NAG Library Routine Document

F11JQF

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

F11JQF solves a complex sparse Hermitian system of linear equations, represented in symmetric coordinate storage format, using a conjugate gradient or Lanczos method, with incomplete Cholesky preconditioning.

2 Specification

```
SUBROUTINE F11JQF (METHOD, N, NNZ, A, LA, IROW, ICOL, IPIV, ISTR, B, TOL,       &
                   MAXITN, X, RNORM, ITN, WORK, LWORK, IFAIL)
INTEGER                 N, NNZ, LA, IROW(LA), ICOL(LA), IPIV(N), ISTR(N+1),       &
                        MAXITN, ITN, LWORK, IFAIL
REAL (KIND=nag_wp)      TOL, RNORM
COMPLEX (KIND=nag_wp) A(LA), B(N), X(N), WORK(LWORK)
CHARACTER(*)            METHOD
```

3 Description

F11JQF solves a complex sparse Hermitian linear system of equations

$$Ax = b,$$

using a preconditioned conjugate gradient method (see Meijerink and Van der Vorst (1977)), or a preconditioned Lanczos method based on the algorithm SYMMLQ (see Paige and Saunders (1975)). The conjugate gradient method is more efficient if A is positive definite, but may fail to converge for indefinite matrices. In this case the Lanczos method should be used instead. For further details see Barrett *et al.* (1994).

F11JQF uses the incomplete Cholesky factorization determined by F11JNF as the preconditioning matrix. A call to F11JQF must always be preceded by a call to F11JNF. Alternative preconditioners for the same storage scheme are available by calling F11JSF.

The matrix A and the preconditioning matrix M are represented in symmetric coordinate storage (SCS) format (see Section 2.1.2 in the F11 Chapter Introduction) in the arrays A, IROW and ICOL, as returned from F11JNF. The array A holds the nonzero entries in the lower triangular parts of these matrices, while IROW and ICOL hold the corresponding row and column indices.

4 References

Barrett R, Berry M, Chan T F, Demmel J, Donato J, Dongarra J, Eijkhout V, Pozo R, Romine C and Van der Vorst H (1994) *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods* SIAM, Philadelphia

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

Paige C C and Saunders M A (1975) Solution of sparse indefinite systems of linear equations *SIAM J. Numer. Anal.* **12** 617–629

5 Parameters

1: METHOD – CHARACTER(*) *Input*

On entry: specifies the iterative method to be used.

METHOD = 'CG'
Conjugate gradient method.

METHOD = 'SYMMLQ'
Lanczos method (SYMMLQ).

Constraint: METHOD = 'CG' or 'SYMMLQ'.

2: N – INTEGER *Input*

On entry: n, the order of the matrix A. This **must** be the same value as was supplied in the preceding call to F11JNF.

Constraint: $\mathrm{N} \geq 1$.

3: NNZ – INTEGER *Input*

On entry: the number of nonzero elements in the lower triangular part of the matrix A. This **must** be the same value as was supplied in the preceding call to F11JNF.

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

4: A(LA) – COMPLEX (KIND=nag_wp) array *Input*

On entry: the values returned in the array A by a previous call to F11JNF.

5: LA – INTEGER *Input*

On entry: the dimension of the arrays A, IROW and ICOL as declared in the (sub)program from which F11JQF is called. This **must** be the same value as was supplied in the preceding call to F11JNF.

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

6: IROW(LA) – INTEGER array *Input*
7: ICOL(LA) – INTEGER array *Input*
8: IPIV(N) – INTEGER array *Input*
9: ISTR(N + 1) – INTEGER array *Input*

On entry: the values returned in arrays IROW, ICOL, IPIV and ISTR by a previous call to F11JNF.

10: B(N) – COMPLEX (KIND=nag_wp) array *Input*

On entry: the right-hand side vector b.

11: TOL – REAL (KIND=nag_wp) *Input*

On entry: the required tolerance. Let x_k denote the approximate solution at iteration k, and r_k the corresponding residual. The algorithm is considered to have converged at iteration k if

$$\|r_k\|_\infty \leq \tau \times \left(\|b\|_\infty + \|A\|_\infty \|x_k\|_\infty\right).$$

If $\mathrm{TOL} \leq 0.0$, $\tau = \max(\sqrt{\epsilon}, \sqrt{n}\epsilon)$ is used, where ϵ is the ***machine precision***. Otherwise $\tau = \max(\mathrm{TOL}, 10\epsilon, \sqrt{n}\epsilon)$ is used.

Constraint: $\mathrm{TOL} < 1.0$.

12: MAXITN – INTEGER *Input*

On entry: the maximum number of iterations allowed.

Constraint: $\mathrm{MAXITN} \geq 1$.

13: X(N) – COMPLEX (KIND=nag_wp) array *Input/Output*

On entry: an initial approximation to the solution vector x.

On exit: an improved approximation to the solution vector x.

14: RNORM – REAL (KIND=nag_wp) *Output*

On exit: the final value of the residual norm $\|r_k\|_\infty$, where k is the output value of ITN.

15: ITN – INTEGER *Output*

On exit: the number of iterations carried out.

16: WORK(LWORK) – COMPLEX (KIND=nag_wp) array *Workspace*
17: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F11JQF is called.

Constraints:

if METHOD = 'CG', LWORK $\geq 6 \times$ N $+ 120$;
if METHOD = 'SYMMLQ', LWORK $\geq 7 \times$ N $+ 120$.

18: 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.

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.**

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, METHOD $\neq$ 'CG' or 'SYMMLQ',
or N < 1,
or NNZ < 1,
or NNZ $>$ N $\times$ (N $+ 1)/2$,
or LA too small,
or TOL ≥ 1.0,
or MAXITN < 1,
or LWORK too small.

IFAIL $= 2$

On entry, the SCS representation of A is invalid. Further details are given in the error message. Check that the call to F11JQF has been preceded by a valid call to F11JNF, and that the arrays A, IROW, and ICOL have not been corrupted between the two calls.

IFAIL $= 3$

On entry, the SCS representation of M is invalid. Further details are given in the error message. Check that the call to F11JQF has been preceded by a valid call to F11JNF, and that the arrays A, IROW, ICOL, IPIV and ISTR have not been corrupted between the two calls.

IFAIL $= 4$

The required accuracy could not be obtained. However, a reasonable accuracy has been obtained and further iterations could not improve the result.

IFAIL $= 5$

Required accuracy not obtained in MAXITN iterations.

IFAIL $= 6$

The preconditioner appears not to be positive definite.

IFAIL $= 7$

The matrix of the coefficients appears not to be positive definite (conjugate gradient method only).

IFAIL $= 8$

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

7 Accuracy

On successful termination, the final residual $r_k = b - Ax_k$, where $k = \text{ITN}$, satisfies the termination criterion

$$\|r_k\|_\infty \leq \tau \times \left(\|b\|_\infty + \|A\|_\infty \|x_k\|_\infty\right).$$

The value of the final residual norm is returned in RNORM.

8 Further Comments

The time taken by F11JQF for each iteration is roughly proportional to the value of NNZC returned from the preceding call to F11JNF. One iteration with the Lanczos method (SYMMLQ) requires a slightly larger number of operations than one iteration with the conjugate gradient method.

The number of iterations required to achieve a prescribed accuracy cannot easily be determined *a priori*, as it can depend dramatically on the conditioning and spectrum of the preconditioned matrix of the coefficients $\bar{A} = M^{-1}A$.

9 Example

This example solves a complex sparse Hermitian positive definite system of equations using the conjugate gradient method, with incomplete Cholesky preconditioning.

9.1 Program Text

```
    Program f11jqfe

!     F11JQF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: f11jnf, f11jqf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
```

```
!     .. Parameters ..
      Integer, Parameter                :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)                :: dscale, dtol, rnorm, tol
      Integer                           :: i, ifail, itn, la, lfill, liwork,    &
                                           lwork, maxitn, n, nnz, nnzc, npivm
      Character (6)                     :: method
      Character (1)                     :: mic, pstrat
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:), b(:), work(:), x(:)
      Integer, Allocatable              :: icol(:), ipiv(:), irow(:), istr(:),  &
                                           iwork(:)
!     .. Executable Statements ..
      Write (nout,*) 'F11JQF Example Program Results'
!     Skip heading in data file
      Read (nin,*)

!     Read algorithmic parameters

      Read (nin,*) n
      Read (nin,*) nnz
      la = 3*nnz
      liwork = 2*la + 7*n + 1
      lwork = 7*n + 120
      Allocate (a(la),b(n),work(lwork),x(n),icol(la),ipiv(n),irow(la), &
        istr(n+1),iwork(liwork))
      Read (nin,*) method
      Read (nin,*) lfill, dtol
      Read (nin,*) mic, dscale
      Read (nin,*) pstrat
      Read (nin,*) tol, maxitn

!     Read the matrix A

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

!     Read rhs vector b and initial approximate solution x

      Read (nin,*) b(1:n)
      Read (nin,*) x(1:n)

!     Calculate incomplete Cholesky factorization

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call f11jnf(n,nnz,a,la,irow,icol,lfill,dtol,mic,dscale,pstrat,ipiv,istr, &
        nnzc,npivm,iwork,liwork,ifail)

!     Solve Ax = b using F11JQF

      Call f11jqf(method,n,nnz,a,la,irow,icol,ipiv,istr,b,tol,maxitn,x,rnorm, &
        itn,work,lwork,ifail)

      Write (nout,99999) 'Converged in', itn, ' iterations'
      Write (nout,99998) 'Final residual norm =', rnorm

!     Output x

      Write (nout,99997) x(1:n)

99999 Format (1X,A,I10,A)
99998 Format (1X,A,1P,E16.3)
99997 Format (1X,'(',E16.4,',',E16.4,')')
    End Program f11jqfe
```

9.2 Program Data

```
F11JQF Example Program Data
  9                       N
  23                      NNZ
 'CG'                     METHOD
  0 0.0                   LFILL, DTOL
  'N' 0.0                 MIC, DSCALE
  'M'                     PSTRAT
  1.0D-6 100              TOL, MAXITN
 ( 6., 0.)   1    1
 (-1., 1.)   2    1
 ( 6., 0.)   2    2
 ( 0., 1.)   3    2
 ( 5., 0.)   3    3
 ( 5., 0.)   4    4
 ( 2.,-2.)   5    1
 ( 4., 0.)   5    5
 ( 1., 1.)   6    3
 ( 2., 0.)   6    4
 ( 6., 0.)   6    6
 (-4., 3.)   7    2
 ( 0., 1.)   7    5
 (-1., 0.)   7    6
 ( 6., 0.)   7    7
 (-1.,-1.)   8    4
 ( 0.,-1.)   8    6
 ( 9., 0.)   8    8
 ( 1., 3.)   9    1
 ( 1., 2.)   9    5
 (-1., 0.)   9    6
 ( 1., 4.)   9    8
 ( 9., 0.)   9    9     A(I), IROW(I), ICOL(I), I=1,...,NNZ
 (  8.,  54.)
 (-10., -92.)
 ( 25.,  27.)
 ( 26., -28.)
 ( 54.,  12.)
 ( 26., -22.)
 ( 47.,  65.)
 ( 71., -57.)
 ( 60.,  70.)             B(I), I=1,...,N
 (  0.,   0.)
 (  0.,   0.)
 (  0.,   0.)
 (  0.,   0.)
 (  0.,   0.)
 (  0.,   0.)
 (  0.,   0.)
 (  0.,   0.)
 (  0.,   0.)             X(I), I=1,...,N
```

9.3 Program Results

```
 F11JQF Example Program Results
 Converged in         5 iterations
 Final residual norm =       3.197E-14
 (      0.1000E+01,      0.9000E+01)
 (      0.2000E+01,     -0.8000E+01)
 (      0.3000E+01,      0.7000E+01)
 (      0.4000E+01,     -0.6000E+01)
 (      0.5000E+01,      0.5000E+01)
 (      0.6000E+01,     -0.4000E+01)
 (      0.7000E+01,      0.3000E+01)
 (      0.8000E+01,     -0.2000E+01)
 (      0.9000E+01,      0.1000E+01)
```