NAG Library Routine Document

F11DKF

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 nonsymmetric, 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, ICOL,
     1                  CHECK, B, X, DIAG, WORK, IFAIL)
      INTEGER           NITER, N, NNZ, IROW(NNZ), ICOL(NNZ), IFAIL
      double precision  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 (1996) and Young (1971)):

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

where $k = 1, 2, \ldots,$NITER and $x_0 = 0$.

F11DKF can be used both for nonsymmetric and symmetric systems of equations. For symmetric matrices, either all nonzero elements of the matrix A can be supplied using co-ordinate storage (CS), or only the nonzero elements of the lower triangle of A, using symmetric co-ordinate storage (SCS) (see the F11 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 nonsymmetric systems of equations.

4 References

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

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 co-ordinate storage (SCS) (applicable only to a symmetric matrix A) or co-ordinate storage (CS) (applicable to both symmetric and nonsymmetric matrices).

STORE = 'N'
: The complete matrix A is stored in CS format.

STORE = 'S'
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 $A^{\mathrm{T}}x = b$ is required.

TRANS = 'N'
The approximate solution of $Ax = b$ is calculated.

TRANS = 'T'
The approximate solution of $A^{\mathrm{T}}x = b$ is calculated.

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'.

INIT = 'N'
DIAG must contain the diagonal of A.

INIT = 'I'
DIAG will store the diagonal of A on exit.

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 ≥ 1.

6: NNZ – INTEGER *Input*

On entry: if STORE = 'N', the number of nonzero elements in the matrix A.

If STORE = 'S', the number of nonzero elements in the lower triangle of the matrix A.

Constraints:

if STORE = 'N', $1 \leq \mathrm{NNZ} \leq \mathrm{N}^2$;
if STORE = 'S', $1 \leq \mathrm{NNZ} \leq [\mathrm{N}(\mathrm{N}+1)]/2$.

7: A(NNZ) – ***double precision*** array *Input*

On entry: if STORE = 'N', the nonzero elements in the matrix A (CS format).

If STORE = 'S', the nonzero 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 nonzero elements supplied in A.

If STORE = 'S', the row and column indices of the nonzero elements of the lower triangle of the matrix A supplied in A.

Constraints:

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

10: CHECK – CHARACTER*1 *Input*

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

CHECK = 'C'
Checks are carried out on the values of N, NNZ, IROW, ICOL; if INIT = 'N', DIAG is also checked.

CHECK = 'N'
None of these checks are carried out.

See also Section 8.2.

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

11: B(N) – ***double precision*** array *Input*

On entry: the right-hand side vector b.

12: X(N) – ***double precision*** array *Output*

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

13: DIAG(N) – ***double precision*** 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) – ***double precision*** array *Workspace*

15: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

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

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

6 Error Indicators 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 or warnings detected by the routine:

IFAIL $= 1$

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

IFAIL $= 2$

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

IFAIL $= 3$

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

$1 \leq \mathrm{IROW}(i) \leq \mathrm{N}$ and

if STORE = 'N', then $1 \leq \mathrm{ICOL}(i) \leq \mathrm{N}$, or

if STORE = 'S', then $1 \leq \mathrm{ICOL}(i) \leq \mathrm{IROW}(i)$, for $i = 1, 2, \ldots, \mathrm{NNZ}$.

$\mathrm{IROW}(i-1) < \mathrm{IROW}(i)$ or $\mathrm{IROW}(i-1) = \mathrm{IROW}(i)$ and $\mathrm{ICOL}(i-1) < \mathrm{ICOL}(i)$, for $i = 2, 3, \ldots, \mathrm{NNZ}$.

Therefore a nonzero 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 $= 4$

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

IFAIL $= 5$

On entry, 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 (1996)) and as a solver, the Jacobi method can only be applied to a limited set of matrices. One condition that guarantees convergence is strict diagonal dominance.

However, the Jacobi method can be used successfully 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 nonsymmetric, 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 = 'C' for the first of such calls, and to set CHECK = 'N' for all subsequent calls.

9 Example

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

9.1 Program Text

```
*     F11DKF Example Program Text
*     Mark 20 Release. NAG Copyright 2001.
*     .. 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, IREVCM, ITERM, ITN, LWREQ, M,
     +                 MAXITN, MONIT, N, NITER, NNZ
      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.GT.NMAX) THEN
         WRITE (NOUT,'(A,I5)') ' N is too large, N =', N
         WRITE (NOUT,'(A,I5)') ' N must be less than ', NMAX
         GO TO 60
      END IF
*
*     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 = 1
      CALL F11BDF(METHOD,PRECON,NORM,WEIGHT,ITERM,N,M,TOL,MAXITN,ANORM,
     +            SIGMAX,MONIT,LWREQ,WORK,LWORK,IFAIL)
      IF (IFAIL.NE.0) THEN
         WRITE (NOUT,99993) IFAIL
         GO TO 60
      END IF
*
*     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 = 1
      IREVCM = 0
      INIT = 'I'
*
   40 CONTINUE
      CALL F11BEF(IREVCM,X,B,WGT,WORK,LWREQ,IFAIL)
      IF (IREVCM.NE.4) THEN
         IFAIL1 = -1
         IF (IREVCM.EQ.-1) THEN
            CALL F11XAF('Transpose',N,NNZ,A,IROW,ICOL,'No checking',X,B,
     +                  IFAIL1)
         ELSE IF (IREVCM.EQ.1) THEN
            CALL F11XAF('No transpose',N,NNZ,A,IROW,ICOL,'No checking',
     +                  X,B,IFAIL1)
         ELSE IF (IREVCM.EQ.2) THEN
            CALL F11DKF('Non symmetric','N',INIT,NITER,N,NNZ,A,IROW,
     +                  ICOL,'Check',X,B,DIAG,WORK(LWREQ+1),IFAIL1)
            INIT = 'N'
         ELSE IF (IREVCM.EQ.3) THEN
            IFAIL1 = 0
            CALL F11BFF(ITN,STPLHS,STPRHS,ANORM,SIGMAX,WORK,LWREQ,
     +                  IFAIL1)
            WRITE (NOUT,99999) ITN, STPLHS
         END IF
         IF (IFAIL1.NE.0) IREVCM = 6
         GO TO 40
      ELSE IF (IFAIL.NE.0) THEN
         WRITE (NOUT,99992) IFAIL
         GO TO 60
      END IF
*
*     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)
   60 CONTINUE
*
99999 FORMAT (/1X,'Monitoring at iteration no.',I4,/1X,1P,'residual no',
     +       'rm: ',E14.4)
99998 FORMAT (/2X,'  Solution vector',2X,' Residual vector')
99997 FORMAT (1X,1P,E16.4,1X,E16.4)
99996 FORMAT (/1X,'Final Results')
99995 FORMAT (1X,A,I4)
99994 FORMAT (1X,A,1P,E14.4)
99993 FORMAT (1X,/1X,' ** F11BDF returned with IFAIL = ',I5)
99992 FORMAT (1X,/1X,' ** F11BEF returned with IFAIL = ',I5)
      END
```

9.2 Program Data

```
F11DKF Example Program Data
  8                       N
  'BICGSTAB'              METHOD
  'P'  '1'  'N'  1        PRECON, NORM, WEIGHT, ITERM
  2  1.0D-6   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.   X(I), I=1,...,N
  0.   0.   0.   0.
  0.   0.   0.   0.   B(I), I=1,...,N
```

9.3 Program Results

```
 F11DKF Example Program Results

 Final Results
 Number of iterations for convergence:       2
 Residual norm:                               1.1177E-04
 Right-hand side of termination criterion:    5.4082E-04
 1-norm of matrix A:                          1.5000E+01

    Solution vector   Residual vector
       1.7035E+00       3.2377E-07
       1.0805E+00      -1.7625E-05
       1.8305E+00       2.7964E-05
```

```
      6.0251E+00      -2.5914E-05
      3.2942E+00       7.8156E-06
      1.9068E+00       9.2064E-06
      4.1365E+00      -3.0848E-06
      5.2111E+00       1.9834E-05
```
