Package ‘binaryTimeSeries’

October 12, 2022

Type  Package
Title  Analyzes a Binary Variable During a Time Series
Version  1.0.2
License  AGPL (>= 3)
Depends  R (>= 3.3.0), rgdal
Imports  dplyr, ggnewscale, ggplot2, graphics, grDevices, magrittr, mice, prettymappr, raster, reshape2, stats, terra, utils
Suggests  knitr, maptools, rmarkdown, spelling
VignetteBuilder  knitr
Encoding  UTF-8
RoxygenNote  7.1.2
Config/testthat/edition  3
Language  en-US
NeedsCompilation  no
Author  Thomas Mumuni Bilintoh [aut, cre]  (https://orcid.org/0000-0002-6852-4439), Robert Gilmore Pontius Jr [aut]  (https://orcid.org/0000-0001-7287-5875)
Maintainer  Thomas Mumuni Bilintoh <tbilintoh@clarku.edu>
changeData

Description

Creates a list containing the raster data sets concerning the number of times the category of interest is present and the number of times the category of interest changes during the time series.

Usage

changeData(x, nodata, category, spres, datacrs = NULL)

Arguments

- **x**: is the data, which must be a RasterStack, RasterBrick, SpatRaster or data frame.
- **nodata**: is alphanumeric, which denotes no data in the data set.
- **category**: is the category of interest. The default is set to 1.
- **spres**: is a 1*2 vector indicating the spatial resolution of the data. The default is set to c(1000,1000).
- **datacrs**: is the Coordinate Reference System (CRS) of the input data.

Value

The output from changeData
**dataStack**

Creates results that serve as input for the stackbarPlot function.

**Description**

Creates results that serve as input for the stackbarPlot function.

**Usage**

```r
dataStack(
  x, 
  nodata = 2, 
  category = 1, 
  unified = "yes", 
  categoryName = "marsh", 
  regionName = "Study Region"
)
```

**Arguments**

- **x** is the data, which must be a RasterStack, RasterBrick, SpatRaster or data frame.
- **nodata** is alphanumeric, which denotes no data in the data set.
- **category** is the category of interest. The default is set to 1.
- **unified** is a string, which can be "yes" or "no" only. If "yes," the change is a percentage of a region’s unified area; else, the change is a percentage of the entire region under consideration.
- **timePoints** is a vector containing the time points under consideration. The default is c(2000, 2001, 2002, 2003, 2005).
- **categoryName** is a character representing the name of the category of interest. Default is "category".
- **regionName** is a string or character the name of the study region.

**Value**

The output from `dataStack`
Examples

```r
example_data <- terra::rast(system.file("external/Example_raster_Y.tif", package="binaryTimeSeries"))
no_data <- 2
cat_interest <- 1
unified.resp <- "yes"
categ_name <- "Category"
region_name <- "Study Region"
datstk_output <- dataStack(x = example_data, nodata = no_data, category = cat_interest,
unified = unified.resp, timePoints = time_points, categoryName = categ_name,
regionName = region_name)
```

```
presencePlot

Creates two maps: a map showing how many times the category is present during the time series and a map showing how many times the category changes during the time series.
```

Description

Creates two maps: a map showing how many times the category is present during the time series and a map showing how many times the category changes during the time series.

Usage

```r
presencePlot(
  input,
  pltunit = "m",
  dataEpsg = 32632,
  scalePos = "bottomleft",
  narrowPos = "topright",
  narrowSize = 1,
  categoryName = "marsh",
  xAxis = "Longitude (m)",
  yAxis = "Latitude (m)",
  axisText = 1.2,
  axisLabel = 1.2,
  plotTitle = 1.5
)
```

Arguments

- `input` is the results from running the "changeData" function.
- `pltunit` is the unit which the current map is plotted in, one of cm, m, km, in, ft, mi. or lat or lon. This parameter is optional if "dataEpsg" is passed.
- `dataEpsg` is the projection of the current map. If extents are valid lat or lons, the projection is assumed to be lat and lon (EPSG:4326), or Spherical Mercator otherwise (EPSG:3857). This is done to work seamlessly with "OpenStreetMap" packages. Currently it is set to 32632.
Create stack bar plots showing trajectories sizes and the three change components.

Description

Create stack bar plots showing trajectories sizes and the three change components.
Usage

```
stackbarPlots(
  input,
  axisSize = 12,
  lbAxSize = 15,
  lgSize = 12,
  titleSize = 15,
  datbreaks = "no",
  upperlym = 35,
  lowerlym = -50,
  lymby = 5,
  upperlym2 = 0.5,
  lymby2 = 0.1,
  xAngle = 0
)
```

Arguments

- **input**: is the results from running the "dataStack" function.
- **axisSize**: is a numerical value that control the size of the labels on tick marks of the horizontal and vertical tick marks.
- **lbAxSize**: is a numerical value to control the size of the labels on the horizontal and vertical axis.
- **lgSize**: is a numerical value to control the size of the legend text.
- **titleSize**: is a numerical value to control the size of the title text.
- **datbreaks**: is a string of "yes" or "no", which controls the range and sub-division of the vertical axis of the stacked bar plots. The default is "no", which automatically generates the range and interval of the vertical axis. If "no" the user need to manual input values for "upperlym", "lowerlym", "lymby", "upperlym2", and "lymby2".
- **upperlym**: if datbreaks set to "yes," is a numerical value to control the upper limit of the trajectory stack bar plot.
- **lowerlym**: if datbreaks set to "yes," is a numerical value to control the lower limit of the trajectory stack bar plot.
- **lymby**: if datbreaks set to "yes," is a numerical value to control interval on the vertical axis of the components of change stack bar plot.
- **upperlym2**: if datbreaks set to "yes," is a numerical value to control the upper limit of the components of change stacked bar plot.
- **lymby2**: if datbreaks set to "yes," is a numerical value to control the interval on the vertical axis of the components of change stacked bar plot.
- **xAngle**: is a numerical value to control the orientation of the text on the vertical axis of the trajectory stack bar plot.

Value

The output from `stackbarPlots`
Examples

```r
example_data <- terra::rast(system.file("external/Example_raster_Y.tif",package="binaryTimeSeries"))
no_data <- 2
cat_interest <- 1
unified_resp <- "yes"
categ_name <- "Category"
region_name <- "Study Region"
datstk_output <- dataStack(x = example_data,nodata = no_data,category = cat_interest,
unified = unified_resp,timePoints = time_points,categName = categ_name,
regionName = region_name)
stkbarplt_output <- stackbarPlots(input = datstk_output,axisSize = 12,
lbAxSize = 15,lgSize = 12,titleSize = 15,upperlym = 35,lowerlym = -50,
lymby = 5,upperlym2 = 0.5,lymby2 = 0.1,xAngle = 0)
```

trajData

creates the data which serves as input for the "trajPlot" function.

Description

creates the data which serves as input for the "trajPlot" function.

Usage

```r
trajData(
  x,
  nodata = 2,
  category = 1,
  spres = c(1000,1000),
  datacrs = NULL,
  unified = "yes"
)
```

Arguments

- **x** is the data, which must be a RasterStack,RasterBrick,SpatRaster or data frame.
- **nodata** is alphanumeric, which denotes no data in the data set.
- **category** is the category of interest. The default is set to 1.
- **spres** is a 1*2 vector indicating the spatial resolution of the data. The default is c(1000,1000).
- **datacrs** is the CRS of the input data.
- **unified** is a string, which can be "yes" or "no" only. If "yes," the change is a percentage of a region’s unified area; else, the change is a percentage of the entire region under consideration.
The output from `trajData`

### Examples

```r
e.example_data <- terra::rast(system.file("external/Example_raster_Y.tif", package="binaryTimeSeries"))
no_data <- 2
cat_interest <- 1
data_res <- c(1000,1000)
data_prj <- "+proj=utm +zone=32 +datum=WGS84 +ellps=GRS80 +units=m +no_defs"
unified_resp <- "yes"
trajdt_output <- trajData(x = example_data, nodata = no_data,
category = cat_interest, spres = data_res, datacrs = data_prj,
unified = unified_resp)
```

---

### Description

Creates a map and pie chart of the trajectories of the category of interest.

### Usage

```r
trajPlot(
  input,
  pltunit = "m",
dataEpsg = 32632,
categoryName = "marsh's",
scalePos = "bottomleft",
narrowPos = "topright",
narrowSize = 1,
xAxis = "Longitude (m)",
yAxis = "Latitude (m)",
axisText = 1.2,
axisLabel = 1.4,
plotTitle = 1.5,
legendTex = 0.9
)
```

### Arguments

- **input**: is the results from running the "trajData" function.
- **pltunit**: is the unit in which the current map is plotted, one of cm, m, km, in, ft, mi. or lat or lon.
dataEpsg is the projection of the current map. If extents are valid lat and lons, the projection is assumed to be lat and lon (EPSG:4326), or Spherical Mercator otherwise.
categoryName is a character representing the name of the category of interest. Default is "marsh".
scalePos is a character indicating label for the horizontal axis. Default is "Longitude (m)".
yAxis is a character indicating label for the vertical axis. Default is "Latitude (m)".
axisText is a numeric value controlling the size of the text on the horizontal and vertical ticks.
axisLabel is a numeric value controlling the size of the horizontal and vertical labels
plotTitle is a numeric value controlling the size of the plot title.
legendTex is a numerical value controlling the size of the legend text.

Value
The output from trajPlot

Examples

e`example_data <- terra::rast(system.file("external/Example_raster_Y.tif",package="binaryTimeSeries"))
no_data <- 2
cat_interest <- 1
data_res <- c(1000,1000)
data_prj <- "+proj=utm +zone=32 +datum=WGS84 +ellps=GRS80 +units=m +no_defs"
unified_resp <- "yes"
trajdt_output <- trajData(x = example_data,nodata = no_data,
category = cat_interest,spres = data_res,datacrs = data_prj,unified = unified_resp)
trjplt_output <- trajPlot(input = trajdt_output)"
Index

changeData, 2, 2
dataStack, 3, 3
presencePlot, 4, 5
stackbarPlots, 5, 6
trajData, 7, 8
trajPlot, 8, 9