NAG Library Routine Document

F11DCF

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

F11DCF solves a real sparse nonsymmetric 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 F11DCF (METHOD, N, NNZ, A, LA, IROW, ICOL, IPIVP, IPIVQ,        &
                   ISTR, IDIAG, B, M, TOL, MAXITN, X, RNORM, ITN, WORK,    &
                   LWORK, IFAIL)

INTEGER            N, NNZ, LA, IROW(LA), ICOL(LA), IPIVP(N), IPIVQ(N),     &
                   ISTR(N+1), IDIAG(N), M, MAXITN, ITN, LWORK, IFAIL
REAL (KIND=nag_wp) A(LA), B(N), TOL, X(N), RNORM, WORK(LWORK)
CHARACTER(*)       METHOD
```

3 Description

F11DCF solves a real sparse nonsymmetric linear system of equations:

$$Ax = b,$$

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

F11DCF uses the incomplete LU factorization determined by F11DAF as the preconditioning matrix. A call to F11DCF must always be preceded by a call to F11DAF. Alternative preconditioners for the same storage scheme are available by calling F11DEF.

The matrix A, and the preconditioning matrix M, are represented in coordinate storage (CS) format (see Section 2.1.1 in the F11 Chapter Introduction) in the arrays A, IROW and ICOL, as returned from F11DAF. The array A holds the nonzero entries in these matrices, while IROW and ICOL hold the corresponding row and column indices.

F11DCF is a Black Box routine which calls F11BDF, F11BEF and F11BFF. 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

Salvini S A and Shaw G J (1996) An evaluation of new NAG Library solvers for large sparse unsymmetric linear systems *NAG Technical Report TR2/96*

Sleijpen G L G and Fokkema D R (1993) BiCGSTAB(ℓ) 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 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 (ℓ) method.

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 F11DAF.

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

3: NNZ – INTEGER *Input*

On entry: the number of nonzero elements in the matrix A. This **must** be the same value as was supplied in the preceding call to F11DAF.

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

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

On entry: the values returned in the array A by a previous call to F11DAF.

5: LA – INTEGER *Input*

On entry: the dimension of the arrays A, IROW and ICOL as declared in the (sub)program from which F11DCF is called. This **must** be the same value as was supplied in the preceding call to F11DAF.

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

6: IROW(LA) – INTEGER array *Input*
7: ICOL(LA) – INTEGER array *Input*
8: IPIVP(N) – INTEGER array *Input*
9: IPIVQ(N) – INTEGER array *Input*
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 F11DAF.

IPIVP and IPIVQ are restored on exit.

12: B(N) – REAL (KIND=nag_wp) 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 ℓ of the polynomial Bi-CGSTAB method; otherwise, M is not referenced.

Constraints:

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

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

Constraint: TOL < 1.0.

15: MAXITN – INTEGER *Input*

On entry: the maximum number of iterations allowed.

Constraint: MAXITN ≥ 1.

16: X(N) – REAL (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.

17: 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.

18: ITN – INTEGER *Output*

On exit: the number of iterations carried out.

19: WORK(LWORK) – REAL (KIND=nag_wp) array *Workspace*
20: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F11DCF is called.

Constraints:

if METHOD = 'RGMRES', LWORK $\geq 4 \times \mathrm{N} + \mathrm{M} \times (\mathrm{M} + \mathrm{N} + 5) + 101$;
if METHOD = 'CGS', LWORK $\geq 8 \times \mathrm{N} + 100$;
if METHOD = 'BICGSTAB', LWORK $\geq 2 \times \mathrm{N} \times (\mathrm{M} + 3) + \mathrm{M} \times (\mathrm{M} + 2) + 100$;
if METHOD = 'TFQMR', LWORK $\geq 11 \times \mathrm{N} + 100$.

21: 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 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 $> \min(\text{N}, 50)$, with METHOD $=$ 'RGMRES',
or M $> \min(\text{N}, 10)$, with METHOD $=$ 'BICGSTAB',
or TOL ≥ 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 F11DCF has been preceded by a valid call to F11DAF, 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 F11DCF has been preceded by a valid call to F11DAF 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 (F11BDF, F11BEF or F11BFF)

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

F11DCF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F11DCF 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 F11DCF for each iteration is roughly proportional to the value of NNZC returned from the preceding call to F11DAF.

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 $\bar{A} = \mathrm{M}^{-1}A$.

Some illustrations of the application of F11DCF to linear systems arising from the discretization of two-dimensional elliptic partial differential equations, and to random-valued randomly structured linear systems, can be found in Salvini and Shaw (1996).

10 Example

This example solves a sparse linear system of equations using the CGS method, with incomplete LU preconditioning.

10.1 Program Text

```
    Program f11dcfe

!     F11DCF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: f11daf, f11dcf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: dtol, rnorm, tol
      Integer                          :: i, ifail, itn, la, lfill, liwork,    &
                                          lwork, m, maxitn, n, nnz, nnzc,      &
                                          npivm
      Character (8)                    :: method
      Character (1)                    :: milu, pstrat
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:), b(:), work(:), x(:)
      Integer, Allocatable             :: icol(:), idiag(:), ipivp(:),         &
                                          ipivq(:), irow(:), istr(:), iwork(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max
!     .. Executable Statements ..
      Write (nout,*) 'F11DCF Example Program Results'
      Write (nout,*)
!     Skip heading in data file
      Read (nin,*)

!     Read algorithmic parameters

      Read (nin,*) n, m
      Read (nin,*) nnz
      la = 3*nnz
      liwork = 7*n + 2
      lwork = max(4*n+m*(m+n+5)+101,8*n+100,2*n*(m+3)+m*(m+2)+100,11*n+100)
      Allocate (a(la),b(n),work(lwork),x(n),icol(la),idiag(n),ipivp(n),        &
        ipivq(n),irow(la),istr(n+1),iwork(liwork))
      Read (nin,*) method
      Read (nin,*) lfill, dtol
      Read (nin,*) pstrat
      Read (nin,*) milu
      Read (nin,*) tol, maxitn

!     Read the matrix A

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

!     Read right-hand side vector b and initial approximate solution x

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

!     Calculate incomplete LU factorization

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call f11daf(n,nnz,a,la,irow,icol,lfill,dtol,pstrat,milu,ipivp,ipivq,     &
        istr,idiag,nnzc,npivm,iwork,liwork,ifail)

!     Solve Ax = b using F11DCF

      ifail = 0
      Call f11dcf(method,n,nnz,a,la,irow,icol,ipivp,ipivq,istr,idiag,b,m,tol,  &
```

```
        maxitn,x,rnorm,itn,work,lwork,ifail)

      Write (nout,99999) itn
      If (rnorm<tol) Then
        Write (nout,99996) tol
      Else
        Write (nout,99998) rnorm
      End If
      Write (nout,*)

!     Output solution, x

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

99999 Format (1X,' Converged in',I4,' iterations')
99998 Format (1X,' Final residual norm =',1P,E15.2)
99997 Format (1X,1P,E16.3)
99996 Format (1X,' Final residual norm < tolerance (',1P,E10.3,')')
    End Program f11dcfe
```

10.2 Program Data

```
F11DCF Example Program Data
  8 4                   N, M
  24                    NNZ
  'CGS'                 METHOD
  0  0.0                LFILL, DTOL
  'C'                   PSTRAT
  'N'                   MILU
  1.0D-12   100         TOL, MAXITN
  2.   1    1
 -1.   1    4
  1.   1    8
  4.   2    1
 -3.   2    2
  2.   2    5
 -7.   3    3
  2.   3    6
  3.   4    1
 -4.   4    3
  5.   4    4
  5.   4    7
 -1.   5    2
  8.   5    5
 -3.   5    7
 -6.   6    1
  5.   6    3
  2.   6    6
 -5.   7    3
 -1.   7    5
  6.   7    7
 -1.   8    2
  2.   8    6
  3.   8    8         A(I), IROW(I), ICOL(I), I=1,...,NNZ
  6.   8.  -9.  46.
 17.  21.  22.  34.   B(I), I=1,...,N
  0.   0.   0.   0.
  0.   0.   0.   0.   X(I), I=1,...,N
```

10.3 Program Results

```
 F11DCF Example Program Results

  Converged in   4 iterations
  Final residual norm < tolerance ( 1.000E-12)

        Solution
        1.000E+00
        2.000E+00
```

```
 3.000E+00
 4.000E+00
 5.000E+00
 6.000E+00
 7.000E+00
 8.000E+00
```