Package 'imputeTS'

October 13, 2022

Version 3.3

Date 2022-08-30

Title Time Series Missing Value Imputation

Description Imputation (replacement) of missing values
in univariate time series.
Offers several imputation functions
and missing data plots.
Available imputation algorithms include:
'Mean', 'LOCF', 'Interpolation',
'Moving Average', 'Seasonal Decomposition',
'Kalman Smoothing on Structural Time Series models',
'Kalman Smoothing on ARIMA models'. Published in Moritz and Bartz-Beielstein (2017)
<doi:10.32614/RJ-2017-009>.

Author Steffen Moritz [aut, cre, cph]
(<https://orcid.org/0000-0002-0085-1804>),
Sebastian Gatscha [aut],
Earo Wang [ctb] (<https://orcid.org/0000-0001-6448-5260>),
Ron Hause [ctb] (<https://orcid.org/0000-0002-5229-7366>)

Maintainer Steffen Moritz <steffen.moritz10@gmail.com>

LazyData yes

Type Package

ByteCompile TRUE

BugReports https://github.com/SteffenMoritz/imputeTS/issues

URL https://github.com/SteffenMoritz/imputeTS,
https://steffenmoritz.github.io/imputeTS/

Repository CRAN

Depends R (>= 3.6)

Imports stats, grDevices, ggplot2 (>= 3.3.0), ggtext, stinepack,
forecast, magrittr, methods, Rcpp

Suggests testthat, R.rsp, knitr, zoo, timeSeries, tis, xts, tibble,
tsibble, rmarkdown, covr

License GPL-3

VignetteBuilder R.rsp, knitr, rmarkdown

RoxygenNote 7.2.0

LinkingTo Rcpp

Encoding UTF-8

NeedsCompilation yes

Date/Publication 2022-09-09 06:52:55 UTC

R topics documented:

`ggplot_na_distribution` *Lineplot to Visualize the Distribution of Missing Values*

Description

Visualize the distribution of missing values within a time series.

Usage

```
ggplot_na_distribution(
  x,
  x_axis_labels = NULL,
  color_points = "steelblue",
  color_lines = "steelblue2",
  color_missing = "indianred",
  color_missing_border = "indianred",
  alpha_missing = 0.5,
  title = "Distribution of Missing Values",
  subtitle = "Time Series with highlighted missing regions",
  xlab = "Time",
  ylab = "Value",
  shape_points = 20,
  size_points = 2.5,
  theme = ggplot2::theme_linedraw()
)
```

Arguments

x	Numeric Vector (vector) or Time Series (ts) object containing NAs. This is the only mandatory parameter - all other parameters are only needed for adjusting the plot appearance.
x_axis_labels	For adding specific x-axis labels. Takes a vector of Date or POSIXct objects as an input (needs the same length as x) . The Default (NULL) uses the observation numbers as x-axis tick labels.
color_points	Color for the Symbols/Points.
color_lines	Color for the Lines.
color_missing	Color used for highlighting the time spans with NA values.
color_missing_border	Color used as border for time spans with NA values.
alpha_missing	Alpha (transparency) value used for color_missing.
title	Title of the Plot (NULL for deactivating title).
subtitle	Subtitle of the Plot (NULL for deactivating subtitle).
xlab	Label for x-Axis.
ylab	Label for y-Axis.
shape_points	Symbol to use for the Observations/Points. See https://ggplot2.tidyverse.org/articles/ggplot2-specs.html as reference.
size_points	Size of Symbols/Points.
theme	Set a Theme for ggplot2. Default is ggplot2::theme_linedraw(). (theme_linedraw)

Details

This function visualizes the distribution of missing values within a time series. If a value is NA, the background is colored differently. This gives a good overview of where most missing values occur.

The only really needed parameter for this function is x (the univariate time series that shall be visualized). All other parameters are solely for altering the appearance of the plot.

As long as the input is univariate and numeric the function also takes data.frame, tibble, tsibble, zoo, xts as an input.

The plot can be adjusted to your needs via the function parameters. Additionally, for more complex adjustments, the output can also be adjusted via ggplot2 syntax. This is possible, since the output of the function is a ggplot2 object. Also take a look at the Examples to see how adjustments are made.

For very long time series it might happen, that the plot gets too crowded and overplotting issues occur. In this case the ggplot_na_distribution2 plotting function can provide a more condensed overview.

Author(s)

Steffen Moritz, Sebastian Gatscha

See Also

ggplot_na_distribution2, ggplot_na_gapsize, ggplot_na_imputations

Examples

```
# Example 1: Visualize the missing values in x
x <- stats::ts(c(1:11, 4:9, NA, NA, NA, 11:15, 7:15, 15:6, NA, NA, 2:5, 3:7))
ggplot_na_distribution(x)

# Example 2: Visualize the missing values in tsAirgap time series
ggplot_na_distribution(tsAirgap)

# Example 3: Same as example 1, just written with pipe operator
x <- ts(c(1:11, 4:9, NA, NA, NA, 11:15, 7:15, 15:6, NA, NA, 2:5, 3:7))
x %>% ggplot_na_distribution()

# Example 4: Visualize NAs in tsAirgap - different color for points
# Plot adjustments via ggplot_na_distribution function parameters
ggplot_na_distribution(tsAirgap, color_points = "grey")

# Example 5: Visualize NAs in tsAirgap - different theme
# Plot adjustments via ggplot_na_distribution function parameters
ggplot_na_distribution(tsAirgap, theme = ggplot2::theme_classic())

# Example 6: Visualize NAs in tsAirgap - title, subtitle in center
# Plot adjustments via ggplot2 syntax
ggplot_na_distribution(tsAirgap) +
  ggplot2::theme(plot.title = ggplot2::element_text(hjust = 0.5)) +
  ggplot2::theme(plot.subtitle = ggplot2::element_text(hjust = 0.5))

# Example 7: Visualize NAs in tsAirgap - title in center, no subtitle
# Plot adjustments via ggplot2 syntax and function parameters
ggplot_na_distribution(tsAirgap, subtitle = NULL) +
  ggplot2::theme(plot.title = ggplot2::element_text(hjust = 0.5))
```

```
# Example 8: Visualize NAs in tsAirgap - x-axis texts with angle
# Plot adjustments via ggplot2 syntax and function parameters
ggplot_na_distribution(tsAirgap, color_points = "grey") +
  ggplot2::theme(axis.text.x = ggplot2::element_text(angle = 60, hjust = 1))
```

`ggplot_na_distribution2` *Stacked Barplot to Visualize Missing Values per Interval*

Description

Visualization of missing values in barplot form. Especially useful when looking at specific intervals and for time series with a lot of observations.

Usage

```
ggplot_na_distribution2(
  x,
  number_intervals = NULL,
  interval_size = NULL,
  measure = "percent",
  color_missing = "indianred2",
  color_existing = "steelblue",
  alpha_missing = 0.8,
  alpha_existing = 0.3,
  title = "Missing Values per Interval",
  subtitle = "Amount of NA and non-NA for successive intervals",
  xlab = "Time Lapse (Interval Size: XX)",
  ylab = NULL,
  color_border = "white",
  theme = ggplot2::theme_linedraw()
)
```

Arguments

`x`	Numeric Vector (`vector`) or Time Series (`ts`) object containing NAs. This is the only mandatory parameter - all other parameters are only needed for adjusting the plot appearance.
`number_intervals`	Defines the number of bins to be created. Default number of intervals (denoted by NULL) is calculated by `nclass.Sturges` using Sturges' formula. If the interval_size parameter is set to a value different to NULL this parameter is ignored.
`interval_size`	Defines how many observations should be in one bin/interval. The required number of overall bins is afterwards calculated automatically. If used this parameter overwrites the number_intervals parameter. For a very long time series

	be sure to make the interval_size not extremely small, otherwise because of overplotting issues nothing can be seen until you also increase the plot width.
measure	Whether the NA / non-NA ratio should be given as percent or absolute numbers. • "percent" - for percentages • "count" - for absolute numbers of NAs
color_missing	Color for the amount of missing values.
color_existing	Color for the amount of existing values.
alpha_missing	Alpha (transparency) value for the missing values.
alpha_existing	Alpha (transparency) value for the existing values.
title	Title of the Plot (NULL for deactivating title).
subtitle	Subtitle of the Plot (NULL for deactivating subtitle).
xlab	Label for x-Axis. Automatically set to the current interval size, if no custom text is chosen.
ylab	Label for y-Axis. As default (NULL), the axis is automatically set to either 'Percent' or 'Count' dependent on the settings of parameter measure.
color_border	Color for the small borders between the intervals/bins. Default is 'white'.
theme	Set a Theme for ggplot2. Default is ggplot2::theme_linedraw(). (theme_linedraw)

Details

This function visualizes the distribution of missing values within a time series. In comparison to the ggplot_na_distribution function this is not done by plotting each observation of the time series separately. Instead observations for time intervals are represented as intervals/bins of multiple values. For these intervals information about the amount of missing values are shown. This has the advantage, that also for large time series a plot which is easy to overview can be created.

The only really needed parameter for this function is x (the univariate time series that shall be visualized). All other parameters are solely for altering the appearance of the plot.

As long as the input is univariate and numeric the function also takes data.frame, tibble, tsibble, zoo, xts as an input.

The plot can be adjusted to your needs via the function parameters. Additionally, for more complex adjustments, the output can also be adjusted via ggplot2 syntax. This is possible, since the output of the function is a ggplot2 object. Also take a look at the Examples to see how adjustments are made.

Author(s)

Steffen Moritz, Sebastian Gatscha

See Also

ggplot_na_distribution, ggplot_na_gapsize, ggplot_na_imputations

Examples

```
# Example 1: Visualize the missing values in tsNH4 time series as percentages
ggplot_na_distribution2(tsNH4)

# Example 2: Visualize the missing values in tsNH4 time series as counts
ggplot_na_distribution2(tsNH4, measure = "count")

# Example 3: Visualize the missing values in tsHeating time series
ggplot_na_distribution2(tsHeating)

# Example 4: Same as example 1, just written with pipe operator
tsNH4 %>% ggplot_na_distribution2()

# Example 5: Visualize NAs in tsNH4 - exactly 8 intervals
ggplot_na_distribution2(tsNH4, number_intervals = 8)

# Example 6: Visualize NAs in tsNH4 - 300 observations per interval
ggplot_na_distribution2(tsNH4, interval_size = 300)

# Example 7: Visualize NAs in tsAirgap - different color for NAs
# Plot adjustments via ggplot_na_distribution2 function parameters
ggplot_na_distribution2(tsAirgap, color_missing = "pink")

# Example 8: Visualize NAs in tsNH4 - different theme
# Plot adjustments via ggplot_na_distribution2 function parameters
ggplot_na_distribution2(tsNH4, theme = ggplot2::theme_classic())

# Example 9: Visualize NAs in tsAirgap - title, subtitle in center
# Plot adjustments via ggplot2 syntax
ggplot_na_distribution2(tsAirgap) +
  ggplot2::theme(plot.title = ggplot2::element_text(hjust = 0.5)) +
  ggplot2::theme(plot.subtitle = ggtext::element_markdown(hjust = 0.5))

# Example 10: Visualize NAs in tsAirgap - title in center, no subtitle
# Plot adjustments via ggplot2 syntax and function parameters
ggplot_na_distribution2(tsAirgap, subtitle = NULL) +
  ggplot2::theme(plot.title = ggplot2::element_text(hjust = 0.5))

# Example 11: Visualize NAs in tsAirgap - x-axis texts with angle
# Plot adjustments via ggplot2 syntax and function parameters
ggplot_na_distribution2(tsAirgap, color_missing = "grey") +
  ggplot2::theme(axis.text.x = ggplot2::element_text(angle = 60, hjust = 1))
```

---

ggplot_na_gapsize *Visualize Occurrences of NA gap sizes*

Description

Visualize the Number of Occurrences for existing NA Gap Sizes (NAs in a row) in a Time Series

Usage

```
ggplot_na_gapsize(
  x,
  limit = 10,
  include_total = TRUE,
  ranked_by = "occurrence",
  color_occurrence = "indianred",
  color_total = "steelblue",
  color_border = "black",
  alpha_bars = 1,
  title = "Occurrence of gap sizes",
  subtitle = "Gap sizes (NAs in a row) ordered by most common",
  xlab = NULL,
  ylab = "Number occurrence",
  legend = TRUE,
  orientation = "horizontal",
  label_occurrence = "Number occurrence gapsize",
  label_total = "Resulting NAs for gapsize",
  theme = ggplot2::theme_linedraw()
)
```

Arguments

`x`	Numeric Vector (`vector`) or Time Series (`ts`) object containing NAs. This is the only mandatory parameter - all other parameters are only needed for adjusting the plot appearance.
`limit`	Specifies how many of the most common gap sizes are shown in the plot.Default is 10. So only the 10 most often occurring gapsizes will be shown. If more or all present gap sizes should be displayed, the limit needs to be increased. Since this might add a lot of additional data, having parameter `orientation` set to 'horizontal' avoids overlaps in the axis labels.
`include_total`	When set to TRUE the total NA count for a gapsize is included in the plot (total = number occurrence x gap size). E.g. if a gapsize of 3 occurs 10 times, this means this gap size makes up for 30 NAs in total. This can be a good indicator of the overall impact of a gapsize.
`ranked_by`	Should the results be sorted according to the number of occurrence or total resulting NAs for a gapsize. Total resulting NAs are calculated by (total = number occurrence x gap size). • "occurrence" - Sorting by 'number of occurrence' of a gap size • "total" - Sorting by 'total resulting NAs' of a gap size The default setting is "occurrence".
`color_occurrence`	Defines the Color for the Bars of 'number of occurrence'.
`color_total`	Defines the color for the bars of 'total resulting NAs'.
`color_border`	Defines the color for the border of the bars.
`alpha_bars`	Alpha (transparency) value used for filling the bars.

title	Title of the Plot.
subtitle	Subtitle of the Plot.
xlab	Label for x-Axis.
ylab	Label for y-Axis.
legend	If TRUE a legend is added at the bottom.
orientation	Can be either 'vertical' or 'horizontal'. Defines if the bars are plotted vertically or horizontally. For large amounts of different gap sizes horizontal illustration is favorable (also see parameter limit).
label_occurrence	Defines the label assigned to 'number of occurrence' in the legend.
label_total	Defines the label assigned to 'total resulting NAs' in the legend.
theme	Set a Theme for ggplot2. Default is ggplot2::theme_linedraw(). (theme_linedraw)

Details

This plotting function can be used to visualize the length of the NA gaps (NAs in a row) in a time series. It shows a ranking of which gap sizes occur most often. This ranking can be ordered by the number occurrence of the gap sizes or by total resulting NAs for this gap size (occurrence * gap length). A NA-gap of 3 occurring 10 times means 30 total resulting NAs.

A resulting plot can for example be described like this: a 2 NA-gap (2 NAs in a row) occurred 27 times, a 9 NA-gap (9 NAs in a row) occurred 11 times, a 27 NA-gap (27 NAs in a row) occurred 1 times, ...

The only really needed parameter for this function is x (the univariate time series with NAs that shall be visualized). All other parameters are solely for altering the appearance of the plot.

As long as the input is univariate and numeric, the function also takes data.frame, tibble, tsibble, zoo, xts as an input.

The plot can be adjusted to your needs via the function parameters. Additionally, for more complex adjustments, the output can also be adjusted via ggplot2 syntax. This is possible, since the output of the function is a ggplot2 object. Also take a look at the Examples to see how adjustments are made.

Value

The output is a ggplot2 object that can be further adjusted by using the ggplot syntax

Author(s)

Steffen Moritz, Sebastian Gatscha

See Also

ggplot_na_distribution, ggplot_na_distribution2, ggplot_na_imputations

Examples

```
# Example 1: Visualize the top gap sizes in tsNH4 (top 10 by default)
ggplot_na_gapsize(tsNH4)

# Example 2: Visualize the top gap sizes in tsAirgap - horizontal bars
ggplot_na_gapsize(tsAirgap, orientation = "vertical")

# Example 3: Same as example 1, just written with pipe operator
tsNH4 %>% ggplot_na_gapsize()

# Example 4: Visualize the top 20 gap sizes in tsNH4
ggplot_na_gapsize(tsNH4, limit = 20)

# Example 5: Visualize top gap sizes in tsNH4 without showing total NAs
ggplot_na_gapsize(tsNH4, limit = 20, include_total = FALSE)

# Example 6: Visualize top gap sizes in tsNH4 but ordered by total NAs
# (total = occurrence * gap length)
ggplot_na_gapsize(tsNH4, limit = 20, ranked_by = "total")

# Example 7: Visualize top gap sizes in tsNH4 - different theme
# Plot adjustments via ggplot_na_gapsize function parameters
ggplot_na_gapsize(tsNH4, theme = ggplot2::theme_classic())

# Example 8: Visualize top gap sizes in tsNH4 - title, subtitle in center
# Plot adjustments via ggplot2 syntax
ggplot_na_gapsize(tsNH4) +
  ggplot2::theme(plot.title = ggplot2::element_text(hjust = 0.5)) +
  ggplot2::theme(plot.subtitle = ggplot2::element_text(hjust = 0.5))

# Example 9: Visualize top gap sizes in tsNH4 - title in center, no subtitle
# Plot adjustments via ggplot2 syntax and function parameters
ggplot_na_gapsize(tsNH4, subtitle = NULL) +
  ggplot2::theme(plot.title = ggplot2::element_text(hjust = 0.5))

# Example 10: Top gap sizes in tsNH4 - legend on the right and color change
# Plot adjustments via ggplot2 syntax and function parameters
ggplot_na_gapsize(tsNH4, color_total = "grey") +
  ggplot2::theme(legend.position = "right")
```

---

ggplot_na_imputations *Visualize Imputed Values*

Description

Visualize the imputed values in a time series.

Usage

```
ggplot_na_imputations(
  x_with_na,
  x_with_imputations,
  x_with_truth = NULL,
  x_axis_labels = NULL,
  title = "Imputed Values",
  subtitle = "Visualization of missing value replacements",
  xlab = "Time",
  ylab = "Value",
  color_points = "steelblue",
  color_imputations = "indianred",
  color_truth = "seagreen3",
  color_lines = "lightslategray",
  shape_points = 16,
  shape_imputations = 18,
  shape_truth = 16,
  size_points = 1.5,
  size_imputations = 2.5,
  size_truth = 1.5,
  size_lines = 0.5,
  linetype = "solid",
  connect_na = TRUE,
  legend = TRUE,
  legend_size = 5,
  label_known = "known values",
  label_imputations = "imputed values",
  label_truth = "ground truth",
  theme = ggplot2::theme_linedraw()
)
```

Arguments

x_with_na	Numeric Vector or Time Series (ts) object with NAs before imputation. This parameter and x_with_imputation shave to be set. The rest of the parameters are mostly needed for adjusting the plot appearance.
x_with_imputations	Numeric Vector or Time Series (ts) object with NAs replaced by imputed values. This parameter and x_with_imputation shave to be set.The rest of the parameters are mostly needed for adjusting the plot appearance.
x_with_truth	Numeric Vector or Time Series (ts) object with the real values (optional parameter). If the ground truth is known (e.g. in experiments where the missing values were artificially added) it can be displayed in the plot with this parameter. Default is NULL (ground truth not known).
x_axis_labels	For adding specific x-axis labels. Takes a vector of Date or POSIXct objects as an input (needs the same length as x_with_na). The Default (NULL) uses the observation numbers as x-axis tick labels.

title	Title of the Plot.
subtitle	Subtitle of the Plot.
xlab	Label for x-Axis.
ylab	Label for y-Axis.
color_points	Color for the Symbols/Points of the non-NA Observations.
color_imputations	Color for the Symbols/Points of the Imputed Values.
color_truth	Color for the Symbols/Points of the NA value Ground Truth (only relevant when x_with_truth available).
color_lines	Color for the Lines connecting the Observations/Points.
shape_points	Shape for the Symbols/Points of the non-NA observations. See https://ggplot2.tidyverse.org/articles/ggplo specs.html as reference.
shape_imputations	Shape for the Symbols/Points of the imputed values. See https://ggplot2.tidyverse.org/articles/ggplot2-specs.html as reference.
shape_truth	Shape for the Symbols/Points of the NA value Ground Truth (only relevant when x_with_truth available).
size_points	Size for the Symbols/Points of the non-NA Observations.
size_imputations	Size for the Symbols/Points of the Imputed Values.
size_truth	Size for the Symbols/Points of the NA value Ground Truth (only relevant when x_with_truth available).
size_lines	Size for the Lines connecting the Observations/Points.
linetype	Linetype for the Lines connecting the Observations/Points.
connect_na	If TRUE the Imputations are connected to the non-NA observations in the plot. Otherwise there are no connecting lines between symbols in NA areas.
legend	If TRUE a Legend is added at the bottom.
legend_size	Size of the Symbols used in the Legend.
label_known	Legend label for the non-NA Observations.
label_imputations	Legend label for the Imputed Values.
label_truth	Legend label for the Ground Truth of the NA values.
theme	Set a Theme for ggplot2. Default is ggplot2::theme_linedraw(). (theme_linedraw)

Details

This plot can be used, to visualize imputed values for a time series. Imputed values (filled NA gaps) are shown in a different color than the other values. If real values (ground truth) for the NA gaps are known, they can be optionally added in a different color.

The only really needed parameters for this function are x_with_na (the time series with NAs before imputation) and x_with_imputations (the time series without NAs after imputation). All other parameters are msotly for altering the appearance of the plot.

As long as the input is univariate and numeric the function also takes data.frame, tibble, tsibble, zoo, xts as an input.

The plot can be adjusted to your needs via the function parameters. Additionally, for more complex adjustments, the output can also be adjusted via ggplot2 syntax. This is possible, since the output of the function is a ggplot2 object. Also take a look at the Examples to see how adjustments are made.

Author(s)

Steffen Moritz, Sebastian Gatscha

See Also

ggplot_na_distribution, ggplot_na_distribution2, ggplot_na_gapsize

Examples

```
# Example 1: Visualize imputation by na_mean
imp_mean <- na_mean(tsAirgap)
ggplot_na_imputations(tsAirgap, imp_mean)


# Example 2: Visualize imputation by na_locf and added ground truth
imp_locf <- na_locf(tsAirgap)
ggplot_na_imputations(x_with_na = tsAirgap,
                      x_with_imputations = imp_locf,
                      x_with_truth = tsAirgapComplete
                      )


# Example 3: Visualize imputation by na_kalman
imp_kalman <- na_kalman(tsAirgap)
ggplot_na_imputations(x_with_na = tsAirgap, x_with_imputations = imp_kalman)


# Example 4: Same as example 1, just written with pipe operator
tsAirgap %>%
  na_mean() %>%
  ggplot_na_imputations(x_with_na = tsAirgap)


# Example 5: Visualize imputation by na_seadec - different color for imputed points
# Plot adjustments via ggplot_na_imputations function parameters
imp_seadec <- na_seadec(tsAirgap)
ggplot_na_imputations(x_with_na = tsAirgap,
                      x_with_imputations = imp_seadec,
                      color_imputations = "gold")


# Example 6: Visualize imputation - different theme, point size imputations
# Plot adjustments via ggplot_na_imputations function parameters
imp_seadec <- na_seadec(tsAirgap)
ggplot_na_imputations(x_with_na = tsAirgap,
```

```
                      x_with_imputations = imp_seadec,
                      theme = ggplot2::theme_classic(),
                      size_imputations = 5)


# Example 7: Visualize imputation - title, subtitle in center
# Plot adjustments via ggplot2 syntax
imp_seadec <- na_seadec(tsAirgap)
ggplot_na_imputations(x_with_na = tsAirgap,  x_with_imputations = imp_seadec) +
    ggplot2::theme(plot.title = ggplot2::element_text(hjust = 0.5)) +
    ggplot2::theme(plot.subtitle = ggplot2::element_text(hjust = 0.5))


# Example 8: Visualize imputation - title in center, no subtitle
# Plot adjustments via ggplot2 syntax and function parameters
imp_mean <- na_mean(tsAirgap)
ggplot_na_imputations(x_with_na = tsAirgap,
                      x_with_imputations = imp_mean,
                      subtitle = NULL) +
     ggplot2::theme(plot.title = ggplot2::element_text(hjust = 0.5))
```

`na_interpolation` *Missing Value Imputation by Interpolation*

Description

Uses either linear, spline or stineman interpolation to replace missing values.

Usage

```
na_interpolation(x, option = "linear", maxgap = Inf, ...)
```

Arguments

`x`	Numeric Vector (`vector`) or Time Series (`ts`) object in which missing values shall be replaced
`option`	Algorithm to be used. Accepts the following input: • "linear" - for linear interpolation using approx (default choice) • "spline" - for spline interpolation using spline • "stine" - for Stineman interpolation using stinterp
`maxgap`	Maximum number of successive NAs to still perform imputation on. Default setting is to replace all NAs without restrictions. With this option set, consecutive NAs runs, that are longer than 'maxgap' will be left NA. This option mostly makes sense if you want to treat long runs of NA afterwards separately.
`...`	Additional parameters to be passed through to approx or spline interpolation functions

Details

Missing values get replaced by values of approx, spline or stinterp interpolation.

The na_interpolation function also supports the use of additional parameters from the respective underlying interpolation functions. While usually not really needed, it is useful to know that this advanced use is in principle possible. These additional parameters are not specified explicitly in the na_interpolation function documentation. Take a look into the documentation of the stinterp, approx and spline functions to get an overview about these additional parameters.

An example for such a parameter is the 'method' argument of spline, which can be used to further specify the type of spline to be used. Possible values are "fmm", "natural", "periodic", "monoH.FC" and "hyman" (as can be seen in the spline documentation). The respective function call using this additional parameter would look like this: `na_interpolation(x, option ="spline", method ="natural")`

Like in this example other additional detail parameters (gained from approx, spline, stinterp documentation) can be used by just including them in the na_interpolation function call. As already mentioned, these advanced possibilities for settings parameters are only helpful for specific use cases. For regular use the standard parameters provided directly in the na_interpolation documentation should be more than enough.

Value

Vector (`vector`) or Time Series (`ts`) object (dependent on given input at parameter x)

Author(s)

Steffen Moritz, Ron Hause

References

Johannesson, Tomas, et al. (2015). "Package stinepack".

See Also

`na_kalman`, `na_locf`, `na_ma`, `na_mean`, `na_random`, `na_replace`, `na_seadec`, `na_seasplit`

Examples

```
# Prerequisite: Create Time series with missing values
x <- ts(c(2, 3, 4, 5, 6, NA, 7, 8))

# Example 1: Perform linear interpolation
na_interpolation(x)

# Example 2: Perform spline interpolation
na_interpolation(x, option = "spline")

# Example 3: Perform stine interpolation
na_interpolation(x, option = "stine")

# Example 4: Perform linear interpolation, with additional parameter pass through from spline()
```

```
# Take a look at the 'Details' section of the na_interpolation documentation
# for more information about advanced parameter pass through options
na_interpolation(x, option ="spline", method ="natural")

# Example 5: Same as example 1, just written with pipe operator
x %>% na_interpolation()

# Example 6: Same as example 2, just written with pipe operator
x %>% na_interpolation(option = "spline")
```

`na_kalman` *Missing Value Imputation by Kalman Smoothing and State Space Models*

Description

Uses Kalman Smoothing on structural time series models (or on the state space representation of an arima model) for imputation.

Usage

```
na_kalman(x, model = "StructTS", smooth = TRUE, nit = -1, maxgap = Inf, ...)
```

Arguments

`x`	Numeric Vector (`vector`) or Time Series (`ts`) object in which missing values shall be replaced
`model`	Model to be used. With this parameter the State Space Model (on which KalmanSmooth is performed) can be chosen. Accepts the following input: • "StructTS" - For using a structural model fitted by maximum likelihood (using StructTS) (default choice) • "auto.arima" - For using the state space representation of arima model (using auto.arima) For both auto.arima and StructTS additional parameters for model building can be given with the ... parameter Additionally it is also possible to use a user created state space model (See code Example 5). This state space model could for example be obtained from another R package for structural time series modeling. Furthermore providing the state space representation of a arima model from arima is also possible. But it is important to note, that user created state space models must meet the requirements specified under KalmanLike. This means the user supplied state space model has to be in form of a list with at least components T, Z, h , V, a, P, Pn. (more details under KalmanLike)
`smooth`	if `TRUE` - `KalmanSmooth` is used for estimation, if `FALSE` - `KalmanRun` is used. Since KalmanRun is often considered extrapolation KalmanSmooth is usually the better choice for imputation.

nit	Parameter from Kalman Filtering (see KalmanLike). Usually no need to change from default.
maxgap	Maximum number of successive NAs to still perform imputation on. Default setting is to replace all NAs without restrictions. With this option set, consecutive NAs runs, that are longer than 'maxgap' will be left NA. This option mostly makes sense if you want to treat long runs of NA afterwards separately.
...	Additional parameters to be passed through to the functions that build the State Space Models (StructTS or auto.arima).

Details

The KalmanSmoother used in this function is `KalmanSmooth`. It operates either on a `Basic Structural Model` obtained by `StructTS` or the state space representation of a ARMA model obtained by `auto.arima`.

For an detailed explanation of Kalman Filtering and Space Space Models the following literature is a good starting point:

- *G. Welch, G. Bishop, An Introduction to the Kalman Filter. SIGGRAPH 2001 Course 8, 1995*
- *Harvey, Andrew C. Forecasting, structural time series models and the Kalman filter. Cambridge university press, 1990*
- *Grewal, Mohinder S. Kalman filtering. Springer Berlin Heidelberg, 2011*

Value

Vector (`vector`) or Time Series (`ts`) object (dependent on given input at parameter x)

Author(s)

Steffen Moritz

References

Hyndman RJ and Khandakar Y (2008). "Automatic time series forecasting: the forecast package for R". Journal of Statistical Software, 26(3).

See Also

`na_interpolation`, `na_locf`, `na_ma`, `na_mean`, `na_random`, `na_replace`, `na_seadec`, `na_seasplit`

Examples

```
# Example 1: Perform imputation with KalmanSmoother and state space representation of arima model
na_kalman(tsAirgap)

# Example 2: Perform imputation with KalmanRun and state space representation of arima model
na_kalman(tsAirgap, smooth = FALSE)

# Example 3: Perform imputation with KalmanSmooth and StructTS model
na_kalman(tsAirgap, model = "StructTS", smooth = TRUE)
```

```
# Example 4: Perform imputation with KalmanSmooth and StructTS model with additional parameters
na_kalman(tsAirgap, model = "StructTS", smooth = TRUE, type = "trend")

# Example 5:  Perform imputation with KalmanSmooth and user created model
usermodel <- arima(tsAirgap, order = c(1, 0, 1))$model
na_kalman(tsAirgap, model = usermodel)

# Example 6: Same as example 1, just written with pipe operator
tsAirgap %>% na_kalman()
```

`na_locf` *Missing Value Imputation by Last Observation Carried Forward*

Description

Replaces each missing value with the most recent present value prior to it (Last Observation Carried Forward- LOCF). Optionally this can also be done starting from the back of the series (Next Observation Carried Backward - NOCB).

Usage

```
na_locf(x, option = "locf", na_remaining = "rev", maxgap = Inf)
```

Arguments

`x`	Numeric Vector (`vector`) or Time Series (`ts`) object in which missing values shall be replaced
`option`	Algorithm to be used. Accepts the following input: • "locf" - for Last Observation Carried Forward (default choice) • "nocb" - for Next Observation Carried Backward
`na_remaining`	Method to be used for remaining NAs. • "rev" - to perform nocb / locf from the reverse direction (default choice) • "keep" - to return the series with NAs • "rm" - to remove remaining NAs • "mean" - to replace remaining NAs by overall mean
`maxgap`	Maximum number of successive NAs to still perform imputation on. Default setting is to replace all NAs without restrictions. With this option set, consecutive NAs runs, that are longer than 'maxgap' will be left NA. This option mostly makes sense if you want to treat long runs of NA afterwards separately.

Details

General Functionality:

Replaces each missing value with the most recent present value prior to it (Last Observation Carried Forward - LOCF). This can also be done in reverse direction, starting from the end of the series (then called Next Observation Carried Backward - NOCB).

Handling for NAs at the beginning of the series:

In case one or more successive observations directly at the start of the time series are NA, there exists no 'last value' yet, that can be carried forward. Thus, no LOCF imputation can be performed for these NAs. As soon as the first non-NA value appears, LOCF can be performed as expected. The same applies to NOCB, but from the opposite direction.

While this problem might appear seldom and will only affect a very small amount of values at the beginning, it is something to consider. The `na_remaining` parameter helps to define, what should happen with these values at the start, that would remain NA after pure LOCF.

Default setting is `na_remaining = "rev"`, which performs nocb / locf from the other direction to fill these NAs. So a NA at the beginning will be filled with the next non-NA value appearing in the series.

With `na_remaining = "keep"` NAs at the beginning (that can not be imputed with pure LOCF) are just left as remaining NAs.

With `na_remaining = "rm"` NAs at the beginning of the series are completely removed. Thus, the time series is basically shortened.

Also available is `na_remaining = "mean"`, which uses the overall mean of the time series to replace these remaining NAs. (but beware, mean is usually not a good imputation choice - even if it only affects the values at the beginning)

Value

Vector (`vector`) or Time Series (`ts`) object (dependent on given input at parameter x)

Author(s)

Steffen Moritz

See Also

`na_interpolation`, `na_kalman`, `na_ma`, `na_mean`, `na_random`, `na_replace`, `na_seadec`, `na_seasplit`

Examples

```
# Prerequisite: Create Time series with missing values
x <- ts(c(NA, 3, 4, 5, 6, NA, 7, 8))

# Example 1: Perform LOCF
na_locf(x)

# Example 2: Perform NOCF
na_locf(x, option = "nocb")

# Example 3: Perform LOCF and remove remaining NAs
```

```
na_locf(x, na_remaining = "rm")

# Example 4: Same as example 1, just written with pipe operator
x %>% na_locf()
```

---

na_ma *Missing Value Imputation by Weighted Moving Average*

Description

Missing value replacement by weighted moving average. Uses semi-adaptive window size to ensure all NAs are replaced.

Usage

```
na_ma(x, k = 4, weighting = "exponential", maxgap = Inf)
```

Arguments

x	Numeric Vector (vector) or Time Series (ts) object in which missing values shall be replaced
k	integer width of the moving average window. Expands to both sides of the center element e.g. k=2 means 4 observations (2 left, 2 right) are taken into account. If all observations in the current window are NA, the window size is automatically increased until there are at least 2 non-NA values present.
weighting	Weighting to be used. Accepts the following input: • "simple" - Simple Moving Average (SMA) • "linear" - Linear Weighted Moving Average (LWMA) • "exponential" - Exponential Weighted Moving Average (EWMA) (default choice)
maxgap	Maximum number of successive NAs to still perform imputation on. Default setting is to replace all NAs without restrictions. With this option set, consecutive NAs runs, that are longer than 'maxgap' will be left NA. This option mostly makes sense if you want to treat long runs of NA afterwards separately.

Details

In this function missing values get replaced by moving average values. Moving Averages are also sometimes referred to as "moving mean", "rolling mean", "rolling average" or "running average".

The mean in this implementation taken from an equal number of observations on either side of a central value. This means for an NA value at position i of a time series, the observations i-1,i+1 and i+1, i+2 (assuming a window size of k=2) are used to calculate the mean.

Since it can in case of long NA gaps also occur, that all values next to the central value are also NA, the algorithm has a semi-adaptive window size. Whenever there are less than 2 non-NA values in the complete window available, the window size is incrementally increased, till at least 2 non-NA values are there. In all other cases the algorithm sticks to the pre-set window size.

There are options for using Simple Moving Average (SMA), Linear Weighted Moving Average (LWMA) and Exponential Weighted Moving Average (EWMA).

SMA: all observations in the window are equally weighted for calculating the mean.

LWMA: weights decrease in arithmetical progression. The observations directly next to a central value i, have weight 1/2, the observations one further away (i-2,i+2) have weight 1/3, the next (i-3,i+3) have weight 1/4, ...

EWMA: uses weighting factors which decrease exponentially. The observations directly next to a central value i, have weight 1/2^1, the observations one further away (i-2,i+2) have weight 1/2^2, the next (i-3,i+3) have weight 1/2^3, ...

Value

Vector (vector) or Time Series (ts) object (dependent on given input at parameter x)

Author(s)

Steffen Moritz

See Also

na_interpolation, na_kalman, na_locf, na_mean, na_random, na_replace, na_seadec, na_seasplit

Examples

```
# Example 1: Perform imputation with simple moving average
na_ma(tsAirgap, weighting = "simple")

# Example 2: Perform imputation with exponential weighted moving average
na_ma(tsAirgap)

# Example 3: Perform imputation with exponential weighted moving average, window size 6
na_ma(tsAirgap, k = 6)

# Example 4: Same as example 1, just written with pipe operator
tsAirgap %>% na_ma(weighting = "simple")
```

---

na_mean *Missing Value Imputation by Mean Value*

Description

Missing value replacement by mean values. Different means like median, mean, mode possible.

Usage

```
na_mean(x, option = "mean", maxgap = Inf)
```

Arguments

x	Numeric Vector (vector) or Time Series (ts) object in which missing values shall be replaced
option	Algorithm to be used. Accepts the following input: • "mean" - take the mean for imputation (default choice) • "median" - take the median for imputation • "mode" - take the mode for imputation • "harmonic" - take the harmonic mean • "geometric" - take the geometric mean
maxgap	Maximum number of successive NAs to still perform imputation on. Default setting is to replace all NAs without restrictions. With this option set, consecutive NAs runs, that are longer than 'maxgap' will be left NA. This option mostly makes sense if you want to treat long runs of NA afterwards separately.

Details

Missing values get replaced by overall mean values. The function calculates the mean, median, mode, harmonic or geometric mean over all the non-NA values and replaces all NAs with this value. Option 'mode' replaces NAs with the most frequent value in the time series. If two or more values occur equally frequent, the function imputes the lower value. Due to their calculation formula geometric and harmonic mean are not well defined for negative values or zero values in the input series.

In general using the mean for imputation imputation is mostly a suboptimal choice and should be handled with great caution.

Value

Vector (vector) or Time Series (ts) object (dependent on given input at parameter x)

Author(s)

Steffen Moritz

See Also

na_interpolation, na_kalman, na_locf, na_ma, na_random, na_replace, na_seadec, na_seasplit

Examples

```
# Prerequisite: Create Time series with missing values
x <- ts(c(2, 3, 4, 5, 6, NA, 7, 8))

# Example 1: Perform imputation with the overall mean
na_mean(x)

# Example 2: Perform imputation with overall median
na_mean(x, option = "median")
```

```
# Example 3: Same as example 1, just written with pipe operator
x %>% na_mean()
```

---

na_random *Missing Value Imputation by Random Sample*

Description

Replaces each missing value by drawing a random sample between two given bounds.

Usage

```
na_random(x, lower_bound = NULL, upper_bound = NULL, maxgap = Inf)
```

Arguments

x	Numeric Vector (vector) or Time Series (ts) object in which missing values shall be replaced
lower_bound	Lower bound for the random samples. If nothing or NULL is set min(x) will be used.
upper_bound	Upper bound for the random samples. If nothing or NULL is set man(x) will be used.
maxgap	Maximum number of successive NAs to still perform imputation on. Default setting is to replace all NAs without restrictions. With this option set, consecutive NAs runs, that are longer than 'maxgap' will be left NA. This option mostly makes sense if you want to treat long runs of NA afterwards separately.

Details

Replaces each missing value by drawing a random sample between two given bounds. The default bounds are the minimum and the maximum value in the non-NAs from the time series. Function uses runif function to get the random values.

Value

Vector (vector) or Time Series (ts) object (dependent on given input at parameter x)

Author(s)

Steffen Moritz

See Also

na_interpolation, na_kalman, na_locf, na_ma, na_mean, na_replace, na_seadec, na_seasplit

Examples

```
# Prerequisite: Create Time series with missing values
x <- ts(c(2, 3, NA, 5, 6, NA, 7, 8))

# Example 1: Replace all NAs by random values that are between min and max of the input time series
na_random(x)

# Example 2: Replace all NAs by random values between 1 and 10
na_random(x, lower_bound = 1, upper_bound = 10)

# Example 3: Same as example 1, just written with pipe operator
x %>% na_random()
```

---

na_remove *Remove Missing Values*

Description

Removes all missing values from a time series.

Usage

```
na_remove(x)
```

Arguments

x	Numeric Vector (vector) or Time Series (ts) object in which missing values shall be replaced

Details

Removes all missing values from a input time series. This shortens the time series by the number of missing values in the series. Should be handled with care, because this can affect the seasonality of the time series. Seasonal patterns might be destroyed. Independent from the input, this function only returns a vector. (the time information of a resulting time series object wouldn't be correct any more).

Value

Vector (vector)

Author(s)

Steffen Moritz

See Also

na_interpolation, na_kalman, na_locf, na_ma, na_mean, na_random, na_replace, na_seadec, na_seasplit

Examples

```
# Example 1: Remove all NAs
# Create Time series with missing values
x <- ts(c(2, 3, NA, 5, 6, NA, 7, 8))

# Example 1: Remove all NAs
na_remove(x)

# Example 2: Remove all NAs in tsAirgap
na_remove(tsAirgap)

# Example 3: Same as example 1, just written with pipe operator
x %>% na_remove()
```

---

na_replace *Replace Missing Values by a Defined Value*

Description

Replaces all missing values with a given value.

Usage

```
na_replace(x, fill = 0, maxgap = Inf)
```

Arguments

x	Numeric Vector (vector) or Time Series (ts) object in which missing values shall be replaced
fill	Value used to replace the missing values
maxgap	Maximum number of successive NAs to still perform imputation on. Default setting is to replace all NAs without restrictions. With this option set, consecutive NAs runs, that are longer than 'maxgap' will be left NA. This option mostly makes sense if you want to treat long runs of NA afterwards separately.

Value

Vector (vector) or Time Series (ts) object (dependent on given input at parameter x)

Author(s)

Steffen Moritz

See Also

na_interpolation, na_kalman, na_locf, na_ma, na_mean, na_random, na_seadec, na_seasplit

Examples

```
# Prerequisite: Create Time series with missing values
x <- ts(c(2, 3, NA, 5, 6, NA, 7, 8))

# Example 1: Replace all NAs with 3.5
na_replace(x, fill = 3.5)

# Example 2: Replace all NAs with 0
na_replace(x, fill = 0)

# Example 3: Same as example 1, just written with pipe operator
x %>% na_replace(fill = 3.5)
```

---

na_seadec *Seasonally Decomposed Missing Value Imputation*

Description

Removes the seasonal component from the time series, performs imputation on the deseasonalized series and afterwards adds the seasonal component again.

Usage

```
na_seadec(
  x,
  algorithm = "interpolation",
  find_frequency = FALSE,
  maxgap = Inf,
  ...
)
```

Arguments

x	Numeric Vector (vector) or Time Series (ts) object in which missing values shall be replaced
algorithm	Algorithm to be used after decomposition. Accepts the following input: • "interpolation" - Imputation by Interpolation (default choice) • "locf" - Imputation by Last Observation Carried Forward • "mean" - Imputation by Mean Value • "random" - Imputation by Random Sample • "kalman" - Imputation by Kalman Smoothing and State Space Models • "ma" - Imputation by Weighted Moving Average
find_frequency	If TRUE the algorithm will try to estimate the frequency of the time-series automatically.

maxgap	Maximum number of successive NAs to still perform imputation on. Default setting is to replace all NAs without restrictions. With this option set, consecutive NAs runs, that are longer than 'maxgap' will be left NA. This option mostly makes sense if you want to treat long runs of NA afterwards separately.
...	Additional parameters for these algorithms that can be passed through. Look at na_interpolation, na_locf, na_random, na_mean for parameter options.

Details

The algorithm first performs a Seasonal Decomposition of Time Series by Loess via stl. Decomposing the time series into seasonal, trend and irregular components. The seasonal component gets then removed (subtracted) from the original series. As a second step the selected imputation algorithm e.g. na_locf, na_ma, ... is applied on the deseasonalized series. Thus, the algorithm can work without being affected by seasonal patterns. After filling the NA gaps, the seasonal component is added to the deseasonalized series again.

Implementation details: A paper about the STL Decomposition procedure is linked in the references. Since the function only works with complete data, the initial NA data is temporarily filled via linear interpolation in order to perform the decomposition. These temporarily imputed values are replaced with NAs again after obtaining the decomposition for the non-NA observations. STL decomposition is run with robust = TRUE and s.window = 11. Additionally, applying STL decomposition needs a preset frequency. This can be passed by the frequency set in the input ts object or by setting 'find_frequency=TRUE' in order to find an appropriate frequency for the time series. The find_frequency parameter internally uses findfrequency, which does a spectral analysis of the time series for identifying a suitable frequency. Using find_frequency will update the previously set frequency of a ts object to the newly found frequency. The default is 'find_frequency = FALSE', which gives a warning if no seasonality is set for the supplied time series object. If neither seasonality is set nor find_frequency is set to TRUE, the function goes on without decomposition and just applies the selected secondary algorithm to the original time series that still includes seasonality.

Value

Vector (vector) or Time Series (ts) object (dependent on given input at parameter x)

Author(s)

Steffen Moritz

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. Journal of Official Statistics, 6, 3–73.

See Also

na_interpolation, na_kalman, na_locf, na_ma, na_mean, na_random, na_replace, na_seasplit

Examples

```
# Example 1: Perform seasonal imputation using algorithm = "interpolation"
na_seadec(tsAirgap, algorithm = "interpolation")

# Example 2: Perform seasonal imputation using algorithm = "mean"
na_seadec(tsAirgap, algorithm = "mean")

# Example 3: Same as example 1, just written with pipe operator
tsAirgap %>% na_seadec(algorithm = "interpolation")
```

---

na_seasplit *Seasonally Splitted Missing Value Imputation*

Description

Splits the times series into seasons and afterwards performs imputation separately for each of the resulting time series datasets (each containing the data for one specific season).

Usage

```
na_seasplit(
  x,
  algorithm = "interpolation",
  find_frequency = FALSE,
  maxgap = Inf,
  ...
)
```

Arguments

x	Numeric Vector (vector) or Time Series (ts) object in which missing values shall be replaced
algorithm	Algorithm to be used after splits. Accepts the following input: • "interpolation" - Imputation by Interpolation (default choice) • "locf" - Imputation by Last Observation Carried Forward • "mean" - Imputation by Mean Value • "random" - Imputation by Random Sample • "kalman" - Imputation by Kalman Smoothing and State Space Models • "ma" - Imputation by Weighted Moving Average
find_frequency	If TRUE the algorithm will try to estimate the frequency of the time-series automatically.
maxgap	Maximum number of successive NAs to still perform imputation on. Default setting is to replace all NAs without restrictions. With this option set, consecutive NAs runs, that are longer than 'maxgap' will be left NA. This option mostly makes sense if you want to treat long runs of NA afterwards separately.
...	Additional parameters for these algorithms that can be passed through. Look at na_interpolation, na_locf, na_random, na_mean for parameter options.

Value

Vector (vector) or Time Series (ts) object (dependent on given input at parameter x)

Author(s)

Steffen Moritz

See Also

na_interpolation, na_kalman, na_locf, na_ma, na_mean, na_random, na_replace, na_seadec

Examples

```
# Example 1: Perform seasonal splitted imputation using algorithm = "interpolation"
na_seasplit(tsAirgap, algorithm = "interpolation")

# Example 2: Perform seasonal splitted imputation using algorithm = "mean"
na_seasplit(tsAirgap, algorithm = "mean")

# Example 3: Same as example 1, just written with pipe operator
tsAirgap %>% na_seasplit(algorithm = "interpolation")
```

---

statsNA *Print Statistics about Missing Values*

Description

Print summary stats about the distribution of missing values in a univariate time series.

Usage

```
statsNA(x, bins = 4, print_only = TRUE)
```

Arguments

x	Numeric Vector (vector) or Time Series (ts) object containing NAs
bins	Split number for bin stats. Number of bins the time series gets divided into. For each bin information about amount/percentage of missing values is printed. Default value is 4 - what means stats about the 1st,2nd,3rd,4th quarter of the time series are shown.
print_only	Choose if the function Prints or Returns. For print_only = TRUE the function has no return value and just prints out missing value stats. If print_only is changed to FALSE, nothing is printed and the function returns a list.Print gives a little bit more information, since the returned list does not include "Stats for Bins" and "overview NA series"

Details

Prints the following information about the missing values in the time series:

- "Length of time series" - Number of observations in the time series (including NAs)
- "Number of Missing Values" - Number of missing values in the time series
- "Percentage of Missing Values" - Percentage of missing values in the time series
- "Number of Gaps" - Number of NA gaps (consisting of one or more consecutive NAs) in the time series
- "Average Gap Size" - Average size of consecutive NAs for the NA gaps in the time series
- "Stats for Bins" - Number/percentage of missing values for the split into bins
- "Longest NA gap" - Longest series of consecutive missing values (NAs in a row) in the time series
- "Most frequent gap size" - Most frequent occurring series of missing values in the time series
- "Gap size accounting for most NAs" - The series of consecutive missing values that accounts for most missing values overall in the time series
- "Overview NA series" - Overview about how often each series of consecutive missing values occurs. Series occurring 0 times are skipped

It is furthermore, important to note, that you are able to choose whether the function returns a list or prints the information only. (see description of parameter "print_only")

Value

A `list` containing the stats. Beware: Function gives only a return value if print_only = FALSE.

Author(s)

Steffen Moritz

See Also

`ggplot_na_distribution`, `ggplot_na_distribution2`, `ggplot_na_gapsize`

Examples

```
# Example 1: Print stats about the missing data in tsNH4
statsNA(tsNH4)

# Example 2: Return list with stats about the missing data in tsAirgap
statsNA(tsAirgap, print_only = FALSE)

# Example 3: Same as example 1, just written with pipe operator
tsNH4 %>% statsNA()
```

`tsAirgap` *Time series of monthly airline passengers (with NAs)*

Description

Monthly totals of international airline passengers, 1949 to 1960. This time series contains missing values. In the package included is also the `tsAirgapComplete` time series providing the true values for the missing values.

Usage

```
tsAirgap
```

Format

Time Series (`ts`) with 144 rows including 13 NAs.

Details

The dataset originates from Box and Jenkins (see citation) and is a commonly used example in time series analysis literature.

It characteristics (strong trend, strong seasonal behavior) make it also a great example for time series imputation. Thus the version with inserted NA gaps was created under the name tsAirgap.

In order to use this series for comparing imputation algorithm results, there are two time series provided. One series without missing values, which can be used as ground truth. Another series with NAs, on which the imputation algorithms can be applied.

There are the two time series:

- tsAirgap - The time series with NAs.
- tsAirgapComplete - Time series without NAs.

Source

Box, G. E. P., Jenkins, G. M., Reinsel, G. C. and Ljung, G. M. (2015). Time series analysis: forecasting and control. Fifth Edition. John Wiley and Sons.

See Also

`tsHeating`, `tsNH4`

`tsAirgapComplete` *Time series of monthly airline passengers (complete)*

Description

Monthly totals of international airline passengers, 1949 to 1960. This time series provides the truth for the missing values of the `tsAirgap` time series. Thus it is identical to the tsAirgap time series except that no value is missing.

Usage

```
tsAirgapComplete
```

Format

Time Series (`ts`) with 144 rows.

Details

The dataset originates from Box and Jenkins (see citation) and is a commonly used example in time series analysis literature.

It characteristics (strong trend, strong seasonal behavior) make it also a great example for time series imputation. Thus the version with inserted NA gaps was created under the name tsAirgap.

In order to use this series for comparing imputation algorithm results, there are two time series provided. One series without missing values, which can be used as ground truth. Another series with NAs, on which the imputation algorithms can be applied.

There are the two time series:

- tsAirgap - The time series with NAs.
- tsAirgapComplete - Time series without NAs.

Source

Box, G. E. P., Jenkins, G. M., Reinsel, G. C. and Ljung, G. M. (2015). Time series analysis: forecasting and control. Fifth Edition. John Wiley and Sons.

See Also

`tsHeating`, `tsNH4`

`tsHeating` *Time series of a heating systems supply temperature (with NAs)*

Description

Time series of a heating systems supply temperature. Measured from 18.11.2013 - 05:12:00 to 13.01.2015 - 15:08:00 in 1 minute steps. This time series contains missing values. In the package included is also the `tsHeatingComplete` time series providing the true values for the missing values.

Usage

```
tsHeating
```

Format

Time Series (`ts`) with 606837 rows including 57391 NAs.

Details

The time series originates from the GECCO Industrial Challenge 2015. This Challenge was about "Recovering missing information in heating system operating data". Goal was to impute missing values in heating system sensor data as accurate as possible. (doi: 10.5281/zenodo.3884899)

In order to use this series for comparing imputation algorithm results, there are two time series provided. One series without missing values, which can be used as ground truth. Another series with NAs, on which the imputation algorithms can be applied. The NAs thereby were inserted according to patterns found in similar time series.

There are the two time series:

- tsHeating - The time series with NAs.
- tsHeatingComplete - Time series without NAs.

Source

Moritz, Steffen, Friese, Martina, Fischbach, Andreas, Schlitt, Christopher, and Bartz-Beielstein, Thomas. (2015, May 1). GECCO Industrial Challenge 2015 Dataset: A heating system dataset for the 'Recovering missing information in heating system operating data' competition at the Genetic and Evolutionary Computation Conference 2015, Madrid, Spain. http://doi.org/10.5281/zenodo.3884899

See Also

`tsAirgap`, `tsNH4`

`tsHeatingComplete` *Time series of a heating systems supply temperature (complete)*

Description

Time series of a heating systems supply temperature. Measured from 18.11.2013 - 05:12:00 to 13.01.2015 - 15:08:00 in 1 minute steps. This time series provides the truth for the missing values of the `tsHeating` time series. Thus it is identical to the heating time series except that no value is missing.

Usage

```
tsHeatingComplete
```

Format

Time Series (`ts`) with 606837 rows.

Details

The time series originates from the GECCO Industrial Challenge 2015. This Challenge was about "Recovering missing information in heating system operating data". Goal was to impute missing values in heating system sensor data as accurate as possible. (doi: 10.5281/zenodo.3884899)

In order to use this series for comparing imputation algorithm results, there are two time series provided. One series without missing values, which can be used as ground truth. Another series with NAs, on which the imputation algorithms can be applied. The NAs thereby were inserted according to patterns found in similar time series.

There are the two time series:

- tsHeating - The time series with NAs.
- tsHeatingComplete - Time series without NAs.

Source

Moritz, Steffen, Friese, Martina, Fischbach, Andreas, Schlitt, Christopher, and Bartz-Beielstein, Thomas. (2015, May 1). GECCO Industrial Challenge 2015 Dataset: A heating system dataset for the 'Recovering missing information in heating system operating data' competition at the Genetic and Evolutionary Computation Conference 2015, Madrid, Spain. http://doi.org/10.5281/zenodo.3884899

See Also

`tsAirgap`, `tsNH4`

tsNH4 *Time series of NH4 concentration in a wastewater system (with NAs)*

Description

Time series of NH4 concentration in a wastewater system. Measured from 30.11.2010 - 16:10 to 01.01.2011 - 6:40 in 10 minute steps. This time series contains missing values. In the package included is also the tsNH4Complete time series providing the true values for the missing values.

Usage

```
tsNH4
```

Format

Time Series (ts) with 4552 rows including 883 NAs.

Details

The time series is derived from the dataset of the GECCO Industrial Challenge 2014.

In order to use this series for comparing imputation algorithm results, there are two time series provided. One series without missing values, which can be used as ground truth. Another series with NAs, on which the imputation algorithms can be applied. The NAs thereby were inserted according to patterns found in similar time series.

There are the two time series:

- tsNH4 - The time series with NAs.
- tsNH4Complete - Time series without NAs.

Source

Friese, Martina, Fischbach, Andreas, Flasch, Oliver, Mersmann, Olaf, Bartz-Beielstein, Thomas, and Walbeck, Klaus. (2014, July 16). GECCO Industrial Challenge 2014 Dataset: A water quality dataset for the 'Active protection against pollution of the surface water' competition at the Genetic and Evolutionary Computation Conference 2015, Vancouver, Canada. http://www.spotseven.de/gecco-challenge/gecco-challenge-2014

See Also

tsAirgap,tsHeating

`tsNH4Complete` *Time series of NH4 concentration in a wastewater system (complete)*

Description

Time series of NH4 concentration in a wastewater system. Measured from 30.11.2010 - 16:10 to 01.01.2011 - 6:40 in 10 minute steps. This time series provides the truth for the missing values of the `tsNH4` time series. Thus it is identical to the heating time series except that no value is missing.

Usage

```
tsNH4Complete
```

Format

Time Series (`ts`) with 4552 rows.

Details

The time series is derived from the dataset of the GECCO Industrial Challenge 2014.

In order to use this series for comparing imputation algorithm results, there are two time series provided. One series without missing values, which can be used as ground truth. Another series with NAs, on which the imputation algorithms can be applied. The NAs thereby were inserted according to patterns found in similar time series.

There are the two time series:

- tsNH4 - The time series with NAs.
- tsNH4Complete - Time series without NAs.

Source

Friese, Martina, Fischbach, Andreas, Flasch, Oliver, Mersmann, Olaf, Bartz-Beielstein, Thomas, and Walbeck, Klaus. (2014, July 16). GECCO Industrial Challenge 2014 Dataset: A water quality dataset for the 'Active protection against pollution of the surface water' competition at the Genetic and Evolutionary Computation Conference 2015, Vancouver, Canada. http://www.spotseven.de/gecco-challenge/gecco-challenge-2014#'

See Also

`tsAirgap`,`tsHeating`

Index