# Package ‘trajectories’

## August 29, 2016

**Version** 0.1-4  
**Date** 2015-08-19  
**Title** Classes and Methods for Trajectory Data  
**Depends** R (>= 3.0.0)  
**Imports** stats, utils, graphics, methods, lattice, sp (>= 1.1-0), spacetime (>= 1.0-0)  
**Suggests** rgdal, rgeos, rgl, OpenStreetMap, RCurl, rjson, adehabitatLT, xts  
**LazyData** no  
**Description** Classes and methods for trajectory data, with nested classes for individual trips, and collections for different entities. Methods include selection, generalization, aggregation, intersection, and plotting.  
**License** GPL (>= 2)  
**URL** [http://github.com/edzer/trajectories](http://github.com/edzer/trajectories)  
**BugReports** [http://github.com/edzer/trajectories/issues](http://github.com/edzer/trajectories/issues)  
**NeedsCompilation** no  
**Author** Edzer Pebesma [cre, aut], Benedikt Klus [aut], Benedikt Graeler [ctb], Nikolai Gorte [ctb]  
**Maintainer** Edzer Pebesma <edzer.pebesma@uni-muenster.de>  
**Repository** CRAN  
**Date/Publication** 2015-08-19 18:38:47

## R topics documented:

- A3 .......................................................... 2  
- compare .................................................. 3
Description

Trajectory, locally stored, from envirocar.org, see example below how it was imported

Usage

data(A3)

Examples

library(spacetime)
data(A3)
dim(A3)
## Not run:
importEnviroCar = function(trackID, url = "https://envirocar.org/api/stable/tracks/") {
require(RCurl)
require(rgdal)
require(rjson)
require(sp)
url = getURL(paste(url, trackID, sep = ""),
.opts = list(ssl.VERIFYpeer = FALSE)) # .opts needed for Windows
# Read data into spatial object.
spdf = readOGR(dsn = url, layer = "OGRGeoJSON", verbose = FALSE)
# Convert time from factor to POSIXct.
time = as.POSIXct(spdf$time, format ="
# Convert phenomena from JSON to data frame.
phenomena = lapply(as.character(spdf$phenomenons), fromJSON)
values = lapply(phenomena, function(x) as.data.frame(lapply(x, function(y) y$value)))
# Get a list of all phenomena for which values exist.
names = vector()
for(i in names)
names = union(names, names(i))
# Make sure that each data frame has the same number of columns.
values = lapply(values, function(x) {
xNames = names(x)
compare

# Get the symmetric difference.
diff = setdiff(union(names, xNames), intersect(names, xNames))
if(length(diff) > 0)
x[diff] = NA
x
#
# Bind values together.
data = do.call(rbind, values)
sp = SpatialPoints(coords = coordinates(spdf),
proj4string = CRS("+proj=longlat +ellps=WGS84"))
stidf = STIDF(sp = sp, time = time, data = data)
Track(track = stidf)
}  
A3 = importEnviroCar("528cf1a3e4b0a727145df093")

## End(Not run)

---

**compare**  
*Compares objects of class Track*

**Description**

Calculates distances between two tracks for the overlapping time interval.

**Usage**

```r
## S4 method for signature 'Track'
compare(tr1, tr2)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tr1</td>
<td>An object of class Track.</td>
</tr>
<tr>
<td>tr2</td>
<td>An object of class Track.</td>
</tr>
</tbody>
</table>

**Value**

A `difftrack` object. Includes both tracks extended with additional points for the timestamps of the other track. Also includes `SpatialLines` representing the distances between the tracks.

**Author(s)**

Nikolai Gorte <n.gorte@gmail.com>
Examples

```r
## example tracks
library(sp)
library(xts)
data(A3)
track2 <- A3
index(track2@time) <- index(track2@time) + 32
track2@sp@coords <- track2@sp@coords + 0.003

## compare and plot
difftrack <- compare(A3, track2)
plot(difftrack)
```

Description

Class that represents differences between two Track objects.

Objects from the Class

Objects can be created by calls of the form `new("difftrack", ...)`. Objects of class `difftrack` contain 2 objects of class Track extended with points for timestamps of the other track and 2 SpatialLinesDataFrame containing the the lines and distances between tracks.

Slots

- `track1`: Extended track1
- `track2`: Extended track2
- `conns1`: Lines between the original track1 and the new points on track2
- `conns2`: Lines between the original track2 and the new points on track1

Methods

- `plot` signature(x = "difftrack", y = "missing"): plot a difftrack

Author(s)

Nikolai Gorte <n.gorte@gmail.com>
dists

Examples

```r
showClass("differtrack")
## example tracks
library(sp)
library(xts)
data(A3)
track2 <- A3
index(track2@time) <- index(track2@time) + 32
track2@sp@coords <- track2@sp@coords + 0.003

## compare and plot
difftrack <- compare(A3, track2)
plot(difftrack)

## space-time cube of the differtrack
## Not run:
stcube(difftrack)

## End(Not run)
```

---

**dists**

*Calculate distances between two Tracks objects*

**Description**

Calculates a distance matrix with distances for each pair of tracks.

**Usage**

```r
## S4 method for signature 'Tracks'
dists(tr1, tr2, f, ...)
```

**Arguments**

- `tr1`: An object of class Tracks.
- `tr2`: An object of class Tracks.
- `f`: A function to calculate distances. Default is `mean`.
- `...`: Additional parameters passed to `f`.

**Details**

`f` can be any function applicable to a numerical vector or `frechetDist`.

**Value**

A matrix with distances between each pair of tracks or `NA` if they don’t overlap in time.
Examples

```r
## example tracks
library(sp)
library(xts)
data(A3)
track2 <- A3
index(track2@time) <- index(track2@time) + 32
track2@sp@coords <- track2@sp@coords + 0.003

## create Tracks objects
tracks1 <- Tracks(list(A3, track2))
tracks2 <- Tracks(list(track2, A3))

## calculate distances
## Not run:
dists(tracks1, tracks2)
dists(tracks1, tracks2, sum)
dists(tracks1, tracks2, frechetDist)

## End(Not run)
```

__ dowmsample__

__Downsample a Track__

Description

Downsamples a Track to the size (amount of points) of another Track.

Usage

```r
## S4 method for signature 'Track'
downsample(track1, track2)
```

Arguments

- `track1` : Track that will be downsampled.
- `track2` : Reference Track.

Value

A Track object. The downsampled `track1`.

Author(s)

Nikolai Gorte <n.gorte@gmail.com>
**frechetDist**

### Description
Calculates the discrete Frechet distance between two Track objects.

### Usage

```r
## S4 method for signature 'Track'
frechetDist(track1, track2)
```

### Arguments

- `track1`: An object of class Track.
- `track2`: An object of class Track.

### Value
Discrete Frechet distance.

### Author(s)
Nikolai Gorte <n.gorte@gmail.com>

### References
http://en.wikipedia.org/wiki/Fréchet_distance

---

**generalize**

### Description
Generalize objects of class Track, Tracks and TracksCollection.

### Usage

```r
## S4 method for signature 'Track'
generalize(t, FUN = mean, ..., timeInterval, distance, n, tol, toPoints)
## S4 method for signature 'Tracks'
generalize(t, FUN = mean, ...)  
## S4 method for signature 'TracksCollection'
generalize(t, FUN = mean, ...)
```
Arguments

- **t**: An object of class Track, Tracks or TracksCollection.
- **FUN**: The generalization method to be applied. Defaults to *mean* if none is passed.
- **timeInterval**: (lower limit) time interval to split Track into segments
- **distance**: (lower limit) distance to split Track into segments
- **n**: number of points to form segments
- **tol**: tolerance passed on to *gSimplify*, to generalize segments using the Douglas-Peucker algorithm.
- **toPoints**: keep mid point rather than forming *SpatialLines* segments
- ... Additional arguments passed to FUN

Value

An object of class Track, Tracks or TracksCollection.

---

**stbox**  
*obtain ranges of space and time coordinates*

Description

obtain ranges of space and time coordinates

Usage

```
stbox(obj)
```

Arguments

- **obj**: object of a class deriving from Tracks or TracksCollection.

Value

*stbox* returns a `data.frame`, with three columns representing x-, y- and time-coordinates, and two rows containing min and max values. *bbox* gives a matrix with coordinate min/max values, compatible to *bbox*

Methods

```
stbox signature(x = "Tracks") : obtain st range from object
stbox signature(x = "TracksCollection") : obtain st range from object
```
Draw a space-time cube.

**Description**

Draw a space-time cube.

**Usage**

```r
## S4 method for signature 'Track'
stcube(x, xlab = "x", ylab = "y", zlab = "t", type = "l",
       aspect, xlim = stbox(x)[[1]], ylim = stbox(x)[[2]],
       zlim = stbox(x)$time, showMap = FALSE, mapType = "osm",
       mapZoom = NULL, ..., y, z)
```

```r
## S4 method for signature 'Tracks'
stcube(x, xlab = "x", ylab = "y", zlab = "t", type = "l",
       aspect, xlim, ylim, zlim, showMap = FALSE, mapType = "osm",
       normalizeBy = "week", mapZoom = NULL, ..., y, z, col)
```

```r
## S4 method for signature 'TracksCollection'
stcube(x, xlab = "x", ylab = "y", zlab = "t",
       type = "l", aspect, xlim, ylim, zlim, showMap = FALSE, mapType = "osm",
       normalizeBy = "week", mapZoom = NULL, ..., y, z, col)
```

```r
## S4 method for signature 'difftrack'
stcube(x, showMap = FALSE, mapType = "osm", normalizeBy = "week", ..., y, z)
```

**Arguments**

- `x` An object of class `Track`, `Tracks`, or `TracksCollection` or `difftrack`.
- `xlab`, `ylab`, `zlab`, `type`, `aspect`, `xlim`, `ylim`, `zlim` Arguments passed to `plot3d()` of package `rgl`.
- `showMap` Flag if a basemap is to be shown on the xy plane; for this to function, you may need to load library `raster` first, see also the `stcube` demo script.
- `mapType` The tile server from which to get the map. Passed as `type` to `openmap()` of package `OpenStreetMap`.
- `normalizeBy` An abstract time period (either `week` or `day`) to be normalized by.
- `mapZoom` Set a zoom level for the map used as background. Null will use the `osm` package default strategie.
- `y`, `z`, `col` Ignored, but included in the method signature for implementation reasons.
- `...` Additional arguments passed to `plot3d()` of package `rgl`.

**Value**

A space-time cube.
Track-class

Examples

```r
## Not run: demo(stcube)
```

- `storms` *Storm trajectories*

Description


Usage

```r
data(storms)
```

Examples

```r
data(storms)
dim(storms)
plot(storms)
x = approxTracksCollection(storms, by = "30 min", FUN = spline)
plot(x, col = 'red', add = TRUE)
## Not run:
demo(storms) # regenerates these data from their source
## End(Not run)
```

Track-class *Classes "Track", "Tracks", and "TracksCollection"*

Description

Classes for representing sets of trajectory data, with attributes, for different IDs (persons, objects, etc)

Usage

```r
Track(track, df = fn(track), fn = TrackStats)
Tracks(webs, tracksData = data.frame(row.names=names(webs)),
      fn = TrackSummary)
TracksCollection(tracksCollection, tracksCollectionData,
                 fn = TracksSummary)
TrackStats(track)
TrackSummary(track)
TracksSummary(tracksCollection)
## S4 method for signature 'Track'
x[i, j, ..., drop = TRUE]
```
## S4 method for signature 'TracksCollection'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'Track, data.frame'
coerce(from, to)

## S4 method for signature 'Tracks, data.frame'
coerce(from, to)

## S4 method for signature 'TracksCollection, data.frame'
coerce(from, to)

### Arguments

- **track**: object of class \texttt{STIDF-class}, representing a single trip
- **df**: optional \texttt{data.frame} with information between track points
- **tracks**: named \texttt{data.frame} with \texttt{Track} objects
- **tracksData**: \texttt{data.frame} with summary data for each \texttt{Track}
- **tracksCollection**: \texttt{list}, with \texttt{Tracks} objects
- **tracksCollectionData**: \texttt{data.frame}, with summary data on \texttt{tracksCollection}
- **fn**: function
- **x**: object of class \texttt{Track} etc
- **i**: selection of spatial entities
- **j**: selection of temporal entities (see syntax in package \texttt{xts})
- **...**: selection of attribute(s)
- **drop**: \texttt{logical}
- **from**: \texttt{from}
- **to**: target class

### Value

Functions \texttt{Track}, \texttt{Tracks} and \texttt{TracksCollection} are constructor functions that take the slots as arguments, check object validity, and compute summary statistics on the track and tracks sets.

\texttt{TrackStats} returns a \texttt{data.frame} with for each track segment the distance, duration, speed, and direction. In case data are geographical coordinates (long/lat), distance is in m, and direction is initial bearing.

\texttt{TrackSummary} reports for each track xmin, xmax, ymin, ymax, tmin, tmax, (number of points) n, (total) distance, and medspeed (median speed).

\texttt{TracksSummary} reports for each Tracks of a \texttt{TracksCollection} (number of tracks) n, xmin, xmax, ymin, ymax, tmin, tmax.
Objects from the Class

Objects of class Track extend STIDF-class and contain single trips or tracks, objects of class Tracks contain multiple Track objects for a single ID (person, object or tracking device), objects of class TracksCollection contain multiple Tracks objects for different IDs.

Slots of class "Track"

sp: spatial locations of the track points, with length n
time: time stamps of the track points
endTime: end time stamps of the track points
data: data.frame with n rows, containing attributes of the track points
connections: data.frame, with n-1 rows, containing attributes between the track points such as distance and speed

Slots of class "Tracks"

tracks: list with Track objects, of length m
tracksData: data.frame with m rows, containing summary data for each Track object

Slots of class "TracksCollection"

tracksCollection: list Tracks objects, of length p
tracksCollectionData: data.frame with p rows, containing summary data for each Tracks object

Methods

[[ signature(obj = "Track"): retrieves the attribute element
[[ signature(obj = "Tracks"): retrieves the attribute element
[[ signature(obj = "TracksCollection"): retrieves the attribute element
[[< signature(obj = "Track"): sets or replaces the attribute element
[[< signature(obj = "Tracks"): sets or replaces the attribute element
[[< signature(obj = "TracksCollection"): sets or replaces the attribute element
$ signature(obj = "Track"): retrieves the attribute element
$ signature(obj = "Tracks"): retrieves the attribute element
$ signature(obj = "TracksCollection"): retrieves the attribute element
$< signature(obj = "Track"): sets or replaces the attribute element
$< signature(obj = "Tracks"): sets or replaces the attribute element
$< signature(obj = "TracksCollection"): sets or replaces the attribute element
coerce Track, data.frame coerce to data.frame
coerce Tracks, data.frame coerce to data.frame
coerce TracksCollection, data.frame coerce to data.frame
plot signature(x = "TracksCollection", y = "missing"): plots sets of sets of tracks
stplot signature(obj = "TracksCollection"): plots sets of sets of tracks
**Note**

segments is a data frame form in which track segments instead of track points form a record, with x0, y0, x1 and y1 the start and end coordinates

**Author(s)**

Edzer Pebesma, edzer.pebesma@uni-muenster.de

**References**

http://www.jstatsoft.org/v51/i07/

**Examples**

```r
library(sp)
library(spacetime)
t0 = as.POSIXct(as.Date("2013-09-30", tz="CET"))
# person A, track 1:
x = c(7,6,5,4,3,3)
y = c(7,7,6,5,5,7)
n = length(x)
set.seed(131)
t = t0 + cumsum(runif(n) * 60)
require(rgdal)
crs = CRS("+proj=longlat +ellps=WGS84") # longlat
stdf = STIDF(SpatialPoints(cbind(x,y),crs), t, data.frame(co2 = rnorm(n)))
A1 = Track(stdf)
# person A, track 2:
x = c(7,6,6,7,7)
y = c(6,5,4,4,3)
n = length(x)
t = max(t) + cumsum(runif(n) * 60)
stdf = STIDF(SpatialPoints(cbind(x,y),crs), t, data.frame(co2 = rnorm(n)))
A2 = Track(stdf)
# Tracks for person A:
A = Tracks(list(A1=A1,A2=A2))
# person B, track 1:
x = c(2,2,1,1,2,3)
y = c(5,4,3,2,2,3)
n = length(x)
t = max(t) + cumsum(runif(n) * 60)
stdf = STIDF(SpatialPoints(cbind(x,y),crs), t, data.frame(co2 = rnorm(n)))
B1 = Track(stdf)
# person B, track 2:
x = c(3,3,4,3,3,4)
y = c(5,4,3,2,1,1)
n = length(x)
t = max(t) + cumsum(runif(n) * 60)
stdf = STIDF(SpatialPoints(cbind(x,y),crs), t, data.frame(co2 = rnorm(n)))
B2 = Track(stdf)
# Tracks for person A:
B = Tracks(list(B1=B1,B2=B2))
```
Tr = TracksCollection(list(A=A, B=B))
stplot(Tr, scales = list(draw=TRUE))

stplot(Tr, attr = "direction", arrows=TRUE, lwd = 3, by = "direction")
stplot(Tr, attr = "direction", arrows=TRUE, lwd = 3, by = "IDs")

plot(Tr, col=2, axes=TRUE)
dim(Tr)
dim(Tr[[2]])
dim(Tr[[2]][1])

u = stack(Tr) # four IDs
dim(u)

dim(unstack(u, c(1,1,2,2))) # regroups to original

dim(unstack(u, c(1,1,2,3))) # regroups to three IDs

dim(unstack(u, c(1,2,2,1))) # regroups differently

as(Tr, "data.frame")[1:10,] # tracks separated by NA rows

as(Tr, "segments")[1:10,] # track segments as records

Tr["distance"] = Tr["distance"] * 1000

Tr$distance = Tr$distance / 1000

Tr$distance

# work with custom TrackStats function:

MyStats = function(track) {
  df = apply(coordinates(track@sp), 2, diff) # requires sp
data.frame(distance = apply(df, 1, function(x) sqrt(sum(x^2))))
}

crs = CRS(as.character(NA))

stidf = STIDF(SpatialPoints(cbind(x,y), crs), t, data.frame(co2 = rnorm(n)))

B2 = Track(stidf) # no longer longlat;

B3 = Track(stidf, fn = MyStats)

all.equal(B3$distance, B2$distance)

# approxTrack:

opar = par()

par(mfrow = c(1, 2))

plot(B2, ylim = c(.5, 6))

plot(B2, pch = 16, add = TRUE)

title("irregular time steps")
i = index(B2)

B3 = approxTrack(B2, seq(min(i), max(i), length.out = 50))

plot(B3, col = 'red', type = 'p', add = TRUE)

B4 = approxTrack(B2, seq(min(i), max(i), length.out = 50), FUN = spline)

plot(B4, col = 'blue', type = 'b', add = TRUE)

# regular time steps:

t = max(t) + (1:n) * 60 # regular

B2 = Track(STIDF(SpatialPoints(cbind(x,y), crs), t, data.frame(co2 = rnorm(n))))

plot(B2, ylim = c(.5, 6))

plot(B2, pch = 16, add = TRUE)

title("constant time steps")
i = index(B2)

B3 = approxTrack(B2)

plot(B3, type = 'p', col = 'red', add = TRUE)

B4 = approxTrack(B2, FUN = spline)

plot(B4, type = 'p', col = 'blue', add = TRUE)

par(opar)

smth = function(x,y,xout,...) predict(smooth.spline(as.numeric(x), y), as.numeric(xout))
data(storms)
plot(storms, type = 'p')
storms.smooth = approxTracksCollection(storms, FUN = smth, n = 200)
plot(storms.smooth, add = TRUE, col = 'red')
Index

*Topic **classes**
  difftrack-class, 4
  Track-class, 10
*Topic **compare**
  compare, 3
*Topic **datasets**
  A3, 2
  storms, 10
*Topic **dists**
  dists, 5
*Topic **downsample**
  downsample, 6
*Topic **dplot**
  stbox, 8
*Topic **generalize**
  generalize, 7
*Topic **methods**
  frechetDist, 7
*Topic **space-time cube**
  stcube, 9
  [,Track-method (Track-class), 10
  [,Tracks-method (Track-class), 10
  [,TracksCollection-method (Track-class), 10
  [[,Track,ANY,missing-method (Track-class), 10
  [[,Tracks,ANY,missing-method (Track-class), 10
  [[,TracksCollection,ANY,missing-method (Track-class), 10
  [[<-,Track,ANY,missing-method (Track-class), 10
  [[<-,Tracks,ANY,missing-method (Track-class), 10
  [[<-,TracksCollection,ANY,missing-method (Track-class), 10
  $,Track-method (Track-class), 10
  $,Tracks-method (Track-class), 10
  $,TracksCollection-method (Track-class), 10
  $<-,Track-method (Track-class), 10
  $<-,Tracks-method (Track-class), 10
  $<-,TracksCollection-method (Track-class), 10
  A3, 2
  aggregate,Track-method (Track-class), 10
  aggregate,Tracks-method (Track-class), 10
  aggregate,TracksCollection-method (Track-class), 10
  approxTrack (Track-class), 10
  approxTracks (Track-class), 10
  approxTracksCollection (Track-class), 10
  bbox, 8
  coerce,Track,data.frame-method (Track-class), 10
  coerce,Tracks,data.frame-method (Track-class), 10
  coerce,TracksCollection,data.frame-method (Track-class), 10
  compare, 3
  compare,Track-method (compare), 3
  coordnames,Track-method (Track-class), 10
  coordnames,Tracks-method (Track-class), 10
  coordnames,TracksCollection-method (Track-class), 10
  diffttrack (diffttrack-class), 4
  diffttrack-class, 4
  dists, 5
  dists,Tracks-method (dists), 5
  downsample, 6
  downsample,Track-method (downsample), 6
  frechetDist, 5, 7
frechetDist, Track-method (frechetDist), 7

generate, 7
generate, Track-method (generate), 7
generate, Tracks-method (generate), 7
generate, TracksCollection-method
   (generate), 7
gSimplify, 8

plot, difftrack, ANY-method
   (difftrack-class), 4
plot, Track, missing-method
   (Track-class), 10
plot, Tracks, ANY-method (Track-class), 10
plot, TracksCollection, ANY-method
   (Track-class), 10

segments-class (Track-class), 10
segPanel (Track-class), 10
SpatialLines, 8
SpatialLinesDataFrame, 4
stbox, 8
stbox, Tracks-method (stbox), 8
stbox, TracksCollection-method (stbox), 8
stcube, 9
stcube, difftrack-method (stcube), 9
stcube, Track-method (stcube), 9
stcube, Tracks-method (stcube), 9
stcube, TracksCollection-method
   (stcube), 9
STIDF-class, 11, 12
storms, 10
stplot, TracksCollection-method
   (Track-class), 10

Track, 4
Track (Track-class), 10
Track-class, 10
Tracks (Track-class), 10
Tracks-class (Track-class), 10
TracksCollection (Track-class), 10
TracksCollection-class (Track-class), 10
tracksPanel (Track-class), 10
TracksSummary (Track-class), 10
TrackStats (Track-class), 10
TrackSummary (Track-class), 10