F04MAF – 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

To solve a sparse symmetric positive-definite system of linear equations, $Ax = b$, using a pre-conditioned conjugate gradient method, where A has been factorized by F01MAF.

2 Specification

```
 SUBROUTINE F04MAF(N, NZ, A, LICN, IRN, LIRN, ICN, B, ACC, NOITS,
1                  WKEEP, WORK, IKEEP, INFORM, IFAIL)
 INTEGER           N, NZ, LICN, IRN(LIRN), LIRN, ICN(LICN),
1                  NOITS(2), IKEEP(2*N), INFORM(4), IFAIL
 real              A(LICN), B(N), ACC(2), WKEEP(3*N), WORK(3*N)
```

3 Description

F04MAF solves the n linear equations

$$Ax = b \tag{1}$$

where A is a sparse symmetric positive-definite matrix, following the incomplete Cholesky factorization by F01MAF, given by

$$C = PLDL^TP^T, \quad WAW = C + E,$$

where P is a permutation matrix, L is a unit lower triangular matrix, D is a diagonal matrix with positive diagonal elements, E is an error matrix representing elements dropped during the factorization and diagonal elements that have been modified to ensure that C is positive-definite, and W is a diagonal matrix, chosen to make the diagonal elements of WAW unity.

Equation (1) is solved by applying a pre-conditioned conjugate gradient method to the equations

$$(WAW)(W^{-1}x) = Wb \tag{2}$$

using C as the pre-conditioning matrix. Details of the conjugate gradient method are given in Munksgaard [1].

The iterative procedure is terminated if

$$\|Wr\|_2 \leq \eta \tag{3}$$

where r is the residual vector $r = b - Ax$, $\|r\|_2$ denotes the Euclidean length of r, η is a user-supplied tolerance and x is the current approximation to the solution. Notice that

$$Wr = Wb - (WAW)(W^{-1}x)$$

so that Wr is the residual of the normalised equations (2).

F04MAF is based on the Harwell Library routine MA31B

4 References

[1] Munksgaard N (1980) Solving sparse symmetric sets of linear equations by pre-conditioned conjugate gradients *ACM Trans. Math. Software* **6** 206–219

5 Parameters

1: N — INTEGER *Input*

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

Constraint: $N \geq 1$.

2: NZ — INTEGER *Input*

On entry: the number of non-zero elements in the upper triangular part of the matrix A, including the number of elements on the leading diagonal.

Constraint: $NZ \geq N$.

3: A(LICN) — ***real*** array *Input*

On entry: the first LROW elements, where LROW is the value supplied in INFORM(1), must contain details of the factorization, as returned by F01MAF.

4: LICN — INTEGER *Input*

On entry: the length of the array A, as declared in the (sub)program from which F04MAF is called. It need never be larger than the value of LICN supplied to F01MAF.

Constraint: $\mathrm{LICN} \geq \mathrm{NZ} - \mathrm{N} + \mathrm{INFORM}(1)$.

5: IRN(LIRN) — INTEGER array *Input*

On entry: the first LCOL elements, where LCOL is the value supplied in INFORM(2), must contain details of the factorization, as returned by F01MAF.

6: LIRN — INTEGER *Input*

On entry: the length of the array IRN, as declared in the (sub)program from which F04MAF is called. It need never be larger than the value of LIRN supplied to F01MAF.

Constraint: $\mathrm{LIRN} \geq \mathrm{INFORM}(2)$.

7: ICN(LICN) — INTEGER array *Input*

On entry: the first LROW elements, where LROW is the value supplied in INFORM(1), must contain details of the factorization, as returned by F01MAF.

8: B(N) — ***real*** array *Input/Output*

On entry: the right-hand side vector b.

On exit: B is overwritten by the solution vector x.

9: ACC(2) — ***real*** array *Input/Output*

On entry: ACC(1) specifies the tolerance for convergence, η, in equation (3) of Section 3. If ACC(1) is outside the range $[\epsilon, 1]$, where ϵ is the ***machine precision***, then the value ϵ is used in place of ACC(1). ACC(2) need not be set.

On exit: ACC(2) contains the actual value of $\|Wr\|_2$ at the final point. ACC(1) is unchanged.

10: NOITS(2) — INTEGER array *Input/Output*

On entry: NOITS(1) specifies the maximum permitted number of iterations. If $\mathrm{NOITS}(1) < 1$, then the value 100 is used in its place. NOITS(2) need not be set.

On exit: NOITS(2) contains the number of iterations taken to converge. NOITS(1) is unchanged.

11: WKEEP(3*N) — ***real*** array *Input*

On entry: WKEEP must be unchanged from the previous call of F01MAF.

12: WORK(3∗N) — ***real*** array *Output*

On exit: WORK(1) contains a lower bound for the condition number of A. The rest of the array is used for workspace.

13: IKEEP(2∗N) — INTEGER array *Input*

On entry: IKEEP must be unchanged from the previous call of F01MAF.

14: INFORM(4) — INTEGER array *Input*

On entry: INFORM must be unchanged from the previous call of F01MAF.

15: IFAIL — INTEGER *Input/Output*

For this routine, the normal use of IFAIL is extended to control the printing of error and warning messages as well as specifying hard or soft failure (see Chapter P01).

Before entry, IFAIL must be set to a value with the decimal expansion cba, where each of the decimal digits c, b and a must have a value of 0 or 1.

$a = 0$ specifies hard failure, otherwise soft failure;
$b = 0$ suppresses error messages, otherwise error messages will be printed (see Section 6);
$c = 0$ suppresses warning messages, otherwise warning messages will be printed (see Section 6).

The recommended value for inexperienced users is 110 (i.e., hard failure with all messages printed).

Unless the routine detects an error (see Section 6), IFAIL contains 0 on exit.

6 Error Indicators and Warnings

For each error, an explanatory error message is output on the current error message unit (as defined by X04AAF), unless suppressed by the value of IFAIL on entry.

Errors detected by the routine:

IFAIL = 1

On entry, N < 1,
or NZ < N,
or LICN < NZ − N + INFORM(1),
or LIRN < INFORM(2).

IFAIL = 2

Convergence has not taken place within the requested NOITS(1) number of iterations. ACC(2) gives the value $\|Wr\|_2$, for the final point. Either too few iterations have been allowed, or the requested convergence criterion cannot be met.

IFAIL = 3

The matrix A is singular, or nearly singular. Singularity has been detected during the conjugate gradient iterations, so that the computations are not complete.

IFAIL = 4

The matrix A is singular, or nearly singular. The message output on the current error message channel will include an estimate of the condition number of A. In the case of soft failure an approximate solution is returned such that the value $\|Wr\|_2$ is given by ACC(2) and the estimate (a lower bound) of the condition number is returned in WORK(1).

7 Accuracy

On successful return, or on return with IFAIL = 2 or IFAIL = 4 the computed solution will satisfy equation (3) of Section 3, with η = ACC(2).

8 Further Comments

The time taken by the routine will depend upon the sparsity of the factorization and the number of iterations required. The number of iterations will be affected by the nature of the factorization supplied by F01MAF. The more incomplete the factorization, the higher the number of iterations required by F04MAF.

When the solution of several systems of equations, all with the same matrix of coefficients, A, is required, then F01MAF need be called only once to factorize A. This is illustrated in the context of an eigenvalue problem in the example program for F02FJF.

9 Example

The example program illustrates the use of F01MAF in conjunction with F04MAF to solve the 16 linear equations $Ax = b$, where

$$A = \left(\begin{array}{cccccccccccccccc}
1 & -\frac{1}{4} & & & -\frac{1}{4} & & & & & & & & & & & \\
-\frac{1}{4} & 1 & -\frac{1}{4} & & & -\frac{1}{4} & & & & & & & & & & \\
 & -\frac{1}{4} & 1 & -\frac{1}{4} & & & -\frac{1}{4} & & & & & & & & & \\
 & & -\frac{1}{4} & 1 & 0 & & & -\frac{1}{4} & & & & & & & & \\
-\frac{1}{4} & & & 0 & 1 & -\frac{1}{4} & & & -\frac{1}{4} & & & & & & & \\
 & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} & & & -\frac{1}{4} & & & & & & \\
 & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} & & & -\frac{1}{4} & & & & & \\
 & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & 0 & & & -\frac{1}{4} & & & & \\
 & & & & -\frac{1}{4} & & & 0 & 1 & -\frac{1}{4} & & & -\frac{1}{4} & & & \\
 & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} & & & -\frac{1}{4} & & \\
 & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} & & & -\frac{1}{4} & \\
 & & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & 0 & & & -\frac{1}{4} \\
 & & & & & & & & -\frac{1}{4} & & & 0 & 1 & -\frac{1}{4} & & \\
 & & & & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} & \\
 & & & & & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} \\
 & & & & & & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1
\end{array}\right)$$

$$b^T = \left(\frac{1}{2} \quad \frac{1}{4} \quad \frac{1}{4} \quad \frac{1}{2} \quad \frac{1}{4} \quad 0 \quad 0 \quad \frac{1}{4} \quad \frac{1}{4} \quad 0 \quad 0 \quad \frac{1}{4} \quad \frac{1}{2} \quad \frac{1}{4} \quad \frac{1}{4} \quad \frac{1}{2}\right)$$

The n by n matrix A arises in the solution of Laplace's equation in a unit-square, using a five-point formula with a 6 by 6 discretisation, with unity on the boundaries.

The drop tolerance, DROPTL, is taken as 0.1 and the density factor, DENSW, is taken as 0.8. The value IFAIL = 111 is used so that advisory and error messages will be printed, but soft failure would occur if IFAIL were returned as non-zero.

A relative accuracy of about 0.0001 is requested in the solution from F04MAF, with a maximum of 50 iterations.

The example program for F02FJF illustrates the use of routines F01MAF and F04MAF in solving an eigenvalue problem.

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.

```
*     F04MAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          M, N, LICN, LIRN, NN
      PARAMETER        (M=4,N=4,LICN=90,LIRN=50,NN=N*M)
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Local Scalars ..
      real             DENSW, DROPTL
      INTEGER          I, IFAIL, J, JJ, K, MM1, NZ
*     .. Local Arrays ..
      real             A(LICN), ACC(2), B(NN), WKEEP(NN,3), WORK(NN,3)
      INTEGER          ICN(LICN), IKEEP(NN,2), INFORM(4), IRN(LIRN),
     +                 IWORK(NN,6), NOITS(2)
      LOGICAL          ABORT(3)
*     .. External Subroutines ..
      EXTERNAL         F01MAF, F04MAF, X04ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F04MAF Example Program Results'
      CALL X04ABF(1,NOUT)
*
*     Initialise the right hand side vector B.
*
      MM1 = M - 1
      DO 40 I = 1, N
         K = (I-1)*M
         B(K+1) = 0.25e0
         DO 20 J = 2, MM1
            B(K+J) = 0.0e0
   20    CONTINUE
         B(K+M) = 0.25e0
   40 CONTINUE
      K = N*MM1
      DO 60 I = 1, M
         B(I) = B(I) + 0.25e0
         B(K+I) = B(K+I) + 0.25e0
   60 CONTINUE
*
*     Set up the coefficient matrix A.
*
      K = 0
      DO 80 I = 1, NN
         K = K + 1
         A(K) = 1.0e0
         IRN(K) = I
         ICN(K) = I
   80 CONTINUE
      DO 120 I = 1, N
         DO 100 J = 1, MM1
            K = K + 1
            JJ = (I-1)*N + J
            A(K) = -0.25e0
            IRN(K) = JJ
            ICN(K) = JJ + 1
```

```
  100    CONTINUE
  120 CONTINUE
      DO 140 I = N + 1, NN
         K = K + 1
         A(K) = -0.25e0
         IRN(K) = I - N
         ICN(K) = I
  140 CONTINUE
      NZ = K
*
*     Call F01MAF to perform the incomplete Cholesky factorization of
*     the sparse, symmetric, positive definite matrix A.
*
      DO 160 I = 1, 3
         ABORT(I) = .TRUE.
  160 CONTINUE
      DROPTL = 0.1e0
      DENSW = 0.8e0
      IFAIL = 111
*
      CALL F01MAF(NN,NZ,A,LICN,IRN,LIRN,ICN,DROPTL,DENSW,WKEEP,IKEEP,
     +            IWORK,ABORT,INFORM,IFAIL)
*
      WRITE (NOUT,*)
      IF (IFAIL.NE.0) THEN
         WRITE (NOUT,99999)
     +     ' Error IFAIL returned from F01MAF with value', IFAIL
      ELSE
         WRITE (NOUT,99999) ' No. of elements of A ( and ICN ) used is '
     +     , INFORM(1)
         WRITE (NOUT,99999) ' No. of elements of IRN used is           '
     +     , INFORM(2)
         WRITE (NOUT,*)
*
*        Call F04MAF to solve the sparse, symmetric, positive definite
*        system of linear equations  A*X = B.
*
         NOITS(1) = 50
         ACC(1) = 0.0001e0
         IFAIL = 111
*
         CALL F04MAF(NN,NZ,A,LICN,IRN,LIRN,ICN,B,ACC,NOITS,WKEEP,WORK,
     +               IKEEP,INFORM,IFAIL)
*
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,99999)
     +        ' Error IFAIL returned from F04MAF with value', IFAIL
         ELSE
            WRITE (NOUT,99999) ' Number of iterations = ', NOITS(2)
            WRITE (NOUT,99998) ' Norm of residual     = ', ACC(2)
            WRITE (NOUT,*)
            WRITE (NOUT,*) ' Solution vector'
            WRITE (NOUT,99997) (B(I),I=1,NN)
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I4)
```

```
99998 FORMAT (1X,A,e12.3)
99997 FORMAT (1X,4e15.5)
      END
```

9.2 Program Data

None.

9.3 Program Results

```
F04MAF Example Program Results

 No. of elements of A ( and ICN ) used is   28
 No. of elements of IRN used is             24

 Number of iterations =    3
 Norm of residual     =    0.836E-05

 Solution vector
    0.10000E+01    0.10000E+01    0.10000E+01    0.10000E+01
    0.10000E+01    0.10000E+01    0.10000E+01    0.10000E+01
    0.10000E+01    0.10000E+01    0.10000E+01    0.10000E+01
    0.10000E+01    0.10000E+01    0.10000E+01    0.10000E+01
```
