Package ‘ipdw’

October 13, 2022

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  1.0-0

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

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

Depends  R (>= 3.0.2), gdistance

Imports  sp, raster, rgeos, methods

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

License  GPL (>= 2)

Encoding  UTF-8

RoxygenNote  7.2.0

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  2022-07-01 21:30:14 UTC

R topics documented:

costrasterGen .......................................................... 2
errorGen ................................................................. 3
ipdw ................................................................. 4
ipdwInterp .............................................................. 5
pathdistGen ........................................................... 7
rm_na_pointslayers .................................................. 8

Index  9
costrasterGen  Generate a cost Raster

Description

Generate a cost raster from an object of class SpatialPolygons, matrix, or SpatialPointsDataFrame

Usage

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

Arguments

xymat  Matrix of coordinates or a SpatialPointsDataFrame object
pols  SpatialPolygons object
extent  Define extent based on extent of xymat/xyspdf (points) or pols (polys). Default is polys.
projstr  proj4 string defining the output projection. An error 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. This can be accomplished by running the projection command on both inputs. If they do not match use the spTransform command.

Value

RasterLayer

See Also

spTransform-methods, rasterize

Examples

## Not run:
Sr1 <- Polygon(cbind(c(0, 0, 1, 1, 0), c(0, 12, 12, 0, 0)))
Sr4 <- Polygon(cbind(c(9, 9, 10, 10, 9), c(0, 12, 12, 0, 0)))
Sr2 <- Polygon(cbind(c(1, 1, 9, 9, 1), c(11, 12, 12, 11, 11)))
Sr3 <- Polygon(cbind(c(1, 1, 9, 9, 1), c(0, 1, 1, 0, 0)))
Sr5 <- Polygon(cbind(c(4, 4, 5, 5, 4), c(4, 8, 8, 4, 4)))
Srs1 <- Polygons(list(Sr1), "s1")
Srs2 <- Polygons(list(Sr2), "s2")
Srs3 <- Polygons(list(Sr3), "s3")
errorGen <- Polygons(list(Sr4), "s4")
Srs5 <- Polygons(list(Sr5), "s5")
pols <- SpatialPolygons(list(Srs1, Srs2, Srs3, Srs4, Srs5), 1:5)

# using a matrix object
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.spdf, validation.data,
  plot = FALSE, title = ""
)
```

**Arguments**

- `finalraster` RasterLayer object
- `validation.spdf` SpatialPointsDataFrame
- `validation.data` data.frame
- `plot` logical. Plot comparison?
- `title` Plot labels

**Value**

List of error statistics
Examples

validation.data <- data.frame(rnorm(10, mean = 0.2, sd = 1))
names(validation.data) <- c("validation")
validation.spdf <- validation.data
validation.data <- as.numeric(unlist(validation.data))
xy <- data.frame(x = c(0:9), y = rep(1, 10))
coordinates(validation.spdf) <- xy

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.spdf, 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

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spdf</td>
<td>SpatialPointsDataFrame object</td>
</tr>
<tr>
<td>costras</td>
<td>RasterLayer. Cost raster</td>
</tr>
<tr>
<td>range</td>
<td>numeric. Range of interpolation neighborhood</td>
</tr>
<tr>
<td>paramlist</td>
<td>character. String representing parameter names</td>
</tr>
<tr>
<td>overlapped</td>
<td>logical. Default is FALSE, specify TRUE if some points lie on top of barriers</td>
</tr>
<tr>
<td>yearmon</td>
<td>character. String specifying the name of the spdf</td>
</tr>
</tbody>
</table>
ipdwInterp

**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 spdf

**Details**

This is a high level function that interpolates a SpatialPointsDataFrame object 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**

```r
# 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**

```r
ipdwInterp(
    spdf,
    rstack,
    paramlist,
    overlapped = FALSE,
    yearmon = "default",
    removefile = TRUE,
    dist_power = 1,
    trim_rstack = FALSE
)
```
Arguments

spdf  
spatialPointsDataFrame object
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 spdf
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 spdf

Details

Under the hood, this function evaluates:

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

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
spdf <- data.frame(rnorm(2))
xy <- data.frame(x = c(4, 2), y = c(8, 4))
coordinates(spdf) <- xy
m <- matrix(NA, 10, 10)
costras <- raster(m, xmn = 0, xmx = ncol(m), ymn = 0, ymx = nrow(m))
# introduce spatial gradient
costras[] <- runif(ncell(costras), min = 1, max = 10)
for (i in 1:nrow(costras)) {
  costras[i, ] <- costras[i, ] + i
  costras[, i] <- costras[, i] + i
}

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

Generate a stack of path distance raster objects

Description

Generate a stack of path accumulated distance raster objects

Usage

pathdistGen(spdf, costras, range, yearmon = "default", progressbar = TRUE)

Arguments

- `spdf`: SpatialPointsDataFrame object
- `costras`: RasterLayer cost raster
- `range`: numeric. Range of interpolation neighborhood
- `yearmon`: character. String specifying the name of the spdf
- `progressbar`: logical show progressbar during processing?

Value

RasterStack object of path distances

Examples

```r
spdf <- data.frame(rnorm(2))
xy <- data.frame(x = c(4, 2), y = c(8, 4))
coordinates(spdf) <- xy

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(spdf, costras, 100, progressbar = FALSE)
```
**rm_na_pointslayers**

*Remove NA SpatialPointsDataFrame features and drop corresponding raster stack layers*

**Description**

Remove NA SpatialPointsDataFrame features and drop corresponding raster stack layers

**Usage**

```r
rm_na_pointslayers(param_name, spdf, rstack)
```

**Arguments**

- **param_name**: character name of data column
- **spdf**: SpatialPointsDataFrame object
- **rstack**: RasterStack or RasterBrick
Index

costrasterGen, 2
errorGen, 3
ipdw, 4
ipdwInterp, 5
pathdistGen, 7
raster, 2
rasterize, 2
rm_na_pointsLayers, 8