F11DKF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

F11DKF computes the **approximate** solution of a real, symmetric or non-symmetric, sparse system of linear equations applying a number of Jacobi iterations. It is expected that F11DKF will be used as a preconditioner for the iterative solution of real sparse systems of equations.

2 Specification

```
 SUBROUTINE F11DKF(STORE, TRANS, INIT, NITER, N, NNZ, A, IROW,
1                  ICOL, CHECK, B, X, DIAG, WORK, IFAIL)
 INTEGER           NITER, N, NNZ, IROW(NNZ), ICOL(NNZ), IFAIL
 real              A(NNZ), B(N), X(N), DIAG(N), WORK(N)
 CHARACTER*1       STORE, TRANS, INIT, CHECK
```

3 Description

F11DKF computes the **approximate** solution of the real sparse system of linear equations $Ax = b$ using NITER iterations of the Jacobi algorithm (see also Golub and van Loan [1], Young [2]):

$$x_{k+1} = D^{-1}(b - Ax_k + D) \tag{1}$$

where $k = 1, \ldots,$NITER and $x_0 = 0$.

F11DKF can be used both for non-symmetric and symmetric systems of equations. For symmetric matrices, either all non-zero elements of the matrix A can be supplied using coordinate storage (CS), or only the non-zero elements of the lower triangle of A, using symmetric coordinate storage (SCS) (see Section 2.1.1 of the Chapter Introduction).

It is expected that F11DKF will be used as a preconditioner for the iterative solution of real sparse systems of equations, using either the suite comprising the routines F11GDF, F11GEF and F11GFF, for symmetric systems, or the suite comprising the routines F11BDF, F11BEF and F11BFF, for non-symmetric systems of equations.

4 References

[1] Golub G H and van Loan C F (1996) *Matrix Computations* Johns Hopkins University Press (3rd Edition), Baltimore

[2] Young D (1971) *Iterative Solution of Large Linear Systems* Academic Press, New York

5 Parameters

1: STORE — CHARACTER*1 *Input*

On entry: specifies whether the matrix A is stored using symmetric coordinate storage (SCS), applicable only to a symmetric matrix A, or coordinate storage (CS) applicable to both symmetric and non-symmetric matrices.

If STORE = 'N', then the complete matrix A is stored in CS format;
If STORE = 'S', then the lower triangle of the symmetric matrix A is stored in SCS format;

Constraint: STORE = 'N' or 'S'.

2: TRANS — CHARACTER*1 *Input*

On entry: if STORE = 'N', specifies whether the approximate solution of $Ax = b$ or of $A^T x = b$ is required:

if TRANS = 'N', then $Ax = b$ is iterated;
if TRANS = 'T', then $A^T x = b$ is iterated;

Constraint: TRANS = 'N' or 'T'.

Suggested value: if the matrix A is symmetric and stored in CS format, it is recommended that TRANS = 'N' for reasons of efficiency.

3: INIT — CHARACTER*1 *Input*

On entry: on first entry, INIT should be set to 'I', unless the diagonal elements of A are already stored in the array $DIAG$. Otherwise, if DIAG already contains the diagonal of A, it can be set to 'N'.

if INIT = 'N', then DIAG must contain the diagonal of A;
if INIT = 'I', then DIAG will store on exit the diagonal of A.

Constraint: INIT = 'N' or 'I'.

Suggested value: INIT = 'I' on first entry; INIT= 'N', subsequently, unless DIAG has been overwritten.

4: NITER — INTEGER *Input*

On entry: the number of Jacobi iterations requested.

Constraint: NITER > 0.

5: N — INTEGER *Input*

On entry: n, the order of the matrix A.

Constraint: N $\geq$ 1.

6: NNZ — INTEGER *Input*

On entry: if STORE = 'N', the number of non-zero elements in the matrix A; if STORE = 'S', the number of non-zero elements in the lower triangle of the matrix A.

Constraint: if STORE = 'N', $1 \leq$ NNZ $\leq$ N^2; if STORE = 'S', $1 \leq$ NNZ$\leq$ [N(N + 1)]/2;

7: A(NNZ) — ***real*** array *Input*

On entry: if STORE = 'N', the non-zero elements in the matrix A (CS format); if STORE = 'S', the non-zero elements in the lower triangle of the matrix A (SCS format). In both cases, the elements of either A or of its lower triangle must be ordered by increasing row index and by increasing column index within each row. Multiple entries for the same row and columns indices are not permitted. The routine F11ZAF or F11ZBF may be used to reorder the elements in this way for CS and SCS storage, respectively.

8: IROW(NNZ) — INTEGER array *Input*
9: ICOL(NNZ) — INTEGER array *Input*

On entry: if STORE = 'N', the row and column indices of the non-zero elements supplied in A; if STORE = 'S', the row and column indices of the non-zero elements of the lower triangle of the matrix A supplied in A.

Constraints: $1 \leq$ IROW(i) $\leq$ N, for $i = 1, 2, \ldots,$ NNZ;
$1 \leq$ ICOL(i) $\leq$ N for $i = 1, 2, \ldots,$ NNZ, when STORE = 'N', or
$1 \leq$ ICOL(i) $\leq$ IROW(i), for $i = 1, 2, \ldots,$ NNZ, when STORE = 'S';
IROW($i - 1$) < IROW(i) or
IROW($i - 1$) = IROW(i) and ICOL($i - 1$) < ICOL(i) for $i = 2, 3, \ldots,$ NNZ.

10: CHECK — CHARACTER*1 *Input*

On entry: specifies whether or not the CS representation of the matrix A should be checked:

if CHECK = 'C', checks are carried on the values of N, NNZ, IROW, ICOL and, if on entry INIT = 'N' DIAG:
if CHECK = 'N', none of these checks are carried out.

See also Section 8.2.

Constraint: CHECK = 'C' or 'N'.

11: B(N) — ***real*** array *Input*

On entry: the right-hand side vector b.

12: X(N) — ***real*** array *Output*

On exit: the approximate solution vector x_{NITER}.

13: DIAG(N) — ***real*** array *Input/Output*

On entry: if INIT = 'N', the diagonal elements of A.

On exit:

if INIT = 'N', unchanged on exit;
if INIT = 'I', the diagonal elements of A.

14: WORK (N) — ***real*** array *Workspace*

15: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, STORE ≠ 'N' or 'S',
or TRANS ≠ 'N' or 'T',
or INIT ≠ 'N' or 'I',
or CHECK ≠ 'C' or 'N'.
or NITER ≤ 0.

IFAIL = 2

On entry, N < 1,
or NNZ < 1,
or NNZ > N^2, if STORE = 'N',
or 1 ≤ NNZ ≤ [N(N + 1)]/2, if STORE = 'S'.

IFAIL = 3

On entry, the arrays IROW and ICOL fail to satisfy the following constraints:

$1 \le \mathrm{IROW}(i) \le \mathrm{N}$ and
  if STORE = 'N' then $1 \le \mathrm{ICOL}(i) \le \mathrm{N}$, or
  if STORE = 'S' then $1 \le \mathrm{ICOL}(i) \le \mathrm{IROW}(i)$
    for $i = 1, 2, \ldots$,NNZ.
$\mathrm{IROW}(i-1) < \mathrm{IROW}(i)$, or
$\mathrm{IROW}(i-1) = \mathrm{IROW}(i)$ and $\mathrm{ICOL}(i-1) < \mathrm{ICOL}(i)$
  for $i = 2, 3, \ldots$,NNZ.

Therefore a non-zero element has been supplied which does not lie within the matrix A, is out of order, or has duplicate row and column indices. Call either F11ZAF or F11ZBF to reorder and sum or remove duplicates when STORE = 'N' or STORE= 'S', respectively.

IFAIL = 4

INIT = 'N', and some diagonal elements of A stored in $DIAG$ are zero.

IFAIL = 5

INIT = 'I', and some diagonal elements of A are zero.

7 Accuracy

In general, the Jacobi method cannot be used on its own to solve systems of linear equations. The rate of convergence is bound by its spectral properties (see, for example, Golub and van Loan [1]) and as a solver, the Jacobi method can only be applied to a limited set of matrices. One condition that guarantees convergence is strictly diagonal dominance.

However, the Jacobi method can used sucessfully as a preconditioner to a wider class of systems of equations. The Jacobi method has good vector/parallel properties, hence it can be applied very efficiently. Unfortunately, it is not possible to provide criteria which define the applicability of the Jacobi method as a preconditioner, and its usefulness must be judged for each case.

8 Further Comments

8.1 Timing

The time taken for a call to F11DKF is proportional to NITER$\times$NNZ.

8.2 Use of CHECK

It is expected that a common use of F11DKF will be as preconditioner for the iterative solution of real, symmetric or non-symmetric, linear systems. In this situation, F11DKF is likely to be called many times. In the interests of both reliability and efficiency, you are recommended to set CHECK to 'C' for the first of such calls, and to 'N' for all subsequent calls.

9 Example

This example solves iteratively the real sparse non-symmetric system of equations $Ax = b$ using F11DKF as a preconditioner.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     F11DKF Example Program Text.
*     NAG Fortran SMP Library, Release 2.  NAG Copyright 2000.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LA, LWORK
      PARAMETER        (NMAX=1000,LA=10000,LWORK=10*NMAX)
*     .. Local Scalars ..
      DOUBLE PRECISION ANORM, SIGMAX, STPLHS, STPRHS, TOL
      INTEGER          I, IFAIL, IFAIL1, IFAILX, IREVCM, ITERM, ITN,
     +                 LWREQ, M, MAXITN, MONIT, N, NITER, NNZ
      LOGICAL          LOOP
      CHARACTER        INIT, NORM, PRECON, WEIGHT
      CHARACTER*8      METHOD
*     .. Local Arrays ..
      DOUBLE PRECISION A(LA), B(NMAX), DIAG(NMAX), WGT(NMAX),
     +                 WORK(LWORK), X(NMAX)
      INTEGER          ICOL(LA), IROW(LA)
*     .. External Subroutines ..
      EXTERNAL         F11BDF, F11BEF, F11BFF, F11DKF, F11XAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F11DKF Example Program Results'
*
*     Skip heading in data file
*
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
*
*        Read or initialize the parameters for the iterative solver
*
         READ (NIN,*) METHOD
         READ (NIN,*) PRECON, NORM, WEIGHT, ITERM
         READ (NIN,*) M, TOL, MAXITN
         READ (NIN,*) MONIT
         ANORM = 0.0D0
         SIGMAX = 0.0D0
*
*        Read the parameters for the preconditioner
*
         READ (NIN,*) NITER
*
*        Read the number of non-zero elements of the matrix A, then read
*        the non-zero elements
*
         READ (NIN,*) NNZ
         DO 20 I = 1, NNZ
            READ (NIN,*) A(I), IROW(I), ICOL(I)
   20    CONTINUE
*
*        Read right-hand side vector b and initial approximate solution
*
         READ (NIN,*) (B(I),I=1,N)
         READ (NIN,*) (X(I),I=1,N)
```

```
*
*        Call F11BDF to initialize the solver
*
         IFAIL = 0
         CALL F11BDF(METHOD,PRECON,NORM,WEIGHT,ITERM,N,M,TOL,MAXITN,
     +               ANORM,SIGMAX,MONIT,LWREQ,WORK,LWORK,IFAIL)
*
*        Call repeatedly F11BEF to solve the equations
*        Note that the arrays B and X are overwritten
*
*        On final exit, X will contain the solution and B the residual
*        vector
*
         IFAIL = 0
         IREVCM = 0
         LOOP = .TRUE.
         INIT = 'I'
*
   40    CONTINUE
         CALL F11BEF(IREVCM,X,B,WGT,WORK,LWREQ,IFAIL)
         IF (IREVCM.EQ.-1) THEN
            IFAILX = -1
            CALL F11XAF('Transpose',N,NNZ,A,IROW,ICOL,'No checking',X,B,
     +                  IFAILX)
         ELSE IF (IREVCM.EQ.1) THEN
            IFAILX = -1
            CALL F11XAF('No transpose',N,NNZ,A,IROW,ICOL,'No checking',
     +                  X,B,IFAILX)
         ELSE IF (IREVCM.EQ.2) THEN
            IFAIL1 = -1
            CALL F11DKF('Non symmetric','N',INIT,NITER,N,NNZ,A,IROW,
     +                  ICOL,'No checking',X,B,DIAG,WORK(LWREQ+1),
     +                  IFAIL1)
            INIT = 'N'
            IF (IFAIL1.NE.0) IREVCM = 6
         ELSE IF (IREVCM.EQ.3) THEN
            IFAIL1 = 0
            CALL F11BFF(ITN,STPLHS,STPRHS,ANORM,SIGMAX,WORK,LWREQ,
     +                  IFAIL1)
            WRITE (NOUT,99999) ITN, STPLHS
         ELSE IF (IREVCM.EQ.4) THEN
            LOOP = .FALSE.
         END IF
         IF (LOOP) GO TO 40
*
*        Obtain information about the computation
*
         IFAIL1 = 0
         CALL F11BFF(ITN,STPLHS,STPRHS,ANORM,SIGMAX,WORK,LWREQ,IFAIL1)
*
*        Print the output data
*
         WRITE (NOUT,99996)
         WRITE (NOUT,99995)
     +     'Number of iterations for convergence:    ', ITN
         WRITE (NOUT,99994)
     +     'Residual norm:                           ', STPLHS
         WRITE (NOUT,99994)
     +     'Right-hand side of termination criterion:', STPRHS
           WRITE (NOUT,99994)
     +     '1-norm of matrix A:                     ', ANORM
*
*        Output x
*
           WRITE (NOUT,99998)
           WRITE (NOUT,99997) (X(I),B(I),I=1,N)
      END IF
      STOP
*
99999 FORMAT (/1X,'Monitoring at iteration no.',I4,/1X,1P,'residual no',
     +       'rm: ',D14.4)
99998 FORMAT (/2X,'  Solution vector',2X,' Residual vector')
99997 FORMAT (1X,1P,D16.4,1X,D16.4)
99996 FORMAT (/1X,'Final Results')
99995 FORMAT (1X,A,I4)
99994 FORMAT (1X,A,1P,D14.4)
      END
```

9.2 Program Data

```
F11DKF Example Program Data
  8                        N
  'BICGSTAB'               METHOD
  'P'  '1'  'N'  1         PRECON, NORM, WEIGHT, ITERM
  2  1.0D-8   20           M, TOL, MAXITN
  1                        MONIT
  4                        NITER
 24                        NNZ
  4.   1    1
 -1.   1    4
  1.   1    8
  4.   2    1
 -5.   2    2
  2.   2    5
 -7.   3    3
  2.   3    6
  2.   4    1
 -1.   4    3
  6.   4    4
  2.   4    7
 -1.   5    2
  8.   5    5
 -2.   5    7
 -2.   6    1
  5.   6    3
  8.   6    6
 -2.   7    3
 -1.   7    5
  7.   7    7
 -1.   8    2
  2.   8    6
  6.   8    8         A(I), IROW(I), ICOL(I), I=1,...,NNZ
  6.   8.  -9.  46.
 17.  21.  22.  34.   B(I), I=1,...,N
  0.   0.   0.   0.
  0.   0.   0.   0.   X(I), I=1,...,N
```

9.3 Program Results

```
F11DKF Example Program Results

Monitoring at iteration no.   2
residual norm:     1.1177D-04

Final Results
Number of iterations for convergence:       4
Residual norm:                               6.2630D-11
Right-hand side of termination criterion:    4.6526D-06
1-norm of matrix A:                          1.2000D+01

   Solution vector   Residual vector
      1.7035D+00      -7.6819D-12
      1.0805D+00       1.6339D-11
      1.8305D+00      -1.1637D-11
      6.0251D+00       1.2434D-11
      3.2942D+00       9.0949D-13
      1.9068D+00      -5.3006D-12
      4.1365D+00       2.7001D-13
      5.2111D+00      -8.0576D-12
```