NAG Library Routine Document

C05NDF

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

C05NDF is a comprehensive reverse communication routine that finds a solution of a system of nonlinear equations by a modification of the Powell hybrid method.

2 Specification

```
SUBROUTINE C05NDF (IREVCM, N, X, FVEC, XTOL, ML, MU, EPSFCN, DIAG, MODE,        &
                   FACTOR, FJAC, LDFJAC, R, LR, QTF, W, IFAIL)

INTEGER            IREVCM, N, ML, MU, MODE, LDFJAC, LR, IFAIL
REAL (KIND=nag_wp) X(N), FVEC(N), XTOL, EPSFCN, DIAG(N), FACTOR,              &
                   FJAC(LDFJAC,N), R(N*(N+1)/2), QTF(N), W(N,4)
```

3 Description

The system of equations is defined as:

$$f_i(x_1, x_2, \ldots, x_n) = 0, \qquad i = 1, 2, \ldots, n.$$

C05NDF is based on the MINPACK routine HYBRD (see Moré *et al.* (1980)). It chooses the correction at each step as a convex combination of the Newton and scaled gradient directions. The Jacobian is updated by the rank-1 method of Broyden. At the starting point, the Jacobian is approximated by forward differences, but these are not used again until the rank-1 method fails to produce satisfactory progress. For more details see Powell (1970).

4 References

Moré J J, Garbow B S and Hillstrom K E (1980) User guide for MINPACK-1 *Technical Report ANL-80-74* Argonne National Laboratory

Powell M J D (1970) A hybrid method for nonlinear algebraic equations *Numerical Methods for Nonlinear Algebraic Equations* (ed P Rabinowitz) Gordon and Breach

5 Parameters

Note: this routine uses **reverse communication.** Its use involves an initial entry, intermediate exits and re-entries, and a final exit, as indicated by the parameter **IREVCM**. Between intermediate exits and re-entries, **all parameters other than FVEC must remain unchanged**.

1: IREVCM – INTEGER *Input/Output*

On initial entry: must have the value 0.

On intermediate exit: specifies what action you must take before re-entering C05NDF with IREVCM **unchanged**. The value of IREVCM should be interpreted as follows:

IREVCM = 1
: Indicates the start of a new iteration. No action is required by you, but X and FVEC are available for printing.

IREVCM = 2
: Indicates that before re-entry to C05NDF, FVEC must contain the function values $f_i(x)$.

On final exit: IREVCM $= 0$, and the algorithm has terminated.

Constraint: IREVCM $= 0$, 1 or 2.

2: N – INTEGER *Input*

On initial entry: n, the number of equations.

Constraint: N > 0.

3: X(N) – REAL (KIND=nag_wp) array *Input/Output*

On initial entry: an initial guess at the solution vector.

On intermediate exit: contains the current point.

On final exit: the final estimate of the solution vector.

4: FVEC(N) – REAL (KIND=nag_wp) array *Input/Output*

On initial entry: need not be set.

On intermediate re-entry: if IREVCM $= 1$, FVEC must not be changed.

If IREVCM $= 2$, FVEC must be set to the values of the functions computed at the current point X.

On final exit: the function values at the final point, X.

5: XTOL – REAL (KIND=nag_wp) *Input*

On initial entry: the accuracy in X to which the solution is required.

Suggested value: $\sqrt{\epsilon}$, where ϵ is the ***machine precision*** returned by X02AJF.

Constraint: XTOL ≥ 0.0.

6: ML – INTEGER *Input*

On initial entry: the number of subdiagonals within the band of the Jacobian matrix. (If the Jacobian is not banded, or you are unsure, set ML $=$ N $- 1$.)

Constraint: ML ≥ 0.

7: MU – INTEGER *Input*

On initial entry: the number of superdiagonals within the band of the Jacobian matrix. (If the Jacobian is not banded, or you are unsure, set MU $=$ N $- 1$.)

Constraint: MU ≥ 0.

8: EPSFCN – REAL (KIND=nag_wp) *Input*

On initial entry: the order of the largest relative error in the functions. It is used in determining a suitable step for a forward difference approximation to the Jacobian. If EPSFCN is less than ***machine precision*** (returned by X02AJF) then ***machine precision*** is used. Consequently a value of 0.0 will often be suitable.

Suggested value: EPSFCN $= 0.0$.

9: DIAG(N) – REAL (KIND=nag_wp) array *Input/Output*

On initial entry: if MODE $= 2$, DIAG must contain multiplicative scale factors for the variables.

Constraint: DIAG$(i) > 0.0$, for $i = 1, 2, \ldots, n$.

On intermediate exit: the scale factors actually used (computed internally if MODE $\neq 2$).

10: MODE – INTEGER *Input*

On initial entry: indicates whether or not you have provided scaling factors in DIAG.

If MODE $= 2$ the scaling must have been supplied in DIAG.

Otherwise, the variables will be scaled internally.

11: FACTOR – REAL (KIND=nag_wp) *Input*

On initial entry: a quantity to be used in determining the initial step bound. In most cases, FACTOR should lie between 0.1 and 100.0. (The step bound is FACTOR $\times \|\text{DIAG} \times \text{X}\|_2$ if this is nonzero; otherwise the bound is FACTOR.)

Suggested value: FACTOR $= 100.0$.

Constraint: FACTOR > 0.0.

12: FJAC(LDFJAC,N) – REAL (KIND=nag_wp) array *Input/Output*

On initial entry: need not be set.

On intermediate exit: must not be changed.

On final exit: the orthogonal matrix Q produced by the QR factorization of the final approximate Jacobian.

13: LDFJAC – INTEGER *Input*

On initial entry: the first dimension of the array FJAC as declared in the (sub)program from which C05NDF is called.

Constraint: LDFJAC $\geq$ N.

14: R(N $\times$ (N + 1)/2) – REAL (KIND=nag_wp) array *Input/Output*

On initial entry: need not be set.

On intermediate exit: must not be changed.

On final exit: the upper triangular matrix R produced by the QR factorization of the final approximate Jacobian, stored row-wise.

15: LR – INTEGER *Dummy*

This parameter is no longer accessed by C05NDF.

16: QTF(N) – REAL (KIND=nag_wp) array *Input/Output*

On initial entry: need not be set.

On intermediate exit: must not be changed.

On final exit: the vector $Q^{\mathrm{T}} f$.

17: W(N,4) – REAL (KIND=nag_wp) array *Communication Array*

18: IFAIL – INTEGER *Input/Output*

On initial 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, because for this routine the values of the output parameters may be useful even if IFAIL $\neq 0$ on exit, the recommended value is -1. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On final 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, N ≤ 0,
or XTOL < 0.0,
or ML < 0,
or MU < 0,
or FACTOR ≤ 0.0,
or LDFJAC $<$ N,
or MODE $= 2$ and DIAG$(i) \leq 0.0$ for some i, $i = 1, 2, \ldots,$ N.

IFAIL $= 2$

On entry, IREVCM < 0 or IREVCM > 2.

IFAIL $= 3$

No further improvement in the approximate solution X is possible; XTOL is too small.

IFAIL $= 4$

The iteration is not making good progress, as measured by the improvement from the last five Jacobian evaluations.

IFAIL $= 5$

The iteration is not making good progress, as measured by the improvement from the last ten iterations.

A value of IFAIL $= 4$ or 5 may indicate that the system does not have a zero, or that the solution is very close to the origin (see Section 7). Otherwise, rerunning C05NDF from a different starting point may avoid the region of difficulty.

7 Accuracy

If $\hat{x}$ is the true solution and D denotes the diagonal matrix whose entries are defined by the array DIAG, then C05NDF tries to ensure that

$$\|D(x - \hat{x})\|_2 \leq \text{XTOL} \times \|D\hat{x}\|_2.$$

If this condition is satisfied with XTOL $= 10^{-k}$, then the larger components of Dx have k significant decimal digits. There is a danger that the smaller components of Dx may have large relative errors, but the fast rate of convergence of C05NDF usually obviates this possibility.

If XTOL is less than ***machine precision*** and the above test is satisfied with the ***machine precision*** in place of XTOL, then the routine exits with IFAIL $= 3$.

Note: this convergence test is based purely on relative error, and may not indicate convergence if the solution is very close to the origin.

The test assumes that the functions are reasonably well behaved. If this condition is not satisfied, then C05NDF may incorrectly indicate convergence. The validity of the answer can be checked, for example, by rerunning C05NDF with a lower value for XTOL.

8 Further Comments

The time required by C05NDF to solve a given problem depends on n, the behaviour of the functions, the accuracy requested and the starting point. The number of arithmetic operations executed by C05NDF to

process the evaluation of functions in the main program in each exit is about $11.5 \times n^2$. The timing of C05NDF will be strongly influenced by the time spent in the evaluation of the functions.

Ideally the problem should be scaled so that, at the solution, the function values are of comparable magnitude.

The number of function evaluations required to evaluate the Jacobian may be reduced if you can specify ML and MU.

9 Example

This example determines the values $x_1, \ldots, x_9$ which satisfy the tridiagonal equations:

$$\begin{aligned} (3 - 2x_1)x_1 - 2x_2 &= -1, \\ -x_{i-1} + (3 - 2x_i)x_i - 2x_{i+1} &= -1, \qquad i = 2, 3, \ldots, 8 \\ -x_8 + (3 - 2x_9)x_9 &= -1. \end{aligned}$$

9.1 Program Text

```
    Program c05ndfe

!     C05NDF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: c05ndf, dnrm2, nag_wp, x02ajf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: n = 9, nout = 6
      Integer, Parameter               :: ldfjac = n
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: epsfcn, factor, fnorm, xtol
      Integer                          :: icount, ifail, irevcm, j, lr, ml,    &
                                          mode, mu
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: diag(:), fjac(:,:), fvec(:), qtf(:), &
                                          r(:), w(:,:), x(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: sqrt
!     .. Executable Statements ..
      Write (nout,*) 'C05NDF Example Program Results'

      Allocate (diag(n),fjac(ldfjac,n),fvec(n),qtf(n),r(n*(n+ &
        1)/2),w(n,4),x(n))

!     The following starting values provide a rough solution.

      x(1:n) = -1.0E0_nag_wp
      xtol = sqrt(x02ajf())
      diag(1:n) = 1.0E0_nag_wp
      ml = 1
      mu = 1
      epsfcn = 0.0E0_nag_wp
      mode = 2
      factor = 100.0E0_nag_wp
      icount = 0
      irevcm = 0
      ifail = -1

revcomm: Do

        Call c05ndf(irevcm,n,x,fvec,xtol,ml,mu,epsfcn,diag,mode,factor,fjac, &
          ldfjac,r,lr,qtf,w,ifail)

        Select Case (irevcm)
        Case (1)
```

```
          icount = icount + 1

!         Insert print statements here to monitor progess if desired.

          Cycle revcomm
        Case (2)

!         Evaluate functions at given point

          fvec(1:n) = (3.0E0_nag_wp-2.0E0_nag_wp*x(1:n))*x(1:n) + 1.0E0_nag_wp
          fvec(2:n) = fvec(2:n) - x(1:(n-1))
          fvec(1:(n-1)) = fvec(1:(n-1)) - 2.0E0_nag_wp*x(2:n)
          Cycle revcomm
        Case Default
          Exit revcomm
        End Select

      End Do revcomm

      Write (nout,*)

      Select Case (ifail)
      Case (0)
!       The NAG name equivalent of dnrm2 is f06ejf
        fnorm = dnrm2(n,fvec,1)
        Write (nout,99999) 'Final 2-norm of the residuals after', icount, &
          ' iterations is ', fnorm
        Write (nout,*)
        Write (nout,*) 'Final approximate solution'
        Write (nout,99998)(x(j),j=1,n)
      Case (3:)
        Write (nout,*) 'Approximate solution'
        Write (nout,99998)(x(j),j=1,n)
      End Select

99999 Format (1X,A,I4,A,E12.4)
99998 Format (5X,3F12.4)
    End Program c05ndfe
```

9.2 Program Data

None.

9.3 Program Results

```
 C05NDF Example Program Results

 Final 2-norm of the residuals after  11 iterations is   0.1193E-07

 Final approximate solution
          -0.5707     -0.6816     -0.7017
          -0.7042     -0.7014     -0.6919
          -0.6658     -0.5960     -0.4164
```