nag_glm_est_func (g02gnc)

1. Purpose

nag_glm_est_func (g02gnc) gives the estimate of an estimable function along with its standard error from the results from fitting a generalized linear model.

2. Specification

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

void nag_glm_est_func(Integer ip, Integer rank, double b[],
         double cov[], double v[], Integer tdv, double f[],
         Boolean *est, double *stat, double *sestat, double *z,
         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_glm_normal (g02gac), nag_glm_binomial (g02gbc), nag_glm_poisson (g02gcc) or nag_glm_gamma (g02gdc). 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^*:

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

as given by nag_glm_normal (g02gac), nag_glm_binomial (g02gbc), nag_glm_poisson (g02gcc) and nag_glm_gamma (g02gdc).

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 z statistic:

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

can be computed. The distribution of z will be approximately Normal.

4. Parameters

ip
Input: the number of terms in the linear model, p.
Constraint: **ip** ≥ 1.

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

b[ip]
Input: the **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 **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**$[j]$, $j \geq i$, is stored in **cov**$[j(j+1)/2+i]$, for $i = 0, 1, \ldots,$ **ip** $- 1$ and $j = i, i+1, \ldots,$ **ip** $- 1$.

v[ip][tdv]
Input: **v** as returned by nag_glm_normal (g02gac), nag_glm_binomial (g02gbc), nag_glm_poisson (g02gcc) and nag_glm_gamma (g02gdc)

tdv
Input: the second dimension of the array **v** as declared in the function from which nag_glm_est_func is called.
Constraint: **tdv** $\geq$ **ip**+6. **tdv** should be as supplied to nag_glm_normal (g02gac), nag_glm_binomial (g02gbc), nag_glm_poisson (g02gcc) or nag_glm_gamma (g02gdc).

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 **z** 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).

z
Output: if **est** = **TRUE**, **z** contains the z 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** ≤ 0.0, then $\sqrt{\textbf{\textit{machine precision}}}$ is used instead.

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

For this function the values of output parameters may be useful even if **fail.code** $\neq$ **NE_NOERROR** on exit. Users are therefore advised to supply the **fail** parameter and set **fail.print** = `TRUE`.

5. Error Indications and Warnings

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

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

NE_2_INT_ARG_LT
On entry, **tdv** = ⟨*value*⟩ while **ip** = ⟨*value*⟩. These parameters must satisfy **tdv** ≥ **ip**+6.

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
sestat, the standard error of the estimate of the function, se(F) = 0.0; probably due to rounding error or due to incorrectly specified input values of **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_glm_est_func may be used to estimate functions of the parameters of the model as computed by nag_glm_tran_model (g02gkc), 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.
McCullagh P and Nelder J A (1983) *Generalized Linear Models* Chapman and Hall.
Searle S R (1971) *Linear Models* Wiley.

7. See Also

nag_glm_normal (g02gac)
nag_glm_binomial (g02gbc)
nag_glm_poisson (g02gcc)
nag_glm_gamma (g02gdc)
nag_glm_tran_model (g02gkc)

8. Example

A loglinear model is fitted to a 3 by 5 contingency table by nag_glm_poisson (g02gcc). The model consists of terms for for rows and columns. The table is:

141	67	114	79	39
131	66	143	72	35
36	14	38	28	16

The number of functions to be tested is read in, then the linear functions themselves are read in and tested with nag_glm_est_func. The results of nag_glm_est_func are printed.

8.1. Program Text

```
/* nag_glm_est_func(g02gnc) Example Program.
 *
 * Copyright 1996 Numerical Algorithms Group.
 *
 * Mark 4, 1996.
 *
 */
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg02.h>

#define NMAX 15
#define MMAX  9
#define TDC MMAX
#define TDX MMAX
#define TDV MMAX+6

main()
{
  Boolean est;
  Integer i, ip, j, m, n, nestfn, tdx, tdv;
  double  ex_power;
  double  sestat, stat, z;
  Integer sx[MMAX];
  double  b[MMAX], f[MMAX], v[NMAX][TDV], wt[NMAX], x[NMAX][MMAX],
  y[NMAX];
  double  *wtptr;
  Integer max_iter;
  Integer print_iter;
  double eps;
  double tol;
  double df;
  double dev;
  Integer rank;
  double cov[MMAX*(MMAX+1)/2], se[MMAX];
  static  NagError fail;

  Vprintf("g02gnc Example Program Results\n");
  /*  Skip heading in data file */
  Vscanf("%*[^\n]");
  Vscanf("%ld %ld %ld", &n, &m, &print_iter);

  if (n<=NMAX && m<MMAX)
    {
      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 control parameters */
      max_iter = 10;
      tol = 5e-5;
      eps = 1e-6;
      ex_power = 0.0;

      /* Fit Log-linear model using g02gcc */

      g02gcc(Nag_Log, Nag_MeanInclude, n, (double *)x, (Integer)TDX,
             m, sx, ip, y, wtptr, (double *)0, ex_power, &dev, &df, b, &rank, se, cov,
             (double *)v, (Integer)TDV, tol, max_iter, print_iter, "", eps, &fail);
```

```
      if (fail.code == NE_NOERROR || fail.code == NE_LSQ_ITER_NOT_CONV ||
          fail.code == NE_RANK_CHANGED || fail.code == NE_ZERO_DOF_ERROR)
        {
          Vprintf("\nDeviance = %12.4e\n", dev);
          Vprintf("Degrees of freedom = %3.1f\n\n", df);
          Vprintf("       Estimate     Standard error\n\n");
          for (i=0; i<ip; i++)
            Vprintf("%14.4f%14.4f\n", b[i], se[i]);
          Vprintf("\n");
          Vscanf("%ld", &nestfn);
          for (i=1; i<=nestfn; ++i)
            {
              for (j=0; j<ip; ++j)
                Vscanf("%lf", &f[j]);

              g02gnc(ip, rank, b, cov, (double *)v,
                     (Integer)TDV, f, &est, &stat, &sestat, &z, 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  sestat = %10.4f   z = %10.4f\n",
                            stat, sestat, z);
                  else
                    Vprintf("Function not estimable\n");
                }
              else
                Vprintf("%s\n", fail.message);
            }
        }
      else
        {
          Vprintf("%s\n", fail.message);
          exit(EXIT_FAILURE);
        }
    }
  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

```
g02gnc Example Program Data
15 8 0
1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 141.
1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0  67.
1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 114.
1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0  79.
1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0  39.
0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 131.
0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0  66.
0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 143.
0.0 1.0 0.0 0.0 0.0 0.0 1.0 0.0  72.
0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0  35.
0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0  36.
0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0  14.
0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0  38.
0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0  28.
0.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0  16.
1   1   1   1   1   1   1   1   9
3
1.0  1.0  0.0 0.0  1.0 0.0 0.0 0.0 0.0
0.0  1.0 -1.0 0.0  0.0 0.0 0.0 0.0 0.0
0.0  1.0  0.0 0.0  0.0 0.0 0.0 0.0 0.0
```

8.3. Program Results

```
g02gnc Example Program Results

Deviance =   9.0379e+00
Degrees of freedom = 8.0

       Estimate     Standard error

        2.5977         0.0258
        1.2619         0.0438
        1.2777         0.0436
        0.0580         0.0668
        1.0307         0.0551
        0.2910         0.0732
        0.9876         0.0559
        0.4880         0.0675
       -0.1996         0.0904


Function 1

    1.00     1.00     0.00     0.00     1.00
    0.00     0.00     0.00     0.00

stat =     4.8903  sestat =     0.0674   z =    72.5934

Function 2

    0.00     1.00    -1.00     0.00     0.00
    0.00     0.00     0.00     0.00

stat =    -0.0158  sestat =     0.0672   z =    -0.2350

Function 3

    0.00     1.00     0.00     0.00     0.00
    0.00     0.00     0.00     0.00

Function not estimable
```