

NAG Toolbox for MATLAB

e04fy

1 Purpose

e04fy is an easy-to-use algorithm for finding an unconstrained minimum of a sum of squares of m nonlinear functions in n variables ($m \geq n$). No derivatives are required.

It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

2 Syntax

```
[x, fsumsq, user, ifail] = e04fy(m, lsfun1, x, 'n', n, 'user', user)
```

3 Description

e04fy is essentially identical to the (sub)program LSNDN1 in the NPL Algorithms Library. It is applicable to problems of the form

$$\text{Minimize } F(x) = \sum_{i=1}^m [f_i(x)]^2$$

where $x = (x_1, x_2, \dots, x_n)^T$ and $m \geq n$. (The functions $f_i(x)$ are often referred to as ‘residuals’.)

You must supply a (sub)program to evaluate functions $f_i(x)$ at any point x .

From a starting point supplied by you, a sequence of points is generated which is intended to converge to a local minimum of the sum of squares. These points are generated using estimates of the curvature of $F(x)$.

4 References

Gill P E and Murray W 1978 Algorithms for the solution of the nonlinear least-squares problem *SIAM J. Numer. Anal.* **15** 977–992

5 Parameters

5.1 Compulsory Input Parameters

1: **m – int32 scalar**

the number m of residuals, $f_i(x)$, and the number n of variables, x_j .

Constraint: $1 \leq n \leq m$.

2: **lsfun1 – string containing name of m-file**

You must supply this function to calculate the vector of values $f_i(x)$ at any point x . It should be tested separately before being used in conjunction with e04fy (see the E04 Chapter Introduction).

Its specification is:

```
[fvecc, user] = lsfun1(m, n, xc, user)
```

Input Parameters1: **m** – int32 scalar2: **n** – int32 scalarThe numbers m and n of residuals and variables, respectively.3: **xc(n)** – double arrayThe point x at which the values of the f_i are required.4: **user** – Any MATLAB object**lsfun1** is called from e04fy with **user** as supplied to e04fy**Output Parameters**1: **fvecc(m)** – double array**fvecc(i)** must contain the value of f_i at the point x , for $i = 1, 2, \dots, m$.2: **user** – Any MATLAB object**lsfun1** is called from e04fy with **user** as supplied to e04fy3: **x(n)** – double array**x(j)** must be set to a guess at the j th component of the position of the minimum, for $j = 1, 2, \dots, n$.**5.2 Optional Input Parameters**1: **n** – int32 scalar*Default:* The dimension of the array **x**.the number m of residuals, $f_i(x)$, and the number n of variables, x_j .*Constraint:* $1 \leq n \leq m$.2: **user** – Any MATLAB object**user** is not used by e04fy, but is passed to **lsfun1**. Note that for large objects it may be more efficient to use a global variable which is accessible from the m-files than to use **user**.**5.3 Input Parameters Omitted from the MATLAB Interface****w**, **lw****5.4 Output Parameters**1: **x(n)** – double arrayThe lowest point found during the calculations. Thus, if **ifail** = 0 on exit, **x(j)** is the j th component of the position of the minimum.2: **fsumsq** – double scalarThe value of the sum of squares, $F(x)$, corresponding to the final point stored in **x**.3: **user** – Any MATLAB object**user** is not used by e04fy, but is passed to **lsfun1**. Note that for large objects it may be more efficient to use a global variable which is accessible from the m-files than to use **user**.

4: **ifail** – **int32 scalar**

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

6 Error Indicators and Warnings

Note: e04fy may return useful information for one or more of the following detected errors or warnings.

ifail = 1

On entry, **n** < 1,
 or **m** < **n**,
 or **lw** < $7 \times \mathbf{n} + \mathbf{n} \times \mathbf{n} + 2 \times \mathbf{m} \times \mathbf{n} + 3 \times \mathbf{m} + \mathbf{n} \times (\mathbf{n} - 1)/2$, when **n** > 1,
 or **lw** < $9 + 5 \times \mathbf{m}$, when **n** = 1.

ifail = 2

There have been $400 \times n$ calls of user-supplied (sub)program **lsfun1**, yet the algorithm does not seem to have converged. This may be due to an awkward function or to a poor starting point, so it is worth restarting e04fy from the final point held in **x**.

ifail = 3

The final point does not satisfy the conditions for acceptance as a minimum, but no lower point could be found.

ifail = 4

An auxiliary function has been unable to complete a singular value decomposition in a reasonable number of sub-iterations.

ifail = 5

ifail = 6

ifail = 7

ifail = 8

There is some doubt about whether the point x found by e04fy is a minimum of $F(x)$. The degree of confidence in the result decreases as **ifail** increases. Thus, when **ifail** = 5, it is probable that the final x gives a good estimate of the position of a minimum but, when **ifail** = 8, it is very unlikely that the function has found a minimum.

If you are not satisfied with the result (e.g., because **ifail** lies between 3 and 8), it is worth restarting the calculations from a different starting point (not the point at which the failure occurred) in order to avoid the region which caused the failure. Repeated failure may indicate some defect in the formulation of the problem.

7 Accuracy

If the problem is reasonably well scaled and a successful exit is made, then, for a computer with a mantissa of t decimals, one would expect to get about $t/2 - 1$ decimals accuracy in the components of x and between $t - 1$ (if $F(x)$ is of order 1 at the minimum) and $2t - 2$ (if $F(x)$ is close to zero at the minimum) decimals accuracy in $F(x)$.

8 Further Comments

The number of iterations required depends on the number of variables, the number of residuals and their behaviour, and the distance of the starting point from the solution. The number of multiplications performed per iteration of e04fy varies, but for $m \gg n$ is approximately $n \times m^2 + O(n^3)$. In addition, each iteration makes at least $n + 1$ calls of user-supplied (sub)program **lsfun1**. So, unless the residuals can be evaluated very quickly, the run time will be dominated by the time spent in **lsfun1**.

Ideally, the problem should be scaled so that the minimum value of the sum of squares is in the range (0,+1), and so that at points a unit distance away from the solution the sum of squares is approximately a unit value greater than at the minimum. It is unlikely that you will be able to follow these recommendations very closely, but it is worth trying (by guesswork), as sensible scaling will reduce the difficulty of the minimization problem, so that e04fy will take less computer time.

When the sum of squares represents the goodness-of-fit of a nonlinear model to observed data, elements of the variance-covariance matrix of the estimated regression coefficients can be computed by a subsequent call to e04yc, using information returned in segments of the workspace array **w**. See e04yc for further details.

9 Example

```
e04fy_lsfun1.m

function [fvecc, user] = lsfun1(m, n, xc, user)
    fvecc=zeros(m,1);

    % y is in user{1} and t is in user{2}
    for i = 1:m
        fvecc(i) = xc(1) + user{2}(i,1)/
        (xc(2)*user{2}(i,2)+xc(3)*user{2}(i,3))- ...
        user{1}(i);
    end

m = int32(15);
x = [0.5;
     1;
     1.5];
y = [0.14,0.18,0.22,0.25,0.29,0.32,0.35,0.39,0.37,0.58,0.73,0.96,
     1.34,2.10,4.39];
t = [[1.0, 15.0, 1.0],
     [2.0, 14.0, 2.0],
     [3.0, 13.0, 3.0],
     [4.0, 12.0, 4.0],
     [5.0, 11.0, 5.0],
     [6.0, 10.0, 6.0],
     [7.0, 9.0, 7.0],
     [8.0, 8.0, 8.0],
     [9.0, 7.0, 7.0],
     [10.0, 6.0, 6.0],
     [11.0, 5.0, 5.0],
     [12.0, 4.0, 4.0],
     [13.0, 3.0, 3.0],
     [14.0, 2.0, 2.0],
     [15.0, 1.0, 1.0]];

user = {y, t, 3};

[xOut, fsumsq, user, ifail] = e04fy(m, 'e04fy_lsfun1', x, 'user', user)

xOut =
    0.0824
    1.1330
    2.3437
fsumsq =
    0.0082
user =
    [1x15 double]    [15x3 double]    [3]
ifail =
    0
```

