nag_regsn_mult_linear_newyvar (g02dgc)

1. Purpose

nag_regsn_mult_linear_newyvar (g02dgc) calculates the estimates of the parameters of a general linear regression model for a new dependent variable after a call to nag_regsn_mult_linear (g02dac).

2. Specification

```c
#include <nag.h>
#include <nagg02.h>

void nag_regsn_mult_linear_newyvar(Integer n, double wt[], double *rss,
Integer ip, Integer rank, double cov[], double q[], Integer tdq,
Boolean svd, double p[], double y[], double b[], double se[],
double res[], double com_ar[], NagError *fail)
```

3. Description

nag_regsn_mult_linear_newyvar uses the results given by nag_regsn_mult_linear (g02dac) to fit the same set of independent variables to a new dependent variable.

nag_regsn_mult_linear (g02dac) computes a QR decomposition of the matrix of \( p \) independent variables and also, if the model is not of full rank, a singular value decomposition (SVD). These results can be used to compute estimates of the parameters for a general linear model with a new dependent variable. The QR decomposition leads to the formation of an upper triangular \( p \) by \( p \) matrix \( R \) and an \( n \) by \( n \) orthogonal matrix \( Q \). In addition the vector \( c = Q^T y \) (or \( Q^T W^{1/2} y \)) is computed. For a new dependent variable, \( y_{\text{new}} \), nag_regsn_mult_linear_newyvar computes a new value of \( c = Q^T y_{\text{new}} \) or \( Q^T W^{1/2} y_{\text{new}} \).

If \( R \) is of full rank, then the least-squares parameter estimates, \( \hat{\beta} \), are the solution to: \( R \hat{\beta} = c_1 \), where \( c_1 \) is the first \( p \) elements of \( c \).

If \( R \) is not of full rank, then nag_regsn_mult_linear (g02dac) will have computed the SVD of \( R \),

\[
R = Q_s \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} P^T
\]

where \( D \) is a \( k \) by \( k \) diagonal matrix with non-zero diagonal elements, \( k \) being the rank of \( R \), and \( Q_s \) and \( P \) are \( p \) by \( p \) orthogonal matrices. This gives the solution

\[
\hat{\beta} = P_1 D^{-1} Q_s^T c_1
\]

\( P_1 \) being the first \( k \) columns of \( P \), i.e., \( P = (P_1 P_0) \) and \( Q_s \) being the first \( k \) columns of \( Q_s \). Details of the SVD are made available by nag_regsn_mult_linear (g02dac) in the form of the matrix \( P^* \):

\[
P^* = \begin{pmatrix} D^{-1} P_1^T \\ P_0^T \end{pmatrix}.
\]

The matrix \( Q_s \) is made available through the \textbf{com_ar} parameter of nag_regsn_mult_linear (g02dac).

In addition to parameter estimates, the new residuals are computed and the variance-covariance matrix of the parameter estimates are found by scaling the variance-covariance matrix for the original regression.

4. Parameters

\( n \)

Input: the number of observations, \( n \).

Constraint: \( n \geq 2 \).
nag_regrsn_mult_linear_newyvar

wt[n]
Input: if weighted estimates are required then wt must contain the weights to be used in the weighted regression. Otherwise wt need not be defined and may be set to the null pointer NULL, i.e., (double *) 0.
If wt[i] = 0.0, then the ith observation is not included in the model, in which case the effective number of observations is the number of observations with non-zero weights. The values of res and h will be set to zero for observations with zero weights.
If wt = NULL, then the effective number of observations is n.
Constraint: wt = NULL or wt[i] ≥ 0.0, for i = 0, 1, . . . , n − 1.

rss
Input: the residual sum of squares for the original dependent variable.
Output: the residual sum of squares for the new dependent variable.

ip
Input: the number p of independent variables in the model (including the mean if fitted).
Constraint: 1 ≤ ip ≤ n.

rank
Input: the rank of the independent variables, as given by nag_regrsn_mult_linear (g02dac).
Constraint: rank > 0 and if svd = FALSE, rank = ip otherwise rank ≤ ip.

cov[ip*(ip+1)/2]
Input: the covariance matrix of the parameter estimates as given by nag_regrsn_mult_linear (g02dac).
Output: the upper triangular part of the variance-covariance matrix of the ip parameter estimates given in b. They are stored packed by column, i.e., the covariance between the parameter estimate given in b[i] and the parameter estimate given in b[j], j ≥ i, is stored in cov[j(j+1)/2+i] for i = 0, 1, . . . , ip − 1 and j = i, i + 1, . . . , ip − 1.

q[n][tdq]
Input: the results of the QR decomposition as returned by nag_regrsn_mult_linear (g02dac).
Output: the first column of q contains the new values of c, the remainder of q will be unchanged.

tdq
Input: the second dimension of the array q as declared in the function from which nag_regrsn_mult_linear_newyvar is called.
Constraint: tdq ≥ ip + 1.

svd
Input: indicates if a singular value decomposition was used by nag_regrsn_mult_linear (g02dac).
If svd = TRUE, a singular value decomposition was used by nag_regrsn_mult_linear (g02dac).
If svd = FALSE, a singular value decomposition was not used by nag_regrsn_mult_linear (g02dac).

p[2*ip+ip+ip]
Input: details of the QR decomposition and SVD, if used, as returned in array p by nag_regrsn_mult_linear (g02dac).
If svd = FALSE, only the first ip elements of p are used, these will contain the zeta values for the QR decomposition (see nag_real_qr (f01qcc) for details).
If svd = TRUE, the first ip elements of p will contain the zeta values for the QR decomposition (see nag_real_qr (f01qcc) for details) and the next ip elements of p contain singular values. The following ip by ip elements contain the matrix P* stored by rows.

y[n]
Input: the new dependent variable ynew.

b[ip]
Output: b[i], i = 0, 1, . . . , ip − 1 contain the least-squares estimates of the parameters of the regression model, β.
g02 – Regression Analysis

se[\text{ip}]

Output: se[i], i = 0, 1, \ldots, \text{ip} - 1 contain the standard errors of the \text{ip} parameter estimates given in \text{b}.

res[\text{n}]

Output: the residuals for the new regression model.

com_ar[5*(\text{ip}\text{-}1)\text{+}\text{ip}]\]

Input: if \text{svd} = \text{TRUE}, \text{com_ar} must be unaltered from the previous call to \text{nag\_regsn\_mult\_linear (g02dac)}.

fail

The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

\text{NE\_INT\_ARG\_LT}

On entry, \text{ip} must not be less than 1: \text{ip} = \langle \text{value} \rangle.

\text{NE\_INT\_ARG\_LE}

On entry, \text{rank} must not be less than or equal to 0: \text{rank} = \langle \text{value} \rangle.

\text{NE\_2\_INT\_ARG\_LT}

On entry, \text{tdq} = \langle \text{value} \rangle while \text{ip} + 1 = \langle \text{value} \rangle. These parameters must satisfy \text{tdq} \geq \text{ip} + 1.

On entry, \text{n} = \langle \text{value} \rangle while \text{ip} = \langle \text{value} \rangle. These parameters must satisfy \text{n} \geq \text{ip}.

\text{NE\_REAL\_ARG\_LE}

On entry, \text{rss} must not be less than or equal to 0.0: \text{rss} = \langle \text{value} \rangle.

\text{NE\_REAL\_ARG\_LT}

On entry, \text{wt}[\langle \text{value} \rangle] must not be less than 0.0: \text{wt}[\langle \text{value} \rangle] = \langle \text{value} \rangle.

\text{NE\_SVD\_RANK\_NE\_IP}

On entry, the Boolean variable, \text{svd}, is \text{FALSE} and \text{rank} must be equal to \text{ip}: \text{rank} = \langle \text{value} \rangle, \text{ip} = \langle \text{value} \rangle.

\text{NE\_SVD\_RANK\_GT\_IP}

On entry, the Boolean variable, \text{svd}, is \text{TRUE} and \text{rank} must not be greater than \text{ip}: \text{rank} = \langle \text{value} \rangle, \text{ip} = \langle \text{value} \rangle.

6. Further Comments

The values of the leverages, \text{h}_i, are unaltered by a change in the dependent variable so a call to \text{nag\_regsn\_std\_resid\_influence (g02fac)} can be made using the value of \text{h} from \text{nag\_regsn\_mult\_linear (g02dac)}.

6.1. Accuracy

The same accuracy as \text{nag\_regsn\_mult\_linear (g02dac)} is obtained.

6.2. References


Searle S R (1971) \textit{Linear Models} Wiley.

7. See Also

\text{nag\_real\_qr (f01qcc)}
\text{nag\_regsn\_mult\_linear (g02dac)}
\text{nag\_regsn\_std\_resid\_influence (g02fac)}
8. Example

A data set consisting of 12 observations with four independent variables and two dependent variables is read in. A model with all four independent variables is fitted to the first dependent variable by \texttt{nag_regsn_mult_linear} \texttt{(g02dac)} and the results printed. The model is then fitted to the second dependent variable by \texttt{nag_regsn_mult_linear\_newyvar} and those results printed.

8.1. Program Text

```c
/* nag_regsn_mult_linear\_newyvar(g02dgc) Example Program*/
* Copyright 1990 Numerical Algorithms Group.*
* Mark 2 revised, 1992.*/
#define NMAX 12
#define MMAX 5
#define TDQ MMAX+1
#define TDXM MMAX

main()
{
  double rss, tol;
  Integer i, ip, rank, j, m, n;
  double df;
  Boolean svd;
  Nag_IncludeMean mean;
  char weight, meanc;
  double b[MMAX], cov[MMAX*(MMAX+1)/2], h[NMAX], newy[NMAX],
        p[MMAX*(MMAX+2)], q[NMAX][MMAX+1], res[NMAX], se[MMAX],
        com_ar[5*(MMAX-1)+MMAX*MMAX], wt[NMAX], xm[NMAX][MMAX], y[NMAX];
  Integer sx[MMAX];
  double *wtptr;

  Vprintf("g02dgc Example Program Results\n");
  /* Skip heading in data file */
  Vscanf("%[\n]");
  Vscanf("%ld %ld %c %c", &n, &m, &weight, &meanc);
  if (meanc=='m')
    mean = Nag_MeanInclude;
  else
    mean = Nag_MeanZero;
  if (n<=NMAX && m<MMAX)
  { 
    if (weight=='w')
    {
      wtptr = wt;
      for (i=0; i<n; i++)
      {
        for (j=0; j<m; j++)
          Vscanf("%lf", &xm[i][j]);
        Vscanf("%lf%lf%lf", &y[i], &wt[i], &newy[i]);
      }
    }
    else
    {
      wtptr = (double *)0;
      for (i=0; i<n; i++)
      {
        for (j=0; j<m; j++)
          Vscanf("%lf", &xm[i][j]);
        Vscanf("%lf%lf", &y[i], &newy[i]);
      }
    }
  }
```
Vscanf("%ld", &sx[j]);
Vscanf("%ld", &ip);
/* Set tolerance */
tol = 0.00001e0;
/* Fit initial model using g02dac */
g02dac(mean, n, (double *)xm, (Integer)TDXM, m, sx, ip, 
y, wtptr, &rss, &df, b, se, cov, res, h, (double *)q, 
(Integer)(TDQ), &svd, &rank, p, tol, com_ar, NAGERR_DEFAULT);
Vprintf("Results from g02dac\n\n");
if (svd)
    Vprintf("Model not of full rank\n\n");
Vprintf("Residual sum of squares = %12.4e\n", rss);
Vprintf("Degrees of freedom = %3.1f\n", df);
Vprintf("Variable Parameter estimate Standard error\n\n");
for (j=0; j<ip; j++)
    Vprintf("%6ld%20.4e%20.4e\n", j+1, b[j], se[j]);
Vprintf("\n");
g02dgc(n, wtptr, &rss, ip, rank, cov, (double *)q, (Integer)(TDQ), svd, p, 
newy, b, se, res, com_ar, NAGERR_DEFAULT);
Vprintf("\n");
Vprintf("Results for second y-variable using g02dgc\n\n");
Vprintf("Residual sum of squares = %12.4e\n", rss);
Vprintf("Degrees of freedom = %3.1f\n", df);
Vprintf("Variable Parameter estimate Standard error\n\n");
for (j=0; j<ip; j++)
    Vprintf("%6ld%20.4e%20.4e\n", j+1, b[j], se[j]);
Vprintf("\n");
}
else
{
    Vfprintf(stderr, "One or both of m and n are out of range:\
    m = %-3ld while n = %-3ld\n", m, n);
    exit(EXIT_FAILURE);
}
exit(EXIT_SUCCESS);

8.2. Program Data

g02dgc Example Program Data
12 4 u m
1 0 0 0 0 0 0.0 33.63 63.0
0.0 0.0 0.0 1.0 39.62 69.0
0.0 1.0 0.0 0.0 38.18 68.0
0.0 0.0 1.0 0.0 41.46 71.0
0.0 0.0 0.0 1.0 38.02 68.0
0.0 1.0 0.0 0.0 35.83 65.0
0.0 0.0 0.0 1.0 35.99 65.0
1.0 0.0 0.0 0.0 36.58 66.0
0.0 0.0 1.0 0.0 42.92 72.0
1.0 0.0 0.0 0.0 37.80 67.0
0.0 0.0 1.0 0.0 40.43 70.0
0.0 1.0 0.0 0.0 37.89 67.0
8.3. Program Results

g02dgc Example Program Results
Results from g02dac

Model not of full rank

Residual sum of squares = 2.2227e+01
Degrees of freedom = 8.0

<table>
<thead>
<tr>
<th>Variable</th>
<th>Parameter estimate</th>
<th>Standard error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.0557e+01</td>
<td>3.8494e-01</td>
</tr>
<tr>
<td>2</td>
<td>5.4467e-00</td>
<td>8.3896e-01</td>
</tr>
<tr>
<td>3</td>
<td>6.7433e+00</td>
<td>8.3896e-01</td>
</tr>
<tr>
<td>4</td>
<td>1.1047e+01</td>
<td>8.3896e-01</td>
</tr>
<tr>
<td>5</td>
<td>7.3200e+00</td>
<td>8.3896e-01</td>
</tr>
</tbody>
</table>

Results for second y-variable using g02dgc

Residual sum of squares = 2.4000e+01
Degrees of freedom = 8.0

<table>
<thead>
<tr>
<th>Variable</th>
<th>Parameter estimate</th>
<th>Standard error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.4067e+01</td>
<td>4.0000e-01</td>
</tr>
<tr>
<td>2</td>
<td>1.1267e+01</td>
<td>8.7178e-01</td>
</tr>
<tr>
<td>3</td>
<td>1.2600e+01</td>
<td>8.7178e-01</td>
</tr>
<tr>
<td>4</td>
<td>1.6933e+01</td>
<td>8.7178e-01</td>
</tr>
<tr>
<td>5</td>
<td>1.3267e+01</td>
<td>8.7178e-01</td>
</tr>
</tbody>
</table>