Package 'TimeSeries.OBeu'

October 5, 2016

Type Package

Title Time Series Analysis OpenBudgets.eu

Version 0.1

Date 2016-09-18

Author Kleanthis Koupidis

Maintainer Kleanthis Koupidis <koupidis.okfgr@gmail.com>

Description Time Series Analysis for OBeu datasets

URL https://github.com/okgreece/OBeU

BugReports https://github.com/okgreece/OBeU/issues

License GPL-3

LazyData TRUE

Suggests testthat

Imports forecast, jsonlite, trend, tseries

RoxygenNote 5.0.1

R topics documented:

Athens_approved_ts — *Time series of Approved Expenditure Budget Phase of Municipality of Athens*

Description

Time series data with the Approved Budget phase expenditure amounts of Municipality of Athens from 2004-2015

- The years of the recorded approved budget phase amounts.
- The approved budget phase amounts of this time range.

Usage

```
Athens_approved_ts
```

Format

A ts object with 12 approved amounts from 2004-2015

Source

add #url#

Athens_draft_ts — *Time series of Draft Expenditure Budget Phase of Municipality of Athens*

Description

Time series data with the Draft Budget phase expenditure amounts of Municipality of Athens from 2004-2015

- The years of the recorded draft budget phase amounts.
- The draft budget phase amounts of this time range.

Usage

```
Athens_draft_ts
```

Format

A ts object with 12 draft amounts from 2004-2015

Source

add #url#

Athens_executed_ts — *Time series of Executed Expenditure Budget Phase of Municipality of Athens*

Description

Time series data with the Executed Budget phase expenditure amounts of Municipality of Athens from 2004-2015

- The years of the recorded executed budget phase amounts.
- The executed budget phase amounts of this time range.

Usage

```
Athens_executed_ts
```

Format

A ts object with 12 draft amounts from 2004-2015

Source

add #url#

Athens_reserved_ts — *Time series of Reserved Expenditure Budget Phase of Municipality of Athens*

Description

Time series data with the Reserved Budget phase expenditure amounts of Municipality of Athens from 2004-2015

- The years of the recorded reserved budget phase amounts.
- The reserved budget phase amounts of this time range.

Usage

```
Athens_reserved_ts
```

Format

A ts object with 12 reserved amounts from 2004-2015

Source

add #url#

Athens_revised_ts — *Time series of Revised Expenditure Budget Phase of Municipality of Athens*

Description

Time series data with the Revised Budget phase expenditure amounts of Municipality of Athens from 2004-2015

- The years of the recorded revised budget phase amounts.
- The revised budget phase amounts of this time range.

Usage

```
Athens_revised_ts
```

Format

A ts object with 12 revised amounts from 2004-2015

Source

add #url#

babbage.tsa.obeu — *Read and analyze time series data from Babbage API*

Description

Extract and analyze time series data from babbage api, using the tsa.obeu function.

Usage

```
babbage.tsa.obeu(json_data,time,amount,prediction_steps)
```

Arguments

json_data	The json string, URL or file from babbage api.
time	Specify the time label of the json time series data.
amount	Specify the amount label of the json time series data.
prediction_steps	The number of prediction steps.

Details

This function extracts the time series data provided by the Babbage API. A json file analyze it using the tsa.obeu function.

Value

A json string with the resulted parameters of the tsa.obeu function.

Author(s)

Kleanthis Koupidis

References

add

See Also

tsa.obeu

`forecast.tsa.obeu` *Time series forecast results of OBEU Time Series*

Description

Univariate time series forecasts for short and long time series data using the appropriate model.

Usage

```
forecast.tsa.obeu(ts_model, h=1)
```

Arguments

`ts_model`	The input univariate time series data
`h`	The number of prediction steps

Details

This function is used internally in tsa.obeu and forecasts the model that fits the input data using the auto.arima function(see forecast package). The model selection depends on the results of some diagnostic tests (acf,pacf,pp adf and kpss). For short time series the selected arima model is among various orders of the AR part using 1st differences and MA(1), with the lower AIC value.

Value

A list with the parameters:

- ts.name: a string indicating the name of the time series data
- ts.model: a string indicating the arima orders
- data_year: The time that time series data were sampled
- data: The time series values
- predict_time: The time that defined by the prediction_steps parameter
- predict_values: The predicted values that defined by the prediction_steps parameter
- up80: The upper limit of the 80% predicted confidence interval
- low80: The lower limit of the 80% predicted confidence interval
- up95: The upper limit of the 95% predicted confidence interval
- low95: The lower limit of the 95% predicted confidence interval

Author(s)

Kleanthis Koupidis

See Also

tsa.obeu, forecast(forecast package)

`stationary.test` *Stationarity testing*

Description

This functions tests the stationarity of the input time series data.

Usage

```
stationary.test(tsdata)
```

Arguments

`tsdata`	The input univariate time series data

Details

This function tests the deterministic and stohastic trend of the input time series data. This function uses ACF and PACF functions from forecast package, Phillips-Perron test, Augmented Dickeyß€“Fuller (ADF) test, Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test, from tseries package and Mann-Kendall test for Monotonic Trend Cox and Stuart trend test from trend package.

Phillips-Perron test tests the null hypothesis of whether a unit root is present in a time series sample, against a stationary alternative. The truncation lag parameter is set to trunc(4*(n/100)^0.25), where n the length of the in input time series data

Augmented Dickeyß€“Fuller (ADF) test, tests the null hypothesis of whether a unit root is present in a time series sample. The truncation lag parameter is set to trunc((n-1)^(1/3))), where n the lengthof the input time series data

Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test, tests a null hypothesis that an observable time series is stationary around a deterministic trend (i.e. trend-stationary) against the alternative of a unit root. The truncation lag parameter is set to trunc(3*sqrt(n)/13), where n the length of the input time series data

The non-parametric Mann-Kendall test is used to detect monotonic trends. The null hypothesis, H0, is that the data come from a population with independent realizations and are identically distributed. The alternative hypothesis, HA, is that the data follow a monotonic trend.

The Cox and Stuart test is a modified sign test. The null hypothesis, H0, is that the input time series assumed to be independent against the fact that there is a time dependent trend (monotonic trend).

Value

A string indicating if the time series is stationary or non stationary for internal use in tsa.obeu.

Author(s)

Kleanthis Koupidis

References

tseries, trend

See Also

tsa.obeu, Acf and Pacf(forecast package),pp.test, adf.test and kpss.test (tseries) mk.test and cs.test (trend package)

`ts.acf.obeu` *Extract the ACF and PACF parameters of time series and their model residuals*

Description

This function is included in tsa.obeu function and aims to extract the ACF and PACF details of the input time series data and the ACF, PACF of the residuals after fitting an Arima model.

Usage

```
ts.acf.obeu(tsdata,model_residuals,a=0.95)
```

Arguments

`tsdata`	The input univariate time series data
`model_residuals`	The model's residuals after fitting a model to the time series
`a`	The significant level (default a=0.95)

Details

This function is used internally in tsa.obeu function and the output is a list with grouped ACF and PACF parameters of the input time series data, as well as the ACF and PACF parameters of the residuals needed for the graphical purposes in OBEU.

Value

A list with the parameters:

- acf.parameters:
  - acf: The estimated acf values of the input time series
  - acf.lag: The lags at which the acf is estimated
  - confidence.interval.up: The upper limit of the confidence interval
  - confidence.interval.low: The lower limit of the confidence interval
- pacf.parameters:
  - pacf: The estimated pacf values of the input time series
  - pacf.lag: The lags at which the pacf is estimated

  - confidence.interval.up: The upper limit of the confidence interval
  - confidence.interval.low: The lower limit of the confidence interval
- acf.residuals.parameters:
  - acf.res: The estimated acf values of the model residuals
  - acf.res.lag: The lags at which the acf is estimated of the model residuals
  - confidence.interval.up: The upper limit of the confidence interval
  - confidence.interval.low: The lower limit of the confidence interval
- pacf.residuals.parameters:
  - pacf.res: The estimated pacf values of the model residuals
  - pacf.res.lag: The lags at which the pacf is estimated of the model residuals
  - confidence.interval.up: The upper limit of the confidence interval
  - confidence.interval.low: The lower limit of the confidence interval

Author(s)

Kleanthis Koupidis

See Also

tsa.obeu

`ts.non.seas.decomp` *Non seasonal decomposition*

Description

Decomposition of time series with no seasonal component

Usage

```
ts.non.seas.decomp(tsdata)
```

Arguments

`tsdata`	The input univariate non seasonal time series data

Details

For non-seasonal time series there is no seasonal component. We use Local Polynomial Regression Fitting (LOESS) in order to extract the trend component and then we subtract the trend from the initial values to extract the irregular terms.

Value

A list with the following components:

- timeseries: The time series data
- season: There is no seasonality, this parameter is set NULL
- loess.trend:
  - trend: The trend values
  - conf.interval.up: The upper limit of the trend confidence interval
  - conf.interval.low: The lower limit of the trend confidence interval
- loess.comparison:
  - number.observation: The time series length
  - loess.residuals: The loess residuals
  - loess.enp:
  - loess.s:
  - loess.onedelta:
  - loess.twodelta:
  - loess.tracehat:
  - loess.divisor:
  - loess.robust:
  - loess.weights:

Author(s)

Kleanthis Koupidis

References

add

See Also

tsa.obeu, loess, predict.loess (stats package)

---

ts.non.seas.model *Decomposition of non seasonal time series*

Description

Decomposition of time series with no seasonal component

Usage

```
ts.non.seas.decomp(tsdata)
```

Arguments

tsdata	The input univariate non seasonal time series data

Details

For non-seasonal time series there is no seasonal component. We use Local Polynomial Regression Fitting (LOESS) in order to extract the trend component and then we subtract the trend from the initial values to extract the irregular terms.

Author(s)

Kleanthis Koupidis

References

add

See Also

tsa.obeu, loess, predict.loess (stats package)

---

ts.seasonal.obeu *Decomposition of seasonal time series*

Description

Decomposition of seasonal time series data using stlm from forecast package. This function is used internally in tsa.obeu.

Usage

```
ts.seasonal.obeu(tsdata)
```

Arguments

tsdata	The input univariate seasonal time series data

Details

Decomposition of seasonal time series data is based on stlm from forecast package and returns a list with useful parameters for OBEU.

Value

- ts_model: Summary of the arima model
- stl.general:
  - trend: The estimated trend component
  - seasonal: The estimated seasonal component
  - remainder: The estimated remainder component
  - weights: The final robust weights (if robust=F all weights are one)
  - window: A vector with the spans used for the "s", "t", and "l" smoothers
  - stl.degree: A vector with the polynomial degrees for these smoothers
  - lambda: Box-Cox transformation parameter
  - tsdata.stl$x:

  - tsdata.stl$m:
  - fitted: The model's fitted values
- ts_model:
  - arima.order: The Arima order
  - arima.coef: A vector of AR, MA and regression coefficients
  - arima.coef.se: The standard error of the coefficients
- residuals: The residuals of the model (fitted innovations)
- residuals.other:
  - resid.variance: The MLE of the innovations variance
  - covariance.coef: The matrix of the estimated variance of the coefficients
- used.notused.observations:
  - not.used.obs: The number of not used observations for the fitting
  - used.obs: the number of used observations for the fitting
- comparison:
  - loglik: The maximized log-likelihood (of the differenced data), or the approximation to it used
  - aic: The AIC value corresponding to the log-likelihood
  - bic: The BIC value corresponding to the log-likelihood
  - aicc: The second-order Akaike Information Criterion corresponding to the log-likelihood
- data: The time series data

Author(s)

Kleanthis Koupidis

References

add

See Also

tsa.obeu, stlm (forecast package)

`tsa.obeu` *Time series analysis results for OBEU Time series*

Description

Univariate time series analysis for short and long time series data using the appropriate model.

Usage

```
tsa.obeu(tsdata, h)
```

Arguments

`tsdata`	The input univariate time series data
`h`	The number of prediction steps

Details

This function automatically tests for stationarity of the input time series data in order to select the appropriate arima model that fits the input data using the auto.arima function(see forecast package). For short time series the selected arima model is among various orders of the AR part using 1st differences and MA(1), with the lower AIC. This function also decomposes both seasonal and non seasonal time series and forecasts h steps ahead the user selected(default h=1).

Value

A json string with the parameters (Missing some): ts_name param forecasts

Author(s)

Kleanthis Koupidis

References

add

See Also

babbage.tsa.obeu

Index