Exponential Smoothing

Shovan Biswas

2020/10/03

Libraries

```
library(tidyverse)
library(kableExtra)
library(corrplot)
library(reshape2)
library(caret)
library(Amelia)
library(dlookr)
library(fpp2)
library(plotly)
library(gridExtra)
library(readxl)
```

Exercise 7.1

1.Consider the pigs series — the number of pigs slaughtered in Victoria each month.

a.Use the ses() function in R to find the optimal values of α and l_0, and generate forecasts for the next four months.

A cursory glance at pigs dataset.

```
head(pigs)
```

```
##         Jan    Feb    Mar    Apr    May    Jun
## 1980  76378  71947  33873  96428 105084  95741
```

```
summary(pigs)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   33873   79080   91662   90640  101493  120184
```

Now, I'll apply function to ses(), with forecasting periods h = 4. Then I'll display the model, with function summary(), and pick the optimal values of α and l_0.

```r
pigs_ses <- ses(pigs, h = 4)
summary(pigs_ses)
```

```
##
## Forecast method: Simple exponential smoothing
##
## Model Information:
## Simple exponential smoothing
##
## Call:
##  ses(y = pigs, h = 4)
##
##   Smoothing parameters:
##     alpha = 0.2971
##
##   Initial states:
##     l = 77260.0561
##
##   sigma:  10308.58
##
##      AIC     AICc      BIC
## 4462.955 4463.086 4472.665
##
## Error measures:
##                    ME    RMSE      MAE       MPE     MAPE      MASE       ACF1
## Training set 385.8721 10253.6 7961.383 -0.922652 9.274016 0.7966249 0.01282239
##
## Forecasts:
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Sep 1995       98816.41 85605.43 112027.4 78611.97 119020.8
## Oct 1995       98816.41 85034.52 112598.3 77738.83 119894.0
## Nov 1995       98816.41 84486.34 113146.5 76900.46 120732.4
## Dec 1995       98816.41 83958.37 113674.4 76092.99 121539.8
```

So, the optimal values of $\alpha = 0.2971$ and $l_0 = 77260.0561$.

b.Compute a 95% prediction interval for the first forecast using $\hat{y} \pm 1.96s$ where s is the standard deviation of the residuals. Compare your interval with the interval produced by R.

```r
s <- sd(pigs_ses$residuals)                                                    # Standard Deviation, usi
m <- pigs_ses$mean[1]                                                          # Storing the mean, using
#
lb <- round(m - 1.96 * s, 2)
ub <- round(m + 1.96 * s, 2)
```

Computing the bound of the intervals, below.

```r
print(paste0('Lower bound CI = ', lb))
```

```
## [1] "Lower bound CI = 78679.97"
```

```r
print(paste0('Upper bound CI = ', ub))
```

```
## [1] "Upper bound CI = 118952.84"
```

R provides functions to directly compute the intervals.

```r
rlb <- round(ses(pigs, h = 4, level = 95)$lower[1], 2)
rub <- round(ses(pigs, h = 4, level = 95)$upper[1], 2)
```

```r
print(paste0('R computed lower bound = ', rlb))
```

```
## [1] "R computed lower bound = 78611.97"
```

```r
print(paste0('R computed lower bound = ', rub))
```

```
## [1] "R computed lower bound = 119020.84"
```

Now, I'll inspect the differences are between manual and R computed upper and lower bounds.

```r
print(paste0('Manually computed difference = ', ub - lb))
```

```
## [1] "Manually computed difference = 40272.87"
```

```r
print(paste0('R computed difference = ', rub - rlb))
```

```
## [1] "R computed difference = 40408.87"
```

So, I observe that the R computed interval is wider than the manually computed one.

Exercise 7.5

5.Data set books contains the daily sales of paperback and hardcover books at the same store. The task is to forecast the next four days' sales for paperback and hardcover books.

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

A cursory glance at books dataset.

```r
head(books)
```

```
## Time Series:
## Start = 1
## End = 6
## Frequency = 1
##   Paperback Hardcover
## 1       199       139
## 2       172       128
## 3       111       172
## 4       209       139
## 5       161       191
## 6       119       168
```

```r
summary(books)
```

```
##    Paperback       Hardcover    
##  Min.   :111.0   Min.   :128.0  
##  1st Qu.:167.2   1st Qu.:170.5  
##  Median :189.0   Median :200.5  
##  Mean   :186.4   Mean   :198.8  
##  3rd Qu.:207.2   3rd Qu.:222.0  
##  Max.   :247.0   Max.   :283.0
```

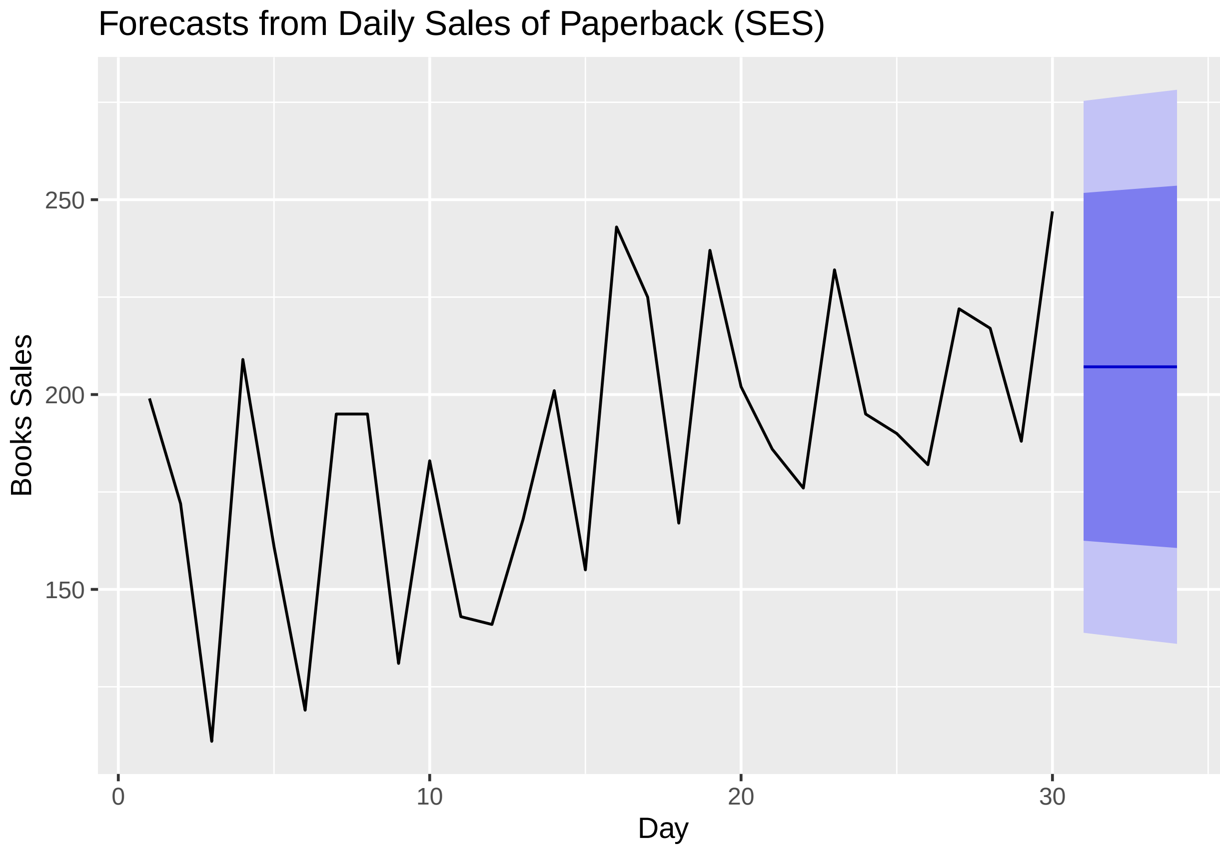
The plot of daily book sales.

```r
autoplot(books) + ggtitle("Daily Book Sales") + xlab("Day") + ylab("Books Sales")
```

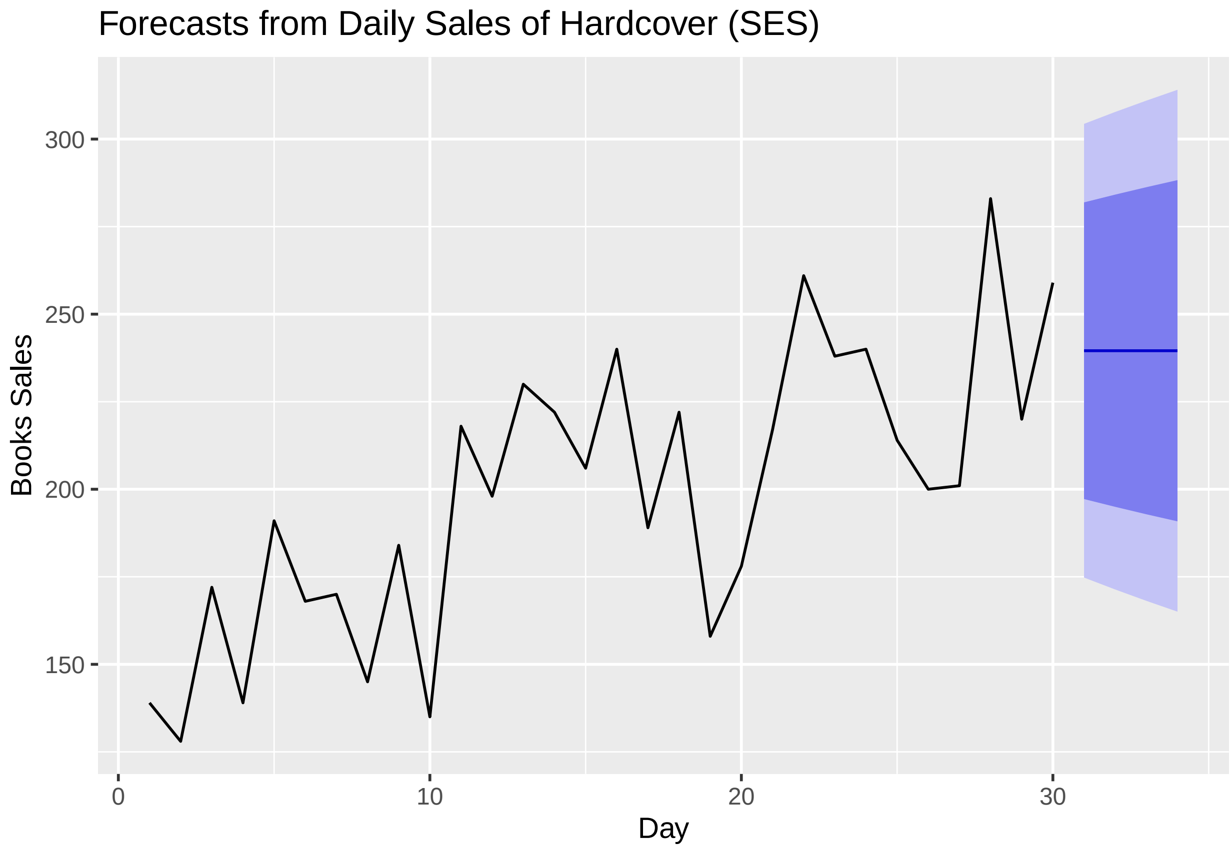
Observations:
- Both paperback and hardcover are upward trending.
- Seasonality is not visible in the 30 days of data.
- In the first 10 days, the paperback books sales are higher, but from 10th to the end of the month, the hardcover books overtakes paperbacks.

b.Use the ses() function to forecast each series, and plot the forecasts.

```r
autoplot(ses(books[, "Paperback"], h = 4)) + ggtitle("Forecasts from Daily Sales of Paperback (SES)") +
```

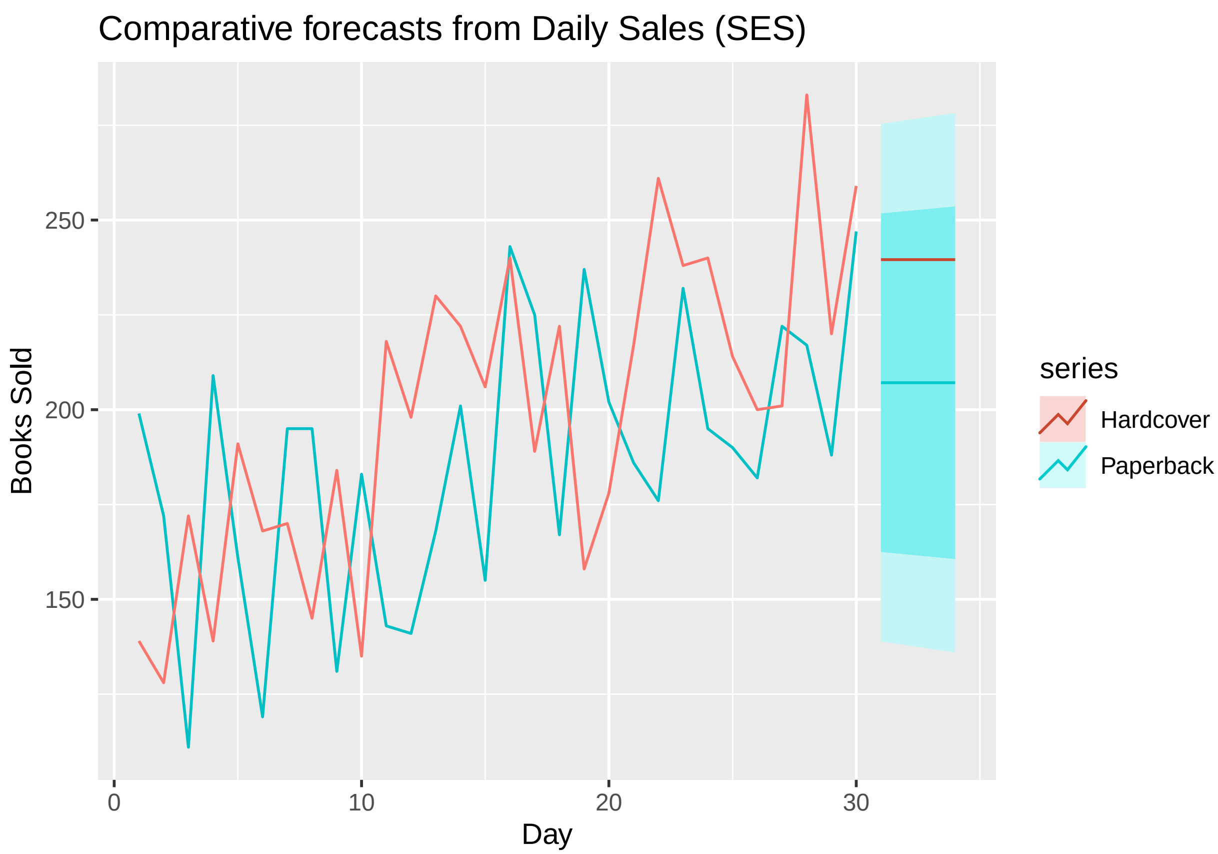



```
autoplot(ses(books[, "Hardcover"], h = 4)) + ggtitle("Forecasts from Daily Sales of Hardcover (SES)") +
```



For a comparative view.

```
autoplot(books[, "Paperback"], series = "Paperback") + autolayer(ses(books[, "Paperback"], h = 4), serie
autolayer(books[, "Hardcover"], series = "Hardcover") + autolayer(ses(books[, "Hardcover"], h = 4), seri
  ggtitle("Comparative forecasts from Daily Sales (SES)") + xlab("Day") + ylab("Books Sold")
```



One important observation at this point is, although daily sales of both paperback and hardcover books are trending upward, the forcast doesn't catch the trend.

c.Compute the RMSE values for the training data in each case.

```
round(accuracy(ses(books[, "Paperback"], h = 4)), 2)  # RMSE for paperback
```

```
##                ME  RMSE   MAE  MPE  MAPE MASE  ACF1
## Training set 7.18 33.64 27.84 0.47 15.58  0.7 -0.21
```

```
round(accuracy(ses(books[, "Hardcover"], h = 4)), 2)  # RMSE for hardcover
```

```
##                ME  RMSE   MAE  MPE  MAPE MASE  ACF1
## Training set 9.17 31.93 26.77 2.64 13.39  0.8 -0.14
```

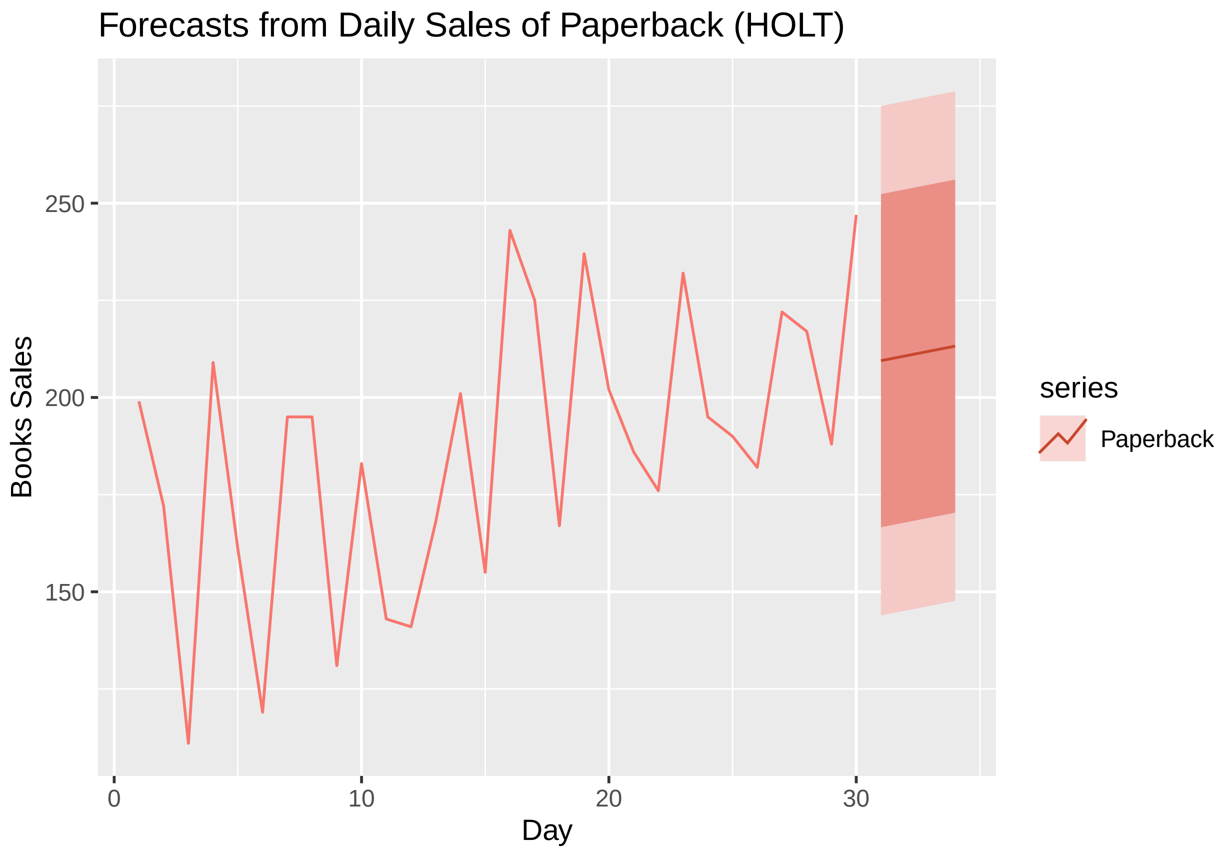
Exercise 7.6

Note: I have a paper and PDF copies of the book Forecasting Principles, but online eidtion differs in some places. One difference is in the wording of question 7.6. I followed online edition, which is often used during meetups.

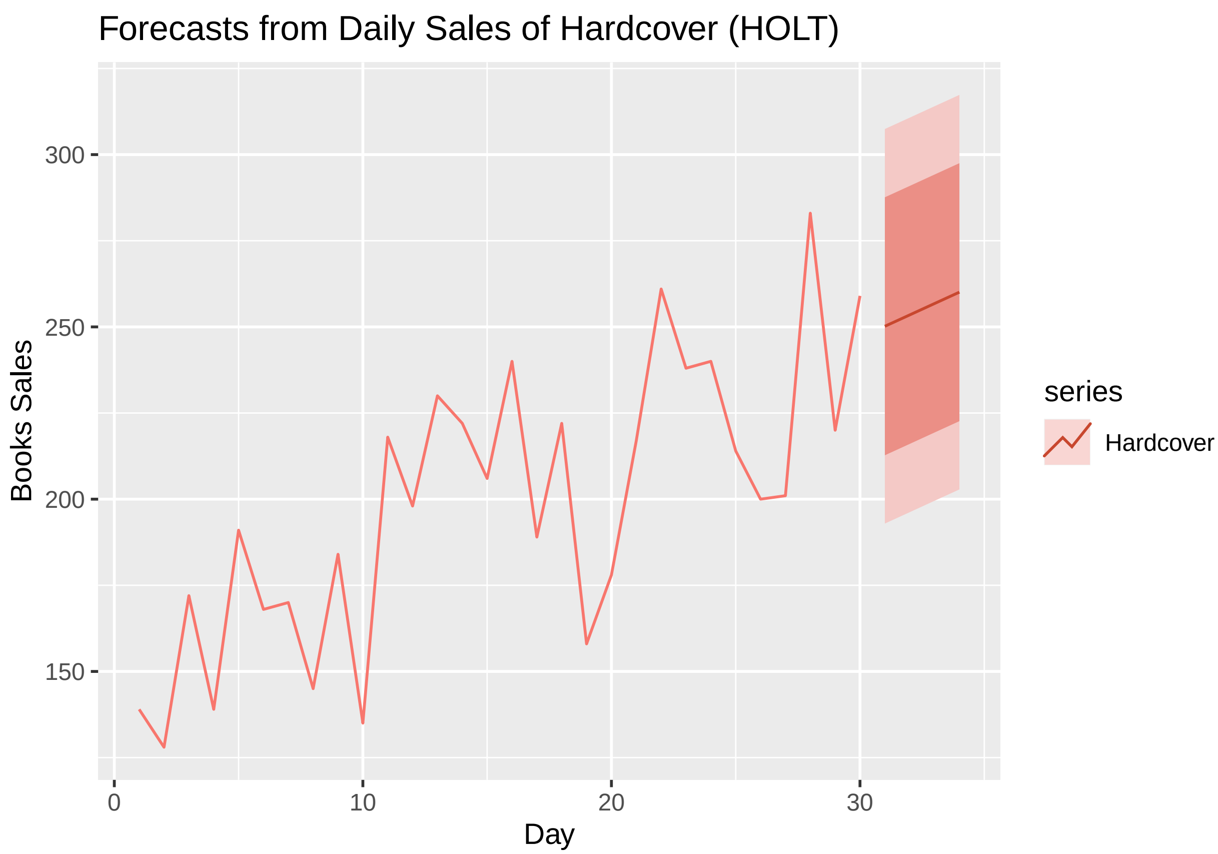
6.We will continue with the daily sales of paperback and hardcover books in data set books.

a.Now apply Holt's linear method to the paperback and hardback series and compute four-day forecasts in each case.

```r
# autoplot(holt(books[, "Paperback"], h = 4)) + ggtitle("Forecasts from Daily Sales of Paperback (HOLT)
autoplot(books[, "Paperback"], series = "Paperback") + autolayer(holt(books[, "Paperback"], h = 4), ser
  ggtitle("Forecasts from Daily Sales of Paperback (HOLT)") + xlab("Day") + ylab("Books Sales")
```

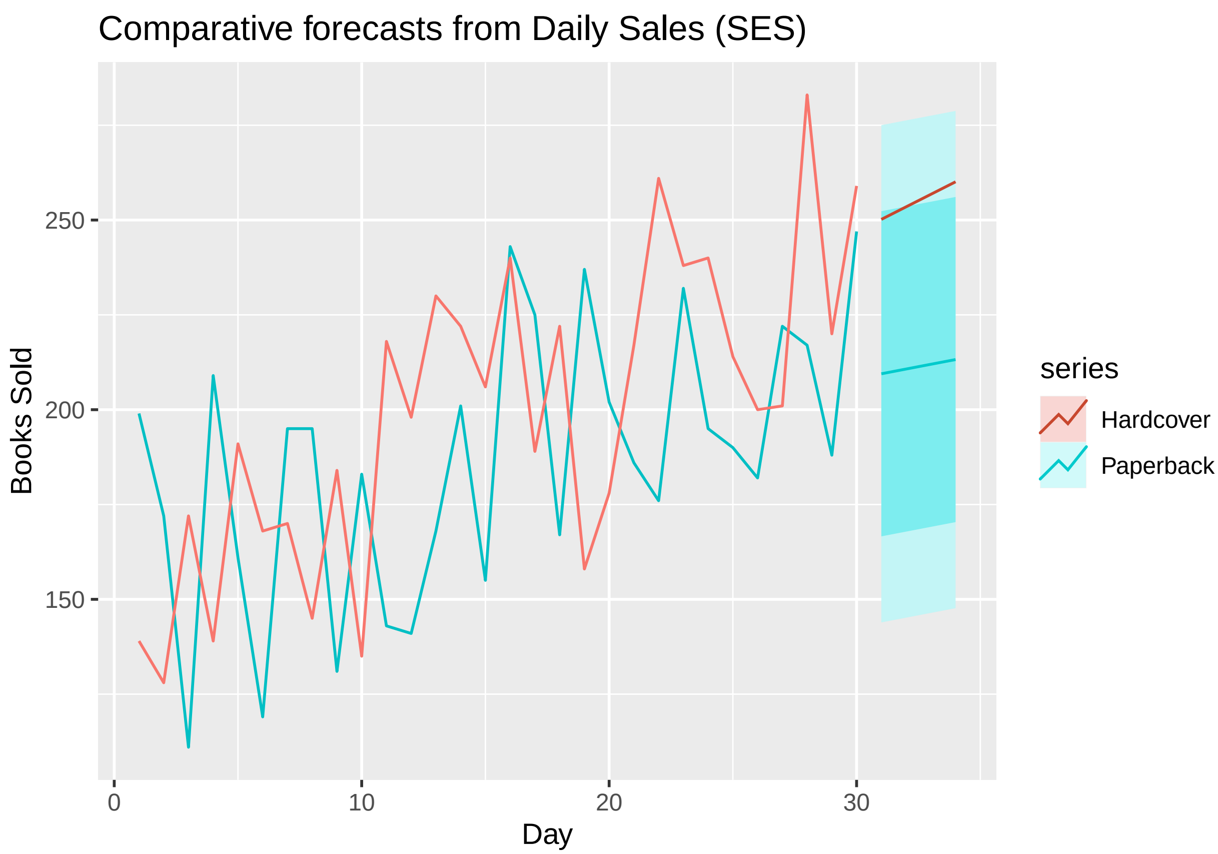



```r
#autoplot(holt(books[, "Hardcover"], h = 4)) + ggtitle("Forecasts from Daily Sales of Hardcover (HOLT)"
autoplot(books[, "Hardcover"], series = "Hardcover") + autolayer(holt(books[, "Hardcover"], h = 4), ser
  ggtitle("Forecasts from Daily Sales of Hardcover (HOLT)") + xlab("Day") + ylab("Books Sales")
```



For a comparative view of Holt's method of forecast.

```
autoplot(books[, "Paperback"], series = "Paperback") + autolayer(holt(books[, "Paperback"], h = 4), seri
autolayer(books[, "Hardcover"], series = "Hardcover") + autolayer(holt(books[, "Hardcover"], h = 4), se
  ggtitle("Comparative forecasts from Daily Sales (SES)") + xlab("Day") + ylab("Books Sold")
```



In Holt's method, we observe that forecast caught the trend, slightly. In sales of paperbacks, the trend is not so perceptble, but in sales of hardcover the mild upward gradinent is perceptible.

b.Compare the RMSE measures of Holt's method for the two series to those of simple exponential smoothing in the previous question. (Remember that Holt's method is using one more parameter than SES.) Discuss the merits of the two forecasting methods for these data sets.

```
round(accuracy(holt(books[, "Paperback"], h = 4)), 2)  # Holt's method for RMSE for paperback
```

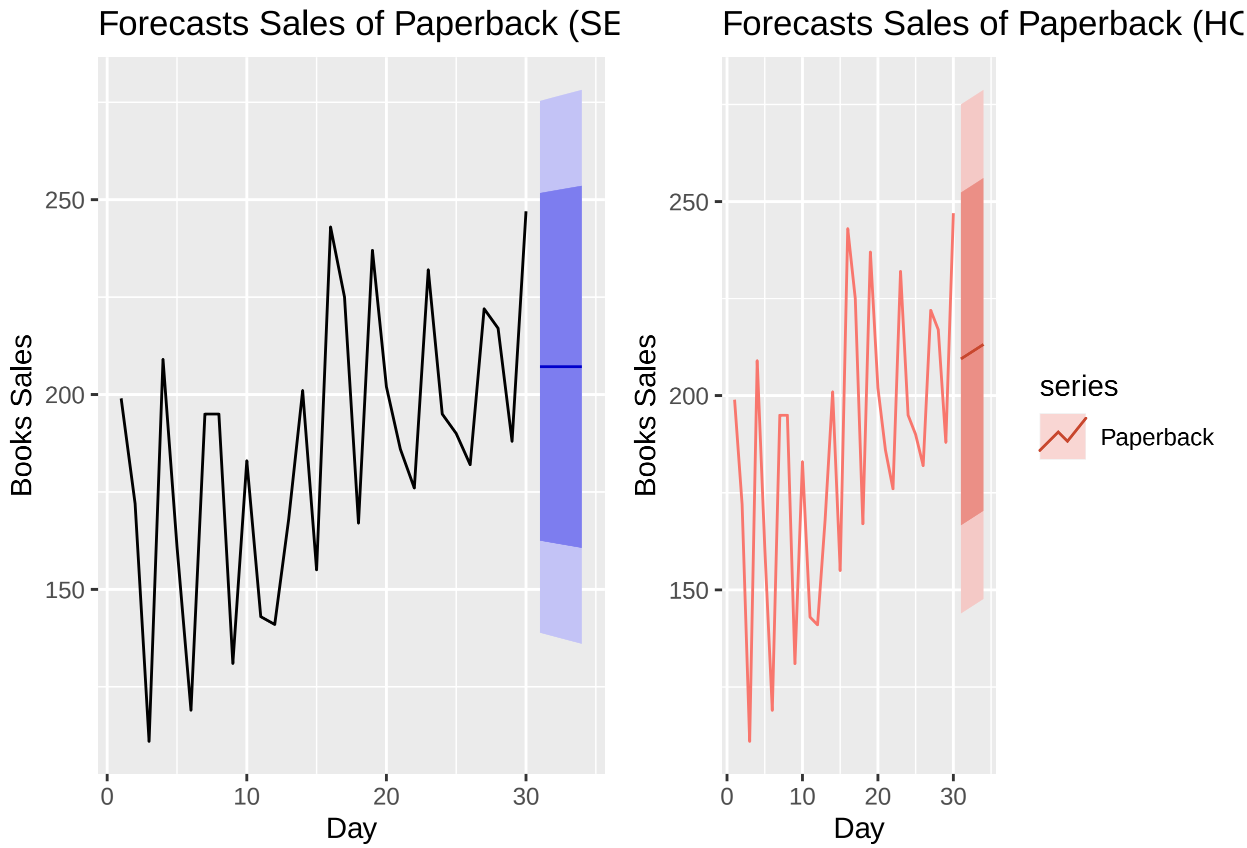
```
##                 ME  RMSE   MAE   MPE  MAPE MASE  ACF1
## Training set -3.72 31.14 26.18 -5.51 15.58 0.66 -0.18
```

```
round(accuracy(holt(books[, "Hardcover"], h = 4)), 2)  # Holt's method for RMSE for hardcover
```

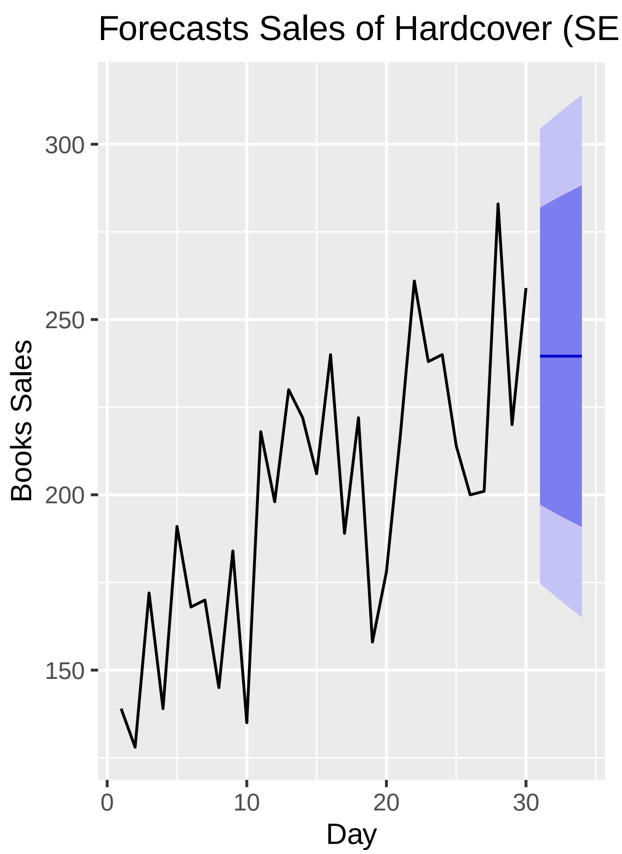
```
##                 ME  RMSE   MAE   MPE  MAPE MASE  ACF1
## Training set -0.14 27.19 23.16 -2.11 12.16 0.69 -0.03
```

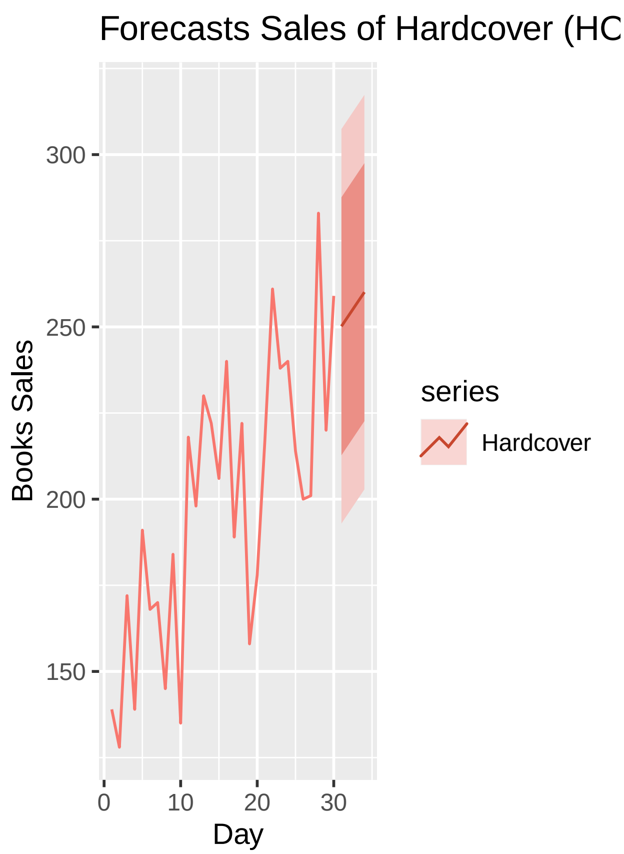
c.Compare the forecasts for the two series using both methods. Which do you think is best?

```
grid.arrange(autoplot(ses(books[, "Paperback"], h = 4)) + ggtitle("Forecasts Sales of Paperback (SES)")
  ggtitle("Forecasts Sales of Paperback (HOLT)") + xlab("Day") + ylab("Books Sales"), ncol = 2)
```




```r
grid.arrange(autoplot(ses(books[, "Hardcover"], h = 4)) + ggtitle("Forecasts Sales of Hardcover (SES)")
  ggtitle("Forecasts Sales of Hardcover (HOLT)") + xlab("Day") + ylab("Books Sales"), ncol = 2)
```





In Holt's method, we observe that forecast caught the trend, slightly. In sales of paperbacks, the trend is not so perceptble, but in sales of hardcover it's perceptible.

So, I would think that Holt's method is an improvement over SES method.

d.Calculate a 95% prediction interval for the first forecast for each series, using the RMSE values and assuming normal errors. Compare your intervals with those produced using ses and holt.

Prediction intervals for paperback

```
RMSE_indx = 2    # RME is in the second index position. SO, I am initializing it with 2. It'll be used
s <- round(accuracy(holt(books[, "Paperback"], h = 4)), 2)[RMSE_indx]
print(paste0('Prediction interval using RMSE: ', "(", holt(books[, "Paperback"], h = 4)$mean[1] - 1.96
```

```
## [1] "Prediction interval using RMSE: (148.432365521051, 270.501165521051)"
```

SES method.

```
print(paste0("SES method of interval: ", "(", ses(books[, "Paperback"], level = 95, h = 4)$lower[1], ",
```

```
## [1] "SES method of interval: (138.867024106993, 275.352305883445)"
```

Holt's method.

```r
print(paste0("SES method of interval: ", "(", holt(books[, "Paperback"], level = 95, h = 4)$lower[1], "
```

```
## [1] "SES method of interval: (143.912985820256, 275.020545221845)"
```

Prediction intervals for hardcover

```r
RMSE_indx = 2    # RME is in the second index position. SO, I am initializing it with 2. It'll be used
s <- round(accuracy(holt(books[, "Hardcover"], h = 4)), 2)[RMSE_indx]
print(paste0('Prediction interval using RMSE: ', "(", holt(books[, "Hardcover"], h = 4)$mean[1] - 1.96
```

```
## [1] "Prediction interval using RMSE: (196.881472212356, 303.466272212356)"
```

SES method.

```r
print(paste0("SES method of interval: ", "(", ses(books[, "Hardcover"], level = 95, h = 4)$lower[1], ",
```

```
## [1] "SES method of interval: (174.779870851487, 304.340313152036)"
```

Holt's method.

```r
print(paste0("SES method of interval: ", "(", holt(books[, "Hardcover"], level = 95, h = 4)$lower[1], "
```

```
## [1] "SES method of interval: (192.922170771941, 307.42557365277)"
```

Exercise 7.7

7.For this exercise use data set eggs, the price of a dozen eggs in the United States from 1900–1993. Experiment with the various options in the holt() 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 argument is doing to the forecasts.

[Hint: use h=100 when calling holt() so you can clearly see the differences between the various options when plotting the forecasts.]

Which model gives the best RMSE?

A cursory glance at eggs dataset.

```r
head(eggs)
```

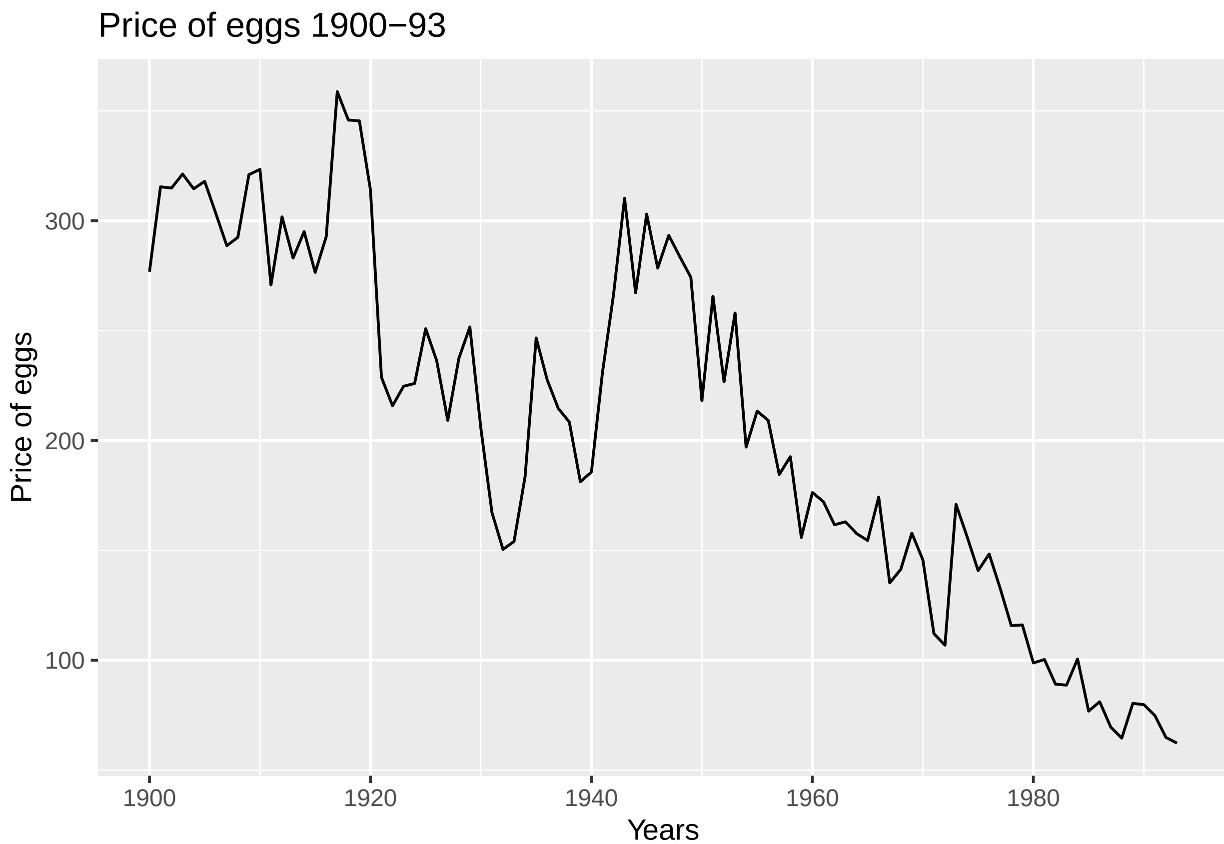
```
## Time Series:
## Start = 1900
## End = 1905
## Frequency = 1
## [1] 276.79 315.42 314.87 321.25 314.54 317.92
```

```r
summary(eggs)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   62.27  148.87  209.15  206.15  276.71  358.78
```

In order to get a rough idea of the price of eggs from 1900-1993, first, let me look at the autpplot.

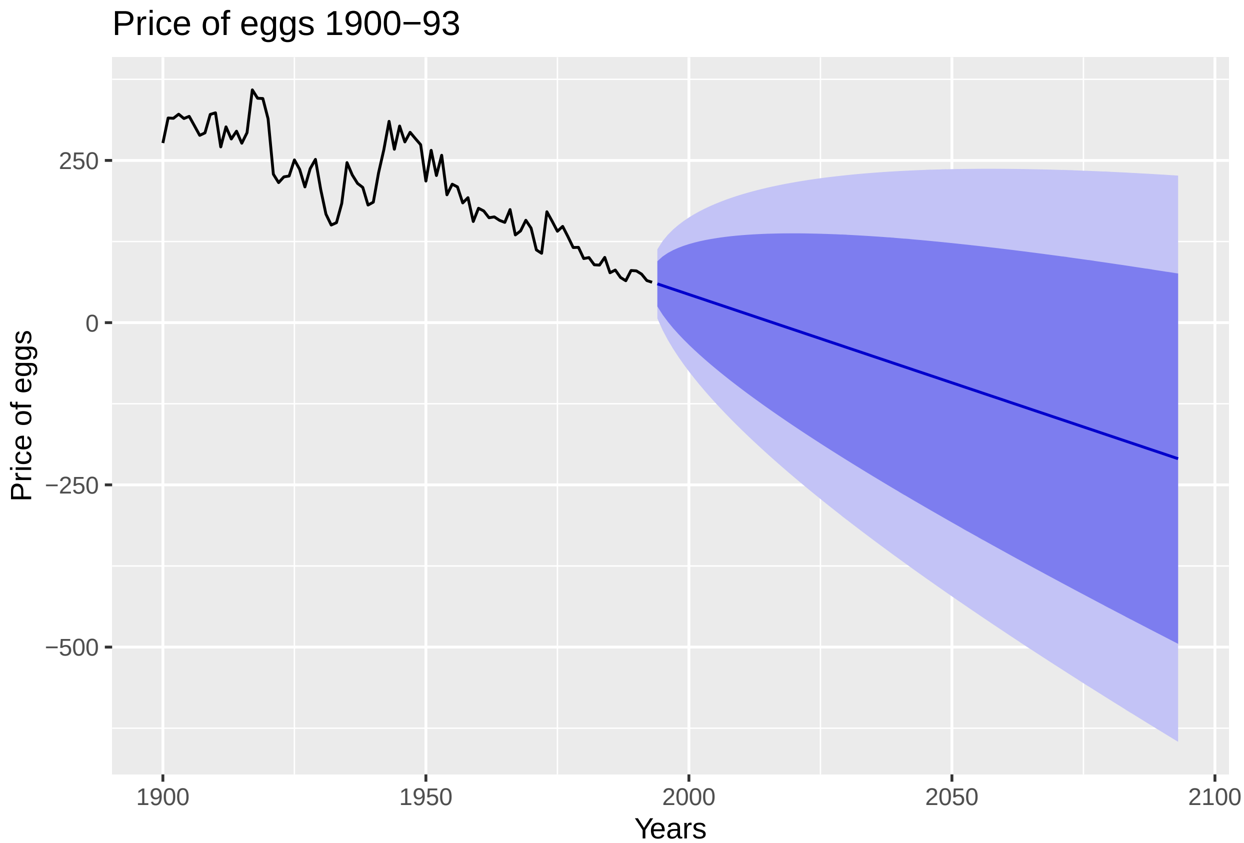
```r
autoplot(eggs) + ggtitle("Price of eggs 1900-93") + xlab("Years") + ylab("Price of eggs")
```



The price of eggs trended downwards in all those years.

Now, let me use Holt's forecasting.

```r
autoplot(eggs) + autolayer(holt(eggs, h = 100)) + ggtitle("Price of eggs 1900-93") + xlab("Years") + yla
```



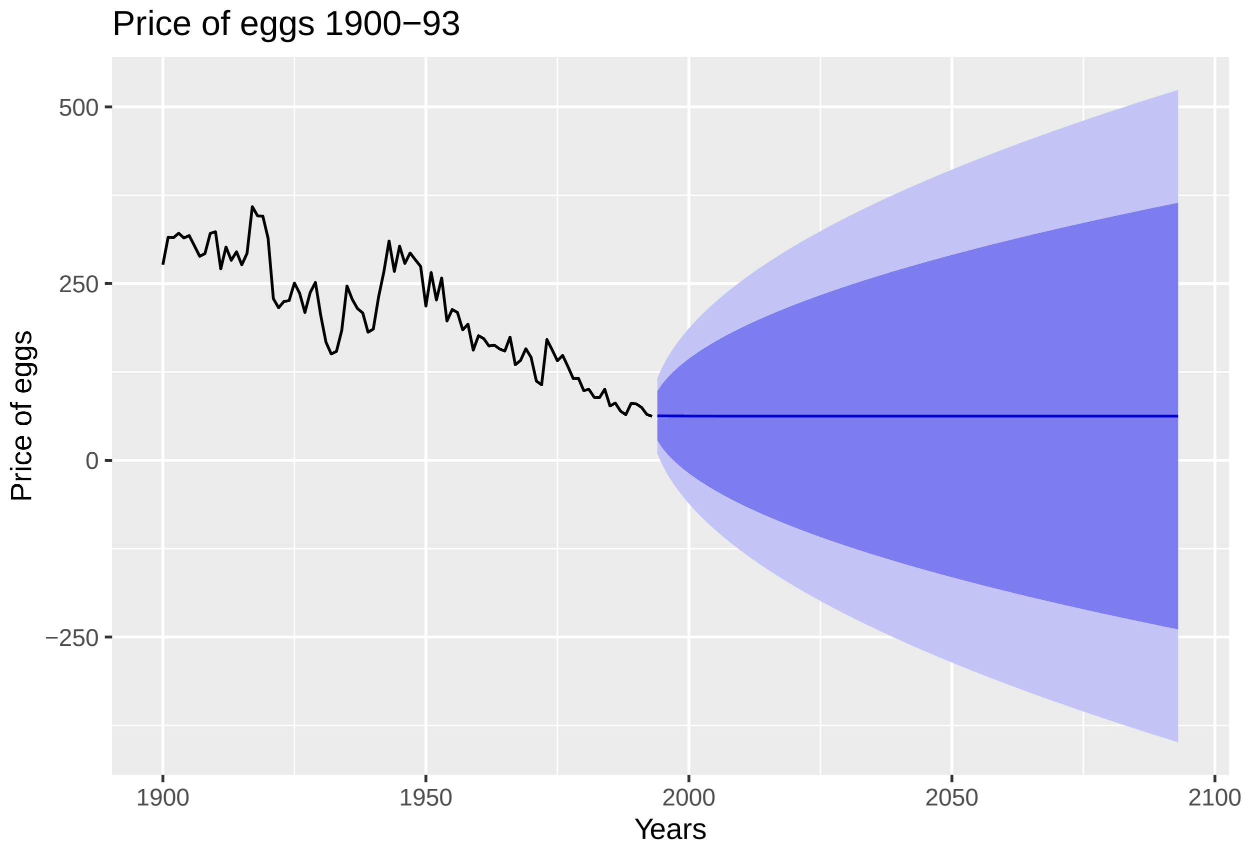
Observation: While Holt's has forecast correcty caught the downward trend, it's an overkill, because price drops to zero and then eventually becomes negative, which is absurd.

We learn from *Forecasting Principles*, "The forecasts generated by Holt's linear method display a constant trend (increasing or decreasing) indefinitely into the future. Empirical evidence indicates that these methods tend to over-forecast, especially for longer forecast horizons".

This might be an example of over-forecast.

In order to alleviate this problem, Gardner & McKenzie introduced a parameter that dampens the trend to a flat line. We'll observe this below.

```
autoplot(eggs) + autolayer(holt(eggs, h = 100, damped = T)) + ggtitle("Price of eggs 1900-93") + xlab("Y
```

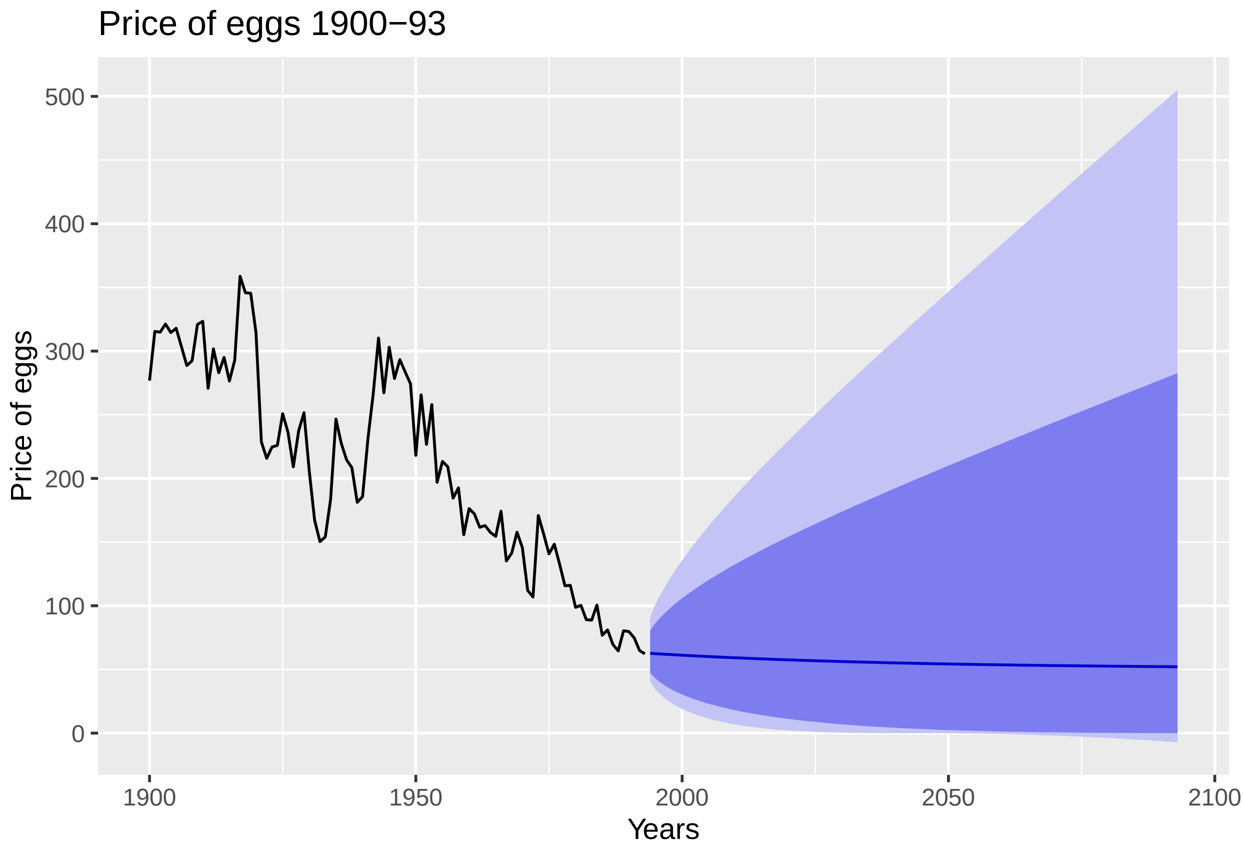


Clearly, the forecast flattened out and removed the absurdity of negative price.

In the context of various methods of Holt, one may suppose that Holt-Winter's method is relevant. But it won't be applicable to eggs dataset, because the data is not seasonal. So, I am not trying Holt-Winter's method.

Now, I'll try with Box-Cox transformation.

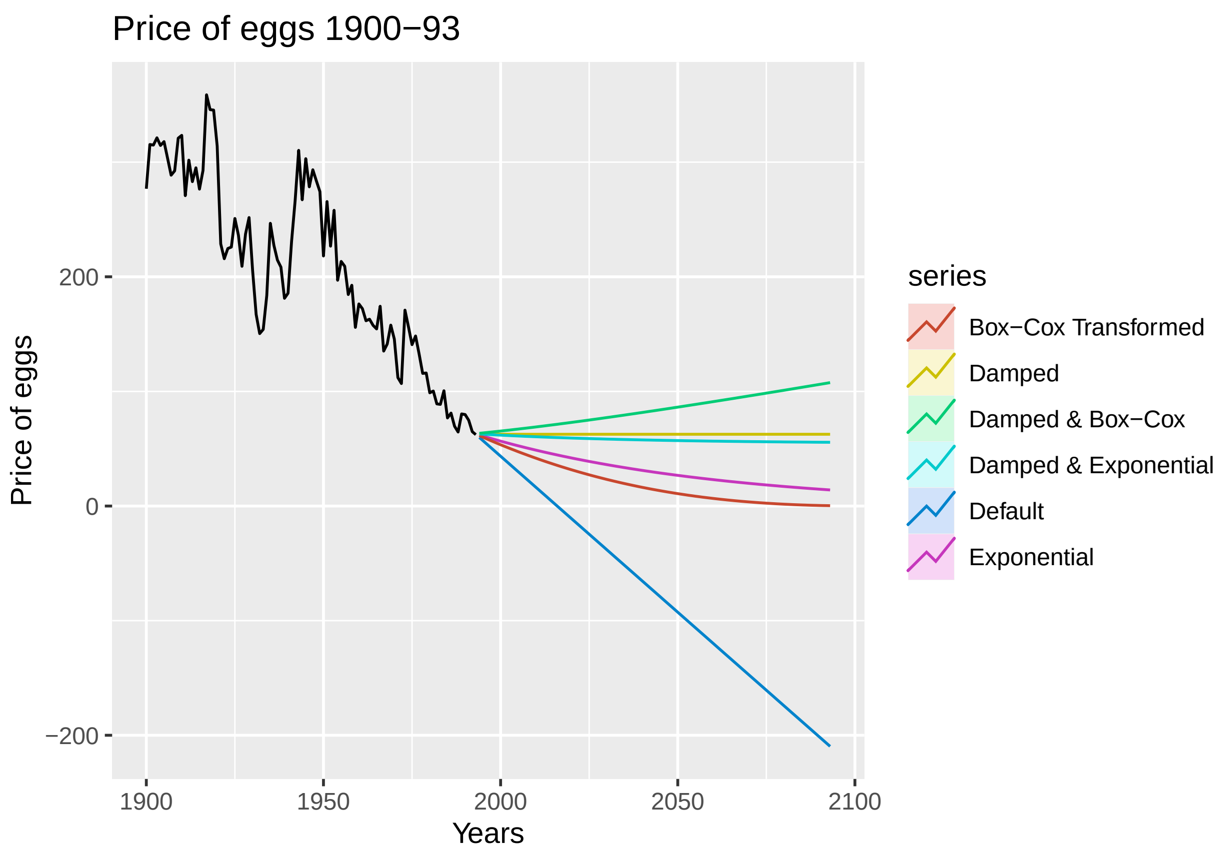
```
autoplot(eggs) + autolayer(holt(eggs, lambda = BoxCox.lambda(eggs), h = 100, damped = T)) + ggtitle("Pri
```



Observe that due to Box-Cox transformation, no part of the inflated blue region is below zero. Could be because of logarithm.

Now, in order to get a comparative view, I'll have them in one graphe.

```
autoplot(eggs) +
  autolayer(holt(eggs, h = 100), series = 'Default', PI = F) +
  autolayer(holt(eggs, h = 100, damped = T), series = 'Damped', PI = F) +
  autolayer(holt(eggs, h = 100, exponential = T), series = 'Exponential', PI = F) +
  autolayer(holt(eggs, lambda = BoxCox.lambda(eggs), h = 100), series = 'Box-Cox Transformed', PI = F) +
  autolayer(holt(eggs, h = 100, exponential = T, damped = T), series = 'Damped & Exponential', PI = F) +
  autolayer(holt(eggs, h = 100, damped = T, lambda = BoxCox.lambda(eggs), biasadj = T), series = 'Dampe
  ggtitle("Price of eggs 1900-93") + xlab("Years") + ylab("Price of eggs")
```



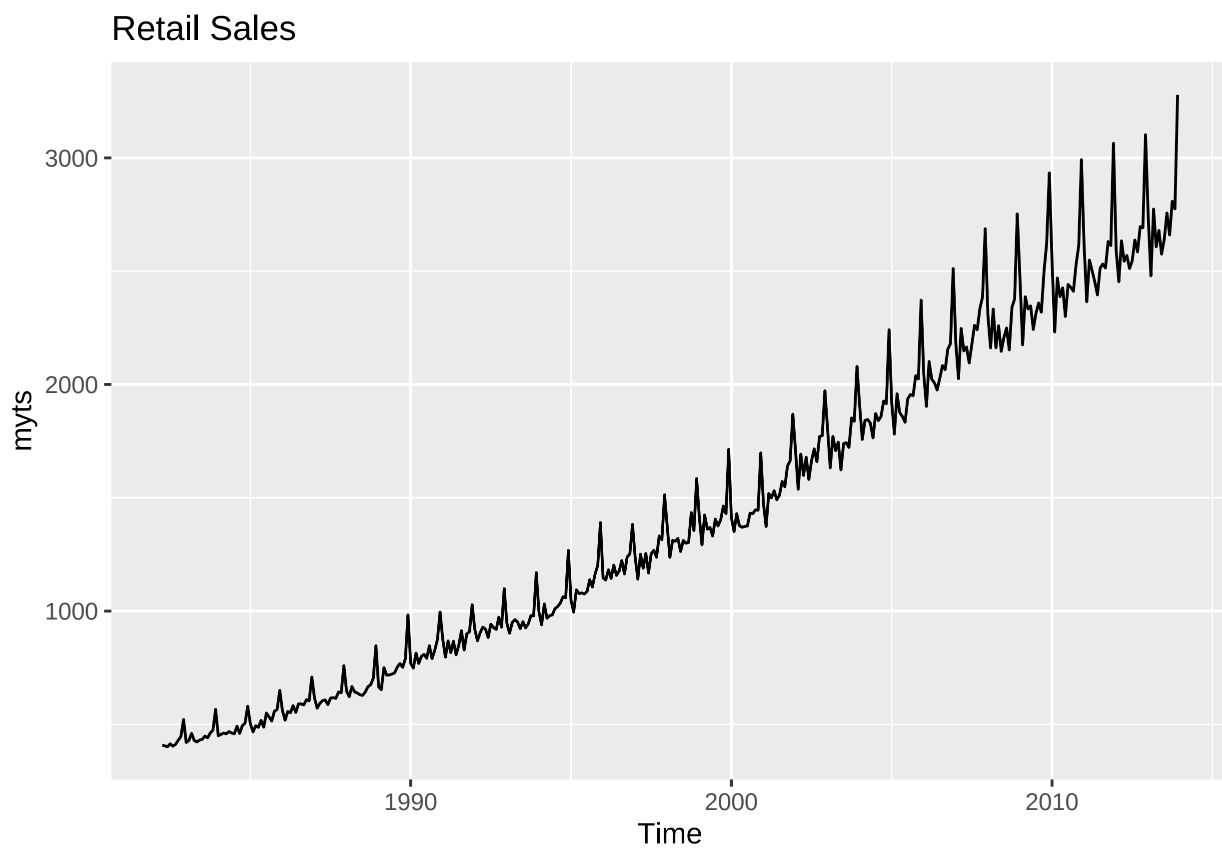
We observed that by Holt's (method without option), the price forecast can become negative. Damped alleviates the problem by flattening the line. Box-Cox and Exponential seem to run asymptotically along the x-axis.

8.Recall your retail time series data (from Exercise 3 in Section 2.10).
a.Why is multiplicative seasonality necessary for this series?

My code from from Exercise 3 in Section 2.10 are in the following code-chunk.

```
retail_data <- read_excel("retail.xlsx", skip = 1)
myts <- ts(retail_data[, "A3349398A"], frequency = 12, start = c(1982, 4))

autoplot(myts) + ggtitle("Retail Sales")
```



The plot clearly shows that the variation increases with time So, multiplicative seaconality is necessary.

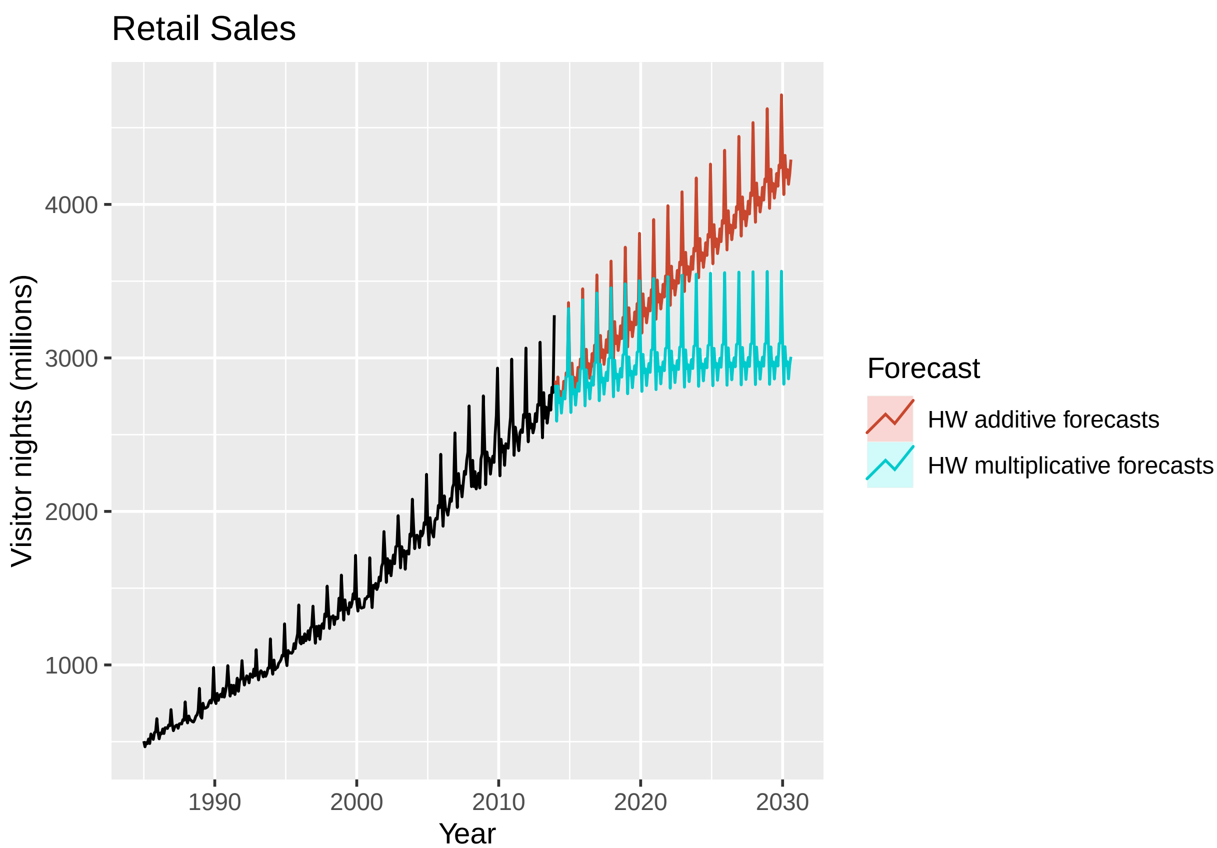
b.Apply Holt-Winters' multiplicative method to the data. Experiment with making the trend damped.

Note: In the following, I used the code from Holt-Winter's section in the textbook and tweaked it to dampen the curve.

```
retail_myts <- window(myts, start = 1985)

fit1 <- hw(retail_myts, seasonal = "additive", h = 200)
fit2 <- hw(retail_myts, seasonal = "multiplicative", damped = TRUE, h = 200)  # Added damped = TRUE

autoplot(retail_myts) +
autolayer(fit1, series = "HW additive forecasts", PI = FALSE) +
autolayer(fit2, series = "HW multiplicative forecasts", PI = FALSE) + xlab("Year") + ylab("Visitor nigh
ggtitle("Retail Sales") + guides(colour = guide_legend(title = "Forecast"))
```



Initially, I tried with damped = TRUE, but the dampening was imperceptible. So, I threw in parameter h = 200, to make the dampening conspicuous.

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

```
print(paste0("Undamped RMSE: ", accuracy(hw(myts, seasonal = "multiplicative", h = 1))[2]))      # This
```

```
## [1] "Undamped RMSE: 29.4305130687359"
```

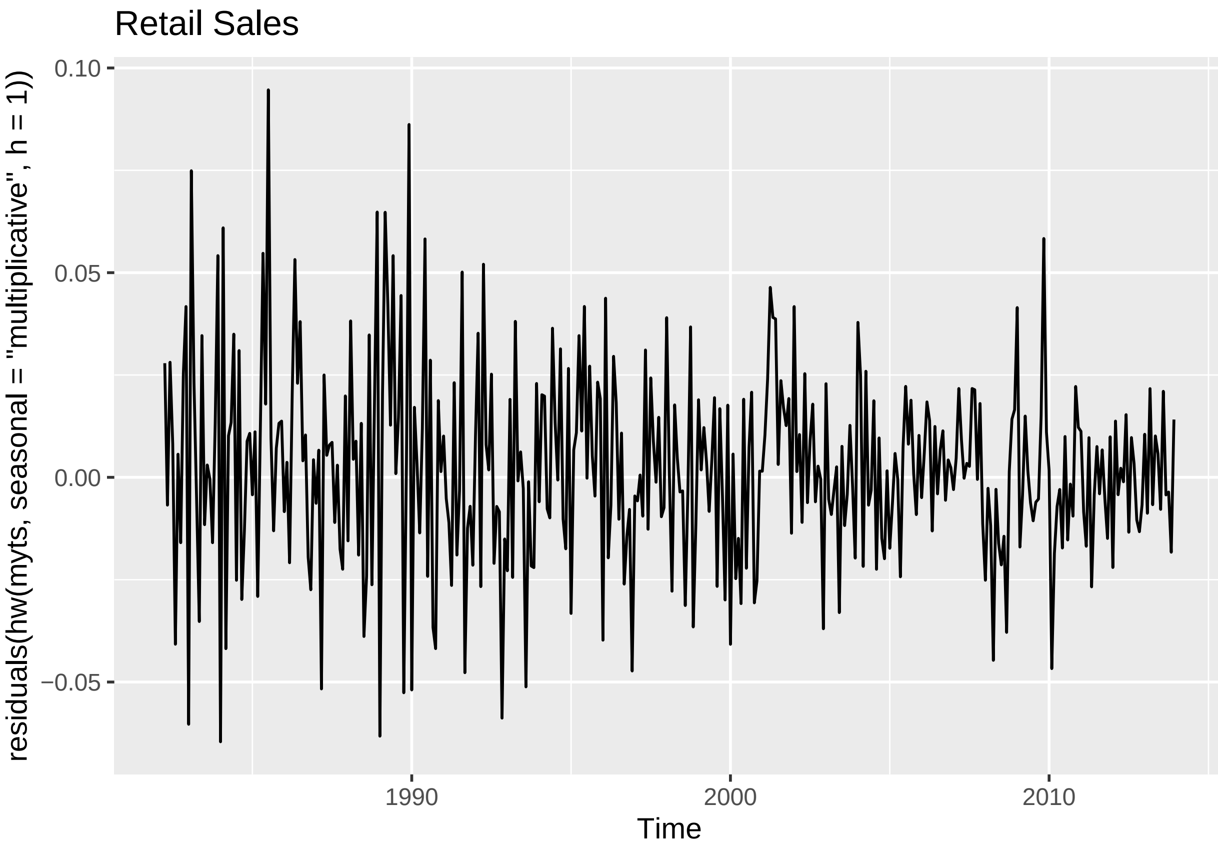
```
print(paste0("Damped RMSE: ", accuracy(hw(myts, damped = TRUE, seasonal = "multiplicative", h = 1))[2]))
```

```
## [1] "Damped RMSE: 29.6308699428669"
```

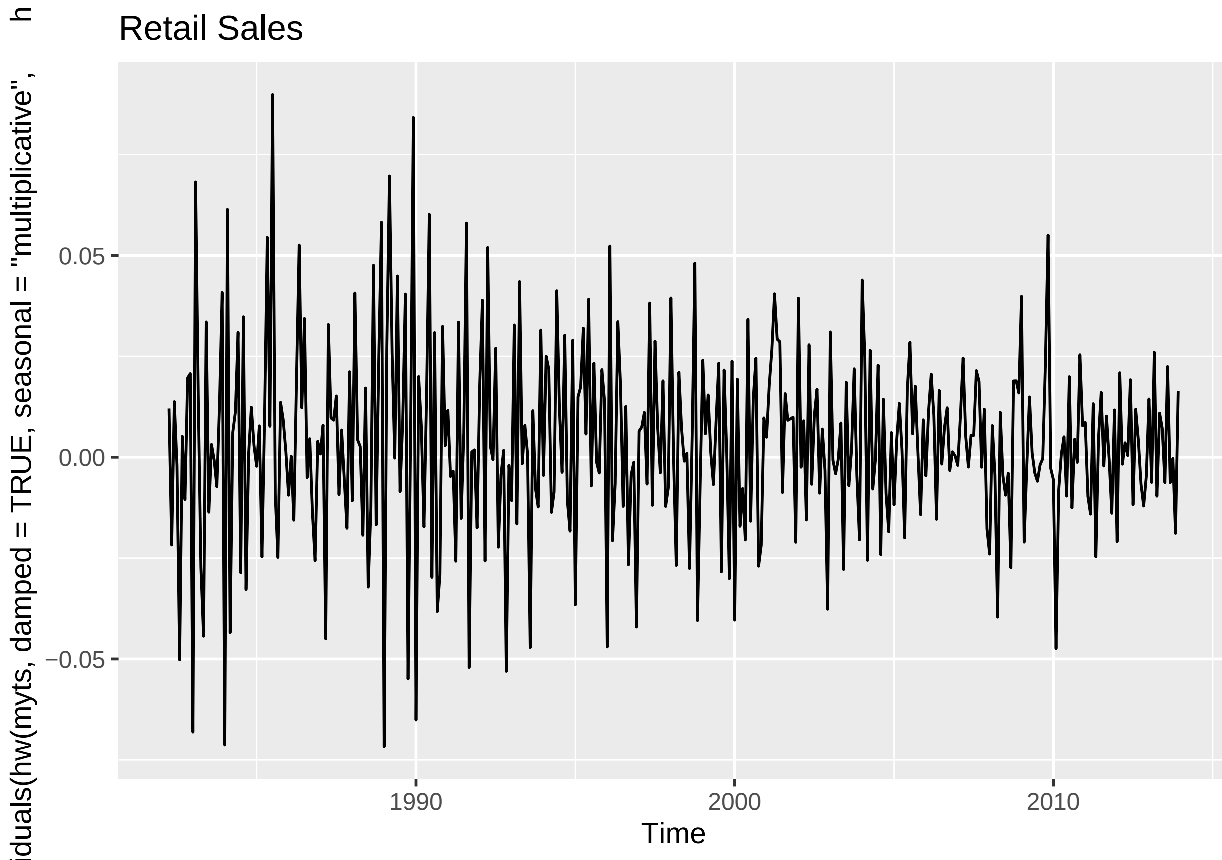
We observe that Holt-Winter's undamped RMSE is slightly lower that of the damped, and therefore it's a better fit.

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

```
autoplot(residuals(hw(myts, seasonal = "multiplicative", h = 1))) + ggtitle("Retail Sales")
```



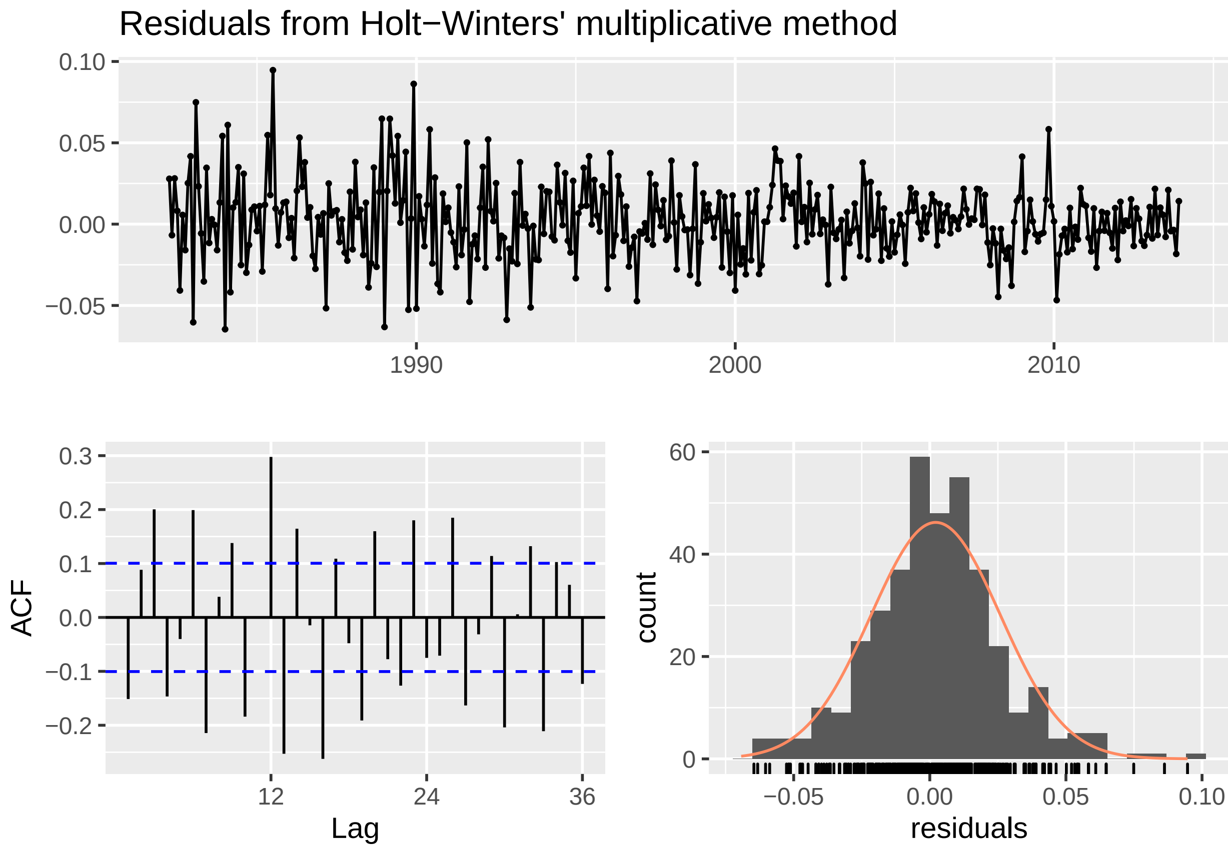

```
autoplot(residuals(hw(myts, damped = TRUE, seasonal = "multiplicative", h = 1))) + ggtitle("Retail Sale
```



Residulas from both undamped and damped Holt-Winter's methods look like whie noise.

The function checkresiduals() gives a better picture. So, I tried with the undamped method below.

```
checkresiduals(hw(myts, seasonal = "multiplicative", h = 1))
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method
## Q* = 244.86, df = 8, p-value < 2.2e-16
##
## Model df: 16.   Total lags used: 24
```

Although the distribution is normal, the graph of residuals is like white noise.

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 8 in Section 3.7?

Here's my code from section 3, Exercise 8, all lumped in one code-chunk.

```
myts.train <- window(myts, end = c(2010, 12))
myts.test <- window(myts, start = 2011)
#
fc <- snaive(myts.train)
#
accuracy(fc, myts.test)
```

```
##                      ME      RMSE       MAE      MPE     MAPE     MASE      ACF1
## Training set  73.94114  88.31208  75.13514 6.068915 6.134838 1.000000 0.6312891
## Test set     115.00000 127.92727 115.00000 4.459712 4.459712 1.530576 0.2653013
##              Theil's U
```

```
## Training set        NA
## Test set     0.7267171
```

Now, I'll try once with damped and once with undamped multiplicative method of Holt-Winter.

Damped.

```
accuracy(hw(myts.train, damped = TRUE, seasonal = "multiplicative", h = 1), myts.test)
```

```
##                      ME     RMSE      MAE       MPE     MAPE      MASE
## Training set  3.777291 29.21058 21.82543 0.2857987 1.875953 0.2904823
## Test set     26.176874 26.17687 26.17687 1.0017555 1.001756 0.3483972
##                     ACF1
## Training set -0.05362944
## Test set              NA
```

undamped.

```
accuracy(hw(myts.train, seasonal = "multiplicative", h = 1), myts.test)
```

```
##                      ME     RMSE      MAE       MPE     MAPE      MASE
## Training set  2.594132 29.87520 22.43516 0.2383033 1.864671 0.2985975
## Test set     31.399564 31.39956 31.39956 1.2016212 1.201621 0.4179079
##                     ACF1
## Training set -0.03687893
## Test set              NA
```

Both of my HW's RMSE (damped and undamped) beat the the RMSE of naive approach, and in fact by a very wide margin.

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?

My prepared dataset, which I used in Secion 3, Exercise 8 is *myts.train*. I just used the same in 7.8e. In the below code-chunk, I'll first apply Box-Cox transformation and then do STL-decomposition. Although this can be achieved in 3 or more steps, I'll do it in one step, for simplicity.

```
stlf_boxcox_myts.train <- stlf(myts.train, lambda = BoxCox.lambda(myts.train))
```

And here is the ETS transformation on seasonally adjusted data. From the wording of problem 7.9, I first thought, I would have to apply ETS transformation on stlf_boxcox_myts.train, but that didn't work. So, I interpreted it to mean what I did below.

```
ets_myts.train <- ets(seasadj(decompose(myts.train, "multiplicative")))
```

Now, in the following, I'll compare the accuracies of both Box-Cox-STLF transformation and ETS transformation.

Box-Cox-STLF transformation first.

```r
accuracy(stlf_boxcox_myts.train, myts.test)
```

```
##                       ME      RMSE       MAE       MPE     MAPE      MASE
## Training set   -1.077737  26.34835  19.95571 -0.020850 1.675349 0.2655976
## Test set     -101.367987 117.88372 101.84801 -3.846709 3.865079 1.3555310
##                     ACF1 Theil's U
## Training set -0.04737883        NA
## Test set      0.52778217  0.656884
```

ETS transformation.

```r
accuracy(forecast(ets_myts.train), myts.test)
```

```
##                      ME      RMSE       MAE       MPE     MAPE      MASE
## Training set   2.007835  29.33086  21.00199  0.133939 1.731450 0.2797569
## Test set     -38.373703 154.24915 113.08350 -1.798220 4.264931 1.5063282
##                     ACF1 Theil's U
## Training set -0.02682440        NA
## Test set      0.07396657 0.8637663
```

It's clear that the RMSE of Box-Cox-STLF transformation outperforms ETS transformation.

But it's not better than the best previous forecast on test set.

Marker: 624-05