Package 'OPCreg'

May 7, 2026

Title Online Principal Component Regression for Online Datasets

Date 2025-06-04

Version 3.0.0

Description
The online principal component regression method can process the online data set. 'OPCreg' implements the online principal component regression method, which is specifically designed to process online datasets efficiently. This method is particularly useful for handling large-scale, streaming data where traditional batch processing methods may be computationally infeasible.The philosophy of the package is described in 'Guo' (2025) <doi:10.1016/j.physa.2024.130308>.

RoxygenNote 7.3.2

Imports MASS, stats, Matrix

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

LazyData true

NeedsCompilation no

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

Maintainer Guangbao Guo <ggb11111111@163.com>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2025-06-04 09:30:02 UTC

Encoding UTF-8

License GPL-3

Contents

`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))
```

IPC *The incremental principal component method can handle online data sets.*

Description

The incremental principal component method can handle online data sets.

Usage

```
IPC(data, m, eta)
```

Arguments

`data`	is an online data set
`m`	is the number of principal component
`eta`	is the proportion of online data to total data

Value

T2,T2k,V,Vhat,lambdahat,time

Examples

```
library(MASS)
n=2000;p=20;m=9;
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)
D=as.matrix(diag(rep(runif(p,0,1))))
epsilon=matrix(mvrnorm(n,rep(0,p),D),nrow=n)
data=mu+F%*%t(A)+epsilon
IPC(data=data,m=m,eta=0.8)
```

IPCR *Incremental Principal Component Regression for Online Datasets*

Description

The IPCR function implements an incremental Principal Component Regression (PCR) method designed to handle online datasets. It updates the principal components recursively as new data arrives, making it suitable for real-time data processing.

Usage

```
IPCR(data, eta, m, alpha)
```

Arguments

`data`	A data frame where the first column is the response variable and the remaining columns are predictor variables.
`eta`	The proportion of the initial sample size used to initialize the principal components ($0 < \text{eta} < 1$). Default is 0.0035.
`m`	The number of principal components to retain. Default is 3.
`alpha`	The significance level used for calculating critical values. Default is 0.05.

Details

The IPCR function performs the following steps: 1. Standardizes the predictor variables. 2. Initializes the principal components using the first `n0 = round(eta * n)` samples. 3. Recursively updates the principal components as each new sample arrives. 4. Fits a linear regression model using the principal component scores. 5. Back-transforms the regression coefficients to the original scale.

This method is particularly useful for datasets where new observations are continuously added, and the model needs to be updated incrementally.

Value

A list containing the following elements:

Bhat	The estimated regression coefficients, including the intercept.
RMSE	The Root Mean Square Error of the regression model.
summary	The summary of the linear regression model.
yhat	The predicted values of the response variable.

See Also

lm: For fitting linear models.

eigen: For computing eigenvalues and eigenvectors.

Examples

```
## Not run:
set.seed(1234)
library(MASS)
n <- 2000
p <- 10
mu0 <- as.matrix(runif(p, 0))
sigma0 <- as.matrix(runif(p, 0, 10))
ro <- as.matrix(c(runif(round(p / 2), -1, -0.8), runif(p - round(p / 2), 0.8, 1)))
R0 <- ro %*% t(ro)
diag(R0) <- 1
Sigma0 <- sigma0 %*% t(sigma0) * R0
x <- mvrnorm(n, mu0, Sigma0)
colnames(x) <- paste("x", 1:p, sep = "")
e <- rnorm(n, 0, 1)
B <- sample(1:3, (p + 1), replace = TRUE)
en <- matrix(rep(1, n * 1), ncol = 1)
y <- cbind(en, x) %*% B + e
colnames(y) <- paste("y")
data <- data.frame(cbind(y, x))

result <- IPCR(data = data, m = 3, eta = 0.0035, alpha = 0.05)
print(result$Bhat)
print(result$yhat)
print(result$RMSE)
print(result$summary)

## End(Not run)
```

PCR *Principal Component Regression (PCR)*

Description

The PCR function performs Principal Component Regression (PCR) on a given dataset. It standardizes the predictor variables, determines the number of principal components to retain based on a specified threshold, and fits a linear regression model using the principal component scores.

Usage

```
PCR(data, threshold)
```

Arguments

data	A data frame where the first column is the response variable and the remaining columns are predictor variables.
threshold	The proportion of variance to retain in the principal components (default is 0.95).

Details

The function performs the following steps: 1. Standardize the predictor variables. 2. Compute the covariance matrix of the standardized predictors. 3. Perform eigen decomposition on the covariance matrix to obtain principal components. 4. Determine the number of principal components to retain based on the cumulative explained variance exceeding the specified threshold. 5. Project the standardized predictors onto the retained principal components. 6. Fit a linear regression model using the principal component scores. 7. Back-transform the regression coefficients to the original scale.

Value

A list containing the following elements:

Bhat	The estimated regression coefficients, including the intercept.
RMSE	The Root Mean Square Error of the regression model.
summary	The summary of the linear regression model.
yhat	The predicted values of the response variable.

See Also

lm: For fitting linear models.

eigen: For computing eigenvalues and eigenvectors.

Examples

```
## Not run:
# Example data
set.seed(1234)
n <- 2000
p <- 10
mu0 <- as.matrix(runif(p, 0))
sigma0 <- as.matrix(runif(p, 0, 10))
ro <- as.matrix(c(runif(round(p / 2), -1, -0.8), runif(p - round(p / 2), 0.8, 1)))
R0 <- ro %*% t(ro)
diag(R0) <- 1
Sigma0 <- sigma0 %*% t(sigma0) * R0
x <- mvrnorm(n, mu0, Sigma0)
colnames(x) <- paste("x", 1:p, sep = "")
e <- rnorm(n, 0, 1)
B <- sample(1:3, (p + 1), replace = TRUE)
en <- matrix(rep(1, n * 1), ncol = 1)
y <- cbind(en, x) %*% B + e
colnames(y) <- paste("y")
data <- data.frame(cbind(y, x))

# Call the PCR function
result <- PCR(data, threshold = 0.9)

# Access the estimated regression coefficients
print(Bhat <- result$Bhat)

# Access the predicted values
print(yhat <- result$yhat)

# Print the summary of the regression model
print(result$summary)

# Print the RMSE
print(paste("RMSE:", result$RMSE))

## End(Not run)
```

PPC *The perturbation principal component method can handle online data sets.*

Description

The perturbation principal component method can handle online data sets.

Usage

```
PPC(data, m, eta)
```

Arguments

data	is an online data set
m	is the number of principal component
eta	is the proportion of online data to total data

Value

T2,T2k,V,Vhat,lambdahat,time

Examples

```
library(MASS)
n=2000;p=20;m=9;
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)
D=as.matrix(diag(rep(runif(p,0,1))))
epsilon=matrix(mvrnorm(n,rep(0,p),D),nrow=n)
data=mu+F%*%t(A)+epsilon
PPC(data=data,m=m,eta=0.8)
```

PPCR *Perturbation-based Principal Component Regression*

Description

This function performs Perturbation-based Principal Component Regression (PPCR) on the provided dataset. It combines Principal Component Analysis (PCA) with linear regression, incorporating perturbation to enhance robustness.

Usage

```
PPCR(data, eta = 0.0035, m = 3, alpha = 0.05, perturbation_factor = 0.1)
```

Arguments

data	A data frame containing the response variable and predictors.
eta	A proportion (between 0 and 1) determining the initial sample size for PCA.
m	The number of principal components to retain.
alpha	Significance level (currently not used in the function).
perturbation_factor	A factor controlling the magnitude of perturbation added to the principal components.

Details

The function first standardizes the predictors, then performs PCA on an initial subset of the data. It iteratively updates the principal components by incorporating new observations and adding random perturbations. Finally, it fits a linear regression model using the principal components as predictors and transforms the coefficients back to the original space.

Value

A list containing the following components:

`Bhat`	Estimated regression coefficients in the original space.
`RMSE`	Root Mean Squared Error of the regression model.
`summary`	Summary of the linear regression model.
`Vhat`	Estimated principal components.
`lambdahat`	Estimated eigenvalues.
`yhat`	Predicted values from the regression model.

See Also

`lm`: For linear regression models.

`prcomp`: For principal component analysis.

Examples

```
## Not run:
# Example data
set.seed(1234)
n <- 2000
p <- 10
mu0 <- as.matrix(runif(p, 0))
sigma0 <- as.matrix(runif(p, 0, 10))
ro <- as.matrix(c(runif(round(p / 2), -1, -0.8), runif(p - round(p / 2), 0.8, 1)))
R0 <- ro %*% t(ro)
diag(R0) <- 1
Sigma0 <- sigma0 %*% t(sigma0) * R0
x <- mvrnorm(n, mu0, Sigma0)
colnames(x) <- paste("x", 1:p, sep = "")
e <- rnorm(n, 0, 1)
B <- sample(1:3, (p + 1), replace = TRUE)
en <- matrix(rep(1, n * 1), ncol = 1)
y <- cbind(en, x) %*% B + e
colnames(y) <- paste("y")
data <- data.frame(cbind(y, x))

# Call the PPCR function
result <- PPCR(data, eta = 0.0035, m = 3, alpha = 0.05, perturbation_factor = 0.1)

# Print results
print(result$Bhat)  # Estimated regression coefficients
```

```
print(result$RMSE)  # RMSE of the model
print(result$summary)  # Summary of the regression model

## End(Not run)
```

---

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))
```

SAPC *The stochastic approximate component method can handle online data sets.*

Description

The stochastic approximate component method can handle online data sets.

Usage

```
SAPC(data, m, eta, alpha)
```

Arguments

data	is a online data set
m	is the number of principal component
eta	is the proportion of online data to total data
alpha	is the step size

Value

T2,T2k,V,Vhat,lambdahat,time

Examples

```
library(MASS)
n=2000;p=20;m=9;
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)
D=as.matrix(diag(rep(runif(p,0,1))))
epsilon=matrix(mvrnorm(n,rep(0,p),D),nrow=n)
data=mu+F%*%t(A)+epsilon
SAPC(data=data,m=m,eta=0.8,alpha=1)
```

SPCR *The stochastic principal component method can handle online data sets.*

Description

The stochastic principal component method can handle online data sets.

Usage

```
SPCR(data, eta, m)
```

Arguments

data	A data frame containing the response variable and predictors.
eta	proportion (between 0 and 1) determining the initial sample size for PCA.
m	The number of principal components to retain.

Value

A list containing the following elements:

Bhat	The estimated regression coefficients, including the intercept.
RMSE	The Root Mean Square Error of the regression model.
summary	The summary of the linear regression model.
yhat	The predicted values of the response variable.

Examples

```
# Example data
library(MASS);library(stats)
set.seed(1234)
n <- 2000
p <- 10
mu0 <- as.matrix(runif(p, 0))
sigma0 <- as.matrix(runif(p, 0, 10))
ro <- as.matrix(c(runif(round(p / 2), -1, -0.8), runif(p - round(p / 2), 0.8, 1)))
R0 <- ro %*% t(ro)
diag(R0) <- 1
Sigma0 <- sigma0 %*% t(sigma0) * R0
x <- mvrnorm(n, mu0, Sigma0)
colnames(x) <- paste("x", 1:p, sep = "")
e <- rnorm(n, 0, 1)
B <- sample(1:3, (p + 1), replace = TRUE)
en <- matrix(rep(1, n * 1), ncol = 1)
y <- cbind(en, x) %*% B + e
colnames(y) <- paste("y")
data <- data.frame(cbind(y, x))
result <- SPCR(data,  eta = 0.0035, m = 3)
```

spcrl *The stochastic principal component regression with varying learning-rate can handle online data sets.*

Description

The stochastic principal component regression with varying learning-rate can handle online data sets.

Usage

```
spcrl(data, m, eta, alpha)
```

Arguments

data	is a online data set
m	is the number of principal component
eta	is the proportion of online data to total data
alpha	is the step size

Value

T2,T2k,V,Vhat,lambdahat,time

Examples

```
library(MASS)
n <- 2000
p <- 20
m <- 9
mu <- t(matrix(rep(runif(p, 0, 1000), p, n)))
mu0 <- as.matrix(runif(p, 0))
sigma0 <- diag(runif(p, 1, 10))
ro <- as.matrix(c(runif(round(p/2), -1, -0.8), runif(p - round(p/2), 0.8, 1)))
R0 <- ro %*% t(ro)
diag(R0) <- 1
Sigma0 <- sigma0 %*% R0 %*% sigma0
x <- mvrnorm(n, mu0, Sigma0)
colnames(x) <- paste0("x", 1:p)
e <- rnorm(n, 0, 1)
B <- sample(1:3, (p + 1), replace = TRUE)
en <- matrix(rep(1, n), ncol = 1)
y <- cbind(en, x) %*% B + e
colnames(y) <- "y"
data <- data.frame(cbind(y, x))
spcrl(data = data, m = m, eta = 0.8, alpha = 0.5)
```

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