ECO374: Forecasting and Time Series Econometrics
Department of Economics, University of Toronto
Problem Set 1

Winter 2024

This Problem Set is based on the software files *Case Study 1* and file *NNAR Application*.

1. Obtain data on "iShares S&P/TSX 60 Index ETF" (series XIU.TO) from the yahoo finance database, available at this link. It is one of the largest ETFs in Canada listed at the Toronto Stock Exchange, with exposure to large, established Canadian companies. Adjust the start of the data series to 2023-05-01 and the end to 2024-02-01. Print the first five rows and the last five rows of the data to verify the time span.
2. Plot the data series, with the title "TSX 60 Index ETF" and x-axis date breaks of 2 months.
3. Difference the data and plot with the title "Differenced TSX 60 Index ETF", date breaks of 2 months.
4. Plot the autocorrelation function (ACF) and the partial autocorrelation function (PACF) of the differenced data series. Based on these, decide on a suitable ARMA model (no seasonal component) with less than 5 lags (there are more than one correct possibilities).
5. Obtain the best-fitting model as suggested by R with the function auto.arima for the differenced data.
6. Obtain the best-fitting Autoregressive Neural Network model for the level (i.e. not differenced) data by time series cross-validation, based on software file *NNAR Application*. Move the location of the legend on the graph to make it nicely visible.
7. Perform time-series validation with differenced data (except NNAR with level data) for the following models, based on software file *Case Study 1*:
    a. The model suggested by auto.arima
    b. ARMA model of your choice from point 4 above
    c. SETAR model with a threshold of 0
    d. LSTAR model with th=0 and all other starting parameters set to 1
    e. NNAR model from point 6
8. Write a brief comment stating the model that yields the smallest time series validation MSE, MAE, and MAPE.

Submission details:

Submit one pdf file with your code, Markdown text, and output. The file should be uploaded to Quercus by the due date and time. There is 10% grade penalty for each new day of late submission.

ECO374 Problem Set 1 Submission

by Filip Zaleski

2024-03-07

Loading necessary Packages

```
options(repos = c(CRAN="http://cran.rstudio.com"))
if (!require("forecast")) install.packages("forecast")
if (!require("quantmod")) install.packages("quantmod")
if (!require("tsDyn")) install.packages("tsDyn")
if (!require("timetk")) install.packages("timetk")
library(quantmod); library(tsDyn); library(xts); library(timetk); library(stats);
library(forecast); library(ggplot2); library(ggfortify); library(ggpubr)
```

Initializing the provided ACF and PACF Plot Function

```
source("C:/Users/filip/OneDrive - University of Toronto/Courses/Third Year/Winter/ECO374 - Time Series and
```

Question 1: Obtaining S&P/TSX Data

Data: iShares S&P/TSX 60 Index ETF (XIU.TO) (Source)

```
XIU.TO_table <- quantmod::getSymbols("XIU.TO", src="yahoo", return.class="xts", auto.assign=F)
XIU.TO_table <- stats::window(XIU.TO_table, start=as.Date("2023-05-01"), end=as.Date("2024-02-01"))
XIU.TO <- XIU.TO_table$XIU.TO.Close
seed <- 2345

head(XIU.TO)
```

```
##            XIU.TO.Close
## 2023-05-01        31.57
## 2023-05-02        31.23
## 2023-05-03        31.16
## 2023-05-04        31.02
## 2023-05-05        31.47
## 2023-05-08        31.54
```

```
tail(XIU.TO)
```

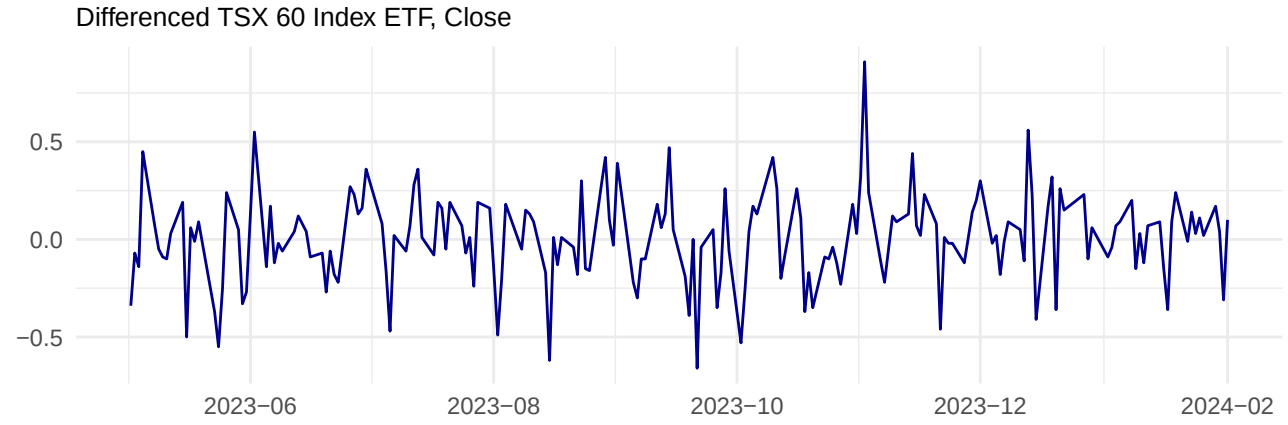
```
##            XIU.TO.Close
## 2024-01-25        32.26
## 2024-01-26        32.28
## 2024-01-29        32.45
## 2024-01-30        32.49
## 2024-01-31        32.18
## 2024-02-01        32.28
```

Question 2: Plotting the Data Series

```
ggplot(XIU.TO, aes(x=index(XIU.TO), y=XIU.TO.Close)) + geom_line(color="springgreen4") +
  labs(x="", y="", title="TSX 60 Index ETF") +
  theme_minimal() + theme(plot.title = element_text(size=10)) +
  scale_x_date(date_breaks="2 months", date_labels = "%Y-%m")
```

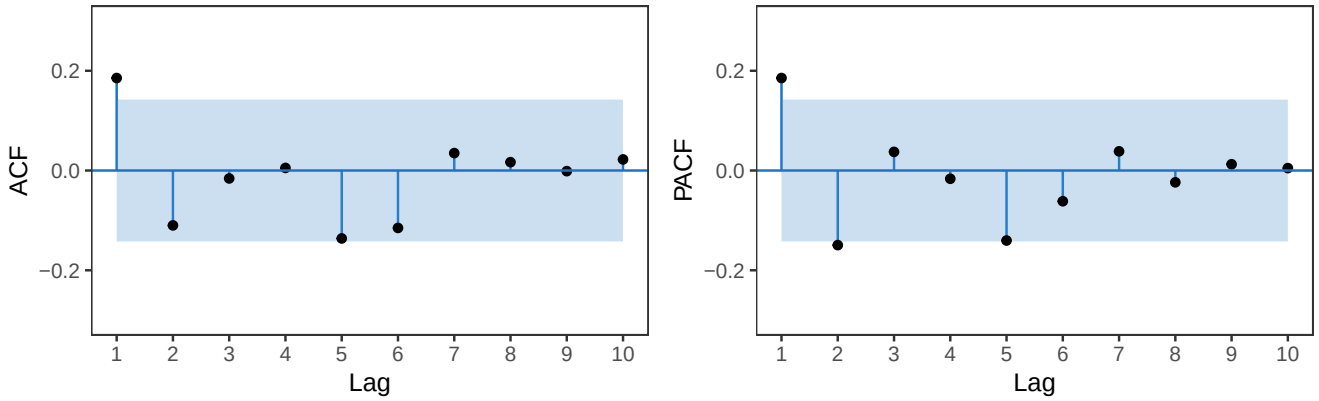
Question 3: Differencing the Data

```
D_XIU.TO <- na.omit(diff(XIU.TO, lag=1, differences=1))
ggplot(D_XIU.TO, aes(x=index(D_XIU.TO), y=XIU.TO.Close)) + geom_line(color="darkblue") +
  labs(x="", y="", title="Differenced TSX 60 Index ETF, Close") +
  theme_minimal() + theme(plot.title = element_text(size=10)) +
  scale_x_date(date_breaks="2 months", date_labels = "%Y-%m")
```



Question 4: ACF and PACF Evaluation

```r
lag_max <- 10
ylu <- c(-0.3,0.3)

ACF <- stats::acf(D_XIU.TO, lag.max=lag_max, plot=FALSE)
ACF_plot <- plot.autocor(type=ACF, ylims=ylu)

PACF <- stats::pacf(D_XIU.TO, lag.max=lag_max, plot=FALSE)
PACF_plot <- plot.autocor(type=PACF, ylims=ylu)

ggpubr::ggarrange(ACF_plot, PACF_plot, ncol=2, nrow=1)
```



Evaluating the ACF and PACF from this plot is tricky, as multiple lags are on the edge of significance. Ultimately, I ended up deducing that an ARMA(2,1) model is adequate. There are two significant spikes on the PACF, outlining an AR order of 2. The ACF, however, shows only 1 significant lag. I therefore choose an MA order of 1.

Question 5: Auto ARIMA

Running Auto ARIMA to determine the best-fitting model available.

```r
forecast::auto.arima(D_XIU.TO)
```

```
## Series: D_XIU.TO
## ARIMA(1,0,1) with zero mean
##
## Coefficients:
##           ar1    ma1
##       -0.2564  0.488
## s.e.   0.2219  0.198
##
## sigma^2 = 0.05081:  log likelihood = 14.44
## AIC=-22.88   AICc=-22.75   BIC=-13.14
```

Question 6: Obtaining the Best-Fitting ANN Model

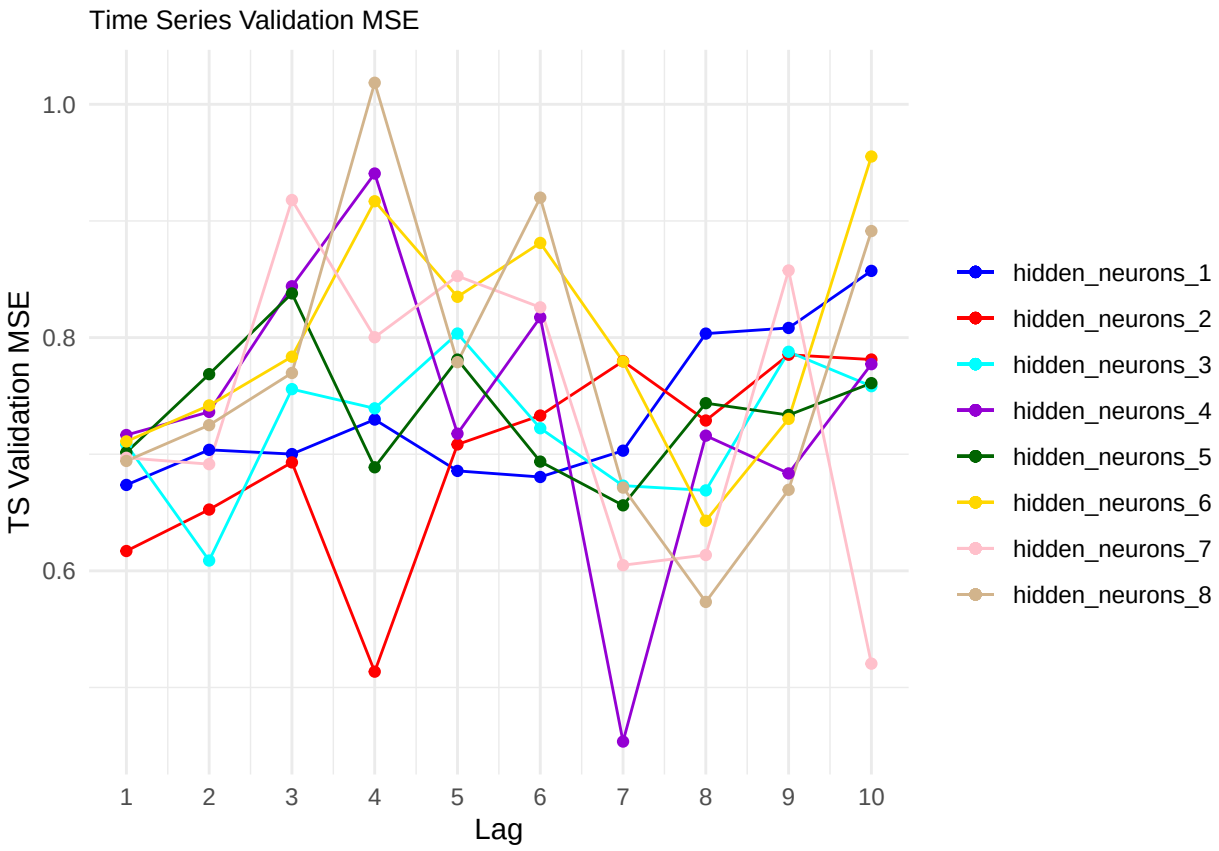
Loading necessary packages

```r
options(repos = c(CRAN="http://cran.rstudio.com"))
if (!require("quantmod")) install.packages("quantmod")
if (!require("stats")) install.packages("stats")
if (!require("forecast")) install.packages("forecast")
if (!require("timetk")) install.packages("timetk")
if (!require("ggplot2")) install.packages("ggplot2")
library(quantmod); library(stats); library(forecast); library(timetk); library(ggplot2)
```

Running TS C-Validation for my NNAR Model

```r
TSCV_nnetar <- function(data, p, P, size) {
  TT <- length(data)
  T1 <- floor(TT/5)
  step <- 20
  MSE_t <- matrix(0,nrow=TT-T1+1,ncol=1)
  y_hat <- MSE_t
  tseq <- seq(from=T1, to=TT, by=step)
  tseq <- tseq[-length(tseq)]
  for (j in tseq) {
    base::set.seed(seed)
    nnetar_model <- forecast::nnetar(y=data[1:j-1], p=p, P=P, size=size)
    NN_f <- forecast::forecast(nnetar_model,h=step)
    y_hat <- as.numeric(NN_f$mean)
    js <- j+step-1
    MSE_t[(j-T1+1):(js-T1+1)] <- (as.numeric(data[j:js])-y_hat)^2
  }
  MSE_validation <- mean(MSE_t[1:(js-T1+1)])
  return(MSE_validation)
}

size_max <- 8
lag_max <- 10
TSCV <- matrix(0, nrow=lag_max, ncol=size_max)

for (s in 1:size_max) {
  for (k in 1:lag_max) {
    TSCV[k,s] <- TSCV_nnetar(data=XIU.TO, p=k, P=0, size=s)
    #writeLines(paste("Size = ", s, "Lag = ", k, "  Validation MSE = ", TSCV[k,s]))
  }}

TSCV <- data.frame(TSCV)
colnames(TSCV) <- paste("hn", 1:size_max , sep="")

colors <- c("hidden_neurons_1"="blue", "hidden_neurons_2"="red", "hidden_neurons_3"="cyan",
            "hidden_neurons_4"="darkviolet", "hidden_neurons_5"="darkgreen", "hidden_neurons_6"="gold",
            "hidden_neurons_7"="pink", "hidden_neurons_8"="tan")
ggplot(data=TSCV, aes(x=index(TSCV))) +
  geom_line(aes(y=hn1, color="hidden_neurons_1")) + geom_line(aes(y=hn2, color="hidden_neurons_2")) +
  geom_line(aes(y=hn3, color="hidden_neurons_3")) + geom_line(aes(y=hn4, color="hidden_neurons_4")) +
  geom_line(aes(y=hn5, color="hidden_neurons_5")) + geom_line(aes(y=hn6, color="hidden_neurons_6")) +
  geom_line(aes(y=hn7, color="hidden_neurons_7")) + geom_line(aes(y=hn8, color="hidden_neurons_8")) +
  geom_point(aes(y=hn1, color="hidden_neurons_1")) + geom_point(aes(y=hn2, color="hidden_neurons_2")) +
  geom_point(aes(y=hn3, color="hidden_neurons_3")) + geom_point(aes(y=hn4, color="hidden_neurons_4")) +
  geom_point(aes(y=hn5, color="hidden_neurons_5")) + geom_point(aes(y=hn6, color="hidden_neurons_6")) +
  geom_point(aes(y=hn7, color="hidden_neurons_7")) + geom_point(aes(y=hn8, color="hidden_neurons_8")) +
  labs(x="Lag", y="TS Validation MSE", title="Time Series Validation MSE") +
```

```
  theme_minimal() + theme(plot.title = element_text(size=10)) + theme(legend.position = c(0.3, 0.75)) +
  scale_color_manual(name="", values=colors) +
  scale_x_continuous(breaks=index(TSCV), labels=paste(index(TSCV))) +
  theme(legend.position = "right") # moving the legend out of the way
```



Question 7: TS Validation for Differenced Data Under Various Models

```
data <- XIU.TO
D_data <- D_XIU.TO

for (m in 1:5) {

  TT <- length(data)
  T1 <- floor(0.2*TT)
  step <- 20
  MSE_t <- matrix(0,nrow=TT-T1+1,ncol=1)
  MAE_t <- MSE_t
  MAPE_t<- MSE_t
  tseq <- seq(from=T1, to=TT, by=step)
  tseq <- tseq[-length(tseq)]

  for (j in tseq) {

    last_data_point <- as.numeric(xts::last(data[1:j-1]))
```

```
    # 7.a: auto.arima model outlined in Q5
    if (m==1) {if (j==tseq[1]) {print("ARMA(1,1)")}
      model<- stats::arima(D_data[1:j-1], order=c(1,0,1), method="ML")
      fcst <- stats::predict(model, n.ahead=step)
      yhat <- as.numeric(fcst$pred)
      yhat <- last_data_point + cumsum(yhat)}

    # 7.b.: ARMA of choice from Q4
    if (m==2) {if (j==tseq[1]) {print("ARMA(2,1)")}
      model<- stats::arima(D_data[1:j-1], order=c(2,0,1), method="ML")
      fcst <- stats::predict(model, n.ahead=step)
      yhat <- as.numeric(fcst$pred)
      yhat <- last_data_point + cumsum(yhat)}

    # SETAR model with a threshold of 0
    if (m==3) {if (j==tseq[1]) {print("SETAR")}
      SETAR<- tsDyn::setar(D_data[1:j-1], mL=1, mH=1, th=0)
      fcst <- stats::predict(SETAR, n.ahead=step)
      yhat <- as.numeric(fcst)
      yhat <- last_data_point + cumsum(yhat)}

    # LSTAR modelwith th=0 and all other starting parameters set to 1
    if (m==4) {if (j==tseq[1]) {print("LSTAR")}
      LSTAR<- tsDyn::lstar(D_data[1:j-1], m=1, d=1, mL=1, mH=1, gamma=1, th=0, trace=FALSE)
      fcst <- stats::predict(LSTAR, n.ahead=step)
      yhat <- as.numeric(fcst)
      yhat <- last_data_point + cumsum(yhat)}

    # NNAR model from Q6
    if (m==5) {if (j==tseq[1]) {print("NNAR")}
      base::set.seed(seed)
      nnetar_model <- forecast::nnetar(data[1:j-1], p=7, P=0, size=4)
      fcst <- forecast::forecast(nnetar_model, h=step)
      yhat <- as.numeric(fcst[[16]][1:step])}

    js <- j+step-1
    MSE_t[(j-T1+1):(js-T1+1)] <- (as.numeric(data[j:js])-yhat)^2
    MAE_t[(j-T1+1):(js-T1+1)] <- abs(as.numeric(data[j:js])-yhat)
    MAPE_t[(j-T1+1):(js-T1+1)] <- 100*abs((as.numeric(data[j:js])-yhat)/yhat)
  }

  print(paste("MSE  =", mean(MSE_t[1:(js-T1+1),])))
  print(paste("MAE  =", mean(MAE_t[1:(js-T1+1),])))
  print(paste("MAPE =", mean(MAPE_t[1:(js-T1+1),])))
  print(" ")
}
```

```
## [1] "ARMA(1,1)"
## [1] "MSE  = 0.742643218585693"
## [1] "MAE  = 0.702863938986606"
## [1] "MAPE = 2.32232115054378"
## [1] " "
## [1] "ARMA(2,1)"
## [1] "MSE  = 0.746390272766187"
## [1] "MAE  = 0.705011624865027"
## [1] "MAPE = 2.32165205203187"
```

```
## [1] " "
## [1] "SETAR"
## [1] "MSE  = 0.653675678496461"
## [1] "MAE  = 0.658843293072703"
## [1] "MAPE = 2.15680271450815"
## [1] " "
## [1] "LSTAR"
## [1] "MSE  = 0.68464257036662"
## [1] "MAE  = 0.67985631882995"
## [1] "MAPE = 2.22512601311576"
## [1] " "
## [1] "NNAR"
## [1] "MSE  = 0.453650247902352"
## [1] "MAE  = 0.529421766444878"
## [1] "MAPE = 1.73426631830551"
## [1] " "
```

Question 8: Comments on Q7 Validation

The NNAR model created in question 6 yields the smallest validation MSE, and as per the model selection method outlined in this course, the NNAR model would be our model of choice. This follows with the NNAR model minimizing both the MAE and MAPE values, and by a considerable margin. Evidently, optimizing a neural network on the S&P/TSX Index was worth it; we managed to obtain a drastically more accurate model that maps the data in the best (most flexible but applicable) way.