Econometrics III HW - part 3

A. Schmidt and P. Assunção

24-3-2020

Loading packages and helper functions

See the source code if interested in all functions (chunks were ommited unless relevant for the assignment). The full code is available in:

Assignment 3

Introduction

Suppose that you are asked to analyze investment opportunities in the stock market by studying the dynamic behavior of 10 major stock prices. In particular, you are asked to study the stocks of Apple, Intel, Microsoft, Ford, General Electrics, Net ix, Nokia, Exxon Mobil, and Yahoo, as well as the S&P500 stock market index.

You can find the entire sample of daily data at your disposal in the Eviews workfile labeled data assign p3.wf1. This data set contains the daily stock prices of all the companies mentioned above and spans from the 14th of February 2007 to the 28th of January of 2013.

Importing and checking data

Import using read.csv2() function - the data is in an online directory and there is no need to change this address.

```
urlRemote  <- "https://raw.githubusercontent.com/aishameriane"
pathGithub <- "/Mphil/master/EconIII/data_assign_p3.csv"
token      <- "?token=AAVGJTT32POPYVK67LDJURK6QMGQG"

url      <- paste0(urlRemote, pathGithub, token)
dfData01 <- read.csv2(url, sep = ",", dec = ".", header = TRUE)
```

Check if everything is ok with the dataset: header and tail and summary statistics to check for missing data/outliers. We can see from the head and tail that the dataset that at least those observations seems to be completely filled with adequate ranges. To avoid breaking the margin spaces, we transposed the table, so the columns are in the rows and the original row names (which are the row positions) are appearing as the column names in the table below.

	1	2	3	1497	1498	1499
obs	1.00	2.00	3.00	1497.00	1498.00	1499.00
APPLE	84.63	85.44	85.25	460.00	451.69	437.83
DATE	732720.00	732721.00	732722.00	734891.00	734892.00	734895.00
EXXON_MOBIL	75.19	75.26	74.90	91.57	91.67	91.21
FORD	8.45	8.55	8.54	13.82	13.83	13.49
GEN_ELECTRIC	35.93	36.36	36.07	21.96	22.28	22.44
INTEL	20.96	21.21	21.26	21.10	21.04	21.01
MICROSOFT	29.17	29.58	28.91	27.70	27.58	28.01
NETFLIX	22.90	22.48	22.52	143.99	145.67	172.51
NOKIA	22.91	22.95	23.03	4.21	4.18	4.26
SP500	1443.91	1455.15	1456.77	1494.81	1494.82	1502.96
YAHOO	29.69	30.82	31.00	20.08	20.43	20.50

The column with the dates (`DATE`) is in numeric format and not in a nice position, so we will change to y-m-d and also switch places with the `APPLE` column.

```
# To discover the origin, first we computed ymd("2007-02-14")-732720.
dfData01$DATE <- as.Date(dfData01$DATE, origin="0001-01-01")
dfData01      <- dfData01[,c(1,3,2,4,5,6,7,8,9,10,11,12)]
```

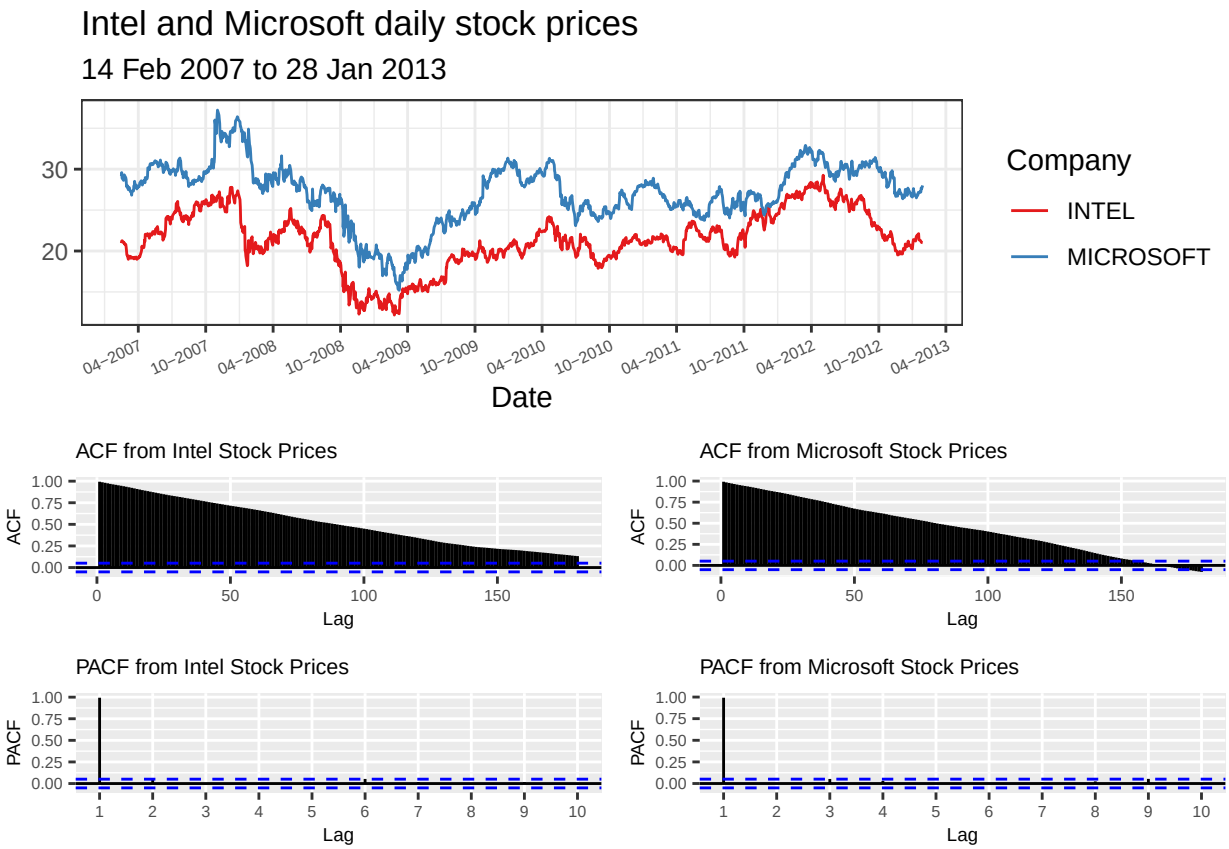
The next table has the descriptive statistics for the columns with information regarding the stocks. We can see that indeed we don't have any missing information and all values are numeric (there are no problems of formatting).

	Minimum	1st quartile	Mean	Median	3rd quartile	Maximum	Desv. Pad.
APPLE	79.39	139.670	275.6362	207.87	369.410	702.41	166.3191
EXXON_MOBIL	56.85	70.130	78.1742	79.69	85.935	95.08	9.2136
FORD	1.31	6.890	9.4389	9.46	12.085	18.81	3.7610
GEN_ELECTRIC	6.75	16.075	22.0685	19.08	28.135	42.03	8.6233
INTEL	12.17	19.630	21.3759	21.39	23.715	29.26	3.5034
MICROSOFT	15.20	25.240	27.1031	27.54	29.640	37.22	3.7298
NETFLIX	16.58	30.255	82.6573	57.45	109.910	300.70	68.9289
NOKIA	1.66	6.280	14.8695	12.55	22.480	41.70	10.3756
SP500	679.28	1104.265	1237.3728	1280.21	1392.240	1564.98	197.8307
YAHOO	9.10	15.010	18.2945	16.17	20.085	34.07	5.2936

The total number of observations is 1499.

Question 1

Provide plots for two stock market time-series at your choice and report 12-period ACF and PACF functions for those two time series. What does the sample ACF tell you about the dynamic properties of these stocks?



Comments on the graphs: Both stocks seems to move closely (top graph), which is not surprising given that they are in the same sector. However, Microsoft' series exhibits a higher level (average) than Intel's, which can be observed by the gap between the two series. In general, the gap seems constant, with exceptions in te beginning of 2009 and the end of 2011, where they it is smaller (series are close to each other). Also, we notice some periods when the gap is wider: in the second semester of 2007, both series were in an ascending escalade, but Microsoft value went higher and the gap was also wider; during 2009, Microsoft showed higher increase over its stock prices, culminating in a wider gap during the transition to 2010.

As for the ACF, we notice that there is a persistent behavior in both series (bottom graphs). That is, there is a slow autocorrelation decay. Almost all the lags up to 180 (6 months, approximately) are significant and there is no indication of a "cut" in some lag, as would be the case in an ARMA model with the MA coefficients different from zero. Moreover, the PACF, which is obtained by computing the autocorrelation that is remaining after considering the previous lags, is highly persistent for the first lag, as for the others up until lag 10 it seems that they all fall inside the confidence intervals (we also did for higher lags and observed the same behavior). So, at least for these two series, it seems that we are dealing with an AR(1) with an unit root (i.e., a random walk).

Question 2

Perform an ADF unit-root test for all the 10 time series using the general-to-specific approach based on the Schwarz Information Criterion (SIC). Report the values of the ADF test statistics. Is the unit-root hypothesis rejected for any time-series at the 90% confidence level? Did you expect to reject the unit-root hypothesis for some time-series at this confidence level? Justify your answer carefully.

From the lecture slides, we have that the Augmented Dickey Fuller (ADF) test for an AR(p) model can be obtained as follows.

Start from the AR(p) equation:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + \epsilon_t \quad \Rightarrow \quad \phi(L) X_t = \epsilon_t \tag{1}$$

where $\phi(L) = 1 - \phi_1 L - \cdots - \phi_p L^p$. There is unit root if $\phi_1 L + \cdots + \phi_p L^p = 1$. An easier way to test if this is true is by writing the model above in terms of first differences in the AR(p) model above.

$$\Delta X_t = (\phi_1 L + \cdots + \phi_p L^p - 1) X_{t-1} + \phi_2^* \Delta X_{t-1} + \phi_3^* \Delta X_{t-2} + \cdots + \phi_p^* \Delta X_{t-p+1} + \epsilon_t \tag{2}$$

$$\Delta X_t = \beta X_{t-1} + \phi_2^* \Delta X_{t-1} + \phi_3^* \Delta X_{t-2} + \cdots + \phi_p^* \Delta X_{t-p+1} + \epsilon_t \tag{3}$$

where $\phi_j^* = -\sum_{i=j}^{p} \phi_i$ for $j = 2, \cdots, p$. Therefore, since zero *beta* would imply unit roots, the ADF test will be testing the null hypothesis $H_0 : \beta = 0$ being equal to zero versus the alternative $H_A : -2 < \beta < 0$, which implies stationarity. The test statistic of such test is $ADF = \frac{\hat{\beta}}{SE(\hat{\beta})}$, which follows a Dickey-Fuller distribution.

Our procedure then will be:

- For each serie, estimate an AR(10) model with drift using the same function used in part 1 (`Arima()`);
- In sequence, remove the lags, and store the SIC value for each combination, until reach an AR(1);
- Recover which model had the smallest SIC (BIC is the same, see: https://en.wikipedia.org/wiki/Bayesian_information_criterion);
- Perform the ADF test considering the model with best BIC.

We opted by not estimating an `ARMA(p,q)` model to avoid parameter redundancy (as discussed in Chapter 7 of Box et al. (2015)). In particular, since we know in advance (from previous courses) that it is not possible to predict stock prices (or returns) and the usual procedure in finance is to model the variance, we already expected beforehand that any `ARMA(p,q)` would not be correctly specified anyway and focused in just making the exercise using the `AR(p)` model. But to make things more interesting, we challenged ourselves in developing the routine to pick the best `AR(p)` model from all possible lag combinations (including the removal of intermediate lags).

```
# This function will take the series and test all combinations of lags
# up until lag max to fit an AR model
# Default method is MLE, in some cases will
# change the estimation method to quasi likelihood (therefore no AIC)
# It returns the model with the minimum BIC

fEstAR <- function(data, series, lagmax){

    # Extract the correct series
    tsData <- xts(data[,series], order.by = data[,2])

    # Prepare the combinations of the lags
    matriz <- list(matrix(rep(NA,1), ncol = 1))

    if (lagmax > 1){
        for (i in 2:lagmax){
          matriz[[i]] <- matrix(rep(NA,(2^(i-1))*i), ncol = i)
        }
    }

        # Assemble a list with all possible combinations of lags up until an AR(15)
        for (j in 1:lagmax){
          # The idea is that our objects are of this type
```

```
  #matriz[[j]][,] <- matrix(rep(0,(2^(j-1))*j), ncol = j, nrow = (2^(j-1)))
  # Now we fill with the bynary representation of the numbers,
  #this gives all the possible combinations
  for (i in (2^(j-1)):(2^(j)-1)){
      # Feeds with the bynary combinations
      matriz[[j]][i-(2^(j-1)-1),] <- as.integer(intToBits(i)[1:j])
  }
}

# Now we just reorganize to use each line in the Arima() function
for (j in 1:lagmax){
  # Converts to what we need for the Arima() fix argument
  matriz[[j]] <- ifelse(matriz[[j]] == 1, NA, 0)
  # Reverses the order to make consistent with the first column being higher lag
  matriz[[j]] <- matriz[[j]][ , ncol(matriz[[j]]):1]
}

# Get the number of models

nModels <- 1
for (j in 2:length(matriz)){
  nModels <- nModels + nrow(matriz[[j]])
}
dfModels <- data.frame(rep(NA, nModels), rep(NA, nModels))
names(dfModels) <- c("BIC", "Model")

#tic("Total time:") ## Use for debugging
for (j in 1:length(matriz)){
    #tic(paste(c("AR ", j, ":"))) ## Use for debugging
    order <- j
    matriz2 <- matriz[[j]]

    for (i in 1:max(1, nrow(matriz2))) {
      coef  <- i

      if (j == 1){
        model <- Arima(tsData, order = c(order,0,0), fixed = c(matriz2[coef], NA))
      } else {
        model <- try2(Arima(tsData, order = c(order,0,0),
                            fixed = c(matriz2[coef, ], NA), method="CSS-ML",
                            optim.method = "BFGS"),
                      Arima(tsData, order = c(order,0,0),
                            fixed = c(matriz2[coef, ], NA), method="CSS"))
      }

      dBIC  <- model$bic
      if (j == 1){
        dfModels[((i-1)+2^(j-1)),] <- c(dBIC,
                                        paste("AR(", as.character(order),
                                              "), with coef (", NA, ")", sep=""))
      } else {
        dfModels[((i-1)+2^(j-1)),] <- c(dBIC,
                                        paste("AR(", as.character(order),
```

```
                                                    "), with coef (", paste(matriz2[coef, ],
                                                     collapse=", "), ")", sep=""))
              }

            }
        #toc() ## Use for debugging
        }
        #toc() ## Use for debugging

        selModel <- dfModels[which(dfModels[,1] == min(dfModels[,1], na.rm=TRUE)),]

    return(selModel)
}
```

Now we are ready to estimate the best (using BIC criteria) AR model for all the stocks.

The results from the best models for each company stock are summarized below.

```
## [1] "Best AR model for APPLE"
```

Variable	Estimate	Std. Error	t-statistic	P-Value
Lag 1	0.88	0.03	34.43	0.00
Lag 2	0.12	0.03	4.48	0.00
Intercept	275.44	143.85	1.91	0.06

```
## [1] "Best AR model for EXXON_MOBIL"
```

Variable	Estimate	Std. Error	t-statistic	P-Value
Lag 1	0.89	0.02	47.04	0
Lag 2	0.10	0.02	5.24	0
Intercept	78.66	3.81	20.67	0

```
## [1] "Best AR model for FORD"
```

Variable	Estimate	Std. Error	t-statistic	P-Value
Lag 1	1.00	0.00	612.13	0
Intercept	9.45	2.11	4.47	0

```
## [1] "Best AR model for GEN_ELECTRIC"
```

Variable	Estimate	Std. Error	t-statistic	P-Value
Lag 1	0.92	0.03	35.75	0
Lag 2	0.08	0.03	3.04	0
Intercept	24.05	6.24	3.85	0

```
## [1] "Best AR model for INTEL"
```

Variable	Estimate	Std. Error	t-statistic	P-Value
Lag 1	0.99	0.00	346.85	0
Intercept	21.37	1.32	16.21	0

```
## [1] "Best AR model for MICROSOFT"
```

Variable	Estimate	Std. Error	t-statistic	P-Value
Lag 1	0.99	0.00	320.25	0
Intercept	27.11	1.35	20.13	0

```
## [1] "Best AR model for NETFLIX"
```

Variable	Estimate	Std. Error	t-statistic	P-Value
Lag 1	1.0	0.00	743.43	0.00
Intercept	97.5	47.83	2.04	0.04

```
## [1] "Best AR model for NOKIA"
```

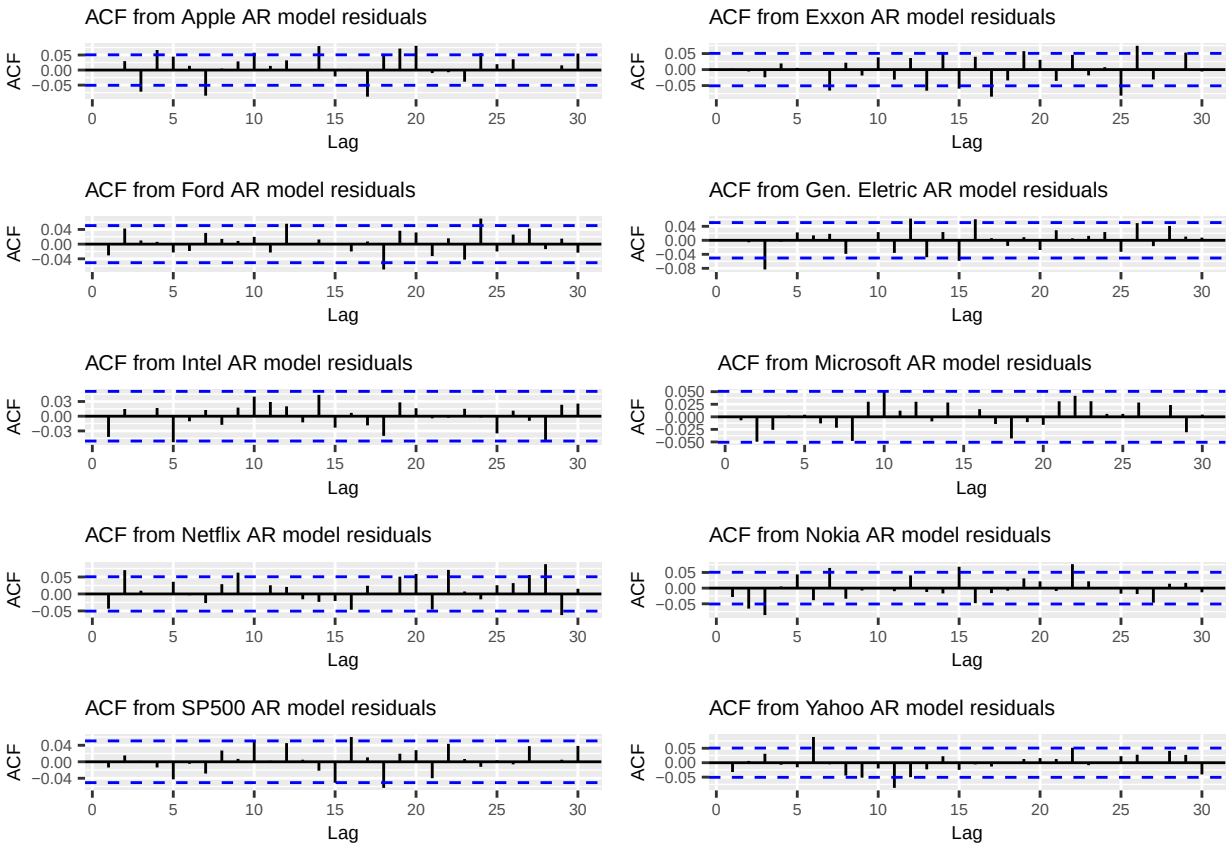
Variable	Estimate	Std. Error	t-statistic	P-Value
Lag 1	1.00	0.00	927.61	0.00
Intercept	14.85	7.01	2.12	0.03

```
## [1] "Best AR model for SP500"
```

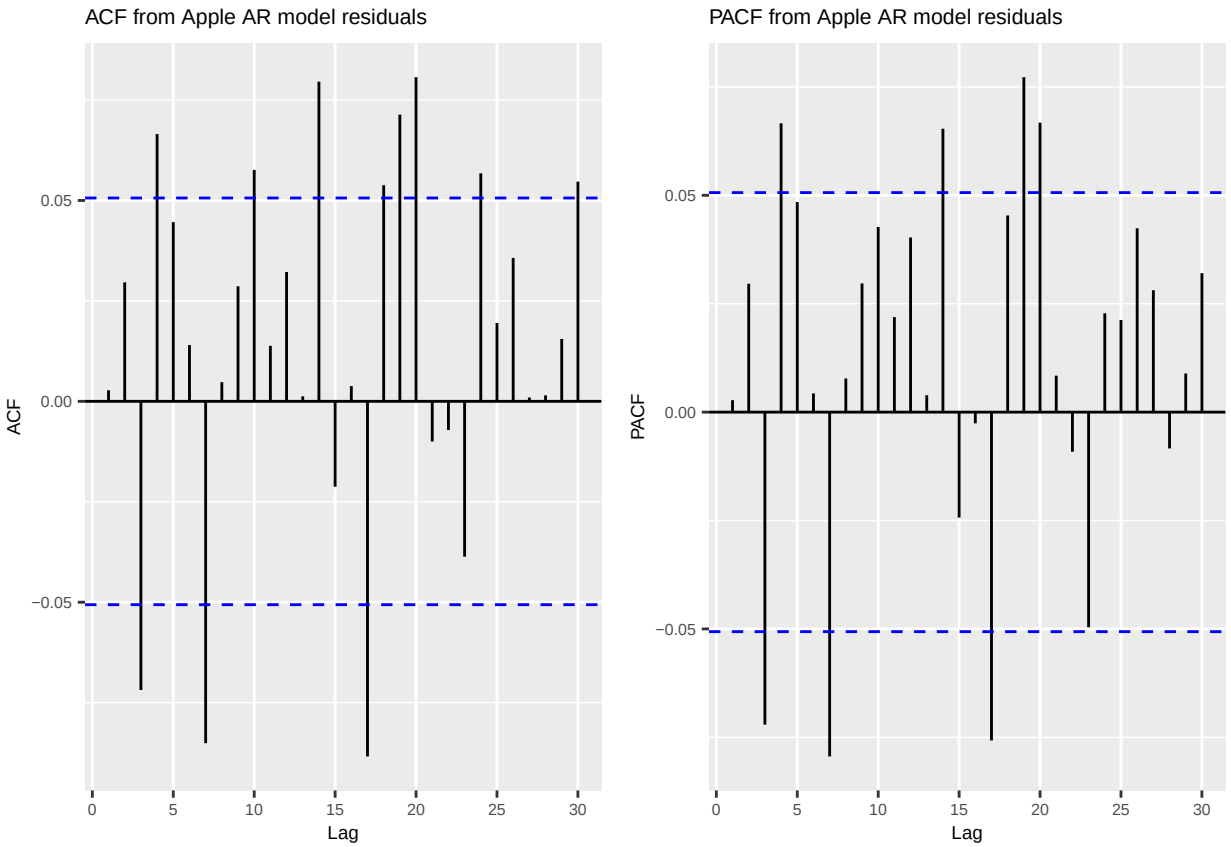
Variable	Estimate	Std. Error	t-statistic	P-Value
Lag 1	0.94	0.02	49.47	0
Lag 2	0.06	0.02	3.25	0
Intercept	1281.50	116.02	11.05	0

```
## [1] "Best AR model for YAHOO"
```

Variable	Estimate	Std. Error	t-statistic	P-Value
Lag 1	0.94	0.02	58.34	0
Lag 2	0.06	0.02	3.57	0
Intercept	19.99	3.15	6.34	0



Comments: Although the models indeed show only significant coefficients, the ACF graphs of the residuals shows that some models might not be correctly specified, since not all lags are inside of the confidence intervals. For example, the ACF and PACF of the residuals from the AR(2) model estimated for Apple stocks are represented below. Clearly they do not resemble the expected graphs from a white noise (which we would have in the case of correct specification).



To further investigate this, we used an automatic model selection function, `auto.arima()`, which fits an ARIMA(p,d,q) model to the data. According to the function, the best model would be an ARIMA(0,1,0), i.e., no lag for the AR neither for the MA part, only a differencing term. However, since the goal of this exercise is not to find the best fit but work with the unit roots, we decided to not continue in this direction.

Now that we have our final models, we can compute the ADF statistic.

Dickey Fuller test

Following the lecture slides, we know that for an $AR(p)$ model, we are interested in computing $\beta = \phi_1 + \phi_2 + \ldots + \phi_p - 1$ and our test is $H_0 : \beta = 0$ versus $H_1 : -2 < \beta < 0$. The test statistic is $ADF = \frac{\hat{\beta}}{SE(\hat{\beta})}$. Although our "best" models are considering combinations of lags (with ommited intermediate lags), the computation of the test statistic would be cumbersome. **Due to restrictions of time, we will use just all the lags instead removing the intermediate ones that might not be significant.**

The function below is based on the `adf.test()` function from the package `tseries`. We changed slightly to take our series more easily from the dataframe and to organize the output for better visualization.

```
adfTest <- function (data, models, series, alternative = c("stationary", "explosive")) {

        model <- models[[series]]
        x     <- xts(data[,series+2], order.by = data[,2])
        k     <- length(model$coef)-1

        if ((NCOL(x) > 1) || is.data.frame(x))
          stop("x is not a vector or univariate time series")
```

```
if (any(is.na(x)))
  stop("NAs in x")
if (k < 0)
  stop("k negative")
alternative <- match.arg(alternative)
#DNAME <- deparse(substitute(x))
DNAME  <- colnames(data[series+2])
k <- k + 1
x <- as.vector(x, mode = "double")
# Takes the first difference, i.e, y = \Delta x_t
y <- diff(x)
n <- length(y)
# creates a length(y)-k \times k matrix with lags of the
#first difference series
z <- embed(y, k)
# This is the 1st lag of the first difference series
yt <- z[, 1]
# This is just to make size compatible
xt1 <- x[k:n]
# This is a sequence of numbers from k to n
tt <- k:n
if (k > 1) {
  # For more than one lag in the original model,
  #you need this adittional guys, they are lags of y
  yt1 <- z[, 2:k]
  # To understand the regression, use head(cbind(yt, xt1, x))
  # The lag of the first difference, yt,
  #is being regressed against the lag of the original series,
  #similar to the slides
  res <- lm(yt ~ xt1 + 1 + tt + yt1)  # This is for AR(p), p>1
}
else res <- lm(yt ~ xt1 + 1 + tt)     # This is for AR(1)
res.sum <- summary(res)
# Compute the test statistic
STAT <- res.sum$coefficients[2, 1]/res.sum$coefficients[2,2]
table <- cbind(c(4.38, 4.15, 4.04, 3.99, 3.98, 3.96), # Critical values
               c(3.95, 3.8, 3.73, 3.69, 3.68, 3.66),
               c(3.6, 3.5, 3.45, 3.43, 3.42, 3.41),
               c(3.24, 3.18, 3.15, 3.13, 3.13, 3.12),
               c(1.14, 1.19, 1.22, 1.23, 1.24, 1.25),
               c(0.8, 0.87, 0.9, 0.92, 0.93, 0.94),
               c(0.5, 0.58, 0.62, 0.64, 0.65, 0.66),
               c(0.15, 0.24, 0.28, 0.31, 0.32, 0.33))
table <- -table
tablen <- dim(table)[2]
tableT <- c(25, 50, 100, 250, 500, 1e+05)
tablep <- c(0.01, 0.025, 0.05, 0.1, 0.9, 0.95, 0.975, 0.99)
tableipl <- numeric(tablen)
for (i in (1:tablen)) tableipl[i] <- approx(tableT, table[,i], n, rule = 2)$y
# The next line locates the statistic in
#terms of the critical values and gives the corresponding p-value
interpol <- approx(tableipl, tablep, STAT, rule = 2)$y
if (!is.na(STAT) && is.na(approx(tableipl, tablep, STAT,rule = 1)$y))
```

```
            if (interpol == min(tablep))
              warning("p-value smaller than printed p-value")
            else warning("p-value greater than printed p-value")
          if (alternative == "stationary")
            # If the test is H1 = stationary, then a p-value
            #above 0.1 will show evidence of an unit root (we are using this test)
            PVAL <- interpol
          else if (alternative == "explosive")
            # If the test is H1 = explosive, then a p-value
            #below 0.1 will show evidence of an unit root
            PVAL <- 1 - interpol
          else stop("irregular alternative")
          PARAMETER <- k - 1
          METHOD <- "Augmented Dickey-Fuller Test"
          names(STAT) <- "Dickey-Fuller"
          names(PARAMETER) <- "Lag order"
          return(data.frame(statistic = STAT, parameter = PARAMETER,
            alternative = alternative, p.value = PVAL, method = METHOD,
            data.name = DNAME))
          #structure(list(statistic = STAT, parameter = PARAMETER,
          #  alternative = alternative, p.value = PVAL, method = METHOD,
          #  data.name = DNAME), class = "htest")
}
```

```
## [1] "Results from the ADF test"
```

Company	p of AR(p)	Test-statistic	p-value
APPLE	2	-1.5281	0.7781
EXXON_MOBIL	3	-1.9190	0.6126
FORD	1	-1.7744	0.6738
GEN_ELECTRIC	2	-0.9946	0.9396
INTEL	1	-2.1926	0.4968
MICROSOFT	1	-2.4319	0.3955
NETFLIX	1	-1.3892	0.8369
NOKIA	1	-2.2834	0.4584
SP500	3	-1.3469	0.8548
YAHOO	4	-2.1792	0.5024

Comments: From the table with the the ADF test, using as null-hypothesis that the series has a unit root, we cannot reject H_0 for any ot the companies using a significance level of 10%. This result means that none of the stock prices series were generated by a stationary process. However, we give a word of caution: as mentioned in the other parts of this assignment, there is evidence that the AR models estimated were not correctly specified (look back at the residual analysis, for example). Overall, the identification of the presence of unit roots is a first step towards investigating issues related to spurious regression.

From the economic point of view, since the series are prices, not returns, it is not surprising that they all exibit unit roots using the test. Prices usually have a long dependency through the price level, which is known to be non-stationary because it is the accumulation of past prices. This is a similar problem we would have faced in part one when using GDP if we used the level GDP and not the growth rate.

Question 3

Assume that both the stocks of Apple and Microsoft follow a random walk process. Produce a 5-day forecast for the stocks of Apple and Microsoft. Add 95% confidence bounds to your forecasts under the assumption of Gaussian innovations. Is there any investment advice you can give on these stocks? Is their value expected to increase or decrease?

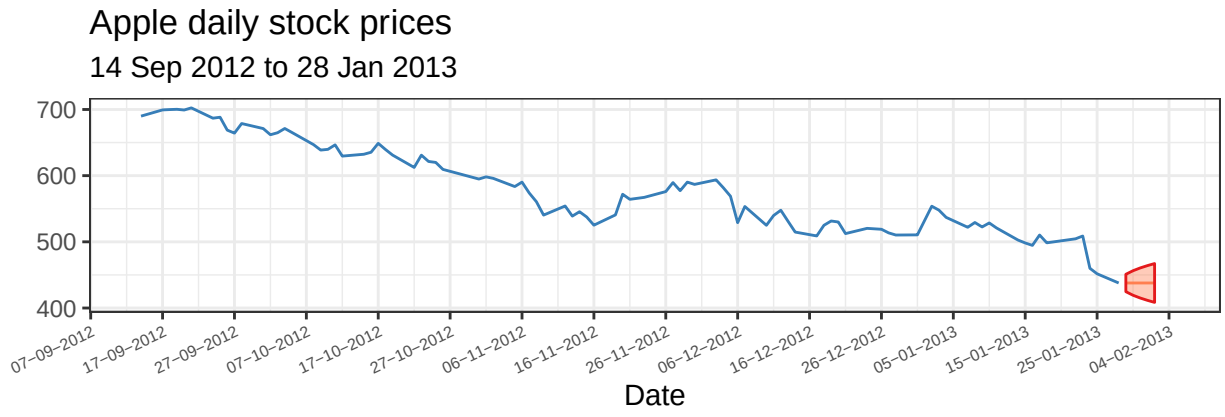
The random walk equation is given by:

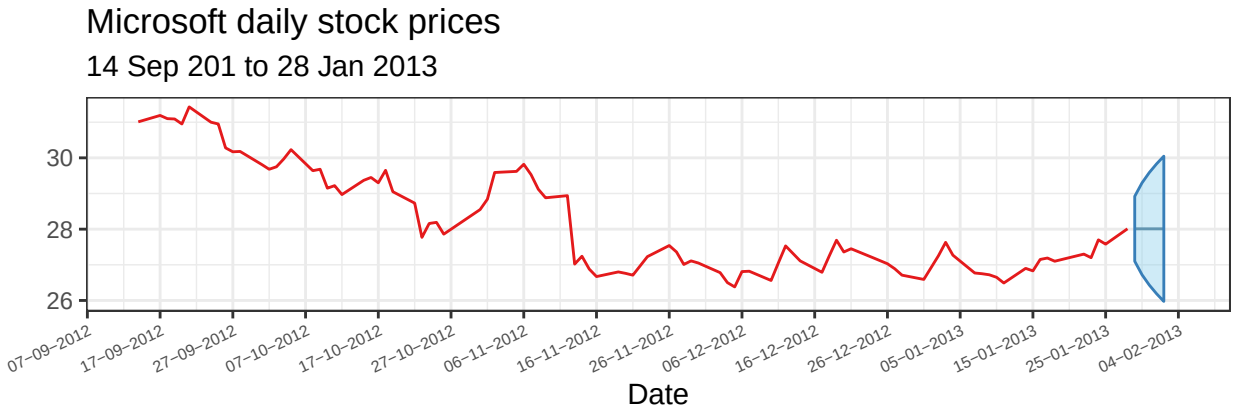
$$X_t = X_{t-1} + \varepsilon_t$$

where $\varepsilon_t \sim \text{WN}(0, \sigma^2)$. As discussed in tutorial 4, the forecast to h period ahead is given by:

$$\hat{X}_{T+h} = 1^h \cdot \sigma^2 = \sigma^2$$

and the variance of the prediction error will be given by $h \cdot \sigma^2$. To compute things properly, we need an estimate for σ^2. Our procedure was to fit an AR(1) model (using the standard `lm()` function from R, since the `Arima()` function complains about the unit root) to obtain the estimate for σ^2 and use the formulas to obtain the forecasts.





Comments: "Plots were made in different graphs due to scale differences and to be able to better visualize the forecast region we only ploted the last 90 days of each series. As expected, the forecasts are not informative about the prices because, as stated above, they are just the standard deviation of the innovations, which means that forecasts basically look at the innovations to compute the next period price. That is, random walk model applied in a finance context would imply that prices tomorrow cannot be predicted from historical trend, since prices today will not tell you what will happen with the prices tomorrow. With this in mind, we would not give any investment advice based on this kind of model, given it does not help when

it comes to predicting stock prices. In other words, it is not possible, based on this model, to say if we expected the price to go up or down, since the forecast is always the same last value

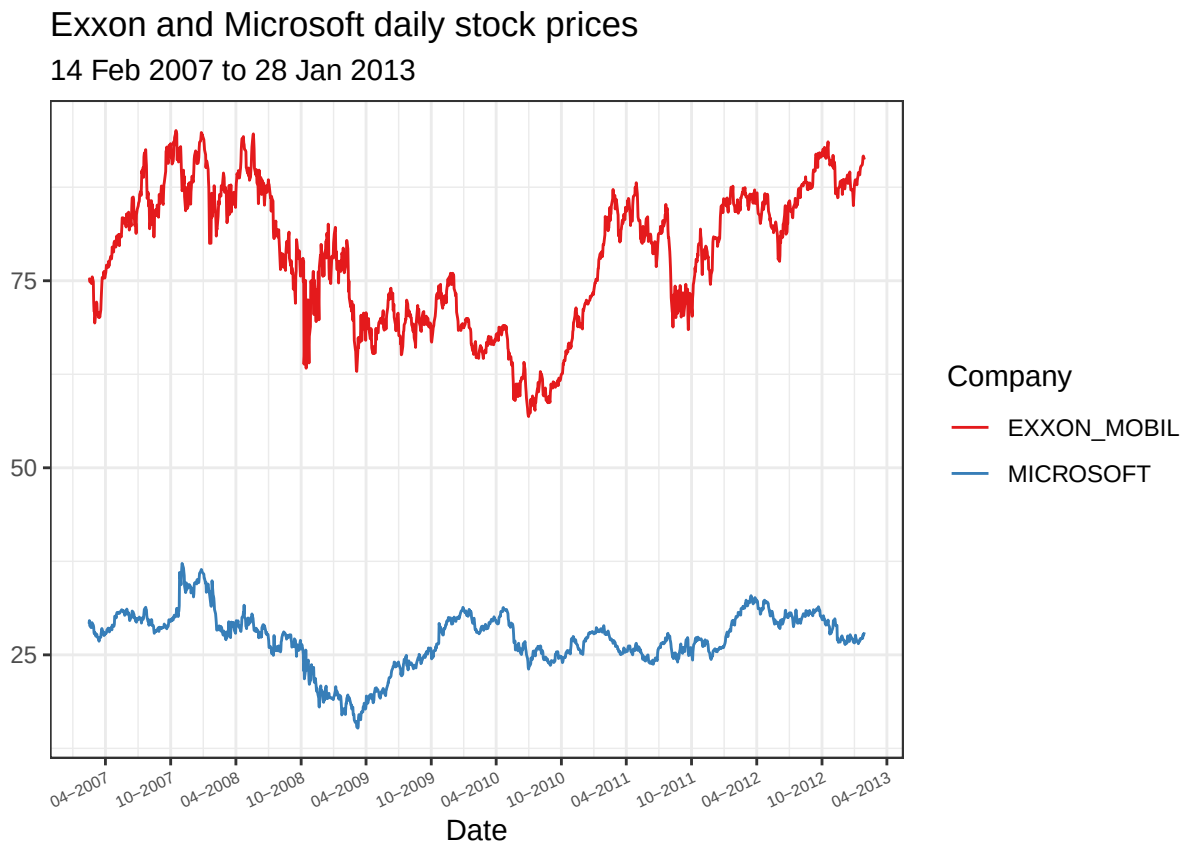
Question 4

Please investigate the following claim:

"Financial analysts have found that changes in the price of Microsoft stocks can be largely explained by fluctuations in the market value of Exxon Mobile. According to these analysts, this shows the extent to which the Microsoft Corporation is currently exposed to the market performance of the oil and gas industry."

Do you find a statistically significant contemporaneous relation between Microsoft and Exxon Mobile stock prices? Do you agree that changes in Microsoft stock prices are largely explained by uctuations in the stock price of Exxon Mobile? Justify your answer.

To better answer this question, first we shall make the visual inspection in both series.



Comments: Given the different scale in the stock prices, the visual inspection is not clear that the series move together, as it was for Intel and Microsoft prices. However, we can notice some common downwards and upward trends in the series, which might suggest that they are both $I(1)$ and candidate to be cointegrated.

So, we proceed to analyze if in fact the series are cointegrated to rule out spurious regression. As we discussed in the lectures, if two series Y_t and X_t are cointegrated, then $Y_t - \lambda X_t$ is stationary. Since we do not have the value for λ, it must first be estimated.

A simple way to do this is first compute the regression $Y_t = \beta_0 + \beta_1 X_t + \varepsilon_t$ to get an estimate for β_1 using OLS. Then, we compute the residuals using $\hat{\varepsilon}_t = Y_t - \hat{\beta}_0 - \hat{\beta}_1 X_t$ and verify the stationarity using the ADF test routine that we used before in previous tests (since now we have a single series and not several as before, it was best to use the package function directly and not our modified version).

Using the null hypothesis that the residuals are not stationary, we obtained a test statistic of -2.2186, which corresponds to a p-value of 0.4858, so the conclusion is that we cannot reject H_0 for a significance level of 10%.

Therefore, the analysis leads us to conclude that both series are not cointegrated and it is not correct to use Exxon prices to explain Microsoft prices. Apart from the statistical analysis, even without this results, we think it would be hard to sell an economic story on how the oil and gas prices could affect Microsoft shares. It is true that oil prices have an impact in production of goods and services in the real world, because they affect directly the transportation sector. However, Microsoft is a software industry, mostly, therefore the delivery of its products should not be so sensible to gas and oil, even if the demand for its products rise as a consequence of the oil price fluctuation. And, of course, in case of increase in demand, it could be that Microsoft needs to increase the inputs, but this means hiring more workers, which also would not be affected by gas prices.

References

Box, George EP, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. 2015. *Time Series Analysis: Forecasting and Control*. John Wiley & Sons.