NAG Library Routine Document

C05NCF

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

C05NCF is a comprehensive routine that finds a solution of a system of nonlinear equations by a modification of the Powell hybrid method.

2 Specification

```
SUBROUTINE C05NCF (FCN, N, X, FVEC, XTOL, MAXFEV, ML, MU, EPSFCN, DIAG,         &
                   MODE, FACTOR, NPRINT, NFEV, FJAC, LDFJAC, R, LR, QTF, W,     &
                   IFAIL)
INTEGER            N, MAXFEV, ML, MU, MODE, NPRINT, NFEV, 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(1,1)
EXTERNAL           FCN
```

3 Description

The system of equations is defined as:

$$f_i(x_1, x_2, \dots, x_n) = 0, \qquad \text{for } i = 1, 2, \dots, n.$$

C05NCF 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

1: FCN – SUBROUTINE, supplied by the user. *External Procedure*

FCN must return the values of the functions f_i at a point x.

The specification of FCN is:

```
SUBROUTINE FCN (N, X, FVEC, IFLAG)
INTEGER            N, IFLAG
REAL (KIND=nag_wp) X(N), FVEC(N)
```

1: N – INTEGER *Input*

On entry: n, the number of equations.

2: X(N) – REAL (KIND=nag_wp) array *Input*

On entry: the components of the point x at which the functions must be evaluated.

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

On entry: if IFLAG $= 0$, FVEC contains the function values $f_i(x)$ and must not be changed.

On exit: if IFLAG > 0 on entry, FVEC must contain the function values $f_i(x)$ (unless IFLAG is set to a negative value by FCN).

4: IFLAG – INTEGER *Input/Output*

On entry: IFLAG ≥ 0.

IFLAG $= 0$
X and FVEC are available for printing (see NPRINT below).

IFLAG > 0
FVEC must be updated.

On exit: in general, IFLAG should not be reset by FCN. If, however, you wish to terminate execution (perhaps because some illegal point X has been reached), then IFLAG should be set to a negative integer. This value will be returned through IFAIL.

FCN must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub)program from which C05NCF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

2: N – INTEGER *Input*

On entry: n, the number of equations.

Constraint: N > 0.

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

On entry: an initial guess at the solution vector.

On exit: the final estimate of the solution vector.

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

On exit: the function values at the final point returned in X.

5: XTOL – REAL (KIND=nag_wp) *Input*

On 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: MAXFEV – INTEGER *Input*

On entry: the maximum number of calls to FCN with IFLAG $\neq 0$. C05NCF will exit with IFAIL $= 2$, if, at the end of an iteration, the number of calls to FCN exceeds MAXFEV.

Suggested value: MAXFEV $= 200 \times (\mathrm{N} + 1)$.

Constraint: MAXFEV > 0.

7: ML – INTEGER *Input*

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

Constraint: $\mathrm{ML} \geq 0$.

8: MU – INTEGER *Input*

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

Constraint: $\mathrm{MU} \geq 0$.

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

On entry: a rough estimate 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: $\mathrm{EPSFCN} = 0.0$.

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

On entry: if $\mathrm{MODE} = 2$, DIAG must contain multiplicative scale factors for the variables.

If $\mathrm{MODE} = 1$, DIAG need not be set.

Constraint: if $\mathrm{MODE} = 2$, $\mathrm{DIAG}(i) > 0.0$, for $i = 1, 2, \ldots, n$.

On exit: the scale factors actually used (computed internally if $\mathrm{MODE} = 1$).

11: MODE – INTEGER *Input*

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

If $\mathrm{MODE} = 2$ the scaling must have been specified in DIAG.

Otherwise, the variables will be scaled internally.

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

On 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 $\mathrm{FACTOR} \times \|\mathrm{DIAG} \times \mathrm{X}\|_2$ if this is nonzero; otherwise the bound is FACTOR.)

Suggested value: $\mathrm{FACTOR} = 100.0$.

Constraint: $\mathrm{FACTOR} > 0.0$.

13: NPRINT – INTEGER *Input*

On entry: indicates whether (and how often) special calls to FCN, with IFLAG set to 0, are to be made for printing purposes.

$\mathrm{NPRINT} \leq 0$
: No calls are made.

$\mathrm{NPRINT} > 0$
: FCN is called at the beginning of the first iteration, every NPRINT iterations thereafter and immediately before the return from C05NCF.

14: NFEV – INTEGER *Output*

On exit: the number of calls made to FCN.

15: FJAC(LDFJAC,N) – REAL (KIND=nag_wp) array *Output*

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

16: LDFJAC – INTEGER *Input*

On entry: the first dimension of the array FJAC as declared in the (sub)program from which C05NCF is called.

Constraint: LDFJAC $\geq$ N.

17: R($\mathrm{N} \times (\mathrm{N}+1)/2$) – REAL (KIND=nag_wp) array *Output*

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

18: LR – INTEGER *Dummy*

This parameter is no longer accessed by C05NCF.

19: QTF(N) – REAL (KIND=nag_wp) array *Output*

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

20: W(1,1) – REAL (KIND=nag_wp) array *Input*

This parameter is no longer accessed by C05NCF. Workspace is provided internally by dynamic allocation instead.

21: 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 < 0

This indicates an exit from C05NCF because you have set IFLAG negative in FCN. The value of IFAIL will be the same as your setting of IFLAG.

IFAIL $= 1$

On entry, N ≤ 0,
or XTOL < 0.0,
or MAXFEV ≤ 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, for $i = 1, 2, \ldots,$ N.

IFAIL $= 2$

There have been at least MAXFEV evaluations of FCN. Consider restarting the calculation from the final point held in X.

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.

IFAIL $= -999$

Internal memory allocation failed.

The values IFAIL $= 4$ and 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 C05NCF 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 C05NCF 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 C05NCF 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 C05NCF may incorrectly indicate convergence. The validity of the answer can be checked, for example, by rerunning C05NCF with a lower value for XTOL.

8 Further Comments

Local workspace arrays of fixed lengths are allocated internally by C05NCF. The total size of these arrays amounts to $4 \times$ N real elements.

The time required by C05NCF 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 C05NCF to process each call of FCN is about $11.5 \times n^2$. Unless FCN can be evaluated quickly, the timing of C05NCF will be strongly influenced by the time spent in FCN.

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

```
!   C05NCF Example Program Text
!   Mark 24 Release. NAG Copyright 2012.

    Module c05ncfe_mod

!     C05NCF Example Program Module:
!            Parameters and User-defined Routines

!     .. Use Statements ..
      Use nag_library, Only: nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter        :: epsfcn = 0.0_nag_wp
      Real (Kind=nag_wp), Parameter        :: factor = 100.0_nag_wp
      Real (Kind=nag_wp), Parameter        :: one = 1.0_nag_wp
      Real (Kind=nag_wp), Parameter        :: three = 3.0_nag_wp
      Real (Kind=nag_wp), Parameter        :: two = 2.0_nag_wp
      Integer, Parameter                   :: maxfev = 2000, ml = 1, mode = 2, &
                                              mu = 1, n = 9, nout = 6,         &
                                              nprint = 0
      Integer, Parameter                   :: nr = n*(n+1)/2
      Integer, Parameter                   :: ldfjac = n
    Contains
      Subroutine fcn(n,x,fvec,iflag)

!       .. Scalar Arguments ..
        Integer, Intent (Inout)              :: iflag
        Integer, Intent (In)                 :: n
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Inout)   :: fvec(n)
        Real (Kind=nag_wp), Intent (In)      :: x(n)
!       .. Executable Statements ..
        If (iflag==0) Then

!         Insert print statements here when NPRINT is positive.

          Continue
        Else
          fvec(1:n) = three*x(1:n) - two*x(1:n)*x(1:n) + one
          fvec(2:n) = fvec(2:n) - x(1:(n-1))
          fvec(1:(n-1)) = fvec(1:(n-1)) - two*x(2:n)
        End If

        Return

      End Subroutine fcn
    End Module c05ncfe_mod
    Program c05ncfe

!     C05NCF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: c05ncf, dnrm2, nag_wp, x02ajf
      Use c05ncfe_mod, Only: epsfcn, factor, fcn, ldfjac, maxfev, ml, mode,    &
```

```
                             mu, n, nout, nprint, nr, one
!     .. Implicit None Statement ..
      Implicit None
!     .. Local Scalars ..
      Real (Kind=nag_wp)                 :: fnorm, xtol
      Integer                            :: ifail, j, lr, nfev
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable    :: diag(:), fjac(:,:), fvec(:),     &
                                            qtf(:), r(:), x(:)
      Real (Kind=nag_wp)                 :: w(1,1)
!     .. Intrinsic Procedures ..
      Intrinsic                          :: sqrt
!     .. Executable Statements ..
      Write (nout,*) 'C05NCF Example Program Results'

      Allocate (diag(n),fjac(ldfjac,n),fvec(n),qtf(n),r(nr),x(n))

!     The following starting values provide a rough solution.

      x(1:n) = -one
      xtol = sqrt(x02ajf())
      diag(1:n) = one

      ifail = -1
      Call c05ncf(fcn,n,x,fvec,xtol,maxfev,ml,mu,epsfcn,diag,mode,factor, &
        nprint,nfev,fjac,ldfjac,r,lr,qtf,w,ifail)

      Select Case (ifail)
      Case (0)
!       The NAG name equivalent of dnrm2 is f06ejf
        fnorm = dnrm2(n,fvec,1)
        Write (nout,*)
        Write (nout,99999) 'Final 2-norm of the residuals =', fnorm
        Write (nout,*)
        Write (nout,99998) 'Number of function evaluations =', nfev
        Write (nout,*)
        Write (nout,*) 'Final approximate solution'
        Write (nout,*)
        Write (nout,99997)(x(j),j=1,n)
      Case (2:)
        Write (nout,*)
        Write (nout,*) 'Approximate solution'
        Write (nout,*)
        Write (nout,99997)(x(j),j=1,n)
      End Select

99999 Format (1X,A,E12.4)
99998 Format (1X,A,I10)
99997 Format (1X,3F12.4)
    End Program c05ncfe
```

9.2 Program Data

None.

9.3 Program Results

```
 C05NCF Example Program Results

 Final 2-norm of the residuals =  0.1193E-07

 Number of function evaluations =        14

 Final approximate solution

      -0.5707     -0.6816     -0.7017
      -0.7042     -0.7014     -0.6919
      -0.6658     -0.5960     -0.4164
```