Manual for the `linfit` function

Javier I. Carrero (jicarrerom@unal.edu.co)

October 5, 2012

1 Description

For y that depends on n independent variables $x_1, x_2, \ldots, x_n$ `linfit` calculates the values of the coefficients a_i that best fit the relation

$$y = a_0 + a_1 x_1 + a_2 x_2 + ... + a_n x_n \tag{1}$$

in the sense of minimizing the sum of square residuals calculated from a data set of m elements $(y; x_1, x_2, \ldots, x_n)$. When invoked with the `noA0` option `linfit` returns the coefficients of an alternate form of eq. 1 with $a_0 = 0$, that is

$$y = a_1 x_1 + a_2 x_2 + ... + a_n x_n. \tag{2}$$

WARNING: despite the similarity between eqs. 1 and 2 the values of a_1, a_2, ..., a_n obtained with the alternative version are different to those from the standard fitting.

2 Function arguments

The standard syntax is

```
[listCoefs, yCalc, statParams] = linfit(xVar, yVar, noA0, doGraph)
```

Arguments (function input):

- `xVar` is a `(m,n)` matrix. Each column in `xVar` contains the `m` values of their respective variable. For example `X(:,1)` contains the `m` values of the x_1 variable, i.e. `xVar(:,1) = [x11, x21, ... xm1]'`.
- `yVar` is a `(m,1)` matrix, i.e. a column vector. The $i-$th element of `yVar` is a value tied to the $i-$th column in `xVar`. Given `yVar = [y1 y2 ... ym]' y1` corresponds to `xVar(1,:)`, `y2` to `xVar(2,:)`, and so on.
- `noA0` (optional): if present produces an output with $a_0 = 0$. To invoke this option add this argument: `noA0=''Y''`.

- `doGraph` (optional) if present produces a graphic in which the `yCalc` values are plotted alongside with the perfect fit, i.e. the `y=y` line. To invoke this option add this argument: `doGraph = ''Y''`.

Results (function output):

- `listCoefs` in the default form is a `(n+1,1)` matrix containing the values of a_i, i.e. `listCoefs = [a0 a1 a2 ... an]'`. If the `noA0` option is used `listCoefs` will contain `n` values, i.e. `listCoefs = [a1 a2 ... an]'`
- `yCalc` is a column vector with `m` elements corresponding to the `y` values calculated with the `m` sets of `x1, x2, ..., xn` values.
- `statParams` is a vector with statistical parameters, `statParams = [St Sr stdv r2]` where
  - `St`: is defined as
    $$S_t = \sum_{i=1}^{m} (y_i - \bar{y})^2 \tag{3}$$
    where $\bar{y}$ is the average of y
  - `Sr`: is the sum of the m residuals, defined as
    $$S_r = \sum_{i=1}^{m} \left(y_i - y_i^{\text{calc}} \right)^2 \tag{4}$$
    where
    $$y_i^{\text{calc}} = a_0 + a_1 x_1 + a_2 x_2 + ... + a_n x_n \tag{5}$$
    (default behavior), or
    $$y_i^{\text{calc}} = a_1 x_1 + a_2 x_2 + ... + a_n x_n \tag{6}$$
    (alternative output with `noA0` option).
  - `stdv`: standard deviation, defined as
    $$\left(\frac{S_t}{m-1} \right)^{1/2} \tag{7}$$
  - `r2`: correlation coefficient, defined as
    $$r^2 = \frac{S_t - S_r}{S_t} \tag{8}$$
  - `Syx`: standard error, defined as
    $$S_{yx} = \left(\frac{S_r}{m-(n+1)} \right)^{1/2} \tag{9}$$

3 Example

Given that $y = y(u,v)$, it is proposed to find the a_0, a_1, a_2 values that best fit the relation

$$y = a_0 + a_1 u + a_2 v \tag{10}$$

from the following available data

y	u	v
27.5	2.0	18.0
28.8	3.5	16.5
28.8	4.5	10.5
29.1	2.5	2.5
30.0	8.5	9.0
31.0	10.5	4.5
32.0	13.5	1.5

First, the data is provided in order, it means that $y = 27.5$ correspond to $u = 2.0, v = 18.0$; $y = 28.8$ correspond to $u = 3.5, v = 16.5$ an so on. In this example there are 2 independent variables, u and v, and 7 elements in the data set, therefore `n=2` and `m=7`. The data can be entered this way

```
y = [27.5 28.8 28.8 29.1 30.0 31.0 32.0]
u = [2.0 3.5 4.5 2.5 8.5 10.5 13.5]
v = [18.0 16.5 10.5 2.5 9.0 4.5 1.5]
```

but it is completely necessary to supply the arguments in COLUMN form. The input argument `xVar` is composed from the independent variables `u`, `v` the, and `y` is transposed. The command

```
[A, ycalc, statpar] = linfit([u' v'], y')
```

produces

```
A = [28.560398 0.2604665 -0.0711004]'
ycalc = [27.801523 28.298873 28.985942 29.033813 30.134459
30.975344 31.970045]'
statpar = [13.82 0.4005838 1.5176737 0.9710142 0.3164585]'
```

With the alternate version

```
[A, ycalc, statpar] = linfit([u' v'], y', noA0="Y")
```

output is

```
A = [2.3959816 1.3537769]'
ycalc = [29.159947 30.723254 24.996574 9.3743961 32.549835
31.249802 34.376416]'
statpar = [13.82 422.23124 1.5176737 -29.552188 10.274133]'
```

For graphic comparison add `doGraphChk="Y"`, for example

```
linfit([u' v'], y', doGraph="Y")
```

Note: as the plot code use the new Scilab graphic structure it is necessary to use version 5.x or later.

4 Background

Let's suppose y that depends on $n+1$ functions represented as $z_0, z_1, z_2, \ldots, z_n$ in the form

$$y = a_0 z_0 + a_1 z_1 + a_2 z_2 + ... + a_n z_n \tag{11}$$

but values of the a_i coefficients are not known. Instead a total of m independent sets of values $[y, z_0, z_1, z_2, .., z_n]$ are available. The set $[a_i]_{i=0,1,2,\ldots,n}$ is obtained from the minimization of the objective function f_{obj} defined as

$$f_{\text{obj}} = \sum_{i=1}^{m} \left[y_i - (a_0 z_0 + a_1 z_1 + a_2 z_2 + ... + a_n z_n)_i \right]^2 \tag{12}$$

which correspond to the sum of the residuals, i.e. the squared differences between the known and the approximated values of y. As f_{obj} depends on $n+1$ unknown variables, i.e. $f_{\text{obj}} = f_{\text{obj}}(a_0, a_1, \ldots, a_n)$ minimization is based on

$$\frac{\partial f_{\text{obj}}}{\partial a_i} = 0 \tag{13}$$

for $i = 0, 1, 2, \ldots, n$ and produces a linear system of equations represented in the matrix form

$$\left(\mathbf{Z}^{\mathrm{T}}\mathbf{Z}\right)\mathbf{A} = \left(\mathbf{Z}^{\mathrm{T}}\mathbf{Y}\right) \tag{14}$$

where the vector $\mathbf{A}$ to be calculated represents the coefficients in eq. 11

$$\mathbf{A} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}, \tag{15}$$

$$\mathbf{Z} = \begin{bmatrix} z_{10} & z_{11} & z_{12} & \cdots & z_{1n} \\ z_{20} & z_{21} & z_{22} & \cdots & z_{2n} \\ z_{30} & z_{31} & z_{32} & \cdots & z_{3n} \\ \vdots & \vdots & \vdots & & \vdots \\ z_{m0} & z_{m1} & z_{m2} & \cdots & z_{mn} \end{bmatrix}, \tag{16}$$

and

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix}. \tag{17}$$

where z_{ij} represents the value of function z_j in the i-th set of values.

The same minimization procedure can be applied to eqs. 1 and 2. It produces a linear system with coefficients related with the terms $\sum_i x_i$, $\sum_i x_i x_j$, $\sum_i (x_i)^2$ and so on. However it results simpler to cast the problem in the form of eq. 11

- If $y = a_0 + a_1x_1 + a_2x_2 + ... + a_nx_n$ the z functions are $z_0 = 1$, $z_1 = x_1$, $z_2 = x_2$, etc.
- If $y = a_1x_1 + a_2x_2 + ... + a_nx_n$ (alternative version) the z functions are $z_0 = 0$, $z_1 = x_1$, $z_2 = x_2$, , etc.

This way both versions differ only in the construction of the Z matrix and the number of independent variables, given that the alternative version has n versus $n+1$ in the default call. For further explanation see Chapra and Canale's "Numerical Methods for Engineers", 5th ed., ch. 17 (McGraw-Hill, 2005)

Warning

This function is provided as-is, the author does not provide any guarantee about its results.