Package ‘nipals’

January 24, 2020

Title Principal Components Analysis using NIPALS or Weighted EMPCA, with Gram-Schmidt Orthogonalization

Version 0.7

Date 2020-01-21


Depends R (>= 3.4.0)

License GPL-3

Encoding UTF-8

LazyData true

Suggests knitr, rmarkdown, testthat

URL http://kwstat.github.io/nipals/

BugReports https://github.com/kwstat/nipals/issues

VignetteBuilder knitr

RoxygenNote 7.0.2

NeedsCompilation no

Author Kevin Wright [aut, cre] (<https://orcid.org/0000-0002-0617-8673>)

Maintainer Kevin Wright <kw.stat@gmail.com>

Repository CRAN

Date/Publication 2020-01-24 06:50:10 UTC

R topics documented:

  empca ................................................................. 2
  nipals ................................................................. 3
  uscrime .............................................................. 5

Index 7
Principal component analysis by weighted EMPCA, expectation maximization principal component-analysis

**Description**

Used for finding principal components of a numeric matrix. Missing values in the matrix are allowed. Weights for each element of the matrix are allowed. Principal Components are extracted one a time. The algorithm computes \( x = TP' \), where \( T \) is the 'scores' matrix and \( P \) is the 'loadings' matrix.

**Usage**

```r
empca(
  x,
  w,
  ncomp = min(nrow(x), ncol(x)),
  center = TRUE,
  scale = TRUE,
  maxiter = 100,
  tol = 1e-06,
  seed = NULL,
  fitted = FALSE,
  gramschmidt = TRUE,
  verbose = FALSE
)
```

**Arguments**

- **x**: Numerical matrix for which to find principal components. Missing values are allowed.
- **w**: Numerical matrix of weights.
- **ncomp**: Maximum number of principal components to extract from \( x \).
- **center**: If TRUE, subtract the mean from each column of \( x \) before PCA.
- **scale**: if TRUE, divide the standard deviation from each column of \( x \) before PCA.
- **maxiter**: Maximum number of EM iterations for each principal component.
- **tol**: Default 1e-6 tolerance for testing convergence of the EM iterations for each principal component.
- **seed**: Random seed to use when initializing the random rotation matrix.
- **fitted**: Default FALSE. If TRUE, return the fitted (reconstructed) value of \( x \).
- **gramschmidt**: Default TRUE. If TRUE, perform Gram-Schmidt orthogonalization at each iteration.
- **verbose**: Default FALSE. Use TRUE or 1 to show some diagnostics.
Value

A list with components eig, scores, loadings, fitted, ncomp, R2, iter, center, scale.

Author(s)

Kevin Wright

References


Examples

```r
B <- matrix(c(50, 67, 90, 98, 120, 
55, 71, 93, 102, 129, 
65, 76, 95, 105, 134, 
50, 80, 102, 130, 138, 
60, 82, 97, 135, 151, 
65, 89, 106, 137, 153, 
75, 95, 117, 133, 155), ncol=5, byrow=TRUE)
rownames(B) <- c("G1","G2","G3","G4","G5","G6","G7")
colnames(B) <- c("E1","E2","E3","E4","E5")
dim(B) # 7 x 5
p1 <- empca(B)
dim(p1$scores) # 7 x 5
dim(p1$loadings) # 5 x 5

B2 = B
p2 = empca(B2, fitted=TRUE)
```

Description

Used for finding principal components of a numeric matrix. Missing values in the matrix are allowed. Principal Components are extracted one at a time. The algorithm computes \( x = TP' \), where \( T \) is the 'scores' matrix and \( P \) is the 'loadings' matrix.

Usage

```r
nipals(
  x,
  ncomp = min(nrow(x), ncol(x))
)
```
center = TRUE,
scale = TRUE,
maxiter = 500,
tol = 1e-06,
startcol = 0,
fitted = FALSE,
force.na = FALSE,
gramschmidt = TRUE,
verbose = FALSE
)

Arguments

x        Numerical matrix for which to find principal components. Missing values are allowed.
ncomp    Maximum number of principal components to extract from x.
center   If TRUE, subtract the mean from each column of x before PCA.
scale    If TRUE, divide the standard deviation from each column of x before PCA.
maxiter  Maximum number of NIPALS iterations for each principal component.
tol      Default 1e-9 tolerance for testing convergence of the NIPALS iterations for each principal component.
startcol Determine the starting column of x for the iterations of each principal component. If 0, use the column of x that has maximum absolute sum. If a number, use that column of x. If a function, apply the function to each column of x and choose the column with the maximum value of the function.
fitted   Default FALSE. If TRUE, return the fitted (reconstructed) value of x.
force.na Default FALSE. If TRUE, force the function to use the method for missing values, even if there are no missing values in x.
gramschmidt Default TRUE. If TRUE, perform Gram-Schmidt orthogonalization at each iteration.
verbose  Default FALSE. Use TRUE or 1 to show some diagnostics.

Details

The R2 values that are reported are marginal, not cumulative.

Value

A list with components eig, scores, loadings, fitted, ncomp, R2, iter, center, scale.

Author(s)

Kevin Wright
uscrime

References


Examples

```r
B <- matrix(c(50, 67, 90, 98, 120, 55, 71, 93, 102, 129, 65, 76, 95, 105, 134, 50, 80, 102, 130, 138, 60, 82, 97, 135, 151, 65, 89, 106, 137, 153, 75, 95, 117, 133, 155), ncol=5, byrow=TRUE)
rownames(B) <- c("G1","G2","G3","G4","G5","G6","G7")
colnames(B) <- c("E1","E2","E3","E4","E5")
dim(B) # 7 x 5
p1 <- nipals(B)
dim(p1$scores) # 7 x 5
dim(p1$loadings) # 5 x 5

B2 = B
p2 = nipals(B2, fitted=TRUE)

# Two ways to make a biplot

# method 1
biplot(p2$scores, p2$loadings)

# method 2
class(p2) <- "princomp"
p2$sdev <- sqrt(p2$eig)
biplot(p2, scale=0)
```

uscrime

U.S. Crime rates per 100,00 people

Description

U.S. Crime rates per 100,00 people for 7 categories in each of the 50 U.S. states in 1977.

Usage

uscrime
Format

A data frame with 50 observations on the following 8 variables.

- **state**: U.S. state
- **murder**: murders
- **rape**: rapes
- **robbery**: robbery
- **assault**: assault
- **burglary**: burglary
- **larceny**: larceny
- **autotheft**: automobile thefts

Details

There are two missing values.

Source

Documentation Example 3 for PROC HPPRINCOMP. http://documentation.sas.com/api/docsets/stathpug/14.2/content/stathpug_code_hppriex3.htm?locale=en

References


Examples

```r
library(nipals)
head(uscrime)

# SAS deletes rows with missing values
dat <- uscrime[complete.cases(uscrime), ]
dat <- as.matrix(dat[, -1])
m1 <- nipals(dat)  # complete-data method

# Traditional NIPALS with missing data
dat <- uscrime
dat <- as.matrix(dat[, -1])
m2 <- nipals(dat, gramschmidt=FALSE)  # missing
round(crossprod(m2$loadings),3)  # Prin Comps not quite orthogonal

# Gram-Schmidt corrected NIPALS
m3 <- nipals(dat, gramschmidt=TRUE)  # TRUE is default
round(crossprod(m3$loadings),3)  # Prin Comps are orthogonal
```
Index

*Topic datasets
  uscrime, 5

empca, 2
nipals, 3
uscrime, 5