Package 'thief'

May 8, 2026

Version 0.3

Title Temporal Hierarchical Forecasting

Description Methods and tools for generating forecasts at different temporal frequencies using a hierarchical time series approach.

Depends R (>= 3.0.2), forecast (>= 7.2)

Imports hts, ggplot2

LazyData yes

ByteCompile TRUE

BugReports https://github.com/robjhyndman/thief/issues

License GPL-3

URL http://pkg.robjhyndman.com/thief, https://github.com/robjhyndman/thief

RoxygenNote 6.0.1.9000

NeedsCompilation no

Author Rob Hyndman [aut, cre, cph], Nikolaos Kourentzes [aut, cph]

Maintainer Rob Hyndman <Rob.Hyndman@monash.edu>

Repository CRAN

Date/Publication 2018-01-24 05:06:33 UTC

Contents

AEdemand 2
plot.tsaggregates 3
reconcilethief 4
thief 5
tsaggregates 7

Index **8**

AEdemand *Accident and Emergency demand in the UK*

Description

Weekly demand of Accident & Emergency departments in the UK, from 7 November 2010 to 7 June 2015.

Usage

```
AEdemand
```

Format

An object of class `ts`.

Examples

```
library(ggplot2)
autoplot(AEdemand, xlab="Year", ylab="Demand ('000)") +
  ggtitle("Accident & Emergency Demand in the UK")

## Not run:

# Demonstration of the adjustment of all temporal aggregates
# using Total Emergency Admissions

total <- AEdemand[,12]
totalagg <- tsaggregates(total)
plot(totalagg, main="Total Emergency Admissions")

# Base forecasts
base <- list()
for(i in 1:5)
  base[[i]] <- forecast(auto.arima(totalagg[[i]]))
base[[6]] <-  forecast(auto.arima(totalagg[[6]]), h=2)

# Reconciled forecasts
reconciled <- reconcilethief(base)

main <- paste(names(totalagg)," (k=",
           52/unlist(lapply(reconciled,frequency)),")",sep="")
par(mfrow=c(2,3))
for(i in 6:1)
{
  ylim <- range(base[[i]]$mean, base[[i]]$x, reconciled[[i]]$mean)
  plot(base[[i]], main=main[i], fcol='white',
      plot.conf=FALSE, ylim=ylim, xlim=c(2010.5,2017.5))
  polygon(c(2015.45, 2020, 2020, 2015.45),
          c(0, 0, 1e5, 1e5), col='grey', border=FALSE)
```

```
  lines(base[[i]]$mean, col='red', lty=2)
  lines(reconciled[[i]]$mean, col='blue')
}

## End(Not run)
```

`plot.tsaggregates` *Plot time series aggregates*

Description

Plots all temporal aggregations of a time series

Usage

```
## S3 method for class 'tsaggregates'
plot(x, series = "all", ...)

## S3 method for class 'tsaggregates'
autoplot(object, series = "all", ...)
```

Arguments

`x`	`tsaggregates` object, produced by `tsaggregates`.
`series`	The indexes of the series to plot. By default, all series are plotted.
`...`	Other arguments passed to `plot.ts` or `autoplot.ts`.
`object`	`tsaggregates` object, produced by `tsaggregates`.

Author(s)

Rob J Hyndman

Examples

```
deathagg <- tsaggregates(USAccDeaths)
plot(deathagg, series=c(1,2,4,6))

library(ggplot2)
autoplot(deathagg)
```

`reconcilethief` *Reconcile temporal hierarchical forecasts*

Description

Takes forecasts of time series at all levels of temporal aggregation and combines them using the temporal hierarchical approach of Athanasopoulos et al (2016).

Usage

```
reconcilethief(forecasts, comb = c("struc", "mse", "ols", "bu", "shr", "sam"),
  mse = NULL, residuals = NULL, returnall = TRUE, aggregatelist = NULL)
```

Arguments

`forecasts`	List of forecasts. Each element must be a time series of forecasts, or a forecast object. The number of forecasts should be equal to k times the seasonal period for each series, where k is the same across all series.
`comb`	Combination method of temporal hierarchies, taking one of the following values: **"struc"** Structural scaling - weights from temporal hierarchy **"mse"** Variance scaling - weights from in-sample MSE **"ols"** Unscaled OLS combination weights **"bu"** Bottom-up combination – i.e., all aggregate forecasts are ignored. **"shr"** GLS using a shrinkage (to block diagonal) estimate of residuals **"sam"** GLS using sample covariance matrix of residuals
`mse`	A vector of one-step MSE values corresponding to each of the forecast series.
`residuals`	List of residuals corresponding to each of the forecast models. Each element must be a time series of residuals. If `forecast` contains a list of forecast objects, then the residuals will be extracted automatically and this argument is not needed. However, it will be used if not `NULL`.
`returnall`	If `TRUE`, a list of time series corresponding to the first argument is returned, but now reconciled. Otherwise, only the most disaggregated series is returned.
`aggregatelist`	(optional) User-selected list of forecast aggregates to consider

Value

List of reconciled forecasts in the same format as `forecast`. If `returnall==FALSE`, only the most disaggregated series is returned.

Author(s)

Rob J Hyndman

See Also

thief, tsaggregates

Examples

```
# Construct aggregates
aggts <- tsaggregates(USAccDeaths)

# Compute forecasts
fc <- list()
for(i in seq_along(aggts))
  fc[[i]] <- forecast(auto.arima(aggts[[i]]), h=2*frequency(aggts[[i]]))

# Reconcile forecasts
reconciled <- reconcilethief(fc)

# Plot forecasts before and after reconcilation
par(mfrow=c(2,3))
for(i in seq_along(fc))
{
  plot(reconciled[[i]], main=names(aggts)[i])
  lines(fc[[i]]$mean, col='red')
}
```

---

thief *Temporal hierarchical forecasting*

Description

Takes a time series as input and produces forecasts using the temporal hierarchical approach of Athanasopoulos et al (2016).

Usage

```
thief(y, m = frequency(y), h = m * 2, comb = c("struc", "mse", "ols",
  "bu", "shr", "sam"), usemodel = c("ets", "arima", "theta", "naive",
  "snaive"), forecastfunction = NULL, aggregatelist = NULL, ...)
```

Arguments

y	Time series input
m	Seasonal period
h	Forecast horizon
comb	Combination method of temporal hierarchies, taking one of the following values:
	"struc" Structural scaling - weights from temporal hierarchy

"mse" Variance scaling - weights from in-sample MSE

"ols" Unscaled OLS combination weights

"bu" Bottom-up combination – i.e., all aggregate forecasts are ignored.

"shr" GLS using a shrinkage (to block diagonal) estimate of residuals

"sam" GLS using sample covariance matrix of residuals

`usemodel` Model used for forecasting each aggregation level:

"ets" exponential smoothing, using the `ets` function.

"arima" arima, using the `auto.arima` function.

"theta" theta method, using the `thetaf` function.

"naive" random walk forecasts

"snaive" seasonal naive forecasts, based on the last year of observed data.

`forecastfunction` User-defined function to be used instead of `usemodel`. The function must take a time series as the first argument, and the forecast horizon as the second argument. It must return an object of class `forecast`.

`aggregatelist` User-selected list of forecast aggregates to consider

`...` Arguments to be passed to the time series modelling function (such as `ets` or `auto.arima`), or to `forecastfunction`.

Details

This function computes the temporal aggregates of `y` using `tsaggregates`, then calculates all forecasts using the model function specified by `usemodel` or `forecastfunction`, and finally reconciles the forecasts using `reconcilethief`. The reconciled forecasts of `y` are returned.

Value

forecast object.

Author(s)

Rob J Hyndman and Nikolaos Kourentzes

See Also

`reconcilethief`

Examples

```
## Not run:

# Select ARIMA models for all series using auto.arima()
z <- thief(AEdemand[,12], usemodel='arima')
plot(z)

# Use your own function
ftbats <- function(y,h,...){forecast(tbats(y),h,...)}
z <- thief(AEdemand[,12], forecastfunction=ftbats)
```

```
plot(z)

## End(Not run)
```

`tsaggregates` *Non-overlapping temporal aggregation of a time series*

Description

Produces all temporal aggregations for frequencies greater than 1

Usage

```
tsaggregates(y, m = frequency(y), align = c("end", "start"),
  aggregatelist = NULL)
```

Arguments

`y`	Univariate time series of class `ts`.
`m`	Integer seasonal period
`align`	Indicates how the aggregates are to be aligned: either with the `start` of the series or the `end` of the series. For forecasting purposes, it should be set to `end`.
`aggregatelist`	User-selected list of aggregates to consider.

Value

A list of time series. The first element is the series 'y', followed by series with increasing levels of aggregation. The last element is the "annual" series (i.e., the series aggregated over all seasons).

Author(s)

Rob J Hyndman

See Also

`plot.tsaggregates`

Examples

```
tsaggregates(USAccDeaths)
```

Index

∗ **datasets**
AEdemand, 2

AEdemand, 2
auto.arima, *6*
autoplot.ts, *3*
autoplot.tsaggregates
*(*plot.tsaggregates*)*, 3

ets, *6*

plot.ts, *3*
plot.tsaggregates, 3, *7*

reconcilethief, 4, *6*

thetaf, *6*
thief, *5*, 5
tsaggregates, *3*, *5*, *6*, 7