Package ‘autohrf’

July 22, 2025

Type Package

Title Automated Generation of Data-Informed GLM Models in Task-Based fMRI Data Analysis

Version 1.1.3

Maintainer Jure Demšar <jure.demsar@fri.uni-lj.si>

Description
Analysis of task-related functional magnetic resonance imaging (fMRI) activity at the level of individual participants is commonly based on general linear modelling (GLM) that allows us to estimate to what extent the blood oxygenation level dependent (BOLD) signal can be explained by task response predictors specified in the GLM model. The predictors are constructed by convolving the hypothesised timecourse of neural activity with an assumed hemodynamic response function (HRF). To get valid and precise estimates of task response, it is important to construct a model of neural activity that best matches actual neuronal activity. The construction of models is most often driven by predefined assumptions on the components of brain activity and their duration based on the task design and specific aims of the study. However, our assumptions about the onset and duration of component processes might be wrong and can also differ across brain regions. This can result in inappropriate or suboptimal models, bad fitting of the model to the actual data and invalid estimations of brain activity. Here we present an approach in which theoretically driven models of task response are used to define constraints based on which the final model is derived computationally using the actual data. Specifically, we developed 'autohrf' — a package for the 'R' programming language that allows for data-driven estimation of HRF models. The package uses genetic algorithms to efficiently search for models that fit the underlying data well. The package uses automated parameter search to find the onset and duration of task predictors which result in the highest fitness of the resulting GLM based on the fMRI signal under predefined restrictions. We evaluate the usefulness of the 'autohrf' package on publicly available datasets of task-related fMRI activity. Our results suggest that by using 'autohrf' users can find better task related brain activity models in a quick and efficient manner.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Imports cowplot (>= 1.1.1), doParallel (>= 1.0.17), dplyr (>= 1.0.8), foreach (>= 1.5.2), ggplot2 (>= 3.3.5), gtools (>= 3.9.2), lubridate (>= 1.8.0), magrittr (>= 2.0.2), RColorBrewer (>=

1.1)

Suggests knitr (>= 1.38), testthat (>= 3.1.3)

VignetteBuilder knitr

RoxygenNote 7.2.0

URL https://github.com/demsarjure/autohrf

BugReports https://github.com/demsarjure/autohrf/issues

NeedsCompilation no

Author Jure Demšar [cre, aut],
Nina Purg [aut],
Grega Repovš [aut]

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2024-01-16 14:00:11 UTC

Contents

autohrf *autohrf*

Description

A function that automatically finds the parameters of model's that best match the underlying data.

Usage

```
autohrf(
  d,
  model_constraints,
  tr,
  roi_weights = NULL,
  allow_overlap = FALSE,
  population = 100,
  iter = 100,
  mutation_rate = 0.1,
  mutation_factor = 0.05,
  elitism = 0.1,
  hrf = "spm",
  t = 32,
  p_boynton = c(2.25, 1.25, 2),
  p_spm = c(6, 16, 1, 1, 6, 0),
  f = 100,
  cores = NULL,
  autohrf = NULL,
  verbose = TRUE
)
```

Arguments

`d`	A dataframe with the signal data: roi, t and y. ROI is the name of the region, t is the timestamp and y the value of the signal.
`model_constraints`	A list of model specifications to use for fitting. Each specification is represented as a data frame containing information about it (event, start_time, end_time, min_duration and max_duration).
`tr`	MRI's repetition time.
`roi_weights`	A data frame with ROI weights: roi, weight. ROI is the name of the region, weight a number that defines the importance of that roi, the default weight for a ROI is 1. If set to 2 for a particular ROI that ROI will be twice as important.
`allow_overlap`	Whether to allow overlap between events.
`population`	The size of the population in the genetic algorithm.
`iter`	Number of iterations in the genetic algorithm.

`mutation_rate` The mutation rate in the genetic algorithm.

`mutation_factor` The mutation factor in the genetic algorithm.

`elitism` The degree of elitism (promote a percentage of the best solutions) in the genetic algorithm.

`hrf` Method to use for HRF generation.

`t` The t parameter for Boynton or SPM HRF generation.

`p_boynton` Parameters for the Boynton's HRF.

`p_spm` Parameters for the SPM HRF.

`f` Upsampling factor.

`cores` Number of cores to use for parallel processing. Set to the number of provided model constraints by default.

`autohrf` Results of a previous autohrf run to continue.

`verbose` Whether to print progress of the fitting process.

Value

A list containing model fits for each of the provided model specifications.

Examples

```
# prepare model specs
model3 <- data.frame(
  event       = c("encoding", "delay", "response"),
  start_time  = c(0,          2.65,     12.5),
  end_time    = c(3,          12.5,     16)
)

model4 <- data.frame(
  event       = c("fixation", "target", "delay", "response"),
  start_time  = c(0,          2.5,      2.65,    12.5),
  end_time    = c(2.5,        3,        12.5,    15.5)
)

model_constraints <- list(model3, model4)

# run autohrf
df <- flanker
autofit <- autohrf(df, model_constraints, tr = 2.5,
                   population = 2, iter = 2, cores = 1)
```

`autohrf-datasets` *Datasets for autohrf examples Example datasets for use in* **autohrf** *examples and vignettes. The datasets were extracted from the internal Mind and Brain Lab's (MBLab,* `http://www.mblab.si` *repository. MBLab is a research lab at the Faculty of Arts, Department of Psychology, University of Ljubljana, Slovenia.*

Description

Datasets for autohrf examples Example datasets for use in **autohrf** examples and vignettes. The datasets were extracted from the internal Mind and Brain Lab's (MBLab, `http://www.mblab.si` repository. MBLab is a research lab at the Faculty of Arts, Department of Psychology, University of Ljubljana, Slovenia.

Format

`swm` fMRI dataset for a spatial working memory experiment.

Source: Internal MBLab repository.

11520 obs. of 3 variables

- `roi` region of interest.
- `time` time stamp.
- y BOLD value.

`swm_autofit` Stored results from a pre-completed autohrf run.

Source: Internal MBLab repository.

`swm_autofit1` Stored results from a pre-completed autohrf run.

Source: Internal MBLab repository.

`swm_autofit2` Stored results from a pre-completed autohrf run.

Source: Internal MBLab repository.

`flanker` fMRI dataset for a flanker experiment.

Source: Internal MBLab repository.

192 obs. of 3 variables

- `roi` region of interest.
- `time` time stamp.
- y BOLD value.

`flanker_autofit` Stored results from a pre-completed autohrf run.

Source: Internal MBLab repository.

Examples

```
# load swm data
data_swm <- swm

# load the previously completed autofits
```

```
autofit <- swm_autofit
autofit1 <- swm_autofit1
autofit2 <- swm_autofit2

# load flanker data
data_flanker <- flanker

# load the previously completed autofits
autofit3 <- flanker_autofit
```

---

convolve_events *convolve_events*

Description

A helper function for convolving events of a model with a generated HRF signal.

Usage

```
convolve_events(
  model,
  tr,
  max_duration,
  hrf = "spm",
  t = 32,
  p_boynton = c(2.25, 1.25, 2),
  p_spm = c(6, 16, 1, 1, 6, 0),
  f = 100
)
```

Arguments

model	A data frame containing information about the model to use and its events (event, start_time and duration).
tr	MRI's repetition time.
max_duration	Maximum duration of the signal.
hrf	Method to use for HRF generation, can be "boynton" or "spm".
t	The t parameter for Boynton or SPM HRF generation.
p_boynton	Parameters for the Boynton's HRF.
p_spm	Parameters for the SPM HRF.
f	Upsampling factor.

Value

Returns a list with the convolved signal and time series.

convolve_hrf *convolve_hrf*

Description

A helper function for convolving HRF with a signal.

Usage

```
convolve_hrf(y, hrf_s)
```

Arguments

y	The signal.
hrf_s	The HRF.

Value

Returns the convolution between HRF and the signal.

create_boynton_hrf *create_boynton_hrf*

Description

A helper function for creating a Boynton HRF.

Usage

```
create_boynton_hrf(tr, t = 32, p = c(2.25, 1.25, 2))
```

Arguments

tr	MRI's repetition time.
t	The t parameter for Boynton or SPM HRF generation.
p	Parameters for the Boynton's HRF.

Value

Returns a Boynton HRF function.

create_child *create_child*

Description

A helper function for creating a child from parents.

Usage

```
create_child(
  start_time,
  end_time,
  n_events,
  mutation_rate,
  mutation_factor,
  current_model,
  p1,
  p2,
  allow_overlap
)
```

Arguments

start_time	A list with model's event start times.
end_time	A list with model's event end times.
n_events	Number of events in the model.
mutation_rate	The mutation rate in the genetic algorithm.
mutation_factor	The mutation factor in the genetic algorithm.
current_model	The constraints of the current model.
p1	The first selected parent.
p2	The second selected parent.
allow_overlap	Whether to allow overlap between events.

Value

A child model created from two parents.

`create_first_generation` *create_first_generation*

Description

A helper function for creating the first generation.

Usage

```
create_first_generation(current_model, n_events, population, allow_overlap)
```

Arguments

`current_model`	The constraints of the current model.
`n_events`	Number of events in the model.
`population`	The size of the population in the genetic algorithm.
`allow_overlap`	Whether to allow overlap between events.

Value

Returns the first generation of models.

`create_new_generation` *create_new_generation*

Description

A helper function for creating a new generation of possible solutions.

Usage

```
create_new_generation(
  elitism,
  population,
  start_time,
  end_time,
  fitness,
  n_events,
  mutation_factor,
  mutation_rate,
  current_model,
  allow_overlap
)
```

Arguments

`elitism`	The degree of elitism (promote a percentage of the best solutions) in the genetic algorithm.
`population`	The size of the population in the genetic algorithm.
`start_time`	A list with model's event start times.
`end_time`	A list with model's event end times.
`fitness`	A fitness score of all candidate models.
`n_events`	Number of events in the model.
`mutation_factor`	The mutation factor in the genetic algorithm.
`mutation_rate`	The mutation rate in the genetic algorithm.
`current_model`	The constraints of the current model.
`allow_overlap`	Whether to allow overlap between events.

Value

A new generation of candidate models.

`create_spm_hrf` *create_boynton_hrf*

Description

A helper function for creating a SPM HRF.

Usage

```
create_spm_hrf(tr, t = 32, p = c(6, 16, 1, 1, 6, 0))
```

Arguments

`tr`	MRI's repetition time.
`t`	The t parameter for Boynton or SPM HRF generation.
`p`	Parameters for the SPM HRF.

Value

Returns a SPM HRF function.

downsample — *downsample*

Description

A helper function for downsampling a given signal.

Usage

```
downsample(y, f = 100)
```

Arguments

y	The signal.
f	Upsampling factor.

Value

Returns the downsampled signal.

evaluate_model — *evaluate_model*

Description

A function for evaluating the model against the data.

Usage

```
evaluate_model(
  d,
  model,
  tr,
  roi_weights = NULL,
  hrf = "spm",
  t = 32,
  p_boynton = c(2.25, 1.25, 2),
  p_spm = c(6, 16, 1, 1, 6, 0),
  f = 100,
  verbose = TRUE
)
```

Arguments

`d`	A dataframe with the signal data: roi, t and y. ROI is the name of the region, t is the timestamp and y the value of the signal.
`model`	A data frame containing information about the model to use and its events (event, start_time and duration).
`tr`	MRI's repetition time.
`roi_weights`	A data frame with ROI weights: roi, weight. ROI is the name of the region, weight a number that defines the importance of that roi, the default weight for a ROI is 1. If set to 2 for a particular ROI that ROI will be twice as important.
`hrf`	Method to use for HRF generation, can be "boynton" or "spm".
`t`	The t parameter for Boynton or SPM HRF generation.
`p_boynton`	Parameters for the Boynton's HRF.
`p_spm`	Parameters for the SPM HRF.
`f`	Upsampling factor.
`verbose`	Whether to print a report of the evaluation results.

Value

Returns a list that contains the model, fits of events for each ROI, convolved events, TR and evaluation scores for each ROI.

Examples

```
# create the model
m <- data.frame(event = c("encoding", "delay", "response"),
start_time = c(0, 2.5, 12.5), duration = c(2.5, 10, 5))

# evaluate
df <- flanker
res <- evaluate_model(df, m, tr = 2.5)
```

`fit_to_constraints` *fit_to_constraints*

Description

A helper function for fitting a model to constraints.

Usage

```
fit_to_constraints(
  model_id,
  d,
  model_constraints,
  tr,
  roi_weights,
  allow_overlap,
  population,
  iter,
  mutation_rate,
  mutation_factor,
  elitism,
  hrf,
  t,
  p_boynton,
  p_spm,
  f,
  autohrf = NULL,
  verbose = TRUE
)
```

Arguments

`model_id`	ID of the model.
`d`	A dataframe with the signal data: roi, t and y. ROI is the name of the region, t is the timestamp and y the value of the signal.
`model_constraints`	A list of model specifications to use for fitting. Each specification is represented as a data frame containing information about it (event, start_time, end_time, min_duration and max_duration).
`tr`	MRI's repetition time.
`roi_weights`	A data frame with ROI weights: roi, weight. ROI is the name of the region, weight a number that defines the importance of that roi, the default weight for a ROI is 1. If set to 2 for a particular ROI that ROI will be twice as important.
`allow_overlap`	Whether to allow overlap between events.
`population`	The size of the population in the genetic algorithm.
`iter`	Number of iterations in the genetic algorithm.
`mutation_rate`	The mutation rate in the genetic algorithm.
`mutation_factor`	The mutation factor in the genetic algorithm.
`elitism`	The degree of elitism (promote a percentage of the best solutions) in the genetic algorithm.
`hrf`	Method to use for HRF generation.
`t`	The t parameter for Boynton or SPM HRF generation.

p_boynton	Parameters for the Boynton's HRF.
p_spm	Parameters for the SPM HRF.
f	Upsampling factor.
autohrf	Results of a previous autohrf run to continue.
verbose	Whether to print progress of the fitting process.

Value

Returns the best model given provided constraints.

---

get_best_models *get_best_models*

Description

Returns and prints the best fitted model for each of the specs used in autohrf.

Usage

```
get_best_models(autofit, return_fitness = FALSE, verbose = TRUE)
```

Arguments

autofit	Output of the autohrf function.
return_fitness	Whether to return models or fitness.
verbose	Whether to print information or only return the result.

Value

Returns a list containing the best models for each of the provided constraints.

Examples

```
# prepare model specs
model3 <- data.frame(
  event        = c("encoding", "delay", "response"),
  start_time   = c(0,          2.65,     12.5),
  end_time     = c(3,          12.5,     16)
)

model4 <- data.frame(
  event        = c("fixation", "target", "delay", "response"),
  start_time   = c(0,          2.5,      2.65,    12.5),
  end_time     = c(2.5,        3,        12.5,    15.5)
)

model_constraints <- list(model3, model4)
```

```
# run autohrf
df <- flanker
autofit <- autohrf(df, model_constraints, tr = 2.5,
                   population = 2, iter = 2, cores = 1)

# print best models
get_best_models(autofit)
```

get_parents *get_parents*

Description

A helper function for getting parents for the child model.

Usage

```
get_parents(fitness)
```

Arguments

fitness	A fitness score of all candidate models.

Value

Parents for the child model.

plot_best_models *plot_best_models*

Description

Plots the best fitted model for each of the specs in autohrf.

Usage

```
plot_best_models(autofit, ncol = NULL, nrow = NULL)
```

Arguments

autofit	Output of the autohrf function.
ncol	Number of columns in the plot.
nrow	Number of rows in the plot.

Value

Plots the grid containing a visualization of the best models for each of the provided constraints.

Examples

```
# prepare model specs
model3 <- data.frame(
  event        = c("encoding", "delay", "response"),
  start_time   = c(0,          2.65,     12.5),
  end_time     = c(3,          12.5,     16)
)

model4 <- data.frame(
  event        = c("fixation", "target", "delay", "response"),
  start_time   = c(0,          2.5,      2.65,    12.5),
  end_time     = c(2.5,        3,        12.5,    15.5)
)

model_constraints <- list(model3, model4)

# run autohrf
df <- flanker
autofit <- autohrf(df, model_constraints, tr = 2.5,
                   population = 2, iter = 2, cores = 1)

# plot best models
plot_best_models(autofit)
```

---

plot_events *plot_events*

Description

A helper function for plotting events of a fitted model.

Usage

```
plot_events(af, i = NULL)
```

Arguments

af	The output from the autohrf function.
i	Model index.

Value

Returns a plot of the events.

plot_fitness *plot_fitness*

Description

Plots how fitness changed through iterations of autohrf. Use this to investigate whether your solution converged.

Usage

```
plot_fitness(autofit)
```

Arguments

autofit	Output of the autohrf function.

Value

A ggplot visualization of fitness through time.

Examples

```
# prepare model specs
model3 <- data.frame(
  event        = c("encoding", "delay", "response"),
  start_time   = c(0,          2.65,     12.5),
  end_time     = c(3,          12.5,     16)
)

model4 <- data.frame(
  event        = c("fixation", "target", "delay", "response"),
  start_time   = c(0,          2.5,      2.65,    12.5),
  end_time     = c(2.5,        3,        12.5,    15.5)
)

model_constraints <- list(model3, model4)

# run autohrf
df <- flanker
autofit <- autohrf(df, model_constraints, tr = 2.5,
                   population = 2, iter = 2, cores = 1)

# plot fitness
plot_fitness(autofit)
```

plot_model *plot_model*

Description

Plots a manually constructed model.

Usage

```
plot_model(
  model_evaluation,
  by_roi = FALSE,
  ncol = NULL,
  nrow = NULL,
  scales = "free_y",
  rois = NULL
)
```

Arguments

model_evaluation	The output from the evaluate_model function.
by_roi	Whether to plot the fit for each ROI independently.
ncol	Number of columns in the facet wrap.
nrow	Number of rows in the facet wrap.
scales	Whether to free certain axes of the facet wrap.
rois	A subset of ROIs to visualize.

Value

A ggplot visualization of the model.

Examples

```
# prepare model specs
model3 <- data.frame(event      = c("encoding", "delay", "response"),
                     start_time = c(0,           2.65,    12.5),
                     duration   = c(2.65,        9.85,    3))
```

`run_model` *run_model*

Description

A helper function for evaluating a model.

Usage

```
run_model(d, ce, model, roi_weights = NULL)
```

Arguments

`d`	A dataframe with the signal data: roi, t and y. ROI is the name of the region, t is the timestamp and y the value of the signal.
`ce`	Result of the convolve_events function.
`model`	A data frame containing information about the model to use and its events (event, start_time and duration).
`roi_weights`	A data frame with ROI weights: roi, weight. ROI is the name of the region, weight a number that defines the importance of that roi, the default weight for a ROI is 1. If set to 2 for a particular ROI that ROI will be twice as important.

Value

Returns the model's evaluation.

Index