Package ‘tstools’

September 11, 2018

Type Package
Version 0.3.8
Title A Time Series Toolbox for Official Statistics
Description Plot official statistics' time series conveniently: automatic legends, highlight windows, stacked bar charts with positive and negative contributions, sum-as-line option, two y-axes with automatic horizontal grids that fit both axes and other popular chart types. 'tstools' comes with a plethora of defaults to let you plot without setting an abundance of parameters first, but gives you the flexibility to tweak the defaults. In addition to charts, 'tstools' provides a super fast, 'data.table' backed time series I/O that allows the user to export / import long format, wide format and transposed wide format data to various file types.
Depends R (>= 3.0.0), zoo (>= 1.7-12),
Imports xts, stats, graphics, jsonlite, data.table,
Suggests knitr, rmarkdown, testthat, reshape2, openxlsx
VignetteBuilder knitr
Date 2018-09-11
License GPL-2
LazyData true
Maintainer Matthias Bannert <bannert@kof.ethz.ch>
NeedsCompilation no
Author Matthias Bannert [aut, cre],
    Severin Thoeni [aut]
RoxygenNote 6.0.1
Repository CRAN
Date/Publication 2018-09-11 12:20:02 UTC

R topics documented:

     CHGDP.................................................................  2
color盲ind...............................................................  3
Description

A list of time series containing sector contributions to Swiss GDP over time.

Usage

CHGDP

Format

List list of six time series of class ts, containing contributions to Swiss GDP growth

manufacturing Growth contribution of manufacturing.
energy Growth contribution of energy, water sector
construction Growth contribution construction sector.
hotels Growth contribution of hotels.
**fin_insur**  Growth contribution of financial services and insurances.

**other**  Growth contribution of other sectors.

**Source**

https://www.seco.admin.ch/seco/en/home/wirtschaftslage---wirtschaftspolitik/Wirtschaftslage/bip-quartalsschaetzungen--/daten.html

---

### color_blind

**Provide Colorblind Compliant Colors**

**Description**

8 Hex RGB color definitions suitable for charts for colorblind people.

**Usage**

```r
color_blind()
```

### compute_decimal_time

**Compute Decimal Time from a ts Period Vector**

**Description**

Standard ts object use a vector of length two to store a period. E.g. 2010.1 means first quarter of 2010, if the series was quarterly and first month if the series was monthly etc.

**Usage**

```r
compute_decimal_time(v, f)
```

**Arguments**

- **v**: integer vector denoting a point in time
- **f**: frequency
**concat_ts**

*Concatenate to Non-Overlapping Time Series*

**Description**

Append one time series to another. This only works for non-overlapping time series of the same frequency. For overlapping time series please see `resolveOverlap`.

**Usage**

```r
concat_ts(ts1, ts2)
```

**Arguments**

- `ts1`: object of class `ts1`, typically the older of two time series.
- `ts2`: object of class `ts1`, typically the younger of two time series.

**create_dummy_ts**

*Flexible Function to Create Time Series Dummy Variables*

**Description**

Generate time series with a default value that is changed within a certain subperiod. The function allows for additional convenience when specifying single period dummies and dummies that go from a certain point in time to the end of the series.

**Usage**

```r
create_dummy_ts(end_basic, dummy_start, dummy_end = NULL, sp = T, 
                 start_basic = c(1980, 1), basic_value = 0, dummy_value = 1, 
                 frequency = 4)
```

**Arguments**

- `end_basic`: numeric vector of form `c(yyyy,p)` defining the end of the time series.
- `dummy_start`: numeric vector of form `c(yyyy,p)` defining the beginning of the period with different value.
- `dummy_end`: numeric vector of form `c(yyyy,p)` defining the end of the period with different value. Defaults to NULL, using the `end_date` of the series.
- `sp`: logical should NULL value for `dummy_end` lead to a single period dummy (TRUE) or to alternative values until the end.
- `start_basic`: numeric vector of form `c(yyyy,p)` defining the start of the time series. Defaults to `c(1980,1)`.
- `basic_value`: default value of the time series, defaults to 0.
- `dummy_value`: the alternative value, defaults to 1.
- `frequency`: integer frequency of the regular time series, defaults to 4 (quarterly).
**df_to_reg_ts**

**Author(s)**
Matthias Bannert

---

**df_to_reg_ts**  
*Turn data.frame to Regular Monthly or Quarterly Time Series*

**Description**

Turn a data.frame with date columns to a regular time series object if possible. Design to work with quarterly and monthly data.

**Usage**

```r
df_to_reg_ts(dframe, var_cols, year_col = "year", period_col = "month", freq = 12, return_ts = T, by = NULL)
```

**Arguments**

- `dframe`: data.frame input
- `var_cols`: columns that contain variables as opposed to date index.
- `year_col`: integer, logical or character vector indicating the year position within the data.frame.
- `period_col`: integer, logical or character vector indicating the period position within the data.frame.
- `freq`: integer indicating the frequency of new time series.
- `return_ts`: logical should a (list of) time series be returned? Defaults to TRUE. FALSE returns data.frame.
- `by`: character overwrite automatically detected (from freq) by parameter. e.g. ‘1 day’. Defaults to NULL.

**Examples**

```r
start_m <- as.Date("2017-01-01")
df_missing <- data.frame(
  date = seq(start_m, by='2 months', length=6),
  value = 1:6,
  another_value = letters[1:6],
  yet_another_col = letters[6:1]
)
df_to_reg_ts(df_missing,c("value","another_value"))
df_to_reg_ts(df_missing, c("value","another_value"), return_ts = FALSE)
```
fill_year_with_nas  
*Fill Up a Time Series with NAs*

**Description**

When plotting a time series you might want set the range of the plot a little wider than just the start and end date of the original series. This function add fills up the current period (typically year) with NA.

**Usage**

```r
fill_year_with_nas(x, add_periods = 1, fill_up_start = FALSE)
```

**Arguments**

- `x` object of class ts
- `add_periods` integer periods to add.
- `fill_up_start` logical should start year be filled up? Defaults to FALSE.

---

generate_random_ts  
*Generate a list of random time series*

**Description**

Useful for development or generating easily reproducible examples

**Usage**

```r
generate_random_ts(n = 1, lengths = 36, starts = 1988, frequencies = 12, ranges_min = -1, ranges_max = 1, shifts = 0, ts_names = sprintf("ts%d", 1:n), seed = 30042018, random_NAs = FALSE, random_NA_proportions = 0.1, normally_distributed = FALSE, normal_means = 0, normal_sds = 1, frequency_shifts = FALSE, frequency_shift_after = 0.5)
```

**Arguments**

- `n` The number of ts objects to generate
- `lengths` The lengths of the time series
- `starts` The start points of the time series in single number notation (e.g. 1990.5)
- `frequencies` The frequencies of the time series
- `ranges_min` The minimum values of the time series (if normally_distributed == FALSE)
- `ranges_max` The maximum values of the time series (if normally_distributed == FALSE)
shifts              The shifts of time series values per series
ts_names           The names of the ts objects in the resulting list
seed               The random seed to be used
random_NAs         Whether or not to introduce NA values at random positions in the ts
random_NA_proportions
                      The fraction of values to be replaced with NAs if random_NAs is TRUE for the series
normally_distributed
                      Use normal distribution instead of uniform
normal_means        The means to use for normal distribution. Ignored unless normally_distributed is set to TRUE.
normal_sds          The sds to use for normal distribution. Ignored unless normally_distributed is set to TRUE.
frequency_shifts    Introduce frequency shifts (from 4 to 12) in the ts
frequency_shift_after
                      After what fraction of the ts to shift frequencies

Details
Except for n and ts_names, all parameters accept either a single value or a vector of values. If a single value is supplied, that value is used for all time series being generated. If a vector is supplied, its values will be used for the corresponding series (e.g. starts[1] is used for the first series, starts[2] for the second and so on). Vectors are recycled if n is larger than their length.
If a ts_names vector is supplied, it must have length n and must not contain duplicates.

Value
A list of ts objects

Examples
generate_random_ts()
generate_random_ts(n = 3, ranges_min = c(-10, 0, 10), ranges_max = 20, starts = 2011)

getCiLegendColors  Helper to calculate ci colors for legends

Description
Helper to calculate ci colors for legends

Usage
getCiLegendColors(color, n = 1, alpha = NULL)
Arguments

- **color**: The color of the ci band
- **n**: The number if ci bands
- **alpha**: The alpha/transparency of the ci band

Details

Color may be specified as either a named color or a hex value. Transparency may be specified as a hex value, number 0-255 or number 0-1.

Value

A vector of non-transparent colors that result from overlaying color over pure white 1:n times.

---

**get_date_vector**

*Compute the Period Vector representation of a Decimal Time value*

**Description**

The period value will be rounded down to the nearest integer. This function is not vectorized so only a single value can be converted at a time.

**Usage**

```r
get_date_vector(dtime, frq)
```

**Arguments**

- **dtime**: numeric decimal time value denoting a point in time
- **frq**: integer frequency

**init_tsplot_theme**

*Initiate Default Theme*

**Description**

The *tsplot* methods provide a theme argument which is used to pass on a plethora of useful defaults. These defaults are essentially stored in a list. Sometimes the user may want to tweak some of these defaults while keeping most of them. Hence the *init_tsplot_theme* function creates a fresh list object containing default values for a lot of different layout parameters etc. By replacing single elements of the list and passing the entire list to the plot function, single aspects can be tweaked while keeping most defaults. *Init defaultTheme* does not need any parameters.

This function provides sensible defaults for margins, font size, line width etc. scaled to the dimensions of the output file.
init_tsplot_theme

Usage

init_tsplot_theme(auto_bottom_margin = FALSE, band_fill_color = c(ETH8 = 
"#007A92", ETH8_60 = "#66b0c2", ETH8_30 = "#b3d7e0", ETH8_20 = "#cce5eb", ETH5 
= "#91056a", ETH5_60 = "#cc67a7", ETH5_30 = "#e6b3d3"), 
bar_border = "#000000", bar_border_lwd = 1, bar_fill_color = c(ETH8 = 
"#007A92", ETH8_60 = "#66b0c2", ETH8_30 = "#b3d7e0", ETH8_20 = "#cce5eb", ETH5 
= "#91056a", ETH5_60 = "#cc67a7", ETH5_30 = "#e6b3d3"), bar_gap = 15, 
bar_group_gap = 30, ci_alpha = "44", ci_colors = line_colors,
 ci_legend_label = "%ci_value%% ci for %series%",
default_bottom_margin = 15, fill_up_start = FALSE, 
fill_year_with_nas = TRUE, highlight_color = "#e9e9e9",
highlight_window = FALSE, highlight_window_end = NA,
highlight_window_freq = 4, highlight_window_start = NA,
label_pos = "mid", legend_all_left = FALSE, legend_box_size = 2,
legend_col = 1, legend_font_size = 1, legend_intersp_x = 1,
legend_intersp_y = 1, legend_margin_bottom = 5, legend_margin_top = 12,
legend_seg_len = 2, line_colors = c(ETH8_100 = "#a9af66", ETH4_100 = 
"#72791c", ETH_8_20 = "#cce5eb", ETH_5_60 = "#cc67a7", ETH_8_60 = "#66b0c2",
ETH_5_100 = "#91056a", ETH_4_60 = "#007a92"), line_to_middle = TRUE,
lty = 1, lwd = c(2, 3, 1, 4, 2, 4), lwd_box = 1.5,
lwd_quarterly_ticks = 1, lwd_x_axis = 1.5, lwd_y_axis = 1.5,
lwd_y_ticks = 1.5, lwd_yearly_ticks = 1.5, margins = c(NA, 7, 12, 7),
NA_continue_line = FALSE, output_wide = FALSE, point_symbol = 1:18,
pointsize = 12, preferred_y_gap_sizes = c(25, 20, 15, 10, 5, 2.5, 1, 0.5),
quarterly_ticks = TRUE, range_must_not_cross_zero = TRUE,
show_left_y_axis = TRUE, show_points = FALSE, show_right_y_axis = TRUE,
show_x_axis = TRUE, show_y_grids = TRUE, subtitle_adj = 0,
subtitle_adj_r = 0.9, subtitle_cex = 1, subtitle_margin = 2,
subtitle_outer = FALSE, subtitle_transform = "toupper",
sum_as_line = FALSE, sum_legend = "sum", sum_line_color = c(ETH8_100 = 
"#007A92"), sum_line_lty = 1, sum_line_lwd = 3,
tcl_quarterly_ticks = -0.4, tcl_y_ticks = -0.75,
tcl_yearly_ticks = -0.75, title_adj = 0, title_cex.main = 1,
title_margin = 5, title_outer = FALSE, title_transform = NA,
total_bar_margin_pct = 0.2, use_bar_gap_in_groups = FALSE,
use_box = FALSE, x_tick_dt = 1, xaxs = "i", y_grid_color = "#cccccc",
y_grid_count = c(5, 6, 8, 10), y_grid_count_strict = FALSE, y_las = 2,
y_range_min_size = NULL, y_tick_force_integers = FALSE,
y_tick_margin = 0.15, yaxs = "i", yearly_ticks = TRUE)

init_tsplot_print_theme(output_wide = FALSE, margins = c(NA, 10/13 
(NA, 10) if (output_wide) 1 + 1/3 else 1, 10, 7/13 if (output_wide) 1 + 1/3 else 1),
lwd = scale_theme_param_for_print(c(2, 3, 1, 4, 2, 4), if (output_wide) c(10 + 2/3, 6) else c(8, 6)), sum_line_lwd = scale_theme_param_for_print(3, if (output_wide) c(10 + 2/3, 6) else c(8, 6)),
lwd_box = scale_theme_param_for_print(1.5, if (output_wide) c(10 + 2/3, 6) else c(8, 6)),
lwd_x_axis = scale_theme_param_for_print(1.5, if (output_wide) c(10 + 2/3, 6) else c(8, 6)),
lwd_yearly_ticks = scale_theme_param_for_print(1.5, if (output_wide) c(10 + 2/3, 6) else c(8, 6)), lwd_quarterly_ticks = scale_theme_param_for_print(1, if (output_wide) c(10 + 2/3, 6) else c(8, 6)),
lwd_y_axis = scale_theme_param_for_print(1.5, if (output_wide) c(10 + 2/3, 6) else c(8, 6)), lwd_y_ticks = scale_theme_param_for_print(1.5, if (output_wide) c(10 + 2/3, 6) else c(8, 6)),
legend_intersp_y = scale_theme_param_for_print(1, if (output_wide) c(10 + 2/3, 6) else c(8, 6)), legend_box_size = scale_theme_param_for_print(2, if (output_wide) c(10 + 2/3, 6) else c(8, 6)), legend_margin_top = 8, legend_margin_bottom = 3, legend_seg.len = scale_theme_param_for_print(2, if (output_wide) c(10 + 2/3, 6) else c(8, 6)),
points_size = scale_theme_param_for_print(12, if (output_wide) c(10 + 2/3, 6) else c(8, 6)), ...

Arguments

auto_bottom_margin
logical Should the bottom margin be automatically calculated? This will be overridden if margins[1] is not NA. Default FALSE

band_fill_color
character vector of hex colors for the bands if left_as_band == TRUE.

bar_border
character hex colors for the border around bars in bar charts.

bar_border_lwd
numeric The line width of the borders of bars in barplots. Default 1

bar_fill_color
character vector of hex colors for the bars if left_as_bar == TRUE

bar_gap
numeric The width of the gap between bars, in % of space allotted to the bar.

bar_group_gap
numeric The width of the gap between groups of bars if group_bar_chart is TRUE.

ci_alpha
numeric 0-255, numeric 0-1 or hey 00-FF, transparency of the confidence interval bands

ci_colors
Named colors or hex values Colors of the confidence interval bands

ci_legend_label
character A formatting template for how the ci bands should be labelled. May contain the placeholders. '%ci_value%' will be replaced with the ci label. '%series%' (will be replaced with the series name) exactly once. Defaults to '%ci_value% ci for %series%

default_bottom_margin
numeric The bottom margin to use when margins[1] is NA but neither auto_legend nor auto_bottom_margin are true. Default 3

fill_up_start
logical Should the start of the year also be filled? Has no effect if fill_year_with_nas == FALSE. Default FALSE

fill_year_with_nas
logical Should year be filled up with missing in order to plot the entire year on the axis. Defaults to TRUE,

highlight_color
character hex color code of highlight background, defaults to "#e9e9e9".

highlight_window
logical should a particular time span be highlighted by different background color. Defaults to FALSE.

highlight_window_end
integer vector highlight window start position, defaults to NA.

highlight_window_freq
integer frequency of the highlight window definition, defaults to 4.

highlight_window_start
integer vector highlight window start position, defaults to NA.

label_pos
character, denotes where the x-axis label is at. defaults to "mid", alternative value: "start".

legend_all_left
logical Should all legend entries be drawn on the left side of the plot? Default FALSE

legend_box_size
numeric The size of the squares denoting bar colors in the legend. Default 2

legend_col
integer number of columns for the legend, defaults to 3.

legend_font_size
numeric passed on to the cex parameter of legend, defaults to 1

legend_intersp_x
numeric same as base legend parameter, defaults to 1

legend_intersp_y
numeric same as base legend parameter, defaults to 1

legend_margin_bottom
numeric Distance between bottom of legend and bottom of graphic in % of device height, default 5

legend_margin_top
numeric Distance between bottom of plot and top of legends % of device height, defaults to 12

legend_segNlen
numeric Length of the line segments in the legend. Default 2

line_colors
character vector of hex colors for 6 lines.

line_to_middle
logical try to put a line into the middle of the plot. defaults to TRUE.

lty
integer vector line type defaults to 1.

lwd
integer vector line width, defaults to c(2,3,1,4,2,4).

lwd_box
numeric Line width of the box around the plot. Default 1.5

lwd_quarterly_ticks
numeric, width of yearly ticks, defaults to 1.

lwd_x_axis
numeric The line width of the x axis. Default 1.5

lwd_y_axis
numeric The line width of the y axis. Default 1.5

lwd_y_ticks
numeric Line width of the y ticks. Default 1.5

lwd_yearly_ticks
numeric, width of yearly ticks, defaults to 1.5.
margins integer vector defaults to c(NA, 4, 3, 3) + 0.1. Set margins[1] to NA to automatically determine the bottom margin such that the legend fits (if either auto_legend or auto_bottom_margin are TRUE)

NA_continue_line boolean If true, NA values in time series are ignored and a continuous line is drawn. Multiple values to turn this behavior on/off for individual series are supported. Default FALSE

output_wide logical Should the output file be in a wide format (16:9) or (4:3)? Only if output_format is not "plot". Default FALSE

point_symbol integer or character The symbol to use for marking data points. Multiple values can be supplied to set the symbol for each individual series See pch in ?par. Default 1:18

points.size Numeric Point size of text, in 1/72 of an inch

preferred_y_gap_sizes numeric c(25, 20, 15, 10, 5, 2.5, 1, 0.5),

quarterly_ticks logical, should quarterly ticks be shown. Defaults to TRUE.

range.must.not.cross.zero logical automatic range finders are forced to do not find ranges below zero. Defaults to TRUE.

show.left.y.axis logical: should left y axis be shown, defaults to TRUE.

show.points boolean Whether to draw the symbol specified by point_symbol at the data points. Multiple values can be supplied to enable/disable showing points for each individual series Default FALSE

show.right.y.axis logical: should left y axis be shown, defaults to TRUE.

show.x.axis locigal: should x axis be shown, defaults to TRUE

show.y.grids logical should y_grids be shown at all, defaults to TRUE.

subtitle_adj numeric same as base plot parameter, defaults to 0.

subtitle_adj.r numeric same as base plot parameter, defaults to .9

subtitle.cex numeric same as base plot parameter, defaults to 1.

subtitle.margin numeric How far above the plot the title is placed in % of the device height. Defaults to 2.

subtitle.outer logical same as base plot parameter, defaults to TRUE

subtitle.transform function to transform the subtitle, defaults to "toupper",

sum.as.line logical should the sum of stacked time series be displayed as a line on top of stacked bar charts. Defaults to FALSE.

sum.legend character Label for the sum line, defaults to "sum". Set to NULL to not label the line at all.

sum.line.color character hex color of of sum_as_line, defaults "#91056a".
sum_line_lty  integer line type of sum_as_line, defaults to 1.
sum_line_lwd  integer line width of sum_as_line, defaults to 3.
tcl_quarterly_ticks
    numeric, length of quarterly ticks. See tcl_yearly_ticks, defaults to -0.4
tcl_y_ticks  numeric Length of y ticks, see tcl_yearly_ticks. Default -0.75
tcl_yearly_ticks
    numeric, length of yearly ticks. Analogous to cex for axis. defaults to -0.75.
title_adj  numeric, same as base plot parameter, defaults to 0.
title_cex.main  numeric, same as base plot parameter defaults to 1
title_margin  numeric How far above the plot the title is placed in % of the device height. Default 8
title_outer  logical, currently undocumented. Defaults to TRUE.
title_transform
    function to transform the title, defaults to NA.
total_bar_margin_pct  numeric definition as in base plot, defaults to "i", defaults to .2,
use_bar_gap_in_groups
    logical Should there be gaps of size bar_gap between the bars in a group if group_bar_chart = TRUE? Default FALSE
use_box  logical use a box around the plot.
x_tick_dt  numeric The distance between ticks on the x axis in years. The first tick will always be at the start of the plotted time series. Defaults to 1.
xaxs  character axis definition as in base plot, defaults to "i".
y_grid_color  character hex color of grids. Defaults to gray "#CCCCCC".
y_grid_count  integer vector preferred y grid counts c(5,6,8,10).
y_grid_count_strict
    logical should we strictly stick to preferred y grid count? Defaults to FALSE.
y_las  integer, same as base plot parameter defaults to 2.
y_range_min_size
    = NULL ,
y_tick_force_integers
    logical Should y ticks be forced (rounded down) to whole numbers? Default FALSE
y_tick_margin
    numeric, minimal percentage of horizontal grid that needs to be clean, i.e., without lines or bars. Defaults to 0.15 (15 percent).
yaxs  character axis definition as in base plot, defaults to "i".
yearly_ticks  logical, should yearly ticks be shown. Defaults to TRUE.
Details
Themes are essentially list that contain par parameters. Below all items are listed, some of them with comments. The per-line parameters (line_colors, lwd, lty, show_points, point_symbol) are recycled if more time series than elements on the corresponding theme vectors are supplied. e.g. if four time series are plotted but only two line_colors are supplied, the first and third series have the first color, while the second and fourth series have the second color. The list contains the following elements:

Author(s)
Matthias Bannert

Examples
# create a list
data(KOF)
tt <- init_tsplo\_theme()
# adjust a single element
tt\$highlight\_window <- TRUE
# pass the list to tsplo\_t
\texttt{tslo\_t(KOF}\$klobarometer,\texttt{theme} = \texttt{tt})
# for more theme examples check the vignette
\texttt{vignette(\textquote{ttools})}

---

KOF **KOF Barometer - Swiss Business Cycle Indicator**

Description
A list of time series containing two time series the KOF Barometer and the growth of Swiss GDP over time. KOF Barometer is a monthly business cycle indicator computed by the KOF Swiss Economic Institute. The GDP growth rate is used as a reference series to the Barometer.

Usage
KOF

Format
A list of two time series of class ts

- **klobarometer** KOF Barometer Indicator
- **reference** Reference series to KOF Barometer, change in Swiss GDP compared to previous month
- **baro\_point\_fc** Auto Arima point forecast of the KOF Barometer
- **baro\_lo\_80** Auto Arima 80 percent CI lower bound of the KOF Barometer forecast
The data frame must have three columns "date", "value" and "series" (identifying the time series).

Usage

long_to_ts(data)

Arguments

data data.frame The data.frame to be transformed

m_to_q  Turn monthly series with regular NAs to quarter

Description

Monthly series with NAs in non-quarter months are turned to quarterly series. Series without NAs are just returned.

Usage

m_to_q(series)

Arguments

series an object of class ts with monthly frequency
**overlap_sorted_ts_lists**

*Concat Time Series list wise*

**Description**

Concat overlapping time series list wise. List needs to be of same length. Takes names of list B.

**Usage**

`overlap_sorted_ts_lists(listA, listB)`

**Arguments**

- **listA**: list of time series
- **listB**: list of time series

---

**overlap_ts_lists_by_name**

*Resolve Overlap Listwise, helpful with SA*

**Description**

Resolve Overlap Listwise, helpful with SA

**Usage**

`overlap_ts_lists_by_name(listA, listB, chunkA = "_f4", chunkB = "_f12")`

**Arguments**

- **listA**: list of time series often of lower frequency
- **listB**: list of time series often of higher frequency
- **chunkA**: character chunk representing frequencies, defaults to _f4.
- **chunkB**: character chunk representing frequencies, defaults to _f12.
**read_swissdata**

Read data generated by the Swissdata project

**Description**

Read data from swissdata compliant .csv files and turn them into a list of time series.

**Usage**

```r
read_swissdata(path, key_columns, filter = NULL)
```

**Arguments**

- `path` character full path to dataset.
- `key_columns` character vector specifying all columns that should be part of the key.
- `filter` function A function that is applied to the raw data.data table after it is read. Useful for filtering out undesired data.

**Details**

The order of dimensions in `key_columns` determines their order in the key. The resulting `ts_key` will be of the form `<swissdata-set-name>.<instance of key_columns[1]>...`

**Examples**

```r
ds_location <- system.file("example_data/ch.seco.css.csv", package = "tstools")
tslist <- read_swissdata(ds_location,"idx_type")
tsp(tplist[1])
```

---

**read_ts**

Import time series data from a file.

**Description**

If importing from a zip file, the archive should contain a single file with the extension .csv, .xlsx or .json.

**Usage**

```r
read_ts(file, format = c("csv", "xlsx", "json", "zip"), sep = ",", skip = 0)
```
Arguments

- **file**: Path to the file to be read
- **format**: Which file format is the data stored in? If no format is supplied, read_ts will attempt to guess from the file extension.
- **sep**: character separator for csv files. defaults to ‘,’.
- **skip**: numeric See data.table’s fread.

Value

A named list of ts objects

```
regularize(x)
```

**Description**

Adds missing values to turn an irregular time series into a regular one. This function is currently experimental. Only works or target frequencies 1,2,4,12.

**Usage**

```
regularize(x)
```

**Arguments**

- **x**: an irregular time series object of class zoo or xts.

**Examples**

```
ts1 <- rnorm(5)
dv <- c(seq(as.Date("2010-01-01"), length = 3, by="3 years"), 
    seq(as.Date("2018-01-01"), length = 2, by="2 years"))
library(zoo)
xx <- zoo(ts1,dv)
regularize(xx)

dv2 <- c(seq(as.Date("2010-01-01"), length = 20, by = "1 months"))
dv2 <- dv2[seq(1,20)]
xx2 <- zoo(rnorm(length(dv2)), dv2)
regularize(xx2)
```
resolve_ts_overlap

Concatenate Time Series and Resolve Overlap Automatically

Description

Append time series to each other. Resolve overlap determines which of two ts class time series is reaching further and arranges the two series into first and second series accordingly. Both time series are concatenated to one if both series had the same frequency. Typically this function is used to concatenate two series that have a certain overlap, but one series clearly starts earlier while the other lasts longer. If one series starts earlier and stops later, all elements of the shorter series will be inserted into the larger series, i.e. elements of the smaller series will replace the elements of the longer series. Usually ts2 is kept.

Usage

```r
resolve_ts_overlap(ts1, ts2, keep_ts2 = T, tolerance = 0.001)
```

Arguments

- `ts1`: ts time series, typically the older series
- `ts2`: ts time series, typically the younger series
- `keep_ts2`: logical should ts2 be kept? Defaults to TRUE.
- `tolerance`: numeric when comparing min and max values with a index vector of a time series R runs in to trouble with precision handling, thus a tolerance needs to be set. Typically this does not need to be adjusted. E.g. 2010 != 2010.000. With the help of the tolerance parameter these two are equal.

Examples

```r
ts1 <- ts(rnorm(100), start = c(1990, 1), frequency = 4)
ts2 <- ts(1:18, start = c(2000, 1), frequency = 4)
resolve_ts_overlap(ts1, ts2)
```

# automatical detection of correction sequence!
```r
ts1 <- ts(rnorm(90), start = c(1990, 1), frequency = 4)
ts2 <- ts(1:6, start = c(2000, 1), frequency = 4)
resolve_ts_overlap(ts1, ts2)
```

# both series are of the same length use sequence of arguments.
```r
ts1 <- ts(rnorm(100), start = c(1990, 1), frequency = 4)
ts2 <- ts(1:48, start = c(2003, 1), frequency = 4)
resolve_ts_overlap(ts1, ts2)
ts1 <- ts(rnorm(101), start = c(1990, 1), frequency = 4)
ts2 <- ts(1:61, start = c(2000, 1), frequency = 4)
resolve_ts_overlap(ts1, ts2)
```

'Clearly dominatn ts2 series
```r
ts1 <- ts(rnorm(50), start = c(1990, 1), frequency = 4)
ts2 <- ts(1:100, start = c(1990, 1), frequency = 4)
resolve_ts_overlap(ts1, ts2)
```
set_month_to_NA  Set Periods to NA

Description
This function is typically used to discard information in non-quarter month. I.e., data is only kept in January, April, July and December and otherwise set to NA. In combination with m_to_q this function is useful to turn monthly series into quarterly series by letting the quarter month values represent the entire quarter. This can be useful when data was interpolated because of mixing data of different frequencies and needs to be converted back to a regular, quarterly time series.

Usage
set_month_to_NA(series, keep_month = c(1, 4, 7, 10))

Arguments
series  ts object
keep_month  integer vector denoting the months that not be set to NA. Defaults to c(1,4,7,10)

Examples
  tsq <- ts(1:20,start=c(1990,1),frequency = 4)
  aa <- tsqm(tsq)
  m_to_q(set_month_to_NA(aa))

start_ts_after_internal_nas
Start a Time Series after the Last Internal NA

Description
Internal NAs can cause trouble for time series operations such as X-13-ARIMA SEATS seasonal adjustment. Often, internal NAs only occur at at the beginning of a time series. Thus an easy solution to the problem is to discard the initial part of the data which contains the NA values. This way only a small part of the information is lost as opposed to not being able to seasonally adjust an entire series.

Usage
start_ts_after_internal_nas(series)

Arguments
series  on object of class ts
strip_ts_of_leading_nas

See Also

stripLeadingNAsFromTs, stripTrailingNAsFromTs

Examples

```r
tsl <- 1:30
tsl[c(3,6)] <- NA
tsl <- ts(tsl, start=c(2000,1), frequency = 4)
start_ts_after_internal_nas(tsl)
```

---

strip_ts_of_leading_nas

*Strip Leading / Trailing NAs from a Time Series Object*

Description

Removes NAs to begin with and starts time series index at the first non-NA value.

Usage

```r
strip_ts_of_leading_nas(s)
```

```r
strip_ts_of_trailing_nas(s)
```

Arguments

- `s` an object of class `ts`.

---

tsplot

*Plot Time Series*

Description

Conveniently plot time series.

Usage

```r
tsplot(..., tsr = NULL, ci = NULL, left_as_bar = FALSE,
group_bar_chart = FALSE, relative_bar_chart = FALSE,
left_as_band = FALSE, plot_title = NULL, plot_subtitle = NULL,
plot_subtitle_r = NULL, find_ticks_function = "findTicks",
overall_xlim = NULL, overall_ylim = NULL, manual_date_ticks = NULL,
manual_value_ticks_l = NULL, manual_value_ticks_r = NULL,
manual_ticks_x = NULL, theme = NULL, quiet = TRUE, auto_legend = TRUE,
output_format = "plot", filename = "tsplot",
close_graphics_device = TRUE)
```
Arguments

... multiple objects of class ts or a list of time series. All objects passed through the
... parameter relate to the standard left y-axis.

\texttt{tsr} list of time series objects of class ts.
\texttt{ci} list of confidence intervals for time series
\texttt{left\_as\_bar} logical should the series that relate to the left bar be drawn as (stacked) bar
charts?
\texttt{group\_bar\_chart} logical should a bar chart be grouped instead of stacked?
\texttt{relative\_bar\_chart} logical Should time series be normalized such that bars range from 0 to 1? De-
defaults to FALSE. That way every sub bar (time series) is related to the global
max. Hence do not expect every single bar to reach 1. This works for stacked
and grouped charts and does not change anything but the scale of the chart.
\texttt{left\_as\_band} logical Should the time series assigned to the left axis be displayed as stacked
area charts?
\texttt{plot\_title} character title to be added to the plot
\texttt{plot\_subtitle} character subtitle to be added to the plot
\texttt{plot\_subtitle\_r} character second subtitle to be added at the top right
\texttt{find\_ticks\_function} function to compute ticks.
\texttt{overall\_xlim} integer overall x-axis limits, defaults to NULL.
\texttt{overall\_ylim} integer overall y-axis limits, defaults to NULL.
\texttt{manual\_date\_ticks} character vector of manual date ticks.
\texttt{manual\_value\_ticks\_l} numeric vector, forcing ticks to the left y-axis
\texttt{manual\_value\_ticks\_r} numeric vector, forcing ticks to the right y-axis
\texttt{manual\_ticks\_x} numeric vector, forcing ticks on the x axis
\texttt{theme} list of default plot output parameters. Defaults to NULL, which leads to \texttt{init\_tsplot\_theme}
being called. Please see the vignette for details about tweaking themes.
\texttt{quiet} logical suppress output, defaults to TRUE.
\texttt{auto\_legend} logical should legends be printed automatically, defaults to TRUE.
\texttt{output\_format} character Should the plot be drawn on screen or written to a file? Possible values
are "plot" for screen output and "pdf". Default "plot"
\texttt{filename} character Path to the file to be written if output\_format is "pdf". Default
"tsplot.pdf"
\texttt{close\_graphics\_device} logical Should the graphics device of the output file be closed after tsplot? Set
this to FALSE to be able to make modifications to the plot after tsplot finishes.
Default TRUE
The `ci` parameter is a 3-level list of the form 
```r
list( ts1 = list( ci_value_1 = list( ub = upper_bound_ts_object, lb = lower_bound_ts_object ), ... ), ... )
```
See vignette("tstools") for details.

---

**Interpolate quarterly time series into monthly**

**Description**
Repeat quarterly variables two times to generate a monthly variable.

**Usage**
```
tsqm(qts)
```

**Arguments**
- `qts` quarterly time series

**Examples**
```
ts <- ts(1:20,start=c(1990,1),frequency = 4)
tsqm(tsq)
```

---

**Deprecated function(s) in tstools**

**Description**
These functions are provided for compatibility with older version of the tstools package. They may eventually be completely removed.

**Arguments**
- `...` Parameters to be passed to the modern version of the function
Details

- `computeDecimalTime` now a synonym for `compute_decimal_time`
- `concatTs` now a synonym for `concat_ts`
- `fillupYearWithNAS` now a synonym for `fill_year_with_nas`
- `importTimeSeries` now a synonym for `read_ts`
- `init_tsplot_theme` now a synonym for `init_tsplot_theme`
- `overlapSortedLists` now a synonym for `overlap_sorted_ts_lists`
- `overlapTsByName` now a synonym for `overlap_ts_lists_by_name`
- `resolveOverlap` now a synonym for `resolve_ts_overlap`
- `stripLeadingNASFromTs` now a synonym for `strip_ts_of_leading_nas`
- `stripTrailingNASFromTs` now a synonym for `strip_ts_of_trailing_nas`
- `writeTimeSeries` now a synonym for `write_ts`

```
wide_to_ts      Transform a wide format data.frame into a tslist

Description

The time series in the data.frame may be stored either rowwise or columnwise. The identifying
column must be called date (for columnwise) or series (for rowwise)

Usage

wide_to_ts(data)

Arguments

data     data.frame The data.frame to be transformed

write_ts     Export a list of time series to a file.

Description

Export a list of time series to a file.

Usage

write_ts(t1, fname = NULL, format = "csv", date_format = NULL,
timestamp_to_fn = FALSE, round_digits = NULL, rdata_varname = "tslist",
...
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tl</td>
<td>list of time series</td>
</tr>
<tr>
<td>fname</td>
<td>character file name. Defaults to NULL, displaying output on console. Set a file name without file extension in order to store a file. Default file names / location are not CRAN compliant which is why the file name defaults to NULL.</td>
</tr>
<tr>
<td>format</td>
<td>character denotes export formats. Defaults to .csv, &quot;csv&quot;, &quot;xlsx&quot;, &quot;json&quot;, &quot;rdata&quot; are available. Spreadsheet formats like csv allow for further optional parameters.</td>
</tr>
<tr>
<td>date_format</td>
<td>character denotes the date format. Defaults to NULL. If set to null the default is used: Jan 2010.</td>
</tr>
<tr>
<td>timestamp_to_fn</td>
<td>If TRUE, the current date will be appended to the file name. Defaults to FALSE.</td>
</tr>
<tr>
<td>round_digits</td>
<td>integer, precision in digits.</td>
</tr>
<tr>
<td>rdata_varname</td>
<td>character name of the list of time series within the store RData. Defaults to &quot;tslist&quot;.</td>
</tr>
</tbody>
</table>

Details

Additional arguments covered by . . .

<table>
<thead>
<tr>
<th>Name</th>
<th>Effect</th>
<th>Format(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>wide</td>
<td>Export data in a wide format (one column per series)</td>
<td>CSV, XLSX</td>
</tr>
<tr>
<td>transpose</td>
<td>Transpose exported data (one row per series)</td>
<td>CSV, XLSX, only if wide = TRUE</td>
</tr>
<tr>
<td>zip</td>
<td>If set to TRUE, the file is compressed into a zip archive after export</td>
<td>any</td>
</tr>
</tbody>
</table>
Index

*Topic datasets
   CHGDP, 2
   KOF, 14
axis, 13
   CHGDP, 2
   color_blind, 3
   compute_decimal_time, 3, 24
   computeDecimalTime
      (tstools-deprecated), 23
   concat_ts, 4, 24
   concatTs(tstools-deprecated), 23
   create_dummy_ts, 4
   df_to_reg_ts, 5
   fill_year_with_nas, 6, 24
   fillupYearWithNAs(tstools-deprecated), 23
   generate_random_ts, 6
   get_date_vector, 8
   getCiLegendColors, 7
   importTimeSeries(tstools-deprecated), 23
   init_tspot_print_theme
      (init_tspot_theme), 8
   init_tspot_theme, 8, 22, 24
   initDefaultTheme(tstools-deprecated), 23
   KOF, 14
   legend, 11
   long_to_ts, 15
   m_to_q, 15, 20
   overlap_sorted_ts_lists, 16, 24
   overlap_ts_lists_by_name, 16, 24
   overlapSortedLists
      (tstools-deprecated), 23
   overlapTslByName(tstools-deprecated), 23
   par, 14
   plot, 12, 13
   read_swissdata, 17
   read_ts, 17, 24
   regularize, 18
   resolve_ts_overlap, 19, 24
   resolveOverlap, 4
   resolveOverlap(tstools-deprecated), 23
   set_month_to_NA, 20
   start_ts_after_internal_nas, 20
   strip_ts_of_leading_nas, 21, 24
   strip_ts_of_trailing_nas, 24
   strip_ts_of_trailing_nas
      (strip_ts_of_leading_nas), 21
   stripLeadingNASFromTs, 21
   stripLeadingNASFromTs
      (tstools-deprecated), 23
   stripTrailingNASFromTs, 21
   stripTrailingNASFromTs
      (tstools-deprecated), 23
   tsplot, 8, 21
   tsqm, 23
   tstools-deprecated, 23
   tstools-deprecated-package
      (tstools-deprecated), 23
   write_ts, 24, 24
   writeTimeSeries(tstools-deprecated), 23

26