NAG Library Routine Document

C05QCF

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

C05QCF 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 C05QCF (FCN, N, X, FVEC, XTOL, MAXFEV, ML, MU, EPSFCN, MODE,    &
                   DIAG, FACTOR, NPRINT, NFEV, FJAC, R, QTF, IUSER,        &
                   RUSER, IFAIL)
INTEGER            N, MAXFEV, ML, MU, MODE, NPRINT, NFEV, IUSER(*),        &
                   IFAIL
REAL (KIND=nag_wp) X(N), FVEC(N), XTOL, EPSFCN, DIAG(N), FACTOR,           &
                   FJAC(N,N), R(N*(N+1)/2), QTF(N), RUSER(*)
EXTERNAL           FCN
```

3 Description

The system of equations is defined as:

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

C05QCF 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 Arguments

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

FCN must return the values of the functions f_i at a point x, unless IFLAG $= 0$ on entry to C05QCF.

The specification of FCN is:

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

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: IUSER($*$) – INTEGER array *User Workspace*
5: RUSER($*$) – REAL (KIND=nag_wp) array *User Workspace*

FCN is called with the arguments IUSER and RUSER as supplied to C05QCF. You should use the arrays IUSER and RUSER to supply information to FCN.

6: IFLAG – INTEGER *Input/Output*

On entry: IFLAG ≥ 0.

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

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.

FCN must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub) program from which C05QCF is called. Arguments 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 $\neq 0$. C05QCF will exit with IFAIL $= 2$, if, at the end of an iteration, the number of calls to FCN exceeds MAXFEV.

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

Constraint: $\text{MAXFEV} > 0$.

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 $\text{ML} = \text{N} - 1$.)

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

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 $\text{MU} = \text{N} - 1$.)

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

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: $\text{EPSFCN} = 0.0$.

10: MODE – INTEGER *Input*

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

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

Otherwise, if $\text{MODE} = 1$, the variables will be scaled internally.

Constraint: $\text{MODE} = 1$ or 2.

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

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

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

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

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

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

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

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

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.

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

$\text{NPRINT} > 0$
: FCN is called at the beginning of the first iteration, every NPRINT iterations thereafter and immediately before the return from C05QCF.

14: NFEV – INTEGER *Output*

On exit: the number of calls made to FCN with IFLAG > 0.

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

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

16: R(N $\times$ (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.

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

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

18: IUSER($*$) – INTEGER array *User Workspace*

19: RUSER($*$) – REAL (KIND=nag_wp) array *User Workspace*

IUSER and RUSER are not used by C05QCF, but are passed directly to FCN and should be used to pass information to this routine.

20: 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 $= 2$

There have been at least MAXFEV calls to FCN: MAXFEV $= \langle$*value*$\rangle$. Consider restarting the calculation from the final point held in X.

IFAIL $= 3$

No further improvement in the solution is possible. XTOL is too small: XTOL $= \langle$*value*$\rangle$.

IFAIL $= 4$

The iteration is not making good progress, as measured by the improvement from the last $\langle$*value*$\rangle$ Jacobian evaluations.

IFAIL $= 5$

The iteration is not making good progress, as measured by the improvement from the last $\langle$*value*$\rangle$ iterations.

IFAIL $= 6$

IFLAG was set negative in FCN. IFLAG $= \langle value\rangle$.

IFAIL $= 11$

On entry, N $= \langle value\rangle$.
Constraint: N > 0.

IFAIL $= 12$

On entry, XTOL $= \langle value\rangle$.
Constraint: XTOL ≥ 0.0.

IFAIL $= 13$

On entry, MODE $= \langle value\rangle$.
Constraint: MODE $= 1$ or 2.

IFAIL $= 14$

On entry, FACTOR $= \langle value\rangle$.
Constraint: FACTOR > 0.0.

IFAIL $= 15$

On entry, MODE $= 2$ and DIAG contained a non-positive element.

IFAIL $= 16$

On entry, ML $= \langle value\rangle$.
Constraint: ML ≥ 0.

IFAIL $= 17$

On entry, MU $= \langle value\rangle$.
Constraint: MU ≥ 0.

IFAIL $= 18$

On entry, MAXFEV $= \langle value\rangle$.
Constraint: MAXFEV > 0.

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.

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 C05QCF 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 C05QCF tries to ensure that

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

If this condition is satisfied with $\text{XTOL} = 10^{-k}$, 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 C05QCF 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 $\text{IFAIL} = 3$.

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 convergence test assumes that the functions are reasonably well behaved. If this condition is not satisfied, then C05QCF may incorrectly indicate convergence. The validity of the answer can be checked, for example, by rerunning C05QCF with a lower value for XTOL.

8 Parallelism and Performance

C05QCF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

C05QCF 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

Local workspace arrays of fixed lengths are allocated internally by C05QCF. The total size of these arrays amounts to $4 \times n$ real elements.

The time required by C05QCF 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 C05QCF to process each evaluation of the functions is approximately $11.5 \times n^2$. The timing of C05QCF is strongly influenced by the time spent evaluating 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 accurately.

10 Example

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

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

10.1 Program Text

```
!   C05QCF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

    Module c05qcfe_mod

!     C05QCF Example Program Module:
!            Parameters and User-defined Routines

!     .. Use Statements ..
      Use nag_library, Only: nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Accessibility Statements ..
      Private
      Public                           :: fcn
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter, Public :: epsfcn = 0.0_nag_wp
      Real (Kind=nag_wp), Parameter, Public :: factor = 100.0_nag_wp
      Integer, Parameter, Public       :: maxfev = 2000, ml = 1, mode = 2,     &
                                          mu = 1, n = 9, nout = 6, nprint = 0
    Contains
      Subroutine fcn(n,x,fvec,iuser,ruser,iflag)

!       .. Scalar Arguments ..
        Integer, Intent (Inout)        :: iflag
        Integer, Intent (In)           :: n
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Inout) :: fvec(n), ruser(*)
        Real (Kind=nag_wp), Intent (In) :: x(n)
        Integer, Intent (Inout)        :: iuser(*)
!       .. Executable Statements ..
        If (iflag==0) Then
          If (nprint>0) Then
!           Insert print statements here if desired.
            Continue
          End If
        Else
          fvec(1:n) = (3.0_nag_wp-2.0_nag_wp*x(1:n))*x(1:n) + 1.0_nag_wp
          fvec(2:n) = fvec(2:n) - x(1:(n-1))
          fvec(1:(n-1)) = fvec(1:(n-1)) - 2.0_nag_wp*x(2:n)
        End If
!       Set iflag negative to terminate execution for any reason.
        iflag = 0
        Return
      End Subroutine fcn
    End Module c05qcfe_mod
    Program c05qcfe

!     C05QCF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: c05qcf, dnrm2, nag_wp, x02ajf
      Use c05qcfe_mod, Only: epsfcn, factor, fcn, maxfev, ml, mode, mu, n,     &
                             nout, nprint
!     .. Implicit None Statement ..
      Implicit None
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: fnorm, xtol
      Integer                          :: i, ifail, nfev
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: diag(:), fjac(:,:), fvec(:), qtf(:), &
                                          r(:), x(:)
      Real (Kind=nag_wp)               :: ruser(1)
      Integer                          :: iuser(1)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: sqrt
!     .. Executable Statements ..
      Write (nout,*) 'C05QCF Example Program Results'
      Allocate (diag(n),fjac(n,n),fvec(n),qtf(n),r(n*(n+1)/2),x(n))

!     The following starting values provide a rough solution.

      x(1:n) = -1.0_nag_wp
      xtol = sqrt(x02ajf())
      diag(1:n) = 1.0_nag_wp

      ifail = -1
      Call c05qcf(fcn,n,x,fvec,xtol,maxfev,ml,mu,epsfcn,mode,diag,factor,      &
        nprint,nfev,fjac,r,qtf,iuser,ruser,ifail)

      If (ifail==0 .Or. ifail==2 .Or. ifail==3 .Or. ifail==4 .Or. ifail==5)    &
        Then
        If (ifail==0) Then
!         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'
        Else
          Write (nout,*)
          Write (nout,*) 'Approximate solution'
        End If
        Write (nout,*)
        Write (nout,99997)(x(i),i=1,n)
      End If

99999 Format (1X,A,E12.4)
99998 Format (1X,A,I10)
99997 Format (1X,3F12.4)
    End Program c05qcfe
```

10.2 Program Data

None.

10.3 Program Results

```
 C05QCF 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
```