Tutorial 12: Nonlinear Models and Neural Networks

1. The unemployment figures mentioned in the discussion of TAR (threshold auto-regression) are found in the file `m-unrate.txt` in the data directory. Plot the time series, observe the phenomenon mentioned (that the series increases rapidly and decreases slowly). Fit a SETAR model. The package **tsDyn** does this:

```
> install.packages("tsDyn")
> library("tsDyn")
>
www<-"https://www.mimuw.edu.pl/~noble/courses/TimeSeries/data/m-unrate
.txt"
> data<-read.table(www,header=T)
> res<-setar(data$Rate,d=1,mL=12,mH=12,model="TAR",nthresh=1)
> summary(res)
```

There are other packages are available:

- The **BAYSTAR** package deals with SETAR (self-exciting threshold autoregressive model which is a special case of TAR model with

$$Z_t = X_{t-d}$$

where d is an integer (X is the time series, Z is the threshold process). This deals with models with Gaussian white noise process with two regimes. The principal features of this package are:

  - Estimate the threshold value and the lag d using the Metropolis-Hastings algorithm.
  - The user must provide the autoregressive orders.
  - It provides the simulation of a series from a SETAR model.

- **TSA** package. It contains R functions and datasets. As with BAYSTAR, this package only considers the case of two regimes. The principal features of this package are:

  - The user must provide the value of d.
  - It estimates the threshold value and using the minimum AIC criterion.
  - It selects the AR orders by minimizing AIC.
  - It can simulate a series from a SETAR model.
  - It provides a likelihood test for threshold nonlinearity.
  - It provides prediction based on a fitted SETAR model.

The TAR package is more versatile (it can estimate d and it can estimate the number of regimes) my experience was that it simply took far too long; estimating two regimes worked fine - more than that needed a faster computer than mine. BAYSTAR works as follows:

```
>install.packages("BAYSTAR")
> library(BAYSTAR)
>BAYSTAR(data$Rate,c(1,2,3,4,12),c(1,2,3,4,12),step.thv=1,Iteration=10000,
Burnin=1000)
```

2. Try fitting a STAR model to the unemployment rate data. Information for **tsDyn** is found here:

   `ftp://cran.r-project.org/pub/R/web/packages/tsDyn/tsDyn.pdf`

   Information on the command `star` is found on page 64.

3. The package **nnet** builds neural networks:

   ```
   > install.packages("nnet")
   > library("nnet")
   ```

   The data for the log returns of IBM stock is found in `m-ibmln2699.txt` (January 1926 - December 1999).

   ```
   >
   www3<-"https://www.mimuw.edu.pl/~noble/courses/TimeSeries/data/m-
   ibmln2699.txt"
   > ibm<-read.table(www3,header=T)
   ```

   Construct a 3-2-1 neural network for the series (3 input nodes, representing $X(t-3), X(t-2), X(t-1)$, one hidden node and one output node representing $X(t)$). This may be accomplished using `nnet`. Use the first 864 observations to construct the model, then use the remaining observations for testing the predictor. The command `predict.nnet` is useful.

   Compare prediction accuracy with that of fitting an AR model using the first 864 observations, and considering the one-step prediction errors for the remainder of the series.

   Note `nnet` constructs a *single* hidden layer model, which is what we want here.

4. Consider the monthly simple returns of GE stock from January 1926 to December 2008. They are found in `m-ge2608.txt` in the course directory. Use the last three years of data for forecast evaluation.

   (a) Using lagged returns $r(t-1)$, $r(t-2)$, $r(t-3)$ as input, build a 3-2-1 feed forward neural network to forecast 1-step-ahead returns. Calculate the mean squared error of forecasts.

   (b) Again, using $r(t-1)$, $r(t-2)$, $r(t-3)$ and also their *signs*, build a 6-5-1 feed forward neural network to forecast the 1-step-ahead GE stock price *movement*, with 1 denoting upward movement. Calculate the mean squared error of the forecasts.

   If `rtn` denotes return, you can create a direction variable by:

   ```
   drtn = ifelse(rtn>0,1,0)
   ```