Week 6

Chi Hang(Philip) Cheung

2025-03-08

```
library(fpp3)
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.4.3
```

8.1

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

a) Use the `ETS()` function to estimate the equivalent model for simple exponential smoothing. Find the optimal values of alpha and level 0, and generate forecasts for the next four months.

Ans: alpha = 0.322; initial level = 100647

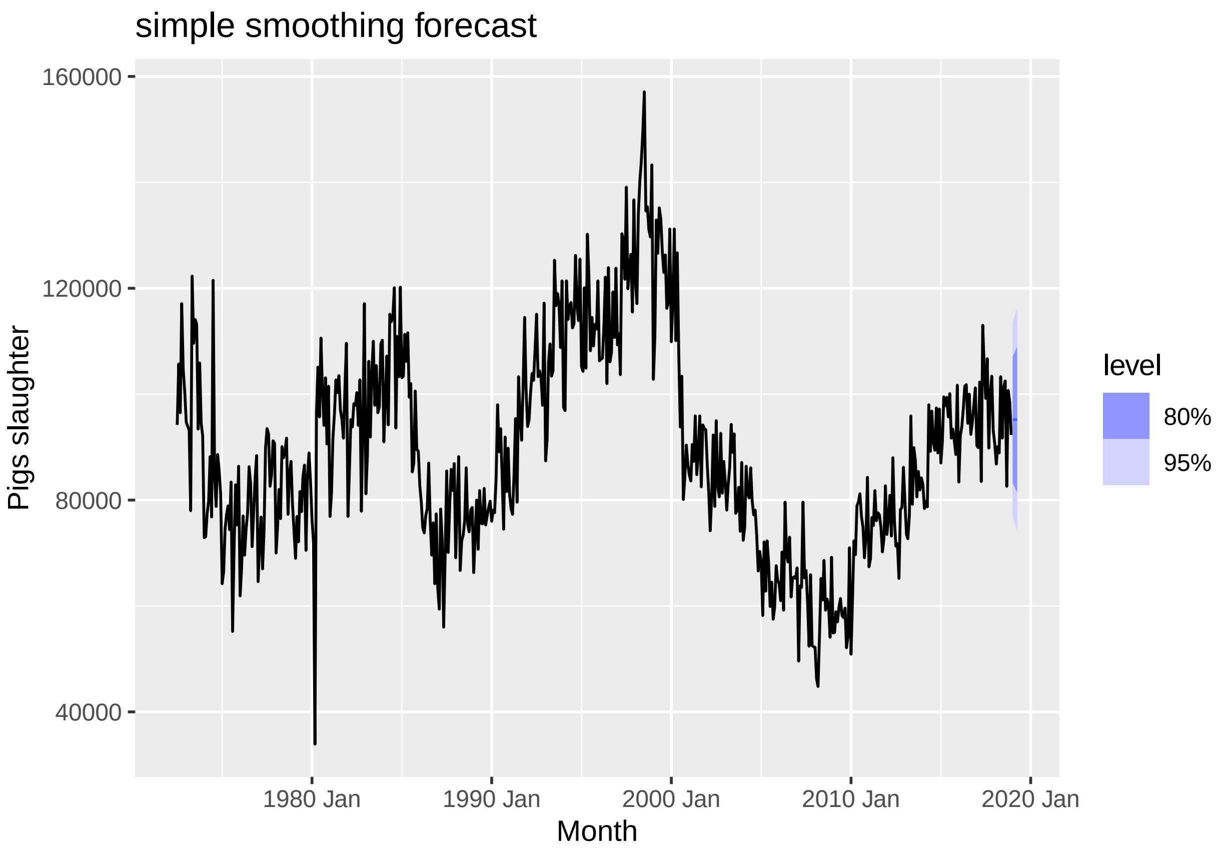
```
og<- aus_livestock %>%
  filter(Animal == 'Pigs', State=='Victoria')

fit<- og %>%
  model(ETS(Count ~ error('A') + trend('N') +season('N')))

#To extract the alpha and l0 values:
report(fit)
```

```
## 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
```

```
#forecasting for 4 months
fc<- fit %>%
  forecast(h = '4 month')
```

```r
fc %>%
  autoplot(og)+
  autolayer(fc, series = 'Forecast')+
  ggtitle('simple smoothing forecast')+
  ylab('Pigs slaughter')
```

```
## Warning in ggdist::geom_lineribbon(without(intvl_mapping, "colour_ramp"), :
## Ignoring unknown parameters: ‘series‘

## Warning in geom_line(mapping = without(mapping, "shape"), data =
## unpack_data(object[single_row[["FALSE"]], : Ignoring unknown parameters:
## ‘series‘

## Scale for fill_ramp is already present.
## Adding another scale for fill_ramp, which will replace the existing scale.
```



b) Compute a 95% prediction interval for the first forecast using $y\pm1.96sy$ ±1.96s where ss is the standard deviation of the residuals. Compare your interval with the interval produced by R.

Ans: Manual calculation for the 95% interval is [76871.01 to 113502.1]. The interval computed by R is "[76854.79, 113518.3]". Both are relatively close to each other.

```r
#To get the y_head( the mean of the first forecast)
y_head<- fc$.mean[1]
#Calculate the residual
residual<-residuals(fit)
#calculate the standard deivation of the residual
sd<- sd(residual$.resid)
#compute the upper and lower range
lower_range<- y_head - 1.96 * sd
upper_range<- y_head + 1.96 * sd
cat("prediction interval:[",lower_range, 'to', upper_range,"]")
```

```
## prediction interval:[ 76871.01 to 113502.1 ]
```

```r
#95% interval computed by R
fc %>%
  hilo(level = 95) %>%
  pull('95%') %>% head(1)
```

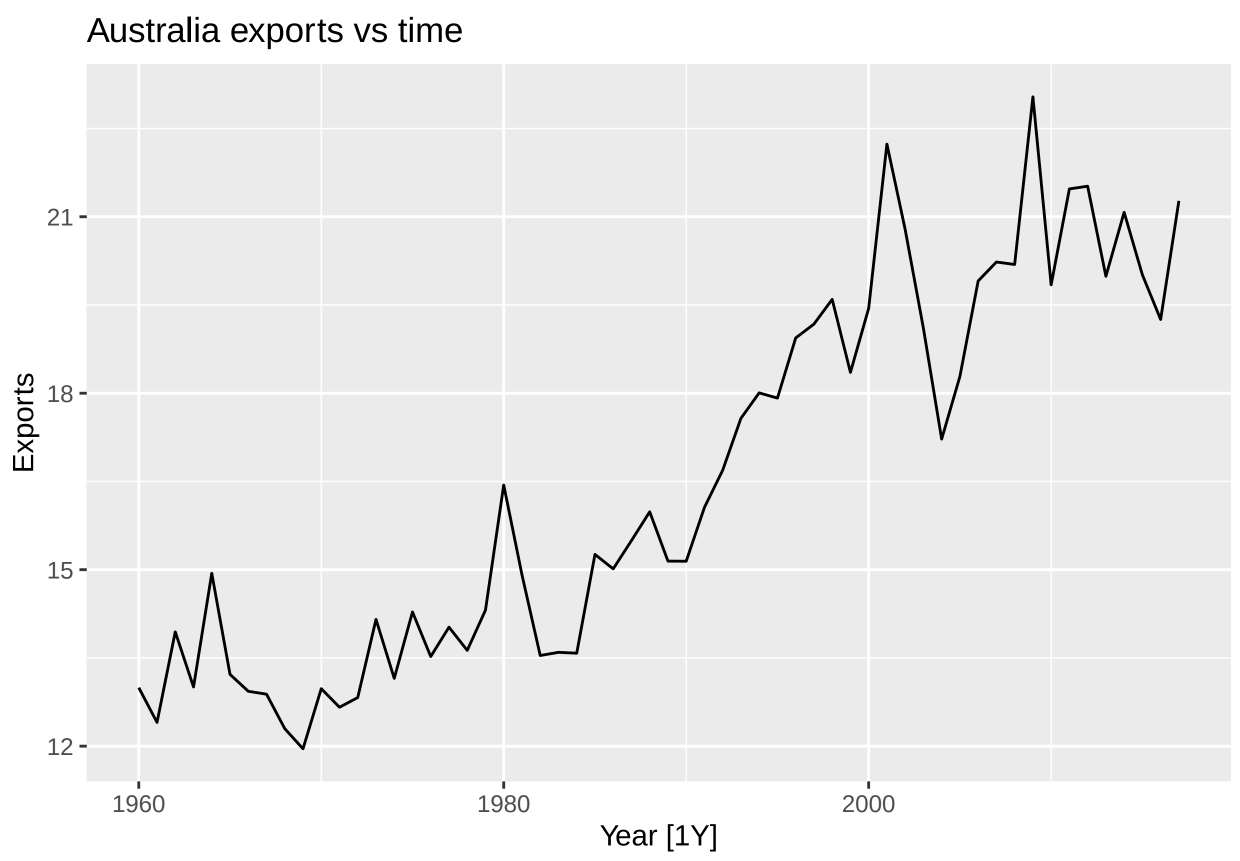
```
## <hilo[1]>
## [1] [76854.79, 113518.3]95
```

8.5

a) Plot the Exports series and discuss the main features of the data.

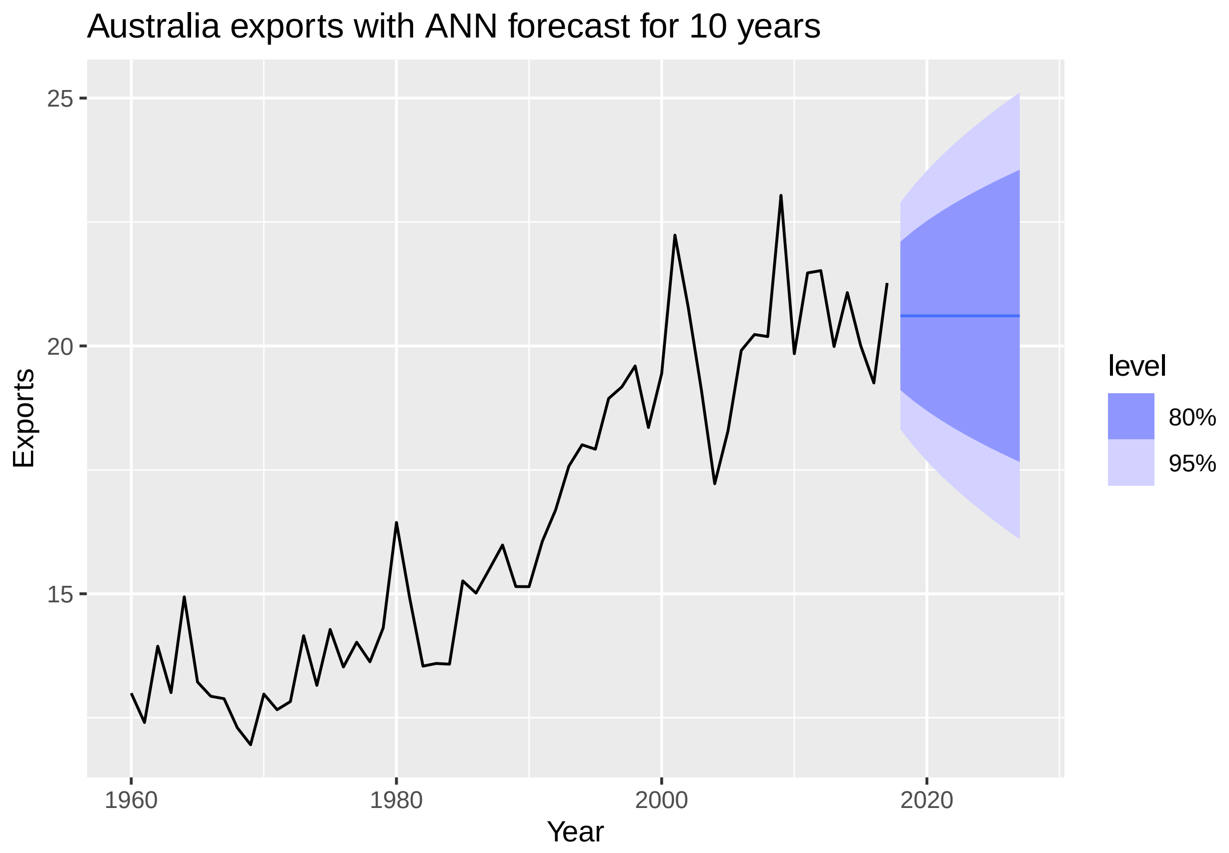
   Ans: There is an upward trend throughout the years. There does not seem to be a seasonality in the dataset.

```r
aus_export<- global_economy %>%
  filter(Country == 'Australia')

autoplot(aus_export, Exports) +
  ggtitle('Australia exports vs time')
```



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

```
aus_exp_fit<- aus_export %>%
  model(ETS(Exports ~ error('A') + trend('N') + season('N')))

aus_exp_fc<- aus_exp_fit %>%
  forecast(h = 10)

autoplot(aus_exp_fc, aus_export)+
  ggtitle('Australia exports with ANN forecast for 10 years')
```



c) Compute the RMSE values for the training data. Ans: RMSE = 1.146794

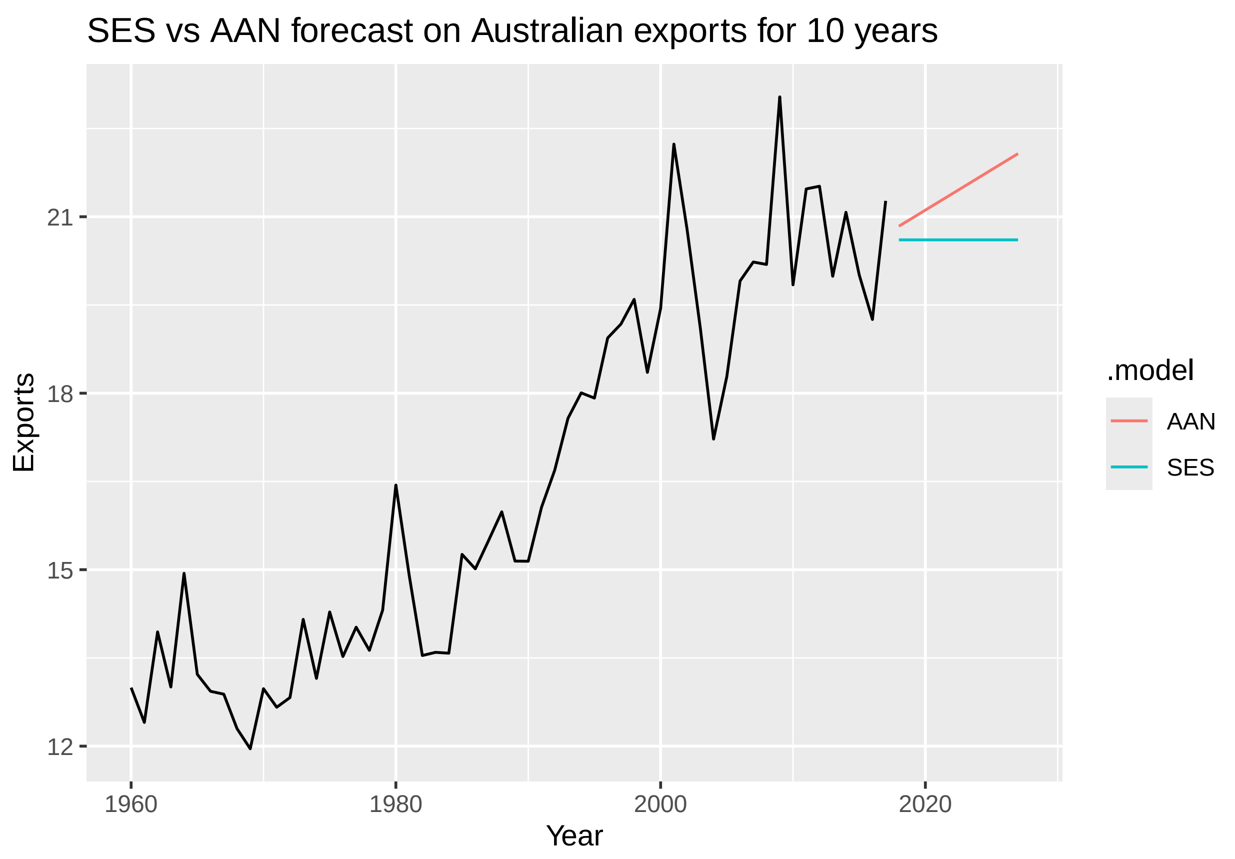
```
aus_exp_fit %>%
  accuracy() %>%
  pull('RMSE')
```

```
## [1] 1.146794
```

d) Compare the results to those from an ETS(A,A,N) model.

Ans: The AAN method has RMSE = 1.116727 while the ANN(SES) = 1.146794. The same relative results are seen across other error estimate parameters. The AAN method might be more appropriately fitted due how it captures the upward trend of the data. The SES weights more heavily on the last data point of the plot while considering less and less of the old data.

```
fit_vs<- aus_export %>%
  model(
    SES = ETS(Exports ~ error('A') + trend('N') + season('N')),
    AAN = ETS(Exports ~ error('A') + trend('A') + season('N')))
#Forecast for 10 years
fc_vs<- fit_vs %>%
  forecast(h=10)

autoplot(fc_vs, aus_export, level = NULL) +
  labs(title='SES vs AAN forecast on Australian exports for 10 years')
```



```r
accuracy(fit_vs)
```

```
## # A tibble: 2 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 Australia SES    Training  2.32e-1  1.15 0.914  1.09   5.41 0.928 0.928 0.0125
## 2 Australia AAN    Training -7.46e-7  1.12 0.893 -0.387  5.39 0.907 0.904 0.109
```

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

   Ans: The AAN method seems to model the data better than the ANN due to lower error estimation.

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.

   Ans: SES prediction interval:[18.38671 to 22.8276];

   AAN prediction interval:[18.39926 to 22.81506];

   R computation = [18.3197, 22.89462]

```r
#95% interval for SES method:
y_head_ses<- fc_vs$.mean[1]
residual_ses<- residuals(fit_vs) %>% filter(.model=='SES')
sd_ses<- sd(residual_ses$.resid)

lower_range_ses<- y_head_ses - 1.96 * sd_ses
upper_range_ses<- y_head_ses + 1.96 * sd_ses
```

```r
cat("SES prediction interval:[",lower_range_ses, 'to', upper_range_ses,"]")
```

```
## SES prediction interval:[ 18.38671 to 22.8276 ]
```

```r
#95% interval for AAN method:
y_head_ses<- fc_vs$.mean[1]
residual_ses<- residuals(fit_vs) %>% filter(.model=='AAN')
sd_ses<- sd(residual_ses$.resid)

lower_range_ses<- y_head_ses - 1.96 * sd_ses
upper_range_ses<- y_head_ses + 1.96 * sd_ses

cat("AAN prediction interval:[",lower_range_ses, 'to', upper_range_ses,"]")
```

```
## AAN prediction interval:[ 18.39926 to 22.81506 ]
```

```r
#computing 95% interval through R:
fc_vs %>%
  hilo(level = 95) %>%
  pull('95%') %>%
  head(1)
```

```
## <hilo[1]>
## [1] [18.3197, 22.89462]95
```

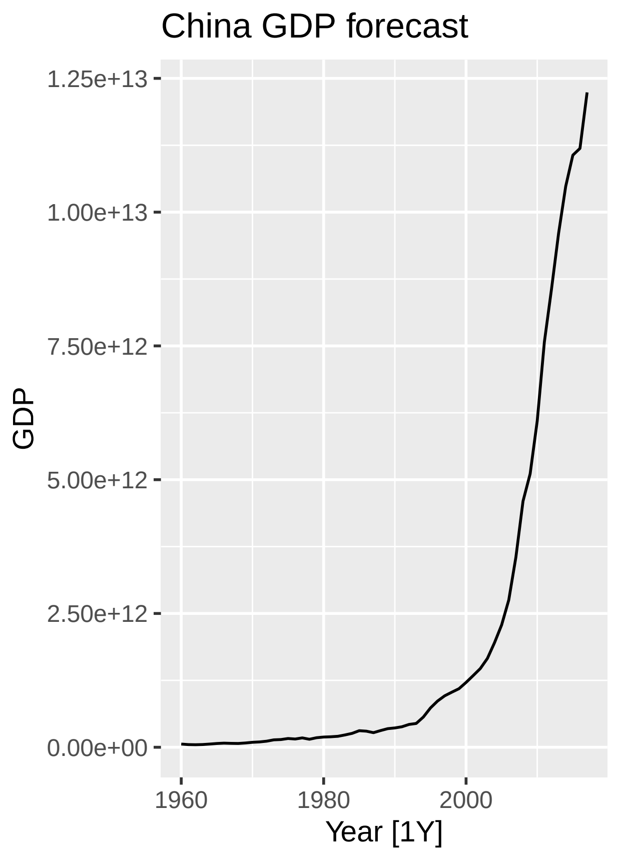
8.6 Forecast Chinese GDP from global_economy data set

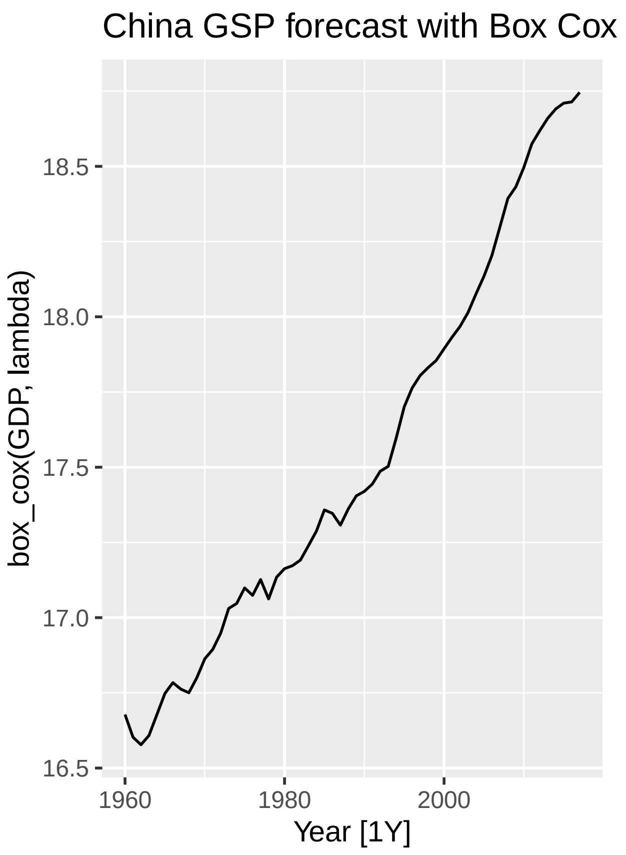
exploratory plot

```r
china<- global_economy %>%
  filter(Country =='China')

p1<- autoplot(china, GDP)+
  labs(title = 'China GDP forecast')
```

With Box_cox Transformation:

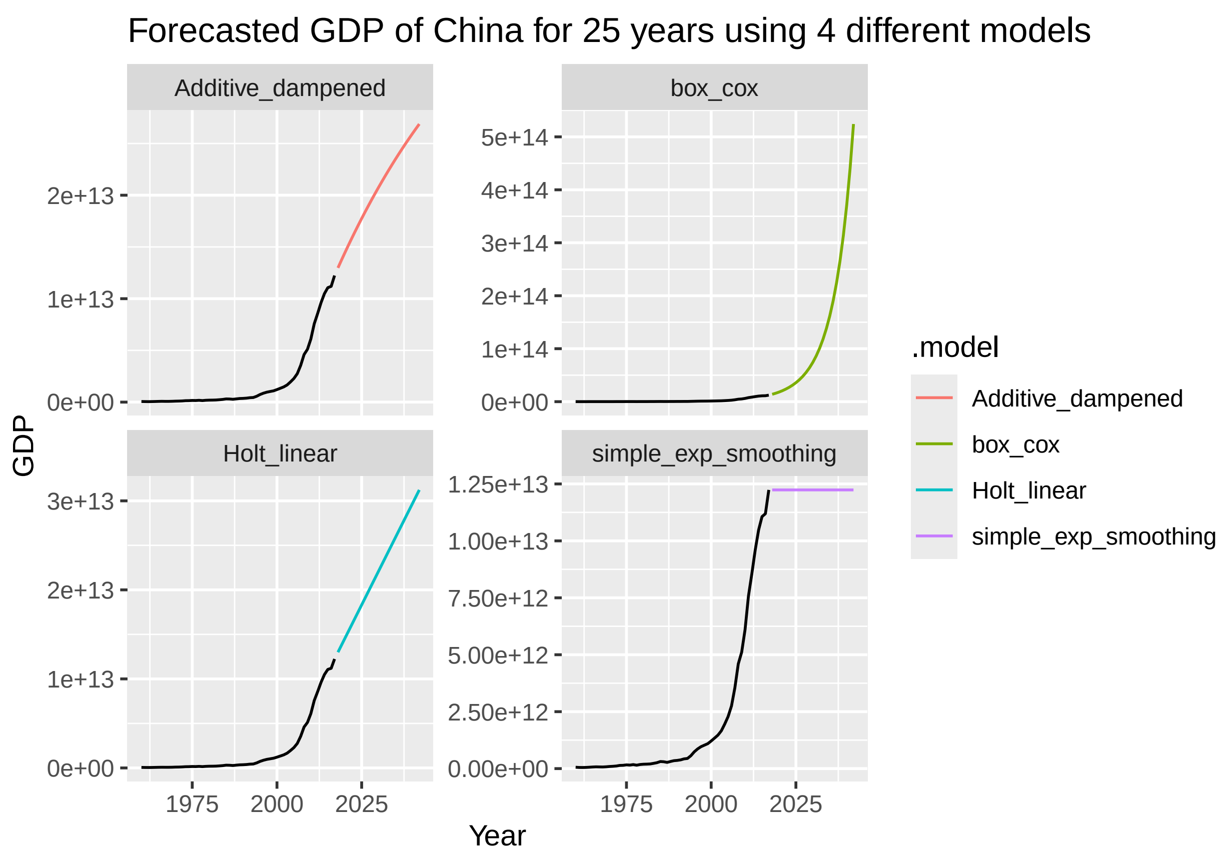
```r
#To find the best lambda
lambda<- china %>%
  features(GDP, features = guerrero) %>%
  pull(lambda_guerrero)

p2<- china %>%
  autoplot(box_cox(GDP, lambda))+
  labs(title = 'China GSP forecast with Box Cox transformation')

grid.arrange(p1, p2, ncol=2)
```





Computing multiple models

```
fit_china<- china %>%
  model(
    simple_exp_smoothing = ETS(GDP ~ error('A') + trend('N') + season('N')),
    Holt_linear = ETS(GDP ~ error('A') + trend('A') + season('N')),
    Additive_dampened = ETS(GDP ~ error('A') + trend('Ad') + season('N')),
    box_cox = ETS(box_cox(GDP, lambda))
  )

fc_china<- fit_china %>%
  forecast(h = 25)

autoplot(fc_china, china, level = NULL)+
  labs(title = 'Forecasted GDP of China for 25 years using 4 different models')+
  facet_wrap(~.model, scale='free_y')
```

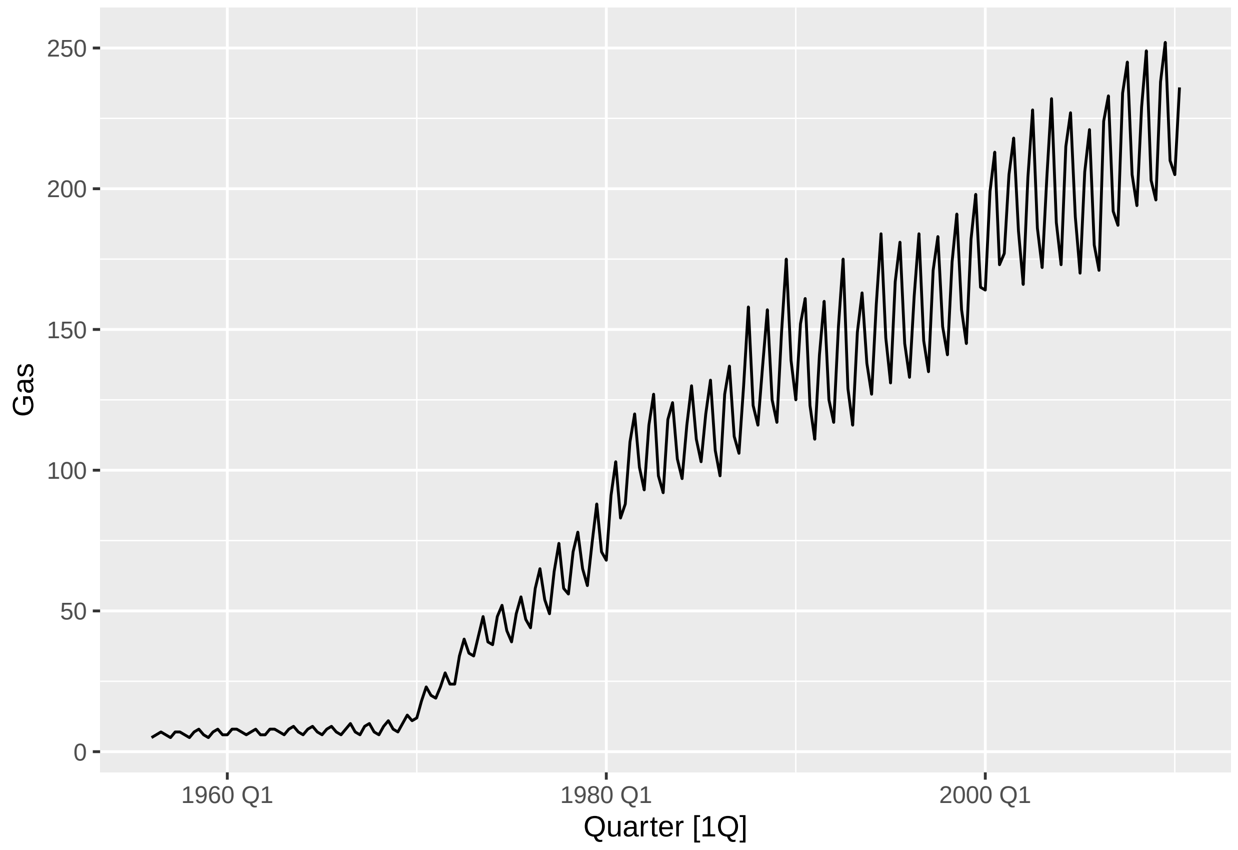


8.7 Find an ETS model for the Gas data from aus_production and forecast the next few years. Why is multiplicative seasonality necessary here? Experiment with making the trend damped. Does it improve the forecasts?

Ans: Multiplicity is needed because of the seasonality pattern observed from the original data. The RSME values for both methods are relatively close. 4.1898 and 4.2165 for dampened and un-dampened, respectively. However, the un-dampened method shows slightly lower in RMSE but then higher in some other error measurements. Both methods will product a relatively good fit model.

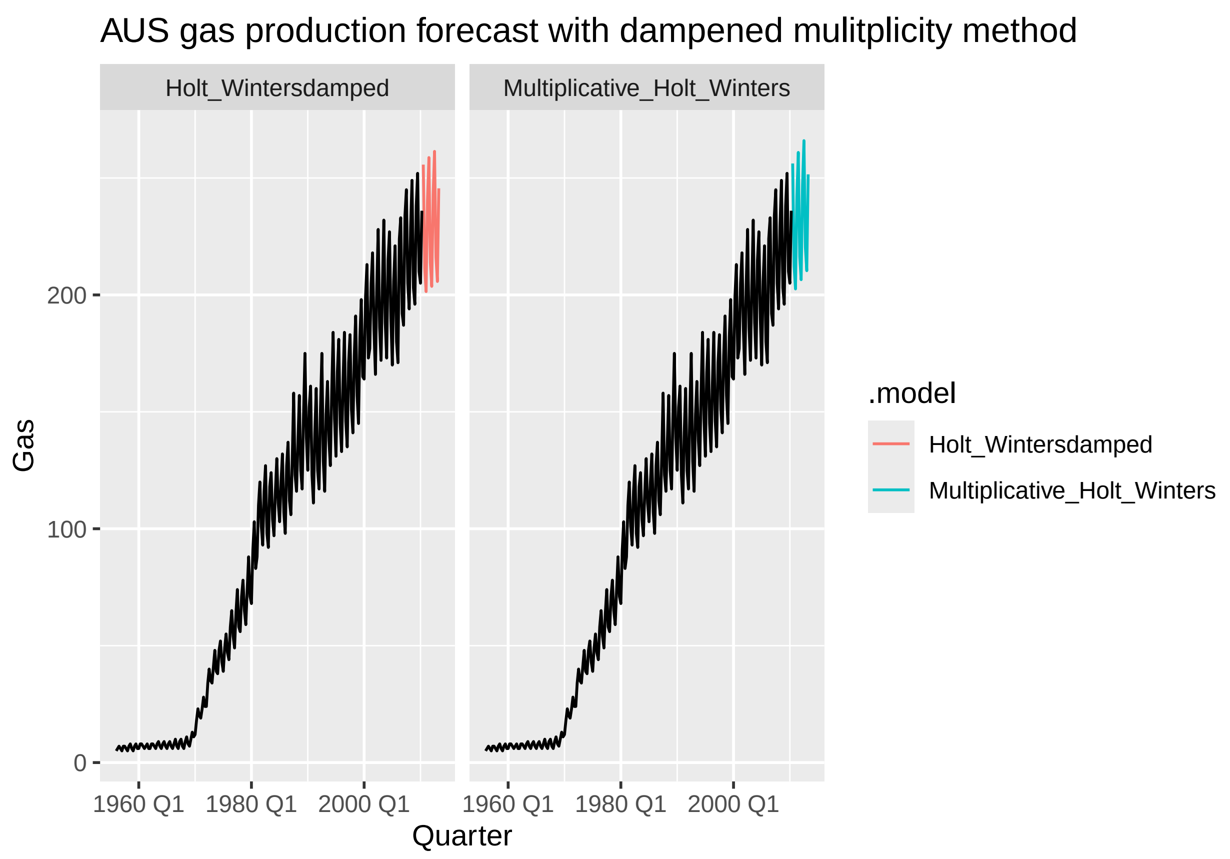
Exploratory data:

```
autoplot(aus_production, Gas)
```



To compute multiple models with seasonality for fitting and forecasting:

```
fit_gas<- aus_production %>%
  model(
    Multiplicative_Holt_Winters = ETS(Gas ~ error('A') + trend('A')+ season('M')),
    Holt_Wintersdamped = ETS(Gas ~ error('A') + trend('Ad')+ season('M'))
    )

fc_gas<- fit_gas %>%
  forecast(h= 12)

autoplot(fc_gas, aus_production, level = NULL)+
  facet_wrap(~ .model)+
  ggtitle('AUS gas production forecast with dampened mulitplicity method')
```



To compare the RMSE:

```
accuracy(fit_gas)
```

```
## # 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 Multiplicative_Holt_W~ Trai~ 0.218  4.19  2.84 -0.920  5.03 0.510 0.553 0.0405
## 2 Holt_Wintersdamped     Trai~ 0.548  4.22  2.81  1.32   4.11 0.505 0.556 0.0265
```

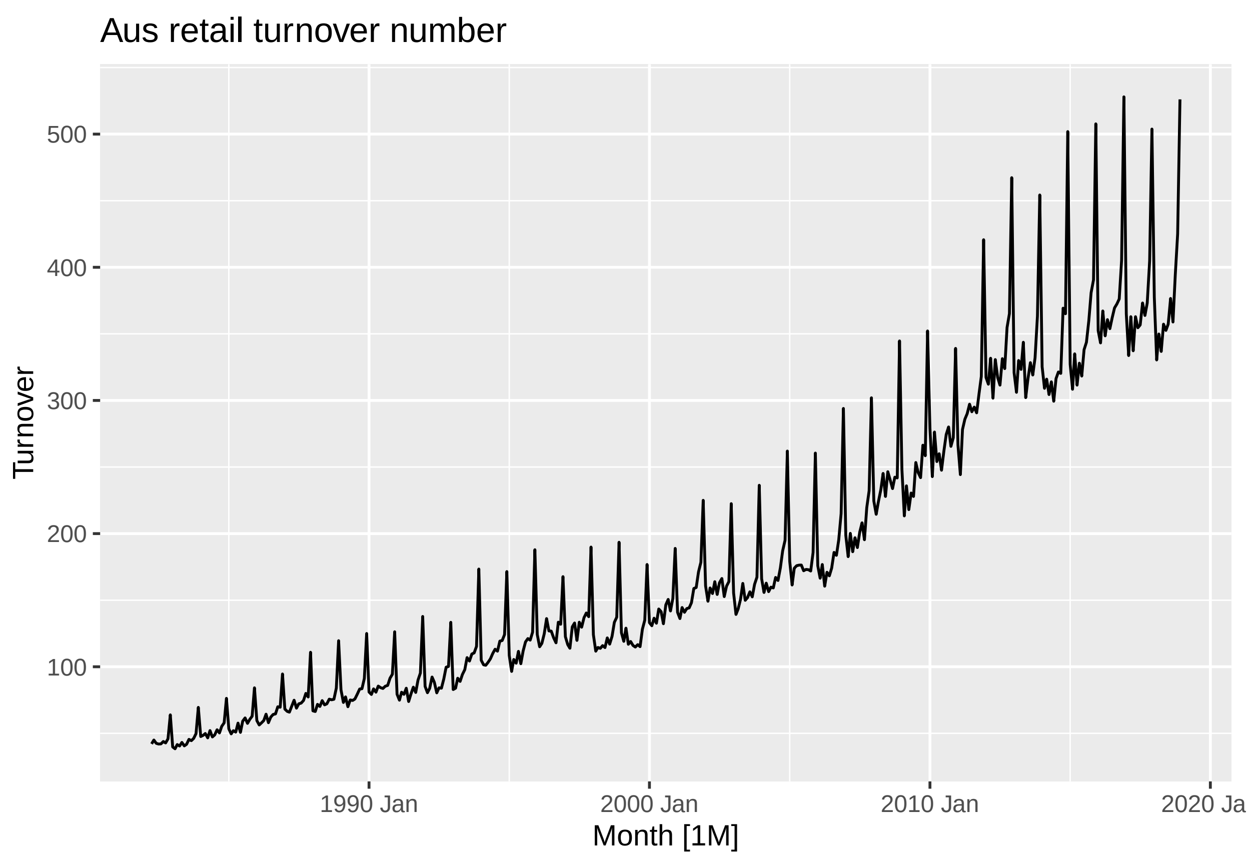
8.8) Recall your retail time series data (from Exercise 7 in Section 2.10).

a) Why is multiplicative seasonality necessary for this series?

   Ans:This series exhibits seasonality pattern. Therefore mulitplicity is needed to forecast and to fit the model.

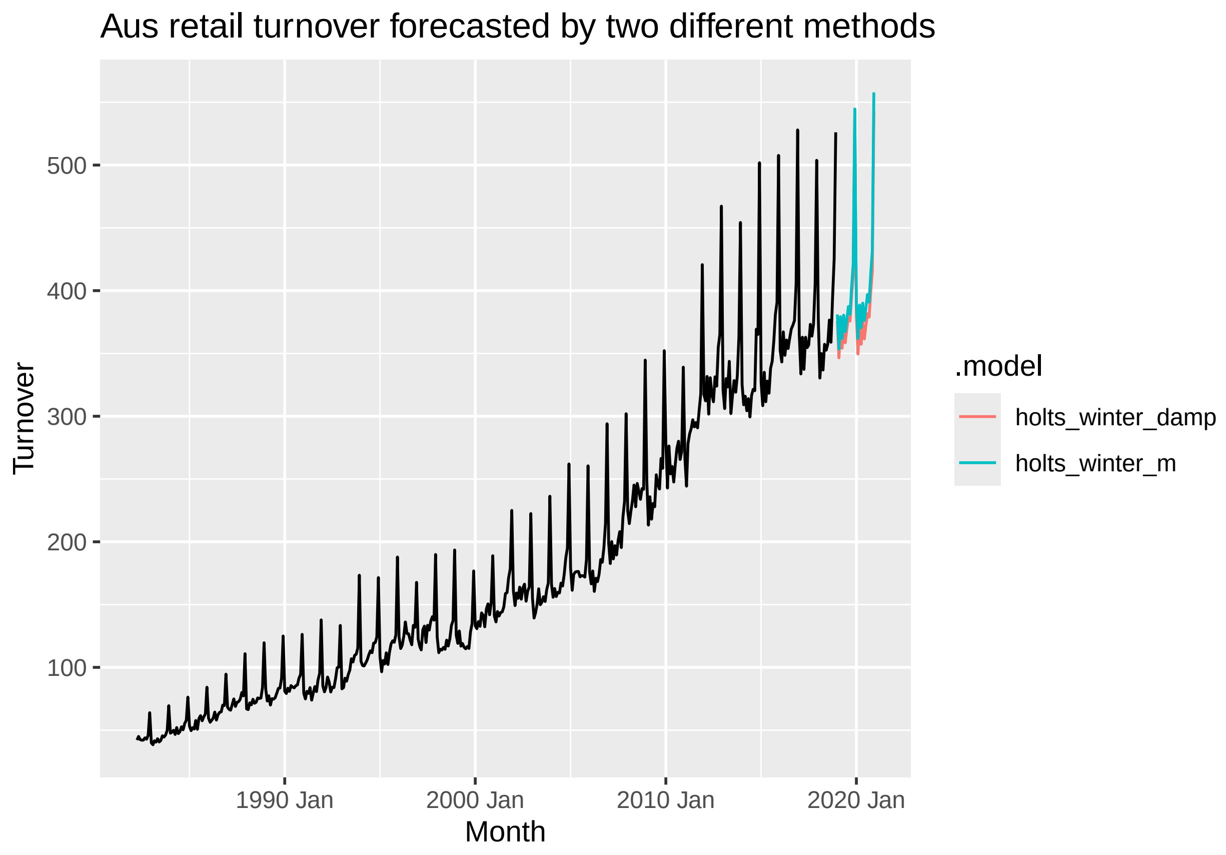
```
#importing the series with a new random seed number
set.seed(128)
myseries <- aus_retail |>
  filter(`Series ID` == sample(aus_retail$`Series ID`,1))
myseries %>%
  autoplot()+ggtitle('Aus retail turnover number')
```

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



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

```
fit_myseries<- myseries %>%
  model(
    holts_winter_m = ETS(Turnover ~ error('A') + trend('A')+ season('M')),
    holts_winter_damp = ETS(Turnover ~ error('A') + trend('Ad') + season('M'))
  )

fit_myseries %>%
  forecast(h = 24) %>%
  autoplot(myseries, level = NULL)+
  labs(title = 'Aus retail turnover forecasted by two different methods')
```



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

Ans: comparing the RMSEs for both methods, the holts winter with dampening is the better choice due to the lower RMSE.

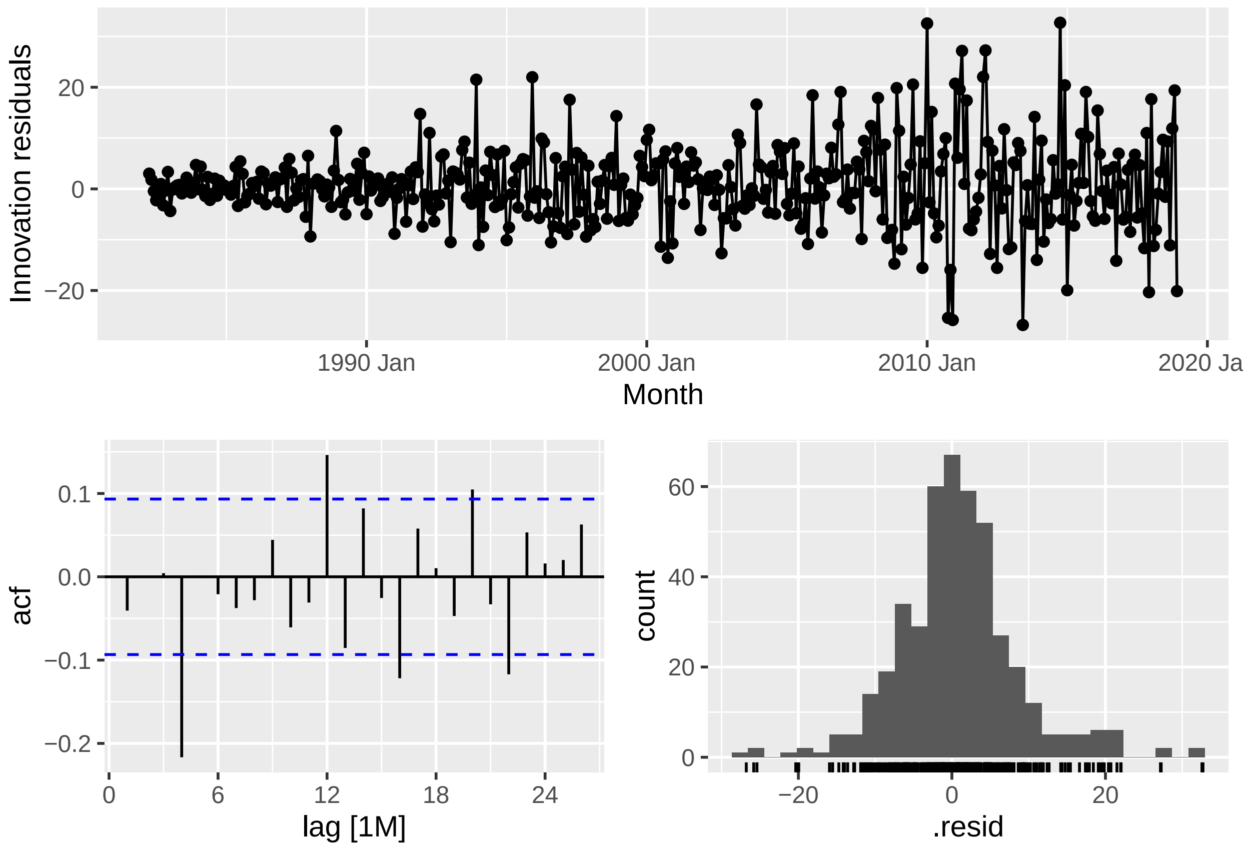
```r
accuracy(fit_myseries)
```

```
## # A tibble: 2 x 12
##   State  Industry .model .type    ME  RMSE   MAE   MPE  MAPE  MASE RMSSE    ACF1
##   <chr>  <chr>    <chr>  <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>
## 1 Weste~ Other r~ holts~ Trai~ 0.556  8.56  6.11 0.242  3.71 0.484 0.492  0.143
## 2 Weste~ Other r~ holts~ Trai~ 0.781  7.92  5.67 0.338  3.40 0.449 0.455 -0.0406
```

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

Ans: The residuals from the Holts winter dampened method look like only white noise. The three diagnostic plots show that the innovation residuals are centered around 0 without significant spikes. The histogram for the residual plot is normally distributed except for one outlier, which could be due to the sudden srop of data in 2010 Jan. Finally, the ACF also does not have any significant spikes outside of the upper and lower range.

```r
myseries %>%
  model(
    holts_winter_damp = ETS(Turnover ~ error('A') + trend('Ad') + season('M'))
    ) %>%
  gg_tsresiduals()
```



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?

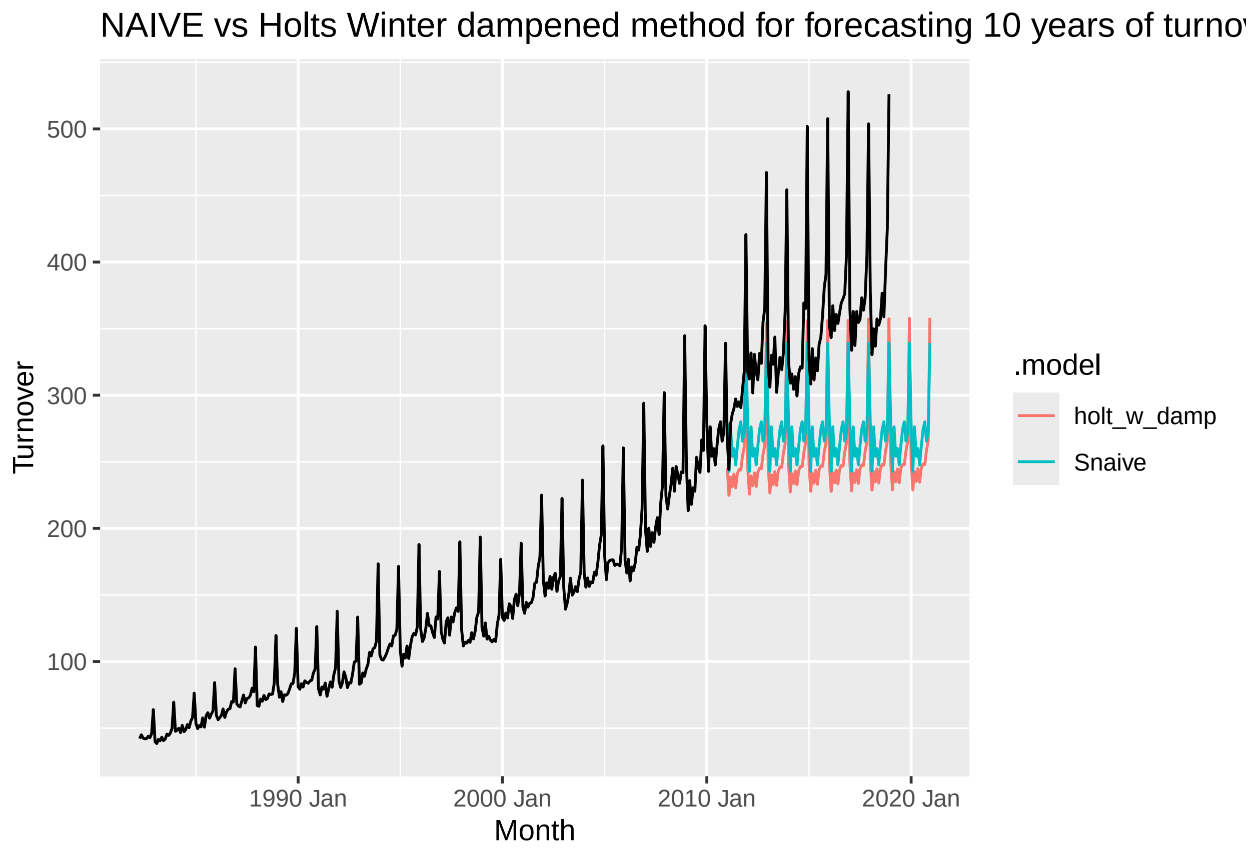
Ans: The Holt's Winter dampened method is performing far better than the SNAIVE method. The RMSE for Holt's Winter dampen method is 6.462 and the SNAIVE is 14.41. The Holt's Winter dampen trumps the SNAIVE.

```r
myseries_train<- myseries %>%
  filter(yearmonth(Month) < yearmonth('2011'))
```

```
## Warning: There was 1 warning in ‘filter()‘.
## i In argument: ‘yearmonth(Month) < yearmonth("2011")‘.
## Caused by warning:
## ! ‘yearmonth()‘ may yield unexpected results.
## i Please use arg ‘format‘ to supply formats.
```

```r
my_fit<- myseries_train %>%
  model(
    holt_w_damp = ETS(Turnover ~ error('A') + trend('Ad') + season('M')),
    Snaive = SNAIVE(Turnover)
  )

my_fc<- my_fit %>%
  forecast(h = '10 year')
```

```r
my_fc %>%
  autoplot(myseries, level = NULL) +
  ggtitle('NAIVE vs Holts Winter dampened method for forecasting 10 years of turnover in retail')
```



To compare the RMSE for both methods:

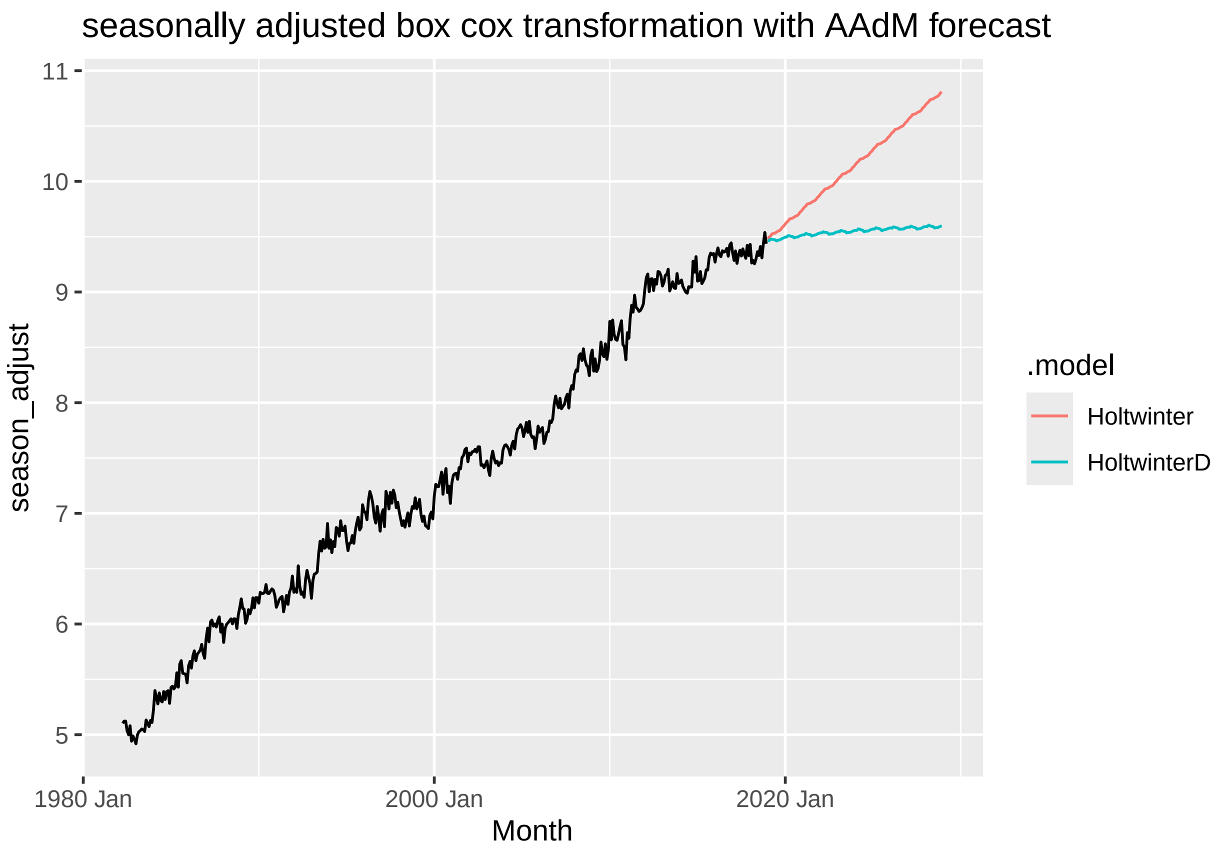
```r
accuracy(my_fit)
```

```
## # A tibble: 2 x 12
##   State  Industry .model .type    ME  RMSE   MAE   MPE  MAPE  MASE RMSSE    ACF1
##   <chr>  <chr>    <chr>  <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>
## 1 Weste~ Other r~ holt_~ Trai~ 0.542  6.46  4.61 0.321  3.53 0.425 0.448 -0.0136
## 2 Weste~ Other r~ Snaive Trai~ 8.18  14.4  10.9  6.05   8.11 1     1      0.725
```

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?

Ans: The RMSE after the box cox transformation seems to be much lower when compared to the un-transformed version, which is 6.462 and the transformed version is 0.0858. Box cox transformation stabilizes the variants and the STL decomposition got rid of the seasonal variation to allow the ETS methods for a better forecast.

```r
#find the best lambda:
lambda<- myseries %>%
  features(Turnover, features = guerrero) %>%
  pull(lambda_guerrero)
#Apply box cos to STL decomposition
bc_myseries<- myseries %>%
  model(
    STL= STL(box_cox(Turnover, lambda) ~ trend() + season(window='periodic'))
  ) %>%
  components() %>%
  select(Month, season_adjust)
#Apply ETS methods to the transformed data
seasonal_series<- bc_myseries %>%
  model(
    HoltwinterD = ETS(season_adjust ~ error('A') + trend('Ad')+season('M')),
    Holtwinter = ETS(season_adjust ~ error('M') + trend('A')+season('A'))
  )
#forecast for 10 years
season_fc<- seasonal_series %>%
  forecast(h = '10 year')

season_fc %>%
  autoplot(bc_myseries, level = NULL)+
  labs(title='seasonally adjusted box cox transformation with AAdM forecast')
```



To compute the RMSE:

```r
accuracy(seasonal_series)
```

```
## # 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 HoltwinterD Training  0.0123  0.0858 0.0691  0.178  0.965 0.431 0.432 0.0828
## 2 Holtwinter  Training -0.00251 0.0841 0.0676 -0.0429 0.941 0.422 0.424 0.0545
```