# NAG Library Routine Document

# F11DSF

**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

F11DSF 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, without preconditioning, with Jacobi, or with SSOR preconditioning.

## 2 Specification

```
SUBROUTINE F11DSF (METHOD, PRECON, N, NNZ, A, IROW, ICOL, OMEGA, B, M,     &
                   TOL, MAXITN, X, RNORM, ITN, WORK, LWORK, IWORK,         &
                   IFAIL)

INTEGER                 N, NNZ, IROW(NNZ), ICOL(NNZ), M, MAXITN, ITN,       &
                        LWORK, IWORK(2*N+1), IFAIL
REAL (KIND=nag_wp)      OMEGA, TOL, RNORM
COMPLEX (KIND=nag_wp)   A(NNZ), B(N), X(N), WORK(LWORK)
CHARACTER(*)            METHOD
CHARACTER(1)            PRECON
```

## 3 Description

F11DSF solves a complex sparse non-Hermitian system of linear equations:

$$Ax = b,$$

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

F11DSF allows the following choices for the preconditioner:

- no preconditioning;
- Jacobi preconditioning (see Young (1971));
- symmetric successive-over-relaxation (SSOR) preconditioning (see Young (1971)).

For incomplete $LU$ (ILU) preconditioning see F11DQF.

The matrix $A$ is represented in coordinate storage (CS) format (see Section 2.1.1 in the F11 Chapter Introduction) in the arrays A, IROW and ICOL. The array A holds the nonzero entries in the matrix, while IROW and ICOL hold the corresponding row and column indices.

F11DSF 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

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

## 5 Arguments

1: METHOD – CHARACTER(*) *Input*

*On entry*: specifies the iterative method to be used.

METHOD = 'RGMRES'
Restarted generalized minimum residual method.

METHOD = 'CGS'
Conjugate gradient squared method.

METHOD = 'BICGSTAB'
Bi-conjugate gradient stabilized ($\ell$) method.

METHOD = 'TFQMR'
Transpose-free quasi-minimal residual method.

*Constraint*: METHOD = 'RGMRES', 'CGS', 'BICGSTAB' or 'TFQMR'.

2: PRECON – CHARACTER(1) *Input*

*On entry*: specifies the type of preconditioning to be used.

PRECON = 'N'
No preconditioning.

PRECON = 'J'
Jacobi.

PRECON = 'S'
Symmetric successive-over-relaxation (SSOR).

*Constraint*: PRECON = 'N', 'J' or 'S'.

3: N – INTEGER *Input*

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

*Constraint*: N $\geq 1$.

4: NNZ – INTEGER *Input*

*On entry*: the number of nonzero elements in the matrix $A$.

*Constraint*: $1 \leq \mathrm{NNZ} \leq \mathrm{N}^2$.

5: A(NNZ) – COMPLEX (KIND=nag_wp) array *Input*

*On entry*: the nonzero elements 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 F11ZNF may be used to order the elements in this way.

6: IROW(NNZ) – INTEGER array *Input*
7: ICOL(NNZ) – INTEGER array *Input*

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

*Constraints*:

IROW and ICOL must satisfy the following constraints (which may be imposed by a call to F11ZNF):

$1 \leq \text{IROW}(i) \leq \text{N}$ and $1 \leq \text{ICOL}(i) \leq \text{N}$, for $i = 1, 2, \ldots, \text{NNZ}$;
either $\text{IROW}(i-1) < \text{IROW}(i)$ or both $\text{IROW}(i-1) = \text{IROW}(i)$ and $\text{ICOL}(i-1) < \text{ICOL}(i)$, for $i = 2, 3, \ldots, \text{NNZ}$.

8: OMEGA – REAL (KIND=nag_wp) *Input*

*On entry*: if PRECON = 'S', OMEGA is the relaxation parameter $\omega$ to be used in the SSOR method. Otherwise OMEGA need not be initialized and is not referenced.

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

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

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

10: 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', $0 < \text{M} \leq \min(\text{N}, 50)$;
if METHOD = 'BICGSTAB', $0 < \text{M} \leq \min(\text{N}, 10)$.

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

*Constraint*: $\text{TOL} < 1.0$.

12: MAXITN – INTEGER *Input*

*On entry*: the maximum number of iterations allowed.

*Constraint*: $\text{MAXITN} \geq 1$.

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 F11DSF is called.

*Constraints*:

if METHOD = 'RGMRES', LWORK $\geq 4 \times \mathrm{N} + \mathrm{M} \times (\mathrm{M} + \mathrm{N} + 5) + nu + 121$;
if METHOD = 'CGS', LWORK $\geq 8 \times \mathrm{N} + nu + 120$;
if METHOD = 'BICGSTAB', LWORK $\geq 2 \times \mathrm{N} \times (\mathrm{M} + 3) + \mathrm{M} \times (\mathrm{M} + 2) + nu + 120$;
if METHOD = 'TFQMR', LWORK $\geq 11 \times \mathrm{N} + nu + 120$.

Where $nu = \mathrm{N}$ for PRECON = 'J' or 'S' and $nu = 0$ otherwise.

18: IWORK($2 \times \mathrm{N} + 1$) – INTEGER array *Workspace*

19: 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, METHOD $\neq$ 'RGMRES', 'CGS', 'BICGSTAB' or 'TFQMR',
or PRECON $\neq$ 'N', 'J' or 'S',
or N $< 1$,
or NNZ $< 1$,
or NNZ $> \mathrm{N}^2$,
or PRECON = 'S' and OMEGA lies outside the interval $(0.0, 2.0)$,
or M $< 1$,
or M $> \min(\mathrm{N}, 50)$, when METHOD = 'RGMRES',
or M $> \min(\mathrm{N}, 10)$, when METHOD = 'BICGSTAB',
or TOL $\geq 1.0$,
or MAXITN $< 1$,
or LWORK is too small.

IFAIL $= 2$

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

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

Therefore a nonzero element has been supplied which does not lie within the matrix $A$, is out of order, or has duplicate row and column indices. Call F11ZNF to reorder and sum or remove duplicates.

IFAIL $= 3$

On entry, the matrix $A$ has a zero diagonal element. Jacobi and SSOR preconditioners are therefore not appropriate for this problem.

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 $= 7$ (F11BRF, F11BSF or F11BTF)

A serious error has occurred in an internal call to one of the specified routines. 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

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 Parallelism and Performance

F11DSF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F11DSF 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

The time taken by F11DSF for each iteration is roughly proportional to NNZ.

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 coefficient matrix $\bar{A} = M^{-1}A$, for some preconditioning matrix $M$.

## 10 Example

This example solves a complex sparse non-Hermitian system of equations using the CGS method, with no preconditioning.

### 10.1 Program Text

```
    Program f11dsfe

!     F11DSF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: f11dsf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: omega, rnorm, tol
      Integer                          :: i, ifail, itn, l, lwork, m, maxitn,  &
                                          n, nnz
      Character (8)                    :: method
      Character (1)                    :: precon
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:), b(:), work(:), x(:)
      Integer, Allocatable             :: icol(:), irow(:), iwork(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max
!     .. Executable Statements ..
      Write (nout,*) 'F11DSF Example Program Results'
      Write (nout,*)
!     Skip heading in data file
      Read (nin,*)

!     Read algorithmic parameters

      Read (nin,*) n
      Read (nin,*) nnz
      Read (nin,*) method, precon
      Read (nin,*) omega
      Read (nin,*) m, tol, maxitn
      l = n
      If (precon=='N' .Or. precon=='n') Then
```

```
        l = 0
      End If
      lwork = max(4*n+m*(m+n+5)+l+121,8*n+l+120,2*n*(m+3)+m*(m+2)+l+120,       &
        11*n+l+120)

      Allocate (a(nnz),b(n),work(lwork),x(n),icol(nnz),irow(nnz),iwork(2*n+1))

!     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)

!     Solve Ax = b using F11DSF

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call f11dsf(method,precon,n,nnz,a,irow,icol,omega,b,m,tol,maxitn,x,      &
        rnorm,itn,work,lwork,iwork,ifail)

      Write (nout,99999) itn
      Write (nout,99998) rnorm
      Write (nout,*)

!     Output x

      Write (nout,*) '                   X'
      Write (nout,99997) x(1:n)

99999 Format (1X,'Converged in',I10,' iterations')
99998 Format (1X,'Final residual norm =',1P,E16.3)
99997 Format (1X,'(',1P,E16.4,',',1P,E16.4,')')
    End Program f11dsfe
```

## 10.2 Program Data

```
F11DSF Example Program Data
  5                       N
 16                       NNZ
  'CGS' 'N'               METHOD, PRECON
  1.05                    OMEGA
  1 1.D-10 1000           M, TOL, MAXITN
 ( 2., 3.)   1    1
 ( 1.,-1.)   1    2
 (-1., 0.)   1    4
 ( 0., 2.)   2    2
 (-2., 1.)   2    3
 ( 1., 0.)   2    5
 ( 0.,-1.)   3    1
 ( 5., 4.)   3    3
 ( 3.,-1.)   3    4
 ( 1., 0.)   3    5
 (-2., 2.)   4    1
 (-3., 1.)   4    4
 ( 0., 3.)   4    5
 ( 4.,-2.)   5    2
 (-2., 0.)   5    3
 (-6., 1.)   5    5      A(I), IROW(I), ICOL(I), I=1,...,NNZ
 ( -3., 3.)
 (-11., 5.)
 ( 23.,48.)
 (-41., 2.)
 (-28.,-31.)             B(I), I=1,...,N
```

```
( 0., 0.)
( 0., 0.)
( 0., 0.)
( 0., 0.)
( 0., 0.)               X(I), I=1,...,N
```

## 10.3 Program Results

```
 F11DSF Example Program Results

 Converged in         5 iterations
 Final residual norm =       1.020E-10

                     X
(      1.0000E+00,      2.0000E+00)
(      2.0000E+00,      3.0000E+00)
(      3.0000E+00,      4.0000E+00)
(      4.0000E+00,      5.0000E+00)
(      5.0000E+00,      6.0000E+00)
```