Package ‘dlmtree’

May 31, 2024

Type Package

Title Bayesian Treed Distributed Lag Models

Version 1.0.0

Description Estimation of distributed lag models (DLMs) based on a Bayesian additive regression trees framework. Includes several extensions of DLMs: treed DLMs and distributed lag mixture models (Mork and Wilson, 2023) <doi:10.1111/biom.13568>; treed distributed lag nonlinear models (Mork and Wilson, 2022) <doi:10.1093/biostatistics/kxaa051>; heterogeneous DLMs (Mork, et. al., 2024) <doi:10.1080/01621459.2023.2258595>; monotone DLMs (Mork and Wilson, 2024) <doi:10.1214/23-BA1412>. The package also includes visualization tools and a 'shiny' interface to help interpret results.

License GPL (>= 3)

Encoding UTF-8

LazyData false

Depends R (>= 3.5.0)

Imports Rcpp (>= 1.0.4), dplyr, ggplot2, shiny, shinythemes, tidyr, mgcv

LinkingTo Rcpp, RcppArmadillo, RcppEigen

RoxygenNote 7.3.1

SystemRequirements C++11

URL `https://github.com/danielmork/dlmtree`, `https://danielmork.github.io/dlmtree/`

BugReports `https://github.com/danielmork/dlmtree/issues`

NeedsCompilation yes

Author Daniel Mork [aut, cre, cph] (<`https://orcid.org/0000-0002-7924-0706`>), Seongwon Im [aut] (<`https://orcid.org/0009-0000-8447-5852`>), Ander Wilson [aut] (<`https://orcid.org/0000-0003-4774-3883`>)

Maintainer Daniel Mork <`dmork@hsph.harvard.edu`>

Repository CRAN

Date/Publication 2024-05-31 13:50:18 UTC

R topics documented:

`adj_coexposure` *Adjusting for expected changes in co-exposure (TDLMM)*

Description

Estimates the marginal effects of an exposure while accounting for expected changes in co-occurring exposures at the same time point. Values of co-occurring exposures are modeled nonlinearly using a spline model with predictions made at the lower an upper values for the exposure of interest.

Usage

```
adj_coexposure(
  exposure.data,
  object,
  contrast_perc = c(0.25, 0.75),
  contrast_exp = list(),
  conf.level = 0.95,
  keep.mcmc = FALSE,
  verbose = TRUE
)
```

Arguments

`exposure.data`	Named list of exposure matrices used as input to TDLMM.
`object`	Model output for TDLMM from dlmtree() function.
`contrast_perc`	2-length vector of percentiles or named list corresponding to lower and upper exposure percentiles of interest. Names must equal list names in 'exposure.data'.
`contrast_exp`	Named list consisting lower and upper exposure values. This takes precedence over contrast_perc if both inputs are used.
`conf.level`	Confidence level used for estimating credible intervals. Default is 0.95.
`keep.mcmc`	If TRUE, return posterior samples.
`verbose`	TRUE (default) or FALSE: print output

Details

adj_coexposure

Value

data.frame of plot data with exposure name, posterior mean, and credible intervals, or posterior samples if keep.mcmc = TRUE

`coExp` *Randomly sampled exposure from Colorado counties*

Description

Matrix of five different exposures, each measured over 40 weeks.

Usage

```
data(coExp)
```

Format

matrix

Source

`https://aqs.epa.gov/aqsweb/airdata/download_files.html`

References

`https://www.epa.gov/outdoor-air-quality-data`

`combine.models` *Combines information from DLMs of single exposure*

Description

Method for combining information from DLMs of single exposure

Usage

```
combine.models(mlist)
```

Arguments

`mlist`	a list of models

Details

combine.models

Value

A data frame with model fit information of the models included in the list

`combine.models.tdlmm` *Combines information from DLMs of mixture exposures.*

Description

Method for combining information from DLMs of mixture exposures.

Usage

```
combine.models.tdlmm(mlist)
```

Arguments

`mlist`	a list of models

Details

combine.models.tdlmm

Value

A data frame with model fit information of the models included in the list `cppIntersection` *fast set intersection tool assumes sorted vectors A and B*

Description

fast set intersection tool assumes sorted vectors A and B

Usage

```
cppIntersection(A, B)
```

Arguments

`A`	sorted integer vector A
`B`	sorted integer vector B

Value

vector of resulting intersection

`dlmEst` *Calculates the distributed lag effect with DLM matrix for linear models.*

Description

Calculates the distributed lag effect with DLM matrix for linear models.

Usage

```
dlmEst(dlm, nlags, nsamp)
```

Arguments

`dlm`	A numeric matrix containing the model fit information
`nlags`	total number of lags
`nsamp`	number of mcmc iterations

Value

A cube object of lag effect x lag x mcmc dlmtree *Fit tree structured distributed lag models*

Description

The 'dlmtree' function accommodates various response variable types, including continuous, binary, and zero-inflated count values. The function is designed to handle both single exposure and exposure mixtures. For a single exposure, users are offered options to model non-linear effects (tdlnm), linear effects (tdlm), or heterogeneous subgroup/individualized effects (hdlm). In the case of exposure mixtures, the function supports lagged interactions (tdlmm), and heterogeneous subgroup/individualized effects (hdlmm) allowing for a comprehensive exploration of mixture exposure heterogeneity. Additionally, users can fine-tune parameters to impose effect shrinkage and perform exposure selection, enhancing the adaptability and precision of the modeling process. For more detailed documentation, visit: dlmtree website.

Usage

```
dlmtree(
  formula,
  data,
  exposure.data,
  dlm.type = "linear",
  family = "gaussian",
  mixture = FALSE,
  het = FALSE,
  n.trees = 20,
  n.burn = 1000,
  n.iter = 2000,
  n.thin = 2,
  shrinkage = "all",
  dlmtree.params = c(0.95, 2),
  dlmtree.step.prob = c(0.25, 0.25),
  binomial.size = 1,
  formula.zi = NULL,
  tdlnm.exposure.splits = 20,
  tdlnm.time.split.prob = NULL,
  tdlnm.exposure.se = NULL,
  hdlm.modifiers = "all",
  hdlm.modifier.splits = 20,
  hdlm.modtree.params = c(0.95, 2),
  hdlm.modtree.step.prob = c(0.25, 0.25, 0.25),
  hdlm.dlmtree.type = "shared",
  hdlm.selection.prior = 0.5,
  mixture.interactions = "noself",
  mixture.prior = 1,
  monotone.gamma0 = NULL,
  monotone.sigma = NULL,
  monotone.tree.time.params = c(0.95, 2),
  monotone.tree.exp.params = c(0.95, 2),
  monotone.time.kappa = NULL,
  subset = NULL,
  lowmem = FALSE,
  verbose = TRUE,
  save.data = TRUE,
  diagnostics = FALSE,
  initial.params = NULL
)
```

Arguments

`formula`	object of class formula, a symbolic description of the fixed effect model to be fitted, e.g. y ~ a + b.
`data`	data frame containing variables used in the formula.
`exposure.data`	numerical matrix of exposure data with same length as data, for a mixture setting (tdlmm, hdlmm): named list containing equally sized numerical matrices of exposure data having same length as data.
`dlm.type`	dlm model specification: "linear" (default), "nonlinear", "monotone".
`family`	'gaussian' for continuous response, 'logit' for binomial, 'zinb' for zero-inflated negative binomial.
`mixture`	flag for mixture, set to TRUE for tdlmm and hdlmm. (default: FALSE)
`het`	flag for heterogeneity, set to TRUE for hdlm and hdlmm. (default: FALSE)
`n.trees`	integer for number of trees in ensemble.
`n.burn`	integer for length of MCMC burn-in.
`n.iter`	integer for number of MCMC iterations to run model after burn-in.
`n.thin`	integer MCMC thinning factor, i.e. keep every tenth iteration.
`shrinkage`	character "all" (default), "trees", "exposures", "none", turns on horseshoe-like shrinkage priors for different parts of model.
`dlmtree.params`	numerical vector of alpha and beta hyperparameters controlling dlm tree depth. (default: alpha = 0.95, beta = 2)
`dlmtree.step.prob`	numerical vector for probability of each step for dlm tree updates: 1) grow/prune, 2) change, 3) switch exposure. (default: c(0.25, 0.25, 0.25))
`binomial.size`	integer type scalar (if all equal, default: 1) or vector defining binomial size for 'logit' family.
`formula.zi`	(only applies to family = 'zinb') object of class formula, a symbolic description of the fixed effect of zero-inflated (ZI) model to be fitted, e.g. y ~ a + b. This only applies to ZINB where covariates for ZI model are different from NB model. This is set to the argument 'formula' by default.
`tdlnm.exposure.splits`	scalar indicating the number of splits (divided evenly across quantiles of the exposure data) or list with two components: 'type' = 'values' or 'quantiles', and 'split.vals' = a numerical vector indicating the corresponding exposure values or quantiles for splits.

`tdlnm.time.split.prob`	probability vector of a spliting probabilities for time lags. (default: uniform probabilities)
`tdlnm.exposure.se`	numerical matrix of exposure standard errors with same size as exposure.data or a scalar smoothing factor representing a uniform smoothing factor applied to each exposure measurement. (default: sd(exposure.data)/2)
`hdlm.modifiers`	string vector containing desired modifiers to be included in a modifier tree. The strings in the vector must match the names of the columns of the data. By default, a modifier tree considers all covariates in the formula as modifiers unless stated otherwise.
`hdlm.modifier.splits`	integer value to determine the possible number of splitting points that will be used for a modifier tree.
`hdlm.modtree.params`	numerical vector of alpha and beta hyperparameters controlling modifier tree depth. (default: alpha = 0.95, beta = 2)
`hdlm.modtree.step.prob`	numerical vector for probability of each step for modifier tree updates: 1) grow, 2) prune, 3) change. (default: c(0.25, 0.25, 0.25))
`hdlm.dlmtree.type`	specification of dlmtree type for HDLM: shared (default) or nested.
`hdlm.selection.prior`	scalar hyperparameter for sparsity of modifiers. Must be between 0.5 and 1. Smaller value corresponds to increased sparsity of modifiers.
`mixture.interactions`	'noself' (default) which estimates interactions only between two different exposures, 'all' which also allows interactions within the same exposure, or 'none' which eliminates all interactions and estimates only main effects of each exposure.
`mixture.prior`	positive scalar hyperparameter for sparsity of exposures. (default: 1)
`monotone.gamma0`	vector (with length equal to number of lags) of means for logit-transformed prior probability of split at each lag; e.g., gamma_0l = 0 implies mean prior probability of split at lag l = 0.5.
`monotone.sigma`	symmetric matrix (usually with only diagonal elements) corresponding to gamma_0 to define variances on prior probability of split; e.g., gamma_0l = 0 with lth diagonal element of sigma=2.701 implies that 95% of the time the prior probability of split is between 0.005 and 0.995, as a second example setting gamma_0l=4.119 and the corresponding diagonal element of sigma=0.599 implies that 95% of the time the prior probability of a split is between 0.8 and 0.99.
`monotone.tree.time.params`	numerical vector of hyperparameters for monotone time tree.
`monotone.tree.exp.params`	numerical vector of hyperparameters for monotone exposure tree.

`monotone.time.kappa`	scaling factor in dirichlet prior that goes alongside `tdlnm.time.split.prob` to control the amount of prior information given to the model for deciding probabilities of splits between adjacent lags.
`subset`	integer vector to analyze only a subset of data and exposures.
`lowmem`	TRUE or FALSE (default): turn on memory saver for DLNM, slower computation time.
`verbose`	TRUE (default) or FALSE: print output
`save.data`	TRUE (default) or FALSE: save data used for model fitting. This must be set to TRUE to use shiny() function on hdlm or hdlmm
`diagnostics`	TRUE or FALSE (default) keep model diagnostic such as the number of terminal nodes and acceptance ratio.
`initial.params`	initial parameters for fixed effects model, FALSE = none (default), "glm" = generate using GLM, or user defined, length must equal number of parameters in fixed effects model.

Details

dlmtree

Model is recommended to be run for at minimum 5000 burn-in iterations followed by 15000 sampling iterations with a thinning factor of 5. Convergence can be checked by re-running the model and validating consistency of results. Examples are provided below for the syntax for running different types of models. For more examples, visit: dlmtree website.

Value

Object of one of the classes: tdlm, tdlmm, tdlnm, hdlm, hdlmm

Examples

```


# The first three examples are for one lagged exposure


# treed distributed lag model (TDLM)
# binary outcome with logit link

D <- sim.tdlmm(sim = "A", mean.p = 0.5, n = 1000)
tdlm.fit <- dlmtree(y ~ .,
                    data = D$dat,
                    exposure.data = D$exposures[[1]],
                    dlm.type = "linear",
                    family = "logit",
                    binomial.size = 1)

# summarize results
tdlm.sum <- summary(tdlm.fit)
tdlm.sum
```

```
# plot results
plot(tdlm.sum)



# Treed distributed lag nonlinear model (TDLNM)
# Gaussian regression model
D <- sim.tdlnm(sim = "A", error.to.signal = 1)
tdlnm.fit <- dlmtree(formula = y ~ .,
                     data = D$dat,
                     exposure.data = D$exposures,
                     dlm.type = "nonlinear",
                     family = "gaussian")

# summarize results
tdlnm.sum <- summary(tdlnm.fit)
tdlnm.sum

# plot results
plot(tdlnm.sum)



# Heterogenious TDLM (HDLM), similar to first example but with heterogenious exposure response
D <- sim.hdlmm(sim = "B", n = 1000)
hdlm.fit <- dlmtree(y ~ .,
                    data = D$dat,
                    exposure.data = D$exposures,
                    dlm.type = "linear",
                    family = "gaussian",
                    het = TRUE)

# summarize results
hdlm.sum <- summary(hdlm.fit)
hdlm.sum

# shiny app for HDLM
if (interactive()) {
  shiny(hdlm.fit)
}



# The next two examples are for a mixture (or multivariate) exposure


# Treed distributed lag mixture model (TDLMM)
# Model for mixutre (or multivariate) lagged exposures
# with a homogenious exposure-time-response function
D <- sim.tdlmm(sim = "B", error = 25, n = 1000)
tdlmm.fit <- dlmtree(y ~ .,
                     data = D$dat, exposure.data = D$exposures,
                     mixture.interactions = "noself",
                     dlm.type = "linear", family = "gaussian",
                     mixture = TRUE)

# summarize results
tdlmm.sum <- summary(tdlmm.fit)

# plot the marginal exposure-response for one exposure
plot(tdlmm.sum, exposure1 = "e1")

# plot exposure-response surface
plot(tdlmm.sum, exposure1 = "e1", exposure2 = "e2")



# heterogenious version of TDLMM
D <- sim.hdlmm(sim = "D", n = 1000)
hdlmm.fit <- dlmtree(y ~ .,
                     data = D$dat,
                     exposure.data = D$exposures,
                     dlm.type = "linear",
                     family = "gaussian",
                     mixture = TRUE,
                     het = TRUE)

# summarize results
hdlmm.sum <- summary(hdlmm.fit)
hdlmm.sum

# summarize results
if (interactive()) {
  shiny(hdlmm.fit)
}
```

dlmtreeGPFixedGaussian

dlmtree model with fixed Gaussian process approach

Description

dlmtree model with fixed Gaussian process approach

Usage

```
dlmtreeGPFixedGaussian(model)
```

Arguments

`model`	A list of parameter and data contained for the model fitting

Value

A list of dlmtree model fit, mainly posterior mcmc samples

`dlmtreeGPGaussian` *dlmtree model with Gaussian process approach*

Description

dlmtree model with Gaussian process approach

Usage

```
dlmtreeGPGaussian(model)
```

Arguments

`model`	A list of parameter and data contained for the model fitting

Value

A list of dlmtree model fit, mainly posterior mcmc samples

`dlmtreeHDLMGaussian` *dlmtree model with shared HDLM approach*

Description

dlmtree model with shared HDLM approach

Usage

```
dlmtreeHDLMGaussian(model)
```

Arguments

`model`	A list of parameter and data contained for the model fitting

Value

A list of dlmtree model fit, mainly posterior mcmc samples

`dlmtreeHDLMMGaussian` *dlmtree model with HDLMM approach*

Description

dlmtree model with HDLMM approach

Usage

```
dlmtreeHDLMMGaussian(model)
```

Arguments

`model`	A list of parameter and data contained for the model fitting

Value

A list of dlmtree model fit, mainly posterior mcmc samples

`dlmtreeTDLMFixedGaussian` *dlmtree model with fixed Gaussian approach*

Description

dlmtree model with fixed Gaussian approach

Usage

```
dlmtreeTDLMFixedGaussian(model)
```

Arguments

`model`	A list of parameter and data contained for the model fitting

Value

A list of dlmtree model fit, mainly posterior mcmc samples

`dlmtreeTDLMNestedGaussian` *dlmtree model with nested Gaussian approach*

Description

dlmtree model with nested Gaussian approach

Usage

```
dlmtreeTDLMNestedGaussian(model)
```

Arguments

`model`	A list of parameter and data contained for the model fitting

Value

A list of dlmtree model fit, mainly posterior mcmc samples

`dlmtreeTDLM_cpp` *dlmtree model with nested HDLM approach*

Description

dlmtree model with nested HDLM approach

Usage

```
dlmtreeTDLM_cpp(model)
```

Arguments

`model`	A list of parameter and data contained for the model fitting

Value

A list of dlmtree model fit, mainly posterior mcmc samples

`dlnmEst` *Calculates the distributed lag effect with DLM matrix for non-linear models.*

Description

Calculates the distributed lag effect with DLM matrix for non-linear models.

Usage

```
dlnmEst(dlnm, predAt, nlags, nsamp, center, se)
```

Arguments

`dlnm`	A numeric matrix containing the model fit information
`predAt`	Number of splits in the model
`nlags`	total number of lags
`nsamp`	number of mcmc iterations
`center`	center parameter
`se`	Standard error parameter

Value

A cube object of lag effect x lag x mcmc

`dlnmPLEst` *Calculates the distributed lag effect with DLM matrix for non-linear models.*

Description

Calculates the distributed lag effect with DLM matrix for non-linear models.

Usage

```
dlnmPLEst(dlnm, predAt, nlags, nsamp, center)
```

Arguments

`dlnm`	A numeric matrix containing the model fit information
`predAt`	Number of splits in the model
`nlags`	total number of lags
`nsamp`	number of mcmc iterations
`center`	center parameter

Value

A cube object of lag effect x lag x mcmc

---

drawTree *Draws a new tree structure*

Description

A recursive method for drawing a new tree structure

Usage

```
drawTree(depth, alpha, beta)
```

Arguments

`depth`	depth of a tree
`alpha`	tree shape parameter, 0 < alpha < 1
`beta`	tree size parameter, beta > 0

Details

drawTree

Value

A integer value of number of terminal nodes

---

estDLM *Calculates subgroup-specific lag effects for heterogeneous models*

Description

Method for calculating subgroup-specific lag effects for heterogeneous models: HDLM, HDLMM

Usage

```
estDLM(
  object,
  new.data,
  group.index,
  conf.level = 0.95,
  exposure = NULL,
  return.mcmc = FALSE,
  mem.safe = FALSE,
  verbose = TRUE
)
```

Arguments

`object`	an object of a model fit. Must be 'hdlm' or 'hdlmm'
`new.data`	a data frame with new observations with the same number of modifiers
`group.index`	a list of index (row numbers) for subgroup specification
`conf.level`	confidence level for credible interval of effects
`exposure`	exposure of interest for 'hdlmm' method
`return.mcmc`	store mcmc in the output
`mem.safe`	boolean memory parameter for rule index
`verbose`	TRUE (default) or FALSE: print output

Details

estDLM

Value

A list of distributed lag effects per subgroups

`exposureCov` *Exposure covariance structure*

Description

Matrix containing pairwise covariances for real exposure data consisting of five different exposures, each measured over 37 weeks.

Usage

```
data(exposureCov)
```

Format

matrix

Source

https://aqs.epa.gov/aqsweb/airdata/download_files.html

References

https://www.epa.gov/outdoor-air-quality-data

`get_sbd_dlmtree` *Download simulated data for dlmtree articles*

Description

Download simulated data for dlmtree articles

Usage

```
get_sbd_dlmtree()
```

Value

A data frame with 10000 rows (observations) and 202 variables. All data is simulated. The variables are:

`bwgaz`	Outcome to be used. Simulated birth weight for gestational age z-score.
`ChildSex`	Binary sex of child.
`MomAge`	Continuous age in years.
`GestAge`	Continuous estimated gestational age at birth in weeks.
`MomHeightIn`	Continuous maternal height in inches.
`MomPriorWeightLbs`	Continuous mothers pre-pregnancy weight in pounds.
`MomPriorBMI`	Continuous mothers pre-pregnancy BMI.
`race`	Categorical race.
`Hispanic`	Binary indicator of Hispanic.
`MomEdu`	Categorical maternal highest educational attainment.
`SmkAny`	Binary indicator of any smoking during pregnancy.
`Marital`	Categorical maternal marital status.
`Income`	Categorical income.
`EstDateConcept`	Estimated date of conception.
`EstMonthConcept`	Estimated month of conception.
`EstYearConcept`	Estimated year of conception.
`pm25_1 - pm25_37`	Weekly average exposure to PM2.5 for weeks 1 to 37.
`no2_1 - no2_37`	Weekly average exposure to NO2 for weeks 1 to 37.
`so2_1 - so2_37`	Weekly average exposure to SO2 for weeks 1 to 37.
`co2_1 - co2_37`	Weekly average exposure to CO for weeks 1 to 37.
`temp_1 - temp_37`	Weekly average exposure to temperature for weeks 1 to 37.
`source`	Variable indicating that the data came from the bdlim package.

Examples

```
sbd_dlmtree <- get_sbd_dlmtree()
```

---

mixEst *Calculates the lagged interaction effects with MIX matrix for linear models.*

Description

Calculates the lagged interaction effects with MIX matrix for linear models.

Usage

```
mixEst(dlm, nlags, nsamp)
```

Arguments

`dlm`	A numeric matrix containing the model fit information
`nlags`	total number of lags
`nsamp`	number of mcmc iterations

Value

A cube object of interaction effect x lag x mcmc

---

monotdlnm_Cpp *dlmtree model with monotone tdlnm approach*

Description

dlmtree model with monotone tdlnm approach

Usage

```
monotdlnm_Cpp(model)
```

Arguments

`model`	A list of parameter and data contained for the model fitting

Value

A list of dlmtree model fit, mainly posterior mcmc samples

pip — *Calculates posterior inclusion probabilities (PIPs) for modifiers in HDLM & HDLMM*

Description

Method for calculating posterior inclusion probabilities (PIPs) for modifiers in HDLM & HDLMM

Usage

```
pip(object, type = 1)
```

Arguments

`object`	An object of class dlmtree.
`type`	Type=1 indicates single modifier PIPs. Type=2 indicates joint modifier PIPs for two modifiers.

Details

pip

Value

A vector (type=1) or data.frame (type=2) of PIPs.

Examples

```
# Posterior inclusion probability with HDLM
D <- sim.hdlmm(sim = "B", n = 1000)
fit <- dlmtree(y ~ .,
               data = D$dat,
               exposure.data = D$exposures,
               dlm.type = "linear",
               family = "gaussian",
               het = TRUE)
pip(fit)
pip(fit, type = 2)
```

plot.summary.monotone *Returns variety of plots for model summary of class 'monotone'*

Description

Method for returning variety of plots for model summary of class 'monotone'

Usage

```
## S3 method for class 'summary.monotone'
plot(x, plot.type = "mean", val = c(), time = c(), ...)
```

Arguments

`x`	object of class 'summary.monotone', output of summary of 'monotone'
`plot.type`	string indicating plot type, options are 'mean' (default) which shows mean exposure-time response surface, 'se', 'ci-min', 'ci-max', 'slice' which takes a slice of the plot at a given 'val' or 'time', 'animate' which creates a animation of slices of the surface plot across exposure values (requires package gganimate)
`val`	exposure value for slice plot
`time`	time value for slice plot
`...`	additional parameters to alter plots: 'main', 'xlab', 'ylab', 'flab' which sets the effect label for surface plots, 'start.time' which sets the first time value

Details

plot.summary.monotone

Value

A plot of distributed lag effect estimated with monotone-TDLNM

plot.summary.tdlm *Plots a distributed lag function for model summary of 'tdlm'*

Description

Method for plotting a distributed lag function for model summary of 'tdlm'

Usage

```
## S3 method for class 'summary.tdlm'
plot(x, trueDLM = NULL, ...)
```

Arguments

`x`	object of class 'summary.tdlm', output of summary of 'tdlm'
`trueDLM`	A vector of true effects that can be obtained from the simulated data. Only applicable for simulation studies
`...`	additional plotting parameters for title and labels 'start.time' which sets the first time value

Details

plot.summary.tdlm

Value

A plot of distributed lag effect estimated with tdlm

`plot.summary.tdlmm` *Plots DLMMs for model summary of class 'tdlmm'*

Description

Method for plotting DLMMs for model summary of class 'tdlmm'. Includes plots for marginal exposure effects as well as interactions between two exposures.

Usage

```
## S3 method for class 'summary.tdlmm'
plot(
  x,
  type = "marginal",
  exposure1 = NULL,
  exposure2 = NULL,
  time1 = c(),
  time2 = c(),
  show.cw = TRUE,
  cw.plots.only = TRUE,
  trueDLM = NULL,
  scale = NULL,
  ...
)
```

Arguments

`x`	an object of type 'summary.tdlmm' from summary.tdlmm() output
`type`	plot type, 'marginal' (default)
`exposure1`	exposure for plotting DLM

`exposure2`	exposure paired with 'exposure1' for plotting interaction
`time1`	plot a cross section from an interaction plot at specific time for 'exposure1'
`time2`	plot a cross section from an interaction plot at specific time for 'exposure2'
`show.cw`	indicate location of critical windows in interaction plot with red points
`cw.plots.only`	show only plots with critical windows
`trueDLM`	A vector of true effects that can be obtained from the simulated data. Only applicable for simulation studies
`scale`	default = NULL, if scale is not NULL, the effects are exponentiated
`...`	additional plotting parameters for title and labels

Details

plot.summary.tdlmm

Value

A plot of distributed lag effect or interaction surface estimated with tdlmm

---

`plot.summary.tdlnm` *Returns variety of plots for model summary of class 'tdlnm'*

Description

Method for returning variety of plots for model summary of class 'tdlnm'

Usage

```
## S3 method for class 'summary.tdlnm'
plot(x, plot.type = "mean", val = c(), time = c(), ...)
```

Arguments

`x`	object of class 'summary.tdlnm', output of summary of 'tdlnm'
`plot.type`	string indicating plot type, options are 'mean' (default) which shows mean exposure-time response surface, 'se', 'ci-min', 'ci-max', 'slice' which takes a slice of the plot at a given 'val' or 'time', 'animate' which creates a animation of slices of the surface plot across exposure values (requires package gganimate)
`val`	exposure value for slice plot
`time`	time value for slice plot
`...`	additional plotting parameters for title and labels 'flab' which sets the effect label for surface plots, 'start.time' which sets the first time value

Details

plot.summary.tdlnm

Value

A plot of distributed lag effect estimated with tdlnm

`pm25Exposures` *PM2.5 Exposure data*

Description

Data.frame containing a sample of weekly average PM2.5 exposures across a range of states/counties. The PM2.5 data was downloaded from US EPA (https://aqs.epa.gov/aqsweb/airdata/download_files.html) daily data summaries and averaged by week. Forty-week ranges were assess for non-missingness and grouped for this dataset.

Usage

```
data(pm25Exposures)
```

Format

data.frame; columns: S = state, C = city, 1-40 = weekly exposure data

Source

`https://aqs.epa.gov/aqsweb/airdata/download_files.html`

References

`https://www.epa.gov/outdoor-air-quality-data`

`ppRange` *Makes a 'pretty' output of a group of numbers*

Description

Method for making a 'pretty' output of a group of numbers. For example: 2,3,4,5,8,9,12,15,16 becomes 2-5,8-9,12,15-16

Usage

```
ppRange(r)
```

Arguments

`r`	set of integers to make 'pretty'

Details

ppRange

Value

character string of values representing 'r'

`predict.hdlm` *Calculates predicted response for HDLM*

Description

Method for calculating predicted response for HDLM

Usage

```
## S3 method for class 'hdlm'
predict(
  object,
  new.data,
  new.exposure.data,
  ci.level = 0.95,
  type = "response",
  outcome = NULL,
  fixed.idx = list(),
  est.dlm = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

`object`	fitted dlmtree model with class hdlm
`new.data`	new data frame which contains the same covariates and modifiers used to fit HDLM model
`new.exposure.data`	new data frame/list which contains the same length of exposure lags used to fit HDLM model
`ci.level`	credible interval level for posterior predictive distribution
`type`	type of prediction: "response" (default) or "waic". "waic" must be specified with ‘outcome‘ parameter
`outcome`	outcome required for WAIC calculation
`fixed.idx`	fixed index
`est.dlm`	flag for estimating dlm effect
`verbose`	TRUE (default) or FALSE: print output
`...`	additional parameters

Details

predict.hdlm

Value

Posterior predictive distribution draws

`predict.hdlmm` *Calculates predicted response for HDLMM*

Description

Method for calculating predicted response for HDLMM

Usage

```
## S3 method for class 'hdlmm'
predict(
  object,
  new.data,
  new.exposure.data,
  ci.level = 0.95,
  type = "response",
  outcome = NULL,
  fixed.idx = list(),
  est.dlm = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

`object`	fitted dlmtree model with class hdlmm
`new.data`	new data frame which contains the same covariates and modifiers used to fit HDLMM model
`new.exposure.data`	new data frame/list which contains the same length of exposure lags used to fit HDLMM model
`ci.level`	credible interval level for posterior predictive distribution
`type`	type of prediction: "response" (default) or "waic". "waic" must be specified with ‘outcome‘ parameter
`outcome`	outcome required for WAIC calculation
`fixed.idx`	fixed index
`est.dlm`	flag for estimating dlm effect
`verbose`	TRUE (default) or FALSE: print output
`...`	additional parameters

Details

predict.hdlmm

Value

Posterior predictive distribution draws

`print.hdlm` *Print a hdlm Object*

Description

Print a hdlm Object

Usage

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

Arguments

`x`	An object of class hdlm.
`...`	Not used.

Value

Assorted model output.

`print.hdlmm` *Print a hdlmm Object*

Description

Print a hdlmm Object

Usage

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

Arguments

`x`	An object of class hdlmm.
`...`	Not used.

Value

Assorted model output.

`print.monotone` *Print a monotone Object*

Description

Print a monotone Object

Usage

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

Arguments

`x`	An object of class monotone
`...`	Not used.

Value

Assorted model output.

`print.summary.hdlm` *Prints an overview with summary of model class 'hdlm'*

Description

Method for printing an overview with summary of model class 'hdlm'

Usage

```
## S3 method for class 'summary.hdlm'
print(x, digits = 3, cw.only = TRUE, ...)
```

Arguments

`x`	an object of type 'summary.hdlm', result of call to summary.hdlm()
`digits`	integer number of digits to round
`cw.only`	print only results for exposures with critical windows
`...`	additional parameters

Details

print.summary.hdlm

Value

output of hdlm fit in R console

print.summary.hdlmm *Prints an overview with summary of model class 'hdlmm'*

Description

Method for printing an overview with summary of model class 'hdlmm'

Usage

```
## S3 method for class 'summary.hdlmm'
print(x, digits = 3, cw.only = TRUE, ...)
```

Arguments

`x`	an object of type 'summary.hdlmm', result of call to summary.hdlmm()
`digits`	integer number of digits to round
`cw.only`	print only results for exposures with critical windows
`...`	additional parameters

Details

print.summary.hdlmm

Value

output of hdlmm fit in R console

print.summary.monotone *Prints an overview with summary of model class 'monotone'*

Description

Method for printing an overview with summary of model class 'monotone'

Usage

```
## S3 method for class 'summary.monotone'
print(x, digits = 3, ...)
```

Arguments

`x`	an object of type 'summary.monotone', result of call to summary.monotone()
`digits`	integer number of digits to round
`...`	additional parameters

Details

print.summary.monotone

Value

output in R console

`print.summary.tdlm` *Prints an overview with summary of model class 'tdlm'*

Description

Method for printing an overview with summary of model class 'tdlm'

Usage

```
## S3 method for class 'summary.tdlm'
print(x, digits = 3, ...)
```

Arguments

`x`	an object of type 'summary.tdlnm', result of call to summary.tdlnm()
`digits`	integer number of digits to round
`...`	additional parameters

Details

print.summary.tdlm

Value

output of tdlm fit in R console

`print.summary.tdlmm` *Prints an overview with summary of model class 'tdlmm'*

Description

Method for printing an overview with summary of model class 'tdlmm'

Usage

```
## S3 method for class 'summary.tdlmm'
print(x, digits = 3, cw.only = TRUE, ...)
```

Arguments

`x`	an object of type 'summary.tdlmm', result of call to summary.tdlmm()
`digits`	integer number of digits to round
`cw.only`	print only results for exposures with critical windows
`...`	additional parameters

Details

print.summary.tdlmm

Value

output of tdlmm fit in R console

`print.summary.tdlnm` *Prints an overview with summary of model class 'tdlnm'*

Description

Method for printing an overview with summary of model class 'tdlnm'

Usage

```
## S3 method for class 'summary.tdlnm'
print(x, digits = 3, ...)
```

Arguments

`x`	an object of type 'summary.tdlnm', result of call to summary.tdlnm()
`digits`	integer number of digits to round
`...`	additional parameters

Details

print.summary.tdlnm

Value

output of tdlnm fit in R console

print.tdlm — *Print a tdlm Object*

Description

Print a tdlm Object

Usage

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

Arguments

`x`	An object of class tdlm.
`...`	Not used.

Value

Assorted model output.

print.tdlmm — *Print a tdlmm Object*

Description

Print a tdlmm Object

Usage

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

Arguments

`x`	An object of class tdlmm.
`...`	Not used.

Value

Assorted model output.

print.tdlnm *Print a tdlnm Object*

Description

Print a tdlnm Object

Usage

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

Arguments

x	An object of class tdlnm.
...	Not used.

Value

Assorted model output.

rcpp_pgdraw *Multiple draw polya gamma latent variable for var c[i] with size b[i]*

Description

Multiple draw polya gamma latent variable for var c[i] with size b[i]

Usage

```
rcpp_pgdraw(b, z)
```

Arguments

b	vector of binomial sizes
z	vector of parameters

Value

Eigen::VectorXd

`rtmvnorm` *Truncated multivariate normal sampler, mean mu, cov sigma, truncated (0, Inf)*

Description

Truncated multivariate normal sampler, mean mu, cov sigma, truncated (0, Inf)

Usage

```
rtmvnorm(mu, sigma, iter)
```

Arguments

`mu`	vector of mean parameters
`sigma`	covariance matrix
`iter`	number of iterations

Value

VectorXd

`ruleIdx` *Calculates the weights for each modifier rule*

Description

Method for calculating the weights for each modifier rule

Usage

```
ruleIdx(mod, mem.safe = FALSE)
```

Arguments

`mod`	a list of modifier splitting rules
`mem.safe`	boolean memory parameter

Value

A list of weights per rule with modifiers

`scaleModelMatrix` *Centers and scales a matrix*

Description

Method for centering and scaling a matrix

Usage

```
scaleModelMatrix(M)
```

Arguments

`M`	a matrix to center and scale

Details

scaleModelMatrix

Value

a scaled matrix

`shiny` *shiny*

Description

shiny generic function for S3method

Usage

```
shiny(fit)
```

Arguments

`fit`	an object of class hdlm or hdlmm to which S3method is applied

Value

A 'shiny' interface for further analysis on heterogeneous analyses. The interface includes tabs for modifier selection, personalized exposure effects and subgroup-specific effects.

`shiny.hdlm` *Executes a 'shiny' app for HDLM.*

Description

Method for executing a 'shiny' app to provide comprehensive analysis with HDLM. The app includes PIP, split points, individualized & subgroup-specific effects.

Usage

```
## S3 method for class 'hdlm'
shiny(fit)
```

Arguments

`fit`	an object of class 'hdlm'

Details

shiny.hdlm

Value

A 'shiny' app interface

`shiny.hdlmm` *Executes a 'shiny' app for HDLMM.*

Description

Method for executing a 'shiny' app to provide comprehensive analysis with HDLMM. The app includes PIP, split points, individualized & subgroup-specific effects for exposure of interest.

Usage

```
## S3 method for class 'hdlmm'
shiny(fit)
```

Arguments

`fit`	an object of class 'hdlmm'

Details

shiny.hdlmm

Value

A 'shiny' app interface

---

sim.hdlmm *Creates simulated data for HDLM & HDLMM*

Description

Method for creating simulated data for HDLM & HDLMM

Usage

```
sim.hdlmm(
  sim = "A",
  n = 1000,
  error = 1,
  effect.size = 1,
  exposure.data = NULL
)
```

Arguments

`sim`	character (A - E) specifying simulation scenario
`n`	sample size
`error`	positive scalar specifying error variance for Gaussian response
`effect.size`	the effect size of the window of susceptibility
`exposure.data`	exposure data. A matrix of exposure data for simulation A, B, C and a named list of exposure data for simulation D, E

Details

sim.hdlmm

Simulation scenarios:

- Scenario A: Two subgroups with early/late windows determined by continuous and binary modifiers
- Scenario B: Two subgroups with scaled effect determined by a continuous modifier
- Scenario C: No heterogeneity i.e., same effect on all individuals
- Scenario D: Three subgroups with three corresponding exposures. Subgroups are determined by continuous and binary modifiers
- Scenario E: Two subgroups with two exposures. First group is associated with the scaled main effect and lagged interaction while the second group is only associated with the scaled main effect, no interaction.

Value

Simulated data and true parameters

Examples

```
sim.hdlmm(sim = "A", n = 1000)
```

`sim.tdlmm` *Creates simulated data for TDLM & TDLMM*

Description

Method for creating simulated data for TDLM & TDLMM

Usage

```
sim.tdlmm(
  sim = "A",
  n = 5000,
  error = 10,
  mean.p = 0.5,
  prop.active = 0.05,
  expList = NULL,
  r = 1
)
```

Arguments

`sim`	character (A - F) specifying simulation scenario
`n`	sample size for simulation
`error`	positive scalar specifying error variance for Gaussian response
`mean.p`	scalar between zero and one specifying mean probability for simulation scenario A
`prop.active`	proportion of active exposures for simulation scenario C
`expList`	named list of exposure data
`r`	dispersion parameter of negative binomial distribution

Details

sim.tdlmm

Simulation scenarios:

- Scenario A: Binary response with single exposure effect
- Scenario B: Continuous response with main effect of PM2.5 and interaction

- Scenario C: Continuous response to test exposure selection using exposure
- Scenario D: Continuous response to test exposure selection using one exposure of main effect and two interaction effects among four exposures
- Scenario E: Zero-inflated count response with single exposure effect
- Scenario F: Zero-inflated count response with single exposure effect with main effect of PM2.5 and interaction

Value

Simulated data and true parameters

Examples

```
sim.tdlmm(sim = "A", mean.p = 0.5, n = 1000)
```

`sim.tdlnm` *Creates simulated data for TDLNM*

Description

Method for creating simulated data for TDLNM

Usage

```
sim.tdlnm(sim = "A", error.to.signal = 1)
```

Arguments

`sim`	character (A - D) specifying simulation scenario
`error.to.signal`	scalar value setting error: sigma^2/var(f)

Details

sim.tdlnm

Simulation scenarios:

- Scenario A: Piecewise constant effect
- Scenario B: Linear effect
- Scenario C: Logistic effect, piecewise in time
- Scenario D: Logistic effect, smooth in time

Value

Simulated data and true parameters

Examples

```
sim.tdlnm(sim = "A", error.to.signal = 1)
```

---

splitPIP *Calculates the posterior inclusion probability (PIP).*

Description

Calculates the posterior inclusion probability (PIP).

Usage

```
splitPIP(dlnm, nlags, niter)
```

Arguments

`dlnm`	A numeric matrix containing the model fit information
`nlags`	total number of lags
`niter`	number of mcmc iterations

Value

A matrix of split counts per mcmc

---

splitpoints *Determines split points for continuous modifiers*

Description

Method for determining split points for continuous modifiers

Usage

```
splitpoints(object, var, round = NULL)
```

Arguments

`object`	An object of class dlmtree with DLM type hdlm & hdlmm
`var`	The name of a continuous variable for which the split points will be reported
`round`	The number of decimal places to round the variable (var) to. No rounding occurs if round=NULL (default) For positive integer values of round, the variable will be rounded and split points will be reported at the resulting level

Details

splitpoints

Value

A data frame with split points and the probability that a split point was >= that split point value

Examples

```
# Split points with HDLM
D <- sim.hdlmm(sim = "B", n = 1000)
fit <- dlmtree(y ~ .,
               data = D$dat,
               exposure.data = D$exposures,
               dlm.type = "linear",
               family = "gaussian",
               het = TRUE)
splitpoints(fit, var = "mod_num", round = 2)
splitpoints(fit, var = "mod_scale", round = 2)
```

`summary.hdlm` *Creates a summary object of class 'hdlm'*

Description

Method for creating a summary object of class 'hdlm'

Usage

```
## S3 method for class 'hdlm'
summary(object, conf.level = 0.95, ...)
```

Arguments

`object`	an object of class 'hdlm'
`conf.level`	confidence level for computation of credible intervals
`...`	additional parameters

Details

summary.hdlm

Value

list of type 'summary.hdlm'

summary.hdlmm *Creates a summary object of class 'hdlmm'*

Description

Method for creating a summary object of class 'hdlm'

Usage

```
## S3 method for class 'hdlmm'
summary(object, conf.level = 0.95, ...)
```

Arguments

`object`	an object of class 'hdlmm'
`conf.level`	confidence level for computation of credible intervals
`...`	additional parameters

Details

summary.hdlmm

Value

list of type 'summary.hdlmm'

summary.monotone *Creates a summary object of class 'monotone'*

Description

Method for creating a summary object of class 'monotone'

Usage

```
## S3 method for class 'monotone'
summary(
  object,
  pred.at = NULL,
  cenval = 0,
  conf.level = 0.95,
  exposure.se = NULL,
  mcmc = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

`object`	an object of class ’monotone’
`pred.at`	numerical vector of exposure values to make predictions for at each time period
`cenval`	scalar exposure value that acts as a reference point for predictions at all other exposure values
`conf.level`	confidence level for computation of credible intervals
`exposure.se`	scalar smoothing factor, if different from model
`mcmc`	TRUE or FALSE (default): return MCMC samplers
`verbose`	TRUE (default) or FALSE: print output
`...`	additional parameters

Details

summary.monotone

Value

Summary of monotone fit

`summary.tdlm` *Creates a summary object of class ’tdlm’*

Description

Method for creating a summary object of class ’tdlm’

Usage

```
## S3 method for class 'tdlm'
summary(object, conf.level = 0.95, ...)
```

Arguments

`object`	an object of dlm class ’tdlm’ (i.e. a linear effect DLM)
`conf.level`	confidence level for computation of credible intervals
`...`	additional parameters

Details

summary.tdlm

Value

list of type ’summary.tdlm’

summary.tdlmm *Creates a summary object of class 'tdlmm'*

Description

Method for creating a summary object of class 'tdlmm'

Usage

```
## S3 method for class 'tdlmm'
summary(
  object,
  conf.level = 0.95,
  marginalize = "mean",
  log10BF.crit = 0.5,
  verbose = TRUE,
  keep.mcmc = FALSE,
  ...
)
```

Arguments

`object`	an object of type 'tdlmm', the output from tdlmm()
`conf.level`	confidence level (default = 0.95)
`marginalize`	value(s) for calculating marginal DLMs, defaults to "mean", can also specify a percentile from 1-99 for all other exposures, or a named vector with specific values for each exposure
`log10BF.crit`	Bayes Factor criteria for selecting exposures and interactions, such that log10(BayesFactor) > x. Default = 0.5
`verbose`	show progress in console
`keep.mcmc`	keep all mcmc iterations (large memory requirement)
`...`	additional parameters

Details

summary.tdlmm

Value

list of type 'summary.tdlmm'

summary.tdlnm *Creates a summary object of class 'tdlnm'*

Description

Method for creating a summary object of class 'tdlnm'

Usage

```
## S3 method for class 'tdlnm'
summary(
  object,
  pred.at = NULL,
  cenval = 0,
  conf.level = 0.95,
  exposure.se = NULL,
  mcmc = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

`object`	an object of class 'tdlnm', result of a call to tdlnm
`pred.at`	numerical vector of exposure values to make predictions for at each time period
`cenval`	scalar exposure value that acts as a reference point for predictions at all other exposure values
`conf.level`	confidence level for computation of credible intervals
`exposure.se`	scalar smoothing factor, if different from model
`mcmc`	TRUE or FALSE (default): return MCMC samplers
`verbose`	TRUE (default) or FALSE: print output
`...`	additional parameters

Details

summary.tdlnm

Value

list of type 'summary.tdlnm'

tdlmm — *Treed Distributed Lag Mixture Models (Deprecated)*

Description

TDLMM is a method for estimating a Treed Distributed Lag Mixture Model. It operates by building an ensemble of pairs of regression trees. Each tree in a tree-pair partitions the time span of the exposure data and estimates a piecewise constant distributed lag effect. The two trees are then intersected to create an interaction surface for estimating the interaction between two exposures. Exposures are selected for each tree stochastically and each exposure or interaction has a unique shrinkage variance component. This allows for exposure variable selection in addition to the estimation of the distributed lag mixture model.

Usage

```
tdlmm(
  formula,
  data,
  exposure.data,
  n.trees = 20,
  n.burn = 2000,
  n.iter = 5000,
  n.thin = 5,
  family = "gaussian",
  binomial.size = 1,
  formula.zi = NULL,
  keep_XZ = FALSE,
  mixture.interactions = "noself",
  tree.params = c(0.95, 2),
  step.prob = c(0.25, 0.25, 0.25),
  mix.prior = 1,
  shrinkage = "exposures",
  subset = NULL,
  verbose = TRUE,
  diagnostics = FALSE,
  initial.params = NULL,
  ...
)
```

Arguments

`formula`	object of class formula, a symbolic description of the fixed effect model to be fitted, e.g. y ~ a + b
`data`	data frame containing variables used in the formula
`exposure.data`	named list containing equally sized numerical matrices of exposure data with same, having same length as data

`n.trees`	integer for number of trees in ensemble
`n.burn`	integer for length of burn-in
`n.iter`	integer for number of iterations to run model after burn-in
`n.thin`	integer thinning factor, i.e. keep every tenth iteration
`family`	'gaussian' for continuous response, 'logit' for binomial response with logit link, or 'zinb' for zero-inflated negative binomial with logit link
`binomial.size`	integer type scalar (if all equal, default = 1) or vector defining binomial size for 'logit' family
`formula.zi`	object of class formula, a symbolic description of the ZI model to be fitted, e.g. y ~ a + b. This only applies to ZINB where covariates for ZI model is different from NB model. This is same as the main formula by default
`keep_XZ`	FALSE (default) or TRUE: keep the model scale exposure and covariate data
`mixture.interactions`	'noself' (default) which estimates interactions only between two different exposures, 'all' which also allows interactions within the same exposure, or 'none' which eliminates all interactions and estimates only main effects of each exposure
`tree.params`	numerical vector of alpha and beta hyperparameters controlling tree depth (see Bayesian CART, 1998), default: alpha = 0.95, beta = 2
`step.prob`	numerical vector for probability of 1) grow/prune, 2) change, 3) switch exposure, defaults to (0.25, 0.25, 0.25) or equal probability of each step for tree updates
`mix.prior`	positive scalar hyperparameter for sparsity of exposures
`shrinkage`	character "all" (default), "trees", "exposures", "none", turns on horseshoe-like shrinkage priors for different parts of model
`subset`	integer vector to analyze only a subset of data and exposures
`verbose`	TRUE (default) or FALSE: print output
`diagnostics`	TRUE or FALSE (default) keep model diagnostic such as terminal nodes, acceptance details, etc.
`initial.params`	initial parameters for fixed effects model, FALSE = none (default), "glm" = generate using GLM, or user defined, length must equal number of parameters in fixed effects model
`...`	NA

Details

tdlmm

Model is recommended to be run for at minimum 5000 burn-in iterations followed by 15000 sampling iterations with a thinning factor of 5. Convergence can be checked by re-running the model and validating consistency of results.

Value

object of class 'tdlmm'

`tdlmm_Cpp` *dlmtree model with tdlmm approach*

Description

dlmtree model with tdlmm approach

Usage

```
tdlmm_Cpp(model)
```

Arguments

`model`	A list of parameter and data contained for the model fitting

Value

A list of dlmtree model fit, mainly posterior mcmc samples

`tdlnm` *Treed Distributed Lag Non-Linear Models (Deprecated)*

Description

TDLNM is a method for estimating Distributed Lag Linear and Non-Linear Models (DLMs/DLNMs). It operates by building an ensemble of regression trees, which each partition the exposure-time- response surface and make estimates at each sector. Trees from the ensemble each contribute a partial estimate of the exposure-time surface, while controlling for a model given by 'formula'.

Usage

```
tdlnm(
  formula,
  data,
  exposure.data,
  exposure.splits = 20,
  exposure.se = sd(exposure.data)/2,
  n.trees = 20,
  n.burn = 1000,
  n.iter = 2000,
  n.thin = 5,
  family = "gaussian",
  binomial.size = 1,
  formula.zi = NULL,
  tree.params = c(0.95, 2),
  step.prob = c(0.25, 0.25),
  monotone = FALSE,
  monotone.gamma0 = rep(0, ncol(exposure.data)),
  monotone.sigma = diag(ncol(exposure.data)) * 1.502^2,
  monotone.tree.time.params = c(0.95, 2),
  monotone.tree.exp.params = c(0.95, 2),
  monotone.time.kappa = NULL,
  shrinkage = ifelse(monotone, FALSE, TRUE),
  subset = NULL,
  lowmem = FALSE,
  verbose = TRUE,
  diagnostics = FALSE,
  initial.params = NULL,
  debug = FALSE,
  ...
)
```

Arguments

`formula`	object of class formula, a symbolic description of the fixed effect model to be fitted, e.g. y ~ a + b
`data`	data frame containing variables used in the formula
`exposure.data`	numerical matrix of complete exposure data with same length as data
`exposure.splits`	scalar indicating the number of splits (divided evenly across quantiles of the exposure data) or list with two components: 'type' = 'values' or 'quantiles', and 'split.vals' = a numerical vector indicating the corresponding exposure values or quantiles for splits. Setting exposure.splits equal to 0 will change the model to a distributed lag model, which assumes a linear effect of exposure.
`exposure.se`	numerical matrix of exposure standard errors with same size as exposure.data or a scalar smoothing factor representing a uniform smoothing factor applied to each exposure measurement, defaults to sd(exposure.data)/2
`n.trees`	integer for number of trees in ensemble, default = 20
`n.burn`	integer for length of burn-in, >=2000 recommended
`n.iter`	integer for number of iterations to run model after burn-in >=5000 recommended
`n.thin`	integer thinning factor, i.e. keep every fifth iteration
`family`	'gaussian' for continuous response, or 'logit' for binomial response with logit link
`binomial.size`	integer type scalar (if all equal, default = 1) or vector defining binomial size for 'logit' family
`formula.zi`	object of class formula, a symbolic description of the ZI model to be fitted, e.g. y ~ a + b. This only applies to ZINB where covariates for ZI model is different from NB model. This is same as the main formula by default
`tree.params`	numerical vector of alpha and beta hyperparameters controlling tree depth (see Bayesian CART, 1998), default: alpha = 0.95, beta = 2

`step.prob`	numerical vector for probability of 1) grow/prune, and 2) change, defaults to (0.25, 0.25) or equal probability of each step for tree updates
`monotone`	FALSE (default) or TRUE: estimate monotone effects
`monotone.gamma0`	———UPDATE———
`monotone.sigma`	———UPDATE———
`monotone.tree.time.params`	———UPDATE———
`monotone.tree.exp.params`	———UPDATE———
`monotone.time.kappa`	———UPDATE———
`shrinkage`	int, 1 (default) turn on tree-specific shrinkage priors, 0 turn off
`subset`	integer vector to analyze only a subset of data and exposures
`lowmem`	FALSE (default) or TRUE: turn on memory saver for DLNM, slower computation time
`verbose`	TRUE (default) or FALSE: print progress bar output
`diagnostics`	TRUE or FALSE (default) keep model diagnostic such as terminal nodes, acceptance details, etc.
`initial.params`	initial parameters for fixed effects model, FALSE = none (default), "glm" = generate using GLM, or user defined, length must equal number of parameters in fixed effects model
`debug`	if TRUE, outputs debugging messages
`...`	NA

Details

tdlnm

Model is recommended to be run for at minimum 5000 burn-in iterations followed by 15000 sampling iterations with a thinning factor of 10. Convergence can be checked by re-running the model and validating consistency of results.

Value

object of class 'tdlnm' or 'tdlm'

`tdlnm_Cpp` *dlmtree model with tdlnm approach*

Description

dlmtree model with tdlnm approach

Usage

```
tdlnm_Cpp(model)
```

Arguments

`model`	A list of parameter and data contained for the model fitting

Value

A list of dlmtree model fit, mainly posterior mcmc samples

`zeroToInfNormCDF` *Integrates (0,inf) over multivariate normal*

Description

Integrates (0,inf) over multivariate normal

Usage

```
zeroToInfNormCDF(mu, sigma)
```

Arguments

`mu`	vector of mean parameters
`sigma`	covariance matrix

Value

double

`zinbCo` *Time-series exposure data for ZINB simulated data*

Description

Data.frame containing a sample of weekly average wildfire PM, PM2.5, O3 across a range of counties of Colorado. The exposure data was downloaded from US EPA (https://aqs.epa.gov/aqsweb/airdata/download_files.html) daily data summaries and averaged by week.

Usage

```
data(zinbCo)
```

Format

data.frame;

Source

https://aqs.epa.gov/aqsweb/airdata/download_files.html

References

https://www.epa.gov/outdoor-air-quality-data

Index