

Package ‘glmmLasso’

October 13, 2022

Type Package

Title Variable Selection for Generalized Linear Mixed Models by
L1-Penalized Estimation

Version 1.6.2

Date 2022-08-22

Author Andreas Groll

Maintainer Andreas Groll <groll@statistik.tu-dortmund.de>

Description A variable selection approach for generalized linear mixed models by L1-penalized estimation is provided, see Groll and Tutz (2014) <[doi:10.1007/s11222-012-9359-z](https://doi.org/10.1007/s11222-012-9359-z)>. See also Groll and Tutz (2017) <[doi:10.1007/s10985-016-9359-y](https://doi.org/10.1007/s10985-016-9359-y)> for discrete survival models including heterogeneity.

Imports stats, minqa, Matrix, Rcpp (>= 0.12.12), methods, GMMBoost

LinkingTo Rcpp, RcppEigen

License GPL-2

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-08-23 09:50:02 UTC

R topics documented:

acat	2
cumulative	3
glmmLasso	4
glmmLassoControl	9
knee	11
soccer	12
Index	14

acat	<i>Family Object for Ordinal Regression with Adjacent Categories Probabilities</i>
------	--

Description

Provides necessary family components to fit an adjacent categories regression model to an ordered response based on the corresponding (multivariate) binary design representation.

Usage

```
acat()
```

Details

For a response variable Y with ordered values $1, 2, \dots, M + 1$ the design of the corresponding (multivariate) binary response representation is automatically created by the [glmLasso](#) function. The result is a linear predictor matrix η with n rows and M columns.

Based on this $(n \times M)$ predictor matrix η or on the corresponding $(n \times M)$ matrix μ the below mentioned family components can be calculated.

Value

linkinv	function: the inverse of the link function as a function of eta.
deriv.mat	function: derivative function as a function of the mean (not of eta as normally).
SigmaInv	function: the inverse of the variance as a function of the mean.
family	character: the family name.
multivariate	Logical. Is always set to TRUE if the family is used.

Author(s)

Andreas Groll <groll@math.lmu.de>

References

Agresti, A. (2013) *Categorical Data Analysis*, 3rd ed. Hoboken, NJ, USA: Wiley.
 Simonoff, J. S. (2003) *Analyzing Categorical Data*, New York: Springer-Verlag.
 Tutz, G. (2012) *Regression for Categorical Data*, Cambridge University Press.

See Also

[cumulative](#), [glmLasso](#), [knee](#)

Examples

```
## Not run:
data(knee)

knee[,c(2,4:6)]<-scale(knee[,c(2,4:6)],center=TRUE,scale=TRUE)
knee<-data.frame(knee)

## fit adjacent category model
glm.obj <- glmmLasso(pain ~ time + th + age + sex, rnd = NULL,
  family = acat(), data = knee, lambda=10,
  switch.NR=TRUE, control=list(print.iter=TRUE))

summary(glm.obj)

## End(Not run)
```

cumulative

*Family Object for Ordinal Regression with Cumulative Probabilities***Description**

Provides necessary family components to fit a proportional odds regression model to an ordered response based on the corresponding (multivariate) binary design representation.

Usage

```
cumulative()
```

Details

For a response variable Y with ordered values $1, 2, \dots, M + 1$ the design of the corresponding (multivariate) binary response representation is automatically created by the `glmmLasso` function. The result is a linear predictor matrix η with n rows and M columns.

Based on this (nxM) predictor matrix η or on the corresponding (nxM) matrix μ the below mentioned family components can be calculated.

Solely the logit link is implemented, hence, a proportional odds model is fitted.

Value

<code>linkinv</code>	function: the inverse of the link function as a function of eta. Solely the logit link is implemented, hence, the response function $h(\eta) = \exp(\eta)/(1 + \exp(\eta))$ is used.
<code>deriv.mat</code>	function: derivative function as a function of the mean (not of eta as normally).
<code>SigmaInv</code>	function: the inverse of the variance as a function of the mean.
<code>family</code>	character: the family name.
<code>multivariate</code>	Logical. Is always set to TRUE if the family is used.

Author(s)

Andreas Groll <groll@math.lmu.de>

References

- Agresti, A. (2013) *Categorical Data Analysis*, 3rd ed. Hoboken, NJ, USA: Wiley.
- Dobson, A. J. and Barnett, A. (2008) *An Introduction to Generalized Linear Models*, 3rd ed. Boca Raton: Chapman & Hall/CRC Press.
- McCullagh, P. and Nelder, J. A. (1989) *Generalized Linear Models*, 2nd ed. London: Chapman & Hall.
- Simonoff, J. S. (2003) *Analyzing Categorical Data*, New York: Springer-Verlag.
- Tutz, G. (2012) *Regression for Categorical Data*, Cambridge University Press.
- Yee, T. W. and Wild, C. J. (1996) Vector generalized additive models. *Journal of the Royal Statistical Society, Series B, Methodological*, **58**, 481–493.

See Also

[acat](#), [glmmLasso](#), [knee](#)

Examples

```
## Not run:
data(knee)

knee[,c(2,4:6)]<-scale(knee[,c(2,4:6)],center=TRUE,scale=TRUE)
knee<-data.frame(knee)

## fit adjacent category model
glm.obj <- glmmLasso(pain ~ time + th + age + sex, rnd = NULL,
  family = cumulative(), data = knee, lambda=10,
  switch.NR=TRUE, control=list(print.iter=TRUE))

summary(glm.obj)

## End(Not run)
```

glmmLasso

Variable Selection for Generalized Linear Mixed Models by L1-Penalized Estimation.

Description

A variable selection approach for generalized linear mixed models by L1-penalized estimation is provided.

Usage

```
glmmLasso(fix=formula, rnd=formula, data, lambda, family = gaussian(link="identity"),
          switch.NR=FALSE, final.re=FALSE, control = list())
```

Arguments

<code>fix</code>	a two-sided linear formula object describing the fixed-effects part of the model, with the response on the left of a <code>~</code> operator and the terms, separated by <code>+</code> operators, on the right. For categorical covariables use <code>as.factor(.)</code> in the formula. Note, that the corresponding dummies are treated as a group and are updated blockwise
<code>rnd</code>	a two-sided linear formula object describing the random-effects part of the model, with the grouping factor on the left of a <code>~</code> operator and the random terms, separated by <code>+</code> operators, on the right; alternatively, the random effects design matrix can be given directly (with suitable column names). If set to <code>NULL</code> , no random effects are included.
<code>data</code>	the data frame containing the variables named in <code>formula</code> .
<code>lambda</code>	the penalty parameter that controls the shrinkage of fixed terms and controls the variable selection. The optimal penalty parameter is a tuning parameter of the procedure that has to be determined, e.g. by use of information criteria or cross validation. (See details or the quick demo for an example.)
<code>family</code>	a GLM family, see <code>glm</code> and <code>family</code> . Also ordinal response models can be fitted: use <code>family=acat()</code> and <code>family=cumulative()</code> for the fitting of an adjacent category or cumulative model, respectively. If <code>family</code> is missing then a linear mixed model is fit; otherwise a generalized linear mixed model is fit.
<code>switch.NR</code>	logical. Should the algorithm switch to a Newton-Raphson update step, when reasonable? Default is <code>FALSE</code> .
<code>final.re</code>	logical. Should the final Fisher scoring re-estimation be performed? Default is <code>FALSE</code> .
<code>control</code>	a list of control values for the estimation algorithm to replace the default values returned by the function <code>glmmLassoControl</code> . Defaults to an empty list.

Details

The `glmmLasso` algorithm is a gradient ascent algorithm designed for generalized linear mixed models, which incorporates variable selection by L1-penalized estimation. In a final re-estimation step a model that includes only the variables corresponding to the non-zero fixed effects is fitted by simple Fisher scoring. For both the main algorithm as well as for the final re-estimation Fisher scoring two methods for the computation of the random-effects variance-covariance parameter estimates can be chosen, an EM-type estimate and an REML-type estimate.

```
Package:    glmmLasso
Type:      Package
Version:   1.6.1
Date:      2022-10-05
License:   GPL-2
LazyLoad:  yes
```

for loading a dataset type `data(nameofdataset)`

Value

Generic functions such as `print`, `predict`, `plot` and `summary` have methods to show the results of the fit. The `predict` function returns predictions on the scale of the response variable and uses also estimates of random effects for prediction, if possible (i.e. for known subjects of the grouping factor). The `plot` function plots the smooth terms, if any have been specified.

<code>call</code>	a list containing an image of the <code>glmmLasso</code> call that produced the object.
<code>coefficients</code>	a vector containing the estimated fixed effects. By default the covariates are standardized/centered within the procedure (see <code>glmmLassoControl</code>), so the coefficients are transformed back to the original scale.
<code>smooth</code>	a vector containing the estimated spline coefficients, if smooth terms have been specified.
<code>ranef</code>	a vector containing the estimated random effects.
<code>StdDev</code>	a scalar or matrix containing the estimates of the random effects standard deviation or variance-covariance parameters, respectively.
<code>fitted.values</code>	a vector of fitted values.
<code>phi</code>	estimated scale parameter, if <code>overdispersion=TRUE</code> is used. Otherwise, it is equal to one.
<code>Deltamatrix</code>	a matrix containing the estimates of fixed and random effects (columns) for each iteration (rows) of the main algorithm (i.e. before the final re-estimation step is performed, see details).
<code>Q_long</code>	a list containing the estimates of the random effects variance-covariance parameters for each iteration of the main algorithm.
<code>fixerror</code>	a vector with standard errors for the fixed effects.
<code>ranerror</code>	a vector with standard errors for the random effects.
<code>aic</code>	AIC: The negative of twice the log-likelihood plus twice the corresponding degrees of freedom. The corresponding degrees of freedom are determined by the sum of nonzero coefficients corresponding to fixed effects plus the number of random effects covariance parameters that have to be estimated.
<code>bic</code>	BIC: The negative of twice the log-likelihood plus the product of the logarithm of the overall number of observations with the corresponding degrees of freedom. The corresponding degrees of freedom are determined by the sum of nonzero coefficients corresponding to fixed effects plus the number of random effects covariance parameters that have to be estimated.
<code>conv.step</code>	number of iterations until the main algorithm has converged.

Author(s)

Andreas Groll <groll@statistik.tu-dortmund.de>

References

Groll, A. and G. Tutz (2014). Variable selection for generalized linear mixed models by L1-penalized estimation. *Statistics and Computing* 24(2), 137–154.

Goeman, J. J. (2010). L1 Penalized Estimation in the Cox Proportional Hazards Model. *Biometrical Journal* 52, 70–84.

See Also

[glmmLassoControl](#), [soccer](#), [knee](#)

Examples

```
## Not run:
data("soccer")

soccer[,c(4,5,9:16)]<-scale(soccer[,c(4,5,9:16)],center=TRUE,scale=TRUE)
soccer<-data.frame(soccer)

## linear mixed model
lm1 <- glmmLasso(points ~ transfer.spendings + ave.unfair.score
  + ball.possession + tackles
  + ave.attend + sold.out, rnd = list(team=~1),
  lambda=50, data = soccer)

summary(lm1)

## similar linear model without random effects
lm1b <- glmmLasso(points ~ transfer.spendings + ave.unfair.score
  + ball.possession + tackles
  + ave.attend + sold.out, rnd = NULL,
  lambda=50, data = soccer)

summary(lm1b)

## linear mixed model with slope on ave.attend;
## the coefficient of ave.attend is not penalized;
lm2 <- glmmLasso(points~transfer.spendings + ave.unfair.score
  + ball.possession + tackles + ave.attend
  + sold.out, rnd = list(team=~1 + ave.attend), lambda=10,
  data = soccer, control = list(index=c(1,2,3,4,NA,5),
  method="REML",print.iter=TRUE))

summary(lm2)

## linear mixed model with categorical covariates
## and final Fisher scoring re-estimation step
lm3 <- glmmLasso(points ~ transfer.spendings + as.factor(red.card)
  + as.factor(yellow.red.card) + ball.possession
  + tackles + ave.attend + sold.out, rnd = list(team=~1),
  data = soccer, lambda=100, final.re=TRUE,
```

```

        control = list(print.iter=TRUE,print.iter.final=TRUE))

summary(lm3)

## generalized linear mixed model
## with starting values
glm1 <- glmmLasso(points~transfer.spending
  + ave.unfair.score + sold.out
  + tackles + ave.attend + ball.possession, rnd = list(team=~1),
  family = poisson(link = log), data = soccer, lambda=100,
  control = list(print.iter=TRUE,start=c(1,rep(0,29)),q_start=0.7))

summary(glm1)

## generalized linear mixed model with a smooth term
glm2 <- glmmLasso(points~ + ave.unfair.score + ave.attend
  + ball.possession + tackles + sold.out,
  rnd = list(team=~1), family = poisson(link = log),
  data = soccer, lambda=100, control = list(smooth=
  list(formula=~-1 + transfer.spending, nbasis=7,
  spline.degree=3, diff.ord=2, penal=5),
  print.iter=TRUE))

summary(glm2)

plot(glm2,plot.data=FALSE)

#####
#####
#####

data(knee)

knee[,c(2,4:6)]<-scale(knee[,c(2,4:6)],center=TRUE,scale=TRUE)
knee<-data.frame(knee)

## fit cumulative model
glm3 <- glmmLasso(pain ~ time + th + age + sex, rnd = NULL,
  family = cumulative(), data = knee, lambda=10,
  control=list(print.iter=TRUE))

summary(glm3)

## fit adjacent category model
glm4 <- glmmLasso(pain ~ time + th + age + sex, rnd = NULL,
  family = acat(), data = knee, lambda=10,
  control=list(print.iter=TRUE))

summary(glm4)

# see also demo("glmmLasso-soccer")

```



```
## End(Not run)
```

```
glmmLassoControl      Control Values for glmmLasso fit
```

Description

The values supplied in the function call replace the defaults and a list with all possible arguments is returned. The returned list is used as the control argument to the `glmmLasso` function.

Usage

```
glmmLassoControl(nue=1, index=NULL, smooth=NULL, start=NULL, q_start=NULL,
                 center = TRUE, standardize = TRUE, steps=1000,
                 method="EM", overdispersion=FALSE,
                 epsilon=1e-4, maxIter=200, print.iter=FALSE,
                 print.iter.final=FALSE, method.final="EM",
                 eps.final=1e-4, Q.fac=5, complexity="hat.matrix",...)
```

Arguments

<code>nue</code>	weakness of the learner. Choose $0 < \text{nue} \leq 1$. Default is 1.
<code>index</code>	vector which defines the grouping of the variables. Components sharing the same number build a group and factor variables get a single number (and are automatically treated as a group). Non-penalized coefficients are marked with NA.
<code>smooth</code>	a list specifying the formula of the smooth terms, together with the number of basis functions <code>nbasis</code> , the degree of the B-splines <code>spline.degree</code> , the order of differences that is used for penalization <code>diff.ord</code> and finally a corresponding penalty parameter <code>penal</code> .
<code>start</code>	a vector containing starting values for fixed and random effects of suitable length. Default is a vector full of zeros.
<code>q_start</code>	a scalar or matrix of suitable dimension, specifying starting values for the random-effects variance-covariance matrix. Default is a scalar 0.1 or diagonal matrix with 0.1 in the diagonal, depending on the dimension of the random effects.
<code>center</code>	logical. If true, the columns of the design matrix will be centered (except a possible intercept column).
<code>standardize</code>	logical. If true, the design matrix will be blockwise orthonormalized such that for each block $X^T X = n1$ (*after* possible centering).
<code>steps</code>	the number of iterations. Default is 1000.
<code>method</code>	two methods for the computation of the random-effects variance-covariance parameter estimates can be chosen, an EM-type estimate and an REML-type estimate. The REML-type estimate uses the <code>n1minb</code> or the <code>bobyqa</code> function for optimization, depending on the dimension of the random effects. Default is EM.

<code>overdispersion</code>	logical scalar. If FALSE, no scale parameter is derived, if TRUE, in each iteration a scale parameter is estimated by use of Pearson residuals. This can be used e.g. to fit overdispersed Poisson models. Default is FALSE. If the Gaussian family is used, overdispersion is automatically set TRUE.
<code>epsilon</code>	controls the speed of convergence. Default is 1e-4.
<code>maxIter</code>	the number of iterations for the final Fisher scoring re-estimation procedure. Default is 200.
<code>print.iter</code>	logical. Should the number of iterations be printed? Default is FALSE.
<code>print.iter.final</code>	logical. Should the number of iterations in the final re-estimation step be printed? Default is FALSE.
<code>method.final</code>	two methods for the computation of the random-effects variance-covariance parameter estimates for the final Fisher scoring re-estimation procedure can be chosen, an EM-type estimate and an REML-type estimate. The REML-type estimate uses the <code>bobyqa</code> function for optimization. Default is EM.
<code>eps.final</code>	controls the speed of convergence in the final re-estimation. Default is 1e-4.
<code>Q.fac</code>	Factor which controls the interval on which is searched for the optimal parameters of the random-effects variance-covariance matrix, if <code>method.final="REML"</code> . Default is 5.
<code>complexity</code>	Character which determines how the model complexity is computed. Default is "hat.matrix", which sums up the trace of the corresponding hat matrix. Alternatively, simply the number of estimated (non-zero) parameters can be used by setting <code>complexity="non-zero"</code> .
<code>...</code>	Further arguments to be passed.

Value

a list with components for each of the possible arguments.

Author(s)

Andreas Groll <groll@math.lmu.de>

See Also

[glmmLasso](#), [bobyqa](#)

Examples

```
# Use REML estimates for random effects covariance parameters
# and lighten the convergence criterion
glmmLassoControl(method="REML", epsilon=1e-4)
```

knee

Clinical pain study on knee data

Description

The knee data set illustrates the effect of a medical spray on the pressure pain in the knee due to sports injuries.

Usage

```
data(knee)
```

Format

A data frame with 381 patients, each with three replicates, and the following 7 variables:

`pain` the magnitude of pressure pain in the knee given in 5 categories (1: lowest pain; 5: strongest pain).

`time` the number of replication

`id` number of patient

`th` the therapy (1: spray; 0: placebo)

`age` age of the patient in years

`sex` sex of the patient (1: male; 0: female)

`pain.start` the magnitude of pressure pain in the knee at the beginning of the study

References

Tutz, G. (2000). *Die Analyse kategorialer Daten - eine anwendungsorientierte Einfuehrung in Logit-Modellierung und kategoriale Regression*. Muenchen: Oldenbourg Verlag.

Tutz, G. and A. Groll (2012). Likelihood-based boosting in binary and ordinal random effects models. *Journal of Computational and Graphical Statistics*, **22(2)**: 356-378

See Also

[OrdinalBoost](#), [glmLasso](#).

 soccer

German Bundesliga data for the seasons 2008-2010

Description

The soccer data contains different covariables for the teams that played in the first German soccer division, the Bundesliga, in the seasons 2007/2008 until 2009/2010.

Usage

```
data(soccer)
```

Format

A data frame with 54 observations on the following 16 variables.

`pos` the final league rank of a soccer team at the end of the season

`team` soccer teams

`points` number of the points a team has earned during the season

`transfer.spending` the amount (in Euro) that a team has spent for new players at the start of the season

`transfer.receipt` the amount (in Euro) that a team has earned for the selling of players at the start of the season

`yellow.card` number of the yellow cards a team has received during the season

`yellow.red.card` number of the yellow-red cards a team has received during the season

`red.card` number of the red cards a team has received during the season

`unfair.score` unfairness score which is derived by the number of yellow cards (1 unfairness point), yellow-red cards (2 unfairness points) and red cards (3 unfairness points) a team has received during the season

`ave.unfair.score` average unfairness score per match

`ball.possession` average percentage of ball possession per match

`tackles` average percentage of head-to-head duels won per match

`capacity` capacity of the team's soccer stadium

`total.attend` total attendance of a soccer team for the whole season

`ave.attend` average attendance of a soccer team per match

`sold.out` number of stadium sold outs during a season

References

Groll, A. and G. Tutz (2012). Regularization for generalized additive mixed models by likelihood-based boosting. *Methods of Information in Medicine*. 51(2), 168-177.

Groll, A. and G. Tutz (2014). Variable selection for generalized linear mixed models by L1-penalized estimation. *Statistics and Computing* 24(2), 137-154.

We are grateful to Jasmin Abedieh for providing the German Bundesliga data, which were part of her bachelor thesis.

See Also

[glmLasso](#), [bGLMM](#), [bGAMM](#).

Index

- * **Generalized linear mixed model**

 - glmLasso, 4

- * **Lasso**

 - glmLasso, 4

- * **Shrinkage**

 - glmLasso, 4

- * **Variable selection**

 - glmLasso, 4

- * **datasets**

 - knee, 11

 - soccer, 12

- * **glmLassoControl**

 - glmLassoControl, 9

- * **ordinal regression**

 - acat, 2

 - cumulative, 3

acat, 2, 4, 5

bGAMM, 13

bGLMM, 13

bobyqa, 10

cumulative, 2, 3, 5

family, 5

glm, 5

glmLasso, 2–4, 4, 10, 11, 13

glmLassoControl, 5–7, 9

knee, 2, 4, 7, 11

OrdinalBoost, 11

soccer, 7, 12