NAG Library Routine Document

E02GCF

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

E02GCF calculates an l_∞ solution to an over-determined system of linear equations.

2 Specification

```
SUBROUTINE E02GCF (M, N, SDA, LDA, A, B, TOL, RELERR, X, RESMAX, IRANK,         &
                   ITER, IFAIL)

INTEGER            M, N, SDA, LDA, IRANK, ITER, IFAIL
REAL (KIND=nag_wp) A(LDA,SDA), B(M), TOL, RELERR, X(N), RESMAX
```

3 Description

Given a matrix A with m rows and n columns $(m \geq n)$ and a vector b with m elements, the routine calculates an l_∞ solution to the over-determined system of equations

$$Ax = b.$$

That is to say, it calculates a vector x, with n elements, which minimizes the l_∞ norm of the residuals (the absolutely largest residual)

$$r(x) = \max_{1 \leq i \leq m} |r_i|$$

where the residuals r_i are given by

$$r_i = b_i - \sum_{j=1}^{n} a_{ij} x_j, \qquad i = 1, 2, \ldots, m.$$

Here a_{ij} is the element in row i and column j of A, b_i is the ith element of b and x_j the jth element of x. The matrix A need not be of full rank. The solution is not unique in this case, and may not be unique even if A is of full rank.

Alternatively, in applications where a complete minimization of the l_∞ norm is not necessary, you may obtain an approximate solution, usually in shorter time, by giving an appropriate value to the parameter RELERR.

Typically in applications to data fitting, data consisting of m points with coordinates (t_i, y_i) is to be approximated in the l_∞ norm by a linear combination of known functions $\phi_j(t)$,

$$\alpha_1 \phi_1(t) + \alpha_2 \phi_2(t) + \cdots + \alpha_n \phi_n(t).$$

This is equivalent to finding an l_∞ solution to the over-determined system of equations

$$\sum_{j=1}^{n} \phi_j(t_i) \alpha_j = y_i, \qquad i = 1, 2, \ldots, m.$$

Thus if, for each value of i and j the element a_{ij} of the matrix A above is set equal to the value of $\phi_j(t_i)$ and b_i is set equal to y_i, the solution vector x will contain the required values of the α_j. Note that the independent variable t above can, instead, be a vector of several independent variables (this includes the case where each ϕ_i is a function of a different variable, or set of variables).

The algorithm is a modification of the simplex method of linear programming applied to the dual formation of the l_∞ problem (see Barrodale and Phillips (1974) and Barrodale and Phillips (1975)). The modifications are designed to improve the efficiency and stability of the simplex method for this particular application.

4 References

Barrodale I and Phillips C (1974) An improved algorithm for discrete Chebyshev linear approximation *Proc. 4th Manitoba Conf. Numerical Mathematics* 177–190 University of Manitoba, Canada

Barrodale I and Phillips C (1975) Solution of an overdetermined system of linear equations in the Chebyshev norm [F4] (Algorithm 495) *ACM Trans. Math. Software* **1(3)** 264–270

5 Parameters

1: M – INTEGER *Input*

On entry: the number of equations, m (the number of rows of the matrix A).

Constraint: $\mathrm{M} \geq \mathrm{N}$.

2: N – INTEGER *Input*

On entry: the number of unknowns, n (the number of columns of the matrix A).

Constraint: $\mathrm{N} \geq 1$.

3: SDA – INTEGER *Input*

On entry: the second dimension of the array A as declared in the (sub)program from which E02GCF is called.

Constraint: $\mathrm{SDA} \geq \mathrm{M} + 1$.

4: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which E02GCF is called.

Constraint: $\mathrm{LDA} \geq \mathrm{N} + 3$.

5: A(LDA,SDA) – REAL (KIND=nag_wp) array *Input/Output*

On entry: $\mathrm{A}(j,i)$ must contain a_{ij}, the element in the ith row and jth column of the matrix A, for $i = 1,2,\ldots,m$ and $j = 1,2,\ldots,n$, (that is, the **transpose** of the matrix). The remaining elements need not be set. Preferably, the columns of the matrix A (rows of the parameter A) should be scaled before entry: see Section 7.

On exit: contains the last simplex tableau.

6: B(M) – REAL (KIND=nag_wp) array *Input/Output*

On entry: $\mathrm{B}(i)$ must contain b_i, the ith element of the vector b, for $i = 1,2,\ldots,m$.

On exit: the ith residual r_i corresponding to the solution vector x, for $i = 1,2,\ldots,m$. Note however that these residuals may contain few significant figures, especially when RESMAX is within one or two orders of magnitude of TOL. Indeed if $\mathrm{RESMAX} \leq \mathrm{TOL}$, the elements $\mathrm{B}(i)$ may all be set to zero. It is therefore often advisable to compute the residuals directly.

7: TOL – REAL (KIND=nag_wp) *Input*

On entry: a threshold below which numbers are regarded as zero. The recommended threshold value is $10.0 \times \epsilon$, where ϵ is the ***machine precision***. If $\mathrm{TOL} \leq 0.0$ on entry, the recommended value is used within the routine. If premature termination occurs, a larger value for TOL may result in a valid solution.

Suggested value: 0.0.

8: RELERR – REAL (KIND=nag_wp) *Input/Output*

On entry: must be set to a bound on the relative error acceptable in the maximum residual at the solution.

If RELERR ≤ 0.0, then the l_∞ solution is computed, and RELERR is set to 0.0 on exit.

If RELERR > 0.0, then the routine obtains instead an approximate solution for which the largest residual is less than $1.0 + \mathrm{RELERR}$ times that of the l_∞ solution; on exit, RELERR contains a smaller value such that the above bound still applies. (The usual result of this option, say with RELERR $= 0.1$, is a saving in the number of simplex iterations).

On exit: is altered as described above.

9: X(N) – REAL (KIND=nag_wp) array *Output*

On exit: if IFAIL $= 0$ or 1, X(j) contains the jth element of the solution vector x, for $j = 1, 2, \ldots, n$. Whether this is an l_∞ solution or an approximation to one, depends on the value of RELERR on entry.

10: RESMAX – REAL (KIND=nag_wp) *Output*

On exit: if IFAIL $= 0$ or 1, RESMAX contains the absolute value of the largest residual(s) for the solution vector x. (See B.)

11: IRANK – INTEGER *Output*

On exit: if IFAIL $= 0$ or 1, IRANK contains the computed rank of the matrix A.

12: ITER – INTEGER *Output*

On exit: if IFAIL $= 0$ or 1, ITER contains the number of iterations taken by the simplex method.

13: 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, because for this routine the values of the output parameters may be useful even if IFAIL $\neq 0$ on exit, the recommended value is -1. **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).

Note: E02GCF may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the routine:

IFAIL $= 1$

An optimal solution has been obtained but this may not be unique (perhaps simply because the matrix A is not of full rank, i.e., IRANK $<$ N).

IFAIL $= 2$

The calculations have terminated prematurely due to rounding errors. Experiment with larger values of TOL or try rescaling the columns of the matrix (see Section 8).

IFAIL $= 3$

On entry, LDA $< \mathrm{N} + 3$,
or SDA $< \mathrm{M} + 1$,
or M $<$ N,
or N < 1.

7 Accuracy

Experience suggests that the computational accuracy of the solution x is comparable with the accuracy that could be obtained by applying Gaussian elimination with partial pivoting to the $n + 1$ equations which have residuals of largest absolute value. The accuracy therefore varies with the conditioning of the problem, but has been found generally very satisfactory in practice.

8 Further Comments

The effects of m and n on the time and on the number of iterations in the simplex method vary from problem to problem, but typically the number of iterations is a small multiple of n and the total time is approximately proportional to mn^2.

It is recommended that, before the routine is entered, the columns of the matrix A are scaled so that the largest element in each column is of the order of unity. This should improve the conditioning of the matrix, and also enable the parameter TOL to perform its correct function. The solution x obtained will then, of course, relate to the scaled form of the matrix. Thus if the scaling is such that, for each $j = 1, 2, \ldots, n$, the elements of the jth column are multiplied by the constant k_j, the element x_j of the solution vector x must be multiplied by k_j if it is desired to recover the solution corresponding to the original matrix A.

9 Example

This example approximates a set of data by a curve of the form

$$y = Ke^t + Le^{-t} + M$$

where K, L and M are unknown. Given values y_i at 5 points t_i we may form the over-determined set of equations for K, L and M

$$e^{t_i}K + e^{-t_i}L + M = y_i, \qquad i = 1, 2, \ldots, 5.$$

E02GCF is used to solve these in the l_∞ sense.

9.1 Program Text

```
    Program e02gcfe

!     E02GCF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: e02gcf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter              :: n = 3, nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)              :: relerr, resmax, t, tol
      Integer                         :: i, ifail, irank, iter, lda, m, sda
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable :: a(:,:), b(:), x(:)
!     .. Intrinsic Procedures ..
      Intrinsic                       :: exp
!     .. Executable Statements ..
      Write (nout,*) 'E02GCF Example Program Results'
```

```
!     Skip heading in data file
      Read (nin,*)

      Read (nin,*) m
      lda = n + 3
      sda = m + 1
      Allocate (a(lda,sda),b(m),x(n))

      Do i = 1, m
        Read (nin,*) t, b(i)
        a(1,i) = exp(t)
        a(2,i) = exp(-t)
        a(3,i) = 1.0E0_nag_wp
      End Do

      tol = 0.0E0_nag_wp
      relerr = 0.0E0_nag_wp

      ifail = -1
      Call e02gcf(m,n,sda,lda,a,b,tol,relerr,x,resmax,irank,iter,ifail)

      Select Case (ifail)
      Case (0,1)
        Write (nout,*)
        Write (nout,99999) 'RESMAX = ', resmax, '  Rank = ', irank, &
          '  Iterations = ', iter, '  IFAIL =', ifail
        Write (nout,*)
        Write (nout,*) 'Solution'
        Write (nout,99998) x(1:n)
      End Select

99999 Format (1X,A,E10.2,3(A,I5))
99998 Format (1X,6F10.4)
    End Program e02gcfe
```

9.2 Program Data

```
E02GCF Example Program Data
  5
   0.0 4.501
   0.2 4.360
   0.4 4.333
   0.6 4.418
   0.8 4.625
```

9.3 Program Results

```
 E02GCF Example Program Results

 RESMAX =   0.10E-02  Rank =     3  Iterations =     4  IFAIL =    0

 Solution
     1.0049    2.0149    1.4822
```