Package 'DSFM'

May 7, 2026

Type Package

Title Distributed Skew Factor Model Estimation Methods

Version 1.0.1

Author Guangbao Guo [aut, cre],
Yu Jin [aut]

Maintainer Guangbao Guo <ggb11111111@163.com>

Description
Provides a distributed framework for simulating and estimating skew factor models under various skewed and heavy-tailed distributions. The methods support distributed data generation, aggregation of local estimators, and evaluation of estimation performance via mean squared error, relative error, and sparsity measures. The distributed principal component (PC) estimators implemented in the package include 'IPC' (Independent Principal Component),'PPC' (Project Principal Component), 'SPC' (Sparse Principal Component), and other related distributed PC methods. The methodological background follows Guo G. (2023) <doi:10.1007/s00180-022-01270-z>.

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 4.0.0)

Imports MASS, matrixcalc, sn, stats, psych, elasticnet, SOPC

Suggests ggplot2, cowplot, testthat (>= 3.0.0)

NeedsCompilation no

Language en-US

License MIT + file LICENSE

Repository CRAN

Date/Publication 2025-12-01 15:10:02 UTC

Contents

`AirQuality` *Air Quality Data Set (UCI)*

Description

Measurements of air quality variables in an Italian city collected over several months in 2004–2005. The data includes hourly averaged responses from chemical sensors embedded in an air quality chemical multi-sensor device.

Usage

```
data(AirQuality)
```

Format

A data frame with 9358 observations on the following 15 variables. Some variable names use parentheses, which may need to be quoted with backticks in R.

- `Date`: Date (in DD/MM/YYYY format)
- `Time`: Time (in HH.MM.SS format)
- `CO.GT.`: Carbon Monoxide concentration (mg/m³)
- `PT08.S1.CO.`: Sensor 1 response
- `NMHC.GT.`: Non-methane hydrocarbons (µg/m³)
- `C6H6.GT.`: Benzene concentration (µg/m³)
- `PT08.S2.NMHC.`: Sensor 2 response
- `NOx.GT.`: Nitric oxide concentration (ppb)
- `PT08.S3.NOx.`: Sensor 3 response
- `NO2.GT.`: Nitrogen dioxide concentration (µg/m³)
- `PT08.S4.NO2.`: Sensor 4 response
- `PT08.S5.O3.`: Sensor 5 response
- `T`: Temperature (°C)

- RH: Relative Humidity (%)
- AH: Absolute Humidity

Some variables contain missing values coded as -200.

Details

The dataset contains air quality data recorded in a densely populated area of an Italian city between March 2004 and February 2005. The data were collected using an array of chemical sensors and meteorological instruments.

This dataset is frequently used for tasks such as missing value imputation, time series analysis, regression, and machine learning model evaluation.

Source

De Vito, S., Massera, E., Piga, M., Martinotto, L., & Di Francia, G. (2008).\ *UCI Machine Learning Repository: Air Quality Data Set*.\ Available at: https://archive.ics.uci.edu/ml/datasets/Air+Quality

References

De Vito, S., Massera, E., Piga, M., Martinotto, L., & Di Francia, G. (2008).\ Semi-Supervised Learning Techniques in Artificial Olfaction: A Novel Approach to Classification Problems and Drift Counteraction.\ *IEEE Sensors Journal*, **8**(12), 2030–2038.

Examples

```
data(AirQuality)

# Replace missing values (-200) with NA
AirQuality[AirQuality == -200] <- NA

# Check if there are non-NA values before plotting
if (sum(!is.na(AirQuality$CO.GT.)) > 0) {
  plot(AirQuality$CO.GT., type = "l", ylab = "CO (mg/m³)",
       main = "Hourly CO Concentration")
} else {
  message("No non-NA values in CO.GT. column to plot")
}
```

`calculate_errors` *calculate_errors Function*

Description

This function calculates the Mean Squared Error (MSE) and relative error for factor loadings and uniqueness estimates obtained from factor analysis.

Usage

```
calculate_errors(data, A, D)
```

Arguments

data	Matrix of SFM data.
A	Matrix of true factor loadings.
D	Matrix of true uniquenesses.

Value

A named vector containing:

MSEA	Mean Squared Error for factor loadings.
MSED	Mean Squared Error for uniqueness estimates.
LSA	Relative error for factor loadings.
LSD	Relative error for uniqueness estimates.

Examples

```
set.seed(123) # For reproducibility
# Define dimensions
n <- 10  # Number of samples
p <- 5   # Number of factors

# Generate matrices with compatible dimensions
A <- matrix(runif(p * p, -1, 1), nrow = p)  # Factor loadings matrix (p x p)
D <- diag(runif(p, 1, 2))  # Uniquenesses matrix (p x p)
data <- matrix(runif(n * p), nrow = n)  # Data matrix (n x p)

# Calculate errors (only if SOPC is installed)
if (requireNamespace("SOPC", quietly = TRUE)) {
  errors <- calculate_errors(data, A, D)
  print(errors)
}
```

DFanPC *Distributed Fan Principal Component Analysis*

Description

This function performs distributed Fan-type principal component analysis on a numeric dataset split across multiple nodes.

Usage

```
DFanPC(data, m, n1, K)
```

Arguments

data	A numeric matrix containing the total dataset.
m	An integer specifying the number of principal components.
n1	An integer specifying the length of each data subset.
K	An integer specifying the number of nodes.

Value

A list with the following components:

AF List of estimated loading matrices for each node.

DF List of diagonal residual variance matrices for each node.

SigmahatF List of covariance matrices for each node.

Examples

```
set.seed(123)
data <- matrix(rnorm(500), nrow = 100, ncol = 5)
DFanPC(data = data, m = 3, n1 = 20, K = 5)
```

DGaoPC *Distributed Gao Principal Component Analysis*

Description

Performs distributed Gao-type principal component analysis on a numeric dataset split across multiple nodes.

Usage

```
DGaoPC(data, m, n1, K)
```

Arguments

data	A numeric matrix containing the total dataset.
m	An integer specifying the number of principal components for the first stage.
n1	An integer specifying the length of each data subset.
K	An integer specifying the number of nodes.

Value

A list with the following components:

AG1 List of estimated loading matrices for the first-stage components for each node.

AG2 List of estimated loading matrices for the second-stage components for each node.

DG3 List of diagonal residual variance matrices for each node.

sGhat List of covariance matrices of reconstructed data for each node.

Examples

```
set.seed(123)
data <- matrix(rnorm(500), nrow = 100, ncol = 5)
DGaoPC(data = data, m = 3, n1 = 20, K = 5)
```

DGulPC *Distributed Gul Principal Component Analysis*

Description

Performs distributed Gul-type principal component analysis on a numeric dataset split across multiple nodes.

Usage

```
DGulPC(data, m, n1, K)
```

Arguments

data	A numeric matrix containing the total dataset.
m	An integer specifying the number of principal components for the first stage.
n1	An integer specifying the length of each data subset.
K	An integer specifying the number of nodes.

Value

A list with the following components:

AU1 List of estimated first-stage loading matrices for each node.

AU2 List of estimated second-stage loading matrices for each node.

DU3 List of diagonal residual variance matrices for each node.

shat List of covariance matrices of reconstructed data for each node.

Examples

```
set.seed(123)
data <- matrix(rnorm(500), nrow = 100, ncol = 5)
DGulPC(data = data, m = 3, n1 = 20, K = 5)
```

DPC *Distributed Principal Component Analysis*

Description

Performs distributed principal component analysis on a numeric dataset split across multiple nodes. Estimates loading matrices, residual variances, and covariance matrices for each node.

Usage

```
DPC(data, m, n1, K)
```

Arguments

data	A numeric matrix containing the total dataset.
m	An integer specifying the number of principal components.
n1	An integer specifying the length of each data subset.
K	An integer specifying the number of nodes.

Value

A list with the following components:

Ahat List of estimated loading matrices for each node.

Dhat List of diagonal residual variance matrices for each node.

Sigmahat List of covariance matrices for each node.

Examples

```
set.seed(123)
data <- matrix(rnorm(500), nrow = 100, ncol = 5)
DPC(data = data, m = 3, n1 = 20, K = 5)
```

DPPC *Distributed Probabilistic Principal Component Analysis*

Description

Performs distributed probabilistic principal component analysis (PPC) on a numeric dataset split across multiple nodes. Estimates loading matrices, residual variances, and covariance matrices for each node using a probabilistic approach.

Usage

```
DPPC(data, m, n1, K)
```

Arguments

data	A numeric matrix containing the total dataset.
m	An integer specifying the number of principal components.
n1	An integer specifying the length of each data subset.
K	An integer specifying the number of nodes.

Value

A list with the following components:

Apro List of estimated loading matrices for each node.

Dpro List of diagonal residual variance matrices for each node.

Sigmahatpro List of covariance matrices for each node.

Examples

```
set.seed(123)
data <- matrix(rnorm(500), nrow = 100, ncol = 5)
DPPC(data = data, m = 3, n1 = 20, K = 5)
```

DSPC *Distributed Sparse Principal Component Analysis*

Description

Performs distributed sparse principal component analysis (DSPC) on a numeric dataset split across multiple nodes. Estimates sparse loading matrices, residual variances, and covariance matrices for each node.

Usage

```
DSPC(data, m, gamma, n1, K)
```

Arguments

data	A numeric matrix containing the total dataset.
m	An integer specifying the number of principal components.
gamma	A numeric value specifying the sparsity parameter for SPC.
n1	An integer specifying the length of each data subset.
K	An integer specifying the number of nodes.

Value

A list with the following components:

Aspro List of sparse loading matrices for each node.

Dspro List of diagonal residual variance matrices for each node.

Sigmahatpro List of covariance matrices for each node.

Examples

```
set.seed(123)
data <- matrix(rnorm(500), nrow = 100, ncol = 5)
DSPC(data = data, m = 3, gamma = 0.03, n1 = 20, K = 5)
```

---

factor.tests *Factor Model Testing with Wald, GRS, PY tests and FDR control*

Description

Performs comprehensive factor model testing including joint tests (Wald, GRS, PY), individual asset t-tests, and False Discovery Rate control.

Usage

```
factor.tests(ret, fac, q.fdr = 0.05)
```

Arguments

ret	A T × N matrix representing the excess returns of N assets at T time points.
fac	A T × K matrix representing the returns of K factors at T time points.
q.fdr	The significance level for FDR (False Discovery Rate) testing, defaulting to 5%.

Value

A list containing the following components:

alpha	N-vector of estimated alphas for each asset
tstat	N-vector of t-statistics for testing individual alphas
pval	N-vector of p-values for individual alpha tests
Wald	Wald test statistic for joint alpha significance
p_Wald	p-value for Wald test
GRS	GRS test statistic (finite-sample F-test)
p_GRS	p-value for GRS test
PY	Pesaran and Yamagata test statistic
p_PY	p-value for PY test

reject_fdr	Logical vector indicating which assets have significant alphas after FDR correction
fdr_p	Adjusted p-values using Benjamini-Hochberg procedure
power_proxy	Number of significant assets after FDR correction

Examples

```
set.seed(42)
T <- 120
N <- 25
K <- 3
fac <- matrix(rnorm(T * K), T, K)
beta <- matrix(rnorm(N * K), N, K)
alpha <- rep(0, N)
alpha[1:3] <- 0.4 / 100  # 3 non-zero alphas
eps <- matrix(rnorm(T * N, sd = 0.02), T, N)
ret <- alpha + fac %*% t(beta) + eps
results <- factor.tests(ret, fac, q.fdr = 0.05)

# View results
cat("Wald test p-value:", results$p_Wald, "\n")
cat("GRS test p-value:", results$p_GRS, "\n")
cat("PY test p-value:", results$p_PY, "\n")
cat("Significant assets after FDR:", results$power_proxy, "\n")
```

Nutrimouse *Nutrimouse: Gene, Lipid and Grouping Data*

Description

A data frame containing gene expression, lipid measurements, and grouping variables (diet and genotype) for 40 mice from a nutrigenomics study.

Usage

```
data(Nutrimouse)
```

Format

A data frame with 40 observations on 143 variables:

- 120 numeric variables for gene expression
- 21 numeric variables for lipid measurements
- 2 categorical variables: `diet` and `genotype`

Details

This dataset was created for integrative analysis of transcriptomic and lipidomic responses of mice to different diets and genotypes.

All numeric variables (genes and lipids) are centered and scaled. The categorical variables indicate the experimental design: five diet types and two genotypes.

This format is convenient for regression, classification, and dimension reduction techniques requiring a single data frame.

Source

Extracted from the **mixOmics** package, based on: \ Martin, P. G. P., et al. (2007). *A systems biology approach to the study of gene expression and lipid metabolism in mice fed high-fat diets*. *Journal of Lipid Research*, **48**(2), 360–377.

References

González, I., Déjean, S., Martin, P. G. P., and Baccini, A. (2009). CCA: An R package to extend canonical correlation analysis. *Journal of Statistical Software*, **23**(12), 1–14.

Examples

```
data(Nutrimouse)

# View structure
str(Nutrimouse)

# Boxplot of a gene across diets
boxplot(Nutrimouse[,1] ~ Nutrimouse$diet, main = "Gene 1 Expression by Diet")

# PCA on all numeric variables (excluding factors)
nutri_numeric <- Nutrimouse[, sapply(Nutrimouse, is.numeric)]
pca_result <- prcomp(nutri_numeric, scale. = TRUE)

# PCA plot
plot(pca_result$x[,1:2], col = as.numeric(Nutrimouse$diet), pch = 19)
legend("topright", legend = levels(Nutrimouse$diet), col = 1:5, pch = 19)
```

Parkinsons_Features *Parkinson's Disease Voice Features Dataset*

Description

A dataset containing biomedical voice measurements from people with Parkinson's disease and healthy controls. The goal is to analyze voice signal features for detecting and monitoring Parkinson's disease.

Usage

```
data(Parkinsons_Features)
```

Format

A data frame with 5,876 observations on 22 variables. Each row corresponds to a voice recording from a subject.

`subject_id`	Identifier for the subject (factor or character)
`age`	Age of the subject (numeric)
`sex`	Sex of the subject (factor: Male/Female)
`test_time`	Time of test (numeric, days since baseline)
`motor_UPDRS`	Unified Parkinson's Disease Rating Scale motor score (numeric)
`total_UPDRS`	Total UPDRS score (numeric)
`Jitter`	Measure of frequency variation (numeric)
`Shimmer`	Measure of amplitude variation (numeric)
`NHR`	Noise-to-harmonics ratio (numeric)
`HNR`	Harmonics-to-noise ratio (numeric)
`RPDE`	Recurrence period density entropy (numeric)
`DFA`	Detrended fluctuation analysis (numeric)
`PPE`	Pitch period entropy (numeric)
`...`	Additional voice signal features and measurements (numeric)

All features are numerical except for identifiers and categorical variables.

Details

This dataset was collected from subjects with Parkinson's disease and healthy controls. Multiple biomedical voice measurements were recorded over time to evaluate disease progression.

The features include various jitter, shimmer, noise, and entropy measures extracted from sustained vowel phonations.

The dataset is widely used for classification and regression models aiming to predict Parkinson's disease severity or presence.

Source

UCI Machine Learning Repository: *Parkinson's Disease Classification Data Set* \ https://archive.ics.uci.edu/ml/datasets/Parkinsons+Telemonitoring

References

Tsanas, A., Little, M.A., McSharry, P.E., & Ramig, L.O. (2010). Accurate telemonitoring of Parkinson's disease progression by noninvasive speech tests. *IEEE Transactions on Biomedical Engineering*, 57(4), 884–893.

Examples

```
data(Parkinsons_Features)

if (all(startsWith(names(Parkinsons_Features), "V"))) {
  colnames(Parkinsons_Features) <- Parkinsons_Features[1, ]
  Parkinsons_Features <- Parkinsons_Features[-1, ]
}

Parkinsons_Features[] <- lapply(Parkinsons_Features, type.convert, as.is = TRUE)

summary(Parkinsons_Features$motor_UPDRS)
boxplot(motor_UPDRS ~ sex, data = Parkinsons_Features,
        main = "Motor UPDRS by Sex", ylab = "Motor UPDRS")
```

SFM *The SFM function is to generate Skew Factor Models data.*

Description

The function supports various distribution types for generating the data, including: Skew-Normal Distribution, Skew-Cauchy Distribution, Skew-t Distribution.

Usage

```
SFM(n, p, m, xi, omega, alpha, distribution_type)
```

Arguments

n	Sample size.
p	Sample dimensionality.
m	Number of factors.
xi	A numerical parameter used exclusively in the "Skew-t" distribution, representing the distribution's xi parameter.
omega	A numerical parameter representing the omega parameter of the distribution, which affects the degree of skewness in the distribution.
alpha	A numerical parameter representing the alpha parameter of the distribution, which influences the shape of the distribution.
distribution_type	The type of distribution.

Value

A list containing:

data	A matrix of generated data.
A	A matrix representing the factor loadings.
D	A diagonal matrix representing the unique variances.

Examples

```
library(MASS)
library(SOPC)
library(sn)
library(matrixcalc)
library(psych)
n <- 100
p <- 10
m <- 5
xi <- 5
omega <- 2
alpha <- 5
distribution_type <- "Skew-Normal Distribution"
X <- SFM(n, p, m, xi, omega, alpha, distribution_type)
```

SOPC *The sparse online principal component can not only process online data sets, but also obtain a sparse solution of online data sets.*

Description

The sparse online principal component can not only process online data sets, but also obtain a sparse solution of online data sets.

Usage

```
SOPC(data, m, gamma, eta)
```

Arguments

data	is a highly correlated online data set
m	is the number of principal component
gamma	is a sparse parameter
eta	is the proportion of online data to total data

Value

Aso,Dso

SPC *The sparse principal component can obtain sparse solutions of the eigenmatrix to better explain the relationship between principal components and original variables.*

Description

The sparse principal component can obtain sparse solutions of the eigenmatrix to better explain the relationship between principal components and original variables.

Usage

```
SPC(data, m, gamma)
```

Arguments

data	is a highly correlated data set
m	is the number of principal component
gamma	is a sparse parameter

Value

As,Ds

wines *Piedmont wines data*

Description

Data refer to chemical properties of 178 specimens of three types of wine produced in the Piedmont region of Italy.

Usage

```
data(wines)
```

Format

A data frame with 178 observations on the following 28 variables.

wine	wine name (categorical, levels: Barbera, Barolo, Grignolino)
alcohol	alcohol percentage (numeric)
sugar	sugar-free extract (numeric)
acidity	fixed acidity (numeric)
tartaric	tartaric acid (numeric)

malic	malic acid (numeric)
uronic	uronic acids (numeric)
pH	pH (numeric)
ash	ash (numeric)
alcal_ash	alcalinity of ash (numeric)
potassium	potassium (numeric)
calcium	calcium (numeric)
magnesium	magnesium (numeric)
phosphate	phosphate (numeric)
cloride	chloride (numeric)
phenols	total phenols (numeric)
flavanoids	flavanoids (numeric)
nonflavanoids	nonflavanoid phenols (numeric)
proanthocyanins	proanthocyanins (numeric)
colour	colour intensity (numeric)
hue	hue (numeric)
OD_dw	OD_{280}/OD_{315} of diluted wines (numeric)
OD_fl	OD_{280}/OD_{315} of flavanoids (numeric)
glycerol	glycerol (numeric)
butanediol	2,3-butanediol (numeric)
nitrogen	total nitrogen (numeric)
proline	proline (numeric)
methanol	methanol (numeric)

Details

The data represent 27 chemical measurements on each of 178 wine specimens belonging to three types of wine produced in the Piedmont region of Italy. The data have been presented and examined by Forina *et al.* (1986) and were freely accessible from the PARVUS web-site until it was active. These data or, more often, a subset of them are now available from various places, including some R packages. The present dataset includes all variables available on the PARVUS repository, which are the variables listed by Forina *et al.* (1986) with the exception of 'Sulphate'. Moreover, it reveals the undocumented fact that the original dataset appears to include also the vintage year; see the final portion of the 'Examples' below.

Source

Forina, M., Lanteri, S. Armanino, C., Casolino, C., Casale, M. and Oliveri, P. V-PARVUS 2008: an extendible package of programs for esplorative data analysis, classification and regression analysis. Dip. Chimica e Tecnologie Farmaceutiche ed Alimentari, Università di Genova, Italia. Web-site (not accessible as of 2014): '`http://www.parvus.unige.it`'

References

Forina M., Armanino C., Castino M. and Ubigli M. (1986). Multivariate data analysis as a discriminating method of the origin of wines. *Vitis* **25**, 189–201.

Examples

```
data(wines)
pairs(wines[,c(2,3,16:18)], col=as.numeric(wines$wine))
#
code <- substr(rownames(wines), 1, 3)
table(wines$wine, code)
#
year <- as.numeric(substr(rownames(wines), 6, 7))
table(wines$wine, year)
# coincides with Table 1(a) of Forina et al. (1986)
```

Index