Package 'blsplotR'

June 7, 2022

Title Plots for Seasonal Adjustment Analysts

Version 1.1

Description Generates several types of time seris plots useful for seasonal adjustment analysis. These routines rely heavily on the seasonal package to extract series and components from the ' seasonal adjustments generated by the US Census Bureau's X-13ARIMA-SEATS software, and can be generated from a single seas object or a list of seas objects. Types of plots include line plots, ratio plots, forecast plots, forecast error diagnostic plots, spectral plots, seasonal factor plots, seasonal adjustment component plots. Users can add grid lines and shade recession regions in selected plots.

License MIT + file LICENSE

Encoding UTF-8

LazyData TRUE

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

Imports colortools, R.devices, sautilities, seasonal, tis, tools

Depends R (>= 2.10)

R topics documented:

`convert_date_to_tis` *Convert ts dates to tis format*

Description

Convert dates used for monthly (or quarterly) `ts` series to `tis` formats

Usage

```
convert_date_to_tis(
  this_date,
  this_freq = 12,
  is_start = TRUE,
  return_tis = FALSE
)
```

Arguments

`this_date`	numeric scalar or vector; `ts` date to be converted
`this_freq`	numeric scalar; frequency of `ts` time series. Default is 12 (monthly).
`is_start`	logical scalar; is date assumed to be the beginning of the month? Default is TRUE; if FALSE, date is assumed to be at the end of the month.
`return_tis`	logical scalar; If true, return as `tis` object; otherwise return as integer Default is FALSE; if FALSE, date is assumed to be at the end of the month.

Value

a `tis` index value that is the equivalent of the codets date

Examples

```
end_this_month <- as.numeric(substr(Sys.Date(),1,4)) +
   (as.numeric(substr(Sys.Date(),6,7)) - 1) / 12
end_this_month_tis <- convert_date_to_tis(end_this_month, this_freq = 12,
                                          is_start = FALSE, return_tis = TRUE)
```

`draw_recession` *Draw NBER recessions*

Description

Draws shaded areas in plots corresponding to NBER recessions

Usage

```
draw_recession(
  this_col_recess = NULL,
  this_density = 50,
  this_border = NA,
  this_add_recess_start = NULL,
  this_sub_recess = TRUE,
  this_sub_line = 3,
  this_sub_cex = 0.75
)
```

Arguments

`this_col_recess`	Character string; color used for shading recession periods. Default is `'lightgrey'`.
`this_density`	the density of shading lines, in lines per inch. The default value is 50. A zero value of density means no shading lines whereas negative values (and NA) suppress shading (and so allow color filling).
`this_border`	Integer scalar; thickness of border around region. Default is `NA`, meaning the border is not generated.
`this_add_recess_start`	numeric scalar; Starting date for an additional recession period at the end of the series. Default is not to add recession dates.
`this_sub_recess`	Logical scalar; indicates if x-axis label for recession is produced for this plot. Default is x-axis label is produced
`this_sub_line`	Integer scalar; position of subtitle of plot. Default is `3`.
`this_sub_cex`	Numeric scalar; scaling for subtitle of plot. Default is `0.75`.

Value

Shades recession dates in plots

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)', x11='')
plot_table(m_air, 'a1', 'AirPassengers', do_grid = TRUE, draw_recess = FALSE,
          use_ratio = FALSE, add_sub_title = TRUE, this_col = 'green')
start_pandemic_recession <- 2020 + 1/12 # start of pandemic recession February 2020
draw_recession(this_col_recess = 'lightblue', this_border = 1,
               this_add_recess_start = start_pandemic_recession,
               this_sub_line = 1.5, this_sub_cex = 0.9)
mtext('NBER Recessions in Blue',1,3,cex=0.75)
```

flag_peak *Flag visual significant peaks in spectra*

Description

Determine positions of visual significant peaks in spectra

Usage

```
flag_peak(m_seas, spec_type, spec_freq_code, max_freq)
```

Arguments

m_seas	seas object generated from a call of seas on a single time series
spec_type	Character string; type of spectrum. Possible values are 'ori','irr','rsd','sa'.
spec_freq_code	Character string; type of frequency being tested. Possible values are 's' or 't'.
max_freq	Numeric string; maximum number of frequencies to test.

Value

If visually significant peaks found, a numeric vector of the position of the peak frequecies. If no peaks found, 0.

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)', x11='')
this_flagged_peak_seas <- flag_peak(m_air,'ori','s',5)
this_flagged_peak_td <- flag_peak(m_air,'ori','t',2)
```

get_recession_dates *Get NBER recession dates*

Description

Generate starting and ending dates for NBER recessions between two monthly (or quarterly) dates

Usage

```
get_recession_dates(
  start_recess = NULL,
  end_recess = NULL,
  add_recess_start = NULL,
  this_freq = 12
)
```

Arguments

start_recess	numeric scalar; Starting date for plot. Default is first recession starting date.
end_recess	numeric scalar; Ending date for plot. Default is last recession ending date.
add_recess_start	numeric scalar; Starting date for an additional recession period at the end of the series. Default is not to add recession dates.
this_freq	numeric scalar; frequency of ts time series. Default is 12 (monthly).

Value

Starting and ending dates for NBER recessions within a span of data

Examples

```
plot_limits <- par('usr')
start_pandemic_recession <- 2020 + 1/12 # start of pandemic recession February 2020
thisRec <-
   get_recession_dates(start_recess = plot_limits[1], end_recess = plot_limits[2],
                       add_recess_start = start_pandemic_recession)
```

---

plot_all *Generate all diagnostic plots this*

Description

Generates a series of diagnostic plots from a single seas object and store the results in a separate file

Usage

```
plot_all(
  m_seas = NULL,
  series_name = NULL,
  file_base = NULL,
  this_dir = NULL,
  this_start = NULL,
  split_plots = FALSE,
  plot_type = "pdf",
  this_grid = FALSE,
  this_draw_recess = FALSE,
  this_add_recess_start = NULL,
  this_recess_col = NULL,
  this_recess_sub = TRUE,
  this_otl = FALSE,
  this_si = FALSE,
  this_mean_line = TRUE,
  this_specturm_axis = TRUE,
  this_ratio = FALSE,
  this_otl_cex = 0.5,
  this_add_identified_otl = FALSE,
```

```
  this_sub_title = FALSE,
  col_ori = "grey",
  col_sa = "green",
  col_one = "blue",
  col_factor = "green",
  col_fcst = c("grey", "green", "red"),
  col_otl = c("red", "blue", "green", "brown", "grey", "yellow"),
  col_sf = c("darkgreen", "darkblue", "grey"),
  col_spec = c("blue", "green", "grey", "brown", "red", "orange")
)
```

Arguments

m_seas	seas object generated from a call of seas on a single time series
series_name	Character scalar; name of the time series used in m_seas.
file_base	Character scalar; base file name for the graphics file generated. Default base file name is 'BLSplot'.
this_dir	Character scalar; directory where the graphics file generated. Default is the current working directory.
this_start	Integer vector of length 2; Starting date for plot. Default is starting date for the time series.
split_plots	Logical scalar; indicates if plots will be split into different files. Default is combine the plots into one file.
plot_type	Character scalar; Type of graphics file generated. Default is 'pdf'.
this_grid	Logical scalar; indicates if certain plots will have grid lines. Default is no grid lines.
this_draw_recess	Logical scalar; indicates if certain plots will have shaded areas for NBER recession dates. Default is no recession shading.
this_add_recess_start	numeric scalar; Starting date for an additional recession period at the end of the series. Default is not to add recession dates.
this_recess_col	Character string; color used for shading of recession region. Default is 'lightgrey'.
this_recess_sub	Logical scalar; indicates if x-axis label for recession is produced for this plot. Default is x-axis label is produced
this_otl	Logical scalar; indicates if lines for identified outliers are included in series plots. Default is not including lines for identified outliers.
this_si	Logical scalar; indicates if seasonal factor plots will include SI ratios for X-11 seasonal adjustments. Default is not including SI ratios.
this_mean_line	Logical scalar; indicates if seasonal factor plots will include lines for seasonal means. Default includes lines for seasonal means.
this_specturm_axis	Logical scalar; indicates if x-axis of spectral plot will be frequency by month rather than the actual frequencies. Default sets x-axis to frequency by month.
this_ratio	Logical scalar; indicates if plots of seasonal factors, irregular, and residuals are done as ratio plots. Default has these plots as time series line plots.
this_otl_cex	Numeric scalar; sets the cex plotting parameter. Default sets cex = 0.5.

this_add_identified_otl
: Logical scalar; indicates if outlier plots will include identified outliers. Default is not including identified outliers.

this_sub_title
: Logical scalar; indicates if certain plots will include subtitles denoting what series are plotted. Default is not including subheaders.

col_ori
: Character scalar; color used for the original series. Default is grey.

col_sa
: Character scalar; color used for the seasonally adjusted series. Default is green.

col_one
: Character scalar; color used for individual series. Default is blue.

col_factor
: Character scalar; color used for factor plots. Default is green.

col_fcst
: Array of character strings; color used for original series, forecast, and forecast bounds. Default is c('grey','green','red').

col_otl
: Character aray of length 6; color used for different outliers, with the order being 'ao','ls','tc','so','rp','tls'. Default is c('red','blue','green','brown','grey','ye

col_sf
: Character aray of length 3; color used for special seasonal plots, with the order being seasonal factors, SI ratio, seasonal mean. Default is c('darkgreen','darkblue','grey').

col_spec
: Character array of length 6; color used in specturm plots, in the order of spectrum of ori, spectrum of SA, line for seasonal frequency, line for TD frequency, star for visually significant seasonal frequency, star for visually significant TD frequency. Default is c('blue','green','grey','brown','red','orange').

Value

Graphics file with number of diagnostic plots routinely used at BLS.

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)', x11='')
start_pandemic_recession <- 2020 + 1/12
plot_all(m_air, series_name = 'AirPassengers', file_base = 'AirPass',
         split_plots = TRUE, plot_type = 'png', this_grid = TRUE,
         this_draw_recess = TRUE, this_add_recess_start = start_pandemic_recession,
         this_ratio = TRUE, this_add_identified_otl = TRUE,
         col_sa = 'darkgreen', col_one = 'darkblue')
```

---

plot_cpgram_resid *Generate cumulative periodogram of the regARIMA residuals*

Description

Generates a plot of the cumulative periodogram of the regARIMA residuals

Usage

```
plot_cpgram_resid(m_seas = NULL, main_title = "Cumulative periodogram")
```

Arguments

m_seas
: seas object generated from a call of seas on a single time series

main_title
: Character string; main title of plot. Default is 'Cumulative periodogram'.

Value

Generates a plot of the Cumulative periodogram of the regARIMA residuals. Diagnostic information is included in subheaders.

Examples

```
m_air <- seasonal::seas(AirPassengers, transform.function= 'log', arima.model = '(0 1 1)(0 1 1)')
plot_cpgram_resid(m_air, main_title = 'Cumulative periodogram for Airline Passenger Residuals')
```

`plot_double_spectrum` *Generate double spectrum plot of the original and seasonally adjusted series.*

Description

Generate plot of spectrum of original series and seasonally adjusted series on same axis.

Usage

```
plot_double_spectrum(
  m_seas = NULL,
  xaxis_bls = TRUE,
  main_title = "AR Spectrum",
  series_name = NULL,
  this_col = c("blue", "green", "grey", "brown", "red", "orange")
)
```

Arguments

`m_seas`	`seas` object generated from a call of `seas` on a single time series
`xaxis_bls`	Logical scalar; indicates if x-axis of spectral plot will be frequency by month rather than the actual frequencies. Default sets x-axis to frequency by month.
`main_title`	Character string; main title of plot. Default is `'AR Spectrum'`.
`series_name`	Character scalar; name of the time series used in m.
`this_col`	Character array of length 6; color used in specturm plots, in the order of spectrum of ori, spectrum of SA, line for seasonal frequency, line for TD frequency, star for visually significant seasonal frequency, star for visually significant TD frequency. Default is `c('blue','green','grey','brown','red','orange')`.

Value

Generate plot of spectrum of original series and seasonally adjusted series on same axis.

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)', x11='')
plot_double_spectrum(m_air, series_name = 'AirPassengers',
         this_col = c('blue', 'green', 'darkblue', 'darkgreen', 'red', 'orange'))
```

plot_fcst *Forecast plot*

Description

Generates regARIMA forecasts with confidence bounds

Usage

```
plot_fcst(
  m_seas = NULL,
  main_title = "ARIMA forecasts",
  do_grid = FALSE,
  do_sub = TRUE,
  length_ori = 2,
  this_col = c("darkgrey", "blue", "darkgreen")
)
```

Arguments

m_seas	seas object generated from a call of seas on a single time series
main_title	Character string; main title of plot. Default is 'ARIMA Residuals'.
do_grid	Logical scalar; indicates if certain plots will have grid lines. Default is no grid lines.
do_sub	Logical scalar; indicates if subtitle is generated. Default is to generate the subtitle.
length_ori	Integer scalar; number of years of the original series to show with forecasts. Default is 2 years.
this_col	Array of character strings; color used for original series, forecast, and forecast bounds. Default is c("darkgrey","blue","darkgreen").

Value

Generates a plot of the regARIMA forecasts with confidence bounds.

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)')
plot_fcst(m_air, main_title = 'Forecasts for Airline Passengers', do_grid = TRUE)
```

plot_fcst_history *Generate forecast history plot*

Description

Generate forecast history plot, which compares the sum of squared forecast errors for two models.

Usage

```
plot_fcst_history(
  seas_mdl1 = NULL,
  seas_mdl2 = NULL,
  start_hist = NULL,
  main_title = "Differences in the Sum of Squared Forecast Errors",
  name_mdl1 = "Model 1",
  name_mdl2 = "Model 2",
  this_col = c("blue", "darkgreen")
)
```

Arguments

seas_mdl1	seas object generated from a call of seas on a single time series for the first model
seas_mdl2	seas object generated from a call of seas on a single time series for the second model
start_hist	integer scalar; starting date for the history analysis. Could be an array of length 2; will be converted to a scalar
main_title	Character string; main title of plot. Default is 'Differences in the Sum of Squared Forecast Errors'.
name_mdl1	Character string; Description of first model for use in the subtitle. Default is 'Model 1'.
name_mdl2	Character string; Description of second model for use in the subtitle. Default is 'Model 2'.
this_col	Character array of length 2; color used for each forecast lag. Default is c('blue','darkgreen').

Value

Generate forecast history plot. Can be more than one series. If series not specified, print out error message and return NULL.

Examples

```
air_m <- seasonal::seas(AirPassengers, x11='', slidingspans = '', transform.function = 'log',
                arima.model = '(0 1 1)(0 1 1)', regression.aictest = NULL, outlier = NULL,
                       forecast.maxlead=36, check.print = c( 'pacf', 'pacfplot' ),
                       history.fstep = c(1, 12), history.estimates = 'fcst',
                       history.save = 'fcsterrors')
air_m2 <- seasonal::seas(AirPassengers, x11='', slidingspans = '', transform.function = 'log',
                       arima.model = '(0 1 1)(0 1 1)',
                       forecast.maxlead=36, check.print = c( 'pacf', 'pacfplot' ),
```

```
                        history.fstep = c(1, 12), history.estimates = 'fcst',
                        history.save = 'fcsterrors')
plot_fcst_history(air_m, air_m2, start_hist = 1957.0,
  main_title = 'Differences in the Sum of Squared Forecast Errors for Airline Passengers',
   name_mdl1 = 'Airline model', name_mdl2 = 'Airline model + regressors')
```

---

plot_fts *Final t-statistics for the outlier identification procedure plot*

Description

Generates a plot of the final t-statistics for the outlier identification procedure.

Usage

```
plot_fts(
  m_seas = NULL,
  fts,
  this_cex = 0.5,
  start_plot = NULL,
  main_title = "Outlier T-Values",
  add_identified_otl = FALSE,
  col_otl = c("red", "blue", "darkgreen")
)
```

Arguments

m_seas	seas object generated from a call of seas on a single time series
fts	time series matrix containing final outlier t-statistics for all types of outlier specified by the user.
this_cex	Numeric scalar; sets the cex plotting parameter. Default sets cex = 0.5.
start_plot	Integer vector of length 2; Starting date for plot. Default is starting date for the time series.
main_title	Character string; main title of plot. Default is 'Outlier T-Values'.
add_identified_otl	Logical scalar; indicates if outlier plots will include identified outliers. Default is not including identified outliers.
col_otl	Character array of length 3; color used for different outliers, with the order being 'ao','ls','tc'. Default is c('red','blue','darkgreen').

Value

Generates a plot of the final t-statistics from the automatic outlier identification procedure.

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)', outlier.types = 'all')
air_fts_good <- seasonal::series(m_air, "fts")
plot_fts(m_air, air_fts_good, main_title = 'Outlier T-Values for Airline Passengers')
```

plot_matrix *Plot time series matrix*

Description

Generate plot of a matrix of user-specified time series.

Usage

```
plot_matrix(
  this_matrix = NULL,
  main_title = NULL,
  main_title_line = 2.75,
  main_title_cex = 1.25,
  y_label = NULL,
  start_plot = NULL,
  do_grid = FALSE,
  draw_recess = FALSE,
  recess_start = NULL,
  recess_col = NULL,
  recess_sub = TRUE,
  this_col = c("grey", "blue", "green", "brown", "red", "yellow"),
  this_line_type = rep(1, 6),
  add_legend = FALSE,
  this_legend_position = "topleft",
  this_legend_title = "Series",
  this_legend_inset = 0,
  this_legend_entry = paste("srs", 1:6, sep = ""),
  this_legend_cex = 0.8
)
```

Arguments

this_matrix	Numeric matrix; columns of time series object to be plotted.
main_title	Character string; main title of plot. Default is column name.
main_title_line	Integer scalar; position of main title of plot. Default is 2.75.
main_title_cex	Numeric scalar; scaling for main title of plot. Default is 1.25.
y_label	Character string; y-axis label for plot, if specified.
start_plot	Integer vector of length 2; Starting date for plot. Default is starting date for the time series.
do_grid	Logical scalar; indicates if certain plots will have grid lines. Default is no grid lines.
draw_recess	Logical scalar; indicates if certain plots will have shaded areas for NBER recession dates. Default is no recession shading.
recess_start	numeric matrix; Rows of dates for additional recession starting and ending dates. Default is not to add recession dates.
recess_col	Character string; color used for shading of recession region. Default is 'lightgrey'.

recess_sub — Logical scalar; indicates if x-axis label for recession is produced for this plot. Default is x-axis label is produced

this_col — Character array of length 6; color used for series in the order specified by the user. Default is `c('grey','blue','green','brown','red','yellow')`.

this_line_type — Integer vector; indicates line type of each plot produced. Default is 1:6

add_legend — Logical scalar; indicates if legend is produced for this plot. Default is legend not produced

this_legend_position
: Character string; indicates position of legend. Default is `'topleft'`.

this_legend_title
: Character string; indicates title of legend. Default is `'Series'`.

this_legend_inset
: Integer scalar; indicates inset for legend. Default is `0`.

this_legend_entry
: Character array; entries for the lengend. Default is `'Srs1'`.

this_legend_cex
: Numeric scalar; scaling for legend. Default is `0.8`.

Value

Generate plot of user-specified series. If matrix not specified, print out error message and return NULL.

Examples

```
BP_Region_Matrix <-
   cbind(xt_data_list$mwto, xt_data_list$neto, xt_data_list$soto, xt_data_list$weto)
plot_matrix(BP_Region_Matrix, y_label = 'Building Permits', do_grid = TRUE,
            draw_recess = TRUE, this_col = c("grey", "blue", "green", "brown"))
```

---

plot_multiple *Multiple plots on a single page*

Description

Generates a page of plots for a time series, seasonal adjustment of the time series, and trend component. Plotting the trend is optional. The series name is used for the title.

Usage

```
plot_multiple(
  seas_obj_list = NULL,
  i1,
  i2,
  this_row,
  this_col,
  plot_trend,
  seas_obj_names,
  plot_start = NULL,
  plot_end = NULL,
```

```
  outer_title,
  group_title,
  col_vec = c("grey", "blue", "green"),
  do_grid = FALSE,
  draw_recess = FALSE,
  recess_start = NULL,
  recess_col = NULL,
  recess_sub = TRUE
)
```

Arguments

seas_obj_list	List of seas arguments generated by seas() of the seasonal package.
i1	Integer scalar; index of first series to be plotted within the plotting frame.
i2	Integer scalar; index of last series to be plotted within the plotting frame.
this_row	Integer scalar; number of rows in multi-frame plot
this_col	Integer scalar; number of columns in multi-frame plot
plot_trend	Logical scalar; if TRUE, trend is included in plot, FALSE trend is not included.
seas_obj_names	Vector of character strings; the names of the series being plotted
plot_start	Integer array of length 2; start date for series to be plotted. If not specified, starting date of series used.
plot_end	Integer array of length 2; end date for series to be plotted. If not specified, ending date of series used.
outer_title	Character string, outer title of set of plots. If not specified, outer title is 'Series (grey), SA (blue), Trend (green) plot' or 'Series (grey), SA (blue)' if trend isn't specified.
group_title	Character string with a group title of series, if specified.
col_vec	Character array of length 5; color vector for lines in the plots. Default is c('grey', 'blue', 'green') for original series, SA, Trend
do_grid	Logical scalar; indicates if certain plots will have grid lines. Default is no grid lines.
draw_recess	Logical scalar; indicates if certain plots will have shaded areas for NBER recession dates. Default is no recession shading.
recess_start	numeric matrix; Rows of dates for additional recession starting and ending dates. Default is not to add recession dates.
recess_col	Character string; color used for shading of recession region. Default is 'lightgrey'.
recess_sub	Logical scalar; indicates if x-axis label for recession is produced for this plot. Default is x-axis label is produced

Value

Generate plots of seasonally adjusted series for every seas element in the list. Plots will be laid out in this_row rows and this_col columns. No values are returned.

Examples

```
xt_lauto <- seasonal::seas(xt_data_list, slidingspans = "", transform.function = "log",
                           arima.model = "(0 1 1)(0 1 1)",
                           forecast.maxlead=36, check.print = c( "pacf", "pacfplot" ))
R.devices::devEval('png', name='bp', tags = 'p01', sep = '_', {
 plot_multiple(xt_lauto, 1, 4, 2, 2, plot_trend = FALSE, seas_obj_names = names(xt_lauto),
          outer_title = 'Series (grey), SA (blue) plot', group_title='Building Permits',
            do_grid = TRUE, draw_recess = TRUE, recess_sub = FALSE)
})
```

---

plot_ratio *Ratio plot*

Description

Generates a high-definition plot around a reference line other than zero.

Usage

```
plot_ratio(
  ratio_series = NULL,
  ratio_range = range(ratio_series),
  main_title = NULL,
  ratio_mean = 1,
  ratio_color = NULL,
  plot_series = TRUE
)
```

Arguments

ratio_series	Time series of ratios/factors for which you want to generate a high definition plot
ratio_range	Range of values you wish the plot to be plotted over. Default is range of the series.
main_title	Title for the plot. Default is character string 'Ratio Plot'.
ratio_mean	Assumed mean value for the ratio. Default is 1.0
ratio_color	Color used for lines in ratio plot. Default is 'black'.
plot_series	Logical scalar. if TRUE, function will generate a plot of the series first of type='n'. If FALSE, the ratio will be plotted on the current defined plot. Default is TRUE.

Value

Generates a high definition plot of rations centered on one, by default.

Examples

```
m_air <- seasonal::seas(AirPassengers, transform.function= 'log', arima.model = '(0 1 1)(0 1 1)')
air_sf <- seasonal::series(m_air, 's10')
plot_ratio(air_sf, main_title = 'SEATS seasonal for Airline Passenger', ratio_color = 'darkblue')
```

plot_resid *Residual plot*

Description

Generates a plot of the regARIMA residuals with diagnostic information

Usage

```
plot_resid(
  m_seas = NULL,
  main_title = "ARIMA Residuals",
  series_name = NULL,
  do_grid = TRUE,
  draw_recess = TRUE,
  recess_start = NULL,
  recess_col = NULL,
  recess_sub = TRUE,
  use_ratio = FALSE,
  this_col = "green"
)
```

Arguments

m_seas	seas object generated from a call of seas on a single time series
main_title	Character string; main title of plot. Default is 'ARIMA Residuals'.
series_name	Character scalar; name of the time series used in m.
do_grid	Logical scalar; indicates if certain plots will have grid lines. Default is grid lines plotted.
draw_recess	Logical scalar; indicates if certain plots will have shaded areas for NBER recession dates. Default is recession shading plotted.
recess_start	numeric matrix; Rows of dates for additional recession starting and ending dates. Default is not to add recession dates.
recess_col	Character string; color used for shading of recession region. Default is 'lightgrey'.
recess_sub	Logical scalar; indicates if x-axis label for recession is produced for this plot. Default is x-axis label is produced
use_ratio	Logical scalar; indicates if plots of seasonal factors, irregular, and residuals are done as ratio plots. Default has these plots as time series line plots.
this_col	Character string; color used for residuals. Default is 'green'.

Value

Generates a plot of the regARIMA residuals with diagnostic information in the sub-headers.

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)')
plot_resid(m_air, main_title = 'ARIMA Residuals for Airline Passengers', use_ratio = TRUE,
           this_col='darkblue')
```

plot_sa_list *Plots of seasonally adjusted series for seasonal objects in a list.*

Description

Generate plots of seasonally adjusted series for every element in the list. The series name is used for the title.

Usage

```
plot_sa_list(
  seas_obj_list = NULL,
  this_row = 2,
  this_col = 2,
  pdf_file = NULL,
  this_dir = NULL,
  plot_trend = TRUE,
  group_title = NULL,
  col_vector = c("grey", "blue", "darkgreen"),
  outer_title = NULL,
  plot_start = NULL,
  plot_end = NULL,
  do_grid = FALSE,
  draw_recess = FALSE,
  recess_start = NULL,
  recess_col = NULL,
  recess_sub = TRUE
)
```

Arguments

seas_obj_list	List object of seas arguments generated by seas() of the seasonal package.
this_row	Number of rows in multi-frame plot
this_col	Number of columns in multi-frame plot
pdf_file	File name of PDF file, if specified.
this_dir	Character scalar; directory where the graphics file generated. Default is the current working directory.
plot_trend	Logical scalar; if TRUE, trend is included in plot, FALSE trend is not included.
group_title	Character string with a group title of series, if specified.
col_vector	Character array of length 3; color vector for lines in the plots. Default is c('grey', 'blue', 'green') for original series, SA, Trend
outer_title	Character string; outer title of set of plots, if specified.
plot_start	Integer array of length 2; start date for series to be plotted. If not specified, starting date of series used.
plot_end	Integer array of length 2; end date for series to be plotted. If not specified, ending date of series used.
do_grid	Logical scalar; if TRUE, grid lines are included in plot, FALSE grid lines are not included.

draw_recess	Logical scalar; if TRUE, recession periods are included in plot, FALSE recession periods are not included.
recess_start	numeric matrix; Rows of dates for additional recession starting and ending dates. Default is not to add recession dates.
recess_col	Character string; color used for shading of recession region. Default is 'lightgrey'.
recess_sub	Logical scalar; indicates if x-axis label for recession is produced for this plot. Default is x-axis label is produced

Value

Generate plots of seasonally adjusted series for every seas element in the list. Plots will be laid out in this_row rows and this_col columns. No values are returned.

Examples

```
xt_lauto <- seasonal::seas(xt_data_list, slidingspans = "", transform.function = "log",
                           arima.model = "(0 1 1)(0 1 1)",
                           forecast.maxlead=36, check.print = c( "pacf", "pacfplot" ))
plot_sa_list(xt_lauto, 3, 2, plot_trend=TRUE, pdf_file='BP_sa_trend_last6.pdf',
       group_title='Building Permits', plot_start=c(2001,1))
```

---

plot_sa_list_split *Plots of seasonally adjusted series for seasonal objects in a list, split into individual graphics files.*

Description

Generates a page of plots for a time series, seasonal adjustment of the time series, and trend component. Plotting the trend is optional.

Usage

```
plot_sa_list_split(
  seas_obj_list = NULL,
  this_row = 2,
  this_col = 2,
  file_name_base = "SAPlot",
  this_dir = NULL,
  plot_type = "png",
  plot_trend = TRUE,
  group_title = NULL,
  outer_title = NULL,
  plot_start = NULL,
  plot_end = NULL,
  do_grid = FALSE,
  do_recess = FALSE,
  col_vector = c("grey", "blue", "darkgreen")
)
```

Arguments

seas_obj_list	List of seas arguments generated by seas() of the seasonal package.
this_row	Number of rows in multi-frame plot
this_col	Number of columns in multi-frame plot
file_name_base	Character string that serves as the base for graphics file name, Default is 'SAPlot'
this_dir	Character scalar; directory where the graphics file generated. Default is the current working directory.
plot_type	Character string; type of graphics file - possible entries include 'pdf', 'png', 'eps'. Default is 'png'
plot_trend	Logical scalar; if TRUE, trend is included in plot, FALSE trend is not included.
group_title	• character string with a group title of series, if specified.
outer_title	• character string, outer title of set of plots. If not specified, outer title is 'Series (grey), SA (blue), Trend (green) plot' or 'Series (grey), SA (blue)' if trend isn't specified.
plot_start	• start date for series to be plotted. If not specified, starting date of series used.
plot_end	• end date for series to be plotted. If not specified, ending date of series used.
do_grid	Logical scalar; indicates if certain plots will have grid lines. Default is no grid lines.
do_recess	Logical scalar; indicates if certain plots will have shaded areas for NBER recession dates. Default is no recession shading.
col_vector	• color vector for lines in the plots. Default is c('grey', 'blue', 'green') for original series, SA, Trend

Value

Generate plots of seasonally adjusted series for every seas element in the list. Plots will be laid out in this_row rows and this_col columns, into individual pages. No values are returned.

Examples

```
xt_lauto <- seasonal::seas(xt_data_list, slidingspans = "", transform.function = "log",
                           arima.model = "(0 1 1)(0 1 1)",
                           forecast.maxlead=36, check.print = c( "pacf", "pacfplot" ))
plot_sa_list_split(xt_lauto, 3, 2, plot_trend=TRUE, file_name_base='BP_sa_trend',
       plot_type = 'png', group_title='Building Permits', plot_start=c(2001,1))
```

---

plot_series *Plot individual series.*

Description

Generate plot of user-specified series.

Usage

```
plot_series(
  this_series = NULL,
  main_title = NULL,
  main_title_line = 2.75,
  main_title_cex = 1.25,
  y_label = NULL,
  y_limit = NULL,
  start_plot = NULL,
  do_grid = FALSE,
  draw_recess = FALSE,
  recess_start = NULL,
  recess_col = NULL,
  recess_sub = TRUE,
  this_trans = TRUE,
  use_ratio = FALSE,
  this_col = "grey",
  this_line_type = 1,
  this_point_type = NULL,
  add_legend = FALSE,
  this_legend_position = "topleft",
  this_legend_title = "Series",
  this_legend_inset = 0,
  this_legend_entry = "Srs1",
  this_legend_cex = 0.8,
  this_legend_col = "grey",
  this_legend_lty = 1
)
```

Arguments

this_series	Numeric vector; time series object to be plotted.
main_title	Character string; main title of plot. Default is no title.
main_title_line	Integer scalar; position of main title of plot. Default is 2.75.
main_title_cex	Numeric scalar; scaling for main title of plot. Default is 1.25.
y_label	Character string; y-axis label for plot, if specified.
y_limit	Numeric vector of length 2; Range of values you wish the plot to be plotted over. Default is range of the seasonal factors.
start_plot	Integer vector of length 2; Starting date for plot. Default is starting date for the time series.
do_grid	Logical scalar; indicates if certain plots will have grid lines. Default is no grid lines.
draw_recess	Logical scalar; indicates if certain plots will have shaded areas for NBER recession dates. Default is no recession shading.
recess_start	numeric matrix; Rows of dates for additional recession starting and ending dates. Default is not to add recession dates.
recess_col	Character string; color used for shading of recession region. Default is 'lightgrey'.

recess_sub Logical scalar; indicates if x-axis label for recession is produced for this plot. Default is x-axis label is produced

this_trans Logical scalar; indicates if the adjustment was done with a log transform. Default is TRUE.

use_ratio Logical scalar; indicates if plots of seasonal factors, irregular, and residuals are done as ratio plots. Default has these plots as time series line plots.

this_col Character array of length 6; color used for series in the order specified by the user. Default is `c('grey','blue','green','brown','red','yellow')`.

this_line_type Integer array; line type used for series

this_point_type
Integer array; point type used for series. Default is no points plotted.

add_legend Logical scalar; indicates if legend is produced for this plot. Default is legend not produced

this_legend_position
Character string; indicates position of legend. Default is `'topleft'`.

this_legend_title
Character string; indicates title of legend. Default is `'Series'`.

this_legend_inset
Integer scalar; indicates inset for legend. Default is `0`.

this_legend_entry
Character array; entries for the lengend. Default is `'Srs1'`.

this_legend_cex
Numeric scalar; scaling for legend. Default is `0.8`.

this_legend_col
Character string; color of lines in the legend. Default is `'grey'`.

this_legend_lty
Numeric scalar; color of lines in the legend. Default is `1`.

Value

Generate plot of user-specified series. Can be first in a series of plots, with other lines or points added after calling this routine. If series not specified, print out error message and return NULL.

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)',
                        outlier.types = "all", x11 = "",
                        forecast.maxlead = 36)
plot_series(AirPassengers, y_label = 'Air Passengers', do_grid = TRUE,
            draw_recess = TRUE, this_col = 'black',
            start_plot = c(1958,1), this_point_type = 1,
            add_legend = TRUE,
            this_legend_position = "topleft",
            this_legend_title = "Air Passengers", this_legend_inset = 0,
            this_legend_entry = c("Series", "SA", "Trend"),
            this_legend_col = c("black", "blue", "darkgreen"),
            this_legend_lty = 1:3)
lines(window(seasonal::final(m_air), start=c(1958,1)), col = "blue", lty = 2)
lines(window(seasonal::trend(m_air), start=c(1958,1)), col = "darkgreen", lty = 3)
```

plot_sf *Seasonal factor (and the SI-ratios) plot grouped by month/quarter*

Description

Generates a special plot of the seasonal factors (and the SI-ratios) grouped by month/quarter

Usage

```
plot_sf(
  m_seas = NULL,
  add_si = FALSE,
  main_title = "Seasonal Sub-Plots",
  this_col = c("darkgreen", "darkblue", "darkgrey"),
  add_mean_line = TRUE,
  add_legend = FALSE,
  this_legend_position = "topleft",
  this_legend_title = "SF Plot",
  this_legend_inset = 0,
  this_legend_cex = 0.8
)
```

Arguments

m_seas	seas object generated from a call of seas on a single time series
add_si	Logical scalar; indicates if seasonal factor plots will include SI ratios for X-11 seasonal adjustments. Default is not including SI ratios.
main_title	Character string; main title of plot. Default is 'Seasonal Sub-Plots'.
this_col	Character array of length 2; color used for seasonal factors, SI-ratios, and seasonal mean. Default is c("darkgreen","darkblue","darkgrey").
add_mean_line	Logical scalar; indicates if seasonal factor plots will include lines for seasonal means. Default includes lines for seasonal means.
add_legend	Logical scalar; indicates if legend is produced for this plot. Default is legend not produced
this_legend_position	Character string; indicates position of legend. Default is 'topleft'.
this_legend_title	Character string; indicates title of legend. Default is 'Series'.
this_legend_inset	Integer scalar; indicates inset for legend. Default is 0.
this_legend_cex	Numeric scalar; scaling for legend. Default is 0.8.

Value

Generates a special plot of the seasonal factors (and the SI-ratios) grouped by month/quarter

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)', x11='')
plot_sf(m_air, add_si = TRUE, main_title = 'Air Passengers Seasonal Sub-Plots',
         this_col = c('darkgreen', 'darkblue', 'grey'), add_legend = TRUE)
```

`plot_sf_mean` *Seasonal factor mean plot*

Description

Generates a plot of the means of the seasonal factors

Usage

```
plot_sf_mean(
  this_sf = NULL,
  this_period = NULL,
  this_col = "green",
  y_limit = range(this_sf),
  this_freq,
  this_trans = TRUE,
  this_title = "Mean of Seasonal Factors",
  forecast = 0,
  this_type = "Seasonal",
  add_line = FALSE,
  add_legend = FALSE,
  this_legend_position = "topleft",
  this_legend_title = "SF Means",
  this_legend_inset = 0,
  this_legend_entry = "Srs1",
  this_legend_col = "green",
  this_legend_lty = 1,
  this_legend_cex = 0.8
)
```

Arguments

`this_sf`	`tis` object of the seasonal factors from a weekly seasonal adjustment
`this_period`	Character scalar; vector with period number of the observations.
`this_col`	Character scalar; color used for factor plots. Default is `green`.
`y_limit`	Numeric vector of length 2; Range of values you wish the plot to be plotted over. Default is range of the seasonal factors.
`this_freq`	integer scalar; time series frequency.
`this_trans`	Logical scalar; indicates if the adjustment was done with a log transform. Default is TRUE.
`this_title`	Character string; main title of plot. Default is `'Mean of Seasonal Factors'`.
`forecast`	Integer scalar; Number of forecasts appended to the seasonal factors. Default is 0.

this_type Character string; type of factors plotted. Default is 'seasonal'.

add_line Logical scalar; indicates if this line is being added to an existing plot. Default is FALSE.

add_legend Logical scalar; indicates if legend is produced for this plot. Default is legend not produced

this_legend_position
Character string; indicates position of legend. Default is 'topleft'.

this_legend_title
Character string; indicates title of legend. Default is 'Series'.

this_legend_inset
Integer scalar; indicates inset for legend. Default is 0.

this_legend_entry
Character array; entries for the lengend. Default is 'Srs1'

this_legend_col
Character array; line colors for legend. Default is 'blue'.

this_legend_lty
Integer array; line types for legend. Default is 1.

this_legend_cex
Numeric scalar; scaling for legend. Default is 0.8.

Value

Generate plot of the means of seasonal factors by period, or add to existing plot. If seasonal factors not specified, print out error message and return NULL.

Examples

```
xt_lauto <- seasonal::seas(xt_data_list, slidingspans = "", transform.function = "log",
                           arima.model = "(0 1 1)(0 1 1)",
                       forecast.maxlead=36, check.print = c( "pacf", "pacfplot" ))
BP_Region_Sf <-
 cbind(seasonal::series(xt_lauto$mwto, "s10"), seasonal::series(xt_lauto$neto, "s10"),
       seasonal::series(xt_lauto$soto, "s10"), seasonal::series(xt_lauto$weto, "s10"))
this_sf_limit <- range(BP_Region_Sf)
plot_sf_mean(BP_Region_Sf[,1], cycle(BP_Region_Sf[,1]),
  this_col = 'blue',
  y_limit = this_sf_limit,
  this_freq = 12,
  forecast = 0,
  this_title = 'Building Permits Seasonal Means',
  add_legend = TRUE,
  this_legend_position = "topleft",
  this_legend_title = "SF Means",
  this_legend_inset = 0,
  this_legend_entry = c("MW", "NE", "SO", "WE"),
  this_legend_col = c("blue", "red", "green", "purple"),
  this_legend_lty = rep(1,4),
  this_legend_cex = 0.8)
plot_sf_mean(BP_Region_Sf[,2], cycle(BP_Region_Sf[,2]),
  this_col = 'red',
  this_freq = 12,
  forecast = 0,
  add_line = TRUE)
```

```
 plot_sf_mean(BP_Region_Sf[,3], cycle(BP_Region_Sf[,3]),
   this_col = 'green',
   this_freq = 12,
   forecast = 0,
   add_line = TRUE)
 plot_sf_mean(BP_Region_Sf[,4], cycle(BP_Region_Sf[,4]),
   this_col = 'purple',
   this_freq = 12,
   forecast=0,
   add_line=TRUE)
legend("topleft", legend=c('MW', 'NE', 'SO', 'WE'),
       col=c('blue', 'red', 'green', "purple"), lty=rep(1,4), cex=0.8)
```

`plot_single_cell` *Single time series plot.*

Description

Generates a single plot of a time series, seasonal adjustment of the time series, and trend component. Plotting the trend is optional. The series name is used for the title.

Usage

```
plot_single_cell(
  this_series = NULL,
  this_sadj = NULL,
  this_trend = NULL,
  this_name = NULL,
  col_vector = c("grey", "blue", "darkgreen")
)
```

Arguments

`this_series`	Original time series
`this_sadj`	Seasonal adjustment of `this_series`.
`this_trend`	Trend component estimated from `this_series`. Default is to not print the trend component.
`this_name`	Name of the original series. If specified, this is used as the title of the plot. Default is to generate the title based on what components are plotted.
`col_vector`	Character vector of length 3; colors used for the lines in the plot. First color is for the oringial series, second is for the SA series, third is for the trend. Default is `col_vector=c('grey','blue','darkgreen')`.

Value

Produces plot of single plot of a time series, seasonal adjustment of the time series, and trend component. No values are returned.

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)', x11='')
plot_single_cell(AirPassengers,seasonal::final(m_air),seasonal::trend(m_air), 'Air Passengers')
```

plot_table *Plot table from X-13ARIMA-SEATS seasonal adjustment.*

Description

Generate plot of user-specified series.

Usage

```
plot_table(
  this_seas_object = NULL,
  this_table = NULL,
  main_title = NULL,
  y_label = NULL,
  y_limit = NULL,
  start_plot = NULL,
  do_grid = FALSE,
  draw_recess = FALSE,
  recess_start = NULL,
  recess_col = NULL,
  recess_sub = TRUE,
  add_otl = FALSE,
  use_ratio = FALSE,
  add_sub_title = FALSE,
  this_line_type = NULL,
  this_col = c("grey", "blue", "green", "brown", "red", "yellow"),
  otl_col = c("red", "blue", "green", "brown", "grey", "yellow")
)
```

Arguments

this_seas_object	seas object generated from a call of seas on a single time series
this_table	Character string; X-13ARIMA-SEATS table name or abbreviation. If not a valid table name, the function will print an error message and return a NULL.
main_title	Character string; main title of plot. Default is 'Cumulative periodogram'.
y_label	Character string; y-axis label for plot, if specified.
y_limit	Numeric vector of length 2; Range of values you wish the plot to be plotted over. Default is range of the seasonal factors.
start_plot	Integer vector of length 2; Starting date for plot. Default is starting date for the time series.
do_grid	Logical scalar; indicates if certain plots will have grid lines. Default is no grid lines.
draw_recess	Logical scalar; indicates if certain plots will have shaded areas for NBER recession dates. Default is no recession shading.
recess_start	numeric matrix; Rows of dates for additional recession starting and ending dates. Default is not to add recession dates.
recess_col	Character string; color used for shading of recession region. Default is 'lightgrey'.

recess_sub	Logical scalar; indicates if x-axis label for recession is produced for this plot. Default is x-axis label is produced
add_otl	Logical scalar; indicates if lines for identified outliers are included in series plots. Default is not including lines for identified outliers.
use_ratio	Logical scalar; indicates if plots of seasonal factors, irregular, and residuals are done as ratio plots. Default has these plots as time series line plots.
add_sub_title	Logical scalar; indicates if plots will include subtitles denoting what series are plotted. Default is not including subheaders.
this_line_type	Integer vector; indicates line type of each plot produced. Default is 1:length(this_table)
this_col	Character array of length 6; color used for series in the order specified by the user. Default is `c('grey','blue','green','brown','red','yellow')`.
otl_col	Character array of length 6; color used for different outliers, with the order being `'ao','ls','tc','so','rp','tls'`. Default is `c('red','blue','green','brown','grey','ye`

Value

Generate plot of user-specified series. Can be more than one series. If series not specified, print out error message and return NULL.

Examples

```
m_air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)', x11='')
plot_table(m_air, c('a1', 'b1', 'd11'), y_label = 'AirPassengers',
         do_grid = TRUE, draw_recess = TRUE, use_ratio = TRUE, add_otl = TRUE,
         this_col = c('grey', 'darkgreen', 'darkblue'))
```

---

plot_year_over_year *Year over year plot of individual series.*

Description

Generate plot of user-specified series with each year as a separate line.

Usage

```
plot_year_over_year(
  this_series = NULL,
  main_title = NULL,
  this_col = NULL,
  start_plot = NULL,
  this_legend = TRUE,
  this_legend_cex = 0.75,
  this_right_mar = 5.6
)
```

Arguments

this_series	Numeric vector; time series object to be plotted.
main_title	Character string; main title of plot. Default is no title.
this_col	Character array; color used for series in the order specified by the user. This array should be as long as the number of years plotted. If only one color is specified, colortools::wheel(this_col) is used to construct an array with enough colors. Default is rainbow(ny), where ny is the number of years plotted.
start_plot	Integer vector of length 2; Starting date for plot. Default is starting date for the time series.
this_legend	Logical scalar; indicates if a legend is produced for the plot. Default is TRUE.
this_legend_cex	Numeric scalar; scaling for legend. Default is 0.75.
this_right_mar	Numeric scalar; value associated with the margin of the right y-axis specified in the mar entry of the graphics parameters (par). Default is 5.6.

Value

Generate year over year plot of user-specified series. If series not specified, print out error message and return NULL.

Examples

```
plot_year_over_year(AirPassengers, "Airline Passenger Series (1949 - 1960)")
```

---

reset_par *Reset* par()

Description

Reset graphics parameters for plots; taken from stackoverflow post https://stackoverflow.com/questions/9292563/reset-the-graphical-parameters-back-to-default-values-without-use-of-dev-off

Usage

```
reset_par()
```

Value

returns default graphics parameters

Examples

```
par(mar=c(5.1, 3.1, 4.1, 1.1), mfrow=c(2,2))
xt_names <- names(xt_data_list)
for (i in 1:4) {
     plot(xt_data_list[[i]], main = xt_names[i], type="l")
}
reset_par()
```

visual_sig_peaks *Flag visual significant peaks in spectra*

Description

Determine positions of visual significant peaks in spectra

Usage

```
visual_sig_peaks(m_seas, spec_type = "sa", spec_freq_code = "seas")
```

Arguments

m_seas	seas object generated from a call of seas on a single time series
spec_type	Character string; type of spectrum. Possible values are 'ori','irr','rsd','sa'. Default is 'sa'.
spec_freq_code	Character string; type of frequency being tested. Possible values are 'seas' or 'td'. Default is 'seas'.

Value

If visually significant peaks found, a numveric vector of the position of the peak frequecies. If no peaks found, 0.

Examples

```
m.air <- seasonal::seas(AirPassengers, arima.model = '(0 1 1)(0 1 1)', x11='')
vp_ori_seas <- visual_sig_peaks(m.air,'ori')
vp_ori_td <- visual_sig_peaks(m.air,'ori','td')
```

wheel_invisible *Invisible version of* colortools *function* wheel

Description

Get values of the function wheel from the colortools package without producing the plot of the resulting color wheel. Adapted from https://stackoverflow.com/questions/20363266/how-can-i-suppress-the-creation-of-a-plot-while-calling-a-function-in-r

Usage

```
wheel_invisible(this_color, n_colors)
```

Arguments

this_color	An R color name or a color in hexadecimal notation
n_colors	Numeric scalar; number of colors to be generated by wheel

Value

A character vector with the color names of the generated wheel in hexadecimal notation.

Examples

```
this_wheel <- wheel_invisible("darkblue", 19)
plot_year_over_year(AirPassengers, "Airline Passenger Series (1949 - 1960)",
                    this_col = this_wheel)
```

---

xt_data_list *US Building Permits*

Description

A list object with 12 components of US Building Permits expressed as time series objects

Usage

```
xt_data_list
```

Format

A list object with 12 time series elements:

mw1u Midwest one family building permits

mwto Midwest total building permits

ne1u Northeast one family building permits

neto Northeast total building permits

so1u South one family building permits

soto South total building permits

we1u West one family building permits

weto West total building permits

us1u US one family building permits

us24 US 2-4 family building permits

us5p US 5+ family building permits

usto US total family building permits

Index