

Package: slasso (via r-universe)

May 20, 2026

Type Package

Title S-LASSO Estimator for the Function-on-Function Linear Regression

Version 1.0.1

Description Implements the smooth LASSO estimator for the function-on-function linear regression model described in Centofanti et al. (2022) <[doi:10.1016/j.csda.2022.107556](https://doi.org/10.1016/j.csda.2022.107556)>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

LinkingTo Rcpp, RcppArmadillo

Depends inline

Imports Rcpp, fda, fda.usc, matrixcalc, parallel, matrixStats, MASS, plot3D, methods, cxxfunplus

URL <https://github.com/fabiocentofanti/slasso>

BugReports <https://github.com/fabiocentofanti/slasso/issues>

SystemRequirements GNU make

Suggests knitr, rmarkdown, testthat

Roxygen list(markdown = TRUE)

Config/pak/sysreqs make

Repository <https://fabiocentofanti.r-universe.dev>

Date/Publication 2026-01-19 14:31:45 UTC

RemoteUrl <https://github.com/fabiocentofanti/slasso>

RemoteRef HEAD

RemoteSha 1235f5123d2fcccb98e38c5e0978388fd43542bd

Contents

slasso-package	2
plot.slasso	3
simulate_data	4
slasso.fr	5
slasso.fr_cv	7
Index	9

slasso-package	<i>Smooth LASSO Estimator for the Function-on-Function Linear Regression Model</i>
----------------	--

Description

Implements the smooth LASSO estimator for the function-on-function linear regression model described in Centofanti et al. (2022) [doi:10.1016/j.csda.2022.107556](https://doi.org/10.1016/j.csda.2022.107556).

Details

Package: slasso
 Type: Package
 Version: 1.0.1
 Date: 2026-01-19
 License: GPL (>= 3)

Author(s)

Fabio Centofanti, Matteo Fontana, Antonio Lepore, Simone Vantini

References

Centofanti, F., Fontana, M., Lepore, A., & Vantini, S. (2022). Smooth lasso estimator for the function-on-function linear regression model. *Computational Statistics & Data Analysis*, 176, 107556.

See Also

[slasso.fr](#), [slasso.fr_cv](#)

Examples

```

library(slasso)
data<-simulate_data("Scenario II",n_obs=150)
X_fd=data$X_fd
Y_fd=data$Y_fd
domain=c(0,1)
n_basis_s<-30
n_basis_t<-30
breaks_s<-seq(0,1,length.out = (n_basis_s-2))
breaks_t<-seq(0,1,length.out = (n_basis_t-2))
basis_s <- fda::create.bspline.basis(domain,breaks=breaks_s)
basis_t <- fda::create.bspline.basis(domain,breaks=breaks_t)

mod_slasso_cv<-slasso.fr_cv(Y_fd = Y_fd,X_fd=X_fd,basis_s=basis_s,basis_t=basis_t,
lambda_L_vec = 10^seq(0,1,by=1),lambda_s_vec = 10^-9,lambda_t_vec = 10^-7,
B0=NULL,max_iterations=10,K=2,invisible=1,ncores=1)
mod_slasso<-slasso.fr(Y_fd = Y_fd,X_fd=X_fd,basis_s=basis_s,basis_t=basis_t,
lambda_L = 10^0.7,lambda_s =10^-5,lambda_t = 10^-6,B0 =NULL,invisible=1,max_iterations=10)

plot(mod_slasso_cv)
plot(mod_slasso)

```

plot.slasso

Plot the results of the S-LASSO method

Description

This function provides plots of the S-LASSO coefficient function estimate when applied to the output of `slasso.fr`, whereas provides the cross-validation plots when applied to the output of `slasso.fr_cv`. In the latter case the first plot displays the CV values as a function of `lambda_L`, `lambda_s` and `lambda_t`, and the second plot displays the CV values as a function of `lambda_L` with `lambda_s` and `lambda_t` fixed at their optimal values.

Usage

```

## S3 method for class 'slasso_cv'
plot(x, ...)

## S3 method for class 'slasso'
plot(x, ...)

```

Arguments

`x` The output of either `slasso.fr_cv` or `slasso.fr`.

`...` No additional parameters, called for side effects.

Value

No return value, called for side effects.

Examples

```
library(slasso)
data<-simulate_data("Scenario II",n_obs=150)
X_fd=data$X_fd
Y_fd=data$Y_fd
domain=c(0,1)
n_basis_s<-30
n_basis_t<-30
breaks_s<-seq(0,1,length.out = (n_basis_s-2))
breaks_t<-seq(0,1,length.out = (n_basis_t-2))
basis_s <- fda::create.bspline.basis(domain,breaks=breaks_s)
basis_t <- fda::create.bspline.basis(domain,breaks=breaks_t)
mod_slasso<-slasso.fr(Y_fd = Y_fd,X_fd=X_fd,basis_s=basis_s,basis_t=basis_t,
lambda_L = -1.5,lambda_s =-8,lambda_t = -7,B0 =NULL,invisible=1,max_iterations=10)
plot(mod_slasso)
```

simulate_data	<i>Simulate data through the function-on-function linear regression model</i>
---------------	---

Description

Generate synthetic data as in the simulation study of Centofanti et al. (2022).

Usage

```
simulate_data(scenario, n_obs = 3000, type_x = "Bspline")
```

Arguments

scenario	A character strings indicating the scenario considered. It could be "Scenario I", "Scenario II", "Scenario III", and "Scenario IV".
n_obs	Number of observations.
type_x	Covariate generating mechanism, either Bspline or Brownian.

Value

A list containing the following arguments:

X: Covariate matrix, where the rows correspond to argument values and columns to replications.

Y: Response matrix, where the rows correspond to argument values and columns to replications.

X_fd: Coovariate functions.

Y_fd: Response functions.

cclus: True cluster membership vector.

References

Centofanti, F., Fontana, M., Lepore, A., & Vantini, S. (2022). Smooth lasso estimator for the function-on-function linear regression model. *Computational Statistics & Data Analysis*, 176, 107556.

Examples

```
library(slasso)
data<-simulate_data("Scenario II",n_obs=150)
```

slasso.fr	<i>Smooth LASSO estimator for the function-on-function linear regression model</i>
-----------	--

Description

The smooth LASSO (S-LASSO) method for the function-on-function linear regression model provides interpretable coefficient function estimates that are both locally sparse and smooth (Centofanti et al., 2020).

Usage

```
slasso.fr(
  Y_fd,
  X_fd,
  basis_s,
  basis_t,
  lambda_L,
  lambda_s,
  lambda_t,
  B0 = NULL,
  ...
)
```

Arguments

Y_fd	An object of class fd corresponding to the response functions.
X_fd	An object of class fd corresponding to the covariate functions.
basis_s	B-splines basis along the s-direction of class basisfd.
basis_t	B-splines basis along the t-direction of class basisfd.
lambda_L	Regularization parameter of the functional LASSO penalty.
lambda_s	Regularization parameter of the smoothness penalty along the s-direction.
lambda_t	Regularization parameter of the smoothness penalty along the t-direction.
B0	Initial estimator of the basis coefficients matrix of the coefficient function. Should have dimensions in accordance with the basis dimensions of basis_s and basis_t.
...	Other arguments to be passed to the Orthant-Wise Limited-memory Quasi-Newton optimization function. See the lbfgs help page of the package lbfgs.

Value

A list containing the following arguments:

- B: The basis coefficients matrix estimate of the coefficient function.
- Beta_hat_fd: The coefficient function estimate of class bifd.
- alpha: The intercept function estimate.
- lambda_s_L: Regularization parameter of the functional LASSO penalty.
- lambda_s: Regularization parameter of the smoothness penalty along the s-direction.
- lambda_t: Regularization parameter of the smoothness penalty along the t-direction.
- Y_fd: The response functions.
- X_fd: The covariate functions.
- per_0: The fraction of domain where the coefficient function is zero.
- type: The output type.

References

Centofanti, F., Fontana, M., Lepore, A., & Vantini, S. (2022). Smooth lasso estimator for the function-on-function linear regression model. *Computational Statistics & Data Analysis*, 176, 107556.

See Also

[slasso.fr_cv](#)

Examples

```
library(slasso)
data<-simulate_data("Scenario II",n_obs=150)
X_fd=data$X_fd
Y_fd=data$Y_fd
domain=c(0,1)
n_basis_s<-30
n_basis_t<-30
breaks_s<-seq(0,1,length.out = (n_basis_s-2))
breaks_t<-seq(0,1,length.out = (n_basis_t-2))
basis_s <- fda::create.bspline.basis(domain,breaks=breaks_s)
basis_t <- fda::create.bspline.basis(domain,breaks=breaks_t)
mod_slasso<-slasso.fr(Y_fd = Y_fd,X_fd=X_fd,basis_s=basis_s,basis_t=basis_t,
lambda_L = -1.5,lambda_s =-8,lambda_t = -7,B0 =NULL,invisible=1,max_iterations=10)
```

slasso.fr_cv

*Cross-validation for the S-LASSO estimator***Description**

K-fold cross-validation procedure to choose the tuning parameters for the S-LASSO estimator (Centofanti et al., 2020).

Usage

```
slasso.fr_cv(
  Y_fd,
  X_fd,
  basis_s,
  basis_t,
  K = 10,
  kss_rule_par = 0.5,
  lambda_L_vec = NULL,
  lambda_s_vec = NULL,
  lambda_t_vec = NULL,
  B0 = NULL,
  ncores = 1,
  ...
)
```

Arguments

<code>Y_fd</code>	An object of class <code>fd</code> corresponding to the response functions.
<code>X_fd</code>	An object of class <code>fd</code> corresponding to the covariate functions.
<code>basis_s</code>	B-splines basis along the <code>s</code> -direction of class <code>basisfd</code> .
<code>basis_t</code>	B-splines basis along the <code>t</code> -direction of class <code>basisfd</code> .
<code>K</code>	Number of folds. Default is 10.
<code>kss_rule_par</code>	Parameter of the k-standard error rule. If <code>kss_rule_par=0</code> the tuning parameters that minimize the estimated prediction error are chosen. Default is 0.5.
<code>lambda_L_vec</code>	Vector of regularization parameters of the functional LASSO penalty.
<code>lambda_s_vec</code>	Vector of regularization parameters of the smoothness penalty along the <code>s</code> -direction.
<code>lambda_t_vec</code>	Vector of regularization parameters of the smoothness penalty along the <code>t</code> -direction.
<code>B0</code>	Initial estimator of the basis coefficients matrix of the coefficient function. Should have dimensions in accordance with the basis dimensions of <code>basis_s</code> and <code>basis_t</code> .
<code>ncores</code>	If <code>ncores>1</code> , then parallel computing is used, with <code>ncores</code> cores. Default is 1.
<code>...</code>	Other arguments to be passed to the Orthant-Wise Limited-memory Quasi-Newton optimization function. See the <code>lbfgs</code> help page of the package <code>lbfgs</code> .

Value

A list containing the following arguments:

- `lambda_opt_vec`: Vector of optimal tuning parameters.
- `CV`: Estimated prediction errors.
- `CV_sd`: Standard errors of the estimated prediction errors.
- `per_0`: The fractions of domain where the coefficient function is zero for all the tuning parameters combinations.
- `comb_list`: The combinations of `lambda_L`, `lambda_s` and `lambda_t` explored.
- `Y_fd`: The response functions.
- `X_fd`: The covariate functions.

References

Centofanti, F., Fontana, M., Lepore, A., & Vantini, S. (2022). Smooth lasso estimator for the function-on-function linear regression model. *Computational Statistics & Data Analysis*, 176, 107556.

See Also

[slasso.fr](#)

Examples

```
library(slasso)
data<-simulate_data("Scenario II",n_obs=150)
X_fd=data$X_fd
Y_fd=data$Y_fd
domain=c(0,1)
n_basis_s<-60
n_basis_t<-60
breaks_s<-seq(0,1,length.out = (n_basis_s-2))
breaks_t<-seq(0,1,length.out = (n_basis_t-2))
basis_s <- fda::create.bspline.basis(domain,breaks=breaks_s)
basis_t <- fda::create.bspline.basis(domain,breaks=breaks_t)
mod_slasso_cv<-slasso.fr_cv(Y_fd = Y_fd,X_fd=X_fd,basis_s=basis_s,basis_t=basis_t,
lambda_L_vec=seq(0,1,by=1),lambda_s_vec=c(-9),lambda_t_vec=-7,B0=NULL,
max_iterations=10,K=2,invisible=1,ncores=1)
```

Index

`plot.slasso`, 3
`plot.slasso_cv` (`plot.slasso`), 3

`simulate_data`, 4
`slasso` (`slasso-package`), 2
`slasso-package`, 2
`slasso.fr`, 2, 5, 8
`slasso.fr_cv`, 2, 6, 7