Package ‘mifa’

June 28, 2017

Type Package

Title Multiple Imputation for Exploratory Factor Analysis

Version 0.1.0

Author Vahid Nassiri,
Anikó Lovik,
Geert Molenberghs,
Geert Verbeke

Maintainer Vahid Nassiri <vahid.nassiri@kuleuven.be>

Imports mice, Matrix

Suggests eigeninv, norm, psych, missMDA

Description This package uses multiple imputation to estimate the covariance matrix of incomplete data. An exploratory factor analysis then can be applied on this estimated covariance matrix. It also provides Fieller and bootstrap confidence intervals for the proportion of explained variance using different number of factors.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

R topics documented:

`ci.mi.each` *ci.mi.each*

Description

This function is used inside ci.mifa.fieller to compute the Fieller's interval for each of specified number of factors.

Usage

```
ci.mi.each(eig.imp, n.factor, alpha, N, M)
```

Arguments

`eig.imp`	A matrix with each of its columns the eigenvalues of the estimated covariance matrix for each imputed data.
`n.factor`	A scalar specifying the number of factors.
`alpha`	The significance level for constructing confidence intervals.
`N`	A scalar specifying sample size
`M`	A scalar specifying number of multiple imputations.

Value

A vector of length 2, containing the lower and upper bouds of estimated Fieller's interval.

Note

Note that one can directly use ci.mifa.fieller. This function will be called in there for internal computations.

Author(s)

Vahid Nassiri, Anikó Lovik, Geert Molenberghs, Geert Verbeke.

See Also

`ci.mifa.fieller`

`ci.mifa.bootstrap` *ci.mifa.bootstrap*

Description

This function computes a bootstrap confidence interval for the proportion of explaiend variance for given numbers of factors for the incomplete data using multiple imputation.

Usage

```
ci.mifa.bootstrap(data.miss, n.factor, rep.boot = 1000, method.mi, maxit.mi, alpha)
```

Arguments

`data.miss`	The incomplete dataset, a matrix with the items as its columns subject as its rows. The missing values should be shown with NA.
`n.factor`	A vector containing number of factors which should be used to compute proportion of explained variance or construct confidence intervals.
`rep.boot`	A scalar specifying the number of bootstrap sub-samples to construct the confidence interval. The default is 1000.
`method.mi`	The imputation method, it can be a string or a vector of strings of the size equal to number of items. For more information see R documentations for mice package. The default is set as 'pmm', i.e., predictive mean matching.
`maxit.mi`	A scalar specifying the number of iterations for each imputation. For more information see R documentations for mice package. The default is 5.
`alpha`	The significance level for constructing confidence intervals.

Details

This function uses the Shao and Sitter (1996) method to combine multiple imputation and bootstrapping. The imputations are done using package mice.

Value

A matrix with three columns, the first column shows the number of factors, and the two other columns show the lower and upper bounds of the estimated bootstrap confidence interval for proportion of explained variance.

Note

Note that by setting ci=TRUE in mifa.cov, this confidence interval can be computed as well.

Author(s)

Vahid Nassiri, Anikó Lovik, Geert Molenberghs, Geert Verbeke.

References

Shao, Jun, and Randy R. Sitter. "Bootstrap for imputed survey data." Journal of the American Statistical Association 91.435 (1996): 1278-1288.

See Also

`mifa.cov`

Examples

```
# Generating incomplete data
# defining the vector of eigenvalues
e.vals=c(50,48,45,25,20,10,5,5,1,1,0.5,0.5,0.5,0.1,0.1)
# loading eigeninv package to generate a covariance matrix with the
# eigenvalues in e.vals, if this package is not installed, one needs
# to install it first using install.packages("eigeninv")
require(eigeninv)
library(eigeninv)
# Defining the sample size, N, and number of items, P.
```

```
P = length(e.vals)
N = 100
# Generate a set of centered indepdent normal data
data.ini1 = matrix(rnorm(N*P),N,P)
mean.data.ini = apply(data.ini1,2,mean)
data.ini = t(t(data.ini1)-mean.data.ini)
# Finding the Cholesky decomposition of the cov.mat
chol.cov=t(chol(cov.mat))
# Using col.cov to generate multivariate normal data
# with the given covariance matrix.
data=matrix(0,N,P)
for (i in 1:N){
  data[i,]=chol.cov
}
# Here we create 5-percent missing data with
# missing completely at random mechanism
data.miss=data
mcar.n.miss=0.05
for (i in 1:P){
  for (j in 1:N){
    rand.u=runif(1)
    if (rand.u<=mcar.n.miss){
      data.miss[j,i]=NA
    }
  }
}
ci.mifa.bootstrap(data.miss,n.factor=1:10,rep.boot=100,method.mi='pmm',maxit.mi=5,alpha=0.05)
      n.factor      2.5
 [1,]        1 0.2503306 0.3466675
 [2,]        2 0.4714291 0.5687062
 [3,]        3 0.6500003 0.7333300
 [4,]        4 0.7889355 0.8430495
 [5,]        5 0.8769695 0.9115910
 [6,]        6 0.9309796 0.9523302
 [7,]        7 0.9612759 0.9750707
 [8,]        8 0.9847418 0.9892099
 [9,]        9 0.9900835 0.9928978
[10,]       10 0.9936964 0.9956437
```

`ci.mifa.fieller` *ci.mifa.fieller*

Description

This function computes Fieller's confidence interval for the proportion of explained variance for given numbers of factors.

Usage

```
ci.mifa.fieller(cov.mi, n.factor, alpha, N)
```

Arguments

`cov.mi`	A list containing the estimated covariance matrix for each imputed data. One can use the outcome of 'mi.cov' with the name 'cov.mice.imp'.

`n.factor`	A vector containing the numbers of factors that should be used to compute the proportion of explained variance or construct confidence intervals. The minimum length of this vector is 1 and its maximum length is the number of items.
`alpha`	The significance level for constructing confidence intervals.
`N`	An scalar specifying the sample size.

Value

A matrix with three columns, the first column shows the number of factor, and the two other columns show the lower and uppor bound of the estimated Fieller's confidence interval for proportion of explained variance.

Note

Note that by setting ci=TRUE in mifa.cov, this confidence interval can be computed as well.

Author(s)

Vahid Nassiri, Anikó Lovik, Geert Molenberghs, Geert Verbeke.

References

Fieller, Edgar C. "Some problems in interval estimation." Journal of the Royal Statistical Society. Series B (Methodological) (1954): 175-185.

See Also

`mifa.cov`

Examples

```
# Generating incomplete data
# defining the vector of eigenvalues
e.vals=c(50,48,45,25,20,10,5,5,1,1,0.5,0.5,0.5,0.1,0.1)
# loading eigeninv package to generate a covariance matrix with the
# eigenvalues in e.vals, if this package is not installed, one needs
# to install it first using install.packages("eigeninv")
require(eigeninv)
library(eigeninv)
# Defining the sample size, N, and number of items, P.
P = length(e.vals)
N = 100
# Generate a set of centered indepdent normal data
data.ini1 = matrix(rnorm(N*P),N,P)
mean.data.ini = apply(data.ini1,2,mean)
data.ini = t(t(data.ini1)-mean.data.ini)
# Finding the Cholesky decomposition of the cov.mat
chol.cov=t(chol(cov.mat))
# Using col.cov to generate multivariate normal data
# with the given covariance matrix.
data=matrix(0,N,P)
for (i in 1:N){
  data[i,]=chol.cov
}
# Here we create 5-percent missing data with
# missing completely at random mechanism
data.miss=data
mcar.n.miss=0.05
for (i in 1:P){
  for (j in 1:N){
    rand.u=runif(1)
    if (rand.u<=mcar.n.miss){
      data.miss[j,i]=NA
    }
  }
}
result.mi=mifa.cov (data.miss,n.factor=1:10,M=10,maxit.mi = 5,method.mi='pmm',
                  alpha = 0.05,rep.boot=500,ci=FALSE)
ci.mifa.fieller (result.mi$cov.mice.imp,1:10,0.05,N=100)
      n.factor     Lower     Upper
 [1,]        1 0.2190723 0.3496561
 [2,]        2 0.4317744 0.5553530
 [3,]        3 0.6440130 0.7350537
 [4,]        4 0.7576282 0.8270035
 [5,]        5 0.8511427 0.8970771
 [6,]        6 0.9188874 0.9448734
 [7,]        7 0.9540987 0.9698366
 [8,]        8 0.9788137 0.9851842
 [9,]        9 0.9852719 0.9899236
[10,]       10 0.9906949 0.9937464
```

---

combine.mi *combine.mi*

Description

This function applies Rubin's rules (Rubin, 2004) to combine eastimates and variance-covariance matrices from different imputations.

Usage

```
combine.mi(mi.parm.est, mi.parm.cov)
```

Arguments

mi.parm.est	A matrix containing estimated parameters in each imputation as its rows.
mi.parm.cov	A list contaning the covariance matrix estimated within each imputation.

Value

parm.est	Combined estimates.
parm.cov	Combined variance-covariance matrix.
between.cov	Between imputations variance-covariance matrix.

Author(s)

Vahid Nassiri, Anikó Lovik, Geert Molenberghs, Geert Verbeke.

References

Rubin D. B. Multiple imputation for nonresponse in surveys. John Wiley & Sons; 2004.

`mifa.cov` *mifa.cov*

Description

This function estimates the covariance matrix of an incomplete dataset using multiple imputation.

Usage

```
mifa.cov(data.miss,n.factor,M,maxit.mi = 5,method.mi='pmm',alpha = 0.05,rep.boot=NULL,ci=FALSE)
```

Arguments

`data.miss`	The incomplete dataset, a matrix with the items as its columns subject as its rows. The missing values should be shown with NA.
`n.factor`	A vector containing numbers of factors which should be used to compute proportion of explained variance or construct confidence intervals.
`M`	A scalar specifying number of multiple imputations.
`maxit.mi`	A scalar specifying number of iterations for each imputation. For more information see R documentation for mice package. The default is 5.
`method.mi`	The imputation method, it can be a string or a vector of strings of the size equal to number of items. For more information see R documentation for mice package. The default is set as 'pmm', i.e., predictive mean matching.
`alpha`	The significance level for constructing confidence intervals. The dafault if 0.05.
`rep.boot`	A scalar specifying number of bootstrap sub-samples to construct the confidence interval. If ci=TRUE rep.boot should be specified.
`ci`	A logical variable indicating whether confidence intervals should be constructed for proportion of explianed variance or not. The default value is FALSE.

Details

Note that one needs to install the package 'mice' before using this function. This can be done use the command: install.packages("mice")

Value

`cov.mice`	The estimatied covariance matrix of the incomplete data using multiple imputations.
`cov.mice.imp`	A list containing th estimated covariance matrix for each of M imputed data.
`exp.var.mice`	A vector containing the estimated proportions of explained variance for each of specified n.factor components.
`ci.mice.fieller`	A matrix containing the estimated Fieller's confidence interval for proportion of explained variance for each of specified n.factor components.
`ci.mice.bootstrap`	A matrix containing the estimated bootstrap confidence interval for proportion of explained variance for each of specified n.factor components.

Author(s)

Vahid Nassiri, Anikó Lovik, Geert Molenberghs, Geert Verbeke.

References

The mice documentation: https://cran.r-project.org/web/packages/mice/mice.pdf

Examples

```
# Generating incomplete data
# defining the vector of eigenvalues
e.vals=c(50,48,45,25,20,10,5,5,1,1,0.5,0.5,0.5,0.1,0.1)
# loading eigeninv package to generate a covariance matrix with the
# eigenvalues in e.vals, if this package is not installed, one needs
# to install it first using install.packages("eigeninv")
require(eigeninv)
library(eigeninv)
cov.mat = eiginv(evals=e.vals, symmetric=TRUE)
# Defining the sample size, N, and number of items, P.
P = length(e.vals)
N = 100
# Generate a set of centered indepdent normal data
data.ini1 = matrix(rnorm(N*P),N,P)
mean.data.ini = apply(data.ini1,2,mean)
data.ini = t(t(data.ini1)-mean.data.ini)
# Finding the Cholesky decomposition of the cov.mat
chol.cov=t(chol(cov.mat))
# Using col.cov to generate multivariate normal data
# with the given covariance matrix.
data=matrix(0,N,P)
for (i in 1:N){
  data[i,]=chol.cov
}
# Here we create 5-percent missing data with
# missing completely at random mechanism
data.miss=data
mcar.n.miss=0.05
for (i in 1:P){
  for (j in 1:N){
    rand.u=runif(1)
    if (rand.u<=mcar.n.miss){
      data.miss[j,i]=NA
    }
  }
}
result.mi=mifa.cov (data.miss,n.factor=1:10,M=10,maxit.mi = 5,method.mi='pmm',
                  alpha = 0.05,rep.boot=500,ci=TRUE)
```

Index