NAG Fortran Library Routine Document

C05PBF

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

C05PBF is an easy-to-use routine to find a solution of a system of nonlinear equations by a modification of the Powell hybrid method. The user must provide the Jacobian.

2 Specification

```
SUBROUTINE C05PBF(FCN, N, X, FVEC, FJAC, LDFJAC, XTOL, WA, LWA, IFAIL)
INTEGER           N, LDFJAC, LWA, IFAIL
real              X(N), FVEC(N), FJAC(LDFJAC,N), XTOL, WA(LWA)
EXTERNAL          FCN
```

3 Description

The system of equations is defined as:

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

C05PBF is based upon the MINPACK routine HYBRJ1 (Moré *et al.* (1980)). It chooses the correction at each step as a convex combination of the Newton and scaled gradient directions. Under reasonable conditions this guarantees global convergence for starting points far from the solution and a fast rate of convergence. The Jacobian is updated by the rank-1 method of Broyden. At the starting point the Jacobian is calculated, but it is not recalculated 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*

Depending upon the value of IFLAG, FCN must either return the values of the functions f_i at a point x or return the Jacobian at x.

Its specification is:

```
SUBROUTINE FCN(N, X, FVEC, FJAC, LDFJAC, IFLAG)
INTEGER         N, LDFJAC, IFLAG
real            X(N), FVEC(N), FJAC(LDFJAC,N)
```

1: N – INTEGER *Input*

On entry: the number of equations, n.

2: X(N) – ***real*** array *Input*

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

3: FVEC(N) – ***real*** array *Output*

On exit: if IFLAG $= 1$ on entry, FVEC must contain the function values $f_i(x)$ (unless IFLAG is set to a negative value by FCN).

If IFLAG $= 2$ on entry, FVEC must not be changed.

4: FJAC(LDFJAC,N) – ***real*** array *Output*

On exit: if IFLAG $= 2$ on entry, FJAC(i,j) must contain the value of $\frac{\partial f_i}{\partial x_j}$ at the point x, for $i,j = 1,2,\ldots,n$ (unless IFLAG is set to a negative value by FCN).

If IFLAG $= 1$ on entry, FJAC must not be changed.

5: LDFJAC – INTEGER *Input*

On entry: the first dimension of FJAC.

6: IFLAG – INTEGER *Input/Output*

On entry: IFLAG $= 1$ or 2:

if IFLAG $= 1$, FVEC is to be updated;

if IFLAG $= 2$, FJAC is to be updated.

On exit: in general, IFLAG should not be reset by FCN. If, however, the user wishes 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 be declared as EXTERNAL in the (sub)program from which C05PBF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

2: N – INTEGER *Input*

On entry: the number of equations, n.

Constraint: N > 0.

3: X(N) – ***real*** 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*** array *Output*

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

5: FJAC(LDFJAC,N) – ***real*** array *Output*

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

6: LDFJAC – INTEGER *Input*

On entry: the first dimension of the array FJAC as declared in the (sub)program from which C05PBF is called.

Constraint: LDFJAC $\geq$ N.

7: XTOL – ***real*** *Input*

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

Suggested value: the square root of the ***machine precision***.

Constraint: XTOL ≥ 0.0.

8: WA(LWA) – ***real*** array *Workspace*

9: LWA – INTEGER *Input*

On entry: the dimension of the array WA.

Constraint: LWA $\geq \mathrm{N} \times (\mathrm{N} + 13)/2$.

10: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL $= 0$ unless the routine detects an error (see Section 6).

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, for users 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.**

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

A negative value of IFAIL indicates an exit from C05PBF because the user has set IFLAG negative in FCN. The value of IFAIL will be the same as the user's setting of IFLAG.

IFAIL $= 1$

On entry, N ≤ 0,
or LDFJAC $<$ N,
or XTOL < 0.0,
or LWA $< \mathrm{N} \times (\mathrm{N} + 13)/2$.

IFAIL $= 2$

There have been $100 \times (\mathrm{N} + 1)$ evaluations of the functions. 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. This failure exit 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 C05PBF from a different starting point may avoid the region of difficulty.

7 Accuracy

If $\hat{x}$ is the true solution, C05PBF tries to ensure that

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

If this condition is satisfied with $\text{XTOL} = 10^{-k}$, then the larger components of x have k significant decimal digits. There is a danger that the smaller components of x may have large relative errors, but the fast rate of convergence of C05PBF usually avoids the 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 and Jacobian are coded consistently and that the functions are reasonably well behaved. If these conditions are not satisfied then C05PBF may incorrectly indicate convergence. The coding of the Jacobian can be checked using C05ZAF. If the Jacobian is coded correctly, then the validity of the answer can be checked by rerunning C05PBF with a tighter tolerance.

8 Further Comments

The time required by C05PBF 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 C05PBF is about $11.5 \times n^2$ to process each evaluation of the functions and about $1.3 \times n^3$ to process each evaluation of the Jacobian. Unless FCN can be evaluated quickly, the timing of C05PBF 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.

9 Example

To determine the values $x_1, \ldots, x_9$ which satisfy the tridiagonal equations:

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

9.1 Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     C05PBF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          N, LDFJAC, LWA
      PARAMETER        (N=9,LDFJAC=N,LWA=(N*(N+13))/2)
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Local Scalars ..
      real             FNORM, TOL
      INTEGER          IFAIL, J
*     .. Local Arrays ..
      real             FJAC(LDFJAC,N), FVEC(N), WA(LWA), X(N)
*     .. External Functions ..
      real             F06EJF, X02AJF
      EXTERNAL         F06EJF, X02AJF
*     .. External Subroutines ..
      EXTERNAL         C05PBF, FCN
*     .. Intrinsic Functions ..
      INTRINSIC        SQRT
*     .. Executable Statements ..
```

```
      WRITE (NOUT,*) 'C05PBF Example Program Results'
      WRITE (NOUT,*)
*     The following starting values provide a rough solution.
      DO 20 J = 1, N
         X(J) = -1.0e0
   20 CONTINUE
      TOL = SQRT(X02AJF())
      IFAIL = 1
*
      CALL C05PBF(FCN,N,X,FVEC,FJAC,LDFJAC,TOL,WA,LWA,IFAIL)
*
      IF (IFAIL.EQ.0) THEN
         FNORM = F06EJF(N,FVEC,1)
         WRITE (NOUT,99999) 'Final 2-norm of the residuals =', FNORM
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Final approximate solution'
         WRITE (NOUT,*)
         WRITE (NOUT,99998) (X(J),J=1,N)
      ELSE
         WRITE (NOUT,99997) 'IFAIL = ', IFAIL
         IF (IFAIL.GE.2) THEN
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Approximate solution'
            WRITE (NOUT,*)
            WRITE (NOUT,99998) (X(J),J=1,N)
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,e12.4)
99998 FORMAT (1X,3F12.4)
99997 FORMAT (1X,A,I2)
      END
*
      SUBROUTINE FCN(N,X,FVEC,FJAC,LDFJAC,IFLAG)
*     .. Parameters ..
      real              ZERO, ONE, TWO, THREE, FOUR
      PARAMETER         (ZERO=0.0e0,ONE=1.0e0,TWO=2.0e0,THREE=3.0e0,
     +                  FOUR=4.0e0)
*     .. Scalar Arguments ..
      INTEGER           IFLAG, LDFJAC, N
*     .. Array Arguments ..
      real              FJAC(LDFJAC,N), FVEC(N), X(N)
*     .. Local Scalars ..
      INTEGER           J, K
*     .. Executable Statements ..
      IF (IFLAG.NE.2) THEN
         DO 20 K = 1, N
            FVEC(K) = (THREE-TWO*X(K))*X(K) + ONE
            IF (K.GT.1) FVEC(K) = FVEC(K) - X(K-1)
            IF (K.LT.N) FVEC(K) = FVEC(K) - TWO*X(K+1)
   20    CONTINUE
      ELSE
         DO 60 K = 1, N
            DO 40 J = 1, N
               FJAC(K,J) = ZERO
   40       CONTINUE
            FJAC(K,K) = THREE - FOUR*X(K)
            IF (K.GT.1) FJAC(K,K-1) = -ONE
            IF (K.LT.N) FJAC(K,K+1) = -TWO
   60    CONTINUE
      END IF
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 C05PBF Example Program Results

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

 Final approximate solution

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