Package ‘dlmBE’

January 4, 2018

Version 0.1

Date 2017-12-1

title Distributed Lag Models for built environment applications

Description This package fits distributed lag models (DLMs) to describe how the association between the presence of built environment features and an outcome varies as a function of distance between locations for study participants and locations for environment features.

Author Andrew Whiteman <awhitem@umich.edu>,
Jonggyu Baek <jongguri@umich.edu>,
Brisa Sanchez <brisa@umich.edu>

Maintainer Andrew Whiteman <awhitem@umich.edu>

License GPL (>= 2)

Depends R (>= 3.0.2), lme4 (>= 1.1), ggplot2, methods

LinkingTo

Imports Matrix

Suggests devtools

RoxygenNote 6.0.1

RemoteType local

RemoteUrl /Users/AprilVKwong/Dropbox/software/dlm

RemoteSha NA

RemoteBranch master

RemoteUsername asw221

RemoteRepo dlm

R topics documented:

dlmBE-package 2
basis 3
changePoint 5
cholfVar 5
dlm 6
dlMod 8
estimands 8
interpret.dlm 9

LagBasis 10
lagIndex 11
lme4.dlm 12
makeDlMod 12
omega 13
parse.names 14
plotDlm 14
predict.LagBasis 15
scaleMat 16
simdata 16
smoothing 17
SmoothLag 18

Index **19**

`dlmBE-package` *dlmBE: Distributed lag models in R using lme4*

Description

This package fits the distributed lag models (DLMs) described by Baek et al (2016) and Baek et al (2017), which estimate the association between the presence of built environment features and an outcome as a function of distance between the locations for study participants and locations for environment features or community resources. These models circumvent the need to pre-specify a radius within which to measure the availability of community resources. Distributed lag models have a long history in a variety of fields. For built environment research, we define the lagged exposure as the value of an environment feature between two radii, r_{l-1} and r_l from study locations, $l = 1, 2, \ldots, L$, where $r_0 = 0$; e.g., the lagged exposure is the number of convenience stores within "ring"-shaped areas around study participants residential address. The package supports generalized linear regression models, as well as generalized linear mixed models. In both instances, multiple lagged exposure covariates maybe included, as well as interactions between the lagged covariates and other categorical covariates (e.g., quartiles of age).

Let Y_{ij} be an outcome measured at location i at visit j, and $X_{ij}(r_{l-1}; r_l)$ be an environment feature measured during visit j within a ring-shaped area around location i between radii r_{l-1} and r_l; and r_L be the maximum distance around locations beyond which there is no association between the environment feature and the outcome. A typical unadjusted generalized linear mixed model that can be fitted in this version of the package is,

$$g(E(Y_{ij}|b_i)) = \beta_0 + \sum_{l=1}^{L} \beta(r_{l-1}; r_l) X(r_{l-1}; r_l) + W_{ij} b_i$$

where $g(\cdot)$ is a link function appropriate for the distribution of the outcome; β_0 represents an intercept; the association of the environment feature measured between radii r_{l-1} and r_l and the outcome is $\beta(r_{l-1}; r_l)$; and W_{ij} are covariates related to random effects, b_i (e.g., random intercepts and slopes). The coefficients $\beta(r_{l-1}; r_l)$ are constrained to follow a smooth function of distance from the locations of interest; the constraint is imposed by modeling the coefficients using smoothing splines. Other models could be used, although smoothing splines are the only supported option at this time.

The model easily simplifies to generalized linear regression modes (e.g., when there is only one visit), and can be extended in the following directions. Adjustment covariates can be easily included. In addition, interaction terms between covariates and the DL covariates are also supported.

For example, terms such as: $\sum_{l=1}^{L} \theta(r_{l-1}; r_l) X(r_{l-1}; r_l) Z_i$, where Z_i is another covariate, can be included. The interaction coefficients $\theta(r_{l-1}; r_l)$ have the usual interpretation, but the magnitude of the interaction can vary over distance from locations of interest; $\theta(r_{l-1}; r_l)$ are also constrained using smoothing splines. Finally, weighted regression models are also supported.

We assume the user has calculated distances from every participant's location to every community resource/feature. The distances can be network distances or Euclidian distances. Those distances are then used to calculate the distributed lag covariates, $X(r_{l-1}; r_l)$, by specifying L and the radii r_l, $l = 1, 2, \ldots, L$. See Baek et al (2016) for guidance on choosing L and r.

The package includes a series of functions to pass formulas and data to **lme4**, which is used for estimation of the DLM. All those functions are documented in this manual, although a typical user will primarily interact with XXX, xxx, and xxx. For example:

References

Baek J, Sanchez BN, Berrocal VJ, & Sanchez-Vaznaugh EV (2016) Epidemiology 27(1):116-24. (PubMed)

Baek J, Hirsch JA, Moore K, Tabb LP, et al. (2017) Epidemiology 28(3):403-11. (PubMed)

Bates D, Maechler M, Bolker BM, & Walker SC (2015) Fitting linear mixed-effects models using lme4. J Stat Softw 67(1). (jstatsoft.org)

`basis` *Basis vectors*

Description

Constructs a set of basis vectors C_0 and K_1 used to constrain distributed lag coefficients, $\beta(\cdot, \cdot)$, using splines. The basis vectors depend on the radii that define ring-shaped areas around participant locations. Typical usage relies on calling basis application functions, like `cr` (e.g. in `dlm` model formulas); users should not often have to interact with `basis` directly.

Usage

```
basis(x, center = TRUE, scale = FALSE, .fun = NULL, ...)
```

Arguments

`x`	radii that define ring-shaped areas around participant locations
`center`	if `TRUE` (the default), `x` will be mean centered prior to computing the basis. Otherwise, if given a `numeric` value, `x` will be centered at `center`
`scale`	if `TRUE` (default = `FALSE`), parameter `x` will be scaled by `sd(x)`. Otherwise, if given a `numeric` value, `x` will be scaled by `scale`
`.fun`	a function to define the type of basis. The default is to compute a cubic radial basis based on pairwise cubed absolute differences among the radii. See Details
`...`	other parameters passed to `.fun`

Details

Alternative distance functions, `.fun`, may be specified, and error checking on the user's choice of `.fun` is deliberately missing. Proper candidates for `.fun` should return an $(L \times L)$ matrix, where L is the same as `length(x)`; elements of this matrix are typically non-negative.

In addition, new distance function definitions should follow the idiom:

```
function(x, y, ...)
  if (missing(y))  y <- x
  ...
```

The default value of `.fun` computes cubic radial distance, which amounts to `abs(outer(x, y, "-"))^3`; the computed vectors are then transformed following Rupert, Wand, and Carroll (2003), such that the spline can be fitted (and penalized) as a mixed-model.

Value

An object of class `LagBasis`

Decomposition

Once a basis function ($\delta(\cdot, \cdot)$) and radii (r) are chosen, define the matrix, $C_{1,ij} = \delta(r_i, r_j)$, and let,

$$C_0 = [1_L, r]$$

$$C_1 = QR$$

$$M_1 = Q_{-(1:2)}$$

$$K_1 = C_1 M_1 (M_1^T C_1 M_1)^{-\frac{1}{2}}$$

where A_{-j} denotes a matrix A with column(s) j removed. Then the (scaled) distributed lag effects are $\beta = C_0\alpha + K_1 b$, where $b_\ell \sim \mathrm{N}(0, \sigma_b^2)$, for $\ell = 1, \ldots, L - 2$.

References

Rupert D, Wand MP, & Carroll RJ (2003) Semiparametric Regression. New York: Cambridge University Press.

See Also

`cr`, `dlm`

changePoint *Lag coefficient change points*

Description

For each set of distributed lag terms in the model, finds and returns the (named) integer indices of radii where the corresponding coefficient is significantly different from zero, but the immediately larger radius is not significantly different from zero.

Usage

```
changePoint(object, ...)

## S4 method for signature 'dlMod'
changePoint(object, ...)
```

Arguments

`object`	a fitted model object with viable `lagIndex` and `confint` methods
`...`	additional arguments passed to `confint`

Value

A list of integer vectors. One list element for each set of DL terms in the model

Examples

```
data (simdata)

## Setup distance count matrix and corresponding lag distances
X <- as.matrix(simdata[, -(1:3)])
lag <- seq(0.1, 10, length.out = ncol(X))

fit <- dlm(Y ~ Age + Gender + cr(lag, X), data = simdata)
changePoint(fit)
```

cholfVar *Extract Cholesky factor of inverse Information matrix*

Description

Computes the Cholesky factor, L, of the inverse of the Fisher Information matrix for all regression coefficients in a fitted model. The coefficient covariance matrix can then be computed as LL^T.

Usage

```
cholfVar(object, ...)

## S4 method for signature 'merMod'
cholfVar(object, ...)
```

Arguments

`object`	a fitted model object
`...`	additional arguments

Value

A square numeric matrix represented as a `Matrix::sparseMatrix` object

Method for 'merMod' objects

For a mixed-effects model of the form, for example,

$$g(E(Y_i|b_i)) = X\beta + Z_i b_i$$

with link function $g(\cdot)$, and $b_i \sim \mathrm{N}(0, \Sigma_b)$, the Cholesky factor returned will be such that $LL^T = I^{-1}(\theta)$, where $\theta = (\beta^T, b_1^T, b_2^T, \ldots)^T$.

`dlm` *Distributed lag models*

Description

Fit distributed lag models using **lme4** to penalize smooth terms. Other random effects terms and generalized mixed models supported.

Usage

```
dlm(formula, data, subset, na.action, weights, offset,
    method = c("REML", "MLE"), family = gaussian(),
    control = list(), ...)
```

Arguments

`formula`	an object of class `stats::formula`: a symbolic description of the model to be fitted. See Details
`data`	an optional data frame, list, or environment containing the variables of the model to be fitted
`subset`	optional vector specifying a subset of observations to be used in the fitting process
`na.action`	optional function that indicates what should happen when the data contains `NA`'s. The default is set by the `na.action` setting of `base::options`, usually `stats::na.omit`
`weights`	optional vector of weights to be used in the fitting process. Should be `NULL` or a numeric vector
`offset`	a known offset term to include in the model, e.g. for `poisson()` family models
`method`	algorithm used to fit the DLM. Partial matching and capitalization allowed. The default is `"REML"` for linear/`gaussian(link = "identity")` family models, and `"MLE"` otherwise

`family`	a description of the error distribution and link function to be used in the model. The default is `gaussian(link = "identity")`. See `stats::family` for possible family functions and details
`control`	either a `list` object with arguments to be passed to the `lme4::lmerControl` sequence, or the output of `[g]lmerConrol` directly
`...`	Additional parameters passed to `lme4::lFormula`

Details

Models are specified using typical **lme4** `formula` syntax with at least one set of lag terms returned by a given smoothing function (e.g. see `cr`). The smoothing function can be any that returns a `SmoothLag` basis object. See Examples for a basic call to `dlm` using the formula interface, and a cubic radial lag basis specified via `cr`, and the **dlmBE** `package documentation` for a discussion of the types of models `dlm` is designed to handle.

Value

An S4 object that inherits from `dlMod` (and `lme4::merMod`, by extension) containing the results of the fitted model. Many standard model summary methods are available for these object types

References

Baek J, Sanchez BN, Berrocal VJ, & Sanchez-Vaznaugh EV (2016) Epidemiology 27(1):116-24. (PubMed)

Baek J, Hirsch JA, Moore K, Tabb LP, et al. (2017) Epidemiology 28(3):403-11. (PubMed)

Bates D, Maechler M, Bolker BM, & Walker SC (2015) Fitting linear mixed-effects models using lme4. J Stat Softw 67(1). (jstatsoft.org)

See Also

`lme4::lmer`, `cr`, `dlMod`

Examples

```
data (simdata)

## Setup distance count matrix and corresponding lag distances
X <- as.matrix(simdata[, -(1:3)])
lag <- seq(0.1, 10, length.out = ncol(X))

fit <- dlm(Y ~ Age + Gender + cr(lag, X), data = simdata)
summary (fit)
```

dlMod *Distributed lag model objects*

Description

A fitted distributed lag model object. Inherits from **lme4**'s merMod so that most methods defined for this parent class should work seamlessly within **dlmBE** analysis

Slots

resp An lme4::lmResp object to store a (mixed) model response variable

bases A list of LagBasis objects corresponding to the unique set of lag bases used to fit the model

index A (named) integer vector providing the index of the basis set in bases corresponding to each lag term in the model

estimands *Extract DLM parameter estimates*

Description

Extract regression coefficients, variances, etc. from fitted dlMod objects

Usage

```
coef(object, ...)

confint(object, ...)

Sigma(object, ...)

vcoef(object, ...)

vcoef0(object, ...)

## S4 method for signature 'dlMod'
Sigma(object, scaled = TRUE, ...)

## S4 method for signature 'dlMod'
vcoef0(object, scaled = TRUE, ...)

## S4 method for signature 'dlMod'
vcoef(object, scaled = TRUE, ...)

## S3 method for class 'dlMod'
coef(object, scaled = TRUE, ...)

## S3 method for class 'dlMod'
confint(object, parm, level = 0.95, scaled = TRUE,
  coef = TRUE, ...)
```

Arguments

`object`	a fitted `dlMod` object
`...`	additional arguments
`scaled`	if `TRUE` (the default), any lag parameters are scaled for more natural interpretation (see Details)
`parm`	an integer or character index to subset parameters
`level`	the desired confidence level
`coef`	if `coef = TRUE` (the default), `confint.dlMod` will include an extra column (with `colname "coef"`) for the regression coefficients themselves

Details

Other typical methods like `residuals`, and `sigma`, etc. are handled via inheritance from **lme4** classes. If the argument `scaled = TRUE`, parameter estimates are scaled by the areas between radii and summed so that they can be interpreted as the estimate up to a given radius (e.g. see the **dlmBE** package documentation).

`coef.dlMod` follows the format of `lme4::coef.merMod` to return the sums of fixed and random effects for each level and grouping factor

`confint.dlMod` returns confidence intervals for regression coefficients following `stats::confint`

`Sigma` returns the regression coefficient covariance matrix. Row and column indices are in the same order as `vcoef` (see below)

`vcoef` returns vectorized coefficients from the fitted model. Fixed effects come before distributed lag coefficients, which come before other random effects coefficients. For example, if β is a vector of fixed effects; θ_1 and θ_2 are vectors of (separately penalized) DL coefficients; and b_1, b_2, ... are additional random effects vectors for groups $1, 2, \ldots$, then `vcoef` will return the (named) vector $(\beta^T, \theta_1^T, \theta_2^T, b_1^T, b_2^T, \ldots)^T$. `vcoef0` returns the same coefficient vector but without names (and is slightly faster).

Value

All of these functions return `numeric` data

`interpret.dlm` *Interpret a DLM formula*

Description

Given an appropriate DL model formula and data-frame, prepares the data to be fit by interpreting the formula and extracting available smooth lag terms.

Usage

```
interpret.dlm(formula, data, .names.func = function(n) paste("pseudoGroups",
  n, sep = ""))
```

Arguments

`formula`	a symbolic description of the model to be fitted. See `dlm` for details.
`data`	a model-frame containing the data for each term in the model. Should already be appropriately subset, etc.
`.names.func`	a function for creating names of dummy variables that act as placeholders for penalized spline terms in **lme4**'s setup. There should not be a need to alter this in normal use cases

Details

Users should not typically have to interact with `interpret.dlm` directly, but it may be useful for extensions.

Uses `R`'s `stats::model.matrix` mechanisms to build and parse the random effects (or penalized) components of spline-lag terms in the model. The object returned is later passed to other **dlmBE** functions in order to fit the specified model.

Value

an S3 object of class `"parsed.dlm"` with list elements:

`formula` the formula passed to `interpret.dlm`

`lme4.formula` a reconstructed formula that is then passed to the **lme4** `modular` functions

`model` a `data.frame` returned by call to `stats::model.frame`; a copy of the fixed effects data needed to fit the model

`Bt` a matrix of the random or penalized lag basis vectors, where each vector is a row. Stored as an object that inherits from `Matrix::dMatrix`

`bases` a list of all the unique bases represented in the `formula`. This may be $\leq$ the number of separate spline-lag terms. All elements should inherit from `SmoothLag`

`lag.group` an integer vector returned by `parse.names` where each unique integer corresponds to a separate spline-lag term. For lag term `i`, `lag.group == i` indexes the rows of `Bt` that correspond to the set of random or penalized basis vectors for that term

`bi` for "basis index." Each set of lag terms indexed in `lag.group` has a matching basis decomposition in `bases`. `bi` keeps track of that matching

---

`LagBasis` *Create and manipulate lag basis functions*

Description

S4 class object to store and query components of computed lag basis functions. User interface for creating this class can be found in `basis`.

Details

See `basis` for details of the decomposition

Slots

- `x` radii that define ring-shaped areas around participant locations
- `x.center` the value `x` was centered to prior to computing the basis
- `x.scale` the value `x` was scaled by prior to computing the basis
- `C0` C_0 part of basis matrix
- `K1` K_1 part of basis matrix
- `dist.fun` the function applied to the elements of `x` to compute the basis

References

Rupert D, Wand MP, & Carroll RJ (2003) Semiparametric Regression. New York: Cambridge University Press.

`lagIndex` *Extract list of indices of lag terms*

Description

Intended for use in conjunction with the `estimands` functions. For each set of distributed lag terms in the model, find and return the integer indices that would extract the corresponding coefficients from the vector returned by `vcoef`.

Usage

```
lagIndex(object, ...)

## S4 method for signature 'dlMod'
lagIndex(object, .fixed = TRUE, ...)
```

Arguments

`object`	a fitted model object
`.fixed`	a logical flag (default `.fixed = TRUE`) to indicate whether the fixed/unpenalized DL term coefficient indices should be included or not.

See Also

`vcoef`

Examples

```
data (simdata)

## Setup distance count matrix and corresponding lag distances
X <- as.matrix(simdata[, -(1:3)])
lag <- seq(0.1, 10, length.out = ncol(X))

fit <- dlm(Y ~ Age + Gender + cr(lag, X), data = simdata)
vcoef(fit)[lagIndex(fit)[[1]]]
```

`lme4.dlm` *Fit distributed lag models using lme4*

Description

Fits an interpreted distributed lag model using **lme4** `modular` functions

Usage

```
lme4.dlm(parsed, family = gaussian(), control = list(), REML = FALSE, ...)
```

Arguments

`parsed`	an interpreted `dlm` formula object returned by `interpret.dlm`
`family`	a description of the error distribution and link function to be used in the model. The default is `gaussian(link = "identity")`. See `stats::family` for possible family functions and details
`control`	either a `list` object with arguments to be passed to the `lme4::lmerControl` sequence, or the output of `[g]lmerConrol` directly
`REML`	if `TRUE` and a linear `dlm` model is specified, `lme4.dlm` will use REML to fit the model. MLE will be use otherwise
`...`	other parameters to be passed to the **lme4** `modular` family functions

Details

Together with `interpret.dlm`, this function does the main grunt work for `dlm`. Given an interpreted model, `lme4.dlm` organizes the parsed data into the **lme4** `modular` functions to fit the model and returns the fit as an **lme4** object.

Value

an object that inherits from `lme4::merMod` containing a fitted model

`makeDlMod` *Convert fitted models to 'dlMod' objects*

Description

Convert an appropriately fit model object into an object of class `dlMod`

Usage

```
## S3 method for class 'merMod'
makeDlMod(object, parsed, call, ...)
```

Arguments

`object`	a fitted model object
`parsed`	an interpreted `dlm` formula object returned by `interpret.dlm`
`call`	an optional matched function call

Value

an S4 object of class dlMod

---

omega *Extract lag basis matrix*

Description

Extract lag basis matrix, $\Omega = [C_0, K_1]$. See the definition below (which is borrowed from basis)

Usage

```
omega(object, ...)

## S4 method for signature 'LagBasis'
omega(object, ...)

## S4 method for signature 'SmoothLag'
omega(object, ...)
```

Arguments

object	An object storing details of the basis decomposition
...	additional arguments

Value

A square numeric matrix

Methods (by class)

- LagBasis: Method for "LagBasis" objects
- SmoothLag: Method for "LagBasis" objects

Decomposition

Once a basis function ($\delta(\cdot, \cdot)$) and radii (r) are chosen, define the matrix, $C_{1,ij} = \delta(r_i, r_j)$, and let,

$$C_0 = [1_L, r]$$

$$C_1 = QR$$

$$M_1 = Q_{-(1:2)}$$

$$K_1 = C_1 M_1 (M_1^T C_1 M_1)^{-\frac{1}{2}}$$

where A_{-j} denotes a matrix A with column(s) j removed. Then the (scaled) distributed lag effects are $\beta = C_0\alpha + K_1 b$, where $b_\ell \sim \mathrm{N}(0, \sigma_b^2)$, for $\ell = 1, \ldots, L-2$.

parse.names *Partition DL term names*

Description

Parse variable names and create a unique integer label for each unique use of a matching base case. `parse.names` is used internally by several **dlmBE** functions to determine which distributed lag terms share the same random effects variance/penalty.

Usage

```
parse.names(base, names, .warn = TRUE)
```

Arguments

`base`	a character vector with literal matches in `names`
`names`	a character vector with substrings that are literal matches to elements in `base`

Value

an S3 object of class `"pnames"` which is of `length` equal to `length(names)` and stores a unique integer label for each matching case in `base` and a look up dictionary for each integer.

Indexing in `pnames` objects takes a character string argument and returns the vector indices of the matching `base` uses. See Examples.

Examples

```
lg <- 1:5
Z <- matrix(rbinom(60, 10, 0.1), 12)
group <- rep(1:4, each = 3)
base <- "cr(lg, Z)"
names <- colnames(model.matrix(~ cr(lg, Z):factor(group)))
pn <- parse.names(base, names, FALSE)
names
pn
pn[""]
pn["cr(lg, Z):factor(group)2"]
```

plotDlm *Plots of lag terms*

Description

Plot estimated lag coefficients and confidence intervals

Usage

```
## S3 method for class 'dlMod'
plot(x, geom = c("pointrange", "line"), level = 0.95, scaled = TRUE, ...)
```

Arguments

`x`	a fitted model object that inherits from `dlMod`
`level`	a desired confidence interval level
`scaled`	whether or not lag coefficients should be scaled (see `estimands`)
`geom`	a **ggplot2** graph geometry. See Details
`...`	additional graphical parameters to be passed to `ggplot2::facet_wrap`

Details

Plots estimated lag coefficients and confidence intervals to allow convenient visual inspection of effects over different radii. Point estimates and confidence intervals are computed using `confint`, and plots are rendered using minimal **ggplot2** commands so that the resulting graphics objects are largely customizable.

For now, the only supported plot geometries correspond to `geom_pointrange` and `geom_line`, and show the estimated functions of Lag radii as either point and interval estimates, or continuous functions with confidence bands, respectively. The default option is `geom = "pointrange"` for point and interval estimates.

Value

a **ggplot2** graphic object

`predict.LagBasis` *Predict new values for fitted lag basis*

Description

Will update w/ description. Not yet implemented

Usage

```
## S3 method for class 'LagBasis'
predict(object, x, ...)
```

Arguments

`object`	A `LagBasis` object

scaleMat *Extract distributed lag scale matrix*

Description

Return lag coefficient scale matrix, S. S should be invertable. Typically users should only interact with scaleMat indirectly through the estimands functions (with the argument scaled = TRUE).

Usage

```
scaleMat(object, ...)

## S4 method for signature 'dlMod'
scaleMat(object, ...)
```

Arguments

object	a fitted model object

Details

If θ is the vector of unscaled regression coefficients, then corresponding distributed lag terms in the vector $S\theta$ can be interpreted as average regression coefficients up to their associated radius. Non-DL coefficients are unchanged by this transformation.

Value

A square numeric matrix represented as a Matrix::sparseMatrix object

simdata *Simulated built environment data*

Description

Simulated built environment data

Usage

```
data(simdata)
```

Format

The first column (Y) is the outcome variable (intended to mimic negative BMI score), covariates Gender (binary) and Age (years) come next, and are followed by 100 location description variables each corresponding to a lag radius equal to the values in seq(0.1, 10, 0.1). These 100 location variables store the counts of simulated built environment features within each radius for each participant.

smoothing *Basis smoothing*

Description

Constructs a set of basis vectors for distances between distributed lag points, and apply as a linear transformation of a concentration matrix.

Usage

```
cr(x, Z, ...)

sm(x, Z, ..., .fun = NULL)
```

Arguments

x	a vector of values to construct the basis from. Missing values are not allowed
Z	a covariate matrix (or object that can be coerced to a `matrix`) to apply the linear basis transformation to. `length(x)` should be the same as `ncol(Z)`
...	arguments to be passed to `basis`

Details

These functions are little more than convenient wrappers to the function `basis` and the `SmoothLag` class constructor. They are intended to simplify the task of specifying lag terms in a model `formula`. The functions compute a set of basis vectors for parameter `x` and applies this basis as a linear transformation of the covariate/concentration matrix parameter, `Z`. For example, if `cr` is used and `Z` is the identity matrix, the model fit will simply be the natural cubic spline of `x`.

Note that other basis extensions should always return an object that inherits from `SmoothLag`

Value

An S4 object of class `SmoothLag`.

Functions

- `cr`: natural cubic radial basis spline
- `sm`: user-defined smoothing

References

Rupert D, Wand MP, & Carroll RJ (2003) Semiparametric Regression. New York: Cambridge University Press.

See Also

`basis`, `SmoothLag`

Examples

```
## load simulated data set and extract concentration matrix
data (simdata)
Conc <- simdata[, -(1:3)]  # First columns are Y, Age, and Gender

## radial lag (distance) each concentration was measured at
x <- seq(0.1, 10, length.out = ncol(Conc))
crb <- cr(x, Conc)
```

SmoothLag *Lag matrix with applied smoothing*

Description

An S4 object for representing lag covariates and storing details about the basis smoothing. Intended for use within the `dlm` modeling framework to assist extraction of basis components treated as "fixed" and "random" effects. Inherits from `matrix`.

Slots

`basis` A `LagBasis` smoothing object containing details about the lag and the smoothing parameters used

`.Data` Contains the "fixed effects" components of the smoothed lag function. This scheme is intended to work conveniently with `stats::model.matrix`

`random` Contains the random effects or penalized components of the smoothed lag function

`signature` Character vector containing the function name and deparsed arguments from whatever smoothing function generated the `SmoothLag` object

Index

∗Topic **datasets**
simdata, 16

basis, 3, *10*, *13*, *17*

changePoint, 5
cholfVar, 5
cholfVar,merMod-method *(*cholfVar*)*, 5
coef.dlMod *(*estimands*)*, 8
coef.merMod, *9*
confint, *5*, *9*, *15*
confint.dlMod *(*estimands*)*, 8
cr, *3*, *4*, *7*
cr *(*smoothing*)*, 17

dlm, *3*, *4*, 6, *10*, *12*, *18*
dlmBE-package, 2
dlMod, *7*, *8*, 8, *9*, *12*, *13*, *15*
dMatrix, *10*

estimands, 8, *11*, *15*, *16*

facet_wrap, *15*
family, *7*, *12*
formula, *6*

geom_line, *15*
geom_pointrange, *15*

interpret.dlm, 9, *12*

LagBasis, *4*, *8*, 10, *13*, *15*, *18*
lagIndex, *5*, 11
lFormula, *7*
lme4.dlm, 12
lmer, *7*
lmerControl, *7*, *12*
lmResp, *8*

makeDlMod, 12
merMod, *7*, *8*, *12*
model.frame, *10*
model.matrix, *10*, *18*
modular, *10*, *12*

na.omit, *6*

omega, 13
omega,LagBasis-method *(*omega*)*, 13
omega,SmoothLag-method *(*omega*)*, 13
options, *6*

package documentation, *9*
parse.names, *10*, 14
plotDlm, 14
predict.LagBasis, 15

scaleMat, 16
Sigma *(*estimands*)*, 8
Sigma,dlMod-method *(*estimands*)*, 8
simdata, 16
sm *(*smoothing*)*, 17
smoothing, 17
SmoothLag, *7*, *10*, *17*, 18
sparseMatrix, *6*, *16*

vcoef, *11*
vcoef *(*estimands*)*, 8
vcoef,dlMod-method *(*estimands*)*, 8
vcoef0 *(*estimands*)*, 8
vcoef0,dlMod-method *(*estimands*)*, 8