Package 'dsa'

April 4, 2019

Title Seasonal Adjustment of Daily Time Series

Version 0.70.3

Description Seasonal- and calendar adjustment of time series with daily frequency using the DSA approach developed by Ollech, Daniel (2018): Seasonal adjustment of daily time series. Bundesbank Discussion Paper 41/2018.

Depends R (>= 3.1.0)

License GPL-3

Encoding UTF-8

LazyData true

Maintainer Daniel Ollech <daniel.ollech@bundesbank.de>

Imports ggplot2, xts, zoo, R2HTML, xtable, grid, tools, tsoutliers, htmlwidgets, forecast, rJava, timeDate, dygraphs, extrafont, gridExtra, reshape2, stats

RoxygenNote 6.0.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Daniel Ollech [aut, cre]

Repository CRAN

Date/Publication 2019-04-04 09:20:06 UTC

R topics documented:

`Add` *Adding xts together*

Description

Adding xts together while treating NAs as zeros.

Usage

```
Add(x, y, ...)
```

Arguments

`x, y`	Input time series
`...`	further time series to be added

Details

Sometimes, if a xts contains missing values, the behaviour of the usual addition-function is not ideal, at least for the purposes of seasonal adjustment of daily time series. This function changes the behaviour.

Author(s)

Daniel Ollech

Examples

```
series1 <- xts::xts(rnorm(5, 5, 5), seq.Date(from=as.Date("2010-01-01"), length.out=5, by="days"))
series2 <- xts::xts(c(3,4,NA, 6,7), seq.Date(from=as.Date("2010-01-01"), length.out=5, by="days"))
Add(series1, series2)
# Compare this to:
series1 + series2
```

`daily_sim` *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], font="sans") # Plot of the time series
xtsplot(time_series[,3:5], font="sans") # Plot of the seasonal factors
```

`day_split` *Forecasts the days of the week*

Description

This function splits a time series into the days of the week and forecasts them using the X-11 heuristic or ETS.

Usage

```
day_split(series = NULL, use = "heur", h = 365)
```

Arguments

`series`	Input time series
`use`	Which method to use. "heur" or "ets".
`h`	Length of the Forecast

Details

This function is used internally in dsa()

Author(s)

Daniel Ollech

Examples

```
day_split(series=ts(rnorm(730, 100,1), start=c(2001,1), frequency=7), use="ets", h=365)
```

`Descaler` *Invert taking logs and differences of a time series*

Description

For a series that has been logged and/or differences, 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 is 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)
```

df2HTML *Output a dataframe to HTML*

Description

Output a dataframe to a HTML file.

Usage

```
df2HTML(dataframe, file)
```

Arguments

dataframe	data to be parsed to HTML
file	path to save to

Details

Function used to create HTML for the results of the seasonal adjustment. But can basically be used to create HTML output for any data.frame.

Author(s)

Daniel Ollech

Examples

```
a=data.frame(lapply(1:4, function(x) round(rnorm(10),2)))
colnames(a) = paste0("x", 1:4)
df2HTML(a, "out.html")
```

dom_dummy *Dummy for the Day of the Month*

Description

Creates dummies for each chosen day of the week.

Usage

```
dom_dummy(day = "01", start = "2010/1/1", end = "2015/01/01",
  delete29 = T)
```

Arguments

day	Day of the Month for which dummy is created
start	Startdate
end	Enddate
delete29	Should the 29th of February be deleted?

Details

This function is used in dsa() to create day of the month dummies.

Author(s)

Daniel Ollech

Examples

```
plot(dom_dummy())
```

---

dow_dummy *Dummy for the Day of the Week*

Description

Creates dummies for each chosen day of the week.

Usage

```
dow_dummy(day = "1", start = "2010/1/1", end = "2015/01/01")
```

Arguments

day	Day of the Week for which dummy is created
start	Startdate
end	Enddate

Details

This function is used in dsa() to create day of the week dummies.

Author(s)

Daniel Ollech

Examples

```
plot(dow_dummy())
```

doy_dummy *Dummy for the Day of the Year*

Description

Creates dummies for each chosen day of the year

Usage

```
doy_dummy(day = "1", start = "2010/1/1", end = "2015/01/01")
```

Arguments

day	Day of the year for which dummy is created
start	Startdate
end	Enddate

Details

This function is used in dsa() to create day of the year dummies.

Author(s)

Daniel Ollech

Examples

```
plot(doy_dummy())
```

drop31 *Cutting spurious days from a series with 31 days a month.*

Description

Changing a series with 31 days a month to a series with the regular number of observations per month.

Usage

```
drop31(x_ts, new_start = 335, new_end = 55)
```

Arguments

x_ts	Input time series in the ts format
new_start	New start date as day of the year. Value from 1 to 366.
new_end	New end date as day of the year. Value from 1 to 366.

Details

This function is used internally in dsa()

Author(s)

Daniel Ollech

Examples

```
x <- xts::xts(rnorm(1095, 100,1), seq.Date(as.Date("2009-01-01"), length.out=1095, by="days"))
a31 <- fill31(x)
a <- drop31(a31, 1, 365)
```

---

dsa *Seasonally Adjust Daily Time Series*

Description

Seasonally adjust daily time series using the dsa approach

Usage

```
dsa(series, span.start = NA, model = NULL, Log = FALSE, Diff = 0,
  automodel = "reduced", ic = "bic", fourier_number = NA,
  s.window1 = 151, s.window2 = 51, s.window3 = 15, 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.types = c("AO", "LS", "TC"), modelspan = NULL, trend_month = 3,
  outer3 = NULL, inner3 = NULL, h = 365, reiterate3 = NULL,
  scaler = 1e+07, progressBar = 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?
Diff	Number of differences taken before STL is run.
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"
fourier_number	Number of trigometric regressors to model annual and monthly seasonality
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.types	The following are possible: "LS", "TC", "AO", "IO"
modelspan	Last x years used for regARIMA modelling.
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
reiterate3	Number of total iterations of STL for the day of the year effect
scaler	for additive model, if max(abs(series)) > 1e5, scale series
progressBar	Should a progress bar be displayed?

Details

This function can be used to seasonally and calendar adjust daily time series using multiplicative model.

Author(s)

Daniel Ollech

References

Ollech, Daniel (2018). Seasonal adjustment of daily time series. Bundesbank Discussion Paper 41/2018.

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, reg.create=NULL)
```

fill31 *Extending a daily time series to having 31 days each month.*

Description

This function extends a time series to have 31 days.

Usage

```
fill31(x_ts, fill = "locf", to_ts = TRUE)
```

Arguments

x_ts	Time series that will be extended to 31 days each month.
fill	Method that is used to fill up time series. "locf": last observation carried forward, "lin": linear interpolation, "spline": spline interpolation.
to_ts	Boolean. Determines format of the output series. Either ts or xts.

Details

This function is used internally in dsa()

Author(s)

Daniel Ollech

Examples

```
x<-xts::xts(rnorm(1095, 100,1), seq.Date(as.Date("2009-01-01"), length.out=1095, by="days"))
a31 <- fill31(x)
a <- drop31(a31, 1, 365)
```

fill_up *Fill up NAs*

Description

Copy values from series to another to fill up missing values

Usage

```
fill_up(fill_up_series = NA, use_series = NA)
```

Arguments

fill_up_series	Series that has missing values
use_series	Series that is used fo fill up missing values

Details

This function is used internally in dsa()

Author(s)

Daniel Ollech

Examples

```
a <- b <- daily_sim(n=3)$original
a[c(355,376)] <- NA
a_new <- fill_up(a, b)
all(b==a_new)
```

---

freq_xts *Obtain the frequency of an xts time series*

Description

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

Usage

```
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

```
get_original(daily.object, forecast = FALSE)
```

Arguments

daily.object	Output from dsa
forecast	Include forecast of component

Author(s)

Daniel Ollech

See Also

get_sa, get_trend

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, reg.create=NULL)
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

get_trend, get_original

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, reg.create=NULL)
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

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

Holiday *Creating Holiday dummy*

Description

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

Usage

```
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

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

makeCal *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

```
makeCal(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: makeCal(holidays="Easter", original=a, original2=xts2ts(a, freq=365))
```

---

makeDummy *Creating set of dummy variables for specified Holidays*

Description

Creating set of dummy variables for specified Holidays

Usage

```
makeDummy(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

`outlier` *Outlier adjustment of daily time series*

Description

Outlier adjust any daily time series with an algorithm similar to that used in TRAMO. This function draws heavily from the tsoutliers package by Javier López-de-Lacalle.

Usage

```
outlier(series, model, cval = 7, types = c("AO", "LS", "TC"),
  maxit.oloop = 1, maxit.iloop = 2, maxit.endloop = 1000,
  holidays = NULL, number.fourier = 13)
```

Arguments

`series`	Input time series
`model`	ARIMA model used
`cval`	Critical Value for outlier Detection
`types`	Types of Outliers included. "AO", "LS", "TC" and "IO" permitted.
`maxit.oloop`	Maximum iterations of the outer loop
`maxit.iloop`	Maximum iterations of the inner loop
`maxit.endloop`	Maximum iterations of the end loop.
`holidays`	Holiday regressors used in regARIMA
`number.fourier`	Number of trigonometric regressors used to model seasonality

Details

This function is used internally in dsa()

References

López-de-Lacalle, Javier (2017). R package tsoutliers.

Examples

```
set.seed(356)
x <- arima.sim(list(order = c(1,1,0), ar = 0.7), n = 365*4)
timeseries <- ts(x, freq=365, start=c(2001,1))
shocks <- rbinom(length(timeseries), 1, 0.002) * 1.5 * timeseries
timeseries <- timeseries + shocks
modelfit <- arima(timeseries, order = c(1,1,0))
out <- outlier(timeseries, model=modelfit, cval=8)
ts.plot(timeseries, out$series_adj, col=c("red", "black"))
Names = c("Original Series", "Outlier Adjusted")
legend(2004.2, 140, Names, col=c("red", "black"), lty=1, bty="n", cex=0.75)
```

output *Creating Output for dsa*

Description

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

Usage

```
output(daily.object, path = getwd(), short = FALSE, SI = TRUE,
  SI365.seed = 3, spec = TRUE, outlier = TRUE, Factor = "auto",
  everyDay = TRUE, seasonals = FALSE, progressBar = 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
everyDay	Boolean. Inclusion of table that summarizes daily results
seasonals	Boolean. Plots of seasonal factors as interactive instead of static graph.
progressBar	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, reg.create=NULL)
## Not run: output(res)
```

`plot.daily` *Plot daily time series*

Description

Plotting output for objects of class "daily"

Usage

```
## S3 method for class 'daily'
plot(x, dy = TRUE, trend = FALSE, ...)
```

Arguments

`x`	Result of dsa() that will be plotted
`dy`	should dygraphs be used for plotting
`trend`	Boolean. Inclusion of a trend estimate.
`...`	Other plot parameters (only if dy=FALSE)

Details

The original series is plotted in black, the seasonally adjusted series is colored in red, and if trend=T, a blue trend line is added.

Author(s)

Daniel Ollech

Examples

```
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)
```

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

Description

Logarithmise and / or difference a time series

Usage

```
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

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

Time *Creating Several Holiday dummy*

Description

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

Usage

```
Time(from = -10, to = 10, dates = timeDate::Easter(2000:2030))
```

Arguments

from	Relative to Holiday, starting date
to	Relative to Holiday, end date
dates	Which Holidays shall be used

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

```
## Not run: output(Time(from=5, to=10, dates=timeDate::Easter(2000:2030))
```

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

```
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

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

ts.sum *Add time series*

Description

Sequentially add a set of time series

Usage

```
ts.sum(...)
```

Arguments

...	list of ts time series that are added together

Details

This function is used internally in dsa()

Author(s)

Daniel Ollech

Examples

```
ts.sum(list(ts(rnorm(100,10,1)), ts(rnorm(100,10,1)), ts(rnorm(100,10,1))))
```

`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))
```

xts2ts *Change xts to ts*

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

```
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

```
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, font = NA)
```

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
font	font to be used

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

```
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(x, font="sans")
xtsplot(y, transform="diff", type="bar", font="sans")
xtsplot(y, font="sans")
xtsplot(y, transform="diff", type="bar", date_breaks="24 months", font="sans")
xtsplot(merge(x,y), names=c("Gaussian", "Uniform"), main="Simulated series", font="sans")
```

Index