Package 'glmmPen'

October 13, 2022

Type Package

Title High Dimensional Penalized Generalized Linear Mixed Models (pGLMM)

Version 1.5.1.10

Date 2022-04-27

Maintainer Hillary Heiling <hheiling@live.unc.edu>

Description Fits high dimensional penalized generalized linear mixed models using the Monte Carlo Expectation Conditional Minimization (MCECM) algorithm. The purpose of the package is to perform variable selection on both the fixed and random effects simultaneously for generalized linear mixed models. The package supports fitting of Binomial, Gaussian, and Poisson data with canonical links, and supports penalization using the MCP, SCAD, or LASSO penalties. The MCECM algorithm is described in Rashid et al. (2020) <doi:10.1080/01621459.2019.1671197>. The techniques used in the minimization portion of the procedure (the M-step) are derived from the procedures of the 'ncvreg' package (Breheny and Huang (2011) <doi:10.1214/10-AOAS388>) and 'grpreg' package (Breheny and Huang (2015) <doi:10.1007/s11222-013-9424-2>), with appropriate modifications to account for the estimation and penalization of the random effects. The 'ncvreg' and 'grpreg' packages also describe the MCP, SCAD, and LASSO penalties.

License GPL (>= 2)

Encoding UTF-8

Imports ggplot2, Matrix, methods, ncvreg, reshape2, rstan (>= 2.18.1), rstantools (>= 2.0.0), stringr, mvtnorm, MASS

Depends lme4, bigmemory, Rcpp (>= 0.12.0), R (>= 3.6.0)

LinkingTo BH (>= 1.66.0), bigmemory, Rcpp (>= 0.12.0), RcppArmadillo, RcppEigen (>= 0.3.3.3.0), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

RoxygenNote 7.1.2

NeedsCompilation yes

Author Hillary Heiling [aut, cre],
Naim Rashid [aut],
Quefeng Li [aut]

Suggests testthat, knitr, rmarkdown

Biarch true

SystemRequirements GNU make

Repository CRAN

Date/Publication 2022-04-29 12:30:07 UTC

R topics documented:

adaptControl 2
basal 3
fit_dat 4
glFormula_edit 9
glmm 10
glmmPen 12
glmmPen_FineSearch 16
lambdaControl 18
LambdaSeq 21
optimControl 22
pglmmObj-class 24
plot_mcmc 28
rControl 30
select_tune 31
sim.data 34

Index **37**

`adaptControl` *Control of Metropolis-within-Gibbs Adaptive Random Walk Sampling Procedure Controls the adaptive random walk Metropolis-within-Gibbs sampling procedure.*

Description

Control of Metropolis-within-Gibbs Adaptive Random Walk Sampling Procedure

Controls the adaptive random walk Metropolis-within-Gibbs sampling procedure.

Usage

```
adaptControl(batch_length = 100, offset = 0)
```

Arguments

batch_length	positive integer specifying the number of posterior samples to collect before the proposal variance is adjusted based on the acceptance rate of the last batch_length accepted posterior samples. Default is set to 100. Batch length restricted to be no less than 50.
offset	non-negative integer specifying an offset value for the increment of the proposal variance adjustment. Optionally used to ensure the required diminishing adaptation condition. Default set to 0. increment = min(0.01, 1 / sqrt(batch*batch_length + offset))

Value

Function returns a list (inheriting from class "adaptControl") containing parameter specifications for the adaptive random walk sampling procedure.

---

basal

Basal dataset: A composition of cancer datasets with top scoring pairs (TSPs) as covariates and binary response indicating if the subject's cancer subtype was basal-like. A dataset composed of four datasets combined from studies that contain gene expression data from subjects with several types of cancer. Two of these datasets contain gene expression data for subjects with Pancreatic Ductal Adenocarcinoma (PDAC), one dataset contains data for subjects with Breast Cancer, and the fourth dataset contains data for subjects with Bladder Cancer. The response of interest is whether or not the subject's cancer subtype was the basal-like subtype. See articles Rashid et al. (2020) "Modeling Between-Study Heterogeneity for Improved Replicability in Gene Signature Selection and Clinical Prediction" and Moffitt et al. (2015) "Virtual microdissection identifies distinct tumor- and stroma-specific subtypes of pancreatic ductal adenocarcinoma" for further details on these four datasets.

Description

Basal dataset: A composition of cancer datasets with top scoring pairs (TSPs) as covariates and binary response indicating if the subject's cancer subtype was basal-like.

A dataset composed of four datasets combined from studies that contain gene expression data from subjects with several types of cancer. Two of these datasets contain gene expression data for subjects with Pancreatic Ductal Adenocarcinoma (PDAC), one dataset contains data for subjects with Breast Cancer, and the fourth dataset contains data for subjects with Bladder Cancer. The response of interest is whether or not the subject's cancer subtype was the basal-like subtype. See articles Rashid et al. (2020) "Modeling Between-Study Heterogeneity for Improved Replicability in Gene Signature Selection and Clinical Prediction" and Moffitt et al. (2015) "Virtual microdissection identifies distinct tumor- and stroma-specific subtypes of pancreatic ductal adenocarcinoma" for further details on these four datasets.

Usage

```
data("basal")
```

Format

A list containing the following elements:

y binary response vector; 1 indicates that the subject's cancer was of the basal-like subtype, 0 otherwise

X matrix of 50 top scoring pair (TSP) covariates

group factor indicating which cancer study the observation belongs to, which are given the following descriptions: UNC PDAC, TCGA PDAC, TCGA Bladder Cancer, and UNC Breast Cancer

Z model matrix for random effects; organized first by variable, then by group (groups {1,2,3,4} correspond to studies UNC_PDAC, TCGA_PDAC, TCGA_Bladder, and UNC_Breast)

`fit_dat` *Fit a Penalized Generalized Mixed Model via Monte Carlo Expectation Conditional Minimization (MCECM)* `fit_dat` *is used to fit a penalized generalized mixed model via Monte Carlo Expectation Conditional Minimization (MCECM) for a single tuning parameter combinations and is called within* `glmmPen` *or* `glmm` *(cannot be called directly by user)*

Description

Fit a Penalized Generalized Mixed Model via Monte Carlo Expectation Conditional Minimization (MCECM)

`fit_dat` is used to fit a penalized generalized mixed model via Monte Carlo Expectation Conditional Minimization (MCECM) for a single tuning parameter combinations and is called within `glmmPen` or `glmm` (cannot be called directly by user)

Usage

```
fit_dat(
  dat,
  lambda0 = 0,
  lambda1 = 0,
  conv_EM = 0.001,
  conv_CD = 1e-04,
  family = "binomial",
  offset_fit = NULL,
  trace = 0,
  penalty = c("MCP", "SCAD", "lasso"),
  alpha = 1,
  gamma_penalty = switch(penalty[1], SCAD = 4, 3),
```

```
  group_X = 0:(ncol(dat$X) - 1),
  nMC_burnin = 250,
  nMC = 250,
  nMC_max = 5000,
  t = 2,
  mcc = 2,
  u_init = NULL,
  coef_old = NULL,
  ufull_describe = NULL,
  maxitEM = 50,
  maxit_CD = 250,
  M = 10^4,
  sampler = c("stan", "random_walk", "independence"),
  adapt_RW_options = adaptControl(),
  covar = c("unstructured", "independent"),
  var_start = 1,
  logLik_calc = FALSE,
  checks_complete = FALSE,
  ranef_keep = rep(1, times = (ncol(dat$Z)/nlevels(dat$group))),
  conv_type = 1,
  progress = TRUE
)
```

Arguments

dat	a list object specifying y (response vector), X (model matrix of all covariates), Z (model matrix for the random effects), and group (numeric factor vector whose value indicates the study, batch, or other group identity to which on observation belongs)
lambda0	a non-negative numeric penalty parameter for the fixed effects coefficients
lambda1	a non-negative numeric penalty parameter for the (grouped) random effects covariance coefficients
conv_EM	a non-negative numeric convergence criteria for the convergence of the EM algorithm. Default is 0.0015. EM algorithm is considered to have converge if the average Euclidean distance between the current coefficient estimates and the coefficient estimates from t EM iterations back is less than conv_EM mcc times in a row. See t and mcc for more details.
conv_CD	a non-negative numeric convergence criteria for the convergence of the grouped coordinate descent loop within the M step of the EM algorithm. Default 0.0005.
family	a description of the error distribution and link function to be used in the model. Currently, the glmmPen algorithm allows the Binomial, Gaussian, and Poisson families with canonical links only.
offset_fit	This can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases.
trace	an integer specifying print output to include as function runs. Default value is 0. See Details for more information about output provided when trace = 0, 1, or 2.

penalty	character describing the type of penalty to use in the variable selection procedure. Options include 'MCP', 'SCAD', and 'lasso'. Default is MCP penalty. If the random effect covariance matrix is "unstructured", then a group MCP, group SCAD, or group LASSO penalty is used on the random effects coefficients. See Breheny and Huang (2011) <doi:10.1214/10-AOAS388> and Breheny and Huang (2015) <doi:10.1007/s11222-013-9424-2> for details of these penalties.
alpha	Tuning parameter for the Mnet estimator which controls the relative contributions from the MCP/SCAD/LASSO penalty and the ridge, or L2, penalty. `alpha=1` is equivalent to the MCP/SCAD/LASSO penalty, while `alpha=0` is equivalent to ridge regression. However, `alpha=0` is not supported; `alpha` may be arbitrarily small, but not exactly zero
gamma_penalty	The scaling factor of the MCP and SCAD penalties. Not used by LASSO penalty. Default is 4.0 for SCAD and 3.0 for MCP. See Breheny and Huang (2011) <doi:10.1214/10-AOAS388> and Breheny and Huang (2015) <doi:10.1007/s11222-013-9424-2> for further details.
group_X	vector describing the grouping of the covariates in the model matrix.
nMC_burnin	positive integer specifying the number of posterior samples to use as burn-in for each E step in the EM algorithm. If set to `NULL`, the algorithm inputs the following defaults: Default 250 when the number of random effects predictors is less than or equal to 10; default 100 otherwise. Function will not allow `nMC_burnin` to be less than 100.
nMC	a positive integer for the initial number of Monte Carlo draws. See the `nMC_start` argument in `optimControl` for more details.
nMC_max	a positive integer for the maximum number of allowed Monte Carlo draws used in each step of the EM algorithm. If set to `NULL`, the algorithm inputs the following defaults: When the number of random effect covariates is greater than 10, the default is set to 1000; when the number of random effect covariates is 10 or less, the default is set to 2500.
t	the convergence criteria is based on the average Euclidean distance between the most recent coefficient estimates and the coefficient estimates from `t` EM iterations back. Positive integer, default equals 2.
mcc	the number of times the convergence criteria must be met before the algorithm is seen as having converged (mcc for 'meet condition counter'). Default set to 2. Value restricted to be no less than 2.
u_init	matrix giving values to initialize samples from the posterior. If Binomial or Poisson families, only need a single row to initialize samples from the posterior; if Gaussian family, multiple rows needed to initialize the estimate of the residual error (needed for the E-step). Columns correspond to the columns of the Z random effect model matrix.
coef_old	vector giving values to initialized the coefficients (both fixed and random effects)
ufull_describe	output from `bigmemory::describe` (which returns a list of the information needed to attach to a big.matrix object) applied to the big.matrix of posterior samples from the 'full' model. The big.matrix described by the object is used to calculate the BIC-ICQ value for the model.

`maxitEM` a positive integer for the maximum number of allowed EM iterations. If set to `NULL`, then the algorithm inputs the following defaults: Default equals 50 for the Binomial and Poisson families, 65 for the Gaussian family.

`maxit_CD` a positive integer for the maximum number of allowed iterations for the coordinate descent algorithms used within the M-step of each EM iteration. Default equals 50.

`M` positive integer specifying the number of posterior samples to use within the Pajor log-likelihood calculation. Default is 10^4; minimum allowed value is 5000.

`sampler` character string specifying whether the posterior samples of the random effects should be drawn using Stan (default, from package rstan) or the Metropolis-within-Gibbs procedure incorporating an adaptive random walk sampler ("random_walk") or an independence sampler ("independence"). If using the random walk sampler, see `adaptControl` for some additional control structure parameters.

`adapt_RW_options` a list of class "adaptControl" from function `adaptControl` containing the control parameters for the adaptive random walk Metropolis-within-Gibbs procedure. Ignored if `optimControl` parameter `sampler` is set to "stan" (default) or "independence".

`covar` character string specifying whether the covariance matrix should be unstructured ("unstructured") or diagonal with no covariances between variables ("independent"). Default is set to `NULL`. If `covar` is set to `NULL` and the number of random effects predictors (not including the intercept) is greater than or equal to 10 (i.e. high dimensional), then the algorithm automatically assumes an independent covariance structure and `covar` is set to "independent". Otherwise if `covar` is set to `NULL` and the number of random effects predictors is less than 10, then the algorithm automatically assumes an unstructured covariance structure and `covar` is set to "unstructured".

`var_start` either the character string "recommend" or a positive number specifying the starting values to initialize the variance of the covariance matrix. Default "recommend" first fits a simple model with a fixed and random intercept only using the lme4 package. The random intercept variance estimate from this model is then multiplied by 2 and used as the starting variance.

`logLik_calc` logical value specifying if the log likelihood (and log-likelihood based calculations BIC, BICh, and BICNgrp) should be calculated for all of the models in the selection procedure. If BIC-ICQ is used for selection, the log-likelihood is not needed for each model. However, if users are interested in comparing the best models from BIC-ICQ and other BIC-type selection criteria, setting `logLik_calc` to `TRUE` will calculate these other quantities for all of the models.

`checks_complete` logical value indicating whether the function has been called within `glmm` or `glmmPen` or whether the function has been called by itself. Used for package testing purposes (user cannot directly call `fit_dat`). If true, performs additional checks on the input data. If false, assumes data input checks have already been performed.

`ranef_keep` vector of 0s and 1s indicating which random effects should be considered as non-zero at the start of the algorithm. For each random effect, 1 indicates the random effect should be considered non-zero at start of algorithm, 0 indicates otherwise. The first element for the random intercept should always be 1.

`conv_type` integer specifying which type of convergence criteria to use. Default 1 specifies using the average Eucledian distance, and 2 specifies using relative change in the Q-function estimate. For now, all calls to `fit_dat` within the `glmmPen` framework restrict this convergence type to be the average Euclidean distance. However, we keep this argument in case we decide to allow multiple convergence type options in future versions of the package.

`progress` a logical value indicating if additional output should be given showing the progress of the fit procedure. If `TRUE`, such output includes iteration-level information for the fit procedure (iteration number EM_iter, number of MCMC samples nMC, average Euclidean distance between current coefficients and coefficients from t–defined in `optimControl`–iterations back EM_conv, and number of non-zero fixed and random effects covariates not including the intercept). Additionally, `progress = TRUE` gives some other information regarding the progress of the variable selection procedure, including the model selection criteria and log-likelihood estimates for each model fit. Default is `TRUE`.

Value

a list with the following elements:

`coef` a numeric vector of coefficients of fixed effects estimates and non-zero estimates of the lower-triangular cholesky decomposition of the random effects covariance matrix (in vector form)

`sigma` random effects covariance matrix

`lambda0, lambda1` the penalty parameters input into the function

`covgroup` Organization of how random effects coefficients are grouped.

`J` a sparse matrix that transforms the non-zero elements of the lower-triangular cholesky decomposition of the random effects covariance matrix into a vector. For unstructured covariance matrices, dimension of dimension q^2 x (q(q+1)/2) (where q = number of random effects). For independent covariance matrices, q^2 x q.

`ll` estimate of the log likelihood, calculated using the Pajor method

`BICh` the hybrid BIC estimate described in Delattre, Lavielle, and Poursat (2014)

`BIC` Regular BIC estimate

`BICNgrps` BIC estimate with N = number of groups in penalty term instead of N = number of total observations.

`BICq` BIC-ICQ estimate

`u` a matrix of the Monte Carlo draws. Organization of columns: first by random effect variable, then by group within variable (i.e. Var1:Grp1 Var1:Grp2 ... Var1:GrpK Var2:Grp1 ... Varq:GrpK)

`gibbs_accept_rate`
: a matrix of the ending gibbs acceptance rates for each variable (columns) and each group (rows) when the sampler is either "random_walk" or "independence"

`proposal_SD`
: a matrix of the ending proposal standard deviations (used in the adaptive random walk version of the Metropolis-within-Gibbs sampling) for each variable (columns) and each group (rows)

`glFormula_edit` *Extracting Useful Vectors and Matrices from Formula and Data Information*

Description

Takes the model `formula` and an optional data frame and converts them into y, X, Z, and group output.

Usage

```
glFormula_edit(
  formula,
  data = NULL,
  family,
  subset,
  weights,
  na.action,
  offset,
  ...
)
```

Arguments

`formula`
: a two-sided linear formula object describing both the fixed-effects and random-effects part of the model, with the response on the left of a ~ operator and the terms, sepearated by + operators, on the right. Random-effects terms are distinguished by vertical bars ("|") separating expression for design matrices from grouping factors. `formula` should be of the same format needed for `glmer` in package **lme4**. Only one grouping factor will be recognized. The random effects covariates need to be a subset of the fixed effects covariates. The offset must be specified outside of the formula in the 'offset' argument.

`data`
: an optional data frame containing the variables named in `formula`. Although `data` is optional, the package authors *strongly* recommend its use. If `data` is omitted, variables will be taken from the environment of `formula` (if specified as a formula).

`family`
: a description of the error distribution and link function to be used in the model (a family function or the result of a call to a family function). (See family for details of family functions.)

subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.
na.action	a function that indicates what should happen when the data contain NAs. The default option na.omit removes observations with any missing values in any of the variables
offset	this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases.
...	potential further arguments

Value

a list with the following elements:

fr	a model frame including all fixed and random covariates, the response, and the grouping variable
X	fixed effects covariates model matrix
reTrms	list containing several items relating to the random effects
family	family specified for data modeling
formula	formula
fixed_vars	vector of variable names used for fixed effects
fwmsgs	indicator for a check of the group levels

---

glmm *Fit a Generalized Mixed Model via Monte Carlo Expectation Conditional Minimization (MCECM)*

Description

glmm is used to fit a single generalized mixed model via Monte Carlo Expectation Conditional Minimization (MCECM). Unlike glmmPen, no model selection is performed.

Usage

```
glmm(
  formula,
  data = NULL,
  family = "binomial",
  covar = NULL,
  offset = NULL,
  optim_options = optimControl(),
  adapt_RW_options = adaptControl(),
```

```
  trace = 0,
  tuning_options = lambdaControl(),
  progress = TRUE,
  ...
)
```

Arguments

formula	a two-sided linear formula object describing both the fixed effects and random effects part of the model, with the response on the left of a ~ operator and the terms, separated by + operators, on the right. Random-effects terms are distinguished by vertical bars ("|") separating expression for design matrices from the grouping factor. formula should be of the same format needed for glmer in package **lme4**. Only one grouping factor will be recognized. The random effects covariates need to be a subset of the fixed effects covariates. The offset must be specified outside of the formula in the 'offset' argument.
data	an optional data frame containing the variables named in formula. If data is omitted, variables will be taken from the environment of formula.
family	a description of the error distribution and link function to be used in the model. Currently, the glmmPen algorithm allows the Binomial, Gaussian, and Poisson families with canonical links only.
covar	character string specifying whether the covariance matrix should be unstructured ("unstructured") or diagonal with no covariances between variables ("independent"). Default is set to NULL. If covar is set to NULL and the number of random effects predictors (not including the intercept) is greater than or equal to 10 (i.e. high dimensional), then the algorithm automatically assumes an independent covariance structure and covar is set to "independent". Otherwise if covar is set to NULL and the number of random effects predictors is less than 10, then the algorithm automatically assumes an unstructured covariance structure and covar is set to "unstructured".
offset	This can be used to specify an *a priori* known component to be included in the linear predictor during fitting. Default set to NULL (no offset). If the data argument is not NULL, this should be a numeric vector of length equal to the number of cases (the length of the response vector). If the data argument specifies a data.frame, the offset argument should specify the name of a column in the data.frame.
optim_options	a structure of class "optimControl" created from function optimControl that specifies several optimization parameters. See the documentation for optimControl for more details on defaults.
adapt_RW_options	a list of class "adaptControl" from function adaptControl containing the control parameters for the adaptive random walk Metropolis-within-Gibbs procedure. Ignored if optimControl parameter sampler is set to "stan" (default) or "independence".
trace	an integer specifying print output to include as function runs. Default value is 0. See Details for more information about output provided when trace = 0, 1, or 2.

`tuning_options` a list of class "selectControl" or "lambdaControl" resulting from `selectControl` or `lambdaControl` containing additional control parameters. When function `glmm` is used,the algorithm may be run using one specific set of penalty parameters `lambda0` and `lambda1` by specifying such values in `lambdaControl()`. The default for `glmm` is to run the model fit with no penalization (`lambda0 = lambda1 = 0`). When function `glmmPen` is run, `tuning_options` is specified using `selectControl()`. See the `lambdaControl` and `selectControl` documentation for further details.

`progress` a logical value indicating if additional output should be given showing the progress of the fit procedure. If `TRUE`, such output includes iteration-level information for the fit procedure (iteration number EM_iter, number of MCMC samples nMC, average Euclidean distance between current coefficients and coefficients from t–defined in `optimControl`–iterations back EM_conv, and number of non-zero fixed and random effects covariates not including the intercept). Additionally, `progress = TRUE` gives some other information regarding the progress of the variable selection procedure, including the model selection criteria and log-likelihood estimates for each model fit. Default is `TRUE`.

`...` additional arguments that could be passed into `glmmPen`. See `glmmPen` for further details.

Details

The `glmm` function can be used to fit a single generalized mixed model. While this approach is meant to be used in the case where the user knows which covariates belong in the fixed and random effects and no penalization is required, one is allowed to specify non-zero fixed and random effects penalties using `lambdaControl` and the (...) arguments. The (...) allow for specification of penalty-related arguments; see `glmmPen` for details. For a high dimensional situation, the user may want to fit a minimal penalty model using a small penalty for the fixed and random effects and save the posterior samples from this minimal penalty model for use in any BIC-ICQ calculations during selection within `glmmPen`. Specifying a file name in the 'BICq_posterior' argument will save the posterior samples from the `glmm` model into a big.matrix with this file name, see the Details section of `glmmPen` for additional details.

Value

A reference class object of class `pglmmObj` for which many methods are available (e.g. `methods(class = "pglmmObj")`)

`glmmPen` *Fit Penalized Generalized Mixed Models via Monte Carlo Expectation Conditional Minimization (MCECM)*

Description

`glmmPen` is used to fit penalized generalized mixed models via Monte Carlo Expectation Conditional Minimization (MCECM). The purpose of the function is to perform variable selection on both the

fixed and random effects simultaneously for the generalized linear mixed model. glmmPen selects the best model using BIC-type selection criteria (see selectControl documentation for further details)

Usage

```
glmmPen(
  formula,
  data = NULL,
  family = "binomial",
  covar = NULL,
  offset = NULL,
  fixef_noPen = NULL,
  penalty = c("MCP", "SCAD", "lasso"),
  alpha = 1,
  gamma_penalty = switch(penalty[1], SCAD = 4, 3),
  optim_options = optimControl(),
  adapt_RW_options = adaptControl(),
  trace = 0,
  tuning_options = selectControl(),
  BICq_posterior = NULL,
  progress = TRUE
)
```

Arguments

formula	a two-sided linear formula object describing both the fixed effects and random effects part of the model, with the response on the left of a ~ operator and the terms, separated by + operators, on the right. Random-effects terms are distinguished by vertical bars ("|") separating expression for design matrices from the grouping factor. formula should be of the same format needed for glmer in package **lme4**. Only one grouping factor will be recognized. The random effects covariates need to be a subset of the fixed effects covariates. The offset must be specified outside of the formula in the 'offset' argument.
data	an optional data frame containing the variables named in formula. If data is omitted, variables will be taken from the environment of formula.
family	a description of the error distribution and link function to be used in the model. Currently, the glmmPen algorithm allows the Binomial, Gaussian, and Poisson families with canonical links only.
covar	character string specifying whether the covariance matrix should be unstructured ("unstructured") or diagonal with no covariances between variables ("independent"). Default is set to NULL. If covar is set to NULL and the number of random effects predictors (not including the intercept) is greater than or equal to 10 (i.e. high dimensional), then the algorithm automatically assumes an independent covariance structure and covar is set to "independent". Otherwise if covar is set to NULL and the number of random effects predictors is less than 10, then the algorithm automatically assumes an unstructured covariance structure and covar is set to "unstructured".

`offset` This can be used to specify an *a priori* known component to be included in the linear predictor during fitting. Default set to `NULL` (no offset). If the data argument is not `NULL`, this should be a numeric vector of length equal to the number of cases (the length of the response vector). If the data argument specifies a data.frame, the offset argument should specify the name of a column in the data.frame.

`fixef_noPen` Optional vector of 0's and 1's of the same length as the number of fixed effects covariates used in the model. Value 0 indicates the variable should not have its fixed effect coefficient penalized, 1 indicates that it can be penalized. Order should correspond to the same order of the fixed effects given in the formula.

`penalty` character describing the type of penalty to use in the variable selection procedure. Options include 'MCP', 'SCAD', and 'lasso'. Default is MCP penalty. If the random effect covariance matrix is "unstructured", then a group MCP, group SCAD, or group LASSO penalty is used on the random effects coefficients. See Breheny and Huang (2011) <doi:10.1214/10-AOAS388> and Breheny and Huang (2015) <doi:10.1007/s11222-013-9424-2> for details of these penalties.

`alpha` Tuning parameter for the Mnet estimator which controls the relative contributions from the MCP/SCAD/LASSO penalty and the ridge, or L2, penalty. `alpha=1` is equivalent to the MCP/SCAD/LASSO penalty, while `alpha=0` is equivalent to ridge regression. However, `alpha=0` is not supported; `alpha` may be arbitrarily small, but not exactly zero

`gamma_penalty` The scaling factor of the MCP and SCAD penalties. Not used by LASSO penalty. Default is 4.0 for SCAD and 3.0 for MCP. See Breheny and Huang (2011) <doi:10.1214/10-AOAS388> and Breheny and Huang (2015) <doi:10.1007/s11222-013-9424-2> for further details.

`optim_options` a structure of class "optimControl" created from function `optimControl` that specifies several optimization parameters. See the documentation for `optimControl` for more details on defaults.

`adapt_RW_options` a list of class "adaptControl" from function `adaptControl` containing the control parameters for the adaptive random walk Metropolis-within-Gibbs procedure. Ignored if `optimControl` parameter `sampler` is set to "stan" (default) or "independence".

`trace` an integer specifying print output to include as function runs. Default value is 0. See Details for more information about output provided when trace = 0, 1, or 2.

`tuning_options` a list of class "selectControl" or "lambdaControl" resulting from `selectControl` or `lambdaControl` containing additional control parameters. When function `glmm` is used,the algorithm may be run using one specific set of penalty parameters `lambda0` and `lambda1` by specifying such values in `lambdaControl()`. The default for `glmm` is to run the model fit with no penalization (`lambda0 = lambda1 = 0`). When function `glmmPen` is run, `tuning_options` is specified using `selectControl()`. See the `lambdaControl` and `selectControl` documentation for further details.

`BICq_posterior` an optional character string expressing the path and file basename of a file combination that will file-back or currently file-backs a `big.matrix` of the posterior samples from the minimal penalty model used for the BIC-ICQ calculation used for model selection. T (BIC-ICQ reference: Ibrahim et al (2011)

<doi:10.1111/j.1541-0420.2010.01463.x>). If this argument is specified as NULL (default) and BIC-ICQ calculations are requested (see selectControl) for details), the posterior samples will be saved in the file combination 'BICq_Posterior_Draws.bin' and 'BICq_Posterior_Draws.desc' in the working directory. See 'Details' section for additional details about the required format of BICq_posterior and the file-backed big matrix.

progress — a logical value indicating if additional output should be given showing the progress of the fit procedure. If TRUE, such output includes iteration-level information for the fit procedure (iteration number EM_iter, number of MCMC samples nMC, average Euclidean distance between current coefficients and coefficients from t–defined in optimControl–iterations back EM_conv, and number of non-zero fixed and random effects covariates not including the intercept). Additionally, progress = TRUE gives some other information regarding the progress of the variable selection procedure, including the model selection criteria and log-likelihood estimates for each model fit. Default is TRUE.

Details

Argument BICq_posterior details: If the BIC_option in selectControl (tuning_options) is specified to be 'BICq', this requests the calculation of the BIC-ICQ criterion during the selection process. For the BIC-ICQ criterion to be calculated, a minimal penalty model assuming a small valued lambda penalty needs to be fit, and the posterior samples from this minimal penalty model need to be used. In order to avoid repetitive calculations of this minimal penalty model (i.e. if secondary rounds of selection are desired in glmmPen_FineSearch or if the user wants to re-run glmmPen with a different set of penalty parameters), a big.matrix of these posterior samples will be file-backed as two files: a backing file with extension '.bin' and a descriptor file with extension '.desc'. The BICq_posterior argument should contain a path and a filename of the form "./path/filename" such that the backingfile and the descriptor file would then be saved as "./path/filename.bin" and "./path/filename.desc", respectively. If BICq_posterior is set to NULL, then by default, the backingfile and descriptor file are saved in the working directory as "BICq_Posterior_Draws.bin" and "BICq_Posterior_Draws.desc". If the big matrix of posterior samples is already file-backed, BICq_posterior should specify the path and basename of the appropriate files (again of form "./path/filename"); the minimal penalty model will not be fit again and the big.matrix of posterior samples will be read using the attach.big.matrix function of the bigmemory package and used in the BIC-ICQ calculations. If the appropriate files do not exist or BICq_posterior is specified as NULL, the minimal penalty model will be fit and the minimal penalty model posterior samples will be saved as specified above. The algorithm will save 10^4 posterior samples automatically.

Trace details: The value of 0 (default) does not output any extra information. The value of 1 additionally outputs the updated coefficients, updated covariance matrix values, and the number of coordinate descent iterations used for the M step for each EM iteration. When pre-screening procedure is used and/or if the BIC-ICQ criterion is requested, trace = 1 gives additional information about the penalties used for the 'minimal penalty model' fit procedure. If Stan is not used as the E-step sampling mechanism, the value of 2 outputs all of the above plus gibbs acceptance rate information for the adaptive random walk and independence samplers and the updated proposal standard deviation for the adaptive random walk.

Value

A reference class object of class pglmmObj for which many methods are available (e.g. `methods(class = "pglmmObj")`, see ?pglmmObj for additional documentation)

`glmmPen_FineSearch` *Fit a Penalized Generalized Mixed Model via Monte Carlo Expectation Conditional Minimization (MCECM) using a finer penalty grid search* `glmmPen_FineSearch` *finds the best model from the selection results of a* `pglmmObj` *object created by* `glmmPen`*, identifies a more targeted grid search around the optimum lambda penalty values, and performs model selection on this finer grid search.*

Description

Fit a Penalized Generalized Mixed Model via Monte Carlo Expectation Conditional Minimization (MCECM) using a finer penalty grid search

`glmmPen_FineSearch` finds the best model from the selection results of a `pglmmObj` object created by `glmmPen`, identifies a more targeted grid search around the optimum lambda penalty values, and performs model selection on this finer grid search.

Usage

```
glmmPen_FineSearch(
  object,
  tuning_options = selectControl(),
  BIC_option = NULL,
  idx_range = 2,
  optim_options = NULL,
  adapt_RW_options = NULL,
  trace = 0,
  BICq_posterior = NULL,
  progress = TRUE
)
```

Arguments

`object`	an object of class `pglmmObj` created by `glmmPen`. This object must contain model selection results.
`tuning_options`	a list of class selectControl resulting from `selectControl` containing model selection control parameters. See the `selectControl` documentation for details. The user can specify their own fine grid search, or if the lambda0_seq and lambda1_seq arguments are left as `NULL`, the algorithm will automatically select a fine grid search based on the best model from the previous selection. See Details for more information. Default value set to 1.

`BIC_option` character string specifying the selection criteria used to select the 'best' model. If left as the default `NULL`, then will use the same selection criteria used during the grid search used to produce the input `pglmmObj` object. See further details in the `BIC_option` description of `selectControl`.

`idx_range` a positive integer that determines what positions within the sequence of the fixed and random effect lambda penalty parameters used in the previous coarse grid search will be used as the new fixed and random effect lambda penalty parameter ranges. See Details for more information.

`optim_options` an optional list of class "optimControl" created from function `optimControl` that specifies optimization parameters. If set to the default `NULL`, will use the optimization parameters used for the previous round of selection stored within the `pglmmObj` object.

`adapt_RW_options` an optional list of class "adaptControl" from function `adaptControl` containing the control parameters for the adaptive random walk Metropolis-within-Gibbs procedure. Ignored if `optimControl` parameter `sampler` is set to "stan" or "independence". If set to the default `NULL`, will use the adaptive random walk paraters used for the previous round of selection stored within the `pglmmObj` object.

`trace` an integer specifying print output to include as function runs. Default value is 0. See Details of `glmmPen` for more information about output provided when trace = 0, 1, or 2.

`BICq_posterior` an optional character string specifying the file-backed `big.matrix` containing the posterior draws used to calculate the BIC-ICQ selection criterion if such a `big.matrix` was created in the previous round of selection. See `glmmPen` documentation for further details.

`progress` a logical value indicating if additional output should be given showing the progress of the fit procedure. If `TRUE`, such output includes iteration-level information for the fit procedure (iteration number EM_iter, number of MCMC draws nMC, average Euclidean distance between current coefficients and coefficients from t–defined in `optimControl`–iterations back EM_conv, and number of non-zero fixed and random effects including the intercept). Additionally, `progress = TRUE` gives some other information regarding the progress of the variable selection procedure, including the model selection criteria and log-likelihood estimates for each model fit. Default is `TRUE`.

Details

The `glmmPen_FineSearch` function extracts the data, the penalty information (penalty type, gamma_penalty, and alpha), the pre-screening results from the initial variable selection procedure, and some other argument specifications from the `pglmmObj` object created during a previous round of variable/model selection. In this finer grid search, the user has the ability to make the following adjustments: the user can change the BIC option used for selection, any optimization control parameters, or any adaptive random walk parameters (if the sampler specified in the optimization parameters is "random_walk"). The user could manually specify the lambda penalty grid to search over within the `selectControl` control parameters, or the user could let the `glmmPen_FineSearch` algorithm calculate a finer grid search automatically (see next paragraph for details).

If the sequences of lambda penalty values are left unspecified in the selectControl tuning options, the glmmPen_FineSearch algorithm performs the following steps to find the finer lambda grid search: (i) The lambda combination from the best model is identified from the earlier selection results saved in the pglmmObj object. (ii) For the fixed and random effects separately, the new max and min lambda values are the lambda values idx_range positions away from the best lambda in the original lambda sequences for the fixed and random effects. For instance, suppose we consider a hypothetical lambda sequence of {0.1,0.2,0.3,0.4,0.5,0.6,0.7} for both fixed and random effects, and the best model was given by the (0.4,0.5) combination. If the idx_lambda = 2, then the fine search would use the fixed effects sequence would have (min,max) = (0.2,0.6) and the fixed effects sequence would have (min,max) = (0.3,0.7).

Value

A reference class object of class pglmmObj for which many methods are available (e.g. methods(class = "pglmmObj"))

---

lambdaControl *Control of Penalization Parameters and Selection Criteria*

Description

Constructs control structures for penalized mixed model fitting.

Usage

```
lambdaControl(lambda0 = 0, lambda1 = 0)

selectControl(
  lambda0_seq = NULL,
  lambda1_seq = NULL,
  nlambda = 10,
  search = c("abbrev", "full_grid"),
  BIC_option = c("BICq", "BICh", "BIC", "BICNgrp"),
  logLik_calc = switch(BIC_option[1], BICq = FALSE, TRUE),
  lambda.min = NULL,
  pre_screen = TRUE,
  lambda.min.presc = NULL
)
```

Arguments

lambda0	a non-negative numeric penalty parameter for the fixed effects coefficients
lambda1	a non-negative numeric penalty parameter for the (grouped) random effects covariance coefficients

`lambda0_seq, lambda1_seq`
: a sequence of non-negative numeric penalty parameters for the fixed and random effect coefficients (`lambda0_seq` and `lambda1_seq`, respectively). If `NULL`, then a sequence will be automatically calculated. See 'Details' section for more details on these default calculations.

`nlambda`
: positive integer specifying number of penalty parameters to use for the fixed and random effects penalty parameters. Default set to 10. Only used if either `lambda0_seq` or `lambda1_seq` remain unspecified by the user (one or both of these arguments set to `NULL`) and, consequently, one or more default sequences need to be calculated.

`search`
: character string of "abbrev" (default) or "full_grid" indicating if the search of models over the penalty parameter space should be the full grid search (total number of models equals 'nlambda'^2 or length('lambda0_seq')*length('lambda1_seq')) or an abbreviated grid search. The abbreviated grid search is described in more detail in the Details section. Te authors highly recommend the abbreviated grid search.

`BIC_option`
: character string specifying the selection criteria used to select the 'best' model. Default "BICq" option specifies the BIC-ICQ criterion (Ibrahim et al (2011) <doi:10.1111/j.1541-0420.2010.01463.x>), which requires a fit of a 'minimum penalty' model; a small penalty (the minimum of the penalty sequence) is used for the fixed and random effects. See "Details" section for what these small penalties will be. The "BICh" option utilizes the hybrid BIC value described in Delattre, Lavielle, and Poursat (2014) <doi:10.1214/14-EJS890>. The regular "BIC" option penalty term uses (total non-zero coefficients)*(length(y) = total number observations). The "BICNgrp" option penalty term uses (total non-zero coefficients)*(nlevels(group) = number groups).

`logLik_calc`
: logical value specifying if the log likelihood (and log-likelihood based calculations BIC, BICh, and BICNgrp) should be calculated for all of the models in the selection procedure. If BIC-ICQ is used for selection, the log-likelihood is not needed for each model. However, if users are interested in comparing the best models from BIC-ICQ and other BIC-type selection criteria, setting `logLik_calc` to `TRUE` will calculate these other quantities for all of the models.

`lambda.min`
: numeric fraction between 0 and 1. The sequence of the lambda penalty parameters ranges from the maximum lambda where all fixed and random effects are penalized to 0 and a minimum lambda value, which equals a small fraction of the maximum lambda. The parameter `lambda.min` specifies this fraction. Default value is set to `NULL`, which automatically selects `lambda.min` to equal 0.01 when the number of observations is greater than the number of fixed effects predictors and 0.05 otherwise. Only used if either `lambda0_seq` or `lambda1_seq` remain unspecified by the user (one or both of these sequence arguments set to `NULL`) and, consequently, one or more default sequences need to be calculated.

`pre_screen`
: logical value indicating whether pre-screening should be performed before model selection (default `TRUE`). If the number of random effects covariates considered is 4 or less, then no pre-screening will be performed. Pre-screening removes random effects from consideration during the model selection process, which can significantly speed up the algorithm. See "Details" section for a further discussion.

`lambda.min.presc`

numeric fraction between 0 and 1. During pre-screening and the minimal penalty model fit for the BIC-ICQ calculation, the small penalty used on the random effect is the fraction `lambda.min.presc` multiplied by the maximum penalty parameter that penalizes all fixed and random effects to 0. If left as `NULL`, the default value is 0.01 when the number of random effect covariates is 10 or less and 0.05 otherwise. Only used if `lambda1_seq` remains unspecified by the user (this argument set to `NULL` so the random effects penalty parameter sequence needs to be automatically calculated) AND either the pre-screening procedure is selected by the argument `pre_screen` or the BIC-ICQ is selected as the model selection criteria, i.e., `BIC_option` = "BICq". See the "Details" section for a further discussion.

Details

If left as the default `NULL` values, the `lambda0_seq` and `lambda1_seq` numeric sequences are automatically calculated. The sequence will be calculated in the same manner as `ncvreg` calculates the range: the max value (let's denote this as `lambda_max`) penalizes all fixed and random effects to 0, the min value is a small portion of max (`lambda.min*lambda_max`), and the sequence is composed of `nlambda` values ranging from these min and max values spread evenly on the log scale. Unlike `ncvreg`, the order of penalty values used in the algorithm must run from the min lambda to the max lambda (as opposed to running from max lambda to min lambda). The length of the sequence is specified by `nlambda`. By default, these sequences are calculated using `LambdaSeq`.

The `lambda0` and `lambda1` arguments used within the `glmm` function allow for a user to fit a model with a single non-zero penalty parameter combination. However, this is generally not recommended.

Abbreviated grid search: The abbreviated grid search proceeds in two stages. In stage 1, the algorithm fits the following series of models: the fixed effects penalty parameter remains a fixed value evaluated at the minimum of the fixed effects penalty parameters, and all random effects penalty parameters are examined. The 'best' model from this first stage of models determines the optimum random effect penalty parameter. In stage 2, the algorithm fits the following series of models: the random effects penalty parameter remains fixed at the value of the optimum random effect penalty parameter (from stage 1) and all fixed effects penalty parameters are considered. The best overall model is the best model from stage 2. This reduces the number of models considered to length(‘lambda0_seq‘) + length(‘lambda1_seq‘). The authors found that this abbreviated grid search worked well in simulations, and performed considerably faster than the full grid search that examined all possible fixed and random effect penalty parameter combinations.

The arguments `nlambda` and `lambda.min` are only used if one or both of the `lambda0_seq` and `lambda1_seq` penalty sequences (corresponding to the fixed and random effects penalty sequences, respectively) remain unspecified by the user (one or both of these arguments left as `NULL`), indicating that the algorithm needs to calculate default penalty sequences.

The argument `lambda.min.presc` is only used under the following condition: `lambda1_seq` remains unspecified by the user (this argument set to `NULL` so the random effects penalty parameter sequence needs to be calculated) AND either the pre-screening procedure is selected by the argument `pre_screen` or the BIC-ICQ is selected as the model selection criteria, i.e., `BIC_option` = "BICq". If `lambda1_seq` is specified by the user, the minimum value in that sequence will be used as the random effect penalty in the pre-screening procedure and/or the minimal penalty model for the BIC-ICQ calculation.

BIC-ICQ calculation: This model selection criteria requires the fitting of a 'minimal penalty' model, which fits a model with a small penalty on the fixed and random effects. For the fixed effects penalty, the minimal penalty is: (a) 0 if the number of fixed effects covariates is 4 or less or (b) the minimum fixed effect penalty from the fixed effects penalty sequence (either from the default sequence or from the sequence specified by the user). For the random effects penalty, the minimal penalty is (a) 0 if the number of random effects covariates is 4 or less; (b) the minimum random effect penalty from the random effects penalty sequence specified by the user, or (c) `lambda.min.presc` multiplied to the `lambda_max` maximum penalty specified above when a default random effects penalty sequence is calculated.

Pre-screening: The minimum fixed effects penalty used in the pre-screening stage will be the minimum penalty of the fixed effects penalty sequence, `lambda0_seq`. The minimum random effects penalty used in the pre-screening stage will be either (a) the minimum random effects penalty in the sequence `lambda1_seq` if this sequence specified by the user, or (b) `lambda.min.pres` x `lambda_max`, where `lambda_max` was described above.

Value

The *Control functions return a list (inheriting from class "`pglmmControl`") containing parameter values that determine settings for variable selection.

`LambdaSeq` *Calculation of Penalty Parameter Sequence (Lambda Sequence)*

Description

Calculates the sequence of penalty parameters used in the model selection procedure. This function calls functions from package `ncvreg`.

Usage

```
LambdaSeq(
  X,
  y,
  family,
  alpha = 1,
  lambda.min = NULL,
  nlambda = 10,
  penalty.factor = NULL
)
```

Arguments

`X`	matrix of standardized fixed effects (see `std` function in `ncvreg` documenation). X should not include intercept.
`y`	numeric vector of response values

family	a description of the error distribution and link function to be used in the model. Currently, the glmmPen algorithm allows the Binomial, Gaussian, and Poisson families with canonical links only.
alpha	Tuning parameter for the Mnet estimator which controls the relative contributions from the MCP/SCAD/LASSO penalty and the ridge, or L2, penalty. alpha=1 is equivalent to the MCP/SCAD/LASSO penalty, while alpha=0 is equivalent to ridge regression. However, alpha=0 is not supported; alpha may be arbitrarily small, but not exactly zero
lambda.min	numeric fraction between 0 and 1. The sequence of the lambda penalty parameters ranges from the maximum lambda where all fixed and random effects are penalized to 0 and a minimum lambda value, which equals a small fraction of the maximum lambda. The parameter lambda.min specifies this fraction. Default value is set to NULL, which automatically selects lambda.min to equal 0.01 when the number of observations is greater than the number of fixed effects predictors and 0.05 otherwise. Only used if either lambda0_seq or lambda1_seq remain unspecified by the user (one or both of these sequence arguments set to NULL) and, consequently, one or more default sequences need to be calculated.
nlambda	positive integer specifying number of penalty parameters (lambda) with which to fit a model.
penalty.factor	an optional numeric vector equal to the fixef_noPen argument in glmmPen

Value

numeric sequence of penalty parameters of length nlambda ranging from the minimum penalty parameter (first element) equal to fraction lambda.min multiplied by the maximum penalty parameter to the maximum penalty parameter (last element)

---

optimControl *Control of Penalized Generalized Linear Mixed Model Fitting*

Description

Constructs the control structure for the optimization of the penalized mixed model fit algorithm.

Usage

```
optimControl(
  conv_EM = 0.0015,
  conv_CD = 5e-04,
  nMC_burnin = NULL,
  nMC_start = NULL,
  nMC_max = NULL,
  nMC_report = 5000,
  maxitEM = NULL,
  maxit_CD = 50,
  M = 10000,
```

```
  t = 2,
  mcc = 2,
  sampler = c("stan", "random_walk", "independence"),
  var_start = "recommend"
)
```

Arguments

conv_EM	a non-negative numeric convergence criteria for the convergence of the EM algorithm. Default is 0.0015. EM algorithm is considered to have converge if the average Euclidean distance between the current coefficient estimates and the coefficient estimates from t EM iterations back is less than conv_EM mcc times in a row. See t and mcc for more details.
conv_CD	a non-negative numeric convergence criteria for the convergence of the grouped coordinate descent loop within the M step of the EM algorithm. Default 0.0005.
nMC_burnin	positive integer specifying the number of posterior samples to use as burn-in for each E step in the EM algorithm. If set to NULL, the algorithm inputs the following defaults: Default 250 when the number of random effects predictors is less than or equal to 10; default 100 otherwise. Function will not allow nMC_burnin to be less than 100.
nMC_start	a positive integer for the initial number of Monte Carlo draws. If set to NULL, the algorithm inputs the following defaults: Default 250 when the number of random effects predictors is less than or equal to 10; default 100 otherwise.
nMC_max	a positive integer for the maximum number of allowed Monte Carlo draws used in each step of the EM algorithm. If set to NULL, the algorithm inputs the following defaults: When the number of random effect covariates is greater than 10, the default is set to 1000; when the number of random effect covariates is 10 or less, the default is set to 2500.
nMC_report	a positive integer for the number of posterior samples to save from the final model. These posterior samples can be used for diagnostic purposes, see plot_mcmc. Default set to 5000.
maxitEM	a positive integer for the maximum number of allowed EM iterations. If set to NULL, then the algorithm inputs the following defaults: Default equals 50 for the Binomial and Poisson families, 65 for the Gaussian family.
maxit_CD	a positive integer for the maximum number of allowed iterations for the coordinate descent algorithms used within the M-step of each EM iteration. Default equals 50.
M	positive integer specifying the number of posterior samples to use within the Pajor log-likelihood calculation. Default is 10^4; minimum allowed value is 5000.
t	the convergence criteria is based on the average Euclidean distance between the most recent coefficient estimates and the coefficient estimates from t EM iterations back. Positive integer, default equals 2.
mcc	the number of times the convergence criteria must be met before the algorithm is seen as having converged (mcc for 'meet condition counter'). Default set to 2. Value restricted to be no less than 2.

`sampler` character string specifying whether the posterior samples of the random effects should be drawn using Stan (default, from package rstan) or the Metropolis-within-Gibbs procedure incorporating an adaptive random walk sampler ("random_walk") or an independence sampler ("independence"). If using the random walk sampler, see `adaptControl` for some additional control structure parameters.

`var_start` either the character string "recommend" or a positive number specifying the starting values to initialize the variance of the covariance matrix. Default "recommend" first fits a simple model with a fixed and random intercept only using the lme4 package. The random intercept variance estimate from this model is then multiplied by 2 and used as the starting variance.

Details

Several arguments are set to a default value of `NULL`. If these arguments are left as `NULL` by the user, then these values will be filled in with appropriate default values as specified above, which may depend on the number of random effects or the family of the data. If the user specifies particular values for these arguments, no additional modifications to these arguments will be done within the algorithm.

Value

Function returns a list inheriting from class `optimControl` containing fit and optimization criteria values used in optimization routine.

---

pglmmObj-class *Class* `pglmmObj` *of Fitted Penalized Generalized Mixed-Effects Models for package* `glmmPen`

Description

The functions `glmm`, `glmmPen`, and `glmmPen_FineSearch` from the package `glmmPen` output the reference class object of type `pglmmObj`.

Usage

```
## S3 method for class 'pglmmObj'
fixef(object)

## S3 method for class 'pglmmObj'
ranef(object)

## S3 method for class 'pglmmObj'
sigma(object, ...)

## S3 method for class 'pglmmObj'
coef(object, ...)
```

```
## S3 method for class 'pglmmObj'
family(object, ...)

## S3 method for class 'pglmmObj'
nobs(object, ...)

## S3 method for class 'pglmmObj'
ngrps(object, ...)

## S3 method for class 'pglmmObj'
formula(x, fixed.only = FALSE, random.only = FALSE, ...)

## S3 method for class 'pglmmObj'
model.frame(formula, fixed.only = FALSE, ...)

## S3 method for class 'pglmmObj'
model.matrix(object, type = c("fixed", "random"), ...)

## S3 method for class 'pglmmObj'
fitted(object, fixed.only = TRUE, ...)

## S3 method for class 'pglmmObj'
predict(
  object,
  newdata = NULL,
  type = c("link", "response"),
  fixed.only = TRUE,
  ...
)

## S3 method for class 'pglmmObj'
residuals(object, type = c("deviance", "pearson", "response", "working"), ...)

## S3 method for class 'pglmmObj'
print(x, digits = c(fef = 4, ref = 4), ...)

## S3 method for class 'pglmmObj'
summary(
  object,
  digits = c(fef = 4, ref = 4),
  resid_type = switch(object$family$family, gaussian = "pearson", "deviance"),
  ...
)

## S3 method for class 'pglmmObj'
logLik(object, ...)
```

```
## S3 method for class 'pglmmObj'
BIC(object, ...)

## S3 method for class 'pglmmObj'
plot(x, fixed.only = FALSE, type = NULL, ...)
```

Arguments

object	pglmmObj object output from glmm, glmmPen, or glmmPen_FineSearch
...	potentially further arguments passed from other methods
x	an R object of class pglmmObj
fixed.only	logical value; default TRUE indicates that only the fixed effects should be used in the fitted value/prediction, while FALSE indicates that both the fixed and random effects posterior modes should be used in the fitted value/prediction
random.only	logical value used in formula; TRUE indicates that only the formula elements relating to the random effects should be returned
formula	in the case of model.frame, a pglmmObj object
type	See details of type options for each function under "Functions" section.
newdata	optional new data.frame containing the same variables used in the model fit procedure
digits	number of significant digits for printing; default of 4
resid_type	type of residuals to summarize in output. See predict.pglmmObj for residual options available.

Value

The pglmmObj object returns the following items:

fixef	vector of fixed effects coefficients
ranef	matrix of random effects coefficients for each explanatory variable for each level of the grouping factor
sigma	random effects covariance matrix
scale	if family is Gaussian, returns the residual error variance
posterior_samples	Samples from the posterior distribution of the random effects, taken at the end of the model fit (after convergence or after maximum iterations allowed). Can be used for diagnositics purposes. Note: These posterior samples are from a single chain.
sampling	character string for type of sampling used to calculate the posterior samples in the E-step of the algorithm
results_all	matrix of results from all model fits during variable selection (if selection performed). Output for each model includes: penalty parameters for fixed (lambda0) and random (lambda1) effects, BIC-derived quantities and the log-likelihood (note: the arguments BIC_option and logLik_calc in selectControl determine which of these quantities are calculated for each model), the number of

	non-zero fixed and random effects (includes intercept), number of EM iterations used for model fit, whether or not the model converged (0 for no vs 1 for yes), and the fixed and random effects coefficients
`results_optim`	results from the 'best' model fit; see results_all for details. BICh, BIC, BICN-grp, and LogLik computed for this best model if not previously calculated.
`family`	Family
`penalty_info`	list of penalty information
`call`	arguments plugged into `glmm`, `glmmPen`, or `glmmPen_FineSearch`
`formula`	formula
`fixed_vars`	names of fixed effects variables
`data`	list of data used in model fit, including the response y, the fixed effects covariates matrix X, the random effects model matrix Z (which is composed of values from the standardized fixed effects model matrix), the grouping factor, offset, model frame, and standarization information used to standardize the fixed effects covariates
`optinfo`	Information about the optimization of the 'best' model
`control_info`	optimization parameters used for the model fit
`Estep_init`	materials that can be used to initialize another E-step (for use in `glmmPen_FineSearch`)
`Gibbs_info`	list of materials to perform diagnositics on the Metropolis-within-Gibbs sample chains, including the Gibbs acceptance rates (included for both the independence and adaptive random walk samplers) and the final proposal standard deviations (included for the adaptive random walk sampler only))
`r_estimation`	list of output related to estimation of number of latent common factors, r. Only relevant for the output of functions `glmm_FA` and `glmmPen_FA`, which are currently in development and are not yet ready for general use.

showClass("pglmmObj") methods(class = "pglmmObj")

Functions

- `fixef.pglmmObj`: Provides the fixed effects coefficients
- `ranef.pglmmObj`: Provides the random effects posterior modes for each explanatory variable for each level of the grouping factor
- `sigma.pglmmObj`: Provides the random effect covariance matrix. If family is Gaussian, also returns the standard deviation of the residual error.
- `coef.pglmmObj`: Computes the sum of the fixed effects coefficients and the random effect posterior modes for each explanatory variable for each level of each grouping factor.
- `family.pglmmObj`: Family of the fitted GLMM
- `nobs.pglmmObj`: Number of observations used in the model fit
- `ngrps.pglmmObj`: Number of levels in the grouping factor
- `formula.pglmmObj`: Formula used for the model fit. Can return the full formula, or just the formula elements relating to the fixed effects (fixed.only = TRUE) or random effects (random.only = TRUE)

- `model.frame.pglmmObj`: Returns the model frame
- `model.matrix.pglmmObj`: Returns the model matrix of either the fixed (type = "fixed") or random effects (type = "random")
- `fitted.pglmmObj`: Fitted values, i.e., the linear predictor of the model.
- `predict.pglmmObj`: Predictions for the model corresponding to the pglmmObj output object from the glmmPen package functions. The function `predict` can predict either the linear predictor of the model or the expected mean of the response, as specified by the `type` argument. Argument `type`: character string for type of predictors: "link" (default), which generates the linear predictor, and "response", which generates the expected mean values of the response.
- `residuals.pglmmObj`: Residuals for the pglmmObj output object from the glmmPen package functions. Argument `type`: character string for type of residuals to report. Options include "deviance" (default), "pearson", "response", and "working", which specify the deviance residuals, Pearson residuals, the difference between the actual response y and the expected mean response (y - mu), and the working residuals (y - mu) / mu
- `print.pglmmObj`: Prints a selection of summary information of fitted model
- `summary.pglmmObj`: Returns a list of summary statistics of the fitted model.
- `logLik.pglmmObj`: Returns the log-likelihood using the Corrected Arithmetic Mean estimator with importance sampling weights developed by Pajor (2017). Degrees of freedom give the sum of the non-zero fixed and random effects coefficients. Citation: Pajor, A. (2017). Estimating the marginal likelihood using the arithmetic mean identity. Bayesian Analysis, 12(1), 261-287.
- `BIC.pglmmObj`: Returns BIC, BICh (hybrid BIC developed by Delattre et al., citation: Delattre, M., Lavielle, M., & Poursat, M. A. (2014). A note on BIC in mixed-effects models. Electronic journal of statistics, 8(1), 456-475.), BICNgrps (BIC using N = number of groups in the penalty term), and possibly BIC-ICQ (labeled as "BICq") if the argument `BIC_option` was set to "BICq" in `selectControl` (citation for BIC-ICQ: Ibrahim, J. G., Zhu, H., Garcia, R. I., & Guo, R. (2011). Fixed and random effects selection in mixed effects models. Biometrics, 67(2), 495-503.)
- `plot.pglmmObj`: Plot residuals for the pglmmObj output object from the glmmPen package. Argument `type`: character string for type of residuals to report. Options include "deviance" (default for non-Gaussian family), "pearson" (default for Gaussian family), "response", and "working", which specify the deviance residuals, Pearson residuals, the difference between the actual response y and the expected mean response (y - mu), and the working residuals (y - mu) / mu

---

`plot_mcmc` *Plot Diagnostics for MCMC Posterior Draws of the Random Effects*

Description

Provides graphical diagnostics of the random effect posterior draws from the (best) model. Availabile diagnostics include the sample path, histograms, cummulative sums, and autocorrelation.

Usage

```
plot_mcmc(
  object,
  plots = "sample.path",
  grps = "all",
  vars = "all",
  numeric_grp_order = FALSE,
  bin_width = NULL
)
```

Arguments

object	an object of class pglmmObj output from either glmmPen or glmmPen_FineSearch.
plots	a character string or a vector of character strings specifying which graphical diagnostics to provide. Options include a sample path plot (default, "sample.path"), autocorrelation plots ("autocorr"), histograms ("histogram"), cumulative sum plots ("cumsum"), and all four possible plot options ("all"). While the "all" option will produce all four possible plots, subsets of the types of plots (e.g. sample path plots and autocorrelation plots only) can be specified with a vector of the relevant character strings (e.g. c("sample.path","autocorr"))
grps	a character string or a vector of character strings specifying which groups should have diagnostics provided. The names of the groups match the input group factor levels. Default is set to 'all' for all groups.
vars	a character string or a vector of character strings specifying which variables should have diagnostics provided. Default is set to 'all', which picks all variables with non-zero random effects. Tip: can find the names of the random effect variables in the output sigma matrix found in the pglmmObj object, run sigma(object).
numeric_grp_order	if TRUE, specifies that the groups factor should be converted to numeric values. This option could be used to ensure that the organization of the groups is in the proper numeric order (e.g. groups with levels 1-10 are ordered 1-10, not 1, 10, 2-9).
bin_width	optional binwidth argument for geom_histogram from the ggplot2 package. Default set to NULL, which specifies the default geom_histogram binwidth. This argument only applies if the "histogram" plot type is selected.

Value

a list of ggplot graphics, each faceted by group and random effect variable. Type of plots specified in the plots argument.

`rControl` *Control of Latent Factor Model Number Estimation Constructs the control structure for the estimation of the number of latent factors (r) for use within the* `glmmPen_FA` *and* `glmm_FA` *estimation procedures.*

Description

Control of Latent Factor Model Number Estimation

Constructs the control structure for the estimation of the number of latent factors (r) for use within the `glmmPen_FA` and `glmm_FA` estimation procedures.

Usage

```
rControl(r = NULL, r_max = NULL, r_est_method = "GR", size = 25)
```

Arguments

`r`	positive integer specifying number of latent common factors to assume in the model. If `NULL` (default), this value estimated from the data. See `r_est_method` for available estimation procedures, and the Details section for further details on the general estimation procedure.
`r_max`	positive integer specifying maximum number of latent factors to consider. If `NULL` (default), this value is automatically calculated.
`r_est_method`	character string indicating method used to estimate number of latent factors `r`. Default "GR" uses the Growth Ratio method of Ahn and Horenstein (2013). Other available options include "ER" for the Eigenvalue Ratio method of Ahn and Horenstein (2013) and "BN1" or "BN2", the Bai and Ng (2002) method using one of two penalties: (1) `(d + p) / (d p) log(d p/(d+p))`, (2) `(d + p) / (d p) log(min(d,p))` where d is the number of groups in the data and p is the number of total random effect covariates (including the intercept)
`size`	positive integer specifying the total number of pseudo random effect estimates to use in the estimation procedure for the number of latent factors r, which is restricted to be no less than 25. If this `size` is greater than the number of groups in the data (i.e.~the number of levels of the grouping variable), then a sampling procedure is used to increase the number of pseudo estimates to the value of `size`.

Details

TODO: additional details on estimation procedure.

select_tune	*Fit a Sequence of Penalized Generalized Mixed Model via Monte Carlo Expectation Conditional Minimization (MCECM)* select_tune *is used to fit a sequence of penalized generalized mixed models via Monte Carlo Expectation Conditional Minimization (MCECM) for multiple tuning parameter combinations and is called within* glmmPen *(cannot be called directly by user)*

Description

Fit a Sequence of Penalized Generalized Mixed Model via Monte Carlo Expectation Conditional Minimization (MCECM)

select_tune is used to fit a sequence of penalized generalized mixed models via Monte Carlo Expectation Conditional Minimization (MCECM) for multiple tuning parameter combinations and is called within glmmPen (cannot be called directly by user)

Usage

```
select_tune(
  dat,
  offset = NULL,
  family,
  covar = c("unstructured", "independent"),
  group_X = 0:(ncol(dat$X) - 1),
  penalty,
  lambda0_seq,
  lambda1_seq,
  alpha = 1,
  gamma_penalty = switch(penalty[1], SCAD = 4, 3),
  trace = 0,
  u_init = NULL,
  coef_old = NULL,
  adapt_RW_options = adaptControl(),
  optim_options = optimControl(),
  BIC_option = c("BICq", "BICh", "BIC", "BICNgrp"),
  BICq_calc = TRUE,
  logLik_calc = switch(BIC_option[1], BICq = FALSE, TRUE),
  BICq_posterior = NULL,
  checks_complete = FALSE,
  pre_screen = TRUE,
  ranef_keep = NULL,
  lambda.min.full,
  stage1 = FALSE,
  progress = TRUE
)
```

Arguments

dat	a list object specifying y (response vector), X (model matrix of all covariates), Z (model matrix for the random effects), and group (numeric factor vector whose value indicates the study, batch, or other group identity to which on observation belongs)
offset	This can be used to specify an *a priori* known component to be included in the linear predictor during fitting. Default set to NULL (no offset). If the data argument is not NULL, this should be a numeric vector of length equal to the number of cases (the length of the response vector). If the data argument specifies a data.frame, the offset argument should specify the name of a column in the data.frame.
family	a description of the error distribution and link function to be used in the model. Currently, the glmmPen algorithm allows the Binomial, Gaussian, and Poisson families with canonical links only.
covar	character string specifying whether the covariance matrix should be unstructured ("unstructured") or diagonal with no covariances between variables ("independent"). Default is set to NULL. If covar is set to NULL and the number of random effects predictors (not including the intercept) is greater than or equal to 10 (i.e. high dimensional), then the algorithm automatically assumes an independent covariance structure and covar is set to "independent". Otherwise if covar is set to NULL and the number of random effects predictors is less than 10, then the algorithm automatically assumes an unstructured covariance structure and covar is set to "unstructured".
group_X	vector describing the grouping of the covariates in the model matrix.
penalty	character describing the type of penalty to use in the variable selection procedure. Options include 'MCP', 'SCAD', and 'lasso'. Default is MCP penalty. If the random effect covariance matrix is "unstructured", then a group MCP, group SCAD, or group LASSO penalty is used on the random effects coefficients. See Breheny and Huang (2011) <doi:10.1214/10-AOAS388> and Breheny and Huang (2015) <doi:10.1007/s11222-013-9424-2> for details of these penalties.
lambda0_seq	a sequence of non-negative numeric penalty parameters for the fixed and random effect coefficients (lambda0_seq and lambda1_seq, respectively). If NULL, then a sequence will be automatically calculated. See 'Details' section for more details on these default calculations.
lambda1_seq	a sequence of non-negative numeric penalty parameters for the fixed and random effect coefficients (lambda0_seq and lambda1_seq, respectively). If NULL, then a sequence will be automatically calculated. See 'Details' section for more details on these default calculations.
alpha	Tuning parameter for the Mnet estimator which controls the relative contributions from the MCP/SCAD/LASSO penalty and the ridge, or L2, penalty. alpha=1 is equivalent to the MCP/SCAD/LASSO penalty, while alpha=0 is equivalent to ridge regression. However, alpha=0 is not supported; alpha may be arbitrarily small, but not exactly zero
gamma_penalty	The scaling factor of the MCP and SCAD penalties. Not used by LASSO penalty. Default is 4.0 for SCAD and 3.0 for MCP. See Breheny and Huang

(2011) <doi:10.1214/10-AOAS388> and Breheny and Huang (2015) <doi:10.1007/s11222-013-9424-2> for further details.

`trace` an integer specifying print output to include as function runs. Default value is 0. See Details for more information about output provided when trace = 0, 1, or 2.

`u_init` matrix giving values to initialize samples from the posterior. If Binomial or Poisson families, only need a single row to initialize samples from the posterior; if Gaussian family, multiple rows needed to initialize the estimate of the residual error (needed for the E-step). Columns correspond to the columns of the Z random effect model matrix.

`coef_old` vector giving values to initialized the coefficients (both fixed and random effects)

`adapt_RW_options` a list of class "adaptControl" from function `adaptControl` containing the control parameters for the adaptive random walk Metropolis-within-Gibbs procedure. Ignored if `optimControl` parameter `sampler` is set to "stan" (default) or "independence".

`optim_options` a structure of class "optimControl" created from function `optimControl` that specifies several optimization parameters. See the documentation for `optimControl` for more details on defaults.

`BIC_option` character string specifying the selection criteria used to select the 'best' model. Default "BICq" option specifies the BIC-ICQ criterion (Ibrahim et al (2011) <doi:10.1111/j.1541-0420.2010.01463.x>), which requires a fit of a 'minimum penalty' model; a small penalty (the minimum of the penalty sequence) is used for the fixed and random effects. See "Details" section for what these small penalties will be. The "BICh" option utilizes the hybrid BIC value described in Delattre, Lavielle, and Poursat (2014) <doi:10.1214/14-EJS890>. The regular "BIC" option penalty term uses (total non-zero coefficients)*(length(y) = total number observations). The "BICNgrp" option penalty term uses (total non-zero coefficients)*(nlevels(group) = number groups).

`BICq_calc` logical value indicating if the BIC-ICQ criterion should be used to select the best model.

`logLik_calc` logical value specifying if the log likelihood (and log-likelihood based calculations BIC, BICh, and BICNgrp) should be calculated for all of the models in the selection procedure. If BIC-ICQ is used for selection, the log-likelihood is not needed for each model. However, if users are interested in comparing the best models from BIC-ICQ and other BIC-type selection criteria, setting `logLik_calc` to `TRUE` will calculate these other quantities for all of the models.

`BICq_posterior` an optional character string expressing the path and file basename of a file combination that will file-back or currently file-backs a `big.matrix` of the posterior samples from the minimal penalty model used for the BIC-ICQ calculation used for model selection. T (BIC-ICQ reference: Ibrahim et al (2011) <doi:10.1111/j.1541-0420.2010.01463.x>). If this argument is specified as `NULL` (default) and BIC-ICQ calculations are requested (see `selectControl`) for details), the posterior samples will be saved in the file combination 'BICq_Posterior_Draws.bin' and 'BICq_Posterior_Draws.desc' in the working directory. See 'Details' section for additional details about the required format of `BICq_posterior` and the file-backed big matrix.

checks_complete
: logical value indicating if several data checks have been completed.

pre_screen
: logical value indicating whether pre-screening should be performed before model selection (default TRUE). If the number of random effects covariates considered is 4 or less, then no pre-screening will be performed. Pre-screening removes random effects from consideration during the model selection process, which can significantly speed up the algorithm. See "Details" section for a further discussion.

ranef_keep
: vector of 0s and 1s indicating which random effects should be considered as non-zero at the start of the algorithm. For each random effect, 1 indicates the random effect should be considered non-zero at start of algorithm, 0 indicates otherwise. The first element for the random intercept should always be 1.

lambda.min.full
: a vector of two numeric values that gives the fixed and random effect penalty values to use in pre-screening and/or the minimal penalty model fit for the BIC-ICQ calculation (if applicable)

stage1
: logical value indicating if the first stage of the abbreviated two-stage grid search in the model selection procedure is being performed. FALSE if either performing the second stage of the abbreviated two-stage grid search or if performing the full grid search over all possible penalty parameter combinations.

progress
: a logical value indicating if additional output should be given showing the progress of the fit procedure. If TRUE, such output includes iteration-level information for the fit procedure (iteration number EM_iter, number of MCMC samples nMC, average Euclidean distance between current coefficients and coefficients from t–defined in optimControl–iterations back EM_conv, and number of non-zero fixed and random effects covariates not including the intercept). Additionally, progress = TRUE gives some other information regarding the progress of the variable selection procedure, including the model selection criteria and log-likelihood estimates for each model fit. Default is TRUE.

Value

A list with the following elements:

results
: matrix of summary results for each lambda tuning parameter combination, used to select the 'best' model

out
: list of fit_dat results for the best model

coef
: matrix of coefficient results for each lambda tuning parameter combination. Rows correspond with the rows of the results matrix.

---

sim.data *Simulates data to use for the glmmPen package Simulates data to use for testing the glmmPen package. Possible parameters to specify includes number of total covariates, number of non-zero fixed and random effects, and the magnitude of the random effect covariance values.*

Description

Simulates data to use for the glmmPen package

Simulates data to use for testing the glmmPen package. Possible parameters to specify includes number of total covariates, number of non-zero fixed and random effects, and the magnitude of the random effect covariance values.

Usage

```
sim.data(
  n,
  ptot,
  pnonzero,
  nstudies,
  sd_raneff = 1,
  family = "binomial",
  corr = NULL,
  seed,
  imbalance = 0,
  beta = NULL,
  pnonzerovar = 0
)
```

Arguments

n	integer specifying total number of samples to generate
ptot	integer specifying total number of covariates to generate (values randomly generated from the standard normal distribution)
pnonzero	integer specifying how may of the covariates should have non-zero fixed and random effects
nstudies	number of studies/groups to have in the data
sd_raneff	non-negative value specifying the standard deviation of the random effects covariance matrix (applied to the non-zero random effects)
family	character string specifying which family to generate data from. Family options include "binomial" (default), "poisson", and "gaussian".
corr	optional value to specify correlation in the random effects covariance matrix. Default NULL
seed	integer to use for the setting of a random seed
imbalance	integer of 0 or 1 indicating whether the observations should be equally distributed among the groups (0) or unequally distributed (1).
beta	numeric vector of the fixed effects (including intercept)
pnonzerovar	non-negative integer specifying the number of covariates with a zero-valued fixed effect but a non-zero random effect.

Value

list containing the following elements:

`y`	vector of the response
`X`	model matrix for the fixed effects
`Z`	model matrix for the random effects, organized first by variable and then by group
`pnonzero`	number of non-zero fixed effects
`z1`	values of the random effects for each variable for each level of the grouping factor
`group`	grouping factor
`X0`	model matrix for just the non-zero fixed effects

Index

∗ **classes**
pglmmObj-class, 24
∗ **datasets**
basal, 3

adaptControl, 2, *7*, *11*, *14*, *17*, *24*, *33*

basal, 3
BIC.pglmmObj *(*pglmmObj-class*)*, 24

coef.pglmmObj *(*pglmmObj-class*)*, 24
coef.pglmmObj, *(*pglmmObj-class*)*, 24

family, *9*
family.pglmmObj *(*pglmmObj-class*)*, 24
fit_dat, 4, *34*
fitted.pglmmObj *(*pglmmObj-class*)*, 24
fitted.pglmmObj, *(*pglmmObj-class*)*, 24
fixef.pglmmObj *(*pglmmObj-class*)*, 24
fixef.pglmmObj, *(*pglmmObj-class*)*, 24
formula.pglmmObj *(*pglmmObj-class*)*, 24
formula.pglmmObj, *(*pglmmObj-class*)*, 24

glFormula_edit, 9
glmer, *9*, *11*, *13*
glmm, 10, *20*, *24*
glmmPen, *12*, 12, *17*, *22*, *24*, *29*, *34*, *35*
glmmPen_FineSearch, *15*, 16, *24*, *29*

lambdaControl, *12*, *14*, 18
LambdaSeq, *20*, 21
lme4, *7*, *24*
logLik.pglmmObj *(*pglmmObj-class*)*, 24
logLik.pglmmObj, *(*pglmmObj-class*)*, 24

model.frame.pglmmObj *(*pglmmObj-class*)*, 24
model.frame.pglmmObj, *(*pglmmObj-class*)*, 24
model.matrix.pglmmObj *(*pglmmObj-class*)*, 24
model.matrix.pglmmObj, *(*pglmmObj-class*)*, 24

ngrps.pglmmObj *(*pglmmObj-class*)*, 24
nobs.pglmmObj *(*pglmmObj-class*)*, 24

optimControl, *6–8*, *11*, *12*, *14*, *15*, *17*, 22, *33*, *34*

pglmmObj, *12*, *16*, *18*
pglmmObj *(*pglmmObj-class*)*, 24
pglmmObj-class, 24
pglmmObj-method, *(*pglmmObj-class*)*, 24
plot.pglmmObj *(*pglmmObj-class*)*, 24
plot.pglmmObj, *(*pglmmObj-class*)*, 24
plot_mcmc, *23*, 28
predict.pglmmObj *(*pglmmObj-class*)*, 24
predict.pglmmObj, *(*pglmmObj-class*)*, 24
print.pglmmObj *(*pglmmObj-class*)*, 24
print.pglmmObj, *(*pglmmObj-class*)*, 24

ranef.pglmmObj *(*pglmmObj-class*)*, 24
ranef.pglmmObj, *(*pglmmObj-class*)*, 24
rControl, 30
residuals.pglmmObj *(*pglmmObj-class*)*, 24
residuals.pglmmObj, *(*pglmmObj-class*)*, 24

select_tune, 31
selectControl, *12–18*, *26*, *28*, *33*
selectControl *(*lambdaControl*)*, 18
show, *(*pglmmObj-class*)*, 24
sigma.pglmmObj *(*pglmmObj-class*)*, 24
sigma.pglmmObj, *(*pglmmObj-class*)*, 24
sim.data, 34
summary.pglmmObj *(*pglmmObj-class*)*, 24