Package ‘sftrack’

July 1, 2021

Title  Modern Classes for Tracking and Movement Data
Version 0.5.3
Date 2021-07-01
Depends R (>= 3.2.0)
Imports sf

Description Modern classes for tracking and movement data, building on 'sf' spatial infrastructure, and early theoretical work from Turchin (1998, ISBN: 9780878938476), and Calenge et al. (2009) <doi:10.1016/j.ecoinf.2008.10.002>. Tracking data are series of locations with at least 2-dimensional spatial coordinates (x,y), a time index (t), and individual identification (id) of the object being monitored; movement data are made of trajectories, i.e. the line representation of the path, composed by steps (the straight-line segments connecting successive locations). 'sftrack' is designed to handle movement of both living organisms and inanimate objects.

License MIT + file LICENSE
Encoding UTF-8
LazyData true
RoxygenNote 7.1.1
VignetteBuilder knitr


BugReports https://github.com/mablab/sftrack/issues
Suggests testthat (>= 2.1.0), utils, ggplot2, adehabitatLT, knitr, geosphere, scales, covr, rmarkdown, lwgeom

NeedsCompilation no

Author Matthew Boone [aut] (<https://orcid.org/0000-0002-8070-4715>), Rocío Joo [aut] (<https://orcid.org/0000-0003-0319-4210>), Emiel Van Loon [ctb] (<https://orcid.org/0000-0002-8895-0427>), Mathieu Basille [aut, cre] (<https://orcid.org/0000-0001-9366-7127>)
Maintainer Mathieu Basille <mathieu@basille.org>
Repository CRAN
Date/Publication 2021-07-01 15:50:02 UTC

R topics documented:

active_group .................................................. 3
active_group<- ............................................... 4
as_sftrack ................................................... 4
as_sftraj ..................................................... 7
calc_sort_index ............................................... 9
check_group_id ............................................... 10
check_group_names .......................................... 10
check_names_exist .......................................... 11
check_NA_coords ........................................... 11
check_NA_group ............................................. 11
check_ordered ............................................... 12
check_time ................................................... 12
check_t_regular ............................................ 13
dup_timestamp ............................................... 13
fix_zero ..................................................... 14
geom_sftrack ............................................... 14
grouping-class ............................................. 15
group_labels ............................................... 17
group_names ................................................. 17
is_linestring ............................................... 18
make_step_geom ............................................ 18
merge_traj .................................................. 19
new_sftrack ................................................ 19
new_sftraj .................................................. 20
plot_sftrack ............................................... 20
Print_sftrack_objects .................................... 21
Print_sftraj_objects ...................................... 22
raccoon ...................................................... 23
step_metrics ................................................. 23
step_recalc ................................................ 24
summary_sftrack .......................................... 24
traj_geom .................................................... 25
which_duplicated .......................................... 25

Index 27
active_group

Access the active_group value

Description

The active group is the combination of group names to group the data sets. The active_group acts essentially like a paste(names_of_groups, sep = '_') grouping variable.

Usage

active_group(x)

Arguments

x       a c_grouping

Examples

```r
#'
data("raccoon")
raccoon$timestamp <- as.POSIXct(raccoon$timestamp, "EST")
burstz <- list(id = raccoon$animal_id, month = as.POSIXlt(raccoon$timestamp)$mon)
mb1 <- make_c_grouping(x = burstz, active_group = c("id", "month"))

# see the current active burst
active_group(mb1)

# change the active burst
active_group(mb1) <- "id"

# Using a full data set
my_track <- as_sftrack(raccoon,
  time = "timestamp",
  error = NA, coords = c("longitude", "latitude"),
  group = burstz
)

summary(my_track)

# change active group
active_group(my_track$sft_group) <- "id"

summary(my_track)
```
active_group<-  Set new active group

Description
Set new active group

Usage
active_group(x) <- value

## S3 replacement method for class 'sftrack'
active_group(x) <- value

## S3 replacement method for class 'sftraj'
active_group(x) <- value

## S3 replacement method for class 'c_grouping'
active_group(x) <- value

Arguments
x  sftrack/sftraj/c_grouping/s_group object
value  character vector of the grouping names to make active

as_sftrack  Convert objects into sftrack objects.

Description
This function converts x,y,z data into an sftrack object with a sf_geometry column of sf_POINTS. Creates a ‘grouping’ column to group movement data and sets dedicated time and error columns.

Raw data inputted in two ways: vector or data.frame. ’Vector’ inputs gives the argument as a vector where length = nrow(data). ’Data.frame’ inputs gives the arguments as the column name of ‘data’ where the input can be found. Either input is allowed on any given argument.

Some options are global and required regardless

Usage
as_sftrack(data = data.frame(), ...)

## S3 method for class 'data.frame'
as_sftrack(
  data = data.frame(),

as_sftrack

...,
coords = c("x", "y"),
group = "id",
active_group = NA,
time = "time",
error = NA,
crs = NA,
zeroNA = FALSE,
group_name = "sft_group",
timestamp_name = "sft_timestamp",
error_name = "sft_error",
overwrite_names = FALSE
)

## S3 method for class 'sftraj'
as_sftrack(data, ...)

## S3 method for class 'ltraj'
as_sftrack(data, ...)

## S3 method for class 'sf'
as_sftrack(
data,
..., coords,
group,
active_group = NA,
time,
error = NA,
group_name = "sft_group",
timestamp_name = "sft_timestamp",
error_name = "sft_error",
overwrite_names = FALSE
)

Arguments

data a data.frame of the movement data, if supplied all data.frame inputs, than is optional
...
extra information to be passed on to as_sftrack
coords a character vector describing where the x,y,z coordinates are located in ‘data’ or a list with x,y,z (optional) vectors
group a list of named vectors describing multiple grouping variables or a character vector naming the other grouping columns in ‘data’.
active_group a character vector of the burst names to be ‘active’ to group data by for analysis
time a vector of time information, can be either POSIX or an integer or a character string naming the column in ‘data’ where the time information is located
error (optional) a vector of error information for the movement data
name naming the column in ‘data’ where the error information is located

crs a crs string from rgdal of the crs and projection information for the spatial data.
Defaults to NA

zeroNA logical whether to convert 0s in spatial data into NAs. Defaults to FALSE.

group_name (optional) new column name for grouping data

timestamp_name (optional) new column name for time data

error_name (optional) new column name for error data

overwrite_names T/F Whether to overwrite data if a group/time/error column name is supplied
but already in data

Details

Convert objects into sftrack objects.

Examples

```
#'
data("raccoon")
raccoon$timestamp <- as.POSIXct(raccoon$timestamp, "EST")
burstz <- list(id = raccoon$animal_id, month = as.POSIXlt(raccoon$timestamp)$mon)
# Input is a data.frame
my_track <- as_sftrack(raccoon,
  group = burstz, time = "timestamp",
  error = NA, coords = c("longitude", "latitude")
)

# Input is a ltraj
library("adehabitatLT")
ltraj_df <- as.ltraj(
  xy = raccoon[, c("longitude", "latitude")],
  date = as.POSIXct(raccoon$timestamp),
  id = raccoon$animal_id, typeII = TRUE,
  infolocs = raccoon[, 1:6]
)

my_sftrack <- as_sftrack(ltraj_df)
head(my_sftrack)

# Input is a sf object
library("sf")
dfl <- raccoon[!is.na(raccoon$latitude), ]
sf_df <- st_as_sf(dfl, coords = c("longitude", "latitude"))

new_sftrack <- as_sftrack(sf_df, group = c(id = "animal_id"), time = "timestamp")
head(new_sftrack)

# Input is an sftraj object
my_traj <- as_sftraj(raccoon,
```
as_sftraj

```r
time = "timestamp",
error = NA, coords = c("longitude", "latitude"),
group = burstz

new_track <- as_sftrack(my_traj)
head(new_track)
```

### as_sftraj

**Convert objects into sftrack objects.**

#### Description

This function converts x,y,z data into an sftrack object with a sfGeometry column of sf_POINTS. Creates a ‘grouping’ column to group movement data and sets dedicated time and error columns.

Raw data inputted in two ways: vector or data.frame. ‘Vector’ inputs gives the argument as a vector where length = nrow(data). ‘Data.frame’ inputs gives the arguments as the column name of ‘data’ where the input can be found. Either input is allowed on any given argument.

Some options are global and required regardless

#### Usage

```r
as_sftraj(data = data.frame(), ...)
```

```r
## S3 method for class 'data.frame'
as_sftraj(
  data = data.frame(),
  ...,
  coords = c("x", "y"),
  group = "id",
  active_group = NA,
  time = "time",
  error = NA,
  crs = NA,
  zeroNA = FALSE,
  group_name = "sft_group",
  timestamp_name = "sft_timestamp",
  error_name = "sft_error",
  overwrite_names = FALSE
)
```

```r
## S3 method for class 'sftrack'
as_sftraj(data, ...)
```

```r
## S3 method for class 'sf'
```
as_sftraj(
  data,
  ..., 
  coords,
  group,
  active_group = NA,
  time,
  error = NA,
  group_name = "sft_group",
  timestamp_name = "sft_timestamp",
  error_name = "sft_error",
  overwrite_names = FALSE
)

## S3 method for class 'ltraj'
as_sftraj(data, ...)

Arguments

data a data.frame of the movement data, if supplied all data.frame inputs, than is optional
...
extra information to be passed on to as_sftrack
coords a character vector describing where the x,y,z coordinates are located in ‘data’ or a list with x,y,z (optional) vectors
group a list of named vectors describing multiple grouping variables or a character vector naming the other grouping columns in ‘data’.
active_group a character vector of the burst names to be 'active' to group data by for analysis
time a vector of time information, can be either POSIX or an integer or a character string naming the column in ‘data’ where the time information is located
error (optional) a vector of error information for the movement dataa character string naming the column in ‘data’ where the error information is located
crs a crs string from rgdal of the crs and projection information for the spatial data. Defaults to NA
zeroNA logical whether to convert 0s in spatial data into NAs. Defaults to FALSE.
group_name (optional) new column name for grouping data
timestamp_name (optional) new column name for time data
error_name (optional) new column name for error data
overwrite_names T/F Whether to overwrite data if a group/time/error column name is supplied but already in data

Details

Convert objects into sftrack objects.
Examples

```r
#'
data("raccoon")
raccoon$timestamp <- as.POSIXct(raccoon$timestamp, "EST")
burstz <- list(id = raccoon$animal_id, month = as.POSIXlt(raccoon$timestamp)$mon)

# Input is a data.frame
my_track <- as_sftrack(raccoon,
    group = burstz, time = "timestamp",
    error = NA, coords = c("longitude", "latitude")
)

# Input is a ltraj
library("adehabitatLT")
ltraj_df <- as.ltraj(
    xy = raccoon[, c("longitude", "latitude")],
    date = as.POSIXct(raccoon$timestamp),
    id = raccoon$animal_id, typeII = TRUE,
    infolocs = raccoon[, 1:6]
)

my_sftrack <- as_sftrack(ltraj_df)
head(my_sftrack)

# Input is a sf object
library("sf")
df1 <- raccoon[!is.na(raccoon$latitude), ]
sf_df <- st_as_sf(df1, coords = c("longitude", "latitude"))

new_sftrack <- as_sftrack(sf_df, group = c(id = "animal_id"), time = "timestamp")
head(new_sftrack)

# Input is an sftrack object
my_track <- as_sftrack(raccoon,
    time = "timestamp",
    error = NA, coords = c("longitude", "latitude"),
    group = burstz
)

new_traj <- as_sftraj(my_track)
head(new_traj)
```

---

calc_sort_index | Calculate a new sort index for groups

Description

Calculate a new sort index for groups
Usage

`calc_sort_index(x, active_group = NA)`

Arguments

- **x**: group or sftrack object
- **active_group**: (optional), a new active group. If not included, defaults to the active group (if a c_grouping) or the group names

---

**check_group_id**

*Check there is a grouping id present*

---

Description

Check there is a grouping id present

Usage

`check_group_id(x)`

Arguments

- **x**: a c_grouping

---

**check_group_names**

*Are group names equivalent for each s_group?*

---

Description

Are group names equivalent for each s_group?

Usage

`check_group_names(x)`

Arguments

- **x**: a c_grouping
check_names_exist  

Check if a set of column names are found in a data frame and return an error if not

Description
Check if a set of column names are found in a data frame and return an error if not

Usage
check_names_exist(data, names)

Arguments
- data: a data.frame to check names against
- names: the inputted column names

check_NA_coords  

Check if coordinates contain NAs in some columns but not others

Description
Check if coordinates contain NAs in some columns but not others

Usage
check_NA_coords(xyz)

Arguments
- xyz: a data.frame of xy or xyz coordinates

check_NA_group  

Check there are no NAs in burst

Description
Check there are no NAs in burst

Usage
check_NA_group(x)

Arguments
- x: a c_grouping
check_ordered

Checks if grouping is ordered by time and then outputs the correct order

Description
Checks if grouping is ordered by time and then outputs the correct order

Usage
check_ordered(group, time_data, return = TRUE)

Arguments
- group: a c_grouping
- time_data: a vector of time
- return: T/F return the new order or just run check?

check_time

Check if time is integer or posix

Description
Check if time is integer or posix

Usage
check_time(time)

Arguments
- time: a vector of time
check_t_regular

Check if time is regular for each burst and returns logical for each burst

Usage

\texttt{check\_t\_regular(x)}

Arguments

\begin{itemize}
  \item \texttt{x} \hspace{1cm} an sftrack/sftraj object
\end{itemize}

dup_timestamp

check that time is unique

Usage

\texttt{dup\_timestamp(x, time)}

Arguments

\begin{itemize}
  \item \texttt{x} \hspace{1cm} An sftrack/sftraj object or a multi_burst
  \item \texttt{time} \hspace{1cm} vector of time, not required if given a sftrack object.
fix_zero

**Description**

fix 0’s to NAs in latitude and longitude

**Usage**

fix_zero(xyz)

**Arguments**

xyz 
a data.frame of xy or xyz coordinates

**Value**

returns a data.frame with 0s replaced with NAs

---

geom_sftrack

**Description**

Function to plot sftrack objects in ggplot

**Usage**

geom_sftrack(mapping, data, ...)

## S3 method for class 'sftrack'
geom_sftrack(mapping = ggplot2::aes(), data = NULL, ...)

## S3 method for class 'sftraj'
geom_sftrack(mapping = ggplot2::aes(), data = NULL, ..., step_mode = FALSE)

**Arguments**

mapping 
mapping aesthetics for ggplot.

data 
the sftraj or sftrack object.

... 
arguments to passed to ggplot

step_mode 
TRUE/FALSE, whether to plot in step_mode, See details
Details

step mode refers to considering the trajectory as individual 'steps'. in the case of plot this means it will plot each line & point individually. This approach is much slower to plot when n(steps)>10,000. The alternative method is to merge the steps into a multilinestring of continuous lines. This is much faster to plot.

Examples

```
#
require("ggplot2")
data("raccoon")
raccoon$timestamp <- as.POSIXct(raccoon$timestamp, "EST")
burstz <- c(id = "animal_id")

# sftraj will as well for the most part, however as its a more complex
# structure to speed up plotting.
my_sftraj <- as_sftraj(raccoon,
    time = "timestamp",
    coords = c("longitude", "latitude"),
    group = burstz
)

ggplot() +
    geom_sftrack(data = my_sftraj)
```

---

**grouping-class**  
__A class to group movement data__

Description

This class describes grouping variables for movement data. The grouping object is composed of a list with named vectors. One of which must be 'id', this is the id of subject being monitored (commonly animal id in movement data) Can be any number of groups after that.

Usage

```
make_s_group(x)

make_c_grouping(x = NULL, active_group = NULL)

## S3 method for class 's_group'
c(...)

## S3 method for class 'c_grouping'
c(..., recursive = FALSE)
```
Arguments

x a list containing named grouping variables, one item must be named 'id'. ex: list(id = 1, month = 'may'). For a c_grouping: A list of s_groups or a list of equal length named vectors which will be combined to create a c_grouping. ex: list(x = 1st_vector, y = 2nd_vector)

active_group a vector of the names of the groups to be considered 'active'.

... objects to be pasted together into a c_grouping

recursive ignored

Details

A grouping is a list of possible categories to group the data. The 'active group' of these is the current grouping variables to be considered for analysis. The 'active group' can be any combination of the categories in a burst, and can change with the use of 'active_group()'.

An 's_group' is a single row group. It is a 1xn dimensional list with any length(n) > 1. Atleast one of the groups must be named 'id' which is the subjects id.

A 'c_grouping' is a collection of 's_groups's, it is a data.frame with dimensions of 1xrown(data). One c_grouping has one 'active group' which describes the set of names in each s_group to group the data. When you change the 'active group', calculations and plots change accordingly to the new grouping levels.

You can create bursts with make_s_group and make_c_grouping.

Examples

# Make a single group
#
make_s_group(x = list(id = "CJ11", month = 3, height = 10))

# Make a c_grouping
data("raccoon")
raccoon$timestamp <- as.POSIXct(raccoon$timestamp, "EST")
burstz <- list(id = raccoon$animal_id, month = as.POSIXlt(raccoon$timestamp)$mon)
mb1 <- make_c_grouping(x = burstz, active_group = c("id", "month"))
str(mb1)

# Make a multi_burst from many ind_bursts
a <- make_s_group(list(id = 1, year = 2020))
b <- make_s_group(list(id = 1, year = 2020))
c <- make_s_group(list(id = 2, year = 2020))
c(a, b, c)
**group_labels**

Shows grouping labels created from the `s_group` and the `c_grouping`

**Description**

Shows grouping labels created from the `s_group` and the `c_grouping`

**Usage**

```r
group_labels(x)
```

```r
## S3 method for class 'sftrack'
group_labels(x)
```

```r
## S3 method for class 'sftraj'
group_labels(x)
```

```r
## S3 method for class 'c_grouping'
group_labels(x)
```

**Arguments**

- `x`: a `sftrack` or grouping object

---

**group_names**

Display the levels of the sort index

**Description**

Display the levels of the sort index

**Usage**

```r
group_names(x)
```

```r
## S3 method for class 'c_grouping'
group_names(x)
```

```r
## S3 method for class 'sftraj'
group_names(x)
```

```r
## S3 method for class 'sftrack'
group_names(x)
```

**Arguments**

- `x`: sftrack/sftraj/c_grouping/s_group object
is_linestring  
Is a trajectory geometry a linestring or a point

Description

A step is a movement from one point to the next, with an sftraj object this manifests as a linestring. If, however, one of these two points is missing, the sftraj is created as a geometry collection of two points, the beginning and the end point, where one of the steps is NA. This function checks a trajectory geometry if its a linestring and returns a vector of T/F.

Usage

is_linestring(x)

Arguments

x  
an sftraj object

make_step_geom  
Calculate step geometries given a set of groupings, time, and geometries

Description

This calculates step geometries as individual line segments based on the active_group.

Usage

make_step_geom(group, time_data, geometry)

Arguments

group  
a c_grouping object
time_data  
time vector
group  
the geometry data from either sf or sf_track. Must be an sf geometry class

Examples

#'  
library("sf")  
geom <- st_as_sf(data.frame(
  x = c(1, 2, 2, 5),
  y = c(0, 1, 5, 7),
), coords = c("x", "y"))  
burst <- list(id = rep(1, 4))
merge_traj

```r
time <- 1:4
cg <- make_c_grouping(burst)
make_step_geom(
  group = cg,
  geometry = geom$geometry,
  time_data = time
)
```

**merge_traj**  
*Merge connected lines and create an sf object*

**Description**

This function returns a sf object grouped by each burst with a geometry column of multilinestrings for each grouping.

**Usage**

```r
merge_traj(x)
```

**Arguments**

- `x`: an sftraj object

---

**new_sftrack**  
*Define an sftrack*

**Description**

Define an sftrack.

**Usage**

```r
new_sftrack(data, group_col, sf_col, time_col, error_col = NA)
```

**Arguments**

- `data`: data.frame with multi_burst column, geometry column, time_col (integer/POSIXct), and error column (optional)
- `group_col`: column name of grouping info in `data`
- `sf_col`: column name of geometry info in `data`
- `time_col`: column name of time info in `data`
- `error_col`: column name of error info in `data`
new_sftraj | Define an sftraj

**Description**

Define an sftraj

**Usage**

```
new_sftraj(data, group_col, sf_col, time_col, error_col = NA)
```

**Arguments**

- `data`: data.frame with multi_burst column, geometry column, time_col (integer/POSIXct), and error column (optional)
- `group_col`: column name of multi_burst in ‘data’
- `sf_col`: column name of geometry in ‘data’
- `time_col`: column name of time in ‘data’
- `error_col`: column name of error in ‘data’

plot_sftrack | Methods for plotting sftrack/sftraj

**Description**

Methods for plotting sftrack/sftraj

**Usage**

````
## S3 method for class 'sftrack'
plot(x, y, key.pos, key.width, ...)

## S3 method for class 'sftraj'
plot(x, y, key.pos, key.width, ..., step_mode)
```

**Arguments**

- `x`: ‘sftrack’ or ‘sftraj’ object
- `y`: ignored
- `key.pos`: Integer; side to plot a color key: 1 bottom, 2 left, 3 top, 4 right; set to NULL to omit key, or -1 to select automatically (defaults to 4; see `plot_sf` for more details).
key.width  Amount of space reserved for the key, including labels (see plot.sf for more details.)

...  Further arguments passed to 'plot.sf'. Among others, arguments for the key are set differently in 'sftrack' to allow for longer labels by default (but can be nevertheless adjusted).

step_mode  Logical; whether to plot in step mode, see details, defaults to TRUE, unless there are more than 10,000 steps.

Details

Step mode refers to considering the trajectory as individual 'steps', in the case of plot this means it will plot each line & point individually. This approach is much slower to plot with large objects, and is thus turned off when n(steps)>10,000. The alternative, much faster method is to merge the steps into a multilinestring as continuous lines.

Examples

```r
## Prepare an 'sftrack' object:
data("raccoon")
raccoon$timestamp <- as.POSIXct(raccoon$timestamp, "EST")
burstz <- c(id = "animal_id")
my_sftrack <- as_sftrack(raccoon,
    time = "timestamp",
    coords = c("longitude", "latitude"),
    group = burstz
)

## Plotting with sftrack is just like sf. `...` will accept most
## arguments as 'plot.sf':
plot(my_sftrack, axes = TRUE, lwd = 5, cex = 5, bgc = "gray50")

## sftraj will as well for the most part; however it is a more complex
## structure that combines points and steps (in step mode):
my_sftraj <- as_sftraj(raccoon,
    time = "timestamp",
    coords = c("longitude", "latitude"),
    group = burstz
)
plot(my_sftraj, lwd = 5, cex = 5, bgc = "gray50", graticule = TRUE)
```
Usage

```r
## S3 method for class 'sftrack'
print(x, n_row, n_col, ...)
```

Arguments

- `x`: sftraj object
- `n_row`: Integer of number of rows to display. Defaults to global option default if non supplied
- `n_col`: Integer of number of columns to display + required sftrack columns (burst, geometry, time, and error). Defaults to global option default if non supplied
- `...`: other arguments passed onto print

Description

Print methods for sftraj

Usage

```r
## S3 method for class 'sftrack'
print(x, n_row, n_col, ...)
```

Arguments

- `x`: sftraj object
- `n_row`: Integer of number of rows to display. Defaults to global option default if non supplied
- `n_col`: Integer of number of columns to display + required sftrack columns (burst, geometry, time, and error). Defaults to global option default if non supplied
- `...`: other arguments passed onto print
 Movements of two raccoons in an urban park in Florida

Description

A dataset of two raccoons collared with GPS collars for one month in January 2019 in Tree Tops Park, Broward County, Florida, US.

Usage

raccoon

Format

A data frame with 445 rows and 10 variables:

animal_id  ID of individual. TTP: tree tops park, i.e the tagging site.
timestamp  The date and time of gps fix in UTC
latitude   Latitude in degrees
longitude  Longitude in degrees
height     Altitude in meters based on satellite positios
hdop       Horizontal precision
vdop       Vertical precision
fix        The number of satellite fixes

step_metrics  Calculates step metrics including distance, dt, dx, and dy.

Description

Calculates step metrics including distance, dt, dx, and dy.

Usage

step_metrics(sftraj)

Arguments

sftraj        an sftrack/sftraj object. sftrack objects will be converted to sftraj internally for calculation.
Examples

```r
#' data("raccoon")
raccoon$timestamp <- as.POSIXct(raccoon$timestamp, "EST")
burstz <- list(id = raccoon$animal_id, month = as.POSIXlt(raccoon$timestamp)$mon)
# Input is a data.frame
my_sftraj <- as_sftraj(raccoon,
    group = burstz, time = "timestamp",
    error = NA, coords = c("longitude", "latitude")
)

step_metrics(my_sftraj)[1:10, ]
```

---

**step_recalc**  
recalculate step geometry

**Description**

Step geometries in sftraj objects are linestrings going from t1 to t2 of a 'step'. As these are stored at the row level they are not dynamic to changes in t2. `step_recalc` allows you to recalculate these geometries if your data.frame has changed because of subsetting or filtering.

**Usage**

`step_recalc(x, return = FALSE)`

**Arguments**

- **x**: an sftraj object.
- **return**: return step_geometry instead of replacing sftraj object with new step geometry. Defaults to FALSE

---

**summary_sftrack**  
Summarize sftrack objects

**Description**

Summarize sftrack objects

**Usage**

`summary_sftrack(x)`

**Arguments**

- **x**: an sftrack object
Return a list of sf_POINTS or a data.frame from a sftraj object

Usage

```r
pts_traj(traj, sfc = FALSE)
coord_traj(traj)
```

Arguments

- `traj` a trajectory geometry from sf_traj
- `sfc` TRUE/FALSE should the return be an sfc or a list of points. Defaults to FALSE

Examples

```r
# data("raccoon")
raccoon$timestamp <- as.POSIXct(raccoon$timestamp, "EST")
burstz <- list(id = raccoon$animal_id, month = as.POSIXlt(raccoon$timestamp)$mon)
# Input is a data.frame
my_traj <- as_sftraj(raccoon,
  time = "timestamp",
  error = NA, coords = c("longitude", "latitude"),
  group = burstz
)
print(my_traj, 5, 10)

# extract a list of points
pts_traj(my_traj)[1:10]

# or a data.frame of points
coord_traj(my_traj)[1:10]
```

which_duplicated Which grouping/time stamp combos are duplicated.

Description

This function returns a data.frame of which rows are duplicated and their time stamps.
Usage

which_duplicated(data = data.frame(), group, time)

Arguments

data a data.frame containing burst or time data (if necessary)
group a list where each entry is a vector of groupings where length == nrow(data)\nrow(time). Or a character vector describing the column name they are located in data
time a vector of as.POSIXct time, or a character of the column name where it can be found in data
Index

* datasets
  raccoon, 23
active_group, 3
active_group<-, 4
active_group_replace (active_group<->), 4
as_sftrack, 4
as_sftraj, 7
c.c_grouping (grouping-class), 15
c.s_group (grouping-class), 15
calc_sort_index, 9
check_group_id, 10
check_group_names, 10
check_NA_coords, 11
check_NA_group, 11
check_names_exist, 11
check_ordered, 12
check_t_regular, 13
check_time, 12
coord_traj (traj_geom), 25
dup_timestamp, 13
fix_zero, 14
geom_sftrack, 14
group_labels, 17
group_name (group_names), 17
group_names, 17
grouping-class, 15
is_linestring, 18
make_c_grouping (grouping-class), 15
make_s_group (grouping-class), 15
make_step_geom, 18
merge_traj, 19
new_sftrack, 19
new_sftraj, 20
plot_sftrack (plot_sftrack), 20
plot_sftraj (plot_sftrack), 20
plot_sf, 20, 21
plot_sftrack, 20
plot_sftrack (Print_sftrack_objects), 21
plot_sftraj (Print_sftraj_objects), 22
Print_sftrack_objects, 21
Print_sftraj_objects, 22
pts_traj (traj_geom), 25
raccoon, 23
step_metrics, 23
step_recalc, 24
summary_sftrack, 24
traj_geom, 25
which_duplicated, 25