Module 25.2: nag_correl
Correlation Analysis

`nag_correl` contains procedures that calculate the correlation coefficients for a set of data values.

Contents

Procedure: nag_prod_mom_correl

1 Description

Given a set of n observations of l variables this procedure computes the (optionally weighted) means and sums of squares and cross-products of the deviations about the means. The variance-covariance matrix, the standard deviations and the Pearson product-moment correlation matrix are then computed from these basic results.

Note: all the output arguments of this procedure are optional. However, at least one output argument must be present in every call statement.

2 Usage

```
USE nag_correl
CALL nag_prod_mom_correl(data  [, 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 '$\mathbf{x}(n)$' 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.

$n > 1$ — the number of observations in the data matrix

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

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

3.1 Mandatory Argument

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

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

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

Input: the variables to be included in the model.

If `var_in_correl(`j`) = .true.`, the jth variable is *included* in the calculations;

if `var_in_correl(`j`) = .false.`, the jth 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.

Note: if `wt` is present the calculations are performed on weighted data.

Default: `wt` = 1.0.

Constraints: `wt` ≥ 0.

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

Output: `index` is used in accessing the elements of `cov`, `correl`, `std` and `mean`.
`index(`j`)` contains the index assigned to the jth variable of the input array `data`. If the jth variable of the input array is not included in the calculations (i.e., `var_in_correl(`j`) = .false.`) then `index(`j`) = 0`.

Note:

If all the variables are included in the calculations then `index(`j`) =` j, $j = 1, \ldots, m$;

if some variables are not included in the calculations then:

`index(`j`) = 0`, if `var_in_correl(`j`) = .false.`;

`index(`j`) = COUNT(var_in_correl(:`j`))`, otherwise.

freq_wt — logical, intent(in), optional

Input: specifies the divisors to be used in the calculation of `std` and `cov`.

If `freq_wt = .true.`, then $\sum_{i=1}^{n}$ `wt(`i`)` -1 is the divisor used to calculate `std` and `cov`;

if `freq_wt = .false.`, then $n_0 - 1$ is the divisor used to calculate `std` and `cov`, where n_0 is the number of observations with non-zero weights.

Default: `freq_wt = .true.`.

mean(l) — real(kind=*wp*), intent(out), optional

Output: if `index(`j`)` $\neq 0$ then `mean(index(`j`))` contains the calculated mean for the jth variable of the array `data`, $\bar{x}_j$.

cov(l, l) — real(kind=*wp*), intent(out), optional

Output: if `index(`j`)` $\neq 0$ and `index(`k`)` $\neq 0$ then `cov(index(`j`), index(`k`))` for any $1 \leq j \leq m$ and $1 \leq k \leq m$, contains the calculated variance-covariance between the jth and kth variables of the array `data`, c_{jk}.

std(l) — real(kind=*wp*), intent(out), optional

Output: if `index(`j`)` $\neq 0$ then `std(index(`j`))` contains the calculated standard deviation for the jth variable of the array `data`, s_j.

correl(l, l) — real(kind=*wp*), intent(out), optional

Output: if `index(`j`)` $\neq 0$ and `index(`k`)` $\neq 0$ then `correl(index(`j`), index(`k`))` for any $1 \leq j \leq m$ and $1 \leq k \leq m$, contains the calculated Pearson product-moment correlation coefficient between the jth and kth variables of the array `data`, r_{jk}.

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.

305 Invalid absence of an optional argument.

320 The procedure was unable to allocate enough memory.

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

Let x_{ij} denote the ith observation for the jth variable and w_i denote the weight for the ith observation. The output statistics are then defined as follows.

- The mean vector:

$$\bar{x}_j = \frac{\sum_{i=1}^{n} w_i \; x_{ij}}{\sum_{i}^{n} w_i}, \qquad j = 1, 2, \ldots, l.$$

- The variance-covariance matrix:

$$c_{jk} = \frac{\sum_{i=1}^{n} w_i \; (x_{ij} - \bar{x}_j) \; (x_{ik} - \bar{x}_k)}{n_w}, \qquad j = 1, 2, \ldots, l, \quad k = 1, 2, \ldots, l;$$

  where $n_w = \sum_{i}^{n} w_i - 1$, if `freq_wt` = `.true.` otherwise $n_w = n_0 - 1$ (n_0 is the number of observations with non-zero weight).

- The standard deviation vector:

$$s_j = \sqrt{c_{jj}}, \qquad j = 1, 2, \ldots, l.$$

- The correlation matrix:

$$r_{jk} = \frac{c_{jk}}{s_j s_k}, \qquad j = 1, 2, \ldots, l, \quad k = 1, 2, \ldots, l.$$

6.2 Algorithmic Detail

A one pass algorithm is used, see West [2].

Procedure: nag_part_correl

1 Description

nag_part_correl calculates the partial correlation matrix and the partial variance-covariance matrix from a correlation or variance-covariance matrix.

In general, let a set of variables be partitioned into two groups Y and X with n_y variables in Y and n_x variables in X and let the variance-covariance matrix of all $n_y + n_x$ variables be partitioned into

$$\begin{bmatrix} \Sigma_{xx} & \Sigma_{yx} \\ \Sigma_{xy} & \Sigma_{yy} \end{bmatrix}.$$

The partial variance-covariance of Y conditional on fixed values of the X variables is given by:

$$\Sigma_{y|x} = \Sigma_{yy} - \Sigma_{yx}\Sigma_{xx}^{-1}\Sigma_{xy}.$$

The partial correlation matrix is then computed by standardising $\Sigma_{y|x}$ as

$$\rho = \mathrm{diag}(\Sigma_{y|x})^{-1/2}\Sigma_{y|x}\,\mathrm{diag}(\Sigma_{y|x})^{-1/2}.$$

2 Usage

```
USE nag_correl
CALL nag_part_correl(cov, var_in_x, part_correl  [, 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.

$m \geq 3$ — the number of variables in the variance-covariance matrix or correlation matrix

$n_y \geq 2$ — the number of Y variables. If the optional argument var_in_model is present then n_y = COUNT((.NOT. var_in_x) .AND. var_in_model), otherwise n_y = COUNT(.NOT. var_in_x)

3.1 Mandatory Arguments

cov(m, m) — real(kind=*wp*), intent(in)

Input: the variance-covariance or correlation matrix for the m variables as calculated by nag_prod_mom_correl. Only the upper triangle need be given.

Constraints: cov must be both full rank and positive definite.

var_in_x(m) — logical, intent(in)

Input: specifies the X variables.

If var_in_x(i) = .true., the ith variable is *included* in X;

if var_in_x(i) = .false., the ith variable is *included* in Y.

Note: if var_in_model is present and var_in_model(i) = .false., then var_in_x(i) has no effect.

part_correl(n_y, n_y) — real(kind=*wp*), intent(out)

Output: the strict upper triangle of `part_correl` contains the strict upper triangular part of ρ the n_y by n_y partial correlation matrix. If the input matrix `cov` is the variance-covariance matrix then the lower triangle of `part_correl` contains the lower triangle of $\Sigma_{y|x}$, the n_y by n_y partial variance-covariance matrix. If the input matrix `cov` is the correlation matrix then the lower triangle of `part_correl` contains the lower triangle of the partial variance-covariance matrix for standardised variables.

If `index(`i`)` $\neq 0$ and `index(`j`)` $\neq 0$ (see optional argument `index`) then the array element `part_correl(index(`i`),index(`j`))` contains the computed result for the ith and jth variables of the input variance-covariance or correlation matrix.

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

Input: the variables to be included in the calculations as either X or Y variables as defined by the argument `var_in_x`.

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

if `var_in_model(`i`) = .false.`, the ith variable is *excluded*.

Default: all variables are included in the model.

Constraints: n_x and n_y, the number of X and Y variables included in the model, respectively, must satisfy $n_x \geq 1$ and $n_y \geq 2$.

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

Output: `index` is used in accessing the elements of `part_correl`.

`index(`i`)` contains the index assigned to the ith variable of the input array `cov`. If the ith variable of the input array is not included as a Y variable in the calculations (i.e., `var_in_x(`i`) = .true.` or `var_in_model(`i`) = .false.`) then `index(`i`) = 0`.

Note:

If `var_in_model` is absent then `index(`i`) = COUNT(.NOT.var_in_x(:`i`))`;

if `var_in_model` is present then: `index(`i`) = 0`;

if `var_in_model(`i`) = .false.`; `index(`i`) = COUNT(.NOT.var_in_x(:`i`).AND.var_in_model(:`i`)))`, otherwise.

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.

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

Partial correlation can be used to explore the association between pairs of random variables in the presence of other variables. For three variables, y_1, y_2 and x_3 the partial correlation coefficient between y_1 and y_2 given x_3 is computed as

$$\frac{r_{12} - r_{13}r_{23}}{\sqrt{(1 - r_{13}^2)(1 - r_{23}^2)}},$$

where r_{ij} is the product-moment correlation coefficient between variables with subscripts i and j. The partial correlation coefficient is a measure of the linear association between y_1 and y_2 having eliminated the effect due to both y_1 and y_2 being linearly associated with x_3. That is, it is a measure of association between y_1 and y_2 conditional upon fixed values of x_3. Like the full correlation coefficients the partial correlation coefficient takes a value in the range $(-1, 1)$ with the value 0 indicating no association.

To test the hypothesis that a partial correlation is zero under the assumption that the data has an approximately Normal distribution a test similar to the test for the full correlation coefficient can be used. If r is the computed partial correlation coefficient then the appropriate t statistic is

$$r\sqrt{\frac{n - n_x - 2}{1 - r^2}},$$

which has approximately a Student's t-distribution with $n - n_x - 2$ degrees of freedom, where n is the number of observations from which the full correlation coefficients were computed, and n_x is the number of variables included as X variables (see argument `var_in_x`).

Example 1: Calculation of correlation and partial correlation coefficients

This example program shows how to use `nag_part_correl` for calculating partial correlations from a set of data values.

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_correl_ex01

  ! Example program for nag_correl
  ! NAG f190, Release 3. NAG Copyright 1997.

  ! .. Use Statements ..
  USE nag_examples_io, ONLY : nag_std_out, nag_std_in
  USE nag_correl, ONLY : nag_prod_mom_correl, nag_part_correl
  USE nag_write_mat, ONLY : nag_write_tri_mat
  ! .. Implicit None Statement ..
  IMPLICIT NONE
  ! .. Intrinsic Functions ..
  INTRINSIC KIND
  ! .. Parameters ..
  INTEGER, PARAMETER :: wp = KIND(1.0D0)
  ! .. Local Scalars ..
  INTEGER :: i, k, m, n, nx
  ! .. Local Arrays ..
  INTEGER, ALLOCATABLE :: index_x(:)
  REAL (wp), ALLOCATABLE :: correl(:,:), data(:,:), part_correl(:,:)
  LOGICAL, ALLOCATABLE :: var_in_x(:)
  CHARACTER (5), ALLOCATABLE :: label(:)
  ! .. Executable Statements ..

  WRITE (nag_std_out,*) 'Example Program Results for nag_correl_ex01'

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

  ALLOCATE (data(n,m),correl(m,m)) ! Allocate storage

  ! Read data
  READ (nag_std_in,*) (data(i,:),i=1,n)

  WRITE (nag_std_out,*)

  CALL nag_prod_mom_correl(data,correl=correl)

  CALL nag_write_tri_mat('u',correl,format='f10.4',int_row_labels=.TRUE., &
   int_col_labels=.TRUE.,title='The correlation matrix for all the data')

  ! Read number of x variables
  READ (nag_std_in,*) nx

  ALLOCATE (part_correl(m-nx,m-nx),index_x(nx),var_in_x(m), &
   label(m-nx))                    ! Allocate storage

  ! Read indexes of x variables
  READ (nag_std_in,*) index_x
  var_in_x = .FALSE.
  var_in_x(index_x) = .TRUE.
```

```
      CALL nag_part_correl(correl,var_in_x,part_correl=part_correl)

      k = 0
      DO i = 1, m
        IF ( .NOT. var_in_x(i)) THEN
          k = k + 1
          WRITE (label(k),'(a2,i2,a)') 'x(', i, ')'
        END IF
      END DO
      WRITE (nag_std_out,*)

      CALL nag_write_tri_mat('u',part_correl,diag='u',format='f10.4', &
       row_labels=label,col_labels=label,title='Partial correlation matrix')

      DEALLOCATE (data,correl,part_correl,index_x,var_in_x, &
       label)                           ! Deallocate storage

    END PROGRAM nag_correl_ex01
```

2 Program Data

```
Example Program Data for nag_correl_ex01
   5    20                      : m, n
   11.25 48.9 7.43 2.270 15.48
   10.87 47.7 7.45 1.971 14.97
   11.18 48.2 7.44 1.979 14.20
   10.62 49.0 7.38 2.026 15.02
   11.02 47.4 7.43 1.974 12.92
   10.83 48.3 7.72 2.124 13.58
   11.18 49.3 7.05 2.064 14.12
   11.05 48.2 6.95 2.001 15.34
   11.15 49.1 7.12 2.035 14.52
   11.23 48.6 7.28 1.970 15.25
   10.94 49.9 7.45 1.974 15.34
   11.18 49.0 7.34 1.942 14.48
   11.02 48.2 7.29 2.063 12.92
   10.99 47.8 7.37 1.973 13.61
   11.03 48.9 7.45 1.974 14.20
   11.09 48.8 7.08 2.039 14.51
   11.46 51.2 6.75 2.008 16.07
   11.57 49.8 7.00 1.944 16.60
   11.07 47.9 7.04 1.947 13.41
   10.89 49.6 7.07 1.798 15.84 : data
  1                            : nx  number of x variables
  5                            : indexes of x variables
```

3 Program Results

```
 Example Program Results for nag_correl_ex01

 The correlation matrix for all the data
           1         2         3         4         5
 1     1.0000    0.4416   -0.5427    0.0696    0.3912
 2               1.0000   -0.5058   -0.0678    0.7057
 3                         1.0000    0.2768   -0.4352
 4                                   1.0000   -0.1494
 5                                             1.0000

 Partial correlation matrix
           x( 1)     x( 2)     x( 3)     x( 4)
```

```
x( 1)     1.0000    0.2538   -0.4495    0.1407
x( 2)               1.0000   -0.3115    0.0538
x( 3)                         1.0000    0.2379
x( 4)                                   1.0000
```

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_correl_ex02`

Calculation of the product-moment correlation coefficients from a set of data values.

`nag_correl_ex03`

Calculation of the Pearson product-moment correlation coefficients using a subset of the data with and without weights.

References

[1] Morrison D F (1967) *Multivariate Statistical Methods* McGraw-Hill

[2] West D H D (1979) Updating mean and variance estimates: An improved method *Comm. ACM* **22** 532–535