nag_mv_prin_comp (g03aac)

1. Purpose

nag_mv_prin_comp (g03aac) performs a principal component analysis on a data matrix; both the principal component loadings and the principal component scores are returned.

2. Specification

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

void nag_mv_prin_comp(Nag_PrinCompMat pcmatrix, Nag_PrinCompScores scores,
            Integer n, Integer m, double x[], Integer tdx, Integer isx[],
            double s[], double wt[], Integer nvar, double e[], Integer tde,
            double p[], Integer tdp, double v[], Integer tdv, NagError *fail)
```

3. Description

Let X be an n by p data matrix of n observations on p variables $x_1, x_2, \ldots, x_p$ and let the p by p variance-covariance matrix of $x_1, x_2, \ldots, x_p$ be S. A vector a_1 of length p is found such that:

$$a_1^T S a_1 \text{ is maximized subject to } a_1^T a_1 = 1.$$

The variable $z_1 = \sum_{i=1}^{p} a_{1i} x_i$ is known as the first principal component and gives the linear combination of the variables that gives the maximum variation. A second principal component, $z_2 = \sum_{i=1}^{p} a_{2i} x_i$, is found such that:

$$a_2^T S a_2 \text{ is maximized subject to } a_2^T a_2 = 1 \text{ and } a_2^T a_1 = 0.$$

This gives the linear combination of variables that is orthogonal to the first principal component that gives the maximum variation. Further principal components are derived in a similar way.

The vectors $a_1, a_2, \ldots, a_p$, are the eigenvectors of the matrix S and associated with each eigenvector is the eigenvalue, λ_i^2. The value of $\lambda_i^2 / \sum \lambda_i^2$ gives the proportion of variation explained by the ith principal component. Alternatively, the a_i's can be considered as the right singular vectors in a singular value decomposition with singular values λ_i of the data matrix centred about its mean and scaled by $1/\sqrt{(n-1)}$, X_s. This latter approach is used in nag_mv_prin_comp, with

$$X_s = V \Lambda P'$$

where Λ is a diagonal matrix with elements λ_i, P' is the p by p matrix with columns a_i and V is an n by p matrix with $V'V = I$, which gives the principal component scores.

Principal component analysis is often used to reduce the dimension of a data set, replacing a large number of correlated variables with a smaller number of orthogonal variables that still contain most of the information in the original data set.

The choice of the number of dimensions required is usually based on the amount of variation accounted for by the leading principal components. If k principal components are selected, then a test of the equality of the remaining $p-k$ eigenvalues is

$$(n - (2p+5)/6) \left\{ -\sum_{i=k+1}^{p} \log(\lambda_i^2) + (p-k) \log \left(\sum_{i=k+1}^{p} \lambda_i^2 / (p-k) \right) \right\}$$

which has, asymptotically, a χ^2 distribution with $\frac{1}{2}(p-k-1)(p-k+2)$ degrees of freedom.

Equality of the remaining eigenvalues indicates that if any more principal components are to be considered then they all should be considered.

Instead of the variance-covariance matrix the correlation matrix, the sums of squares and cross-products matrix or a standardised sums of squares and cross-products matrix may be used. In the last case S is replaced by $\sigma^{-1/2}S\sigma^{-1/2}$ for a diagonal matrix σ with positive elements. If the correlation matrix is used, the χ^2 approximation for the statistic given above is not valid.

The principal component scores, F, are the values of the principal component variables for the observations. These can be standardised so that the variance of these scores for each principal component is 1.0 or equal to the corresponding eigenvalue.

Weights can be used with the analysis, in which case the matrix X is first centred about the weighted means then each row is scaled by an amount $\sqrt{w_i}$, where w_i is the weight for the ith observation.

4. Parameters

pcmatrix

Input: indicates for which type of matrix the principal component analysis is to be carried out.

If **pcmatrix** = **Nag_MatCorrelation**, then it is for the correlation matrix.

If **pcmatrix** = **Nag_MatStandardised**, then it is for the standardised matrix, with standardisations given by **s**.

If **pcmatrix** = **Nag_MatSumSq**, then it is for the sums of squares and cross-products matrix.

If **pcmatrix** = **Nag_MatVarCovar**, then it is for the variance-covariance matrix.

Constraint: **pcmatrix** = **Nag_MatCorrelation**, **Nag_MatStandardised**, **Nag_MatSumSq** or **Nag_MatVarCovar**.

scores

Input: specifies the type of principal component scores to be used.

If **scores** = **Nag_ScoresStand**, then the principal component scores are standardised so that $F'F = I$, i.e., $F = X_sP\Lambda^{-1} = V$.

If **scores** = **Nag_ScoresNotStand**, then the principal component scores are unstandardised, i.e., $F = X_sP = V\Lambda$.

If **scores** = **Nag_ScoresUnitVar**, then the principal component scores are standardised so that they have unit variance.

If **scores** = **Nag_ScoresEigenval**, then the principal component scores are standardised so that they have variance equal to the corresponding eigenvalue.

Constraint: **scores** = **Nag_ScoresStand**, **Nag_ScoresNotStand**, **Nag_ScoresUnitVar**, or **Nag_ScoresEigenval**.

n

Input: the number of observations, n.
Constraint: **n** ≥ 2.

m

Input: the number of variables in the data matrix, m.
Constraint: **m** ≥ 1.

x[n][tdx]

Input: **x**$[i-1][j-1]$ must contain the ith observation for the jth variable, for $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, m$.

tdx

Input: the last dimension of the array **x** as declared in the calling program.
Constraint: **tdx** $\geq$ **m**.

isx[m]

Input: **isx**$[j-1]$ indicates whether or not the jth variable is to be included in the analysis. If **isx**$[j-1] > 0$, then the variable contained in the jth column of **x** is included in the principal component analysis, for $j = 1, 2, \ldots, m$.

Constraint: **isx**$[j-1] > 0$ for **nvar** values of j.

s[m]

Input: the standardisations to be used, if any.

If **pcmatrix** = **Nag_MatStandardised**, then the first m elements of **s** must contain the standardisation coefficients, the diagonal elements of σ.

Constraint: if **isx**$[j-1] > 0$, then **s**$[j-1] > 0.0$, for $j = 1, 2, \ldots, m$.

Output: if **pcmatrix** = **Nag_MatStandardised**, then **s** is unchanged on exit.
If **pcmatrix** = **Nag_MatCorrelation**, then **s** contains the variances of the selected variables. **s**$[j-1]$ contains the variance of the variable in the jth column of **x** if **isx**$[j-1] > 0$.

If **pcmatrix** = **Nag_MatSumSq** or **Nag_MatVarCovar**, then **s** is not referenced.

wt[n]

Input: the elements of **wt** must contain the weights to be used in the principal component analysis. The effective number of observations is the sum of the weights.

Constraint: **wt**$[i-1] \geq 0.0$, for $i = 1, 2, \ldots, n$ and the sum of weights ≥ **nvar**+1.

If **wt**$[i-1] = 0.0$ then the ith observation is not included in the analysis.

Note: If **wt** is set to the null pointer **NULL**, i.e., (double *)0, then **wt** is not referenced and the effective number of observations is n.

nvar

Input: the number of variables in the principal component analysis, p.

Constraint: 1 ≤ **nvar** ≤ min(**n**−1, **m**).

e[nvar][tde]

Output: the statistics of the principal component analysis.
e$[i-1][0]$, the eigenvalues associated with the ith principal component, λ_i^2, for $i = 1, 2, \ldots, p$.

e$[i-1][1]$, the proportion of variation explained by the ith principal component, for $i = 1, 2, \ldots, p$.

e$[i-1][2]$, the cumulative proportion of variation explained by the first i principal components, for $i = 1, 2, \ldots, p$.

e$[i-1][3]$, the χ^2 statistics, for $i = 1, 2, \ldots, p$.

e$[i-1][4]$, the degrees of freedom for the χ^2 statistics, for $i = 1, 2, \ldots, p$.

If **pcmatrix** ≠ **Nag_MatCorrelation**, then **e**$[i-1][5]$ contains the significance level for the χ^2 statistic, for $i = 1, 2, \ldots, p$.

If **pcmatrix** = **Nag_MatCorrelation**, then **e**$[i-1][5]$ is returned as zero.

tde

Input: the last dimension of the array **e** as declared in the calling program.
Constraint: **tde** ≥ 6.

p[nvar][tdp]

Output: the first **nvar** columns of **p** contain the principal component loadings, a_i. The jth column of **p** contains the **nvar** coefficients for the jth principal component.

tdp

Input: the last dimension of the array **p** as declared in the calling program.
Constraint: **tdp** ≥ **nvar**.

v[n][tdv]

Output: the first **nvar** columns of **v** contain the principal component scores. The jth column of **v** contains the **n** scores for the jth principal component.

If weights are supplied in the array **wt**, then any rows for which **wt**$[i-1]$ is zero will be set to zero.

tdv

Input: the last dimension of the array **v** as declared in the calling program.
Constraint: **tdv** ≥ **nvar**.

fail

The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

NE_BAD_PARAM

On entry, parameter **pcmatrix** had an illegal value.
On entry, parameter **scores** had an illegal value.

NE_INT_ARG_LT

On entry, **m** must not be less than 1: **m** = ⟨*value*⟩.
On entry, **n** must not be less than 2: **n** = ⟨*value*⟩.
On entry, **nvar** must not be less than 1: **nvar** = ⟨*value*⟩.
On entry, **tde** must not be less than 6: **tde** = ⟨*value*⟩.

NE_2_INT_ARG_LT

On entry, **tdx** = ⟨*value*⟩ while **m** = ⟨*value*⟩. These parameters must satisfy **tdx** ≥ **m**.
On entry, **tdv** = ⟨*value*⟩ while **nvar** = ⟨*value*⟩. These parameters must satisfy **tdv** ≥ **nvar**.
On entry, **tdp** = ⟨*value*⟩ while **nvar** = ⟨*value*⟩. These parameters must satisfy **tdp** ≥ **nvar**.

NE_2_INT_ARG_GT

On entry, **nvar** = ⟨*value*⟩ while **m** = ⟨*value*⟩. These parameters must satisfy **nvar** ≤ **m**.

NE_2_INT_ARG_GE

On entry, **nvar** = ⟨*value*⟩ while **n** = ⟨*value*⟩. These parameters must satisfy **nvar** < **n**.

NE_NEG_WEIGHT_ELEMENT

On entry, **wt**[⟨*value*⟩] = ⟨*value*⟩.
Constraint: when referenced, all elements of **wt** must be non-negative.

NE_VAR_INCL_INDICATED

The number of variables, **nvar** in the analysis = ⟨*value*⟩, while the number of variables included in the analysis via array **isx** = ⟨*value*⟩.
Constraint: these two numbers must be the same.

NE_VAR_INCL_STANDARD

On entry, the standardisation element **s**[⟨*value*⟩] = ⟨*value*⟩, while the variable to be included **isx**[⟨*value*⟩] = ⟨*value*⟩.
Constraint: when a variable is to included, the standardisation element must be positive.

NE_OBSERV_LT_VAR

With weighted data, the effective number of observations given by the sum of weights = ⟨*value*⟩, while the number of variables included in the analysis, **nvar** = ⟨*value*⟩.
Constraint: effective number of observations > **nvar** + 1.

NE_SVD_NOT_CONV

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

NE_ZERO_EIGVALS

All eigenvalues/singular values are zero.
This will be caused by all the variables being constant.

NE_ALLOC_FAIL

Memory allocation failed.

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 consult NAG for assistance.

6. Further Comments

6.1. Accuracy

As nag_mv_prin_comp uses a singular value decomposition of the data matrix, it will be less affected by ill-conditioned problems than traditional methods using the eigenvalue decomposition of the variance-covariance matrix.

6.2. References

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

Cooley W C and Lohnes P R (1971) *Multivariate Data Analysis* Wiley.

Hammarling S (1985) The singular value decomposition in multivariate statistics *SIGNUM* **20(3)** 2–25.

Kendall M G and Stuart A (1979) *The Advanced Theory of Statistics (3 Volumes)* Griffin (4th Edition).

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

7. See Also

None.

8. Example

A data set is taken from Cooley and Lohnes (1971), it consists of ten observations on three variables. The unweighted principal components based on the variance-covariance matrix are computed and unstandardised principal component scores requested.

8.1. Program Text

```
/* nag_mv_prin_comp Example Program.
 *
 * Copyright 1998 Numerical Algorithms Group.
 *
 * Mark 5, 1998.
 *
 */
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg03.h>

#define NMAX 12
#define MMAX 3

main()
{
  double p[MMAX][MMAX] ,s[MMAX];
  double e[MMAX][6];
  double v[NMAX][MMAX], x[NMAX][MMAX], wt[NMAX];
  double *wtptr=0;

  Integer isx[MMAX];
  Integer nvar,tdx=MMAX, tde=6, tdp=MMAX, tdv=MMAX;
  Integer i, j, m, n;

  Nag_PrinCompMat pcmatrix;
  Nag_PrinCompScores scores;

  char weight[2], matrix[2], std[2];

  Vprintf("g03aac Example Program Results\n\n");

  /*  Skip heading in data file */
  Vscanf("%*[^\n]");

  Vscanf("%s",matrix);
  Vscanf("%s",std);
  Vscanf("%s",weight);
  Vscanf("%ld",&n);
  Vscanf("%ld",&m);

  if (*matrix == 'C')
    pcmatrix = Nag_MatCorrelation;
  else if (*matrix == 'S')
    pcmatrix = Nag_MatStandardised;
```

```
else if (*matrix == 'U')
  pcmatrix =  Nag_MatSumSq;
else
  pcmatrix =  Nag_MatVarCovar;

if (*std == 'S')
  scores = Nag_ScoresStand;
else if (*std == 'U')
  scores = Nag_ScoresNotStand;
else if (*std == 'Z')
  scores = Nag_ScoresUnitVar;
else
  scores = Nag_ScoresEigenval;

if (n <= NMAX && m <= MMAX)
  {
    if (*weight == 'U')
      {
        for (i = 0; i < n; ++i)
          {
            for (j = 0; j < m; ++j)
              Vscanf("%lf",&x[i][j]);
          }
      }
    else
      {
        for (i = 0; i < n; ++i)
          {
            for (j = 0; j < m; ++j)
              Vscanf("%lf",&x[i][j]);
            Vscanf("%lf",&wt[i]);
          }
        wtptr = wt;
      }
    for (j = 0; j < m; ++j)
      {
        Vscanf("%ld",&isx[j]);
      }
    Vscanf("%ld",&nvar);
    if (pcmatrix == Nag_MatStandardised)
      {
        for (j = 0; j < m; ++j)
          Vscanf("%lf",&s[j]);
      }

    g03aac(pcmatrix, scores, n, m, (double *)x, tdx, isx, s, wtptr, nvar,
           (double *)e, tde, (double *)p, tdp, (double *)v, tdv, NAGERR_DEFAULT);

    Vprintf("Eigenvalues  Percentage  Cumulative    Chisq      DF     Sig\n");
    Vprintf("               variation   variation\n\n");
    for (i = 0; i < nvar; ++i)
      {
        for (j = 0; j < 6; ++j)
          Vprintf("%11.4f",e[i][j]);
        Vprintf("\n");
      }
    Vprintf("\nEigenvalues \n\n");
    for (i = 0; i < nvar; ++i)
      {
        for (j = 0; j < nvar; ++j)
          Vprintf("%9.4f",p[i][j]);
        Vprintf("\n");
      }
    Vprintf("\nPrincipal component scores \n\n");
    for (i = 0; i < n; ++i)
      {
        Vprintf("%2ld", i+1);
        for (j = 0; j < nvar; ++j)
          Vprintf("%9.3f", v[i][j]);
        Vprintf("\n");
```

```
          }
        exit(EXIT_SUCCESS);
      }
    else
      {
        Vprintf("Incorrect input value of n or m.\n");
        exit(EXIT_FAILURE);
      }
}
```

8.2. Program Data

```
g03aac Example Program Data
V E U 10 3
7.0 4.0 3.0
4.0 1.0 8.0
6.0 3.0 5.0
8.0 6.0 1.0
8.0 5.0 7.0
7.0 2.0 9.0
5.0 3.0 3.0
9.0 5.0 8.0
7.0 4.0 5.0
8.0 2.0 2.0
 1   1   1   3
```

8.3. Program Results

```
g03aac Example Program Results

Eigenvalues  Percentage  Cumulative     Chisq      DF     Sig
               variation   variation

     8.2739     0.6515     0.6515     8.6127     5.0000     0.1255
     3.6761     0.2895     0.9410     4.1183     2.0000     0.1276
     0.7499     0.0590     1.0000     0.0000     0.0000     0.0000

Eigenvalues

   0.1376   0.6990   0.7017
   0.2505   0.6609  -0.7075
  -0.9583   0.2731  -0.0842

Principal component scores

 1    2.151   -0.173   -0.107
 2   -3.804   -2.887   -0.510
 3   -0.153   -0.987   -0.269
 4    4.707    1.302   -0.652
 5   -1.294    2.279   -0.449
 6   -4.099    0.144    0.803
 7    1.626   -2.232   -0.803
 8   -2.114    3.251    0.168
 9    0.235    0.373   -0.275
10    2.746   -1.069    2.094
```