

NAG Library Routine Document

F11JAF

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

F11JAF computes an incomplete Cholesky factorization of a real sparse symmetric matrix, represented in symmetric coordinate storage format. This factorization may be used as a preconditioner in combination with F11GEF or F11JCF.

2 Specification

```

SUBROUTINE F11JAF (N, NNZ, A, LA, IROW, ICOL, LFILL, DTOL, MIC, DSCALE,      &
                  PSTRAT, IPIV, ISTR, NNZC, NPIVM, IWORK, LIWORK,        &
                  IFAIL)
INTEGER              N, NNZ, LA, IROW(LA), ICOL(LA), LFILL, IPIV(N),      &
                  ISTR(N+1), NNZC, NPIVM, IWORK(LIWORK), LIWORK,        &
                  IFAIL
REAL (KIND=nag_wp)  A(LA), DTOL, DSCALE
CHARACTER(1)        MIC, PSTRAT

```

3 Description

F11JAF computes an incomplete Cholesky factorization (see Meijerink and Van der Vorst (1977)) of a real sparse symmetric n by n matrix A . It is designed specifically for positive definite matrices, but may also work for some mildly indefinite cases. The factorization is intended primarily for use as a preconditioner with one of the symmetric iterative solvers F11GEF or F11JCF.

The decomposition is written in the form

$$A = M + R$$

where

$$M = PLDL^T P^T$$

and P is a permutation matrix, L is lower triangular with unit diagonal elements, D is diagonal and R is a remainder matrix.

The amount of fill-in occurring in the factorization can vary from zero to complete fill, and can be controlled by specifying either the maximum level of fill LFILL, or the drop tolerance DTOL. The factorization may be modified in order to preserve row sums, and the diagonal elements may be perturbed to ensure that the preconditioner is positive definite. Diagonal pivoting may optionally be employed, either with a user-defined ordering, or using the Markowitz strategy (see Markowitz (1957)), which aims to minimize fill-in. For further details see Section 9.

The sparse matrix A is represented in symmetric coordinate storage (SCS) format (see Section 2.1.2 in the F11 Chapter Introduction). The array A stores all the nonzero elements of the lower triangular part of A , while arrays IROW and ICOL store the corresponding row and column indices respectively. Multiple nonzero elements may not be specified for the same row and column index.

The preconditioning matrix M is returned in terms of the SCS representation of the lower triangular matrix

$$C = L + D^{-1} - I.$$

4 References

Chan T F (1991) Fourier analysis of relaxed incomplete factorization preconditioners *SIAM J. Sci. Statist. Comput.* **12**(2) 668–680

Markowitz H M (1957) The elimination form of the inverse and its application to linear programming *Management Sci.* **3** 255–269

Meijerink J and Van der Vorst H (1977) An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix *Math. Comput.* **31** 148–162

Salvini S A and Shaw G J (1995) An evaluation of new NAG Library solvers for large sparse symmetric linear systems *NAG Technical Report TRI/95*

Van der Vorst H A (1990) The convergence behaviour of preconditioned CG and CG-S in the presence of rounding errors *Lecture Notes in Mathematics* (eds O Axelsson and L Y Kolotilina) **1457** Springer–Verlag

5 Arguments

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 nonzero elements in the lower triangular part of the matrix A .

Constraint: $1 \leq \text{NNZ} \leq N \times (N + 1)/2$.

3: A(LA) – REAL (KIND=nag_wp) array *Input/Output*

On entry: the nonzero 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.

On exit: the first NNZ elements of A contain the nonzero elements of A and the next NNZC elements contain the elements of the lower triangular matrix C . Matrix elements are ordered by increasing row index, and by increasing column index within each row.

4: LA – INTEGER *Input*

On entry: the dimension of the arrays A, IROW and ICOL as declared in the (sub)program from which F11JAF is called. These arrays must be of sufficient size to store both A (NNZ elements) and C (NNZC elements).

Constraint: $LA \geq 2 \times \text{NNZ}$.

5: IROW(LA) – INTEGER array *Input/Output*

6: ICOL(LA) – INTEGER array *Input/Output*

On entry: the row and column indices of the nonzero elements supplied in A .

Constraints:

IROW and ICOL must satisfy these constraints (which may be imposed by a call to F11ZBF):

$$1 \leq \text{IROW}(i) \leq N \text{ and } 1 \leq \text{ICOL}(i) \leq \text{IROW}(i), \text{ for } i = 1, 2, \dots, \text{NNZ};$$

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

On exit: the row and column indices of the nonzero elements returned in A .

- 7: LFILL – INTEGER *Input*
On entry: if LFILL ≥ 0 its value is the maximum level of fill allowed in the decomposition (see Section 9.2). A negative value of LFILL indicates that DTOL will be used to control the fill instead.
- 8: DTOL – REAL (KIND=nag_wp) *Input*
On entry: if LFILL < 0 , DTOL is used as a drop tolerance to control the fill-in (see Section 9.2); otherwise DTOL is not referenced.
Constraint: if LFILL < 0 , DTOL ≥ 0.0 .
- 9: MIC – CHARACTER(1) *Input*
On entry: indicates whether or not the factorization should be modified to preserve row sums (see Section 9.3).
MIC = 'M'
The factorization is modified.
MIC = 'N'
The factorization is not modified.
Constraint: MIC = 'M' or 'N'.
- 10: DSCALE – REAL (KIND=nag_wp) *Input*
On entry: the diagonal scaling parameter. All diagonal elements are multiplied by the factor (1 + DSCALE) at the start of the factorization. This can be used to ensure that the preconditioner is positive definite. See Section 9.3.
- 11: PSTRAT – CHARACTER(1) *Input*
On entry: specifies the pivoting strategy to be adopted.
PSTRAT = 'N'
No pivoting is carried out.
PSTRAT = 'M'
Diagonal pivoting aimed at minimizing fill-in is carried out, using the Markowitz strategy.
PSTRAT = 'U'
Diagonal pivoting is carried out according to the user-defined input value of IPIV.
Suggested value: PSTRAT = 'M'.
Constraint: PSTRAT = 'N', 'M' or 'U'.
- 12: IPIV(N) – INTEGER array *Input/Output*
On entry: if PSTRAT = 'U', then IPIV(i) must specify the row index of the diagonal element used as a pivot at elimination stage i . Otherwise IPIV need not be initialized.
Constraint: if PSTRAT = 'U', IPIV must contain a valid permutation of the integers on [1,N].
On exit: the pivot indices. If IPIV(i) = j then the diagonal element in row j was used as the pivot at elimination stage i .
- 13: ISTR(N + 1) – INTEGER array *Output*
On exit: ISTR(i), for $i = 1, 2, \dots, N$, is the starting address in the arrays A, IROW and ICOL of row i of the matrix C . ISTR(N + 1) is the address of the last nonzero element in C plus one.
- 14: NNZC – INTEGER *Output*
On exit: the number of nonzero elements in the lower triangular matrix C .

15: NPIVM – INTEGER *Output*

On exit: the number of pivots which were modified during the factorization to ensure that M was positive definite. The quality of the preconditioner will generally depend on the returned value of NPIVM. If NPIVM is large the preconditioner may not be satisfactory. In this case it may be advantageous to call F11JAF again with an increased value of either LFILL or DSCALE. See also Section 9.4.

16: IWORK(LIWORK) – INTEGER array *Workspace*

17: LIWORK – INTEGER *Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which F11JAF is called.

Constraints:

the minimum permissible value of LIWORK depends on LFILL as follows:

if $LFILL \geq 0$, $LIWORK \geq 2 \times LA - 3 \times NNZ + 7 \times N + 1$;

if $LFILL < 0$, $LIWORK \geq LA - NNZ + 7 \times N + 1$.

18: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

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 argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

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, $N < 1$,

or $NNZ < 1$,

or $NNZ > N \times (N + 1)/2$,

or $LA < 2 \times NNZ$,

or $DTOL < 0.0$,

or $MIC \neq 'M'$ or $'N'$,

or $PSTRAT \neq 'N'$, $'M'$ or $'U'$,

or $LIWORK < 2 \times LA - 3 \times NNZ + 7 \times N + 1$, and $LFILL \geq 0$,

or $LIWORK < LA - NNZ + 7 \times N + 1$, and $LFILL < 0$.

IFAIL = 2

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

$1 \leq IROW(i) \leq N$ and $1 \leq ICOL(i) \leq IROW(i)$, for $i = 1, 2, \dots, NNZ$;

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

Therefore a nonzero 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.

IFAIL = 3

On entry, PSTRAT = 'U', but IPIV does not represent a valid permutation of the integers in $[1, N]$. An input value of IPIV is either out of range or repeated.

IFAIL = 4

LA is too small, resulting in insufficient storage space for fill-in elements. The decomposition has been terminated before completion. Either increase LA or reduce the amount of fill by setting PSTRAT = 'M', reducing LFILL, or increasing DTOL.

IFAIL = 5 (F11ZBF)

A serious error has occurred in an internal call to the specified routine. Check all subroutine calls and array sizes. Seek expert help.

IFAIL = -99

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -399

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -999

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The accuracy of the factorization will be determined by the size of the elements that are dropped and the size of any modifications made to the diagonal elements. If these sizes are small then the computed factors will correspond to a matrix close to A . The factorization can generally be made more accurate by increasing LFILL, or by reducing DTOL with $LFILL < 0$.

If F11JAF is used in combination with F11GEF or F11JCF, the more accurate the factorization the fewer iterations will be required. However, the cost of the decomposition will also generally increase.

8 Parallelism and Performance

F11JAF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

9.1 Timing

The time taken for a call to F11JAF is roughly proportional to $(NNZC)^2/N$.

9.2 Control of Fill-in

If $\text{LFILL} \geq 0$ the amount of fill-in occurring in the incomplete factorization is controlled by limiting the maximum **level** of fill-in to LFILL . The original nonzero elements of A are defined to be of level 0. The fill level of a new nonzero location occurring during the factorization is defined as

$$k = \max(k_e, k_c) + 1,$$

where k_e is the level of fill of the element being eliminated, and k_c is the level of fill of the element causing the fill-in.

If $\text{LFILL} < 0$ the fill-in is controlled by means of the **drop tolerance** DTOL . A potential fill-in element a_{ij} occurring in row i and column j will not be included if

$$|a_{ij}| < \text{DTOL} \times \sqrt{|a_{ii}a_{jj}|}.$$

For either method of control, any elements which are not included are discarded if $\text{MIC} = \text{'N'}$, or subtracted from the diagonal element in the elimination row if $\text{MIC} = \text{'M'}$.

9.3 Choice of Arguments

There is unfortunately no choice of the various algorithmic arguments which is optimal for all types of symmetric matrix, and some experimentation will generally be required for each new type of matrix encountered.

If the matrix A is not known to have any particular special properties the following strategy is recommended. Start with $\text{LFILL} = 0$, $\text{MIC} = \text{'N'}$ and $\text{DSCALE} = 0.0$. If the value returned for NPIVM is significantly larger than zero, i.e., a large number of pivot modifications were required to ensure that M was positive definite, the preconditioner is not likely to be satisfactory. In this case increase either LFILL or DSCALE until NPIVM falls to a value close to zero. Once suitable values of LFILL and DSCALE have been found try setting $\text{MIC} = \text{'M'}$ to see if any improvement can be obtained by using **modified** incomplete Cholesky.

F11JAF is primarily designed for positive definite matrices, but may work for some mildly indefinite problems. If NPIVM cannot be satisfactorily reduced by increasing LFILL or DSCALE then A is probably too indefinite for this routine.

If A has non-positive off-diagonal elements, is nonsingular, and has only non-negative elements in its inverse, it is called an 'M-matrix'. It can be shown that no pivot modifications are required in the incomplete Cholesky factorization of an M-matrix (see Meijerink and Van der Vorst (1977)). In this case a good preconditioner can generally be expected by setting $\text{LFILL} = 0$, $\text{MIC} = \text{'M'}$ and $\text{DSCALE} = 0.0$.

For certain mesh-based problems involving M-matrices it can be shown in theory that setting $\text{MIC} = \text{'M'}$, and choosing DSCALE appropriately can reduce the order of magnitude of the condition number of the preconditioned matrix as a function of the mesh steplength (see Chan (1991)). In practise this property often holds even with $\text{DSCALE} = 0.0$, although an improvement in condition can result from increasing DSCALE slightly (see Van der Vorst (1990)).

Some illustrations of the application of F11JAF to linear systems arising from the discretization of two-dimensional elliptic partial differential equations, and to random-valued randomly structured symmetric positive definite linear systems, can be found in Salvini and Shaw (1995).

9.4 Direct Solution of positive definite Systems

Although it is not their primary purpose, F11JAF and F11JBF may be used together to obtain a **direct** solution to a symmetric positive definite linear system. To achieve this the call to F11JBF should be preceded by a **complete** Cholesky factorization

$$A = PLDL^T P^T = M.$$

A complete factorization is obtained from a call to F11JAF with $\text{LFILL} < 0$ and $\text{DTOL} = 0.0$, provided $\text{NPIVM} = 0$ on exit. A nonzero value of NPIVM indicates that A is not positive definite, or is ill-conditioned. A factorization with nonzero NPIVM may serve as a preconditioner, but will not result in a

direct solution. It is therefore **essential** to check the output value of NPIVM if a direct solution is required.

The use of F11JAF and F11JBF as a direct method is illustrated in Section 10 in F11JBF.

10 Example

This example reads in a symmetric sparse matrix A and calls F11JAF to compute an incomplete Cholesky factorization. It then outputs the nonzero elements of both A and $C = L + D^{-1} - I$.

The call to F11JAF has LFILL = 0, MIC = 'N', DSCALE = 0.0 and PSTRAT = 'M', giving an unmodified zero-fill factorization of an unperturbed matrix, with Markowitz diagonal pivoting.

10.1 Program Text

```

Program f11jafe

!      F11JAF Example Program Text

!      Mark 26 Release. NAG Copyright 2016.

!      .. Use Statements ..
Use nag_library, Only: f11jaf, nag_wp
!      .. Implicit None Statement ..
Implicit None
!      .. Parameters ..
Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..
Real (Kind=nag_wp)         :: dscale, dtol
Integer                    :: i, ifail, la, lfill, liwork, n, nnz, &
                          nnzc, npivm
Character (1)              :: mic, pstrat
!      .. Local Arrays ..
Real (Kind=nag_wp), Allocatable :: a(:)
Integer, Allocatable        :: icol(:), ipiv(:), irow(:), istr(:), &
                          iwork(:)
!      .. Executable Statements ..
Write (nout,*) 'F11JAF Example Program Results'
!      Skip heading in data file
Read (nin,*)

!      Read algorithmic parameters

Read (nin,*) n
Read (nin,*) nnz
la = 2*nnz
liwork = 2*la + 7*n + 1
Allocate (a(la),icol(la),ipiv(n),irow(la),istr(n+1),iwork(liwork))
Read (nin,*) lfill, dtol
Read (nin,*) mic, dscale
Read (nin,*) pstrat

!      Read the matrix A

Do i = 1, nnz
  Read (nin,*) a(i), irow(i), icol(i)
End Do

!      Calculate incomplete Cholesky factorization

!      ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
ifail = 0
Call f11jaf(n,nnz,a,la,irow,icol,lfill,dtol,mic,dscale,pstrat,ipiv,istr, &
  nnzc,npivm,iwork,liwork,ifail)

!      Output original matrix

```

```

Write (nout,*) ' Original Matrix'
Write (nout,99997) 'N      =', n
Write (nout,99997) 'NNZ   =', nnz
Do i = 1, nnz
  Write (nout,99999) i, a(i), irow(i), icol(i)
End Do
Write (nout,*)

!      Output details of the factorization

Write (nout,*) ' Factorization'
Write (nout,99997) 'N      =', n
Write (nout,99997) 'NNZ   =', nnzc
Write (nout,99997) 'NPIVM =', npivm
Do i = nnz + 1, nnz + nnzc
  Write (nout,99999) i, a(i), irow(i), icol(i)
End Do
Write (nout,*)

Write (nout,*) '          I      IPIV(I)'
Do i = 1, n
  Write (nout,99998) i, ipiv(i)
End Do

99999 Format (1X,I8,E16.4,2I8)
99998 Format (1X,2I8)
99997 Format (1X,A,I16)
End Program f11jaf

```

10.2 Program Data

F11JAF Example Program Data

```

7          N
16         NNZ
0 0.0      LFILL, DTOL
'N' 0.0    MIC, DSCALE
'M'        PSTRAT
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

```

10.3 Program Results

F11JAF Example Program Results

Original Matrix

```

N      =          7
NNZ    =          16
  1     0.4000E+01    1     1
  2     0.1000E+01    2     1
  3     0.5000E+01    2     2
  4     0.2000E+01    3     3
  5     0.2000E+01    4     2
  6     0.3000E+01    4     4
  7    -0.1000E+01    5     1
  8     0.1000E+01    5     4
  9     0.4000E+01    5     5
 10     0.1000E+01    6     2

```


11	-0.2000E+01	6	5
12	0.3000E+01	6	6
13	0.2000E+01	7	1
14	-0.1000E+01	7	2
15	-0.2000E+01	7	3
16	0.5000E+01	7	7
Factorization			
N	=	7	
NNZ	=	16	
NPIVM	=	0	
17	0.5000E+00	1	1
18	0.3333E+00	2	2
19	0.3333E+00	3	2
20	0.2727E+00	3	3
21	-0.5455E+00	4	3
22	0.5238E+00	4	4
23	-0.2727E+00	5	3
24	0.2683E+00	5	5
25	0.6667E+00	6	2
26	0.5238E+00	6	4
27	0.2683E+00	6	5
28	0.3479E+00	6	6
29	-0.1000E+01	7	1
30	0.5366E+00	7	5
31	-0.5345E+00	7	6
32	0.9046E+00	7	7
I IPIV(I)			
1	3		
2	4		
3	5		
4	6		
5	1		
6	2		
7	7		
