207 Final Project - Trading Analysis and Stock Price Prediction

```
rm(list = ls())
library(tidyverse)
library(magrittr)
library(patchwork)

library(lubridate)

library(tsibble)
library(feasts)
install.packages('forecast')
library(forecast)

library(sandwich)
library(lmtest)

library(nycflights13)
install.packages('blsR')
library(blsR)

install.packages('gridExtra')
library(gridExtra)
```

Load in the Apple Stock Data

```
apple_df <- read_csv("apple_processed_data.csv")
```

```
## Rows: 2223 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): news
## dbl  (8): close, open, lowest, highest, total_vol, mean_vol, std_vol, is_up
## date (1): Date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(apple_df)
```

```
## # A tibble: 6 x 10
##   Date       close  open lowest highest total_vol mean_vol std_vol news    is_up
##   <date>     <dbl> <dbl>  <dbl>   <dbl>     <dbl>    <dbl>   <dbl> <chr>   <dbl>
## 1 2015-01-02  24.5  24.9   24.0    25.0 188181988  482518. 453959. "[\"Is~     0
## 2 2015-01-05  23.8  24.2   23.6    24.4 200586492  514324. 426711. "[\"Is~     0
## 3 2015-01-06  23.8  23.8   23.4    24.1 237766160  609657. 452107. "[\"Is~     1
## 4 2015-01-07  24.1  24.0   23.9    24.3 137809632  353358. 315345. "Apple~     1
## 5 2015-01-08  25.1  24.4   24.3    25.2 201020076  515436. 344929. "Xiaom~     1
## 6 2015-01-09  25.1  25.2   24.7    25.4 194014564  497473. 418949. "Infos~     0
```

```r
head(apple_df, 10)
```

```
## # A tibble: 10 x 10
##    Date       close  open lowest highest total_vol mean_vol std_vol news   is_up
##    <date>     <dbl> <dbl>  <dbl>   <dbl>     <dbl>    <dbl>   <dbl> <chr>  <dbl>
##  1 2015-01-02  24.5  24.9   24.0    25.0 188181988  482518. 453959. "[\"I~     0
##  2 2015-01-05  23.8  24.2   23.6    24.4 200586492  514324. 426711. "[\"I~     0
##  3 2015-01-06  23.8  23.8   23.4    24.1 237766160  609657. 452107. "[\"I~     1
##  4 2015-01-07  24.1  24.0   23.9    24.3 137809632  353358. 315345. "Appl~     1
##  5 2015-01-08  25.1  24.4   24.3    25.2 201020076  515436. 344929. "Xiao~     1
##  6 2015-01-09  25.1  25.2   24.7    25.4 194014564  497473. 418949. "Info~     0
##  7 2015-01-12  24.5  25.1   24.4    25.3 177972644  456340. 357777. "Zoma~     1
##  8 2015-01-13  24.7  25.0   24.4    25.3 242520180  621847. 402626. "Zoma~     0
##  9 2015-01-14  24.6  24.5   24.3    24.8 166619520  427230. 308053. "['60~     0
## 10 2015-01-15  23.9  24.5   23.9    24.7 205693220  527419. 391931. "['60~     0
```

```r
summary(apple_df)
```

```
##       Date                close             open            lowest      
##  Min.   :2015-01-02   Min.   : 20.82   Min.   : 20.75   Min.   : 20.60  
##  1st Qu.:2017-03-18   1st Qu.: 32.88   1st Qu.: 32.84   1st Qu.: 32.67  
##  Median :2019-06-04   Median : 50.80   Median : 50.69   Median : 50.20  
##  Mean   :2019-06-02   Mean   : 80.28   Mean   : 80.26   Mean   : 79.35  
##  3rd Qu.:2021-08-16   3rd Qu.:136.53   3rd Qu.:136.46   3rd Qu.:134.39  
##  Max.   :2023-10-31   Max.   :196.22   Max.   :195.96   Max.   :195.02  
##     highest          total_vol            mean_vol          std_vol       
##  Min.   : 21.14   Min.   : 25621051   Min.   :  65695   Min.   :  62676  
##  1st Qu.: 33.03   1st Qu.: 59097242   1st Qu.: 151690   1st Qu.: 139764  
##  Median : 51.21   Median : 81815982   Median : 209888   Median : 196982  
##  Mean   : 81.13   Mean   : 97459556   Mean   : 250168   Mean   : 247092  
##  3rd Qu.:138.38   3rd Qu.:117076364   3rd Qu.: 300196   3rd Qu.: 285696  
##  Max.   :197.96   Max.   :546209744   Max.   :1400538   Max.   :2157419  
##      news               is_up       
##  Length:2223        Min.   :0.0000  
##  Class :character   1st Qu.:0.0000  
##  Mode  :character   Median :1.0000  
##                     Mean   :0.5263  
##                     3rd Qu.:1.0000  
##                     Max.   :1.0000
```

```r
apple_tsib <- apple_df %>% as_tsibble(index=Date)
apple_tsib
```

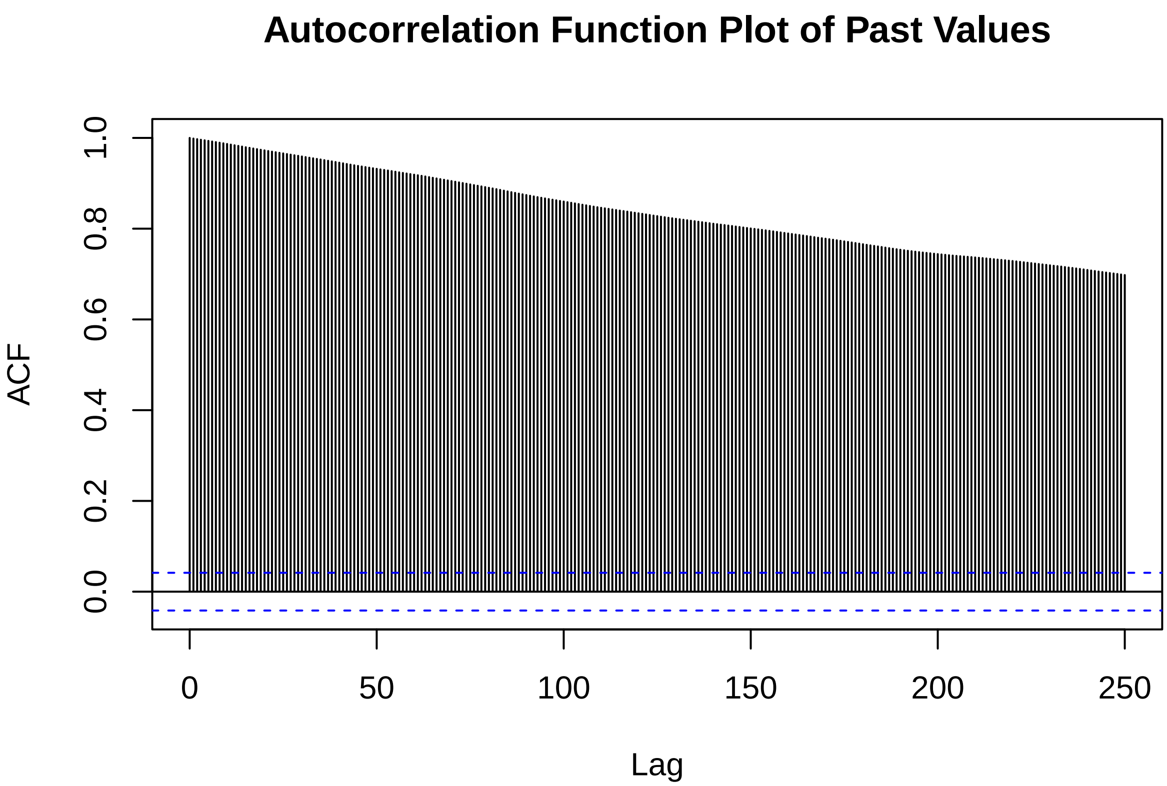
```
## # A tsibble: 2,223 x 10 [1D]
##    Date       close  open lowest highest total_vol mean_vol std_vol news   is_up
##    <date>     <dbl> <dbl>  <dbl>   <dbl>     <dbl>    <dbl>   <dbl> <chr>  <dbl>
##  1 2015-01-02  24.5  24.9   24.0    25.0 188181988  482518. 453959. "[\"I~     0
##  2 2015-01-05  23.8  24.2   23.6    24.4 200586492  514324. 426711. "[\"I~     0
##  3 2015-01-06  23.8  23.8   23.4    24.1 237766160  609657. 452107. "[\"I~     1
##  4 2015-01-07  24.1  24.0   23.9    24.3 137809632  353358. 315345. "Appl~     1
##  5 2015-01-08  25.1  24.4   24.3    25.2 201020076  515436. 344929. "Xiao~     1
##  6 2015-01-09  25.1  25.2   24.7    25.4 194014564  497473. 418949. "Info~     0
##  7 2015-01-12  24.5  25.1   24.4    25.3 177972644  456340. 357777. "Zoma~     1
##  8 2015-01-13  24.7  25.0   24.4    25.3 242520180  621847. 402626. "Zoma~     0
##  9 2015-01-14  24.6  24.5   24.3    24.8 166619520  427230. 308053. "['60~     0
```

```
## 10 2015-01-15  23.9  24.5   23.9    24.7 205693220  527419. 391931. "['60~     0
## # i 2,213 more rows
```

```
apple_close_price_stock_plot <- apple_tsib |>
  ggplot(aes(x=Date, y = close)) +
  geom_line() +
  labs(x = "Date", y = "Closing Price",
       title = "Apple Stock Closing Price from 2015-2024")
apple_close_price_stock_plot
```

Aside from the clear non-stationarity, we also see the series having an increased variance throughout the right side of the plot.

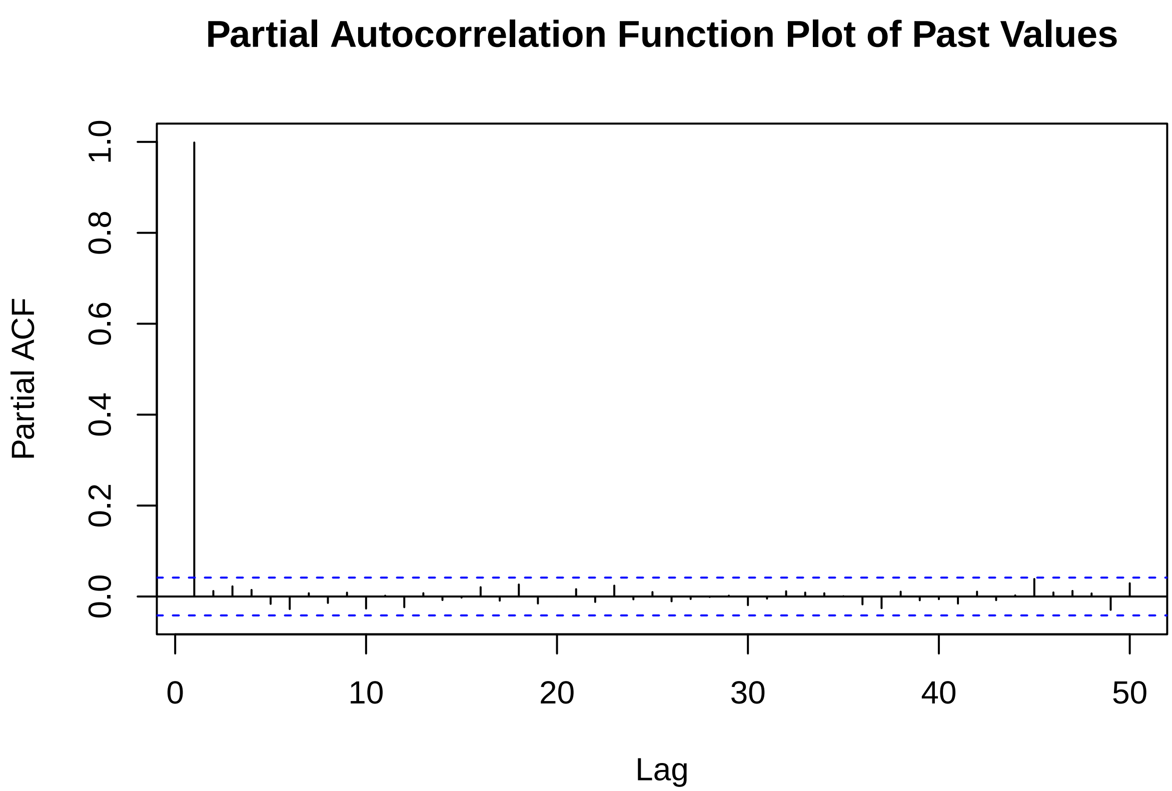
```
acf(apple_tsib$close, lag.max = 250,
    main = "Autocorrelation Function Plot of Past Values")
```



This plot reveals statistically significant correlations between past values of the closing price and current/future closing price values. The lack of "scalloped" shape in the autocorrelations does not suggest evidence of strong seasonality.

Significant autocorrelations with lags past 200 hint at underlying trend (nonstationary), which rules out using only a Moving Average (MA) process.

```
pacf(apple_tsib$close, lag.max = 50,
     main = "Partial Autocorrelation Function Plot of Past Values")
```



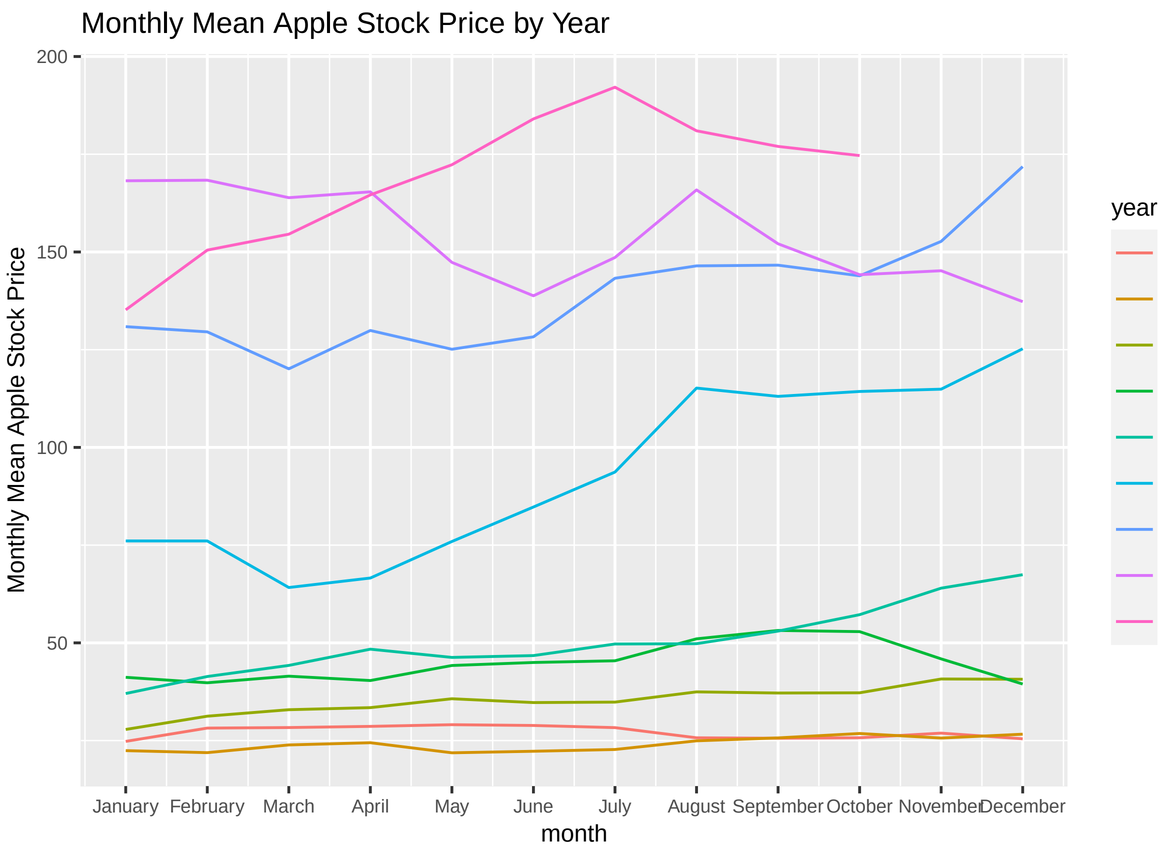
The partial autocorrelations drop drastically after the first partial autocorrelation and remain insignificant throughout.

```r
apple_tsib_month_year <-apple_tsib
monthly_price_avg_tsib <- apple_tsib_month_year %>%
index_by(yearMonth=yearmonth(Date)) %>% group_by(yearMonth) %>%
summarise(mean_price=mean(close))

monthly_price_avg_tsib
```

```
## # A tsibble: 106 x 2 [1M]
##    yearMonth mean_price
##        <mth>      <dbl>
##  1  2015 Jan       24.8
##  2  2015 Feb       28.2
##  3  2015 Mar       28.3
##  4  2015 Apr       28.6
##  5  2015 May       29.1
##  6  2015 Jun       28.9
##  7  2015 Jul       28.3
##  8  2015 Aug       25.7
##  9  2015 Sep       25.6
## 10  2015 Oct       25.7
## # i 96 more rows
```

```r
monthly_price_avg_tsib$year <- year(monthly_price_avg_tsib$yearMonth)
monthly_price_avg_tsib$month <- month(monthly_price_avg_tsib$yearMonth)
monthly_price_avg <- as_tibble(monthly_price_avg_tsib) %>%
  select(mean_price, year, month)

monthly_price_avg$year <- as.character(monthly_price_avg$year)
```

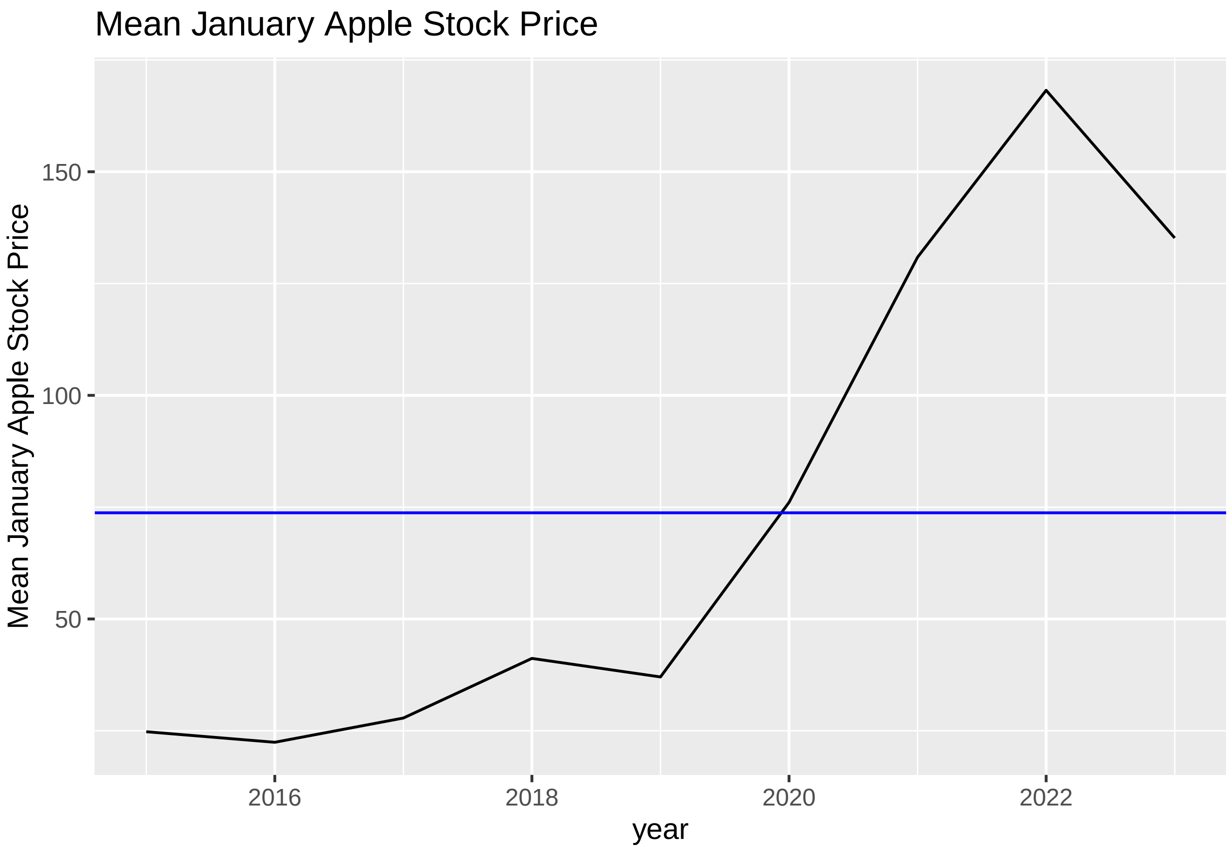
```r
options(repr.plot.width =15, repr.plot.height =15)

monthly_avg_plot <- monthly_price_avg %>%
  ggplot(aes(x=month, y = mean_price, color= year)) +
  geom_line() + ylab("Monthly Mean Apple Stock Price") +
  scale_x_continuous(
    breaks = seq_along(month.name),
    labels = month.name
  ) +
  ggtitle('Monthly Mean Apple Stock Price by Year') +
  theme(text = element_text(size = 9))
monthly_avg_plot
```



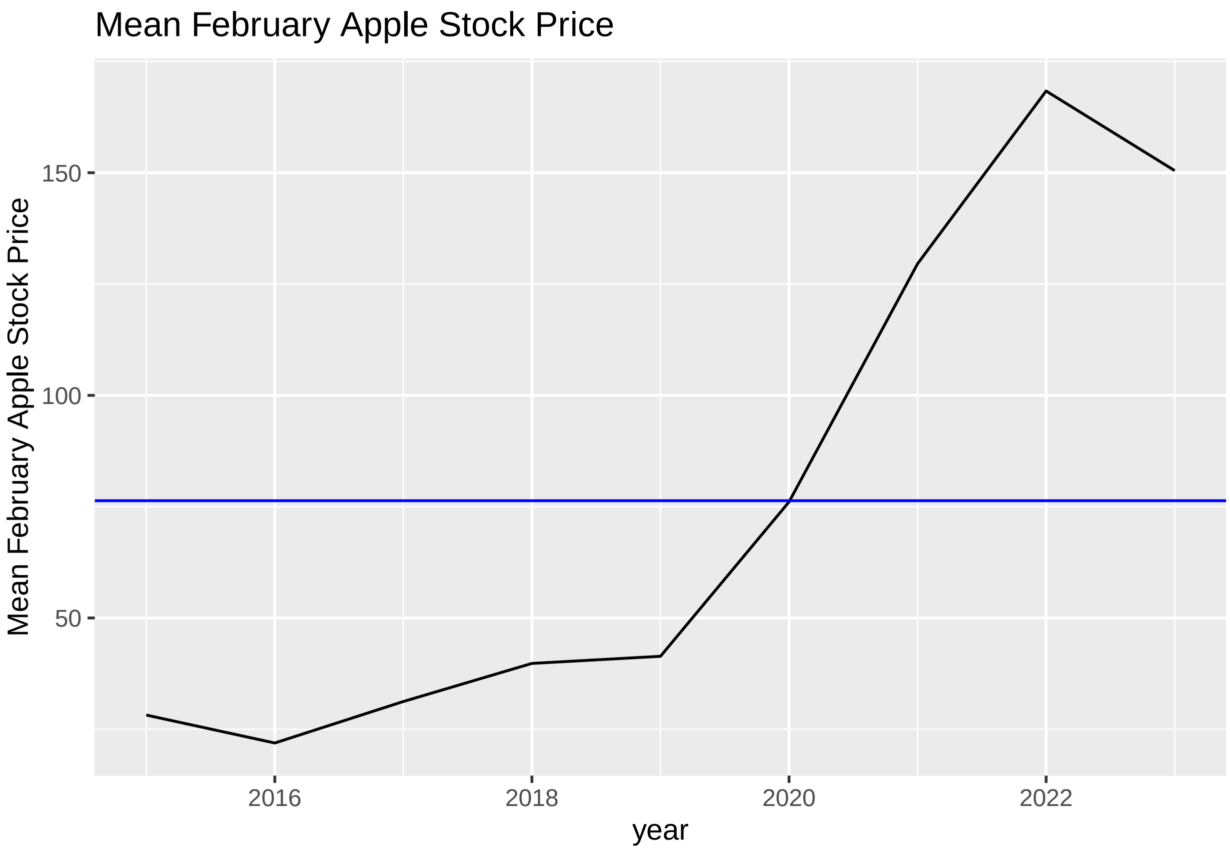
We see that Apple stock was at a low in 2016 before rising consistently from 2017 to 2023. We see that stocks rise slightly throughout the year in 2016. Stocks rose a little but fell slightly throughout 2015. Stocks rose gradually throughout 2017. Stocks experienced a rise but late fall in 2018. Stocks rose by almost 15% throughout 2019. Stocks rose dramatically in 2020. Overall, the 2021 stocks were higher than overall stock prices in any of the previous years since 2015. Stocks declined slightly throughout 2022. Finally, stocks rose significantly in 2023 until July, before falling through October. We cannot really see strong evidence of seasonality. However, we can later look at the components of the monthly average price to detect any seasonal component.

```r
january_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>%
  select(mean_price, year, month) %>% filter(month==1)
jan_apple_price_plot <- january_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean January Apple Stock Price") +
```

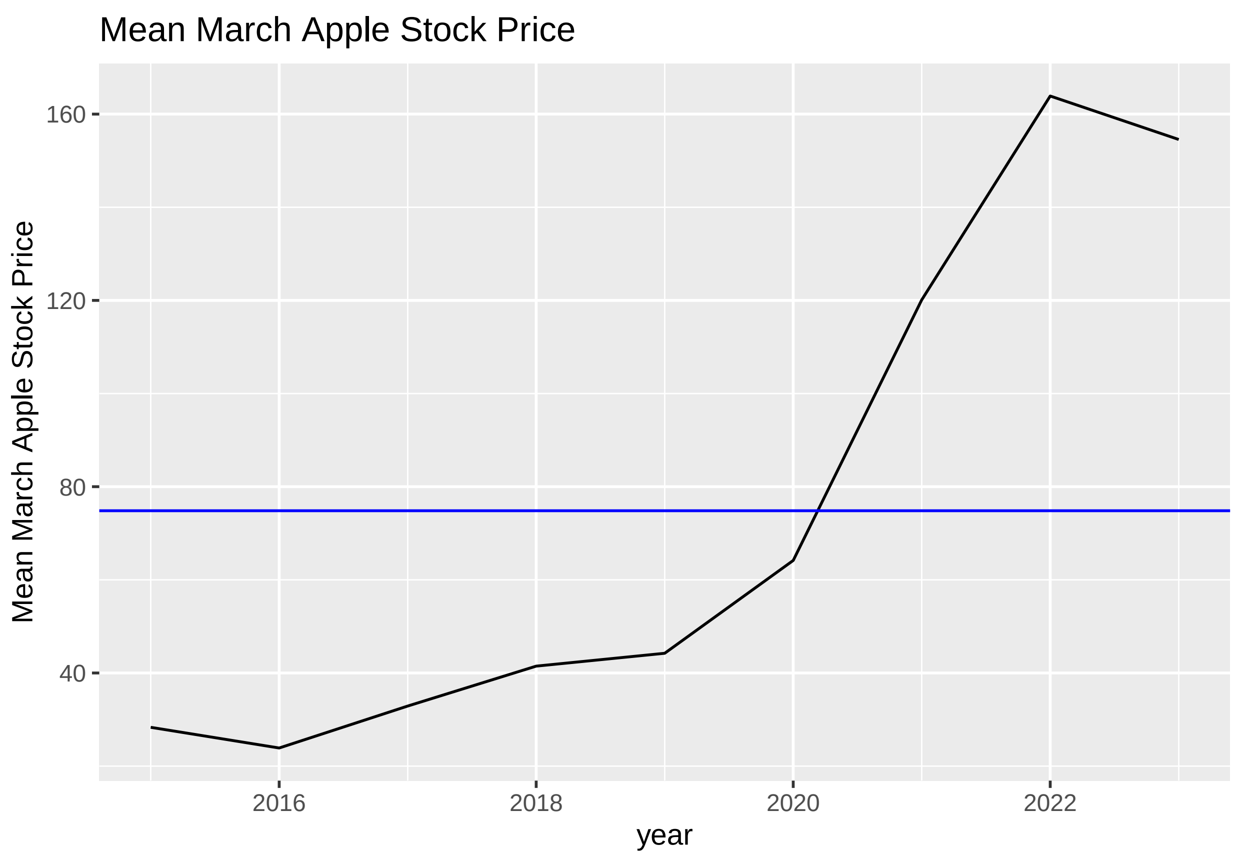
```
  ggtitle('Mean January Apple Stock Price') +
  geom_hline(yintercept = mean(january_monthly_price$mean_price), color="blue")
jan_apple_price_plot
```



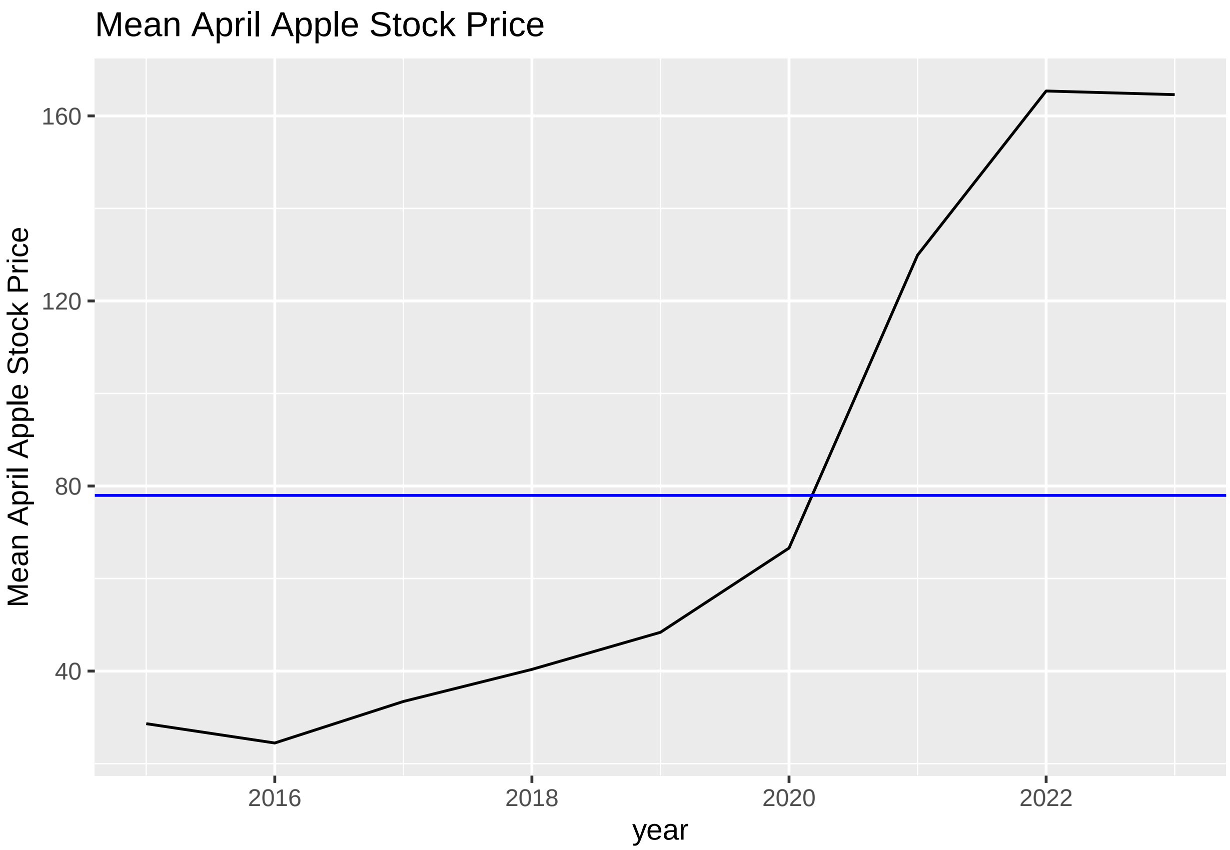

```
feb_monthly_price <- monthly_price_avg_tsib %>%
  as_tibble() %>% select(mean_price, year, month) %>%
  filter(month==2)
feb_apple_price_plot <- feb_monthly_price %>%
  ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean February Apple Stock Price") +
  ggtitle('Mean February Apple Stock Price') +
  geom_hline(yintercept = mean(feb_monthly_price$mean_price), color="blue")
feb_apple_price_plot
```



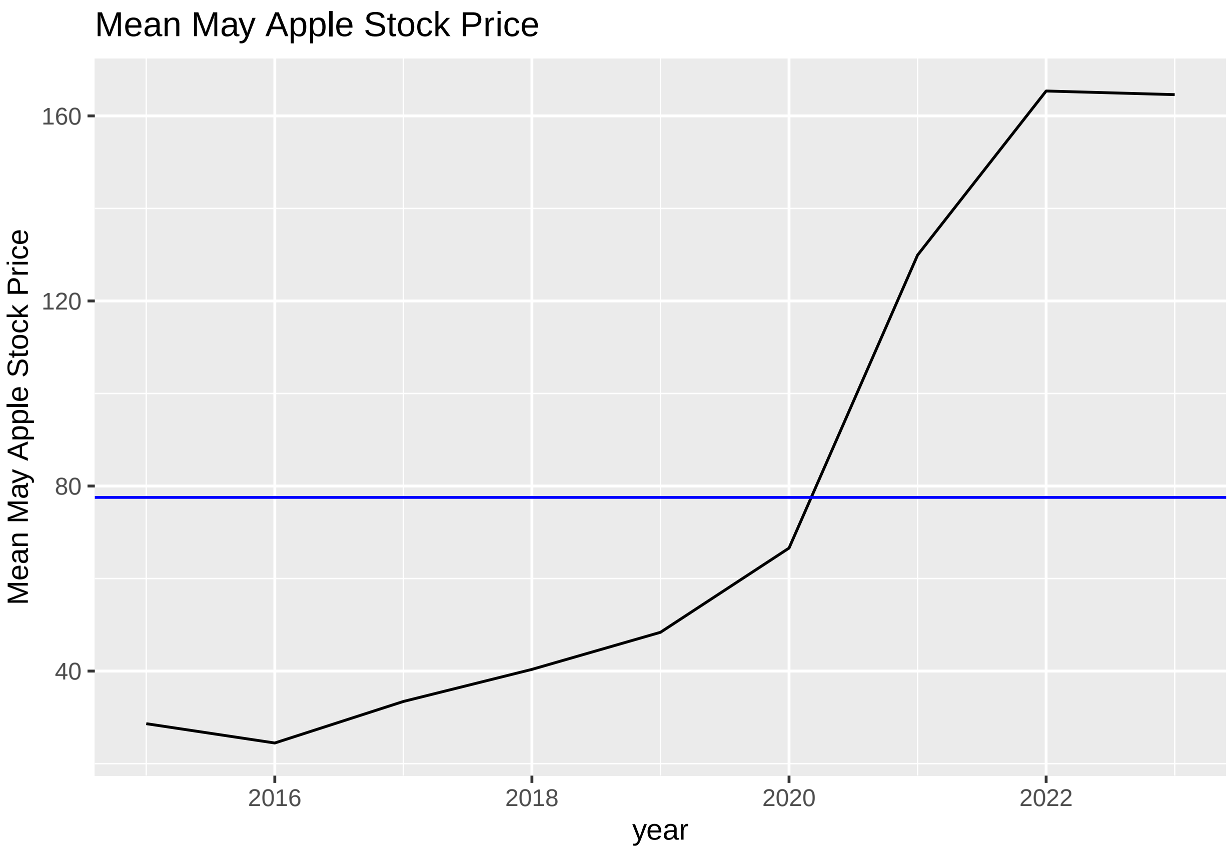

```
mar_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>%
  select(mean_price, year, month) %>% filter(month==3)
mar_apple_price_plot <- mar_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean March Apple Stock Price") +
  ggtitle('Mean March Apple Stock Price') +
  geom_hline(yintercept = mean(mar_monthly_price$mean_price), color="blue")
mar_apple_price_plot
```



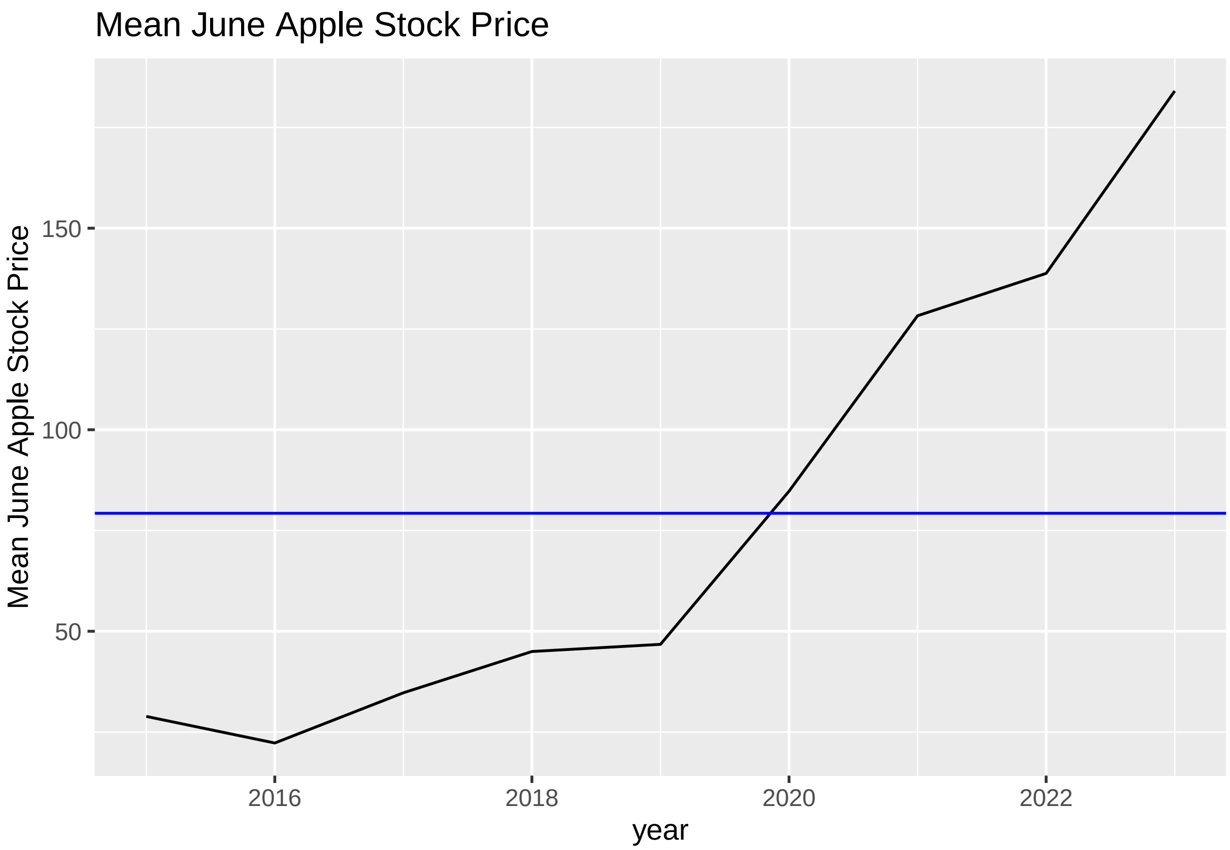

```r
apr_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>%
  select(mean_price, year, month) %>% filter(month==4)
apr_apple_price_plot <- apr_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean April Apple Stock Price") +
  ggtitle('Mean April Apple Stock Price') +
  geom_hline(yintercept = mean(apr_monthly_price$mean_price), color="blue")
apr_apple_price_plot
```



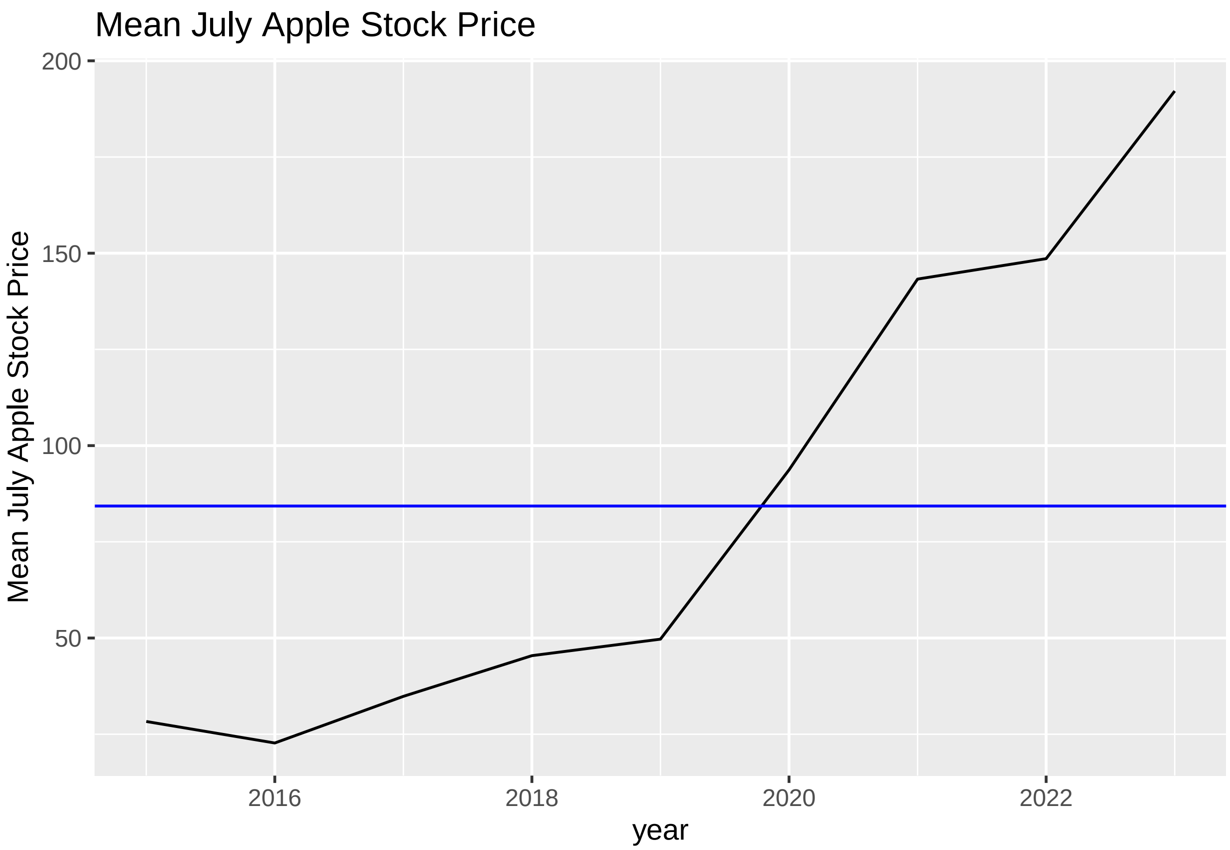

```
may_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>%
  select(mean_price, year, month) %>% filter(month==5)
may_apple_price_plot <- apr_monthly_price %>%
  ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean May Apple Stock Price") +
  ggtitle('Mean May Apple Stock Price') +
  geom_hline(yintercept = mean(may_monthly_price$mean_price), color="blue")
may_apple_price_plot
```



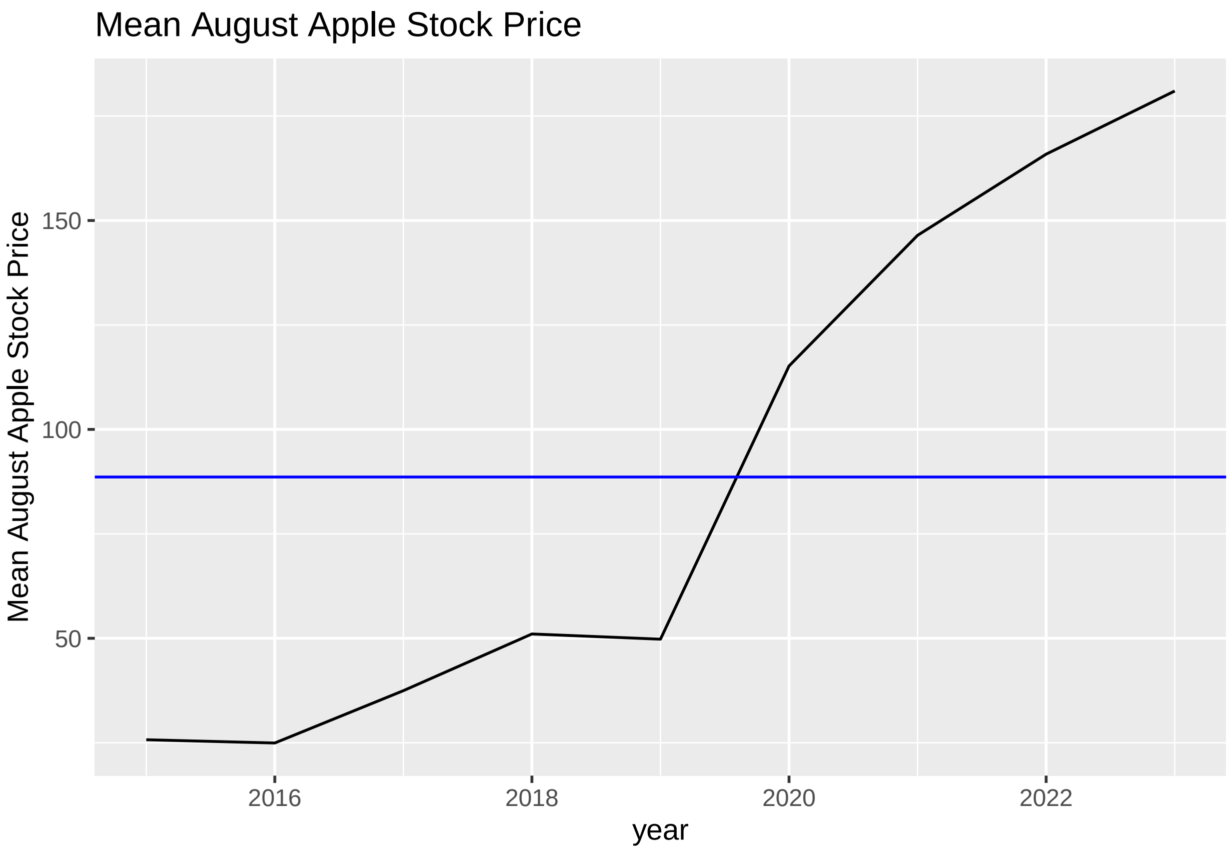

```
jun_monthly_price <- monthly_price_avg_tsib %>%
  as_tibble() %>% select(mean_price, year, month) %>% filter(month==6)
jun_apple_price_plot <- jun_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean June Apple Stock Price") +
  ggtitle('Mean June Apple Stock Price') +
  geom_hline(yintercept = mean(jun_monthly_price$mean_price), color="blue")
jun_apple_price_plot
```



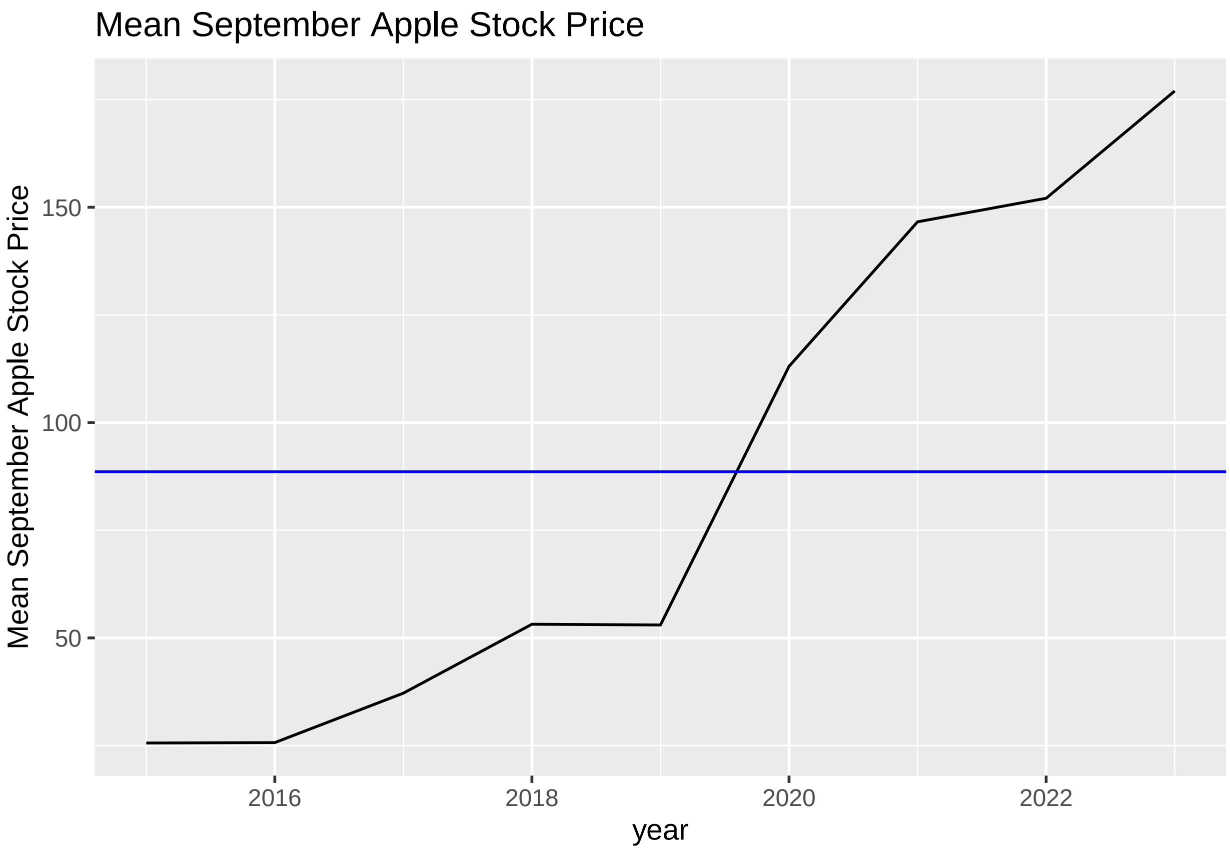

```
jul_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>%
  select(mean_price, year, month) %>% filter(month==7)
jul_apple_price_plot <- jul_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean July Apple Stock Price") +
  ggtitle('Mean July Apple Stock Price') +
  geom_hline(yintercept = mean(jul_monthly_price$mean_price), color="blue")
jul_apple_price_plot
```



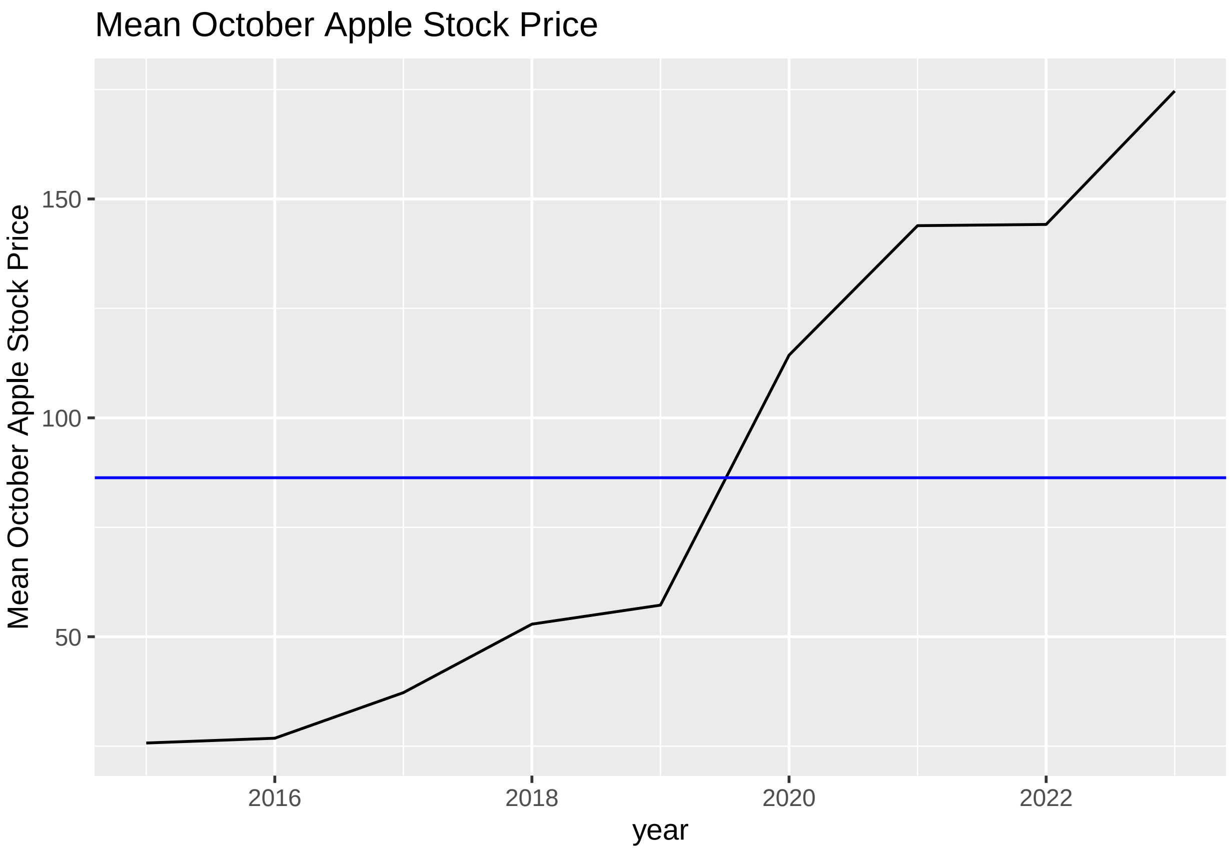

```r
aug_monthly_price <- monthly_price_avg_tsib %>%
  as_tibble() %>% select(mean_price, year, month) %>% filter(month==8)
aug_apple_price_plot <- aug_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean August Apple Stock Price") +
  ggtitle('Mean August Apple Stock Price') +
  geom_hline(yintercept = mean(aug_monthly_price$mean_price), color="blue")
aug_apple_price_plot
```



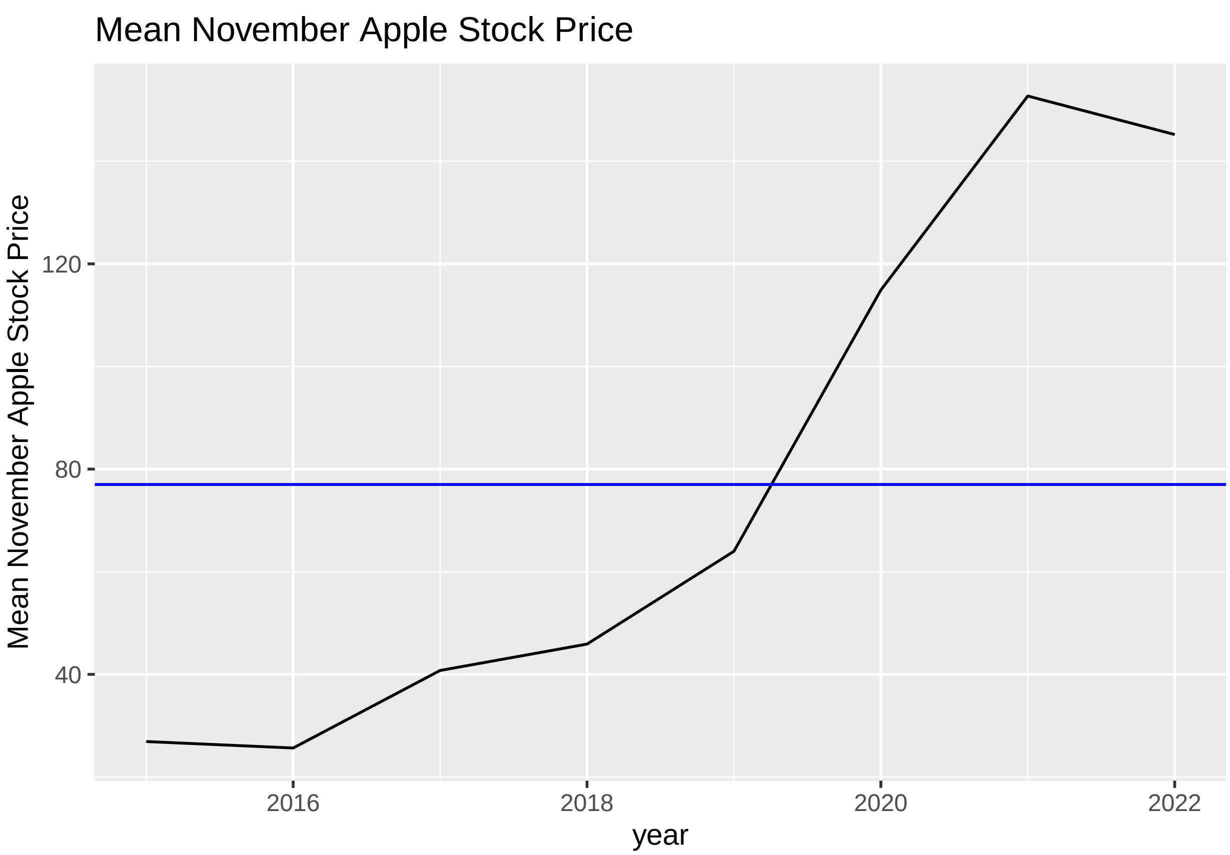

```r
sep_monthly_price <- monthly_price_avg_tsib %>%
  as_tibble() %>% select(mean_price, year, month) %>% filter(month==9)
sep_apple_price_plot <- sep_monthly_price %>%
  ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean September Apple Stock Price") +
  ggtitle('Mean September Apple Stock Price') +
  geom_hline(yintercept = mean(aug_monthly_price$mean_price), color="blue")
sep_apple_price_plot
```



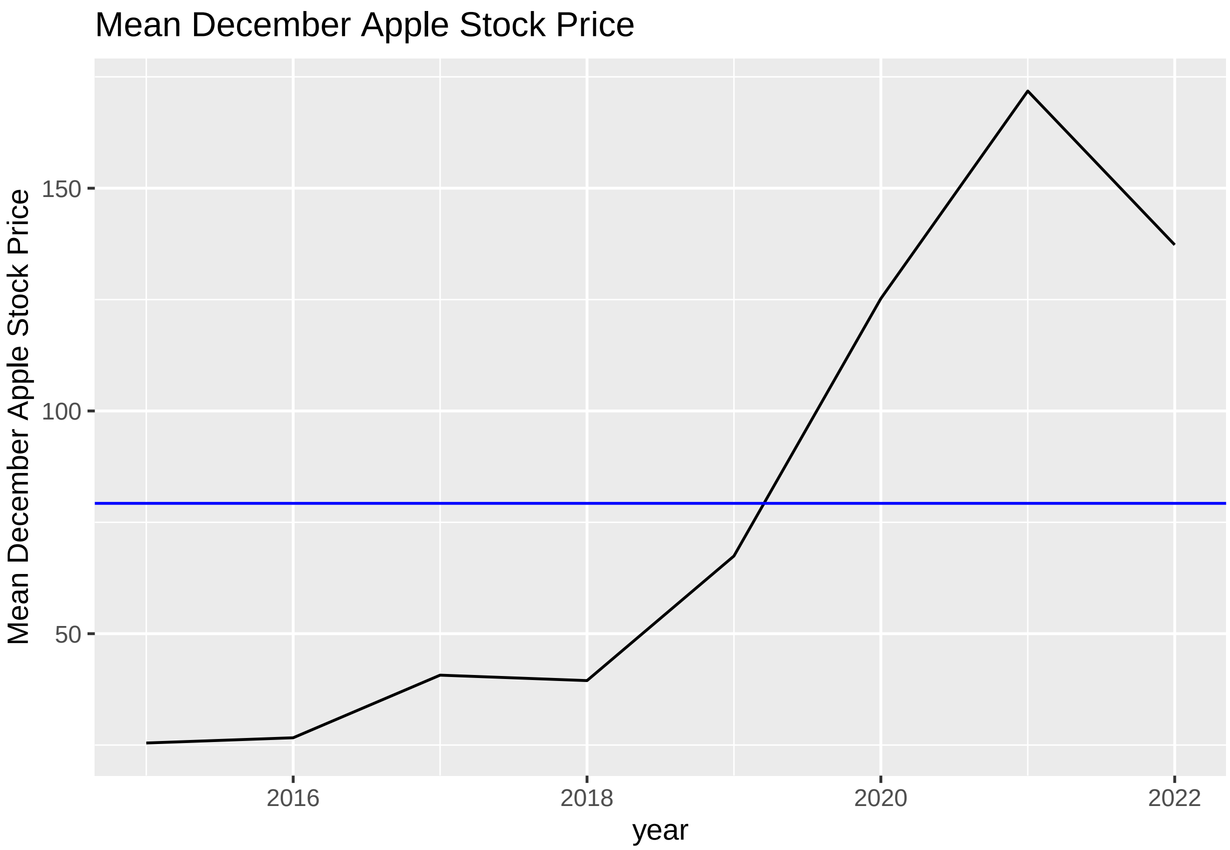

```r
oct_monthly_price <- monthly_price_avg_tsib %>%
  as_tibble() %>% select(mean_price, year, month) %>% filter(month==10)
oct_apple_price_plot <- oct_monthly_price %>%
  ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean October Apple Stock Price") +
  ggtitle('Mean October Apple Stock Price') +
  geom_hline(yintercept = mean(oct_monthly_price$mean_price), color="blue")
oct_apple_price_plot
```



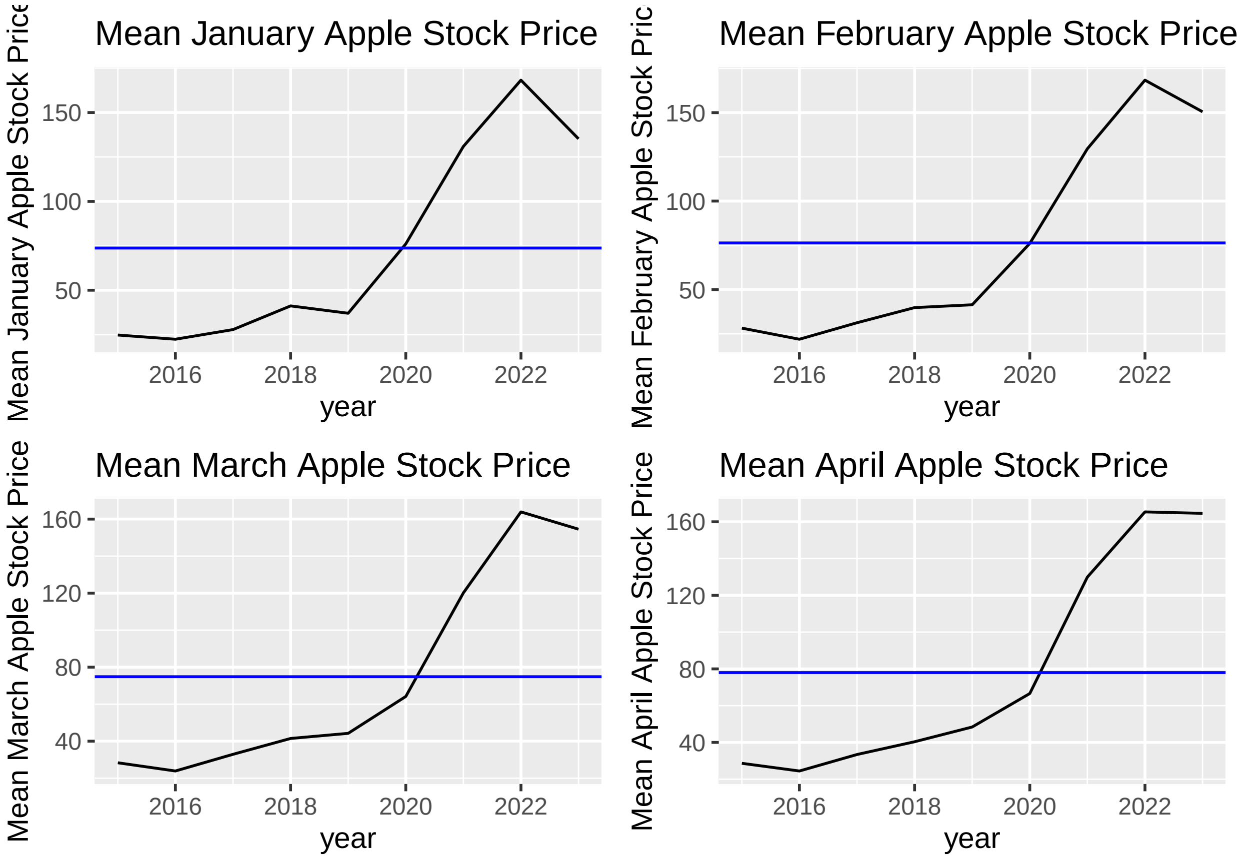

```
nov_monthly_price <- monthly_price_avg_tsib %>%
  as_tibble() %>% select(mean_price, year, month) %>% filter(month==11)
nov_apple_price_plot <- nov_monthly_price %>%
  ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean November Apple Stock Price") +
  ggtitle('Mean November Apple Stock Price') +
  geom_hline(yintercept = mean(nov_monthly_price$mean_price), color="blue")
nov_apple_price_plot
```




```r
dec_monthly_price <- monthly_price_avg_tsib %>%
  as_tibble() %>% select(mean_price, year, month) %>% filter(month==12)
dec_apple_price_plot <- dec_monthly_price %>%
  ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean December Apple Stock Price") +
  ggtitle('Mean December Apple Stock Price') +
  geom_hline(yintercept = mean(dec_monthly_price$mean_price), color="blue")
dec_apple_price_plot
```

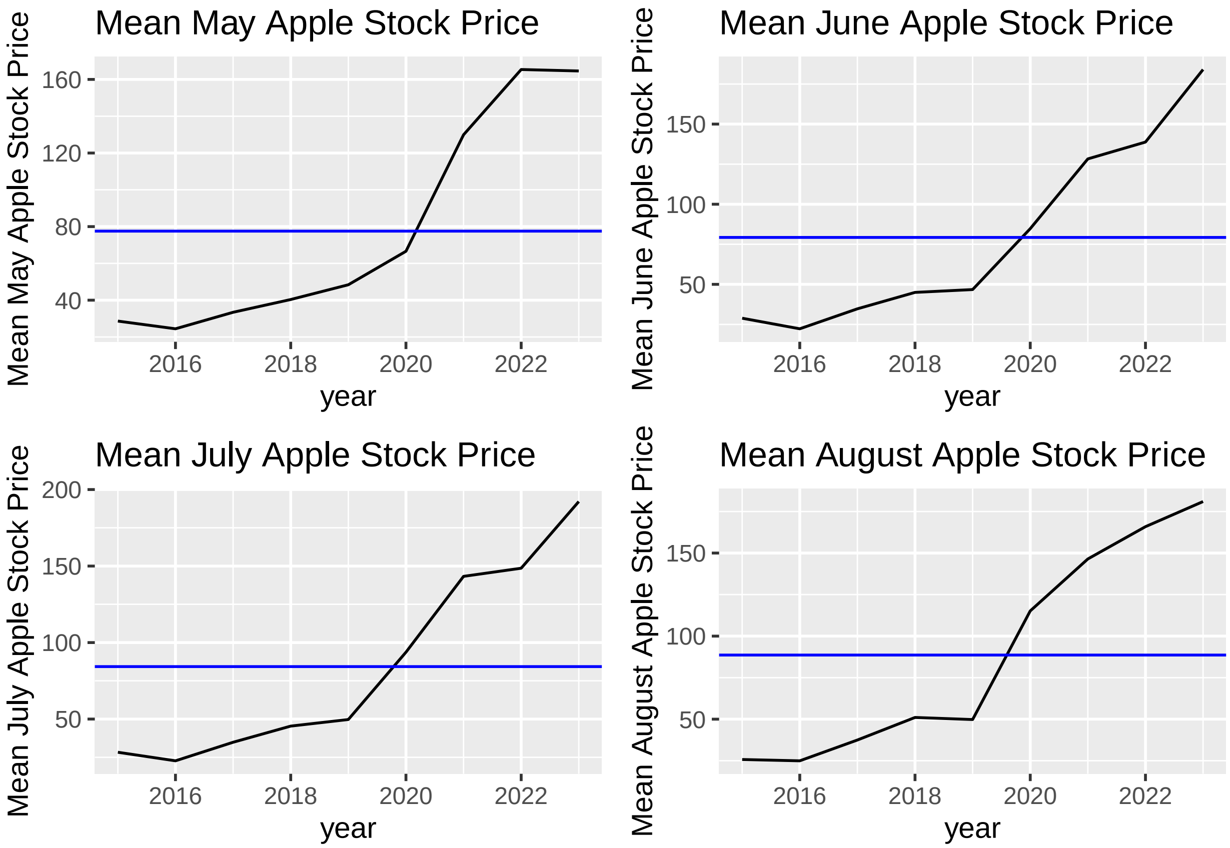



```
grid.arrange(jan_apple_price_plot, feb_apple_price_plot, mar_apple_price_plot, apr_apple_price_plot, nr
```



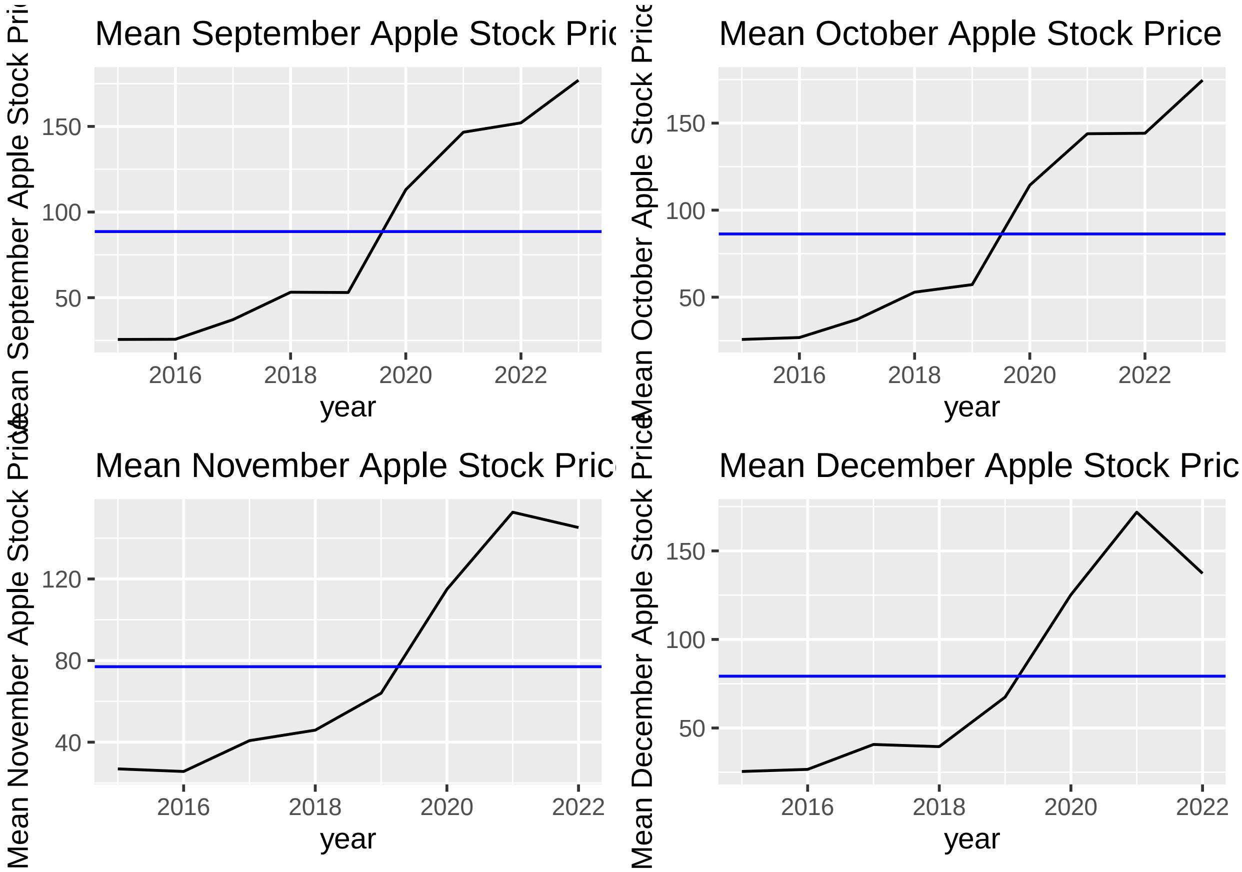
We see that average monthly Apple stock prices rise slightly from January to April.

```
grid.arrange(may_apple_price_plot, jun_apple_price_plot,
             jul_apple_price_plot, aug_apple_price_plot, nrow=2)
```



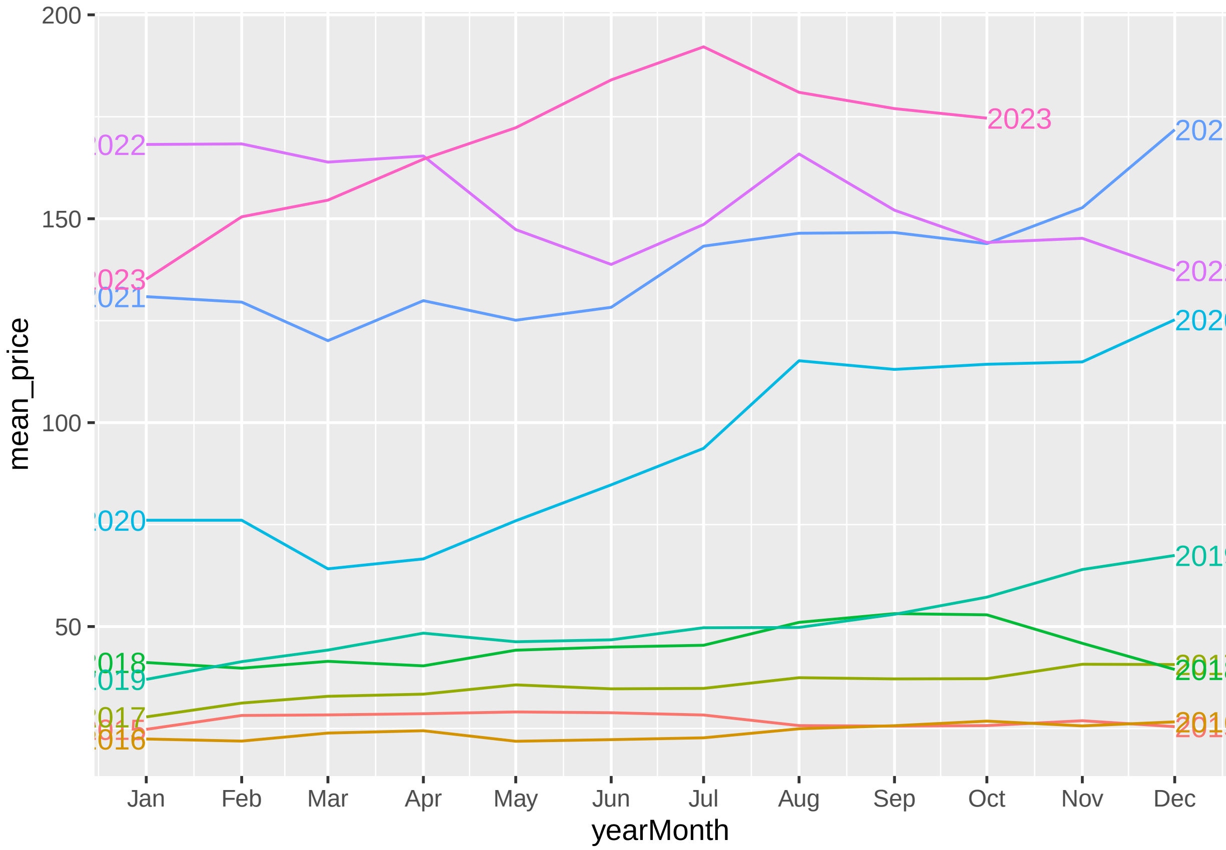
We see that average monthly Apple stock prices dip slightly from May to June but bump up in July and August.

```
grid.arrange(sep_apple_price_plot, oct_apple_price_plot,
             nov_apple_price_plot, dec_apple_price_plot, nrow=2)
```

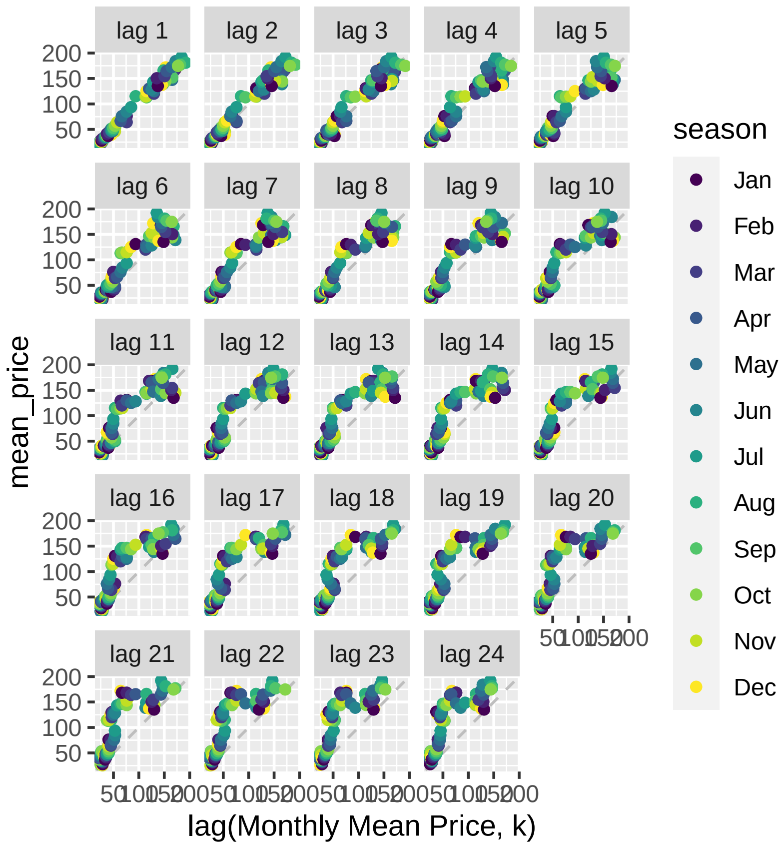


Average monthly Apple stock prices remain high relative to other months. Stock prices dip slightly in November and December.

```
monthly_price_avg_tsib |>
  gg_season(mean_price, labels = "both")
```



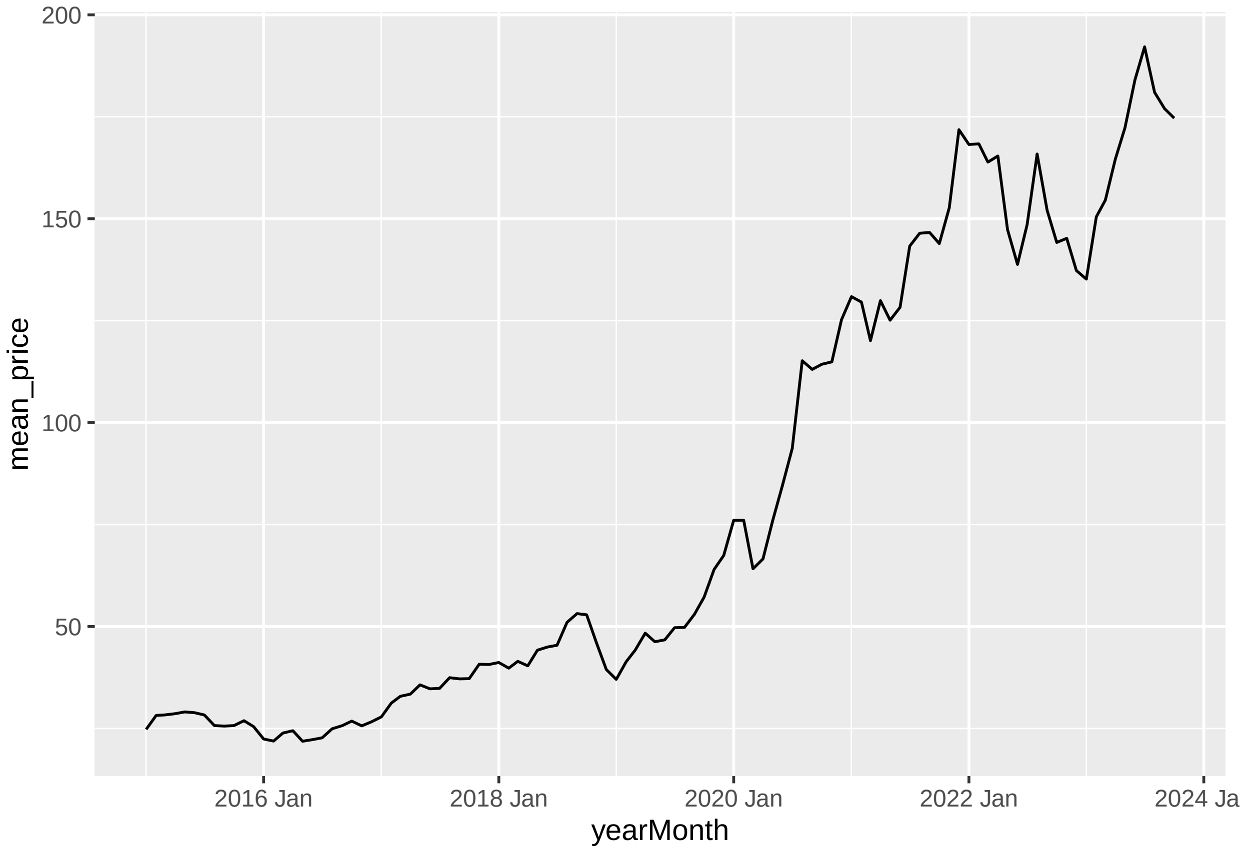

```
monthly_price_avg_tsib |>
  gg_lag(mean_price, lags=1:24, geom = "point") +
  labs(x = "lag(Monthly Mean Price, k)")
```




```r
monthly_price_avg_tsib
```

```
## # A tsibble: 106 x 4 [1M]
##     yearMonth mean_price  year month
##         <mth>      <dbl> <dbl> <dbl>
##  1  2015 Jan       24.8  2015     1
##  2  2015 Feb       28.2  2015     2
##  3  2015 Mar       28.3  2015     3
##  4  2015 Apr       28.6  2015     4
##  5  2015 May       29.1  2015     5
##  6  2015 Jun       28.9  2015     6
##  7  2015 Jul       28.3  2015     7
##  8  2015 Aug       25.7  2015     8
##  9  2015 Sep       25.6  2015     9
## 10  2015 Oct       25.7  2015    10
## # i 96 more rows
```

```r
monthly_price_avg_tsib |> ggplot(aes(x=yearMonth, y = mean_price)) +
  geom_line()
```




```r
apple_tsib
```

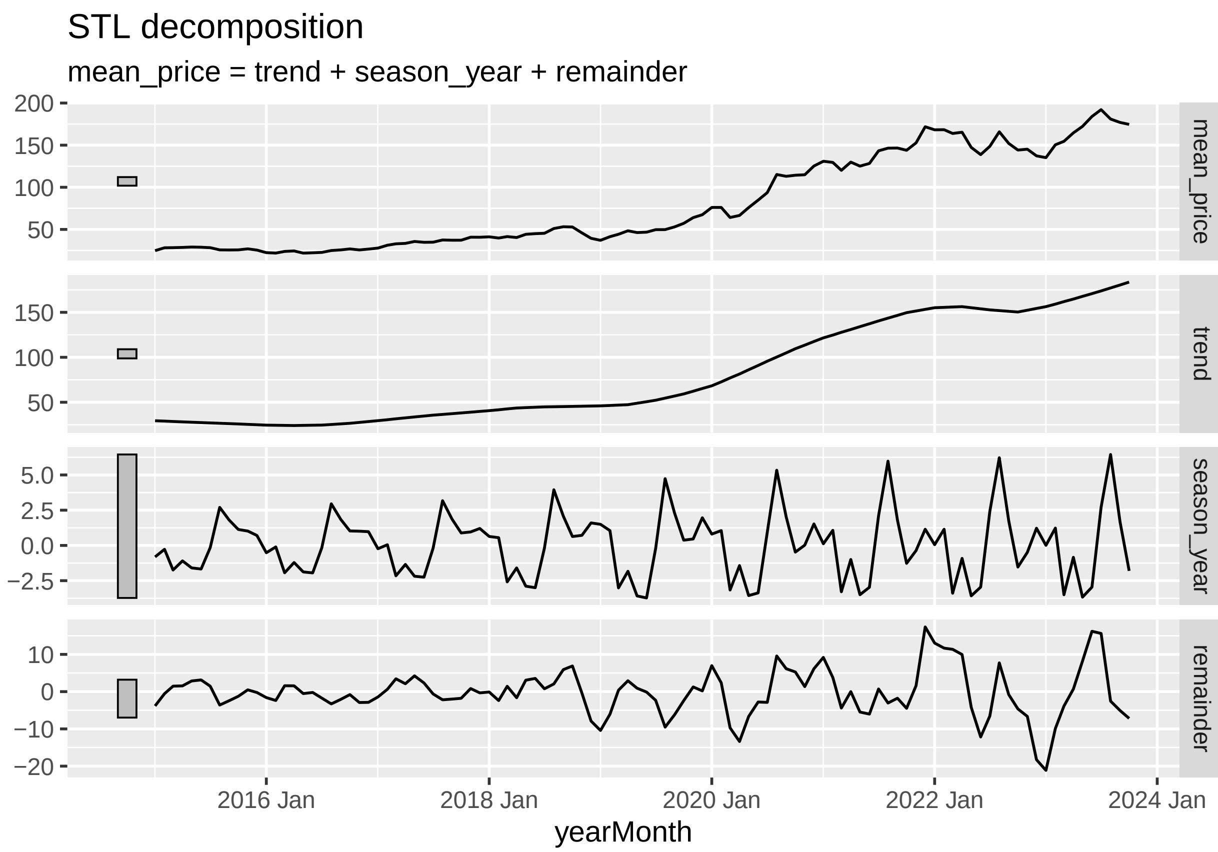
```
## # A tsibble: 2,223 x 10 [1D]
##    Date       close  open lowest highest total_vol mean_vol std_vol news   is_up
##    <date>     <dbl> <dbl>  <dbl>   <dbl>     <dbl>    <dbl>   <dbl> <chr>  <dbl>
##  1 2015-01-02  24.5  24.9   24.0    25.0 188181988  482518. 453959. "[\"I~     0
##  2 2015-01-05  23.8  24.2   23.6    24.4 200586492  514324. 426711. "[\"I~     0
##  3 2015-01-06  23.8  23.8   23.4    24.1 237766160  609657. 452107. "[\"I~     1
##  4 2015-01-07  24.1  24.0   23.9    24.3 137809632  353358. 315345. "Appl~     1
##  5 2015-01-08  25.1  24.4   24.3    25.2 201020076  515436. 344929. "Xiao~     1
##  6 2015-01-09  25.1  25.2   24.7    25.4 194014564  497473. 418949. "Info~     0
##  7 2015-01-12  24.5  25.1   24.4    25.3 177972644  456340. 357777. "Zoma~     1
##  8 2015-01-13  24.7  25.0   24.4    25.3 242520180  621847. 402626. "Zoma~     0
##  9 2015-01-14  24.6  24.5   24.3    24.8 166619520  427230. 308053. "['60~     0
## 10 2015-01-15  23.9  24.5   23.9    24.7 205693220  527419. 391931. "['60~     0
## # i 2,213 more rows
```

```r
dcmp <- monthly_price_avg_tsib |>
  model(stl = STL(mean_price))
components(dcmp)
```

```
## # A dable: 106 x 7 [1M]
## # Key:     .model [1]
## # :        mean_price = trend + season_year + remainder
##    .model yearMonth mean_price trend season_year remainder season_adjust
##    <chr>      <mth>      <dbl> <dbl>       <dbl>     <dbl>         <dbl>
##  1 stl     2015 Jan       24.8  29.4      -0.814    -3.83           25.6
##  2 stl     2015 Feb       28.2  29.0      -0.275    -0.549          28.5
```

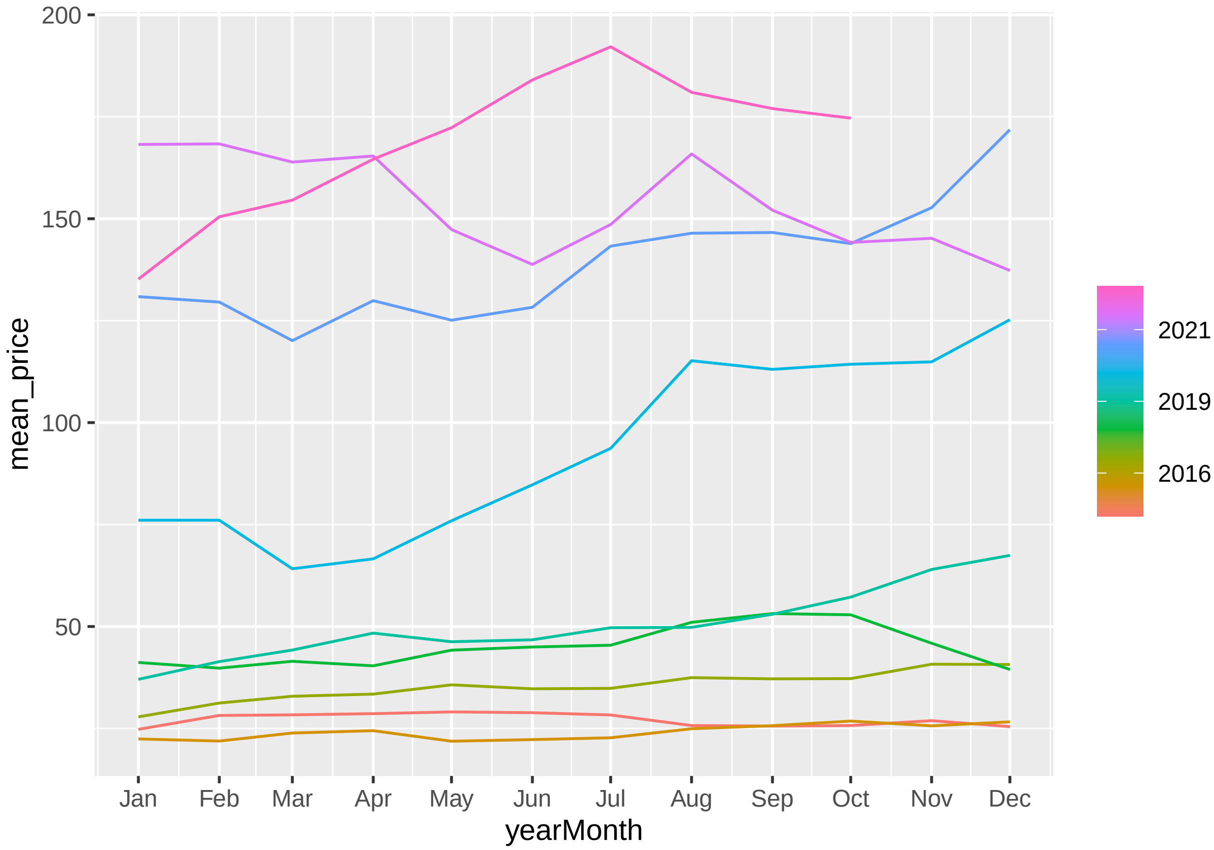
```
## 3 stl      2015 Mar       28.3  28.6       -1.74      1.48            30.1
## 4 stl      2015 Apr       28.6  28.2       -1.10      1.54            29.7
## 5 stl      2015 May       29.1  27.8       -1.59      2.87            30.7
## 6 stl      2015 Jun       28.9  27.4       -1.67      3.15            30.5
## 7 stl      2015 Jul       28.3  27.0       -0.148     1.46            28.5
## 8 stl      2015 Aug       25.7  26.6        2.70     -3.60            23.0
## 9 stl      2015 Sep       25.6  26.2        1.80     -2.43            23.8
## 10 stl     2015 Oct       25.7  25.8        1.13     -1.25            24.6
## # i 96 more rows
```

```r
components(dcmp) |> autoplot()
```



There seems to be a peak in monthly average stock prices every middle of every year (around June), with peaks increasing and troughs decreasing (increasing variability) as we move from left to right.

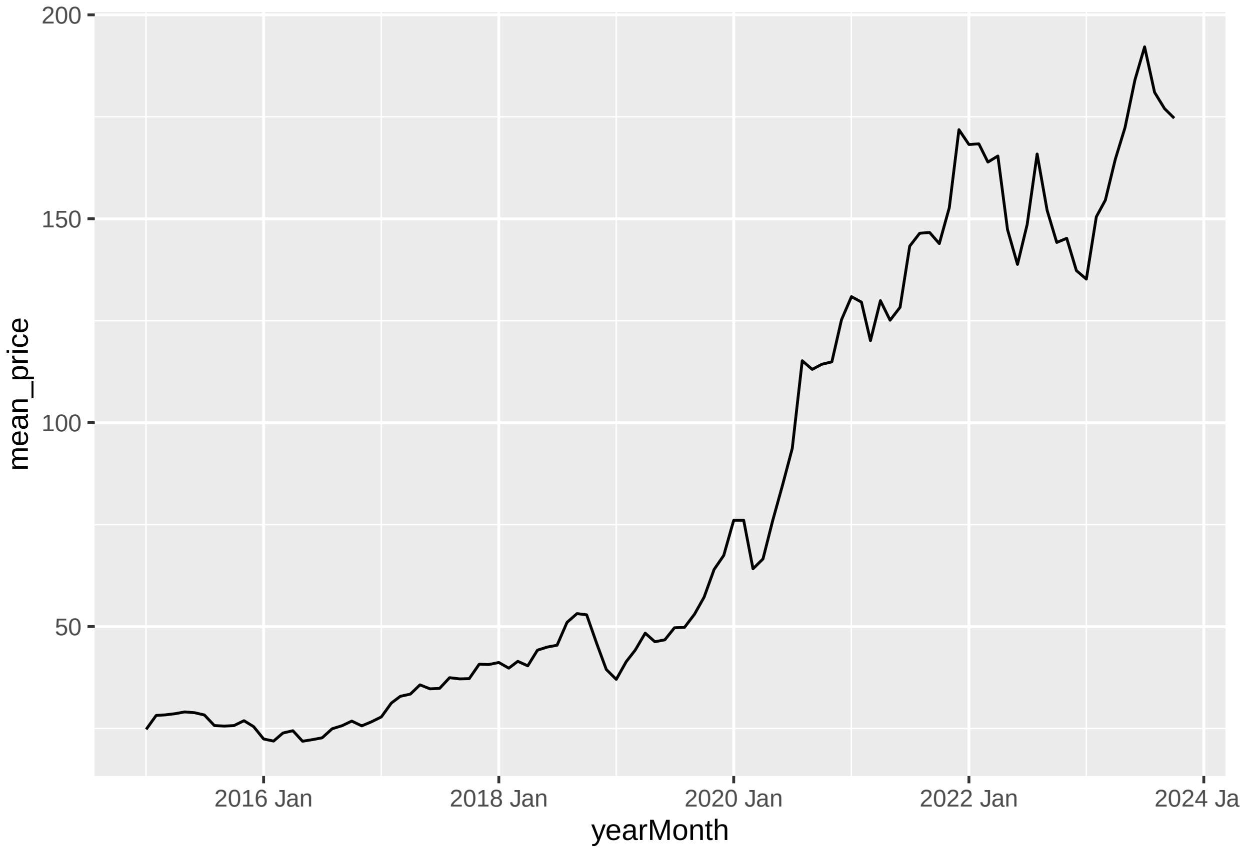
```r
monthly_price_avg_tsib |>
  gg_season(mean_price, period = "year")
```




```
monthly_price_avg_tsib
```

```
## # A tsibble: 106 x 4 [1M]
##    yearMonth mean_price  year month
##        <mth>      <dbl> <dbl> <dbl>
##  1  2015 Jan       24.8  2015     1
##  2  2015 Feb       28.2  2015     2
##  3  2015 Mar       28.3  2015     3
##  4  2015 Apr       28.6  2015     4
##  5  2015 May       29.1  2015     5
##  6  2015 Jun       28.9  2015     6
##  7  2015 Jul       28.3  2015     7
##  8  2015 Aug       25.7  2015     8
##  9  2015 Sep       25.6  2015     9
## 10  2015 Oct       25.7  2015    10
## # i 96 more rows
```

```
monthly_price_avg_tsib |> ggplot(aes(x=yearMonth, y = mean_price)) +
  geom_line()
```

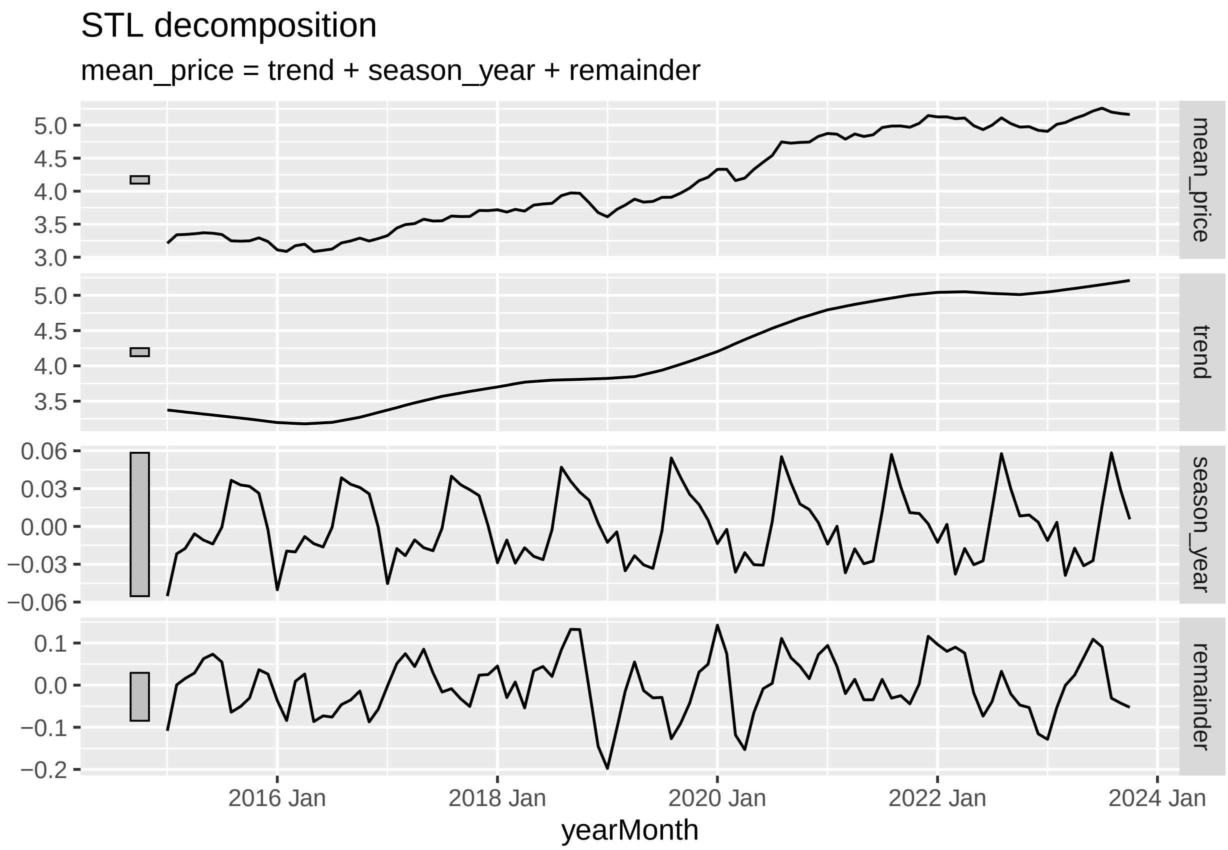


To stabilize the variance, we apply a Box-Cox transformation (log).

```r
monthly_price_avg_tsib_transformed <- monthly_price_avg_tsib %>%
  mutate(mean_price = log(mean_price))
monthly_price_avg_tsib_transformed <- monthly_price_avg_tsib_transformed %>%
  select(mean_price)

dcmp <- monthly_price_avg_tsib_transformed |>
  model(stl = STL(mean_price))
components(dcmp)
```

```
## # A dable: 106 x 7 [1M]
## # Key:     .model [1]
## # :        mean_price = trend + season_year + remainder
##    .model yearMonth mean_price trend season_year remainder season_adjust
##    <chr>      <mth>      <dbl> <dbl>       <dbl>     <dbl>         <dbl>
##  1 stl     2015 Jan       3.21  3.37   -0.0554   -0.108             3.27
##  2 stl     2015 Feb       3.34  3.36   -0.0217    0.000806          3.36
##  3 stl     2015 Mar       3.34  3.35   -0.0176    0.0158            3.36
##  4 stl     2015 Apr       3.35  3.33   -0.00584   0.0288            3.36
##  5 stl     2015 May       3.37  3.32   -0.0109    0.0628            3.38
##  6 stl     2015 Jun       3.36  3.30   -0.0139    0.0735            3.38
##  7 stl     2015 Jul       3.34  3.29   -0.000593  0.0549            3.34
##  8 stl     2015 Aug       3.25  3.27    0.0366   -0.0642            3.21
##  9 stl     2015 Sep       3.24  3.26    0.0329   -0.0507            3.21
## 10 stl     2015 Oct       3.25  3.25    0.0318   -0.0304            3.22
## # i 96 more rows
```

```r
components(dcmp) |> autoplot()
```



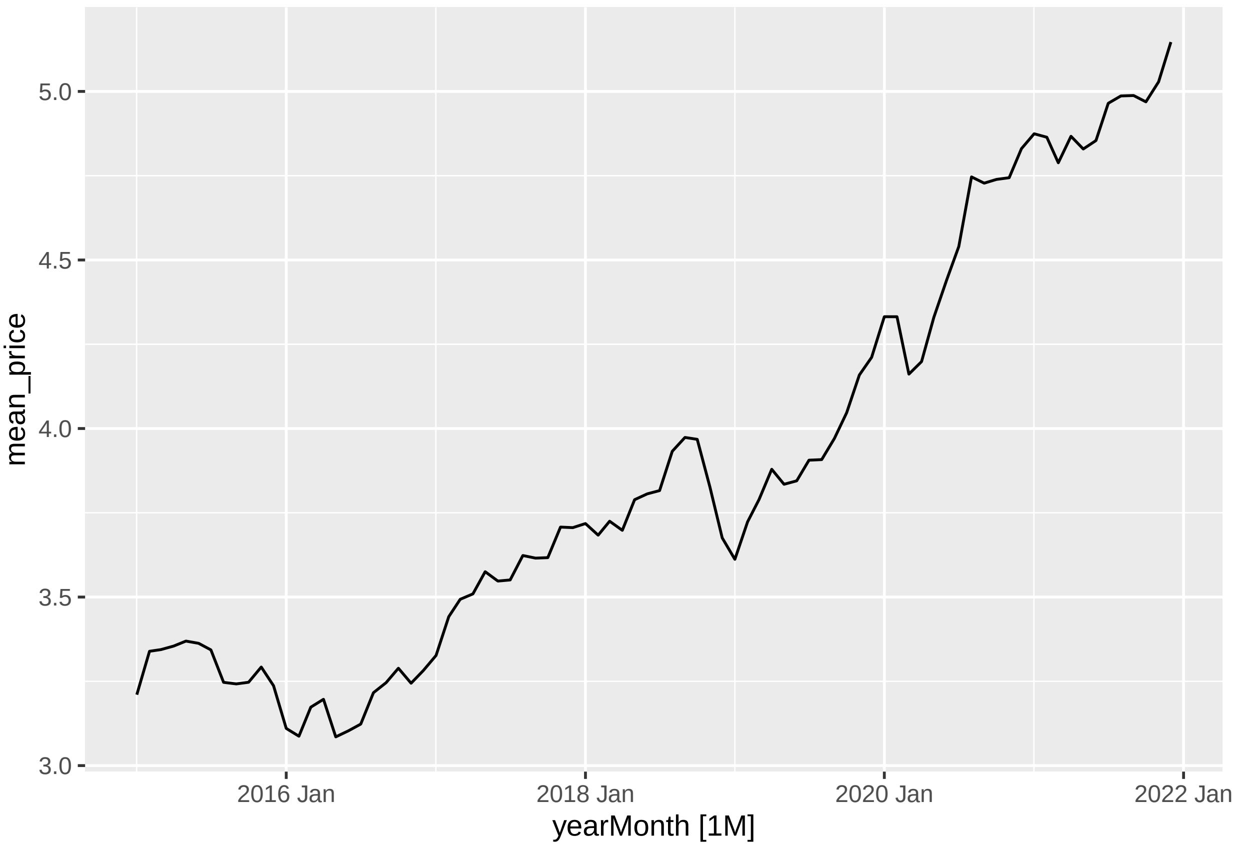

```r
monthly_price_avg_tsib_transformed$is_up <- append(diff(
  monthly_price_avg_tsib_transformed$mean_price) > 0, FALSE)
monthly_price_avg_tsib_transformed <- monthly_price_avg_tsib_transformed %>%
  mutate(is_up = case_when(is_up == TRUE ~ 1, TRUE ~ 0))


monthly_price_avg_tsib_train <- subset(monthly_price_avg_tsib_transformed,
                                       yearMonth <
                                         yearmonth(as.Date("2022-01-01")))
monthly_price_avg_tsib_test <- subset(monthly_price_avg_tsib_transformed,
                                      yearMonth >=
                                        yearmonth(as.Date("2022-01-01")))
```

```r
monthly_price_avg_tsib_train
```

```
## # A tsibble: 84 x 3 [1M]
##    mean_price yearMonth is_up
##         <dbl>     <mth> <dbl>
##  1       3.21  2015 Jan     1
##  2       3.34  2015 Feb     1
##  3       3.34  2015 Mar     1
##  4       3.35  2015 Apr     1
##  5       3.37  2015 May     0
##  6       3.36  2015 Jun     0
##  7       3.34  2015 Jul     0
```

```
## 8        3.25  2015 Aug     0
## 9        3.24  2015 Sep     1
## 10       3.25  2015 Oct     1
## # i 74 more rows
```

```
monthly_price_avg_tsib_train %>% autoplot(mean_price)
```

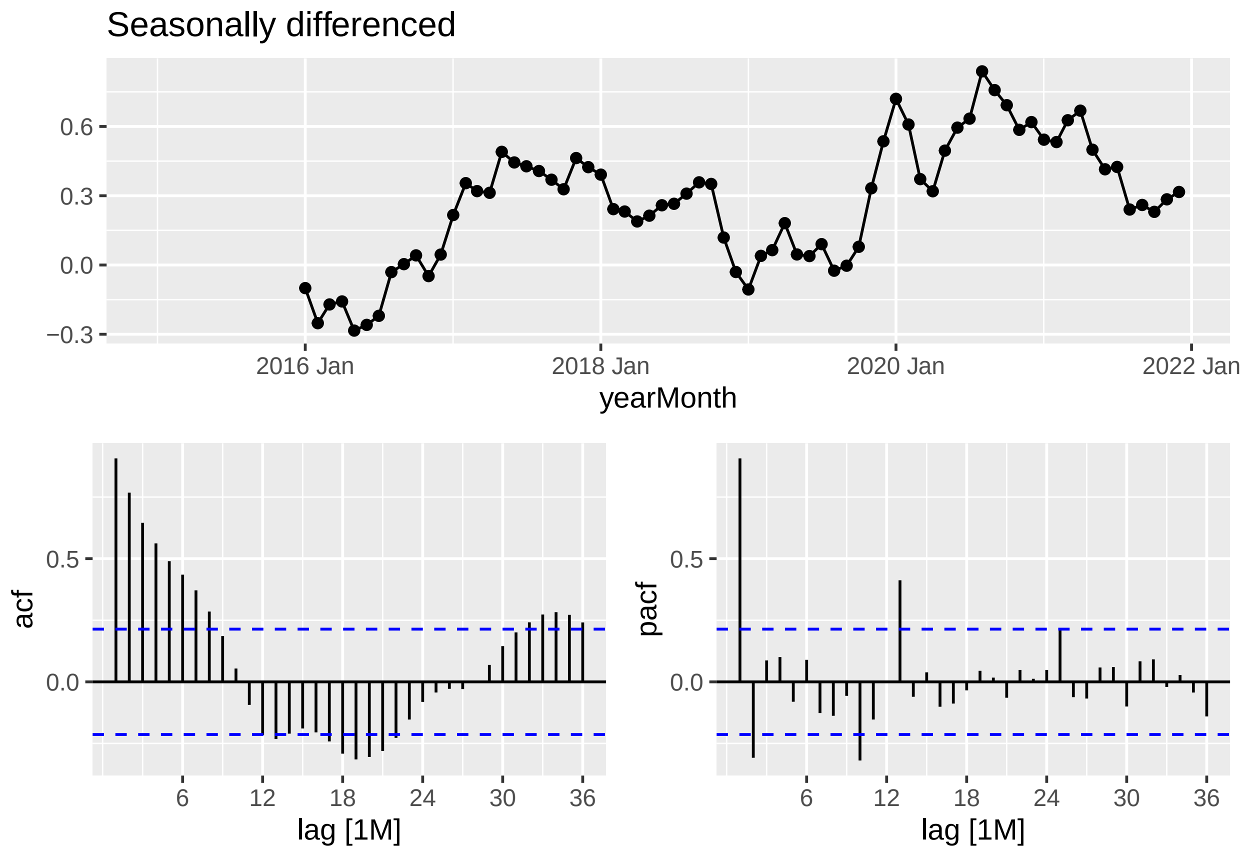


Due to the observed non-stationarity and seasonality with period 12, we take a seasonal difference.

```
monthly_price_avg_tsib_train |>
  gg_tsdisplay(difference(mean_price, 12),
               plot_type='partial', lag=36) +
  labs(title="Seasonally differenced", y="")
```

```
## Warning: Removed 12 rows containing missing values (`geom_line()`).
```

```
## Warning: Removed 12 rows containing missing values (`geom_point()`).
```



The series is still non-stationary, and the formal KPSS test below confirms this.

```
monthly_price_avg_tsib_train %>%
  mutate(diff_log = difference(mean_price, 12)) %>%
  features(diff_log, unitroot_kpss)
```

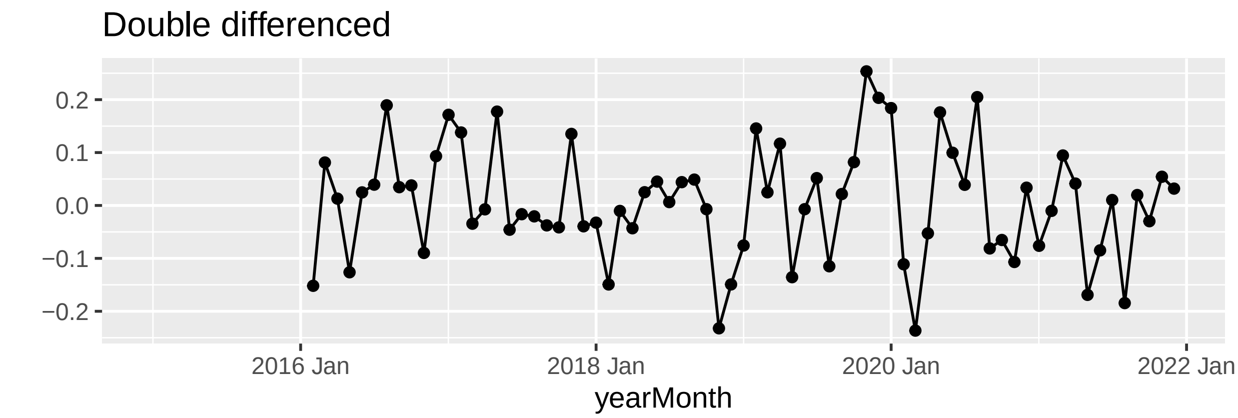
```
## # A tibble: 1 x 2
##   kpss_stat kpss_pvalue
##       <dbl>       <dbl>
## 1     0.780        0.01
```

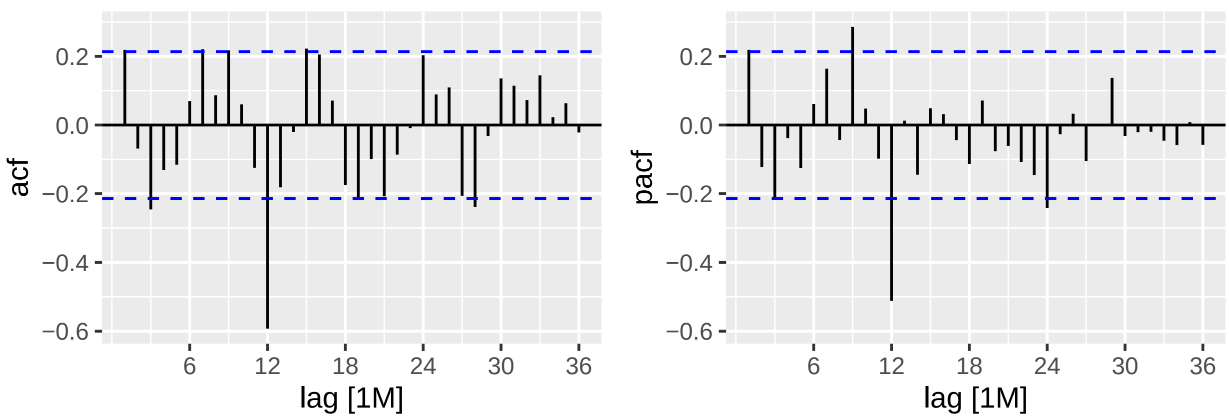
We take a further first difference.

```
monthly_price_avg_tsib_train |>
  gg_tsdisplay(difference(mean_price, 12) |> difference(),
               plot_type='partial', lag=36) +
  labs(title = "Double differenced", y="")
```

```
## Warning: Removed 13 rows containing missing values (`geom_line()`).
```

```
## Warning: Removed 13 rows containing missing values (`geom_point()`).
```





We can see now that the data are closer to stationary, despite a one or two significant lags.

A formal test of stationarity using KPSS unit root test is applied below, and the insignificant result implies that we do not reject the null hypothesis that the differenced series is stationary. Hence, the series is stationary.

```
monthly_price_avg_tsib_train %>% mutate(diff_log =
                                          difference(mean_price, 12)) %>%
  mutate(diff_log = difference(diff_log)) %>% features(diff_log, unitroot_kpss)
```

```
## # A tibble: 1 x 2
##   kpss_stat kpss_pvalue
##       <dbl>       <dbl>
## 1    0.0816         0.1
```

Models to use: NAIVE, ARIMA, SARIMA

We'll start off with a simple forecasting method - the NAIVE method. Using this method, we set the forecasts to be the value of the last observation, a method that works surprisingly well in economics and finance. We'll also add ARIMA models with a range of parameters. The range covers reasonable values we would expect based on our exploratory data analysis of the ACF, PACF, and STL decomposition plots. Note that the SARIMA models result from ARIMA() calls that use the PDQ() function. ARIMA() calls without PDQ() but still use pdq() reduce to ordinary ARIMA models.

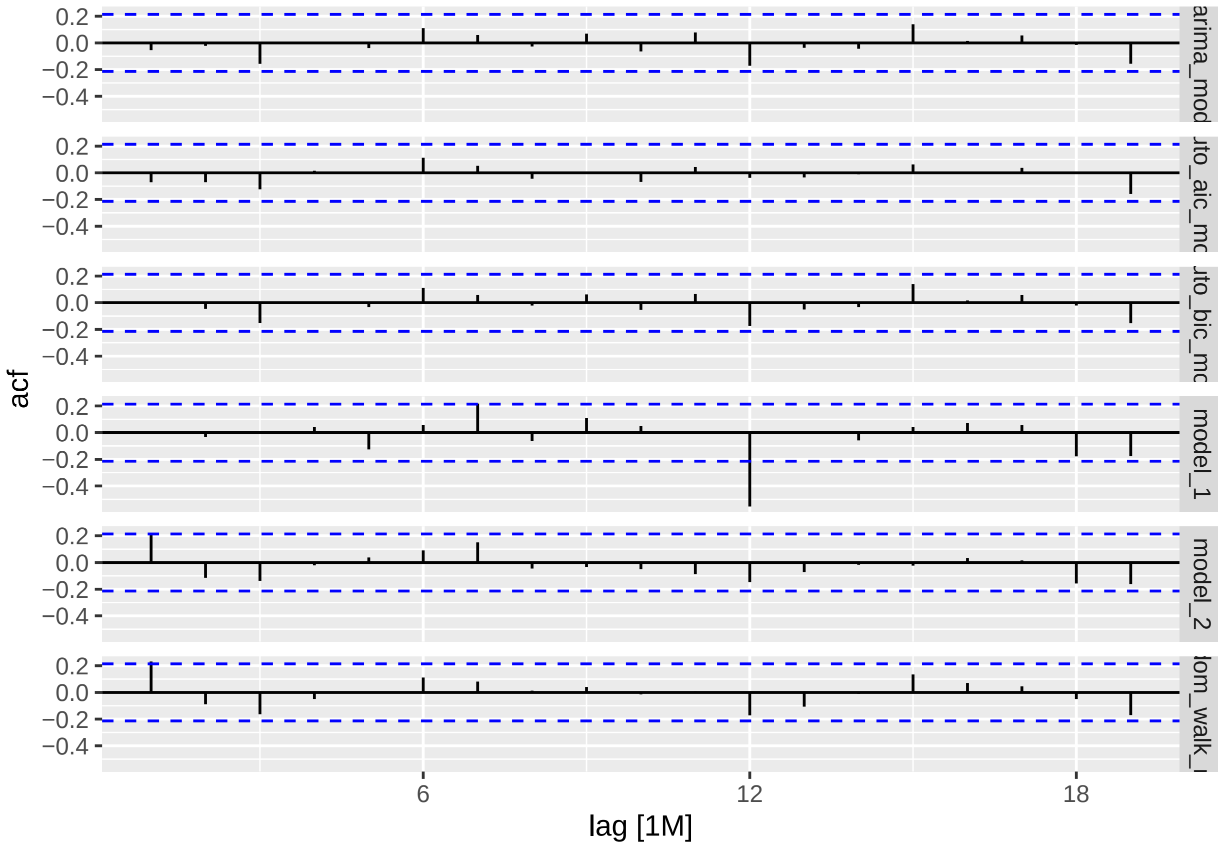
```
install.packages('forecast')
```

```
## Installing package into '/usr/local/lib/R/site-library'
## (as 'lib' is unspecified)
```

```r
library(forecast)
library(fable)

model_comp <- monthly_price_avg_tsib_train %>%
  model(model_1 = ARIMA(mean_price ~ 0 + pdq(3, 1, 0) + PDQ(0, 1, 0)),
        model_2 = ARIMA(mean_price ~ 0 + pdq(0,1,0) + PDQ(1, 1, 0)),
        auto_aic_mod = ARIMA(mean_price ~ 0 + pdq(1:10, 1:2, 1:10) +
                                          PDQ(1:2,0:1,0), ic="aic",
                                          stepwise=F, greedy=F),
        auto_bic_mod = ARIMA(mean_price ~ pdq(0:10, 1:2, 0:10) +
                                          PDQ(0:2,0:1,0), ic="bic",
                                          stepwise=F, greedy=F),
        arima_mod = ARIMA(mean_price ~ 0 + pdq(0:3,1, 0:3)),
        random_walk_mod = NAIVE(mean_price)
        )
model_comp
```

```
## # A mable: 1 x 6
##                      model_1                     model_2              auto_aic_mod
##                      <model>                     <model>                   <model>
## 1 <ARIMA(3,1,0)(0,1,0)[12]> <ARIMA(0,1,0)(1,1,0)[12]> <ARIMA(1,1,1)(2,0,0)[12]>
## # i 3 more variables: auto_bic_mod <model>, arima_mod <model>,
## #   random_walk_mod <model>
```

```r
model_comp %>%
  augment() %>%
  ACF(.resid) %>%
  autoplot()
```




```
model_comp %>%
  augment() %>%
  filter(.model == "model_1") %>%
  select(.resid) %>%
  as.ts() %>%
  Box.test(., lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  .
## X-squared = 8.212, df = 10, p-value = 0.6081
```

```
model_comp %>%
  augment() %>%
  filter(.model == "model_2") %>%
  select(.resid) %>%
  as.ts() %>%
  Box.test(., lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  .
## X-squared = 10.274, df = 10, p-value = 0.4168
```

```
model_comp %>%
  augment() %>%
```

```r
  filter(.model == "auto_aic_mod") %>%
  select(.resid) %>%
  as.ts() %>%
  Box.test(., lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  .
## X-squared = 4.3618, df = 10, p-value = 0.9296
```

```r
model_comp %>%
  augment() %>%
  filter(.model == "auto_bic_mod") %>%
  select(.resid) %>%
  as.ts() %>%
  Box.test(., lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  .
## X-squared = 4.5263, df = 10, p-value = 0.9205
```

```r
model_comp %>%
  augment() %>%
  filter(.model == "random_walk_mod") %>%
  select(.resid) %>%
  as.ts() %>%
  Box.test(., lag=10, type="Ljung-Box")
```

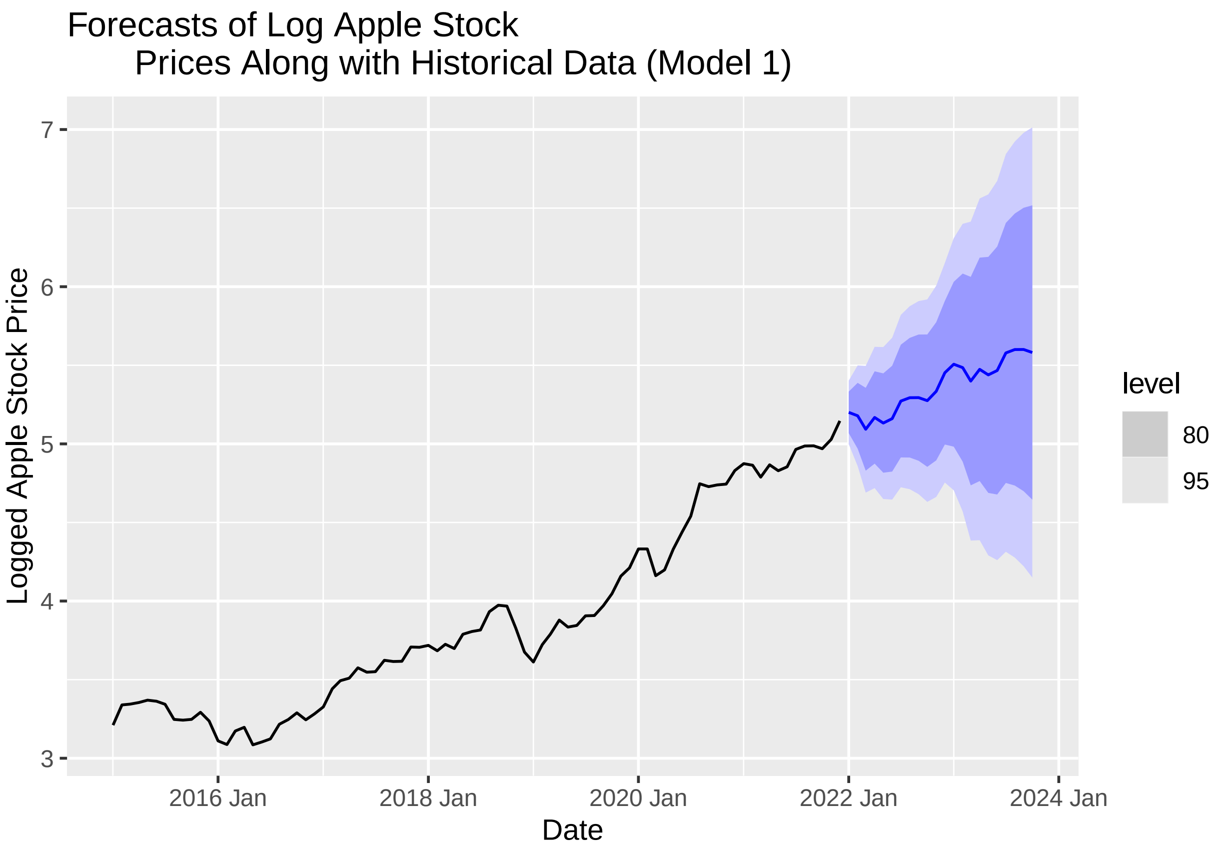
```
##
##  Box-Ljung test
##
## data:  .
## X-squared = 9.8324, df = 10, p-value = 0.4553
```

```r
new_apple_stock <- new_data(monthly_price_avg_tsib_train, 22)
new_apple_stock
```

```
## # A tsibble: 22 x 1 [1M]
##    yearMonth
##        <mth>
##  1  2022 Jan
##  2  2022 Feb
##  3  2022 Mar
##  4  2022 Apr
##  5  2022 May
##  6  2022 Jun
##  7  2022 Jul
##  8  2022 Aug
##  9  2022 Sep
## 10  2022 Oct
## # i 12 more rows
```

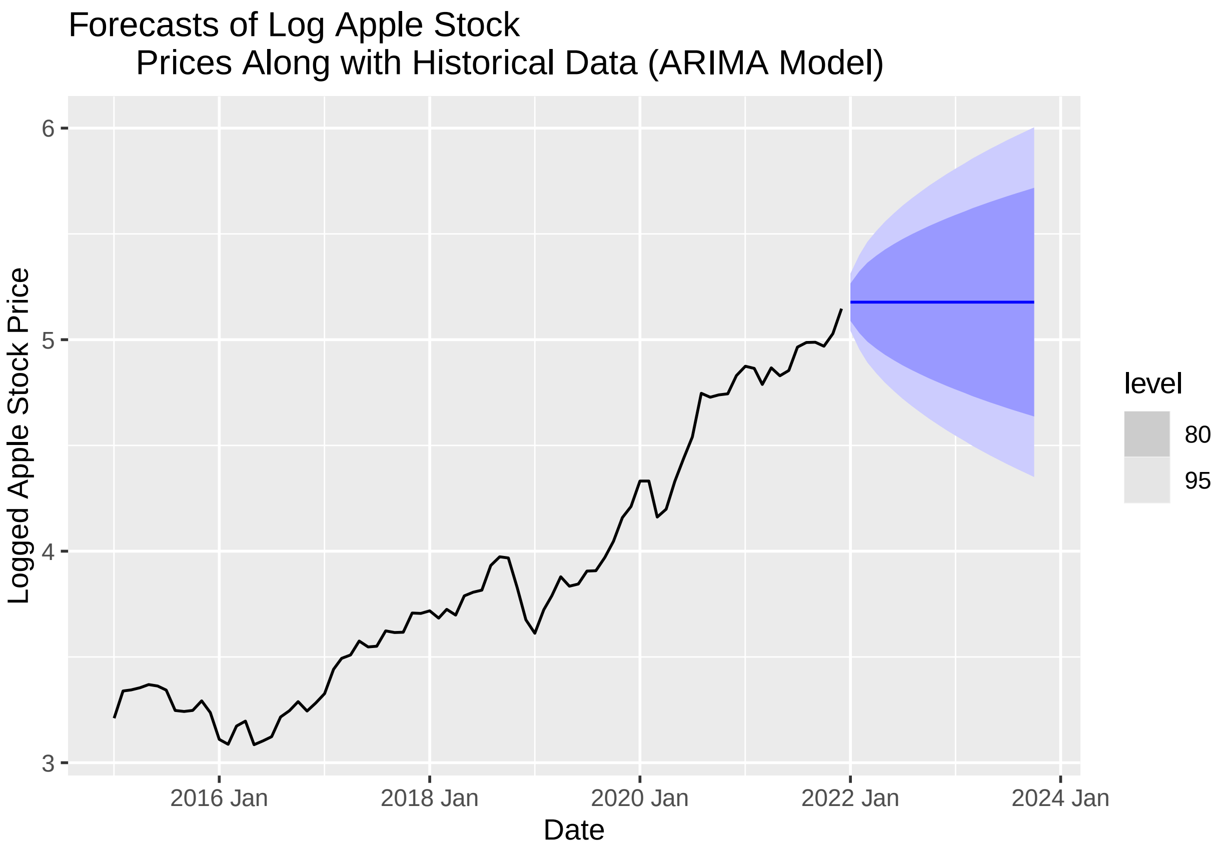
```
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "model_1") %>% autoplot(monthly_price_avg_tsib_train) +
  labs(x = "Date", y = "Logged Apple Stock Price",
       title = "Forecasts of Log Apple Stock
       Prices Along with Historical Data (Model 1)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```




```
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "arima_mod") %>% autoplot(monthly_price_avg_tsib_train) +
  labs(x = "Date", y = "Logged Apple Stock Price",
       title = "Forecasts of Log Apple Stock
       Prices Along with Historical Data (ARIMA Model)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```




```
reference_data <- monthly_price_avg_tsib_train %>% mutate(mean_price = exp(mean_price))
reference_data
```

```
## # A tsibble: 84 x 3 [1M]
##    mean_price yearMonth is_up
##         <dbl>     <mth> <dbl>
##  1       24.8  2015 Jan     1
##  2       28.2  2015 Feb     1
##  3       28.3  2015 Mar     1
##  4       28.6  2015 Apr     1
##  5       29.1  2015 May     0
##  6       28.9  2015 Jun     0
##  7       28.3  2015 Jul     0
##  8       25.7  2015 Aug     0
##  9       25.6  2015 Sep     1
## 10       25.7  2015 Oct     1
## # i 74 more rows
```

```
reference_years <- reference_data %>% select(yearMonth)
reference_years
```

```
## # A tsibble: 84 x 1 [1M]
##    yearMonth
##        <mth>
##  1  2015 Jan
##  2  2015 Feb
##  3  2015 Mar
##  4  2015 Apr
```

```
## 5  2015 May
## 6  2015 Jun
## 7  2015 Jul
## 8  2015 Aug
## 9  2015 Sep
## 10 2015 Oct
## # i 74 more rows
```

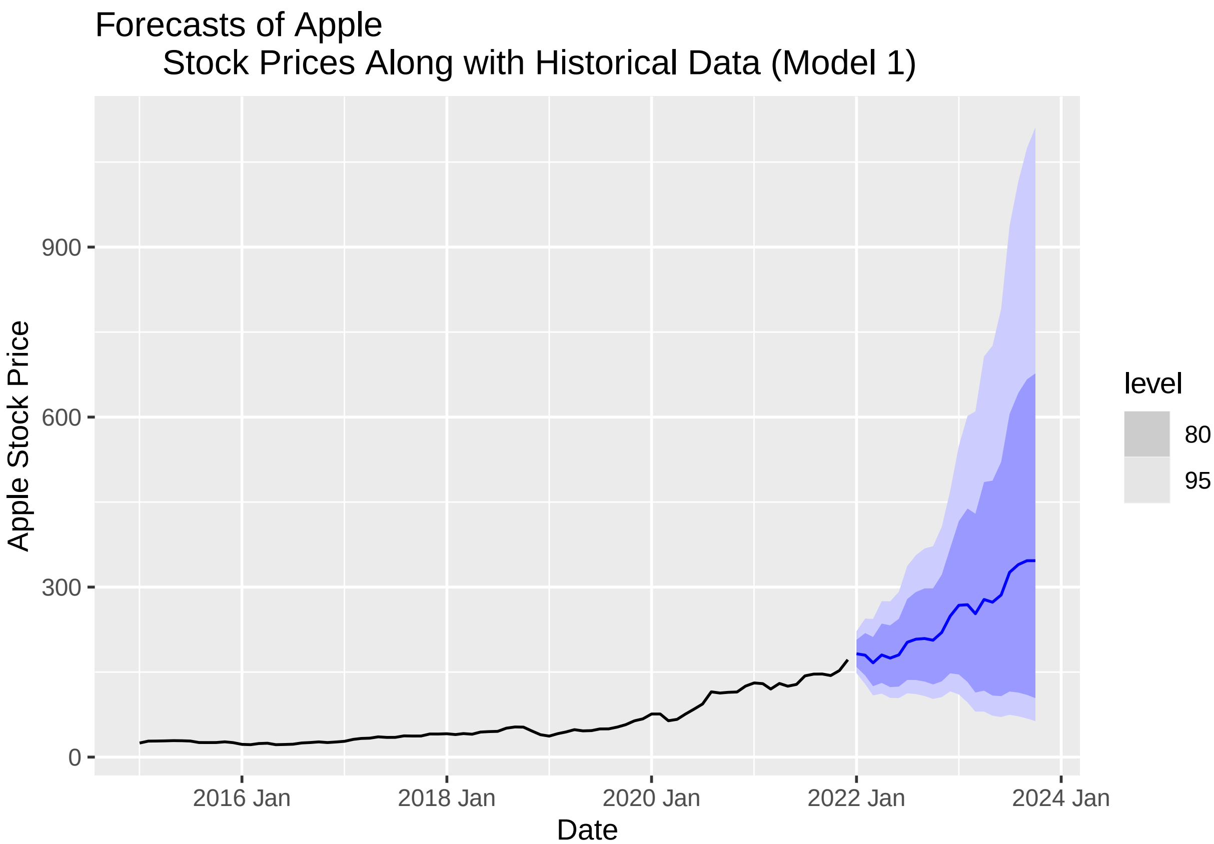
```
new_apple_stock
```

```
## # A tsibble: 22 x 1 [1M]
##    yearMonth
##        <mth>
##  1  2022 Jan
##  2  2022 Feb
##  3  2022 Mar
##  4  2022 Apr
##  5  2022 May
##  6  2022 Jun
##  7  2022 Jul
##  8  2022 Aug
##  9  2022 Sep
## 10  2022 Oct
## # i 12 more rows
```

We need to use fable object before autoplot when making a plots of forecasts along with historical data. When we transform log values back to their original values, we must also back-transform the probability distribution that is created in the fable object, as that probability distribution is centered around the log transformed forecast value.

```
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "model_1") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price)) %>%
  autoplot(reference_data) + labs(x = "Date",
                                  y = "Apple Stock Price",
      title = "Forecasts of Apple
      Stock Prices Along with Historical Data (Model 1)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```




```
model_comp
```

```
## # A mable: 1 x 6
##                    model_1                   model_2              auto_aic_mod
##                    <model>                   <model>                   <model>
## 1 <ARIMA(3,1,0)(0,1,0)[12]> <ARIMA(0,1,0)(1,1,0)[12]> <ARIMA(1,1,1)(2,0,0)[12]>
## # i 3 more variables: auto_bic_mod <model>, arima_mod <model>,
## #   random_walk_mod <model>
```

```
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "model_1") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price))
```

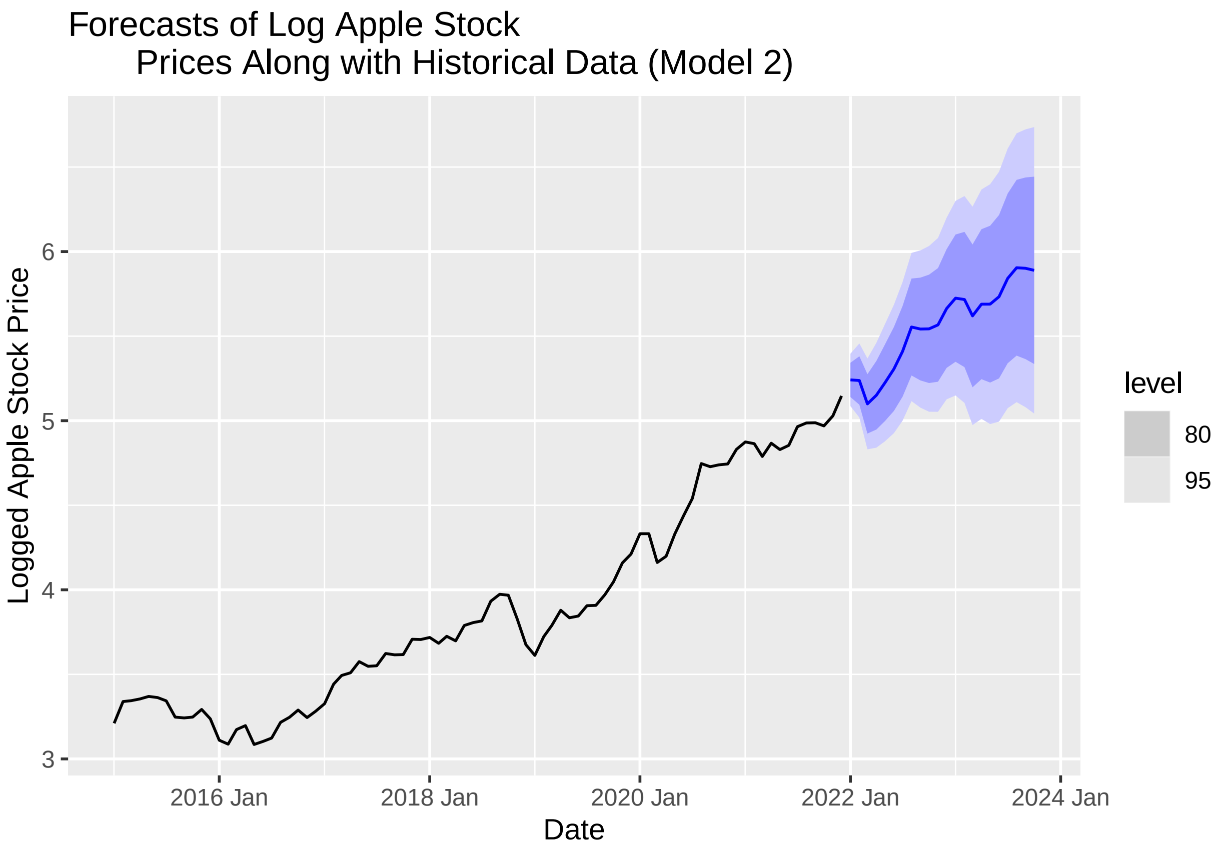
```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```

```
## # A fable: 22 x 4 [1M]
## # Key:     .model [1]
##    .model  yearMonth     mean_price .mean
##    <chr>       <mth>         <dist> <dbl>
##  1 model_1  2022 Jan lN(5.2, 0.011)  181.
##  2 model_1  2022 Feb lN(5.2, 0.027)  177.
##  3 model_1  2022 Mar lN(5.1, 0.042)  163.
##  4 model_1  2022 Apr lN(5.2, 0.053)  176.
##  5 model_1  2022 May lN(5.1, 0.061)  169.
##  6 model_1  2022 Jun lN(5.2, 0.069)  174.
##  7 model_1  2022 Jul lN(5.3, 0.078)  195.
##  8 model_1  2022 Aug lN(5.3, 0.088)  199.
##  9 model_1  2022 Sep lN(5.3, 0.098)  199.
```

```
## 10 model_1  2022 Oct  lN(5.3, 0.11)  195.
## # i 12 more rows
```

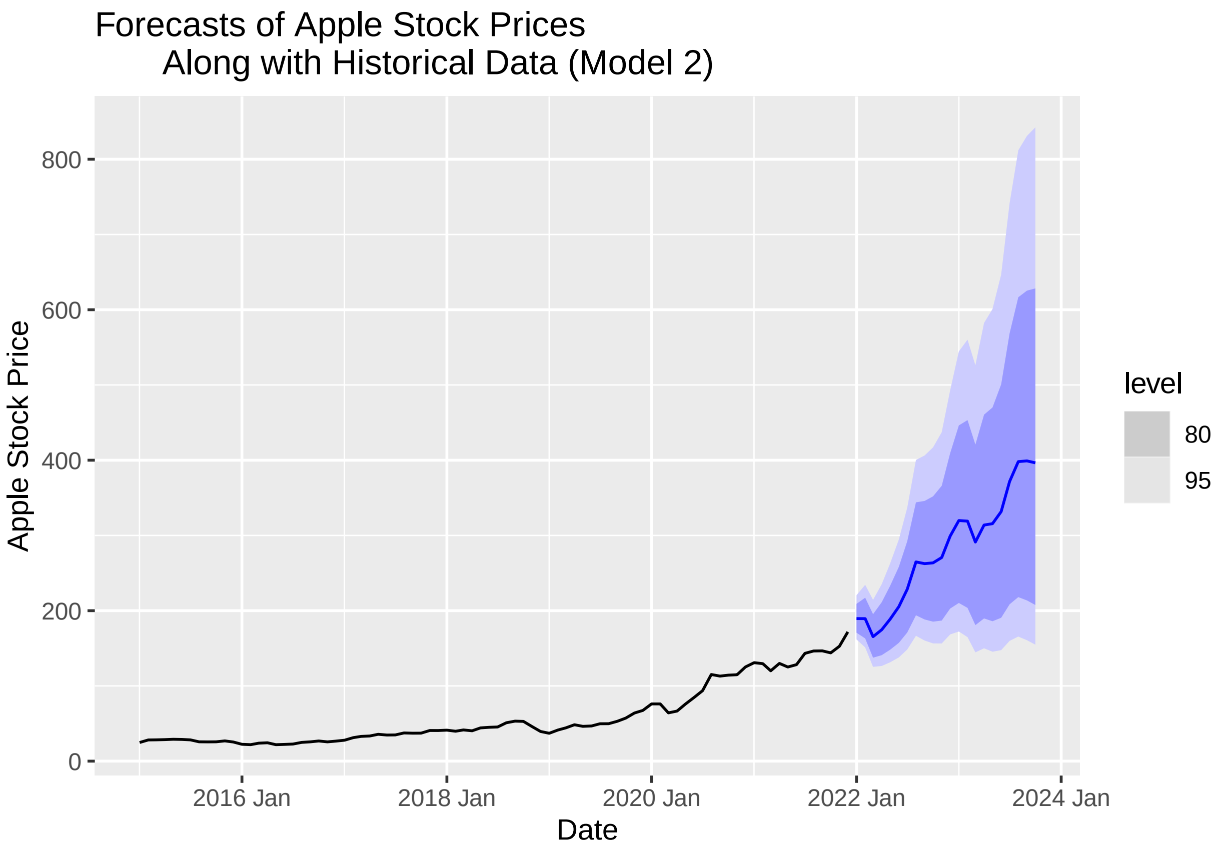
```
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "model_2") %>%
  autoplot(monthly_price_avg_tsib_train) +
  labs(x = "Date", y = "Logged Apple Stock Price",
       title = "Forecasts of Log Apple Stock
       Prices Along with Historical Data (Model 2)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```



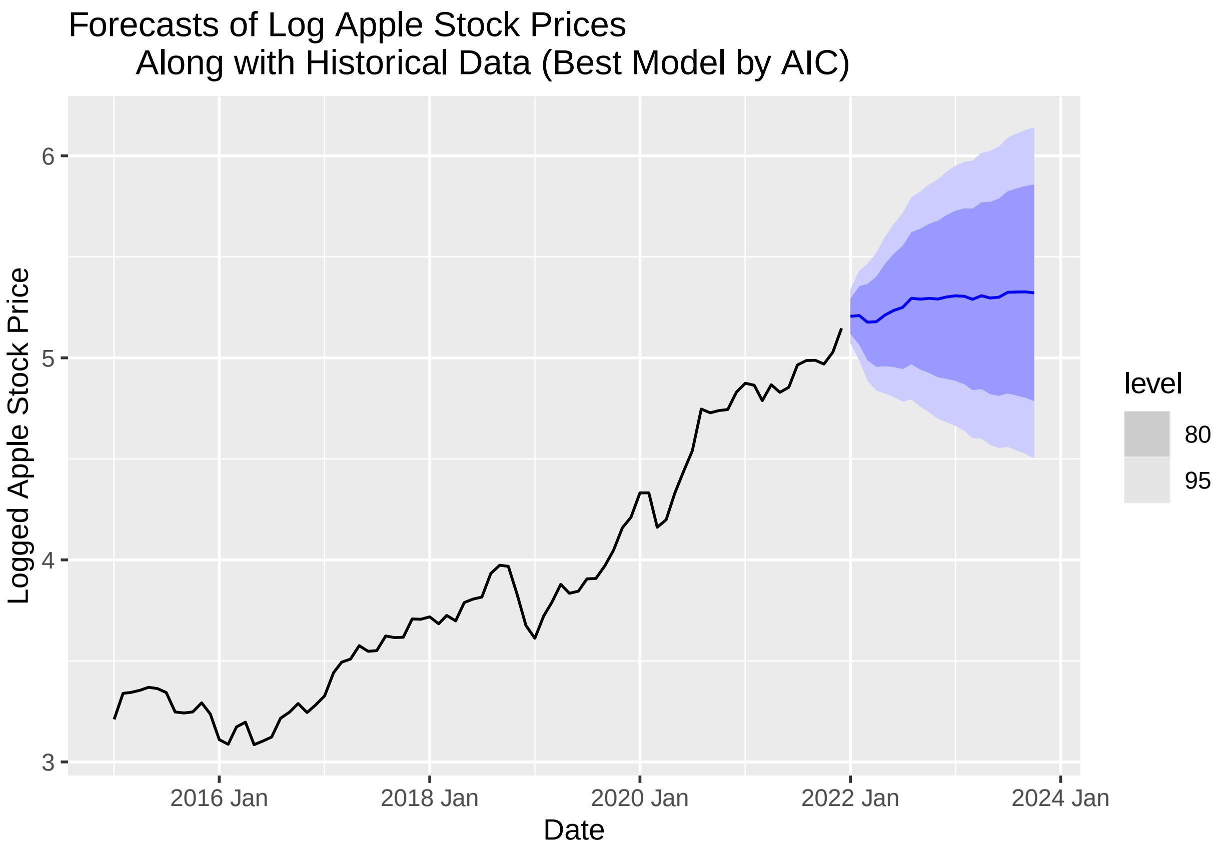

```
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "model_2") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price)) %>%
  autoplot(reference_data) + labs(x = "Date", y = "Apple Stock Price",
       title = "Forecasts of Apple Stock Prices
       Along with Historical Data (Model 2)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```



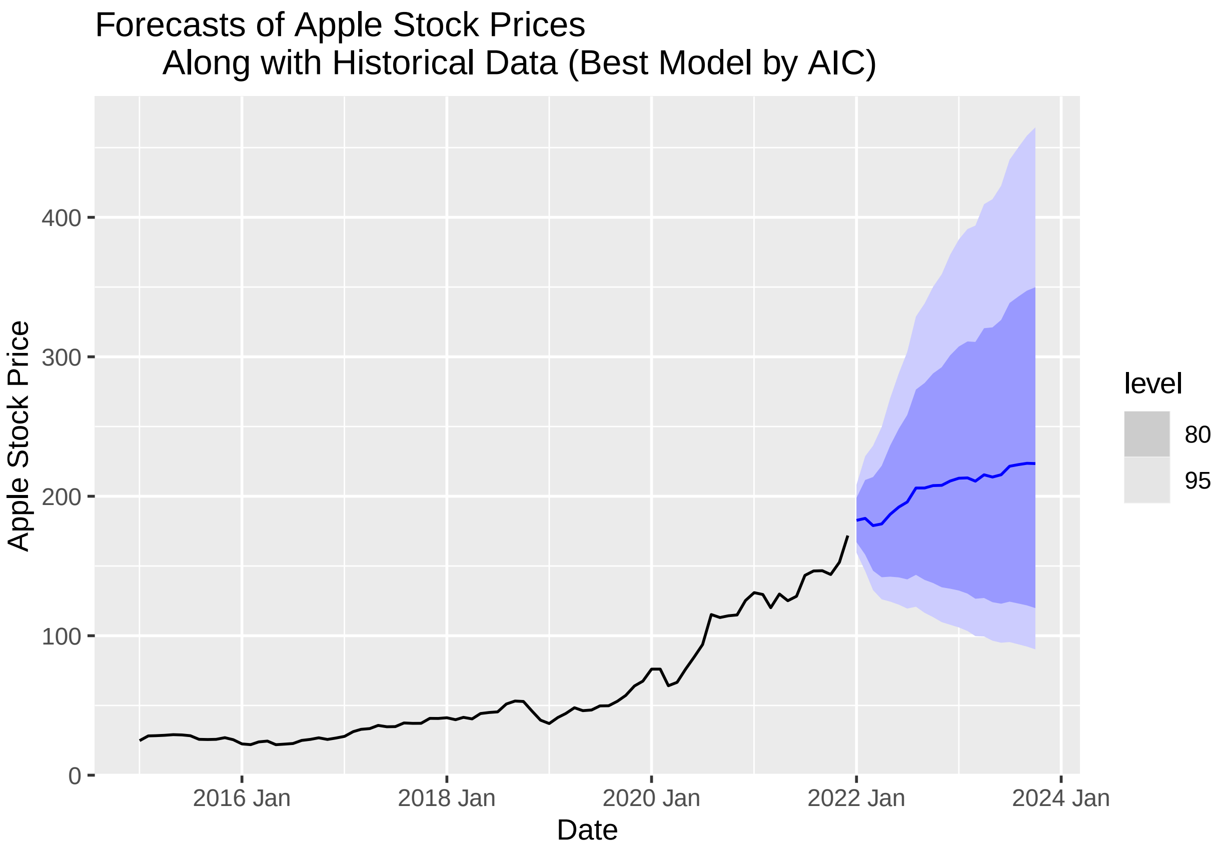

```
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "auto_aic_mod") %>%
  autoplot(monthly_price_avg_tsib_train) + labs(x = "Date",
                                               y = "Logged Apple Stock Price",
       title = "Forecasts of Log Apple Stock Prices
       Along with Historical Data (Best Model by AIC)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```



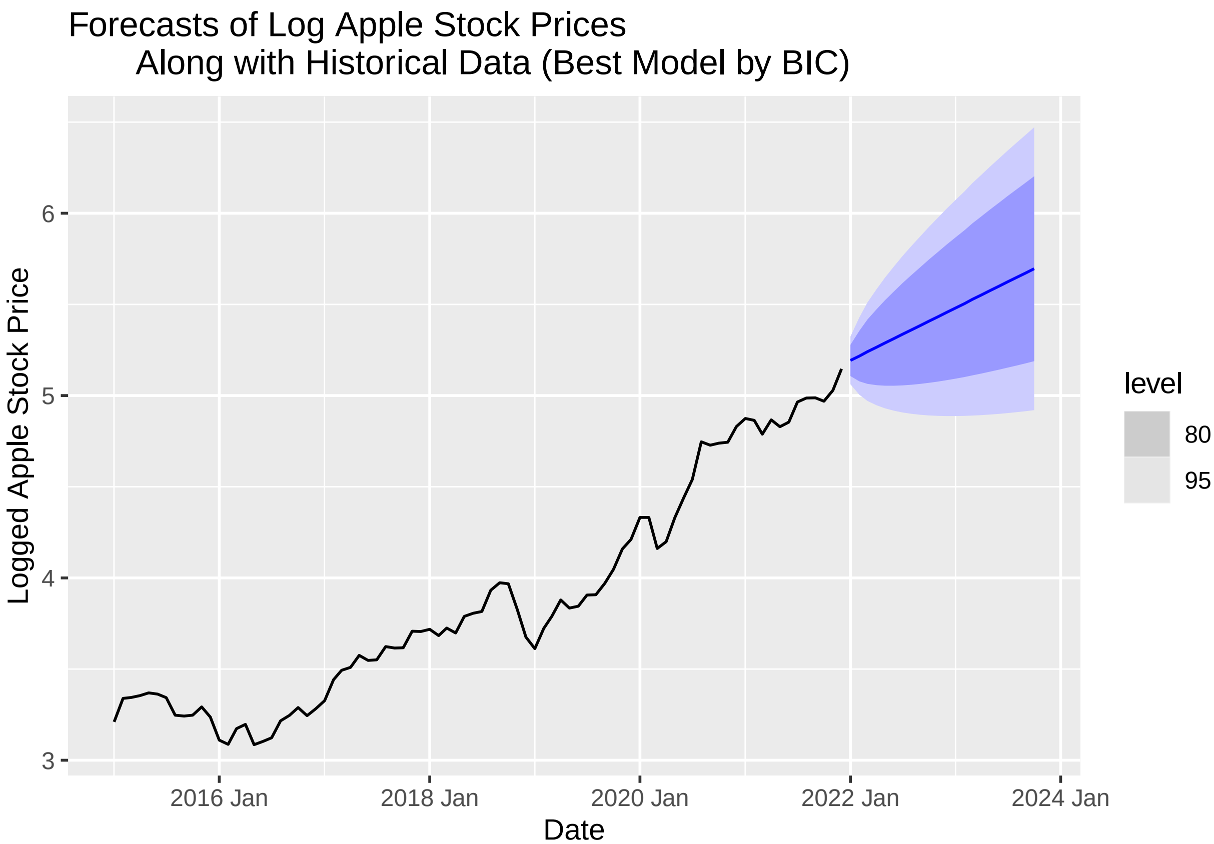

```r
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "auto_aic_mod") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price)) %>%
  autoplot(reference_data) + labs(x = "Date", y = "Apple Stock Price",
       title = "Forecasts of Apple Stock Prices
       Along with Historical Data (Best Model by AIC)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```



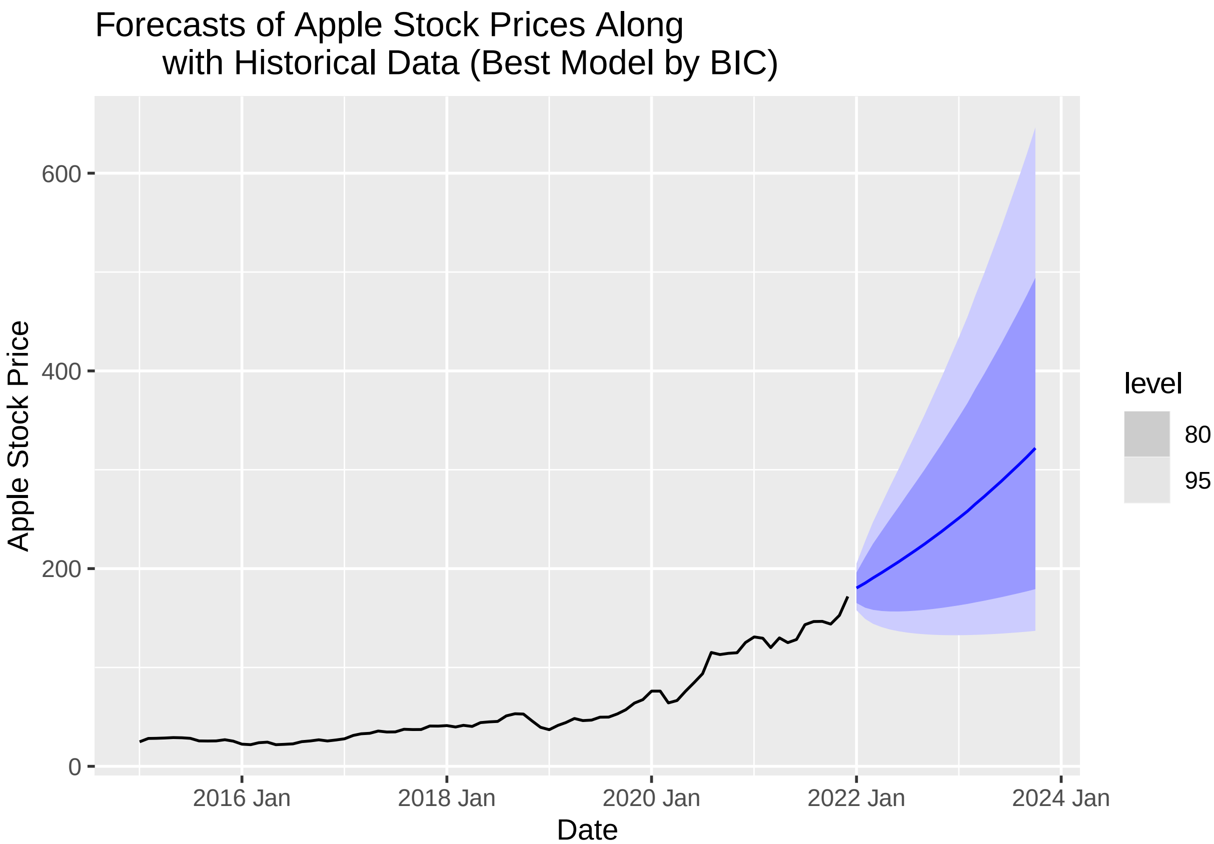

```
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "auto_bic_mod") %>%
  autoplot(monthly_price_avg_tsib_train) +
  labs(x = "Date", y = "Logged Apple Stock Price",
       title = "Forecasts of Log Apple Stock Prices
       Along with Historical Data (Best Model by BIC)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```



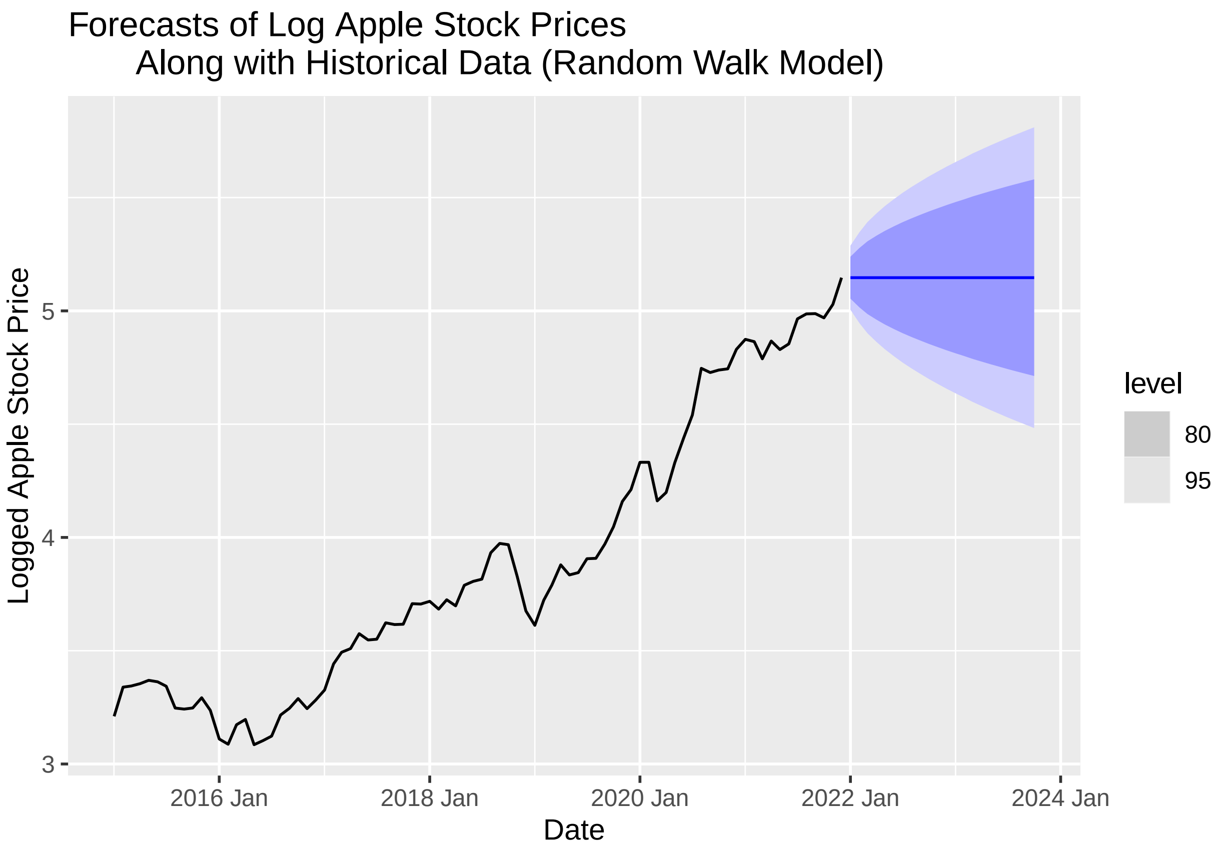

```r
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "auto_bic_mod") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price)) %>%
  autoplot(reference_data) + labs(x = "Date", y = "Apple Stock Price",
       title = "Forecasts of Apple Stock Prices Along
       with Historical Data (Best Model by BIC)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```



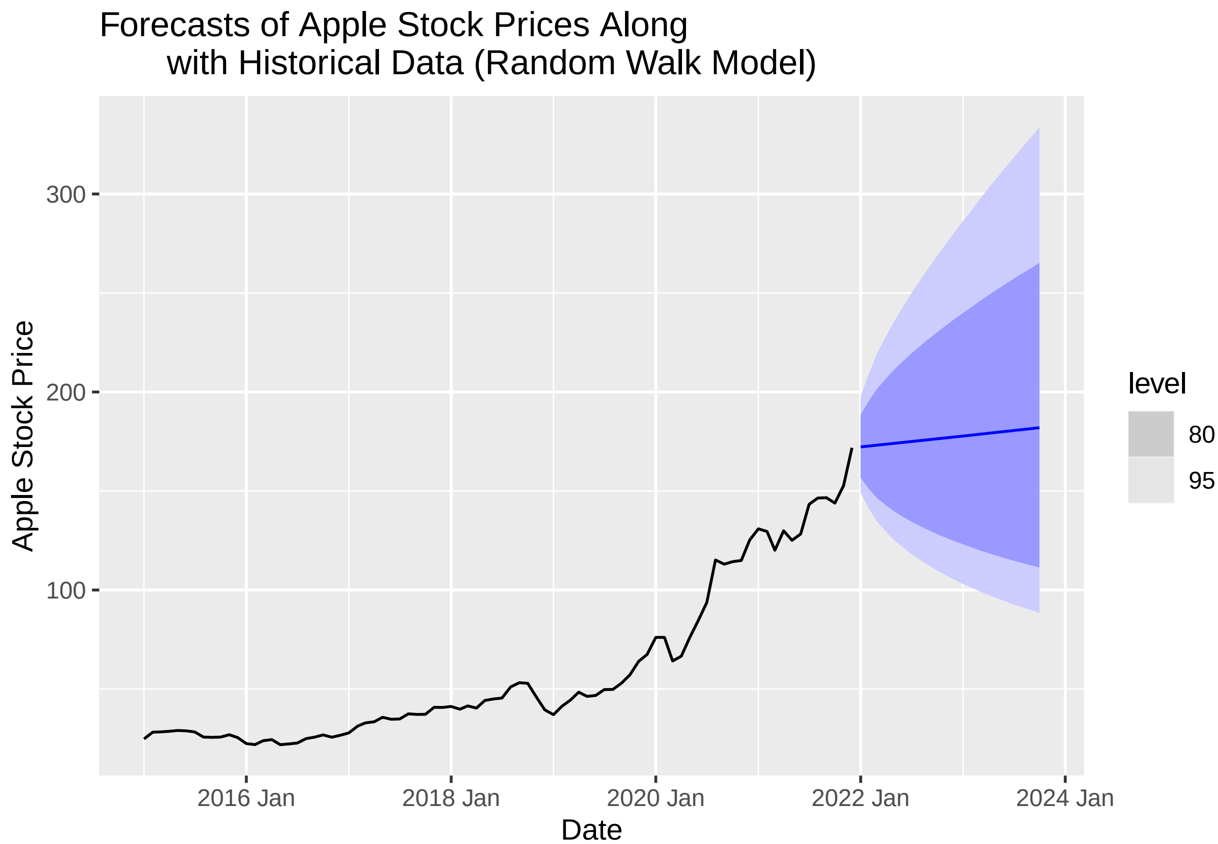

```
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "random_walk_mod") %>%
  autoplot(monthly_price_avg_tsib_train) +
  labs(x = "Date", y = "Logged Apple Stock Price",
       title = "Forecasts of Log Apple Stock Prices
       Along with Historical Data (Random Walk Model)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```




```r
model_comp %>% forecast(new_apple_stock, h=22) %>%
  filter(.model == "random_walk_mod") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price)) %>%
  autoplot(reference_data) + labs(x = "Date", y = "Apple Stock Price",
       title = "Forecasts of Apple Stock Prices Along
       with Historical Data (Random Walk Model)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```




```r
model_forecasts <- model_comp %>% forecast(new_apple_stock, h=22) %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price))
```

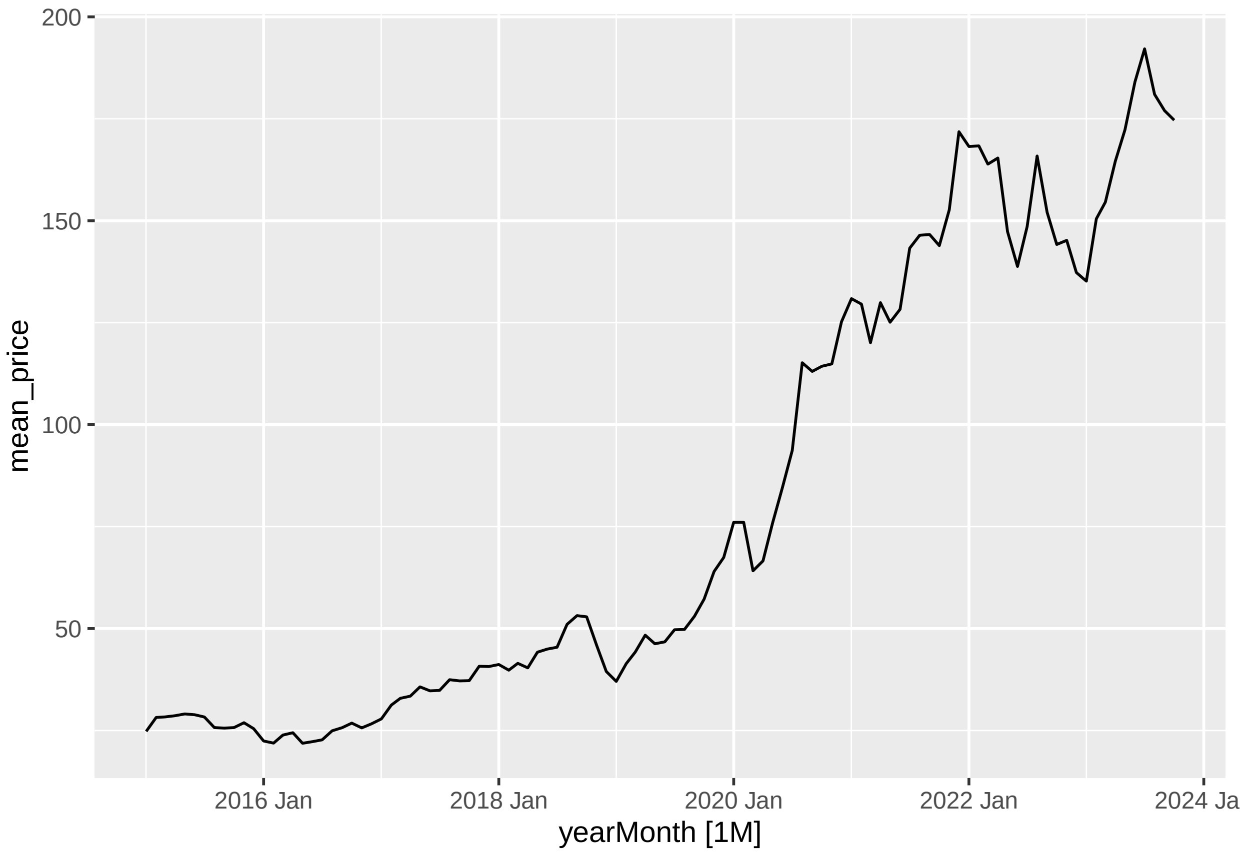
```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```

```r
model_forecasts
```

```
## # A fable: 132 x 4 [1M]
## # Key:     .model [6]
##    .model  yearMonth     mean_price .mean
##    <chr>       <mth>         <dist> <dbl>
##  1 model_1  2022 Jan lN(5.2, 0.011)  181.
##  2 model_1  2022 Feb lN(5.2, 0.027)  177.
##  3 model_1  2022 Mar lN(5.1, 0.042)  163.
##  4 model_1  2022 Apr lN(5.2, 0.053)  176.
##  5 model_1  2022 May lN(5.1, 0.061)  169.
##  6 model_1  2022 Jun lN(5.2, 0.069)  174.
##  7 model_1  2022 Jul lN(5.3, 0.078)  195.
##  8 model_1  2022 Aug lN(5.3, 0.088)  199.
##  9 model_1  2022 Sep lN(5.3, 0.098)  199.
## 10 model_1  2022 Oct  lN(5.3, 0.11)  195.
## # i 122 more rows
```

Compare the RMSE of the different time series models we've created.

```r
monthly_price_avg_tsib %>% autoplot(mean_price)
```




```r
comparison_data <- monthly_price_avg_tsib %>% select(mean_price)
comparison_data
```

```
## # A tsibble: 106 x 2 [1M]
##    mean_price yearMonth
##         <dbl>     <mth>
##  1       24.8  2015 Jan
##  2       28.2  2015 Feb
##  3       28.3  2015 Mar
##  4       28.6  2015 Apr
##  5       29.1  2015 May
##  6       28.9  2015 Jun
##  7       28.3  2015 Jul
##  8       25.7  2015 Aug
##  9       25.6  2015 Sep
## 10       25.7  2015 Oct
## # i 96 more rows
```

```r
accuracy(model_forecasts, comparison_data)
```

```
## # A tibble: 6 x 10
##   .model          .type     ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <chr>           <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 arima_mod       Test   -25.0  28.9  25.1 -16.6  16.6 1.26   1.04 0.760
## 2 auto_aic_mod    Test   -44.6  48.2  44.6 -28.8  28.8 2.24   1.73 0.800
## 3 auto_bic_mod    Test   -84.1  92.3  84.1 -52.8  52.8 4.22   3.32 0.812
```

```
## 4 model_1         Test   -82.4  97.5  82.4 -51.1  51.1 4.14  3.51  0.844
## 5 model_2         Test  -119.  137.  119.  -74.5  74.5 6.00  4.95  0.858
## 6 random_walk_mod Test   -16.6  22.2  18.0 -11.3  12.1 0.903 0.798 0.777
```

Besides the random walk model, the ARIMA model has the lowest RMSE of 28.86. However, we note the tradeoff in its inability to capture seasonality compared to the next best time series model - the SARIMA(1,1,1)(2,0,0)[12] model chosen by the AIC. The ARIMA model produces a flat forecast line, while the SARIMA model appears to produce forecasts that better resemble a natural progression of the time series. Hence, we proceed with comparing forecasts from both models when predicting direction of the stock prices below.

Comparing both the ARIMA and SARIMA models, both forecast the direction of the Apple stock prices with 50% accuracy.

```
future_compare <- monthly_price_avg_tsib_test %>%
  mutate(mean_price = exp(mean_price))
append(as.numeric(diff(
  future_compare$mean_price) > 0), 0) == future_compare$is_up
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
aic_mod_forecasts <- model_forecasts %>% filter(.model == "auto_aic_mod") %>%
  as_tibble() %>% select(yearMonth, .mean)
aic_mod_forecasts$is_up <- as.numeric(append(diff(aic_mod_forecasts$.mean) >
                                               0, 0))
aic_mod_forecasts$is_up
```

```
##  [1] 1 0 1 1 1 1 1 0 1 0 1 1 0 0 1 0 1 1 1 1 0 0
```

```
sum(aic_mod_forecasts$is_up ==
      future_compare$is_up) / length(aic_mod_forecasts$is_up)
```

```
## [1] 0.5
```

```
future_compare <- monthly_price_avg_tsib_test %>%
  mutate(mean_price = exp(mean_price))
append(as.numeric(diff(future_compare$mean_price) > 0), 0) ==
  future_compare$is_up
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
arima_mod_forecasts <- model_forecasts %>% filter(.model == "arima_mod") %>%
  as_tibble() %>% select(yearMonth, .mean)
arima_mod_forecasts$is_up <- as.numeric(append(diff(
  arima_mod_forecasts$.mean) > 0, 0))
arima_mod_forecasts$is_up
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
sum(arima_mod_forecasts$is_up ==
      future_compare$is_up) / length(arima_mod_forecasts$is_up)
```

```
## [1] 0.5
```

```
model_forecasts %>% filter(.model == "random_walk_mod")
```

```
## # A fable: 22 x 4 [1M]
## # Key:     .model [1]
```

```
##    .model          yearMonth      mean_price .mean
##    <chr>               <mth>          <dist> <dbl>
##  1 random_walk_mod  2022 Jan lN(5.1, 0.0052)  172.
##  2 random_walk_mod  2022 Feb   lN(5.1, 0.01)  172.
##  3 random_walk_mod  2022 Mar  lN(5.1, 0.016)  172.
##  4 random_walk_mod  2022 Apr  lN(5.1, 0.021)  172.
##  5 random_walk_mod  2022 May  lN(5.1, 0.026)  172.
##  6 random_walk_mod  2022 Jun  lN(5.1, 0.031)  172.
##  7 random_walk_mod  2022 Jul  lN(5.1, 0.036)  172.
##  8 random_walk_mod  2022 Aug  lN(5.1, 0.042)  172.
##  9 random_walk_mod  2022 Sep  lN(5.1, 0.047)  172.
## 10 random_walk_mod  2022 Oct  lN(5.1, 0.052)  172.
## # i 12 more rows
```

```
rand_walk_mod_forecasts <- model_forecasts %>% filter(.model ==
                                                        "random_walk_mod") %>%
  as_tibble() %>% select(yearMonth, .mean)
rand_walk_mod_forecasts$is_up <- as.numeric(append(diff(
  rand_walk_mod_forecasts$.mean) > 0, 0))
rand_walk_mod_forecasts$is_up
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
sum(rand_walk_mod_forecasts$is_up ==
      future_compare$is_up) / length(rand_walk_mod_forecasts$is_up)
```

```
## [1] 0.5
```