DATA 624 - Homework 5

Richie Rivera

Question 8.1

Consider the the number of pigs slaughtered in Victoria, available in the aus_livestock dataset.

1. Use the ETS() function to estimate the equivalent model for simple exponential smoothing. Find the optimal values of α and λ_0, and generate forecasts for the next four months.

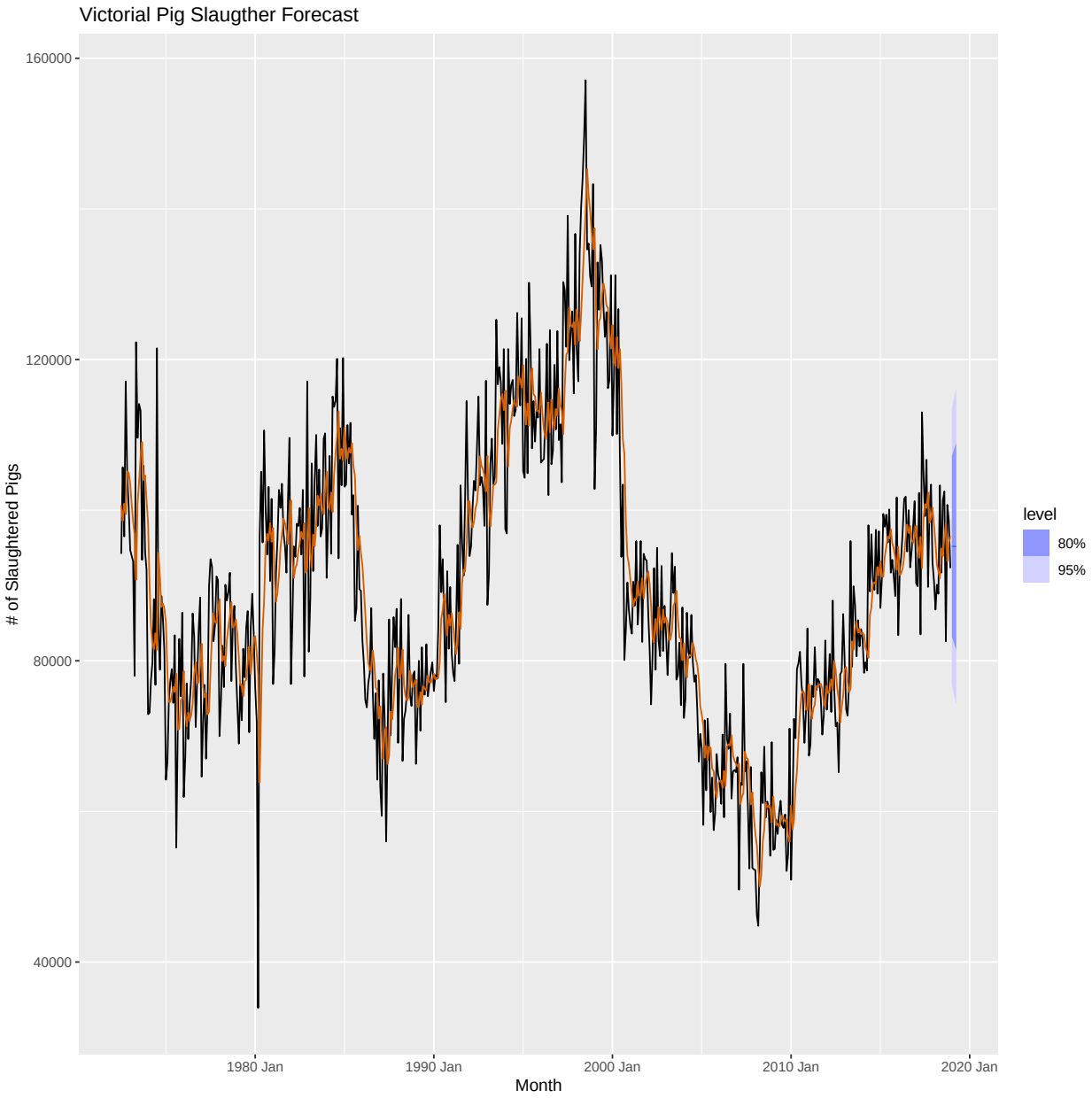
```
library(fpp3)
```

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

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

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

## v tibble      3.2.1     v tsibble     1.1.5
## v dplyr       1.1.4     v tsibbledata 0.4.1
## v tidyr       1.3.0     v feasts      0.3.2
## v lubridate   1.9.3     v fable       0.3.4
## v ggplot2     3.5.1     v fabletools  0.4.2

## Warning: package 'ggplot2' was built under R version 4.3.3

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

## Warning: package 'tsibbledata' was built under R version 4.3.3

## Warning: package 'feasts' was built under R version 4.3.3

## Warning: package 'fabletools' was built under R version 4.3.3

## Warning: package 'fable' was built under R version 4.3.3

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

```r
data(aus_livestock)

head(aus_livestock)
```

```
## # A tsibble: 6 x 4 [1M]
## # Key:       Animal, State [1]
##      Month Animal                    State                           Count
##      <mth> <fct>                     <fct>                           <dbl>
## 1 1976 Jul Bulls, bullocks and steers Australian Capital Territory  2300
## 2 1976 Aug Bulls, bullocks and steers Australian Capital Territory  2100
## 3 1976 Sep Bulls, bullocks and steers Australian Capital Territory  2100
## 4 1976 Oct Bulls, bullocks and steers Australian Capital Territory  1900
## 5 1976 Nov Bulls, bullocks and steers Australian Capital Territory  2100
## 6 1976 Dec Bulls, bullocks and steers Australian Capital Territory  1800
```

```r
vic_pig <- aus_livestock |>
  filter(
    Animal == "Pigs",
    State == "Victoria"
  ) |>
  select(
    -Animal, -State
  )

piggy_fit <- vic_pig |>
  model(ETS(Count ~ error("A") + trend("N") + season("N")))

piggy_fc <- piggy_fit |>
  forecast(h = 4)

piggy_fc |>
  autoplot(vic_pig) +
  geom_line(aes(y = .fitted), col = "#D55E00", data = augment(piggy_fit)) +
  labs(y = "# of Slaughtered Pigs", title = "Victorial Pig Slaugther Forecast")
```




```
piggy_fit |>
  report()
```

```
## 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
```

```
piggy_fc
```

```
## # A fable: 4 x 4 [1M]
## # Key:     .model [1]
##   .model                                              Month             Count  .mean
##   <chr>                                               <mth>            <dist>  <dbl>
## 1 "ETS(Count ~ error(\"A\") + trend(\"N\") + ~ 2019 Jan N(95187, 8.7e+07) 95187.
## 2 "ETS(Count ~ error(\"A\") + trend(\"N\") + ~ 2019 Feb N(95187, 9.7e+07) 95187.
## 3 "ETS(Count ~ error(\"A\") + trend(\"N\") + ~ 2019 Mar N(95187, 1.1e+08) 95187.
## 4 "ETS(Count ~ error(\"A\") + trend(\"N\") + ~ 2019 Apr N(95187, 1.1e+08) 95187.
```

From the above report, we can see that the optimal α is `0.322` and the optimal λ_0 is `100646.6`.

2. Compute a 95% prediction interval for the first forecast using $\hat{y}$ś1.96s where s is the standard deviation of the residuals. Compare your interval with the interval produced by R.

```
piggy_sd <- augment(piggy_fit) |>
  pull(.resid) |>
  sd()

piggy_val <- piggy_fc |>
  pull(Count) |>
  head(1)

piggy_uci <- piggy_val - 1.96 * piggy_sd
piggy_lci <- piggy_val + 1.96 * piggy_sd

print(piggy_uci)
```

```
## <distribution[1]>
## [1] N(76871, 8.7e+07)
```

```
print(piggy_lci)
```

```
## <distribution[1]>
## [1] N(113502, 8.7e+07)
```

The forecast's 95% confidence inverval are N(113502, 8.7e+07) and N(76871, 8.7e+07)

```
head(hilo(piggy_fc$Count, 95), 1)
```

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

Comparing the two, thereis a very slight difference. This difference is about 0.022% for the lower bound and -0.014% different for the upper bound.

Question 8.5

Data set global_economy contains the annual Exports from many countries. Select one country to analyse.

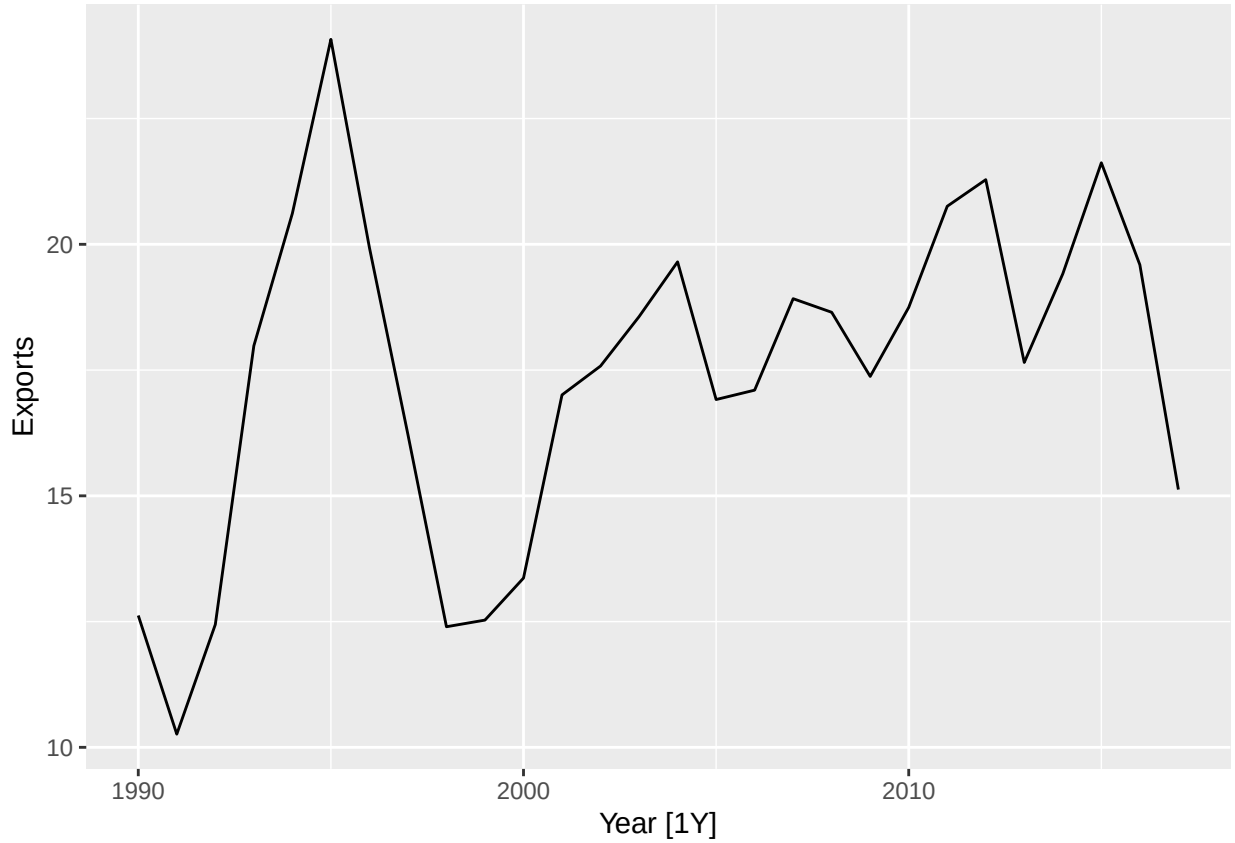
1. Plot the Exports series and discuss the main features of the data.

```
data(global_economy)

head(global_economy)
```

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

```
taz_exp <- global_economy |>
  filter(
    Country == "Tanzania"
  ) |>
  select(
    Country, Exports
  ) |>
  drop_na()

autoplot(taz_exp)
```

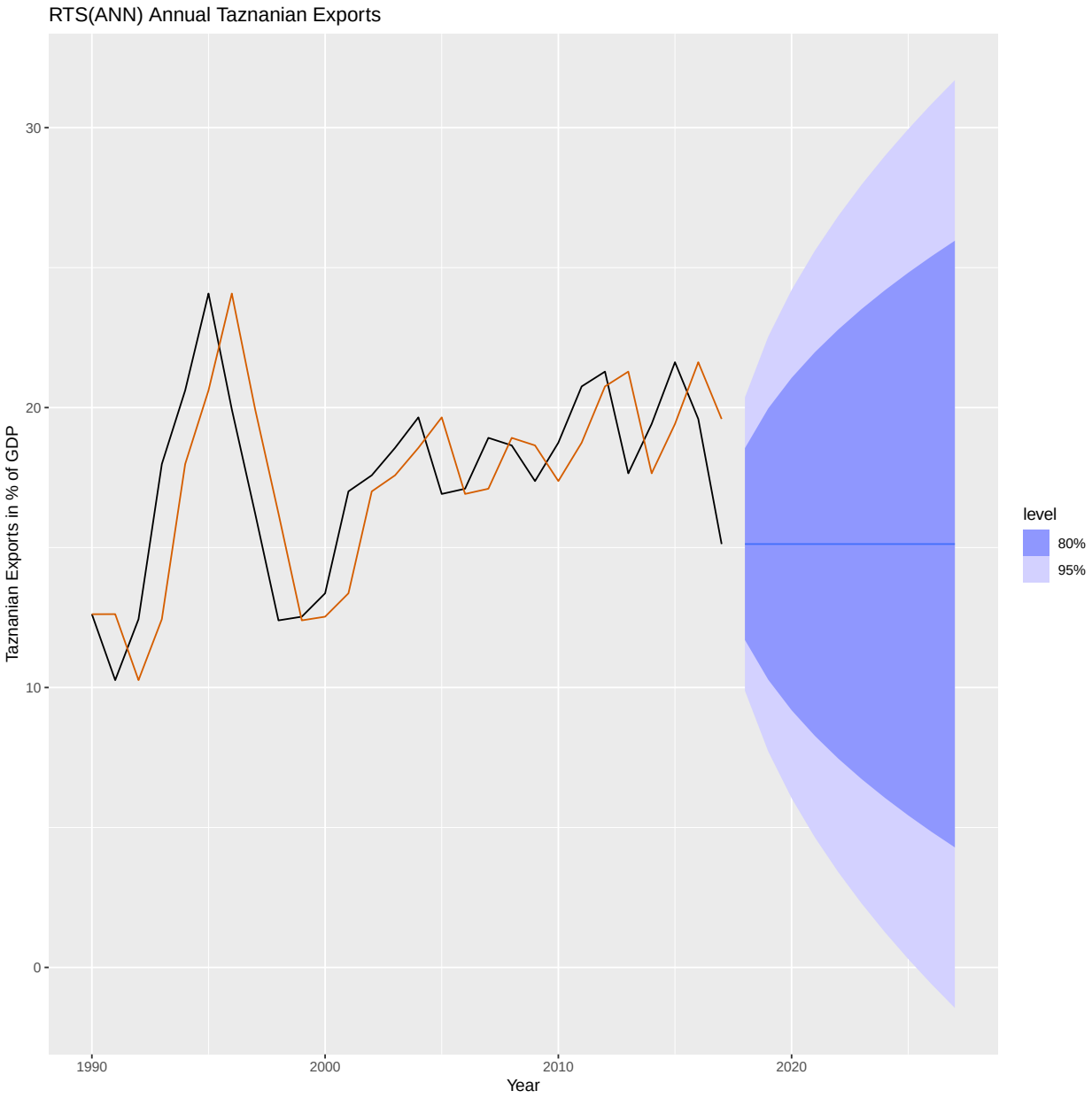
```
## Plot variable not specified, automatically selected `.vars = Exports`
```



Although the dataset starts much earlier, the earliest data point for Tanzania is 1990. The data is annual and there does seem to be some seasonality in Tanzania's exports starting around the year 2002. That being said, Tanzania seems to expereience some strong increases and decreases from year to year.

2. Use an ETS(A,N,N) model to forecast the series, and plot the forecasts.

```
taz_fit_ann <- taz_exp |>
  model(ETS(Exports ~ error("A") + trend("N") + season("N")))

taz_fc_ann <- taz_fit_ann |>
  forecast(h = 10)

taz_fc_ann |>
  autoplot(taz_exp) +
  geom_line(aes(y = .fitted), col = "#D55E00", data = augment(taz_fit_ann)) +
  labs(y = "Taznanian Exports in % of GDP", title = "RTS(ANN) Annual Taznanian Exports")
```



3. Compute the RMSE values for the training data.

```
taz_fit_ann |>
  accuracy()
```

```
## # 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 Tanzania "ETS(Exports~ Trai~ 0.0895  2.58  2.12 -0.541  12.5 0.964 0.982 0.272
```

```
taz_fit_ann_rmse <- taz_fit_ann |>
  accuracy() |>
  pull(RMSE)
```

The RMSE of the training data is 2.5768974.

4. Compare the results to those from an ETS(A,A,N) model. (Remember that the trended model is using one more parameter than the simpler model.) Discuss the merits of the two forecasting methods for this data set.

```
taz_fit_aan <- taz_exp |>
  model(ETS(Exports ~ error("A") + trend("A") + season("N")))

taz_fc_aan <- taz_fit_ann |>
  forecast(h = 10)

taz_fit_aan_rmse <- taz_fit_aan |>
  accuracy() |>
  pull(RMSE)

taz_fit_aan_rmse
```

```
## [1] 2.586862
```

With an RMSE of 2.587 for the ETS(A,A,N) model and an RMSE of 2.577 we can see that these two models have very similar RMSE.

```
taz_fit_ann |>
  accuracy()
```

```
## # 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 Tanzania "ETS(Exports~ Trai~ 0.0895  2.58  2.12 -0.541  12.5 0.964 0.982 0.272
```

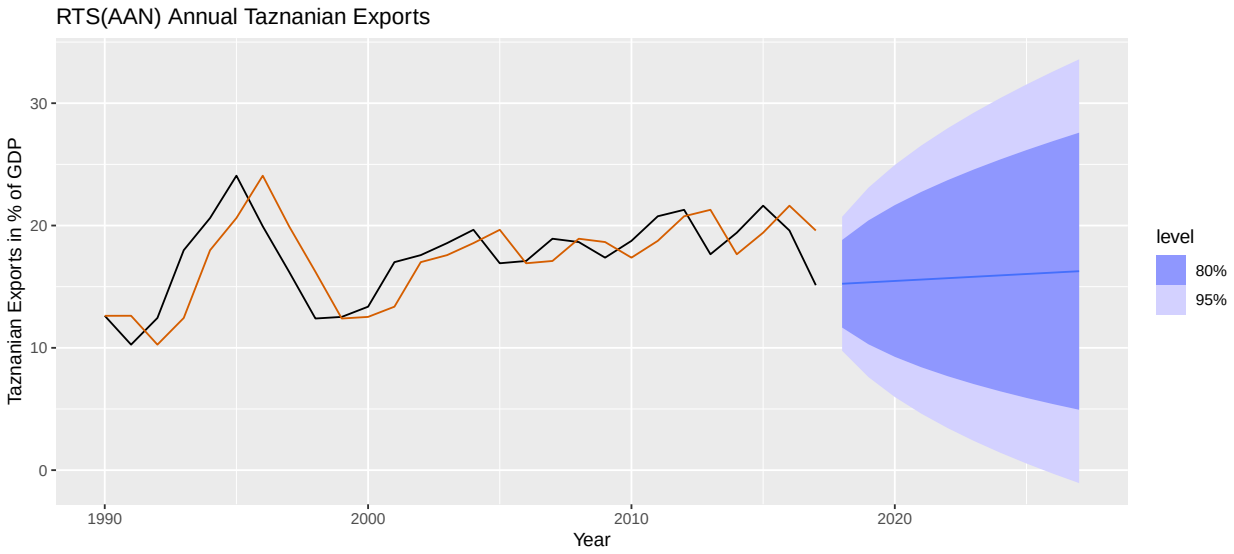
```
taz_fit_aan |>
  accuracy()
```

```
## # 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 Tanzania "ETS(Exports~ Trai~ -0.0667  2.59  2.14 -1.56  12.8 0.972 0.986 0.286
```

Across the other metrics, they are very similar, each within a percentage point of eachother. I believe that this is the case because the Tanzanian exports are very volitile which doesn't lend itself well to smoothing.

5. Compare the forecasts from both methods. Which do you think is best?

```
taz_fit_aan |>
  forecast(h = 10) |>
  autoplot(taz_exp) +
  geom_line(aes(y = .fitted), col = "#D55E00", data = augment(taz_fit_ann)) +
  labs(y = "Taznanian Exports in % of GDP", title = "RTS(AAN) Annual Taznanian Exports")
```



The band of confidence intervals for each is very large so both forecasts are pretty much equally similar but I believe that the RTS(A, A, N) is better because it does take the trend into consideration and despite the volitility, the Taznanian Exports is showing to grow since 1990.

6. Calculate a 95% prediction interval for the first forecast for each model, using the RMSE values and assuming normal errors. Compare your intervals with those produced using R.

```
taz_exp_ann_rmse <- taz_fit_ann |>
  accuracy() |>
  pull(RMSE)

taz_exp_ann_lci <- taz_fc_ann$.mean |>
  head(1) - 1.96 * taz_exp_ann_rmse

taz_exp_ann_uci <- taz_fc_ann$.mean |>
  head(1) + 1.96 * taz_exp_ann_rmse

taz_ann_ci <- c(taz_exp_ann_lci, taz_exp_ann_uci)

taz_exp_aan_rmse <- taz_fit_aan |>
  accuracy() |>
  pull(RMSE)

taz_exp_aan_lci <- taz_fc_aan$.mean |>
  head(1) - 1.96 * taz_exp_aan_rmse

taz_exp_aan_uci <- taz_fc_aan$.mean |>
  head(1) + 1.96 * taz_exp_aan_rmse

taz_aan_ci <- c(taz_exp_aan_lci, taz_exp_aan_uci)

print(taz_ann_ci)
```

```
## [1] 10.07410 20.17554
```

```r
print(taz_aan_ci)
```

```
## [1] 10.05457 20.19507
```

```r
head(hilo(taz_fc_ann$Exports, 95), 1)
```

```
## <hilo[1]>
## [1] [9.883534, 20.3661]95
```

In the above 3 prints, the first one corresponds to the ETS(A,N,N) model, the second corresponds to the ETS(A,A,N) model, and the last one is the one calculated by R. Across these, we can see that all three are very similar but the R model has a significantly larger margin of error when compared to the two models. Despite this difference, they are all still very close to eachoether.

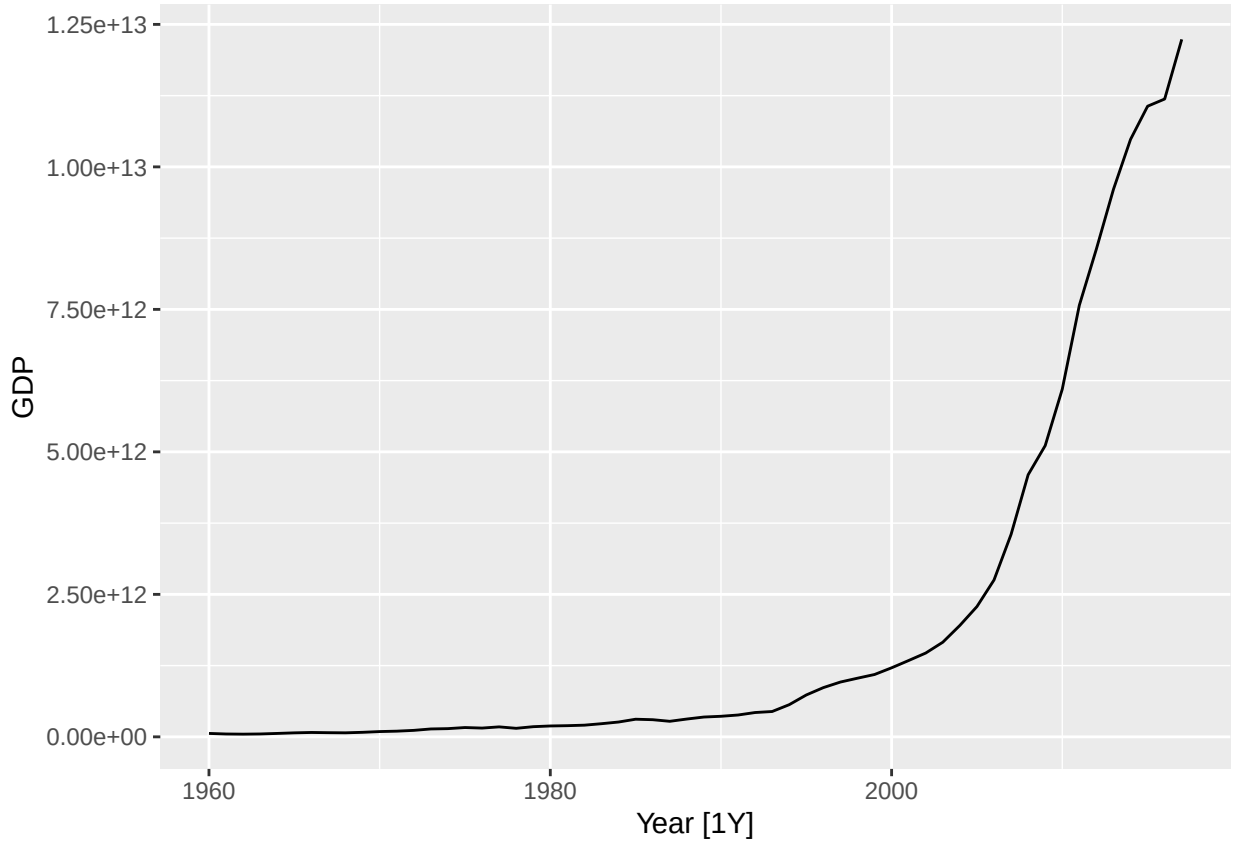
Question 8.6

Forecast the Chinese GDP from the global_economy data set using an ETS model. Experiment with the various options in the ETS() function to see how much the forecasts change with damped trend, or with a Box-Cox transformation. Try to develop an intuition of what each is doing to the forecasts.

[Hint: use a relatively large value of `h` when forecasrting, so you can clearly see the differences between the various options when plotting the forecasts.]

```r
china_h <- 30

china <- global_economy |>
  filter(
    Country == "China"
  ) |>
  select(
    GDP
  )

china |>
  autoplot()
```

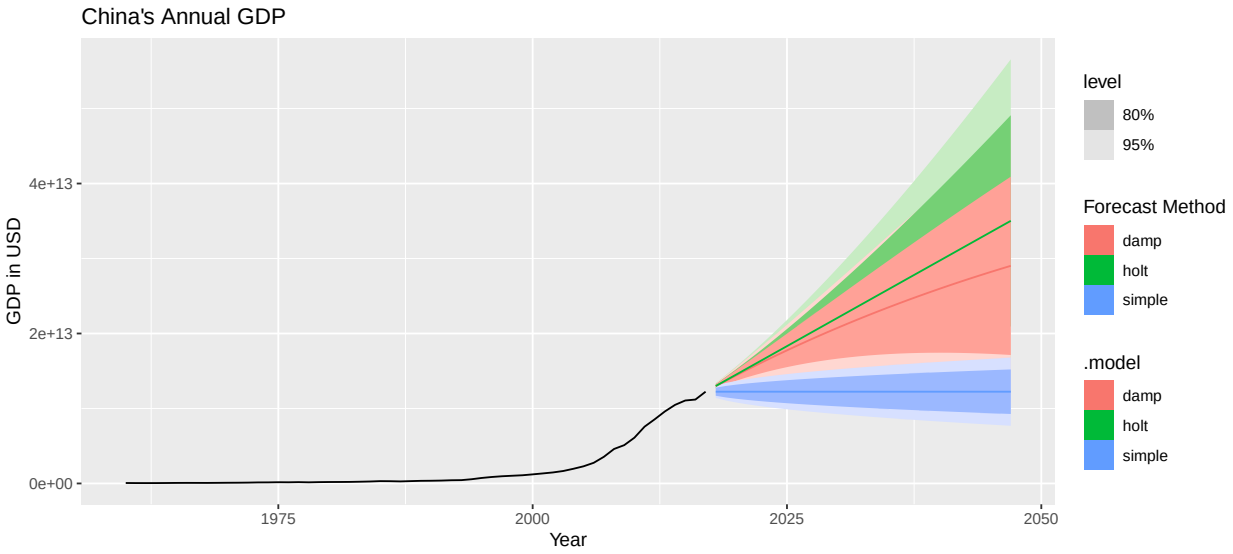
```
## Plot variable not specified, automatically selected `.vars = GDP`
```



The first thing I notice is that this data seems to have exponential growth and that there doesn't seem to be any seasonality.

We'll first see how a Simple, Holt's, and damped model would perform here:

```
china_fits <- china |>
  model(
    simple = ETS(GDP ~ error("A") + trend("N") + season("N")),
    holt = ETS(GDP ~ error("A") + trend("A") + season("N")),
    damp = ETS(GDP ~ error("A") + trend("Ad") + season("N")),
  )

china_fcs <- china_fits |>
  forecast(h = china_h)

china_fcs |>
  autoplot(china) +
  labs(y = "GDP in USD", title="China's Annual GDP") +
  guides(colour = guide_legend(title = "Forecast Method"))
```



From these 3 models, I would prefer the dampened one. It does have the largest confidence band but it also assumes that the growth will slow down or stop and that aligns well with my intuition of GDP. Additionally, we can see how the simple model is essentially using the last value while the Holt's and dampened models are extending values into the future.

Taking a box-cox trasnform:

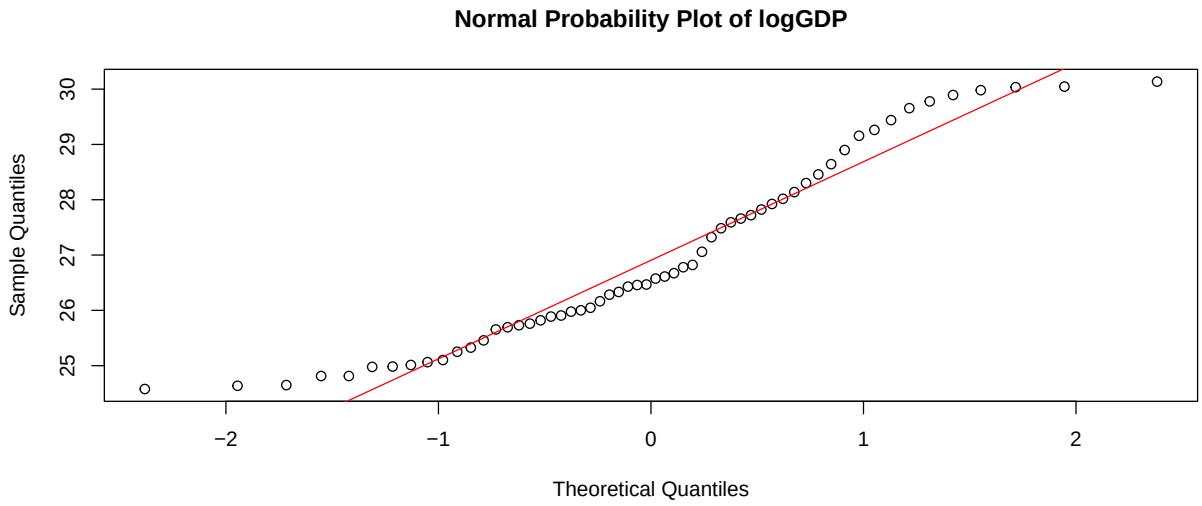
```
china_guerro <- china |>
  features(GDP, features = guerrero) %>%
  pull(lambda_guerrero)

print(china_guerro)
```

```
## [1] -0.03446284
```

With a lambda of -0.034 we can use the chart here to see that it's similar to taking the log of GDP. A check that should be done is to make sure that the result is normal which we can do with a QQ plot.

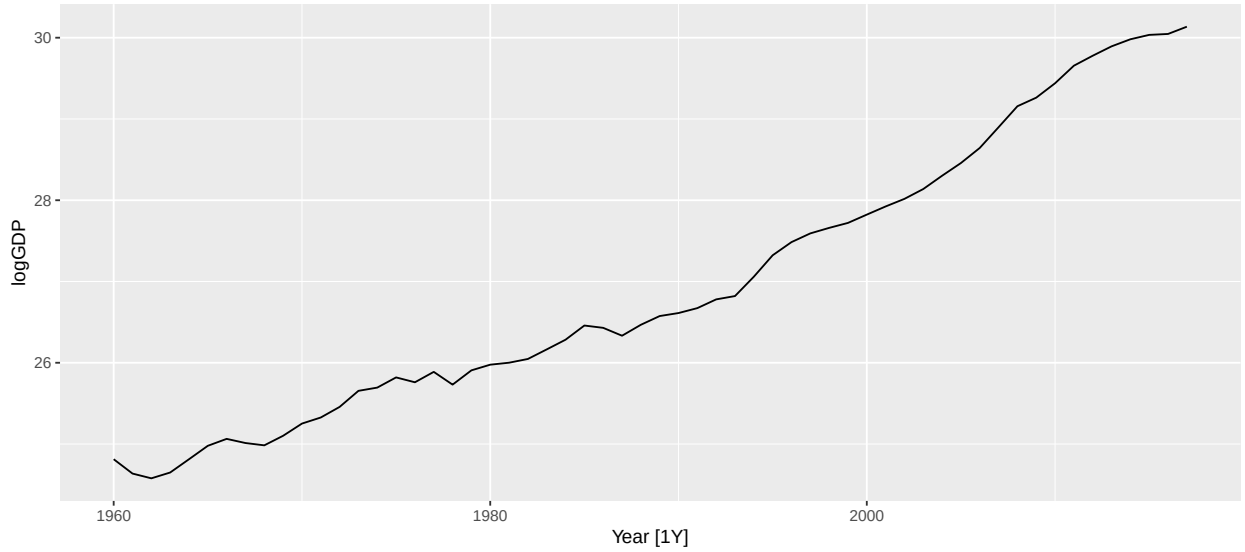
Doing so:

```
china_log <- china |>
  mutate(logGDP = log(GDP)) |>
  select(logGDP)

qqnorm(china_log$logGDP, main = "Normal Probability Plot of logGDP")
qqline(china_log$logGDP, col = "red")  # Adds a reference line
```




```
china_log |>
  autoplot()
```

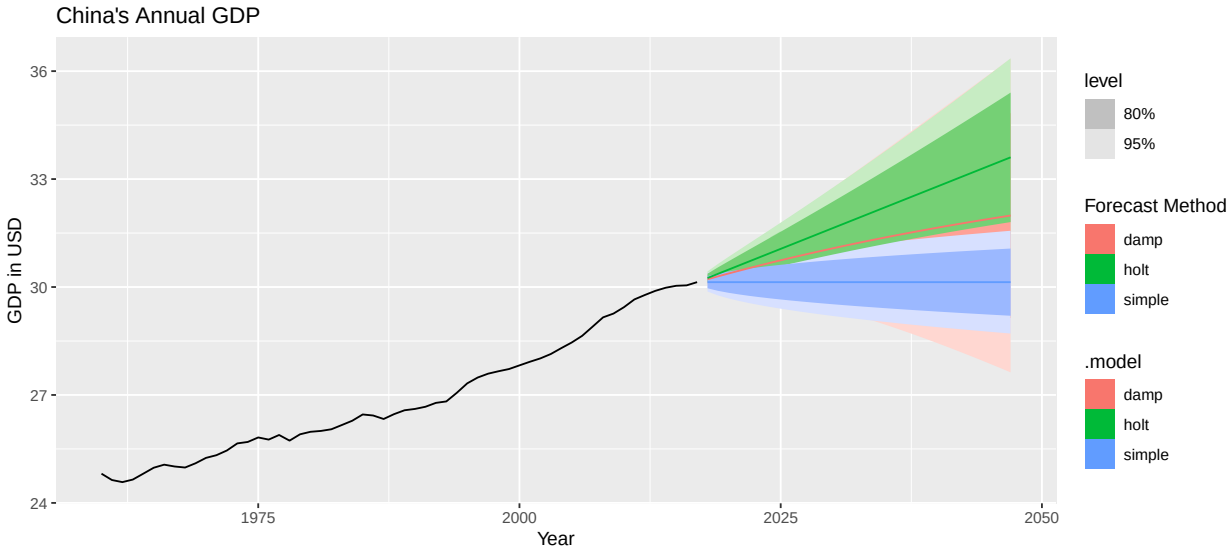
```
## Plot variable not specified, automatically selected `.vars = logGDP`
```



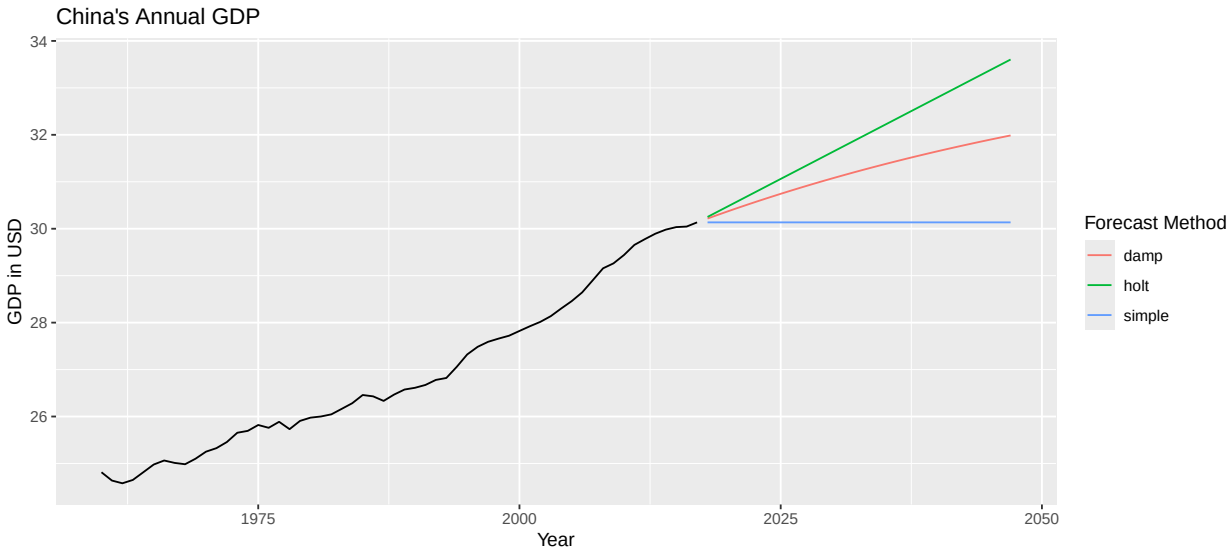
We can see that the distribution is relatively close to the normal line (in red above.) So we'll continue and create forecasts with the tranformed data:

```
china_log_fits <- china_log |>
  model(
    simple = ETS(logGDP ~ error("A") + trend("N") + season("N")),
    holt = ETS(logGDP ~ error("A") + trend("A") + season("N")),
    damp = ETS(logGDP ~ error("A") + trend("Ad") + season("N")),
  )

china_log_fcs <- china_log_fits |>
```

```
  forecast(h = china_h)

china_log_fcs |>
  autoplot(china_log) +
  labs(y = "GDP in USD", title="China's Annual GDP") +
  guides(colour = guide_legend(title = "Forecast Method"))
```




```
china_log_fcs |>
  autoplot(china_log, level = NULL) +
  labs(y = "GDP in USD", title="China's Annual GDP") +
  guides(colour = guide_legend(title = "Forecast Method"))
```



There are two graphs here because the confidence intervals obscured the forecasts of the other models, but looking at the second graph we can see much of what we observed in the first set of forecasts.

Question 8.7

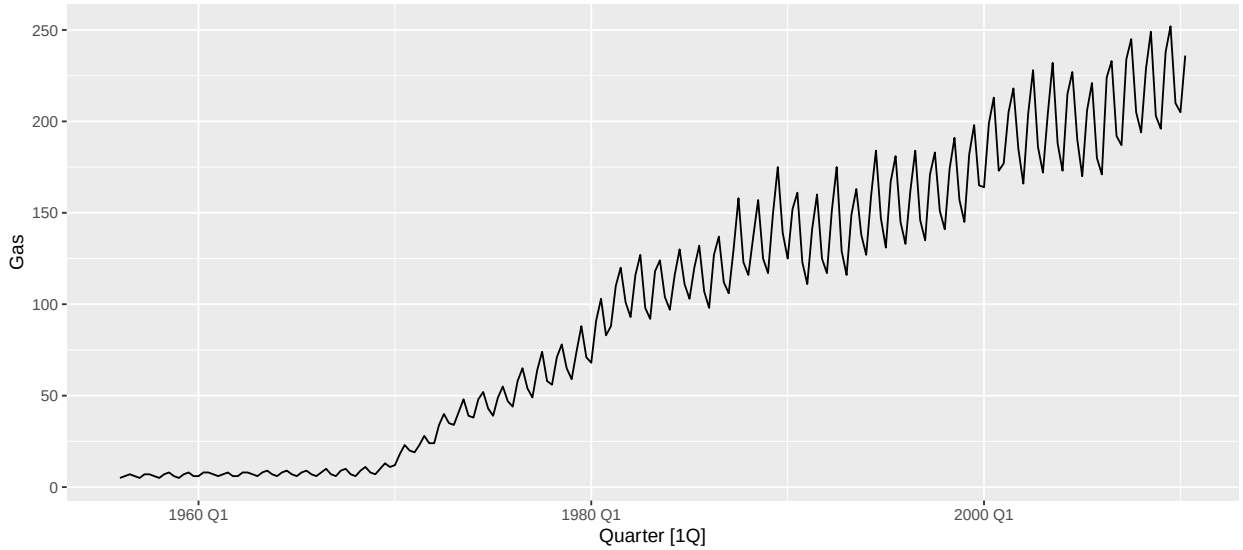
Find an ETS model for the Gas data from aus_production and forecast the next few years. Why is multiplicative seasonality necessary here? Experiment with making the trend damped. Does it improve the forecasts?

```
data(aus_production)
head(aus_production)
```

```
## # A tsibble: 6 x 7 [1Q]
##   Quarter  Beer Tobacco Bricks Cement Electricity   Gas
##     <qtr> <dbl>   <dbl>  <dbl>  <dbl>       <dbl> <dbl>
## 1 1956 Q1   284    5225    189    465        3923     5
## 2 1956 Q2   213    5178    204    532        4436     6
## 3 1956 Q3   227    5297    208    561        4806     7
## 4 1956 Q4   308    5681    197    570        4418     6
## 5 1957 Q1   262    5577    187    529        4339     5
## 6 1957 Q2   228    5651    214    604        4811     7
```

```
gas <- aus_production |>
  select(
    Gas
  )

gas |>
  autoplot()
```

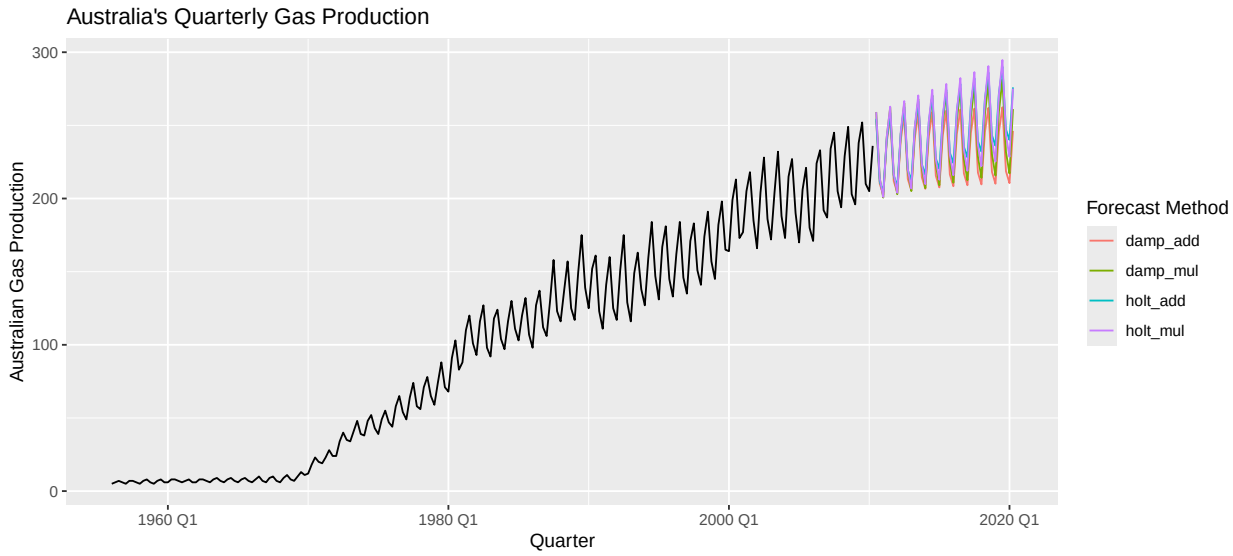
```
## Plot variable not specified, automatically selected `.vars = Gas`
```



From just the graph, we can see that the magnitude of seasonal changes increases over time. That may be an indication that we would need to use a multiplicative seasonality component.

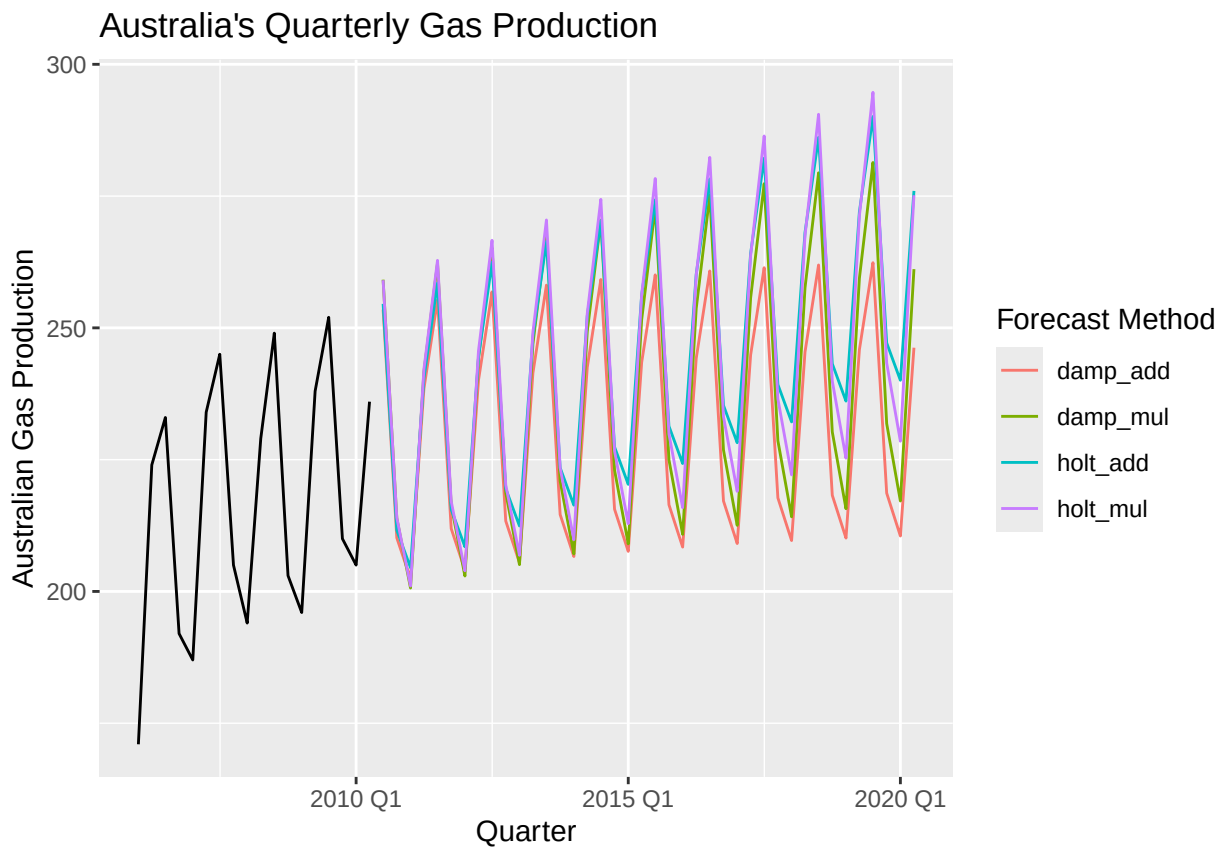
We'll start by creating a series of models:

```
gas_fit <- gas |>
  model(
    holt_add = ETS(Gas ~ error("A") + trend("A") + season("A")),
    holt_mul = ETS(Gas ~ error("M") + trend("A") + season("M")),
    damp_add = ETS(Gas ~ error("A") + trend("Ad") + season("A")),
    damp_mul = ETS(Gas ~ error("M") + trend("Ad") + season("M"))
  )

gas_fc <- gas_fit |>
  forecast(h = 4 * 10)

gas_fc |>
  autoplot(gas, level = NULL) +
  labs(y = "Australian Gas Production", title="Australia's Quarterly Gas Production") +
  guides(colour = guide_legend(title = "Forecast Method"))
```



All of these forecasts seem very similar but that could be due to the scale. Let's filter the data to only show entries after 2006:

```
gas_fc |>
  autoplot(gas |> filter(Quarter >= yearquarter("2006 Q1")), level = NULL) +
  labs(y = "Australian Gas Production", title="Australia's Quarterly Gas Production") +
  guides(colour = guide_legend(title = "Forecast Method"))
```



Narrowing in is much better as we can see the slight differences between the methods. A noticible difference is when comparing the damped methods against the non-damped methods. We can see that as the forecast duration is increased, the dampened forecasts show lower lows and lower high values than the non-damped forecasts. Although we did say that multiplicative seasonality is necessary, the multiplicative forecasts don't seem to have much difference until the forecast window is 6 years out.

Question 8.8

Recall your retail time series data (from Exercise 7 in Section 2.10).

1. Why is multiplicative seasonality necessary for this series?

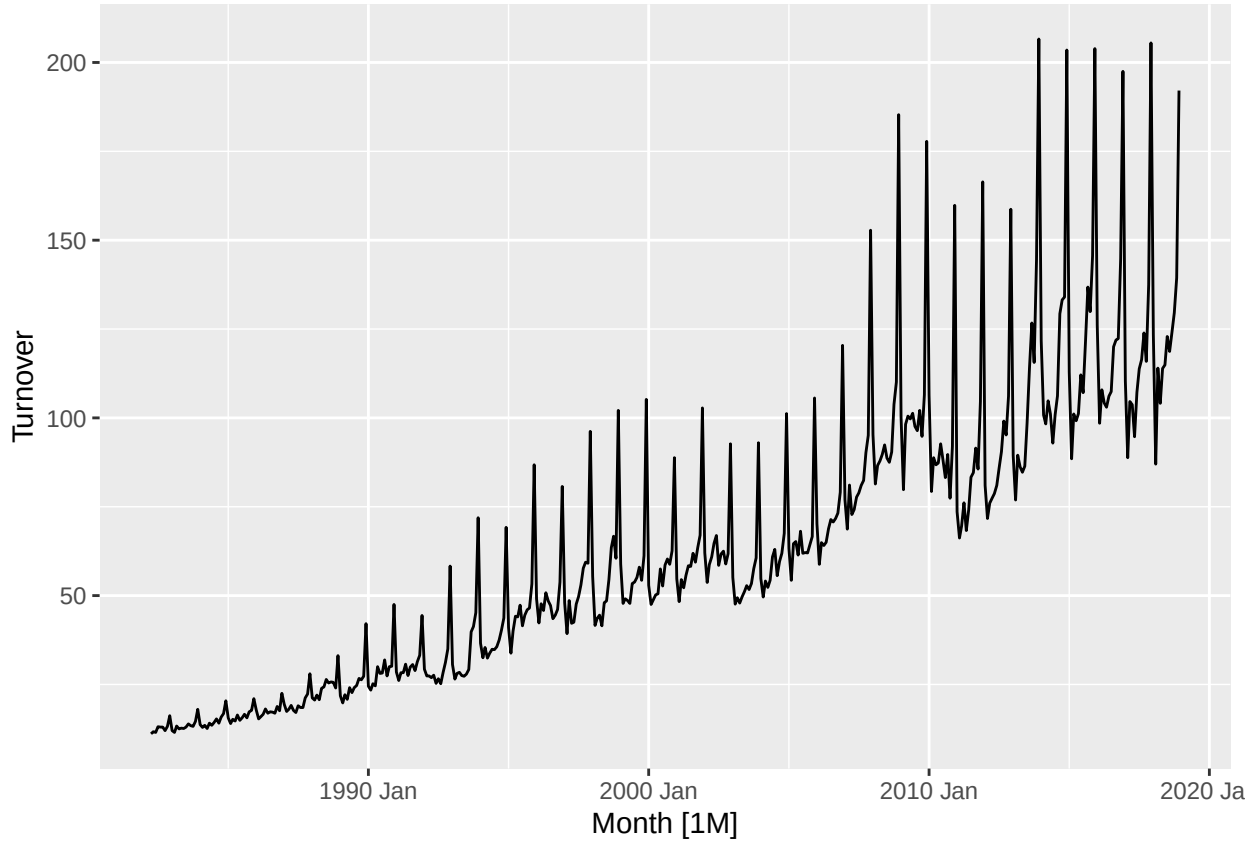
```
set.seed(2111994)

myseries <- aus_retail |>
  filter(`Series ID` == sample(aus_retail$`Series ID`, 1)) |>
  select(Turnover)

head(myseries)
```

```
## # A tsibble: 6 x 2 [1M]
##   Turnover    Month
##      <dbl>    <mth>
## 1     11.1 1982 Apr
```

```
## 2     11.7 1982 May
## 3     11.5 1982 Jun
## 4     13.1 1982 Jul
## 5     13   1982 Aug
## 6     13   1982 Sep
```

```
myseries |>
  autoplot()
```

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

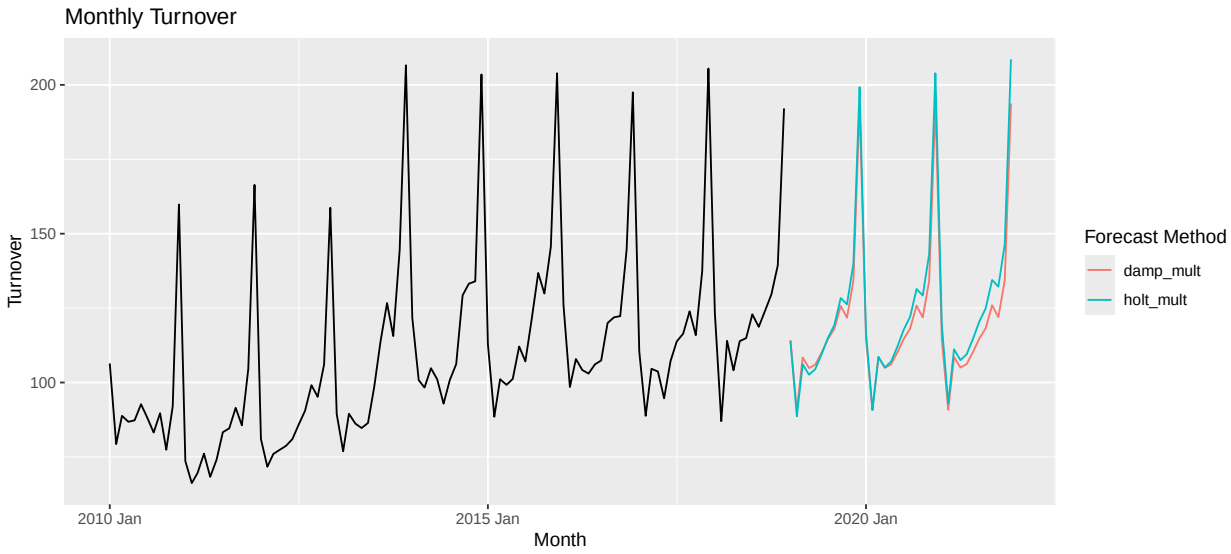


Although `myseries` is very volitile, I can see that the magnitude of what appears to be seasonal impact changes over time. This would be the reason to use multiplicative seasonality.

2. Apply Holt-Winters' multiplicative method to the data. Experiment with making the trend damped.

```
myfit <- myseries |>
  model(
    holt_mult = ETS(Turnover ~ error("M") + trend("A") + season("M")),
    damp_mult = ETS(Turnover ~ error("M") + trend("Ad") + season("M"))
  )

myfc <- myfit |>
  forecast(h = 12 * 3)
```

```
myfc |>
  autoplot(myseries |> filter(Month >= yearmonth("2010 January")), level=NULL) +
  labs(y = "Turnover", title="Monthly Turnover") +
  guides(colour = guide_legend(title = "Forecast Method"))
```



A significant difference can be seen here where the undamped Holt-Winters' method shows much higher values very soon relative to the damped version.

3. Compare the RMSE of the one-step forecasts from the two methods. Which do you prefer?

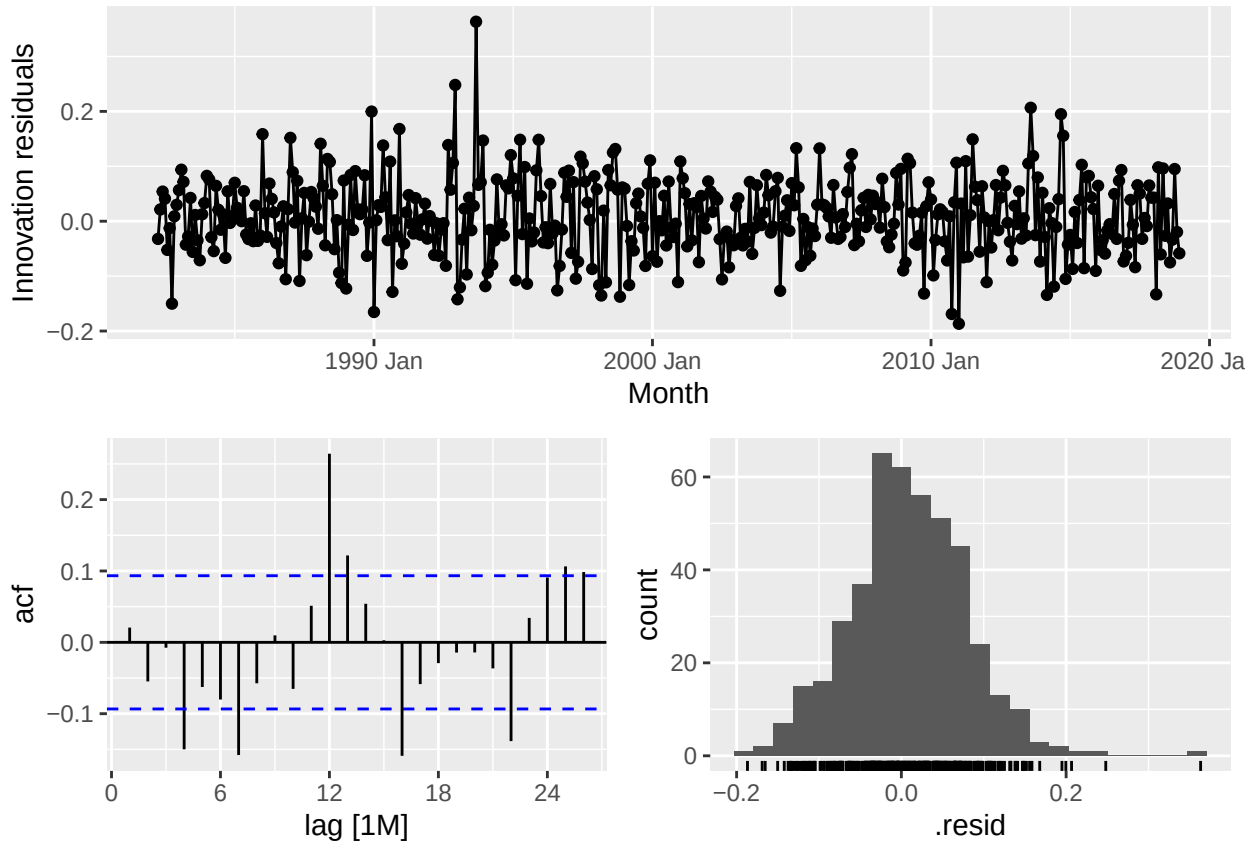
```
myfit |>
  accuracy()
```

```
## # A tibble: 2 x 10
##   .model    .type        ME  RMSE   MAE    MPE  MAPE  MASE RMSSE    ACF1
##   <chr>     <chr>     <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl>   <dbl>
## 1 holt_mult Training 0.0520  5.09  3.46 -0.493  5.51 0.558 0.581  0.155
## 2 damp_mult Training 0.346   5.12  3.48  0.352  5.47 0.561 0.585 -0.0219
```

From the above table, the damped version has a slightly lower RMSE, which makes it a bit perferred. Moving on to the MAPE, we can see something similar where the dampened version is about 0.61% different.

4. Check that the residuals from the best method look like white noise.

```
myfit |>
  select(damp_mult) |>
  gg_tsresiduals()
```

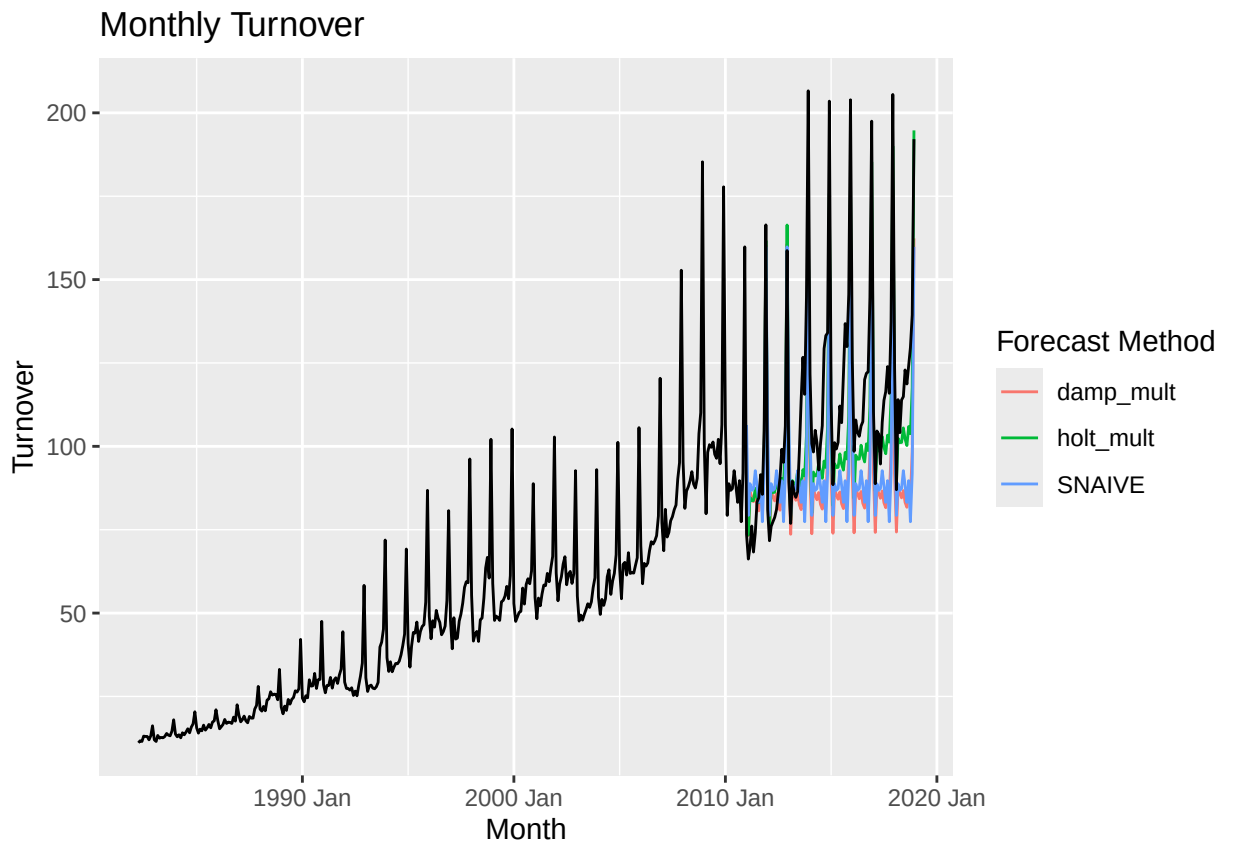


The top graph seems to show that the residuals are evenly spaced around 0 and the histogram appears to be normal. Looking at the ACF, we can see that there are a few outliers but the vast majority of values are within the

5. Now find the test set RMSE, while training the model to the end of 2010. Can you beat the seasonal naïve approach from Exercise 7 in Section 5.11?

```
mytrain <- myseries |>
  filter(year(Month) <= 2010)

mytest <- myseries |>
  filter(year(Month) > 2010)

mytrainfit <- mytrain |>
  model(
    holt_mult = ETS(Turnover ~ error("M") + trend("A") + season("M")),
    damp_mult = ETS(Turnover ~ error("M") + trend("Ad") + season("M")),
    SNAIVE = SNAIVE(Turnover)
  )

mytrainfc <- mytrainfit |>
  forecast(h = 12 * 8)

mytrainfc |>
  autoplot(myseries, level=NULL) +
```

```
  labs(y = "Turnover", title="Monthly Turnover") +
  guides(colour = guide_legend(title = "Forecast Method"))
```



From the graph above, we have ot note that there was an unexpected drop in 2013 followed by a spike which retained its magnitude in around 2014. Our training data was obviously blind to this but even with that we can see that the Holt-Winters' undamped forecast is the closest and performed much better than the `SNAIVE` and the damped version.

Question 8.9

For the same retail data, try an STL decomposition applied to the Box-Cox transformed series, followed by ETS on the seasonally adjusted data. How does that compare with your best previous forecasts on the test set?

```
mytrainlambda <- mytrain |>
  features(Turnover, guerrero) |>
  pull(lambda_guerrero)

mytrainlambda
```

```
## [1] -0.2565856
```

Using the same table from before, -0.257 is very close to a square-root transform:

```
mytrainsqrt <- mytrain |>
  mutate(sqrtTurnover = sqrt(Turnover)) |>
  select(sqrtTurnover)

mysqrtfit <- mytrainsqrt |>
  model(
    stl_sqrt = STL(sqrtTurnover ~ season(window = "periodic"), robust = TRUE),
    ets_sqrt = ETS(sqrtTurnover),
    holt_mult = ETS(sqrtTurnover ~ error("M") + trend("A") + season("M"))
  )

mysqrtfit |>
  accuracy()
```

```
## # A tibble: 3 x 10
##   .model    .type          ME  RMSE   MAE    MPE  MAPE  MASE RMSSE  ACF1
##   <chr>     <chr>       <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 stl_sqrt  Training -0.0132  0.345 0.193 -0.623  3.09 0.536 0.767 0.250
## 2 ets_sqrt  Training -0.00293 0.236 0.178 -0.142  2.68 0.495 0.525 0.116
## 3 holt_mult Training -0.00293 0.236 0.178 -0.142  2.68 0.495 0.525 0.116
```

Looking at just he RMSE, we can see that the ETS and holt's multiplicative method both have much better RMSE than the STL method and that improvement is consistent across all of the other methods. Although that being said the magnitude of the error is relatively close across the three methods here.