Package 'TFM'

June 9, 2025

Type Package

Title Sparse Online Principal Component for Truncated Factor Model

Version 0.5.2

Description The Truncated Factor Model is a statistical model designed to handle specific data structures in data analysis. This R package focuses on the Sparse Online Principal Component Estimation method, which is used to calculate data such as the loading matrix and specific variance matrix for truncated data, thereby better explaining the relationship between common factors and original variables. Additionally, the R package also provides other equations for comparison with the Sparse Online Principal Component Estimation method.The philosophy of the package is described in thesis. (2023) <doi:10.1007/s00180-022-01270-z>.

License MIT + file LICENSE

Suggests rmarkdown, psych

Depends R (>= 3.5.0)

RoxygenNote 7.3.2

Encoding UTF-8

Language en-US

Author Beibei Wu [aut],
Guangbao Guo [aut, cre]

Maintainer Guangbao Guo <ggb11111111@163.com>

Imports relliptical, SOPC, MASS, mvtnorm, matrixcalc, corrplot,
ggplot2

NeedsCompilation no

LazyData true

Repository CRAN

Date/Publication 2025-06-09 02:20:02 UTC

Contents

`admission_predict` *Graduate Admissions Prediction Dataset A dataset for predicting postgraduate admission probability, integrating standardized test scores, academic performance, and application - related metrics.*

Description

Graduate Admissions Prediction Dataset

A dataset for predicting postgraduate admission probability, integrating standardized test scores, academic performance, and application - related metrics.

Usage

```
admission_predict
```

Format

A data frame with ‘n‘ observations (rows) and 8 variables:

`GRE.Score` Graduate Record Examinations (GRE) score. A standardized test for graduate admissions (score range varies by exam version; common scales: 260–340 or old 130–170 per section).

TOEFL.Score Test of English as a Foreign Language (TOEFL) score. Measures English proficiency (standard range: 0–120).

University.Rating Target university's academic rating (1–5 scale). A higher score indicates stronger institutional reputation/resources.

SOP Statement of Purpose (SOP) rating (1–5 scale). Reflects the quality of the applicant's research motivation and fit with the program.

LOR Letter of Recommendation (LOR) rating (1–5 scale). Captures the referee's evaluation of the applicant's academic potential.

CGPA Cumulative Grade Point Average (CGPA). Summarizes undergraduate academic performance (scale depends on institution; e.g., 4.0 or 10.0).

Research Research experience indicator (0 = no research, 1 = has research). A binary flag for involvement.

Chance.of.Admit Admission probability (continuous value between 0 and 1). The target variable, with higher values indicating greater admission likelihood.

---

concrete *Concrete Mixture and Compressive Strength Dataset A dataset that includes the proportions of concrete ingredients (such as cement, slag, fly ash, water, superplasticizer, and aggregates) as well as the mechanical and workability properties (slump, flow, and 28-day compressive strength) of high-performance concrete (HPC).*

Description

Concrete Mixture and Compressive Strength Dataset

A dataset that includes the proportions of concrete ingredients (such as cement, slag, fly ash, water, superplasticizer, and aggregates) as well as the mechanical and workability properties (slump, flow, and 28-day compressive strength) of high-performance concrete (HPC).

Usage

```
concrete
```

Format

A data frame with 103 observations and 10 variables:

Cement The mass of Portland cement. Primary binder providing compressive strength through hydration.

Slag The mass of blast furnace slag. Supplementary cementitious material (SCM) that reduces hydration heat and enhances durability.

Fly.ash The mass of fly ash. SCM from coal combustion that improves workability and reduces cement usage.

Water The mass of mixing water. Essential for cement hydration; water-to-binder ratio determines strength and workability.

`SP` The mass of superplasticizer. Chemical admixture that enhances workability while reducing water content.

`Coarse.Aggr.` The mass of coarse aggregate (e.g., crushed stone). Provides structural rigidity and volume stability.

`Fine.Aggr.` The mass of fine aggregate (e.g., river sand). Fills voids between coarse aggregates for optimal particle packing.

`SLUMP.cm.` Concrete slump in cm. Key workability metric; higher values indicate greater flowability.

`FLOW.cm.` Concrete flow diameter in cm. Supplementary workability metric for self-consolidating concretes.

`Compressive.Strength..28.day..Mpa.` 28-day compressive strength in MPa. Critical mechanical property measured after standard curing.

Examples

```
data(concrete)
# Basic summary
summary(concrete)
# Strength vs Cement content
plot(concrete$Cement, concrete$Compressive.Strength..28.day..Mpa.,
     xlab = "Cement (kg/m³)", ylab = "28-day Compressive Strength (MPa)",
     main = "Cement Content vs Strength")
```

`FanPC_TFM` *Apply the FanPC method to the Truncated 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_TFM(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

```
## Not run:
library(SOPC)
library(relliptical)
library(MASS)
results <- FanPC_TFM(data, m, A, D, p)
print(results)

## End(Not run)
```

`GOOG` *Google (GOOG) Stock Price Dataset*

Description

A dataset containing various stock price - related information for Google (GOOG). It can be used for a wide range of financial analyses such as studying price trends, volatility, and relationships with trading volume.

Usage

```
GOOG
```

Format

A data frame with 6 rows (in the provided preview, actual may vary) and 10 variables:

`close` The closing price of the stock on a particular day. This is the price at which the stock last traded during the regular trading session. It's an important metric as it reflects the market's perception of the stock's value at the end of the day.

`high` The highest price at which the stock traded during the day. It shows the peak level of investor interest and buying pressure during that trading period.

`low` The lowest price at which the stock traded during the day. It indicates the lowest level the stock reached, which might be due to selling pressure or other market factors.

open The opening price of the stock when the market started trading for the day. It can set the tone for the day's price movements based on overnight news, global market trends, etc.

volume The number of shares traded during the day. High volume often signals significant market activity and can confirm price trends. For example, a large volume increase with a rising price might indicate strong buying interest.

adjClose The adjusted closing price. This value takes into account corporate actions like stock splits, dividends, and rights offerings. By adjusting for these events, it provides a more accurate picture of the stock's performance over time compared to just the raw closing price.

adjHigh The adjusted highest price for the day. Similar to adjClose, it incorporates corporate action adjustments to give a more precise view of the intraday price high.

adjLow The adjusted lowest price for the day. Incorporates corporate action adjustments to accurately represent the intraday price low.

adjOpen The adjusted opening price for the day. Adjusted for corporate actions to reflect the true starting price in the context of the overall stock history.

adjVolume The adjusted volume. Adjusted for corporate actions (e.g., if there was a stock split, the volume might be adjusted proportionally) to provide a consistent measure over time.

Examples

```
data(GOOG)
# Basic summary statistics
summary(GOOG)
# Plotting the closing price over time
if (requireNamespace("ggplot2", quietly = TRUE)) {
  ggplot2::ggplot(GOOG, ggplot2::aes(x = seq_along(close), y = close)) +
    ggplot2::geom_line() +
    ggplot2::labs(x = "Trading Day", y = "Closing Price")
}
```

GulPC_TFM *Apply the GulPC method to the Truncated 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_TFM(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

```
## Not run:
library(SOPC)
library(relliptical)
library(MASS)
results <- GulPC_TFM(data, m, A, D)
print(results)
## End(Not run)
```

IPC_TFM *Incremental Principal Component Analysis*

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_TFM(x, m, A, D, p)
```

Arguments

x 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

```
## Not run:
library(MASS)
library(relliptical)
library(SOPC)

IPC_MSESigmaA = c()
IPC_MSESigmaD = c()
IPC_LSigmaA = c()
IPC_LSigmaD = c()
data_M = data.frame(n = n, MSEA = IPC_MSESigmaA, MSED = IPC_MSESigmaD,
 LSA = IPC_LSigmaA, LSD = IPC_LSigmaD)
print(data_M)
## End(Not run)
```

OPC_TFM *Apply the OPC method to the Truncated 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_TFM(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

```
## Not run:
library(SOPC)
library(relliptical)
library(MASS)
results <- OPC_TFM(data, m, A, D, p)
print(results)
## End(Not run)
```

`PC1_TFM` *Apply the PC method to the Truncated 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_TFM(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

```
## Not run:
library(SOPC)
library(relliptical)
library(MASS)
results <- PC1_TFM(data, m, A, D)
print(results)
## End(Not run)
```

PC2_TFM *Apply the PC method to the Truncated 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_TFM(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

```
## Not run:
library(SOPC)
library(relliptical)
library(MASS)
results <- PC2_TFM(data, m, A, D)
print(results)
## End(Not run)
```

`PPC1_TFM` *Projected Principal Component Analysis*

Description

This function computes Projected 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_TFM(x, m, A, D, p)
```

Arguments

`x` A matrix of input data.

`m` The number of principal components to extract (integer).

`A` The true factor loadings matrix (matrix).

`D` The true uniquenesses matrix (matrix).

`p` The number of variables (integer).

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

```
## Not run:
library(MASS)
library(relliptical)
library(SOPC)
PPC_MSESigmaA <- c()
PPC_MSESigmaD <- c()
PPC_LSigmaA <- c()
PPC_LSigmaD <- c()
result <- PPC1_TFM(data, m, A, D, p)
print(result)
## End(Not run)
```

PPC2_TFM *Apply the PPC method to the Truncated 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_TFM(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

```
## Not run:
library(SOPC)
library(relliptical)
library(MASS)
results <- PPC2_TFM(data, m, A, D)
print(results)

## End(Not run)
```

`protein` *Data Frame 'protein'*

Description

This is the Protein Data Set from the UCI Machine Learning Repository. It contains information about protein concentration in different samples.

Usage

```
protein
```

Format

A data frame with 45730 rows and 10 columns.

- `SampleID`: A unique identifier for each sample.
- `Protein1`: Concentration of Protein 1.
- `Protein2`: Concentration of Protein 2.
- `Protein3`: Concentration of Protein 3.
- `Protein4`: Concentration of Protein 4.
- `Protein5`: Concentration of Protein 5.
- `Protein6`: Concentration of Protein 6.

- `Protein7`: Concentration of Protein 7.
- `Protein8`: Concentration of Protein 8.
- `Protein9`: Concentration of Protein 9.
- `Protein10`: Concentration of Protein 10.

`real_estate_valuation` *Real Estate Valuation Dataset A dataset containing features related to residential property valuation in New Taipei City, Taiwan, including house age, proximity to transit (MRT), local amenities (convenience stores), geographic coordinates, and the target valuation.*

Description

Real Estate Valuation Dataset

A dataset containing features related to residential property valuation in New Taipei City, Taiwan, including house age, proximity to transit (MRT), local amenities (convenience stores), geographic coordinates, and the target valuation.

Usage

```
real_estate_valuation
```

Format

A data frame with 414 observations (rows) and 6 variables:

`X1 house age` House age (*years*). Older properties typically experience depreciation, which influences valuation.

`X2 distance to the nearest MRT station` Distance to the nearest MRT (Mass Rapid Transit) station (*meters*). Closer transit access generally increases property value.

`X3 number of convenience stores` Number of nearby convenience stores. More amenities correlate with higher desirability and valuation.

`X4 latitude` Latitude (*decimal degrees*). Geographic coordinate for spatial analysis of location-based value trends.

`X5 longitude` Longitude (*decimal degrees*). Geographic coordinate for spatial analysis of location-based value trends.

`Y` Real estate valuation (*10,000 New Taiwan Dollars per ping*). The target variable representing property value. (*Note*: 1 ping = 3.3058 m², a local unit of area in Taiwan.)

review *Travel Review Dataset*

Description

This dataset contains travel reviews from TripAdvisor.com, covering destinations in 10 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

```
data(review)
```

Format

A data frame with multiple rows and 10 columns.

`1` Unique identifier for each user (Categorical)

`2` Average user feedback on art galleries

`3` Average user feedback on dance clubs

`4` Average user feedback on juice bars

`5` Average user feedback on restaurants

`6` Average user feedback on museums

`7` Average user feedback on resorts

`8` Average user feedback on parks and picnic spots

`9` Average user feedback on beaches

`10` Average user feedback on theaters

Details

The dataset is populated by crawling TripAdvisor.com and includes reviews on destinations in 10 categories across East Asia. Each traveler's rating is mapped as follows:

- Excellent (4)
- Very Good (3)
- Average (2)
- Poor (1)
- 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.

Source

UCI Machine Learning Repository

Examples

```
data(review)
head(review)
review$`1`  # User IDs
mean(review$`5`)  # Average rating for restaurants
```

---

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).

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.

Examples

```
data(riboflavin)
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.

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.

Examples

```
data(riboflavinv100)
print(dim(riboflavinv100$x))
print(length(riboflavinv100$y))
```

SAPC_TFM *Stochastic Approximation Principal Component Analysis*

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_TFM(x, m, A, D, p)
```

Arguments

x 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

```
## Not run:
library(MASS)
library(relliptical)
library(SOPC)
SAPC_MSESigmaA <- c()
SAPC_MSESigmaD <- c()
SAPC_LSigmaA <- c()
SAPC_LSigmaD <- c()
result <- SAPC_TFM(data, m = m, A = A, D = D, p = p)
print(result)
## End(Not run)
```

SOPC_TFM *Sparse Online Principal Component Analysis*

Description

This function calculates various metrics for the Sparse Online Principal Component Analysis (SOPC) method. It estimates the factor loadings and uniquenesses while calculating mean squared errors and loss metrics for comparison with true values. Additionally, it computes the proportion of zero factor loadings in the estimated loadings matrix.

Usage

```
SOPC_TFM(data, m, p, gamma, eta, A, D)
```

Arguments

data	The data used in the SOPC analysis.
m	the number of common factors
p	the number of variables
gamma	Tuning parameter for the sparseness of the loadings matrix.
eta	Tuning parameter for the sparseness of the uniquenesses matrix.
A	The true A matrix.
D	The true D matrix.

Value

A list of metrics including:

Aso	Estimated factor loadings matrix obtained from the SOPC analysis.
Dso	Estimated uniquenesses vector obtained from the SOPC analysis.
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), indicating the sparsity of the loadings.

Examples

```
## Not run:
library(MASS)
library(relliptical)
library(SOPC)
SOPC_MSEA <- c()
SOPC_MSED <- c()
SOPC_LSA <- c()
SOPC_LSD <- c()
SOPC_TAUA <- c()
result <- SOPC_TFM(data, m = m, A = A, D = D, p = p)
print(result)
## End(Not run)
```

SPC_TFM *Sparse Principal Component Analysis*

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. Additionally, it computes the proportion of zero factor loadings.

Usage

```
SPC_TFM(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

```
## Not run:
library(MASS)
library(relliptical)
library(SOPC)
SPC_MSESigmaA <- c()
SPC_MSESigmaD <- c()
SPC_LSigmaA <- c()
SPC_LSigmaD <- c()
SPC_tau <- c()
result <- SPC_TFM(data, A, D, m, p)
print(result)
## End(Not run)
```

`taxi_trip_pricing` *Taxi Trip Pricing Dataset*

Description

A dataset containing various factors related to taxi trips and their corresponding prices.

Usage

```
taxi_trip_pricing
```

Format

A data frame with ‘n‘ observations (rows, where ‘n‘ is the number of taxi trips) and 11 variables:

`Trip_Distance_km` Trip distance in kilometers. This variable measures how far the taxi has traveled.

`Time_of_Day` A categorization of the time of the day (e.g., 1 might represent morning, 2 afternoon, etc.). It can potentially affect pricing due to demand patterns.

`Day_of_Week` The day of the week (0 - 6, where 0 could represent Sunday). Weekend vs. weekday trips might have different pricing considerations.

`Passenger_Count` Number of passengers in the taxi. It could influence the pricing structure in some taxi systems.

`Traffic_Conditions` A measure of traffic conditions (e.g., 1 for light traffic, 4 for heavy traffic). Traffic can impact trip duration and thus price.

`Weather` A classification of weather conditions (e.g., 1 for clear, 3 for rainy). Weather might have an impact on demand and thus pricing.

`Base_Fare` The base fare amount for the taxi trip. This is a fixed component of the price.

`Per_Km_Rate` The rate charged per kilometer traveled.

`Per_Minute_Rate` The rate charged per minute of the trip (usually applicable when the taxi is idling or in slow - moving traffic).

`Trip_Duration_Minutes` The duration of the trip in minutes.

`Trip_Price` The final price of the taxi trip.

Examples

```
data(taxi_trip_pricing)
summary(taxi_trip_pricing)
if (requireNamespace("ggplot2", quietly = TRUE)) {
  ggplot2::ggplot(taxi_trip_pricing, ggplot2::aes(x = Trip_Distance_km, y = Trip_Price)) +
    ggplot2::geom_point() +
    ggplot2::labs(x = "Trip Distance (km)", y = "Trip Price")
}
```

TFM *The TFM function is to generate Truncated factor model data.*

Description

The TFM function generates truncated factor model data supporting various distribution types for related analyses using multiple methods.

Usage

```
TFM(n, mu, sigma, lower, upper, distribution_type)
```

Arguments

n	Total number of observations.
mu	The mean of the distribution.
sigma	The parameter of the distribution.
lower	The lower bound of the interval.
upper	The upper bound of the interval.
distribution_type	String specifying the distribution type to use.

Value

A list containing:

X A matrix of generated truncated factor model data based on the specified distribution type. Each row corresponds to an observation, and each column corresponds to a variable.

Examples

```
library(relliptical)
set.seed(123)
mu <- c(0, 1)
n <- 100
sigma <- matrix(c(1, 0.70, 0.70, 3), 2, 2)
lower <- c(-2, -3)
```

```
upper <- c(3, 3)
distribution_type <- "truncated_normal"
X <- TFM(n, mu, sigma, lower, upper, distribution_type)
```

ttest.TFM *T-test for Truncated Factor Model*

Description

This function performs a simple t-test for each variable in the dataset of a truncated factor model and calculates the False Discovery Rate (FDR) and power.

Usage

```
ttest.TFM(X, p, alpha = 0.05)
```

Arguments

`X`	A matrix or data frame of simulated or observed data from a truncated factor model.
`p`	The number of variables (columns) in the dataset.
`alpha`	The significance level for the t-test.

Value

A list containing:

`FDR`	The False Discovery Rate calculated from the rejected hypotheses.
`Power`	The power of the test, representing the proportion of true positives among the non-zero hypotheses.
`pValues`	A numeric vector of p-values obtained from the t-tests for each variable.
`RejectedHypotheses`	A logical vector indicating which hypotheses were rejected based on the specified significance level.

Examples

```
library(MASS)
library(mvtnorm)
set.seed(100)
p <- 400
n <- 120
K <- 5
true_non_zero <- 100
B <- matrix(rnorm(p * K), nrow = p, ncol = K)
FX <- MASS::mvrnorm(n, rep(0, K), diag(K))
U <- mvtnorm::rmvt(n, df = 3, sigma = diag(p))
mu <- c(rep(1, true_non_zero), rep(0, p - true_non_zero))
```

```
X <- rep(1, n) %*% t(mu) + FX %*% t(B) + U  # The observed data
results <- ttest.TFM(X, p, alpha = 0.05)
print(results)
```

`winequality.white` *White Wine Quality Dataset A dataset containing physicochemical properties and sensory quality ratings of Portuguese "Vinho Verde" white wine samples.*

Description

White Wine Quality Dataset

A dataset containing physicochemical properties and sensory quality ratings of Portuguese "Vinho Verde" white wine samples.

Usage

```
winequality.white
```

Format

An object of class `data.frame` with 4898 rows and 12 columns.

Examples

```
## Not run:
data(winequality.white)
summary(winequality.white)
hist(winequality.white$quality,
     main = "White Wine Quality Distribution",
     xlab = "Quality Rating",
     col = "lightblue")
boxplot(fixed.acidity ~ quality,
        data = winequality.white,
        xlab = "Quality Rating",
        ylab = "Fixed Acidity (g/dm³)",
        col = "lightgreen")
if (requireNamespace("corrplot", quietly = TRUE)) {
  key_features <- winequality.white[, c("fixed.acidity", "volatile.acidity",
                                        "citric.acid", "alcohol", "quality")]
  corr_matrix <- cor(key_features, use = "complete.obs")
  corrplot::corrplot(corr_matrix, method = "color", type = "upper",
                     order = "hclust", tl.col = "black")
}

## End(Not run)
```

`yacht_hydrodynamics` *Yacht Hydrodynamics (Residuary Resistance) Dataset A dataset for predicting the* **residuary resistance** *of sailing yachts during early design stages. It contains hull geometry, operational parameters, and experimental resistance measurements from the Delft Ship Hydromechanics Laboratory.*

Description

Yacht Hydrodynamics (Residuary Resistance) Dataset

A dataset for predicting the **residuary resistance** of sailing yachts during early design stages. It contains hull geometry, operational parameters, and experimental resistance measurements from the Delft Ship Hydromechanics Laboratory.

Usage

```
yacht_hydrodynamics
```

Format

A data frame with 308 observations (rows) and 7 variables:

- `V1` Hull length (*typical unit: meters*). A core geometric parameter that shapes hydrodynamic performance.
- `V2` Hull beam (width) (*typical unit: meters*). Influences the yacht's stability and resistance properties.
- `V3` Hull draft (*typical unit: meters*). The depth of the hull beneath the waterline, a vital factor for hydrodynamics.
- `V4` Displacement (*typical unit: kilograms or metric tons*). The total mass of the yacht (hull + payload), a key design limitation.
- `V5` Trim angle (*typical unit: degrees*). The longitudinal tilt of the hull, which has an impact on resistance and speed.
- `V6` Boat velocity (*typical unit: m/s or knots*). The speed of the yacht during resistance testing.
- `V7` Residuary resistance (*typical unit: Newtons*). The target variable, representing resistance from wave formation and hull friction (air resistance not included).

Examples

```
data(yacht_hydrodynamics)
summary(yacht_hydrodynamics)
plot(
  x = yacht_hydrodynamics$V6,
  y = yacht_hydrodynamics$V7,
  xlab = "Boat Velocity",
  ylab = "Residuary Resistance",
  main = "Velocity vs Residuary Resistance"
)
if (requireNamespace("corrplot", quietly = TRUE)) {
  yacht_corr <- cor(yacht_hydrodynamics, use = "complete.obs")
  corrplot::corrplot(yacht_corr, method = "color", type = "upper", order = "hclust")
}
model <- lm(V7 ~ V6 + V1, data = yacht_hydrodynamics)
summary(model)
```

Index