Package ‘forecTheta’

November 11, 2022

Type Package

Title Forecasting Time Series by Theta Models

Version 2.6.2

Author Jose Augusto Fiorucci [aut, cre, cph]
(https://orcid.org/0000-0002-1201-9089),
Francisco Louzada [aut, cph] (https://orcid.org/0000-0001-7815-9554)

Maintainer Jose Augusto Fiorucci <jafiorucci@gmail.com>

Description Routines for forecasting univariate time series using Theta Models.

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 2.0), forecast, tseries

BugReports https://github.com/jafiorucci/forecTheta/issues


NeedsCompilation no

Repository CRAN

Date/Publication 2022-11-11 22:30:02 UTC

R topics documented:

Cross Validation .................................................. 2
Error Metric ...................................................... 3
expSmoot .......................................................... 5
forecTheta-Package ............................................ 6
otm.arxiv ......................................................... 8
Plot .............................................................. 10
Theta Models ..................................................... 11

Index 15
Cross Validation

Generalised Rolling Origin Evaluation

Description

Usage
```r
groe(y, forecFunction, g="sAPE", n1=length(y)-10, m=5,
    H=length(y)-n1, p=1+floor((length(y)-n1)/m), ...)  
rolOrig(y, forecFunction, g="sAPE", n1=length(y)-10, ...)  
fixOrig(y, forecFunction, g="sAPE", n1=length(y)-10, ...)  
```

Arguments
- `y`: Object of time series class or a vector
- `forecFunction`: A forecasting method as one object of the forecast class of forecast package.
- `g`: The prediction error type of errorMetric function. The possible values are "sAPE", "APE", "AE" and "SE".
- `n1`: The index of the first origin element.
- `m`: The number of movements of the origin in each update.
- `H`: The number of predictions forward of each origin.
- `p`: The number of origin updates. Default is the maximum.
- `...`: Additional arguments for `forecFunction`.

Details
If `m=1` is computed the Rolling Origin Evaluation. If `m>=length(y)-n1` is computed the Fixed Origin Evaluation.

Value
The sum of the prediction errors.

Note
The `otm.arxiv` function use this function for estimate the theta parameter when the theta argument is NULL. Your computer may go into an infinite looping if you use `forecFunction = otm.arxiv` without specific a numeric value for the theta argument.
Author(s)

Jose Augusto Fiorucci and Francisco Louzada

References


See Also

forecTheta-package, dotm, otm.arxiv

Examples

```r
y1 = 2 + 0.15*(1:20) + rnorm(20,2)
y2 = y1[20]+ 0.3*(1:30) + rnorm(30,2)
y = as.ts(c(y1,y2))

## Rolling Origin Evaluation
rolOrig( y=y, forecFunction = dotm, n1=40)
rolOrig( y=y, forecFunction = expSmoot, n1=40)
rolOrig( y=y, forecFunction = stheta, n1=40)
rolOrig( y=y, forecFunction = otm.arxiv, n1=40, theta=3)

## Fixed Origin Evaluation
fixOrig( y=y, forecFunction = dotm, n1=40)
fixOrig( y=y, forecFunction = expSmoot, n1=40)
fixOrig( y=y, forecFunction = stheta, n1=40)
fixOrig( y=y, forecFunction = otm.arxiv, n1=40, theta=3)

## Generalised Rolling Origin Evaluation with two origin updates.
## Where the first is the 40th element and second is the 45th element
groe( y=y, forecFunction = dotm, m=5, n1=40)
groe( y=y, forecFunction = expSmoot, m=5, n1=40)
groe( y=y, forecFunction = stheta, m=5, n1=40)
groe( y=y, forecFunction = otm.arxiv, m=5, n1=40, theta=3)
```

<table>
<thead>
<tr>
<th>Error Metric</th>
<th>Error Metric Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Description

This function implements some of the more used error metrics. These metrics are "sMAPE", "MAPE", "MAE", "MSE" and they respectively versions with median "sMdAPE", "MdAPE", "MdAE", "MdSE".
Usage

   errorMetric(obs, forec, type="sAPE", statistic="M")

Arguments

   obs        A vector or a matrix with the real values.
   forec      A vector or a matrix with the estimated values.
   type       The error type of "sAPE", "APE", "AE" and "SE".
   statistic  The statistic to be returned. Use "M" or "Md" for return the mean or median of
               the errors. If "N" so a vector with all errors will be returned.

Details

   The metric sMAPE is obtained using type = "sAPE" and statistic = "M"
   The metric sMdAPE is obtained using type = "sAPE" and statistic = "Md"
   The metric MAPE is obtained using type = "APE" and statistic = "M"
   The metric MdAPE is obtained using type = "APE" and statistic = "Md"
   The metric MAE is obtained using type = "AE" and statistic = "M"
   The metric MdAE is obtained using type = "AE" and statistic = "Md"
   The metric MSE is obtained using type = "SE" and statistic = "M"
   The metric MdSE is obtained using type = "SE" and statistic = "Md"

Value

   If statistic="M" or statistic="Md" it is returned the respectively error metric result. If statistic="N"
   so is returned a vector with all errors points according to the chosen error type.

Author(s)

   Jose Augusto Fiorucci and Francisco Louzada

See Also

   forecTheta-package, groe

Examples

   ###################################################################
   y1 = 2+ 0.15*(1:20) + rnorm(20,2)
   y2 = y1[20]+ 0.3*(1:30) + rnorm(30,2)
   y = as.ts(c(y1,y2))

   out <- dotm(y=as.ts(y[1:40]), h=10)

   ### sMAPE metric
   errorMetric(obs=as.ts(y[1:40]), forec=out$mean)
expSmoot

### sMdAPE metric
errorMetric(obs=as.ts(y[41:50]), forec=out$mean, statistic = "Md")

### MASE metric
meanDiff1 = mean(abs(diff(as.ts(y[1:40]), lag = 1)))
errorMetric(obs=as.ts(y[41:50]), forec=out$mean, type = "AE", statistic = "M") / meanDiff1

**expSmoot**

**Simple Exponential Smoothing Method**

**Description**

Estimation of Simple Exponential Smoothing Method

**Usage**

expSmoot(y, h=5, ell0=NULL, alpha=NULL, lower = c(-1e+10, 0.1), upper = c(1e+10, 0.99))

**Arguments**

- **y** Object of time series class.
- **h** Number of required forecasting periods.
- **ell0** The value of ell0^* parameter.
- **alpha** The value of alpha parameter.
- **lower** The lower limit of parametric space.
- **upper** The upper limit of parametric space.

**Value**

A list containing the elements:

- **$y** The original time series.
- **$par** The estimated values for (ell^*, alpha) parameters
- **$mean** The forecasting values
- **$fitted** A time series element with the fitted points.
- **$residuals** A time series element with the residual points.

**Author(s)**

Jose Augusto Fiorucci, Francisco Louzada and Bao Yiqi

**See Also**

forecTheta-package, sttheta, dotm
Examples

\begin{verbatim}
y1 = 2 + 0.15*(1:20) + rnorm(20,2)
y2 = y1[20] + 0.3*(1:30) + rnorm(30,2)
y = as.ts(c(y1,y2))
expSmoot(y, h=10)
\end{verbatim}

---

**Description**

In this package we implement functions for forecast univariate time series using the several Theta Models (Fiorucci et al, 2015 and 2016) and the Standard Theta Method of Assimakopoulos and Nikolopoulos (2000).

**Details**

- Package: forecTheta
- Type: Package
- Version: 2.6.2
- Date: 2022-11-11
- License: GPL (>=2.0)

\begin{verbatim}
dotm(y, h)
stheta(y, h)
errorMetric(obs, forec, type = "sAPE", statistic = "M")
groe(y, forecFunction = ses, g = "sAPE", n1 = length(y)-10)
\end{verbatim}

**Author(s)**

Jose Augusto Fiorucci, Francisco Louzada

Maintainer: Jose Augusto Fiorucci <jafiorucci@gmail.com>

**References**


**See Also**

dotm, stheta, otm.arxiv, groe, rolOrig, fixOrig, errorMetric

**Examples**

```r
### sMAPE metric
errorMetric(obs=as.ts(y[41:50]), forec=out$mean, type = "sAPE", statistic = "M")
errorMetric(obs=as.ts(y[41:50]), forec=out$mean, type = "sAPE", statistic = "M")

### sMdAPE metric
errorMetric(obs=as.ts(y[41:50]), forec=out$mean, type = "sAPE", statistic = "Md")
errorMetric(obs=as.ts(y[41:50]), forec=out$mean, type = "sAPE", statistic = "Md")

### MASE metric
meanDiff1 = mean(abs(diff(as.ts(y[1:40]), lag = 1)))
errorMetric(obs=as.ts(y[41:50]), forec=out$mean, type = "AE", statistic = "M") / meanDiff1
errorMetric(obs=as.ts(y[41:50]), forec=out$mean, type = "AE", statistic = "M") / meanDiff1

#### cross validation (2 origins)
#groe( y=y, forecFunction = otm.arxiv, m=5, n1=40, p=2, theta=5)
#groe( y=y, forecFunction = stheta, m=5, n1=40, p=2)

#### cross validation (rolling origin evaluation)
#rolOrig( y=y, forecFunction = otm.arxiv, n1=40, theta=5)
#rolOrig( y=y, forecFunction = stheta, n1=40)
```
Optimised Theta Method

Description

Functions for forecast univariate time series using the Optimised Theta Method presented in the arxiv paper (Fioruci et al, 2015). If the theta parameter is not specified so the Generalised Rolling Origin Evaluation is used for select the theta value over the thetaList argument.

Usage

```r
otm.arxiv( y, h=5, s=NULL, theta=NULL, tLineExtrap=expSmoot, g="sAPE", approach="c", n1=NULL, m=NULL, H=NULL, p=NULL, thetaList=seq(from=1,to=5,by=0.5), mc.cores=1, ...)
```

Arguments

- `y`: Object of time series class
- `h`: Number of required forecasting periods
- `s`: If TRUE, the multiplicative seasonal decomposition is used. If NULL, quarterly and monthly time series are tested for statistically seasonal behaviour, with 95% of significance. Default is NULL.
- `theta`: The value of theta parameter. If theta = NULL the theta parameter is estimated using the Generalised Rolling Origin Evaluation.
- `tLineExtrap`: A forecasting function for extrapolation the second theta-line. Default is expSmoot.
- `g`: The error type that will be used by groe function for select the theta value in the estimation process. The possibility values for g is "sAPE", "APE", "AE" and "SE". If theta is not NULL the g argument is not used. Default is "sAPE".
- `approach`: The approach set-up for groe parameters (n1, m, H, p). One letter between 'a' to 'h' according to Fioruci et al (2015).
- `n1`: The first origin for Generalised Rolling Origin Evaluation. This argument is not used if theta!=NULL or approach!=NULL.
- `m`: The number of movements of the origin in each step. This argument is not used if theta!=NULL or approach!=NULL.
- `H`: The number of predictions in each step. This argument is not used if theta!=NULL or approach!=NULL.
- `p`: The number of origin updates. This argument is not used if theta!=NULL or approach!=NULL.
- `thetaList`: A vector with the possible values for theta. This argument is not used if theta argument is not NULL.
- `mc.cores`: Number of cores that will be used for estimate the theta parameter. It is not accepted mc.cores>1 on Windows SO.
- `...`: Additional arguments for tLineExtrap.
Details

These functions are fully automatic, you just need to pass your time series. Particular cases are obtained by: If theta = 1 the `tLineExtrapModel` method is computed; If theta = 2 so the Standard Theta Method of Assimakopoulos and Nikolopoulos (2000) is computed.

By default (s=NULL), the 90% significance seasonal Z-test, used by Assimakopoulos and Nikolopoulos (2000), is applied for quarterly and monthly time series.

Value

An list containing the elements:

- `$y` The original time series.
- `$mean` A time series element with the forecasting points.
- `$fitted` A time series element with the fitted points.
- `$residuals` A time series element with the residual points.
- `$theta` The estimated theta value.
- `$tLineExtrap_par` The estimated parameters of `tLineExtrap` method.
- `$weights` The estimated weights values.

Note

The `thetaM` function is just a particular case of `otm` with theta=2.

Author(s)

Jose Augusto Fiorucci, Francisco Louzada

References


See Also

`forecTheta-package`, `dotm`, `groe`

Examples

```r
y1 = 2+ 0.15*(1:20) + rnorm(20,2)
y2 = y1[20]+ 0.3*(1:30) + rnorm(30,2)
y = as.ts(c(y1,y2))

otm.arxiv(y, h=10)
```
### running the M3-competition data base by OTM approach (a) ###

```r
#require(Mcomp)
data(M3)

#forec = matrix(NA, nrow=3003, ncol=18) #matrix of the out-sample values
#obs = matrix(NA, nrow=3003, ncol=18)

#for(i in 1:3003){
# if(i %% 100 == 0){print(i)}
# x=M3[[i]]$x
# h=M3[[i]]$h
# out = otm.arxiv(x,h,approach='a',tLineExtrap=ses)
# forec[i,1:h] = out$mean
# obs[i,1:h] = M3[[i]]$xx
#}

#sAPE = errorMetric(obs, forec, type="sAPE", statistic="N") # sAPE matrix

###### sMAPE results ####
### Yearly
#mean( sAPE[1:645, 1:6] )
### QUARTERLY
#mean( sAPE[646:1401, 1:8] )
### MONTHLY
#mean( sAPE[1402:2829, 1:18] )
### Other
#mean( sAPE[2830:3003, 1:8] )
### ALL
#mean( sAPE, na.rm=TRUE )
```

---

**Plot**

*Plot forecasts points and prediction intervals for thetaModel objects*

**Description**

Produces a figure of the time series and the forecasts points from Optimised Theta Method.

**Usage**

```
# S3 method for class 'thetaModel'
plot(x, ylim=NULL, xlim=NULL, ylab=NULL, xlab=NULL, main=NULL, ...)  
```

**Arguments**

- `x`: Object of class "thetaModel".
- `ylim`: the y limits of the plot.
- `xlim`: the x limits of the plot.
- `ylab`: a label for the y axis.
**Theta Models**

- **xlab** a label for the x axis.
- **main** a main title for the plot.
- ... Other plotting parameters passed to `par`.

**Value**

None. Function produces a plot

**Author(s)**

Jose A Fiorucci

**See Also**

`dotm`, `forecTheta-package`

**Examples**

```r
y1 = 2 + 0.15*(1:20) + rnorm(20,2)
y2 = y1[20] + 0.3*(1:30) + rnorm(30,2)
y = as.ts(c(y1,y2))
out <- dotm(y, h=10)
plot(out)
```

---

**Description**

Functions for forecast univariate time series using the Dynamic Optimised Theta Model, Dynamic Standard Theta Model, Optimised Theta Model and Standard Theta Model (Fiorucci et al, 2016). We also provide an implementation for the Standard Theta Method (STheta) of Assimakopoulos and Nikolopoulos (2000).

**Usage**

```r
dotm(y, h=5, level=c(80,90,95), s=NULL, par_ini=c(y[1]/2, 0.5, 2), estimation=TRUE, lower=c(-1e+10, 0.1, 1.0), upper=c(1e+10, 0.99, 1e+10), opt.method="Nelder-Mead", xreg=NULL)
dstm(y, h=5, level=c(80,90,95), s=NULL, par_ini=c(y[1]/2, 0.5), estimation=TRUE, lower=c(-1e+10, 0.1), upper=c(1e+10, 0.99), opt.method="Nelder-Mead", xreg=NULL)
ottom(y, h=5, level=c(80,90,95), s=NULL, par_ini=c(y[1]/2, 0.5, 2), estimation=TRUE, lower=c(-1e+10, 0.1, 1.0), upper=c(1e+10, 0.99, 1e+10), opt.method="Nelder-Mead", xreg=NULL)
stm(y, h=5, level=c(80,90,95), s=NULL, par_ini=c(y[1]/2, 0.5), estimation=TRUE,
```

---
lower=c(-1e+10, 0.1), upper=c(1e+10, 0.99), opt.method="Nelder-Mead", xreg=NULL)

stheta(y, h=5, s=NULL)

Arguments

y Object of time series class.

h Number of required forecasting periods.

level Levels for prediction intervals.

s If TRUE, the multiplicative seasonal decomposition is used. If NULL and frequency(y)>=4 the time series is tested for statistically seasonal behaviour, with 90% of significance. If s='additive' or close zero values been find in the multiplicative decomposition, the additive decomposition is performed hatter than multiplicative. Default is NULL.

par_ini Vector of initialization for (ell, alpha, theta) parameters.

estimation If TRUE, the optim() function is consider for compute the minimum square estimator of parameters. If FALSE, the models/methods are computed for par_ini values.

lower The lower limit of parametric space.

upper The upper limit of parametric space.

opt.method The numeric optimisation method for optim() function. Choose one among 'Nelder-Mead', 'L-BFGS-B', 'SANN'.

xreg A matrix with the regressor variables including the out-of-sample data.

Details

By default (s=NULL), the 90% significance seasonal Z-test, used by Assimakopoulos and Nikolopoulos (2000), is applied for quarterly and monthly time series.

For details of each model see Fiorucci et al, 2016. If you are looking for the methods presented in the arXiv paper (Fiorucci et al, 2015), see otm.arxiv() function.

Value

An object of thetaModel class with one list containing the elements:

$method The name of the model/method

$y The original time series.

$s A binary indication for seasonal decomposition.

type Classical seasonal decomposition type.

opt.method The optimisation method used in the optim() function.

$par The estimated values for (ell, alpha, theta) parameters

$weights The estimated weights values.

$fitted A time series element with the fitted points.

$residuals A time series element with the residual points.
Theta Models

$\text{mean}$  The forecasting values.
$\text{level}$  The levels for prediction intervals.
$\text{lower}$  Lower limits for prediction intervals.
$\text{upper}$  Upper limits for prediction intervals.
$\text{tests}$  The p.value of Terasvirta Neural Network test applied on unseasoned time series and the p.value of Shapiro-Wilk test applied on unseasoned residuals.

Author(s)

Jose Augusto Fiorucci, Francisco Louzada

References


See Also

forecTheta-package, otm.arxiv

Examples

```r
y1 = 2+ 0.15*(1:20) + rnorm(20)
y2 = y1[20]+ 0.3*(1:30) + rnorm(30)
y = as.ts(c(y1,y2))
out <- dotm(y, h=10)
summary(out)
plot(out)

#### additive seasonal decomposition ####
x = sin(2*pi*seq(0,9,len=300)) + exp((1:300)/150) + rnorm(mean=0,sd=0.5,n=300)
y = ts(x, frequency=33)
out <- dotm(y, h=50, s='additive')
summary(out)
plot(out)
```

# # Reproducing the M3 results by DOTM #
# # library(Mcomp)
# data(M3)
# # forec = matrix(NA, nrow=3003, ncol=18)
# # obs = matrix(NA, nrow=3003, ncol=18) #matrix of the out-sample values
Theta Models

```r
# meanDiff <- rep(1, 3003)
#
# for(i in 1:3003){
#  x=M3[[i]]$x
#  h=M3[[i]]$h
#  out = dotm(x,h,level=NULL)
#  forec[i,1:h] = out$mean
#  obs[i,1:h] = M3[[i]]$xx
#  meanDiff[i] = mean(abs(diff(x, lag = frequency(x))))
# }
#
# ############## sMAPE ###################
# sAPE_matrix = errorMetric(obs=obs, forec=forec, type="sAPE", statistic="N")
# #### Yearly ###
# mean( sAPE_matrix[1:645, 1:6] )
# #### QUARTERLY ###
# mean( sAPE_matrix[646:1401, 1:8] )
# #### MONTHLY ###
# mean( sAPE_matrix[1402:2829, 1:18] )
# #### Other ###
# mean( sAPE_matrix[2830:3003, 1:8] )
# #### ALL ###
# mean( sAPE_matrix, na.rm=TRUE )
#
# ############# MASE ####################
# AE_matrix = errorMetric(obs=obs, forec=forec, type="AE", statistic="N")
# ASE_matrix=AE_matrix/meanDiff
# #### Yearly ###
# #### QUARTERLY ###
# mean( ASE_matrix[646:1401, 1:8] )
# #### MONTHLY ###
# mean( ASE_matrix[1402:2829, 1:18] )
# #### Other ###
# mean( ASE_matrix[2830:3003, 1:8] )
# #### ALL ###
# mean( ASE_matrix, na.rm=TRUE )
```
Index

* Cross Validation
  Cross Validation, 2
* DOTM
  Theta Models, 11
* DSTM
  Theta Models, 11
* Dynamic Optimised Theta Model
  forecTheta-Package, 6
* Fixed Origin Evaluation
  Cross Validation, 2
* Generalised Rolling Origin Evaluation
  Cross Validation, 2
* MAE
  Error Metric, 3
* MSE
  Error Metric, 3
* MdAE
  Error Metric, 3
* MdSE
  Error Metric, 3
* OTM
  Theta Models, 11
* Rolling Origin Evaluation
  Cross Validation, 2
* STM
  Theta Models, 11
* STheta
  Theta Models, 11
* Theta Method
  forecTheta-Package, 6
* error metric
  Error Metric, 3
* otm
  otm.arxiv, 8
* plot
  Plot, 10
* sMAPE
  Error Metric, 3
* sMdAPE
  Error Metric, 3

Error Metric, 3

* simple exponential smoothing
  expSmoot, 5
* theta-method
  otm.arxiv, 8
* thetaM
  otm.arxiv, 8
* time series forecasting
  forecTheta-Package, 6
  otm.arxiv, 8
  Theta Models, 11

Cross Validation, 2
dotm, 3, 5, 7, 9, 11
dotm(Theta Models), 11
dstm(Theta Models), 11

Error Metric, 3
errorMetric, 7
errorMetric (Error Metric), 3
expSmoot, 5

fixOrig, 7
fixOrig (Cross Validation), 2
forecTheta (forecTheta-Package), 6
forecTheta-Package, 6
forecTheta-package (forecTheta-Package), 6

groe, 4, 7, 9
groe (Cross Validation), 2

otm (Theta Models), 11
otm.arxiv, 3, 7, 8, 13

par, 11
Plot, 10
plot.thetaModel (Plot), 10
rolOrig, 7
rolOrig (Cross Validation), 2
stheta.5, 7
stheta (Theta Models), 11
stm (Theta Models), 11

Theta Models, 11