Package 'SitesInterest'

October 16, 2018

Type Package

Title Inferring an Animal's Sites of Interest from High Resolution Data

Version 1.0

Date 2018-09-21

Author Rhys Munden <rdmunden1@sheffield.ac.uk>

Maintainer Rhys Munden <rdmunden1@sheffield.ac.uk>

Description Within the area of movement ecology, it can be useful to identify an animal's sites of interest. These will be places that an animal uses a lot and so will spend a proportionately longer time there. The aim of this package is to identify both the position and size of these sites of interest from high resolution (sub-second) positional data.

Depends R (>= 3.3.1)

Imports graphics, plotrix, utils

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Suggests

NeedsCompilation no

Repository CRAN

Date/Publication 2018-10-16 15:10:10 UTC

R topics documented:

`Alt_Alg` *Calculates the residence times and identifies the sites of interest*

Description

From positional data, this function finds the residence times as well as the number of visits. It also identifies which areas are sites of interest.

Usage

```
Alt_Alg(Name, t, X, Y, R, s = 10, m = 500, first = 'n', save = 'n')
```

Arguments

`Name`	name of the data, which is used for any saved files and plot titles
`t`	array of the times that the positions are recorded at
`X`	array of the x-coordinates describing the trajectory
`Y`	array of the y-coordinates describing the trajectory
`R`	radius value to use
`s`	number of time steps between checks for entrances and exits
`m`	estimate of the maximum number of crossings across all circles
`first`	if `'y'`, the algorithm will look for the second greatest maximum percent drop if the first results in the first circle being the only non-identified site
`save`	if `'y'`, save the files

Details

This is the main part of the algorithm. Given the trajectory of an animal, the algorithm centres a circle at the start of the trajectory. The next circle is centred at the point when the animal first leaves the previous circle. This is then repeated all along the trajectory until the whole trajectory is covered. The algorithm then finds the times of when the animal has crossed the perimeter of these circles, the results of which are saved in two csv files (comma separated values), with titles

'*Name*'_F_alt_R'*R*'.csv and '*Name*'_B_alt_R'*R*'.csv for the forward and backward crossings respectively. A forward crossing is one that occurs after the time point that the circle is centred on and backward crossings occur before this.

The residence times and number of visits for each circle are calculated, which is also stored in a csv file with the title '*Name*'_UD_alt_R'*R*'.csv. The function orders all the circles in descending order of residence time and then moving down the list, any that overlap with one of a higher residence time are removed. The relative difference between consecutive residence times (percent drops) are calculated and the maximum is identified (`maximum_percent_drop`). The number of identified sites (`Number_identified_sites`) is the number of circles before this maximum percent drop. This can be seen visually, using `plot_bar_chart`.

Value

max_percent_drop - maximum percent drop, which is the relative difference between the bars, indicating the difference between sites and non-site circles

number_identified_sites - number of circles identified as sites of interest

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

##Calculate the residence time with a radius of 0.3 and not including the first circle
Alt_Alg("OU_14",t,X,Y,0.3,first='y',save='y')

##Reset the original working directory
setwd(wd)
```

`Alt_Alg_discont` *Application of Alt_Alg to discontinuous data*

Description

Calculates the residence times from discontinuous data, which can then be used to identify sites of interest. This could also be used to identify sites for a group of animals, by treating each animal's trajectory as one segment of a discontinuous set.

Usage

```
Alt_Alg_discont(Overall_name,Names, t_all, X_all, Y_all, R, s = 10, m = 500, save = 'n')
```

Arguments

`Overall_name`	name for the set of separate trajectories
`Names`	list of names for each trajectory
`t_all`	list of arrays, one for each trajectory of times when the positions were recorded
`X_all`	list of arrays, one for each trajectory of x-coordinates
`Y_all`	list of arrays, one for each trajectory of y-coordinates
`R`	radius value to use
`s`	number of time steps between checks for entrances and exits
`m`	estimate of the maximum number of crossings across all circles
`save`	if `'y'`, save the files

Details

This function is used specifically with discontinuous data to calculate the residence times for the trajectories. It works in the same way as for `Alt_Alg`, by linking together `Alt_Alg_mini` and `combining`.

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also `Alt_Alg` for the algorithim used on continuous data. `Alt_Alg_mini` is used to calculate the residence times for a particular set of circles and a particular trajectory, then using `combining` all the residence times for the same circles are summed.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

##Number of path sections
n=5
##Number of recorded locations
N = length(t)

##A list of arrays of the time recoding for the 3 of the trajectory segments
t_all = list(t[seq(1,floor(N/n))], t[seq(floor(N/n)*2,floor(N/n)*3)],
t[seq(floor(N/n)*4,floor(N/n)*5)])

##A list of arrays of the x-coordinates for the 3 of the trajectory segments
X_all = list(X[seq(1,floor(N/n))], X[seq(floor(N/n)*2,floor(N/n)*3)],
X[seq(floor(N/n)*4,floor(N/n)*5)])

##A list of arrays of the y-coordinates for the 3 of the trajectory segments
Y_all = list(Y[seq(1,floor(N/n))], Y[seq(floor(N/n)*2,floor(N/n)*3)],
Y[seq(floor(N/n)*4,floor(N/n)*5)])

##The calculation of the residence time for discontibuous data
Alt_Alg_discont("OU_14_discont",c("OU_14.1","OU_14.3","OU14.5"),t_all,X_all,Y_all,0.3,save='y')

##Reset the original working directory
setwd(wd)
```

`Alt_Alg_mini` *Calculates the residence times for circles taken from one trajectory, but applied to another*

Description

The circles found from applying `Alt_Alg` to one trajectory are used to find the residence times of another trajectory passing through these circles.

Usage

```
Alt_Alg_mini(Circles_name, t_centers, X_centers, Y_centers, Path_name, t, X, Y, R,
s = 10, m = 500, save = 'n')
```

Arguments

Circles_name	name of the trajectory used to find the circles
t_centers	array of times when the positions were recorded
X_centers	array of the x-coordinates of the circles' centres
Y_centers	array of the y-coordinates of the circles' centres
Path_name	name of the trajectory that the residence times are found from
t	array of the times that the positions are recorded at
X	array of the x-coordinates describing the trajectory
Y	array of the y-coordinates describing the trajectory
R	radius value to use
s	number of time steps between checks for entrances and exits
m	estimate of the maximum number of crossings across all circles
save	if 'y', save the files

Details

This functions works in a similar way to Alt_Alg, but the circles are found from one trajectory and are applied to another. The results are stored in a csv file '*Circles_name*'_multi_'*Path_name*'_UD_alt_R'*R*'.csv and the crossing times are stored in '*Circles_name*'_multi_'*Path_name*'_M_alt_R'*R*'.csv

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also Alt_Alg for how to apply the algorithm to continuous data. Alt_Alg_discont perform Alt_Alg_mini on all trjectories, then combining combines the results from each application of Alt_Alg_mini.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])
```

```
##Number of path sections
n=5
##Number of recorded locations
N = length(t)

##A list of arrays of the time recoding for the 3 of the path segments
t_all = list(t[seq(1,floor(N/n))], t[seq(floor(N/n)*2,floor(N/n)*3)],
t[seq(floor(N/n)*4,floor(N/n)*5)])

##A list of arrays of the x-coordinates for the 3 of the path segments
X_all = list(X[seq(1,floor(N/n))], X[seq(floor(N/n)*2,floor(N/n)*3)],
X[seq(floor(N/n)*4,floor(N/n)*5)])

##A list of arrays of the y-coordinates for the 3 of the path segments
Y_all = list(Y[seq(1,floor(N/n))], Y[seq(floor(N/n)*2,floor(N/n)*3)],
Y[seq(floor(N/n)*4,floor(N/n)*5)])

##Calculates the residence time for one particular path segment
Alt_Alg("OU_14.1",unlist(t_all[1]),unlist(X_all[1]),unlist(Y_all[1]),0.3,first='y',save='y')

##Load the data of the circles found from Alt_Alg
df = read.csv(paste("OU_14.1","_UD_alt_R",0.3,".csv",sep=''))
t_centers = unlist(df[1])
X_centers = unlist(df[2])
Y_centers = unlist(df[3])

##Calculates the residence time from path segment 3, using circles from path segment 1
Alt_Alg_mini("OU14.1", t_centers, X_centers, Y_centers, "OU_14.3", unlist(t_all[2]),
unlist(X_all[2]), unlist(Y_all[2]), 0.3,save='y')

##Reset the original working directory
setwd(wd)
```

`combining` *Combines the residence times for the same set of circles and different trajectories*

Description

Combines the residence times for the same set of circles and different trajectories and also for the entire set.

Usage

```
combining(Overall_name, Circle_names, Path_names, R, save = 'n')
```

Arguments

Overall_name	name for the set of separate trajectories
Circle_names	the list of names for each set of circles
Path_names	the list of names for each trajectory
R	radius value to use
save	if 'y', save the files

Details

This function combines the already calculated residence times for each of the path segments as well as combining the residence times of all path segments across the same set of circles. The results are stored in a csv file named '*Overall_name*'_combined_UD_alt_R'*R*'.csv.

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also Alt_Alg_mini for how to calculate the residence times for a particular set of circles and a particular trajectory. Alt_Alg_discont is used to calculate the residence times across a whole set of discontinuous trajectories.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

##Number of path sections
n=5
#Number of recorded locations
N = length(t)

##A list of arrays of the time recoding for the 3 of the path segments
t_all = list(t[seq(1,floor(N/n))], t[seq(floor(N/n)*2,floor(N/n)*3)],
t[seq(floor(N/n)*4,floor(N/n)*5)])
##A list of arrays of the x-coordinates for the 3 of the path segments
```

```
X_all = list(X[seq(1,floor(N/n))], X[seq(floor(N/n)*2,floor(N/n)*3)],
X[seq(floor(N/n)*4,floor(N/n)*5)])
##A list of arrays of the y-coordinates for the 3 of the path segments
Y_all = list(Y[seq(1,floor(N/n))], Y[seq(floor(N/n)*2,floor(N/n)*3)],
Y[seq(floor(N/n)*4,floor(N/n)*5)])

##The names of each path segment
Names = c("OU_14.1","OU_14.3","OU_14.5")

##Calculates the residence time for each path segment individually
Alt_Alg("OU_14.1",unlist(t_all[1]),unlist(X_all[1]),unlist(Y_all[1]),0.3,first='y',save='y')
Alt_Alg("OU_14.3",unlist(t_all[2]),unlist(X_all[2]),unlist(Y_all[2]),0.3,first='y',save='y')
Alt_Alg("OU_14.5",unlist(t_all[3]),unlist(X_all[3]),unlist(Y_all[3]),0.3,first='y',save='y')

Circle_names = Names
Path_names = Names

##Calculate the residence time for each set of circles and each path segment
for (Circles_name in Circle_names){
  df = read.csv(paste(Circles_name,"_UD_alt_R",0.3,".csv",sep=''))
  t_centers_list = df[1]
  X_centers_list = df[2]
  Y_centers_list = df[3]
  t_centers = unlist(t_centers_list)
  X_centers = unlist(X_centers_list)
  Y_centers = unlist(Y_centers_list)
  for (Path_name in Path_names){
  if (Circles_name != Path_name){
       index = match(Path_name,Names)
       Alt_Alg_mini(Circles_name, t_centers, X_centers, Y_centers, Path_name,
      unlist(t_all[index]),unlist(X_all[index]),unlist(Y_all[index]),0.3,s=10,m=500,save='y')}}}

##Combine all the residence times for the same circles
combining("OU_14_discont", Circle_names, Path_names, 0.3,save='y')

##Reset the original working directory
setwd(wd)
```

OU_14 *Trajectory of an Ornstein-Uhlenbeck (OU) simulation.*

Description

The trajectory of a switching OU simulation with 5 points of attraction.

Usage

```
data("OU_14")
```

Format

A data frame of 30000 rows and 3 columns.

Details

A switching OU simulation is one where the simulated object will move toward a point of attraction, where the strength of attraction is proportional to the distance from the point. The point of attraction changes at particular points in time. There is also an element of (Gaussian) randomness in the movement. In this particular simulation the points of attraction were (8,3), (5,9), (9,4), (8,2) and (6,9). The long term standard deviation about these points is 15.689, however it should be noted that the simulation was stopped before reaching this.

- t. simulated time recordings.
- X. simulated x-coordinates.
- Y. simulated y-coordinates.

Source

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

Examples

```
##Load the data
data(OU_14)
```

---

plot.displacement *Plots the displacement from a particular point over a chosen window*

Description

The displacement from a particular point to every other point along the animal's trajectory is calculated and this is then plotted over a defined window.

Usage

```
## S3 method for class 'displacement'
plot(x, y, Name, t, R, t_a, t_1, t_2, ...)
```

Arguments

x	array of the x-coordinates describing the trajectory
y	array of the y-coordinates describing the trajectory
Name	name of the data, which is used for any saved files and plot titles
t	an array of the times that the positions are recorded at

R	radius value to use
t_a	starting time to calculate the displacement from
t_1	start of interval to view the displacement over
t_2	end of interval to view the displacement over
...	additional arguments to plot

Details

The displacement from a particular time point (t_a) to every other point along the trajectory is calculated and this is then plotted over a defined window [t_1, t_2]. A line representing the radius is also drawn to see when the trajectory enters or leaves the circle centred at the point (t_a). This plot can be used to see how far away the animal moves after leaving a particular circle.

Value

Plot of the displacement

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

Examples

```
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

class(X) = "displacement"
class(Y) = "displacement"

##Plot the displacement from the starting point (t=0) for t=0 to t=2.9999
plot(X, Y, "OU14", t, 0.3, 0, 0, 2.9999)
```

plot.hoops *Plots the trajectory with sites of interest and other hoops*

Description

Plots the trajectory of the animal with the different hoops (sites of interest, removed circles and other circles) overlaid.

Usage

```
## S3 method for class 'hoops'
plot(x, y, Name, R, first = 'n', colours = c('orange','darkgreen','red'),
lwds = c(2,2,2), number_sites=-1, ...)
```

Arguments

x	array of the x-coordinates describing the trajectory
y	array of the y-coordinates describing the trajectory
Name	name of the data, which is used for any saved files and plot titles
R	radius value to use
first	if 'y', the algorithm will look for the second greatest maximum percent drop if the first results in the first circle being the only non-identified site
colours	list of the hoops' colours
lwds	list of hoop widths
number_sites	number of sites to manually show the results for
...	additional arguments to plot

Details

This function plots the trajectory of the animal with the identified sites of interest, removed circles and other circles plotted on top. The colours of the three types of the circles and their widths can also be defined. This can be used to see visually where the sites are along the trajectory.

Value

Plot of the identified sites' positions

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also Alt_Alg to find the residence times. Sites can be used to find the coordinates of the centres of the sites and non-overlapping circles from the csv files produced by Alt_Alg.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

class(X) = "hoops"
class(Y) = "hoops"

##Calculate the residence time with a radius of 0.3 and not including the first circle
Alt_Alg("OU_14",t,X,Y,0.3,first='y',save='y')

##Plot the positions of the identified sites as well as the non-overlapping circles
plot(X,Y,"OU_14",0.3)

##The colours for the hoops can be changed
plot(X,Y,"OU_14",0.3,first='y',colours=c('tan','chocolate','maroon'))

##The thickness of hoops can also be changed
plot(X,Y,"OU_14",0.3,first='y',lwds=c(0.5,2,3.5))

##It is also possible to manually choose the number of sites
plot(X,Y,"OU_14",0.3,first='y',number_sites=4)

##Reset the original working directory
setwd(wd)
```

---

plot.schematic *Plots a schematic representation of the movement trajectory*

Description

Plots a schematic representation of the movement from site to site, but does not include returns to the same site.

Usage

```
## S3 method for class 'schematic'
plot(x, y, Name, R, first = 'n', number_sites = -1, len_arrow = 0, lwd_arrow = 0.5,
lwd_r = 1, text_size = 1, legend_loc = "topright", ...)
```

Arguments

x	array of the x-coordinates describing the trajectory
y	array of the y-coordinates describing the trajectory
Name	name of the data, which is used for any saved files and plot titles
R	radius value to use
first	if 'y', the algorithm will look for the second greatest maximum percent drop if the first results in the first circle being the only non-identified site
number_sites	number of sites to manually show the results for
len_arrow	length of the arrows
lwd_arrow	thickness of the arrows
lwd_r	width of the hoops
text_size	size of the labels
legend_loc	location of the legend
...	additional arguments to plot

Details

This function plots a schematic representation of the animal's movements from site to site, with arrows indicating the direction of movement. This plot simplifies the movements of the animal so that movement is only direct from site to site. Several features of the plot can be defined including; the size of the arrow heads, thickness of the arrows and hoops, text size of the arrow labels and the location of the legend.

Value

Plot of the schematic representation

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also Alt_Alg to find the residence times. Sites can be used to find the coordinates of the centres of the sites from the csv files produced by Alt_Alg. print_site_visits prints the order of sites visited, the length of time for each visit and the amount of time spent between site visits.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

class(X) = "schematic"
class(Y) = "schematic"

##Calculate the residence time with a radius of 0.3 and not including the first circle
Alt_Alg("OU_14",t,X,Y,0.3,first='y',save='y')

##Plot the schematic representation of movements between sites
plot(X,Y,"OU_14",0.3,first='y')

##There is also the option to make changes to:
##the length of the arrow head
plot(X,Y,"OU_14",0.3,first='y',len_arrow=0.25)

##the thickness of the arrow
plot(X,Y,"OU_14",0.3,first='y',lwd_arrow=2)

##the thickness of the hoops
plot(X,Y,"OU_14",0.3,first='y',lwd_r=2)

##the size of the arrow labels
plot(X,Y,"OU_14",0.3,first='y',text_size=2)

##the location of the legend
plot(X,Y,"OU_14",0.3,first='y',legend_loc="bottomleft")

##Reset the original working directory
setwd(wd)
```

---

plot_bars_and_hoops *Plots both the bar chart and hoops plot in one figure*

Description

The bar chart is the plot of the ranked non-overlapping residence times and the hoops plot shows where along the trajectory these different circles are located.

Usage

```
plot_bars_and_hoops(x, y, Name, R, colours = c('orange','darkgreen','red'), lwds =
c(2,2,2), number_sites=-1, first='n')
```

Arguments

x	array of the x-coordinates describing the trajectory
y	array of the y-coordinates describing the trajectory
Name	the name of the data used to save the files and also in the title of the plot
R	radius value to use
colours	list of colours used for the bars and hoops
lwds	list of the widths of the circles
number_sites	number of sites to manually show the results for
first	if 'y', the algorithm will look for the second greatest maximum percent drop if the first results in the first circle being the only non-identified site

Details

The first subplot (plot_bar_chart) shows the residence times of circles that are left after overlapping ones are removed and identifies where the maximum percent drop occurs by a change in the colour of bars.

The second subplot (plot.hoops) includes the animal's trajectory as well as the three different types of circles overlaid (sites, non-overlapping circles and removed circles). The colour and thickness of these circles can be changed.

Value

Plot of the bar chart of residence times and site positions

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also Alt_Alg to find the residence times. Sites can be used to find the coordinates of the centres of the sites and non-overlapping circles from the csv files produced by Alt_Alg. See plot_bar_chart and plot.hoops for how to plot each separately and also the different presentation options.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
```

```
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

class(X) <- "bars_and_hoops"
class(Y) <- "bars_and_hoops"

##Calculate the residence time with a radius of 0.3 and not including the first circle
Alt_Alg("OU_14",t,X,Y,0.3,first='y',save='y')

##Plot the bar chart of ranked non-overlapping residence times and the plot showing
##the positions of these circles
plot_bars_and_hoops(X,Y,"OU_14",0.3,first='y')

##Reset the original working directory
setwd(wd)
```

`plot_bar_chart` *Plot the bar chart of residence times*

Description

This function plots the bar chart of residence times with the ones that correspond to the sites of interest highlighted by a change in colour.

Usage

```
plot_bar_chart(Name, R, first='n', number_sites=-1, colours= c("red","darkgreen"))
```

Arguments

`Name`	name of the data, which is used for any saved files and plot titles
`R`	radius value to use
`first`	if `'y'`, the algorithm will look for the second greatest maximum percent drop if the first results in the first circle being the only non-identified site
`number_sites`	number of sites to manually show the results for
`colours`	list of the bars' colours

Details

Plots the bar chart of ranked residency times of non-overlapping circles. The percent drops are the relative differences between these consecutive bars and where the maximum percent drop occurs is identified using a change in colour between the bars. This plot can be used to see the percent drops visually and if the maximum percent drop can easily be identified by sight.

Value

Bar chart plot of the the residence times of non-overlapping circles

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also Alt_Alg to find the residence times. Sites can be used to find the coordinates of the centres of the sites and non-overlapping circles from the csv files produced by Alt_Alg.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

##Calculate the residence time with a radius of 0.3 and not including the first circle
Alt_Alg("OU_14",t,X,Y,0.3,first='y',save='y')

##Plot the bar chart of ranked non-overlapping residence times
plot_bar_chart("OU_14",0.3,first='y')

##It is possible to choose manually where the cut off between sites and none sites should be
plot_bar_chart("OU_14",0.3,first='y', number_sites=4)

##The colours can also be changed
plot_bar_chart("OU_14",0.3,first='y', colours = c("darkgreen","red"))

##Reset the original working directory
setwd(wd)
```

`plot_radii_results` *Plots the results from using different radii*

Description

Plots the results (maximum percent drop and number of identified sites) from already calculated residence times for different radii values.

Usage

```
plot_radii_results(Name, Radii, first = 'n', legend_loc = "topright")
```

Arguments

`Name`	name of the data, which is used for any saved files and plot titles
`Radii`	set of radius values
`first`	if `'y'`, the algorithm will look for the second greatest maximum percent drop if the first results in the first circle being the only non-identified site
`legend_loc`	location of the legend

Details

After finding the residence times for various radii values using `Alt_Alg`, this function plots the number of identified sites of interest and their corresponding maximum percent drops for each radii value. This plot could be used to identify which radius value to use.

Value

Plot of results from various radius values

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also `Alt_Alg` to find the residence times. `Sites` can be used to find the number of identified sites of interest and corresponding maximum percent drop from the csv files produced by `Alt_Alg`.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

##Run the algorithm for multiple radii values
Radii=seq(0.2,1.0,0.1)
for (R in Radii){
  Alt_Alg("OU_14",t,X,Y,R,first='y',save='y')}

##Plot the results (i.e. maximum percent drop and number of sites identified) from applying
##the algorithm with various radius values
plot_radii_results("OU_14",Radii,first='y')

##The location of the legend can be changed
plot_radii_results("OU_14",Radii,first='y', legend_loc="bottomleft")

##Reset the original working directory
setwd(wd)
```

`print_colour_assignment` *Prints a summary table of results from the different criteria*

Description

Prints a summary of the results from using the stability criterion and threshold criterion as well as the colour assigned.

Usage

```
print_colour_assignment(Name, Threshold, Radii, first = 'n', number_sites = -1)
```

Arguments

`Name`	name of the data, which is used for any saved files and plot titles
`Threshold`	threshold value
`Radii`	set of radius values
`first`	if `'y'`, the algorithm will look for the second greatest maximum percent drop if the first results in the first circle being the only non-identified site
`number_sites`	number of sites to manually show the results for

Details

This function prints a clear summary of the results from using both the stability and threshold criteria, including the maximum percent drop and number of identified sites. The threshold criterion requires that the maximum percent drop must be greater than a given value (`Threshold`) and also be a local maximum. The stability criterion requires again that the maximum percent drop be a local maximum and also that the radius values either side, result in the same number of sites identified.

The colour assigned is also printed, where Green is assigned if the two criteria result in the same radius, Amber is assigned if the number of sites are the same for the two criteria, but not the same radius and Red is assigned if the number of sites are different. This gives a qualitative level of confidence in the results produced.

Value

Summary table of the colour assignment part of the algorithm.

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also `Alt_Alg` to find the residence times. `Sites` can be used to find the maximum percent drop and number of sites of interest from the csv files produced by `Alt_Alg`.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

##Run the algorithm for multiple radius values
Radii=seq(0.2,1.0,0.1)
for (R in Radii){
  Alt_Alg("OU_14",t,X,Y,R,first='y',save='y')}

##Print a summary table of the results from the two criteria and the colour assigned
print_colour_assignment("OU_14",65,Radii,first='y')

##Reset the original working directory
setwd(wd)
```

`print_sites_pos` *Prints the positions of the identified sites*

Description

Prints the positions of the identified sites of interest, which are defined by the radius and circle centre.

Usage

```
print_sites_pos(Name,R, first = 'n', number_sites=-1, save="n")
```

Arguments

`Name`	name of the data, which is used for any saved files and plot titles
`R`	radius value to use
`first`	if `'y'`, the algorithm will look for the second greatest maximum percent drop if the first results in the first circle being the only non-identified site
`number_sites`	number of sites to manually show the results for
`save`	if `'y'`, the results will be saved as a csv file

Details

For a given radius value the already identified sites' positions are clearly printed as a table, where their positions are given by the x and y coordinates of the circle centres. There is also the option of saving the data in a csv file, with the title '*Name*'_sites_R'*R*'.csv.

Value

Prints a summary of the site positions.

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See `Alt_Alg` to find the residence times. `Sites` can be used to find the coordinates of the site's centres and number of identified sites of interest from the csv files produced by `Alt_Alg`.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

##Calculate the residence time with a radius of 0.3 and not including the first circle
Alt_Alg("OU_14",t,X,Y,0.3,first='y',save='y')

##Print the coordinates of the centres of all identified sites
print_sites_pos("OU_14",0.3,first='y')

##There is also the option of saving the results as a csv file
print_sites_pos("OU_14",0.3,first='y')

##Reset the original working directory
setwd(wd)
```

---

print_site_visits *Prints the site visitation results*

Description

Prints a summary of the site positions and the time spent at and in between each site.

Usage

```
print_site_visits(Name, X, Y, R, first = 'n', number_sites = -1, save = 'n')
```

Arguments

Name	name of the data, which is used for any saved files and plot titles
X	array of the x-coordinates describing the trajectory
Y	array of the y-coordinates describing the trajectory
R	radius value to use
first	if 'y', the algorithm will look for the second greatest maximum percent drop if the first results in the first circle being the only non-identified site
number_sites	number of sites to manually show the results for
save	if 'y', the results will be saved as a csv file

Details

A summary is printed including the position of the site centres in the order that they are visited. The time spent at each site for each visit is also included as well as the time spent in between sites. There is also the option of saving the data in a csv file, with the title '*Name*'_visits_R'*R*'.csv.

Value

A summary table of the sites visited.

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also `Alt_Alg` to find the residence times. `Sites` can be used to find the coordinates of the centres of the sites and non-overlapping circles from the csv files produced by `Alt_Alg`. Then `plot.schematic` gives a visual representation of the order of sites visited.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

##Calculate the residence time with a radius of 0.3 and not including the first circle
Alt_Alg("OU_14",t,X,Y,0.3,first='y',save='y')

##Prints a summary of the site visitation results
print_site_visits("OU_14",X,Y,0.3,first='y')

##There is also the option of saving the results as a csv file
print_site_visits("OU_14",X,Y,0.3,first='y',save='y')

##Reset the original working directory
setwd(wd)
```

`Sites` *Finds the number of sites of interest from already calculated residence times*

Description

Finds the number of sites of interest as well as other information from already calculated residence times.

Usage

```
Sites(Name, R, first = 'n', number_sites = -1)
```

Arguments

`Name`	name of the data, which is used for any saved files and plot titles
`R`	radius value to use
`first`	if `'y'`, the algorithm will look for the second greatest maximum percent drop if the first results in the first circle being the only non-identified site
`number_sites`	number of sites to manually show the results for

Details

This function finds all the necessary information from the results of the already applied algorithm `Alt_Alg`. It returns information relating to both the non-overlapping circles and also the identified sites, which is then used by other functions. The information is extracted from the already saved csv files.

Value

sites_index - array of indices of the sites of interest among all the circles

N_no_overlap - number of non-overlapping circles

X_no_overlap - x-coordinates of non-overlapping circles

Y_no_overlap - y-coordinates of non-overlapping circles

X_sites - x-coordinates of identified sites of interest

Y_sites - y-coordinates of identified sites of interest

max_percent_drop - maximum percent drop

number_identified_sites - number of identified sites

psi_sort_no_overlap2 - ordered list of non-overlapping residence times

Author(s)

Rhys Munden <rdmunden1@sheffield.ac.uk>

References

Munden, R., Borger , L., Wilson, R.P., Redcliffe, J., Loison, A., Garel, M. and Potts, J.P. in review. Making sense of ultra-high-resolution movement data: an algorithm for inferring sites of interest.

See Also

See also `Alt_Alg` to find the residence times.

Examples

```
##Find the current working directory
wd = getwd()
##Set the working directory as the temporary one
setwd(tempdir())
##Set the working directory as the temporary one
setwd(tempdir())
##Load the data
data(OU_14)
t=unlist(OU_14["t"])
X=unlist(OU_14["X"])
Y=unlist(OU_14["Y"])

##Calculate the residence time with a radius of 0.3 and not including the first circle
Alt_Alg("OU_14",t,X,Y,0.3,first='y',save='y')

##Calculate all the necessary information to be used elsewhere
Sites("OU_14",0.3,first='y')

##Reset the original working directory
setwd(wd)
```

Index