Package 'tsforecast'

May 8, 2026

Type Package

Title Time Series Forecasting Functions

Version 1.3.0

Date 2026-01-15

Maintainer Ka Yui Karl Wu <karlwuky@suss.edu.sg>

Language en-GB

Description Fundamental time series forecasting models such as autoregressive integrated moving average (ARIMA), exponential smoothing, and simple moving average are included. For ARIMA models, the output follows the traditional parameterisation by Box and Jenkins (1970, ISBN: 0816210942, 9780816210947). Furthermore, there are functions for detailed time series exploration and decomposition, respectively. All data and result visualisations are generated by 'ggplot2' instead of conventional R graphical output. For more details regarding the theoretical background of the models see Hyndman, R.J. and Athanasopoulos, G. (2021) <https://otexts.com/fpp3/>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Depends R (>= 3.5.0)

Imports ggplot2, lubridate, forecast, tseries, scales

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Ka Yui Karl Wu [aut, cre]

Repository CRAN

Date/Publication 2026-01-21 12:40:02 UTC

Contents

`airport` *Airport Travellers Time Series.*

Description

A dataset containing the total number of monthly travellers recorded in an unnamed airport between January 2013 and December 2019.

Usage

```
airport
```

Format

A data frame with 84 rows and 2 variables:

Date Months in which the traveller numbers are recorded.

Travellers Total travellers in the airport of that month.

AvgTemp Monthly average temperature (in °C) of the city where the airport is located.

AvgRain Monthly average rain fall (in mm) in the city where the airport is located.

Examples

```
data(airport)
head(airport)
```

`is.outlier` *Outlier Identification*

Description

The function '`is.outlier`' checks whether any time series observations are outliers based on the interquartile range (IQR) rule.

Usage

```
is.outlier(x, method = c("iqr", "sigma", "zscore"), param = NULL)
```

Arguments

`x`	a time series or any other R data type.
`method`	method based on which the outliers are identified. Available options are '`iqr`', '`sigma`', and '`zscore`'.
`param`	parameter value for setting specific boundary criteria. Default is `NULL`.

Details

With `method = "iqr"`, the interquartile range rule for outlier identification is applied. An observation x_i will be identified as outlier if one of the following conditions fulfils:

$$x_i < q_1 - m \cdot (q_3 - q_1)$$

$$x_i > q_3 + m \cdot (q_3 - q_1)$$

where q_1 and q_3 are the 1st and 3rd quartiles of the time series `x`, respectively. `m` is the value specified by `param`. If omitted, it will be set as 1.5.

By using `method = "sigma"`, the following criteria for outlier identification, known as the 3-sigma rule, are applied:

$$x_i < \mu(x) - m \cdot \sigma(x)$$

$$x_i > \mu(x) + m \cdot \sigma(x)$$

where $\mu(x)$ and $\sigma(x)$ are the mean and standard deviation of the time series `x`, respectively. `m` is the value specified by `param`. If omitted, it will be set as 3.

The z-score rule, specified by `method = "zscore"`, compares the standardised observation values to a specific threshold:

$$\left| \frac{(x_i - \mu(x))}{\sigma(x)} \right| > m$$

where $\mu(x)$ and $\sigma(x)$ are the mean and standard deviation of the time series `x`, respectively. `m` is the value specified by `param`. If omitted, it will be set as 2. Note that 2 is the threshold for mild outliers. If checking for extreme outliers is required, the value should be set as 3.

Value

A vector indicating whether the values in `x` are outliers (`TRUE`) or not (`FALSE`).

Author(s)

Ka Yui Karl Wu

Examples

```
is.outlier(airport$Travellers, method = "zscore")
```

`predict.tsarima` *Predict Time Series Values*

Description

The function 'predict' is generic and predicts past/future values of a time series.

Usage

```
## S3 method for class 'tsarima'
predict(
  object,
  n.ahead = 1L,
  newxreg = NULL,
  newxreg.est = c("none", "x", "auto.arima"),
  se.fit = TRUE,
  alpha = 0.05,
  log = NULL,
  ...
)

## S3 method for class 'tsesm'
predict(object, n.ahead = 1L, se.fit = TRUE, alpha = 0.05, ...)

## S3 method for class 'tspredict'
print(x, ...)
```

```
## S3 method for class 'tslm'
predict(object, n.ahead = 1L, se.fit = TRUE, alpha = 0.05, ...)
```

Arguments

object	a time series or time series model for which prediction is required.
n.ahead	number of forecasting periods. Default is 1.
newxreg	new values of the regressors. Only necessary if ARIMA model is built with independent variables.
newxreg.est	character strings to indicate how the new values of the regressors in an ARIMAX model should be estimated in case they are not yet available. If newxreg is not NULL, this argument will be ignored. Available options are 'none', 'x', and 'auto.arima'. The option 'none' means that no estimation is needed, and the regressors will have no effect in future forecasts. The option 'x' means that the regressors' value will forecasted based on the same model as 'x'. The option 'auto.arima' means that the regressors' value will forecasted based on the best model found by auto.arima(). Default is 'none'.
se.fit	logical. If TRUE, standard error of each prediction will be calculated and included. Default is TRUE.
alpha	significance level. (1 - alpha) indicates is the confidence level of the prediction interval. Default is 0.05.
log	optional. A logical value indicating whether the forecasted values are log-transformed and should be inverted back to the original series scale. If the object is an tsarima model and this parameter is omitted, the value will be taken over by the settings of the model given in object. Default is NULL here.
...	additional arguments affecting the forecasts produced.
x	a 'tspredict' object.

Value

An object of class "tspredict".

The function print is used to obtain and print the prediction results, including the predicted values, the corresponding standard errors, as well as the lower and upper limit of the prediction intervals.

An object of class "tspredict" is a list usually containing the following elements:

x.time	list of time in which the series values were observed.
x.timegap	time gap between the series and forecasted values.
x.name	name of the time series for which forecasts was requested.
pred	predicted past values and forecasted future values.
se	standard errors of the forecasted values.
cil, ciu	lower and upper limits of the prediction interval.
n.ahead	number of forecasting periods.
log	logical. Indicates whether series values are log-transformed for model fitting or not. (Only available for class "tsarima")
alpha	significance level.

Author(s)

Ka Yui Karl Wu

References

Box, G. E. P., & Jenkins, G. M. (1970). Time series analysis: Forecasting and control. Holden-Day.

Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts.
https://otexts.com/fpp3/

Examples

```
predict(tsarima(airport$Travellers, order = c(1, 1, 0),
                seasonal = c(0, 1, 1), log = TRUE, include.const = TRUE),
        n.ahead = 6, alpha = 0.05)

predict(tsesm(airport$Travellers, order = "holt-winters"),
        n.ahead = 6, alpha = 0.05)
```

---

ts-functions *Extract Information of a Time Series*

Description

The function ‘tsfreq‘ extract the frequency of a time series, while ‘tstime‘ extract the time periods in which the series data were observed, and ‘tstimegap‘ returns the time gap between the observations of a time series.

The function ‘tsname‘ can be used to extract or specify the name of a time series.

Usage

```
## Extract frequency of a time series
tsfreq(x)

## Extract observation time periods of a time series
tstime(x)

## Extract time gaps between a time series' observations
tstimegap(t)

## Get or set the name of a time series
tsname(x, x.name = NULL)

## Copy Attributes from a Time Series to Another
tsattrcopy(x, x.orig)
```

Arguments

x	a univariate time series object or a numeric vector or matrix.
t	a vector or list of time in which the series values were observed.
x.name	a new name for x. If the parameter is omitted, the current name of the time series will be returned to the user.
x.orig	a univariate time series object whose attributes will be transferred to x.

Details

To set a new name for a time series, the function must be assigned to an object. Otherwise, the new name will not be taken over.

Value

For 'tstime', a list will be returned to the user with two elements: time (observation time) and frequnecy (observation frequency).

For 'tsfreq', R extracts the attribute 'seasonal.cycle' from the time series object x.

For 'tstimegap', R calculates the time gap between the time periods stored in the vector t.

So, if x and t are consistent and refer to the data and time of the same time series, the results of 'tsfreq' and 'tstimegap' as well as the frequnecy element of 'tstime' must be identical.

If x.name is NULL, the attribute series.name of x will be returned. Otherwise, the series will be returned with a new value for the attribute series.name specified by x.name.

For tsattrcopy, the function does the same as attributes. However, attributes only works if both x and x.orig share the same length, whereas tsattrcopy does not require this property and returns x with all the attributes originated from the series x.orig.

Author(s)

Ka Yui Karl Wu

See Also

time, frequency, attributes

Examples

```
tsfreq(x = airport$Travellers)
tstime(x = airport$Travellers)
tstimegap(t = airport$Date)
airport$Travellers <- tsname(airport$Travellers, x.name = "Number of Travellers in Airport X")
tsname(airport$Travellers)
tsattrcopy(airport$Travellers[1:60], airport$Travellers)
```

tsacf *Auto- Covariance and -Correlation Function Estimation*

Description

The function ‘tsacf‘ computes (and by default plots) estimates of the autocorrelation function. Function pacf is the function used for the partial autocorrelations, and function tsacov computes the autocovariance function.

Usage

```
## Autocorrelation Function (ACF)
tsacf(
  x, y = NULL, lag.max = 8,
  type = c("correlation", "covariance", "partial",
           "cross-correlation", "cross-covariance"),
  show.plot = TRUE, na.action = na.omit,
  demean = TRUE, alpha = 0.05,
  x.name = NULL, title = NULL,
  prewhiten = list(order = c(0L, 0L, 0L),
                   seasonal = c(0L, 0L, 0L),
                   period = NA))

## Partial Autocorrelation Function (PACF)
tspacf(...)

## Autocovariance (ACOV)
tsacov(...)

## cross-correlation (CCF)
tsccf(...)

## cross-ovariance (CCOV)
tsccov(...)

## S3 method for class 'tsacf'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

## S3 method for class 'tsacf'
plot(x, title = NULL, ...)
```

Arguments

x, y	univariate time series object(s) or a numeric vector(s) or matrix/matrices, or a ‘tsacf‘ object.

lag.max	maximum lag at which to calculate the acf. Default is 8. Will be automatically limited to one less than the number of observations in the series. If the series has less than 8 observations.
type	character string giving the type of acf to be computed. Allowed values are "correlation" (the default), "covariance", "partial", "cross-correlation", or "cross-covariance". Will be partially matched.
show.plot	logical. If TRUE, the acf/pacf/acov will be plotted. Default is TRUE.
na.action	function to be called to handle missing values. Default is na.omit.
demean	logical. If TRUE, the covariances will be about the sample means. Default is TRUE.
alpha	significance level. (1 - alpha) indicates is the confidence level of the prediction interval. Default is 0.05.
x.name	name of the series. If omitted here, the series name found by tsname() will be taken over here. If tsname() is NULL, the variable name will be used instead. Default is NULL.
title	character string indicating the plot title.
prewhiten	a list of seasonal ARIMA orders to prewhiten x and y for cross-correlation or cross-covariance. Only applicable if type = "cross-correlation" or type = "cross-covariance".
...	other printing or plotting parameters.
digits	number of decimal digits displayed in the results.

Details

For type = "correlation" and "covariance", the estimates are based on the sample covariance of x_t and x_{t-k} (lag 0 autocorrelation is fixed at 1 by convention.). For "cross-correlation" and "cross-covariance", the estimates are based on the sample covariance of x_t and y_{t-k}.

By default, no missing values are allowed. However, by default, na.action = na.omit, the covariances are computed only from complete cases. This means that the estimate computed may well not be a valid autocorrelation sequence, and may contain missing values. Missing values are not allowed when computing the PACF of a multivariate time series.

The partial correlation coefficient is estimated by fitting autoregressive models of successively higher orders up to lag.max.

The lag is returned and plotted in units of time, and not numbers of observations.

Different from acf, the lags here are not converted based on the seasonal cycle length. It simply reflects the time lag k between X_t and X_{t-k}. Furthermore, the ACF/PACF/ACOV/CCF/CCOV plots are created using ggplot2.

The generic functions plot and print have both methods for objects of class "tsacf".

For 'cross-correlation' and 'cross-covariance', positive lags indicate that y is shifted forward in time relative to x, while negative lags indicate y is shifted backward.

Value

An object of class "tsacf", which is a list with the following elements:

plot	bar chart of the estimated ACF/PACF/ACOV. Only available if save.plot is TRUE.
acf, pacf, acov, ccf, ccov	An array with the same dimensions as lag containing the estimated ACF/PACF/ACOV/CCF/CCOV.
clim	upper limits of the estimated confidence intervals for each lag. Since confidence intervals are symmetrical around 0, the lower limits are simply the negative values of clim.
type	type of correlation (same as the type argument).
n.used	number of observations in the time series.
lag	lags at which the acf is estimated.
x.name	name of the time series for which forecasts was requested.
alpha	significance level.

Author(s)

Ka Yui Karl Wu

References

Box, G. E. P., & Jenkins, G. M. (1970). Time series analysis: Forecasting and control. Holden-Day.

Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts.
https://otexts.com/fpp3/

Venables, W. N., & Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth Edition. Springer-Verlag.

Examples

```
# Autocorrelation (ACF)
airport.acf <- tsacf(airport$Travellers, lag.max = 24, show.plot = FALSE)
print(airport.acf, digits = 4)
plot(airport.acf)

# Partial Autocorrelation (PACF)
tspacf(airport$Travellers, lag.max = 24)

# Autocovariance (ACOV)
tsacov(airport$Travellers, lag.max = 24, show.plot = FALSE)

# Cross-Correlation (CCF)
tsccf(airport$AvgRain, airport$Travellers, lag.max = 24)

# Cross-Covariance (CCOV)
tsccov(airport$AvgRain, airport$Travellers, lag.max = 24)
```

`tsarima` *Fitting ARIMA Models*

Description

The ‘`tsarima`‘ function is used to fit an ARIMA model to a univariate time series.

Usage

```
tsarima(
  x,
  order = c(0L, 0L, 0L),
  seasonal = list(order = c(0L, 0L, 0L), period = NA),
  xreg = NULL,
  xreg.order = NA,
  include.const = TRUE,
  log = FALSE,
  train.prop = 1,
  arch.test = FALSE,
  transform.pars = TRUE,
  fixed = NULL,
  init = NULL,
  method = c("CSS-ML", "ML", "CSS"),
  SSinit = c("Gardner1980", "Rossignol2011"),
  optim.method = "BFGS",
  optim.control = list(),
  kappa = 1e+06
)

## S3 method for class 'tsarima'
print(
  x,
  digits = max(3L, getOption("digits") - 3L),
  se = TRUE,
  signif.stars = TRUE,
  ...
)

## S3 method for class 'tsarima'
summary(
  object,
  digits = max(3L, getOption("digits") - 3L),
  se = TRUE,
  signif.stars = TRUE,
  ...
)
```

Arguments

x	a univariate time series or an 'tsarima' object.
order	a specification of the non-seasonal part of the ARIMA model: the three integer components (p, d, q) are the AR order, the degree of differencing, and the MA order.
seasonal	a specification of the seasonal part of the ARIMA model (P, D, Q), the seasonal AR order, the degree of seasonal differencing, and the seasonal MA order, plus the period (which defaults to frequency(x)). This should be a list with components order and period, but a specification of just a numeric vector of length 3 will be turned into a suitable list with the specification as the order.
xreg	optional. A vector or matrix of external regressors, which must have the same number of rows as x.
xreg.order	optional. A vector, list, or numeric value of the external regressors' lags included in the model. If 'xreg' has more than one column, the number of values provided in 'xreg.order' will be assigned to the columns. the orders should be provided as a list with the entries in the same sequence as the columns in 'xreg'. If the number of values does not match the number of columns, the values in 'xreg.order' will be trimmed (too many) or repeated (too less). If omitted, no lagged version of 'xreg' will be generated. Default is NA.
include.const	logical. Indicates if the ARMA model should include a mean/intercept term. The default is TRUE for non-differenced series. For ARIMA models with differencing, it may fail to estimate the standard errors.
log	optional. A logical value indicating whether the forecasted values are log-transformed and should be inverted back to the original series scale. If the object is an tsarima model and this parameter is omitted, the value will be taken over by the settings of the model given in object. Default is NULL here.
train.prop	a numerical value specifying the proportion of training data in the series. The value must be between 0 and 1. Default is 1.
arch.test	optional. A logical value indicating whether the ARCH effect in the residuals should be tested by the McLeod-Li test of not. Default is FALSE.
transform.pars	logical. If TRUE, the AR parameters are transformed to ensure that they remain in the region of stationarity. Not used for method = "CSS". For method = "ML", it has been advantageous to set transform.pars = FALSE in some cases, see also fixed.
fixed	optional. Numeric vector of the same length as the total number of coefficients to be estimated. It should be of the form $$(\phi_1, \ldots, \phi_p, \theta_1, \ldots, \theta_q, \Phi_1, \ldots, \Phi_P, \Theta_1, \ldots, \Theta_Q, \mu)$$ where ϕ_i are the AR coefficients, θ_i are the MA coefficients, Φ_i are the seasonal AR coefficients, Θ_i are the seasonal MA coefficients and μ is the intercept term. Note that the μ entry is required if and only if include.const is TRUE. In particular it should not be present if the model is an ARIMA model with differencing. The entries of the fixed vector should consist of the values at which the user wishes to 'fix' the corresponding coefficient, or NA if that coefficient should *not*

	be fixed, but estimated. The argument `transform.pars` will be set to `FALSE` if any AR parameters are fixed. A warning will be given if `transform.pars` is set to (or left at its default) `TRUE`. It may be wise to set `transform.pars = FALSE` even when fixing MA parameters, especially at values that cause the model to be nearly non-invertible.
`init`	optional. Numeric vector of initial parameter values. Missing values will be filled in, by zeroes except for regression coefficients. Values already specified in `fixed` will be ignored.
`method`	fitting method. Maximum likelihood or minimize conditional sum-of-squares. The default (unless there are missing values) is to use conditional-sum-of-squares to find starting values, then maximum likelihood. Can be abbreviated.
`SSinit`	a string specifying the algorithm to compute the state-space initialization of the likelihood; see `KalmanLike` for details. Can be abbreviated.
`optim.method`	The value passed as the method argument to `optim`.
`optim.control`	List of control parameters for `optim`.
`kappa`	the prior variance (as a multiple of the innovations variance) for the past observations in a differenced model. Do not reduce this.
`digits`	the number of significant digits.
`se`	logical. If `TRUE`, standard error will be included in displaying the result. Default is `TRUE`.
`signif.stars`	logical. If `TRUE`, a shorthand used to indicate the statistical significance of a result will be displayed next to the p-values with *** for p < 0.001, ** for p < 0.01, * for p < 0.05, and . for p < 0.1. Default is `TRUE`.
`...`	other printing or summary parameters.
`object`	a `tsarima` object for summary.

Details

Different definitions of ARMA models have different signs for the AR and/or MA coefficients. The definition used here is the original Box & Jenkins (1970) formulation:

$$x_t = \phi_1 x_{t-1} + \ldots + \phi_p x_{t-p} + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \ldots - \theta_p \varepsilon_{t-q}$$

and so the MA coefficients differ in sign from the output of `stats::arima`. Further, if `include.const` is `TRUE` (the default for an ARMA model), this formula applies to $x_t - \mu$ rather than x_t. For ARIMA models with differencing, the differenced series usually follows a zero-mean ARMA model, but `include.const` is still available in case a constant term is required for the model. However, the estimation of the coefficients' standard error may not be successful. If an `xreg` term is included, a linear regression (with a constant term if `include.mean` is `TRUE` and there is no differencing) is fitted with an ARMA model for the error term.

The variance matrix of the estimates is found from the Hessian of the log-likelihood, and so may only be a rough guide.

Optimization is done by `optim`. It will work best if the columns in `xreg` are roughly scaled to zero mean and unit variance, but does attempt to estimate suitable scalings.

If `train.prop` is smaller than 1, the function will only treat the training part of the series as past data. When applying '`tsforecast`' or '`predict`', the forecast will start after the end of the training part of the original series.

Value

A list of class 'tsarima' with components:

coef	a vector of AR, MA and regression coefficients, which can be extracted by the coef method.
const	a value of the model's constant term. Return NULL if include.const = FALSE.
sigma2	the maximum likelihood estimate of the white noise variance.
var.coef	he estimated variance matrix of the coefficients coef, which can be extracted by the vcov method.
loglik	the maximized log-likelihood (of the differenced data), or the approximation to it used.
aic, aicc, bic	the AIC, AICc, and BIC values corresponding to the log-likelihood. Only valid for method = 'ML' fits.
error	a list of prediction error estimators, including $ME for mean error, $RMSE for root mean squared error, $MAE for mean absolute error, $MPE for mean percentage error, $MAPE for mean absolute percentage error, $MASE for mean absolute scaled error, $MASE.S for seasonal mean absolute scaled error, and $ACF1 for lag 1 autocorrelation.
arma	a vector of the ARIMA order: (p, P, q, Q, ℓ, d, D).
train.prop	proportion of training data.
x	data of the original series.
x.time	list of time in which the series values were observed.
x.timegap	time gap between the series and forecasted values.
x.name	name of the time series for which forecasts was requested.
x.time.used	list of time in which the series values were used for model fitting. It will be the same as x.time if train.prop = 1.
x.used	data of the original series which were used for model fitting. It will be the same as x if train.prop = 1.
fitted	a vector of fitted series values. If train.prop is smaller than 1, it will have the same length as x.used.
residuals	a vector of the series residuals. If train.prop is smaller than 1, it will have the same length as x.used.
exp.fitted	a vector of fitted series values after inverted from log-transformation. Only available if log = TRUE. If train.prop is smaller than 1, it will have the same length as x.used.
exp.residuals	a vector of the series residuals after inverted from log-transformation. Only available if log = TRUE. If train.prop is smaller than 1, it will have the same length as x.used.
log	logical. Indicates whether series values are log-transformed for model fitting or not.
call	the matched call.
series	series name x in match call.

code	the convergence value returned by optim.
nobs	the number of 'used' observations for the fitting, can also be extracted via nobs and is used by BIC.
model	a list representing the Kalman filter used in the fitting. See KalmanLike.
mcleod.li.test	resulting chi-square test statistics and the corresponding p-values of the McLeod-Li test for ARCH effect. Only available if arch.test = TRUE.
model.test	a list of information regarding the prediction of the testing data including 'x.test' (part of 'x' used for testing), 'fitted.test' (predicted values of the testing data), 'residuals.test' (prediction error of the testing data), and 'error.test' (prediction error measurements based on the testing data). Only available if train.prop is smaller than 1.

Fitting methods

The exact likelihood is computed via a state-space representation of the ARIMA process, and the innovations and their variance found by a Kalman filter. The initialization of the differenced ARMA process uses stationarity and is based on Gardner et. al. (1980). For a differenced process the non-stationary components are given a diffuse prior (controlled by kappa). Observations which are still controlled by the diffuse prior (determined by having a Kalman gain of at least 1e4) are excluded from the likelihood calculations. (This gives comparable results to arima0 in the absence of missing values, when the observations excluded are precisely those dropped by the differencing.)

Missing values are allowed, and are handled exactly in method 'ML'.

If transform.pars is TRUE, the optimisation is done using an alternative parametrization which is a variation on that suggested by Jones (1980) and ensures that the model is stationary. For an AR(p) model the parametrisation is via the inverse tanh of the partial autocorrelations: the same procedure is applied (separately) to the AR and seasonal AR terms. The MA terms are not constrained to be invertible during optimisation, but they will be converted to invertible form after optimisation if transform.pars is TRUE.

Conditional sum-of-squares is provided mainly for expositional purposes. This computes the sum of squares of the fitted innovations from observation n.cond on, (where n.cond is at least the maximum lag of an AR term), treating all earlier innovations to be zero. Argument n.cond can be used to allow comparability between different fits. The 'part log-likelihood' is the first term, half the log of the estimated mean square. Missing values are allowed, but will cause many of the innovations to be missing.

When regressors are specified, they are orthogonalised prior to fitting unless any of the coefficients is fixed. It can be helpful to roughly scale the regressors to zero mean and unit variance.

Author(s)

Ka Yui Karl Wu

References

Box, G. E. P., & Jenkins, G. M. (1970). Time series analysis: Forecasting and control. Holden-Day.

Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts.
https://otexts.com/fpp3/

Hyndman, R. J., Athanasopoulos, G., Bergmeir, C., Caceres, G., Chhay, L., O'Hara-Wild, M., Petropoulos, F., Razbash, S., Wang, E., & Yasmeen, F. (2025). *forecast: Forecasting functions for time series and linear models*. R package version 8.24.0,
https://pkg.robjhyndman.com/forecast/.

Hyndman, R. J., & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*, **27**(3), 1-22. doi:10.18637/jss.v027.i03.

Brockwell, P. J., & Davis, R. A. (1996). Introduction to Time Series and Forecasting. Springer, New York. Sections 3.3 and 8.3.

Durbin, J., & Koopman, S. J. (2001). Time Series Analysis by State Space Methods. Oxford University Press.

Gardner, G, Harvey, A. C., & Phillips, G. D. A. (1980). Algorithm AS 154: An algorithm for exact maximum likelihood estimation of autoregressive-moving average models by means of Kalman filtering. *Applied Statistics*, **29**, 311-322. doi:10.2307/2346910.

Harvey, A. C. (1993). Time Series Models. 2nd Edition. Harvester Wheatsheaf. Sections 3.3 and 4.4.

Jones, R. H. (1980). Maximum likelihood fitting of ARMA models to time series with missing observations. *Technometrics*, **22**, 389-395. doi:10.2307/1268324.

Ripley, B. D. (2002). Time series in R 1.5.0. *R News*, **2**(2), 2-7. https://www.r-project.org/doc/Rnews/Rnews_2002-2.pdf

See Also

arima

Examples

```
tsarima(airport$Travellers,
        order = c(1, 1, 0), seasonal = c(0, 1, 1),
        log = TRUE, include.const = TRUE)
```

`tsboxplot` *Box Plots*

Description

Produce box-and-whisker plot of a given univariate time series.

Usage

```
tsboxplot(
  x,
  title = NULL,
  x.name = NULL,
  x.col = "darkgrey",
  mean.col = "steelblue4"
)
```

Arguments

x	a univariate time series object or a numeric vector or matrix.
title	title of the box plot
x.name	name of the series. If omitted here, the series name found by tsname() will be taken over here. If tsname() is NULL, the variable name will be used instead. Default is NULL.
x.col	line colour of the box plot.
mean.col	colour of the dot indicating the series mean.

Details

Since x is a univariate time series, no parallel box plots can be plotted here.

Missing values are ignored when forming boxplots.

Value

A boxplot of x will be displayed with no further values or objects returned.

Author(s)

Ka Yui Karl Wu

References

Chambers, J. M., Cleveland, W. S., Kleiner, B., & Tukey, P. A. (1983). Graphical Methods for Data Analysis. Wadsworth & Brooks/Cole.

Examples

```
tsboxplot(airport$Travellers)
```

`tsconvert` *Convert One-Dimensional Data to Time Series*

Description

The function ‘`tsconvert`‘ is used to convert any one-dimensional vector/list into a time-series objects.

Usage

```
tsconvert(x, t, frequency = NULL, format = NULL, x.name = NULL)
```

Arguments

`x`	a univariate time series object or a numeric vector or matrix.
`t`	a vector or list of time in which the series values were observed.
`frequency`	the number of observations per unit of time. If omitted, R will identify the frequency based on the time vector specified in `t`. Default is `NULL`.
`format`	time format specified for the variable provided in `t`. If omitted, R will identify this automatically. Default is `NULL`.
`x.name`	name of the series. If omitted here, the series name found by `tsname()` will be taken over here. If `tsname()` is `NULL`, the variable name will be used instead. Default is `NULL`.

Details

The function `tsconvert` is used to convert vectors/lists into time-series objects. These are vectors or matrices which inherit from class "ts" (and have additional attributes) and represent data sampled at equispaced points in time. Time series must have at least one observation, and although they need not be numeric there is very limited support for non-numeric series.

Argument `frequency` indicates the sampling frequency of the time series, with the default value 1 indicating one sample in each unit time interval. For example, one could use a value of 7 for frequency when the data are sampled daily, and the natural time period is a week, or 12 when the data are sampled monthly and the natural time period is a year. Values of 4 and 12 are assumed in (e.g.) print methods to imply a quarterly and monthly series respectively. Note that `frequency` does not need to be a whole number: for example, `frequency = 0.2` would imply sampling once every five time units.

Value

`x` is returned as a ‘`ts`‘ object with the attributes ‘`tsp`‘ (start, end, and frequency of `x`), ‘`series.name`‘ (series name, *optional*), and ‘`seasonal.cycle`‘ (time gap between series observations).

Author(s)

Ka Yui Karl Wu

See Also

attributes

Examples

```
travellers <- tsconvert(x = airport$Travellers, t = airport$Date)
```

---

tsdecomp *Decompose a Time Series*

Description

Decompose a time series into trend, cyclical, seasonal and irregular components. Deals with additive or multiplicative components.

Usage

```
tsdecomp(
  x,
  type = c("additive", "multiplicative"),
  trend.method = c("lm", "loess"),
  tcc.order = 3,
  x.name = NULL,
  show.plot = TRUE,
  ...
)

## S3 method for class 'tsdecomp'
print(
  x,
 decomp.incl = c("all", "tc", "tcc", "cc", "detrend", "ic", "scadj", "sc", "sceffect"),
  ...
)

## S3 method for class 'tsdecomp'
plot(
  x,
 plot.incl = c("all", "tc", "tcc", "cc", "detrend", "ic", "scadj", "sc", "sceffect"),
  ...
)
```

Arguments

x	a time series for which decomposition is required or a 'tsdecomp' object.
type	type of the time series components. Available options are 'additive' and 'multiplicative'. Can be abbreviated. Default is 'additive'.

trend.method	estimating method of the trend. Available options are 'lm' (linear) and 'loess' (non-linear). Default is 'lm'.
tcc.order	moving average order for the estimation of the trend-cycle component.
x.name	name of the series. If omitted here, the series name found by tsname() will be taken over here. If tsname() is NULL, the variable name will be used instead. Default is NULL.
show.plot	logical. If TRUE, forecasting plot will be displayed directly. Default is TRUE.
...	parameter values that can affect the time series decomposition plots.
decomp.incl	time series components that should be printed. Available options are 'all' (default), 'tc' (trend), 'tcc' (trend-cycles), 'tcc' (cycles), 'detrend', 'ic' (irregular), 'scadj' (seasonally adjusted), 'sc' (seasonality), and 'sceffect' (seasonal effects).
plot.incl	time series components that should be plotted. Available options are 'all' (default), 'tc' (trend), 'tcc' (trend-cycles), 'tcc' (cycles), 'detrend', 'ic' (irregular), 'scadj' (seasonally adjusted), 'sc' (seasonality), and 'sceffect' (seasonal effects). Ignored if show.plot = FALSE.

Details

The additive model used is:

$$Y_t = T_t + C_t + S_t + I_t$$

The multiplicative model used is:

$$Y_t = T_t \cdot C_t \cdot S_t \cdot I_t$$

The function first determines the trend-cycle component using a moving average, and removes it from the time series. Then, the seasonal figure is computed by averaging, for each time unit, over all periods. The seasonal figure is then centred. Finally, the error component is determined by removing trend and seasonal figure (recycled as needed) from the original time series.

This only works well if 'x' covers an integer number of complete periods.

The function 'plot' generates the following plots:

Trend	a time series line plot together with a trend line.
Trend-Cycles	a time series line plot together with the trend-cycle component (moving averages).
Cycles	a line plot of the trend-cycle component (moving averages).
Detrended	a line plot of the time series after removing the trend component.
Irregular	a line plot of the irregular component.
Seasonally Adjusted	a line plot of the time series after removing the seasonal component.
Seasonal	a line plot of the seasonal component.
Seasonal Effect	a line plot of the overall estimated effect of each season in the time series.

Value

An object of class "tsdecomp" with following components:

x	original series data

x.time	list of time in which the series values were observed.
x.timegap	time gap between the series and forecasted values.
x.name	name of the time series for which forecasts was requested.
trend	value of the trend component for each observation.
cycle	value of the cyclical component for each observation.
trend.cycle	trend-cyclical component value of each observation.
detrended	value of each observation after removing the trend component.
seasonal	value of the seasonal component for each observation.
seasonal.adjusted	value of each observation after removing the seasonal component
random	value of the irregular component for each observation.
seasonal.effect	value expressing the estimated overall effect of each season in the time series.
type	type of the time series decomposition.

Author(s)

Ka Yui Karl Wu

References

Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts.
https://otexts.com/fpp3/

Kendall, M., & Stuart, A. (1983) The Advanced Theory of Statistics, Vol.3, Griffin. pp. 410-414.

Examples

```
tsdecomp(airport$Travellers, type = "multiplicative")
```

---

tsdiff *Difference a Time Series*

Description

The function ‘tsdiff‘ generates the differenced series of a non-stationary time series.

Usage

```
tsdiff(x, lag = 1L, order = 1L, lag.D = 0L, order.D = 0L)
```

Arguments

x	a univariate time series object or a numeric vector or matrix.
lag	number of lags for non-seasonal differencing. Default is 1.
order	order of non-seasonal differencing. Default is 1.
lag.D	number of lags for seasonal differencing. Default is 0.
order.D	order of seasonal differencing. Default is 0.

Details

The parameters lag and lag.D are only necessary if the lag difference for differencing is not 1 or ℓ, the seasonal cycle length, respectively. If order.D > 0 but lag.D is omitted, R will use the frequency of the series for this parameter.

Value

The differences of x will be returned with all the attributes being carried over. Unlike the function diff, the output series by tsdiff has the same length as the original series by adding NA to those observations at the beginning of the series for which no differencing can be carried out.

Author(s)

Ka Yui Karl Wu

See Also

diff

Examples

```
travellers_d <- tsdiff(x = airport$Travellers, order = 1, order.D = 1)
tsexplore(travellers_d, lag.max = 60)
```

---

tsesm *Exponential Smoothing Forecasts*

Description

The 'tsesm' function forecasts future values of a univariate time series using exponential smoothing.

Usage

```
tsesm(
  x,
  order = c("simple", "holt", "holt-winters"),
  damped = FALSE,
  initial = c("optimal", "simple"),
  type = c("additive", "multiplicative"),
  alpha = NULL,
  beta = NULL,
  gamma = NULL,
  lambda = NULL,
  phi = NULL,
  biasadj = FALSE,
  exp.trend = FALSE,
  seasonal.period = NULL,
  train.prop = 1
)

## S3 method for class 'tsesm'
print(x, ...)

## S3 method for class 'tsesm'
summary(object, ...)
```

Arguments

x	a univariate time series or a ‘tsesm‘ object.
order	a specification of the exponential smoothing method. The available options are "simple", "holt", "holt-winters".
damped	logical. If TRUE, a damped trend is used. Default is FALSE.
initial	method used for selecting initial state values. Available options are ‘optimal‘ and ‘simple‘. If ‘optimal‘, the initial values are optimised along with the smoothing parameters using ‘ets‘. If ‘simple‘, the initial values are set to values obtained using simple calculations on the first few observations.
type	specify whether the time series is ‘additive‘ or ‘multiplicative‘.
alpha	value of smoothing parameter for the level. If NULL, it will be estimated.
beta	value of smoothing parameter for the trend. If NULL, it will be estimated.
gamma	value of smoothing parameter for the seasonal component If NULL, it will be estimated.
lambda	parameter of the Box-Cox transformation. If lambda = "auto", a transformation is automatically selected using ‘BoxCox.lambda‘. The transformation is ignored if NULL. Otherwise, data transformed before model is estimated.
phi	value of damping parameter if damped = TRUE. If NULL, it will be estimated.

biasadj	use adjusted back-transformed mean for Box-Cox transformations. If transformed data is used to produce forecasts and fitted values, a regular back transformation will result in median forecasts. If biasadj == TRUE, an adjustment will be made to produce mean forecasts and fitted values.
exp.trend	logical. If TRUE, an exponential trend is fitted. Otherwise, the trend is (locally) linear. Default is FALSE.
seasonal.period	number of seasons within a seasonal cycle. If NULL, the value of frequency(x) will be taken. Default is NULL.
train.prop	a numerical value specifying the proportion of training data in the series. The value must be between 0 and 1. Default is 1.
...	other printing parameters
object	a 'tsesm' object to summarise.

Details

The 'tsesm' function uses the same mechanism for exponential smoothing like 'forecast::ses', 'forecast::holt', and 'forecast::hw'. Instead of using three different functions, all of them are integrated in tsesm, and user can choose the method by specifying the value of the order parameter.

The option 'simple' is the lowest exponential smoothing order, or the first exponential smoothing order (that's why the parameter is called 'order' and not 'method'). It can be used to forecast series with only level (ℓ) component. The corresponding forecasting formula is given by:

$$\tilde{x}_{t+h|t} = \ell_t = \alpha x_{t-1} + (1-\alpha)\ell_{t-1},$$

where α is the smoothing parameter for the level component, and h is the number of periods ahead for forecasting.

With the 'holt' option, the second exponential smoothing order can be chosen. It is suitable for the forecasting of series with level (ℓ) and trend (b) components.

$$\ell_t = \alpha x_t + (1-\alpha)(\ell_{t-1} + b_{t-1})$$

$$b_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta)b_{t-1}$$

$$\tilde{x}_{t+h|t} = \ell_t + hb_t$$

where α and β are the smoothing parameters for the level and trend components, respectively, h is the number of periods ahead for forecasting.

The third exponential smoothing order can be specified by the 'holt-winters' option. It can be used to forecast time series with level (ℓ), trend (b), and seasonal (s) components.

$$\ell_t = \alpha(x_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})$$

$$b_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta)b_{t-1}$$

$$s_t = \gamma(x_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}$$

$$\tilde{x}_{t+h|t} = \ell_t + hb_t + s_{t+h-m(k+1)}$$

where α, β, and γ are the smoothing parameters for the level, trend, and seasonal components, respectively, h is the number of periods ahead for forecasting, m is the number of periods within a seasonal cycle, and k is the integer part of $(h-1)/m$, which ensures that the estimates of the seasonal indices used for forecasting come from the final period of the series.

If `train.prop` is smaller than 1, the function will only treat the training part of the series as past data. When applying ‘`tsforecast`‘ or ‘`predict`‘, the forecast will start after the end of the training part of the original series.

Value

A list of class "`tsesm`" with components:

`coef`	a vector of smoothing parameters.
`m`	number of seasons in the series, usually equivalent to the series' frequency obtained by `tsfreq`.
`components`	values used for fitting exponential smoothing model.
`states`	estimated values of all smoothing parameter for each observation.
`initstate`	initial values of the smoothing parameters.
`sigma2`	residual variance.
`loglik`	the maximized log-likelihood, or the approximation to it used.
`aic`, `aicc`, `bic`	the AIC, AICc, and BIC values corresponding to the log-likelihood. Only valid for method = "ML" fits.
`x`	original series data or a ‘`tsesm`‘ object.
`x.time`	list of time in which the series values were observed.
`x.timegap`	time gap between the series and forecasted values.
`x.name`	name of the time series for which forecasts was requested.
`fitted`	a vector of fitted series values.
`residuals`	a vector of the series residuals.
`damped`	logical value indicating whether damped trend was used or not.
`initial`	method used for selecting initial state values.
`type`	indicator of an additive or multiplicative time series.
`exp.trend`	indicator of the use of exponential trend.
`lambda`	parameter value of the Box-Cox transformation.
`biasadj`	indicator of the use of adjusted back-transformed mean for Box-Cox transformations.
`series`	series name `x` in match call.
`call`	the matched call.

`error` a list of prediction error estimators, including `$ME` for mean error, `$RMSE` for root mean squared error, `$MAE` for mean absolute error, `$MPE` for mean percentage error, `$MAPE` for mean absolute percentage error, `$MASE` for mean absolute scaled error, `$MASE.S` for seasonal mean absolute scaled error, and `$ACF1` for lag 1 autocorrelation.

`model.test` a list of information regarding the prediction of the testing data including '`x.test`' (part of '`x`' used for testing), '`fitted.test`' (predicted values of the testing data), '`residuals.test`' (prediction error of the testing data), and '`error.test`' (prediction error measurements based on the testing data). Only available if `train.prop` is smaller than 1.

Author(s)

Ka Yui Karl Wu

References

Box, G. E. P., & Jenkins, G. M. (1970). Time series analysis: Forecasting and control. Holden-Day.

Hyndman, R. J., Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts. https://otexts.com/fpp3/

Hyndman, R. J., Athanasopoulos, G., Bergmeir, C., Caceres, G., Chhay, L., O'Hara-Wild, M., Petropoulos, F., Razbash, S., Wang, E., Yasmeen, F. (2025).
forecast: Forecasting functions for time series and linear models. R package version 8.24.0, https://pkg.robjhyndman.com/forecast/.

Hyndman, R. J., Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*, **27**(3), 1-22.

Examples

```
tsesm(airport$Travellers, order = "simple")
tsesm(airport$Travellers, order = "holt")
tsesm(airport$Travellers, order = "holt-winters")
```

---

tsexplore *Explore a Time Series Numerically and Graphically*

Description

The function '`tsexplore`' generates statistics, various tests, and graphs regarding the location, deviation, distribution of a time series.

Usage

```
tsexplore(
  x,
  show.plot = TRUE,
  x.name = NULL,
  mu = 0,
  adf.lag = 0,
  lag.max = 8,
  ...
)

## S3 method for class 'tsexplore'
plot(
  x,
  trend = c("linear", "smooth", "none"),
  histbin = 15,
  lwidth = 0.7,
  pwidth = 0.7,
  x.col = "darkgrey",
  extra.col = "red",
  plot.incl = c("all", "line", "hist", "box", "qq", "acf", "pacf"),
  ...
)

## S3 method for class 'tsexplore'
print(
  x,
  digits = max(3L, getOption("digits") - 3L),
  stats.incl = c("all", "stats", "var", "qtls", "acf", "tests"),
  ...
)
```

Arguments

x	a time series to be explored or a ‘tsexplore‘ object.
show.plot	logical. If TRUE, all exploration charts will be displayed directly. Default is TRUE.
x.name	name of the series. If omitted here, the series name found by tsname() will be taken over here. If tsname() is NULL, the variable name will be used instead. Default is NULL.
mu	test value specified under the null hypothesis of the t-test for the mean location. Default is 0.
adf.lag	number of AR lags included in the ADF test. Default is 0.
lag.max	maximum lag at which to calculate the acf. Default is 8. Will be automatically limited to one less than the number of observations in the series. If the series has less than 8 observations.
...	parameter values that can affect the plots created for time series exploration.

trend	a character string indicating whether and how the trend line should be fitted in the time series line plot. Available options are 'linear', 'smooth', 'none'. Default is 'linear'.
histbin	a numeric value to specify the number of bins in the histogram. Can be omitted. Default is 15.
lwidth	line width of the series line plot. Default is 0.7.
pwidth	size of the markers in the QQ plot. Default is 0.7.
x.col	line colour of the time series line plot. Default is 'darkgrey'.
extra.col	colour of extra information in the plots. Default is 'red'.
plot.incl	time series components that should be plotted. Available options are 'all' (default), 'line' (line plot), 'hist' (histogram), 'box' (boxplot), 'qq' (QQ plot), 'acf' (ACF plot), and 'pacf' (PACF plot). Ignored if show.plot = FALSE.
digits	the number of significant digits.
stats.incl	time series statistics that should be printed. Available options are 'all' (default), 'stats' (statistics), 'var' (variability), 'qtls' (quantiles), 'acf' (autocorrelation), and 'tests' (tests).

Value

An object of class "tsexplore" with following components:

x	original series data
x.time	list of time in which the series values were observed.
x.timegap	time gap between the series and forecasted values.
x.name	name of the time series for which forecasts was requested.
stats	a list of statistics and test results conducted on the time series

Details of the 'stats' component

The following statistics and test results are stored in the component 'stats' of the 'tsexplore' object:

statistics	n (number of observations), nvalid (number of valid observations), sum, mean, median, skewness, ku
variability	variance, sd (standard deviation), range, iqr (interquartile range)
quantiles	minimum, q1 (1st quartile), median, q3 (3rd quartile), maximum
autocorrelation	acf (autocorrelation function), pacf (partial autocorrelation function) - from lag 0 (ACF) or lag 1 (PA
tests	location (t-test), normality (Shapiro-Wilk-test), stationarity (ADF-test), independence (Ljung

Author(s)

Ka Yui Karl Wu

References

Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts.
https://otexts.com/fpp3/

Examples

```
tsexplore(airport$Travellers, lag.max = 24, histbin = 10)
```

`tsforecast` *Forecast Time Series based on Fitted Models*

Description

The generic function ‘`tsforecast`‘ is used for forecasting time series or time series models.

Usage

```
## S3 method for default
tsforecast(object, n.ahead = 1,
  alpha = 0.05,
  show.plot = TRUE,
  forecast.incl = c("all", "forecast", "predict"),
  nobs.incl = NULL,
  log = NULL,
  newxreg = NULL,
  newxreg.est = c("none", "x", "auto.arima"),
  x.name = NULL, pred.name = "Forecasts", ...)

## S3 method for class 'tsarima'
tsforecast(object, ...)

## S3 method for class 'tsesm'
tsforecast(object, ...)

## S3 method for class 'tslm'
tsforecast(object, ...)

## S3 method for class 'tsforecast'
print(x, ...)

## S3 method for class 'tsforecast'
plot(
  x,
  ylim = NULL,
  title = NULL,
  x.lwidth = 0.7,
  pred.lwidth = 0.7,
  x.col = "darkgrey",
  pred.col = "red",
  ci.col = "royalblue",
  ...
)
```

Arguments

object	a time series or time series model for which forecasts are required.
n.ahead	number of forecasting periods. Default is 1.
alpha	significance level. (1 - alpha) indicates is the confidence level of the prediction intervals. Default is 0.05.
show.plot	logical. If TRUE, forecasting plot will be displayed directly. Default is TRUE.
forecast.incl	character string giving which part of the series should be predicted or forecasted. Default value is "all", which indicates that predicted values of the entire series, together with the forecasted values of the number of future periods specified with n.ahead, will be included. Specify with "forecast" if only forecasts of future periods should be included, and "predict" is used if only fitted values of past periods should be included in the output.
nobs.incl	number of past observations for which the predicted values should be included in the output. If forecast.incl is set as "forecast", the value of this parameter will be ignored. Default is NULL,
log	optional. A logical value indicating whether the forecasted values are log-transformed and should be inverted back to the original series scale. If the object is an tsarima model and this parameter is omitted, the value will be taken over by the settings of the model given in object. Default is NULL here.
newxreg	new values of the regressors. Only necessary if ARIMA model is built with independent variables.
newxreg.est	character strings to indicate how the new values of the regressors in an ARIMAX model should be estimated in case they are not yet available. If newxreg is not NULL, this argument will be ignored. Available options are 'none', 'x', and 'auto.arima'. The option 'none' means that no estimation is needed, and the regressors will have no effect in future forecasts. The option 'x' means that the regressors' value will forecasted based on the same model as 'x'. The option 'auto.arima' means that the regressors' value will forecasted based on the best model found by auto.arima(). Default is 'none'.
x.name	name of the series. If omitted here, the series name found by tsname() will be taken over here. If tsname() is NULL, the variable name will be used instead. Default is NULL.
pred.name	name of the forecasted series here. Default is 'Forecasts'.
...	additional arguments affecting the forecasts produced.
x	an object of class 'tsforecast'.
ylim	value limit of the y-axis. The values should be specified in c(lower_limit, upper_limit), where lower_limit and upper_limit are the values of the smallest and largest number of the y-axis, respectively. Default is NULL.
title	title of the forecasting chart. Default is NULL.
x.lwidth	line width of the original series in the output plot. Default is 0.7.
pred.lwidth	line width of the predicted past values or forecasted future values in the output plot. Default is 0.7.
x.col	colour of the original series in the output plot. Default is 'darkgrey'.

pred.col	colour of the predicted past values or forecasted future values in the output plot. Default is 'red'.
ci.col	colour of the forecasted prediction interval in the output plot. Default is 'royalblue'.

Value

The function print is used to obtain and print the forecasting results, while the function plot produces a plot of the forecasts and prediction intervals.

An object of class "tsforecast" is a list usually containing the following elements:

x	original series data. Note that for future periods, no data can be displayed, and the corresponding cells are therefore blank.
x.time	list of time in which the series values were observed.
x.timegap	time gap between the series and forecasted values.
x.name	name of the time series for which forecasts was requested.
pred	predicted past values and forecasted future values.
pred.time	list of time in which the predictions/forecasts were estimated.
pred.name	name of the series containing the predicted/forecasted values.
se	standard errors of the forecasted values. Note that standard errors are only calculated for future periods. The corresponding cells for past observations are therefore blank.
cil, ciu	lower and upper limits of the prediction interval. Note that confidence (prediction) intervals are only calculated for future periods. The corresponding cells for past observations are therefore blank.
n.ahead	number of forecasting periods.
forecast.incl	indication of the series part that should be predicted or forecasted.
log	logical. Indicates whether series values are log-transformed for model fitting or not.
alpha	significance level.

Author(s)

Ka Yui Karl Wu

References

Box, G. E. P., & Jenkins, G. M. (1970). Time series analysis: Forecasting and control. Holden-Day.

Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts.
https://otexts.com/fpp3/

Examples

```
tsforecast(tsarima(airport$Travellers, order = c(1, 1, 0),
                   seasonal = c(0, 1, 1), log = TRUE, include.const = TRUE),
           n.ahead = 6, forecast.incl = "all", nobs.incl = 12,
           title = "Forecast of Travellers by ARIMA")

tsforecast(tsesm(airport$Travellers, order = "holt-winters"),
           n.ahead = 6, title = "Forecast of Travellers by Holt-Winters")
```

---

tshistogram *Histograms*

Description

Produce a histogram of a given univariate time series.

Usage

```
tshistogram(
  x,
  bins = NULL,
  density = FALSE,
  density.lwidth = 0.7,
  title = NULL,
  x.name = NULL,
  x.col = "darkgrey",
  density.col = "steelblue4"
)
```

Arguments

x	a univariate time series object or a numeric vector or matrix.
bins	a numeric value to specify the number of bins in the histogram. Can be omitted. Default is NULL.
density	logical. Indicate whether the density curve of the normal distribution should be included. Default is FALSE.
density.lwidth	line width of the density curve in the output plot. Will be ignored if density = FALSE. Default is 0.7.
title	title of the histogram. Default is NULL.
x.name	name of the series. If omitted here, the series name found by tsname() will be taken over here. If tsname() is NULL, the variable name will be used instead. Default is NULL.
x.col	colour of the histogram bars. Default is 'darkgrey'.
density.col	colour of the density curve. Will be ignored if density = FALSE. Default is 'steelblue4'.

Value

A histogram of x will be displayed with no further values or objects returned.

Author(s)

Ka Yui Karl Wu

References

Venables, W. N., & Ripley. B. D. (2002) Modern Applied Statistics with S. Springer.

Examples

```
tshistogram(airport$Travellers, density = TRUE)
```

---

tslag *Lag a Time Series*

Description

The function ‘tslag‘ back- or foreshifts a time series and generates the lagged version of it.

Usage

```
tslag(x, lag = 1L, return.matrix = FALSE)
```

Arguments

x	a univariate time series object.
lag	number of lags (in units of observations). Default is 1. If multiple values are given as a vector, multiple lagged version of the series will be returned as a matrix.
return.matrix	optional. A boolean value indicating whether the lagged version of ‘x‘ should be returned as a matrix. Only meaningful if lag has only one value.

Details

Note the sign of lag: a series lagged by a positive lag starts earlier.

Value

The same time series object as x after being back- or foreshifted for the number of periods specified in lag.

Author(s)

Ka Yui Karl Wu

See Also

lag

Examples

```
tslag(airport$Travellers, lag = 12)
```

`tslineplot` *Time Series Line Plots*

Description

Produce line plot of a given univariate time series.

Usage

```
tslineplot(
  x,
  t = NULL,
  pred = NULL,
  pred.t = NULL,
  cil = NULL,
  ciu = NULL,
  ci.t = NULL,
  trend = c("none", "linear", "smooth"),
  title = NULL,
  x.name = NULL,
  pred.name = "Predicted",
  x.lwidth = 0.7,
  pred.lwidth = 0.7,
  x.col = "darkgrey",
  pred.col = "steelblue4",
  ci.col = "royalblue",
  t.name = "Date/Time",
  t.text.angle = 90,
  t.numbreak = 10,
  ylim = NULL
)
```

Arguments

`x`	a univariate time series object or a numeric vector or matrix.
`t`	a vector or list of time periods in which the series values were observed. The length of `t` must be identical to the length of `x`.
`pred`	a vector of predicted or forecasted values of a univariate time series. This parameter can be omitted. Default is `NULL`.

pred.t	a vector of time periods in which the predicted or forecasted values of a univariate time series are estimated. This parameter can be omitted. Default is NULL.
cil	a vector of the prediction intervals' lower limits. Only necessary if pred and pred.t are provided. This parameter can be omitted. Default is NULL.
ciu	a vector of the prediction intervals' upper limits. Only necessary if pred and pred.t are provided. This parameter can be omitted. Default is NULL.
ci.t	a vector of the time periods in which the prediction intervals are estimated.This parameter can be omitted. Default is NULL.
trend	indicate whether a trend line should be included in the time series plot. Available options are 'none', 'linear', and 'smooth'. If 'linear', a straight trend line estimated by linear regression model will be included. If 'smooth', the trend line will be estimated by LOESS regression model. Default is NULL, indicating that no trend line should be displayed.
title	title of the time series line plot.
x.name	name of the series. If omitted here, the series name found by tsname() will be taken over here. If tsname() is NULL, the variable name will be used instead. Default is NULL.
pred.name	name of the series' predicted/forecasted values. Only necessary if pred and pred.t are provided. Default is 'Predicted'.
x.lwidth	line width of the series line plot. Default is 0.7.
pred.lwidth	line width of the line plot for the predicted/forecasted values. Default is 0.7.
x.col	line colour of the time series line plot. Default is 'darkgrey'.
pred.col	line colour of the line plot for the predicted/forecasted values. Default is 'steelblue4'.
ci.col	area colour of the prediction intervals. Default is 'royalblue'.
t.name	name of the x-axis (time axis). Default is 'Date/Time'.
t.text.angle	angle of the tick labels on the x-axis (time axis). Default is 90 (vertical).
t.numbreak	number of tick labels on the x-axis (time axis). Default is 10.
ylim	value limit of the y-axis. The values should be specified in c(lower_limit, upper_limit), where lower_limit and upper_limit are the values of the smallest and largest number of the y-axis, respectively. Default is NULL.

Details

If x is a ts object, parameter t can be omitted. You can convert a vector, a matrix or data frame column using tsconvert to a ts object.

Value

A line plot of x will be displayed with no further values or objects returned.

Author(s)

Ka Yui Karl Wu

Examples

```
tslineplot(airport$Travellers, trend = "linear")
```

`tslm` *Generate Time Series Regression Model*

Description

Fit Time Series Regression Models.

Usage

```
tslm(
  x,
  trend.order = 1,
  seasonal = FALSE,
  period = NA,
  type = c("additive", "multiplicative"),
  train.prop = 1
)

## S3 method for class 'tslm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

## S3 method for class 'tslm'
summary(
  object,
  anova = TRUE,
  digits = max(3L, getOption("digits") - 3L),
  se = TRUE,
  signif.stars = TRUE,
  ...
)
```

Arguments

`x`	a univariate time series or a ‘`tslm`‘ object.
`trend.order`	an integer specifying the polynomial order of the trend line estimation. If `trend.order = 0`, no trend component will be included in the model. Default is 1.
`seasonal`	logical. If `TRUE`, seasonal component will be included in the model. Default is `FALSE`.
`period`	a numerical value specifying the seasonal cycle length of the series. If omitted, `frequency(x)` will be used here. Only effective if `seasonal = TRUE`. Default is `NA`.

type	string characters specifying the series type. Available options are 'additive' and 'multiplicative'. If 'type = multiplicative', interaction terms between trend and seasonal components will be added to the model. Default is 'additive'.
train.prop	a numerical value specifying the proportion of training data in the series. The value must be between 0 and 1. Default is 1.
digits	the number of significant digits to use when printing
...	other printing or summary parameters.
object	a tslm object for summary
anova	logical. If TRUE, an anova table with significance tests for trend, seasonality and their interaction will be included in the summary.
se	logical. If TRUE, standard error will be included in displaying the result. Default is TRUE.
signif.stars	logical. If TRUE, 'significance stars' are printed for each coefficient

Value

An object of class 'tslm' is a list containing at least the following components:

coefficients	a named vector of coefficients
residuals	the residuals, that is response minus fitted values
rank	the numeric rank of the fitted linear model
fitted.values	the fitted mean values
df.residual	the residual degrees of freedom
call	the matched call
terms	the terms object used
xlevels	(only where relevant) a record of the levels of the factors used in fitting
offset	the offset used (missing if none were used)
model	if requested (the default), the model frame used
train.prop	proportion of training data.
x	data of the original series.
x.time	list of time in which the series values were observed.
x.timegap	time gap between the series and forecasted values.
x.name	name of the time series for which forecasts was requested.
x.time.used	list of time in which the series values were used for model fitting. It will be the same as x.time if train.prop = 1.
x.used	data of the original series which were used for model fitting. It will be the same as x if train.prop = 1.
series	series name x in match call.

`error` a list of prediction error estimators, including `$ME` for mean error, `$RMSE` for root mean squared error, `$MAE` for mean absolute error, `$MPE` for mean percentage error, `$MAPE` for mean absolute percentage error, `$MASE` for mean absolute scaled error, `$MASE.S` for seasonal mean absolute scaled error, and `$ACF1` for lag 1 autocorrelation.

`model.test` a list of information regarding the prediction of the testing data including '`x.test`' (part of '`x`' used for testing), '`fitted.test`' (predicted values of the testing data), '`residuals.test`' (prediction error of the testing data), and '`error.test`' (prediction error measurements based on the testing data). Only available if `train.prop` is smaller than 1.

Author(s)

Ka Yui Karl Wu

References

Hyndman, R. J., Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts. https://otexts.com/fpp3/

Examples

```
tslm(airport$Travellers, trend.order = 2,
     seasonal = TRUE, type = "multiplicative")
```

---

tsmltest *McLeod-Li Test for ARCH Effect*

Description

The function '`tsmltest`' applies the McLeod-Li test to examine whether ARCH effect exists in the squared residuals of an ARIMA model.

Usage

```
tsmltest(object, lag.max = NULL)
```

Arguments

`object` a univariate time series object or a numeric vector or matrix.

`lag.max` maximum lag at which to examine the ARCH effect. Default is `NULL`. Will be automatically calculated using the formula $10 \cdot \log_{10}(n)$, where n is the series length, if the parameter is omitted.

Value

A list with two elements:

`test.value`	chi-square test statistics of the McLeod-Li test for the lags 1 to `lag.max`.
`p.value`	corresponding p-values of the McLeod-Li test for the lags 1 to `lag.max`.

Author(s)

Ka Yui Karl Wu

References

McLeod, A. I., & Li, W. K. (1983). Diagnostic Checking ARMA Time Series Models Using Squared-Residual Autocorrelations. *Journal of Time Series Analysis*, **4**(4), 269-273. doi:10.1111/j.14679892.1983.tb00373.x.

Examples

```
tsmltest(tsarima(airport$Travellers, order = c(1, 1, 0),
                 seasonal = c(0, 1, 1), log = TRUE, include.const = TRUE))
```

---

tsmodeleval *Goodness of Fit of a Time Series Model*

Description

The function ‘`tsmodeleval`‘ can be used to evaluate the goodness of fit of a time series model.

Usage

```
tsmodeleval(object)
```

Arguments

`object`	a time series model of class ‘`tsarima`‘, ‘`tsesm`‘, or ‘`tsmovav`‘. It can also be a list with at least the two elements: ‘`x`‘ and ‘`fitted`‘.

Value

A list with the following model evaluation criteria:

`ME`	mean error
`RMSE`	Root mean square error
`MAE`	mean absolute error
`MPE`	mean percentage error
`MAPE`	mean absolute percentage error
`MASE`	mean absolute scaled error
`MASE.S`	seasonal mean absolute scaled error
`ACF1`	lag 1 autocorrelation

Author(s)

Ka Yui Karl Wu

References

Hyndman, R. J., Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts. https://otexts.com/fpp3/

Examples

```
tsmodeleval(tsarima(airport$Travellers,
                    order = c(1, 1, 0), seasonal = c(0, 1, 1),
                    log = TRUE, include.const = TRUE))
```

`tsmovav` *Generate Moving Averages of a Time Series*

Description

The function ‘tsmovav‘ calculates the moving averages of a time series.

Usage

```
tsmovav(
  x,
  order = 3,
  type = c("backward", "center"),
  n.ahead = 0,
  x.name = NULL,
  show.plot = TRUE
)

## S3 method for class 'tsmovav'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

## S3 method for class 'tsmovav'
plot(x, title = NULL, ...)
```

Arguments

x	a univariate time series object or a numeric vector or matrix, or a ‘tsmovav‘ object.
order	moving average order. Default is 3.
type	type of moving average to be calculated. Available options are "backward" and "center". While backward assigns the moving averages to the next period after the averaging window, which is more useful for forecasting purpose, center assigns the moving averages to the middle period of the averaging window, which is more suitable for time series smoothing. Default is ‘backward‘.

n.ahead	number of forecasting periods. Only useful if "type = backward". Default is 0.
x.name	a new name for x. If the parameter is omitted, the current name of the time series will be returned to the user.
show.plot	logical. If TRUE, the smoothing/forecasting plot will be displayed directly. Default is TRUE.
digits	the number of significant digits.
...	other printing or plotting parameters.
title	title of the moving average plot. Default is NULL.

Details

Centred moving averages are better suited for smoothing a time series than for forecasting. By definition, each moving average is aligned with the midpoint of its averaging window. When the number of periods in the averaging window (i.e., the moving average order) is even, the averages calculated for the two central positions must be combined. Specifically, the mean of these two middle moving averages is assigned to the central period that lies closest to the true midpoint of the series, forming its final centred moving average.

Mathematically, for odd number order r:

$$\tilde{y}_t = \frac{y_{t-\frac{r-1}{2}} + \ldots + y_{t+\frac{r+1}{2}}}{r}$$

For even number order r:

$$\tilde{y}_t = \frac{0.5y_{t-\frac{r}{2}} + y_{t-\frac{r}{2}+1} + \ldots + y_{t+\frac{r}{2}-1} + 0.5y_{t+\frac{r}{2}}}{2r}$$

Backward moving average is a forecasting method that assigns each computed average to the period immediately following the observation window. This approach works the same regardless of whether the moving average order is odd or even.

$$\tilde{y}_t = \frac{y_{t-1} + \ldots + x_{y-r}}{r}$$

Value

x	original series data
x.time	list of time in which the series values were observed.
x.timegap	time gap between the series and forecasted values.
x.name	name of the time series for which forecasts was requested.
pred	predicted past values and forecasted future values.
pred.time	list of time in which the predictions/forecasts were estimated.
pred.name	name of the series containing the predicted/forecasted values.
se	standard errors of the forecasted values.
cil, ciu	lower and upper limits of the prediction interval.

n.ahead	number of forecasting periods.
forecast.incl	indication of the series part that should be predicted or forecasted.
log	logical. Indicates whether series values are log-transformed for model fitting or not.
alpha	significance level.
order	moving average order.
type	type of moving average.

Author(s)

Ka Yui Karl Wu

References

Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts.
https://otexts.com/fpp3/

Examples

```
## Backward Moving Average
tsmovav(airport$Travellers, order = 12, type = "backward", n.ahead = 6, show.plot = TRUE)

## Centered Moving Average
tsmovav(airport$Travellers, order = 12, type = "center")
```

---

tsqqplot *Quantile-Quantile Plots*

Description

'tsqqplot' is a function to produce a normal QQ plot of the values in y.

Usage

```
tsqqplot(
  x,
  title = NULL,
  x.name = NULL,
  qq.pwidth = 0.7,
  qq.lwidth = 0.7,
  qq.col = "black",
  qqline.col = "red"
)
```

Arguments

x	a univariate time series object or a numeric vector or matrix.
title	title of the QQ plot. Default is NULL.
x.name	name of the series. If omitted here, the series name found by tsname() will be taken over here. If tsname() is NULL, the variable name will be used instead. Default is NULL.
qq.pwidth	size of the markers in the QQ plot. Default is 0.7.
qq.lwidth	line width of the theoretical normal line in the QQ plot. Default is 0.7.
qq.col	colour of the data points in the QQ plot. Default is 'black'.
qqline.col	colour of the theoretical normal line in the QQ plot. Default is 'red'.

Value

A QQ plot of x will be displayed with no further values or objects returned.

Author(s)

Ka Yui Karl Wu

References

Switzer, P. (1976). Confidence procedures for two-sample problems. *Biometrika*, **63**(1), 13-25. doi:10.1093/biomet/63.1.13.

Examples

```
tsqqplot(airport$Travellers)
```

---

tsscatterplot *Scatter Plot*

Description

Produce a scatter plot of two given univariate time series.

Usage

```
tsscatterplot(
  x,
  y,
  reg = FALSE,
  title = NULL,
  x.name = NULL,
  y.name = NULL,
  pwidth = 1,
```

```
  pcol = "steelblue4",
  regwidth = 0.7,
  regcol = "red"
)
```

Arguments

x, y	two univariate time series object or a numeric vector or matrix.
reg	optional. A logical value indicating whether a trend line estimated by regression should be included in the scatter plot. Default is FALSE.
title	title of the histogram. Default is NULL.
x.name	name of the series 'x'. If omitted here, the series name found by tsname(x) will be taken over here. If tsname() is NULL, the variable name will be used instead. Default is NULL.
y.name	name of the series 'y'. If omitted here, the series name found by tsname(y) will be taken over here. If tsname() is NULL, the variable name will be used instead. Default is NULL.
pwidth	size of the markers in the scatter plot. Default is 1.
pcol	colour of the data points in the scatter plot. Default is 'steelblue4'.
regwidth	width of the trend line in the scatter plot. Default is 0.7.
regcol	colour of the trend line in the scatter plot. Default is 'red'.

Value

A scatter plot of x (values on the x-axis) and y (values on the y-axis) will be displayed with no further values or objects returned.

Author(s)

Ka Yui Karl Wu

Examples

```
tsscatterplot(airport$AvgRain, airport$Travellers)
```

Index