Package 'DTSR'

June 20, 2026

Version 0.2.2

Date 2026-06-12

Type Package

Title Distributed Trimmed Scores Regression for Handling Missing Data

Author Guangbao Guo [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0002-4115-6218>),
Ruiling Niu [aut]

Maintainer Guangbao Guo <ggb11111111@163.com>

Description Provides functions for handling missing data using Distributed Trimmed Scores Regression and other imputation methods. It includes facilities for data imputation, evaluation metrics, and clustering analysis. It is designed to work in distributed computing environments to handle large datasets efficiently. The philosophy of the package is described in Guo G. (2024) <doi:10.1080/03610918.2022.2091779>.

License GPL-3

Encoding UTF-8

Depends R (>= 3.5.0)

Repository CRAN

Imports stats, MASS, mvdalab, VIM, cluster

RoxygenNote 7.3.3

NeedsCompilation no

Date/Publication 2026-06-20 13:30:07 UTC

Contents

`abalone` *Abalone Data*

Description

The Abalone data consist of data from 4177 abalones. The data consist of measurements of the type (male, female and infant), the longest shell measurement, the diameter, height and several weights (whole, shucked, viscera and shell). The outcome is the number of rings. The age of the abalone is the number of rings plus 1.5.

The data are taken from the UCI database.

Usage

```
data(abalone)
```

Value

`abalone` a data frame with 4177 rows and 9 columns

Examples

```
data(abalone)
```

`DEM` *Distributed EM Imputation (DEM) for Handling Missing Data*

Description

This function performs DEM to handle missing data by dividing the dataset into D blocks, applying the EM imputation method within each block, and then combining the results. It calculates various evaluation metrics including RMSE, MMAE, RRE, and Consistency Proportion Index (CPP) using different hierarchical clustering methods.

Usage

```
DEM(data0, data.sample, data.copy, mr, km, D)
```

Arguments

data0	The original dataset containing the response variable and features.
data.sample	The dataset used for sampling, which may contain missing values.
data.copy	A copy of the original dataset, used for comparison or validation.
mr	Indices of the rows with missing values that need to be predicted.
km	The number of clusters for k-means clustering.
D	The number of blocks to divide the data into.

Value

A list containing:

XDEM	The imputed dataset.
RMSEDEM	The Root Mean Squared Error.
MAEDEM	The Mean Absolute Error.
REDEM	The Relative Eelative Error.
GCVDEM	The DEM Imputation for Generalized Cross-Validation.
timeDEM	The DEM algorithm execution time.

See Also

EM for the original EM function.

Examples

```
# Create a sample dataset with missing values
set.seed(123)
n <- 100
p <- 5
data.sample <- matrix(rnorm(n * p), nrow = n)
data.sample[sample(1:(n*p), 20)] <- NA
data.copy <- data.sample
data0 <- data.frame(data.sample, response = rnorm(n))
mr <- sample(1:n, 10)  # Sample rows for evaluation
km <- 3  # Number of clusters
D <- 2  # Number of blocks
# Perform DEM imputation
result <- DEM(data0, data.sample, data.copy, mr, km, D)
# Print the results
print(result$XDEM)
```

DRPCA *Distributed Robust Principal Component Analysis (DRPCA) for Handling Missing Data*

Description

This function performs DRPCA to handle missing data by dividing the dataset into D blocks, applying the Robust Principal Component Analysis (RPCA) method to each block, and then combining the results. It calculates various evaluation metrics including RMSE, MMAE, RRE, and Generalized Cross-Validation (GCV) using different hierarchical clustering methods.

Usage

```
DRPCA(data0, data.sample, data.copy, mr, km, D)
```

Arguments

`data0`	The original dataset containing the response variable and features.
`data.sample`	The dataset used for sampling, which may contain missing values.
`data.copy`	A copy of the original dataset, used for comparison or validation.
`mr`	Indices of the rows with missing values that need to be predicted.
`km`	The number of clusters for k-means clustering.
`D`	The number of blocks to divide the data into.

Value

A list containing:

`XDRPCA`	The imputed dataset.
`RMSEDRPCA`	The Root Mean Squared Error.
`MAEDRPCA`	The Mean Absolute Error.
`REDRPCA`	The Relative Eelative Error.
`GCVDRPCA`	Distributed DRPCA Imputation for Generalized Cross-Validation.
`timeDRPCA`	The DRPCA algorithm execution time.

See Also

`RPCA` for the original RPCA function.

Examples

```
# Create a sample dataset with missing values
set.seed(123)
n <- 100
p <- 10
D <- 2
data.sample <- matrix(rnorm(n * p), nrow = n)

# Randomly select 10 rows to have missing values
mr <- sort(sample(1:n, 10))
missing_cols <- sample(1:p, 2)
data.copy <- data.sample
for (i in mr) {
  data.sample[i, missing_cols] <- NA
}

data0 <- data.frame(data.sample, response = rnorm(n))
km <- 3
result <- DRPCA(data0, data.sample, data.copy, mr, km, D)
print(result$XDRPCA)
```

DTSR *Distributed Trimmed Scores Regression (DTSR) for Handling Missing Data*

Description

This function performs DTSR to handle missing data by dividing the dataset into D blocks, applying the Trimmed Scores Regression (TSR) method to each block, and then combining the results. It calculates various evaluation metrics including RMSE, MMAE, RRE, and Consistency Proportion Index (CPP) using different hierarchical clustering methods.

Usage

```
DTSR(data0, data.sample, data.copy, mr, km, D)
```

Arguments

`data0`	The original dataset containing the response variable and features.
`data.sample`	The dataset used for sampling, which may contain missing values.
`data.copy`	A copy of the original dataset, used for comparison or validation.
`mr`	Indices of the rows with missing values that need to be predicted.
`km`	The number of clusters for k-means clustering.
`D`	The number of blocks to divide the data into.

Value

A list containing:

XDTSR	The imputed dataset.
RMSEDTSR	The Root Mean Squared Error.
MAEDTSR	The Mean Absolute Error.
REDTSR	The Relative Eelative Error.
GCVDTSR	The DTSR for Generalized Cross-Validation.
timeDTSR	The DTSR algorithm execution time.

See Also

TSR for the original TSR function.

Examples

```
# Create a sample matrix with random values and introduce missing values
set.seed(123)
n <- 100
p <- 10
D <- 2
data.sample <- matrix(rnorm(n * p), nrow = n)
data.sample[sample(1:(n-10), (p-2))] <- NA
data.copy <- data.sample
data0 <- data.frame(data.sample, response = rnorm(n))
mr <- sample(1:n, 10)  # Sample rows for evaluation
km <- 3  # Number of clusters
# Perform DTSR imputation
result <- DTSR(data0, data.sample, data.copy, mr, km,D)
# Print the results
print(result$XDTSR)
```

EM *Expectation-Maximization Imputation with Evaluation Metrics*

Description

This function performs Expectation-Maximization (EM) imputation on a dataset with missing values. It uses the ‘imputeEM‘ function from the ‘mvdalab‘ package to estimate the missing values. The function also calculates various evaluation metrics including RMSE, MMAE, and RRE. Additionally, it performs k-means and hierarchical clustering to assess the quality of the imputation.

Usage

```
EM(data0, data.sample, data.copy, mr, km)
```

Arguments

data0	The original dataset containing the response variable and features.
data.sample	The dataset used for sampling, which may contain missing values.
data.copy	A copy of the original dataset, used for comparison or validation.
mr	Indices of the rows with missing values that need to be predicted.
km	The number of clusters for k-means clustering.

Value

A list containing:

Xnew	The imputed dataset.
RMSE	The Root Mean Squared Error.
MMAE	The Mean Absolute Error.
RRE	The Relative Eelative Error.
CPP1	The K-means clustering Consistency Proportion Index.
CPP2	The Hierarchical Clustering Complete Linkage Consistency Proportion Index.
CPP3	The Hierarchical Clustering Single Linkage Consistency Proportion Index.
CPP4	The Hierarchical Clustering Average Linkage Consistency Proportion Index.
CPP5	The Hierarchical Clustering Centroid linkage Consistency Proportion Index.
CPP6	The Hierarchical Clustering Median Linkage Consistency Proportion Index.
CPP7	The Hierarchical Clustering Ward's Method Consistency Proportion Index.
timeEM	The EM algorithm execution time.

Examples

```
# Create a sample matrix with random values and introduce missing values
set.seed(123)
n <- 100
p <- 5
data.sample <- matrix(rnorm(n * p), nrow = n)
data.sample[sample(1:(n*p), 20)] <- NA
data.copy <- data.sample
data0 <- data.frame(data.sample, response = rnorm(n))
mr <- sample(1:n, 10)  # Sample rows for evaluation
km <- 3  # Number of clusters
# Perform EM imputation
result <- EM(data0, data.sample, data.copy, mr, km)
# Print the results
print(result$RMSE)
print(result$MMAE)
print(result$RRE)
print(result$CPP1)
print(result$Xnew)
```

Frogs *Frogs Data*

Description

The Frogs data consist of Mel-Frequency Cepstral Coefficients (MFCCs) extracted from syllables of anuran (frog) calls. The dataset includes labels for family, genus, and species of the frogs.

Usage

```
data(Frogs)
```

Value

Frogs	a data frame with 7195 rows and 22 columns

Examples

```
data(Frogs)
head(Frogs)
```

IndexCPP *Calculate the Consistency Proportion Index (CPP)*

Description

This function calculates the Consistency Proportion Index (CPP), a measure of the consistency of clustering results. The CPP is calculated by determining the most common cluster assignment for each group and then computing the proportion of cases that are assigned to these clusters.

Usage

```
IndexCPP(I)
```

Arguments

I	A matrix where each row represents a case and each column represents a cluster assignment. The last column should indicate the group membership (1, 2, or 3).

Value

A list containing:

ICPP	The Consistency Proportion Index.

Examples

```
# Example usage
set.seed(123)
n <- 100
values1 <- sample(1:3, 30, replace = TRUE)
values2 <- sample(1:3, 30, replace = TRUE) + 1
values3 <- sample(1:3, 40, replace = TRUE) + 2
values <- c(values1, values2, values3)
categories <- c(rep(1, 30), rep(2, 30), rep(3, 40))
I <- cbind(1:n, values, categories)
CPP <- IndexCPP(I)
print(CPP)
```

KNN *This function performs imputation using the K-Nearest Neighbors (KNN) algorithm and calculates various evaluation metrics including RMSE, MMAE, RRE, and Consistency Proportion Index (CPP) using different hierarchical clustering methods. It also records the execution time of the process.*

Description

This function performs imputation using the K-Nearest Neighbors (KNN) algorithm and calculates various evaluation metrics including RMSE, MMAE, RRE, and Consistency Proportion Index (CPP) using different hierarchical clustering methods. It also records the execution time of the process.

Usage

```
KNN(data0, data.sample, data.copy, mr, km)
```

Arguments

data0	The original dataset containing the response variable and features.
data.sample	The dataset used for sampling, which may contain missing values.
data.copy	A copy of the original dataset, used for comparison or validation.
mr	Indices of the rows with missing values that need to be predicted.
km	The number of clusters for k-means clustering.

Value

A list containing:

Xnew	The imputed dataset.
RMSE	The Root Mean Squared Error.

MMAE	The Mean Absolute Error.
RRE	The Relative Eelative Error.
CPP1	The K-means clustering Consistency Proportion Index.
CPP2	The Hierarchical Clustering Complete Linkage Consistency Proportion Index.
CPP3	The Hierarchical Clustering Single Linkage Consistency Proportion Index.
CPP4	The Hierarchical Clustering Average Linkage Consistency Proportion Index.
CPP5	The Hierarchical Clustering Centroid linkage Consistency Proportion Index.
CPP6	The Hierarchical Clustering Median Linkage Consistency Proportion Index.
CPP7	The Hierarchical Clustering Ward's Method Consistency Proportion Index.
timeKNN	The KNN algorithm execution time.

`meanImpute` *Mean Imputation with Evaluation Metrics*

Description

This function performs mean imputation on a dataset with missing values. It replaces missing values with the column means and calculates various evaluation metrics including RMSE, MMAE, and RRE. Additionally, it performs k-means and hierarchical clustering to assess the quality of the imputation.

Usage

```
meanImpute(data0, data.sample, data.copy, mr, km)
```

Arguments

data0	The original dataset containing the response variable and features.
data.sample	The dataset used for sampling, which may contain missing values.
data.copy	A copy of the original dataset, used for comparison or validation.
mr	Indices of the rows with missing values that need to be predicted.
km	The number of clusters for k-means clustering.

Value

A list containing:

Xnew	The imputed dataset.
RMSE	The Root Mean Squared Error.
MMAE	The Mean Absolute Error.
RRE	The Relative Eelative Error.
CPP1	The K-means clustering Consistency Proportion Index.

CPP2	The Hierarchical Clustering Complete Linkage Consistency Proportion Index.
CPP3	The Hierarchical Clustering Single Linkage Consistency Proportion Index.
CPP4	The Hierarchical Clustering Average Linkage Consistency Proportion Index.
CPP5	The Hierarchical Clustering Centroid linkage Consistency Proportion Index.
CPP6	The Hierarchical Clustering Median Linkage Consistency Proportion Index.
CPP7	The Hierarchical Clustering Ward's Method Consistency Proportion Index.
timemean	The mean algorithm execution time.

See Also

kmeans in the stats package for more information on k-means clustering.

hclust in the stats package for more information on hierarchical clustering.

Examples

```
# Create a sample matrix with random values and introduce missing values
set.seed(123)
n <- 100
p <- 5
data.sample <- matrix(rnorm(n * p), nrow = n)
data.sample[sample(1:(n*p), 20)] <- NA
data.copy <- data.sample
data0 <- data.frame(data.sample, response = rnorm(n))
mr <- sample(1:n, 10)  # Sample rows for evaluation
km <- 3  # Number of clusters
# Perform mean imputation
result <- meanImpute(data0, data.sample, data.copy, mr, km)
# Print the results
print(result$RMSE)
print(result$MMAE)
print(result$RRE)
print(result$CPP1)
print(result$Xnew)
```

MLPCA *Multilinear Principal Component Analysis with Missing Data*

Description

This function performs Multilinear Principal Component Analysis (MLPCA) to handle missing data by imputing the missing values based on the correlation structure within the data. It also calculates the RMSE and Consistency Proportion Index (CPP) using different hierarchical clustering methods.

Usage

```
MLPCA(data0, data.sample, data.copy, mr, km)
```

Arguments

data0	The original dataset containing the response variable and features.
data.sample	The dataset used for sampling, which may contain missing values.
data.copy	A copy of the original dataset, used for comparison or validation.
mr	Indices of the rows with missing values that need to be predicted.
km	The number of clusters for k-means clustering.

Value

A list containing:

Xnew	The imputed dataset.
RMSE	The Root Mean Squared Error.
CPP1	The K-means clustering Consistency Proportion Index.
CPP2	The Hierarchical Clustering Complete Linkage Consistency Proportion Index.
CPP3	The Hierarchical Clustering Single Linkage Consistency Proportion Index.
CPP4	The Hierarchical Clustering Average Linkage Consistency Proportion Index.
CPP5	The Hierarchical Clustering Centroid linkage Consistency Proportion Index.
CPP6	The Hierarchical Clustering Median Linkage Consistency Proportion Index.
CPP7	The Hierarchical Clustering Ward's Method Consistency Proportion Index.
timeKNN	The MLPCA algorithm execution time.

See Also

princomp and svd for more information on PCA and SVD.

Examples

```
# Create a sample matrix with random values and introduce missing values
set.seed(123)
n <- 100
p <- 5
data.sample <- matrix(rnorm(n * p), nrow = n)
data.sample[sample(1:(n*p), 20)] <- NA
data.copy <- data.sample
data0 <- data.frame(data.sample, response = rnorm(n))
mr <- sample(1:n, 10)  # Sample rows for evaluation
km <- 3  # Number of clusters
# Perform MLPCA imputation
result <- MLPCA(data0, data.sample, data.copy, mr, km)
# Print the results
print(result$RMSE)
print(result$CPP1)
print(result$Xnew)
```

NIPALS *NIPALS Algorithm with RPCA and Clustering*

Description

This function performs the NIPALS (Nonlinear Iterative Partial Least Squares) algorithm to handle missing data by imputing the missing values based on the correlation structure within the data. It also calculates the RMSE and Consistency Proportion Index (CPP) using different hierarchical clustering methods.

Usage

```
NIPALS(data0, data.sample, data.copy, mr, km)
```

Arguments

`data0`	The original dataset containing the response variable and features.
`data.sample`	The dataset used for sampling, which may contain missing values.
`data.copy`	A copy of the original dataset, used for comparison or validation.
`mr`	Indices of the rows with missing values that need to be predicted.
`km`	The number of clusters for k-means clustering.

Value

A list containing:

`Xnew`	The imputed dataset.
`RMSE`	The Root Mean Squared Error.
`CPP1`	The K-means clustering Consistency Proportion Index.
`CPP2`	The Hierarchical Clustering Complete Linkage Consistency Proportion Index.
`CPP3`	The Hierarchical Clustering Single Linkage Consistency Proportion Index.
`CPP4`	The Hierarchical Clustering Average Linkage Consistency Proportion Index.
`CPP5`	The Hierarchical Clustering Centroid linkage Consistency Proportion Index.
`CPP6`	The Hierarchical Clustering Median Linkage Consistency Proportion Index.
`CPP7`	The Hierarchical Clustering Ward's Method Consistency Proportion Index.
`timeNIPALS`	The NIPALS algorithm execution time.

See Also

`princomp` and `svd` for more information on PCA and SVD.

Examples

```
# Create a sample matrix with random values and introduce missing values
set.seed(123)
n <- 100
p <- 5
data.sample <- matrix(rnorm(n * p), nrow = n)
data.sample[sample(1:(n*p), 20)] <- NA
data.copy <- data.sample
data0 <- data.frame(data.sample, response = rnorm(n))
mr <- sample(1:n, 10)  # Sample rows for evaluation
km <- 3  # Number of clusters
# Perform NIPALS imputation
result <- NIPALS(data0, data.sample, data.copy, mr, km)
# Print the results
print(result$RMSE)
print(result$CPP1)
print(result$Xnew)
```

RPCA *Robust Principal Component Analysis with Missing Data*

Description

This function performs Robust Principal Component Analysis (RPCA) to handle missing data by imputing the missing values based on the correlation structure within the data. It also calculates various evaluation metrics including RMSE, MMAE, RRE, and Consistency Proportion Index (CPP) using different hierarchical clustering methods.

Usage

```
RPCA(data0, data.sample, data.copy, mr, km)
```

Arguments

`data0`	The original dataset containing the response variable and features.
`data.sample`	The dataset used for sampling, which may contain missing values.
`data.copy`	A copy of the original dataset, used for comparison or validation.
`mr`	Indices of the rows with missing values that need to be predicted.
`km`	The number of clusters for k-means clustering.

Value

A list containing:

`Xnew`	The imputed dataset.
`RMSE`	The Root Mean Squared Error.

MMAE	The Mean Absolute Error.
RRE	The Relative Relative Error.
CPP1	The K-means clustering Consistency Proportion Index.
CPP2	The Hierarchical Clustering Complete Linkage Consistency Proportion Index.
CPP3	The Hierarchical Clustering Single Linkage Consistency Proportion Index.
CPP4	The Hierarchical Clustering Average Linkage Consistency Proportion Index.
CPP5	The Hierarchical Clustering Centroid linkage Consistency Proportion Index.
CPP6	The Hierarchical Clustering Median Linkage Consistency Proportion Index.
CPP7	The Hierarchical Clustering Ward's Method Consistency Proportion Index.
timeRPCA	The RPCA algorithm execution time.

See Also

princomp and svd for more information on PCA and SVD.

Examples

```
# Create a sample matrix with random values and introduce missing values
set.seed(123)
n <- 100
p <- 5
data.sample <- matrix(rnorm(n * p), nrow = n)
data.sample[sample(1:(n*p), 20)] <- NA
data.copy <- data.sample
data0 <- data.frame(data.sample, response = rnorm(n))
mr <- sample(1:n, 10)  # Sample rows for evaluation
km <- 3  # Number of clusters
# Perform RPCA imputation
result <- RPCA(data0, data.sample, data.copy, mr, km)
# Print the results
print(result$RMSE)
print(result$MMAE)
print(result$RRE)
print(result$CPP1)
print(result$Xnew)
```

SVD *This function performs imputation using Singular Value Decomposition (SVD) and calculates various evaluation metrics including RMSE, MMAE, RRE, and Consistency Proportion Index (CPP) using different hierarchical clustering methods.*

Description

This function performs imputation using Singular Value Decomposition (SVD) and calculates various evaluation metrics including RMSE, MMAE, RRE, and Consistency Proportion Index (CPP) using different hierarchical clustering methods.

Usage

```
SVD(data0, data.sample, data.copy, mr, km)
```

Arguments

data0	The original dataset containing the response variable and features.
data.sample	The dataset used for sampling, which may contain missing values.
data.copy	A copy of the original dataset, used for comparison or validation.
mr	Indices of the rows with missing values that need to be predicted.
km	The number of clusters for k-means clustering.

Value

A list containing:

Xnew	The imputed dataset.
RMSE	The Root Mean Squared Error.
MMAE	The Mean Absolute Error.
RRE	The Relative Eelative Error.
CPP1	The K-means clustering Consistency Proportion Index.
CPP2	The Hierarchical Clustering Complete Linkage Consistency Proportion Index.
CPP3	The Hierarchical Clustering Single Linkage Consistency Proportion Index.
CPP4	The Hierarchical Clustering Average Linkage Consistency Proportion Index.
CPP5	The Hierarchical Clustering Centroid linkage Consistency Proportion Index.
CPP6	The Hierarchical Clustering Median Linkage Consistency Proportion Index.
CPP7	The Hierarchical Clustering Ward's Method Consistency Proportion Index.
timeSVD	The SVD algorithm execution time.

See Also

princomp and svd for more information on PCA and SVD.

SVDImpute *Improved SVD Imputation*

Description

This function performs imputation using Singular Value Decomposition (SVD) with iterative refinement. It begins by filling missing values with the mean of their respective columns. Then, it computes a low-rank (k) approximation of the data matrix. Using this approximation, it refills the missing values. This process of recomputing the rank-k approximation with the newly imputed values and refilling the missing data is repeated for a specified number of iterations, 'num.iters'.

Usage

```
SVDImpute(x, k, num.iters = 10, verbose = TRUE)
```

Arguments

x	A data frame or matrix where each row represents a different record.
k	The rank-k approximation to use for the data matrix.
num.iters	The number of times to compute the rank-k approximation and impute the missing data.
verbose	If TRUE, print status updates during the process.

Value

A list containing:

data.matrix	The imputed matrix with missing values filled.

Examples

```
# Create a sample matrix with random values and introduce missing values
x = matrix(rnorm(100), 10, 10)
x[x > 1] = NA

# Perform SVD imputation
imputed_x = SVDImpute(x, 3)

# Print the imputed matrix
print(imputed_x)
```

TSR *Trimmed Scores Regression with Missing Data*

Description

This function performs Trimmed Scores Regression (TSR) to handle missing data by imputing the missing values based on the correlation structure within the data. It also calculates various evaluation metrics including RMSE, MMAE, RRE, and Consistency Proportion Index (CPP) using different hierarchical clustering methods.

Usage

```
TSR(data0, data.sample, data.copy, mr, km)
```

Arguments

data0	The original dataset containing the response variable and features.
data.sample	The dataset used for sampling, which may contain missing values.
data.copy	A copy of the original dataset, used for comparison or validation.
mr	Indices of the rows with missing values that need to be predicted.
km	The number of clusters for k-means clustering.

Value

A list containing:

Xnew	The imputed dataset.
RMSE	The Root Mean Squared Error.
MMAE	The Mean Absolute Error.
RRE	The Relative Relative Error.
CPP1	The K-means clustering Consistency Proportion Index.
CPP2	The Hierarchical Clustering Complete Linkage Consistency Proportion Index.
CPP3	The Hierarchical Clustering Single Linkage Consistency Proportion Index.
CPP4	The Hierarchical Clustering Average Linkage Consistency Proportion Index.
CPP5	The Hierarchical Clustering Centroid linkage Consistency Proportion Index.
CPP6	The Hierarchical Clustering Median Linkage Consistency Proportion Index.
CPP7	The Hierarchical Clustering Ward's Method Consistency Proportion Index.
timeTSR	The TSR algorithm execution time.

See Also

princomp and svd for more information on PCA and SVD.

Examples

```
# Create a sample matrix with random values and introduce missing values
set.seed(123)
n <- 100
p <- 5
data.sample <- matrix(rnorm(n * p), nrow = n)
data.sample[sample(1:(n*p), 20)] <- NA
data.copy <- data.sample
data0 <- data.frame(data.sample, response = rnorm(n))
mr <- sample(1:n, 10)  # Sample rows for evaluation
km <- 3  # Number of clusters
# Perform TSR imputation
result <- TSR(data0, data.sample, data.copy, mr, km)
# Print the results
print(result$RMSE)
print(result$MMAE)
print(result$RRE)
```

```
print(result$CPP1)
print(result$Xnew)
```

Index