Package 'rtsVis'

October 14, 2022

Type Package

Title Raster Time Series Visualization

Version 0.0.3

Depends R (>= 3.5.0)

Date 2021-05-26

Description A lightweight 'R' package to visualize large raster time series, building on a fast temporal interpolation core.

License GPL-3

Encoding UTF-8

Imports ggplot2, RStoolbox, moveVis, raster, stats, dplyr, pbapply, sp, sf, tidyr, forcats, assertthat, magrittr

BugReports https://github.com/JohMast/rtsVis/issues

RoxygenNote 7.1.1

Suggests spelling

LazyData true

Language en-US

NeedsCompilation no

Author Johannes Mast [aut, cre] (<https://orcid.org/0000-0001-6595-5834>), Jakob Schwalb-Willmann [aut] (<https://orcid.org/0000-0003-2665-1509>)

Maintainer Johannes Mast <johannes.mast@stud-mail.uni-wuerzburg.de>

Repository CRAN

Date/Publication 2021-05-26 17:00:02 UTC

R topics documented:

`MODIS_SI_ds` *Example MODIS time series for southern Slovenia.*

Description

105 MODIS MCD43A4 Images covering the time from 2020-02-15 to 2020-05-29 , downsampled by a factor of 4 and subset to bands 1 to 4 to serve as lightweight example time series. Acquired from LP DAAC using the getSpatialData package.

Usage

```
MODIS_SI_ds
```

Format

A list of 105 raster objects

Source

https://lpdaac.usgs.gov/tools/usgs-earthexplorer/

`SI_positions` *Example vector data for Slovenia.*

Description

Three sets of example vector data for Slovenia. Two sp objects and one matrix object. Chosen to match the "MODIS_SI_ds" raster data.

Usage

```
SI_positions
```

Format

A list of three sets of example vector data:

points_matrix A matrix of coordinates corresponding to the coordinates of two points in Slovenia. Manually created. MODIS sinusoidal projection.

points SpatialPointsDataFrame of four points in Slovenian municipalities. Manually created. MODIS sinusoidal projection.

polygons SpatialPolygonsDataFrame of three polygons corresponding to borders of municipalities. Acquired from GADM. MODIS sinusoidal projection. ...

Source

https://gadm.org/data.html

`ts_add_positions_to_frames` *Add points, coordinates, or polygons to a list of spatial plots*

Description

Add points, coordinates, or polygons to a list of spatial plots

Usage

```
ts_add_positions_to_frames(
  r_frame_list,
  positions,
  position_names = NULL,
  pcol = "red",
  tcol = "red",
  psize = 2,
  tsize = 7,
  ttype = "text",
  t_hjust = 0,
  t_vjust = 0,
  position_legend_title = "Position",
  legend_position = "right",
  aes_by_pos = FALSE,
  col_by_pos = FALSE,
  add_text = FALSE
)
```

Arguments

`r_frame_list`	list of ggplots, as generated by `ts_makeframes`.
`positions`	object containing the coordinates. One of • A two-column `matrix` of coordinates where the first column corresponds to the longitude and the second column corresponds to the latitude. • A `SpatialPolygonsDataFrame` from which coordinates can be extracted. • A `SpatialPointsDataFrame` from which coordinates can be extracted. • An sf object containing `POINTS`,`POLYGONS` or `MULTIPOLYGONS`
`position_names`	(Optional) character, names of the positions to be added in legend or text (If `add_text`). By default, will create placeholder names by combining the object type and Id (Example: *"Polygon 3"*)
`pcol`	(Optional) character, color of the spatial objects. Default is `"red"`.
`tcol`	(Optional) character, if `add_text`: The color of the text. Default is `"red"`.

psize (Optional) numeric, plot size of the spatial objects. Default is 2.

tsize (Optional) numeric, if add_text: The size of the text. Default is 7.

ttype (Optional) character, if add_text: The type of the text. Either "label" or "text". Default is "text".

t_hjust (Optional) numeric, if add_text: Horizontal offset of the text in map units. Default is 0.

t_vjust (Optional) numeric, if add_text: Vertical offset of the text in map units. Default is 0.

position_legend_title
(Optional) character, title of the legend. Default is "Position".

legend_position
(Optional) character, position of the legend. Use "none" to disable the legend. Default is "right"

aes_by_pos (Optional) logical. If TRUE: vary some aesthetic (linetype for polygons, shape for points) to be different for each position? If FALSE, this also disables the legend, as no notable classes will be plotted. Default is FALSE.

col_by_pos (Optional) logical. If TRUE: vary the color to be different for each position? If TRUE, overrides aes_by_pos and pcol. Default is FALSE.

add_text (Optional) logical. If TRUE: add a text to each position using add_text. Default is "False".

Details

The function takes a positions object, which can be a spatial object or a matrix of coordinates, and adds them to each of the elements of r_frame_list. Optionally it also adds text at their respective positions usingadd_text.

- ts_add_positions_to_frames is intended to be an easy way to add multiple objects to the spatial frames at fixed positions. For adding individual positions or text, potentially at varying positions, it is recommended to all add_gg and add_text directly.

Value

A list of ggplots with added positions.

Author(s)

Johannes Mast

See Also

add_text add_gg

Examples

```
#Setup
 library(rtsVis)
 library(ggplot2)
 # Load example dataset at a greatly increased interval
x_list <- MODIS_SI_ds[seq(1,length(MODIS_SI_ds),30)]
x_dates <- do.call(c, lapply(MODIS_SI_ds,attr,"time") )[seq(1,length(MODIS_SI_ds),30)]

#Fill NAs
x_list_filled <- ts_fill_na(x_list)

#Make a sequence of output dates, double the length of input dates
out_dates <-seq.POSIXt(from = x_dates[1],
                       to = x_dates[length(x_dates)],length.out = length(x_dates)*2 )

#For each output date, interpolate a raster image from the input files
r_list_out <- ts_raster(r_list = x_list_filled,
                        r_times = x_dates,
                        out_times = out_dates,
                        fade_raster = TRUE)
#Create the frames
# as from the desired layers
r_frames <- ts_makeframes(x_list = r_list_out,samplesize = 10,
                          l_indices = c(1,4,3))

#optional: Use moveVis functions to add additional elements to our frames
#library(magrittr)
# r_frames <- r_frames %>%
#   moveVis::add_labels(x = "Longitude", y = "Latitude")%>%
#   moveVis::add_northarrow(colour = "white", position = "bottomright") %>%
#   moveVis::add_timestamps(type = "label") %>%
#   moveVis::add_progress()


#### Add the polygons
# Add polygons to the frames
polygons <- SI_positions$polygons #Polygons of Slovenian municipalities covered by the raster
r_frames_style_poly <-
  ts_add_positions_to_frames(
    r_frame_list = r_frames,
    positions = polygons,
    psize = 1,
    pcol = "red",
    position_names = c("Radece", "Ljubljana", "Kocevje"),
    position_legend_title = "Obcina",
    legend_position = "left",
    aes_by_pos = FALSE
  )
#Look at one of the new frames
r_frames_style_poly[5]

#Alternatively add points
```

```
points <- SI_positions$points #Points in Slovenia
r_frames_style_point <- rtsVis::ts_add_positions_to_frames(r_frame_list = r_frames,
                                                           positions = points,
                                                           psize = 4,
                                                           pcol = "orange",
                                                          position_names = c("Ljubljana",
                                                                       "Ivancna Gorica",
                                                                     "Dolenjske Toplice",
                                                                          "Loski Potok"),
                                                        position_legend_title = "Obcina",
                                                           legend_position = "right",
                                                           aes_by_pos = TRUE,
                                                           add_text = TRUE,
                                                           ttype = "label",
                                                           tsize = 3,
                                                           t_hjust = -3000,
                                                           t_vjust = 1000)
#Look at one of the new frames
# r_frames_style_point[5]


#Alternatively add points
# points_mat <- SI_positions$points_matrix #Points in Slovenia
# r_frames_style_point_mat <- ts_add_positions_to_frames(r_frame_list = r_frames,
#                                                        positions = points_mat,
#                                                        psize = 4,
#                                                        pcol = "orange",
#                                                        position_names = c("A",
#                                                                           "B" ),
#                                                       position_legend_title = "Point",
#                                                        legend_position = "right",
#                                                        aes_by_pos = TRUE,
#                                                        add_text = TRUE,
#                                                        ttype = "label",
#                                                        tsize = 3,
#                                                        t_hjust = -3000,
#                                                        t_vjust = 1000)
#Look at one of the new frames
# r_frames_style_point_mat[5]
```

`ts_copy_frametimes` *Sets frametimes for a list of frames*

Description

Sets frametimes for a list of frames

Usage

```
ts_copy_frametimes(x, y)
```

Arguments

x	A list to which timestamps are added.
y	A list from which the timestamps are derived.

Value

A list identical to x with timestamps carried over from y.

---

ts_fill_na *Fill NA values in a raster time series*

Description

Fill NA values in a raster time series

Usage

```
ts_fill_na(x_list_fill, maskvalues = NULL, verbose = FALSE, ...)
```

Arguments

x_list_fill	a list of raster objects.
maskvalues	numeric, a vector of values to be set to NA before the masking.
verbose	(Optional) logical. If TRUE outputs progress. Default is FALSE.
...	additional arguments to be passed on to approxNA. Of particular interest is the rule argument which defines how first and last cells are dealt with.

Details

Loads all layers of a specific bands into a stack and uses approxNA to fill the NAs if possible. Note that the procedure requires the entire list of raster layery for each band to be be stacked. It is therefore very memory intensive and likely to fail for very large time series.

Value

A list of rasters with NAs filled.

Author(s)

Johannes Mast

Examples

```
#Setup
 library(rtsVis)
x_list <- MODIS_SI_ds[seq(1,length(MODIS_SI_ds),15)]   #A list of raster objects

#Fill NAs
x_list_filled <- ts_fill_na(x_list)
```

`ts_flow_frames` *Create a series of charts of a raster time series*

Description

Create a series of charts of a raster time series

Usage

```
ts_flow_frames(
  r_list,
  positions = NULL,
  position_names = NULL,
  band_names = NULL,
  band_colors = NULL,
  val_min = NULL,
  val_max = NULL,
  val_by = NULL,
  plot_size = 1,
  position_legend = NULL,
  legend_position = "right",
  band_legend = NULL,
  band_legend_title = NULL,
  position_legend_title = NULL,
  pbuffer = NULL,
  plot_function = "line",
  aes_by_pos = TRUE,
  FUN = NULL,
  return_df = FALSE,
  ...
)
```

Arguments

`r_list`	list of rasters, as generated by `ts_raster`.
`positions`	(Optional) object containing the coordinates. One of

- A two-column matrix of coordinates where the first column corresponds to the longitude and the second column corresponds to the latitude.
- A SpatialPolygonsDataFrame
- A SpatialPointsDataFrame
- An sf object containing POINTS,POLYGONS or MULTIPOLYGONS

If no positions are specified, one position is assumed to be the entire extent of the rasters.

position_names (Optional) character, names of the positions to be added in legend or text. By default, will create placeholder names by combining the object type and Id (Example: "Polygon 3")

band_names (Optional) character, names of the bands to be added in legend, if band_legend. By default, will create placeholder names by band index (Example: "Band 3").

band_colors (Optional) character, colors which represent the bands in the plot. Must be in matching order to band_names. If r_list is discrete, the colors will be mapped to the different levels. By default will use ggplot colors.

val_min (Optional) numeric, minimum value of the y-axis. By default chooses a rounded minimum value of the rasters contained within r_list.

val_max (Optional) numeric, maximum value of the y-axis. By default chooses a rounded maximum value of the rasters contained within r_list.

val_by (Optional) numeric, interval value of the y-axis. Default is 0.1.

plot_size (Optional) numeric, size for the ggplot objects. Default is 1.

position_legend
(Optional) logical. If TRUE: Add a legend for the positions. Only recommended if aes_by_pos is also TRUE.

legend_position
(Optional) character, position of the legend. Use "none" to disable all legends. Default is "right".

band_legend (Optional) logical. If TRUE: Add a legend for the bands. Default is TRUE.

band_legend_title
(Optional) character, title of the band legend. Default is "Bands".

position_legend_title
(Optional) character, title of the band legend. Default is "Positions".

pbuffer (Optional) numeric. The radius of a buffer around each object which will be applied before extraction. By default, no buffer is used.

plot_function (Optional) character or function, type of the plots to produce. Currently supported are 'line', 'line2', 'vio', 'dens', 'dens2', 'bar_stack', 'bar_fill', 'pie'.One of

- 'line' A line chart, suited for comparing trends between bands
- 'line2' A line chart, suited for comparing trends between positions
- 'vio' A density chart, suited for comparing of distributions across bands and positions
- 'dens' A density chart, suited for comparing of distributions across positions

- 'dens2' A density chart, suited for comparing of distributions across bands
- 'bar_stack' A horizontal bar chart, suited for visualizing counts and proportions among discrete data.
- 'bar_fill' A horizontal bar chart, suited for visualizing proportions among discrete data.
- 'pie' A pie chart, suited for visualizing rough proportions among discrete data with few categories.

Alternatively, a custom function with similar structure and arguments can be passed to create other types of plots. Default is "line".

aes_by_pos (Optional) logical. If TRUE: vary the linetype aesthetic to be different for each position? If FALSE, this also disables the position_legend, as no notable classes will be plotted. Ignored by some plot types which inherently map position to facets. Default is TRUE.

FUN (Optional) function to summarize the values (e.g. mean) during the extraction. See extract for more details. Default is "NULL". Summarizing in this way is not sensible for many plot types which visualize distribution or count. Note that usually, summarize statistics will be calculated in an appropriate way by the plot_function rather than during the extraction.

return_df (Optional) logical. Return a dataframe with the extracted values instead of a plot? This can be useful for experimenting with plot creation. Default is FALSE.

... (Optional) additional arguments for plot_function.

Details

Values are extracted using extract and plotted on a ggplot. The type of the ggplot is specified by plot_function. Currently supported are "line" and "violin" as well as custom functions which accept similar inputs. The function may fail for large polygons and long time series. Be aware that if ts_raster is used with fade, interpolation may be used to generate raster values.

Value

A list of ggplots, one for each element of r_list.

Author(s)

Johannes Mast

See Also

ts_raster

Examples

```
#' #Setup
library(rtsVis)
# Load example dataset at a greatly increased interval
x_list <- MODIS_SI_ds[seq(1,length(MODIS_SI_ds),30)]
x_dates <- do.call(c, lapply(MODIS_SI_ds,attr,"time") )[seq(1,length(MODIS_SI_ds),30)]
```

```
#Fill NAs
x_list_filled <- ts_fill_na(x_list)

#Make a sequence of output dates, double the length of input dates
out_dates <-seq.POSIXt(from = x_dates[1],
                       to = x_dates[length(x_dates)],length.out = length(x_dates)*2 )

#For each output date, interpolate a raster image from the input files
r_list_out <- ts_raster(r_list = x_list_filled,
                        r_times = x_dates,
                        out_times = out_dates,
                        fade_raster = TRUE)
#Create the frames
# as from the desired layers
r_frames <- ts_makeframes(x_list = r_list_out,samplesize = 10,
                          l_indices = c(1,4,3))

# Create a line plot from the data extracted over points
points <- SI_positions$points #Polygons of Slovenian municipalities covered by the raster
flow_frames_point_line <- rtsVis::ts_flow_frames(r_list = r_list_out,
 position_names = c("Ljubljana","Ivancna Gorica","Dolenjske Toplice","Loski Potok"),
 band_names = c("620 - 670","841 - 876","459 - 479","545 - 565"),
 positions = points,
 band_colors = c("firebrick3","darkorchid3","dodgerblue3","olivedrab3"),
           band_legend_title = "Wavelength [nm]",
           position_legend_title = "Obcina",
           legend_position = "bottom",
           position_legend = FALSE,
           band_legend=TRUE,aes_by_pos = TRUE)

#Check one of the frames
flow_frames_point_line[[5]]


# Create a violin plot from the data extracted over polygons
# polygons <- SI_positions$polygons
#flow_frames_poly_vio <-
#rtsVis::ts_flow_frames(r_list = r_list_out,
#           position_names = c("Radece","Ljubljana","Kocevje"),
#           band_names = c("620 - 670","841 - 876","459 - 479","545 - 565"),
#           positions = polygons,
#           band_colors = c("firebrick3","darkorchid3","dodgerblue3","olivedrab3"),
#           band_legend_title = "Wavelength [nm]",
#           position_legend_title = "Obcina",
#           position_legend = FALSE,
#          legend_position = "left",
#           band_legend=TRUE,aes_by_pos = FALSE,
#          plot_function = "vio")
#Check one of the frames
# flow_frames_poly_vio[[5]]
```

`ts_makeframes` *Create spatial ggplots of a raster time series*

Description

Create spatial ggplots of a raster time series

Usage

```
ts_makeframes(
  x_list,
  r_type = NULL,
  minq = 0.02,
  maxq = 0.98,
  samplesize = 1000,
  blacken_NA = FALSE,
  l_indices = NULL,
  alpha = NULL,
  hillshade = NULL,
  ...
)
```

Arguments

`x_list`	a list of raster objects.
`r_type`	(Optional) character, one of `"discrete"`, `"gradient"` or `"RGB"`. By default, attempts to discern the `r_type` from the content of `r_list`.
`minq`	(Optional) numeric, lower quantile boundary for the stretch. Default is `0.02`.
`maxq`	(Optional) numeric, upper quantile boundary for the stretch. Default is `0.98`.
`samplesize`	(Optional) numeric, number of samples per layer to determine the quantile from. See sampleRegular for details on the sampling. Default is `1000`.
`blacken_NA`	(Optional) logical. If `TRUE`: set `NA` to `0`. Default is `FALSE`.
`l_indices`	(Optional) numeric, a vector of layer indices specifying which layers are to be plotted. Should contain 3 values for an RGB image or a single value for a discrete or gradient image. By default, chooses the first three layers if `r_type` is `"RGB"` and the first layer if `r_type` is `"discrete"` or `"gradient"`.
`alpha`	(Optional) numeric. The opacity of the plot. Default is 1.
`hillshade`	(Optional) A raster layer. If one is provided, it will be used as the base layer and plotted with ggR while the raster layers from x_list will be plotted over it with a default `alpha` of 0.5. By default, no hillshade is used.
`...`	(Optional) further arguments for the plotting.

Details

A linear percent stretch will be applied to each band to improve contrast. #

$$(X - Low_{in}) * \frac{((High_{out} - Low_{out})}{(High_{in} - Low_{in}))} + Low_{out}$$

The stretch parameters # $High_{out}$,Low_{out},$High_{in}$,Low_{in} # are calculated separately for each band based on the `minq` and `maxq` which are first applied to `samplesize` regular samples (see sampleRegular) of each individual layer. From these, across all layers belonging to a certain band, the minimum and maximum values are taken as the stretching parameters for the linear stretch, which is performed using rescaleImage. Discrete `r_types` will not be stretched. To further enhance the plots, consider using functionalities implemented in **moveVis**, (see `http://movevis.org/`). For example, a northarrow may be added to all frames using add_northarrow.

Value

A list of ggplots

Author(s)

Johannes Mast

Examples

```
#Setup
# Load example dataset at a greatly increased interval
x_list <- MODIS_SI_ds[seq(1,length(MODIS_SI_ds),25)]
x_dates <- do.call(c, lapply(MODIS_SI_ds,attr,"time") )[seq(1,length(MODIS_SI_ds),25)]

#Fill NAs
x_list_filled <- ts_fill_na(x_list)

#Make a sequence of output dates, double the length of input dates
out_dates <-seq.POSIXt(from = x_dates[1],
                       to = x_dates[length(x_dates)],length.out = length(x_dates)*2 )

#For each output date, interpolate a raster image from the input files
r_list_out <- ts_raster(r_list = x_list_filled,
                        r_times = x_dates,
                        out_times = out_dates,
                        fade_raster = TRUE)
#Create the frames
# as from the desired layers
r_frames <- ts_makeframes(x_list = r_list_out,samplesize = 10,
                          l_indices = c(1,4,3))
```

ts_raster *Assemble/interpolate a raster time series*

Description

This function assembles a raster time series by assigning or interpolating input rasters to a target time series.

Usage

```
ts_raster(
  r_list,
  r_times,
  out_times = NA,
  fade_raster = FALSE,
  ...,
  verbose = TRUE
)
```

Arguments

r_list	a list of raster objects.
r_times	POSIXct, a vector of times corresponding to the elements of r_list.
out_times	POSIXct, a vector of times for which output rasters will be created.
fade_raster	(Optional) logical. If TRUE performs a linear interpolation to calculate the values for the output raster. Otherwise uses a nearest temporal neighbor approach. Default is FALSE.
...	additional arguments.
verbose	(Optional) logical. If TRUE outputs progress. Default is TRUE.

Value

a list of raster objects.

Author(s)

Jakob Schwalb-Willmann, Johannes Mast

Examples

```
#Setup

 library(rtsVis)
 # Load example dataset at a greatly increased interval
x_list <- MODIS_SI_ds[seq(1,length(MODIS_SI_ds),30)]
x_dates <- do.call(c, lapply(MODIS_SI_ds,attr,"time") )[seq(1,length(MODIS_SI_ds),30)]
```

```
#Fill NAs
x_list_filled <- ts_fill_na(x_list)

#Make a sequence of output dates, double the length of input dates
out_dates <-seq.POSIXt(from = x_dates[1],
                       to = x_dates[length(x_dates)],length.out = length(x_dates)*2 )

#For each output date, interpolate a raster image from the input files
r_list_out <- ts_raster(r_list = x_list_filled,
                        r_times = x_dates,
                        out_times = out_dates,
                        fade_raster = TRUE)
```

Index