Module 28.2: nag_canon_analysis

Canonical Analysis

`nag_canon_analysis` contains a procedure that performs canonical variate analysis for multivariate data.

Contents

Introduction 28.2.3

Procedures

`nag_canon_var` 28.2.5
Performs canonical variate analysis

Examples

Example 1: Calculation of Canonical Variates 28.2.11

Additional Examples 28.2.15

References 28.2.16

Introduction

Let the n by p data matrix consist of n observations of p variables, $x_1, x_2, \ldots, x_p$. If the individuals are classified into groups then *canonical variate analysis* examines the between-group structure. If the variables can be considered as coming from two sets then *canonical correlation analysis* examines the relationships between the two sets of variables.

For observations on one variable from a number of groups the standard method of analysis is the one-way analysis of variance which computes the ratio of the between-group sum of squares to the within-group sum of squares. Canonical variate analysis extends this to the multivariate case and looks at the matrix which is the ratio of the matrices of the between-group and within-group sums of squares and cross-products. This ratio measures the discrimination between the groups. The canonical variates are linear transformations of the original variables that are orthogonal and have the maximum discrimination in the smallest number of variates. It can be shown that the canonical variates correspond to the eigenvectors of the above matrix and that the eigenvalues give the amount of variation, and hence discrimination, for the variates. By examining the eigenvalues the number of variates required to provide the adequate discrimination can be decided upon. The canonical variate loadings give the relationship between the original variables and the canonical variates. By examining the individual observations in terms of the canonical variates (the scores) or the group means of the scores, the discrimination between groups can be investigated. The scores are adjusted so that the centroid is at the origin.

Only one procedure that performs *canonical variate analysis* is available at this release.

- `nag_canon_var` performs a canonical variate analysis (canonical discrimination) on a data matrix.

Procedure: nag_canon_var

1 Description

nag_canon_var calculates the canonical variables and scores for a (optionally weighted) data matrix.

2 Usage

```
USE nag_canon_analysis

CALL nag_canon_var(data, group, canon_var  [, optional arguments])
```

3 Arguments

Note. All array arguments are assumed-shape arrays. The extent in each dimension must be exactly that required by the problem. Notation such as 'x(n)' is used in the argument descriptions to specify that the array x must have exactly n elements.

This procedure derives the values of the following problem parameters from the shape of the supplied arrays.

$m \geq 1$ — the number of variables in the data matrix.

$p \geq 1$ — the number of variables included in the calculations. If the optional argument var_in_comp is not present then $p = m$, otherwise $p =$ COUNT(var_in_comp).

$g \geq 2$ — the number of groups into which the observations are partitioned, $g =$ MAXVAL(group).

$n \geq p + g$ — the number of observations in the data matrix.

ν — the number of canonical variates. The procedure calculates ν using $\nu = \min(r, g-1)$, where $r \leq p$ is the rank of the data matrix.

3.1 Mandatory Arguments

data(n, m) — real(kind=wp), intent(in)

Input: data(i, j) must contain the ith observation for the jth variable, for $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, m$.

group(n) — integer, intent(in)

Input: group(i) must contain the group number of the ith observation, for $i = 1, 2, \ldots, n$.

Constraints: group ≥ 1.

canon_var(:, :) — real(kind=wp), pointer

Output: canon_var(i, 1) contains the eigenvalue, γ_i^2, associated with the ith canonical variate, for $i = 1, 2, \ldots, \nu$. canon_var(i, 2) contains the proportion of the variation explained by the ith canonical variate, for $i = 1, 2, \ldots, \nu$. canon_var(i, 3) contains the canonical correlation, δ_i, associated with the ith canonical variate, for $i = 1, 2, \ldots, \nu$.

Note: the procedure creates a pointer array of shape (ν, 3).

3.2 Optional Arguments

Note. Optional arguments must be supplied by keyword, not by position. The order in which they are described below may differ from the order in which they occur in the argument list.

var_in_comp(m) — logical, intent(in), optional

Input: the variables to be included in the model.

If `var_in_comp(`i`)` = `.true.`, the ith variable is *included* in the calculations;

if `var_in_comp(`i`)` = `.false.`, the ith variable is *excluded* from the calculations.

Default: all variables are included in the calculations.

wt(n) — real(kind=*wp*), intent(in), optional

Input: the weights that are associated with the data values.

Default: an unweighted analysis is performed.

Constraints: `wt` ≥ 0.

freq_wt — logical, intent(in), optional

Input: specifies the type of weights supplied in `wt`.

If `freq_wt` = `.true.`, the weights given in `wt` are treated as frequencies and the effective number of observations is the sum of the weights;

if `freq_wt` = `.false.`, the weights given in `wt` are treated as being inversely proportional to the variance of the observations and the effective number of observations is the number of observations with non-zero weights.

Default: `freq_wt` = `.true.`.

Constraints: `freq_wt` need *not* be present if `wt` is *not* present and hence will be ignored.

tol — real(kind=*wp*), intent(in), optional

Input: `tol` is used to decide if the variables are of full rank and, if not, what is the rank of the variables. Decreasing the value of `tol` will have the effect of increasing the likelihood of the data matrix being treated as having full rank.

Default: `tol` = `MIN(`10^{-5}`, SQRT(EPSILON(1.0_`*wp*`)))`.

Constraints: `EPSILON(1.0_`*wp*`)` $\leq$ `tol` < 1.0.

score$(:,:)$ — real(kind=*wp*), pointer, optional

Output: the canonical variate scores. The jth column contains the scores for jth canonical variate, and `score(`i,j`)` contains the score for the ith observation on the jth canonical variate.

Note: the procedure creates a pointer array of shape (n, ν).

mean_score$(:,:)$ — real(kind=*wp*), pointer, optional

Output: the jth column contains the mean of the scores for jth canonical variate, and `score(`i,j`)` contains the mean score for the jth canonical variate in the ith group.

Note: the procedure creates a pointer array of shape (g, ν).

score_adjustment$(:)$ — real(kind=*wp*), pointer, optional

Output: `score_adjustment(`i`)` contains the adjustment term associated with ith canonical variate, for $i = 1, 2, \ldots, \nu$.

Note: the procedure creates a pointer array of shape (ν).

index(p) — integer, intent(out), optional

Output: the indexes of the variables included in the calculations.

loading$(:,:)$ — real(kind=*wp*), pointer, optional

Output: the canonical variate loadings. The jth column contains the loadings for the jth canonical variate, with `loading`(i,j) containing the loading of the ith variable included in the calculations (the original `index`(i) variable), on the jth canonical variate.

Note: the procedure creates a pointer array of shape (k,ν).

group_size(g) — integer, intent(out), optional

Output: `group_size`(i) contains the number of observations in the ith group.

chi_stat$(:)$ — real(kind=*wp*), pointer, optional

Output: `chi_stat`(i) contains the χ^2-statistic for the ith canonical variate, for $i = 1, 2, \ldots, \nu$.

Note: the procedure creates a pointer array of shape (ν).

sig_chi_stat$(:)$ — real(kind=*wp*), pointer, optional

Output: `sig_chi_stat`(i) contains significance level for the χ^2-statistic for the ith canonical variate, for $i = 1, 2, \ldots, \nu$.

Note: the procedure creates a pointer array of shape (ν).

chi_df$(:)$ — integer, pointer, optional

Output: `chi_df`(i) contains the number of degrees of freedom associated with the ith χ^2-statistic, for $i = 1, 2, \ldots, \nu$.

Note: the procedure creates a pointer array of shape (ν).

error — type(nag_error), intent(inout), optional

The NAG *fl*90 error-handling argument. See the Essential Introduction, or the module document `nag_error_handling` (1.2). You are recommended to omit this argument if you are unsure how to use it. If this argument is supplied, it *must* be initialized by a call to `nag_set_error` before this procedure is called.

4 Error Codes

Fatal errors (error%level = 3):

error%code	Description
301	An input argument has an invalid value.
302	An array argument has an invalid shape.
303	Array arguments have inconsistent shapes.
320	The procedure was unable to allocate enough memory.

Failures (error%level = 2):

error%code	Description
201	The routine has failed to converge. A singular value decomposition has failed to converge.
202	A canonical correlation is equal to 1. This will happen if the variables provide an exact indication as to which group every observation is allocated.
203	Invalid number of groups or variables. The effective number of groups is less than 2, or the effective number of variables plus the number of groups is greater than the effective number of observations.
204	The rank of the data matrix is zero. This will happen if all the variables are constant.

Warnings (error%level = 1):

error%code	**Description**
101	Optional argument is present but will be ignored. freq_wt is present when wt is not present.

5 Examples of Usage

A complete example of the use of this procedure appears in Example 1 of this module document.

6 Further Comments

6.1 Mathematical Background

For p variables of rank r observed on g groups, if B is the between-groups sum of squares and cross-products matrix and W is the within-group sum of squares and cross-products matrix, then the vector, a_1, which maximises the ratio

$$F = \frac{a_1^T Ba}{a_1^T Wa}$$

is an eigenvector of the pencil $B - \gamma^2 W$, and γ_1^2 is the corresponding maximum value of F.

The elements of the eigenvector a_1 are the component loadings of the first canonical variate. In a similar manner other eigenvectors, a_i, $i = 2, \ldots, \nu$, can be found such that $\gamma_1^2 \geq \gamma_2^2 \geq \cdots \geq \gamma_\nu^2$, where ν is $\min(r, g-1)$. The elements of the eigenvector a_i then correspond to the component loadings of the ith canonical variate.

The value $\pi_i = \gamma_i^2 / \Sigma\gamma_i^2$ gives the proportion of variation explained by the ith canonical variate. The values of the π_i give an indication as to how many canonical variates are needed to adequately describe the data, i.e., the dimensionality of the problem.

To test for a significant dimensionality greater than k, the χ^2-statistic

$$\left(n - 1 - g - \tfrac{1}{2}(p - g)\right) \sum_{j=k+1}^{\nu} \log(1 + \gamma_j)$$

can be used. This is asymptotically distributed as a χ^2-distribution with $(k - i)(g - 1 - i)$ degrees of freedom. If the test for $k = k_0$ is not significant, then the remaining tests for $k > k_0$ should be ignored.

The scores for the jth canonical variate are computed as $x_i a_j - \alpha_j$, where x_i is the ith row of the raw data matrix and α_j is the adjustment for the jth canonical variate.

6.2 Algorithmic Detail

This procedure calculates the canonical variates by means of a singular value decomposition (SVD) of a matrix V. Let the data matrix, with variable (column) means subtracted, be X and let its rank be r; then the r by $(g-1)$ matrix V is given by $V = Q_X^T Q_g$, where Q_g is an n by $(g-1)$ orthogonal matrix that defines the groups and Q_X is the first r rows of the orthogonal matrix Q either from the QR factorization of X:

$$X = QR$$

if X is of full column rank, i.e., $r = p$, or else from the SVD of X:

$$X = QDP^T.$$

Let the SVD of V be

$$V = U_x \Delta U_g^T;$$

then the non-zero elements of the diagonal matrix Δ, δ_i, for $i = 1, 2, \ldots, \nu$, are the ν canonical correlations associated with the ν canonical variates, where $\nu = \min(p, g - 1)$.

The eigenvalues, γ_i^2, of the within-group sums of squares matrix are given by

$$\gamma_i^2 = \frac{\delta_i^2}{1 - \delta_i^2}.$$

The loadings for the canonical variates are calculated from the matrix U_x. This matrix is scaled so that the canonical variates have unit within-group variance.

6.3 Accuracy

As the computation involves the use of orthogonal matrices and SVD rather than the traditional computing of a sum of squares matrix and the use of an eigenvalue decomposition, this procedure should be less affected by ill conditioned problems.

Example 1: Calculation of Canonical Variates

An unweighted canonical variate analysis is performed on a data set of nine observations. Each observation belongs to one of three groups and consists of four variables, although in this analysis the second variable is omitted from the calculations.

1 Program Text

Note. The listing of the example program presented below is double precision. Single precision users are referred to Section 5.2 of the Essential Introduction for further information.

```
PROGRAM nag_canon_analysis_ex01

  ! Example Program Text for nag_canon_analysis
  ! NAG fl90, Release 4. NAG Copyright 2000.

  ! .. Use Statements ..
  USE nag_canon_analysis, ONLY : nag_canon_var
  USE nag_examples_io, ONLY : nag_std_in, nag_std_out
  USE nag_write_mat, ONLY : nag_write_gen_mat
  ! .. Implicit None Statement ..
  IMPLICIT NONE
  ! .. Intrinsic Functions ..
  INTRINSIC COUNT, KIND, MAXVAL, SIZE
  ! .. Parameters ..
  INTEGER, PARAMETER :: wp = KIND(1.0D0)
  ! .. Local Scalars ..
  INTEGER :: g, i, k, m, n, num_drop_var, v
  ! .. Local Arrays ..
  INTEGER, POINTER :: chi_df(:)
  INTEGER, ALLOCATABLE :: group(:), index_drop_var(:)
  REAL (wp), POINTER :: canon_var(:,:), chi_stat(:), loading(:,:), &
   mean_score(:,:), score(:,:), score_adjustment(:), sig_chi_stat(:)
  REAL (wp), ALLOCATABLE :: data(:,:)
  LOGICAL, ALLOCATABLE :: var_in_comp(:)
  ! .. Executable Statements ..
  WRITE (nag_std_out,*) &
   'Example Program Results for nag_canon_analysis_ex01'

  READ (nag_std_in,*)          ! Skip heading in data file
  READ (nag_std_in,*) n, m, num_drop_var

  ALLOCATE (data(n,m),group(n),var_in_comp(m), &
   index_drop_var(num_drop_var)) ! Allocate storage

  NULLIFY (score,loading,canon_var,chi_stat,mean_score,score_adjustment, &
   sig_chi_stat)

  DO i = 1, n
    READ (nag_std_in,*) data(i,:), group(i)
  END DO

  READ (nag_std_in,*) index_drop_var
  var_in_comp = .TRUE.
  var_in_comp(index_drop_var) = .FALSE.

  g = MAXVAL(group)
  k = COUNT(var_in_comp)

  CALL nag_canon_var(data,group,canon_var,var_in_comp=var_in_comp, &
   loading=loading,score=score,mean_score=mean_score, &
   score_adjustment=score_adjustment,chi_stat=chi_stat, &
```

```
     sig_chi_stat=sig_chi_stat,chi_df=chi_df)

    WRITE (nag_std_out,*)
    WRITE (nag_std_out,*) &
     'Eigenvalues   Percentage     Chisq        Sig      DF'
    WRITE (nag_std_out,*) '               variation    '

    v = SIZE(canon_var,1)
    DO i = 1, v
      WRITE (nag_std_out,'(4f12.4,i6)') canon_var(i,1), canon_var(i,2), &
       chi_stat(i), sig_chi_stat(i), chi_df(i)
    END DO

    WRITE (nag_std_out,*)

    CALL nag_write_gen_mat(loading,format='f12.4',title='Loadings')

    WRITE (nag_std_out,*)

    CALL nag_write_gen_mat(mean_score,format='f12.4', &
     title='Group mean canonical variate scores')

    WRITE (nag_std_out,*)

    CALL nag_write_gen_mat(score,format='f12.4',title= &
     'Canonical variate scores')

    WRITE (nag_std_out,*)
    WRITE (nag_std_out,*) 'The score adjustment terms'
    WRITE (nag_std_out,'(10f12.4)') score_adjustment(1:v)

    DEALLOCATE (data,group,var_in_comp,score,loading,canon_var,chi_stat, &
     mean_score,score_adjustment,sig_chi_stat) ! Deallocate storage

  END PROGRAM nag_canon_analysis_ex01
```

2 Program Data

```
Example Program Data for nag_canon_analysis_ex01
 9 4 1                  : n, m, num_drop_var
 13.3 99.1 10.6 21.2  1 : data (1,1:m), group(1)
 13.6 89.2 10.2 21.0  2 : data (2,1:m), group(2)
 14.2 76.3 10.7 21.1  3
 13.4 44.4  9.4 21.0  1
 13.2 77.2  9.6 20.1  2
 13.9 89.2 10.4 19.8  3
 12.9 72.4 10.0 20.5  1
 12.2 89.3  9.9 20.7  2
 13.9 77.1 11.0 19.1  3 : data (n,1:m), group(n)
 2                      : index of var NOT included
```

3 Program Results

```
Example Program Results for nag_canon_analysis_ex01

Eigenvalues   Percentage     Chisq        Sig      DF
               variation
     3.5238      0.9795      7.9032      0.2453     6
     0.0739      0.0205      0.3564      0.8368     2

Loadings
      -1.7070      0.7277
      -1.3481      0.3138
       0.9327      1.2199

Group mean canonical variate scores
       0.9841      0.2797
       1.1805     -0.2632
      -2.1646     -0.0164

Canonical variate scores
       0.2844      0.9067
       0.1250      0.7555
      -1.4800      1.4710
       1.5448      0.3589
       0.7772     -0.8218
      -1.7760     -0.4273
       1.1231     -0.4266
       2.6394     -0.7234
      -3.2378     -1.0930

The score adjustment terms
   -17.5041     37.9600
```

Additional Examples

Not all example programs supplied with NAG *fl*90 appear in full in this module document. The following additional examples, associated with this module, are available.

`nag_canon_analysis_ex02`

Weighted canonical variate analysis.

References

[1] Chatfield C and Collins A J (1980) *Introduction to Multivariate Analysis.* Chapman and Hall.

[2] Gnanadesikan R (1977) *Methods for Statistical Data Analysis of Multivariate Observations* Wiley

[3] Hammarling S (1985) The singular value decomposition in multivariate statistics *ACM Signum Newsletter* **20(3)** 2–25

[4] Kendall M G and Stuart A (1976) *Advanced Theory of Statistics, Vol 3* Griffin

[5] Krzanowski W J (1988) *Principles of Multivariate Analysis* Oxford University Press