

# Package ‘metagam’

November 14, 2021

**Type** Package

**Title** Meta-Analysis of Generalized Additive Models

**Version** 0.3.1

**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](https://doi.org/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](https://doi.org/10.6000/1929-6029.2018.07.02.1)>.

**License** GPL-3

**Encoding** UTF-8

**Imports** mgcv, ggplot2, metafor, rlang

**RoxygenNote** 7.1.2

**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

**NeedsCompilation** no

**Author** Oystein Sorensen [aut, cre] (<<https://orcid.org/0000-0003-0724-3542>>),  
Andreas M. Brandmaier [aut] (<<https://orcid.org/0000-0001-8765-6982>>),  
Athanasia Mo Mowinckel [aut] (<<https://orcid.org/0000-0002-5756-0223>>)

**Maintainer** Oystein Sorensen <oystein.sorensen@psykologi.uio.no>

**Repository** CRAN

**Date/Publication** 2021-11-14 07:50:02 UTC

## R topics documented:

metagam-package . . . . .	2
metagam . . . . .	3
plot.metagam . . . . .	5
plot_dominance . . . . .	6
plot_heterogeneity . . . . .	7
print.metagam . . . . .	8
print.striprawdata . . . . .	8
print.summary.metagam . . . . .	9
strip_rawdata . . . . .	9
summary.metagam . . . . .	11
summary.striprawdata . . . . .	12
<b>Index</b>	<b>13</b>

---

metagam-package	<i>metagam: Meta-analysis of generalized additive models.</i>
-----------------	---

---

### Description

The main functions in the metagam package are described below.

### Stripping rawdata

The function `strip_rawdata` takes a fit produced by the mgcv package 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`.

---

metagam	<i>Meta-analysis of generalized additive models</i>
---------	---

---

## 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,
  intercept = FALSE,
  restrict_range = NULL
)
```

## Arguments

models	List of generalized additive models, each of which has been returned by <a href="#">strip_rawdata</a> . 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 <code>terms</code> argument, with length given by <code>grid_size</code> 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 <code>grid=NULL</code> . When multiple terms are supplied, each combination of values of explanatory variables are generated, and the number of grid points becomes <code>grid_size</code> 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 <code>mgcv::predict.gam</code> for details. Note that <code>type="terms"</code> 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 <code>type="iterms"</code> . 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 <code>metafor::rma.uni</code> . Defaults to "FE". See the documentation to <code>metafor::rma</code> 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.

ci_alpha	Significance level for simultaneous confidence bands. Ignored if nsim is NULL, and defaults to 0.05.
intercept	logical defining whether or not to include the intercept in each smooth term. Only applies when type = "iterms".
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 fit
plot(meta_analysis)
```

```
## We can also compute p-values and simultaneous confidence intervals, by setting the nsim argument.
## For details, see the separate vignette.
meta_analysis <- metagam(models, terms = "s(x0)", grid_size = 30, nsim = 1000)
summary(meta_analysis)
```

---

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, ...)
```

## Arguments

x                    Object returned by `metagam`.  
...                   Other arguments to plot.

## Details

This function currently works for meta-analytic estimates of a single smooth term, which can be either univariate or bivariate. It also works for alternatively meta-analysis of response or link functions.

## Value

A ggplot object plotting a smooth term of interest along an axis. The meta-analytic fit is shown as a solid black line, and the cohort fits are shown as dashed lines, separated by color codes.

## 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 fit
plot(meta_analysis)

## We can also compute p-values and simultaneous confidence intervals, by setting the nsim argument.
## For details, see the separate vignette.
meta_analysis <- metagam(models, terms = "s(x0)", grid_size = 30, nsim = 1000)
summary(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, axis = NULL, term = NULL, relative = TRUE, width = NULL)
```

## Arguments

x	Object returned by <code>metagam</code> .
axis	Character specifying which variable to plot. Defaults to NULL; if x was fitted with a single term, the explanatory variable corresponding to this term is selected.
term	Character specifying which smooth term to plot. Default to NULL; if x was fitted with a single term, this one 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,
  axis = NULL,
  term = NULL,
  type = "Q",
  alpha_thresh = 0.05
)
```

## Arguments

x	Object returned by <code>metagam</code> .
axis	Character specifying which variable to plot. Defaults to NULL; if x was fitted with a single term, the explanatory variable corresponding to this term is selected.
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.stripawdata  Print method for stripawdata
```

---

**Description**

Print method for stripawdata

**Usage**

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

**Arguments**

x	Object of class stripawdata.
...	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 <code>mgcv::gam</code> , <code>mgcv::bam</code> , <code>mgcv::gamm</code> , or <code>gamm4::gamm4</code> .
path	Optional path in which to save the object as a <code>.rds</code> 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 <code>var.summary</code> 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)**

- `list`: Strip rawdata from list object returned by `gamm4`
- `gamm`: Strip rawdata from `gamm` object
- `bam`: Strip rawdata from `gam` object
- `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 fit
plot(meta_analysis)

## We can also compute p-values and simultaneous confidence intervals, by setting the nsim argument.
## For details, see the separate vignette.
meta_analysis <- metagam(models, terms = "s(x0)", grid_size = 30, nsim = 1000)
summary(meta_analysis)
```

---

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 <code>metagam</code> .
...	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.stripawdata *Summary method for GAMs stripped for rawdata*

---

**Description**

Summary method for GAMs stripped for rawdata

**Usage**

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

**Arguments**

object	Object returned by <a href="#">strip_rawdata</a> .
...	Other arguments.

**Value**

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

# Index

`.data` (metagam-package), 2  
`:=` (metagam-package), 2

`as_label` (metagam-package), 2  
`as_name` (metagam-package), 2

`enquo` (metagam-package), 2  
`enquos` (metagam-package), 2  
`expr` (metagam-package), 2

`metagam`, 2, 3, 5–7, 11  
`metagam-package`, 2

`plot.metagam`, 5  
`plot_dominance`, 2, 6  
`plot_heterogeneity`, 2, 7  
`print.metagam`, 8  
`print.stripawdata`, 8  
`print.summary.metagam`, 9

`strip_rawdata`, 2, 3, 9, 12  
`summary.metagam`, 11  
`summary.stripawdata`, 12  
`sym` (metagam-package), 2  
`syms` (metagam-package), 2