Package ‘sweep’

Type Package

Title Tidy Tools for Forecasting

Version 0.2.5

Maintainer Matt Dancho <mdancho@business-science.io>

Description Tidies up the forecasting modeling and prediction work flow, extends the 'broom' package with 'sw_tidy', 'sw_glance', 'sw_augment', and 'sw_tidy_decomp' functions for various forecasting models, and enables converting 'forecast' objects to `tidy` data frames with 'sw_sweep'.

URL https://github.com/business-science/sweep

BugReports https://github.com/business-science/sweep/issues

License GPL (>= 3)

Encoding UTF-8

LazyData true

Depends R (>= 3.3.0)

Imports broom (>= 0.5.6), dplyr (>= 1.0.0), forecast (>= 8.0), lubridate (>= 1.6.0), tibble (>= 1.2), tidyr (>= 1.0.0), timetk (>= 2.1.0), rlang, tidyverse, tidyquant

Suggests forcats, knitr, rmarkdown, testthat, purrr, readr, stringr, scales, fracdiff

RoxygenNote 7.2.3

VignetteBuilder knitr

NeedsCompilation no

Author Matt Dancho [aut, cre], Davis Vaughan [aut]

Repository CRAN

Date/Publication 2023-07-06 12:40:02 UTC
**add_index**

Adds a sequential index column to a data frame

**Description**

Adds a sequential index column to a data frame

**Usage**

```
add_index(ret, rename_index)
```

**Arguments**

- `ret`: An object of class tibble
- `rename_index`: A variable indicating the index name to be used in the tibble returned
**arima_string**

*Print the ARIMA model parameters*

**Description**

Refer to forecast::arima.string. forecast arima.R

**Usage**

`arima_string(object, padding = FALSE)`

**Arguments**

- **object**: An object of class Arima
- **padding**: Add padding to the name returned

---

**bats_string**

*Print the BATS model parameters*

**Description**

Refer to forecast::makeText. forecast bats.R

**Usage**

`bats_string(object)`

**Arguments**

- **object**: An object of class bats

---

**bike_sales**

*Fictional sales data for bike shops purchasing Cannondale bikes*

**Description**

A dataset containing the fictional bicycle orders spanning 2011 through 2015. Hypothetically, the bike_sales data are similar to sales data maintained in a business’ sales data base. The unit price and model names come from data provided by model for the bicycle manufacturer, Cannondale (2016). The customers (bicycle shops) including name, location, etc and the orders including quantity purchased and order dates are fictional. The data is intended for implementing business analytics techniques (e.g. forecast, clustering, etc) to identify underlying trends.
Usage

bike_sales

Format

A data frame with 15644 rows and 17 variables:

- `order.date` Date the order was placed
- `order.id` A unique order identification number
- `order.line` The sequential identification number for products on and order
- `quantity` Number of units purchased
- `price` The unit price of the bicycle
- `price.ext` The extended price = price x quantity
- `customer.id` A unique customer identification number
- `bikeshop.name` The customer name
- `bikeshop.city` The city that the bike shop is located
- `bikeshop.state` The state that the bike shop is located
- `latitude` The geographic latitude of the customer location
- `longitude` The geographic longitude of the customer location
- `product.id` A unique product identification number
- `model` The model name of the bicycle
- `category.primary` The main bicycle category, either "Mountain" or "Road"
- `category.secondary` One of nine more specific bicycle categories
- `frame` The bicycle frame material, either "Carbon" or "Aluminum"

Source

The 2016 bicycle model names and prices originated from [https://www.cannondale.com/en-us](https://www.cannondale.com/en-us)

sweep_package

---

sweep: Extending broom to time series forecasting

Description

The sweep package "tidies" up the modeling workflow of the forecast package.

Details

The model and forecast objects are not covered by the broom package. It includes the `sw_tidy()`, `sw_glance()`, and `sw_augment()` functions that work in a similar capacity as broom functions. In addition, it provides `sw_tidy_decomp()` to tidy decompositions, and `sw_sweep()` to coerce forecast objects to "tibbles" for easy visualization with ggplot2 and manipulation with dplyr. To learn more about sweep, start with the vignettes: browseVignettes(package = "sweep")
**sw_augment**

Augment data according to a tidied model

**Description**

Given an R statistical model or other non-tidy object, add columns to the original dataset such as predictions, residuals and cluster assignments.

**Usage**

```
sw_augment(x, ...)
```

**Arguments**

- `x` model or other R object to convert to data frame
- `...` other arguments passed to methods

**Details**

`sw_augment()` is a wrapper for `broom::augment()`. The benefit of `sw_augment` is that it has methods for various time-series model classes such as `HoltWinters`, `ets`, `Arima`, etc.

For non-time series, `sw_augment()` defaults to `broom::augment()`. The only difference is that the return is a tibble.

Note that by convention the first argument is almost always `data`, which specifies the original data object. This is not part of the S3 signature, partly because it prevents `rowwise_df_tidiers` from taking a column name as the first argument.

**See Also**

`broom::augment()`

---

**sw_augment.default**

Default augment method

**Description**

By default, `sw_augment()` uses `broom::augment()` to convert its output.

**Usage**

```
## Default S3 method:
sw_augment(x, ...)
```
Arguments

- x: an object to be tidied
- ... extra arguments passed to `broom::augment()`

Value

A tibble generated by `broom::augment()`

---

### sw_augment_columns

**Augments data**

**Description**

Augments data

**Usage**

```r
sw_augment_columns(ret, data, rename_index, timetk_idx = FALSE)
```

**Arguments**

- `ret`: An object of class tibble
- `data`: Any time series data that is to be augmented
- `rename_index`: A variable indicating the index name to be used in the tibble returned
- `timetk_idx`: Uses the timetk index (irregular time index) if present.

---

### sw_glance

Construct a single row summary "glance" of a model, fit, or other object

**Description**

Construct a single row summary "glance" of a model, fit, or other object

**Usage**

```r
sw_glance(x, ...)
```

**Arguments**

- `x`: model or other R object to convert to single-row data frame
- `...`: other arguments passed to methods
Details

`sw_glance()` is a wrapper for `broom::glance()`. The benefit of `sw_glance` is that it has methods for various time-series model classes such as `HoltWinters`, `ets`, `Arima`, etc. `sw_glance` methods always return either a one-row tibble or `NULL`. The single row includes summary statistics relevant to the model accuracy, which can be used to assess model fit and quality.

For non-time series, `sw_glance()` defaults to `broom::glance()`. The only difference is that the return is a tibble.

Value

A single-row tibble with model summary information.

See Also

`broom::glance()`

Description

By default, `sw_glance()` uses `broom::glance()` to convert its output.

Usage

```r
## Default S3 method:
sw_glance(x, ...)
```

Arguments

- `x` an object to be tidied
- `...` extra arguments passed to `broom::glance()`

Value

A tibble generated by `broom::glance()`
**sw_sweep**

*Tidy forecast objects*

**Description**

Tidy forecast objects

**Usage**

```r
sw_sweep(x, fitted = FALSE, timetk_idx = FALSE, rename_index = "index", ...)
```

**Arguments**

- **x**: A time-series forecast of class `forecast`.
- **fitted**: Whether or not to return the fitted values (model values) in the results. FALSE by default.
- **timetk_idx**: If timetk index (non-regularized index) is present, uses it to develop forecast. Otherwise uses default index.
- **rename_index**: Enables the index column to be renamed.
- **...**: Additional arguments passed to `tk_make_future_timeseries()`

**Details**

`sw_sweep` is designed to coerce forecast objects from the `forecast` package into `tibble` objects in a "tidy" format (long). The returned object contains both the actual values and the forecasted values including the point forecast and upper and lower confidence intervals.

The `timetk_idx` argument is used to modify the return format of the index.

- If `timetk_idx = FALSE`, a regularized time index is always constructed. This may be in the format of numeric values (e.g. 2010.000) or the higher order `yearmon` and `yearqtr` classes from the `zoo` package. A higher order class is attempted to be returned.
- If `timetk_idx = TRUE` and a timetk index is present, an irregular time index will be returned that combines the original time series (i.e. date or datetime) along with a computed future time series created using `tk_make_future_timeseries()` from the `timetk` package. The `...` can be used to pass additional arguments to `tk_make_future_timeseries()` such as `inspect_weekdays`, `skip_values`, etc that can be useful in tuning the future time series sequence.

The index column name can be changed using the `rename_index` argument.

**Value**

Returns a `tibble` object.

**See Also**

`tk_make_future_timeseries()`
sw_tidy

Examples

library(forecast)
library(sweep)
library(dplyr)

# ETS forecasts
USAccDeaths %>%
  ets() %>%
  forecast(level = c(80, 95, 99)) %>%
  sw_sweep()

-------------------

sw_tidy

Tidy the result of a time-series model into a summary tibble

Description

Tidy the result of a time-series model into a summary tibble

Usage

sw_tidy(x, ...)

Arguments

x An object to be converted into a tibble ("tidy" data.frame)
...
extra arguments

Details

sw_tidy() is a wrapper for broom::tidy(). The main benefit of sw_tidy() is that it has methods for various time-series model classes such as HoltWinters, ets, Arima, etc. sw_tidy() methods always return a "tidy" tibble with model coefficient / parameters.

For non-time series, sw_tidy() defaults to broom::tidy(). The only difference is that the return is a tibble. The output of sw_tidy() is always a tibble with disposable row names. It is therefore suited for further manipulation by packages like dplyr and ggplot2.

Value

a tibble

See Also

broom::tidy()
Examples

```r
library(dplyr)
library(forecast)
library(sweep)

WWWusage %>%
  auto.arima() %>%
  sw_tidy(conf.int = TRUE)
```

---

**sw_tidy.default**  
*Default tidying method*

### Description

By default, `sw_tidy()` uses `broom::tidy()` to convert its output.

### Usage

```r
## Default S3 method:
sw_tidy(x, ...)
```

#### Arguments

- **x**  
an object to be tidied

- **...**  
extra arguments passed to `broom::tidy()`

#### Value

A tibble generated by `broom::tidy()`

---

**sw_tidy_decomp**  
*Coerces decomposed time-series objects to tibble format.*

### Description

Coerces decomposed time-series objects to tibble format.

### Usage

```r
sw_tidy_decomp(x, timetk_idx = FALSE, rename_index = "index", ...)
```
**Arguments**

- `x` A time-series object of class `stl`, `ets`, `decomposed.ts`, `HoltWinters`, `bats` or `tbats`.
- `timetk_idx` When `TRUE`, uses a timetk index (irregular, typically date or datetime) if present.
- `rename_index` Enables the index column to be renamed.
- `...` Not used.

**Details**

`sw_tidy_decomp` is designed to coerce time-series objects with decompositions to `tibble` objects. A regularized time index is always constructed. If no time index is detected, a sequential index is returned as a default. The index column name can be changed using the `rename_index` argument.

**Value**

Returns a `tibble` object.

**Examples**

```r
library(dplyr)
library(forecast)
library(sweep)

# Decompose ETS model
USAccDeaths %>%
ext() %>%
sw_tidy_decomp()

# Decompose STL object
USAccDeaths %>%
stl(s.window = 'periodic') %>%
sw_tidy_decomp()
```

---

**tbats_string**

*Print the TBATS model parameters*

**Description**

Refer to `forecast::makeTextTBATS`. `forecast bats.R`

**Usage**

`tbats_string(object)`

**Arguments**

- `object` An object of class `bats` or `tbats`
Tidiers_arima  Tidying methods for ARIMA modeling of time series

Description

These methods tidy the coefficients of ARIMA models of univariate time series.

Usage

```r
## S3 method for class 'Arima'
sw_tidy(x, ...)
```

```r
## S3 method for class 'Arima'
sw_glance(x, ...)
```

```r
## S3 method for class 'Arima'
sw_augment(x, data = NULL, rename_index = "index", timetk_idx = FALSE, 
           ...)
```

```r
## S3 method for class 'stlm'
sw_tidy(x, ...)
```

Arguments

- **x**: An object of class "Arima"
- **...**: Additional parameters (not used)
- **data**: Used with `sw_augment` only. NULL by default which simply returns augmented columns only. User can supply the original data, which returns the data + augmented columns.
- **rename_index**: Used with `sw_augment` only. A string representing the name of the index generated.
- **timetk_idx**: Used with `sw_augment` only. Uses a irregular timetk index if present.

Value

`sw_tidy()` returns one row for each coefficient in the model, with five columns:
- **term**: The term in the nonlinear model being estimated and tested
- **estimate**: The estimated coefficient

`sw_glance()` returns one row with the columns
- **model.desc**: A description of the model including the three integer components (p, d, q) are the AR order, the degree of differencing, and the MA order.
- **sigma**: The square root of the estimated residual variance
- **logLik**: The data’s log-likelihood under the model
- **AIC**: The Akaike Information Criterion
• BIC: The Bayesian Information Criterion
• ME: Mean error
• RMSE: Root mean squared error
• MAE: Mean absolute error
• MPE: Mean percentage error
• MAPE: Mean absolute percentage error
• MASE: Mean absolute scaled error
• ACF1: Autocorrelation of errors at lag 1

sw_augment() returns a tibble with the following time series attributes:

• index: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
• .actual: The original time series
• .fitted: The fitted values from the model
• .resid: The residual values from the model

sw_tidy() returns the underlying ETS or ARIMA model's sw_tidy() one row for each coefficient in the model, with five columns:

• term: The term in the nonlinear model being estimated and tested
• estimate: The estimated coefficient

See Also

arima(), Arima()

Examples

library(dplyr)
library(forecast)
library(sweep)

fit_arima <- WWWusage %>%
  auto.arima()

sw_tidy(fit_arima)
sw_glance(fit_arima)
sw_augment(fit_arima)
Description

Tidying methods for BATS and TBATS modeling of time series

Usage

```r
## S3 method for class 'bats'
sw_tidy(x, ...)

## S3 method for class 'bats'
sw_glance(x, ...)

## S3 method for class 'bats'
sw_augment(x, data = NULL, rename_index = "index", timetk_idx = FALSE, ...)

## S3 method for class 'bats'
sw_tidy_decomp(x, timetk_idx = FALSE, rename_index = "index", ...)
```

Arguments

- `x`: An object of class "bats" or "tbats"
- `...`: Additional parameters (not used)
- `data`: Used with `sw_augment` only. NULL by default which simply returns augmented columns only. User can supply the original data, which returns the data + augmented columns.
- `rename_index`: Used with `sw_augment` only. A string representing the name of the index generated.
- `timetk_idx`: Used with `sw_augment` and `sw_tidy_decomp`. When TRUE, uses a timetk index (irregular, typically date or datetime) if present.

Value

`sw_tidy()` returns one row for each model parameter, with two columns:
- `term`: The various parameters (lambda, alpha, gamma, etc)
- `estimate`: The estimated parameter value

`sw_glance()` returns one row with the columns
- `model.desc`: A description of the model including the three integer components (p, d, q) are the AR order, the degree of differencing, and the MA order.
- `sigma`: The square root of the estimated residual variance
- `logLik`: The data’s log-likelihood under the model
• AIC: The Akaike Information Criterion
• BIC: The Bayesian Information Criterion (NA for bats / tbats)
• ME: Mean error
• RMSE: Root mean squared error
• MAE: Mean absolute error
• MPE: Mean percentage error
• MAPE: Mean absolute percentage error
• MASE: Mean absolute scaled error
• ACF1: Autocorrelation of errors at lag 1

sw_augment() returns a tibble with the following time series attributes:
• index: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
• .actual: The original time series
• .fitted: The fitted values from the model
• .resid: The residual values from the model

sw_tidy_decomp() returns a tibble with the following time series attributes:
• index: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
• observed: The original time series
• level: The level component
• slope: The slope component (Not always present)
• season: The seasonal component (Not always present)

See Also
bats(), tbats()

Examples
library(dplyr)
library(forecast)
library(sweep)

fit_bats <- WWWusage %>%
  bats()
sw_tidy(fit_bats)
sw_glance(fit_bats)
sw_augment(fit_bats)
Description

Tidying methods for decomposed time series

Usage

## S3 method for class 'decomposed.ts'
sw_tidy_decomp(x, timetk_idx = FALSE, rename_index = "index", ...)

Arguments

x
An object of class "decomposed.ts"

timetk_idx
Used with sw_augment and sw_tidy_decomp. When TRUE, uses a timetk index
(irregular, typically date or datetime) if present.

rename_index
Used with sw_augment and sw_tidy_decomp. A string representing the name
of the index generated.

...
Not used.

Value

sw_tidy_decomp() returns a tibble with the following time series attributes:

• index: An index is either attempted to be extracted from the model or a sequential index
is created for plotting purposes

• season: The seasonal component

• trend: The trend component

• random: The error component

• seasadj: observed - season

See Also

decompose()

Examples

library(dplyr)
library(forecast)
library(sweep)

fit_decomposed <- USAccDeaths %>%
  decompose()

sw_tidy_decomp(fit_decomposed)
Tidy methods for ETS (Error, Trend, Seasonal) exponential smoothing modeling of time series

Description

Tidying methods for ETS (Error, Trend, Seasonal) exponential smoothing modeling of time series

Usage

## S3 method for class 'ets'
sw_tidy(x, ...)

## S3 method for class 'ets'
sw_glance(x, ...)

## S3 method for class 'ets'
sw_augment(x, data = NULL, timetk_idx = FALSE, rename_index = "index", ...)

## S3 method for class 'ets'
sw_tidy_decomp(x, timetk_idx = FALSE, rename_index = "index", ...)

Arguments

x An object of class "ets"
...
Not used.

data Used with sw_augment only. NULL by default which simply returns augmented columns only. User can supply the original data, which returns the data + augmented columns.

timetk_idx Used with sw_augment and sw_tidy_decomp. When TRUE, uses a timetk index (irregular, typically date or datetime) if present.

rename_index Used with sw_augment and sw_tidy_decomp. A string representing the name of the index generated.

Value

sw_tidy() returns one row for each model parameter, with two columns:

- term: The smoothing parameters (alpha, gamma) and the initial states (l, s0 through s10)
- estimate: The estimated parameter value

sw_glance() returns one row with the columns

- model.desc: A description of the model including the three integer components (p, d, q) are the AR order, the degree of differencing, and the MA order.
- sigma: The square root of the estimated residual variance
• logLik: The data’s log-likelihood under the model
• AIC: The Akaike Information Criterion
• BIC: The Bayesian Information Criterion
• ME: Mean error
• RMSE: Root mean squared error
• MAE: Mean absolute error
• MPE: Mean percentage error
• MAPE: Mean absolute percentage error
• MASE: Mean absolute scaled error
• ACF1: Autocorrelation of errors at lag 1

sw_augment() returns a tibble with the following time series attributes:
• index: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
• .actual: The original time series
• .fitted: The fitted values from the model
• .resid: The residual values from the model

sw_tidy_decomp() returns a tibble with the following time series attributes:
• index: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
• observed: The original time series
• level: The level component
• slope: The slope component (Not always present)
• season: The seasonal component (Not always present)

See Also
ets()

Examples
library(dplyr)
library(forecast)
library(sweep)

fit_ets <- WWWusage %>%
  ets()

sw_tidy(fit_ets)
sw_glance(fit_ets)
sw_augment(fit_ets)
sw_tidy_decomp(fit_ets)
Description

These methods tidy HoltWinters models of univariate time series.

Usage

## S3 method for class 'HoltWinters'
sw_tidy(x, ...)

## S3 method for class 'HoltWinters'
sw_glance(x, ...)

## S3 method for class 'HoltWinters'
sw_augment(x, data = NULL, rename_index = "index", timetk_idx = FALSE, ...)

## S3 method for class 'HoltWinters'
sw_tidy_decomp(x, timetk_idx = FALSE, rename_index = "index", ...)

Arguments

x         An object of class "HoltWinters"
...
  Additional parameters (not used)
data     Used with sw_augment only. NULL by default which simply returns augmented columns only. User can supply the original data, which returns the data + augmented columns.
rename_index     Used with sw_augment only. A string representing the name of the index generated.
timetk_idx     Used with sw_augment and sw_tidy_decomp. When TRUE, uses a timetk index (irregular, typically date or datetime) if present.

Value

sw_tidy() returns one row for each model parameter, with two columns:
  • term: The various parameters (alpha, beta, gamma, and coefficients)
  • estimate: The estimated parameter value

sw_glance() returns one row with the following columns:
  • model.desc: A description of the model
  • sigma: The square root of the estimated residual variance
  • logLik: The data’s log-likelihood under the model
  • AIC: The Akaike Information Criterion
• BIC: The Bayesian Information Criterion (NA for bats / tbats)
• ME: Mean error
• RMSE: Root mean squared error
• MAE: Mean absolute error
• MPE: Mean percentage error
• MAPE: Mean absolute percentage error
• MASE: Mean absolute scaled error
• ACF1: Autocorrelation of errors at lag 1

sw_augment() returns a tibble with the following time series attributes:
• index: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
• .actual: The original time series
• .fitted: The fitted values from the model
• .resid: The residual values from the model

sw_tidy_decomp() returns a tibble with the following time series attributes:
• index: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
• observed: The original time series
• season: The seasonal component
• trend: The trend component
• remainder: observed - (season + trend)
• seasadj: observed - season (or trend + remainder)

See Also
HoltWinters()

Examples

library(dplyr)
library(forecast)
library(sweep)

fit_hw <- USAccDeaths %>%
  stats::HoltWinters()

sw_tidy(fit_hw)
sw_glance(fit_hw)
sw_augment(fit_hw)
sw_tidy_decomp(fit_hw)
Description

These methods tidy the coefficients of NNETAR models of univariate time series.

Usage

```r
## S3 method for class 'nnetar'
sw_tidy(x, ...)
## S3 method for class 'nnetar'
sw_glance(x, ...)
## S3 method for class 'nnetar'
sw_augment(x, data = NULL, timetk_idx = FALSE, rename_index = "index", ...)
```

Arguments

- `x` An object of class "nnetar"
- `...` Additional parameters (not used)
- `data` Used with `sw_augment` only. NULL by default which simply returns augmented columns only. User can supply the original data, which returns the data + augmented columns.
- `timetk_idx` Used with `sw_augment` only. Uses a irregular timetk index if present.
- `rename_index` Used with `sw_augment` only. A string representing the name of the index generated.

Value

- `sw_tidy()` returns one row for each model parameter, with two columns:
  - `term`: The smoothing parameters (alpha, gamma) and the initial states (l, s0 through s10)
  - `estimate`: The estimated parameter value

- `sw_glance()` returns one row with the columns:
  - `model.desc`: A description of the model including the three integer components (p, d, q) are the AR order, the degree of differencing, and the MA order.
  - `sigma`: The square root of the estimated residual variance
  - `logLik`: The data’s log-likelihood under the model (NA)
  - `AIC`: The Akaike Information Criterion (NA)
  - `BIC`: The Bayesian Information Criterion (NA)
  - `ME`: Mean error
- RMSE: Root mean squared error
- MAE: Mean absolute error
- MPE: Mean percentage error
- MAPE: Mean absolute percentage error
- MASE: Mean absolute scaled error
- ACF1: Autocorrelation of errors at lag 1

`sw_augment()` returns a tibble with the following time series attributes:

- `index`: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
- `actual`: The original time series
- `fitted`: The fitted values from the model
- `resid`: The residual values from the model

See Also

`nnetar()`

Examples

```r
library(dplyr)
library(forecast)
library(sweep)

fit_nnetar <- lynx %>%
  nnetar()

sw_tidy(fit_nnetar)
sw_glance(fit_nnetar)
sw_augment(fit_nnetar)
```

---

| tidiers_stl | Tidying methods for STL (Seasonal, Trend, Level) decomposition of time series |

**Description**

Tidying methods for STL (Seasonal, Trend, Level) decomposition of time series
### Usage

```r
## S3 method for class 'stl'
sw_tidy(x, ...)

## S3 method for class 'stl'
sw_tidy_decomp(x, timetk_idx = FALSE, rename_index = "index", ...)

## S3 method for class 'stlm'
sw_tidy_decomp(x, timetk_idx = FALSE, rename_index = "index", ...)

## S3 method for class 'stlm'
sw_glance(x, ...)

## S3 method for class 'stlm'
sw_augment(x, data = NULL, rename_index = "index", timetk_idx = FALSE, ...)
```

### Arguments

- **x**: An object of class "stl"
- **...**: Not used.
- **timetk_idx**: Used with `sw_tidy_decomp`. When TRUE, uses a timetk index (irregular, typically date or datetime) if present.
- **rename_index**: Used with `sw_tidy_decomp`. A string representing the name of the index generated.
- **data**: Used with `sw_augment` only.

### Value

- **sw_tidy()** wraps **sw_tidy_decomp()**
  - sw_tidy_decomp() returns a tibble with the following time series attributes:
    - **index**: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
    - **season**: The seasonal component
    - **trend**: The trend component
    - **remainder**: observed - (season + trend)
    - **seasadj**: observed - season (or trend + remainder)

- **sw_glance()** returns the underlying ETS or ARIMA model’s sw_glance() results one row with the columns
  - **model.desc**: A description of the model including the three integer components (p, d, q) are the AR order, the degree of differencing, and the MA order.
  - **sigma**: The square root of the estimated residual variance
  - **logLik**: The data’s log-likelihood under the model
  - **AIC**: The Akaike Information Criterion
• BIC: The Bayesian Information Criterion
• ME: Mean error
• RMSE: Root mean squared error
• MAE: Mean absolute error
• MPE: Mean percentage error
• MAPE: Mean absolute percentage error
• MASE: Mean absolute scaled error
• ACF1: Autocorrelation of errors at lag 1

sw_augment() returns a tibble with the following time series attributes:

• index: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
• .actual: The original time series
• .fitted: The fitted values from the model
• .resid: The residual values from the model

See Also
stl()

Examples

library(dplyr)
library(forecast)
library(sweep)

fit_stl <- USAccDeaths %>%
   stl(s.window = "periodic")

sw_tidy_decomp(fit_stl)
Usage

```r
## S3 method for class 'StructTS'
sw_tidy(x, ...)

## S3 method for class 'StructTS'
sw_glance(x, ...)

## S3 method for class 'StructTS'
sw_augment(x, data = NULL, timetk_idx = FALSE, rename_index = "index", ...)
```

Arguments

- `x` An object of class "StructTS"
- `...` Additional parameters (not used)
- `data` Used with `sw_augment` only. NULL by default which simply returns augmented columns only. User can supply the original data, which returns the data + augmented columns.
- `timetk_idx` Used with `sw_augment` only. Uses a irregular timetk index if present.
- `rename_index` Used with `sw_augment` only. A string representing the name of the index generated.

Value

- `sw_tidy()` returns one row for each model parameter, with two columns:
  - `term`: The model parameters
  - `estimate`: The estimated parameter value

- `sw_glance()` returns one row with the columns
  - `model.desc`: A description of the model including the three integer components (p, d, q) are the AR order, the degree of differencing, and the MA order.
  - `sigma`: The square root of the estimated residual variance
  - `logLik`: The data’s log-likelihood under the model
  - `AIC`: The Akaike Information Criterion
  - `BIC`: The Bayesian Information Criterion
  - `ME`: Mean error
  - `RMSE`: Root mean squared error
  - `MAE`: Mean absolute error
  - `MPE`: Mean percentage error
  - `MAPE`: Mean absolute percentage error
  - `MASE`: Mean absolute scaled error
  - `ACF1`: Autocorrelation of errors at lag 1

- `sw_augment()` returns a tibble with the following time series attributes:
• index: An index is either attempted to be extracted from the model or a sequential index is created for plotting purposes
• .actual: The original time series
• .fitted: The fitted values from the model
• .resid: The residual values from the model

See Also

StructTS()

Examples

library(dplyr)
library(forecast)
library(sweep)

fit_StructTS <- WWWusage %>%
  StructTS()

sw_tidy(fit_StructTS)
sw_glance(fit_StructTS)
sw_augment(fit_StructTS)

---

validate_index Validate data frame has column named the same name as variable rename_index

Description

Validates data frame has column named the same name as variable rename_index

Usage

validate_index(ret, rename_index)

Arguments

ret An object of class tibble
rename_index A variable indicating the index name to be used in the tibble returned
Index

* datasets
  - bike_sales, 3

add_index, 2
Arima(), 13
arima(), 13
arima_string, 3
bats(), 15
bats_string, 3
bike_sales, 3
broom::augment(), 5, 6
broom::glance(), 7
broom::tidy(), 9, 10
decompose(), 16
ets(), 18
HoltWinters(), 20
nnetar(), 22
stl(), 24
StructTS(), 26
sw_augment, 5
sw_augment(), 4
sw_augment.Arima(tidders_arima), 12
sw_augment.bats(tidders_bats), 14
sw_augment.default, 5
sw_augment.ets(tidders_ets), 17
sw_augment.HoltWinters (tidders_HoltWinters), 19
sw_augment.nnetar(tidders_nnetar), 21
sw_augment.stlm(tidders_stl), 22
sw_augment.StructTS(tidders_StructTS), 24
sw_augment_columns, 6
sw_glance, 6
sw_glance(), 4
sw_glance.Arima(tidders_arima), 12
sw_glance.bats (tidders_bats), 14
sw_glance.default, 7
sw_glance.ets (tidders_ets), 17
sw_glance.HoltWinters (tidders_HoltWinters), 19
sw_glance.nnetar (tidders_nnetar), 21
sw_glance.stlm (tidders_stl), 22
sw_glance.StructTS (tidders_StructTS), 24
sw_sweep, 8
sw_sweep(), 4
sw_tidy, 9
sw_tidy(), 4
sw_tidy.Arima (tidders_arima), 12
sw_tidy.bats (tidders_bats), 14
sw_tidy.default, 10
sw_tidy.ets (tidders_ets), 17
sw_tidy.HoltWinters (tidders_HoltWinters), 19
sw_tidy.nnetar (tidders_nnetar), 21
sw_tidy.stl (tidders_stl), 22
sw_tidy.stlm (tidders_arima), 12
sw_tidy.StructTS (tidders_StructTS), 24
sw_tidy.decomp, 10
sw_tidy.decomp(), 4
sw_tidy.decomp.bats (tidders_bats), 14
sw_tidy.decomp.decomposed.ts (tidders_decomposed_ts), 16
sw_tidy.decomp.ets (tidders_ets), 17
sw_tidy.decomp.HoltWinters (tidders_HoltWinters), 19
sw_tidy.decomp.stl (tidders_stl), 22
sw_tidy.decomp.stlm (tidders_stl), 22
sweep_package, 4
tbats(), 15
tbats_string, 11
tidders_arima, 12
tidders_bats, 14
tidders_decomposed_ts, 16
tidiers_ets, 17
tidiers_HoltWinters, 19
tidiers_nnetar, 21
tidiers_stl, 22
tidiers_StructTS, 24
tk_make_future_timeseries(), 8
validate_index, 26