nag_mv_discrim_group (g03dcc)

1 Purpose

nag_mv_discrim_group (g03dcc) allocates observations to groups according to selected rules. It is intended for use after nag_mv_discrim (g03dac).

2 Specification

```c
#include <nag.h>
#include <nagg03.h>

void nag_mv_discrim_group (Nag_DiscrimMethod type, Nag_GroupCovars equal,
                         Nag_PriorProbability priors, Integer nvar, Integer ng,
                         const Integer nig[], const double gmean[], Integer tdg,
                         const double gc[], const double det[], Integer noba, Integer m,
                         const Integer isx[], const double x[], Integer tdx, double prior[],
                         double p[], Integer tdp, Integer iag[], Nag_Boolean atiq, double ati[],
                         NagError *fail)
```

3 Description

Discriminant analysis is concerned with the allocation of observations to groups using information from other observations whose group membership is known, \(X_t\); these are called the training set. Consider \(p\) variables observed on \(ng\) populations or groups. Let \(x_j\) be the sample mean and \(S_j\) the within-group variance-covariance matrix for the \(j\)th group; these are calculated from a training set of \(n\) observations with \(n_j\) observations in the \(j\)th group, and let \(x_k\) be the \(k\)th observation from the set of observations to be allocated to the \(ng\) groups. The observation can be allocated to a group according to a selected rule.

The allocation rule or discriminant function will be based on the distance of the observation from an estimate of the location of the groups, usually the group means. A measure of the distance of the observation from the \(j\)th group mean is given by the Mahalanobis distance,

\[
D^2_{kj} := \left( x_k - \bar{x}_j \right)^T S_j^{-1} \left( x_k - \bar{x}_j \right).
\]  

(1)

If the pooled estimate of the variance-covariance matrix \(S\) is used rather than the within-group variance-covariance matrices, then the distance is:

\[
D^2_{kj} := \left( x_k - \bar{x}_j \right)^T S^{-1} \left( x_k - \bar{x}_j \right).
\]  

(2)

Instead of using the variance-covariance matrices \(S\) and \(S_j\), nag_mv_discrim_group (g03dcc) uses the upper triangular matrices \(R\) and \(R_j\) supplied by nag_mv_discrim (g03dac) such that \(S = R^T R\) and \(S_j = R_j^T R_j\). \(D^2_{kj}\) can then be calculated as \(z^T z\) where \(R_j z = \left( x_k - \bar{x}_j \right)\) or \(R z = \left( x_k - \bar{x}_j \right)\) as appropriate.

In addition to the distances, a set of prior probabilities of group membership, \(\pi_j\), for \(j = 1, 2, \ldots, ng\), may be used, with \(\sum \pi_j = 1\). The prior probabilities reflect your view as to the likelihood of the observations coming from the different groups. Two common cases for prior probabilities are \(\pi_1 = \pi_2 = \cdots = \pi_{ng}\), that is, equal prior probabilities, and \(\pi_j = n_j/n\) for \(j = 1, 2, \ldots, ng\), that is, prior probabilities proportional to the number of observations in the groups in the training set.

nag_mv_discrim_group (g03dcc) uses one of four allocation rules. In all four rules the \(p\) variables are assumed to follow a multivariate Normal distribution with mean \(\mu_j\) and variance-covariance matrix \(\Sigma_j\) if the observation comes from the \(j\)th group. The different rules depend on whether or not the within-group variance-covariance matrices are assumed equal, i.e., \(\Sigma_1 = \Sigma_2 = \cdots = \Sigma_{ng}\), and whether a predictive or estimative approach is used. If \(p(x_k | \mu_j, \Sigma_j)\) is the probability of observing the observation \(x_k\) from group \(j\), then the posterior probability of belonging to group \(j\) is:
\[ p(j \mid x_k, \mu_j, \Sigma_j) \propto p(x_k \mid \mu_j, \Sigma_j) \pi_j. \]  

(3)

In the estimative approach, the arguments \( \mu_j \) and \( \Sigma_j \) in (3) are replaced by their estimates calculated from \( X_t \). In the predictive approach, a non-informative prior distribution is used for the arguments and a posterior distribution for the arguments, \( p(\mu_j, \Sigma_j \mid X_t) \), is found. A predictive distribution is then obtained by integrating \( p(j \mid x_k, \mu_j, \Sigma_j)p(\mu_j, \Sigma_j \mid X) \) over the argument space. This predictive distribution then replaces \( p(x_k \mid \mu_j, \Sigma_j) \) in (3). See Aitchison and Dunsmore (1975), Aitchison et al. (1977) and Moran and Murphy (1979) for further details.

The observation is allocated to the group with the highest posterior probability. Denoting the posterior probabilities, \( p(j \mid x_k, \mu_j, \Sigma_j) \), by \( q_j \), the four allocation rules are:

(i) Estimative with equal variance-covariance matrices – Linear Discrimination.

\[
\log(q_j) \propto -\frac{1}{2} D_{kj}^2 + \log \pi_j
\]


\[
\log(q_j) \propto -\frac{1}{2} D_{kj}^2 + \log \pi_j - \frac{1}{2} \log |S_j|
\]

(iii) Predictive with equal variance-covariance matrices.

\[
q_j^{-1} \propto \left( \frac{(n_j + 1)/n_j}{1 + [(n_j/n_j)(n_j + 1)]D_{kj}^2} \right)^{(n+1-n_j)/2}
\]

(iv) Predictive with unequal variance-covariance matrices

\[
q_j^{-1} \propto C \left\{ \left( \frac{n_j^2 - 1}{n_j} \right) |S_j| \right\}^{p/2} \left\{ 1 + \left( \frac{n_j}{(n_j^2 - 1)} \right) D_{kj}^2 \right\}^{n_j/2}
\]

where

\[
C = \frac{\Gamma\left(\frac{1}{2}(n_j - p)\right)}{\Gamma\left(\frac{1}{2}n_j\right)}
\]

In the above the appropriate value of \( D_{kj}^2 \) from (1) or (2) is used. The values of the \( q_j \) are standardized so that,

\[
\sum_{j=1}^{n_j} q_j = 1.
\]

Moran and Murphy (1979) show the similarity between the predictive methods and methods based upon likelihood ratio tests.

In addition to allocating the observation to a group, nag_mv_discrim_group (g03dcc) computes an atypicality index, \( I_j(x_k) \). This represents the probability of obtaining an observation more typical of group \( j \) than the observed \( x_k \) (see Aitchison and Dunsmore (1975) and Aitchison et al. (1977)). The atypicality index is computed as:

\[
I_j(x_k) = P\left( B \leq z : \frac{1}{2}p, \frac{1}{2}(n_j - d) \right)
\]

where \( P(B \leq \beta : a, b) \) is the lower tail probability from a beta distribution where, for unequal within-group variance-covariance matrices,

\[
z = D_{kj}^2 / \left( D_{kj}^2 + \left( n_j^2 - 1 \right)/n_j \right),
\]

and for equal within-group variance-covariance matrices,

\[
z = D_{kj}^2 / \left( D_{kj}^2 + (n - n_j)(n_j - 1)/n_j \right).
If $I_j(x_k)$ is close to 1 for all groups it indicates that the observation may come from a grouping not represented in the training set. Moran and Murphy (1979) provide a frequentist interpretation of $I_j(x_k)$.

4 References


5 Arguments

1: \(\text{type} \rightarrow \text{Nag\_DiscrimMethod}\)

*Input*

*On entry:* indicates whether the estimative or predictive approach is to be used.

- \(\text{type} = \text{Nag\_DiscrimEstimate}\)
  - The estimative approach is used.

- \(\text{type} = \text{Nag\_DiscrimPredict}\)
  - The predictive approach is used.

*Constraint:* \(\text{type} = \text{Nag\_DiscrimEstimate}\) or \(\text{Nag\_DiscrimPredict}\).

2: \(\text{equal} \rightarrow \text{Nag\_GroupCovars}\)

*Input*

*On entry:* indicates whether or not the within-group variance-covariance matrices are assumed to be equal and the pooled variance-covariance matrix used.

- \(\text{equal} = \text{Nag\_EqualCovar}\)
  - The within-group variance-covariance matrices are assumed equal and the matrix $R$ stored in the first $p(p+1)/2$ elements of gc is used.

- \(\text{equal} = \text{Nag\_NotEqualCovar}\)
  - The within-group variance-covariance matrices are assumed to be unequal and the matrices $R_i$, for $i = 1, 2, \ldots, n_g$, stored in the remainder of gc are used.

*Constraint:* \(\text{equal} = \text{Nag\_EqualCovar}\) or \(\text{Nag\_NotEqualCovar}\).

3: \(\text{priors} \rightarrow \text{Nag\_PriorProbability}\)

*Input*

*On entry:* indicates the form of the prior probabilities to be used.

- \(\text{priors} = \text{Nag\_EqualPrior}\)
  - Equal prior probabilities are used.

- \(\text{priors} = \text{Nag\_GroupSizePrior}\)
  - Prior probabilities proportional to the group sizes in the training set, $n_j$, are used.

- \(\text{priors} = \text{Nag\_UserPrior}\)
  - The prior probabilities are input in prior.

*Constraint:* \(\text{priors} = \text{Nag\_EqualPrior}\), \(\text{Nag\_GroupSizePrior}\) or \(\text{Nag\_UserPrior}\).
4: **nvar** – Integer
   
   *Input*
   
   *On entry:* the number of variables, \( p \), in the variance-covariance matrices as specified to `nag_mv_discrim (g03dac)`.
   
   *Constraint:* \( nvar \geq 1 \).

5: **ng** – Integer
   
   *Input*
   
   *On entry:* the number of groups, \( n_g \).
   
   *Constraint:* \( ng \geq 2 \).

6: **nig[ng]** – const Integer
   
   *Input*
   
   *On entry:* the number of observations in each group training set, \( n_j \).
   
   *Constraints:*
   
   - if `equal` = Nag_EQUALCOVAR, \( \sum_{j=1}^{ng} nig[j-1] > ng + nvar \), for \( j = 1, 2, \ldots, ng \);
   - if `equal` = Nag_NOTEQUALCOVAR, \( nig[j-1] > nvar \), for \( j = 1, 2, \ldots, ng \).

7: **gmean[ng \times tdg]** – const double
   
   *Input*
   
   *Note:* the \((i, j)\)th element of the matrix is stored in `gmean[(i-1) \times tdg + j - 1]`.
   
   *On entry:* the \( j \)th row of `gmean` contains the means of the \( p \) variables for the \( j \)th group, for \( j = 1, 2, \ldots, n_j \). These are returned by `nag_mv_discrim (g03dac)`.

8: **tdg** – Integer
   
   *Input*
   
   *On entry:* the stride separating matrix column elements in the array `gmean`.
   
   *Constraint:* \( tdg \geq nvar \).

9: **gc[dim]** – const double
   
   *Input*
   
   *Note:* the dimension, \( dim \), of the array `gc` must be at least \((ng + 1) \times nvar \times (nvar + 1)/2\).
   
   *On entry:* the first \( p(p+1)/2 \) elements of `gc` should contain the upper triangular matrix \( R \) and the next \( n_g \) blocks of \( p(p+1)/2 \) elements should contain the upper triangular matrices \( R_j \).
   
   All matrices must be stored packed by column. These matrices are returned by `nag_mv_discrim (g03dac)`. If `equal` = Nag_EQUALCOVAR, only the first \( p(p+1)/2 \) elements are referenced, if `equal` = Nag_NOTEQUALCOVAR, only the elements \( p(p+1)/2 \) to \( (ng + 1)p(p+1)/2 - 1 \) are referenced.
   
   *Constraints:*
   
   - if `equal` = Nag_EQUALCOVAR, the diagonal elements of \( R \) must be \( \neq 0 \);
   - if `equal` = Nag_NOTEQUALCOVAR, the diagonal elements of the \( R_j \) must be \( \neq 0 \), for \( j = 1, 2, \ldots, n_g \).

10: **det[ng]** – const double
    
    *Input*
    
    *On entry:* if `equal` = Nag_NOTEQUALCOVAR, the logarithms of the determinants of the within-group variance-covariance matrices as returned by `nag_mv_discrim (g03dac)`. Otherwise `det` is not referenced.

11: **nobs** – Integer
    
    *Input*
    
    *On entry:* the number of observations in \( x \) which are to be allocated.
    
    *Constraint:* \( nobs \geq 1 \).
12: \( m \) – Integer

*Input*

*On entry:* the number of variables in the data array \( x \).

*Constraint:* \( m \geq nvar \).

13: \( \text{isx}[m] \) – const Integer

*Input*

*On entry:* \( \text{isx}[l - 1] \) indicates if the \( l \)th variable in \( x \) is to be included in the distance calculations. If \( \text{isx}[l - 1] > 0 \) the \( l \)th variable is included, for \( l = 1, 2, \ldots, m \); otherwise the \( l \)th variable is not referenced.

*Constraint:* \( \text{isx}[l - 1] > 0 \) for \( nvar \) values of \( l \).

14: \( x[nobs \times tdx] \) – const double

*Input*

*On entry:* \( x[(k - 1) \times tdx + l - 1] \) must contain the \( k \)th observation for the \( l \)th variable, for \( k = 1, 2, \ldots, nobs \) and \( l = 1, 2, \ldots, m \).

15: \( tdx \) – Integer

*Input*

*On entry:* the stride separating matrix column elements in the array \( x \).

*Constraint:* \( tdx \geq m \).

16: \( \text{priors}[ng] \) – double

*Input/Output*

*On entry:* if \( \text{priors} = \text{Nag UserPrior} \) the prior probabilities for the \( ng \) groups.

*Constraint:* if \( \text{priors} = \text{Nag UserPrior} \), \( \text{priors}[j - 1] > 0.0 \) and

\[
1 - \sum_{j=1}^{n_g} \text{priors}[j - 1] \leq 10 \times \text{machine precision}, \text{ for } j = 1, 2, \ldots, n_g.
\]

*On exit:* if \( \text{priors} = \text{Nag GroupSizePrior} \), the computed prior probabilities in proportion to group sizes for the \( ng \) groups.

If \( \text{priors} = \text{Nag UserPrior} \), the input prior probabilities will be unchanged.

If \( \text{priors} = \text{Nag EqualPrior} \), \( \text{priors} \) is not set.

17: \( p[nobs \times tdp] \) – double

*Output*

*On exit:* \( p[(k - 1) \times tdp + j - 1] \) contains the posterior probability \( p_{kj} \) for allocating the \( k \)th observation to the \( j \)th group, for \( k = 1, 2, \ldots, nobs \) and \( j = 1, 2, \ldots, n_g \).

18: \( tdp \) – Integer

*Input*

*On entry:* the stride separating matrix column elements in the arrays \( p \), \( ati \).

*Constraint:* \( tdp \geq ng \).

19: \( iag[nobs] \) – Integer

*Output*

*On exit:* the groups to which the observations have been allocated.

20: \( atiq \) – Nag Boolean

*Input*

*On entry:* \( atiq \) must be Nag_TRUE if atypicality indices are required. If \( atiq \) is Nag_FALSE, the array \( ati \) is not set.

21: \( ati[nobs \times tdp] \) – double

*Output*

*On exit:* if \( atiq \) is Nag_TRUE, \( ati[(k - 1) \times tdp + j - 1] \) will contain the atypicality index for the \( k \)th observation with respect to the \( j \)th group, for \( k = 1, 2, \ldots, nobs \) and \( j = 1, 2, \ldots, n_g \). If \( atiq \) is Nag_FALSE, \( ati \) is not set.
6 Error Indicators and Warnings

**NE_2_INT_ARG_LT**

On entry, \( m = \langle \text{value} \rangle \) while \( \text{nvar} = \langle \text{value} \rangle \). These arguments must satisfy \( m \geq n\text{var} \).

On entry, \( \text{tdg} = \langle \text{value} \rangle \) while \( \text{nvar} = \langle \text{value} \rangle \). These arguments must satisfy \( \text{tdg} \geq \text{nvar} \).

On entry, \( \text{tdp} = \langle \text{value} \rangle \) while \( \text{ng} = \langle \text{value} \rangle \). These arguments must satisfy \( \text{tdp} \geq \text{ng} \).

On entry, \( \text{tdx} = \langle \text{value} \rangle \) while \( m = \langle \text{value} \rangle \). These arguments must satisfy \( \text{tdx} \geq m \).

**NE_ALLOC_FAIL**

Dynamic memory allocation failed.

**NE_BAD_PARAM**

On entry, argument \( \text{equal} \) had an illegal value.

On entry, argument \( \text{priors} \) had an illegal value.

On entry, argument \( \text{type} \) had an illegal value.

**NE_DIAG_0_COND**

A diagonal element of \( R \) is zero when \( \text{equal} = \text{Nag}_{\text{EqualCovar}} \).

**NE_DIAG_0_J_COND**

A diagonal element of \( R \) is zero for some \( j \), when \( \text{equal} = \text{Nag}_{\text{NotEqualCovar}} \).

**NE_GROUP_SUM**

On entry, the \( \sum_{j=1}^{\text{ng}} \text{nig}[j-1] = \langle \text{value} \rangle \), \( \text{ng} = \langle \text{value} \rangle \), \( \text{nvar} = \langle \text{value} \rangle \).

Constraint: \( \sum_{j=1}^{\text{ng}} \text{nig}[j-1] > \text{ng} + \text{nvar} \) when \( \text{equal} = \text{Nag}_{\text{EqualCovar}} \).

**NE_INT_ARG_LT**

On entry, \( \text{ng} = \langle \text{value} \rangle \).

Constraint: \( \text{ng} \geq 2 \).

On entry, \( \text{nobs} = \langle \text{value} \rangle \).

Constraint: \( \text{nobs} \geq 1 \).

On entry, \( \text{nvar} = \langle \text{value} \rangle \).

Constraint: \( \text{nvar} \geq 1 \).

**NE_INTARR**

On entry, \( \text{nig}[\langle \text{value} \rangle] = \langle \text{value} \rangle \).

Constraint: \( \text{nig}[i-1] > 0 \), for \( i = 1, 2, \ldots, \text{ng} \), when \( \text{equal} = \text{Nag}_{\text{EqualCovar}} \).

**NE_INTARR_INT**

On entry, \( \text{nig}[\langle \text{value} \rangle] = \langle \text{value} \rangle \), \( \text{nvar} = \langle \text{value} \rangle \).

Constraint: \( \text{nig}[i-1] > \text{nvar} \), \( i = 1, 2, \ldots, \text{ng} \) when \( \text{equal} = \text{Nag}_{\text{NotEqualCovar}} \).

**NE_INTERNAL_ERROR**

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

The NAG error argument (see Section 3.6 in the Essential Introduction).
**NE_PRIOR_SUM**

On entry, \( \sum_{j=1}^{ng} \text{prior}[j-1] = \langle \text{value} \rangle \).

Constraint: \( \sum_{j=1}^{ng} \text{prior}[j-1] \) must be within 10 times \( \text{machine precision} \) of 1 when \( \text{priors} = \text{Nag_UserPrior} \).

**NE_REALARR**

On entry, \( \text{prior}[(\text{value})] = \langle \text{value} \rangle \).

Constraint: \( \text{prior}[j-1] > 0, j = 1, 2, \ldots, \text{ng} \) when \( \text{priors} = \text{Nag_UserPrior} \).

**NE_VAR_INCL_INDICATED**

The number of variables, \( \text{nvar} \) in the analysis = \( \langle \text{value} \rangle \), while number of variables included in the analysis via array \( \text{isx} = \langle \text{value} \rangle \).

Constraint: these two numbers must be the same.

7 **Accuracy**

The accuracy of the returned posterior probabilities will depend on the accuracy of the input \( R \) or \( R_j \) matrices. The atypicality index should be accurate to four significant places.

8 **Parallelism and Performance**

Not applicable.

9 **Further Comments**

The distances \( D_{kj}^2 \) can be computed using \( \text{nag_mv_discrim_mahaldist} \) (g03dbc) if other forms of discrimination are required.

10 **Example**

The data, taken from Aitchison and Dunsmore (1975), is concerned with the diagnosis of three ‘types’ of Cushing’s syndrome. The variables are the logarithms of the urinary excretion rates (mg/24hr) of two steroid metabolites. Observations for a total of 21 patients are input and the group means and \( R \) matrices are computed by \( \text{nag_mv_discrim} \) (g03dac). A further six observations of unknown type are input and allocations made using the predictive approach and under the assumption that the within-group covariance matrices are not equal. The posterior probabilities of group membership, \( q_j \), and the atypicality index are printed along with the allocated group. The atypicality index shows that observations 5 and 6 do not seem to be typical of the three types present in the initial 21 observations.

10.1 **Program Text**

/* nag_mv_discrim_group (g03dcc) Example Program. *
 * * Copyright 2014 Numerical Algorithms Group. *
 * * Mark 5, 1998. *
 * * Mark 8 revised, 2004. *
 * */
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg03.h>
#define ATI(I, J) ati[(I) *tdati + J]
#define GMEAN(I, J) gmean[(I) *tdgmean + J]
#define P(I, J) p[(I) *tdp + J]
#define X(I, J) x[(I) *tdx + J]
int main(void)
{
    Integer exit_status = 0, i, *iag = 0, *ing = 0, *isx = 0, j, m, n,
    ng, *nig = 0, nobs;
    Integer nvar, tdati, tdgmean, tdp, tdx;
    double *ati = 0, *det = 0, df, *gc = 0, *gmean = 0, *p = 0;
    double *prior = 0, sig, stat, *wt = 0, *wtptr = 0, *x = 0;
    char nag_enum_arg[40];
    Nag_Boolean atiq = Nag_TRUE, weight;
    Nag_DiscrimMethod type;
    Nag_GroupCovars equal;
    NagError fail;

    INIT_FAIL(fail);

    printf("nag_mv_discrim_group (g03dcc) Example Program Results\n\n");
    /* Skip headings in data file */
    #ifdef _WIN32
        scanf_s("%*[\n"]);
    #else
        scanf("%*[\n"]);
    #endif
    #ifdef _WIN32
        scanf_s("%"NAG_IFMT", &n);
    #else
        scanf("%"NAG_IFMT", &n);
    #endif
    #ifdef _WIN32
        scanf_s("%"NAG_IFMT", &m);
    #else
        scanf("%"NAG_IFMT", &m);
    #endif
    #ifdef _WIN32
        scanf_s("%"NAG_IFMT", &nvar);
    #else
        scanf("%"NAG_IFMT", &nvar);
    #endif
    #ifdef _WIN32
        scanf_s("%39s", nag_enum_arg, _countof(nag_enum_arg));
    #else
        scanf("%39s", nag_enum_arg);
    #endif
    /* nag_enum_name_to_value (x04nac).
     * Converts NAG enum member name to value
     */
    weight = (Nag_Boolean) nag_enum_name_to_value(nag_enum_arg);
    if (n >= 1 && nvar >= 1 && m >= nvar && ng >= 2)
    {
        if (!det || gc || gmean || prior || wt || x || ing || nig)
        {
            printf("Allocation failure\n");
            exit_status = -1;
            goto END;
        }
    }
}

NAG Library Manual

```g03dcc
```
tdati = ng;
tdmean = nvar;
tdp = ng;
tdx = m;
}
else
{
    printf("Invalid n or nvar or ng.\n\n");
    exit_status = 1;
    return exit_status;
}
if (weight)
{
    for (i = 0; i < n; ++i)
    {
        for (j = 0; j < m; ++j)
            ifdef _WIN32
                scanf_s("%lf", &X(i, j));
            else
                scanf("%lf", &X(i, j));
            endif
        #ifdef _WIN32
            scanf_s("%NAG_IFMT"", &ing[i]);
        #else
            scanf("%NAG_IFMT"", &ing[i]);
        #endif
        #ifdef _WIN32
            scanf_s("%lf", &wt[i]);
        #else
            scanf("%lf", &wt[i]);
        #endif
    }
    wtptr = wt;
}
else
{
    for (i = 0; i < n; ++i)
    {
        for (j = 0; j < m; ++j)
            #ifdef _WIN32
                scanf_s("%lf", &X(i, j));
            #else
                scanf("%lf", &X(i, j));
            #endif
        #ifdef _WIN32
            scanf_s("%NAG_IFMT"", &ing[i]);
        #else
            scanf("%NAG_IFMT"", &ing[i]);
        #endif
    }
    for (j = 0; j < m; ++j)
        #ifdef _WIN32
            scanf_s("%NAG_IFMT"", &isx[j]);
        #else
            scanf("%NAG_IFMT"", &isx[j]);
        #endif
        /* nag_mv_discrim (g03dac).
         * Test for equality of within-group covariance matrices
         */
        nag_mv_discrim(n, m, x, tdx, isx, nvar, ing, ng, wtptr, nig,
            gmean, tdmean, det, gc, &stat, &df, &sig, &fail);
        if (fail.code != NE_NOERROR)
        {
            printf("Error from nag_mv_discrim (g03dac).\n\n", fail.message);
            exit_status = 1;
            goto END;
        }
    #ifdef _WIN32
        scanf_s("%NAG_IFMT"", &nobs);
#else
scand("%NAG_IFMT", &nobs);
#endif
#endif
#endif
#if defined(_WIN32)
scanf_s("%39s", nag_enum_arg, _countof(nag_enum_arg));
#else
scanf("%39s", nag_enum_arg);
#endif
equal = (Nag_GroupCovars) nag_enum_name_to_value(nag_enum_arg);
#endif
#endif
type = (Nag_DiscrimMethod) nag_enum_name_to_value(nag_enum_arg);
if (nobs >= 1)
{
    if (!(ati = NAG_ALLOC((nobs)*(ng), double)) ||
        !(p = NAG_ALLOC((nobs)*(ng), double)) ||
        !(iag = NAG_ALLOC(nobs, Integer)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }

tdati = ng;
tdp = ng;
for (i = 0; i < nobs; ++i)
{
    for (j = 0; j < m; ++j)
    {
#if defined(_WIN32)
        scanf_s("%lf", &X(i, j));
#else
        scanf("%lf", &X(i, j));
#endif
    }
}
/* nag_mv_discrim_group (g03dcd).
 * Allocates observations to groups, following
 * nag_mv_discrim (g03dac)
 */
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_mv_discrim_group (g03dcd).\n%s
", fail.message);
    exit_status = 1;
    goto END;
}
printf("\n");
printf(" Obs Posterior Allocated ");
printf(" Atypicality ");
printf("\n");
printf(" probabilities to group index ");
printf("\n");
printf("\n");
for (i = 0; i < nobs; ++i)
{
    for (j = 0; j < ng; ++j)
    {
        printf("%6.3f", P(i, j));
    }
    printf("%6"NAG_IFMT"", iag[i]);
    for (j = 0; j < ng; ++j)
    {

END:
NAG_FREE(ati);
NAG_FREE(det);
NAG_FREE(gc);
NAG_FREE(gmean);
NAG_FREE(p);
NAG_FREE(prior);
NAG_FREE(iag);
NAG_FREE(x);
NAG_FREE(wt);
NAG_FREE(isx);
NAG_FREE(nig);
return exit_status;
}

10.2 Program Data

nag_mv_discrim_group (g03dcc) Example Program Data
21 2 2 3 Nag_FALSE
1.1314 2.4596 1
1.0986 0.2624 1
0.6419 -2.3026 1
1.3350 -3.2189 1
1.4110 0.0953 1
0.6419 -0.9163 1
2.1163 0.0000 2
1.3350 -1.6094 2
1.3610 -0.5108 2
2.0541 0.1823 2
2.2083 -0.5108 2
2.7344 1.2809 2
2.0412 0.4700 2
1.8718 -0.9163 2
1.7405 -0.9163 2
2.6101 0.4700 2
2.3224 1.8563 3
2.2192 2.0669 3
2.2618 1.1314 3
3.9853 0.9163 3
2.7600 2.0281 3
1 1
6 Nag_NotEqualCovar Nag_DiscrimPredict
1.6292 -0.9163
2.5572 1.6094
2.5649 -0.2231
0.9555 -2.3026
3.4012 -2.3026
3.0204 -0.2231

10.3 Program Results

nag_mv_discrim_group (g03dcc) Example Program Results

<table>
<thead>
<tr>
<th>Obs</th>
<th>Posterior probabilities</th>
<th>Allocated to group</th>
<th>Atypicality index</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.094 0.905 0.002</td>
<td>2</td>
<td>0.596 0.254 0.975</td>
</tr>
<tr>
<td></td>
<td>Column 1</td>
<td>Column 2</td>
<td>Column 3</td>
</tr>
<tr>
<td>---</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>2</td>
<td>0.005</td>
<td>0.168</td>
<td>0.827</td>
</tr>
<tr>
<td>3</td>
<td>0.019</td>
<td>0.920</td>
<td>0.062</td>
</tr>
<tr>
<td>4</td>
<td>0.697</td>
<td>0.303</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>0.317</td>
<td>0.013</td>
<td>0.670</td>
</tr>
<tr>
<td>6</td>
<td>0.032</td>
<td>0.366</td>
<td>0.601</td>
</tr>
</tbody>
</table>