Package ‘splmm’

June 13, 2024

Type Package

Title Simultaneous Penalized Linear Mixed Effects Models

Version 1.2.0

Date 2024-06-12

Maintainer Eli Sun <eli_sun@urmc.rochester.edu>

Description Contains functions that fit linear mixed-effects models
for high-dimensional data (p>>n) with penalty for both the fixed effects and random effects for variable selection.
The details of the algorithm can be found in Luoying Yang PhD thesis (Yang and Wu 2020). The algorithm implementation
is based on the R package 'lmmlasso'.
Reference: Yang L, Wu TT (2020). Model-Based Clustering of Longitudinal Data in High-Dimensionality. Unpublished thesis.

License GPL-3

Imports Rcpp (>= 1.0.1), emulator, miscTools, penalized, ggplot2,
gridExtra, plot3D, MASS, progress, methods

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Depends R (>= 3.5.0)

Author Luoying Yang [aut],
Eli Sun [aut, cre],
Tong Tong Wu [aut]

Repository CRAN

Date/Publication 2024-06-13 09:40:02 UTC

Contents

`splmm-package` *Simultaneous Penalized Linear Mixed Effects Models*

Description

Contains functions that fit linear mixed-effects models for high-dimensional data (p»n) with penalty for both the fixed effects and random effects for variable selection. The details of the algorithm can be found in Luoying Yang PhD thesis (Yang and Wu 2020). The algorithm implementation is based on the R package 'lmmlasso'. Reference: Yang L, Wu TT (2020). Model-Based Clustering of Longitudinal Data in High-Dimensionality. Unpublished thesis.

Details

The DESCRIPTION file:

Package:	splmm
Type:	Package
Title:	Simultaneous Penalized Linear Mixed Effects Models
Version:	1.2.0
Date:	2024-06-12
Authors@R:	c(person(given = "Luoying", family = "Yang", role = c("aut"), email = "lyang19@u.rochester.edu"), pers
Maintainer:	Eli Sun <eli_sun@urmc.rochester.edu>
Description:	Contains functions that fit linear mixed-effects models for high-dimensional data (p»n) with penalty for
License:	GPL-3
Imports:	Rcpp (>= 1.0.1), emulator, miscTools, penalized, ggplot2, gridExtra, plot3D, MASS, progress, methods
LinkingTo:	Rcpp, RcppArmadillo
NeedsCompilation:	yes
Packaged:	2024-06-05 18:57:45 UTC; elisunorig
Depends:	R (>= 3.5.0)
Author:	Luoying Yang [aut], Eli Sun [aut, cre], Tong Tong Wu [aut]
Repository:	CRAN
Date/Publication:	2021-09-08 10:00:02 UTC

Index of help topics:

```
cognitive               Kenya School Lunch Intervention Cognitive
                        Dataset
plot.splmm              Plot the tuning results of a 'splmm.tuning'
```

	object
plot3D.splmm	3D Plot the tuning results of a "splmm.tuning" object when tuning over both lambda 1 and lambda 2 grids
print.splmm	Print a short summary of a splmm object.
simulated_data	Dataset simulated for toy example
splmm	Function to fit linear mixed-effects model with double penalty for fixed effects and random effects
splmm-package	Simultaneous Penalized Linear Mixed Effects Models
splmmControl	Options for the 'splmm' Algorithm
splmmTuning	Tuning funtion of "splmm" object
summary.splmm	Summarize an 'splmm' object

Contains functions that fit linear mixed-effects models for high-dimensional data (p»n) with penalty for both the fixed effects and random effects for variable selection.

Author(s)

Luoying Yang [aut], Eli Sun [aut, cre], Tong Tong Wu [aut]

Maintainer: Eli Sun <eli_sun@urmc.rochester.edu>

References

Luoying Yang PhD thesis

SCHELLDORFER, J., BUHLMANN, P. and DE GEER, S.V. (2011), Estimation for High-Dimensional Linear Mixed-Effects Models Using L1-Penalization. Scandinavian Journal of Statistics, 38: 197-214. doi:10.1111/j.1467-9469.2011.00740.x

Examples

```
## Use splmm on the Kenya school cognitive data set


data(cognitive)

x <- model.matrix(ravens ~schoolid+treatment+year+sex+age_at_time0
                  +height+weight+head_circ+ses+mom_read+mom_write
                  +mom_edu, cognitive)
z <- x

fit <- splmm(x=x,y=cognitive$ravens,z=z,grp=cognitive$id,lam1=0.1,
lam2=0.1,penalty.b="lasso", penalty.L="lasso")
summary(fit)
```

`cognitive` *Kenya School Lunch Intervention Cognitive Dataset*

Description

In the Kenya school lunch intervention, children were given one of four school lunch interventions: meat, milk, calorie, or control. The first three groups were fed a school lunch of a stew called githeri supplemented with either meat, milk, or oil to create a lunch with a given caloric level, while the control group did not receive a lunch. Three schools were randomized to each group and the lunch program is the same for all children within a school. The data is available in *modeling-longitudinal-data-rob-weiss* and is broken up into sub data sets from four domains: Anthropometry, Cognitive, Morbidity, and Nutrition. We will be using the cognitive dataset for analyzing how the cognition level of the school children change over time and how the change is associated with other variables. The main cognitive measures is Raven's colored progressive matrices (Raven's), a measure of cognitive ability. There are three additional response variables: arithmetic score (arithmetic), verbal meaning (vmeaning), and total digit span score (dstotal) where digit span is a test of memory while others are considered measures of intelligence or education. The cognitive measurement baseline was taken prior to the lunch program onset and measurements were assessed at up to five times, called rounds, for each subject. More information about this dataset please see the reference:

Robert E Weiss.Modeling longitudinal data. Springer Science & Business Media, 2005.

Usage

```
data(cognitive)
```

Format

A data frame of 1562 observations and 26 variables.

Grouping variable. Unique ID for each subject.

schoolid School id 1-12.

treatment Calorie, meat, milk, control

rn round.

year Time in years from baseline.

revans Raven's colored matrices score.

arithmetic Arithmetic score.

vmeaning Verbal meaning.

dstotal Total digit span score.

sex Girl or Boy.

age_at_time0 age at baseline.

height height at baseline.

weight weight at baseline.

head_circ Head circumference at baseline.

ses Socio-Economic Status score.

mom_read Mother's reading test.

mom_write Mother's writing test.

mom_edu Mother's years of educations.

morbscore Morbidity score: none/mild/severe.

complete Logical variable specifying whether the subject has all five rounds. 1-Yes, 0-No.

rnone Logical variable specifying whether the observation is the baseline. 1-round one (baseline), 0-not round one.

relmonth Time in months from baseline.

Examples

```
data(cognitive)
```

`plot.splmm` *Plot the tuning results of a* `splmm.tuning` *object*

Description

This function inputs an `splmm.tuning` object and plot the model selection criterion values over the tuning parameters grid.

Usage

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

Arguments

`x`	a `'splmm.tuning'` object
`...`	not used

Value

A line plot of BIC, AIC, BICC, EBIC values against lam1 or lam2 depending on the inout.

See Also

`plot.splmm`

Examples

```
data(cognitive)

x <- model.matrix(ravens ~schoolid+treatment+year+sex+age_at_time0
                  +height+weight+head_circ+ses+mom_read+mom_write
                  +mom_edu, cognitive)
z <- x

## Tuning over lambda1 grid
lam1 = seq(0.1,0.5,0.1)
lam2 = 0.1
fit1 <-splmmTuning(x=x,y=cognitive$ravens,z=z,grp=cognitive$id,lam1=lam1,
lam2=lam2,penalty.b="scad", penalty.L="scad")
plot.splmm(fit1)
```

`plot3D.splmm` *3D Plot the tuning results of a* 'splmm.tuning' *object when tuning over both lambda 1 and lambda 2 grids*

Description

This function inputs an 'splmm.tuning' object and plot the model selection criterion values in a 3D plot over the lambda 1 and lambda 2 tuning parameters grid.

Usage

```
## S3 method for class 'splmm'
plot3D(x, criteria=c("BIC","AIC","BICC","EBIC"),type=c("line","surface"),...)
```

Arguments

x	a 'splmm.tuning' object with both lam1.tuning=TRUE and lam2.tuning=TRUE
criteria	A parameter specifying whether the criteria value the user want to plot is BIC, AIC, BICC or EBIC. The default is BIC
type	A parameter specifying which type of 3D plot to use for plotting. Currently the available options include line plot and surface plot. The default is surface plot.
...	not used

Value

A 3D line/surface plot of BIC/AIC/BICC/EBIC values against lam1 and lam2.

See Also

`plot3D`

Examples

```
data(cognitive)

x <- model.matrix(ravens ~schoolid+treatment+year+sex+age_at_time0
                  +height+weight+head_circ+ses+mom_read+mom_write
                  +mom_edu, cognitive)
z <- x

## Tuning over lambda1 grid and lambda2 grid
lam1 = seq(0.1,0.5,0.1)
lam2 = seq(0.1,0.5,0.1)
fit1 <-splmmTuning(x=x,y=cognitive$ravens,z=z,grp=cognitive$id,lam1=lam1,
lam2=lam2,penalty.b="scad", penalty.L="scad")
plot3D.splmm(fit1)
```

`print.splmm` *Print a short summary of a splmm object.*

Description

Prints a short summary of an `'splmm'` object comprising information about the nonzero fixed-effects coefficients and the nonzero random effect variance components.

Usage

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

Arguments

`x`	a `'splmm'` object
`...`	not used

Value

No return value, a print-out of a `'splmm'` object's short summary is produced.

See Also

`print.splmm`

Examples

```
data(simulated_data)

set.seed(144)
fit = splmm(x=simulated_data$x,y=simulated_data$y,
z=simulated_data$z,grp=simulated_data$grp,
lam1=0.1,lam2=0.01, penalty.b="scad", penalty.L="scad")
print(fit)


data(cognitive)

x <- model.matrix(ravens ~schoolid+treatment+year+sex+age_at_time0
                  +height+weight+head_circ+ses+mom_read+mom_write
                  +mom_edu, cognitive)
z <- x

fit <- splmm(x=x,y=cognitive$ravens,z=z,grp=cognitive$id,lam1=0.1,
lam2=0.1,penalty.b="scad", penalty.L="scad")
print(fit)
```

`simulated_data` *Dataset simulated for toy example*

Description

A toy dataset simulated for demonstration for the `splmm` function.

Usage

```
data(simulated_data)
```

Format

y Response variable.

x Fixed-effects design matrix.

z Random-effects design matrix

grp Subject ID.

Examples

```
data(simulated_data)
```

splmm — *Function to fit linear mixed-effects model with double penalty for fixed effects and random effects*

Description

All the details of the algorithm can be found in the manuscript.

Usage

```
splmm(x, y, z, grp, lam1, lam2, nonpen.b=1,nonpen.L=1,penalty.b=c("lasso","scad"),
      penalty.L=c("lasso","scad"),CovOpt=c("nlminb","optimize"),
      standardize=TRUE,control=splmmControl())

## Default S3 method:
splmm(x, y, z, grp, lam1, lam2, nonpen.b=1,nonpen.L=1,penalty.b=c("lasso","scad"),
      penalty.L=c("lasso","scad"),CovOpt=c("nlminb","optimize"),
      standardize=TRUE,control=splmmControl())
```

Arguments

`x`	matrix of dimension N x p including the fixed-effects covariables. An intercept has to be included in the first column as (1,...,1).
`y`	response variable of length N.
`z`	random effects matrix of dimension N x q. It has to be a matrix, even if q=1.
`grp`	grouping variable of length N
`lam1`	regularization parameter for fixed effects penalization.
`lam2`	regularization parameter for random effects penalization.
`nonpen.b`	Index of indices of fixed effects not penalized. The default value is 1, which means the fixed intercept is not penalized
`nonpen.L`	Index of indices of random effects not penalized. The default value is 1, which means the random intercept is not penalized
`penalty.b`	The penalty method for fixed effects penalization. Currently available options include LASSO penalty and SCAD penalty.
`penalty.L`	The penalty method for fixed effects penalization. Currently available options include LASSO penalty and SCAD penalty.
`CovOpt`	which optimization routine should be used for updating the variance parameter. The available options include optimize and nlminb. nlminb uses the estimate of the last iteration as a starting value. nlminb is faster if there are many Gauss-Seidel iterations.
`standardize`	A logical parameter specifying whether the fixed effects matrix x and random effects matrix z should be standardized such that each column has mean 0 and standard deviation 1. The default value is `TRUE`
`control`	control parameters for the algorithm and the Armijo Rule, see `'splmmControl'` for the details

Value

A 'splmm' object is returned, for which `coef`,`resid`, `fitted`, `print`, `summary` methods exist.

`data`	data set used for fitting the model, as a list with four components: x, y, z, grp (see above)
`coefInit`	list of the starting values for beta, random effects covariance structure, and variance structure
`penalty.b`	The penalty method for fixed effects penalization.
`penalty.L`	The penalty method for random effects penalization.
`nonpen.b`	Index of indices of fixed effects not penalized.
`nonpen.L`	Index of indices of random effects not penalized.
`lambda1`	regularization parameter for fixed effects penalization scaled by the number of subjects.
`lambda2`	regularization parameter for random effects penalization the number of subjects.
`sigma`	standard deviation $\hat{\sigma}$ of the errors
`D`	The estimates of the random effects covariance matrix $\hat{D}$.
`Lvec`	Vectorized $\hat{L}$, the lower triangular matrix of $\hat{D}$ from Cholesky Decomposition.
`coefficients`	estimated fixed-effects coefficients $\hat{\beta}$
`random`	vector with random effects, sorted by groups
`ranef`	vector with random effects, sorted by effect
`u`	vector with the standardized random effects, sorted by effect
`fixef`	estimated fixed-effects coeffidients $\hat{\beta}$
`fitted.values`	The fitted values $\hat{y} = \hat{X}\beta + Z\hat{b}_i$
`residuals`	raw residuals $y - \hat{y}$
`corD`	Correlation matrix of the random effects
`logLik`	value of the log-likelihood function
`deviance`	deviance=-2*logLik
`npar`	Number of parameters. Corresponds to the cardinality of the set of nonzero `coefficients` plus the number of nonzero variance in `D`
`aic`	AIC
`bic`	BIC
`bicc`	Modified BIC defined by Wang et al (2009)
`ebic`	Extended BIC defined by Chen and Chen (2008)
`converged`	Does the algorithm converge? 0: correct convergence ; an odd number means that maxIter was reached ; an even number means that the Armijo step was not succesful. For each unsuccessfull Armijo step, 2 is added to converged. If converged is large compared to the number of iterations `counter`, you may increase maxArmijo.
`counter`	The number of iterations used.

stopped	logical indicating whether the algorithm stopped due to too many parameters, if yes need to increase lam1 or lam2
CovOpt	optimization routine
control	see splmmControl
objective	Value of the objective function at the final estimates
call	call

Examples

```
### Use splmm for a toy dataset.

data(simulated_data)

set.seed(144)
fit = splmm(x=simulated_data$x,y=simulated_data$y,
z=simulated_data$z,grp=simulated_data$grp,
lam1=0.1,lam2=0.01, penalty.b="scad", penalty.L="scad")
summary(fit)



## Use splmm on the Kenya school cognitive data set


data(cognitive)

x <- model.matrix(ravens ~schoolid+treatment+year+sex+age_at_time0
                  +height+weight+head_circ+ses+mom_read+mom_write
                  +mom_edu, cognitive)
z <- x

fit <- splmm(x=x,y=cognitive$ravens,z=z,grp=cognitive$id,lam1=0.1,
lam2=0.1,penalty.b="lasso", penalty.L="lasso")
summary(fit)
```

---

splmmControl *Options for the 'splmm' Algorithm*

Description

Definition of various kinds of options in the algorithm.

Usage

```
splmmControl(tol=10^(-4),trace=1,maxIter=100,maxArmijo=20,number=5,a_init=1,
        delta=0.1,rho=0.001,gamma=0,lower=10^(-6),upper=10^8,seed=532,VarInt=c(0,10),
            CovInt=c(-5,5),thres=10^(-4))
```

Arguments

`tol`	convergence tolerance
`trace`	integer. 1 prints no output, 2 prints warnings, 3 prints the current function values and warnings (not recommended)
`maxIter`	maximum number of (outer) iterations
`maxArmijo`	maximum number of steps to be chosen in the Armijo Rule. If the maximum is reached, the algorithm continues with optimizing the next coordinate.
`number`	integer. Determines the active set algorithm. The zero fixed-effects coefficients are only updated each number iteration. It may be that a smaller number increases the speed of the algorithm. Use $0 \leq number \leq 5$.
`a_init`	α_{init} in the Armijo step. See Schelldorfer et. al. (2010).
`delta`	δ in the Armijo step. See Schelldorfer et. al. (2010)
`rho`	ρ in the Armijo step. See Schelldorfer et. al. (2010)
`gamma`	γ in the Armijo step. See Schelldorfer et. al. (2010)
`lower`	lower bound for the Hessian
`upper`	upper bound for the Hessian
`seed`	set.seed for calculating the starting value, which performs a 10-fold cross-validation.
`VarInt`	Only for opt="optimize". The interval for the variance parameters used in "optimize". See help("optimize")
`CovInt`	Only for opt="optimize". The interval for the covariance parameters used in "optimize". See help("optimize")
`thres`	If a variance or covariance parameter has smaller absolute value than thres, the parameter is set to exactly zero.

Details

For the Armijo step parameters, see Bertsekas (2003)

Value

Exactly the same as `arguments`.

`splmmTuning` *Tuning funtion of* `'splmm'` *object*

Description

This function fits `'splmm'` function over grids of lambda1 and/or lambda2 and determine the best fit model based on model selection information criterion. The function takes a scalar or a grid of lambda1 and/or lambda2 and determine the optimal tuning parameter value for the best model fit. If both lambda1 and lambda2 are inputted as scalars, an `'splmm'` object is returned; if either or both lambda1 and lambda2 are inputted as grids, an `'splmm.tuning'` object is returned. Currently the model selection criterion include AIC and BIC, and BIC is used to determine the optimal model.

Usage

```
splmmTuning(x, y, z, grp, lam1.seq, lam2.seq, nonpen.b=1,nonpen.L=1,
                          penalty.b=c("lasso","scad"),
                          penalty.L=c("lasso","scad"),
                          CovOpt=c("nlminb","optimize"),
                          standardize=TRUE,control=splmmControl())
```

Arguments

`x`	matrix of dimension N x p including the fixed-effects covariables. An intercept has to be included in the first column as (1,...,1).
`y`	response variable of length N.
`z`	random effects matrix of dimension N x q. It has to be a matrix, even if q=1.
`grp`	grouping variable of length N
`lam1.seq`	a grid of regularization parameter for fixed effects penalization, could be a scalar if no need to tune.
`lam2.seq`	a grid of regularization parameter for random effects penalization, could be a scalar if no need to tune.
`nonpen.b`	Index of indices of fixed effects not penalized. The default value is 1, which means the fixed intercept is not penalized.
`nonpen.L`	Index of indices of random effects not penalized. The default value is 1, which means the random intercept is not penalized.
`penalty.b`	The penalty method for fixed effects penalization. Currently available options include LASSO penalty and SCAD penalty.
`penalty.L`	The penalty method for fixed effects penalization. Currently available options include LASSO penalty and SCAD penalty.
`CovOpt`	which optimization routine should be used for updating the variance parameter. The available options include optimize and nlminb. nlminb uses the estimate of the last iteration as a starting value. nlminb is faster if there are many Gauss-Seidel iterations.
`standardize`	A logical parameter specifying whether the fixed effects matrix x and random effects matrix z should be standardized such that each column has mean 0 and standard deviation 1. The default value is `TRUE`.
`control`	control parameters for the algorithm and the Armijo Rule, see `'splmmControl'` for the details.

Value

A `'splmm.tuning'` object is returned, for which `plot` method exist.

`lam1.seq`	lambda1 grid used for tuning. Only available when lambda1 is inputted as a vector.
`lam2.seq`	lambda2 grid used for tuning. Only available when lambda2 is inputted as a vector.

BIC.lam1	A vector of BIC values of splmm models fitting over a lambda1 grid.
BIC.lam2	A vector of BIC values of splmm models fitting over a lambda2 grid.
fit.BIC	An array of BIC values of splmm models fitting over lambda 1 grid x lambda2 grid.
AIC.lam1	A vector of AIC values of splmm models fitting over a lambda1 grid.
AIC.lam2	A vector of AIC values of splmm models fitting over a lambda2 grid.
fit.AIC	An array of AIC values of splmm models fitting over lambda 1 grid x lambda2 grid.
BICC.lam1	A vector of BICC values of splmm models fitting over a lambda1 grid.
BICC.lam2	A vector of BICC values of splmm models fitting over a lambda2 grid.
fit.BICC	An array of BICC values of splmm models fitting over lambda 1 grid x lambda2 grid.
EBIC.lam1	A vector of EBIC values of splmm models fitting over a lambda1 grid.
EBIC.lam2	A vector of EBIC values of splmm models fitting over a lambda2 grid.
fit.EBIC	An array of EBIC values of splmm models fitting over lambda 1 grid x lambda2 grid.
min.BIC	The minimum BIC value from tuning over a grid. This is only available when either lambda1 or lambda2 is a scalar.
min.AIC	The minimum AIC value from tuning over a grid. This is only available when either lambda1 or lambda2 is a scalar.
min.BICC	The minimum BICC value from tuning over a grid. This is only available when either lambda1 or lambda2 is a scalar.
min.EBIC	The minimum EBIC value from tuning over a grid. This is only available when either lambda1 or lambda2 is a scalar.
best.model	The index of the optimal model. This is only available when either lambda1 or lambda2 is a scalar.
best.fit	The optimal model chosen by the minimum BIC as an `splmm` object.
min.lam1	lambda1 value that results in the optimal model. This is only available when input lambda1 is a vector.
min.lam2	lambda2 value that results in the optimal model. This is only available when input lambda2 is a vector.
lam1.tuning	A `logical` parameter specifying if tuning is performed over lamdbda1 grid. `lam1.tuning=TRUE` if input lambda1 is a vector.
lam2.tuning	A `logical` parameter specifying if tuning is performed over lamdbda2 grid. `lam1.tuning=TRUE` if input lambda2 is a vector.

Examples

```
data(cognitive)

x <- model.matrix(ravens ~schoolid+treatment+year+sex+age_at_time0
                  +height+weight+head_circ+ses+mom_read+mom_write
```

```
                  +mom_edu, cognitive)
z <- x

## Tuning over lambda1 grid
lam1 = seq(0.1,0.5,0.1)
lam2 = 0.1
fit1 <-splmmTuning(x=x,y=cognitive$ravens,z=z,grp=cognitive$id,lam1.seq=lam1,
lam2.seq=lam2,penalty.b="scad", penalty.L="scad")
plot.splmm(fit1)

## Tuning over lambda2 grid
lam1 = 0.1
lam2 = seq(0.1,0.5,0.1)
fit2 <-splmmTuning(x=x,y=cognitive$ravens,z=z,grp=cognitive$id,lam1.seq=lam1,
lam2.seq=lam2,penalty.b="scad", penalty.L="scad")
plot.splmm(fit2)

## Tuning over both lambda1 and lambda2 grid
lam1 = seq(0.1,0.5,0.2)
lam2 = seq(0.1,0.5,0.2)
fit3 <-splmmTuning(x=x,y=cognitive$ravens,z=z,grp=cognitive$id,lam1.seq=lam1,
lam2.seq=lam2,penalty.b="scad", penalty.L="scad")
plot.splmm(fit3)
```

---

summary.splmm — *Summarize an 'splmm' object*

Description

Providing an elaborate summary of a `'splmm'` object.

Usage

```
## S3 method for class 'splmm'
summary(object, ...)
```

Arguments

`object`	a `'splmm'` object
`...`	not used.

Details

This functions shows a detailed summary of a `'splmm'` object.

Value

No return value, a print-out of a `'splmm'` object's detailed summary is produced.

Examples

```
data(simulated_data)

set.seed(144)
fit = splmm(x=simulated_data$x,y=simulated_data$y,
z=simulated_data$z,grp=simulated_data$grp,
lam1=0.1,lam2=0.01, penalty.b="scad", penalty.L="scad")
summary(fit)


data(cognitive)

x <- model.matrix(ravens ~schoolid+treatment+year+sex+age_at_time0
                  +height+weight+head_circ+ses+mom_read+mom_write
                  +mom_edu, cognitive)
z <- x

fit <- splmm(x=x,y=cognitive$ravens,z=z,grp=cognitive$id,lam1=0.1,lam2=0.1,
penalty.b="scad", penalty.L="scad")
summary(fit)
```

Index