Package 'LFM'

April 16, 2025

Type Package

Title Laplace Factor Model Analysis and Evaluation

Date 2025-4-16

Version 0.3.0

Description Enables the generation of Laplace factor models across diverse Laplace distributions and facilitates the application of Sparse Online Principal Component (SOPC), Incremental Principal Component (IPC), Projected Principal Component (PPC), Perturbation Principal Component (PPC), Stochastic Approximation Principal Component (SAPC), Sparse Principal Component (SPC) and other PC methods and Farm Test methods to these models. Evaluates the efficacy of these methods within the context of Laplace factor models by scrutinizing parameter estimation accuracy, mean square error, and the degree of sparsity.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports stats, FarmTest, MASS, SOPC, LaplacesDemon, matrixcalc,
relliptical

NeedsCompilation no

Language en-US

Author Guangbao Guo [aut, cre],
Siqi Liu [aut]

Depends R (>= 3.5.0)

LazyData true

BuildManual yes

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Maintainer Guangbao Guo <ggb11111111@163.com>

Repository CRAN

Date/Publication 2025-04-16 08:10:06 UTC

Contents

`bankruptcy` *Bankruptcy data*

Description

The data set contain the ratio of retained earnings (RE) to total assets, and the ratio of earnings before interests and taxes (EBIT) to total assets of 66 American firms recorded in the form of ratios. Half of the selected firms had filed for bankruptcy.

Usage

```
data(bankruptcy)
```

Format

A data frame with the following variables:

Y The status of the firm: `0` bankruptcy or `1` financially sound;

RE Ratio of retained earnings to total assets;

EBIT Ratio of earnings before interests and taxes to total assets

Examples

```
data(bankruptcy)
```

`concrete` *Concrete Slump Test Data*

Description

This dataset contains measurements related to the slump test of concrete, including input variables (concrete ingredients) and output variables (slump, flow, and compressive strength).

Usage

```
concrete
```

Format

A data frame with 103 rows and 10 columns.

- `Cement`: Amount of cement (kg in one M^3 concrete).
- `Slag`: Amount of slag (kg in one M^3 concrete).
- `Fly_ash`: Amount of fly ash (kg in one M^3 concrete).
- `Water`: Amount of water (kg in one M^3 concrete).
- `SP`: Amount of superplasticizer (kg in one M^3 concrete).
- `Coarse_Aggr`: Amount of coarse aggregate (kg in one M^3 concrete).
- `Fine_Aggr`: Amount of fine aggregate (kg in one M^3 concrete).
- `SLUMP`: Slump of the concrete (cm).
- `FLOW`: Flow of the concrete (cm).
- `Compressive_Strength`: 28-day compressive strength of the concrete (MPa).

Details

The dataset includes 7 input variables (concrete ingredients) and 3 output variables (slump, flow, and compressive strength). The initial dataset had 78 data points, with an additional 25 data points added later.

Note

The dataset assumes that all measurements are accurate and does not account for measurement errors. The slump flow of concrete is influenced by multiple factors, including water content and other ingredients.

Source

Donor: I-Cheng Yeh \ Email: icyeh 'at' chu.edu.tw \ Institution: Department of Information Management, Chung-Hua University (Republic of China) \ Other contact information: Department of Information Management, Chung-Hua University, Hsin Chu, Taiwan 30067, R.O.C.

Examples

```
# Load the dataset
data(concrete)

# Print the first few rows of the dataset
print(head(concrete))
```

`FanPC_LFM` *Apply the FanPC method to the Laplace factor model*

Description

This function performs Factor Analysis via Principal Component (FanPC) on a given data set. It calculates the estimated factor loading matrix (AF), specific variance matrix (DF), and the mean squared errors.

Usage

```
FanPC_LFM(data, m, A, D, p)
```

Arguments

`data`	A matrix of input data.
`m`	The number of principal components.
`A`	The true factor loadings matrix.
`D`	The true uniquenesses matrix.
`p`	The number of variables.

Value

A list containing:

`AF`	Estimated factor loadings.
`DF`	Estimated uniquenesses.
`MSESigmaA`	Mean squared error for factor loadings.
`MSESigmaD`	Mean squared error for uniquenesses.
`LSigmaA`	Loss metric for factor loadings.
`LSigmaD`	Loss metric for uniquenesses.

Examples

```
library(SOPC)
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- FanPC_LFM(data, m, A, D, p)
print(results)
```

Ftest_LFM *Apply the Farmtest method to the Laplace factor model*

Description

This function simulates data from a Lapalce factor model and applies the FarmTest for multiple hypothesis testing. It calculates the false discovery rate (FDR) and power of the test.

Usage

```
Ftest_LFM(data, p1)
```

Arguments

`data`	A matrix or data frame of simulated or observed data from a Laplace factor model.
`p1`	The proportion of non-zero hypotheses.

Value

A list containing the following elements:

`FDR`	The false discovery rate, which is the proportion of false positives among all discoveries (rejected hypotheses).
`Power`	The statistical power of the test, which is the probability of correctly rejecting a false null hypothesis.
`PValues`	A vector of p-values associated with each hypothesis test.
`RejectedHypotheses`	The total number of hypotheses that were rejected by the FarmTest.

Examples

```
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
p1=40
results <- Ftest_LFM(data, p1)
print(results$FDR)
print(results$Power)
```

GulPC_LFM — *Apply the GulPC method to the Laplace factor model*

Description

This function performs General Unilateral Loading Principal Component (GulPC) analysis on a given data set. It calculates the estimated values for the first layer and second layer loadings, specific variances, and the mean squared errors.

Usage

```
GulPC_LFM(data, m, A, D)
```

Arguments

`data`	A matrix of input data.
`m`	The number of principal components.
`A`	The true factor loadings matrix.
`D`	The true uniquenesses matrix.

Value

A list containing:

`AU1`	The first layer loading matrix.
`AU2`	The second layer loading matrix.
`DU3`	The estimated specific variance matrix.
`MSESigmaD`	Mean squared error for uniquenesses.
`LSigmaD`	Loss metric for uniquenesses.

Examples

```
library(SOPC)
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- GulPC_LFM(data, m, A, D)
print(results)
```

IPC_LFM *Apply the IPC method to the Laplace factor model*

Description

This function performs Incremental Principal Component Analysis (IPC) on the provided data. It updates the estimated factor loadings and uniquenesses as new data points are processed, calculating mean squared errors and loss metrics for comparison with true values.

Usage

```
IPC_LFM(data, m, A, D, p)
```

Arguments

data	The data used in the IPC analysis.
m	The number of common factors.
A	The true factor loadings matrix.
D	The true uniquenesses matrix.
p	The number of variables.

Value

A list of metrics including:

Ai	Estimated factor loadings updated during the IPC analysis, a matrix of estimated factor loadings.

Di	Estimated uniquenesses updated during the IPC analysis, a vector of estimated uniquenesses corresponding to each variable.
MSESigmaA	Mean squared error of the estimated factor loadings (Ai) compared to the true loadings (A).
MSESigmaD	Mean squared error of the estimated uniquenesses (Di) compared to the true uniquenesses (D).
LSigmaA	Loss metric for the estimated factor loadings (Ai), indicating the relative error compared to the true loadings (A).
LSigmaD	Loss metric for the estimated uniquenesses (Di), indicating the relative error compared to the true uniquenesses (D).

Examples

```
library(SOPC)
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- IPC_LFM(data, m, A, D, p)
print(results)
```

LFM *Generate Laplace factor models*

Description

The function is to generate Laplace factor model data. The function supports various distribution types for generating the data, including: - ‘truncated_laplace‘: Truncated Laplace distribution - ‘log_laplace‘: Univariate Symmetric Log-Laplace distribution - ‘Asymmetric Log_Laplace‘: Log-Laplace distribution - ‘Skew-Laplace‘: Skew-Laplace distribution

Usage

```
LFM(n, p, m, distribution_type)
```

Arguments

- `n` An integer specifying the sample size.
- `p` An integer specifying the sample dimensionality or the number of variables.
- `m` An integer specifying the number of factors in the model.
- `distribution_type` A character string indicating the type of distribution to use for generating the data.

Value

A list containing the following elements:

- `data` A numeric matrix of the generated data.
- `A` A numeric matrix representing the factor loadings.
- `D` A numeric matrix representing the uniquenesses, which is a diagonal matrix.

Examples

```
library(MASS)
library(matrixcalc)
library(relliptical)
n <- 1000
p <- 10
m <- 5
sigma1 <- 1
sigma2 <- matrix(c(1,0.7,0.7,1), 2, 2)
distribution_type <- "truncated_laplace"
results <- LFM(n, p, m, distribution_type)
print(results)
```

OPC_LFM *Apply the OPC method to the Laplace factor model*

Description

This function computes Online Principal Component Analysis (OPC) for the provided input data, estimating factor loadings and uniquenesses. It calculates mean squared errors and sparsity for the estimated values compared to true values.

Usage

```
OPC_LFM(data, m = m, A, D, p)
```

Arguments

data	A matrix of input data.
m	The number of principal components.
A	The true factor loadings matrix.
D	The true uniquenesses matrix.
p	The number of variables.

Value

A list containing:

Ao	Estimated factor loadings.
Do	Estimated uniquenesses.
MSEA	Mean squared error for factor loadings.
MSED	Mean squared error for uniquenesses.
tau	The sparsity.

Examples

```
library(SOPC)
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- OPC_LFM(data, m, A, D, p)
print(results)
```

PC1_LFM *Apply the PC method to the Laplace factor model*

Description

This function performs Principal Component Analysis (PCA) on a given data set to reduce dimensionality. It calculates the estimated values for the loadings, specific variances, and the covariance matrix.

Usage

```
PC1_LFM(data, m, A, D)
```

Arguments

data	The total data set to be analyzed.
m	The number of principal components to retain in the analysis.
A	The true factor loadings matrix.
D	The true uniquenesses matrix.

Value

A list containing:

A1	Estimated factor loadings.
D1	Estimated uniquenesses.
MSESigmaA	Mean squared error for factor loadings.
MSESigmaD	Mean squared error for uniquenesses.
LSigmaA	Loss metric for factor loadings.
LSigmaD	Loss metric for uniquenesses.

Examples

```
library(SOPC)
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- PC1_LFM(data, m, A, D)
print(results)
```

PC2_LFM *Apply the PC method to the Laplace factor model*

Description

This function performs Principal Component Analysis (PCA) on a given data set to reduce dimensionality. It calculates the estimated values for the loadings, specific variances, and the covariance matrix.

Usage

```
PC2_LFM(data, m, A, D)
```

Arguments

data	The total data set to be analyzed.
m	The number of principal components to retain in the analysis.
A	The true factor loadings matrix.
D	The true uniquenesses matrix.

Value

A list containing:

A2	Estimated factor loadings.
D2	Estimated uniquenesses.
MSESigmaA	Mean squared error for factor loadings.
MSESigmaD	Mean squared error for uniquenesses.
LSigmaA	Loss metric for factor loadings.
LSigmaD	Loss metric for uniquenesses.

Examples

```
library(SOPC)
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
```

```
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- PC2_LFM(data, m, A, D)
print(results)
```

PPC1_LFM *Apply the PPC method to the Laplace factor model*

Description

This function computes Perturbation Principal Component Analysis (PPC) for the provided input data, estimating factor loadings and uniquenesses. It calculates mean squared errors and loss metrics for the estimated values compared to true values.

Usage

```
PPC1_LFM(data, m, A, D, p)
```

Arguments

data	A matrix of input data.
m	The number of principal components.
A	The true factor loadings matrix.
D	The true uniquenesses matrix.
p	The number of variables.

Value

A list containing:

Ap	Estimated factor loadings.
Dp	Estimated uniquenesses.
MSESigmaA	Mean squared error for factor loadings.
MSESigmaD	Mean squared error for uniquenesses.
LSigmaA	Loss metric for factor loadings.
LSigmaD	Loss metric for uniquenesses.

Examples

```
library(SOPC)
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
```

```
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- PPC1_LFM(data, m, A, D, p)
print(results)
```

PPC2_LFM *Apply the PPC method to the Laplace factor model*

Description

This function performs Projected Principal Component Analysis (PPC) on a given data set to reduce dimensionality. It calculates the estimated values for the loadings, specific variances, and the covariance matrix.

Usage

```
PPC2_LFM(data, m, A, D)
```

Arguments

`data`	The total data set to be analyzed.
`m`	The number of principal components.
`A`	The true factor loadings matrix.
`D`	The true uniquenesses matrix.

Value

A list containing:

`Ap2`	Estimated factor loadings.
`Dp2`	Estimated uniquenesses.
`MSESigmaA`	Mean squared error for factor loadings.
`MSESigmaD`	Mean squared error for uniquenesses.
`LSigmaA`	Loss metric for factor loadings.
`LSigmaD`	Loss metric for uniquenesses.

Examples

```
library(SOPC)
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- PPC2_LFM(data, m, A, D)
print(results)
```

`protein` *Protein Secondary Structure Data*

Description

This dataset contains protein sequences and their corresponding secondary structures, including beta-sheets (E), helices (H), and coils (_).

Usage

```
protein
```

Format

A data frame with multiple rows and columns representing protein sequences and their secondary structures.

- `Sequence`: Amino acid sequence (using 3-letter codes).
- `Structure`: Secondary structure of the protein (E for beta-sheet, H for helix, _ for coil).
- `Parameters`: Additional parameters for neural networks (to be ignored).
- `Biophysical_Constants`: Biophysical constants (to be ignored).

Details

The dataset is used for predicting protein secondary structures from amino acid sequences. The first few numbers in each sequence are parameters for neural networks and should be ignored. The '<' symbol is used as a spacer between proteins and to mark the beginning and end of sequences.

Note

The biophysical constants included in the dataset were found to be unhelpful and are generally ignored in analysis.

Source

Vince G. Sigillito, Applied Physics Laboratory, Johns Hopkins University.

Examples

```
# Load the dataset
data(protein)

# Print the first few rows of the dataset
print(head(protein))
```

`review` *Review*

Description

This dataset contains travel reviews from TripAdvisor.com, covering destinations in 11 categories across East Asia. Each traveler's rating is mapped to a scale from Terrible (0) to Excellent (4), and the average rating for each category per user is provided.

Usage

```
review
```

Format

A data frame with multiple rows and 12 columns.

- `User_ID`: Unique identifier for each user (Categorical).
- `Art_Galleries`: Average user feedback on art galleries.
- `Dance_Clubs`: Average user feedback on dance clubs.
- `Juice_Bars`: Average user feedback on juice bars.
- `Restaurants`: Average user feedback on restaurants.
- `Museums`: Average user feedback on museums.
- `Resorts`: Average user feedback on resorts.
- `Parks_Picnic_Spots`: Average user feedback on parks and picnic spots.
- `Beaches`: Average user feedback on beaches.
- `Theaters`: Average user feedback on theaters.
- `Religious_Institutions`: Average user feedback on religious institutions.

Details

The dataset is populated by crawling TripAdvisor.com and includes reviews on destinations in 11 categories across East Asia. Each traveler's rating is mapped as follows: Excellent (4), Very Good (3), Average (2), Poor (1), and Terrible (0). The average rating for each category per user is used.

Note

This dataset is licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) license, which allows for sharing and adaptation of the data for any purpose, provided that appropriate credit is given.

Source

UCI Machine Learning Repository

Examples

```
# Load the dataset
data(review)

# Print the first few rows of the dataset
print(head(review))
```

---

riboflavin *Riboflavin Production Data*

Description

This dataset contains measurements of riboflavin (vitamin B2) production by Bacillus subtilis, a Gram-positive bacterium commonly used in industrial fermentation processes. The dataset includes $n = 71$ observations with $p = 4088$ predictors, representing the logarithm of the expression levels of 4088 genes. The response variable is the log-transformed riboflavin production rate.

Usage

```
data(riboflavin)
```

Format

y Log-transformed riboflavin production rate (original name: `q_RIBFLV`). This is a continuous variable indicating the efficiency of riboflavin production by the bacterial strain.

x A matrix of dimension 71×4088 containing the logarithm of the expression levels of 4088 genes. Each column corresponds to a gene, and each row corresponds to an observation (experimental condition or time point).

Details

The riboflavin dataset is a high-dimensional dataset commonly used in statistical research, particularly in the fields of bioinformatics and systems biology. It was originally collected to study the genetic regulation of riboflavin biosynthesis in Bacillus subtilis. The data were generated using DNA microarray technology to measure gene expression levels under various experimental conditions.

Note

The dataset is provided by DSM Nutritional Products Ltd., a leading company in the field of nutritional ingredients. The data have been preprocessed and normalized to account for technical variations in the microarray measurements.

Source

DSM Nutritional Products Ltd., Basel, Switzerland.

Examples

```
# Load the riboflavin dataset
data(riboflavin)

# Display the dimensions of the dataset
print(dim(riboflavin$x))
print(length(riboflavin$y))
```

---

riboflavinv100 *Riboflavin Production Data (Top 100 Genes)*

Description

This dataset is a subset of the riboflavin production data by Bacillus subtilis, containing $n = 71$ observations. It includes the response variable (log-transformed riboflavin production rate) and the 100 genes with the largest empirical variances from the original dataset.

Usage

```
data(riboflavinv100)
```

Format

- **y** Log-transformed riboflavin production rate (original name: `q_RIBFLV`). This is a continuous variable indicating the efficiency of riboflavin production by the bacterial strain.
- **x** A matrix of dimension 71×100 containing the logarithm of the expression levels of the 100 genes with the largest empirical variances.

Details

This dataset is derived from the original riboflavin dataset, which contains 4088 gene expressions. The riboflavinV100 dataset is created for ease of reproduction in examples and contains only the 100 genes with the largest empirical variances. It is commonly used in statistical research for high-dimensional data analysis.

Note

The dataset is provided by DSM Nutritional Products Ltd., a leading company in the field of nutritional ingredients. The data have been preprocessed and normalized.

Source

DSM Nutritional Products Ltd., Basel, Switzerland.

Examples

```
# Load the riboflavinv100 dataset
data(riboflavinv100)

# Display the dimensions of the dataset
print(dim(riboflavinv100$x))
print(length(riboflavinv100$y))
```

SAPC_LFM *Apply the SAPC method to the Laplace factor model*

Description

This function calculates several metrics for the SAPC method, including the estimated factor loadings and uniquenesses, and various error metrics comparing the estimated matrices with the true matrices.

Usage

```
SAPC_LFM(data, m, A, D, p)
```

Arguments

`data`	The data used in the SAPC analysis.
`m`	The number of common factors.
`A`	The true factor loadings matrix.
`D`	The true uniquenesses matrix.
`p`	The number of variables.

Value

A list of metrics including:

- `Asa` Estimated factor loadings matrix obtained from the SAPC analysis.
- `Dsa` Estimated uniquenesses vector obtained from the SAPC analysis.
- `MSESigmaA` Mean squared error of the estimated factor loadings (Asa) compared to the true loadings (A).
- `MSESigmaD` Mean squared error of the estimated uniquenesses (Dsa) compared to the true uniquenesses (D).
- `LSigmaA` Loss metric for the estimated factor loadings (Asa), indicating the relative error compared to the true loadings (A).
- `LSigmaD` Loss metric for the estimated uniquenesses (Dsa), indicating the relative error compared to the true uniquenesses (D).

Examples

```
library(SOPC)
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- SAPC_LFM(data, m, A, D, p)
print(results)
```

SOPC_LFM *Apply the SOPC method to the Laplace factor model*

Description

This function calculates various metrics for the SOPC analysis on the Laplace factor model. It estimates the factor loadings and uniquenesses, and computes metrics such as mean squared error, loss metrics, and sparsity.

Usage

```
SOPC_LFM(data, m, p, A, D)
```

Arguments

data	A numeric matrix containing the data used in the SOPC analysis.
m	An integer specifying the number of subsets or common factors.
p	An integer specifying the number of variables in the data.
A	A numeric matrix representing the true factor loadings.
D	A numeric matrix representing the true uniquenesses.

Value

A list containing the following metrics:

Aso	Estimated factor loadings matrix.
Dso	Estimated uniquenesses matrix.
MSEA	Mean squared error of the estimated factor loadings (Aso) compared to the true loadings (A).
MSED	Mean squared error of the estimated uniquenesses (Dso) compared to the true uniquenesses (D).
LSA	Loss metric for the estimated factor loadings (Aso), indicating the relative error compared to the true loadings (A).
LSD	Loss metric for the estimated uniquenesses (Dso), indicating the relative error compared to the true uniquenesses (D).
tauA	Proportion of zero factor loadings in the estimated loadings matrix (Aso), representing the sparsity.

Examples

```
library(MASS)
library(SOPC)
library(matrixcalc)
library(LaplacesDemon)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- SOPC_LFM(data, m, p, A, D)
print(results)
```

SPC_LFM *Apply the SPC method to the Laplace factor model*

Description

This function performs Sparse Principal Component Analysis (SPC) on the input data. It estimates factor loadings and uniquenesses while calculating mean squared errors and loss metrics for comparison with true values.

Usage

```
SPC_LFM(data, A, D, m, p)
```

Arguments

`data`	The data used in the SPC analysis.
`A`	The true factor loadings matrix.
`D`	The true uniquenesses matrix.
`m`	The number of common factors.
`p`	The number of variables.

Value

A list containing:

`As`	Estimated factor loadings, a matrix of estimated factor loadings from the SPC analysis.
`Ds`	Estimated uniquenesses, a vector of estimated uniquenesses corresponding to each variable.
`MSESigmaA`	Mean squared error of the estimated factor loadings (As) compared to the true loadings (A).
`MSESigmaD`	Mean squared error of the estimated uniquenesses (Ds) compared to the true uniquenesses (D).
`LSigmaA`	Loss metric for the estimated factor loadings (As), indicating the relative error compared to the true loadings (A).
`LSigmaD`	Loss metric for the estimated uniquenesses (Ds), indicating the relative error compared to the true uniquenesses (D).
`tau`	Proportion of zero factor loadings in the estimated loadings matrix (As).

Examples

```
library(SOPC)
library(LaplacesDemon)
library(MASS)
n=1000
p=10
m=5
mu=t(matrix(rep(runif(p,0,1000),n),p,n))
mu0=as.matrix(runif(m,0))
sigma0=diag(runif(m,1))
F=matrix(mvrnorm(n,mu0,sigma0),nrow=n)
A=matrix(runif(p*m,-1,1),nrow=p)
lanor <- rlaplace(n*p,0,1)
epsilon=matrix(lanor,nrow=n)
D=diag(t(epsilon)%*%epsilon)
data=mu+F%*%t(A)+epsilon
results <- SPC_LFM(data, A, D, m, p)
print(results)
```

`vehicle` *In Vehicle Coupon Recommendation Data*

Description

This dataset contains information about coupon recommendations made to drivers in a vehicle, including various contextual features and the outcome of whether the coupon was accepted.

Usage

```
vehicle
```

Format

A data frame with multiple rows and 27 columns representing different features related to coupon recommendations.

- `destination`: Driver's destination - No Urgent Place, Home, Work.
- `passanger`: Passengers in the car - Alone, Friend(s), Kid(s), Partner.
- `weather`: Current weather - Sunny, Rainy, Snowy.
- `temperature`: Temperature in Fahrenheit - 55, 80, 30.
- `time`: Time of day - 2PM, 10AM, 6PM, 7AM, 10PM.
- `coupon`: Type of coupon - Restaurant(<$20), Coffee House, Carry out & Take away, Bar, Restaurant($20-$50).
- `expiration`: Coupon expiration - 1d (1 day), 2h (2 hours).
- `gender`: Driver's gender - Female, Male.
- `age`: Driver's age group - 21, 46, 26, 31, 41, 50plus, 36, below21.

- `maritalStatus`: Driver's marital status - Unmarried partner, Single, Married partner, Divorced, Widowed.
- `has_Children`: Whether the driver has children - 1, 0.
- `education`: Driver's education level - Some college - no degree, Bachelors degree, Associates degree, High School Graduate, Graduate degree (Masters or Doctorate), Some High School.
- `occupation`: Driver's occupation - Various categories including Unemployed, Student, etc.
- `income`: Driver's income range - Various ranges such as $37500 - $49999, $62500 - $74999, etc.
- `Bar`: Frequency of bar visits per month - never, less1, 1~3, gt8, nan4~8.
- `CoffeeHouse`: Frequency of coffeehouse visits per month - never, less1, 4~8, 1~3, gt8, nan.
- `CarryAway`: Frequency of getting take-away food per month - n4~8, 1~3, gt8, less1, never.
- `RestaurantLessThan20`: Frequency of visiting restaurants with average expense <$20 per month - 4~8, 1~3, less1, gt8, never.
- `Restaurant20To50`: Frequency of visiting restaurants with average expense $20-$50 per month - 1~3, less1, never, gt8, 4~8, nan.
- `toCoupon_GEQ15min`: Driving distance to the coupon location greater than 15 minutes - 0, 1.
- `toCoupon_GEQ25min`: Driving distance to the coupon location greater than 25 minutes - 0, 1.
- `direction_same`: Whether the coupon location is in the same direction as the current destination - 0, 1.
- `direction_opp`: Whether the coupon location is in the opposite direction of the current destination - 1, 0.
- `Y`: Whether the coupon was accepted - 1, 0.

Details

The dataset includes various contextual features such as driver demographics, weather conditions, time of day, and coupon details. It also includes features related to driving habits and preferences. The target variable is whether the coupon was accepted (Y).

Note

This dataset can be used for classification tasks to predict coupon acceptance based on contextual features.

Source

UCI Machine Learning Repository

Examples

```
# Load the dataset
data(vehicle)

# Print the first few rows of the dataset
print(head(vehicle))
```

`wholesale` *Wholesale Customers Data*

Description

This dataset contains the annual spending amounts of wholesale customers on various product categories, along with their channel and region information.

Usage

```
wholesale
```

Format

A data frame with 440 rows and 8 columns.

- `FRESH`: Annual spending (m.u.) on fresh products.
- `MILK`: Annual spending (m.u.) on milk products.
- `GROCERY`: Annual spending (m.u.) on grocery products.
- `FROZEN`: Annual spending (m.u.) on frozen products.
- `DETERGENTS_PAPER`: Annual spending (m.u.) on detergents and paper products.
- `DELICATESSEN`: Annual spending (m.u.) on delicatessen products.
- `CHANNEL`: Customers' channel - Horeca (Hotel/Restaurant/Café) or Retail channel (Nominal).
- `REGION`: Customers' region - Lisbon, Oporto or Other (Nominal).

Details

The dataset includes both continuous spending variables and nominal classification variables (CHANNEL and REGION). Descriptive statistics for the continuous variables are provided, including minimum, maximum, mean, and standard deviation.

Note

The data can be used for clustering analysis to segment customers based on their purchasing behavior.

Source

UCI Machine Learning Repository

Examples

```
# Load the dataset
data(wholesale)
```

yacht_hydrodynamics *Yacht Hydrodynamics Data*

Description

This dataset contains the hydrodynamic characteristics of sailing yachts, including design parameters and performance metrics.

Usage

```
yacht_hydrodynamics
```

Format

A data frame with 308 rows and 7 columns.

- `Residuary Resistance`: Residuary resistance per unit weight of displacement (performance metric).
- `Longitudinal Position of Center of Buoyancy`: Longitudinal position of the center of buoyancy.
- `Prismatic Coefficient`: Prismatic coefficient.
- `Length-Displacement Ratio`: Length-displacement ratio.
- `Beam-Draft Ratio`: Beam-draft ratio.
- `Length-Beam Ratio`: Length-beam ratio.
- `Froude Number`: Froude number.

Details

The dataset contains hydrodynamic data for sailing yachts, with the goal of predicting the residuary resistance from various design parameters.

Note

The dataset is commonly used for regression analysis and machine learning tasks to model the relationship between design parameters and performance metrics.

Source

UCI Machine Learning Repository

Examples

```
# Load the dataset
data(yacht_hydrodynamics)

# Print the first few rows of the dataset
print(head(yacht_hydrodynamics))
```

Index