Homework 4

2023-11-15

Problem 1

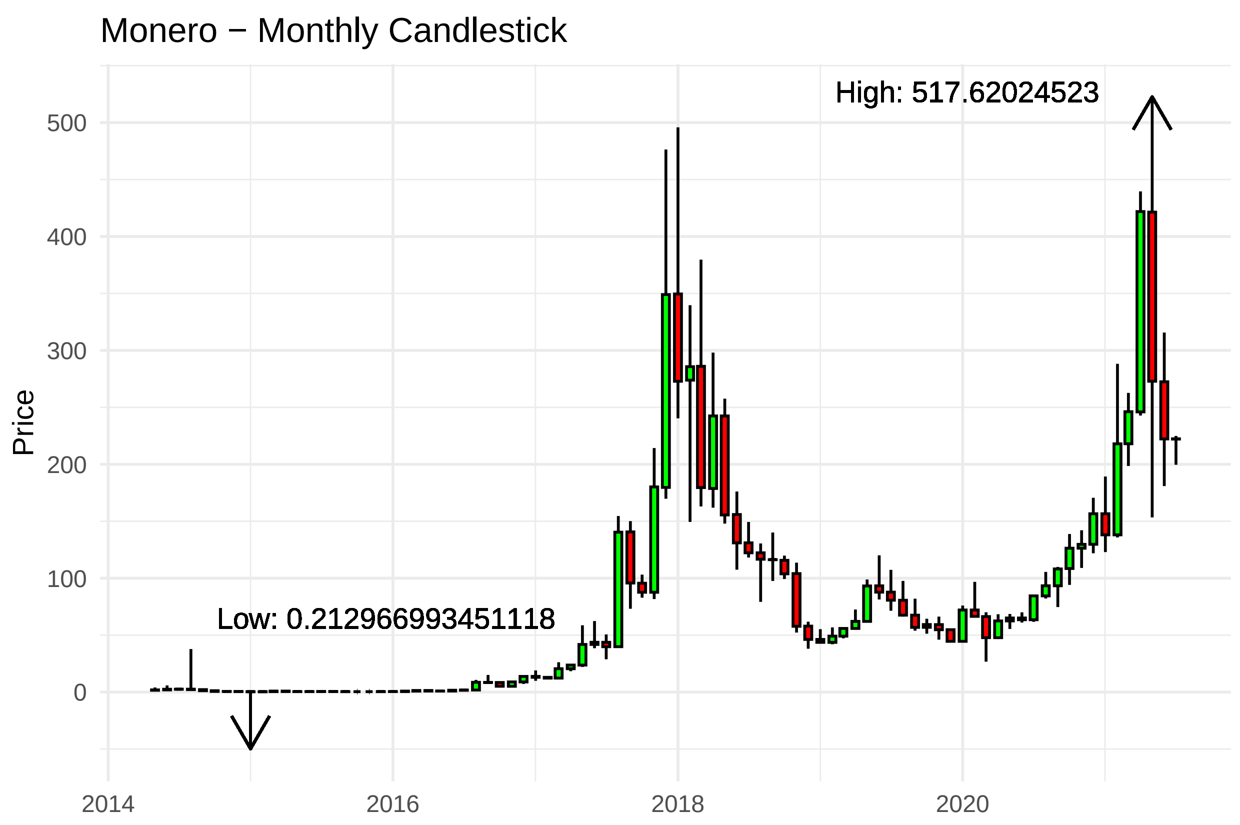
```r
monero <- read.csv("coin_Monero.csv")
nemo <- read.csv("coin_NEM.csv")
polkadot <- read.csv("coin_Polkadot.csv")
```

Part 1

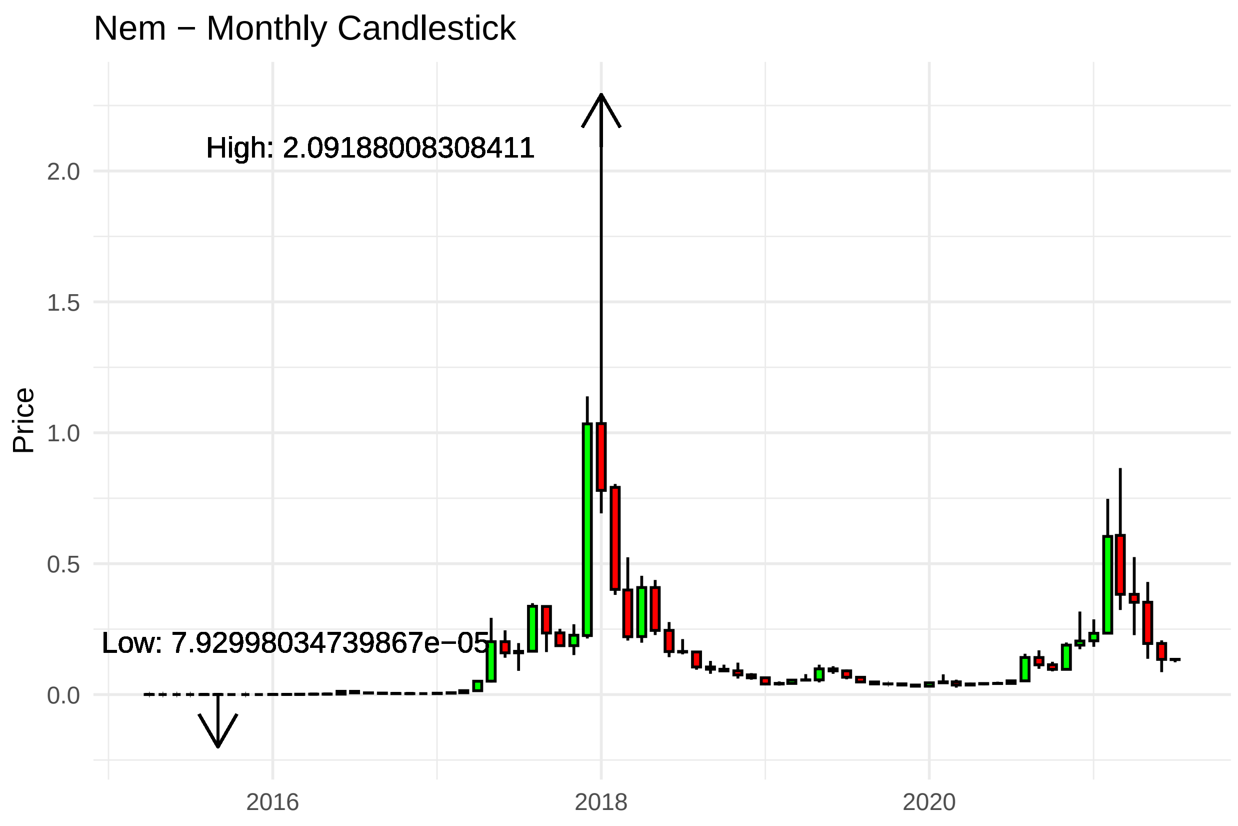
Candlestick plots

```r
# Call the candlestick plot function
candlestick_plot(monthly_data_monero)
```

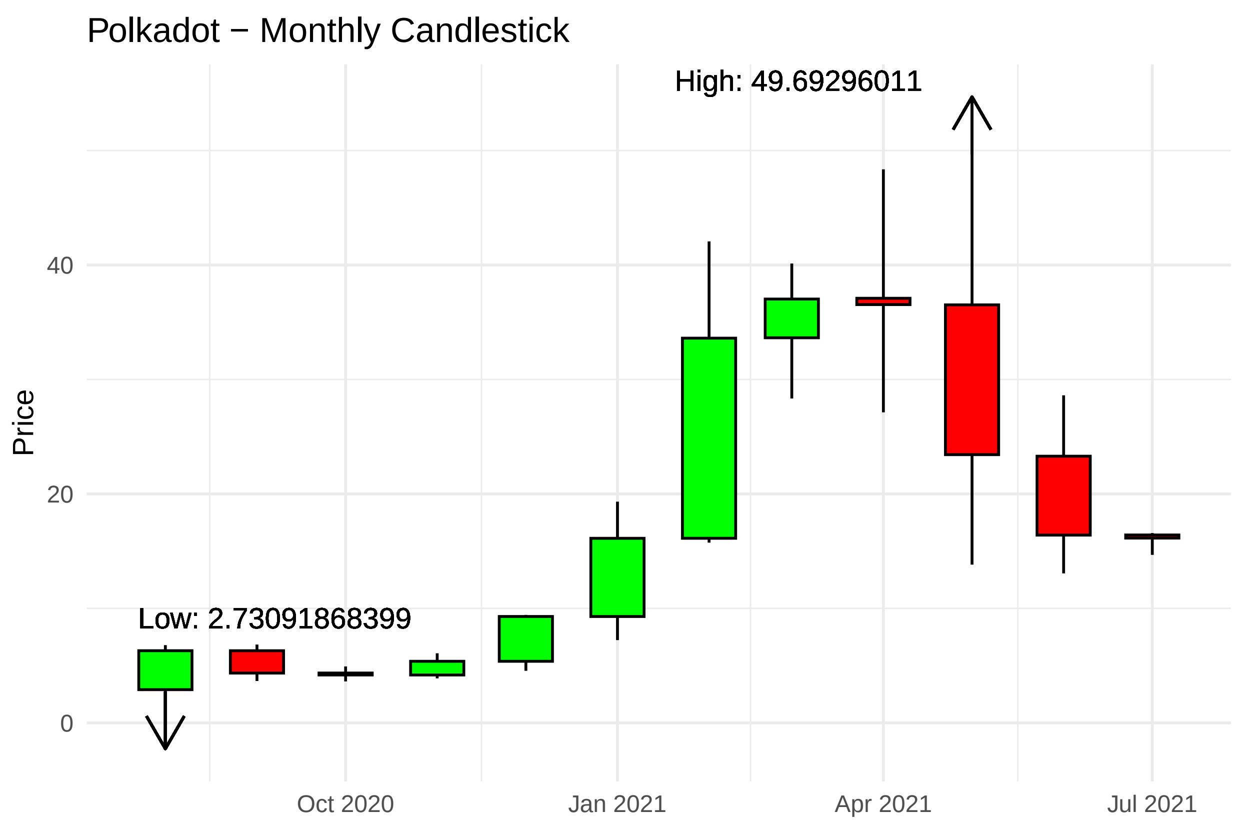
```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



```r
# Convert 'Date' to Date type and sort
nemo$Date <- as.Date(nemo$Date, format = "%Y-%m-%d %H:%M:%S")
nemo <- nemo[order(nemo$Date),]

# Aggregate data by month
monthly_data_nemo <- nemo %>%
mutate(Month = floor_date(Date, "month")) %>% group_by(Month) %>%
summarise(Open = first(Open),
High = max(High), Low = min(Low), Close = last(Close))

# Define the candlestick plot function with annotations
candlestick_plot <- function(data) {
  p <- ggplot(data, aes(x = Month)) +
    geom_linerange(aes(ymin = Low, ymax = High), colour = "black") +
    # Represents the high-low range
    geom_segment(aes(x = Month, xend = Month, y = Open, yend = Close),
                 size = 1,
                 colour = "black") +
    geom_rect(aes(xmin = as.Date(Month) - 9,
                  xmax = as.Date(Month) + 9,
                  ymin = pmin(Open, Close),
                  ymax = pmax(Open, Close),
                  fill = Open < Close),
              # Determines color based on up or down day
              colour = "black") +
    scale_fill_manual(values = c("TRUE" = "green", "FALSE" = "red")) +
    labs(title = "Nem - Monthly Candlestick", x = "", y = "Price") +
    theme_minimal() +
    theme(legend.position = "none")

  highest <- data[which.max(data$High),]
  lowest <- data[which.min(data$Low),]

  # Add annotations for the highest and lowest points
  p + geom_text(aes(x = highest$Month, y = highest$High,
                    label = paste("High:", highest$High)),
                vjust = 0.5, hjust = 1.2, colour = "black", size = 4) +
      geom_text(aes(x = lowest$Month, y = lowest$Low,
                    label = paste("Low:", lowest$Low)),
                vjust = -2, hjust = 0.3, colour = "black", size = 4) +
      # Add arrows pointing to the high and low points
      geom_segment(aes(x = highest$Month, xend = highest$Month,
                       y = highest$High, yend = highest$High + 0.2),
                   arrow = arrow(length = unit(0.5, "cm")), colour = "black") +
      geom_segment(aes(x = lowest$Month, xend = lowest$Month,
                       y = lowest$Low, yend = lowest$Low - 0.2),
                   arrow = arrow(length = unit(0.5, "cm")), colour = "black")
}
# Call the candlestick plot function
candlestick_plot(monthly_data_nemo)
```



```r
# Convert 'Date' to Date type and sort
polkadot$Date <- as.Date(polkadot$Date, format = "%Y-%m-%d %H:%M:%S")
polkadot <- polkadot[order(polkadot$Date),]

# Aggregate data by month
monthly_data_polkadot <- polkadot %>%
  mutate(Month = floor_date(Date, "month")) %>%
  group_by(Month) %>%
  summarise(Open = first(Open),
            High = max(High),
            Low = min(Low),
            Close = last(Close))

# Define the candlestick plot function with annotations
candlestick_plot <- function(data) {
  p <- ggplot(data, aes(x = Month)) +
    geom_linerange(aes(ymin = Low, ymax = High), colour = "black") +
    # Represents the high-low range
    geom_segment(aes(x = Month, xend = Month, y = Open, yend = Close),
                 size = 1,
                 colour = "black") +
    geom_rect(aes(xmin = as.Date(Month) - 9,
                  xmax = as.Date(Month) + 9,
                  ymin = pmin(Open, Close),
                  ymax = pmax(Open, Close),
                  fill = Open < Close),  # Determines color based on up or down day
              colour = "black") +
    scale_fill_manual(values = c("TRUE" = "green", "FALSE" = "red")) +
    labs(title = "Polkadot - Monthly Candlestick", x = "", y = "Price") +
    theme_minimal() +
    theme(legend.position = "none")

  highest <- data[which.max(data$High),]
  lowest <- data[which.min(data$Low),]

  # Add annotations for the highest and lowest points
  p + geom_text(aes(x = highest$Month, y = highest$High,
                    label = paste("High:", highest$High)),
                vjust = -3, hjust = 1.2, colour = "black", size = 4) +
      geom_text(aes(x = lowest$Month, y = lowest$Low,
                    label = paste("Low:", lowest$Low)),
                vjust = -3, hjust = 0.1, colour = "black", size = 4) +
      # Add arrows pointing to the high and low points
      geom_segment(aes(x = highest$Month, xend = highest$Month,
                       y = highest$High, yend = highest$High + 5),
                   arrow = arrow(length = unit(0.5, "cm")), colour = "black") +
      geom_segment(aes(x = lowest$Month, xend = lowest$Month,
                       y = lowest$Low, yend = lowest$Low - 5),
                   arrow = arrow(length = unit(0.5, "cm")), colour = "black")
}

candlestick_plot(monthly_data_polkadot)
```

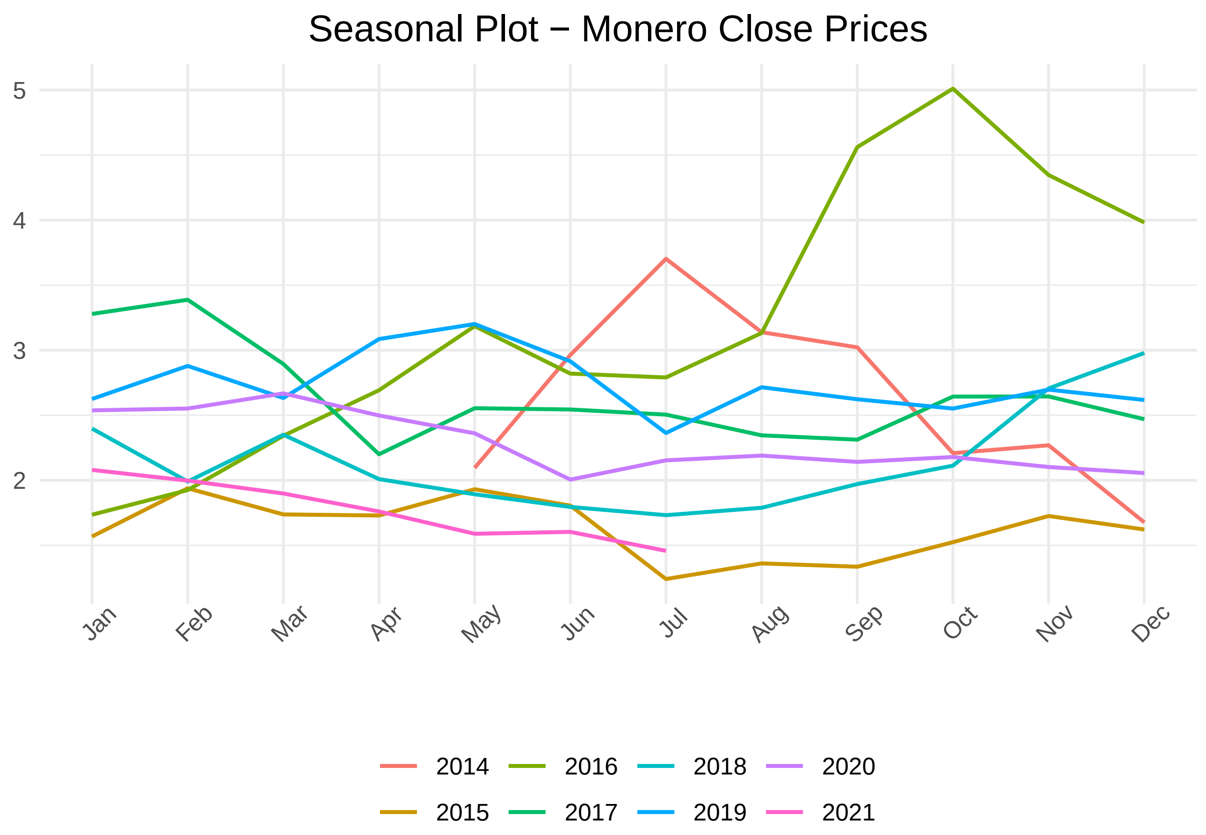


Part 2

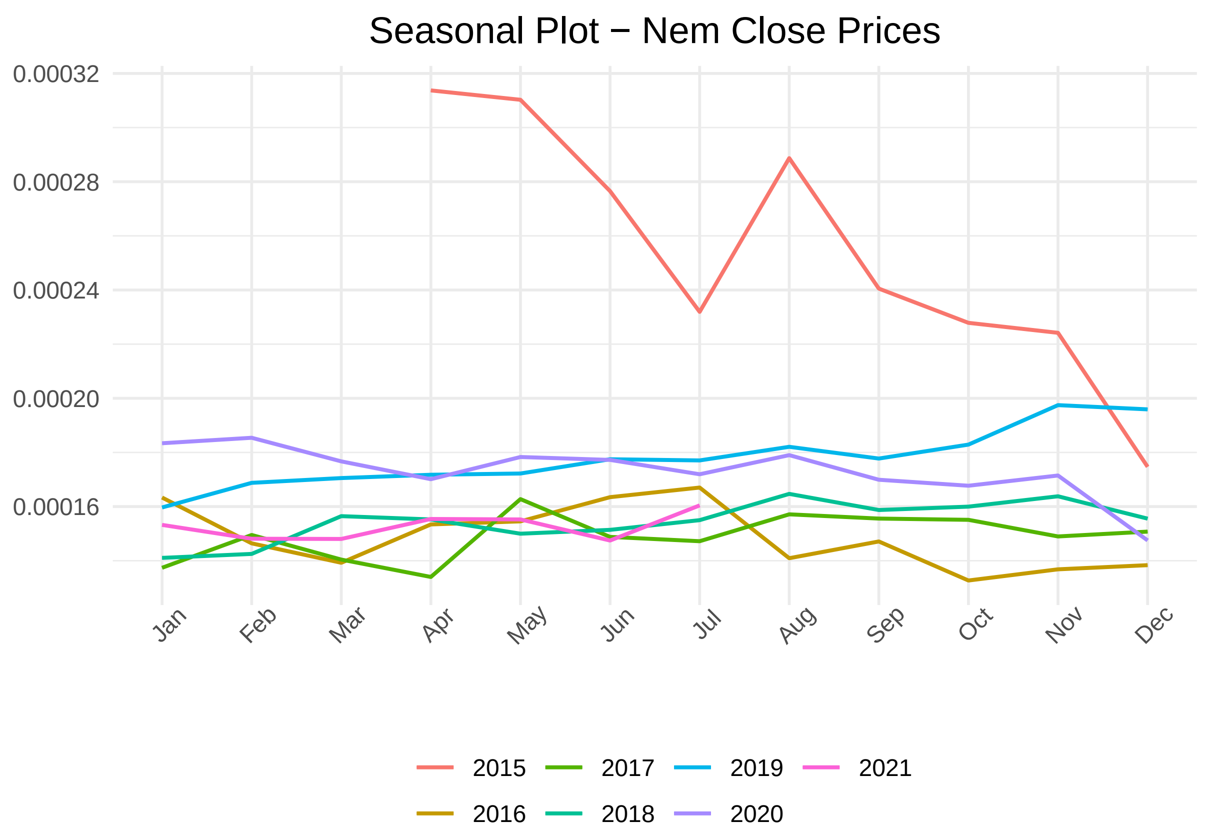
```r
# Create a time series object
monero_ts <- ts(monero$Close, start = c(year(min(monero$Date)),
                                        month(min(monero$Date))),
                end = c(year(max(monero$Date)),
                        month(max(monero$Date))),
                frequency = 12)

# Create a time series object
nemo_ts <- ts(nemo$Close, start = c(year(min(nemo$Date)),
                                    month(min(nemo$Date))),
                  end = c(year(max(nemo$Date)),
                          month(max(nemo$Date))),
                  frequency = 12)

polkadot_ts <- ts(polkadot$Close,
                  start = c(year(min(polkadot$Date)),
                            month(min(polkadot$Date))),
                  end = c(year(max(polkadot$Date)), month(max(polkadot$Date))),
                  frequency = 12)
```

Seasonal plots

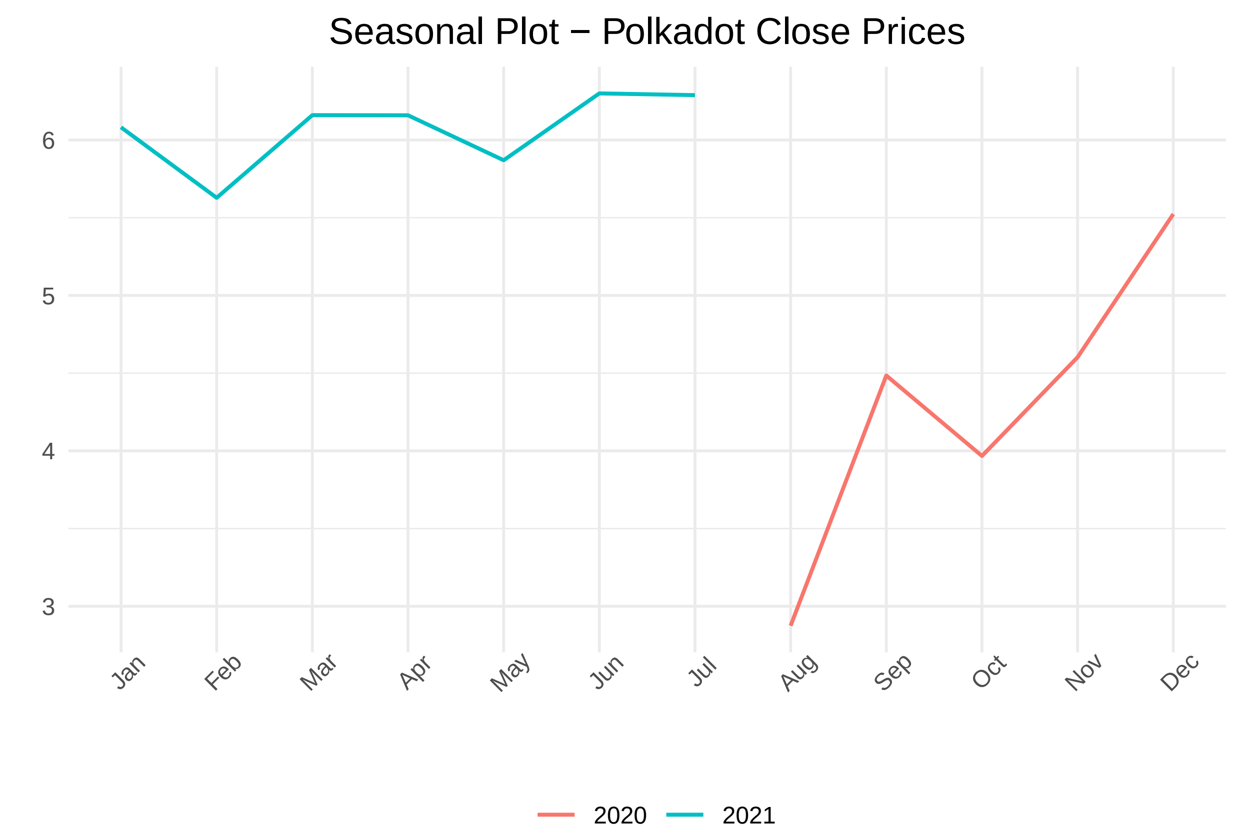
```r
ggseasonplot(monero_ts, year.labels = F) +
  ggtitle("Seasonal Plot - Monero Close Prices") +
  xlab("") +
  ylab("") +
  theme(panel.grid.minor = element_blank(),
        plot.title = element_text(size = 14, hjust = 0.5)) +
  geom_line(linetype = "solid", size = 0.7) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, hjust = 0.5),
    axis.text.x = element_text(angle = 45, vjust = 1),
    legend.position = "bottom",
    legend.title = element_blank()
  )
```



```r
ggseasonplot(nemo_ts, year.labels = F) +
  ggtitle("Seasonal Plot - Nem Close Prices")  +
  xlab("") +
  ylab("") +
  theme(panel.grid.minor = element_blank(),
        plot.title = element_text(size = 14, hjust = 0.5)) +
  geom_line(linetype = "solid", size = 0.7) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, hjust = 0.5),
    axis.text.x = element_text(angle = 45, vjust = 1),
    legend.position = "bottom",
    legend.title = element_blank()
  )
```

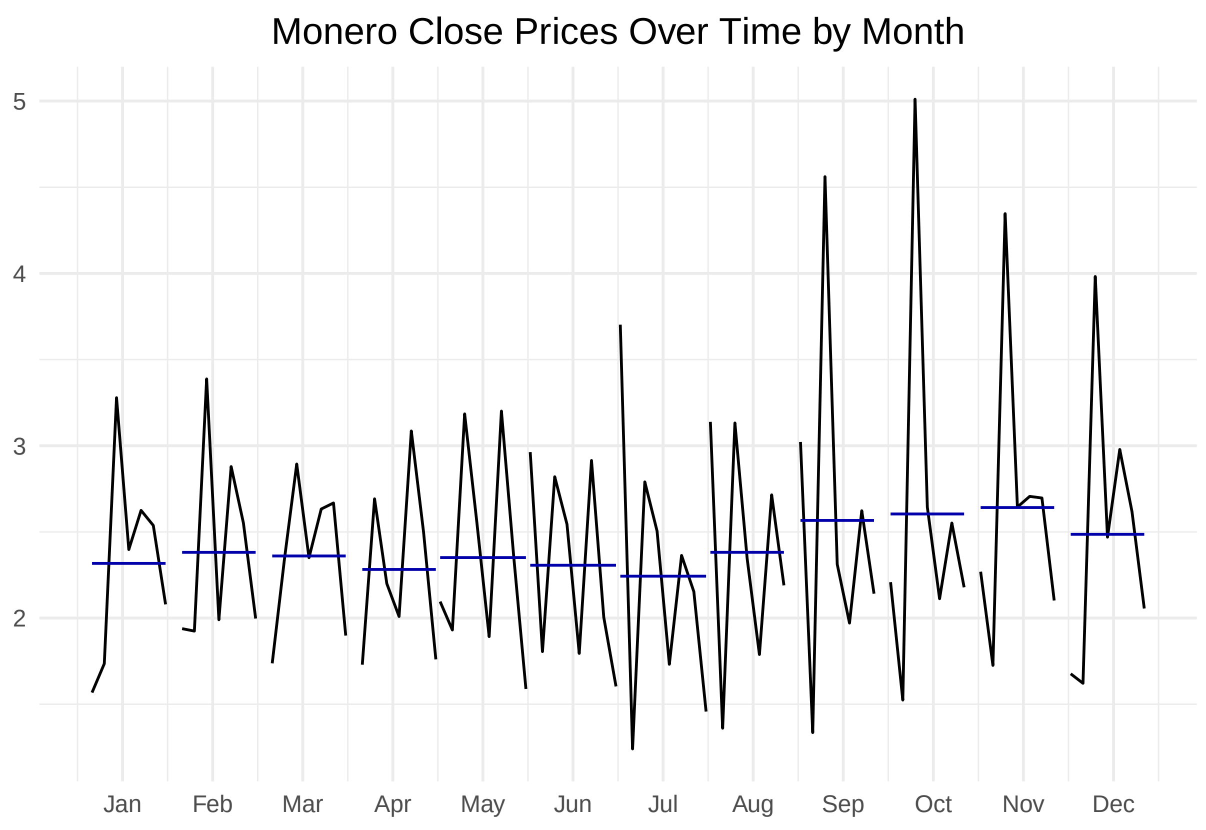


```
ggseasonplot(polkadot_ts, year.labels = F) +
  ggtitle("Seasonal Plot - Polkadot Close Prices")  +
  xlab("") +
  ylab("") +
  theme(panel.grid.minor = element_blank(),
        plot.title = element_text(size = 14, hjust = 0.5)) +
  geom_line(linetype = "solid", size = 0.7) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, hjust = 0.5),
    axis.text.x = element_text(angle = 45, vjust = 1),
    legend.position = "bottom",
    legend.title = element_blank()
  )
```

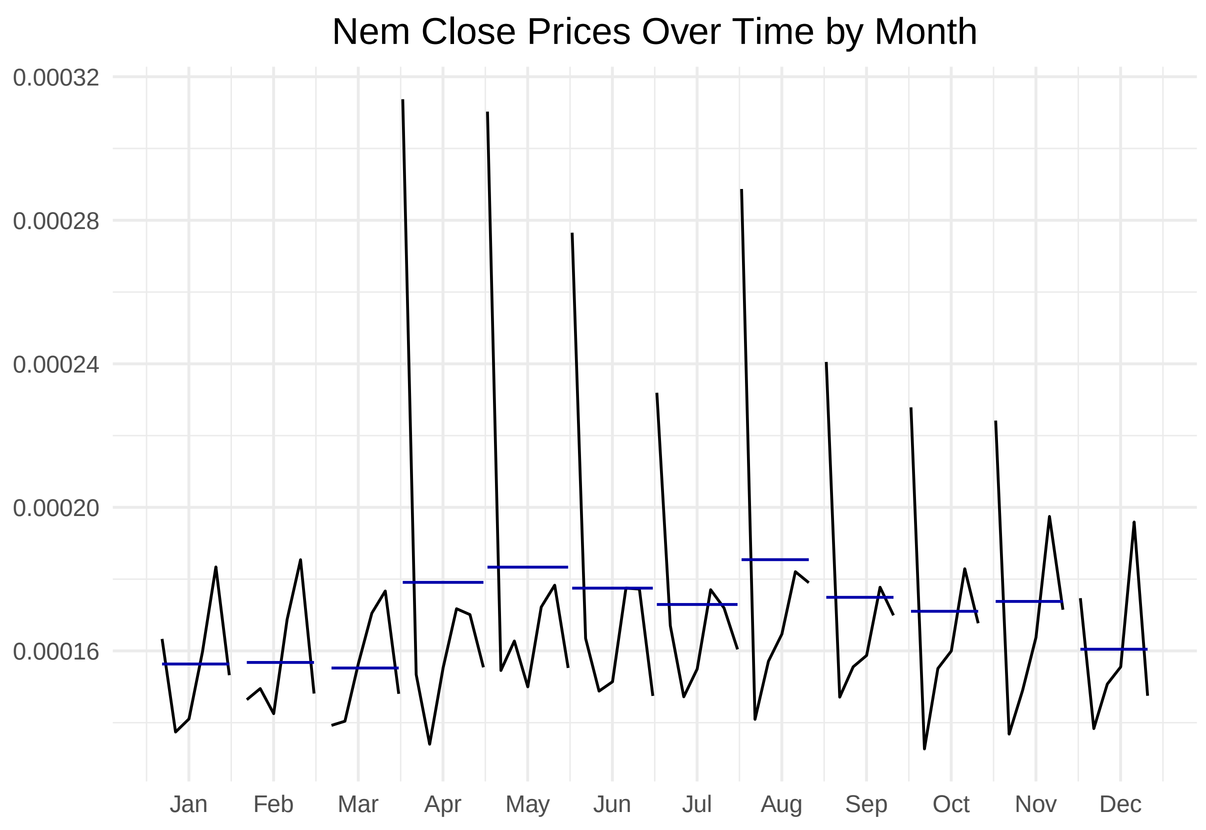


Subseries plot

```r
# Plot the subseries plot of the monero time series object
ggsubseriesplot(monero_ts) + ylab("") +
  ggtitle("Monero Close Prices Over Time by Month") +
  xlab("") + theme_minimal() +
  theme(plot.title = element_text(size = 14, hjust = 0.5))
```



```r
# Plot the subseries plot of the nemo time series object
ggsubseriesplot(nemo_ts) + ylab("") +
  ggtitle("Nem Close Prices Over Time by Month") +
  xlab("") + theme_minimal() +
  theme(plot.title = element_text(size = 14, hjust = 0.5))
```

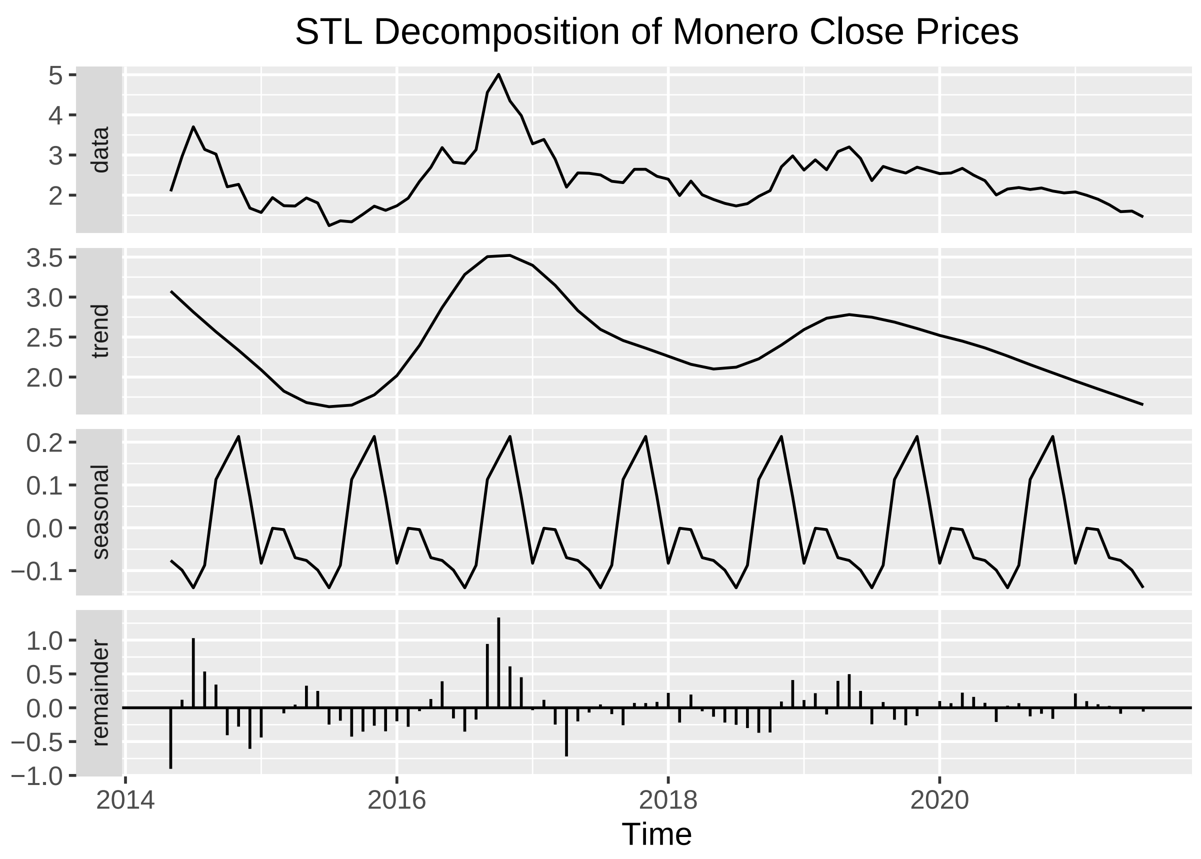


Decomposition plots

```
# Assuming monero_ts is your time series object
decomposed <- stl(monero_ts, s.window = 'periodic')

# Get the range of data without NA values for plotting
time_range <- time(na.omit(decomposed$time.series))

# Plot the decomposed object
autoplot(decomposed) +
  ggtitle("STL Decomposition of Monero Close Prices") +
  theme(
    plot.title = element_text(size = 14, hjust = 0.5),
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10)
  ) +
  xlim(min(time_range), max(time_range)) # Set limits to avoid NA values
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

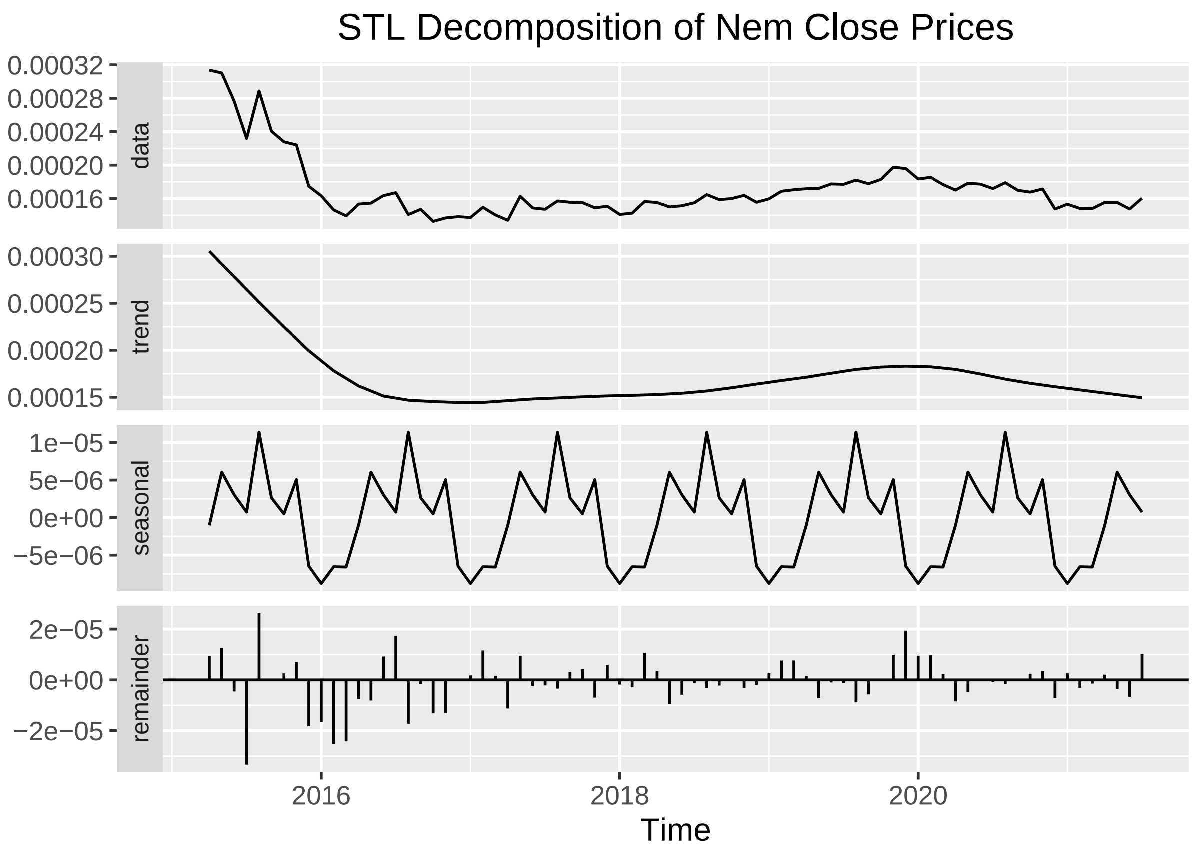
```
## Warning: Removed 4 rows containing missing values (`geom_rect()`).
```



```r
# Assuming nemo_ts is your time series object
decomposed <- stl(nemo_ts, s.window = 'periodic')

# Get the range of data without NA values for plotting
time_range <- time(na.omit(decomposed$time.series))

# Plot the decomposed object
autoplot(decomposed) +
  ggtitle("STL Decomposition of Nem Close Prices") +
  theme(
    plot.title = element_text(size = 14, hjust = 0.5),
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10)
  ) +
  xlim(min(time_range), max(time_range)) # Set limits to avoid NA values
```

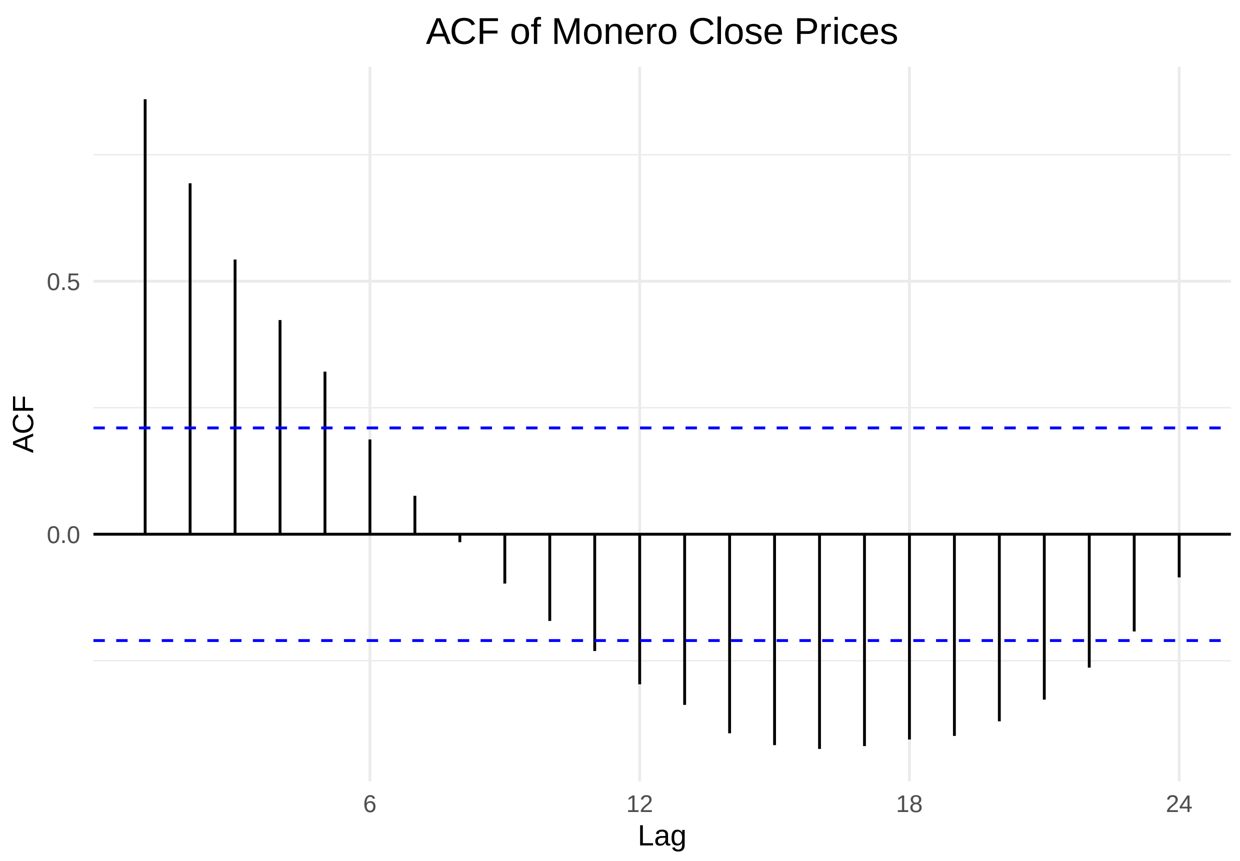
```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

```
## Warning: Removed 4 rows containing missing values (`geom_rect()`).
```

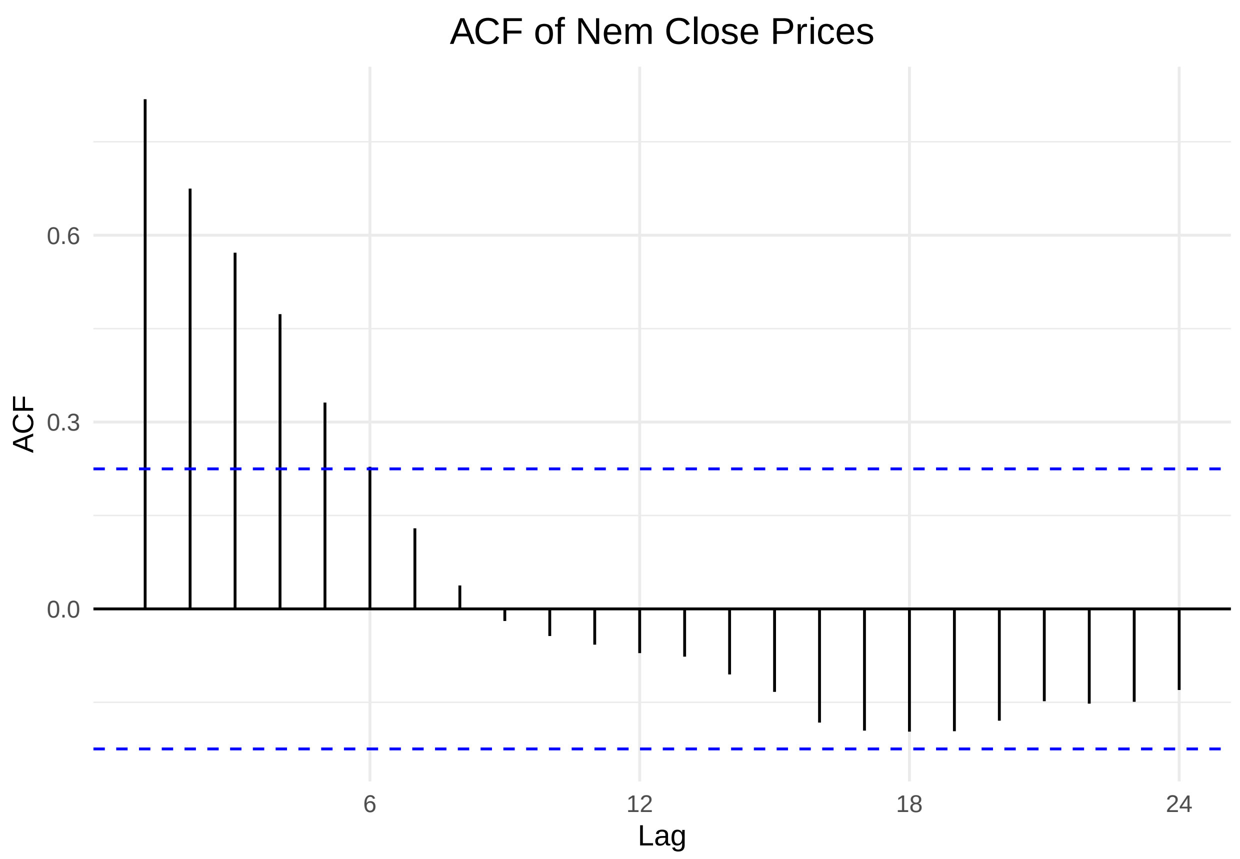


ACF plots

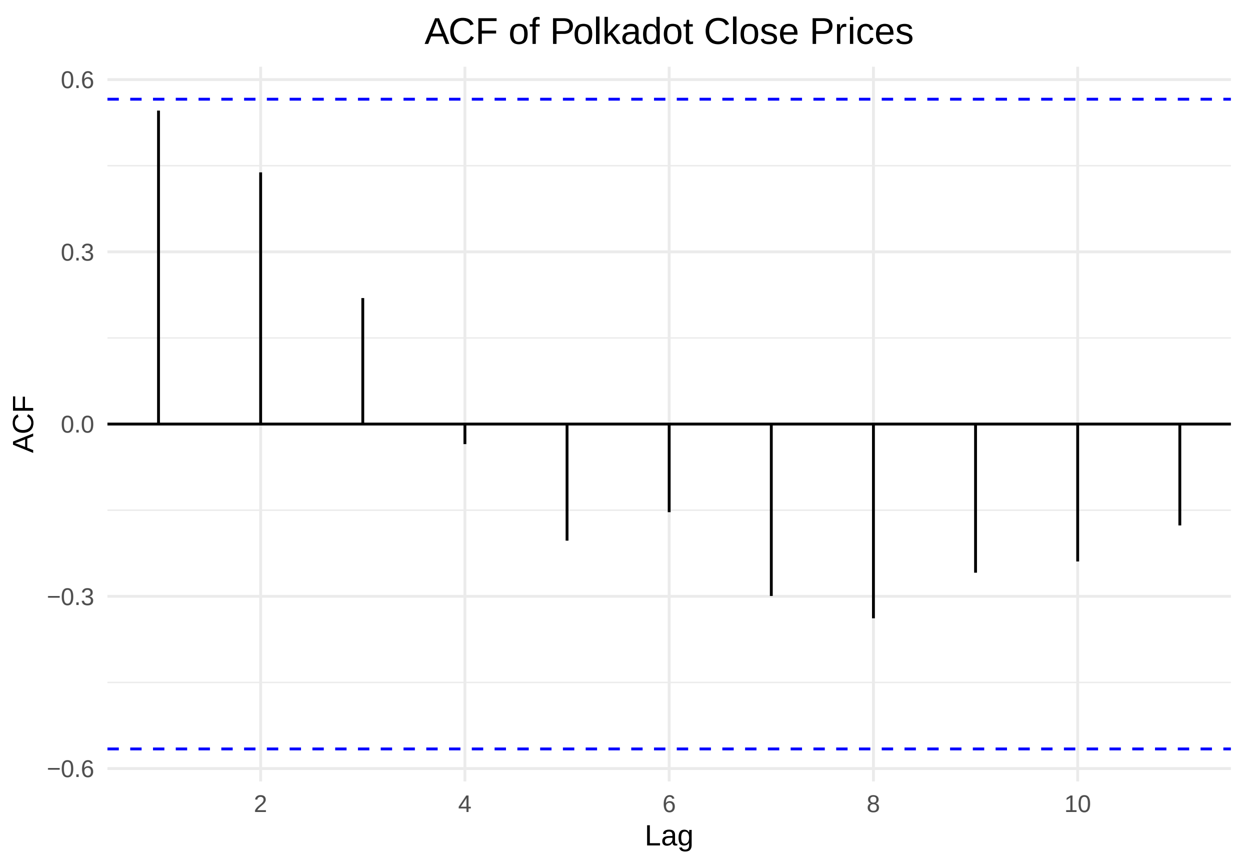
```r
# Plot the ACF of the monero time series object
ggAcf(monero_ts) + ggtitle("ACF of Monero Close Prices") + theme_minimal() +
  theme(plot.title = element_text(size = 14, hjust = 0.5))
```



```
# Plot the ACF of the nemo time series object
ggAcf(nemo_ts) + ggtitle("ACF of Nem Close Prices") + theme_minimal() +
  theme(plot.title = element_text(size = 14, hjust = 0.5))
```

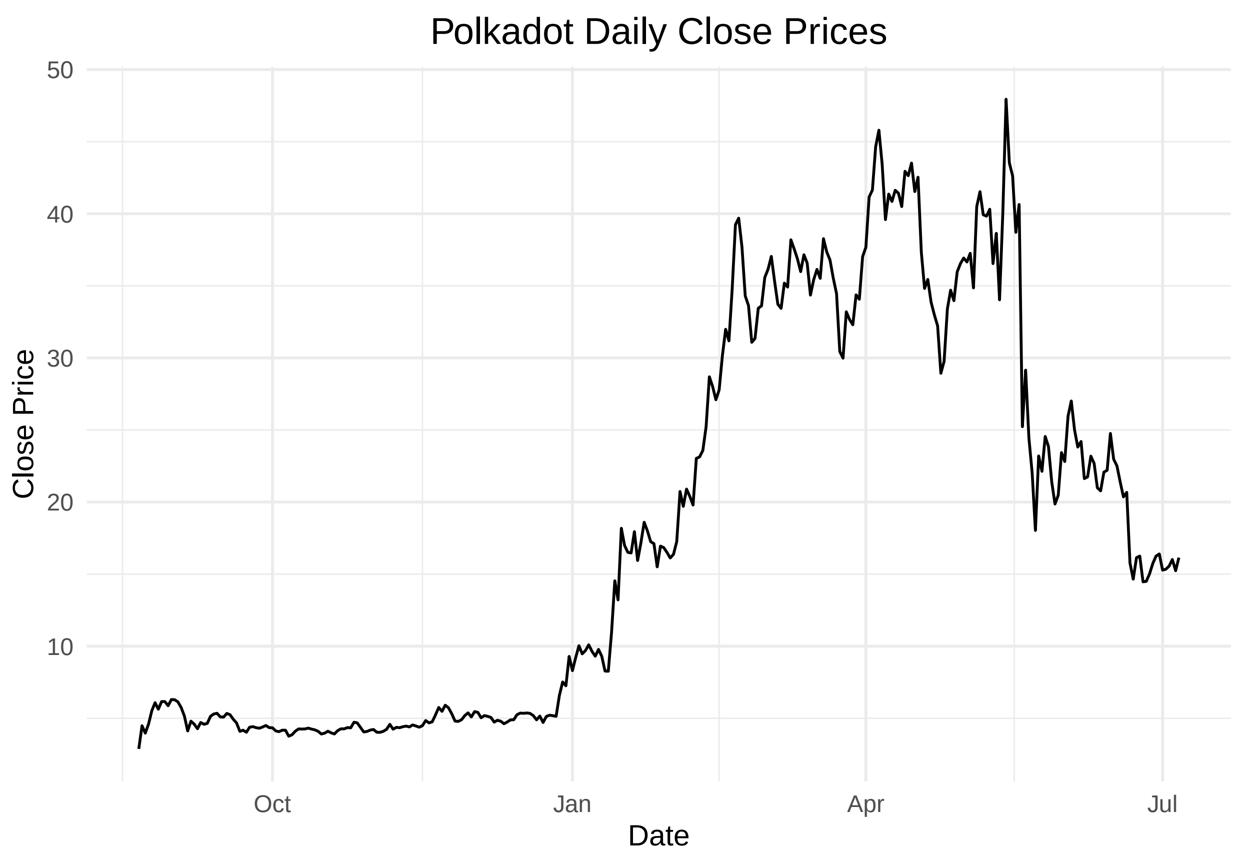


```r
ggAcf(polkadot_ts) + ggtitle("ACF of Polkadot Close Prices") + theme_minimal() +
  theme(plot.title = element_text(size = 14, hjust = 0.5))
```



Polkadot contain data less than for 2 years

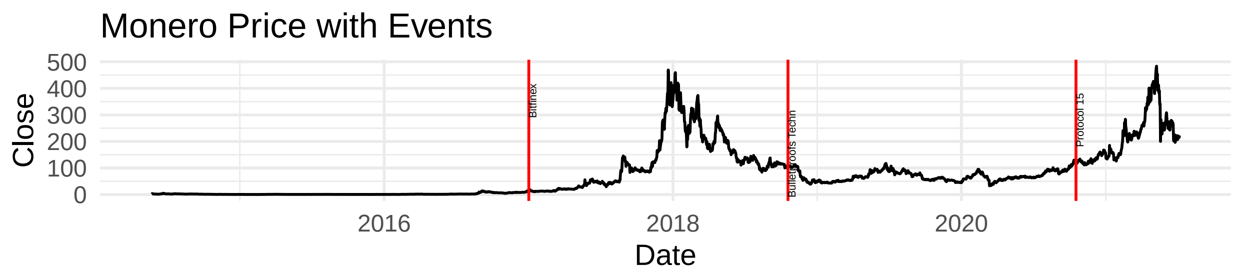
```r
# Plot the line graph for daily close prices of Polkadot
ggplot(polkadot, aes(x = Date, y = Close)) +
  geom_line(linetype = "solid") +
  ggtitle("Polkadot Daily Close Prices") +
  xlab("Date") +
  ylab("Close Price") +
  theme_minimal()+
  theme(plot.title = element_text(size = 14, hjust = 0.5))
```

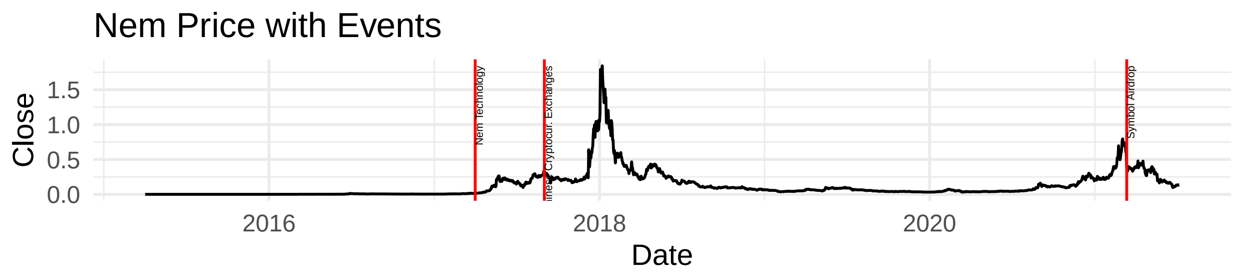


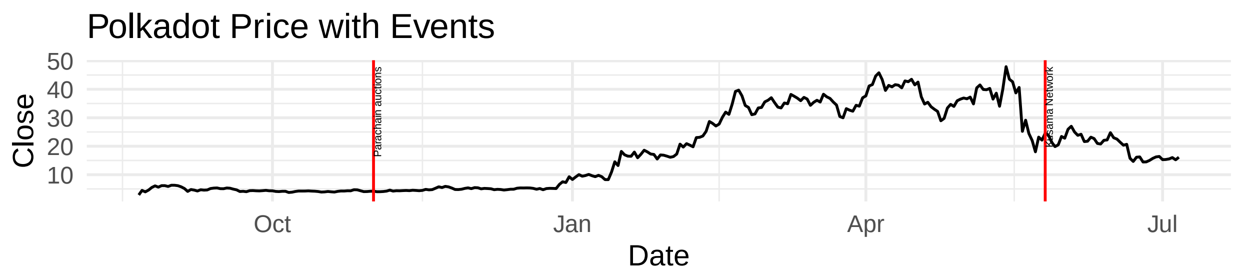
As a result, decomposition describes the seasoanlity the best because it separates the overall trend, which shows long-term changes, from the recurring patterns that repeat at regular intervals. By isolating these components, decomposition makes it easier to identify and understand the seasonal patterns in the data. Part 3

```
# Assuming you have defined 'events_monero' with the Monero events data
events_monero <- data.frame(
  Date = as.Date(c('2018-10-19', '2020-10-17', '2017-01-01')),
  Event = c('Bulletproofs Techn', 'Protocol 15', 'Bitfinex')
)

# Create a ggplot for Monero data
p_monero <- ggplot(monero, aes(x = Date, y = Close)) +
  geom_line() +
  theme_minimal() +
  ggtitle("Monero Price with Events")

# Add event annotations for Monero
p_monero <- p_monero +
  geom_segment(data = events_monero, aes(x = Date, xend = Date,
                                         y = Inf, yend = -Inf),
               color = "red") +
  annotate("text", x = events_monero$Date, y = max(monero$Close),
           label = events_monero$Event, vjust = 1, hjust = 1.5, angle = 90, size = 1.5)
```

```r
events_nemo <- data.frame(
  Date = as.Date(c('2017-04-01', '2017-09-01', '2021-03-12')),
  Event = c('Nem Technology', 'Chinese Cryptocur. Exchanges', 'Symbol Airdrop')
)

# Create a ggplot for NEMO data
p_nemo <- ggplot(nemo, aes(x = Date, y = Close)) +
  geom_line() +
  theme_minimal() +
  ggtitle("Nem Price with Events")

# Add event annotations for NEMO
p_nemo <- p_nemo +
  geom_segment(data = events_nemo, aes(x = Date, xend = Date,
                                       y = Inf, yend = -Inf),
               color = "red") +
  annotate("text", x = events_nemo$Date, y = max(nemo$Close),
           label = events_nemo$Event, vjust = 1, hjust = 1, angle = 90, size = 1.5)


events_polkadot <- data.frame(
  Date = as.Date(c('2020-11-01', '2021-05-26')),
  Event = c("Parachain auctions", 'Kusama Network')
)

# Create a ggplot for Polkadot data
p_polkadot <- ggplot(polkadot, aes(x = Date, y = Close)) +
  geom_line() +
  theme_minimal() +
  ggtitle("Polkadot Price with Events")

# Add event annotations for Polkadot
p_polkadot <- p_polkadot +
  geom_segment(data = events_polkadot, aes(x = Date, xend = Date,
                                           y = Inf, yend = -Inf),
               color = "red") +
  annotate("text", x = events_polkadot$Date, y = max(polkadot$Close),
           label = events_polkadot$Event, vjust = 1, hjust = 1, angle = 90, size = 1.5)

# Combine the three plots vertically
grid.arrange(p_monero, p_nemo, p_polkadot, ncol = 1)
```



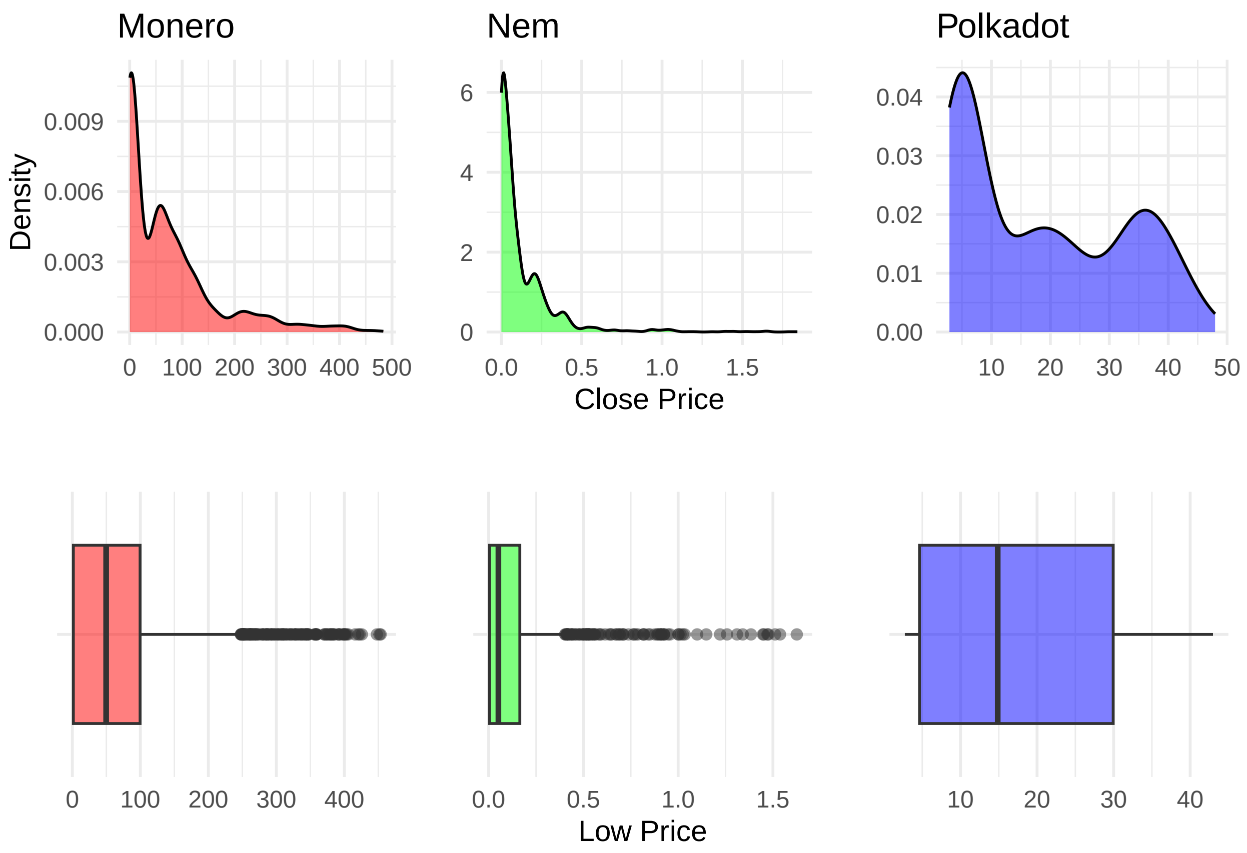




Part 4

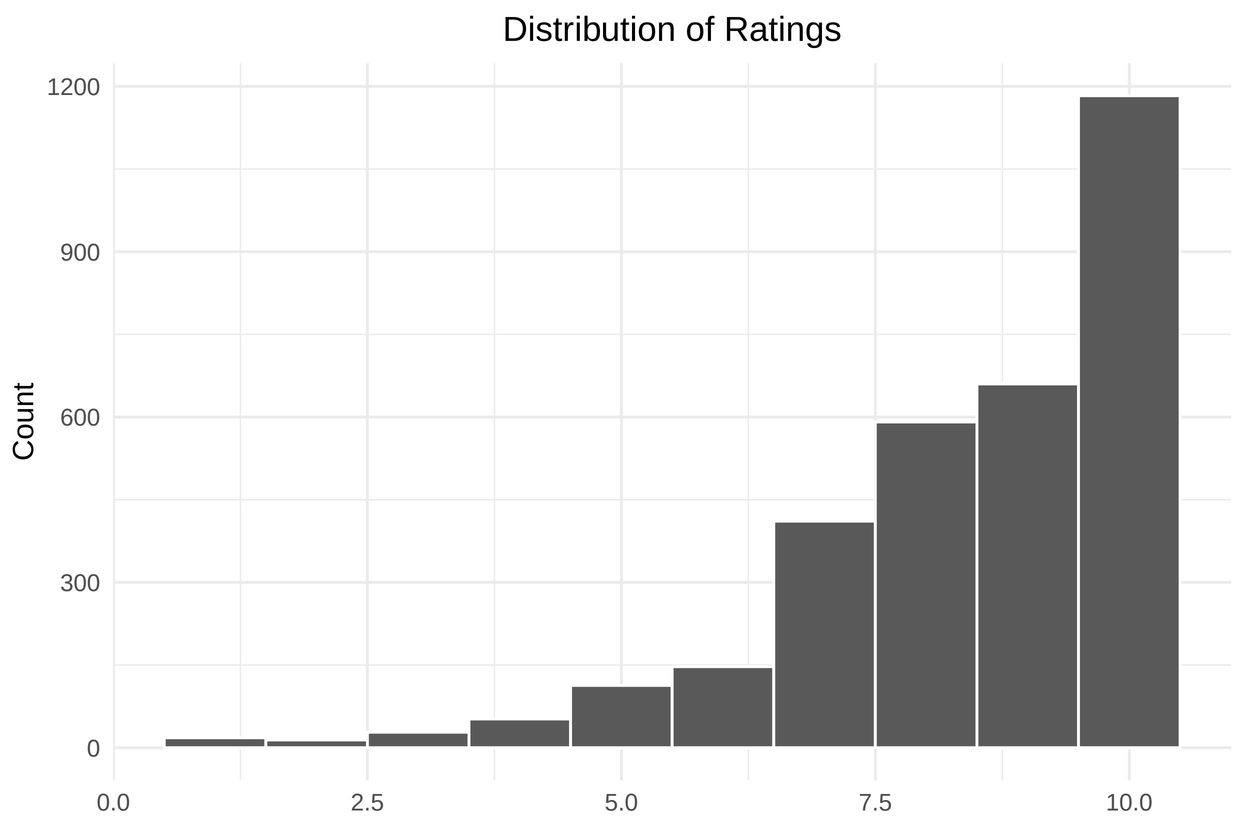
```
library(ggplot2)
library(gridExtra)

# Monero Close Price Density Plot
monero_plot <- ggplot(monero, aes(x = Close)) +
  geom_density(fill = "red", alpha = 0.5) +
  labs(title = "Monero", x = "", y = "Density") + theme_minimal()

# NEM Close Price Density Plot
nem_plot <- ggplot(nemo, aes(x = Close)) +
  geom_density(fill = "green", alpha = 0.5) +
  labs(title = "Nem", x = "Close Price", y = "") + theme_minimal()

# Polkadot Close Price Density Plot
polkadot_plot <- ggplot(polkadot, aes(x = Close)) +
  geom_density(fill = "blue", alpha = 0.5) +
  labs(title = "Polkadot", x = "", y = "") + theme_minimal()

# Monero Low Price Box Plot
monero_box <- ggplot(monero, aes(x = Low, y = "")) +
  geom_boxplot(fill = "red", alpha = 0.5) +
  labs(title = "", x = "", y = "") + theme_minimal()

# NEM Low Price Box Plot
```

```r
nem_box <- ggplot(nemo, aes(x = Low, y = "")) +
  geom_boxplot(fill = "green", alpha = 0.5) +
  labs(title = "", x = "Low Price", y = "") + theme_minimal()

# Polkadot Low Price Box Plot
polkadot_box <- ggplot(polkadot, aes(x = Low, y = "")) +
  geom_boxplot(fill = "blue", alpha = 0.5) +
  labs(title = "", x = "", y = "") + theme_minimal()

# Arrange the plots side by side
grid.arrange(monero_plot, nem_plot, polkadot_plot, monero_box, nem_box, polkadot_box, ncol = 3)
```



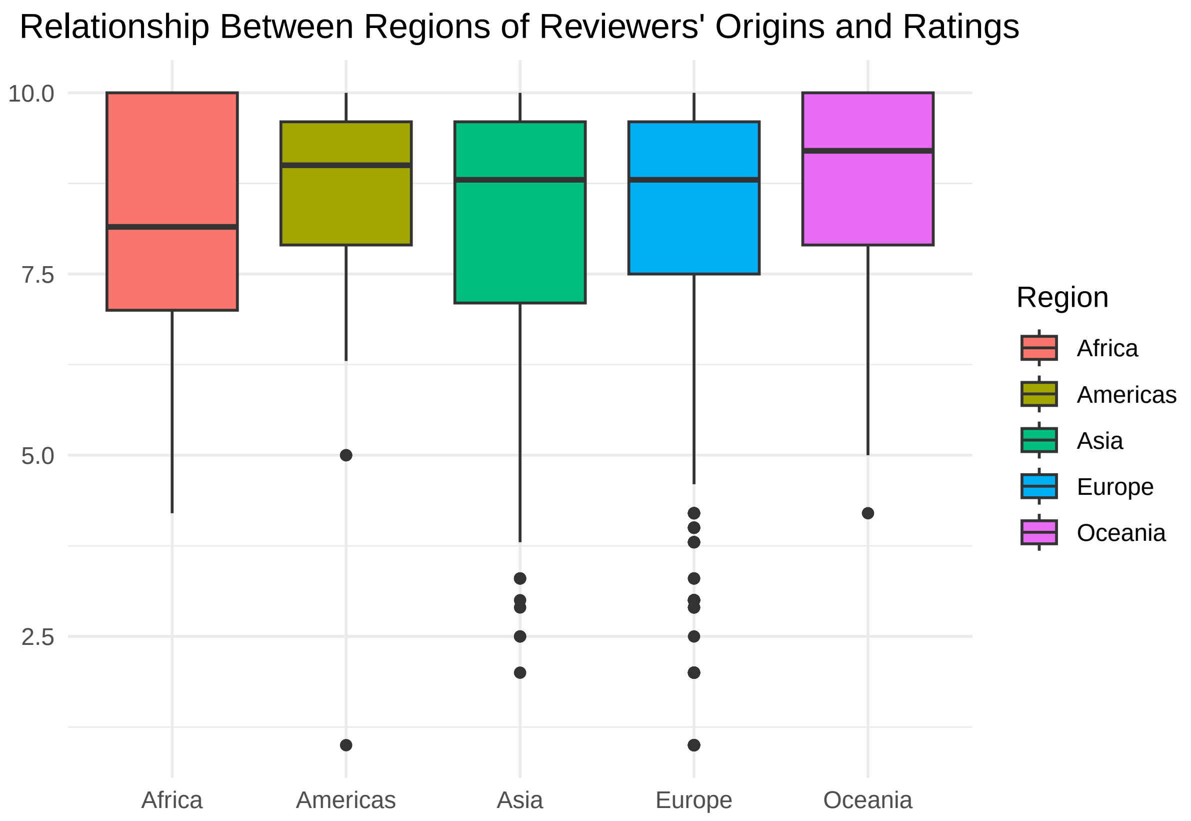
Problem 2

```r
# Plot the histogram of ratings
ggplot(group_5_hotels, aes(x = rating)) +
  geom_histogram(binwidth = 1, color = "white") +
  labs(title = "Distribution of Ratings",
       x = "",
       y = "Count") +
  theme_minimal() +  theme(plot.title = element_text(hjust = 0.5))
```



The distribution of all ratings is very left-skewed meaning there are more positive feedback about hotels - the most frequent grades are from 9 to 10.

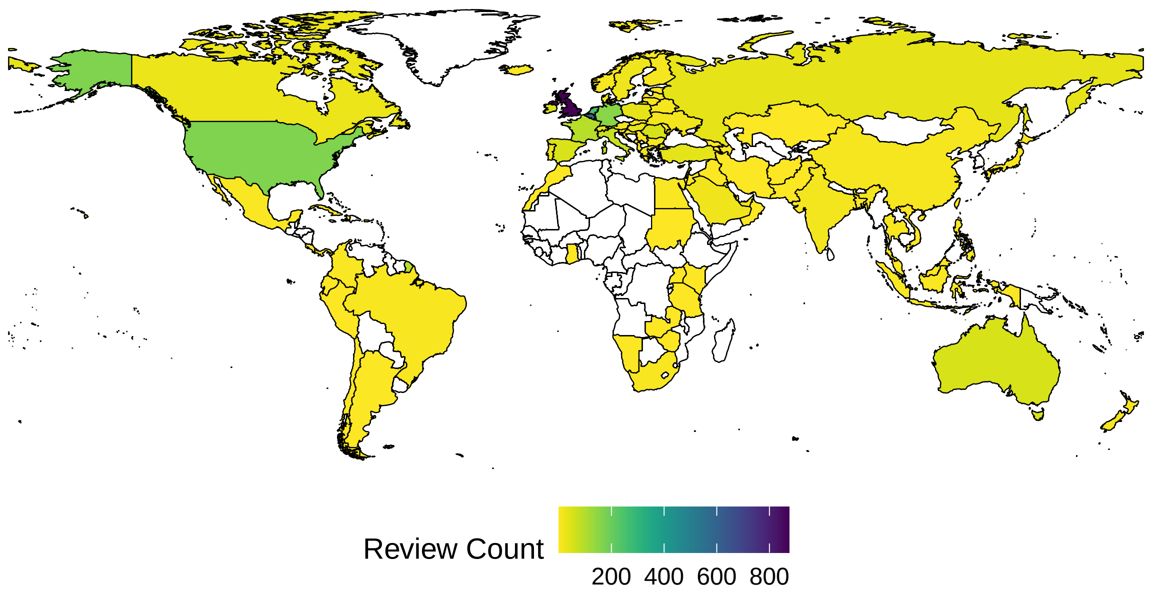
```r
# Boxplot of ratings by region
ggplot(na.omit(merged_data[, c("rating", "region")]),
       aes(x = region, y = rating, fill = region)) +
  geom_boxplot() +
  labs(title = "Relationship Between Regions of Reviewers' Origins and Ratings",
       x = "",
       y = "",
       fill = "Region") +
  theme_minimal() + theme(plot.title = element_text(hjust = 0.5))
```



The boxplot visually summarizes hotel ratings across the regions. They all have median ratings above 7.5, indicating generally positive ratings (it was also shown in the previous plot). Oceania shows the highest median ratings, so we may make a conclusion people with Oceanian origins are more satisfied with the hotels. Median of ratings from African reviewers is lower compared to others suggesting Africans are more strict rating hotels or they may face discrimination based on their race.

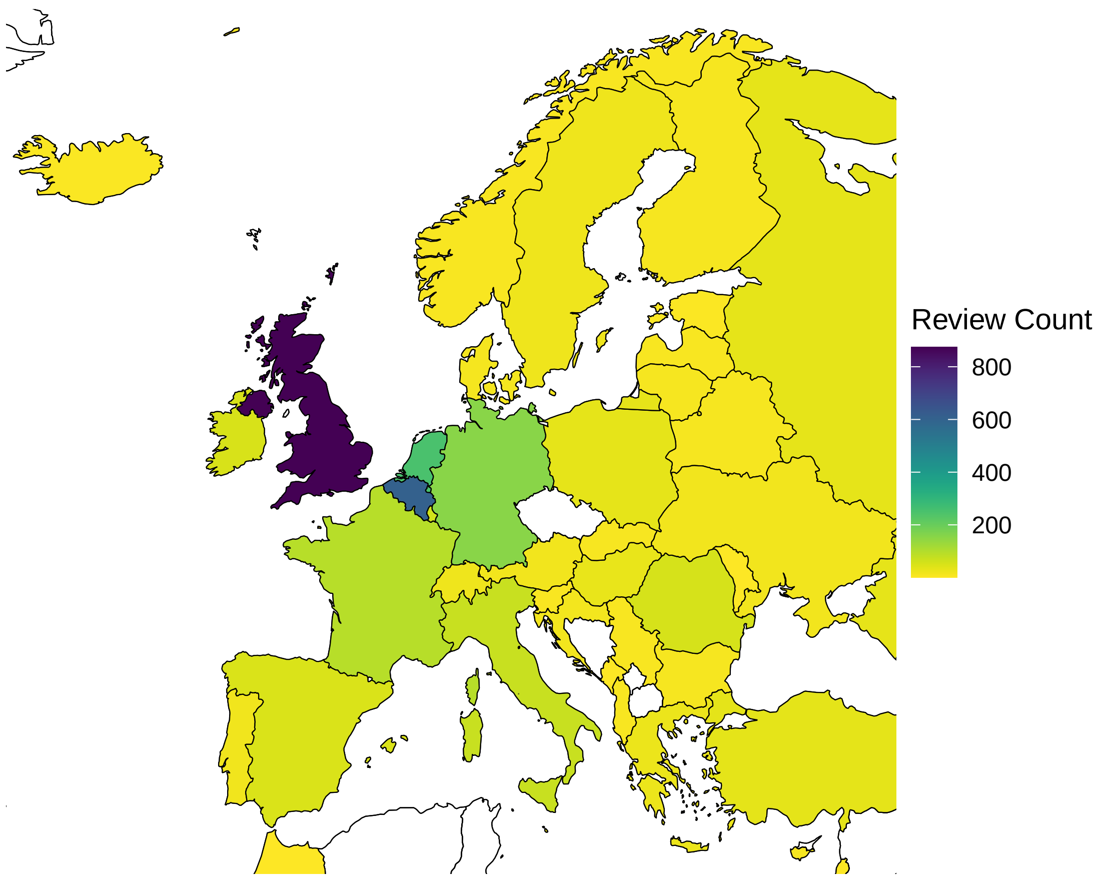
After testing who is “kinder” when giving reviews to the hotels we are checking people of which nationality gave most reviews.

```r
# Create the map plot
ggplot() +
  geom_sf(data = world, fill = "white", color = "black") + # Plot all countries in white
  geom_sf(data = world_with_reviews, aes(fill = review_count), color = "black") + # Overlay countries w
  scale_fill_viridis(direction = -1, na.value = NA, option = "D") + # Use reversed viridis palette, wit
  theme_void() +
  theme(legend.position = "bottom") + # Move legend to bottom
  labs(fill = "Review Count")
```



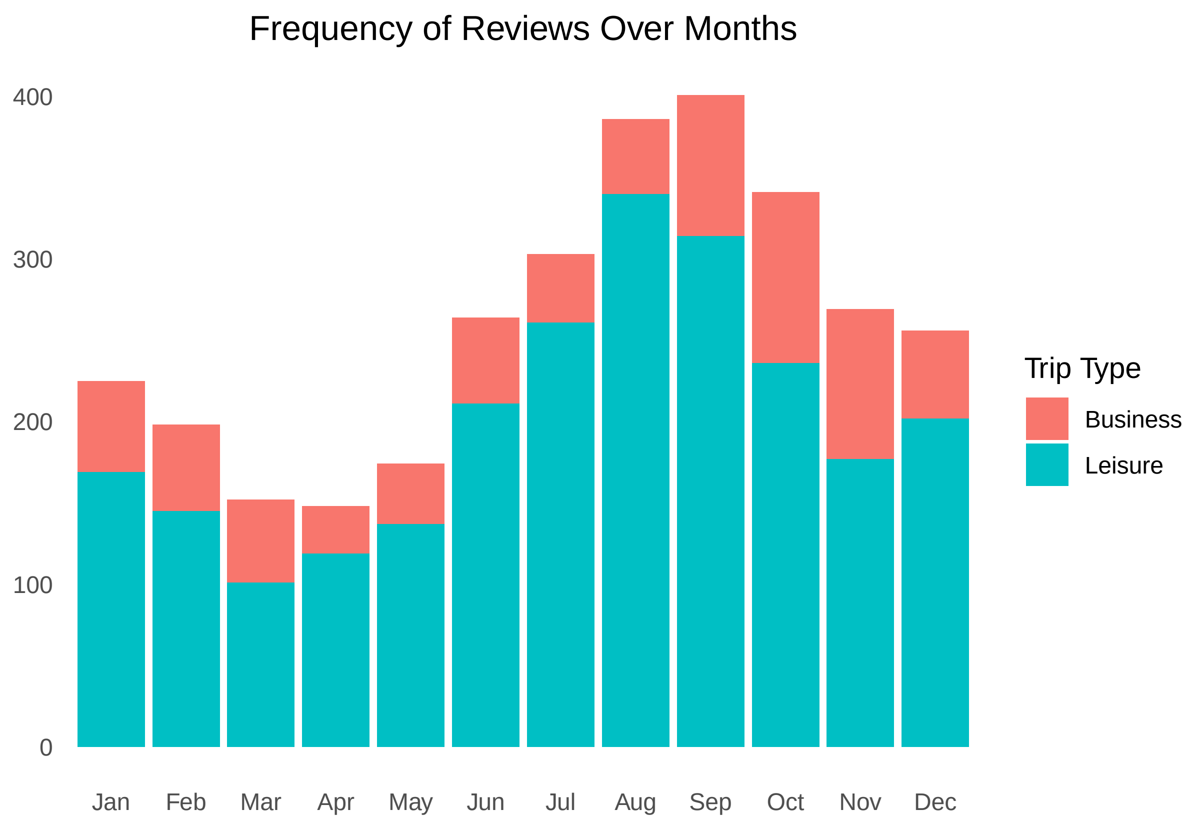
We see that most of the reviewers have European origins. Let’s zoom in to see in more details from which European countries there are more reviewers.

```r
xlims <- c(-25, 40) # Adjust these limits for longitude as needed
ylims <- c(34, 72) # Adjust these limits for latitude as needed

# Create the map plot focused on Europe
ggplot() +
  geom_sf(data = world, fill = "white", color = "black") +
  geom_sf(data = world_with_reviews, aes(fill = review_count), color = "black") +
  scale_fill_viridis(direction = -1, na.value = NA, option = "D") +
  theme_void() +
  theme() +
  labs(fill = "Review Count") +
  coord_sf(xlim = xlims, ylim = ylims, expand = FALSE) # Zoom in on Europe
```



There are over 800 Britanian reviewers! Compared to other nationalities it's a lot because most of the countries have about 200-300 reviewers in this dataset. The second most frequent nationality is Belgian. Others are relatively similar.

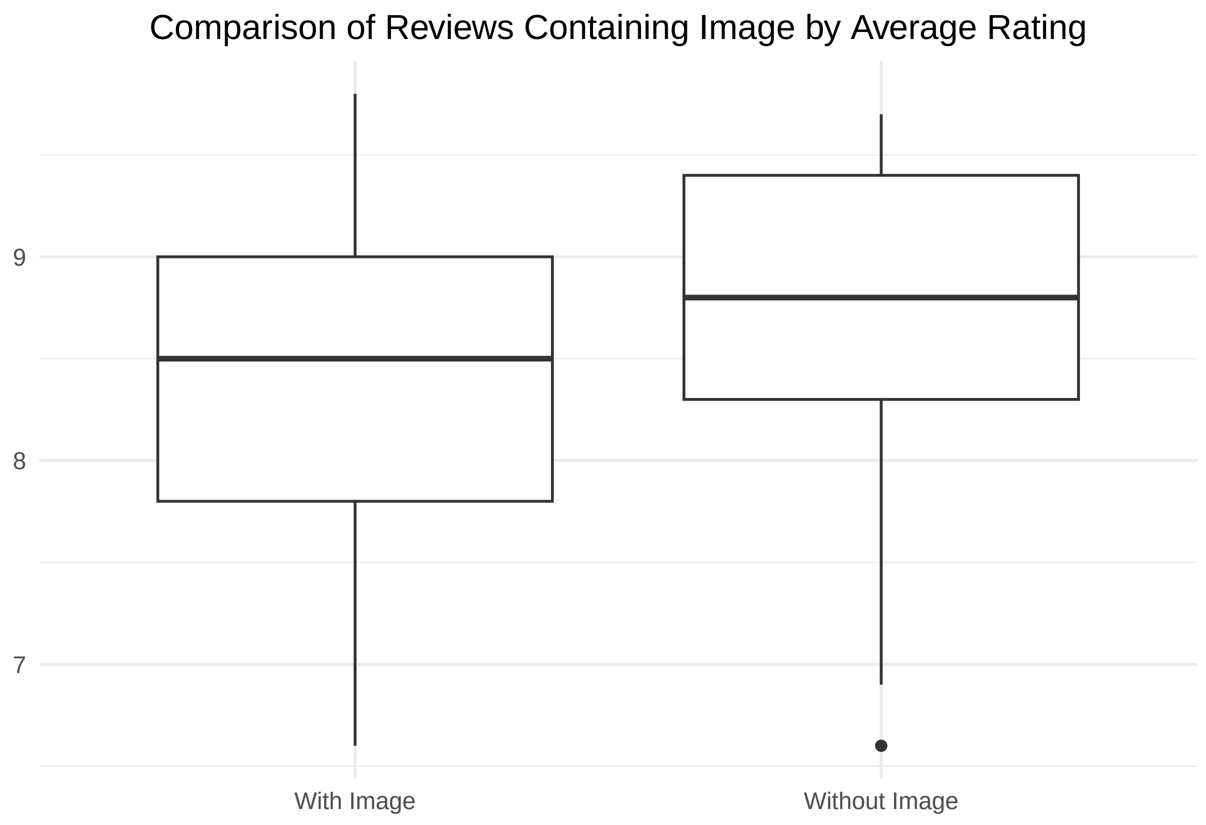
```r
ggplot(group_5_hotels[!is.na(group_5_hotels$trip_type), ],
       aes(x = factor(format(reviewed_at, "%b"), levels = month.abb),
           fill = trip_type)) +
  geom_bar(position = "stack") +
  labs(title = "Frequency of Reviews Over Months",
       x = "",
       y = "",
       fill = "Trip Type") +
  theme_minimal() +  theme(plot.title = element_text(hjust = 0.5), panel.grid.minor = element_blank(), 
```



We assumed that in summer there would be less business and more leisure trips because people take vacation in the season with better weather. In fact, surprisingly, in each month reviewers having leisure trip were significanlty more. Also, we see that in autumn there are significantly more business trips.

We also assumed that if a person includes a picture in their review, it may either mean they liked the hotel so much and wanted to share beautiful pictures from there or they are disappointed because of something (maybe the rooms were dirty, or something was broken) and wanted to share that with a photo. We are checking which hypothesis is true usinq boxplots.

```r
ggplot(group_5_hotels, aes(x = has_image, y = avg_rating)) +
  geom_boxplot() +
  labs(title = "Comparison of Reviews Containing Image by Average Rating",
       x = "",
       y = "") + theme_minimal() + theme() + theme(plot.title = element_text(hjust = 0.5))
```



People that icnluded images in their review gave higher ratings, according to the visualization. Let's also check the hypothesis statistically using linear regression:

```r
linear_model <- lm(avg_rating ~ has_image, data = group_5_hotels)
summary(linear_model)
```

```
##
## Call:
## lm(formula = avg_rating ~ has_image, data = group_5_hotels)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1156 -0.5462  0.1538  0.6538  1.4538
##
## Coefficients:
```

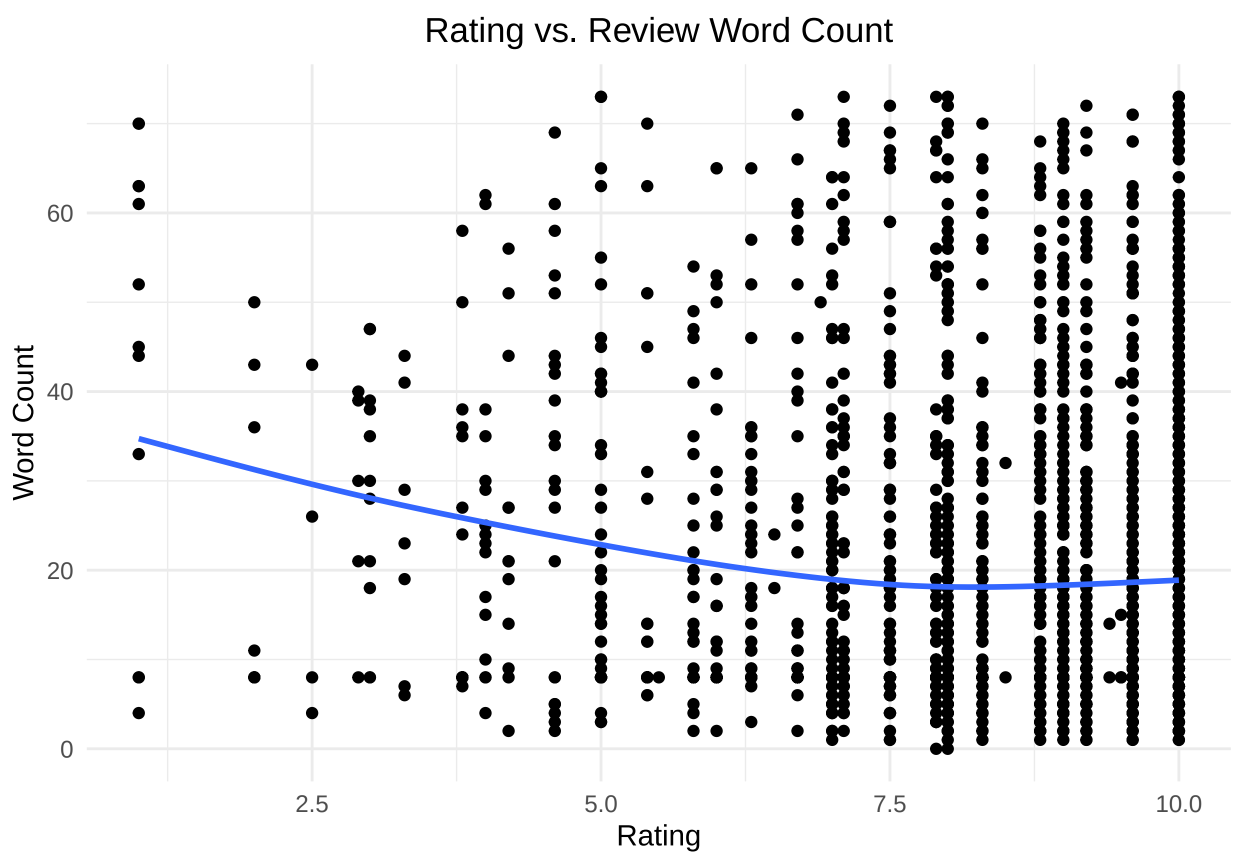
```
##                          Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                8.3462     0.0138 604.777  < 2e-16 ***
## has_imageWithout Image     0.3694     0.0892   4.141 3.55e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7733 on 3215 degrees of freedom
## Multiple R-squared:  0.005305,   Adjusted R-squared:  0.004996 
## F-statistic: 17.15 on 1 and 3215 DF,  p-value: 3.549e-05
```

P-value of “has_image” coefficient is very small meaning it’s statistically significant and we retain our hypothesis.

Next thing we decided to do is to consider the length of the review texts. We assumed that if a person wrote a lot, it’s possible they are complaining about some drawbacks, that’s why their review is long. We stated a hypothesis that people who gave longer reviews rated hotels lower.

```r
# Create the plot with the filtered data
ggplot(group_5_hotels_filtered, aes(y = word_count, x = rating)) +
  geom_point() +
  labs(title = "Rating vs. Review Word Count",
       y = "Word Count",
       x = "Rating") +
  geom_smooth(se = FALSE) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



As we can see, indeed, reviews with lower ratings contain more words than reviews with higher ratings. The fitted lined shows that trend.

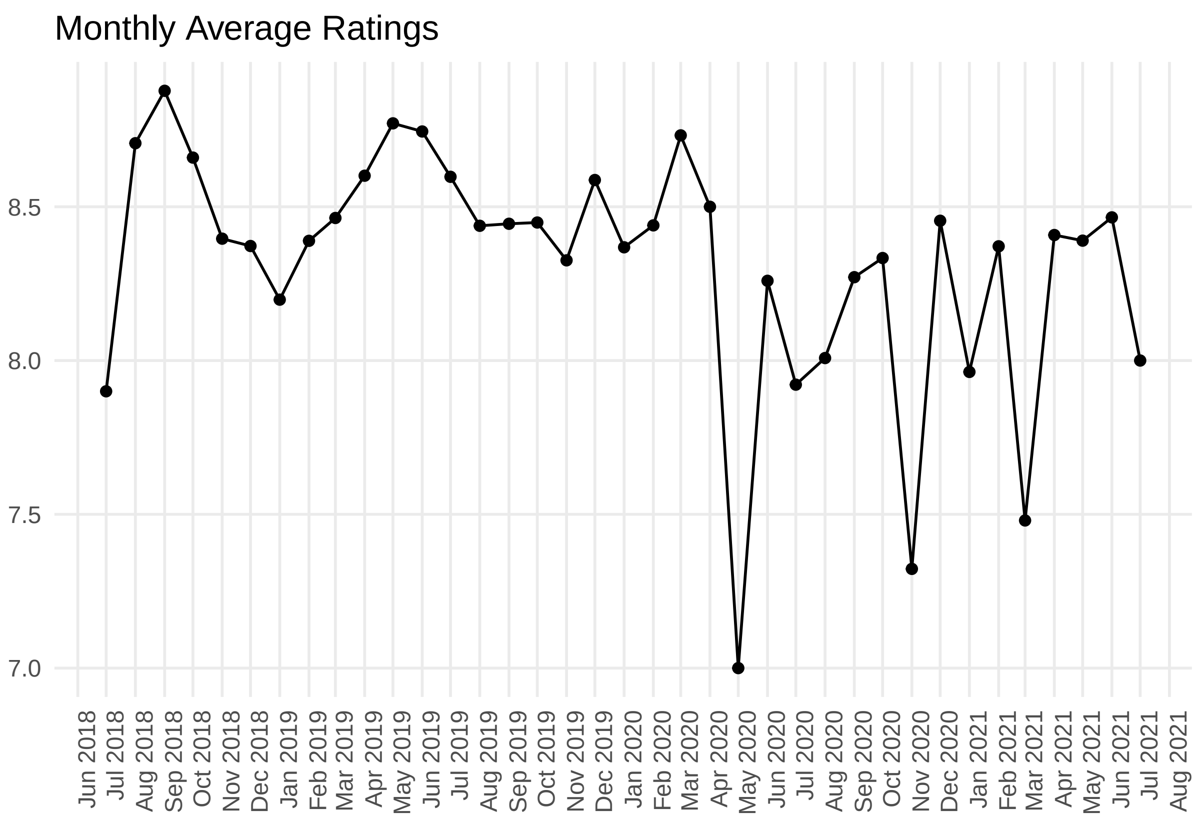
```r
linear_model <- lm(word_count ~ rating, data = group_5_hotels_filtered)
summary(linear_model)
```

```
##
## Call:
## lm(formula = word_count ~ rating, data = group_5_hotels_filtered)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -21.456 -11.503  -9.802   7.857  55.198
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  26.3063     1.6471  15.971  < 2e-16 ***
## rating       -0.8504     0.1901  -4.474 7.96e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.79 on 2947 degrees of freedom
## Multiple R-squared:  0.006747,   Adjusted R-squared:  0.00641
## F-statistic: 20.02 on 1 and 2947 DF,  p-value: 7.959e-06
```

Statistical analysis confirmed that there is such correlation: more words mean worse rating.

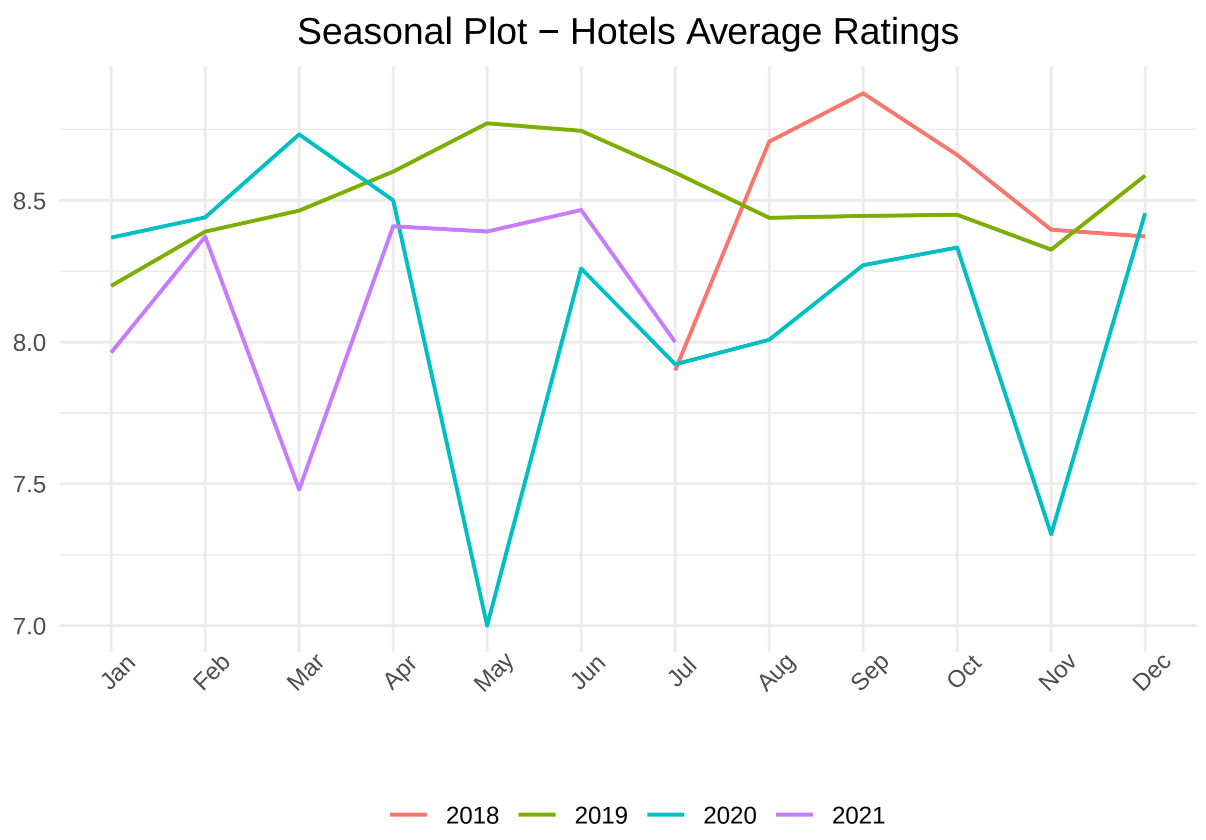
Time series analysis

```r
# Plot the time series of average ratings per month
ggplot(monthly_ratings_complete, aes(x = year_month, y = avg_rating)) +
  geom_line(na.rm = TRUE) +  # Draw lines
  geom_point(na.rm = TRUE) +  # Add points
  scale_x_date(date_breaks = "1 month", date_labels = "%b %Y") +
  # Set breaks to 1 month
  theme_minimal() +
  labs(title = "Monthly Average Ratings",
       x = "",
       y = "") +
  # Rotate x-axis labels for better readability
  theme(axis.text.x = element_text(angle = 90, hjust = 1), panel.grid.minor = element_blank(), )
```



This shows the actual observed data, which seems to be fluctuating around a value just below 8.5 and doesn't show any clear long-term upward or downward trend within the displayed time frame.

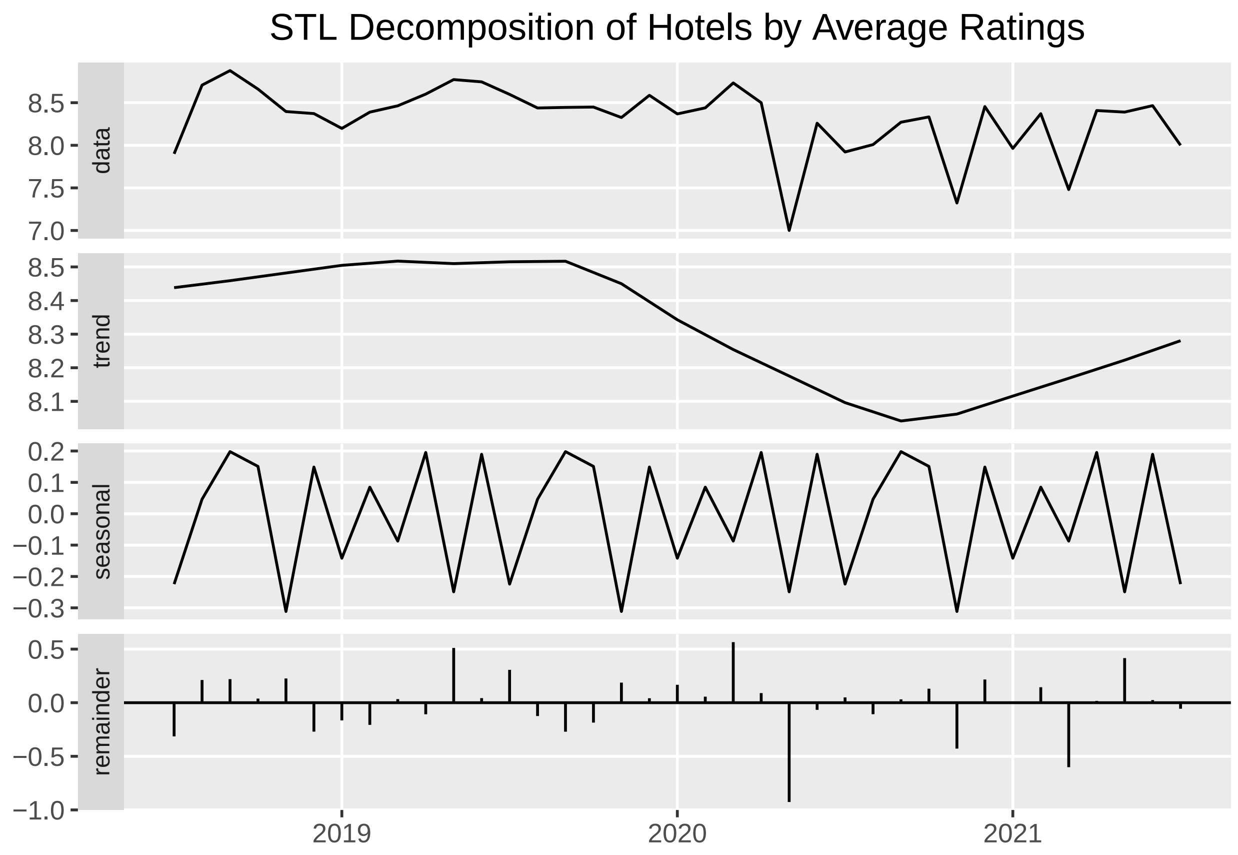
```r
ggseasonplot(monthly_ts, year.labels = F) +
  ggtitle("Seasonal Plot - Hotels Average Ratings") +
  xlab("") +
  ylab("") +
  theme(panel.grid.minor = element_blank(),
        plot.title = element_text(size = 14, hjust = 0.5)) +
  geom_line(linetype = "solid", size = 0.7) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, hjust = 0.5),
    axis.text.x = element_text(angle = 45, vjust = 1),
    legend.position = "bottom",
    legend.title = element_blank()
  )
```



We see that in 2018 and 2019 ratings were higher. In 2020, when COVID started, average ratings went lower, in May 2020 it was at its worst. It was the peak of the epidemic which may negatively affect people mood that's why they gave lower ratings and average ratings were that low. In 2021 the data is more stable excluding March. Overall, 5 years ago, average ratings were higher than then.

```
# Plot the decomposed object using autoplot
autoplot(decomposed) +
  ggtitle("STL Decomposition of Hotels by Average Ratings") +
  theme(
    plot.title = element_text(size = 14, hjust = 0.5),
    axis.title = element_blank(),
    axis.text = element_text(size = 10),
    panel.grid.minor = element_blank()  # Remove minor grid lines
  ) +
  xlim(c(start_time, max(time_range)))  # Set x-axis limits from July 2018 to the maximum of your time
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## Warning: Removed 4 rows containing missing values (`geom_rect()`).
```



There's a noticeable downward trend starting from early 2020. This could reflect a variety of factors, including a possible impact of external events (such as the COVID-19 pandemic) on hotel ratings.

There's a repeating pattern that seems to occur on a regular basis throughout the period. This seasonality could correspond to factors like holidays, vacation seasons, or other cyclical events that influence hotel ratings.