Nikolaos Kourentzes

Forecasting research

TStools for R

By Nikos | April 19, 2014 0 Comment

This is a collection of functions for time series analysis/modelling for R. Follow link to GitHub. If you need help installing this package in R have a look at this post. Alternatively just type in R the following commands:

```
> if (!require("devtools"))
    install.packages("devtools")
> devtools::install_github("trnnick/TStools")
```

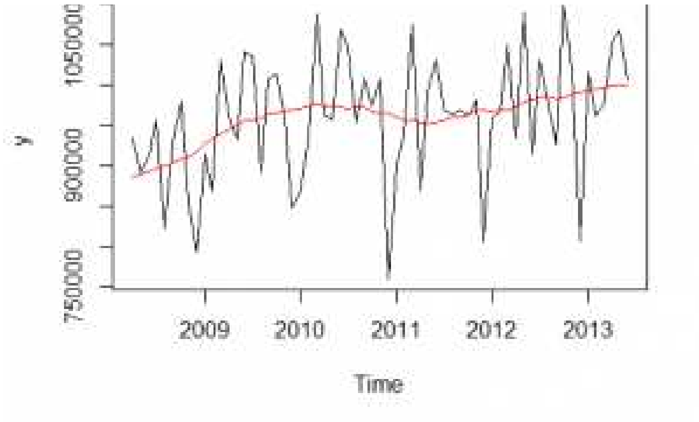
At the time of posting the following functions are included:

- cmav: Centred moving average;
- coxstuart: Cox-Stuart test;
- decomp: Time series decomposition;
- kdemode: Mode estimation via KDE;
- nemenyi: Friedman and Nemenyi tests;
- regopt: Identify regression beta using various cost functions;
- seasplot: Seasonal plots.

Here are some examples of what these functions do.

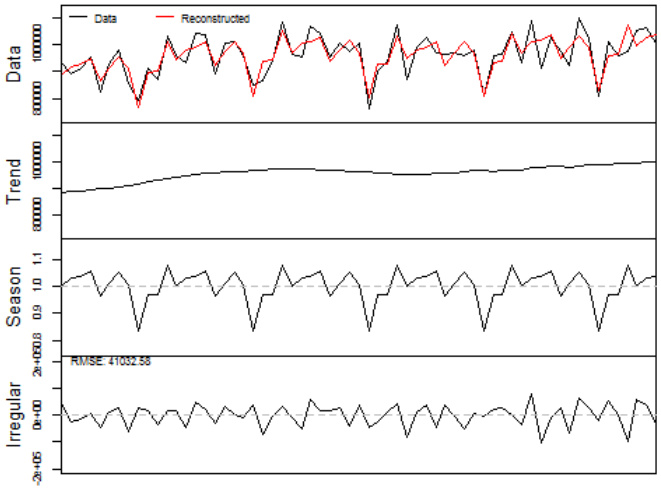
The cmav function will calculate the centred moving average of a time series. It differs from the ma function in the forecast package in the sense that it can provide values for the first and last segment of the time series, where the centred moving average cannot be calculated, by fitting an ETS model.

```
> cmav(referrals,outplot=TRUE)
```



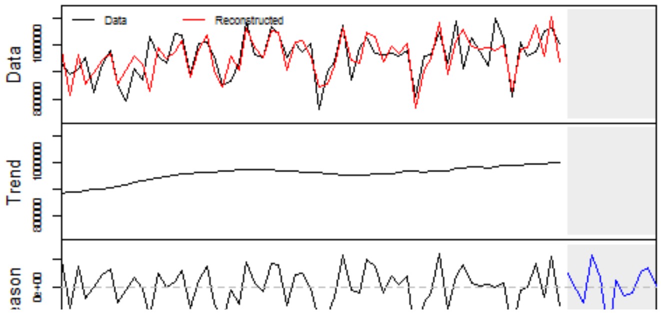
The decomp function performs classical decomposition on a time series.

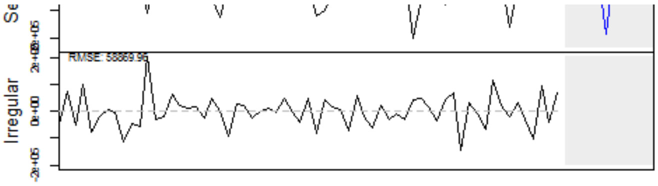
```
> decomp(referrals,outplot=TRUE)
```



You can estimate the season using either the mean or the median of the historical indices or fit a pure seasonal model. Here, I also choose the type of decomposition and ask the seasonal indices for the next 12 months.

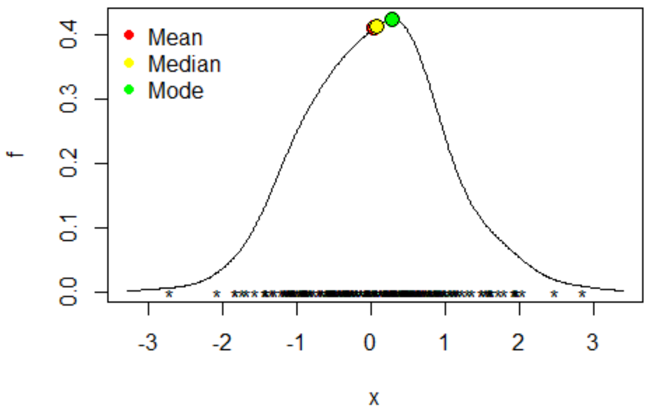
```
> decomp(referrals,outplot=TRUE,type="pure.seasonal",decomposition="additive",h=12)
```





The kdemode can be used to find the mode of a distribution by kernel density estimation. The bandwidth is automatically identified.

```
> data <- rlnorm(200,mean=0,sd=1)
kdemode(data,outplot=TRUE)
```

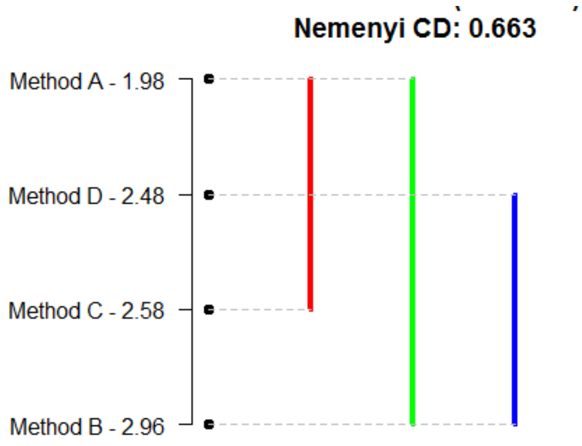


The bandwidth is estimated automatically using the excellent estimation proposed by Botev et al. Alternatively the bandwidth can be calculated using either the Sheater and Jones method or the Silverman heuristic. This was quite useful for producing the mode ensemble neural networks that are proposed here.

The nemenyi function can be used to run the Nemenyi test to compare models. Here is an example:

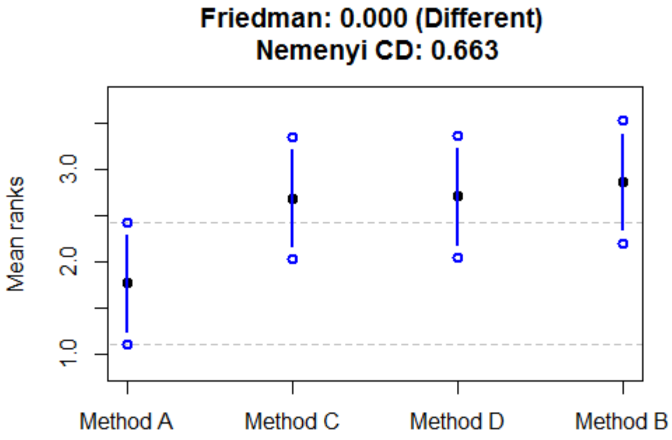
```
> N <- 50
> M <- 4
> data <- matrix( rnorm(N*M,mean=0,sd=1), N, M)
> data[,2] <- data[,2]+1
> data[,3] <- data[,3]+0.7
> data[,4] <- data[,4]+0.5
> colnames(data) <- c("Method A","Method B","Method C","Method D");
> nemenyi(data,conf.int=0.95,plottype="vline")
```

Friedman: 0.002 (Different)



I can also get the MCB test by using:

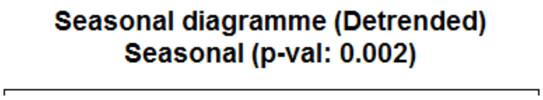
```
> nemenyi(data,conf.int=0.95,plottype="mcb")
```

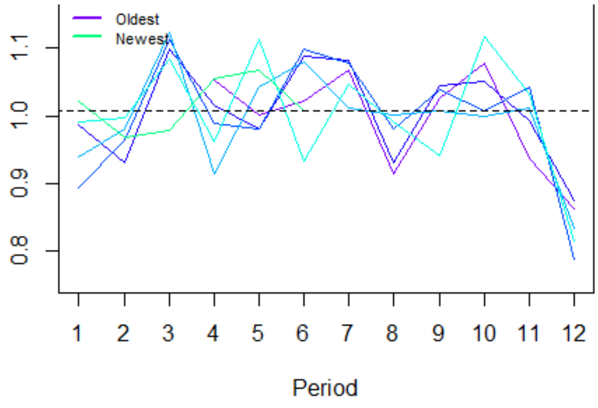


These tests use the same statistic. A discussion on this and various visualisations can be found here.

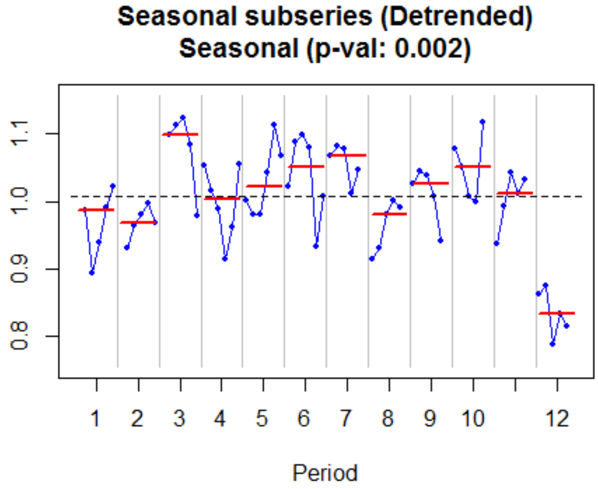
Finally, the seasplot function produces seasonal plots. It differs from the similar seasonplot in the forecast package as it can automatically test for presence of trend and remove it, if present.

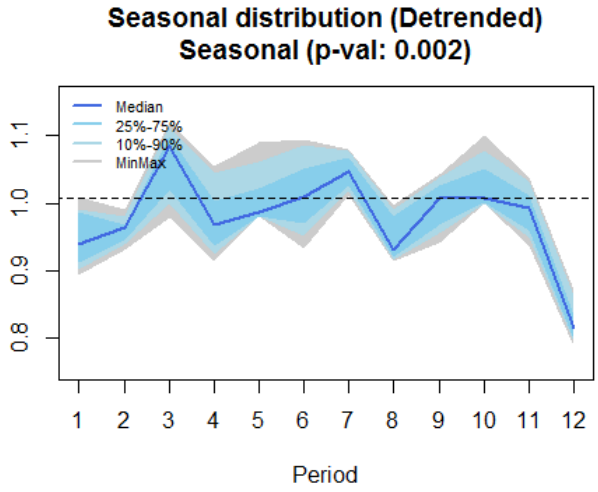
```
> seasplot(referrals)
```





Furthermore, there are a few alternative visualisations you can do with seasplot. Here are some examples:





You may wonder how is that p-value calculated. Based on the definition of deterministic seasonality, I just compare the distributions of the seasonal elements and test whether at least one is different. I do this by using the nonparametric Friedman test.

The plan is to keep on updating TStools on GitHub. When there are substantial updates, I will post them here.

For examples of some of the new features of TStools have a look here.

Related Posts

- RShiny demo for basic time series exploration
- Another update for tsintermittent
- Forecasting multiple time series with tsintermittent

Category: R code Tags: code, R, statistical tests, time series exploration