Unit 10 Live Session (Solutions)

Time Series Analysis Lecture 5: Vector Autoregressive (VAR) Models

Figure 1: South Hall

Class Announcements

- Lab-2 due in 1 week

Roadmap

Rearview Mirror

- Univariate Time Series Models
  - Estimation, model diagnostics, model identification, model selection, assumption testing, and statistical inference/forecasting

Today

- Regression with multiple trending time series
- Spurious regression
- Cointegration
- Multivariate Time Series Models: Vector Autoregressive (VAR) model
- Notion of cross-correlation

Looking Ahead

- Introduction to panel data
- Using the OLS regression model on panel data
- Exploratory panel data analysis
- First-Difference models
- Distributed Lag models

Start-up Code

```
# Start with a clean R environment
rm(list = ls())
## Load a set of packages including broom, cli, crayon, dbplyr , dplyr, dtplyr, forecast,
#googledrive, googlesheets4, ggplot2, haven, hms, httr, jsonlite, lubridate , magrittr,
#modelr, pillar, purrr, readr, readxl, reprex, rlang, rstudioapi, rvest, stringr, tibble,
#tidyr, xml2
library(tidyverse)
## To load All data sets in the book "Forecasting: principles and practice."
#by Rob J Hyndman and George Athanasopoulos
library(fpp3)
# To create and work with tidy temporal data
library(tsibble)
# To work with date-times and time-spans
library(lubridate)
# Provides a collection of commonly used univariate and multivariate time series models
library(fable )
## To interact directly with the Quandl API and download data
library(Quandl)
# For analyzing tidy time series data.
library(feasts)
# Provides methods and tools for displaying and analyzing univariate time series forecasts
library(forecast)
# For estimation, lag selection, diagnostic testing, forecasting, and impulse response functions of VAR
library(vars)
#provides tools for statistical calculations
library(stats)
# To assist the quantitative trader in the development,
#testing and deployment of statistically based trading models.
library(quantmod)
# For statistical analysis
library(car)
## To retrieve and display the information returned online by Google Trends
library(gtrendsR)
# To do time series analysis and computational finance.
library(tseries)
```

Multivariate Time Series Models

The goal of a researcher working with time series data does not differ much from that of a researcher working with cross-sectional data. They use regression to explore the relationship between two or more variables.

However, we will face three problems in time series data that we will not encounter using cross-sectional data:

1- A time series variable can be influenced by lags of itself or lags of other variables

2- If the variable is non-stationary, a problem known as spurious regression may arise.

We can address the first problem by following two types of models:

1- The DL(q) (Distributed Lag) Model:

$$Y_t = \alpha + \beta_0 X_t + \beta_1 X_{t-1} + ... + \beta_q X_{t-q} + \epsilon_t$$

Here, the effect of the explanatory variable does not happen all at once but over several periods, and DL(q) incorporates such dynamic effects.

2- ARDL(p,q) (Autoregressive Distributed Lag) Model

$$Y_t = \alpha + \phi_1 Y_{t-1} + ... + \phi_p Y_{t-p} + \beta_0 X_t + \beta_1 X_{t-1} + ... + \beta_q X_{t-q} + \epsilon_t$$

Here, The dependent variable Y depends on p lags of itself and q lags of X.

Estimation and interpretation of the DL(q) and ARDL(p,q) model depend on whether the series X and Y are stationary or not.

If Y_t and X_t are stationary, the DL(q) and ARDL(p,q) models can be estimated consistently by ordinary least squares.

However, if Y_t and X_t are non-stationary, a problem known as spurious regression may arise.

Spurious Regression Example

- In time series analysis, we have to be particularly careful since an apparent relationship with significant coefficients of the (spurious) regression can be obtained not because the response 'truly' depends on the explanatory variables but because of the deterministic or stochastic time trends "hidden" in these variables.
- If Y_t is stationary, or its residuals ϵ_t in the decomposition $Yt = T_t + S_t + \epsilon_t$ are stationary, then Y_t is called a **Trend Stationary (or TS)** series;
- If Y_t has a unit root then its difference $\Delta Y_t = Y_t - Y_{t-1}$ is stationary, and it is called a **Difference Stationary (or DS)** series;

A correct way to distinguish between TS and DS stationary is through a unit root test.

Spurious Regression when both Y and X are Trend stationery (TS)

- Consider the following time series:

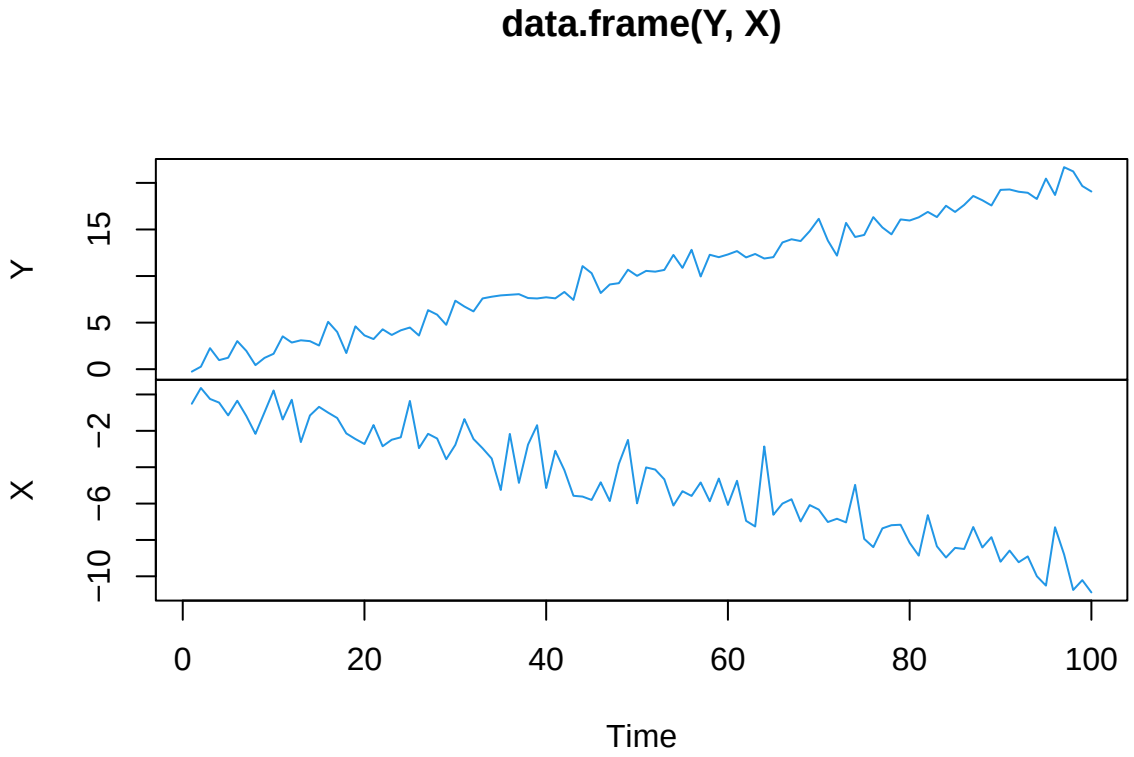
$$Y_t = 0.1 + 0.2 \cdot t + W_t^1$$
$$X_t = 0.3 - 0.1 \cdot t + W_t^2$$

- W_t^i is Gaussian white noise with mean zero and standard deviation 1.

a) Simulate 100 realizations of Y_t and X_t and estimate following model. What do you notice?

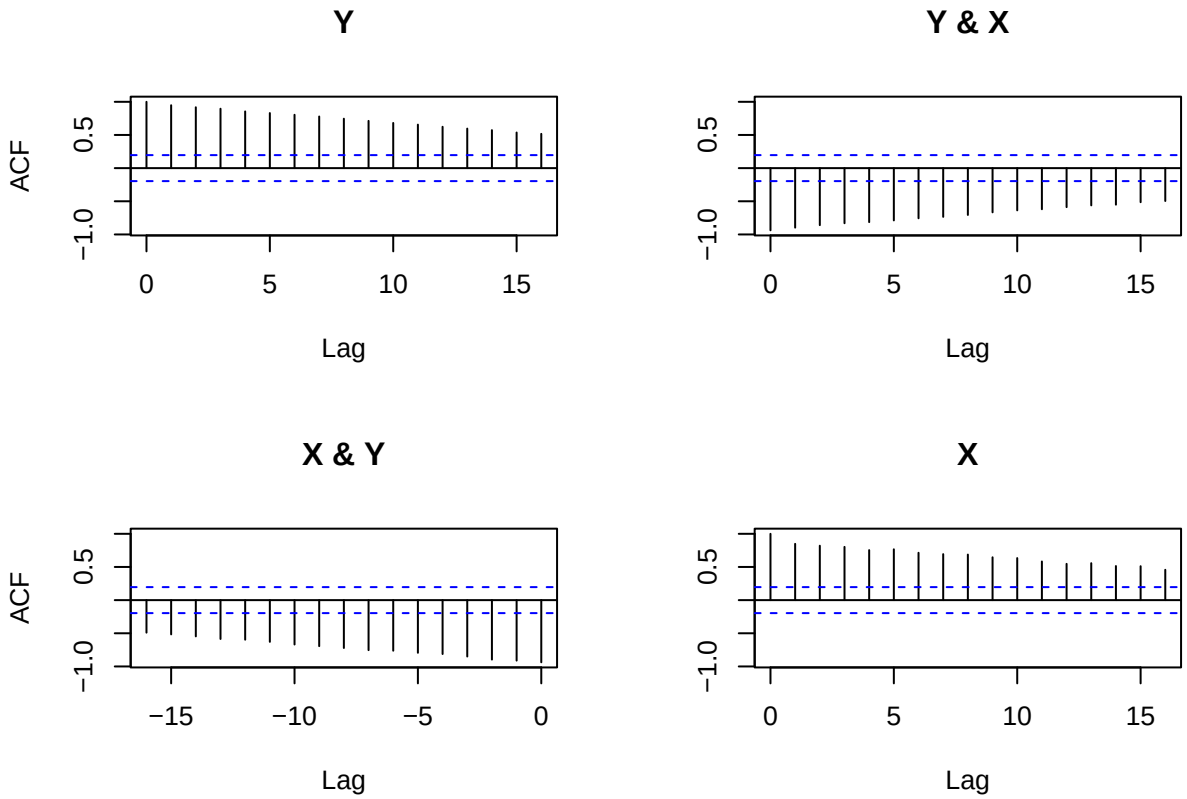
$$Y_t = \beta_0 + \beta_1 X_t + \epsilon_t$$

```
set.seed(123)
n = 100
Y <- 0.1 + 0.2 * 1:n + rnorm(n)
X <- 0.3 - 0.1 * 1:n + rnorm(n)
Y <- ts(Y)
X <- ts(X)
plot.ts(data.frame(Y, X), col = 4)
```



Here, we know that both X and Y were generated independently; however, because of these deterministic trends, X and Y might seem to be negatively correlated

```
acf(data.frame(Y, X))
```



The cross-correlation plots of Y and X also show that the correlation between these variables is negative

```
model1 <- lm(Y ~ X)
round(summary(model1)$coefficients, 4)
```

```
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept)   1.1684     0.3996   2.9237   0.0043
## X            -1.8779     0.0703 -26.7157   0.0000
```

Also, The estimated coefficient of X is negative and statistically significant.

In this case, the seemingly (or spuriously) significant coefficients of the (spurious) regression can be obtained not because the Y 'truly' depends on the X, but because of the trends 'hidden' in these variables, So both Y and X increase because of a trend, and

increase in Y is not because of increase in X. So we have a spurious regression here as it may seem that Y depends on X when they are indeed independent.

b) Now estimate the following model. What do you expect before estimating the model?

$$Y_t = \beta_0 + \beta_1 t + \beta_2 X_t + \epsilon_t$$

```
model2 <- lm(Y ~ X + time(X))
round(summary(model2)$coefficients, 4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.0653     0.1850  0.3529   0.7249
## X            -0.0545     0.0958 -0.5687   0.5708
## time(X)       0.1972     0.0098 20.1151   0.0000
```

If we include a trend in the model, we can see that the coefficient of X is not significant anymore.

So, if both Y and X are TS stationary and have deterministic trends, we can avoid spurious regression problems by including a trend variable in the model.

Spurious Regression when both Y and X are Difference Stationery (DS)

- Assume X and Y are two independent random walks without drift:

$$Y_t = Y_{t-1} + W_t^1$$

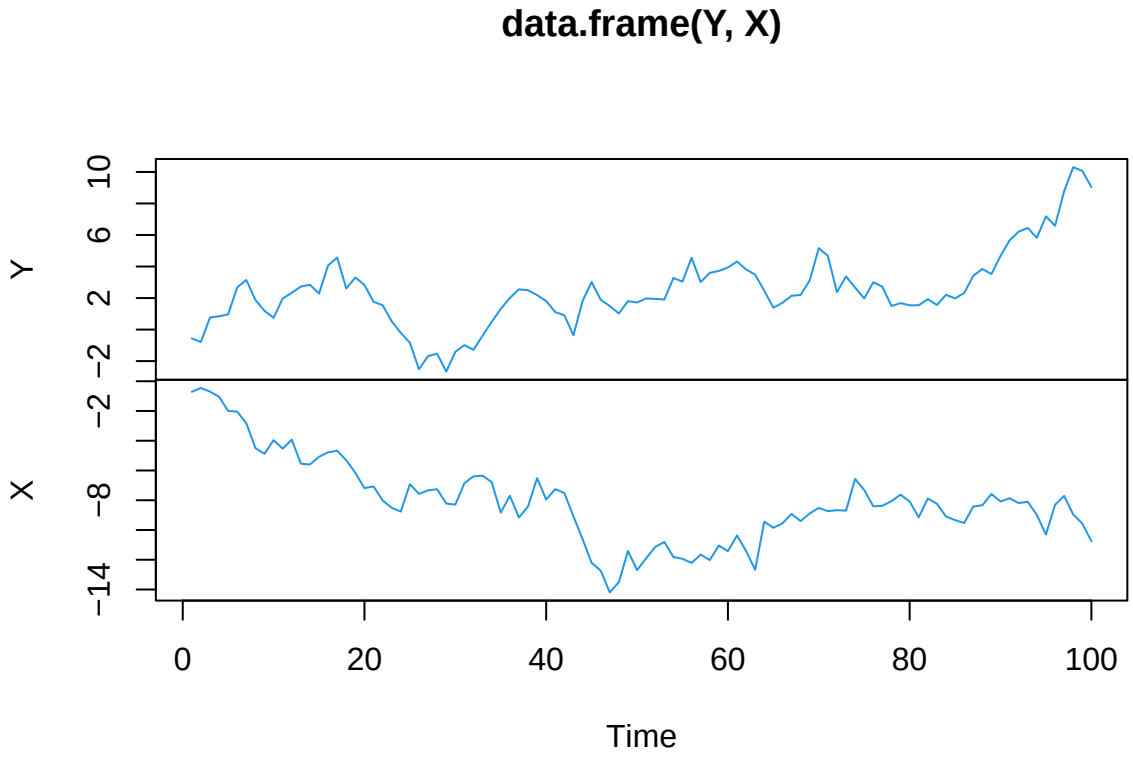
$$X_t = X_{t-1} + W_t^2$$

- W_t^i is Gaussian white noise with mean zero and standard deviation 1.

a) Randomly draw 100 observations from Y_t and X_T and estimate the following regression:

$$Y_t = \beta_0 + \beta_1 X_t + \epsilon_t$$

```
set.seed(123)
Y <- ts(cumsum(rnorm(100)))
X <- ts(cumsum(rnorm(100)))
plot.ts(data.frame(Y, X), col = 4)
```



Both X and Y look non-stationary, and it may seem that Y and X are correlated

```
model3 <-lm(Y~X)
round(summary(model3)$coefficients, 4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.8315     0.7009  1.1862   0.2384
## X            -0.2027     0.0817 -2.4822   0.0148
```

Here, we know that the actual value of β_1 is 0, but the OLS estimate is different from zero, and statistical tests indicate that β_1 is not zero. which is misleading and incorrect

So, if Y and X contain unit roots, then the OLS estimation of this regression can yield completely wrong results.

b) Estimate a model on variable differences. What do you notice?

$$\Delta Y_t = \beta_0 + \beta_1 \Delta X_t + \epsilon_t$$

```
model4 <-lm(diff(Y)~diff(X))
round(summary(model4)$coefficients, 4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.0918     0.0928  0.9889   0.3252
## diff(X)      -0.0512     0.0956 -0.5357   0.5934
```

If we estimate the ΔY_t and ΔX_t, the estimated coefficient of ΔX_t is not statistically significant anymore, which means there is no association between them

We should never run a regression of Y on X if the variables have unit roots unless they are cointegrated

If X_t and Y_t have unit roots and they are not cointegrated, the general advice is to estimate a model using differences ΔY_t and ΔX_t

Cointegration

- The one time we do not have to worry about the spurious regression problem occurs when X and Y are cointegrated.
- If Y and X have unit roots but some linear combination of them, $\gamma_1 Y_t + \gamma_2 Xt$, is (trend) stationary, then we say that Y and X are cointegrated.
- In other words: $Yt \sim I(1)$ and $X \sim I(1)$ are cointegrated if they share a common trend such that $\gamma_1 Y_t + \gamma_2 X_t \sim I(0)$.
- As mentioned above, if Y and X are cointegrated, then the spurious regression problem does not apply; consequently, we can run an OLS regression of Y on X and obtain valid results.

Cointegration test

1- Engle-Granger test

The null hypothesis of the Engle-Granger test is no cointegration, and we conclude cointegration is present only if we reject this hypothesis.

H_0 : No cointegration exists

H_1 : Cointegration exists

This cointegration test involves the following steps:

1- Carry out an (Augmented) Dickey-Fuller test on the null hypothesis that Y and X each have a unit root. If both time series are $I(0)$, standard regression analysis will be valid. If they are integrated into the same order (usually $I(1)$), proceed to the next step.

2- Run a regression of Y on X and save the residuals;

3- Carry out a unit root test on the residuals (without including a constant or a deterministic trend);

4- If the unit root hypothesis is rejected, then conclude that Y and X are cointegrated. However, if the unit root hypothesis is not rejected, then conclude that cointegration does not occur.

Thus, if Y_t and X_t are cointegrated, in $Y_t = \alpha + \beta X_t + \epsilon_t$, the error term is $I(0)$. If not, ϵ_t will be $I(1)$. Hence, one can test for the presence of a cointegration relationship by testing for a unit root in the OLS residuals e_t.

2- The Phillips-Ouliaris cointegration test

The Engle-Ganger test assumes that regression errors are independent with a common variance, which is rarely true in real life. The Philips-Ouliaris test improves the Engle-Ganger test by considering that the errors are not white noise.

H_0 : No cointegration exists

H_1 : Cointegration exists

3- Johansen test

Another improvement over the Engle-Granger test is the test developed by Johansen. This test can detect multiple cointegrating vectors.

Case Study: Cointegration and Pairs trading in finance

Basic Idea of Pairs Trading

- Recall that if two time series are cointegrated, they remain close to each other in the long term. In other words, the spread(OLS residual) between them $z_t = y_{1t} - \beta y_{2t}$ is mean-reverting.
- This mean-reverting property of the spread can be exploited for trading, and it is commonly referred to as "pairs trading" or "statistical arbitrage." The idea behind pairs trading is to:

Assume that spread $z_t = y_{1t} - \beta y_{2t}$ is stationary or mean-reverting with zero mean:

- if spread is low ($z_t < -s_0$), then stock 1 is undervalued and stock 2 overvalued:
  - buy the spread (i.e., buy stock 1 and short-sell stock 2)
  - unwind the positions when it reverts to zero after i time steps $z_{t+i} = 0$
- if spread is high ($z_t > s_0$), then stock 1 is overvalued and stock 2 undervalued:
  - short-sell the spread (i.e., short-sell stock 1 and buy stock 2)
  - unwind the positions when it reverts to zero after i time steps ($z_{t+i} = 0$)
- Here s_0 is some threshold like 3 standard deviations of the historical spread.

The profit from buying low and unwinding at zero is $z_{t+i} - z_t = s_0$.

Design of Pairs Trading

- In practice, pairs trading contains three main steps:

1- Pairs selection: identify stock pairs that could potentially be cointegrated.

2- Cointegration test: test whether the identified stock pairs are indeed cointegrated or not.

3- Trading strategy design: study the spread dynamics and design proper trading rules.

Netflix v.s Amazon

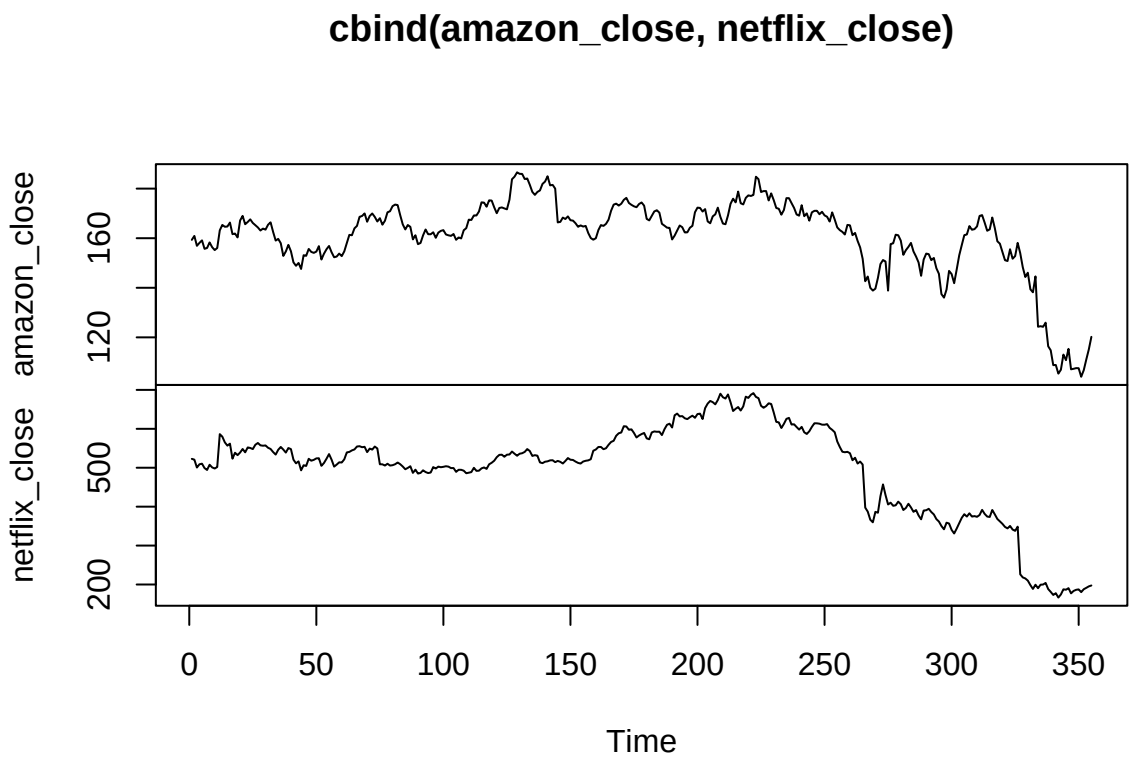
- Let us focus on the NFLX vs. AMZN pair, which are the Netflix and Amazon stocks.

a) Use the following code to get the stock prices using the quantmod package.

```
start <- as.Date("2021-01-01")
end <- as.Date("2022-06-01")
symbols <- c("AMZN", "NFLX")
# The auto.assign parameter allows for the returned object to be stored in a local variable rather than the R session's
amazon<- getSymbols("AMZN", src = "yahoo", from = start, to = end, auto.assign = FALSE, return.class= "ts")
netflix<- getSymbols("NFLX", src = "yahoo", from = start, to = end, auto.assign = FALSE, return.class= "ts")
```

b) Plot NFLX and AMZN closing prices, the ACF/PACF, and examine their stationary.

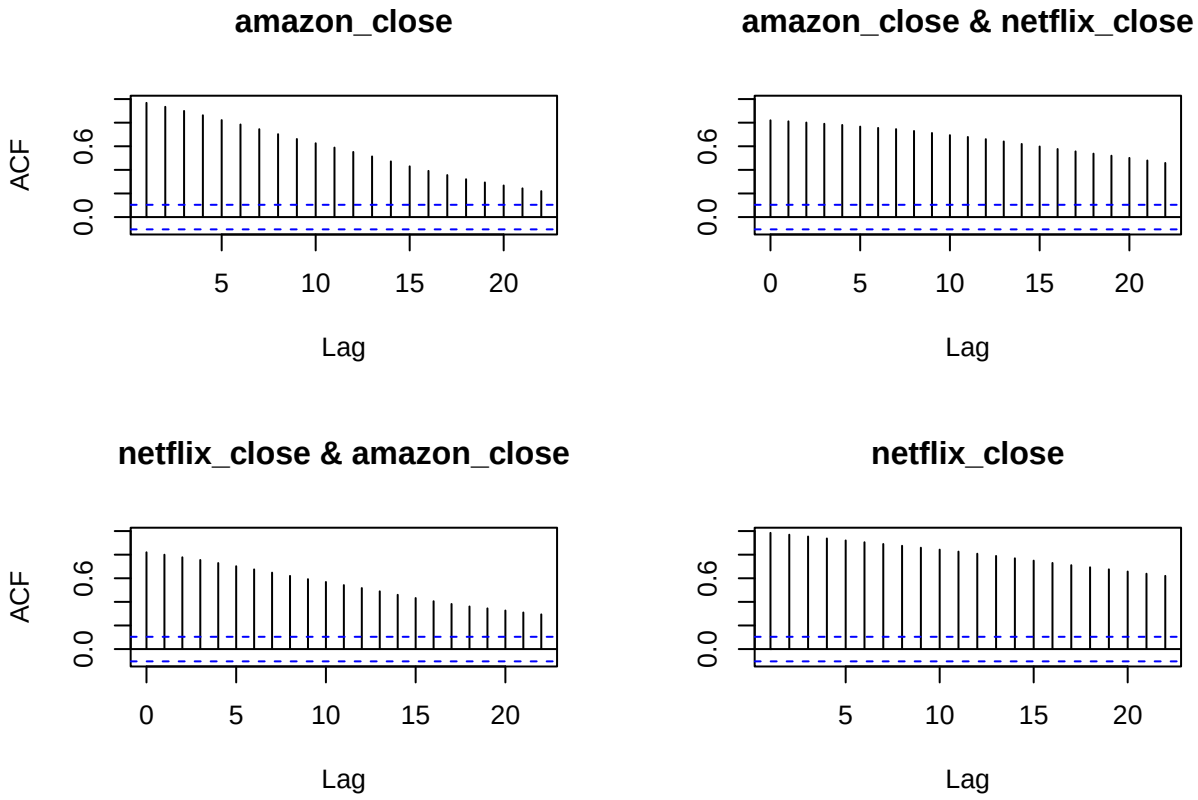
```
amazon_close <- amazon[,4]
netflix_close <- netflix[,4]
plot(cbind(amazon_close, netflix_close))
```



The time plots of Netflix and amazon show some non-stationarity, with an overall decrease in prices since 2021.

It seems that variance change over time; we'll use a log transformation

```
forecast::Acf(cbind(amazon_close, netflix_close))
```



The ACF decay very slowly, which suggests that these series could have a unit root.

Also, from the cross-correlation plots, we can see that the correlation between amazon and Netflix closing prices is positive, but we have to be careful. This positive correlation could be obtained because of 'hidden' stochastic trends in these variables. We have to check whether they are cointegrated or not.

c) Carry out unit root tests for NFLX and AMZN closing prices. Are they stationary? Are they both integrated of the same order?

```
tseries::adf.test(log(amazon_close), alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  log(amazon_close)
## Dickey-Fuller = -2.3473, Lag order = 7, p-value = 0.4303
## alternative hypothesis: stationary
```

```
tseries::adf.test(log(netflix_close), alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  log(netflix_close)
## Dickey-Fuller = -0.45515, Lag order = 7, p-value = 0.9836
## alternative hypothesis: stationary
```

```
tseries::adf.test(diff(log(amazon_close)), alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(log(amazon_close))
## Dickey-Fuller = -5.9655, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
tseries::adf.test(diff(log(netflix_close)), alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(log(netflix_close))
## Dickey-Fuller = -6.6909, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

For both log of Netflix and amazon closing prices, the p-value is much higher than the 5%, so we fail to reject the null hypothesis of the non-stationarity.

For their differences, both p-values are less than 5%, and we reject the null hypothesis of a unit root.

So both logs of Amazon and Netflix closing prices have a unit root and are I(1), and we can proceed to the next step.

d) Carry out the Engle-Granger test for cointegration

```
coint_reg <- lm(log(netflix_close) ~ log(amazon_close))
coef(summary(coint_reg))
```

```
##                    Estimate Std. Error   t value      Pr(>|t|)
## (Intercept)       -6.779378 0.39766372 -17.04802  5.705163e-48
## log(amazon_close)  2.550723 0.07833995  32.55967 2.315879e-108
```

```
coint_res <- ts(coint_reg$res)
pp.test(coint_res)
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  coint_res
## Dickey-Fuller Z(alpha) = -22.031, Truncation lag parameter = 5, p-value
## = 0.04491
## alternative hypothesis: stationary
```

```
#Phillips-Ouliaris cointegration test
po.test(cbind(log(netflix_close),log(amazon_close)))
```

```
##
##  Phillips-Ouliaris Cointegration Test
##
## data:  cbind(log(netflix_close), log(amazon_close))
## Phillips-Ouliaris demeaned = -17.893, Truncation lag parameter = 3,
## p-value = 0.0852
```

To test for cointegration using the Engle-Granger test, we first run a log of Netflix closing price on the log of amazon closing price and then carry out a unit root test on the residuals.

Based on the Phillips-Perron Unit Root Test, the p-value is less than 5%, and we reject the null hypothesis of unit root for the residuals. so the Engle-Granger test provides evidence that the log of Netflix and amazon closing prices are cointegrated since the residuals are stationary.

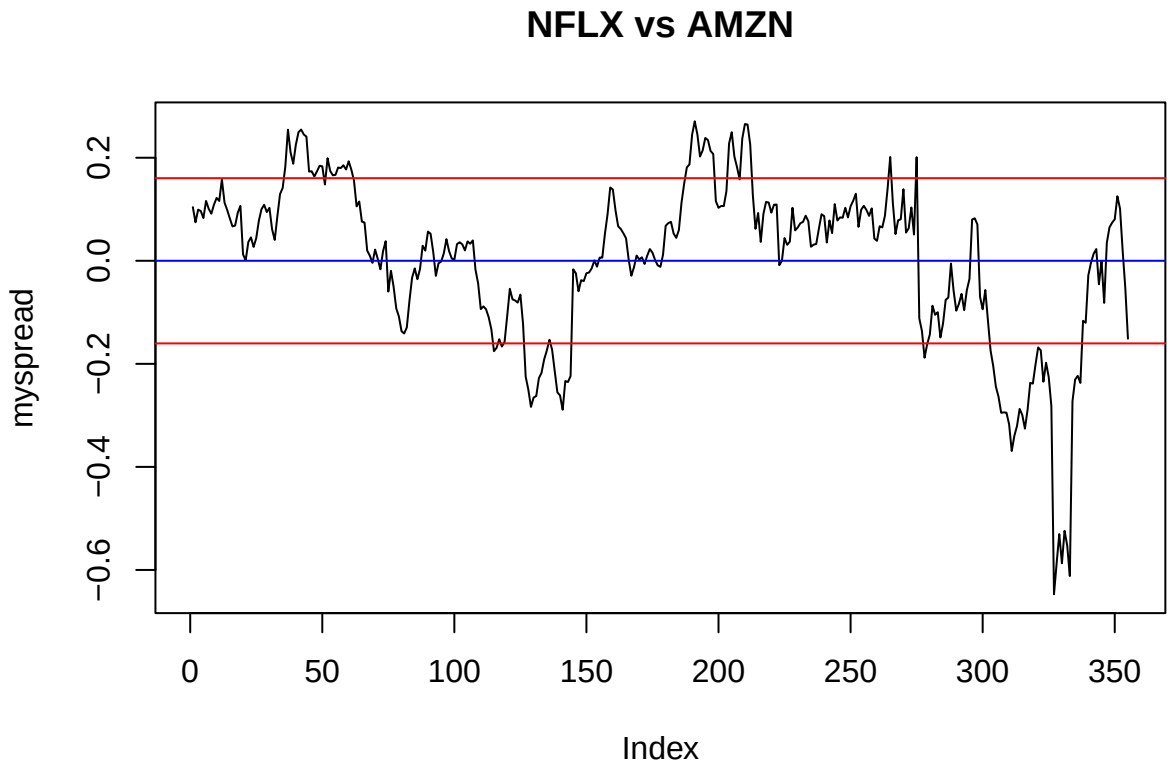
We can also use a more robust cointegration test or Phillips-Ouliaris cointegration test. Based on this test, we fail to reject the null hypothesis that no cointegration exists at 5%, but we reject it at the 10% significance level.

e) Plot the spread(residual) and discuss a possible trading strategy

```
myspread<- coint_reg$residuals
plot(myspread, main = "NFLX vs AMZN", type = "l")
sd(myspread)
```

```
## [1] 0.1603209
```

```
abline(a=mean(myspread), b=0,  col= "blue")
abline(a=sd(myspread), b=0,  col= "red")
abline(a=-sd(myspread), b=0, col = "red")
```



By Analyzing the spread of residuals, we can define trading signals for when to open a position and close. We can use the standard

deviations of the spread of residual as a threshold.

For example, let's consider that our trading signals are based on one standard deviation of residuals, so here 0.16 and -0.16, respectively.

so our strategy is as the following:

When the spread is above 0.16, this means that Netflix is overvalued compared to amazon, and we are going to sell the NFLX and we buy AMZN

When the spread is below -0.16, this means that Netflix is undervalued compared to amazon, and we buy NFLX and sell AMZN

And Whenever the spread converges again to 0, we close our position.

Vector Autoregressive (VAR) model

Introduction: Granger Causality

- To motivate why the VAR model is essential, we begin by discussing **Granger causality**.
- VARs can be used to investigate Granger causality
- What is Granger causality?
- In the following regression, we call Y_t the dependent variable and X_t the explanatory variable. In many cases, because the latter 'explained' the former, it was reasonable to talk about X 'causing' Y.

$$Y_t = \beta_0 + \beta_1 X_t$$

- However, the causality could run in either direction - or both! Hence, when using the word 'cause' with regression or correlation results, a great deal of caution has to be taken, and common sense has to be used.

In the time-series data, we can make slightly stronger statements about causality simply by exploiting the fact that time does not run backward!

That is, if event A happens before event B, then it is possible that A is causing B. However, it is not possible that B is causing A. In other words, events in the past can cause events to happen today. Future events cannot.

- These intuitive ideas can be investigated through regression models incorporating the notion of Granger or regressive causality. The basic idea is that a variable X Granger causes Y if past values of X can help explain Y.
- Of course, if Granger causality holds, this does not guarantee that X causes Y. This is why we say 'Granger causality rather than just 'causality.' Nevertheless, if past values of X have explanatory power for current Y values, it at least suggests that X might be causing Y. Granger causality is only relevant with time-series variables.
- For example, consider Granger causality between two stationary variables (X and Y). Since X and Y are stationary, the following ARDL(q,p) model is appropriate.

$$Y_t = \alpha + \phi_1 Y_{t-1} + ... + \phi_p Y_{t-p} + \beta_1 X_{t-1} + ... + \beta_q X_{t-q} + \epsilon_t$$

- **X does not Granger cause Y if all** $\beta_i = 0$**.**

- In many cases, it is not apparent which way causality should run. In such cases, when causality may be in either direction, we must check for it if Y and X are the two variables under study; in addition to running a regression of Y on lags of itself and lags of X (as above), you should also run a regression of X on lags of itself and lags of Y.
- In other words, we should work with two separate equations: one with Y as the dependent variable and one with X as the dependent variable. These two equations comprise a VAR. A VAR is an extension of the autoregressive (AR) model to the case where there is more than one variable under study.

$$Y_t = \alpha_1 + \phi_{11}Y_{t-1} + ... + \phi_{1p}Y_{t-p} + \beta_{11}X_{t-1} + ... + \beta_{1q}X_{t-q} + \epsilon_{1t}$$

$$X_t = \alpha_2 + \phi_{21}Y_{t-1} + ... + \phi_{2p}Y_{t-p} + \beta_{21}X_{t-1} + ... + \beta_{2q}X_{t-q} + \epsilon_{2t}$$

- The first of these equations tests whether X Granger causes Y; the second, whether Y Granger causes X. Note that now the coefficients have subscripts indicating which equation they are in. The errors now have subscripts to denote that they will differ in the two equations.
- The VAR model can be extended to the case of many variables, and we could include more than two variables in a VAR model.
- The variable in VAR should be stationary. If the original variables have unit roots, we assume that differences have been taken such that the model includes the changes in the original variables (which do not have unit roots).
- If the original variables have unit roots but are cointegrated, then we should work with a vector error correction model (VECM) involving these variables, which is beyond the scope of this course.

The bottom line - if X Granger causes Y, this does not mean that X causes Y; it only means that X improves Y's predictability (i.e., reduces residuals of the model).

VAR: Estimation

Building a VAR model involves three steps:

1- Use some information criterion (AIC, etc.) to identify the order.

2- Estimate the specified model using the least-squares method and, if necessary, re-estimate the model by removing statistically insignificant parameters.

3- Use the Portmanteau test statistic of the residuals to check the adequacy of a fitted model (this is a multivariate analog of the Ljung-Box Q-stat in an ARIMA model and is to test for autocorrelation and cross-correlation in residuals). If the fitted model is adequate, then it can be used to obtain forecasts.

Analytical Exercise

- Consider the following bivariate VAR

$$Y_t = 0.3Y_{t-1} + 0.8X_{t-1} + \epsilon_{1t}$$
$$X_t = 0.9Y_{t-1} + 0.4X_{t-1} + \epsilon_{2t}$$

- Is this system covariance stationary?

$$\begin{bmatrix} y_{1t} \\ y_{2t} \end{bmatrix} = \begin{bmatrix} 0.3 & 0.8 \\ 0.9 & 0.4 \end{bmatrix} \cdot \begin{bmatrix} y_{1t-1} \\ y_{2t-1} \end{bmatrix} + \begin{bmatrix} \epsilon_{1t} \\ \epsilon_{2t} \end{bmatrix} \tag{1}$$

This VAR system is stationary if the eigenvalues λ of the matrix of coefficients are less than 1. If λ is the eigenvalue of this matrix, then the determinant of this matrix is:

$$\begin{vmatrix} \lambda - 0.3 & -0.8 \\ -0.9 & \lambda - 0.4 \end{vmatrix} = 0 \tag{2}$$

$$\lambda^2 - 0.7\lambda - 0.6 = 0$$

$$\lambda_1 = \frac{0.7 + \sqrt{(0.7)^2 + 4 \cdot 0.6}}{2} = 1.2$$

$$\lambda_2 = \frac{0.7 - \sqrt{(0.7)^2 + 4 \cdot 0.6}}{2} = -0.5$$

Since the $\lambda_1 > 1$, this VAR system is non-stationary.

Empirical Exercise: Bitcoin price and public attention

In this exercise, we will examine the linkage between the bitcoin prices and public attention, proxied by Google Trends data. Nowadays, many investors gather market information mainly through the internet, and Google searches signal investors' attention. Google Trends allows analysts to see how often specific terms are searched.

a) Use the code below to pull the following time series from the Quandl and Google trends API.

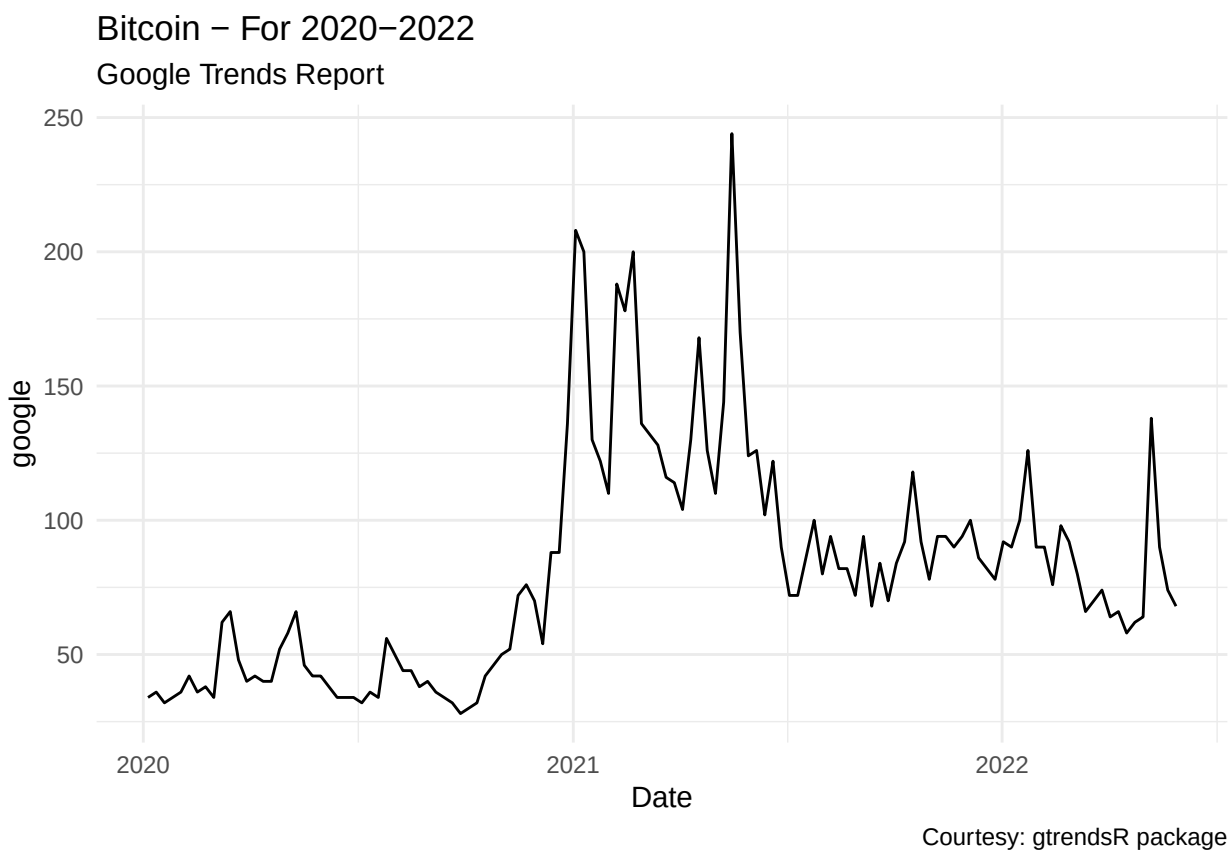
1- Weakly Bitcoin price from Quandl API since 01/01/2020.

2- Weakly Google search volume for four main words associated with bitcoin, including "Bitcoin," "bitcoin," "BTC," and "btc" from Google trend API since 01/01/2020

```
### qundl
Quandl.api_key("mbGCKg2ifLUx_DmxzbGv")
bitcoin_weekly = Quandl("BCHAIN/MKPRU", start_date="2020-01-01", collapse = "weekly")

# The gtrends default method performs a Google Trends query for the 'query' argument and session
# define search keyword
keyword <- c("Bitcoin", "bitcoin","BTC", "btc")
# define the location
geo <- "all"
#define the channels "web","news", "image", "youtube"
grop = c("web")
#define the time window
time <- "all"
#extract trend
google <- gtrends(keyword, geo = "", grop, time = "2020-01-01 2022-06-01")

df <- data.frame(google[1]) %>%
  rename (Date = interest_over_time.date) %>%
  mutate(Date = as.Date(Date)) %>%
  group_by(Date) %>%
  summarise(google = sum(as.double(interest_over_time.hits)))
## join google trend and bitcoin data
df <- df %>%
  left_join(bitcoin_weekly, by = "Date") %>%
  rename (bitcoin = Value)

## Import data from csv file
#df <- read_csv("./data/google_bitcoin.csv")
```

b) Plot the time series of bitcoin prices and google trends. Do they look stationary?

```
df %>% ggplot() +
  geom_line(aes(x= Date, y= google)) +
  theme_minimal() +
  labs(title = "Bitcoin - For 2020-2022",
       subtitle = "Google Trends Report",
       caption = "Courtesy: gtrendsR package")
```



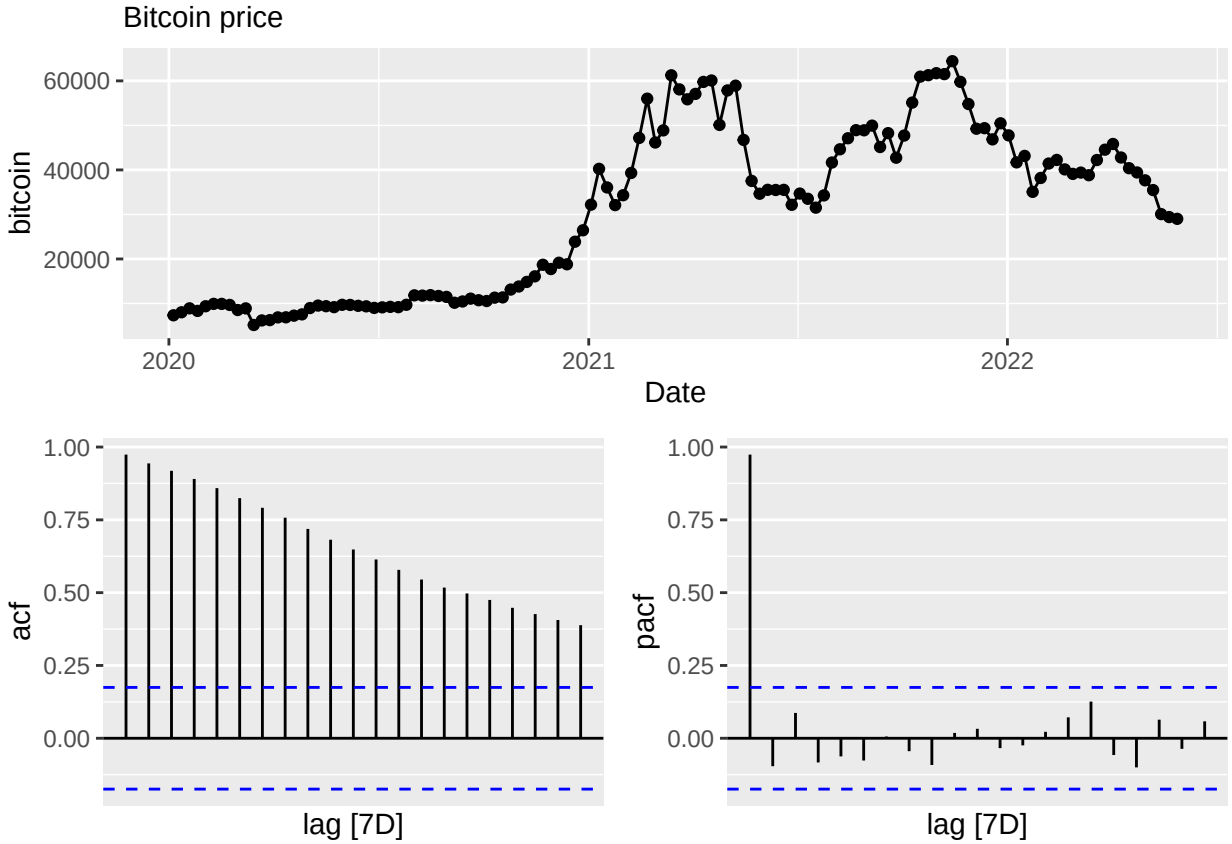

```
df %>% ggplot() +
  geom_line(aes(x= Date, y= bitcoin)) +
  theme_minimal() +
```

```
labs(title = "Bitcoin prices - For 2020-2022")
```

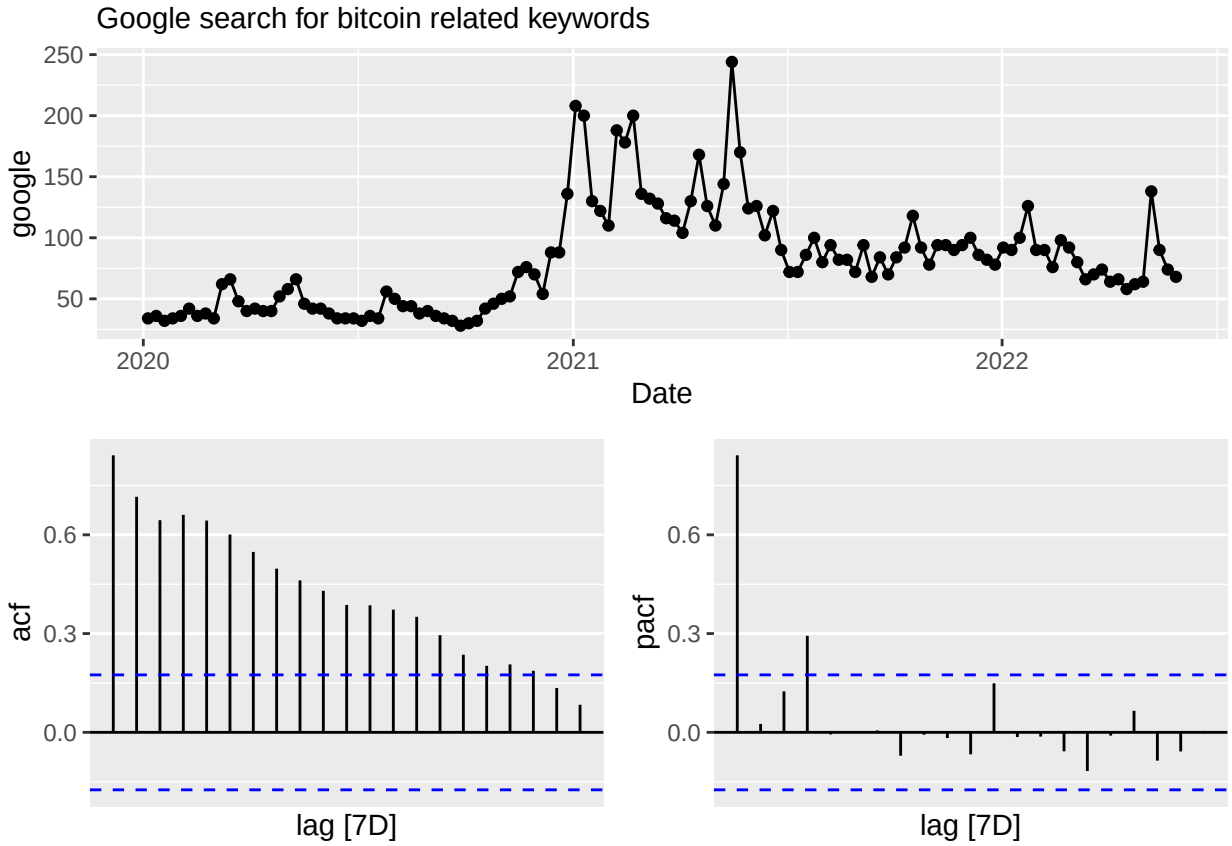
The time plot of both bitcoin price and google trend shows some non-stationarity, with an overall increase since 2020

c) Plot ACF/PACF the Perform the unit root test on the bitcoin prices and google trends and report the results. Do you reject the null of unit root for them?

```
df %>% as_tsibble(index=Date)%>% gg_tsdisplay(bitcoin, plot_type="partial") +labs(subtitle = "Bitcoin price")
```




```
df %>% as_tsibble(index=Date)%>% gg_tsdisplay(google, plot_type="partial") +labs(subtitle = "Google search for bitcoin related keywords")
```



The ACF of both series decay very slowly, and The first lag of their PACF is 1, which suggests that these series should have a unit root.

```
df %>%
  as_tsibble(index=Date)%>%
  features(bitcoin, unitroot_kpss)
```

```
## # A tibble: 1 x 2
##   kpss_stat kpss_pvalue
##       <dbl>       <dbl>
## 1      1.84        0.01
```

```
df %>%
  as_tsibble(index=Date)%>%
  features(google, unitroot_kpss)
```

```
## # A tibble: 1 x 2
##   kpss_stat kpss_pvalue
##       <dbl>       <dbl>
## 1     0.836        0.01
```

The KPSS unit root tests results suggest that we reject the null of stationarity for the bitcoin prices and google trend.

d) Now calculate the first difference for the log of bitcoin prices and google search volume. Are they stationary? Test using the unit root tests.

```
df %>%
  as_tsibble(index=Date)%>%
  mutate(diff_bitcoin = difference(bitcoin)) %>%
  features(diff_bitcoin, unitroot_kpss)
```

```
## # A tibble: 1 x 2
##   kpss_stat kpss_pvalue
##       <dbl>       <dbl>
## 1     0.193         0.1
```

```
df %>%
  as_tsibble(index=Date)%>%
  mutate(diff_google = difference(google)) %>%
  features(diff_google, unitroot_kpss)
```

```
## # A tibble: 1 x 2
##   kpss_stat kpss_pvalue
##       <dbl>       <dbl>
## 1    0.0816         0.1
```

The p-values of the KPSS unit root test are larger than 0.05 for both first difference series, and we fail to reject the null hypothesis of stationarity for the first difference of bitcoin prices and google search volume.

So we can use these two differenced series to estimate the VAR model as they are now stationary.

e) Determine the lag length of the VAR using the information criteria. Estimate the VAR and comment on the fit.

```
## To delete NA from first row filter it
df1 <- df %>%
  mutate(
    diff_bitcoin = difference(bitcoin),
    diff_google = difference(google)
    ) %>%
  filter(Date >= "2020-01-12") %>%
   dplyr::select(diff_bitcoin,diff_google)

VARselect(df1, lag.max = 4, type="none")
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      3      3      1      3
##
## $criteria
##                  1            2            3            4
## AIC(n) 2.275427e+01 2.277213e+01 2.265907e+01 2.271581e+01
## HQ(n)  2.279181e+01 2.284720e+01 2.277168e+01 2.286596e+01
## SC(n)  2.284670e+01 2.295697e+01 2.293634e+01 2.308550e+01
## FPE(n) 7.621803e+09 7.759442e+09 6.930777e+09 7.337004e+09
```

We use different model selection criteria to choose the optimal number of lags. AIC, HQ, and FPE suggest three lags while SC (akin to BIC in the univariate case) suggests 1 lag. Here we estimate VAR(3)

```
var_diff = vars::VAR(df1, p = 3, type = "none")
summary(var_diff)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: diff_bitcoin, diff_google
## Deterministic variables: none
## Sample size: 122
## Log Likelihood: -1715.391
## Roots of the characteristic polynomial:
## 0.7281 0.7281 0.6804 0.5697 0.5697 0.4982
## Call:
## vars::VAR(y = df1, p = 3, type = "none")
```

```
##
##
## Estimation results for equation diff_bitcoin:
## ============================================
## diff_bitcoin = diff_bitcoin.l1 + diff_google.l1 + diff_bitcoin.l2 + diff_google.l2 + diff_bitcoin.l3 + diff_google.l3
##
##                  Estimate Std. Error t value Pr(>|t|)
## diff_bitcoin.l1   0.17791    0.09305   1.912   0.0583 .
## diff_google.l1  -22.06228   14.58063  -1.513   0.1330
## diff_bitcoin.l2  -0.13746    0.09461  -1.453   0.1490
## diff_google.l2    0.08827   14.58554   0.006   0.9952
## diff_bitcoin.l3   0.12337    0.09477   1.302   0.1956
## diff_google.l3  -19.59633   14.78480  -1.325   0.1876
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 3651 on 116 degrees of freedom
## Multiple R-Squared: 0.0686,  Adjusted R-squared: 0.02042
## F-statistic: 1.424 on 6 and 116 DF,  p-value: 0.2113
##
##
## Estimation results for equation diff_google:
## ===========================================
## diff_google = diff_bitcoin.l1 + diff_google.l1 + diff_bitcoin.l2 + diff_google.l2 + diff_bitcoin.l3 + diff_google.l3
##
##                   Estimate Std. Error t value Pr(>|t|)
## diff_bitcoin.l1  0.0011531  0.0005570   2.070 0.040631 *
## diff_google.l1  -0.2247966  0.0872734  -2.576 0.011258 *
## diff_bitcoin.l2  0.0001103  0.0005663   0.195 0.845953
## diff_google.l2  -0.2008020  0.0873028  -2.300 0.023233 *
## diff_bitcoin.l3 -0.0009343  0.0005673  -1.647 0.102268
## diff_google.l3  -0.3243877  0.0884955  -3.666 0.000374 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 21.85 on 116 degrees of freedom
## Multiple R-Squared: 0.2132,  Adjusted R-squared: 0.1725
```

```
## F-statistic:  5.24 on 6 and 116 DF,  p-value: 8.295e-05
##
##
##
## Covariance matrix of residuals:
##              diff_bitcoin diff_google
## diff_bitcoin     13299678     13010.2
## diff_google         13010       476.9
##
## Correlation matrix of residuals:
##              diff_bitcoin diff_google
## diff_bitcoin       1.0000      0.1634
## diff_google        0.1634      1.0000
```

In the first equation, none of the coefficients are statistically significant in 5% for change in bitcoin regression. R-squared is small, and only 6.9 percent of the variations of change in bitcoin prices can be explained by the lagged change in google search volume and lagged change in bitcoin prices.

In the second equation, for change in google trend, The estimated VAR model suggests that the past values of change in bitcoin prices have explanatory power for current values of change in google trend.

However, we find that only lag one and three of the bitcoin prices is significant at a 10% percent significance level.

So apparently, large fluctuations in bitcoin prices lead to higher attention to the bitcoin and higher google search volume.

Also, 21 percent of the variations of change in google search volume can be explained by the lag of change in google search volume and the lag of change in bitcoin prices.

So, based on these results, we can conclude that higher bitcoin prices could predict higher google search volume, but not the other way around.

f) Test for Granger-causality. Does google trend Granger-cause bitcoin prices? Does bitcoin prices Granger-cause google trend?

```
vars::causality(var_diff,cause="diff_google")$Granger
```

```
##
##  Granger causality H0: diff_google do not Granger-cause diff_bitcoin
##
## data:  VAR object var_diff
## F-Test = 1.1773, df1 = 3, df2 = 232, p-value = 0.3191
```

```
vars::causality(var_diff,cause="diff_bitcoin")$Granger
```

```
##
##  Granger causality H0: diff_bitcoin do not Granger-cause diff_google
##
## data:  VAR object var_diff
## F-Test = 2.6651, df1 = 3, df2 = 232, p-value = 0.04861
```

The p-value for the test of change in google search volume(diff_goole) not Granger causing a change in bitcoin prices(diff_bitoin) is 0.3087. which suggests that we fail to reject the null hypothesis that diff_googl does not Granger-cause diff_bitcoin at all significance levels.

However, the p-value for the test of change in bitcoin prices(diff_bitcoin) is not Granger causing a change in google search volume (diff_google) is less than 5%, and we reject the null hypothesis of change in bitcoin prices(diff_bitcoin), not Granger causing google search volume(diff_google).

These results suggest that google search volume has no predictive power for change in bitcoin prices, but bitcoin price has some predictive power for google search volume.

Note that evidence in favor of Granger causality does not necessarily imply that bitcoin prices have significant out-of-sample predictive power for google search volume.

g) Do diagnostic checking of the VAR model.

```
## check stability
roots(var_diff)
```

```
## [1] 0.7281204 0.7281204 0.6803743 0.5697397 0.5697397 0.4981735
```

```
# Test of no serial correlation:
var_diff_test<- serial.test(var_diff, lags.pt = 12)
var_diff_test
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var_diff
## Chi-squared = 36.015, df = 36, p-value = 0.4679
```

First, we need to check if the estimated VAR(3) is a stable process, and we will need to check if the eigenvalues of the companion matrix are all less than one.

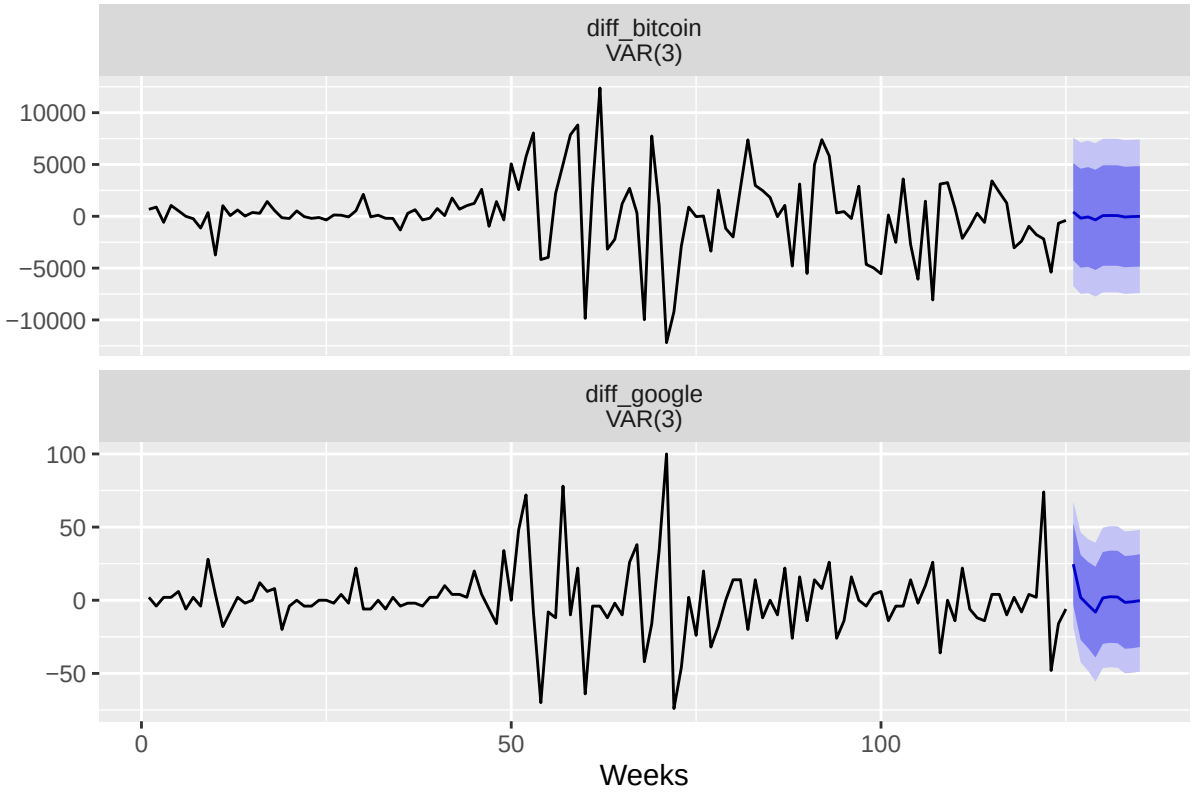
Here, since here all eigenvalues are less than 1, VAR(3) is a stable process.

Then, we need to check for tests for autocorrelation in residuals. The serial.test() computes the multivariate Portmanteau- for serial correlation.

Based on the test results, the null hypothesis of no autocorrelation is not rejected since the p-value is 0.4868.

h) We finally conduct a 3-step ahead forecast:

```
var_diff = vars::VAR(as.ts(df1), p = 3, type = "none")
forecast(var_diff) %>%
  autoplot() +
  xlab("Weeks")
```



Both changes in log of bitcoin prices and google trend search revert to their means, which is onsistent with stationarity.

Reminders

- Before the next live session:
  1. Complete and turn in the Lab-2
  2. Complete all videos and reading for unit 11