K) = B(K) + SDOT(LM,ABD(M+1,K),1,B(K+1),1)
681                 L = IPVT(K)
682                 IF (L .EQ. K) GO TO 70
683                    T = B(L)
684                    B(L) = B(K)
685                    B(K) = T
686     70          CONTINUE
687     80       CONTINUE
688     90    CONTINUE
689    100 CONTINUE
690        RETURN
691        END
```

```
      subroutine adummy
c..... cliche storage set up here

      cliche param
      parameter (izx=150, jrx=50, izx2=izx-2  )
      parameter ( itbw=2*izx-1  )
      parameter (kxp=(izx-2)*(jrx-2), ibw=izx-1 ,jbw=jrx-1)
      parameter (kbw=(ibw/jbw)*(jbw-ibw)/(jbw/ibw+ibw/jbw)+ibw )
      parameter (lda=3*kbw+1)
      parameter (nplt=3000, nps=100 )
c     parameter (ixpf=4*(izx-2),nfourxf=150)
c..... special version for large grid, no fourier analysis
       parameter (ixpf=2,nfourxf=2)
      parameter (ixpg=2*izx2)
       parameter (neng=500)
      parameter (ksim=51)
      endcliche
      cliche fstor
      use param
      common/fun/f1(izx,jrx),f2(izx,jrx),f3(izx,jrx),f4(izx,jrx)
     c ,f5(izx,jrx),f7(izx,jrx),
     c g1(izx,jrx),g2(izx,jrx),g3(izx,jrx),g4(izx,jrx)
     c ,swg1,swg2,swg3,swg4

      common/equil/b(izx,jrx),rho(izx,jrx),qub(izx,jrx),r(izx,jrx)
     c ,phi(izx,jrx),yyy(izx,jrx),xxx(izx,jrx),qv(izx,jrx)
      common/pertur/xioo(kxp),xio(kxp),xiol(kxp)
     c ,xroo(kxp),xrol(kxp),xrol(kxp)
      endcliche

      cliche matrix
      use param
      common/coeff/a1(kxp,9),a2(kxp,9),a3(kxp,9),b1(kxp,3)
     c ,rhs1(kxp),rhs2(kxp),abar(lda,kxp),ipvt(kxp),zwork(kxp)
c.....unnamed common for dynamic memeory expansion
      common ww(1), ww1(1)
      endcliche

      cliche const
      use param
      common/title/aname(5)
      common/con/gam1,gam2,ix,jx,mm,izxp,kxx,nmax,lmax,isw,ihbw
     c ,fac1,fac2,bias,du,dv,dt,ndiag,ex0,b0,rho0,ex1,fi1,fizx,fj1
     c ,fjrx,kplot,npm,fpsi,fz,fu,fv,azm,apsim,u0,v0,amass,nengx
     c ,fourpi,omegst,omegr,omegexb,flr,sf6,sf8,kplotm,kzs,zedge
     c ,cpuo,cio,syso,valfk,xu,xv,n,pi,vw,psiw,dvin,dvout,ltt,nen,lee
      common/contm/
     c psi0rel,z1rel,z2rel,z3rel,z0rel,nslosh,bmg
     c ,ncenter,pslosh,pcenter,rp,ztrans,ltrans,bm,ltran,ztran,rp1
     c ,bcen,pring,epsp,phicen,phiplg,kin,xpot,ypot,wpot,pfudge,rpx
     c ,phice,phipl,betslsh,betcent,z0,z1,z2,z3,z4,psi0
     c ,betcene,betslse,psloshe,pcentee,bmax,alsi,bm1,psls1,cold
     c ,p2wide,psi3rel,psi3,p1max,bv0,bv3,bv4,bceng,psloshin,psloshen
     c ,nsloshin,pxp1,pxp2,p3a,p3b,p3c,p3d,psim,pe10,ae1,be1,ce1,de1
     c ,psi0erel,psi0e,psime,p2ewide,wp2e,p2floor,p1floor,p2flag
     c ,fring,long,no3d,no1d,dphi,dip,psistr,psislp,psihrel,psih
     c ,dpsihrel,dpsih
       common/pcons/at0,bt0,ct0,cp0
     c ,ap1,ap2,ap3,at1,at2,at3,bp1,bp2,bp3,bt1,bt2,bt3,cp1,cp2,cp3
     c ,ct1,ct2,ct3,bmx1,bmx2,bmx3,bmn1,bmn2,ppas1,ppas2,ppas3,p1trap
     c ,z1c,z2c,z3c,z1min,z2min,as(3),als(3),zs(3),bs(3),dpas1,d1trap
     c ,betrap,betpas1,bvx2,bvx3,ncoil,z2ct
      common/mesh/psi(jrx),z(izx),u(izx),v(jrx),dpsi(jrx),dz(izx)
```

```
      c ,vpsi(jrx),uuz(izx),vpsih(jrx),uuzh(izx)
       common/graf/  xrtime(nplt),xrspz(izx,nps),xrspps(jrx,2*nps)
      c ,time(nplt),xflute(izx,nps),enpot(neng),tenergy(neng)
      x ,enkin(neng),timengy(neng),tenrel(neng),enbend(neng)
      c ,encurve(neng),enflr(neng)
       common/curvco/cr,lb,rw,beta0,delrho,stable,en0,cee,r0,
      c echarg,omeg1,omeg2,en1,besarg,zol,dtrel,p0,omeg0
      c ,omana1,omana2,groana,theta0
       common/tmcon/h12(izx),hzt0(izx),h34(izx),abp(izx),bbp(izx)
      c ,cbp(izx),abf(izx),bbf(izx),cbf(izx),hp3(jrx)
      c ,htrans(izx),abq(izx),bbq(izx),cbq(izx),ebp(izx)
      c ,fbp(izx),gbp(izx),betring,hp0(jrx),hpm(jrx),hpme(jrx),hflr(jrx)
      c ,hzp0(izx),hzp1(izx),hzp2(izx),hzp3(izx),hzt1(izx),hzt2(izx)
      c ,hzt3(izx)
        common/tmfield/bvac(izx),dbvdz(izx),d2bvdz2(izx),dp1dps1(jrx)
      c ,p1(jrx),p1k(ksim,jrx),hpk0(ksim),del11(ksim)
      c ,del12(ksim),del13(ksim),del14(ksim),rzz(izx,jrx),dbdps1(izx,jrx)
      c  ,phi1(izx),phi2(izx),pperp(izx,jrx),ppar(izx,jrx),dflute3(izx)
      c ,qubv(izx,jrx),p2(jrx),dp2dps1(jrx),dflute1(izx),dflute2(izx)
      c ,flute1(jrx),flute2(jrx),flute3(jrx),p2k(ksim,jrx),del15(ksim)
      c ,pperps(izx,jrx),errprp(izx,jrx),errpr1(izx2,jrx-1),xxxfreq(jrx)
      c ,pperpe(izx,jrx),eps1(izx,jrx),omeg1wkb(jrx),omeg2wkb(jrx)
      c ,gamwkb(jrx),dflute4(izx),rhoave(jrx),xxxave(jrx),yyyave(jrx)
      c ,p2t(jrx),dp2dpst(jrx),p3(jrx),dp3dps1(jrx),hpkm(ksim)
      c ,hpkme(ksim),droave(jrx),droterm(izx),ering(izx,jrx)
      c ,grow,growmax,xfreqmax,rzzrmax
       common/forzed/nfour,nfourx,nfourmax,nfourp,jfour,ixp,iocv

       real lb,ltrans,ltran,nslosh,ncenter,nsloshin
       endcliche

       return
       end
```

```
788
789 c..... the main routine
790
791 c ***************************************************************************
792 c *                                                                         *
793 c    FLORA is an initial value stability code developed by R. Freis and
794 c    B. Cohen, based on Newcomb's long thin axisymmetric formalism, including
795 c    finite larmor radius effects. FLORA calculates the linear response to
796 c    low frequency perturbations of the equilibrium magnetic field .
797
798 c*************************************************************************
799 c..... notice of 4/8/82. this version runs correctly for isw=1, and
800 c..... runs correctly for isw=-1 .
801
802 c.....,5/12/82. flora runs testcase 1 , 0 beta, 0 pressure, homogeneous
803 c..... plasma, correctly.
804
805 c..... flora1 transforms variables z,psi to u,v which are always equally
806 c..... spaced. transformation: z=au*u**xu, and psi=apsi*v**xv, where
807 c..... zmax=umax, psimax=vmax, and fz*zmax=fu*umax, fpsi*psimax=fv*vmax .
808 c..... fz, fu, fpsi, fv, input. xu=ln fz / ln fu, xv=ln fpsi / ln fv .
809 c..... au=umax**(-xx+1) , apsi=vmax**(-yy+1) .
810
811 c..... flora2 solves test case 2 , rotating rigid rotor stability, ref:
812 c..... freidberg and pearlstein, phys fluids 21(7) july 1978 1207
813
814
815 c.....flora4 includes background constant density, enbar ( as does flora3 ),
816 c..... and kzs switch which when set to zero, generates initial perturbations
817 c..... independent of z in random spatial generator (ex0=1.) .
818
819
820 c..... flora5
821 c.....  is vectorized version of flora4,(calls rightvec instead of
822 c..... right ), also has timing routine from b. langdon (requires
823 c..... bzohar loaded as a binary ).
824 c..... insert cliche storage here
825
826
827 c..... flora7 is mod. flora5, with psi stretching function
828 c..... exactly centered in amat. (flora5 used linear interpolation
829 c..... to get vpsi(j+1/2)). Also revised diagnostic plots included.
830
831 c.....flora12 is flora11 (rigid rotor with corrected equil. and
832 c..... corrected curvature terms (flora10)) with fourier mode analyses
833 c..... added ( using cpft and rpft) and data for zed post processing.
834 c.....additional input data: jfour (v index at which xr is analyzed in
835 c.....z ), nfourp ( analyze xr every nfour'th time step ),nfourmax
836 c..... (number of times the buffer is read to the history file). note
837 c..... xr is extended a factor of 4 to look like a periodic full wave
838 c..... for cpft. If jfour is input 0, code sets it to jx/4 .
839 c
840 c.....flora13 is flora12 with curvature driven flute mode equilibrium
841 c.....(equilrot replaced by equilcur, rigidcon replaced by curvecon )
842
843 c.....floratm, tandem mirror equilibrium
844 c
845 c.....flortm1, tandem mirror equilibrium, with 3-d plot of equilib.
846 c     quantities added. ( uses tv80 and graflib )
847
848 c......flortex, tandem mirror equilb, with corrections to flortm1. In-
849 c..... put switches swg1, swg2, swg3, swg4 added.
850
```

```
c.....flortm2, like flortex with revised electron ring, a la D'Ippolito
c.....(e-ring pperp in b field only, and additional term in curvature
c..... drive ),

c.....flortm3, like flortm2 with corrections to pressusre normalization,
c..... and additional diagnostics, ( 3-d plots of curvature drive-e ring
c..... term, and perp. pressure balance check ) , also 3-d plots of
c..... pparallel pressure check, and e-psi (=-dphi/dpsi ) , Phi2 modified
c..... to = 1.-(psi/psi3)**ypot ,

c.....flortm4, modified plasma pperp with addition of p3(j) to give
c..... a positive slope near the center,

c.....flortm5, modified p1 in flortm4 to be two functions, pe1 and
c..... pe2, joined at psime with equal slope and value.pe1=ae1+be1*(
c..... psi/psime)+ce1*(psi/psime)**2, and pe2=.5*(1-tanh((psi-psi0e)/p2ewide))
c
c..... flortm6, modified flortm5 as follows; for p2(psi)le. to p2flag ( an
c..... input value), p2 set to p2floor (an input value) and p1 set to
c..... p1floor (an input value). Long-thin ering option added. This modifies
c..... b dependence of ering pperp to look longer (by changing abf, bbf, cbf)
c..... if long ( an input value ) = 1, otherwise leaves pperp of ring un-
c..... changed. Plot output options, no1d=1, prevents graflib plots, no3d=1
c..... prevents tv80lib 3d plots.
c
c..... flortm8 (18 for larger psi grid) modified flortm6 as follows;
c..... the analytic calculation of gamwkb corrected to include drho/dpsi
c..... term (important in the limit of large exb rotation), also phi1
c..... changed to be constants in core and plug ( phicen in core, and
c..... phicen+dphi in plug, dphi a new input variable ), Also b calculated
c..... with expansion in low beta regions.

c..... flortn8, like flortl8 with first order energy check added ,


c.....flrm1 like flortn8 with multi region equilibrium,
c..... Bvac is generated from 3 solonoids ( 1 choke coil and 2
c..... mirror coils). Pressures are the sum of passing and trapped
c..... components. See the glossary of input parameters in subroutine
c..... inputtm for the revised list .

c.....flrm2, either 2 or 3 regions, depending on the number of solonoids
c..... specified (ncoil=2, or 3 ) in the input. For ncoil=2, no passing
c..... pressures are allowed, and the situation is similar to earlier
c..... versions, except that the vaccum fields are generated by solonoids
c..... instead of circular filaments.

c..... flrm3, like flrm2 with corrections to energy subroutine, Also
c..... option to remove hollowness from pperp psi profile (dip=0.), and
c..... ering psi profile changed from quadratic to cubic in inner region.

c.....flrm4, cold plasma halo modeled by changing zmax boundary conditions
c..... for psi > psih (psihrel an input parameter )

c..... flrm6, like flrm4 and flrm5, (mixed boundary condition at zmax,
c..... higher order b. c.) with special yyy to force "rigid mode" in psi.

c.....flrm7, like flrm6 except phi1, p2, yyy restored to original functions

c.....flrm8, like flrm7 with error in qub corrected to read
c.....qub=(b**2-ppar+pperp)/b . (it was qub=(b**2+ppar-pperp)/b)

c.....flrm9,like flrm8 with error in g1 (curvature drive term calculated
c..... in f1to11 ) corrected. Error appeared for cases with grid stretching.
```

```
c.....Also added dpas1 to region 0 as  a constant density.

c.....flrm10, like flrm9 with errorend added if equilibrium tries to
c..... generate negative square roots in calculating B .
c..... Diagnostic message sent to terminal. Also added, maximum xxxfreq
c..... (flr real frequency * dt), and maximum rzz*r. which with grow
c..... and growmax are sent to the screen and 1-d graphics .

c.....flrd10, changed matrix solution method from banfac, bansol to linpac
c..... package sgbco in order to use the cray 2-b machine.
c..... This is still a direct lu factorization method. Elliminated dynamic
c..... memory calls. Changed sub comat to pack the diagonals of the original
c..... banded matrix in rows for sgbco. (Sgbco is in omnilib).

c.....flrd11, improved the accurracy of calculating line integrals
c..... flute1,flute2,flute3,rhoave,xxxave,yyyave,droave by assuming
c..... a quadratic integrand from z=0 to z(1).

c..... flrd12, for kzs=1 (input), a "stiff" initial radial perturbation
c.....  is used. This is of the form xro=constant for psi < psi0 and
c.....  goes linearly to 0 for psi0 < psi(j) < psi(jx) .
      use param
      use fstor
      use matrix
      use const

      data tim/1.e6/
      integer tallyb(2000b)
      common / q8iocs/iocf(0:15)
      common/pic100/npete
      data itally/1/

c.....call link call here
      call link('unit59=terminal,unit2=(infld12,open),unit3=(output,
     c create) //')

      if(itally.gt.0) then
      do 200 ii=1,15
 200  if(iocf(ii).eq.0)go to 210
      ii=0
 210  ioctally=ii
      ioctally=14
      if(ioctally.eq.0)go to 299
c     call timer(ioctally,'ztally00',tallyb,2000b,floratim,1)
 299  itally=-1
      endif
      isw=1
      if(izx.gt.jrx)isw=-1
      jtbw=.5*(1+isw)*itbw+.5*(1-isw)*(2*jrx-1)
      ihbw=.5*(1+isw)*ibw+.5*(1-isw)*(jrx-1)
      nn=jtbw*kxp
      nn1=jtbw+kxp
      namelist/noplot/no1d,no3d
      call ddi(noplot,2,0,0)
      if(no1d.ne.1)call pstart(dev,4rplot,1,'box u21$',1)
c     npete=1
      if(no1d.ne.1)call p100
      call input
      call inputtm
c.....temporary input to test three region model
c     call inptemp
c     call rigidcon
c     call curvecon
      call constant
```

```
      call tmcon2
      call equiltm
c     call equilrot
c     call equilcur
      call f1to11
      call amat
      ml1=kbw
      lda1=lda
      call comat
      call initial
c.....special version for testing fourier analysis and zed file
c.....  maker
c     call fourplay
c     call fourier
      call mymove(xrol(1),xro(1),kxx)
      call mymove(xiol(1),xio(1),kxx)
      if (kzs.eq.1)then
      call mymove(xroo(1),xro(1),kxx)
      call mymove(xioo(1),xio(1),kxx)
      endif
      call sgbco(abar,lda1,kxx,ml1,ml1,ipvt,rcond,zwork)
       call energy
      t=0.
      do 100 n=1,nmax
      t=t+dt
      time(n)=t
      fac1=-1./dt
      fac2=1./dt
      do 90 l=0,lmax
      call rightvec
      call sgbsl(abar,lda1,kxx,ml1,ml1,ipvt,rhs1,0)
      call zmovewrd(xrol,rhs1,kxx)
      call sgbsl(abar,lda1,kxx,ml1,ml1,ipvt,rhs2,0)
      call zmovewrd(xiol,rhs2,kxx)
      fac1=-.5/dt
90    fac2=.5/dt
      call zmovewrd(xioo,xio,kxx)
      call zmovewrd(xio,xiol,kxx)
      call zmovewrd(xroo,xro,kxx)
      call zmovewrd(xro,xrol,kxx)
c.....  time array
      xrtime(n)=xro(kplot)
      if(mod(n,ndiag).eq.0)call diagno
       if(mod(n,nfourp).eq.0)call fourier
      if(((mod(n,nen)).eq.0).or.((ltt.ne.0).and.((lee.le.neng))call energy
100   continue
      call clsdsk(iocv,0)
      call timeused(icp,io,isy)
      cpuo=icp*tim
      cio=io*tim
      syso=isy*tim
      if(no1d.ne.1)
     c call picsher
      call close(100)
      if(no3d.eq.1)go to 300
      call keep80(1,3)
      call fr80id
      call threed
      call plote
300   continue
c     call timend
      call exit(1)
      end
```

```
      subroutine amat

c..... calculates the matrix coefficients for a1, a2, a3, b1, b2
c..... in the equation a1*x(n+1)=a2*x(n)+a3*x(n-1)+b1*y(n)+b2*y(n-1) ,
c..... uses f1 to f11 from subroutine f1to11 and equilibrium quantities.

c..... cliche storage here

      use param
      use fstor
      use matrix
      use const

      dimension bc(jrx),delco1(jrx),delco2(jrx)
      data unit/1./

      gam3=-gam2
      du2=du**2
      dt2=dt**2
      dv2=dv**2
      dvt=2.*dv
      m2=mm**2
      jx=jrx
      ix=izx
      do 10 i=2,ix-1
      do 10 j=2,jx-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      r2=r(i,j)**2
      vp=vpsi(j)
      uz=uuz(i)
      bijmh=(b(i,j)+b(i,j-1))*.5
      bijph=(b(i,j)+b(i,j+1))*.5
      bip1jph=(b(i+1,j+1)+b(i+1,j))*.5
      bip1jmh=(b(i+1,j-1)+b(i+1,j))*.5
      bim1jph=(b(i-1,j+1)+b(i-1,j))*.5
      bim1jmh=(b(i-1,j-1)+b(i-1,j))*.5
      g4iphjph=(g4(i+1,j+1)+g4(i,j)+g4(i+1,j)+g4(i,j+1))*.25*uuzh(i)
      g4iphjmh=(g4(i+1,j-1)+g4(i,j)+g4(i+1,j)+g4(i,j-1))*.25*uuzh(i)
      g4imhjph=(g4(i-1,j+1)+g4(i,j)+g4(i-1,j)+g4(i,j+1))*.25*uuzh(i)
      g4imhjmh=(g4(i-1,j-1)+g4(i,j)+g4(i-1,j)+g4(i,j-1))*.25*uuzh(i)
      g2iphj=(g2(i+1,j)+g2(i,j))*.5*uuzh(i)
      g2imhj=(g2(i-1,j)+g2(i,j))*.5*uuzh(i-1)
      g3iphj=(g3(i+1,j)+g3(i,j))*.5*uuzh(i)
      g3imhj=(g3(i-1,j)+g3(i,j))*.5*uuzh(i-1)

      f1ijph=(f1(i,j)+f1(i,j+1))*.5*vpsih(j)
      f1ijmh=(f1(i,j)+f1(i,j-1))*.5*vpsih(j-1)
      f5ijph=(f5(i,j)+f5(i,j+1))*.5*vpsih(j)
      f5ijmh=(f5(i,j)+f5(i,j-1))*.5*vpsih(j-1)

      if(j.gt.2)go to 60
      f1ijmh=0.
      f6ijmh=0.
60    continue
      uzbar=-uuz(i)*r(i,j)/(du2*dv2)
      a1(k,1)=-gam1*bim1jmh*g4imhjmh*bijmh*uzbar*vpsih(j-1)
     c *r(i-1,j-1)
      a2(k,1)=-gam2*bim1jmh*g4imhjmh*bijmh*uzbar*vpsih(j-1)
     c *r(i-1,j-1)
      a3(k,1)=-gam3*bim1jmh*g4imhjmh*bijmh*uzbar*vpsih(j-1)
     c *r(i-1,j-1)
```

```
      a1(k,2)=-f1ijmh/((dt*dv)**2)+gam1*(f5ijmh/dv2+bijmh**2*uzbar
     c *vpsih(j-1)*r(i,j-1)*(g4imhjmh+g4.phjmh))
      a2(k,2)=-f1ijmh/((dt*dv)**2)+gam2*(f5ijmh/dv2+bijmh**2*uzbar
     c *vpsih(j-1)*r(i,j-1)*(g4imhjmh+g4iphjmh))
      a3(k,2)=-f1ijmh/((dt*dv)**2)+gam3*(f5ijmh/dv2+bijmh**2*uzbar
     c *vpsih(j-1)*r(i,j-1)*(g4imhjmh+g4iphjmh))

      a1(k,3)=-gam1*bip1jmh*g4iphjmh*bijmh*uzbar*vpsih(j-1)*r(i+1,j-1)
      a2(k,3)=-gam2*bip1jmh*g4iphjmh*bijmh*uzbar*vpsih(j-1)*r(i+1,j-1)
      a3(k,3)=-gam3*bip1jmh*g4iphjmh*bijmh*uzbar*vpsih(j-1)*r(i+1,j-1)
      a1(k,4)=gam1*((bim1jmh*g4imhjmh*vpsih(j-1)*bijmh+bim1jph*g4imhjph
     c *vpsih(j)*bijph)*uzbar*r(i-1,j)+mm**2*b(i,j)*uzbar*
     c g2imhj*g3(i-1,j)*dv2/vpsi(j))
      a2(k,4)=gam2*((bim1jmh*g4imhjmh*vpsih(j-1)*bijmh+bim1jph*g4imhjph
     c *vpsih(j)*bijph)*uzbar*r(i-1,j)+mm**2*b(i,j)*uzbar*
     c g2imhj*g3(i-1,j)*dv2/vpsi(j))
      a3(k,4)=gam3*((bim1jmh*g4imhjmh*vpsih(j-1)*bijmh+bim1jph*g4imhjph
     c *vpsih(j)*bijph)*uzbar*r(i-1,j)+mm**2*b(i,j)*uzbar*
     c g2imhj*g3(i-1,j)*dv2/vpsi(j))


      a1(k,5)=((f1ijph+f1ijmh)/dv2-f2(i,j))/dt2+gam1*(-(f5ijph+f5ijmh
     c )/dv2+f7(i,j)+g1(i,j)+(-bijmh**2*(g4imhjmh+g4iphjmh)*vpsih(j-1)
     c -bijph**2*(g4imhjph+g4iphjph)*vpsih(j))*r(i,j)*uzbar-
     c mm**2*b(i,j)*uzbar*(g2imhj+g2iphj)*g3(i,j)*dv2/vpsi(j))
      a2(k,5)=((f1ijph+f1ijmh)/dv2-f2(i,j))/dt2+gam2*(-(f5ijph+f5ijmh
     c )/dv2+f7(i,j)+g1(i,j)+(-bijmh**2*(g4imhjmh+g4iphjmh)*vpsih(j-1)
     c -bijph**2*(g4imhjph+g4iphjph)*vpsih(j))*r(i,j)*uzbar-
     c mm**2*b(i,j)*uzbar*(g2imhj+g2iphj)*g3(i,j)*dv2/vpsi(j))
      a3(k,5)=((f1ijph+f1ijmh)/dv2-f2(i,j))/dt2+gam3*(-(f5ijph+f5ijmh
     c )/dv2+f7(i,j)+g1(i,j)+(-bijmh**2*(g4imhjmh+g4iphjmh)*vpsih(j-1)
     c -bijph**2*(g4imhjph+g4iphjph)*vpsih(j))*r(i,j)*uzbar-
     c mm**2*b(i,j)*uzbar*(g2imhj+g2iphj)*g3(i,j)*dv2/vpsi(j))

      a1(k,6)=gam1*((bip1jmh*g4iphjmh*bijmh*vpsih(j-1)+bip1jph*g4iphjph
     c *bijph*vpsih(j))*uzbar*r(i+1,j)+mm**2*b(i,j)*uzbar*
     c g2iphj*g3(i+1,j)*dv2/vpsi(j))
      a2(k,6)=gam2*((bip1jmh*g4iphjmh*bijmh*vpsih(j-1)+bip1jph*g4iphjph
     c *bijph*vpsih(j))*uzbar*r(i+1,j)+mm**2*b(i,j)*uzbar*
     c g2iphj*g3(i+1,j)*dv2/vpsi(j))
      a3(k,6)=gam3*((bip1jmh*g4iphjmh*bijmh*vpsih(j-1)+bip1jph*g4iphjph
     c *bijph*vpsih(j))*uzbar*r(i+1,j)+mm**2*b(i,j)*uzbar*
     c g2iphj*g3(i+1,j)*dv2/vpsi(j))

      a1(k,7)=gam1*(-bim1jph*g4imhjph*bijph*vpsih(j)*uzbar*r(i-1,j+1))
      a2(k,7)=gam2*(-bim1jph*g4imhjph*bijph*vpsih(j)*uzbar*r(i-1,j+1))
      a3(k,7)=gam3*(-bim1jph*g4imhjph*bijph*vpsih(j)*uzbar*r(i-1,j+1))
      a1(k,8)=-f1ijph/(dt2*dv2)+gam1*(f5ijph/dv2+(bijph**2*(g4imhjph
     c +g4iphjph)*vpsih(j)*r(i,j+1)*uzbar))
      a2(k,8)=-f1ijph/(dt2*dv2)+gam2*(f5ijph/dv2+(bijph**2*(g4imhjph
     c +g4iphjph)*vpsih(j)*r(i,j+1)*uzbar))
      a3(k,8)=-f1ijph/(dt2*dv2)+gam3*(f5ijph/dv2+(bijph**2*(g4imhjph
     c +g4iphjph)*vpsih(j)*r(i,j+1)*uzbar))
      a1(k,9)=gam1*(-bip1jph*g4iphjph*bijph*vpsih(j)*uzbar*r(i+1,j+1))
      a2(k,9)=gam2*(-bip1jph*g4iphjph*bijph*vpsih(j)*uzbar*r(i+1,j+1))
      a3(k,9)=gam3*(-bip1jph*g4iphjph*bijph*vpsih(j)*uzbar*r(i+1,j+1))
c.....b1 array for rhs
       f3ijmh=(f3(i,j)+f3(i,j-1))*.5*vpsih(j-1)
      f3ijph=(f3(i,j)+f3(i,j+1))*.5*vpsih(j)
      denom=1./(dv2)
      b1(k,1)=f3ijmh*denom
      b1(k,3)=f3ijph*denom
      b1(k,2)=-(f3ijmh+f3ijph-f4(i,j)*dv2/vp)*denom
```

```
 10   continue
c.....correct coefficients on boundaries
      sfi1=sign(unit,fi1)
      sfj1=sign(unit,fj1)
      sfjrx=sign(unit,fjrx)
      sfizx=sign(unit,fizx)
c.....  set corners to 0
      k1i=ix-2
      k1j=1+(ix-3)*(jx-2)
      k1=.5*(1+isw)*k1i+.5*(1-isw)*k1j
      fac1=-1.
      if(sfj1.eq.1.and.sfizx.eq.1)fac1=r(ix-1,2)*b(ix-1,2)/
     c (r(ix,2)*b(ix,2))
      a1(k1,5)=a1(k1,5)+fac1*a1(k1,3)
      a2(k1,5)=a2(k1,5)+fac1*a2(k1,3)
      a3(k1,5)=a3(k1,5)+fac1*a3(k1,3)
      a1(k1,3)=0.
      a2(k1,3)=0.
      a3(k1,3)=0.
      k2i=1+(ix-2)*(jx-3)
      k2j=jx-2
      k2=.5*(1+isw)*k2i+.5*(1-isw)*k2j
      fac3=-dvout/dvin
      if(sfjrx.eq.1.and.sfi1.eq.1)fac3=r(2,jx-1)*b(2,jx-1)/
     c (r(1,jx-1)*b(1,jx-1))
       a1(k2,5)=a1(k2,5)+fac3*a1(k2,7)
       a2(k2,5)=a2(k2,5)+fac3*a2(k2,7)
       a3(k2,5)=a3(k2,5)+fac3*a3(k2,7)
      a1(k2,7)=0.
      a2(k2,7)=0.
      a3(k2,7)=0.
      fac2=-1.
      if(sfj1.eq.1.and.sfi1.eq.1)fac2=r(2,2)*b(2,2)/(r(1,2)*b(1,2))
      a1(1,5)=a1(1,5)+fac2*a1(1,1)
      a2(1,5)=a2(1,5)+fac2*a2(1,1)
      a3(1,5)=a3(1,5)+fac2*a3(1,1)
      a1(1,1)=0.
      a2(1,1)=0.
      a3(1,1)=0.
      fac4=-dvout/dvin
c     if(sfjrx.eq.1.and.sfizx.eq.1)fac4=r(ix-1,jx-1)*b(ix-1,jx-1)/
c    c (r(ix,jx-1)*b(ix,jx-1))
      a1(kxp,5)=a1(kxp,5)+fac4*a1(kxp,9)
      a2(kxp,5)=a2(kxp,5)+fac4*a2(kxp,9)
      a3(kxp,5)=a3(kxp,5)+fac4*a3(kxp,9)
      a1(kxp,9)=0.
      a2(kxp,9)=0.
      a3(kxp,9)=0.
      i=2
      do 11 j=2,jx-1
      if(sfi1.eq.1.)sfi1=r(2,j)*b(2,j)/(r(1,j)*b(1,j))
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 11 m=2,8,3
      a1(k,m)=a1(k,m)+sfi1*a1(k,m-1)
      a2(k,m)=a2(k,m)+sfi1*a2(k,m-1)
      a3(k,m)=a3(k,m)+sfi1*a3(k,m-1)
 11   continue
 13   continue
      i=ix-1
      do 12 j=2,jx-1
c     bc(j)=0.
c     if(psi(j).lt.(psih-dpsih))bc(j)=1.
```

```
c      if(psi(j).ge.(psih-dpsih).and.psi(j).le.(psih+dpsih))
       bc(j)=.5*(tanh((-psi(j)+psih)/dpsih)+1)
       rbt=r(ix-1,j)*b(ix-1,j)
       rbt1=r(ix,j)*b(ix,j)
       down2=8.*bc(j)*uuzh(ix-1)/du+3.*(1.-bc(j))
       up2=1.-bc(j)
       gm2=up2/down2
       up1=-up2*rbt1*.75+bc(j)*rbt*uuzh(ix-1)/du
       down1=(bc(j)*uuzh(ix-1)/du+up2*3./8.)*rbt1
       gm1=up1/down1
       delco1(j)=gm1
       delco2(j)=gm2
       up=(bc(j)*r(ix-1,j)*b(ix-1,j)*uuzh(ix-1)/du+rbt*(bc(j)-1.))
       down=(bc(j)*r(ix,j)*b(ix,j)*uuzh(ix-1)/du+rbt*(-bc(j)+1.))
       sfizx=up/down
       k1=i-1+(j-2)*(ix-2)
       k2=j-1+(jx-2)*(i-2)
       k=.5*(1+isw)*k1+.5*(1-isw)*k2
       do 12 m=2,8,3
       a1(k,m)=a1(k,m)+gm1*a1(k,m+1)
       a1(k,m-1)=a1(k,m-1)+gm2*a1(k,m+1)
       a2(k,m)=a2(k,m)+gm1*a2(k,m+1)
       a2(k,m-1)=a2(k,m-1)+gm2*a2(k,m+1)
       a3(k,m)=a3(k,m)+gm1*a3(k,m+1)
       a3(k,m-1)=a3(k,m-1)+gm2*a3(k,m+1)
  12   continue
  20   continue
       i=2
       do 21  j=2,jx-1
       k1=i-1+(j-2)*(ix-2)
       k2=j-1+(jx-2)*(i-2)
       k=.5*(1+isw)*k1+.5*(1-isw)*k2
       do 21  m=1,7,3
       a1(k,m)=0.
       a2(k,m)=0.
       a3(k,m)=0.
  21   continue
       i=ix-1
       do 22 j=2,jx-1
       k1=i-1+(j-2)*(ix-2)
       k2=j-1+(jx-2)*(i-2)
       k=.5*(1+isw)*k1+.5*(1-isw)*k2
       do 22 m=3,9,3
       a1(k,m)=0.
       a2(k,m)=0.
       a3(k,m)=0.
  22   continue
       j=2
       do 31 i=2,ix-1
       k1=i-1+(j-2)*(ix-2)
       k2=j-1+(jx-2)*(i-2)
       k=.5*(1+isw)*k1+.5*(1-isw)*k2
       do 30 m=4,6
       a1(k,m)=a1(k,m)+sfj1*a1(k,m-3)
       a2(k,m)=a2(k,m)+sfj1*a2(k,m-3)
       a3(k,m)=a3(k,m)+sfj1*a3(k,m-3)
  30   continue
       b1(k,2)=b1(k,2)+sfj1*b1(k,1)
  31   continue
  32   continue
       j=jx-1
       fac5=sfjrx
       if(sfjrx.eq.-1)fac5=fac5*dvout/dvin
       do 35 i=2,ix-1
```

```
1297          k1=i-1+(j-2)*(ix-2)
1298          k2=j-1+(jx-2)*(i-2)
1299          k=.5*(1+isw)*k1+.5*(1-isw)*k2
1300          do 34 m=4,6
1301          a1(k,m)=a1(k,m)+fac5*a1(k,m+3)
1302          a2(k,m)=a2(k,m)+fac5*a2(k,m+3)
1303          a3(k,m)=a3(k,m)+fac5*a3(k,m+3)
1304    34    continue
1305          b1(k,2)=b1(k,2)+fac5*b1(k,3)
1306    35    continue
1307    40    continue
1308          j=2
1309          do 45 i=2,ix-1
1310          k1=i-1+(j-2)*(ix-2)
1311          k2=j-1+(jx-2)*(i-2)
1312          k=.5*(1+isw)*k1+.5*(1-isw)*k2
1313          do 44 m=4,6
1314          a1(k,m-3)=0.
1315          a2(k,m-3)=0.
1316          a3(k,m-3)=0.
1317     44   continue
1318          b1(k,1)=0.
1319     45   continue
1320          j=jx-1
1321          do 48 i=2,ix-1
1322          k1=i-1+(j-2)*(ix-2)
1323          k2=j-1+(jx-2)*(i-2)
1324          k=.5*(1+isw)*k1+.5*(1-isw)*k2
1325          do 47 m=4,6
1326          a1(k,m+3)=0.
1327          a3(k,m+3)=0.
1328          a2(k,m+3)=0.
1329     47   continue
1330          b1(k,3)=0.
1331     48   continue
1332          return
1333          end
```

```
1334
1335 c*********************************************************************
1336       subroutine bvcal(bs,zs,as,als,z,n,zcc,bcc,znorm,bv,bvp,bvpp,
1337      1   b4,b5,z4,z5,nc)
1338       dimension bs(1),zs(1),as(1),als(1),z(1),bv(1),bvp(1),bvpp(1)
1339       call bcccal(bs,zs,as,als,z,n,bv,bvp,bvpp,b4,b5,z4,z5,nc)
1340       do 15 i=1,n
1341       bcorr=bcc
1342       bcorrp=0.
1343       bcorrpp=0.
1344       tanhyp=tanh((z(i)-zcc)/znorm)
1345        bcorrp=-bcc*.5*(1.- tanhyp**2)/znorm
1346        bcorrpp=+bcc*tanhyp*(1.-tanhyp**2)/znorm**2
1347       bcorr=bcc*.5*(1.-tanhyp)
1348  18   bv(i)=bv(i)+bcorr
1349       bvp(i)=bvp(i)+bcorrp
1350       bvpp(i)=bvpp(i)+bcorrpp
1351       if(z(i).eq.z4)b4=b4+bcorr
1352       if(z(i).eq.z5)b5=b5+bcorr
1353  15   continue
1354       return
1355       end
```

```
1356 c*********************************************************************
1357       subroutine bcccal(bs,zs,as,als,z,n,bv,bvp,bvpp,b4,b5,z4,z5,nc)
1358       dimension z(n),bv(n),bvp(n),bvpp(n)
1359       dimension bs(3),zs(3),as(3),als(3),alpha(3,3),ak(3),is(3)
1360
1361 c....compute matrix elements
1362
1363       do 10 j=1,nc
1364       do 10 i=1,nc
1365       alpha(i,j)=bfun(zs(i),zs(j),als(j),as(j),0)
1366  10   continue
1367
1368 c....determinant
1369       if(nc.eq.2)alpha(3,3)=1.
1370
1371       det=alpha(1,1)*(alpha(2,2)*alpha(3,3)-alpha(3,2)*alpha(2,3))
1372      1   -alpha(1,2)*(alpha(2,1)*alpha(3,3)-alpha(3,1)*alpha(2,3))
1373      2   +alpha(1,3)*(alpha(2,1)*alpha(3,2)-alpha(3,1)*alpha(2,2))
1374
1375 c....solution
1376
1377       ak(1)=(bs(1)*(alpha(2,2)*alpha(3,3)-alpha(3,2)*alpha(2,3))
1378      1       -bs(2)*(alpha(1,2)*alpha(3,3)-alpha(3,2)*alpha(1,3))
1379      2       +bs(3)*(alpha(1,2)*alpha(2,3)-alpha(2,2)*alpha(1,3)))
1380      3       /det
1381       ak(2)=(-bs(1)*(alpha(2,1)*alpha(3,3)-alpha(3,1)*alpha(2,3))
1382      1        +bs(2)*(alpha(1,1)*alpha(3,3)-alpha(3,1)*alpha(1,3))
1383      2        -bs(3)*(alpha(1,1)*alpha(2,3)-alpha(2,1)*alpha(1,3)))
1384      3       /det
1385       ak(3)=(bs(1)*(alpha(1,2)*alpha(2,3)-alpha(1,3)*alpha(2,2))
1386      1       -bs(2)*(alpha(1,1)*alpha(2,3)-alpha(1,3)*alpha(2,1))
1387      2       +bs(3)*(alpha(1,1)*alpha(2,2)-alpha(1,2)*alpha(2,1)))
1388      3       /det
1389
1390 c....fields and derivatives
1391
1392       do 50 i=1,n
1393       bv(i)=0.
1394       bvp(i)=0.
1395       bvpp(i)=0.
1396       do 50 j=1,nc
1397       bv(i)=bv(i)+ak(j)*bfun(z(i),zs(j),als(j),as(j),0)
1398       bvp(i)=bvp(i)+ak(j)*bfun(z(i),zs(j),als(j),as(j),1)
1399       bvpp(i)=bvpp(i)+ak(j)*bfun(z(i),zs(j),als(j),as(j),2)
1400  50   continue
1401
1402 c....minima and their positions
1403
1404       do 70 j=1,nc
1405       do 60 i=2,n
1406       if(zs(j).lt.z(i)) go to 63
1407  60   continue
1408  63   is(j)=i-1
1409  70   continue
1410       b4=aminaf(bv,is(1),is(2),1,i4,amin)
1411       z4=z(i4)
1412       if(ncoil.eq.2)return
1413       b5=aminaf(bv,is(2),is(3),1,i5,amin)
1414       z5=z(i5)
1415
1416       return
1417       end
```

```
1418 c*************************************************************************
1419       function bfun(z,zx,al,a,ind)
1420       ind1=ind+1
1421       up=zx+.5*al
1422       um=zx-.5*al
1423       go to (10,20,30),ind1
1424  10   t1=gfun(z,up,al)
1425       t2=gfun(z,um,al)
1426       go to 40
1427  20   t1=gfun1(z,up,al)
1428       t2=gfun1(z,um,al)
1429       go to 40
1430  30   t1=gfun2(z,up,al)
1431       t2=gfun2(z,um,al)
1432  40   bfun=(t1-t2)/a**2
1433       return
1434       end
```

```
1435 c*******************************************************************************
1436       function gfun(z,u,a)
1437       x1=u-z
1438       x2=sqrt(x1**2+a**2)
1439       gfun=x1/x2
1440       return
1441       end
```