

Package: ggdemetra3 (via r-universe)

October 18, 2026

Type Package

Title 'ggplot2' Extension for Seasonal and Trading Day Adjustment with 'JDemetra+' 3.0

Description Provides 'ggplot2' extensions for seasonal and trading day adjustment based on the R interface to 'JDemetra+' version 3.0.

Version 0.0.3

Depends R (>= 4.1.0), ggplot2 (>= 2.0.0)

Imports ggrepel, gridExtra, rjd3jars(>= 0.0.6), rjd3toolkit (>= 3.7.0), rjd3x13 (>= 3.7.0), rjd3tramoseats (>= 3.7.0), rjd3filters (>= 2.1.1), rjd3highfreq (>= 3.0.0), MASS, utils, rJava

Suggests knitr, rmarkdown

Remotes github::rjdverse/rjd3highfreq

SystemRequirements Java (>= 17)

License EUPL

URL <https://aqlt.github.io/ggdemetra/>,
<https://github.com/AQLT/ggdemetra3>

BugReports <https://github.com/AQLT/ggdemetra3/issues>

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

VignetteBuilder knitr

Config/roxygen2/version 8.1.1

Config/pak/sysreqs make default-jdk libprotobuf-dev protobuf-compiler libprotoc-dev

Repository <https://aqlt.r-universe.dev>

Date/Publication 2026-10-18 14:08:30 UTC

RemoteUrl <https://github.com/AQLT/ggdemetra3>

RemoteRef HEAD

RemoteSha 598431c64a4a59f58d266b5982c7fa8dec0e363b

Contents

autoplot_ggdemetra3	2
components	4
geom_arima	5
geom_diagnostics	7
geom_outlier	10
geom_sa	12
ggplot_filters	15
init_ggplot	16
ipi_c_eu	17
siratio	18
ts2df	20
usclaims	21
Index	22

autoplot_ggdemetra3	<i>Plot Seasonal Adjustment Objects</i>
---------------------	---

Description

Plot Seasonal Adjustment Objects

Usage

```
## S3 method for class 'JD3_X13_OUTPUT'
autoplot(
  object,
  components = c("y", "sa", trend = "t", seasonal = "s", irregular = "i"),
  forecast = FALSE,
  ...
)

## S3 method for class 'JD3_TRAMOSEATS_OUTPUT'
autoplot(
  object,
  components = c("y", "sa", trend = "t", seasonal = "s", irregular = "i"),
  forecast = FALSE,
  ...
)

## S3 method for class 'JD3_X13_RSLTS'
autoplot(
  object,
  components = c("y", "sa", trend = "t", seasonal = "s", irregular = "i"),
  forecast = FALSE,
  ...
)
```

```

)

## S3 method for class 'JD3_TRAMOSEATS_RSLTS'
autoplot(
  object,
  components = c("y", "sa", trend = "t", seasonal = "s", irregular = "i"),
  forecast = FALSE,
  ...
)

## S3 method for class 'JD3_Object'
autoplot(
  object,
  components = c("y", "sa", trend = "t", seasonal = "s", irregular = "i"),
  forecast = FALSE,
  ...
)

## S3 method for class 'hf_decomposition'
autoplot(
  object,
  components = c("y", "sa", trend = "t", seasonal = "s", irregular = "i"),
  dates = NULL,
  ...
)

```

Arguments

object	A seasonal adjustment model (from rjd3x13 , rjd3tramoseats , or rjd3highfreq).
components	Components to display. Can be "y" (input time series), "sa" (seasonally adjusted), "t" (trend-cycle), "y_cal" (calendar-adjusted), "s" (seasonal), "i" (irregular), and "cal" (calendar). The vector can be named to change the strip label.
forecast	Logical indicating whether the forecast series should be plotted (default is FALSE). Ignored for "hf_decomposition" objects (rjd3highfreq).
...	Additional arguments passed to methods.
dates	For "hf_decomposition" objects (rjd3highfreq), a numeric or date vector corresponding to the time series. If NULL, a sequence of integers is used. For models created with rjd3highfreq::fractional_airline_decomposition() or rjd3highfreq::multi_airline_decomposition() , the <code>series_time</code> parameter is used if available.

Value

A **ggplot2** object representing the faceted components.

Examples

```
autoplot(rjd3x13::jx13(ipi_c_eu[, "FR"]))
```

```

hf_data <- usclaims[usclaims$date <= "2019-12-01",]
mod_hf <- rjd3highfreq::fractional_airline_decomposition(
  hf_data[, "Initial claims"],
  period = 52,
  series_time = hf_data[, "date"]
)
autoplot(mod_hf, dates = usclaims$date[usclaims$date <= "2019-12-01"])

```

components

Extract Components from Seasonal Adjustment Objects

Description

Extract Components from Seasonal Adjustment Objects

Usage

```
seasonal(x, forecast = FALSE, calendar = FALSE, seas_freq = NULL, ...)
```

```
trendcycle(x, forecast = FALSE, ...)
```

```
irregular(x, forecast = FALSE, corrected = FALSE, ...)
```

```
seasonaladj(x, forecast = FALSE, corrected = FALSE, ...)
```

```
calendaradj(x, forecast = FALSE, spec = NULL, ...)
```

```
calendar(x, forecast = FALSE, spec = NULL, ...)
```

```
raw(x, forecast = FALSE, backcast = FALSE, spec = NULL, ...)
```

Arguments

x	A seasonal adjustment model (from rjd3x13 , rjd3tramoseats , or rjd3highfreq).
forecast, backcast	Logicals indicating whether the forecast or backcast series should be returned. If forecast = TRUE and backcast = TRUE, the function only returns the forecast series. Default is FALSE. Ignored for rjd3highfreq objects.
calendar	If TRUE, seasonal() returns the seasonal and calendar components. Ignored for rjd3highfreq objects.
seas_freq	For methods with multiple seasonal frequencies (rjd3highfreq::multi_airline_decomposition() , rjd3highfreq::istl() or rjd3highfreq::mstl()), the seasonal frequency to extract. If NULL, the first seasonal frequency is returned.
...	Additional arguments passed to methods.

corrected	If TRUE, large extreme values are removed from the components. Ignored for rjd3tramoseats and rjd3highfreq objects.
spec	The specification used to estimate the model. Only used for "JD3_TRAMOSEATS_OUTPUT" objects for functions <code>calendaradj()</code> , <code>calendar()</code> and <code>raw()</code> (when <code>forecast = FALSE</code> and <code>backcast = TRUE</code>) because the output doesn't contain the desired component and the specification.

Details

These functions are S3 generics supporting several classes from the `rjdverse` ecosystem:

- X-13 models (**rjd3x13**): `JD3_X13_OUTPUT`, `JD3_X13_RSLTS`, `JD3_Object`;
- TRAMO-SEATS models (**rjd3tramoseats**): `JD3_TRAMOSEATS_OUTPUT`, `JD3_TRAMOSEATS_RSLTS`;
- High-frequency decompositions (**rjd3highfreq**): `hf_decomposition` (except for `calendaradj()` and `calendar()`, which are not supported for this class).

Value

A `ts` object (or a vector of numeric for "hf_decomposition" objects) containing the requested component.

A `ts` object with the requested component.

Examples

```
model <- rjd3x13::jx13(ipi_c_eu[, "FR"])
seasonal(model)
raw(model, forecast = TRUE)
```

geom_arima

Add ARIMA Model to Plots

Description

Function to directly add the ARIMA model used in the pre-adjustment process of seasonal adjustment to the plot. Only available for X-13 (`rjd3x13::x13()`) and TRAMO-SEATS (`rjd3tramoseats::tramoseats()`) methods.

Usage

```
geom_arima(
  mapping = NULL,
  data = NULL,
  stat = "arima",
  geom = c("text", "label"),
  position = "identity",
  ...,

```

```

method = c("x13", "tramoseats"),
spec = NULL,
frequency = NULL,
message = TRUE,
x_arima = NULL,
y_arima = NULL,
show.legend = NA,
inherit.aes = TRUE
)

```

Arguments

mapping	Set of aesthetic mappings created by <code>ggplot2::aes()</code> . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	A data.frame that contains the data used for the seasonal adjustment.
stat	The statistical transformation to use on the data for this layer, as a string.
geom	Character. The geometric object to use to display the data: <code>GeomText</code> (<code>geom = "text"</code> , the default, see <code>ggplot2::geom_text()</code>) or <code>GeomLabel</code> (<code>geom = "label"</code> , see <code>ggplot2::geom_label()</code>).
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
...	Other arguments passed on to <code>ggplot2::layer()</code> . These are often aesthetics, used to set an aesthetic to a fixed value, like <code>colour = "red"</code> or <code>size = 3</code> .
method	The method used for the seasonal adjustment. "x13" (by default) for the X-13 method (using <code>rjd3x13::jx13()</code>) and "tramoseats" for TRAMO-SEATS (using <code>rjd3tramoseats::jtramoseats()</code>). Both methods are only compatible with series of frequency 12 (monthly), 6 (bimonthly), 4 (quarterly), and 2 (semi-annual). For other frequencies, the following methods are available: "x11-extended" (see <code>rjd3highfreq::x11plus()</code>), "fractionalairline" (see <code>rjd3highfreq::fractional_airline_decomposition()</code>), "multiairline" (see <code>rjd3highfreq::multi_airline_decomposition()</code>), "stl" (see <code>rjd3highfreq::stlplus()</code>), "istl" (see <code>rjd3highfreq::istl()</code>), and "mstl" (see <code>rjd3highfreq::mstl()</code>).
spec	the specification used for the seasonal adjustment. For "x13" and "tramoseats" methods, it can be the name of a pre-defined specification or a user-defined specification created with <code>rjd3x13::spec_x13()</code> or <code>rjd3tramoseats::spec_tramoseats()</code> .
frequency	The frequency of the time series. By default (<code>frequency = NULL</code>), the frequency is computed automatically.
message	A logical value indicating whether a message is printed with the frequency used.
x_arima, y_arima	Position of the ARIMA model text. By default, the first position of the data is used.
show.legend	Logical. Should this layer be included in the legends? NA, the default, includes it if any aesthetics are mapped. FALSE never includes it, and TRUE always includes it. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them.

Details

With `geom = "text"`, the ARIMA model used in the pre-adjustment process of seasonal adjustment is directly added to the plot. With `geom = "label"`, a rectangle is drawn behind the ARIMA model text, making it easier to read.

Value

A **ggplot2** layer object to be added to a plot.

Examples

```
p_sa_ipi_fr <- ggplot(data = ipi_c_eu_df, mapping = aes(x = date, y = FR)) +
  geom_line() +
  labs(
    title = "Seasonal adjustment of the French industrial production index",
    x = "time", y = NULL
  ) +
  geom_sa(color = "red", message = FALSE)

# To add the ARIMA model
p_sa_ipi_fr +
  geom_arima(
    geom = "label",
    x_arima = -Inf, y_arima = -Inf,
    vjust = -1, hjust = -0.1,
    message = FALSE
  )
```

geom_diagnostics

Table of Diagnostics

Description

Adds a table of diagnostics to the plot. Only available for X-13 (`rjd3x13::x13()`) and TRAMO-SEATS (`rjd3tramoseats::tramoseats()`) methods.

Usage

```
geom_diagnostics(
  mapping = NULL,
  data = NULL,
  position = "identity",
  ...,
  method = c("x13", "tramoseats"),
  spec = NULL,
  frequency = NULL,
  message = TRUE,
```

```

diagnostics = NULL,
digits = 2,
xmin = -Inf,
xmax = Inf,
ymin = -Inf,
ymax = Inf,
table_theme = ttheme_default(),
inherit.aes = TRUE
)

```

Arguments

mapping	Set of aesthetic mappings created by <code>ggplot2::aes()</code> . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	A data frame that contains the data used for the seasonal adjustment.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
...	Other arguments passed on to <code>ggplot2::layer()</code> . These are often aesthetics, used to set an aesthetic to a fixed value, like <code>colour = "red"</code> or <code>size = 3</code> .
method	The method used for the seasonal adjustment. "x13" (by default) for the X-13 method (using <code>rjd3x13::jx13()</code>) and "tramoseats" for TRAMO-SEATS (using <code>rjd3tramoseats::jtramoseats()</code>). Both methods are only compatible with series of frequency 12 (monthly), 6 (bimonthly), 4 (quarterly), and 2 (semi-annual). For other frequencies, the following methods are available: "x11-extended" (see <code>rjd3highfreq::x11plus()</code>), "fractionalairline" (see <code>rjd3highfreq::fractional_airline_decomposition()</code>), "multiairline" (see <code>rjd3highfreq::multi_airline_decomposition()</code>), "stl" (see <code>rjd3highfreq::stlplus()</code>), "istl" (see <code>rjd3highfreq::istl()</code>), and "mstl" (see <code>rjd3highfreq::mstl()</code>).
spec	The specification used for the seasonal adjustment. For "x13" and "tramoseats" methods, it can be the name of a pre-defined specification or a user-defined specification created with <code>rjd3x13::spec_x13()</code> or <code>rjd3tramoseats::spec_tramoseats()</code> . For other methods, it can be a list of parameters used in the associated seasonal adjustment function.
frequency	The frequency of the time series. By default (<code>frequency = NULL</code>), the frequency is computed automatically.
message	A logical value indicating whether a message is printed with the frequency used.
diagnostics	Character vector containing the names of the diagnostics to plot. See <code>rjd3x13::userdefined_variables</code> or <code>rjd3tramoseats::userdefined_variables_tramoseats()</code> for the available parameters.
digits	Integer indicating the number of decimal places to be used for numeric diagnostics (default is 2).
xmin, xmax	Horizontal location of the raster in data coordinates.
ymin, ymax	Vertical location of the raster in data coordinates.
table_theme	List of theme parameters for the table of diagnostics (see <code>gridExtra::ttheme_default()</code>).
inherit.aes	If <code>FALSE</code> , overrides the default aesthetics, rather than combining with them.

Value

A **ggplot2** layer object to be added to a plot.

Examples

```
p_sa_ipi_fr <- ggplot(data = ipi_c_eu_df, mapping = aes(x = date, y = FR)) +
  geom_line() +
  labs(
    title = "Seasonal adjustment of the French industrial production index",
    x = "time", y = NULL
  ) +
  geom_sa(color = "red", message = FALSE)

# To add a table of diagnostics with the result of the X-11 combined test
# and the p-values of the residual seasonality QS and F-tests:
diagnostics <- c("diagnostics.seas-sa-combined", "diagnostics.seas-sa-qs", "diagnostics.seas-sa-f")
p_sa_ipi_fr +
  geom_diagnostics(
    diagnostics = diagnostics,
    ymin = 58, ymax = 72, xmin = 2010,
    table_theme = gridExtra::ttheme_default(base_size = 8),
    message = FALSE
  )

# To customize the names of the diagnostics in the plot:
diagnostics <- c(
  `Combined test` = "diagnostics.seas-sa-combined",
  `Residual qs-test (p-value)` = "diagnostics.seas-sa-qs",
  `Residual f-test (p-value)` = "diagnostics.seas-sa-f"
)
p_sa_ipi_fr +
  geom_diagnostics(
    diagnostics = diagnostics,
    ymin = 58, ymax = 72, xmin = 2010,
    table_theme = gridExtra::ttheme_default(base_size = 8),
    message = FALSE
  )

# To add the table below the plot:
p_diag <- ggplot(data = ipi_c_eu_df, mapping = aes(x = date, y = FR)) +
  geom_diagnostics(
    diagnostics = diagnostics,
    table_theme = gridExtra::ttheme_default(base_size = 8),
    message = FALSE
  ) +
  theme_void()

gridExtra::grid.arrange(
  p_sa_ipi_fr, p_diag,
  nrow = 2, heights = c(4, 1)
)
```

geom_outlier

*Outliers Texts***Description**

Function to directly add the outliers used in the pre-adjustment process of the seasonal adjustment to the plot. Only available for X-13 ([rjd3x13::x13\(\)](#)) and TRAMO-SEATS ([rjd3tramoseats::tramoseats\(\)](#)) methods.

Usage

```
geom_outlier(
  mapping = NULL,
  data = NULL,
  stat = "outlier",
  geom = c("text", "label", "text_repel", "label_repel"),
  position = "identity",
  ...,
  method = c("x13", "tramoseats"),
  spec = NULL,
  frequency = NULL,
  message = TRUE,
  first_date = NULL,
  last_date = NULL,
  coefficients = FALSE,
  digits = 1,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping	Set of aesthetic mappings created by ggplot2::aes() . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	A <code>data.frame</code> that contains the data used for the seasonal adjustment.
stat	The statistical transformation to use on the data for this layer, as a string.
geom	Character. The geometry to use to display the data: <code>GeomText</code> (<code>geom = "text"</code> , see ggplot2::geom_text()); <code>GeomLabel</code> (<code>geom = "label"</code> , see ggplot2::geom_label()); <code>GeomTextRepel</code> (<code>geom = "text_repel"</code> , the default, see ggrepel::geom_text_repel()); <code>GeomLabelRepel</code> (<code>geom = "label_repel"</code> , see ggrepel::geom_label_repel()).
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
...	Other arguments passed on to ggplot2::layer() . They may be parameters of ggplot2::geom_text() (if <code>geom = "text"</code>), ggplot2::geom_label() (if

	geom = "label"), <code>ggrepel::geom_text_repel()</code> (if geom = "text_repel") or <code>ggrepel::geom_label_repel()</code> (if geom = "label_repel").
method	The method used for the seasonal adjustment. "x13" (by default) for the X-13 method (using <code>rjd3x13::jx13()</code>) and "tramoseats" for TRAMO-SEATS (using <code>rjd3tramoseats::jtramoseats()</code>). Both methods are only compatible with series of frequency 12 (monthly), 6 (bimonthly), 4 (quarterly), and 2 (semi-annual). For other frequencies, the following methods are available: "x11-extended" (see <code>rjd3highfreq::x11plus()</code>), "fractionalairline" (see <code>rjd3highfreq::fractional_airline_decomposition()</code>), "multiairline" (see <code>rjd3highfreq::multi_airline_decomposition()</code>), "stl" (see <code>rjd3highfreq::stlplus()</code>), "istl" (see <code>rjd3highfreq::istl()</code>), and "mstl" (see <code>rjd3highfreq::mstl()</code>).
spec	The specification used for the seasonal adjustment. For "x13" and "tramoseats" methods, it can be the name of a pre-defined specification or a user-defined specification created with <code>rjd3x13::spec_x13()</code> or <code>rjd3tramoseats::spec_tramoseats()</code> . For other methods, it can be a list of parameters used in the associated seasonal adjustment function.
frequency	The frequency of the time series. By default (frequency = NULL), the frequency is computed automatically.
message	A logical value indicating whether a message is printed with the frequency used.
first_date	A numeric specifying the first date from which the outliers are plotted. By default (first_date = NULL), the outliers are plotted from the beginning of the time series.
last_date	A numeric specifying the last date up to which the outliers are plotted. By default (last_date = NULL), the outliers are plotted until the end of the time series.
coefficients	Logical indicating whether the estimated coefficients are printed. By default coefficients = FALSE.
digits	Integer indicating the number of decimal places to be used for the estimated coefficients of the outliers. By default digits = 1.
show.legend	Logical. Should this layer be included in the legends? NA, the default, includes it if any aesthetics are mapped. FALSE never includes it, and TRUE always includes it. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them.

Details

With geom = "text", the outliers used in the pre-adjustment process of the seasonal adjustment are directly added to the plot. With geom = "label", a rectangle is drawn behind the names of the outliers, making them easier to read. The same applies to geom = "text_repel" or geom = "label_repel", but text labels are also repelled away from each other and away from the data points (see `ggrepel::geom_label_repel()`).

Value

A **ggplot2** layer object to be added to a plot.

Examples

```
p_sa_ipi_fr <- ggplot(data = ipi_c_eu_df, mapping = aes(x = date, y = FR)) +
  geom_line() +
  labs(
    title = "Seasonal adjustment of the French industrial production index",
    x = "time", y = NULL
  ) +
  geom_sa(color = "red", message = FALSE)

# To add the outliers:
p_sa_ipi_fr + geom_outlier(
  geom = "label",
  message = FALSE
)

# To have a more readable plot with outlier names repelled away from each other
# and from the data points:
p_sa_ipi_fr +
  geom_outlier(
    geom = "label_repel",
    message = FALSE,
    vjust = 4,
    ylim = c(NA, 65), force = 10,
    arrow = arrow(
      length = unit(0.03, "npc"),
      type = "closed", ends = "last"
    )
  )

# To only plot the outliers from a specific date (2009):
p_sa_ipi_fr +
  geom_outlier(
    geom = "label_repel",
    message = FALSE,
    first_date = 2009,
    vjust = 4,
    ylim = c(NA, 65), force = 10,
    arrow = arrow(
      length = unit(0.03, "npc"),
      type = "closed", ends = "last"
    )
  )
```

geom_sa

Seasonal Adjustment Time Series

Description

Performs a seasonal adjustment and plots a specific component of the decomposition. `geom_sa()` and `stat_sa()` are aliases: they both use the same arguments. Use `stat_sa()` if you want to

display the results with a non-standard geom.

Usage

```
geom_sa(
  mapping = NULL,
  data = NULL,
  stat = "sa",
  position = "identity",
  ...,
  method = c("x13", "tramoseats", "x11-extended", "fractionalairline", "multiairline",
    "stl", "istl", "mstl"),
  spec = NULL,
  frequency = NULL,
  message = TRUE,
  component = "sa",
  show.legend = NA,
  inherit.aes = TRUE
)

stat_sa(
  mapping = NULL,
  data = NULL,
  geom = "line",
  position = "identity",
  ...,
  method = c("x13", "tramoseats", "x11-extended", "fractionalairline",
    "fractionalairlineestimation", "multiairline", "stl", "istl", "mstl"),
  spec = NULL,
  frequency = NULL,
  message = TRUE,
  component = "sa",
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping	Set of aesthetic mappings created by <code>ggplot2::aes()</code> . If specified and <code>inherit.aes</code> = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	A data.frame that contains the data used for the seasonal adjustment.
stat	The statistical transformation to use on the data for this layer, as a string.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
...	Other arguments passed on to <code>ggplot2::layer()</code> . These are often aesthetics, used to set an aesthetic to a fixed value, like <code>colour = "red"</code> or <code>size = 3</code> .

method	The method used for the seasonal adjustment. "x13" (by default) for the X-13 method (using <code>rjd3x13::jx13()</code>) and "tramoseats" for TRAMO-SEATS (using <code>rjd3tramoseats::jtramoseats()</code>). Both methods are only compatible with series of frequency 12 (monthly), 6 (bimonthly), 4 (quarterly), and 2 (semi-annual). For other frequencies, the following methods are available: "x11-extended" (see <code>rjd3highfreq::x11plus()</code>), "fractionalairline" (see <code>rjd3highfreq::fractional_airline_decomposition()</code>), "multiairline" (see <code>rjd3highfreq::multi_airline_decomposition()</code>), "stl" (see <code>rjd3highfreq::stlplus()</code>), "istl" (see <code>rjd3highfreq::istl()</code>), and "mstl" (see <code>rjd3highfreq::mstl()</code>).
spec	The specification used for the seasonal adjustment. For "x13" and "tramoseats" methods, it can be the name of a pre-defined specification or a user-defined specification created with <code>rjd3x13::spec_x13()</code> or <code>rjd3tramoseats::spec_tramoseats()</code> . For other methods, it can be a list of parameters used in the associated seasonal adjustment function.
frequency	The frequency of the time series. By default (frequency = NULL), the frequency is computed automatically.
message	A logical value indicating whether a message is printed with the frequency used.
component	A character string specifying the component to plot. The result must be a time series. See <code>rjd3x13::userdefined_variables_x13()</code> and <code>rjd3tramoseats::userdefined_variables_tramoseats()</code> for the available parameters for "x13" and "tramoseats". For other methods, the available components are the columns of the decomposition element of the result of the seasonal adjustment function (see documentation of the associated function for more details). By default (component = "sa"), the seasonally adjusted component is plotted.
show.legend	Logical. Should this layer be included in the legends? NA, the default, includes it if any aesthetics are mapped. FALSE never includes it, and TRUE always includes it. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them.
geom	The geometric object to use to display the data.

Value

A **ggplot2** layer object to be added to a plot.

Examples

```
p_ipi_fr <- ggplot(data = ipi_c_eu_df, mapping = aes(x = date, y = FR)) +
  geom_line() +
  labs(
    title = "Seasonal adjustment of the French industrial production index",
    x = "time", y = NULL
  )

# To add the seasonally adjusted series:
p_ipi_fr +
  geom_sa(color = "red")

# To add the forecasts of the input data and the seasonally adjusted series:
```

```
p_sa <- p_ipi_fr +
  geom_sa(component = "y_f", linetype = 2, message = FALSE) +
  geom_sa(component = "sa", color = "red", message = FALSE) +
  geom_sa(component = "sa_f", color = "red", linetype = 2, message = FALSE)
p_sa
```

ggplot_filters	<i>'ggplot2' Functions for 'rjd3filters' plots</i>
----------------	--

Description

Set of **ggplot2** plotting functions for filters and moving averages generated by the **rjd3filters** package.

Usage

```
ggplot_coef(x, zero_as_na = TRUE, ylab = "Coefficients", ...)

## S3 method for class 'finite_filters'
ggplot_coef(x, zero_as_na = TRUE, ylab = "Coefficients", q = 0, ...)

## S3 method for class 'moving_average'
ggplot_coef(x, zero_as_na = TRUE, ylab = "Coefficients", ...)

ggplot_gain(x, nxlab = 7, xlim = c(0, pi), ..., n = 101)

## S3 method for class 'finite_filters'
ggplot_gain(x, nxlab = 7, xlim = c(0, pi), q = 0, ..., n = 101)

## S3 method for class 'moving_average'
ggplot_gain(x, nxlab = 7, xlim = c(0, pi), ..., n = 101)

ggplot_phase(x, nxlab = 7, xlim = c(0, pi), normalized = FALSE, ..., n = 101)

## S3 method for class 'finite_filters'
ggplot_phase(
  x,
  nxlab = 7,
  xlim = c(0, pi),
  normalized = FALSE,
  q = 0,
  ...,
  n = 101
)

## S3 method for class 'moving_average'
ggplot_phase(x, nxlab = 7, xlim = c(0, pi), normalized = FALSE, ..., n = 101)
```

Arguments

x	coefficients, gain or phase.
zero_as_na	boolean indicating if the trailing zero of the coefficients should be plotted (FALSE) or removed (TRUE).
ylab	A character string specifying the y-axis label.
...	other arguments to <code>matplot</code> .
q	q.
nxlabel	number of xlabel.
xlim	vector containing x limits.
n	number of points used to plot the functions.
normalized	boolean indicating if the phase function is normalized by the frequency.

Value

A **ggplot2** object representing the coefficients, gain or phase of the filter.

Examples

```
filter <- rjd3filters::lp_filter(6, endpoints = "DAF", kernel = "Henderson")
ggplot_coef(filter, q = c(0,3))
ggplot_gain(filter, q = c(0,3))
ggplot_phase(filter, q = c(0,3))
```

init_ggplot

Initialise a 'ggplot2' Object with an SA Model

Description

Initialise a 'ggplot2' Object with an SA Model

Usage

```
init_ggplot(x, ...)
```

```
## S3 method for class 'JD3_X13_OUTPUT'
init_ggplot(x, context = NULL, ...)
```

```
## S3 method for class 'JD3_TRAMOSEATS_OUTPUT'
init_ggplot(x, context = NULL, ...)
```

```
## S3 method for class 'JD3_Object'
init_ggplot(x, ...)
```

```
## S3 method for class 'hf_decomposition'
init_ggplot(x, dates = NULL, ...)
```

Arguments

<code>x</code>	A seasonally adjusted model.
<code>...</code>	Other parameters passed to <code>ggplot2::ggplot()</code> .
<code>context</code>	The optional context used to estimate the model.
<code>dates</code>	Parameters used to re-estimate the model (see details).

Details

`init_ggplot()` initializes a **ggplot2** object by storing the seasonal adjustment model used.

Value

A **ggplot2** object that can be further customized.

Examples

```
mod <- rjd3x13::jx13(window(rjd3toolkit::ABS$X0.2.20.10.M, start = 2000))
init_ggplot(mod) +
  geom_line(color = "#F0B400") +
  geom_sa(component = "sa", color = "#155692")
```

 ipi_c_eu

Industrial Production Indices in manufacturing in the European Union

Description

A dataset containing on monthly industrial production indices in manufacturing in the European Union (from `sts_inpr_m` dataset of Eurostat). Data are based 100 in 2015 and are unadjusted, i.e. neither seasonally adjusted nor calendar adjusted.

Usage

```
ipi_c_eu
ipi_c_eu_df
```

Format

A monthly `ts` object from january 1990 to december 2017 with 34 variables for `ipi_c_eu` and a `data.frame` for `ipi_c_eu_df`.

An object of class `data.frame` with 360 rows and 35 columns.

Details

The dataset contains 34 time series corresponding to the following geographical area

BE	Belgium
BG	Bulgaria
CZ	Czechia
DK	Denmark
DE	Germany (until 1990 former territory of the FRG)
EE	Estonia
IE	Ireland
EL	Greece
ES	Spain
FR	France
HR	Croatia
IT	Italy
CY	Cyprus
LV	Latvia
LT	Lithuania
LU	Luxembourg
HU	Hungary
MT	Malta
NL	Netherlands
AT	Austria
PL	Poland
PT	Portugal
RO	Romania
SI	Slovenia
SK	Slovakia
FI	Finland
SE	Sweden
UK	United Kingdom
NO	Norway
CH	Switzerland
ME	Montenegro
MK	Former Yugoslav Republic of Macedonia, the
RS	Serbia
TR	Turkey
BA	Bosnia and Herzegovina

Source

Eurostat

Description

Computes and visualizes the Seasonal-Irregular (SI) ratio from a seasonal adjustment model (X-13 or TRAMO-SEATS).

Usage

```
siratio(x, ...)

siratioplot(
  x,
  labels = NULL,
  add = FALSE,
  box = TRUE,
  col.s = "darkblue",
  col.i = "gray",
  col.mean = "red",
  cex.i = 0.1,
  lwd.s = par("lwd"),
  lwd.mean = lwd.s,
  main = "SI ratio",
  xlab = NULL,
  ylab = NULL,
  xlim = NULL,
  ylim = NULL,
  ...
)

ggsiratioplot(
  x,
  labels = NULL,
  col.s = "darkblue",
  col.i = "gray",
  col.mean = "red",
  cex.i = 0.5,
  lwd.s = 1,
  lwd.mean = lwd.s,
  main = "SI ratio",
  xlab = NULL,
  ylab = NULL,
  ...
)
```

Arguments

x	An input model (X-13 or TRAMO-SEATS) or data object.
...	Additional arguments passed to methods or low-level plotting functions.
labels	Character vector of labels for the periods (e.g., months or quarters).
add	Logical indicating whether the plot should be added to an existing one.

`box` Logical indicating whether a box should be drawn around the plot.
`col.s, col.i, col.mean` Colors for the seasonal component, the SI points, and the mean lines.
`cex.i, lwd.s, lwd.mean` Graphical parameters for point size and line widths.
`main, xlab, ylab` Character strings for the main title, X-axis label, and Y-axis label.
`xlim, ylim` Numeric vectors of length 2 specifying the X and Y axis limits.

Value

- For `siratio()`: A `ts` object containing the SI ratio and the seasonal component.
- For `siratioplot()`: Called for its side effects (base 'R' plot).
- For `ggsiratioplot()`: A **ggplot2** object representing the SI ratio chart.

Examples

```
x <- rjd3x13::x13(ipi_c_eu[, "FR"])
siratioplot(x)
ggsiratioplot(x)
```

 ts2df

Convert a 'ts' or 'mts' Object to a 'data.frame'

Description

Converts a `ts` or `mts` object into a `data.frame` that can be directly used in plotting functions.

Usage

```
ts2df(x)

## S3 method for class 'ts'
ts2df(x)

## S3 method for class 'mts'
ts2df(x)
```

Arguments

`x` A `ts` or `mts` object.

Value

A `data.frame` object.

Examples

```
# To get the ipi_c_eu_df object:  
ts2df(ipi_c_eu)
```

usclaims	<i>US Claims</i>
----------	------------------

Description

Weekly US claims

Usage

```
usclaims
```

Format

An object of class `data.frame` with 2868 rows and 3 columns.

Index

* datasets

- [ipi_c_eu](#), [17](#)
 - [usclaims](#), [21](#)
- [autoplot.hf_decomposition](#)
 - [\(autoplot_ggdemetra3\)](#), [2](#)
- [autoplot.JD3_Object](#)
 - [\(autoplot_ggdemetra3\)](#), [2](#)
- [autoplot.JD3_TRAMOSEATS_OUTPUT](#)
 - [\(autoplot_ggdemetra3\)](#), [2](#)
- [autoplot.JD3_TRAMOSEATS_RSLTS](#)
 - [\(autoplot_ggdemetra3\)](#), [2](#)
- [autoplot.JD3_X13_OUTPUT](#)
 - [\(autoplot_ggdemetra3\)](#), [2](#)
- [autoplot.JD3_X13_RSLTS](#)
 - [\(autoplot_ggdemetra3\)](#), [2](#)
- [autoplot_ggdemetra3](#), [2](#)
- [calendar \(components\)](#), [4](#)
- [calendaradj \(components\)](#), [4](#)
- [components](#), [4](#)
- [geom_arima](#), [5](#)
- [geom_diagnostics](#), [7](#)
- [geom_outlier](#), [10](#)
- [geom_sa](#), [12](#)
- [ggplot2::aes\(\)](#), [6](#), [8](#), [10](#), [13](#)
- [ggplot2::geom_label\(\)](#), [6](#), [10](#)
- [ggplot2::geom_text\(\)](#), [6](#), [10](#)
- [ggplot2::ggplot\(\)](#), [17](#)
- [ggplot2::layer\(\)](#), [6](#), [8](#), [10](#), [13](#)
- [ggplot_coef \(ggplot_filters\)](#), [15](#)
- [ggplot_filters](#), [15](#)
- [ggplot_gain \(ggplot_filters\)](#), [15](#)
- [ggplot_phase \(ggplot_filters\)](#), [15](#)
- [ggrepel::geom_label_repel\(\)](#), [10](#), [11](#)
- [ggrepel::geom_text_repel\(\)](#), [10](#), [11](#)
- [ggsiratioplot \(siratio\)](#), [18](#)
- [gridExtra::ttheme_default\(\)](#), [8](#)
- [init_ggplot](#), [16](#)
- [ipi_c_eu](#), [17](#)
- [ipi_c_eu_df \(ipi_c_eu\)](#), [17](#)
- [irregular \(components\)](#), [4](#)
- [raw \(components\)](#), [4](#)
- [rjd3highfreq::fractional_airline_decomposition\(\)](#),
[3](#), [6](#), [8](#), [11](#), [14](#)
- [rjd3highfreq::istl\(\)](#), [4](#), [6](#), [8](#), [11](#), [14](#)
- [rjd3highfreq::mstl\(\)](#), [4](#), [6](#), [8](#), [11](#), [14](#)
- [rjd3highfreq::multi_airline_decomposition\(\)](#),
[3](#), [4](#), [6](#), [8](#), [11](#), [14](#)
- [rjd3highfreq::stlplus\(\)](#), [6](#), [8](#), [11](#), [14](#)
- [rjd3highfreq::x11plus\(\)](#), [6](#), [8](#), [11](#), [14](#)
- [rjd3tramoseats::jtramoseats\(\)](#), [6](#), [8](#), [11](#),
[14](#)
- [rjd3tramoseats::spec_tramoseats\(\)](#), [6](#), [8](#),
[11](#), [14](#)
- [rjd3tramoseats::tramoseats\(\)](#), [5](#), [7](#), [10](#)
- [rjd3tramoseats::userdefined_variables_tramoseats\(\)](#),
[8](#), [14](#)
- [rjd3x13::jx13\(\)](#), [6](#), [8](#), [11](#), [14](#)
- [rjd3x13::spec_x13\(\)](#), [6](#), [8](#), [11](#), [14](#)
- [rjd3x13::userdefined_variables_x13\(\)](#),
[8](#), [14](#)
- [rjd3x13::x13\(\)](#), [5](#), [7](#), [10](#)
- [seasonal \(components\)](#), [4](#)
- [seasonaladj \(components\)](#), [4](#)
- [siratio](#), [18](#)
- [siratioplot \(siratio\)](#), [18](#)
- [stat_sa \(geom_sa\)](#), [12](#)
- [trendcycle \(components\)](#), [4](#)
- [ts2df](#), [20](#)
- [usclaims](#), [21](#)