Package 'tfarima'

November 14, 2020

Type Package

Title Transfer Function and ARIMA Models

Version 0.1.1

Date 2020-11-14

Description Building customized transfer function and ARIMA models with multiple operators and parameter restrictions. Functions for model identification, model estimation (exact or conditional maximum likelihood), model diagnostic checking, automatic outlier detection, calendar effects, forecasting and seasonal adjustment. See Bell and Hillmer (1983) <doi:10.1080/01621459.1983.10478005>, Box, Jenkins, Reinsel and Ljung <ISBN:978-1-118-67502-1>, Box, Pierce and Newbold (1987) <doi:10.1080/01621459.1987.10478430>, Box and Tiao (1975) <doi:10.1080/01621459.1975.10480264>, Che
son and Tiao (1970) <http://old-www.stat.wisc.edu/sites/default/files/TR222.pdf>.

Author Jose L. Gallego [aut, cre]

Maintainer Jose L. Gallego <jose.gallego@unican.es>

License GPL-2

Imports Rcpp (>= 1.0.0), stats, numDeriv, zoo

LinkingTo Rcpp, RcppArmadillo

Suggests knitr, rmarkdown

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Depends R (>= 2.10)

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-11-14 13:00:02 UTC

R topics documented:

`tfarima-package` *Transfer Function and ARIMA Models.*

Description

The tfarima package provides classes and methods to build customized transfer function and ARIMA models with multiple operators and parameter restrictions. The package also includes functions for model identification, model estimation (exact or conditional maximum likelihood), model diagnostic checking, automatic outlier detection, calendar effects, forecasting and seasonal adjustment.

Author(s)

Jose Luis Gallego <jose.gallego@unican.es>

References

Bell, W.R. and Hillmer, S.C. (1983) Modeling Time Series with Calendar Variation, Journal of the American Statistical Association, Vol. 78, No. 383, pp. 526-534.

Box, G.E., Jenkins, G.M., Reinsel, G.C. and Ljung, G.M. (2015) Time Series Analysis: Forecasting and Control. John Wiley & Sons, Hoboken.

Box, G.E.P., Pierce, D.A. and Newbold, D. A. (1987) Estimating Trend and Growth Rates in Seasonal Time Series, Journal of the American Statistical Association, Vol. 82, No. 397, pp. 276-282.

Box, G.E.P. and Tiao, G.C. (1975) "Intervention Analysis with Applications to Economic and Environmental Problems", Journal of the American Statistical Association, Vol. 70, No. 349, pp. 70-79.

Chen, C. and Liu, L. (1993) Joint Estimation of Model Parameters and Outlier Effects in Time Series, Journal of the American Statistical Association, Vol. 88, No. 421, pp. 284-297

Thompson, H. E. and Tiao, G. C. (1971) "Analysis of Telephone Data: A Case Study of Forecasting Seasonal Time Series," Bell Journal of Economics, The RAND Corporation, vol. 2(2), pages 515-541, Autumn.

as.lagpol *Lag polynomial*

Description

as.lagpol converts a numeric vector c(1, -a_1, ..., -a_d) into a lag polynomial $(1 - a_1 B - ... - a_p B^p)$.

Usage

```
as.lagpol(pol, p = 1)
```

Arguments

pol	a numeric vector.
p	integer power.

Value

An object of class lagpol.

Examples

```
as.lagpol(c(1, -0.8))
as.lagpol(c(1, 0, 0, 0, -0.8))
```

as.um *Convert* arima *into* um.

Description

as.um converts an object of class arima into an object of class um.

Usage

```
as.um(arima)
```

Arguments

arima	an object of class arima.

Value

An object of class um.

Examples

```
z <- AirPassengers
a <- arima(log(z), order = c(0,1,1),
seasonal = list(order = c(0,1,1), frequency = 12))
um1 <- as.um(a)
```

---

autocorr *Theoretical simple/partial autocorrelations of an ARMA model*

Description

autocorr computes the simple/partial autocorrelations of an ARMA model.

Usage

```
autocorr(um, ...)

## S3 method for class 'um'
autocorr(um, lag.max = 10, par = FALSE, ...)
```

Arguments

um	an object of class um.
...	additional arguments.
lag.max	maximum lag for autocovariances.
par	logical. If TRUE partial autocorrelations are computed.

Value

A numeric vector.

Note

The I polynomial is ignored.

Examples

```
ar1 <- um(ar = "1-0.8B")
autocorr(ar1, lag.max = 13)
autocorr(ar1, lag.max = 13, par = TRUE)
```

autocov *Theoretical autocovariances of an ARMA model*

Description

autocov computes the autocovariances of an ARMA model.

Usage

```
autocov(um, ...)

## S3 method for class 'um'
autocov(um, lag.max = 10, ...)
```

Arguments

um	an object of class um.
...	additional arguments.
lag.max	maximum lag for autocovariances.

Value

A numeric vector.

Note

The I polynomial is ignored.

Examples

```
ar1 <- um(ar = "1-0.8B")
autocov(ar1, lag.max = 13)
```

calendar *Calendar effects*

Description

calendar extends the ARIMA model um by estimating a transfer function model with seven deterministic variables to capture the calendar variation in a monthly time series. Two equivalent representations are available: (1) D1, D2, ..., D7, (2) L, D1-D7, ..., D6-D7 where D1, D2, ..., D7 are deterministic variables representing the number of Mondays, Tuesdays, ..., Sundays, L = D1 + D2 + ... + D7 is the of the month. Optionally, a deterministic variable to estimate the Easter effect can also be included.

Usage

```
calendar(um, ...)

## S3 method for class 'um'
calendar(
  um,
  z = NULL,
  form = c("dif", "td"),
  easter = FALSE,
  n.ahead = 0,
  p.value = 1,
  ...
)
```

Arguments

um	an object of class um.
...	additional arguments.
z	a time series.
form	representation for calendar effects: form = td form (1) above, form = dif form (1).
easter	logical. If TRUE an Easter effect is also estimated.
n.ahead	a positive integer to extend the sample period of the deterministic variables with n.ahead observations, which could be necessary to forecast the output.
p.value	estimates with a p-value greater than p.value are omitted.

Value

An object of class "tfm".

References

W. R. Bell & S. C. Hillmer (1983) Modeling Time Series with Calendar Variation, Journal of the American Statistical Association, 78:383, 526-534, DOI: 10.1080/01621459.1983.10478005

Examples

```
Y <- tfarima::rsales
um1 <- um(Y, i = list(1, c(1, 12)), ma = list(1, c(1, 12)), bc = TRUE)
tfm1 <- calendar(um1)
```

CalendarVar *Calendar variables*

Description

CalendarVar creates a set of deterministic variables to capture calendar effects.

Usage

```
CalendarVar(x, form = c("dif", "lom", "td"), easter = FALSE, n.ahead = 0)
```

Arguments

x	an object of class ts used to determine the sample period and frequency.
form	a character indicated the set of calendar variables.
easter	logical. If TRUE an additional deterministic variable is generated to capture Easter effects.
n.ahead	number of additional observations to extend the sample period.

Value

A matrix of explanatory variables.

References

Bell, W.R. and Hillmer, S.C. (1983) "Modeling time series with calendar variation", Journal of the American Statistical Society, Vol. 78, pp. 526–534.

Examples

```
Y <- rsales
X <- CalendarVar(Y, easter = TRUE)
```

ccf.tfm *Cross-correlation check*

Description

ccf displays ccf between prewhitened inputs and residuals.

Usage

```
ccf.tfm(tfm, lag.max = NULL, method = c("exact", "cond"), ...)
```

Arguments

tfm	a tfm object.
lag.max	number of lags.
method	Exact/conditional residuals.
...	additional arguments.

---

diagchk.tfm *Diagnostic checking*

Description

diagchk displays tools for diagnostic checking.

Usage

```
## S3 method for class 'tfm'
diagchk(
  mdl,
  y = NULL,
  method = c("exact", "cond"),
  lag.max = NULL,
  std = TRUE,
  ...
)

diagchk(mdl, ...)

## S3 method for class 'um'
diagchk(
  mdl,
  z = NULL,
  method = c("exact", "cond"),
  lag.max = NULL,
  std = TRUE,
  ...
)
```

Arguments

mdl	an object of class um.
y	an object of class ts.
method	exact or conditional residuals.
lag.max	number of lags for ACF/PACF.
std	logical. If TRUE standardized residuals are shown.
...	additional arguments.
z	optional, an object of class ts.

Examples

```
z <- AirPassengers
airl <- um(z, i = list(1, c(1,12)), ma = list(1, c(1,12)), bc = TRUE)
diagchk(airl)
```

`display` *Graphs for ARMA models*

Description

`display` shows graphs characterizing one or a list of ARMA models.

Usage

```
display(um, ...)

## S3 method for class 'um'
display(
  um,
  lag.max = 25,
  n.freq = 501,
  log.spec = FALSE,
  graphs = c("acf", "pacf", "spec"),
  byrow = FALSE,
  eq = TRUE,
  ...
)

## Default S3 method:
display(um, ...)
```

Arguments

`um`	an object of class `um` or a list of these objects.
`...`	additional arguments.
`lag.max`	number of lags for ACF/PACF.
`n.freq`	number of frequencies for the spectrum.
`log.spec`	logical. If TRUE log spectrum is computed.
`graphs`	vector of graphs.
`byrow`	orientation of the graphs.
`eq`	logical. If TRUE the model equation is used as title.

Examples

```
um1 <- um(ar = "(1 - 0.8B)(1 - 0.8B^12)")
um2 <- um(ma = "(1 - 0.8B)(1 - 0.8B^12)")
display(list(um1, um2))
```

`easter` *Easter effect*

Description

`easter` extends the ARIMA model `um` by including a regression variable to capture the Easter effect.

Usage

```
easter(um, ...)

## S3 method for class 'um'
easter(um, z = NULL, n.ahead = 0, ...)
```

Arguments

`um`	an object of class `um`.
`...`	additional arguments.
`z`	a time series.
`n.ahead`	a positive integer to extend the sample period of the Easter regression variable with `n.ahead` observations, which could be necessary to forecast the output.

Value

An object of class "`tfm`".

Examples

```
Y <- rsales
um1 <- um(Y, i = list(1, c(1, 12)), ma = list(1, c(1, 12)), bc = TRUE)
tfm1 <- easter(um1)
```

`fit.tfm` *Estimation of the ARIMA model*

Description

`fit` fits the univariate model to the time series z.

Usage

```
## S3 method for class 'tfm'
fit(
  mdl,
  y = NULL,
  method = c("exact", "cond"),
  optim.method = "BFGS",
  show.iter = FALSE,
  fit.noise = TRUE,
  ...
)

fit(mdl, ...)

## S3 method for class 'um'
fit(
  mdl,
  z = NULL,
  method = c("exact", "cond"),
  optim.method = "BFGS",
  show.iter = FALSE,
  ...
)
```

Arguments

`mdl`	an object of class `um` or `tfm`.
`y`	a `ts` object.
`method`	Exact/conditional maximum likelihood.
`optim.method`	the `method` argument of the `optim` function.
`show.iter`	logical value to show or hide the estimates at the different iterations.
`fit.noise`	logical. If TRUE parameters of the noise model are fixed.
`...`	additional arguments.
`z`	a time series.

Value

A `tfm` object.

An object of class "um" with the estimated parameters.

Note

The `um` function estimates the corresponding ARIMA model when a time series is provided. The `fit` function is useful to fit a model to several time series, for example, in a Monte Carlo study.

Examples

```
z <- AirPassengers
airl <- um(i = list(1, c(1, 12)), ma = list(1, c(1, 12)), bc = TRUE)
airl <- fit(airl, z)
```

`ide` *Identification plots*

Description

`ide` displays graphs useful to identify a tentative ARIMA model for a time series.

Usage

```
ide(
  Y,
  transf = list(),
  order.polreg = 0,
  lag.max = NULL,
  wn.bands = TRUE,
  graphs = c("plot", "acf", "pacf"),
  set.layout = TRUE,
  byrow = TRUE,
  main = "",
  ...
)
```

Arguments

`Y`	Univariate or multivariate time series.
`transf`	Data transformations, list(bc = F, d = 0, D = 0, S = F), where bc is the Box-Cox logarithmic transformation, d and D are the number of nonseasonal and seasonal differences, and S is the annual sum operator.
`order.polreg`	an integer indicating the order of a polynomial trend.
`lag.max`	number of autocorrelations.

wn.bands	logical. If TRUE confidence intervals for sample autocorrelations are computed assuming a white noise series.
graphs	graphs to be shown: plot, hist, acf, pacf, pgram, cpgram (cummulative periodogram), rm (range-median).
set.layout	logical. If TRUE the layout is set by the function, otherwise it is set by the user.
byrow	logical. If TRUE the layout is filled by rows, otherwise it is filled by columns.
main	title of the graph.
...	additional arguments.

Examples

```
Y <- AirPassengers
ide(Y, graphs = c("plot", "rm"))
ide(Y, transf = list(list(bc = TRUE, S = TRUE), list(bc = TRUE, d = 1, D = 1)))
```

InterventionVar *Intervention variables*

Description

InterventionVar creates an intervention variable to capture the effect of an external event.

Usage

```
InterventionVar(Y, date, type = c("P", "S", "R"), n.ahead = 0)
```

Arguments

Y	an object of class ts used to determine the sample period and frequency.
date	the date of the event, c(year, month).
type	a character indicating the type of intervention variables: (P) pulse, (S) step, (R).
n.ahead	number of additional observations to extend the sample period.

Value

An intervention variable, a 'ts' object.

References

G. E. P. Box, G. C. Tiao, "Intervention Analysis with Applications to Economic and Environmental Problems", Journal of the American Statistical Association, Vol. 70, No. 349. (Mar., 1975), pp. 70-79.

Examples

```
Y <- seriesJ$Y
P58 <- InterventionVar(Y, date = 58, type = "P")
```

`inv` *Inverse of a lag polynomial*

Description

`inv` inverts a lag polynomial until the indicated lag.

Usage

```
inv(lp, ...)

## S3 method for class 'lagpol'
inv(lp, lag.max = 10, ...)
```

Arguments

`lp`	an object of class `lagpol`.
`...`	additional arguments.
`lag.max`	largest order of the inverse lag polynomial.

Value

`inv` returns a numeric vector with the coefficients of the inverse lag polynomial truncated at lag.max.

Examples

```
inv(as.lagpol(c(1, 1.2, -0.8)))
```

`lagpol` *Lag polynomials*

Description

`lagpol` creates a lag polynomial of the form $(1 - coef_1 B^s - ... - coef_d B^s d)^p$. This class of lag polynomials is defined by a vector of d coefficients c(coef_1, ..., coef_d), the powers s and p, and a vector of k parameters c(param_1, ..., param_k). The vector c(coef_1, ..., coef_d) is actually a vector of math expressions to compute the value of each coefficient in terms of the parameters.

Usage

```
lagpol(param = NULL, s = 1, p = 1, lags = NULL, coef = NULL)
```

Arguments

`param`	a vector/list of named parameters.
`s`	the seasonal period, integer.
`p`	the power of lag polynomial, integer.
`lags`	a vector of lags for sparse polynomials.
`coef`	a vector of math expressions.

Value

`lagpol` returns an object of class "lagpol" with the following components:

coef Vector of coefficients c(coef_1, ..., coef_p) provided to create the lag polynomial.

pol Base lag polynomial, c(1, -coef_1, ..., -coef_d).

Pol Power lag polynomial when p > 1.

Examples

```
lagpol(param = c(phi = 0.8) )
lagpol(param = c(phi1 = 1.2, phi2 = -0.6), s = 4)
lagpol(param = c(delta = 1), p = 2)
```

logLik.um *Log-likelihood of an ARIMA model*

Description

logLik computes the exact or conditional log-likelihood of object of the class um.

Usage

```
## S3 method for class 'um'
logLik(object, z = NULL, method = c("exact", "cond"), ...)
```

Arguments

object	an object of class um.
z	an object of class ts.
method	exact or conditional.
...	additional arguments.

Value

The exact or conditional log-likelihood.

modify.tfm *Modifying a TF or an ARIMA model*

Description

modify modifies an object of class um or tfm by adding and/or removing lag polynomials.

Usage

```
## S3 method for class 'tfm'
modify(mdl, ...)

modify(mdl, ...)

## S3 method for class 'um'
modify(
  mdl,
  ar = NULL,
  i = NULL,
  ma = NULL,
  mu = NULL,
  sig2 = NULL,
```

```
  bc = NULL,
  fit = TRUE,
  ...
)
```

Arguments

mdl	an object of class um or tfm.
...	additional arguments.
ar	list of stationary AR lag polynomials.
i	list of nonstationary AR (I) polynomials.
ma	list of MA polynomials.
mu	mean of the stationary time series.
sig2	variance of the error.
bc	logical. If TRUE logs are taken.
fit	logical. If TRUE, model is fitted.

Value

An object of class um or um.

Examples

```
um1 <- um(ar = "(1 - 0.8B)")
um2 <- modify(um1, ar = list(0, "(1 - 0.9B)"), ma = "(1 - 0.5B)")
```

---

nabla *Unscramble I polynomial*

Description

nabla multiplies the I polynomials of an object of the um class.

Usage

```
nabla(um)

## S3 method for class 'um'
nabla(um)
```

Arguments

um	an object of class um.

Value

A numeric vector `c(1,a1,...,ad)`

Note

This function returns the member variable `um$nabla`.

Examples

```
um1 <- um(i = "(1 - B)(1 - B^12)")
nabla(um1)
```

`noise` *Noise of a transfer function model*

Description

`noise` computes the noise of a linear transfer function model.

Usage

```
noise(tfm, ...)

## S3 method for class 'tfm'
noise(tfm, y = NULL, diff = TRUE, exp = FALSE, ...)
```

Arguments

`tfm`	an object of the class `tfm`.
`...`	additional arguments.
`y`	output of the TF model if it is different to that of the `tfm` object.
`diff`	logical. If TRUE, the noise is differenced with the "i" operator of the univariate model of the noise.
`exp`	logical. If TRUE, the antilog transformation is applied.

Value

A "ts" object.

outlierDates *Outlier dates*

Description

`outlierDates` shows the indeces and dates of outliers.

Usage

```
outlierDates(x, c = 3)
```

Arguments

`x`	an `ts` object.
`c`	critical value to determine whether or not an observation is an outlier.

Value

A table with the indices, dates and z-scores of the outliers.

outliers.tfm *Outliers detection at known/unknown dates*

Description

`outliers` performs a detection of four types of anomalies (AO, TC, LS and IO) in a time series described by an ARIMA model. If the dates of the outliers are unknown, an iterative detection process like that proposed by Chen and Liu (1993) is conducted.

Usage

```
## S3 method for class 'tfm'
outliers(
  mdl,
  y = NULL,
  dates = NULL,
  c = 3,
  calendar = FALSE,
  easter = FALSE,
  n.ahead = NULL,
  p.value = 1,
  ...
)

outliers(mdl, ...)
```

```
## S3 method for class 'um'
outliers(
  mdl,
  z = NULL,
  dates = NULL,
  c = 3,
  calendar = FALSE,
  easter = FALSE,
  n.ahead = 0,
  p.value = 1,
  ...
)
```

Arguments

mdl	an object of class um or tfm.
y	an object of class ts
dates	a list of dates c(year, season). If dates = NULL, an iterative detection process is conducted.
c	a positive constant to compare the z-ratio of the effect of an observation and decide whether or not it is an outlier. This argument is only used when dates = NULL.
calendar	logical; if true, calendar effects are also estimated.
easter	logical; if true, Easter effect is also estimated.
n.ahead	a positive integer to extend the sample period of the intervation variables with n.ahead observations, which could be necessary to forecast the output.
p.value	estimates with a p-value greater than p.value are omitted.
...	additional arguments.
z	a time series.

Value

an object of class "tfm" or a table.

Examples

```
Y <- rsales
um1 <- um(Y, i = list(1, c(1, 12)), ma = list(1, c(1, 12)), bc = TRUE)
outliers(um1)
```

output.tf *Output of a transfer function*

Description

output filters the input using the transfer function.

Usage

```
output.tf(tf)
```

Arguments

tf	an object of the S3 class "tf".

Value

A "ts" object

pccf *Prewhitened cross correlation function*

Description

pccf displays cross correlation function between input and output after prewhitening both through a univariate model.

Usage

```
pccf(
  x,
  y,
  um.x = NULL,
  um.y = NULL,
  lag.max = NULL,
  plot = TRUE,
  main = NULL,
  nu.weights = FALSE,
  ...
)
```

Arguments

x	input, a 'ts' object or a numeric vector.
y	output, a 'ts' object or a numeric vector.
um.x	univariate model for input.
um.y	univariate model for output.
lag.max	number of lags, integer.
plot	logical value to indicate if the ccf graph must be graphed or computed.
main	title of the graph.
nu.weights	logical. If TRUE the coefficients of the IRF are computed instead of the cross-correlations.
...	additional arguments.

Value

The estimated cross correlations are displayed in a graph or returned into a numeric vector.

---

phi *Unscramble AR polynomial*

Description

phi multiplies the AR polynomials of an object of the um class.

Usage

```
phi(um)

## S3 method for class 'um'
phi(um)
```

Arguments

um	an object of class um.

Value

A numeric vector c(1,a1,...,ad)

Note

This function returns the member variable um$phi.

Examples

```
um1 <- um(ar = "(1 - 0.8B)(1 - 0.5B)")
phi(um1)
```

pi.weights *Pi weights of an AR(I)MA model*

Description

pi.weights computes the pi-weights of an AR(I)MA model.

Usage

```
pi.weights(um, ...)

## S3 method for class 'um'
pi.weights(um, lag.max = 10, var.pi = FALSE, ...)
```

Arguments

um	an object of class um.
...	additional arguments.
lag.max	largest AR(Inf) coefficient required.
var.pi	logical. If TRUE (FALSE), the I polynomials is considered (ignored).

Value

A numeric vector.

Examples

```
um1 <- um(i = "(1 - B)(1 - B^12)", ma = "(1 - 0.8B)(1 - 0.8B^12)")
pi.weights(um1, var.pi = TRUE)
```

predict.tfm *Forecasting with transfer function models*

Description

predict computes point and interval predictions for a time series based on a tfm object.

Usage

```
## S3 method for class 'tfm'
predict(object, y = NULL, ori = NULL, n.ahead = NULL, level = 0.95, ...)
```

Arguments

object	an object of class um.
y	an object of class ts.
ori	the origin of prediction. By default, it is the last observation.
n.ahead	number of steps ahead.
level	confidence level.
...	additional arguments.

---

predict.um *Forecasts from an ARIMA model*

Description

predict computes point and interval predictions for a time series from models of class um.

Usage

```
## S3 method for class 'um'
predict(object, z = NULL, ori = NULL, n.ahead = 1, level = 0.95, ...)
```

Arguments

object	an object of class um.
z	an object of class ts.
ori	the origin of prediction. By default, it is the last observation.
n.ahead	number of steps ahead.
level	confidence level.
...	additional arguments.

Value

An object of class "tfm".

Examples

```
Z <- AirPassengers
um1 <- um(Z, i = list(1, c(1, 12)), ma = list(1, c(1, 12)), bc = TRUE)
p <- predict(um1, n.ahead = 12)
p
plot(p, n.back = 60)
```

printLagpol *Print numeric vector as a lagpol object*

Description

Print numeric vector as a lagpol object

Usage

```
printLagpol(pol, digits = 2)
```

Arguments

pol	numeric vectors with the coefficients of a normalized polynomial.
digits	number of decimals.

printLagpolList *Print a list of lagpol objects*

Description

Print a list of lagpol objects

Usage

```
printLagpolList(llp, digits = 2)
```

Arguments

llp	a list of lagpol objects.
digits	number of decimals.

psi.weights *Psi weights of an AR(I)MA model*

Description

psi computes the psi-weights of an AR(I)MA model.

Usage

```
psi.weights(um, ...)

## S3 method for class 'um'
psi.weights(um, lag.max = 10, var.psi = FALSE, ...)
```

Arguments

um	an object of class um.
...	additional arguments.
lag.max	Largest MA(Inf) coefficient required.
var.psi	logical. If TRUE the I polynomials is also inverted. If FALSE it is ignored.

Value

A numeric vector.

Examples

```
um1 <- um(i = "(1 - B)(1 - B^12)", ma = "(1 - 0.8B)(1 - 0.8B^12)")
psi.weights(um1)
psi.weights(um1, var.psi = TRUE)
```

residuals.tfm *Residuals of a transfer function model*

Description

residuals computes the exact or conditional residuals of a TF model.

Usage

```
## S3 method for class 'tfm'
residuals(object, y = NULL, method = c("exact", "cond"), ...)
```

Arguments

object	a tfm object.
y	output of the TF model (if it is different to that of the "tfm" object).
method	a character string specifying the method to compute the residuals, exact or conditional.
...	additional arguments.

Value

A "ts" object.

---

residuals.um *Residuals of the ARIMA model*

Description

residuals computes the exact or conditional residuals.

Usage

```
## S3 method for class 'um'
residuals(object, z = NULL, method = c("exact", "cond"), ...)
```

Arguments

object	an object of class um.
z	an object of class ts.
method	exact/conditional residuals.
...	additional arguments.

Value

An object of class um.

Examples

```
z <- AirPassengers
airl <- um(z, i = list(1, c(1, 12)), ma = list(1, c(1, 12)), bc = TRUE)
r <- residuals(airl)
summary(r)
```

roots *Roots of the lag polynomials of an ARIMA model*

Description

roots compute the roots of the AR, I, MA lag polynomials an ARIMA model.

Usage

```
roots(x, ...)

## S3 method for class 'um'
roots(x, opr = c("arma", "ar", "ma", "i", "arima"), ...)
```

Arguments

x	an object of class um.
...	additional arguments.
opr	character that indicates which operators are selected.

Value

List of matrices with the roots of each single polynomial.

Examples

```
um1 <- um(ar = "(1 - 0.8B)(1 - 0.8B^12)")
roots(um1)
```

roots.lagpol *Roots of a lag polynomial*

Description

roots.lagpol computes the roots of a lag polynomial.

Usage

```
## S3 method for class 'lagpol'
roots(x, table = TRUE, ...)

## Default S3 method:
roots(x, ...)
```

Arguments

x	an object of class lagpol.
table	logical. If TRUE, it returns a five columns table showing the real and imaginary parts, the modulus, the frequency and the period of each root.
...	additional arguments.

Value

A vector or a table.

Examples

```
roots(c(1, 1.2, -0.8))
```

---

rsales *Retail Sales of Variety Stores (U.S. Bureau of the Census)*

Description

156 monthly observations from January 1967 to December 1979.

Usage

```
rsales
```

Format

An object of class ts of length 156.

Source

https://www.census.gov/retail/mrts/mrtshist.html

References

Chen, C. and Liu, L. (1993) Joint Estimation of Model Parameters and Outlier Effects in Time Series, Journal of the American Statistical Association, Vol. 88, No. 421, pp. 284-297

seasadj *Seasonal adjustment*

Description

seasadj removes the seasonal component of time series.

Usage

```
seasadj(mdl, ...)

## S3 method for class 'um'
seasadj(mdl, z = NULL, method = c("mixed", "forecast", "backcast"), ...)
```

Arguments

mdl	an object of class um or tfm.
...	additional arguments.
z	an object of class ts.
method	forward/backward forecasts or a mixture of the two.

Value

seasadj returns a seasonal adjusted time series.

Examples

```
Y <- AirPassengers
um1 <- um(Y, bc = TRUE, i = list(1, c(1,12)), ma = list(1, c(1,12)))
Y <- seasadj(um1)
ide(Y)
```

seriesC *Series C Chemical Process Temperature Readings: Every Minute.*

Description

226 observations.

Usage

```
seriesC
```

Format

An object of class numeric of length 226.

Source

ftp://ftp.wiley.com/public/sci_tech_med/times_series_example

References

Box, G.E., Jenkins, G.M., Reinsel, G.C. and Ljung, G.M. (2015) Time Series Analysis: Forecasting and Control. John Wiley & Sons, Hoboken.

---

seriesJ *Gas furnace data*

Description

Sampling interval 9 seconds; observations for 296 pairs of data points.

Usage

```
seriesJ
```

Format

A object of class data.frame with 296 rows and 2 columns:

X 0.60-0.04 (input gas rate in cubir feet per minute.)

Y % CO2 in outlet gas.

Source

ftp://ftp.wiley.com/public/sci_tech_med/times_series_example

References

Box, G.E., Jenkins, G.M., Reinsel, G.C. and Ljung, G.M. (2015) Time Series Analysis: Forecasting and Control. John Wiley & Sons, Hoboken.

setinputs — setinputs *adds new inputs into a transfer function model.*

Description

setinputs adds new inputs into a transfer function model.

Usage

```
setinputs(tfm, ...)

## S3 method for class 'tfm'
setinputs(tfm, xreg = NULL, inputs = NULL, ...)
```

Arguments

tfm	a tfm object.
...	further arguments to be passed to particular methods
xreg	a matrix of inputs.
inputs	a list of tf objects.

Value

A tfm object.

signal — *Signal component of a TF model*

Description

signal extracts the signal of a TF model.

Usage

```
signal(mdl, ...)

## S3 method for class 'tfm'
signal(mdl, y = NULL, diff = TRUE, ...)
```

Arguments

mdl	an object of the class tfm.
...	additional arguments.
y	output of the TF model if it is different to that of the tfm object.
diff	logical. If TRUE, the noise is differenced with the "i" operator of the univariate model of the noise.

Value

A "ts" object.

`sim.tfm` *Time series simulation form an ARIMA or TF model*

Description

`sim` generates a random time series from an object of class `um` or `tfm`.

Usage

```
## S3 method for class 'tfm'
sim(mdl, n = 100, y0 = NULL, seed = NULL, ...)

sim(mdl, ...)

## S3 method for class 'um'
sim(mdl, n = 100, z0 = NULL, seed = NULL, ...)
```

Arguments

`mdl`	an object of class `um` or `tfm`.
`n`	number of observations.
`y0`	initial conditions for the nonstationary series.
`seed`	an integer.
`...`	additional arguments.
`z0`	initial conditions for the nonstationary series.

Value

An object of class `ts`.

spec *Spectrum of an ARMA model*

Description

spec computes the spectrum of an ARMA model.

Usage

```
spec(um, ...)

## S3 method for class 'um'
spec(um, n.freq = 501, ...)
```

Arguments

um	an object of class um.
...	additional parameters.
n.freq	number of frequencies.

Value

A matrix with the frequencies and the power spectral densities.

Note

The I polynomial is ignored.

Examples

```
um1 <- um(i = "(1 - B)(1 - B^12)", ma = "(1 - 0.8B)(1 - 0.8B^12)")
s <- spec(um1, lag.max = 13)
```

std *Standardize time series*

Description

std standardizes a time series.

Usage

```
std(x)
```

Arguments

x	a ts object.

Value

The standardized time series.

---

summary.tfm *Summarizing Transfer Function models*

Description

summary method for class "tfm".

Usage

```
## S3 method for class 'tfm'
summary(
  object,
  y = NULL,
  method = c("exact", "cond"),
  digits = max(3L, getOption("digits") - 3L),
  ...
)
```

Arguments

object	a tfm object.
y	a "ts" object.
method	exact or conditional maximum likelihood.
digits	number of significant digits to use when printing.
...	additional arguments.

Value

A tfm object.

summary.um *Summary of um model*

Description

summary prints a summary of the estimation and diagnosis.

Usage

```
## S3 method for class 'um'
summary(
  object,
  z = NULL,
  method = c("exact", "cond"),
  digits = max(3L, getOption("digits") - 3L),
  ...
)
```

Arguments

object	an object of class um.
z	an object of class ts.
method	exact/conditional maximum likelihood.
digits	number of significant digits to use when printing.
...	additional arguments.

Value

A list with the summary of the estimation and diagonosis.

Examples

```
z <- AirPassengers
airl <- um(z, i = list(1, c(1,12)), ma = list(1, c(1,12)), bc = TRUE)
summary(airl)
```

sum_um *Sum of univariate (ARIMA) models*

Description

sum_um creates a univariate (ARIMA) model from the sum of serveral univariate (arima) models.

Usage

```
sum_um(...)
```

Arguments

...	List of "um" S3 objects.

Value

A "um" S3 object.

Examples

```
um1 <- um(i = "(1 - B)", ma = "(1 - 0.8B)")
um2 <- um(i = "(1 - B12)", ma = "(1 - 0.8B^12)")
um3 <- sum_um(um1, um2)
```

tf *Transfer function for input*

Description

tf creates a rational transfer function for an input, V(B) = w0(1 - w_1B - ... - w_qB^q)/(1-d_1B - ... - d_pB^p)B^dX_t. Note that in this specification the constant term of the MA polynomial is factored out so that both polynomials in the numerator and denominator are normalized and can be specified with the lagpol function in the same way as the operators of univariate models.

Usage

```
tf(
  x = NULL,
  delay = 0,
  w0 = 0,
  ar = NULL,
  ma = NULL,
  um = NULL,
  n.back = NULL,
  par.prefix = ""
)
```

Arguments

x	input, a ts object or a numeric vector.
delay	integer.
w0	constant term of the polynomial V(B), double.
ar	list of stationary AR polynomials.
ma	list of MA polynomials.
um	univariate model for stochastic input.
n.back	number of backcasts to extend the input.
par.prefix	prefix name for parameters.

Value

An object of the class "tf".

References

Box, G.E., Jenkins, G.M., Reinsel, G.C. and Ljung, G.M. (2015) Time Series Analysis: Forecasting and Control. John Wiley & Sons, Hoboken.

Wei, W.W.S. (2006) Time Series Analysis Univariate and Multivariate Methods. 2nd Edition, Addison Wesley, New York, 33-59.

See Also

um.

Examples

```
x <- rep(0, 100)
x[50] <- 1
tfx <- tf(x, w0 = 0.8, ar = "(1 - 0.5B)(1 - 0.7B^12)")
```

---

tfest *Preestimates of a transfer function*

Description

tfest provides preestimates of the transfer function between an output and an input.

Usage

```
tfest(y, x, delay = 0, p = 1, q = 2, um.y = NULL, um.x = NULL, n.back = NULL)
```

Arguments

y	output, a ts object or a numeric vector.
x	input, a ts object or a numeric vector.
delay	integer.
p	order of the AR polynomial, double.
q	order of the MA polynomial, double.
um.y	univariate model for output, um object or NULL.
um.x	univariate model for input, um object or NUL.
n.back	number of backcasts.

Value

A "tf" S3 object

---

tfm *Transfer function models*

Description

tfm creates a multiple input transfer function model.

Usage

```
tfm(output = NULL, xreg = NULL, inputs = NULL, noise, fit = TRUE, ...)
```

Arguments

output	a ts object or a numeric vector.
xreg	a matrix of regressors.
inputs	a list of tf objects.
noise	a um object for the noise.
fit	logical. If TRUE, model is fitted.
...	additional arguments.

Value

An object of the class tfm.

References

Box, G.E., Jenkins, G.M., Reinsel, G.C. and Ljung, G.M. (2015) Time Series Analysis: Forecasting and Control. John Wiley & Sons, Hoboken.

See Also

tf and um.

theta *Unscramble MA polynomial*

Description

Unscramble MA polynomial

Usage

```
theta(um)

## S3 method for class 'um'
theta(um)
```

Arguments

um	an object of class um.

Value

A numeric vector c(1,a1,...,ad)

Note

This function returns the member variable um$theta.

Examples

```
um1 <- um(ma = "(1 - 0.8B)(1 - 0.5B)")
theta(um1)
```

tsdiag.tfm *Diagnostic Plots for Time-Series Fits Description*

Description

tsdiag.tfm is a wrap of the stats::tsdiag function.

Usage

```
## S3 method for class 'tfm'
tsdiag(object, gof.lag = 10, ...)
```

Arguments

object	a fitted um object.
gof.lag	the maximum number of lags for a Portmanteau goodness-of-fit test
...	additional arguments.

See Also

stats::tsdiag.

---

tsdiag.um *Diagnostic Plots for Time-Series Fits Description*

Description

tsdiag.um is a wrap of the stats::tsdiag function.

Usage

```
## S3 method for class 'um'
tsdiag(object, gof.lag = 10, ...)
```

Arguments

object	a fitted um object.
gof.lag	the maximum number of lags for a Portmanteau goodness-of-fit test
...	additional arguments.

See Also

stats::tsdiag.

---

ucomp.tfm *Unobserved components*

Description

ucomp estimates the unobserved components of a time series (trend, seasonal, cycle, stationary and irregular) from the eventual forecast function.

Usage

```
## S3 method for class 'tfm'
ucomp(mdl, y = NULL, method = c("mixed", "forecast", "backcast"), ...)

ucomp(mdl, ...)

## S3 method for class 'um'
ucomp(mdl, z = NULL, method = c("mixed", "forecast", "backcast"), ...)
```

Arguments

mdl	an object of class um or tfm.
y	an object of class ts.
method	forward/backward forecasts or a mixture of the two.
...	additional arguments.
z	an object of class ts.

Value

A matrix with the unobserved components.

Examples

```
Z <- AirPassengers
um1 <- um(Z, i = list(1, c(1, 12)), ma = list(1, c(1, 12)), bc = TRUE)
uc <- ucomp(um1)
```

---

um *Univariate (ARIMA) model*

Description

um creates an S3 object representing a univariate ARIMA model, which can contain multiple AR, I and MA polynomials, as well as parameter restrictions.

Usage

```
um(
  z = NULL,
  ar = NULL,
  i = NULL,
  ma = NULL,
  mu = NULL,
  sig2 = 1,
  bc = FALSE,
  fit = TRUE,
  ...
)
```

Arguments

z	an object of class ts.
ar	list of stationary AR lag polynomials.
i	list of nonstationary AR (I) polynomials.
ma	list of MA polynomials.
mu	mean of the stationary time series.
sig2	variance of the error.
bc	logical. If TRUE logs are taken.
fit	logical. If TRUE, model is fitted.
...	additional arguments.

Value

An object of class um.

References

Box, G.E.P., Jenkins, G.M., Reinsel, G.C. and Ljung, G.M. (2015) Time Series Analysis: Forecasting and Control. John Wiley & Sons, Hoboken.

Examples

```
ar1 <- um(ar = "(1 - 0.8B)")
ar2 <- um(ar = "(1 - 1.4B + 0.8B^2)")
ma1 <- um(ma = "(1 - 0.8B)")
ma2 <- um(ma = "(1 - 1.4B + 0.8B^2)")
arma11 <- um(ar = "(1 - 1.4B + 0.8B^2)", ma = "(1 - 0.8B)")
```

---

varsel *Variable selection*

Description

varsel omits non-significant inputs from a transfer function model.

Usage

```
varsel(tfm, ...)

## S3 method for class 'tfm'
varsel(tfm, p.value = 0.1, ...)
```

Arguments

tfm	a tfm object.
...	additional arguments.
p.value	probability value to decide whether or not to omit an input.

Value

A tfm object or a "um" if no input is significant at that level.

Wtelephone *Wisconsin Telephone Company*

Description

Monthly data from January 1951 to October 1966.

Usage

```
Wtelephone
```

Format

A object of class data.frame with 215 rows and 2 columns:

X Monthly outward station movements.

Y Montly inward station movements.

Source

http://old-www.stat.wisc.edu/sites/default/files/TR222.pdf

References

Thompson, H. E. and Tiao, G. C. (1971) "Analysis of Telephone Data: A Case Study of Forecasting Seasonal Time Series," Bell Journal of Economics, The RAND Corporation, vol. 2(2), pages 515-541, Autumn.

Index