Package ‘scanpath’

February 8, 2014

Type Package

Title Tools for analyzing spatio-temporal patterns in eye movements.

Version 1.05

Date 2014-02-04

Author Titus von der Malsburg

Maintainer Titus von der Malsburg <malsburg@posteo.de>

Description Provides a similarity measure for scanpaths that is sensitive to
spatial and temporal patterns. This measure can be used, for example, to
perform cluster analyses of sets of scanpaths or to evaluate models of oculo-motor control.

License GPL-2

LazyLoad yes

Depends reshape, lattice, gdata

R topics documented:

avg.group.dist *Mean similarities among sets of scanpaths*

Description

Calculates the mean similarities among sub-sets of scanpaths given by a grouping variable. For instance, when the groups are the participants of the experiment, we get a matrix of the average similarities between the scanpaths of the subjects.

Usage

```
avg.group.dist(d, groups)
```

Arguments

`d`	a symmetric matrix of similarities as calculated by the function `scasim` or a dist object (see `dist`).
`groups`	a vector that assigns each column to a group.

Value

a matrix containing the pair-wise average similarities for all groups.

See Also

`scasim`, `dist`

Examples

```
data(eyemovements)
d <- scasim(eyemovements, dur ~ x + y | trial, 512, 384, 60, 1/30)
s <- constant.vars(eyemovements, trial)$subject
avg.group.dist(d, s)
```

constant.vars *Extract variables that are constant within the levels of a grouping variable.*

Description

Given a data frame and a grouping variable, this function returns a new data frame that contains only the variables that are constant within the levels of the grouping variable. The results data frame contains only one line for each level of the grouping variable. This function can be used to extract trial-based information from a data frame containing experimental data with multiple measurements per trial.

Usage

```
constant.vars(data, groups)
```

Arguments

`data`	a data frame.
`groups`	a grouping variable. Either a vector with the same length as the data frame or simply a column in `data`.

Value

A data frame with one entry for each level of the grouping variable.

Examples

```
data(eyemovements)
head(eyemovements)
constant.vars(eyemovements, trial)
```

`eyemovements` *Fixational eye movements during reading a sentence*

Description

This data set consists of nine trials. One of these trials was recorded in an experiment where the participants had to read single sentences on a screen, one at a time. The other eight trials are modifications of the recorded trial. There are three imaginary participants and for each participant there are three trials: in the first trial the participant read the sentence straight from left to right. In the second trial there was a short regression from the last word to the beginning of the sentence. In the third trial the regression was longer and the eyes check several intermediate words before returning to the end of the sentence. In order to simulate different reading speeds, the recorded fixation durations were modified by adding additional time that was sampled from a normal distribution with different means for each subject.

Usage

```
data(eyemovements)
```

Format

In the data frame each row represents one fixation of the eyes. The data frame has the following columns:

`subject` the id of the subject

`trial_id` the id of the trial (unique only within subjects)

`dur` the duration of a fixation in milliseconds

`x` the x coordinate of a fixation in pixels `y` the y coordinate of a fixation in pixels

`trial` the id of the trial (unique in the whole data set)

`word` the word that was targeted by the fixation

`find.fixation` *Mark the beginning of specified pattern in a letter sequence*

Description

This function uses regular expressions to identify elements in a vector of letters where a specific pattern begins. Among other things, this can be used to identify certain eye movement patterns in fixation data.

Usage

```
find.fixation(l, groups, expr, nth = NA, subpattern = 0)
```

Arguments

`l`	a vector of single characters.
`groups`	a grouping variable. The search for matching patterns in performed in each group separately. Matches crossing group boundaries are not considered.
`expr`	a regular expression describing the pattern.
`nth`	specifies which match should be returned if there were several within a group. The default is to return all matches.
`subpattern`	the subpattern of interest. If zero, the beginning of the full pattern will be marked. If n>0, the beginning of the n-th subpattern will be marked. See examples.

Value

A vector giving the indices in l where the matches begin. If subpattern > 0, then the indices will point to the beginning of the subpatterns.

Examples

```
data(eyemovements)
words <- eyemovements$word
l <- replace.all(words, letters[1:length(unique(words))])
trial <- eyemovements$trial
# Find fixations on word 6 (which is represented by letter "f"):
idx <- find.fixation(l, trial, "f")
words[idx]
# Find fixations on word 6 but only the second match within
# a group (i.e., trial):
idx <- find.fixation(l, trial, "f", nth=2)
words[idx]
```

```
# Find fixations on word 6 that are followed by fixations on word 7 ("g"):
idx <- find.fixation(l, trial, "fg")
words[idx]
# Find those fixations on word 6 that followed fixations on word 5 ("e"):
idx <- find.fixation(l, trial, "e(f)", subpattern=1)
words[idx]
# Find the beginning of a second sweep over the sentence:
idx <- find.fixation(l, trial, "a[b-h]+(a+[b-g]+)", subpattern=1)
words[idx]
```

`inverse.gnomonic` *Project points from a plane to a sphere.*

Description

Projects from points given in plane coordinates to lat-lon, i.e., to the surface of a sphere (inverse gnomonic projection). This can be used to project screen coordinates to coordinates of the visual field of somebody sitting in front of the screen.

Usage

```
inverse.gnomonic(x, y, center_x, center_y, distance, unit_size = 1)
```

Arguments

`x`	the x-coordinates of the points.
`y`	the y-coordinates of the points.
`center_x`	is the x-coodrinate of the point that is targeted when the eye looks straight ahead (usually the center of the screen).
`center_y`	is the y-coodrinate of the point that is targeted when the eye looks straight ahead (usually the center of the screen).
`distance`	is the distance of the plane to the center of the sphere.
`unit_size`	is ratio of one unit of the plane coordinate system and the unit in which the viewing distance was given. So, one unit in the coordinate system is `unit_size * unit of distance`.

Value

A data frame containing lat and lon coordinates.

References

See http://mathworld.wolfram.com/GnomonicProjection.html.

Examples

```
data(eyemovements)
x <- eyemovements$x
y <- eyemovements$y

latlon <- inverse.gnomonic(x, y, 268, 382, 8, 1/30)

# Before projection:
plot(x, y)
# After:
with(latlon, plot(lon, lat))
```

`replace.all` *Replace all values in a vector by other values*

Description

Given a vector of values `x` and a set of replacement values `r` (one for each unique value in `x`), this function replaces each value in `x` by the corresponding new value in `r`.

Usage

```
replace.all(x, r, res.type = as.character, na.value = NA, ...)
```

Arguments

`x`	a vector.
`r`	a vector of replacement values. One value for each element in `dot(unique(x))`.
`res.type`	a function for converting the result vector to a desired data type.
`na.value`	a replacement value for NAs.
`...`	extra arguments that are passed to `res.type`.

Value

a version of `x` in which all original values are substituted by the replacements specified in `r`.

Note

This function can be used to replace arbitrary region identifiers in fixation data by single letters. This format is needed by the functions `find.fixation` and `match.scanpath`.

Examples

```
replace.all(1:10, letters[1:10])
replace.all(10:1, letters[1:10])

data(eyemovements)
words <- eyemovements$word

replace.all(words, letters[1:length(unique(words))])
replace.all(words, letters[1:length(unique(words))], paste, collapse="")
```

`scanpath` *Tools for analyzing scanpaths*

Description

This package provides tools for analyzing spatio-temporal patterns in eye movements, a.k.a. scanpaths.

Author(s)

Titus von der Malsburg <`malsburg@posteo.de`>

References

von der Malsburg, T. and Vasishth, S. (2011). What is the scanpath signature of syntactic reanalysis? Journal of Memory and Language, 65(2):109-127.

See Also

`scasim`, `avg.group.dist`, `constant.vars`, `which.centroid`

`scasim` *Calculate pair-wise similarities among a set of scanpaths*

Description

Given a set of scanpaths, this function calculates the similarities between all pairs of scanpaths in the set.

Usage

```
scasim(data, formula, center_x, center_y, viewing_distance, unit_size,
  modulator = 0.83, data2 = NULL, formula2 = formula,
  normalize = "fixations")
```

Arguments

`data`	is a data frame containing the eye movement data. Each line represents one fixation of the eyes. The fixations of a trial have to be listed en bloc and in chronological order. Required columns are: trial ID (unique in the whole data set), x- and y- coordinate of fixations, and the fixation durations. See the example data set provided with this package.
`formula`	specifies which columns in the given data frame are relevant. The left-hand side specifies the column that contains the fixation duration The right-hand side consist of terms that specify x- and y-coordinate of the fixations (in that order) conditioned on trial IDs. See examples below.
`center_x`	is the x-coodrinate of the point that is targeted when the eye looks straight ahead (usually the center of the screen).
`center_y`	is the y-coodrinate of the point that is targeted when the eye looks straight ahead (usually the center of the screen).
`viewing_distance`	is the distance of the eyes to the screen.
`unit_size`	is ratio of one unit of the coordinate system and the unit in which the viewing distance was given. So, one unit in the coordinate system is `unit_size * unit of distance`. Example: If the coordinates are pixels on a screen with 60 dpi and the unit of the distance is inches, `unit_size` has to be set to 1/60.
`modulator`	specifies how spatial distances between fixations are assessed. When set to 0, any spatial divergence of two compared scanpaths is penalized independently of its degree. When set to 1, the scanpaths are compared only with respect to their temporal patterns. The default value approximates the sensitivity to spatial distance found in the human visual system.
`data2`	optionally provides a second set of scanpaths. In this case `scasim` computes the distances from each scanpath in the first set to each scanpath in the second set. The resulting matrix of distances is therefore not necessarily square.
`formula2`	specifies the relevant columns in `data2` in case they are named differently in `data2` than in `data`. `formula2` defaults to `formula`.
`normalize`	specifies how each calculated similarity value for scanpaths s and t should be normalized. There are three possible modes: `fixations` normalize by dividing by the total number of fixations in s and t, `durations` divide by the total duration of s and t, `FALSE` don't normalize at all. The choice of normalization mode can have a strong effect when fitting maps of scanpaths using `isoMDS` or `cmdscale`. Not normalizing at all can yield maps that are difficult to interpret when scanpaths differ markedly in their duration or number of fixations.

Details

When using `scasim`, the dissimilarity of two scanpaths is assessed based on the distances of their fixation targets. If the distances are really small, the dissimilarity will mostly be a function of just the differences of the fixation durations. The evaluation of distance accounts for the particular way the human visual system magnifies the center of the visual field and the drop in acuity at its periphery.

Value

A matrix containing the pair-wise dissimilarities. Columns and rows correspond to trials. Their order is the same as the order of trials in `data`.

Author(s)

Titus von der Malsburg <`malsburg@posteo.de`>

References

von der Malsburg, T. and Vasishth, S. (2011). What is the scanpath signature of syntactic reanalysis? Journal of Memory and Language, 65(2):109-127.

von der Malsburg, T., Vasishth, S., and Kliegl, R. (2012). Scanpaths in reading are informative about sentence processing. In Michael Carl, P. B. and Choudhary, K. K., editors, Proceedings of the First Workshop on Eye-tracking and Natural Language Processing, pages 37-53, Mumbai, India. The COLING 2012 organizing committee.

See Also

`isoMDS` can be applied to the output of `scasim` in order to fitting maps of scanpaths.

Examples

```
data(eyemovements)

# Calculating dissimilarities: when looking straight ahead the gaze
# targets the point with the coordinates (512,384), viewing
# distance (eye to screen) is 60 cm, 1 unit in the data (pixel) is
# 1/30 cm.

dissimilarities <- scasim(eyemovements, dur ~ x + y | trial,
                          512, 384, 60, 1/30)

# Using cmdscale for fitting a map:

map <- cmdscale(dissimilarities)
plot(map, cex=4)
text(map, labels(rownames(map)))
```

`which.centroid` *Find item with minimal average distance to other items*

Description

Given a matrix of similarities, this function identifies the item that has the smallest average distance to the other items.

Usage

```
which.centroid(d, select = NULL)
```

Arguments

`d`	a symmetric matrix of similarities or a `dist` object.
`select`	a vector of names of items that should be included in the analysis. These names should correspond to the row and column names of `d`. Items not listed are ignored completely. The default is to take all items into account.

Details

Although the function is called which.centroid, the selected item is not necessarily the centroid in the strict sense. However, it is the item which is closer to the centroid that any other item in the set.

Value

the name of the item that has the minimal average distance to the other items.

Examples

```
data(eyemovements)
d <- scasim(eyemovements, dur ~ x + y | trial, 512, 384, 60, 1/30)
which.centroid(d)
which.centroid(d, c("1:1", "1:2", "1:3"))
```

Index