Package 'bdlnm'

June 19, 2026

Title Bayesian Distributed Lag Non-Linear Models (B-DLNM)

Version 0.1.1

Description A Bayesian framework for estimating distributed lag linear and non-linear models. Model fitting is implemented using Integrated Nested Laplace Approximation (R package 'INLA'), together with prediction and visualization of exposure-lag-response associations. Additional functions allow estimation of optimal exposure values (e.g., minimum mortality temperature) and computation of attributable fractions and numbers. Models with 'crossbasis' or 'onebasis' terms are supported (R package 'dlnm').

License GPL (>= 3)

Config/testthat/edition 3

Encoding UTF-8

URL https://github.com/pasahe/bdlnm, https://pasahe.github.io/bdlnm/

BugReports https://github.com/pasahe/bdlnm/issues

Depends R (>= 4.4)

LazyData true

VignetteBuilder knitr

Imports dlnm, cli, tsModel, crs, graphics, grDevices, stats, utils

Suggests INLA (>= 23.4.24), knitr, rmarkdown, testthat (>= 3.0.0), sn, splines, tidyr, lubridate, ggplot2

Additional_repositories https://inla.r-inla-download.org/R/stable

Config/Needs/website rmarkdown

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Pau Satorra [aut, cre] (ORCID: <https://orcid.org/0000-0002-8144-4089>),
Marcos Quijal-Zamorano [aut],
Joan Ballester [aut],
Cristian Tebé [ctb],
Miguel A. Martínez-Beneito [aut, ths],
Marc Marí-Dell'Olmo [aut, ths]

Maintainer Pau Satorra <psatorra@igtp.cat>

Repository CRAN

Date/Publication 2026-06-19 17:20:02 UTC

Contents

`attributable` *Calculate attributable number and fractions from a Bayesian distributed-lag model (B-DLNM).*

Description

Compute attributable numbers (AN) and attributable fractions (AF) from a fitted Bayesian distributed lag non-linear model (`bdlnm()`). The function uses posterior predicted relative risks from `bcrosspred()` and applies a forward or backward lag algorithm to compute per-time and total attributable measures, optionally filtering the calculation to a subset of time points.

Usage

```
attributable(
  object,
  data,
  name_date = NULL,
  name_exposure,
  name_cases = NULL,
  name_filter = NULL,
  dir = "back",
  basis = NULL,
  cen,
  range = NULL,
  lag_average = TRUE
)
```

Arguments

`object`	A fitted `"bdlnm"` class object returned by `bdlnm()`.

data	A data frame containing the temporal series needed to calculate attributable measures. It can include a date column (optional, see name_date), the exposure values (mandatory, see name_exposure), the number of cases (optional, see name_cases) and a binary (0/1) filter column (optional, see name_filter).
name_date	Optional single string with the name of the column in data containing the date. The column must be of class Date or POSIXt. When provided the function checks that the series is regularly spaced (see Details).
name_exposure	Single string with the name of the exposure column in data.
name_cases	Optional single string with the name of the cases column in data. If not provided, the function returns only attributable fractions (AF) per time point.
name_filter	Optional single string with the name of a binary (0/1) column in data. Only rows with value 1 are used to compute total AF and AN and per-time results are returned only for the filtered rows.
dir	Character; direction of the algorithm to calculate attributable measures. "back" (default) calculates AF and AN attributing current-time outcome to past exposures; "forw" calculates AF and AN attributing current-time exposure to future outcomes.
basis	If the bdlnm model has more than one basis, the name of the basis to use to compute predictions. It must be one of names(object$basis). If the model contains only one basis it is selected automatically.
cen	Numeric scalar; centering exposure value used to compute predictions. If missing the function will attempt to read it from the attributes of the specified basis. If no centering is available the function aborts.
range	Optional numeric vector of length 2 with the exposure range for which attributable measures will be calculated. Values outside range are coerced to cen before prediction.
lag_average	Logical (default TRUE). When TRUE use lag-averaged contributions to compute AN in the forward algorithm; when FALSE use the full lag-structured contributions instead.

Details

The function first obtains posterior predicted effects at the observed exposure values by calling bcrosspred() with exp_at = data[[name_exposure]]. Predictions must include relative-risk scale predictions so the previously fitted "bdlnm" model must have a log or logit link; otherwise the function aborts. These predictions require a defined centering value (cen), as attributable measures are always computed with respect to a reference exposure. This reference exposure is usually an optimal exposure computed with the optimal_exposure() function, such as the Minimum Mortality Temperature (MMT).

Two different algorithms can be chosen to calculate attributable measures:

- Backward (dir = "back"): for each time point, contributions from past exposures (over the lag window) are combined to calculate the daily AF/AN.
- Forward (dir = "forw"): for each time point, the contribution of that time point exposure to future outcomes (over the lag window) is combined to calculate the daily AF/AN.

Both algorithms are fully described by Gasparrini and Leone (2014) doi:10.1186/1471-2288-14-55.

Required columns to calculate AF and AN are name_exposure and name_cases columns. If name_cases is not supplied only AF per time can be computed and the output will only contain two elements: $af and $af.summary. If name_date is provided the function checks that dates are equispaced (checks seconds, minutes, hours, days, weeks, months or years). Time series have to be equispaced because the algorithms used to calculate attributable measures rely on consecutive time points over the lag window,. For example, if you only have seasonal observations (e.g., summers) expand the data to the full sequence and insert NA for missing exposures/cases and use name_filter to compute measures only for the seasonal subset.

Only "bdlnm" objects fitted with a cross-basis are supported; models fitted with a one-basis (no lag) are not suitable for attributable calculations.

Value

A list with components:

- af: matrix (rows = time points, columns = posterior samples) with attributable fractions per time point.
- an: matrix (rows = time points, columns = posterior samples) with attributable numbers per time point.
- aftotal: numeric vector (length = number of posterior samples) with posterior samples of total AF across all the selected period.
- antotal: numeric vector (length = number of posterior samples) with posterior samples of total AN across all the selected period.
- af.summary: data.frame with summary statistics (mean, sd, quantiles, mode) for AF per time point.
- an.summary: data.frame with summary statistics (mean, sd, quantiles, mode) for AN per time point.
- aftotal.summary: data.frame with summary statistics (mean, sd, quantiles, mode) for total AF.
- antotal.summary: data.frame with summary statistics (mean, sd, quantiles, mode) for total AN.

Note

This function is inspired by attrdl() developed by Gasparrini and Leone (2014) doi:10.1186/1471-2288-14-55. It has been adapted to work in a Bayesian framework within the **bdlnm** package.

Author(s)

Pau Satorra, Marcos Quijal-Zamorano.

References

Gasparrini A., Leone M. (2014). Attributable risk from distributed lag models. *BMC Medical Research Methodology*, 14, 55. doi:10.1186/1471-2288-14-55.

Quijal-Zamorano M., Martinez-Beneito M.A., Ballester J., Marí-Dell'Olmo M. (2024). Spatial Bayesian distributed lag non-linear models (SB-DLNM) for small-area exposure-lag-response epidemiological modelling. *International Journal of Epidemiology*, 53(3), dyae061. doi:10.1093/ije/dyae061.

See Also

`bcrosspred()` to predict exposure–lag–response associations for a `"bdlnm"` object,

`bdlnm()` to fit a Bayesian distributed lag non-linear model (`"bdlnm"`),

`optimal_exposure()` to estimate exposure values that optimize the predicted effect for a `"bdlnm"` object.

Examples

```
# Filter the dataset to reduce computational time:

# Exposure-response and lag-response spline parameters
dlnm_var <- list(
  var_prc = c(10, 75, 90),
  var_fun = "ns",
  lag_fun = "ns",
  max_lag = 21,
  lagnk = 3
)

# Cross-basis parameters
argvar <- list(fun = dlnm_var$var_fun,
               knots = stats::quantile(london$tmean,
                               dlnm_var$var_prc/100, na.rm = TRUE),
               Bound = range(london$tmean, na.rm = TRUE))

arglag <- list(fun = dlnm_var$lag_fun,
               knots = dlnm::logknots(dlnm_var$max_lag, nk = dlnm_var$lagnk))

# Create crossbasis
cb <- dlnm::crossbasis(london$tmean, lag = dlnm_var$max_lag, argvar, arglag)

# Seasonality of mortality time series
seas <- splines::ns(london$date, df = round(8 * length(london$date) / 365.25))

 # Prediction values (equidistant points)
 temp <- round(seq(min(london$tmean), max(london$tmean), by = 0.1), 1)
 # Ensure it falls inside the range of temperatures after rounding:
 temp <- temp[temp >= min(london$tmean) & temp <= max(london$tmean)]

# Model
```

```
if (check_inla()) {
mod <- bdlnm(mort_75plus ~ cb + factor(dow) + seas,
             data = london,
             family = "poisson",
             sample.arg = list(n = 1000, seed = 432))

# Predict
cpred <- bcrosspred(mod, exp_at = temp)

# compute centering (MMT) using optimal_exposure
mmt <- optimal_exposure(mod, exp_at = temp)
cen <- mmt$summary[["0.5quant"]]

# Attributable numbers and fractions (using the backwards algorithm):
attr <- attributable(mod, london, name_date = "date",
name_exposure = "tmean", name_cases = "mort_75plus", cen = cen, dir = "back")
}
```

`bcrosspred` *Predict exposure-lag-response effects from a Bayesian distributed-lag model (B-DLNM).*

Description

Calculate predictions from a fitted Bayesian distributed lag non-linear model (`bdlnm()`). Predicted associations are computed on a grid of values of the exposure and lags, relative to a reference exposure center value. The function gives posterior samples of exposure–lag-specific associations, overall cumulative associations (summed across lags) and optionally incremental cumulative associations, together with summary statistics (mean, sd, credible interval quantiles and mode).

Usage

```
bcrosspred(
  object,
  basis = NULL,
  exp_at = NULL,
  lag_at = NULL,
  cen = NULL,
  model.link = NULL,
  ci.level = 0.95,
  cumul = FALSE
)
```

Arguments

`object`	A fitted `"bdlnm"` object returned by `bdlnm()`.

basis	If the bdlnm model has more than one basis, the name of the basis to use to compute predictions. It must be one of names(object$basis). If the model contains only one basis it is selected automatically.
exp_at	Numeric vector of exposure values at which to evaluate predictions. If NULL, the exposure range is extracted from the attributes of basis and a grid of 50 values is constructed using pretty().
lag_at	Numeric vector of integer lag values at which to evaluate predictions. Only used when basis is a crossbasis. If NULL, the full lag range stored in basis is used with a step of 1.
cen	Centering exposure value for predictions. If NULL the centering value depends on the exposure basis function or set to a mid-range value (see Details).
model.link	Optional character specifying the model link (if NULL it is inferred from the fitted model).
ci.level	Numeric in (0,1) giving the credible interval level (default 0.95). Credible interval quantiles are computed from the posterior samples.
cumul	Logical; if TRUE compute incremental cumulative effects along lags (default FALSE). It will give the cumulative effect from lag 0 up to each subsequent lag (e.g., lag 0, lag 0–1, lag 0–2, etc.).

Details

The function computes predictions for specific combinations of exposure and lag values specified in exp_at and lag_at. All values must lie within the range defined by the specified basis. Note that if the specified basis is a onebasis, lag_at is ignored. If either argument is NULL, the grid is derived from the attributes of the specified basis: for exposure values, a grid of approximately 50 values is constructed using pretty() within the exposure range; for lag values, an integer sequence covering the full lag range with a step of 1 is used.

Predictions are computed relative to a centering/reference value (cen). If NULL, the default cen depends on the exposure-response basis function: for strata, thr and integer the reference corresponds to the reference category, and for lin the reference is set to 0. For other choices, such as ns, bs, poly or other existing or user-defined functions, the default centering value is set to an approximate mid-range value. For non-linear exposure-response associations is sometimes recommended to manually set the centering value to a data-driven center such as the optimal exposure value (see optimal_exposure()).

Posterior sample of the predicted associations are stored as matrices for the overall cumulative effect and 3D arrays for the exposure-lag-specific predictions. Summaries across these samples are computed using the mean, sd, credible-interval quantiles (the mid and the lower/upper tails according to ci.level) and an approximate mode obtained from a kernel density estimate. Relative risks versions of these associations (exponentiated predictions) are also included if model.link is equal to "log" or "logit". The model.link can be manually specified or, if NULL, it is tried to be inferred from the model type in object.

Be aware of memory usage: exposure-lag-specific predictions are stored in 3D arrays of dimension length(predvar) × length(predlag) × n_sim. For dense grids and many posterior samples this can be computationally intensive.

This function can also be used to compute predictions for models with simple uni-dimensional basis functions not including lags, if the specified basis is "onebasis" instead of "crossbasis". In this case, only unlagged predicted associations are returned.

Value

An object of class "bcrosspred" (a list) with elements including:

- exp_at: the exposure grid used for prediction as supplied via the exp_at argument (vector).
- lag_at: if basis is crossbasis, the lag grid used for prediction as supplied via the lag_at argument (vector).
- cen: the exposure centering value used for prediction (number).
- coefficients: matrix of posterior coefficient draws (columns = samples).
- coefficients.summary: matrix of coefficient summaries (mean, sd, quantiles, mode).
- matfit: 3D array of sampled lag-specific effects (exp × lag × samples).
- matfit.summary: 3D array of summaries for matfit (exp × lag × summary-statistics).
- allfit: matrix of sampled overall cumulative (summed across lags) effects (exp × samples).
- allfit.summary: matrix of summaries for allfit (exp x summary-statistics).
- cumfit: (optional) 3D array of sampled incremental cumulative effects (exp × lag × samples).
- cumfit.summary: (optional) 3D array of summaries for cumfit (exp × lag × summary-statistics).
- matRRfit, allRRfit, matRRfit.summary, allRRfit.summary, cumRRfit.summary (optional), cumRRfit.summary (optional: relative-risk versions (only when link is log or logit).
- ci.level, model.class, model.link.

Note

This function is inspired by dlnm::crosspred() developed by Gasparrini (2011) doi:10.18637/jss.v043.i08. It has been adapted to work in a Bayesian framework within the **bdlnm** package.

Author(s)

Pau Satorra, Marcos Quijal-Zamorano.

References

Gasparrini A. (2011). Distributed lag linear and non-linear models in R: the package dlnm. *Journal of Statistical Software*, 43(8), 1-20. doi:10.18637/jss.v043.i08.

Quijal-Zamorano M., Martinez-Beneito M.A., Ballester J., Marí-Dell'Olmo M. (2024). Spatial Bayesian distributed lag non-linear models (SB-DLNM) for small-area exposure-lag-response epidemiological modelling. *International Journal of Epidemiology*, 53(3), dyae061. doi:10.1093/ije/dyae061.

See Also

plot.bcrosspred() to plot the predicted associations stored in a "bcrosspred" object,

bdlnm() to fit a Bayesian distributed lag non-linear model ("bdlnm").

attributable() to calculate attributable fractions and numbers for a "bdlnm" object,

optimal_exposure() to estimate exposure values that optimize the predicted effect for a "bdlnm" object.

Examples

```
# Set exposure-response and lag-response spline parameters
 dlnm_var <- list(
   var_prc = c(10, 75, 90),
   var_fun = "ns",
   lag_fun = "ns",
   max_lag = 21,
   lagnk = 3
 )


# Set cross-basis parameters
 argvar <- list(fun = dlnm_var$var_fun,
                knots = stats::quantile(london$tmean,
                                        dlnm_var$var_prc/100, na.rm = TRUE),
                Bound = range(london$tmean, na.rm = TRUE))

 arglag <- list(fun = dlnm_var$lag_fun,
                knots = dlnm::logknots(dlnm_var$max_lag, nk = dlnm_var$lagnk))

 # Create crossbasis
 cb <- dlnm::crossbasis(london$tmean, lag = dlnm_var$max_lag, argvar, arglag)

 # Seasonality of mortality time series
 seas <- splines::ns(london$date, df = round(8 * length(london$date) / 365.25))

 # Prediction values (equidistant points)
 temp <- round(seq(min(london$tmean), max(london$tmean), by = 0.1), 1)
 # Ensure it falls inside the range of temperatures after rounding:
 temp <- temp[temp >= min(london$tmean) & temp <= max(london$tmean)]

if (check_inla()) {
 # Fit the model
 mod <- bdlnm(mort_75plus ~ cb + factor(dow) + seas, data = london, family = "poisson",
              sample.arg = list(n = 1000, seed = 432))

 # Prediction
 cpred <- bcrosspred(mod, exp_at = temp)
}
```

`bdlnm` *Fit a Bayesian distributed lag non-linear model (B-DLNM)*

Description

Fit a distributed lag non-linear model (DLNM) using a Bayesian framework. The function calls `INLA::inla()` to fit the model and then draws posterior samples of the model fixed effects with `INLA::inla.posterior.sample()`. See the package vignette for worked examples and recommended workflows.

Usage

```
bdlnm(
  formula,
  data,
  family = "gaussian",
  sample.arg = list(n = 1000, seed = 0L),
  ci.level = 0.95,
  na.action = getOption("na.action"),
  ...
)
```

Arguments

formula	A model formula (as for INLA::inla()). The model must be a distributed lag linear or non-linear model (DLNM), so a "crossbasis" (dlnm::crossbasis()) or a "onebasis" (dlnm::onebasis()) must be included.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which bdlnm is called.
family	Character. Family name passed to INLA::inla() (default "gaussian").
sample.arg	List of arguments passed to INLA::inla.posterior.sample(). Defaults to list(n = 1000, seed = 0L) (draws 1000 posterior samples; seed at random). For reproducible sampling set a non-zero numeric seed.
ci.level	Numeric in (0,1) giving the credible interval level (default 0.95). Credible interval quantiles are computed to summarize coefficients from the posterior samples.
na.action	A function specifying how to handle NA values when constructing the model frame. The default is taken from the na.action setting of options, which is by default na.omit (drops rows with any NA among the variables referenced in formula). In the presence of a random effect term f() in the formula, this argument is ignored and NAs are not discarded. When rows containing missing values are retained, INLA handles them internally (see Details below).
...	Additional arguments passed to INLA::inla().

Value

An S3 object of class "bdlnm" with the following components:

- model: the fitted INLA model returned by INLA::inla().
- basis: a named list containing all basis of class crossbasis or onebasis included in the model formula.
- coefficients: a matrix whose columns are posterior sample draws returned by INLA::inla.posterior.sample() (named sample1, sample2, ...) and whose rows are all model coefficients.
- coefficients.summary: a matrix of summary statistics for all the posterior samples stored in coefficients (mean, sd, quantiles, mode).

Distributed lag non-linear model

The fitted model must be a distributed lag linear or non-linear model (DLNM). DLNMs describe potentially non-linear and delayed (lagged) associations between an exposure and an outcome, commonly referred to as exposure–lag–response relationships. This modelling framework is based on the definition of a cross-basis (a bi-dimensional space of functions) constructed with `dlnm::crossbasis()`, which defines the exposure–response and lag–response functions simultaneously. The cross-basis object must be created beforehand and supplied as an object in the calling environment (not as a column inside data) and explicitly included in the model formula (e.g., y ~ cb + ...). A basis object constructed with `dlnm::onebasis()` can be used instead when the model is restricted to a uni-dimensional exposure–response relationship (i.e., without lagged effects). All basis objects included in the model `formula` are stored and returned as a named list in the `basis` component. Any of these basis objects can later be supplied to `bcrosspred()` to extract predictions for the corresponding exposure–lag–response (or exposure-response, if created with `onebasis`) association.

INLA

Models are fit using Integrated Nested Laplace approximation (INLA) via `INLA::inla()`. INLA is a method for approximate Bayesian inference. In the last years it has established itself as an alternative to other methods such as Markov chain Monte Carlo because of its speed and ease of use via the R-INLA package (What is INLA?).

Additional arguments supplied via `...` are forwarded to `INLA::inla()` (see documentation for all available arguments). Internally, the function ensures that `control.compute = list(config = TRUE)` in order to enable posterior sample drawing with `INLA::inla.posterior.sample()`.

In the presence of missing values in variables referenced in formula, the `na.action` argument controls how the model frame is constructed. If `na.action` is set to na.omit (the default set by options), a complete-case analysis is performed and any row with a missing value in a variable appearing in formula is removed. If `na.action` is set to na.pass instead, missing values are retained in the model frame and passed to INLA.

Note that when the model formula includes a random effect term, specified via `f(k, model = ...)`, the `na.action` is ignored and rows with missing values are not dropped.

When missing values are present in the data supplied to `INLA::inla()` (either because a random effect term is included or because `na.action = na.pass`), `INLA` handles them internally as follows:

- If `NA` values occur in the response, the corresponding observation contributes nothing to the likelihood (the response is treated as unobserved for that observation).
- If `NA` values occur in fixed-effect covariates, `INLA::inla()` replaces them internally with zero so that the covariate does not contribute to the linear predictor for that observation.
- If `NA` values occur in a fixed-effect covariate that is a factor, this is not allowed unless `NA` is explicitly included as a level, or `control.fixed = list(expand.factor.strategy = "inla")` is specified. With this option, `NA` is interpreted similarly as in the fixed-effect case, producing no contribution from that covariate to the linear predictor.
- If `NA` values occur in a random effect, the random effect does not contribute to the linear predictor for the corresponding observation.

Posterior samples

After fitting the model, the function draw samples from the approximate posterior distribution of the latent field via `INLA::inla.posterior.sample()`. These samples are collected into a matrix and summarized across samples (mean, sd, quantiles and mode). For a `"crossbasis"` built from an exposure basis with C parameters and a lag basis with L parameters, there will be C × L cross-basis associated coefficients (named e.g. v1.l1, ..., vC.lL). For a `"onebasis"` object the coefficients follow the simpler form b1, ... bC.

Additional arguments supplied via `sample.arg` are forwarded to `INLA::inla.posterior.sample()` (see documentation for all available arguments). By default, the number of samples is `1000`. Be aware of the computation and memory cost when increasing the number of samples drawn. By default, the seed is set at random. For reproducible samplings, you need to set a non-zero numeric `seed` in `sample.arg`.

Posterior sample estimations are then summarized across samples using mean, sd, credible-interval quantiles (the mid and the lower/upper tails according to `ci.level`) and an approximate mode obtained from a kernel density estimate.

Requirements

The INLA package must be installed from the R-INLA repository (R-INLA Project); if not available the function aborts with a short instruction on how to install it.

Author(s)

Pau Satorra, Marcos Quijal-Zamorano.

References

Quijal-Zamorano M., Martinez-Beneito M.A., Ballester J., Marí-Dell'Olmo M. (2024). Spatial Bayesian distributed lag non-linear models (SB-DLNM) for small-area exposure-lag-response epidemiological modelling. *International Journal of Epidemiology*, 53(3), dyae061. doi:10.1093/ije/dyae061.

Quijal-Zamorano M., Martinez-Beneito M.A., Ballester J., Marí-Dell'Olmo M. (2025). Spatial Bayesian distributed lag non-linear models with R-INLA. *International Journal of Epidemiology*, 54(4), dyaf120. doi:10.1093/ije/dyaf120.

Gasparrini A. (2011). Distributed lag linear and non-linear models in R: the package dlnm. *Journal of Statistical Software*, 43(8), 1-20. doi:10.18637/jss.v043.i08.

Rue H., Martino S., Chopin N. (2009). Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal Statistical Society: Series B*, 71(2), 319-392. doi:10.1111/j.1467-9868.2008.00700.x.

See Also

`bcrosspred()` to predict exposure–lag–response associations for a `bdlnm` object,

`attributable()` to calculate attributable fractions and numbers for a `bdlnm` object,

`optimal_exposure()` to estimate exposure values that optimise the predicted effect for a `bdlnm` object.

Examples

```
# Set exposure-response and lag-response spline parameters
 dlnm_var <- list(
   var_prc = c(10, 75, 90),
   var_fun = "ns",
   lag_fun = "ns",
   max_lag = 21,
   lagnk = 3
 )


# Set cross-basis parameters
 argvar <- list(fun = dlnm_var$var_fun,
                knots = stats::quantile(london$tmean,
                                        dlnm_var$var_prc/100, na.rm = TRUE),
                Bound = range(london$tmean, na.rm = TRUE))

 arglag <- list(fun = dlnm_var$lag_fun,
                knots = dlnm::logknots(dlnm_var$max_lag, nk = dlnm_var$lagnk))

 # Create crossbasis
 cb <- dlnm::crossbasis(london$tmean, lag = dlnm_var$max_lag, argvar, arglag)

 # Seasonality of mortality time series
 seas <- splines::ns(london$date, df = round(8 * length(london$date) / 365.25))

 # Prediction values (equidistant points)
 temp <- round(seq(min(london$tmean), max(london$tmean), by = 0.1), 1)
 # Ensure it falls inside the range of temperatures after rounding:
 temp <- temp[temp >= min(london$tmean) & temp <= max(london$tmean)]

if (check_inla()) {
 # Fit the model
 mod <- bdlnm(mort_75plus ~ cb + factor(dow) + seas, data = london, family = "poisson",
              sample.arg = list(n = 1000, seed = 432))
}
```

---

london *London temperature and mortality data*

Description

The dataset includes observed daily mean temperature and total number of deaths in London between 2000 and 2011. Mortality data is stratified for <75 years and 75+ years age groups. The dataset is based on the data used in Vicedo-Cabrera et al. (2019).

Usage

```
data(london)
```

Format

`london`:

A tibble with 8.279 rows and 7 columns:

time Date index

date Date

year Year

dow Day of the week

tmean Temperature mean

mort_00_74 Mortality in the age group <75 years

mort_75plus Mortality in the age group +75 years

mort All mortality

Source

Data originally published by Vicedo-Cabrera (2019) doi:10.1097/EDE.0000000000000982.

References

Vicedo-Cabrera A.M., Sera F., Gasparrini A. (2019). Hands-on tutorial on a modeling framework for projections of climate change impacts on health. *Epidemiology*, 30(3), 321-329. doi:10.1097/EDE.0000000000000982.

`optimal_exposure` *Calculate the exposure value that minimizes or maximizes the overall cumulative effect of a Bayesian distributed lag non-linear model (B-DLNM)*

Description

Find exposure values that optimize the overall effect for each posterior sample drawn from a Bayesian distributed lag non-linear model (`bdlnm()`). The function returns the exposure value that minimizes or maximizes the overall cumulative effect (summed across lags) for each posterior sample, together with summary statistics (mean, sd, credible-interval quantiles and mode). When used to find the minimum effect in temperature–mortality analyses this optimal exposure value is commonly called the Minimum Mortality Temperature (MMT).

Usage

```
optimal_exposure(
  object,
  basis = NULL,
  exp_at = NULL,
  lag_at = NULL,
  which = "min",
  local_optimal = FALSE,
  ci.level = 0.95
)
```

Arguments

object	A fitted "bdlnm" object returned by bdlnm().
basis	If the bdlnm model has more than one basis, the name of the basis to use to compute predictions. It must be one of names(object$basis). If the model contains only one basis it is selected automatically.
exp_at	Numeric vector of exposure values at which to evaluate predictions. If NULL, the exposure range is extracted from the attributes of the specified basis and a grid of 50 values is constructed using pretty().
lag_at	Numeric vector of integer lag values ver which to compute the overall cumulative effect. Only used when basis is a crossbasis. If NULL, the overall effect is computed by summing over the full lag range stored in basis (with step size 1).
which	Selection criterion to calculate the optimal exposure: "min" (default) chooses the exposure with the minimum overall cumulative effect, "max" chooses the exposure with maximum overall cumulative effect.
local_optimal	Logical (default FALSE). When TRUE find a local optimal (minimum or maximum) point with the optimal effect instead of the absolute optimal point. If a local optimal point doesn't exist it will fall back to finding the absolute optimal point.
ci.level	Numeric in (0,1) giving the credible-interval level (default 0.95). Credible interval quantiles are computed from the posterior samples.

Details

The function internally calls bcrosspred() to compute the posterior distribution of the overall cumulative exposure effect for the grid specified by exp_at. For each posterior sample the function calculates the exposure value that optimizes (minimizes or maximizes) the overall cumulative effect and then summarizes these optimal values across samples using mean, sd, credible-interval quantiles and the mode (most frequent observed value).

The overall cumulative effect is computed by summing for each exposure the lag-specific effects over the lags specified in lag_at. If lag_at is NULL, the cumulative effect is computed for each exposure by summing over the full lag range stored in basis (with step size 1). If basis is a onebasis, the function optimizes the exposure-response association stored in $matfit, and lag_at is ignored.

The function searches for the absolute optimal value (minimum or maximum) of each sample in the posterior distribution, by default. If `local_optimal` is set to `TRUE`, the function searches for a local optimal point instead. If more than one optimal point is found, the function will return the one with the optimal effect. If a posterior sample has no local optimal values, the function returns the absolute optimal value.

This optimal exposure value can be used as the reference exposure value to estimate effects passing it to the `bcrosspred()` and `attributable()` functions as the center exposure. In temperature-mortality studies, for example, the minimum exposure value is typically used as the optimal exposure value to center the effects and it's called Minimum Mortality Temperature (MMT). However, note that in the Bayesian framework, this reference temperature is characterized by a full posterior distribution (in contrast to the frequentist approach, where the association is centered on a single point estimate). This distribution may be asymmetric and non-unimodal, so reporting a single summary statistic (e.g., the median) as the reference value can be misleading in such cases. Therefore, before selecting an optimal exposure value as the center, it is recommended that you visualize the distribution of the optimal exposure values using `plot.optimal_exposure()`.

This function cannot be used when the specified basis function is one of `thr`, `strata`, `integer`, or `lin`. The exposure-response relationship is discrete, piecewise, or strictly linear in these situations, so searching for an optimum is not meaningful.

Value

An S3 object of class `"optimal_exposure"` containing:

- `est`: numeric vector with the optimal exposure value for each posterior sample (named sample1, sample2, ...).
- `summary`: a one-row data frame with summary statistics for the optimal values across all samples (mean, sd, quantiles, mode).

Author(s)

Pau Satorra, Marcos Quijal-Zamorano.

References

Quijal-Zamorano M., Martinez-Beneito M.A., Ballester J., Marí-Dell'Olmo M. (2024). Spatial Bayesian distributed lag non-linear models (SB-DLNM) for small-area exposure-lag-response epidemiological modelling. *International Journal of Epidemiology*, 53(3), dyae061. doi:10.1093/ije/dyae061.

Gasparrini A. (2011). Distributed lag linear and non-linear models in R: the package dlnm. *Journal of Statistical Software*, 43(8), 1-20. doi:10.18637/jss.v043.i08.

Armstrong B. (2006). Models for the relationship between ambient temperature and daily mortality. *Epidemiology*, 17(6), 624-631. doi:10.1097/01.ede.0000239732.50999.8f.

See Also

`plot.optimal_exposure()` to plot the optimal exposure values stored in a `"optimal_exposure"` object.

`bcrosspred()` to predict exposure–lag–response associations for a `"bdlnm"` object,

bdlnm() to fit a Bayesian distributed lag non-linear model ("bdlnm").

attributable() to calculate attributable fractions and numbers for a "bdlnm" object.

Examples

```
# Set exposure-response and lag-response spline parameters
 dlnm_var <- list(
   var_prc = c(10, 75, 90),
   var_fun = "ns",
   lag_fun = "ns",
   max_lag = 21,
   lagnk = 3
 )


# Set cross-basis parameters
 argvar <- list(fun = dlnm_var$var_fun,
                knots = stats::quantile(london$tmean,
                                        dlnm_var$var_prc/100, na.rm = TRUE),
                Bound = range(london$tmean, na.rm = TRUE))

 arglag <- list(fun = dlnm_var$lag_fun,
                knots = dlnm::logknots(dlnm_var$max_lag, nk = dlnm_var$lagnk))

 # Create crossbasis
 cb <- dlnm::crossbasis(london$tmean, lag = dlnm_var$max_lag, argvar, arglag)

 # Seasonality of mortality time series
 seas <- splines::ns(london$date, df = round(8 * length(london$date) / 365.25))

 # Prediction values (equidistant points)
 temp <- round(seq(min(london$tmean), max(london$tmean), by = 0.1), 1)
 # Ensure it falls inside the range of temperatures after rounding:
 temp <- temp[temp >= min(london$tmean) & temp <= max(london$tmean)]

if (check_inla()) {
 # Fit the model
 mod <- bdlnm(mort_75plus ~ cb + factor(dow) + seas, data = london, family = "poisson",
              sample.arg = list(n = 1000, seed = 432))

 # Find minimum risk exposure value
 mmt <- optimal_exposure(mod, "cb", exp_at = temp)
}
```

---

plot.bcrosspred *Plot the predicted effects from a Bayesian distributed-lag model (B-DLNM).*

Description

It plots the lag-specific, overall, slices, contour or 3D representations for predicted effects generated by `bcrosspred()`.

Usage

```
## S3 method for class 'bcrosspred'
plot(
  x,
  ptype,
  exp_at = NULL,
  lag_at = NULL,
  ci = "area",
  ci.arg,
  ci.level = x$ci.level,
  cumul = FALSE,
  exponentiate = NULL,
  ...
)
```

Arguments

`x`	An object of class `"bcrosspred"` returned by `bcrosspred()`.
`ptype`	Character. Plot type: one of `"slices"`, `"overall"`, `"3d"` or `"contour"`.
`exp_at`	Numeric vector of exposure values (predictor) for the `"slices"` plot type (maximum length of 4).
`lag_at`	Numeric vector of lags for the `"slices"` plot type (maximum length of 4).
`ci`	Character. How to plot credible intervals: one of `"area"`, `"bars"`, `"lines"`, `"sampling"` (draw a line for each posterior sample) or `"none"`. Default: `"area"`.
`ci.arg`	List of graphical arguments for the plotting of credible intervals passed to base plotting functions (see `Details`).
`ci.level`	Numeric in `(0,1)`. Credible interval level. Default is taken from `x$ci.level`.
`cumul`	Logical. If `TRUE`, plot incremental cumulative predictions (requires cumulative predictions in `x`).
`exponentiate`	Logical. Set to `TRUE` to plot exponentiated effects (relative risks) and to `FALSE` to not plot them. The default is `NULL` in which the function uses `x$model.link` to automatically detect whether to use relative risks (when link is `log` or `logit`).
`...`	Additional graphical arguments passed to base plotting functions (see `Details`).

Details

The function supports different visualizations for posterior effects predicted by `bcrosspred()` specified in `ptype`:

- `"slices"`: (optionally multi-panel) line plot of the predicted effect over lag(s) for selected exposure value(s) or over exposure value(s) for selected lag(s). Use `exp_at` and/or `lag_at`

to specify which slices to draw (at least one must be supplied). At most 4 slices of each type are allowed to keep layouts readable. See `plot.default()`, `lines()` and `points()` for information on additional graphical arguments.

- `"overall"`: plot the predicted overall cumulative effect over the whole lag period for each exposure value in the prediction grid. See `plot.default()`, `lines()` and `points()` for information on additional graphical arguments.
- `"3d"`: 3D plot of the surface of the predicted effect for each exposure and lag. Uses the median of the posterior samples. Not meaningful for single lag models. Additional graphical arguments can be included, such as `theta` or `phi` (perspective), `border` or `shade` (surface), `xlab`, `ylab` or `zlab` (axis labelling) and `col`. See `persp()` for additional information.
- `"contour"`: filled contour of the surface of the predicted effect for each exposure and lag. Uses the median of the posterior samples. Not meaningful for single lag models. Additional graphical arguments can be included, such as `plot.title`, `plot.axes` or `key.title` for titles and axis and key labelling. See `filled.contour()` for additional information.

The function will call the specified original base plotting functions for each different `ptype`. Via the argument `...` the user can change the graphical parameters that will be passed to these functions. See the original functions for a complete list of the arguments. Some arguments, if not specified, are set to different default values than the original functions.

Credible intervals will only be drawn for `ptype` equal to `"slices"` or `"overall"`. The type of credible interval will be given by the `ci` argument, with options `"area"` (default), `"bars"`, `"lines"`, `"sampling"` (draw a line for each posterior sample) or `"none"` (no credible intervals). Their appearance may be modified through `ci.arg`, a list of arguments passed to to low-level plotting functions: `polygon()` for `"area"`, `segments()` for `"bars"` and `lines()` for `"lines"`. See the original functions for a complete list of the arguments. As above, some unspecified arguments are set to different default values. The credible interval level will be given by `ci.level` or inferred from `x$ci.level` by default.

If exponentiated effects (relative risks) are specified (`exponentiate = TRUE`) or auto-detected if `x$model.link` is `log` or `logit`, the function will use the predicted relative risks stored in `x$matRRfit`, `x$allRRfit` or `x$cumRRfit` (if `cumul=TRUE`).

In the presence of unlagged or single lag associations, only the `overall` plot can be produced. In this case, the `overall` plot represents the effect at each predictor value, rather than the overall cumulative effect across lags.

Value

No return value, called for side effects.

Note

This function is inspired by `dlnm::plot.crosspred()` developed by Gasparrini (2011) doi:10.18637/jss.v043.i08. It has been adapted to work in a Bayesian framework within the **bdlnm** package.

Author(s)

Pau Satorra, Marcos Quijal-Zamorano.

References

Gasparrini A. (2011). Distributed lag linear and non-linear models in R: the package dlnm. *Journal of Statistical Software*, 43(8), 1-20. doi:10.18637/jss.v043.i08.

Quijal-Zamorano M., Martinez-Beneito M.A., Ballester J., Marí-Dell'Olmo M. (2024). Spatial Bayesian distributed lag non-linear models (SB-DLNM) for small-area exposure-lag-response epidemiological modelling. *International Journal of Epidemiology*, 53(3), dyae061. doi:10.1093/ije/dyae061.

See Also

`bcrosspred()` to predict exposure–lag–response associations for a `"bdlnm"` object,

`bdlnm()` to fit a Bayesian distributed lag non-linear model (`"bdlnm"`).

Examples

```
# Set exposure-response and lag-response spline parameters
 dlnm_var <- list(
   var_prc = c(10, 75, 90),
   var_fun = "ns",
   lag_fun = "ns",
   max_lag = 21,
   lagnk = 3
 )


# Set cross-basis parameters
 argvar <- list(fun = dlnm_var$var_fun,
                knots = stats::quantile(london$tmean,
                                dlnm_var$var_prc/100, na.rm = TRUE),
                Bound = range(london$tmean, na.rm = TRUE))

 arglag <- list(fun = dlnm_var$lag_fun,
                knots = dlnm::logknots(dlnm_var$max_lag, nk = dlnm_var$lagnk))

 # Create crossbasis
 cb <- dlnm::crossbasis(london$tmean, lag = dlnm_var$max_lag, argvar, arglag)

 # Seasonality of mortality time series
 seas <- splines::ns(london$date, df = round(8 * length(london$date) / 365.25))

 # Prediction values (equidistant points)
 temp <- round(seq(min(london$tmean), max(london$tmean), by = 0.1), 1)
 # Ensure it falls inside the range of temperatures after rounding:
 temp <- temp[temp >= min(london$tmean) & temp <= max(london$tmean)]

if (check_inla()) {
 # Fit the model
 mod <- bdlnm(mort_75plus ~ cb + factor(dow) + seas, data = london, family = "poisson",
             sample.arg = list(n = 1000, seed = 432))

 # Prediction
```

```
cpred <- bcrosspred(mod, exp_at = temp)

# Perform the plots:

#Overall
plot(cpred, "overall", xlab = "Temperature (ºC)", ylab = "Relative Risk", col = 4,
main="Overall", log = "y")

#3-D plot
plot(cpred, "3d", zlab = "Relative risk", col = 4, lphi = 60, cex.axis = 0.6,
xlab = "Temperature (ºC)", main = "3D graph of temperature effect")

#Contour
plot(cpred, "contour", xlab = "Temperature (ºC)", ylab = "Lag", main="Contour plot")

#Slices (for a high temperature)
htemp <- 23
plot(cpred , "slices", exp_at = htemp, col=3, ylab="RR",
main=paste0("Association for a high temperature (", htemp, "ºC)"))

#Slices (for lag 0)
plot(cpred , "slices", lag_at = 0, col=4, ylab="RR", main=paste0("Association at Lag 0"))

}
```

`plot.optimal_exposure` *Plot posterior distribution of optimal effect exposure values*

Description

It plots a histogram of the posterior distribution of the optimal effect exposure values returned by `optimal_exposure()`.

Usage

```
## S3 method for class 'optimal_exposure'
plot(x, show_median = TRUE, vline.arg = NULL, ...)
```

Arguments

`x`	An object of class `"optimal_exposure"` returned by `optimal_exposure()`.
`show_median`	Logical. If `TRUE` (default) the function draws a vertical line for the median of all the posterior samples. If `FALSE` it doesn't draw any additional line.
`vline.arg`	Optional list of graphical arguments passed to `graphics::abline()` when drawing the median vertical line.
`...`	Optional graphical parameters passed to `graphics::hist()`.

Details

The histogram uses the original prediction grid in `attr(object, "xvar")` as the x-axis values, ensuring that the bars align with prediction exposure values. The function plots the posterior distribution of the optimal exposure values (stored in `x$est`) and highlights the posterior median across samples (stored in `x$summary[["0.5quant"]]`) with a vertical line if `show_median = TRUE`. Use `vline.arg` to change the appearance of that line passed to `graphics::abline()` and `...` to change the graphical parameters of the histogram passed to `graphics::hist()` (to control axis labels, title, colours, etc.). See the original functions for a complete list of the arguments. Some arguments, if not specified, are set to different default values than the original functions.

Value

No return value, called for side effects.

Author(s)

Pau Satorra, Marcos Quijal-Zamorano.

References

Quijal-Zamorano M., Martinez-Beneito M.A., Ballester J., Marí-Dell'Olmo M. (2024). Spatial Bayesian distributed lag non-linear models (SB-DLNM) for small-area exposure-lag-response epidemiological modelling. *International Journal of Epidemiology*, 53(3), dyae061. doi:10.1093/ije/dyae061.

Gasparrini A. (2011). Distributed lag linear and non-linear models in R: the package dlnm. *Journal of Statistical Software*, 43(8), 1-20. doi:10.18637/jss.v043.i08.

Armstrong B. Models for the relationship between ambient temperature and daily mortality. Epidemiology. 2006;17(6):624-31. doi:10.1097/01.ede.0000239732.50999.8f.

See Also

`optimal_exposure()` to estimate exposure values that optimize the predicted effect for a `"bdlnm"` object.

Examples

```
# Set exposure-response and lag-response spline parameters
 dlnm_var <- list(
   var_prc = c(10, 75, 90),
   var_fun = "ns",
   lag_fun = "ns",
   max_lag = 21,
   lagnk = 3
 )


# Set cross-basis parameters
 argvar <- list(fun = dlnm_var$var_fun,
                knots = stats::quantile(london$tmean,
                                        dlnm_var$var_prc/100, na.rm = TRUE),
```

```
                Bound = range(london$tmean, na.rm = TRUE))

 arglag <- list(fun = dlnm_var$lag_fun,
                knots = dlnm::logknots(dlnm_var$max_lag, nk = dlnm_var$lagnk))

 # Create crossbasis
 cb <- dlnm::crossbasis(london$tmean, lag = dlnm_var$max_lag, argvar, arglag)

 # Seasonality of mortality time series
 seas <- splines::ns(london$date, df = round(8 * length(london$date) / 365.25))

 # Prediction values (equidistant points)
 temp <- round(seq(min(london$tmean), max(london$tmean), by = 0.1), 1)
 # Ensure it falls inside the range of temperatures after rounding:
 temp <- temp[temp >= min(london$tmean) & temp <= max(london$tmean)]

if (check_inla()) {
 # Fit the model
 mod <- bdlnm(mort_75plus ~ cb + factor(dow) + seas, data = london,
 family = "poisson", sample.arg = list(n = 1000, seed = 432))

 # Find minimum risk exposure value
 mmt <- optimal_exposure(mod, exp_at = temp)

 # Plot
 plot(mmt, xlab = "Temperature (ºC)",
 main = paste0("MMT (Median = ", round(mmt$summary[["0.5quant"]], 1), "ºC)"))

}
```

Index