Package 'TimeSeries.OBeu'

November 6, 2016

Type Package

Title Time Series Analysis OpenBudgets.eu

Version 1.1.4

Date 2016-09-18

Description Time Series Analysis for OBeu datasets.

Author Kleanthis Koupidis

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

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

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

License GPL-3

LazyData TRUE

Suggests testthat

Imports forecast, locfit, 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

Format

A ts object with 12 approved amounts from 2004-2015

Source

The ttl and rdf expenditure data are stored in: https://github.com/openbudgets/datasets/tree/master/greek-municipalities/municipality-of-athens/dataset

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.

Format

A ts object with 12 draft amounts from 2004-2015

Source

The ttl and rdf expenditure data are stored in: https://github.com/openbudgets/datasets/tree/master/greek-municipalities/municipality-of-athens/dataset

`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.

Format

A ts object with 12 draft amounts from 2004-2015

Source

The ttl and rdf expenditure data are stored in: https://github.com/openbudgets/datasets/tree/master/greek-municipalities/municipality-of-athens/dataset

`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.

Format

A ts object with 12 reserved amounts from 2004-2015

Source

The ttl and rdf expenditure data are stored in: https://github.com/openbudgets/datasets/tree/master/greek-municipalities/municipality-of-athens/dataset

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.

Format

A ts object with 12 revised amounts from 2004-2015

Source

The ttl and rdf expenditure data are stored in: https://github.com/openbudgets/datasets/tree/master/greek-municipalities/municipality-of-athens/dataset

open_spending.ts — *Read and analyze univariate time series data from Open Spending API*

Description

Extract and analyze univariate time series data from Open Spending API, using the ts.analysis function.

Usage

```
open_spending.ts(json_data,time,amount,order=NULL,prediction_steps=1)
```

Arguments

`json_data`	The json string, URL or file from Open Spending API
`time`	Specify the time label of the json time series data
`amount`	Specify the amount label of the json time series data
`order`	An integer vector of length 3 specifying the order of the Arima model
`prediction_steps`	The number of prediction steps

Details

This function extracts the time series data provided by the Open Spending API, in order to return the results from the `ts.analysis` function.

Value

A json string with the resulted parameters of the ts.analysis function.

Author(s)

Kleanthis Koupidis

See Also

ts.analysis

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

Description

This function is included in ts.analysis 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(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 ts.analysis 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

ts.analysis

Examples

```
ts.acf(Athens_draft_ts)
```

`ts.analysis` *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

```
ts.analysis(tsdata,x.order=NULL,h=1)
```

Arguments

`tsdata`	The input univariate time series data
`x.order`	An integer vector of length 3 specifying the order of the Arima model
`h`	The number of prediction steps

Details

This function automatically tests for stationarity of the input time series data using `ts.stationary.test` function. Depending the nature of the time series data and the stationary tests there are four branches: a.)short and non seasonal, b.)short and seasonal, c.)long and non seasonal and d.)long and seasonal. For branches a and c `ts.non.seas.model` is used and for b and d `ts.seasonal.model` is used.

This function also decomposes both seasonal and non seasonal time series through `ts.non.seas.decomp` and `ts.seasonal.decomp` and forecasts h steps ahead the user selected(default h=1) using `ts.forecast`.

Value

A json string with the parameters:

- acf.param
  - 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
- param
  - stl.plot:
    - trend: The estimated trend component
    - trend.ci.up: The estimated up limit for trend component (for non seasonal time series)
    - trend.ci.low: The estimated low limit for trend component (for non seasonal time series)
    - seasonal: The estimated seasonal component
    - remainder: The estimated remainder component
    - time: The time of the series was sampled
  - stl.general:
    - stl.degree: The degree of fit
    - degfr: The effective degrees of freedom for non seasonal time series
    - degfr.fitted: The fitted degrees of freedom for non seasonal time series
    - fitted: The model's fitted values
  - residuals: The residuals of the model (fitted innovations)
  - compare:
    - arima.order: The Arima order for seasonal time series
    - arima.coef: A vector of AR, MA and regression coefficients for seasonal time series
    - arima.coef.se: The standard error of the coefficients for seasonal time series
    - covariance.coef: The matrix of the estimated variance of the coefficients for seasonal time series

    - resid.variance: The residuals variance
    - not.used.obs: The number of not used observations for the fitting for seasonal time series
    - used.obs: The used observations for the fitting
    - 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
    - gcv: The generalized cross-validation statistic for non seasonal time series or
    - aicc: The second-order Akaike Information Criterion corresponding to the log-likelihood for seasonal time series
- forecasts
  - 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

References

add

See Also

ts.analysis

Examples

```
ts.analysis(Athens_draft_ts,h=3)
```

`ts.forecast` *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

```
ts.forecast(ts_modelx,h=1)
```

Arguments

ts_modelx	The input univariate time series data
h	The number of prediction steps

Details

This function is used internally in ts.analysis 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 the first differences and the first order moving average component, with the lower AIC value.

Value

A list with the parameters:

- 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

ts.analysis, forecast(forecast package)

Examples

```
Athens_draft <- ts.non.seas.model(Athens_draft_ts)
#Hold the model object of non seasonal modeling
draft<-Athens_draft$model.summary
ts.forecast(draft)
```

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

Description

Decomposition of time series with no seasonal component using local regression models

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. Local regression and likelihood models (locfit package) are used in order to extract the trend and remaider components.

Value

A list with the following components:

- stl.plot:
  - trend: The estimated trend component
  - trend.ci.up: The estimated up limit for trend component
  - trend.ci.low: The estimated low limit for trend component
  - seasonal: The estimated seasonal component
  - remainder: The estimated remainder component
  - time: The time of the series was sampled
- stl.general:
  - stl.degree: The degree of fit
  - degfr: The effective degrees of freedom
  - degfr.fitted: The fitted degrees of freedom
- residuals_fitted:
  - residuals: The residuals of the model (fitted innovations)
  - fitted: The model's fitted values
- compare:
  - resid.variance: The residuals variance
  - used.obs: The used observations for the fitting
  - 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
  - gcv: The generalized cross-validation statistic

Author(s)

Kleanthis Koupidis

References

add

See Also

ts.analysis, locfit, predict.locfit

Examples

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

---

ts.non.seas.model *Model fit of non seasonal time series*

Description

Model fit of non seasonal time series

Usage

```
ts.non.seas.model(tsdata,x.ord=NULL)
```

Arguments

tsdata	The input univariate non seasonal time series data
x.ord	An integer vector of length 3 specifying the order of the Arima model

Details

Model fit of non seasonal time series using arima models of non seasonal time series data. The model with the lowest AIC value is selected for forecasts.

Value

A list with the following components:

- model.summary:
  - ts_model: The summary model details returned as Arima object for internal use in ts.analysis function
- model:
  - 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)

- compare:
  - variance.coef: The matrix of the estimated variance of the coefficients
  - resid.variance: The MLE of the innovations variance
  - not.used.obs: The number of not used observations for the fitting
  - used.obs: the number of used observations for the fitting
  - 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

Author(s)

Kleanthis Koupidis

References

add

See Also

`ts.analysis`, Arima

Examples

```
ts.non.seas.model(Athens_draft_ts)
```

`ts.seasonal.decomp` *Decomposition of seasonal time series*

Description

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

Usage

```
ts.seasonal.decomp(tsdata)
```

Arguments

`tsdata`	The input univariate seasonal time series data

Details

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

Value

A list with the following components:

- stl.plot:
  - trend: The estimated trend component
  - seasonal: The estimated seasonal component
  - remainder: The estimated remainder component
  - time: The time of the series was sampled
- stl.general:
  - model.summary The summary object of the arima model to use in forecast if needed
  - stl.win: An integer vector of length 3 indicating the spans used for the "s", "t", and "l" smoothers
  - stl.degree: An integer vector of length 3 indicating the polynomial degrees for these smoothers
- residuals_fitted:
  - residuals: The residuals of the model (fitted innovations)
  - fitted: The model's fitted values
- compare:
  - arima.order: The Arima order
  - arima.coef: A vector of AR, MA and regression coefficients
  - arima.coef.se: The standard error of the coefficients
  - covariance.coef: The matrix of the estimated variance of the coefficients
  - resid.variance: The MLE of the innovations variance
  - not.used.obs: The number of not used observations for the fitting
  - used.obs: the number of used observations for the fitting
  - 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

Author(s)

Kleanthis Koupidis

References

add

See Also

ts.analysis, stlm (forecast package)

`ts.seasonal.model` *Model fit of seasonal time series*

Description

Model fit of seasonal time series

Usage

```
ts.seasonal.model(tsdata,x.ord=NULL)
```

Arguments

`tsdata`	The input univariate seasonal time series data
`x.ord`	An integer vector of length 3 specifying the order of the Arima model

Details

Model fit of seasonal time series using arima models of seasonal time series data. The model with the lowest AIC value is selected for forecasts.

Value

A list with the following components:

- model.summary:
  - ts_model: The summary model details returned as Arima object for internal use in ts.analysis function
- model:
  - 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)
- compare:
  - variance.coef: The matrix of the estimated variance of the coefficients
  - resid.variance: The MLE of the innovations variance
  - not.used.obs: The number of not used observations for the fitting
  - used.obs: the number of used observations for the fitting
  - 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

Author(s)

Kleanthis Koupidis

See Also

ts.analysis, Arima

`ts.stationary.test` *Stationarity testing*

Description

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

Usage

```
ts.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 Dickey 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 Dickey 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 length of 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 ts.analysis.

Author(s)

Kleanthis Koupidis

References

tseries, trend

See Also

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

Examples

```
ts.stationary.test(Athens_approved_ts)
```

Index