

IS624 - Assignment5

James Quacinella

07/07/2015

Contents

Question 6.1	2
Question 6.2	3
Question 7.3	9
Question 7.4	21

Question 6.1

Show that a 3×5 MA is equivalent to a 7-term weighted moving average with weights of 0.067, 0.133, 0.200, 0.200, 0.133, and 0.067.

Answer:

$$\text{MA: } \hat{T}_t = \frac{1}{m} \sum_{j=-k}^k y_{t+j}$$

With $m = 5, k = 2$:

$$\begin{aligned} \hat{T}_{5,t} &= \frac{1}{5} \sum_{j=-2}^2 y_{t+j} \\ &= \frac{1}{5} * (y_{t-2} + y_{t-1} + y_t + y_{t+1} + y_{t+2}) \end{aligned}$$

With $m = 3, k = 1$:

$$\begin{aligned} \hat{T}_{3,t} &= \frac{1}{3} \sum_{j=-1}^1 y_{t+j} \\ &= \frac{1}{3} * (y_{t-1} + y_t + y_{t+1}) \end{aligned}$$

To find 3 x 5 MA, the y values in the above equation are the values from $\hat{T}_{5,t}$:

$$\begin{aligned} \hat{T}_{3 \times 5,t} &= \frac{1}{3} \sum_{j=-1}^1 y_{t+j} \\ &= \frac{1}{3} * (\hat{T}_{5,t-1} + \hat{T}_t + \hat{T}_{5,t+1}) \end{aligned}$$

The three terms above:

$$\hat{T}_{5,t-1} = \frac{1}{5}y_{t-3} + \frac{1}{5}y_{t-2} + \frac{1}{5}y_{t-1} + \frac{1}{5}y_t + \frac{1}{5}y_{t+1} + 0 * y_{t+2} + 0 * y_{t+3}$$

$$\hat{T}_t = 0 * y_{t-3} + \frac{1}{5}y_{t-2} + \frac{1}{5}y_{t-1} + \frac{1}{5}y_t + \frac{1}{5}y_{t+1} + \frac{1}{5} * y_{t+2} + 0 * y_{t+3}$$

$$\hat{T}_{5,t+1} = 0 * y_{t-3} + 0 * y_{t-2} + \frac{1}{5}y_{t-1} + \frac{1}{5}y_t + \frac{1}{5}y_{t+1} + \frac{1}{5} * y_{t+2} + \frac{1}{5}y_{t+3}$$

Adding them together:

$$\begin{aligned} \hat{T}_{5,t-1} + \hat{T}_t + \hat{T}_{5,t+1} &= \\ \frac{1}{5}y_{t-3} + \frac{2}{5}y_{t-2} + \frac{3}{5}y_{t-1} + \frac{3}{5}y_t + \frac{3}{5}y_{t+1} + \frac{2}{5} * y_{t+2} + \frac{1}{5}y_{t+3} \end{aligned}$$

Substituting:

$$\begin{aligned} \hat{T}_{3 \times 5,t} &= \frac{1}{3} * (\hat{T}_{5,t-1} + \hat{T}_t + \hat{T}_{5,t+1}) \\ &= \frac{1}{15}y_{t-3} + \frac{2}{15}y_{t-2} + \frac{1}{5}y_{t-1} + \frac{1}{5}y_t + \frac{1}{5}y_{t+1} + \frac{2}{15} * y_{t+2} + \frac{1}{15}y_{t+3} \end{aligned}$$

Therefore the coefficients are the ones listed in the question.

Question 6.2

The data below represent the monthly sales (in thousands) of product A for a plastics manufacturer for years 1 through 5 (data set plastics).

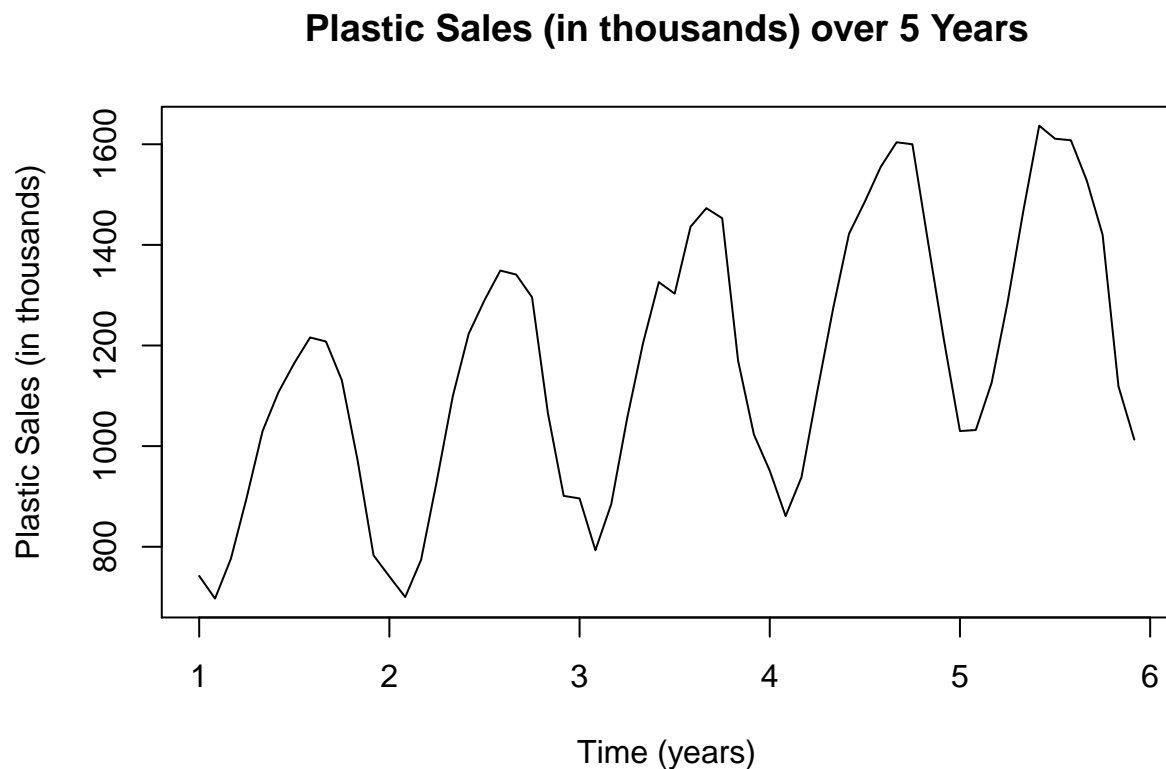
plastics

```
##      Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
## 1  742   697   776   898 1030 1107 1165 1216 1208 1131  971  783
## 2  741   700   774   932 1099 1223 1290 1349 1341 1296 1066  901
## 3  896   793   885 1055 1204 1326 1303 1436 1473 1453 1170 1023
## 4  951   861   938 1109 1274 1422 1486 1555 1604 1600 1403 1209
## 5 1030 1032 1126 1285 1468 1637 1611 1608 1528 1420 1119 1013
```

a) Plot the time series of sales of product A. Can you identify seasonal fluctuations and/or a trend?

Yes, there is a clear upward trend and a seasonal component in this time series:

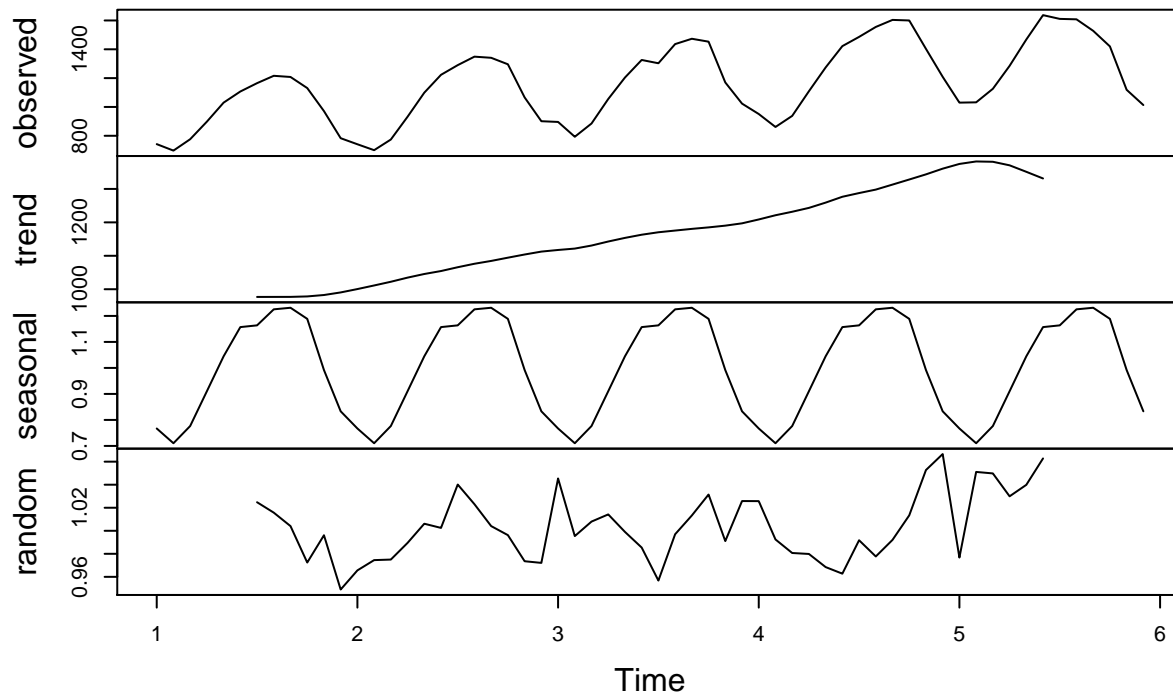
```
plot(plastics, main = "Plastic Sales (in thousands) over 5 Years",
     ylab = "Plastic Sales (in thousands)", xlab = "Time (years)")
```



b) Use a classical multiplicative decomposition to calculate the trend-cycle and seasonal indices.

```
plastics.fit <- decompose(plastics, type = "multiplicative")
plot(plastics.fit)
```

Decomposition of multiplicative time series



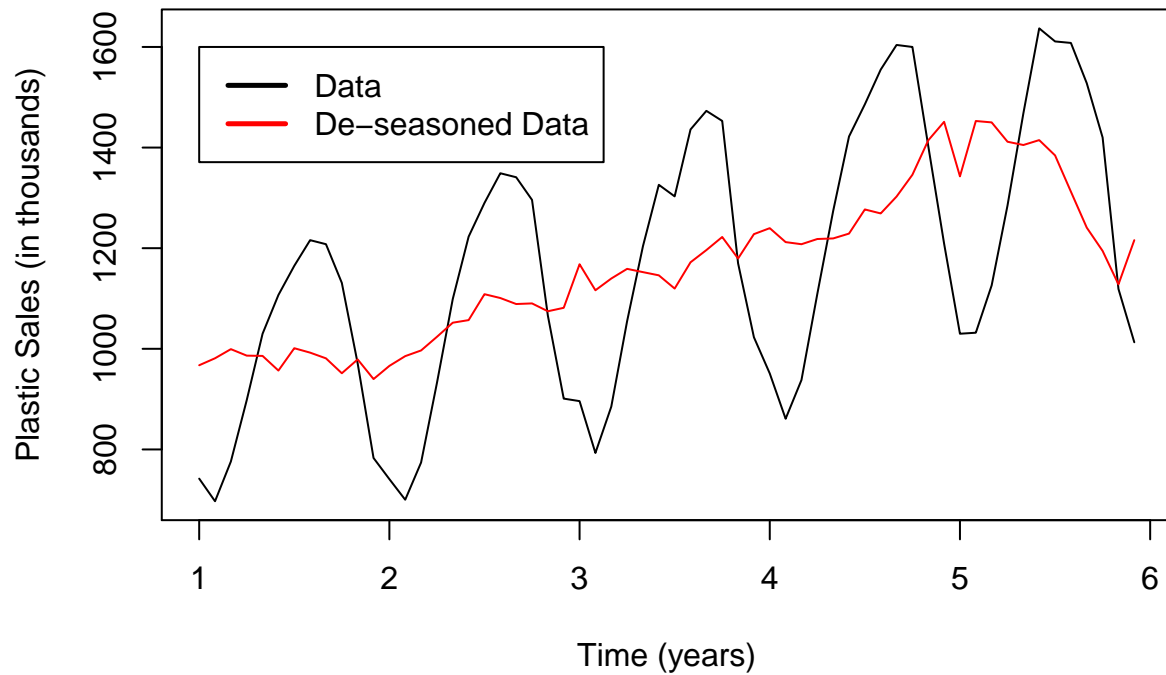
c) Do the results support the graphical interpretation from part (a)?

Yes, there is an upward trend in the first component and a pretty stable seasonal component.

d) Compute and plot the seasonally adjusted data.

```
plastics.seasadj <- seasadj(plastics.fit)
plot(plastics, main = "Plastic Sales (in thousands) over 5 Years",
     ylab = "Plastic Sales (in thousands)", xlab = "Time (years)")
lines(plastics.seasadj, col = "red")
legend(1, 1600, c("Data", "De-seasoned Data"), lty = c(1,
1), lwd = c(2.5, 2.5), col = c("black", "red"))
```

Plastic Sales (in thousands) over 5 Years



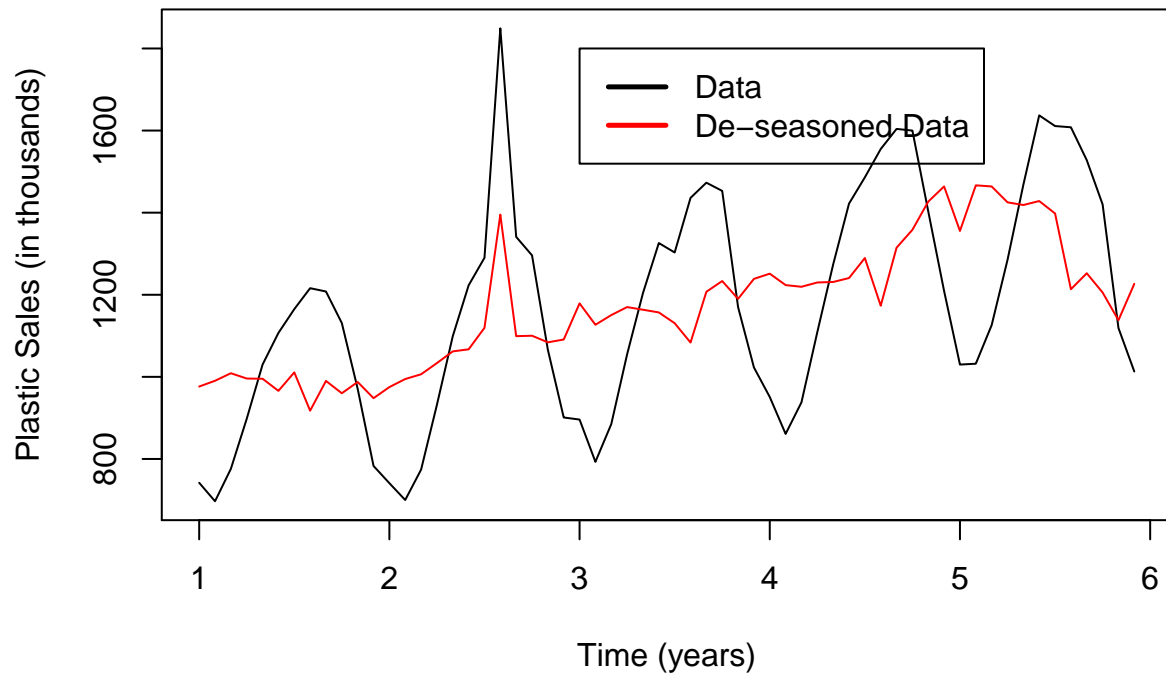
- e) Change one observation to be an outlier (e.g., add 500 to one observation), and recompute the seasonally adjusted data. What is the effect of the outlier?

```
plastics.outlier <- plastics
plastics.outlier[20] <- plastics[20] + 500

plastics.outlier.fit <- decompose(plastics.outlier,
    type = "multiplicative")

plot(plastics.outlier, main = "Plastic Sales (in thousands) over 5 Years",
    ylab = "Plastic Sales (in thousands)", xlab = "Time (years)")
lines(seasadj(plastics.outlier.fit), col = "red")
legend(3, 1800, c("Data", "De-seasoned Data"), lty = c(1,
    1), lwd = c(2.5, 2.5), col = c("black", "red"))
```

Plastic Sales (in thousands) over 5 Years



This looks like the single outlier really distorts the deseasoned data, which means that this form of decomposition is sensitive to outliers.

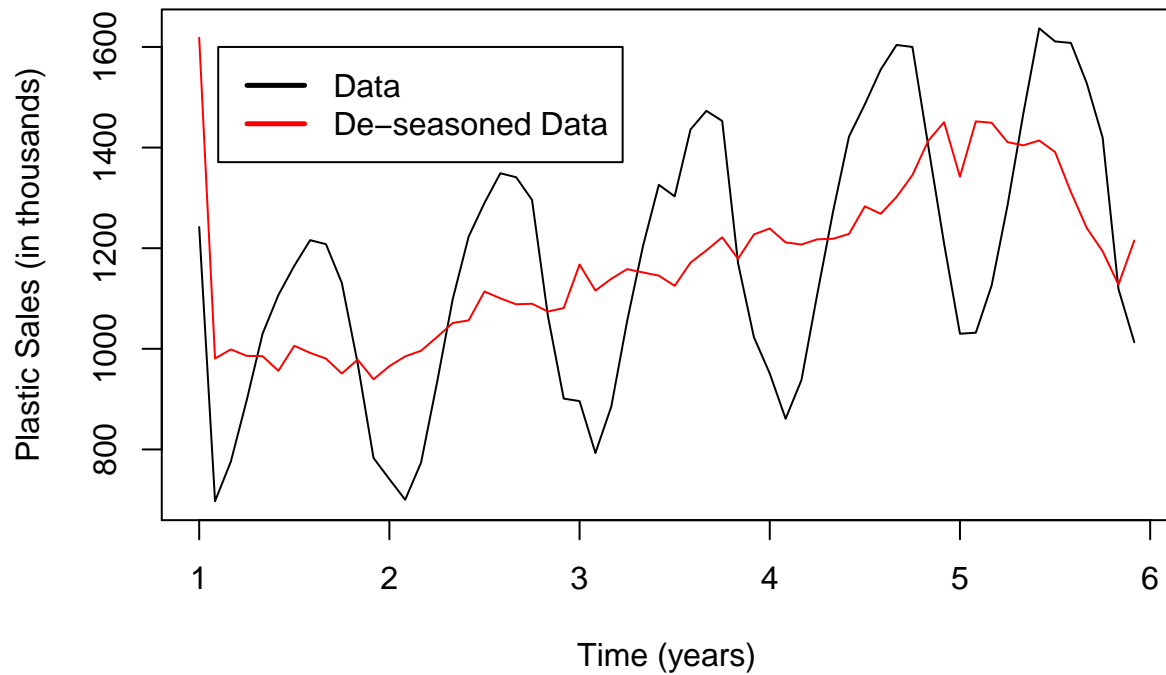
f) Does it make any difference if the outlier is near the end rather than in the middle of the time series?

```
plastics.outlier2 <- plastics
plastics.outlier2[1] <- plastics[1] + 500

plastics.outlier2.fit <- decompose(plastics.outlier2,
  type = "multiplicative")

plot(plastics.outlier2, main = "Plastic Sales (in thousands) over 5 Years",
  ylab = "Plastic Sales (in thousands)", xlab = "Time (years)")
lines(seasadj(plastics.outlier2.fit), col = "red")
legend(1.1, 1600, c("Data", "De-seasoned Data"), lty = c(1,
  1), lwd = c(2.5, 2.5), col = c("black", "red"))
```

Plastic Sales (in thousands) over 5 Years



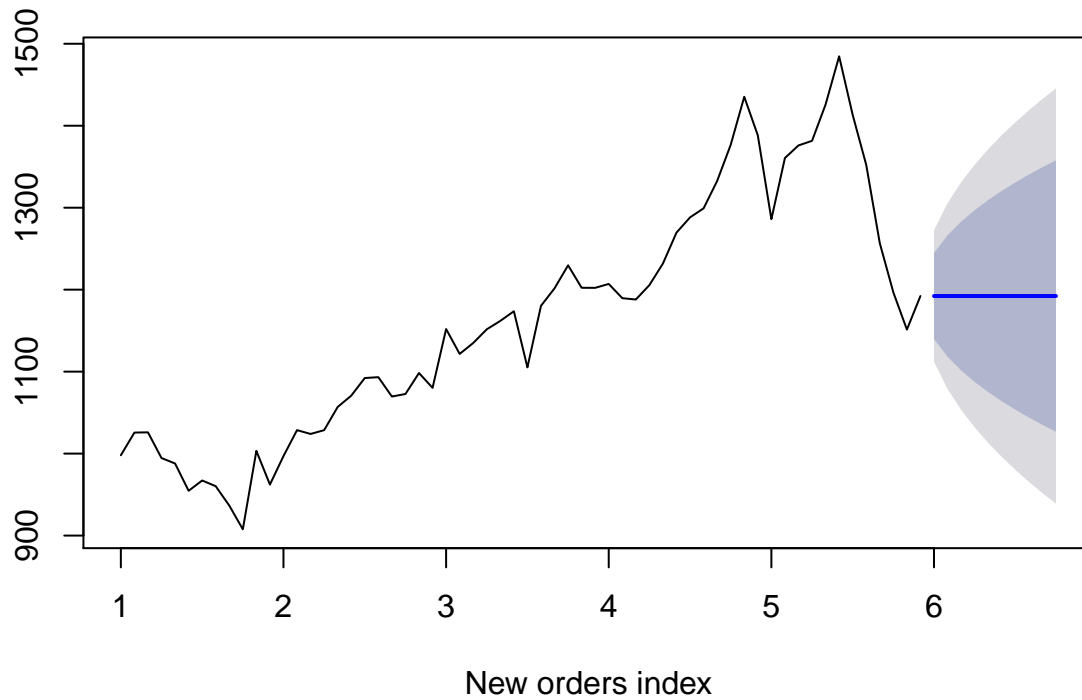
You can see the deseasoned data is very off at the beginning but since the edge points are only in a few of the calculations, it seems that the rest of the deseasoned data would be useful. Nonetheless, this shows that the classical decomposition method is sensitive to outliers.

- g) Use a random walk with drift to produce forecasts of the seasonally adjusted data. Reseasonalize the results to give forecasts on the original scale.

```
# STL Fit
plastics.stl.fit <- stl(plastics, t.window = 15, s.window = "periodic",
  robust = TRUE)

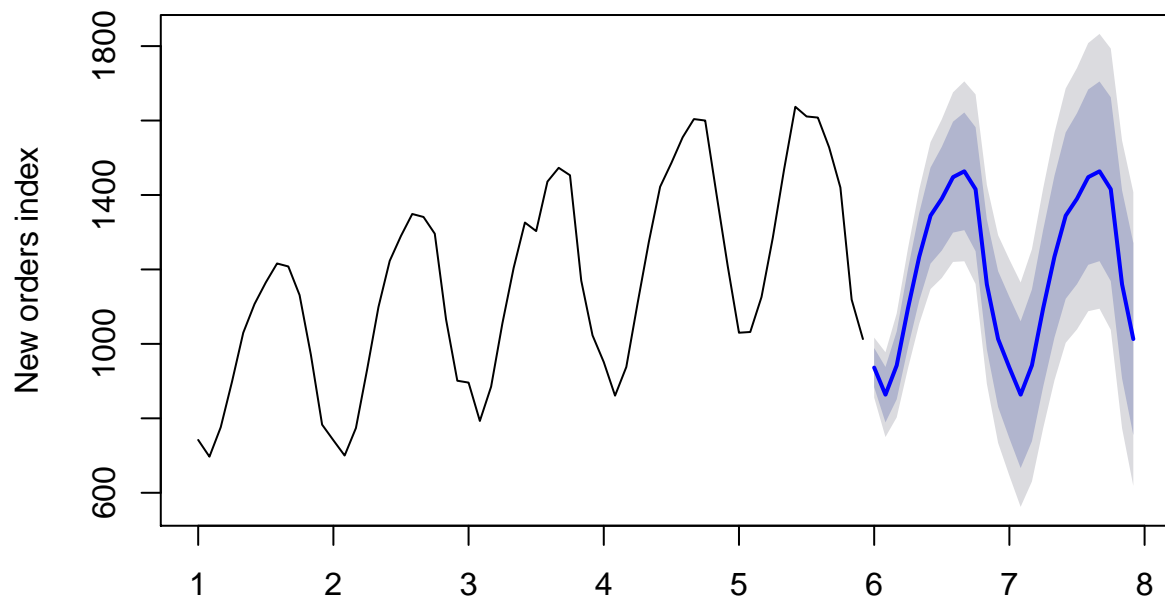
# Naive forecasting
plastics.stl.seasadj <- seasadj(plastics.stl.fit)
plot(naive(plastics.stl.seasadj), xlab = "New orders index")
```

Forecasts from Naive method



```
# Random walk forecasting
fcast <- forecast(plastics.stl.fit, method = "naive")
plot(fcast, ylab = "New orders index")
```

Forecasts from STL + Random walk



Using STL is the only way I could get a random walk forecast.

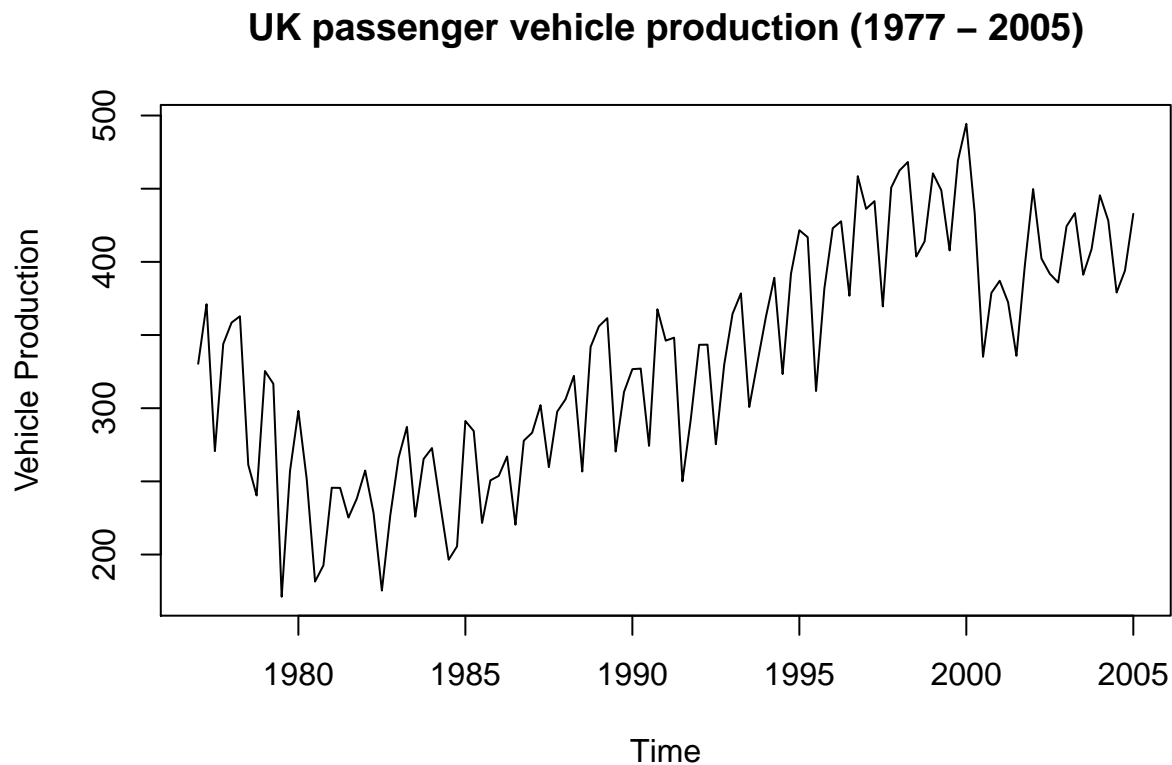
Reference: page 92 on <http://robjhyndman.com/talks/RevolutionR/4-Decomposition.pdf>

Question 7.3

For this exercise, use the quarterly UK passenger vehicle production data from 1977:1–2005:1 (data set ukcars).

- a) Plot the data and describe the main features of the series.

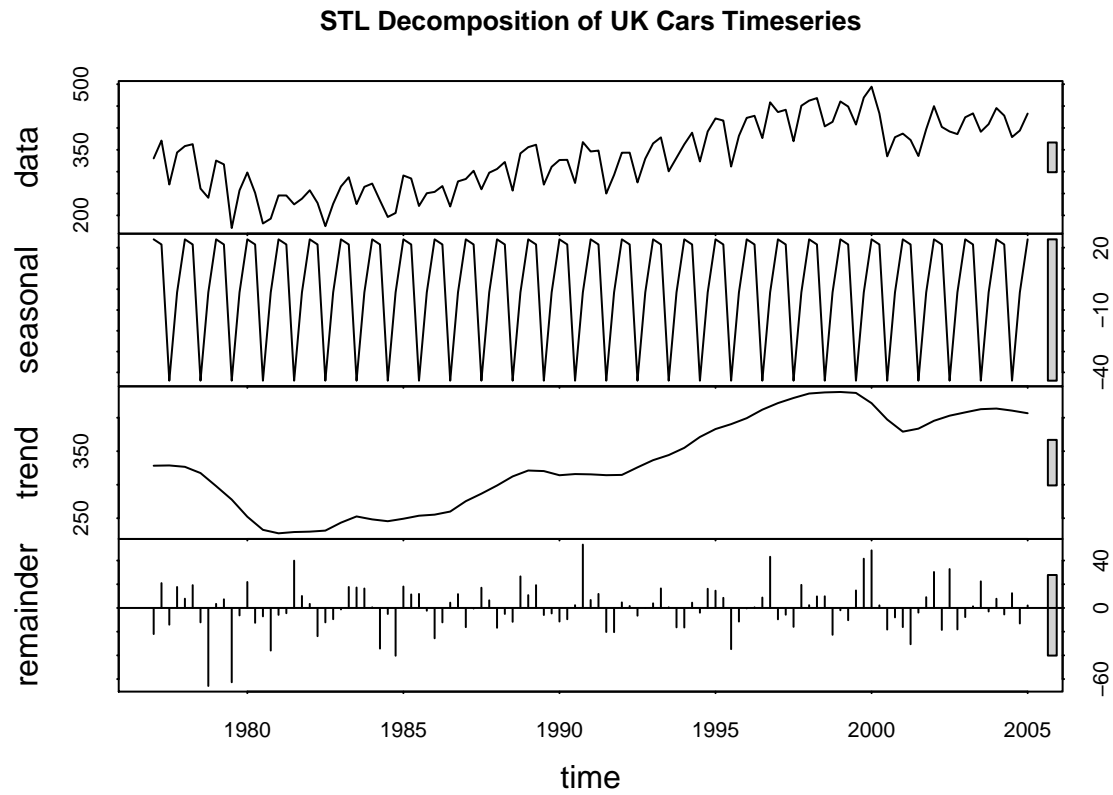
```
plot(ukcars, main = "UK passenger vehicle production (1977 - 2005)",  
     ylab = "Vehicle Production")
```



Looks like there is a general upward trend up until 2000, which we see a huge drop in production. We see a slight increase after that drop but there seems to be no trend at this point.

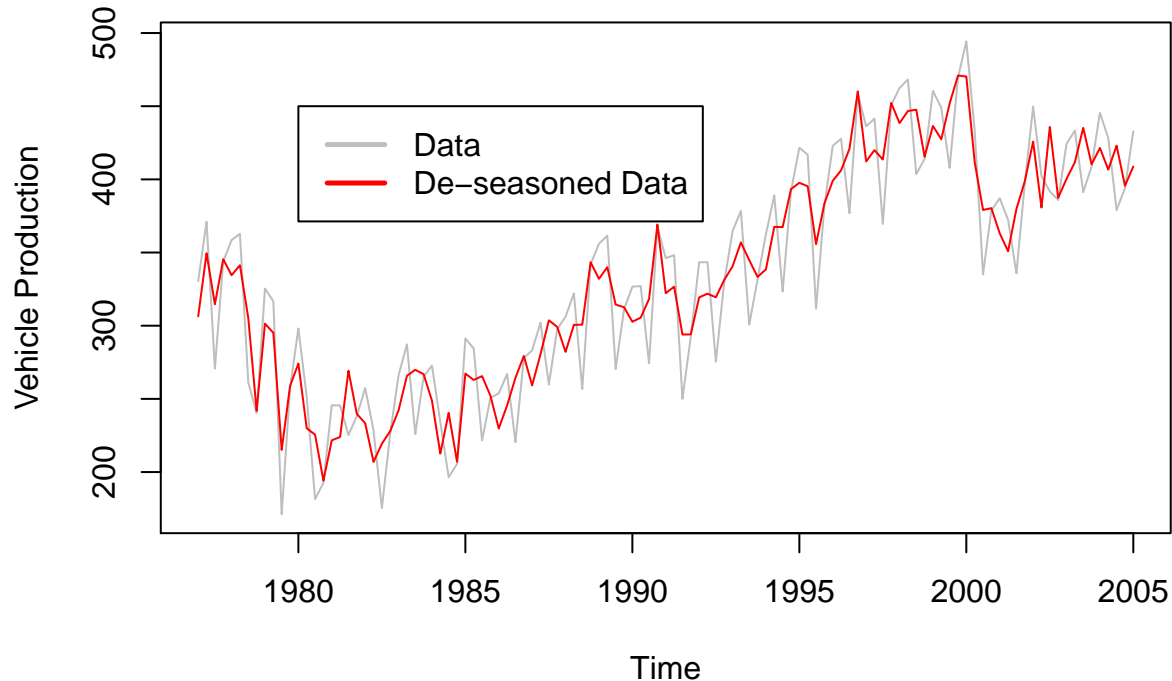
- b) Decompose the series using STL and obtain the seasonally adjusted data.

```
ukcars.stl = stl(ukcars, t.window = 11, s.window = "periodic",  
                 robust = TRUE)  
ukcars.seasadj = seasadj(ukcars.stl)  
  
# Plot the breakdown  
plot(ukcars.stl, main = "STL Decomposition of UK Cars Timeseries")
```



```
# Plot the seasonally adjusted data
plot(ukcars, col = "grey", main = "UK passenger vehicle production (1977 - 2005)",
     ylab = "Vehicle Production")
lines(ukcars.seasadj, col = "red")
legend(1980, 450, c("Data", "De-seasoned Data"), lty = c(1,
1), lwd = c(2.5, 2.5), col = c("grey", "red"))
```

UK passenger vehicle production (1977 – 2005)



- c) Forecast the next two years of the series using an additive damped trend method applied to the seasonally adjusted data. Then reseasonalize the forecasts. Record the parameters of the method and report the RMSE of the one-step forecasts from your method.

```
# Is this how to re-seasonalize?
ukcars.addDamped <- holt(ukcars, damped = TRUE, h = 8)
ukcars.seasadj.addDamped <- holt(ukcars.seasadj, damped = TRUE,
  h = 8)

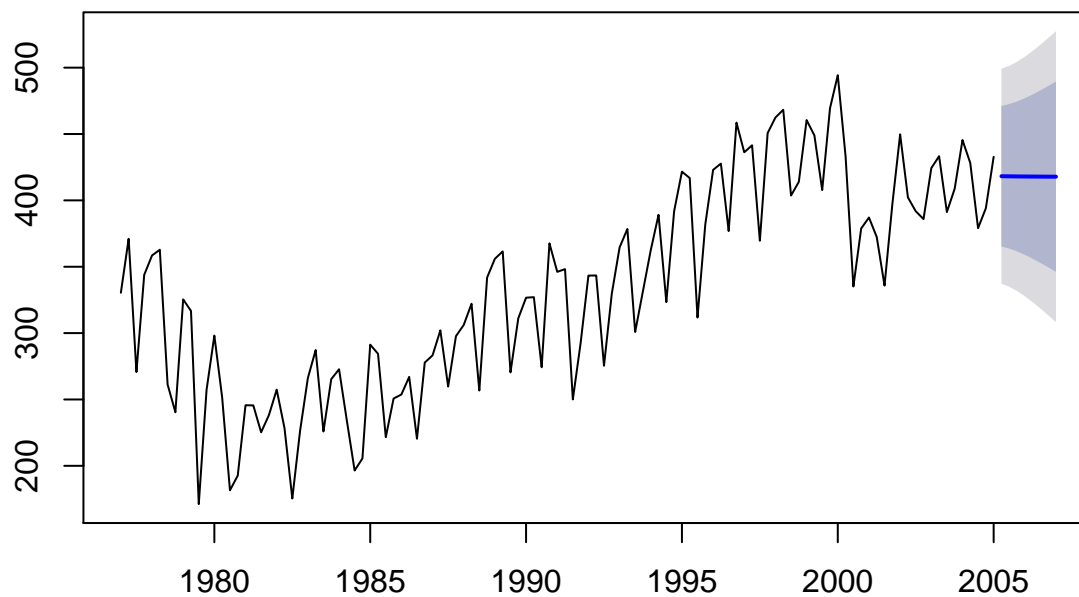
# Model params + RMSE of the one-step forecasts
summary(ukcars.addDamped$model)
```

```
## ETS(A,Ad,N)
##
## Call:
## holt(x = ukcars, h = 8, damped = TRUE)
##
## Smoothing parameters:
##   alpha = 0.1044
##   beta  = 0.0922
##   phi   = 0.8
##
## Initial states:
##   l = 348.7593
##   b = -5.0103
##
## sigma: 41.3135
##
##      AIC      AICc      BIC
```

```
## 1385.184 1385.745 1398.821
##
## Training set error measures:
##           ME      RMSE      MAE      MPE      MAPE      MASE
## Training set 1.665264 41.31353 34.56757 -1.069292 11.40068 1.12654
##           ACF1
## Training set 0.1322275
```

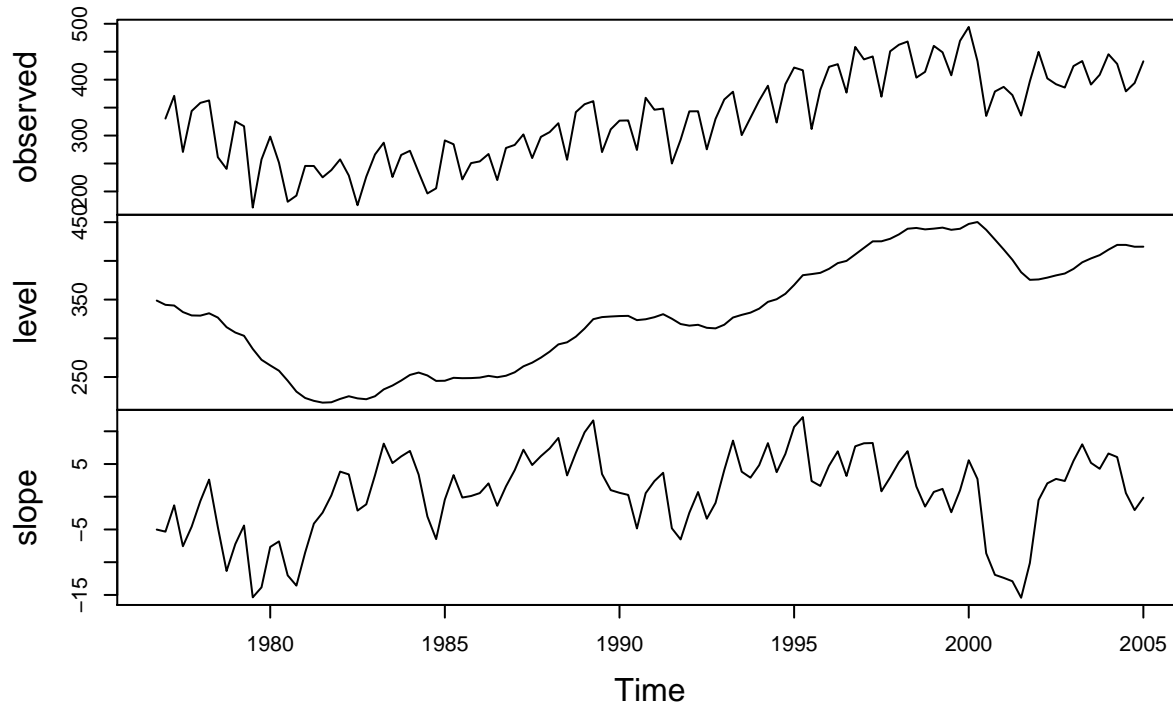
```
# Show the forecast for 2 years
plot(ukcars.addDamped)
```

Forecasts from Damped Holt's method



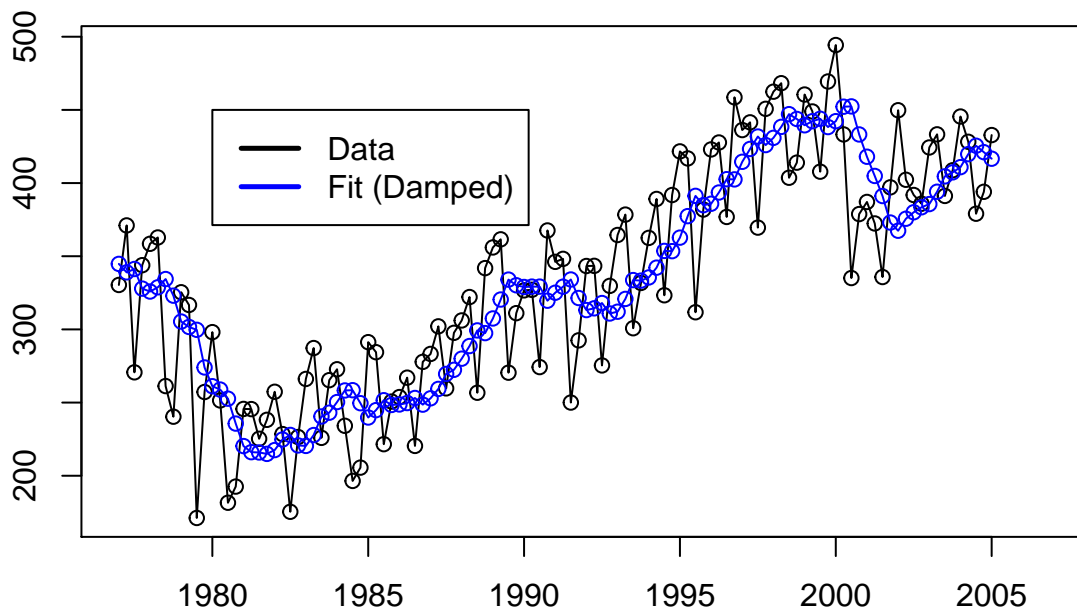
```
plot(ukcars.addDamped$model)
```

Decomposition by ETS(A,Ad,N) method



```
# Show the fitted value
plot(ukcars.addDamped, plot.conf = FALSE, fcol = "white",
     type = "o")
lines(fitted(ukcars.addDamped), col = "blue", type = "o")
legend(1980, 450, c("Data", "Fit (Damped)"), lty = c(1,
    1), lwd = c(2.5, 2.5), col = c("black", "blue"))
```

Forecasts from Damped Holt's method



- d) Forecast the next two years of the series using Holt's linear method applied to the seasonally adjusted data. Then reseasonalize the forecasts. Record the parameters of the method and report the RMSE of the one-step forecasts from your method.

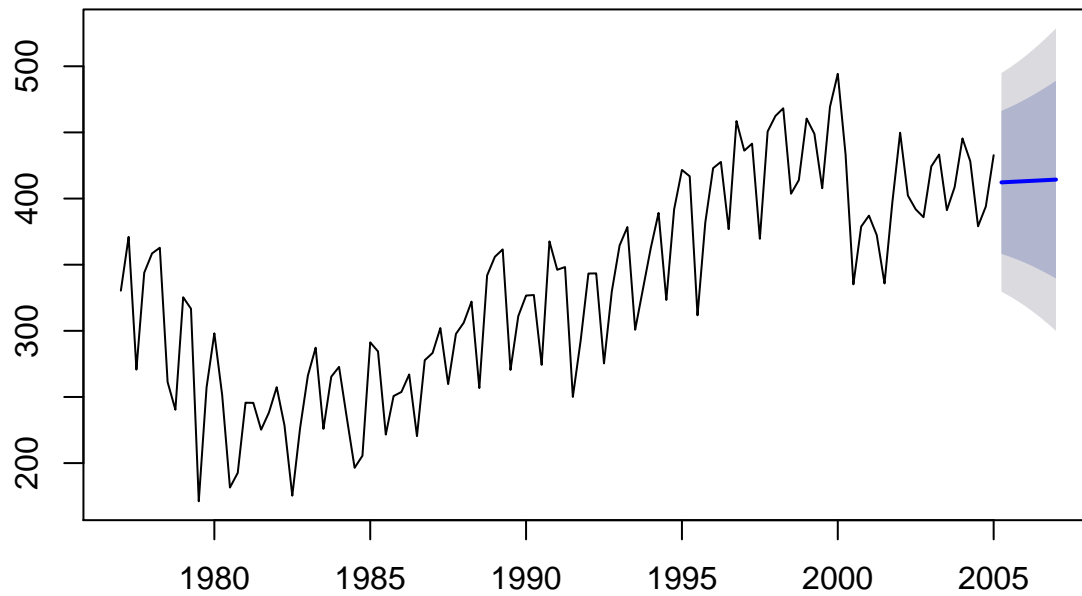
```
# Is this how to re-seasonalize?
ukcars.linear <- holt(ukcars, h = 8)
ukcars.seasadj.linear <- holt(ukcars.seasadj, h = 8)

# Model params + RMSE of the one-step forecasts
summary(ukcars.linear$model)
```

```
## ETS(A,A,N)
##
## Call:
## holt(x = ukcars, h = 8)
##
## Smoothing parameters:
##   alpha = 0.2503
##   beta  = 0.0217
##
## Initial states:
##   l = 341.4051
##   b = -5.6675
##
## sigma: 42.1756
##
##      AIC      AICc      BIC
## 1387.851 1388.221 1398.760
##
## Training set error measures:
##              ME      RMSE      MAE      MPE      MAPE      MASE
## Training set 2.438564 42.17557 35.11518 -0.3849453 11.55836 1.144387
##              ACF1
## Training set 0.09409457
```

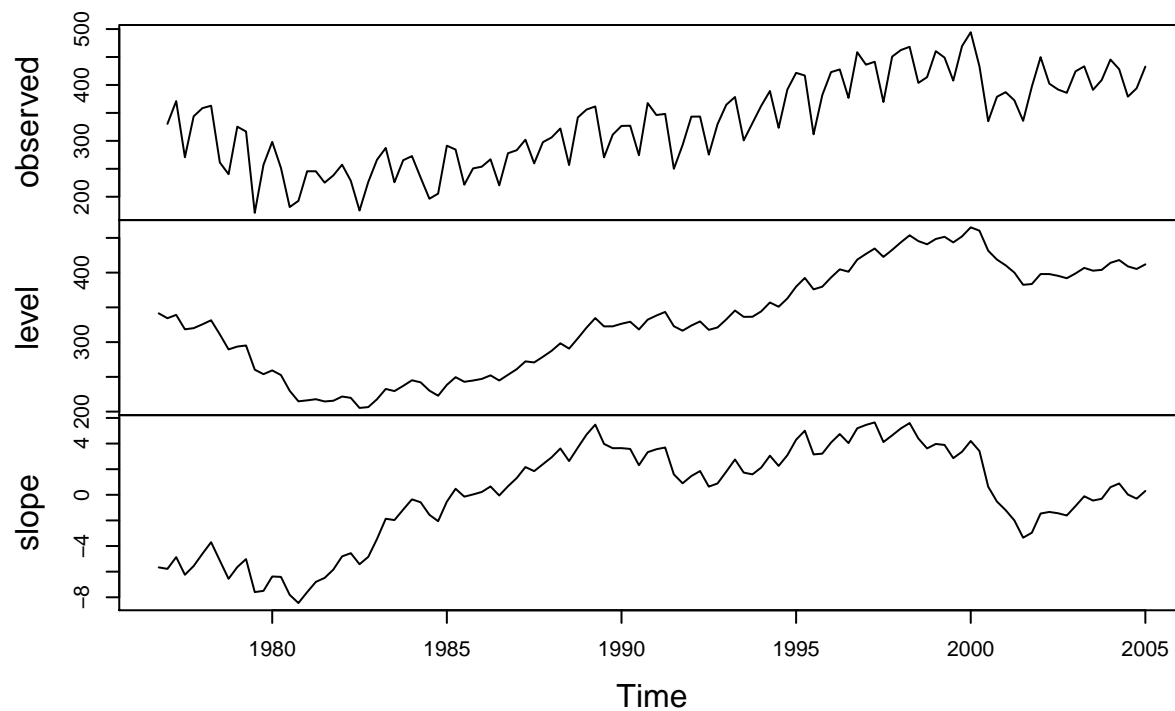
```
# Show the forecast for 2 years
plot(ukcars.linear)
```

Forecasts from Holt's method



```
plot(ukcars.linear$model)
```

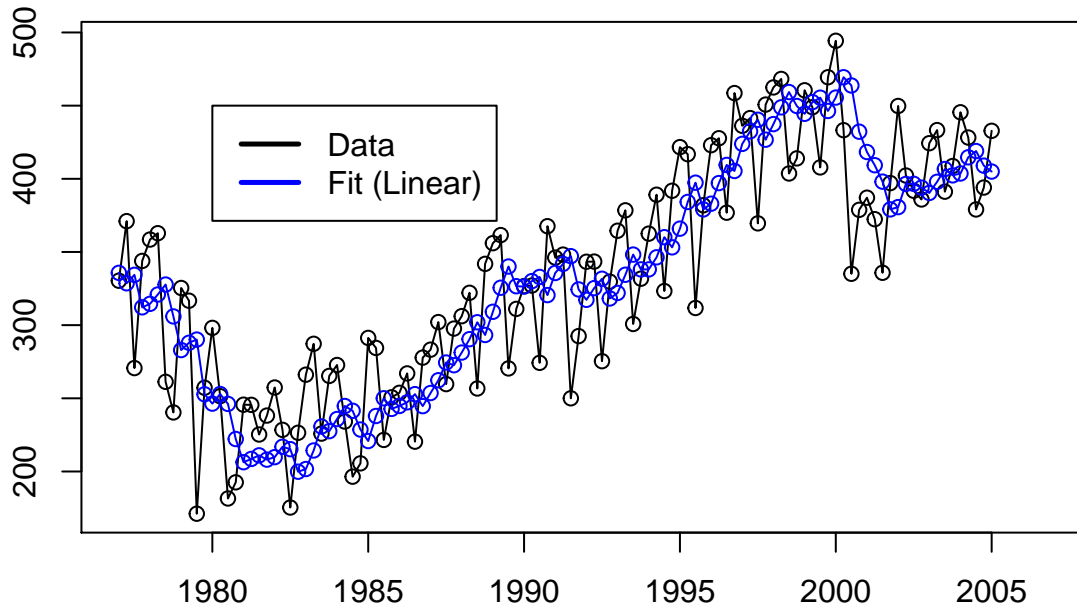
Decomposition by ETS(A,A,N) method



```
# Show the fitted value
plot(ukcars.linear, plot.conf = FALSE, fcol = "white",
     type = "o")
lines(fitted(ukcars.linear), col = "blue", type = "o")
```

```
legend(1980, 450, c("Data", "Fit (Linear)"), lty = c(1,
1), lwd = c(2.5, 2.5), col = c("black", "blue"))
```

Forecasts from Holt's method



e) Now use `ets()` to choose a seasonal model for the data.

```
# Is this how to re-seasonalize?
ukcars.ets <- holt(ukcars, h = 8, exponential = TRUE)
ukcars.ets.linear <- holt(ukcars.seasadj, h = 8)

# Model params + RMSE of the one-step forecasts
summary(ukcars.ets$model)
```

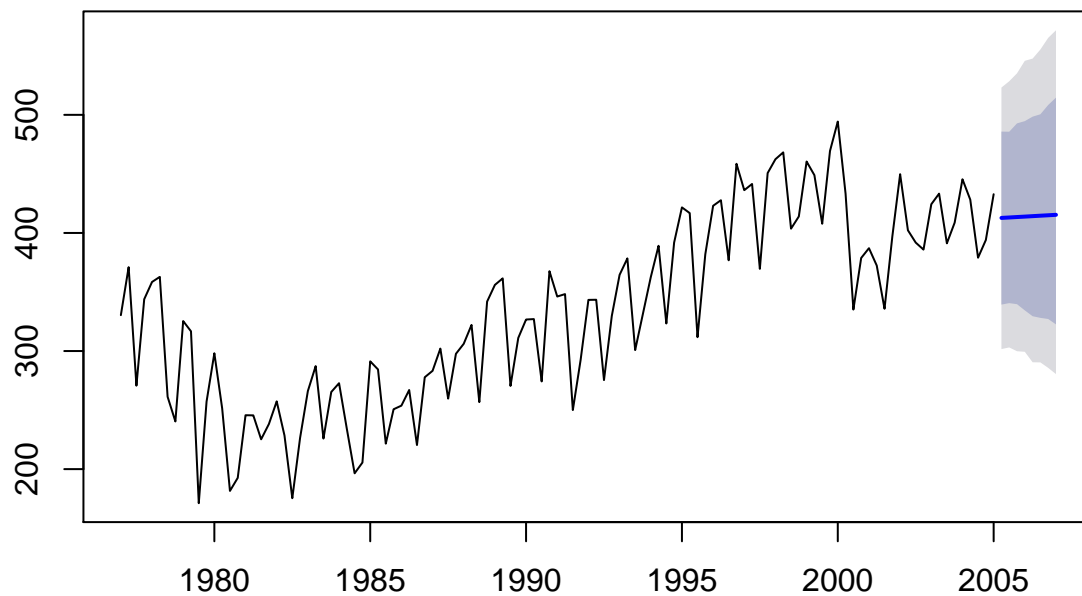
```
## ETS(M,M,N)
##
## Call:
## holt(x = ukcars, h = 8, exponential = TRUE)
##
## Smoothing parameters:
##   alpha = 0.2477
##   beta  = 0.0206
##
## Initial states:
##   l = 353.1012
##   b = 0.9769
##
## sigma: 0.1344
##
##      AIC      AICc      BIC
## 1394.695 1395.065 1405.605
##
```



```
## Training set error measures:  
##           ME      RMSE      MAE      MPE      MAPE      MASE  
## Training set 0.7669464 41.99915 34.825 -0.8920113 11.46995 1.13493  
##           ACF1  
## Training set 0.09236732
```

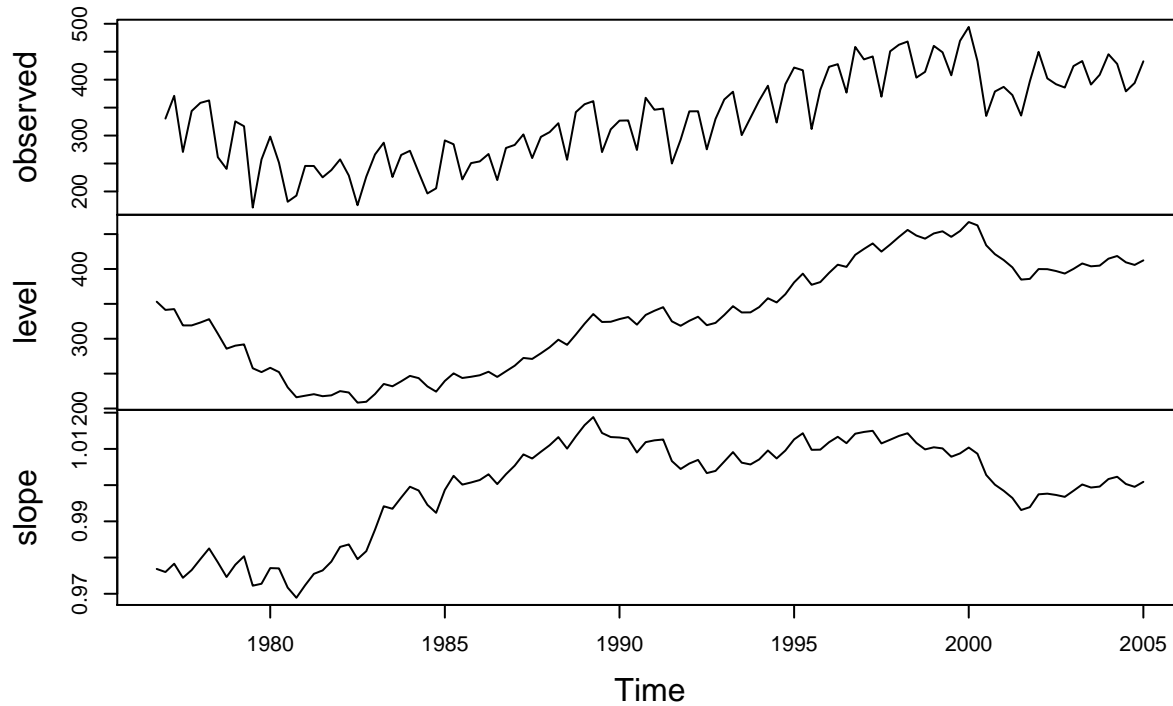
```
# Show the forecast for 2 years  
plot(ukcars.ets)
```

Forecasts from Holt's method with exponential trend



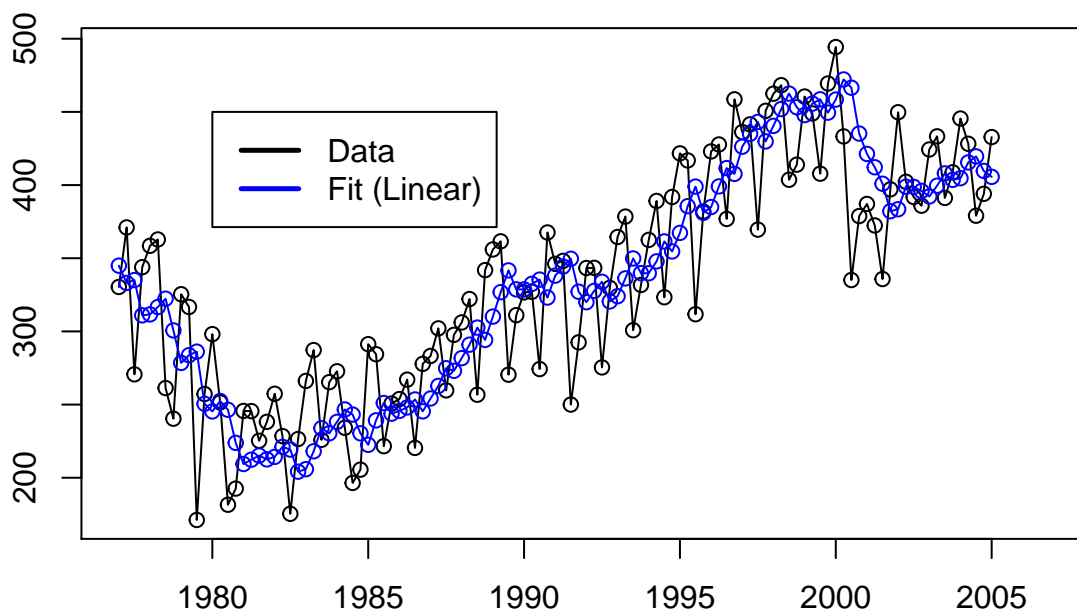
```
plot(ukcars.ets$model)
```

Decomposition by ETS(M,M,N) method



```
# Show the fitted value
plot(ukcars.ets, plot.conf = FALSE, fcol = "white",
     type = "o")
lines(fitted(ukcars.ets), col = "blue", type = "o")
legend(1980, 450, c("Data", "Fit (Linear)"), lty = c(1,
1), lwd = c(2.5, 2.5), col = c("black", "blue"))
```

Forecasts from Holt's method with exponential trend



- f) Compare the RMSE of the fitted model with the RMSE of the model you obtained using an STL decomposition with Holt's method. Which gives the better in-sample fits?

```
rbind(accuracy(ukcars.ets), accuracy(ukcars.linear),
      accuracy(ukcars.addDamped))
```

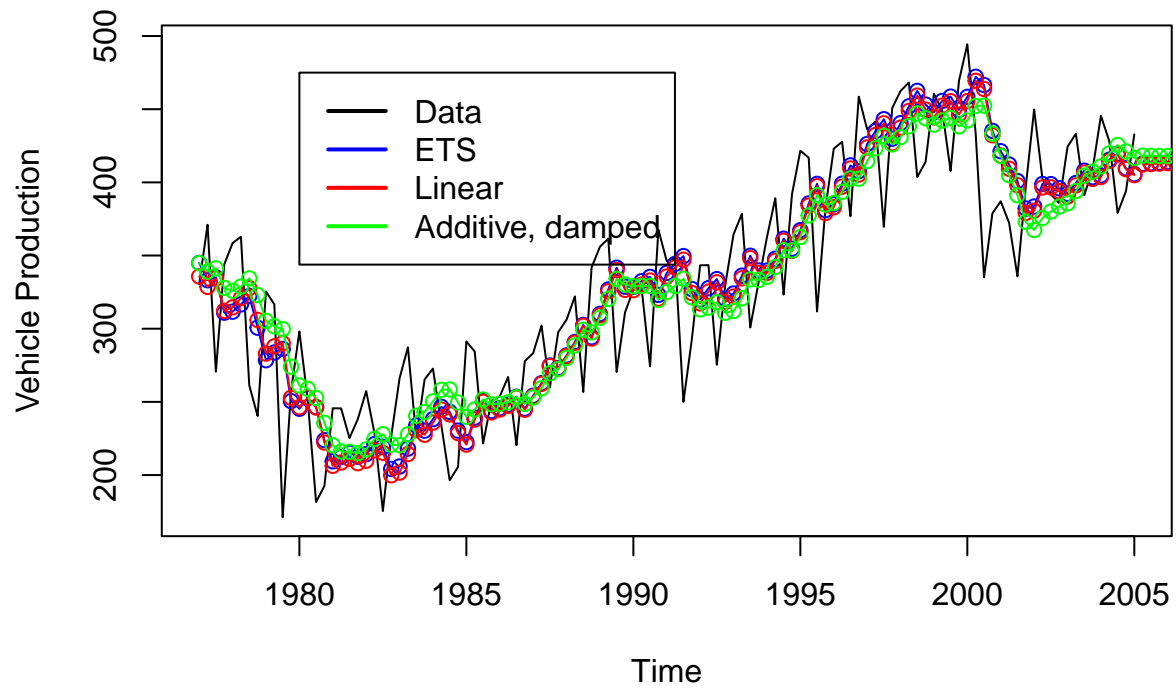
```
##              ME      RMSE      MAE      MPE      MAPE      MASE
## Training set 0.7669464 41.99915 34.82500 -0.8920113 11.46995 1.134930
## Training set 2.4385638 42.17557 35.11518 -0.3849453 11.55836 1.144387
## Training set 1.6652645 41.31353 34.56757 -1.0692923 11.40068 1.126540
##              ACF1
## Training set 0.09236732
## Training set 0.09409457
## Training set 0.13222750
```

As we can see, the accuracy of the models seems pretty consistent.

- g) Compare the forecasts from the two approaches? Which seems most reasonable?

```
plot(ukcars, main = "UK passenger vehicle production (1977 - 2005)\nwith predictions from models",
     ylab = "Vehicle Production")
lines(fitted(ukcars.ets), col = "blue", type = "o")
lines(ukcars.ets$mean, col = "blue", type = "o")
lines(fitted(ukcars.linear), col = "red", type = "o")
lines(ukcars.linear$mean, col = "red", type = "o")
lines(fitted(ukcars.addDamped), col = "green", type = "o")
lines(ukcars.addDamped$mean, col = "green", type = "o")
legend(1980, 475, c("Data", "ETS", "Linear", "Additive, damped"),
      lty = c(1, 1, 1, 1), lwd = c(2, 2, 2, 2), col = c("black",
        "blue", "red", "green"))
```

UK passenger vehicle production (1977 – 2005) with predictions from models



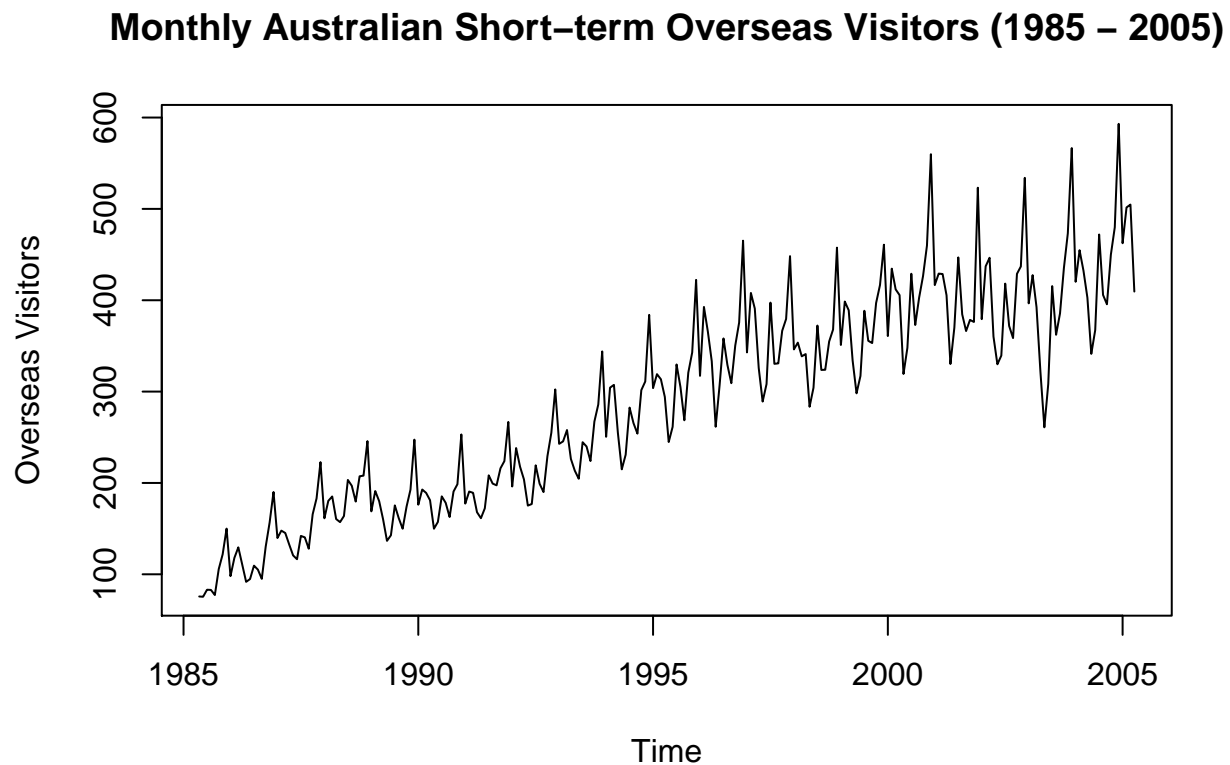
All the predictions seem to be consistent with one another.

Question 7.4

For this exercise, use the monthly Australian short-term overseas visitors data, May 1985–April 2005. (Data set: visitors.)

- a) Make a time plot of your data and describe the main features of the series.

```
plot(visitors, main = "Monthly Australian Short-term Overseas Visitors (1985 - 2005)",  
     ylab = "Overseas Visitors")
```



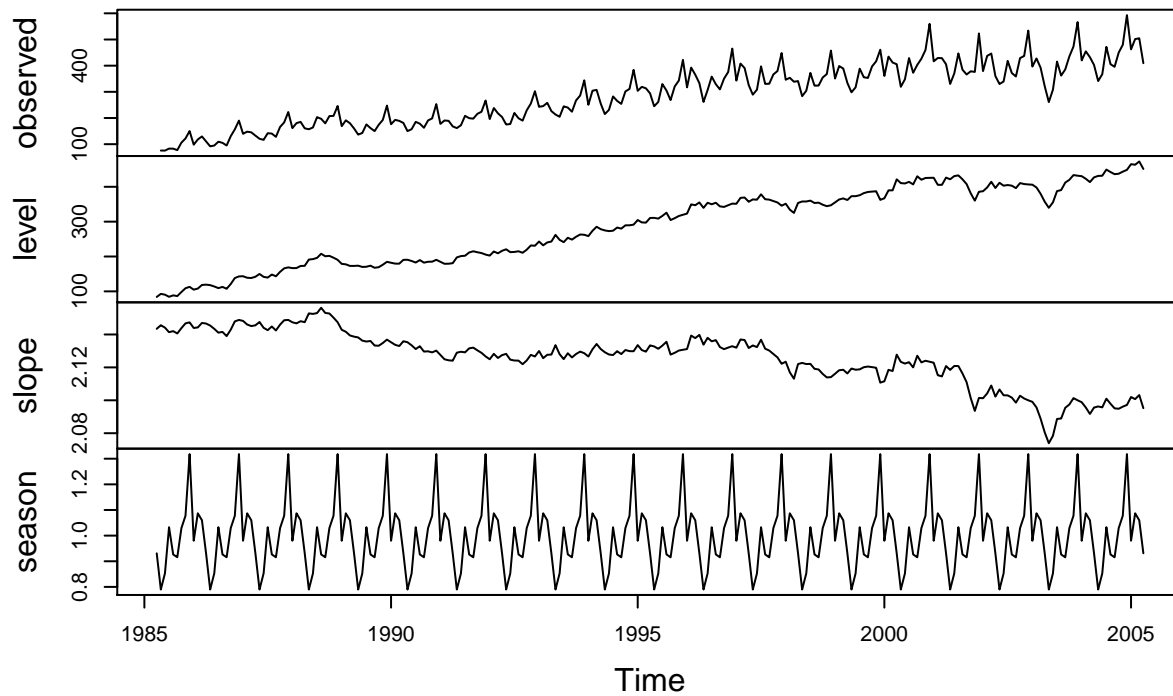
- b) Forecast the next two years using Holt-Winters' multiplicative method.

```
visitors.hw <- hw(visitors, seasonal = "multiplicative",  
                  h = 24)  
# summary(visitors.hw)  
accuracy(visitors.hw)
```

```
##              ME      RMSE      MAE      MPE      MAPE      MASE  
## Training set -0.8614726 14.52211 10.86884 -0.4799156 4.168399 0.4013761  
##              ACF1  
## Training set -0.03448764
```

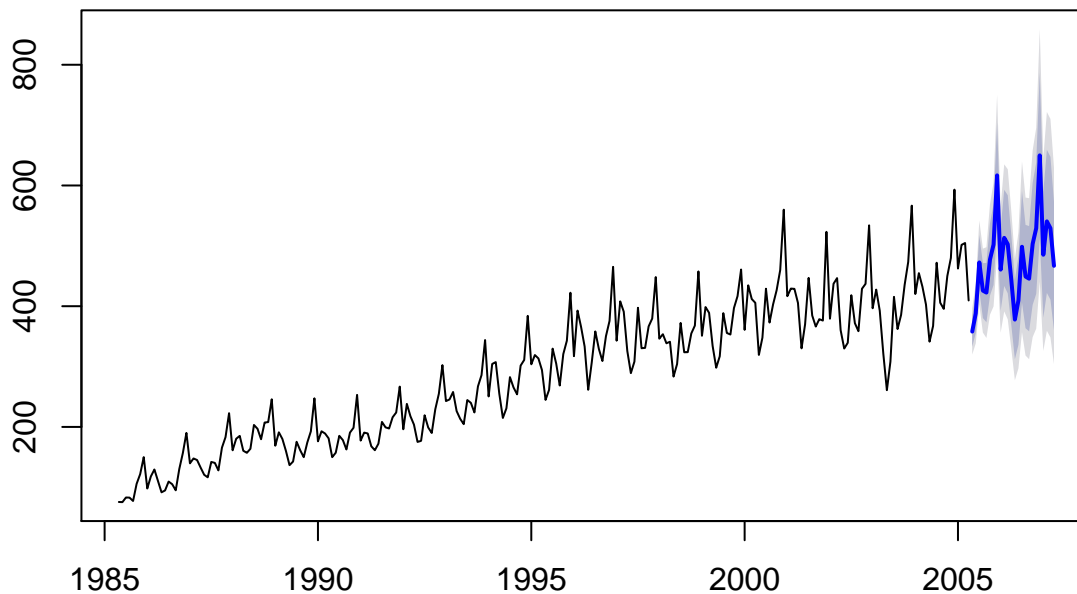
```
plot(visitors.hw$model)
```

Decomposition by ETS(M,A,M) method



```
plot(visitors.hw)
```

Forecasts from Holt-Winters' multiplicative method



c) Why is multiplicative seasonality necessary here?

From the book:

“The additive method is preferred when the seasonal variations are roughly constant through the series, while the multiplicative method is preferred when the seasonal variations are changing proportional to the level of the series”.

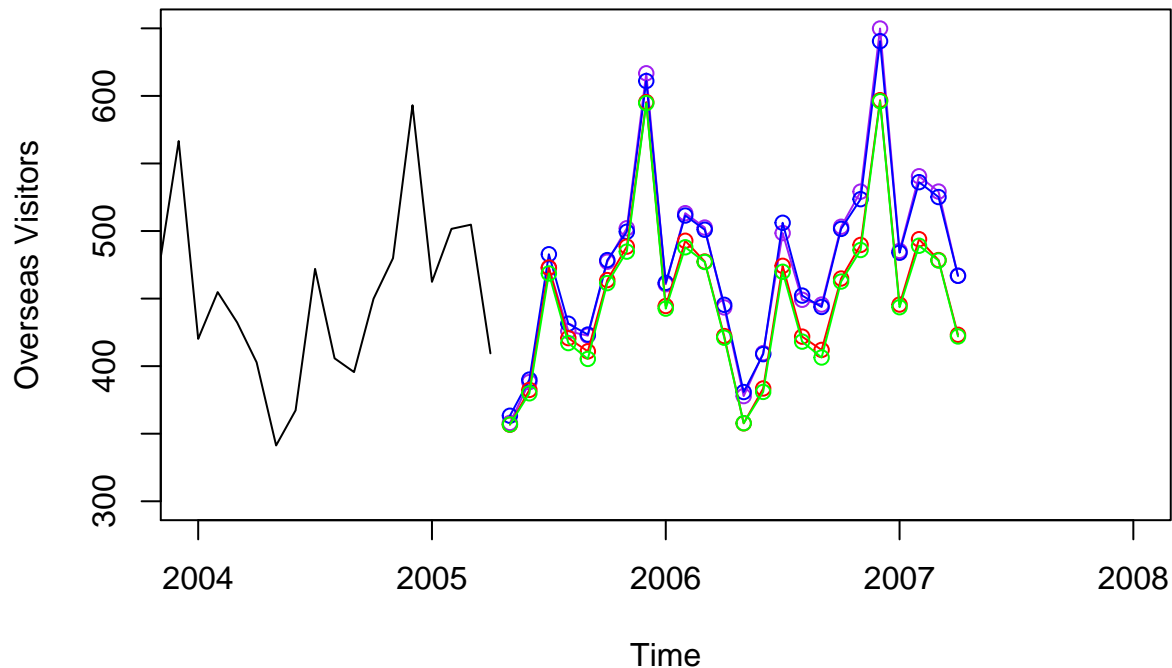
This seems appropriate since the data seems to show that as time goes on, the seasonal variation is getting larger (i.e. the spread towards the end is greater than the beginning of the time series).

d) Experiment with making the trend exponential and/or damped.

```
visitors.hw.trendexp <- hw(visitors, seasonal = "multiplicative",
  exponential = TRUE, h = 24)
visitors.hw.trenddamp <- hw(visitors, seasonal = "multiplicative",
  damped = TRUE, h = 24)
visitors.hw.trendboth <- hw(visitors, seasonal = "multiplicative",
  exponential = TRUE, damped = TRUE, h = 24)

plot(visitors, main = "Monthly Australian Short-term Overseas Visitors (1985 - 2005)",
  ylab = "Overseas Visitors", xlim = c(2004, 2008),
  ylim = c(300, 650))
# lines(fitted(visitors.hw), col='purple',
# type='o')
lines(visitors.hw$mean, col = "purple", type = "o")
# lines(fitted(visitors.hw.trendexp), col='blue',
# type='o')
lines(visitors.hw.trendexp$mean, col = "blue", type = "o")
# lines(fitted(visitors.hw.trenddamp), col='red',
# type='o')
lines(visitors.hw.trenddamp$mean, col = "red", type = "o")
# lines(fitted(visitors.hw.trendboth), col='green',
# type='o')
lines(visitors.hw.trendboth$mean, col = "green", type = "o")
legend(2000, 200, c("Data", "HW Method", "w/ trend exp",
  "w/ trend damped", "w/ trend damped+exp"), lty = c(1,
  1, 1, 1), lwd = c(2.5, 2.5, 2.5, 2.5), col = c("black",
  "purple", "blue", "red", "green"))
```

Monthly Australian Short-term Overseas Visitors (1985 – 2005)



The blue and purple forecasts are pretty identical, while the red and green ones are close as well.

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

```
accuracies <- rbind(accuracy(visitors.hw), accuracy(visitors.hw.trendexp),
                    accuracy(visitors.hw.trenddamp), accuracy(visitors.hw.trendboth))
rownames(accuracies) <- c("Holt-Winters' seasonal mult",
                          "with exp trend", "with damped trend", "with exp, damped trend")
accuracies
```

	ME	RMSE	MAE	MPE
Holt-Winters' seasonal mult	-0.8614726	14.52211	10.86884	-0.47991560
with exp trend	-0.6175624	14.68990	11.00618	-0.35580852
with damped trend	1.5236427	14.40219	10.64283	0.35913329
with exp, damped trend	0.5595893	14.46091	10.66091	-0.07611252

	MAPE	MASE	ACF1
Holt-Winters' seasonal mult	4.168399	0.4013761	-0.03448764
with exp trend	4.230296	0.4064480	0.08654357
with damped trend	4.057262	0.3930297	0.01526565
with exp, damped trend	4.075176	0.3936972	-0.02683110

Tough to say, as the RMSE seems to be all very close. Technically, the Holt-Winters' multiplicative method with a damped trend component is a slight favorite based on RMSE.

f) Now fit each of the following models to the same data:

- a multiplicative Holt-Winters' method;
- an ETS model;

- an additive ETS model applied to a Box-Cox transformed series;
- a seasonal naive method applied to the Box-Cox transformed series;
- an STL decomposition applied to the Box-Cox transformed data followed by an ETS model applied to the seasonally adjusted (transformed) data.
- For each model, look at the residual diagnostics and compare the forecasts for the next two years. Which do you prefer?

```
visitors.lambda <- BoxCox.lambda(visitors) # = 0.2775249
visitors.transformed <- BoxCox(visitors, visitors.lambda)

# Multiplicative Holt-Winters' method;
visitors.hw <- hw(visitors, seasonal = "multiplicative",
  h = 24)

# ETS model;
visitors.ets <- ets(visitors)
visitors.ets.forecast <- forecast(visitors.ets, h = 24)

# an additive ETS model applied to a Box-Cox
# transformed series;
visitors.ets.additive <- ets(visitors, additive = TRUE,
  lambda = visitors.lambda)
visitors.ets.additive.forecast <- forecast(visitors.ets.additive,
  h = 24)

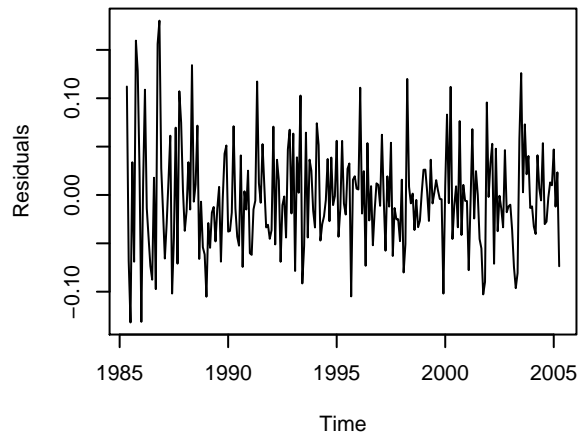
# a seasonal naive method applied to the Box-Cox
# transformed series;
visitors.seasonal.naive <- snaive(visitors, h = 24)

# an STL decomposition applied to the Box-Cox
# transformed data followed by an ETS model applied
# to the seasonally adjusted (transformed) data.
visitors.stl <- stl(visitors.transformed, t.window = 15,
  s.window = "periodic", robust = TRUE)
visitors.seasadj <- seasadj(visitors.stl)
visitors.stl.ets <- ets(visitors.seasadj)
visitors.stl.ets.forecast <- forecast(visitors.stl.ets,
  h = 24)

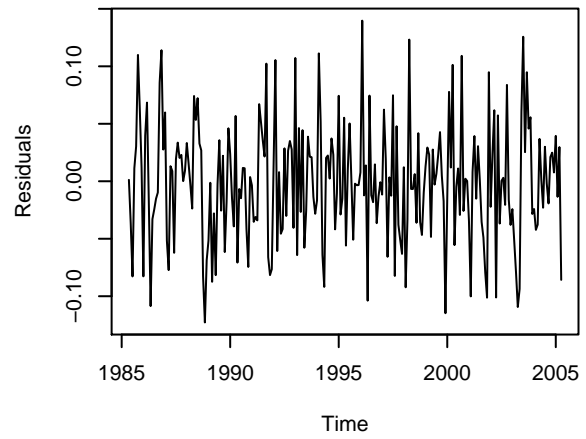
# plot residuals
par(mfrow = c(3, 2), oma = c(0, 0, 2, 0))
plot(residuals(visitors.hw), main = "Residuals for Holt-Winters' Model",
  ylab = "Residuals")
plot(residuals(visitors.ets.forecast), main = "Residuals from ETS Model",
  ylab = "Residuals")
plot(residuals(visitors.ets.additive.forecast), main = "Residuals from ETS Additive Model",
  ylab = "Residuals")
plot(residuals(visitors.seasonal.naive), main = "Residuals from Seasonal Naive Model",
  ylab = "Residuals")
plot(residuals(visitors.stl.ets.forecast), main = "Residuals from STL then ETS Model",
  ylab = "Residuals")
mtext("Residual Analysis of Models", outer = TRUE,
  cex = 1.5)
```

Residual Analysis of Models

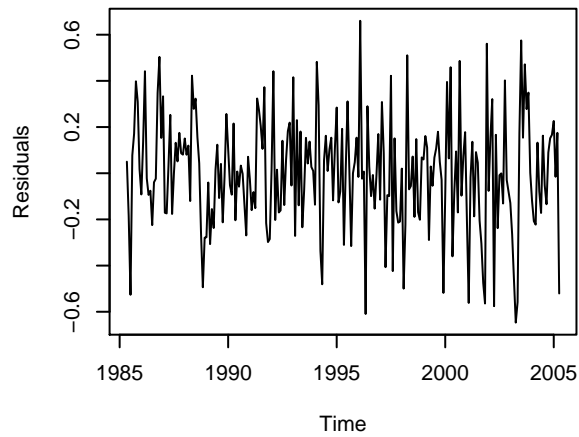
Residuals for Holt-Winters' Model



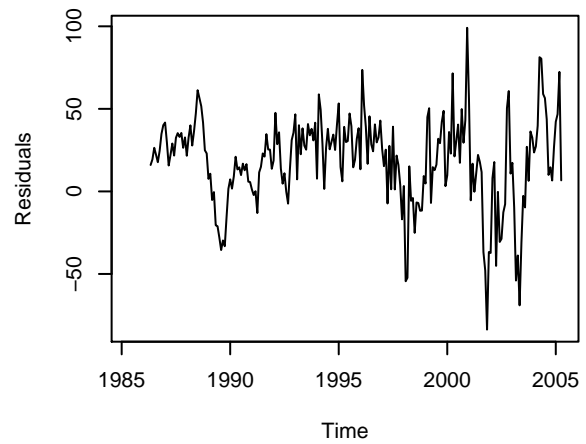
Residuals from ETS Model



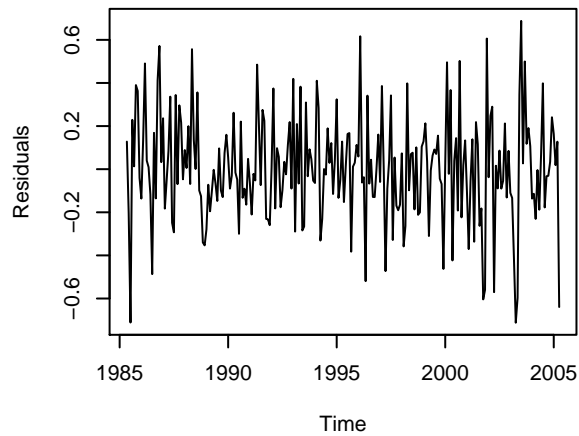
Residuals from ETS Additive Model



Residuals from Seasonal Naive Model

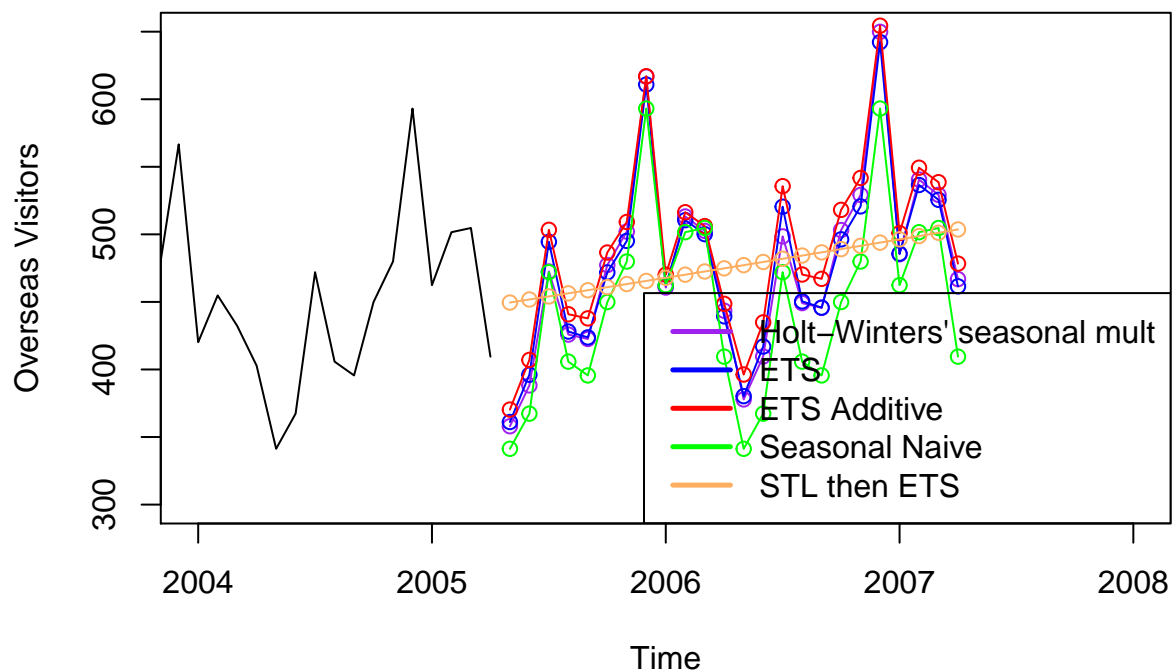


Residuals from STL then ETS Model



```
# Plot forecasts from various models
par(mfrow = c(1, 1), xpd = FALSE)
plot(visitors, main = "Monthly Australian Short-term Overseas Visitors (1985 - 2005)",
     ylab = "Overseas Visitors", xlim = c(2004, 2008),
     ylim = c(300, 650))
lines(visitors.hw$mean, col = "purple", type = "o")
lines(visitors.ets.forecast$mean, col = "blue", type = "o")
lines(visitors.ets.additive.forecast$mean, col = "red",
     type = "o")
lines(visitors.seasonal.naive$mean, col = "green",
     type = "o")
lines(InvBoxCox(visitors.stl.ets.forecast$mean, visitors.lambda),
     col = "#fdae61", type = "o")
legend("bottomright", inset = c(0, 0), legend = c("Holt-Winters' seasonal mult",
     "ETS", "ETS Additive", "Seasonal Naive", "STL then ETS"),
     lty = c(1, 1, 1, 1), lwd = c(2.5, 2.5, 2.5, 2.5),
     col = c("purple", "blue", "red", "green", "#fdae61"))
```

Monthly Australian Short-term Overseas Visitors (1985 – 2005)



```
accuracies <- rbind(accuracy(visitors.hw), accuracy(visitors.ets),
    accuracy(visitors.ets.additive), accuracy(visitors.seasonal.naive),
    accuracy(visitors.stl.ets))
rownames(accuracies) <- c("Holt-Winters' seasonal mult",
    "ETS", "ETS Additive", "Seasonal Naive", "STL then ETS")
accuracies
```

	ME	RMSE	MAE	MPE
Holt-Winters' seasonal mult	-0.861472602	14.5221105	10.8688434	-0.47991560
ETS	-0.956474336	15.8469958	11.5214997	-0.43070776

## ETS Additive	-0.164069702	15.5690033	11.3425826	-0.11052159
## Seasonal Naive	18.223684211	32.5694117	27.0789474	7.01179783
## STL then ETS	0.001698817	0.2472803	0.1877206	0.01135358
##	MAPE	MASE	ACF1	
## Holt-Winters' seasonal mult	4.168399	0.4013761	-0.03448764	
## ETS	4.075378	0.4254781	0.02434609	
## ETS Additive	4.016548	0.4188709	0.06357764	
## Seasonal Naive	10.129350	1.0000000	0.66004052	
## STL then ETS	1.432080	0.3836938	0.01686850	

The last model, which does STL then ETS, is not directly comparable since the ETS model is based off the seasonally adjusted data (i.e only the trend cycle component). However, it does a nice job of showing the predicted trend for 2 years.

Out of the other models, only the seasonal naive method does poorly with a much higher RMSE than the others. Most of the residual plots do show that the models are not making systemic errors, though the seasonal naive method has problems: the errors do not fluctuate around 0 and they diverge towards later years.