Package: rMEA (via r-universe)

October 18, 2024

Type Package

Title Synchrony in Motion Energy Analysis (MEA) Time-Series

Version 1.2.3

Date 2022-02-17

Author Johann R. Kleinbub, Fabian Ramseyer

Maintainer Johann R. Kleinbub <johann.kleinbub@gmail.com>

Description A suite of tools useful to read, visualize and export bivariate motion energy time-series. Lagged synchrony between subjects can be analyzed through windowed cross-correlation. Surrogate data generation allows an estimation of pseudosynchrony that helps to estimate the effect size of the observed synchronization. Kleinbub, J. R., & Ramseyer, F. T. (2020). rMEA: An R package to assess nonverbal synchronization in motion energy analysis time-series. Psychotherapy research, 1-14. <doi:10.1080/10503307.2020.1844334>.

URL https://github.com/kleinbub/rMEA https://psync.ch

BugReports https://github.com/kleinbub/rMEA/issues

Imports grDevices, graphics, methods, stats, utils

Depends R (>= 4.0.0)

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.1

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

RemoteUrl https://github.com/kleinbub/rmea

RemoteRef HEAD

RemoteSha af9a64c56d18532f8f81e365900943def784d359

Contents

`CCFartefacts` *Detection of potential artefacts in CCF results.*

Description

High synchronization values for extended time durations may be the effect of artefacts in the MEA data. For instance subject 2 movement may have been captured by subject 1's camera (or ROI) as well, or some environment characteristic (e.g. light) is changing for both cameras (or ROIs). This function identifies those moments allowing to inspect the original videos with temporal precision. **Please note that is impossible to discriminate real high synchronization phenomena from artefacts without inspecting the original videos.**

Usage

```
CCFartefacts(mea, threshold, delta = 1, duration = attributes(mea)$ccf$inc)
```

Arguments

mea	an object of class MEA or a list of MEA objects (see function readMEA)
threshold	A single numeric value specifying the absolute correlation value above (and below) which a window must be considered artefactual.
delta	Integer. The maximum numbers of consecutive CCF windows below threshold to be allowed in an artefactual streak without determining its end. A value of 1 is default and recommended, but it can be increased if the reports become too long (e.g. with very noisy source material, or when working with very small ccf windows), to achieve the desired level of reporting detail.
duration	Integer. Minimum duration of the artefacts to be reported. Note that artefacts smaller than the CCF windows increments cannot be detected.

Details

The function only considers lag_zero correlations as MEA artefacts are expected to be non-lagged phenomena.

Value

a data.frame object with all potential artefact epochs

Examples

```
## read the first 4 minutes of the normal sample
## (intake interviews of patients that carried on therapy)
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist",
                      idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)
mea_normal <- MEAccf(mea_normal, lagSec = 5, winSec = 30, incSec = 10, ABS = FALSE)

## find potential artefacts with various granularity of reporting
print(CCFartefacts(mea_normal, threshold = 0.2, delta=1))
print(CCFartefacts(mea_normal, threshold = 0.2, delta=5))
print(CCFartefacts(mea_normal, threshold = 0.2, delta=5, duration=60))
```

---

ccfResNames *Extract the names of the ccf analysis summaries in a MEA objects*

Description

Extract the names of the ccf analysis summaries in a MEA objects

Usage

```
ccfResNames(mea)
```

Arguments

mea	an object of class MEA or a list of MEA objects (see function readMEA)

Value

a vector containing the labels of the ccfRes indexes

---

cohens_d *Cohen's d A simple function to calculate Cohen's d effect size*

Description

Cohen's d A simple function to calculate Cohen's d effect size

Usage

```
cohens_d(x, y)
```

Arguments

x, y	two numeric vectors containing the scores of the two samples

Examples

```
# Generates two samples with means distance of 1 sd
x = rnorm(1000, mean = 10, sd = 1.5)
y = rnorm(1000, mean = 11.5, sd = 1.5)
# cohen's d should approximate to 1
cohens_d(x,y)
```

---

colTrans *Transform color*

Description

Transform color

Usage

```
colTrans(col, luminosity = NA, alpha = NA)
```

Arguments

col	a color to begin with in hex format
luminosity	numeric. negative numbers darken the color, positive lighten it. Eg: a value of -2 make the color two times darker.
alpha	numeric from 0 to 1. the value of opacity of the resulting color

Value

a color string

`diagnosticPlot` *Plots the initial, middle and ending part of a MEA object*

Description

This is typically useful to check if the motion energy time-series are good. The middle section is chosen randomly among possible middle sections.

Usage

```
diagnosticPlot(mea, width = 60, ...)
```

Arguments

`mea`	an object of class `MEA` (see function `readMEA`).
`width`	integer. The number of seconds to be plotted for each panel
`...`	further arguments passed to `plot`

Details

Motion energy time-series should always be visually inspected for possible artifacts. Periodic peaks or drops in time-series are indicators of e.g. key-frames or duplicated video-frames. For further information regarding the program MEA, please refer to the documentation available at `http://www.psync.ch`.

Examples

```
## read a single file
path_normal <- system.file("extdata/normal/200_01.txt", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                    s1Name = "Patient", s2Name = "Therapist", skip=1,
                    idOrder = c("id","session"), idSep="_")
## Visual inspection of the data
diagnosticPlot(mea_normal[[1]])
```

getCCF *Extract ccf values from MEA objects*

Description

Extract ccf values from MEA objects

Usage

```
getCCF(mea, type)
```

Arguments

mea	an object of class MEA or a list of MEA objects (see function readMEA)
type	A character vector defining which ccf must be extracted. Either "matrix", "fullMatrix", one of the ccfRes indexes identified with ccfResNames, or the name of one lag value which can be identified with lagNames

Value

If type="matrix", the ccf matrix with discrete lag-seconds is returned. If type="fullMatrix", the whole ccf matrix with all lags is returned. Otherwise a vector containing the ccf time-series for the selected lag, or aggregated values is returned. If mea is a list, the return value is a list of the individual ccf of each MEA object.

id *Get MEA attributes*

Description

Get MEA attributes

Usage

```
id(mea)

group(mea)

session(mea)

sampRate(mea)

s1Name(mea)

s2Name(mea)

uid(mea)
```

Arguments

mea	an object of class MEA or a list of MEA objects (see function readMEA)

Details

if a well formatted list of MEA objects is provided, the function returns a vector of results for id, session, group and uid. sampRate, s1Name, and s2Name return always a single value, as they are not allowed to be mixed.

Value

A string or a vector of strings containing the metadata.

---

lagNames *Extract the lag names of a ccf analysis in MEA objects*

Description

Extract the lag names of a ccf analysis in MEA objects

Usage

```
lagNames(mea)
```

Arguments

mea	an object of class MEA or a list of MEA objects (see function readMEA)

Value

a vector containing the labels of the lag values

---

lines.MEA *Adds lines of a* MEA *object to a Plot*

Description

Adds lines of a MEA object to a Plot

Usage

```
## S3 method for class 'MEA'
lines(x, from = 0, to = NULL, duration = NULL, ccf = F, rescale = F, ...)
```

Arguments

x	an object of class MEA (see function readMEA).
from	either an integer or a string in the format hh:mm:ss or mm:ss representing the starting second.
to	if duration is not specified, either an integer or a string in the format hh:mm:ss or mm:ss representing the ending second.
duration	if to is not specified, either an integer or a string in the format hh:mm:ss or mm:ss representing the amount of seconds to be plotted.
ccf	either FALSE or a string representing the type of ccf to be overlayed. Possible values can be found with the ccfResNames function.
rescale	logical. Should the motion energy time-series be rescaled?
...	further arguments passed to lines

Details

Note: if more of than 10s of trailing zeroes are found at the end of both s1 and s2 signals they are truncated.

Examples

```
## read a single file
path_normal <- system.file("extdata/normal/200_01.txt", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist", skip=1,
                      idOrder = c("id","session"), idSep="_")
mea_normal <- MEAccf(mea_normal, lagSec = 5, winSec = 30, incSec = 10, ABS = FALSE)
mea_smoothed <- MEAsmooth(mea_normal)
## Visual inspection of the data
plot(mea_normal[[1]], from = 240, duration=20)
lines(mea_smoothed[[1]], from = 240, duration=20, lty=3, col=c(1,2))
```

MEA MEA *class constructor*

Description

The preferred way to create an object of class MEA is through the function readMEA.

Usage

```
MEA(
  dataframe,
  sampRate,
  filter = "raw",
  id,
  session,
```

```
  group,
  s1Name,
  s2Name,
  uid = paste(group, id, session, sep = "_")
)

is.MEA(x)
```

Arguments

dataframe	a data frame with 2 columns containing MEA data respectively for subject 1 (s1) and subject 2 (s2).
sampRate	integer. The sampling rate of the MEA data. Normally derived from the framerate of the analyzed video sequence (frames per second; fps).
filter	a string describing the pre-processing that has been applied on the raw data.
id	a string representing a unique identifier of the dyad that the MEA data belong to.
session	an integer representing the session (or experiment, interaction, etc); if each dyad is measured only once, the default value is 1.
group	a string naming the group the dyad belongs to, such as diagnostic group, clinic, etc.
s1Name	a string naming subject 1.
s2Name	a string naming subject 2.
uid	a string providing a unique identifier of the file. By default 'group_id_session'.
x	object to be tested.

Details

It is advised to **not** create the MEA object manually but to always use the function readMEA instead.

Value

A list containing three objects:

MEA: the dataframe containing the motion energy data

ccf: the matrix of lagged cross-correlations between s1 and s2 (if MEAccf was run)

ccfRes: some useful row marginals

is.MEA returns TRUE if and only if its argument is of class MEA

MEAccf *Moving-windows lagged cross-correlation routine for* MEA *objects*

Description

This function analyzes a bivariate MEA signal represented by two time-series (subject 1 "s1", subject 2 "s2") resulting from a dyadic interaction. MEAccf performs windowed cross-correlations with specified increments. The cross-correlation analysis is repeated for each lag step, with discrete increments of 1 sample in both directions.

Usage

```
MEAccf(mea, lagSec, winSec, incSec, r2Z = T, ABS = T)
```

Arguments

mea	an object of class MEA or a list of MEA objects (see function readMEA)
lagSec	an integer specifying the maximum number of lags (in seconds) for which the time-series will be shifted forwards and backwards.
winSec	an integer specifying the cross-correlation window size (in seconds).
incSec	an integer specifying the step size (in seconds) between successive windows. Values lower than winSec result in overlapping windows.
r2Z	logical. The default value TRUE applies Fisher's r to Z transformation (inverse hyperbolic tangent function) to all computed correlations.
ABS	logical. The default value TRUE transforms the (Fisher's Z-transformed) correlations to absolute values.

Details

The choice of lagSec depends on the type of synchronization expected from the specific interaction. In the literature, lags of ±5 seconds have been reported by multiple authors. Function MEAlagplot can be used for visual inspection of the appropriateness of the chosen lag.

The choice of winSec represents the temporal resolution of the analysis. The combination of incSec and winSec settings has a big impact on the results. These parameters should be chosen carefully, guided by theoretical and empirical considerations.

If r2Z is TRUE, values of Fisher's Z are constrained to an upper bound of 10.

Using absolute values (ABS) treats positive and negative cross-correlations as equal. The underlying assumption is that both simultaneous movement (positive correlation) and when one subject accelerates and the other decelerates (negative correlation), are both signs of interrelatedness and should thus contribute equally to overall synchrony.

Value

The function returns a copy of the `mea` object in which the `ccf` and `ccfRes` objects are populated. `mea$ccf` includes the complete lagged cross-correlation table for each window and each lag of S1 and S2 MEA signals. `mea$ccfRes` contains various aggregate values, typically used in research:

- lag_zero: a numeric vector containing for each window, the non-lagged cross-correlation value.
- all_lags: a numeric vector containing for each window, the average across all lags.
- s1_lead/s2_lead: a numeric vector containing for each window, the average of positive/negative lags, summing up the strength of S1/S2 in "leading" the synchronization.
- s1_lead_0/s2_lead_0: the same as s1_lead/s2_lead, but including lag_zero values in the average.
- bestLag: for each window, the lag value (in seconds) that has the highest correlation value.
- grandAver: a single numeric value of the grand-average of the whole cross-correlation table.
- winTimes: a data frame containing the start and end times of each window in the format hh:mm:ss

Examples

```
## read a single file
path_normal <- system.file("extdata/normal/200_01.txt", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist", skip=1,
                      idOrder = c("id","session"), idSep="_")

## perform ccf analysis
mea_ccf = MEAccf(mea_normal, lagSec = 5, winSec = 60, incSec = 30, r2Z = TRUE, ABS = TRUE)
summary(mea_ccf)

##extract ccf values
res <- getCCF(mea_ccf, type="grandAver")
print(res)

#visualize the analysis results for the first file
MEAheatmap(mea_ccf[[1]])
```

`MEAdistplot` *Distribution of cross-correlations*

Description

Plots the distribution of the average cross-correlations in a list of `MEA` objects.

Usage

```
MEAdistplot(
  mea,
  contrast = FALSE,
  by = c("none", "group", "id", "session"),
  by.group = FALSE,
  sub.line = 0.5,
  ...
)
```

Arguments

mea	a well formatted list of MEA objects (see function MEAlist).
contrast	either FALSE or a list of MEA objects to be used as a contrast
by	Either "none", "group", "id", or "session". Defines the grouping to be used.
by.group	deprecated argument. Use by="group" instead.
sub.line	on which margin line should the Effect Size subtitle be printed, starting at 0 counting outwards.
...	further graphical parameters passed to plot

Details

If contrast is defined, then a normalized difference (Cohen's *d*) between the means of each group and the contrast is provided. Otherwise, if the mea object has 3 or less groups, Cohen's *d* will be calculated on the group differences.

Examples

```
## This example is excluded from test as it may take more than 10s to run
## read the first 4 minutes of the normal sample
##   (intake interviews of patients that carried on therapy)
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist",
                      idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)
mea_normal <- setGroup(mea_normal, "normal")

## read the dropout sample (intake interviews of patients that dropped out)
path_dropout <- system.file("extdata/dropout", package = "rMEA")
mea_dropout <- readMEA(path_dropout, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist",
                      idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)
mea_dropout <- setGroup(mea_dropout, "dropout")

## Combine into a single object
mea_all = c(mea_normal, mea_dropout)

## Create a shuffled sample
mea_rand = shuffle(mea_all, 20)
```

```
## Compute ccf
mea_all = MEAccf(mea_all, lagSec = 5, winSec = 60, incSec = 30, r2Z = TRUE, ABS = TRUE)
mea_rand = MEAccf(mea_rand, lagSec = 5, winSec = 60, incSec = 30, r2Z = TRUE, ABS = TRUE)

## Visualize the effects:

MEAdistplot(mea_all, contrast = mea_rand, by.group = TRUE)
```

MEAheatmap *Plot a heatmap of dyadic cross-correlations*

Description

Graphical representation of the lagged cross-correlations in time. Provides an intuitive description of synchronization dynamics.

Usage

```
MEAheatmap(
  mea,
  legendSteps = 10,
  rescale = FALSE,
  colors = c("#F5FBFF", "#86E89E", "#FFF83F", "#E8A022", "#FF3700"),
  bias = 1,
  mirror = TRUE
)
```

Arguments

mea	an object of class MEA (see function readMEA).
legendSteps	integer. the number of levels used for the color-coding of the legend.
rescale	logical. If TRUE, the color range will represent the minimum and maximum of the data. Otherwise the theoretical correlation range -1 to 1.
colors	a vector of colors defining the plot scale.
bias	a positive number. Allows to skew the color scale. Values larger than 1 give more widely spaced colors at the high end, and vice versa.
mirror	logical. If TRUE, colors are mirrored for negative correlation values. This has effect only if MEAccf was run with ABS=FALSE

Details

The cross-correlation values are rescaled to be in a range from 0 to 1 before plotting.

`MEAlagplot` *Plots the average cross-correlation at different lags*

Description

Provides a graphical representation of the comparison between two lists of `MEA` objects. The X-axis represents the lag values over which cross-correlation was calculated (in seconds), the Y-axis represents the averaged strength of the cross-correlation. Typically, the is useful for a visual inspection of the strength of synchrony from real dyads in relation to synchrony expected by coincidence (pseudosynchrony).

Usage

```
MEAlagplot(
  mea,
  contrast = F,
  by.group = T,
  sub.line = 0.5,
  mea.lines = TRUE,
  mea.alpha = 0.8,
  contrast.lines = TRUE,
  contrast.alpha = 0.5,
  ...
)
```

Arguments

`mea`	a list of `MEA` objects (see function `MEAlist`).
`contrast`	either FALSE or a list of `MEA` objects to be used as a contrast
`by.group`	logical. Should the different groups of `mea` be plotted separately?
`sub.line`	on which margin line should the 'social presence' subtitle be printed, starting at 0 counting outwards.
`mea.alpha`	numeric from 0 to 1. The value of opacity of individual lines for the main MEA data. If set to zero, drawing is suppressed to improve performance.
`contrast.alpha`	numeric from 0 to 1. The value of opacity of individual lines for contrast data. If set to zero, drawing is suppressed to improve performance.
`...`	further arguments and `par` parameters passed to `plot`

Details

A typical application of `MEAlagplot` is to represent the difference between real dyads and random dyads obtained through a `shuffle` procedure. It may also be used to see the difference among various filtering procedures or different regions of interest (e.g. head-synchrony versus body-synchrony, female vs. male dyads, etc).

Percentages indicate the relative amount of synchrony where the values are higher than the contrast sample.

Examples

```
## This example is excluded from test as it takes more than 10s to run
## read the first 4 minutes of the normal sample
##   (intake interviews of patients that carried on therapy)
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist",
                      idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)
mea_normal <- setGroup(mea_normal, "normal")

## read the dropout sample (intake interviews of patients that dropped out)
path_dropout <- system.file("extdata/dropout", package = "rMEA")
mea_dropout <- readMEA(path_dropout, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist",
                      idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)
mea_dropout <- setGroup(mea_dropout, "dropout")

## Combine into a single object
mea_all = c(mea_normal, mea_dropout)

## Create a shuffled sample
mea_rand = shuffle(mea_all, 20)

## Compute ccf
mea_all = MEAccf(mea_all, lagSec = 5, winSec = 60, incSec = 30, r2Z = TRUE, ABS = TRUE)
mea_rand = MEAccf(mea_rand, lagSec = 5, winSec = 60, incSec = 30, r2Z = TRUE, ABS = TRUE)

## Visualize the effects:

MEAlagplot(mea_all, contrast = mea_rand, by.group = TRUE)
MEAlagplot(mea_all, contrast = mea_rand, by.group = FALSE, col=c(2,4))
```

`MEAlist` *Well formatted list of* `MEA` *objects*

Description

This constructor function checks if all the supplied `MEA` objects share the same sampling rate, preprocessing, and metadata, and returns an object with additional attributes summarizing the contained `MEA` objects.

Usage

```
MEAlist(listOfMea)

is.MEAlist(x)
```

Arguments

listOfMea	a list containing MEA objects
x	object to be tested.

Value

an object of class MEAlist

is.MEAlist returns TRUE if and only if its argument is of class MEAlist

---

MEAmap *Apply a function to a single or a list of MEA objects*

Description

MEApply is a wrapper to do.call that allows to apply a function on the motion energy data of one or multiple MEA objects. Complex constructions are possible, see details.

Usage

```
MEAmap(mea, FUN, label = "", ...)
```

Arguments

mea	an object of class MEA or a list of MEA objects (see function readMEA)
FUN	function to apply, found via match.fun.
label	a character vector to update the 'filter' attribute of mea.
...	further arguments passed to FUN. If a function is provided, it will be run on each MEA object and then passed as an argument to FUN.

Details

FUN will be applied on the motion energy time-series of MEA objects, which is stored as a data frame with 2 columns, respectively for s1 and s2.

Value

an object of the same class of the provided 'mea' object, with the transformed motion energy data

MEAoutlier *Replace outliers with given values*

Description

Sometimes motion energy analysis generates excessively high peaks resulting from video artifacts or other anomalies in the video source.

Usage

```
MEAoutlier(
  mea,
  threshold = function(x) {     stats::sd(x) * 10 },
  direction = c("greater", "less"),
  replace = NA
)
```

Arguments

mea	an object of class MEA or a list of MEA objects (see function readMEA)
threshold	a numeric value, or a function returning the threshold value to consider data as outliers.
direction	a text string. One of "greater" or "less": can be abbreviated.
replace	a numeric, NULL, or NA value to use as substitution.

Value

The same mea object with all extreme values substituted.

Examples

```
## read the first 4 minutes of the normal sample
##   (intake interviews of patients that carried on therapy)
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_raw <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                    s1Name = "Patient", s2Name = "Therapist",
                    idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)

## Remove extreme values, higher than 10 times the standard deviation
mea_clean = MEAoutlier(mea_raw, threshold=function(x){sd(x)*10}, direction = "greater")
```

MEAreplace *Substitute values from MEA data*

Description

This function allows to substitute MEA data from a list of time epochs. This is useful to mark and remove artefacts, or to substitute extreme values.

Usage

```
MEAreplace(mea, epochs, replacement, filterLabel = "replaced")
```

Arguments

mea	an object of class MEA or a list of MEA objects (see function readMEA)
epochs	a data.frame containing a list of epochs that must be changed (see Details)
replacement	the value used to mark artefacts. Use 'NA' to remove artefacts and '0' to apply thresholds. Other values are allowed but should not be used without a good reason.
filterLabel	can be used to update the filter attribute, to keep track of the data transformations.

Details

the artefacts data.frame must contain a "start" and "end" columns, with the boundaries of the epochs that must be marked as artefacts. The start and end values can be either integers (denoting seconds), or string values in the format hh:mm:ss, or mm:ss. Furthermore the data.frame must contain a uid column containing strings in the format "group_id_session", OR three columns group, id, session presenting the information separately. Data identifiers must match those of the mea object.

The data.frame can be either hand crafted, for instance by importing a csv file (see read.table), or generated through the packages own artefact detection tools such as CCFartefacts

Value

returns the same MEA or MEAlist object, with all artefactual data substituted.

Examples

```
#' ## read the first 4 minutes of the normal sample
## (intake interviews of patients that carried on therapy)
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist",
                      idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)
mea_normal <- MEAccf(mea_normal, lagSec = 5, winSec = 30, incSec = 10, ABS = FALSE)
```

```
## find potential artefacts
artefacts = CCFartefacts(mea_normal, threshold = 0.2, delta=1)

##replace values
mea_replaced <- MEAreplace(mea_normal, epochs = artefacts,
                           replace = NA, filterLabel = "artefacts deletion")

#visualize results on first case
plot(mea_normal$all_200_1)
plot(mea_replaced$all_200_1)
```

`MEAscale` *Scaling (and centering) of motion energy time-series*

Description

Scaling (and centering) of motion energy time-series

Usage

```
MEAscale(mea, scale = "sd", ..., center = F, removeNA = T)
```

Arguments

`mea`	an object of class `MEA` or a list of `MEA` objects (see function `readMEA`)
`scale`	either a numeric value or a function to be applied to each motion energy time-series to calculate a scaling factor. Default is standard deviation.
`...`	further arguments passed to `scale` if it is a function.
`center`	either a logical value or a numeric vector of length 2 specifying separate centering values for s1 and s2.
`removeNA`	logical. If `scale` is a function, defines whether NAs be removed prior to calculating the scaling factor.

Details

If `scale` is a function, it is found by a call to `match.fun` and typically is either a function or a symbol (e.g., a backquoted name) or a character string specifying a function to be searched for from the environment of the call to apply. Note that the chosen function must return a single numeric value.

`center` is directly passed to `scale`. If `center` is `TRUE` then centering is done by subtracting the means (omitting NAs) from the motion energy time-series. If `center` is a numeric vector, the first value will be subtracted from s1 and the second from s2. Note: the s1 and s2 signals are scaled independently.

Value

returns the same MEA or MEAlist object, with all motion energy data rescaled

Examples

```
## read the first 4 minutes of the normal sample
##   (intake interviews of patients that carried on therapy)
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_raw <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                    s1Name = "Patient", s2Name = "Therapist",
                    idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)

## rescale by factor 0.7
mea_scaled = MEAscale(mea_raw, scale = 0.7)

## rescale with standard deviation
mea_scaled = MEAscale(mea_raw, scale = "sd", removeNA = TRUE)

## assign groups names
mea_raw <- setGroup(mea_raw, "raw")
mea_scaled <- setGroup(mea_scaled, "scaled")

## Compute ccf
mea_raw <- MEAccf(mea_raw, lagSec = 5, winSec = 60, incSec = 30, r2Z = TRUE, ABS = FALSE)
mea_scaled <- MEAccf(mea_scaled, lagSec = 5, winSec = 60, incSec = 30, r2Z = TRUE, ABS = FALSE)

## Compare the effect of scaling on ccf
MEAlagplot(mea_scaled, contrast = mea_raw)
```

MEAsmooth *Moving average smoothing for motion energy data*

Description

This function applies a moving average filter, based on SAS "proc expand" procedure. The moving average is applied independently on each subject's motion energy data. NA values are set to 0.

Usage

```
MEAsmooth(mea, moving.average.win = 0.5)
```

Arguments

mea	an object of class MEA or a list of MEA objects (see function readMEA)
moving.average.win	numeric. The size of the filter window, in seconds or fractions of seconds.

Value

The filtered object(s)

Examples

```
## read the first 4 minutes of the normal sample
##   (intake interviews of patients that carried on therapy)
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_raw <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                    s1Name = "Patient", s2Name = "Therapist",
                    idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)

## filter with moving average
mea_filter = MEAsmooth(mea_raw)

## assign groups names
mea_raw <- setGroup(mea_raw, "raw")
mea_filter <- setGroup(mea_filter, "filtered")

## Compute ccf
mea_raw <- MEAccf(mea_raw, lagSec = 5, winSec = 60, incSec = 30, r2Z = TRUE, ABS = FALSE)
mea_filter <- MEAccf(mea_filter, lagSec = 5, winSec = 60, incSec = 30, r2Z = TRUE, ABS = FALSE)

## Compare the effect of filtering on ccf
MEAlagplot(mea_filter, contrast = mea_raw)
```

---

plot.MEA *Plots an object of class* MEA

Description

Plots an object of class MEA

Usage

```
## S3 method for class 'MEA'
plot(x, from = 0, to = NULL, duration = NULL, ccf = F, rescale = F, ...)
```

Arguments

x	an object of class MEA (see function readMEA).
from	either an integer or a string in the format hh:mm:ss or mm:ss representing the starting second.
to	if duration is not specified, either an integer or a string in the format hh:mm:ss or mm:ss representing the ending second.
duration	if to is not specified, either an integer or a string in the format hh:mm:ss or mm:ss representing the amount of seconds to be plotted.

ccf	either FALSE or a string representing the type of ccf to be overlayed. One of "all_lags" "s1_lead" "s2_lead" "lag_zero" "s1_lead_0" "s2_lead_0" "bestLag" "grandAver" "winTimes".
rescale	logical. Should the motion energy time-series be rescaled?
...	further arguments passed to plot

Details

Note: if more of than 10s of trailing zeroes are found at the end of both s1 and s2 signals they are truncated.

Examples

```
## read a single file
path_normal <- system.file("extdata/normal/200_01.txt", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist", skip=1,
                      idOrder = c("id","session"), idSep="_")
mea_normal <- MEAccf(mea_normal, lagSec = 5, winSec = 30, incSec = 10, ABS = FALSE)
## Visual inspection of the data
plot(mea_normal[[1]], from = 60, to = "2:00")
plot(mea_normal[[1]], from = 0, duration = "5:00")

#' ## Visualize CCF inspection of the data
plot(mea_normal[[1]], from = 0, duration = "2:00", ccf = "lag_zero", rescale=TRUE)
```

---

readMEA *Import MEA raw data*

Description

readMEA reads the output of MEA software.

Usage

```
readMEA(
  path,
  s1Col,
  s2Col,
  sampRate,
  namefilt = NA,
  s1Name = "s1",
  s2Name = "s2",
  idOrder = c("id", "session", "group"),
  idSep = "_",
  removeShortFiles = NULL,
  ...
)
```

Arguments

path	a character vector of full path names; may point to an individual file or a directory containing MEA files. Only .txt or .csv file extensions are considered in directories.
s1Col, s2Col	the index of one or multiple columns in the data, identifying the two dyad's members (e.g. patient and therapist) motion energy data. If multiple columns are selected for a subject (e.g. because of multiple regions of interest per subject), their MEA values will be summed.
sampRate	sampling rate at which the data is acquired (usually frames per second of the original video recording).
namefilt	either NA or a character string specifying a pattern to be matched in the filenames. Regular expressions can be used.
s1Name, s2Name	the label describing each participant. (e.g. Right/Left, or Patient/Therapist, etc).
idOrder	either NA or a character vector that contains one or more of the three strings: "id", "session","group" in a given order. These are used to interpret the filenames and correctly label the cases. The strings can be abbreviated. If the filenames contains other data the character "x" can be used to skip a position. If NA, no attempt to identify cases will be done.
idSep	character vector (or object which can be coerced to such) containing regular expression(s). If idOrder is not NA, this will be used as separator to split the filenames and identify "id", "session", and "group" informations.
removeShortFiles	Either NULL or an number ranging from 0 to 1. Specifies the proportion of the average file length below which a file should be excluded. (E.g. a value of 0.7 will exclude all files with a duration smaller than 70% of the mean duration of all other files in the directory.)
...	Additional arguments passed to read.table. E.g. sep, skip, header, etc.

Details

For instance if s1Col = c(1,3) and s2Col = c(2,4), the returned values will be the sum of column 1 and 3 for the first participant and columns 2 and 4 for the second one.

Value

an object of class MEAlist

Examples

```
## read the first sample (intake interviews of patients that carried on therapy)
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                    s1Name = "Patient", s2Name = "Therapist", skip=1,
                    idOrder = c("id","session"), idSep="_")
mea_normal <- setGroup(mea_normal, "normal")

## read the second sample (intake interviews of patients that dropped out)
```

```
path_dropout <- system.file("extdata/dropout", package = "rMEA")
mea_dropout <- readMEA(path_dropout, sampRate = 25, s1Col = 1, s2Col = 2,
                    s1Name = "Patient", s2Name = "Therapist", skip=1,
                    idOrder = c("id","session"), idSep="_")
mea_dropout <- setGroup(mea_dropout, "dropout")

## Combine into a single object
mea_all = c(mea_normal, mea_dropout)

summary(mea_all)
```

`setGroup` *Sets the group of MEA objects*

Description

Sets the group of MEA objects

Usage

```
setGroup(mea, group)
```

Arguments

`mea`	a single or a list of `MEA` objects (see function `readMEA`)
`group`	a text string specifying a group name

Value

an object of the same type of `'mea'`, with the group attributes set to `group`.

Examples

```
## read a sample
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                    s1Name = "Patient", s2Name = "Therapist",
                    idOrder = c("id","session"), idSep = "_",  skip = 1)
mea_normal <- setGroup(mea_normal, "normal")
```

shuffle *Shuffle MEA data (between subjects)*

Description

This function recombines the s1 and s2 motion energy time-series between all MEA objects in the supplied list. It is typically used to compare genuine synchrony of real data with pseudosynchrony of shuffled (recombined) data.

Usage

```
shuffle(mea, size = "max", keepRoles = FALSE)
```

Arguments

mea	a list of MEA objects (see function readMEA).
size	either "max" or an integer specifying the number of combinations to be returned.
keepRoles	Boolean. If TRUE the resulting random dyad will preserve the roles, i.e. they will all have a new s1 sampled among all s1s and a new s2 sampled among all s2s. If FALSE (default), the role will be disregared.

Details

The shuffling process first creates all possible combinations between s1 and s2 of all MEA objects in the supplied list, then removes the original pairings, and finally extracts the desired numbers of dyads without replacement.

Note: all the ccf data, if present, are discarded from the shuffled objects and have to be calculated again using MEAccf

Value

an object of class MEAlist containing randomly combined dyads.

Examples

```
## read the first 4 minutes of the normal sample
##   (intake interviews of patients that carried on therapy)
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist",
                      idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)
mea_normal <- setGroup(mea_normal, "normal")

## Create a shuffled sample
mea_rand = shuffle(mea_normal, 50)

summary(mea_rand)
```

`shuffle_segments` *Shuffle MEA data (within subjects)*

Description

This function generates fakes dyads to be used for pseudosynchrony calculations following the Ramseyer & Tschacher (2010) within-subject segment shuffling approach. Between subjects shuffling `shuffle` is probably more conservative, and suggested for most cases. This function is provided for replicability of older studies, and can be useful to quickly assess pseudosynchrony in single sessions, or very small samples.

Usage

```
shuffle_segments(mea, n_each, segSec)
```

Arguments

`mea`	a list of `MEA` objects (see function `readMEA`).
`n_each`	the number of random dyads to be generated from each real dyad.
`segSec`	the width (in seconds) of the shuffling segments.

Details

For each `MEA` object, the shuffling procedure first divides s1 and s2 MEA data in segments of size `segSec`, then shuffles them within subject (so that the new segments of s1, are the old segments of s1 in a new order). This is repeated for `n_each` times, before getting to the next `MEA` object

Note: all the ccf data, if present, are discarded from the shuffled objects and have to be calculated again using `MEAccf`

Value

an object of class `MEAlist` containing `n_each * length(mea)` random dyads.

Examples

```
## read the first 4 minutes of the normal sample
##   (intake interviews of patients that carried on therapy)
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_normal <- readMEA(path_normal, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist",
                      idOrder = c("id","session"), idSep="_", skip=1, nrow = 6000)
mea_normal <- setGroup(mea_normal, "normal")

## Create a shuffled sample
mea_rand = shuffle_segments(mea_normal, n_each=10, segSec=30)

summary(mea_rand)
```

timeMaster *Transform time values between different formats*

Description

This function allows to

Usage

```
timeMaster(
  baseTime,
  add = 0,
  out = c("auto", "hour", "min", "sec"),
  baseSep = "[\\.,:,\\,',-,\"]"
)
```

Arguments

baseTime, add	either integer of seconds or a time string in the format h:m:s, m:s, or s, with or without leading zeroes
out	a character string indicating the format of the output. One of "auto" (the default which tries to keep the format of 'baseTime'), "hour", "min", or "sec": can be abbreviated.
baseSep	a character string or a regular expression identifying separators in baseTime

Examples

```
## Adding seconds to minutes
timeMaster(30, add="10:00", out = "min")

## Adding strings to integer seconds and returning a numeric value
timeMaster(30, add="10:00")

## Automatic detection of format
timeMaster("01:30:55",add="10:00",out = "auto")
```

writeMEA *Exports analyzed MEA data to .txt files*

Description

Exports analyzed MEA data to .txt files

Usage

```
writeMEA(mea, save.directory, what = c("mea", "ccf"), ...)
```

Arguments

mea	an object of class MEA or MEAlist (see function readMEA)
save.directory	a character string naming a directory
what	a character vector defining what has to be exported. Can be one of 'mea' or 'ccf'.
...	further arguments passed to write.table

Details

'mea' exports the (filtered) MEA data. 'ccf' instead exports the cross-correlation matrix together with all summarizing statistics calculated by MEAccf.

Examples

```
## This example is excluded from test as it takes more than 10s to run
## define a regex filter for the filenames to be read
to_read = c("20[123]_*")

## read the first 4 minutes of the files
path_normal <- system.file("extdata/normal", package = "rMEA")
mea_normal <- readMEA(path_normal, namefilt = to_read, sampRate = 25, s1Col = 1, s2Col = 2,
                      s1Name = "Patient", s2Name = "Therapist",
                      idOrder = c("id","session"), idSep = "_",  skip = 1, nrow = 6000)

## perform ccf analysis
mea_ccf = MEAccf(mea_normal, lagSec = 5, winSec = 60, incSec = 30, r2Z = TRUE, ABS = TRUE)

## export data and analysis
save_path = tempdir()
writeMEA(mea_ccf, save.directory = save_path, what="ccf")
```

Index