g02 – Regression Analysis

nag_glm_tran_model (g02gkc)

1. Purpose

nag_glm_tran_model (g02gkc) calculates the estimates of the parameters of a generalized linear model for given constraints from the singular value decomposition results.

2. Specification

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

void nag_glm_tran_model(Integer ip, Integer nclin, double v[],
                        Integer tdv, double c[], Integer tdc, double b[], double scale,
                        double se[], double cov[], NagError *fail)
```

3. Description

This function computes the estimates given a set of linear constraints for a generalized linear 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).

In the case of a model not of full rank the functions use a singular value decomposition (SVD) to find the parameter estimates $\hat{\beta}_{\text{svd}}$, and their variance-covariance matrix. Details of the SVD are made available, in the form of the matrix $P^*$:

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

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

Alternative solutions can be formed by imposing constraints on the parameters. If there are $p$ parameters and the rank of the model is $k$, then $n_c = p - k$ constraints will have to be imposed to obtain a unique solution.

Let $C$ be a $p$ by $n_c$ matrix of constraints, such that

$$C^T \hat{\beta} = 0,$$

then the new parameter estimates $\hat{\beta}_c$ are given by:

$$\hat{\beta}_c = A\hat{\beta}_{\text{svd}} = (I - P_0(C^T P_0)^{-1})\hat{\beta}_{\text{svd}},$$

where $I$ is the identity matrix, and the variance-covariance matrix is given by:

$$AP_1 D^{-2}P_0^T A^T$$

provided $(C^T P_0)^{-1}$ exists.

4. Parameters

- **ip**
  - Input: the number of terms in the linear model, $p$.
  - Constraint: $ip \geq 1$.

- **nclin**
  - Input: the number of constraints to be imposed on the parameters, $n_c$.
  - Constraint: $0 < nclin < ip$.

- **v[ip][tdv]**
  - Input: $v$ as returned by nag_glm_normal (g02gac), nag_glm_binomial (g02gbc), nag_glm_poisson (g02gcc) or nag_glm_gamma (g02gdc).
nag_glm_tran_model

**tdv**
Input: the second dimension of the array \( v \) as declared in the function from which nag_glm_tran_model 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).

c[ip][tdc]
Input: the \( nclin \) constraints stored by column, i.e., the \( i \)th constraint is stored in the \( i \)th column of \( c \).

**tdc**
Input: the last dimension of the array \( c \) as declared in the function from which nag_glm_tran_model is called.
Constraint: \( tdc \geq nclin \).

**b[ip]**
Input: the parameter estimates computed by using the singular value decomposition, \( \hat{\beta}_{svd} \).
Output: the parameter estimates of the parameters with the constraints imposed, \( \hat{\beta}_c \).

**scale**
Input: the estimate of the scale parameter.
For results from nag_glm_normal (g02gac) and nag_glm_gamma (g02gdc) then scale is the scale parameter, for the model \( \sigma^2 \) and \( \nu^{-1} \) respectively.
For results from nag_glm_binomial (g02gbc) and nag_glm_poisson (g02gcc) then scale should be set to 1.0.
Constraint: \( scale > 0.0 \).

**se[ip]**
Output: the standard error of the parameter estimates in \( b \).

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

**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: \( ip = \langle value \rangle \).

#### NE_INT_ARG_LE
On entry, \( nclin \) must not be less than or equal to 0: \( nclin = \langle value \rangle \).

#### NE_2_INT_ARG_GE
On entry, \( nclin = \langle value \rangle \) while \( ip = \langle value \rangle \). These parameters must satisfy \( nclin < ip \).

#### NE_2_INT_ARG_LT
On entry, \( tdv = \langle value \rangle \) while \( ip = \langle value \rangle \). These parameters must satisfy \( tdv \geq ip+6 \).
On entry, \( tdc = \langle value \rangle \) while \( nclin = \langle value \rangle \). These parameters must satisfy \( tdc \geq nclin \).

#### NE_REAL_ARG_LE
On entry, \( scale = \langle value \rangle \).

#### NE_MAT_NOT_FULL_RANK
Matrix \( c \) does not give a model of full rank.

#### NE_ALLOC_FAIL
Memory allocation failed.
6. Further Comments

This function is intended for use in situations in which dummy (0-1) variables have been used such as in the analysis of designed experiments when the user does not wish to change the parameters of the model to give a full rank model. The function is not intended for situations in which the relationships between the independent variables are only approximate.

6.1. Accuracy

It should be noted that due to rounding errors a parameter that should be zero when the constraints have been imposed may be returned as a value of order machine precision.

6.2. References


7. See Also

nag_glm_normal (g02gac)
nag_glm_binomial (g02gbc)
nag_glm_poisson (g02gcc)
nag_glm_gamma (g02gdc)
nag_glm_est_func (g02gnc)

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 rows and columns. The table is:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>141</td>
<td>67</td>
<td>114</td>
<td>79</td>
<td>39</td>
</tr>
<tr>
<td>131</td>
<td>66</td>
<td>143</td>
<td>72</td>
<td>35</td>
</tr>
<tr>
<td>36</td>
<td>14</td>
<td>38</td>
<td>28</td>
<td>16</td>
</tr>
</tbody>
</table>

The constraints that the sum of row effects and the sum of column effects are zero are then read in and the parameter estimates with these constraints imposed are computed by nag_glm_tran_model and printed.

8.1. Program Text

/* nag_glm_tran_model(g02gkc) 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()
{
    Integer i, nclin, ip, j, m, n;
double ex_power;
    Integer sx[MMAX];
double b[MMAX], c[NMAX][MMAX], cov[MMAX*(MMAX+1)/2], se[MMAX],
v[NMAX][TDV], x[NMAX][MMAX], y[NMAX];
double *wtptr;
Integer max_iter;
Integer print_iter;
double eps;
double tol;
double df;
double dev;
Integer rank;
static NagError fail;

Vprintf("g02gkc Example Program Results\n\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);
        /* Input constraints */
        nclin = ip - rank;
        for (i=0; i<ip; ++i)
        { for (j=0; j<nclin; ++j)
            { Vscanf("%lf", &c[i][j]);
            }
        }
g02gkc(ip, nclin, (double *)v, (Integer)TDV, (double *)c, (Integer)TDC, b, 1.0e0, se, cov, &fail);
        if (fail.code == NE_NOERROR) {
            Vprintf(" Estimate Standard error\n");
            for (i=0; i<ip; i++)
            { Vprintf(" %14.4f%14.4f\n", b[i], se[i]);
                Vprintf("\n");
            }else {
                Vprintf("%s\n",fail.message);
                exit(EXIT_FAILURE);
            }
        }
    }
}
else {
    Vprintf("%s\n",fail.message);
}
8.2. Program Data

g02gkc Example Program Data

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>8</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8.3. Program Results

g02gkc Example Program Results

Deviance   =  9.0379e+00  
Degrees of freedom =  8.0  

| 3.9831 | 0.0396 | 0.3961 | 0.0458 | 0.4118 | 0.0457 | -0.8079 | 0.0622 | 0.5112 | 0.0562 | -0.2285 | 0.0727 | 0.4680 | 0.0569 | -0.0316 | 0.0675 | -0.7191 | 0.0887 |