Package: spacetools (via r-universe)

August 27, 2024

Title Easy Spatial Tools for R

Version 0.0.0.9000

Description Provide a series of functions designed to make complicated spatial operations in R much easier. Applications are focused on aquatic distance related uses, with built in functionality for the Sacramento San Joaquin Delta. Users can calculate in-water distances between points, move points outside a shapefile to the closest location within the shapefile, and cluster points within a given radius. More functions will be added in the future.

License GPL-3

Depends R (>= 3.5.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Imports lwgeom, magrittr, dplyr, igraph, gdistance, raster, rlang, sf, tibble, stars, methods, graphics, tidyr, stats

URL https://github.com/sbashevkin/spacetools

BugReports https://github.com/sbashevkin/spacetools/issues

Language en-US

Suggests testthat (>= 2.1.0), spelling, covr

Repository https://sbashevkin.r-universe.dev

RemoteUrl https://github.com/sbashevkin/spacetools

RemoteRef HEAD

RemoteSha b9ac9f56f65e0583b774cb76890615f219325697

Contents

`Delta` *Delta shapefile*

Description

Shapefile of delta waterways

Usage

```
Delta
```

Format

a sf tibble with 282 rows and 10 columns.

AREA Area.

PERIMETER Perimeter.

HYDRO_POLY HYDRO_POLY.

HYDRO_PO_1 HYDRO_PO_1.

HYDRO_24K_ HYDRO_24K_.

HNAME HNAME.

Shape_Leng Shape_Length

Shape_Area Shape_Area.

geometry Shapefile polygon coordinates.

GGdist *In-water distances between a set of points and the Golden Gate*

Description

Calculate a distance matrix for a set of points based on in-water distances, using a raster-based approach.

Usage

```
GGdist(
  Water_map,
  Points,
  EndPoint = NULL,
  Latitude_column,
  Longitude_column,
  PointID_column,
  Points_crs = 4326,
  Water_map_transitioned = NULL,
  Calculation_crs = 32610,
  Grid_size = 75
)
```

Arguments

`Water_map`	Object of class sf representing a map of all waterways in your region of interest
`Points`	A dataframe of points with latitude and longitude which you would like to calculate distance to a given reference point.
`EndPoint`	A dataframe containing a single point with latitude and longitude. Defaults to a point 92 m East of the Golden Gate (37.819539, -122.477). Latitude and longitude column names must be the same as the points data frame. This point must fall within the bounds of the `Water_map`.
`Latitude_column`	The unquoted name of the column in the `Points` and `EndPoint` dataframe representing Latitude.
`Longitude_column`	The unquoted name of the column in the `Points` and `EndPoint` dataframe representing Longitude.
`PointID_column`	The unquoted name of the column in the `Points` dataframe with the unique identifier of each point.
`Points_crs`	Coordinate reference system for your `Points` dataframe. Integer with the EPSG code or character with proj4string.
`Water_map_transitioned`	A rasterized, transitioned, and geo-corrected version of the water map. This is optional to save time if you will be running this function frequently with the same base map.

`Calculation_crs`
: Coordinate reference system used for the calculation. If a latitude/longitude system are used, errors may be returned since some calculations assume a planar surface. Defaults to `Calculation_crs = 32610`.

`Grid_size`
: Grid size (in meters) used to rasterize the map. Defaults to 75.

Value

A tibble with 1 column for the PointID_column values and another with the distances from the EndPoint.

See Also

`Maptransitioner` `Pointmover`

Examples

```
library(tibble)

Points <- tibble(Latitude = c(38.23333, 38.04813, 38.05920,
                                    37.94900, 38.23615, 38.47387),
                         Longitude = c(-121.4889, -121.9149, -121.8684,
                                       -121.5591, -121.6735, -121.5844),
                         ID = c("EMP NZP02", "FMWT 508", "FMWT 513",
                                "FMWT 915", "FMWT 723", "FMWT 796"))

## Not run:
distance<GGdist(Water_map = spacetools::Delta, Points = Points, Latitude_column = Latitude,
                   Longitude_column = Longitude, PointID_column = ID)


# Including a pre-transitioned map to save time.
# See Maptransitioner for creating this pre-transitioned map.

distance<-GGdist(Water_map = spacetools::Delta, Points = Points, Latitude_column = Latitude,
                   Longitude_column = Longitude, PointID_column = ID,
                   Water_map_transitioned = Delta_transitioned)

## End(Not run)
```

`Maptransitioner` *Rasterize and transition a shapefile*

Description

Rasterize and transition a shapefile for input into `Waterdist`

Usage

```
Maptransitioner(
  Water_map,
  Calculation_crs = 32610,
  Grid_size = 75,
  Process_map = TRUE,
  Plot = FALSE
)
```

Arguments

Water_map	Object of class sf representing a map of all waterways in your region of interest
Calculation_crs	Coordinate reference system used for the calculation. If a latitude/longitude system are used, errors may be returned since some calculations assume a planar surface. Defaults to Calculation_crs = 32610.
Grid_size	Grid size (in meters) used to rasterize the map. Defaults to 75.
Process_map	Should the Water_map be processed by unioning all features and transforming to the Calculation_crs? This should almost always be set to TRUE, the default.
Plot	Should the rasterized map be plotted for inspection?

Value

A rasterized, transitioned, and geo-corrected map.

See Also

Waterdist

Examples

```
## Not run:
Map <- Maptransitioner(spacetools::Delta)

## End(Not run)
```

Pointcluster *Cluster nearby points*

Description

Cluster points within a given distance of one another

Usage

```
Pointcluster(
  Points,
  Distance,
  In_water_distance = FALSE,
  Latitude_column,
  Longitude_column,
  PointID_column,
  Points_crs = 4326,
  Calculation_crs = 32610,
  Water_map = NULL,
  Water_map_transitioned = NULL,
  Grid_size = 75,
  Expand = TRUE
)
```

Arguments

`Points`	A dataframe of points with latitude and longitude
`Distance`	Clustering distance in meters.
`In_water_distance`	Should clustering be based on in-water distance? If yes, in-water distances will be calculated with `Waterdist` and this function will be slower. NOTE: For in-water distances, any points outside the `Water_map` shapefile polygons will be moved inside before distances are calculated for clustering. The parameters `Water_map`, `Water_map_transitioned` and `Grid_size` will be ignored unless `In_water_distance = TRUE`.
`Latitude_column`	The unquoted name of the column in the `Points` dataframe representing Latitude.
`Longitude_column`	The unquoted name of the column in the `Points` dataframe representing Longitude.
`PointID_column`	The unquoted name of the column in the `Points` dataframe with the unique identifier of each point.
`Points_crs`	Coordinate reference system for your `Points` dataframe. Integer with the EPSG code or character with proj4string.
`Calculation_crs`	Coordinate reference system used for the calculation. If a latitude/longitude system are used, errors may be returned since some calculations assume a planar surface. Defaults to `Calculation_crs = 32610`.
`Water_map`	Object of class sf representing a map of all waterways in your region of interest
`Water_map_transitioned`	A rasterized, transitioned, and geo-corrected version of the water map. This is optional to save time if you will be running this function frequently with the same base map.

Grid_size	Grid size (in meters) used to rasterize the map. Defaults to 75.
Expand	Should data be expanded at the end? If TRUE (the default), the result will be expanded so there is 1 row per unique value of the PointID_column. If FALSE , the returned object will have 1 row per cluster and a list column with the names of each PointID_column value contained in each cluster.

Details

The parameters Water_map, Water_map_transitioned and Grid_size will be ignored unless In_water_distance = TRUE.

Value

A tibble relating each point to its assigned cluster. The returned Latitude and Longitude values represent the mean values across all points in each cluster.

Examples

```
library(tibble)
Points <- tibble(Latitude = c(38.07194, 38.09306, 38.11722,
                              38.11528, 38.07020, 38.09383,
                              38.11783, 38.06481, 38.11400,
                              38.06750, 38.11556),
                 Longitude = c(-122.0961, -122.0692, -122.0472,
                               -122.0519, -122.0941, -122.0697,
                               -122.0418, -122.0978, -122.0462,
                               -122.0956, -122.0424),
                 ID = c("EMP NZ022", "EMP NZ024", "EMP NZ028",
                        "EMP NZ030", "FMWT 416", "FMWT 418",
                        "FMWT 602", "TNS 418", "TNS 602",
                        "twentymm 418", "twentymm 602"))
Points_clust<-Pointcluster(Points, 1000, FALSE, Latitude, Longitude, ID, Expand=TRUE)
```

Pointmover *Move points within a shapefile*

Description

Move points to closest location within shapefile

Usage

```
Pointmover(Data, Attribute, Shapefile)
```

Arguments

Data	Dataframe of points with sf geometry column and an attribute column with NAs for points that need to be moved.
Attribute	Name of the column in Data that indicates points that need to be moved (by the presence of NAs)
Shapefile	Object of class sf representing the shapefile you want all points to fall within.

See Also

Waterdist

Examples

```
## Not run:
library(tibble)
library(dplyr)
library(sf)
Points <- tibble(Latitude = c(38.23333, 38.04813, 38.05920,
                              37.94900, 38.23615, 38.47387),
                 Longitude = c(-121.4889, -121.9149, -121.8684,
                               -121.5591, -121.6735, -121.5844),
                 ID = c("EMP NZP02", "FMWT 508", "FMWT 513",
                        "FMWT 915", "FMWT 723", "FMWT 796"))%>%
st_as_sf(coords=c("Longitude", "Latitude"), crs=4326)%>%
  st_transform(crs=32610)

Map <- st_union(spacetools::Delta)%>%
st_as_sf()%>%
  mutate(Inside=TRUE)%>%
  rename(geometry = x)%>%
  st_transform(crs=32610)

Points_joined <- st_join(Points, Map, join = st_intersects)

Points_fixed<-Pointmover(Points_joined, Inside, Map)

## End(Not run)
```

---

spacetools *spacetools: Easy spatial tools for R*

Description

This package contains functions for easy spatial operations in R, focused on aquatic distance related applications. Internal datasets for the Sacramento San Joaquin Delta are also included to make operations very easy for this location.

Functions

- `Waterdist`
- `Pointmover`
- `Maptransitioner`
- `Pointcluster`

Internal datasets

- `Delta`
- `Stations`

`Stations` *Zooplankton sampling stations*

Description

Zooplankton sampling stations in the Sacramento San Joaquin Delta from the zooper package.

Usage

```
Stations
```

Format

a tibble with 362 rows and 3 columns

Station Sampling station name

Latitude Latitude in decimal degrees

Longitude Longitude in decimal degrees

See Also

zooper

`Waterdist` *In-water distances between points*

Description

Calculate a distance matrix for a set of points based on in-water distances, using a raster-based approach

Usage

```
Waterdist(
  Water_map,
  Points,
  Latitude_column,
  Longitude_column,
  PointID_column,
  Points_crs = 4326,
  Water_map_transitioned = NULL,
  Calculation_crs = 32610,
  Grid_size = 75
)
```

Arguments

`Water_map`	Object of class sf representing a map of all waterways in your region of interest
`Points`	A dataframe of points with latitude and longitude
`Latitude_column`	The unquoted name of the column in the `Points` dataframe representing Latitude.
`Longitude_column`	The unquoted name of the column in the `Points` dataframe representing Longitude.
`PointID_column`	The unquoted name of the column in the `Points` dataframe with the unique identifier of each point.
`Points_crs`	Coordinate reference system for your `Points` dataframe. Integer with the EPSG code or character with proj4string.
`Water_map_transitioned`	A rasterized, transitioned, and geo-corrected version of the water map. This is optional to save time if you will be running this function frequently with the same base map.
`Calculation_crs`	Coordinate reference system used for the calculation. If a latitude/longitude system are used, errors may be returned since some calculations assume a planar surface. Defaults to `Calculation_crs = 32610`.
`Grid_size`	Grid size (in meters) used to rasterize the map. Defaults to 75.

Value

Distance matrix.

See Also

Maptransitioner Pointmover

Examples

```
library(tibble)

Points <- tibble(Latitude = c(38.23333, 38.04813, 38.05920,
                                      37.94900, 38.23615, 38.47387),
                         Longitude = c(-121.4889, -121.9149, -121.8684,
                                       -121.5591, -121.6735, -121.5844),
                         ID = c("EMP NZP02", "FMWT 508", "FMWT 513",
                                "FMWT 915", "FMWT 723", "FMWT 796"))

## Not run:
distance<-Waterdist(Water_map = spacetools::Delta, Points = Points, Latitude_column = Latitude,
                    Longitude_column = Longitude, PointID_column = ID)


# Including a pre-transitioned map to save time.
# See Maptransitioner for creating this pre-transitioned map.

distance<-Waterdist(Water_map = spacetools::Delta, Points = Points, Latitude_column = Latitude,
                    Longitude_column = Longitude, PointID_column = ID,
                    Water_map_transitioned = Delta_transitioned)

## End(Not run)
```

Index