Assignment 2: Time Series Analysis – Fundamental Concepts

Andrew G. Dunn

January 21, 2016

Andrew G. Dunn, Northwestern University Predictive Analytics Program

Prepared for PREDICT-413: Time Series Analysis and Forecasting.

Formatted using the LaTeX, via pandoc and R Markdown. References managed using pandoc-citeproc.

Setup

```r
require(fBasics)    # for calculations
require(fpp)        # for data
require(knitr)      # for table output
require(ggplot2)    # for graphing
require(ggthemes)   # for graphing beautifully
require(gridExtra)  # for laying out graphs
```

Part 1

Consider the monthly returns for General Electric (GE) stock, Center for Research In Security Prices (CRSP) value-weighted intex (VW), CRSP equal-weighted index (EW), and S&P composite index (SP) from January 1981 to December 2013. The returns include dividend distributions. Data file is `m-ge3dx8113.txt` with column names permno of GE, date, ge, vwretd, ewretd, and sprtrn, respectively.

```r
d1 = read.table("data/m-ge3dx8113.txt", header=T)
head(d1)
```

```
##   PERMNO     date        ge    vwretd    ewretd    sprtrn
## 1  12060 19810130  0.000000 -0.040085  0.005615 -0.045742
## 2  12060 19810227  0.089796  0.015521  0.002150  0.013277
## 3  12060 19810331  0.014981  0.046184  0.072674  0.036033
## 4  12060 19810430 -0.020522 -0.011268  0.027885 -0.023456
## 5  12060 19810529  0.001905  0.013551  0.027187 -0.001657
## 6  12060 19810630 -0.046768 -0.010242 -0.013194 -0.010408
```

Part A

Compute the sample mean, standard deviation, skewness, excess kurtosis, minimum, and maximum of each simple return series.

```r
d1a_stats = basicStats(d1)
kable(d1a_stats[c('Mean', 'Stdev', 'Skewness', 'Kurtosis', 'Minimum', 'Maximum'), -(1:2)],
      caption='Basic Statistics of the Simple Return Series')
```

Table 1: Basic Statistics of the Simple Return Series

	ge	vwretd	ewretd	sprtrn
Mean	0.012900	0.009698	0.011022	0.007594
Stdev	0.071073	0.045036	0.053461	0.043921
Skewness	-0.226160	-0.780736	-0.499120	-0.658830
Kurtosis	1.373376	2.526277	3.259182	2.204877
Minimum	-0.272877	-0.225363	-0.272248	-0.217630
Maximum	0.251236	0.128496	0.225012	0.131767

Part B

Transform the simple returns to log returns. Compute the sample mean, standard deviation, skewness, excess kurtosis, minimum, and maximum of each log return series.

```
d1b = log(d1[,-(1:2)]+1)  # Log Transform, +1 as an offset so that we don't compute log(0)
d1b_stats=ts = basicStats(d1b)
kable(d1b_stats[c('Mean', 'Stdev', 'Skewness', 'Kurtosis', 'Minimum', 'Maximum'),],
      caption='Basic Statistics of the Log Transformed Simple Return Series')
```

Table 2: Basic Statistics of the Log Transformed Simple Return Series

	ge	vwretd	ewretd	sprtrn
Mean	0.010307	0.008630	0.009529	0.006594
Stdev	0.071445	0.045617	0.054030	0.044422
Skewness	-0.615128	-1.076565	-0.942621	-0.933307
Kurtosis	2.273898	3.817684	4.784350	3.276645
Minimum	-0.318660	-0.255361	-0.317795	-0.245428
Maximum	0.224132	0.120886	0.202951	0.123780

Part C

Test the null hypothesis that the mean of the log returns of GE stock is zero.

```
t.test(d1b$ge)
```

```
##
##  One Sample t-test
##
## data:  d1b$ge
## t = 2.8708, df = 395, p-value = 0.004315
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.003248467 0.017365132
## sample estimates:
## mean of x
## 0.0103068
```

Reject the null hypothesis that the mean of the log return of GE stock is zero at the 0.05 level, based on p-value.

Part D

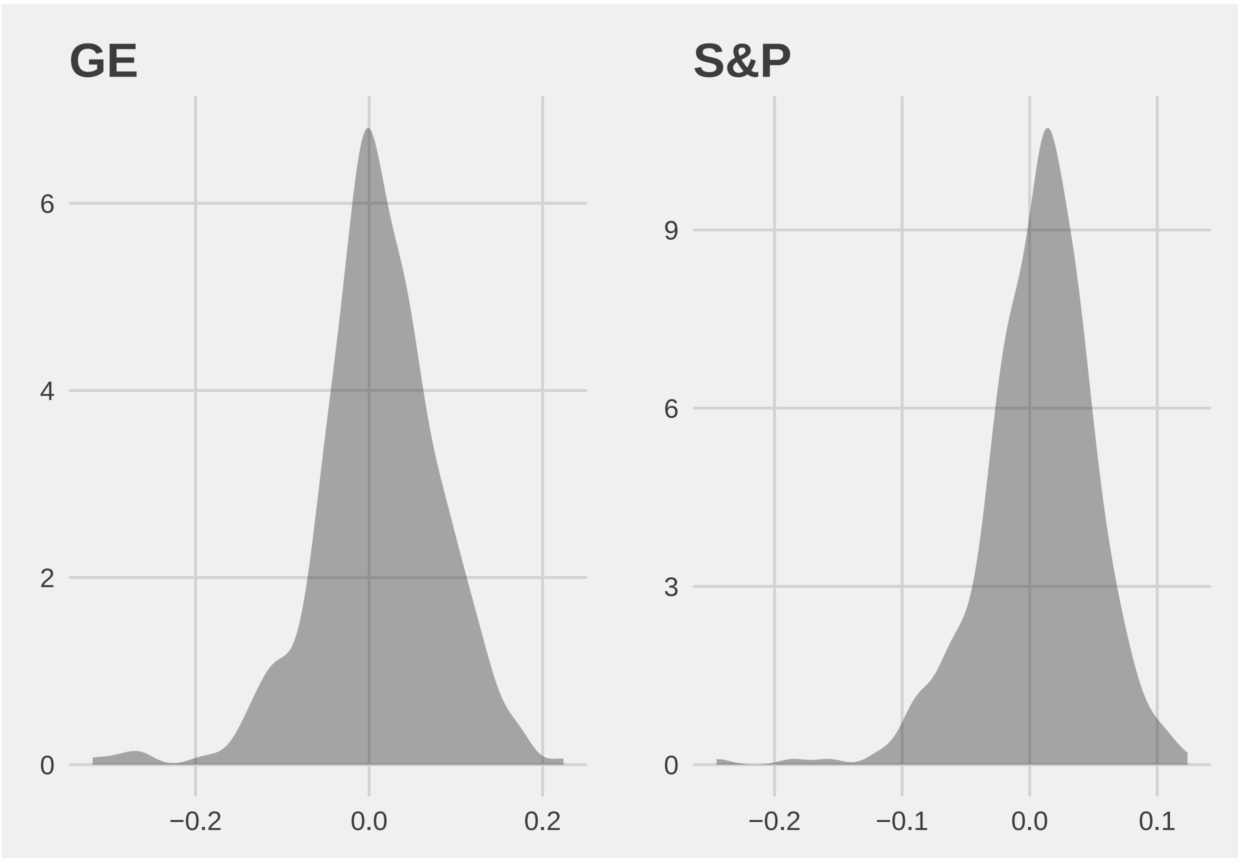
Obtain the empirical density plot of the daily log returns of GE stock and the S&P composite index.

```
pge = ggplot(d1b, aes(ge)) +
  stat_density(alpha = 0.4) +
  labs(x="Returns", y="Density") +
```

```
  ggtitle("GE") + theme_fivethirtyeight()

psprtrn = ggplot(d1b, aes(sprtrn)) +
  stat_density(alpha = 0.4) +
  labs(x="Returns", y="Density") +
  ggtitle("S&P") + theme_fivethirtyeight()

grid.arrange(pge, psprtrn, ncol=2)
```



Part 2

Consider the daily log returns of Netflix stock from January 2, 2009 to December 31, 2013 as in Problem 1, Assignment 1. Perform the following tests:

```
d2 = read.table("data/d-nflx3dx0913.txt", header=T)
head(d2)
```

```
##   PERMNO     date      nflx    vwretd    ewretd    sprtrn
## 1  89393 20090102 -0.000669  0.030501  0.038274  0.031608
## 2  89393 20090105  0.069300 -0.000580  0.016764 -0.004668
## 3  89393 20090106  0.031309  0.011297  0.033647  0.007817
## 4  89393 20090107 -0.006982 -0.030489 -0.022271 -0.030010
## 5  89393 20090108  0.013452  0.006283  0.011896  0.003397
## 6  89393 20090109 -0.026848 -0.022410 -0.018748 -0.021303
```

Part A

Test the null hypothesis that the log return is symmetric with respect to its mean;

```
d2l = log(d2[,-(1:2)] + 1)  # Log Transform, +1 as an offset so that we don't compute log(0)
st = skewness(d2l$nflx) / sqrt(6 / length(d2l$nflx))  # compute skewness test
print(paste("Skewness Statistic: ", st))
```

```
## [1] "Skewness Statistic:  -6.29427781659625"
```

```
p_st = 2 * (1 - pnorm(abs(st)))  # computing the p-value
print(paste("p-value: ", p_st))
```

```
## [1] "p-value:  3.08834291473659e-10"
```

Test $H_0 : M_3 = 0$ versus $H_a : M_3 \neq 0$, where M_3 denotes the skewness of the return. Reject the null hypothesis at a 0.05 level based on p-value.

Part B

Test the null hypothesis that the excess kurtosis of the returns is zero;

Test $H_0 : K = 3$ versus $H_a : K! = 3$, where K denotes the kurtosis.

```
kt = kurtosis(d2l$nflx) / sqrt(24 / length(d2l$nflx))  # compute kurtosis test
print(paste("Kurtosis Statistic: ", kt))
```

```
## [1] "Kurtosis Statistic:  170.499517930473"
```

```
p_kt = 2 * (1 - pnorm(abs(kt)))
print(paste("p-value: ", p_kt))
```

```
## [1] "p-value:  0"
```

Reject null hypothesis at a 0.05 level.

Part C

Construct a 95% confidence interval for the expected daily log return of Netflix stock.

```r
t_test = t.test(d2l$nflx)
print(paste("Confidence Interval, Confidence Level 95%: ", t_test[4]))
```

```
## [1] "Confidence Interval, Confidence Level 95%:  c(-0.000126484354881271, 0.00411859146786966)"
```

Part 3

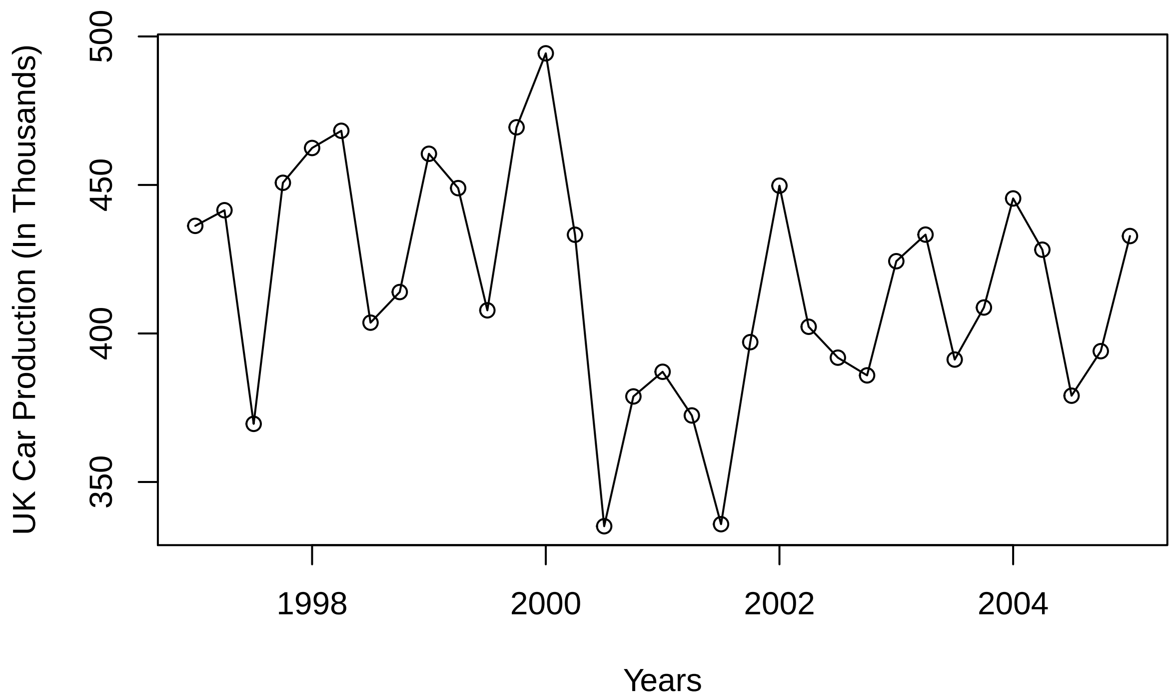
For this exercise, use the quarterly UK passenger vehicle production data from 1997:1 to 2005:1 (data set ukcars) from the Hyndeman text.

```
d3 = window(ukcars, start=1997)
```

Part A

Plot the data and describe the main features of the series.

```
plot(d3, type="o", xlab = "Years", ylab = "UK Car Production (In Thousands)")
```

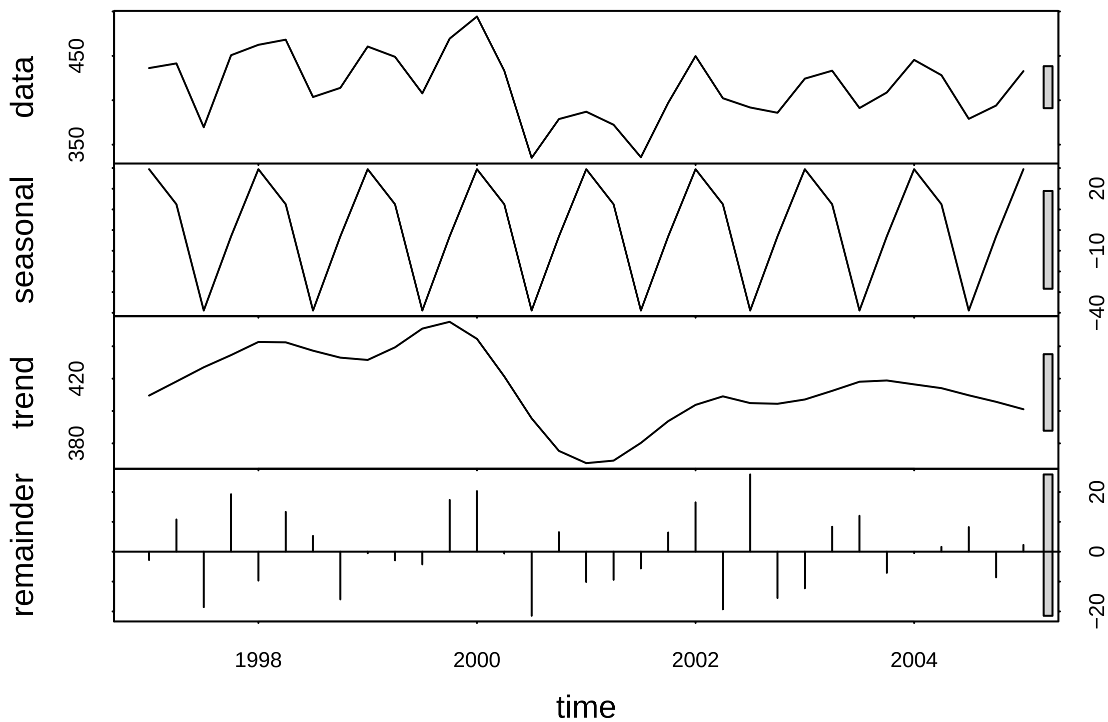


The plot indicates the presence of seasonality. It appears that Q3 would be the yearly low, aside from 2002 where the fourth quarter is lower than the third.

Part B

Decompose the series using STL and obtain the seasonally adjusted data.

```
fit_stl = stl(d3, s.window = "periodic")
plot(fit_stl)
```



The plot indicates that the seasonal fluctuations do not vary with the level of the time series. The smoothed trend plot indicates a gradual up-trend until just before 2000, with a steep down-trend starting around 2000, then a gradual up-trend starting around 2001.

```
seas_adj = seasadj(fit_stl)
seas_factors = fit_stl$time.series[2:11, "seasonal"]  # Acquire the seasonal Factors
```

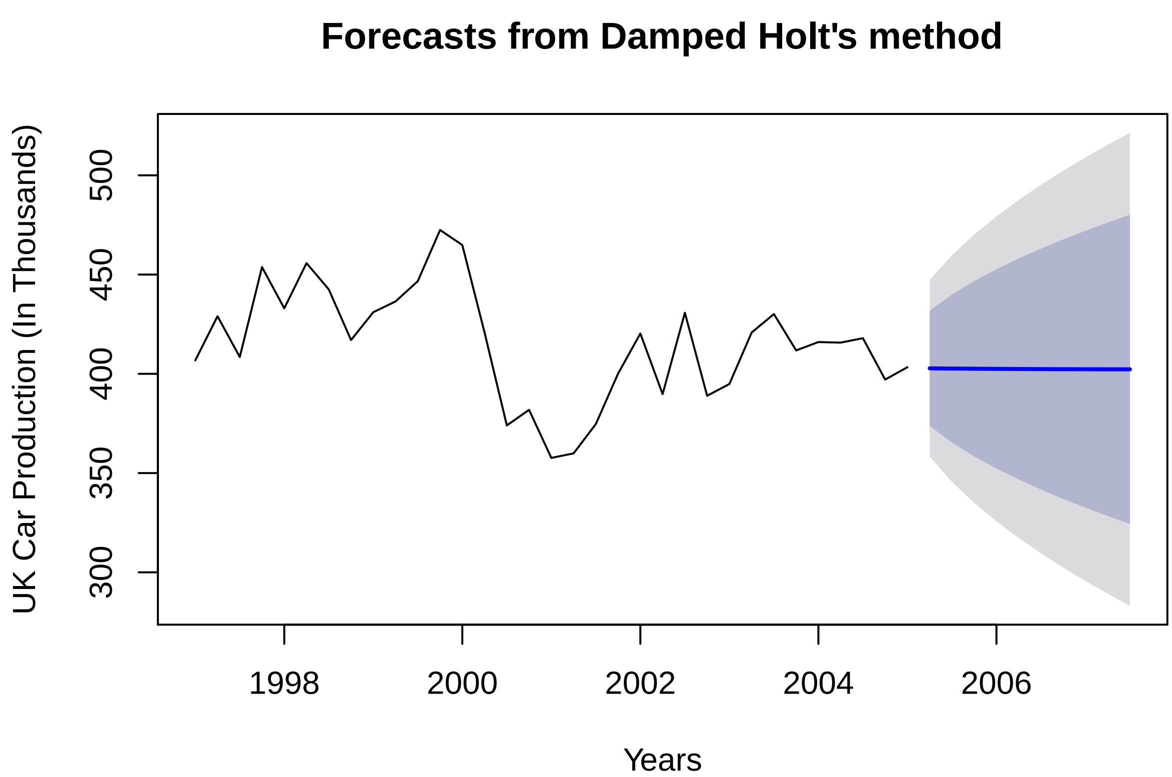
Part C

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.

```r
fit_damped_seas_adj = holt(seas_adj, damped = TRUE)
print(fit_damped_seas_adj)
```

```
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2005 Q2       402.7663 373.6921 431.8404 358.3011 447.2314
## 2005 Q3       402.6509 365.3414 439.9604 345.5910 459.7109
## 2005 Q4       402.5587 358.3842 446.7332 334.9996 470.1177
## 2006 Q1       402.4849 352.2749 452.6948 325.6954 479.2743
## 2006 Q2       402.4258 346.7575 458.0942 317.2884 487.5632
## 2006 Q3       402.3786 341.6862 463.0709 309.5576 495.1995
## 2006 Q4       402.3408 336.9687 467.7129 302.3628 502.3188
## 2007 Q1       402.3105 332.5411 472.0800 295.6074 509.0137
## 2007 Q2       402.2864 328.3574 476.2153 289.2218 515.3509
## 2007 Q3       402.2670 324.3827 480.1514 283.1532 521.3808
```

```r
plot(fit_damped_seas_adj, xlab = "Years", ylab = "UK Car Production (In Thousands)")
```



```r
print(fit_damped_seas_adj$model)
```

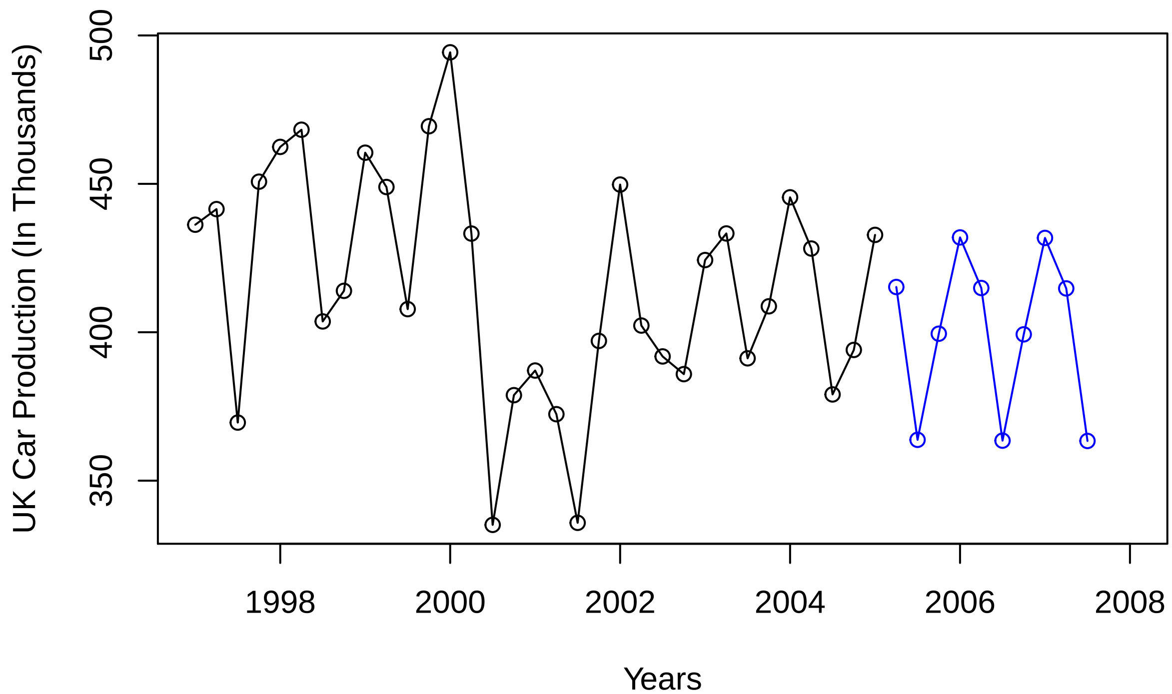
```
## Damped Holt's method
##
## Call:
##  holt(x = seas_adj, damped = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.7781
##     beta  = 0.0181
##     phi   = 0.8
##
##   Initial states:
##     l = 409.7158
##     b = 10.8562
##
##   sigma:  22.6867
##
##      AIC     AICc      BIC
## 331.4222 333.6444 338.9047
```

```r
print(accuracy(fit_damped_seas_adj))
```

```
##                     ME    RMSE      MAE        MPE     MAPE      MASE
## Training set -1.815098 22.6867 17.97465 -0.6075613 4.365406 0.5728646
##                     ACF1
## Training set -0.02372109
```

```r
resea_fit_damed_seas_adj = fit_damped_seas_adj$mean + seas_factors   # reseasonalize the forecasted dat
```

```r
plot(d3, type = "o", xlab = "Years", ylab = "UK Car Production (In Thousands)", xlim = c(1997, 2008))
lines(resea_fit_damed_seas_adj, type = "o", col = "blue")
```


UK Car Production (In Thousands)
500
450
400
350
1998
2000
2002
2004
2006
2008
Years

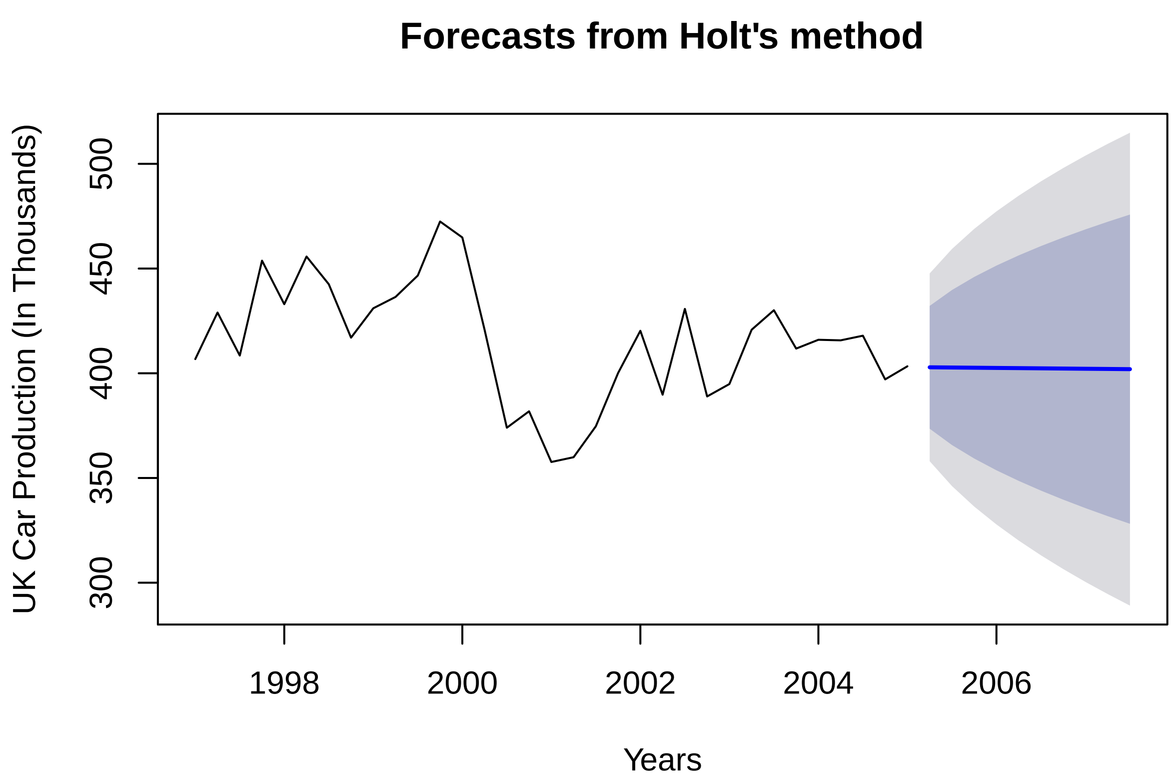
Part D

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.

```r
fit_linear = holt(seas_adj)
print(fit_linear)
```

```
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2005 Q2       402.8474 373.5357 432.1591 358.0190 447.6758
## 2005 Q3       402.7490 365.7538 439.7443 346.1697 459.3283
## 2005 Q4       402.6507 359.3119 445.9895 336.3698 468.9316
## 2006 Q1       402.5523 353.6853 451.4193 327.8167 477.2880
## 2006 Q2       402.4540 348.6223 456.2857 320.1255 484.7825
## 2006 Q3       402.3556 343.9788 460.7324 313.0760 491.6353
## 2006 Q4       402.2573 339.6634 464.8511 306.5283 497.9862
## 2007 Q1       402.1589 335.6138 468.7040 300.3869 503.9309
## 2007 Q2       402.0605 331.7850 472.3361 294.5834 509.5377
## 2007 Q3       401.9622 328.1436 475.7807 289.0665 514.8579
```

```r
plot(fit_linear, xlab = "Years", ylab = "UK Car Production (In Thousands)")
```



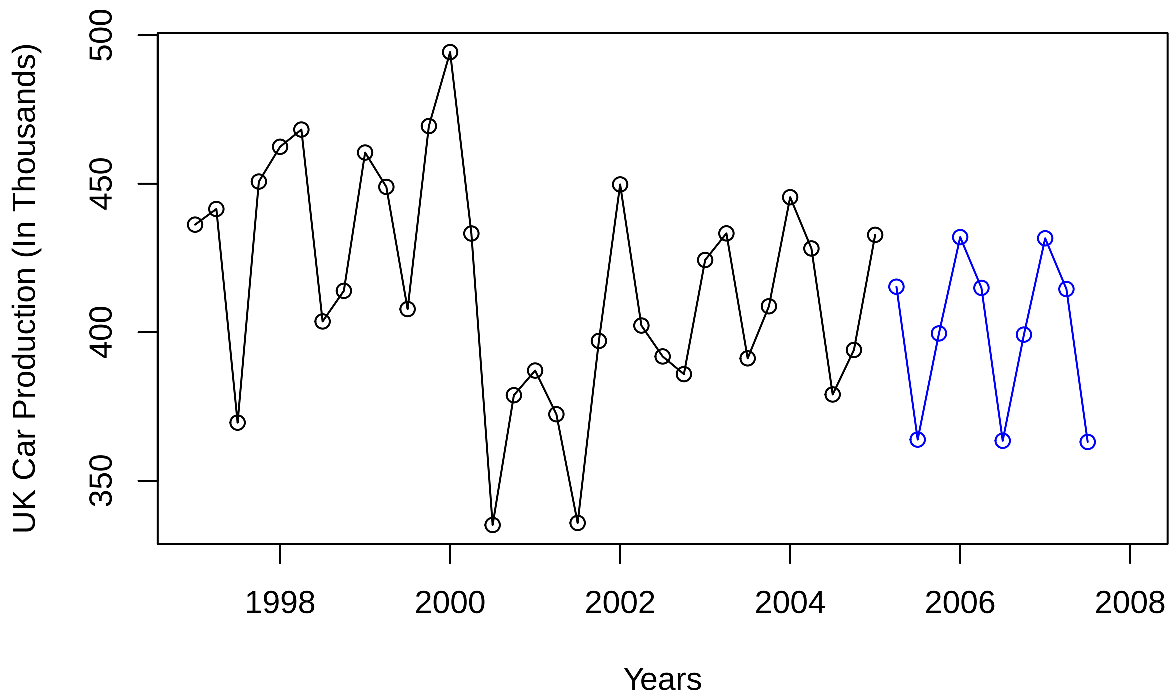
```r
print(fit_linear$model)
```

```
## Holt's method
##
## Call:
##  holt(x = seas_adj)
##
##   Smoothing parameters:
##     alpha = 0.7698
##     beta  = 1e-04
##
##   Initial states:
##     l = 426.7218
##     b = -0.0957
##
##   sigma:  22.872
##
##      AIC     AICc      BIC
## 329.9591 331.3877 335.9452
```

```r
print(accuracy(fit_linear))
```

```
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -0.8100727 22.87203 18.18449 -0.3841676 4.412324 0.5795523
##                      ACF1
## Training set -0.007837827
```

```r
resea_linear = fit_linear$mean + seas_factors
plot(d3, type = "o", xlab = "Years", ylab = "UK Car Production (In Thousands)", xlim = c(1997, 2008))
lines(resea_linear, type = "o", col = "blue")
```



UK Car Production (In Thousands)
500
450
400
350
1998
2000
2002
2004
2006
2008
Years

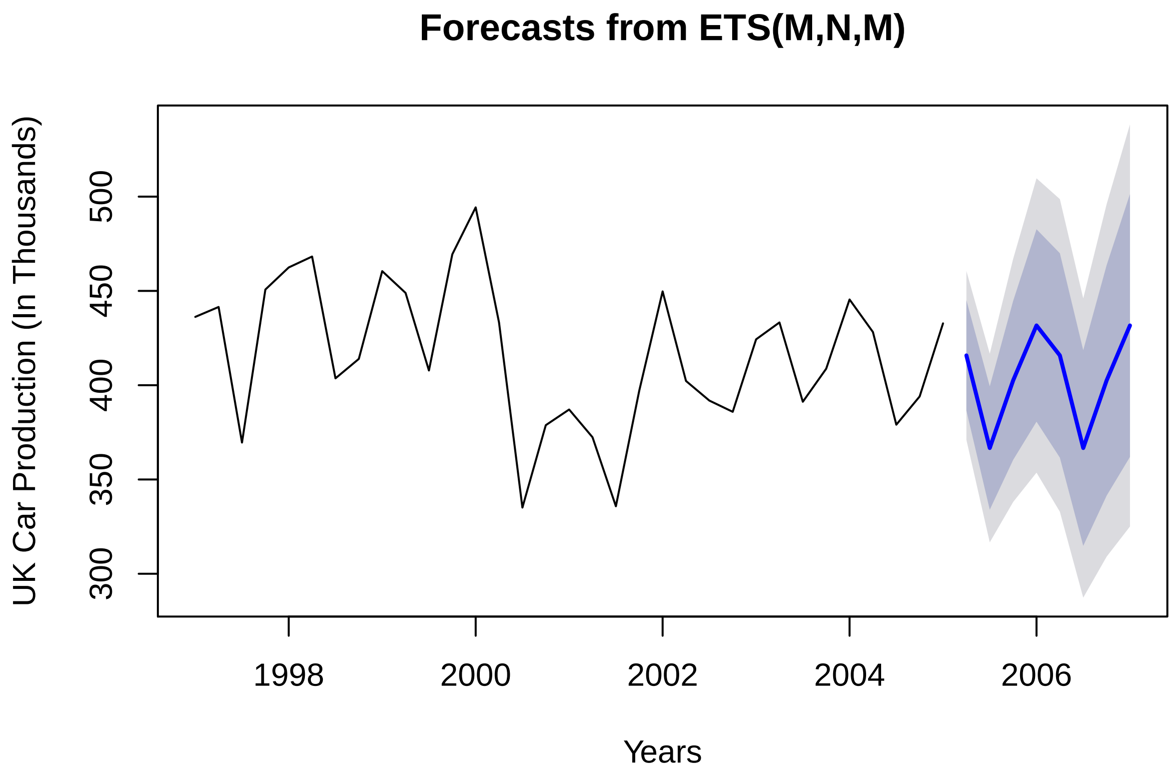
Part E

Now use ETS to choose a seasonal model for the data.

```
fit_ets = ets(d3, model = "ZZZ")
print(fit_ets)
```

```
## ETS(M,N,M)
##
## Call:
##  ets(y = d3, model = "ZZZ")
##
##   Smoothing parameters:
##     alpha = 0.777
##     gamma = 1e-04
##
##   Initial states:
##     l = 423.437
##     s=0.9959 0.9073 1.0288 1.0681
##
##   sigma:  0.0549
##
##      AIC     AICc      BIC
## 333.7918 337.0225 342.7708
```

```
plot(forecast(fit_ets), xlab = "Years", ylab = "UK Car Production (In Thousands)")
```




```r
print(accuracy(fit_ets))
```

```
##                      ME     RMSE     MAE        MPE     MAPE      MASE
## Training set -0.7222169 22.61134 17.9612 -0.3731362 4.396839 0.5724358
##                     ACF1
## Training set -0.03674353
```

Part 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?

```r
print(paste("Additive-Damped Model RMSE: ", accuracy(fit_damped_seas_adj)[2]))
```

```
## [1] "Additive-Damped Model RMSE:  22.6867027922175"
```

```r
print(paste("Holt Model RMSE: ", accuracy(fit_linear)[2]))
```

```
## [1] "Holt Model RMSE:  22.8720348802071"
```

```r
print(paste("ETS Model RMSE: ", accuracy(fit_ets)[2]))
```

```
## [1] "ETS Model RMSE:  22.6113445439822"
```

The ETS model had the lowest RMSE.

Part G

Compare the forecasts from the two approaches? Which seems most reasonable?

The ETS model seems to be the most reasonable, showing a continuation of the most recent observed trends. We can see, if we compare side by side that both Holt Models show abrupt discontinuation of the recent observed trends.

```r
par(mfrow = c(3,1))
p_damped = plot(forecast(fit_damped_seas_adj), type = "o", xlab = "Years", ylab = "Production")
p_linear = plot(forecast(fit_linear), type = "o", xlab = "Years", ylab = "Production")
p_ets = plot(forecast(fit_ets), type = "o", xlab = "Years", ylab = "Production")
```