NAG Toolbox for MATLAB

e02ga

1 Purpose

e02ga calculates an l_1 solution to an over-determined system of linear equations.

2 Syntax

```
[a, b, x, resid, irank, iter, ifail] = e02ga(a, b, 'm', m, 'nplus2',
nplus2, 'toler', toler)
```

3 Description

Given a matrix A with m rows and n columns $(m \geq n)$ and a vector b with m elements, the function calculates an l_1 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_1 norm (the sum of the absolute values) of the residuals

$$r(x) = \sum_{i=1}^{m} |r_i|,$$

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.

Typically in applications to data fitting, data consisting of m points with co-ordinates (t_i, y_i) are to be approximated in the l_1 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 fitting an l_1 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 in the previous paragraph 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 primal formulation of the l_1 problem (see Barrodale and Roberts 1973 and Barrodale and Roberts 1974). The modification allows several neighbouring simplex vertices to be passed through in a single iteration, providing a substantial improvement in efficiency.

4 References

Barrodale I and Roberts F D K 1973 An improved algorithm for discrete l_1 linear approximation *SIAM J. Numer. Anal.* **10** 839–848

Barrodale I and Roberts F D K 1974 Solution of an overdetermined system of equations in the l_1-norm *Comm. ACM* **17 (6)** 319–320

5 Parameters

5.1 Compulsory Input Parameters

1: **a(lda,nplus2) – double array**

a(i,j) must contain a_{ij}, the element in the ith row and jth column of the matrix A, for $i = 1,2,\ldots,m$ and $j = 1,2,\ldots,n$. The remaining elements need not be set.

2: **b(m) – double array**

b(i) must contain b_i, the ith element of the vector b, for $i = 1,2,\ldots,m$.

5.2 Optional Input Parameters

1: **m – int32 scalar**

Default: The dimension of the array **b**.

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

Constraint: $\mathbf{m} \geq n \geq 1$.

2: **nplus2 – int32 scalar**

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

$n+2$, where n is the number of unknowns (the number of columns of the matrix A).

Constraint: $3 \leq \mathbf{nplus2} \leq \mathbf{m}+2$.

3: **toler – double scalar**

A nonnegative value. In general **toler** specifies a threshold below which numbers are regarded as zero. The recommended threshold value is $\epsilon^{2/3}$ where ϵ is the ***machine precision***. The recommended value can be computed within the function by setting **toler** to zero. If premature termination occurs a larger value for **toler** may result in a valid solution.

Suggested value: 0.0.

Default: 0.0

5.3 Input Parameters Omitted from the MATLAB Interface

lda, iwork

5.4 Output Parameters

1: **a(lda,nplus2) – double array**

Contains the last simplex tableau generated by the simplex method.

2: **b(m) – double array**

The ith residual r_i corresponding to the solution vector x, for $i = 1,2,\ldots,m$.

3: **x(nplus2) – double array**

x(j) contains the jth element of the solution vector x, for $j = 1,2,\ldots,n$. The elements **x**$(n+1)$ and **x**$(n+2)$ are unused.

4: **resid – double scalar**

The sum of the absolute values of the residuals for the solution vector x.

5: **irank – int32 scalar**

The computed rank of the matrix A.

6: **iter – int32 scalar**

The number of iterations taken by the simplex method.

7: **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 $= 1$

An optimal solution has been obtained but this may not be unique.

ifail $= 2$

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

ifail $= 3$

On entry, **nplus2** < 3,
or **nplus2** $> \mathbf{m} + 2$,
or **lda** $< \mathbf{m} + 2$.

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 equations satisfied by this algorithm (i.e., those equations with zero residuals). 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 taken is approximately proportional to mn^2.

It is recommended that, before the function 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 **toler** 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

```
a = zeros(7, 5);
for i = 1:5
  a(i, 1) = exp((i-1)/5);
  a(i, 2) = exp(-(i-1)/5);
  a(i, 3) = 1;
```

```
end
b = [4.501;
     4.36;
     4.333;
     4.418;
     4.625];
[aOut, bOut, x, resid, irank, iter, ifail] = e02ga(a, b)
```

```
aOut =
    2.4750    4.1341   -1.6591    1.0014    1.0000
    3.6923    9.2006   -5.5083    2.0035    2.0000
   -6.1673  -13.3347    6.1673    1.4960    3.0000
    0.1213    0.7525    0.3688    0.0005    5.0000
    0.3688   -0.7525    0.1213    0.0023   -7.0000
   -0.5098   -1.0000   -0.5098    0.0028         0
    8.0000   -6.0000   -4.0000         0         0
bOut =
         0
    0.0005
         0
   -0.0023
         0
x =
    1.0014
    2.0035
    1.4960
         0
         0
resid =
    0.0028
irank =
           3
iter =
           5
ifail =
           0
```