

nag_regsn_mult_linear_est_func (g02dnc)

1. Purpose

nag_regsn_mult_linear_est_func (g02dnc) gives the estimate of an estimable function along with its standard error.

2. Specification

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

void nag_regsn_mult_linear_est_func(Integer ip, Integer rank, double b[],
    double cov[], double p[], double f[], Boolean *est, double *stat,
    double *sestat, double *t, double tol, NagError *fail)
```

3. Description

This function computes the estimates of an estimable function for a general linear regression model which is not of full rank. It is intended for use after a call to `nag_regsn_mult_linear (g02dac)` or `nag_regsn_mult_linear_upd_model (g02ddc)`. An estimable function is a linear combination of the parameters such that it has a unique estimate. For a full rank model all linear combinations of parameters are estimable.

In the case of a model not of full rank the functions use a singular value decomposition (SVD) to find the parameter estimates, $\hat{\beta}$, and their variance-covariance matrix. Given the upper triangular matrix R obtained from the QR decomposition of the independent variables the SVD gives:

$$R = Q_* \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} P^T$$

where D is a k by k diagonal matrix with non-zero diagonal elements, k being the rank of R , and Q_* and P are p by p orthogonal matrices. This leads to a solution:

$$\hat{\beta} = P_1 D^{-1} Q_{*1}^T c_1$$

P_1 being the first k columns of P , i.e., $P = (P_1 P_0)$, Q_{*1} being the first k columns of Q_* and c_1 being the first p elements of c .

Details of the SVD are made available, in the form of the matrix P^* :

$$P^* = \begin{pmatrix} D^{-1} P^T \\ P_0^{-1} \end{pmatrix}$$

as given by `nag_regsn_mult_linear (g02dac)` and `nag_regsn_mult_linear_upd_model (g02ddc)`.

A linear function of the parameters, $F = f^T \beta$, can be tested to see if it is estimable by computing $\zeta = P_0^T f$. If ζ is zero, then the function is estimable, if not, the function is not estimable. In practice $|\zeta|$ is tested against some small quantity η .

Given that F is estimable it can be estimated by $f^T \hat{\beta}$ and its standard error calculated from the variance-covariance matrix of $\hat{\beta}$, C_β , as

$$\text{se}(F) = \sqrt{f^T C_\beta f}$$

Also a t -statistic:

$$t = \frac{f^T \hat{\beta}}{\text{se}(F)},$$

can be computed. The t -statistic will have a Student's t -distribution with degrees of freedom as given by the degrees of freedom for the residual sum of squares for the model.

4. Parameters

ip

Input: the number of terms in the linear model, p .
Constraint: $\mathbf{ip} \geq 1$.

rank

Input: the rank of the independent variables, k .
Constraint: $1 \leq \mathbf{rank} \leq \mathbf{ip}$.

b[ip]

Input: the \mathbf{ip} values of the estimates of the parameters of the model, $\hat{\beta}$.

cov[ip*(ip+1)/2]

Input: the upper triangular part of the variance-covariance matrix of the \mathbf{ip} parameter estimates given in \mathbf{b} . They are stored packed by column, i.e., the covariance between the parameter estimate given in $\mathbf{b}[i]$ and the parameter estimate given in $\mathbf{b}[j]$, $j \geq i$, is stored in $\mathbf{cov}[j(j+1)/2+i]$, for $i = 0, 1, \dots, \mathbf{ip} - 1$ and $j = i, i+1, \dots, \mathbf{ip} - 1$.

p[ip*ip+2*ip]

Input: \mathbf{p} as returned by nag_regsn_mult_linear (g02dac) or nag_regsn_mult_linear_upd_model (g02ddc).

f[ip]

Input: the linear function to be estimated, f .

est

Output: **est** indicates if the function was estimable.
If **est** = **TRUE**, then the function is estimable.
If **est** = **FALSE**, the function is not estimable and **stat**, **sestat** and **t** are not set.

stat

Output: if **est** = **TRUE**, **stat** contains the estimate of the function, $f^T \hat{\beta}$.

sestat

Output: if **est** = **TRUE**, **sestat** contains the standard error of the estimate of the function, $se(F)$.

t

Output: if **est** = **TRUE**, **t** contains the t -statistic for the test of the function being equal to zero.

tol

Input: **tol** is the tolerance value used in the check for estimability, η .
If **tol** \leq 0.0, then $\sqrt{\mathbf{machine\ precision}}$ is used instead.

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** = $\langle value \rangle$.
On entry, **rank** must not be less than 1: **rank** = $\langle value \rangle$.

NE_2_INT_ARG_GT

On entry **ip** = $\langle value \rangle$ while **rank** = $\langle value \rangle$. These parameters must satisfy $\mathbf{rank} \leq \mathbf{ip}$.

NE_RANK_EQ_IP

On entry, **rank** = **ip**. In this case, the boolean variable **est** is returned as **TRUE** and all statistics are calculated.

NE_STDES_ZERO

$se(F) = 0.0$ probably due to rounding error or due to incorrectly specified inputs **cov** and **f**.

NE_ALLOC_FAIL

Memory allocation failed.

6. Further Comments

The value of estimable functions is independent of the solution chosen from the many possible solutions. While `nag_regsn_mult_linear_est_func` may be used to estimate functions of the parameters of the model as computed by `nag_regsn_mult_linear_tran_model` (g02dkc), β_c , these must be expressed in terms of the original parameters, β . The relation between the two sets of parameters may not be straightforward.

6.1. Accuracy

The computations are believed to be stable.

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.

Searle S R (1971) *Linear Models* Wiley.

7. See Also

`nag_regsn_mult_linear` (g02dac)
`nag_regsn_mult_linear_upd_model` (g02ddc)
`nag_regsn_mult_linear_tran_model` (g02dkc)

8. Example

Data from an experiment with four treatments and three observations per treatment are read in. A model, with a mean term, is fitted by `nag_regsn_mult_linear` (g02dac). The number of functions to be tested is read in, then the linear functions themselves are read in and tested with `nag_regsn_mult_linear_est_func`. The results of `nag_regsn_mult_linear_est_func` are printed.

8.1. Program Text

```
/* nag_regsn_mult_linear_est_func(g02dnc) 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 TDQ MMAX+1
#define TDX MMAX

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

    Vprintf("g02dnc 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", &x[i][j]);
            Vscanf("%lf%lf", &y[i], &wt[i]);
        }
    }
    else
    {
        wtptr = (double *)0;
        for (i=0; i<n; i++)
        {
            for (j=0; j<m; j++)
                Vscanf("%lf", &x[i][j]);
            Vscanf("%lf", &y[i]);
        }
    }
    for (j=0; j<m; j++)
        Vscanf("%ld", &sx[j]);
    Vscanf("%ld", &ip);
    /* Set tolerance */
    tol = 0.00001e0;

    /*
     * Find initial estimates using g02dac
     */
    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("\\n");
    Vprintf("Estimates from g02dac\\n\\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");

    Vscanf("%ld", &nestern);
    for (i=1; i<=nestern; ++i)
    {
        for (j=0; j<ip; ++j)
            Vscanf("%lf", &f[j]);

        g02dnc(ip, rank, b, cov, p, f, &est, &stat, &sestat, &t, tol,
              &fail);

        if (fail.code==NE_NOERROR || fail.code==NE_RANK_EQ_IP)
        {
            Vprintf("\\n");
            Vprintf("Function %ld\\n\\n", i);
            for (j=0; j<ip; ++j)
                Vprintf("%8.2f%c", f[j], (j%5==4 || j==ip-1) ? '\\n' : ' ');
            Vprintf("\\n");
            if (est)
                Vprintf(" stat = %10.4f se = %10.4f t = %10.4f\\n",
                        stat, sestat, t);
            else
                Vprintf("Function not estimable\\n");
        }
    }
}

```

```

        }
        else
            Vprintf("%s\n", fail.message);
    }
}
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

g02dnc Example Program Data

```

12 4 u m
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
0.0 1.0 0.0 0.0 37.89
1 1 1 1 5
3
1.0 1.0 0.0 0.0 0.0
0.0 1.0 -1.0 0.0 0.0
0.0 1.0 0.0 0.0 0.0

```

8.3. Program Results

g02dnc Example Program Results

Estimates from g02dac

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

Variable	Parameter estimate	Standard error
1	3.0557e+01	3.8494e-01
2	5.4467e+00	8.3896e-01
3	6.7433e+00	8.3896e-01
4	1.1047e+01	8.3896e-01
5	7.3200e+00	8.3896e-01

Function 1

```

1.00 1.00 0.00 0.00 0.00

```

```

stat = 36.0033 se = 0.9623 t = 37.4119

```

Function 2

```

0.00 1.00 -1.00 0.00 0.00

```

```

stat = -1.2967 se = 1.3610 t = -0.9528

```

Function 3

```

0.00 1.00 0.00 0.00 0.00

```

Function not estimable