

# Package ‘rtsVis’

February 24, 2021

**Type** Package

**Title** Raster Time Series Visualization

**Version** 0.0.1

**Depends** R (>= 3.5.0)

**Date** 2021-02-23

**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, assertthat

**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](mailto:johannes.mast@stud-mail.uni-wuerzburg.de)>

**Repository** CRAN

**Date/Publication** 2021-02-24 10:00:02 UTC

## R topics documented:

MODIS_SI_ds . . . . .	2
SI_positions . . . . .	2
ts_add_positions_to_frames . . . . .	3
ts_fill_na . . . . .	6

ts_flow_frames . . . . .	7
ts_makeframes . . . . .	10
ts_raster . . . . .	12

<b>Index</b>	<b>14</b>
--------------	-----------

---

MODIS_SI_ds	<i>Example MODIS time series for southern Slovenia.</i>
-------------	---------------------------------------------------------

---

### 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	<i>Example vector data for Slovenia.</i>
--------------	------------------------------------------

---

### 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,
  add_text = FALSE
)
```

**Arguments**

r_frame_list	list of ggplots, as generated by <code>ts_makeframes</code> .
positions	object containing the coordinates. One of <ul style="list-style-type: none"> <li>• A two-column matrix of coordinates where the first column corresponds to the longitude and the second column corresponds to the latitude.</li> <li>• A <code>SpatialPolygonsDataFrame</code> from which <code>coordinates</code> can be extracted.</li> <li>• A <code>SpatialPointsDataFrame</code> from which <code>coordinates</code> can be extracted.</li> <li>• An <code>sf</code> object containing POINTS,POLYGONS or MULTIPOLYGONS</li> </ul>
position_names	(Optional) character, names of the positions to be added in legend or text (If <code>add_text</code> ). By default, will create placeholder names by combining the object type and Id (Example: <code>*"Polygon 3"*</code> )
pcol	(Optional) character, color of the spatial objects. Default is "red".
tcol	(Optional) character, if <code>add_text</code> : 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.
add_text	(Optional) logical. If TRUE: add a text to each position using <a href="#">add_text</a> . 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 using [add\\_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)]
```



```

                                "Loški Potok"),
position_legend_title = "Občina",
  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\_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, verbose = FALSE, ...)
```

## Arguments

x_list_fill	a list of raster objects.
verbose	(Optional) logical. If TRUE outputs progress. Default is FALSE.
...	additional arguments to be passed on to <a href="#">approxNA</a> . 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 layers for each band to 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 = mean,
    return_df = FALSE,
    ...
)

```

## Arguments

r_list	list of rasters, as generated by <a href="#">ts_raster</a> .
positions	(Optional) object containing the coordinates. One of <ul style="list-style-type: none"> <li>• A two-column matrix of coordinates where the first column corresponds to the longitude and the second column corresponds to the latitude.</li> <li>• A <code>SpatialPolygonsDataFrame</code></li> <li>• A <code>SpatialPointsDataFrame</code></li> <li>• An <code>sf</code> object containing POINTS,POLYGONS or MULTIPOLYGONS</li> </ul> <p>If no positions are specified, one position is assumed to be the entire extent of the rasters.</p>
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".



<code>pbuffer</code>	(Optional) numeric. The radius of a buffer around each object which will be applied before extraction. By default, no buffer is used.
<code>plot_function</code>	(Optional) character or function, type of the plots to produce. Currently supported are "line", "line_flp" and "violin". Alternatively, a custom function with similar structure and arguments can be passed to create other types of plots. Default is "line".
<code>aes_by_pos</code>	(Optional) logical. If TRUE: vary the linetype aesthetic to be different for each position? If FALSE, this also disables the <code>position_legend</code> , as no notable classes will be plotted. Default is TRUE.
<code>FUN</code>	(Optional) function to summarize the values (e.g. mean) during the extraction. See <a href="#">extract</a> for more details. Default is "mean". For some <code>plot_function</code> which map the full distribution of values the FUN is ignored.
<code>return_df</code>	(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.
<code>...</code>	(Optional) additional arguments for <code>plot_function</code> .

### 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","Ivančna Gorica","Dolenjske Toplice","Loški 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 = "Občina",
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 = "Občina",
# position_legend = FALSE,
# legend_position = "left",
# band_legend=TRUE,aes_by_pos = FALSE,
# plot_function = "violin")
#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**

<code>x_list</code>	a list of raster objects.
<code>r_type</code>	(Optional) character, one of "discrete", "gradient" or "RGB". By default, attempts to discern the <code>r_type</code> from the content of <code>r_list</code> .
<code>minq</code>	(Optional) numeric, lower quantile boundary for the stretch. Default is 0.02.
<code>maxq</code>	(Optional) numeric, upper quantile boundary for the stretch. Default is 0.98.
<code>samplesize</code>	(Optional) numeric, number of samples per layer to determine the quantile from. See <a href="#">sampleRegular</a> for details on the sampling. Default is 1000.
<code>blacken_NA</code>	(Optional) logical. If TRUE: set NA to 0. Default is FALSE.
<code>l_indices</code>	(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 <code>r_type</code> is "RGB" and the first layer if <code>r_type</code> is "discrete" or "gradient".
<code>alpha</code>	(Optional) numeric. The opacity of the plot. Default is 1.
<code>hillshade</code>	(Optional) A raster layer. If one is provided, it will be used as the base layer and plotted with <a href="#">ggR</a> while the raster layers from <code>x_list</code> will be plotted over it with a default alpha of 0.5. By default, no hillshade is used.

**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

## \* datasets

MODIS\_SI\_ds, 2  
SI\_positions, 2

add\_gg, 4  
add\_northarrow, 11  
add\_text, 4  
approxNA, 6, 7

coordinates, 3

extract, 9

ggplot, 9  
ggR, 11

MODIS\_SI\_ds, 2

rescaleImage, 11

sampleRegular, 11  
SI\_positions, 2

ts\_add\_positions\_to\_frames, 3  
ts\_fill\_na, 6  
ts\_flow\_frames, 7  
ts\_makeframes, 3, 10  
ts\_raster, 8, 9, 12