Package ‘ipdw’

January 5, 2023

Title Spatial Interpolation by Inverse Path Distance Weighting

Description Functions are provided to interpolate geo-referenced point data via Inverse Path Distance Weighting. Useful for coastal marine applications where barriers in the landscape preclude interpolation with Euclidean distances.

Version 2.0-0

URL https://github.com/jsta/ipdw

BugReports https://github.com/jsta/ipdw/issues

Depends R (>= 3.0.2), gdistance

Imports sf, raster, methods

Suggests gstat, gdata, spatstat, testthat, knitr, rmarkdown

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.2.3

VignetteBuilder knitr

NeedsCompilation no

Author Jemma Stachelek [aut, cre] (<https://orcid.org/0000-0002-5924-2464>)

Maintainer Jemma Stachelek <jemma.stachelek@gmail.com>

Repository CRAN

Date/Publication 2023-01-05 17:50:05 UTC

R topics documented:

costRasterGen .......................................................... 2
errorGen ................................................................. 3
ipdw ................................................................. 4
ipdwInterp ............................................................ 5
pathDistGen .......................................................... 6
rm_na_pointsLayers ................................................. 7

Index
costrasterGen

Generate a cost Raster

Description
Generate a cost raster from an object of class sf with point or polygon geometries

Usage
costrasterGen(xymat, pols, extent = "polys", projstr, resolution = 1)

Arguments
xymat Matrix of coordinates or an sf object with point geometries
pols sf object with polygon geometries
extent Define extent based on extent of xymat/sf (points) or pols (polys). Default is polys.
projstr proj4 string defining the output projection. A warning will be thrown if projstr does not match the projection of the extent target. Pass NULL for non-geographic grids.
resolution Numeric defaults to 1. See raster.

Details
Ensure that the projection of the xymat coordinates and pols match. If they do not match use the st_transform command.

Value
RasterLayer

See Also
rasterize

Examples
## Not run:
library(sf)
Sr1 <- st_polygon(list(cbind(c(0, 0, 1, 1, 0), c(0, 12, 12, 0, 0))))
Sr4 <- st_polygon(list(cbind(c(9, 9, 10, 10, 9), c(0, 12, 12, 0, 0))))
Sr2 <- st_polygon(list(cbind(c(1, 1, 9, 9, 1), c(11, 12, 12, 11, 11))))
Sr3 <- st_polygon(list(cbind(c(1, 1, 9, 9, 1), c(0, 1, 1, 0, 0))))
Sr5 <- st_polygon(list(cbind(c(4, 4, 5, 5, 4), c(4, 8, 8, 4, 4))))
pols <- st_as_sf(st_sfc(Sr1, Sr2, Sr3, Sr4, Sr5,
crs = "+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0"))

# using a matrix object
errorGen

```r
xymat <- matrix(3, 3, nrow = 1, ncol = 2)
costras <- costrasterGen(xymat, pols, projstr = NULL)

# plotting
plot(costras)
points(xymat)
```

## End(Not run)

---

**errorGen**

*Generate interpolation error stats from validation datasets*

**Description**

Generate error statistics from validation point datasets overlaid on a raster surface

**Usage**

```r
errorGen(
  finalraster,
  validation.sf_ob,
  validation.data,
  plot = FALSE,
  title = ""
)
```

**Arguments**

- `finalraster`  
  RasterLayer object
- `validation.sf_ob`  
  sf object with points geometry
- `validation.data`  
  data.frame
- `plot`  
  logical. Plot comparison?
- `title`  
  Plot labels

**Value**

List of error statistics

**Examples**

```r
library(sf)
validation.data <- data.frame(rnorm(10, mean = 0.2, sd = 1))
names(validation.data) <- c("validation")
validation.sf_ob <- validation.data
validation.data <- as.numeric(unlist(validation.data))
xy <- data.frame(x = c(0:9), y = rep(1, 10))
```
validation.sf_ob <- st_as_sf(cbind(validation.sf_ob, xy), coords = c("x", "y"))

m <- matrix(NA, 1, 10)
out.ras <- raster(m, xmn = 0, xmx = ncol(m), ymn = 0, ymx = nrow(m))
out.ras[] <- validation.data + rnorm(ncell(out.ras), mean = 0.01, sd = 0.2)
valid.stats <- errorGen(out.ras, validation.sf_ob, validation.data, plot = TRUE, title = "Validation Plot")
valid.stats

---

ipdw

**Inverse Path Distance Weighting**

**Description**

Interpolate geo-referenced point data using inverse path distance weighting.

**Usage**

```r
ipdw(
  sf_ob, 
  costras, 
  range, 
  paramlist, 
  overlapped = FALSE, 
  yearmon = "default", 
  removefile = TRUE, 
  step = 16, 
  dist_power = 1, 
  trim_rstack = FALSE 
)
```

**Arguments**

- `sf_ob` : sf object with point geometries
- `costras` : RasterLayer. Cost raster
- `range` : numeric. Range of interpolation neighborhood
- `paramlist` : character. String representing parameter names
- `overlapped` : logical. Default is FALSE, specify TRUE if some points lie on top of barriers
- `yearmon` : character. String specifying the name of the sf_ob
- `removefile` : logical. Remove files after processing?
- `step` : numeric. Number of sub loops to manage memory during raster processing.
- `dist_power` : numeric. Distance decay power (p)
- `trim_rstack` : logical. Trim the raster output by the convex hill of sf_ob
ipdwInterp

Details
This is a high level function that interpolates an sf object with point geometries in a single pass. Points must be located within a single contiguous area. The presence of "landlocked" points will cause errors. It may be necessary to increase the value assigned to land areas when using a large range value in combination with a large sized cost rasters (grain x extent). In these cases, the value of land areas should be increased to ensure that it is always greater than the maximum accumulated cost path distance of any given geo-referenced point.

Value
RasterLayer

Examples
# see vignette

ipdwInterp

Inverse Distance Weighting with custom distances

Description
This function takes a rasterstack of pathdistances and generates surfaces by weighting parameter values by these distances

Usage
ipdwInterp(
  sf_ob,
  rstack,
  paramlist,
  overlapped = FALSE,
  yearmon = "default",
  removefile = TRUE,
  dist_power = 1,
  trim_rstack = FALSE
)

Arguments
  sf_ob    sf object with point geometries
  rstack   RasterStack of path distances
  paramlist character. String representing parameter names
  overlapped logical. Default is FALSE, specify TRUE if some points lie on top of barriers
  yearmon  character. String specifying the name of the sf object
  removefile logical. Remove files after processing?
  dist_power numeric. Distance decay power (p)
  trim_rstack logical. Trim the raster stack by the convex hull of sf_ob
Details

Under the hood, this function evaluates:

\[
V = \frac{\sum_{i=1}^{n} \frac{1}{d_i^p}}{\sum_{i=1}^{n} 1}
\]

where \(d\) is the distance between prediction and measurement points, \(v_i\) is the measured parameter value, and \(p\) is a power parameter.

Value

RasterLayer

Examples

```r
library(sf)
sf_ob <- data.frame(rnorm(2))
xy <- data.frame(x = c(4, 2), y = c(8, 4))
sf_ob <- st_as_sf(cbind(sf_ob, xy), coords = c("x", "y"))
m <- matrix(NA, 10, 10)
costras <- raster(m, xmn = 0, xmx = ncol(m), ymn = 0, ymx = nrow(m))

# introduce spatial gradient
gostra[i] <- runif(10(i), min = 1, max = 10)
for (i in 1:nrow(costras)) {
  costras[i, ] <- costras[i, ] + i
  costras[, i] <- costras[, i] + i
}

rstack <- pathdistGen(sf_ob, costras, 100, progressbar = FALSE)
final.raster <- ipdwInterp(sf_ob, rstack, paramlist = c("rnorm.2".), overlapped = TRUE)
plot(final.raster)
plot(sf_ob, add = TRUE)
```

---

**pathdistGen**

*Generate a stack of path distance raster objects*

Description

Generate a stack of path accumulated distance raster objects

Usage

`pathdistGen(sf_ob, costras, range, yearmon = "default", progressbar = TRUE)`
Arguments

- `sf_ob`: sf object with point geometries
- `costras`: RasterLayer cost raster
- `range`: numeric. Range of interpolation neighborhood
- `yearmon`: character. String specifying the name of the `sf_ob`
- `progressbar`: logical. Show progressbar during processing?

Value

RasterStack object of path distances

Examples

```r
library(sf)
sf_ob <- data.frame(rnorm(2))
xy <- data.frame(x = c(4, 2), y = c(8, 4))
sf_ob <- st_as_sf(cbind(sf_ob, xy), coords = c("x", "y"))

m <- matrix(NA, 10, 10)
costras <- raster(m, xmn = 0, xmx = ncol(m), ymn = 0, ymx = nrow(m))
costras[] <- runif(ncell(costras), min = 1, max = 10)
# introduce spatial gradient
for (i in 1:nrow(costras)) {
  costras[i, ] <- costras[i, ] + i
  costras[, i] <- costras[, i] + i
}

rstack <- pathdistGen(sf_ob, costras, 100, progressbar = FALSE)
```

---

**rm_na_pointslayers**: Remove NA points features and drop corresponding raster stack layers

Description

Remove NA points features and drop corresponding raster stack layers

Usage

`rm_na_pointslayers(param_name, sf_ob, rstack)`

Arguments

- `param_name`: character name of data column
- `sf_ob`: sf object with point geometries
- `rstack`: RasterStack or RasterBrick
Index

costRasterGen, 2
errorGen, 3
ipdw, 4
ipdwInterp, 5
pathdistGen, 6
raster, 2
rasterize, 2
rm_na_pointslayers, 7