Package: segmag (via r-universe)

August 22, 2024

Type Package

Version 1.2.4

Date 2016-08-11

Title Determine Event Boundaries in Event Segmentation Experiments

Maintainer Frank Papenmeier <frank.papenmeier@uni-tuebingen.de>

Author Frank Papenmeier [aut, cre], Konstantin Sering [ctb]

Description Contains functions that help to determine event boundaries in event segmentation experiments by bootstrapping a critical segmentation magnitude under the null hypothesis that all key presses were randomly distributed across the experiment. Segmentation magnitude is defined as the sum of Gaussians centered at the times of the segmentation key presses performed by the participants. Within a participant, the maximum of the overlaid Gaussians is used to prevent an excessive influence of a single participant on the overall outcome (e.g. if a participant is pressing the key multiple times in succession). Further functions are included, such as plotting the results.

License GPL (>= 3)

Suggests testthat

Imports Rcpp (>= 0.11.0), utils, stats, grDevices, graphics, plyr

LinkingTo Rcpp

RoxygenNote 5.0.1

NeedsCompilation yes

Date/Publication 2016-08-11 18:40:42

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

RemoteUrl https://github.com/cran/segmag

RemoteRef HEAD

RemoteSha f24540505563b284ed8dfd2651485a55b6fe8304

Contents

`bootstrap_critical_cutoffs` *Bootstrap critical segmentation magnitude*

Description

Bootstraps critical segmentation magnitude values for a `segmag` object under the null hypothesis that all key presses were randomly distributed (uniformly) across the experiment (time_min to time_max).

Usage

```
bootstrap_critical_cutoffs(segmag, n_bootstrap, critical_probs,
  segmag_substract = NULL, visualize = FALSE, save_as = NULL)
```

Arguments

`segmag`	object of class `segmag`
`n_bootstrap`	numeric, number of bootstrap iterations
`critical_probs`	numeric vector of probabilities, e.g. c(.95,.99)
`segmag_substract`	object of class `segmag`. If this value is set than keypress times in segmag and segmag_substract are both randomized and the critical_cutoffs relate to the difference of the segmentation magnitude in segmag minus the segmentation magnitude in segmag_substract
`visualize`	logical, visualize ordered maxima (and minima if segmag_substract is set) of bootstrapping iterations (Note: Enabling this option might require a lot of RAM with large data sets or large values of n_bootstrap)
`save_as`	character, filename where to save raw bootstrapping data and plot (optional)

Details

During each bootstrapping iteration, the key presses are randomly distributed (drawn from uniform distribution ranging from time_min to time_max). Then, segmentation magnitude is calculated with those random key press times (note that ids are retained, that is each participant "makes" the same amount of key presses as in the original experiment). The local maxima in segmentation magnitude resulting from the random key press times are ordered according to their size. The largest maximum is kept.

The function returns the `critical_probs` quantiles of the vector of those largest maxima obtained across `n_bootstap` iterations

This function can also be used to bootstrap the critical maxima and minima cutoffs of a difference function of two segmag objects. To do so, segmag and segmag_substract must be defined. All values will be related to the difference of segmag - segmag_substract (Keypress times in segmag and segmag_substract are randomized independently).

Value

critical segmentation magnitudes; If segmag_substract is NULL, then the return value is a numeric vector. Otherwise a list with critical maxima cutoffs and critical minima cutoffs is returned.

See Also

`get_eb_times`, `segmag`

Examples

```
#see ?segmag for an example
```

---

flag_maxima_positions *Detect local maxima/minima of a numeric vector*

Description

Fast detection of local maxima and minima of a numeric vector. This function takes a numeric vector as input and returns a logical vector with the same length and TRUE values at local maxima/minima (depending on function). If multiple succeeding values at a local maximum/minimum are equal, then only the center (rounded up if necessary) of the maximum/minimum is marked with TRUE.

Usage

```
flag_maxima_positions(values)

flag_minima_positions(values)
```

Arguments

`values`	numeric vector of values

Value

logical vector with TRUE at the center of local maxima/minima

Examples

```
var <- c(1,2,3,3,2,1,4,5,6,7,5,4,3)

## Using the Maxima functions
flag_maxima_positions(var)

# values of maxima
var[flag_maxima_positions(var)]

# indices of maxima
which(flag_maxima_positions(var))


## Using the Minima functions
flag_minima_positions(var)

# values of maxima
var[flag_minima_positions(var)]

# indices of maxima
which(flag_minima_positions(var))
```

---

get_eb_times *Retrieve event boundary times from a segmag object*

Description

Returns the times of event boundaries from a segmag object. Event boundaries are defined as the center of local maxima in segmentation magnitude that are higher than a critical cutoff value. The critical cutoff value should be determined with the `bootstrap_critical_cutoffs` function.

Usage

```
get_eb_times(segmag, cutoff)
```

Arguments

segmag	object of class segmag
cutoff	numeric value determining the critical cutoff in segmentation magnitude

Value

numeric vector with event boundary times

See Also

get_eb_times_segmag_diff, bootstrap_critical_cutoffs

Examples

```
#see ?segmag for an example
```

---

get_eb_times_segmag_diff *Retrieve event boundary times for a difference of segmag objects*

Description

Specific function when calculating event boundaries for the difference of two segmag objects (segmag - segmag_substract). Returns the times of event boundaries that are defined as the center of local maxima (minima) in segmentation magnitude that are higher (lower) than a critical cutoff max/min values. The critical cutoff max/min values should be determined with the bootstrap_critical_cutoffs function.

Usage

```
get_eb_times_segmag_diff(segmag, segmag_substract, cutoff_max = NULL,
  cutoff_min = NULL)
```

Arguments

segmag	object of class segmag
segmag_substract	object of class segmag.
cutoff_max	numeric value determining the critical cutoff for maxima in segmentation magnitude
cutoff_min	numeric value determining the critical cutoff for minima in segmentation magnitude

Value

numeric vector with event boundary times. If cutoff_max or cutoff_min is NULL than the respective event boundaries are omitted.

See Also

get_eb_times, bootstrap_critical_cutoffs

plot.segmag *Plot segmentation magnitude*

Description

Draws a plot depicting the segmentation magnitude resulting from overlaid Gaussians for each key press time across participants. If segmag_substract is defined then the difference in segmentation magnitude of segmag - segmag_substract is drawn.

Usage

```
## S3 method for class 'segmag'
plot(x, cutoffs = NULL, eb_times = NULL,
  segmag_substract = NULL, save_as_png = NULL, ...)
```

Arguments

x	object of class segmag
cutoffs	numeric vector of critical cutoffs that are drawn as horizontal red lines. Use bootstrap_critical_cutoffs in order to determine the cutoffs for a specific segmag object.
eb_times	numeric vector of event boundary times to highlight in the plot
segmag_substract	object of class segmag. If this value is set than the difference in segmentation magnitude of segmag - segmag_substract is drawn.
save_as_png	string, optional name of file where to save plot (.png is added automatically)
...	paramters passed to generic plot function

See Also

segmag

segmag *Create Segmentation Object*

Description

This function creates a segmag object from a vector of participant ids and a vector of times when the participants pressed the segmentation key on the keyboard. All functions in the segmag package work on this object (e.g., plotting results, determining event boundaries). The additional parameters define the size and offset of the Gaussian that is centered around each key press and the time window in the data set to consider (time_min, time_max, time_steps).

Usage

```
segmag(ids, time_keypresses, data = NULL,
  time_min = (min(min(time_keypresses), (min(time_keypresses) + gauss_offset))
  - gauss_cutoff), time_max = (max(max(time_keypresses), (max(time_keypresses)
  + gauss_offset)) + gauss_cutoff), time_steps = 0.01, gauss_offset = 0,
  gauss_sd = 1, gauss_cutoff = 6 * gauss_sd)
```

Arguments

ids	factor assigning a participant id to each key press (same length as time_keypresses)
time_keypresses	numeric vector with the times when the segmentation key was pressed on the keyboard
data	optional data.frame where ids and time_keypresses are stored in
time_min	time window of key press times used in calculations
time_max	time window of key press times used in calculations
time_steps	interval length of time steps within time window
gauss_offset	offset the overlaid Gaussian relative to the key press times in order to account for manual reaction times (0: centered, negative values: assume that event boundary occurred before key press)
gauss_sd	sd of overlaid Gaussian
gauss_cutoff	speed up calculations by not considering values of the overlaid Gaussian that are more than gauss_cutoff units from the center of the Gaussian away (because they are very close to 0)

Details

First, segmentation magnitude for each participant across time is calculated by centering a Gaussian around each key press. If multiple Gaussians overlap across time then only the maximum values is used (not sum) to ensure an equal weight of each participant on the overall segmentation magnitude. Thereafter, the segmentation magnitudes of the participants are summed up to define the overall segmentation magnitude across time. The higher the segmentation magnitude at one point in time the more participants pressed a key around this time point. To account for the fact that participants have a certain temporal error in their key presses, Gaussians are used to expand the influence of a single key press into time. Furthermore, an offset to these Gaussians can be defined in order to account for manual reaction times and to get a better estimate of the "real" time point of an event boundary.

In order to achieve a decent calculation speed, a fixed time scale with interval length time_steps and starting from time_min is used. All key-press times are rounded to their closest interval. A warning is issued if this changes the raw key-press times.

Value

segmag object (also contains a data.frame with segmentation magnitude across time as $data)

See Also

bootstrap_critical_cutoffs, get_eb_times, plot.segmag

Examples

```
# segmentation responses (key presses) of 6 participants watching a movie (30 seconds long)
participant_ids <- factor(c(1,1,1,1,2,2,3,3,3,3,4,4,4,5,5,6,6,6))
time_keypresses <- c(7,12,18,25,12.1,24.9,6.9,10,25.2,29,7.2,12.05,17.5,7.05,25,6.9,16.1,25)

# create segmag object
segmag1 <- segmag(participant_ids, time_keypresses, time_min=0, time_max=30)

## Not run:
# estimate the critical cutoff against an alpha level of 0.05
# Note: This is an estimate an will vary slightly against multiple calls of this function
#       (variation is the lower the higher n_bootstrap is set)
critical_cutoff <- bootstrap_critical_cutoffs(segmag1, 5000, .95)

## End(Not run)


# timestamps of significant event boundaries within the movie
eb_times <- get_eb_times(segmag1, critical_cutoff)

plot(segmag1, critical_cutoff, eb_times)
```

Index