Package ‘stppSim’

July 24, 2024

Type Package
Title Spatiotemporal Point Patterns Simulation
Version 1.3.4
Author Monsuru Adepeju [cre, aut]
Maintainer Monsuru Adepeju <monsuur2010@yahoo.com>
Description Generates artificial point patterns marked by their spatial and temporal signatures. The resulting point cloud may exhibit inherent interactions between both signatures. The simulation integrates microsimulation (Holm, E., (2017)<doi:10.1002/9781118786352.wbieg0320>) and agent-based models (Bonabeau, E., (2002)<doi:10.1073/pnas.082080899>), beginning with the configuration of movement characteristics for the specified agents (referred to as ‘walkers’) and their interactions within the simulation environment. These interactions (Quaglietta, L. and Porto, M., (2019)<doi:10.1186/s40462-019-0154-8>) result in specific spatiotemporal patterns that can be visualized, analyzed, and used for various analytical purposes. Given the growing scarcity of detailed spatiotemporal data across many domains, this package provides an alternative data source for applications in social and life sciences.

Language en-US
License GPL-3
URL https://github.com/MAnalytics/stppSim
BugReports https://github.com/Manalytics/stppSim/issues/new/choose
Depends R (>= 4.1.0)
Encoding UTF-8
LazyData true
Imports splancs, dplyr, tidyr, magrittr, sf, sp, ks, terra, raster, SiMRiv, data.table, tibble, stringr, lubridate, spatstat.geom, sparr, chron, ggplot2, geosphere, leaflet, methods, cowplot, gstat, otuSummary, progressr, future.apply
artif_spo

Artificial spatial origins

Description

Simulates spatial locations to serve as origins of walkers. If provided, spaces covered by restriction features are avoided. Final origins are assigned probability values indicating the strengths of the origins.
artif_spo

Usage

artif_spo(poly, n_origin=50, restriction_feat = NULL,
         n_foci=5, foci_separation = 10, mfocal = NULL,
         conc_type = "nucleated", p_ratio)

Arguments

poly (An sf or S4 object) a polygon shapefile defining the extent of the landscape
n_origin number of locations to serve as origins for walkers. Default: 50.
restriction_feat (An S4 object) optional shapefile containing features in which walkers cannot
                  walk through. Default: NULL.
n_foci number of focal points amongst the origin locations. The origins to serve as focal points are based on random selection. n_foci must be smaller than n_origins.
foci_separation a value from 1 to 100 indicating the nearness of focal points to one another. A 0 separation indicates that focal points are in close proximity of one another, while a 100 indicates focal points being evenly distributed across space.
mfocal the c(x, y) coordinates of a single point, representing a pre-defined main focal point (origin) in the area. The default is NULL in which a random coordinate is chosen within the polygon area.
conc_type concentration of the rest of the origins (non-focal origins) around the focal ones. The options are "nucleated" and "dispersed".
p_ratio the smaller of the two terms of proportional ratios. For example, a value of 20 implies 20:80 proportional ratios.

Details

The focal origins (n_foci) serve as the central locations (such as, city centres). The foci_separation indicates the nearness of focal origins from one another. The conc_type argument allows a user to specify the type of spatial concentration exhibited by the non-focal origin around the focal ones. If restriction_feat is provided, its features help to prevent the occurrence of any events in the areas occupied by the features.

Value

Returns a list detailing the properties of the generated spatial origins with associated strength (probability) values.

Examples

#load boundary of Camden
load(file = system.file("extdata", "camden.rda",
               package="stppSim"))
boundary = camden$boundary # get boundary
landuse <- camden$landuse
spo <- artif_spo(poly = boundary, n_origin = 50,
restriction_feat = landuse, n_foci=5, foci_separation = 0, mfocal = NULL, conc_type = "dispersed", p_ratio=20)

---

camden_crimes: Records of crimes of Camden Borough of London, UK, 2021 (Source: https://data.police.uk/data/)

**Description**

Data comprising 'Theft' and 'Criminal Damage' records of Camden Borough of London, UK for the year 2021 (Source: https://data.police.uk/). Note: Police.uk data is aggregated at monthly scale (yyyy-mm). But, the data provided here has been disaggregated to daily scale by adding fake 'daily' stamps (to give yyyy-mm-dd). So, caution should be taken when interpreting the results based on full date.

**Usage**

camden_crimes

**Format**

A matrix containing four variables

- x: x coordinate
- y: y coordinate
- date: date of occurrence
- type: types of crime

---

chull_poly: Boundary surrounding a set of points

**Description**

Generates a boundary (polygon) around a set of points, using Convex Hull technique (Eddy, W. F, 1977).

**Usage**

chull_poly(xycoords, crsys = NULL)
compare_areas

Arguments

xycoords (matrix) A 2-column coordinate vectors of points: x - the eastings, and y - the northing.

crsys Optional string specifying the coordinate reference system (crs) of the resulting boundary, e.g., the crs string "+proj=longlat +datum=WGS84" transform the resulting boundary to wgs84 system.

Details

Draws an arbitrary boundary around spatial points by joining the outer-most points by lines.

Value

Returns a "SpatialPolygonsDataFrame" object representing the boundary surround the spatial points

References


Examples

```r
data(xyt_data)
#extract xy coordinates only
xy <- matrix(as.numeric(xyt_data[,1:2]),,2)
bry <- chull_poly(xy, crsys = NULL)
#visualise result
#plot(bry) #to plot
#points(xy[,1], xy[,2], add=TRUE)
```

---

compare_areas  
**Compare two areas**

Description

To compare the sizes of two areas (boundary shapefiles).

Usage

```r
compare_areas(area1, area2,
display_output = FALSE)
```

Arguments

area1 (as spatialPolygons, spatialPolygonDataFrames, or simple features). the polygon object of the first area.

area2 (as spatialPolygons, spatialPolygonDataFrames, or simple features). the polygon object of the second area.

display_output (logical) Whether to print output in the console. Default: FALSE
Details

The function `compare_areas` compares the sizes of two areas (polygon shapefiles). The two shapefiles can be in any `crs`, and any spatial object formats. If enabled, the output (a value) comparing the area of the two polygons is printed. This value can be used to scale some specific spatial parameters, including `n_origin`, `s_threshold`, and `step_length`.

Value

Returns a plot and a text (string) comparing the sizes of two areas.

Examples

```r
#load 'area1' object - boundary of Camden, UK
load(file = system.file("extdata", "camden.rda", package="stppSim"))
camden_boundary = camden$boundary

#load 'area2' - boundary of Birmingham, UK
load(file = system.file("extdata", "birmingham_boundary.rda", package="stppSim"))

#run
compare_areas(area1 = camden_boundary,
area2 = birmingham_boundary, display_output = FALSE)
```

---

**date_checker**

### Date (Format) Checker

**Description**

Checks if date is in a specified format (i.e. `yyyy-mm-dd`).

**Usage**

```r
date_checker(x)
```

**Arguments**

- `x` A date or a vector of date values

**Details**

Returns "TRUE" if all date entries are in the specified format ("yyyy-mm-dd"), and FALSE if at least one date is not in the format.

**Value**

Returns TRUE or FALSE
**extract_coords**

**Examples**

```r
date_list_1 <- c("2021-09-12", "2016-xx-02", "09/08/2012")
date_checker(date_list_1)
#> FALSE (Entries 2 and 3 are incorrect date inputs)
date_list_2 <- c("2021-09-12", "1998-03-09")
date_checker(date_list_2)
#> TRUE
```

---

**Description**

Extracts the bounding (edges) coordinates of a polygon object.

**Usage**

```r
extract_coords(poly)
```

**Arguments**

- `poly` (An sf or S4 object) A polygon shapefile.

**Details**

Given a spatial polygon object, the function extracts its bounding coordinates.

**Value**

Returns 2-column xy coordinates representing points of directional change along the boundary.

**Examples**

```r
#load boundary of Camden
load(file = system.file("extdata", "camden.rda", package="stppSim"))
boundary = camden$boundary # get boundary
extract_coords(poly=boundary)
```
### Description
Models the global temporal pattern, as combining the long-term trend and seasonality.

### Usage
```r
gtp(start_date, trend = "stable", 
slope = NULL, shortTerm = "cyclical", 
fPeak = 90, show.plot = FALSE)
```

### Arguments
- `start_date`: the start date of the temporal pattern. The date should be in the format "yyyy-mm-dd". The GTP will normally cover a 1-year period.
- `trend`: specifies the direction of the long-term trend. Options are: "falling", "stable", and "rising". Default value is "stable".
- `slope`: slope of the long-term trend when an "rising" or "falling" trend is specified. Options: "gentle" or "steep". The default value is set as NULL for the stable trend.
- `shortTerm`: type of short- to medium-term fluctuations (patterns) of the time series. Options are: "cyclical" and "acyclical". Default is "cyclical".
- `fPeak`: first seasonal peak of cyclical short term. Default value is 90. Set as NULL for "acyclical" short term pattern.
- `show.plot`: (logical) Shows `gtp`. Default is FALSE.

### Details
Models the GTP for anchoring the temporal trends and patterns of the point patterns to be simulated.

### Value
Returns a time series (list) of 365 data points representing 1-year global temporal pattern.

### Examples
```r
gtp(start_date = "2020-01-01", trend = "stable", 
slope = NULL, shortTerm = "cyclical", 
fPeak = 90, show.plot = FALSE)
```
make_grids

Make square grids

Description

Generates a system of square grids over an area (boundary shapefile).

Usage

```r
make_grids(poly, size = 350,
show_output = FALSE, interactive = FALSE)
```

Arguments

- **poly** (as spatialPolygons, spatialPolygonDataFrames, or simple features).
  A polygon object over which square grids are to be created.
- **size** Size of square grids to be created. For example, the input size for a 350 by 350 square grids is 350.
- **show_output** (logical) Display the output. Default: FALSE
- **interactive** (logical) to show interactive map of the grids generated. Default: FALSE.

Details

Generates a square grid system in a shapefile format (in the same crs as the input poly). If interactive argument is TRUE, an interactive map is shown from which the centroid coordinates of any grid can be displayed by hovering the mouse over the grid. If internet connection is available on the PC, a basemap (OpenStreetmap) is added to help identify places.

Value

Returns a "SpatialPolygonsDataFrames" object representing a system of square grids covering the polygon area.

Examples

```r
#load boundary of Camden
load(file = system.file("extdata", "camden.rda",
package="stppSim"))
boundary = camden$boundary
make_grids(poly=boundary, size = 350,
show_output = FALSE, interactive = FALSE)
```
NRepeat

Near Repeat calculator using the Knox test

Description

This function uses the Knox test for space-time clustering to quantify the spatio-temporal association between events (Credit: Wouter Steenbeek).

Usage

NRepeat(x, y, time, sds, tds, s_include.lowest = FALSE, s_right = FALSE, t_include.lowest = FALSE, t_right = FALSE, method = "manhattan", nrep = 999, saveSimulations = FALSE, future.seed = TRUE,...)

Arguments

x a vector of x coordinates
y a vector of y coordinates
time a vector of time. This can be of type integer, numeric, or date
sds A vector of break points of the spatial intervals. For example c(0,50,120,300) to specify spatial intervals from 0-50, 50-120, 120-300 meters. Or c(0,50,100,Inf) to specify spatial intervals from 0-50, 50-100, and 100-Inf meters. (More accurately, on the scale of the provided x and y coordinates. For example, data may be projected in feet and thus the distances refer to feet instead of meters).

s_right logical, indicating if the spatial intervals should be closed on the right (and open on the left) or vice versa. Default = FALSE. See vignette("NearRepeat_breaks") for details.

s_include.lowest the descriptions above are ambiguous on how exactly the spatial break points are handled. For example, does c(0,100,200) refer to 0-100, 101-200? Or to 0-99 and 100-199? s_include.lowest follows the arguments of cut (see ?cut). Logical, indicating if a spatial distance equal to the lowest (or highest, for right = FALSE) 'breaks' value should be included. Default = FALSE. See vignette("NearRepeat_breaks") for details.

tds A vector of break points of the temporal intervals. For example c(0,2,4,Inf) to specify temporal intervals from 0-2, 2-4, 4-Inf days.

s_include.lowest logical, indicating if a temporal distance equal to the lowest (or highest, for right = FALSE) 'breaks' value should be included. Default = FALSE.

t_right logical, indicating if the temporal intervals should be closed on the right (and open on the left) or vice versa. Default = FALSE. See vignette("NearRepeat_breaks") for details.

t_include.lowest t.include.lowest follows the arguments of cut (see ?cut). Logical, indicating if a temporal distance equal to the lowest (or highest, for right = FALSE) 'breaks' value should be included. Default = FALSE.

t_right logical, indicating if the temporal intervals should be closed on the right (and open on the left) or vice versa. Default = FALSE. See vignette("NearRepeat_breaks") for details.
The method to calculate the spatial distances between crime events. Methods possible as in the 'dist' function (see ?dist). Default is 'manhattan', which seems to be a fair approximation of the distance travelled by a road network. Alternatively, the user can specify 'euclidean' to get the 'as the crow flies' distance.

The number of replications of the Monte Carlo simulation (default = 999).

Should all simulated contingency tables be saved as a 3-dimensional array? Default = FALSE.

A logical or an integer (of length one or seven), or a list of length(X) with pre-generated random seeds. Default = TRUE. See R package future.apply for details.

(Optional) Additional arguments passed to future_lapply()
# Only do 99 replications
set.seed(38673)
NRepeat(x = mydata$x, y = mydata$y, time = mydata$date,
sds = c(0,0.001,100,Inf), tds = c(0,2,4),
method = "euclidean", nrep = 99)

# The plot() function can be used to plot a Heat Map of Near Repeat results
# based on p-values
set.seed(4622)
myoutput <- NRepeat(x = mydata$x, y = mydata$y, time = mydata$date,
sds = c(0,100,200,300,400), td = c(0,1,2,3,4,5))

# The default range of p-values that will be highlighted (0-.05) can be
# adjusted using the 'pvalue_range' parameter. By default the Knox ratios
# are printed in the cells, but this can be adjusted using the 'text'
# parameter. The default is "knox_ratio". Possible values are "observed",
# "knox_ratio", "knox_ratio_median", "pvalues", or NA.

## End(Not run)

<table>
<thead>
<tr>
<th>poly</th>
<th>Boundary coordinates</th>
</tr>
</thead>
</table>

**Description**

Boundary coordinates of Camden Borough of London

**Usage**

poly

**Format**

A dataframe containing one variable:

- x: x coordinate
- y: y coordinate
**poly_tester**

*Geometry and Coordinate Reference System test of a polygon*

**Description**

Tests whether a polygon has the correct geometry, namely; S4 or sf. Also, tests that there is a valid projection attached to the polygon.

**Usage**

```r
poly_tester(poly)
```

**Arguments**

- `poly` (as `spatialPolygons`, `spatialPolygonDataFrames`, or `simple features`). A spatial polygon object.

**Details**

Returns an error message if the polygon is not in the correct geometry or CRS.

**Value**

Returns error messages, or mute

**Examples**

```r
# load boundary of Camden
load(file = system.file("extdata", "camden.rda", package="stppSim"))
boundary = camden$boundary # get boundary
poly_tester(poly=boundary)
```

---

**psim_artif**

*Stpp from synthetic origins*

**Description**

Generates spatiotemporal point patterns based on a set of synthesized origins.
psim_artif(n_events = 1000, start_date = "2021-01-01", poly, netw = NULL, n_origin, restriction_feat = NULL, field, n_foci, foci_separation, mfocal = NULL, conc_type = "dispersed", p_ratio = 20, s_threshold = 50, step_length = 20, trend = "stable", shortTerm = "cyclical", fPeak = 90, s_band = c(0, 200), t_band = c(1, 5, 10), slope = NULL, interactive = FALSE, show.plot = FALSE, show.data = FALSE, ...)

Arguments

n_events number of points (events) to simulate. Default: 1000. A vector of integer values can be supplied, such as, c(a1, a2, ...), where a1, a2, ... represent different integer values.

start_date the start date of the temporal pattern. The date should be in the format "yyyy-mm-dd". The 'gtp' will normally cover a 1-year period.

poly (An sf or S4 object) a polygon shapefile defining the extent of the landscape.

netw (An sf or S4 object) The network path of the landscape (e.g. road and/or street). Default: NULL. If provided each event is snapped to the closest network path/segment.

n_origin number of locations to serve as origins for walkers. Default: 50.

restriction_feat (An S4 object) optional shapefile containing features in which walkers cannot walk through. Default: NULL.

field a number in the range of [0-1] (i.e. restriction values) assigned to all features; or the name of a numeric field to extract such restriction values for different classes of feature. Restriction value 0 and 1 indicate the lowest and the highest obstructions, respectively. Default: NULL.

n_foci number of focal points amongst the origin locations. The origins to serve as focal points are based on random selection. n_foci must be smaller than n_origins.

definition of the nearness of focal points to one another. A 0 separation indicates that focal points are in close proximity of one another, while a 100 indicates focal points being evenly distributed across space.

mfocal the c(x, y) coordinates of a single point, representing a pre-defined main focal point (origin) in the area. The default is NULL in which a random coordinate is chosen within the polygon area.

conc_type concentration of the rest of the origins (non-focal origins) around the focal ones. The options are "nucleated" and "dispersed".

p_ratio the smaller of the two terms of proportional ratios. For example, a value of 20 implies 20:80 proportional ratios.

s_threshold defines the spatial perception range of a walker at a given location. Default: 250 (in the same linear unit as the poly - polygon shapefile).

step_length the maximum step taken by a walker from one point to the next.
psim_artif

- **trend**: specifies the direction of the long-term trend. Options are: "falling", "stable", and "rising". Default value is "stable".

- **shortTerm**: type of short- to medium-term fluctuations (patterns) of the time series. Options are: "cyclical" and "acyclical". Default is "cyclical".

- **fPeak**: first seasonal peak of cyclical short term. Default value is 90. Only used for "cyclical" short term pattern.

- **s_band**: distance bandwidth within which the event re-occurrences are maximized (i.e., interactions are maximum). Specified as a vector of two distance values. Default: c(0, 200).

- **t_band**: temporal bandwidth within which event re-occurrences are maximized (i.e., interactions are maximum). Specified as a vector of values (in days) c(1, 5, 7, 14).

- **slope**: slope of the long-term trend when an "rising" or "falling" trend is specified. Options: "gentle" or "steep". The default value is set as NULL for the stable trend.

- **interactive**: Whether to run the process in interactive mode. Default is FALSE. If TRUE, a user is able to preview the spatial and temporal models of the expected distribution of the final simulated events (points).

- **show.plot**: (logical) Shows GTP. Default is FALSE.

- **show.data**: (TRUE or FALSE) To show the output data. Default is FALSE.

- **...**: additional arguments to pass from gtp, walker and artif_spo functions.

**Details**

Simulate artificial spatiotemporal patterns and interactions based user specifications.

**Value**

Returns a list of artificial spatiotemporal point patterns based on user-defined parameters.

**Examples**

```r
## Not run:

# load boundary and land use of Camden
# load(file = system.file("extdata", "camden.rda",
# package="stppSim"))
# boundary = camden$boundary # get boundary
# landuse = camden$landuse # get landuse
boundary <- stppSim:::boundary
landuse <- stppSim:::landuse
# In this example, we will use a minimal number of
# 'n_origin' (i.e. '20') for faster computation:

# simulate data
simulated_stpp <- psim_artif(n_events=200, start_date = "2021-01-01",
poly=boundary, netw = NULL, n_origin=20, restriction_feat = NULL,
field = NULL,
```

```r
```
psim_real

Stpp from real (sample) origins

Description

Generates spatiotemporal point pattern from origins sampled based on real sample dataset.

Usage

psim_real(n_events, ppt, start_date = NULL, poly = NULL, netw = NULL, s_threshold = NULL, step_length = 20, n_origin=50, restriction_feat=NULL, field=NA, p_ratio=20, interactive = FALSE, s_range = 150, s_interaction = "medium", tolerance = 0.07, crsys = NULL)

Arguments

n_events number of points (events) to simulate. Default: 1000. A vector of integer values can be supplied, such as, c(a1, a2, ...), where a1, a^2, ... represent different integer values.

ppt A 3-column matrix or list containing x - eastings, y - northing, and t - time of occurrence (in the format: 'yyyy-mm-dd')
**start_date**

the start date of the temporal pattern. The date should be in the format "yyyy-mm-dd". The temporal pattern will normally cover 1-year period.

**poly**

(An sf or S4 object) a polygon shapefile defining the extent of the landscape

**netw**

(An sf or S4 object) The network path of the landscape (e.g. road and/or street). Default: NULL. If provided each event is snapped to the closest network path/segment.

**s_threshold**

defines the spatial perception range of a walker at a given location. Default: 250 (in the same linear unit as the poly - polygon shapefile).

**step_length**

the maximum step taken by a walker from one point to the next.

**n_origin**

number of locations to serve as origins for walkers. Default: 50.

**restriction_feat**

(An S4 object) optional shapefile containing features in which walkers cannot walk through. Default: NULL.

**field**

a number in the range of [0-1] (i.e. restriction values) assigned to all features; or the name of a numeric field to extract such restriction values for different classes of feature. Restriction value 0 and 1 indicate the lowest and the highest obstructions, respectively. Default: NULL.

**p_ratio**

the smaller of the two terms of proportional ratios. For example, a value of 20 implies 20:80 proportional ratios.

**interactive**

Whether to run the process in interactive mode. Default is FALSE. If TRUE, a user is able to preview the spatial and temporal models of the expected distribution of the final simulated events (points).

**s_range**

A value (in metres), not less than 150, specifying the maximum range of spatial interaction across the space. For example, for 150m, the intervals of spatial interactions are created as (0, 50], (50~100], and (100~150], representing the "small", "medium", and "large", spatial interaction ranges, respectively. If s_range is set as NULL, simulation focusses only on generating point pattern with similar spatiotemporal patterns as the sample dataset.

**s_interaction**

(string) indicating the type of spatial interaction to detect. Default: "medium" (See parameter 's_range')

**tolerance**

P-value to use for the extraction of space-time interaction in the sample data. Default value: 0.05.

**crsys**

(string) the EPSG code of the projection system of the ppt coordinates. This is only used if poly argument is NULL. See "http://spatialreference.org/" for the list of EPSG codes for different regions of the world. As an example, the EPSG code for the British National Grid projection system is: "EPSG:27700".

**Details**

The spatial and temporal patterns and interactions detected in sample datasets are extrapolated to synthetise larger data size. Details of the spatiotemporal interactions detected in the sample dataset are provided. If the street network of the area is provided, each point is snapped to its nearest street segment.

**Value**

A list of artificial spatiotemporal point patterns and interaction generated based on a sample (real) data.
References


Examples

```r
## Not run:
data(camden_crimes)
# subset 'theft' crime
theft <- camden_crimes[which(camden_crimes$type == "Theft"),]
# specify the proportion of full data to use
sample_size <- 0.3
set.seed(1000)
dat_sample <- theft[sample(1:nrow(theft),
round((sample_size * nrow(theft)), digits=0),
replace=FALSE),1:3]
#plot(dat_sample$x, dat_sample$y) # preview

# load boundary and land use of Camden
# load(file = system.file("extdata", "camden.rda",
# package="stppSim"))
landuse <- camden$landuse # get landuse
landuse <- stppSim:::landuse

# simulate data
simulated_stpp <- psim_real(n_events=2000, ppt=dat_sample,
start_date = NULL, poly = NULL, netw = NULL, s_threshold = NULL,
step_length = 20, n_origin=20,
restriction_feat = NULL, field=NULL,
p_ratio=20, interactive = FALSE, s_range = 150,
s_interaction = "medium", tolerance = 0.07,
crsys = "EPSG:27700")
# If `n_events` is a vector of values,
# retrieve the simulated data for the
# corresponding vector element by using
# `simulated_stpp[[enter-element-index-here]]`, e.g.,
# to retrieve the first dataframe, use
# simulated_stpp[[1]].

#The above example simulates point patterns on
# an unrestricted landscape. If `restriction_feat = landuse` and `field = "restrVal"`,
# then the simulation
# is run with the landuse features as restrictions
# on the landscape.

## End(Not run)
```

---

**p_prob**

Proportional (probability) distribution
Description
Generates an \( n \) probability values in accordance with a specified proportional ratios.

Usage
\[
p_{\text{prob}}(n, \ p_{\text{ratio}} = 20)
\]

Arguments
- \( n \) a number of data points.
- \( p_{\text{ratio}} \) the smaller of the terms of specified proportional ratios. For instance, for a 30:70 ratio, \( p_{\text{ratio}} \) is equal to 30. Default value is set as 20. Valid \( p_{\text{ratio}} \) values are: (5, 10, 20, 30, 40).

Details
Proportional ratios are used to divide the area under curve (auc) of an exponential function such that for any given percentage ratios \( a:b \), the auc is divided into \( b:a \).

Value
Returns a dataframe with a probability field.

Examples
\[
p_{\text{prob}}(n = 15, \ p_{\text{ratio}} = 20)
\]

Description
Snapping point to network

Snaps points to the nearest segment of a network data.

Usage
\[
snap_{\text{points_to_lines}}(\text{points}, \lines, \verbbose = \text{FALSE})
\]

Arguments
- \( \text{points} \) point data (sf object)
- \( \lines \) line/street/road network (sf object)
- \( \verbbose \) Whether to output processing messages.
Details

Function snaps points (within 300m) to the nearest segment on a network. The remaining points outside 300m buffer are returned in their original locations (Credit: Michal Kvasnicka)

Value

Point (sf object) with adjusted coordinates to fit on the network data

Examples

```r
# get line and point data
# load(file = system.file("extdata", "camden.rda",
# package="stppSim"))
lines <- stppSim:::lines
ts <- stppSim:::pts
my_points <- snap_points_to_lines(points=pts,
lines=lines,
verbose = FALSE)

# preview result
# ggplot()+
# geom_sf(data = lines, col = 'red')+ # geom_sf(data = pts, shape = 1)
```

---

### space_restriction

**Space restriction raster map**

Description

Builds a space restriction map from one or more shapefiles. A space restriction raster map showing the restriction levels of various features across the landscape. The function builds on raster- and SimRlv-packages.

Usage

```r
space_restriction(shp, baseMap, res, binary = is.na(field),
field = NA, background = 1)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>shp</td>
<td>shapefile object containing features to serve as obstructions to the movement of walkers.</td>
</tr>
<tr>
<td>baseMap</td>
<td>if provided, a raster onto which to stack the restriction features (shp).</td>
</tr>
<tr>
<td>res</td>
<td>the desired pixel resolution of the raster to be created, when baseMap is not provided.</td>
</tr>
<tr>
<td>binary</td>
<td>if TRUE, the shapefile will be rasterized so that all features are assigned a value of 0 (minimum restriction level), and the background is assigned 1 (maximum restriction level).</td>
</tr>
</tbody>
</table>
space_restriction

field

- a number in the range of [0-1] (i.e. restriction values) assigned to all features;
- or the name of a numeric field to extract such restriction values ([0 <= value < 1] for different classes of feature. Restriction value 0 and 1 indicate the lowest and the highest obstructions, respectively. Default: NULL.

background

- the value in the range 0 and 1 to assign to all pixels that are not covered by any shapefile object.

Details

Helps to create a complete space restriction map with cell values ranging from 0 (minimum restriction level) and 1 (maximum restriction level). All other areas not covered by any features are assigned the value of background. When stacking additional features to existing baseMap, only the areas covered by features are updated, while the remaining areas retain the original values of baseMap.

Value

Returns a raster map showing the restriction levels across the landscape.

References


Examples

```r
#load boundary of Camden and land use data
load(file = system.file("extdata", "camden.rda", package="stppSim"))
boundary = camden$boundary # get boundary
restrct_map <- space_restriction(shp = boundary, res = 20, binary = TRUE)
#plot the result
#plot(restrct_space)
#Setting 'restrct_space' raster as basemap, the landuse
#map can now be stacked onto the basemap as follows:
landuse = camden$landuse # get landuse
restrct_Landuse <- space_restriction(shp = landuse, baseMap = restrct_map, res = 20, field = "restrVal", background = 1)
#plot(restrct_Landuse)
```
Spatial and temporal model

Description

To generate graphics depicting the spatial and temporal models of the final simulation

Usage

```
stm(pt, poly, df, crsys = NULL,
    display_output = FALSE)
```

Arguments

- **pt**: a data frame with the first three fields being 'x', 'y', and 'z' information.
- **poly**: (An sf or S4 object) a polygon shapefile defining the extent of a landscape. Default: NULL, in which the spatial extent of pt is utilized.
- **df**: a vector or 1-column data frame containing values for the time series.
- **crsys**: (string) the EPSG code of the projection system of the ppt coordinates. This only used if poly argument is NULL. See "http://spatialreference.org/" for the list of EPSG codes for different regions of the world. As an example, the EPSG code for the British National Grid projection system is: "EPSG:27700".
- **display_output**: (logical) display the output. Default: FALSE

Details

Incorporated into psim_artif and psim_real functions to allow the preview of the spatial and the temporal model of the simulation. The spatial model is the strength distribution of origin which is the likeness of the spatial patterns to be simulated. The temporal model is the preview of the trend and seasonal patterns to be expected from the simulation.

Value

A graphics showing the spatial and temporal model of the simulation.

Examples

```r
## Not run:
#load polygon shapefile
load(file = system.file("extdata", "camden.rda", package="stppSim"))
camden_boundary = camden$boundary
#read xyz data
data(xyz)
#create a time series
t <- seq(0,5,0.5)
df <- data.frame(data = abs(min(sin(t))) + sin(t))
```
stp_learner

#run function
stm(pt = xyz, poly=camden_boundary, df=df, 
crsys = NULL, display_output = FALSE)

## End(Not run)

stp_learner Learning the spatiotemporal properties of a sample data

Description

Learns both the spatial and the temporal properties of a real sample dataset.

Usage

stp_learner(ppt, start_date = NULL, poly = NULL, 
n_origin=50, p_ratio, gridSize = 150, s_range = 150, 
tolerance = 0.07, 
crsys = NULL, show.plot = FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppt</td>
<td>A 3-column matrix or list containing x - eastings, y - northing, and t - time of occurrence (in the format: 'yyyy-mm-dd').</td>
</tr>
<tr>
<td>start_date</td>
<td>the start date of the temporal pattern. The date should be in the format &quot;yyyy-mm-dd&quot;. The temporal pattern will normally cover 1-year period.</td>
</tr>
<tr>
<td>poly</td>
<td>(An sf or S4 object) a polygon shapefile defining the extent of the landscape</td>
</tr>
<tr>
<td>n_origin</td>
<td>number of locations to serve as origins for walkers. Default:50.</td>
</tr>
<tr>
<td>p_ratio</td>
<td>(an integer) The smaller of the two terms of a Pareto ratio. For example, a value of 20 implies a 20:80 Pareto ratio.</td>
</tr>
<tr>
<td>gridSize</td>
<td>the size of square grid to use for discretizing the space. Default is: 150.</td>
</tr>
<tr>
<td>s_range</td>
<td>A value (in metres), not less than 150, specifying the maximum range of spatial interaction across the space. For example, for 150m, the intervals of spatial interactions are created as (0, 50], (50 - 100], and (100-150], representing the &quot;small&quot;, &quot;medium&quot;, and &quot;large&quot;, spatial interaction ranges, respectively. If s_range is set as NULL, simulation focusses only on generating point pattern with similar spatiotemporal patterns as the sample dataset.</td>
</tr>
<tr>
<td>tolerance</td>
<td>Pvalue to use for the extraction of space-time interaction in the sample data. Default value: 0.07.</td>
</tr>
<tr>
<td>crsys</td>
<td>(string) the EPSG code of the projection system of the ppt coordinates. This only used if poly argument is NULL. See &quot;<a href="http://spatialreference.org/">http://spatialreference.org/</a>&quot; for the list of EPSG codes for different regions of the world. As an example, the EPSG code for the British National Grid projection system is: &quot;EPSG:27700&quot;.</td>
</tr>
<tr>
<td>show.plot</td>
<td>(TRUE or FALSE) Whether to show some displays.</td>
</tr>
</tbody>
</table>
Details

Returns an object of the class `real_spo`, storing details of the spatiotemporal properties of the sample data learnt.

Value

an object (list) containing specific spatial and temporal properties of a sample dataset.

References


Examples

```r
## Not run:
#Goal: To learn the ST properties
#of a sample data, for the purpose of
#simulating the full dataset (see `psim_real`).
data(camden_crimes)
#subset 'theft' crime
theft <- camden_crimes[which(camden_crimes$type ==
"Theft"),1:3]
#specify the proportion of full data to use
sample_size <- 0.3
set.seed(1000)
dat_sample <- theft[sample(1:nrow(theft),
round((sample_size * nrow(theft)), digits=0),
replace=FALSE),]
#plot(dat_sample$x, dat_sample$y) #preview
stp_learner(dat_sample,
start_date = NULL, poly = NULL, n_origin=50,
p_ratio=20, gridSize = 150,
s_range = 150, tolerance = 0.07,
crsys = "EPSG:27700",
show.plot = FALSE)
## End(Not run)
```

walker

A landscape walker

Description

A dynamic object capable of moving and avoiding obstacles on a landscape.
**walker**

**Usage**

walker(n = 5, s_threshold = 250, step_length = 20, 
poly = NULL, restriction_feat=NULL, field = NA, coords=c(0,0), 
pt_itx = TRUE, show.plot = FALSE)

**Arguments**

- **n** number of events to be generated by a walker within a temporal bin.
- **s_threshold** defines the spatial perception range of a walker at a given location. Default: 250 (in the same linear unit as the poly - polygon shapefile).
- **step_length** the maximum step taken by a walker from one point to the next.
- **poly** (An sf or S4 object) a polygon shapefile defining the extent of the landscape
- **restriction_feat** (An S4 object) optional shapefile containing features in which walkers cannot walk through. Default: NULL.
- **field** a number in the range of [0-1] (i.e. restriction values) assigned to all features; or the name of a numeric field to extract such restriction values for different classes of feature. Restriction value 0 and 1 indicate the lowest and the highest obstructions, respectively. Default: NULL.
- **coords** a vector of the form c(x, y) giving the initial coordinates of a walker (i.e., coordinates of origins). Default value is c(0,0) for an arbitrary square space.
- **pt_itx** To check whether any of the specified initial origin coordinates falls outside the boundary. Default: TRUE.
- **show.plot** (TRUE or False) To show the time series plot. Default is FALSE.

**Details**

A walker is propelled by an in-built stochastic transition matrix and a specified set of spatial and temporal parameters. The transition matrix defines two states, namely; the exploratory and a performative states. A walker is capable of avoiding obstructions (i.e., restriction_feat) if included. The resulting number of events may be slightly different from the value n because of the stochastic process involved.

**Value**

Returns a trace of walker’s path, and the resulting events.

**References**

Examples

```r
# load boundary of Camden
load(file = system.file("extdata", "camden.rda", package="stppSim"))
boundary = camden$boundary # get boundary
walkerpath <- walker(n = 5, s_threshold = 250, step_length = 20,
poly = boundary, restriction_feat=NULL, field = NULL,
coords = c(0,0), pt_itx = TRUE, show.plot = FALSE)
#plot(walkerpath)
```

---

xyt_data

**Spatiotemporal point data**

**Description**

Example spatiotemporal point data of a part of San Francisco City, California, US

**Usage**

```
xyt_data
```

**Format**

A matrix containing three variables

- x: x coordinate
- y: y coordinate
- t: t time

---

xyz

**xyz data**

**Description**

Example data with 'x', 'y', and a 'z' information

**Usage**

```
xyz
```

**Format**

A matrix containing three variables

- x: x coordinate
- y: y coordinate
- z: z height/probability/etc
Index

* datasets
  camden_crimes, 4
  poly, 12
  xyt_data, 26
  xyz, 26

artif_spo, 2

camden_crimes, 4
chull_poly, 4
compare_areas, 5

date_checker, 6
extract_coords, 7

gtp, 8

make_grids, 9

NRepeat, 10

p_prob, 18
poly, 12
poly_tester, 13
psim_artif, 13
psim_real, 16

snap_points_to_lines, 19
space_restriction, 20
stm, 22
stp_learner, 23

walker, 24

xyt_data, 26
xyz, 26