# Package ‘PSGD’

December 18, 2024

**Type** Package

**Title** Projected Subset Gradient Descent

**Version** 1.0.4

**Date** 2024-12-16

**Maintainer** Anthony Christidis <anthony.christidis@stat.ubc.ca>

**Description** Functions to generate ensembles of generalized linear models using a greedy projected subset gradient descent algorithm. The sparsity and diversity tuning parameters are selected by cross-validation.

**License** GPL (>= 2)

**Biarch** true

**LinkingTo** Rcpp, RcppArmadillo

**RoxygenNote** 7.3.2

**Imports** Rcpp (>= 1.0.7)

**Suggests** testthat, mvnfast, vctrs

**NeedsCompilation** yes

**Author** Anthony Christidis [aut, cre],
Stefan Van Aelst [aut],
Ruben Zamar [aut]

**Repository** CRAN

**Date/Publication** 2024-12-18 15:20:06 UTC

# Contents

coef.cv.PSGD *Coefficients for cv.PSGD Object*

## Description

coef.cv.PSGD returns the coefficients for a cv.PSGD object.

## Usage

```
## S3 method for class 'cv.PSGD'
coef(object, group_index = NULL, ...)
```

## Arguments

| | |
|---|---|
| object | An object of class cv.PSGD |
| group_index | Groups included in the ensemble. Default setting includes all the groups. |
| ... | Additional arguments for compatibility. |

## Value

The coefficients for the cv.PSGD object.

## Author(s)

Anthony-Alexander Christidis, <anthony.christidis@stat.ubc.ca>

## See Also

cv.PSGD

## Examples

```
# Required Libraries
library(mvnfast)

# Setting the parameters
p <- 100
n <- 40
n.test <- 1000
sparsity <- 0.2
rho <- 0.5
SNR <- 3

# Generating the coefficient
p.active <- floor(p*sparsity)
a <- 4*log(n)/sqrt(n)
neg.prob <- 0.2
nonzero.betas <- (-1)^(rbinom(p.active, 1, neg.prob))*(a + abs(rnorm(p.active)))
```

```
# Correlation structure
Sigma <- matrix(0, p, p)
Sigma[1:p.active, 1:p.active] <- rho
diag(Sigma) <- 1
true.beta <- c(nonzero.betas, rep(0 , p - p.active))

# Computing the noise parameter for target SNR
sigma.epsilon <- as.numeric(sqrt((t(true.beta) %*% Sigma %*% true.beta)/SNR))

# Simulate some data
set.seed(1)
x.train <- mvnfast::rmvn(n, mu=rep(0,p), sigma=Sigma)
y.train <- 1 + x.train %*% true.beta + rnorm(n=n, mean=0, sd=sigma.epsilon)
x.test <- mvnfast::rmvn(n.test, mu=rep(0,p), sigma=Sigma)
y.test <- 1 + x.test %*% true.beta + rnorm(n.test, sd=sigma.epsilon)

# CV PSGD Ensemble
output <- cv.PSGD(x = x.train, y = y.train, n_models = 5,
                  model_type = c("Linear", "Logistic")[1], include_intercept = TRUE,
                  split_grid = c(2, 3), size_grid = c(10, 15),
                  max_iter = 20,
                  cycling_iter = 0,
                  n_folds = 5,
                  n_threads = 1)
psgd.coef <- coef(output, group_index = 1:output$n_models)
psgd.predictions <- predict(output, newx = x.test, group_index = 1:output$n_models)
mean((y.test - psgd.predictions)^2)/sigma.epsilon^2
```

---

`coef.PSGD` *Coefficients for PSGD Object*

---

### Description

`coef.PSGD` returns the coefficients for a PSGD object.

### Usage

```
## S3 method for class 'PSGD'
coef(object, group_index = NULL, ...)
```

### Arguments

| | |
|---|---|
| `object` | An object of class PSGD. |
| `group_index` | Groups included in the ensemble. Default setting includes all the groups. |
| `...` | Additional arguments for compatibility. |

### Value

The coefficients for the PSGD object.

## Author(s)

Anthony-Alexander Christidis, <anthony.christidis@stat.ubc.ca>

## See Also

PSGD

## Examples

```
# Required Libraries
library(mvnfast)

# Setting the parameters
p <- 100
n <- 40
n.test <- 1000
sparsity <- 0.2
rho <- 0.5
SNR <- 3

# Generating the coefficient
p.active <- floor(p*sparsity)
a <- 4*log(n)/sqrt(n)
neg.prob <- 0.2
nonzero.betas <- (-1)^(rbinom(p.active, 1, neg.prob))*(a + abs(rnorm(p.active)))

# Correlation structure
Sigma <- matrix(0, p, p)
Sigma[1:p.active, 1:p.active] <- rho
diag(Sigma) <- 1
true.beta <- c(nonzero.betas, rep(0 , p - p.active))

# Computing the noise parameter for target SNR
sigma.epsilon <- as.numeric(sqrt((t(true.beta) %*% Sigma %*% true.beta)/SNR))

# Simulate some data
set.seed(1)
x.train <- mvnfast::rmvn(n, mu=rep(0,p), sigma=Sigma)
y.train <- 1 + x.train %*% true.beta + rnorm(n=n, mean=0, sd=sigma.epsilon)
x.test <- mvnfast::rmvn(n.test, mu=rep(0,p), sigma=Sigma)
y.test <- 1 + x.test %*% true.beta + rnorm(n.test, sd=sigma.epsilon)

# PSGD Ensemble
output <- PSGD(x = x.train, y = y.train, n_models = 5,
               model_type = c("Linear", "Logistic")[1], include_intercept = TRUE,
               split = 3, size = 10,
               max_iter = 20,
               cycling_iter = 0)
psgd.coef <- coef(output, group_index = 1:output$n_models)
psgd.predictions <- predict(output, newx = x.test, group_index = 1:output$n_models)
mean((y.test - psgd.predictions)^2)/sigma.epsilon^2
```

cv.PSGD *Cross-Validation - Projected Subset Gradient Descent*

## Description

cv.PSGD performs the CV procedure for a projected subset gradient descent algorithm.

## Usage

```
cv.PSGD(
  x,
  y,
  n_models,
  model_type = c("Linear", "Logistic")[1],
  include_intercept = TRUE,
  split_grid,
  size_grid,
  max_iter = 100,
  cycling_iter = 5,
  n_folds = 5,
  n_threads = 1
)
```

## Arguments

| | |
|---|---|
| x | Design matrix. |
| y | Response vector. |
| n_models | Number of models into which the variables are split. |
| model_type | Model type. Must be one of "Linear or Logistic". Default is "Linear". |
| include_intercept | TRUE or FALSE parameter for the inclusion of an intercept term. Default is TRUE. |
| split_grid | Grid for number of models that may share a variable. |
| size_grid | Grid for number of variables that a model may have. |
| max_iter | Maximum number of iterations in PSGD algorithm. |
| cycling_iter | Number of random cycling permutations. |
| n_folds | Number of cross-validation folds. Default is 5 |
| n_threads | Number of threads. Default is 1. |

## Value

An object of class cv.PSGD

## Author(s)

Anthony-Alexander Christidis, <anthony.christidis@stat.ubc.ca>

## See Also

coef.cv.PSGD, predict.cv.PSGD

## Examples

```
# Required Libraries
library(mvnfast)

# Setting the parameters
p <- 100
n <- 40
n.test <- 1000
sparsity <- 0.2
rho <- 0.5
SNR <- 3

# Generating the coefficient
p.active <- floor(p*sparsity)
a <- 4*log(n)/sqrt(n)
neg.prob <- 0.2
nonzero.betas <- (-1)^(rbinom(p.active, 1, neg.prob))*(a + abs(rnorm(p.active)))

# Correlation structure
Sigma <- matrix(0, p, p)
Sigma[1:p.active, 1:p.active] <- rho
diag(Sigma) <- 1
true.beta <- c(nonzero.betas, rep(0 , p - p.active))

# Computing the noise parameter for target SNR
sigma.epsilon <- as.numeric(sqrt((t(true.beta) %*% Sigma %*% true.beta)/SNR))

# Simulate some data
set.seed(1)
x.train <- mvnfast::rmvn(n, mu=rep(0,p), sigma=Sigma)
y.train <- 1 + x.train %*% true.beta + rnorm(n=n, mean=0, sd=sigma.epsilon)
x.test <- mvnfast::rmvn(n.test, mu=rep(0,p), sigma=Sigma)
y.test <- 1 + x.test %*% true.beta + rnorm(n.test, sd=sigma.epsilon)

# CV PSGD Ensemble
output <- cv.PSGD(x = x.train, y = y.train, n_models = 5,
                  model_type = c("Linear", "Logistic")[1], include_intercept = TRUE,
                  split_grid = c(2, 3), size_grid = c(10, 15),
                  max_iter = 20,
                  cycling_iter = 0,
                  n_folds = 5,
                  n_threads = 1)
psgd.coef <- coef(output, group_index = 1:output$n_models)
psgd.predictions <- predict(output, newx = x.test, group_index = 1:output$n_models)
```

```
mean((y.test - psgd.predictions)^2)/sigma.epsilon^2
```

---

predict.cv.PSGD    *Predictions for cv.PSGD Object*

---

### Description

predict.cv.PSGD returns the predictions for a cv.PSGD object.

### Usage

```
## S3 method for class 'cv.PSGD'
predict(object, newx, group_index = group_index, ...)
```

### Arguments

| | |
|---|---|
| object | An object of class cv.PSGD |
| newx | New data for predictions. |
| group_index | Groups included in the ensemble. Default setting includes all the groups. |
| ... | Additional arguments for compatibility. |

### Value

The predictions for the cv.PSGD object.

### Author(s)

Anthony-Alexander Christidis, <anthony.christidis@stat.ubc.ca>

### See Also

cv.PSGD

### Examples

```
# Required Libraries
library(mvnfast)

# Setting the parameters
p <- 100
n <- 40
n.test <- 1000
sparsity <- 0.2
rho <- 0.5
SNR <- 3

# Generating the coefficient
```

```
p.active <- floor(p*sparsity)
a <- 4*log(n)/sqrt(n)
neg.prob <- 0.2
nonzero.betas <- (-1)^(rbinom(p.active, 1, neg.prob))*(a + abs(rnorm(p.active)))

# Correlation structure
Sigma <- matrix(0, p, p)
Sigma[1:p.active, 1:p.active] <- rho
diag(Sigma) <- 1
true.beta <- c(nonzero.betas, rep(0 , p - p.active))

# Computing the noise parameter for target SNR
sigma.epsilon <- as.numeric(sqrt((t(true.beta) %*% Sigma %*% true.beta)/SNR))

# Simulate some data
set.seed(1)
x.train <- mvnfast::rmvn(n, mu=rep(0,p), sigma=Sigma)
y.train <- 1 + x.train %*% true.beta + rnorm(n=n, mean=0, sd=sigma.epsilon)
x.test <- mvnfast::rmvn(n.test, mu=rep(0,p), sigma=Sigma)
y.test <- 1 + x.test %*% true.beta + rnorm(n.test, sd=sigma.epsilon)

# CV PSGD Ensemble
output <- cv.PSGD(x = x.train, y = y.train, n_models = 5,
                  model_type = c("Linear", "Logistic")[1], include_intercept = TRUE,
                  split_grid = c(2, 3), size_grid = c(10, 15),
                  max_iter = 20,
                  cycling_iter = 0,
                  n_folds = 5,
                  n_threads = 1)
psgd.coef <- coef(output, group_index = 1:output$n_models)
psgd.predictions <- predict(output, newx = x.test, group_index = 1:output$n_models)
mean((y.test - psgd.predictions)^2)/sigma.epsilon^2
```

---

`predict.PSGD` *Predictions for PSGD Object*

---

## Description

`predict.PSGD` returns the predictions for a PSGD object.

## Usage

```
## S3 method for class 'PSGD'
predict(object, newx, group_index = NULL, ...)
```

## Arguments

| | |
|---|---|
| `object` | An object of class PSGD |
| `newx` | New data for predictions. |

`group_index` Groups included in the ensemble. Default setting includes all the groups.

`...` Additional arguments for compatibility.

## Value

The predictions for the PSGD object.

## Author(s)

Anthony-Alexander Christidis, <anthony.christidis@stat.ubc.ca>

## See Also

PSGD

## Examples

```
# Required Libraries
library(mvnfast)

# Setting the parameters
p <- 100
n <- 40
n.test <- 1000
sparsity <- 0.2
rho <- 0.5
SNR <- 3

# Generating the coefficient
p.active <- floor(p*sparsity)
a <- 4*log(n)/sqrt(n)
neg.prob <- 0.2
nonzero.betas <- (-1)^(rbinom(p.active, 1, neg.prob))*(a + abs(rnorm(p.active)))

# Correlation structure
Sigma <- matrix(0, p, p)
Sigma[1:p.active, 1:p.active] <- rho
diag(Sigma) <- 1
true.beta <- c(nonzero.betas, rep(0 , p - p.active))

# Computing the noise parameter for target SNR
sigma.epsilon <- as.numeric(sqrt((t(true.beta) %*% Sigma %*% true.beta)/SNR))

# Simulate some data
set.seed(1)
x.train <- mvnfast::rmvn(n, mu=rep(0,p), sigma=Sigma)
y.train <- 1 + x.train %*% true.beta + rnorm(n=n, mean=0, sd=sigma.epsilon)
x.test <- mvnfast::rmvn(n.test, mu=rep(0,p), sigma=Sigma)
y.test <- 1 + x.test %*% true.beta + rnorm(n.test, sd=sigma.epsilon)

# PSGD Ensemble
output <- PSGD(x = x.train, y = y.train, n_models = 5,
```

```
               model_type = c("Linear", "Logistic")[1], include_intercept = TRUE,
               split = 3, size = 10,
               max_iter = 20,
               cycling_iter = 0)
psgd.coef <- coef(output, group_index = 1:output$n_models)
psgd.predictions <- predict(output, newx = x.test, group_index = 1:output$n_models)
mean((y.test - psgd.predictions)^2)/sigma.epsilon^2
```

---

PSGD *Projected Subset Gradient Descent*

---

## Description

PSGD performs a projected subset gradient descent algorithm.

## Usage

```
PSGD(
  x,
  y,
  n_models,
  model_type = c("Linear", "Logistic")[1],
  include_intercept = TRUE,
  split,
  size,
  max_iter = 100,
  cycling_iter = 5
)
```

## Arguments

| | |
|---|---|
| x | Design matrix. |
| y | Response vector. |
| n_models | Number of models into which the variables are split. |
| model_type | Model type. Must be one of "Linear or Logistic". Default is "Linear". |
| include_intercept | TRUE or FALSE parameter for the inclusion of an intercept term. Default is TRUE. |
| split | Number of models that may share a variable. |
| size | Number of variables that a model may have. |
| max_iter | Maximum number of iterations in PSGD algorithm. |
| cycling_iter | Number of random cycling permutations. |

## Value

An object of class PSGD

## Author(s)

Anthony-Alexander Christidis, <anthony.christidis@stat.ubc.ca>

## See Also

coef.PSGD, predict.PSGD

## Examples

```
# Required Libraries
library(mvnfast)

# Setting the parameters
p <- 100
n <- 40
n.test <- 1000
sparsity <- 0.2
rho <- 0.5
SNR <- 3

# Generating the coefficient
p.active <- floor(p*sparsity)
a <- 4*log(n)/sqrt(n)
neg.prob <- 0.2
nonzero.betas <- (-1)^(rbinom(p.active, 1, neg.prob))*(a + abs(rnorm(p.active)))

# Correlation structure
Sigma <- matrix(0, p, p)
Sigma[1:p.active, 1:p.active] <- rho
diag(Sigma) <- 1
true.beta <- c(nonzero.betas, rep(0 , p - p.active))

# Computing the noise parameter for target SNR
sigma.epsilon <- as.numeric(sqrt((t(true.beta) %*% Sigma %*% true.beta)/SNR))

# Simulate some data
set.seed(1)
x.train <- mvnfast::rmvn(n, mu=rep(0,p), sigma=Sigma)
y.train <- 1 + x.train %*% true.beta + rnorm(n=n, mean=0, sd=sigma.epsilon)
x.test <- mvnfast::rmvn(n.test, mu=rep(0,p), sigma=Sigma)
y.test <- 1 + x.test %*% true.beta + rnorm(n.test, sd=sigma.epsilon)

# PSGD Ensemble
output <- PSGD(x = x.train, y = y.train, n_models = 5,
               model_type = c("Linear", "Logistic")[1], include_intercept = TRUE,
               split = 3, size = 10,
               max_iter = 20,
               cycling_iter = 0)
psgd.coef <- coef(output, group_index = 1:output$n_models)
psgd.predictions <- predict(output, newx = x.test, group_index = 1:output$n_models)
mean((y.test - psgd.predictions)^2)/sigma.epsilon^2
```

# Index