# NAG Toolbox

# nag_roots_sys_func_rcomm (c05qd)

## 1 Purpose

nag_roots_sys_func_rcomm (c05qd) is a comprehensive reverse communication function that finds a solution of a system of nonlinear equations by a modification of the Powell hybrid method.

## 2 Syntax

```
[irevcm, x, fvec, diag, fjac, r, qtf, iwsav, rwsav, ifail] =
nag_roots_sys_func_rcomm(irevcm, x, fvec, ml, mu, mode, diag, fjac, r, qtf,
iwsav, rwsav, 'n', n, 'xtol', xtol, 'epsfcn', epsfcn, 'factor', factor)
```

```
[irevcm, x, fvec, diag, fjac, r, qtf, iwsav, rwsav, ifail] = c05qd(irevcm, x,
fvec, ml, mu, mode, diag, fjac, r, qtf, iwsav, rwsav, 'n', n, 'xtol', xtol,
'epsfcn', epsfcn, 'factor', factor)
```

## 3 Description

The system of equations is defined as:

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

nag_roots_sys_func_rcomm (c05qd) 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 function uses **reverse communication.** Its use involves an initial entry, intermediate exits and re-entries, and a final exit, as indicated by the argument **irevcm**. Between intermediate exits and re-entries, **all arguments other than fvec must remain unchanged**.

### 5.1 Compulsory Input Parameters

1: **irevcm** – INTEGER

*On initial entry*: must have the value 0.

*Constraint*: **irevcm** $= 0$, 1 or 2.

2: **x**(**n**) – REAL (KIND=nag_wp) array

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

3: **fvec**(**n**) – REAL (KIND=nag_wp) array

*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**.

4: **ml** – INTEGER

*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** $\geq 0$.

5: **mu** – INTEGER

*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** $\geq 0$.

6: **mode** – INTEGER

*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, if **mode** $= 1$, the variables will be scaled internally.

*Constraint*: **mode** $= 1$ or $2$.

7: **diag**(**n**) – REAL (KIND=nag_wp) array

If **mode** $= 2$, **diag** must contain multiplicative scale factors for the variables.

If **mode** $= 1$, **diag** need not be set.

*Constraint*: if **mode** $= 2$, **diag**$(i) > 0.0$, for $i = 1, 2, \ldots, n$.

8: **fjac**(**n**, **n**) – REAL (KIND=nag_wp) array

*On initial entry*: need not be set.

9: **r**(**n** $\times$ (**n** $+$ **1**)/**2**) – REAL (KIND=nag_wp) array

*On initial entry*: need not be set.

10: **qtf**(**n**) – REAL (KIND=nag_wp) array

*On initial entry*: need not be set.

11: **iwsav**(**17**) – INTEGER array
12: **rwsav**(**4** $\times$ **n** $+$ **10**) – REAL (KIND=nag_wp) array

The arrays **iwsav** and **rwsav** **must not** be altered between calls to nag_roots_sys_func_rcomm (c05qd).

## 5.2 Optional Input Parameters

1: **n** – INTEGER

*Default*: the dimension of the arrays **x**, **fvec**, **diag**, **qtf** and the first dimension of the array **fjac** and the second dimension of the array **fjac**. (An error is raised if these dimensions are not equal.)

$n$, the number of equations.

*Constraint*: **n** $> 0$.

2: **xtol** – REAL (KIND=nag_wp)

*Suggested value*: $\sqrt{\epsilon}$, where $\epsilon$ is the ***machine precision*** returned by nag_machine_precision (x02aj).

*Default*: $\sqrt{\textit{\textbf{machine precision}}}$

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

*Constraint*: **xtol** $\geq 0.0$.

3: **epsfcn** – REAL (KIND=nag_wp)

*Suggested value*: **epsfcn** $= 0.0$.

*Default*: 0.0

*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 nag_machine_precision (x02aj)) then ***machine precision*** is used. Consequently a value of 0.0 will often be suitable.

4: **factor** – REAL (KIND=nag_wp)

*Suggested value*: **factor** $= 100.0$.

*Default*: 100.0

*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** $\times$ $\|$**diag** $\times$ **x**$\|_2$ if this is nonzero; otherwise the bound is **factor**.)

*Constraint*: **factor** $> 0.0$.

## 5.3 Output Parameters

1: **irevcm** – INTEGER

*On intermediate exit*: specifies what action you must take before re-entering nag_roots_sys_func_rcomm (c05qd) 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 nag_roots_sys_func_rcomm (c05qd), **fvec** must contain the function values $f_i(x)$.

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

2: **x**(**n**) – REAL (KIND=nag_wp) array

*On intermediate exit*: contains the current point.

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

3: **fvec**(**n**) – REAL (KIND=nag_wp) array

*On final exit*: the function values at the final point, **x**.

4: **diag**(**n**) – REAL (KIND=nag_wp) array

The scale factors actually used (computed internally if **mode** $= 1$).

5: **fjac**($\mathbf{n},\mathbf{n}$) – REAL (KIND=nag_wp) array

*On intermediate exit*: must not be changed.

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

6: **r**($\mathbf{n} \times (\mathbf{n} + \mathbf{1})/\mathbf{2}$) – REAL (KIND=nag_wp) array

*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.

7: **qtf**($\mathbf{n}$) – REAL (KIND=nag_wp) array

*On intermediate exit*: must not be changed.

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

8: **iwsav**($\mathbf{17}$) – INTEGER array
9: **rwsav**($\mathbf{4} \times \mathbf{n} + \mathbf{10}$) – REAL (KIND=nag_wp) array

10: **ifail** – INTEGER

*On final exit*: **ifail** $= 0$ unless the function detects an error (see Section 5).

# 6 Error Indicators and Warnings

Errors or warnings detected by the function:

**ifail** $= 2$

Constraint: **irevcm** $= 0$, 1 or 2.

**ifail** $= 3$ (*warning*)

No further improvement in the solution is possible.

**ifail** $= 4$ (*warning*)

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

**ifail** $= 5$ (*warning*)

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

**ifail** $= 11$

Constraint: **n** $> 0$.

**ifail** $= 12$

Constraint: **xtol** $\geq 0.0$.

**ifail** $= 13$

Constraint: **mode** $= 1$ or 2.

**ifail** $= 14$

Constraint: **factor** $> 0.0$.

**ifail** $= 15$

On entry, **mode** $= 2$ and **diag** contained a non-positive element.

**ifail** $= 16$

Constraint: **ml** $\geq 0$.

**ifail** $= 17$

Constraint: **mu** $\geq 0$.

**ifail** $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

**ifail** $= -399$

Your licence key may have expired or may not have been installed correctly.

**ifail** $= -999$

Dynamic memory allocation failed.

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 nag_roots_sys_func_rcomm (c05qd) 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 nag_roots_sys_func_rcomm (c05qd) tries to ensure that

$$\|D(x - \hat{x})\|_2 \leq \mathbf{xtol} \times \|D\hat{x}\|_2.$$

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 nag_roots_sys_func_rcomm (c05qd) 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 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 convergence test assumes that the functions are reasonably well behaved. If this condition is not satisfied, then nag_roots_sys_func_rcomm (c05qd) may incorrectly indicate convergence. The validity of the answer can be checked, for example, by rerunning nag_roots_sys_func_rcomm (c05qd) with a lower value for **xtol**.

## 8 Further Comments

The time required by nag_roots_sys_func_rcomm (c05qd) 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 nag_roots_sys_func_rcomm (c05qd) to process the evaluation of functions in the main program in each exit is approximately $11.5 \times n^2$. The timing of nag_roots_sys_func_rcomm (c05qd) 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.

# 9 Example

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

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

## 9.1 Program Text

```
function c05qd_example

fprintf('c05qd example results\n\n');

% The following starting values provide a rough solution.
x = -ones(9, 1);
diagnl = ones(9,1);
ml = nag_int(1);
mu = nag_int(1);
mode = nag_int(2);
irevcm = nag_int(0);
fjac = zeros(9, 9);
fvec = zeros(9, 1);
qtf = zeros(9, 1);
r = zeros(45, 1);
rwsav = zeros(46, 1);
iwsav = zeros(17, 1, nag_int_name);

icount = nag_int(0);
first = true;
while (irevcm ~= 0 || first )
  first = false;
  [irevcm, x, fvec, diag, fjac, r, qtf, iwsav, rwsav, ifail] = ...
      c05qd(irevcm, x, fvec, ml, mu, mode, diagnl, fjac, r, qtf, iwsav, rwsav);

  switch irevcm
    case {1}
      icount = icount + 1;
    case {2}
      fvec(1:9) = (3.0-2.0.*x).*x + 1.0;
      fvec(2:9) = fvec(2:9) - x(1:8);
      fvec(1:8) = fvec(1:8) - 2.0.*x(2:9);
  end
end

switch ifail
  case {0}
    fprintf('\nFinal 2-norm of the residuals after %d iterations = %12.4e\n',...
      icount, norm(fvec));
    fprintf('\nFinal approximate solution\n');
    disp(x);
  case {2, 3, 4}
    fprintf('\nApproximate solution\n');
    disp(x);
end
```

## 9.2 Program Results

```
c05qd example results

Final 2-norm of the residuals after 11 iterations =   1.1926e-08

Final approximate solution
   -0.5707
   -0.6816
   -0.7017
   -0.7042
```

```
  -0.7014
  -0.6919
  -0.6658
  -0.5960
  -0.4164
```