

# Package ‘palasso’

April 19, 2019

**Version** 0.0.5

**Title** Paired Lasso Regression

**Description**

Implements sparse regression with paired covariates (Rauschenberger et al. 2019). For the optional shrinkage, install ashR (<<https://github.com/stephens999/ashr>>) and CorShrink (<<https://github.com/kkdey/CorShrink>>) from GitHub (see README).

**Depends** R (>= 3.0.0)

**Imports** glmnet, Matrix, survival

**Suggests** knitr, testthat

**Enhances** ashR, CorShrink, pROC, edgeR

**VignetteBuilder** knitr

**License** GPL-3

**Encoding** UTF-8

**RoxygenNote** 6.1.1

**LazyData** false

**URL** <https://github.com/rauschenberger/palasso>

**BugReports** <https://github.com/rauschenberger/palasso/issues>

**NeedsCompilation** no

**Author** Armin Rauschenberger [aut, cre]

**Maintainer** Armin Rauschenberger <[a.rauschenberger@vumc.nl](mailto:a.rauschenberger@vumc.nl)>

**Repository** CRAN

**Date/Publication** 2019-04-19 21:00:03 UTC

## R topics documented:

.args . . . . .	2
.combine . . . . .	2
.cor . . . . .	3
.cv . . . . .	4

<code>.dims</code>	5
<code>.extract</code>	5
<code>.fit</code>	6
<code>.folds</code>	7
<code>.loss</code>	7
<code>.weight</code>	8
arguments	8
methods	9
palasso	10

## Index 12

---

<code>.args</code>	<i>Arguments</i>
--------------------	------------------

---

### Description

Checks the validity of the provided arguments.

### Usage

```
.args(...)
```

### Arguments

... Arguments supplied to `palasso`, other than `y`, `X` and `max`.

### Value

Returns the arguments as a list, including default values for missing arguments.

### Examples

```
NA
```

---

<code>.combine</code>	<i>Combining p-values</i>
-----------------------	---------------------------

---

### Description

This function combines local  $p$ -values to a global  $p$ -value.

### Usage

```
.combine(x, method = "simes")
```

**Arguments**

x                    local *p*-values: numeric vector of length *k*  
 method             character "fisher", "tippet", "sidak", or "simes"

**Value**

These functions return a numeric vector of length *p* (main effects), or a numeric matrix with *p* rows and *p* columns (interaction effects).

**References**

Westfall, P. H. (2005). "Combining p-values". Encyclopedia of Biostatistics <https://doi.org/10.1002/0470011815.b2a15181>

**Examples**

```
# independence
p <- runif(10)
palasso:::.combine(p)

## dependence
#runif <- function(n,cor=0){
#   Sigma <- matrix(cor,nrow=n,ncol=n)
#   diag(Sigma) <- 1
#   mu <- rep(0,times=n)
#   q <- MASS::mvrnorm(n=1,mu=mu,Sigma=Sigma)
#   stats::pnorm(q=q)
#}
#p <- runif(n=10,cor=0.8)
#combine(p)
```

---

.cor	<i>Correlation</i>
------	--------------------

---

**Description**

Calculates the correlation between the response and the covariates. Shrinks the correlation coefficients for each covariate set separately.

**Usage**

```
.cor(y, x, args)
```

**Arguments**

y                    vector of length *n*  
 x                    matrix with *n* rows and *p* columns  
 args                options for paired lasso: list of arguments (output from [.dims](#) and [.args](#))

**Value**

list of vectors

**Examples**

NA

---

.cv

*Cross-validation*

---

**Description**

Repeatedly leaves out samples, and predicts their response.

**Usage**

```
.cv(y, x, foldid, lambda, args)
```

**Arguments**

y	response: vector of length $n$
x	covariates: matrix with $n$ rows (samples) and $k * p$ columns (variables)
foldid	fold identifiers: vector of length $n$ , with entries from 1 to n folds
lambda	lambda sequence: vector of decreasing positive values
args	options for paired lasso: list of arguments (output from <a href="#">.dims</a> and <a href="#">.args</a> )

**Value**

Returns matrix of predicted values (except "cox")

**Examples**

NA

---

.dims                      *Dimensionality*

---

### Description

This function extracts the dimensions.

### Usage

```
.dims(y, X, args = NULL)
```

### Arguments

y	response: vector of length $n$
X	covariates: list of matrices, each with $n$ rows (samples) and $p$ columns (variables)
args	options for paired lasso: list of arguments (output from <a href="#">.dims</a> and <a href="#">.args</a> )

### Value

The function `.dims` extracts the dimensionality. It returns the numbers of samples, covariate pairs and covariate sets. It also returns the number of weighting schemes, and the names of these weighting schemes.

### Examples

```
NA
```

---

.extract                      *Extraction*

---

### Description

Extracts cv.glmnet-like object.

### Usage

```
.extract(fit, lambda, cvm, type.measure)
```

**Arguments**

fit	matrix with one row for each sample ("gaussian", "binomial" and "poisson"), or one row for each fold (only "cox"), and one column for each lambda (output from <a href="#">.fit</a> )
lambda	lambda sequence: vector of decreasing positive values
cvm	mean cross-validated loss: vector of same length as lambda (output from <a href="#">.loss</a> )
type.measure	... loss function: character "deviance", "mse", "mae", "class", or "auc"

**Examples**

NA

---

.fit *Model bag*

---

**Description**

Fits all models from the chosen bag.

**Usage**

```
.fit(y, x, args)
```

**Arguments**

y	response: vector of length $n$
x	covariates: matrix with $n$ rows (samples) and $k * p$ columns (variables)
args	options for paired lasso: list of arguments (output from <a href="#">.dims</a> and <a href="#">.args</a> )

**Value**

list of glmnet-like objects

**Examples**

NA

---

.folds *Cross-validation folds*

---

**Description**

Assigns samples to cross-validation folds, balancing the folds in the case of a binary or survival response.

**Usage**

```
.folds(y, n folds, foldid = NULL)
```

**Arguments**

y	response: vector of length $n$
n folds	number of folds: positive integer ( $\geq 10$ recommended)
foldid	fold identifiers: vector of length $n$ , with entries from 1 to n folds

**Value**

Returns the fold identifiers.

**Examples**

```
NA
```

---

.loss *Cross-validation loss*

---

**Description**

Calculates mean cross-validated loss

**Usage**

```
.loss(y, fit, family, type.measure, foldid = NULL)
```

**Arguments**

y	response: vector of length $n$
fit	matrix with one row for each sample ("gaussian", "binomial" and "poisson"), or one row for each fold (only "cox"), and one column for each lambda (output from <code>.fit</code> )
family	model family: character "gaussian", "binomial", "poisson", or "cox"
type.measure	... loss function: character "deviance", "mse", "mae", "class", or "auc"
foldid	fold identifiers: vector of length $n$ , with entries from 1 to n folds

**Value**

Returns list of vectors, one for each model.

**Examples**

NA

---

<code>.weight</code>	<i>Weighting schemes</i>
----------------------	--------------------------

---

**Description**

Calculates the weighting schemes.

**Usage**

```
.weight(cor, args)
```

**Arguments**

<code>cor</code>	correlation coefficients: list of $k$ vectors of length $p$ (one vector for each covariate set with one entry for each covariate)
<code>args</code>	options for paired lasso: list of arguments (output from <code>.dims</code> and <code>.args</code> )

**Value**

list of named vectors (one for each weighting scheme)

**Examples**

NA

---

arguments	<i>Arguments for "palasso"</i>
-----------	--------------------------------

---

**Description**

This page lists the arguments for the (internal) "palasso" function(s).



**Arguments**

<code>y</code>	response: vector of length $n$
<code>X</code>	covariates: list of matrices, each with $n$ rows (samples) and $p$ columns (variables)
<code>max</code>	maximum number of non-zero coefficients: positive numeric, or NULL (no sparsity constraint)
<code>...</code>	further arguments for <code>cv.glmnet</code> or <code>glmnet</code>
<code>x</code>	covariates: matrix with $n$ rows (samples) and $k * p$ columns (variables)
<code>args</code>	options for paired lasso: list of arguments (output from <code>.dims</code> and <code>.args</code> )
<code>nfolds</code>	number of folds: positive integer ( $\geq 10$ recommended)
<code>foldid</code>	fold identifiers: vector of length $n$ , with entries from 1 to <code>nfolds</code>
<code>cor</code>	correlation coefficients: list of $k$ vectors of length $p$ (one vector for each covariate set with one entry for each covariate)
<code>lambda</code>	lambda sequence: vector of decreasing positive values
<code>family</code>	model family: character "gaussian", "binomial", "poisson", or "cox"
<code>type.measure</code>	... loss function: character "deviance", "mse", "mae", "class", or "auc"
<code>fit</code>	matrix with one row for each sample ("gaussian", "binomial" and "poisson"), or one row for each fold (only "cox"), and one column for each lambda (output from <code>.fit</code> )
<code>cvm</code>	mean cross-validated loss: vector of same length as <code>lambda</code> (output from <code>.loss</code> )

---

 methods

*Methods for class "palasso"*


---

**Description**

This page lists the main methods for class "palasso".

**Usage**

```
## S3 method for class 'palasso'
predict(object, newdata, model = "paired",
        s = "lambda.min", max = NULL, ...)

## S3 method for class 'palasso'
coef(object, model = "paired", s = "lambda.min",
      max = NULL, ...)

## S3 method for class 'palasso'
weights(object, model = "paired", max = NULL, ...)

## S3 method for class 'palasso'
```

```
fitted(object, model = "paired", s = "lambda.min",
       max = NULL, ...)

## S3 method for class 'palasso'
residuals(object, model = "paired", s = "lambda.min",
          max = NULL, ...)

## S3 method for class 'palasso'
deviance(object, model = "paired", max = NULL, ...)

## S3 method for class 'palasso'
logLik(object, model = "paired", max = NULL, ...)

## S3 method for class 'palasso'
summary(object, model = "paired", ...)
```

### Arguments

object	<a href="#">palasso</a> object
newdata	covariates: list of matrices, each with $n$ rows (samples) and $p$ columns (variables)
model	character "paired", or an entry of names(object)
s	penalty parameter: character "lambda.min" or "lambda.1se", positive numeric, or NULL (entire sequence)
max	maximum number of non-zero coefficients, positive integer, or NULL
...	further arguments for <a href="#">predict.cv.glmnet</a> , <a href="#">coef.cv.glmnet</a> , or <a href="#">deviance.glmnet</a>

### Details

By default, the function `predict` returns the linear predictor (`type="link"`). Consider predicting the response (`type="response"`).

### See Also

Use [palasso](#) to fit the paired lasso.

---

palasso

*Paired lasso*

---

### Description

The function `palasso` fits the paired lasso. Use this function if you have *paired covariates* and want a *sparse model*.

### Usage

```
palasso(y = y, X = X, max = 10, ...)
```

**Arguments**

<code>y</code>	response: vector of length $n$
<code>X</code>	covariates: list of matrices, each with $n$ rows (samples) and $p$ columns (variables)
<code>max</code>	maximum number of non-zero coefficients: positive numeric, or NULL (no sparsity constraint)
<code>...</code>	further arguments for <code>cv.glmnet</code> or <code>glmnet</code>

**Details**

Let  $x$  denote one entry of the list  $X$ . See `glmnet` for alternative specifications of  $y$  and  $x$ . Among the further arguments, `family` must equal "gaussian", "binomial", "poisson", or "cox", and `penalty.factor` must not be used.

Hidden arguments: Deactivate adaptive lasso by setting `adaptive` to FALSE, activate standard lasso by setting `standard` to TRUE, and deactivate shrinkage by setting `shrink` to FALSE.

**Value**

This function returns an object of class `palasso`. Available methods include `predict`, `coef`, `weights`, `fitted`, `residuals`, `deviance`, `logLik`, and `summary`.

**References**

A Rauschenberger, I Ciocanea-Teodorescu, RX Menezes, MA Jonker, and MA van de Wiel (2018). "Sparse regression with paired covariates." *Manuscript in preparation*. <a.rauschenberger@vumc.nl>

**Examples**

```
set.seed(1)
n <- 50; p <- 20
y <- rbinom(n=n, size=1, prob=0.5)
X <- lapply(1:2, function(x) matrix(rnorm(n*p), nrow=n, ncol=p))
object <- palasso(y=y, X=X, family="binomial", standard=TRUE)
```

# Index

## \*Topic **methods**

- .combine, [2](#)
- .args, [2](#), [3–6](#), [8](#), [9](#)
- .combine, [2](#)
- .cor, [3](#)
- .cv, [4](#)
- .dims, [3–5](#), [5](#), [6](#), [8](#), [9](#)
- .extract, [5](#)
- .fit, [6](#), [6](#), [7](#), [9](#)
- .folds, [7](#)
- .loss, [6](#), [7](#), [9](#)
- .weight, [8](#)

arguments, [8](#)

coef, [11](#)

coef.cv.glmnet, [10](#)

coef.palasso (methods), [9](#)

cv.glmnet, [9](#), [11](#)

deviance, [11](#)

deviance.glmnet, [10](#)

deviance.palasso (methods), [9](#)

fitted, [11](#)

fitted.palasso (methods), [9](#)

glmnet, [9](#), [11](#)

logLik, [11](#)

logLik.palasso (methods), [9](#)

methods, [9](#)

palasso, [2](#), [10](#), [10](#)

palasso-package (palasso), [10](#)

predict, [11](#)

predict.cv.glmnet, [10](#)

predict.palasso (methods), [9](#)

residuals, [11](#)

residuals.palasso (methods), [9](#)

summary, [11](#)

summary.palasso (methods), [9](#)

weights, [11](#)

weights.palasso (methods), [9](#)