Package: metagam (via r-universe)

August 20, 2024

Type Package

Title Meta-Analysis of Generalized Additive Models

Version 0.4.0.9000

Description Meta-analysis of generalized additive models and generalized additive mixed models. A typical use case is when data cannot be shared across locations, and an overall meta-analytic fit is sought. 'metagam' provides functionality for removing individual participant data from models computed using the 'mgcv' and 'gamm4' packages such that the model objects can be shared without exposing individual data. Furthermore, methods for meta-analysing these fits are provided. The implemented methods are described in Sorensen et al. (2020), <doi:10.1016/j.neuroimage.2020.117416>, extending previous works by Schwartz and Zanobetti (2000) and Crippa et al. (2018) <doi:10.6000/1929-6029.2018.07.02.1>.

License GPL-3

Encoding UTF-8

Imports mgcv, ggplot2, metafor, rlang

RoxygenNote 7.3.2

Suggests roxygen2, knitr, rmarkdown, devtools, covr, testthat (>= 2.1.0)

URL https://lifebrain.github.io/metagam/, https://github.com/Lifebrain/metagam

BugReports https://github.com/Lifebrain/metagam/issues

VignetteBuilder knitr

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

RemoteUrl https://github.com/Lifebrain/metagam

RemoteRef HEAD

RemoteSha 60f4f988a175e4d0e93ac8ac8ab6aa68fc264d49

Contents

`metagam-package` *metagam: Meta-analysis of generalized additive models.*

Description

The main functions in the metagam package are described below.

Stripping rawdata

The function `strip_rawdata` takes a fit produced by functions in packages `mgcv` or `gamm4` and removes all individual participants data.

Meta-analysis

The function `metagam` takes a list of fits produced by `strip_rawdata` and computes meta-analytic fits.

Plotting

The functions `plot_dominance` and `plot_heterogeneity` can be used to study the meta-analytic fit computed by `strip_rawdata`.

Author(s)

Maintainer: Oystein Sorensen <oystein.sorensen@psykologi.uio.no> (ORCID)

Authors:

- Andreas M. Brandmaier <brandmaier@mpib-berlin.mpg.de> (ORCID)
- Athanasia Mo Mowinckel <a.m.mowinckel@psykologi.uio.no> (ORCID)

See Also

Useful links:

- https://lifebrain.github.io/metagam/
- https://github.com/Lifebrain/metagam
- Report bugs at https://github.com/Lifebrain/metagam/issues

`metagam` *Meta-analysis of generalized additive models*

Description

Meta-analysis of generalized additive models

Usage

```
metagam(
  models,
  grid = NULL,
  grid_size = 100,
  type = "iterms",
  terms = NULL,
  method = "FE",
  nsim = NULL,
  ci_alpha = 0.05,
  restrict_range = NULL
)
```

Arguments

`models`	List of generalized additive models, each of which has been returned by `strip_rawdata`. If the list is named, the names will be used in the output.
`grid`	Grid of values of the explanatory variables over which to compute the estimated smooth terms. Defaults to `NULL`, which means that a grid is set up for the smooth terms defined by the `terms` argument, with length given by `grid_size` for numeric variables and a single value of each factor variable.
`grid_size`	Numeric value giving the number of elements to use in the grid of explanatory variables when `grid=NULL`. When multiple terms are supplied, each combination of values of explanatory variables are generated, and the number of grid points becomes `grid_size` to the power of the number of terms.
`type`	Type of prediction to use. Defaults to `"iterms"`. Available options are `"iterms"`, `"link"`, and `"response"`. See the documentation of `mgcv::predict.gam` for details. Note that `type="terms"` is not supported, since it may result in estimated zero standard deviation for smooth terms.

terms	Character vector of terms, smooth or parametric, to be included in function estimate. Only used if type="iterms". Defaults to NULL, which means that the first smooth term when listed in alphabetic order is taken.
method	Method of meta analysis, passed on to metafor::rma.uni. Defaults to "FE". See the documentation to metafor::rma for all available options.
nsim	Number of simulations to conduct in order to compute p-values and simultaneous confidence bands for the meta-analytic fit. Defaults to NULL, which means that no simulations are performed. Only used if type="iterms".
ci_alpha	Significance level for simultaneous confidence bands. Ignored if nsim is NULL, and defaults to 0.05.
restrict_range	Character vector of explanatory variables to restrict such that only values within the range for each cohort contribute to the meta-analysis. Default to NULL, which means that each model contributes across the whole range specified by grid. Currently not implemented.

Details

It is currently assumed that all models have been fit with the same smooth terms, although they do not need to have the same basis functions or knot placement. Future versions will also include meta-analysis of parametric terms in the models.

p-values are truncated below at 1e-16 before computing meta-analytic p-values to ensure that no values are identically zero, which would imply that the alternative hypothesis be true with no uncertainty.

Value

An object of type metagam.

Examples

```
library(metagam)
library(mgcv)

## Create 5 datasets
set.seed(1234)
datasets <- lapply(1:5, function(x) gamSim(scale = 5, verbose = FALSE))

## Fit a GAM in each dataset, then use strip_rawdata() to remove
## individual participant data
models <- lapply(datasets, function(dat){
  ## This uses the gam() function from mgcv
  model <- gam(y ~ s(x0, bs = "cr") + s(x1, bs = "cr") + s(x2, bs = "cr"), data = dat)
  ## This uses strip_rawdata() from metagam
  strip_rawdata(model)
})

## Next, we meta-analyze the models.
## It is often most convenient to analyze a single term at a time. We focus on s(x1).
meta_analysis <- metagam(models, terms = "s(x1)", grid_size = 30)
```

```
## We can print some information
summary(meta_analysis)

## We can plot the meta-analytic fit together with the individual fits
plot(meta_analysis)
plot(meta_analysis, ci = "pointwise")

## We can also compute p-values and simultaneous confidence intervals, by setting the nsim argument.
## For details, see the separate vignette.
## Not run:
  meta_analysis <- metagam(models, terms = "s(x0)", grid_size = 30, nsim = 1000)
  summary(meta_analysis)
  plot(meta_analysis, ci = "both")
  plot(meta_analysis, ci = "simultaneous")

## End(Not run)
```

`plot.metagam` *Plot estimated smooth terms*

Description

Plot the meta-analytic estimate of a smooth term along with the separate fits in each cohort.

Usage

```
## S3 method for class 'metagam'
plot(x, term = NULL, ci = "none", legend = FALSE, only_meta = FALSE, ...)
```

Arguments

`x`	Object returned by `metagam`.
`term`	The smooth term to plot. Defaults to `NULL`, which means that the first term is plotted.
`ci`	Type of confidence bands to plot around the meta-analytic fit. Defaults to "none", which means the no bands are plotted. Other options are "simultaneous", "pointwise", and "both". Simultaneous confidence bands require that `metagam` was run with `nsim` not equal to `NULL`.
`legend`	Logical specifying whether or not to plot a legend. Defaults to `FALSE`.
`only_meta`	Logical specifying whether to include the fits for each study, or to only plot the meta-analytic fit. Defaults to `FALSE`.
`...`	Other arguments to plot.

Value

The function is called for its side effect of producing a plot.

Examples

```
library(metagam)
library(mgcv)

## Create 5 datasets
set.seed(1234)
datasets <- lapply(1:5, function(x) gamSim(scale = 5, verbose = FALSE))

## Fit a GAM in each dataset, then use strip_rawdata() to remove
## individual participant data
models <- lapply(datasets, function(dat){
  ## This uses the gam() function from mgcv
  model <- gam(y ~ s(x0, bs = "cr") + s(x1, bs = "cr") + s(x2, bs = "cr"), data = dat)
  ## This uses strip_rawdata() from metagam
  strip_rawdata(model)
})

## Next, we meta-analyze the models.
## It is often most convenient to analyze a single term at a time. We focus on s(x1).
meta_analysis <- metagam(models, terms = "s(x1)", grid_size = 30)

## We can print some information
summary(meta_analysis)

## We can plot the meta-analytic fit together with the individual fits
plot(meta_analysis)
plot(meta_analysis, ci = "pointwise")

## We can also compute p-values and simultaneous confidence intervals, by setting the nsim argument.
## For details, see the separate vignette.
## Not run:
  meta_analysis <- metagam(models, terms = "s(x0)", grid_size = 30, nsim = 1000)
  summary(meta_analysis)
  plot(meta_analysis, ci = "both")
  plot(meta_analysis, ci = "simultaneous")

## End(Not run)
```

---

plot_between_study_sd *Plot between-study standard deviation*

Description

When a random effects meta analysis has been used, this function visualizes how the between-study standard deviation depends on the explanatory variable.

Usage

```
plot_between_study_sd(x)
```

Arguments

x	Object returned from metagam.

Value

A ggplot object.

Examples

```
library("mgcv")
set.seed(1233)
shifts <- c(0, .5, 1, 0, -1)
datasets <- lapply(shifts, function(x) {
  ## Simulate data
  dat <- gamSim(scale = .1, verbose = FALSE)
  ## Add a shift
  dat$y <- dat$y + x * dat$x2^2
  ## Return data
  dat
})

models <- lapply(datasets, function(dat){
  b <- gam(y ~ s(x2, bs = "cr"), data = dat)
  strip_rawdata(b)
})

meta_analysis <- metagam(models, method = "REML")

plot_between_study_sd(meta_analysis)
```

---

plot_dominance *Dominance plot*

Description

Plots the (relative) contribution of the individual GAMs to each data point on a given axis. It shows whether and how parts of the axis are dominated by certain individual GAMs.

Usage

```
plot_dominance(x, term = NULL, relative = TRUE, width = NULL)
```

Arguments

x	Object returned by metagam.
term	Character specifying which smooth term to plot. Default to NULL which means that the first term (in alphabetic order) is taken.
relative	Logical specifying whether to have relative or absolute scales. Defaults to TRUE.
width	Width of bars. Default to NULL, which means it is automatically determined based on the minimum grid spacing in x.

Value

A ggplot object.

Examples

```
# See the vignette, either at https://lifebrain.github.io/metagam/articles/articles/dominance.html
# or by typing the following in the console:
# vignette("Dominance")
```

---

plot_heterogeneity *Heterogeneity Plot*

Description

Heterogeneity Plot

Usage

```
plot_heterogeneity(x, term = NULL, type = "Q", alpha_thresh = 0.05)
```

Arguments

x	Object returned by metagam.
term	Character specifying which smooth term to plot. Defaults to NULL; if x was fitted with a single term, this one is taken.
type	Character specifying which type of plot. Either "Q" for the test statistic or "p" for the p-value. Defaults to "Q".
alpha_thresh	Significance level. Defaults to .05.

Details

This plot visualizes the heterogeneity along the given axis, using Cochran's Q test.

Value

A ggplot object.

Examples

```
# See the vignette, either at https://lifebrain.github.io/metagam/articles/heterogeneity.html
# or by typing the following in the console:
# vignette("heterogeneity")
```

`print.metagam` *Print method for metagam objects.*

Description

Print method for metagam objects.

Usage

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

Arguments

`x`	Object of class metagam.
`...`	Other arguments (not used).

Value

The function invisibly returns its input argument `x`.

`print.striprawdata` *Print method for striprawdata*

Description

Print method for striprawdata

Usage

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

Arguments

`x`	Object of class `striprawdata`.
`...`	Other arguments.

Value

The function invisibly returns its argument.

print.summary.metagam *Print output from summary of metagam fit.*

Description

Print output from summary of metagam fit.

Usage

```
## S3 method for class 'summary.metagam'
print(x, digits = 8, ...)
```

Arguments

x	A summary.metagam object
digits	Number of digits to print for meta-analytic p-values
...	Other arguments

Value

The function invisibly returns its input argument x.

strip_rawdata *Strip rawdata from a generalized additive model*

Description

This function removes all individual participant data from a generalized additive model object, while keeping aggregated quantities. The resulting object can be shared without exposing individual participant data.

Usage

```
strip_rawdata(model, path = NULL, save_ranges = TRUE, ...)

## S3 method for class 'list'
strip_rawdata(model, path = NULL, save_ranges = TRUE, ...)

## S3 method for class 'gamm'
strip_rawdata(model, path = NULL, save_ranges = TRUE, ...)

## S3 method for class 'bam'
strip_rawdata(model, path = NULL, save_ranges = TRUE, ...)

## S3 method for class 'gam'
strip_rawdata(model, path = NULL, save_ranges = TRUE, ...)
```

Arguments

model	A model fitted using mgcv::gam, mgcv::bam, mgcv::gamm, or gamm4::gamm4.
path	Optional path in which to save the object as a .rds file.
save_ranges	Logical specifying whether to save the ranges of each variable used by the model. For numeric variables this amounts to the minimum and maximum, and for factors all levels are saved. The values will be in the list element var.summary of the returned object.
...	Other arguments (not used).

Details

Thin plate regression splines (bs='tp' and bs='ts') and Duchon splines bs='ds' are currently not supported, since for these splines mgcv requires the unique values of the explanatory variables for each smooth term for the predict method to work. Future updates to this package will fix this.

Value

Model object with individual participant data removed.

Methods (by class)

- strip_rawdata(list): Strip rawdata from list object returned by gamm4
- strip_rawdata(gamm): Strip rawdata from gamm object
- strip_rawdata(bam): Strip rawdata from gam object
- strip_rawdata(gam): Strip rawdata from gam object

Examples

```
library(metagam)
library(mgcv)

## Create 5 datasets
set.seed(1234)
datasets <- lapply(1:5, function(x) gamSim(scale = 5, verbose = FALSE))

## Fit a GAM in each dataset, then use strip_rawdata() to remove
## individual participant data
models <- lapply(datasets, function(dat){
  ## This uses the gam() function from mgcv
  model <- gam(y ~ s(x0, bs = "cr") + s(x1, bs = "cr") + s(x2, bs = "cr"), data = dat)
  ## This uses strip_rawdata() from metagam
  strip_rawdata(model)
})

## Next, we meta-analyze the models.
## It is often most convenient to analyze a single term at a time. We focus on s(x1).
meta_analysis <- metagam(models, terms = "s(x1)", grid_size = 30)

## We can print some information
```

```
summary(meta_analysis)

## We can plot the meta-analytic fit together with the individual fits
plot(meta_analysis)
plot(meta_analysis, ci = "pointwise")

## We can also compute p-values and simultaneous confidence intervals, by setting the nsim argument.
## For details, see the separate vignette.
## Not run:
  meta_analysis <- metagam(models, terms = "s(x0)", grid_size = 30, nsim = 1000)
  summary(meta_analysis)
  plot(meta_analysis, ci = "both")
  plot(meta_analysis, ci = "simultaneous")

## End(Not run)
```

`summary.metagam` *Summary method for metagam objects*

Description

Summary method for metagam objects

Usage

```
## S3 method for class 'metagam'
summary(object, ...)
```

Arguments

`object`	A metagam object as returned by `metagam`.
`...`	other arguments (not used).

Value

A list of class `summary.metagam` containing the following information:

- `meta_pvals`: dataframe with p-values from each individual fit. These can be meta-analytically combined using the `metap` package.
- `terms`: smooth terms that have been meta-analyzed.
- `method`: method used for meta-analysis. See the `metafor` package for detailed description.
- `intercept`: logical specifying whether or not the intercept has been included in the meta-analysis.
- `cohorts`: Number of datasets ("cohorts") used in the meta-analysis.

summary.striprawdata *Summary method for GAMs stripped for rawdata*

Description

Summary method for GAMs stripped for rawdata

Usage

```
## S3 method for class 'striprawdata'
summary(object, ...)
```

Arguments

object	Object returned by strip_rawdata.
...	Other arguments.

Value

The function returns its input argument, which is printed to the console.

Index