

# NAG Fortran Library Routine Document

## F11DQF

**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

F11DQF solves a complex sparse non-Hermitian system of linear equations, represented in coordinate storage format, using a restarted generalized minimal residual (RGMRES), conjugate gradient squared (CGS), stabilized bi-conjugate gradient (Bi-CGSTAB), or transpose-free quasi-minimal residual (TFQMR) method, with incomplete *LU* preconditioning.

### 2 Specification

```

SUBROUTINE F11DQF(METHOD, N, NNZ, A, LA, IROW, ICOL, IPIVP, IPIVQ, ISTR,
1             IDIAG, B, M, TOL, MAXITN, X, RNORM, ITN, WORK, LWORK,
2             IFAIL)
INTEGER      N, NNZ, LA, IROW(LA), ICOL(LA), IPIVP(N), IPIVQ(N),
1             ISTR(N+1), IDIAG(N), M, MAXITN, ITN, LWORK, IFAIL
real       TOL, RNORM
complex   A(LA), B(N), X(N), WORK(LWORK)
CHARACTER*(*) METHOD

```

### 3 Description

This routine solves a complex sparse non-Hermitian linear system of equations

$$Ax = b,$$

using a preconditioned RGMRES (Saad and Schultz (1986)), CGS (Sonneveld (1989)), Bi-CGSTAB( $\ell$ ) (Van der Vorst (1989), Sleijpen and Fokkema (1993)), or TFQMR (Freund and Nachtigal (1991), Freund (1993)) method.

F11DQF uses the incomplete *LU* factorization determined by F11DNF as the preconditioning matrix. A call to F11DQF must always be preceded by a call to F11DNF. Alternative preconditioners for the same storage scheme are available by calling F11DSF.

The matrix *A*, and the preconditioning matrix *M*, are represented in coordinate storage (CS) format (see Section 2.1.1 of the F11 Chapter Introduction) in the arrays *A*, *IROW* and *ICOL*, as returned from F11DNF. The array *A* holds the non-zero entries in these matrices, while *IROW* and *ICOL* hold the corresponding row and column indices.

F11DQF is a black-box routine which calls F11BRF, F11BSF and F11BTF. If you wish to use an alternative storage scheme, preconditioner, or termination criterion, or require additional diagnostic information, you should call these underlying routines directly.

### 4 References

Freund R W (1993) A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems *SIAM J. Sci. Comput.* **14** 470–482

Freund R W and Nachtigal N (1991) QMR: a Quasi-Minimal Residual Method for Non-Hermitian Linear Systems *Numer. Math.* **60** 315–339

Saad Y and Schultz M (1986) GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems *SIAM J. Sci. Statist. Comput.* **7** 856–869

Sleijpen G L G and Fokkema D R (1993) BiCGSTAB( $\ell$ ) for linear equations involving matrices with complex spectrum *ETNA* **1** 11–32

Sonneveld P (1989) CGS, a fast Lanczos-type solver for nonsymmetric linear systems *SIAM J. Sci. Statist. Comput.* **10** 36–52

Van der Vorst H (1989) Bi-CGSTAB, a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems *SIAM J. Sci. Statist. Comput.* **13** 631–644

## 5 Parameters

- 1: METHOD – CHARACTER\*(\*) *Input*  
*On entry:* specifies the iterative method to be used. The possible choices are:  
 if METHOD = 'RGMRES', restarted generalized minimum residual method;  
 if METHOD = 'CGS', conjugate gradient squared method;  
 if METHOD = 'BICGSTAB', bi-conjugate gradient stabilized ( $\ell$ ) method;  
 if METHOD = 'TFQMR', transpose-free quasi-minimal residual method.  
*Constraint:* METHOD = 'RGMRES', 'CGS', 'BICGSTAB' or 'TFQMR'.
- 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 F11DNF.  
*Constraint:*  $N \geq 1$ .
- 3: NNZ – INTEGER *Input*  
*On entry:* the number of non-zero elements in the matrix  $A$ . This **must** be the same value as was supplied in the preceding call to F11DNF.  
*Constraint:*  $1 \leq \text{NNZ} \leq N^2$ .
- 4: A(LA) – **complex** array *Input*  
*On entry:* the values returned in array A by a previous call to F11DNF.
- 5: LA – INTEGER *Input*  
*On entry:* the dimension of the arrays A, IROW and ICOL as declared in the (sub)program from which F11DQF is called. This **must** be the same value as was supplied in the preceding call to F11DNF.  
*Constraint:*  $LA \geq 2 \times \text{NNZ}$ .
- 6: IROW(LA) – INTEGER array *Input*  
 7: ICOL(LA) – INTEGER array *Input*  
 8: IPIVP(N) – INTEGER array *Input/Output*  
 9: IPIVQ(N) – INTEGER array *Input/Output*  
 10: ISTR(N+1) – INTEGER array *Input*  
 11: IDIAG(N) – INTEGER array *Input*  
*On entry:* the values returned in arrays IROW, ICOL, IPIVP, IPIVQ, ISTR and IDIAG by a previous call to F11DNF.  
*On exit:* IPIVP and IPIVQ are used as internal workspace prior to being restored and hence are unchanged.
- 12: B(N) – **complex** array *Input*  
*On entry:* the right-hand side vector  $b$ .

- 13: M – INTEGER Input
- On entry:* if METHOD = 'RGMRES', M is the dimension of the restart subspace; if METHOD = 'BICGSTAB', M is the order  $\ell$  of the polynomial Bi-CGSTAB method; otherwise, M is not referenced.
- Constraints:*
- if METHOD = 'RGMRES',  $0 < M \leq \min(N, 50)$ ,  
if METHOD = 'BICGSTAB',  $0 < M \leq \min(N, 10)$ .
- 14: TOL – *real* 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
- $$\|r_k\|_\infty \leq \tau \times (\|b\|_\infty + \|A\|_\infty \|x_k\|_\infty).$$
- If  $TOL \leq 0.0$ ,  $\tau = \max(\sqrt{\epsilon}, \sqrt{n}\epsilon)$  is used, where  $\epsilon$  is the *machine precision*. Otherwise  $\tau = \max(TOL, 10\epsilon, \sqrt{n}\epsilon)$  is used.
- Constraint:* TOL < 1.0.
- 15: MAXITN – INTEGER Input
- On entry:* the maximum number of iterations allowed.
- Constraint:* MAXITN  $\geq 1$ .
- 16: X(N) – *complex* array Input/Output
- On entry:* an initial approximation to the solution vector  $x$ .
- On exit:* an improved approximation to the solution vector  $x$ .
- 17: RNORM – *real* Output
- On exit:* the final value of the residual norm  $\|r_k\|_\infty$ , where  $k$  is the output value of ITN.
- 18: ITN – INTEGER Output
- On exit:* the number of iterations carried out.
- 19: WORK(LWORK) – *complex* array Workspace
- 20: LWORK – INTEGER Input
- On entry:* the dimension of the array WORK as declared in the (sub)program from which F11DQF is called.
- Constraints:*
- if METHOD = 'RGMRES',  $LWORK \geq 4 \times N + M \times (M + N + 5) + 121$ ,  
if METHOD = 'CGS',  $LWORK \geq 8 \times N + 120$ ,  
if METHOD = 'BICGSTAB',  $LWORK \geq 2 \times N \times (M + 3) + M \times (M + 2) + 120$ ,  
if METHOD = 'TFQMR',  $LWORK \geq 11 \times N + 120$ .
- 21: IFAIL – INTEGER Input/Output
- On entry:* IFAIL must be set to 0, -1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 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, for users 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,  $METHOD \neq 'RGMRES', 'CGS', 'BICGSTAB',$  or  $'TFQMR',$   
 or  $N < 1,$   
 or  $NNZ < 1,$   
 or  $NNZ > N^2,$   
 or  $LA < 2 \times NNZ,$   
 or  $M < 1$  and  $METHOD = 'RGMRES'$  or  $METHOD = 'BICGSTAB',$   
 or  $M > \min(N, 50),$  with  $METHOD = 'RGMRES',$   
 or  $M > \min(N, 10),$  with  $METHOD = 'BICGSTAB',$   
 or  $TOL \geq 1.0,$   
 or  $MAXITN < 1,$   
 or  $LWORK$  too small.

$IFAIL = 2$

On entry, the CS representation of  $A$  is invalid. Further details are given in the error message. Check that the call to F11DQF has been preceded by a valid call to F11DNF, and that the arrays  $A,$   $IROW,$  and  $ICOL$  have not been corrupted between the two calls.

$IFAIL = 3$

On entry, the CS representation of the preconditioning matrix  $M$  is invalid. Further details are given in the error message. Check that the call to F11DQF has been preceded by a valid call to F11DNF, and that the arrays  $A,$   $IROW,$   $ICOL,$   $IPIVP,$   $IPIVQ,$   $ISTR$  and  $IDIAG$  have not been corrupted between the two calls.

$IFAIL = 4$

The required accuracy could not be obtained. However, a reasonable accuracy may have been obtained, and further iterations could not improve the result. You should check the output value of  $RNORM$  for acceptability. This error code usually implies that your problem has been fully and satisfactorily solved to within or close to the accuracy available on your system. Further iterations are unlikely to improve on this situation.

$IFAIL = 5$

Required accuracy not obtained in  $MAXITN$  iterations.

$IFAIL = 6$

Algorithmic breakdown. A solution is returned, although it is possible that it is completely inaccurate.

$IFAIL = 7$

A serious error has occurred in an internal call to F11BRF, F11BSF or F11BTF. 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

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

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

## 8 Further Comments

The time taken by F11DQF for each iteration is roughly proportional to the value of NNZC returned from the preceding call to F11DNF.

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

## 9 Example

This example program solves a complex sparse non-Hermitian linear system of equations using the CGS method, with incomplete *LU* preconditioning.

### 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.

```
*      F11DQF Example Program Text.
*      Mark 19 Release. NAG Copyright 1999.
*      .. Parameters ..
INTEGER          NIN, NOUT
PARAMETER       (NIN=5,NOUT=6)
INTEGER          NMAX, LA, LIWORK, LWORK
PARAMETER       (NMAX=1000,LA=10000,LIWORK=7*NMAX+2,LWORK=10000)
*      .. Local Scalars ..
real           DTOL, RNORM, TOL
INTEGER          I, IFAIL, ITN, LFILL, LWREQ, M, MAXITN, N, NNZ,
+              NNZC, NPIVM
CHARACTER        MILU, PSTRAT
CHARACTER*8      METHOD
*      .. Local Arrays ..
complex        A(LA), B(NMAX), WORK(LWORK), X(NMAX)
INTEGER          ICOL(LA), IDIAG(NMAX), IPIVP(NMAX), IPIVQ(NMAX),
+              IROW(LA), ISTR(NMAX+1), IWORK(LIWORK)
*      .. External Subroutines ..
EXTERNAL         F11DNF, F11DQF
*      .. Intrinsic Functions ..
INTRINSIC        MAX
*      .. Executable Statements ..
WRITE (NOUT,*) 'F11DQF Example Program Results'
WRITE (NOUT,*)
*      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,*) LFILL, DTOL
  READ (NIN,*) PSTRAT
  READ (NIN,*) MILU
  READ (NIN,*) M, TOL, MAXITN
*

```

```

*      Check size of workspace
*
      LWREQ = MAX(4*N+M*(M+N+5)+121,8*N+120,2*N*(M+3)+M*(M+2)+120,
+          11*N+120)
*
      IF (LWORK.LT.LWREQ) THEN
          WRITE (NOUT,'(A,I4)') ' LWORK must be at least', LWREQ
          STOP
      END IF
*
*      Read the matrix A
*
      DO 20 I = 1, NNZ
          READ (NIN,*) A(I), IROW(I), ICOL(I)
20    CONTINUE
*
*      Read rhs vector b and initial approximate solution x
*
      READ (NIN,*) (B(I),I=1,N)
      READ (NIN,*) (X(I),I=1,N)
*
*      Calculate incomplete LU factorization
*
      IFAIL = 0
      CALL F11DNF(N,NNZ,A,LA,IROW,ICOL,LFILL,DTOL,PSTRAT,MILU,IPIVP,
+          IPIVQ,ISTR,IDIAG,NNZC,NPIVM,IWORK,LIWORK,IFAIL)
*
*      Solve Ax = b using F11DQF
*
      CALL F11DQF(METHOD,N,NNZ,A,LA,IROW,ICOL,IPIVP,IPIVQ,ISTR,IDIAG,
+          B,M,TOL,MAXITN,X,RNORM,ITN,WORK,LWORK,IFAIL)
*
      WRITE (NOUT,'(1X,A,I10,A)') 'Converged in', ITN, ' iterations'
      WRITE (NOUT,'(1X,A,1P,D16.3)') 'Final residual norm =', RNORM
      WRITE (NOUT,*)
*
*      Output x
*
      WRITE (NOUT,*) '
          X'
      DO 40 I = 1, N
          WRITE (NOUT,'(1X,,''('',1P,D16.4,','',1P,D16.4,')')') X(I)
40    CONTINUE
      END IF
*
      STOP
      END

```

## 9.2 Program Data

F11DQF Example Program Data

8			N
24			NNZ
'CGS'			METHOD
0 0.0			LFILL, DTOL
'C'			PSTRAT
'N'			MILU
4 1.0e-10 100			M, TOL, MAXITN
( 2., 1.)	1	1	
(-1., 1.)	1	4	
( 1.,-3.)	1	8	
( 4., 7.)	2	1	
(-3., 0.)	2	2	
( 2., 4.)	2	5	
(-7.,-5.)	3	3	
( 2., 1.)	3	6	
( 3., 2.)	4	1	
(-4., 2.)	4	3	
( 0., 1.)	4	4	
( 5.,-3.)	4	7	
(-1., 2.)	5	2	

```

( 8., 6.) 5 5
(-3.,-4.) 5 7
(-6.,-2.) 6 1
( 5.,-2.) 6 3
( 2., 0.) 6 6
( 0.,-5.) 7 3
(-1., 5.) 7 5
( 6., 2.) 7 7
(-1., 4.) 8 2
( 2., 0.) 8 6
( 3., 3.) 8 8      A(I), IROW(I), ICOL(I), I=1,...,NNZ
( 7., 11.)
( 1., 24.)
(-13.,-18.)
(-10., 3.)
( 23., 14.)
( 17., -7.)
( 15., -3.)
( -3., 20.)      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

F11DQF Example Program Results

```

Converged in      4 iterations
Final residual norm =      7.779E-12

```

```

      X
( 1.0000E-00, 1.0000E-00)
( 2.0000E+00, -1.0000E+00)
( 3.0000E+00, 1.0000E-00)
( 4.0000E+00, -1.0000E-00)
( 3.0000E+00, -1.0000E+00)
( 2.0000E+00, 1.0000E-00)
( 1.0000E+00, -1.0000E+00)
( 5.4090E-13, 3.0000E+00)

```

---