Week 6 homework 2_Alex_C_Parra

Alex Parra

29/6/2022

```
library(tsibble)
```

```
## Warning: package 'tsibble' was built under R version 4.1.3
```

```
##
## Attaching package: 'tsibble'
```

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

```
library(fpp3)
```

```
## Warning: package 'fpp3' was built under R version 4.1.3
```

```
## -- Attaching packages -------------------------------------------- fpp3 0.4.0 --
```

```
## v tibble      3.1.6     v ggplot2     3.3.5
## v dplyr       1.0.7     v tsibbledata 0.4.0
## v tidyr       1.2.0     v feasts      0.2.2
## v lubridate   1.8.0     v fable       0.3.1
```

```
## Warning: package 'tsibbledata' was built under R version 4.1.3
```

```
## Warning: package 'feasts' was built under R version 4.1.3
```

```
## Warning: package 'fabletools' was built under R version 4.1.3
```

```
## Warning: package 'fable' was built under R version 4.1.3
```

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

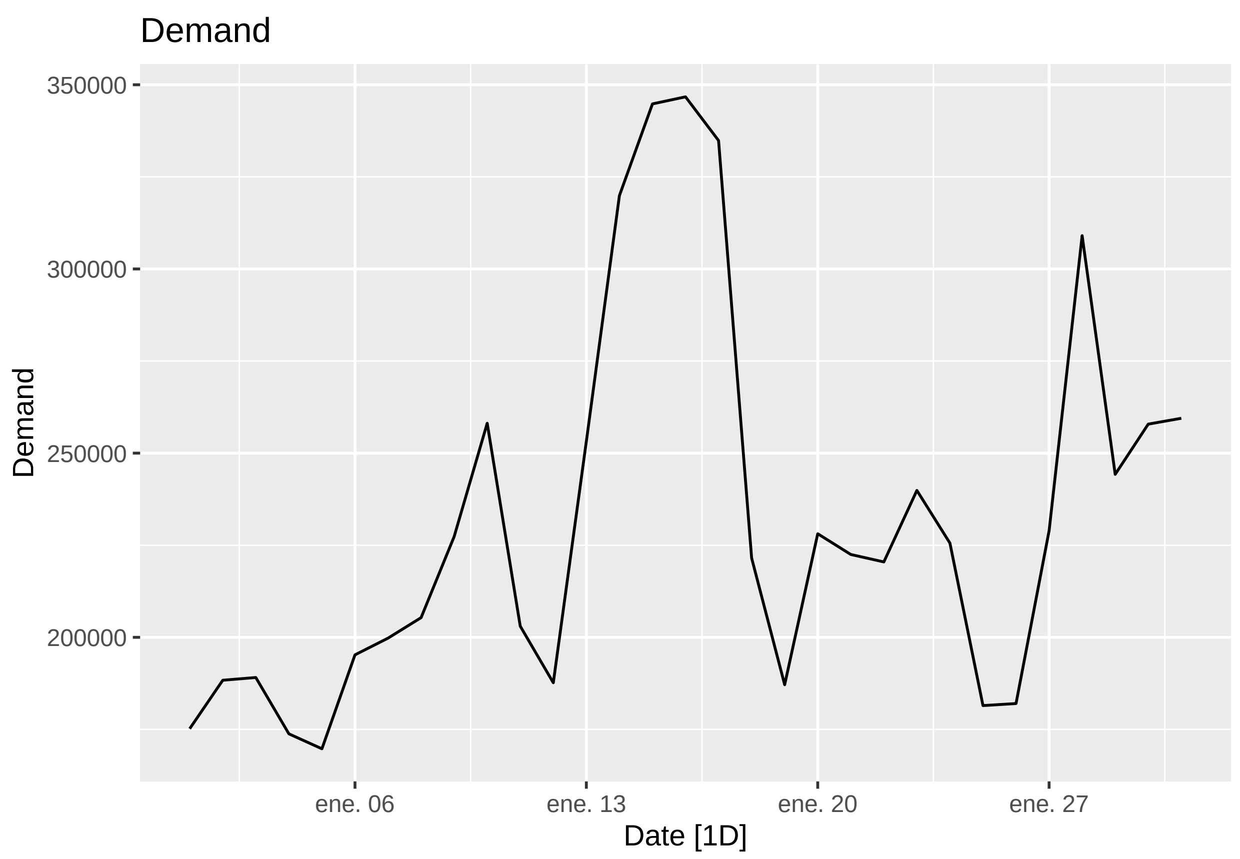
7.10 exercise 1

```r
jan14_vic_elec <- vic_elec %>%
  filter(yearmonth(Time) == yearmonth("2014 Jan")) %>%
  index_by(Date = as_date(Time)) %>%
  summarise(
    Demand = sum(Demand),
    Temperature = max(Temperature)
  )
jan14_vic_elec
```

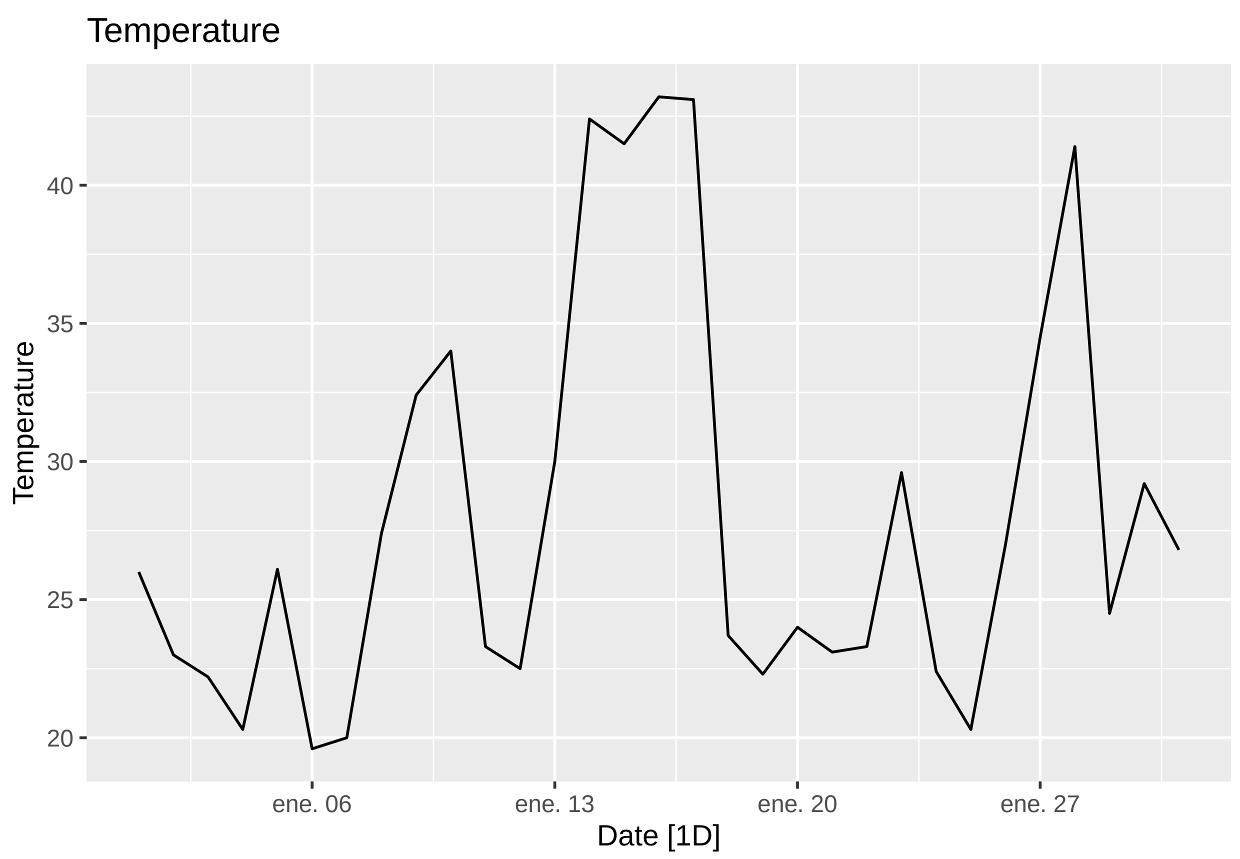
```
## # A tsibble: 31 x 3 [1D]
##    Date        Demand Temperature
##    <date>       <dbl>       <dbl>
##  1 2014-01-01 175185.        26
##  2 2014-01-02 188351.        23
##  3 2014-01-03 189086.        22.2
##  4 2014-01-04 173798.        20.3
##  5 2014-01-05 169733.        26.1
##  6 2014-01-06 195241.        19.6
##  7 2014-01-07 199770.        20
##  8 2014-01-08 205339.        27.4
##  9 2014-01-09 227334.        32.4
## 10 2014-01-10 258111.        34
## # ... with 21 more rows
```

a)

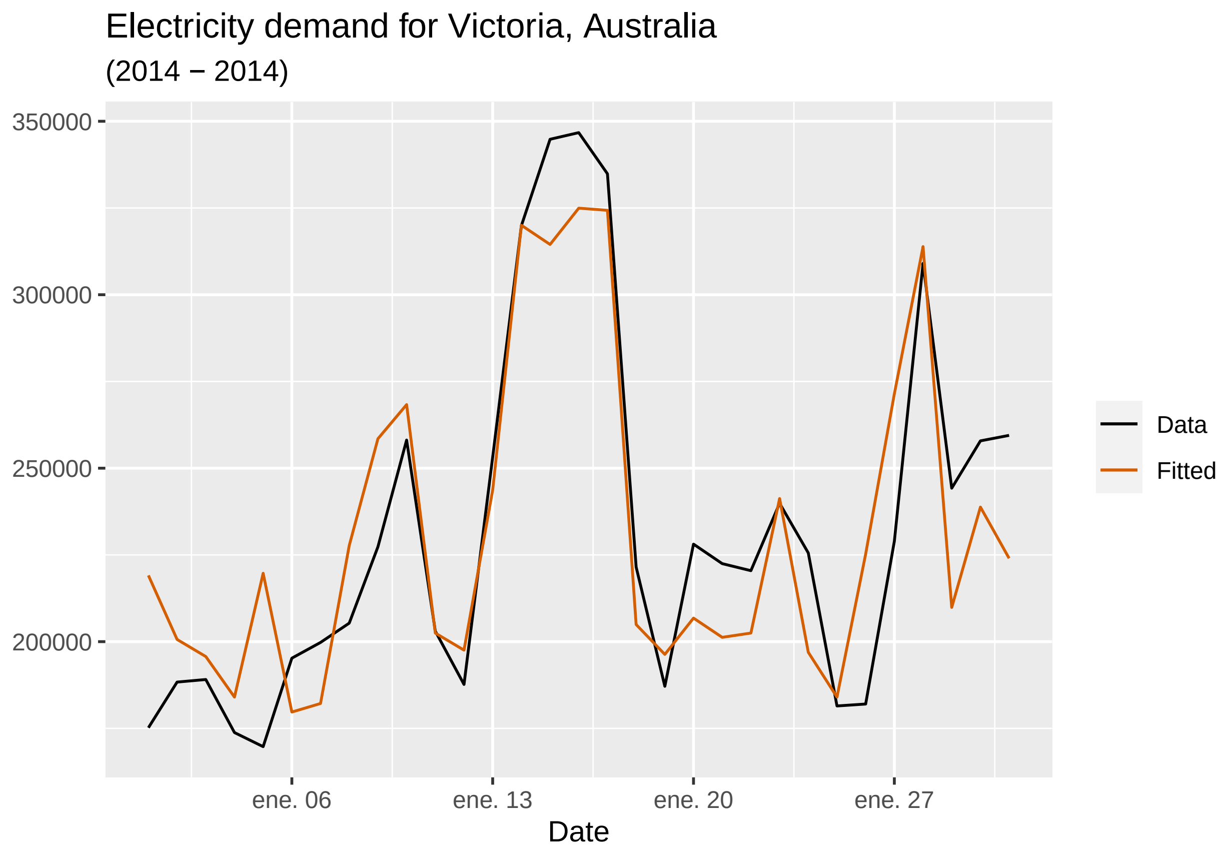
```r
jan14_vic_elec %>% autoplot(Demand) + labs(title = "Demand")
```



```
jan14_vic_elec %>% autoplot(Temperature) + labs(title = "Temperature")
```



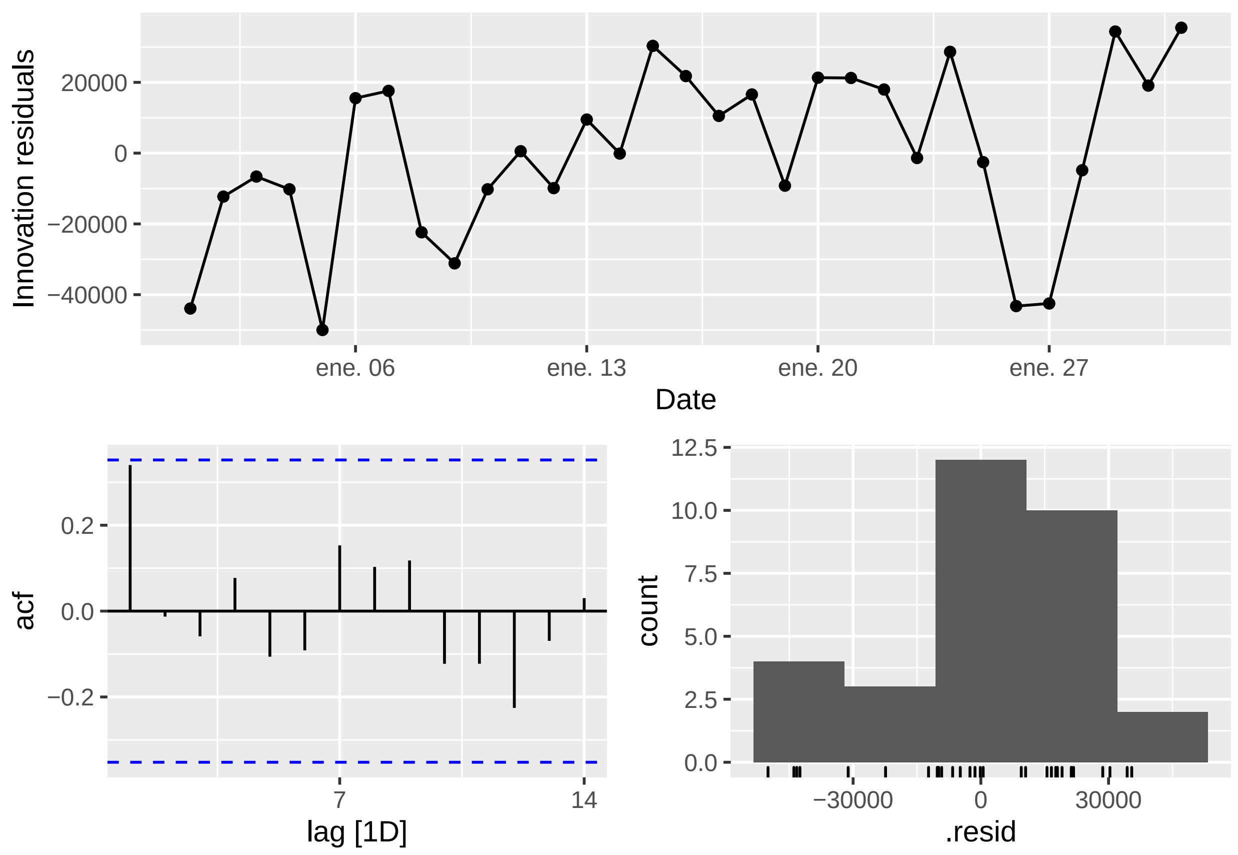
```r
#model
fit <- jan14_vic_elec %>%
  model(demand_temp = TSLM(Demand ~ Temperature))


augment(fit) %>%
  ggplot(aes(x = Date)) +
  geom_line(aes(y = Demand, colour = "Data")) +
  geom_line(aes(y = .fitted, colour = "Fitted")) +
  labs(y = NULL,
    title = "Electricity demand for Victoria, Australia", subtitle = "(2014 - 2014)") +
  scale_colour_manual(values=c(Data="black",Fitted="#D55E00")) +
  guides(colour = guide_legend(title = NULL))
```



There is a positive relation because people increase their demand for electricity as temperature raises, since they turn on AC in order to cool the room or house.

b)

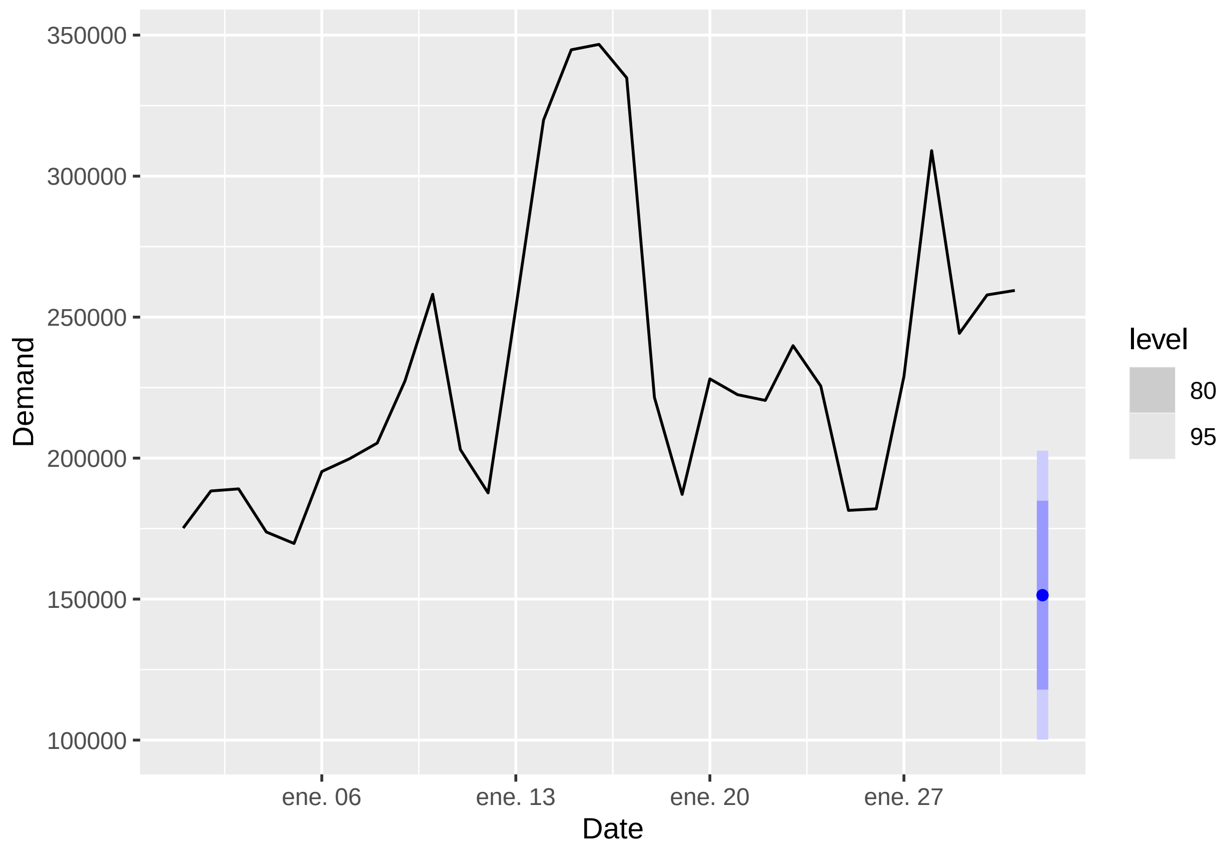
```
fit %>% gg_tsresiduals()
```



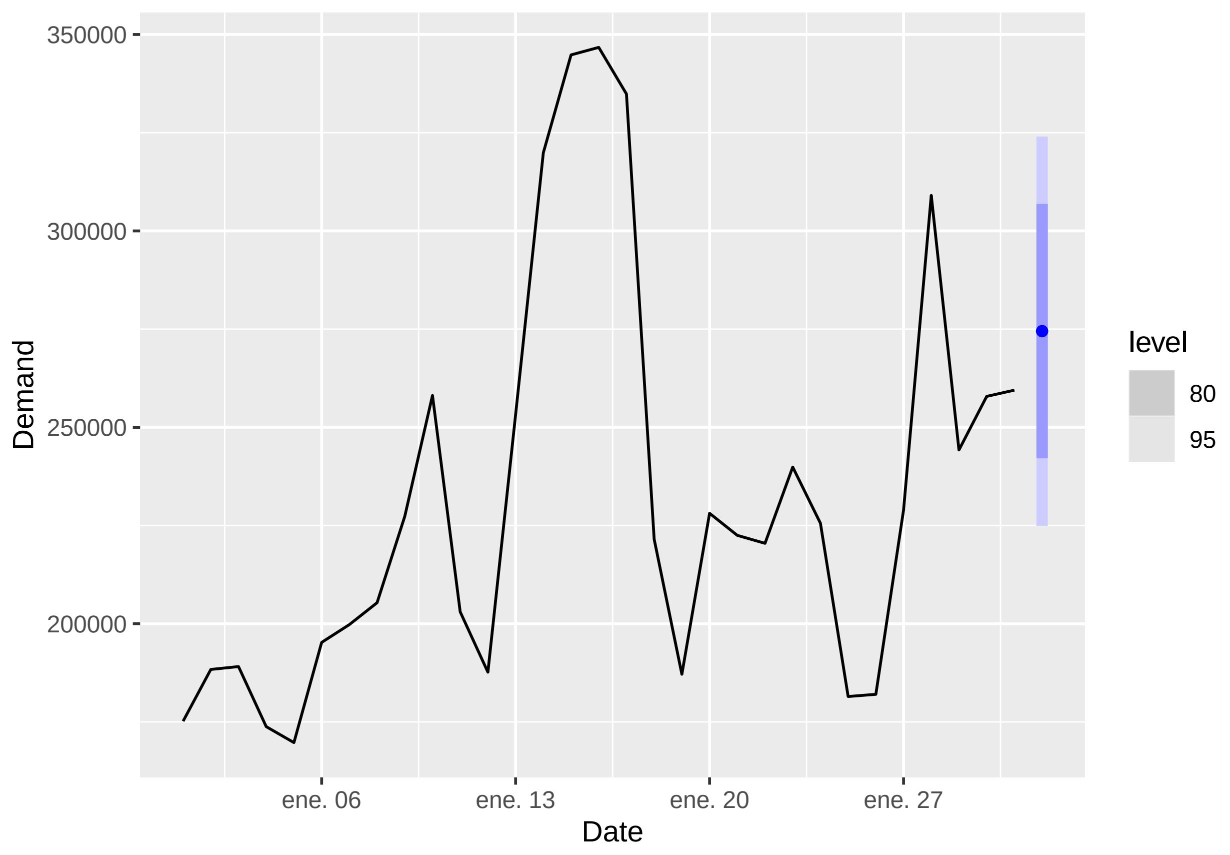
We can see that the residuals seem to be slightly skewed to the positives value, there also seems to be some outliers on the negative size at the biggening and in the last January. We can see that there isn't any lag on the autocorrelation plot.

c)

```
jan14_vic_elec %>%
  model(TSLM(Demand ~ Temperature)) %>%
  forecast(
    new_data(jan14_vic_elec, 1) %>%
      mutate(Temperature = 15)
  ) %>%
  autoplot(jan14_vic_elec)
```



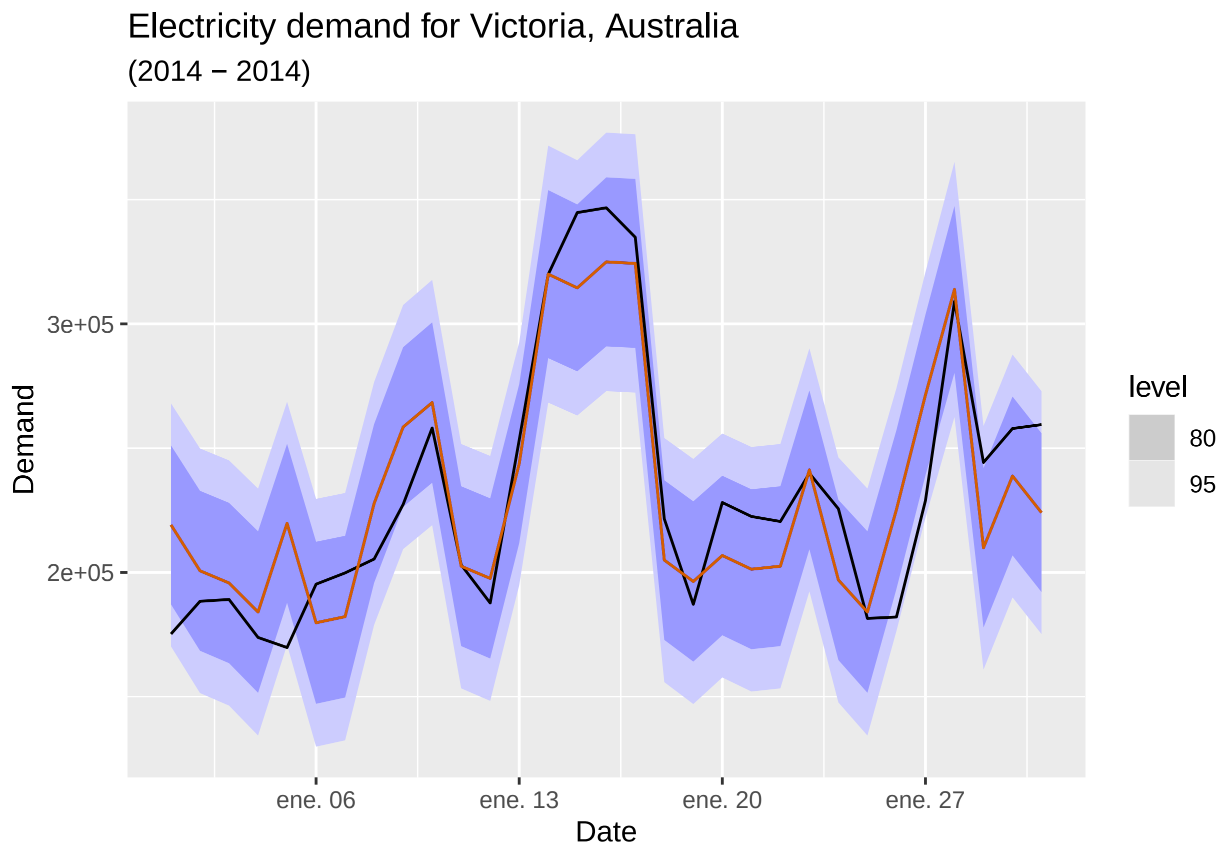
```r
jan14_vic_elec %>%
  model(TSLM(Demand ~ Temperature)) %>%
  forecast(
    new_data(jan14_vic_elec, 1) %>%
      mutate(Temperature = 35)
  ) %>%
  autoplot(jan14_vic_elec)
```



We have seen that there is a relationship between Temperature and Electricity Demand, that means that if the temperature for the next day is 15 or 35 degrees it will drive up or down the demand. But we have to take into consideration that at no point in the time series there is a temperature of 15 ºC, this means that we do not know if the relation between the variables holds outside the data that the model is train with.

d)

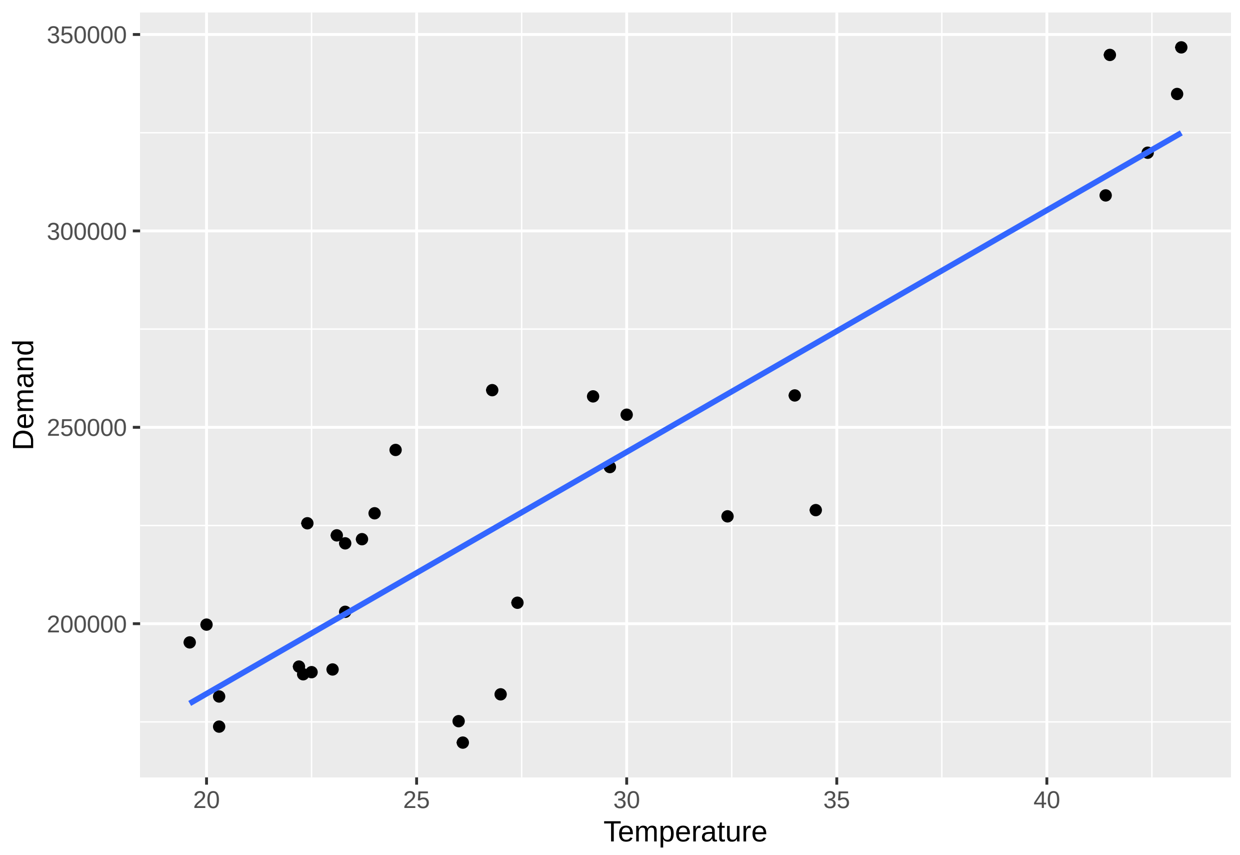
```
fc <- fit %>%
  forecast(jan14_vic_elec)

fc %>%
  autoplot(jan14_vic_elec) +
  geom_line(aes(y = .fitted), col="#D55E00",
            data = augment(fit)) +
  labs(title = "Electricity demand for Victoria, Australia", subtitle = "(2014 - 2014)") +
  guides(colour = guide_legend(title = "Forecast"))
```



e)

```
jan14_vic_elec %>%
  ggplot(aes(x = Temperature, y = Demand)) +
  labs(y = "Demand",
       x = "Temperature") +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



We can see that there is a positive relation between Demand and Temperature. As temperature increases the demand for electricity goes up.

```
augment(fit)
```

```
## # A tsibble: 31 x 6 [1D]
## # Key:       .model [1]
##    .model      Date        Demand .fitted  .resid  .innov
##    <chr>       <date>       <dbl>   <dbl>   <dbl>   <dbl>
##  1 demand_temp 2014-01-01 175185. 219096. -43911. -43911.
##  2 demand_temp 2014-01-02 188351. 200633. -12282. -12282.
##  3 demand_temp 2014-01-03 189086. 195709.  -6624.  -6624.
##  4 demand_temp 2014-01-04 173798. 184016. -10219. -10219.
##  5 demand_temp 2014-01-05 169733. 219711. -49978. -49978.
##  6 demand_temp 2014-01-06 195241. 179708.  15533.  15533.
##  7 demand_temp 2014-01-07 199770. 182170.  17601.  17601.
##  8 demand_temp 2014-01-08 205339. 227712. -22373. -22373.
##  9 demand_temp 2014-01-09 227334. 258483. -31149. -31149.
## 10 demand_temp 2014-01-10 258111. 268330. -10219. -10219.
## # ... with 21 more rows
```

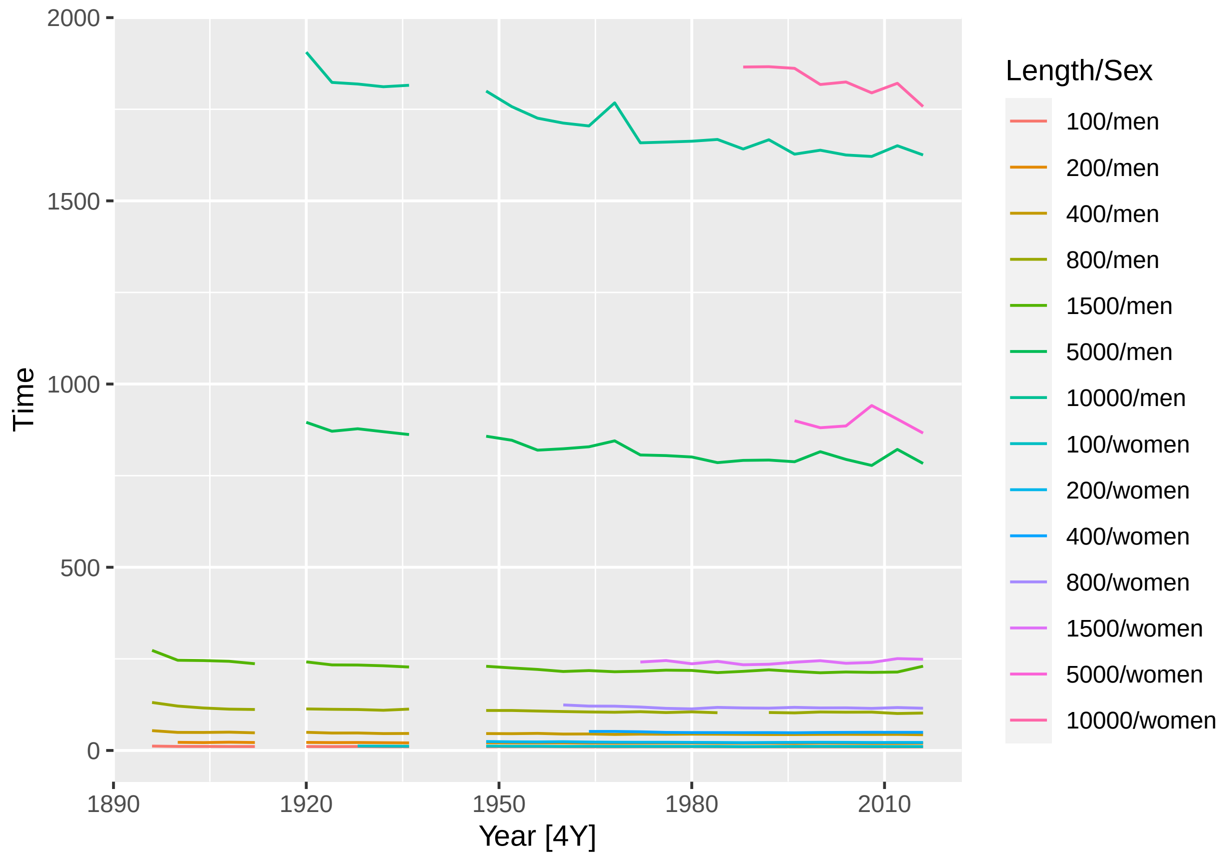
7.10 exercise 2

```r
olympic_running
```

```
## # A tsibble: 312 x 4 [4Y]
## # Key:       Length, Sex [14]
##     Year Length Sex    Time
##    <int>  <int> <chr> <dbl>
##  1  1896    100 men    12
##  2  1900    100 men    11
##  3  1904    100 men    11
##  4  1908    100 men    10.8
##  5  1912    100 men    10.8
##  6  1916    100 men    NA
##  7  1920    100 men    10.8
##  8  1924    100 men    10.6
##  9  1928    100 men    10.8
## 10  1932    100 men    10.3
## # ... with 302 more rows
```

```r
autoplot(olympic_running)
```

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



b)

```r
fit <- olympic_running %>%
  model(time = TSLM(Time ~ Year))
print('')
```

```
## [1] ""
```

```r
print('############################################################')
```

```
## [1] "############################################################"
```

```r
print('########################## 100 m ##########################')
```

```
## [1] "########################## 100 m ##########################"
```

```r
print('############################################################')
```

```
## [1] "############################################################"
```

```r
print('')
```

```
## [1] ""
```

```r
fit %>%
  filter(Sex == "men" & Length== 100) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.3443995 -0.1081360  0.0007715  0.0750701  0.9015537
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 35.011581   2.347437   14.91 2.94e-14 ***
## Year        -0.012612   0.001198  -10.52 7.24e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2321 on 26 degrees of freedom
## Multiple R-squared: 0.8099,  Adjusted R-squared: 0.8026
## F-statistic: 110.8 on 1 and 26 DF, p-value: 7.2403e-11
```

```r
print('-----------------------------------------------------------')
```

```
## [1] "-----------------------------------------------------------"
```

```
fit %>%
  filter(Sex == "women" & Length== 100) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44833 -0.11159  0.05775  0.11731  0.36057
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 39.188164   3.697686  10.598 2.05e-09 ***
## Year        -0.014185   0.001872  -7.577 3.72e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2232 on 19 degrees of freedom
## Multiple R-squared: 0.7513,  Adjusted R-squared: 0.7382
## F-statistic:  57.4 on 1 and 19 DF, p-value: 3.7226e-07
```

```
print('')
```

```
## [1] ""
```

```
print('############################################################')
```

```
## [1] "############################################################"
```

```
print('########################## 200 m ##########################')
```

```
## [1] "########################## 200 m ##########################"
```

```
print('############################################################')
```

```
## [1] "############################################################"
```

```
print('')
```

```
## [1] ""
```

```
fit %>%
  filter(Sex == "men" & Length== 200) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6112 -0.1872 -0.1046  0.1935  0.6959
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 69.376498   3.553448   19.52  < 2e-16 ***
## Year        -0.024881   0.001812  -13.73  3.8e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3315 on 25 degrees of freedom
## Multiple R-squared: 0.8829,  Adjusted R-squared: 0.8782
## F-statistic: 188.5 on 1 and 25 DF, p-value: 3.7956e-13
```

```
print('-----------------------------------------------------------')
```

```
## [1] "-----------------------------------------------------------"
```

```
fit %>%
  filter(Sex == "women" & Length== 200) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.93598 -0.39344  0.08043  0.37624  0.78230
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 89.137613  11.267724   7.911 6.41e-07 ***
## Year        -0.033633   0.005685  -5.916 2.17e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5005 on 16 degrees of freedom
## Multiple R-squared: 0.6863,  Adjusted R-squared: 0.6667
## F-statistic:    35 on 1 and 16 DF, p-value: 2.1706e-05
```

```
print('')
```

```
## [1] ""
```

```
print('############################################################')
```

```
## [1] "############################################################"
```

```
print('########################## 400 m ##########################')
```

```
## [1] "########################## 400 m ##########################"
```

```
print('###########################################################')
```

```
## [1] "###########################################################"
```

```
print('')
```

```
## [1] ""
```

```
fit %>%
  filter(Sex == "men" & Length== 400) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6001 -0.5747 -0.2858  0.5751  4.1505
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 172.481477  11.487522   15.02 2.52e-14 ***
## Year         -0.064574   0.005865  -11.01 2.75e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.136 on 26 degrees of freedom
## Multiple R-squared: 0.8234,  Adjusted R-squared: 0.8166
## F-statistic: 121.2 on 1 and 26 DF, p-value: 2.7524e-11
```

```
print('-----------------------------------------------------------')
```

```
## [1] "-----------------------------------------------------------"
```

```
fit %>%
  filter(Sex == "women" & Length== 400) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1593 -1.0142  0.1024  0.7749  1.4797
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 129.39165   33.89755   3.817  0.00245 **
## Year         -0.04008    0.01703  -2.353  0.03652 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.028 on 12 degrees of freedom
## Multiple R-squared: 0.3157,  Adjusted R-squared: 0.2587
## F-statistic: 5.536 on 1 and 12 DF, p-value: 0.036523
```

```
print('')
```

```
## [1] ""
```

```
print('############################################################')
```

```
## [1] "############################################################"
```

```
print('########################## 800 m #########################')
```

```
## [1] "########################## 800 m #########################"
```

```
print('############################################################')
```

```
## [1] "############################################################"
```

```
print('')
```

```
## [1] ""
```

```
fit %>%
  filter(Sex == "men" & Length== 800) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.7306 -1.8734 -0.8449  0.6851 12.9408
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 405.8457    34.6506  11.712 1.21e-11 ***
## Year         -0.1518     0.0177  -8.576 6.47e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.387 on 25 degrees of freedom
## Multiple R-squared: 0.7463,  Adjusted R-squared: 0.7361
## F-statistic: 73.54 on 1 and 25 DF, p-value: 6.4705e-09
```

```r
print('------------------------------------------------------------')
```

```
## [1] "------------------------------------------------------------"
```

```r
fit %>%
  filter(Sex == "women" & Length== 800) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.90197 -1.54437 -0.08512  1.55387  7.10393
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 511.36950   76.14729   6.716 9.85e-06 ***
## Year         -0.19796    0.03837  -5.159 0.000145 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.401 on 14 degrees of freedom
## Multiple R-squared: 0.6553,  Adjusted R-squared: 0.6307
## F-statistic: 26.61 on 1 and 14 DF, p-value: 0.00014512
```

```r
print('')
```

```
## [1] ""
```

```r
print('############################################################')
```

```
## [1] "############################################################"
```

```r
print('######################### 1500 m ##########################')
```

```
## [1] "######################### 1500 m ##########################"
```

```r
print('############################################################')
```

```
## [1] "############################################################"
```

```r
print('')
```

```
## [1] ""
```

```r
fit %>%
  filter(Sex == "men" & Length== 1500) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.302  -4.585  -1.215   1.925  27.133
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 843.43666   81.81773  10.309 1.12e-10 ***
## Year         -0.31507    0.04177  -7.543 5.23e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.09 on 26 degrees of freedom
## Multiple R-squared: 0.6864,  Adjusted R-squared: 0.6743
## F-statistic:  56.9 on 1 and 26 DF, p-value: 5.2345e-08
```

```r
print('-----------------------------------------------------------')
```

```
## [1] "-----------------------------------------------------------"
```

```r
fit %>%
  filter(Sex == "women" & Length== 1500) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.8003 -3.9012  0.7371  3.3202  6.4808
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -50.9926   211.3851  -0.241    0.814
## Year          0.1468     0.1060   1.384    0.196
##
## Residual standard error: 5.071 on 10 degrees of freedom
## Multiple R-squared: 0.1608,  Adjusted R-squared: 0.07691
## F-statistic: 1.917 on 1 and 10 DF, p-value: 0.19635
```

```r
print('')
```

```
## [1] ""
```

```r
print('############################################################')
```

```
## [1] "############################################################"
```

```r
print('######################### 5000 m #########################')
```

```
## [1] "######################### 5000 m #########################"
```

```r
print('############################################################')
```

```
## [1] "############################################################"
```

```r
print('')
```

```
## [1] ""
```

```r
fit %>%
  filter(Sex == "men" & Length== 5000) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -24.311 -11.668  -1.096   7.515  40.596
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2853.1995   205.1246  13.910 2.22e-12 ***
## Year          -1.0299     0.1042  -9.881 1.50e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.66 on 22 degrees of freedom
## Multiple R-squared: 0.8161,   Adjusted R-squared: 0.8078
## F-statistic: 97.64 on 1 and 22 DF, p-value: 1.4995e-09
```

```r
print('----------------------------------------------------------')
```

```
## [1] "----------------------------------------------------------"
```

```r
fit %>%
  filter(Sex == "women" & Length== 5000) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1504.175   3467.321   0.434    0.687
## Year          -0.303      1.728  -0.175    0.869
##
## Residual standard error: 28.92 on 4 degrees of freedom
## Multiple R-squared: 0.007624,    Adjusted R-squared: -0.2405
## F-statistic: 0.03073 on 1 and 4 DF, p-value: 0.86936
```

```
print('')
```

```
## [1] ""
```

```
print('###########################################################')
```

```
## [1] "###########################################################"
```

```
print('######################### 10000 m #########################')
```

```
## [1] "######################### 10000 m #########################"
```

```
print('###########################################################')
```

```
## [1] "###########################################################"
```

```
print('')
```

```
## [1] ""
```

```
fit %>%
  filter(Sex == "men" & Length== 10000) %>%
  report()
```

```
## Series: Time
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.964 -24.222  -4.091  11.911  58.859
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6965.4594   378.4635   18.41 7.52e-15 ***
## Year          -2.6659     0.1923  -13.86 2.37e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 28.89 on 22 degrees of freedom
## Multiple R-squared: 0.8973,   Adjusted R-squared: 0.8926
## F-statistic: 192.2 on 1 and 22 DF, p-value: 2.3727e-12
```

```
print('-----------------------------------------------------------')
```

```
## [1] "-----------------------------------------------------------"
```

```
fit %>%
  filter(Sex == "women" & Length== 10000) %>%
  report()
```

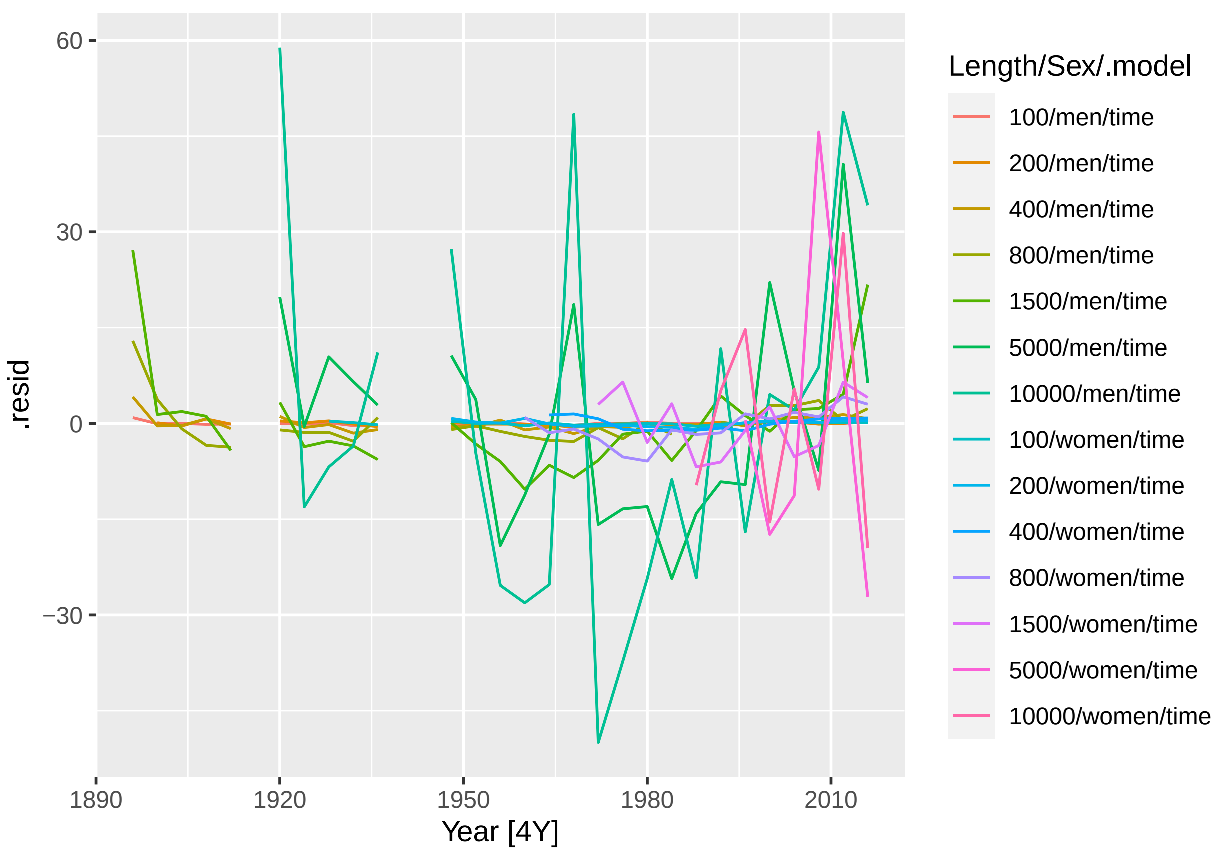
```
## Series: Time
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.550 -11.594  -2.285   7.731  29.765
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8825.2600  1402.4211   6.293  0.00075 ***
## Year          -3.4962     0.7005  -4.991  0.00247 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.16 on 6 degrees of freedom
## Multiple R-squared: 0.8059,   Adjusted R-squared: 0.7735
## F-statistic: 24.91 on 1 and 6 DF, p-value: 0.0024746
```

```
#report(fit)
```

We can see that the decrease in time change depending on the distance, we can see that the values are for: 100 meters: -0.012 (men) and -0.014 (women) 200 meters: -0.024 (men) and -0.033 (women) 400 meters: -0.064 (men) and -0.040 (women) 800 meters: -0.151 (men) and -0.197 (women) 1500 meters: -0.315 (men) and 0.146 *(women) 5000 meters: -1.029 (men) and -0.303* (women) 10000 meters: -2.665 (men) and -3.496 (women) We can see that the longer the distance the greater reduction in time we can see over the years. We can see a couple of outliers in the 1500 m for women where the times increased over the years, and in 5000 m for women where the time decrease much less than on the males, where normally the times are really close between men and women, even in most cases the women got greater reductions in time than males.

c)

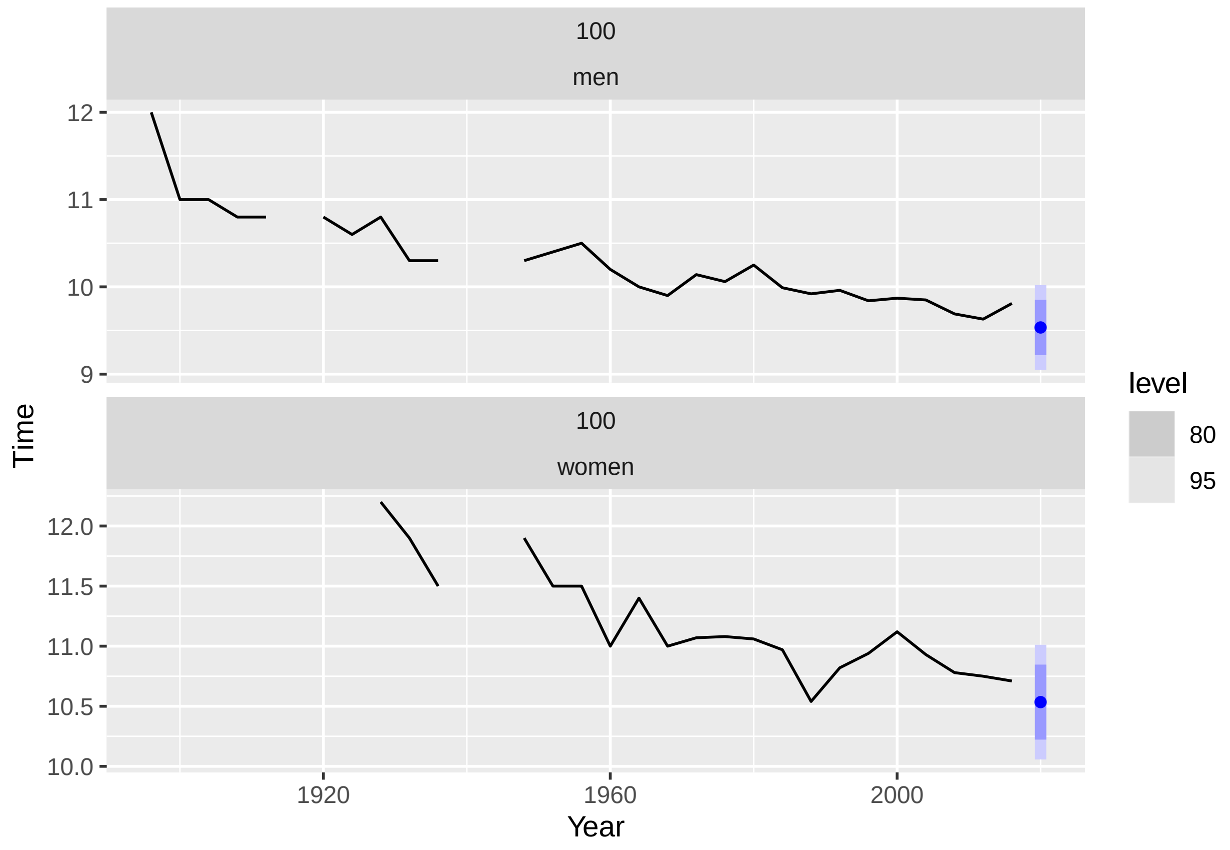
```
residuals(fit) %>%
  autoplot(.resid)
```



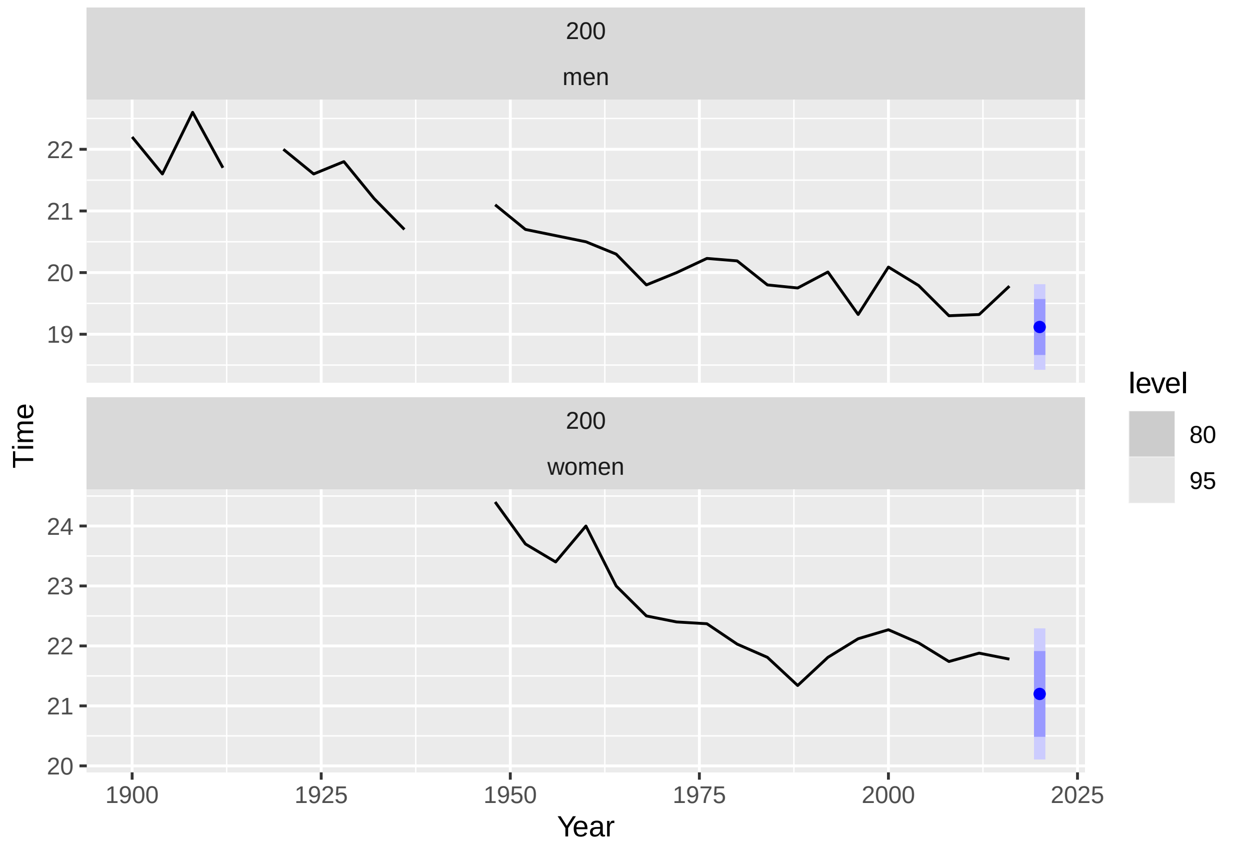
We can see that some models have huge errors over the years, most of those errors, we can also see that some models errors are smaller than the rest, this is probably because some models distance are way shorter 100 m vs 10,000 m meaning the time is also way different, this results in the errors of the models with shorter distances being way smaller than the others ones.

d)

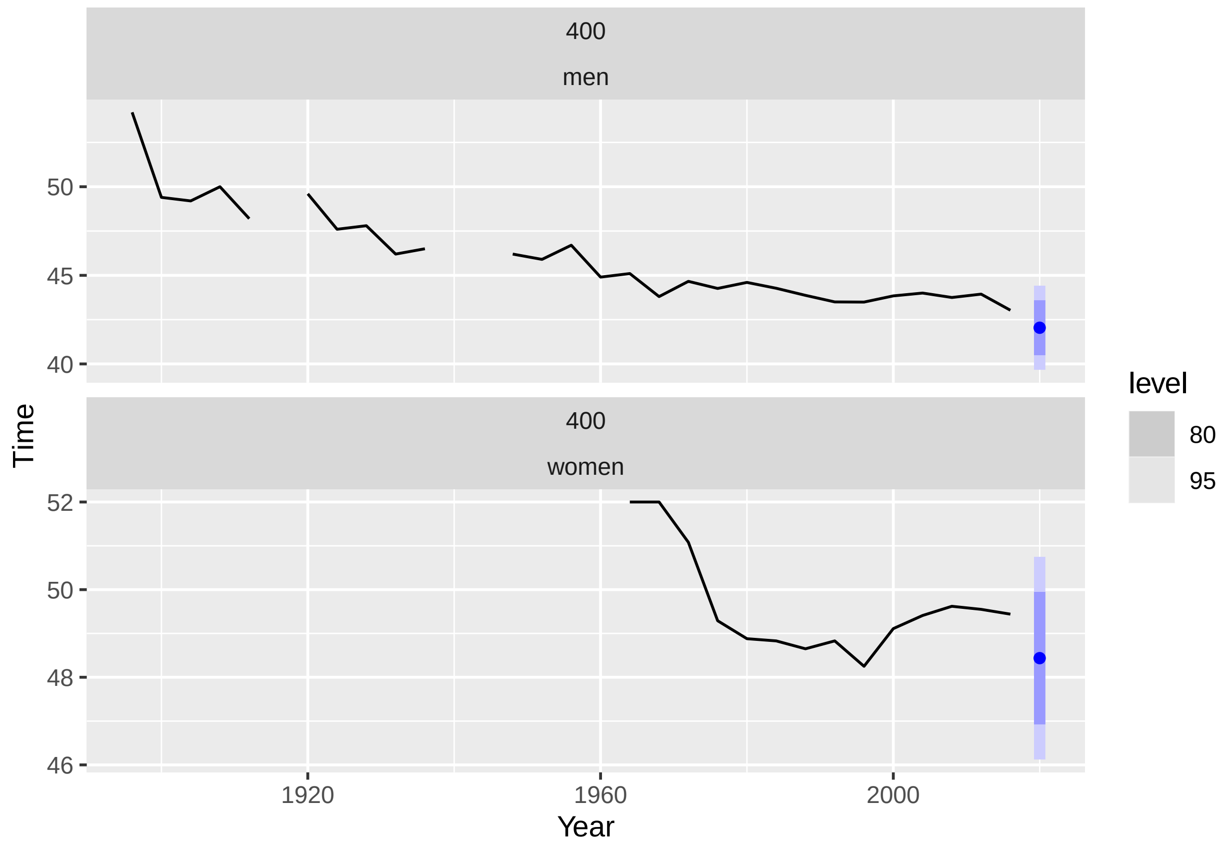
```
fit %>%
  filter(Length == 100) %>%
  forecast(
    new_data(olympic_running, 1) %>%
      mutate(Year = 2020)
  ) %>%
  autoplot(olympic_running)
```



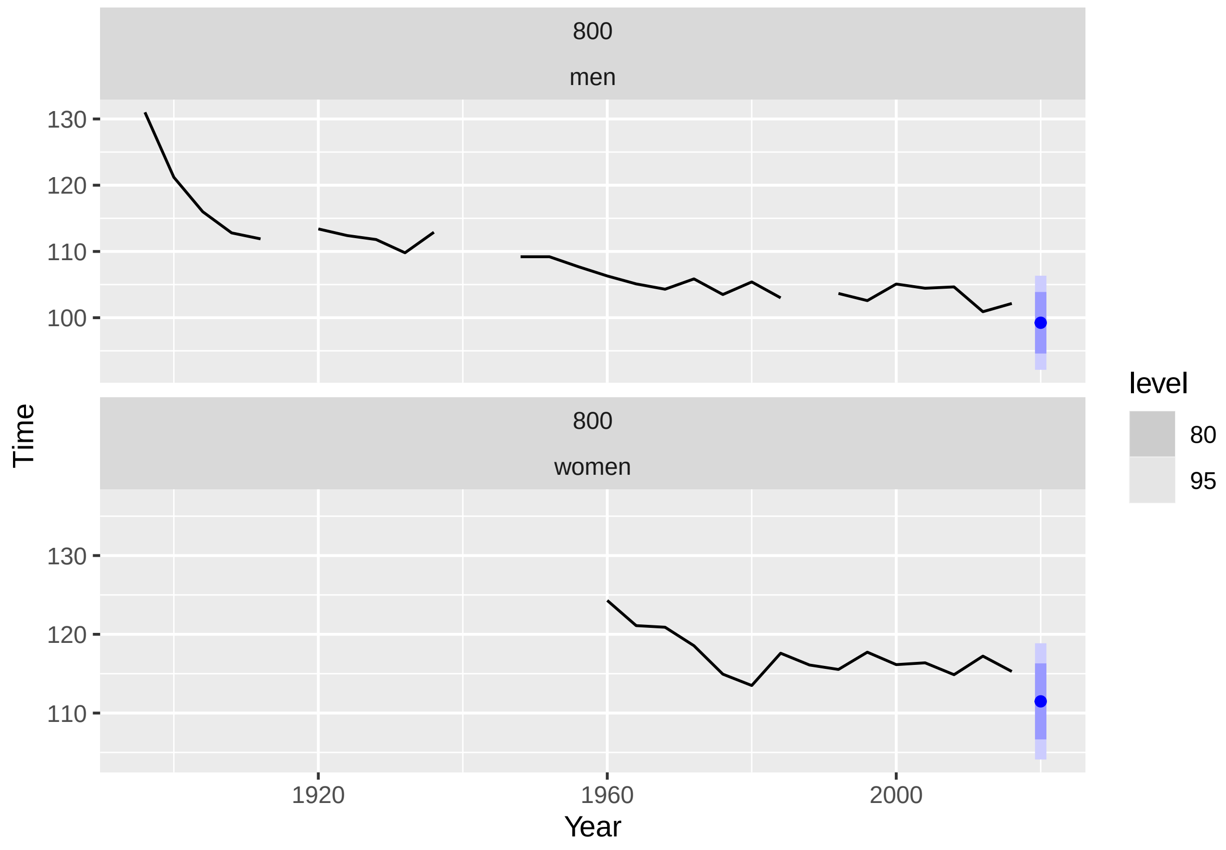
```r
fit %>%
  filter(Length == 200) %>%
  forecast(
    new_data(olympic_running, 1) %>%
      mutate(Year = 2020)
  ) %>%
  autoplot(olympic_running)
```



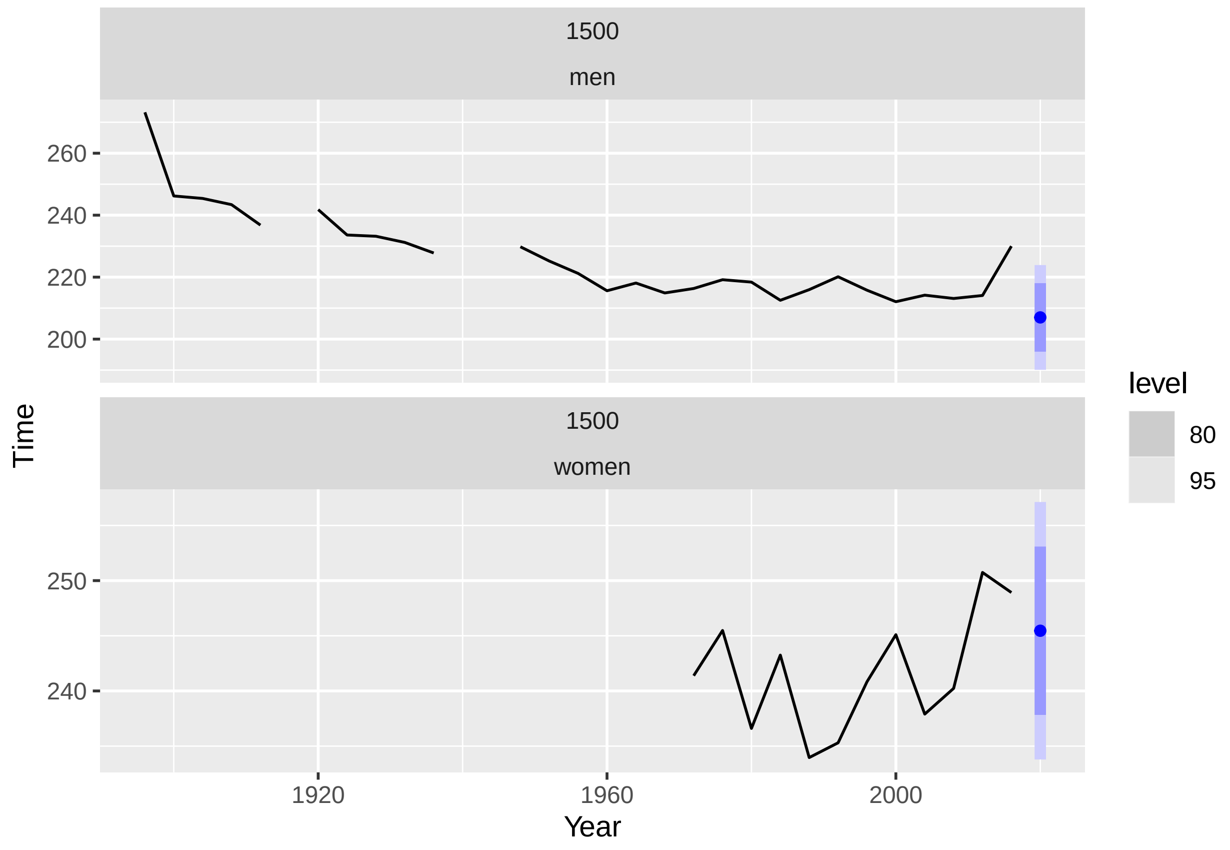
```
fit %>%
  filter(Length == 400) %>%
  forecast(
    new_data(olympic_running, 1) %>%
      mutate(Year = 2020)
  ) %>%
  autoplot(olympic_running)
```



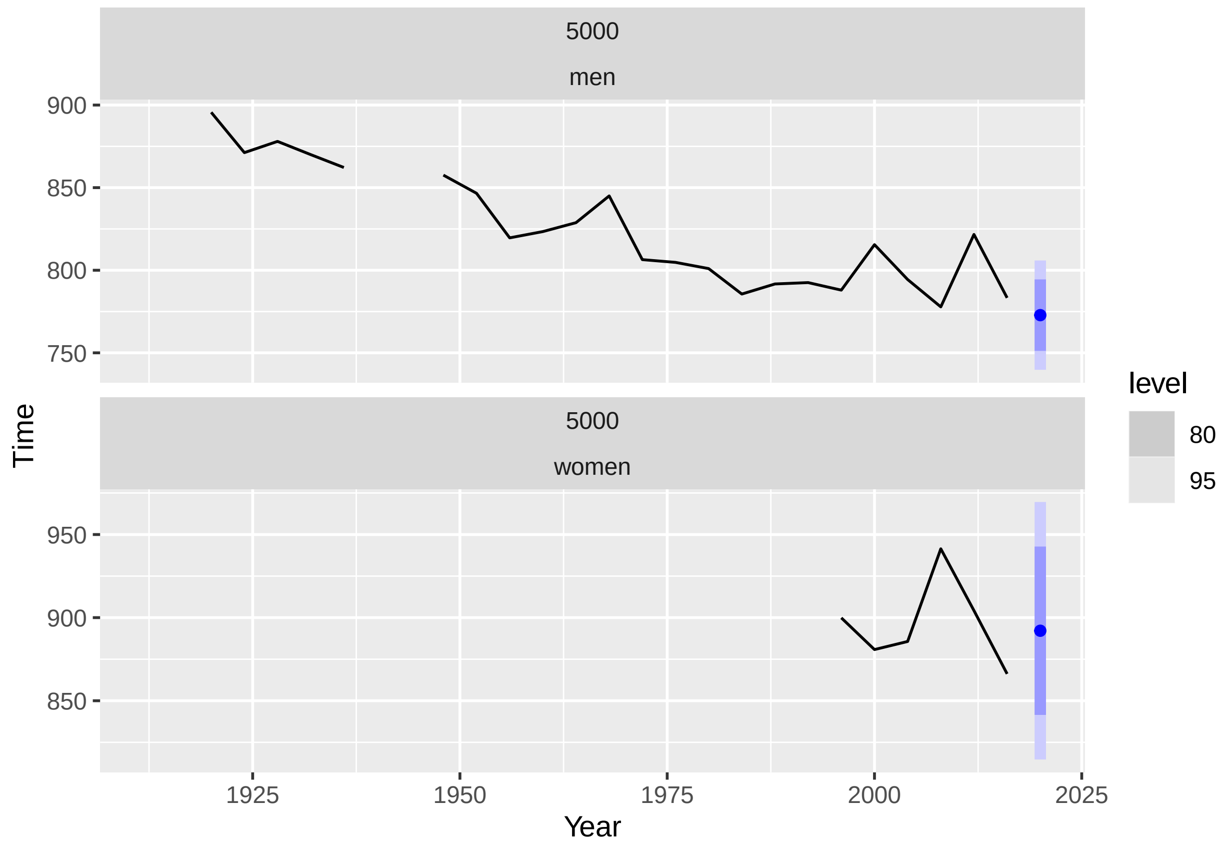
```
fit %>%
  filter(Length == 800) %>%
  forecast(
    new_data(olympic_running, 1) %>%
      mutate(Year = 2020)
  ) %>%
  autoplot(olympic_running)
```



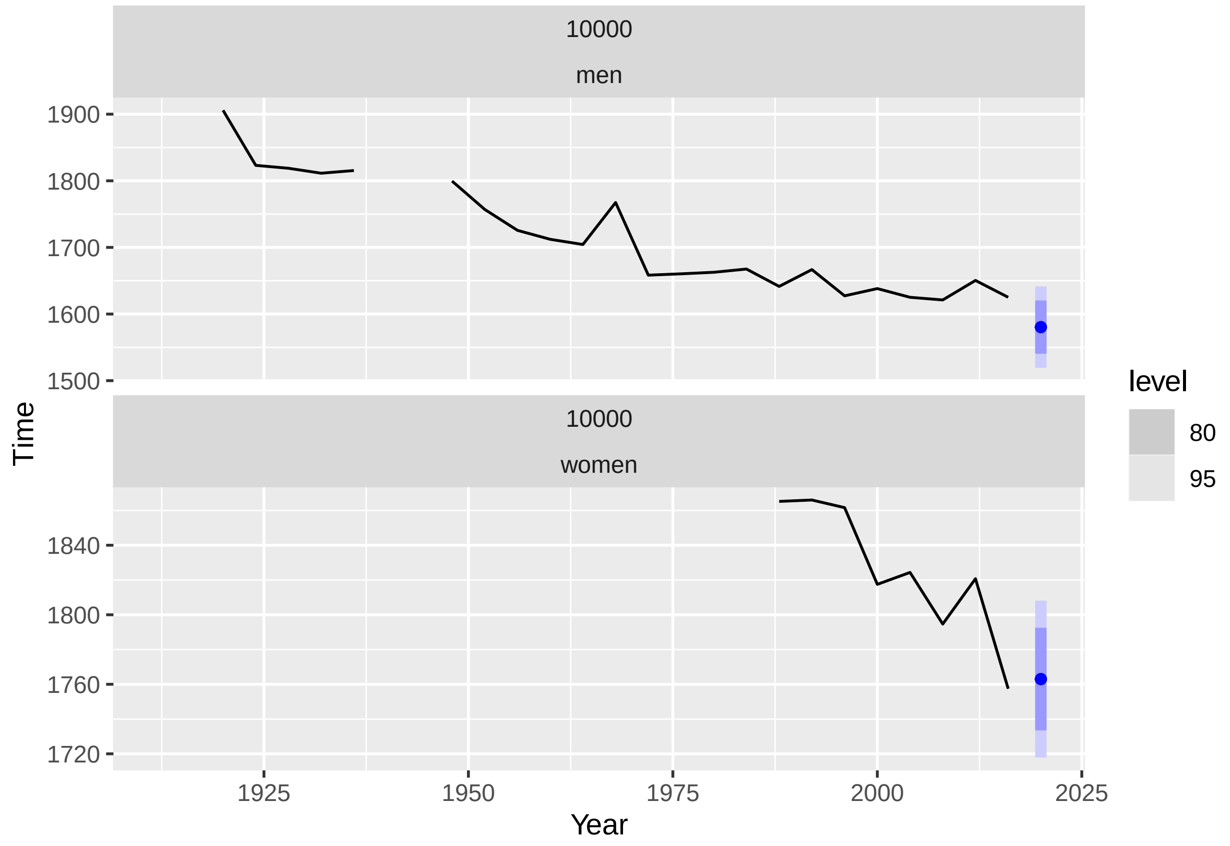
```
fit %>%
  filter(Length == 1500) %>%
  forecast(
    new_data(olympic_running, 1) %>%
      mutate(Year = 2020)
  ) %>%
  autoplot(olympic_running)
```



```
fit %>%
  filter(Length == 5000) %>%
  forecast(
    new_data(olympic_running, 1) %>%
      mutate(Year = 2020)
  ) %>%
  autoplot(olympic_running)
```



```
fit %>%
  filter(Length == 10000) %>%
  forecast(
    new_data(olympic_running, 1) %>%
      mutate(Year = 2020)
  ) %>%
  autoplot(olympic_running)
```



This are the predictions using the year as the input variable to the model. Of course, the model is very poor, since it only takes the year, meaning that as the year goes on the time will decrease even into negative values, which makes no sense, what will probably happened is that the values will start to stabilize at some point, we can actually start to see that on some of the data.

7.10 exercise 3

$$\log y = \beta_0 + \beta_1 \log x + e$$

$\beta_1 = \frac{dy}{dx} * \frac{x}{y}$

$\beta_1 * \frac{dx}{x} = \frac{dy}{y}$

7.10 exercise 4

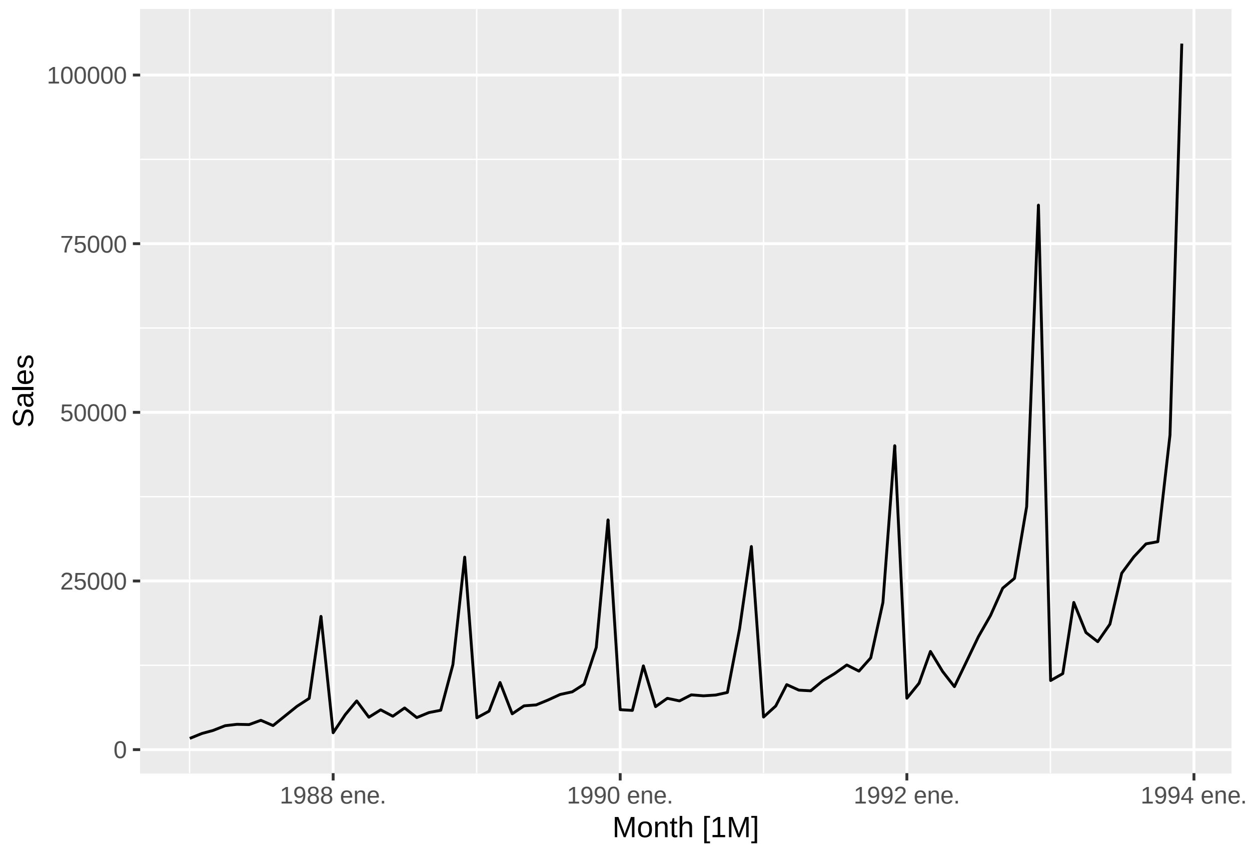
a)

```
souvenirs
```

```
## # A tsibble: 84 x 2 [1M]
##         Month Sales
##         <mth> <dbl>
##  1 1987 ene. 1665.
##  2 1987 feb. 2398.
##   3 1987 mar. 2841.
##   4 1987 abr. 3547.
##   5 1987 may. 3753.
##   6 1987 jun. 3715.
##   7 1987 jul. 4350.
##   8 1987 ago. 3566.
##   9 1987 sep. 5022.
##  10 1987 oct. 6423.
## # ... with 74 more rows
```

```
souvenirs %>%
  autoplot()
```

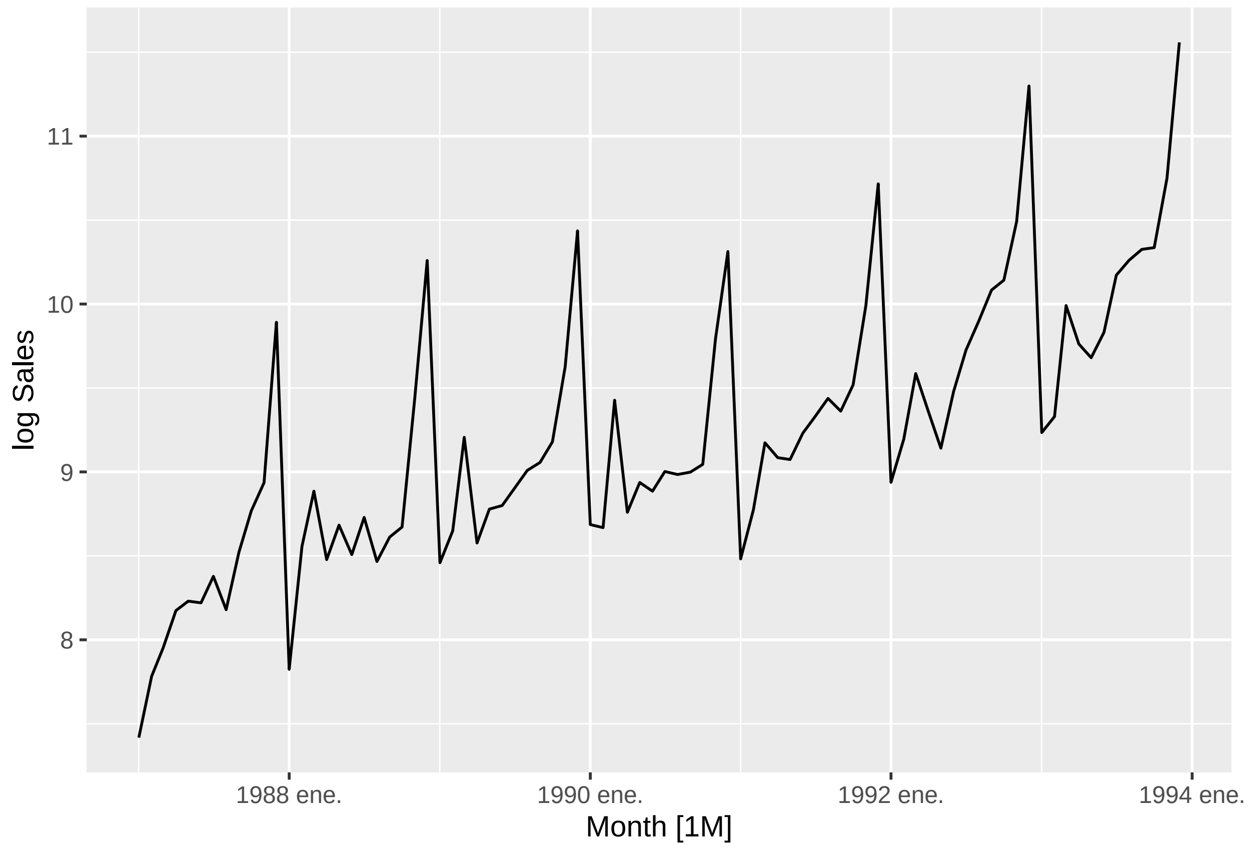
```
## Plot variable not specified, automatically selected '.vars = Sales'
```



We can see the seasonal component on the winter months for Christmas, we can also see a smaller peak in March for the surfing festival, except for 1987. Also, since the shop has been expanding we can see that the sells increase over time.

b)

```
souvenirs %>%
  mutate(log_Sales = log(Sales)) %>%
  autoplot(log_Sales)+ ylab("log Sales")
```



We can see that by taking log, the data looks more linear over time

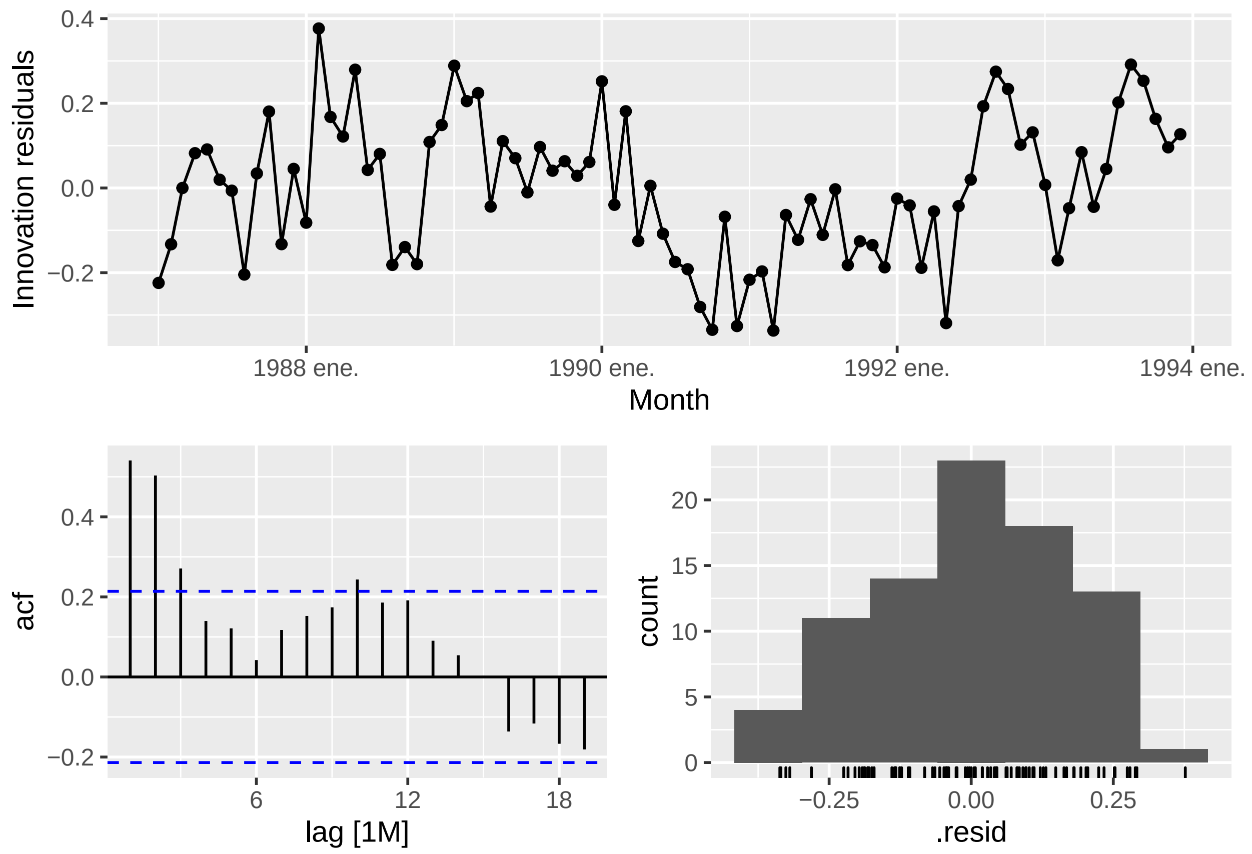
c)

```
souvenirs %>%
  mutate(surfing_festival = case_when(
    (month(Month)==3 & year(Month)>=1988) ~ 1,
    (month(Month)!=3 | year(Month)==1987) ~ 0 ) )  -> new_souvenirs

fit <- new_souvenirs %>%
  model(TSLM(log(Sales) ~ trend() + season() + surfing_festival))


report(fit)
```

```
## Series: Sales
## Model: TSLM
## Transformation: log(Sales)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.336727 -0.127571  0.002568  0.109106  0.376714
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       7.6196670  0.0742471 102.626  < 2e-16 ***
## trend()           0.0220198  0.0008268  26.634  < 2e-16 ***
## season()year2    0.2514168  0.0956790   2.628 0.010555 *
## season()year3    0.2660828  0.1934044   1.376 0.173275
## season()year4    0.3840535  0.0957075   4.013 0.000148 ***
## season()year5    0.4094870  0.0957325   4.277 5.88e-05 ***
## season()year6    0.4488283  0.0957647   4.687 1.33e-05 ***
## season()year7    0.6104545  0.0958039   6.372 1.71e-08 ***
## season()year8    0.5879644  0.0958503   6.134 4.53e-08 ***
## season()year9    0.6693299  0.0959037   6.979 1.36e-09 ***
## season()year10   0.7473919  0.0959643   7.788 4.48e-11 ***
## season()year11   1.2067479  0.0960319  12.566  < 2e-16 ***
## season()year12   1.9622412  0.0961066  20.417  < 2e-16 ***
## surfing_festival 0.5015151  0.1964273   2.553 0.012856 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.179 on 70 degrees of freedom
## Multiple R-squared: 0.9567,  Adjusted R-squared: 0.9487
## F-statistic:   119 on 13 and 70 DF, p-value: < 2.22e-16
```

d)

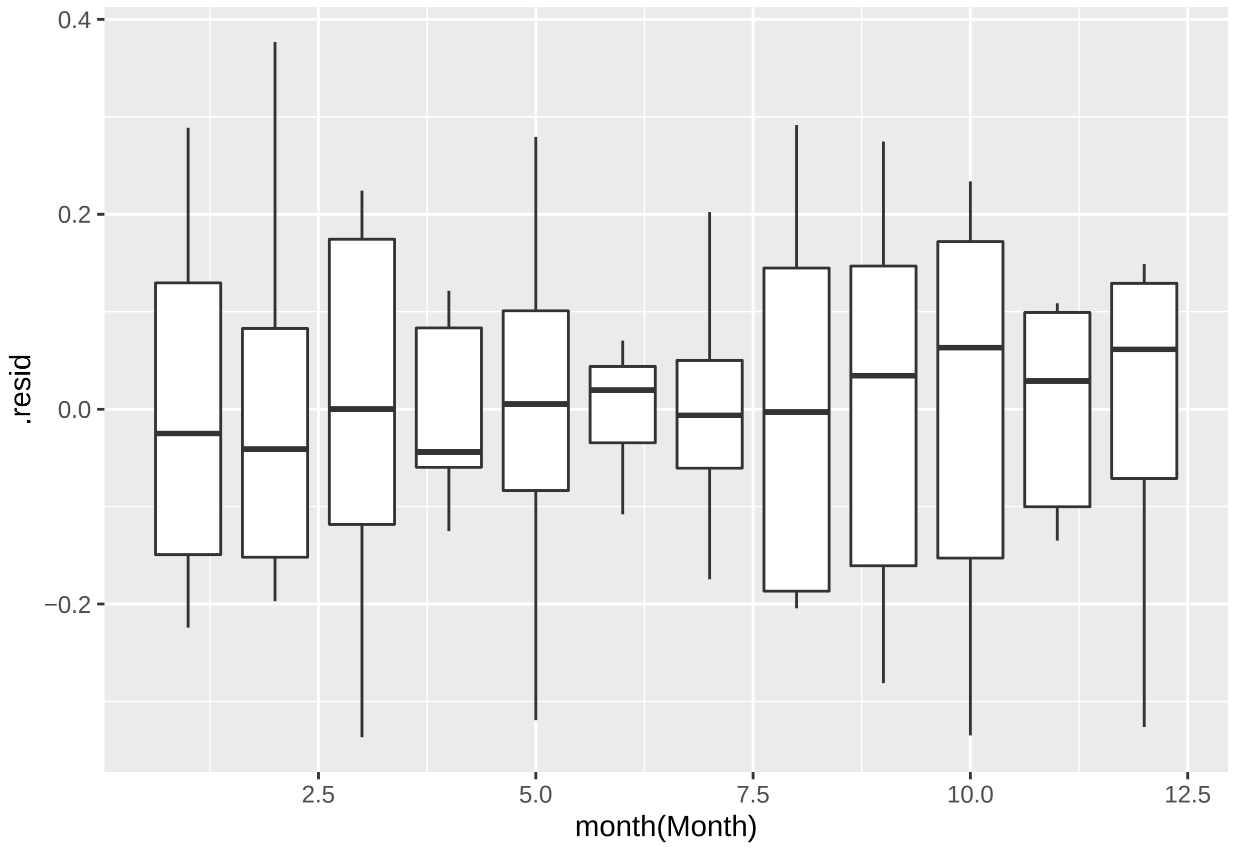
```
fit %>% gg_tsresiduals()
```



We can see that the residuals are not perfect, they are skewed to the positives size, we can also see that there are lag on the first 3 values, and then on the 10º value. Meaning that there are still some relations that we haven't capture.

e)

```
residuals(fit) %>%
  ggplot(aes(x = month(Month), y = .resid, group=month(Month))) +
    geom_boxplot()
```



We can see that residuals are really similar between all months, of course we can see that some months residuals are really grouped by, like month April, June, July while others have higher dispersion.

f)

```
report(fit)
```

```
## Series: Sales
## Model: TSLM
## Transformation: log(Sales)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.336727 -0.127571  0.002568  0.109106  0.376714
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      7.6196670  0.0742471 102.626  < 2e-16 ***
## trend()          0.0220198  0.0008268  26.634  < 2e-16 ***
## season()year2    0.2514168  0.0956790   2.628 0.010555 *
## season()year3    0.2660828  0.1934044   1.376 0.173275
## season()year4    0.3840535  0.0957075   4.013 0.000148 ***
## season()year5    0.4094870  0.0957325   4.277 5.88e-05 ***
## season()year6    0.4488283  0.0957647   4.687 1.33e-05 ***
## season()year7    0.6104545  0.0958039   6.372 1.71e-08 ***
## season()year8    0.5879644  0.0958503   6.134 4.53e-08 ***
## season()year9    0.6693299  0.0959037   6.979 1.36e-09 ***
## season()year10   0.7473919  0.0959643   7.788 4.48e-11 ***
## season()year11   1.2067479  0.0960319  12.566  < 2e-16 ***
## season()year12   1.9622412  0.0961066  20.417  < 2e-16 ***
## surfing_festival 0.5015151  0.1964273   2.553 0.012856 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.179 on 70 degrees of freedom
## Multiple R-squared: 0.9567,   Adjusted R-squared: 0.9487
## F-statistic:   119 on 13 and 70 DF, p-value: < 2.22e-16
```

The trend has a positive relation, although it is a really small value compare with other ones. In the season component, we can see that the last two months have really high coefficients above 1.0, we can also see that the surfing festival has a positive coefficients of 0.5

g)

```r
augment(fit) %>%
  features(.innov, ljung_box, lag = 10, dof = 0)
```

```
## # A tibble: 1 x 3
##   .model                                                 lb_stat lb_pvalue
##   <chr>                                                    <dbl>     <dbl>
## 1 TSLM(log(Sales) ~ trend() + season() + surfing_festival)    69.8  4.89e-11
```

Since the Q* have a really high value of 69, we can say that the correlation do not come from a white noise.

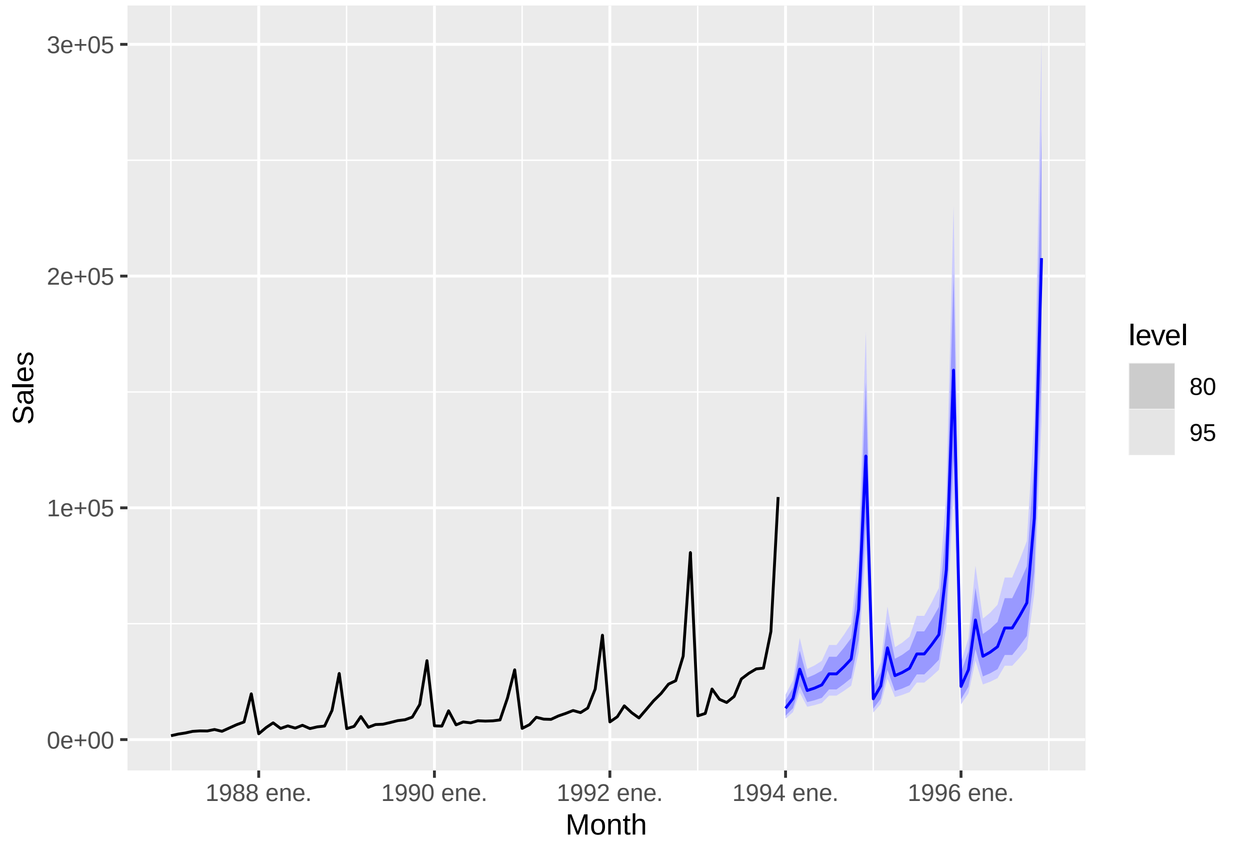
h)

```r
vals <- rep(0, 36)
yyy <- rep(0, 36)
mmm <- rep(0, 36)
for(i in 1:36){
  if(i %% 12 == 3){
    vals[i] <- 1
  }
  yyy[i] = 1994 + as.integer((i-1)/12)
  mmm[i] = ((i-1) %% 12) + 1
}

df <- data.frame(unlist(yyy), unlist(mmm), unlist(vals))
names(df) <- c("year", "month", "surfing_festival")

df %>%
  mutate(Month = yearmonth(paste(year, " ", month))) %>%
  as_tsibble(index = Month) %>%
  select(Month,surfing_festival) -> future_data
```

```r
future_data
```

```
## # A tsibble: 36 x 2 [1M]
##        Month surfing_festival
##        <mth>            <dbl>
##  1 1994 ene.                0
##  2 1994 feb.                0
##  3 1994 mar.                1
##  4 1994 abr.                0
##  5 1994 may.                0
##  6 1994 jun.                0
##  7 1994 jul.                0
##  8 1994 ago.                0
##  9 1994 sep.                0
## 10 1994 oct.                0
## # ... with 26 more rows
```

```r
fit %>%
  forecast(future_data) %>%
  autoplot(souvenirs)
```



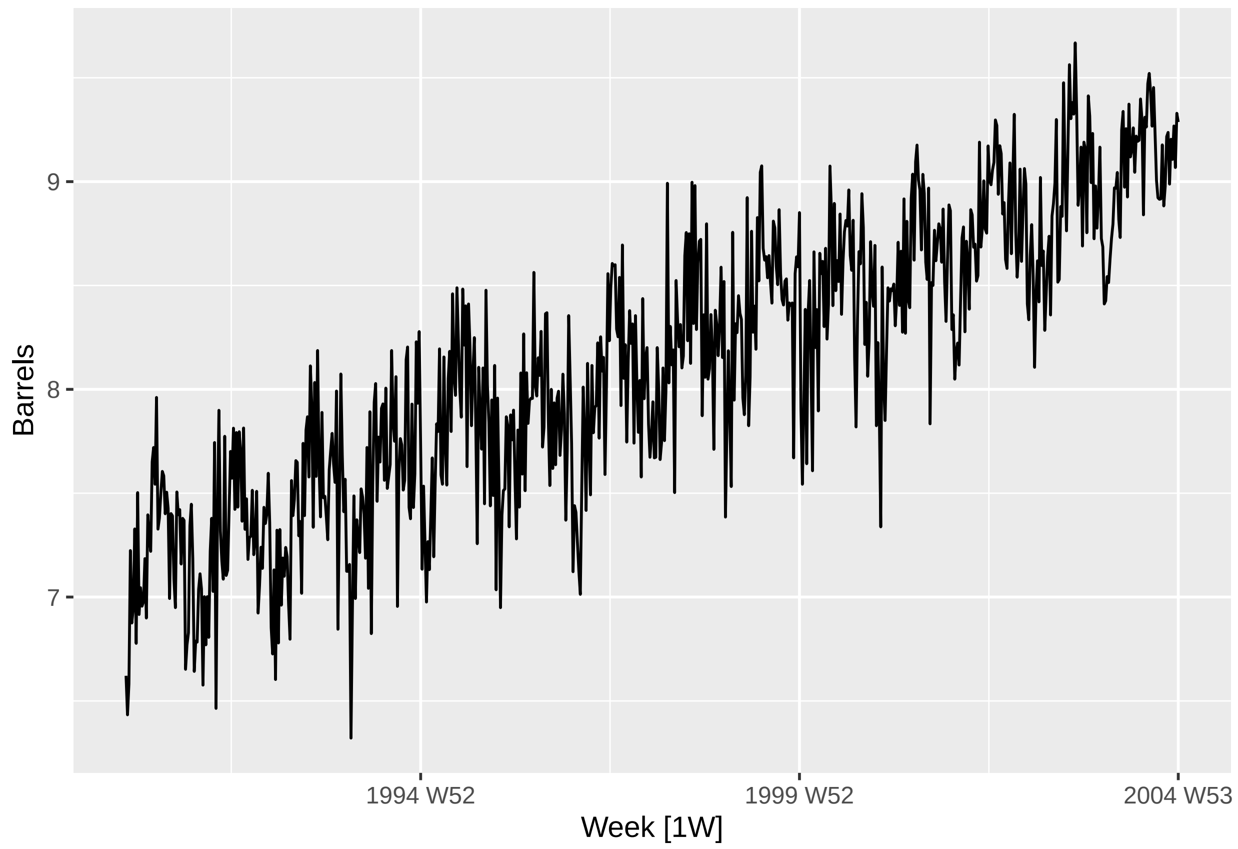
i) Since we have built the model using the log transformation we can see that the predictions reflects the data really good, but other types of transformations can be tested in order to get a better knowledge of the data

7.10 exercise 5

```
us_gasoline %>%
  filter(year(Week)<=2004) -> us_gasoline_1991_2004

us_gasoline_1991_2004
```

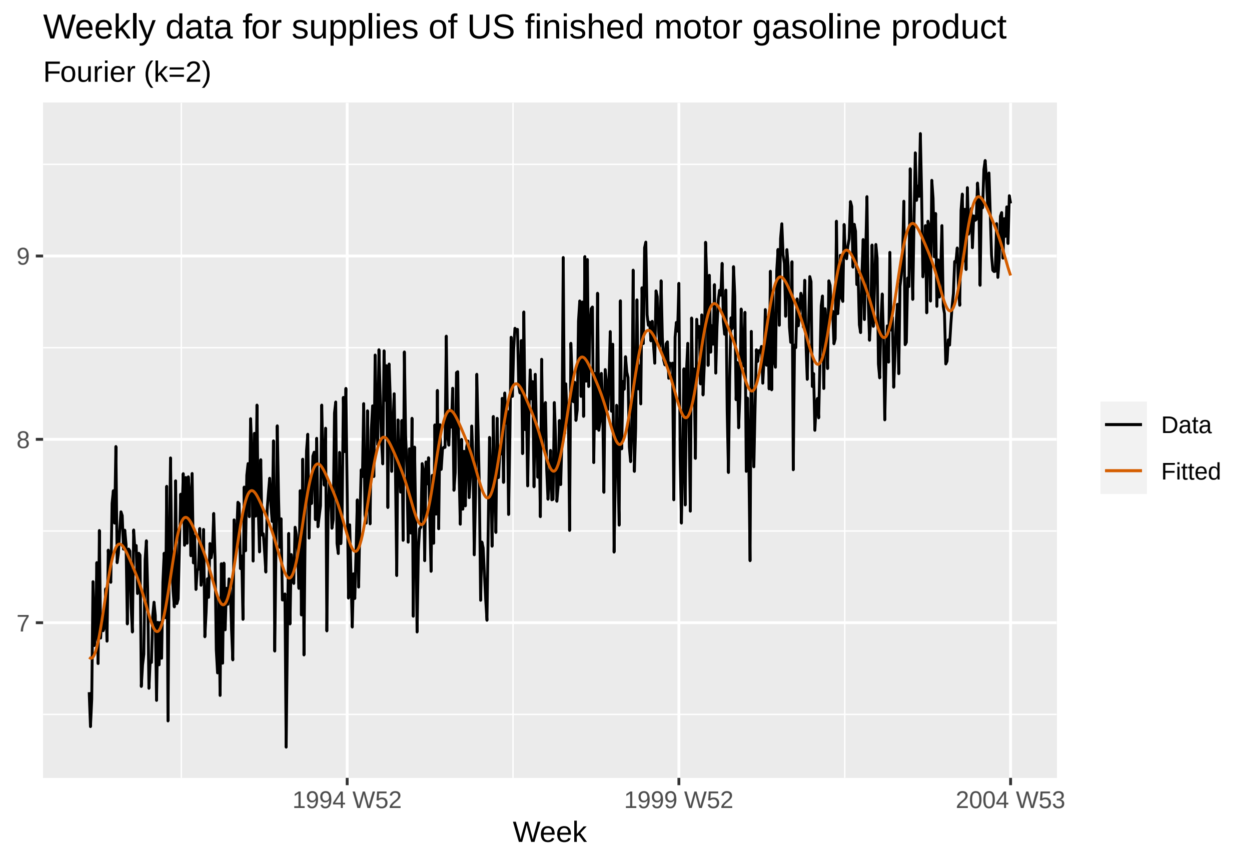
```
## # A tsibble: 726 x 2 [1W]
##        Week Barrels
##      <week>   <dbl>
##  1 1991 W06    6.62
##  2 1991 W07    6.43
##  3 1991 W08    6.58
##  4 1991 W09    7.22
##  5 1991 W10    6.88
##  6 1991 W11    6.95
##  7 1991 W12    7.33
##  8 1991 W13    6.78
##  9 1991 W14    7.50
## 10 1991 W15    6.92
## # ... with 716 more rows
```

```
us_gasoline_1991_2004 %>%
  autoplot(Barrels)
```

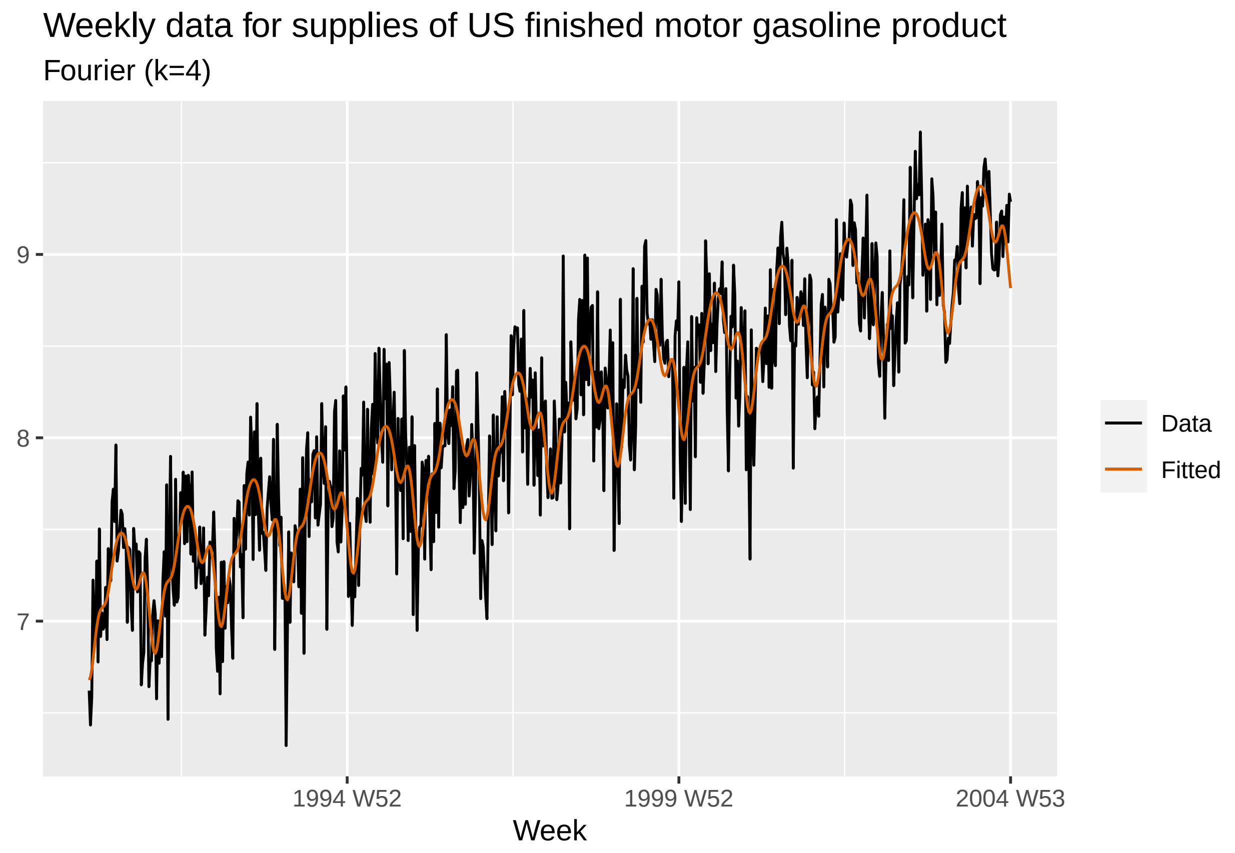


a)

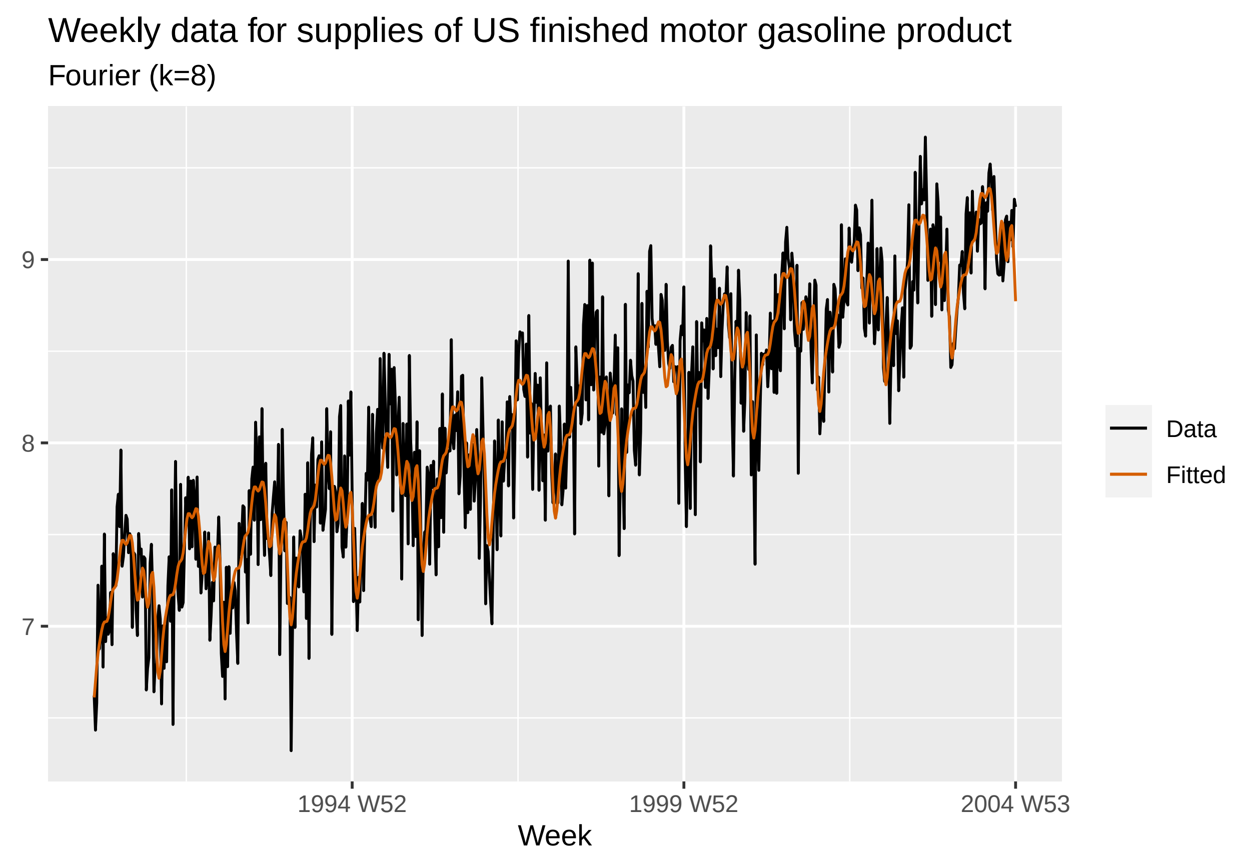
```r
fourier2_us_gasoline_1991_2004 <- us_gasoline_1991_2004 %>%
  model(f2 = TSLM(Barrels ~ trend() + fourier(K = 2)))
#report(fourier2_us_gasoline_1991_2004)
augment(fourier2_us_gasoline_1991_2004) %>%
  ggplot(aes(x = Week)) +
  geom_line(aes(y = Barrels, colour = "Data")) +
  geom_line(aes(y = .fitted, colour = "Fitted")) +
  labs(y = NULL,
    title = "Weekly data for supplies of US finished motor gasoline product",
    subtitle = "Fourier (k=2)"
  ) +
  scale_colour_manual(values=c(Data="black",Fitted="#D55E00")) +
  guides(colour = guide_legend(title = NULL))
```



```r
fourier2_us_gasoline_1991_2004 <- us_gasoline_1991_2004 %>%
  model(f2 = TSLM(Barrels ~ trend() + fourier(K = 4)))
#report(fourier2_us_gasoline_1991_2004)
augment(fourier2_us_gasoline_1991_2004) %>%
  ggplot(aes(x = Week)) +
  geom_line(aes(y = Barrels, colour = "Data")) +
  geom_line(aes(y = .fitted, colour = "Fitted")) +
  labs(y = NULL,
    title = "Weekly data for supplies of US finished motor gasoline product",
    subtitle = "Fourier (k=4)"
  ) +
  scale_colour_manual(values=c(Data="black",Fitted="#D55E00")) +
  guides(colour = guide_legend(title = NULL))
```



```
fourier2_us_gasoline_1991_2004 <- us_gasoline_1991_2004 %>%
  model(f2 = TSLM(Barrels ~ trend() + fourier(K = 8)))
augment(fourier2_us_gasoline_1991_2004) %>%
  ggplot(aes(x = Week)) +
  geom_line(aes(y = Barrels, colour = "Data")) +
  geom_line(aes(y = .fitted, colour = "Fitted")) +
  labs(y = NULL,
    title = "Weekly data for supplies of US finished motor gasoline product",
    subtitle = "Fourier (k=8)"
  ) +
  scale_colour_manual(values=c(Data="black",Fitted="#D55E00")) +
  guides(colour = guide_legend(title = NULL))
```



We can see that as the Fourier term K increases the more detail has the fitted line, this is because the parameter k indicates the number of sines and cosines used in the Fourier series. By increasing the number of sines and cosines we can get a series with more details, and this is what we can see on the graphs.

b)

```
fourier_us_gasoline_1991_2004 <- us_gasoline_1991_2004 %>%
  model(f2 = TSLM(Barrels ~ trend() + fourier(K=2)),
        f3 = TSLM(Barrels ~ trend() + fourier(K=3)),
        f4 = TSLM(Barrels ~ trend() + fourier(K=4)),
        f5 = TSLM(Barrels ~ trend() + fourier(K=5)),
        f6 = TSLM(Barrels ~ trend() + fourier(K=6)),
        f7 = TSLM(Barrels ~ trend() + fourier(K=7)),
        f8 = TSLM(Barrels ~ trend() + fourier(K=8)),
        f9 = TSLM(Barrels ~ trend() + fourier(K=9)),
        f10 = TSLM(Barrels ~ trend() + fourier(K=10)),
        f11 = TSLM(Barrels ~ trend() + fourier(K=11)),
        f12 = TSLM(Barrels ~ trend() + fourier(K=12)),
        f13 = TSLM(Barrels ~ trend() + fourier(K=13)),
        f14 = TSLM(Barrels ~ trend() + fourier(K=14)),
        f15 = TSLM(Barrels ~ trend() + fourier(K=15)),
        f16 = TSLM(Barrels ~ trend() + fourier(K=16)),
        f17 = TSLM(Barrels ~ trend() + fourier(K=17)),
        f18 = TSLM(Barrels ~ trend() + fourier(K=18)),
        f19 = TSLM(Barrels ~ trend() + fourier(K=19)),
        f20 = TSLM(Barrels ~ trend() + fourier(K=20)),
        f21 = TSLM(Barrels ~ trend() + fourier(K=21)),
```

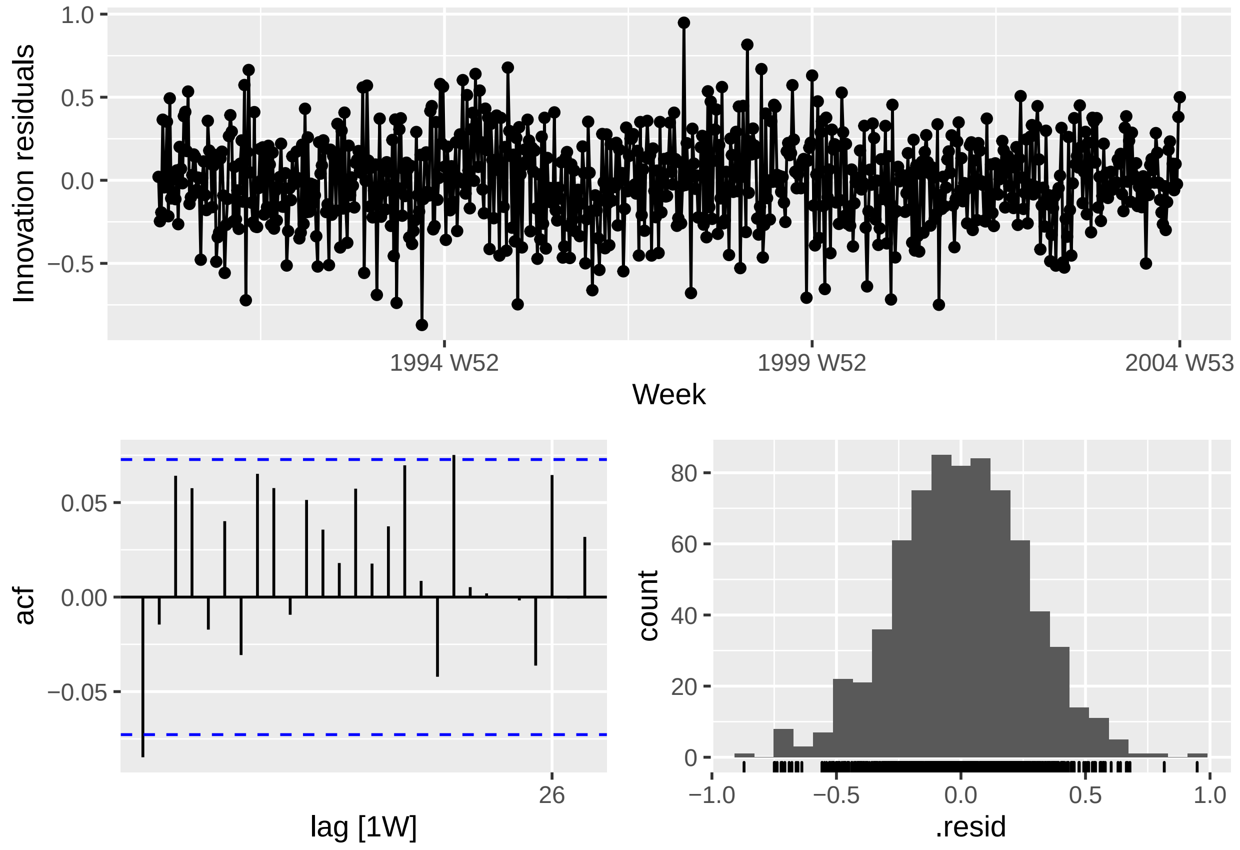
```r
        f22 = TSLM(Barrels ~ trend() + fourier(K=22)),
        f23 = TSLM(Barrels ~ trend() + fourier(K=23)),
        f24 = TSLM(Barrels ~ trend() + fourier(K=24)),
        f25 = TSLM(Barrels ~ trend() + fourier(K=25)),
        f26 = TSLM(Barrels ~ trend() + fourier(K=26)))




glance(fourier_us_gasoline_1991_2004) %>%
  select(.model, CV, AICc) %>%
  arrange(AICc)
```

```
## # A tibble: 25 x 3
##    .model     CV   AICc
##    <chr>   <dbl>  <dbl>
##  1 f7     0.0740 -1887.
##  2 f11    0.0742 -1885.
##  3 f8     0.0743 -1885.
##  4 f12    0.0742 -1884.
##  5 f6     0.0744 -1884.
##  6 f9     0.0745 -1882.
##  7 f10    0.0746 -1881.
##  8 f13    0.0745 -1881.
##  9 f14    0.0748 -1879.
## 10 f15    0.0750 -1876.
## # ... with 15 more rows
```

We can see that the model with the lower CV is the model with 7 Fourier terms, followed by 11, 8, 12…

c)

```r
fourier_us_gasoline_1991_2004 %>%
  select(f7) %>%
  gg_tsresiduals()
```



```
augment(fourier_us_gasoline_1991_2004) %>%
  features(.innov, ljung_box, lag=10, dof=5) %>%
  filter(.model == 'f7')
```

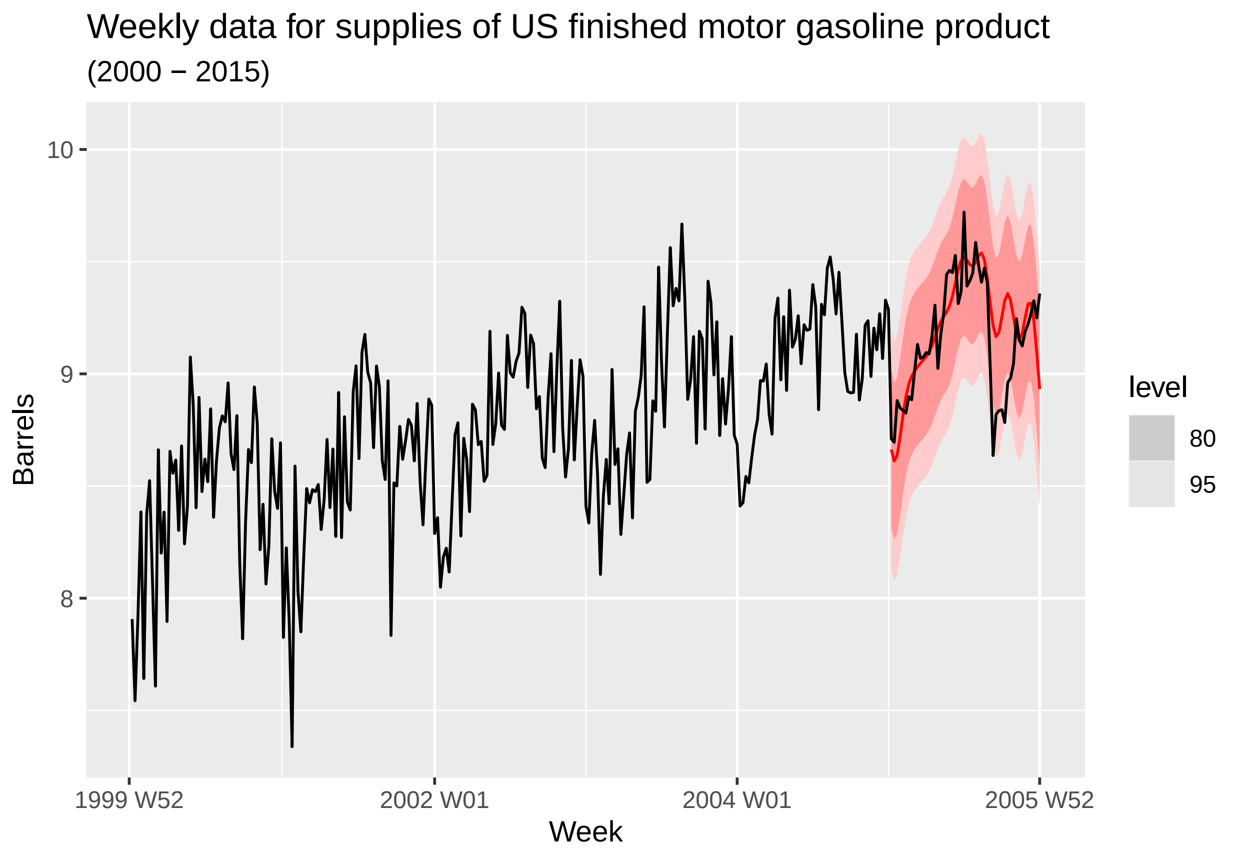
```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 f7        18.6   0.00233
```

We can see on the time plot the residuals over time, that the values are constant over the time around the 0 value. In the histogram we can see that it follows a normal distribution. We can see that there are some lags at 1 and 20, but just by a little bit.

```
fourier_us_gasoline_1991_2004 %>%
  select(f7) %>%
  forecast(h = 52) -> fc_us_gasoline

fc_us_gasoline
```

```
## # A fable: 52 x 4 [1W]
## # Key:     .model [1]
##    .model     Week       Barrels .mean
##    <chr>    <week>        <dist> <dbl>
##  1 f7     2005 W01 N(8.7, 0.074)  8.66
##  2 f7     2005 W02 N(8.6, 0.074)  8.61
##  3 f7     2005 W03 N(8.6, 0.074)  8.64
##  4 f7     2005 W04 N(8.7, 0.074)  8.72
##  5 f7     2005 W05 N(8.8, 0.074)  8.81
##  6 f7     2005 W06 N(8.9, 0.074)  8.90
##  7 f7     2005 W07   N(9, 0.074)  8.95
##  8 f7     2005 W08   N(9, 0.074)  8.99
##  9 f7     2005 W09   N(9, 0.074)  9.01
## 10 f7     2005 W10   N(9, 0.074)  9.03
## # ... with 42 more rows
```

```
fc_us_gasoline %>%
  autoplot(us_gasoline %>% filter(year(Week)<=2005 & year(Week)>=2000)
           , colour = 'red') +
  labs(title = "Weekly data for supplies of US finished motor gasoline product",
       subtitle = "(2000 - 2015)") +
  guides(colour = guide_legend(title = "Forecast"))
```



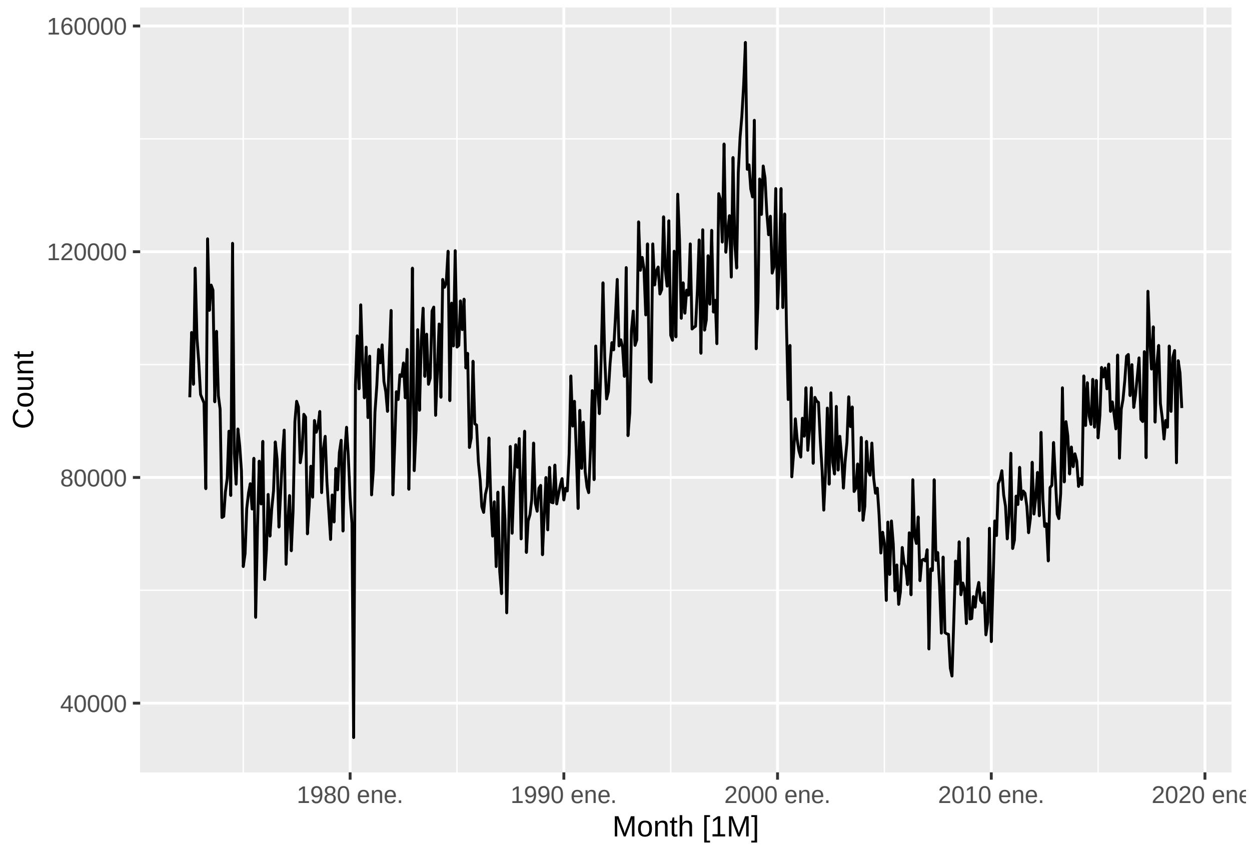
We can see that the forecast represented really good the data, except for some points, the end of the prediction is wrong, as the actual data decreased way more than the prediction. But overall is a good forecast.

8.8 exercise 1

```
aus_livestock %>%
  filter(Animal == 'Pigs' & State == 'Victoria') -> aus_pigs

aus_pigs
```

```
## # A tsibble: 558 x 4 [1M]
## # Key:       Animal, State [1]
##        Month Animal State     Count
##        <mth> <fct>  <fct>     <dbl>
##  1 1972 jul. Pigs   Victoria  94200
##  2 1972 ago. Pigs   Victoria 105700
##  3 1972 sep. Pigs   Victoria  96500
##  4 1972 oct. Pigs   Victoria 117100
##  5 1972 nov. Pigs   Victoria 104600
##  6 1972 dic. Pigs   Victoria 100500
##  7 1973 ene. Pigs   Victoria  94700
##  8 1973 feb. Pigs   Victoria  93900
##  9 1973 mar. Pigs   Victoria  93200
## 10 1973 abr. Pigs   Victoria  78000
## # ... with 548 more rows
```

```
aus_pigs %>%
  autoplot(Count)
```

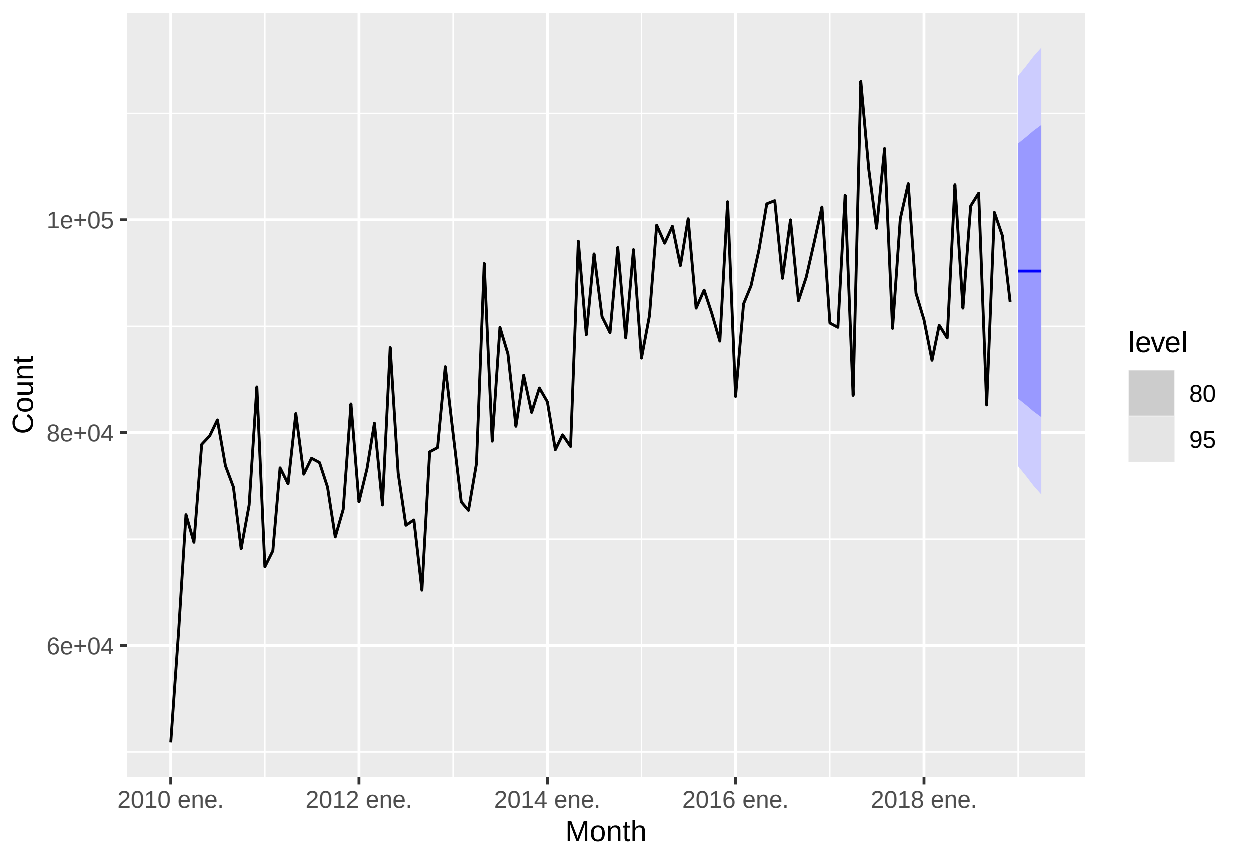


a)

```
fit_aus_pigs <- aus_pigs %>%
  model(ETS(Count ~ error("A") + trend("N") + season("N")))

report(fit_aus_pigs)
```

```
## Series: Count
## Model: ETS(A,N,N)
##   Smoothing parameters:
##     alpha = 0.3221247
##
##   Initial states:
##       l[0]
##   100646.6
##
##   sigma^2:  87480760
##
##       AIC     AICc      BIC
## 13737.10 13737.14 13750.07
```

```
fc_aus_pigs <- fit_aus_pigs %>%
  forecast(h = 4)

fc_aus_pigs %>%
  autoplot(aus_pigs %>% filter(year(Month)>=2010))
```



b)

```
aug_fit_aus_pigs <- augment(fit_aus_pigs)
standard_dev = sd(aug_fit_aus_pigs$.resid)
#standard_dev

mean_forecast1 = fc_aus_pigs$.mean[1]
#mean_forecast1

lower95 = mean_forecast1 - (1.96 * standard_dev)
upper95 = mean_forecast1 + (1.96 * standard_dev)

print("Our prediction interval: ")
```

```
## [1] "Our prediction interval: "
```

```
print(paste0("Lower Interval (95%): ", lower95))
```

```
## [1] "Lower Interval (95%): 76871.0124775157"
```

```
print(paste0("Upper Interval (95%): ", upper95))
```

```
## [1] "Upper Interval (95%): 113502.102384467"
```

```r
print("-------------------------")
```

```
## [1] "-------------------------"
```

```r
print("Prediction interval by R: ")
```

```
## [1] "Prediction interval by R: "
```

```r
fc_aus_pigs%>%
  hilo() %>%
  filter(month(Month)==1) %>% #Get the first prediction
  select(Animal, State, .model, Month, "95%") -> r_95_interval

r_95_interval$"95%"
```

```
## <hilo[1]>
## [1] [76854.79, 113518.3]95
```

We can see that the intervals are not the same. They are similar in the first 3 digits, but after that point they are different. This is because R uses a more precise method than the one, we applied. We used 1.96 but this is just an approximation of the real value.

8.8 exercise 2

```r
my_function <- function(y, alpha, level) {
  count=1
  predictions=c()
  for (y_i in y){
    if(count==1){
      #First Iter

      # Calculate value
      y_i_1 = alpha * y_i + (1-alpha) * level
      # save value
      predictions <- append(predictions, c(y_i_1))
    }else{
      #Not First Iter

      # Calculate value
      y_i_1 = alpha * y_i + (1-alpha) * y_i_1
      # save value
      predictions <- append(predictions, c(y_i_1))
    }
    count = count+1
  }
  #print(predictions)
  return(predictions[length(predictions)])
}
```

```r
#yy = c(1, 2, 3, 4, 5)
my_function(aus_pigs$Count, 0.3221247, 100646.6)
```

```
## [1] 95186.56
```

```r
(fc_aus_pigs %>%
  filter(month(Month)==1))$.mean
```

```
## [1] 95186.56
```

We can see that both values are the same, meaning that that the function worked.

8.8 exercise 3

```r
my_function2 <- function(pars = c(alpha, level), y) {
  alpha = pars[1]
  level = pars[2]

  count=1
  predictions=c()
  SSE = 0
  squared_errors=c()
  for (y_i in y){
    if(count==1){
      #First Iter

      # Calculate value
      y_i_1 = alpha * y_i + (1-alpha) * level
      # save value
      predictions <- append(predictions, c(y_i_1))
    }else{
      #Not First Iter

      squared_error = (y_i-y_i_1)^2
      squared_errors <- append(squared_errors, c(squared_error))
      SSE = SSE + squared_error

      # Calculate value
      y_i_1 = alpha * y_i + (1-alpha) * y_i_1
      # save value
      predictions <- append(predictions, c(y_i_1))



    }
    count = count+1
  }
  #print(predictions)
  return(SSE)
}

sum_squared_errors_res = my_function2( c(0.3221247, 100646.6), aus_pigs$Count)
sum_squared_errors_res
```

```
## [1] 48597744010
```

```
response = optim( par = c(0.5, 100), y = aus_pigs$Count, fn=my_function2)
```

```
al = response$par[1]
lev = response$par[2]
print(paste0("optimized Alpha: ",al))
```

```
## [1] "optimized Alpha: 0.322808965018238"
```

```
print(paste0("optimized level: ",lev))
```

```
## [1] "optimized level: 108010.568478638"
```

```
print("---------------------------------")
```

```
## [1] "---------------------------------"
```

```
print(paste0("True Alpha: ",0.3221247))
```

```
## [1] "True Alpha: 0.3221247"
```

```
print(paste0("True level: ",100646.6))
```

```
## [1] "True level: 100646.6"
```

We can see that the values are really close by, and they start to differ after 3 digits

8.8 exercise 4

```
my_function3 <- function(y) {
  #optimize
  response = optim( par = c(0.5, 100), y = y, fn=my_function2)
  alpha = response$par[1]
  level = response$par[2]

  #predict
  prediction = my_function(y, alpha, level)

  return(prediction)
}

print("My function: ")
```

```
## [1] "My function: "
```

```
my_function3(aus_pigs$Count)
```

```
## [1] 95185.57
```

```
print("Prediction from R: ")
```

```
## [1] "Prediction from R: "
```

```
(fc_aus_pigs %>%
  filter(month(Month)==1))$.mean
```

```
## [1] 95186.56
```

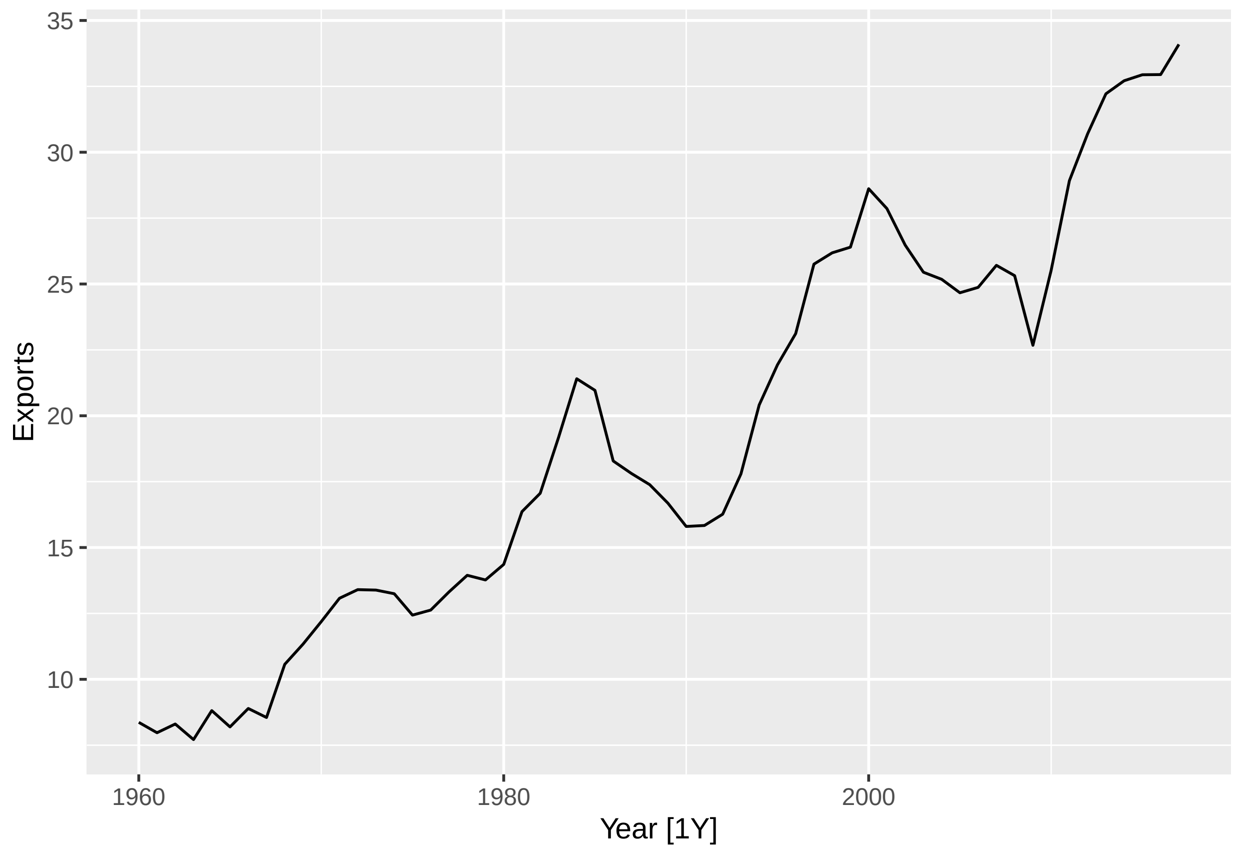
8.8 exercise 5

```
global_economy %>%
  filter(Country == "Spain") ->spain
spain
```

```
## # A tsibble: 58 x 9 [1Y]
## # Key:       Country [1]
##    Country Code   Year          GDP Growth   CPI Imports Exports Population
##    <fct>   <fct> <dbl>        <dbl>  <dbl> <dbl>   <dbl>   <dbl>      <dbl>
##  1 Spain   ESP    1960 12072126075.  NA     2.86    6.90    8.37   30455000
##  2 Spain   ESP    1961 13834300571.  11.8   2.88    8.60    7.97   30739250
##  3 Spain   ESP    1962 16138545209.   9.95  3.05   10.1     8.30   31023366
##  4 Spain   ESP    1963 19074913948.   9.60  3.31   10.8     7.71   31296651
##  5 Spain   ESP    1964 21343844644.   5.31  3.54   11.4     8.81   31609195
##  6 Spain   ESP    1965 24756958695.   6.25  4.01   13.1     8.19   31954292
##  7 Spain   ESP    1966 28721062242.   7.25  4.26   13.6     8.89   32283194
##  8 Spain   ESP    1967 31647119228.   4.34  4.54   11.9     8.55   32682947
##  9 Spain   ESP    1968 31475548481.   6.60  4.76   12.6    10.6    33113134
## 10 Spain   ESP    1969 36038711600.   8.91  4.86   13.1    11.3    33441054
## # ... with 48 more rows
```

a)

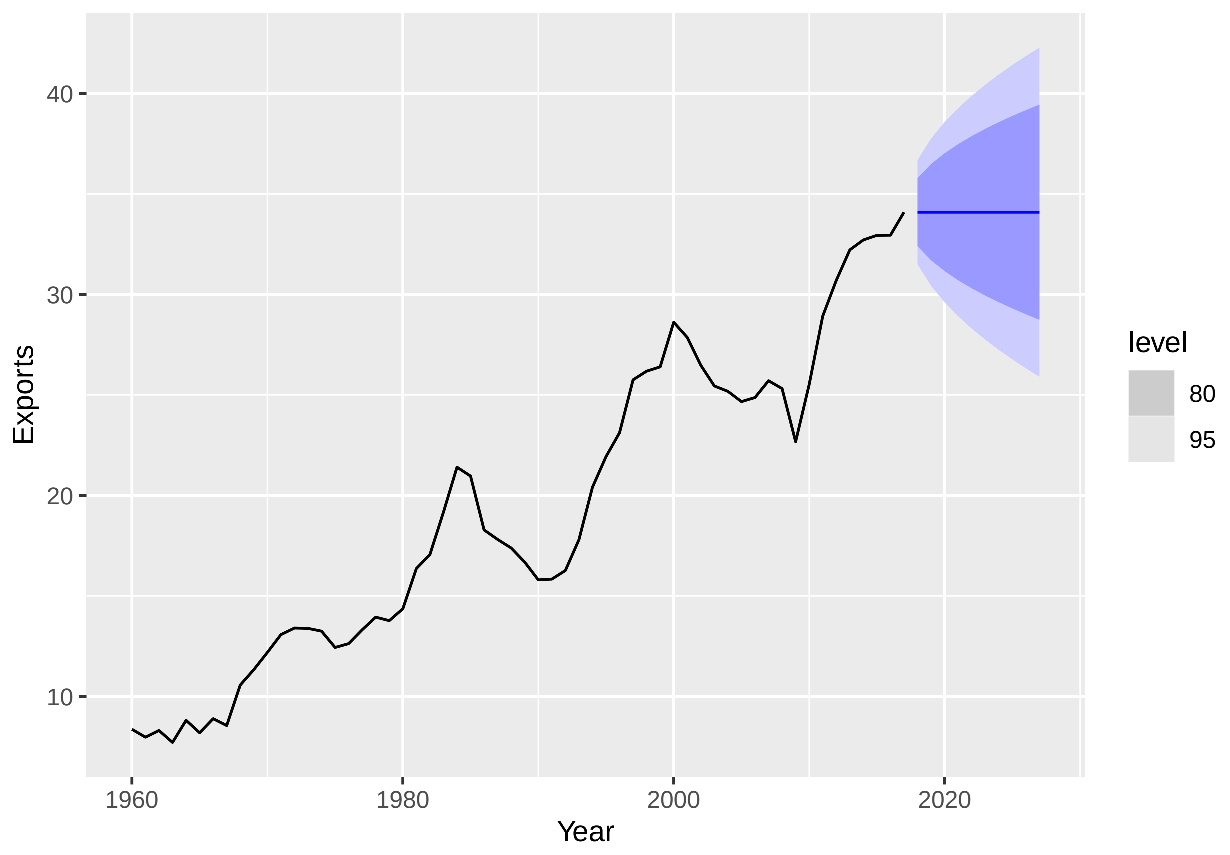
```
spain %>%
  autoplot(Exports)
```



We can see a positive trend, meaning that the values keep rising as the years goes on. We also can see raises and falls over the years in a somewhat cyclic pattern.

b)

```
fit_spain <- spain %>%
  model(ANN = ETS(Exports ~error("A") + trend("N") + season("N")))

fc_spain <- fit_spain %>%
  forecast(h=10)

fc_spain %>%
  autoplot(spain)
```



c)

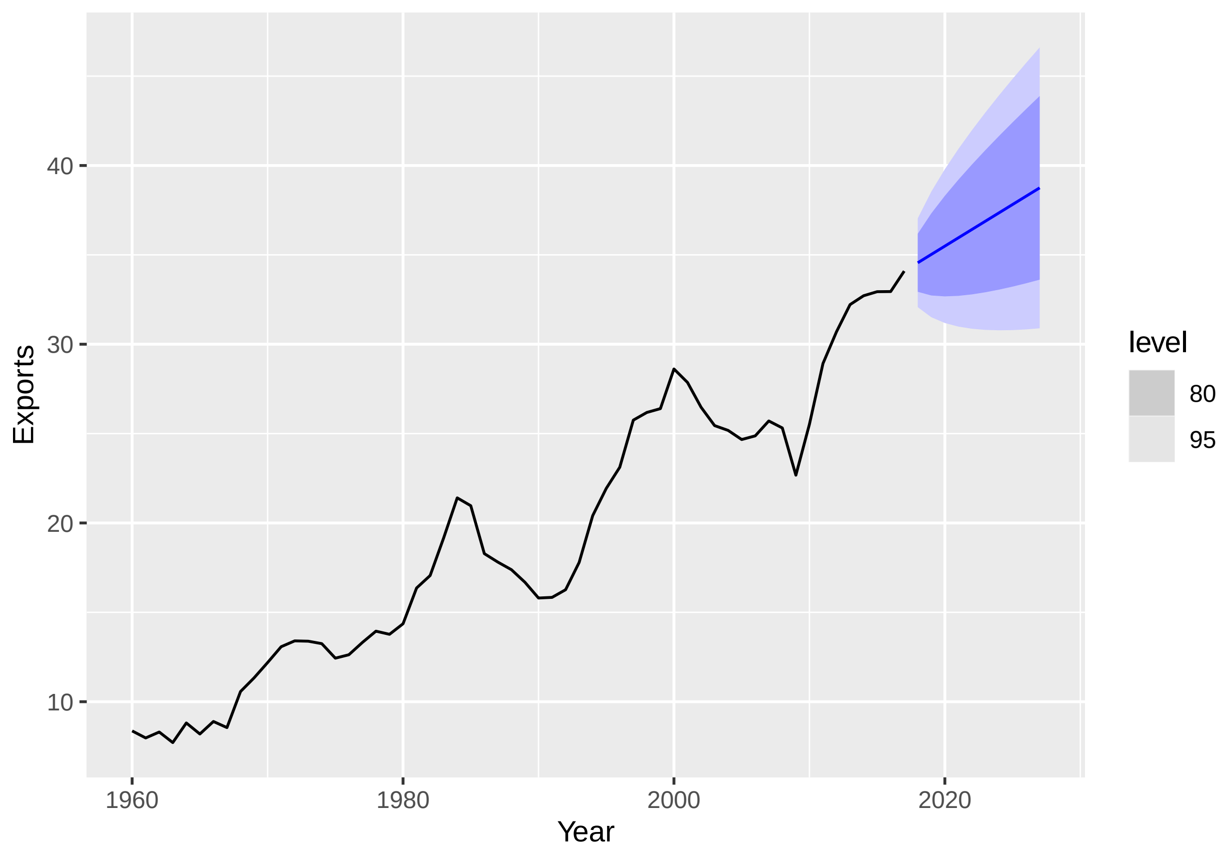
```
accuracy(fit_spain)
```

```
## # A tibble: 1 x 11
##   Country .model .type       ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <fct>   <chr>  <chr>    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Spain   ANN    Training 0.444  1.30 0.984  2.18  5.39 0.983 0.991 0.345
```

RMSE value: 1.298316

d)

```
fit_spain <- spain %>%
  model(AAN = ETS(Exports ~error("A") + trend("A") + season("N")))

fc_spain <- fit_spain %>%
  forecast(h=10)

fc_spain %>%
  autoplot(spain)
```



```
accuracy(fit_spain)
```

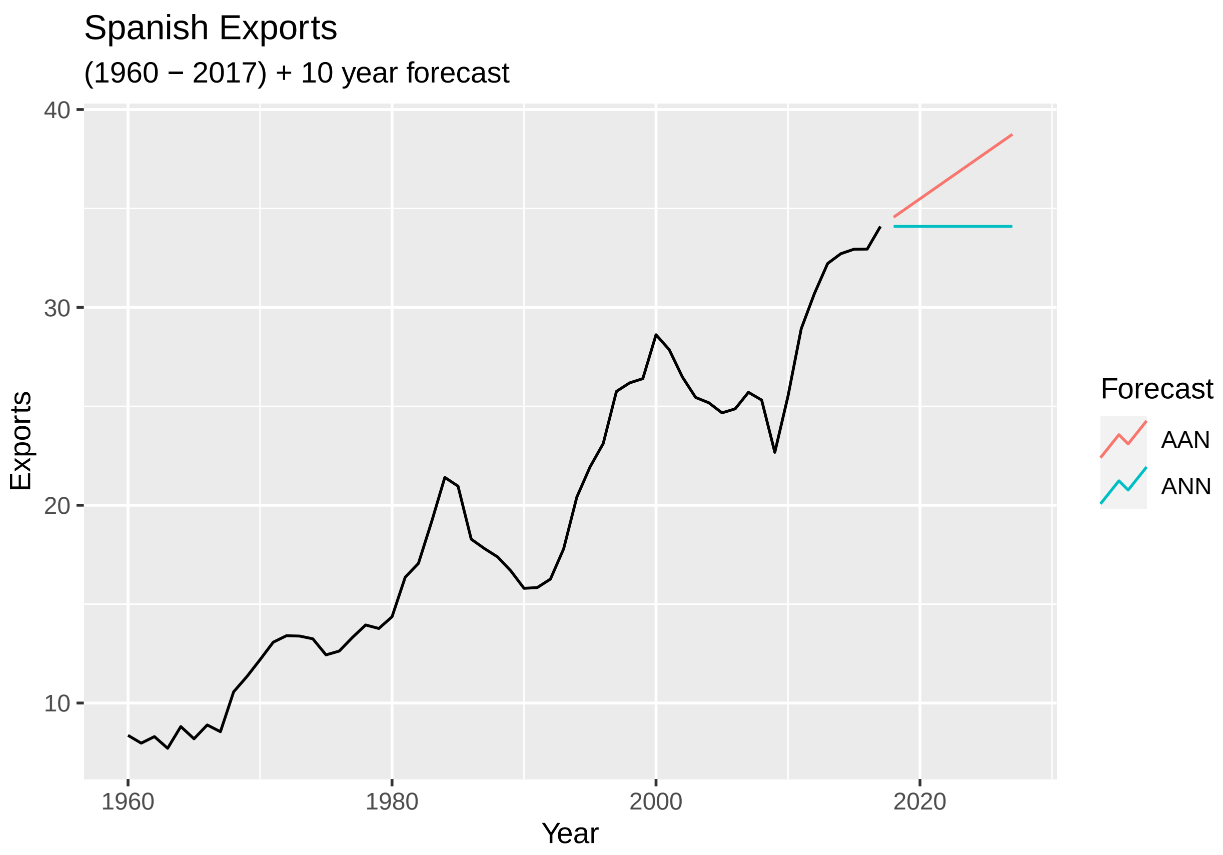
```
## # A tibble: 1 x 11
##   Country .model .type            ME  RMSE   MAE    MPE  MAPE  MASE RMSSE  ACF1
##   <fct>   <chr>  <chr>         <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Spain   AAN    Training -0.000967  1.22 0.955 -0.474  5.42 0.953 0.934 0.332
```

We can see that the RMSE for the AAN is 1.222947 while for the ANN is 1.298316, this is an increase of more than 6% from the AAN to ANN. The AAN method was able to capture the upward trend of the data, while the AAN was just a flat prediction. This is expected since the AAN incorporate the trend parameter.

e)

```
fit_spain <- spain %>%
  model(
    ANN = ETS(Exports ~error("A") + trend("N") + season("N")),
    AAN = ETS(Exports ~error("A") + trend("A") + season("N"))
  )

fc_spain <- fit_spain %>%
  forecast(h=10)

fc_spain %>%
  autoplot(spain, level=NULL) +
  labs(title = "Spanish Exports",
```

```r
      subtitle = "(1960 - 2017) + 10 year forecast") +
  guides(colour = guide_legend(title = "Forecast"))
```



The forecast from the AAN is probably better since it was able to capture the upward trend of the dataset. Having said this if the next years would behave like the downward part of the cycle the ANN method will probably perform best, since it is flat, but this will only hold while the cycle is downward, once it starts to increase the AAN will again perform better.

f)

```r
# ANN model:
print("########################################")
```

```
## [1] "########################################"
```

```r
print("################# ANN #################")
```

```
## [1] "################# ANN #################"
```

```r
print("########################################")
```

```
## [1] "########################################"
```

```r
#mean
ANN_mean = (fc_spain %>% filter(.model=='ANN'))$.mean[1]

#standard dev of residuals
ANN_standard_dev = sd((augment(fit_spain)%>% filter(.model=='ANN'))$.resid)

lower95 = ANN_mean - (1.96 * ANN_standard_dev)
upper95 = ANN_mean + (1.96 * ANN_standard_dev)

print("Our prediction interval (ANN): ")
```

```
## [1] "Our prediction interval (ANN): "
```

```r
print(paste0("Lower Interval (95%): ", lower95))
```

```
## [1] "Lower Interval (95%): 31.6802319099457"
```

```r
print(paste0("Upper Interval (95%): ", upper95))
```

```
## [1] "Upper Interval (95%): 36.5051459908254"
```

```r
print("--------------------------")
```

```
## [1] "--------------------------"
```

```r
print("Prediction interval by R (ANN): ")
```

```
## [1] "Prediction interval by R (ANN): "
```

```r
fc_spain %>%
  filter(.model=='ANN') %>%
  hilo() %>%
  filter(Year==2018) %>% #Get the first prediction
  select(Country, .model, Year, "95%") -> r_95_interval

print(paste0("Interval (95%): ", r_95_interval$"95%"))
```

```
## [1] "Interval (95%): [31.5029949729872, 36.6823829277839]95"
```

```r
#r_95_interval$"95%"



# AAN model:
print("########################################")
```

```
## [1] "########################################"
```

```
print("################# AAN #################")
```

```
## [1] "################# AAN #################"
```

```
print("########################################")
```

```
## [1] "########################################"
```

```
#mean
ANN_mean = (fc_spain %>% filter(.model=='AAN'))$.mean[1]

#standard dev of residuals
ANN_standard_dev = sd((augment(fit_spain)%>% filter(.model=='AAN'))$.resid)

lower95 = ANN_mean - (1.96 * ANN_standard_dev)
upper95 = ANN_mean + (1.96 * ANN_standard_dev)

print("Our prediction interval (AAN): ")
```

```
## [1] "Our prediction interval (AAN): "
```

```
print(paste0("Lower Interval (95%): ", lower95))
```

```
## [1] "Lower Interval (95%): 32.1405669304971"
```

```
print(paste0("Upper Interval (95%): ", upper95))
```

```
## [1] "Upper Interval (95%): 36.9763871714275"
```

```
print("-------------------------")
```

```
## [1] "-------------------------"
```

```
print("Prediction interval by R (AAN): ")
```

```
## [1] "Prediction interval by R (AAN): "
```

```
fc_spain %>%
  filter(.model=='AAN') %>%
  hilo() %>%
  filter(Year==2018) %>% #Get the first prediction
  select(Country, .model, Year, "95%") -> r_95_interval

print(paste0("Interval (95%): ", r_95_interval$"95%"))
```

```
## [1] "Interval (95%): [32.0743554212292, 37.0425986806953]95"
```

```
#r_95_interval$"95%"
```

We can see that the intervals are similar but not the same, we can see that the first 2 digits are the same, while the decimal part starts to differ.