Package ‘trajectories’

June 15, 2020

Version 0.2-2
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), zoo
Suggests rgdal, rgeos, OpenStreetMap, RCurl, rjson, adehabitatLT, xts, knitr, rgl, forecast, MASS, spatstat, taxidata
Description Classes and methods for trajectory data, with support for nesting individual Track objects in track sets (Tracks) and track sets for different entities in collections of Tracks. Methods include selection, generalization, aggregation, intersection, simulation, and plotting.
License GPL (>= 2)
URL http://github.com/edzer/trajectories
Additional_repositories http://gis-bigdata.uni-muenster.de/pebesma/
BugReports http://github.com/edzer/trajectories/issues
VignetteBuilder knitr
RoxygenNote 6.0.1
NeedsCompilation no
Author Edzer Pebesma [aut, cre] (<https://orcid.org/0000-0001-8049-7069>), Benedikt Klus [aut], Benedikt Graeler [ctb], Nikolai Gorte [ctb], Mehdi Moradi [aut]
Maintainer Edzer Pebesma <edzer.pebesma@uni-muenster.de>
Repository CRAN
Date/Publication 2020-06-15 13:50:03 UTC
### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A3</td>
<td>3</td>
</tr>
<tr>
<td>as.list.Tracks</td>
<td>3</td>
</tr>
<tr>
<td>as.list.TracksCollection</td>
<td>4</td>
</tr>
<tr>
<td>as.Track</td>
<td>4</td>
</tr>
<tr>
<td>as.Track.arrow</td>
<td>5</td>
</tr>
<tr>
<td>as.Track.ppp</td>
<td>6</td>
</tr>
<tr>
<td>auto.arima.Track</td>
<td>7</td>
</tr>
<tr>
<td>avedistTrack</td>
<td>8</td>
</tr>
<tr>
<td>avemove</td>
<td>9</td>
</tr>
<tr>
<td>chimaps</td>
<td>10</td>
</tr>
<tr>
<td>compare</td>
<td>11</td>
</tr>
<tr>
<td>cut</td>
<td>12</td>
</tr>
<tr>
<td>density.list</td>
<td>13</td>
</tr>
<tr>
<td>difftrack-class</td>
<td>14</td>
</tr>
<tr>
<td>dists</td>
<td>15</td>
</tr>
<tr>
<td>downsample</td>
<td>16</td>
</tr>
<tr>
<td>frechetDist</td>
<td>16</td>
</tr>
<tr>
<td>generalize</td>
<td>17</td>
</tr>
<tr>
<td>Kinhom.Track</td>
<td>18</td>
</tr>
<tr>
<td>pcfinhom.Track</td>
<td>19</td>
</tr>
<tr>
<td>plot.arwlen</td>
<td>20</td>
</tr>
<tr>
<td>plot.distrack</td>
<td>21</td>
</tr>
<tr>
<td>plot.gTrack</td>
<td>21</td>
</tr>
<tr>
<td>plot.KTrack</td>
<td>22</td>
</tr>
<tr>
<td>print.ArimaTrack</td>
<td>23</td>
</tr>
<tr>
<td>print.arwlen</td>
<td>23</td>
</tr>
<tr>
<td>print.distrack</td>
<td>24</td>
</tr>
<tr>
<td>print.gTrack</td>
<td>25</td>
</tr>
<tr>
<td>print.KTrack</td>
<td>25</td>
</tr>
<tr>
<td>print.ppplist</td>
<td>26</td>
</tr>
<tr>
<td>print.Track</td>
<td>26</td>
</tr>
<tr>
<td>print.Tracks</td>
<td>27</td>
</tr>
<tr>
<td>print.TracksCollection</td>
<td>27</td>
</tr>
<tr>
<td>print.Trow</td>
<td>28</td>
</tr>
<tr>
<td>range.Track</td>
<td>28</td>
</tr>
<tr>
<td>reTrack</td>
<td>29</td>
</tr>
<tr>
<td>rTrack</td>
<td>30</td>
</tr>
<tr>
<td>stbox</td>
<td>31</td>
</tr>
<tr>
<td>stcube</td>
<td>32</td>
</tr>
<tr>
<td>storms</td>
<td>33</td>
</tr>
<tr>
<td>Track-class</td>
<td>34</td>
</tr>
<tr>
<td>Track.idw</td>
<td>38</td>
</tr>
<tr>
<td>tsqTracks</td>
<td>39</td>
</tr>
<tr>
<td>unique.Track</td>
<td>40</td>
</tr>
</tbody>
</table>

**Index** 42
A3 Trajectory

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

Usage
data(A3)

Examples
library(spacetime)
data(A3)
dim(A3)
# see demo(A3) to see how A3 was fetched, and created from the web service

as.list.Tracks as.list.Tracks

Description
Convert a "Tracks" object to a list of tracks

Usage
## S3 method for class 'Tracks'
as.list(x,...)

Arguments
x an object of class "Tracks"
... passed to arguments of as.list

Author(s)
Mohammad Mehdi Moradi <moradi@uji.es>

See Also
rTrack, rTracks, rTracksCollection, as.list

Examples
x <- rTracks()
as.list(x)
as.list.TracksCollection

\texttt{as.list(TracksCollection)}

\section*{Description}

Convert a "TracksCollection" object to a list of tracks

\section*{Usage}

\begin{verbatim}
## S3 method for class 'TracksCollection'
as.list(x,...)
\end{verbatim}

\section*{Arguments}

\begin{itemize}
  \item \texttt{x} an object of class "TracksCollection"
  \item \texttt{...} passed to arguments of \texttt{as.list}
\end{itemize}

\section*{Author(s)}

Mohammad Mehdi Moradi <moradi@uji.es>

\section*{See Also}

\texttt{rTrack, rTracks, rTracksCollection, as.list}

\section*{Examples}

\begin{verbatim}
x <- rTracksCollection()
as.list(x)
\end{verbatim}

as.Track

\textit{Converts data to an object of class "Track"}

\section*{Description}

Function as.Track accepts converts x,y coordinates and their corresponding time/date to an object of class Track. It can also accepts covariates for the corresponding locations, covariates must be a dataframe with some columns and length of each column is equal to length of x,y,t.

\section*{Usage}

\begin{verbatim}
as.Track(x,y,t,covariate)
\end{verbatim}
as.Track.arrow

Arguments

- x: x coordinate.
- y: y coordinate.
- t: corresponding time and date of x,y.
- covariate: additional information.

Details

An object of class "Track" can be created by some geographical locations and corresponding time/dates. Function as.Track converts locations and dates/times to an object of class "Track". time/date should be from class "POSIXct" "POSIXt". See example below.

Value

An object of class "Track".

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

See Also

Track, as.POSIXct

Examples

```r
x <- runif(10,0,1)
y <- runif(10,0,1)
date <- seq(as.POSIXct("2015-1-1 0:00"), as.POSIXct("2015-1-1 9:00"), by = "hour")
Z <- as.Track(x,y,date)
plot(Z)
```

Convert trajectory pattern to a list of marked point patterns

Description

Converting a list of Track objects to a list of marked point patterns. Each mark shows the length of movement.

Usage

```r
as.Track.arrow(X,timestamp,epsilon=epsilon)
```
Arguments

- **X**: A list of Track objects
- **timestamp**: based on secs, mins,...
- **epsilon**: (optional) movements with length less than epsilon are not considered in the calculation

Details

Converting a list of Track objects to a list of marked point patterns. Marks show the length of movement with respect to the previous location.

Value

A list of marked point patterns.

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

See Also

rTrack, as.Track.ppp

Examples

```r
X <- list()
for(i in 1:10){
  m <- matrix(c(0,10,0,10),nrow=2,byrow = TRUE)
  X[[i]] <- rTrack(bbox = m,transform = TRUE)
}
Y <- as.Track.arrow(X,timestamp="120 secs")
```

Description

This function converts a list of Tracks to a list of point patterns (class "ppp")

Usage

```
as.Track.ppp(X,timestamp)
```

Arguments

- **X**: a list of Track objects
- **timestamp**: based on secs, mins,...
auto.arima.Track

Details
as.Track.ppp converts a list of Track objects to a list of ppp objects.

Value
A list of point patterns, objects of class "ppp".

Author(s)
Mohammad Mehdi Moradi <moradi@uji.es>

See Also
avedistTrack, as.ppp

Examples
X <- list()
for(i in 1:10){
  m <- matrix(c(0,10,0,10),nrow=2,byrow = TRUE)
  X[[i]] <- rTrack(bbox = m, transform = TRUE)
}
Y <- as.Track.ppp(X, timestamp="120 secs")

auto.arima.Track  Fitting arima model to a track

Description
Fit arima models to objects of class "Track".

Usage
auto.arima.Track(X, ...)

Arguments
X an object of class "Track"

Details
This fit arima models to the x,y locations of objects of class "Track".

Value
an object of class "ArimaTrack"
avedistTrack

**Author(s)**
Mohammad Mehdi Moradi <moradi@uji.es>

**See Also**
*rTrack, auto.arima*

**Examples**
```r
X <- rTrack()
auto.arima.Track(X)
```

---

**avedistTrack**

*Average pairwise distance of trajectory pattern over time*

**Description**
This measures the average of pairwise distances between tracks over time.

**Usage**
```r
avedistTrack(X, timestamp)
```

**Arguments**

- `X`:
  a list of some objects of class "Track"

- `timestamp`:
  timestamp to calculate the pairwise distances between tracks

**Details**
This function calculates the average pairwise distance between a list of tracks according to a given timestamp.

**Value**
An object of class "distrack". It can be plotted over time.

**Author(s)**
Mohammad Mehdi Moradi <moradi@uji.es>

**See Also**
*as.Track.ppp*
Examples

```r
X <- list()
for(i in 1:10){
  m <- matrix(c(0,10,0,10),nrow=2,byrow = TRUE)
  X[[i]] <- rTrack(bbox = m,transform = TRUE)
}

ave <- avedistTrack(X,timestamp = "120 secs")
plot(ave,type="l")
```

Description

This returns the average movements of a list of objects of class "Track" over time.

Usage

`avemove(X,timestamp,epsilon=epsilon)`

Arguments

- `X`: a list of some objects of class Track
- `timestamp`: timestamp to calculate the pairwise distances between tracks
- `epsilon`: (optional) movements with length less than epsilon are not considered in the calculation

Details

When analysing a list of tracks, avemove calculate the average of movements based on given timestamp.

Value

An object of class "numeric" or "arwlen".

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

See Also

`as.Track.arrow`
chimaps

Examples

```r
X <- list()
for(i in 1:10){
  m <- matrix(c(0,10,0,10),nrow=2,byrow = TRUE)
  X[[i]] <- rTrack(bbox = m,transform = TRUE)
}
avemove(X,timestamp = "30 secs")
```

chimaps

Chimaps of trajectory pattern.

Description

Computes the chimaps, corresponding to a list of objects of class "Track". chimaps are based on the discrepancy between computed and expected intensity in a given location.

Usage

```r
chimaps(X,timestamp,rank,...)
```

Arguments

- `X`: A list of Track objects
- `timestamp`: based on secs,mins,...
- `rank`: a number between one and the length of corresponding time sequence which is created based on given timestamp.
- `...`: passed to arguments of density.Track

Details

\[
\frac{\text{estimated intensity} - \text{expected intensity}}{\sqrt{\text{expected intensity}}}
\]

Value

an image of class "im".

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

See Also

density.list, density.ppp
Examples

```r
X <- list()
for(i in 1:10){
  m <- matrix(c(0,10,0,10),nrow=2,byrow = TRUE)
  X[[i]] <- rTrack(bbox = m,transform = TRUE)
}
chimaps(X, timestamp = "180 secs",rank = 2)
```

---

**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**

- `tr1` An object of class `Track`.
- `tr2` An object of class `Track`.

**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)
```
cut

obtain ranges of space and time coordinates

Description

obtain ranges of space and time coordinates

Usage

## S3 method for class 'Track'
cut(x, breaks, ..., include.lowest = TRUE, touch = TRUE)
## S3 method for class 'Tracks'
cut(x, breaks, ...)
## S3 method for class 'TracksCollection'
cut(x, breaks, ...)

Arguments

x           object of class Track, Tracks or TracksCollection
breaks      define the breaks; see cut
...          passed down to Tracks and Track methods, then to cut
include.lowest see cut
touch       logical; if FALSE, Track objects will be formed from unique sets of points,
            meaning that gaps between two consecutive Track objects will arise; if FALSE,
            the first point from each next track is copied, meaning that sets of Track are
            seamless.

Details

sub-trajectories can be invalid, if they have only one point, and are ignored. This can happen at the
start only if touch=FALSE, and at the end in any case.

Value

The cut method applied to a Track object cuts the track in pieces, and hence returns a Tracks ob-
ject. cut.Tracks returns a Tracks object, cut.TracksCollection returns a TracksCollection.

Examples

data(storms)
dim(storms)
dim(cut(storms, "week", touches = FALSE))  # same number of geometries
dim(cut(storms, "week"))  # increase of geometries = increase of tracks
Kernel estimate of intensity of trajectory pattern

Description
Estimating the intensity of a list of tracks.

Usage

```r
## S3 method for class 'list'
density(x, timestamp, ...)
```

Arguments
- `x`: a list of "Track" objects, an object of class "Tracks" or "TracksCollection"
- `timestamp`: based on secs, mins, ...
- `...`: passed to arguments of density.ppp

Details
This estimate the average intensity function of moving objects over time. Bandwidth selection methods such as bw.diggle, bw.scott and bw.ppl can be passed to this density.list.

Value
an image of class "im".

Author(s)
Mohammad Mehdi Moradi <moradi@uji.es>

See Also
- `rTrack`, `density.ppp`

Examples

```r
X <- list()
for(i in 1:10){
  m <- matrix(c(0,10,0,10),nrow=2,byrow = TRUE)
  X[[i]] <- rTrack(bbox = m,transform = TRUE)
}
density(X, timestamp = "180 secs")
```
Class “diftrack”

Description

Class that represents differences between two Track objects.

Objects from the Class

Objects can be created by calls of the form new("diftrack",...). Objects of class diftrack 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 = "diftrack",y = "missing"): plot a diftrack

Author(s)

Nikolai Gorte <n.gorte@gmail.com>

Examples

showClass("diftrack")
## 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 diftrack
## Not run:
stcube(difftrack)

## End(Not run)
Description

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

Usage

```r
## S4 method for signature 'Tracks,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)
```
### downsample

**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**

Compute 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/Frechet_distance

---

**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`. 
Inhomogeneous K-function for trajectory pattern

Description

Estimate the variability area of K-function of a list of tracks.

Usage

Kinhom.Track(X, timestamp, correction=c("border", "bord.modif", "isotropic", "translate"), q, sigma=c("default", "bw.diggle", "bw.ppl", "bw.scott"), ...)

Arguments

- `X` A list of Track objects
- `timestamp` based on secs, mins, ...
- `correction` the type of correction to be used in computing K-function
- `q` (optional) a numeric value between 0 and 1. quantile to be applied to calculate the variability area
- `sigma` method to be used in computing intensity function
- `...` passed to the arguments of Kinhom

Details

This calculates the variability area of K-function over time. If `sigma=default`, it calculates the variability area using the defaults of Kinhom, otherwise it first estimate the intensity function using the given sigma as bandwidth selection method and then using the estimated intensity function, it estimates the variability area.

Value

an object of class "KTrack".

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

See Also

rTrack, as.Track.ppp, Kinhom
Examples

```r
library(spatstat)
X <- list()
for(i in 1:100){
  m <- matrix(c(0,10,0,10),nrow=2,byrow = TRUE)
  X[[i]] <- rTrack(bbox = m,transform = TRUE)
}
Kinhom.Track(X,timestamp = "180 secs")
```

pcfinhom.Track

Pair correlation function of trajectory pattern

Description

Pair correlation function of trajectory pattern

Usage

```r
pcfinhom.Track(X,timestamp,correction = c("translate", "Ripley"),q,
                      sigma=c("default","bw.diggle","bw.ppl","bw.scott"),...)
```

Arguments

- `X`: A list of Track objects
- `timestamp`: based on secs,mins,...
- `correction`: the type of correction to be used in computing pair correlation function
- `q`: (optional) a numeric value between 0 and 1. quantile to be applied to calculate the variability area
- `sigma`: method to be used in computing intensity function
- `...`: passed to the arguments of pcfinhom

Details

This calculates the variability area of pair correlation function over time. If sigma=default, it calculates the variability area using the defaults of pcfinhom, otherwise it first estimate the intensity function using the given sigma as bandwidth selection method and then using the estimated intensity function, it estimates the variability area.

Value

an object of class "gTrack"

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>
See Also

rTrack, as.Track.ppp, pcfinhom

Examples

X <- list()
for(i in 1:100){
  m <- matrix(c(0,10,0,10),nrow=2,byrow = TRUE)
  X[[i]] <- rTrack(bbox = m,transform = TRUE)
}
g <- pcfinhom.Track(X,timestamp = "180 sec")
plot(g)
Description
The plot method for "distrack" objects.

Usage
## S3 method for class 'distrack'
plot(x, ...)

Arguments
- x: an object of class "distrack"
- ...: ignored

Details
This plots an object of class "distrack".

Author(s)
Mohammad Mehdi Moradi <moradi@uji.es>

Description
plot method

Usage
## S3 method for class 'gTrack'
plot(x, type = "l", col = "grey70", cex=1, line=2.2, ...)

Arguments
- x: an object of class "gTrack"
- type: line type
- col: line color
- cex: used for size of legend
- line: specifying a value for line overrides the default placement of labels, and places them this many lines outwards from the plot edge
- ...: passed on to plot
Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

Description

Methods for class "KTrack"

Usage

## S3 method for class 'KTrack'
plot(x, type = "l", col = "grey70", cex=1, line=2.2, ...)

Arguments

- **x**: an object of class KTrack
- **type**: line type
- **col**: color
- **cex**: used for size of legend
- **line**: specifying a value for line overrides the default placement of labels, and places them this many lines outwards from the plot edge
- **...**: passed on to plot

Details

plotting the variability area of K-function of a list of tracks.

Value

a plot.

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>
print.ArimaTrack

Methods for class "ArimaTrack"

Description

print method.

Usage

## S3 method for class 'ArimaTrack'
print(x, ...)

Arguments

x an object of class "ArimaTrack"
... ignored

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

print.arwlen

Methods for class "arwlen"

Description

to print an object of class "arwlen".

Usage

## S3 method for class 'arwlen'
print(x,...)

Arguments

x an object of class "arwlen"
... ignored

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>
print.distrack Methods for class "distrack"

Description

This is a method for class "distrack".

Usage

## S3 method for class 'distrack'
print(x,...)

Arguments

x an object of class "distrack"

... ignored

Details

This is a method for class "distrack".

Value

See the documentation on the corresponding generic function.

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

Examples

X <- list()
for(i in 1:10){
  m <- matrix(c(0,10,0,10),nrow=2,byrow = TRUE)
  X[[i]] <- rTrack(bbox = m,transform = TRUE)
}

ave <- avedistTrack(X,timestamp = "30 secs")
plot(ave,type="l")
Methods for class "gTrack"

Description
print method.

Usage
## S3 method for class 'gTrack'
print(x,...)

Arguments
x an object of class "gTrack"
... ignored

Author(s)
Mohammad Mehdi Moradi <moradi@uji.es>

Methods for class "KTrack"

Description
Methods for class "KTrack"

Usage
## S3 method for class 'KTrack'
print(x,...)

Arguments
x an object of class "KTrack"
... ignored

Details
to print an object of class "KTrack".

Author(s)
Mohammad Mehdi Moradi <moradi@uji.es>
print.ppplist

Methods for class "Track"

Description

method to print an object of class "ppplist"

Usage

```r
## S3 method for class 'ppplist'
print(x,...)
```

Arguments

- `x` an object of class "ppplist"
- `...` ignored

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

print.Track

Methods for class "Track"

Description

method to print an object of class "Track"

Usage

```r
## S3 method for class 'Track'
print(x,...)
```

Arguments

- `x` an object of class "Track"
- `...` ignored

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>
print.Tracks

Methods for class "Tracks"

Description
method to print an object of class "Tracks"

Usage
print.Tracks(X)

Arguments
X an object of class "Tracks"

Author(s)
Mohammad Mehdi Moradi <moradi@uji.es>

print.TracksCollection

Methods for class "TracksCollection"

Description
method to print an object of class "TracksCollection"

Usage
print.TracksCollection(X)

Arguments
X an object of class "TracksCollection"

Author(s)
Mohammad Mehdi Moradi <moradi@uji.es>
print.Trrow

Methods for class "Trrow"

Description
Print objects of class "Trrow"

Usage
## S3 method for class 'Trrow'
print(x,...)

Arguments
x an object of class "Trrow"
... ignored

Author(s)
Mohammad Mehdi Moradi <moradi@uji.es>

See Also
as.Track.arrow

range.Track

Description
Retrieves the range of a "Track" object

Usage
## S3 method for class 'Track'
range(X,...)

Arguments
X an object of class "Track"
... passed to arguments of range

Author(s)
Mohammad Mehdi Moradi <moradi@uji.es>
reTrack

See Also

rTrack, rTracks, rTracksCollection, range

Examples

```r
x <- rTrack()
range(x)
```

---

**reTrack**

*Reconstruct objects of class "Track"

**Description**

Function reTrack accepts X as an object of class "Track". Output is a reconstructed Track (again an object of class Track), based on a regular "timestamp". It only returns the interpolated points.

**Usage**

```r
reTrack(X, at=c("track","dfrm"), timestamp=timestamp, tsq=NULL)
```

**Arguments**

- `X` an object of class Track
- `at` to set the type of output as either an object of class "Track" or data.frame
- `timestamp` timestamp which Track be reconstructed based on
- `tsq` a time sequence to reconstruct Track X based on it. This is optional. If this is not given, the function creates the time sequence based on timestamp.

**Details**

Sometimes tracks data are not collected according to a regular timestamp. In order to compare different tracks which share some time intervals, we might need to be aware of the locations in a regular timestamp. Function reTrack enables us to reconstruct an object of class "Track" based on a regular timestamp. Time sequence can be given by user, if not reTrack creates a regular time sequence based on the given timestamp.

**Value**

Either an object of class "Track" or a data.frame

**Author(s)**

Mohammad Mehdi Moradi <moradi@uji.es>

**See Also**

rTrack, as.Track, as.POSIXct, compare
Examples

```r
library(sp)
library(spacetime)
t0 = as.POSIXct(as.Date("2013-09-30",tz="CET"))
# person A, track 1:
x = c(7,6,5,5,4,3,3)
y = c(7,7,6,5,5,6,7)
n = length(x)
set.seed(131)
t = t0 + cumsum(runif(n) * 60)
require(rgdal)
crs = CRS("+proj=longlat +datum=WGS84") # longlat
stidf = STIDF(SpatialPoints(cbind(x,y),crs), t, data.frame(co2 = rnorm(n)))
A1 = Track(stidf)
reTrack(A1,timestamp = "1 sec")
```

---

rTrack  Generate random Track, Tracks or TracksCollection objects

Description

Generate random Track, Tracks or TracksCollection objects

Usage

```r
rTrack(n = 100, origin = c(0,0), start = as.POSIXct("1970-01-01"), ar = .8,
step = 60, sd0 = 1, bbox = bbox, transform = FALSE, nrandom = FALSE, ...)
rTracks(m = 20, start = as.POSIXct("1970-01-01"), delta = 7200, sd1 = 0,
origin = c(0,0), ...)
rTracksCollection(p = 10, sd2 = 0, ...)
```

Arguments

- `n`: number of points per Track
- `origin`: numeric, length two, indicating the origin of the Track
- `start`: POSIXct, indicating the start time of the Track
- `ar`: numeric vector, indicating the amount of correlation in the Track
- `step`: numeric; time step(s) in seconds between Track fixes
- `sd0`: standard deviation of the random steps in a Track
- `sd1`: standard deviation of the consecutive Track origin values (using rnorm)
- `bbox`: bbox object
- `transform`: logical; FIXME:fill in
- `nrandom`: logical; if TRUE, draw n from rpois(n)
rTrack: arguments passed on to arima.sim, rTracks: arguments passed on to rTrack; rTracksCollection: arguments passed on to rTracks

\( m \)
number of Track objects to simulate

\( \text{delta} \)
time difference between consecutive Track start times

\( \text{p} \)
number of IDs with Tracks to generate

**Details**

\( \text{ar} \) is passed on to arima.sim as \( \text{ar} \) element, and may contain multiple AR coefficients. The generated track is a cumsum over the simulated AR values, for each dimension. In case it has length 1 and value 0, random walk is created using rnorm. If bbox is given, the generated track will be transformed to bbox. If transform is TRUE and no bbox is given, it transforms the track to a unit box. If \( \text{nrandom} \) is TRUE, it generates a random number using rpois with parameter n as the number of locations per track.

**Value**

An object of class Track, Tracks or TracksCollection.

**Author(s)**

Edzer Pebesma <edzer.pebesma@uni-muenster.de>, Mohammad Mehdi Moradi <moradi@uji.es>

**Examples**

\[ x = \text{rTrack()} \]
\[ \text{dim}(x) \]
\[ \text{plot}(x) \]
\[ \# x = \text{rTracks}(\text{sd1} = 120) \]
\[ \# \text{dim}(x) \]
\[ \# \text{plot}(\text{as}(x, \"SpatialLines"), \text{col} = 1:\text{dim}(x)[1], \text{axes} = \text{TRUE}) \]
\[ \# x = \text{rTracksCollection()} \# \text{star} \]
\[ \# \text{dim}(x) \]
\[ \# \text{plot}(x) \]
\[ x = \text{rTracksCollection}(\text{sd2} = 200, \text{p} = 4, m = 10) \]
\[ \text{plot}(x, \text{col} = 1:\text{dim}(x)[1]) \]

**Description**

obtain ranges of space and time coordinates

**Usage**

\[ \text{stbox}(\text{obj}) \]
**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`

---

**stcube**

*Draw a space-time cube.*

**Description**

Draw a space-time cube for a `Track`, `Tracks`, `TracksCollection`, `difftrack` or `STI(DF)` class.

**Usage**

```r
# S4 method for signature 'Track'
stcube(x, xlab = "x", ylab = "y", zlab = "t", type = "l", aspect, xlim = stbox(x)[[1]] + c(-0.1,0.1) * diff(stbox(x)[[1]]), ylim = stbox(x)[[2]] + c(-0.1,0.1) * diff(stbox(x)[[2]]), zlim = stbox(x)$time, showMap = FALSE, mapType = "osm", mapZoom = NULL, ..., y, z)
# S4 method for signature 'Tracks'
stcube(x, xlab = "x", ylab = "y", zlab = "t", type = "l", aspect, xlim, ylim, zlim, showMap = FALSE, mapType = "osm", normalizeBy = "week", mapZoom = NULL, ..., y, z, col)
# S4 method for signature 'TracksCollection'
stcube(x, xlab = "x", ylab = "y", zlab = "t", type = "l", aspect, xlim, ylim, zlim, showMap = FALSE, mapType = "osm", normalizeBy = "week", mapZoom = NULL, ..., y, z, col)
# S4 method for signature 'difftrack'
stcube(x, showMap = FALSE, mapType = "osm", normalizeBy = "week", ..., y, z)
# S4 method for signature 'STI'
stcube(x, xlab = "x", ylab = "y", zlab = "t", type = "p", aspect, xlim = stbox(x)[[1]] + c(-0.1,0.1) * diff(stbox(x)[[1]]), ylim = stbox(x)[[2]] + c(-0.1,0.1) * diff(stbox(x)[[2]]), zlim = stbox(x)$time, showMap = FALSE, mapType = "osm", mapZoom = NULL, ..., y, z)
```
## S4 method for signature 'STIDF'

`stcube(x, xlab = "x", ylab = "y", zlab = "t", type = "p", aspect,
  xlim = stbox(x)[[1]] + c(-0.1,0.1) * diff(stbox(x)[[1]]),
  ylim = stbox(x)[[2]] + c(-0.1,0.1) * diff(stbox(x)[[2]]),
  zlim = stbox(x)$time,
  showMap = FALSE, mapType = "osm", mapZoom = NULL, col, ..., 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 strategy.

- **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.

### Examples

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

---

### Description


### Usage

```r
data(storms)
```
Examples

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**

Track(track, df = fn(track), fn = TrackStats)
Tracks(tracks, tracksData = data.frame(row.names=names(tracks)), 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 **STIDF-class**, representing a single trip
- **df**: optional data.frame with information between track points
- **tracks**: named list with Track objects
**Track-class**

- **tracksData**: data.frame with summary data for each Track
- **tracksCollection**: list, with Tracks objects
- **tracksCollectionData**: data.frame, with summary data on tracksCollection
- **fn**: function;
- **x**: object of class Track etc
- **i**: selection of spatial entities
- **j**: selection of temporal entities (see syntax in package xts)
- **drop**: logical
- **from**: from
- **to**: target class

**Value**

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

**TrackStats** returns a 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.

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

**TracksSummary** reports for each Tracks of a TracksCollection (number of tracks) n, xmin, xmax, ymin, ymax, tmin, tmin, tmax, 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

\[
\begin{align*}
\text{[[ signature(obj = "Track")}: & \text{ retrieves the attribute element} \\
\text{[[ signature(obj = "Tracks")}: & \text{ retrieves the attribute element} \\
\text{[[ signature(obj = "TracksCollection")}: & \text{ retrieves the attribute element} \\
\text{[[ signature(obj = "Track")}: & \text{ sets or replaces the attribute element} \\
\text{[[ signature(obj = "Tracks")}: & \text{ sets or replaces the attribute element} \\
\text{[[ signature(obj = "TracksCollection")}: & \text{ sets or replaces the attribute element} \\
\text{$ signature(obj = "Track")}: & \text{ retrieves the attribute element} \\
\text{$ signature(obj = "Tracks")}: & \text{ retrieves the attribute element} \\
\text{$ signature(obj = "TracksCollection")}: & \text{ retrieves the attribute element} \\
\text{$<- signature(obj = "Track")}: & \text{ sets or replaces the attribute element} \\
\text{$<- signature(obj = "Tracks")}: & \text{ sets or replaces the attribute element} \\
\text{$<- signature(obj = "TracksCollection")}: & \text{ sets or replaces the attribute element} \\
\text{coerce Track, data.frame}: & \text{ coerce to data.frame} \\
\text{coerce Tracks, data.frame}: & \text{ coerce to data.frame} \\
\text{coerce TracksCollection, data.frame}: & \text{ coerce to data.frame} \\
\text{plot signature(x = "TracksCollection", y = "missing")}: & \text{ plots sets of sets of tracks} \\
\text{stplot signature(obj = "TracksCollection")}: & \text{ plots sets of sets of tracks}
\end{align*}
\]

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, 6, 7)
n = length(x)
set.seed(131)
t = t0 + cumsum(runif(n) * 60)
require(rgdal)
crs = CRS("+proj=longlat +datum=WGS84") # longlat
stidf = STIDF(SpatialPoints(cbind(x, y), crs), t, data.frame(co2 = rnorm(n))
A1 = Track(stidf)
# 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)
A2 = Track(stidf)

# 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)
B1 = Track(stidf)

# person B, track 2:
x = c(3, 3, 4, 3, 2, 1)
y = c(5, 4, 3, 2, 1, 1)
n = length(x)
t = max(t) + cumsum(runif(n) * 60)
B2 = Track(stidf)

# Tracks for person B:
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")
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) # good to do, but would generate warnings
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')

---

**Track.idw**

Movement smoothing of trajectory pattern

---

**Description**

Movement smoothing of trajectory pattern
Usages

```
Track.idw(X,timestamp,epsilon=epsilon,...)
```

Arguments

- **X**: a list of objects of class "Track"
- **timestamp**: based on secs, mins, ...
- **epsilon**: (optional) movements with length less than epsilon are not considered in the calculation
- **...**: passed to arguments of function `idw` in spatstat

Details

Performs spatial smoothing to the movements of a list of tracks.

Value

An image of class "im".

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

See Also

- `as.Track.arrow`, `idw`

Examples

```r
X <- list()
for(i in 1:10){
  m <- matrix(c(0,10,0,10),nrow=2,byrow = TRUE)
  X[[i]] <- rTrack(bbox = m,transform = TRUE)
}
Track.idw(X,timestamp="180 secs")
```

Description

tsqtracks returns a sequence of time based on a list of tracks (or a single object of class "Track") and an argument `timestamp`.

Usage

```
tsqTracks(X,timestamp)
```
Arguments

\[ X \] either an object of class "Track" or a list of some objects of class "Track"

\[ \text{timestamp} \] a timestamp to create the time sequence based on it

Details

This creates a sequence of time based on a track or a list of tracks.

Value

An object of class " POSIXct" or " POSIXt".

Author(s)

Mohammad Mehdi Moradi <moradi@uji.es>

See Also

rTrack

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(7,7,6,5,5,6,7)
n = length(x)
set.seed(131)
t = t0 + cumsum(runif(n) * 60)
require(rgdal)
crs = CRS("+proj=longlat +datum=WGS84") # longlat
stidf = STIDF(SpatialPoints(cbind(x,y),crs), t, data.frame(co2 = rnorm(n)))
A1 = Track(stidf)
tsqTracks(A1,timestamp = "1 sec")
```
**unique.Track**

**Arguments**

- `x` an object of class "Track"
- ... passed to arguments of `unique`

**Details**

This function removes duplicated points in an object of class "Track".

**Value**

An object of class Track with no duplicated point.

**Author(s)**

Mohammad Mehdi Moradi <moradi@uji.es>

**See Also**

`rTrack`, `rTracks`, `rTracksCollection`, `unique`

**Examples**

```r
x <- rTrack()
unique(x)
```
Index

*Topic **classes**
  difftrack-class, 14
  Track-class, 34
*Topic **compare**
  compare, 11
*Topic **datasets**
  A3, 3
  storms, 33
*Topic **dists**
  dists, 15
*Topic **downsample**
  downsample, 16
*Topic **dpplot**
  cut, 12
  stbox, 31
*Topic **generalize**
  generalize, 17
*Topic **methods**
  frechetDist, 16
*Topic **random**
  rTrack, 30
*Topic **space-time cube**
  stcube, 32
  [,Track,ANY,ANY,ANY-method (Track-class), 34
  [,Track-method (Track-class), 34
  [,Tracks,ANY,ANY,ANY-method (Track-class), 34
  [,Tracks-method (Track-class), 34
  [,TracksCollection,ANY,ANY,ANY-method (Track-class), 34
  [,TracksCollection-method (Track-class), 34
  <<-Track,ANY,missing-method (Track-class), 34
  <<-Tracks,ANY,missing-method (Track-class), 34
  <<-TracksCollection,ANY,missing-method (Track-class), 34
  $,Track-method (Track-class), 34
  $,Tracks-method (Track-class), 34
  $,TracksCollection-method (Track-class), 34
  $<-,Track-method (Track-class), 34
  $<-,Tracks-method (Track-class), 34
  $<-,TracksCollection-method (Track-class), 34
  A3, 3
  aggregate,Track-method (Track-class), 34
  aggregate,Tracks-method (Track-class), 34
  aggregate,TracksCollection-method (Track-class), 34
  approxTrack (Track-class), 34
  approxTracks (Track-class), 34
  approxTracksCollection (Track-class), 34
  arima.sim, 31
  as.list, 3, 4
  as.list.Tracks, 3
  as.list.TracksCollection, 4
  as.POSIXct, 5, 29
  as.ppp, 7
  as.Track, 4, 29
  as.Track.arrow, 5, 9, 39
  as.Track.ppp, 6, 6, 8, 18, 20
  auto.arima, 7, 8
  auto.arima.Track, 7
  avedistTrack, 7, 8
  avemove, 9

  bbox, 32
chimaps, 10  
coerce, Track, data.frame-method (Track-class), 34  
coerce, Tracks, data.frame-method (Track-class), 34  
coerce, TracksCollection, data.frame-method (Track-class), 34  
compare, 11, 29  
compare, Track-method  
coordnames, Track-method (Track-class), 34  
coordnames, Tracks-method (Track-class), 34  
coordnames, TracksCollection-method (Track-class), 34  
cumsum, 31  
cut, 12, 12  
density.list, 10, 13  
density.ppp, 10, 13  
differtrack (difftrack-class), 14  
differtrack-class, 14  
dists, 15  
dists, Tracks, Tracks-method (dists), 15  
downsamp, 16  
downsamp, Track-method (downsamp), 16  
frechetDist, 15, 16  
frechetDist, Track-method (frechetDist), 16  
generalize, 17  
generalize, Track-method (generalize), 17  
generalize, Tracks-method (generalize), 17  
generalize, TracksCollection-method (generalize), 17  
gSimplify, 17  
idw, 39  
Kinhom, 18  
Kinhom.Track, 18  
pcfinhom, 20  
pcfinhom.Track, 19  
plot, differtrack, ANY-method (diffrack-class), 14  
plot, Track, missing-method (Track-class), 34  
plot, Tracks, ANY-method (Track-class), 34  
plot, TracksCollection, ANY-method (Track-class), 34  
plot.arwlen, 20  
plot.distrack, 21  
plot.gTrack, 21  
plot.KTrack, 22  
print.ArimaTrack, 23  
print.arwlen, 23  
print.distrack, 24  
print.KTrack, 25  
print.ppplist, 26  
print.Track, 26  
print.Tracks, 27  
print.TracksCollection, 27  
print.Trow, 28  
rnorm, 31  
rpois, 31  
rTrack, 3, 4, 6, 8, 13, 18, 20, 29, 30, 41  
rTracks, 3, 4, 29, 41  
rTracks (rTrack), 30  
rTracksCollection, 3, 4, 29, 41  
rTracksCollection (rTrack), 30  
segments-class (Track-class), 34  
segPanel (Track-class), 34  
SpatialLines, 17  
SpatialLinesDataFrame, 14  
spTransform, Track, CRS-method (Track-class), 34  
spTransform, Tracks, CRS-method (Track-class), 34  
spTransform, TracksCollection, CRS-method (Track-class), 34  
stbox, 31  
stbox, Tracks-method (stbox), 31  
stbox, TracksCollection-method (stbox), 31  
stcube, 32  
stcube, differtrack-method (stcube), 32  
stcube, STI-method (stcube), 32  
stcube, STIDF-method (stcube), 32  
stcube, Track-method (stcube), 32  
stcube, Tracks-method (stcube), 32
stcube, TracksCollection-method
   (stcube), 32
STIDF-class, 34, 35
storms, 33
stplot, TracksCollection-method
   (Track-class), 34

Track, 5, 14
Track (Track-class), 34
Track-class, 34
Track.idw, 38
Tracks (Track-class), 34
Tracks-class (Track-class), 34
TracksCollection (Track-class), 34
TracksCollection-class (Track-class), 34
tracksPanel (Track-class), 34
TracksSummary (Track-class), 34
TrackStats (Track-class), 34
TrackSummary (Track-class), 34
tsqTracks, 39

unique, 41
unique.Track, 40