nag_regsn_mult_linear_delete_var (g02dfc)

1. Purpose

nag_regsn_mult_linear_delete_var (g02dfc) deletes an independent variable from a general linear regression model.

2. Specification

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

void nag_regsn_mult_linear_delete_var(Integer ip, double q[],
     Integer tdq, Integer indx, double *rss, NagError *fail)
```

3. Description

When selecting a linear regression model it is sometimes useful to drop independent variables from the model and to examine the resulting sub-model. nag_regsn_mult_linear_delete_var updates the QR decomposition used in the computation of the linear regression model. The QR decomposition may come from nag_regsn_mult_linear (g02dac), nag_regsn_mult_linear_addrem_obs (g02dcc), nag_regsn_mult_linear_add_var (g02dec) or a previous call to nag_regsn_mult_linear_delete_var.

For the general linear regression model with p independent variables fitted, nag_regsn_mult_linear (g02dac) or nag_regsn_mult_linear_add_var (g02dec) computes a QR decomposition of the (weighted) independent variables and forms an upper triangular matrix R and a vector c. To remove an independent variable R and c have to be updated. The column of R corresponding to the variable to be dropped is removed and the matrix is then restored to upper triangular form by applying a series of Givens rotations. The rotations are then applied to c. Note that only the first p elements of c are affected.

The method used means that while the updated values of R and c are computed an updated value of Q from the QR decomposition is not available so a call to nag_regsn_mult_linear_add_var (g02dec) cannot be made after a call to nag_regsn_mult_linear_delete_var.

nag_regsn_mult_linear_upd_model (g02ddc) can be used to calculate the parameter estimates, $\hat{\beta}$, from the information provided by nag_regsn_mult_linear_delete_var.

4. Parameters

ip
Input: the number of independent variables already in the model, p.
Constraint: **ip** ≥ 1.

q[ip][tdq]
Input: the results of the QR decomposition as returned by nag_regsn_mult_linear (g02dac), nag_regsn_mult_linear_addrem_obs (g02dcc), nag_regsn_mult_linear_add_var (g02dec) or previous calls to nag_regsn_mult_linear_delete_var.
Output: the updated QR decomposition. The first **ip** elements of the first column of **q** contain the updated value of c, the upper triangular part of columns 2 to **ip** contain the updated R matrix.

tdq
Input: the last dimension of the array **q** as declared in the function from which nag_regsn_mult_linear_delete_var is called.
Constraint: **tdq** $\geq$ **ip**$+1$.

indx
Input: indicates which independent variable is to be deleted from the model.
Constraint: **1** $\leq$ **indx** $\leq$ **ip**.

rss
Input: the residual sum of squares for the full regression.
Constraint: **rss** ≥ 0.0.
Output: the residual sum of squares with the (**indx**)th variable removed. Note that the residual sum of squares will only be valid if the regression is of full rank.

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

5. Error Indications and Warnings

NE_INT_ARG_LT
On entry, **ip** must not be less than 1: **ip** = ⟨*value*⟩.
On entry, **indx** must not be less than 1: **indx** = ⟨*value*⟩.

NE_2_INT_ARG_LT
On entry **tdq** = ⟨*value*⟩ while **ip**+1 = ⟨*value*⟩. These parameters must satisfy **tdq** $\geq$ **ip**+1.

NE_2_INT_ARG_GT
On entry **indx** = ⟨*value*⟩ while **ip** = ⟨*value*⟩. These parameters must satisfy **indx** $\leq$ **ip**.

NE_REAL_ARG_LT
On entry, **rss** must not be less than 0.0: **rss** = ⟨*value*⟩.

NE_DIAG_ELEM_ZERO
On entry, a diagonal element, ⟨*value*⟩, of R is zero.

NE_ALLOC_FAIL
Memory allocation failed.

6. Further Comments

6.1. Accuracy

There will inevitably be some loss in accuracy in fitting a model by dropping terms from a more complex model rather than fitting it afresh using nag_regsn_mult_linear (g02dac).

6.2. References

Golub G H and Van Loan C F (1983) *Matrix Computations* Johns Hopkins University Press, Baltimore.

Hammarling S (1985) The Singular Value Decomposition in Multivariate Statistics *ACM Signum Newsletter* **20** (3) 2–25.

7. See Also

nag_regsn_mult_linear (g02dac)
nag_regsn_mult_linear_addrem_obs (g02dcc)
nag_regsn_mult_linear_upd_model (g02ddc)
nag_regsn_mult_linear_add_var (g02dec)

8. Example

A data set consisting of 12 observations on four independent variables and one dependent variable is read in. The full model, including a mean term, is fitted using nag_regsn_mult_linear (g02dac). The value of **indx** is read in and that variable dropped from the regression. The parameter estimates are calculated by nag_regsn_mult_linear_upd_model (g02ddc) and printed. This process is repeated until **indx** is 0.

8.1. Program Text

```
/* nag_resgn_mult_linear_delete_var(g02dfc) Example Program
 *
 * Copyright 1991 Numerical Algorithms Group.
 *
 * Mark 2, 1991.
 */
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg02.h>

#define NMAX 12
#define MMAX 5
#define TDX MMAX
#define TDQ MMAX+1

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

  Vprintf("g02dfc 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 (weight=='w')
    wtptr = wt;
  else
    wtptr = (double *)0;

  if (n<=NMAX && m<MMAX)
    {
      if (wtptr)
        {
          for (i=0; i<n; ++i)
            {
              for (j=0; j<m; ++j)
                Vscanf("%lf", &x[i][j]);
              Vscanf("%lf%lf", &y[i], &wt[i]);
            }
        }
      else
        {
          for (i=0; i<n; ++i)
            {
              for (j=0; j<m; ++j)
                Vscanf("%lf", &x[i][j]);
              Vscanf("%lf", &y[i]);
            }
        }
      for (i=0; i<m; ++i)
        sx[i] = 1;
      ip = m;
      if (mean==Nag_MeanInclude)
```

```
        ip += 1;
      /*    Set tolerance  */
      tol = 0.00001e0;
      g02dac(mean, n, (double *)x, (Integer)TDX, 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 full model\n");
      if (svd)
        Vprintf("Model not of full rank\n\n");
      Vprintf("Residual sum of squares = %13.4e\n", rss);
      Vprintf("Degrees of freedom = %3.1f\n\n", df);
      while(scanf("%ld", &indx)!=EOF)
        {
          if (indx!=0)
            {
              g02dfc(ip, (double *)q, (Integer)(TDQ), indx, &rss,
                     NAGERR_DEFAULT);

              ip = ip - 1;
              if (ip==0)
                Vprintf("No terms left in model\n");
              else
                {
                  Vprintf("Variable %4ld dropped\n", indx);
                  g02ddc(n, ip, (double *)q, (Integer)(TDQ), &rss, &df, b, se,
                         cov, &svd, &rank, p, tol, NAGERR_DEFAULT);
                  Vprintf("Residual sum of squares = %13.4e\n", rss);
                  Vprintf("Degrees of freedom = %3.1f\n\n", df);
                  Vprintf("Parameter estimate   Standard error\n\n");
                  for (j=0; j<ip; j++)
                    Vprintf("%15.4e%15.4e\n", b[j], se[j]);
                }
            }
        }
    }
  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

```
g02dfc Example Program Data
 12 4 u m
1.0 1.4 0.0 0.0 4.32
1.5 2.2 0.0 0.0 5.21
2.0 4.5 0.0 0.0 6.49
2.5 6.1 0.0 0.0 7.10
3.0 7.1 0.0 0.0 7.94
3.5 7.7 0.0 0.0 8.53
4.0 8.3 1.0 4.0 8.84
4.5 8.6 1.0 4.5 9.02
5.0 8.8 1.0 5.0 9.27
5.5 9.0 1.0 5.5 9.43
6.0 9.3 1.0 6.0 9.68
6.5 9.2 1.0 6.5 9.83
2
4
0
```

8.3. Program Results

```
g02dfc Example Program Results
Results from full model
Residual sum of squares =    8.4066e-02
Degrees of freedom = 7.0

Variable    2 dropped
Residual sum of squares =    2.1239e-01
Degrees of freedom = 8.0

Parameter estimate   Standard error

     3.6372e+00     1.5083e-01
     6.1264e-01     2.8007e-02
    -6.0154e-01     4.2335e-01
     1.6709e-01     7.8656e-02
Variable    4 dropped
Residual sum of squares =    3.3220e-01
Degrees of freedom = 9.0

Parameter estimate   Standard error

     3.5974e+00     1.7647e-01
     6.2088e-01     3.2706e-02
     2.4247e-01     1.7235e-01
```