NAG Toolbox for MATLAB

c05nb

1 Purpose

c05nb is an easy-to-use function to find a solution of a system of nonlinear equations by a modification of the Powell hybrid method.

2 Syntax

```
[x, fvec, ifail] = c05nb(fcn, x, 'n', n, 'xtol', xtol)
```

3 Description

The system of equations is defined as:

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

c05nb is based on the MINPACK routine HYBRD1 (see 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 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

5.1 Compulsory Input Parameters

1: **fcn – string containing name of m-file**

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

Its specification is:

```
[fvec, iflag] = fcn(n, x, iflag)
```

Input Parameters

1: **n – int32 scalar**

n, the number of equations.

2: **x(n) – double array**

The components of the point x at which the functions must be evaluated.

3: **iflag – int32 scalar**

iflag > 0.

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**.

Output Parameters

1: **fvec(n) – double array**

The function values $f_i(x)$ (unless **iflag** is set to a negative value by **fcn**).

2: **iflag – int32 scalar**

iflag > 0.

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**.

2: **x(n) – double array**

An initial guess at the solution vector.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The dimension of the arrays **x**, **fvec**. (An error is raised if these dimensions are not equal.)

n, the number of equations.

Constraint: **n** > 0.

2: **xtol – double scalar**

The accuracy in **x** to which the solution is required.

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

Default: $\sqrt{}$

Constraint: **xtol** ≥ 0.0.

5.3 Input Parameters Omitted from the MATLAB Interface

wa, lwa

5.4 Output Parameters

1: **x(n) – double array**

The final estimate of the solution vector.

2: **fvec(n) – double array**

The function values at the final point, **x**.

3: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail < 0

You have set **iflag** negative in the user-supplied (sub)program **fcn**. The value of **ifail** will be the same as your setting of **iflag**.

ifail $= 1$

On entry, $\mathbf{n} \leq 0$,
or $\mathbf{xtol} < 0.0$,
or $\mathbf{lwa} < \mathbf{n} \times (3 \times \mathbf{n} + 13)/2$.

ifail $= 2$

There have been at least $200 \times (\mathbf{n} + 1)$ evaluations of user-supplied (sub)program **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. 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 c05nb from a different starting point may avoid the region of difficulty.

7 Accuracy

If $\hat{x}$ is the true solution, c05nb tries to ensure that

$$\|x - \hat{x}\| \leq \mathbf{xtol} \times \|\hat{x}\|.$$

If this condition is satisfied with $\mathbf{xtol} = 10^{-k}$, 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 c05nb usually avoids 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 function 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 c05nb may incorrectly indicate convergence. The validity of the answer can be checked, for example, by rerunning c05nb with a tighter tolerance.

8 Further Comments

The time required by c05nb 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 c05nb to process each call of user-supplied (sub)program **fcn** is about $11.5 \times n^2$. Unless **fcn** can be evaluated quickly, the timing of c05nb 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

c05nb_fcn.m

```
function [fvec, iflag] = c05nb_fcn(n,x,iflag)
  for k = 1:n
    fvec(k) = (3.0-2.0*x(k))*x(k)+1.0;
    if k > 1
      fvec(k) = fvec(k) - x(k-1);
    end
    if k < n
      fvec(k) = fvec(k) - 2*x(k+1);
    end
  end
```

```
x = [-1;
     -1;
     -1;
     -1;
     -1;
     -1;
     -1;
     -1;
     -1];
[xOut, fvec, ifail] = c05nb('c05nb_fcn', x)
```

```
xOut =
   -0.5707
   -0.6816
   -0.7017
   -0.7042
   -0.7014
   -0.6919
   -0.6658
   -0.5960
   -0.4164
fvec =
   1.0e-08 *
    0.6560
   -0.4175
   -0.5193
   -0.2396
    0.2022
    0.4818
    0.2580
   -0.3884
   -0.0136
ifail =
           0
```