Homework 2

Heleine Fouda

2024-09-14

Instructions: Do exercises 3.1, 3.2, 3.3, 3.4, 3.5, 3.7, 3.8 and 3.9 from the online Hyndman book: Forcasting: Principles and Practice (3rd ed.).

Setting up the environment: Load Required libraries

Exercise 3.1

Consider the GDP information in global_economy. Plot the GDP per capita for each country over time. Which country has the highest GDP per capita? How has this changed over time?

```
global_economy
```

```
## # A tsibble: 15,150 x 9 [1Y]
## # Key:       Country [263]
##    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
##  7 Afghanistan AFG    1966 1399999967.     NA    NA   18.6     8.57   10152331
##  8 Afghanistan AFG    1967 1673333418.     NA    NA   14.2     6.77   10372630
##  9 Afghanistan AFG    1968 1373333367.     NA    NA   15.2     8.90   10604346
## 10 Afghanistan AFG    1969 1408888922.     NA    NA   15.0    10.1    10854428
## # i 15,140 more rows
```

```
glimpse(global_economy)
```

```
## Rows: 15,150
## Columns: 9
## Key: Country [263]
## $ Country    <fct> "Afghanistan", "Afghanistan", "Afghanistan", "Afghanistan",~
## $ Code       <fct> AFG, AFG, AFG, AFG, AFG, AFG, AFG, AFG, AFG, AFG, AFG, AFG,~
## $ Year       <dbl> 1960, 1961, 1962, 1963, 1964, 1965, 1966, 1967, 1968, 1969,~
## $ GDP        <dbl> 537777811, 548888896, 546666678, 751111191, 800000044, 1006~
## $ Growth     <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ CPI        <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Imports    <dbl> 7.024793, 8.097166, 9.349593, 16.863910, 18.055555, 21.4128~
## $ Exports    <dbl> 4.132233, 4.453443, 4.878051, 9.171601, 8.888893, 11.258279~
## $ Population <dbl> 8996351, 9166764, 9345868, 9533954, 9731361, 9938414, 10152~
```

```
# Calculate GDP per capita and relocate the new column after 'Country'
global_economy <- global_economy |>
```

```r
  mutate(GDP_per_capita = GDP / Population) |>
  relocate(GDP_per_capita, .after = Country)
global_economy
```

```
## # A tsibble: 15,150 x 10 [1Y]
## # Key:       Country [263]
##    Country     GDP_per_capita Code   Year      GDP Growth   CPI Imports Exports
##    <fct>                <dbl> <fct> <dbl>    <dbl>  <dbl> <dbl>   <dbl>   <dbl>
##  1 Afghanistan           59.8 AFG    1960   5.38e8     NA    NA    7.02    4.13
##  2 Afghanistan           59.9 AFG    1961   5.49e8     NA    NA    8.10    4.45
##  3 Afghanistan           58.5 AFG    1962   5.47e8     NA    NA    9.35    4.88
##  4 Afghanistan           78.8 AFG    1963   7.51e8     NA    NA   16.9     9.17
##  5 Afghanistan           82.2 AFG    1964   8.00e8     NA    NA   18.1     8.89
##  6 Afghanistan          101.  AFG    1965   1.01e9     NA    NA   21.4    11.3
##  7 Afghanistan          138.  AFG    1966   1.40e9     NA    NA   18.6     8.57
##  8 Afghanistan          161.  AFG    1967   1.67e9     NA    NA   14.2     6.77
##  9 Afghanistan          130.  AFG    1968   1.37e9     NA    NA   15.2     8.90
## 10 Afghanistan          130.  AFG    1969   1.41e9     NA    NA   15.0    10.1
## # i 15,140 more rows
## # i 1 more variable: Population <dbl>
```

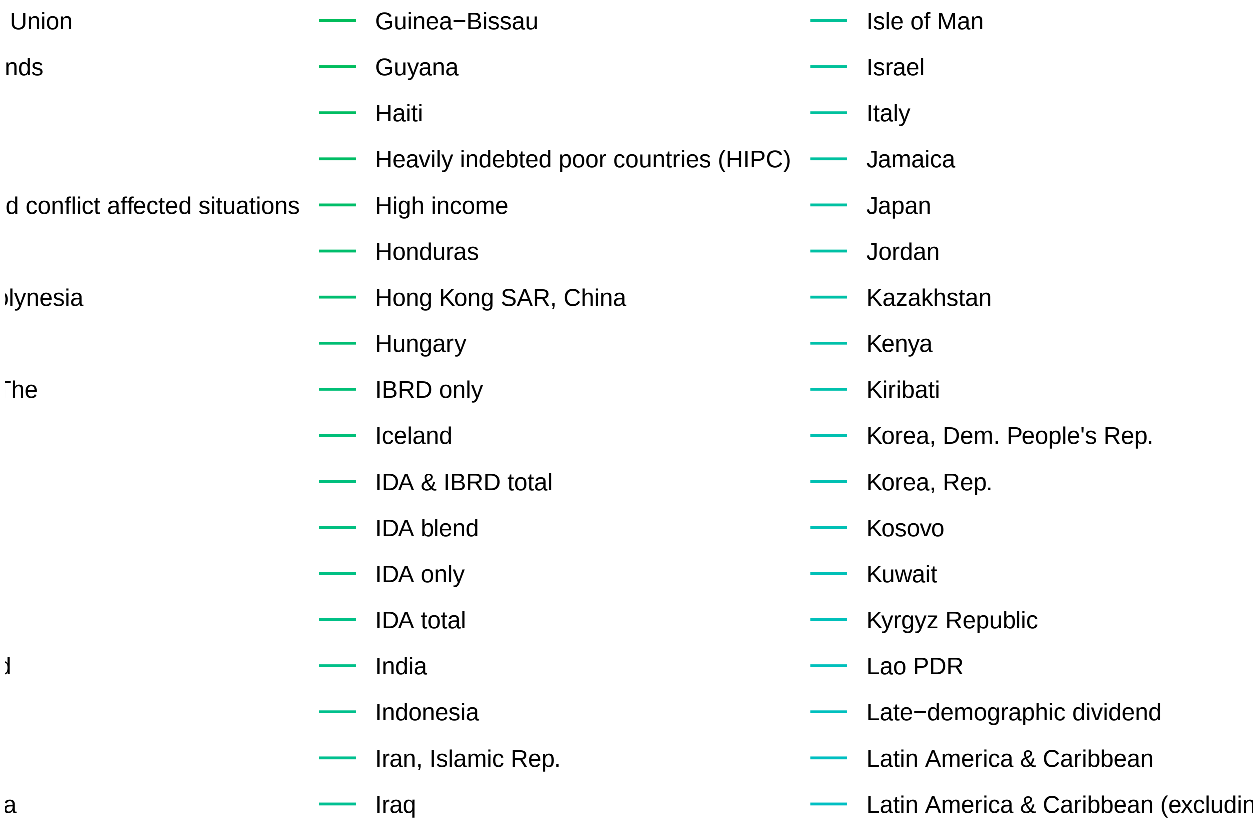
```r
# Identifyng the country with the highest GDP per capita over time
highest_gdp_capita <- global_economy |>
  index_by(Year) |>
  slice_max(GDP_per_capita, with_ties = FALSE) |>
  ungroup()

# print
highest_gdp_capita
```

```
## # A tsibble: 58 x 10 [1Y]
## # Key:       Country [263]
##    Country       GDP_per_capita Code   Year      GDP Growth   CPI Imports Exports
##    <fct>                  <dbl> <fct> <dbl>    <dbl>  <dbl> <dbl>   <dbl>   <dbl>
##  1 United States          3007. USA    1960 5.43e11 NA     13.6    4.20    4.97
##  2 United States          3067. USA    1961 5.63e11   2.30 13.7    4.03    4.90
##  3 United States          3244. USA    1962 6.05e11   6.10 13.9    4.13    4.81
##  4 United States          3375. USA    1963 6.39e11   4.40 14.0    4.09    4.87
##  5 United States          3574. USA    1964 6.86e11   5.80 14.2    4.10    5.10
##  6 Kuwait                 4429. KWT    1965 2.10e 9 NA     NA     23.1    67.7
##  7 Kuwait                 4556. KWT    1966 2.39e 9 12.3   NA     24.4    65.7
##  8 United States          4336. USA    1967 8.62e11   2.50 15.3    4.63    5.05
##  9 United States          4696. USA    1968 9.42e11   4.80 16.0    4.94    5.08
## 10 United States          5032. USA    1969 1.02e12   3.10 16.8    4.95    5.09
## # i 48 more rows
## # i 1 more variable: Population <dbl>
```

```r
# visualizing GDP per capita for each country over time
autoplot(global_economy) +
  labs(title = "GDP per Capita Over Time for Each Country",
       y = "GDP per Capita",
       x = "Year") +
  facet_wrap(~Country, scales = "free_y") +
  theme_minimal()
```

```
## Plot variable not specified, automatically selected `.vars = GDP_per_capita`
```

```
## Warning: Removed 3242 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

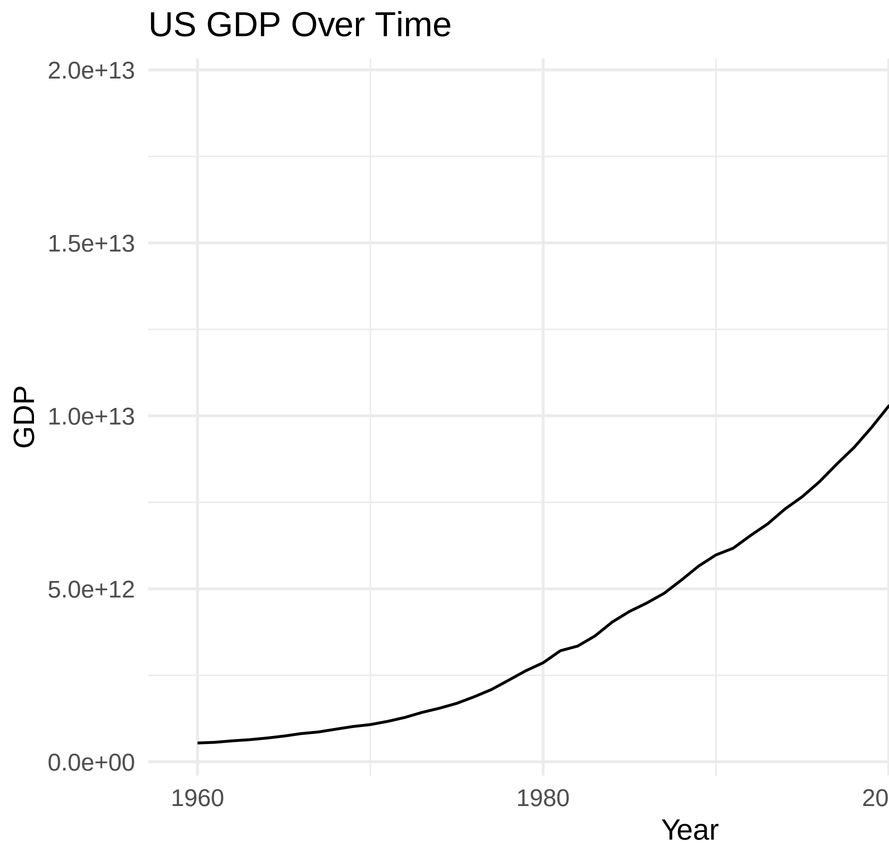


Exercise 3.2

For each of the following series, make a graph of the data. If transforming seems appropriate, do so and describe the effect.

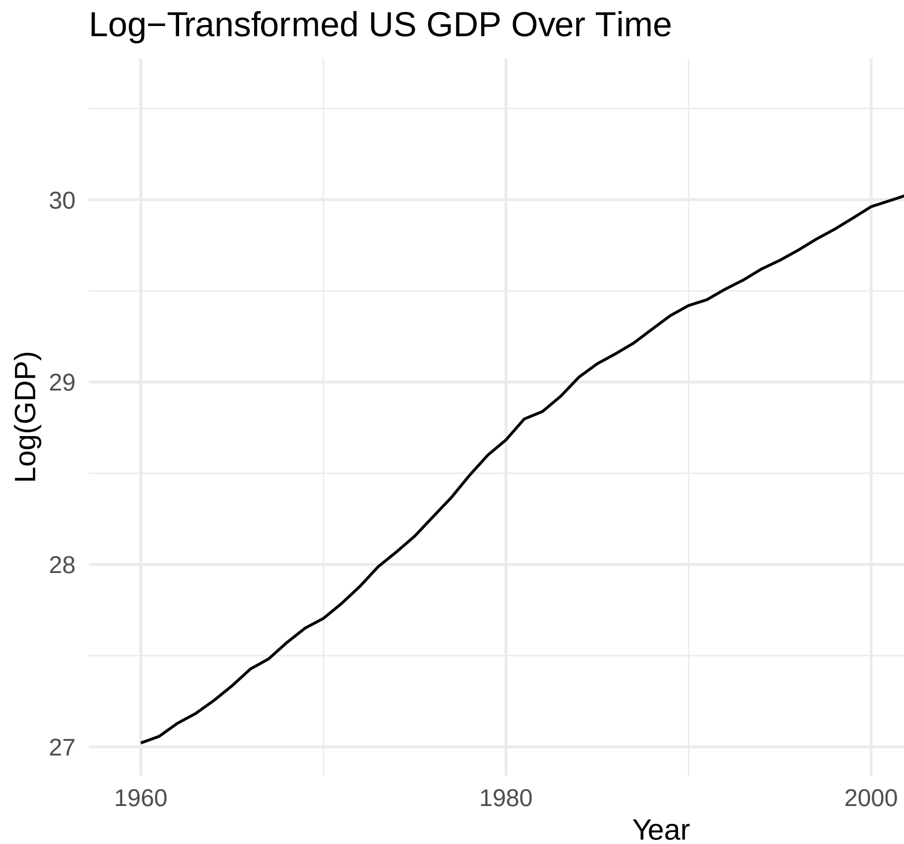
United States GDP from global_economy

```r
# Extract US GDP data from global_economy

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

# Plot original US GDP
ggplot(us_gdp, aes(x = Year, y = GDP)) +
  geom_line() +
  labs(title = "US GDP Over Time", x = "Year", y = "GDP") +
  theme_minimal()
```



Graph of US GDP before transformations

```r
# Plot log-transformed US GDP (to stabilize growth rate)
ggplot(us_gdp, aes(x = Year, y = log(GDP))) +
  geom_line() +
  labs(title = "Log-Transformed US GDP Over Time", x = "Year", y = "Log(GDP)") +
  theme_minimal()
```



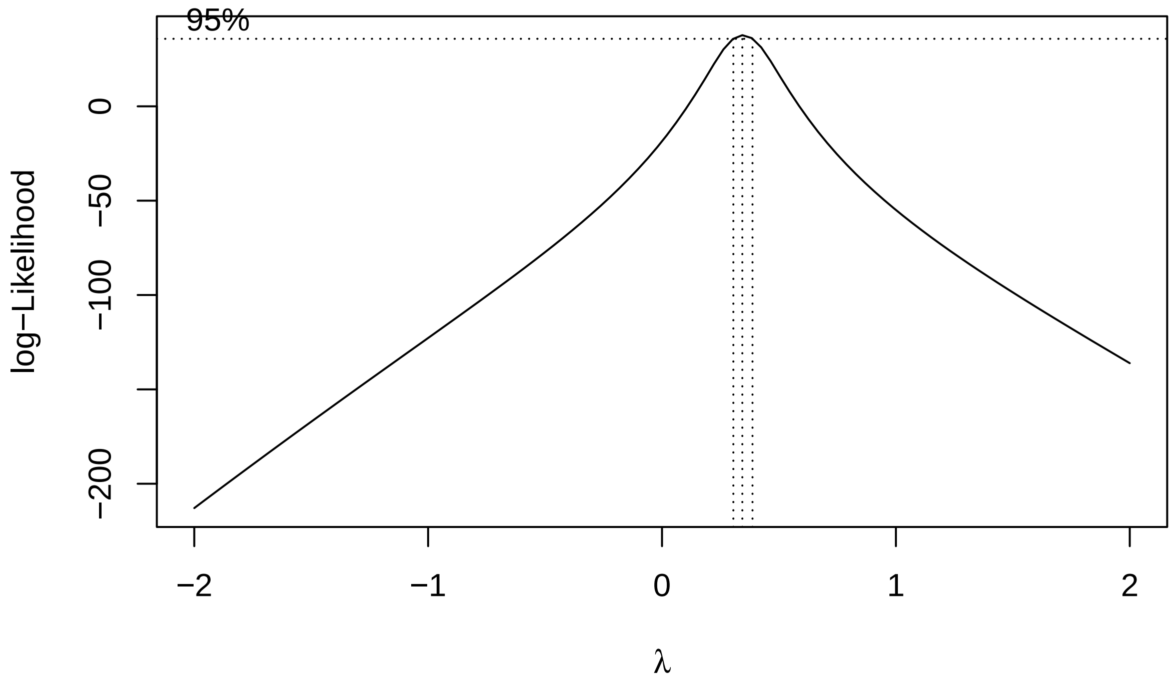
Graph of US GDP after Transformations

```
library(MASS) # For boxcox function
```

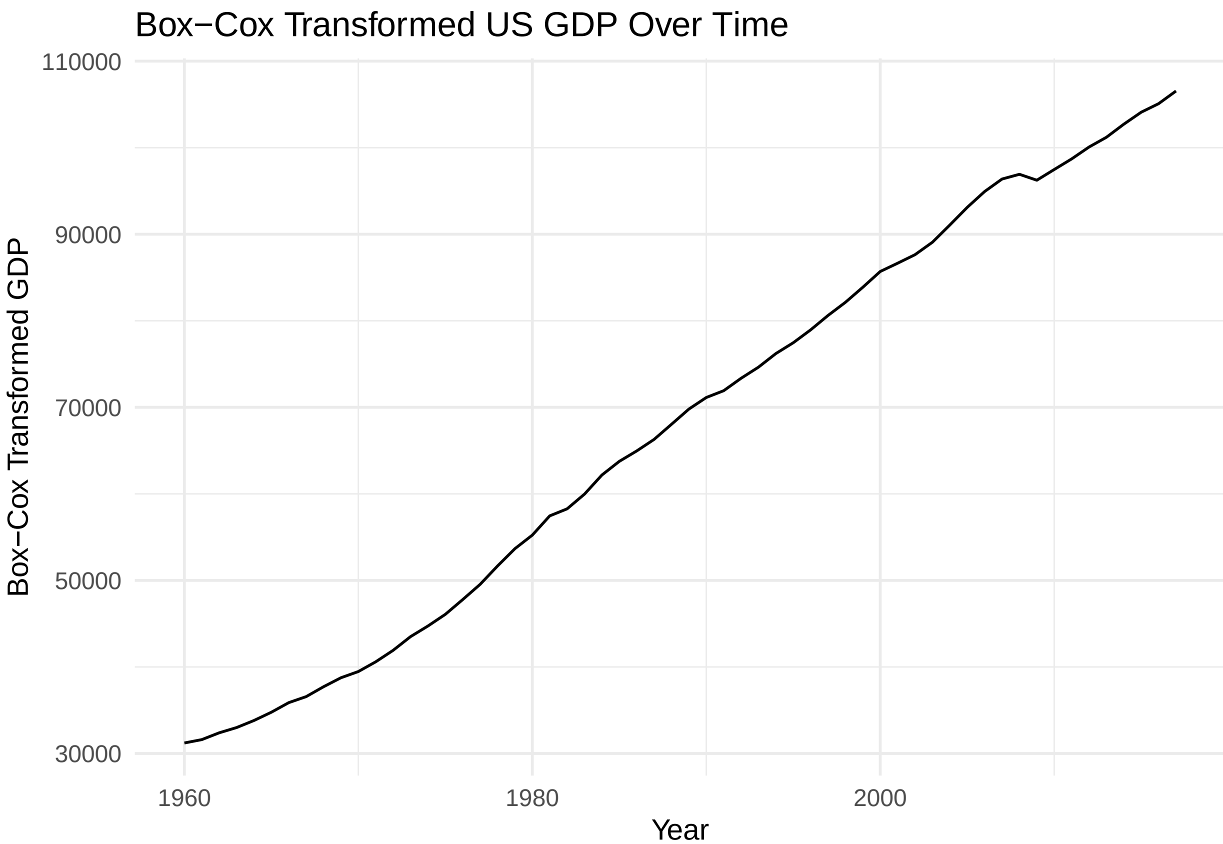
```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
# Fit a linear model to use with Box-Cox
fit_us_gdp <- lm(GDP ~ Year, data = us_gdp)

# Apply Box-Cox transformation and find the optimal lambda
boxcox_result <- boxcox(fit_us_gdp, lambda = seq(-2, 2, 1/10))
```



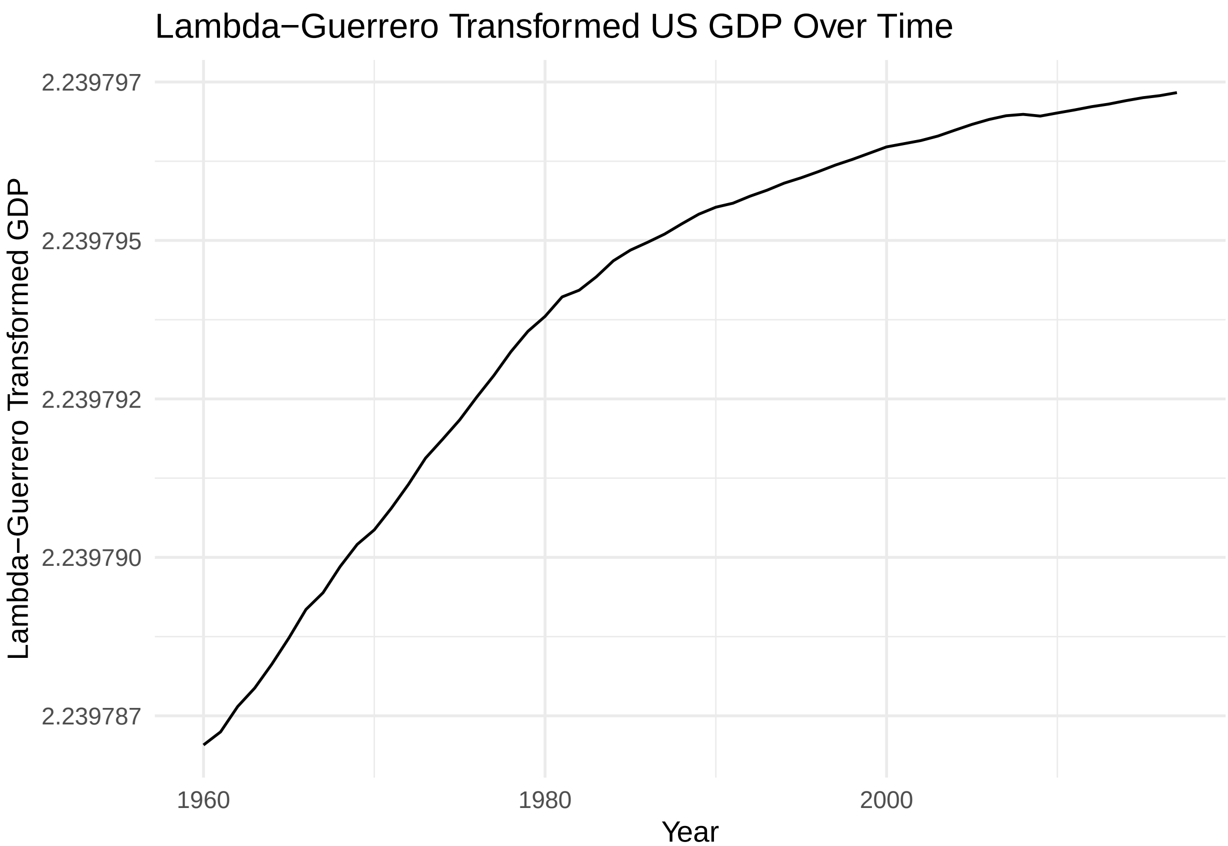

```
lambda_boxcox <- boxcox_result$x[which.max(boxcox_result$y)]

# Apply Box-Cox transformation with the optimal lambda
us_gdp_transformed_boxcox <- us_gdp |>
  mutate(GDP_BoxCox = (GDP^lambda_boxcox - 1) / lambda_boxcox)

# Plot Box-Cox transformed data
ggplot(us_gdp_transformed_boxcox, aes(x = Year, y = GDP_BoxCox)) +
  geom_line() +
  labs(title = "Box-Cox Transformed US GDP Over Time", x = "Year", y = "Box-Cox Transformed GDP") +
  theme_minimal()
```




```
# Function to calculate optimal lambda using Guerrero's method
guerrero_lambda <- function(x) {
  # Basic Guerrero method to estimate lambda


  # Check for positive values
  if (any(x <= 0)) {
    stop("Data must be positive for Guerrero transformation.")
  }

  # Log transformation of data
  log_x <- log(x)

  # Fit linear model on log data
  fit <- lm(log_x ~ 1)

  # Calculate lambda
  lambda <- -0.5 * (sum(residuals(fit)^2) / length(x))^(-1/2)

  return(lambda)
}

# Function to apply the Lambda-Guerrero transformation
guerrero_transform <- function(x, lambda) {
  if (lambda == 0) {
    return(log(x))
  } else {
```

```
    return((x^lambda - 1) / lambda)
  }
}
```

```
# Calculate optimal lambda
lambda_guerrero <- guerrero_lambda(us_gdp$GDP)

 # Apply Lambda-Guerrero transformation
us_gdp_transformed_guerrero <- us_gdp |>
  mutate(GDP_Guerrero = guerrero_transform(GDP, lambda_guerrero))

 # Plot Lambda-Guerrero transformed data
ggplot(us_gdp_transformed_guerrero, aes(x = Year, y = GDP_Guerrero)) +
  geom_line() +
  labs(title = "Lambda-Guerrero Transformed US GDP Over Time",
       x = "Year",
       y = "Lambda-Guerrero Transformed GDP") +
  theme_minimal()
```



Conclusion: For a better modeling of US GDP, a log transformation seems appropriate to stabilize variance and highlight percentage changes.

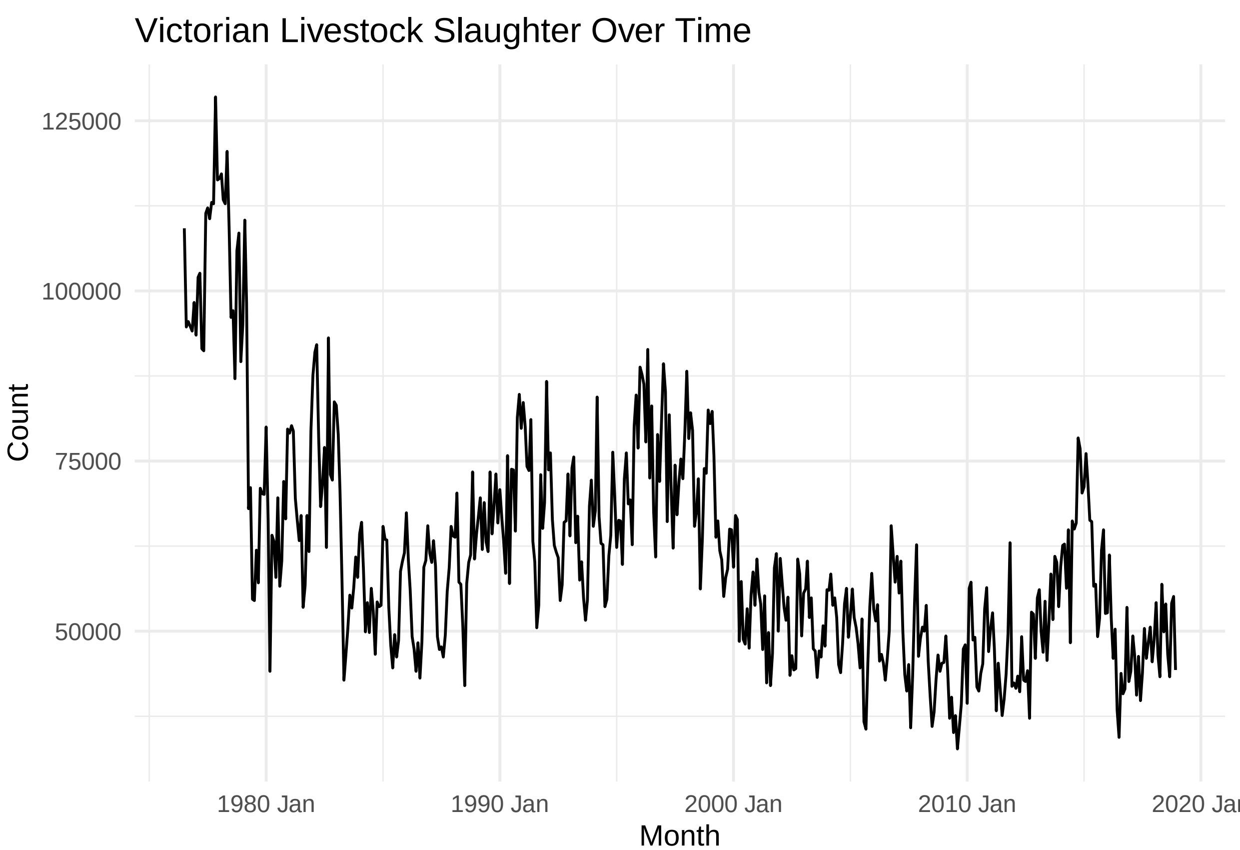
Slaughter of Victorian "Bulls, bullocks and steers" in aus_livestock

```
aus_livestock
```

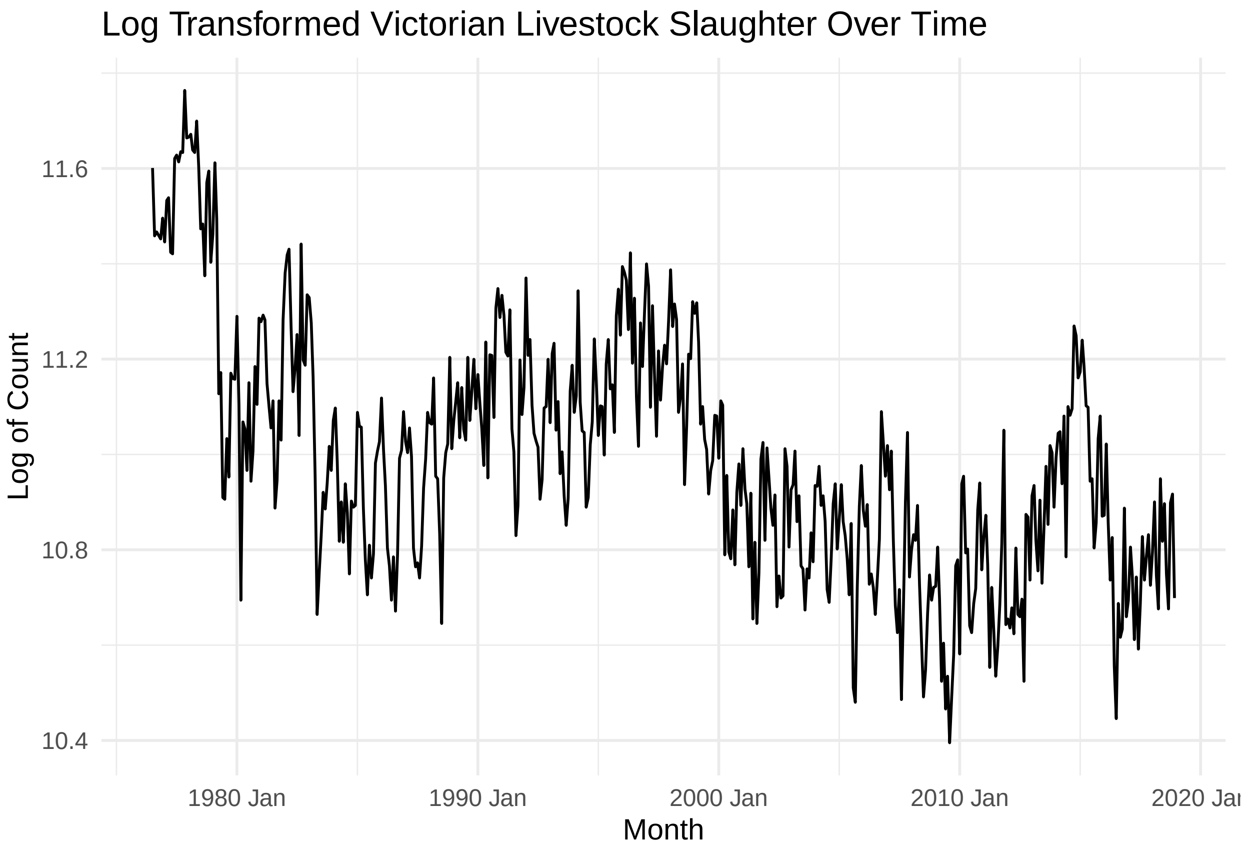
```
## # A tsibble: 29,364 x 4 [1M]
```

```
## # Key:       Animal, State [54]
##       Month Animal                        State                         Count
##       <mth> <fct>                         <fct>                         <dbl>
##  1 1976 Jul Bulls, bullocks and steers Australian Capital Territory  2300
##  2 1976 Aug Bulls, bullocks and steers Australian Capital Territory  2100
##  3 1976 Sep Bulls, bullocks and steers Australian Capital Territory  2100
##  4 1976 Oct Bulls, bullocks and steers Australian Capital Territory  1900
##  5 1976 Nov Bulls, bullocks and steers Australian Capital Territory  2100
##  6 1976 Dec Bulls, bullocks and steers Australian Capital Territory  1800
##  7 1977 Jan Bulls, bullocks and steers Australian Capital Territory  1800
##  8 1977 Feb Bulls, bullocks and steers Australian Capital Territory  1900
##  9 1977 Mar Bulls, bullocks and steers Australian Capital Territory  2700
## 10 1977 Apr Bulls, bullocks and steers Australian Capital Territory  2300
## # i 29,354 more rows
```

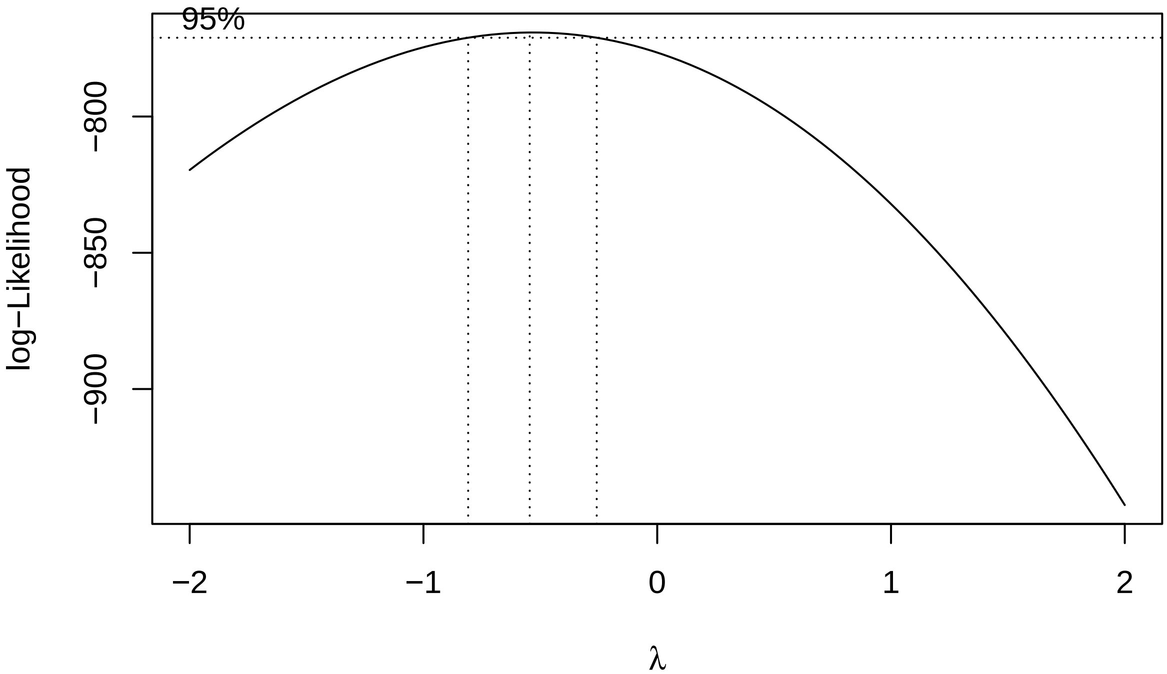
```r
# Extract Victorian livestock data
vic_livestock <- aus_livestock |>
  dplyr::filter(Animal == "Bulls, bullocks and steers", State == "Victoria") |>
  dplyr::select(Month, Count)


# Plot original livestock data
ggplot(vic_livestock, aes(x = Month, y = Count)) +
  geom_line() +
  labs(title = "Victorian Livestock Slaughter Over Time", x = "Month", y = "Count") +
  theme_minimal()
```



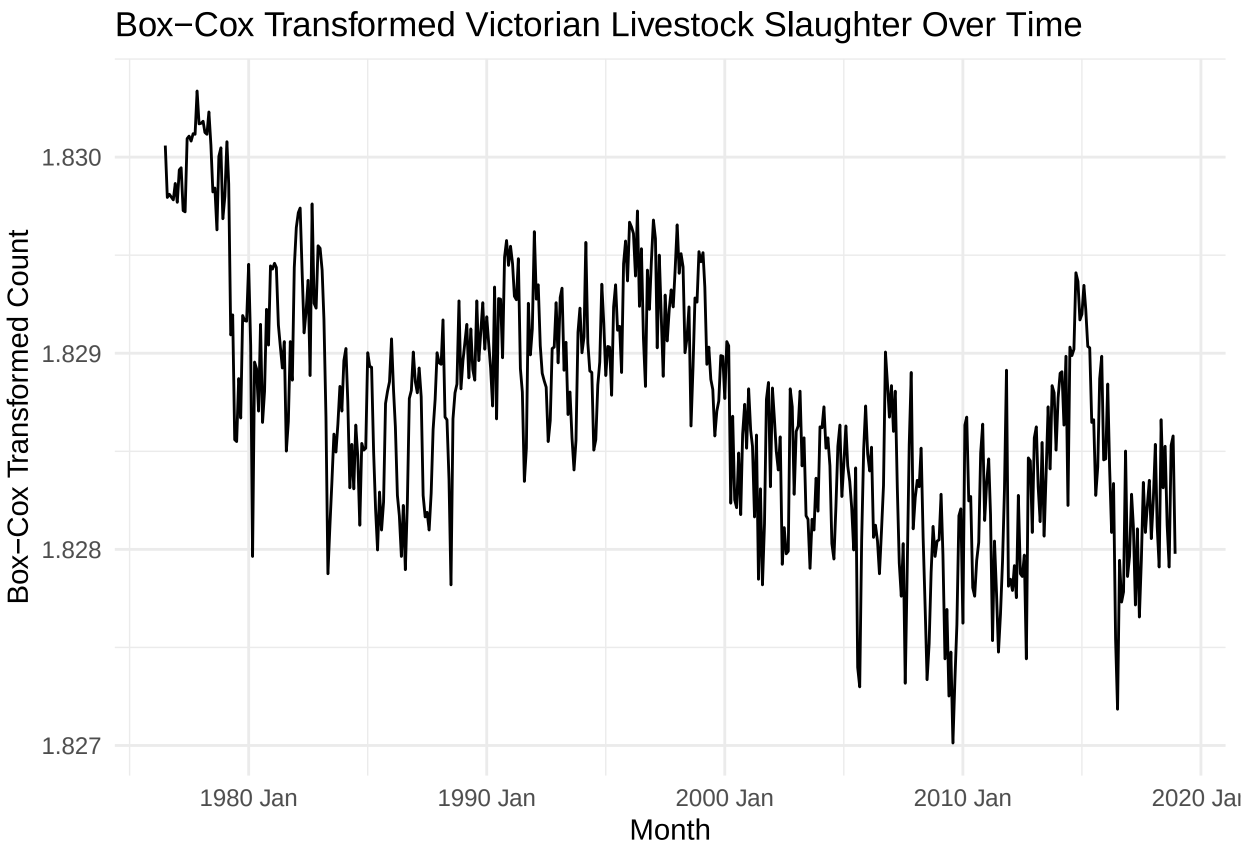
```r
# Log Transformation
vic_livestock_log <- vic_livestock %>%
  mutate(Count_Log = log(Count))

# Plot Log-transformed livestock
ggplot(vic_livestock_log, aes(x = Month, y = Count_Log)) +
  geom_line() +
  labs(title = "Log Transformed Victorian Livestock Slaughter Over Time", x = "Month", y = "Log of Coun
  theme_minimal()
```



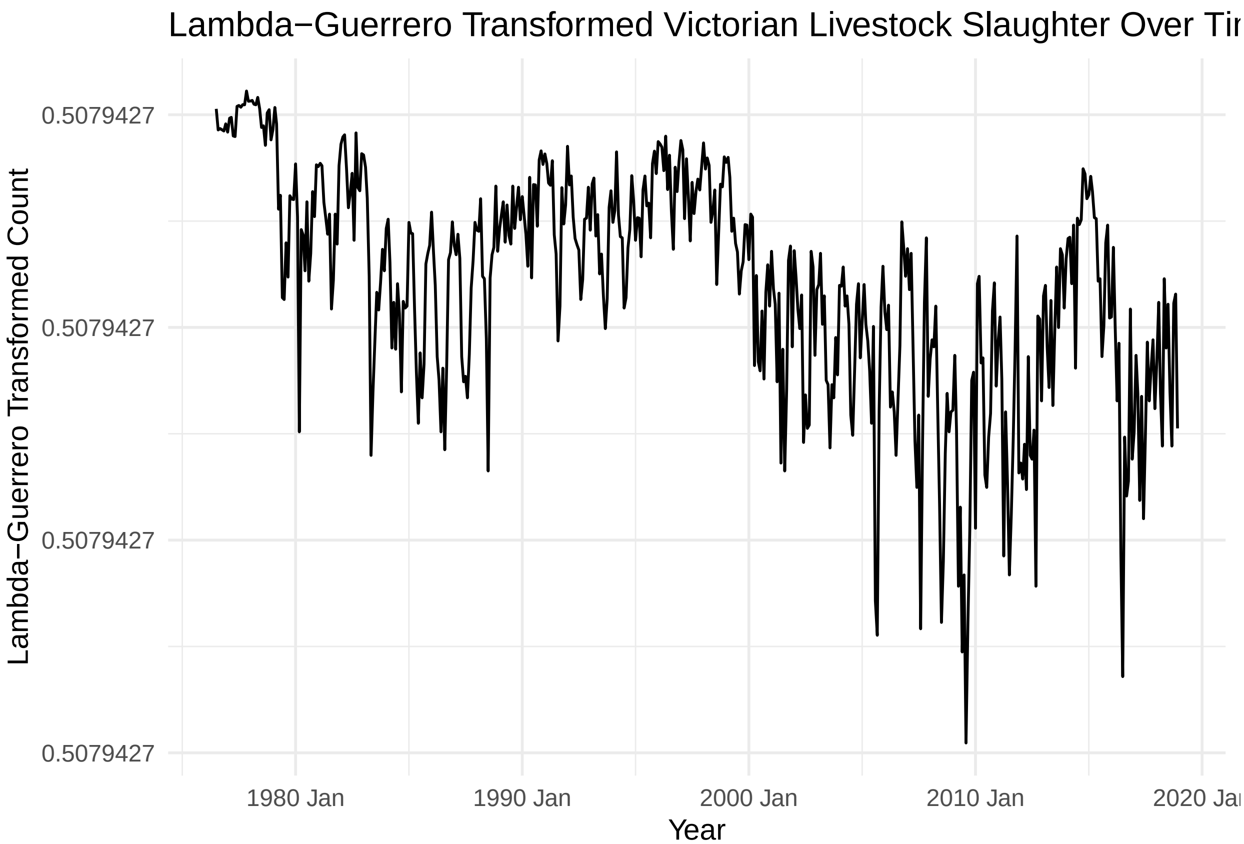

```r
# Fit a linear model to use with Box-Cox
fit_vic_livestock <- lm(Count ~ Month, data = vic_livestock)

# Apply Box-Cox transformation and find the optimal lambda
boxcox_result <- boxcox(fit_vic_livestock, lambda = seq(-2, 2, 1/10))
```




```
lambda_boxcox <- boxcox_result$x[which.max(boxcox_result$y)]

# Apply Box-Cox transformation with the optimal lambda
vic_livestock_transformed_boxcox <- vic_livestock |>
  mutate(Count_BoxCox = (Count^lambda_boxcox - 1) / lambda_boxcox)

# Plot Box-Cox transformed data
ggplot(vic_livestock_transformed_boxcox, aes(x = Month, y = Count_BoxCox)) +
  geom_line() +
  labs(title = "Box-Cox Transformed Victorian Livestock Slaughter Over Time", x = "Month", y = "Box-Cox
  theme_minimal()
```




```r
# Calculate optimal lambda
lambda_guerrero <- guerrero_lambda(vic_livestock$Count)

# Calculate Lambda-Guerrero transformation

vic_livestock_transformed_guerrero <- vic_livestock |>
  mutate(Count_Guerrero = guerrero_transform(Count, lambda_guerrero))

# Plot Lambda-Guerrero transformed data
ggplot(vic_livestock_transformed_guerrero, aes(x = Month, y = Count_Guerrero)) +
  geom_line() +
  labs(title = "Lambda-Guerrero Transformed Victorian Livestock Slaughter Over Time", x = "Year", y = "I
  theme_minimal()
```



In conlcusion, for aus_livestock, no transformation is really needed unless variance grows over time or seasonal patterns exist

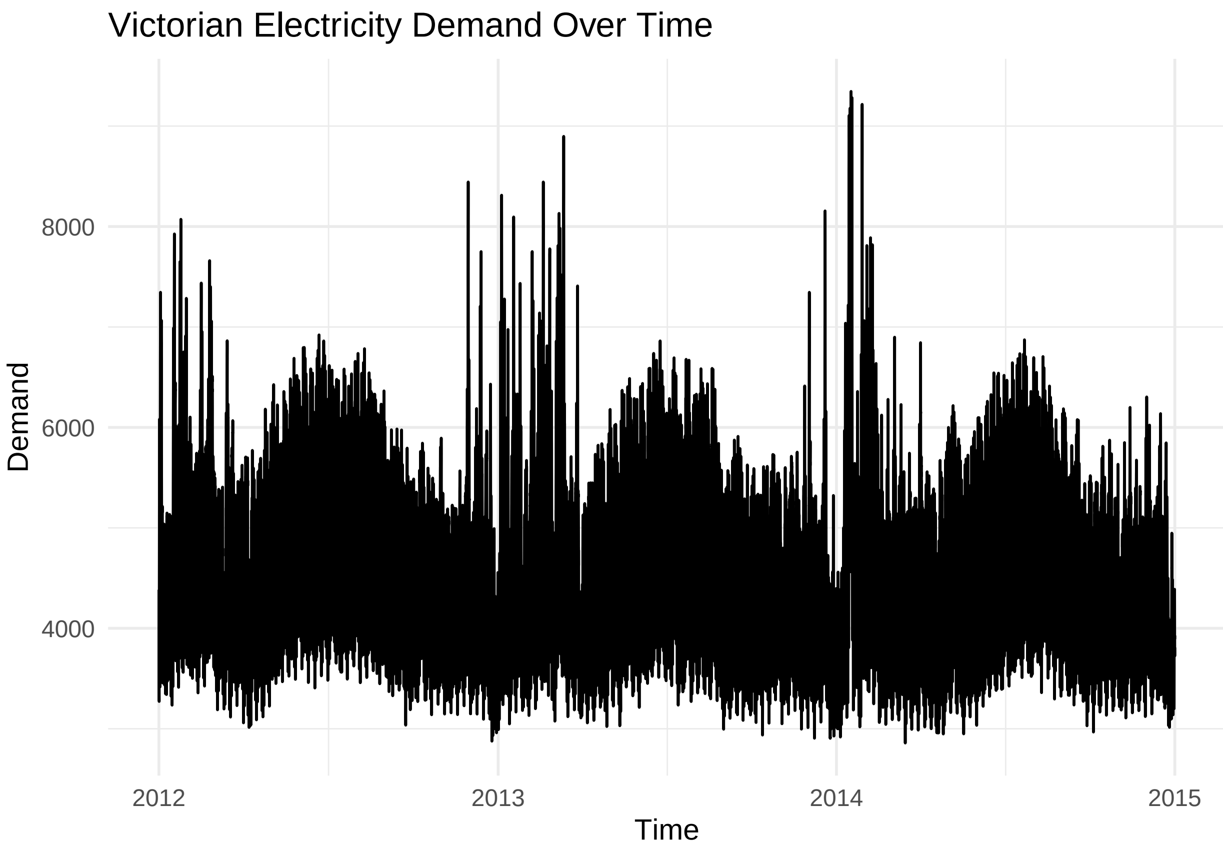
Victorian Electricity Demand from vic_elec.

```
vic_elec
```

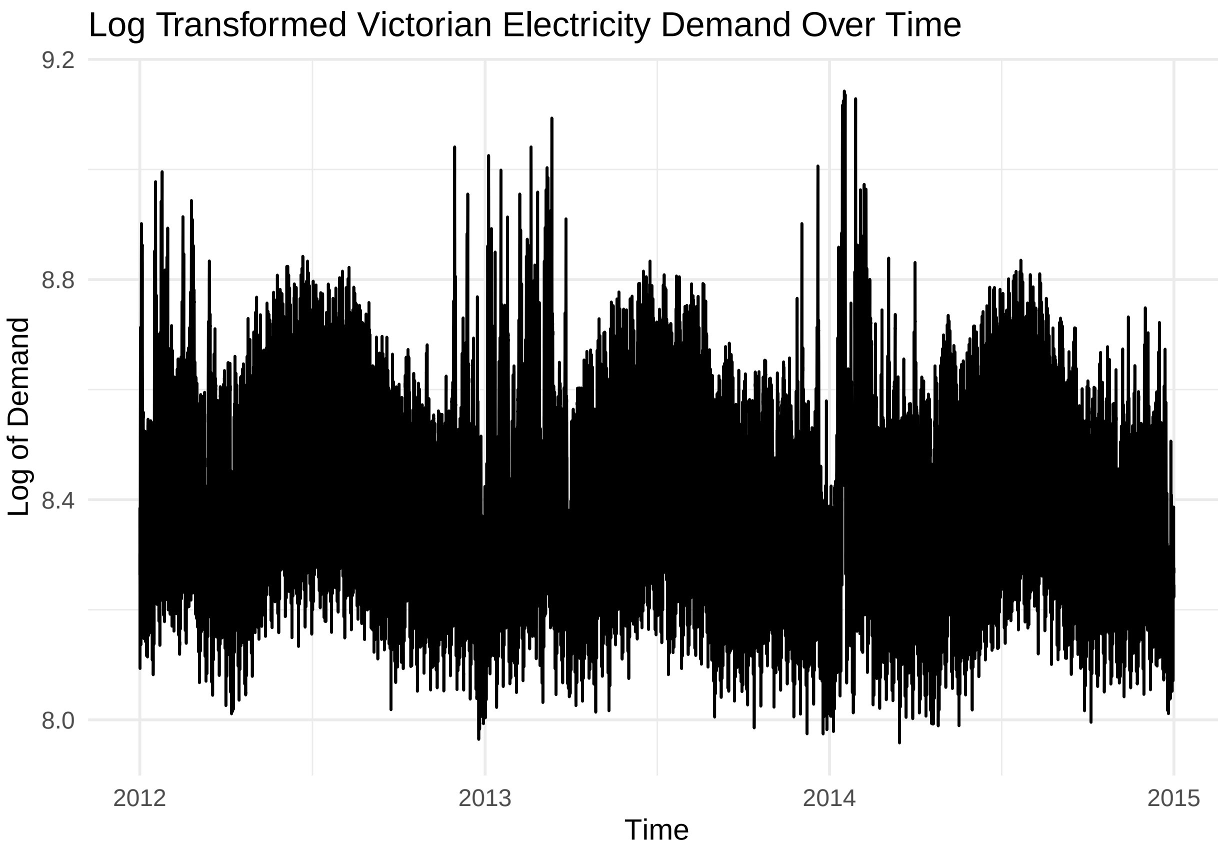
```
## # A tsibble: 52,608 x 5 [30m] <Australia/Melbourne>
##    Time                Demand Temperature Date       Holiday
##    <dttm>               <dbl>       <dbl> <date>     <lgl>
##  1 2012-01-01 00:00:00  4383.        21.4 2012-01-01 TRUE
##  2 2012-01-01 00:30:00  4263.        21.0 2012-01-01 TRUE
##  3 2012-01-01 01:00:00  4049.        20.7 2012-01-01 TRUE
##  4 2012-01-01 01:30:00  3878.        20.6 2012-01-01 TRUE
##  5 2012-01-01 02:00:00  4036.        20.4 2012-01-01 TRUE
##  6 2012-01-01 02:30:00  3866.        20.2 2012-01-01 TRUE
##  7 2012-01-01 03:00:00  3694.        20.1 2012-01-01 TRUE
##  8 2012-01-01 03:30:00  3562.        19.6 2012-01-01 TRUE
##  9 2012-01-01 04:00:00  3433.        19.1 2012-01-01 TRUE
## 10 2012-01-01 04:30:00  3359.        19.0 2012-01-01 TRUE
## # i 52,598 more rows
```

```
# Extract Victorian electricity demand data
vic_elec <- vic_elec |>
dplyr::select(Time, Demand)

# Plot original electricity demand data
```

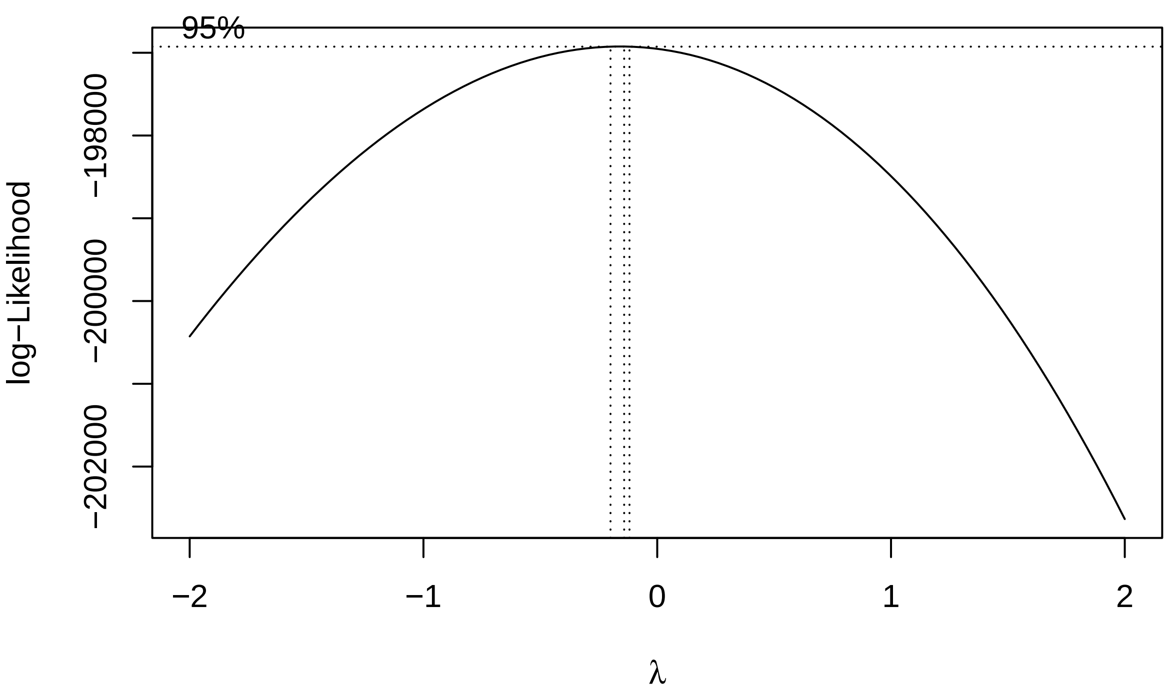
```r
ggplot(vic_elec, aes(x = Time, y = Demand)) +
  geom_line() +
  labs(title = "Victorian Electricity Demand Over Time", x = "Time", y = "Demand") +
  theme_minimal()
```



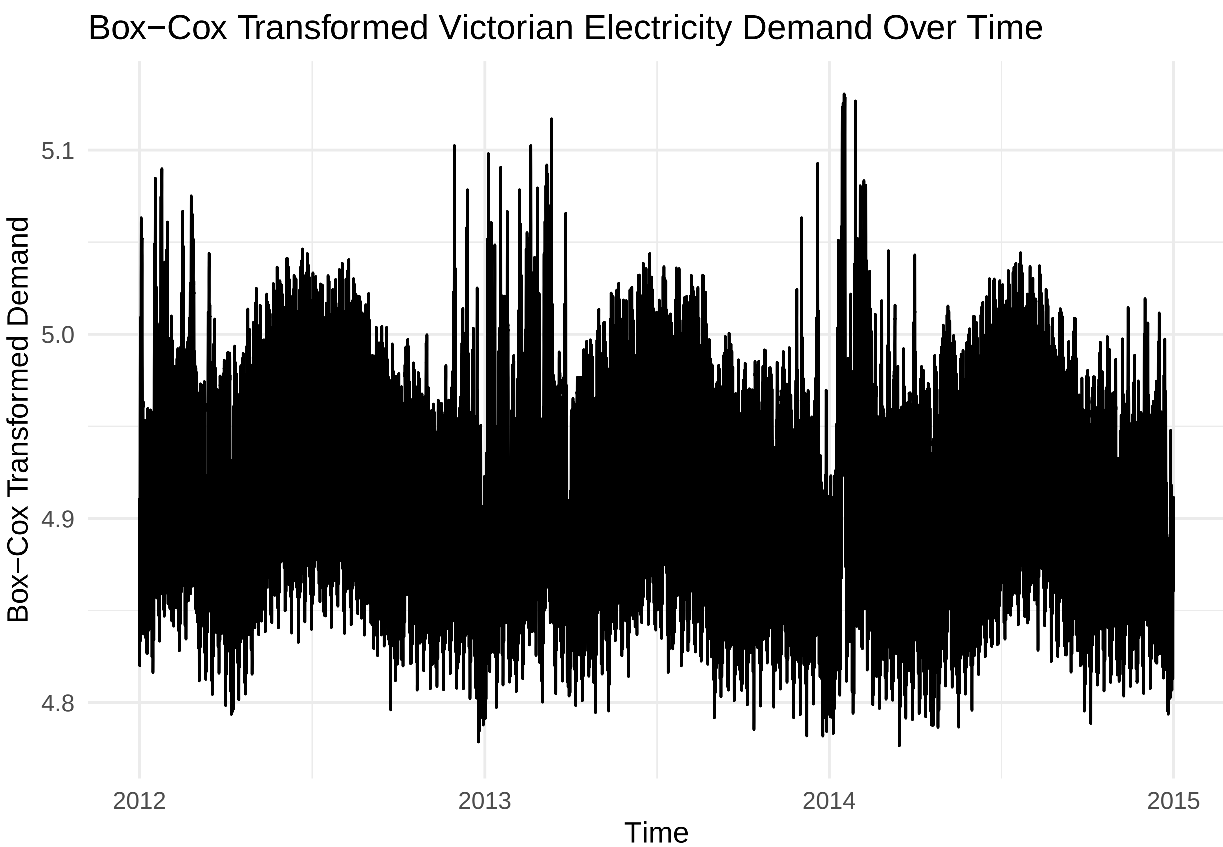

```r
# Log Transformation
vic_elec_log <- vic_elec |>
  mutate(Demand_Log = log(Demand))

# Plot Log-transformed electricity demand
ggplot(vic_elec_log, aes(x = Time, y = Demand_Log)) +
  geom_line() +
  labs(title = "Log Transformed Victorian Electricity Demand Over Time", x = "Time", y = "Log of Demand
  theme_minimal()
```



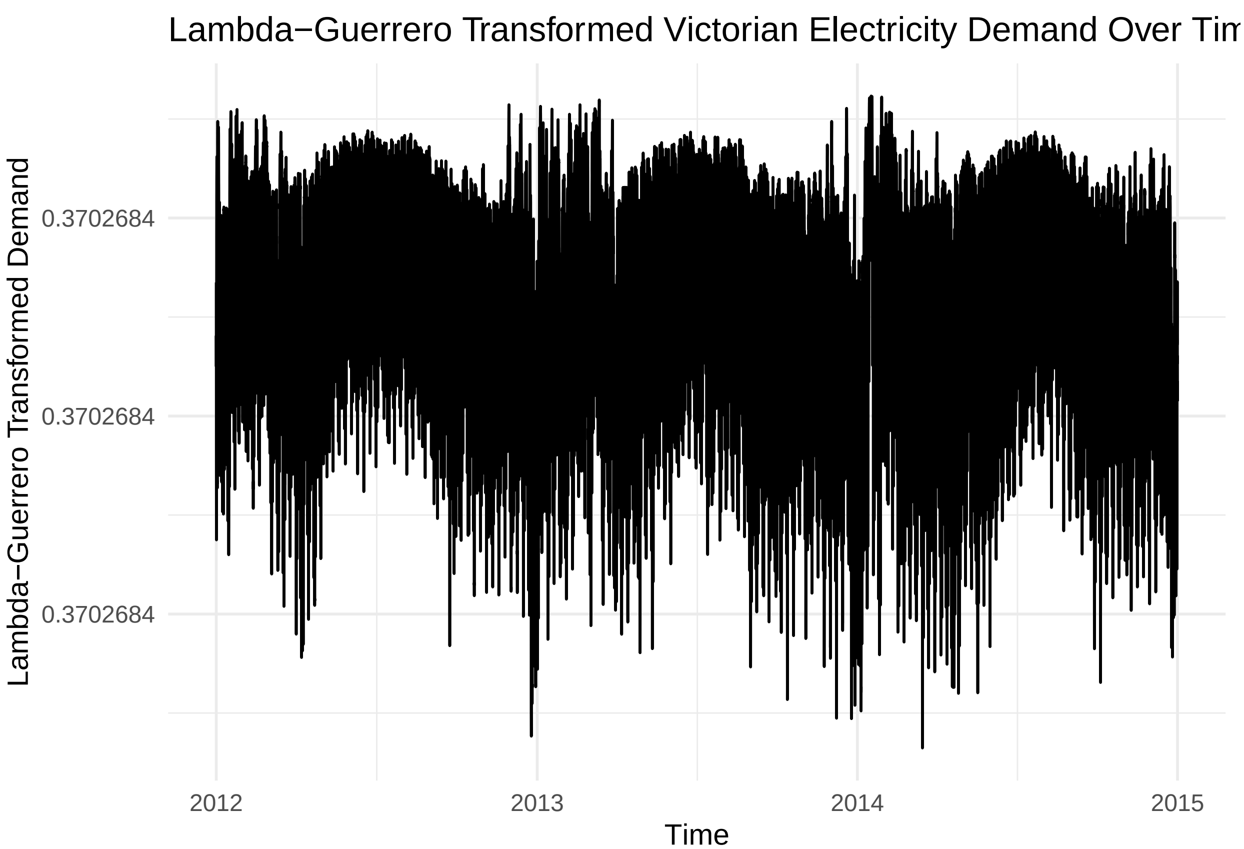

```
# Fit a linear model to use with Box-Cox
fit_vic_elec <- lm(Demand ~ Time, data = vic_elec)

# Apply Box-Cox transformation and find the optimal lambda
boxcox_result <- boxcox(fit_vic_elec, lambda = seq(-2, 2, 1/10))
```



```r
lambda_boxcox <- boxcox_result$x[which.max(boxcox_result$y)]

# Apply Box-Cox transformation with the optimal lambda
vic_elec_transformed_boxcox <- vic_elec |>
  mutate(Demand_BoxCox = (Demand^lambda_boxcox - 1) / lambda_boxcox)

# Plot Box-Cox transformed data
ggplot(vic_elec_transformed_boxcox, aes(x = Time, y = Demand_BoxCox)) +
  geom_line() +
  labs(title = "Box-Cox Transformed Victorian Electricity Demand Over Time", x = "Time", y = "Box-Cox Tr
  theme_minimal()
```




```r
# Calculate Lambda-Guerrero transformation
lambda_guerrero <- guerrero_lambda(vic_elec$Demand)

vic_elec_transformed_guerrero <- vic_elec |>
  mutate(Demand_Guerrero = guerrero_transform(Demand, lambda_guerrero))

# Plot Lambda-Guerrero transformed data
ggplot(vic_elec_transformed_guerrero, aes(x = Time, y = Demand_Guerrero)) +
  geom_line() +
  labs(title = "Lambda-Guerrero Transformed Victorian Electricity Demand Over Time", x = "Time", y = "La
  theme_minimal()
```



In conclusion, in vic_eclec, a transformation or differencing seems suitable to stabilize variance and emphasize seasonal trends.

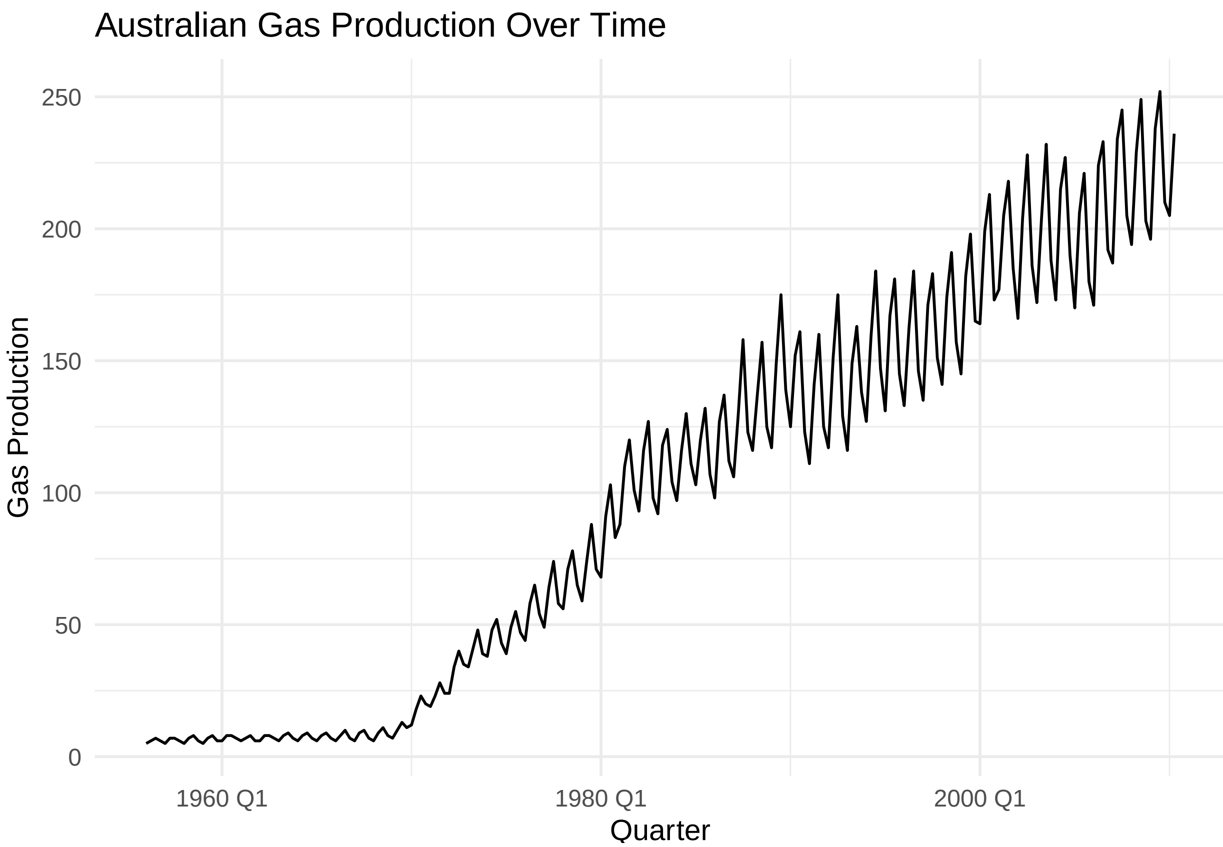
Gas production from aus_production.

```
aus_production
```

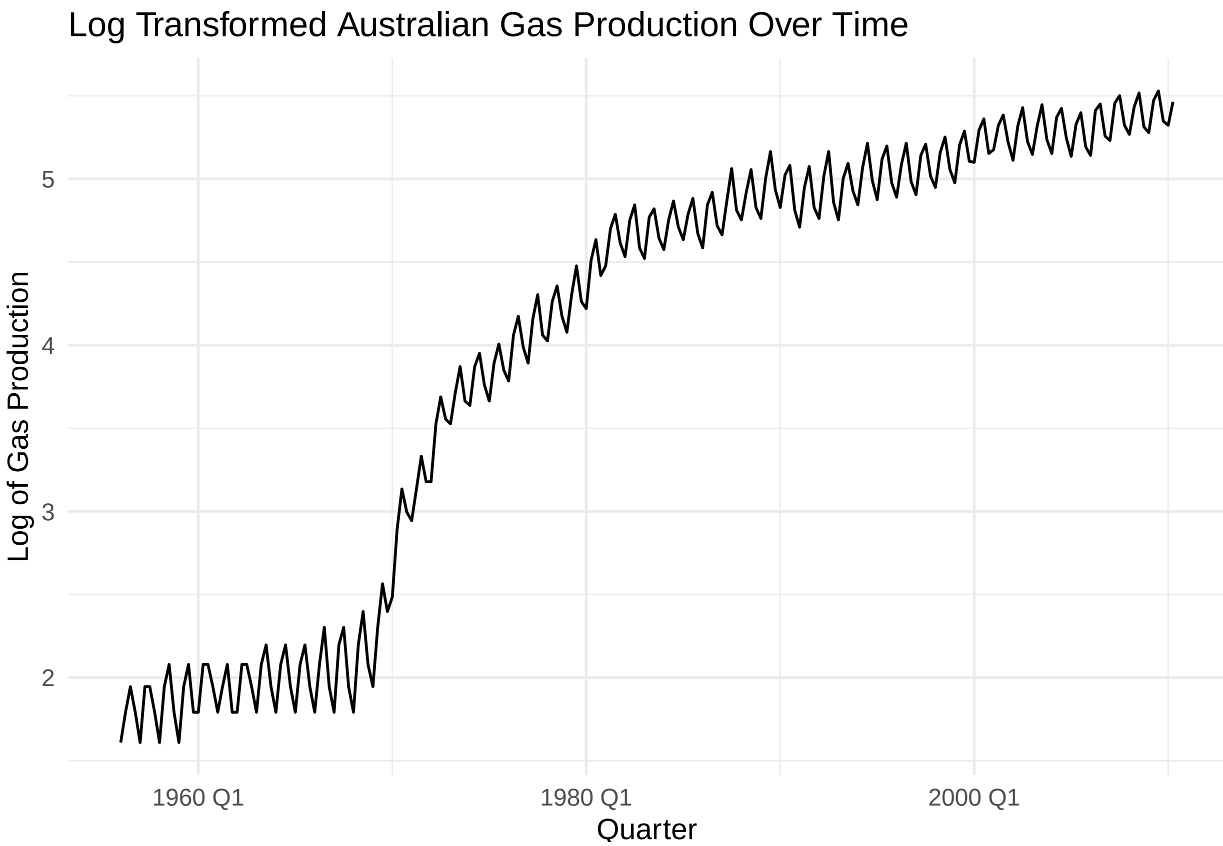
```
## # A tsibble: 218 x 7 [1Q]
##    Quarter  Beer Tobacco Bricks Cement Electricity   Gas
##      <qtr> <dbl>   <dbl>  <dbl>  <dbl>       <dbl> <dbl>
##  1 1956 Q1   284    5225    189    465        3923     5
##  2 1956 Q2   213    5178    204    532        4436     6
##  3 1956 Q3   227    5297    208    561        4806     7
##  4 1956 Q4   308    5681    197    570        4418     6
##  5 1957 Q1   262    5577    187    529        4339     5
##  6 1957 Q2   228    5651    214    604        4811     7
##  7 1957 Q3   236    5317    227    603        5259     7
##  8 1957 Q4   320    6152    222    582        4735     6
##  9 1958 Q1   272    5758    199    554        4608     5
## 10 1958 Q2   233    5641    229    620        5196     7
## # i 208 more rows
```

```
# Extract Australian gas production data
aus_production <- aus_production |>
 dplyr:: select(Quarter, Gas)

# Plot original gas production data
```

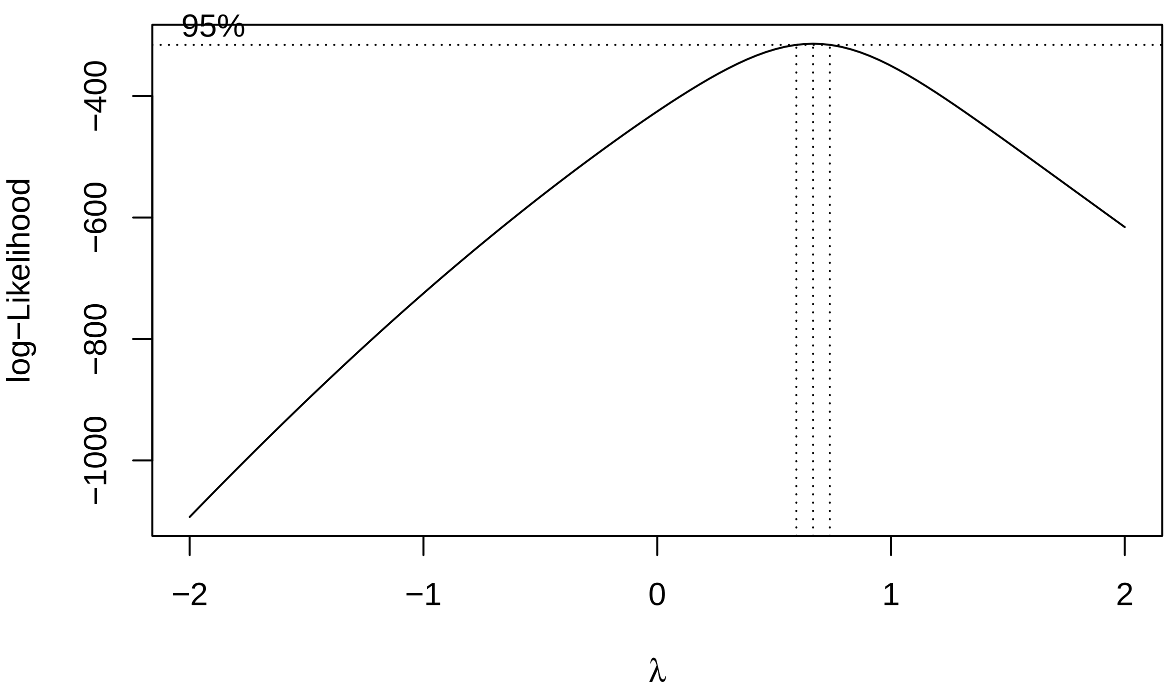
```r
ggplot(aus_production, aes(x = Quarter, y = Gas)) +
  geom_line() +
  labs(title = "Australian Gas Production Over Time", x = "Quarter", y = "Gas Production") +
  theme_minimal()
```




```r
# Log Transformation
aus_production_log <- aus_production %>%
  mutate(Gas_Log = log(Gas))

# Plot Log-transformed gas production
ggplot(aus_production_log, aes(x = Quarter, y = Gas_Log)) +
  geom_line() +
  labs(title = "Log Transformed Australian Gas Production Over Time", x = "Quarter", y = "Log of Gas Pro
  theme_minimal()
```

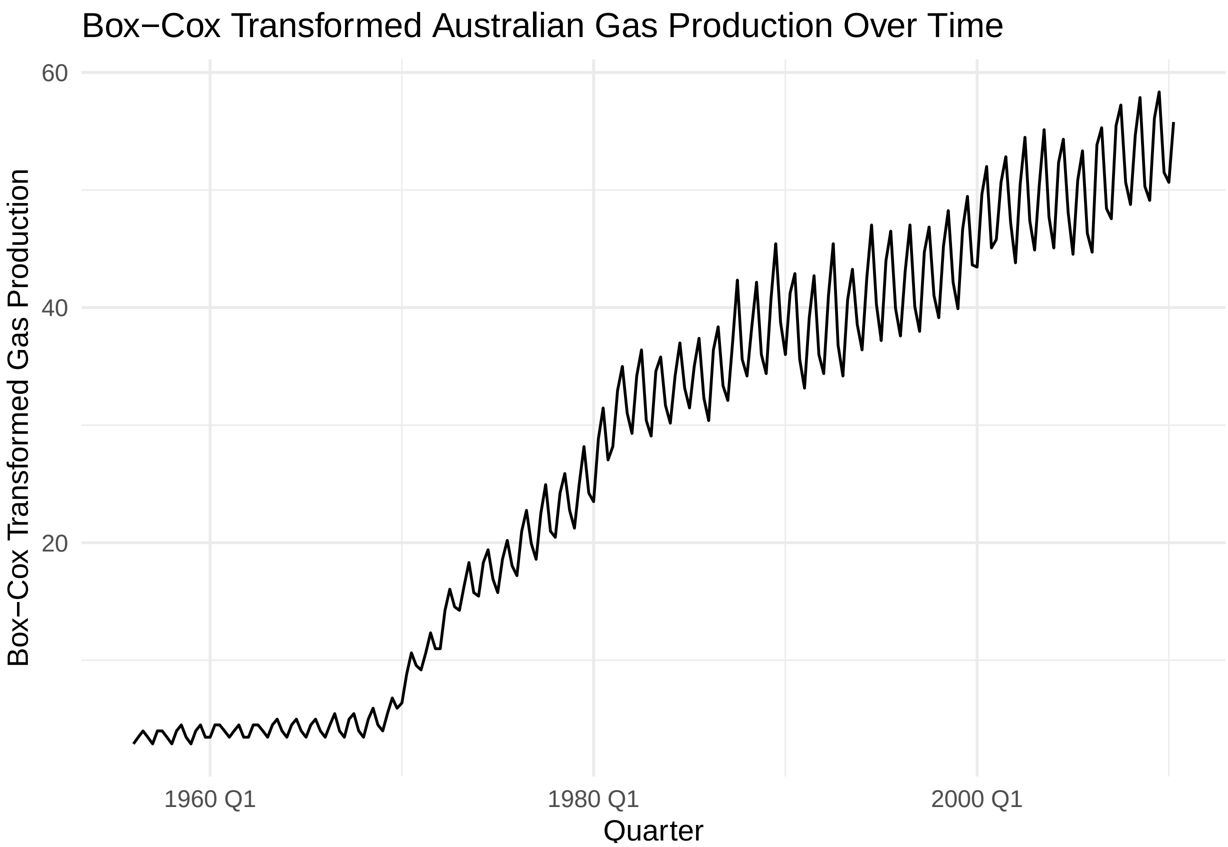



```r
# Fit a linear model to use with Box-Cox
fit_aus_production <- lm(Gas ~ Quarter, data = aus_production)

# Apply Box-Cox transformation and find the optimal lambda
boxcox_result <- boxcox(fit_aus_production, lambda = seq(-2, 2, 1/10))
```

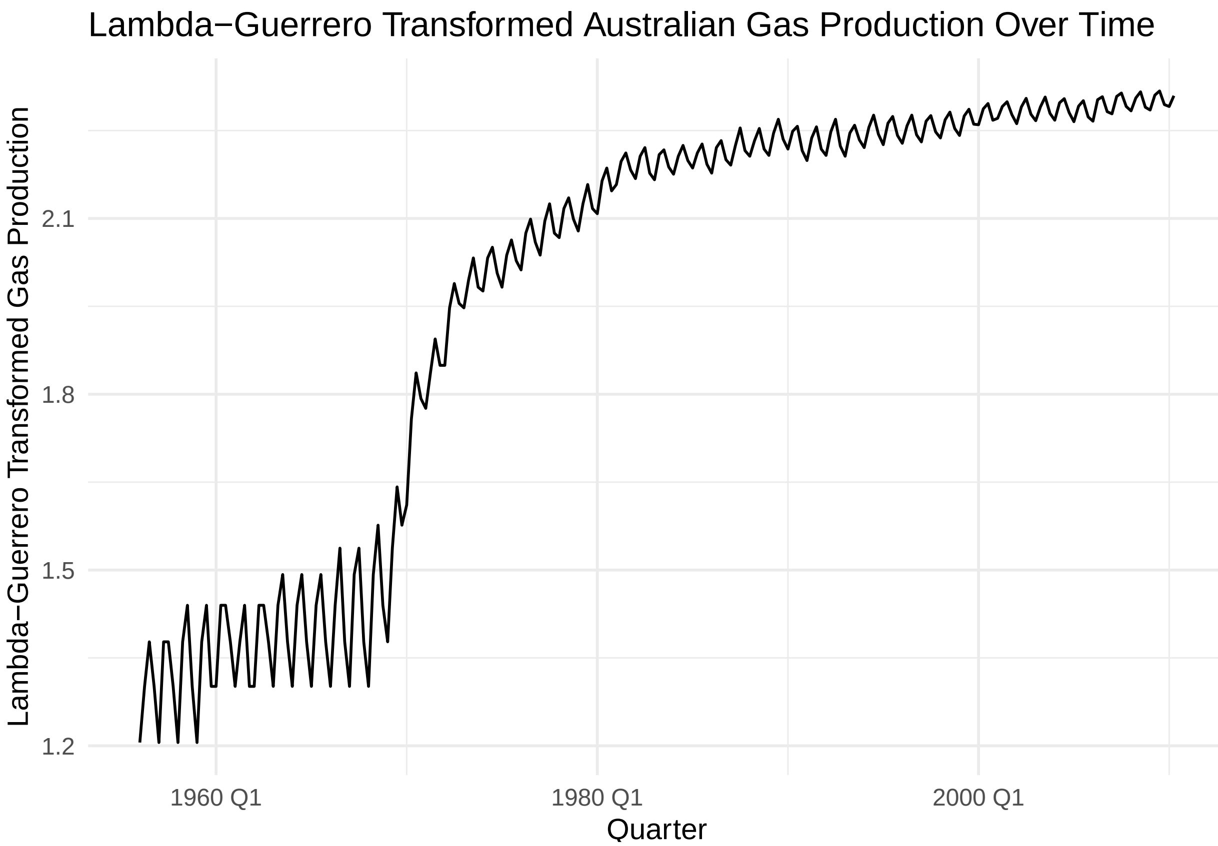


```
lambda_boxcox <- boxcox_result$x[which.max(boxcox_result$y)]

# Apply Box-Cox transformation with the optimal lambda
aus_production_transformed_boxcox <- aus_production |>
  mutate(Gas_BoxCox = (Gas^lambda_boxcox - 1) / lambda_boxcox)
```

```
# Lambda-Guerrero Transformation
lambda_guerrero <- guerrero_lambda(aus_production$Gas)

aus_production_transformed_guerrero <- aus_production |>
  mutate(Gas_Guerrero = guerrero_transform(Gas, lambda_guerrero))

# Plot Box-Cox transformed data
ggplot(aus_production_transformed_boxcox, aes(x = Quarter, y = Gas_BoxCox)) +
  geom_line() +
  labs(title = "Box-Cox Transformed Australian Gas Production Over Time", x = "Quarter", y = "Box-Cox Tr
  theme_minimal()
```




```
# Plot Lambda-Guerrero transformed data
ggplot(aus_production_transformed_guerrero, aes(x = Quarter, y = Gas_Guerrero)) +
  geom_line() +
  labs(title = "Lambda-Guerrero Transformed Australian Gas Production Over Time", x = "Quarter", y = "La
  theme_minimal()
```



In conclusion, in aus_production, a transformation appears to be useful in order to handle exponential growth and stabilize variance.

Exercise 3.3: Why is a Box-Cox transformation unhelpful for the canadian_gas data?

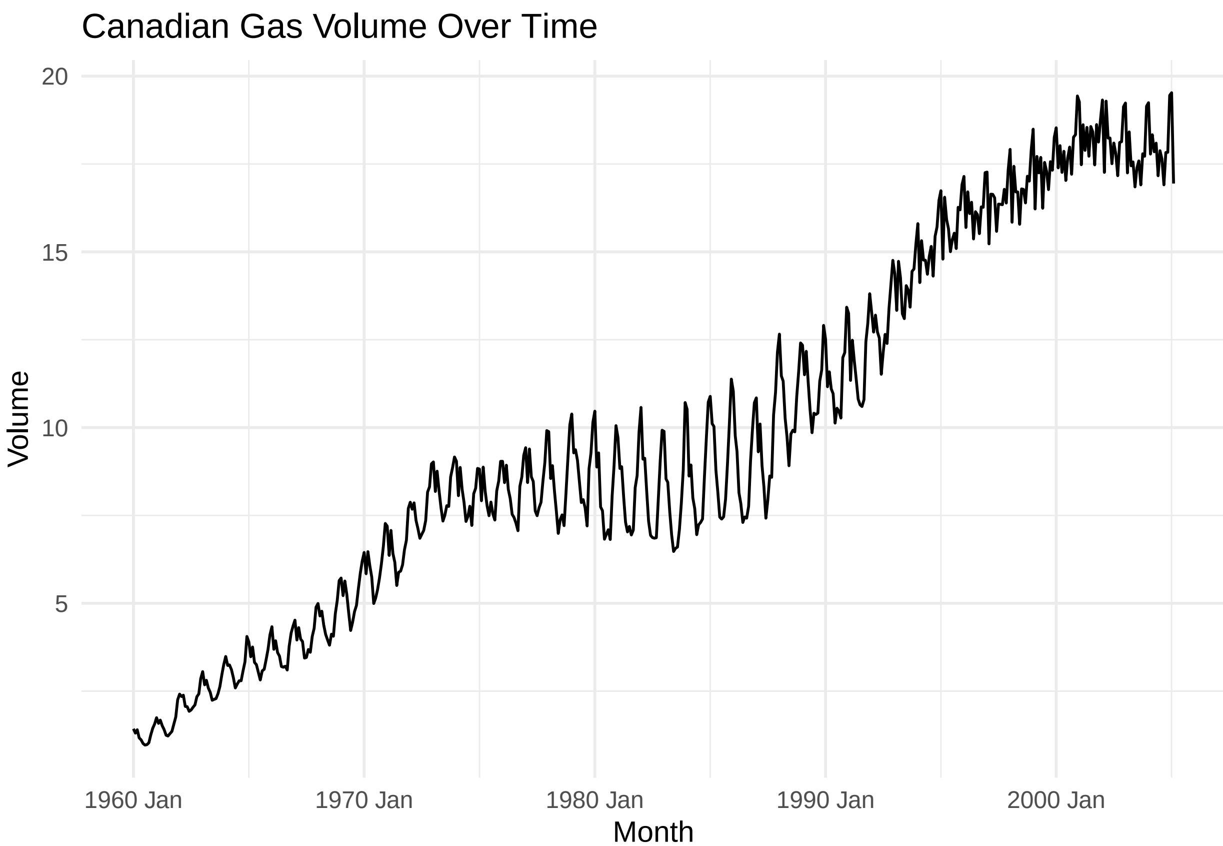
```
canadian_gas
```

```
## # A tsibble: 542 x 2 [1M]
##       Month Volume
##       <mth>  <dbl>
##  1 1960 Jan  1.43
##  2 1960 Feb  1.31
##  3 1960 Mar  1.40
##  4 1960 Apr  1.17
##  5 1960 May  1.12
##  6 1960 Jun  1.01
##  7 1960 Jul  0.966
##  8 1960 Aug  0.977
##  9 1960 Sep  1.03
## 10 1960 Oct  1.25
## # i 532 more rows
```

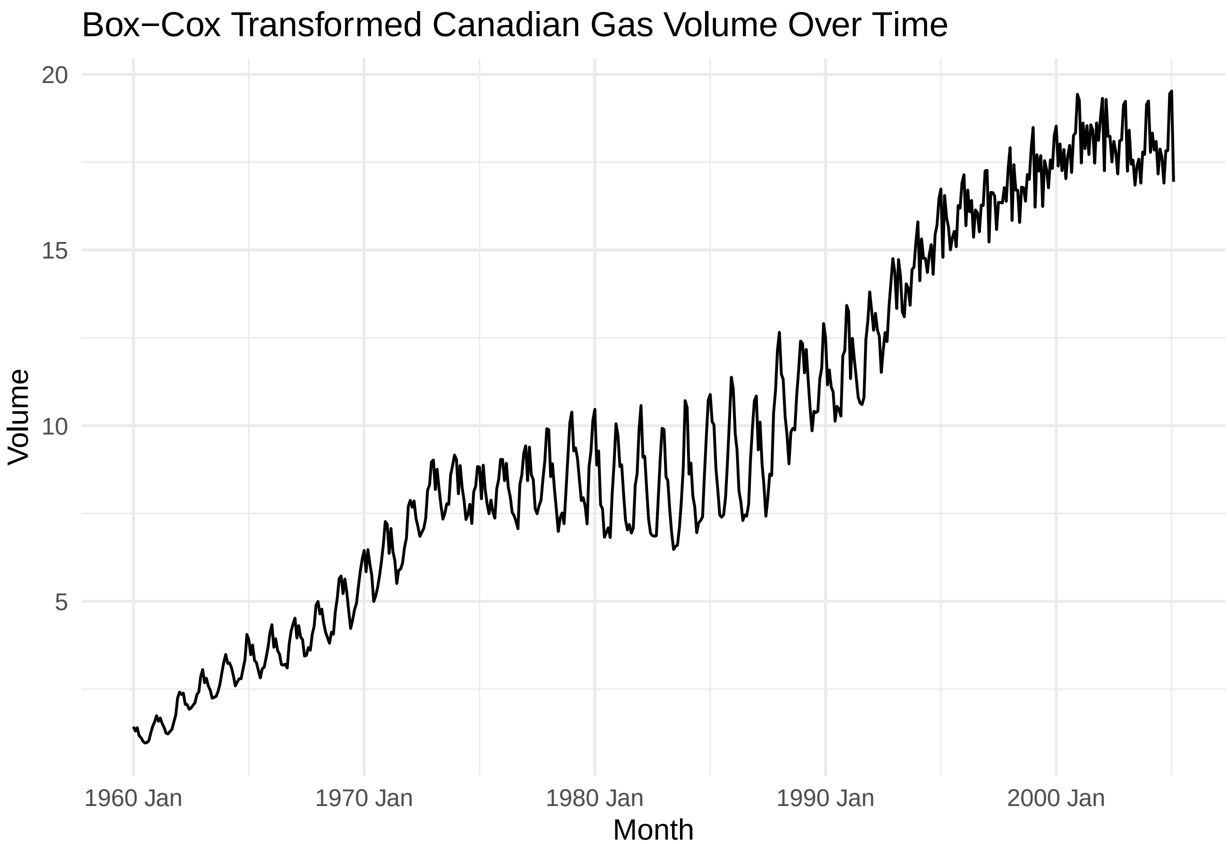
A box tranformation is unhelpful for the canadian_gas dataset, because the data values are already close to 1.

Indeed, the values in the `Volume` column are small and close to **1** (e.g., 1.43, 1.30, 1.40, etc.). For values near 1, the box cox transformation (especially with lambda approximately equal to 0) would result in very minor changes to the data. Also, when values are already in a small range, transformations like Box-Cox won't significantly alter the data, which makes the transformation less useful. The primary benefit of Box-Cox (variance stabilization and normalizing skewed data) won't be realized here.

```r
# Plot original data
ggplot(canadian_gas, aes(x = Month, y = Volume)) +
  geom_line() +
  labs(title = "Canadian Gas Volume Over Time", x = "Month", y = "Volume") +
  theme_minimal()
```




```r
# Apply Box-Cox transformation with the optimal lambda
Volume <- canadian_gas$Volume
 canadian_transformed_boxcox <- canadian_gas |>
  mutate(Gas_BoxCox = (Volume^lambda_boxcox - 1) / lambda_boxcox)

 ggplot(canadian_transformed_boxcox , aes(x = Month, y = Volume)) +
  geom_line() +
  labs(title = "Box-Cox Transformed Canadian Gas Volume Over Time", x = "Month", y = "Volume") +
  theme_minimal()
```



Exercise 3.4: What Box-Cox transformation would you select for your retail data aus_retail?

```
aus_retail
```

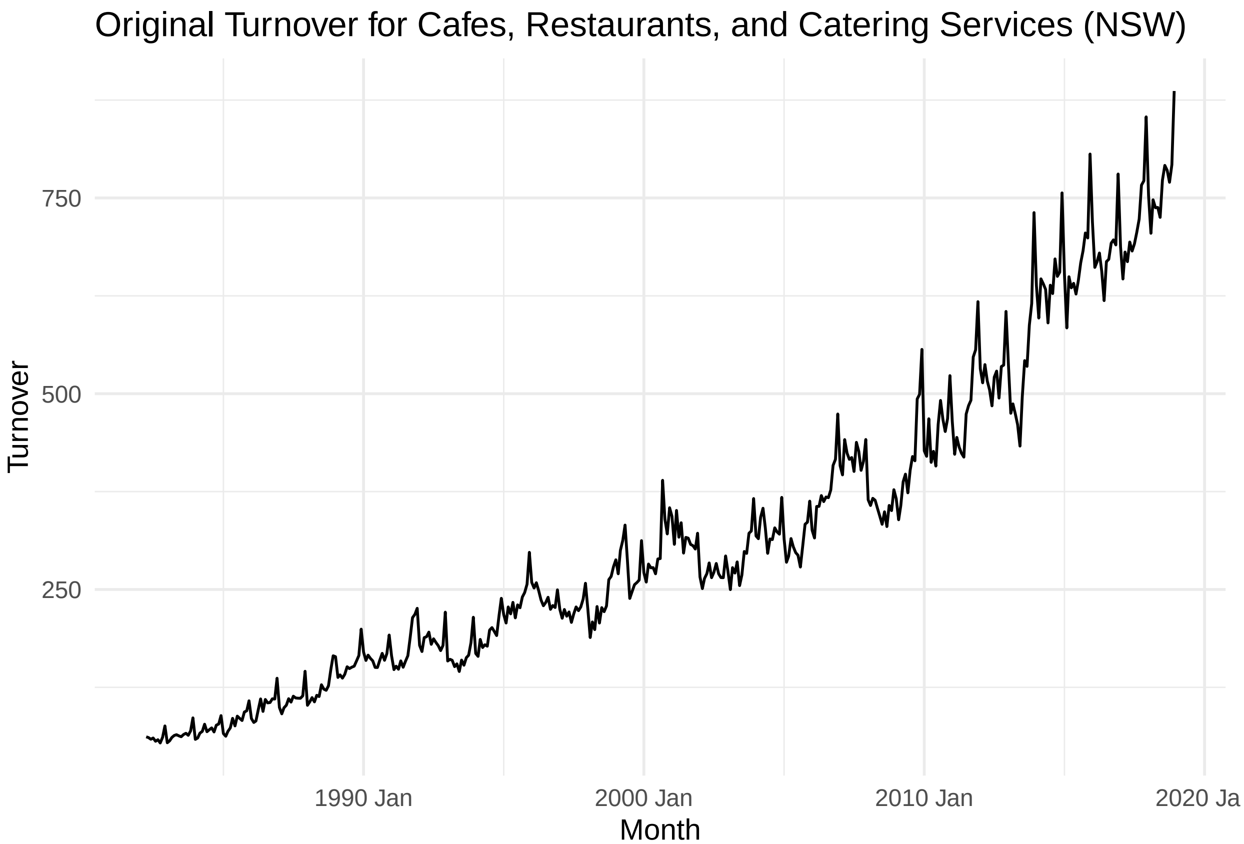
```
## # A tsibble: 64,532 x 5 [1M]
## # Key:       State, Industry [152]
##    State                        Industry           `Series ID`    Month Turnover
##    <chr>                        <chr>              <chr>          <mth>    <dbl>
##  1 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Apr      4.4
##  2 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 May      3.4
##  3 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Jun      3.6
##  4 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Jul      4
##  5 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Aug      3.6
##  6 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Sep      4.2
##  7 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Oct      4.8
##  8 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Nov      5.4
##  9 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Dec      6.9
## 10 Australian Capital Territory Cafes, restaurant~ A3349849A   1983 Jan      3.8
## # i 64,522 more rows
```

To determine the appropriate Box - Cox transformation to use,for the `aus retail` dataset, I ideally need to analyze seasonal patterns or trend decomposition and then calculate the optimal lambda. This value of lambda would indicate the power to which the data should be transformed to stabilize variance and make the data more normally distributed.

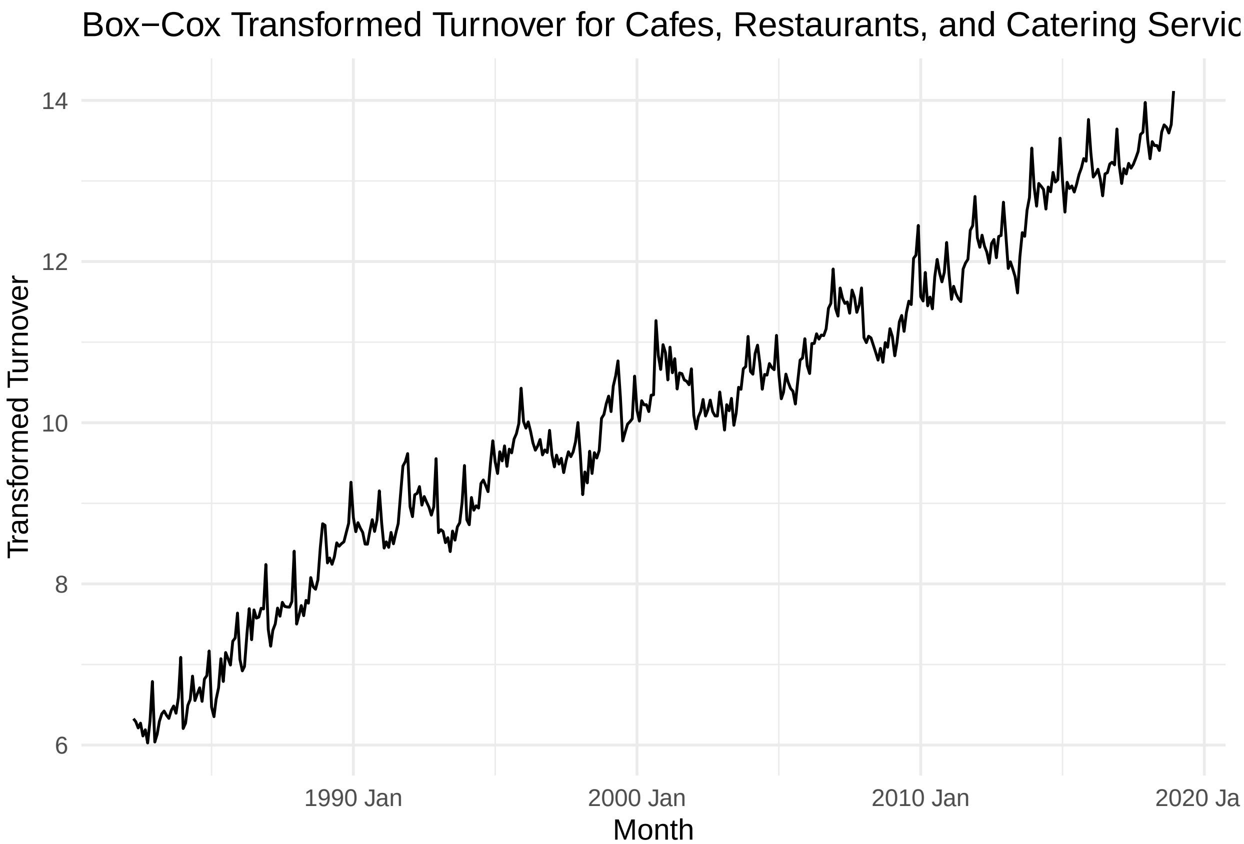
```
# Filter out a subset for simplicity (e.g., a specific state and industry)
aus_retail_filtered <- aus_retail %>%
  filter(State == "New South Wales", Industry == "Cafes, restaurants and catering services")

# Calculate the optimal lambda using guerrero
lambda <- aus_retail_filtered |>
  features(Turnover, features = guerrero) |>
  pull(lambda_guerrero)

# Print the optimal lambda value
lambda
```

```
## [1] 0.1946443
```

```
# Apply the Box-Cox transformation using the calculated lambda
aus_retail_transformed <- aus_retail_filtered |>
  mutate(Turnover_BoxCox = box_cox(Turnover, lambda))

# Plot the original and transformed data

ggplot(aus_retail_filtered, aes(x = Month, y = Turnover)) +
  geom_line() +
  labs(title = "Original Turnover for Cafes, Restaurants, and Catering Services (NSW)",
       x = "Month", y = "Turnover") +
  theme_minimal()
```




```r
ggplot(aus_retail_transformed, aes(x = Month, y = Turnover_BoxCox)) +
  geom_line() +
  labs(title = "Box-Cox Transformed Turnover for Cafes, Restaurants, and Catering Services (NSW)",
       x = "Month", y = "Transformed Turnover") +
  theme_minimal()
```



Exercise 3.5:

For the following series, find an appropriate Box-Cox transformation in order to stabilize the variance. Tobacco from aus_ production, Economy class passengers between Melbourne and Sydney from ansett, and Pedestrian counts at Southern Cross Station from pedestrian.

Tobacco (aus_production): Finding the appropriate lambda for stabilizing the variance in tobacco production.

```
aus_production
```

```
## # A tsibble: 218 x 2 [1Q]
##    Quarter   Gas
##      <qtr> <dbl>
##  1 1956 Q1     5
##  2 1956 Q2     6
##  3 1956 Q3     7
##  4 1956 Q4     6
##  5 1957 Q1     5
##  6 1957 Q2     7
##  7 1957 Q3     7
##  8 1957 Q4     6
##  9 1958 Q1     5
## 10 1958 Q2     7
## # i 208 more rows
```

Economy class passengers (ansett): Finding the appropriate lambda for economy class passengers between Melbourne and Sydney.

Since the tobacco column is missing, I will apply the box cox transformation to the existing gas column in order to stabilize the variance in the data set.

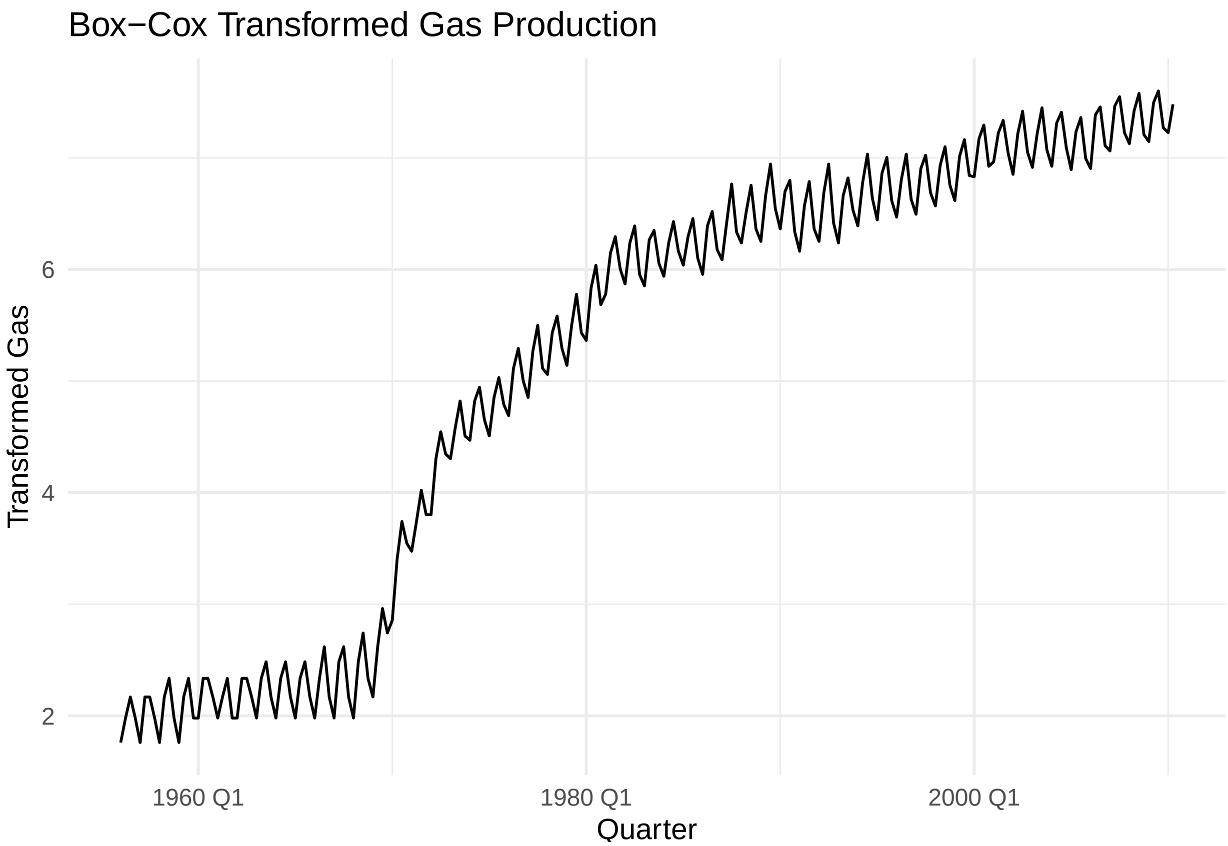
```r
# Calculate the optimal lambda for the 'Gas' column using Guerrero's method
gas_lambda <- aus_production |>
  features(Gas, guerrero) |>
  pull(lambda_guerrero)

# Print the optimal lambda for Gas
cat("Optimal lambda for Gas:", gas_lambda, "\n")
```

```
## Optimal lambda for Gas: 0.1095171
```

```r
# Apply Box-Cox transformation to the 'Gas' column
aus_production_transformed <- aus_production %>%
  mutate(Gas_BoxCox = box_cox(Gas, gas_lambda))

# Check the transformed dataset
glimpse(aus_production_transformed)
```

```
## Rows: 218
## Columns: 3
## $ Quarter    <qtr> 1956 Q1, 1956 Q2, 1956 Q3, 1956 Q4, 1957 Q1, 1957 Q2, 1957 ~
## $ Gas        <dbl> 5, 6, 7, 6, 5, 7, 7, 6, 5, 7, 8, 6, 5, 7, 8, 6, 6, 8, 8, 7,~
## $ Gas_BoxCox <dbl> 1.759993, 1.979642, 2.168806, 1.979642, 1.759993, 2.168806,~
```

```r
# Then, plot the transformed data
ggplot(aus_production_transformed, aes(x = Quarter, y = Gas_BoxCox)) +
  geom_line() +
  labs(title = "Box-Cox Transformed Gas Production",
       y = "Transformed Gas", x = "Quarter") +
  theme_minimal()
```



Pedestrian counts (pedestrian): Finding the lambda for pedestrian counts at Southern Cross Station.

```
ansett
```

```
## # A tsibble: 7,407 x 4 [1W]
## # Key:       Airports, Class [30]
##         Week Airports Class    Passengers
##       <week> <chr>    <chr>         <dbl>
##  1 1989 W28 ADL-PER  Business        193
##  2 1989 W29 ADL-PER  Business        254
##  3 1989 W30 ADL-PER  Business        185
##  4 1989 W31 ADL-PER  Business        254
##  5 1989 W32 ADL-PER  Business        191
##  6 1989 W33 ADL-PER  Business        136
##  7 1989 W34 ADL-PER  Business          0
##  8 1989 W35 ADL-PER  Business          0
##  9 1989 W36 ADL-PER  Business          0
## 10 1989 W37 ADL-PER  Business          0
## # i 7,397 more rows
```

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

# Print the optimal lambda for Economy class passengers
```

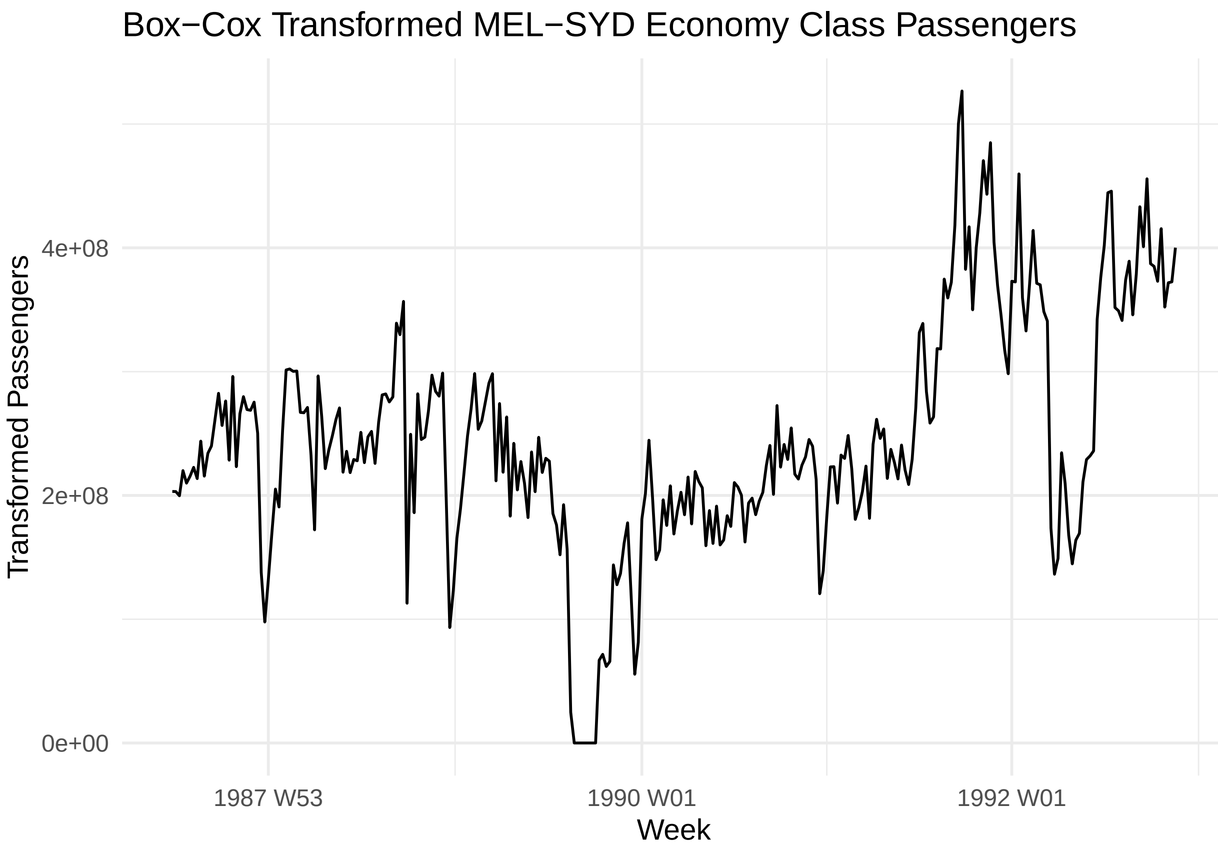
```r
cat("Optimal lambda for MEL-SYD Economy class passengers:", ansett_lambda, "\n")
```

```
## Optimal lambda for MEL-SYD Economy class passengers: 1.999927
```

```r
# Apply Box-Cox transformation for Economy class passengers
ansett_transformed <- ansett |>
  filter(Airports == "MEL-SYD", Class == "Economy") |>
  mutate(Passengers_BoxCox = box_cox(Passengers, ansett_lambda))
ansett_transformed
```

```
## # A tsibble: 282 x 5 [1W]
## # Key:       Airports, Class [1]
##        Week Airports Class   Passengers Passengers_BoxCox
##      <week> <chr>    <chr>        <dbl>             <dbl>
##  1 1987 W26 MEL-SYD  Economy      20167        203213842.
##  2 1987 W27 MEL-SYD  Economy      20161        203092945.
##  3 1987 W28 MEL-SYD  Economy      19993        199722456.
##  4 1987 W29 MEL-SYD  Economy      20986        220053743.
##  5 1987 W30 MEL-SYD  Economy      20497        209918530.
##  6 1987 W31 MEL-SYD  Economy      20770        215547379.
##  7 1987 W32 MEL-SYD  Economy      21111        222682889.
##  8 1987 W33 MEL-SYD  Economy      20675        213580173.
##  9 1987 W34 MEL-SYD  Economy      22092        243858478.
## 10 1987 W35 MEL-SYD  Economy      20772        215588890.
## # i 272 more rows
```

```r
# Plot for Economy class passengers
ggplot(ansett_transformed, aes(x = Week, y = Passengers_BoxCox)) +
  geom_line() +
  labs(title = "Box-Cox Transformed MEL-SYD Economy Class Passengers",
       y = "Transformed Passengers", x = "Week") +
  theme_minimal()
```




```
pedestrian
```

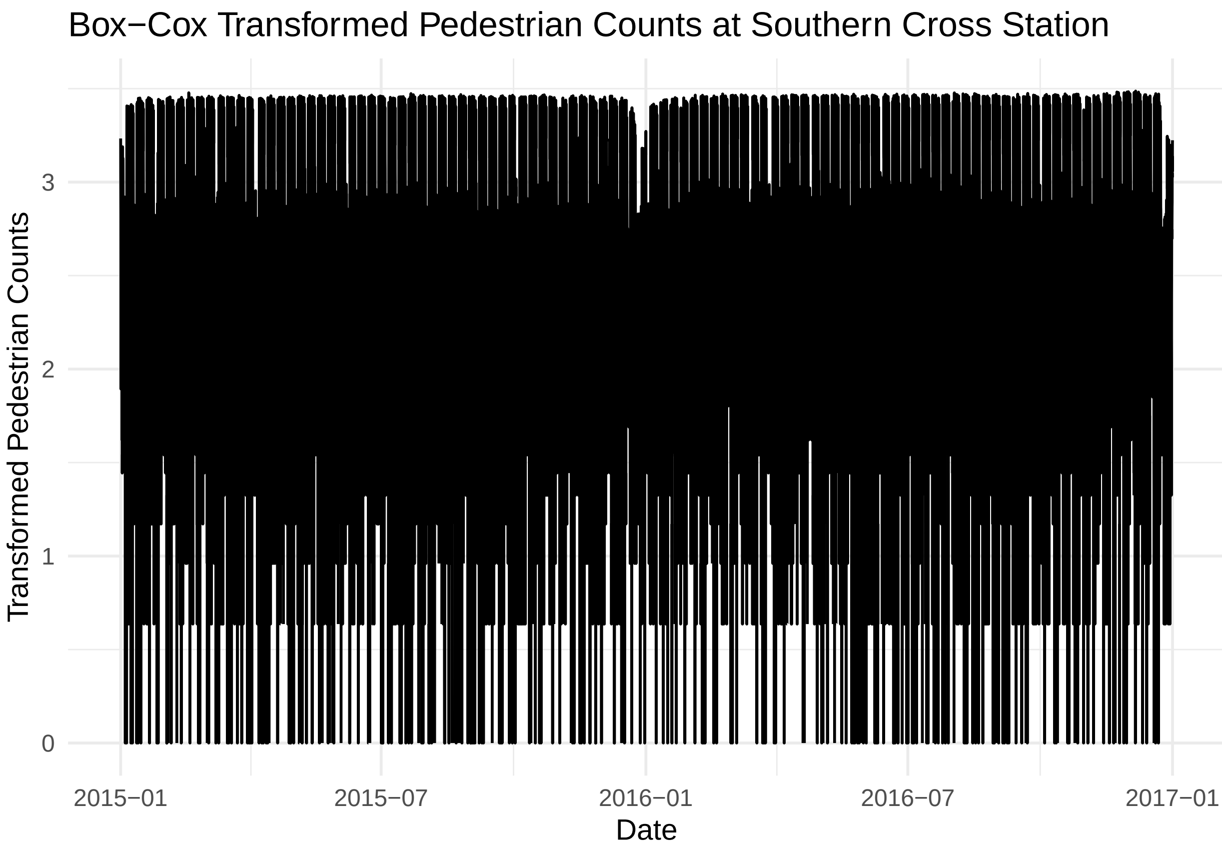
```
## # A tsibble: 66,037 x 5 [1h] <Australia/Melbourne>
## # Key:       Sensor [4]
##    Sensor         Date_Time           Date        Time Count
##    <chr>          <dttm>              <date>     <int> <int>
##  1 Birrarung Marr 2015-01-01 00:00:00 2015-01-01     0  1630
##  2 Birrarung Marr 2015-01-01 01:00:00 2015-01-01     1   826
##  3 Birrarung Marr 2015-01-01 02:00:00 2015-01-01     2   567
##  4 Birrarung Marr 2015-01-01 03:00:00 2015-01-01     3   264
##  5 Birrarung Marr 2015-01-01 04:00:00 2015-01-01     4   139
##  6 Birrarung Marr 2015-01-01 05:00:00 2015-01-01     5    77
##  7 Birrarung Marr 2015-01-01 06:00:00 2015-01-01     6    44
##  8 Birrarung Marr 2015-01-01 07:00:00 2015-01-01     7    56
##  9 Birrarung Marr 2015-01-01 08:00:00 2015-01-01     8   113
## 10 Birrarung Marr 2015-01-01 09:00:00 2015-01-01     9   166
## # i 66,027 more rows
```

```
pedestrian_lambda <- pedestrian |>
  filter(Sensor == "Southern Cross Station") |>
  features(Count, features = guerrero) |>
  pull(lambda_guerrero)

# Print the optimal lambda for Pedestrian counts
cat("Optimal lambda for Pedestrian counts at Southern Cross Station:", pedestrian_lambda, "\n")
```

```
## Optimal lambda for Pedestrian counts at Southern Cross Station: -0.2501616
```

```r
# Apply Box-Cox transformation for Pedestrian counts
pedestrian_transformed <- pedestrian |>
  filter(Sensor == "Southern Cross Station") |>
  mutate(Count_BoxCox = box_cox(Count, pedestrian_lambda))
```

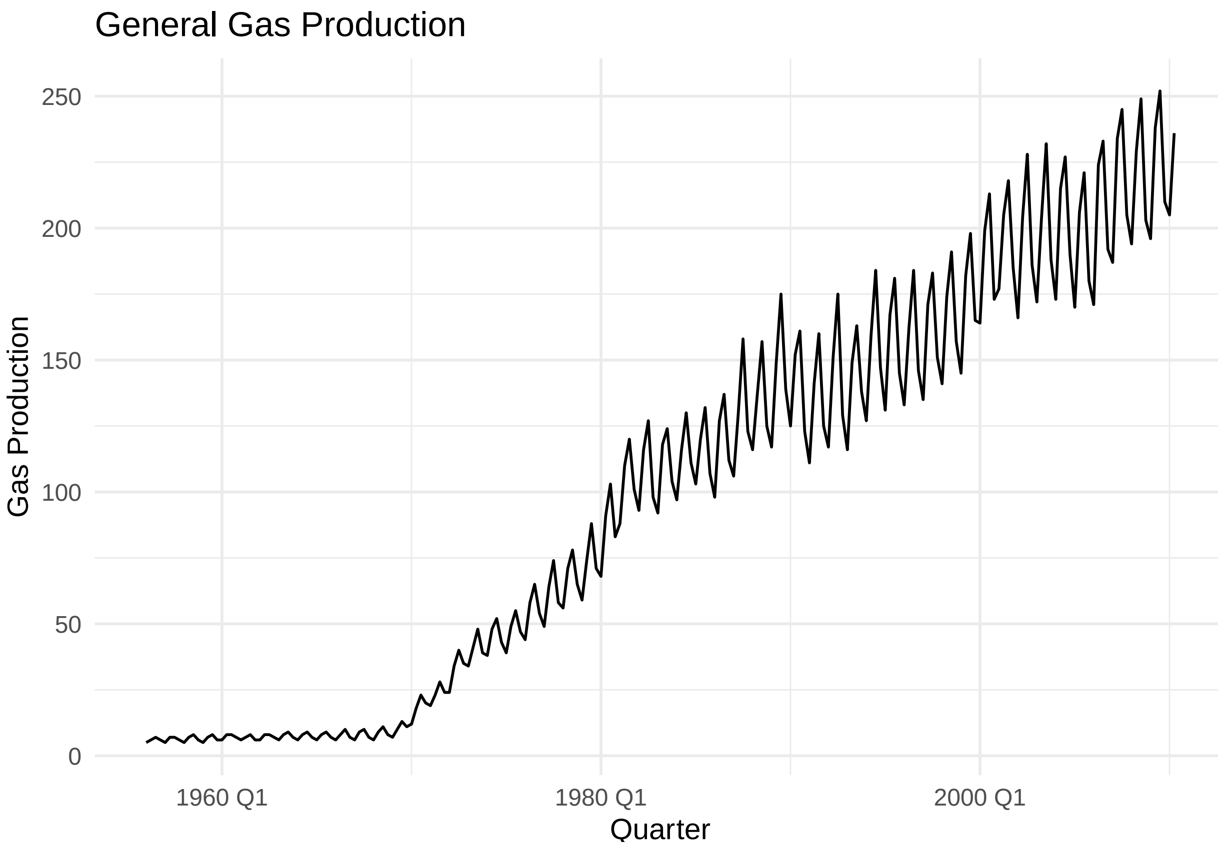
```r
# Plot for Pedestrian counts at Southern Cross Station
ggplot(pedestrian_transformed, aes(x = Date_Time, y = Count_BoxCox)) +
  geom_line() +
  labs(title = "Box-Cox Transformed Pedestrian Counts at Southern Cross Station",
       y = "Transformed Pedestrian Counts", x = "Date") +
  theme_minimal()
```



3.7: Consider the last five years of the Gas data from aus_ production.

Plot the time series. Can you identify seasonal fluctuations and/or a trend-cycle? Use classical_decomposition with type=multiplicative to calculate the trend-cycle and seasonal indices. Do the results support the graphical interpretation from part a? Compute and plot the seasonally adjusted data. Change one observation to be an outlier (e.g., add 300 to one observation), and recompute the seasonally adjusted data. What is the effect of the outlier? Does it make any difference if the outlier is near the end rather than in the middle of the time series?

```r
# General gas production plot
ggplot(aus_production, aes(x = Quarter, y = Gas)) +
  geom_line() +
  labs(title = "General Gas Production",
       y = "Gas Production", x = "Quarter") +
  theme_minimal()
```



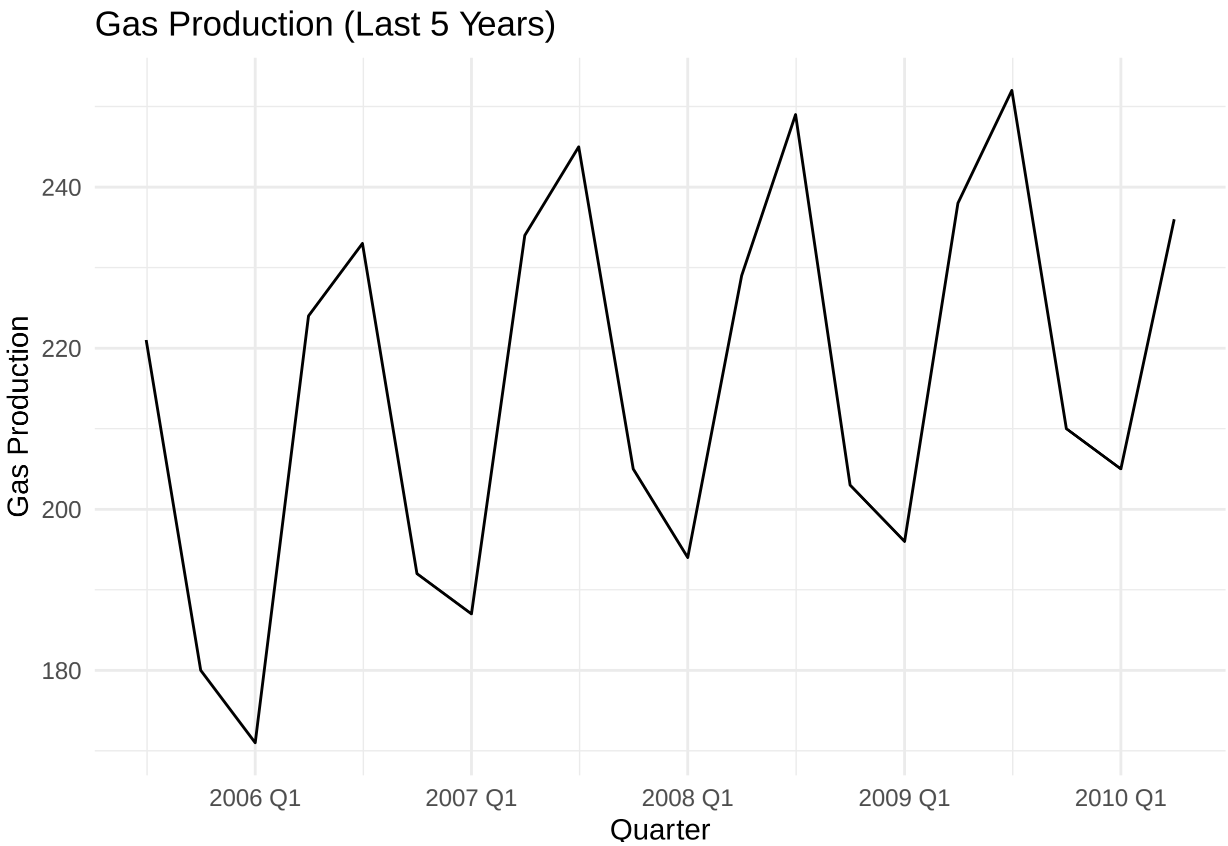
Extracting the 5 last years and plotting the series

```
# Extract the last five years of quarterly data
gas <- tail(aus_production, 5*4)|>
dplyr::select(Gas)
gas
```

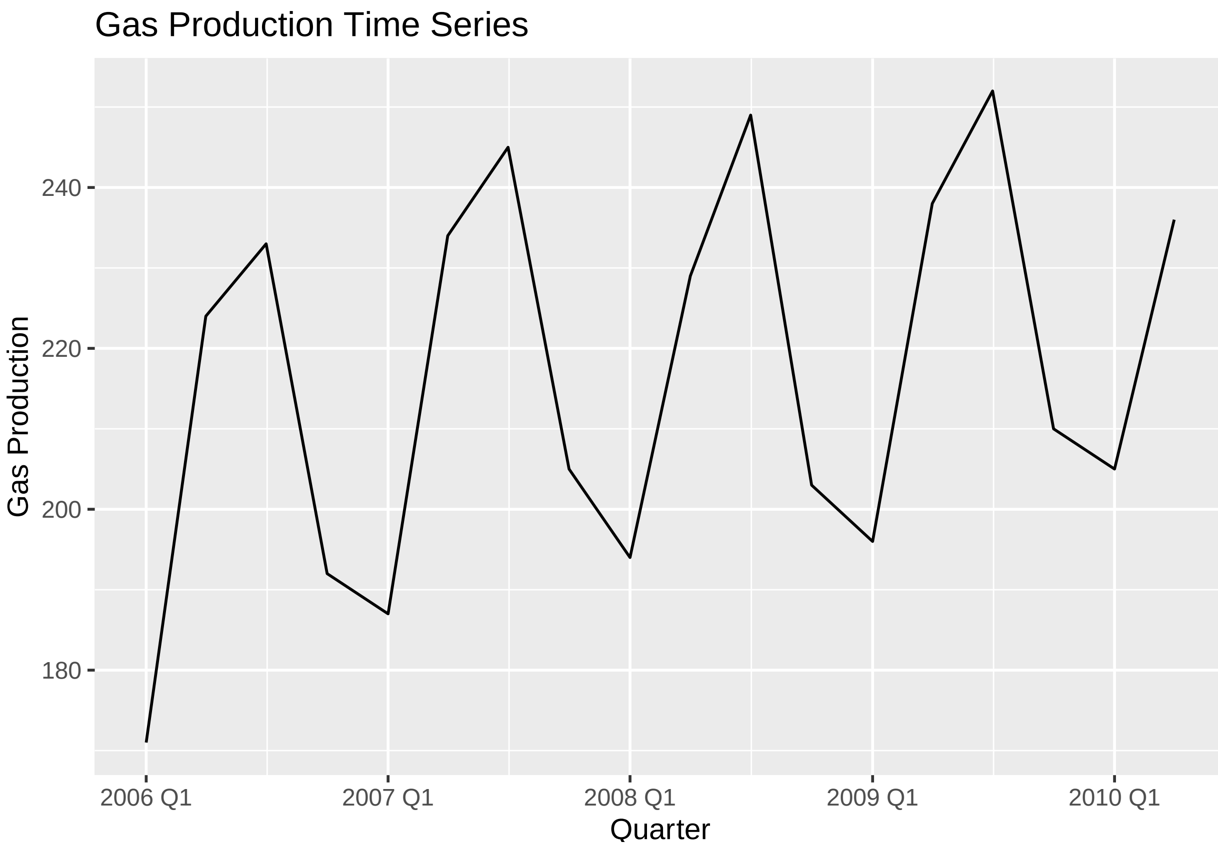
```
## # A tsibble: 20 x 2 [1Q]
##      Gas Quarter
##    <dbl>   <qtr>
##  1   221 2005 Q3
##  2   180 2005 Q4
##  3   171 2006 Q1
##  4   224 2006 Q2
##  5   233 2006 Q3
##  6   192 2006 Q4
##  7   187 2007 Q1
##  8   234 2007 Q2
##  9   245 2007 Q3
## 10   205 2007 Q4
## 11   194 2008 Q1
## 12   229 2008 Q2
## 13   249 2008 Q3
## 14   203 2008 Q4
## 15   196 2009 Q1
## 16   238 2009 Q2
## 17   252 2009 Q3
```

```
## 18   210 2009 Q4
## 19   205 2010 Q1
## 20   236 2010 Q2
```

```
# Plot the time series
ggplot(gas, aes(x = Quarter, y = Gas)) +
  geom_line() +
  labs(title = "Gas Production (Last 5 Years)",
       y = "Gas Production", x = "Quarter") +
  theme_minimal()
```




```
gas <- aus_production |> filter(Quarter >= yearquarter("2006 Q1"))

# Plot the time series
gas |>
  autoplot(Gas) +
  labs(title = "Gas Production Time Series",
       x = "Quarter",
       y = "Gas Production")
```



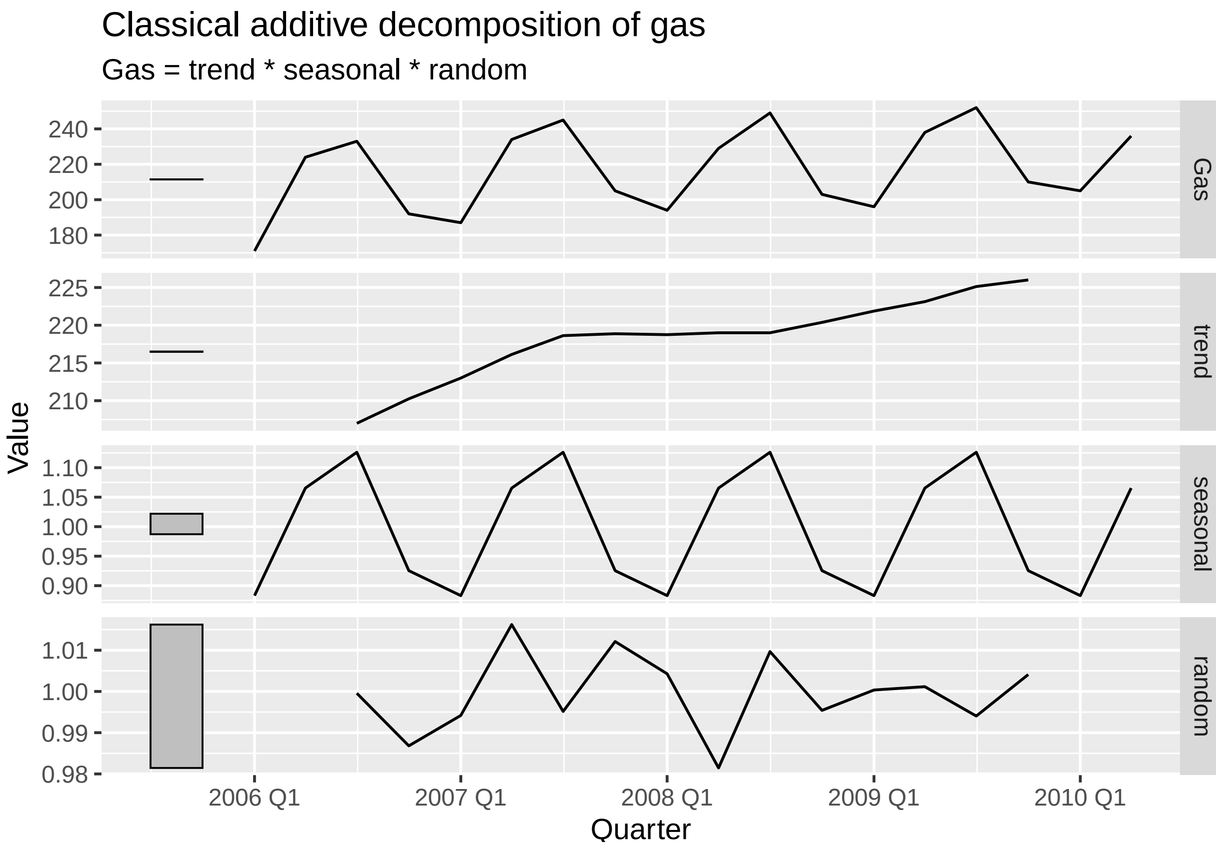

```r
# Classical decomposition with multiplicative type
decomposition <- gas |>
  model(
    classical_decomposition(Gas ~ trend(window = 8), type = "multiplicative")
  )
```

```
## Warning: 1 error encountered for classical_decomposition(Gas ~ trend(window = 8), type = "multiplica
## [1] Exogenous regressors are not supported for Classical decomposition.
```

Classical decomposition using a multiplicative model

```r
# classical_decomposition with type=multiplicative (to calculate the trend-cycle and seasonal indices).
gas |>
  model(classical_decomposition(Gas, type = "multiplicative")) |>
  components() |>
  autoplot() +
  labs(title = "Classical additive decomposition of gas",
       x = "Quarter",
       y = "Value")
```

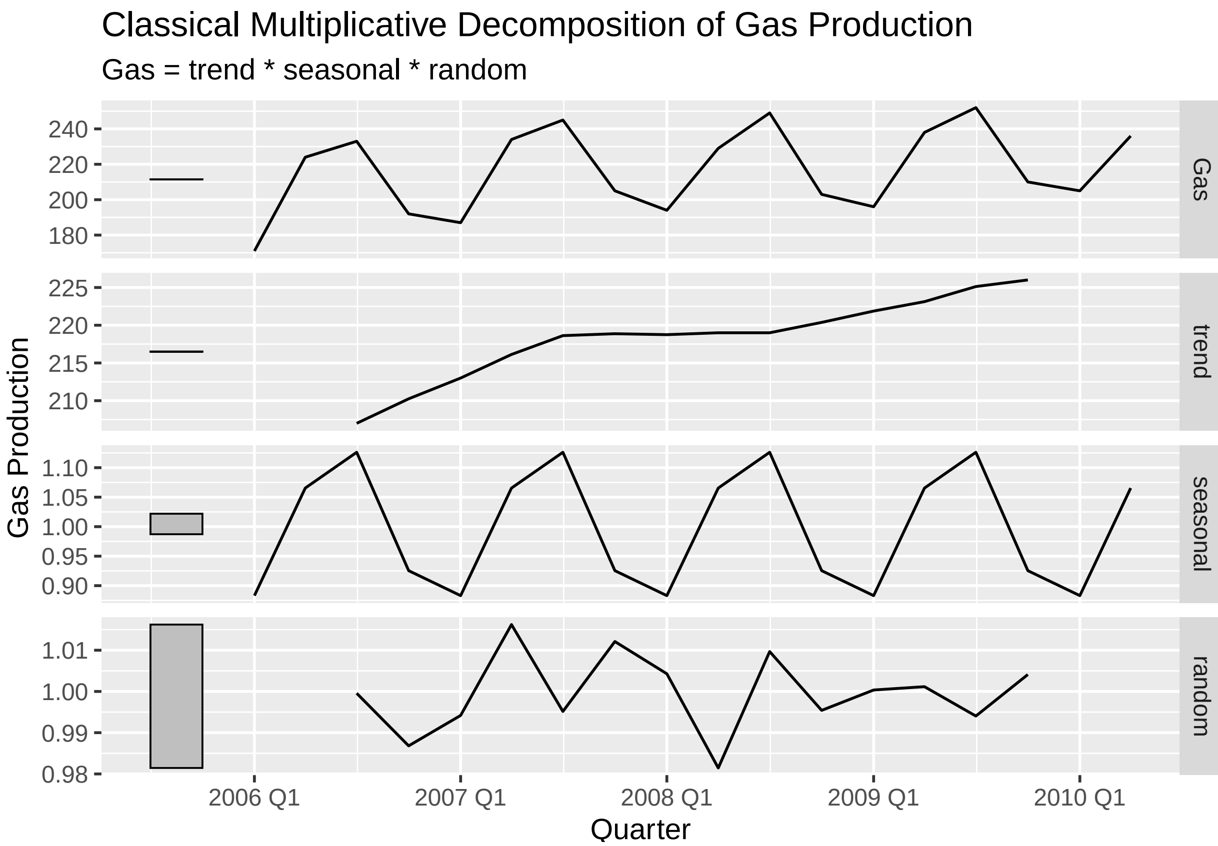
```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



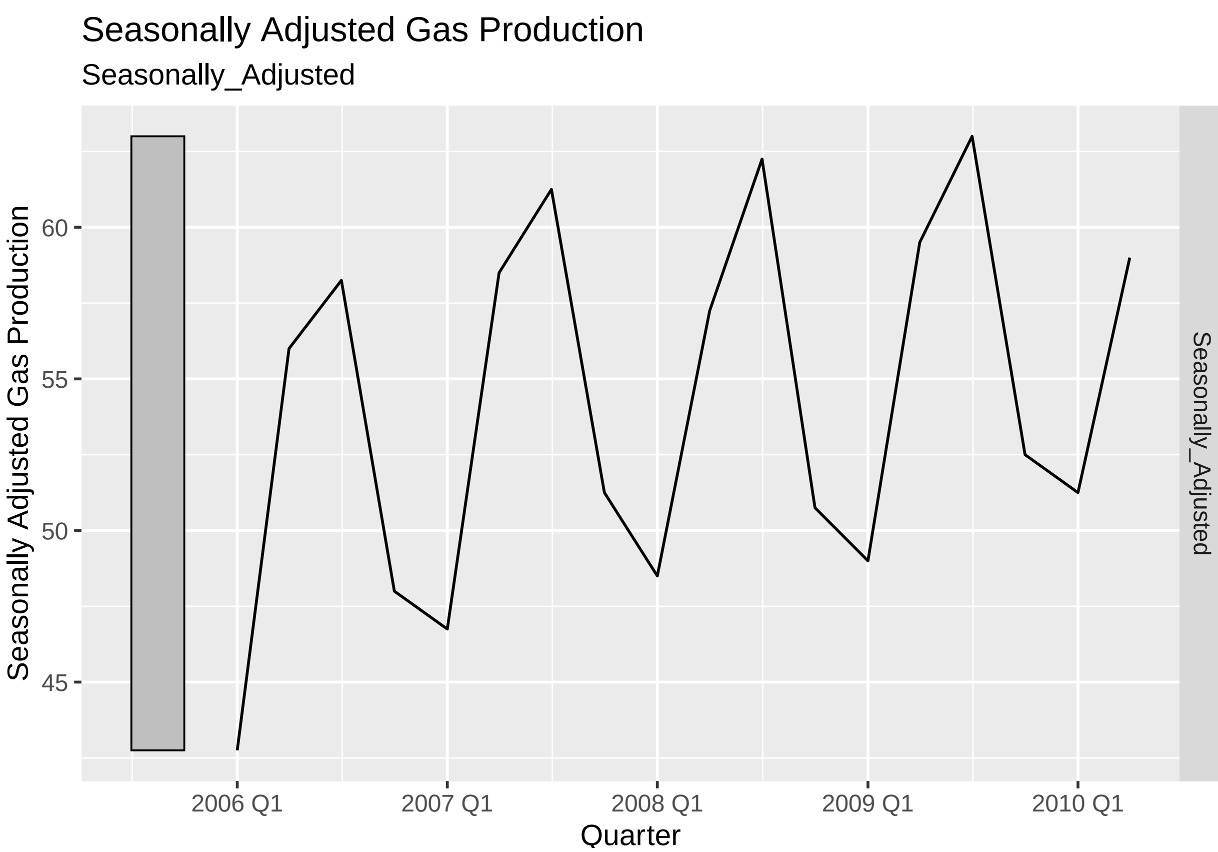
Computing and plotting seasonally adjusted data

```r
# Step 1: Classical decomposition with multiplicative type
decomposition <- gas |>
  model(classical_decomposition(Gas, type = "multiplicative")) |>
  components()

# Step 2: Plot the components to visualize the trend, seasonal, and remainder
autoplot(decomposition) +
  labs(title = "Classical Multiplicative Decomposition of Gas Production",
       x = "Quarter", y = "Gas Production")
```

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

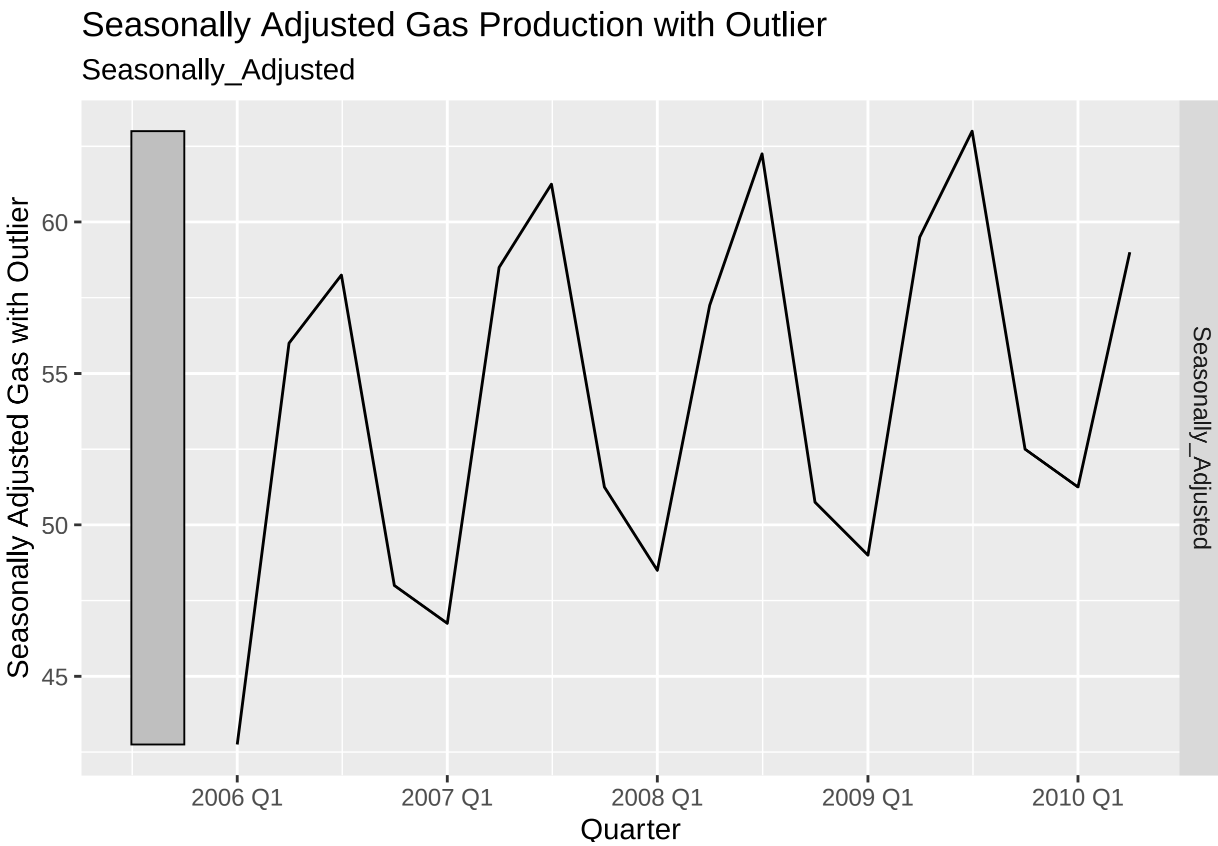



```r
# Step 3: Calculate seasonally adjusted data (Seasonally Adjusted = Gas / Seasonal)
seasonally_adjusted <- decomposition |>
  mutate(Seasonally_Adjusted = Gas /4)

# Step 4: Plot the seasonally adjusted data
autoplot(seasonally_adjusted, Seasonally_Adjusted) +
  labs(title = "Seasonally Adjusted Gas Production",
       x = "Quarter", y = "Seasonally Adjusted Gas Production")
```



Changing one observation to be an outlier (e.g., add 300 to one observation), and recompute the seasonally adjusted data.

```
# Introduce an outlier by adding 300 to the Gas value in '2005 Q4'
gas_with_outlier <- gas |>
  mutate(Gas = if_else(Quarter == yearquarter("2005 Q4"), Gas + 300, Gas))

# Perform decomposition with the outlier
decomposition_with_outlier <- gas_with_outlier |>
  model(classical_decomposition(Gas, type = "multiplicative")) |>
  components()

# Compute the seasonally adjusted data with the outlier (Seasonally Adjusted = Gas / Seasonal)
seasonally_adjusted_with_outlier <- decomposition_with_outlier |>
  mutate(Seasonally_Adjusted = Gas / 4)

# Plot the seasonally adjusted data with the outlier
seasonally_adjusted_with_outlier |>
  autoplot(Seasonally_Adjusted) +
  labs(title = "Seasonally Adjusted Gas Production with Outlier",
       x = "Quarter",
       y = "Seasonally Adjusted Gas with Outlier")
```



The `components()` function extracts the seasonal, trend, and remainder components from the decomposition model. The **seasonally adjusted data** is computed by dividing the original `Gas` value by the seasonal component (m equals to 4 for quarterly data, 12 for monthly data and 7 for daily data with weekly patterns. The outlier's impact can be visually seen in the seasonally adjusted data by comparing the original and the modified plots.

Exercise 3.8:

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

```
aus_retail
```

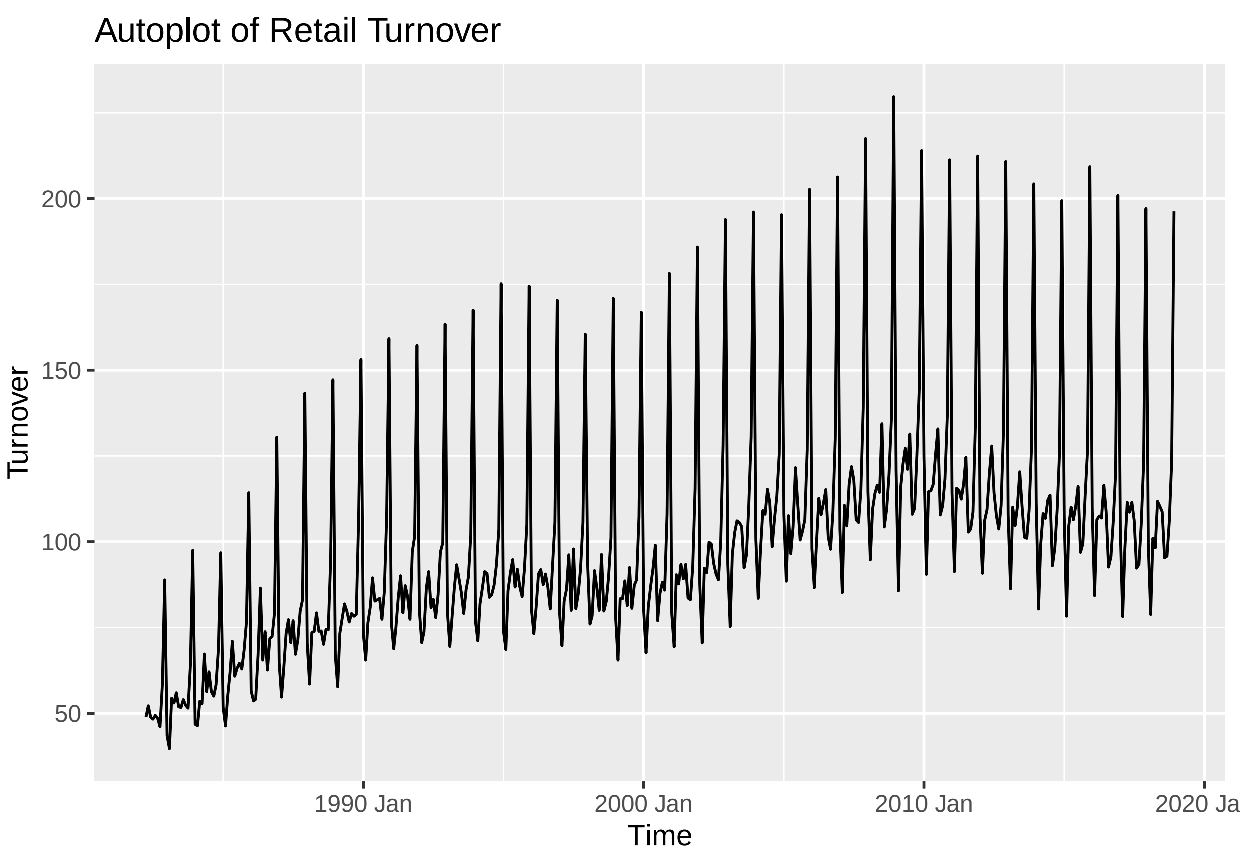
```
## # A tsibble: 64,532 x 5 [1M]
## # Key:       State, Industry [152]
##    State                        Industry           `Series ID`    Month Turnover
##    <chr>                        <chr>              <chr>          <mth>    <dbl>
##  1 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Apr      4.4
##  2 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 May      3.4
##  3 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Jun      3.6
##  4 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Jul      4
##  5 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Aug      3.6
##  6 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Sep      4.2
##  7 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Oct      4.8
##  8 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Nov      5.4
##  9 Australian Capital Territory Cafes, restaurant~ A3349849A   1982 Dec      6.9
## 10 Australian Capital Territory Cafes, restaurant~ A3349849A   1983 Jan      3.8
## # i 64,522 more rows
```

```r
glimpse(aus_retail)
```

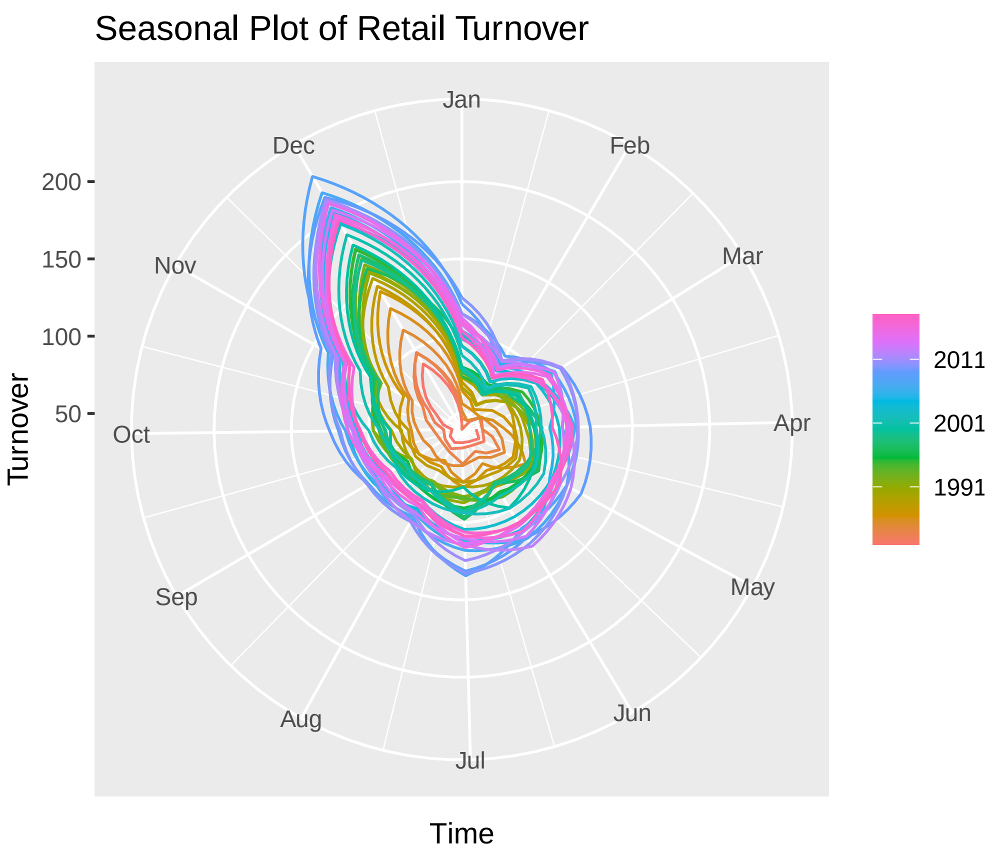
```
## Rows: 64,532
## Columns: 5
## Key: State, Industry [152]
## $ State       <chr> "Australian Capital Territory", "Australian Capital Territ~
## $ Industry    <chr> "Cafes, restaurants and catering services", "Cafes, restau~
## $ `Series ID` <chr> "A3349849A", "A3349849A", "A3349849A", "A3349849A", "A3349~
## $ Month       <mth> 1982 Apr, 1982 May, 1982 Jun, 1982 Jul, 1982 Aug, 1982 Sep~
## $ Turnover    <dbl> 4.4, 3.4, 3.6, 4.0, 3.6, 4.2, 4.8, 5.4, 6.9, 3.8, 4.2, 4.0~
```

```r
set.seed(13101917)
myseries <- aus_retail |>
  filter(`Series ID` == sample(aus_retail$`Series ID`,1))
```

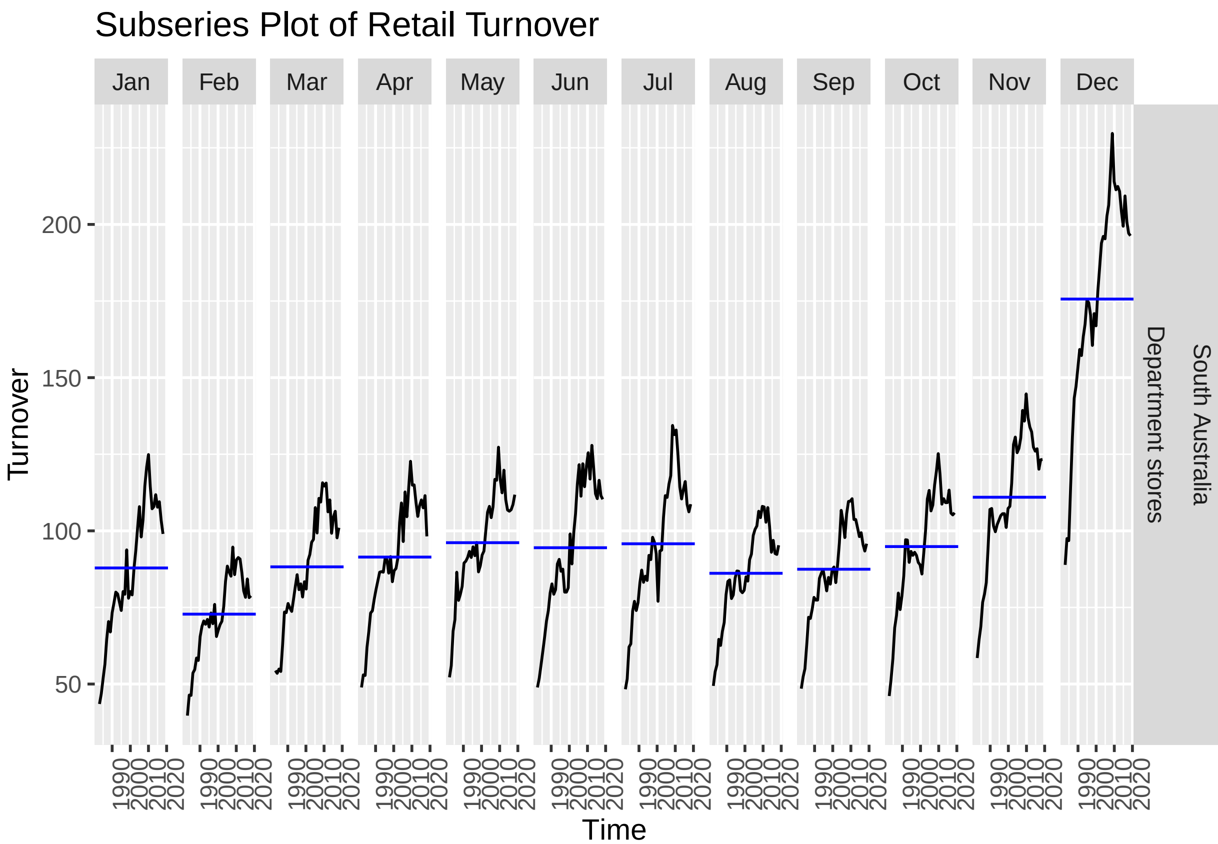
```r
library(ggplot2)
library(fpp3)  # For aus_retail dataset

set.seed(13101917)
myseries <- aus_retail |>
  filter(`Series ID` == sample(aus_retail$`Series ID`, 1))

# Autoplot
autoplot(myseries, Turnover) +
  ggtitle("Autoplot of Retail Turnover") +
  ylab("Turnover") +
  xlab("Time")
```



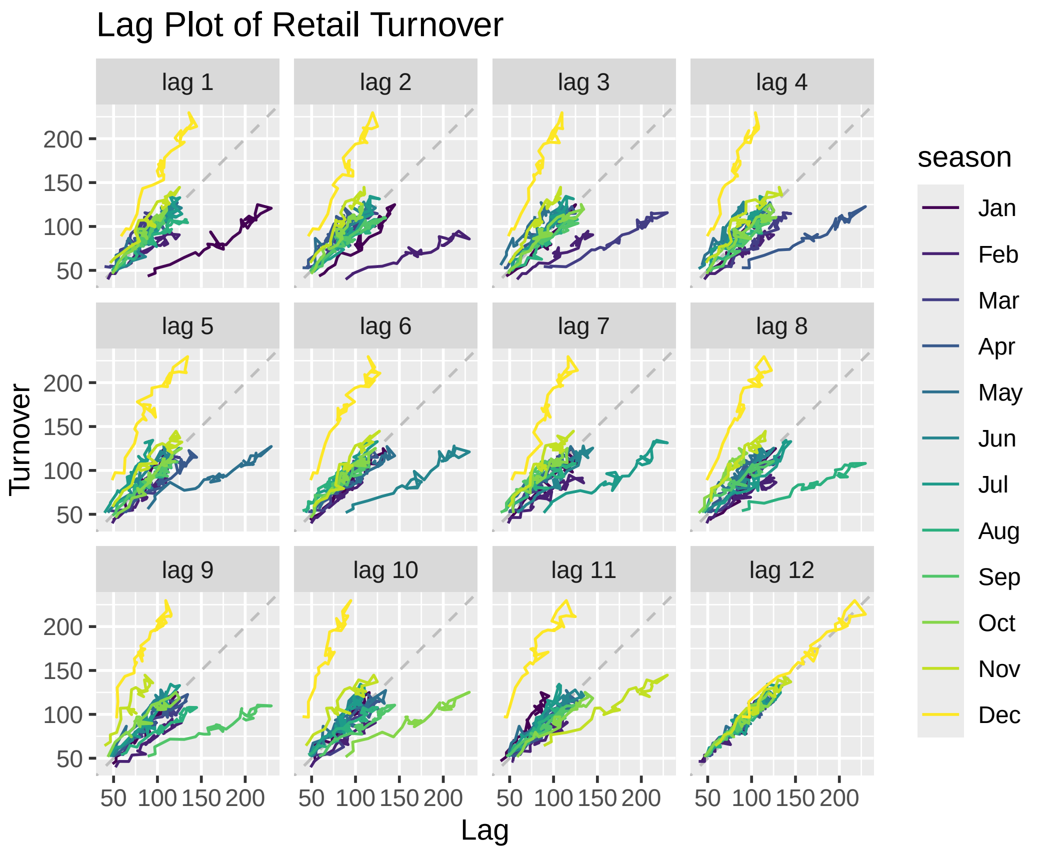
```r
# Seasonal plot
gg_season(myseries, Turnover, polar = TRUE) +
  ggtitle("Seasonal Plot of Retail Turnover") +
  ylab("Turnover") +
  xlab("Time")
```




```r
# Subseries plot
gg_subseries(myseries, Turnover) +
  ggtitle("Subseries Plot of Retail Turnover") +
  ylab("Turnover") +
  xlab("Time")
```



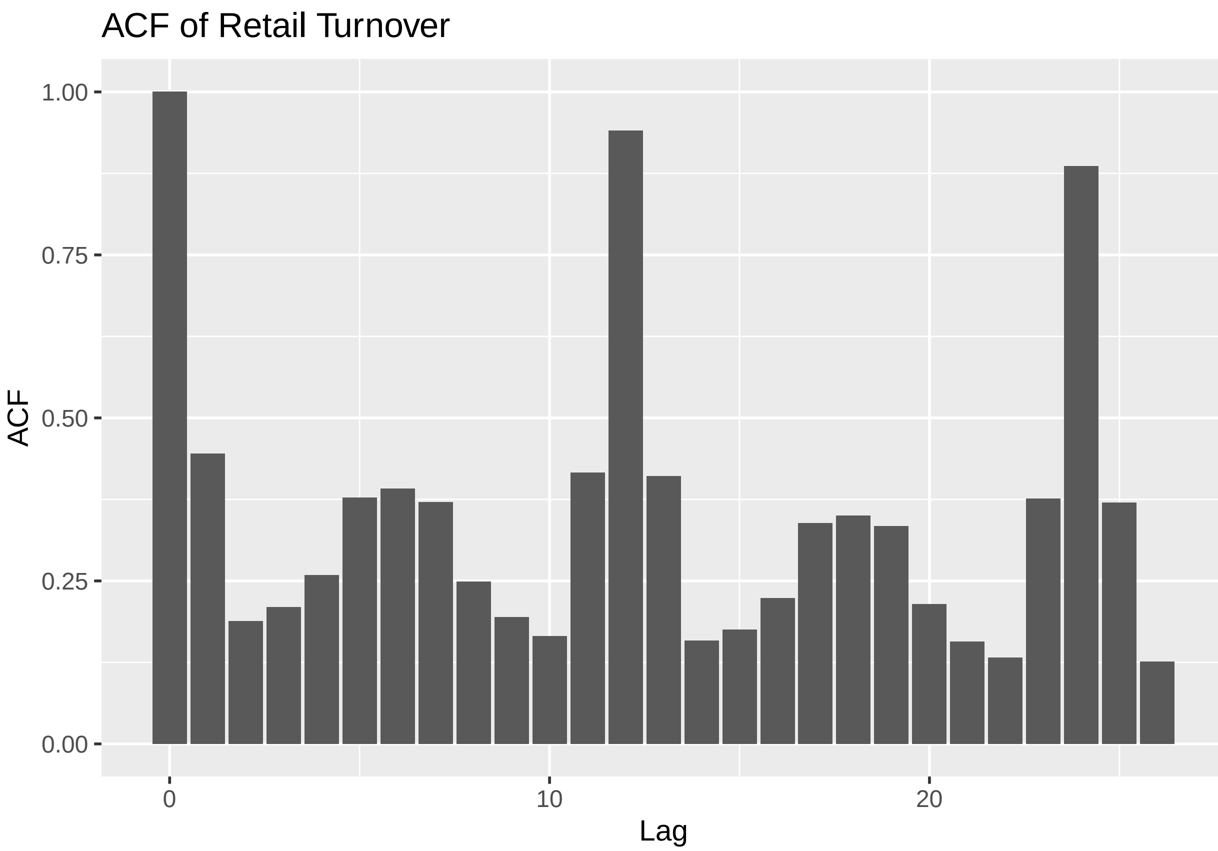

```r
# Lag plot
gg_lag(myseries, Turnover,lags = 1:12) +
  ggtitle("Lag Plot of Retail Turnover") +
  ylab("Turnover") +
  xlab("Lag")
```



Autocorrelation Function(ACF) plot

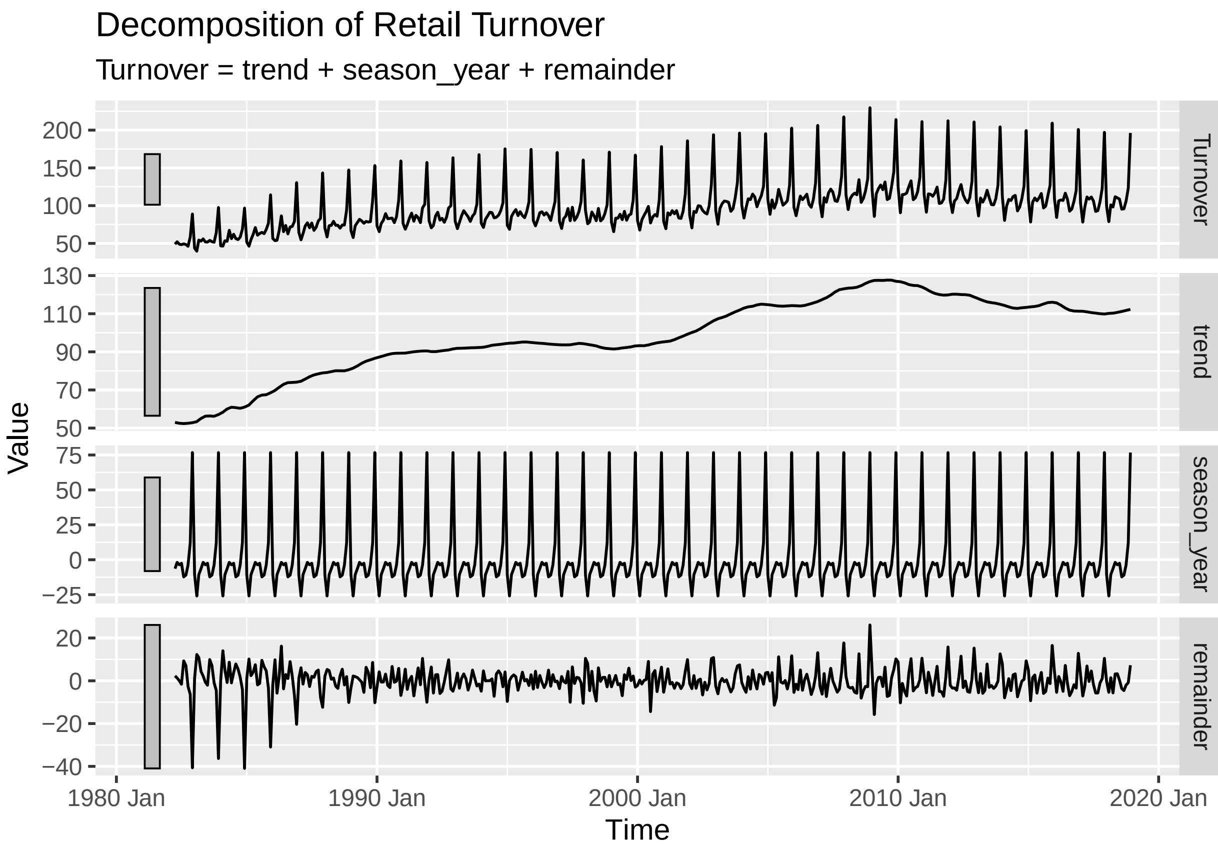
```
# Autocorrelation Function plot
library(ggplot2)
library(fpp3)

# Convert to tsibble if needed
myseries_ts <- myseries |>
  as_tsibble(index = Month)

# Convert Turnover to a numeric vector
turnover_vector <- myseries_ts$Turnover

# Calculate ACF using base R
acf_values <- acf(turnover_vector, plot = FALSE)

# Convert ACF values to a data frame for plotting
acf_df <- data.frame(
  Lag = acf_values$lag,
  ACF = acf_values$acf
)

# Plot ACF
ggplot(acf_df, aes(x = Lag, y = ACF)) +
  geom_bar(stat = "identity") +
  ggtitle("ACF of Retail Turnover") +
```

```
  ylab("ACF") +
  xlab("Lag")
```



Decompose the Series using X-11

```
# Decomposition
decomposition <- myseries |>
  model(STL(Turnover ~ season(window = "periodic"))) |>
  components()

# Plot decomposition
autoplot(decomposition) +
  ggtitle("Decomposition of Retail Turnover") +
  ylab("Value") +
  xlab("Time")
```



Exercise 3.9:Figures 3.19 and 3.20 show the result of decomposing the number of persons in the civilian labour force in Australia each month from February 1978 to August 1995.

Understanding a decomposition graph:

The above decomposition plot shows the breakdown of a time series into its trend, seasonal, and residual components.

The Trend Component, aka Overall Trend of this graph highlights an upward trend over time. This indicates a general increase in employment or labor force participation, which could be attributed to economic growth or demographic changes.

The Seasonal Component, aka Seasonal Patterns reflects predictable variations in labor force data due to annual cycles, such as holiday seasons or fiscal year-end activities affecting employment, like seasonal hiring. This seasonal plot displays regular, repeating patterns within each year, suggesting that employment figures vary predictably over the year.

Finally, the Residual Component, aka Residual Variability plot (or residual noise) captures the irregular, random fluctuations that remain after accounting for the trend and seasonality. These fluctuations are relatively small compared to the trend and seasonal components, indicating that most of the variation in the labor force data is explained by the trend and seasonal patterns.

Is the recession of 1991/1992 visible in the estimated components? Answer: YES

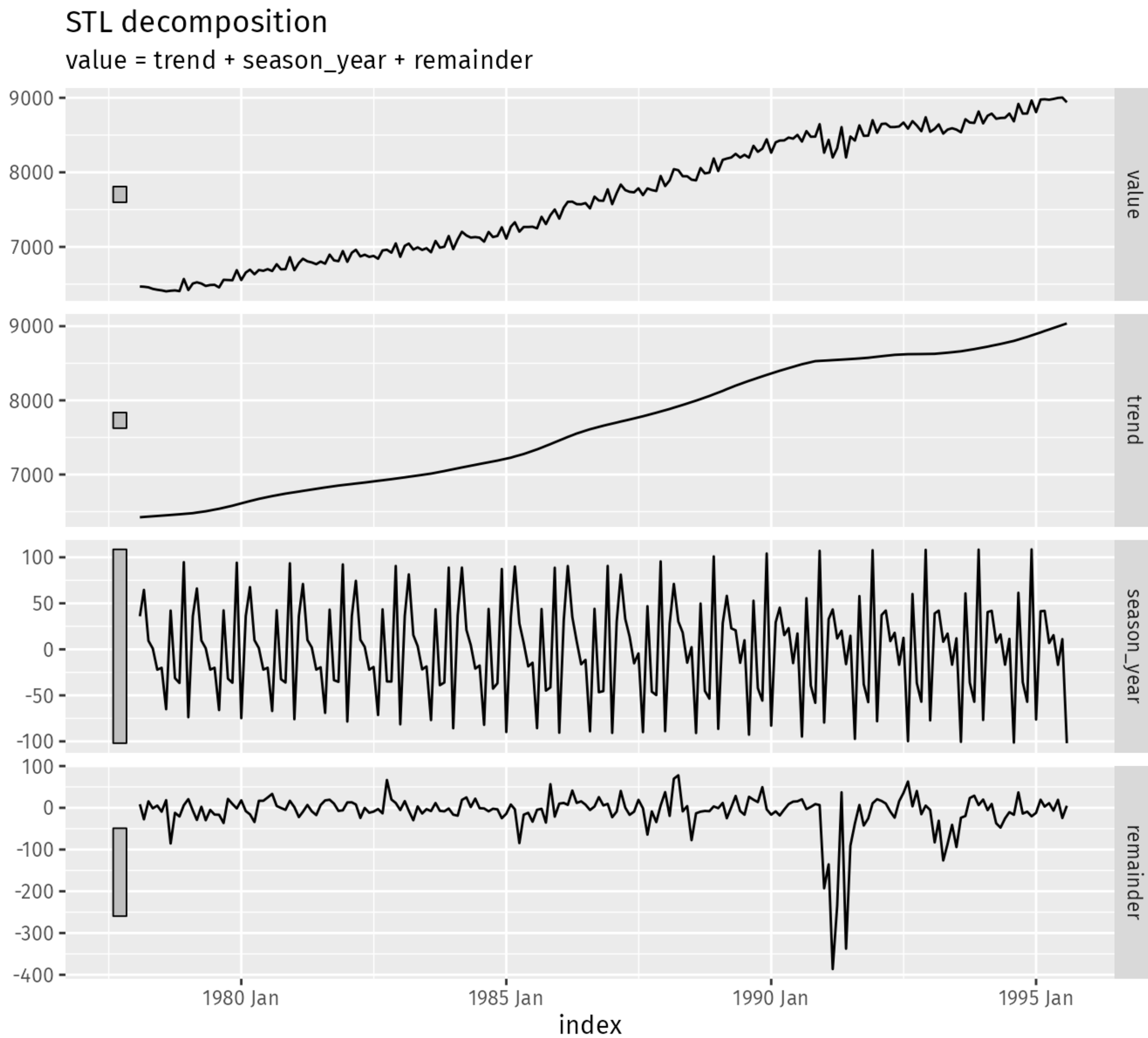


Figure 1: STL Decomposition