Package ‘rsleep’

July 28, 2019

Type Package
Title Analysis of Sleep Data
Version 1.0.1
Author Paul Bouchequet <paul.bouchequet@frenchkpi.com>
Maintainer Paul Bouchequet <paul.bouchequet@frenchkpi.com>
License MIT + file LICENSE
Encoding UTF-8
LazyData true
Imports edfReader, jsonlite, ggplot2, signal, phonTools
Suggests testthat
RoxygenNote 6.1.1
NeedsCompilation no
Repository CRAN
Date/Publication 2019-07-28 11:10:02 UTC

R topics documented:

bands_power ................................................................. 2
check_events ............................................................. 2
detect_rpeaks ............................................................. 3
epochs ........................................................................... 4
eexample_ecg_200hz ....................................................... 4
bands_power

Computes spectral power of bands listed in the bands argument.

Description

`bands_power` calculates power spectral densities estimates using Welch’s method on bands. Bands are computed from spectrogram bands equal or greater than lower limit and inferior to the upper limit.

Usage

```r
bands_power(bands, signal, sRate, broadband = c(0.5, 40))
```

Arguments

- **bands**: A list of bands to compute with lower and upper limits in the form ‘list(c(0,4),c(4,8))’
- **signal**: Numerical vector of the signal.
- **sRate**: Signal sample rate in Hertz.
- **broadband**: The broadband to normalize by.

Value

A list of bands powers.

Examples

```r
bands_power(bands = list(c(0,4),c(4,8)),signal = sin(c(1:10000)),sRate = 200)
```
check_events

Check events dataframe.

Description

Check events dataframe.

Usage

check_events(e)

Arguments

e  Events dataframe. Dataframe must have `begin` (POSIXt), `end` (POSIXt) and `event` (character) columns.

detect_rpeaks

Detects R peaks in raw ECG signal.

Description

Implements the first part of the Pan & Tompkins algorithms to detect R peaks from a raw ECG signal. Inspiration from https://zenodo.org/record/826614.

Usage

detect_rpeaks(signal, sRate, lowcut = 0, highcut = 15, filter_order = 1, integration_window = 15, refractory = 200)

Arguments

signal  Numerical vector of ECG signal.
sRate   ECG signal sample rate.
lowcut  Butterworth bandpass filter low cut value.
highcut Butterworth bandpass filter high cut value.
filter_order  Butterworth bandpass filter order value.
integration_window  Convolution window size.
refractory  Minimal space between peaks in milliseconds.

Value

A vector of each detected R peaks in seconds from the start.
References


Examples

data("example_ecg_200hz")
detect_rpeaks(example_ecg_200hz, 200)

epochs

Split signals into a list of epochs according to an events dataframe or an epoch duration.

Description

Split signals into a list of epochs according to an events dataframe or an epoch duration.

Usage

ePOCHS(sIgNaLs, sRateS, resample = 100, epoch = 30, startTIme = 0)

Arguments

signals A list of numeric vectors containing signals.
sRates A vector or list of integer values of the signals sample rates.
resample The sample rate to resample all signals. Defaults to 100.
ePOCHs Epochs reference. Can be an events dataframe or the number of seconds of each epoch. Defaults to 30.
startTIme The start timestamp of the signal, used to join events to epoch.

Value

A list of signal chunks

Examples

ePOCHS(list(c(1:1000), c(1:1000)), 100, 2)
**example_ecg_200hz**  
*Sample electrocardiogram signal*

**Description**

10 seconds of ECG from Resmed Nox A1 polysomnograph sampled at 200 Hz expressed in Volts.

**Usage**

example_ecg_200hz

**Format**

A vector of 2000 values.

**Source**

http://www.sommeil-vigilance.fr/

**example_hypnogram_30s**  
*Example hypnogram scored on 30 seconds.*

**Description**

Example hypnogram scored on 30 seconds.

**Usage**

example_hypnogram_30s

**Format**

Dataframe

**Source**

http://cloud.frenchkpi.com/s/wreGqkitWNnWwnP/download
**hypnogram**

Filter and reorder an events dataframe to keep only sleep stages related-events.

**Description**

Remove non-sleep stages events and reorder dataframe rows using the `begin` column.

**Usage**

```r
hypnogram(events, labels = c("N3", "N2", "N1", "REM", "AWA"))
```

**Arguments**

- `events` Events dataframe. Dataframe must have `begin (POSIXt)`, `end (POSIXt)` and `event`
- `labels` Sleep stages labels. Defaults to `c("N3", "N2", "N1", "REM", "AWA")`.

**Value**

hypnogram dataframe.

**Examples**

```r
e <- data.frame(begin = as.POSIXlt(c(1536967800,1536967860,1536967830),origin = "1970-01-01"),
e$end <- as.POSIXlt(c(1536967830,1536967890,1536967860), origin = "1970-01-01"),
e$event = c("back-position","N3","REM")
hypnogram(e)
```

**normalize_cycles**

Normalize sleep cycles scored on Noxturnal software from start and stop flags to unique events.

**Description**

Normalize sleep cycles scored on Noxturnal software from start and stop flags to unique events.

**Usage**

```r
normalize_cycles(events)
```

**Arguments**

- `events` Events dataframe. Dataframe must have `begin (POSIXt)`, `end (POSIXt)` and `event`. Cycles flags must be named `Activity-CLASSICstart`, `Activity-BNstart`, `Activity-BNend`, `Activity-REMstart`, `Activity-REMend`, `Activity-ENstart` or `Activity-ENend`. 
plot_hypnogram

**Examples**

cycles <- data.frame(event = c("Activity-CLASSICstart","Activity-CLASSICend"))
cycles$begin <- as.POSIXct(c("2016-01-16 01:13:30","2016-01-16 01:15:30"))
cycles$end <- as.POSIXct(c("2016-01-16 01:13:30","2016-01-16 01:15:30"))
normalize_cycles(cycles)

---

**plot_hypnogram**  
*Draw a hypnogram with ggplot2.*

**Description**

A hypnogram represents the stages of sleep as a function of time. plot_hypnogram() plot a hypnogram using the ggplot2 library from stages sleep in an event dataframe. REM stage is highlighted in red.

**Usage**

plot_hypnogram(events, labels = c("N3", "N2", "N1", "REM", "AWA"))

**Arguments**

- **events**  
  Events dataframe. Dataframe must have `begin (POSIXt), end (POSIXt)` and `event`

- **labels**  
  Sleep stages labels. Defaults to `c("N3", "N2", "N1", "REM", "AWA")`.

**Value**

a ggplot object.

**References**


**Examples**

e <- data.frame(begin = as.POSIXlt(c(1536967800, 1536967830, 1536967860), origin = "1970-01-01"), end = as.POSIXlt(c(1536967830, 1536967860, 1536967890), origin = "1970-01-01"))
e$event = c("N3", "N3", "REM")
plot_hypnogram(e)
read_events_noxturnal

Read a Noxturnal events file (Unicode CSV format)

Description

Read a Noxturnal events file (Unicode CSV format)

Usage

read_events_noxturnal(dir)

Arguments

dir  Noxturnal events file path.

Value

A dataframe of scored events.

read_mdf  Read a Morpheo Data Format (MDF) directory to a list.

Description

Read a Morpheo Data Format (MDF) directory to a list.

Usage

read_mdf(mdfPath, channels = c(NA), metadata = TRUE)

Arguments

mdfPath  character. MDF path.
channels  character. Channels to read.
metadata  boolean. Read or not the metadata.

Value

A list.

References

Examples

download.file("http://cloud.frenchkpi.com/s/65cm6DMq7SYKQ6J/download",
paste0(tempdir(),"/15012016HD.edf"))
download.file("http://cloud.frenchkpi.com/s/wreGqkitWNnWwnP/download",
paste0(tempdir(),"/15012016HD.csv"))
mdfPath = paste0(tempdir(),"/15012016HD/")
channels = c("C3-M2","ECG")
events = read_events_noxturnal(paste0(tempdir(),"15012016HD.csv"))
write_mdf(paste0(tempdir(),"/15012016HD.edf"),
  mdfPath,
  channels,
  events
)
mdf <- read_mdf(paste0(tempdir(),"/15012016HD.edf"))

spectrogram

Plot the spectrogram of signal.

Description

'spectrogram' resamples signal and use the 'specgram' function from the 'signal' library to compute the spectrogram. Results resolution can be then reduced to quickly plot large signals.

Usage

spectrogram(signal, sRate, maxFreq = 25, n = 1024, window = n * 2,
  overlap = 0, cols = c(rep("#3B9AB2", 9), "#78B7C5", "#EBCC2A",
  "#E1AF00", rep("#F21A00", 6)), freq = 4, plot = TRUE,
  startTime = as.POSIXct("1970/01/01 00:00:00"))

Arguments

  signal  Numerical vector of the signal.
  sRate   Signal sample rate in Hertz.
  maxFreq Maximal frequency to plot in Hertz. Signal will be resampled at maxFreq*2 sample rate.
  n       The size of the Fourier transform window.
  window  Shape of the fourier transform window, defaults to n*2.
  overlap Overlap with previous window, defaults to 0.
  cols    Color scale used for the underlying plot function.
  freq    Aggregate frequency used to lower spectrogram resolution. Defaults to 4.
  plot    Boolean, plot or not the spectrogram.
  startTime Posixct of the signal start. Adjust the x axis labels accordingly.
Value

A spectrogram.

Examples

```r
library(signal)
spectrogram(chirp(seq(-2, 15, by = 0.001), 400, 10, 100, 'quadratic'), 20, n=1024)
```

---

`stages_stats`  
Get stages events related stats in a named vector.

Description

`stages_stats` computes stages related stats.

Usage

`stages_stats(e)`

Arguments

- `e`  
  Events dataframe. Dataframe must have `begin` (POSIXt), `end` (POSIXt) and `event` (character) columns.

Value

- stages vector

Examples

```r
e <- data.frame(begin = as.POSIXlt(seq(from = 0, to = 30*10, by = 30), origin = "1970-01-01"))
e$end <- as.POSIXlt(seq(from = 30, to = 30*11, by = 30), origin = "1970-01-01")
e$event = c("AWA", "N1", "N2", "N3", "REM", "N2", "REM", "N2", "REM", "AWA")
stages_stats(e)
```
write_channel

Write a timeserie to disk using Morpheo Data Format (MDF) guidelines.

Description
Write a timeserie to disk using Morpheo Data Format (MDF) guidelines.

Usage
write_channel(channel, signals, headers, mdfPath, endian = "little")

Arguments
channel character. Channel name.
signals list. European Data Format (EDF) signals list.
headers list. European Data Format (EDF) file headers.
mdfPath character. Morpheo Data Format (MDF) directory path.
endian character. Endianess. "big" or "little". Defaults to "little".

References

write_mdf
Write a European Data Format (EDF) record file to disk using Morpheo Data Format (MDF) guidelines. Target directory is erased if it already exists. Signals are stored in binary file, events and metadata in JavaScript Object Notation (JSON) files.

Description
Write a European Data Format (EDF) record file to disk using Morpheo Data Format (MDF) guidelines. Target directory is erased if it already exists. Signals are stored in binary file, events and metadata in JavaScript Object Notation (JSON) files.

Usage
write_mdf(edfPath, mdfPath, channels = c(NA), events = c(), endian = "little")
Arguments

- `edfPath` character. European Data Format (EDF) file path.
- `mdfPath` character. Morpheo Data Format (MDF) directory path.
- `channels` character. Vector of channels labels to write.
- `events` dataframe. Events dataframe to write. Dataframe must contain `begin (POSIXt)`, `end (POSIXt)` and `event (character)` columns.
- `endian` character. Endianess. "big" or "little". Defaults to "little".

References


Examples

download.file("http://cloud.frenchkpi.com/s/65cm6DMq7SYKQ6J/download", paste0(tempdir(),"/15012016HD.edf"))
download.file("http://cloud.frenchkpi.com/s/wreGqkitWNnWwnP/download", paste0(tempdir(),"15012016HD.csv"))
mdfPath = paste0(tempdir(),"/15012016HD/")
channels = c("C3-M2","ECG")
events = read_events_nocturnal(paste0(tempdir(),"15012016HD.csv"))
write_mdf(paste0(tempdir(),"/15012016HD.edf"),
  mdfPath,
  channels,
  events
)