Package 'psyosphere'

May 3, 2018

Type Package

Title Analyse GPS Data

Version 0.1.4

Date 2018-05-03

Maintainer Benjamin Ziepert <feedback-psyosphere@analyse-gps.com>

Description Analyse location data such as latitude,
longitude, and elevation. Based on spherical trigonometry, variables such as speed, bearing, and distances can be calculated from moment to moment, depending on the sampling frequency of the equipment used, and independent of scale. Additionally, the package can plot tracks, coordinates, and shapes on maps, and sub-tracks can be selected with point-in-polygon or other techniques. The package is optimized to support behavioural science experiments with multiple tracks. It can detect and clean up errors in the data, and resulting data can be exported to be analysed in statistical software or geographic information systems (GIS).

Depends R (>= 2.10)

License MIT + file LICENSE

Imports ggmap, rgdal, plyr, geosphere, ggplot2, sp, lubridate,
RgoogleMaps, SDMTools, stats

RoxygenNote 6.0.1

NeedsCompilation no

Author Benjamin Ziepert [cre, aut],
Elze G Ufkes [ctb],
Peter W de Vries [ctb]

Repository CRAN

Date/Publication 2018-05-03 11:38:56 UTC

R topics documented:

`about analysing tips` *Guide how to analyse GPS data*

Description

Guide how to analyse GPS data

Details

The following guide explains the steps to analyse GPS data.

1. **Save data as `psyo` data frame**. You should create a psyo data frame by reading GPX files with dir_get_gpx and add additional information for each participant with dir_add_csv.
2. **Remove not needed data**. By removing data, you can greatly improve calculation speed. (1) You can remove data by specifying a begin and end time (See example section below). (2) You could create a polygon of the area you want to select with mark_inside_polygon and then select only the data you want. To determine a good polygon, you can draw the tracks with plot_tracks. (3) You can select data between a start and finish polygon with select_between_polygons.
3. **Create a test sample**. If you have a lot of data calculations can take hours. Especially if you use the functions distance_psyo and distance_peers. To speed things up you can first create a test sample with average_coordinates and select_test_sample. In most cases, it is better to use average_coordinates. When everything works, you can run your script with the original coordinates.
4. **clean-up the data**. Before the clean-up calculate the descriptive summary with des_summary so you know which data is removed. After that you can clean-up the data with the following steps. (1) You should average coordinates that have the same time stamp with average_duplicates. You can mark gaps with (2) mark_speed_gaps and (3) mark_time_gaps.
5. **Plot tracks**. You can plot the tracks with plot_tracks to check how the tracks look like, if the right data is removed, if the gaps are marked correctly, ...
6. **Calculate your data**. You now can calculate different measures. It is important to do this after removing not needed data and the clean-up. Otherwise unwanted data could be included into the calculations. If you already calculated a measure you can calculate it again and the old column will be overwritten.
7. **Create summary for each participant / tracker**. With des_summary and other functions with the prefix "des_" you can calculate a summary for each participant.

You can see these steps implemented with the demo smuggler1. See also about_demos.

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

about_common_mistakes, about_demos

Examples

```
# Remove data before begin and after end -------------------------------------
data(psyo_rounds2)
tracks <- psyo_rounds2
begin <- as.POSIXct("2015-09-03 14:00:00")
end <- as.POSIXct("2015-09-03 14:20:00")
tracks <- tracks[ tracks[,"time"] > begin & tracks[,"time"] < end ,]
```

---

about common mistakes *Tips to prevent common mistakes*

Description

Tips to prevent common mistakes

Details

Forgetting to use weighted statistics. Mostly the coordinates don't have an even time interval. This can be because of the missing data, planned data gaps or deviations in the GPS tracker. To prevent this, you can calculate the time difference between coordinates with difftime and used it as weight for weighted descriptive statistics.

Recalculating track data after removing gaps. After removing gaps, you should be careful to recalculate speed, time difference, etc. since this function can't see that the gaps are removed. To work around this, you can just omit gaps with the descriptive functions that begin with "des_". They can ignore gaps.

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

about_analysing_tips, about_demos, difftime, wt.mean

Examples

```
# Example forgetting to use weighted statistics --------------------------------
data(psyo)
tracks <- t_speed(psyo)
tracks <- t_time_difference(tracks, units = "secs")

# Without weighted statistics
mean(tracks[,c("speed")], na.rm = TRUE)
sd(tracks[,c("speed")], na.rm = TRUE)

# With weighted statistics
des_mean(tracks, "speed", "time_difference", t_id = "")

SDMTools::wt.mean(
  tracks[,c("speed")], as.numeric(tracks[,c("time_difference")])
)
SDMTools::wt.sd(
  tracks[,c("speed")], as.numeric(tracks[,c("time_difference")])
)
```

---

about demos *Explanation how the demo files can be accessed*

Description

The package contains some demonstrations how the different functions can be used. The demonstrations are experiments that are used to develop 'psyosphere', thus it is real experimental data.

Be aware the that it can take two hours ore more to complete some of the demos. Therefore, the demos frequently will save the progress. You can access the demo files directly to restore your progress.

You can find more information about the experiments on [analyse-gps.com](https://analyse-gps.com/experiments/ut-smuggle-experiment/).

Details

1. **Move demo file**. The demo file will download a zip file of about 2MB that unpacks into about 200MB. You can move the demo files to an appropriate location. In the example section is explained how you can find the demo files.
2. **Set working directory**. For the demo files to work, the working directory must be set to the directory that contains the demo. You can see in the example section how you can do that.
3. **Step by step execution**. The demos contain sometimes hundreds of thousands of coordinates. Therefore, it is advisable to calculate the demos step by step and to compress the coordinates first with average_coordinates.

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **Creative Commons:** "" by B. Ziepert, E. G. Ufkes & P. W. de Vries from analyse-gps.com / CC-BY-SA-4.0
- **APA**: de Vries, P. W., et al. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

Examples

```
# Get a list of the demo's
demo(package = "psyosphere")

# Find the location of a demo file
system.file("demo", "smuggler1.R", package = "psyosphere")

# Run demo
demo("smuggler1", package = "psyosphere")

# How to restore progress ----------------------------------------------------

# Set directory (see top of the file)
```

```
demo_dir <- tempdir()
dir <- paste0(demo_dir,"/leadership1/")

# Clean environment and load data (see top of each section)
rm(list=setdiff(ls(), "dir"))
load(paste0(dir,"rdata/01.RData"))
```

`about documentation` *psyosphere documentation guideline*

Description

Guideline for creating 'psyosphere' documentation.

Details

File names

1. The file names of documentation about 'psyosphere' in general begins with "about ". Exception is the package documentation file.
2. The file names cannot begin with "aa" or "ab" to prevent that they are listed before the "about" files..
3. The file names for functions documentations are identical to the function names. Exception are private functions. See also about functions.
4. The file name for data always begin with "psyo".

Files

For each documentation 2 files will be created.

1. A documentation file. A file that contains the documentation for the function and has the same file name as the function file. See also: about functions.
2. A documentation example file. A file that contains the example code for the documentation. This file has the same file name as the function file with the prefix "man_" and is stored in the directory "code_examples/man".

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

`about functions` *psyosphere functions guideline*

Description

Guideline for creating 'psyosphere' functions.

Details

Function patterns

The different function categories follow different structures. The functions are alphabetically grouped by a prefix. The prefixes are:

apply meta tasks for data frames

average average location of multiple coordinates to one coordinate

des descriptive about the tracks

dir getting data from directories

distance calculate distance to something else for each coordinate

mark `logical` lists

plot create Google map plots

select select coordinates within tracks

t calculations per coordinate.

val validate variables

t group

The calculation information of the t group will be stored at the arriving coordinate. For example, the bearing from point 1 to point 2 will be stored with point 2 and the first point 1 of a track has a `NA`. Storing with point 1 or point 2 has both advantages and disadvantages. The data is stored with point 2 because of the compatibility with `select_gaps` and `select_without_gaps` When all coordinates with gaps get removed the data of the t group like speed, bearing, etc. gets also removed and for instance an average without the gaps can be calculated.

In the example section, you can find the basic structure of the t group. All groups follow a similar basic structure.

Function guidelines

The following list is a guide how a function should look like.

Function format:

1. Function name is it short.
2. Function name is it alphabetical logical sorted. For instance, start stored variables with "data" or directory operation with "dir".

3. Function name cannot begin with "aa" or "ab" to prevent that they are listed before the "about" documentation files..
4. Check each input variable. With `val_var` or `val_psyo`
5. Every function can handle the psyo format.
6. No longer than 30 lines.
7. Childe function are private. Functions that only support a main function and don't need to be accessed by the package user end with "_private" and are stored in the same file as the main function. Private functions can be accessed with `"psysophere:::"`

File format:

1. Each function has its own file. Exception are private functions that end on "_private" and support the main function.
2. The file name is identical with the function name

Files

For each function 4 different files will be created.

1. A function file. A file that contains the functions itself.
2. A documentation file. A file that contains the documentation for the function and has the same file name as the function file. See also: about documentation.
3. A documentation example file. A file that contains the example code for the documentation. This file has the same file name as the function file with the prefix "man_" and is stored in the directory "code_examples/man".
4. A test file. A file that contains the test code for the package test. This file has the same file name as the function file with the prefix "test_" and is stored in the directory "tests/testthat".

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: `<feedback-psyosphere@analyse-gps.com>`.

Examples

```
# ------------------------------------------------------------------------------
# Example calculation R file ---------------------------------------------------
# ------------------------------------------------------------------------------

t_distance <- function(
  tracks, bind = TRUE, drop = TRUE, cname = "distances_in_m", t_id = "id"
) {

  # Check variables
  e <- val_psyo(tracks, 0, 0, 0, 2, 2); if (e != "") {stop(e)}
  e <- val_var(bind, "logical"); if (e != "") {stop(e)}
  e <- val_var(drop, "logical"); if (e != "") {stop(e)}
  e <- val_var(cname, "character"); if (e != "") {stop(e)}
  e <- val_cname(tracks, t_id); if (e != "") {stop(e)}

  # Add bearings per track
  result <- apply_tracks(
    tracks,
    "distance_exec_private(eval_track)",
    t_id = t_id
  )

  # Reformat result
  result <- data.frame(result)
  result <- plyr::rename(result, c("result" = cname))

  # Return result
  result <- bind_drop_private(tracks, result, bind, drop)
  return(result)

}

distance_exec_private <- function(tracks) {

  # Get lat and lon from next observation
  current <- subset(tracks, select = c("lon","lat"))
  previous <- apply_shift(
    tracks, "-1", FALSE, c("lon","lat"), t_id = ""
  )

  # Get distances
  distances_in_m <- geosphere::distHaversine(previous, current)

  return(distances_in_m)

}

# ------------------------------------------------------------------------------
# Template for test file -------------------------------------------------------
# ------------------------------------------------------------------------------
```

```
# Print title
cat("\nTesting <function_name>()\n")

# Get data
data("psyo_rounds2")
tracks <- psyo_rounds2

# Calculations


# Check results
# if (NROW(________) != ________) { stop("Wrong number of observations") }
# if (NCOL(________) != ________) { stop("Wrong number of variables") }
# val_psyo(________)
# test_sum <- sum(____)
# if (round(test_sum,3) != round(____,3)) {stop("Wrong test_sum")}
```

`apply_shift` *Copy columns and offset the index of the copied column*

Description

Copy columns and offset the index of the copied column

Usage

```
apply_shift(tracks, factor = 1, bind = TRUE, csubset = "", t_id = "id")
```

Arguments

`tracks`	psyo. Data frame with tracks.
`factor`	`character` or `numeric`. Number of shifted copied to be created. With - or + a direction can be indicated. For instance, +1 copies the value of the following coordinate. -1 copies the value of the previous coordinate.
`bind`	`logical`. Whether to bind the row to `tracks` or to return it as separate column.
`csubset`	`list`. A list of column names in `tracks` that will be copied.
`t_id`	*character* or *numeric*. Column name in `tracks` that identifies the separate tracks.

Value

psyo

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Note

This function drops sometimes attributes.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

psyo,apply_tracks

Examples

```
data(psyo)
apply_shift(psyo, csubset = c("lon", "lat"))
```

---

apply_tracks *Run function on each track in a psyo data frame*

Description

Run function on each track in a psyo data frame. The function is in form of a *character* expression.

Usage

```
apply_tracks(tracks, exp,
  arg1 = "", arg2 = "", arg3 = "", arg4 = "",arg5 = "", arg6 = "", arg7 = "",
  arg8 = "", arg9 = "", t_id = "id", info = FALSE
)
```

Arguments

tracks	psyo. Data frame containing tracks in psyo format.
exp	*character*. The function and arguments that will be evaluated.
arg1	*multiple*. Arguments that will be sent to the target function.
arg2	*multiple*. Arguments that will be sent to the target function.
arg3	*multiple*. Arguments that will be sent to the target function.
arg4	*multiple*. Arguments that will be sent to the target function.
arg5	*multiple*. Arguments that will be sent to the target function.
arg6	*multiple*. Arguments that will be sent to the target function.
arg7	*multiple*. Arguments that will be sent to the target function.
arg8	*multiple*. Arguments that will be sent to the target function.
arg9	*multiple*. Arguments that will be sent to the target function.
t_id	Unique by time sorted ID for every coordinate within a track. Use `t_id = ""` to make no selection but take all data.
info	*logical*. Measures the time consumption for each track calculation.

Value

psyo

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Warning

Don't use this function for 'psyosphere' main functions. Most of the 'psyosphere' functions have `apply_tracks()` already included. By adding it again you can get strange results or break the function.

Only return the changed "eval_track" as result. The function is splitting a psyo data frame in sub tracks. After changes are applied that sub tracks or merged together again. Therefore, it is important to only work in the sub track. If for instance every time the psyo data frame is returned by the evaluated function than the data frame gets stacked again and again on top of itself. See the examples below for how this can look like.

Author(s)

Benjamin Ziepert

See Also

psyo, apply_shift

Examples

```
# Working examples ------------------------------------------------------------

# Test function for examples
test_sum <- function(track, more = 0) {
  track$lon_sum <- sum(track$lon) + more
  return(track)
}

# Simple example
data(psyo)
psyo <- apply_tracks(
  psyo,"test_sum(eval_track)"
)

# See all data as one track
data(psyo)
psyo <- apply_tracks(
  psyo,"test_sum(eval_track)", t_id =""
)

# Use of arguments
data(psyo)
psyo <- apply_tracks(
  psyo,"test_sum(eval_track, arg1)", arg1 = 5
)

# What not to do --------------------------------------------------------------

# Only return the changed "eval_track" as result. The following examples show
# what can go wrong otherwise.

test_wrong <- function(selected_track, all_tracks) {
  all_tracks$sum <- all_tracks$lon + all_tracks$lat
  return(all_tracks)
}

data(psyo)
psyo <- psyo[psyo[,c("p_id")]== 0,]

psyo <- apply_tracks(
  psyo,"test_wrong(eval_track, arg1)", arg1 = psyo
)
```

`average_coordinates` *Calculates the mean position of coordinates within a time interval.*

Description

Calculates the mean position of coordinates within a time interval.

Usage

```
average_coordinates(tracks, num, units = "minutes", t_id = "id")
```

Arguments

`tracks`	`psyo`. Data frame with tracks.
`num`	*num* The amount of time.
`units`	*char* Unit for time measurement. For example: "seconds", "minutes" or "days". See `duration` for more details.
`t_id`	*character* or *numeric*. Column name in `tracks` that identifies the separate tracks.

Details

This function can be used to compress huge amounts of data to speed up the calculations.

The average for the columns time, lon and lat is calculated. See `geomean` for details. Other columns will be preserved but only the first row of every interval is considered.

Value

data frame

`lon`	*num* averaged longitude
`lat`	*num* averaged latitude
`time`	*POSIXct* averaged time
`other`	Preserved columns from input

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

geomean, average_duplicates, select_test_sample

Examples

```
data(psyo_geomean)
average_coordinates(psyo_geomean,30,"seconds")
```

---

average_duplicates *Correct coordinates with the same time*

Description

Correct coordinates with the same time by replacing with one coordinate with a geomean position.

Usage

```
average_duplicates(tracks, t_id = "id")
```

Arguments

tracks	psyo. Data frame with tracks.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.

Value

psyo

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

geomean, average_coordinates

Examples

```
# Get data
data(psyo)
psyo <- psyo[c(1,15),]
psyo[2, c("time","id")] <- psyo[1, c("time","id")]

# Plot coordinates
map <- plot_map(psyo)
plot <- map + ggplot2::geom_point(data = psyo, size = 5)
plot <- plot_line(psyo[,c("lon", "lat")], plot = plot)
plot

# Calculate mean position
psyo <- average_duplicates(psyo)

# Plot mean position
plot + ggplot2::geom_point(data = psyo, size = 5)
```

---

des_duplicates *Count duplicates within each track*

Description

Count duplicates within each track

Usage

```
des_duplicates(
  tracks, cduplicated, cgaps = "", cname = "duplicates", drop = TRUE,
  t_id = "id", des_df = "")
```

Arguments

tracks	psyo. Data frame with tracks.
cduplicated	character. Column name of tracks that contains the variable for the calculation.
cgaps	character. Column name of tracks that marks gaps with TRUE.
cname	character. Column name of the returned calculation result.
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
des_df	data frame. Function results will be merge with this data frame.

Value

Data frame

t_id	unique id of the track
cname	calculated result of the track

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

des_summary, des_first, des_last, des_length, des_max, des_min, des_mean, des_sd, des_sum

Examples

```
# Get data
data(psyo_rounds2)
tracks <- psyo_rounds2

# Calculations
descriptive <- des_duplicates(tracks, "time")
```

---

des_first *Get first value within each track*

Description

Get first value within each track

Usage

```
des_first(
  tracks, ctarget, cgaps = "", cname = "first", drop = TRUE, t_id = "id",
  des_df = ""
)
```

Arguments

tracks	psyo. Data frame with tracks.
ctarget	character. Column name of tracks that contains the variable for the calculation.
cgaps	character. Column name of tracks that marks gaps with TRUE.
cname	character. Column name of the returned calculation result.
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
des_df	data frame. Function results will be merge with this data frame.

Value

Data frame

t_id	unique id of the track
cname	calculated result of the track

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

des_summary, des_duplicates, des_last, des_length, des_max, des_min, des_mean, des_sd, des_sum

Examples

```
# Print title
cat("\nTesting des_first()\n")

# Calculations
data(psyo_rounds2)
first <- des_first(psyo_rounds2, "p_id")
```

`des_last` *Get last value within each track*

Description

Get last value within each track

Usage

```
des_last(
  tracks, ctarget, cgaps = "", cname = "last", drop = TRUE, t_id = "id",
  des_df = ""
)
```

Arguments

`tracks`	psyo. Data frame with tracks.
`ctarget`	`character`. Column name of `tracks` that contains the variable for the calculation.
`cgaps`	`character`. Column name of `tracks` that marks gaps with `TRUE`.
`cname`	`character`. Column name of the returned calculation result.
`drop`	`logical`. If `TRUE` and only one observation is returned drop the `data frame` and collapse the return value to a `vector`.
`t_id`	*character* or *numeric*. Column name in `tracks` that identifies the separate tracks.
`des_df`	`data frame`. Function results will be merge with this data frame.

Value

Data frame

`t_id`	unique id of the track
`cname`	calculated result of the track

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

des_summary, des_duplicates, des_first, des_length, des_max, des_min, des_mean, des_sd, des_sum

Examples

```
data(psyo_rounds2)
last <- des_last(psyo_rounds2, "p_id")
```

---

des_length *Get the number of coordinates within each track*

Description

Get the number of coordinates within each track

Usage

```
des_length(
  tracks, cgaps = "", cname = "length", drop = TRUE, t_id = "id", des_df = ""
)
```

Arguments

tracks	psyo. Data frame with tracks.
cgaps	character. Column name of tracks that marks gaps with TRUE.
cname	character. Column name of the returned calculation result.
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
des_df	data frame. Function results will be merge with this data frame.

Value

Data frame

t_id	unique id of the track
cname	calculated result of the track

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

des_summary, des_duplicates, des_first, des_last, des_max, des_min, des_mean, des_sd, des_sum

Examples

```
# Get data
data(psyo_rounds2)
tracks <- psyo_rounds2

# Calculations
descriptive <- des_length(tracks)
```

---

des_max *Get the highest value within each track*

Description

Get the highest value within each track

Usage

```
des_max(
  tracks, ctarget, cgaps = "", cname = "max", drop = TRUE, t_id = "id",
  des_df = ""
)
```

Arguments

tracks	psyo. Data frame with tracks.
ctarget	character. Column name of tracks that contains the variable for the calculation.
cgaps	character. Column name of tracks that marks gaps with TRUE.
cname	character. Column name of the returned calculation result.
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
des_df	data frame. Function results will be merge with this data frame.

Value

Data frame

t_id	unique id of the track
cname	calculated result of the track

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

des_summary, des_duplicates, des_first, des_last, des_length, des_min, des_mean, des_sd, des_sum

Examples

```
data(psyo_rounds2)
last_time <- des_max(psyo_rounds2,"time")
```

des_mean *Calculate normal and weighted means while excluding gaps in data*

Description

Calculate normal and weighted means while excluding gaps in data

Usage

```
des_mean(
  tracks, ctarget, cweight = "", cgaps = "", cname = "mean", drop = TRUE,
  t_id = "id", des_df = ""
)
```

Arguments

tracks	psyo. Data frame with tracks.
ctarget	character. Column name of tracks that contains the variable for the calculation.
cweight	character. Column name of tracks that contains the weight for the calculation.
cgaps	character. Column name of tracks that marks gaps with TRUE.
cname	character. Column name of the returned calculation result.
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
des_df	data frame. Function results will be merge with this data frame.

Value

Data frame

id	id of the track
mean	calculated mean of the track

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

des_summary, des_duplicates, des_first, des_last, des_length, des_max, des_min, des_sd, des_sum

Examples

```
# Get example data
data(psyo_rounds2)
psyo_rounds2 <- psyo_rounds2[ c(1:5,11:15) ,]

# clean-up data
psyo_rounds2 <- average_duplicates(psyo_rounds2)

# Add gap segments
psyo_rounds2 <- t_time_difference(psyo_rounds2)
psyo_rounds2 <- mark_time_gaps(psyo_rounds2)

# Add speed
psyo_rounds2 <- t_speed(psyo_rounds2)

# Calculate different means
normal <- des_mean(psyo_rounds2, "speed", cname = "normal")

weighted <- des_mean(
  psyo_rounds2, "speed", cweight = "time_difference", cname = "weighted"
)

segmented <- des_mean(
  psyo_rounds2, "speed", cgaps= "time_gap", cname = "segmented"
)

segmented_weighted <- des_mean(
  psyo_rounds2,
  "speed",
  cweight = "time_difference",
  cgaps = "time_gap",
  cname = "segmented_weighted"
)
```

---

des_min *Get the lowest value within each track*

Description

Get the lowest value within each track

Usage

```
des_min(
  tracks, ctarget, cgaps = "", cname = "min", drop = TRUE, t_id = "id",
  des_df = ""
)
```

Arguments

tracks	psyo. Data frame with tracks.
ctarget	character. Column name of tracks that contains the variable for the calculation.
cgaps	character. Column name of tracks that marks gaps with TRUE.
cname	character. Column name of the returned calculation result.
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
des_df	data frame. Function results will be merge with this data frame.

Value

Data frame

t_id	unique id of the track
cname	calculated result of the track

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

des_summary, des_duplicates, des_first, des_last, des_length, des_max, des_mean, des_sd, des_sum

Examples

```
data(psyo_rounds2)
first_time <- des_min(psyo_rounds2,"time")
```

---

des_sd *Calculate normal and weighted sds while excluding gaps in data*

Description

Calculate normal and weighted sds while excluding gaps in data

Usage

```
des_sd(
  tracks, ctarget, cweight = "", cgaps = "", cname = "sd", drop = TRUE, t_id = "id",
  des_df = ""
)
```

Arguments

tracks	psyo. Data frame with tracks.
ctarget	character. Column name of tracks that contains the variable for the calculation.
cweight	character. Column name of tracks that contains the weight for the calculation.
cgaps	character. Column name of tracks that marks gaps with TRUE.
cname	character. Column name of the returned calculation result.
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
des_df	data frame. Function results will be merge with this data frame.

Value

Data frame

id	id of the track
sd	calculated sd of the track

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

des_summary, des_duplicates, des_first, des_last, des_length, des_max, des_min, des_mean, des_sum

Examples

```
# Get example data
data(psyo_rounds2)
psyo_rounds2 <- psyo_rounds2[ c(1:5,11:15) ,]

# clean-up data
psyo_rounds2 <- average_duplicates(psyo_rounds2)

# Add gap segments
psyo_rounds2 <- t_time_difference(psyo_rounds2)
psyo_rounds2 <- mark_time_gaps(psyo_rounds2)

# Add speed
psyo_rounds2 <- t_speed(psyo_rounds2)

# Calculate different sds
normal <- des_sd(psyo_rounds2, "speed", cname = "normal")

weighted <- des_sd(
  psyo_rounds2, "speed", cweight = "time_difference", cname = "weighted"
)

segmented <- des_sd(
  psyo_rounds2, "speed", cgaps= "time_gap", cname = "segmented"
)

segmented_weighted <- des_sd(
  psyo_rounds2,
  "speed",
  cweight = "time_difference",
```

```
  cgaps = "time_gap",
  cname = "segmented_weighted"
)
```

`des_sum` *Calculates sum for each track*

Description

Calculates sum for each track

Usage

```
des_sum(
  tracks, ctarget, cgaps = "", cname = "sum", drop = TRUE, t_id = "id",
  des_df = ""
)
```

Arguments

`tracks`	`psyo`. Data frame with tracks.
`ctarget`	`character`. Column name of `tracks` that contains the variable for the calculation.
`cgaps`	`character`. Column name of `tracks` that marks gaps with `TRUE`.
`cname`	`character`. Column name of the returned calculation result.
`drop`	`logical`. If `TRUE` and only one observation is returned drop the `data frame` and collapse the return value to a `vector`.
`t_id`	*character* or *numeric*. Column name in `tracks` that identifies the separate tracks.
`des_df`	`data frame`. Function results will be merge with this data frame.

Value

Data frame

`t_id`	unique id of the track
`cname`	calculated result of the track

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

des_summary, des_duplicates, des_first, des_last, des_length, des_max, des_min, des_mean, des_sd

Examples

```
# Get data
data(psyo_rounds2)
tracks <- psyo_rounds2

# Calculations
tracks <- t_distance(tracks)
summary <- des_sum(tracks, "distances_in_m")
```

---

des_summary — *Creates a data frame with a summary of descriptive information for each track*

Description

Creates a data frame with a summary of descriptive information for each track

Usage

```
des_summary(tracks, cweight = "auto", cgaps = "", t_id = "id", des_df = "")
```

Arguments

tracks	psyo. Data frame with tracks.
cweight	character. Column name of tracks that contains the weight for the calculation.
cgaps	character. Column name of tracks that marks gaps with TRUE.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
des_df	data frame. Function results will be merge with this data frame.

Value

Data frame

`id`	id of the track
`begin_time`	begin time of the track
`end_time`	end time of the track
`duration_in_mins`	duration in minutes
`tracker_interval_in_seconds`	tracker interval in seconds
`coordinates`	number of coordinates without gaps
`coordinates_gapped`	number of coordinates with gaps
`coordinates_all`	number of all coordinates
`time_duplicates`	coordinates with the same time
`time_gap`	coordinates that are marked as gaps because they differ to much from the tracker interval
`speed_gap`	coordinates that are mark's as gaps because they have a higher speed than expected
`sum_km`	total number of kilometres
`mean_kmh`	average speed in kmh
`mean_kmh_no_stop`	average speed in kmh without coordinates with 0 speed
`movement_time_sum`	total time without speed higher than 0
`no_movement_time_sum`	total time with speed equal to 0
`move_by_no_move_ratio`	ratio between movement and no movement time
`time_good_sum`	data with good data
`time_missing_sum`	data that is missing
`time_gap_sum`	data that is excluded by gaps

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

des_duplicates, des_first, des_last, des_length, des_max, des_min, des_mean, des_sd, des_sum

Examples

```
data(psyo_rounds2)
tracks <- psyo_rounds2
descriptive <- des_summary(tracks)
```

---

dir_add_csv *Bind CSV data to data frame*

Description

Read CSV file in directory and bind data to a data frame.

Usage

```
dir_add_csv(
  tracks, dir, merge_by = "id", stringsAsFactors = default.stringsAsFactors()
)
```

Arguments

tracks	psyo. Data frame with tracks.
dir	The path to the CSV file.
merge_by	The column in the *data_frame* and the CSV file that is used to merge the data.
stringsAsFactors	*logical*. Should character vectors be converted to factors.

Value

Data frame

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere

- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Note

The function ignores multiple segments in the GPX file. If you want to find the gaps between the segments you could use the functions `t_time_difference` and `mark_time_gaps`.

Further does the function sort the data by time.

Author(s)

Benjamin Ziepert

See Also

`dir_get_gpx`

Examples

```
data(psyo)
csv_dir <- system.file("extdata", "ids.csv", package = "psyosphere")
psyo <- dir_add_csv(psyo, csv_dir)
```

---

dir_get_gpx *Read GPX files from directory into data frame.*

Description

Read GPX files from directory into data frame.

Usage

```
dir_get_gpx(dir, tz = "", stringsAsFactors = default.stringsAsFactors())
```

Arguments

`dir`	*character*. The directory of the GPX files relative to the working directory.
`tz`	*character*. The time zone for the time stamp of the coordinates.
`stringsAsFactors`	*logical*. Should character vectors be converted to factors.

Details

To avoid problems with the time zone it is advisable to set it. Otherwise the system time is used and this can result in different times on different computers. See timezones.

Value

Data frame as psyo.

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert Dr. Elze Ufkes

See Also

dir_add_csv, timezones

Examples

```
gpx_dir <- system.file("extdata", package="psyosphere")
psyo_rounds <- dir_get_gpx(gpx_dir, tz="MET")
```

---

distance_line *Add shortest distance to a line*

Description

Add shortest distance to a line

Usage

```
distance_line(
  tracks, line, bind = TRUE, drop = TRUE, cname = "distances_to_line"
)
```

Arguments

tracks	psyo. Data frame with tracks.
line	*list*. A list with the column lon (*numeric*) and lat (*numeric*).
bind	*logical*. Return the distance as list (FALSE) or add it to tracks (TRUE).
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
cname	character. Column name of the returned calculation result.

Value

psyo or list. Distance in meter.

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

distance_peers, distance_point, distance_psyo

Examples

```
# Get data
data(psyo_rounds)

# Set start and finish
lon <- c(6.849975, 6.849627, 6.850001, 6.850350, 6.849975)
lat <- c(52.241745, 52.241100, 52.241004, 52.241649, 52.241745)
polygon_start <- data.frame(lon, lat)
remove(lon, lat)

lon <- c(6.851810, 6.851000, 6.851489, 6.852296, 6.851810)
lat <- c(52.241800, 52.240300, 52.240163, 52.241657, 52.241794)
polygon_finish <- data.frame(lon, lat)
remove(lon, lat)
```

```
# Select between start and finish
psyo_rounds <- select_between_polygons(
  psyo_rounds, polygon_start, polygon_finish
)

# Finish line
finish <- data.frame(lon = c(6.851810,6.851000), lat = c(52.241800,52.240300))

# Plot tracks, selection polygons and finish line
plot <- plot_tracks(psyo_rounds, t_id = "")
plot <- plot_polygon(polygon_start, plot = plot)
plot <- plot_polygon(polygon_finish, plot = plot)
plot_line(finish, plot = plot)

# Add distance to line to dataframe
psyo_rounds <- distance_line(psyo_rounds,finish, TRUE)
```

`distance_peers` *Add distance to peers*

Description

Add distance in meters to peers within the same selection

Usage

```
distance_peers(
  tracks, cpeer = "", single = FALSE, average = TRUE, cname = "average_dis",
  bind = TRUE, drop = TRUE, t_id = "id"
)
```

Arguments

`tracks`	psyo. Data frame with tracks.
`cpeer`	*character*. Column that identifies peers.
`single`	*logical*. Append distances to each participant independently.
`average`	*logical*. Append average distances to peers.
`cname`	`character`. Column name for the average distance.
`bind`	*logical*. Return the distance as list (`FALSE`) or add it to `tracks` (`TRUE`).
`drop`	`logical`. If `TRUE` and only one observation is returned drop the `data frame` and collapse the return value to a `vector`.
`t_id`	*character* or *numeric*. Column name in `tracks` that identifies the peers that get compared with each other

Details

The function also determines the distance if the time stamps of the coordinates don't match. Please look at the example section for the details.

Only the distance to peers is determined. Therefore, distance to one-self is NA.

Value

psyo

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Note

Be aware: this function can take a lot time. You can use average_coordinates first to test your script with a small sample.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

distance_line, distance_point, distance_psyo

Examples

```
# Simple example ------------------------------------------------------------
data(psyo_rounds2)
time <- as.POSIXct("2015-09-03 13:51:07")
tracks <- psyo_rounds2[ psyo_rounds2[,"time"] == time, ]
tracks <- distance_peers(tracks)

# Example with intersect position -------------------------------------------

#  p1 --------x-------> p2
#             ^
#             |
#             |
#            p3
```

```
# We have two tracks. From track 1 we have p1 and p2 at a certain time. From
# tracks 2 we have point 3. Point 3 has a time between p1 and p2. We want to
# know what the distance from point 3 is to track 1 at the time of point 3. For
# this we need determine point x.

# Create the points
p1 <- data.frame(time = "2016-01-01 01:00:00", lon = 0, lat = 0, id = "1")
p2 <- data.frame(time = "2016-01-01 03:00:00", lon = 2, lat = 0, id = "1")
p3 <- data.frame(time = "2016-01-01 02:00:00", lon = 1, lat = 1, id = "2")
p1$time <- as.POSIXct(p1$time)
p2$time <- as.POSIXct(p2$time)
p3$time <- as.POSIXct(p3$time)

# Combine into a track
tracks <- rbind(p1, p2, p3)

# Get point x for illustration
x <- psyosphere:::timed_destination_point_private(p1, p2, p3, "id")

# Plot points as track for illustration
plot <- plot_tracks(tracks)
plot

# Add x to plot for illustration
plot_tracks(x, plot = plot)

# Get distances
tracks <- distance_peers(tracks)
```

`distance_point` *Add the distances to a point from each coordinate*

Description

Add the distances to a point from each coordinate

Usage

```
distance_point(
  tracks, point, bind = TRUE, drop = TRUE, cname = "dis_to_point_in_m"
)
```

Arguments

`tracks`	psyo. Data frame with tracks.
`point`	*list*. A list, matrix or data.frame with the columns lon (*numeric*) and lat (*numeric*) and one row.
`bind`	*logical*. Return the distance as list (`FALSE`) or add it to `tracks` (`TRUE`).

drop	logical. If TRUE drop the data frame and return value as vector or list.
cname	character. Column name of the returned calculation result.

Value

psyo

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

distance_line,distance_peers, distance_psyo

Examples

```
data(psyo)

# Un-named list
point <- c(4.936197, 52.314701)
distance_point(psyo[1,], point, bind = FALSE)

# Named list
point <- c(lat = 52.314701, lon = 4.936197)
distance_point(psyo[1,], point, bind = FALSE)

# Multiple distance to point
coordinates <- distance_point(psyo, point)
```

`distance_psyo` *Add distance to another track in psyo format*

Description

Add distance to another track in psyo format.

Usage

```
distance_psyo(
  tracks1, tracks2, t_id1 = "id", t_id2 = "id", bind = TRUE, drop = TRUE
)
```

Arguments

tracks1	psyo. Data frame with tracks.
tracks2	psyo. Data frame with tracks.
t_id1	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
t_id2	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
bind	*logical*. Return the distance as list (FALSE) or add it to tracks (TRUE).
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.

Value

psyo

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Note

Be aware: this function can take a lot time. You can use average_coordinates first to test your script with a small sample.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

distance_line,distance_peers, distance_point

Examples

```
data(psyo_rounds2)
psyo_distance <- select_test_sample(psyo_rounds2, 5)
psyo_distance <- distance_psyo(psyo_rounds2, psyo_rounds2)
```

---

distance_to_direct_line

Add deviation from shortest route from begin of track to a line

---

Description

Add deviation from shortest route from begin of track to a line

Usage

```
distance_to_direct_line(
  tracks, line, bind = TRUE, drop = TRUE, cname = "distance_to_direct_line",
  t_id = "id"
)
```

Arguments

tracks	psyo. Data frame with tracks.
line	*list*. A list with the column lon (*numeric*) and lat (*numeric*).
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
bind	*logical*. Return the distance as list (FALSE) or add it to tracks (TRUE).
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
cname	character. Column name of the returned calculation result.

Value

psyo. Distance in meter.

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

distance_line

Examples

```
# Get data
data(psyo_rounds2)

# Create finish line
finish <- data.frame(lon = c(6.851810,6.851000), lat = c(52.241800,52.240300))

# Plot tracks and finish
plot <- plot_tracks(psyo_rounds2, t_id = "")
plot_line(finish, plot = plot)

# Get deviation from shortest rout from begin to finish
psyo_rounds2 <- distance_to_direct_line(psyo_rounds2, finish)
```

---

export_gpx *Export tracks as gpx files*

Description

Export tracks as gpx files.

Usage

```
export_gpx(tracks, t_id = "id")
```

Arguments

tracks	psyo. Data frame with tracks. The columns track_name and file_name have to be defined.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.

Details

track_name is the name of the track.

file_name is the file name of the gpx file.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

codeexport_kml

Examples

```
## Not run:

# Get tracks

data(psyo)

# Add columns

dir <- tempdir()
psyo[,"track_name"] <- psyo[,"id"]
psyo[,"file_name"] <- file.path(dir, psyo[,"track_name"])

# Export files

export_gpx(tracks = psyo)

## End(Not run)
```

---

export_kml *Export tracks as kml files*

Description

Export tracks as kml files.

Usage

```
export_kml(tracks, t_id = "id")
```

Arguments

tracks	psyo. Data frame with tracks. The columns track_name, file_name, track_color have to be defined.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.

Details

track_name is the name of the track.

file_name is the file name of the kml file.

color is the color of the track. For instance "ffaa00bb" sets the transparency to bb in exdecimal or 73%, sets blue to 00, sets green to aa, and sets red to ff. See also aes_colour_fill_alpha.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

codeexport_gpx

Examples

```
## Not run:

# Get tracks

data(psyo)

# Add columns

dir <- tempdir()
psyo[,"track_color"] <- "bb00aaff"
psyo[,"track_name"] <- psyo[,"id"]
psyo[,"file_name"] <- file.path(dir, paste0(psyo[,"track_name"], ".kml"))

# Export files

export_kml(psyo)

## End(Not run)
```

`mark_gap_segments` *Adding column with segment names between gaps*

Description

Adding column with segment names between gaps

Usage

```
mark_gap_segments(
  tracks, cgaps, bind = TRUE, drop = TRUE, cname = "gap_segments", t_id = "id"
)
```

Arguments

`tracks`	psyo. Data frame with tracks.
`cgaps`	`character`. Column name of `tracks` that marks gaps with `TRUE`.
`bind`	*logical*. Return the distance as list (`FALSE`) or add it to `tracks` (`TRUE`).
`drop`	`logical`. If `TRUE` and only one observation is returned drop the `data frame` and collapse the return value to a `vector`.
`cname`	`character`. Column name of new column in `tracks` that contains the segment names.
`t_id`	*character* or *numeric*. Column name in `tracks` that identifies the separate tracks.

Value

psyo

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

mark_speed_gaps, mark_time_gaps

Examples

```
data(psyo_rounds2)
psyo_rounds2 <- average_duplicates(psyo_rounds2)
psyo_rounds2 <- t_time_difference(psyo_rounds2)

psyo_rounds2 <- mark_time_gaps(psyo_rounds2)
psyo_rounds2 <- mark_gap_segments(psyo_rounds2,"time_gap")
```

---

mark_inside_polygon *Mark coordinates within a polygon*

Description

A column will be created that indicates whether a coordinate lies within a polygon or not. See also point.in.polygon.

Usage

```
mark_inside_polygon(
  tracks, polygon, bind = TRUE, drop = TRUE, cname = "in_polygon"
)
```

Arguments

tracks	psyo. Data frame with tracks.
polygon	*list*. A list with the column lon (*numeric*) and lat (*numeric*).
bind	*logical*. Return the distance as list (FALSE) or add it to tracks (TRUE).
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
cname	character. The name of the new column.

Value

psyo

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere

- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

point.in.polygon

Examples

```
# Create polygon
lon <- c(4.92, 4.93, 4.93, 4.92, 4.92)
lat <- c(52.311, 52.311, 52.308, 52.308, 52.311)
poly <- data.frame(lon, lat)
remove(lon, lat)

# Get data
data(psyo)

# Plot polygon and data
plot <- plot_tracks(psyo, t_id = "")
plot <- plot_polygon(poly, plot = plot)
plot

# Mark coordinates within plygon
psyo <- mark_inside_polygon(psyo, poly)

# Plot inside polygon in different color
in_poly <- psyo[ psyo[,"in_polygon"] != 0,]
in_poly[,"dot_color"] <- "red"
plot_tracks(in_poly, plot = plot)
```

---

mark_speed_gaps *Mark speeds that exceed a certain speed limit as gaps*

Description

Mark speeds that exceed a certain speed limit as gaps

Usage

```
mark_speed_gaps(
  tracks, speed_limit, cspeed = "speed", bind = TRUE, drop = TRUE,
  cname = "speed_gap", t_id = "id"
)
```

Arguments

tracks	psyo. Data frame with tracks.
speed_limit	numeric. Values in column cspeed that are equal or higher than this value will be marked as gaps in column cgaps as TRUE.
cspeed	character. Column name of tracks that contains the speed as numeric values.
bind	*logical*. Return the distance as list (FALSE) or add it to tracks (TRUE).
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
cname	character. Column name of tracks that marks gaps with TRUE. If the column does not exist it will be created.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.

Value

psyo

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

mark_gap_segments, mark_time_gaps, select_gaps

Examples

```
# Get example data
data(psyo_rounds2)
speedt <- psyo_rounds2[ c(1:5,11:15) ,]
remove(psyo_rounds2)

# clean-up data
speedt <- average_duplicates(speedt)
speedt <- t_time_difference(speedt)
speedt <- mark_time_gaps(speedt)
```

```
# Add speed
speedt <- t_speed(speedt)

# Advanced mean speed without speed gap
des_mean(speedt, "speed", "time_difference", "time_gap")

# Mark speed gap
speedt <- mark_speed_gaps(speedt, 8)

# Advanced mean speed with speed gap
des_mean(speedt, "speed", "time_difference", c("time_gap","speed_gap"))
```

`mark_time_gaps` *Mark segments between data gaps*

Description

Mark segments between data gaps

Usage

```
mark_time_gaps(
  tracks, interval = 0, factor = 3, ctime_difference = "time_difference",
  bind = TRUE, drop = TRUE, cname = "time_gap", t_id = "id"
)
```

Arguments

`tracks`	psyo. Data frame with tracks.
`interval`	`numeric`. Recording interval of the GPS tracker in seconds. Use 0 to automatically determine the interval. For this the most frequent interval is used.
`factor`	`numeric`. Multiplier to determine gaps. If a time difference between coordinates is bigger than `tracker_interval * factor` it is marked as gap.
`ctime_difference`	`character`. Column name of `tracks` that contains the time difference as `numeric` values.
`bind`	*logical*. Return the distance as list (`FALSE`) or add it to `tracks` (`TRUE`).
`drop`	`logical`. If `TRUE` and only one observation is returned drop the `data frame` and collapse the return value to a `vector`.
`cname`	`character`. Column name of `tracks` that marks gaps with `TRUE`. If the column does not exist it will be created.
`t_id`	*character* or *numeric*. Column name in `tracks` that identifies the separate tracks.

Value

psyo

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

mark_gap_segments, mark_speed_gaps, select_gaps

Examples

```
# Get example data
data(psyo_rounds2)

# clean-up data
psyo_rounds2 <- average_duplicates(psyo_rounds2)

# Add gap segments
psyo_rounds2 <- t_time_difference(psyo_rounds2)
psyo_rounds2 <- mark_time_gaps(psyo_rounds2)

# Check result
psyo_rounds2 <- psyo_rounds2[ which(psyo_rounds2[,"time_gap"]) ,]
```

---

plot_line *Plot line on map*

Description

Adds a line to an existing ggmap object.

Usage

```
plot_line(
  line, colour = "", size = 1, plot = "", zoom = -1
)
```

Arguments

line	*list*. A list with the column lon (*numeric*) and lat (*numeric*).
colour	*character*. Colour of the line.
size	*numeric*. Thickness of the line.
plot	ggmap. An existing map / plot where the tracks are added. If no plot is provided a Google map will be downloaded.
zoom	*numeric*. Zoom factor of the map. See get_googlemap.

Value

A ggmap object.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

plot_map, plot_tracks, plot_polygon

Examples

```
finish <- data.frame(lon = c(6.851810,6.851000), lat = c(52.241800,52.240300))
plot_line(finish)
```

---

plot_map *Get a Google map*

Description

Get a Google map that fits to the tracks that are provided in psyo format.

Usage

```
plot_map(
  tracks, zoom = -1, maptype = "terrain", extent = "panel"
)
```

Arguments

tracks	psyo. Data frame with tracks.
zoom	*numeric*. Zoom level. -1 for auto zoom or from 3 (continent) to 21 (building). See also get_map.
maptype	character. See get_googlemap.
extent	character. See ggmap.

Value

A ggmap object.

Author(s)

Benjamin Ziepert Dr. Elze Ufkes

See Also

plot_line, plot_tracks, plot_polygon

Examples

```
data(psyo)
plot_map(psyo)
```

---

plot_polygon *Plot polygon on map*

Description

Adds a polygon to an existing ggmap object.

Usage

```
plot_polygon(polygon, colour = "blue", plot = "", zoom = -1)
```

Arguments

polygon	*list*. A list with the column lon (*numeric*) and lat (*numeric*).
colour	*character*. Colour of the line.
plot	ggmap. An existing map / plot where the tracks are added. If no plot is provided a Google map will be downloaded.
zoom	*numeric*. Zoom factor of the map. See get_googlemap.

Value

A ggmap object.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

plot_map, plot_line, plot_tracks

Examples

```
lon <- c(6.849975, 6.849627, 6.850001, 6.850350, 6.849975)
lat <- c(52.241745, 52.241100, 52.241004, 52.241649, 52.241745)
polygon <- data.frame(lon, lat)
remove(lon, lat)

plot <- plot_polygon(polygon)
plot
```

---

plot_tracks *Plot tracks on a map*

Description

Plot tracks in the psyo format on map.

Usage

```
plot_tracks(
  tracks, single = FALSE, line = TRUE, dots = TRUE, plot = "", zoom = -1,
  save_dir = "", cgaps = "", t_id = "id"
)
```

Arguments

tracks	psyo. Tracks that will be plotted.
single	*logical*. Display the plotted maps. Between each plot the script will wait for user confirmation until the next plot will be displayed.
line	logical. Whether to draw a line between the coordinates.
dots	logical. Whether to display the coordinates.
plot	ggmap. An existing map / plot where the tracks are added. If no plot is provided a Google map will be downloaded.
zoom	*numeric*. Zoom factor of the map. See get_googlemap.
save_dir	*character*. Save the plots in a directory. If empty no plots will be saved.
cgaps	character. Column name of tracks that marks gaps with TRUE.
t_id	*character* or *integer*. The column name that identifies the different tracks. Use t_id = "" to process all tracks.

Value

A ggmap object.

Author(s)

Benjamin Ziepert

See Also

plot_map, plot_line, plot_polygon

Examples

```
# Plot tracks
data(psyo_rounds2)
plot <- plot_tracks(psyo_rounds2)
plot

# Get zoom level
plot$zoom
```

---

psyo *Example how data should be formatted.*

Description

Example how data in psyosphere should be formatted.

Usage

```
data(psyo)
```

Format

A data frame with 15 observations on the following 5 variables.

`id` *A character or numeric vector.* Id for each unique track.

`p_id` *A character or numeric vector*. Unique by time sorted ID for every coordinate within a track.

`time` *A POSIXct.* Date and time of the coordinate.

`lon` *A numeric vector.* Longitude of a coordinate in degree.

`lat` *A numeric vector.* Latitude of a coordinate in degree.

Details

The example data contain three different tracks ("01.gpx", "02.gpx", "03.gpx") and 5 observations / coordinates for each track.

The data frame should at least contain the variables mentioned above. Additional columns can be added. Where possible the package will preserve these columns.

See Also

val_psyo

Examples

```
# Simple example ------------------------------------------------------------

data(psyo)
print(psyo)

# Result:
#
#     id p_id                time     lon      lat
# 01.gpx    0 2016-06-19 12:37:53 4.93078 52.31003
# 01.gpx    1 2016-06-19 12:37:58 4.93038 52.30985
# 01.gpx    2 2016-06-19 12:38:08 4.92958 52.30953
# 01.gpx    3 2016-06-19 12:38:18 4.92803 52.30883
# 01.gpx    4 2016-06-19 12:38:28 4.92652 52.30800
# 02.gpx    0 2016-06-19 11:28:25 4.93580 52.31450
# 02.gpx    1 2016-06-19 11:28:38 4.93580 52.31450
# 02.gpx    2 2016-06-19 11:32:03 4.93580 52.31450
# 02.gpx    3 2016-06-19 11:32:13 4.93580 52.31450
# 02.gpx    4 2016-06-19 11:32:28 4.93580 52.31450
# 03.gpx    0 2016-06-20 10:17:08 5.00828 52.35005
# 03.gpx    1 2016-06-20 10:17:18 5.00843 52.35010
# 03.gpx    2 2016-06-20 10:17:28 5.00847 52.35028
# 03.gpx    3 2016-06-20 10:17:43 5.00847 52.35028
# 03.gpx    4 2016-06-20 10:17:53 5.00847 52.35028

# How to create a geodata data frame from scratch ---------------------------

id <- c("01.gpx", "01.gpx", "01.gpx", "01.gpx", "01.gpx",
        "02.gpx", "02.gpx", "02.gpx", "02.gpx", "02.gpx",
        "03.gpx", "03.gpx", "03.gpx", "03.gpx", "03.gpx")

p_id <- c(0, 1, 2, 3, 4, 0, 1, 2, 3, 4, 0, 1, 2, 3, 4)

time <- c("2016-06-19 12:37:53", "2016-06-19 12:37:58", "2016-06-19 12:38:08",
          "2016-06-19 12:38:18", "2016-06-19 12:38:28",
          "2016-06-19 11:28:25", "2016-06-19 11:28:38", "2016-06-19 11:32:03",
          "2016-06-19 11:32:13", "2016-06-19 11:32:28",
          "2016-06-20 10:17:08", "2016-06-20 10:17:18", "2016-06-20 10:17:28",
          "2016-06-20 10:17:43", "2016-06-20 10:17:53")
```

```
lon <- c(4.93078, 4.93038, 4.92958, 4.92803, 4.92652,
         4.93580, 4.93580, 4.93580, 4.93580, 4.93580,
         5.00828, 5.00843, 5.00847, 5.00847, 5.00847)

lat <- c(52.31003, 52.30985, 52.30953, 52.30883, 52.30800,
         52.31450, 52.31450, 52.31450, 52.31450, 52.31450,
         52.35005, 52.35010, 52.35028, 52.35028, 52.35028)

psyo <- data.frame(id, p_id, time, lon, lat)
psyo <- as.POSIXct(psyo$time)

remove(id, p_id, time, lon, lat)

print(psyo)
```

`psyosphere` *psyosphere details*

Description

'psyosphere' can be used to analyse location data (latitude, longitude, elevation). Based on spherical trigonometry, variables such as speed, bearing, and distances can be calculated from moment to moment, depending on the sampling frequency of the equipment used, and independent of scale. Additionally, the package can plot tracks, coordinates, and shapes on maps, and sub-tracks can be selected with point-in-polygon or other techniques. The package is optimized to support behavioural science experiments with multiple tracks. It can detect and clean up errors in the data, and resulting data can be exported to be analysed in statistical software or geographic information systems (GIS).

Details

'psyosphere' uses geodata data frames to store tracks. For the format, you can read psyo.

The package handles latitude and longitude and mostly ignores elevation.

For the first steps please read about_analysing_tips

Credit

If you use 'psyosphere' for commercial use or research, please support us by include one off the following references:

- **MIT license:** "psyosphere" by B. Ziepert, E. G. Ufkes & P. W. de Vries from https://CRAN.R-project.org/package=psyosphere
- **APA**: Ziepert, B., Ufkes, E., & de Vries, P. W. (2018). psyosphere: Analyse GPS Data. Retrieved from https://CRAN.R-project.org/package=psyosphere
- **APA**: Vries, P., Ziepert, B., & Ufkes, E. (2016). "De psychologie van bewegingen GPS-technologie voor de analyse van natuurlijk gedrag." Tijdschrift voor Human Factors 2: 11-15.

Credits

For 'psyosphere' we made use of the following software:

- "R" by The R foundation from r-project.org / GNU-2
- "RStudio" by RStudio from rstudio.com / AGPL v3
- "ggmap" by D. Kahle & H. Wickham from cran.rstudio.com / GPL-2
- "rgdal" by R. Bivand, T. Keitt, B. Rowlingson, E. Pebesma, M. Sumner, R. Hijmans & E. Rouault from cran.rstudio.com / GPL-2
- "plyr" by H. Wickham from cran.rstudio.com / MIT
- "geosphere" by R. Hijmans, E. Williams & C. Vennes from cran.rstudio.com / GPL-3
- "ggplot2" by H. Wickham, W. Chang & RStudio from cran.rstudio.com / GPL-2
- "sp" by E. Pebesma, R. Bivand, B. Rowlingson, V. Gomez-Rubio, R. Hijmans, M. Sumner, D. MacQueen, J. Lemon & J. O'Brien from cran.rstudio.com / GPL-2
- "lubridate" by G. Grolemund, V. Spinu, H. Wickham, I. Lyttle, I. Constigan, J. Law, D. Mitarotonda, J. Larmarange, J. Boiser & C. Lee from cran.rstudio.com / GPL-2
- "RgoogleMaps" by M. Loecher from cran.rstudio.com / GPL
- "SDMTools" by J. VanDerWal, L. Falconi, S. Januchowski, L. Shoo & C. Storlie from cran.rstudio.com / GPL-3

Author(s)

Benjamin Ziepert. <feedback-psyosphere@analyse-gps.com>.

Dr. Elze G. Ufkes <elze.ufkes@utwente.nl>.

Dr. Ir. Peter W. de Vries <p.w.devries@utwente.nl>.

`psyo_geomean` *Example data to demonstrate the geomean function*

Description

Data set in the psyo format.

Usage

```
data(psyo_geomean)
```

Format

A data frame with 4 observations on the following 6 variables.

`id` *A character vector.* With 1 level: `"01.gpx"`

`p_id` *A numeric vector.* Unique by time sorted ID for every coordinate within a track.

`time` *A POSIXct.* Date and time of the coordinate.

`lon` *A numeric vector.* Longitude of a coordinate in degree.

`lat` *A numeric vector.* Latitude of a coordinate in degree.

`other` *A character vector.* With 4 levels `"L"`, `"e"`, `"t"`, and `"o"`

See Also

val_psyo

Examples

```
# Simple example ------------------------------------------------------------

# Get data frame and compress 4 coordinates
data("psyo_geomean")
compressed_tracks <- average_coordinates(psyo_geomean,
                                                                    30,
                                                                    "seconds")
```

---

psyo_rounds *GPS example of walking in circles*

Description

GPS example of walking in circles in psyo format.

Usage

```
data(psyo_rounds)
```

Format

A data frame with 2896 observations on the following 6 variables.

`id` *A character or numeric vector.* Id for each unique track.

`p_id` *A character or numeric vector*. Unique by time sorted ID for every coordinate within a track.

`time` *A POSIXct.* Date and time of the coordinate.

`lon` *A numeric vector.* Longitude of a coordinate in degree.

`lat` *A numeric vector.* Latitude of a coordinate in degree.

`ele` *A numeric vector.* Elevation of a coordinate in degree.

`tracker` a numeric vector

`team` a factor with levels `1` `11` `12` `13` `14` `15` `16` `17` `18` `2` `3` `4` `5` `6` `7` `8` `D1` `D2` `D3`

`ppn` a numeric vector

See Also

val_psyo

Examples

```
data(psyo_rounds)
plot_tracks(psyo_rounds, t_id = "")
```

psyo_rounds2 *GPS example with 3 selected rounds*

Description

GPS example with 3 selected rounds in psyo format.

Usage

```
data(psyo_rounds2)
```

Format

A data frame with 258 observations on the following 6 variables.

`track` a numeric vector

`id` *A character or numeric vector.* Id for each unique track.

`p_id` *A character or numeric vector.* Unique by time sorted ID for every coordinate within a track.

`time` *A POSIXct.* Date and time of the coordinate.

`lon` *A numeric vector.* Longitude of a coordinate in degree.

`lat` *A numeric vector.* Latitude of a coordinate in degree.

`ele` *A numeric vector.* Elevation of a coordinate in degree.

`tracker` a numeric vector

`team` a factor with levels `1` `11` `12` `13` `14` `15` `16` `17` `18` `2` `3` `4` `5` `6` `7` `8` `D1` `D2` `D3`

`ppn` a numeric vector

See Also

val_psyo

Examples

```
data(psyo_rounds2)
plot_tracks(psyo_rounds2, t_id = "")
```

psyo_rounds_map *Map for data psyo_rounds and psyo_rounds2*

Description

Google map as ggplot object

Usage

```
data(psyo_rounds_map)
```

Format

See ggplot

Examples

```
data(psyo_rounds_map)
plot(psyo_rounds_map)
```

select_between_polygons *Select tracks between two polygons.*

Description

Select tracks between a start and a finish polygon. Only the data between the polygons will remain. Data that is not between the start and finish polygon will be disregarded. If a track passes multiple times first the start and then the finish will be split up in rounds and new track id's will be created for each round.

Usage

```
select_between_polygons(
  tracks, poly1, poly2, t_id = "id", merge_id = TRUE
)
```

Arguments

tracks	psyo. Data frame with tracks.
poly1	*data frame*. A data frame with the columns *lon* (*numeric*) and *lat* (*numeric*). All coordinates will be selected that start after leaving this polygon and enter *polygon_finish*. The polygon should be closed, therefore the first and last coordinate must be the same. See also point.in.polygon.

poly2 — *data frame*. A data frame with the columns *lon* (*numeric*) and *lat* (*numeric*). All coordinates will be selected that start after leaving `polygon_start` and enter this polygon. The polygon should be closed, therefore the first and last coordinate must be the same. See also `point.in.polygon`.

t_id — *character* or *numeric*. Column name in `tracks` that identifies the separate tracks.

merge_id — *logical*. If `TRUE` append the round to the current track id column `t_id`. If `FALSE` create a separate column with the round number.

Details

The following image shows a track before selection.

The following image shows a track after selection with 3 rounds.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

Examples

```
# Create polygons
lon <- c(6.849975, 6.849627, 6.850001, 6.850350, 6.849975)
lat <- c(52.241745, 52.241100, 52.241004, 52.241649, 52.241745)
polygon_start <- data.frame(lon, lat)
remove(lon, lat)

lon <- c(6.851810, 6.851000, 6.851489, 6.852296, 6.851810)
lat <- c(52.241800, 52.240300, 52.240163, 52.241657, 52.241794)
polygon_finish <- data.frame(lon, lat)
remove(lon, lat)

# Get a track
data(psyo_rounds)

# Plot tracks
plot <- plot_tracks(psyo_rounds, zoom = 17, t_id = "")
plot

# Add start and finish polyon
plot <- plot_polygon(polygon_start, plot = plot)
plot_polygon(polygon_finish, plot = plot)

# Select data between polygon
psyo_rounds <- select_between_polygons(
  psyo_rounds, polygon_start, polygon_finish
```

```
)

# Plot the remaining data
plot <- plot_tracks(psyo_rounds, t_id = "")
plot <- plot_polygon(polygon_start, plot = plot)
plot_polygon(polygon_finish, plot = plot)
```

`select_gaps` *Select all coordinates with a gap*

Description

Select all coordinates with a gap

Usage

```
select_gaps(tracks, cgaps)
```

Arguments

`tracks`	`psyo`. Data frame with tracks.
`cgaps`	`character`. Column name of `tracks` that marks gaps with `TRUE`.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

`mark_time_gaps`, `mark_speed_gaps`, `select_without_gaps`

Examples

```
# Get data
data(psyo_rounds2)
tracks <- psyo_rounds2

# Calculations
tracks <- average_duplicates(tracks)
tracks <- t_time_difference(tracks)

tracks <- mark_time_gaps(tracks)
tracks <- select_gaps(tracks,"time_gap")
```

select_test_sample *Select a sample from each track*

Description

Select a sample from each track to test functions quicker.

Usage

```
select_test_sample(tracks, size = 3, t_id = "id")
```

Arguments

tracks	psyo. Data frame with tracks.
size	*numeric*. Remaining number of coordinates of each track in tracks
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

average_coordinates

Examples

```
data(psyo)
test_tracks <- select_test_sample(psyo)
```

select_without_gaps *Select all coordinates without gap*

Description

Select all coordinates without gap

Usage

```
select_without_gaps(tracks, cgaps)
```

Arguments

tracks	psyo. Data frame with tracks.
cgaps	character. Column name of tracks that marks gaps with TRUE.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

mark_time_gaps, mark_speed_gaps, select_gaps

Examples

```
# Get data
data(psyo_rounds2)
tracks <- psyo_rounds2

# Calculations
tracks <- average_duplicates(tracks)
tracks <- t_time_difference(tracks)

tracks <- mark_time_gaps(tracks)

tracks <- select_without_gaps(tracks,"time_gap")
```

---

t_bearing *Add bearings*

Description

Bearing towards the next coordinate in the tracks.

Usage

```
t_bearing(
  tracks, t_id = "id", bind = TRUE, drop = TRUE, cname = "bearings"
)
```

Arguments

tracks	psyo. Data frame with tracks.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.
bind	*logical*. Return the distance as list (FALSE) or add it to tracks (TRUE).
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
cname	character. Column name of the returned calculation result.

Details

For the last coordinate within a track the function returns `empty` because there are no bearings towards a following coordinate possible.

If the succeeding coordinate is the same like the current coordinate, the function return `empty` for the current coordinate.

Value

psyo

Note

Please be aware that this function calculates the initial bearing from the first to the second point and that this bearing is saved with the second point. This seems counter intuitive for an initial bearing but is done for better compatibility with the gap functions.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

bearing, t_distance, t_speed, t_time_difference

Examples

```
data(psyo)
psyo <- t_bearing(psyo)
```

---

t_distance *Add distance to next coordinate*

Description

Distance towards the next coordinate in the tracks.

Usage

```
t_distance(
  tracks, bind = TRUE, drop = TRUE, cname = "distances_in_m", t_id = "id"
)
```

Arguments

tracks	psyo. Data frame with tracks.
bind	*logical*. Return the distance as list (FALSE) or add it to tracks (TRUE).
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
cname	character. Column name of the returned calculation result.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.

Details

For the last coordinate within a track the function returns empty because there is no distance towards a following coordinate possible. Using 0 instead of NA may be an unwanted bias within the data.

Value

psyo

Note

The distance between the first and the second point is stored with the second point. This is done for higher compatibility with the gap functions.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

distHaversine, t_bearing, t_speed, t_time_difference

Examples

```
data(psyo)
psyo <- t_distance(psyo)
```

---

t_speed *Add speed*

Description

Speed towards the next coordinate in the track in kmh.

Usage

```
t_speed(
  tracks, bind = TRUE, drop = TRUE, cname = "speed", t_id = "id"
)
```

Arguments

tracks	psyo. Data frame with tracks.
bind	*logical*. Return the distance as list (FALSE) or add it to tracks (TRUE).
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
cname	character. Column name of the returned calculation result.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.

Details

For the last coordinate within a track the function returns NA because there is no speed towards a following coordinate possible.

Value

psyo

Note

The speed between the first and the second point is stored with the second point. This is done for higher compatibility with the gap functions.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

t_bearing,t_distance, t_time_difference

Examples

```
data(psyo)
psyo <- t_speed(psyo)
```

---

t_time_difference *Add time difference column for weighted statistics*

Description

This functions generates a column with the time difference to the next coordinate. This is important since the GPS variables seldom have the same time difference and means and standard deviations should be weighted. The generated column can be used as "weight" variable.

Usage

```
t_time_difference(
  tracks, units = "secs", bind = TRUE, drop = TRUE, cname = "time_difference",
  t_id = "id"
)
```

Arguments

tracks	psyo. Data frame with tracks.
units	character. Same as for link[base]{difftime} but avoid using "auto". Auto could generate different units for the different tracks.
bind	*logical*. Return the distance as list (FALSE) or add it to tracks (TRUE).
drop	logical. If TRUE and only one observation is returned drop the data frame and collapse the return value to a vector.
cname	character. Column name of the returned calculation result.
t_id	*character* or *numeric*. Column name in tracks that identifies the separate tracks.

Value

psyo

Note

The time difference between the first and the second point is stored with the second point. This is done for higher compatibility with the gap functions.

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

difftime, t_bearing,t_distance, t_speed

Examples

```
data(psyo)
t_time_difference(psyo, units = "secs")
```

val_cname *Validate the column name of a data frame*

Description

Validate the column name of a data frame

Usage

```
val_cname(tracks, cname, type = "", size = 0, force = 2, def = TRUE)
```

Arguments

tracks	psyo. Data frame with tracks.
cname	character. Column name of column in tracks that is to be validated.
type	character. Type of column in tracks determined by mode.
size	size. Observation count of column in tracks determined by length.
force	*numeric*. An error with force_id will be reported as stop when 2, warning when 1 or nothing when 0.
def	*logical*. Ignore this check if cname = "".

Value

character

Author(s)

Benjamin Ziepert. Please send feedback to: <feedback-psyosphere@analyse-gps.com>.

See Also

val_cname,val_psyo, val_var

Examples

```
data(psyo)

# Test t_id
t1 <- psyo
t_id <- "id"
e <- val_cname(t1, t_id, size = 15, type = "numeric"); if (e != "") {stop(e)}

# Be aware that id column is saved as "factor" and therefore mode() returns
# numeric.

t2 <- psyo
```

```
t_id <- 1
e <- val_cname(t2, t_id); if (e != "") {stop(e)}

t3 <- psyo
t_id <- "id"
e <- val_cname(t3, t_id, size = 15, type = "character"); if (e != "") {stop(e)}

t4 <- psyo
t_id <- "id"
e <- val_cname(t4, t_id, size = 20); if (e != "") {stop(e)}

t5 <- psyo
t_id <- "id"
e <- val_cname(t5, t_id, size = "20"); if (e != "") {stop(e)}
```

`val_psyo` *Validate psyo format*

Description

Checks if the provided data frame is conforming to the format that is used by 'psyosphere' and returns a `warning` or `stop` if necessary.

Usage

```
val_psyo(tracks, id = 1, p_id = 1, time = 1, lon = 2, lat = 2)
```

Arguments

`tracks`	*psyo*. The data frame that is to be check if it confirms to the `psyo` format.
`id`	*numeric*. An error with `force_id` will be reported as stop when `2`, warning when `1` or nothing when `0`.
`p_id`	*numeric*. An error with `force_p_id` will be reported as stop when `2`, warning when `1` or nothing when `0`.
`time`	*numeric*. An error with `force_time` will be reported as stop when `2`, warning when `1` or nothing when `0`.
`lon`	*numeric*. An error with `force_lon` will be reported as stop when `2`, warning when `1` or nothing when `0`.
`lat`	*numeric*. An error with `force_lat` will be reported as stop when `2`, warning when `1` or nothing when `0`.

Author(s)

Benjamin Ziepert. Please send feedback to: `<feedback-psyosphere@analyse-gps.com>`.

See Also

`val_cname`,`val_psyo`, `val_var`

Examples

```
# Produce a warning ------------------------------------------------------------
data(psyo)
psyo$time <- NULL # remove time column
e <- val_psyo(psyo); if (e != "") {stop(e)}

# Produce a stop ---------------------------------------------------------------
data(psyo)
psyo$time <- NULL # remove time column
e <- val_psyo(psyo, time = 2); if (e != "") {stop(e)}

# Produce a stop without setting "force" ---------------------------------------
data(psyo)
psyo$lon <- NULL # remove time column
e <- val_psyo(psyo); if (e != "") {stop(e)}
```

`val_var` *Validate variables*

Description

Validates variables before further procedure execution.

Usage

```
val_var(test_var, type, force = 2, size = 0, def = FALSE)
```

Arguments

`test_var`	*Multiple.* The variable that is to be tested.
`type`	*Character.* The variable type determined by `mode` or `lubridate::is.POSIXct`. For example *numeric*, *character*, *logical*, *list*, *POSIXct* or *ggplot*.
`force`	*numeric.* Error message is sent as warning (*1*) or stop (*2*).
`size`	*numeric.* If `size` is not 0 the length of `test_var` will be checked with `size`.
`def`	*logical*. Ignore this check if `cname = ""`.

Author(s)

Benjamin Ziepert

See Also

`val_cname`,`val_psyo`, `val_var`

Examples

```
# Create variables
id <- 10
name <- "test"
time <- as.POSIXct("1986-08-31 02:15:00")

# Check variables
# e <- val_var(id, "character"); if (e != "") {stop(e)} # error and stop
# e <- val_var(name, "logical", FALSE); if (e != "") {stop(e)} # error and warning
e <- val_var(time, "POSIXct"); if (e != "") {stop(e)} # no error
```

Index