Package: hierNest (via r-universe)

June 17, 2026

Type Package

Title Penalized Regression with Hierarchical Nested Parameterization Structure

Version 1.0.2

Date 2026-04-11

Description Efficient implementation of penalized regression with hierarchical nested parametrization for grouped data. The package provides penalized regression methods that decompose subgroup specific effects into shared global effects, Major subgroup specific effects, and Minor subgroup specific effects, enabling structured borrowing of information across related clinical subgroups. Both lasso and hierarchical overlapping group lasso penalties are supported to encourage sparsity while respecting the nested subgroup structure. Efficient computation is achieved through a modified design matrix representation and a custom algorithm for overlapping group penalties.

License GPL (>= 2)

Depends R (>= 3.5),

Imports cli, dotCall64, ggplot2, magrittr, Matrix, methods, rlang (>= 1.1.4), RSpectra, tidyr (>= 1.3.0), rTensor, pROC, plotly

Suggests dplyr, gglasso, glmnet, knitr, markdown, rmarkdown, splines, testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

LazyData true

NeedsCompilation yes

VignetteBuilder knitr

RoxygenNote 7.3.3

Maintainer Ziren Jiang <jian0746@umn.edu>

URL https://github.com/ZirenJiang/hierNest

BugReports https://github.com/ZirenJiang/hierNest/issues

Author Ziren Jiang [aut, cre], Jared Huling [aut], Jue Hou [aut], Lingfeng Huo [aut], Daniel J. McDonald [ctb], Xiaoxuan Liang [ctb], Anibal Solon Heinsfeld [ctb], Aaron Cohen [ctb], Yi Yang [ctb], Hui Zou [ctb], Jerome Friedman [ctb], Trevor Hastie [ctb], Rob Tibshirani [ctb], Balasubramanian Narasimhan [ctb], Kenneth Tay [ctb], Noah Simon [ctb], Junyang Qian [ctb], James Yang [ctb]

Repository https://cran.r-universe.dev

Date/Publication 2026-04-18 19:12:08 UTC

RemoteUrl https://github.com/cran/hierNest

RemoteRef HEAD

RemoteSha 0a7e0c3d43f63056281e9eb1e0ae4eb85ff51b4a

Contents

`coef.cv.hierNest` *Extract subgroup-specific coefficients from a cv.hierNest object*

Description

Computes the composite subgroup-specific coefficient vectors from a cross-validated hierarchical nested model. Each subgroup's coefficient is the sum of the overall mean effect, its group (MDC) effect, and its subgroup (DRG) effect: $\beta^{(s)} = \beta^{overall} + \beta^{group} + \beta^{subgroup}$.

Usage

```
## S3 method for class 'cv.hierNest'
coef(
  object,
  s = c("lambda.min", "lambda.1se"),
  type = c("subgroup", "hierarchical"),
  ...
)
```

Arguments

object	A fitted cv.hierNest object.
s	Value of the penalty parameter lambda at which coefficients are extracted. Either "lambda.min" (default, the lambda that minimizes CV error) or "lambda.1se" (largest lambda within 1 SE of the minimum).
type	Character string specifying the format of the returned coefficients. "subgroup" (default) returns the composite subgroup-specific coefficients (one row per subgroup). "hierarchical" returns the raw hierarchical parameterization (overall, group, and subgroup effect rows).
...	Not used.

Value

A numeric matrix of coefficients. Columns correspond to covariates (including "Intercept"). For type = "subgroup", rows are named by subgroup ID. For type = "hierarchical", rows are labeled with overall, group, and subgroup identifiers.

See Also

cv.hierNest, plot_contribution

---

coef.cv.sparsegl *Extract coefficients from a 'cv.sparsegl' object.*

Description

This function etracts coefficients from a cross-validated [sparsegl()] model, using the stored '"sparsegl.fit"' object, and the optimal value chosen for 'lambda'.

Usage

```
## S3 method for class 'cv.sparsegl'
coef(object, s = c("lambda.1se", "lambda.min"), ...)
```

Arguments

object	Fitted [cv.sparsegl()] object.
s	Value(s) of the penalty parameter ‘lambda‘ at which coefficients are desired. Default is the single value ‘s = "lambda.1se"‘ stored in the CV object (corresponding to the largest value of ‘lambda‘ such that CV error estimate is within 1 standard error of the minimum). Alternatively ‘s = "lambda.min"‘ can be used (corresponding to the minimum of cross validation error estimate). If ‘s‘ is numeric, it is taken as the value(s) of ‘lambda‘ to be used.
...	Not used.

Value

The coefficients at the requested value(s) for ‘lambda‘.

See Also

[cv.sparsegl()] and [predict.cv.sparsegl()].

---

coef.sparsegl *Extract model coefficients from a ‘sparsegl‘ object.*

Description

Computes the coefficients at the requested value(s) for ‘lambda‘ from a [sparsegl()] object.

Usage

```
## S3 method for class 'sparsegl'
coef(object, s = NULL, ...)
```

Arguments

object	Fitted [sparsegl()] object.
s	Value(s) of the penalty parameter ‘lambda‘ at which coefficients are required. Default is the entire sequence.
...	Not used.

Details

‘s‘ is the new vector of ‘lambda‘ values at which predictions are requested. If ‘s‘ is not in the lambda sequence used for fitting the model, the ‘coef‘ function will use linear interpolation to make predictions. The new values are interpolated using a fraction of coefficients from both left and right ‘lambda‘ indices.

Value

The coefficients at the requested values for ‘lambda‘.

See Also

[sparsegl()] and [predict.sparsegl()].

`cv.hierNest` *Cross-validated hierarchical nested regularization for subgroup models*

Description

Fits regularization paths for hierarchical subgroup-specific penalized learning problems, leveraging nested group structure (such as Major Diagnostic Categories [MDC] and Diagnosis-Related Groups [DRG]) with options for lasso or overlapping group lasso penalties. Performs cross-validation to select tuning parameters for penalization and subgroup structure.

This function enables information sharing across related subgroups by reparameterizing covariate effects into overall, group-specific, and subgroup-specific components and supports structured shrinkage through hierarchical regularization. Users may select between the lasso (hierNest-Lasso) and overlapping group lasso (hierNest-OGLasso) frameworks, as described in Jiang et al. (2024, submitted, see details below).

Usage

```
cv.hierNest(
  x,
  y,
  group = NULL,
  family = c("gaussian", "binomial"),
  nlambda = 100,
  lambda.factor = NULL,
  pred.loss = c("default", "mse", "deviance", "mae", "misclass", "ROC"),
  lambda = NULL,
  pf_group = NULL,
  pf_sparse = NULL,
  intercept = FALSE,
  asparse1 = c(0.5, 20),
  asparse2 = c(0.01, 0.2),
  asparse1_num = 4,
  asparse2_num = 4,
  standardize = TRUE,
  lower_bnd = -Inf,
  upper_bnd = Inf,
  eps = 1e-08,
  maxit = 3e+06,
  hier_info = NULL,
  method = "overlapping",
  partition = "subgroup",
  cvmethod = "general"
)
```

Arguments

x	Matrix of predictors, of dimension $n \times p$; each row is an observation. Can be a dense or sparse matrix.
y	Response variable. For ‘family="gaussian"‘, should be numeric. For ‘family="binomial"‘, should be a factor with two levels or a numeric vector with two unique values.
group	Optional vector or factor indicating group assignments for variables. Used for custom grouping.
family	Character string specifying the model family. Options are ‘"gaussian"‘ (default) for least-squares regression, or ‘"binomial"‘ for logistic regression.
nlambda	Number of lambda values to use for regularization path. Default is 100.
lambda.factor	Factor determining the minimal value of lambda in the sequence, where ‘min(lambda) = lambda.factor * max(lambda)‘. See Details.
pred.loss	Character string indicating loss to minimize during cross-validation. Options include ‘"default"‘, ‘"mse"‘, ‘"deviance"‘, ‘"mae"‘, ‘"misclass"‘, and ‘"ROC"‘.
lambda	Optional user-supplied sequence of lambda values (overrides ‘nlambda‘/‘lambda.factor‘).
pf_group	Optional penalty factors on the groups, as a numeric vector. Default adjusts for group size.
pf_sparse	Optional penalty factors on the l1-norm (for sparsity), as a numeric vector.
intercept	Logical; whether to include an intercept in the model. Default is TRUE.
asparse1	Relative weight(s) for the first (e.g., group) layer of the overlapping group lasso penalty. Default is c(0.5, 20).
asparse2	Relative weight(s) for the second (e.g., subgroup) layer. Default is c(0.01, 0.2).
asparse1_num	Number of values in asparse1 grid (for grid search). Default is 4.
asparse2_num	Number of values in asparse2 grid (for grid search). Default is 4.
standardize	Logical; whether to standardize predictors prior to model fitting. Default is TRUE.
lower_bnd	Lower bound(s) for coefficient values. Default is -Inf.
upper_bnd	Upper bound(s) for coefficient values. Default is Inf.
eps	Convergence tolerance for optimization. Default is 1e-8.
maxit	Maximum number of optimization iterations. Default is 3e6.
hier_info	Required for ‘method = "overlapping"‘; a matrix describing the hierarchical structure of the subgroups (see Details).
method	Character; either ‘"overlapping"‘ for overlapping group lasso, ‘"sparsegl"‘ for sparse group lasso, or ‘"general"‘ for other hierarchical regularization. Default is ‘"overlapping"‘.
partition	Character string; determines subgroup partitioning. Default is ‘"subgroup"‘.
cvmethod	Cross-validation method. Options include ‘"general"‘ (default), ‘"grid_search"‘, or ‘"user_supply"‘ (for user-supplied grid).

Details

The hierarchical nested framework decomposes covariate effects into overall, group, and subgroup-specific components, with regularization encouraging fusion or sparsity across these hierarchical levels. The function can fit both the lasso penalty (allowing arbitrary zero/non-zero patterns) and the overlapping group lasso penalty (enforcing hierarchical selection structure), as described in Jiang et al. (2024, submitted).

The argument ‘hier_info‘ must be supplied for ‘"overlapping"‘ method, and encodes the hierarchical relationship between groups and subgroups (e.g., MDCs and DRGs).

Cross-validation is used to select tuning parameters, optionally over a grid for hierarchical penalty weights (asparse1, asparse2), and the regularization parameter lambda.

Value

An object containing the fitted hierarchical model and cross-validation results, including:

`fit`	Fitted model object.
`lambda`	Sequence of lambda values considered.
`cv_error`	Cross-validation error/loss for each combination of tuning parameters.
`best_params`	Best tuning parameters selected.
`...`	Additional diagnostic and output fields.

References

Jiang, Z., Huo, L., Hou, J., & Huling, J. D. "Heterogeneous readmission prediction with hierarchical effect decomposition and regularization".

`estimate_risk` *Calculate information criteria.*

Description

This function uses the degrees of freedom to calculate various information criteria. This function uses the "unknown variance" version of the likelihood. Only implemented for Gaussian regression. The constant is ignored (as in [stats::extractAIC()]).

Usage

```
estimate_risk(object, x, type = c("AIC", "BIC", "GCV"), approx_df = FALSE)
```

Arguments

`object`	fitted object from a call to [sparsegl()].
`x`	Matrix. The matrix of predictors used to estimate the ‘sparsegl‘ object. May be missing if ‘approx_df = TRUE‘.
`type`	one or more of AIC, BIC, or GCV.
`approx_df`	the ‘df‘ component of a ‘sparsegl‘ object is an approximation (albeit a fairly accurate one) to the actual degrees-of-freedom. However, the exact value requires inverting a portion of ‘X’X‘. So this computation may take some time (the default computes the exact df).

Value

a ‘data.frame‘ with as many rows as ‘object$lambda‘. It contains columns ‘lambda‘, ‘df‘, and the requested risk types.

References

Vaiter S, Deledalle C, Peyré G, Fadili J, Dossal C. (2012). *The Degrees of Freedom of the Group Lasso for a General Design*. https://arxiv.org/pdf/1212.6478.pdf.

See Also

[sparsegl()] method.

`example_data` *Example dataset*

Description

Example dataset for the sample code of hierNest

Usage

```
data(example_data)
```

Format

A data frame with X rows and Y columns:

X Matrix of predictors

Y Outcome variable

hier_info Group information passed to hierNest functions

`hierNest` *Fit Hierarchical Nested Regularization Model (hierNest)*

Description

Fits a hierarchical nested penalized regression model for subgroup-specific effects using overlapping group lasso penalties. This function encodes the hierarchical structure (e.g., MDC and DRG) via a reparameterized design matrix and enables information borrowing across related subgroups.

Usage

```
hierNest(
  x,
  y,
  group = NULL,
  family = c("gaussian", "binomial"),
  nlambda = 100,
  lambda.factor = NULL,
  lambda = NULL,
  pf_group = NULL,
  pf_sparse = NULL,
  intercept = FALSE,
  asparse1 = 1,
  asparse2 = 0.05,
  standardize = TRUE,
  lower_bnd = -Inf,
  upper_bnd = Inf,
  eps = 1e-08,
  maxit = 3e+06,
  hier_info = NULL,
  random_asparse = FALSE,
  method = "overlapping"
)
```

Arguments

`x`	Matrix of predictors ($n \times p$), where each row is an observation.
`y`	Response variable (numeric for "gaussian", binary or factor for "binomial").
`group`	Optional grouping vector (not required for "overlapping" method).
`family`	Model family; either "gaussian" for least squares or "binomial" for logistic regression.
`nlambda`	Number of lambda values in the regularization path (default: 100).
`lambda.factor`	Factor for minimal value of lambda in the sequence.
`lambda`	Optional user-supplied lambda sequence.
`pf_group`	Penalty factor(s) for each group; defaults to sqrt(group size).

pf_sparse	Penalty factors for individual predictors (L1 penalty).
intercept	Logical; should an intercept be included? Default is FALSE.
asparse1	Relative weight for group-level penalty (default: 1).
asparse2	Relative weight for subgroup-level penalty (default: 0.05).
standardize	Logical; standardize predictors? Default is TRUE.
lower_bnd	Lower bound for coefficients (default: -Inf).
upper_bnd	Upper bound for coefficients (default: Inf).
eps	Convergence tolerance (default: 1e-8).
maxit	Maximum number of optimization iterations (default: 3e6).
hier_info	Required. Matrix encoding the hierarchical structure (see Details).
random_asparse	Logical; use random sparse penalty? Default: FALSE.
method	Type of hierarchical regularization ("overlapping" [default], "sparsegl", or "general").

Details

This function builds a hierarchical design matrix reflecting group/subgroup structure (e.g., Major Diagnostic Categories [MDCs] and Diagnosis Related Groups [DRGs]), encoding overall, group-specific, and subgroup-specific effects. It fits a penalized model using overlapping group lasso, as described in Jiang et al. (2024, submitted). The main computational engine is `hierNest::overlapping_gl`.

Value

Returns a model fit object as produced by `hierNest::overlapping_gl`, including selected coefficients, cross-validation results, and tuning parameters.

References

Jiang, Z., Huo, L., Hou, J., Vaughan-Sarrazin, M., Smith, M. A., & Huling, J. D. (2024). Heterogeneous readmission prediction with hierarchical effect decomposition and regularization.

---

make_irls_warmup *Create starting values for iterative reweighted least squares*

Description

This function may be used to create potentially valid starting values for calling [sparsegl()] with a [stats::family()] object. It is not typically necessary to call this function (as it is used internally to create some), but in some cases, especially with custom generalized linear models, it may improve performance.

Usage

```
make_irls_warmup(nobs, nvars, b0 = 0, beta = double(nvars), r = double(nobs))
```

Arguments

nobs	Number of observations in the response (or rows in ‘x‘).
nvars	Number of columns in ‘x‘
b0	Scalar. Initial value for the intercept.
beta	Vector. Initial values for the coefficients. Must be length ‘nvars‘ (or a scalar).
r	Vector. Initial values for the deviance residuals. Must be length ‘nobs‘ (or a scalar).

Details

Occasionally, the irls fitting routine may fail with an admonition to create valid starting values.

Value

List of class ‘irlsspgl_warmup‘

---

overlapping_gl *Fit an Overlapping Group Lasso Model*

Description

This function fits an overlapping group lasso model with hierarchical regularization, allowing both sparsity within groups and across groups.

Usage

```
overlapping_gl(
  x,
  y,
  group = NULL,
  family = c("gaussian", "binomial"),
  nlambda = 100,
  lambda.factor = ifelse(nobs < nvars, 0.01, 1e-04),
  lambda = NULL,
  pf_group = sqrt(bs),
  pf_sparse = rep(1, nvars),
  intercept = TRUE,
  asparse1 = 1,
  asparse2 = 0.05,
  standardize = TRUE,
  lower_bnd = -Inf,
  upper_bnd = Inf,
  weights = NULL,
  offset = NULL,
  warm = NULL,
  trace_it = 0,
```

```
  dfmax = as.integer(max(group)) + 1L,
  pmax = min(dfmax * 1.2, as.integer(max(group))),
  eps = 1e-08,
  maxit = 3e+06,
  cn,
  drgix,
  drgiy,
  cn_s,
  cn_e,
  random_asparse = FALSE
)
```

Arguments

x	A numeric matrix of predictor variables (no missing values allowed).
y	A numeric vector of response variable values.
group	An integer vector defining the group membership for each predictor. Default is NULL (each predictor forms its own group).
family	A character string specifying the model family. Options are "gaussian" (default) and "binomial".
nlambda	Number of lambda values. Default is 100.
lambda.factor	Factor determining minimum lambda as a fraction of maximum lambda. Default depends on dimensionality.
lambda	Numeric vector of lambda values. If provided, overrides ‘nlambda‘ and ‘lambda.factor‘.
pf_group	Penalty factor for groups. Default is square root of group size.
pf_sparse	Penalty factor for individual predictors. Default is 1 for each predictor.
intercept	Logical; whether to include an intercept. Default is TRUE.
asparse1	Sparsity penalty factor controlling group-level sparsity. Default is 1.
asparse2	Sparsity penalty factor controlling within-group sparsity. Default is 0.05.
standardize	Logical; if TRUE, standardizes predictors before fitting. Default is TRUE.
lower_bnd	Numeric vector specifying lower bounds for coefficients. Default is -Inf.
upper_bnd	Numeric vector specifying upper bounds for coefficients. Default is Inf.
weights	Optional numeric vector of observation weights. Currently limited functionality.
offset	Optional numeric vector specifying a known component to be included in the linear predictor.
warm	Optional initial values for optimization.
trace_it	Integer indicating the verbosity level. Default is 0 (no output).
dfmax	Maximum number of groups allowed in the model. Default derived from groups.
pmax	Maximum number of predictors allowed in the model. Default derived from groups.
eps	Numeric convergence threshold for optimization. Default is 1e-08.
maxit	Maximum number of iterations for optimization. Default is 3e+06.

cn	Additional internal numeric parameter for optimization.
drgix, drgiy	Numeric vectors specifying indices for specific group and predictor structures.
cn_s, cn_e	Numeric vectors specifying starting and ending indices for substructures.
random_asparse	Logical; if TRUE, randomly selects sparsity parameters. Default is FALSE.

Value

An object of class 'sparsegl' containing:

call	The matched function call.
lambda	The lambda values used for fitting.
asparse1, asparse2	Sparsity parameters used.
nobs	Number of observations.
pf_group, pf_sparse	Penalty factors used.
coefficients	Estimated coefficients.

Additional components relevant to model diagnostics and fitting.

`plot.cv.hierNest` *Plot method for cv.hierNest objects*

Description

Plot method for cv.hierNest objects

Usage

```
## S3 method for class 'cv.hierNest'
plot(x, type = c("coefficients", "Subgroup effects"), ...)
```

Arguments

x	An object of class `cv.hierNest`
type	Plot type, e.g. "fit" for observed vs fitted (default), others you may add later (e.g., "coef").
...	Other parameters

Value

Invisibly returns `x`

`plot_contribution` *Boxplot or bar chart of per-covariate contributions to the linear predictor*

Description

For each subject, computes the contribution of each covariate to the linear predictor as $x_j \cdot \beta_j$, where β_j is the subject's subgroup-specific coefficient for covariate j.

When no `subject_id` is supplied, produces a boxplot showing the distribution of contributions across all subjects. When a single `subject_id` is supplied, produces a bar chart for that subject.

Usage

```
plot_contribution(
  object,
  newx,
  hier_info,
  s = c("lambda.min", "lambda.1se"),
  subject_id = NULL,
  top_n = NULL,
  include_intercept = FALSE
)
```

Arguments

`object`	A fitted `cv.hierNest` object.
`newx`	Numeric matrix of predictor values ($n \times p$), with the same columns as the original training data.
`hier_info`	Numeric matrix encoding the hierarchical structure for the subjects in `newx`. First column is group (MDC) membership; second column is subgroup (DRG) membership. Must have `nrow(newx)` rows.
`s`	Value of `lambda` to use for coefficient extraction. Either `"lambda.min"` (default) or `"lambda.1se"`.
`subject_id`	Optional. A row index (integer) or row name (character) identifying a single subject in `newx`. If provided, a bar chart of that subject's covariate contributions is returned instead of a boxplot.
`top_n`	Optional integer. If provided, only the top `top_n` covariates (by median absolute contribution across subjects, or by absolute contribution for a single subject) are shown.
`include_intercept`	Logical; if `TRUE`, include the intercept contribution as a separate covariate. Default is `FALSE`.

Details

The function uses `coef.cv.hierNest` to obtain the composite subgroup-specific coefficients. Each subject is mapped to their subgroup via `hier_info`, and the element-wise product $x_j\beta_j$ quantifies how much each covariate contributes to that subject's predicted risk (linear predictor scale).

Covariates are ordered on the x-axis by descending median absolute contribution (boxplot) or descending absolute contribution (bar chart), so the most influential variables appear on the left.

Value

A `ggplot2` object.

See Also

`coef.cv.hierNest`, `cv.hierNest`

`predict.cv.sparsegl` *Make predictions from a 'cv.sparsegl' object.*

Description

This function makes predictions from a cross-validated [cv.sparsegl()] object, using the stored 'sparsegl.fit' object, and the value chosen for 'lambda'.

Usage

```
## S3 method for class 'cv.sparsegl'
predict(
  object,
  newx,
  s = c("lambda.1se", "lambda.min"),
  type = c("link", "response", "coefficients", "nonzero", "class"),
  ...
)
```

Arguments

`object`	Fitted [cv.sparsegl()] object.
`newx`	Matrix of new values for 'x' at which predictions are to be made. Must be a matrix. This argument is mandatory.
`s`	Value(s) of the penalty parameter 'lambda' at which coefficients are desired. Default is the single value 's = "lambda.1se"' stored in the CV object (corresponding to the largest value of 'lambda' such that CV error estimate is within 1 standard error of the minimum). Alternatively 's = "lambda.min"' can be used (corresponding to the minimum of cross validation error estimate). If 's' is numeric, it is taken as the value(s) of 'lambda' to be used.

type	Type of prediction required. Type ‘"link"‘ gives the linear predictors for ‘"binomial"‘; for ‘"gaussian"‘ models it gives the fitted values. Type ‘"response"‘ gives predictions on the scale of the response (for example, fitted probabilities for ‘"binomial"‘); for ‘"gaussian"‘ type ‘"response"‘ is equivalent to type ‘"link"‘. Type ‘"coefficients"‘ computes the coefficients at the requested values for ‘s‘. Type ‘"class"‘ applies only to ‘"binomial"‘ models, and produces the class label corresponding to the maximum probability. Type ‘"nonzero"‘ returns a list of the indices of the nonzero coefficients for each value of s.
...	Not used.

Value

A matrix or vector of predicted values.

See Also

[cv.sparsegl()] and [coef.cv.sparsegl()].

---

predict.sparsegl *Make predictions from a ‘sparsegl‘ object.*

Description

Similar to other predict methods, this function produces fitted values and class labels from a fitted [‘sparsegl‘] object.

Usage

```
## S3 method for class 'sparsegl'
predict(
  object,
  newx,
  s = NULL,
  type = c("link", "response", "coefficients", "nonzero", "class"),
  ...
)
```

Arguments

object	Fitted [sparsegl()] model object.
newx	Matrix of new values for ‘x‘ at which predictions are to be made. Must be a matrix. This argument is mandatory.
s	Value(s) of the penalty parameter ‘lambda‘ at which predictions are required. Default is the entire sequence used to create the model.

type	Type of prediction required. Type ‘"link"‘ gives the linear predictors for ‘"binomial"‘; for ‘"gaussian"‘ models it gives the fitted values. Type ‘"response"‘ gives predictions on the scale of the response (for example, fitted probabilities for ‘"binomial"‘); for ‘"gaussian"‘ type ‘"response"‘ is equivalent to type ‘"link"‘. Type ‘"coefficients"‘ computes the coefficients at the requested values for ‘s‘. Type ‘"class"‘ applies only to ‘"binomial"‘ models, and produces the class label corresponding to the maximum probability. Type ‘"nonzero"‘ returns a list of the indices of the nonzero coefficients for each value of s.
...	Not used.

Details

‘s‘ is the new vector of ‘lambda‘ values at which predictions are requested. If ‘s‘ is not in the lambda sequence used for fitting the model, the ‘coef‘ function will use linear interpolation to make predictions. The new values are interpolated using a fraction of coefficients from both left and right ‘lambda‘ indices.

Value

The object returned depends on type.

See Also

[sparsegl()], [coef.sparsegl()].

---

predict_hierNest *Predict Method for hierNest Objects*

Description

Provides predictions from a fitted hierarchical model (‘hierNest‘) using new data.

Usage

```
predict_hierNest(
  object,
  newx,
  hier_info,
  type = c("link", "response", "coefficients", "nonzero", "class"),
  ...
)
```

Arguments

`object`	A fitted hierNest model object.
`newx`	A numeric matrix of new predictor values for prediction.
`hier_info`	A numeric matrix with hierarchical grouping information. First column is MDC-level grouping; second column is DRG-level grouping.
`type`	Character string specifying the type of prediction required. Options include "link", "response", "coefficients", "nonzero", and "class".
`...`	Additional arguments passed to lower-level prediction methods.

Details

This function prepares a hierarchical design matrix based on ‘hier_info‘, constructs the required Khatri-Rao product, and reorganizes it before generating predictions from the provided ‘object‘.

Value

Predictions based on the specified ‘type‘. Typically, returns:

- Numeric vector or matrix of predicted values (for "link" or "response").
- Model coefficients (for "coefficients").
- Nonzero coefficient indices (for "nonzero").
- Class labels for categorical outcomes (for "class").

`zero_norm` *Calculate common norms*

Description

Calculate different norms of vectors with or without grouping structures.

Usage

```
zero_norm(x)

one_norm(x)

two_norm(x)

grouped_zero_norm(x, gr)

grouped_one_norm(x, gr)

grouped_two_norm(x, gr)

grouped_sp_norm(x, gr, asparse)
```

```
gr_one_norm(x, gr)

gr_two_norm(x, gr)

sp_group_norm(x, gr, asparse = 0.05)
```

Arguments

x	A numeric vector.
gr	An integer (or factor) vector of the same length as x.
asparse	Scalar. The weight to put on the l1 norm when calculating the group norm.

Value

A numeric scalar or vector

Functions

- `zero_norm()`: l0-norm (number of nonzero entries).
- `one_norm()`: l1-norm (Absolute-value norm).
- `two_norm()`: l2-norm (Euclidean norm).
- `grouped_zero_norm()`: A vector of group-wise l0-norms.
- `grouped_one_norm()`: A vector of group-wise l1-norms.
- `grouped_two_norm()`: A vector of group-wise l2-norms.
- `grouped_sp_norm()`: A vector of length ‘unique(gr)‘ consisting of the ‘asparse‘ convex combination of the l1 and l2-norm for each group.
- `gr_one_norm()`: The l1-norm norm of a vector (a scalar).
- `gr_two_norm()`: The sum of the group-wise l2-norms of a vector (a scalar).
- `sp_group_norm()`: The sum of the ‘asparse‘ convex combination of group l1 and l2-norms vectors (a scalar).

Index