Assignment 4

Jaewoo Cho

FPP3 9.11: 2 and 6

2. A classic example of a non-stationary series are stock prices.

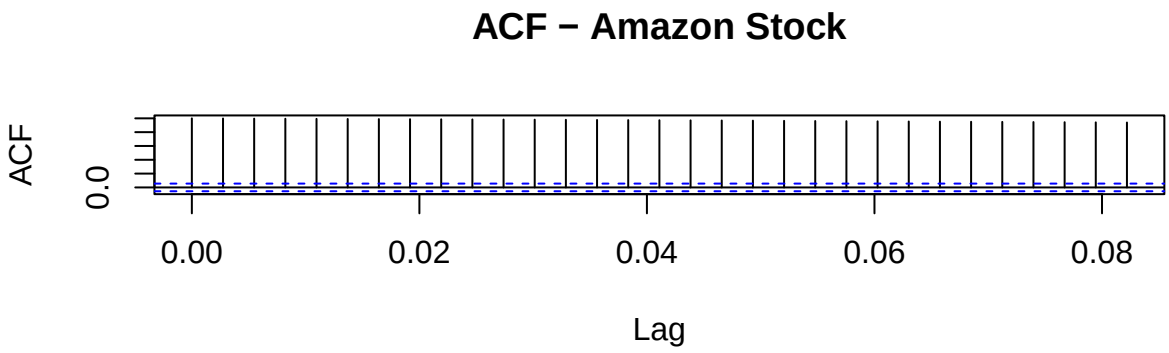
```
stock <- gafa_stock

amazon_stock <- stock %>% filter(Symbol == "AMZN")
amazon_stock
```

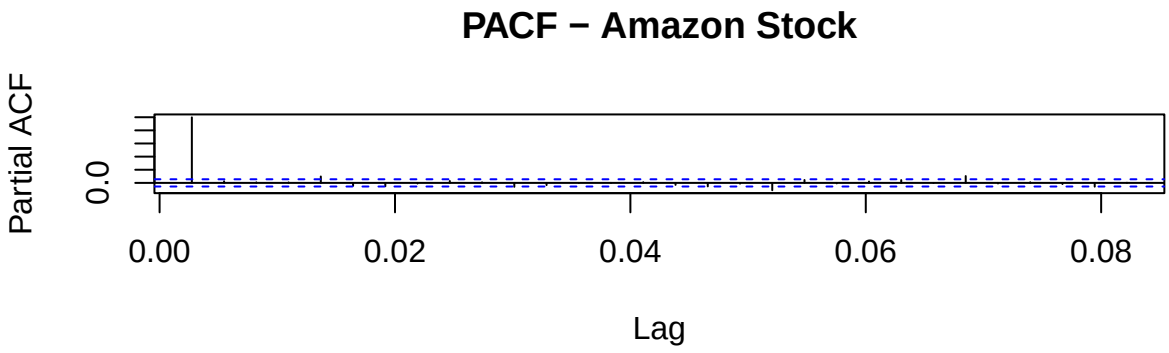
```
# A tsibble: 1,258 x 8 [!]
# Key:       Symbol [1]
   Symbol Date        Open  High   Low Close Adj_Close  Volume
   <chr>  <date>     <dbl> <dbl> <dbl> <dbl>     <dbl>   <dbl>
 1 AMZN   2014-01-02  399.  399.  394.  398.      398. 2137800
 2 AMZN   2014-01-03  398.  403.  396.  396.      396. 2210200
 3 AMZN   2014-01-06  396.  397   388.  394.      394. 3170600
 4 AMZN   2014-01-07  395.  398.  394.  398.      398. 1916000
 5 AMZN   2014-01-08  398.  403   396.  402.      402. 2316500
 6 AMZN   2014-01-09  404.  407.  398.  401.      401. 2103000
 7 AMZN   2014-01-10  403.  404.  394.  398.      398. 2679500
 8 AMZN   2014-01-13  398.  400.  388.  391.      391. 2844900
 9 AMZN   2014-01-14  392.  399.  391.  398.      398. 2340100
10 AMZN   2014-01-15  399.  399.  393.  396.      396. 2678300
# ... with 1,248 more rows
```

a. Plot the daily closing prices for Amazon stock (contained in `gafa_stock`), along with the ACF and PACF.

```
# Plot time series, ACF, and PACF
# ACF (AutoCorrelation Function) and PACF (Partial AutoCorrelation Function)
## Use Amazon's stock symbol ("AMZN")
amazon_stock <- gafa_stock %>%
  filter(Symbol == "AMZN") %>%
  select(Date, Close)
amazon_ts <- ts(amazon_stock$Close, frequency = 365)

ggplot(amazon_stock, aes(x = Date, y = Close)) +
  geom_line() +
  labs(title = "Amazon Daily Closing Prices", x = "Date", y = "Close Price")
```


```
par(mfrow = c(2, 1))  # Set up a 2x1 grid for plotting ACF and PACF

# Plot ACF
acf(amazon_ts, main = "ACF - Amazon Stock")

# Plot PACF
pacf(amazon_ts, main = "PACF - Amazon Stock")
```





b. Explain how each plot shows that the series is non-stationary.

Answer "Explain how each plot shows that the series is non-stationary." - Non-stationarity in a time series generally implies that the statistical properties of the series change over time, making it difficult to model and predict. The daily closing prices plot shows the actual stock prices over time that increase as time passes with a increasing trend. Non-stationarity can often be observed visually when the series exhibits trends, seasonality, or irregular fluctuations. - The ACF plot helps us understand the autocorrelation between each observation and its lagged values. Non-stationary time series often exhibit slow decay in autocorrelation.A straight line at or near zero on the ACF plot suggests that there is no systematic relationship between the observations at different lags. Each observation is independent of the others, and there is no serial correlation. - The PACF plot helps identify the direct relationship between an observation and its lagged values, while removing the indirect effects of shorter lags. In a non-stationary series, the PACF plot may show spikes at multiple lags, indicating that past values have a direct influence on current values, even when accounting for shorter lags.A straight line at or near zero on the PACF plot indicates that there are no direct relationships between the current observation and its lagged values. This means that past observations do not have a significant impact on the current observation.

c. Perform KPSS unit root test and interpret the test statistic (i.e., *p*-value).

```
# KPSS test
# Need differencing?
amazon_stock %>%
```

```
  features(Close, unitroot_kpss) %>%
  as.matrix()
```

```
     kpss_stat kpss_pvalue
[1,]  14.03978        0.01
```

Answer "Interpret the test statistic (i.e., p-value)" The KPSS (Kwiatkowski-Phillips-Schmidt-Shin) unit root test is used to determine whether a time series is stationary around a deterministic trend or not.The test statistic measures the deviation of the time series from stationarity. It is a numerical value calculated by the KPSS test.The test statistic is 14.03978, which indicates that there is a significant deviation from strict stationarity around a deterministic trend. The p-value is approximately 0.01, which is less than the typical significance level of 0.05. This low p-value suggests that the deviation from stationarity is statistically significant, supporting the rejection of the null hypothesis of stationarity. Therefore, based on both the test statistic and the p-value, the conclusion is that the Amazon stock price series (Close) is not stationary around a deterministic trend, and differencing or other transformations may be necessary for further analysis.

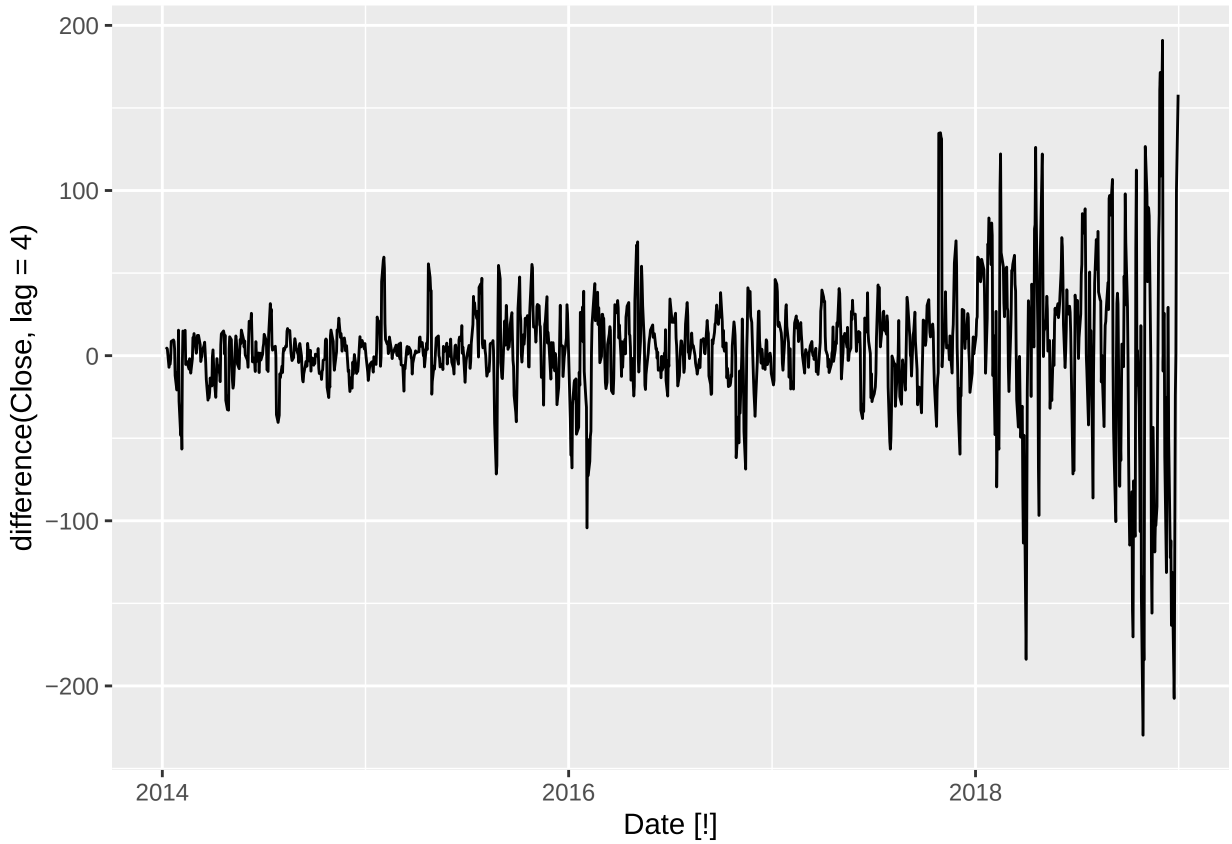
d. Perform unit root tests to determine whether seasonal differencing or differencing are necessary.

```
# Number of seasonal differencings

amazon_stock %>%
  features(Close, unitroot_nsdiffs) %>%
  as.matrix()
```

Seasonal differencing should be performed before differencing (best pratice).

```
     nsdiffs
[1,]       0
```

```
# Plot with seasonal differrencing
amazon_stock %>%
  autoplot(difference(Close, lag = 4))
```



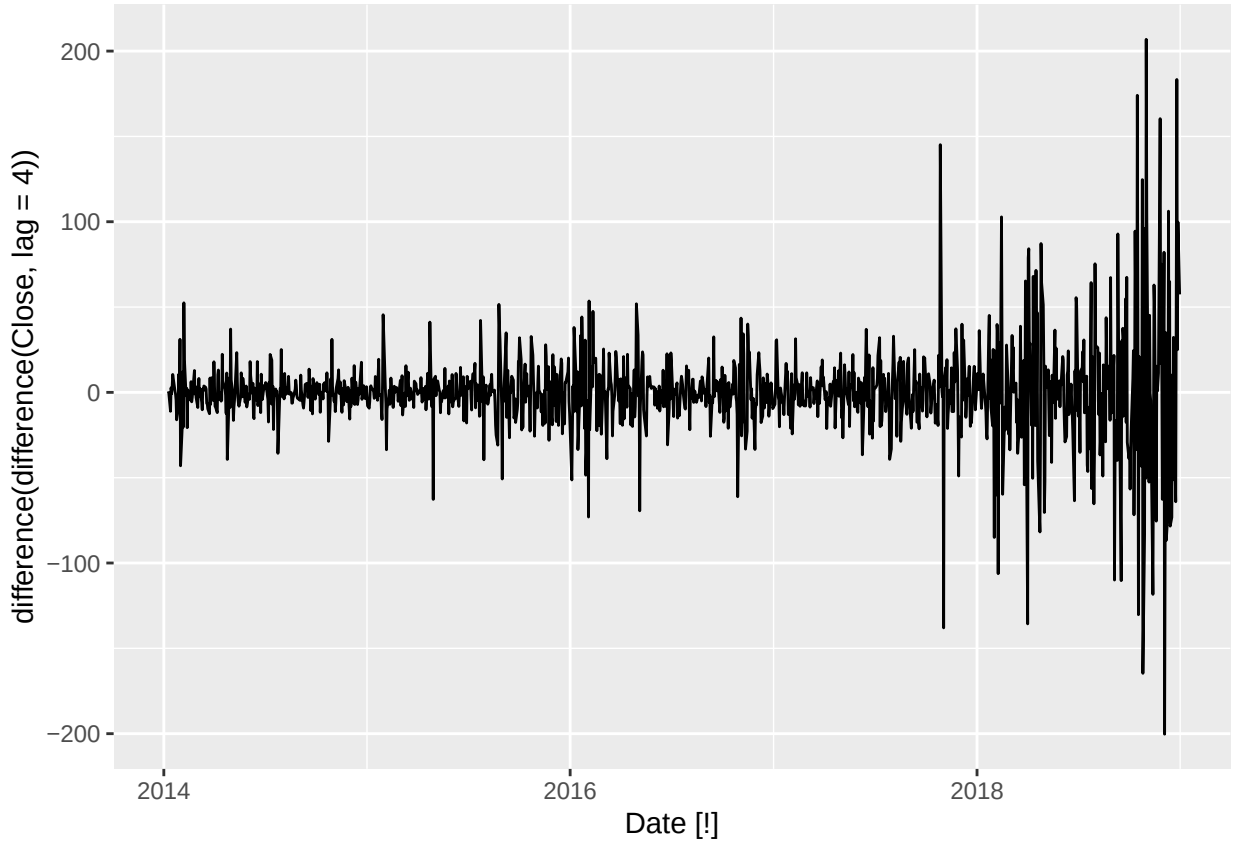

```
# Number of differencings
amazon_stock %>%
  features(difference(Close, lag = 4), unitroot_kpss) %>%
  as.matrix()
```

```
     kpss_stat kpss_pvalue
[1,] 0.1815849         0.1
```

```
#%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

# Number of differencings
amazon_stock %>%
  features(difference(Close, lag = 4), unitroot_ndiffs) %>%
  as.matrix()
```

```
     ndiffs
[1,]      0
```

```
#%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

# Plot with seasonal + one differencing
amazon_stock %>%
  autoplot(difference(difference(Close, lag = 4)))
```




```
# KPSS
amazon_stock %>%
  features(difference(difference(Close, lag = 4)), unitroot_kpss) %>%
  as.matrix()
```

```
      kpss_stat kpss_pvalue
[1,] 0.01742806         0.1
```

```
# Perform unit root tests and determine differencing requirements
ns_diffs_result <- amazon_stock %>%
  features(Close, unitroot_nsdiffs) %>%
  as.matrix()

reg_diffs_result <- amazon_stock %>%
  features(difference(Close, lag = 4), unitroot_kpss) %>%
  as.matrix()

# Interpret the results
if (ns_diffs_result > 0 && reg_diffs_result > 0) {
  cat("Both seasonal and regular differencing are necessary.\n")
} else if (ns_diffs_result > 0) {
  cat("Seasonal differencing is necessary, but regular differencing may not be needed.\n")
} else if (reg_diffs_result > 0) {
  cat("Regular (non-seasonal) differencing is necessary, but seasonal differencing may not be needed.\n
} else {
```

```
  cat("No differencing may be required as the time series may already be stationary.\n")
}
```

```
Regular (non-seasonal) differencing is necessary, but seasonal differencing may not be needed.
```

```
# Additional step: KPSS test on differenced series
kpss_result <- amazon_stock %>%
  features(difference(difference(Close, lag = 4)), unitroot_kpss) %>%
  as.matrix()

if (kpss_result < 0.05) {
  cat("The combination of seasonal and regular differencing has made the time series stationary based o
} else {
  cat("Further analysis may be needed to ensure stationarity.\n")
}
```

```
The combination of seasonal and regular differencing has made the time series stationary based on the K
```

> Answer "Are seasonal differencing or differencing necessary?" Regular (non-seasonal) differencing is necessary, but seasonal differencing may not be needed. The combination of seasonal and regular differencing has made the time series stationary based on the KPSS test.The initial analysis with the unitroot_kpss test (regular KPSS test) suggested that regular (non-seasonal) differencing is necessary to achieve stationarity. This implies that there were likely trends or non-stationary components in the original time series that needed to be removed.In summary, the initial analysis suggested that regular (non-seasonal) differencing was necessary to remove trends or non-stationary components from the time series. After applying this regular differencing, the KPSS test on the differenced series confirmed that the time series had become stationary, indicating that the combination of regular differencing alone was sufficient to achieve stationarity. As a result, seasonal differencing was not needed in this case.

e. Perform KPSS unit root test with your *differenced* data and interpret the test statistic (i.e., *p*-value).

```
# KPSS test
# KPSS test on differenced data
kpss_diff_result <- amazon_stock %>%
  features(difference(Close, lag = 4), unitroot_kpss) %>%
  as.matrix()

# Interpret the KPSS test result
kpss_test_statistic <- kpss_diff_result[1, "kpss_stat"]
kpss_p_value <- kpss_diff_result[1, "kpss_pvalue"]

cat("KPSS Test Statistic:", kpss_test_statistic, "\n")
```

```
KPSS Test Statistic: 0.1815849
```

```
cat("KPSS p-value:", kpss_p_value, "\n")
```

```
KPSS p-value: 0.1
```

```r
# Interpret the p-value
if (kpss_p_value <= 0.05) {
  cat("The KPSS test indicates that the differenced data is stationary (p-value <= 0.05).\n")
} else {
  cat("The KPSS test suggests that the differenced data may not be stationary (p-value > 0.05).\n")
}
```

```
The KPSS test suggests that the differenced data may not be stationary (p-value > 0.05).
```

> Answer "Interpret the test statistic (i.e., p-value)" The test statistics have the results of KPSS Test Statistic: 0.1815849, KPSS p-value: 0.1 . The KPSS test suggests that the differenced data may not be stationary because the p-value (0.1) is greater than the typical significance level of 0.05. In the context of the KPSS test. If the p-value is less than or equal to 0.05, it would typically indicate that the data is stationary. If the p-value is greater than 0.05, it suggests that the data may not be stationary. With a p-value of 0.1, the evidence for stationarity is not strong, and it indicates that there may still be non-stationary components present in the differenced data.

6. Simulate and plot some data from ARIMA models.

Ask some generative AI to write a function that will simulate data from an ARIMA model

The model should have arguments for `time_points`, `constant`, `phi`, `theta`, and `sigma2`

Use `arima_simulate` function to generate data from an AR(1) model with $\phi_1 = 0.6$ and $\sigma^2 = 1$. The process starts with $y_1 = 0$.

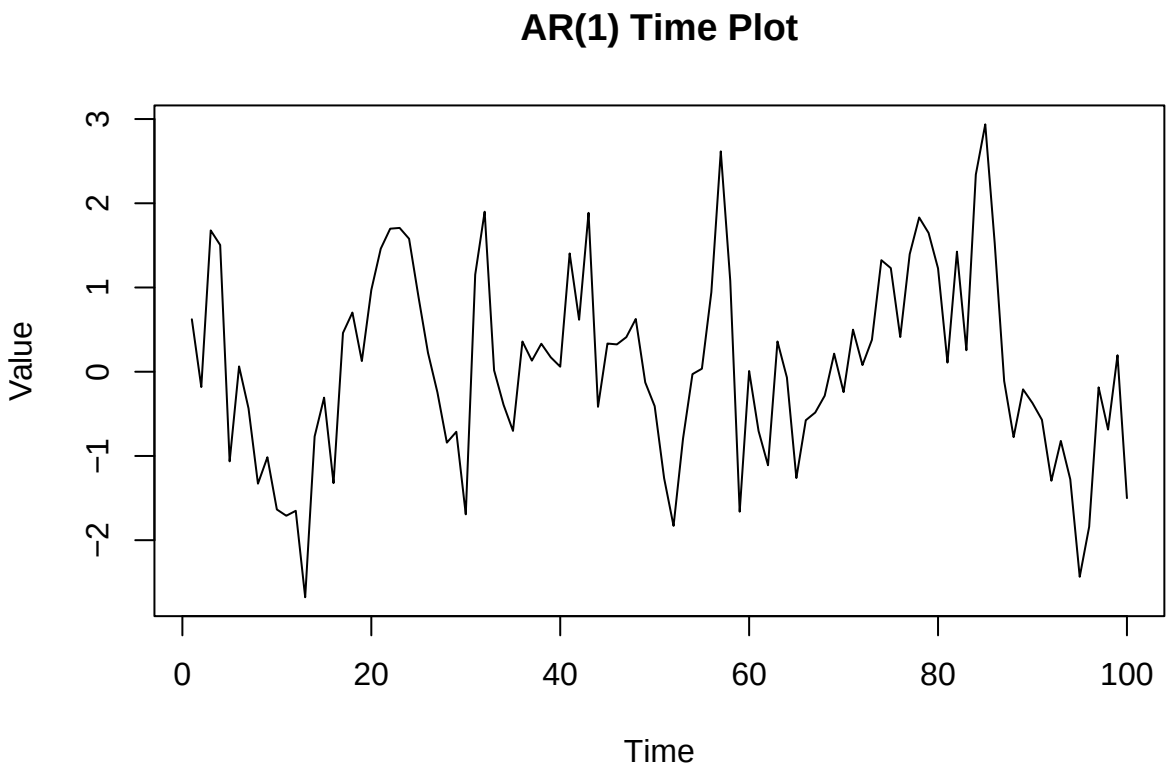
```r
# Set seed for reproducibility
set.seed(123) # Don't change
arima_simulate <- function(time_points, constant, phi, sigma2) {
  # Define the ARIMA model for AR(1)
  model <- list(order = c(1, 0, 0), ar = phi)

  # Simulate data from the ARIMA model
  simulated_data <- arima.sim(model, n = time_points, innov = rnorm(time_points, mean = 0, sd = sqrt(si

  # Add the constant term if specified
  if (constant != 0) {
    simulated_data <- simulated_data + constant
  }

  return(simulated_data)
}

# Parameters for the AR(1) model
phi1 <- 0.6
sigma2 <- 1
constant <- 0  # Start with y_1 = 0
time_points <- 100  # Number of time points to simulate

# Simulate data from the AR(1) model
```

```r
simulated_data <- arima_simulate(time_points, constant, phi1, sigma2)

# Print the simulated data
print(simulated_data)
```

```
Time Series:
Start = 1
End = 100
Frequency = 1
  [1]  0.622750417 -0.182190884  1.677598606  1.504409642 -1.063971371
  [6]  0.062973079 -0.435007561 -1.328828242 -1.015271860 -1.635167564
 [11] -1.709991768 -1.651034329 -2.677313908 -0.768601300 -0.307787662
 [16] -1.322809534  0.460129200  0.702541742  0.126453562  0.970997798
 [21]  1.460732166  1.698020382  1.707452483  1.578389143  0.885121775
 [26]  0.225110402 -0.245404760 -0.841949835 -0.713087179 -1.693248659
 [31]  1.153006770  1.899766060  0.016751053 -0.392834204 -0.702355876
 [36]  0.358551593  0.131761889  0.332375648  0.170878633  0.059656723
 [41]  1.404396318  0.616866805  1.886590687 -0.416798392  0.334534715
 [46]  0.324575073  0.410686612  0.626051450 -0.126692583 -0.409222933
 [51] -1.264109143 -1.830256712 -0.794625386 -0.028565453  0.035864955
 [56]  0.943786441  2.616356550  1.078782764 -1.661899217  0.008598994
 [61] -0.704041366 -1.110433436  0.359311308 -0.069186222 -1.262229446
 [66] -0.576034188 -0.484511875 -0.284942939  0.214314638 -0.242071249
 [71]  0.499133799  0.078993718  0.379178194  1.324345930  1.229789049
 [76]  0.411941844  1.395972725  1.831087491  1.647049454  1.226961407
 [81]  0.108270768  1.425614910  0.255109359  2.340398608  2.936849791
 [86]  1.526409516 -0.110575191 -0.776751678 -0.209167298 -0.372192257
 [91] -0.570857954 -1.294133339 -0.821507728 -1.277809107 -2.434627401
 [96] -1.841002961 -0.185605167 -0.686710063  0.195938284 -1.500319738
```

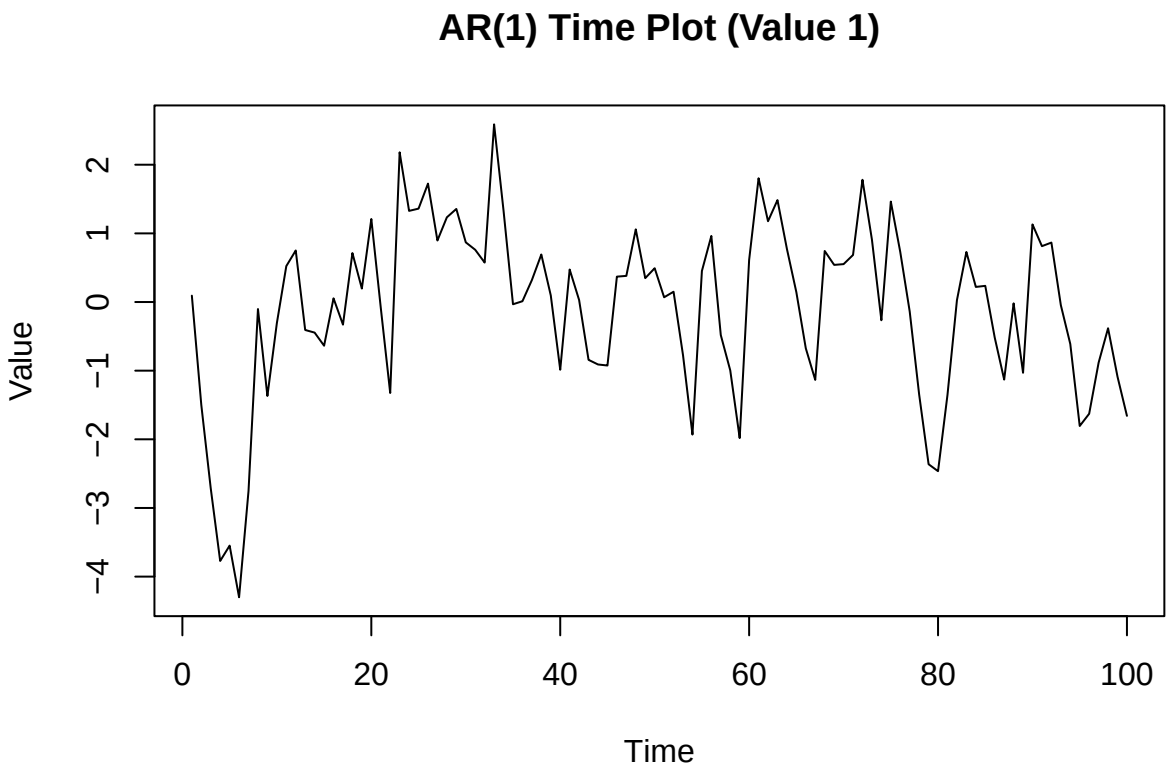
a. Produce a time plot for the series.

```r
# Plot original y
# Create a time sequence for the x-axis
time_sequence <- 1:time_points

# Plot the simulated time series
plot(time_sequence, simulated_data, type = 'l', xlab = 'Time', ylab = 'Value', main = 'AR(1) Time Plot'
```

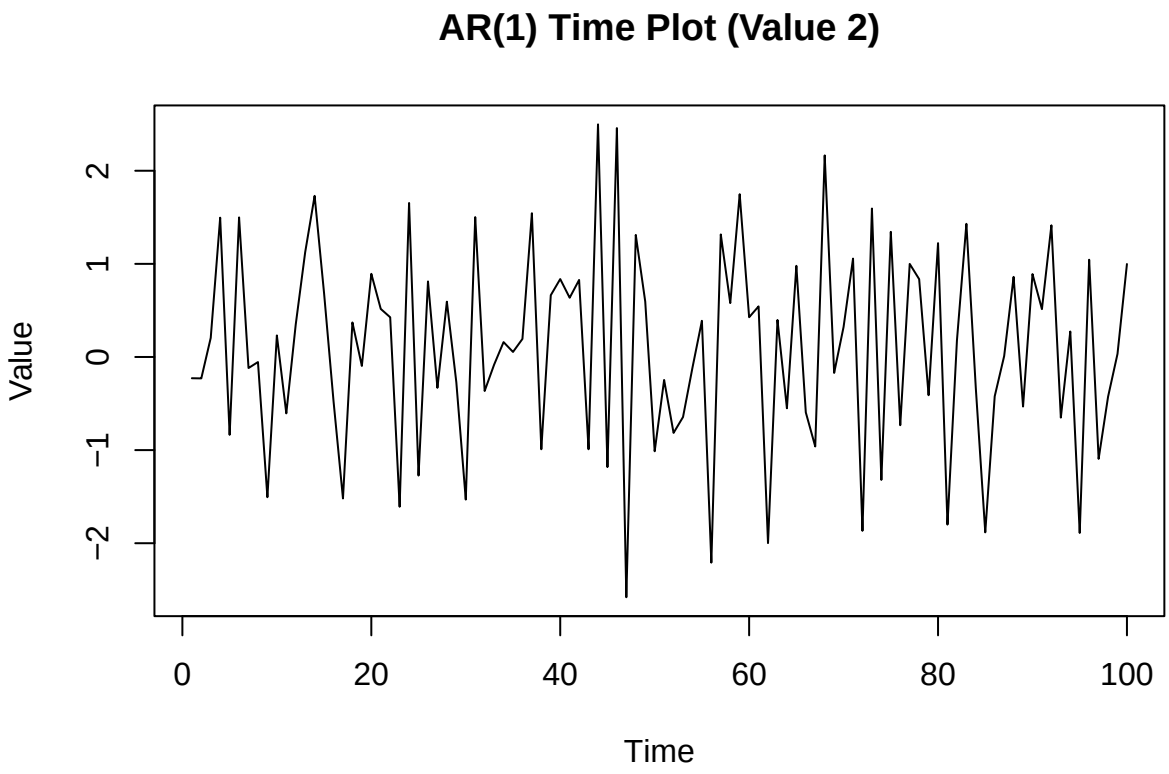


b. Plot three other ϕ values (use some negative and positive values)

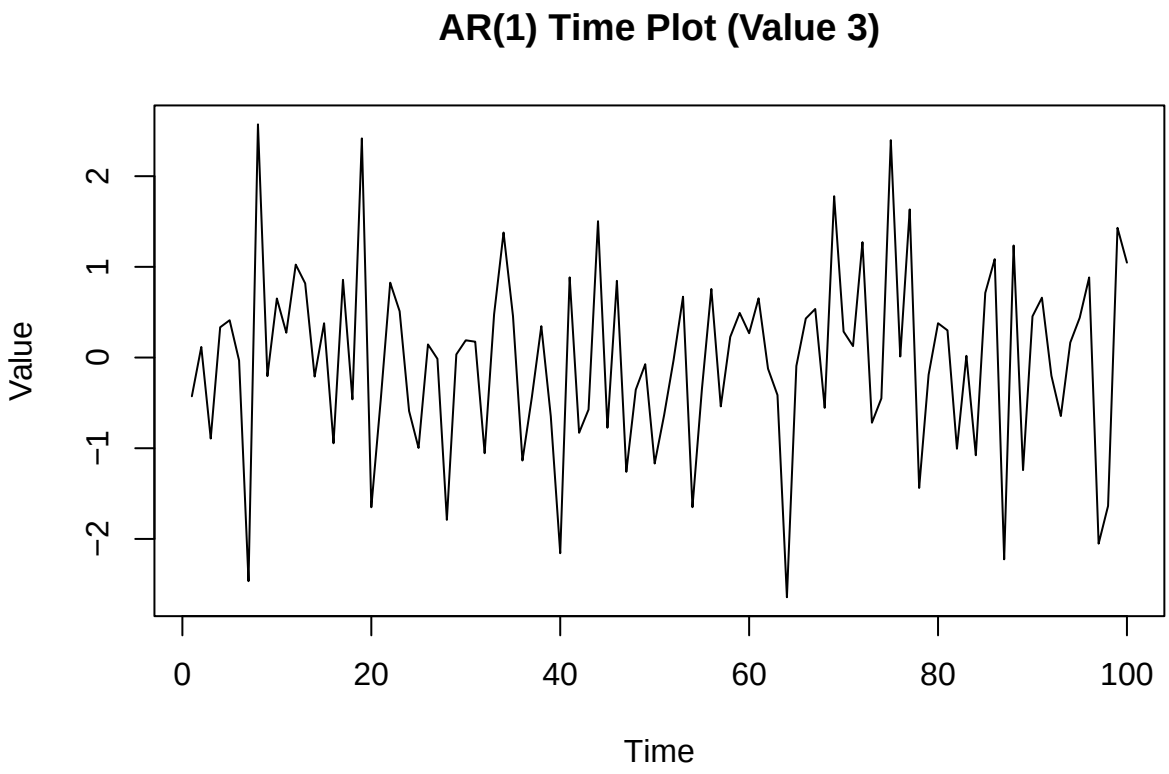
```
# Value 1: $\phi = 0.8$ (Positive Value)
# Set parameters for Value 1
phi1_value1 <- 0.8
simulated_data_value1 <- arima_simulate(time_points, constant, phi1_value1, sigma2)

# Plot Value 1
plot(time_sequence, simulated_data_value1, type = 'l', xlab = 'Time', ylab = 'Value', main = 'AR(1) Time
```




```r
# Value 2: $\phi = -0.5$ (Negative Value)
# Set parameters for Value 2
phi1_value2 <- -0.5
simulated_data_value2 <- arima_simulate(time_points, constant, phi1_value2, sigma2)

# Plot Value 2
plot(time_sequence, simulated_data_value2, type = 'l', xlab = 'Time', ylab = 'Value', main = 'AR(1) Tim
```




```r
# Value 3: $\phi = 0$ (Zero Value)
# Set parameters for Value 3
phi1_value3 <- 0
simulated_data_value3 <- arima_simulate(time_points, constant, phi1_value3, sigma2)

# Plot Value 3
plot(time_sequence, simulated_data_value3, type = 'l', xlab = 'Time', ylab = 'Value', main = 'AR(1) Tim
```



What does `phi` seem to affect in the data generating process?

Answer: The phi seems to affect the data generating process as it changes the trend and stationality that is shown in the positive value 1 and negative value 2 graphs. - Positive phi (e.g., phi > 0): When phi is positive, it indicates a positive correlation between the current observation and past observations. In other words, a positive phi value implies that the data tends to follow a trend or exhibit positive autocorrelation. As a result, the time series data will tend to move in the same direction as its past values, leading to a gradual trend or upward movement. - Negative phi (e.g., phi < 0): When phi is negative, it indicates a negative correlation between the current observation and past observations. In this case, a negative phi value implies that the data tends to exhibit negative autocorrelation or mean reversion. The time series data will tend to move in the opposite direction of its past values, leading to oscillations around a mean or a stationary behavior. - Zero phi (e.g., phi $= 0$): When phi is zero, there is no autoregressive component in the data generating process. This means that each observation is independent of its past values, and the time series data will appear as white noise or exhibit no discernible trend or pattern.

c. Simulate ARIMA data using the best fitting parameters of the Egyptian exports data. Discuss the similarities and differences between the parameters.

```
# Code for Egyptian exports is provided for you
## Obtain the Egyptian exports data
actual_data <- global_economy %>%
  filter(Code == "EGY")%>%
```

```
  as_tsibble(index = Year)
frequency <- 1
yearstart <- min(actual_data$Year)
egyptts <- ts(actual_data$Exports, start=yearstart, frequency=frequency)

## Model the data with the best fit
fit <- auto.arima(egyptts)

## Report the fit
fit
```

```
Series: egyptts
ARIMA(2,0,1) with non-zero mean

Coefficients:
         ar1      ar2      ma1     mean
      1.6764  -0.8034  -0.6896  20.1790
s.e.  0.1111   0.0928   0.1492   0.9142

sigma^2 estimated as 8.046:  log likelihood=-141.57
AIC=293.13   AICc=294.29   BIC=303.43
```

```
# Set seed for reproducibility
set.seed(1234) # Don't change

# Simulate ARIMA data using the same parameters as the Egyptian export data
## You will need to set all parameters!
## time_points = nrow(actual_data)
## constant = one value
## phi = two values!
## theta = one value
## sigma2 = one value
# Extract ARIMA parameters
const <- fit$coef["intercept"]
phi <- c(fit$coef["ar1"], fit$coef["ar2"])
theta <- fit$coef["ma1"]
sigma2 <- fit$sigma2
time_points <- nrow(actual_data)

# Simulate ARIMA data
set.seed(1234)  # For reproducibility
simulated_data <- arima.sim(n = time_points, model = list(order = c(length(phi), 0, length(theta)), ar =

# Report fit
fit
```

```
Series: egyptts
ARIMA(2,0,1) with non-zero mean

Coefficients:
         ar1      ar2      ma1     mean
      1.6764  -0.8034  -0.6896  20.1790
s.e.  0.1111   0.0928   0.1492   0.9142
```

```
sigma^2 estimated as 8.046:  log likelihood=-141.57
AIC=293.13   AICc=294.29   BIC=303.43
```

Answer "Discuss the similarities and differences between the parameters." - Similarities - AR Coefficients (ar1 and ar2): The ARIMA model's AR coefficients obtained from the fitting process (1.6764 and -0.8034) are the same as the AR coefficients used to simulate the data. Both sets of coefficients indicate the same autoregressive relationship within the time series data. -MA Coefficient (ma1): The MA coefficient estimated from the ARIMA model (-0.6896) is identical to the MA coefficient used for simulating the data. This indicates that both the original data and simulated data have the same moving average effect. - Constant (mean): The constant term estimated from the ARIMA model (20.1790) is the same as the constant used for simulating the data. This constant represents the same mean level for both the actual and simulated data. - Variance (sigma^2): The estimated variance of the ARIMA model (8.046) is used as the variance parameter for simulating the data. This means that both the original and simulated data have the same level of volatility. - Differences - ata Source: The primary difference is the source of the data. The parameters obtained from the ARIMA model are based on the actual Egyptian exports data, while the simulated data is generated synthetically using those parameters. Therefore, the data values themselves will differ between the original data and the simulated data. - Randomness: The simulated data incorporates randomness during the simulation process. As a result, even though the model parameters are the same, the simulated data will exhibit different random fluctuations compared to the actual data.

d. Using your data that's supposed to be similar to the Egyptian exports data, check for whether the data are stationary using the KPSS test. If not, then report on whether seasonal differencing and/or differencing are necessary.

```
# KPSS
year_start <- 1960
year_end <- year_start + length(simulated_data) - 1
simulated_df <- tibble(Year = seq(year_start, year_end), Value = simulated_data)
# Convert "Year" column to a "year" type
simulated_df$Year <- as.Date(as.character(simulated_df$Year), format="%Y")
simulated_df_ts <-  simulated_df%>% as_tsibble(index = Year)
simulated_df_ts %>% features(Value, unitroot_kpss) %>% as.matrix()
```

```
     kpss_stat kpss_pvalue
[1,] 0.2820456         0.1
```

```
# Number of seasonal diff
simulated_df_ts %>% features(Value, unitroot_nsdiffs) %>% as.matrix()
```

```
     nsdiffs
[1,]       0
```

```
# Number of diff
simulated_df_ts %>% features(Value, unitroot_ndiffs) %>% as.matrix()
```

```
     ndiffs
[1,]      0
```

Answer “Whether data are stationary (based on KPSS p-value) and whether any differencing is necessary.” - Based on the results of the KPSS (Kwiatkowski-Phillips-Schmidt-Shin) test, we can determine whether the simulated data is stationary and whether any differencing is necessary. - Stationarity Assessment: KPSS Statistic: The KPSS statistic is 0.2820456. KPSS p-value: The p-value associated with the KPSS test is 0.1. - Null Hypothesis (H0): The null hypothesis of the KPSS test is that the data is stationary around a deterministic trend. - Alternative Hypothesis (H1): The alternative hypothesis is that the data has a unit root, indicating non-stationarity. - KPSS p-value Interpretation: If the p-value is less than a significance level (e.g., 0.05), we would reject the null hypothesis, indicating non-stationarity. - In this case, the KPSS p-value is 0.1, which is greater than the typical significance level of 0.05. Therefore, we fail to reject the null hypothesis. - Since the data appears to be stationary based on the KPSS test (p-value > 0.05), there is no immediate need for differencing to achieve stationarity. - The results also indicate that the number of seasonal differences (nsdiffs) and the number of differences (ndiffs) required are both 0. This further confirms that no differencing is necessary to make the data stationary. - In summary, based on the KPSS test, there is no evidence to suggest that differencing is necessary to achieve stationarity in the simulated data, and the data is stationary as indicated by the p-value exceeding the significance level.