NAG Toolbox

nag_correg_lars_xtx (g02mb)

1 Purpose

nag_correg_lars_xtx (g02mb) performs Least Angle Regression (LARS), forward stagewise linear regression or Least Absolute Shrinkage and Selection Operator (LASSO) using cross-product matrices.

2 Syntax

```
[b, fitsum, ifail] = nag_correg_lars_xtx(mtype, n, dtd, dty, yty, 'pred', pred, 'intcpt', intcpt, 'm', m, 'isx', isx, 'mnstep', mnstep, 'ropt', ropt)
[b, fitsum, ifail] = g02mb(mtype, n, dtd, dty, yty, 'pred', pred, 'intcpt', intcpt, 'm', m, 'isx', isx, 'mnstep', mnstep, 'ropt', ropt)
```

3 Description

nag_correg_lars_xtx (g02mb) implements the LARS algorithm of Efron *et al.* (2004) as well as the modifications needed to perform forward stagewise linear regression and fit LASSO and positive LASSO models.

Given a vector of n observed values, $y = \{y_i : i = 1, 2, \ldots, n\}$ and an $n \times p$ design matrix X, where the jth column of X, denoted x_j, is a vector of length n representing the jth independent variable x_j, standardized such that $\sum_{i=1}^{n} x_{ij} = 0$, and $\sum_{i=1}^{n} x_{ij}^2 = 1$ and a set of model parameters β to be estimated from the observed values, the LARS algorithm can be summarised as:

1. Set $k = 1$ and all coefficients to zero, that is $\beta = 0$.
2. Find the variable most correlated with y, say x_{j_1}. Add x_{j_1} to the 'most correlated' set $\mathcal{A}$. If $p = 1$ go to 8.
3. Take the largest possible step in the direction of x_{j_1} (i.e., increase the magnitude of β_{j_1}) until some other variable, say x_{j_2}, has the same correlation with the current residual, $y - x_{j_1}\beta_{j_1}$.
4. Increment k and add x_{j_k} to $\mathcal{A}$.
5. If $|\mathcal{A}| = p$ go to 8.
6. Proceed in the 'least angle direction', that is, the direction which is equiangular between all variables in $\mathcal{A}$, altering the magnitude of the parameter estimates of those variables in $\mathcal{A}$, until the kth variable, x_{j_k}, has the same correlation with the current residual.
7. Go to 4.
8. Let $K = k$.

As well as being a model selection process in its own right, with a small number of modifications the LARS algorithm can be used to fit the LASSO model of Tibshirani (1996), a positive LASSO model, where the independent variables enter the model in their defined direction, forward stagewise linear regression (Hastie *et al.* (2001)) and forward selection (Weisberg (1985)). Details of the required modifications in each of these cases are given in Efron *et al.* (2004).

The LASSO model of Tibshirani (1996) is given by

$$\underset{\alpha,\beta_k \in \mathbb{R}^p}{\text{minimize}} \left\| y - \alpha - X^{\mathrm{T}}\beta_k \right\|^2 \quad \text{subject to} \quad \|\beta_k\|_1 \leq t_k$$

for all values of t_k, where $\alpha = \bar{y} = n^{-1}\sum_{i=1}^{n} y_i$. The positive LASSO model is the same as the standard

LASSO model, given above, with the added constraint that

$$\beta_{kj} \geq 0, \quad j = 1, 2, \ldots, p.$$

Unlike the standard LARS algorithm, when fitting either of the LASSO models, variables can be dropped as well as added to the set $\mathcal{A}$. Therefore the total number of steps K is no longer bounded by p.

Forward stagewise linear regression is an iterative procedure of the form:

1. Initialize $k = 1$ and the vector of residuals $r_0 = y - \alpha$.
2. For each $j = 1, 2, \ldots, p$ calculate $c_j = x_j^{\mathrm{T}} r_{k-1}$. The value c_j is therefore proportional to the correlation between the jth independent variable and the vector of previous residual values, r_k.
3. Calculate $j_k = \underset{j}{\operatorname{argmax}} \left|c_j\right|$, the value of j with the largest absolute value of c_j.
4. If $\left|c_{j_k}\right| < \epsilon$ then go to 7.
5. Update the residual values, with

   $$r_k = r_{k-1} + \delta \ \operatorname{sign}\left(c_{j_k}\right) x_{j_k}$$

   where δ is a small constant and $\operatorname{sign}\left(c_{j_k}\right) = -1$ when $c_{j_k} < 0$ and 1 otherwise.
6. Increment k and go to 2.
7. Set $K = k$.

If the largest possible step were to be taken, that is $\delta = \left|c_{j_k}\right|$ then forward stagewise linear regression reverts to the standard forward selection method as implemented in nag_correg_linregm_fit_onestep (g02ee).

The LARS procedure results in K models, one for each step of the fitting process. In order to aid in choosing which is the most suitable Efron *et al.* (2004) introduced a C_p-type statistic given by

$$C_p^{(k)} = \frac{\left\|y - X^{\mathrm{T}}\beta_k\right\|^2}{\sigma^2} - n + 2\nu_k,$$

where ν_k is the approximate degrees of freedom for the kth step and

$$\sigma^2 = \frac{n - y^{\mathrm{T}} y}{\nu_K}.$$

One way of choosing a model is therefore to take the one with the smallest value of $C_p^{(k)}$.

4 References

Efron B, Hastie T, Johnstone I and Tibshirani R (2004) Least Angle Regression *The Annals of Statistics (Volume 32)* **2** 407–499

Hastie T, Tibshirani R and Friedman J (2001) *The Elements of Statistical Learning: Data Mining, Inference and Prediction* Springer (New York)

Tibshirani R (1996) Regression Shrinkage and Selection via the Lasso *Journal of the Royal Statistics Society, Series B (Methodological) (Volume 58)* **1** 267–288

Weisberg S (1985) *Applied Linear Regression* Wiley

5 Parameters

5.1 Compulsory Input Parameters

1: **mtype** – INTEGER

Indicates the type of model to fit.

mtype $= 1$
LARS is performed.

mtype $= 2$
Forward linear stagewise regression is performed.

mtype $= 3$
LASSO model is fit.

mtype $= 4$
A positive LASSO model is fit.

Constraint: **mtype** $= 1$, 2, 3 or 4.

2: **n** – INTEGER

n, the number of observations.

Constraint: **n** ≥ 1.

3: **dtd** – REAL (KIND=nag_wp) array

The the array **dtd** must be a vector of length at least $\mathbf{m}(\mathbf{m}+1)/2$ or a matrix of size at least (**m**,**m**)

$D^{\mathrm{T}}D$, the cross-product matrix, which along with **isx**, defines the design matrix cross-product $X^{\mathrm{T}}X$.

If supplied in compressed format, $\mathbf{dtd}(i \times (i-1)/2 + j)$ must contain the cross-product of the ith and jth variable, for $i = 1, 2, \ldots, \mathbf{m}$ and $j = 1, 2, \ldots, \mathbf{m}$. That is the cross-product stacked by columns as returned by nag_correg_ssqmat (g02bu), for example.

Otherwise $\mathbf{dtd}(i, j)$ must contain the cross-product of the ith and jth variable, for $i = 1, 2, \ldots, \mathbf{m}$ and $j = 1, 2, \ldots, \mathbf{m}$. It should be noted that, even though $D^{\mathrm{T}}D$ is symmetric, the full matrix must be supplied.

The matrix specified in **dtd** must be a valid cross-products matrix.

4: **dty**(**m**) – REAL (KIND=nag_wp) array

$D^{\mathrm{T}}y$, the cross-product between the dependent variable, y, and the independent variables D.

5: **yty** – REAL (KIND=nag_wp)

$y^{\mathrm{T}}y$, the sums of squares of the dependent variable.

Constraint: **yty** > 0.0.

5.2 Optional Input Parameters

1: **pred** – INTEGER

Default: 2

Indicates the type of preprocessing to perform on the cross-products involving the independent variables, i.e., those supplied in **dtd** and **dty**.

pred $= 0$
No preprocessing is performed.

pred $= 2$

Each independent variable is normalized, with the jth variable scaled by $1/\sqrt{x_j^{\mathrm{T}} x_j}$. The scaling factor used by variable j is returned in $\mathbf{b}(j, nstep + 1)$.

Constraint: **pred** $= 0$ or 2.

2: **intcpt** – INTEGER

Default: 1

Indicates the type of data preprocessing that was perform on the dependent variable, y, prior to calling this function.

intcpt $= 0$
No preprocessing was performed.

intcpt $= 1$
The dependent variable, y, was mean centered.

Constraint: **intcpt** $= 0$ or 1.

3: **m** – INTEGER

Default: the second dimension of the array **dtd** and the dimension of the array **dty**. (An error is raised if these dimensions are not equal.)

m, the total number of independent variables**m** $= p$ when all variables are included.

Constraint: **m** ≥ 1.

4: **isx**$(lisx)$ – INTEGER array

Indicates which independent variables from **dtd** will be included in the design matrix, X.

If **isx** is **NULL**, all variables are included in the design matrix.

Otherwise,, for $j = 1, 2, \ldots,$ **m** when **isx**(j) must be set as follows:

isx$(j) = 1$
To indicate that the jth variable, as supplied in **dtd**, is included in the design matrix;

isx$(j) = 0$
To indicate that the jth variable, as supplied in **dtd**, is not included in the design matrix;

and $p = \sum_{j=1}^{m} \mathbf{isx}(j)$.

Constraint: **isx**$(j) = 0$ or 1 and at least one value of **isx**$(j) \neq 0$, for $j = 1, 2, \ldots,$ **m**.

5: **mnstep** – INTEGER

Default:

if **mtype** $= 1$, **m**;
otherwise $200 \times$ **m**.

The maximum number of steps to carry out in the model fitting process.

If **mtype** $= 1$, i.e., a LARS is being performed, the maximum number of steps the algorithm will take is $\min(p, n)$ if **intcpt** $= 0$, otherwise $\min(p, n - 1)$.

If **mtype** $= 2$, i.e., a forward linear stagewise regression is being performed, the maximum number of steps the algorithm will take is likely to be several orders of magnitude more and is no longer bound by p or n.

If **mtype** $= 3$ or 4, i.e., a LASSO or positive LASSO model is being fit, the maximum number of steps the algorithm will take lies somewhere between that of the LARS and forward linear stagewise regression, again it is no longer bound by p or n.

Constraint: **mnstep** ≥ 1.

6: **ropt**$(lropt)$ – REAL (KIND=nag_wp) array

Optional parameters to control various aspects of the LARS algorithm.

The default value will be used for **ropt**(i) if $lropt < i$. The default value will also be used if an invalid value is supplied for a particular argument, for example, setting **ropt**$(i) = -1$ will use the default value for argument i.

ropt(1)
The minimum step size that will be taken.

Default is $100 \times eps$ is used, where eps is the ***machine precision*** returned by nag_machine_precision (x02aj).

ropt(2)
General tolerance, used amongst other things, for comparing correlations.

Default is **ropt**(1).

ropt(3)
If set to 1 then parameter estimates are rescaled before being returned. If set to 0 then no rescaling is performed. This argument has no effect when **pred** $= 0$.

Default is for the parameter estimates to be rescaled.

Constraints:

ropt$(1) >$ ***machine precision***;
ropt$(2) >$ ***machine precision***.

5.3 Output Parameters

1: **b**$(p, :)$ – REAL (KIND=nag_wp) array

The second dimension of the array **b** will be $nstep + 1$.

Note: **nstep** is equal to K, the actual number of steps carried out in the model fitting process. See Section 3 for further information.

β the parameter estimates, with **b**$(j, k) = \beta_{kj}$, the parameter estimate for the jth variable, $j = 1, 2, \ldots, p$ at the kth step of the model fitting process, $k = 1, 2, \ldots, nstep$.

The number of parameter estimates, p, is the number of variables in **dtd** when **isx** is **NULL**, i.e., $p =$ **m**. Otherwise p is the number of nonzero values in **isx**.

By default, when **pred** $= 2$ the parameter estimates are rescaled prior to being returned. If the parameter estimates are required on the normalized scale, then this can be overridden via **ropt**.

The values held in the remaining part of **b** depend on the type of preprocessing performed.

If **pred** $= 0$ $\quad$ **b**$(j, nstep + 1) \quad = \quad 1,$

if **pred** $= 2$ $\quad$ **b**$(j, nstep + 1) \quad = \quad 1/\sqrt{x_j^{\mathrm{T}} x_j},$

for $j = 1, 2, \ldots p$.

2: **fitsum**$(\mathbf{6}, \mathbf{mnstep}+\mathbf{1})$ – REAL (KIND=nag_wp) array

The second dimension of the array **fitsum** will be $nstep+1$.

Summaries of the model fitting process. When $k = 1, 2, \ldots, nstep$

fitsum$(1, k)$
$\|\beta_k\|_1$, the sum of the absolute values of the parameter estimates for the kth step of the modelling fitting process. If **pred** $= 2$, the scaled parameter estimates are used in the summation.

fitsum$(2, k)$
RSS_k, the residual sums of squares for the kth step, where $\mathrm{RSS}_k = \|y - X^{\mathrm{T}}\beta_k\|^2$.

fitsum$(3, k)$
ν_k, approximate degrees of freedom for the kth step.

fitsum$(4, k)$
$C_p^{(k)}$, a C_p-type statistic for the kth step, where $C_p^{(k)} = \frac{\mathrm{RSS}_k}{\sigma^2} - n + 2\nu_k$.

fitsum$(5, k)$
$\hat{C}_k$, correlation between the residual at step $k-1$ and the most correlated variable not yet in the active set $\mathcal{A}$, where the residual at step 0 is y.

fitsum$(6, k)$
$\hat{\gamma}_k$, the step size used at step k.

In addition

fitsum$(1, nstep+1)$
0.

fitsum$(2, nstep+1)$
RSS_0, the residual sums of squares for the null model, where $\mathrm{RSS}_0 = y^{\mathrm{T}}y$.

fitsum$(3, nstep+1)$
ν_0, the degrees of freedom for the null model, where $\nu_0 = 0$ if **intcpt** $= 0$ and $\nu_0 = 1$ otherwise.

fitsum$(4, nstep+1)$
$C_p^{(0)}$, a C_p-type statistic for the null model, where $C_p^{(0)} = \frac{\mathrm{RSS}_0}{\sigma^2} - n + 2\nu_0$.

fitsum$(5, nstep+1)$
σ^2, where $\sigma^2 = \frac{n - \mathrm{RSS}_K}{\nu_K}$ and $K = nstep$.

Although the C_p statistics described above are returned when **ifail** $= 122$ they may not be meaningful due to the estimate σ^2 not being based on the saturated model.

3: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Note: nag_correg_lars_xtx (g02mb) may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the function:

ifail $= 11$

Constraint: **mtype** $= 1, 2, 3$ or 4.

ifail $= 21$

Constraint: **pred** $= 0$ or 2.

ifail $= 31$

Constraint: **intcpt** $= 0$ or 1.

ifail $= 41$

Constraint: **n** ≥ 1.

ifail $= 51$

Constraint: **m** ≥ 1.

ifail $= 61$

The cross-product matrix supplied in **dtd** is not symmetric.

ifail $= 62$

Constraint: diagonal elements of $D^{\mathrm{T}}D$ must be positive.

ifail $= 81$

Constraint: **isx**$(i) = 0$ or 1 for all i.

ifail $= 82$

On entry, all values of **isx** are zero.
Constraint: at least one value of **isx** must be nonzero.

ifail $= 111$

Constraint: **yty** > 0.0.

ifail $= 112$

A negative value for the residual sums of squares was obtained. Check the values of **dtd**, **dty** and **yty**.

ifail $= 121$

Constraint: **mnstep** ≥ 1.

ifail $= 122$ (*warning*)

Fitting process did not finished in **mnstep** steps. Try increasing the size of **mnstep**.
All output is returned as documented, up to step **mnstep**, however, σ and the C_p statistics may not be meaningful.

ifail $= 171$ (*warning*)

σ^2 is approximately zero and hence the C_p-type criterion cannot be calculated. All other output is returned as documented.

ifail $= 172$ (*warning*)

$\nu_K = n$, therefore sigma has been set to a large value. Output is returned as documented.

ifail $= 173$ (*warning*)

Degenerate model, no variables added and $nstep = 0$. Output is returned as documented.

ifail $= 191$

Constraint: $0 \leq lropt \leq 3$.

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.

7 Accuracy

Not applicable.

8 Further Comments

The solution path to the LARS, LASSO and stagewise regression analysis is a continuous, piecewise linear. nag_correg_lars_xtx (g02mb) returns the parameter estimates at various points along this path. nag_correg_lars_param (g02mc) can be used to obtain estimates at different points along the path.

If you have the raw data values, that is D and y, then nag_correg_lars (g02ma) can be used instead of nag_correg_lars_xtx (g02mb).

9 Example

This example performs a LARS on a simulated dataset with 20 observations and 6 independent variables.

The example uses nag_correg_ssqmat (g02bu) to get the cross-products of the augmented matrix $[D\ y]$. The first $m(m+1)/2$ elements of the (column packed) cross-products matrix returned therefore contain the elements of $D^{\mathrm{T}}D$, the next m elements contain $D^{\mathrm{T}}y$ and the last element $y^{\mathrm{T}}y$.

9.1 Program Text

```
function g02mb_example

fprintf('g02mb example results\n\n');

% Going to be fitting a LAR model via g02mb
mtype = nag_int(1);

% Augmented matrix [D y]
dy = [10.28  1.77  9.69 15.58  8.23 10.44  -46.47;
       9.08  8.99 11.53  6.57 15.89 12.58  -35.80;
      17.98 13.10  1.04 10.45 10.12 16.68 -129.22;
      14.82 13.79 12.23  7.00  8.14  7.79  -42.44;
      17.53  9.41  6.24  3.75 13.12 17.08  -73.51;
       7.78 10.38  9.83  2.58 10.13  4.25  -26.61;
      11.95 21.71  8.83 11.00 12.59 10.52  -63.90;
      14.60 10.09 -2.70  9.89 14.67  6.49  -76.73;
       3.63  9.07 12.59 14.09  9.06  8.19  -32.64;
       6.35  9.79  9.40 12.79  8.38 16.79  -83.29;
       4.66  3.55 16.82 13.83 21.39 13.88  -16.31;
       8.32 14.04 17.17  7.93  7.39 -1.09   -5.82;
      10.86 13.68  5.75 10.44 10.36 10.06  -47.75;
       4.76  4.92 17.83  2.90  7.58 11.97   18.38;
       5.05 10.41  9.89  9.04  7.90 13.12  -54.71;
       5.41  9.32  5.27 15.53  5.06 19.84  -55.62;
       9.77  2.37  9.54 20.23  9.33  8.82  -45.28;
      14.28  4.34 14.23 14.95 18.16 11.03  -22.76;
      10.17  6.80  3.17  8.57 16.07 15.93 -104.32;
       5.39  2.67  6.37 13.56 10.68  7.35  -55.94];
```

```
% Number of observations in the dataset
n = nag_int(size(dy,1));

% Calculate the means and cross-product matrix around the mean
mean_p = 'M';
[~,wmean,dtd,ifail] = g02bu( ...
                             dy,'mean_p',mean_p);

% Number of variables
m = nag_int(size(dy,2) - 1);

% The first pm elements of dtd contain the cross-products of D
% with itself, the next m elements hold the cross-product of D
% and y and the last element holds the cross-product of y with
% itself
pm = m*(m+1)/2;

% g02mb can issue warnings, but return sensible results,
% so save current warning state and turn warnings on
warn_state = nag_issue_warnings();
nag_issue_warnings(true);

[b,fitsum,ifail] = g02mb( ...
                         mtype,n,dtd(1:pm),dtd((pm+1):(pm+m)),dtd(end));

% Reset the warning state to its initial value
nag_issue_warnings(warn_state);

% Print the results
ip = size(b,1);
nstep = size(b,2) - 1;

fprintf('  Step %s Parameter Estimate\n ',repmat(' ',1,max(ip-2,0)*5));
fprintf(repmat('-',1,5+ip*10));
fprintf('\n');
for k = 1:nstep
  fprintf('  %3d',k);
  for j = 1:ip
    fprintf(' %9.3f',b(j,k));
  end
  fprintf('\n');
end
fprintf('\n');
fprintf(' alpha: %9.3f\n', wmean(m+1));
fprintf('\n');
fprintf('  Step     Sum      RSS       df       Cp       Ck     Step Size\n ');
fprintf(repmat('-',1,64));
fprintf('\n');
for k = 1:nstep
  fprintf('  %3d %9.3f %9.3f %6.0f  %9.3f %9.3f %9.3f\n', ...
          k,fitsum(1,k),fitsum(2,k),fitsum(3,k), ...
          fitsum(4,k),fitsum(5,k),fitsum(6,k));
end
fprintf('\n');
fprintf(' sigma^2: %9.3f\n', fitsum(5,nstep+1));

% Plot the parameter estimates
fig1 = figure;
ip = size(b,1);
nstep = size(b,2) - 2;

% Extract the sum of the absolute parameter estimates
xpos = transpose(repmat(fitsum(1,1:nstep),ip,1));

% Extract the parameter estimates
ypos = transpose(b(1:ip,1:nstep));

% Start both xpos and ypos at zero
xpos = [zeros(1,ip);xpos];
ypos = [zeros(1,ip);ypos];
```

```
% Get min and max for X and Y
xmin = min(min(xpos));
xmax = max(max(xpos));
ymin = min(min(ypos));
ymax = max(max(ypos));

% Get a range that is 10% past the data
ext = 1 + [-0.1 0.1];
xrng = [min(xmin*ext),max(xmax*ext)];
yrng = [min(ymin*ext),max(ymax*ext)];

% Extend the end of the lines we plot to cover this range
xpos = [xpos;xrng(2)*ones(1,ip)];
ypos = [ypos;ypos(end,:)];

% Produce the plot
plot(xpos,ypos);

% Change the axis limits
xlim(xrng);
ylim(yrng);

% Add title and labels
title({'{\bf g02mb Example Plot}'; ...
       'Estimates for LAR model fitted to simulated dataset'});
xlabel('{\bf \Sigma_j |\beta_{kj} |}');
ylabel('{\bf Parameter Estimates (\beta_{kj})}');

% Add legend
label = [repmat('\beta_{k',ip,1) num2str(transpose(linspace(1,ip,ip))) ...
         repmat('}',ip,1)];
h = legend(label,'Location','SouthOutside','Orientation','Horizontal');
set(h,'FontSize',get(h,'FontSize')*0.8);
```

9.2 Program Results

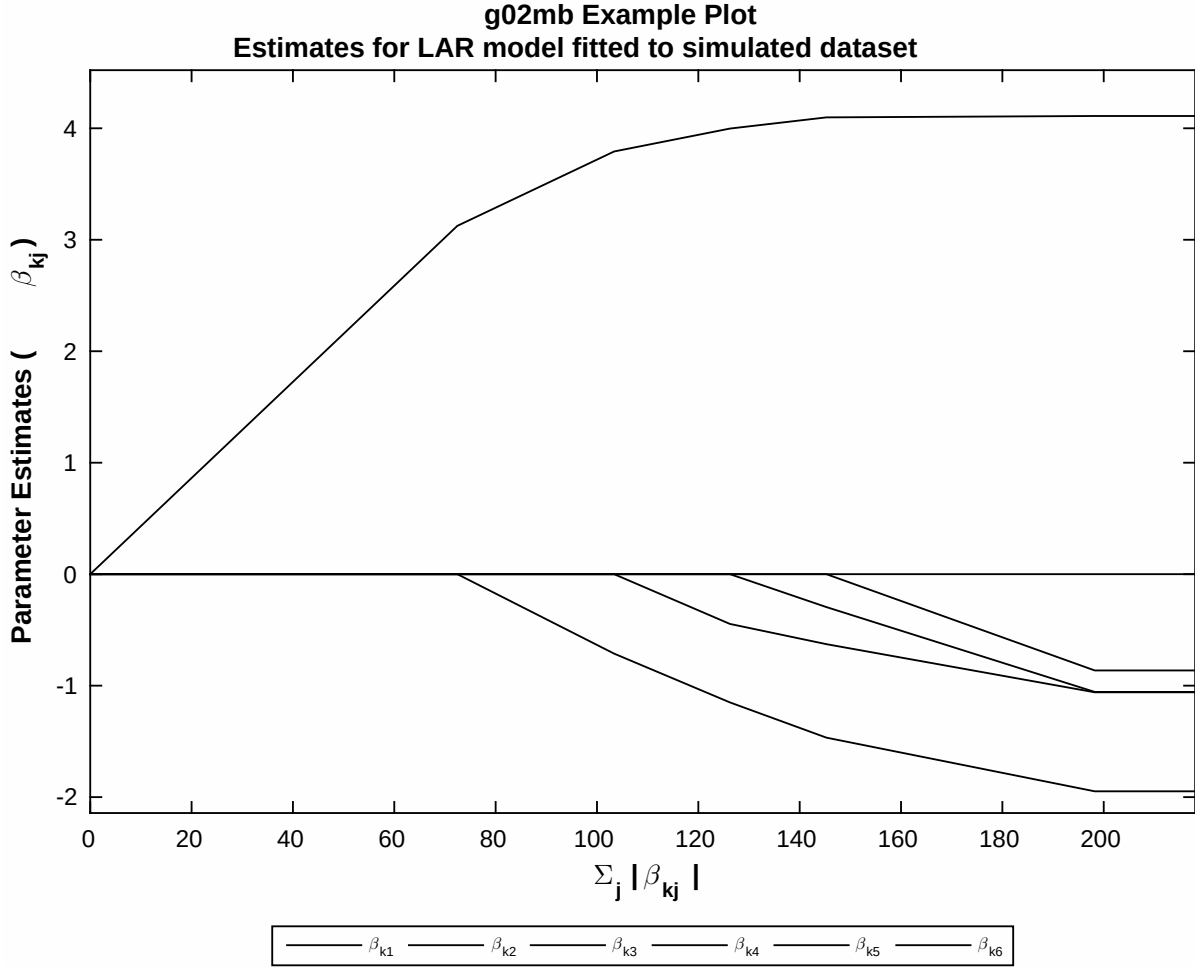
```
     g02mb example results

 Step                       Parameter Estimate
-------------------------------------------------------------------
   1     0.000     0.000     3.125     0.000     0.000     0.000
   2     0.000     0.000     3.792     0.000     0.000    -0.713
   3    -0.446     0.000     3.998     0.000     0.000    -1.151
   4    -0.628    -0.295     4.098     0.000     0.000    -1.466
   5    -1.060    -1.056     4.110    -0.864     0.000    -1.948
   6    -1.073    -1.132     4.118    -0.935    -0.059    -1.981

 alpha:   -50.037

 Step      Sum       RSS       df       Cp       Ck     Step Size
-------------------------------------------------------------------
   1    72.446  8929.855      2     13.355   123.227    72.446
   2   103.385  6404.701      3      7.054    50.781    24.841
   3   126.243  5258.247      4      5.286    30.836    16.225
   4   145.277  4657.051      5      5.309    19.319    11.587
   5   198.223  3959.401      6      5.016    12.266    24.520
   6   203.529  3954.571      7      7.000     0.910     2.198

 sigma^2:   304.198
```



This example plot shows the regression coefficients (β_k) plotted against the scaled absolute sum of the parameter estimates ($\|\beta_k\|_1$).