

# Package ‘SelectBoost’

January 20, 2025

**Type** Package

**Title** A General Algorithm to Enhance the Performance of Variable Selection Methods in Correlated Datasets

**Version** 2.2.2

**Date** 2022-11-29

**Depends** R (>= 2.10)

**Imports** lars, glmnet, igraph, parallel, msgps, Rfast, methods, Cascade, graphics, grDevices, varbvs, spls, abind

**Suggests** knitr, markdown, rmarkdown, mixOmics, CascadeData

**Author** Frederic Bertrand [cre, aut] (<<https://orcid.org/0000-0002-0837-8281>>),  
Myriam Maumy-Bertrand [aut] (<<https://orcid.org/0000-0002-4615-1512>>),  
Ismail Aouadi [ctb],  
Nicolas Jung [ctb]

**Maintainer** Frederic Bertrand <[frederic.bertrand@utt.fr](mailto:frederic.bertrand@utt.fr)>

**Description** An implementation of the selectboost algorithm (Bertrand et al. 2020, 'Bioinformatics', <[doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)>), which is a general algorithm that improves the precision of any existing variable selection method. This algorithm is based on highly intensive simulations and takes into account the correlation structure of the data. It can either produce a confidence index for variable selection or it can be used in an experimental design planning perspective.

**License** GPL-3

**Encoding** UTF-8

**Classification/MSC** 62H11, 62J12, 62J99

**LazyData** true

**VignetteBuilder** knitr

**RoxygenNote** 7.2.1

**URL** <https://fbertran.github.io/SelectBoost/>,  
<https://github.com/fbertran/SelectBoost/>

**BugReports** <https://github.com/fbertran/SelectBoost/issues/>

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2022-11-30 09:10:06 UTC

## Contents

AICc_BIC_glmnetB . . . . .	2
auto.analyze . . . . .	4
autoboost . . . . .	5
autoboost.res.x . . . . .	8
autoboost.res.x.adapt . . . . .	8
autoboost.res.x2 . . . . .	8
autoboost.res.x2.adapt . . . . .	9
boost . . . . .	9
Cascade_confidence . . . . .	12
Cascade_example . . . . .	13
fastboost . . . . .	14
fastboost.res.x . . . . .	16
fastboost.res.x.adapt . . . . .	16
fastboost.res.x2 . . . . .	17
fastboost.res.x2.adapt . . . . .	17
force.non.inc . . . . .	17
group_func_1 . . . . .	18
group_func_2 . . . . .	19
miscplot . . . . .	20
network.confidence-class . . . . .	21
plot.selectboost . . . . .	22
plot.summary.selectboost . . . . .	23
plot_selectboost_cascade . . . . .	24
results_simuls_reverse_engineering_v3 . . . . .	26
SelectBoost . . . . .	31
selectboost_cascade . . . . .	32
simulation . . . . .	34
summary.selectboost . . . . .	36
trajC0 . . . . .	38
var_select . . . . .	40
var_select_all . . . . .	44
<b>Index</b>	<b>48</b>

---

AICc\_BIC\_glmnetB

*AICc and BIC for glmnet logistic models*


---

### Description

Compute AICc and BIC for glmnet logistic models.

**Usage**

```
rerr(v1, v2)

ridge_logistic(X, Y, lambda, beta0, beta, maxiter = 1000, tol = 1e-10)

BIC_glmnetB(Z, Y, glmnet.model, alpha, modelSet, reducer = "median")

AICc_glmnetB(Z, Y, glmnet.model, alpha, modelSet, reducer = "median")
```

**Arguments**

v1	A numeric vector.
v2	A numeric vector.
X	A numeric matrix
Y	A numeric 0/1 vector.
lambda	A numeric value.
beta0	A numeric value Initial intercept value.
beta	A numeric vector. Initial coefficient values.
maxiter	A numeric value. Maximum number of iterations.
tol	A numeric value. Tolerance value.
Z	A numeric matrix
glmnet.model	A fitted glmnet model.
alpha	A numeric value.
modelSet	Modelset to consider.
reducer	A character value. Reducer function. Either 'median' or 'mean'.

**Details**

Calculate AICc and BIC for glmnet logistic models from the glmnetB function of the package rLogistic <https://github.com/echi/rLogistic> and adapted to deal with non finite exponential values in AICc and BIC computations

**Value**

A list relevant to model selection.

**Author(s)**

Frederic Bertrand, <frederic.bertrand@utt.fr>

**References**

*Robust Parametric Classification and Variable Selection by a Minimum Distance Criterion*, Chi and Scott, Journal of Computational and Graphical Statistics, **23**(1), 2014, p111–128, doi:10.1080/10618600.2012.737296.

**See Also**[var\\_select](#)**Examples**

```

set.seed(314)
xran=matrix(rnorm(150),30,5)
ybin=sample(0:1,30,replace=TRUE)
glmnet.fit <- glmnet.fit <- glmnet::glmnet(xran,ybin,family="binomial",standardize=FALSE)
set.seed(314)
rerr(1:10,10:1)

set.seed(314)
ridge_logistic(xran,ybin,lambda=.5,beta0=rnorm(5),beta=rnorm(5,1))

set.seed(314)
if(is.factor(ybin)){ynum=unclass(ybin)-1} else {ynum=ybin}
subSample <- 1:min(ncol(xran),100)
BIC_glmnetB(xran,ynum,glmnet.fit,alpha=1,subSample, reducer='median')

set.seed(314)
if(is.factor(ybin)){ynum=unclass(ybin)-1} else {ynum=ybin}
subSample <- 1:min(ncol(xran),100)
AICc_glmnetB(xran,ynum,glmnet.fit,alpha=1,subSample, reducer='median')

```

---

`auto.analyze`*Find limits for selectboost analysis*

---

**Description**

Find limits for selectboost analysis.

**Usage**

```

auto.analyze(x, ...)

## S3 method for class 'selectboost'
auto.analyze(x, ...)

```

**Arguments**

`x` Numerical matrix. Selectboost object.

`...` . Passed to the `summary.selectboost` function.

## Details

`plot.summary.selectboost` returns an invisible list and creates four graphics. Two plots the proportion of selection with respect to `c0` (by step or according to real scale). On the third graph, no bar means a proportion of selection less than `prop.level`. Confidence intervals are computed at the `conf.int.level` level. Barplot of the confidence index ( $1 - \min(c0, \text{such that } \text{proportion}lc0 > \text{conf.threshold})$ ).

## Value

list of results.

## Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

## References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. doi:[10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

## See Also

[fastboost](#) and [autoboost](#)

Other Selectboost analyze functions: [plot.summary.selectboost\(\)](#), [trajC0\(\)](#)

## Examples

```
data(autoboost.res.x)
auto.analyze(autoboost.res.x)

data(autoboost.res.x2)
auto.analyze(autoboost.res.x2)
```

---

autoboost

*Autoboost*

---

## Description

All in one use of `selectboost` that avoids redundant fitting of distributions and saves some memory.

**Usage**

```

autoboost(
  X,
  Y,
  ncores = 4,
  group = group_func_1,
  func = lasso_msgps_AICc,
  corrfunc = "cor",
  use.parallel = FALSE,
  B = 100,
  step.num = 0.1,
  step.limit = "none",
  risk = 0.05,
  verbose = FALSE,
  step.scale = "quantile",
  normalize = TRUE,
  steps.seq = NULL,
  debug = FALSE,
  version = "lars",
  ...
)

```

**Arguments**

X	Numerical matrix. Matrix of the variables.
Y	Numerical vector or factor. Response vector.
ncores	Numerical value. Number of cores for parallel computing. Defaults to 4.
group	Function. The grouping function. Defaults to group_func_1.
func	Function. The variable selection function. Defaults to lasso_msgps_AICc.
corrfunc	Character value or function. Used to compute associations between the variables. Defaults to "cor".
use.parallel	Boolean. To use parallel computing (doMC) download the extended package from Github. Set to FALSE.
B	Numerical value. Number of resampled fits of the model. Defaults to 100.
step.num	Numerical value. Step value for the c0 sequence. Defaults to 0.1.
step.limit	Character value. If "Pearson", truncates the c0 sequence using a Pearson based p-value. Defaults to "none".
risk	Numerical value. Risk level when finding limits based on c0=0 values. Defaults to 0.05.
verbose	Boolean. Defaults to FALSE.
step.scale	Character value. How to compute the c0 sequence if not user-provided: either "quantile" or "linear". Defaults to "quantile".
normalize	Boolean. Shall the X matrix be centered and scaled? Defaults to TRUE.
steps.seq	Numeric vector. User provided sequence of c0 values to use. Defaults to NULL.

debug	Boolean value. If more results are required. Defaults to FALSE.
version	Character value. Passed to the <code>boost.select</code> function. Defaults to <code>lars</code>
...	. Arguments passed to the variable selection function used in <code>boost.apply</code> .

### Details

`autoboost` returns a numeric matrix. For each of the variable (column) and each of the `c0` (row), the entry is proportion of times that the variable was selected among the `B` resampled fits of the model. Fitting to the same group of variables is only performed once (even if it occurred for another value of `c0`), which greatly speeds up the algorithm.

### Value

A numeric matrix with attributes.

### Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

### References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

### See Also

[boost](#), [fastboost](#), [plot.selectboost](#)

Other Selectboost functions: [boost](#), [fastboost\(\)](#), [plot\\_selectboost\\_cascade](#), [selectboost\\_cascade](#)

### Examples

```
set.seed(314)
xran=matrix(rnorm(75),15,5)
ybin=sample(0:1,15,replace=TRUE)
yran=rnorm(15)
set.seed(314)
#For quick test purpose, not meaningful, should be run with greater value of B
#and disabling parallel computing as well
res.autoboost <- autoboost(xran,yran,B=3,use.parallel=FALSE)

autoboost(xran,yran)
#Customize resampling levels
autoboost(xran,yran,steps.seq=c(.99,.95,.9))

#Binary logistic regression
autoboost(xran,ybin,func=lasso_cv_glmnet_bin_min)
```

---

autoboost.res.x      *Autoboost lasso diabetes first order.*

---

**Description**

Result of autoboost analysis of diabetes data from *lars* package with lasso and first order model

**Usage**

autoboost.res.x

**Format**

A numerical matrix frame with 13 rows and 10 variables with attributes.

---

autoboost.res.x.adapt      *Autoboost adaptative lasso diabetes first order.*

---

**Description**

Result of autoboost analysis of diabetes data from *lars* package with adaptative lasso and first order model

**Usage**

autoboost.res.x.adapt

**Format**

A numerical matrix frame with 13 rows and 10 variables with attributes.

---

autoboost.res.x2      *Autoboost lasso diabetes second order.*

---

**Description**

Result of autoboost analysis of diabetes data from *lars* package with lasso and second order model

**Usage**

autoboost.res.x2

**Format**

A numerical matrix frame with 13 rows and 64 variables with attributes.



---

autoboost.res.x2.adapt

*Autoboost adaptative lasso diabetes second order.*


---

**Description**

Result of autoboost analysis of diabetes data from *lars* package with adaptative lasso and second order model

**Usage**

```
autoboost.res.x2.adapt
```

**Format**

A numerical matrix frame with 13 rows and 64 variables with attributes.

---

boost

*Boost step by step functions*


---

**Description**

Step by step functions to apply the selectboost algorithm.

**Usage**

```
boost.normalize(X, eps = 1e-08)
```

```
boost.compcorrs(
  Xnorm,
  corrfunc = "cor",
  verbose = FALSE,
  testvarindic = rep(TRUE, ncol(Xnorm))
)
```

```
boost.correlation_sign(Correlation_matrice, verbose = FALSE)
```

```
boost.findgroups(Correlation_matrice, group, corr = 1, verbose = FALSE)
```

```
boost.Xpass(nrowX, ncolX)
```

```
boost.adjust(
  X,
  groups,
  Correlation_sign,
```

```

Xpass = boost.Xpass(nrowX, ncolX),
verbose = FALSE,
use.parallel = FALSE,
ncores = 4
)

boost.random(
  X,
  Xpass,
  vmf.params,
  verbose = FALSE,
  B = 100,
  use.parallel = FALSE,
  ncores = 4
)

boost.apply(
  X,
  cols.simul,
  Y,
  func,
  verbose = FALSE,
  use.parallel = FALSE,
  ncores = 4,
  ...
)

boost.select(Boost.coeffs, eps = 10^(-4), version = "lars", verbose = FALSE)

```

### Arguments

<code>X</code>	Numerical matrix. Matrix of the variables.
<code>eps</code>	Numerical value. Response vector.
<code>Xnorm</code>	Numerical matrix. Needs to be centered and l2 normalized.
<code>corrfunc</code>	Character value or function. The function to compute associations between the variables.
<code>verbose</code>	Boolean. Defaults to FALSE.
<code>testvarindic</code>	Boolean vector. Compute associations for a subset of variables. By default, the scope of the computation is the whole dataset, i.e. <code>rep(TRUE, ncol(Xnorm))</code> .
<code>Correlation_matrice</code>	Numerical matrix.
<code>group</code>	Character value or function. The grouping function.
<code>corr</code>	Numerical value. Thresholding value. Defaults to 1.
<code>nrowX</code>	Numerical value
<code>ncolX</code>	Numerical value.
<code>groups</code>	List. List of groups or communities (compact form).

Correlation_sign	Numerical -1/1 matrix.
Xpass	Numerical value. Transformation matrix. Defaults to <code>boost.Xpass(nrowX, ncolX)</code> , with <code>nrowX=nrow(X)</code> and <code>ncolX=ncol(X)</code> .
use.parallel	Boolean. Defaults to <code>FALSE</code> .
ncores	Numerical value. Number of cores to use. Defaults to 4.
vmf.params	List. List of the parameters of the fitted von-Mises distributions.
B	Integer value. Number of resampling.
cols.simul	Numerical value. Transformation matrix.
Y	Numerical vector or factor. Response.
func	Function. Variable selection function.
...	. Additionnal parameters passed to the <code>func</code> function.
Boost.coeffs	Numerical matrix. l2 normed matrix of predictors.
version	Character value. "lars" (no intercept value) or "glmnet" (first coefficient is the intercept value).

## Details

`boost.normalize` returns a numeric matrix whose colun are centered and l2 normalized.

`boost.compcorrs` returns a correlation like matrix computed using the `corrfunc` function.

`boost.Xpass` returns the transformation matrix.

`boost.findgroups` returns a list of groups or communities found using the `group` function.

`boost.Xpass` returns the transformation matrix.

`boost.adjust` returns the list of the parameters of the fitted von-Mises distributions.

`boost.random` returns an array with the resampled datasets.

`boost.apply` returns a matrix with the coefficients estimated using the resampled datasets.

`boost.select` returns a vector with the proportion of times each variable was selected.

## Value

Various types depending on the function.

## Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

## References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

**See Also**

[fastboost](#), [autoboost](#)

Other Selectboost functions: [autoboost\(\)](#), [fastboost\(\)](#), [plot\\_selectboost\\_cascade](#), [selectboost\\_cascade](#)

**Examples**

```
set.seed(314)
xran=matrix(rnorm(200),20,10)
yran=rnorm(20)
xran_norm <- boost.normalize(xran)

xran_corr<- boost.compcorrs(xran_norm)

xran_corr_sign <- boost.correlation_sign(xran_corr)

xran_groups <- boost.findgroups(xran_corr, group=group_func_1, .3)
xran_groups_2 <- boost.findgroups(xran_corr, group=group_func_2, .3)

xran_Xpass <- boost.Xpass(nrow(xran_norm),ncol(xran_norm))

xran_adjust <- boost.adjust(xran_norm, xran_groups$groups, xran_corr_sign)

#Not meaningful, should be run with B>=100
xran_random <- boost.random(xran_norm, xran_Xpass, xran_adjust$vmf.params, B=5)

xran_random <- boost.random(xran_norm, xran_Xpass, xran_adjust$vmf.params, B=100)

xran_apply <- boost.apply(xran_norm, xran_random, yran, lasso_msgps_AICc)

xran_select <- boost.select(xran_apply)
```

---

Cascade\_confidence      *Confidence indices*

---

**Description**

Result for confidence indices derivation using the Cascade package

**Usage**

```
net_confidence
net_confidence_.5
net_confidence_thr
```

**Format**

A network.confidence object with four slots :

**network.confidence** The confidence matrix

**name** Names of the variables (genes)

**F** F array, see Cascade for more details

**time\_pt** Repeated measurements

**cv.subjects** Logical. Was crossvalidation carried out subjectwise?

An object of class network.confidence of length 1.

An object of class network.confidence of length 1.

---

Cascade\_example

*Simulated Cascade network and inference*

---

**Description**

Result for the reverse engineering of a simulated Cascade network

**Usage**

M

Net

Net\_inf\_C

**Format**

Three objects :

**M** Simulated microarray

**Net** Simulated network

**Net\_inf\_C** Inferred network

An object of class network of length 1.

An object of class network of length 1.

fastboost

*Fastboost***Description**

All in one use of selectboost that avoids redondant fitting of distributions and saves some memory.

**Usage**

```
fastboost(
  X,
  Y,
  ncores = 4,
  group = group_func_1,
  func = lasso_msgps_AICc,
  corrfunc = "cor",
  use.parallel = FALSE,
  B = 100,
  step.num = 0.1,
  step.limit = "none",
  verbose = FALSE,
  step.scale = "quantile",
  normalize = TRUE,
  steps.seq = NULL,
  debug = FALSE,
  version = "lars",
  c0lim = TRUE,
  ...
)
```

**Arguments**

X	Numerical matrix. Matrix of the variables.
Y	Numerical vector or factor. Response vector.
ncores	Numerical value. Number of cores for parallel computing. Defaults to 4.
group	Function. The grouping function. Defaults to group_func_1.
func	Function. The variable selection function. Defaults to lasso_msgps_AICc.
corrfunc	Character value or function. Used to compute associations between the variables. Defaults to "cor".
use.parallel	Boolean. To use parallel computing (doMC) download the extended package from Github. Set to FALSE.
B	Numerical value. Number of resampled fits of the model. Defaults to 100.
step.num	Numerical value. Step value for the c0 sequence. Defaults to 0.1.
step.limit	Defaults to "none".

verbose	Boolean. Defaults to FALSE.
step.scale	Character value. How to compute the c0 sequence if not user-provided: either "quantile" or "linear", "zoom_l", "zoom_q" and "mixed". Defaults to "quantile".
normalize	Boolean. Shall the X matrix be centered and scaled? Defaults to TRUE.
steps.seq	Numeric vector. User provided sequence of c0 values to use. Defaults to NULL.
debug	Boolean value. If more results are required. Defaults to FALSE.
version	Character value. Passed to the boost.select function. Defaults to lars
c0lim	Boolean. Shall the c0=0 and c0=1 values be used? Defaults to TRUE
...	. Arguments passed to the variable selection function used in boost.apply.

### Details

fastboost returns a numeric matrix. For each of the variable (column) and each of the c0 (row), the entry is proportion of times that the variable was selected among the B resampled fits of the model. Fitting to the same group of variables is only performed once (even if it occurred for another value of c0), which greatly speeds up the algorithm. In order to limit memory usage, fastboost uses a compact way to save the group memberships, which is especially useful with community grouping function and fairly big datasets.

### Value

A numeric matrix with attributes.

### Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

### References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. doi:10.1093/bioinformatics/btaa855

### See Also

[boost](#), [autoboost](#), [plot.selectboost](#)

Other Selectboost functions: [autoboost\(\)](#), [boost](#), [plot\\_selectboost\\_cascade](#), [selectboost\\_cascade](#)

### Examples

```
set.seed(314)
xran=matrix(rnorm(75),15,5)
ybin=sample(0:1,15,replace=TRUE)
yran=rnorm(15)
set.seed(314)
#For quick test purpose, not meaningful, should be run with greater value of B
#and disabling parallel computing as well
```

```

res.fastboost <- fastboost(xran,yran,B=3,use.parallel=FALSE)

fastboost(xran,yran)
#Customize resampling levels
fastboost(xran,yran,steps.seq=c(.99,.95,.9),c0lim=FALSE)
fastboost(xran,yran,step.scale="mixed",c0lim=TRUE)
fastboost(xran,yran,step.scale="zoom_l",c0lim=FALSE)
fastboost(xran,yran,step.scale="zoom_l",step.num = c(1,.9,.01),c0lim=FALSE)
fastboost(xran,yran,step.scale="zoom_q",c0lim=FALSE)
fastboost(xran,yran,step.scale="linear",c0lim=TRUE)
fastboost(xran,yran,step.scale="quantile",c0lim=TRUE)

#Binary logistic regression
fastboost(xran,ybin,func=lasso_cv_glmnet_bin_min)

```

---

fastboost.res.x      *Fastboost lasso diabetes first order.*

---

### Description

Result of fastboost analysis of diabetes data from *lars* package with lasso and first order model

### Usage

```
fastboost.res.x
```

### Format

A numerical matrix frame with 13 rows and 10 variables with attributes.

---

fastboost.res.x.adapt      *Fastboost adaptive lasso diabetes first order.*

---

### Description

Result of fastboost analysis of diabetes data from *lars* package with adaptive lasso and first order model

### Usage

```
fastboost.res.x.adapt
```

### Format

A numerical matrix frame with 13 rows and 10 variables with attributes.



---

fastboost.res.x2	<i>Fastboost lasso diabetes second order.</i>
------------------	---

---

**Description**

Result of fastboost analysis of diabetes data from *lars* package with lasso and second order model

**Usage**

```
fastboost.res.x2
```

**Format**

A numerical matrix frame with 13 rows and 64 variables with attributes.

---

fastboost.res.x2.adapt	<i>Fastboost adaptative lasso diabetes second order.</i>
------------------------	--

---

**Description**

Result of fastboost analysis of diabetes data from *lars* package with adaptative lasso and second order model

**Usage**

```
fastboost.res.x2.adapt
```

**Format**

A numerical matrix frame with 13 rows and 64 variables with attributes.

---

force.non.inc	<i>Non increasing post processinng step for selectboost analysis</i>
---------------	--

---

**Description**

Post processes a selectboost analysis.

**Usage**

```
force.non.inc(object)
```

**Arguments**

object            Numerical matrix. Result of selectboost (autoboost, fastboost, ...).

**Details**

force.non.inc returns a vector after ensuring that the proportion of times each variable was selected is non increasing with respect to the 1-c0 value.

**Value**

A matrix with the results.

**Author(s)**

Frederic Bertrand, <frederic.bertrand@utt.fr>

**References**

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

**See Also**

[fastboost](#), [autoboost](#)

Other Selectboost analyse functions: [plot.selectboost\(\)](#), [summary.selectboost\(\)](#)

**Examples**

```
data(autoboost.res.x)
res.fastboost.force.non.inc <- force.non.inc(autoboost.res.x)
```

---

group\_func\_1

*Generate groups by thresholding.*

---

**Description**

group\_func\_1 creates groups of variables based on thresholding the input matrix.

**Usage**

```
group_func_1(absXcor, c0)
```

**Arguments**

absXcor      A numeric matrix. The absolute value of a correlation or distance matrix.  
c0            A numeric scalar. The thresholding

**Details**

This is a function used to create a list of groups using an input matrix and a thresholding value c0. A group is made, for every column in the input matrix.

**Value**

A list with one entry: the list of groups. Attributes:

- "type": "normal"
- "length.groups" the length of each groups.

**Author(s)**

Frederic Bertrand, <frederic.bertrand@utt.fr>

**References**

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

**See Also**

[group\\_func\\_2](#) and [boost.findgroups](#)

**Examples**

```
set.seed(314)
group_func_1(cor(matrix(rnorm(50),10,5)),.4)
```

---

group\_func\_2      *Generate groups using community analysis.*

---

**Description**

group\_func\_2 creates groups of variables based on community analysis.

**Usage**

```
group_func_2(absXcor, c0)
```

### Arguments

`absXcor` A numeric matrix. The absolute value of a correlation or distance matrix.  
`c0` A numeric scalar. The thresholding

### Details

This is a function used to create a list of groups using an input matrix and a thresholding value `c0`. A group is made, for every column in the input matrix. It uses the `infomap.community` function of the `igraph` package.

### Value

A list with one entry: the list of groups. Attributes:

- "type": "normal"
- "length.groups" the length of each groups.

### Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

### References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

### See Also

[group\\_func\\_2](#) [boost.findgroups](#), [infomap.community](#) and [igraph](#).

### Examples

```
set.seed(314)
group_func_2(cor(matrix(rnorm(100), 10, 10)), .5)
```

---

miscplot

*Miscellaneous plot functions*

---

### Description

Define some additional plot functions to be used in the demos of the package.

### Usage

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

**Arguments**

`x` A numeric matrix. A matrix to be plotted.  
`...` . Additional arguments passed to the plot function.

**Details**

`matrixplot` plots a numeric matrix `x`.

**Value**

`matrixplot` returns 1.

**Author(s)**

Frederic Bertrand, <frederic.bertrand@utt.fr> with contributions from Nicolas Jung.

**References**

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. doi:[10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

**Examples**

```
set.seed(3141)
randmat=matrix(rnorm(360),60,60)
plot(randmat)
```

---

network.confidence-class

*Network confidence class.*

---

**Description**

Some details about this class and my plans for it in the body.

**Details**

**network.confidence** Matrix of confidence indices.

**name** Vector.

**array** F array

**time\_pt** Vector

**cv.subjects** Logical. Was crossvalidation carried out subjectwise?

---

plot.selectboost      *Plot selectboost object*

---

### Description

Plot a selectboost object.

### Usage

```
## S3 method for class 'selectboost'
plot(
  x,
  verbose = FALSE,
  prop.level = 0.95,
  conf.int.level = 0.95,
  conf.threshold = 0.95,
  ...
)
```

### Arguments

x	Numerical matrix. Result of selectboost (autoboost, fastboost, ...).
verbose	Boolean. Defaults to FALSE.
prop.level	Numeric value. Used to compute the proportion of selection is greater than prop.level. Defaults to .95.
conf.int.level	Numeric value. Confidence level for confidence intervals on estimated proportions of selection. Defaults to .95.
conf.threshold	Numeric value. Used to compute the number of steps (c0) for which the proportion of selection remains greater than conf.threshold. Defaults to .95.
...	. Passed to the plotting functions.

### Details

plot.selectboost returns an invisible list and creates four graphics. Two plots the proportion of selection with respect to c0 (by step or according to real scale). On the third graph, no bar means a proportion of selection less than prop.level. Confidence intervals are computed at the conf.int.level level. Barplot of the confidence index (1-min(c0, such that proportion|c0>conf.threshold)).

### Value

An invisible list.

### Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

## References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

## See Also

[fastboost](#), [autoboost](#)

Other Selectboost analyse functions: [force.non.inc\(\)](#), [summary.selectboost\(\)](#)

## Examples

```
set.seed(314)
xran=matrix(rnorm(75),15,5)
ybin=sample(0:1,15,replace=TRUE)
yran=rnorm(15)
layout(matrix(1:4,2,2))
```

```
data(autoboost.res.x)
plot(autoboost.res.x)
```

```
data(autoboost.res.x2)
plot(autoboost.res.x2)
```

---

```
plot.summary.selectboost
```

*Plot a summary of selectboost results*

---

## Description

Plot a summary of selectboost results.

## Usage

```
## S3 method for class 'summary.selectboost'
plot(x, ...)
```

## Arguments

x	Numerical matrix. Summary of selectboost object.
...	. Passed to the plotting functions.

## Details

`plot.summary.selectboost` returns an invisible list and creates four graphics. Two plots the proportion of selection with respect to `c0` (by step or according to real scale). On the third graph, no bar means a proportion of selection less than `prop.level`. Confidence intervals are computed at the `conf.int.level` level. Barplot of the confidence index ( $1 - \min(c0, \text{such that } \text{proportion} \leq \text{conf.threshold})$ ).

## Value

An invisible list.

## Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

## References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

## See Also

[fastboost](#), [autoboost](#) and [summary.selectboost](#)

Other Selectboost analyze functions: [auto.analyze\(\)](#), [trajC0\(\)](#)

## Examples

```
data(autoboost.res.x)
plot(summary(autoboost.res.x))
```

```
data(autoboost.res.x2)
plot(summary(autoboost.res.x2))
```

---

```
plot_selectboost_cascade
      plot_Selectboost_cascade
```

---

## Description

Plot result of Selectboost for Cascade inference.

## Usage

```
## S4 method for signature 'network.confidence,ANY'
plot(x, col = gray((1:99)/100, alpha = NULL), ...)
```



## Arguments

x	A network.confidence object to be plotted.
col	Colors for the plot.
...	Additional arguments passed to the heatmap function.

## Details

Extending results from the Cascade package: providing confidence indices for the reverse engineered links.

Reference for the Cascade modelling Vallat, L., Kemper, C. a., Jung, N., Maumy-Bertrand, M., Bertrand, F., Meyer, N., Pocheville, A., Fisher, J. W., Gribben, J. G. et Bahram, S. (2013). Reverse-engineering the genetic circuitry of a cancer cell with predicted intervention in chronic lymphocytic leukemia. *Proceedings of the National Academy of Sciences of the United States of America*, 110(2), 459-64.

Reference for the Cascade package Jung, N., Bertrand, F., Bahram, S., Vallat, L. et Maumy-Bertrand, M. (2014). Cascade : A R package to study, predict and simulate the diffusion of a signal through a temporal gene network. *Bioinformatics*. ISSN 13674803..

## Value

Nothing.

## Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

## References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, *Bioinformatics*, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

## See Also

[boost](#), [fastboost](#), [selectboost](#), [inference](#)

Other Selectboost functions: [autoboost\(\)](#), [boost](#), [fastboost\(\)](#), [selectboost\\_cascade](#)

## Examples

```
data(net_confidences)
plot(net_confidence)
plot(net_confidence_.5)
plot(net_confidence_thr)
```

---

results\_simuls\_reverse\_engineering\_v3

*Simulations for reverse-engineering*

---

**Description**

Result of fastboost analysis applied to biological network reverse engineering

**Usage**

test.seq\_C

test.seq\_PL

test.seq\_PL2

test.seq\_PL2\_W

test.seq\_PL2\_tW

test.seq\_PSel

test.seq\_PSel.5

test.seq\_PSel.e2

test.seq\_PSel.5.e2

test.seq\_PSel\_W

test.seq\_robust

test.seq\_PB

test.seq\_PB\_095\_075

test.seq\_PB\_075\_075

test.seq\_PB\_W

sensitivity\_C

sensitivity\_PL

sensitivity\_PL2

sensitivity\_PL2\_W

sensitivity\_PL2\_tW  
sensitivity\_PSel  
sensitivity\_PSel.5  
sensitivity\_PSel.e2  
sensitivity\_PSel.5.e2  
sensitivity\_PSel\_W  
sensitivity\_robust  
sensitivity\_PB  
sensitivity\_PB\_095\_075  
sensitivity\_PB\_075\_075  
sensitivity\_PB\_W  
predictive\_positive\_value\_C  
predictive\_positive\_value\_PL  
predictive\_positive\_value\_PL2  
predictive\_positive\_value\_PL2\_W  
predictive\_positive\_value\_PL2\_tW  
predictive\_positive\_value\_PSel  
predictive\_positive\_value\_PSel.5  
predictive\_positive\_value\_PSel.e2  
predictive\_positive\_value\_PSel.5.e2  
predictive\_positive\_value\_PSel\_W  
predictive\_positive\_value\_robust  
predictive\_positive\_value\_PB  
predictive\_positive\_value\_PB\_095\_075

predictive\_positive\_value\_PB\_075\_075

predictive\_positive\_value\_PB\_W

F\_score\_C

F\_score\_PL

F\_score\_PL2

F\_score\_PL2\_W

F\_score\_PL2\_tW

F\_score\_PSel

F\_score\_PSel.5

F\_score\_PSel.e2

F\_score\_PSel.5.e2

F\_score\_PSel\_W

F\_score\_robust

F\_score\_PB

F\_score\_PB\_095\_075

F\_score\_PB\_075\_075

F\_score\_PB\_W

nv\_C

nv\_PL

nv\_PL2

nv\_PL2\_W

nv\_PL2\_tW

nv\_PSel

nv\_PSel.5

nv\_PSe1.e2

nv\_PSe1.5.e2

nv\_PSe1\_W

nv\_robust

nv\_PB

nv\_PB\_095\_075

nv\_PB\_075\_075

nv\_PB\_W

### **Format**

A numerical matrix frame with 100 rows and 200 variables or a numerical vector of length 100.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.



An object of class `matrix` (inherits from `array`) with 100 rows and 200 columns.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.  
An object of class `numeric` of length 100.

---

SelectBoost

*SelectBoost*

---

## Description

**Motivation:** With the growth of big data, variable selection has become one of the major challenges in statistics. Although many methods have been proposed in the literature their performance in terms of recall and precision are limited in a context where the number of variables by far exceeds the number of observations or in a high correlated setting. **Results:** This package implements a new general algorithm which improves the precision of any existing variable selection method. This algorithm is based on highly intensive simulations and takes into account the correlation structure of the data. Our algorithm can either produce a confidence index for variable selection or it can be used in an experimental design planning perspective.

## References

F. Bertrand, I. Aouadi, N. Jung, R. Carapito, L. Vallat, S. Bahram, M. Maumy-Bertrand (2020). SelectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets, *Bioinformatics*. doi:10.1093/bioinformatics/btaa855

SelectBoost was used to decypher networks in C. Schleiss, [...], M. Maumy-Bertrand, S. Bahram, F. Bertrand, and L. Vallat. (2021). Temporal multiomic modelling reveals a B-cell receptor proliferative program in chronic lymphocytic leukemia. *Leukemia*.

**Examples**

```

set.seed(314)
xran=matrix(rnorm(75),15,5)
ybin=sample(0:1,15,replace=TRUE)
yran=rnorm(15)

#For quick test purpose, not meaningful, should be run with greater value of B
#(disabling parallel computing as well)
res.fastboost <- fastboost(xran,yran,B=3,use.parallel=FALSE)

fastboost(xran,yran)
#Customize resampling levels
fastboost(xran,yran,steps.seq=c(.99,.95,.9),c0lim=FALSE)

#Binary logistic regression
fastboost(xran,ybin,func=lasso_cv_glmnet_bin_min)

```

---

```
selectboost_cascade  Selectboost_cascade
```

---

**Description**

Selectboost for Cascade inference.

**Usage**

```

selectboost(M, ...)

## S4 method for signature 'micro_array'
selectboost(
  M,
  Fabhat,
  K = 5,
  eps = 10^-5,
  cv.subjects = TRUE,
  ncores = 4,
  use.parallel = FALSE,
  verbose = FALSE,
  group = group_func_2,
  c0value = 0.95
)

```

**Arguments**

M                    Microarray class from the Cascade package.  
 ...                  Additionnal arguments. Not used.



Fabhat	F matrix inferred using the inference function from the Cascade package.
K	Number of crossvalidation folds.
eps	Threshold for assinging a zero value to an inferred parameter. Defaults to $10^{-5}$ .
cv.subjects	Crossvalidation is made subjectwise using leave one out. Discards the K option.
ncores	Numerical value. Number of cores for parallel computing. Defaults to 4.
use.parallel	Boolean. To use parallel computing (doMC) download the extended package from Github. Set to FALSE.
verbose	Boolean. Defaults to FALSE.
group	Function. The grouping function. Defaults to group_func_2.
c0value	Numeric. c0 value to use for confidence computation. Defaults to TRUE

### Details

Extending results from the Cascade package: providing confidence indices for the reverse engineered links.

Reference for the Cascade modelling Vallat, L., Kemper, C. a., Jung, N., Maumy-Bertrand, M., Bertrand, F., Meyer, N., Pocheville, A., Fisher, J. W., Gribben, J. G. et Bahram, S. (2013). Reverse-engineering the genetic circuitry of a cancer cell with predicted intervention in chronic lymphocytic leukemia. *Proceedings of the National Academy of Sciences of the United States of America*, 110(2), 459-64.

Reference for the Cascade package Jung, N., Bertrand, F., Bahram, S., Vallat, L. et Maumy-Bertrand, M. (2014). Cascade : A R package to study, predict and simulate the diffusion of a signal through a temporal gene network. *Bioinformatics*. ISSN 13674803..

### Value

A network.confidence object.

### Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

### References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, *Bioinformatics*, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

### See Also

[boost](#), [fastboost](#), [plot.selectboost](#), [inference](#)

Other Selectboost functions: [autoboost\(\)](#), [boost](#), [fastboost\(\)](#), [plot\\_selectboost\\_cascade](#)

**Examples**

```

set.seed(314)
set.seed(314)

data(Cascade_example)
Fab_inf_C <- Net_inf_C@F
#By default community grouping of variables
set.seed(1)
net_confidence <- selectboost(M, Fab_inf_C)
net_confidence_.5 <- selectboost(M, Fab_inf_C, c0value = .5)
#With group_func_1, variables are grouped by thresholding the correlation matrix
net_confidence_thr <- selectboost(M, Fab_inf_C, group = group_func_1)

```

---

simulation

*Miscellaneous simulation functions*


---

**Description**

Define several simulation functions to be used in the demos of the package.

**Usage**

```

simulation_cor(group, cor_group, v = 1)

simulation_X(N, Cor)

simulation_DATA(X, supp, minB, maxB, stn)

compsim(x, ...)

## S3 method for class 'simuls'
compsim(x, result.boost, level = 1, ...)

```

**Arguments**

group	A numeric vector. Group membership of each of the variables.
cor_group	A numeric vector. Intra-group Pearson correlation.
v	A numeric value. The diagonal value of the generated matrix.
N	A numeric value. The number of observations.
Cor	A numeric matrix. A correlation matrix to be used for random sampling.
X	A numeric matrix. Observations*variables.
supp	A numeric vector. The true predictors.
minB	A numeric value. Minimum absolute value for a beta coefficient.

maxB	A numeric value. Maximum absolute value for a beta coefficient.
stn	A numeric value. A scaling factor for the noise in the response. The higher, the smaller the noise.
x	List. Simulated dataset.
...	For compatibility issues.
result.boost	Row matrix of numerical value. Result of selecboost for a given c0.
level	List. Threshold for proportions of selected variables.

### Details

simulation\_cor returns a numeric symmetric matrix  $c$  whose order is the number of variables. An entry  $c_{i,j}$  is equal to

- $i = j$ , entries on the diagonal are equal to the  $v$  value
- $i <> j$ , 0 if the variable  $i$  and  $j$  do not belong to the same group
- $i <> j$ ,  $cor\_group[k]$  if the variable  $i$  and  $j$  belong to the group  $k$

simulation\_X returns a numeric matrix of replicates (by row) of random samples generated according to the Cor matrix.

simulation\_DATA returns a list with the X matrix, the response vector Y, the true predictors, the beta coefficients, the scaling factor and the standard deviation.

comsim.simuls computes recall (sensitivity), precision (positive predictive value), and several Fscores (non-weighted Fscore, F1/2 and F2 weighted Fscores).

### Value

simulation\_cor returns a numeric matrix.

simulation\_X returns a numeric matrix.

simulation\_DATA returns a list.

comsim.simuls returns a numerical vector.

### Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr> with contributions from Nicolas Jung.

### References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

### See Also

[glmnet](#), [cv.glmnet](#), [AICc\\_BIC\\_glmnetB](#), [lars](#), [cv.lars](#), [msgps](#)

**Examples**

```

N<-10
group<-c(rep(1:2,5))
cor_group<-c(.8,.4)
supp<-c(1,1,1,0,0,0,0,0,0,0)
minB<-1
maxB<-2
stn<-5
C<-simulation_cor(group,cor_group)

set.seed(314)
X<-simulation_X(10,C)
G<-abs(cor(X))
hist(G[lower.tri(G)])

set.seed(314)
DATA_exemple<-simulation_DATA(X,supp,1,2,stn)

set.seed(314)
result.boost = fastboost(DATA_exemple$X, DATA_exemple$Y, steps.seq = .7, c0lim = FALSE,
use.parallel = FALSE, B=10)
compsim(DATA_exemple, result.boost, level=.7)

```

---

```
summary.selectboost  Summarize a selectboost analysis
```

---

**Description**

Summarize a selectboost analysis.

**Usage**

```

## S3 method for class 'selectboost'
summary(
  object,
  crit.func = mean,
  crit.int = "mean",
  custom.values.lim = NULL,
  index.lim = NULL,
  alpha.conf.level = 0.99,
  force.dec = TRUE,
  ...
)

```

**Arguments**

object            Numerical matrix. Result of selectboost (autoboost, fastboost, ...).

crit.func	Function . Defaults to the mean function.
crit.int	Character value. Mean or median based confidence intervals. Defaults to "mean" based confidence intervals.
custom.values.lim	Vector of numeric values. Defaults to NULL.
index.lim	Vector of numeric values. Defaults to NULL.
alpha.conf.level	Numeric value. Defaults to 0.99.
force.dec	Boolean. Force trajectories to be non-increasing.
...	Additional arguments. Passed to the crit.func function.

### Details

summary.selectboost returns a list with the results.

### Value

A list with the results.

### Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

### References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

### See Also

[fastboost](#), [autoboost](#)

Other Selectboost analyse functions: [force.non.inc\(\)](#), [plot.selectboost\(\)](#)

### Examples

```
data(autoboost.res.x)
summary(autoboost.res.x)
summary(autoboost.res.x, force.dec=FALSE)

data(autoboost.res.x.adapt)
summary(autoboost.res.x.adapt)

data(autoboost.res.x2)
summary(autoboost.res.x2)
summary(autoboost.res.x2, force.dec=FALSE)

data(autoboost.res.x2.adapt)
```

```

summary(autoboost.res.x2.adapt)

data(fastboost.res.x)
summary(fastboost.res.x)
summary(fastboost.res.x, force.dec=FALSE)

data(fastboost.res.x.adapt)
summary(fastboost.res.x.adapt)

data(fastboost.res.x2)
summary(fastboost.res.x2)
summary(fastboost.res.x2, force.dec=FALSE)

data(fastboost.res.x2.adapt)
summary(fastboost.res.x2.adapt)

```

---

trajC0

*Plot trajectories*


---

### Description

Plot trajectories.

### Usage

```

trajC0(x, ...)

## S3 method for class 'selectboost'
trajC0(
  x,
  summary.selectboost.res,
  lasso.coef.path,
  type.x.axis = "noscale",
  type.graph = "boost",
  threshold.level = NULL,
  ...
)

```

### Arguments

x	Numerical matrix. Selectboost object.
...	. Passed to the plotting functions.
summary.selectboost.res	List. Summary of selectboost object.
lasso.coef.path	List. Result of predict.lars.
type.x.axis	Character value. "scale" or "noscale" for the X axis.

type.graph      Character value. Type of graphs: "bars", "lasso" and "boost".  
 threshold.level      Numeric value. Threshold for the graphs.

### Details

trajC0 returns an invisible list and creates four graphics.

### Value

An invisible list.  
 invisible list.

### Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

### References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

### See Also

[fastboost](#), [autoboost](#) and [summary.selectboost](#)

Other Selectboost analyze functions: [auto.analyze\(\)](#), [plot.summary.selectboost\(\)](#)

### Examples

```
data(autoboost.res.x)
data(diabetes, package="lars")

### With lasso trajectories
m.x<-lars::lars(diabetes$x,diabetes$y)
plot(m.x)
mm.x<-predict(m.x,type="coef",mode="lambda")
autoboost.res.x.mean = summary(autoboost.res.x)

par(mfrow=c(2,2),mar=c(4,4,1,1))
trajC0(autoboost.res.x,autoboost.res.x.mean,lasso.coef.path=mm.x,type.graph="lasso")
trajC0(autoboost.res.x,autoboost.res.x.mean)
trajC0(autoboost.res.x,autoboost.res.x.mean,type.graph="bars")
trajC0(autoboost.res.x,autoboost.res.x.mean,type.x.axis ="scale")
```

---

`var_select`*Variable selection functions*

---

**Description**

Compute coefficient vector after variable selection.

**Usage**

```
lasso_cv_glmnet_bin_min(X, Y)
lasso_cv_glmnet_bin_1se(X, Y)
lasso_glmnet_bin_AICc(X, Y)
lasso_glmnet_bin_BIC(X, Y)
lasso_cv_lars_min(X, Y)
lasso_cv_lars_1se(X, Y)
lasso_cv_glmnet_min(X, Y)
lasso_cv_glmnet_min_weighted(X, Y, priors)
lasso_cv_glmnet_1se(X, Y)
lasso_cv_glmnet_1se_weighted(X, Y, priors)
lasso_msgps_Cp(X, Y, penalty = "enet")
lasso_msgps_AICc(X, Y, penalty = "enet")
lasso_msgps_GCV(X, Y, penalty = "enet")
lasso_msgps_BIC(X, Y, penalty = "enet")
enetf_msgps_Cp(X, Y, penalty = "enet", alpha = 0.5)
enetf_msgps_AICc(X, Y, penalty = "enet", alpha = 0.5)
enetf_msgps_GCV(X, Y, penalty = "enet", alpha = 0.5)
enetf_msgps_BIC(X, Y, penalty = "enet", alpha = 0.5)
lasso_cascade(M, Y, K, eps = 10^-5, cv.fun)
```



**Arguments**

X	A numeric matrix. The predictors matrix.
Y	A binary factor. The 0/1 classification response.
priors	A numeric vector. Weighting vector for the variable selection. When used with the glmnet estimation function, the weights share the following meanings: <ul style="list-style-type: none"> <li>• 0: the variable is always included in the model</li> <li>• 1: neutral weight</li> <li>• Inf: variable is always excluded from the model</li> </ul>
penalty	A character value to select the penalty term in msgps (Model Selection Criteria via Generalized Path Seeking). Defaults to "enet". "genet" is the generalized elastic net and "lasso" is the adaptive lasso, which is a weighted version of the lasso.
alpha	A numeric value to set the value of $\alpha$ on "enet" and "genet" penalty in msgps (Model Selection Criteria via Generalized Path Seeking).
M	A numeric matrix. The transposed predictors matrix.
K	A numeric value. Number of folds to use.
eps	A numeric value. Threshold to set to 0 the inferred value of a parameter.
cv.fun	A function. Fonction used to create folds. Used to perform corss-validation subckectwise.

**Details**

lasso\_cv\_glmnet\_bin\_min returns the vector of coefficients for a binary logistic model estimated by the lasso using the lambda.min value computed by 10 fold cross validation. It uses the glmnet function of the glmnetpackage.

lasso\_cv\_glmnet\_bin\_1se returns the vector of coefficients for a binary logistic model estimated by the lasso using the lambda.1se (lambda.min+1se) value computed by 10 fold cross validation. It uses the glmnet function of the glmnetpackage.

lasso\_glmnet\_bin\_AICc returns the vector of coefficients for a binary logistic model estimated by the lasso and selected according to the bias-corrected AIC (AICC) criterion. It uses the glmnet

lasso\_glmnet\_bin\_BIC returns the vector of coefficients for a binary logistic model estimated by the lasso and selected according to the BIC criterion. It uses the glmnet

lasso\_cv\_lars\_min returns the vector of coefficients for a linear model estimated by the lasso using the lambda.min value computed by 5 fold cross validation. It uses the lars function of the lars package.

lasso\_cv\_lars\_1se returns the vector of coefficients for a linear model estimated by the lasso using the lambda.1se (lambda.min+1se) value computed by 5 fold cross validation. It uses the lars function of the lars package.

lasso\_cv\_glmnet\_min returns the vector of coefficients for a linear model estimated by the lasso using the lambda.min value computed by 10 fold cross validation. It uses the glmnet function of the glmnet package.

lasso\_cv\_glmnet\_min\_weighted returns the vector of coefficients for a linear model estimated by the weighted lasso using the lambda.min value computed by 10 fold cross validation. It uses the glmnet function of the glmnet package.

`lasso_cv_glmnet_1se` returns the vector of coefficients for a linear model estimated by the lasso using the `lambda.1se` (`lambda.min+1se`) value computed by 10 fold cross validation. It uses the `glmnet` function of the `glmnet` package.

`lasso_cv_glmnet_1se_weighted` returns the vector of coefficients for a linear model estimated by the weighted lasso using the `lambda.1se` (`lambda.min+1se`) value computed by 10 fold cross validation. It uses the `glmnet` function of the `glmnet` package.

`lasso_msgps_Cp` returns the vector of coefficients for a linear model estimated by the lasso selected using Mallows' `Cp`. It uses the `msgps` function of the `msgps` package.

`lasso_msgps_AICc` returns the vector of coefficients for a linear model estimated by the lasso selected according to the bias-corrected AIC (AICC) criterion. It uses the `msgps` function of the `msgps` package.

`lasso_msgps_GCV` returns the vector of coefficients for a linear model estimated by the lasso selected according to the generalized cross validation criterion. It uses the `msgps` function of the `msgps` package.

`lasso_msgps_BIC` returns the vector of coefficients for a linear model estimated by the lasso selected according to the BIC criterion. It uses the `msgps` function of the `msgps` package.

`enetf_msgps_Cp` returns the vector of coefficients for a linear model estimated by the elastic net selected using Mallows' `Cp`. It uses the `msgps` function of the `msgps` package.

`enetf_msgps_AICc` returns the vector of coefficients for a linear model estimated by the elastic net selected according to the bias-corrected AIC (AICC) criterion. It uses the `msgps` function of the `msgps` package.

`enetf_msgps_GCV` returns the vector of coefficients for a linear model estimated by the elastic net selected according to the generalized cross validation criterion. It uses the `msgps` function of the `msgps` package.

`enetf_msgps_BIC` returns the vector of coefficients for a linear model estimated by the elastic net selected according to the BIC criterion. It uses the `msgps` function of the `msgps` package.

`lasso_cascade` returns the vector of coefficients for a linear model estimated by the lasso. It uses the `lars` function of the `lars` package.

## Value

A vector of coefficients.

## Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

## References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. [doi:10.1093/bioinformatics/btaa855](https://doi.org/10.1093/bioinformatics/btaa855)

**See Also**

[glmnet](#), [cv.glmnet](#), [AICc\\_BIC\\_glmnetB](#), [lars](#), [cv.lars](#), [msgps](#)

Other Variable selection functions: [var\\_select\\_all](#)

**Examples**

```
set.seed(314)
xran=matrix(rnorm(150),30,5)
ybin=sample(0:1,30,replace=TRUE)
yran=rnorm(30)
set.seed(314)
lasso_cv_glmnet_bin_min(xran,ybin)
```

```
set.seed(314)
lasso_cv_glmnet_bin_1se(xran,ybin)
```

```
set.seed(314)
lasso_glmnet_bin_AICc(xran,ybin)
```

```
set.seed(314)
lasso_glmnet_bin_BIC(xran,ybin)
```

```
set.seed(314)
lasso_cv_lars_min(xran,yran)
```

```
set.seed(314)
lasso_cv_lars_1se(xran,yran)
```

```
set.seed(314)
lasso_cv_glmnet_min(xran,yran)
```

```
set.seed(314)
lasso_cv_glmnet_min_weighted(xran,yran,c(1000,0,0,1,1))
```

```
set.seed(314)
lasso_cv_glmnet_1se(xran,yran)
```

```
set.seed(314)
lasso_cv_glmnet_1se_weighted(xran,yran,c(1000,0,0,1,1))
```

```
set.seed(314)
lasso_msgps_Cp(xran,yran)
```

```
set.seed(314)
lasso_msgps_AICc(xran,yran)
```

```
set.seed(314)
lasso_msgps_GCV(xran,yran)
```

```
set.seed(314)
lasso_msgps_BIC(xran,yran)
```

```

set.seed(314)
enetf_msggps_Cp(xran,yran)

set.seed(314)
enetf_msggps_AICc(xran,yran)

set.seed(314)
enetf_msggps_GCV(xran,yran)

set.seed(314)
enetf_msggps_BIC(xran,yran)

set.seed(314)
lasso_cascade(t(xran),yran,5,cv.fun=lars::cv.folds)

```

---

var_select_all	<i>Variable selection functions (all)</i>
----------------	---

---

### Description

Compute coefficient vector after variable selection for the fitting criteria of a given model. May be used for a step by step use of Selectboost.

### Usage

```

lasso_msggps_all(X, Y, penalty = "enet")

enet_msggps_all(X, Y, penalty = "enet", alpha = 0.5)

alasso_msggps_all(X, Y, penalty = "alasso")

alasso_enet_msggps_all(X, Y, penalty = "alasso", alpha = 0.5)

lasso_cv_glmnet_all_5f(X, Y)

spl_spls_all(X, Y, K.seq = c(1:5), eta.seq = (1:9)/10, fold.val = 5)

varbvs_linear_all(X, Y, include.threshold.list = (1:19)/20)

lasso_cv_glmnet_bin_all(X, Y)

lasso_glmnet_bin_all(X, Y)

spl_spls_all(X, Y, K.seq = c(1:10), eta.seq = (1:9)/10)

sgpls_spls_all(X, Y, K.seq = c(1:10), eta.seq = (1:9)/10)

varbvs_binomial_all(X, Y, include.threshold.list = (1:19)/20)

```

**Arguments**

X	A numeric matrix. The predictors matrix.
Y	A binary factor. The 0/1 classification response.
penalty	A character value to select the penalty term in msgps (Model Selection Criteria via Generalized Path Seeking). Defaults to "enet". "genet" is the generalized elastic net and "alasso" is the adaptive lasso, which is a weighted version of the lasso.
alpha	A numeric value to set the value of $\alpha$ on "enet" and "genet" penalty in msgps (Model Selection Criteria via Generalized Path Seeking).
K.seq	A numeric vector. Number of components to test.
eta.seq	A numeric vector. Eta sequence to test.
fold.val	A numeric value. Number of folds to use.
include.threshold.list	A numeric vector. Vector of threshold to use.
K	A numeric value. Number of folds to use.

**Details**

lasso\_msgps\_all returns the matrix of coefficients for an optimal linear model estimated by the LASSO estimator and selected by model selection criteria including Mallows' Cp, bias-corrected AIC (AICc), generalized cross validation (GCV) and BIC. The the msgps function of the msgps package implements Model Selection Criteria via Generalized Path Seeking to compute the degrees of freedom of the LASSO.

enet\_msgps\_all returns the matrix of coefficients for an optimal linear model estimated by the ELASTIC NET estimator and selected by model selection criteria including Mallows' Cp, bias-corrected AIC (AICc), generalized cross validation (GCV) and BIC. The the msgps function of the msgps package implements Model Selection Criteria via Generalized Path Seeking to compute the degrees of freedom of the ELASTIC NET.

alasso\_msgps\_all returns the matrix of coefficients for an optimal linear model estimated by the adaptive LASSO estimator and selected by model selection criteria including Mallows' Cp, bias-corrected AIC (AICc), generalized cross validation (GCV) and BIC. The the msgps function of the msgps package implements Model Selection Criteria via Generalized Path Seeking to compute the degrees of freedom of the adaptive LASSO.

alasso\_enet\_msgps\_all returns the matrix of coefficients for an optimal linear model estimated by the adaptive ELASTIC NET estimator and selected by model selection criteria including Mallows' Cp, bias-corrected AIC (AICc), generalized cross validation (GCV) and BIC. The the msgps function of the msgps package implements Model Selection Criteria via Generalized Path Seeking to compute the degrees of freedom of the adaptive ELASTIC NET.

lasso\_cv\_glmnet\_all\_5f returns the matrix of coefficients for a linear model estimated by the LASSO using the lambda.min and lambda.1se (lambda.min+1se) values computed by 5 fold cross validation. It uses the glmnet and cv.glmnet functions of the glmnet package.

spls\_spls\_all returns the matrix of the raw (coef.spls) and correct.spls and bootstrap corrected coefficients for a linear model estimated by the SPLS (sparse partial least squares) and 5 fold cross validation. It uses the spls, cv.spls, ci.spls, coef.spls and correct.spls functions of the spls package.

varbvs\_linear\_all returns the matrix of the coefficients for a linear model estimated by the varbvs (variational approximation for Bayesian variable selection in linear regression, family = gaussian) and the requested threshold values. It uses the varbvs, coef and variable.names functions of the varbvs package.

lasso\_cv\_glmnet\_bin\_all returns the matrix of coefficients for a logistic model estimated by the LASSO using the lambda.min and lambda.1se (lambda.min+1se) values computed by 5 fold cross validation. It uses the glmnet and cv.glmnet functions of the glmnet package.

lasso\_glmnet\_bin\_all returns the matrix of coefficients for a logistic model estimated by the LASSO using the AICc\_glmnetB and BIC\_glmnetB information criteria. It uses the glmnet function of the glmnet package and the AICc\_glmnetB and BIC\_glmnetB functions of the SelectBoost package that were adapted from the AICc\_glmnetB and BIC\_glmnetB functions of the rLogistic (<https://github.com/echi/rLogistic>) package.

splsda\_spls\_all returns the matrix of the raw (coef.splsda) coefficients for logistic regression model estimated by the SGPLS (sparse généralized partial least squares) and 5 fold cross validation. It uses the splsda, cv.splsda and coef.splsda functions of the sgpls package.

sgpls\_spls\_all returns the matrix of the raw (coef.sgpls) coefficients for logistic regression model estimated by the SGPLS (sparse généralized partial least squares) and 5 fold cross validation. It uses the sgpls, cv.sgpls and coef.sgpls functions of the sgpls package.

varbvs\_binomial\_all returns the matrix of the coefficients for a linear model estimated by the varbvs (variational approximation for Bayesian variable selection in logistic regression, family = binomial) and the requested threshold values. It uses the varbvs, coef and variable.names functions of the varbvs package.

## Value

A vector or matrix of coefficients.

## Author(s)

Frederic Bertrand, <frederic.bertrand@utt.fr>

## References

*selectBoost: a general algorithm to enhance the performance of variable selection methods in correlated datasets*, Frédéric Bertrand, Ismail Aouadi, Nicolas Jung, Raphael Carapito, Laurent Vallat, Seiamak Bahram, Myriam Maumy-Bertrand, Bioinformatics, 2020. doi:10.1093/bioinformatics/btaa855

## See Also

[glmnet](#), [cv.glmnet](#), [msgpls](#), [AICc\\_BIC\\_glmnetB](#), [spls](#), [cv.spls](#), [correct.spls](#), [splsda](#), [cv.splsda](#), [sgpls](#), [cv.sgpls](#), [varbvs](#)

Other Variable selection functions: [var\\_select](#)

**Examples**

```
set.seed(314)
xran <- matrix(rnorm(100*6),100,6)
beta0 <- c(3,1.5,0,0,2,0)
epsilon <- rnorm(100,sd=3)
yran <- c(xran %*% beta0 + epsilon)
ybin <- ifelse(yran>=0,1,0)
set.seed(314)
lasso_msgps_all(xran,yran)

set.seed(314)
enet_msgps_all(xran,yran)

set.seed(314)
alasso_msgps_all(xran,yran)

set.seed(314)
alasso_enet_msgps_all(xran,yran)

set.seed(314)
lasso_cv_glmnet_all_5f(xran,yran)

set.seed(314)
spl_spls_all(xran,yran)

set.seed(314)
varbvs_linear_all(xran,yran)

set.seed(314)
lasso_cv_glmnet_bin_all(xran,ybin)

set.seed(314)
lasso_glmnet_bin_all(xran,ybin)

set.seed(314)

spl_sda_spls_all(xran,ybin, K.seq=1:3)

set.seed(314)

sgpls_spls_all(xran,ybin, K.seq=1:3)

set.seed(314)
varbvs_binomial_all(xran,ybin)
```

# Index

- \* **Plot functions**
  - `miscplot`, 20
- \* **Selectboost analyse functions**
  - `force.non.inc`, 17
  - `plot.selectboost`, 22
  - `summary.selectboost`, 36
- \* **Selectboost analyze functions**
  - `auto.analyze`, 4
  - `plot.summary.selectboost`, 23
  - `trajC0`, 38
- \* **Selectboost functions**
  - `autoboost`, 5
  - `boost`, 9
  - `fastboost`, 14
  - `plot_selectboost_cascade`, 24
  - `selectboost_cascade`, 32
- \* **Simulation functions**
  - `simulation`, 34
- \* **Variable selection functions**
  - `var_select`, 40
  - `var_select_all`, 44
- \* **datasets**
  - `autoboost.res.x`, 8
  - `autoboost.res.x.adapt`, 8
  - `autoboost.res.x2`, 8
  - `autoboost.res.x2.adapt`, 9
  - `Cascade_confidence`, 12
  - `Cascade_example`, 13
  - `fastboost.res.x`, 16
  - `fastboost.res.x.adapt`, 16
  - `fastboost.res.x2`, 17
  - `fastboost.res.x2.adapt`, 17
  - `results_simuls_reverse_engineering_v3`, 26
- `AICc_BIC_glmnetB`, 2, 35, 43, 46
- `AICc_glmnetB (AICc_BIC_glmnetB)`, 2
- `alasso_enet_msgps_all (var_select_all)`, 44
- `alasso_msgps_all (var_select_all)`, 44
- `auto.analyze`, 4, 24, 39
- `autoboost`, 5, 5, 12, 15, 18, 23–25, 33, 37, 39
- `autoboost.res.x`, 8
- `autoboost.res.x.adapt`, 8
- `autoboost.res.x2`, 8
- `autoboost.res.x2.adapt`, 9
- `BIC_glmnetB (AICc_BIC_glmnetB)`, 2
- `boost`, 7, 9, 15, 25, 33
- `boost.findgroups`, 19, 20
- `Cascade_confidence`, 12
- `Cascade_example`, 13
- `compsim (simulation)`, 34
- `correct.spls`, 46
- `cv.glmnet`, 35, 43, 46
- `cv.lars`, 35, 43
- `cv.sgpls`, 46
- `cv.spls`, 46
- `cv.splsda`, 46
- `enet_msgps_all (var_select_all)`, 44
- `enetf_msgps_AICc (var_select)`, 40
- `enetf_msgps_BIC (var_select)`, 40
- `enetf_msgps_Cp (var_select)`, 40
- `enetf_msgps_GCV (var_select)`, 40
- `F_score_C (results_simuls_reverse_engineering_v3)`, 26
- `F_score_PB (results_simuls_reverse_engineering_v3)`, 26
- `F_score_PB_075_075 (results_simuls_reverse_engineering_v3)`, 26
- `F_score_PB_095_075 (results_simuls_reverse_engineering_v3)`, 26



- F\_score\_PB\_W  
     (results\_simuls\_reverse\_engineering\_v3), 26  
     lasso\_cv\_glmnet\_bin\_min (var\_select), 40  
     lasso\_cv\_glmnet\_min (var\_select), 40  
     lasso\_cv\_glmnet\_min\_weighted (var\_select), 40
- F\_score\_PL  
     (results\_simuls\_reverse\_engineering\_v3), 26  
     lasso\_cv\_lars\_1se (var\_select), 40  
     lasso\_cv\_lars\_min (var\_select), 40
- F\_score\_PL2  
     (results\_simuls\_reverse\_engineering\_v3), 26  
     lasso\_glmnet\_bin\_AICc (var\_select), 40  
     lasso\_glmnet\_bin\_all (var\_select\_all), 44
- F\_score\_PL2\_tW  
     (results\_simuls\_reverse\_engineering\_v3), 26  
     lasso\_glmnet\_bin\_BIC (var\_select), 40  
     lasso\_msgps\_AICc (var\_select), 40  
     lasso\_msgps\_all (var\_select\_all), 44
- F\_score\_PL2\_W  
     (results\_simuls\_reverse\_engineering\_v3), 26  
     lasso\_msgps\_BIC (var\_select), 40  
     lasso\_msgps\_Cp (var\_select), 40  
     lasso\_msgps\_GCV (var\_select), 40
- F\_score\_PSel  
     (results\_simuls\_reverse\_engineering\_v3), 26  
     M (Cascade\_example), 13  
     miscplot, 20
- F\_score\_PSel\_W  
     (results\_simuls\_reverse\_engineering\_v3), 26  
     msgps, 35, 43, 46  
     Net (Cascade\_example), 13
- F\_score\_robust  
     (results\_simuls\_reverse\_engineering\_v3), 26  
     net\_confidence (Cascade\_confidence), 12  
     net\_confidence\_.5 (Cascade\_confidence), 12  
     net\_confidence\_thr (Cascade\_confidence), 12  
     Net\_inf\_C (Cascade\_example), 13  
     network.confidence-class, 21  
     nv\_C (results\_simuls\_reverse\_engineering\_v3), 26  
     nv\_PB (results\_simuls\_reverse\_engineering\_v3), 26  
     nv\_PB\_075\_075 (results\_simuls\_reverse\_engineering\_v3), 26  
     nv\_PB\_095\_075 (results\_simuls\_reverse\_engineering\_v3), 26  
     nv\_PB\_W (results\_simuls\_reverse\_engineering\_v3), 26  
     nv\_PL (results\_simuls\_reverse\_engineering\_v3), 26  
     nv\_PL2 (results\_simuls\_reverse\_engineering\_v3), 26
- fastboost, 5, 7, 12, 14, 18, 23–25, 33, 37, 39  
 fastboost.res.x, 16  
 fastboost.res.x.adapt, 16  
 fastboost.res.x2, 17  
 fastboost.res.x2.adapt, 17  
 force.non.inc, 17, 23, 37
- glmnet, 35, 43, 46  
 group\_func\_1, 18  
 group\_func\_2, 19, 19, 20
- igraph, 20  
 inference, 25, 33  
 infomap.community, 20
- lars, 35, 43  
 lasso\_cascade (var\_select), 40  
 lasso\_cv\_glmnet\_1se (var\_select), 40  
 lasso\_cv\_glmnet\_1se\_weighted (var\_select), 40  
 lasso\_cv\_glmnet\_all\_5f (var\_select\_all), 44  
 lasso\_cv\_glmnet\_bin\_1se (var\_select), 40  
 lasso\_cv\_glmnet\_bin\_all (var\_select\_all), 44

nv\_PL2\_tW predictive\_positive\_value\_PL2\_W  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

nv\_PL2\_W predictive\_positive\_value\_PSel  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

nv\_PSel predictive\_positive\_value\_PSel\_W  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

nv\_PSel\_W predictive\_positive\_value\_robust  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

nv\_robust  
 (results\_simuls\_reverse\_engineering\_v3), rrr (AICc\_BIC\_glmnetB), 2  
 26 results\_simuls\_reverse\_engineering\_v3,  
 26

plot,network.confidence,ANY-method ridge\_logistic (AICc\_BIC\_glmnetB), 2  
 (plot\_selectboost\_cascade), 24

plot,network.confidence,network.confidence-method SelectBoost, 31  
 (plot\_selectboost\_cascade), 24 selectboost, 25

plot.matrix (miscplot), 20 selectboost (selectboost\_cascade), 32

plot.selectboost, 7, 15, 18, 22, 33, 37 selectboost,micro\_array,micro\_array-method  
 (selectboost\_cascade), 32

plot.summary.selectboost, 5, 23, 39 selectboost,micro\_array-method  
 (selectboost\_cascade), 32

plot\_selectboost\_cascade, 7, 12, 15, 24, 33 selectboost\_cascade, 7, 12, 15, 25, 32

predictive\_positive\_value\_C sensitivity\_C  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

predictive\_positive\_value\_PB sensitivity\_PB  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

predictive\_positive\_value\_PB\_075\_075 sensitivity\_PB\_075\_075  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

predictive\_positive\_value\_PB\_095\_075 sensitivity\_PB\_095\_075  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

predictive\_positive\_value\_PB\_W sensitivity\_PB\_W  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

predictive\_positive\_value\_PL sensitivity\_PL  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

predictive\_positive\_value\_PL2 sensitivity\_PL2  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

predictive\_positive\_value\_PL2\_tW sensitivity\_PL2\_tW  
 (results\_simuls\_reverse\_engineering\_v3), (results\_simuls\_reverse\_engineering\_v3),  
 26 26

[26](#)  
 sensitivity\_PL2\_W  
     (results\_simuls\_reverse\_engineering\_v3), [26](#)  
[26](#)  
 sensitivity\_PSel  
     (results\_simuls\_reverse\_engineering\_v3), [26](#)  
[26](#)  
 sensitivity\_PSel\_W  
     (results\_simuls\_reverse\_engineering\_v3), [26](#)  
[26](#)  
 sensitivity\_robust  
     (results\_simuls\_reverse\_engineering\_v3), [26](#)  
[26](#)  
 trajC0, [5](#), [24](#), [38](#)  
 sgpls, [46](#)  
 sgpls\_spls\_all (var\_select\_all), [44](#)  
 simulation, [34](#)  
 simulation\_cor (simulation), [34](#)  
 simulation\_DATA (simulation), [34](#)  
 simulation\_X (simulation), [34](#)  
 spls, [46](#)  
 spls\_spls\_all (var\_select\_all), [44](#)  
 splsda, [46](#)  
 splsda\_spls\_all (var\_select\_all), [44](#)  
 summary.selectboost, [18](#), [23](#), [24](#), [36](#), [39](#)  
  
 test.seq\_C  
     (results\_simuls\_reverse\_engineering\_v3),  
     [26](#)  
 test.seq\_PB  
     (results\_simuls\_reverse\_engineering\_v3),  
     [26](#)  
 test.seq\_PB\_075\_075  
     (results\_simuls\_reverse\_engineering\_v3),  
     [26](#)  
 test.seq\_PB\_095\_075  
     (results\_simuls\_reverse\_engineering\_v3),  
     [26](#)  
 test.seq\_PB\_W  
     (results\_simuls\_reverse\_engineering\_v3),  
     [26](#)  
 test.seq\_PL  
     (results\_simuls\_reverse\_engineering\_v3),  
     [26](#)  
 test.seq\_PL2  
     (results\_simuls\_reverse\_engineering\_v3),  
     [26](#)  
 test.seq\_PL2\_tW  
     (results\_simuls\_reverse\_engineering\_v3),  
     [26](#)  
     test.seq\_PL2\_W  
         (results\_simuls\_reverse\_engineering\_v3),  
         [26](#)  
     test.seq\_PSel  
         (results\_simuls\_reverse\_engineering\_v3),  
         [26](#)  
     test.seq\_PSel\_W  
         (results\_simuls\_reverse\_engineering\_v3),  
         [26](#)  
     test.seq\_robust  
         (results\_simuls\_reverse\_engineering\_v3),  
         [26](#)  
     var\_select, [4](#), [40](#), [46](#)  
     var\_select\_all, [43](#), [44](#)  
     varbvs, [46](#)  
     varbvs\_binomial\_all (var\_select\_all), [44](#)  
     varbvs\_linear\_all (var\_select\_all), [44](#)