nag_regsn_mult_linear_addrem_obs (g02dcc)

1. Purpose

nag_regsn_mult_linear_addrem_obs (g02dcc) adds or deletes an observation from a general regression model fitted by nag_regsn_mult_linear (g02dac).

2. Specification

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

void nag_regsn_mult_linear_addrem_obs(Nag_UpdateObserv update, Nag_IncludeMean
     mean, Integer m, Integer sx[], double q[], Integer tdq, Integer ip, double
     x[], Integer nr, Integer tdx, Integer ix, double y, double *iot,
     double *rss, NagError *fail)
```

3. Description

nag_regsn_mult_linear (g02dac) fits a general linear regression model to a data set. The user may wish to change the model by either adding or deleting an observation from the data set. nag_regsn_mult_linear_addrem_obs takes the results from nag_regsn_mult_linear (g02dac) and makes the required changes to the vector c and the upper triangular matrix R produced by nag_regsn_mult_linear (g02dac). The regression coefficients, standard errors and the variance-covariance matrix of the regression coefficients can be obtained from nag_regsn_mult_linear_upd_model (g02ddc) after all required changes to the data set have been made.

nag_regsn_mult_linear (g02dac) performs a QR decomposition on the (weighted) X matrix of independent variables. To add a new observation to a model with p parameters the upper triangular matrix R and vector c_1, the first p elements of c, are augmented by the new observation on independent variables in x^T and dependent variable y. Givens rotations are then used to restore the upper triangular form.

$$\begin{pmatrix} R & : & c_1 \\ x & : & y \end{pmatrix} \longrightarrow \begin{pmatrix} R^* & : & c_1^* \\ 0 & : & y^* \end{pmatrix}$$

To delete an observation Givens rotations are applied to give:

$$(R \;:\; c_1) \longrightarrow \begin{pmatrix} R^* & : & c_1^* \\ x & : & y \end{pmatrix}$$

Note: only the R and upper part of the c are updated, the remainder of the Q matrix is unchanged.

4. Parameters

update
Input: indicates if an observation is to be added or deleted. If **update** = **Nag_ObservAdd**, then the observation is added. If **update** = **Nag_ObservDel**, then the observation is deleted.
Constraint: **update** = **Nag_ObservAdd** or **Nag_ObservDel**.

mean
Input: indicates if a mean has been used in the model. If **mean** = **Nag_MeanInclude**, then a mean term or intercept will have been included in the model by nag_regsn_mult_linear (g02dac). If **mean** = **Nag_MeanZero**, then a model with no mean term or intercept will have been fitted by nag_regsn_mult_linear (g02dac).
Constraint: **mean** = **Nag_MeanInclude** or **Nag_MeanZero**.

m
Input: the total number of independent variables in the data set.
Constraint: **m** ≥ 1.

sx[m]
Input: if **sx**[j] is greater than 0, then the value contained in **x**[**tdx**$*$(**ix**$-1)+j$] is to be included as a value of x^T, an observation on an independent variable, for $j = 0, 1, \ldots, m-1$.
Constraint: if **mean** = **Nag_MeanInclude**, then exactly **ip** $-$ 1 elements of **sx** must be > 0 and if **mean** = **Nag_MeanZero**, then exactly **ip** elements of **sx** must be > 0.

q[ip][tdq]
Input: **q** must be array **q** as output by nag_regsn_mult_linear (g02dac), nag_regsn_mult_linear_add_var (g02dec), nag_regsn_mult_linear_delete_var (g02dfc), or a previous call to nag_regsn_mult_linear_addrem_obs.
Output: the first **ip** elements of the first column of **q** will contain c_1^*, the upper triangular part of columns 2 to **ip** + 1 will contain R^*, the remainder is unchanged.

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

ip
Input: the number of linear terms in general linear regression model (including mean if there is one).
Constraint: **ip** ≥ 1.

x[nr$*$tdx]
Input: the **ip** values for the dependent variables of the observation to be added or deleted, x^T. The positions of the values **x** extracted depends on **ix** and **tdx**.

nr
Input: the number of rows of the notional two dimensional array **x**.
Constraint: **nr** ≥ 1.

tdx
Input: the trailing dimension of the notional two dimensional array **x**.
Constraint: **tdx** $\geq m$.

ix
Input: the row of the notional two dimensional array **x** that contains the values for the dependent variables of the observation to be added or deleted.
Constraint: $1 \leq$ **ix** $\leq nr$.

y
Input: the value of the dependent variable for the observation to be added or deleted, y.

wt
Input: if the new observation is to be weighted, then **wt** must contain the weight to be used with the new observation. If **wt** = 0.0, then the observation is not included in the model. If the new observation is to be unweighted, then a null pointer, `(double *)0`, must be passed.
Constraint: if the new observation is to be weighted **wt** $\geq$ 0.0.

rss
Input: the value of the residual sums of squares for the original set of observations.
Constraint: **rss** $\geq$ 0.0.
Output: the updated values of the residual sums of squares.

Note: this will only be valid if the model 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, **m** must not be less than 1: **m** = ⟨*value*⟩.
On entry, **ix** must not be less than 1: **ix** = ⟨*value*⟩.
On entry, **nr** must not be less than 1: **nr** = ⟨*value*⟩.

NE_2_INT_ARG_LT
On entry **tdq** = ⟨*value*⟩ while **ip** + 1 = ⟨*value*⟩. These parameters must satisfy **tdq** ≥ **ip** + 1.
On entry **tdx** = ⟨*value*⟩ while **m** = ⟨*value*⟩. These parameters must satisfy **tdx** ≥ **m**.

NE_2_INT_ARG_GT
On entry **ix** = ⟨*value*⟩ while **nr** = ⟨*value*⟩. These parameters must satisfy **ix** ≤ **nr**.

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

NE_BAD_PARAM
On entry, **update** had an illegal value.
On entry, **mean** had an illegal value.

NE_IP_INCOMP_WITH_SX
On entry, for **mean** = **Nag_MeanInclude**, number of non-zero
values of **sx** must be equal to **ip** − 1: number of non-zero values of **sx** = ⟨*value*⟩, **ip** − 1 = ⟨*value*⟩.
On entry, for **mean** = **Nag_MeanZero**, number of non-zero
values of **sx** must be equal to **ip**: number of non-zero values of **sx** = ⟨*value*⟩, **ip** = ⟨*value*⟩.

NE_RSS_NOT_UPD
The **rss** could not be updated because the input **rss** was less than the calculated decrease in **rss** when the new observation was deleted.

NE_MAT_NOT_UPD
The R matrix could not be updated: to, either,
delete non-existent observation, or, add an observation to R matrix with
zero diagonal element.

NE_ALLOC_FAIL
Memory allocation failed.

6. Further Comments

Care should be taken with the use of this function.

(a) It is possible to delete observations which were not included in the original model.

(b) If several additions/deletions have been performed the user is advised to recompute the regression using nag_regsn_mult_linear (g02dac).

(c) Adding or deleting observations can alter the rank of the model. Such changes will only be detected when a call to nag_regsn_mult_linear_upd_model (g02ddc) has been made. nag_regsn_mult_linear_upd_model (g02ddc) should also be used to compute the new residual sum of squares when the model is not of full rank.

nag_regsn_mult_linear_addrem_obs may also be used after nag_regsn_mult_linear_add_var (g02dec) and nag_regsn_mult_linear_delete_var (g02dfc).

6.1. Accuracy

Higher accuracy is achieved by updating the R matrix rather than the traditional methods of updating X'X.

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_upd_model (g02ddc)
nag_regsn_mult_linear_add_var (g02dec)
nag_regsn_mult_linear_delete_var (g02dfc)

8. Example

A data set consisting of 12 observations with four independent variables is read in and a general linear regression model fitted by nag_regsn_mult_linear (g02dac) and parameter estimates printed. The last observation is then dropped and the parameter estimates recalculated, using nag_regsn_mult_linear_upd_model (g02ddc), and printed.

8.1. Program Text

```
/* nag_regsn_mult_linear_addrem_obs(g02dcc) 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, ip, rank, j, m, n;
  double df;
  Boolean  svd;
  char meanc, weight;
  Nag_IncludeMean  mean;
  Nag_UpdateObserv update;
  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], xm[NMAX][MMAX], y[NMAX];
  double   *wtptr;
  Integer  sx[MMAX];

  Vprintf("g02dcc 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", &xm[i][j]);
              Vscanf("%lf%lf", &y[i], &wt[i]);
            }
        }
      else
        {
          for (i=0; i<n; i++)
            {
              for (j=0; j<m; j++)
```

```
                Vscanf("%lf", &xm[i][j]);
              Vscanf("%lf", &y[i]);
            }
        }
      for (j=0; j<m; ++j)
        Vscanf("%ld", &sx[j]);
      Vscanf("%ld", &ip);
      /* Set tolerance */
      tol = 0.00001e0;

      /* Fit initial model using g02dac */
      g02dac(mean, n, (double *)xm, (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 g02dac\n\n");
      if (svd)
        Vprintf("Model not of full rank\n");
      Vprintf("Residual sum of squares = %12.4e\n", rss);
      Vprintf("Degrees of freedom = %3.1f\n\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");
      update = Nag_ObservDel;
      g02dcc(update, mean, m, sx, (double *)q, (Integer)(TDQ), ip,
             (double *)xm,  (Integer)NMAX, (Integer)MMAX, (Integer)12,
             y[11], wtptr, &rss, NAGERR_DEFAULT);
      Vprintf("Results from dropping an observation using g02dcc\n");
      n = n - 1;
      g02ddc(n, ip, (double *)q, (Integer)(TDQ), &rss, &df, b, se, cov,
             &svd, &rank, p, tol, NAGERR_DEFAULT);

      Vprintf("Residual sum of squares = %12.4e\n", rss);
      Vprintf("Degrees of freedom = %3.1f\n\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]);
    }
  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

```
g02dcc Example Program Data
 12 4 u z
1.0 0.0 0.0 0.0 33.63
0.0 0.0 0.0 1.0 39.62
0.0 1.0 0.0 0.0 38.18
0.0 0.0 1.0 0.0 41.46
0.0 0.0 0.0 1.0 38.02
0.0 1.0 0.0 0.0 35.83
0.0 0.0 0.0 1.0 35.99
1.0 0.0 0.0 0.0 36.58
0.0 0.0 1.0 0.0 42.92
1.0 0.0 0.0 0.0 37.80
0.0 0.0 1.0 0.0 40.43
1.0 1.0 1.0 1.0 37.89
 1   1   1   1   4
```

8.3. Program Results

```
g02dcc Example Program Results
Results from g02dac

Residual sum of squares =   5.2748e+03
Degrees of freedom = 8.0

Variable    Parameter estimate   Standard error

     1          2.0724e+01          1.3801e+01
     2          1.4085e+01          1.6240e+01
     3          2.6324e+01          1.3801e+01
     4          2.2597e+01          1.3801e+01

Results from dropping an observation using g02dcc
Residual sum of squares =   2.1705e+01
Degrees of freedom = 7.0

Variable    Parameter estimate   Standard error

     1          3.6003e+01          1.0166e+00
     2          3.7005e+01          1.2451e+00
     3          4.1603e+01          1.0166e+00
     4          3.7877e+01          1.0166e+00
```