Data 624 Week2

Souleymane Doumbia

2024-09-22

Libraries

```
library(fpp3)
```

```
## Warning: package 'fpp3' was built under R version 4.3.3
```

```
## Registered S3 method overwritten by 'tsibble':
##   method               from
##   as_tibble.grouped_df dplyr
```

```
## -- Attaching packages --------------------------------------- fpp3 1.0.0 --
```

```
## v tibble      3.2.1     v tsibble     1.1.5
## v dplyr       1.1.3     v tsibbledata 0.4.1
## v tidyr       1.3.0     v feasts      0.3.2
## v lubridate   1.9.3     v fable       0.3.4
## v ggplot2     3.5.1     v fabletools  0.4.2
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
## Warning: package 'tsibble' was built under R version 4.3.3
```

```
## Warning: package 'feasts' was built under R version 4.3.2
```

```
## Warning: package 'fabletools' was built under R version 4.3.2
```

```
## Warning: package 'fable' was built under R version 4.3.2
```

```
## -- Conflicts ------------------------------------------------ fpp3_conflicts --
## x lubridate::date()    masks base::date()
## x dplyr::filter()      masks stats::filter()
## x tsibble::intersect() masks base::intersect()
## x tsibble::interval()  masks lubridate::interval()
## x dplyr::lag()         masks stats::lag()
## x tsibble::setdiff()   masks base::setdiff()
## x tsibble::union()     masks base::union()
```

```
library(plotly)
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(x13binary)
```

```
## Warning: package 'x13binary' was built under R version 4.3.3
```

```
library(seasonal)
```

```
## The system variable 'X13_PATH' has been manually set to:
##   /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/library/x13binary/bin/x13ashtml
## Since version 1.2, 'seasonal' relies on the 'x13binary'
## package and does not require 'X13_PATH' to be set anymore.
## Only set 'X13_PATH' manually if you intend to use your own
## binaries. See ?seasonal for details.
```

```
## Binary executable file /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/library/x13binar
## See ?seasonal for details.
```

```
##
## Attaching package: 'seasonal'
```

```
## The following object is masked from 'package:tibble':
##
##     view
```

Exercise 3.7.1

Loading the Dataset global_economy

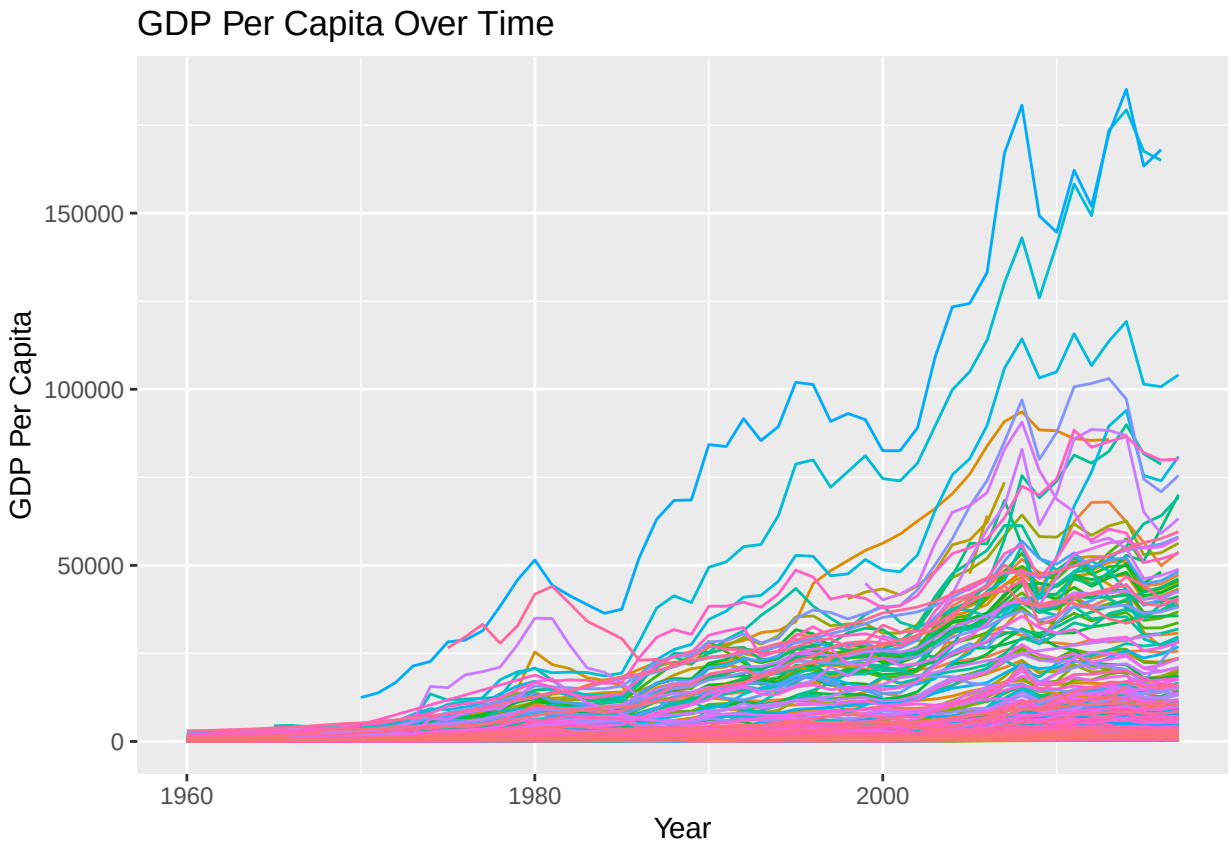
```
# View the structure of the dataset
head(global_economy)
```

```
## # A tsibble: 6 x 9 [1Y]
## # Key:       Country [1]
##   Country     Code   Year         GDP Growth   CPI Imports Exports Population
##   <fct>       <fct> <dbl>       <dbl>  <dbl> <dbl>   <dbl>   <dbl>      <dbl>
## 1 Afghanistan AFG    1960  537777811.     NA    NA    7.02    4.13    8996351
## 2 Afghanistan AFG    1961  548888896.     NA    NA    8.10    4.45    9166764
## 3 Afghanistan AFG    1962  546666678.     NA    NA    9.35    4.88    9345868
## 4 Afghanistan AFG    1963  751111191.     NA    NA   16.9     9.17    9533954
## 5 Afghanistan AFG    1964  800000044.     NA    NA   18.1     8.89    9731361
## 6 Afghanistan AFG    1965 1006666638.     NA    NA   21.4    11.3     9938414
```

Plotting GDP per Capita Over Time

```
# Filtering and create the plot using ggplot first
gdp_plot <- global_economy %>%
  filter(!is.na(GDP), !is.na(Population)) %>%
  mutate(GDP_per_capita = GDP / Population) %>%
  ggplot(aes(x = Year, y = GDP_per_capita, color = Country, text = Country)) +
  geom_line() +
  labs(title = "GDP Per Capita Over Time",
       x = "Year",
       y = "GDP Per Capita") +
  theme(legend.position = "none") # too many many can not be shown in the legend, then we delete it ins
```

```
# Converting ggplot to plotly for interactivity since there are more countries
#interactive_gdp_plot <- ggplotly(gdp_plot, tooltip = "text")

# Interactive plot
#interactive_gdp_plot
gdp_plot
```



Country with highest GDP per Capita

```
# Find the overall highest GDP per capita
highest_highest_gdp <- global_economy %>%
  filter(!is.na(GDP), !is.na(Population)) %>%
  mutate(GDP_per_capita = GDP / Population) %>%
  filter(GDP_per_capita == max(GDP_per_capita)) %>%
  select(Country, Year, GDP_per_capita)

highest_highest_gdp
```

```
## # A tsibble: 1 x 3 [1Y]
## # Key:       Country [1]
##   Country  Year GDP_per_capita
##   <fct>   <dbl>          <dbl>
## 1 Monaco   2014        185153.
```

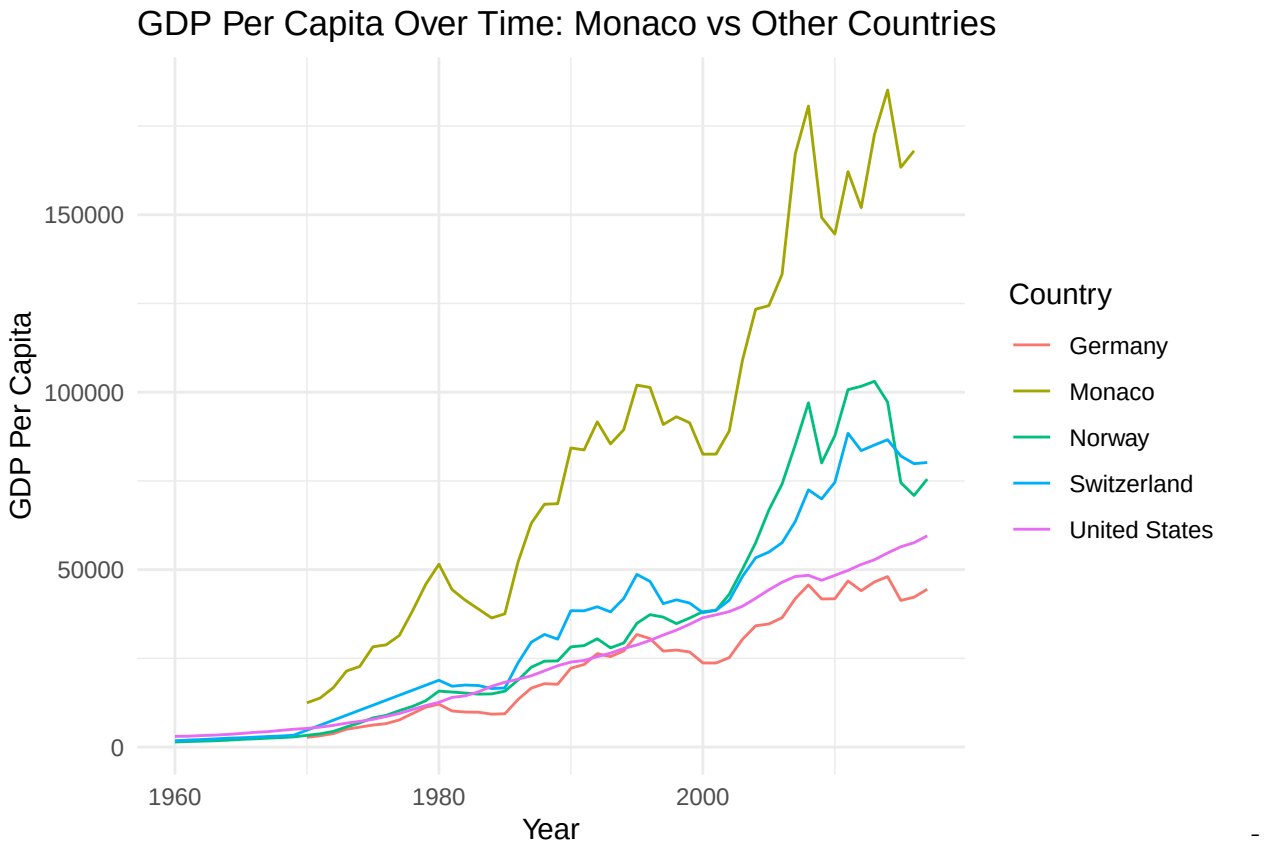
- Country with the highest `GDP per capital`: **Monaco in 2014**

How has Monaco's GDP per Capita changed over time

```
# Finding the highest GDP per capita each year to highlight Monaco's evolution
highest_gdp_per_capita_over_time <- global_economy %>%
  filter(!is.na(GDP), !is.na(Population)) %>%
  mutate(GDP_per_capita = GDP / Population) %>%
  index_by(Year) %>%
  filter(GDP_per_capita == max(GDP_per_capita)) %>%
  select(Country, Year, GDP_per_capita) %>%
  arrange(Year)

head(highest_gdp_per_capita_over_time,10)
```

```
## # A tsibble: 10 x 3 [1Y]
## # Key:       Country [2]
## # Groups:    @ Year [10]
##    Country        Year GDP_per_capita
##    <fct>         <dbl>          <dbl>
##  1 United States  1960          3007.
##  2 United States  1961          3067.
##  3 United States  1962          3244.
##  4 United States  1963          3375.
##  5 United States  1964          3574.
##  6 Kuwait         1965          4429.
##  7 Kuwait         1966          4556.
##  8 United States  1967          4336.
##  9 United States  1968          4696.
## 10 United States  1969          5032.
```

```
# Subsetting for Monaco and some other countries for comparison
countries_to_compare <- c("Monaco", "United States", "Germany", "Switzerland", "Norway")

gdp_comparison_plot <- global_economy %>%
  filter(Country %in% countries_to_compare & !is.na(GDP) & !is.na(Population)) %>%
  mutate(GDP_per_capita = GDP / Population) %>%
  ggplot(aes(x = Year, y = GDP_per_capita, color = Country)) +
  geom_line() +
  labs(title = "GDP Per Capita Over Time: Monaco vs Other Countries",
       x = "Year",
       y = "GDP Per Capita") +
  theme_minimal()

gdp_comparison_plot
```



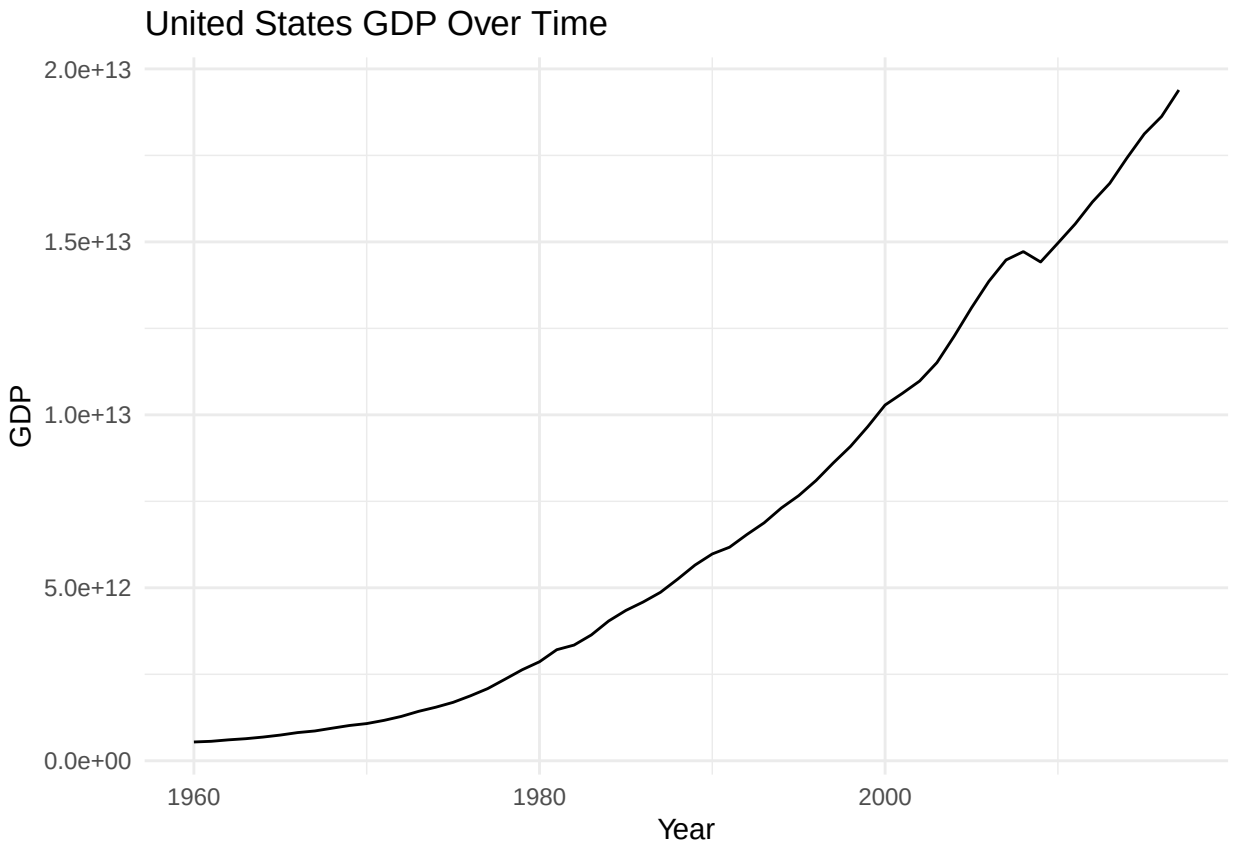
Since the 1970s, Monaco's GDP per capita has grown significantly compared to other top countries.

- Between 1970 and 1990, Monaco's GDP per capita was similar to countries like Norway and Switzerland.
- From the 1990s onward, Monaco experienced a surge in GDP per capita, far surpassing these other nations.
- By the early 2000s, Monaco became the clear leader, with GDP per capita exceeding 150,000, driven by its unique economic structure and rapid growth.

Exercise 3.7.2

United States GDP from global_economy:

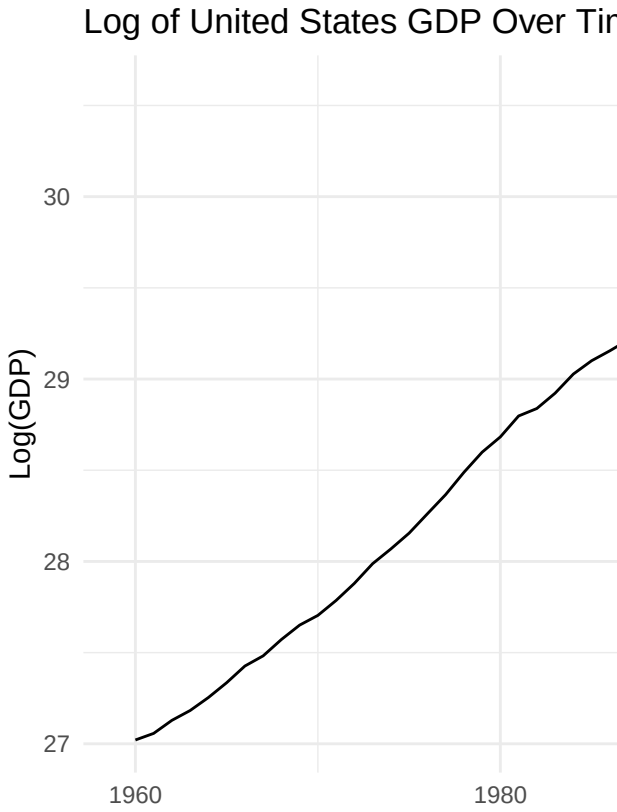
```
us_gdp <- global_economy %>%
  filter(Country == "United States") %>%
  select(Year, GDP)

us_gdp_plot <- us_gdp %>%
  autoplot(GDP) +
  labs(title = "United States GDP Over Time",
       x = "Year",
       y = "GDP") +
  theme_minimal()

us_gdp_plot
```



Comment:

- The United States GDP exhibits exponential growth over time. The upward trend is continuous and steepens towards the 2000s.
- Transformation: Applying a log transformation to stabilize the variance and make the trend clearer.

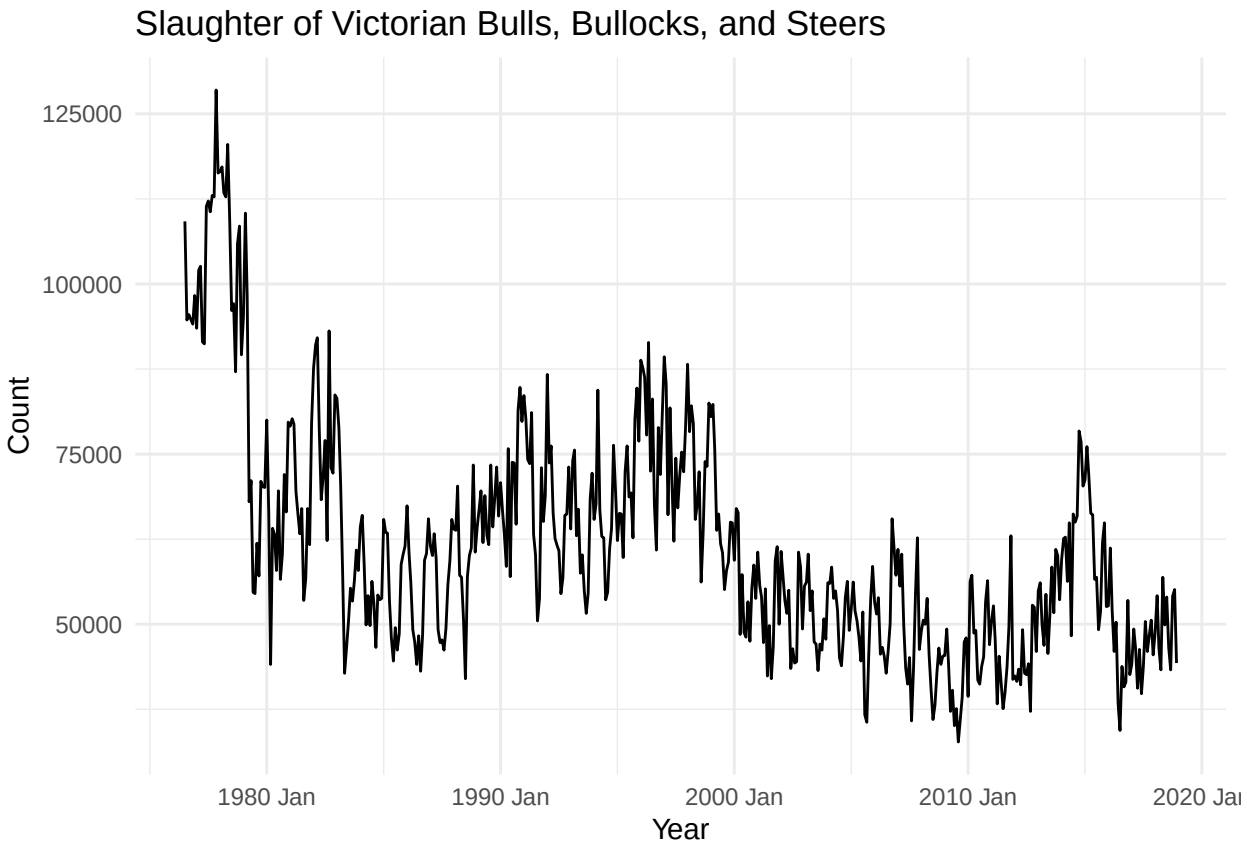
```
us_gdp %>%
  mutate(log_GDP = log(GDP)) %>%
  autoplot(log_GDP) +
  labs(title = "Log of United States GDP Over Time",
       x = "Year",
       y = "Log(GDP)") +
  theme_minimal()
```



Log transformation United States GDP from global_economy:
Effect: The log transformation smooths the rapid growth, making the trend more linear and easier to analyze.

Slaughter of Victorian "Bulls, bullocks and steers" in aus_livestock

```
# Victorian Bulls, Bullocks, and Steers
victoria_livestock <- aus_livestock %>%
  filter(State == "Victoria", Animal == "Bulls, bullocks and steers")

victoria_livestock_plot <- victoria_livestock %>%
  autoplot(Count) +
  labs(title = "Slaughter of Victorian Bulls, Bullocks, and Steers",
       x = "Year",
       y = "Count") +
  theme_minimal()

victoria_livestock_plot
```



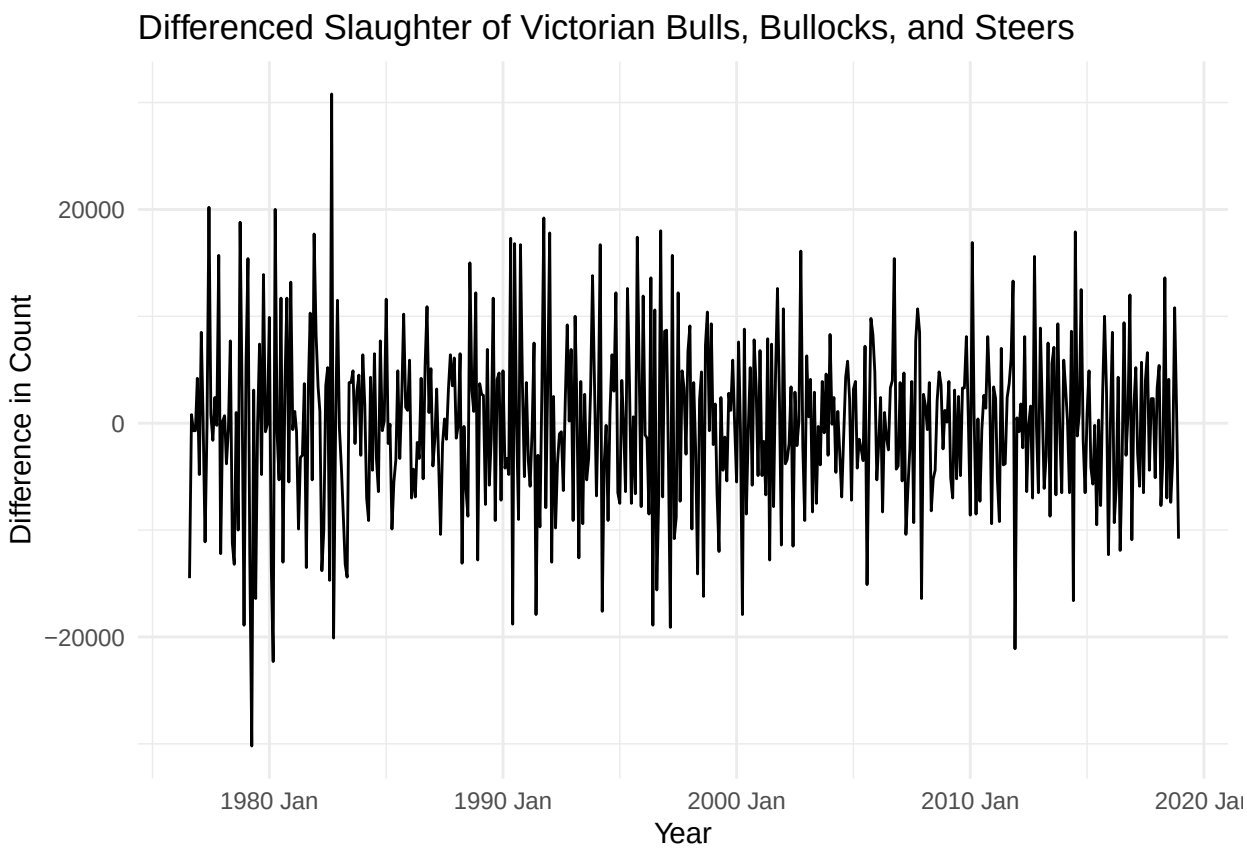
Comment:

- The data shows seasonality or fluctuations based on demand for livestock. The count is high in the 1970s but shows a downward trend with some seasonal variation afterward.
- Transformation: Since seasonality is present, let's consider differencing or using seasonal decomposition.

```
# Differencing the livestock count
victoria_livestock %>%
  mutate(Difference = difference(Count)) %>%
  autoplot(Difference) +
  labs(title = "Differenced Slaughter of Victorian Bulls, Bullocks, and Steers",
       x = "Year",
       y = "Difference in Count") +
  theme_minimal()
```

Differencing to remove trend: Slaughter of Victorian "Bulls, bullocks and steers" in aus_livestock

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```

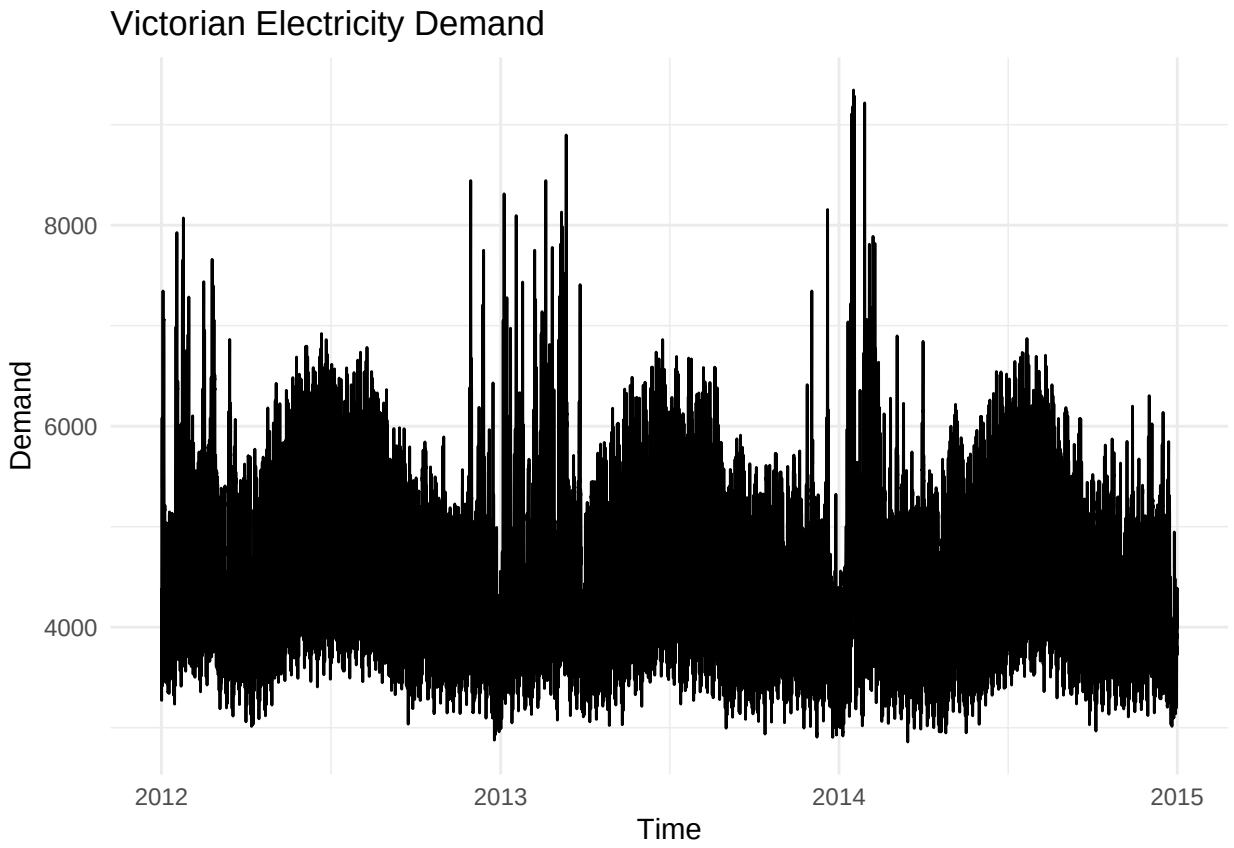


Effect:

- Differencing removes the long-term trend, which then highlight any underlying seasonality or cycles.

Victorian Electricity Demand from vic_elec:

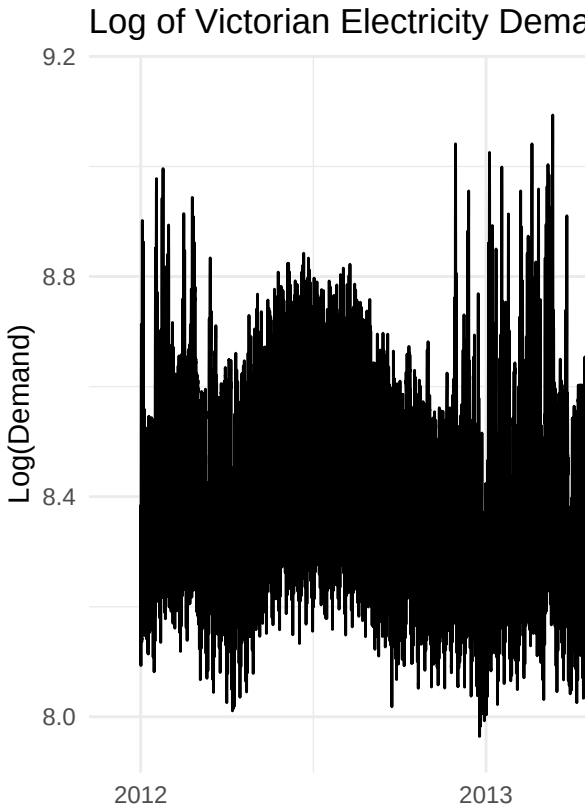
```
# Victorian Electricity Demand
vic_elec_demand <- vic_elec %>%
  select(Time, Demand)

vic_elec_plot <- vic_elec_demand %>%
  autoplot(Demand) +
  labs(title = "Victorian Electricity Demand",
       x = "Time",
       y = "Demand") +
  theme_minimal()

vic_elec_plot
```



Comment:

- Electricity demand is showing seasonal patterns. Demand tends to spike during certain times of the year, reflecting periodic behavior likely influenced by seasonal factors (weather conditions for example)
- Transformation: Seasonal decomposition or applying a log transformation to stabilize the variance.

```
# Log transformation of electricity demand
vic_elec_demand %>%
  mutate(log_Demand = log(Demand)) %>%
  autoplot(log_Demand) +
  labs(title = "Log of Victorian Electricity Demand",
       x = "Time",
       y = "Log(Demand)") +
  theme_minimal()
```

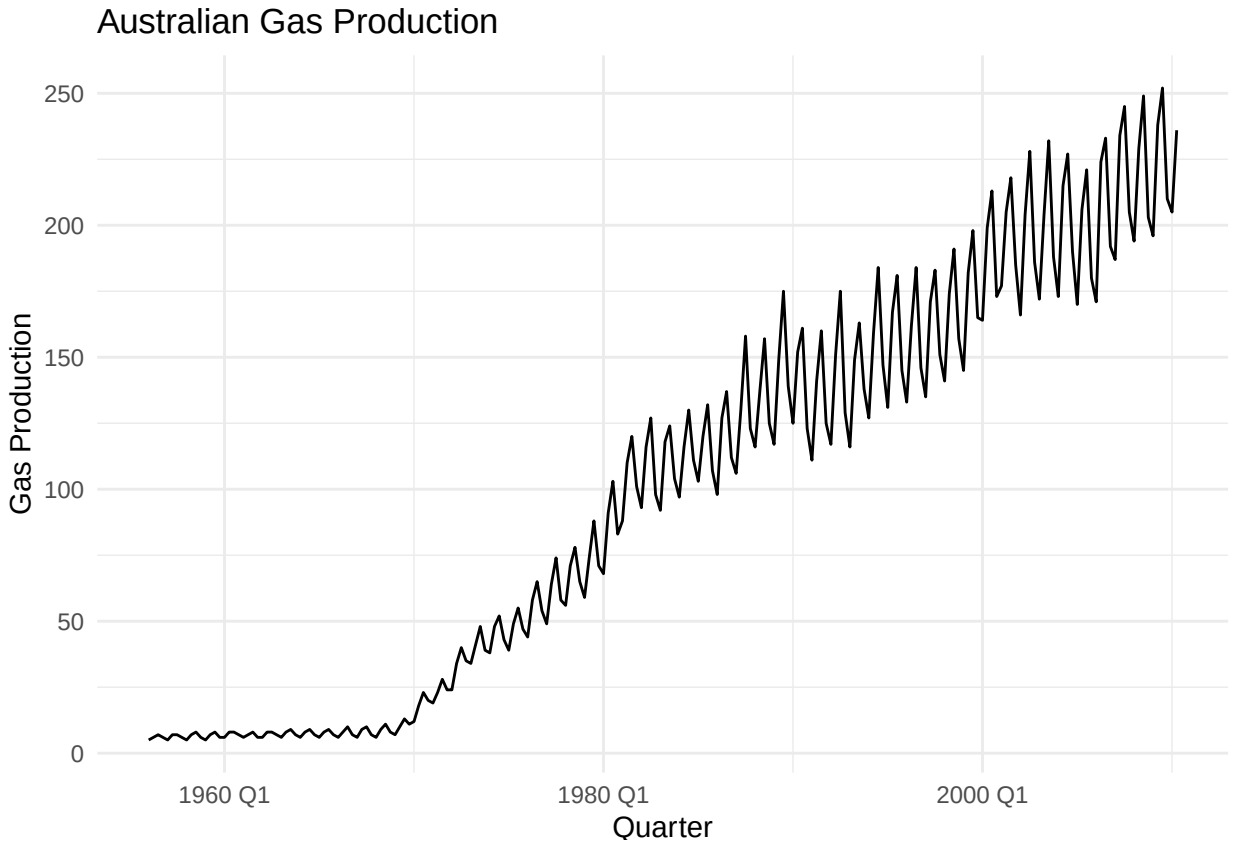


Log transformation: Victorian Electricity Demand from vic_elec
Effect:

- The log transformation is smoothing the demand and stabilizing the variance,which is making the pattern easier to observe.

Gas production from aus_production:

```
# Gas production
gas_production <- aus_production %>%
  select(Quarter, Gas)

gas_production_plot <- gas_production %>%
  autoplot(Gas) +
  labs(title = "Australian Gas Production",
       x = "Quarter",
       y = "Gas Production") +
  theme_minimal()

gas_production_plot
```



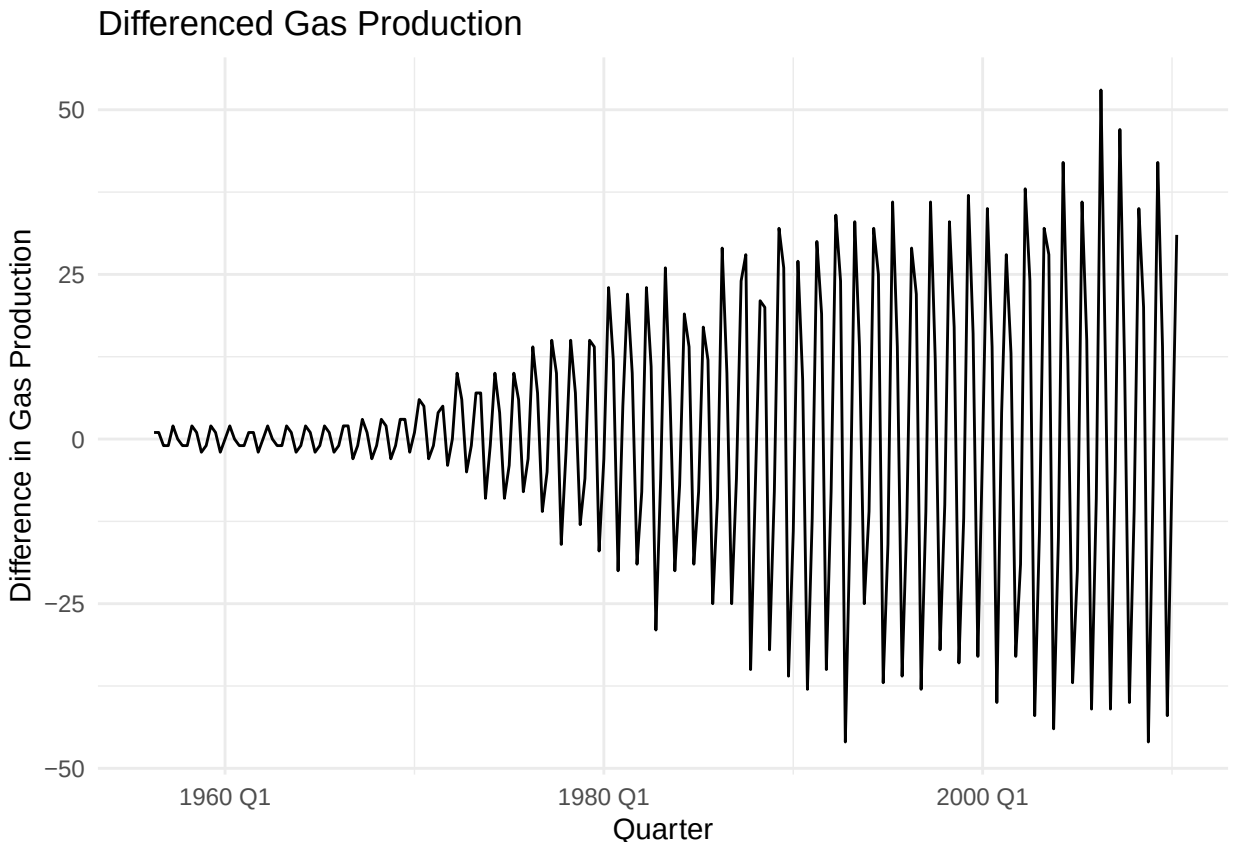
Comment:

- The graph illustrates a steady upward trend in Australian gas production, with increasing variability over time. There is seasonality, with periodic fluctuations that become more pronounced as production levels rise.
- Transformation: Let's consider differencing to remove the trend and observe seasonality.

```
gas_production %>%
  mutate(Difference = difference(Gas)) %>%
  autoplot(Difference) +
  labs(title = "Differenced Gas Production",
       x = "Quarter",
       y = "Difference in Gas Production") +
  theme_minimal()
```

Differencing to remove trend: Gas production from aus_production

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```



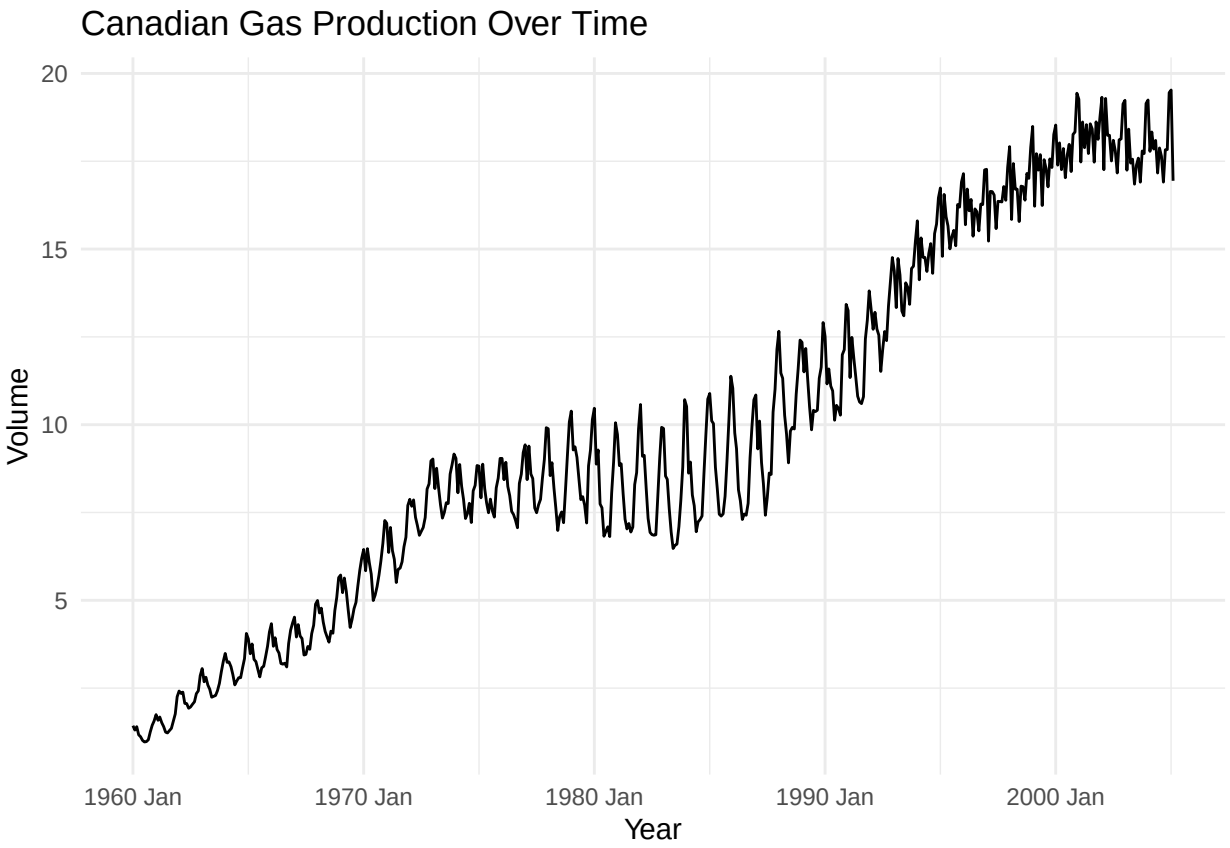
Effect:

- The differencing transformation effectively removes the trend which is revealing regular cycles.

Exercise 3.7.3

Visualizing the Canadian Gas data:

```
canadian_gas_plot <- canadian_gas %>%
  autoplot(Volume) +
  labs(title = "Canadian Gas Production Over Time",
       x = "Year",
       y = "Volume") +
  theme_minimal()

canadian_gas_plot
```



Original Data (Canadian Gas Production Over Time):

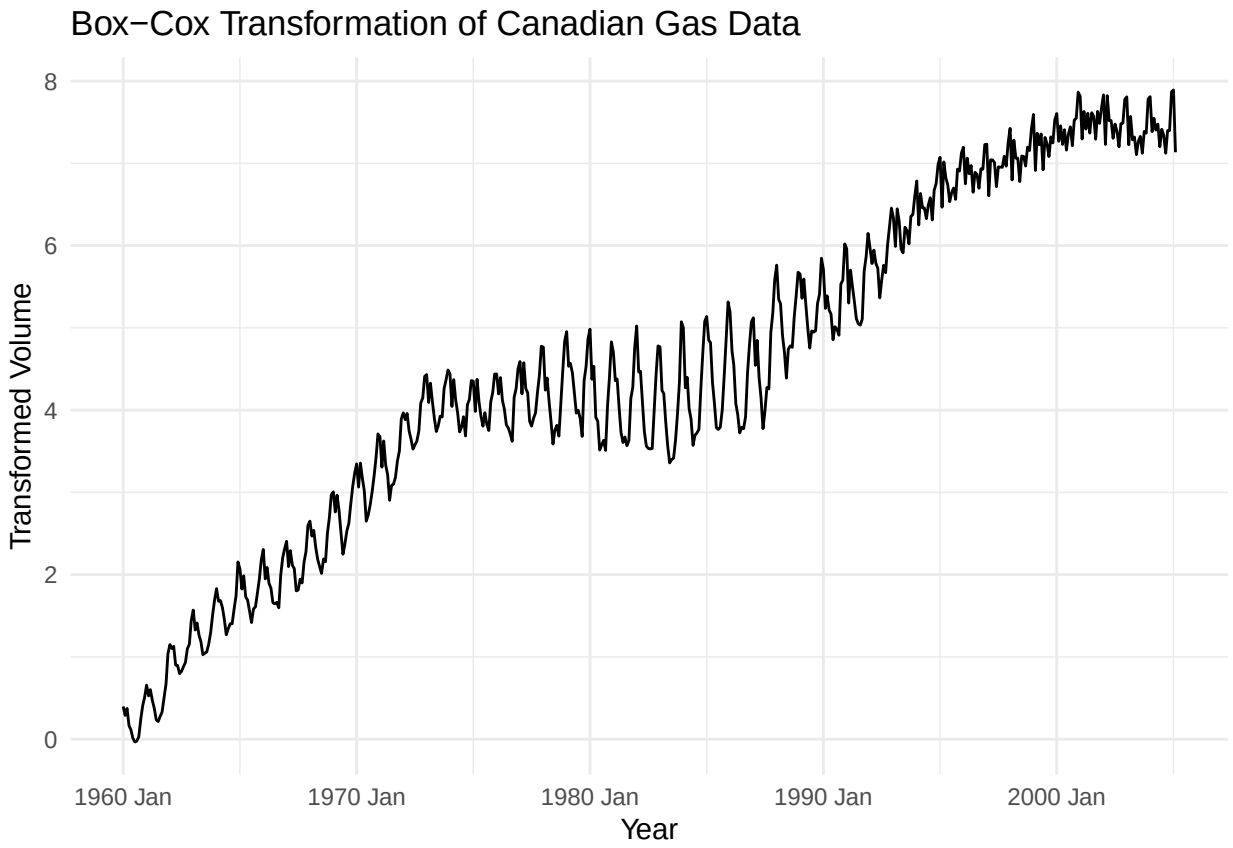
- The original data shows a clear upward trend and seasonal fluctuations but no major changes in variance over time. The pattern is relatively smooth, and the variance (height of seasonal peaks and troughs) appears stable.

Applying Box-Cox transformation: Canadian Gas data

```
# Estimating the optimal lambda value for Box-Cox transformation
lambda_opt <- canadian_gas %>%
  features(Volume, features = guerrero) %>%
  pull(lambda_guerrero)

lambda_opt  # Optimal lambda value
```

```
## [1] 0.5767648
```

```
# Applying Box-Cox transformation with the Optimal lambda value above
canadian_gas %>%
  mutate(Transformed_Volume = box_cox(Volume, lambda_opt)) %>%
  autoplot(Transformed_Volume) +
  labs(title = "Box-Cox Transformation of Canadian Gas Data",
       x = "Year",
       y = "Transformed Volume") +
  theme_minimal()
```

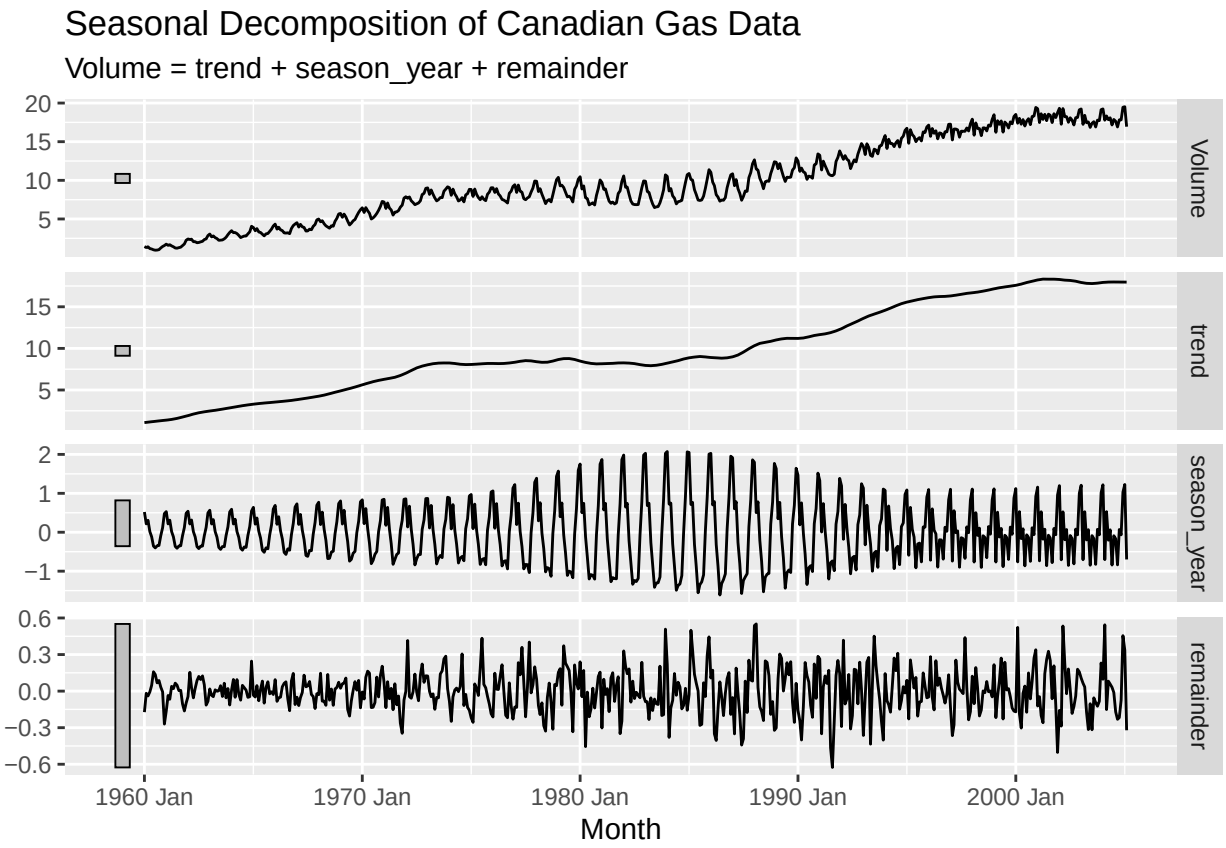


Box-Cox Transformation (Box-Cox Transformed Data)

- After applying the Box-Cox transformation, the data still shows a similar trend and seasonality. There is no substantial change in the structure of the data, and the transformation does not provide any additional clarity or benefits.

Checking seasonality and trends:

```
# Seasonal decomposition of Canadian Gas data
canadian_gas %>%
  model(STL(Volume)) %>%
  components() %>%
  autoplot() +
  labs(title = "Seasonal Decomposition of Canadian Gas Data")
```



Seasonal Decomposition (Decomposition Plot):

- The seasonal decomposition reveals that the trend and seasonality are the dominant features of the data, and the remainder (residuals) is relatively small and stable. This suggests that the primary issue with the data is the presence of seasonality and trend, not variance instability.

Conclusion

A Box-Cox transformation is unhelpful for the canadian_gas data because:

- The data already has stable variance, so the transformation does not make a significant difference.
- The key features of the data (the trend and seasonality) are better handled by other techniques like differencing or seasonal decomposition rather than a Box-Cox transformation, which mainly addresses variance issues.

Exercise 3.7.4

Estimate the optimal lambda for the Box-Cox transformation using the Guerrero method:

```
set.seed(23432)
myseries <- aus_retail %>%
  filter(`Series ID` == sample(aus_retail$`Series ID`,1))

# Estimating the optimal lambda using Guerrero method
lambda_opt <- myseries %>%
  features(Turnover, features = guerrero) %>%
  pull(lambda_guerrero)

print("Optimal Lambda Value is:"); lambda_opt
```

```
## [1] "Optimal Lambda Value is:"
```

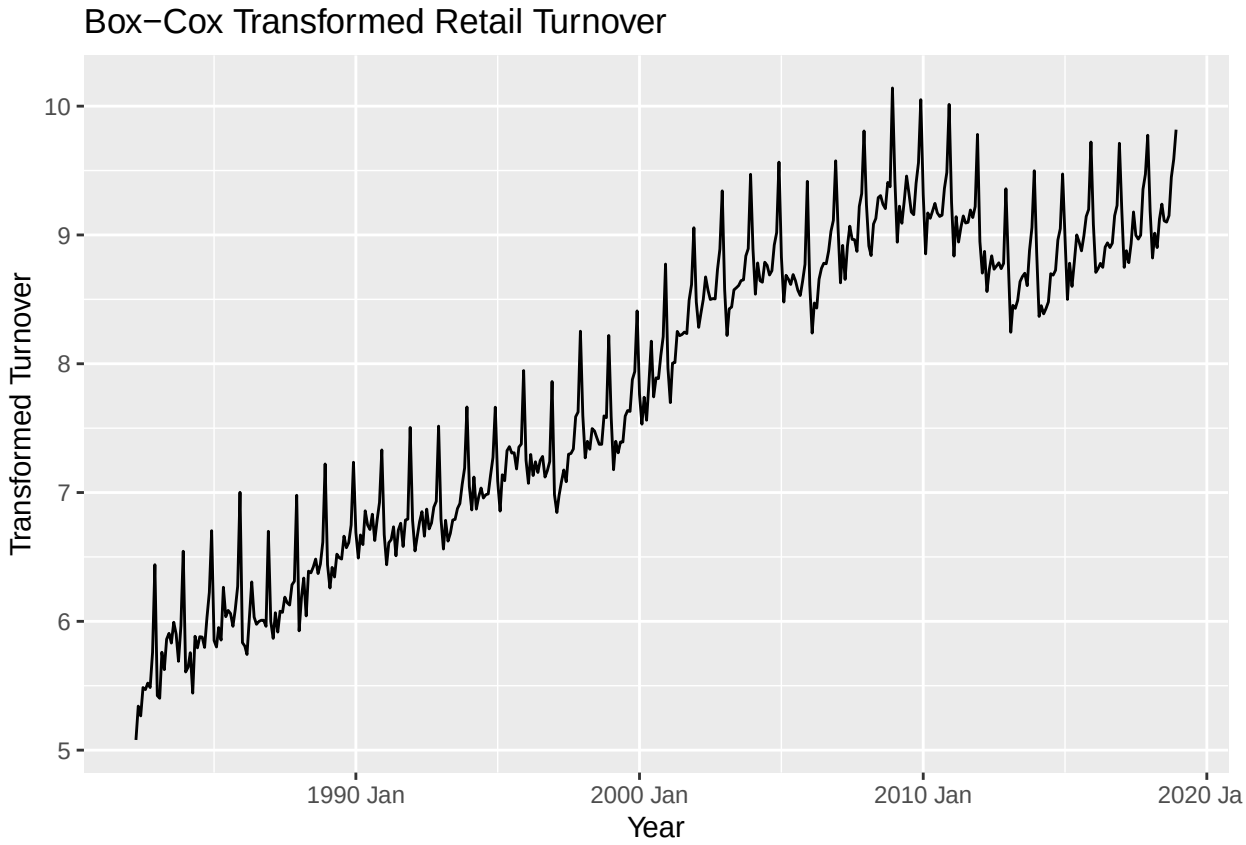
```
## [1] 0.1663741
```

- Based on the output from this code, we will select the optimal lambda value, which is in this case *lambda_opt*, for the Box-Cox transformation.

Applying the Box-Cox Transformation:

Now that we have the optimal lambda, we can apply the Box-Cox transformation:

```
myseries %>%
  mutate(Transformed_Turnover = box_cox(Turnover, lambda_opt)) %>%
  autoplot(Transformed_Turnover) +
  labs(title = "Box-Cox Transformed Retail Turnover",
       x = "Year",
       y = "Transformed Turnover")
```



Exercise 3.7.5

For each series, we will:

- Estimate the optimal Box-Cox lambda using the Guerrero method.
- Apply the Box-Cox transformation based on the estimated lambda. Briefly interpret the results.

Tobacco from aus_production

```
lambda_tobacco <- aus_production %>%
  features(Tobacco, features = guerrero) %>%
  pull(lambda_guerrero)

print('Optimum Value Lambda Tobacco:');lambda_tobacco
```

Estimating Optimal Lambda

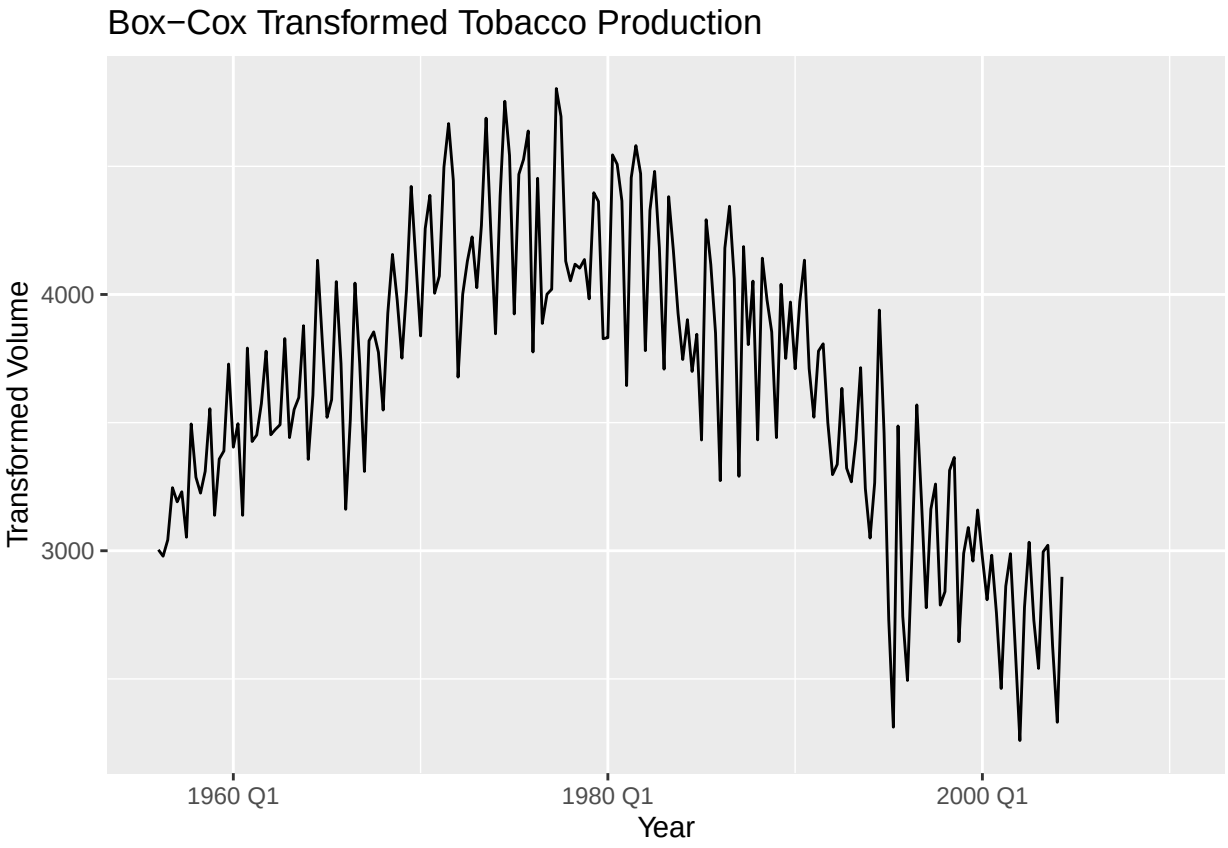
```
## [1] "Optimum Value Lambda Tobacco:"
```

```
## [1] 0.9264636
```

```
aus_production %>%
  mutate(Transformed_Tobacco = box_cox(Tobacco, lambda_tobacco)) %>%
  autoplot(Transformed_Tobacco) +
  labs(title = "Box-Cox Transformed Tobacco Production",
       x = "Year",
       y = "Transformed Volume")
```

Applying the Box-Cox Transformation: Tobacco from aus_production

```
## Warning: Removed 24 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



Economy Class Passengers from ansett

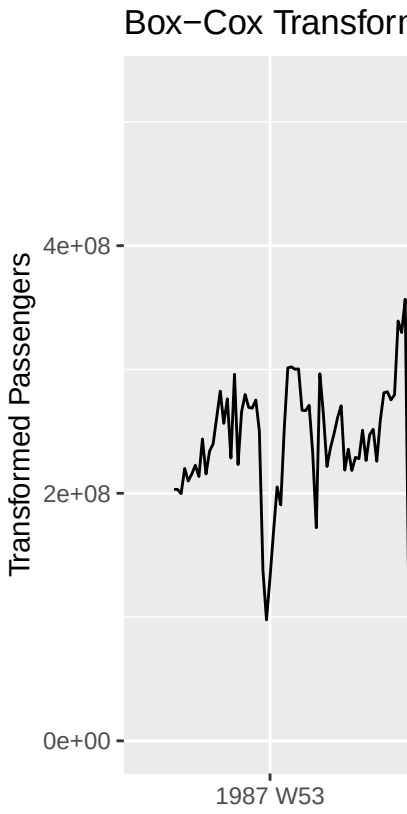
```
lambda_passengers <- ansett %>%
  filter(Class == "Economy", Airports == "MEL-SYD") %>%
  features(Passengers, features = guerrero) %>%
  pull(lambda_guerrero)

print('Optimum Value Lambda Passengers:');lambda_passengers
```

Estimate Optimal Lambda

```
## [1] "Optimum Value Lambda Passengers:"
```

```
## [1] 1.999927
```

```
ansett %>%
  filter(Class == "Economy", Airports == "MEL-SYD") %>%
  mutate(Transformed_Passengers = box_cox(Passengers, lambda_passengers)) %>%
  autoplot(Transformed_Passengers) +
  labs(title = "Box-Cox Transformed Economy Class Passengers (Melbourne-Sydney)",
       x = "Year",
       y = "Transformed Passengers")
```



Applying the Box-Cox Transformation: Economy Class Passengers from ansett

Pedestrian Counts from pedestrian

```
# Estimate the optimal lambda for Pedestrian counts at Southern Cross Station
lambda_pedestrian <- pedestrian %>%
```

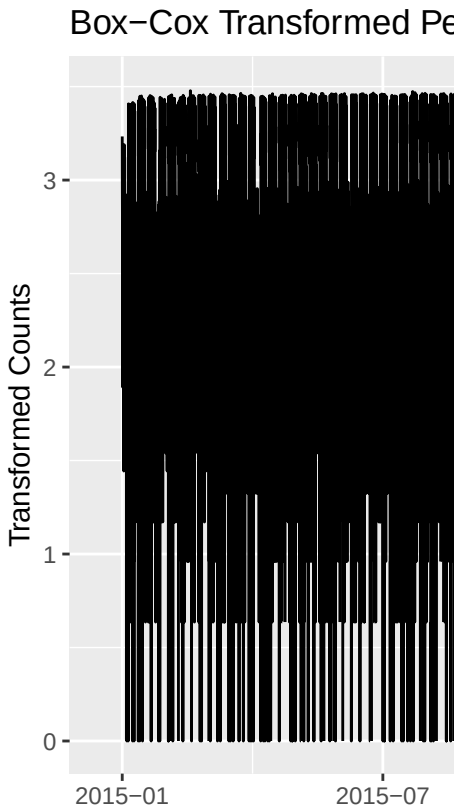
```
  filter(Sensor == "Southern Cross Station") %>%
  features(Count, features = guerrero) %>%
  pull(lambda_guerrero)

print('Optimum Value Lambda Pedestrian:'); lambda_pedestrian
```

Estimating Optimal Lambda

```
## [1] "Optimum Value Lambda Pedestrian:"
```

```
## [1] -0.2501616
```

```
pedestrian %>%
  filter(Sensor == "Southern Cross Station") %>%
  mutate(Transformed_Count = box_cox(Count, lambda_pedestrian)) %>%
  autoplot(Transformed_Count) +
  labs(title = "Box-Cox Transformed Pedestrian Counts at Southern Cross Station",
       x = "Year",
       y = "Transformed Counts")
```



Applying the Box-Cox Transformation: Pedestrian Counts from pedestrian

Exercise 3.7.7

a. Plotting the time series. Can you identify seasonal fluctuations and/or a trend-cycle?

```
gas <- tail(aus_production, 5*4) %>% select(Gas)

gas %>%  autoplot(Gas) +
```

```
  labs(title = paste("PGas Production Over 5 Years"),
       x = "Year",
       y = "Gas Production")
```

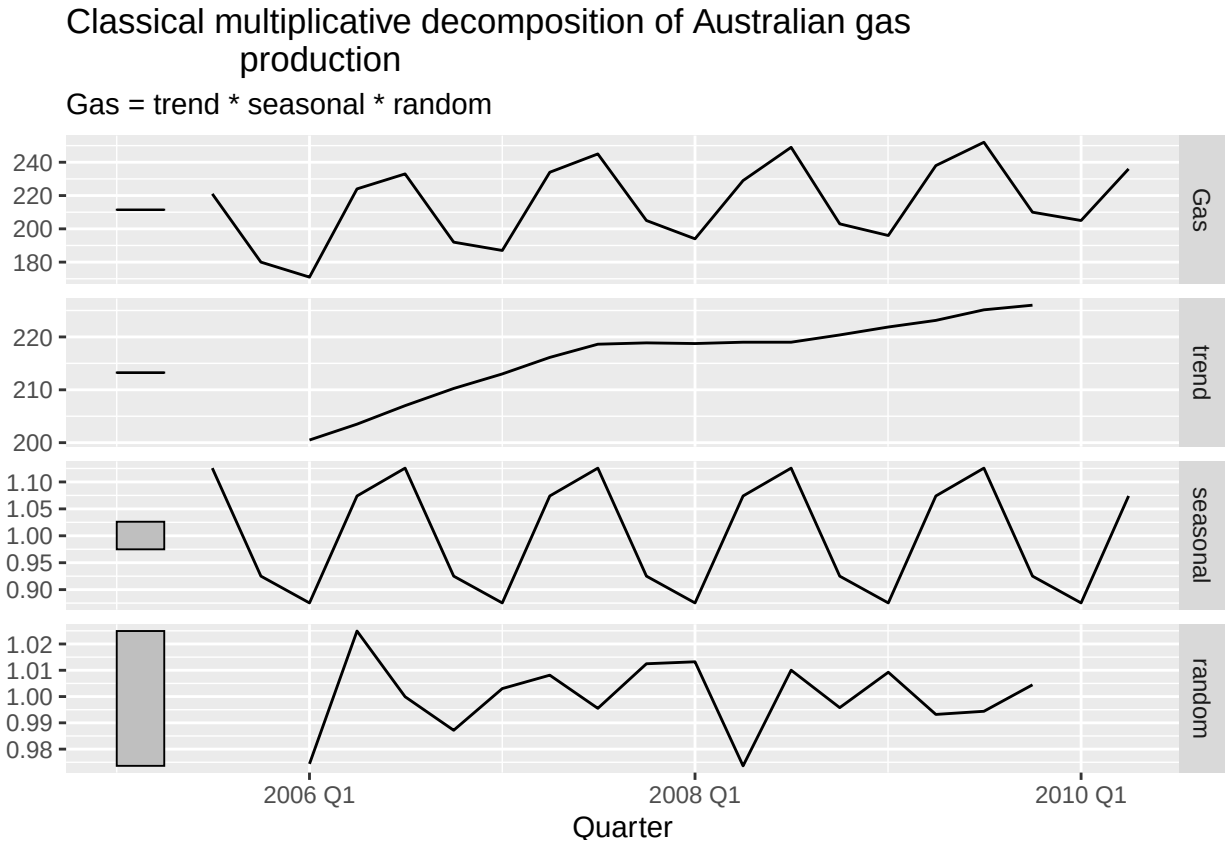
Clear seasonal fluctuations: The plot shows a recurring pattern in gas production each year.

- **Consistent rise and fall:** The production tends to increase and decrease in a predictable manner.
- **Likely indication of seasonality:** The observed pattern suggests that seasonality is a factor in the data.

b. Use classical_decomposition with type=multiplicative to calculate the trend-cycle and seasonal indices.

```
gas_class_decompos <- gas %>%
  model(classical_decomposition(Gas, type = "multiplicative")) %>%
  components() %>%
  autoplot() +
  labs(title = "Classical multiplicative decomposition of Australian gas
                  production")

gas_class_decompos
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



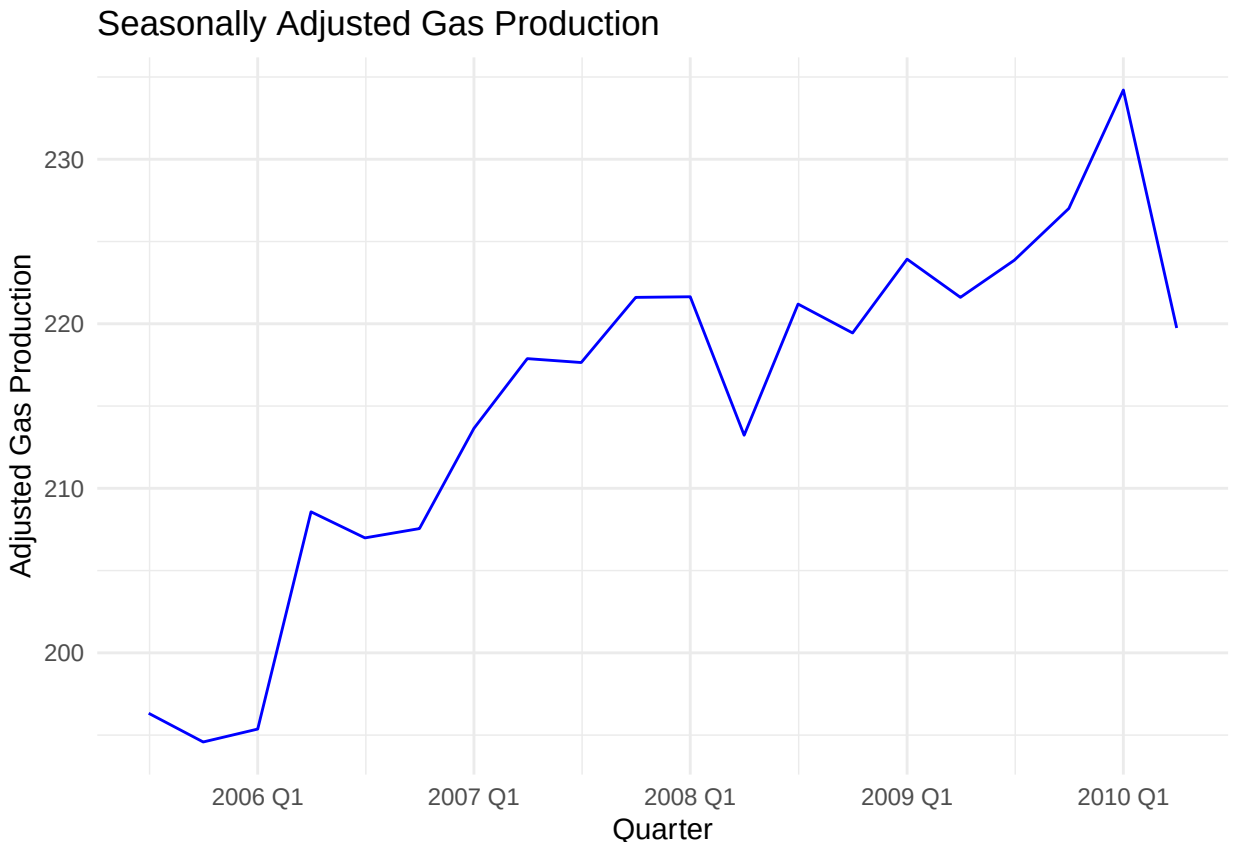
c. Do the results support the graphical interpretation from part a?

The results from the classical multiplicative decomposition in **part b** match the graphical interpretation from **part a**.

- **Part a**: The original gas production plot shows clear seasonal fluctuations with a repeating pattern every year. There is also a slight upward trend over time.
- **Part b**: The decomposition confirms this:
  - The **seasonal component** captures the repeating fluctuations seen in the original plot.
  - The **trend component** shows a gradual increase over time, just as observed in part a.
  - The **random component** accounts for the remaining noise or irregularities after removing the trend and seasonality.

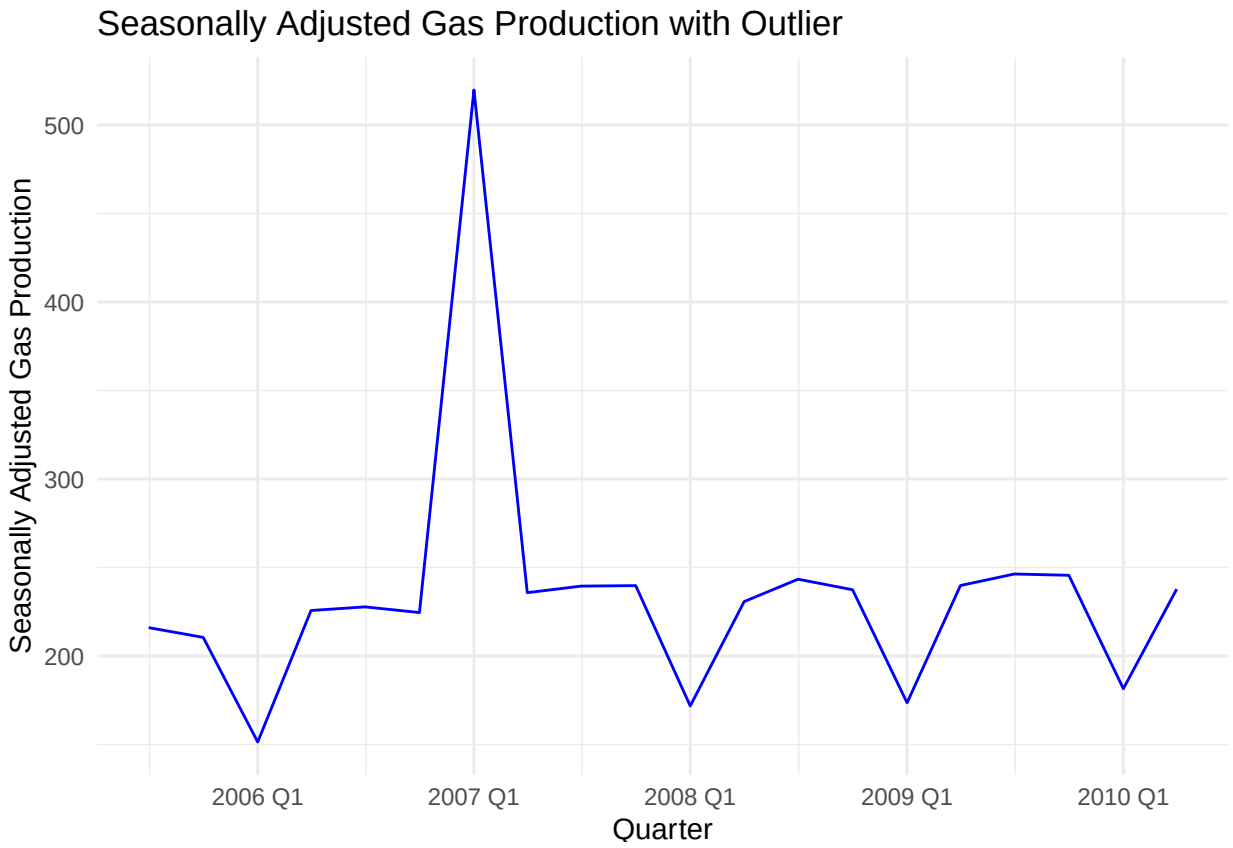
d. Compute and plot the seasonally adjusted data.

```
gas_class_decompos <- gas %>%
  model(classical_decomposition(Gas, type = "multiplicative")) |>
  components()

# Plotting the seasonally adjusted data
ggplot(gas_class_decompos, aes(x = Quarter, y = season_adjust)) +
  geom_line(color = "blue") +
  labs(title = "Seasonally Adjusted Gas Production",
       x = "Quarter",
       y = "Adjusted Gas Production") +
  theme_minimal()
```



e. Effect of outlier? (e.g., by adding 300 to one observation)

```
outlier_gas_400 <- gas

# Introducing an outlier by adding 400 to the observation in 2007 Q1
outlier_gas_400 <- outlier_gas_400 %>%
  mutate(Gas = if_else(Quarter == yearquarter("2007 Q1"), Gas + 400, Gas))


# Use of Classical decomposition with the outlier
outlier_gas_class_decompos <- outlier_gas_400 %>%
  model(classical_decomposition(Gas, type = "multiplicative")) %>%
  components()

ggplot(outlier_gas_class_decompos, aes(x = Quarter, y = season_adjust)) +
  geom_line(color = "blue") +
  labs(title = "Seasonally Adjusted Gas Production with Outlier",
       x = "Quarter",
       y = "Seasonally Adjusted Gas Production") +
  theme_minimal()
```

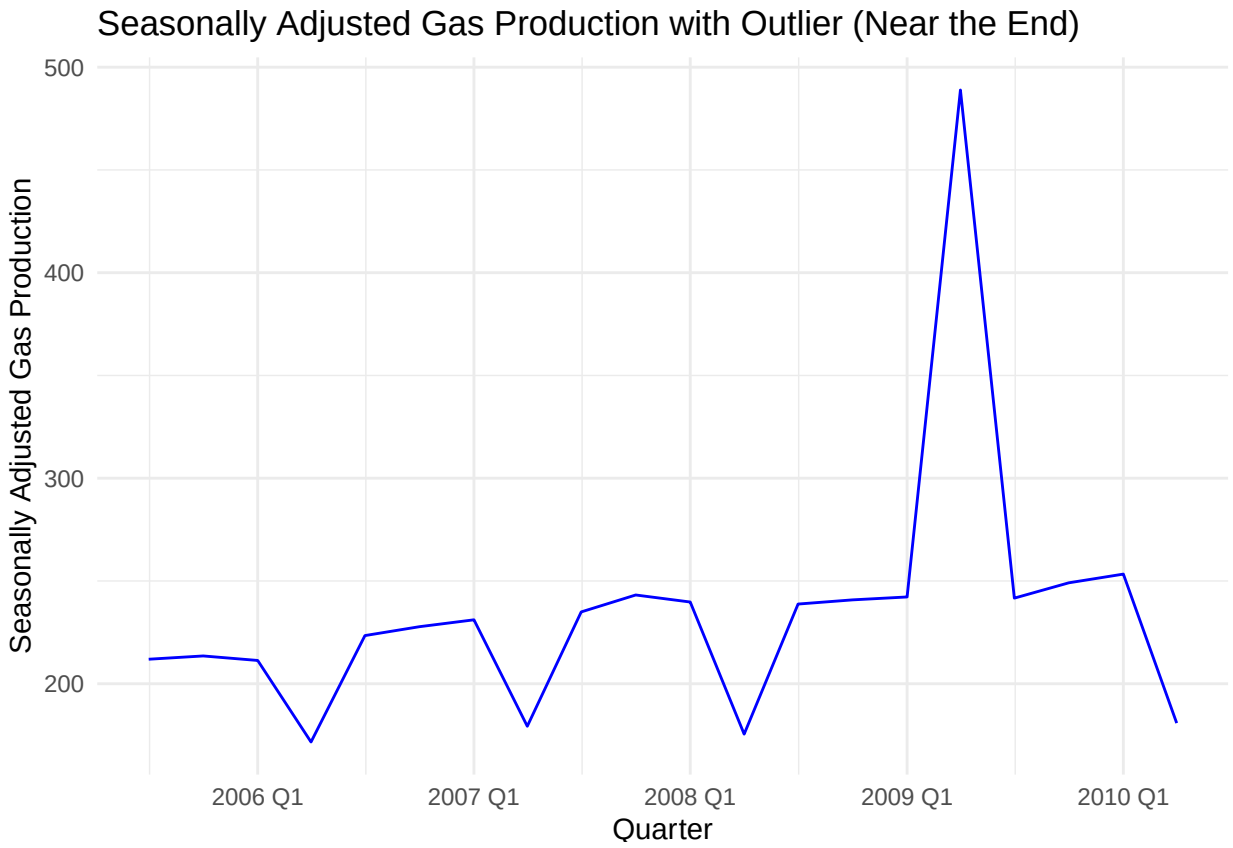


Effect of the Outlier

- As seen in the graph above, the outlier disrupts the trend and seasonal components, leading to a noticeable jump in the seasonally adjusted data.

f. Does it make any difference if the outlier is near the end rather than in the middle of the time series?

```
outlier_gas_near_end <- gas

# Introducing an outlier by adding 400 to the observation in 2009 Q2 (near the end)
outlier_gas_near_end <- outlier_gas_near_end %>%
  mutate(Gas = if_else(Quarter == yearquarter("2009 Q2"), Gas + 400, Gas))

outlier_gas_class_decompos <- outlier_gas_near_end %>%
  model(classical_decomposition(Gas, type = "multiplicative")) %>%
  components()

ggplot(outlier_gas_class_decompos, aes(x = Quarter, y = season_adjust)) +
  geom_line(color = "blue") +
  labs(title = "Seasonally Adjusted Gas Production with Outlier (Near the End)",
       x = "Quarter",
       y = "Seasonally Adjusted Gas Production") +
  theme_minimal()
```



- The position of the outlier does indeed impact how the seasonally adjusted gas production is affected.
- When the outlier is placed in the middle of the time series (e.g., 2007 Q4), it causes a significant spike that distorts the surrounding data, making the seasonal adjustment rise sharply around the middle of the series. This results in a more pronounced distortion in that period.
- On the other hand, when the outlier is near the end of the series (e.g., 2009 Q2), the effect is concentrated towards the end of the time series. This causes a noticeable spike followed by a sharp decline, affecting the final observations more heavily.
- In both cases, the outlier introduces an artificial spike in the data. However, its placement alters how the subsequent values are adjusted because the seasonal and trend components are influenced by the entire structure of the time series.
- The closer the outlier is to the beginning or end of the series, the more localized its impact on the seasonally adjusted values.

Exercise 3.7.8: Recall your retail time series data (from Exercise 7 in Section 2.10). Decompose the series using X-11. Does it reveal any outliers, or unusual features that you had not noticed previously?

I couldn't get the X-13ARIMA-SEATS results due to compatibility issues on my Mac. Despite trying multiple solutions (reinstalling x13binary, checking paths, and attempting alternative methods), system incompatibilities prevented successful execution. - Then I commented my code below

```
set.seed(123210)
myseries <- aus_retail %>%
  filter(`Series ID` == sample(aus_retail$`Series ID`, 1))
```

```r
# Convert myseries to a tsibble and set Month as the index
myseries <- myseries %>%
  as_tsibble(index = Month)

# Perform X-13ARIMA-SEATS decomposition using X-11 on your retail data (renamed Sales)
#x11_dcmp <- myseries %>%
  #model(x11 = X_13ARIMA_SEATS(Turnover ~ x11()))

# Extract the decomposition components
#x11_components <- components(x11_dcmp)

# Plot the decomposed components
#autoplot(x11_components) +
  #labs(title = "Decomposition of Retail Sales using X-11",
       #x = "Month",
       #y = "Turnover")

# Analyzing the remainder (irregular component) for outliers or unusual features
#autoplot(x11_components, remainder) +
  #labs(title = "Remainder Component of Retail Time Series",
       #x = "Year",
       #y = "Remainder (Irregular Component)")
```

Exercise 3.7.9

Analyinsg the result of decomposing the number of persons in the civilian labour force in Australia each month from February 1978 to August 1995:

Decomposition Interpretation:

- The **trend** component shows a steady increase in the number of persons in the Australian civilian labor force from 1978 to 1995, indicating overall growth over time.
- The **seasonal component** reveals clear cyclical patterns repeating each year, with notable increases in March, April, and December, and decreases in January and August. The magnitude of these seasonal fluctuations remains relatively constant over the entire period.
- The **remainder component** (irregular component) displays short-term fluctuations and noise, showing no clear pattern. However, there is a significant dip around 1991, suggesting that an unusual event may have affected the labor force during that time.
- The scales of the graphs highlight the magnitude of each component, with the trend showing a large-scale increase and the seasonal and remainder components oscillating around zero, indicating that the main changes come from the trend.

Recession of 1991/1992:

- Yes, the recession of 1991/1992 is visible, particularly in the **remainder component**. There is a noticeable sharp drop in the remainder starting around 1991, indicating an unusual deviation from the expected values during that period. This dip likely corresponds to the impact of the recession on employment figures. The **trend** component shows a flattening during this period, suggesting a temporary slowdown in growth.
