NAG Library Function Document

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

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

void nag_mv_prin_comp (Nag_PrinCompMat pcmatrix, Nag_PrinCompScores scores,
                        Integer n, Integer m, const double x[], Integer tdx,
                        const Integer isx[], double s[], const 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:

\[
a_1^T S a_1
\]

is maximized subject to

\[
a_1^T a_1 = 1.
\]

The variable \( z_1 = \sum_{i=1}^{p} a_1_i x_i \) is known as the first principal component and gives the linear combination of the variables that gives the maximum variation. A second principal component, \( z_2 = \sum_{i=1}^{p} a_2_i x_i \), is found such that:

\[
a_2^T S a_2
\]

is maximized subject to

\[
a_2^T a_2 = 1
\]

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, \( \lambda_i^2 \). The value of \( \lambda_i^2 / \sum \lambda_i^2 \) gives the proportion of variation explained by the \( i \)th principal component. Alternatively, the \( a_i \)'s can be considered as the right singular vectors in a singular value decomposition with singular values \( \lambda_i \) 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 (g03aac), with

\[
X_s = V A P'
\]

where \( A \) is a diagonal matrix with elements \( \lambda_i \), \( P' \) is the \( p \) by \( p \) matrix with columns \( a_i \) and \( V \) is an \( n \) by \( p \) matrix with \( V^T V = I \), which gives the principal component scores.

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

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

which has, asymptotically, a \( \chi^2 \) 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 standardized 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 \( \sigma \) with positive elements. If the correlation matrix is used, the \( \chi^2 \) 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 standardized 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 \( i \)th observation.

4 References

5 Arguments
1: \textbf{pcmatrix} – Nag_PrinCompMat

*Input*

\textit{On entry:} indicates for which type of matrix the principal component analysis is to be carried out.

\texttt{pcmatrix} = Nag_MatCorrelation

It is for the correlation matrix.

\texttt{pcmatrix} = Nag_MatStandardised

It is for the standardized matrix, with standardizations given by \( s \).

\texttt{pcmatrix} = Nag_MatSumSq

It is for the sums of squares and cross-products matrix.

\texttt{pcmatrix} = Nag_MatVarCovar

It is for the variance-covariance matrix.

\textit{Constraint:} \texttt{pcmatrix} = Nag_MatCorrelation, Nag_MatStandardised, Nag_MatSumSq or Nag_MatVarCovar.

2: \textbf{scores} – Nag_PrinCompScores

*Input*

\textit{On entry:} specifies the type of principal component scores to be used.

\texttt{scores} = Nag_ScoresStand

The principal component scores are standardized so that \( F^T F = I \), i.e., \( F = X_s P A^{-1} = V \).
scores = Nag_ScoresNotStand
The principal component scores are unstandardized, i.e., \( F = X, P = VA \).

scores = Nag_ScoresUnitVar
The principal component scores are standardized so that they have unit variance.

scores = Nag_ScoresEigenval
The principal component scores are standardized so that they have variance equal to the corresponding eigenvalue.

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

3: n – Integer
   Input
   On entry: the number of observations, \( n \).
   Constraint: \( n \geq 2 \).

4: m – Integer
   Input
   On entry: the number of variables in the data matrix, \( m \).
   Constraint: \( m \geq 1 \).

5: x[n x tdx] – const double
   Input
   On entry: \( x[(i-1) \times tdx + j - 1] \) must contain the \( i \)th observation for the \( j \)th variable, for \( i = 1, 2, \ldots, n \) and \( j = 1, 2, \ldots, m \).

6: tdx – Integer
   Input
   On entry: the stride separating matrix column elements in the array x.
   Constraint: \( tdx \geq m \).

7: isx[m] – const Integer
   Input
   On entry: \( isx[j-1] \) indicates whether or not the \( j \)th variable is to be included in the analysis. If \( isx[j-1] > 0 \), then the variable contained in the \( j \)th 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 \( j \).

8: s[m] – double
   Input/Output
   On entry: the standardizations to be used, if any.
   If \( pcmatrix = \text{Nag} \_\text{MatStandardised} \), then the first \( m \) elements of \( s \) must contain the standardization coefficients, the diagonal elements of \( \sigma \).
   Constraint: if \( isx[j-1] > 0 \), \( s[j-1] > 0.0 \), for \( j = 1, 2, \ldots, m \).
   On exit: if \( pcmatrix = \text{Nag} \_\text{MatStandardised} \), then \( s \) is unchanged on exit.
   If \( pcmatrix = \text{Nag} \_\text{MatCorrelation} \), then \( s \) contains the variances of the selected variables. \( s[j-1] \) contains the variance of the variable in the \( j \)th column of \( x \) if \( isx[j-1] > 0 \).
   If \( pcmatrix = \text{Nag} \_\text{MatSumSq} \) or \( \text{Nag} \_\text{MatVarCovar} \), then \( s \) is not referenced.

9: wt[n] – const double
   Input
   On entry: optionally, the weights to be used in the principal component analysis.
   If \( wt[i-1] = 0.0 \), then the \( i \)th observation is not included in the analysis. The effective number of observations is the sum of the weights.
If weights are not provided then \( wt \) must be set to \text{NULL} and the effective number of observations is \( n \).

**Constraints:**

- if \( wt \) is not \text{NULL}, \( wt[i - 1] \geq 0.0 \), for \( i = 1, 2, \ldots, n \);  
- if \( wt \) is not \text{NULL}, the sum of weights \( \geq nvar + 1 \).

10: \text{nvar} – Integer  
   \text{Input}  
   On entry: the number of variables in the principal component analysis, \( p \).  
   \text{Constraint}: \( 1 \leq nvar \leq \min(n - 1, m) \).

11: \( e[nvar \times tde] \) – double  
   \text{Output}  
   On exit: the statistics of the principal component analysis.  
   - \( e[(i - 1) \times tde] \), the eigenvalues associated with the \( i \)th principal component, \( \lambda_i^2 \), for \( i = 1, 2, \ldots, p \).  
   - \( e[(i - 1) \times tde + 1] \), the proportion of variation explained by the \( i \)th principal component, for \( i = 1, 2, \ldots, p \).  
   - \( e[(i - 1) \times tde + 2] \), the cumulative proportion of variation explained by the first \( i \) principal components, for \( i = 1, 2, \ldots, p \).  
   - \( e[(i - 1) \times tde + 3] \), the \( \chi^2 \) statistics, for \( i = 1, 2, \ldots, p \).  
   - \( e[(i - 1) \times tde + 4] \), the degrees of freedom for the \( \chi^2 \) statistics, for \( i = 1, 2, \ldots, p \).  
   - If \( pc\text{matrix} \neq \text{Nag_MatCorrelation} \), then \( e[(i - 1) \times tde + 5] \) contains the significance level for the \( \chi^2 \) statistic, for \( i = 1, 2, \ldots, p \).  
   - If \( pc\text{matrix} = \text{Nag_MatCorrelation} \), then \( e[(i - 1) \times tde + 5] \) is returned as zero.

12: \( tde \) – Integer  
   \text{Input}  
   On entry: the stride separating matrix column elements in the array \( e \).  
   \text{Constraint}: \( tde \geq 6 \).

13: \( p[nvar \times tdp] \) – double  
   \text{Output}  
   Note: the \( (i, j) \)th element of the matrix \( P \) is stored in \( p[(i - 1) \times tdp + j - 1] \).  
   On exit: the first \( nvar \) columns of \( p \) contain the principal component loadings, \( a_i \).  
   The \( j \)th column of \( p \) contains the \( nvar \) coefficients for the \( j \)th principal component.

14: \( tdp \) – Integer  
   \text{Input}  
   On entry: the stride separating matrix column elements in the array \( p \).  
   \text{Constraint}: \( tdp \geq nvar \).

15: \( v[n \times tdv] \) – double  
   \text{Output}  
   Note: the \( (i, j) \)th element of the matrix \( V \) is stored in \( v[(i - 1) \times tdv + j - 1] \).  
   On exit: the first \( nvar \) columns of \( v \) contain the principal component scores.  
   The \( j \)th column of \( v \) contains the \( n \) scores for the \( j \)th 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.

16: \( tdv \) – Integer  
   \text{Input}  
   On entry: the stride separating matrix column elements in the array \( v \).  
   \text{Constraint}: \( tdv \geq nvar \).
6 Error Indicators and Warnings

**NE_2_INT_ARG_GE**

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

**NE_2_INT_ARG_GT**

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

**NE_2_INT_ARG_LT**

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

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

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

**NE_ALLOC_FAIL**

Dynamic memory allocation failed.

**NE_BAD_PARAM**

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

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

**NE_INT_ARG_LT**

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

Constraint: \( m \geq 1 \).

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

Constraint: \( n \geq 2 \).

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

Constraint: \( nvar \geq 1 \).

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

Constraint: \( tde \geq 6 \).

**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_NEG_WEIGHT_ELEMENT**

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

Constraint: when referenced, all elements of \( \text{wt} \) must be non-negative.

**NE_OBSERV_LT_VAR**

With weighted data, the effective number of observations given by the sum of weights \( = \langle \text{value} \rangle \), while the number of variables included in the analysis, \( nvar = \langle \text{value} \rangle \).

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.
The number of variables, \(nvar\) in the analysis = \(\text{value}\), while the number of variables included in the analysis via array \(\text{isx}\) = \(\text{value}\).
Constraint: these two numbers must be the same.

On entry, the standardization element \(s[\text{value}] = \text{value}\), while the variable to be included \(\text{isx}[\text{value}] = \text{value}\).
Constraint: when a variable is to be included, the standardization element must be positive.

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

As \text{nag_mv_prin_comp (g03aac)} 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.

Not applicable.

None.

A dataset 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 unstandardized principal component scores requested.

/* \text{nag_mv_prin_comp (g03aac)} Example Program. */
* Copyright 2014 Numerical Algorithms Group.
*/
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg03.h>
#define X(I, J) x[(I) *tdx + J]
#define P(I, J) p[(I) *tdp + J]
#define E(I, J) e[(I) *tde + J]
#define V(I, J) v[(I) *tdv + J]
int main(void)
{
    Integer exit_status = 0, i, *isx = 0, j, m, n, nvar, tde = 6, tdp,
    tdv, tdx;
    Nag_PrinCompMat matrix;
    Nag_PrinCompScores scores;
    Nag_Boolean weight;

char nag_enum_arg[40];
double *e = 0, *p = 0, *s = 0, *v = 0, *wt = 0, *wt ptr = 0;
double *x = 0;
NagError fail;

INIT_FAIL(fail);

printf("nag_mv_prin_comp (g03aac) Example Program Results\n\n");

/\* Skip heading in data file */
#ifdef _WIN32
scanf_s("%*[\n"]);
#else
scanf("%*[\n"]);
#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
*/
matrix = (Nag_PrinCompMat) nag_enum_name_to_value(nag_enum_arg);
#ifdef _WIN32
scanf_s("%39s", nag_enum_arg, _countof(nag_enum_arg));
#else
scanf("%39s", nag_enum_arg);
#endif
scores = (Nag_PrinCompScores) nag_enum_name_to_value(nag_enum_arg);
#ifdef _WIN32
scanf_s("%39s", nag_enum_arg, _countof(nag_enum_arg));
#else
scanf("%39s", nag_enum_arg);
#endif
weight = (Nag_Boolean) nag_enum_name_to_value(nag_enum_arg);
#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

if (n >= 2 && m >= 1)
{
    if (!(x = NAG_ALLOC((n)*(m), double)) ||
        !(wt = NAG_ALLOC(n, double)) ||
        !(s = NAG_ALLOC(m, double)) ||
        !(isx = NAG_ALLOC(m, Integer)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }
    tdx = m;
}
else
{
    printf("Invalid n or m.\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
#else
    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("%lf", &wt[i]);
            #else
                scanf("%lf", &wt[i]);
            #endif
        }
    } wtptr = wt;
    for (j = 0; j < m; ++j)
    { #ifdef _WIN32
        scanf_s("%NAG_IFMT", &isx[j]);
    #else
        scanf("%NAG_IFMT", &isx[j]);
    #endif
    #ifdef _WIN32
        scanf_s("%NAG_IFMT", &nvar);
    #else
        scanf("%NAG_IFMT", &nvar);
    #endif
    if (nvar >= 1 && nvar <= MIN(n-1, m))
    { if (!p = NAG_ALLOC(nvar*nvar, double)) ||
        !e = NAG_ALLOC(nvar*6, double)) ||
        !v = NAG_ALLOC(n*nvar, double))
        {
            printf("Allocation failure\n");
            exit_status = -1;
            goto END;
        }
    } tdp = nvar;
    tde = 6;
    tdv = nvar;
} else
{ printf("Invalid nvar.\n");
    exit_status = 1;
    goto END;
}
if (matrix == Nag_MatStandardised)
{ for (j = 0; j < m; ++j)
    #ifdef _WIN32
        scanf_s("%lf", &s[j]);
    #else
        scanf("%lf", &s[j]);
    #endif
} /* nag_mv_prin_comp (g03aac).
* Principal component analysis
*/
nag_mv_prin_comp(matrix, scores, n, m, x, tdx, isx, s, wtptr, nvar, 
e, tde, p, tdp, v, tdv, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_mv_prin_comp (g03aac).\n", fail.message);
    exit_status = 1;
    goto END;
}
printf(  
"Eigenvalues Percentage Cumulative Chisq DF Sig\n"
);  
printf(" variation variation\n"");
for (i = 0; i < nvar; ++i)
{
    for (j = 0; j < 6; ++j)
        printf("%11.4f", E(i, j));
    printf("\n");
}
printf("nPrincipal component loadings \n\n");
for (i = 0; i < nvar; ++i)
{
    for (j = 0; j < nvar; ++j)
        printf("%9.4f", P(i, j));
    printf("\n");
}
printf("nPrincipal component scores \n\n");
for (i = 0; i < n; ++i)
{
    printf("%2"NAG_IFMT", i+1);
    for (j = 0; j < nvar; ++j)
        printf("%9.3f", V(i, j));
    printf("\n");
}

END:
NAG_FREE(x);
NAG_FREE(wt);
NAG_FREE(s);
NAG_FREE(isx);
NAG_FREE(p);
NAG_FREE(e);
NAG_FREE(v);

return exit_status;
}

10.2 Program Data

g03 – Multivariate Methods

nag_mv_prin_comp (g03aac) Example Program Data
Nag_MatVarCovar Nag_ScoresEigenval Nag_FALSE

7.0 4.0 3.0
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
10.3 Program Results

nag_mv_prin_comp (g03aac) Example Program Results

<table>
<thead>
<tr>
<th>Eigenvalues</th>
<th>Percentage variation</th>
<th>Cumulative variation</th>
<th>Chisq</th>
<th>DF</th>
<th>Sig</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.2739</td>
<td>0.6515</td>
<td>0.6515</td>
<td>8.6127</td>
<td>5.0000</td>
<td>0.1255</td>
</tr>
<tr>
<td>3.6761</td>
<td>0.2895</td>
<td>0.9410</td>
<td>4.1183</td>
<td>2.0000</td>
<td>0.1276</td>
</tr>
<tr>
<td>0.7499</td>
<td>0.0590</td>
<td>1.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

Principal component loadings

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1376</td>
<td>0.6990</td>
<td>0.7017</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2505</td>
<td>0.6609</td>
<td>-0.7075</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.9583</td>
<td>0.2731</td>
<td>-0.0842</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Principal component scores

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.151</td>
<td>-0.173</td>
<td>-0.107</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>-3.804</td>
<td>-2.887</td>
<td>-0.510</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>-0.153</td>
<td>-0.987</td>
<td>-0.269</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4.707</td>
<td>1.302</td>
<td>-0.652</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-1.294</td>
<td>2.279</td>
<td>-0.449</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>-4.099</td>
<td>0.144</td>
<td>0.803</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1.626</td>
<td>-2.232</td>
<td>-0.803</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>-2.114</td>
<td>3.251</td>
<td>0.168</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.235</td>
<td>0.373</td>
<td>-0.275</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>2.746</td>
<td>-1.069</td>
<td>2.094</td>
<td></td>
</tr>
</tbody>
</table>