Data 624 Homework 1

Warner Alexis

2025-02-09

Problem 2.1

Explore the following four time series: `Bricks` from `aus_production`, `Lynx` from pelt, `Close$` from `gafa_stock`, `Demand` from `vic_elec`.

Use `?` (or `help()`) to find out about the data in each series. What is the time interval of each series? Use autoplot() to produce a time plot of each series. For the last plot, modify the axis labels and title.

These are the libraries that are important for this assignments. See code below:

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union

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

##
## Attaching package: 'tsibble'

## The following object is masked from 'package:lubridate':
##
##     interval

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, union

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats 1.0.0     v stringr 1.5.1
## v ggplot2 3.5.1     v tibble  3.2.1
## v purrr   1.0.2     v tidyr   1.3.1
## v readr   2.1.5
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter()     masks stats::filter()
## x tsibble::interval() masks lubridate::interval()
## x dplyr::lag()        masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
## -- Attaching packages -------------------------------------------- fpp3 1.0.1 --
##
## v tsibbledata 0.4.1     v fable       0.4.1
## v feasts      0.4.1
##
## -- 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()
##
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

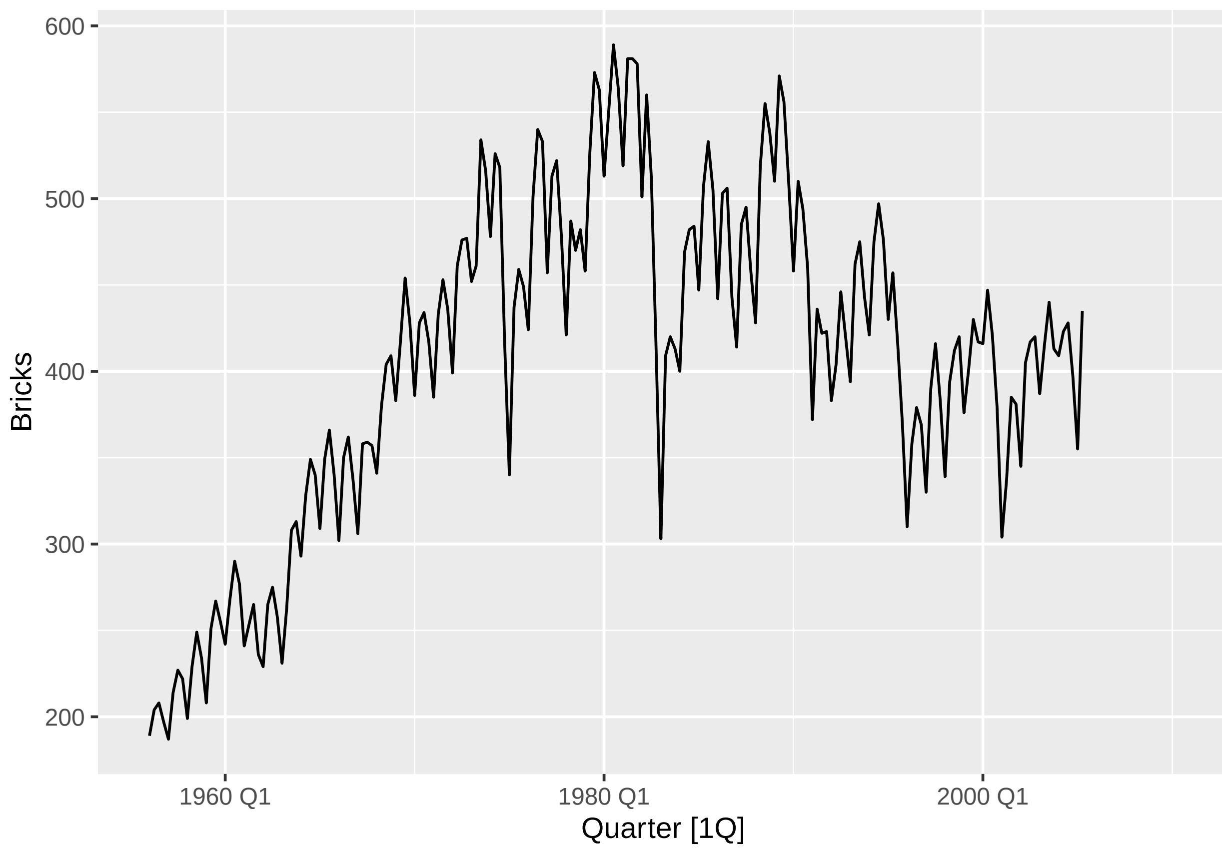
```r
data("aus_production")
?aus_production
```

```
## starting httpd help server ... done
```

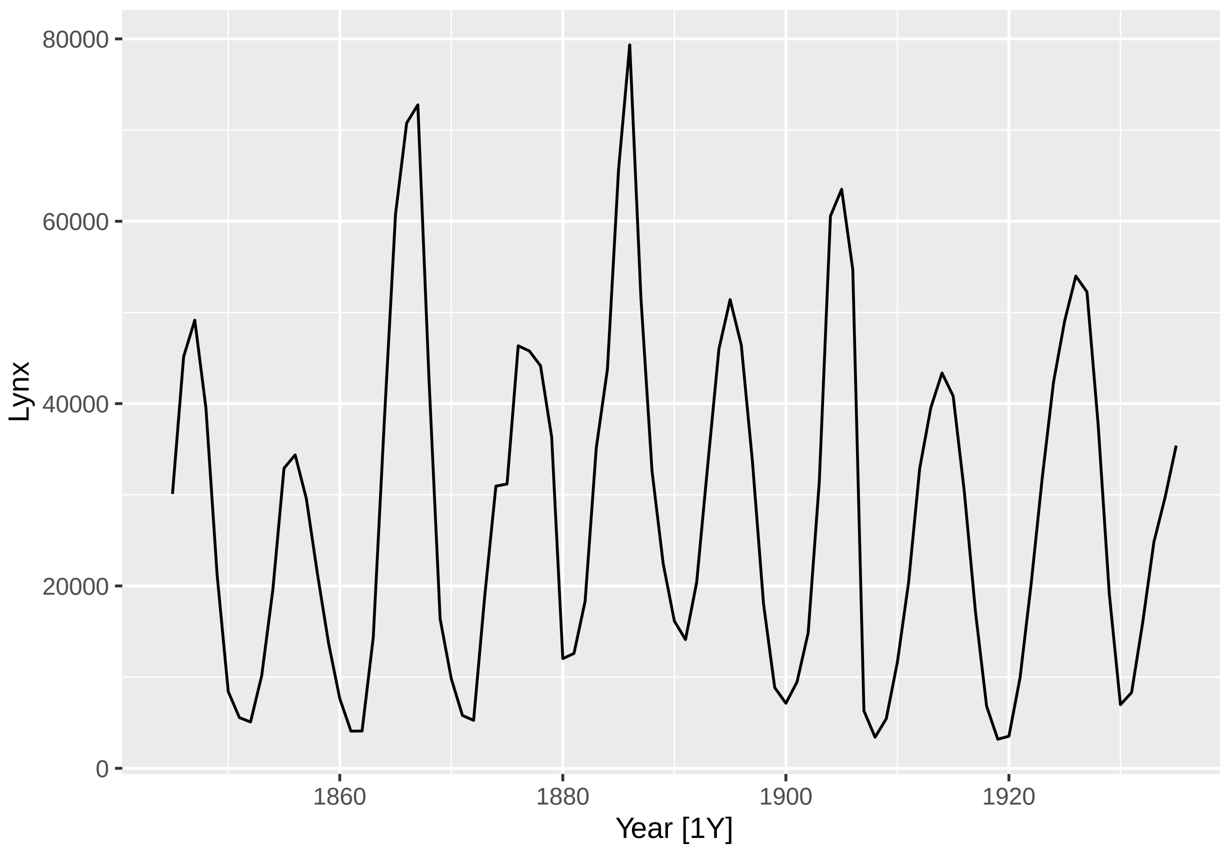
```r
data("pelt")
?pelt

data("gafa_stock")
?gafa_stock

data("vic_elec")
?vic_elec
```

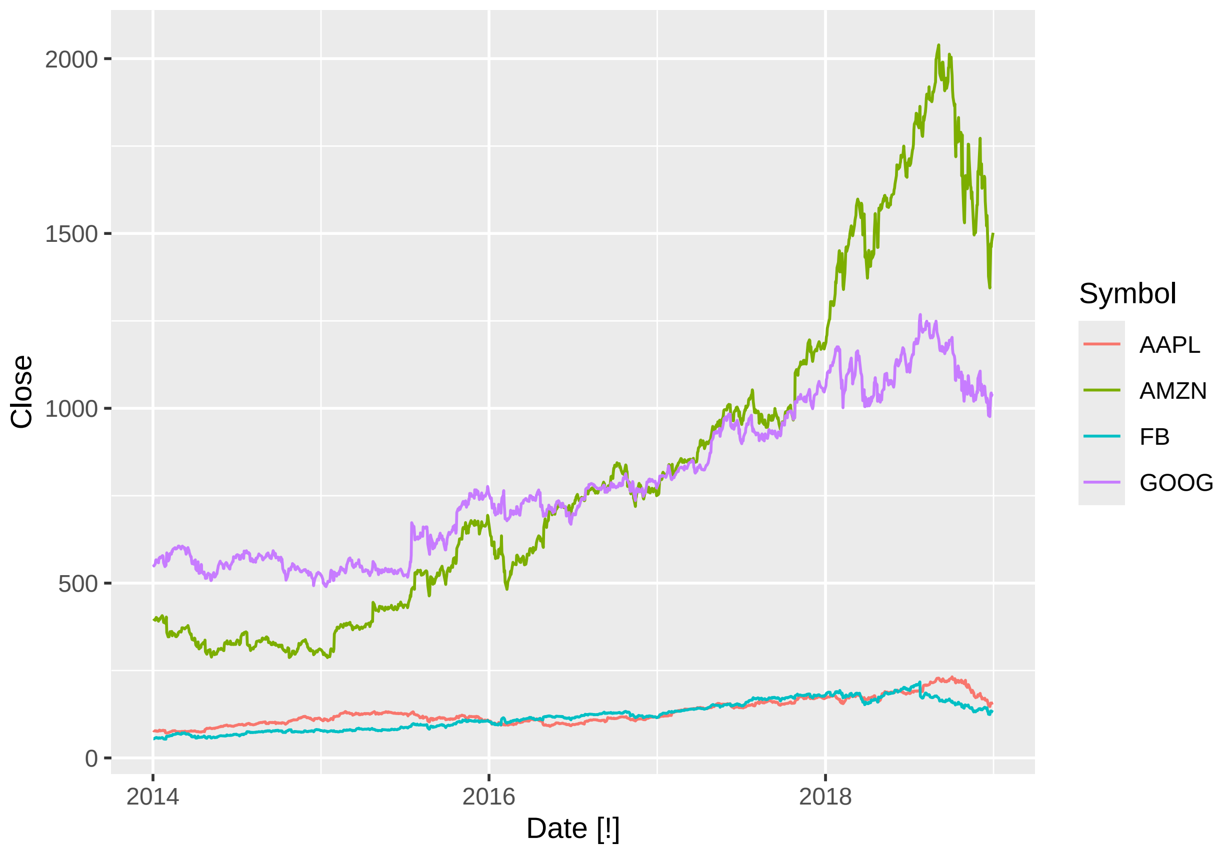
Producing the autoplot for all four dataset and modify the axis labels and title

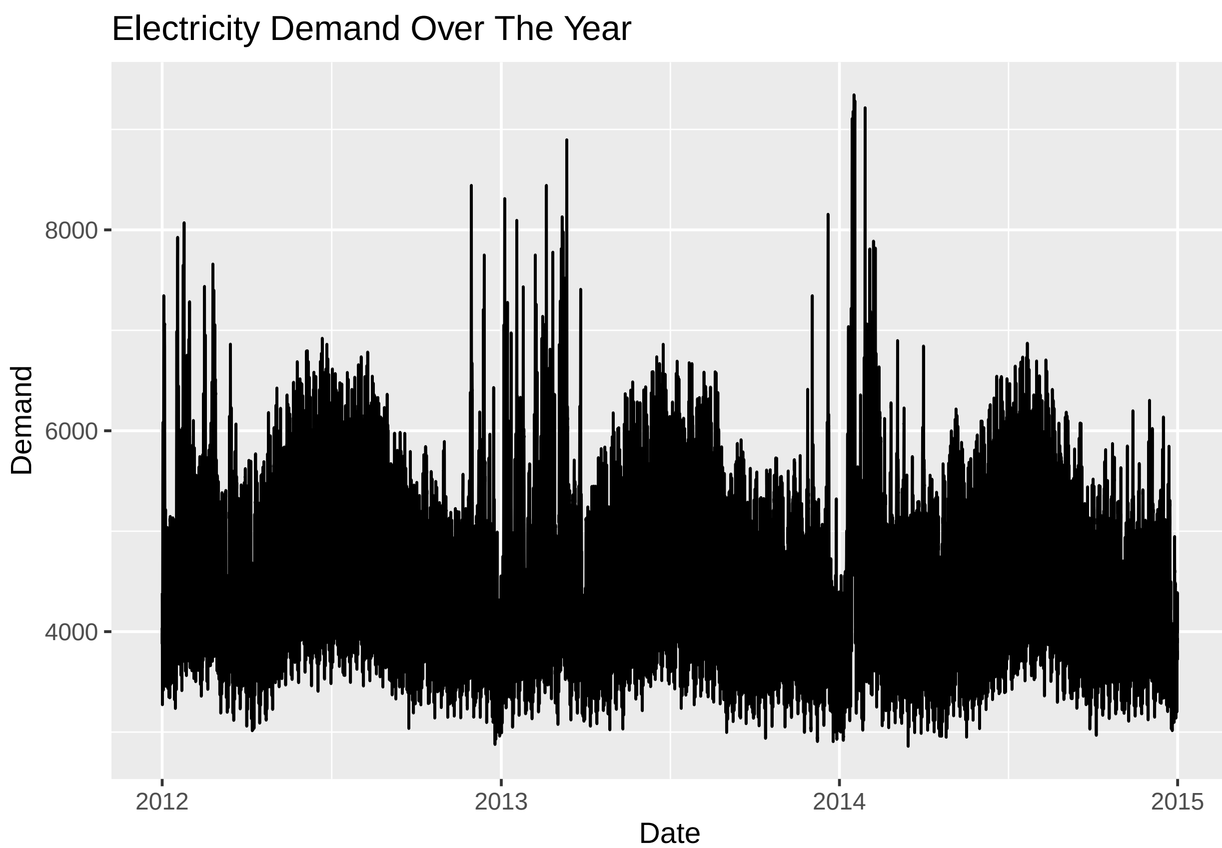
```
## Warning: Removed 20 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



600
500
400
300
200
Bricks
1960 Q1
1980 Q1
2000 Q1
Quarter [1Q]


80000
60000
40000
20000
0
Lynx
1860
1880
1900
1920
Year [1Y]


2000
1500
1000
500
0
Close
2014
2016
2018
Date [!]
Symbol
AAPL
AMZN
FB
GOOG



Problem 2.2

Use $filter() to find what days corresponded to the peak closing price for each of the four stocks in *gafa_stoc*.

Amazon had the highest stock closing price on September 8, 2018, at 2,039. In comparison, other companies recorded lower closing prices on the same day, with Facebook at 217.50, Apple at 232.07, and Google at $1,268.33.

```
gafa_stock %>% group_by(Symbol) %>%
  filter(Close==max(Close)) %>%
  select(Symbol,
         Date,
         Close)
```

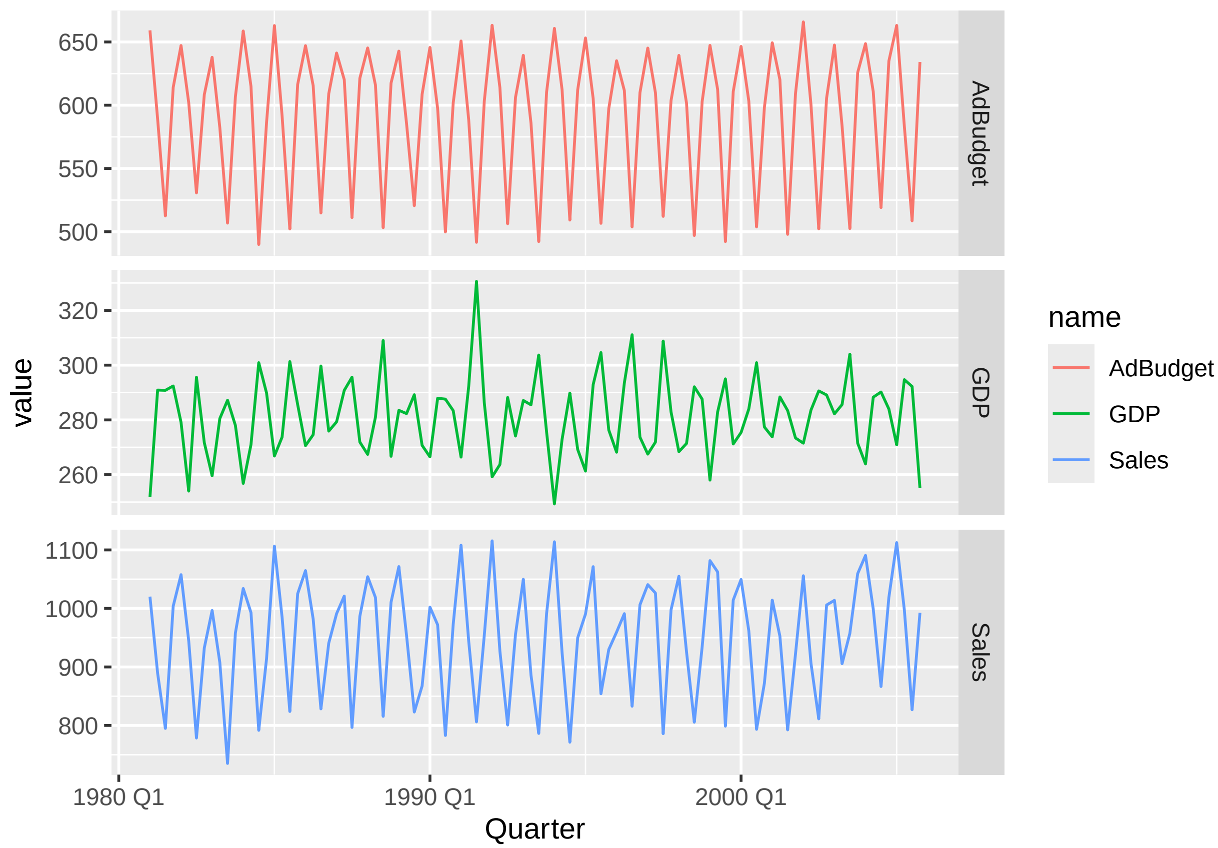
```
## # A tsibble: 4 x 3 [!]
## # Key:       Symbol [4]
## # Groups:    Symbol [4]
##   Symbol Date       Close
##   <chr>  <date>     <dbl>
## 1 AAPL   2018-10-03  232.
## 2 AMZN   2018-09-04 2040.
## 3 FB     2018-07-25  218.
## 4 GOOG   2018-07-26 1268.
```

Problem 2.3

Download the file tute1.csv from the book website, open it in Excel (or some other spreadsheet application), and review its contents. You should find four columns of information. Columns B through D each contain a quarterly series, labelled Sales, AdBudget and GDP. Sales contains the quarterly sales for a small company over the period 1981-2005. AdBudget is the advertising budget and GDP is the gross domestic product. All series have been adjusted for inflation.

When `facet_grid` is not included, all the plots appear together on a single graph. However, when it is added, three separate graphs are generated for each category: AdBudget, GDP, and Sales.

```
# read the data set from Github
tute1 <- read.csv('https://raw.githubusercontent.com/joewarner89/Data-624-Predictive-Anaytics/refs/heads

# convert the data set into tsibble
mytimeseries <- tute1 |>
  mutate(Quarter = yearquarter(Quarter)) |>
  as_tsibble(index = Quarter)

mytimeseries |>
  pivot_longer(-Quarter) |>
  ggplot(aes(x = Quarter, y = value, colour = name)) +
  geom_line() +
  facet_grid(name ~ ., scales = "free_y")
```



Problem 2.4

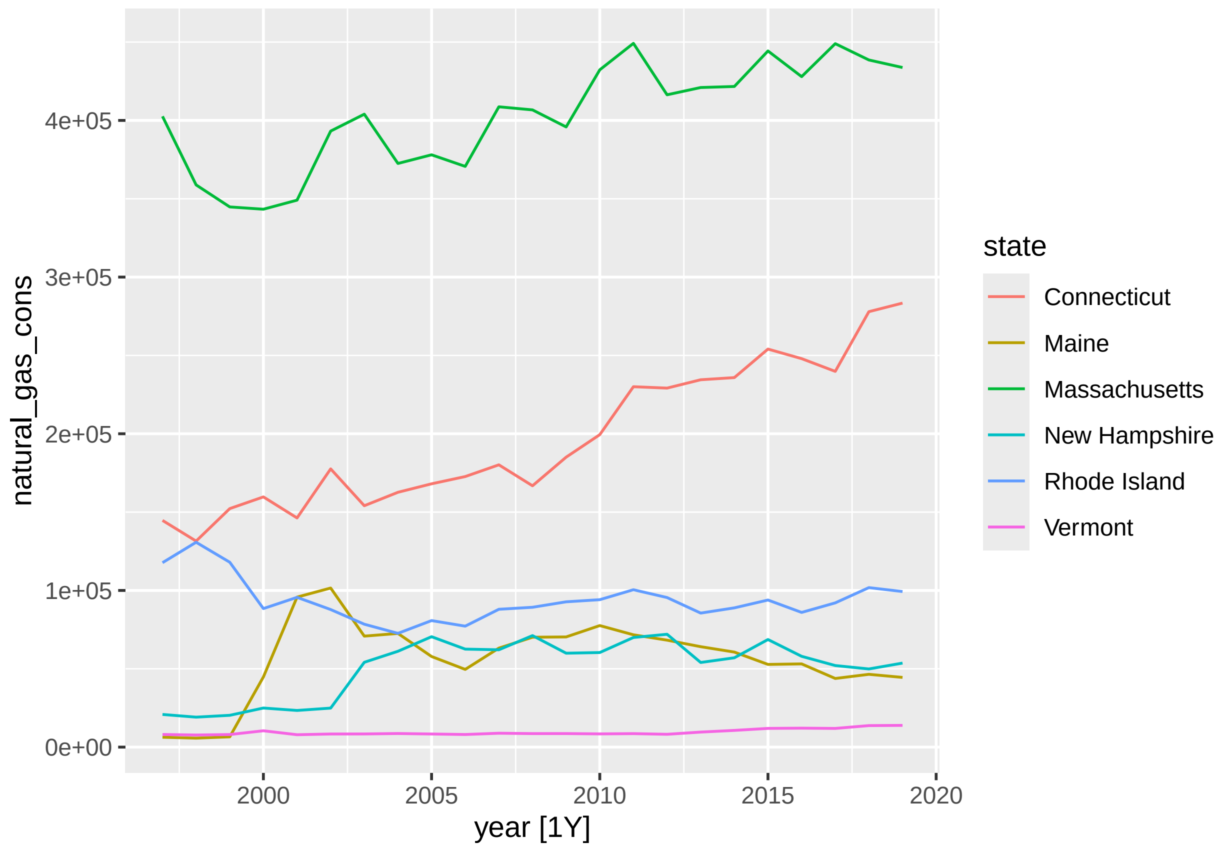
The `USgas` package contains data on the demand for natural gas in the US.

Install the USgas package. Create a tsibble from us_total with year as the index and state as the key. Plot the annual natural gas consumption by state for the New England area (comprising the states of Maine, Vermont, New Hampshire, Massachusetts, Connecticut and Rhode Island).

```
# package has been installed
# install.packages('USgas')
library(USgas)
data('us_total')
str(us_total)
```

```
## 'data.frame':    1266 obs. of  3 variables:
##  $ year : int  1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 ...
##  $ state: chr  "Alabama" "Alabama" "Alabama" "Alabama" ...
##  $ y    : int  324158 329134 337270 353614 332693 379343 350345 382367 353156 391093 ...
```

```
# filter the data and create the tsibble
us_gas <- us_total |>
  rename(natural_gas_cons = y)
us_gas_tsibble <- us_gas |> filter(state %in% c("Maine", "Vermont", "New Hampshire", "Massachusetts", "(


# create the autoplot
us_gas_tsibble |> autoplot(natural_gas_cons)
```



Problem 2.5

Download `tourism.xlsx` from the book website and read it into R using `readxl::read_excel()`. Create a tsibble which is identical to the `tourism` tsibble from the tsibble package. Find what combination of `Region` and `Purpose` had the maximum number of overnight trips on average. Create a new tsibble which combines the Purposes and Regions, and just has total trips by State

```
library(readxl)
library(httr)
library(openxlsx)
url <- 'https://raw.githubusercontent.com/joewarner89/Data-624-Predictive-Anaytics/main/workspace/touris

temp_file <- tempfile(fileext = ".xlsx")  # Create a temporary file

download.file(url, temp_file, mode = "wb")  # Download the file
tourism <- read_excel(temp_file)  # Read the Excel file

head(tourism)
```

```
## # A tibble: 6 x 5
##   Quarter    Region   State           Purpose  Trips
##   <chr>      <chr>    <chr>           <chr>    <dbl>
## 1 1998-01-01 Adelaide South Australia Business  135.
## 2 1998-04-01 Adelaide South Australia Business  110.
## 3 1998-07-01 Adelaide South Australia Business  166.
## 4 1998-10-01 Adelaide South Australia Business  127.
## 5 1999-01-01 Adelaide South Australia Business  137.
## 6 1999-04-01 Adelaide South Australia Business  200.
```

```
# Convert data to  tsibble
data <- tourism |> mutate(Quarter = as.Date(Quarter),
                          Trips = as.numeric(Trips)) |> as_tibble(key = c(Region, State,Purpose), index

max_avg_trips <- data |> group_by(Region, Purpose) |> summarize(avg_trips = mean(Trips)) |> arrange(desc
```

```
## `summarise()` has grouped output by 'Region'. You can override using the
## `.groups` argument.
```

```
head(max_avg_trips)
```

```
## # A tibble: 6 x 3
## # Groups:   Region [4]
##   Region          Purpose  avg_trips
##   <chr>           <chr>        <dbl>
## 1 Sydney          Visiting      747.
## 2 Melbourne       Visiting      619.
## 3 Sydney          Business      602.
## 4 North Coast NSW Holiday       588.
## 5 Sydney          Holiday       550.
## 6 Gold Coast      Holiday       528.
```

This table presents data on average trips (`avg_trips`) taken for different purposes (`Purpose`) across various regions (`Region`). It includes information about three types of trips: **Visiting, Business, and Holiday**, with Sydney appearing multiple times across different categories. Sydney has the highest average trips for visiting (747.27), followed by Melbourne (618.90). Business trips in Sydney average 602.04, while holiday trips are more evenly distributed among North Coast NSW (587.90), Sydney (550.33), and Gold Coast (528.34).

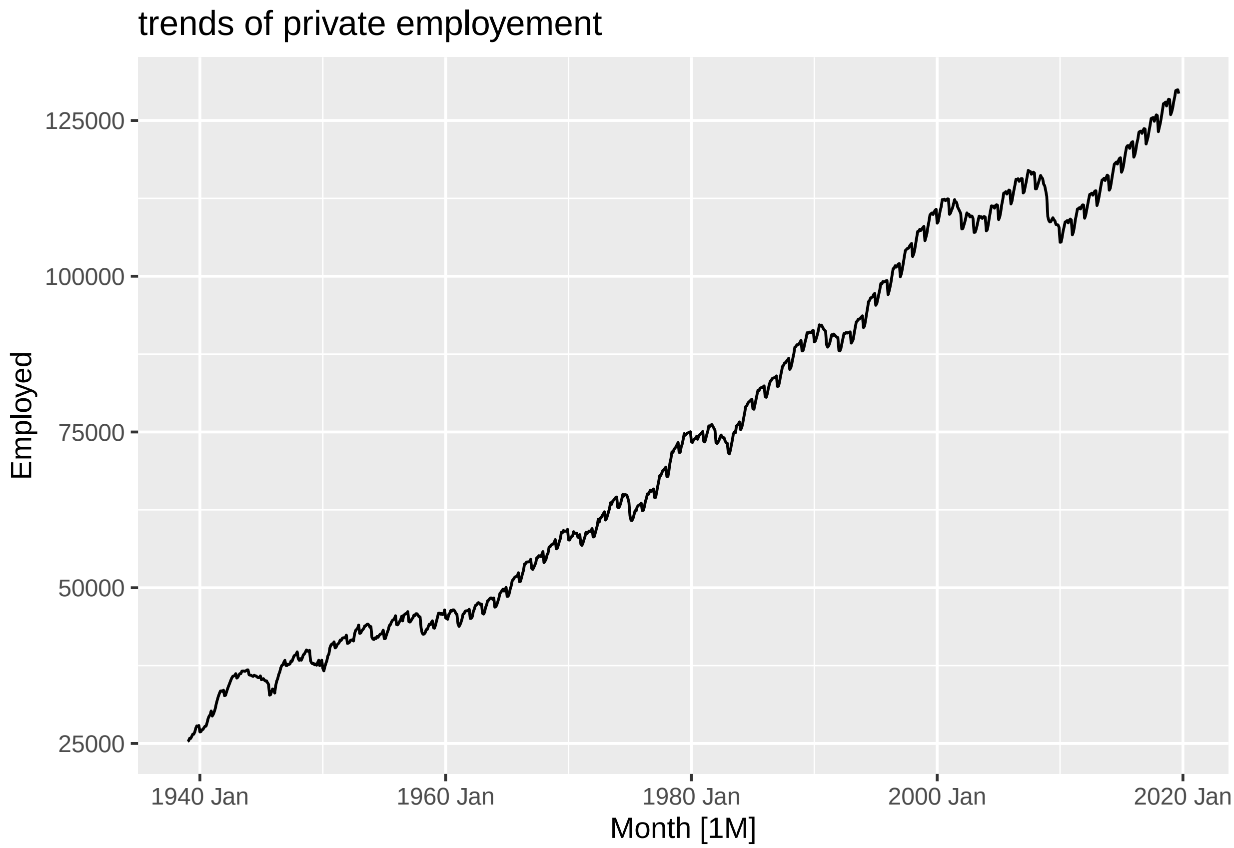
```
total_trips_state <- data |>group_by(State) |> summarize(total_trips = sum(Trips)) |>
  arrange(desc(total_trips))
head(total_trips_state)
```

```
## # A tibble: 6 x 2
##   State              total_trips
##   <chr>                    <dbl>
## 1 New South Wales        557367.
## 2 Victoria               390463.
## 3 Queensland             386643.
## 4 Western Australia      147820.
## 5 South Australia        118151.
## 6 Tasmania                54137.
```

This table summarizes the total number of trips (`total_trips`) taken in different Australian states (`State`). **New South Wales** has the highest total trips at **557,367.43**, followed by **Victoria (390,462.91)** and **Queensland (386,642.91)**. The numbers drop significantly for **Western Australia (147,819.65)**, **South Australia (118,151.35)**, and **Tasmania (54,137.09)**, indicating that travel activity is more concentrated in the eastern states.

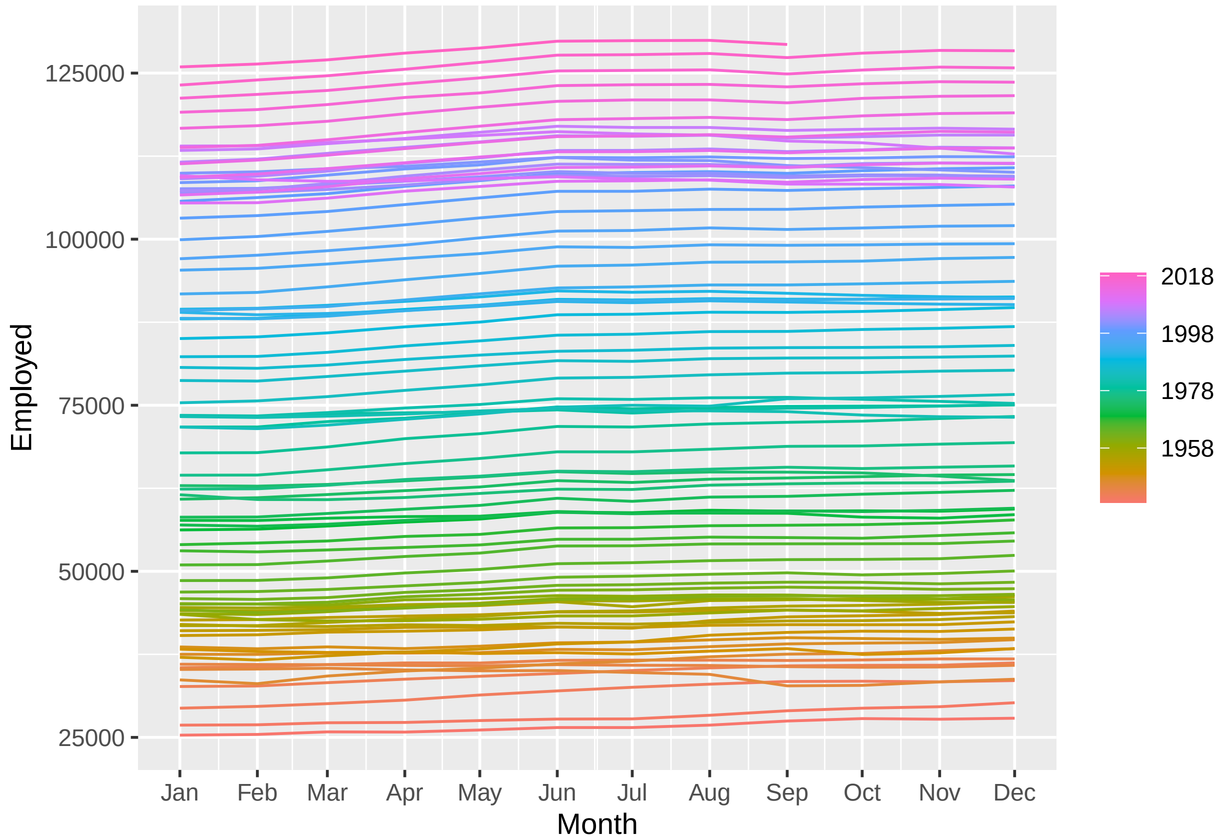
Problem 2.8

```
data("PBS")
data("us_employment")
data("us_gasoline")


# #mployement data set autioplot
us_employment |> filter(Title == 'Total Private') |> autoplot(Employed) + labs(title = 'trends of priva
```

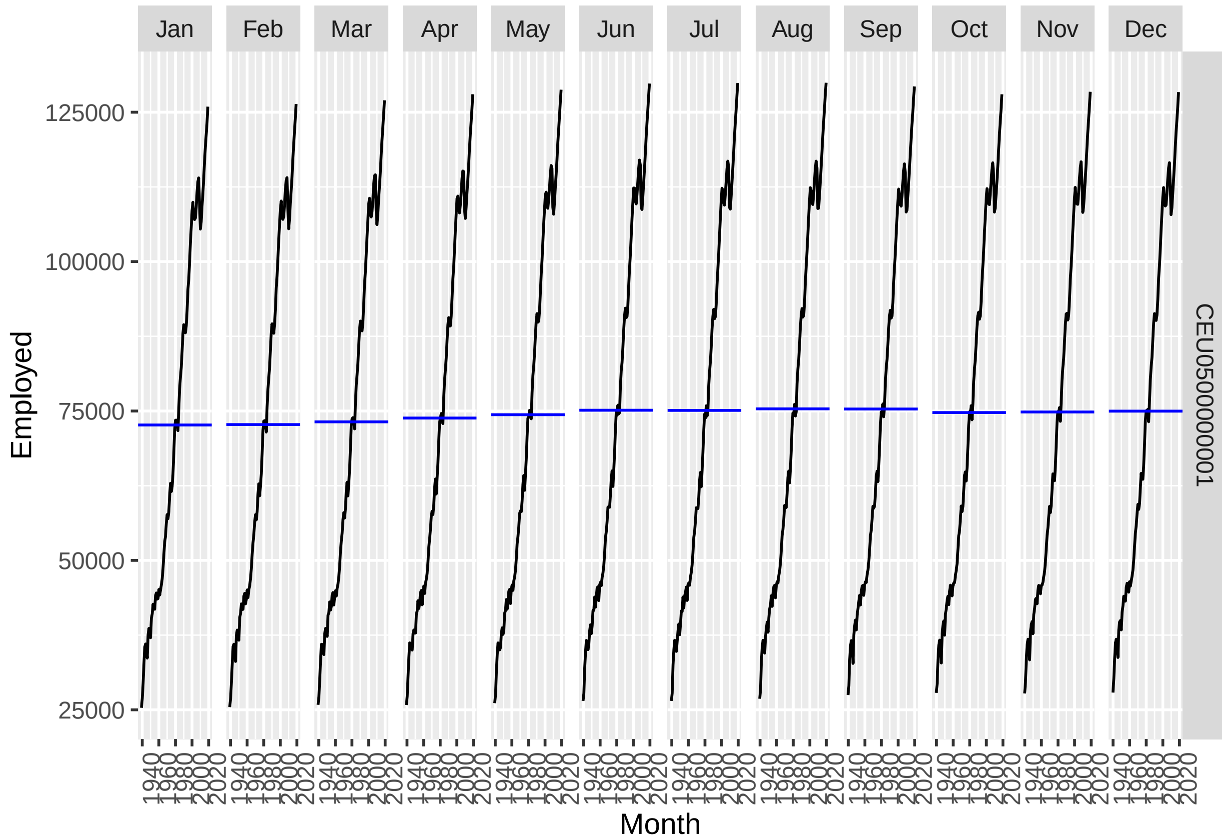


US Employment The graph displays the trend of private employment over time, spanning from approximately 1940 to 2020. The x-axis represents time in months, while the y-axis shows the number of employed individuals in the private sector.

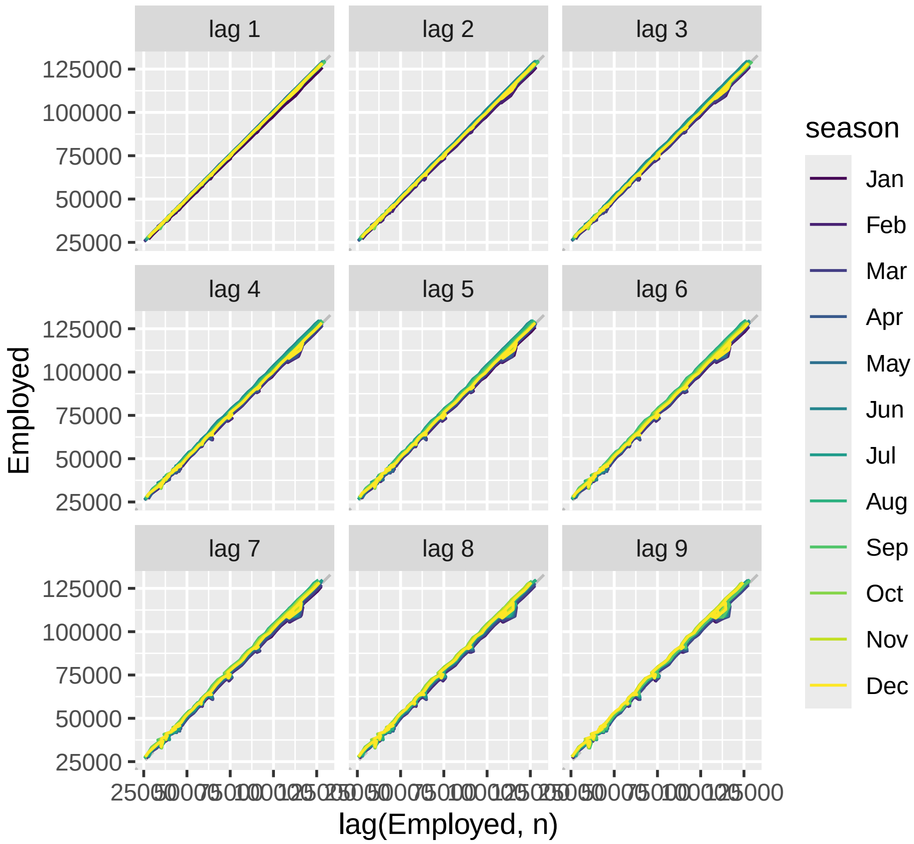
```
us_employment %>%filter(Title == "Total Private") %>% gg_season(Employed) +
  labs(tittle = "Seasonal Decomposition")
```




```
us_employment %>%
  filter(Title == "Total Private") %>%
  gg_subseries(Employed) +
  labs("Subseries Plot")
```




```
us_employment %>%
  filter(Title == "Total Private") %>%
  gg_lag(Employed) +
  labs("Lag Plot")
```



These visualizations analyze **private employment trends** over time.

1. **Subseries Plot** – Displays employment trends by month, showing a consistent seasonal pattern across years.
2. **Lag Plot** – Highlights strong autocorrelation in employment data, indicating that past values are highly predictive of future trends.
3. **Seasonal Decomposition** – Illustrates long-term employment growth while capturing seasonal variations. Higher employment levels in recent years (2018) are evident compared to earlier decades (1958).

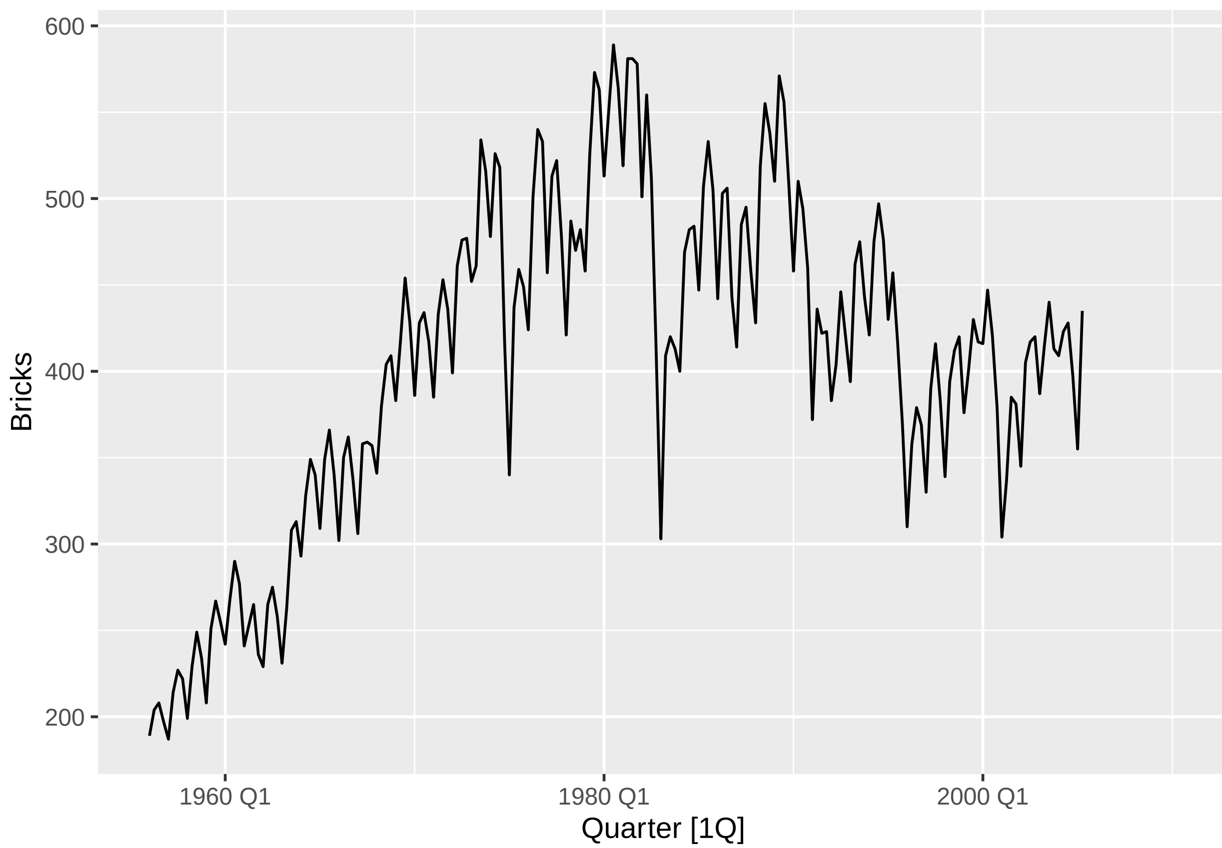
BRICKS

These visualizations analyze the **time series of brick production** over time.

1. **Autoplot** – Shows the overall trend in brick production, with an increase until around 1980, followed by a decline and fluctuations.
2. **Subseries Plot** – Displays seasonal patterns across quarters (Q1–Q4), highlighting variations in production levels.
3. **Lag Plot** – Indicates a strong correlation between past and present values, suggesting high persistence in trends.
4. **Autocorrelation Function (ACF) Plot** – Confirms significant autocorrelation, meaning past production levels strongly influence future values.

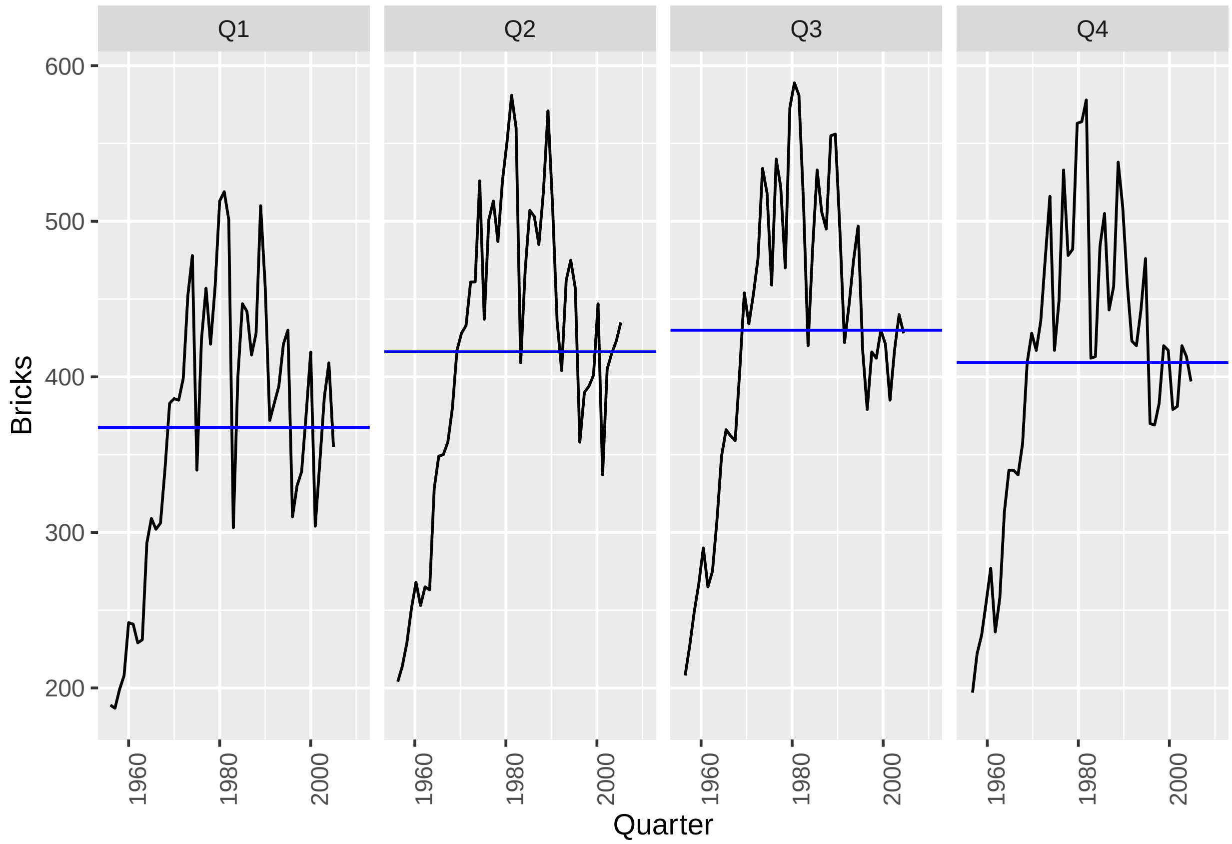
```
aus_production |> autoplot(Bricks) + labs("Obsvation of Bricks overtime")
```

```
## Warning: Removed 20 rows containing missing values or values outside the scale range
## (‘geom_line()‘).
```



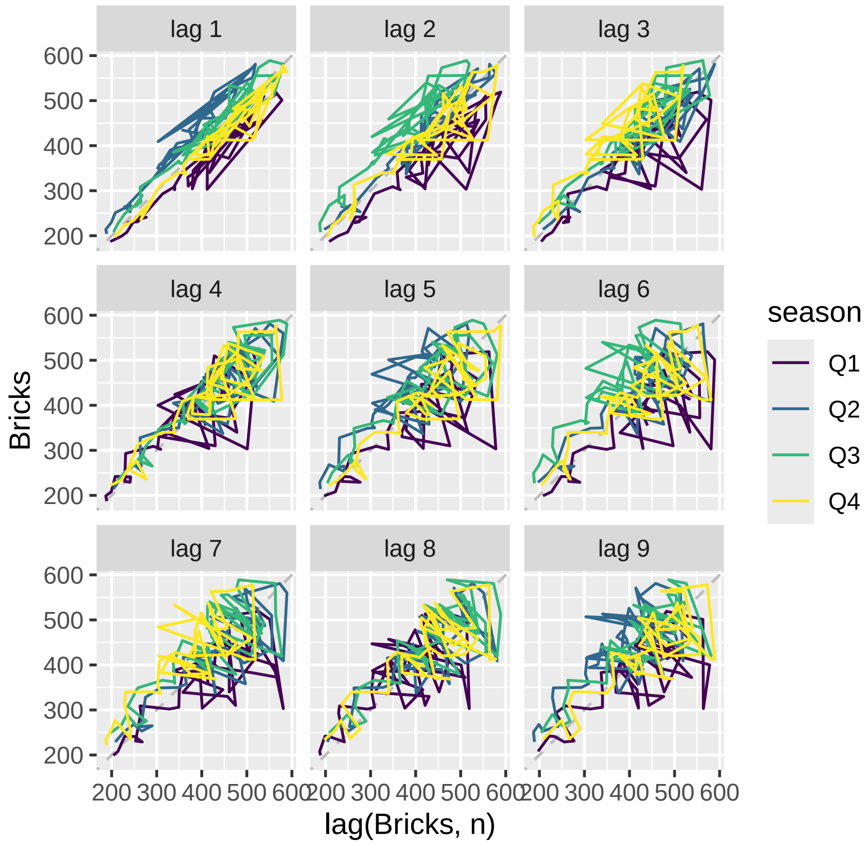

```
aus_production |> gg_subseries(Bricks) + labs("Subseries plot of Bricks by quarter")
```

```
## Warning: Removed 5 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

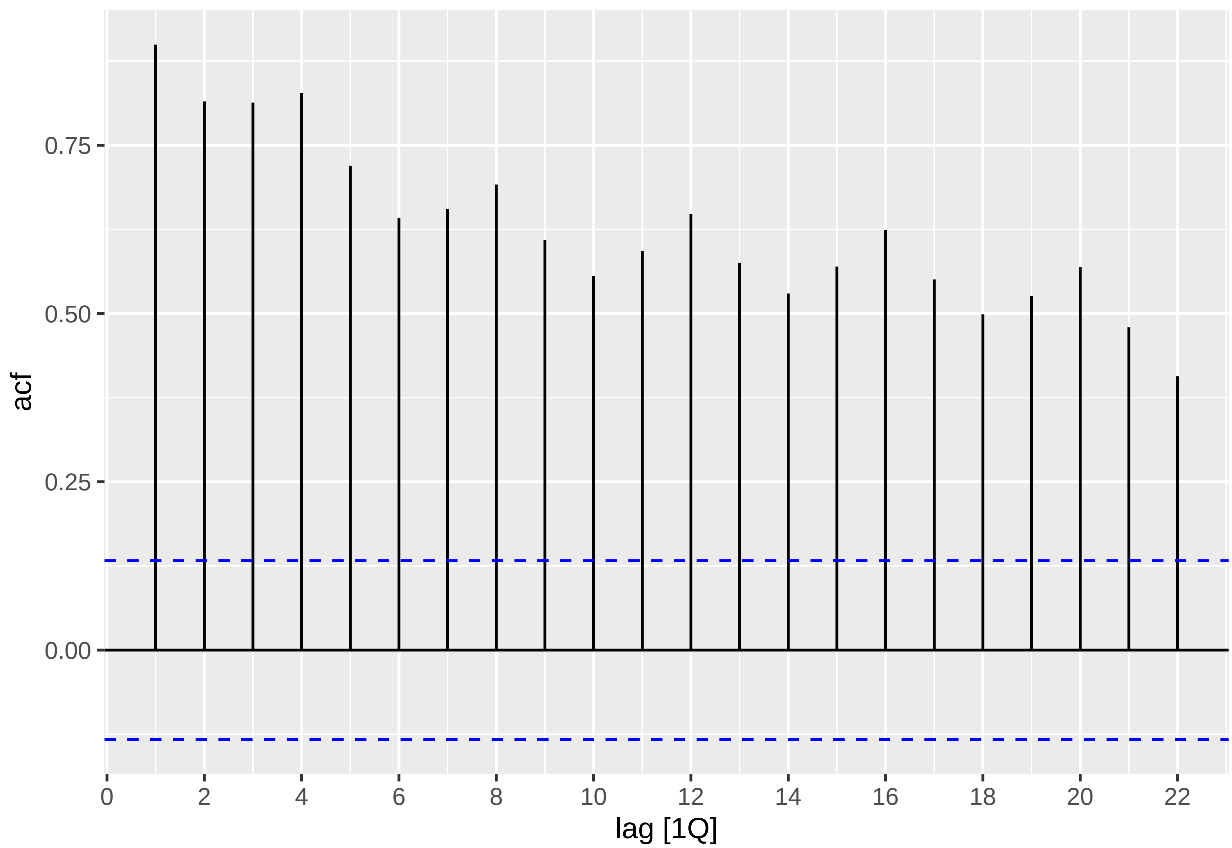



```r
aus_production |> gg_lag(Bricks) + labs("Lag plot  of Bricks overtime")
```

```
## Warning: Removed 20 rows containing missing values (gg_lag).
```




```
aus_production |> ACF(Bricks) |> autoplot() + labs("Obsvation of Bricks overtime")
```

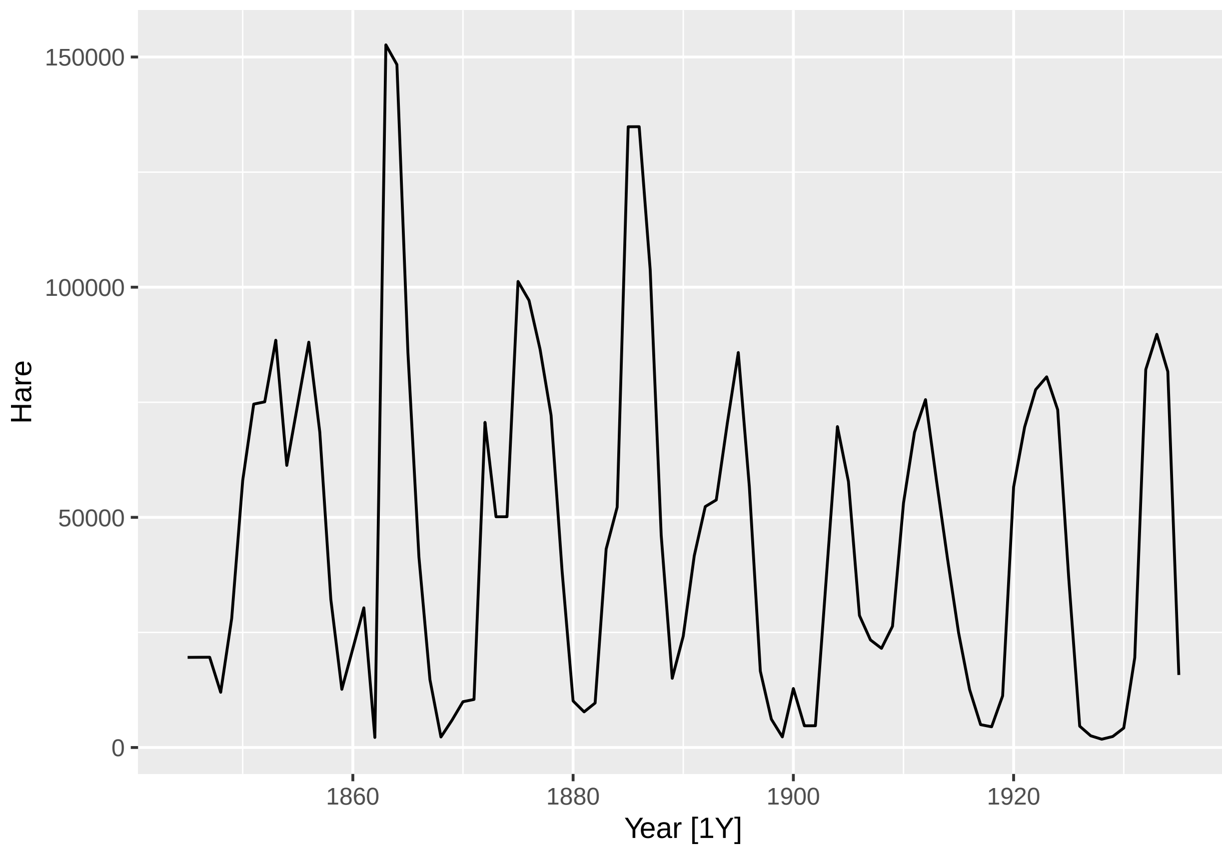


PELT

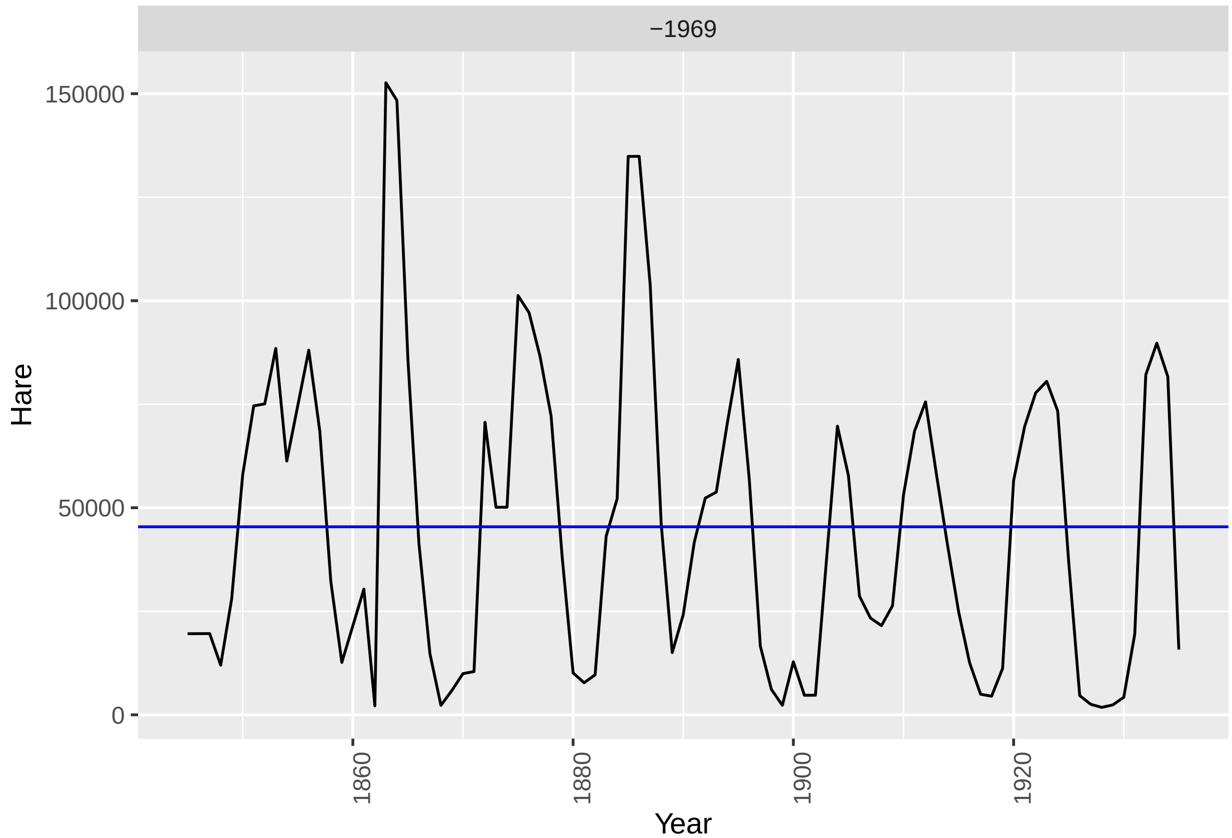
These visualizations analyze **hare population trends** over time.

1. **Autoplot** – Shows cyclical fluctuations in the hare population, suggesting a recurring pattern of population growth and decline.
2. **Subseries Plot** – Highlights seasonal variations in population levels, with an average population trend indicated by the blue line.
3. **Lag Plot** – Reveals complex dependencies between past and present population levels, indicating nonlinear relationships.
4. **Autocorrelation Function (ACF) Plot** – Demonstrates significant autocorrelation, confirming strong cyclicality in the hare population.

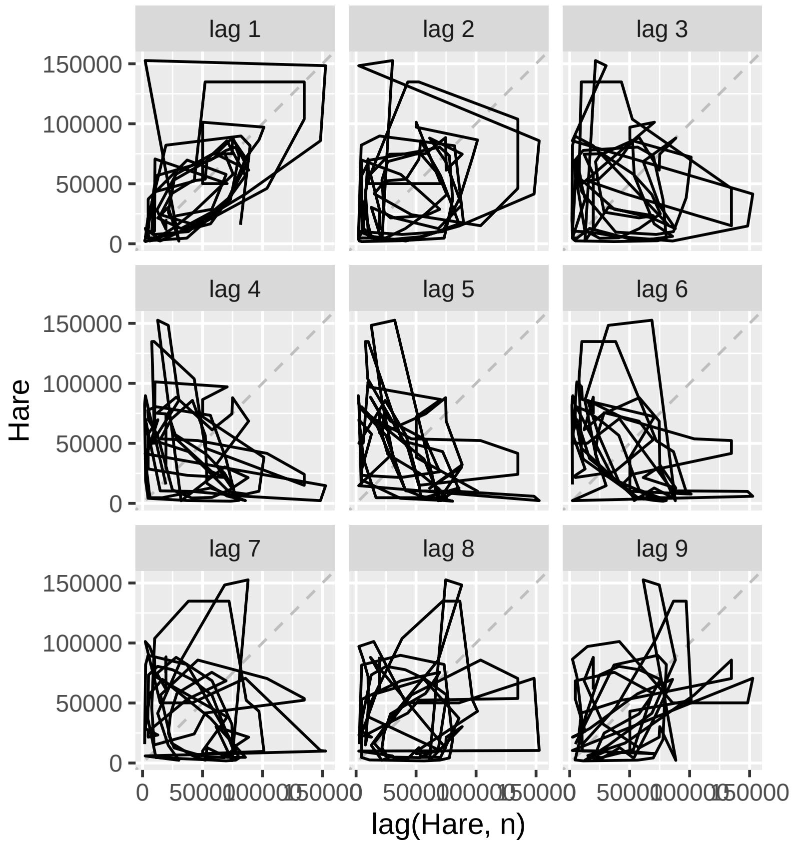
```
pelt|>
  autoplot(Hare) +
  labs("Autoplot")
```



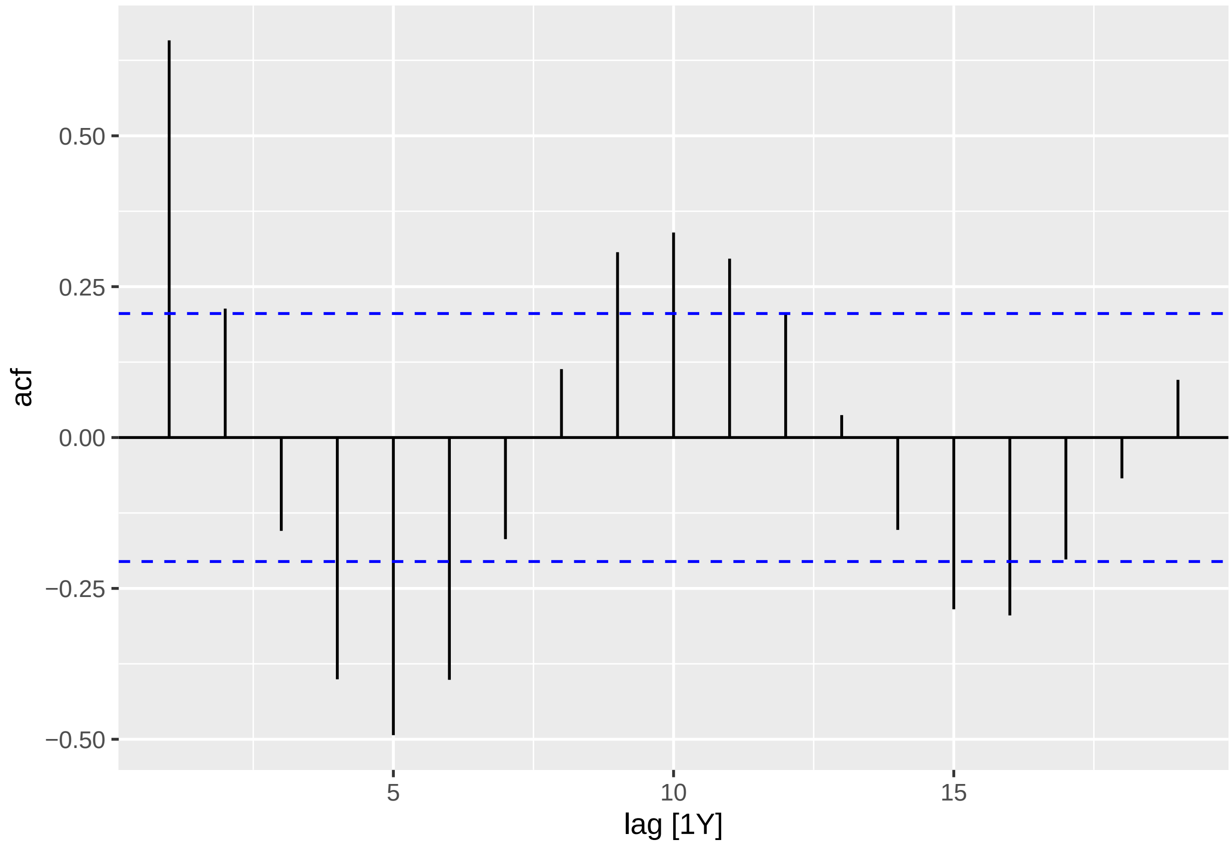

```r
pelt|>
  gg_subseries(Hare) +
  labs("Subseries Plot")
```




```r
pelt|>
  gg_lag(Hare) +
  labs("Autoplot")
```




```
pelt|> ACF(Hare) |>
  autoplot() +
  labs("Autoplot")
```



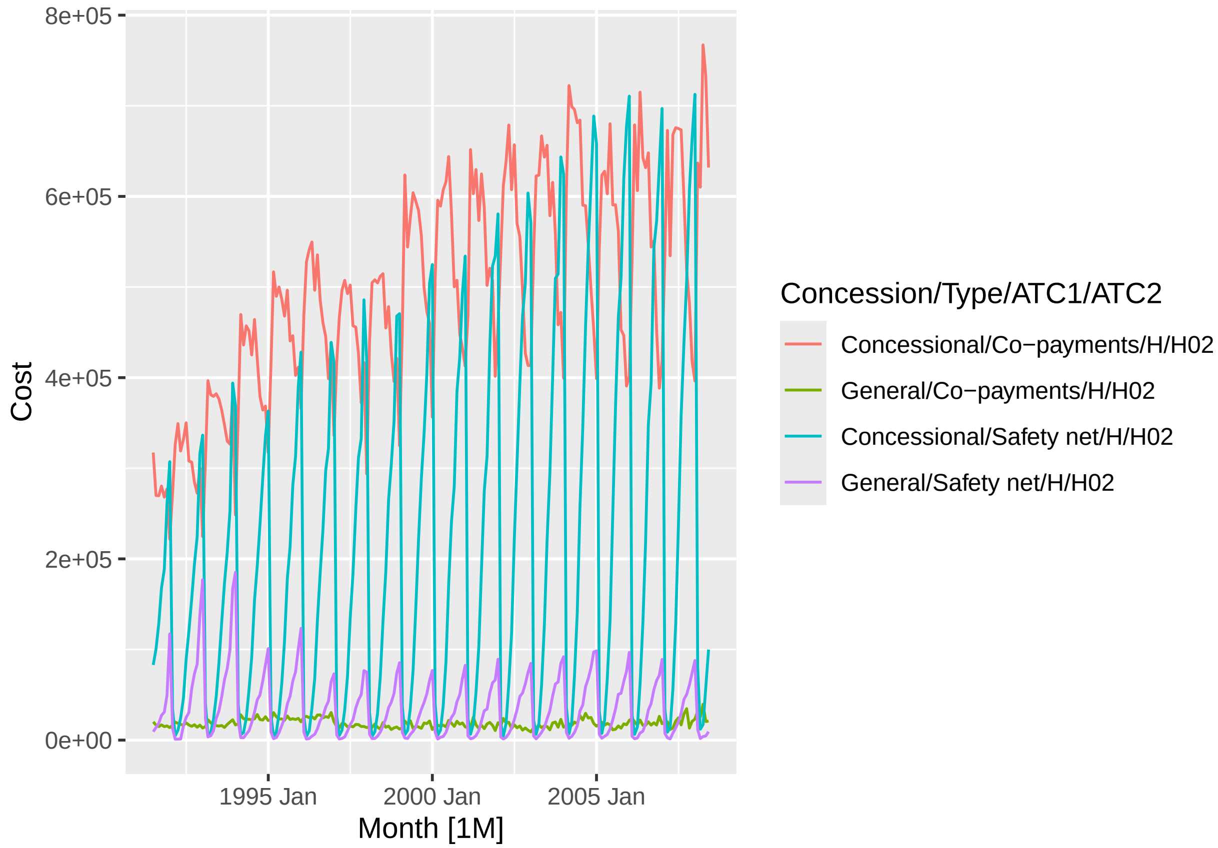
PBS

These visualizations analyze the **cost trends of pharmaceutical payments** over time.

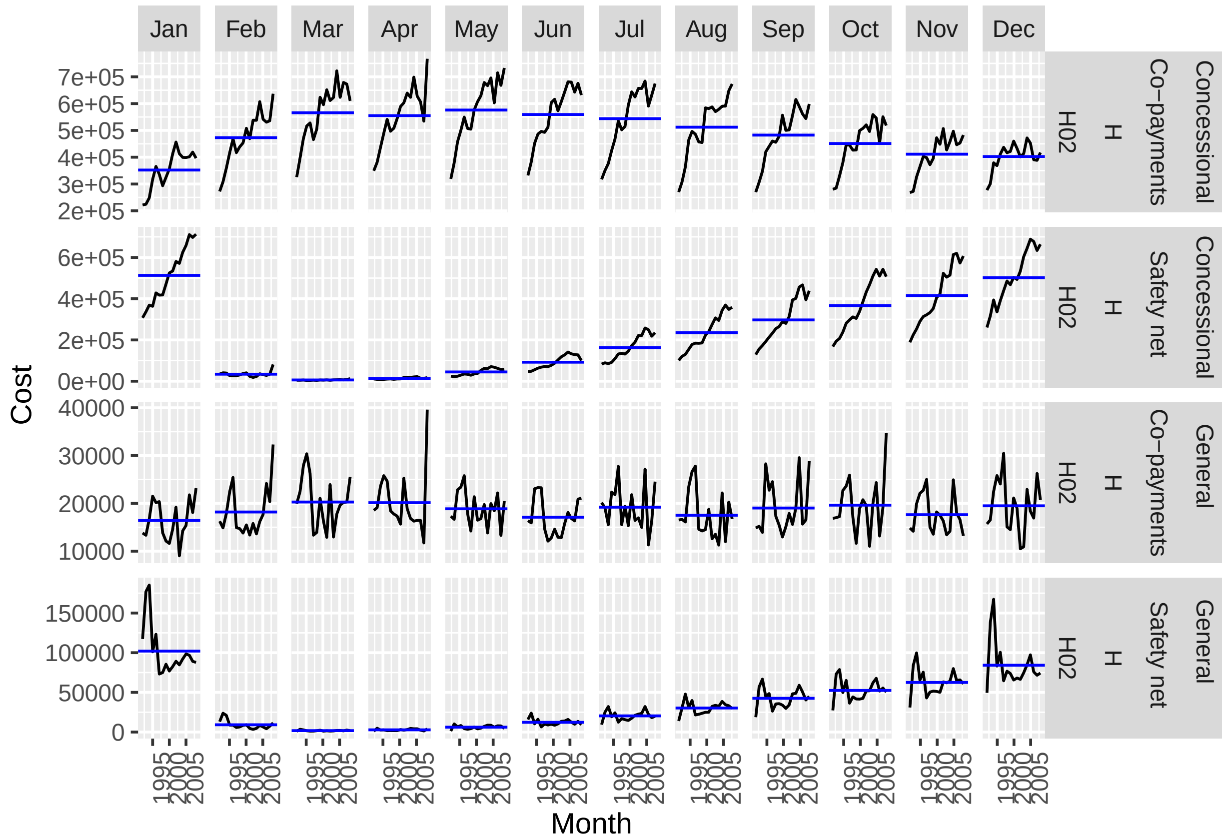
1. **Subseries Plot** – Displays monthly trends for different payment types, showing **consistent growth** over the years with seasonal variations.
2. **Autoplot** – Highlights the **long-term increase** in pharmaceutical costs across different concession categories. The **Concessional Co-payments (red line)** have the highest cost, followed by **Concessional Safety Net (blue line)**, while General categories remain lower.

These graphs suggest a **steady rise in pharmaceutical costs** with clear **seasonal patterns** and a **significant difference between concessional and general payments**.

```
PBS %>%
  filter(ATC2 == "H02")  |> autoplot(Cost) + labs("Observation of H02 over time")
```




```
PBS %>%
  filter(ATC2 == "H02")  |> gg_subseries(Cost) + labs("Observation of H02 over time")
```



GAS

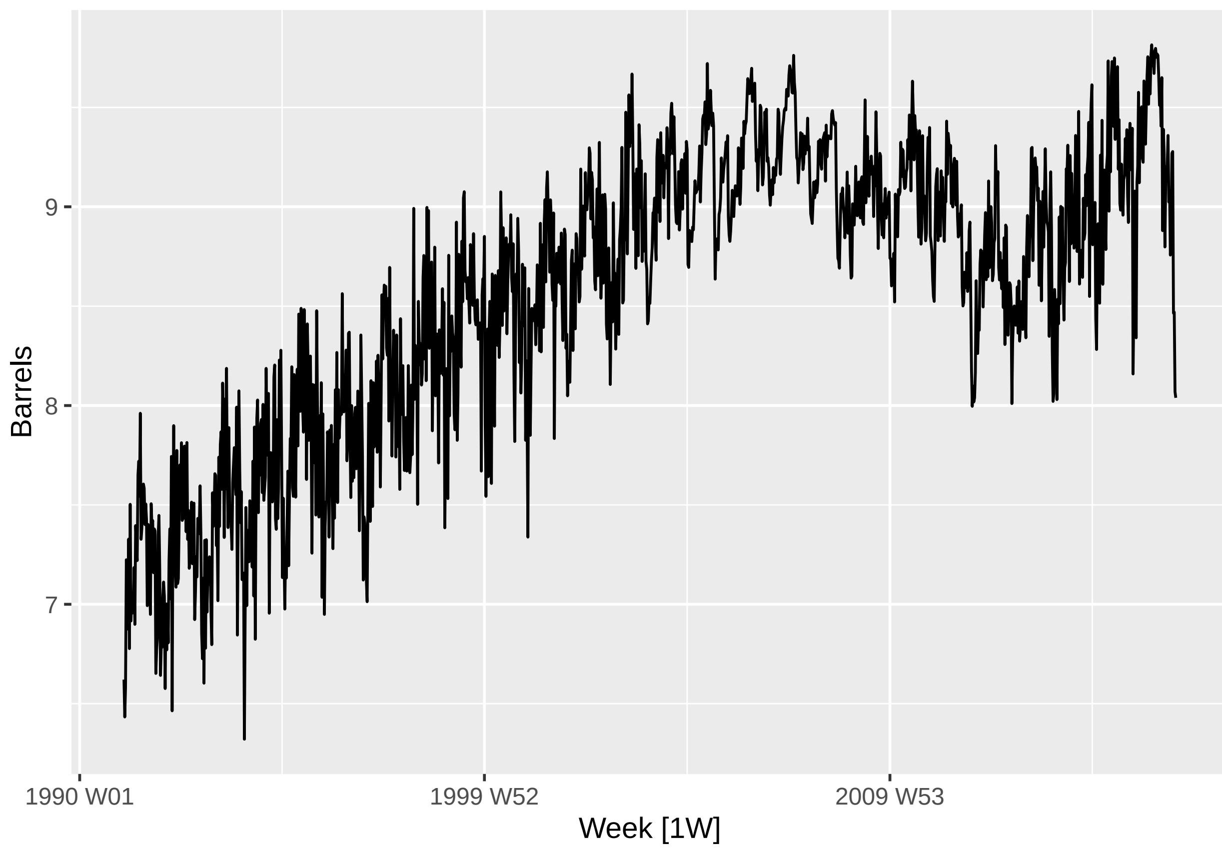
These visualizations analyze **weekly oil production trends** over time.

1. **Autoplot** – Shows a long-term **upward trend** in oil production from 1990 to 2010, with noticeable fluctuations.
2. **Seasonal Plot** – Highlights seasonal patterns, showing **higher production in later years (2015)** compared to earlier years (1995).
3. **Subseries Plot** – Displays variations in weekly production, with certain weeks experiencing **higher volatility** than others.
4. **Lag Plot** – Indicates strong autocorrelation, meaning past production levels influence future values.
5. **Autocorrelation Function (ACF) Plot** – Confirms a **high degree of correlation** over time, reinforcing the **persistence of trends**.

These graphs suggest **steady growth with seasonal fluctuations**, making forecasting essential for supply planning.

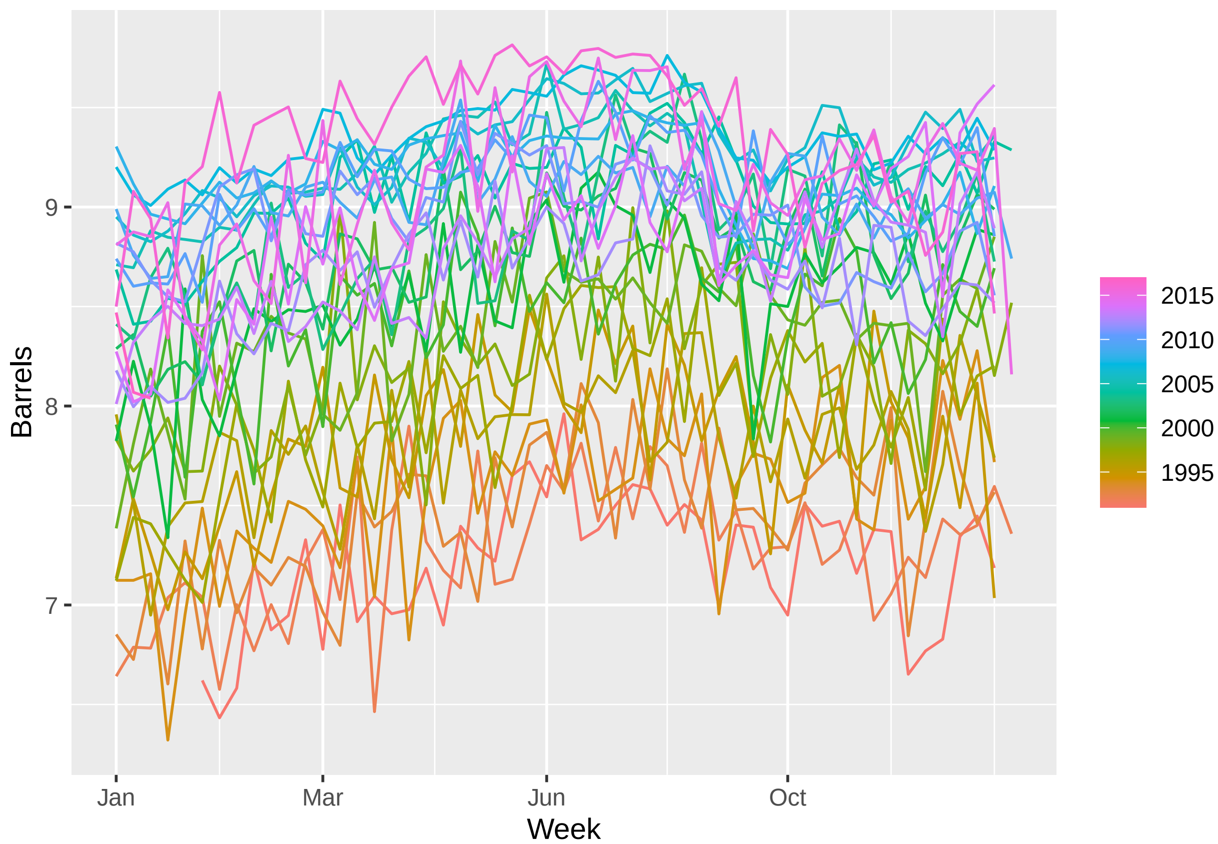
```
us_gasoline |> autoplot() + labs("Observation of Natutal gas Barrels overtime")
```

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



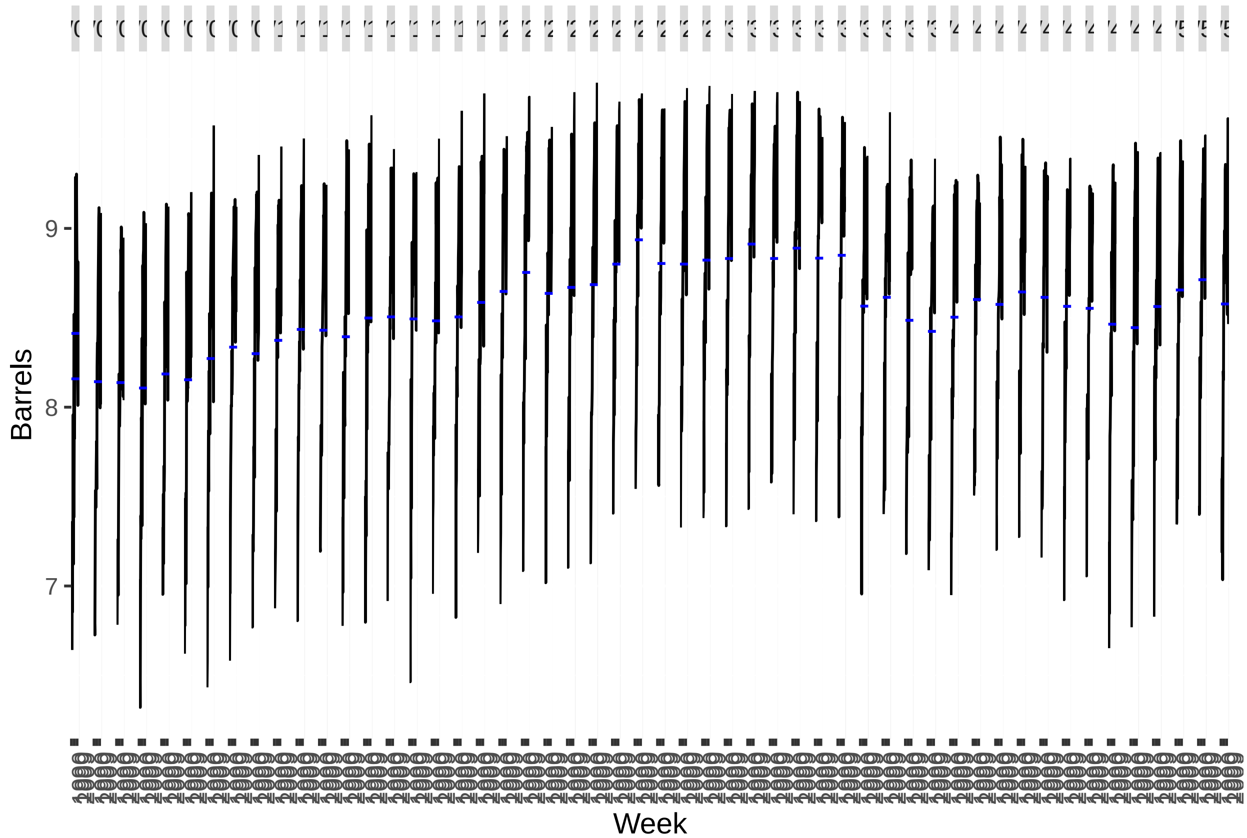

```
us_gasoline |> gg_season() + labs("Seasonal plot of Barrels by quarter")
```

```
## Plot variable not specified, automatically selected `y = Barrels`
```



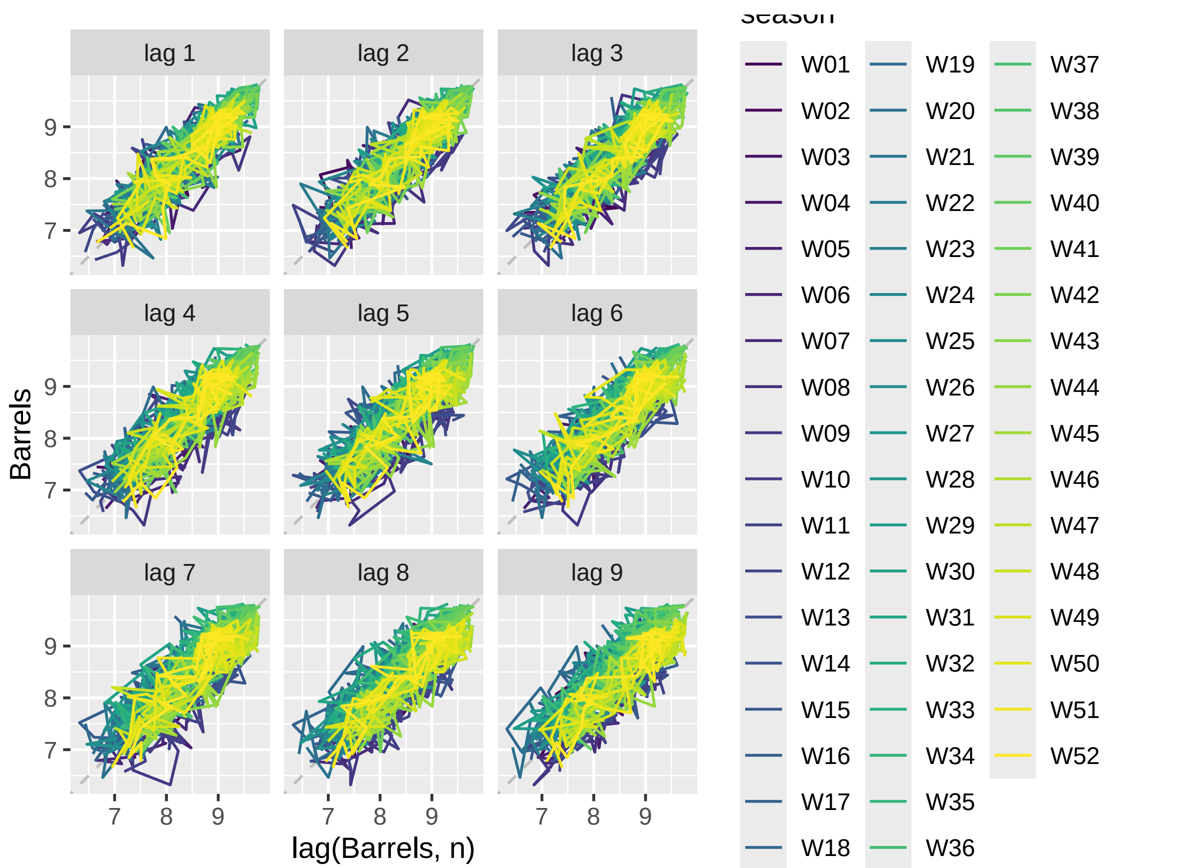

```
us_gasoline |> gg_subseries() + labs("Subseries plot of Barrels by quarter")
```

```
## Plot variable not specified, automatically selected `y = Barrels`
```



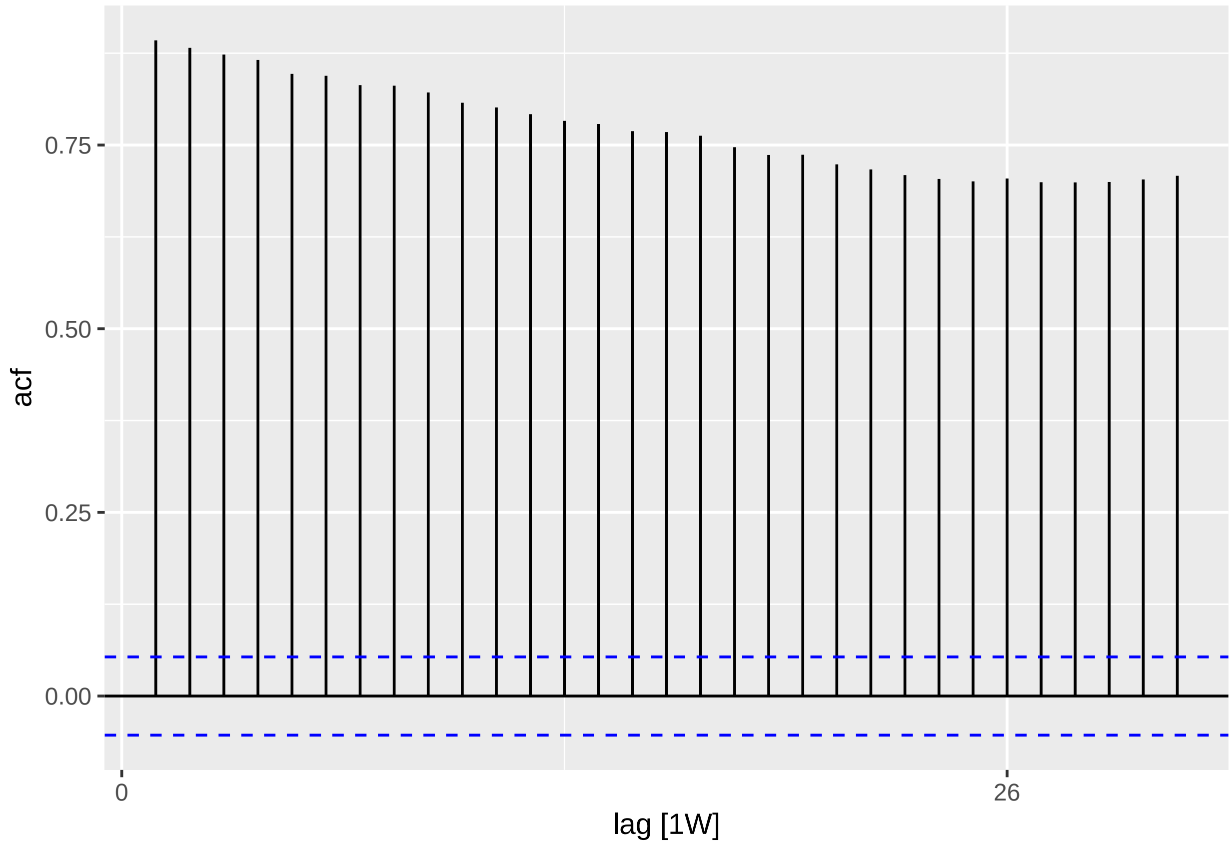

```r
us_gasoline |> gg_lag() + labs("Lag plot  of Bricks overtime")
```

```
## Plot variable not specified, automatically selected 'y = Barrels'
```




```
us_gasoline |> ACF() |> autoplot() + labs("Obsvation of Bricks overtime")
```

```
## Response variable not specified, automatically selected `var = Barrels`
```


0.75
0.50
0.25
0.00
acf
0
26
lag [1W]