Package 'BTLLasso'

September 29, 2015

Type Package

Title Modelling Heterogeneity in Paired Comparison Data

Version 0.1-2

Date 2015-09-07

Author Gunther Schauberger

Maintainer Gunther Schauberger <gunther@stat.uni-muenchen.de>

Description Performs 'BTLLasso', a method to model heterogeneity in paired comparison data. Subject-specific covariates are allowed to have an influence on the attractivity/strength of the objects. An L1 penalty on the pairwise differences between the object-specific parameters allows for both clustering of objects with regard to covariates and elimination of irrelevant covariates.

License GPL (>= 2)

Imports Rcpp (>= 0.11.3), stringr

Depends Matrix, parallel

LinkingTo Rcpp, RcppArmadillo

SystemRequirements C++11

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-09-29 01:30:29

R topics documented:

BTLLasso-package *BTLLasso*

Description

Performs BTLLasso, a method to model heterogeneity in paired comparison data. Subject-specific covariates are allowd to have an influence on the attractivity/strength of the objects. An L1 penalty on the pairwise differences between the object-specific parameters allows for both clustering of object with regard to covariates and elimination of irrelevant covariates. Several additional functions are provided, such as cross-validation, bootstraped confidence intervals, and several plot functions.

Author(s)

Gunther Schauberger
<gunther@stat.uni-muenchen.de>
http://www.statistik.lmu.de/~schauberger/

References

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

See Also

BTLLasso, cv.BTLLasso

Examples

```
## Not run:
# load data set
data(GLESsmall)

# define response and covariate matrix
X <- scale(GLESsmall[, 11:14])
Y <- as.matrix(GLESsmall[, 1:10])

# vector of subtitles, containing the coding of the single covariates
subs <- c("(in years)","female (1); male (0)",
"East Germany (1); West Germany (0)","(very) good (1); else (0)")

# vector of tuning parameters
lambda <- exp(seq(log(31),log(1),length=50))-1

# compute BTLLasso model
m <- BTLLasso(Y = Y, X = X, lambda = lambda)

op <- par(no.readonly = TRUE)
```

```
par(mar=c(5,4,4,8))

# plot covariate paths
paths(m)

# plot parameter paths
singlepaths(m, subs = subs)

# compute 10-fold cross-validation
set.seed(5)
m.cv <- cv.BTLLasso(Y = Y, X = X, folds = 10, lambda = lambda, cores = 10)

# plot covariate paths, together with cv-optimal model
paths(m.cv)

# plot parameter paths, together with cv-optimal model
singlepaths(m.cv, subs = subs)

# compute bootstrap confidence intervals
m.boot <- boot.BTLLasso(m.cv, B = 100, cores = 25)

# plot bootstrap confidence intervals
par(mar=c(5,5,4,3))
ci.BTLLasso(m.boot, subs = subs)

par(op)

## End(Not run)
```

`boot.BTLLasso` *Bootstrap function for BTLLasso*

Description

Performs bootstrap for BTLLasso to get bootstrap confidence intervals. Main input argument is a `cv.BTLLasso` object. The bootstrap is performed on level of the cross-validation. Therefore, within every bootstrap iteration the complete cross-validation procedure from the `cv.BTLLasso` object is performed. The resulting `boot.BTLLasso` object is put into `ci.BTLLasso` to plot bootstrap confidence intervals.

Usage

```
boot.BTLLasso(model, B = 500, lambda = NULL, cores = 1, trace = TRUE, trace.cv = TRUE,
              control = BTLLasso.ctrl())
```

Arguments

`model`	A `cv.BTLLasso` object.
`B`	Number of bootstrap iterations.

lambda	Vector of tuning parameters. If not specified (default), tuning parameters from cv.BTLLasso object are used. See also details.
cores	Number of cores for (parallelized) computation.
trace	Should the trace of the BTLLasso algorithm be printed?
trace.cv	Should the trace fo the cross-validation be printed? If parallelized, the trace is not working on Windows machines.
control	Function for control arguments, mostly for internal use. See also BTLLasso.ctrl.

Details

The method can be highly time-consuming, for high numbers of tuning parameters, high numbers of folds in the crossvalidation and high number of bootstrap iterations B. The number of tuning parameters can be reduced by specifying lambda in the boot.BTLLasso function. You can control if the range of prespecified tuning parameters was to small by looking at the output values lambda.max.alert and lambda.min.alert. They are set TRUE if the smallest or largest of the specifed lambda values was chosen in at least one bootstrap iteration.

Value

cv.model	cv.BTLLasso object
estimatesB	Matrix containing all B estimates for original parameters. For internal use.
estimatesBrepar	Matrix containing all B estimates for reparameterized (symmetric side constraints) parameters.
lambdaB	vector of used tuning parameters
conf.ints	Bootstrap confidence intervals for original parameters. For internal use.
conf.ints.repar	Bootstrap confidence intervals for reparameterized (symmetric side constraint) parameters.
lambda.max.alert	Was the largest value of lambda chosen in at least one bootstrap iteration?
lambda.min.alert	Was the smallest value of lambda chosen in at least one bootstrap iteration?

Author(s)

Gunther Schauberger
<gunther@stat.uni-muenchen.de>
http://www.statistik.lmu.de/~schauberger/

References

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

See Also

BTLLasso, cv.BTLLasso, ci.BTLLasso

Examples

```
## Not run:
# load data set
data(GLESsmall)

# define response and covariate matrix
X <- scale(GLESsmall[, 11:14])
Y <- as.matrix(GLESsmall[, 1:10])

# vector of subtitles, containing the coding of the single covariates
subs <- c("(in years)","female (1); male (0)",
"East Germany (1); West Germany (0)","(very) good (1); else (0)")

# vector of tuning parameters
lambda <- exp(seq(log(31),log(1),length=50))-1

# compute 10-fold cross-validation
set.seed(5)
m.cv <- cv.BTLLasso(Y = Y, X = X, folds = 10, lambda = lambda, cores = 10)

# compute bootstrap confidence intervals
m.boot <- boot.BTLLasso(m.cv, B = 100, cores = 25)

# plot bootstrap confidence intervals
op <- par(no.readonly = TRUE)
par(mar=c(5,5,4,3))
ci.BTLLasso(m.boot, subs = subs)

par(op)

## End(Not run)
```

BTLLasso *Function to perform BTLLasso*

Description

Performs BTLLasso, a method to model heterogeneity in paired comparison data. Subject-specific covariates are allowd to have an influence on the attractivity/strength of the objects. An L1 penalty on the pairwise differences between the object-specific parameters allows for both clustering of object with regard to covariates and elimination of irrelevant covariates.

Usage

```
BTLLasso(Y, X, lambda, control = BTLLasso.ctrl(), trace = TRUE)
```

Arguments

Y	Matrix containing paired comparisons, ordered or binary. For K categories, the colmuns have to be ordered as follows: 1 vs 2, 1 vs 3, ..., 1 vs K, 2 vs 3, 2 vs 4, ..., 2 vs K, ..., K-1 vs K. One row represents one subject.
X	Matrix containing all subject-specific covariates. One row represents one subject, one column represents one covariate. Has to be standardized.
lambda	Vector of tuning parameters.
control	Function for control arguments, mostly for internal use. See also BTLLasso.ctrl.
trace	Should the trace of the BTLLasso algorithm be printed?

Details

Y matrix needs a special format. Column names need to be in the style "1 vs 2","1 vs 3",... If so, object labels "1", "2", ... are extracted from the column names.

Value

coefs	Matrix containing all (original) coefficients, one row per tuning parameter, one column per coefficient.
coefs.repar	Matrix containing all reparameterized (for symmetric side constraint) coefficients, one row per tuning parameter, one column per coefficient.
logLik	Vector of log-likelihoods, one per tuning parameter.
design	Design matrix, NULL if return.design=FALSE in BTLLasso.ctrl
Y	Y matrix
q	q=K-1, K is the number of different categories in Y
acoefs	Matrix containing penalties, for internal use.
response	Vector containing 0-1 coded response.
n	Number of persons/subjects
I	Number of paired comparisons
m	Number of objects
p	Number of covariates
X	X matrix
n.theta	Number of estimated threshold parameters
lambda	Vector of tuning parameters
labels	Labels of objects, only correct if Y specified correctly by "1 vs 2","1 vs 3",...
epsilon	Threshold value for convergence of the algorithm, specified in BTLLasso.ctrl

Author(s)

Gunther Schauberger
<gunther@stat.uni-muenchen.de>
http://www.statistik.lmu.de/~schauberger/

References

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

See Also

cv.BTLLasso, boot.BTLLasso, singlepaths, paths

Examples

```
## Not run:
# load data set
data(GLESsmall)

# define response and covariate matrix
X <- scale(GLESsmall[, 11:14])
Y <- as.matrix(GLESsmall[, 1:10])

# vector of subtitles, containing the coding of the single covariates
subs <- c("(in years)","female (1); male (0)",
"East Germany (1); West Germany (0)","(very) good (1); else (0)")

# vector of tuning parameters
lambda <- exp(seq(log(31),log(1),length=50))-1

# compute BTLLasso model
m <- BTLLasso(Y = Y, X = X, lambda = lambda)

op <- par(no.readonly = TRUE)
par(mar=c(5,4,4,8))

# plot covariate paths
paths(m)

# plot parameter paths
singlepaths(m, subs = subs)

par(op)

## End(Not run)
```

BTLLasso.ctrl *Control function for BTLLasso*

Description

Contains some additional parameters, mostly for internal use.

Usage

```
BTLLasso.ctrl(adaptive = TRUE, norm = c("L1", "L2"), epsilon = 1e-04, lambda2 = 1e-04,
              c = 1e-09, penal.diffs = TRUE,return.design = TRUE)
```

Arguments

adaptive	Should adaptive lasso be used? Default is TRUE.
norm	Specifies the norm used in the penalty term. Currently, only "L1" and "L2" are possible. Default is to "L1", only "L1" allows for clustering and variable selection.
epsilon	Threshold value for convergence of the algorithm.
lambda2	Tuning parameter for ridge penalty on all coefficients. Should be small, only used to stabilize results.
c	Internal parameter for the quadratic approximation of the L1 penalty. Should be sufficiently small. For details see cat_control.
penal.diffs	Should also differences between all parameters be penalized? Default is TRUE. If FALSE, only absolute values are penalized.
return.design	Should the design matrix be returned from the BTLLasso function. Necessary for internal use.

Author(s)

Gunther Schauberger
<gunther@stat.uni-muenchen.de>
http://www.statistik.lmu.de/~schauberger/

References

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

See Also

BTLLasso, cv.BTLLasso

Examples

```
## Not run:
# load data set
data(GLESsmall)

# define response and covariate matrix
X <- scale(GLESsmall[, 11:14])
Y <- as.matrix(GLESsmall[, 1:10])

# vector of subtitles, containing the coding of the single covariates
subs <- c("(in years)","female (1); male (0)",
```

```
"East Germany (1); West Germany (0)","(very) good (1); else (0)")

# vector of tuning parameters
lambda <- exp(seq(log(31),log(1),length=50))-1

# compute BTLLasso model, increase accuracy of the algorithm
m <- BTLLasso(Y = Y, X = X, lambda = lambda, control = BTLLasso.ctrl(epsilon = 1e-05))


## End(Not run)
```

`ci.BTLLasso` *Plot confidence intervals for BTLLasso*

Description

Plots confidence intervals for every single coefficient. Confidence intervals are separated by covariates, every covariate is plotted separately. Confidence intervals are based on bootstrap, performed by `boot.BTLLasso`.

Usage

```
ci.BTLLasso(object, subs = NULL, rows = NULL, subset = NULL)
```

Arguments

`object`	boot.BTLLasso object
`subs`	Optional vector of subtitles for the single plots. Can be used to note the encoding of the single covariates, especially for dummy variables.
`rows`	Optional argument for the number of rows,
`subset`	Optional argument if only a subset of the covariates should be plotted. Vector specifying the numbers of the covariates that should be plotted.

Author(s)

Gunther Schauberger
<gunther@stat.uni-muenchen.de>
http://www.statistik.lmu.de/~schauberger/

References

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

See Also

`boot.BTLLasso`, `BTLLasso`, `cv.BTLLasso`

Examples

```
## Not run:
# load data set
data(GLESsmall)

# define response and covariate matrix
X <- scale(GLESsmall[, 11:14])
Y <- as.matrix(GLESsmall[, 1:10])

# vector of subtitles, containing the coding of the single covariates
subs <- c("(in years)","female (1); male (0)",
"East Germany (1); West Germany (0)","(very) good (1); else (0)")

# vector of tuning parameters
lambda <- exp(seq(log(31),log(1),length=50))-1

# compute 10-fold cross-validation
set.seed(5)
m.cv <- cv.BTLLasso(Y = Y, X = X, folds = 10, lambda = lambda, cores = 10)

# plot covariate paths, together with cv-optimal model
paths(m.cv)

# plot parameter paths, together with cv-optimal model
singlepaths(m.cv, subs = subs)

# compute bootstrap confidence intervals
m.boot <- boot.BTLLasso(m.cv, B = 100, cores = 25)

# plot bootstrap confidence intervals
op <- par(no.readonly = TRUE)
par(mar=c(5,5,4,3))
ci.BTLLasso(m.boot, subs = subs)

par(op)

## End(Not run)
```

`cv.BTLLasso` *Cross-validation function for BTLLasso*

Description

Performs crossvalidation of BTLLasso, including the BTLLasso algorithm for the whole data set.

Usage

```
cv.BTLLasso(Y, X, folds = 10, lambda, control = BTLLasso.ctrl(),
            cores = folds, trace = TRUE, trace.cv = TRUE)
```

Arguments

Y	Matrix containing paired comparisons, ordered or binary. For K categories, the colmuns have to be ordered as follows: 1 vs. 2, 1 vs.3, ..., 1 vs. K, 2 vs. 3, 2 vs. 4, ..., 2 vs. K, ..., K-1 vs. K. One row represents one subject.
X	Matrix containing all subject-specific covariates. One row represents one subject, one column represents one covariate. Has to be standardized.
folds	Number of folds for the crossvalidation. Default is 10.
lambda	Vector of tuning parameters.
control	Function for control arguments, mostly for internal use. See also BTLLasso.ctrl.
cores	Number of cores used for (parallelized) cross-validation. By default, equal to the number of folds.
trace	Should the trace of the BTLLasso algorithm be printed?
trace.cv	Should the trace fo the cross-validation be printed? If parallelized, the trace is not working on Windows machines.

Details

Cross validation can be performed parallel, default is 10-fold cross validation on 10 cores. Output is a cv.BTLLasso object which can then be used for bootstrap confidence intervalls using boot.BTLLasso and ci.BTLLasso.

Value

coefs	Matrix containing all (original) coefficients, one row per tuning parameter, one column per coefficient.
coefs.repar	Matrix containing all reparameterized (for symmetric side constraint) coefficients, one row per tuning parameter, one column per coefficient.
logLik	Vector of log-likelihoods, one per tuning parameter.
design	Design matrix, NULL if return.design=FALSE in BTLLasso.ctrl
Y	Y matrix
q	q=K-1, K is the number of different categories in Y
acoefs	Matrix containing penalties, for internal use.
response	Vector containing 0-1 coded response.
n	Number of persons/subjects
I	Number of paired comparisons
m	Number of objects
p	Number of covariates
X	X matrix
n.theta	Number of estimated threshold parameters
lambda	Vector of tuning parameters
deviances	Vector of cross-validation deviances, one value per tuning parameter.
folds	Number of folds in cross validation.
labels	Labels of objects, only correct if Y specified correctly by "1 vs. 2","1 vs. 3",...
epsilon	Threshold value for convergence of the algorithm, specified in BTLLasso.ctrl

Author(s)

Gunther Schauberger
<gunther@stat.uni-muenchen.de>
http://www.statistik.lmu.de/~schauberger/

References

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

See Also

BTLLasso, boot.BTLLasso, singlepaths, paths

Examples

```
## Not run:
# load data set
data(GLESsmall)

# define response and covariate matrix
X <- scale(GLESsmall[, 11:14])
Y <- as.matrix(GLESsmall[, 1:10])

# vector of subtitles, containing the coding of the single covariates
subs <- c("(in years)","female (1); male (0)",
"East Germany (1); West Germany (0)","(very) good (1); else (0)")

# vector of tuning parameters
lambda <- exp(seq(log(31),log(1),length=50))-1

# compute BTLLasso model
m <- BTLLasso(Y = Y, X = X, lambda = lambda)

op <- par(no.readonly = TRUE)
par(mar=c(5,4,4,8))

# plot covariate paths
paths(m)

# plot parameter paths
singlepaths(m, subs = subs)

# compute 10-fold cross-validation
set.seed(5)
m.cv <- cv.BTLLasso(Y = Y, X = X, folds = 10, lambda = lambda, cores = 10)

# plot covariate paths, together with cv-optimal model
paths(m.cv)
```

```
# plot parameter paths, together with cv-optimal model
singlepaths(m.cv, subs = subs)

# compute bootstrap confidence intervals
m.boot <- boot.BTLLasso(m.cv, B = 100, cores = 25)

# plot bootstrap confidence intervals
par(mar=c(5,5,4,3))
ci.BTLLasso(m.boot, subs = subs)

par(op)

## End(Not run)
```

GLES *German Longitudinal Election Study (GLES)*

Description

Data from the German Longitudinal Election Study (GLES), see Rattinger et al. (2014). The GLES is a long-term study of the German electoral process. It collects pre- and post-election data for several federal elections, the data used here originate from the pre-election study for 2013.

Usage

```
data("GLES")
```

Format

A data frame with 1155 observations on the following 18 variables.

'SPD vs FDP' Ordinal paired comparison between SPD and FDP

'SPD vs Left Party' Ordinal paired comparison between SPD and Left Party

'SPD vs Greens' Ordinal paired comparison between SPD and Greens

'SPD vs CDU_CSU' Ordinal paired comparison between SPD and CDU/CSU

'FDP vs Left Party' Ordinal paired comparison between FDP and Left Party

'FDP vs Greens' Ordinal paired comparison between FDP and Greens

'FDP vs CDU_CSU' Ordinal paired comparison between FDP and CDU/CSU

'Left Party vs Greens' Ordinal paired comparison between Left Party and Greens

'Left Party vs CDU_CSU' Ordinal paired comparison between Left Party and CDU/CSU

'Greens vs CDU_CSU' Ordinal paired comparison between Greens and CDU/CSU

Age Age in years

Gender 0: male, 1: female

EastWest 0: West Germany, 1:East Germany

`PersEcon` Personal economic situation, 1: good or very good, 0: else

`Abitur` School leaving certificate, 1: Abitur/A levels, 0: else

`Unemployment` 1: currently unemployed, 0: else

`Church` Frequency of attendence in a church/synagogue/mosque/..., 1: at least once a month, 0: else

`Migration` Have you been a German citizen since birth? 1: yes, 0: no

Details

Variables 1 to 10 represent the response, variables 11 to 18 represent the subject-specific covariates. The response variables are ordinal, with values from 1 to 5. Low values represent string preference of the first-names party, high values represent strong preference of the last-named party.

Source

http://www.gesis.org/en/elections-home/gles/data-and-documents/

References

Rattinger, H., S. Rossteutscher, R. Schmitt-Beck, B. Wessels, and C. Wolf (2014): Pre-election cross section (GLES 2013). *GESIS Data Archive, Cologne ZA5700 Data file Version 2.0.0.*

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

Examples

```
data(GLES)
```

GLESsmall *Subset of the GLES data set with 200 observations and 4 covariates.*

Description

This is a subset of the GLES data set from the German Longitudinal Election Study (GLES), see Rattinger et al. (2014). The GLES is a long-term study of the German electoral process. It collects pre- and post-election data for several federal elections, the data used here originate from the pre-election study for 2013.

Usage

```
data("GLESsmall")
```

Format

A data frame with 200 observations on the following 14 variables.

'SPD vs FDP' Ordinal paired comparison between SPD and FDP

'SPD vs Left Party' Ordinal paired comparison between SPD and Left Party

'SPD vs Greens' Ordinal paired comparison between SPD and Greens

'SPD vs CDU_CSU' Ordinal paired comparison between SPD and CDU/CSU

'FDP vs Left Party' Ordinal paired comparison between FDP and Left Party

'FDP vs Greens' Ordinal paired comparison between FDP and Greens

'FDP vs CDU_CSU' Ordinal paired comparison between FDP and CDU/CSU

'Left Party vs Greens' Ordinal paired comparison between Left Party and Greens

'Left Party vs CDU_CSU' Ordinal paired comparison between Left Party and CDU/CSU

'Greens vs CDU_CSU' Ordinal paired comparison between Greens and CDU/CSU

Age Age in years

Gender 0: male, 1: female

EastWest 0: West Germany, 1:East Germany

PersEcon Personal economic situation, 1: good or very good, 0: else

Details

This is a subset of GLES. Variables 1 to 10 represent the response, variables 11 to 14 represent the subject-specific covariates. The response variables are ordinal, with values from 1 to 5. Low values represent string preference of the first-names party, high values represent strong preference of the last-named party.

Source

http://www.gesis.org/en/elections-home/gles/data-and-documents/

References

Rattinger, H., S. Rossteutscher, R. Schmitt-Beck, B. Wessels, and C. Wolf (2014): Pre-election cross section (GLES 2013). *GESIS Data Archive, Cologne ZA5700 Data file Version 2.0.0.*

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

Examples

```
## Not run:
# load data set
data(GLESsmall)

# define response and covariate matrix
X <- scale(GLESsmall[, 11:14])
```

```
Y <- as.matrix(GLESsmall[, 1:10])

# vector of subtitles, containing the coding of the single covariates
subs <- c("(in years)","female (1); male (0)",
"East Germany (1); West Germany (0)","(very) good (1); else (0)")

# vector of tuning parameters
lambda <- exp(seq(log(31),log(1),length=50))-1

# compute BTLLasso model
m <- BTLLasso(Y = Y, X = X, lambda = lambda)

op <- par(no.readonly = TRUE)
par(mar=c(5,4,4,8))

# plot covariate paths
paths(m)

# plot parameter paths
singlepaths(m, subs = subs)

# compute 10-fold cross-validation
set.seed(5)
m.cv <- cv.BTLLasso(Y = Y, X = X, folds = 10, lambda = lambda, cores = 10)

# plot covariate paths, together with cv-optimal model
paths(m.cv)

# plot parameter paths, together with cv-optimal model
singlepaths(m.cv, subs = subs)

# compute bootstrap confidence intervals
m.boot <- boot.BTLLasso(m.cv, B = 100, cores = 25)

# plot bootstrap confidence intervals
par(mar=c(5,5,4,3))
ci.BTLLasso(m.boot, subs = subs)

par(op)

## End(Not run)
```

---

paths *Plot covariate paths for BTLLasso*

Description

Plots paths for every covariate of a BTLLasso object or a cv.BTLLasso object. In contrast, to singlepaths, only one plot is created, every covariate is illustrated by one path. For cv.BTLLasso object, the optimal model according to the cross-validation is marked by a vertical dashed line.

Usage

```
paths(model)
```

Arguments

model	BTLLasso or cv.BTLLasso object

Details

Plots for BTLLasso and cv.BTLLasso objects only differ by the additional vertical line indicating the optimal model according to cross-validation.

Author(s)

Gunther Schauberger
<gunther@stat.uni-muenchen.de>
http://www.statistik.lmu.de/~schauberger/

References

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

See Also

BTLLasso, cv.BTLLasso, singlepaths

Examples

```
## Not run:
# load data set
data(GLESsmall)

# define response and covariate matrix
X <- scale(GLESsmall[, 11:14])
Y <- as.matrix(GLESsmall[, 1:10])

# vector of subtitles, containing the coding of the single covariates
subs <- c("(in years)","female (1); male (0)",
"East Germany (1); West Germany (0)","(very) good (1); else (0)")

# vector of tuning parameters
lambda <- exp(seq(log(31),log(1),length=50))-1

# compute BTLLasso model
m <- BTLLasso(Y = Y, X = X, lambda = lambda)

op <- par(no.readonly = TRUE)
par(mar=c(5,4,4,8))
```

```
# plot covariate paths
paths(m)

# compute 10-fold cross-validation
set.seed(5)
m.cv <- cv.BTLLasso(Y = Y, X = X, folds = 10, lambda = lambda, cores = 10)

# plot covariate paths, together with cv-optimal model
paths(m.cv)

par(op)

## End(Not run)
```

`print.cv.BTLLasso` *Print function for cv.BTLLasso objects*

Description

Prints the most important output of cv.BTLLasso objects.

Usage

```
## S3 method for class 'cv.BTLLasso'
print(x, ...)
```

Arguments

`x`	`cv.BTLLasso` object
`...`	possible further arguments for print command

Value

`x`	cv.BTLLasso object

Author(s)

Gunther Schauberger
<gunther@stat.uni-muenchen.de>
http://www.statistik.lmu.de/~schauberger/

References

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

See Also

cv.BTLLasso

Examples

```
## Not run:
## load data set
data(GLESsmall)

# define response and covariate matrix
X <- scale(GLESsmall[, 11:14])
Y <- as.matrix(GLESsmall[, 1:10])

# vector of subtitles, containing the coding of the single covariates
subs <- c("(in years)","female (1); male (0)",
"East Germany (1); West Germany (0)","(very) good (1); else (0)")

# vector of tuning parameters
lambda <- exp(seq(log(31),log(1),length=50))-1


# compute 10-fold cross-validation
set.seed(5)
m.cv <- cv.BTLLasso(Y = Y, X = X, folds = 10, lambda = lambda, cores = 10)

print(m.cv)

## End(Not run)
```

`singlepaths` *Plot parameter paths for BTLLasso*

Description

Plots single paths for every parameter of a BTLLasso object or a cv.BTLLasso object. In contrast, to paths, one plot per covariate is created, every single parameter is illustrated by one path. For cv.BTLLasso object, the optimal model according to the cross-validation is marked by a vertical dashed line.

Usage

```
singlepaths(model, colors = NULL, equal.ranges = FALSE, subs = NULL)
```

Arguments

`model`	BTLLasso or cv.BTLLasso object
`colors`	Optional. If specified, vector with length equal to the number of objects. Each object can be represented by another color.

equal.ranges	Should all single plots (for different covariates) have equal ranges on the y-axes. FALSE by default.
subs	Optional vector of subtitles for the single plots. Can be used to note the encoding of the single covariates, especially for dummy variables.

Details

Plots for BTLLasso and cv.BTLLasso objects only differ by the additional vertical line indicating the optimal model according to cross-validation.

Author(s)

Gunther Schauberger
<gunther@stat.uni-muenchen.de>
http://www.statistik.lmu.de/~schauberger/

References

Schauberger, Gunther and Tutz, Gerhard (2015): Modelling Heterogeneity in Paired Comparison Data - an L1 Penalty Approach with an Application to Party Preference Data, *Department of Statistics, LMU Munich*, Technical Report 183

See Also

BTLLasso, cv.BTLLasso, paths

Examples

```
## Not run:
# load data set
data(GLESsmall)

# define response and covariate matrix
X <- scale(GLESsmall[, 11:14])
Y <- as.matrix(GLESsmall[, 1:10])

# vector of subtitles, containing the coding of the single covariates
subs <- c("(in years)","female (1); male (0)",
"East Germany (1); West Germany (0)","(very) good (1); else (0)")

# vector of tuning parameters
lambda <- exp(seq(log(31),log(1),length=50))-1

# compute BTLLasso model
m <- BTLLasso(Y = Y, X = X, lambda = lambda)

op <- par(no.readonly = TRUE)
par(mar=c(5,4,4,8))

# plot parameter paths
singlepaths(m, subs = subs)
```

```
# compute 10-fold cross-validation
set.seed(5)
m.cv <- cv.BTLLasso(Y = Y, X = X, folds = 10, lambda = lambda, cores = 10)

# plot parameter paths, together with cv-optimal model
singlepaths(m.cv, subs = subs)

par(op)

## End(Not run)
```

Index