

# Package ‘StabilizedRegression’

May 7, 2026

**Title** Stabilizing Regression and Variable Selection

**Version** 1.1

**Description** Contains an implementation of 'StabilizedRegression', a regression framework for heterogeneous data introduced in Pfister et al. (2021) <doi:10.48550/arXiv.1911.01850>. The procedure uses averaging to estimate a regression of a set of predictors X on a response variable Y by enforcing stability with respect to a given environment variable. The resulting regression leads to a variable selection procedure which allows to distinguish between stable and unstable predictors. The package further implements a visualization technique which illustrates the trade-off between stability and predictiveness of individual predictors.

**BugReports** <https://github.com/NiklasPfister/StabilizedRegression-R/issues>

**Depends** R (>= 3.5)

**License** GPL-3

**Encoding** UTF-8

**RoxygenNote** 7.2.0

**Imports** MASS, R6, glmnet, corpcor, ggplot2, ggrepel

**NeedsCompilation** no

**Author** Niklas Pfister [aut, cre],  
Evan Williams [ctb]

**Maintainer** Niklas Pfister <np@math.ku.dk>

**Repository** CRAN

**Date/Publication** 2022-06-30 08:40:02 UTC

## Contents

coef.StabilizedRegression . . . . .	2
learn_network . . . . .	2
linear_regressor . . . . .	4
plot.SRanalysis . . . . .	6
predict.StabilizedRegression . . . . .	6
SRanalysis . . . . .	7
StabilizedRegression . . . . .	9

<b>Index</b>	<b>12</b>
--------------	-----------

---

```
coef.StabilizedRegression
      coefficients function
```

---

### Description

Coefficients functions for 'StabilizedRegression' objects.

### Usage

```
## S3 method for class 'StabilizedRegression'
coef(object, predictive_model = FALSE, ...)
```

### Arguments

```
object          object of class 'StabilizedRegression'.
predictive_model  boolean specifying whether to use the
...              additional arguments affecting the summary produced.
```

### Author(s)

Niklas Pfister

---

```
learn_network      Learn network model
```

---

### Description

Learn a network model for a collection of variables.

### Usage

```
learn_network(
  X,
  A = NA,
  method = "correlation",
  resampling_method = "stability_selection",
  numB = 100,
  cutoff = 0,
  pars = list(m = ncol(X), B = NA, alpha_stab = 0.05, alpha_pred = 0.05, size_weight =
    "linear", use_resampling = FALSE, prescreen_size = nrow(X) - 1, prescreen_type =
    "correlation", stab_test = "exact", pred_score = "mse", variable_importance =
    "scaled_coefficient"),
  verbose = 0,
  cores = 1
)
```

**Arguments**

X	data matrix. Numeric matrix of size n times d, where columns correspond to individual variables.
A	stabilizing variable. Numeric vector of length n which can be interpreted as a factor.
method	specifies which method to use. "SR" for Stabilized Regression (both standard and predictive version), "SRstab" for only the standard version of SR, "SRpred" for only the predictive version of SR, "OLS" for linear OLS regression, "lasso" for Lasso and "correlation" for correlation test.
resampling_method	specifies which resampling method to use. Should be one of "none", "stability_selection" or "permutation".
numB	number of resamples to use.
cutoff	tuning parameter used in stability selection to determine which sets count as selected.
pars	list of additional parameters passed to SR regression. See <a href="#">StabilizedRegression</a> for more details.
verbose	0 for no output, 1 for text output and 2 for text and diagnostic plots.
cores	number of cores to use in resampling step.

**Details**

Uses `StabilizedRegression`, `Lasso` or `correlation` to construct a node-wise network between all variables in X.

**Value**

A list consisting of the following elements

Amat	adjacency matrix, where <code>Amat[i,j]</code> is a score (depending on the <code>resampling_method</code> ) for the edge from i to j. For "stability_selection" scores correspond to selection probabilities, for "permutation" scores correspond to permutation p-values and for "none" scores correspond to variable importance of the method.
p	Total number of potential edges which can be used to compute upper bound on false discovery rate (only computed if <code>resampling_method == "stability_selection"</code> ).
qest	Average number of selected edges in stability selection, which can be used to compute upper bound on false discovery rate (only computed if <code>resampling_method == "stability_selection"</code> ).

If `method=="SR"` result is a list with two entries `SRstab` and `SRpred` each consisting of a list of the form described above.

**Author(s)**

Niklas Pfister

**Examples**

```
## Example
set.seed(1)
X1 <- rnorm(200)
X2 <- X1 + rnorm(200)
X3 <- 0.5 * X1 + X2 + 0.2 * c(rnorm(100), rnorm(100)+20)

X <- cbind(X1, X2, X3)
A <- as.factor(rep(c(0, 1), each=100))

network <- learn_network(X, A, method="SR", resampling_method="none")

print(network[[1]]$Amat)
print(network[[2]]$Amat)
```

---

linear\_regressor

*R6 Class Representing a Linear Regression*


---

**Description**

An R6-class for linear regression that is used within the StabilizedRegression framework.

Currently this is the only regression procedure that has been implemented. In order to extend the StabilizedRegression framework to a different regression procedure a custom R6-class with the same structure as this function can be written and used within StabilizedRegression.

**Details**

Constructor method initializes a linear regression object specifying on which subset of variables  $S$  to fit the regression and which type of stability test and prediction score to compute. The methods `fit()` and `predict()` can be applied to the object to fit and predict, respectively.

**Public fields**

`estimator` Numeric vector of regression coefficients.

`S` Numeric vector specifying the subset of variables to perform regression on.

`scores` Numeric vector of fitted stability and prediction scores.

`pars` List specifying the stability test via `test` and prediction score via `pred_score`.

**Methods****Public methods:**

- `linear_regressor$new()`
- `linear_regressor$fit()`
- `linear_regressor$predict()`
- `linear_regressor$clone()`

**Method** `new()`: Create a new `linear_regression` object.

*Usage:*

```
linear_regressor$new(  
  S = numeric(),  
  pars = list(test = "mean", pred_score = c("mse", "mse"))  
)
```

*Arguments:*

S Subset of variables.

pars Parameters.

*Returns:* A new 'linear\_regression' object.

**Method** `fit()`: Fit a 'linear\_regression' object on data and computes the stability and prediction scores.

*Usage:*

```
linear_regressor$fit(X, Y, A, extra = NA)
```

*Arguments:*

X Predictor matrix.

Y response vector.

A environment indicator.

extra not required (placeholder)

*Returns:* A fitted 'linear\_regression' object.

**Method** `predict()`: Predict using a fitted 'linear\_regression' object.

*Usage:*

```
linear_regressor$predict(X)
```

*Arguments:*

X Predictor matrix on which to predict response.

*Returns:* Numeric vector of predicted response.

**Method** `clone()`: The objects of this class are cloneable with this method.

*Usage:*

```
linear_regressor$clone(deep = FALSE)
```

*Arguments:*

deep Whether to make a deep clone.

## Author(s)

Niklas Pfister

---

plot.SRanalysis      *plot function*

---

### Description

Plot functions for 'SRanalysis' objects. Allows to visualize the stability and predictiveness trade-off of individual predictors. Green area corresponds to area for which stability selection bounds the false discovery rate (FDR) by 1.

### Usage

```
## S3 method for class 'SRanalysis'
plot(x, x_axis = "SRdiff", varnames = NA, labels = FALSE, ...)
```

### Arguments

x	object of class 'SRanalysis'.
x_axis	either "SRdiff" or "SRpred".
varnames	vector of variables names given in same ordering as columns of X. If NA the variable names saved in the SRanalysis object are used.
labels	boolean specifying whether to print names for all variables with selection probability greater than 0.5. Only works if varnames has been specified.
...	arguments to be passed to or from other methods.

### Author(s)

Niklas Pfister

---

predict.StabilizedRegression  
*predict function*

---

### Description

Predict functions for 'StabilizedRegression' objects.

### Usage

```
## S3 method for class 'StabilizedRegression'
predict(object, newdata, predictive_model = FALSE, ...)
```

**Arguments**

object	object of class 'StabilizedRegression'.
newdata	matrix or data.frame for which the response should be predicted.
predictive_model	boolean. If TRUE the model SR (pred) is used to predict, if FALSE the model SR is used.
...	additional arguments affecting the prediction produced.

**Author(s)**

Niklas Pfister

---

SRanalysis

*Stability analysis*

---

**Description**

Stability analysis based on stabilized regression used to analyze the trade-off between stability and predictiveness of individual predictors.

**Usage**

```
SRanalysis(
  X,
  Y,
  A,
  num_reps = 100,
  pred_scores = c("mse", "mse_env"),
  prescreen_types = c("correlation", "correlation_env"),
  pars_SR = list(m = ncol(X), B = 100, alpha_stab = 0.05, alpha_pred = 0.05,
    size_weight = "linear", use_resampling = FALSE, prescreen_size = NA, stab_test =
    "exact", variable_importance = "scaled_coefficient"),
  threshold = 0,
  cores = 1,
  verbose = 0,
  seed = NA
)
```

**Arguments**

X	predictor matrix. Numeric matrix of size n times d, where columns correspond to individual predictors.
Y	response variable. Numeric vector of length n.
A	stabilizing variable. Numeric vector of length n which can be interpreted as a factor.

num_reps	number of resamples to use in stability selection.
pred_scores	character vector of length 2, specifying the pred_score for SR and SRpred.
prescreen_types	character vector of length 2, specifying the prescreen_type for SR and SRpred.
pars_SR	list of all remaining parameters going into StabilizedRegression. compute_predictive, pred_score and prescreen_type are ignored.
threshold	numeric value between 0 and 1, specifying in stability selection at which value to select variables.
cores	number of cores used in mclapply.
verbose	0 for no output, 1 for text output and 2 for text and diagnostic plots.
seed	fix the seed value at the beginning of the function.

### Details

This function performs two version of StabilizedRegression: SR which selects a stable and predictive model and SRpred which fits a plain predictive model. Stability selection is then performed using the variable importance measures from both these methods and from their difference SRdiff as variable selection criterion. This allows to distinguish between which predictive variables are stable and which are unstable with respect to the stabilizing variable A. The results can be visualized by plotting the resulting object using the plot() function.

Due to the resampling this function can be quite computationally involved, we therefore recommend making use of the cores parameter for parallel computations.

### Value

Object of class 'SRanalysis' consisting of the following elements

results	List of stability selection results for for SR, SRpred and SRdiff.
varnames	Vector of variable names taken from the column names of X.
avgcoefsign_SR	Vector of average coefficient signs for SR
avgcoefsign_SRpred	Vector of average coefficient signs for SRpred

### Author(s)

Niklas Pfister

### References

Pfister, N., E. Williams, R. Aebbersold, J. Peters and P. Buhlmann (2019). Stabilizing Variable Selection and Regression. arXiv preprint arXiv:1911.01850.

**Examples**

```
## Example
set.seed(1)
X1 <- rnorm(200)
Y <- X1 + rnorm(200)
X2 <- 0.5 * X1 + Y + 0.2 * c(rnorm(100), rnorm(100)+3)

X <- cbind(X1, X2)
A <- as.factor(rep(c(0, 1), each=100))

obj <- SRanalysis(X, Y, A, 10,
                 pars_SR=list(B=NA))
plot(obj, varnames = c("X1", "X2"), labels=TRUE)
print(obj$results)
```

---

StabilizedRegression    *StabilizedRegression*

---

**Description**

StabilizedRegression based on linear OLS

**Usage**

```
StabilizedRegression(
  X,
  Y,
  A,
  pars = list(m = ncol(X), B = 100, alpha_stab = 0.05, alpha_pred = 0.05, size_weight =
    "linear", compute_predictive_model = TRUE, use_resampling = FALSE, prescreen_size =
    NA, prescreen_type = "correlation", stab_test = "exact", pred_score = "mse", topk =
    1, variable_importance = "scaled_coefficient"),
  verbose = 0,
  seed = NA
)
```

**Arguments**

X	predictor matrix. Numeric matrix of size n times d, where columns correspond to individual predictors.
Y	response variable. Numeric vector of length n.
A	stabilizing variable. Numeric vector of length n which can be interpreted as a factor.
pars	list of additional parameters. m (default ncol(X)) integer specifying the largest possible subset size. B (default 100) integer specifying the number of random subsets to sample, if NA all subsets will be used. alpha_stab (default 0.05) value between 0 and 1 specifying the stability cutoff. alpha_pred (default

0.05) value between 0 and 1 specifying the predictive cutoff. `size_weight` (default "linear") one of the strings "linear", "constant", "quadratic", "rbf" or numeric weight vector specifying a probability for each potential set size from 1 to `m`. `compute_predictive_model` (default TRUE) boolean specifying whether to additionally compute SR (pred) and SR (diff) as well. `prescreen_size` (default NA) integer specifying the number of variables to screen down to before applying SR, if NA then no screening is applied. `prescreen_type` (default "correlation") one of the strings "correlation", "ols", "lasso", "deconfounding", "correlation\_env", "deconfounding\_env" specifying the type of screening. `stab_test` (default "exact") specifies which stability test to use. Either "exact" for a Bonferroni-corrected version of Chow's test, "mean\_sres" a mean test based on resampling of the scaled residuals or "meanvar\_sres" a mean and variance test based on resampling of the scaled residuals. `pred_score` (default "mse") specifies the prediction score. Either "mse" for the mean squared error, "mse\_env" for the environment-wise best mean squared error, "aic" for the Akaike information criterion or "bic" for the Bayesian information criterion. `topk` (default 1) is a tuning parameter that can be used to increase the number of predictive sets. It should be an integer value, where higher values lead to more accepted sets based on the predictive cutoff. `variable_importance` (default "scaled\_coefficient") specifies the type of variable ranking. Either "weighted" for a weighted average of all selected subsets, "scaled\_coefficient" for a ranking based on the scaled average regression parameter or "permutation" for a permutation based ranking.

`verbose` 0 for no output, 1 for text output and 2 for text and diagnostic plots.  
`seed` fix the seed value at the beginning of the function.

### Details

Performs a linear regression of a response  $Y$  on a set of predictors  $X$  while ensuring stability across different values of a stabilizing variable  $A$ .

### Value

Object of class 'StabilizedRegression' consisting of the following elements

`learner_list` List of all fitted linear OLS regressions (fitted R6 'linear\_regression' objects).  
`weighting` Weighting of the individual regressions in SR.  
`weighting_pred` Weighting of the individual regressions in SR (pred). Only computed if `compute_predictive_model` is TRUE.  
`variable_importance` Variable importance measure for all predictors based on SR.  
`variable_importance_pred` Variable importance measure for all predictors based on SR (pred). Only computed if `compute_predictive_model` is TRUE.  
`variable_importance_diff` Variable importance measure for all predictors based on difference between SR and SR (pred). Only computed if `compute_predictive_model` is TRUE.

**Author(s)**

Niklas Pfister

**References**

Pfister, N., E. Williams, R. Aebersold, J. Peters and P. B"uhlmann (2019). Stabilizing Variable Selection and Regression. arXiv preprint arXiv:1911.01850.

**Examples**

```
## Example
set.seed(1)
X1 <- rnorm(200)
Y <- X1 + rnorm(200)
X2 <- 0.5 * X1 + Y + 0.2 * c(rnorm(100), rnorm(100)+2)

X <- cbind(X1, X2)
A <- as.factor(rep(c(0, 1), each=100))

fit_sr <- StabilizedRegression(X, Y, A, pars=list(B=NA))
fit_lm <- lm(Y ~ X)

print(paste("Coefficients of SR:", toString(coefficients(fit_sr))))
print(paste("Coefficients of SR (pred):", toString(coefficients(fit_sr, predictive_model=TRUE))))
print(paste("Coefficients of OLS:", toString(coefficients(fit_lm))))
```

# Index

`coef.StabilizedRegression`, [2](#)

`learn_network`, [2](#)

`linear_regressor`, [4](#)

`plot.SRanalysis`, [6](#)

`predict.StabilizedRegression`, [6](#)

`SRanalysis`, [7](#)

`StabilizedRegression`, [3](#), [9](#)