Package 'TrackMateR'

July 30, 2022

Type Package

Title Working with TrackMate outputs in R

Version 0.1.0

Author Stephen Royle

Maintainer quantixed <admin@quantixed.org>

Description TrackMate, a plugin for ImageJ/Fiji, is a popular single-particle tracking solution. Building on the trackR package by Julien Godet, the aim is to import TrackMate data into R for further analysis and visualization.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

Depends R (>= 2.10)

Imports doParallel, dplyr, foreach, ggplot2, patchwork, reshape2, XML, zoo

R topics documented:

calculateAlpha *Calculate alpha (relationship between MSD and normal diffusion)*

Description

Normal diffusion is alpha = 1. Subdiffusion is alpha < 1 and superdiffusion is alpha > 1. Input is a data matrix of msd curves. Output is mean of log2(alpha), one value for each trace.

Usage

```
calculateAlpha(alphaMat, tstep)
```

Arguments

alphaMat	matrix of msd curves, each col is a track, each row is time lag (will contain NAs)
tstep	variable. Time step in seconds

Value

numeric vector

calculateJD *Calculate Jump Distance (JD)*

Description

Calculation of the JD of multiple tracks. Calculation is equivalent to a single time lag point on the ensemble MSD curve, typically represented as a histogram Input is a data frame of tracks imported using readTrackMateXML() The default time step is one frame - which is the equivalent to the plot generated to show displacement versus time.

Usage

```
calculateJD(df, deltaT = 1)
```

Arguments

df	data frame must include at a minimum - trace (track ID), x, y and t (in real coords)
deltaT	integer to represent the multiple of frames that are to be analysed

Value

vector of jump distances, NAs removed

Examples

```
xmlPath <- "~/Desktop/FakeTracks.xml"
datalist <- readTrackMateXML(XMLpath = xmlPath)
data <-  datalist[[1]]
data <- correctTrackMateData(data, xy = 0.04)
jdvec <- calculateJD(data, deltaT = 2)
```

calculateMSD *Calculate Mean Squared Displacement (MSD)*

Description

Calculation of the MSD of multiple tracks. There are two methods for everaging MSD data from multiple tracks: ensemble = for each time lag average all squared displacements from all tracks time-averaged = find MSD for each track and then generate the average MSD from these curves The MSD curves will be identical if all tracks are the same length, and diverge if not. Standard deviation will be large for ensemble and smaller for time-averaged data. Input is a data frame of tracks imported using readTrackMateXML()

Usage

```
calculateMSD(df, method = "timeaveraged", N = 4, short = 0)
```

Arguments

df	data frame must include at a minimum - trace (track ID), x, y and t (in real coords)
method	string. Either "ensemble" or "timeaveraged" (default)
N	numeric variable for MSD. dt should be up to 1/N of number of data points (4 recommended)
short	numeric variable for the shortest number of points we will analyse. Note, this uses the number of frames from start, not number of points in track, i.e. a track with <short points and many gaps will remain

Value

list of a data frame and a vector

Examples

```
xmlPath <- "~/Desktop/FakeTracks.xml"
datalist <- readTrackMateXML(XMLpath = xmlPath)
data <-  datalist[[1]]
data <- correctTrackMateData(data, xy = 0.04)
msdobj <- calculateMSD(data, method = "ensemble", N = 3, short = 8)
```

correctTrackMateData *Correct distance and time of imported TrackMate data.*

Description

If the TrackMate data is in pixels and/or frames, the data frame can be converted with this function.

Usage

```
correctTrackMateData(df, xyscalar = 1, tscalar = 1)
```

Arguments

df	data frame of imported track mate data
xyscalar	numeric multiplier to correct pixel size of original movie. Assumes isotropic scaling, i.e. pixel height = pixel width
tscalar	numeric multiplier to correct frame interval of original movie. Frame interval of tracked data.

Value

data frame

Examples

```
xmlPath <- "~/Desktop/FakeTracks.xml"
datalist <- readTrackMateXML(XMLpath = xmlPath)
data <-  datalist[[1]]
# in the case where pixel size is 0.03 um and original data is 1 pixel, xyscalar = 0.03
data <- correctTrackMateData(df = data, xyscalar = 0.03)
```

---

fittingJD *Fitting jump distance (JD) data*

Description

Jump Distances have been calculated for a given time lag. They can be described by fitting curves to the data, either using a histogram or cumulative probability density function. Firtting to a histogram is sensitive to binning parameters and ECDF performs better for general use. The idea behind this analysis is given in: - Weimann et al. (2013) A quantitative comparison of single-dye tracking analysis tools using Monte Carlo simulations. PloS One 8, e64287. - Menssen & Mani (2019) A Jump-Distance-Based Parameter Inference Scheme for Particulate Trajectories, Biophysical Journal, 117: 1, 143-156. The bulk of this code is taken from trackR by JuG

Usage

```
fittingJD(
  df,
  mode = "ECDF",
  nPop = 1,
  init,
  units = c("um", "s"),
  timeRes = 1,
  breaks = 100
)
```

Arguments

df	data frame with a column named jump of jump distances
mode	string indicated ECDF (default) or hist (histogram)
nPop	number of populations of diffusing species (1, 2 or 3)

init	initialisation parameters for the nls fit for example list(D2 = 200, D1 = 0.1) or list(D2 = 0.01, D1=0.1, D3=10, D4=100)
units	character vector to describe units (defaults are um, micrometres and s, seconds)
timeRes	time resolution per unit of jump. Frame interval is 0.5 s and jump interval is two steps, timeRes = 1.
breaks	number of bins for histogram. With ECDF breaks can be high e.g. 100, for mode = "hist" they should be low, perhaps 30.

Value

ggplot

Examples

```
xmlPath <- "~/Desktop/FakeTracks.xml"
datalist <- readTrackMateXML(XMLpath = xmlPath)
data <-  datalist[[1]]
data <- correctTrackMateData(data, xy = 0.04)
jdvec <- calculateJD(data, deltaT = 2)
jdDF <- data.frame(jump = jdvec)
fittingJD(df = jdDF, mode = "ECDF", nPop = 2, breaks = 100, timeRes = 0.06)
```

---

makeReport *Make Report*

Description

Generate several plots to visualise TrackMate data and generate a report. Note that the units are hard-coded as um and s. The use of um is because ggsave does not currently save unicode to PDF reliably.

Usage

```
makeReport(
  df,
  msdlist,
  jumpdata,
  jumptime,
  units = c("um", "s"),
  titleStr = "",
  subStr = ""
)
```

Arguments

df	imported TrackMate data with correct units
msdlist	MSD summary and alpha list = output from calculateMSD()
jumpdata	data frame of jump data
jumptime	variable to be passed to timeRes
units	character vector to describe units (defaults are um, micrometres and s, seconds)
titleStr	string used as the title for the report
subStr	string used as the subtitle for the report

Value

patchwork ggplot

Examples

```
xmlPath <- "~/Desktop/FakeTracks.xml"
datalist <- readTrackMateXML(XMLpath = xmlPath)
data <-  datalist[[1]]
data <- correctTrackMateData(data, xy = 0.04)
calibration <- datalist[[2]]
msdobj <- calculateMSD(df = data, method = "ensemble", N = 3, short = 8)
jdDF <- calculateJD(data, deltaT = 1)
jdDF <- data.frame(jump = jdDF)
fileName <- tools::file_path_sans_ext(basename(xmlPath))
makeReport(df = data, msdlist = msdobj, jumpdata = jdDF, jumptime = 0.06,
titleStr = "Report", subStr = fileName)
```

---

makeSummary *Make Summary*

Description

Generate several plots to visualise TrackMate data and generate a report. Note that the units are hard-coded as um and s. The use of um is because ggsave does not currently save unicode to PDF reliably.

Usage

```
makeSummary(
  df,
  msdlist,
  jumpdata,
  jumptime,
  units = c("um", "s"),
  titleStr = "Summary",
  subStr = NULL
)
```

Arguments

df	imported TrackMate data with correct units
msdlist	list of 2 data frames: MSD summary and alpha summary = collated outputs from calculateMSD()
jumpdata	data frame of jump data
jumptime	variable to be passed to timeRes
units	character vector to describe units (defaults are um, micrometres and s, seconds)
titleStr	string used as the title for the summary
subStr	string used as the subtitle for the summary

Value

patchwork ggplot

plotMSD *Make a plot of MSD data*

Description

Generate a plot of MSD over a series of increasing time lags. Input is the output from CalculateMSD(), so the plot will display the ensemble or time-averaged MSD (whatever was requested) A fit to the first four points is displayed to evaluate alpha. Diffusion coefficient from this fit is displayed top-left.

Usage

```
plotMSD(df, units = c("um", "s"), bars = FALSE, xlog = FALSE, ylog = FALSE)
```

Arguments

df	MSD summary = output from calculateMSD()
units	character vector to describe units (defaults are um, micrometres and s, seconds)
bars	boolean to request error bars (1 x SD)
xlog	boolean to request log10 x axis
ylog	boolean to request log10 y axis

Value

S3 ggplot

Examples

```
xmlPath <- "~/Desktop/FakeTracks.xml"
datalist <- readTrackMateXML(XMLpath = xmlPath)
data <-  datalist[[1]]
# use the ensemble method and only look at tracks with more than 8 points
msdobj <- calculateMSD(df = data, method = "ensemble", N = 3, short = 8)
msddf <- msdobj[[1]]
plotMSD(msddf, bars = FALSE)
```

plotNMSD *Plot several (n) MSD curves*

Description

Generate a plot of several MSD curves together with a summary curve.

Usage

```
plotNMSD(df)
```

Arguments

df	dataframe of MSD summary data from multiple datasets (labelled by dataid)

Value

ggplot

`readTrackMateXML` *Read TrackMate XML output files.*

Description

Produces a data frame of all spots from filtered tracks, ordered by track number. A warning is generated if the scaling is in pixels rather than real units.

Usage

```
readTrackMateXML(XMLpath)
```

Arguments

`XMLpath`	path to the xml file

Value

list of two data frames

Examples

```
xmlPath <- "~/Desktop/FakeTracks.xml"
datalist <- readTrackMateXML(XMLpath = xmlPath)
# get the track data in a data frame
data <-  datalist[[1]]
# get the calibration data in a data frame
calibration <- datalist[[2]]
```

Index