Homework Five for Fall 2024 DATA 624: Exponential Smoothing

Kevin Kirby

2024-09-26

Overview

The below are answers to exercises 8.1, 8.5, 8.6, 8.7, 8.8, 8.9 from the exercise section of chapter 8 of [Forecasting: Principles and Practice (3rd ed)] (https://otexts.com/fpp3/graphics-exercises.html). This is homework five of the DATA 624 class "Predictive Analytics & Forecasting." Unless otherwise noted, all datasets used below are from the fpp3 package that's owned and maintained by the book's authors.

First, I'll load the required libraries:

```r
library(fpp3)
```

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

## -- Attaching packages -------------------------------------------- fpp3 1.0.1 --

## v tibble      3.2.1     v tsibble     1.1.5
## v dplyr       1.1.4     v tsibbledata 0.4.1
## v tidyr       1.3.1     v feasts      0.4.0
## v lubridate   1.9.3     v fable       0.4.0
## v ggplot2     3.5.1

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

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats 1.0.0     v readr   2.1.5
## v purrr   1.0.2     v stringr 1.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter()     masks stats::filter()
## x tsibble::interval() masks lubridate::interval()
## x dplyr::lag()        masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(ggplot2)
library(fable)
library(fabletools)
library(feasts)
library(tsibble)
library(dplyr)
```

8.1:Pigs slaughtered in Victoria from aus_livestock

A: Use the ETS() function to estimate the equivalent model for simple exponential smoothing.

"Find the optimal values of alpha and l0, and generate forecasts for the next four months."

Answer: the optimal value of alpha is 0.322 and the optimal value of l0 is 100647

```
data("aus_livestock")

vict_ets <- aus_livestock |>
  filter(Animal == "Pigs", State == "Victoria") |>
  model(SES = ETS(Count ~ error("A") + trend("N") + season("N")))

tidy(vict_ets) |>
  select(term,estimate) |>
  mutate(estimate = round(estimate,6))
```

```
## # A tibble: 2 x 2
##   term    estimate
##   <chr>      <dbl>
## 1 alpha      0.322
## 2 l[0]  100647.
```

```
etc_forecasting <- vict_ets |>
  forecast(h = 4)
```

B: Compute a 95% prediction interval for the first forecast using ^y±1.96s

"...where s is the standard deviation of the residuals. Compare your interval with the interval produced by R."

Answer: Manual range: 76871.01 to 113502.1 R's automatic range using Hilo: 76854.79, 113518.3

Bottom range difference: 23.78 Top range difference: 16.2

```
meany <- etc_forecasting |>
  slice(1) |>
  pull(.mean)

std <- augment(vict_ets) |>
  pull(.resid) |>
  sd()

cat("ANN Model - 95% confidence range: ", meany - 1.96 * std, ",", meany + 1.96 * std, "\n")
```

```
## ANN Model - 95% confidence range:  76871.01 , 113502.1
```

```
r_interval <- etc_forecasting |>
  slice(1) |>
  hilo(95) |>
  pull()
```

```r
print(r_interval)
```

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

8.5: Review of Thai exports data from global_economy

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

I have chosen Thailand for the purpose of this question.

```r
data("global_economy")

thai_exports <- global_economy |>
  filter(Code == "THA") |>
  select(Exports)
```

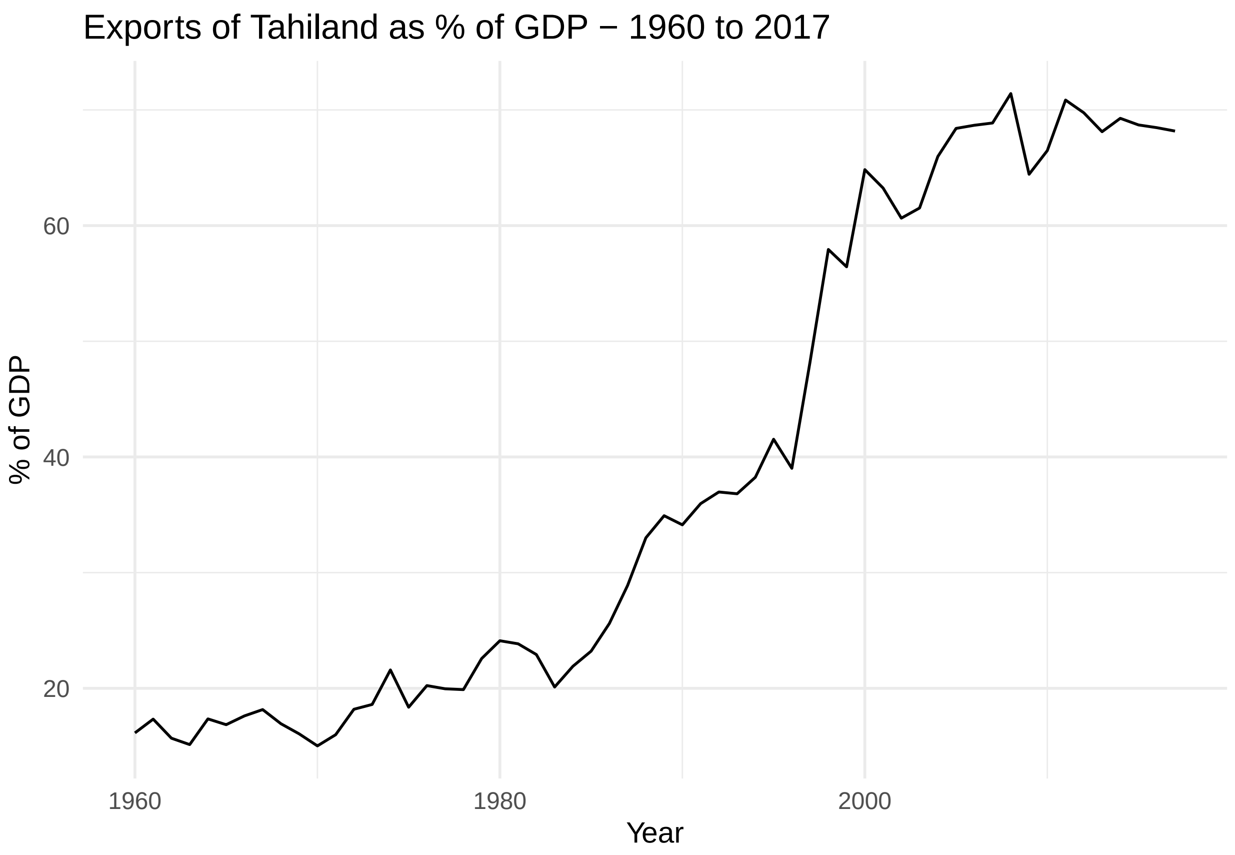
A: Plot the Exports series and discuss the main features of the data

Discussion of main features: pretty wild to see that, in recent years, exports have become the bsis for over 60% of Thailand’s economy. You can see the sharp rise in exports in the late 90’s that was driven by a general push by western companies to move their manufacturing to much cheaper locales. It’s most likely leveled off recently as a function of being at the maximum capacity it can realistically achieve.

The Reagan and Thatcher years in the 80s also cause an initial first surge. Considering their general disdain for workers, it’s not surprising this happened on their watch.

```r
autoplot(thai_exports) +
    labs(title = "Exports of Tahiland as % of GDP - 1960 to 2017",
       x = "Year",
       y = "% of GDP") +
    scale_y_continuous(labels = scales::comma_format()) +
    theme_minimal()
```

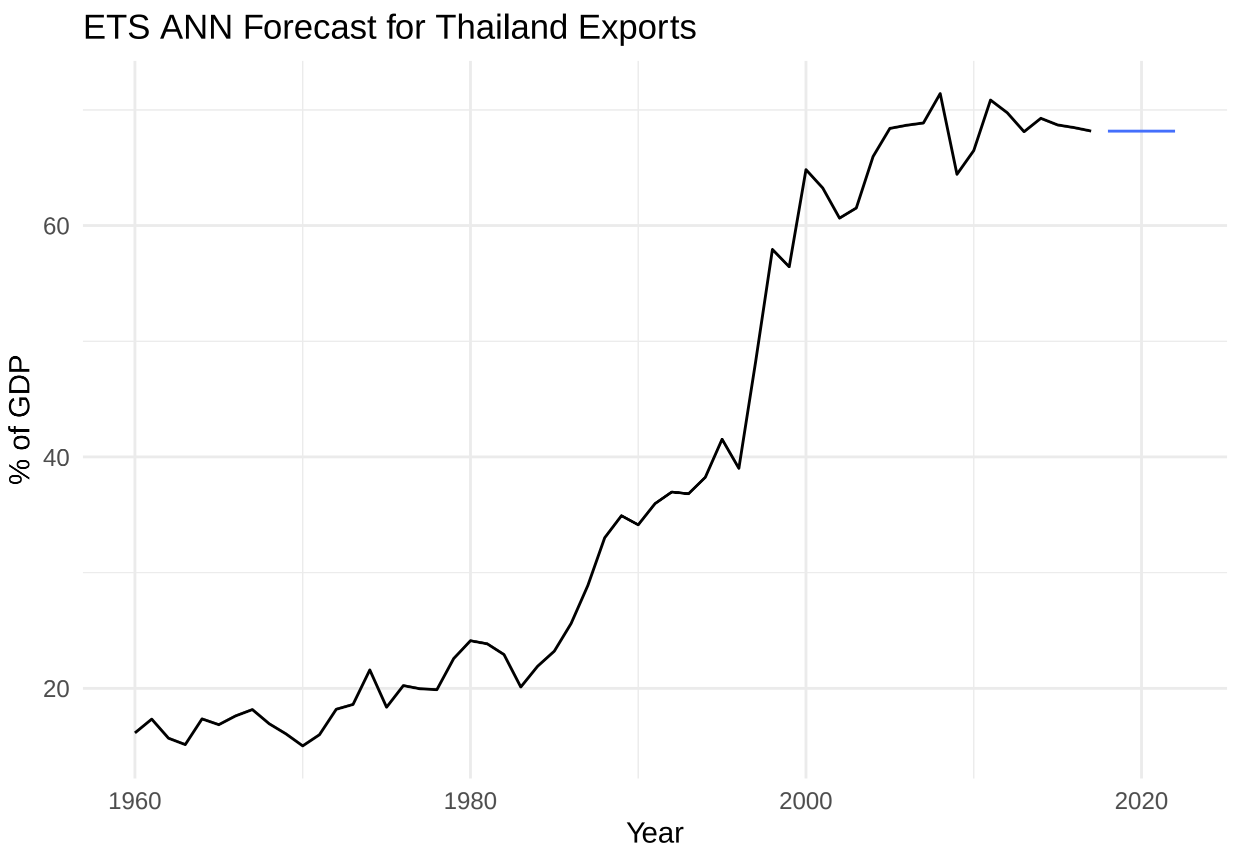
```
## Plot variable not specified, automatically selected `.vars = Exports`
```



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

Answer: this forecast looks pretty decent overall. Exports as a percent of GDP had leveled offf for a 15 year period or so. Expecting it to continue in a relatively leveled off state is a better bet that expecting growth all of a sudden.

```r
thai_cull_ann <- thai_exports |>
  model(ANN = ETS(Exports ~ error("A") + trend("N") + season("N")))

thai_ann <- forecast(thai_cull_ann, h = "5 years")

thai_ann %>%
  autoplot(thai_exports, level=NULL) +
    labs(title = "ETS ANN Forecast for Thailand Exports",
       x = "Year",
       y = "% of GDP") +
    scale_y_continuous(labels = scales::comma_format()) +
    theme_minimal()
```



C: Compute the RMSE values for the training data

RMSE: 2.93416

```
thai_rmse_ann <- accuracy(thai_cull_ann) |>
  pull(RMSE)

thai_rmse_ann
```

```
## [1] 2.93416
```

D: Compare the results to those from an ETS(A,A,N) mode

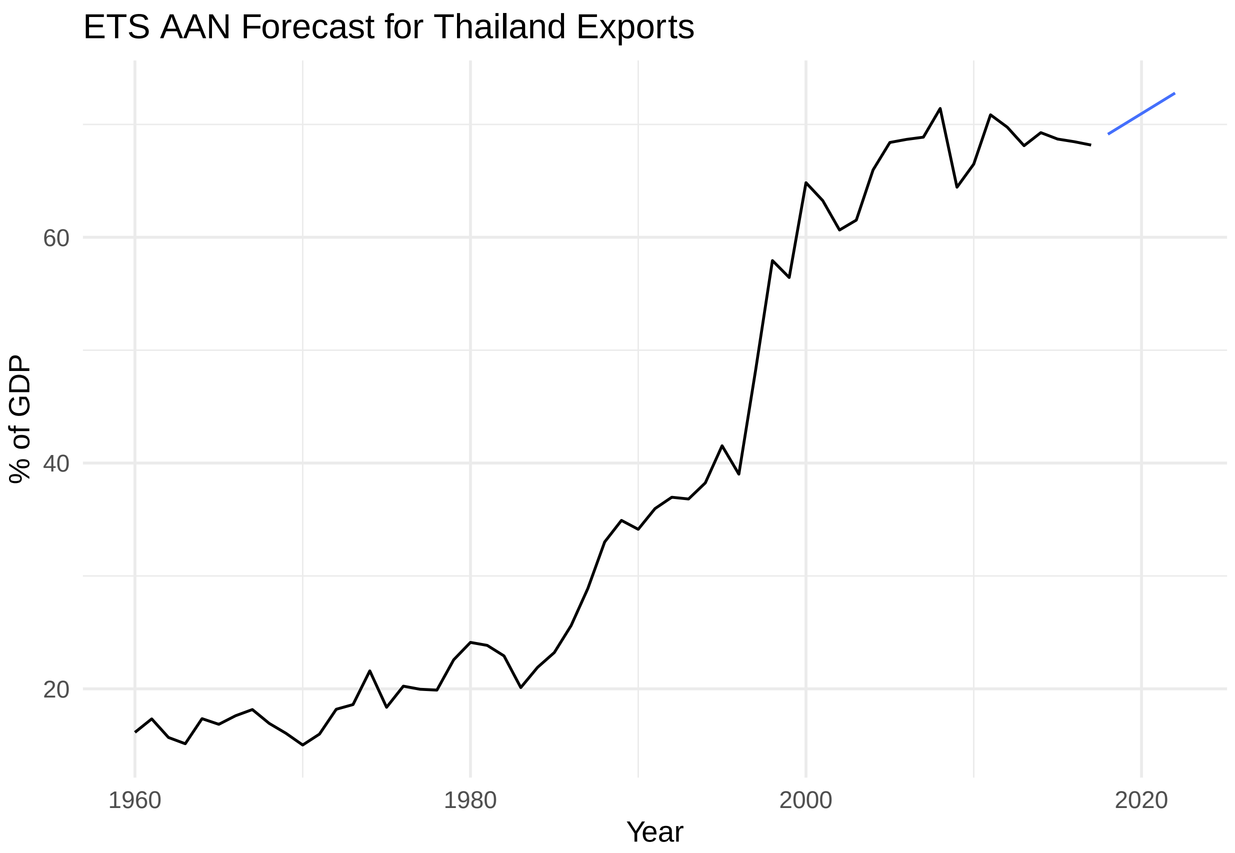
"(Remember that the trended model is using one more parameter than the simpler model. Discuss the merits of the two forecasting methods for this data set."

Answer:

The RMSE for the AAN run came in slightly lower at 2.78955 versus the ANN RMSE of 2.93416, which is interesting because I don't look at the last 15 years of data and see a basis for the trend line going up at a 45 degree angle like it is in AAN. While I'm skeptical of not including a trend component with this data, which is what ANN is doing, the trend accounting of AAN is producing a forecast I'm skeptical of. For this dataset, given recent trends, I would use the ANN unless there was a basis in recently economic fact that warranted a more positive outlook.

```
thai_cull_aan <- thai_exports |>
  model(AAN = ETS(Exports ~ error("A") + trend("A") + season("N")))

thai_aan <- forecast(thai_cull_aan, h = "5 years")
```

```r
thai_aan %>%
  autoplot(thai_exports, level=NULL) +
    labs(title = "ETS AAN Forecast for Thailand Exports",
       x = "Year",
       y = "% of GDP") +
    scale_y_continuous(labels = scales::comma_format()) +
    theme_minimal()
```



```r
thai_rmse_aan <- accuracy(thai_cull_aan) |>
  pull(RMSE)

thai_rmse_aan
```

```
## [1] 2.78955
```

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

Answer: Unless there's some other indicator suggesting there's about to be a structural change in the Thai economy that pointe to increased exports, I would use ANN for this data. The main external data I would look at is what's been happening with Chinese manufacturing and where are those manfuacturers going instead. I could see that causing a structural change.

F: 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

```r
thai_cull_aan <- thai_exports |>
  model(AAN = ETS(Exports ~ error("A") + trend("A") + season("N")))
```

```r
thai_aan <- forecast(thai_cull_aan, h = "5 years")

thai_ann <- as_tsibble(thai_ann)
thai_aan <- as_tsibble(thai_aan)

thai_rmse_aan <- accuracy(thai_cull_aan) |>
  pull(RMSE)


thai_ann$RMSE <- thai_rmse_ann
thai_aan$RMSE <- thai_rmse_aan


thai_ann <- thai_ann |>
  mutate(upper_ci = .mean + 1.96 * RMSE,
         lower_ci = .mean - 1.96 * RMSE)

thai_aan <- thai_aan |>
  mutate(upper_ci = .mean + 1.96 * RMSE,
         lower_ci = .mean - 1.96 * RMSE)


ann_compare <- thai_ann |>
  select(.model, Year, .mean, lower_ci, upper_ci)

aan_compare <- thai_aan |>
  select(.model, Year, .mean, lower_ci, upper_ci)


comparison <- bind_rows(ann_compare, aan_compare)


print(comparison)
```

```
## # A tsibble: 10 x 5 [1Y]
## # Key:       .model [2]
##    .model  Year .mean lower_ci upper_ci
##    <chr>  <dbl> <dbl>    <dbl>    <dbl>
##  1 AAN     2018  69.1     63.7     74.6
##  2 AAN     2019  70.0     64.6     75.5
##  3 AAN     2020  71.0     65.5     76.4
##  4 AAN     2021  71.9     66.4     77.3
##  5 AAN     2022  72.8     67.3     78.2
##  6 ANN     2018  68.2     62.4     73.9
##  7 ANN     2019  68.2     62.4     73.9
##  8 ANN     2020  68.2     62.4     73.9
##  9 ANN     2021  68.2     62.4     73.9
## 10 ANN     2022  68.2     62.4     73.9
```

8.6: Forecast the Chinese GDP from the global_economy data set using an ETS model

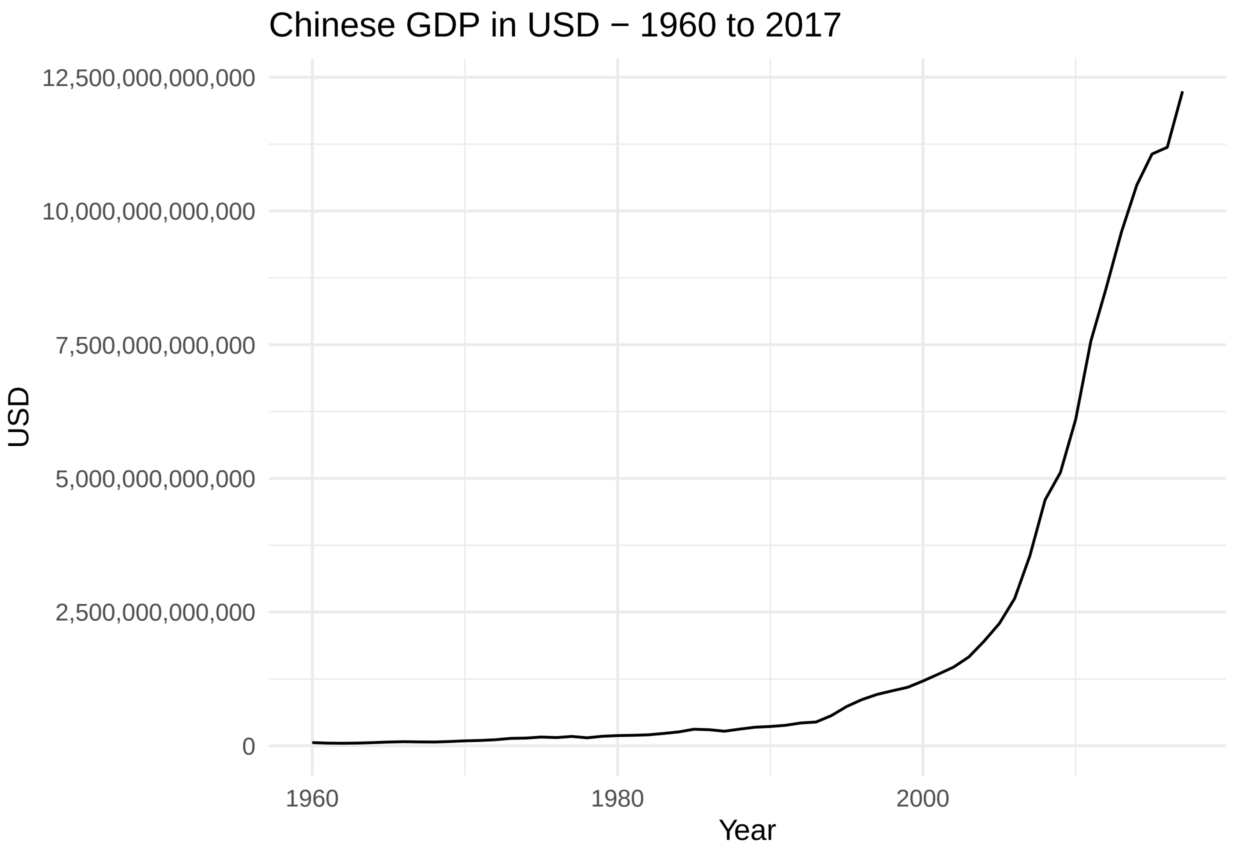
"Experiment with the various options in the ETS() function to see how much the forecasts change with damped trend, or with a Box-Cox transformation. Try to develop an intuition of what each is doing to the forecasts.

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

Filter and plot Chinese GDP

Interesting to see a large lag between opening up to the world in the 1970s and the explosion in GDP that more or less parallels the explosion personal computing use around the world in the 2000s, which drove an urgent need for manufacturing capacity.

```r
china_gdp_tb <- global_economy |>
  filter(Code == "CHN") |>
  select(GDP)

china_gdp_tb %>%
  autoplot(GDP) +
    labs(title = "Chinese GDP in USD - 1960 to 2017",
       x = "Year",
       y = "USD") +
    scale_y_continuous(labels = scales::comma_format()) +
    theme_minimal()
```



Forecasting using multiple ETS methods:

I forecasted using the following ETS models:

- MAM: a multiplicative error model with an additive trend (constant change over time) and multiplicative seasonality. The multiplier on the seasonality components allows it to scale.
- MAD: a multiplicative error model with an additive trend and dampened seasonality, which builds in diminishes seasonality overtime.
- MMMN: A multiplicative error and trend model with no seasonality, which is often used where the trend exhibits multiplicative behavior.
- MADN: A multiplicative error model with damped additive trend and no seasonality. This will show a trend that slows over time without any seasonal variation.
- BOXLOG: when 0 is set as the lambda parameter, you get a classic log transformation. This can stabilize variance under an ETS model.

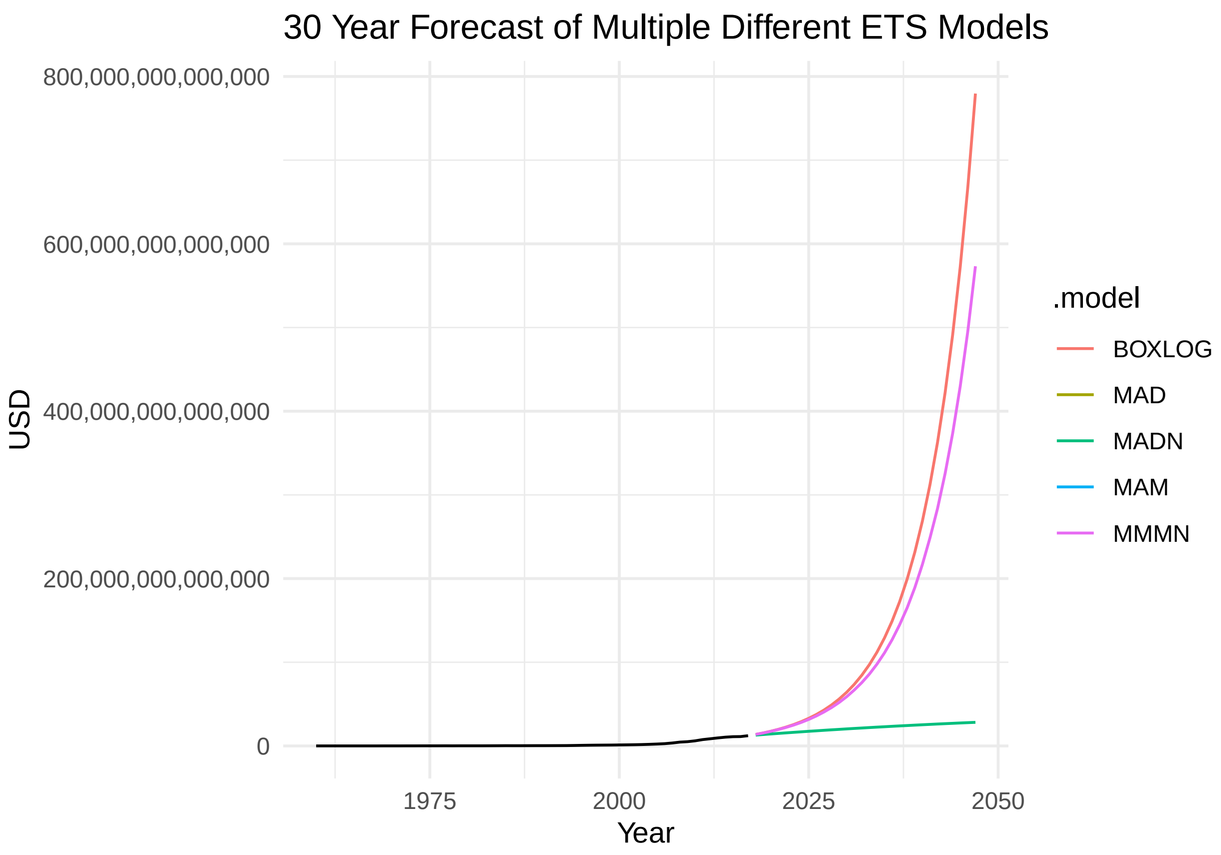
```
china_fitter<- china_gdp_tb|>
  model(
    MAM = ETS(GDP ~ error("M") + trend("A") + season("M")),
    MAD = ETS(GDP ~ error("M") + trend("A") + season("D")),
    MMMN = ETS(GDP ~ error("M") + trend("M") + season("N")),
    MADN = ETS(GDP ~ error("M") + trend("Ad") + season("N")),
    BOXLOG = ETS(box_cox(GDP,0)),
  )
```

```
## Warning: 1 error encountered for MAM
## [1] A seasonal ETS model cannot be used for this data.
```

```
## Warning: 1 error encountered for MAD
## [1] Invalid season type
```

```
china_soars <- forecast(china_fitter, h=30)


china_soars %>%
  autoplot(china_gdp_tb, level=NULL) +
    labs(title = "30 Year Forecast of Multiple Different ETS Models",
       x = "Year",
       y = "USD") +
    scale_y_continuous(labels = scales::comma_format()) +
    theme_minimal()
```

```
## Warning: Removed 60 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



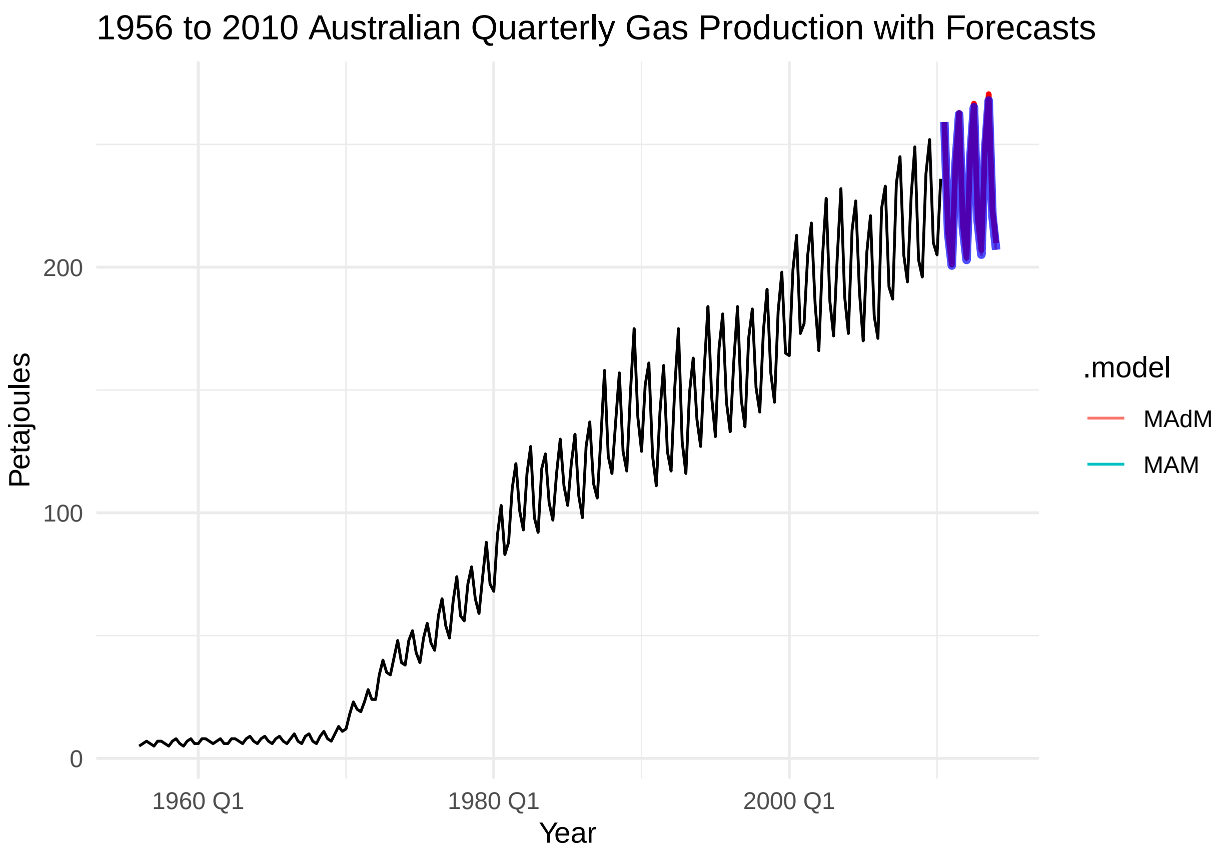
8.7: Find an ETS model for the Gas data from aus_production and forecast the next few years

"Why is multiplicative seasonality necessary here? Experiment with making the trend damped. Does it improve the forecasts?"

Answer: All the gas data below is measured in petajoules, where one PJ equals one quadrillion joules.

Dampening the seasoning trend on a dataset that has a strong seasonal component, which has been consistent since the start of the dataset and has only become more pronounced as overall volume has grown, would make this forecast unreliable. The growth trend has been growing, not flat lining or decreasing, so a dampened seaonality means the forecast will, as shown below, quickly deviate from the pattern.

```r
data("aus_production")

aus_gas <- aus_production |>
  select(Gas)

aus_gas_fit <- aus_gas |>
  model(
    MAM = ETS(Gas ~ error("M") + trend("A") + season("M")),
    MAdM = ETS(Gas ~ error("M") + trend("Ad") + season("M"))
  )
aussi_forecast <- forecast(aus_gas_fit, h=15)

aussi_forecast %>%
  autoplot(aus_gas, level=NULL) +
  autolayer(aussi_forecast %>% filter(.model == "MAM"),
            color = "red",
            size = 1,
            linetype = "dashed",
            level = NULL) +
  autolayer(aussi_forecast %>% filter(.model == "MAdM"),
            color = "blue",
            size = 1.5,
            linetype = "solid",
            alpha = 0.7,
            level = NULL) +
  labs(title = "1956 to 2010 Australian Quarterly Gas Production with Forecasts",
       x = "Year",
       y = "Petajoules") +
  scale_y_continuous(labels = scales::comma_format()) +
  theme_minimal()
```



8.8: Analysis of aus_retail using ETS modeling

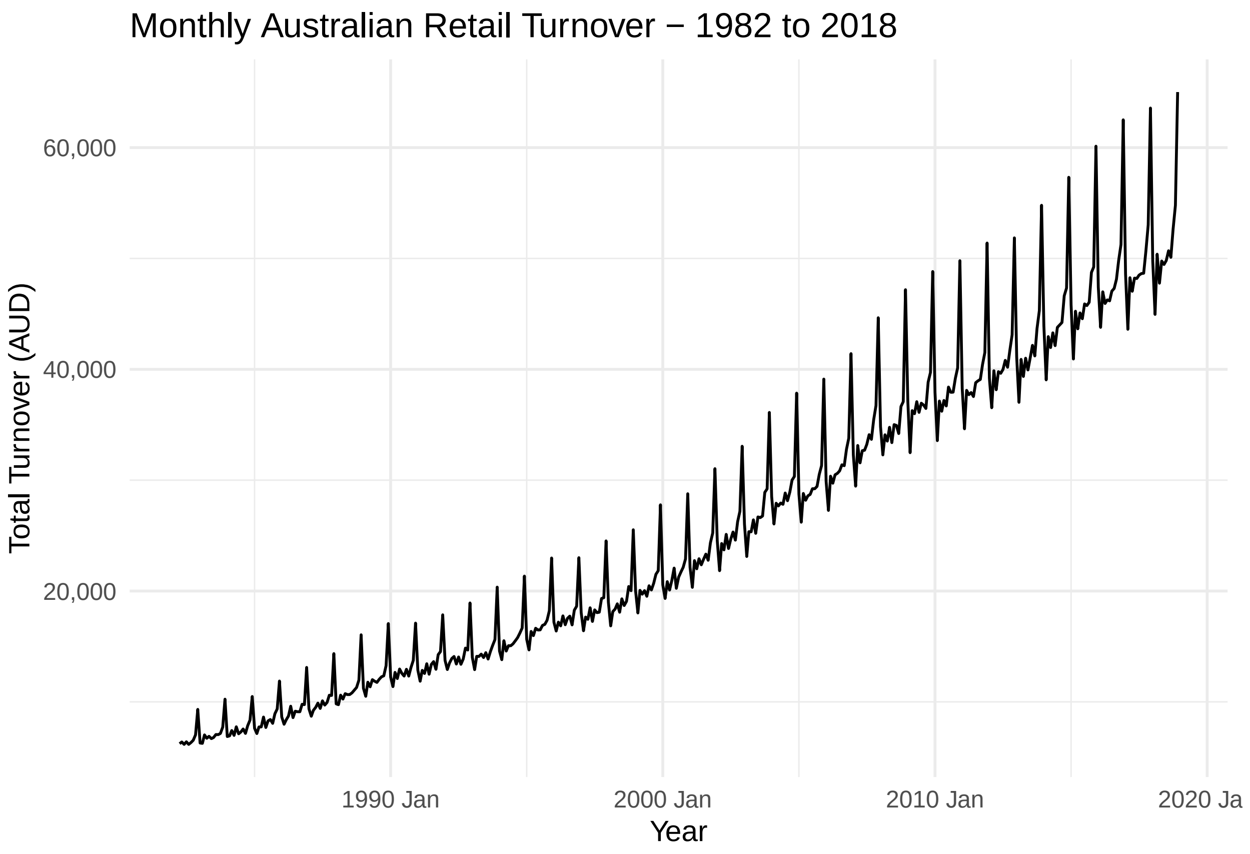
This exercise consistents of components A through E and is based on the aus_retail dataset

```
data("aus_retail")
```

A. Why is multiplicative seasonality necessary for this series?

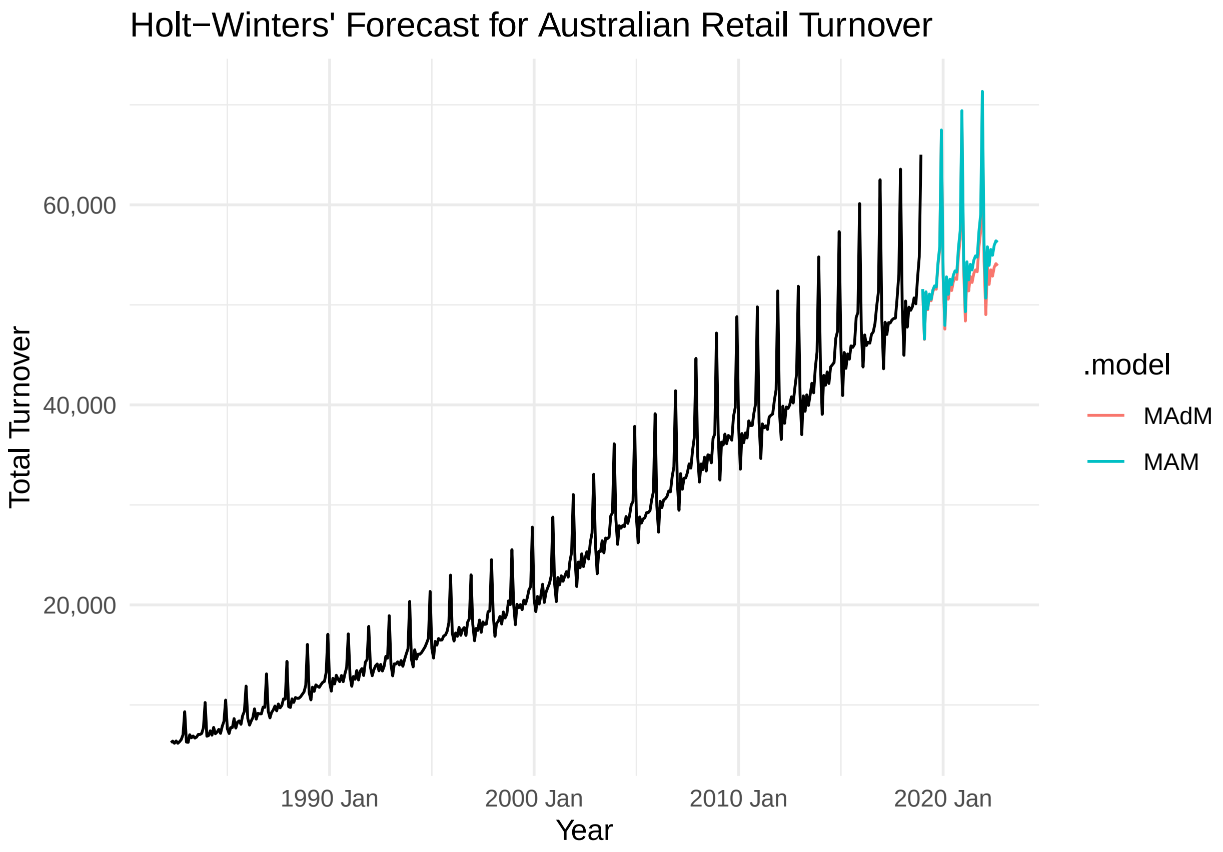
Answer: this trend has a sesonality trend that increases proportionally with the overall amount of turnover. Multiplicative sesonality can account for this type of trend by forecasting a continuation of that proportional growth.

```r
aus_turnover <- aus_retail %>%
  index_by(Month) %>%
  summarise(turnover_sum = sum(Turnover))

autoplot(aus_turnover, turnover_sum) +
  labs(title = "Monthly Australian Retail Turnover - 1982 to 2018",
       x = "Year",
       y = "Total Turnover (AUD)") +
  scale_y_continuous(labels = scales::comma_format()) +
  theme_minimal()
```



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

```r
aus_fitter <- aus_turnover |>
  model(MAM = ETS(turnover_sum ~ error("M") + trend("A") + season("M")),
        MAdM = ETS(turnover_sum ~ error("M") + trend("Ad") + season("M")))

turnt_forecast <- forecast(aus_fitter, h=45)

# Plotting the forecast with the original data
autoplot(turnt_forecast, aus_turnover, level=NULL) +
  labs(title = "Holt-Winters' Forecast for Australian Retail Turnover",
       x = "Year",
       y = "Total Turnover") +
  scale_y_continuous(labels = scales::comma_format()) +
  theme_minimal()
```



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

Based on RMSE alone, I would take the dampened MAdM model. However, in totality and based on the overall components of the data, I would take the MAM and not add a sesonality component. Looking at the forecasts above, I;m more inclined to believe the MAM over the MAdM. Sometimes business instinct over what looks right will triumph over one statistic breaking against you. However, bad habits form when you simply disregard data for your own judgement on a regular basis.

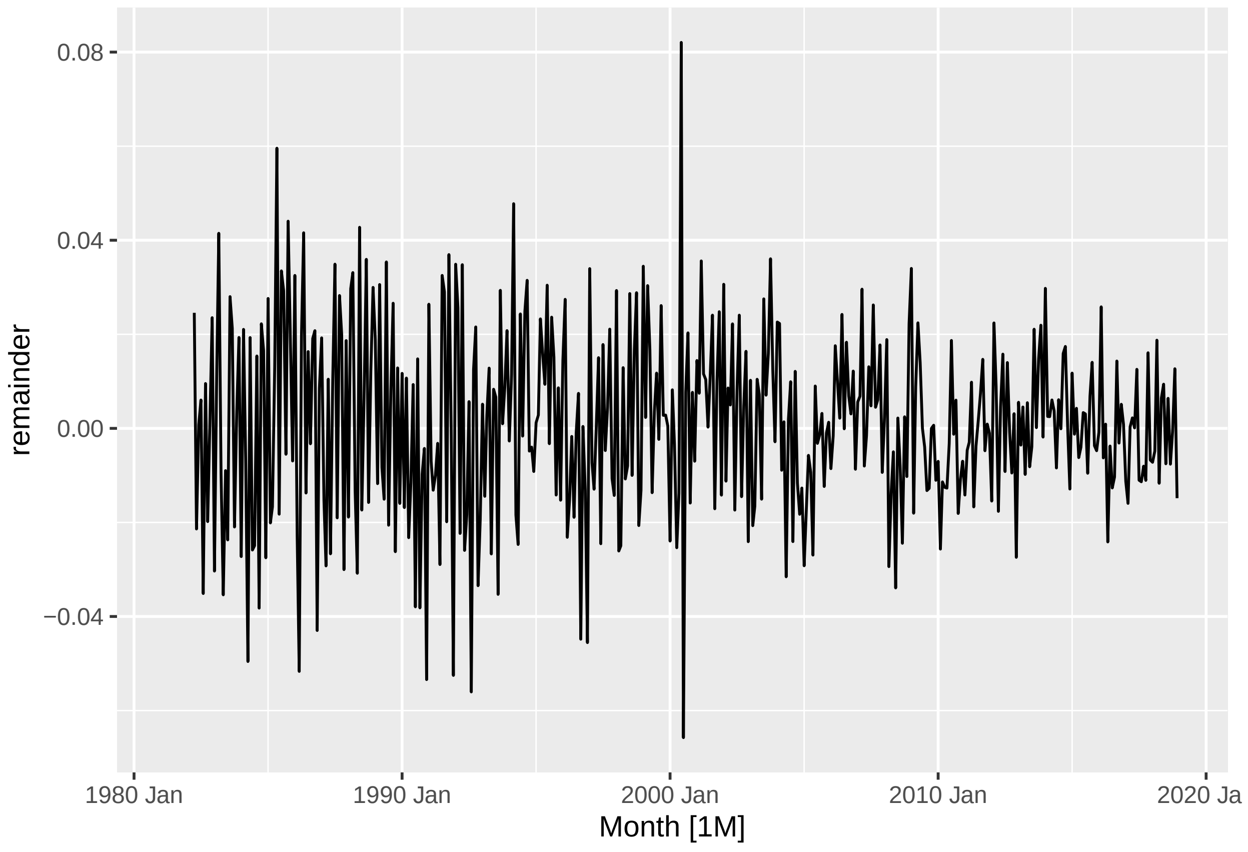
```
aus_fitter |>
  accuracy()
```

```
## # 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 MAM    Training  7.76  421.  327. 0.0486  1.55 0.261 0.294 -0.101
## 2 MAdM   Training 40.3   428.  327. 0.180   1.55 0.262 0.298 -0.0886
```

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

The remainders are quite negligible, coming in with a mean right around zero. It's also been getting tighter over the last 20 years of the data, suggesting the MAM is performing quite well.

```
aus_fitter |>
  components() |>
  filter(.model == "MAM") |>
  select(remainder) |>
  autoplot(.vars = remainder, na.rm = TRUE)
```



E. 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?"

Answer:

The SNAIVE RMSE from exercise 7 in section 5.11 was was 8,089 and the below produces 5,928. That's a significant improvement in the overall error.

```
aus_turnover_tester <- aus_turnover %>%
  mutate(Month = yearmonth(Month)) %>%
  as_tsibble(index = Month)

aus_train <- aus_turnover_tester %>% filter(Month <= yearmonth("2010 Dec"))
aus_test <- aus_turnover_tester %>% filter(Month > yearmonth("2010 Dec"))

aus_fitter_tester <- aus_train |>
  model(MAM = ETS(turnover_sum ~ error("M") + trend("A") + season("M")),
       MAdM = ETS(turnover_sum ~ error("M") + trend("Ad") + season("M")))

tester_forecast <- forecast(aus_fitter_tester, h = nrow(aus_test))

original_rmse <- accuracy(tester_forecast, aus_test) %>%
  filter(.model == "MAdM") %>%
  pull(RMSE)
```

```
original_rmse
```

```
## [1] 5927.808
```

8.9: Box-Cox versus ETS on sesonally adjusted data

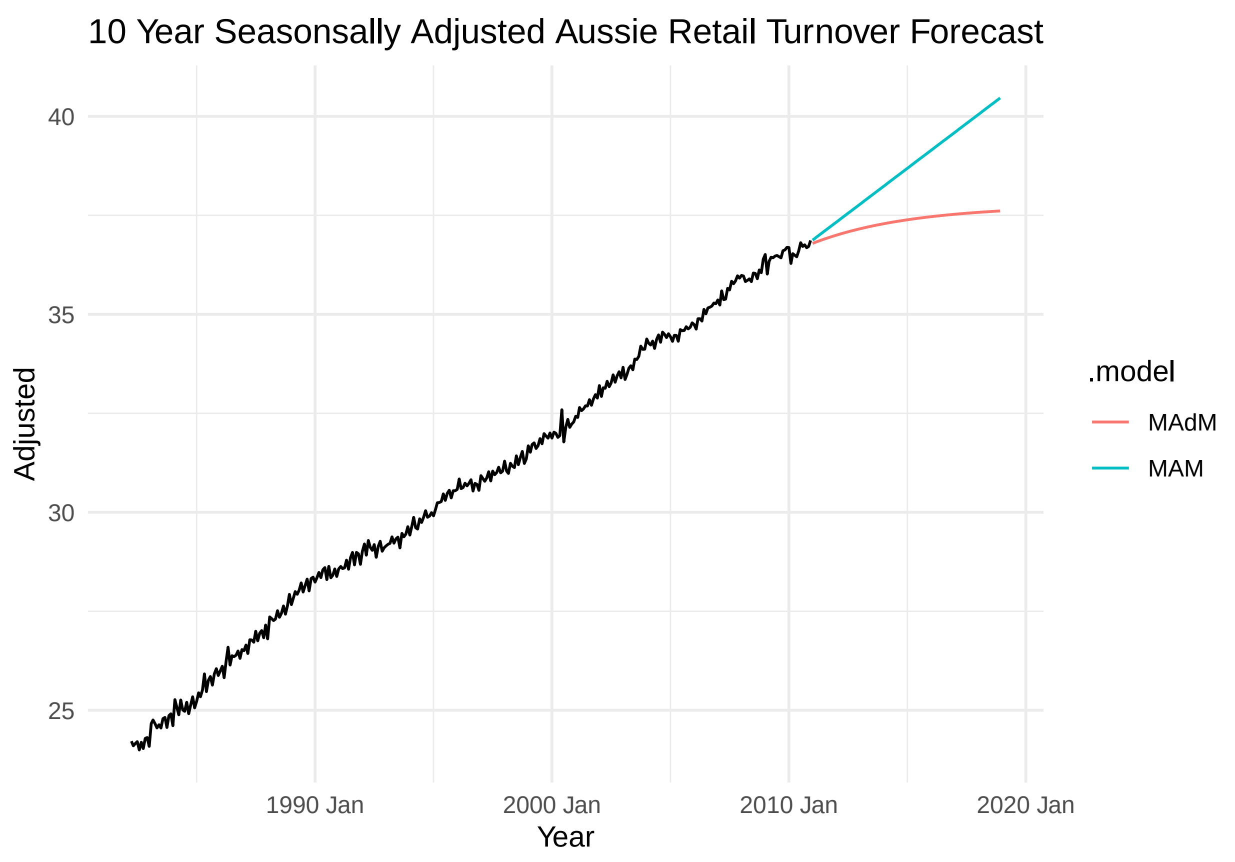
"For the same retail data, try an STL decomposition applied to the Box-Cox transformed series, followed by ETS on the seasonally adjusted data. How does that compare with your best previous forecasts on the test set?"

Answer:

Original RMSE from from 8.8.E: 5927.808 MAM Model RMSE: 5480.975 MAdM Model RMSE: 6421.878

The revised RMSE based on the seasonally adjusted data does not compare favorably. The MAM model performed the best overall because it properly fits the seasonality trend of this dataset, where as the dampening component of MAdM is causing an arch that flattens the future growth.

```
aus_la <- aus_train %>%
  features(turnover_sum, features = guerrero) %>%
  pull(lambda_guerrero)

# Apply the Box-Cox transformation to the monthly data
aus_box_trans <- aus_train %>%
  mutate(turnover_bc = box_cox(turnover_sum, lambda = aus_la))

# Fit the STL decomposition model on the transformed data
stl_fit_abt <- aus_box_trans %>%
  model(STL_turnover = STL(turnover_bc ~ trend() + season(window = "periodic")))

# Extract the seasonally adjusted component
season_adjust <- stl_fit_abt %>%
  components() %>%  # Get decomposition components
  select(Month, season_adjust)

ets_fit <- season_adjust %>%
  model(MAM = ETS(season_adjust ~ error("M") + trend("A") + season("N")),
        MAdM = ETS(season_adjust ~ error("M") + trend("Ad") + season("N")))


ets_forecast <- ets_fit %>%
  forecast(h = 96)

ets_forecast %>%
  autoplot(season_adjust, level=NULL) +
  labs(title = "10 Year Seasonsally Adjusted Aussie Retail Turnover Forecast",
       x = "Year",
       y = "Adjusted") +
  scale_y_continuous(labels = scales::comma_format()) +
  theme_minimal()
```



```r
ets_forecast_inv <- ets_forecast %>%
  filter(.model == "MAM") %>%
  mutate(.mean = inv_box_cox(.mean, aus_la))

ets_forecast_madm <- ets_forecast %>%
  filter(.model == "MAdM") %>%
  mutate(.mean = inv_box_cox(.mean, aus_la))

aus_test_df <- as_tibble(aus_test)
ets_forecast_inv_df <- as_tibble(ets_forecast_inv)

aus_test_madm <- as_tibble(aus_test)
ets_forecast_inv_madm <- as_tibble(ets_forecast_madm)

test_forecast_compare <- aus_test_df %>%
  select(Month, turnover_sum) %>%
  left_join(ets_forecast_inv_df, by = "Month")

test_forecast_compare_madm <- aus_test_madm %>%
  select(Month, turnover_sum) %>%
  left_join(ets_forecast_inv_madm, by = "Month")

mam_rmse <- sqrt(mean((test_forecast_compare$.mean - test_forecast_compare$turnover_sum)^2))
madm_rmse <- sqrt(mean((test_forecast_compare_madm$.mean - test_forecast_compare_madm$turnover_sum)^2))

cat("Original RMSE from from 8.8.E: ", original_rmse, "\n")
```

```
## Original RMSE from from 8.8.E:  5927.808
```

```r
cat("MAM Model RMSE:", mam_rmse, "\n")
```

```
## MAM Model RMSE: 5480.975
```

```r
cat("MAdM Model RMSE:", madm_rmse, "\n")
```

```
## MAdM Model RMSE: 6421.878
```