NAG Library Function Document

nag_mv_canon_corr (g03adc)

1 Purpose

nag_mv_canon_corr (g03adc) performs canonical correlation analysis upon input data matrices.

2 Specification

```
#include <nag.h>
#include <nagg03.h>
void nag_mv_canon_corr (Integer n, Integer m, const double z[], Integer tdz,
     const Integer isz[], Integer nx, Integer ny, const double wt[],
     double e[], Integer tde, Integer *ncv, double cvx[], Integer tdcvx,
     double cvy[], Integer tdcvy, double tol, NagError *fail)
```

3 Description

Let there be two sets of variables, x and y. For a sample of n observations on n_x variables in a data matrix X and n_y variables in a data matrix Y, canonical correlation analysis seeks to find a small number of linear combinations of each set of variables in order to explain or summarise the relationships between them. The variables thus formed are known as canonical variates.

Let the variance-covariance matrix of the two datasets be

$$\begin{pmatrix} S_{xx} & S_{xy} \\ S_{yx} & S_{yy} \end{pmatrix}$$

and let

$$\Sigma = S_{yy}^{-1} S_{yx} S_{xx}^{-1} S_{xy}$$

then the canonical correlations can be calculated from the eigenvalues of the matrix Σ. However, nag_mv_canon_corr (g03adc) calculates the canonical correlations by means of a singular value decomposition (SVD) of a matrix V. If the rank of the data matrix X is k_x and the rank of the data matrix Y is k_y, and both X and Y have had variable (column) means subtracted, then the k_x by k_y matrix V is given by:

$$V = Q_x^{\mathrm{T}} Q_y,$$

where Q_x is the first k_x rows of the orthogonal matrix Q either from the QR decomposition of X if X is of full column rank, i.e., $k_x = n_x$:

$$X = Q_x R_x$$

or from the SVD of X if $k_x < n_x$:

$$X = Q_x D_x P_x^{\mathrm{T}}.$$

Similarly Q_y is the first k_y rows of the orthogonal matrix Q either from the QR decomposition of Y if Y is of full column rank, i.e., $k_y = n_y$:

$$Y = Q_y R_y$$

or from the SVD of Y if $k_y < n_y$:

$$Y = Q_y D_y P_y^{\mathrm{T}}.$$

Let the SVD of V be:

$$V = U_x \Delta U_y^{\mathrm{T}}$$

then the nonzero elements of the diagonal matrix Δ, δ_i, for $i = 1, 2, \ldots, l$, are the l canonical correlations associated with the l canonical variates, where $l = \min(k_x, k_y)$.

The eigenvalues, λ_i^2, of the matrix Σ are given by:

$$\lambda_i^2 = \frac{\delta_i^2}{1 + \delta_i^2}.$$

The value of $\pi_i = \lambda_i^2 / \sum \lambda_i^2$ 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 i the χ^2 statistic:

$$\left(n - \frac{1}{2}(k_x + k_y + 3)\right) \sum_{j=i+1}^{l} \log\left(1 + \lambda_j^2\right)$$

can be used. This is asymptotically distributed as a χ^2 distribution with $(k_x - i)(k_y - i)$ degrees of freedom. If the test for $i = k_{\min}$ is not significant, then the remaining tests for $i > k_{\min}$ should be ignored.

The loadings for the canonical variates are calculated from the matrices U_x and U_y respectively. These matrices are scaled so that the canonical variates have unit variance.

4 References

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

Kendall M G and Stuart A (1976) *The Advanced Theory of Statistics (Volume 3)* (3rd Edition) Griffin

Morrison D F (1967) *Multivariate Statistical Methods* McGraw–Hill

5 Arguments

1: **n** – Integer *Input*

On entry: the number of observations, n.

Constraint: **n** $>$ **nx** $+$ **ny**.

2: **m** – Integer *Input*

On entry: the total number of variables, m.

Constraint: **m** $\geq$ **nx** $+$ **ny**.

3: **z**[**n** $\times$ **tdz**] – const double *Input*

On entry: **z**[$(i-1) \times$ **tdz** $+ j - 1$] must contain the ith observation for the jth variable, for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$.

Both x and y variables are to be included in **z**, the indicator array, **isz**, being used to assign the variables in **z** to the x or y sets as appropriate.

4: **tdz** – Integer *Input*

On entry: the stride separating matrix column elements in the array **z**.

Constraint: **tdz** $\geq$ **m**.

5: **isz**[**m**] – const Integer *Input*

On entry: **isz**$[j-1]$ indicates whether or not the jth variable is to be included in the analysis and to which set of variables it belongs.

If **isz**$[j-1] > 0$, then the variable contained in the jth column of **z** is included as an x variable in the analysis.

If **isz**$[j-1] < 0$, then the variable contained in the jth column of **z** is included as a y variable in the analysis.

If **isz**$[j-1] = 0$, then the variable contained in the jth column of **z** is not included in the analysis.

Constraint: only **nx** elements of **isz** can be > 0 and only **ny** elements of **isz** can be < 0.

6: **nx** – Integer *Input*

On entry: the number of x variables in the analysis, n_x.

Constraint: **nx** ≥ 1.

7: **ny** – Integer *Input*

On entry: the number of y variables in the analysis, n_y.

Constraint: **ny** ≥ 1.

8: **wt**[**n**] – const double *Input*

On entry: the elements of **wt** must contain the weights to be used in the analysis. The effective number of observations is the sum of the weights. If **wt**$[i-1] = 0.0$ then the ith observation is not included in the analysis.

If weights are not provided then **wt** must be set to **NULL** and the effective number of observations is **n**.

Constraints:

if **wt** is not **NULL**, **wt**$[i-1] \geq 0.0$, for $i = 1, 2, \ldots, n$;
$\sum_{i=1}^{n}$**wt**$[i-1] \geq$ **nx** + **ny** + 1.

9: **e**[**min**(**nx**, **ny**) × **tde**] – double *Output*

On exit: the statistics of the canonical variate analysis. **e**$[(i-1) \times$ **tde**$]$, the canonical correlations, δ_i, for $i = 1, 2, \ldots, l$.

e$[(i-1) \times$ **tde** $+ 1]$, the eigenvalues of Σ, λ_i^2, for $i = 1, 2, \ldots, l$.

e$[(i-1) \times$ **tde** $+ 2]$, the proportion of variation explained by the ith canonical variate, for $i = 1, 2, \ldots, l$.

e$[(i-1) \times$ **tde** $+ 3]$, the χ^2 statistic for the ith canonical variate, for $i = 1, 2, \ldots, l$.

e$[(i-1) \times$ **tde** $+ 4]$, the degrees of freedom for χ^2 statistic for the ith canonical variate, for $i = 1, 2, \ldots, l$.

e$[(i-1) \times$ **tde** $+ 5]$, the significance level for the χ^2 statistic for the ith canonical variate, for $i = 1, 2, \ldots, l$.

10: **tde** – Integer *Input*

On entry: the stride separating matrix column elements in the array **e**.

Constraint: **tde** ≥ 6.

11: **ncv** – Integer * *Output*

On exit: the number of canonical correlations, l. This will be the minimum of the rank of X and the rank of Y.

12: **cvx**[**nx** × **tdcvx**] – double *Output*

On exit: the canonical variate loadings for the x variables. **cvx**$[(i-1) \times$ **tdcvx** $+ j - 1]$ contains the loading coefficient for the ith x variable on the jth canonical variate.

13: **tdcvx** – Integer *Input*

On entry: the stride separating matrix column elements in the array **cvx**.

Constraint: **tdcvx** $\geq \min($**nx**,**ny**$)$.

14: **cvy**[**ny** × **tdcvy**] – double *Output*

On exit: the canonical variate loadings for the y variables. **cvy**$[(i-1) \times$ **tdcvy** $+ j - 1]$ contains the loading coefficient for the ith y variable on the jth canonical variate.

15: **tdcvy** – Integer *Input*

On entry: the stride separating matrix column elements in the array **cvy**.

Constraint: **tdcvy** $\geq \min($**nx**,**ny**$)$.

16: **tol** – double *Input*

On entry: the value of **tol** is used to decide if the variables are of full rank and, if not, what is the rank of the variables. The smaller the value of **tol** the stricter the criterion for selecting the singular value decomposition. If a non-negative value of **tol** less than ***machine precision*** is entered, then the square root of ***machine precision*** is used instead.

Constraint: **tol** ≥ 0.0.

17: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_2_INT_ARG_LT

On entry, **tdz** $= \langle value \rangle$ while **m** $= \langle value \rangle$. These arguments must satisfy **tdz** $\geq$ **m**.

NE_3_INT_ARG_CONS

On entry, **m** $= \langle value \rangle$, **nx** $= \langle value \rangle$ and **ny** $= \langle value \rangle$. These arguments must satisfy **m** $\geq$ **nx** $+$ **ny**.

On entry, **n** $= \langle value \rangle$, **nx** $= \langle value \rangle$ and **ny** $= \langle value \rangle$. These arguments must satisfy **n** $>$ **nx** $+$ **ny**.

On entry, **tdcvx** $= \langle value \rangle$, **nx** $= \langle value \rangle$ and **ny** $= \langle value \rangle$. These arguments must satisfy **tdcvx** $\geq \min($**nx**,**ny**$)$.

On entry, **tdcvy** $= \langle value \rangle$, **nx** $= \langle value \rangle$ and **ny** $= \langle value \rangle$. These arguments must satisfy **tdcvy** $\geq \min($**nx**,**ny**$)$.

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_CANON_CORR_1

A canonical correlation is equal to one. This will happen if the x and y variables are perfectly correlated.

NE_INT_ARG_LT

On entry, **nx** $= \langle value \rangle$.
Constraint: **nx** ≥ 1.

On entry, **ny** $= \langle value \rangle$.
Constraint: **ny** ≥ 1.

On entry, **tde** $= \langle value \rangle$.
Constraint: **tde** ≥ 6.

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.

NE_MAT_RANK_ZERO

The rank of the X matrix or the rank of the Y matrix is zero. This will happen if all the x and y variables are constants.

NE_NEG_WEIGHT_ELEMENT

On entry, **wt**$[\langle value \rangle] = \langle value \rangle$.
Constraint: when referenced, all elements of **wt** must be non-negative.

NE_OBSERV_LT_VAR

With weighted data, the effective number of observations given by the sum of weights $= \langle value \rangle$, while number of variables included in the analysis, **nx** + **ny** $= \langle value \rangle$.
Constraint: Effective number of observations $\geq$ **nx** + **ny** + 1.

NE_REAL_ARG_LT

On entry, **tol** must not be less than 0.0: **tol** $= \langle value \rangle$.

NE_SVD_NOT_CONV

The singular value decomposition has failed to converge. This is an unlikely error exit.

NE_VAR_INCL_INDICATED

The number of variables, **nx** in the analysis $= \langle value \rangle$, while the number of x variables included in the analysis via array **isz** $= \langle value \rangle$.
Constraint: these two numbers must be the same.

The number of variables, **ny** in the analysis $= \langle value \rangle$, while the number of y variables included in the analysis via array **isz** $= \langle value \rangle$.
Constraint: these two numbers must be the same.

7 Accuracy

As the computation involves the use of orthogonal matrices and a singular value decomposition rather than the traditional computing of a sum of squares matrix and the use of an eigenvalue decomposition, nag_mv_canon_corr (g03adc) should be less affected by ill conditioned problems.

8 Parallelism and Performance

nag_mv_canon_corr (g03adc) is not threaded in any implementation.

9 Further Comments

None.

10 Example

A sample of nine observations with two variables in each set is read in. The second and third variables are x variables while the first and last are y variables. Canonical variate analysis is performed and the results printed.

10.1 Program Text

```
/* nag_mv_canon_corr (g03adc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 *
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg03.h>

#define Z(I, J)   z[(I) *tdz + J]
#define CVX(I, J) cvx[(I) *tdcvx + J]
#define CVY(I, J) cvy[(I) *tdcvy + J]
#define E(I, J)   e[(I) *tde + J]
int main(void)
{
  Integer exit_status = 0, i, *isz = 0, j, m, n, ncv, nx, ny, tdcvx, tdcvy,
          tde = 6;
  Integer tdz;
  NagError fail;
  char weight[2];
  double *cvx = 0, *cvy = 0, *e = 0, tol, *wt = 0, *wtptr, *z = 0;

  INIT_FAIL(fail);

  printf("nag_mv_canon_corr (g03adc) Example Program Results\n\n");

  /* Skip heading 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 "", &nx);
#else
  scanf("%" NAG_IFMT "", &nx);
#endif
#ifdef _WIN32
  scanf_s("%" NAG_IFMT "", &ny);
```

```
#else
  scanf("%" NAG_IFMT "", &ny);
#endif
#ifdef _WIN32
  scanf_s("%1s", weight, (unsigned)_countof(weight));
#else
  scanf("%1s", weight);
#endif

  if (nx >= 1 && ny >= 1 && n > nx + ny && m >= nx + ny) {
    if (!(z = NAG_ALLOC(n * m, double)) ||
        !(wt = NAG_ALLOC(n, double)) ||
        !(isz = NAG_ALLOC(m, Integer)) ||
        !(cvx = NAG_ALLOC(nx * (MIN(nx, ny)), double)) ||
        !(cvy = NAG_ALLOC(ny * (MIN(nx, ny)), double)) ||
        !(e = NAG_ALLOC((MIN(nx, ny)) * 6, double))

           )
    {
      printf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }
    tdz = m;
    tdcvx = MIN(nx, ny);
    tdcvy = MIN(nx, ny);
    tde = 6;
  }
  else {
    printf("Invalid nx or ny or n or m.\n");
    exit_status = 1;
    return exit_status;
  }
  if (*weight == 'W') {
    for (i = 0; i < n; ++i) {
      for (j = 0; j < m; ++j)
#ifdef _WIN32
        scanf_s("%lf", &Z(i, j));
#else
        scanf("%lf", &Z(i, j));
#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", &Z(i, j));
#else
        scanf("%lf", &Z(i, j));
#endif
    }
    wtptr = 0;
  }
  for (j = 0; j < m; ++j)
#ifdef _WIN32
    scanf_s("%" NAG_IFMT "", &isz[j]);
#else
    scanf("%" NAG_IFMT "", &isz[j]);
#endif
  tol = 1e-6;

  /* nag_mv_canon_corr (g03adc).
   * Canonical correlation analysis
   */
```

```
  nag_mv_canon_corr(n, m, z, tdz, isz, nx, ny, wtptr, e, tde,
                    &ncv, cvx, tdcvx, cvy, tdcvy, tol, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_mv_canon_corr (g03adc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }

  printf("\n%s%2" NAG_IFMT "%s%2" NAG_IFMT "\n\n", "Rank of x = ", nx,
         " Rank of y = ", ny);
  printf("    Canonical    Eigenvalues Percentage      Chisq "
         "          DF          Sig\n");
  printf("    correlations              variation\n");

  for (i = 0; i < ncv; ++i) {
    for (j = 0; j < 6; ++j)
      printf("%12.4f", E(i, j));
    printf("\n");
  }
  printf("\nCanonical coefficients for x\n");
  for (i = 0; i < nx; ++i) {
    for (j = 0; j < ncv; ++j)
      printf("%9.4f", CVX(i, j));
    printf("\n");
  }
  printf("\nCanonical coefficients for y\n");
  for (i = 0; i < ny; ++i) {
    for (j = 0; j < ncv; ++j)
      printf("%9.4f", CVY(i, j));
    printf("\n");
  }
END:
  NAG_FREE(z);
  NAG_FREE(wt);
  NAG_FREE(isz);
  NAG_FREE(cvx);
  NAG_FREE(cvy);
  NAG_FREE(e);
  return exit_status;
}
```

10.2 Program Data

```
nag_mv_canon_corr (g03adc) Example Program Data
 9 4 2 2 U
 80.0 58.4 14.0 21.0
 75.0 59.2 15.0 27.0
 78.0 60.3 15.0 27.0
 75.0 57.4 13.0 22.0
 79.0 59.5 14.0 26.0
 78.0 58.1 14.5 26.0
 75.0 58.0 12.5 23.0
 64.0 55.5 11.0 22.0
 80.0 59.2 12.5 22.0
 -1    1    1    -1
```

10.3 Program Results

```
nag_mv_canon_corr (g03adc) Example Program Results


Rank of x =  2 Rank of y =  2

    Canonical    Eigenvalues Percentage      Chisq           DF          Sig
    correlations              variation
      0.9570     10.8916      0.9863     14.3914      4.0000      0.0061
      0.3624      0.1512      0.0137      0.7744      1.0000      0.3789

Canonical coefficients for x
  -0.4261   1.0337
```

```
  -0.3444  -1.1136

Canonical coefficients for y
  -0.1415   0.1504
  -0.2384  -0.3424
```