Package 'activAnalyzer'

October 12, 2022

Title A 'Shiny' App to Analyze Accelerometer-Measured Daily Physical Behavior Data

Version 1.0.5

Description A tool to analyse 'ActiGraph' accelerometer data and to implement the use of the PROactive Physical Activity in COPD (chronic obstructive pulmonary disease) instruments. Once analysis is completed, the app allows to export results to .csv files and to generate a report of the measurement. All the configured inputs relevant for interpreting the results are recorded in the report. In addition to the existing 'R' packages that are fully integrated with the app, the app uses some functions from the 'actigraph.sleepr' package developed by Petkova (2021) <https://github.com/dipetkov/actigraph.sleepr/>.

License GPL (>= 3)

Imports dbplyr (>= 2.1.1), dplyr (>= 1.0.10), flextable (>= 0.8.2), forcats (>= 0.5.2), ggplot2 (>= 3.3.6), golem (>= 0.3.4), hms (>= 1.1.2), lubridate (>= 1.8.0), magrittr (>= 2.0.3), modelr (>= 0.1.9), patchwork (>= 1.1.2), PhysicalActivity, reactable (>= 0.3.0), rmarkdown (>= 2.16), RSQLite, shiny (>= 1.7.2), shinycssloaders (>= 1.0.0), shinydashboard (>= 0.7.2), shinydashboardPlus (>= 2.0.3), shinyFeedback (>= 0.4.0), shinyjs (>= 2.1.0), stringr (>= 1.4.1), tidyr (>= 1.2.1), zoo

Encoding UTF-8

RoxygenNote 7.2.1

URL https://pydemull.github.io/activAnalyzer/, https://github.com/pydemull/activAnalyzer

BugReports https://github.com/pydemull/activAnalyzer/issues

Suggests shinytest, covr, knitr, spelling, testthat (>= 3.0.0), processx, globals, config, tidyselect, DBI, assertthat, htmltools, officer, pkgload, scales, tibble, rlang, tinytex

Language en-US

Depends R (>= 3.4.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Pierre-Yves de Müllenheim [cre, aut]
(<https://orcid.org/0000-0001-9157-7371>)

Maintainer Pierre-Yves de Müllenheim <pydemull@uco.fr>

Repository CRAN

Date/Publication 2022-10-12 14:42:42 UTC

R topics documented:

average_results *Average results over valid days*

Description

This function computes, using valid days only, the mean of each of the metrics obtained using the recap_by_day function. The median can also be obtained with an appropriate configuration of the function.

Usage

```
average_results(data, minimum_wear_time = 10, fun = c("mean", "median"))
```

Arguments

data	A dataframe obtained using the prepare_dataset, mark_wear_time, mark_intensity, and then the recap_by_day functions.
minimum_wear_time	A numeric value (in hours) to set the minimum wear time duration for validating a day.
fun	A character value indicating whether means or medians should be computed.

Value

A dataframe.

Examples

```
file <- system.file("extdata", "acc.agd", package = "activAnalyzer")
mydata <- prepare_dataset(data = file)
mydata_with_wear_marks <- mark_wear_time(
    dataset = mydata,
    TS = "TimeStamp",
    to_epoch = 60,
    cts  = "vm",
    frame = 90,
    allowanceFrame = 2,
    streamFrame = 30
    )
mydata_with_intensity_marks <- mark_intensity(
    data = mydata_with_wear_marks,
    col_axis = "vm",
    equation = "Sasaki et al. (2011) [Adults]",
    sed_cutpoint = 200,
    mpa_cutpoint = 2690,
    vpa_cutpoint = 6167,
    age = 32,
    weight = 67,
    sex = "male",
```

```
    )
summary_by_day <- recap_by_day(
    data = mydata_with_intensity_marks,
    age = 32,
    weight = 67,
    sex = "male",
    valid_wear_time_start = "07:00:00",
    valid_wear_time_end = "22:00:00"
    )
average_results(data = summary_by_day, minimum_wear_time = 10)
```

`compute_bmr` *Compute Basal Metabolic Rate (BMR)*

Description

This function computes Basal Metabolic Rate in kcal/d using a Henry et al. (2005; doi: 10.1079/PHN2005801) equation. This function is wrapped within the `mark_intensity` and `recap_by_day` functions.

Usage

```
compute_bmr(age = 40, sex = c("male", "female", "undefined"), weight = 70)
```

Arguments

`age`	A numeric value in yr.
`sex`	A character value.
`weight`	A numeric value in kg.

Value

A numeric value.

Examples

```
compute_bmr(age = 32, sex = "male", weight = 67)
```

compute_mets *Compute metabolic equivalent of task (MET) values*

Description

This function computes metabolic equivalent of task (METs) from weight, sex, accelerometer counts, and a published equation from one of the following scientific articles: Sasaki et al. (2011; doi:10.1016/j.jsams.2011.04.003); Santos-Lozano et al. (2013; 10.1055/s-0033-1337945); Freedson et al. (1998; doi: 10.1097/00005768-199805000-00021). This function is wrapped within the mark_intensity function.

Usage

```
compute_mets(
  data,
 equation = c("Sasaki et al. (2011) [Adults]", "Santos-Lozano et al. (2013) [Adults]",
   "Freedson et al. (1998) [Adults]", "Santos-Lozano et al. (2013) [Older adults]"),
  weight = 70,
  sex = c("male", "female", "undefined")
)
```

Arguments

data	A dataframe obtained using the prepare_dataset function.
equation	A character string to indicate the equation to be used for estimating METs.
weight	A numeric value in kg.
sex	A character value.

Value

A numeric vector.

Examples

```
library(magrittr)
file <- system.file("extdata", "acc.agd", package = "activAnalyzer")
mydata <- prepare_dataset(data = file)
mydata_with_wear_marks <- mydata %>% mark_wear_time() %>%
dplyr::filter(days == 2 & time >= hms::as_hms("14:00:00") & time <= hms::as_hms("15:00:00"))
mets <- compute_mets(
    data = mydata_with_wear_marks,
    equation = "Sasaki et al. (2011) [Adults]",
    weight = 67,
    sex = "male"
    )
mets
```

compute_pro_actigraph_score *Compute PROactive monitor-based physical activity score for C-PPAC tool*

Description

This function computes the PROactive activity score based on the daily median or mean of step count or vector magnitude unit (in counts/min) obtained using an ActiGraph accelerometer.

Usage

```
compute_pro_actigraph_score(
  x,
  quest = c("C-PPAC", "D-PPAC"),
  metric = c("steps", "vmu"),
  fun = c("median", "mean")
)
```

Arguments

x	A numeric value that should be the daily median or the daily mean of step count or vector magnitude unit following a measurement of physical activity; see Gimeno-Santos et al. (2015, online supplement, p.71, doi: 10.1183/09031936.00183014) and Garcia-Aymerich et al. (2021, supplemental material, p.17; doi: 10.1136/thoraxjnl-2020-214554).
quest	A character value to indicate for which PROactive questionnaire a score of the amount of physical activity should be computed.
metric	A character value to indicate the metric for which the PROactive score should be obtained.
fun	A character value to indicate if the metric used in the function is the median or the mean of the results obtained each day of the measurement.

Value

A numeric value.

Examples

```
compute_pro_actigraph_score(x = 3500, quest = "C-PPAC", metric = "steps", fun = "median")

compute_pro_actigraph_score(x = 340, quest = "C-PPAC", metric = "vmu",  fun = "mean")
```

`compute_pro_score_cppac` *Provide score for each question of the C-PPAC*

Description

This function provides a score (from 0 to 4) in relation to the response to a given question from the C-PPAC questionnaire.

Usage

```
compute_pro_score_cppac(
  x,
 question = c("q1", "q2", "q3", "q4", "q5", "q6", "q7", "q8", "q9", "q10", "q11", "q12"),
  language = c("en", "fr")
)
```

Arguments

`x`	A character string that is the exact response to the considered question from the C-PPAC questionnaire.
`question`	A character value to identify the question to be considered when providing the score.
`language`	A character value for setting the language of the considered questionnaire.

Value

A numeric value.

Examples

```
compute_pro_score_cppac(
    x = "A lot (about 1 hour every day)",
    question = "q1",
    language = "en"
    )
```

compute_pro_score_dppac *Provide score for each question of the D-PPAC*

Description

This function provides a score (from 0 to 4) in relation to the response to a given question from the D-PPAC questionnaire.

Usage

```
compute_pro_score_dppac(
  x,
 question = c("q1", "q2", "q3", "q4", "q5", "q6", "q7", "q8", "q9", "q10", "q11", "q12"),
  language = c("en", "fr")
)
```

Arguments

x	A character string that is the exact response to the considered question from the D-PPAC questionnaire.
question	A character value to identify the question to be considered when providing the score.
language	A character value for setting the language of the considered questionnaire.

Value

A numeric value.

Examples

```
compute_pro_score_dppac(
    x = "Un petit peu (jusqu\u2019\u00e0 10 minutes au total)",
    question = "q1",
    language = "fr"
    )
```

create_fig_mvpa *Create a figure showing the mean daily MVPA time*

Description

The function generates a figure showing mortality hazard ratio in correspondence with daily MVPA minutes. The figure is based on data extracted from Ekelund et al. paper (2019; doi: 10.1136/bmj.l4570).

Usage

```
create_fig_mvpa(score, language = c("en", "fr"))
```

Arguments

score	A numeric value for mean daily MVPA time in minutes.
language	A character value for setting the language with which the figure should be created: en for english; fr for french.

Value

A ggplot object.

Examples

```
create_fig_mvpa(score = 27)
```

create_fig_pal *Create a figure showing the mean daily Physical Activity Level (PAL)*

Description

The function generates a figure showing the daily mean of PAL in correspondence with the FAO (2004; http://www.fao.org/3/y5686e/y5686e07.htm#bm07.3) categories.

Usage

```
create_fig_pal(score, language = c("en", "fr"))
```

Arguments

score	A numeric value for mean daily PAL.
language	A character value for setting the language with which the figure should be created: en for english; fr for french.

Value

A ggplot object.

Examples

```
create_fig_pal(score = 1.8)
```

`create_fig_ratio_mvpa_sed` *Create a figure showing the mean daily MVPA/SED ratio*

Description

The function generates a figure showing mortality hazard ratio in correspondence with the daily mean of the MVPA/SED ratio. The figure is based on data extracted from Chastin et al. paper (2021; doi: 10.1123/jpah.2020-0635).

Usage

```
create_fig_ratio_mvpa_sed(score, language = c("en", "fr"))
```

Arguments

`score`	A numeric value for mean daily MVPA/SED ratio.
`language`	A character value for setting the language with which the figure should be created: `en` for english; `fr` for french.

Value

A ggplot object.

Examples

```
create_fig_ratio_mvpa_sed(score = 0.06)
```

create_fig_res_by_day *Create a figure with metrics shown for each day*

Description

The function generates a figure with several common metrics shown for each day of the physical behavior measurement.

Usage

```
create_fig_res_by_day(
  data,
  minimum_wear_time_for_analysis,
  start_day_analysis,
  end_day_analysis,
  language = c("en", "fr")
)
```

Arguments

data	A dataframe with results obtained using the prepare_dataset, mark_wear_time, mark_intensity, and then the recap_by_day functions.
minimum_wear_time_for_analysis	A numeric value to indicate the minimum number of hours of wear time that was considered to valid a day.
start_day_analysis	A character value to indicate the start of the period that was considered to valid a day based on wear time.
end_day_analysis	A character value to indicate the end of the period that was considered to valid a day based on wear time.
language	A character value for setting the language with which the figure should be created: en for english; fr for french.

Value

A ggplot object

Examples

```
file <- system.file("extdata", "acc.agd", package = "activAnalyzer")
mydata <- prepare_dataset(data = file)
mydata_with_wear_marks <- mark_wear_time(
    dataset = mydata,
    TS = "TimeStamp",
    cts  = "vm",
    frame = 90,
```

```
    allowanceFrame = 2,
    streamFrame = 30
    )
mydata_with_intensity_marks <- mark_intensity(
    data = mydata_with_wear_marks,
    col_axis = "vm",
    equation = "Sasaki et al. (2011) [Adults]",
    sed_cutpoint = 200,
    mpa_cutpoint = 2690,
    vpa_cutpoint = 6167,
    age = 32,
    weight = 67,
    sex = "male",
    )
summary_by_day <- recap_by_day(
    data = mydata_with_intensity_marks,
    age = 32,
    weight = 67,
    sex = "male",
    valid_wear_time_start = "07:00:00",
    valid_wear_time_end = "22:00:00"
    )
create_fig_res_by_day(summary_by_day,
    minimum_wear_time_for_analysis = 10,
    start_day_analysis = "00:00:00",
    end_day_analysis = "23:59:00",
    language = "en")
```

`create_fig_sed` *Create a figure showing the mean daily sedentary (SED) time*

Description

The function generates a figure showing mortality hazard ratio in correspondence with daily SED hours. The figure is based on data extracted from Ekelund et al. paper (2019; doi: 10.1136/bmj.l4570).

Usage

```
create_fig_sed(score, language = c("en", "fr"))
```

Arguments

`score`	A numeric value for mean daily SED time in minutes.
`language`	A character value for setting the language with which the figure should be created: `en` for english; `fr` for french.

Value

A ggplot object.

Examples

```
create_fig_sed(score = 400)
```

`create_fig_steps` *Create a figure showing the mean daily step count*

Description

The function generates a figure showing the daily mean of the daily step count in correspondence with the Tudor-Locke et al. (2011; doi: 10.1186/1479-5868-8-79) categories.

Usage

```
create_fig_steps(score, language = c("en", "fr"))
```

Arguments

`score`	A numeric value for mean daily step count.
`language`	A character value for setting the language with which the figure should be created: `en` for english; `fr` for french.

Value

A ggplot object.

Examples

```
create_fig_steps(score = 12500)
```

`create_flextable_summary`
Create a formatted table of results

Description

The function generates a formatted table with both means and medians of the metrics obtained following the physical behavior measurement.

Usage

```
create_flextable_summary(
  results_summary_means,
  results_summary_medians,
  language = c("en", "fr")
)
```

Arguments

results_summary_means	A dataframe with mean results obtained using the prepare_dataset, mark_wear_time, mark_intensity, recap_by_day, and then the average_results functions.
results_summary_medians	A dataframe with median results obtained using the prepare_dataset, mark_wear_time, mark_intensity, recap_by_day, and then the average_results functions.
language	A character value for setting the language with which the table should be created: en for english; fr for french.

Value

A flextable object

Examples

```
file <- system.file("extdata", "acc.agd", package = "activAnalyzer")
mydata <- prepare_dataset(data = file)
mydata_with_wear_marks <- mark_wear_time(
    dataset = mydata,
    TS = "TimeStamp",
    cts  = "vm",
    frame = 90,
    allowanceFrame = 2,
    streamFrame = 30
    )
mydata_with_intensity_marks <- mark_intensity(
    data = mydata_with_wear_marks,
    col_axis = "vm",
    equation = "Sasaki et al. (2011) [Adults]",
    sed_cutpoint = 200,
    mpa_cutpoint = 2690,
    vpa_cutpoint = 6167,
    age = 32,
    weight = 67,
    sex = "male",
    )
summary_by_day <- recap_by_day(
    data = mydata_with_intensity_marks,
    age = 32,
    weight = 67,
    sex = "male",
    valid_wear_time_start = "07:00:00",
    valid_wear_time_end = "22:00:00"
    )
results_summary_means <- average_results(
    data = summary_by_day,
    minimum_wear_time = 10,
    fun = "mean"
    )
results_summary_medians <- average_results(
```

```
    data = summary_by_day,
    minimum_wear_time = 10,
    fun = "median"
    )
create_flextable_summary(
    results_summary_means,
    results_summary_medians,
    language = "en"
    )
```

`get_guidelines_status` *Get WHO physical activity guidelines status*

Description

Get WHO physical activity guidelines status

Usage

```
get_guidelines_status(value, language = c("en", "fr"))
```

Arguments

`value`	A numeric value to indicate the daily mean of MET-hours spent at moderate-to-vigorous physical activity intensity.
`language`	A character value for setting the language with which the table should be created: `en` for english; `fr` for french.

Value

A character string.

Examples

```
get_guidelines_status(value = 5)
```

get_pal_status *Get FAO physical activity level (PAL) status (http://www.fao.org/3/y5686e/y5686e07.htm#bm07.3)*

Description

Get FAO physical activity level (PAL) status (http://www.fao.org/3/y5686e/y5686e07.htm#bm07.3)

Usage

```
get_pal_status(value, language = c("en", "fr"))
```

Arguments

value	A numeric value to indicate the daily mean of PAL.
language	A character value for setting the language with which the table should be created: en for english; fr for french.

Value

A character string.

Examples

```
get_pal_status(value = 1.8)
```

get_ratio_mvpa_sed_comment *Get comment about the MPVA/SED ratio*

Description

Get comment about the MPVA/SED ratio

Usage

```
get_ratio_mvpa_sed_comment(value, language = c("en", "fr"))
```

Arguments

value	A numeric value to indicate the daily mean of MVPA/SED ratio.
language	A character value for setting the language with which the table should be created: en for english; fr for french.

Value

A character string.

Examples

```
get_ratio_mvpa_sed_comment(value = 0.03)
```

`mark_intensity` *Add intensity metrics*

Description

This function adds several columns to a dataset that contains accelerometer counts data. These columns concern respectively sedentary time (SED), light physical activity time (LPA), moderate physical activity time (MPA), vigorous physical activity time (VPA), metabolic equivalent of task (METs), kilocalories (kcal), and MET-hours when time is spent in moderate-to-vigorous physical activity. For the SED, LPA, MPA, and VPA columns, the function provides, for each epoch, the numeric value 1 when the value of the configured counts variable respectively fulfills the criteria of the SED, LPA, MPA, and VPA category (e.g., for the SED column, 1 may be provided if VM counts are <150 counts/min); otherwise 0 is provided. METs are computed using the `compute_mets` function. METs are computed using a published equation from one of the following scientific articles: Sasaki et al. (2011; doi:10.1016/j.jsams.2011.04.003); Santos-Lozano et al. (2013; 10.1055/s-0033-1337945); Freedson et al. (1998; doi: 10.1097/00005768-199805000-00021). Kilocalories are computed as follows. For non-SED epochs, MET values are multiplied by BMR expressed in kcal/min when using the Santos-Lozano et al. (2013) equations since, in that study, METs were multiples of the measured (not standard) resting metabolic rate. When using the Sasaki et al. (2011) and Freedson et al. (1998) equations, the MET values are multiplied by weight and 1/60 since, in those studies, METs were multiples of standard resting metabolic rate (i.e., 3.5 mLO2/min/kg) and a standard MET is approximately equivalent to 1 kcal/kg/h (Butte et al., 2012; doi: 10.1249/MSS.0b013e3182399c0e). For SED epochs, BMR expressed in kcal/min is directly used. BMR is computed using the `compute_bmr` function that uses sex, age, and weight inputs, and one of the equations retrieved from the paper by Henry et al. (2005; doi: 10.1079/PHN2005801). MET-hours are obtained by multiplying METs by time related to each epoch (e.g., 1/60e of an hour for 1-min epochs), only when the MET value is >=3.

Usage

```
mark_intensity(
  data,
  col_axis = "vm",
  sed_cutpoint = 200,
  mpa_cutpoint = 2690,
  vpa_cutpoint = 6167,
 equation = c("Sasaki et al. (2011) [Adults]", "Santos-Lozano et al. (2013) [Adults]",
  "Freedson et al. (1998) [Adults]", "Santos-Lozano et al. (2013) [Older adults]"),
  age = 40,
  weight = 70,
```

```
  sex = c("male", "female", "undefined"),
  dates = NULL
)
```

Arguments

data	A dataframe obtained using the prepare_dataset and then the mark_wear_time functions.
col_axis	A character value to indicate the name of the variable to be used for determining intensity categories.
sed_cutpoint	A numeric value below which time is considered as spent in sedentary behavior (in counts/min). You must convert to counts/min if you want to use a cut-point initially developed using an epoch shorter than 60 s. In the case where the epoch of the dataset would be shorter than 60 s, the function will multiply the counts data so that it corresponds to the cut-point expressed in counts/min. You must provide a value inferior or equal to 60.
mpa_cutpoint	A numeric value at and above which time is considered as spent in moderate-to-vigorous physical activity (in counts/min). You must convert to counts/min if you want to use a cut-point initially developed using an epoch shorter than 60 s. In the case where the epoch of the dataset would be shorter than 60 s, the function will multiply the counts data so that it corresponds to the cut-point expressed in counts/min. You must provide a value inferior or equal to 60.
vpa_cutpoint	A numeric value at and above which time is considered as spent in vigorous physical activity (in counts/min). You must convert to counts/min if you want to use a cut-point initially developed using an epoch shorter than 60 s. In the case where the epoch of the dataset would be shorter than 60 s, the function will multiply the counts data so that it corresponds to the cut-point expressed in counts/min. You must provide a value inferior or equal to 60.
equation	A character string to indicate the equation to be used for estimating METs.
age	A numeric value in yr.
weight	A numeric value in kg.
sex	A character value.
dates	A character vector containing the dates to be retained for analysis. The dates must be with the "YYYY-MM-DD" format.

Value

A dataframe.

Examples

```
file <- system.file("extdata", "acc.agd", package = "activAnalyzer")
mydata <- prepare_dataset(data = file)
mydata_with_wear_marks <- mark_wear_time(
    dataset = mydata,
    TS = "TimeStamp",
    to_epoch = 60,
```

```
    cts  = "vm",
    frame = 90,
    allowanceFrame = 2,
    streamFrame = 30
    )
mydata_with_intensity_marks <- mark_intensity(
    data = mydata_with_wear_marks,
    col_axis = "vm",
    equation = "Sasaki et al. (2011) [Adults]",
    sed_cutpoint = 200,
    mpa_cutpoint = 2690,
    vpa_cutpoint = 6167,
    age = 32,
    weight = 67,
    sex = "male",
    )
head(mydata_with_intensity_marks)
```

`mark_wear_time` *Mark dataset for nonwear/wear time*

Description

This function wraps the `dataCollapser` and the `wearingMarking` functions from the `PhysicalActivity` package. After collapsing data, the function adds `time` and `date` columns. Then, the function analyzes the dataset for nonwear time detection. Finally, the function adds two variables to the dataset: the variable `non_wearing_count` that contains the number 1 when the device was *not* worn (otherwise, 0 is used), and the variable `wearing_count` that contains the number 1 when the device was worn (otherwise, 0 is used).

Usage

```
mark_wear_time(
  dataset,
  TS = "TimeStamp",
  to_epoch = 60,
  cts = "vm",
  frame = 90,
  allowanceFrame = 2,
  streamFrame = 30
)
```

Arguments

`dataset`	A dataframe obtained using the `prepare_dataset` function.
`TS`	A character value indicating the name of the variable where date and time information are provided.

to_epoch	A numeric value indicating the length of the epoch to use (in seconds) for accumulating data. The value must be superior or equal to the recording epoch that was used for the measurement.
cts	A character value indicating the name of the variable used by the nonwear/wear detection algorithm.
frame	A numeric value for the length of the time window (in minutes) used to detect nonwear/wear time.
allowanceFrame	A numeric value for the length of the time window (in minutes) with nonzero counts allowed within the detected nonwear period.
streamFrame	A numeric value for the length of the time window required around the detected activity to validate nonwear time.

Value

A dataframe.

Examples

```
file <- system.file("extdata", "acc.agd", package = "activAnalyzer")
mydata <- prepare_dataset(data = file)
mydata_with_wear_marks <- mark_wear_time(
    dataset = mydata,
    TS = "TimeStamp",
    to_epoch = 60,
    cts  = "vm",
    frame = 90,
    allowanceFrame = 2,
    streamFrame = 30
    )
head(mydata_with_wear_marks)
```

---

plot_data *Plot accelerometer data for each day*

Description

This function plots accelerometer data against time for each day of measurement, with the possibility to specify the metric to visualize.

Usage

```
plot_data(
  data,
  metric = "axis1",
  col_time = "time",
  col_nonwear = "non_wearing_count",
  col_wear = "wearing_count"
)
```

Arguments

data	A dataframe obtained using the prepare_dataset and then the mark_wear_time functions.
metric	A character value to indicate the name of the variable to be plotted against time.
col_time	A character value to indicate the name of the variable to plot time data.
col_nonwear	A character value to indicate the name of the variable used to count nonwear time.
col_wear	A character value to indicate the name of the variable used to count wear time.

Value

A ggplot object.

Examples

```
file <- system.file("extdata", "acc.agd", package = "activAnalyzer")
mydata <- prepare_dataset(data = file)
mydata_with_wear_marks <- mark_wear_time(
    dataset = mydata,
    TS = "TimeStamp",
    to_epoch = 60,
    cts  = "vm",
    frame = 90,
    allowanceFrame = 2,
    streamFrame = 30
    )
plot_data(
    data = mydata_with_wear_marks,
    metric = "vm",
    col_time = "time",
    col_nonwear = "non_wearing_count",
    col_wear = "wearing_count"
    )
```

---

plot_data_with_intensity

Plot accelerometer data for each day with both nonwear time and physical activity intensity categories

---

Description

This function plots accelerometer data with intensity categories against time for each day of measurement, with the possibility to specify the metric to visualize.

Usage

```
plot_data_with_intensity(
  data,
  metric = "axis1",
  col_time = "time",
  col_nonwear = "non_wearing_count",
  col_wear = "wearing_count",
  valid_wear_time_start = "00:00:00",
  valid_wear_time_end = "23:59:00"
)
```

Arguments

data	A dataframe obtained using the prepare_dataset, mark_wear_time, and then the mark_intensity functions.
metric	A character value to indicate the name of the variable to be plotted against time.
col_time	A character value to indicate the name of the variable to plot time data.
col_nonwear	A character value to indicate the name of the variable used to count nonwear time.
col_wear	A character value to indicate the name of the variable used to count wear time.
valid_wear_time_start	A character value with the HH:MM:SS format to set the start of the daily period that will be considered for computing valid wear time.
valid_wear_time_end	A character value with the HH:MM:SS format to set the end of the daily period that will be considered for computing valid wear time.

Value

A ggplot object.

Examples

```
file <- system.file("extdata", "acc.agd", package = "activAnalyzer")
mydata <- prepare_dataset(data = file)
mydata_with_wear_marks <- mark_wear_time(
    dataset = mydata,
    TS = "TimeStamp",
    to_epoch = 60,
    cts  = "vm",
    frame = 90,
    allowanceFrame = 2,
    streamFrame = 30
    )
mydata_with_intensity_marks <- mark_intensity(
    data = mydata_with_wear_marks,
    col_axis = "vm",
```

```
    equation = "Sasaki et al. (2011) [Adults]",
    sed_cutpoint = 200,
    mpa_cutpoint = 2690,
    vpa_cutpoint = 6167,
    age = 32,
    weight = 67,
    sex = "male",
    )
plot_data_with_intensity(
    data = mydata_with_intensity_marks,
    metric = "vm",
    valid_wear_time_start = "00:00:00",
    valid_wear_time_end = "23:59:00"
    )
```

`prepare_dataset` *Prepare accelerometer data*

Description

This function reads an .agd file and then creates the vector magnitude variable as follows: $vm = \sqrt{axis1^2 + axis2^2 + axis3^2}$. The .agd file must contain at least the following columns:

- **axis1**
- **axis2**
- **axis3**
- **steps**
- **inclineStanding**
- **inclineSitting**
- **inclineLying**
- **inclineOff**

Usage

```
prepare_dataset(data)
```

Arguments

`data`	Path to an .agd file that was exported from ActiLife software.

Value

A dataframe.

Examples

```
file <- system.file("extdata", "acc.agd", package = "activAnalyzer")
mydata <- prepare_dataset(data = file)
head(mydata)
```

`rasch_transform` *Compute Rasch transformation for PROactive scores*

Description

This function provides the 0-100 Rasch scaled score of a given C-PPAC or D-PPAC raw score (based on: Garcia-Aymerich J, et al. Thorax 2021;0:1–11. doi: 10.1136/thoraxjnl-2020-214554).

Usage

```
rasch_transform(
  x,
  quest = c("C-PPAC", "D-PPAC"),
  score = c("difficulty", "quantity")
)
```

Arguments

`x`	A numeric value that is the difficulty score (between 0 and 40 for C-PPAC or 0 and 20 for D-PPAC) or the quantity score (between 0 and 15 for C-PPAC or 0 and 17 for D-PPAC) obtained using a PROactive questionnaire.
`quest`	A character value to indicate with which PROactive questionnaire the raw score has been obtained.
`score`	A character value.

Value

A numeric value.

Examples

```
rasch_transform(33, quest = "C-PPAC", score = "difficulty")
```

read_agd *Read activity counts from an *.agd file*

Description

Read ActiGraph sleep watch data from a database stored in an AGD file. Return a tibble. (Code is from actigraph.sleepr package https://github.com/dipetkov/actigraph.sleepr/. See LICENCE.note file in the app skeleton.)

Usage

```
read_agd(file, tz = "UTC")
```

Arguments

file	Full path to an agd file to read.
tz	Time zone to convert DateTime ticks to POSIX time.

Value

A tibble of activity data with at least two columns: timestamp and axis1 counts. Optional columns include axis2, axis2, steps, lux and inclinometer indicators (incline off, standing, sitting and lying). The device settings are stored as attributes, which include epochlength.

References

The AGD file format is described in the ActiLife 6 Manual. https://actigraphcorp.com/support/manuals/actilife-6-manual/

See Also

read_agd_raw()

Examples

```
file <- system.file("extdata", "acc.agd",
  package = "activAnalyzer"
)
read_agd(file)
```

read_agd_raw *Read an *.agd file, with no post-processing*

Description

Read ActiGraph sleep watch data from an SQLite database stored in an AGD file and return a list with (at least) five tables: data, sleep, filters, settings, awakenings. The tables have the schema described in the ActiLife 6 User manual and the timestamps are converted from Unix time format to human-readable POSIXct representation. Code is from actigraph.sleepr package https://github.com/dipetkov/actigraph.sleepr/. See LICENCE.note file in the app skeleton.

Usage

```
read_agd_raw(file, tz = "UTC")
```

Arguments

file	Full path to an agd file to read.
tz	Time zone to convert DateTime ticks to POSIX time.

Details

Some ActiGraph devices contain a capacitive sensor to detect monitor removal when worn against the skin. If that data is available, the return list includes a capsense table as well.

Value

A list of five tables: settings, data, filters, sleep, awakenings and, if available, capsense.

References

ActiLife 6 User's Manual by the ActiGraph Software Department. 04/03/2012.

covertagd: R package for converting agd files from ActiGraph into data.frames.

See Also

read_agd()

Examples

```
file <- system.file("extdata", "acc.agd",
  package = "activAnalyzer"
)
str(read_agd_raw(file))
```

recap_by_day *Summarize results by day*

Description

This function summarizes accelerometer results for each day of the measurement period.

Usage

```
recap_by_day(
  data,
  col_time = "time",
  valid_wear_time_start = "00:00:00",
  valid_wear_time_end = "23:59:59",
  age = 40,
  weight = 70,
  sex = c("male", "female", "undefined")
)
```

Arguments

data	A dataframe obtained using the prepare_dataset, mark_wear_time, and then the mark_intensity functions.
col_time	A character value indicating the name of the variable where time information is provided.
valid_wear_time_start	A character value with the HH:MM:SS format to set the start of the daily period to consider for computing valid wear time.
valid_wear_time_end	A character value with the HH:MM:SS format to set the end of the daily period to consider for computing valid wear time.
age	A numeric value in yr.
weight	A numeric value in kg.
sex	A character value.

Details

The following metrics are computed from epochs corresponding to valid wear time:

- **wear_time:** total wear time computed using the daily period defined in the function
- **total_counts_axis1:** total counts for the vertical axis
- **total_counts_vm:** total counts for the vector magnitude
- **axis1_per_min:** mean of the counts per minute for the vertical axis
- **vm_per_min:** mean of the counts per minute for the vector magnitude

- **total_steps:** total step count
- **total_kcal:** total kilocalories
- **minutes_SED:** total minutes spent in SED behavior
- **minutes_LPA:** total minutes spent in LPA behavior
- **minutes_MPA:** total minutes spent in MPA behavior
- **minutes_VPA:** total minutes spent in VPA behavior
- **minutes_MVPA:** total minutes spent in MVPA behavior
- **percent_SED:** proportion of wear time spent in SED behavior
- **percent_LPA:** proportion of wear time spent in LPA behavior
- **percent_MPA:** proportion of wear time spent in MPA behavior
- **percent_VPA:** proportion of wear time spent in VPA behavior
- **percent_MVPA:** proportion of wear time spent in MPVA behavior
- **max_steps_60min:** best step accumulation per minute averaged over a window of 60 continuous minutes
- **max_steps_30min:** best step accumulation per minute averaged over a window of 30 continuous minutes
- **max_steps_20min:** best step accumulation per minute averaged over a window of 20 continuous minutes
- **max_steps_5min:** best step accumulation per minute averaged over a window of 5 continuous minutes
- **max_steps_1min:** best step accumulation per minute over a window of 1 minute
- **peak_steps_60min:** step accumulation per minute averaged over the best 60 continuous or discontinuous minutes
- **peak_steps_30min:** step accumulation per minute averaged over the best 30 continuous or discontinuous minutes
- **peak_steps_20min:** step accumulation per minute averaged over the best 20 continuous or discontinuous minutes
- **peak_steps_5min:** step accumulation per minute averaged over the best 5 continuous or discontinuous minutes
- **peak_steps_1min:** step accumulation per minute over the best minute (same result as for `max_steps_1min`)
- **mets_hours_mvpa:** total MET-hours spent during MPVA behavior
- **ratio_mvpa_sed:** ratio between MVPA and SED times (`minutes_MVPA` / `minutes_SED`)

PAL is computed by dividing total energy expenditure (TEE) by BMR. TEE is obtained by summing the kilocalories measured during valid wear time epochs and the kilocalories related to BMR expended during nonwear time epochs (it is assumed that the periods where the device was not worn corresponded to sleeping periods, during which energy expenditure is near of BMR), and by multiplying this sum by 10/9 to take into account the thermic effect of food. Of course, such calculations may conduct to underestimate TEE and PAL if the device was removed during prolonged periods of physical activity. Moreover, even if the device was correctly worn, the estimate of PAL is very approximate since both BMR and kilocalories are estimated using methods that may not be accurate at the individual level.

Value

A dataframe.

Examples

```
file <- system.file("extdata", "acc.agd", package = "activAnalyzer")
mydata <- prepare_dataset(data = file)
mydata_with_wear_marks <- mark_wear_time(
    dataset = mydata,
    TS = "TimeStamp",
    to_epoch = 60,
    cts  = "vm",
    frame = 90,
    allowanceFrame = 2,
    streamFrame = 30
    )
mydata_with_intensity_marks <- mark_intensity(
    data = mydata_with_wear_marks,
    col_axis = "vm",
    equation = "Sasaki et al. (2011) [Adults]",
    sed_cutpoint = 200,
    mpa_cutpoint = 2690,
    vpa_cutpoint = 6167,
    age = 32,
    weight = 67,
    sex = "male",
    )
recap_by_day(
    data = mydata_with_intensity_marks,
    age = 32,
    weight = 67,
    sex = "male",
    valid_wear_time_start = "07:00:00",
    valid_wear_time_end = "22:00:00"
    )
```

---

run_app *Run the Shiny Application*

Description

Run the Shiny Application

Usage

```
run_app(
  onStart = NULL,
  options = list(),
```

```
  enableBookmarking = NULL,
  uiPattern = "/",
  ...
)
```

Arguments

onStart	A function that will be called before the app is actually run. This is only needed for shinyAppObj, since in the shinyAppDir case, a global.R file can be used for this purpose.
options	Named options that should be passed to the runApp call (these can be any of the following: "port", "launch.browser", "host", "quiet", "display.mode" and "test.mode"). You can also specify width and height parameters which provide a hint to the embedding environment about the ideal height/width for the app.
enableBookmarking	Can be one of "url", "server", or "disable". The default value, NULL, will respect the setting from any previous calls to enableBookmarking(). See enableBookmarking() for more information on bookmarking your app.
uiPattern	A regular expression that will be applied to each GET request to determine whether the ui should be used to handle the request. Note that the entire request path must match the regular expression in order for the match to be considered successful.
...	arguments to pass to golem_opts. See ?golem::get_golem_options for more details.

Value

No return value, called for side effects.

---

tbl_agd *A* tibble *of activity data exported by an ActiGraph device*

Description

This tibble has several attributes, most importantly, epochlength. (Code is from actigraph.sleepr package https://github.com/dipetkov/actigraph.sleepr/. See LICENCE.note file in the app skeleton.)

Usage

```
tbl_agd(data, settings)
```

Arguments

data	A data frame of raw activity counts.
settings	A data frame of device settings.

Value

A tibble containing accelerometer data and having measurement settings as attributes.

Index