

Package ‘tsbox’

October 14, 2022

Type Package

Title Class-Agnostic Time Series

Version 0.3.1

Description Time series toolkit with identical behavior for all time series classes: 'ts', 'xts', 'data.frame', 'data.table', 'tibble', 'zoo', 'timeSeries', 'tsibble', 'tis' or 'irts'. Also converts reliably between these classes.

Imports data.table (>= 1.10.0), anytime

Suggests testthat, dplyr, tibble, forecast, seasonal, dygraphs, xts, ggplot2, scales, knitr, rmarkdown, tsibble (>= 0.8.2), tsibbledata, tibblertime, tseries, zoo, tis, timeSeries, nycflights13, imputeTS, spelling

License GPL-3

Encoding UTF-8

URL <https://www.tsbox.help>, <https://github.com/christophsax/tsbox>

BugReports <https://github.com/christophsax/tsbox/issues>

RoxygenNote 7.1.2

VignetteBuilder knitr

Depends R (>= 2.10)

Language en-US

NeedsCompilation no

Author Christoph Sax [aut, cre] (<<https://orcid.org/0000-0002-7192-7044>>)

Maintainer Christoph Sax <christoph.sax@gmail.com>

Repository CRAN

Date/Publication 2021-09-16 07:00:02 UTC

R topics documented:

tsbox-package	2
copy_class	3

relevant_class	4
tsbox-deprecated	4
ts_	5
ts_arithmetic	6
ts_bind	7
ts_boxable	8
ts_c	8
ts_default	9
ts_dts	10
ts_examples	10
ts_first_of_period	12
ts_frequency	12
ts_ggplot	14
ts_index	15
ts_lag	16
ts_long	17
ts_na_omit	18
ts_pc	19
ts_pick	20
ts_plot	21
ts_regular	22
ts_save	23
ts_scale	23
ts_span	24
ts_summary	25
ts_trend	26
ts_ts	27

Index	30
--------------	-----------

tsbox-package	<i>tsbox: Class-Agnostic Time Series</i>
---------------	--

Description

The R ecosystem knows a vast number of time series classes: `ts`, `xts`, `zoo`, `tsibble`, `tibbletime`, `tis`, or `timeSeries`. The plethora of standards causes confusion. As different packages rely on different classes, it is hard to use them in the same analysis. `tsbox` provides a set of tools that make it easy to switch between these classes. It also allows the user to treat time series as plain data frames, facilitating the use with tools that assume rectangular data.

Details

The package is built around a set of functions that convert time series of different classes to each other. They are frequency-agnostic, and allow the user to combine multiple non-standard and irregular frequencies. Because coercion works reliably, it is easy to write functions that work identically for all classes. So whether we want to smooth, scale, differentiate, chain-link, forecast, regularize or seasonally adjust a time series, we can use the same `tsbox`-command for any time series class.

The best way to start is to check out the package [website](#).

Author(s)

Christoph Sax <christoph.sax@gmail.com>

copy_class

Re-Class ts-Boxable Object

Description

Copies class attributes from an existing ts-boxable series. Mainly used internally.

Usage

```
copy_class(  
  x,  
  template,  
  preserve.mode = TRUE,  
  preserve.names = FALSE,  
  preserve.time = FALSE  
)
```

Arguments

x	ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.
template	ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.
preserve.mode	should the mode the time column be preserved (data frame only)
preserve.names	should the name of the time column be preserved (data frame only)
preserve.time	should the values time column be preserved (data frame only)

Details

Inspired by `xts::reclass`, which does something similar.

relevant_class	<i>Extract Relevant Class</i>
----------------	-------------------------------

Description

Mainly used internally.

Usage

```
relevant_class(x)
```

Arguments

x ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.

Examples

```
relevant_class(AirPassengers)
relevant_class(ts_df(AirPassengers))
```

tsbox-deprecated	<i>Start and end of time series</i>
------------------	-------------------------------------

Description

Start and end of time series

Usage

```
ts_start(x)
```

```
ts_end(x)
```

Arguments

x ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.

Description

ts_ turns an existing function into a function that can deal with ts-boxable time series objects.

Usage

```
load_suggested(pkg)
```

```
ts_(fun, class = "ts", vectorize = FALSE, reclass = TRUE)
```

```
ts_apply(x, fun, ...)
```

Arguments

pkg	external package, to be suggested (automatically added by ts_) predict(). (See examples)
fun	function, to be made available to all time series classes
class	class that the function uses as its first argument
vectorize	should the function be vectorized? (not yet implemented)
reclass	logical, should the new function return the same same ts-boxable output as imputed?
x	ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.
...	arguments passed to subfunction

Details

The ts_ function is a constructor function for tsbox time series functions. It can be used to wrap any function that works with time series. The default is set to R base "ts" class. ts_ deals with the conversion stuff, 'vectorizes' the function so that it can be used with multiple time series.

Value

A function that accepts ts-boxable time series as an input.

See Also

[ts_examples](#), for a few useful examples of functions generated by ts_.

[Vignette](#) on how to make arbitrary functions ts-boxable.

Examples

```

ts_(rowSums)(ts_c(mdeaths, fdeaths))
ts_plot(mean = ts_(rowMeans)(ts_c(mdeaths, fdeaths)), mdeaths, fdeaths)
ts_(function(x) predict(prcomp(x)))(ts_c(mdeaths, fdeaths))
ts_(function(x) predict(prcomp(x, scale = TRUE)))(ts_c(mdeaths, fdeaths))
ts_(dygraphs::dygraph, class = "xts")

# attach series to serach path
ts_attach <- ts_(attach, class = "tslist", reclass = FALSE)
ts_attach(EuStockMarkets)
ts_plot(DAX, SMI)
detach()

```

ts_arithmetic

Arithmetic Operators for ts-boxable objects

Description

Arithmetic Operators for ts-boxable objects

Usage

e1 %ts+% e2

e1 %ts-% e2

e1 %ts*% e2

e1 %ts/% e2

Arguments

e1 ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.

e2 ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.

Value

a ts-boxable time series, with the same class as the left input.

Examples

```

head(fdeaths - mdeaths)
head(fdeaths %ts-% mdeaths)
head(ts_df(fdeaths) %ts-% mdeaths)

```

`ts_bind`*Bind Time Series*

Description

Combine time series to a new, single time series. `ts_bind` combines time series as they are, `ts_chain` chains them together, using percentage change rates.

Usage

```
ts_bind(...)
```

```
ts_chain(...)
```

Arguments

... `ts`-boxable time series, an object of class `ts`, `xts`, `zoo`, `data.frame`, `data.table`, `tbl`, `tbl_ts`, `tbl_time`, `tis`, `irts` or `timeSeries`.

Value

A `ts`-boxable object of the same class as the input. If series of different classes are combined, the class of the first series is used (if possible).

See Also

[ts_c](#) to collect multiple time series

Examples

```
ts_bind(ts_span(mdeaths, end = "1975-12-01"), fdeaths)
ts_bind(mdeaths, c(2, 2))
ts_bind(mdeaths, 3, ts_bind(fdeaths, c(99, 2)))
ts_bind(ts_dt(mdeaths), AirPassengers)

# numeric vectors
ts_bind(12, AirPassengers, c(2, 3))

ts_chain(ts_span(mdeaths, end = "1975-12-01"), fdeaths)

ts_plot(ts_pc(ts_c(
  comb = ts_chain(ts_span(mdeaths, end = "1975-12-01"), fdeaths),
  fdeaths
)))
```

ts_boxable	<i>Test if an Object is ts-Boxable</i>
------------	--

Description

Mainly used internally.

Usage

```
ts_boxable(x)
```

Arguments

x ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.

Value

logical, either TRUE or FALSE

Examples

```
ts_boxable(AirPassengers)
ts_boxable(lm)
```

ts_c	<i>Collect Time Series</i>
------	----------------------------

Description

Collect time series as multiple time series.

Usage

```
ts_c(...)
```

Arguments

... ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.

Details

In data frame objects, multiple time series are stored in a long data frame. In ts and xts objects, time series are combined horizontally.

Value

a ts-boxable object of the same class as the input. If series of different classes are combined, the class of the first series is used (if possible).

See Also

[ts_bind](#), to bind multiple time series to a single series.

Examples

```
head(ts_c(ts_df(EuStockMarkets), AirPassengers))

# labeling
x <- ts_c(
  `International Airline Passengers` = ts_xts(AirPassengers),
  `Deaths from Lung Diseases` = ldeaths
)
head(x)
```

ts_default

Default Column Names

Description

In data frame objects (`data.frame`, `tibble`, `data.table`), `tsbox` automatically detects the time and the value column. This function changes the column names to the defaults (`time`, `value`), so that auto-detection can be avoided in future operations.

Usage

```
ts_default(x)
```

Arguments

`x` ts-boxable time series, an object of class `ts`, `xts`, `zoo`, `data.frame`, `data.table`, `tbl`, `tbl_ts`, `tbl_time`, `tis`, `irts` or `timeSeries`.

Value

a ts-boxable time series, with the same class as the input.

Examples

```
df <- ts_df(ts_c(mdeaths, fdeaths))
# non-default colnames
colnames(df) <- c("id", "date", "count")
# switch back to default colnames
head(ts_default(df))
```

ts_dts	<i>Internal Time Series Class</i>
--------	-----------------------------------

Description

Internal Time Series Class

Usage

ts_dts(x)

Arguments

x ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.

ts_examples	<i>Principal Components, Dygraphs, Forecasts, Seasonal Adjustment</i>
-------------	---

Description

Example Functions, Generated by [ts_](#). ts_prcomp calculates the principal components of multiple time series, ts_dygraphs generates an interactive graphical visualization, ts_forecast return an univariate forecast, ts_seas the seasonally adjusted series. ts_na_interpolation imputes missing values.

Usage

ts_prcomp(x, ...)

ts_dygraphs(x, ...)

ts_forecast(x, ...)

ts_seas(x, ...)

ts_na_interpolation(x, ...)

Arguments

x ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.

... further arguments, passed to the underlying function. For help, consider these functions, e.g., [stats::prcomp](#).

Details

With the exception of `ts_prcomp`, these functions depend on external packages.

Value

Usually, a ts-boxable time series, with the same class as the input. `ts_dygraphs` draws a plot.

See Also

[Vignette](#) on how to make arbitrary functions ts-boxable.

Examples

```
ts_plot(
  ts_scale(ts_c(
    Male = mdeaths,
    Female = fdeaths,
    `First principal component` = -ts_prcomp(ts_c(mdeaths, fdeaths))[, 1]
  )),
  title = "Deaths from lung diseases",
  subtitle = "Normalized values"
)

ts_plot(ts_c(
  male = mdeaths, female = fdeaths,
  ts_forecast(ts_c(`male (fct)` = mdeaths, `female (fct)` = fdeaths))),
  title = "Deaths from lung diseases",
  subtitle = "Exponential smoothing forecast"
)

ts_plot(
  `Raw series` = AirPassengers,
  `Adjusted series` = ts_seas(AirPassengers),
  title = "Airline passengers",
  subtitle = "X-13 seasonal adjustment"
)

# See ?imputeTS::na_interpolation for options
dta <- ts_c(mdeaths, fdeaths)
dta[c(1, 3, 10), c(1, 2)] <- NA
head(ts_na_interpolation(dta, option = "spline"))

ts_dygraphs(ts_c(mdeaths, EuStockMarkets))
```

ts_first_of_period	<i>Use First Date of a Period</i>
--------------------	-----------------------------------

Description

Replace date or time values by the first of the period. tsbox usually relies on timestamps being the first value of a period.

Usage

```
ts_first_of_period(x)
```

Arguments

x ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.

Value

A ts-boxable object of the same class as the input.

Examples

```
x <- ts_c(
  a = ts_lag(ts_df(mdeaths), "14 days"),
  b = ts_lag(ts_df(mdeaths), "-2 days")
)
ts_first_of_period(x)
ts_first_of_period(ts_lag(ts_df(austres), "14 days"))

x <- ts_lag(data.frame(
  time = seq(anytime::anytime(1970), length.out = 10, by = "10 sec"),
  value = rnorm(10)
), "3 sec")
ts_first_of_period(x)
```

ts_frequency	<i>Change Frequency</i>
--------------	-------------------------

Description

Changes the frequency of a time series. By default, incomplete periods of regular series are omitted.

Usage

```
ts_frequency(  
  x,  
  to = c("year", "quarter", "month", "week", "day", "hour", "min", "sec"),  
  aggregate = "mean",  
  na.rm = FALSE  
)
```

Arguments

x	ts-boxable time series, an object of class <code>ts</code> , <code>xts</code> , <code>zoo</code> , <code>data.frame</code> , <code>data.table</code> , <code>tbl</code> , <code>tbl_ts</code> , <code>tbl_time</code> , <code>tis</code> , <code>irts</code> or <code>timeSeries</code> .
to	desired frequency, either a character string ("year", "quarter", "month") or an integer (1, 4, 12).
aggregate	character string, or function. Either "mean", "sum", "first", or "last", or any aggregate function, such as <code>base::mean()</code> .
na.rm	logical, if TRUE, incomplete periods are aggregated as well. For irregular series, incomplete periods are always aggregated.

Details

The `tempdisagg` package can convert low frequency to high frequency data and has support for ts-boxable objects. See `vignette("hf-disagg", package = "tempdisagg")`.

Value

a ts-boxable time series, with the same class as the input.

Examples

```
ts_frequency(cbind(mdeaths, fdeaths), "year", "sum")  
ts_frequency(cbind(mdeaths, fdeaths), "quarter", "last")  
  
ts_frequency(AirPassengers, 4, "sum")  
  
# Note that incomplete years are omitted by default  
ts_frequency(EuStockMarkets, "year")  
ts_frequency(EuStockMarkets, "year", na.rm = TRUE)
```

 ts_ggplot

Plot Time Series, Using ggplot2

Description

ts_ggplot() has the same syntax and produces a similar plot as [ts_plot\(\)](#), but uses the [ggplot2](#) graphic system, and can be customized. With [theme_tsbox\(\)](#) and [scale_color_tsbox\(\)](#), the output of ts_ggplot has a similar look and feel.

Usage

```
ts_ggplot(..., title, subtitle, ylab = "")

theme_tsbox(base_family = getOption("ts_font", ""), base_size = 12)

colors_tsbox()

scale_color_tsbox(...)

scale_fill_tsbox(...)
```

Arguments

...	ts-boxable time series, objects of class <code>ts</code> , <code>xts</code> , <code>data.frame</code> , <code>data.table</code> , or <code>tibble</code> . For <code>scale_</code> functions, arguments passed to subfunctions.
title	title (optional)
subtitle	subtitle (optional)
ylab	ylab (optional)
base_family	base font family (can also be set via options)
base_size	base font size

Details

Both [ts_plot\(\)](#) and [ts_ggplot\(\)](#) combine multiple ID dimensions into a single dimension. To plot multiple dimensions in different shapes, facets, etc., use standard `ggplot` (see examples).

See Also

[ts_plot\(\)](#), for a simpler and faster plotting function. [ts_dygraphs\(\)](#), for interactive time series plots.

Examples

```

# using the ggplot2 graphic system
p <- ts_ggplot(total = ldeaths, female = fdeaths, male = mdeaths)
p

# with themes for the look and feel of ts_plot()
p + theme_tsbox() + scale_color_tsbox()

# also use themes with standard ggplot
suppressMessages(library(ggplot2))
df <- ts_df(ts_c(total = ldeaths, female = fdeaths, male = mdeaths))
ggplot(df, aes(x = time, y = value)) +
  facet_wrap("id") +
  geom_line() +
  theme_tsbox() +
  scale_color_tsbox()

## Not run:
library(dataseries)
dta <- ds(c("GDP.PBRTT.A.R", "CCI.CCIIR"), "xts")
ts_ggplot(ts_scale(ts_span(
  ts_c(
    `GDP Growth` = ts_pc(dta[, 'GDP.PBRTT.A.R']),
    `Consumer Sentiment Index` = dta[, 'CCI.CCIIR']
  ),
  start = "1995-01-01"
))) +
ggplot2::ggtitle("GDP and Consumer Sentiment", subtitle = "normalized") +
theme_tsbox() +
scale_color_tsbox()

## End(Not run)

```

ts_index

Indices from Levels or Percentage Rates

Description

ts_index returns an indexed series, with value of 1 at the base date or range. ts_compound builds an index from percentage change rates, starting with 1 and compounding the rates.

Usage

```
ts_compound(x, denominator = 100)
```

```
ts_index(x, base = NULL)
```

Arguments

x	ts-boxable time series, an object of class <code>ts</code> , <code>xts</code> , <code>zoo</code> , <code>data.frame</code> , <code>data.table</code> , <code>tbl</code> , <code>tbl_ts</code> , <code>tbl_time</code> , <code>tis</code> , <code>irts</code> or <code>timeSeries</code> .
denominator	numeric, set equal to one if percentage change rate is given a decimal fraction
base	base date, character string, <code>Date</code> or <code>POSIXct</code> , at which the index is set to 1. If two dates are provided, the average with the range is set equal to 1 (see examples).

Value

a ts-boxable time series, with the same class as the input.

Examples

```
head(ts_compound(ts_pc(ts_c(fdeaths, mdeaths))))
head(ts_index(ts_df(ts_c(fdeaths, mdeaths)), "1974-02-01"))

ts_plot(
  `My Expert Knowledge` = ts_chain(
    mdeaths,
    ts_compound(ts_bind(ts_pc(mdeaths), 15, 23, 33))),
  `So Far` = mdeaths,
  title = "A Very Manual Forecast"
)

# average of 1974 = 1
ts_index(mdeaths, c("1974-01-01", "1974-12-31"))
```

 ts_lag

Lag or Lead of Time Series

Description

Shift time stamps in ts-boxable time series, either by a number of periods or by a fixed amount of time.

Usage

```
ts_lag(x, by = 1)
```

Arguments

x	ts-boxable time series, an object of class <code>ts</code> , <code>xts</code> , <code>zoo</code> , <code>data.frame</code> , <code>data.table</code> , <code>tbl</code> , <code>tbl_ts</code> , <code>tbl_time</code> , <code>tis</code> , <code>irts</code> or <code>timeSeries</code> .
by	integer or character, either the number of shifting periods (integer), or an absolute amount of time (character). See details.

Details

The lag order, `by`, is defined the opposite way as in R base. Thus, `-1` is a lead and `+1` a lag.

If `by` is integer, the time stamp is shifted by the number of periods. This requires the series to be regular.

If `by` is character, the time stamp is shifted by a specific amount of time. This can be one of `"sec"`, `"min"`, `"hour"`, `"day"`, `"week"`, `"month"`, `"quarter"` or `"year"`, optionally preceded by a (positive or negative) integer and a space, or followed by plural `"s"`. This is passed to `base::seq.Date()`. This does not require the series to be regular.

Value

a ts-boxable time series, with the same class as the input. If time stamp shifting causes the object to be irregular, a data frame is returned.

Examples

```
ts_plot(AirPassengers, ts_lag(AirPassengers), title = "The need for glasses")
```

```
head(ts_lag(fdeaths, "1 month"))
head(ts_lag(fdeaths, "1 year"))
head(ts_lag(ts_df(fdeaths), "2 day"))
head(ts_lag(ts_df(fdeaths), "2 min"))
head(ts_lag(ts_df(fdeaths), "-1 day"))
```

 ts_long

Reshaping Multiple Time Series

Description

Functions to reshape multiple time series from 'wide' to 'long' and vice versa. Note that long format data frames are ts-boxable objects, where wide format data frames are not. `ts_long` automatically identifies a **time** column, and uses columns on the left as id columns.

Usage

```
ts_long(x)
```

```
ts_wide(x)
```

Arguments

`x` a ts-boxable time series, or a wide data.frame, data.table, or tibble.

Value

object with the same class as input

Examples

```
df.wide <- ts_wide(ts_df(ts_c(mdeaths, fdeaths)))
head(df.wide)
head(ts_long(df.wide))
```

ts_na_omit

Omit NA values

Description

Remove NA values in ts-boxable objects, turning explicit into implicit missing values.

Usage

```
ts_na_omit(x)
```

Arguments

x ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.

Details

Note that internal NAs in ts time series will not be removed, as this conflicts with the regular structure.

Value

a ts-boxable time series, with the same class as the input.

See Also

[ts_regular](#), for the opposite, turning implicit into explicit missing values.

Examples

```
x <- AirPassengers
x[c(2, 4)] <- NA

# A ts object does only know explicit NAs
head(ts_na_omit(x))

# by default, NAs are implicit in data frames
head(ts_df(x))

# make NAs explicit
head(ts_regular(ts_df(x)))

# and implicit again
head(ts_na_omit(ts_regular(ts_df(x))))
```

`ts_pc`*First Differences and Percentage Change Rates*

Description

`ts_pcy` and `ts_diffy` calculate the percentage change rate and the difference compared to the previous period, `ts_pcy` and `ts_diffy` calculate the percentage change rate compared to the same period of the previous year. `ts_pca` calculates annualized percentage change rates compared to the previous period.

Usage

```
ts_pc(x)
ts_diff(x)
ts_pca(x)
ts_pcy(x)
ts_diffy(x)
```

Arguments

`x` ts-boxable time series, an object of class `ts`, `xts`, `zoo`, `data.frame`, `data.table`, `tbl`, `tbl_ts`, `tbl_time`, `tis`, `irts` or `timeSeries`.

Value

a ts-boxable time series, with the same class as the input.

Examples

```
tail(ts_diff(ts_c(fdeaths, mdeaths)))
tail(ts_pc(ts_c(fdeaths, mdeaths)))
tail(ts_pca(ts_c(fdeaths, mdeaths)))
tail(ts_pcy(ts_c(fdeaths, mdeaths)))
tail(ts_diffy(ts_c(fdeaths, mdeaths)))
```

ts_pick	<i>Pick Series (Experimental)</i>
---------	-----------------------------------

Description

Pick (and optionally rename) series from multiple time series.

Usage

```
ts_pick(x, ...)
```

Arguments

x	ts-boxable time series, an object of class <code>ts</code> , <code>xts</code> , <code>zoo</code> , <code>data.frame</code> , <code>data.table</code> , <code>tbl</code> , <code>tbl_ts</code> , <code>tbl_time</code> , <code>tis</code> , <code>irts</code> or <code>timeSeries</code> .
...	character string(s), names of the series to be picked, or integer, with positions. If arguments are named, the series will be renamed.

Value

a ts-boxable time series, with the same class as the input.

Examples

```
# Interactive use

ts_plot(ts_pick(
  EuStockMarkets,
  `My Dax` = "DAX",
  `My SMI` = "SMI"
))
head(ts_pick(EuStockMarkets, c(1, 2)))
head(ts_pick(EuStockMarkets, `My Dax` = 'DAX', `My SMI` = 'SMI'))

# Programming use
to.be.picked.and.renamed <- c(`My Dax` = "DAX", `My SMI` = "SMI")
head(ts_pick(EuStockMarkets, to.be.picked.and.renamed))
```

ts_plot	<i>Plot Time Series</i>
---------	-------------------------

Description

ts_plot() is a fast and simple plotting function for ts-boxable time series, with limited customizability. For more theme options, use [ts_ggplot\(\)](#).

Usage

```
ts_plot(..., title, subtitle, ylab = "", family = getOption("ts_font", "sans"))
```

Arguments

...	ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.
title	title (optional)
subtitle	subtitle (optional)
ylab	ylab (optional)
family	font family (optional, can also be set via options)

Details

Both ts_plot() and [ts_ggplot\(\)](#) combine multiple ID dimensions into a single dimension. To plot multiple dimensions in different shapes, facets, etc., use standard ggplot.

Limited customizability of ts_plot is available via options. See examples.

See Also

[ts_ggplot\(\)](#), for a plotting function based on ggplot2. [ts_dygraphs\(\)](#), for interactive time series plots. [ts_save\(\)](#) to save a plot to the file system.

Examples

```
ts_plot(
  AirPassengers,
  title = "Airline passengers",
  subtitle = "The classic Box & Jenkins airline data"
)

# naming arguments
ts_plot(total = ldeaths, female = fdeaths, male = mdeaths)

# using different ts-boxable objects
ts_plot(ts_scale(ts_c(
  ts_xts(airmiles),
```

```

    ts_tbl(co2),
    JohnsonJohnson,
    ts_df(discoveries)
  )))

# customize ts_plot
op <- options(
  tsbox.lwd = 3,
  tsbox.col = c("gray51", "gray11"),
  tsbox.lty = "dashed"
)
ts_plot(
  "Female" = fdeaths,
  "Male" = mdeaths
)
options(op) # restore defaults

```

ts_regular

Enforce Regularity

Description

Enforces regularity in data frame and xts objects, by turning implicit NAs into explicit NAs. In ts objects, regularity is automatically enforced.

Usage

```
ts_regular(x, fill = NA)
```

Arguments

x	ts-boxable time series, an object of class ts, xts, zoo, data.frame, data.table, tbl, tbl_ts, tbl_time, tis, irts or timeSeries.
fill	instead of NA, an alternative value can be specified

Examples

```

x0 <- AirPassengers
x0[c(10, 15)] <- NA
x <- ts_na_omit(ts_dts(x0))
ts_regular(x)
ts_regular(x, fill = 0)

m <- mdeaths
m[c(10, 69)] <- NA
f <- fdeaths
f[c(1, 3, 15)] <- NA

ts_regular(ts_na_omit(ts_dts(ts_c(f, m))))

```

ts_save	<i>Save Previous Plot</i>
---------	---------------------------

Description

Save Previous Plot

Usage

```
ts_save(
  filename = tempfile(fileext = ".pdf"),
  width = 10,
  height = 5,
  device = NULL,
  open = TRUE
)
```

Arguments

filename	filename
width	width
height	height
device	device
open	logical, should the saved plot be opened?

ts_scale	<i>Normalized Time Series</i>
----------	-------------------------------

Description

Subtract mean and divide by standard deviation. Based on [base::scale\(\)](#).

Usage

```
ts_scale(x, center = TRUE, scale = TRUE)
```

Arguments

x	ts_boxable time series
center	logical
scale	logical

Examples

```
ts_plot(ts_scale((ts_c(airmiles, co2, JohnsonJohnson, discoveries))))
ts_plot(ts_scale(ts_c(AirPassengers, DAX = EuStockMarkets[, 'DAX'])))
```

ts_span

Limit Time Span

Description

Filter time series for a time span.

Usage

```
ts_span(x, start = NULL, end = NULL, template = NULL, extend = FALSE)
```

Arguments

x	ts-boxable time series, an object of class <code>ts</code> , <code>xts</code> , <code>zoo</code> , <code>data.frame</code> , <code>data.table</code> , <code>tbl</code> , <code>tbl_ts</code> , <code>tbl_time</code> , <code>tis</code> , <code>irts</code> or <code>timeSeries</code> .
start	start date, character string, <code>Date</code> or <code>POSIXct</code> .
end	end date, character string, <code>Date</code> or <code>POSIXct</code> .
template	ts-boxable time series, an object of class <code>ts</code> , <code>xts</code> , <code>data.frame</code> , <code>data.table</code> , or <code>tibble</code> . If provided, <code>from</code> and <code>to</code> will be extracted from the object.
extend	logical. If true, the start and end values are allowed to extend the series (by adding NA values).

Details

All date and times, when entered as character strings, are processed by `anytime::anydate()` or `anytime::anytime()`. Thus a wide range of inputs are possible. See examples.

`start` and `end` can be specified relative to each other, using one of "sec", "min", "hour", "day", "week", "month", "quarter" or "year", or an abbreviation. If the series are of the same frequency, the shift can be specified in periods. See examples.

Value

a ts-boxable time series, with the same class as the input.

Examples

```

# use 'anytime' shortcuts
ts_span(mdeaths, start = "1979")      # shortcut for 1979-01-01
ts_span(mdeaths, start = "1979-4")   # shortcut for 1979-04-01
ts_span(mdeaths, start = "197904")   # shortcut for 1979-04-01

# it's fine to use an to date outside of series span
ts_span(mdeaths, end = "2001-01-01")

# use strings to set start or end relative to each other

ts_span(mdeaths, start = "-7 month") # last 7 months
ts_span(mdeaths, start = -7)         # last 7 periods
ts_span(mdeaths, start = -1)         # last single value
ts_span(mdeaths, end = "1e4 hours")  # first 10000 hours

ts_plot(
  ts_span(mdeaths, start = "-3 years"),
  title = "Three years ago",
  subtitle = "The last three years of available data"
)

ts_ggplot(
  ts_span(mdeaths, end = "28 weeks"),
  title = "28 weeks later",
  subtitle = "The first 28 weeks of available data"
) + theme_tsbox() + scale_color_tsbox()

# Limit span of 'discoveries' to the same span as 'AirPassengers'
ts_span(discoveries, template = AirPassengers)
ts_span(mdeaths, end = "19801201", extend = TRUE)

```

ts_summary

Time Series Properties

Description

Extract time series properties, such as the number of observations (`obs`), the time differences between observations (`obs`), the number of observations per year (`freq`), and the start time stamp (`start`) and the end time stamp (`end`) of the series.

Usage

```
ts_summary(x, spark = FALSE)
```

Arguments

x	ts-boxable time series, an object of class <code>ts</code> , <code>xts</code> , <code>zoo</code> , <code>data.frame</code> , <code>data.table</code> , <code>tbl</code> , <code>tbl_ts</code> , <code>tbl_time</code> , <code>tis</code> , <code>irts</code> or <code>timeSeries</code> .
spark	logical should an additional column with a spark-line added to the data frame (experimental, ASCII only on Windows.)

Value

`ts_summary` returns a `data.frame`. Individual column can be accessed through the `$` notation (see examples).

Examples

```
ts_summary(ts_c(mdeaths, austres))
ts_summary(ts_c(mdeaths, austres), spark = TRUE)
# Extracting specific properties
ts_summary(AirPassengers)$start
ts_summary(AirPassengers)$freq
ts_summary(AirPassengers)$obs
```

ts_trend

Loess Trend Estimation

Description

Trend estimation that uses `stats::loess()`.

Usage

```
ts_trend(x, ...)
```

Arguments

x	ts-boxable time series, an object of class <code>ts</code> , <code>xts</code> , <code>zoo</code> , <code>data.frame</code> , <code>data.table</code> , <code>tbl</code> , <code>tbl_ts</code> , <code>tbl_time</code> , <code>tis</code> , <code>irts</code> or <code>timeSeries</code> .
...	arguments, passed to <code>stats::loess()</code> : <ul style="list-style-type: none"> • degree degree of Loess smoothing • span smoothing parameter, if NULL, an automated search performed (see Details)

Examples

```
ts_plot(  
  `Raw series` = fdeaths,  
  `Loess trend` = ts_trend(fdeaths),  
  title = "Deaths from Lung Diseases",  
  subtitle = "per month"  
)
```

ts_ts

Convert Everything to Everything

Description

tsbox is built around a set of converters, which convert time series stored as `ts`, `xts`, `zoo`, `data.frame`, `data.table`, `tbl`, `tbl_ts`, `tbl_time`, `tis`, `irts` or `timeSeries` to each other.

Usage

`ts_data.frame(x)`

`ts_df(x)`

`ts_data.table(x)`

`ts_dt(x)`

`ts_tbl(x)`

`ts_tibbletime(x)`

`ts_timeSeries(x)`

`ts_tis(x)`

`ts_ts(x)`

`ts_irts(x)`

`ts_tsibble(x)`

`ts_tslist(x)`

`ts_xts(x)`

`ts_zoo(x)`

Arguments

`x` ts-boxable time series, an object of class `ts`, `xts`, `zoo`, `data.frame`, `data.table`, `tbl`, `tbl_ts`, `tbl_time`, `tis`, `irts` or `timeSeries`.

Details

In data frames, multiple time series will be stored in a 'long' format. `tsbox` detects a *value*, a *time* and zero to several *id* columns. Column detection is done in the following order:

1. Starting **on the right**, the first numeric or integer column is used as **value column**.
2. Using the remaining columns, and starting on the right again, the first Date, POSIXct, numeric or character column is used as **time column**. character strings are parsed by `anytime::anytime()`. The time stamp, `time`, indicates the beginning of a period.
3. **All remaining** columns are **id columns**. Each unique combination of id columns points to a time series.

Alternatively, the **time** column and the **value** column to be explicitly named as `time` and `value`. If explicit names are used, the column order will be ignored.

Whenever possible, `tsbox` relies on **heuristic time conversion**. When a monthly "ts" time series, e.g., `AirPassengers`, is converted to a data frame, each time stamp (of class "Date") is the first day of the month. In most circumstances, this reflects the actual meaning of the data stored in a "ts" object. Technically, of course, this is not correct: "ts" objects divide time in period of equal length, while in reality, February is shorter than January. Heuristic conversion is done for frequencies of 0.1 (decades), 1 (years), 4 (quarters) and 12 (month).

For other frequencies, e.g. 260, of `EuStockMarkets`, `tsbox` uses **exact time conversion**. The year is divided into 260 equally long units, and time stamp of a period will be a point in time (of class "POSIXct").

Value

ts-boxable time series of the desired class, an object of class `ts`, `xts`, `zoo`, `data.frame`, `data.table`, `tbl`, `tbl_ts`, `tbl_time`, `tis`, `irts` or `timeSeries`.

Examples

```
x.ts <- ts_c(mdeaths, fdeaths)
head(x.ts)
head(ts_df(x.ts))

suppressMessages(library(dplyr))
head(ts_tbl(x.ts))

suppressMessages(library(data.table))
head(ts_dt(x.ts))

suppressMessages(library(xts))
head(ts_xts(x.ts))
```

```
# heuristic time conversion
# 1 month: approx. 1/12 year
head(ts_df(AirPassengers))

# exact time conversion
# 1 trading day: exactly 1/260 year
head(ts_df(EuStockMarkets))

# multiple id
multi.id.df <- rbind(
  within(ts_df(ts_c(fdeaths, mdeaths)), type <- "level"),
  within(ts_pc(ts_df(ts_c(fdeaths, mdeaths))), type <- "pc")
)
head(ts_ts(multi.id.df))
ts_plot(multi.id.df)
```

Index

- * **package**
 - tsbox-package, 2
- %ts*(ts_arithmetic), 6
- %ts+(ts_arithmetic), 6
- %ts-(ts_arithmetic), 6
- %ts/(ts_arithmetic), 6

- anytime::anytime(), 28

- base::mean(), 13
- base::scale(), 23
- base::seq.Date(), 17

- colors_tsbox (ts_ggplot), 14
- copy_class, 3

- load_suggested (ts_), 5

- relevant_class, 4

- scale_color_tsbox (ts_ggplot), 14
- scale_color_tsbox(), 14
- scale_fill_tsbox (ts_ggplot), 14
- stats::loess(), 26
- stats::prcomp, 10

- theme_tsbox (ts_ggplot), 14
- theme_tsbox(), 14
- ts_, 5, 10
- ts_apply (ts_), 5
- ts_arithmetic, 6
- ts_bind, 7, 9
- ts_boxable, 8
- ts_c, 7, 8
- ts_chain (ts_bind), 7
- ts_compound (ts_index), 15
- ts_data.frame (ts_ts), 27
- ts_data.table (ts_ts), 27
- ts_default, 9
- ts_df (ts_ts), 27
- ts_diff (ts_pc), 19
- ts_diffy (ts_pc), 19
- ts_dt (ts_ts), 27
- ts_dts, 10
- ts_dygraphs (ts_examples), 10
- ts_dygraphs(), 14, 21
- ts_end (tsbox-deprecated), 4
- ts_examples, 5, 10
- ts_first_of_period, 12
- ts_forecast (ts_examples), 10
- ts_frequency, 12
- ts_ggplot, 14
- ts_ggplot(), 21
- ts_index, 15
- ts_irts (ts_ts), 27
- ts_lag, 16
- ts_long, 17
- ts_na_interpolation (ts_examples), 10
- ts_na_omit, 18
- ts_pc, 19
- ts_pca (ts_pc), 19
- ts_pcy (ts_pc), 19
- ts_pick, 20
- ts_plot, 21
- ts_plot(), 14
- ts_prcomp (ts_examples), 10
- ts_regular, 18, 22
- ts_save, 23
- ts_save(), 21
- ts_scale, 23
- ts_seas (ts_examples), 10
- ts_span, 24
- ts_start (tsbox-deprecated), 4
- ts_summary, 25
- ts_tbl (ts_ts), 27
- ts_tibbletime (ts_ts), 27
- ts_timeSeries (ts_ts), 27
- ts_tis (ts_ts), 27
- ts_trend, 26
- ts_ts, 27

`ts_tsibble` (`ts_ts`), [27](#)
`ts_tslist` (`ts_ts`), [27](#)
`ts_wide` (`ts_long`), [17](#)
`ts_xts` (`ts_ts`), [27](#)
`ts_zoo` (`ts_ts`), [27](#)
`tsbox` (`tsbox-package`), [2](#)
`tsbox-deprecated`, [4](#)
`tsbox-package`, [2](#)