Package 'GeoCleanR'

January 7, 2020

Title Functions and Packages to Process Geospatial Data in R

Version 0.1.3

Date 2020-01-07

Author Jordan Adamson [aut, cre]

Maintainer Jordan Adamson <jordan.m.adamson@gmail.com>

Description

License MIT + file LICENSE

URL <https://github.com/Jadamso/GeoCleanR>

Encoding UTF-8

LazyData true

Published 2020-01-07

Roxygen list(markdown = TRUE)

Imports raster,
sp,
rgeos,
rgdal,
maptools,
spam,
spam64,
gdalUtils,
fields,
cleangeo

RoxygenNote 7.0.2

R topics documented:

`agg_fun` *Aggregate Raster*

Description

Aggregate Raster

Usage

```
agg_fun(i, rast)
```

Arguments

`i`	aggregation factor
`rast`	raster being aggregated

Value

raster

COUNTIESm *Functions to read in County Shapefiles*

Description

Functions to read in County Shapefiles

Usage

```
COUNTIESm(myears, rddir)

COUNTIESmm(mmyears, rddir)

COUNTIESh(hyears, ddir)
```

Arguments

myears	years to grab
rdir	from what directory

Details

Modern Counties
Middle Modern Counties
Historical Counties from 1690:2000

Value

shapefile

count_in *count number of neighbours locally*

Description

count number of neighbours locally
More user-friendly but less-optimized version of count_in

Usage

```
count_in(inraster)

count_in2(inraster)
```

Arguments

inraster

Value

a vector of length 1

CRAST_fun *Transform Polygon Shapefile to Raster*

Description

Transform Polygon Shapefile to Raster

Usage

```
CRAST_fun(SHP, field_name, Base, GDAL = TRUE, cropmask = TRUE)
```

Arguments

SHP	spatialPolygonDF object
field_name	which variable to turn into raster?
Base	project SHP to this raster
GDAL	which method
cropmask	mask the raster afterwards

Value

raster

create_counties *Wrapper Functions for County Shapefiles*

Description

Wrapper Functions for County Shapefiles

Usage

```
create_counties(
  hyears = c(1980, 1990, 2000),
  mmyears = 2008:2010,
  myears = 2011:2015,
  ddir,
  rdir
)
```

Arguments

hyears, myears, mmyears	vectors of years
ddir, rdir	directory of shapefiles

Value

list of county shapefiles

DF2Raster *Formatting dataframe as rasterstack*

Description

Formatting dataframe as rasterstack

Usage

```
DF2Raster(DF)
```

Arguments

DF	dataframe

Value

rasterstack

See Also

DF2stack

df2raster *rdname DF2Raster*

Description

rdname DF2Raster

Usage

```
df2raster(DF)
```

DF2stack *Formatting dataframe as rasterstack*

Description

Formatting dataframe as rasterstack

Usage

```
DF2stack(DF, dfname)
```

Arguments

DF	dataframe
dfname	names to keep

Value

rasterstack

See Also

DF2Raster

`df2stack` *rdname DF2stack*

Description

rdname DF2stack

Usage

```
df2stack(DF, dfname)
```

`download.raster` *Download Rasters*

Description

Download Rasters

Usage

```
download.raster(shape_url, layer, outdir = getwd(), layer_new = layer)
```

Arguments

`shape_url`	the directory containing the shape files (.shp, .shx, ...)
`layer`	the name of file to download
`outdir`	the directory where to save the files
`layer_new`	the filename to save

download.shapefile *Download Shapefiles*

Description

Download Shapefiles

Usage

```
download.shapefile(shape_url, layer, outdir = getwd(), layer_new = layer)
```

Arguments

shape_url	the directory containing the shape files (.shp, .shx, ...)
layer	the name of file to download
outdir	the directory where to save the files
layer_new	the filename to save

References

jw hollister, Oct 10, 2012

dyad.maker0 *Make Skeleton for Dyadic Panel*

Description

Make Skeleton for Dyadic Panel

Usage

```
dyad.maker0(times, dyad_name = c("Dyad", "Year"))
```

Arguments

times	time period
dyad_name	

Value

An empty list to be filled in dyad.maker1

dyad.maker1 *Make A list of dyads*

Description

Make A list of dyads

Usage

```
dyad.maker1(
  dyad,
  d_times,
  t_span,
  d_df,
  d_df_id,
  d_df_names,
  d_df2,
  d_df2_names,
  d_df2_aggnames1,
  d_df2_aggnames2,
  d_tab = NULL,
  d_tabx,
  d_taby
)
```

Arguments

dyad	, empty list from DYADmaker0 to be filled
d_times	, time periods to create dyads
t_span	number of periods after d_times to include for data-grouping
d_df	, X variables to merge ()
d_df_id	, X merger ID
d_df_names	, X var name of ID
d_df2	<- DT
d_df2_names	<- "Start"
d_tab	, Y variable to merge (table of battles)
d_tabx	, "BTABx"
d_taby	, "BTABy"
d_df_aggnames1	, X merger name
d_df_aggnames2	<- X merger name

Value

A list

DYADmaker0 *Make Skeleton for Dyadic Panel*

Description

Make Skeleton for Dyadic Panel

Usage

```
DYADmaker0(times, dyad_name = c("Political", "Dyad", "Year"))
```

Arguments

times	time periods
dyad_name	

Value

An empty list to be filled in DYADmaker1

DYADmaker1 *Make A list of dyads*

Description

Make A list of dyads

Usage

```
DYADmaker1(
  dyad,
  d_times,
  t_span,
  d_df,
  d_df_var,
  d_df_id,
  d_df_names,
  d_df_aggnames1,
  d_df_aggnames2,
  d_df2,
  d_df2_names,
  d_tab = NULL,
  d_tabx,
  d_taby
)
```

Arguments

dyad	, empty list from DYADmaker0 to be filled
d_times	, time periods to create dyads
t_span	number of periods after d_times to include for data-grouping
d_df	, X variables to merge (POLIS2)
d_df_var	, X variable names
d_df_id	, X merger ID
d_df_names	, X var name of ID
d_df_aggnames1	, X merger name
d_df_aggnames2	<- X merger name
d_df2	<- DT
d_df2_names	<- "Start"
d_tab	, Y variable to merge (table of battles)
d_tabx	, "BTABx"
d_taby	, "BTABy"

Value

A list

DYADmaker2 *Dyad List Formatting*

Description

Dyad List Formatting

Usage

```
DYADmaker2(dfname, dyad, ...)
```

Arguments

dyad	, DYAD
dfname=c("Polis1",	"Polis2", "Battles", "Pol1.Politic", "Pol2.Politic", "Year")

Details

Transforming List into Dyadic DF (Part 3)

Value

A dataframe with Battle and Political Data

`ExtractClosest` *Spatial Points/Polygon Extract Closest from Raster*

Description

Spatial Points/Polygon Extract Closest from Raster

Wrapper for Extract Closest

Usage

```
ExtractClosest(rast, spdf, ncore = 24, setvals = FALSE, returnvec = TRUE)

extract_closest(rast, spdf, ncore = 24, setvals = FALSE)
```

Arguments

`rast`	A raster
`spdf`	A SpatialPoints, SpatialPointsDataFrame, Matrix or Dataframe of coordinates
`ncore`	the size of the window used in the neighbourhood calculations
`setvals`	set raster values to extract
`returnvec`	return a list (defaults to vector)

Details

Extract Closest non-NA Raster Values to Spatial Points in parallel. Use returnlist to return a list when the nearest raster locations are not unique

library(raster) xy <- cbind(x=seq(-1,2,by=.1), y=seq(2,-1,by=-.1)) spdf <- sp::SpatialPoints(xy) rast <- raster::raster(matrix(runif(100), 10, 10)) raster::crs(spdf) <- raster::crs(rast) <- "+proj=moll +lon_0=0 +x_0=0 +y_0=0 +ellps=WGS84 +datum=WGS84 +units=m +no_defs" ExtractClosest(rast, spdf, 1)

Value

A list with raster values for each spatial point

`getSmallPolys` *Trim Polygon of Small Areas*

Description

Trim Polygon of Small Areas

Usage

```
getSmallPolys(poly, minarea = 0.01)
```

Arguments

poly	shapefile
minarea	only get polygons > minarea

Value

shapefile

HHI *Calculate HHI for each raster cell*

Description

Calculate HHI for each raster cell

Usage

```
HHI(
  rast,
  wind,
  FUN = hhi_in,
  mask = NA,
  writedir = NA,
  HHIname = paste0("HHIrast_", wind[1], "_", wind[2])
)
```

Arguments

wind	size of local windows to consider
FUN	what to calculate
mask	mask the values afterwards
writedir	write the raster to hard disk
HHIname	what to name the raster
Crast	raster from which to perform calculations

Value

a raster

hhi_in *calculate HHI locally*

Description

calculate HHI locally

More user-friendly but less-optimized version of hhi_in

Usage

```
hhi_in(inraster)

hhi_in2(inraster)
```

Arguments

```
inraster
```

Value

a vector of length 1

Examples

```
rast <- raster::raster( matrix(runif(100), 10, 10) )
inrast <- as.integer(rast> .2)
hhi_rast <- raster::focal(inrast, w=matrix(1,3,3), hhi_in2)
```

layer_list *Transform RasterStack to Data.Table using Parallel Processing*

Description

Transform RasterStack to Data.Table using Parallel Processing

Usage

```
layer_list(stack)
```

Arguments

stack	stack of rasters to be converted, must have coordinate columns (x,y)

Value

datatable

quickdf *Formatting at dataframe*

Description

Formatting at dataframe

Usage

```
quickdf(l)
```

Arguments

l	list

Value

rasterstack

Raster2DF *rdname stack2df*

Description

rdname stack2df

Usage

```
Raster2DF(Rstack)
```

stack2df *Transform RasterStack into DataFrame*

Description

Transform RasterStack into DataFrame

Usage

```
stack2df(Rstack)
```

Arguments

Rstack	stack of rasters

Value

data.frame

TRI *Calculate Terrain Ruggedness with Padding*

Description

Calculate Terrain Ruggedness with Padding

Usage

```
TRI(E1, nr = 3, nc = nr)
```

Arguments

E1	A raster measuring elevation
nr	the number of rows in the window used in the neighbourhood calculations
nc	the number of columns in the window used in the neighbourhood calculations

Value

A raster with TRI values

Examples

```
TRI( raster::raster(matrix( runif(9), 3,3) ) )
```

Index