Package ‘dsa’

June 21, 2021

Title Seasonal Adjustment of Daily Time Series
Version 1.0.12
Maintainer Daniel Ollech <daniel.ollech@bundesbank.de>
Description Seasonal- and calendar adjustment of time series
  with daily frequency using the DSA approach developed by Ollech,
License GPL-3
Depends R (>= 3.1.0)
Suggests knitr, rmarkdown, stR
VignetteBuilder knitr
Encoding UTF-8
LazyData true
RoxygenNote 7.1.1
Imports ggplot2, xts, zoo, R2HTML, grid, tools, tsoutliers,
  htmlwidgets, forecast, RJava, timeDate, dygraphs, gridExtra,
  reshape2, stats, seastests
NeedsCompilation no
Author Daniel Ollech [aut, cre]
Repository CRAN
Date/Publication 2021-06-21 05:20:02 UTC

R topics documented:

daily_data .......................................................... 2
daily_sim ........................................................... 3
del_names ........................................................... 4
Descaler ............................................................. 5
dsa ................................................................. 6
dsa_examples ..................................................... 9
freq_xts .......................................................... 10
daily_data

Exemplary time series

Description

Three time series that have been analysed by Ollech (2021) and their seasonally and calendar adjusted variants.

Usage
daily_data

Format

An xts data set containing 3 time series:

currency_circulation  Currency in circulation in Germany, in billion Euros, sum of small denominations: i.e. 5 Euro + 10 Euro + 20 Euro + 50 Euro. Series compiled by Deutsche Bundesbank
elec_consumption  Electricity consumption in Germany in GWh. Compiled by Bundesnetzagentur (German Federal Network Agency)
no2  Nitrogen dioxide (NO2) immissions averaged over all available measuring stations in Europe that are made available by the European Environment Agency (EEA) #'
currency_circulation_sa  Seasonally and calendar adjusted version using dsa of currency_circulation
elec_consumption_sa  Seasonally and calendar adjusted version using dsa of elec_consumption
no2_sa  Seasonally and calendar adjusted version using dsa of no2
Create a simple, exemplary, seasonal, daily time series

Description
Create a seasonal daily time series and its seasonal and non-seasonal components

Usage
daily_sim(
  n = 8,
  week_effect = 1,
  month_effect = 1,
  year_effect = 1,
  model = c(3, 1, 1),
  ar = c(-0.2, 0.5, 0.1),
  ma = -0.4,
  moving = T,
  week_cycles = 2,
  month_cycles = 3,
  year_cycles = 8
)

Arguments

  n     length of time series in years
  week_effect  increase size of seasonal factor for day-of-the-week
  month_effect increase size of seasonal factor for day-of-the-month
  year_effect  increase size of seasonal factor for day-of-the-year
  model  ARIMA model for trend and irregular component of series
  ar     coefficients for AR terms
  ma     coefficients for MA terms
  moving should seasonal factors be moving (=T) or constant (=F)
week_cycles  number of cycles per week
month_cycles  number of cycles per month
year_cycles   number of cycles per year

Details
The output is an xts time series containing the time series, the true seasonally adjusted series, the day-of-the-week seasonal component, the day-of-the-month seasonal component and the day-of-the-year seasonal component.

Author(s)
Daniel Ollech

Examples
time_series <- daily_sim(n=4, year_effect=3)
xtsplot(time_series[,1]) # Plot of the time series
xtsplot(time_series[,3:5]) # Plot of the seasonal factors

del_names Delete name of xts

Description
Delete name of xts

Usage
del_names(x)

Arguments
x           xts time series

Details
This function can be helpful if one xts is created to be equal to another xts and then changed afterwards. In these cases the new xts inherits the column name of the old xts.

Author(s)
Daniel Ollech
Examples

timeseries <- dsa::daily_sim()$original # timeseries inherits name from original
colnames(timeseries)
colnames(del_names(timeseries))
y <- del_names(timeseries)
colnames(merge(timeseries, y))

Descaler

Invert taking logs and differences of a time series

Description

For a series that has been logged and/or differenced, this function reverses these transformations.

Usage

Descaler(x, y = NA, Diff = 0, Sdiff = 0, Log = FALSE, Lag = NA)

Arguments

x time series
y time series used as benchmark
Diff number of differences to be taken
Sdiff number of seasonal differences to be taken
Log Should time series be logarithmised
Lag Lag for Sdiff can be specified

Details

The time series used as a benchmark (y) is necessary, if regular or seasonal differences have to be inversed, because the first values of this series are used to reconstruct the original values or benchmark the new series.

Author(s)

Daniel Ollech

Examples

a = ts(rnorm(100, 100, 10), start=c(2015,1), frequency=12)
b = Scaler(a, Diff=1, Log=TRUE)
Descaler(b, a, Diff=1, Log=TRUE)
**Description**

Seasonally adjust daily time series using the dsa approach

**Usage**

```r
dsa(
  series,
  span.start = NULL,
  model = NULL,
  Log = FALSE,
  automodel = "reduced",
  ic = "bic",
  include.constant = FALSE,
  fourier.number = 24,
  max_fourier = 30,
  s.window1 = 53,
  s.window2 = 53,
  s.window3 = 13,
  t.window1 = NULL,
  t.window2 = NULL,
  t.window3 = NULL,
  cval = 7,
  robust1 = TRUE,
  robust2 = TRUE,
  robust3 = TRUE,
  regressor = NULL,
  forecast_regressor = NULL,
  reg_create = NULL,
  reg_dummy = NULL,
  outlier = TRUE,
  outlier_types = c("AO", "LS", "TC"),
  delta = 0.7,
  model_span = NULL,
  feb29 = "sfac",
  trend_month = 3,
  outer3 = NULL,
  inner3 = NULL,
  h = 365,
  reiterate3 = NULL,
  scaler = 1e+07,
  mean_correction = TRUE,
  progress_bar = TRUE
)
```
Arguments

series  
Input time series in xts format

span.start  
Define when seasonal adjustment should begin

model  
ARIMA order of non-seasonal part

Log  
Boolean. Should multiplicate or additive model be used

automodel  
Set of models to be considered for automatic model detection. Either "full" or "reduced" set of fourier regressors included

ic  
Information criterion that is used for automodelling. One of "bic", "aic" or "aicc"

include.constant  
Should drift be allowed for model that includes differencing

fourier.number  
Number of trigometric regressors to model annual and monthly seasonality

max_fourier  
Maximum number of trigonometric regressors allowed if the number is selected automatically, i.e. fourier.number=NULL

s.window1  
STL parameter s.window for the day of the week effect

s.window2  
STL parameter s.window for the day of the month effect

s.window3  
STL parameter s.window for the day of the year effect

t.window1  
STL parameter t.window for the day of the week effect

t.window2  
STL parameter t.window for the day of the month effect

t.window3  
STL parameter t.window for the day of the year effect

cval  
Critical value for outlier adjustment

robust1  
Boolean. Should robust STL be used for the day of the week effect

robust2  
Boolean. Should robust STL be used for the day of the month effect

robust3  
Boolean. Should robust STL be used for the day of the year effect

regressor  
Pre-specified regressors

forecast_regressor  
Pre-specified regressors to be used for forecasting

reg_create  
Names of Holidays for which regressors will be created

reg_dummy  
If specified dummy variables of specified length are created and used as regressors

outlier  
Should an outlier adjustment be conducted?

outlier_types  
The following are possible: "LS", "TC", "AO", "IO"

delta  
The decay rate for TC outliers

model_span  
Last x years used for regARIMA modelling

feb29  
How should February 29th be derived: interpolation of adjusted series ("sa") or combined factor ("sfac")

trend_month  
Length of support period for trend estimation

outer3  
Number of iterations of outer loop in STL for day of the year effect

inner3  
Number of iterations of inner loop in STL for day of the year effect

h  
Forecast horizon in number of days
Number of total iterations of STL for the day of the year effect
for additive model, if max(abs(series)) > 1e5, scale series
Boolean. Should seasonal factors be standardised so that their mean (over all full cycles) is 0 for additive and 1 for multiplicative models
Boolean. Should a progress bar be displayed

Details
This function can be used to seasonally and calendar adjust daily time series and decomposing the series into a seasonally adjusted series, a day-of-the-week, a moving holiday, a day-of-the-month and a day-of-the-year component.

If mean_correction=TRUE (default), the seasonal and calendar factors are corrected, so that over all full years, the mean of the components is 0 in additive models. They will be close to 1 if a multiplicative decomposition (i.e. Log=TRUE) is used. Deviations from 1 may result, because the mean correction is applied to the components before inverting taking logs.

For long series, the ARIMA modelling and the outlier adjustment may take a long time. It may therefore be a good idea, to specify the ARIMA model used, e.g. model=c(3,1,0). If the series does not contain influential outliers, the outlier adjustment could be skipped by setting outlier=FALSE.

See vignette for further examples.

Value
dsa returns a daily object which contains the output of the seasonal adjustment of a daily time series.
output Contains the calendar and seasonally adjusted series, original series, implicit calendar and seasonal component, and Loess based trend as an xts object
fourier_terms The number of sine and cosine terms used to model the seasonal pattern in the RegARIMA model
reg RegARIMA results
info Basic information on transformation (Log/Level), differencing and forecast horizon
stl A list of length 3, containing the STL results of the day-of-week, day-of-the-month and day-of-the-year adjustment, respectively
outlier Result of the outlier adjustment
sa_result The original series and the intermediate adjustment results after the day-of-week adjustment (s1_adjusted), calendar adjustment (s1k1_adjusted), day-of-the-month adjustment (s1k1s2_adjusted), and the final adjusted series after the day-of-the-year adjustment (seas_adj) as an xts object
sa_result2 The original series only adjusted for single components as an xts object. Namely the original series itself (original), the original only adjusted for the day-of-the-week (s1_adjusted), calendar (k1_adjusted), day-of-the-month (s2_adjusted), and day-of-the-year (s3_adjusted)
sfac_result The seasonal and calendar components as an xts object. Namely, the day-of-the-week (s1_fac), calendar (cal_fac), day-of-the-month (s2_fac), and day-of-the-year component (s3_fac)
Author(s)
Daniel Ollech

References

Examples
x = daily_sim(n=4)$original  # series with length 4 years
res <- dsa(x, cval=7, model=c(3,1,0),fourier_number = 13)

---

Exemplary dsa outputs

Description
The dsa results for the three time series that have been analysed by Ollech (2021). Details on the specification can be found in the vignette.

Usage
dsa_examples

Format
A list containing the following three objects

- **cic_dsa** Results from a call to dsa() for the currency in circulation in Germany, in billion Euros, sum of small denominations: i.e. 5 Euro + 10 Euro + 20 Euro + 50 Euro. Series compiled by Deutsche Bundesbank.
- **elec_dsa** Results from a call to dsa() for the electricity consumption in Germany in GWh. Compiled by Bundesnetzagentur (German Federal Network Agency)
- **no2_dsa** Results from a call to dsa() for the nitrogen dioxide (NO2) immissions averaged over all available measuring stations in Europe that are made available by the European Environment Agency (EEA)

Author(s)
Daniel Ollech

Source
Own calculations, Deutsche Bundesbank, Bundesnetzagentur, EEA
References


freq_xts

Obtain the frequency of an xts time series

Description

Estimate the number of periods per year of an xts time series

Usage

def freq_xts(series)

Arguments

series        time series

Author(s)

Daniel Ollech

Examples

x <- xts::xts(rnorm(100), seq.Date(from=as.Date("2010-01-01"), by="months", length.out=100))
frequency(x)

get_original

Get Original Time Series

Description

Get the original time series from a seasonal adjustment object created by the dsa function. Can deviate from the input data as missings are filled up, usually using zoo::na.locf().

Usage

def get_original(daily.object, forecast = FALSE)

Arguments

daily.object    Output from dsa
forecast        Include forecast of component
get_sa

Author(s)
Daniel Ollech

See Also
get_sa, get_trend

Examples
set.seed(123)
x = daily_sim(n=4)$original # series with length 4 years
res <- dsa(x, cval=7, model=c(3,1,0),fourier_number = 13)
get_original(res)

get_sa

Get Seasonally Adjusted Series

Description
Get the calendar- and seasonally adjusted series from a seasonal adjustment object created by the dsa function

Usage
get_sa(daily.object, forecast = FALSE)

Arguments
daily.object Output from dsa
forecast Include forecast of component

Author(s)
Daniel Ollech

See Also
c_trend, get_original

Examples
set.seed(123)
x = daily_sim(n=4)$original # series with length 4 years
res <- dsa(x, cval=7, model=c(3,1,0),fourier_number = 13)
get_sa(res)
get_trend | Get Trend-Cycle

Description

Calculate the trend-cycle based on a seasonally adjusted series obtained from a seasonal adjustment object created by the dsa function.

Usage

get_trend(daily.object, trend_length = 93, forecast = FALSE)

Arguments

daily.object  Output from dsa

trend_length  Number of neighbouring points to use, in days

forecast  Include forecast of component

Details

If not odd the parameter trend_length is set to the next highest odd number.

Author(s)

Daniel Ollech

See Also

get_sa, get_original

Examples

set.seed(123)
x = daily_sim(n=4)$original # series with length 4 years
res <- dsa(x, cval=7, model=c(3,1,0),fourier_number = 13)
get_trend(res)
Data set for frequently used regressors

Description

Daily time series in xts format containing many regressors for holidays potentially used in the adjustment of daily time series

Usage

holidays

Format

An xts data set containing 131 regressors for the time span 1950 to 2075:

- **AllSaints** AllSaints, Nov 1
- **Ascension** Ascension
- **AscensionAft1Day** Captures the first day after Ascension
- **AscensionBef1Day** Captures the last day before Ascension
- **AssumptionOfMary** Assumption of Mary, Aug 15
- **Aug15ZZZ** Captures if Assumption of Mary, Aug 15, is a certain weekday (Monday to Sunday)
- **Base** Regressor made up of 0s, can be used to create other regressors
- **BoxingDay** Boxing Day, Dec 26
- **CarnivalMonday** Carnival Monday
- **ChristmasDay** Christmas Day, Dec 25
- **ChristmasEve** Christmas Eve, Dec 24
- **CorpusChristi** Corpus Christi
- **CorpusChristiAft1Day** Captures the first day after Corpus Christi
- **CorpusChristiBef1Day** Captures the last day before Corpus Christi
- **Dec24ZZZ** Captures if Dec 24 is a certain weekday (Monday to Sunday)
- **Dec25ZZZ** Captures if Dec 25 is a certain weekday (Monday to Sunday)
- **Dec26ZZZ** Captures if Dec 26 is a certain weekday (Monday to Sunday)
- **Dec31ZZZ** Captures if Dec 31 is a certain weekday (Monday to Sunday)
- **Dst** Daylight Saving Time, Spring=-1, Autumn=1
- **DstAutumn** Daylight Saving Time, Autumn=1
- **DstSpring** Daylight Saving Time, Spring=1
- **EasterMonday** Easter Monday
- **EasterMondayAft1Day** Captures the first day after Easter Monday
- **EasterPeriod** Captures all days from Holy Thursday to Easter Monday


**Easter Sunday**  Easter Sunday

**Epiphany**  Epiphany, Jan 6

**German Unity**  German Unity, Oct 3

**Good Friday**  Good Friday

**Holy Thursday**  Holy Thursday

**Holy Saturday**  Holy Saturday

**Jan1ZZZ**  Captures if Jan 1 is a certain weekday (Monday to Sunday)

**Jan6ZZZ**  Captures if Jan 1 is a certain weekday (Monday to Sunday)

**Labour Day**  Labour Day, May 1

**Labour Bridge**  Captures the bridge days created by May 1, i.e. if surrounding days are either a Monday or Friday

**Mardi Gras**  Mardi Gras

**May1ZZZ**  Captures if Labour Day, May 1, is a certain weekday (Monday to Sunday)

**New Year's Day**  New Year's Day, Jan 1

**New Year's Eve**  New Year's Eve, Dec 31

**Nov1ZZZ**  Captures if Nov 1 is a certain weekday (Monday to Sunday)

**Nov1 Bridge**  Captures the bridge days created by Nov 1, i.e. if surrounding days are either a Monday or Friday

**Oct3ZZZ**  Captures if German Unity, Oct 3, is a certain weekday (Monday to Sunday)

**Oct3 Bridge**  Captures the bridge days created by Nov 1, i.e. if surrounding days are either a Monday or Friday

**Reformation Day**  Reformation Day, Oct 31

**PostXmas Sat 10d**  Captures Saturdays in the period from Dec 27 to Jan 5

**Reformation Day 2017**  Reformation Day, Oct 31 2017 (National holiday that year)

**Xmas Period ZZZ**  Captures weekdays (Monday to Sunday) in the Christmas period from Dec 21 to Jan 5
**Author(s)**

Daniel Ollech

**Source**

Own calculations

---

**make_cal**

*Creating holiday regressor that increases linearly up to holiday and decreases afterwards*

---

**Description**

Creating holiday regressor that increases linearly up to holiday and decreases afterwards

**Usage**

```
make_cal(holidays = NULL, h = 365, original = NA, original2 = NA)
```

**Arguments**

- `holidays`  
  Holidays for which regressor will be created
- `h`  
  Forecast horizon
- `original`  
  xts time series which characteristics will be used
- `original2`  
  ts time series which characteristics will be used

**Details**

This function is used internally in dsa()

**Author(s)**

Daniel Ollech

**Examples**

```
a <- daily_sim(n=8)$original
## Not run: make_cal(holidays="Easter", original=a, original2=xts2ts(a, freq=365))
```
**Description**

Creating set of dummy variables for specified Holidays

**Usage**

```r
make_dummy(
    holidays = NULL,
    from = -5,
    to = 5,
    h = 365,
    original = NA,
    original2 = NA
)
```

**Arguments**

- **holidays**: holidays for which dummy variables will be created
- **from**: start of holiday regressor. Relative to specified holiday
- **to**: end of holiday regressor. Relative to specified holiday
- **h**: forecast horizon
- **original**: xts time series which characteristics will be used
- **original2**: ts time series which characteristics will be used

**Details**

This function is used internally in dsa()

**Author(s)**

Daniel Ollech

---

**Description**

This function uses the Holiday dates of the timeDate::timeDate package to create dummies on a specified holiday.
Usage

```r
make_holiday(dates = timeDate::Easter(2000:2030), shift = 0)
```

Arguments

dates: Holiday and period for which dummy shall be created
shift: shifting point in time for dummy

Details

With shift the user can specify for how many days before (negative value) or after (positive value) the holiday a dummy will be created.

Author(s)

Daniel Ollech

Examples

```r
make_holiday(dates=timeDate::Easter(2000:2030), shift=-1)
```

---

### multi_xts2ts

**Change multiple xts to a multivariate ts**

**Description**

Change multiple xts to a multivariate ts

**Usage**

```r
multi_xts2ts(x, short = FALSE)
```

**Arguments**

x: xts time series
short: Is series too short for xts2ts to work?

**Details**

If the ts are used for forecasting

**Author(s)**

Daniel Ollech
Examples

```r
x <- dsa::daily_sim()$original
y <- dsa::daily_sim()$original
multi_xts2ts(merge(x,y))
```

Description

This function creates HTML output in a specified folder for objects of class daily

Usage

```r
output(
daily_object,
path = getwd(),
short = FALSE,
SI = TRUE,
SI365.seed = 3,
spec = TRUE,
outlier = TRUE,
Factor = "auto",
every_day = TRUE,
seasonals = FALSE,
spectrum_linesize = 0.5,
seasonality_tests = TRUE,
progress_bar = TRUE
)
```

Arguments

daily_object  output of dsa() function
path          Path that HTML file is written to
short         Boolean. If true only short version of output is produced
SI            Including graphs of SI-ratios
SI365.seed   This seed influences which days of the year are shown as SI-ratios
spec          Boolean. Inclusion of spectral plots
outlier       Boolean. Inclusion of outlier plots
Factor        Scaling factor for series with large values
every_day     Boolean. Inclusion of table that summarizes daily results
seasonals     Boolean. Plots of seasonal factors as interactive instead of static graph
spectrum_linesize  Width of lines in spectrum
plot.daily

seasonality_tests
  Boolean. Inclusion of seasonality tests
progress_bar
  Should a progress bar be displayed?

Details

This function can be used to create plots and tables necessary for the analysis of seasonally and calendar adjusted daily time series. Uses the output of dsa() as an input.

Author(s)

Daniel Ollech

Examples

res <- dsa(daily_sim(4)$original, cval=7, model=c(3,1,0),fourier_number = 13)
## Not run: output(res)
Examples

```r
x <- daily_sim(3)$original
## Not run: res<- dsa(x, fourier_number = 24, outlier.types="AO", reg.create=NULL, model=c(3,1,0))
## Not run: plot(res, dy=FALSE)
```

plot_spectrum

Plot the periodogram of a daily time series

Description

Plot the periodogram of a daily time series

Usage

```r
plot_spectrum(
  x,
  xlog = FALSE,
  size = 1,
  color = "black",
  vline_color = "#6F87B2"
)
```

Arguments

- `x`: xts or ts, daily timeseries
- `xlog`: should x-axis be log transformed
- `size`: linesize
- `color`: color of line
- `vline_color`: color of vertical lines

Details

Plot uses ggplot2 and can be changed accordingly. The spectrum is build around the spec.pgram() function

Author(s)

Daniel Ollech

Examples

```r
x <- daily_sim(3)$original
plot_spectrum(x)
```
**print.daily**  
*Print daily time series*

### Description

Print output for objects of class "daily"

### Usage

```r
## S3 method for class 'daily'
print(x, ...)
```

### Arguments

- `x`: Result of `dsa()` that will be printed
- `...`: further arguments handed to `print()`

### Author(s)

Daniel Ollech

### Examples

```r
x <- daily_sim(3)$original
## Not run: res<- dsa(x, fourier_number = 24, outlier.types="AO", reg.create=NULL, model=c(3,1,0))
## Not run: print(res)
```

---

**Scaler**  
*Take logs and differences of a time series*

### Description

Logarithmise and / or difference a time series

### Usage

```r
Scaler(x, Diff = 0, Sdiff = 0, Log = FALSE)
```

### Arguments

- `x`: time series
- `Diff`: number of differences to be taken
- `Sdiff`: number of seasonal differences to be taken
- `Log`: Should time series be logarithmised
Details

Function is used in dsa to let the user decide whether logs and differences should be taken.

Author(s)

Daniel Ollech

Examples

```r
a = ts(rnorm(100, 100, 10), start=c(2015,1), frequency=12)
Scaler(a, Diff=1, Log=TRUE)
```

---

**to_weekly**  
Change a daily to a weekly differenced time series

Description

This function computes the weekly aggregates or differences (by default Friday to Friday) for any daily time series in the xts format.

Usage

```r
to_weekly(x, incl_forecast = T, forecast_length = 365, diff = T, dayofweek = 5)
```

Arguments

- `x` input series
- `incl_forecast` whether the series contains a forecast that shall be omitted
- `forecast_length` length of forecast
- `diff` should series be differenced
- `dayofweek` which day of the week (friday=5)

Author(s)

Daniel Ollech

Examples

```r
to_weekly(xts::xts(rnorm(365, 10,1), seq.Date(as.Date("2010-01-01"), length.out=365, by="days")))
```
ts2xts

Change ts to xts

Description
Change the format of a time series from ts to xts. Has been optimised for the use in dsa(), i.e. for daily time series.

Usage
ts2xts(x_ts)

Arguments

x_ts ts series to be changed to xts

Details
This function is used internally in dsa(). Does not create values for the 29th of February.

Author(s)
Daniel Ollech

Examples
ts2xts(stats::ts(rnorm(1000, 10,1), start=c(2001,1), freq=365))

ts_sum Add time series

Description
Sequentially add a set of time series

Usage
ts_sum(...)
xts2ts

Description
Change the format of a time series from xts to ts. Has been optimised for the use in dsa(), i.e. for daily time series.

Usage
```r
xts2ts(series, freq = NULL)
```

Arguments
- `series`: xts series to be changed to ts
- `freq`: frequency of ts series

Details
This function is used internally in dsa(). Does not create values for the 29th of February.

Author(s)
Daniel Ollech

Examples
```r
xts2ts(xts::xts(rnorm(1095, 10, 1), seq.Date(as.Date("2010-01-01"), length.out=1095, by="days")))
```
xtsplot

Create a plot for xts series

Description

Creates a plot using an xts series

Usage

xtsplot(
  xts,
  transform = "none",
  type = "line",
  years = NA,
  scale = 1,
  names = NA,
  color = NA,
  main = "",
  legend = NA,
  textsize = 1,
  textsize_x = NA,
  textsize_y = NA,
  textsize_legend = NA,
  textsize_title = NA,
  linesize = 1.1,
  WeekOfYear = F,
  date_breaks = NA,
  date_labels = NA,
  submain = NULL
)

Arguments

xts        one or many series
transform   one of "none", "diff", "change" (can be abbreviated)
type       either "bar", "bar2" or "line"
years      number of years to include
scale      by what factor should data be scaled.
names      change names of series
color      color of the series
main       title of the plot
legend     alignment of legend. "horizontal" or "vertical"
textsize   scale the size of all the text
textsize_x scale size of x-axis labels
textsize_y  scale size of y-axis labels

textsize_legend  scale size of legend text

textsize_title  scale size of title

linesize  scale the size of the lines

WeekOfYear  should x axis be week of year

date_breaks  distance between labels (see examples)

date_labels  format of the date label for x-axis

submain  subtitle of the plot

Details

This function uses the ggplot2 package. The difference between type="bar" and type="bar2" is that the former produces barcharts with bars of the second series in front of the bars of the first series (and accordingly for more than two series), while "bar2" creates side-by-side barcharts. If a scale is supplied, the data will be divided by this number.

Author(s)

Daniel Ollech

Examples

```r
x <- xts::xts(rnorm(100), seq.Date(as.Date("2010-01-01"), length.out=100, by="months"))
y <- xts::xts(runif(100), seq.Date(as.Date("2010-01-01"), length.out=100, by="months"))
xtsplot(y, transform="diff", type="bar")
xtsplot(y, transform="diff", type="bar", date_breaks="24 months")
xtsplot(merge(x,y), names=c("Gaussian", "Uniform"), main="Simulated series")
```
Index

* datasets
  daily_data, 2
  dsa_examples, 9
  holidays, 13

  daily_data, 2
  daily_sim, 3
  del_names, 4
  Descaler, 5
  dsa, 6
  dsa_examples, 9

  freq_xts, 10
  get_original, 10
  get_sa, 11
  get_trend, 12
  holidays, 13

  make_cal, 15
  make_dummy, 16
  make_holiday, 16
  multi_xts2ts, 17

  output, 18

  plot.daily, 19
  plot_spectrum, 20
  print.daily, 21

  Scaler, 21

  to_weekly, 22
  ts2xts, 23
  ts_sum, 23

  xts2ts, 24
  xtsplot, 25