# F11JDF – 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

F11JDF solves a system of linear equations involving the preconditioning matrix corresponding to SSOR applied to a real sparse symmetric matrix, represented in symmetric coordinate storage format.

## 2 Specification

```
      SUBROUTINE F11JDF(N, NNZ, A, IROW, ICOL, RDIAG, OMEGA, CHECK, Y,
     1                  X, IWORK, IFAIL)
      INTEGER           N, NNZ, IROW(NNZ), ICOL(NNZ), IWORK(N+1), IFAIL
      real              A(NNZ), RDIAG(N), OMEGA, Y(N), X(N)
      CHARACTER*1       CHECK
```

## 3 Description

This routine solves a system of equations:

$$Mx = y$$

involving the preconditioning matrix:

$$M = \frac{1}{\omega(2-\omega)}(D+\omega L)D^{-1}(D+\omega L)^T$$

corresponding to symmetric successive-over-relaxation (SSOR) [1] on a linear system $Ax = b$, where $A$ is a sparse symmetric matrix stored in symmetric coordinate storage (SCS) format (see Section 2.1.2 of the Chapter Introduction).

In the definition of $M$ given above $D$ is the diagonal part of $A$, $L$ is the strictly lower triangular part of $A$, and $\omega$ is a user-defined relaxation parameter.

It is envisaged that a common use of F11JDF will be to carry out the preconditioning step required in the application of F11GBF to sparse linear systems. For an illustration of this use of F11JDF see the example program given in Section 9.1. F11JDF is also used for this purpose by the black-box routine F11JEF.

## 4 References

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

## 5 Parameters

**1:** N — INTEGER *Input*

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

*Constraint:* $N \geq 1$.

**2:** NNZ — INTEGER *Input*

*On entry:* the number of non-zero elements in the lower triangular part of $A$.

*Constraint:* $1 \leq NNZ \leq N \times (N+1)/2$.

**3:** A(NNZ) — ***real*** array *Input*

*On entry:* the non-zero elements in the lower triangular part of the matrix $A$, ordered by increasing row index, and by increasing column index within each row. Multiple entries for the same row and column indices are not permitted. The routine F11ZBF may be used to order the elements in this way.

**4:** IROW(NNZ) — INTEGER array *Input*

**5:** ICOL(NNZ) — INTEGER array *Input*

*On entry:* the row and column indices of the non-zero elements supplied in A.

*Constraints:* IROW and ICOL must satisfy the following constraints (which may be imposed by a call to F11ZBF):

$1 \leq \text{IROW}(i) \leq \text{N}$, $1 \leq \text{ICOL}(i) \leq \text{IROW}(i)$, 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 $i = 2, 3, \ldots,\text{NNZ}$.

**6:** RDIAG(N) — ***real*** array *Input*

*On entry:* the elements of the diagonal matrix $D^{-1}$, where $D$ is the diagonal part of $A$.

**7:** OMEGA — ***real*** *Input*

*On entry:* the relaxation parameter $\omega$.

*Constraint:* $0.0 \leq \text{OMEGA} \leq 2.0$.

**8:** CHECK — CHARACTER*1 *Input*

*On entry:* specifies whether or not the input data should be checked:

if CHECK = 'C', checks are carried out on the values of N, NNZ, IROW, ICOL and OMEGA;
if CHECK = 'N', none of these checks are carried out.

See also Section 8.2.

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

**9:** Y(N) — ***real*** array *Input*

*On entry:* the right-hand side vector $y$.

**10:** X(N) — ***real*** array *Output*

*On exit:* the solution vector $x$.

**11:** IWORK(N+1) — INTEGER array *Workspace*

**12:** 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, CHECK ≠ 'C' or 'N'.

IFAIL = 2

On entry, N < 1,
or NNZ < 1,
or NNZ > N × (N+1)/2,
or OMEGA lies outside the interval [0.0,2.0],

IFAIL = 3

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

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

Therefore a non-zero element has been supplied which does not lie in the lower triangular part of $A$, is out of order, or has duplicate row and column indices. Call F11ZBF to reorder and sum or remove duplicates.

# 7 Accuracy

The computed solution $x$ is the exact solution of a perturbed system of equations $(M + \delta M)x = y$, where

$$|\delta M| \leq c(n)\epsilon|D + \omega L||D^{-1}||(D + \omega L)^T|,$$

$c(n)$ is a modest linear function of $n$, and $\epsilon$ is the ***machine precision***.

# 8 Further Comments

## 8.1 Timing

The time taken for a call to F11JDF is proportional to NNZ.

## 8.2 Use of CHECK

It is expected that a common use of F11JDF will be to carry out the preconditioning step required in the application of F11GBF to sparse symmetric linear systems. In this situation F11JDF is likely to be called many times with the same matrix $M$. 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 program solves a sparse symmetric linear system of equations:

$$Ax = b,$$

using the conjugate-gradient (CG) method with SSOR preconditioning.

The CG algorithm itself is implemented by the reverse communication routine F11GBF, which returns repeatedly to the calling program with various values of the parameter IREVCM. This parameter indicates the action to be taken by the calling program.

If IREVCM = 1 a matrix-vector product $v = Au$ is required. This is implemented by a call to F11XEF.

If IREVCM = 2 a solution of the preconditioning equation $Mv = u$ is required. This is achieved by a call to F11JDF.

If IREVCM = 4 F11GBF has completed its tasks. Either the iteration has terminated, or an error condition has arisen.

For further details see the routine document for F11GBF.

## 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.

```
*     F11JDF Example Program Text
*     Mark 19 Revised. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LA, LIWORK, LWORK
      PARAMETER        (NMAX=1000,LA=10000,LIWORK=NMAX+1,
     +                 LWORK=6*NMAX+120)
*     .. Local Scalars ..
      real             ANORM, OMEGA, SIGERR, SIGMAX, SIGTOL, STPLHS,
     +                 STPRHS, TOL
      INTEGER          I, IFAIL, IREVCM, ITERM, ITN, ITS, LWNEED,
     +                 MAXITN, MAXITS, MONIT, N, NNZ
      CHARACTER        CKJDF, CKXEF, NORM, PRECON, SIGCMP, WEIGHT
      CHARACTER*6      METHOD
*     .. Local Arrays ..
      real             A(LA), B(NMAX), RDIAG(NMAX), WGT(NMAX),
     +                 WORK(LWORK), X(NMAX)
      INTEGER          ICOL(LA), IROW(LA), IWORK(LIWORK)
*     .. External Subroutines ..
      EXTERNAL         F11GDF, F11GEF, F11GFF, F11JDF, F11XEF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F11JDF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
*
*     Read algorithmic parameters
*
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
         READ (NIN,*) NNZ
         READ (NIN,*) METHOD
         READ (NIN,*) PRECON, SIGCMP, NORM, ITERM
         READ (NIN,*) TOL, MAXITN
         READ (NIN,*) ANORM, SIGMAX
         READ (NIN,*) SIGTOL, MAXITS
         READ (NIN,*) OMEGA
*
*     Read the matrix A
*
         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 x
*
         READ (NIN,*) (B(I),I=1,N)
         READ (NIN,*) (X(I),I=1,N)
*
*     Call F11GDF to initialize solver
*
         WEIGHT = 'N'
         MONIT = 0
         IFAIL = 0
         CALL F11GDF(METHOD,PRECON,SIGCMP,NORM,WEIGHT,ITERM,N,TOL,
     +               MAXITN,ANORM,SIGMAX,SIGTOL,MAXITS,MONIT,LWNEED,
     +               WORK,LWORK,IFAIL)
*
*     Calculate reciprocal diagonal matrix elements.
*
         DO 40 I = 1, N
            IWORK(I) = 0
   40    CONTINUE
*
         DO 60 I = 1, NNZ
            IF (IROW(I).EQ.ICOL(I)) THEN
               IWORK(IROW(I)) = IWORK(IROW(I)) + 1
               IF (A(I).NE.0.0e0) THEN
                  RDIAG(IROW(I)) = 1.0e0/A(I)
               ELSE
                  WRITE (NOUT,*) 'Matrix has a zero diagonal element'
                  GO TO 140
               END IF
            END IF
   60    CONTINUE
*
         DO 80 I = 1, N
            IF (IWORK(I).EQ.0) THEN
               WRITE (NOUT,*) 'Matrix has a missing diagonal element'
               GO TO 140
            END IF
            IF (IWORK(I).GE.2) THEN
               WRITE (NOUT,*) 'Matrix has a multiple diagonal element'
               GO TO 140
            END IF
   80    CONTINUE
*
*     Call F11GEF to solve the linear system
*
         IREVCM = 0
         CKXEF = 'C'
         CKJDF = 'C'
*
  100    CONTINUE
*
         CALL F11GEF(IREVCM,X,B,WGT,WORK,LWORK,IFAIL)
*
         IF (IREVCM.EQ.1) THEN
*
*        Compute matrix vector product
*
            CALL F11XEF(N,NNZ,A,IROW,ICOL,CKXEF,X,B,IFAIL)
```

```
            CKXEF = 'N'
            GO TO 100
*
         ELSE IF (IREVCM.EQ.2) THEN
*
*         SSOR preconditioning
*
            CALL F11JDF(N,NNZ,A,IROW,ICOL,RDIAG,OMEGA,CKJDF,X,B,IWORK,
     +                  IFAIL)
            CKJDF = 'N'
            GO TO 100
*
         ELSE IF (IREVCM.EQ.4) THEN
*
*         Termination
*
            CALL F11GFF(ITN,STPLHS,STPRHS,ANORM,SIGMAX,ITS,SIGERR,WORK,
     +                  LWORK,IFAIL)
*
            WRITE (NOUT,99999) 'Converged in', ITN, ' iterations'
            WRITE (NOUT,99998) 'Final residual norm =', STPLHS
*
*         Output x
*
            DO 120 I = 1, N
               WRITE (NOUT,99997) X(I)
  120       CONTINUE
*
         END IF
*
  140    CONTINUE
      END IF
      STOP
*
99999 FORMAT (1X,A,I10,A)
99998 FORMAT (1X,A,1P,e16.3)
99997 FORMAT (1X,1P,e16.4)
      END
```

## 9.2 Program Data

```
F11JDF Example Program Data
  7                    N
 16                    NNZ
 'CG'                  METHOD
 'P' 'N' 'I' 1         PRECON, SIGCMP, NORM, ITERM
 1.0E-6 100            TOL, MAXITN
 0.0E0 0.0E0           ANORM, SIGMAX
 0.0E0 10              SIGTOL, MAXITS
 1.0E0                 OMEGA
  4.   1    1
  1.   2    1
  5.   2    2
  2.   3    3
  2.   4    2
  3.   4    4
 -1.   5    1
  1.   5    4
```

```
  4.   5    5
  1.   6    2
 -2.   6    5
  3.   6    6
  2.   7    1
 -1.   7    2
 -2.   7    3
  5.   7    7         A(I), IROW(I), ICOL(I), I=1,...,NNZ
 15.  18.  -8.   21.
 11.  10.   29.       B(I), I=1,...,N
  0.   0.   0.    0.
  0.   0.   0.        X(I), I=1,...,N
```

## 9.3 Program Results

```
F11JDF Example Program Results
Converged in         6 iterations
Final residual norm =       7.105E-15
      1.0000E+00
      2.0000E+00
      3.0000E+00
      4.0000E+00
      5.0000E+00
      6.0000E+00
      7.0000E+00
```