# Package 'trackdf'

July 22, 2025

**Type** Package

**Title** Data Frame Class for Tracking Data

**Version** 0.3.3

**Date** 2024-01-28

**Maintainer** Simon Garnier <garnier@njit.edu>

**Description** Data frame class for storing collective movement data (e.g. fish schools, ungulate herds, baboon troops) collected from GPS trackers or computer vision tracking software.

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.1

**Depends** R (>= 3.5.0)

**Imports** sf, tibble, data.table, lubridate, methods, dplyr

**Suggests** readr, ggplot2, mapproj, knitr, rmarkdown, adehabitatLT, move, ctmm, moveHMM, sp, terra, covr, testthat (>= 3.0.0)

**VignetteBuilder** knitr

**URL** https://swarm-lab.github.io/trackdf/, https://github.com/swarm-lab/trackdf

**BugReports** https://github.com/swarm-lab/trackdf/issues

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Simon Garnier [aut, cre] (ORCID: <https://orcid.org/0000-0002-3886-3974>)

**Repository** CRAN

**Date/Publication** 2024-01-29 10:00:02 UTC

# Contents

.Mode . . . 2
.reclass . . . 3
bind_tracks . . . 3
conversions . . . 4
dplyr_track . . . 6
duplicated_data . . . 8
filter . . . 9
inconsistent_data . . . 9
is_geo . . . 10
is_track . . . 11
missing_data . . . 12
n_dims . . . 13
n_tracks . . . 13
projection . . . 14
short_tracks . . . 15
tracks . . . 16
track_ . . . 16
[.track . . . 18

**Index** **20**

---

`.Mode` *Compute The Mode(s) Of A Discrete Distribution*

---

## Description

This is an internal utility function to compute the mode(s) of a discrete distribution.

## Usage

```
.Mode(x, na.rm = TRUE)
```

## Arguments

| | |
|---|---|
| `x` | A vector or matrix of discrete values. |
| `na.rm` | A logical value indicating whether NA values should be stripped before the computation proceeds (default: TRUE). |

## Value

A vector of values corresponding to the mode(s) of x.

## Author(s)

Simon Garnier, <garnier@njit.edu>

## .reclass *Maintain Class After Modification*

### Description

Copy class and attributes from the original version of an object to a modified version.

### Usage

```
.reclass(x, result)
```

### Arguments

| | |
|---|---|
| x | The original object, which has a class/attributes to copy |
| result | The modified object, which is/might be missing the class/attributes. |

### Value

result, now with class/attributes restored.

### Author(s)

Simon Garnier, <garnier@njit.edu>

## bind_tracks *Bind Multiple Track Tables by Row*

### Description

bind_tracks uses data.table::rbindlist to combine track tables by rows, but makes sure that you cannot bind together two tables with different projections or time zones, that the projection attribute is inherited by the resulting track table, and that track tables based on different table classes are coerced to the same table class.

### Usage

```
bind_tracks(...)
```

### Arguments

| | |
|---|---|
| ... | A list containing track table objects, or the names of track table objects separated by commas. The track tables must have the same projection and time zone. |

### Value

A track table.

## Author(s)

Simon Garnier, <garnier@njit.edu>

## Examples

```
data(short_tracks)

bind_tracks(short_tracks, short_tracks)
bind_tracks(list(short_tracks, short_tracks))
```

---

conversions *Convert a Track Table to/from Other Formats*

---

## Description

The following methods will convert track tables to and from other common formats used for processing tracking and spatial data.

## Usage

```
as_track(x, table = "df", ...)

## S3 method for class 'MoveStack'
as_track(x, table = "df", ...)

## S3 method for class 'Move'
as_track(x, table = "df", ...)

as_move(x, ...)

## S3 method for class 'SpatialPointsDataFrame'
as_track(x, table = "df", ...)

as_sp(x, ...)

## S3 method for class 'track'
as_sp(x, ...)

## S3 method for class 'ltraj'
as_track(x, table = "df", ...)

as_ltraj(x, ...)

## S3 method for class 'track'
as_ltraj(x, ...)
```

```
## S3 method for class 'telemetry'
as_track(x, table = "df", ...)

## S3 method for class 'list'
as_track(x, table = "df", ...)

as_telemetry(x, ...)

## S3 method for class 'track'
as_telemetry(x, ...)

## S3 method for class 'moveData'
as_track(x, table = "df", type = c("LL", "UTM"), ...)

as_moveHMM(x, ...)

## S3 method for class 'track'
as_moveHMM(x, ...)
```

## Arguments

| | |
|---|---|
| x | An object to convert. |
| table | A string indicating the class of the table on which the track table should be built. It can be a `data.frame` ("df", the default), a `tibble` ("tbl"), or a `data.table` ("dt"). |
| ... | Other parameters to be passed to:<br>• `track_df`, `track_tbl` or `track_dt` if `as_track` is used.<br>• `sp::SpatialPointsDataFrame` if `as_sp` is used.<br>• `adehabitatLT::as.ltraj` if `as_ltraj` is used.<br>• `ctmm::as.telemetry` if `as_telemetry` is used.<br>• `moveHMM::prepData` if `as_moveHMM` is used. |
| type | For converting `moveHMM::moveData` to track table only, a character string indicating the type of coordinates stored in the `moveHMM::moveData` object: "LL" if longitude/latitude (default), "UTM" if easting/northing. |

## Value

The coordinates converted in the chosen format.

## Author(s)

Simon Garnier, <garnier@njit.edu>

## See Also

`track_df`, `track_tbl`, `track_dt`

## Examples

```
## Not run:
data(short_tracks)

if (requireNamespace("sp", quietly = TRUE)) {
  sp <- as_sp(short_tracks)
  as_track(sp)
}

if (requireNamespace("adehabitatLT", quietly = TRUE)) {
  lt <- as_ltraj(short_tracks)
  as_track(lt)
}

if (requireNamespace("ctmm", quietly = TRUE)) {
  tl <- as_telemetry(short_tracks)
  as_track(tl)
}

if (requireNamespace("moveHMM", quietly = TRUE)) {
  hhm <- as_moveHMM(short_tracks, type = "LL")
  as_track(hhm)
}

## End(Not run)
```

---

`dplyr_track` *Dplyr Methods For Track Tables*

---

## Description

dplyr methods for track tables objects.

## Usage

```
## S3 method for class 'track'
select(.data, ...)

## S3 method for class 'track'
rename(.data, ...)

## S3 method for class 'track'
filter(.data, ...)

## S3 method for class 'track'
arrange(.data, ...)
```

```
## S3 method for class 'track'
mutate(.data, ...)

## S3 method for class 'track'
transmute(.data, ...)

## S3 method for class 'track'
summarise(.data, ...)

## S3 method for class 'track'
summarize(.data, ...)

## S3 method for class 'track'
group_by(.data, ...)

## S3 method for class 'track'
ungroup(x, ...)

## S3 method for class 'track'
slice_sample(.data, ...)

## S3 method for class 'track'
do(.data, ...)

## S3 method for class 'track'
slice(.data, ...)

## S3 method for class 'track'
semi_join(x, ...)

## S3 method for class 'track'
anti_join(x, ...)

## S3 method for class 'track'
inner_join(x, ...)

## S3 method for class 'track'
left_join(x, ...)

## S3 method for class 'track'
right_join(x, ...)

## S3 method for class 'track'
full_join(x, ...)

## S3 method for class 'track'
nest_join(x, ...)
```

```
## S3 method for class 'track'
distinct(.data, ...)

## S3 method for class 'track'
rowwise(data, ...)
```

### Arguments

| | |
|---|---|
| `...` | Additional arguments to be passed to the corresponding `dplyr` method. |
| `data`, `.data`, `x` | A track table. |

### See Also

`dplyr`

---

## duplicated_data *Find Duplicated Data in a Track Table*

---

### Description

This function attempts to automatically detect duplicated data in `track` tables.

### Usage

```
duplicated_data(x, type = "txy")
```

### Arguments

| | |
|---|---|
| `x` | `track` table as produced by the `track` function. |
| `type` | A character string or a vector of character strings indicating the type of duplications to look for. The strings can be any combination of `"t"` (for time duplications) and `"x"`, `"y"`, `"z"` (for coordinate duplications). For instance, the string `"txy"` will return data with duplicated time stamps and duplicated x and y coordinates. |

### Value

A track table of all observations that are duplicated, as per the duplication rule defined by `type`.

### Note

Incomplete data (that is, data containing "NAs") are ignored.

### Author(s)

Simon Garnier, <garnier@njit.edu>

## See Also

track

## Examples

```
# Create data set with duplicated data
data(short_tracks)
t_df <- track(x = short_tracks$x, y = short_tracks$y, t = short_tracks$t,
              id = short_tracks$id, proj = "+proj=longlat",
              tz = "Africa/Windhoek", table = "df")
t_df <- bind_tracks(t_df, t_df[1:10, ], t_df[100:110, ])

# Find duplicated timestamps
duplicated <- duplicated_data(t_df)
```

---

# filter *Subset rows using column values*

---

## Description

This is a re-export of the `filter` function in the `dplyr` package. See the help

## Arguments

| | |
|---|---|
| .data | See `filter` for an explanation. |
| ... | Other args |

---

# inconsistent_data *Find Inconsistent Locations in a Track Table*

---

## Description

This function attempts to automatically detect inconsistent locations (for instance due to a writing error or GPS inaccuracies) in `track` tables.

## Usage

```
inconsistent_data(x, s = 15)
```

## Arguments

| | |
|---|---|
| x | `track` table as produced by the `track` function. |
| s | The discrimination threshold for the outlier detection algorithm. Higher values correspond to less outliers. |

## Value

A track table of all inconsistent data.

## Author(s)

Simon Garnier, <garnier@njit.edu>

## See Also

track

## Examples

```
# Create data set with inconsistent data
data(tracks)
t_df <- track(x = tracks$x, y = tracks$y, t = tracks$t,
              id = tracks$id, proj = "+proj=longlat",
              tz = "Africa/Windhoek", table = "df")
t_df$x[1000] <- t_df$x[1000] * 1.0001
t_df$y[4000] <- t_df$y[4000] * 1.00001

# Find inconsistent data
inconsistent <- inconsistent_data(t_df)
```

---

# is_geo *Check if Track Table Uses Geographic Coordinates*

---

## Description

Track tables produced by track_df can use a cartesian (x, y, z) or a geographic (latitude, longitude, altitude) coordinate system. This function helps determine which is being used in a particular table.

## Usage

```
is_geo(x)
```

## Arguments

| | |
|---|---|
| x | A track data table as produced by track_df. |

## Value

A logical.

## Author(s)

Simon Garnier, <garnier@njit.edu>

## Examples

```
data(short_tracks)

is_geo(short_tracks)
```

---

`is_track` *Check Validity of Track Table*

---

## Description

Test whether a variable contains a track table as produced by `track_df`, `track_tbl`, or `track_dt`.

## Usage

```
is_track(x)
```

## Arguments

| | |
|---|---|
| `x` | An object to test. |

## Value

A logical indicating whether the variable contains a track table (`TRUE`) or not (`FALSE`).

## Author(s)

Simon Garnier, <garnier@njit.edu>

## See Also

`track_df`, `track_tbl`, `track_dt`

## Examples

```
data(short_tracks)

is_track(short_tracks)
```

missing_data *Find Missing Data in a Track Table*

## Description

This function attempts to automatically detect missing data (for instance due to writing errors) in track tables.

## Usage

```
missing_data(x, begin = NULL, end = NULL, step = NULL)
```

## Arguments

| | |
|---|---|
| x | A track table as produced by the track function. |
| begin | A full time stamp (date+time) in POSIXct format corresponding to the time from which the missing data should be looked for. If not set, the first time stamp of the track table will be used. |
| end | A full time stamp (date+time) in POSIXct format corresponding to the time until which the missing data should be looked for. If not set, the last time stamp of the track table will be used. |
| step | A difftime object representing the expected time between two consecutive locations of the trajectory. If not set, it is set to the most common time difference between successive locations in x. |

## Value

A track table of all observations with missing data. The missing data is indicated with NA.

## Author(s)

Simon Garnier, <garnier@njit.edu>

## See Also

track

## Examples

```
# Create data set with missing time stamps
data(short_tracks)
t_df <- track(x = short_tracks$x, y = short_tracks$y, t = short_tracks$t,
              id = short_tracks$id, proj = "+proj=longlat",
              tz = "Africa/Windhoek", table = "df")
t_df <- t_df[-c(10, 100), ]

# Find missing data
missing <- missing_data(t_df)
```

## n_dims *Number of Spatial Dimensions of a Track Table*

### Description

Track tables produced by track_df can have 2 (x,y) or 3 (x, y, z) spatial dimensions. This function returns the number of spatial dimensions of a track table.

### Usage

```
n_dims(x)
```

### Arguments

| | |
|---|---|
| x | A track data table as produced by track_df. |

### Value

A numeric value.

### Author(s)

Simon Garnier, <garnier@njit.edu>

### Examples

```
data(short_tracks)

n_dims(short_tracks)
```

## n_tracks *Number of Tracks in a Track Table*

### Description

Track tables produced by track_df can contain multiple tracks (e.g., from different animals). This function returns the number of tracks in a track table.

### Usage

```
n_tracks(x)
```

### Arguments

| | |
|---|---|
| x | A track data table as produced by track_df. |

## Value

A numeric value.

## Author(s)

Simon Garnier, <garnier@njit.edu>

## Examples

```
data(short_tracks)

n_tracks(short_tracks)
```

---

`projection` *Access/Modify the Projection of a Track Table*

---

## Description

Functions to access or modify the projection of a data table. Changing the projection will trigger automatically the conversion of the locations in the new coordinate system.

## Usage

```
projection(x)

projection(x) <- value

project(x, value)
```

## Arguments

| | |
|---|---|
| `x` | A track table. |
| `value` | A character string or a `terra::crs` object representing the projection of the coordinates. `"+proj=longlat"` is suitable for the outputs of most GPS trackers. |

## Value

A track table.

## Note

It is not possible to modify the projection if missing coordinates are present.

## Author(s)

Simon Garnier, <garnier@njit.edu>

## See Also

st_crs

## Examples

```
data(short_tracks)

projection(short_tracks)
tracks_projected <- project(short_tracks, "+proj=somerc")
projection(tracks_projected)
projection(tracks_projected) <- "+proj=longlat"
projection(tracks_projected)
```

---

# short_tracks *Trajectories of Two Goats Through the Namibian Desert (short version)*

---

## Description

A dataset containing the trajectories of two goats through the Namibian desert.

## Usage

```
short_tracks
```

## Format

A track table with 18 rows and 4 variables:

**id** Identity of the goat

**t** Time of the observation

**x** Longitude

**y** Latitude

## tracks *Trajectories of Two Goats Through the Namibian Desert*

### Description

A dataset containing the trajectories of two goats through the Namibian desert.

### Usage

```
tracks
```

### Format

A track table with 7194 rows and 4 variables:

**id** Identity of the goat

**t** Time of the observation

**x** Longitude

**y** Latitude

## track_ *Build a Track Table*

### Description

`track` constructs track tables based on `data.frame` (the default), `tibble`, or `data.table`. `track` is a convenience function that executes `track_df`, `track_tbl`, or `track_dt` based on the value of the 'table' parameter. Track tables can be used like the data structure they are build upon but with a notable difference: they have an extra attribute to store the projection of the track coordinates, and modifying the projection will automatically trigger the appropriate conversion of the coordinates.

### Usage

```
track(x, y, z, t, id, ..., proj, origin, period, tz, format, table = "df")

track_df(x, y, z, t, id, ..., proj, origin, period, tz, format)

track_tbl(x, y, z, t, id, ..., proj, origin, period, tz, format)

track_dt(x, y, z, t, id, ..., proj, origin, period, tz, format)
```

## Arguments

| | |
|---|---|
| x, y, z | Numeric vectors representing the coordinates of the locations. x and y are required. z can be ignored if the trajectories are 2-dimensional. Note: if the vectors are not of the same length, the shorter ones will be recycled to match the length of the longer one. |
| t | A numeric vector or a vector of objects that can be coerced to date-time objects by as_datetime representing the times (or frames) at which each location was recorded. If numeric, the origin and period of the time points can be set using origin and period below. |
| id | A vector that can be coerced to a character vector by as.character representing the identity of the individual to which each location belong. |
| ... | A set of name-value pairs. Arguments are evaluated sequentially, so you can refer to previously created elements. These arguments are processed with rlang::quos() and support unquote via !! and unquote-splice via !!!. Use := to create columns that start with a dot. |
| proj | A character string or a crs object (see st_crs for more information) representing the projection of the coordinates. Leave empty if the coordinates are not projected (e.g., output of video tracking). "+proj=longlat" is suitable for the output of most GPS trackers. |
| origin | Something that can be coerced to a date-time object by as_datetime representing the start date and time of the observations when t is a numeric vector. |
| period | A character vector in a shorthand format (e.g. "1 second") or ISO 8601 specification. This is used when t is a numeric vector to represent time unit of the observations. All unambiguous name units and abbreviations are supported, "m" stands for months, "M" for minutes unless ISO 8601 "P" modifier is present (see examples). Fractional units are supported but the fractional part is always converted to seconds. See period for more details. |
| tz | A time zone name. See OlsonNames. |
| format | A character string indicating the formatting of 't'. See strptime for how to specify this parameter.<br>When supplied parsing is performed by strptime(). For this reason consider using specialized parsing functions in lubridate. |
| table | A string indicating the class of the table on which the track table should be built. It can be a data.frame ("df", the default), a tibble ("tbl"), or a data.table ("dt"). |

## Value

A track table, which is a colloquial term for an object of class track.

## Author(s)

Simon Garnier, <garnier@njit.edu>

## Examples

```
data(short_tracks)

t_df <- track(x = short_tracks$x, y = short_tracks$y, t = short_tracks$t,
  id = short_tracks$id, proj = "+proj=longlat", tz = "Africa/Windhoek", table = "df")

t_df <- track_df(x = short_tracks$x, y = short_tracks$y, t = short_tracks$t,
  id = short_tracks$id, proj = "+proj=longlat", tz = "Africa/Windhoek")

t_tbl <- track_tbl(x = short_tracks$x, y = short_tracks$y, t = short_tracks$t,
id = short_tracks$id, proj = "+proj=longlat", tz = "Africa/Windhoek")

t_dt <- track_dt(x = short_tracks$x, y = short_tracks$y, t = short_tracks$t,
  id = short_tracks$id, proj = "+proj=longlat", tz = "Africa/Windhoek")
```

---

# [.track *Extract or Replace Parts of a Track Table*

---

## Description

Accessing columns, rows, or cells via $, [[, or [ is mostly similar to regular `data frames`. However, the behavior is sometimes different for track tables based on `tibble` and `data.table`. For more info, refer to tibble's and `data.table`'s subsetting documentation.

## Usage

```
## S3 method for class 'track'
x[...]

## S3 replacement method for class 'track'
x[...] <- value
```

## Arguments

| | |
|---|---|
| `x` | A track table. |
| `...` | Other parameters to be passed to the extracting/subsetting functions of `data.frame`, `tibble`, and `data.table`. |
| `value` | A suitable replacement value: it will be repeated a whole number of times if necessary and it may be coerced: see the 'Coercion' section in `data.frame`. If 'NULL', deletes the column if a single column is selected. |

## Value

A subset of the track table is `[` is called, or a modified version of the track table if `[<-` is called.

## Author(s)

Simon Garnier, <garnier@njit.edu>

## See Also

track_df, track_tbl, track_dt

## Examples

```
data(short_tracks)

short_tracks[1]
short_tracks[1, ]
short_tracks[1, 1]
short_tracks$id[short_tracks$id == "1"] <- "0"
short_tracks[short_tracks[, 1] == "0", 1] <- "1"
```

# Index


∗ **datasets**
    short_tracks, 15
    tracks, 16
.Mode, 2
.reclass, 3
[.track, 18
[<-.track *([.track)*, 18

adehabitatLT::as.ltraj, *5*
anti_join.track *(*dplyr_track*)*, 6
arrange.track *(*dplyr_track*)*, 6
as.character, *17*
as_datetime, *17*
as_ltraj *(*conversions*)*, 4
as_move *(*conversions*)*, 4
as_moveHMM *(*conversions*)*, 4
as_sp *(*conversions*)*, 4
as_telemetry *(*conversions*)*, 4
as_track *(*conversions*)*, 4

bind_tracks, 3

conversions, 4
ctmm::as.telemetry, *5*

data.frame, *5*, *16–18*
data.table, *5*, *16–18*
data.table::rbindlist, *3*
difftime, *12*
distinct.track *(*dplyr_track*)*, 6
do.track *(*dplyr_track*)*, 6
dplyr, *6*, *8, 9*
dplyr_track, 6
duplicated_data, 8

filter, *9*, 9
filter.track *(*dplyr_track*)*, 6
full_join.track *(*dplyr_track*)*, 6

group_by.track *(*dplyr_track*)*, 6

inconsistent_data, 9
inner_join.track *(*dplyr_track*)*, 6
is_geo, 10
is_track, 11

left_join.track *(*dplyr_track*)*, 6

missing_data, 12
moveHMM::moveData, *5*
moveHMM::prepData, *5*
mutate.track *(*dplyr_track*)*, 6

n_dims, 13
n_tracks, 13
NA, *12*
nest_join.track *(*dplyr_track*)*, 6

OlsonNames, *17*

period, *17*
POSIXct, *12*
project *(*projection*)*, 14
projection, 14
projection<- *(*projection*)*, 14

rename.track *(*dplyr_track*)*, 6
right_join.track *(*dplyr_track*)*, 6
rlang::quos(), *17*
rowwise.track *(*dplyr_track*)*, 6

select.track *(*dplyr_track*)*, 6
semi_join.track *(*dplyr_track*)*, 6
short_tracks, 15
slice.track *(*dplyr_track*)*, 6
slice_sample.track *(*dplyr_track*)*, 6
sp::SpatialPointsDataFrame, *5*
st_crs, *15*, *17*
strptime, *17*
summarise.track *(*dplyr_track*)*, 6
summarize.track *(*dplyr_track*)*, 6

terra::crs, *14*
tibble, *5*, *16–18*
track, *8–10*, *12*
track *(*track_*)*, 16
track_, 16
track_df, *5*, *10, 11*, *13*, *19*
track_df *(*track_*)*, 16
track_dt, *5*, *11*, *19*
track_dt *(*track_*)*, 16
track_tbl, *5*, *11*, *19*
track_tbl *(*track_*)*, 16
tracks, 16
transmute.track *(*dplyr_track*)*, 6

ungroup.track *(*dplyr_track*)*, 6