FORECASTING PAPER

US Ecommerce Retail Sales

ECON 6210

P. Sadorsky

Author: Xiao Dian Zhu

212873303

Introduction

E-commerce retail sales are sales of goods and services where orders are placed and prices are negotiated over electronic networks, primarily the internet and now mobile device. I became interested in this topic as e-commerce has been changing the way people shop in the last decade with the rise of internet. In major consumer markets such as China and USA, e-commerce has taken over consumer transactions. For instance, this year Alibaba's Singles' Day smashes record with $25B of sales. In USA, share of e-commerce sales in total retail sales accounts for 9.1% in the third quarter of 2017. Amazon is the biggest e-commerce retailer; 70% of revenues comes from online sales. I decided to use this project as an opportunity to study the trend of e-commerce retail sales in US and make forecasts.

Descriptive Data

The graph below shows the quarterly US ecommerce retails sales in millions of dollars from 2000 Q1 to 2017 Q2. Data was not seasonally adjusted and was obtained from FRED. We can see an apparent and consistent uptrend from 2000 to 2017, as well as seasonality. There was a small decrease around 2008 and 2009, possibly due to 2008 Financial Crisis; afterwards sales increased in an accelerated pace.

```
usesales <- read.csv("C:/Schulich/econ6120/ecom.csv")

sales <- ts(usesales,start=2000,frequency=4)

esales <- sales[,"ECOMNSA"]


plot(esales, ylab="sales in millions of USD", main="US Ecommerce Retail Quarterly S
ales", col='blue', lwd=2)
```

We can also see on the seasonal plot that sales have been increasing from year to year. The seasonal plot demonstrates that in the past, for most of the times sales tended to increase going from the 1st quarter to the 2nd, decrease slightly or stays the same in the 3rd quarter, and then jump significantly in the 4th quarter; in the recent years, the jump in Q4 seems more significant than in the past, pushing the sales higher. The monthly plot tells a similar story. The highest sales of the year occurred in the 4th season.

```
seasonplot(esales,ylab="sales in millions of USD", main="US Ecommcerce Retail Quart
erly Sales Seasonal Plot", year.labels.left=TRUE, col=1:16)
```


```
monthplot(esales,ylab="sales in millions of USD", main="US Ecommcerce Retail Quarte
rly Sales Monthly Plot")
```

Summary Statistics

Looking at the summary statistics, we see that the sales data has a wide range from the minimum of $5562M to a maximum of $122515M, with an interquartile range of $42900M. The mean ($42272M) is larger than the median ($34380M), suggesting a non-normal and right-skewed distribution. This distribution can be seen on the histogram and the boxplot. There is no outlier.

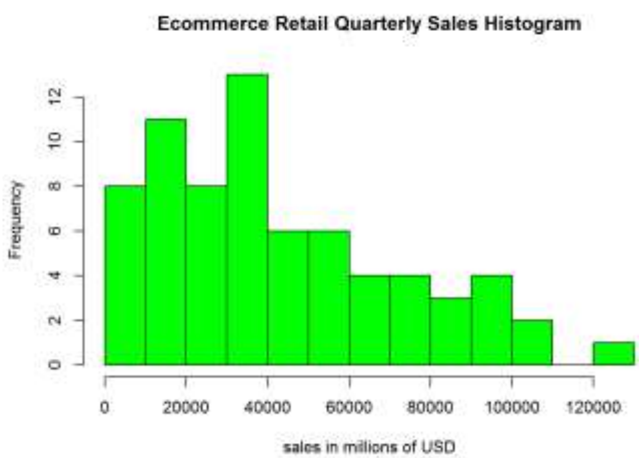
```
summary(esales)
```

```
##     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5562   17511   34380   42272   60411  122515
```

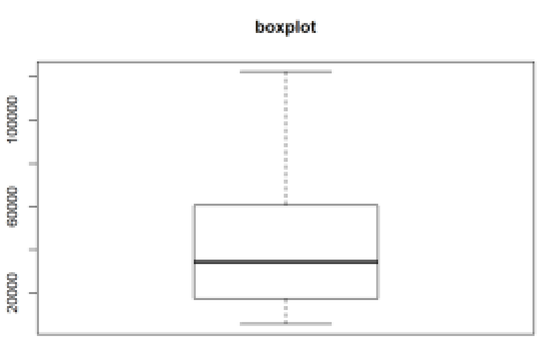
```
sd(esales)
```

```
## [1] 29398.49
```

```
hist(esales,breaks=10,xlab="sales in millions of USD", main="Ecommerce Retail Quarterly Sales Histogram", col="green")
```



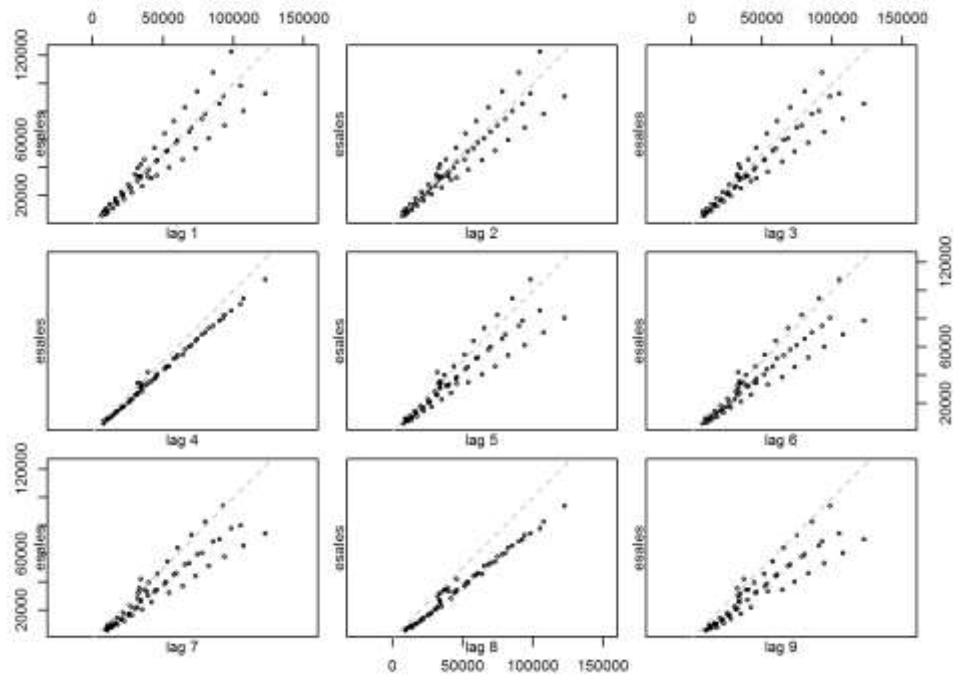

```
boxplot(esales, main="boxplot")
```



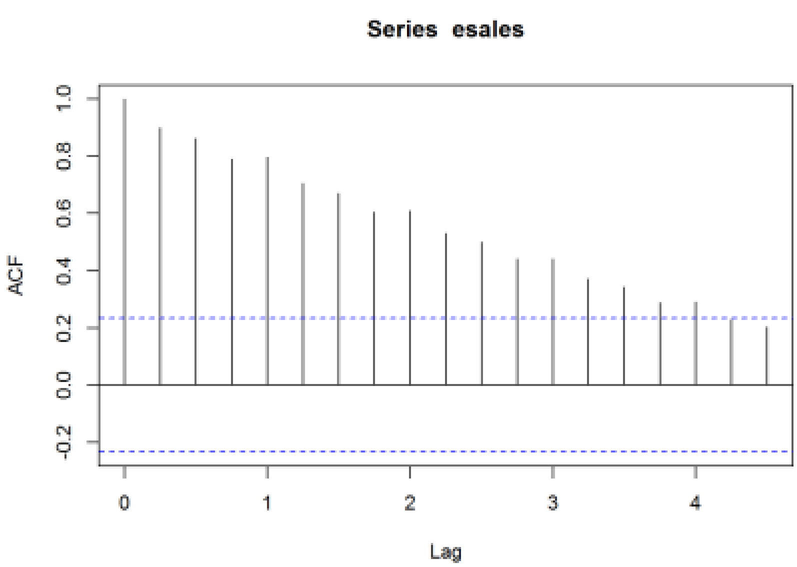
Autocorrelation Test

The lagged plots exhibit strong positive linear patterns. The correlogram shows positive autocorrelations, starting with a very high autocorrelation and then slowly declines. This suggests that there has been an obvious trend in ecommerce retail sales in the past 17 years, which provides high predictability if modelled correctly.

```
lag.plot(esales,lag=9,do.lines=FALSE)
```




```
acf(esales)
```



Sales Growth Rate

To confirm if seasonal patterns are consistent and strong, I took natural logs to compute the percentage sales growth. Cumulatively, sales growth rate has been increasing in the past 17 years. The growth rate is strongly affected by seasonality; as demonstrated by the graph on the right, the growth varies hugely throughout the year from approximately high 30% to low 20% going to a different quarter, and thus seasonality is consistent and strong. The minimum growth rate (-23.51%) occurred in the first quarter of 2013; the maximum (33.37%) was in the last quarter of 2011. The range is very wide. The histogram and the boxplot below show a roughly symmetrical distribution. The boxplot shows no outlier.

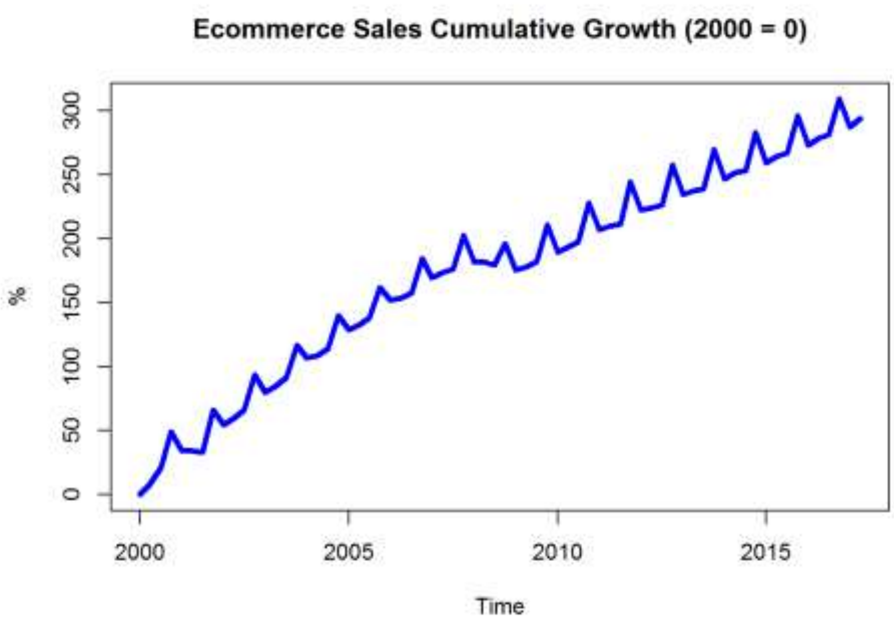
```
ly = log(esales)
ly.base = 100 * (ly - ly[1])
head(ly.base)
```

```
##           Qtr1      Qtr2      Qtr3      Qtr4
## 2000  0.000000  8.723609 21.527376 49.000630
## 2001 34.900446 34.021516
```

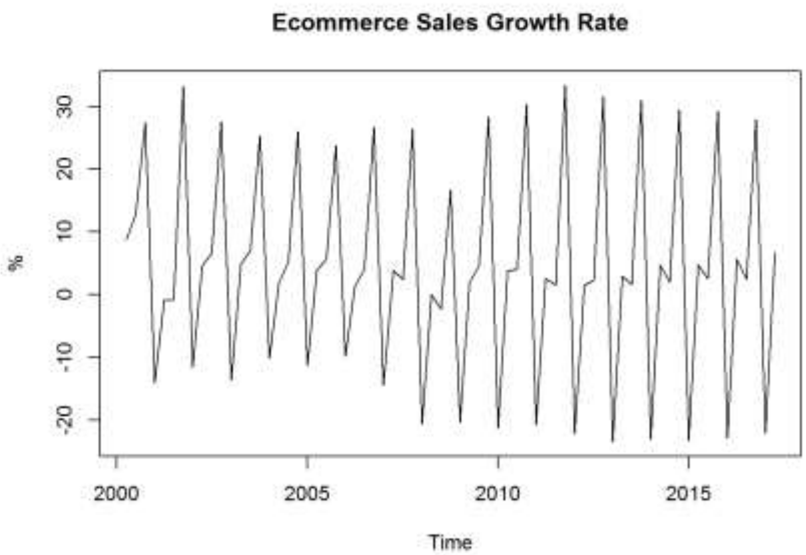
```
tail(ly.base)
```

```
##          Qtr1     Qtr2     Qtr3     Qtr4
## 2016 273.1751 278.8253 281.2806 309.2276
## 2017 287.1985 293.8916
```

```
plot(ly.base, main="Ecommerce Sales Cumulative Growth (2000 = 0)", ylab="%",col = '
blue', lwd=4)
```

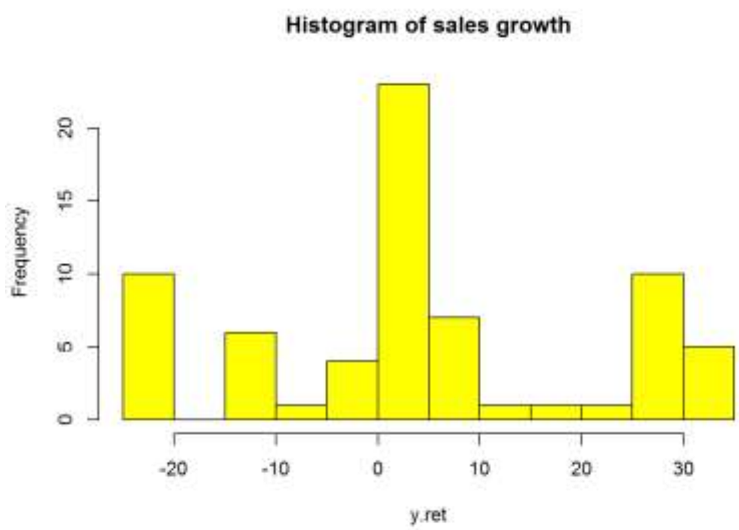


```r
# generate and plot sales growth
y.ret = diff(log(esales)) * 100
plot(y.ret, main="Ecommerce Sales Growth Rate", ylab="%")
```

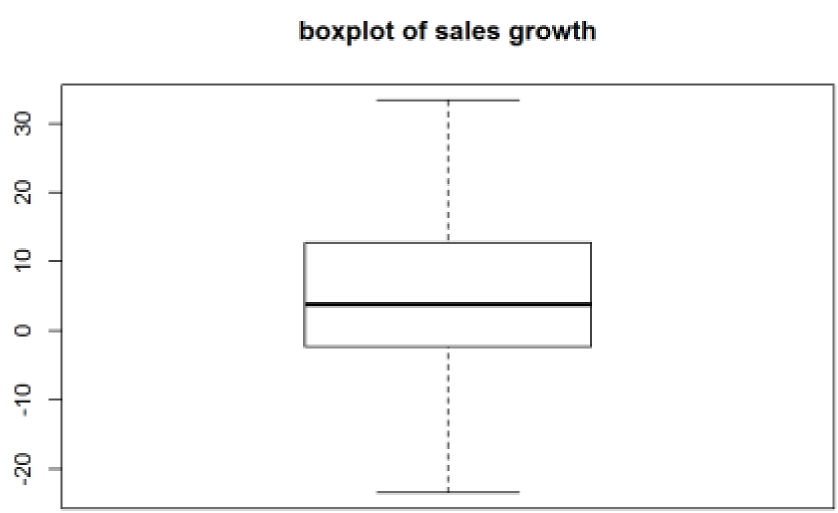



```r
# find the minimum growth rate
which.min(y.ret)
## [1] 52
y.ret[which.min(y.ret)]
## [1] -23.51092
usesales[which.min(y.ret)+1,]
##          DATE ECOMNSA      GDP     PCEC      DPI        AMZN
## 53 01/01/2013   57952 16475.44 11256.66 12259.29  16,070.00
# find the maximum growth rate
which.max(y.ret)
## [1] 47
y.ret[which.max(y.ret)]
## [1] 33.37138
usesales[which.max(y.ret)+1,]
##          DATE ECOMNSA      GDP     PCEC      DPI        AMZN
## 48 01/10/2011   64224 15785.31 10827.85 11924.91  17,431.00
# compute summary statistics and autocorrelation test for growth rate
```

```
summary(y.ret)
##     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -23.511  -2.291   3.777   4.259  12.804  33.371
sd(y.ret)
## [1] 16.65659
hist(y.ret, breaks=10,main="Histogram of sales growth",col="yellow")
```

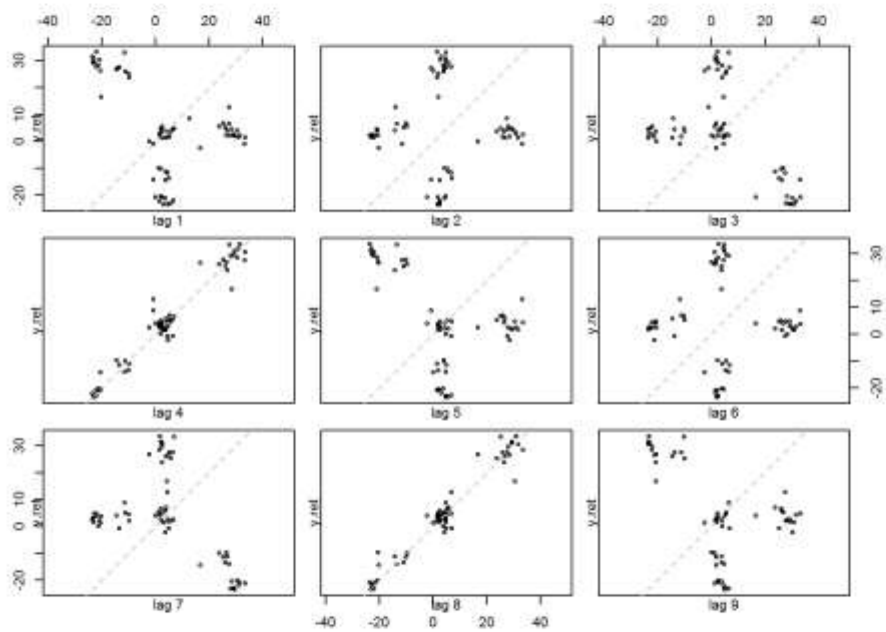



```
boxplot(y.ret, main="boxplot of sales growth")
```

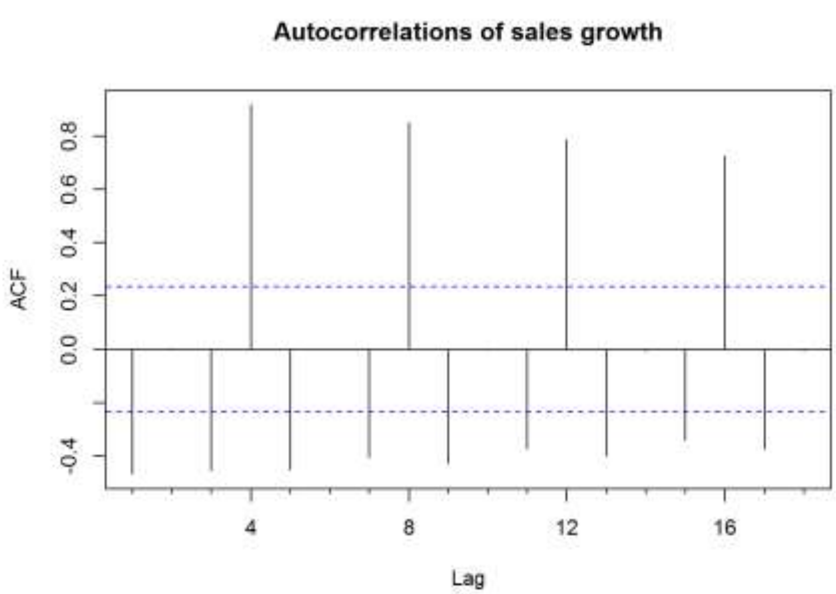


The lagged plots exhibit four types of patterns (shown by lag 1 to lag 4) corresponding to the four quarters. The correlogram, being statistically significant, shows a pattern of having one negative autocorrelation coefficient followed by a coefficient close to zero, and then another slightly smaller negative coefficient followed by a very positive spike. The peaks are four quarters apart. This indicates strong and consistent seasonal patterns.

```
lag.plot(y.ret,lag=9,do.lines=FALSE)
```




```
Acf(y.ret,main="Autocorrelations of sales growth")
```



Univariate Analysis

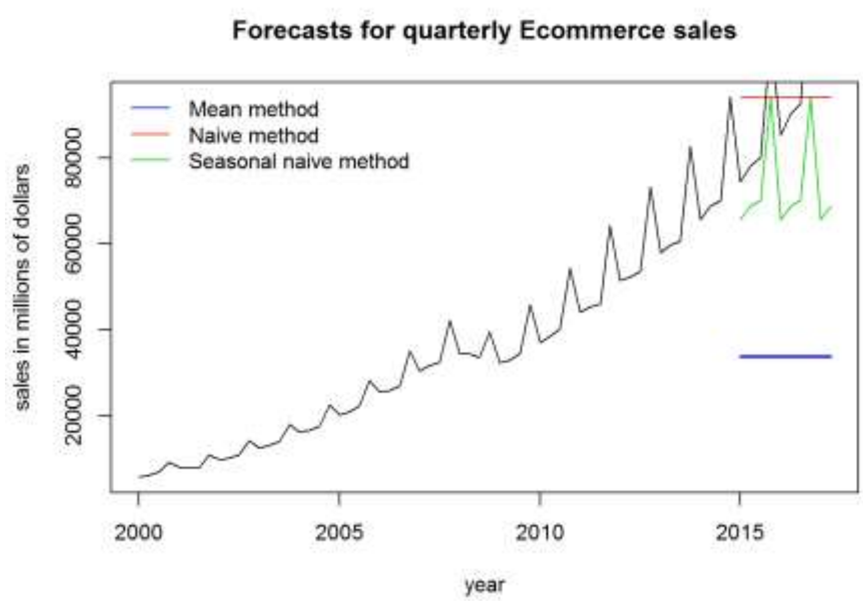
I set 80% of data as training data (2000 to 2014) and 20% as testing data (2015 to Q2 2017).

```
train <- window(esales,start=c(2000, 1),end=c(2014, 4))
test <- window(esales, start=c(2015,1),end=c(2017,2))
h=length(test)
```

Method 1 to 3

I used the three simple forecasting methods first. Looking at the graph below, the three forecasts do not fit well with the actual values during the test period. The Naive method captures the uptrend of the data best; however, it does not show the seasonal patterns. While the Seasonal Naive method captures only the seasonality, the Mean method deviates greatly from the actual values, missing both the uptrend and the seasonal patterns.

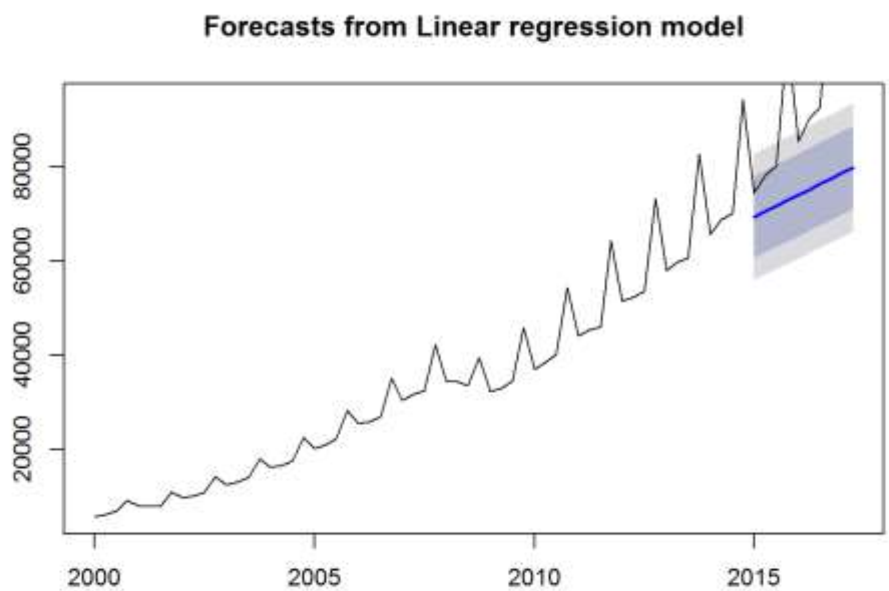
```
fit1 <- meanf(train, h=h)
fit2 <- naive(train, h=h)
fit3 <- snaive(train, h=h)
plot(fit1, PI=FALSE,ylab="sales in millions of dollars",xlab="year",
     main="Forecasts for quarterly Ecommerce sales")
lines(fit2$mean,col=2)
lines(fit3$mean,col=3)
lines(esales) #actual
legend("topleft",lty=1,col=c(4,2,3),
       legend=c("Mean method","Naive method","Seasonal naive method"),bty="n")
```



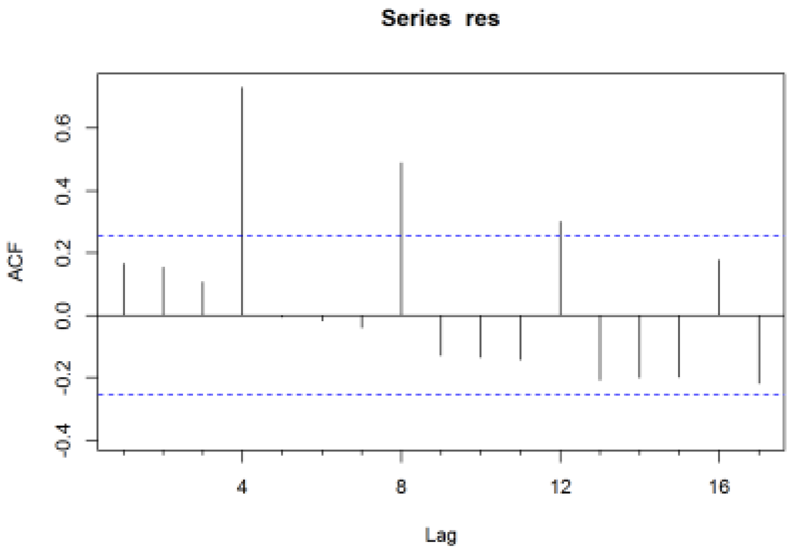
Method 4 and 5

Then I used the linear trend method. Although the linear trend method captures the uptrend, it misses seasonality. So I used linear trend + seasonal to ensure seasonality is captured. However, it is not steep enough to fit the actual values. Looking as the residuals of the linear trend method, they get large towards 2015 and the ACF shows significant spikes at lag 4, 8 and 12. The residuals of the linear trend +seasonal method look slightly better, whereas there are still a few spikes at beginning and lag=8. The estimated linear models violate the assumption of no autocorrelation in the errors, implying there is some information left over which should be utilized in order to obtain better forecasts. Thus, we can be sure that the both models are not a good for forecasting ecommerce sales.

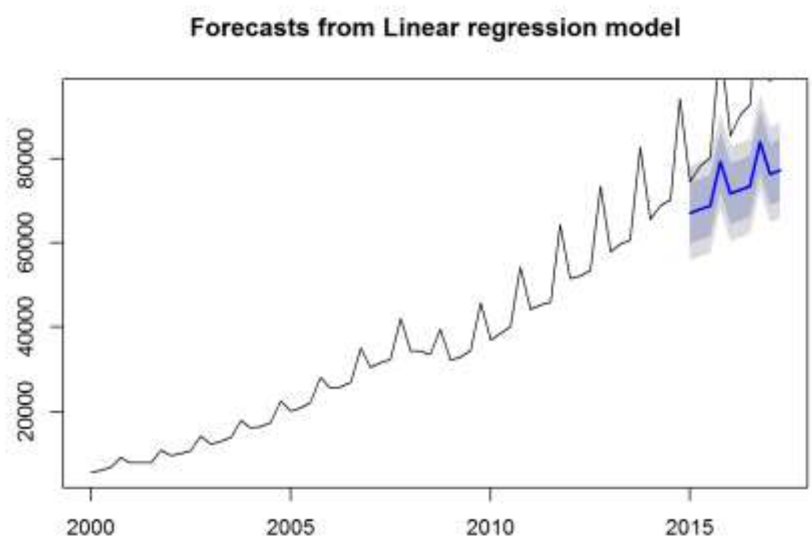
```
#Linear trend
y.lt <- tslm(train ~ trend)
fit4 <- forecast(y.lt, h=h,level=c(80,95))
plot(fit4)
lines(esales)
```



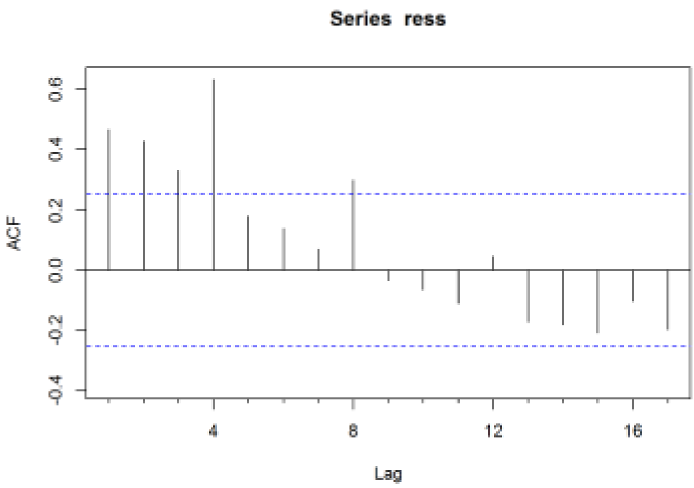

```
res=resid(y.lt)
Acf(res)
```




```
#Linear trend plus seasonal
y.lts <- tslm(train ~ trend + season)
fit5 <- forecast(y.lts, h=h,level=c(80,95))
plot(fit5)
lines(esales)
```




```
ress=resid(y.lts)
Acf(ress)
```



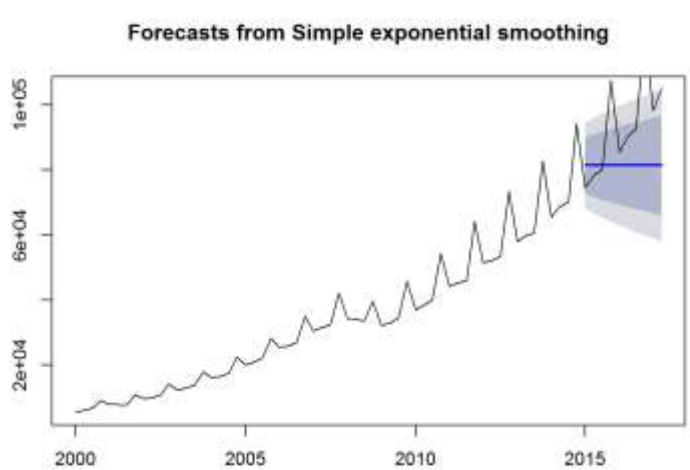
Method 6

The next method is exponential smoothing. Visually we can that the forecast fits the actual data poorly as it does not capture the seasonality and the uptrend. The confidence bands are wide. The ACF of residuals shows significant positive spikes, and thus there is information not captured by this model.

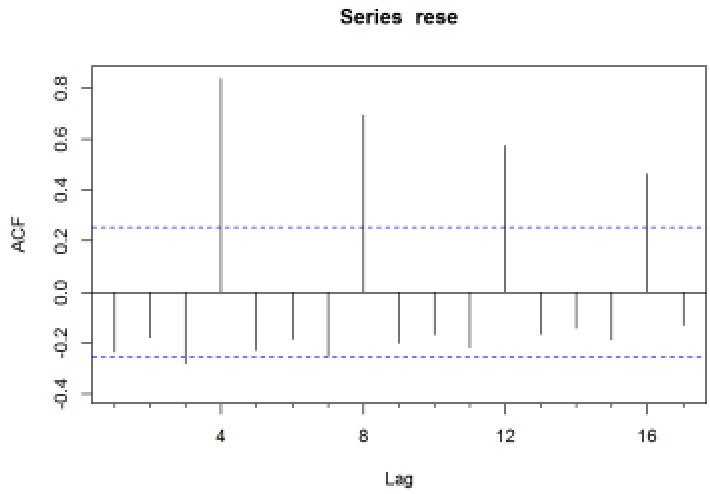
```
fit6 <- ses(train, h = h)
summary(fit6)
##
## Forecast method: Simple exponential smoothing
##
## Model Information:
## Simple exponential smoothing
##
## Call:
##  ses(y = train, h = h)
##
##   Smoothing parameters:
##     alpha = 0.4845
##
##   Initial states:
##     l = 6280.9254
##
##   sigma:  6737.88
##
```

```
##       AIC     AICc      BIC
## 1309.521 1309.949 1315.804
##
## Error measures:
##                     ME    RMSE      MAE      MPE    MAPE      MASE
## Training set 2583.205 6737.88 3678.506 6.983645 9.90216 0.7283603
##                     ACF1
## Training set -0.2338662
##
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95     Hi 95
## 2015 Q1        81375.8 72740.85 90010.74 68169.79  94581.80
## 2015 Q2        81375.8 71780.72 90970.87 66701.39  96050.20
## 2015 Q3        81375.8 70908.29 91843.31 65367.12  97384.47
## 2015 Q4        81375.8 70103.17 92648.42 64135.81  98615.78
## 2016 Q1        81375.8 69351.85 93399.74 62986.76  99764.83
## 2016 Q2        81375.8 68644.79 94106.80 61905.40 100846.19
## 2016 Q3        81375.8 67974.98 94776.61 60881.02 101870.57
## 2016 Q4        81375.8 67337.10 95414.50 59905.46 102846.13
## 2017 Q1        81375.8 66726.96 96024.63 58972.34 103779.26
## 2017 Q2        81375.8 66141.24 96610.35 58076.56 104675.04
```

```
plot(fit6)
lines(esales)
```



```
rese=resid(fit6)
Acf(rese)
```



Method 7

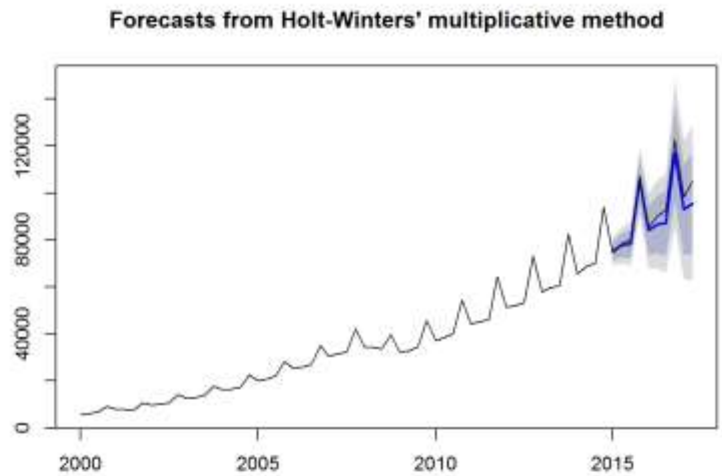
Holt Winter's Multiplicative method seems to be the best method so far, as on the graph the forecast almost fits the actual line with uptrend and seasonality. The residuals (except the first one) are all within the confidence bands, implying no autocorrelations in the errors and the model captures the data very well.

```
fit7 <- hw(train, seasonal="multiplicative", h=h)
summary(fit7)
```

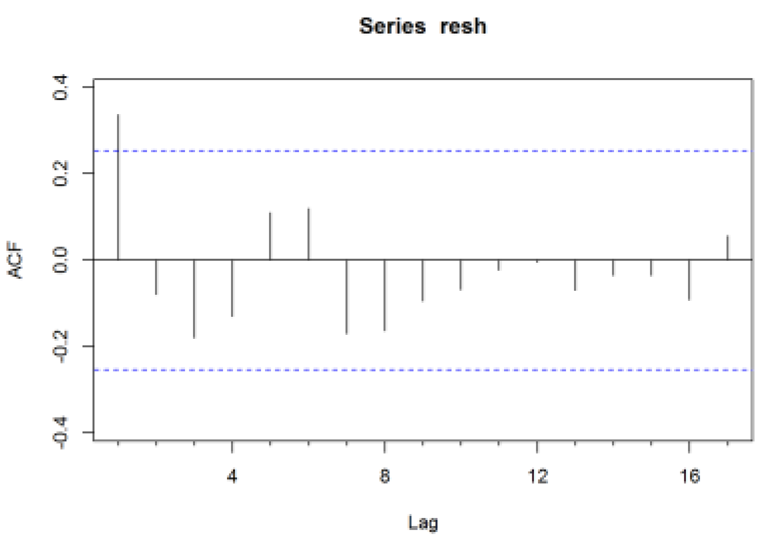
```
##
## Forecast method: Holt-Winters' multiplicative method
##
## Model Information:
## Holt-Winters' multiplicative method
##
## Call:
##  hw(y = train, h = h, seasonal = "multiplicative")
##
##   Smoothing parameters:
##     alpha = 0.5239
##     beta  = 0.1728
##     gamma = 0.4761
##
```

```
##   Initial states:
##     l = 5558.4439
##     b = 748.8957
##     s=1.1664 0.9194 0.9189 0.9954
##
##   sigma:  0.0416
##
##      AIC     AICc      BIC
## 1104.798 1108.398 1123.648
##
## Error measures:
##                    ME     RMSE      MAE      MPE     MAPE      MASE
## Training set 150.5839 1271.044 846.2931 0.175977 3.048849 0.1675698
##                   ACF1
## Training set 0.3221062
##
## Forecasts:
##         Point Forecast    Lo 80     Hi 80    Lo 95     Hi 95
## 2015 Q1       75337.34 71321.18  79353.49 69195.15  81479.52
## 2015 Q2       77835.01 72823.11  82846.90 70169.98  85500.03
## 2015 Q3       78413.79 72304.23  84523.35 69070.03  87757.55
## 2015 Q4      105470.38 95634.73 115306.02 90428.06 120512.70
## 2016 Q1       84188.36 73644.47  94732.25 68062.87 100313.85
## 2016 Q2       86722.91 74481.92  98963.90 68001.93 105443.90
## 2016 Q3       87123.55 73335.83 100911.26 66037.06 108210.04
## 2016 Q4      116874.51 96265.39 137483.63 85355.58 148393.44
## 2017 Q1       93091.13 73797.04 112385.22 63583.37 122598.89
## 2017 Q2       95664.02 74059.48 117268.57 62622.73 128705.32
```

```
plot(fit7)
lines(esales)
```




```
resh=resid(fit7)
Acf(resh)
```



Method 8 to 10

I also used other Holt's methods below. We can tell visually that they do not fit the actual data as good as the multiplicative method, since they failed to capture seasonality.

```
# holt's linear trend method
fit8 <- holt(train,  h=h)
summary(fit8)
```

```
##
```

```
## Forecast method: Holt's method
##
## Model Information:
## Holt's method
##
## Call:
##  holt(y = train, h = h)
##
##   Smoothing parameters:
##     alpha = 0.1712
##     beta  = 0.0565
##
##   Initial states:
##     l = 5457.1112
##     b = 696.4417
##
##   sigma:  5718.061
##
##      AIC     AICc      BIC
## 1293.827 1294.938 1304.299
##
## Error measures:
##                    ME     RMSE      MAE         MPE     MAPE      MASE
## Training set 652.3981 5718.061 3526.302 -0.03127299 9.402263 0.6982233
##                    ACF1
## Training set -0.1293338
##
## Forecasts:
##         Point Forecast    Lo 80     Hi 80    Lo 95     Hi 95
## 2015 Q1       82095.11 74767.12  89423.10 70887.92  93302.30
## 2015 Q2       85003.73 77488.18  92519.27 73509.69  96497.76
## 2015 Q3       87912.34 80113.56  95711.12 75985.14  99839.54
## 2015 Q4       90820.96 82632.25  99009.67 78297.41 103344.51
## 2016 Q1       93729.57 85038.86 102420.29 80438.27 107020.87
```

```
## 2016 Q2       96638.19 87333.11 105943.27 82407.29 110869.09
## 2016 Q3       99546.80 89518.53 109575.07 84209.89 114883.72
## 2016 Q4      102455.42 91601.10 113309.75 85855.16 119055.68
## 2017 Q1      105364.04 93587.87 117140.21 87353.94 123374.13
## 2017 Q2      108272.65 95486.16 121059.15 88717.39 127827.91
```

```r
# holt's exponential trend method
fit9 <- holt(train, exponential=TRUE, h=h)
summary(fit9)
```

```
##
## Forecast method: Holt's method with exponential trend
##
## Model Information:
## Holt's method with exponential trend
##
## Call:
##  holt(y = train, h = h, exponential = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.1893
##     beta  = 0.0516
##
##   Initial states:
##     l = 6043.1754
##     b = 1.0258
##
##   sigma:  0.1414
##
##      AIC     AICc      BIC
## 1244.358 1245.469 1254.830
##
## Error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -145.6273 5779.713 3923.074 -1.214496 10.47137 0.7767858
##                    ACF1
```

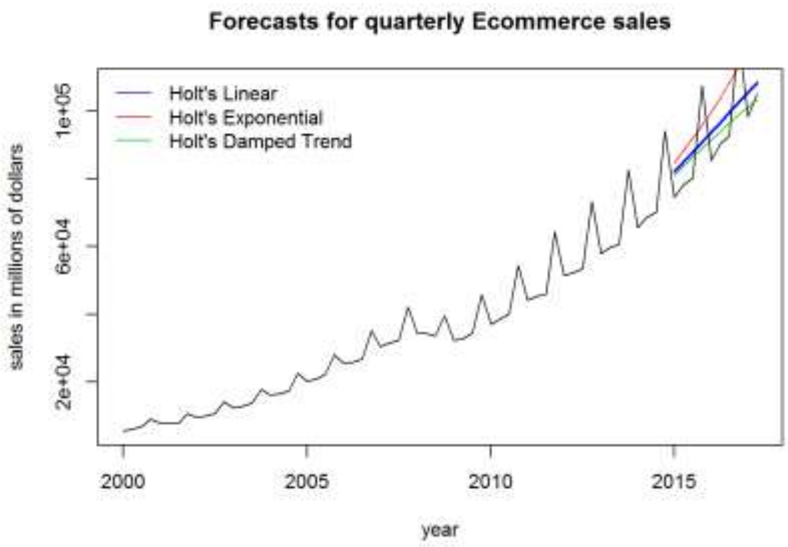
```
## Training set -0.1006896
##
## Forecasts:
##         Point Forecast    Lo 80     Hi 80    Lo 95    Hi 95
## 2015 Q1       84639.54 69383.64  99995.72 61251.15 108268.6
## 2015 Q2       88191.78 72315.99 105100.71 63420.86 113954.2
## 2015 Q3       91893.10 74218.40 109431.05 65524.68 119892.8
## 2015 Q4       95749.76 76062.16 115165.10 66007.12 126854.5
## 2016 Q1       99768.28 79055.85 121668.45 68598.47 134420.4
## 2016 Q2      103955.46 81634.29 129722.80 70245.05 145336.0
## 2016 Q3      108318.37 83576.23 136982.32 71873.03 155167.1
## 2016 Q4      112864.38 85000.00 146132.99 73393.29 165816.2
## 2017 Q1      117601.19 86919.68 154838.33 73056.10 180525.6
## 2017 Q2      122536.80 88076.81 165170.22 73172.94 192940.4
```

```r
# holt's damped trend method
fit10 <- holt(train, damped=TRUE, h=h)
summary(fit10)
```

```
##
## Forecast method: Damped Holt's method
##
## Model Information:
## Damped Holt's method
##
## Call:
##  holt(y = train, h = h, damped = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.1475
##     beta  = 0.0708
##     phi   = 0.98
##
##   Initial states:
##     l = 5456.6887
##     b = 779.4827
```

```
##
##   sigma:  5735.24
##
##      AIC     AICc      BIC
## 1296.187 1297.772 1308.753
##
## Error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 792.3264 5735.24 3571.035 0.5438349 9.533172 0.7070806
##                    ACF1
## Training set -0.124346
##
## Forecasts:
##         Point Forecast    Lo 80     Hi 80    Lo 95     Hi 95
## 2015 Q1       81000.20 73650.19  88350.20 69759.33  92241.06
## 2015 Q2       83693.63 76172.65  91214.61 72191.29  95195.98
## 2015 Q3       86333.20 78526.00  94140.40 74393.12  98273.28
## 2015 Q4       88919.97 80696.07  97143.87 76342.61 101497.33
## 2016 Q1       91455.01 82678.59 100231.43 78032.63 104877.39
## 2016 Q2       93939.34 84477.58 103401.10 79468.83 108409.86
## 2016 Q3       96373.99 86102.77 106645.22 80665.51 112082.48
## 2016 Q4       98759.95 87566.72 109953.18 81641.38 115878.52
## 2017 Q1      101098.19 88882.79 113313.58 82416.35 119780.02
## 2017 Q2      103389.66 90063.80 116715.52 83009.52 123769.80
```

```
plot(fit8, PI=FALSE,ylab="sales in millions of dollars",xlab="year",
     main="Forecasts for quarterly Ecommerce sales")
lines(fit9$mean,col=2)
lines(fit10$mean,col=3)
lines(esales) #actual
legend("topleft",lty=1,col=c(4,2,3),
       legend=c("Holt's Linear","Holt's Exponential","Holt's Damped Trend"),bty="n
")
```



Method 11

ETS is a good method, as on the graph the forecast almost perfectly fits the actual line and most of the residuals are not autocorrelated as they are within the confidence bands.

```
y.ets <- ets(train, model="ZZZ")
summary(y.ets)
```

```
## ETS(M,A,M)
##
## Call:
##  ets(y = train, model = "ZZZ")
##
##   Smoothing parameters:
##     alpha = 0.8555
##     beta  = 0.1601
##     gamma = 4e-04
##
##   Initial states:
##     l = 5561.9563
##     b = 609.276
##     s=1.1798 0.931 0.9382 0.951
##
##   sigma:  0.0365
```

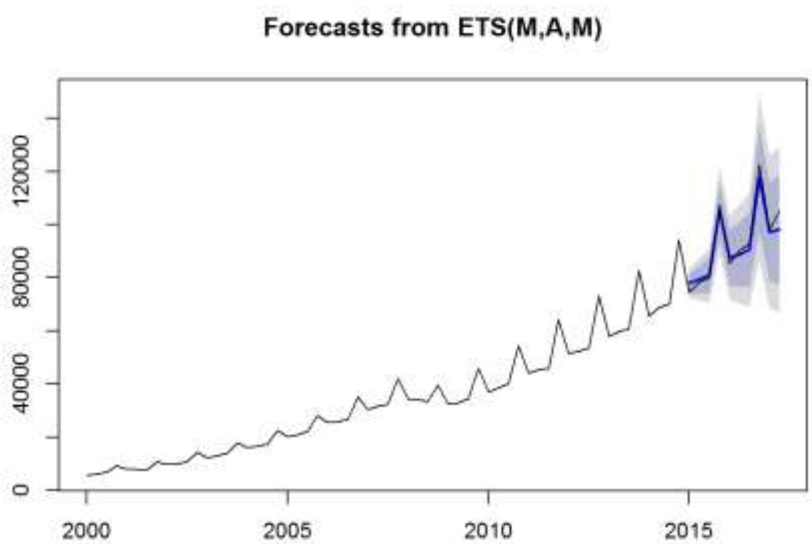
```
##
##      AIC     AICc      BIC
## 1088.910 1092.510 1107.759
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 225.4651 1430.024 967.2594 0.4233357 2.932154 0.1915216
##                     ACF1
## Training set -0.2121755
```

```
fit11 <- forecast(y.ets, h=h)
summary(fit11)
```

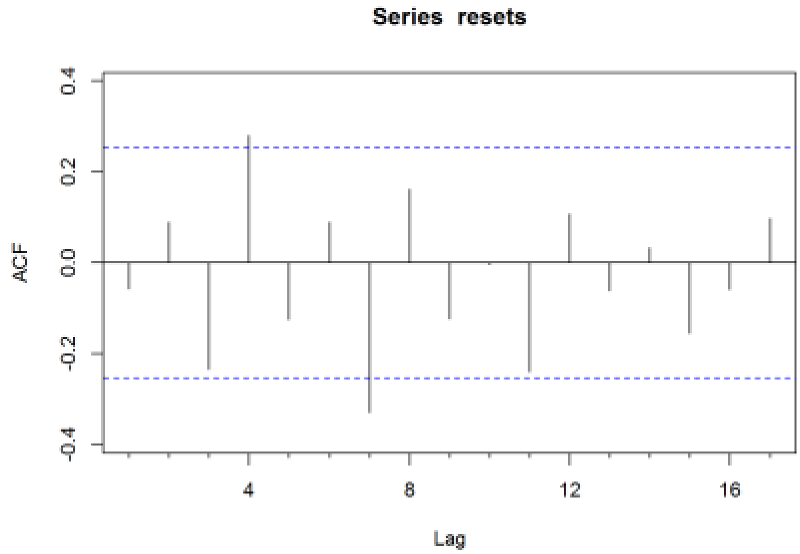
```
##
## Forecast method: ETS(M,A,M)
##
## Model Information:
## ETS(M,A,M)
##
## Call:
##  ets(y = train, model = "ZZZ")
##
##   Smoothing parameters:
##     alpha = 0.8555
##     beta  = 0.1601
##     gamma = 4e-04
##
##   Initial states:
##     l = 5561.9563
##     b = 609.276
##     s=1.1798 0.931 0.9382 0.951
##
##   sigma:  0.0365
##
##      AIC     AICc      BIC
## 1088.910 1092.510 1107.759
```

```
##
## Error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 225.4651 1430.024 967.2594 0.4233357 2.932154 0.1915216
##                     ACF1
## Training set -0.2121755
##
## Forecasts:
##         Point Forecast    Lo 80     Hi 80    Lo 95     Hi 95
## 2015 Q1       77989.69 74339.48  81639.89 72407.18  83572.19
## 2015 Q2       79324.24 74110.86  84537.63 71351.07  87297.42
## 2015 Q3       81070.72 74280.37  87861.08 70685.77  91455.67
## 2015 Q4      105728.54 94991.08 116466.01 89307.01 122150.08
## 2016 Q1       87631.44 77171.93  98090.95 71635.00 103627.88
## 2016 Q2       88836.98 76645.13 101028.82 70191.16 107482.79
## 2016 Q3       90509.91 76458.29 104561.53 69019.81 112000.00
## 2016 Q4      117690.49 97282.03 138098.95 86478.45 148902.54
## 2017 Q1       97273.22 78623.44 115922.99 68750.85 125795.59
## 2017 Q2       98349.74 77677.83 119021.65 66734.78 129964.70
```

```
plot(fit11)
lines(esales)
```



```
resets=resid(fit11)
Acf(resets)
```

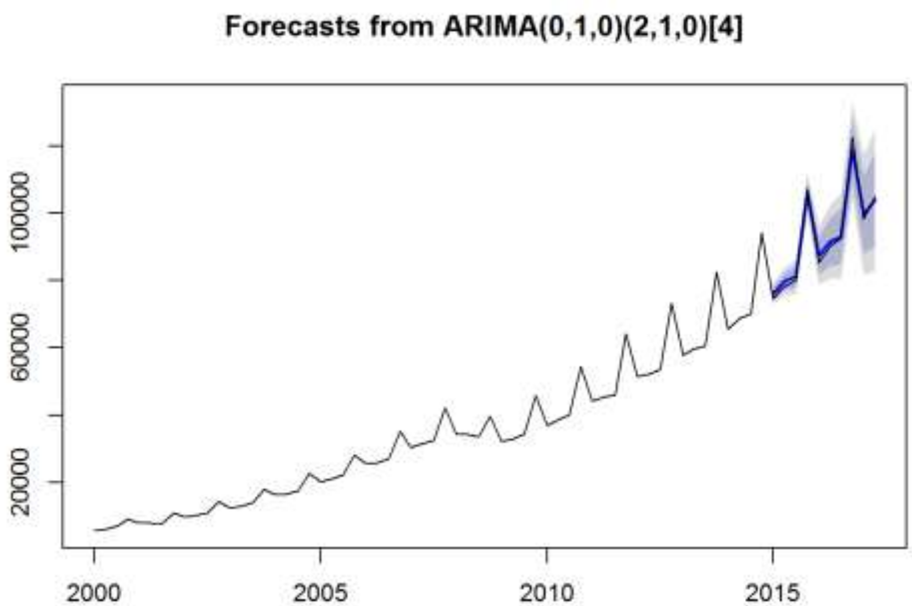


Method 12 and 13

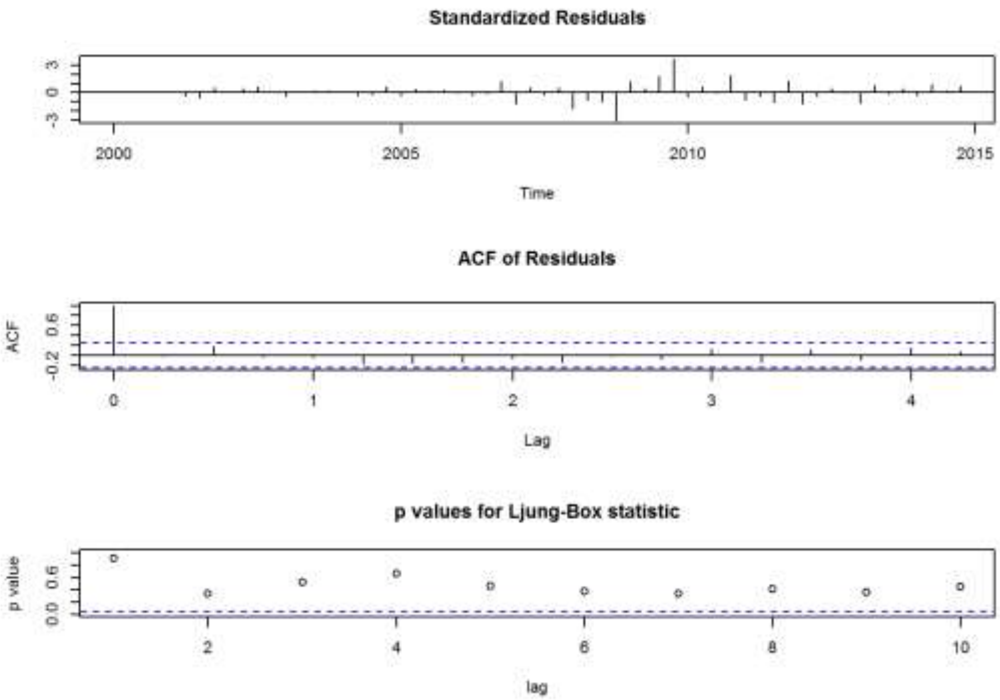
Now the ARIMA forecasts look the best as they almost perfectly fit the actual line, capturing the trend and seasonality precisely. Looking at the standardized residuals, we do not see many spikes other than the lag around 2009 and 2010 (probably due to financial crisis).

```
adf.test(train, alternative = "stationary")
## Warning in adf.test(train, alternative = "stationary"): p-value greater
## than printed p-value
##
##  Augmented Dickey-Fuller Test
##
## data:  train
## Dickey-Fuller = 1.6128, Lag order = 3, p-value = 0.99
## alternative hypothesis: stationary
kpss.test(train)
## Warning in kpss.test(train): p-value smaller than printed p-value
##
##  KPSS Test for Level Stationarity
```

```
##
## data:  train
## KPSS Level = 2.9178, Truncation lag parameter = 1, p-value = 0.01
ndiffs(train)
## [1] 1
y.arima <- auto.arima(train)
fit12 <- forecast(y.arima, h=h)
plot(fit12)
lines(esales)
```

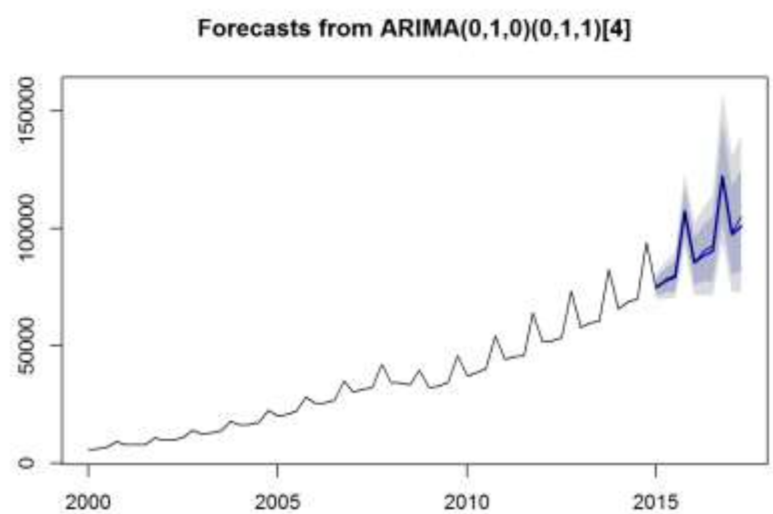



```
tsdiag(y.arima)
```

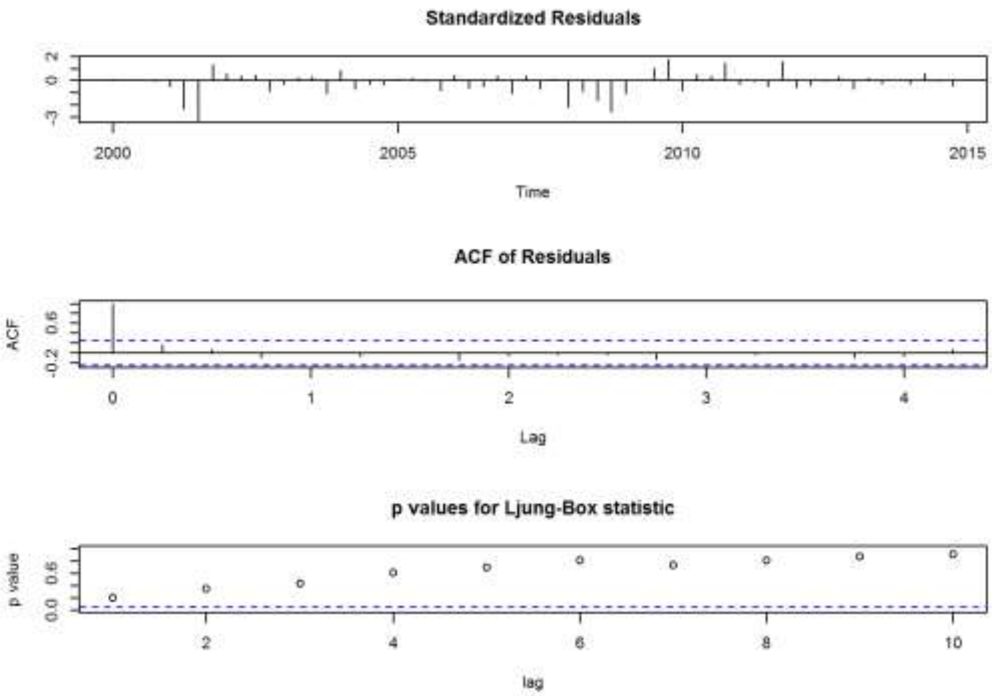


I also used the log transformed ARIMA to reduce variability and see if the forecasts would be even better. The results look similar to those of the ARIMA; however, confidence bands seem to be wider.

```
y.arima.lambda <- auto.arima(train, lambda=0)
fit13 <- forecast(y.arima.lambda, h=h)
plot(fit13)
lines(esales)
```



```r
tsdiag(y.arima.lambda)
```



Method 14

Forecasts from STL + Random walk do not look fit with wide confidence bands.

```r
y.stl <- stl(train, t.window=15, s.window="periodic", robust=TRUE)
summary(y.stl)
```

```
##  Call:
##  stl(x = train, s.window = "periodic", t.window = 15, robust = TRUE)
##
##  Time.series components:
##     seasonal              trend              remainder
##  Min.   :-2698.765   Min.   : 7855.55   Min.   :-7588.029
##  1st Qu.:-2404.905   1st Qu.:17090.20   1st Qu.: -597.894
##  Median :-2148.392   Median :33571.50   Median : -200.411
##  Mean   :    0.000   Mean   :34009.93   Mean   : -272.717
##  3rd Qu.:  256.512   3rd Qu.:46448.99   3rd Qu.:  257.020
##  Max.   : 6995.549   Max.   :74500.63   Max.   :12670.821
##  IQR:
##      STL.seasonal STL.trend STL.remainder data
```

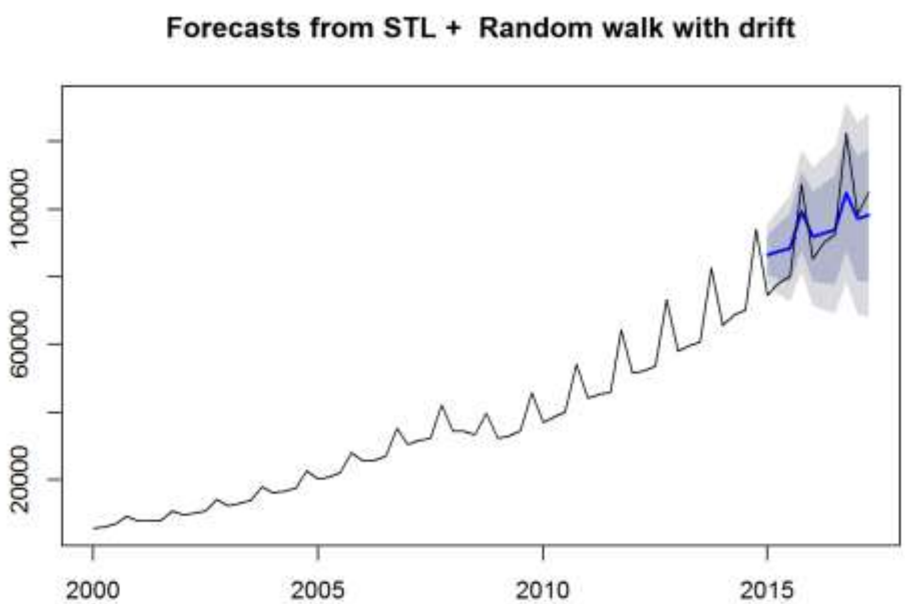
```
##       2661.4      29358.8     854.9       30171.8
##    %   8.8          97.3       2.8         100.0
##
##  Weights:
##     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.3788  0.9452  0.6984  0.9836  0.9998
##
##  Other components: List of 5
##  $ win  : Named num [1:3] 601 15 5
##  $ deg  : Named int [1:3] 0 1 1
##  $ jump : Named num [1:3] 61 2 1
##  $ inner: int 1
##  $ outer: int 15
```

```
fit14 <- forecast(y.stl,method="rwdrift", h=h)
summary(fit14)
```

```
##
## Forecast method: STL +  Random walk with drift
##
## Model Information:
## Call: rwf(y = x, h = h, drift = TRUE, level = level)
##
## Drift: 1349.485  (se 588.0014)
## Residual sd: 4516.5247
##
## Error measures:
##                         ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -7.397924e-13 4478.085 2961.554 -1.545368 13.66768 0.5864007
##                    ACF1
## Training set -0.4157671
##
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95     Hi 95
## 2015 Q1       86531.10 80742.94 92319.26 77678.88  95383.33
## 2015 Q2       87563.47 79308.70 95818.24 74938.89 100188.05
```

```
## 2015 Q3       88521.14 78327.25  98715.03 72930.93 104111.35
## 2015 Q4       99564.94 87697.96 111431.92 81415.96 117713.92
## 2016 Q1       91929.04 78554.78 105303.30 71474.88 112383.21
## 2016 Q2       92961.41 78194.82 107728.00 70377.87 115544.95
## 2016 Q3       93919.08 77845.22 109992.94 69336.24 118501.93
## 2016 Q4      104962.88 87647.52 122278.24 78481.33 131444.43
## 2017 Q1       97326.98 78822.66 115831.30 69027.07 125626.90
## 2017 Q2       98359.35 78709.06 118009.64 68306.83 128411.87
```

```
plot(fit14)
lines(esales)
```

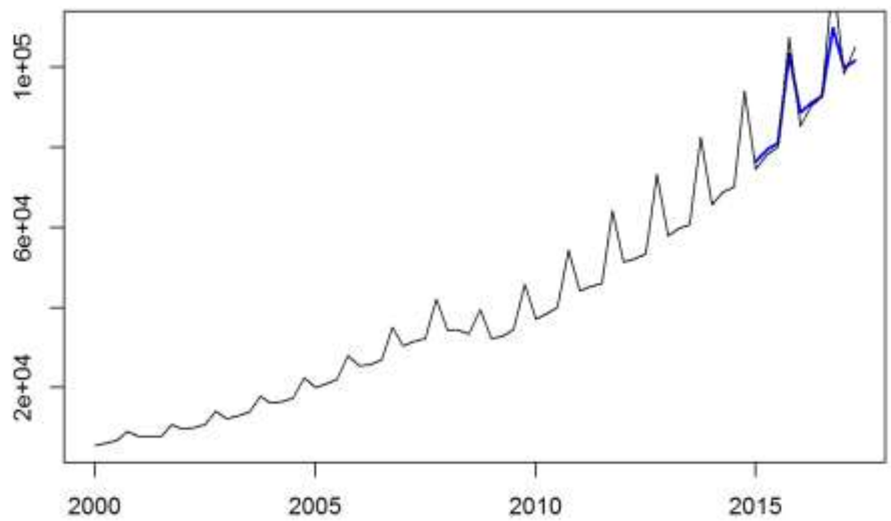


Method 15 and 16

Moreover, I used more advanced models: ANN and BATS. Unfortunately, forecasts produced by them do not look as fit as ETS and ARIMA.

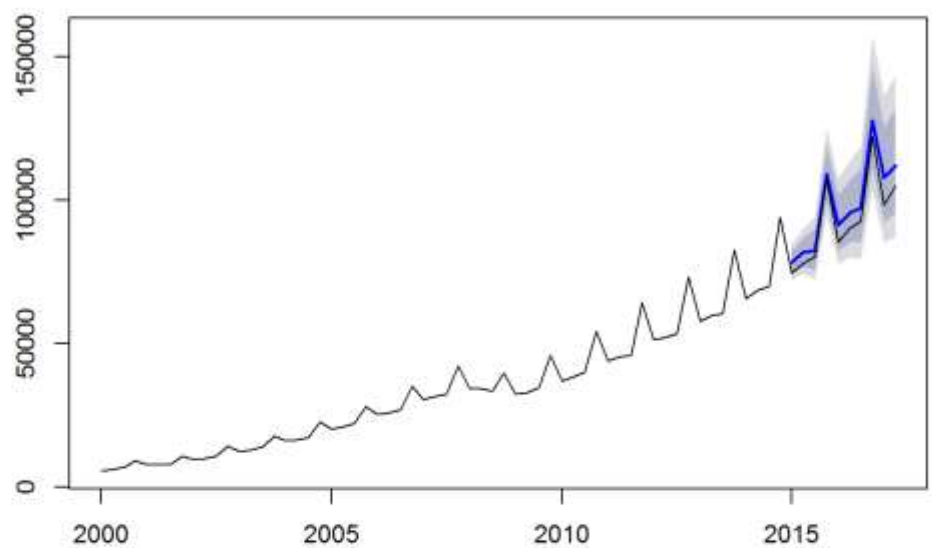
```
y.ann <- nnetar(train)
fit15 <- forecast(y.ann, h=h)
plot(forecast(fit15,h=h))
lines(esales)
```

Forecasts from NNAR(1,1,2)[4]




```
tbats = tbats(train)
fit16 <- forecast(tbats, h=h)
plot(fit16)
lines(esales)
```

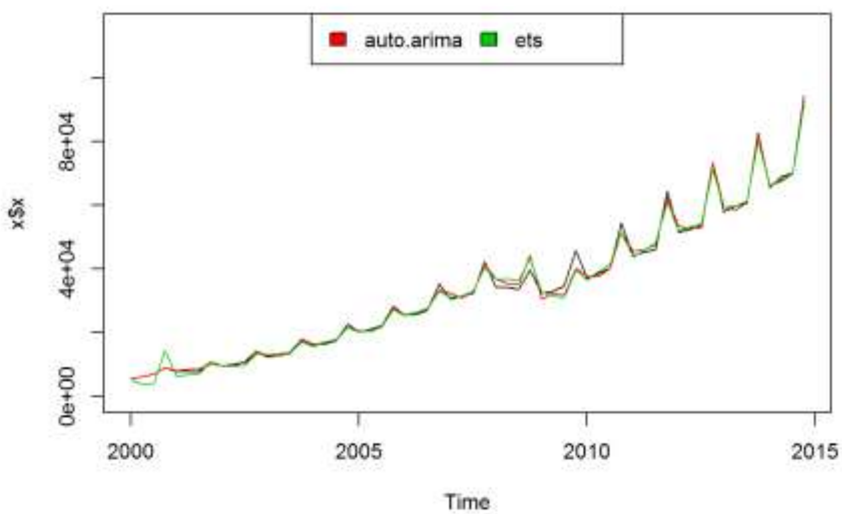
Forecasts from TBATS(0.047, {4,0}, 1, {<4,1>})



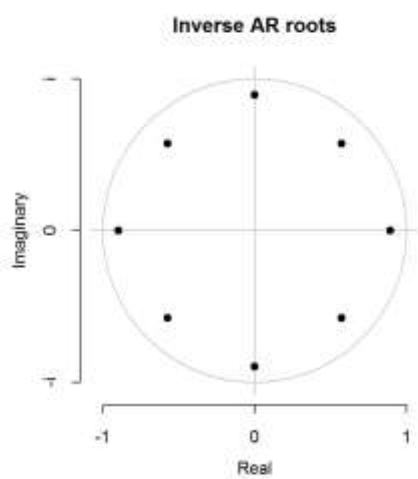
Method 17

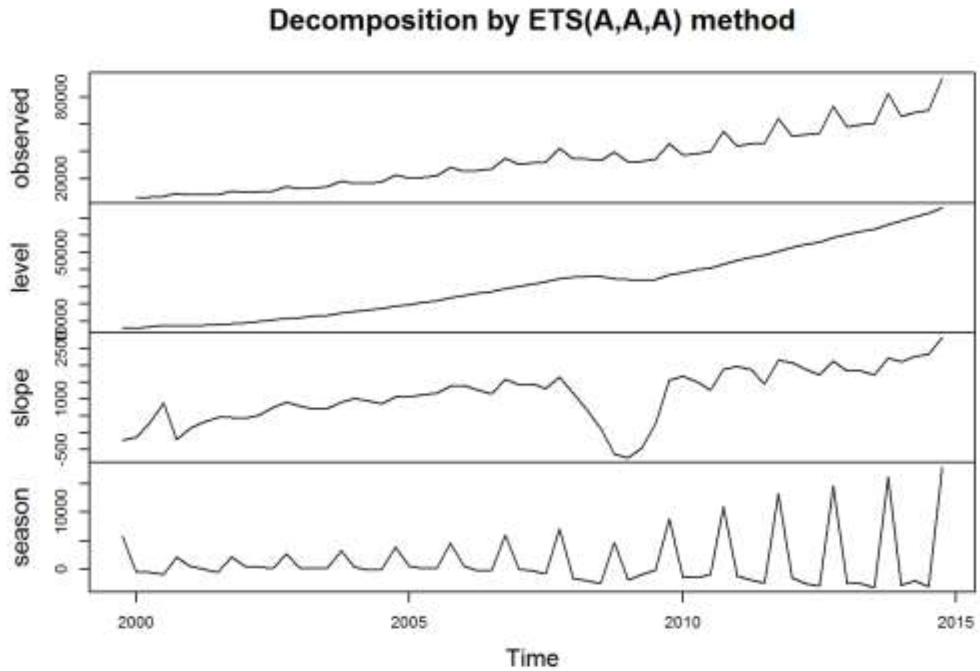
The last method I used is the hybrid. I equally combined ETS and ARIMA as they are the two best models so far. The forecasts look fit to the actual data during the test period, although the confidence bands seem wider than ARIMA's.

```
hmod <- hybridModel(train, model="ae",lambda=TRUE)
## Fitting the auto.arima model
## Fitting the ets model
plot(hmod)
```

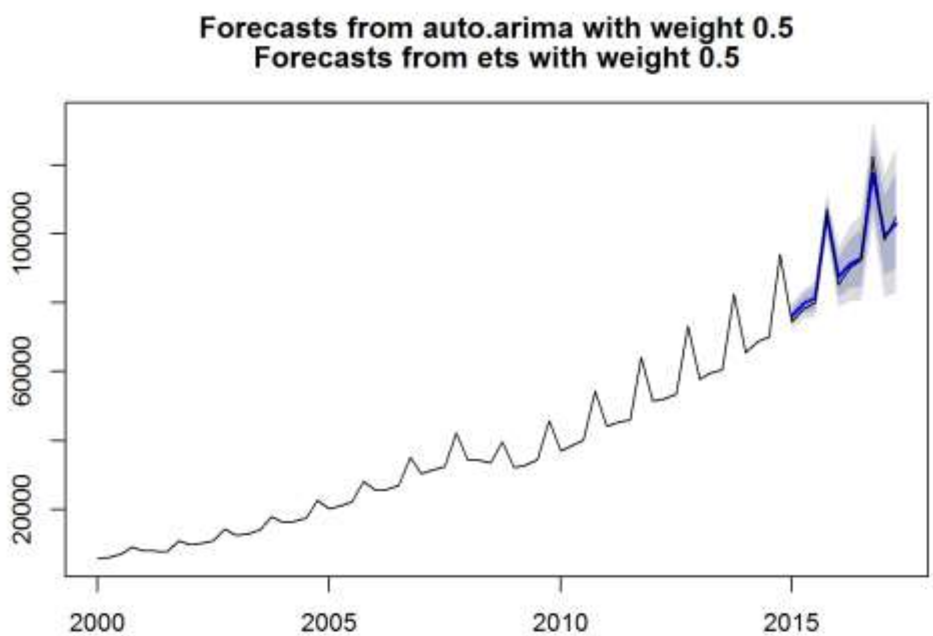



```
plot(hmod, type = "models")
```






```
fit17 <- forecast(hmod, h=h)
plot(fit17)
lines(esales)
```



Accuracy Measures

I computed the accuracy measures and organized them in the table below to see which method was the best for the e-commerce data. The log transformed ARIMA method produces the smallest errors across all measures for the test set. We can conclude that it is the best univariate forecasting method. The ARIMA and Hybrid models would be suitable too, as their error measures are very close to the smallest errors from the log transformed ARIMA.

```
#accuracy measures
a1 = accuracy(fit1, test)
a2 = accuracy(fit2, test)
a3 = accuracy(fit3, test)
a4 = accuracy(fit4, test)
a5 = accuracy(fit5, test)
a6 = accuracy(fit6, test)
a7 = accuracy(fit7, test)
a8 = accuracy(fit8, test)
a9 = accuracy(fit9, test)
a10 = accuracy(fit10, test)
a11 = accuracy(fit11, test)
a12 = accuracy(fit12, test)
a13 = accuracy(fit13, test)
a14 = accuracy(fit14, test)
a15 = accuracy(fit15, test)
a16 = accuracy(fit16, test)
a17 = accuracy(fit17, test)


a.table<-rbind(a1, a2, a3, a4, a5, a6, a7, a8, a9, a10, a11, a12, a13, a14, a15, a1
6, a17)
row.names(a.table)<-c('Mean training','Mean test', 'Naive training', 'Naive test',
'S. Naive training', 'S. Naive test' ,
                      'linear trend training','linear trend test', 'linear t+s tran
ing', 'linear t+s test',
                      'ES training','ES test','Holt Winters training', 'Holt Winter
s test',
                      'Holt linear training', 'Holt linear test', 'Holt ES training
', 'Holt ES test' ,
                      'Holt dampled training','Holt damped test', 'ETS training', '
ETS test',
```

```r
                      'ARIMA training', 'ARIMA test', 'ARIMA training (log)', 'ARIM
A test (log)',
                      'STL training','STL test','ANN training', 'ANN test',
                      'TBATS training', 'TBATS test',
                      'Hybrid training', 'Hybrid test')

write.csv(a.table, "C:/Schulich/econ6120/atable.csv")

# order the table according to MASE
a.table<-as.data.frame(a.table)
a.table<-a.table[order(a.table$MASE),]
a.table
```

```
##                                ME     RMSE       MAE         MPE
## ARIMA training (log)  -1.392320e+02 1175.772  770.7830 -0.75090635
## Holt Winters training  1.505839e+02 1271.044  846.2931  0.17597700
## ARIMA training         9.803767e+01 1405.355  937.0338  0.17496104
## ETS training           2.254651e+02 1430.024  967.2594  0.42333569
## TBATS training        -1.408556e+02 1431.166 1016.1801 -0.21713495
## Hybrid training        1.748198e+02 1505.001 1034.3390  0.88338509
## ARIMA test (log)       9.344749e+02 1587.035 1066.9622  0.96252332
## ANN training           6.197664e+00 1938.935 1266.1512 -0.40186524
## ARIMA test            -3.243540e+02 1674.208 1471.4778 -0.57204647
## Hybrid test           -9.046320e+01 2012.242 1696.2760 -0.37620280
## ETS test               1.038503e+03 3134.330 2559.1213  0.73402770
## STL training          -7.397924e-13 4478.085 2961.5542 -1.54536824
## ANN test               1.033519e+03 4586.521 3014.3388  0.56103229
## Holt linear training   6.523981e+02 5718.061 3526.3018 -0.03127299
## Holt Winters test      3.406901e+03 4484.507 3559.9683  3.38548201
## Holt dampled training  7.923264e+02 5735.240 3571.0349  0.54383491
## linear t+s traning    -8.940700e-13 5033.456 3620.7700  5.89970037
## ES training            2.583205e+03 6737.880 3678.5056  6.98364514
## Holt ES training      -1.456273e+02 5779.713 3923.0736 -1.21449593
## linear trend training  1.665631e-13 6380.753 4524.3416  4.07013302
## Naive training         1.501780e+03 7689.305 4750.0169  3.43600574
## TBATS test            -4.943455e+03 5410.614 4943.4549 -5.29755508
```

```
## S. Naive training      4.840607e+03  5659.009  5050.3929  15.39468124
## STL test              -6.849404e+02  8754.818  7309.3096  -2.11491616
## Holt damped test       9.826861e+02 10438.301  7812.1980  -0.28846736
## Holt linear test      -1.704881e+03 10276.939  9039.2054  -3.08417891
## Naive test            -6.880000e+02 14291.829 12021.6000  -3.00523621
## Holt ES test          -9.072866e+03 13709.343 13339.6370 -10.70309657
## ES test                1.210320e+04 18715.517 14332.2817  10.98651259
## Mean training         -4.863665e-13 21215.070 17149.0050 -67.88648902
## linear trend test      1.885121e+04 22400.418 18851.2094  18.78294270
## linear t+s test        1.965402e+04 21699.250 19654.0193  20.05305670
## S. Naive test          2.030500e+04 22373.200 20305.0000  21.13641225
## Mean test              5.974178e+04 61423.641 59741.7833  63.09630792
##                            MAPE      MASE        ACF1  Theil's U
## ARIMA training (log)   2.482677  0.1526184  0.07689886         NA
## Holt Winters training  3.048849  0.1675698  0.32210621         NA
## ARIMA training         3.027342  0.1855368 -0.01535440         NA
## ETS training           2.932154  0.1915216 -0.21217551         NA
## TBATS training         3.204732  0.2012081  0.14394909         NA
## Hybrid training        4.114391  0.2048037  0.08826688         NA
## ARIMA test (log)       1.134630  0.2112632  0.02168395 0.09482795
## ANN training           3.951814  0.2507035  0.69400304         NA
## ARIMA test             1.561673  0.2913591 -0.21399467 0.09931664
## Hybrid test            1.771250  0.3358701 -0.16956383 0.12097996
## ETS test               2.667839  0.5067173  0.14987997 0.17538053
## STL training          13.667680  0.5864007 -0.41576710         NA
## ANN test               2.929174  0.5968523 -0.23781407 0.27882524
## Holt linear training   9.402263  0.6982233 -0.12933380         NA
## Holt Winters test      3.590743  0.7048894  0.50036944 0.26290577
## Holt dampled training  9.533172  0.7070806 -0.12434597         NA
## linear t+s traning    18.377015  0.7169284  0.46698053         NA
## ES training            9.902160  0.7283603 -0.23386619         NA
## Holt ES training      10.471371  0.7767858 -0.10068965         NA
## linear trend training 18.174289  0.8958395  0.16580084         NA
## Naive training        12.383297  0.9405242 -0.42113903         NA
```

```
## TBATS test              5.297555  0.9788258  0.45638256 0.29958427
## S. Naive training      15.989567  1.0000000  0.79985485         NA
## STL test                7.923312  1.4472754  0.12137867 0.51282300
## Holt damped test        7.937521  1.5468496 -0.27502095 0.66487940
## Holt linear test        9.451351  1.7898024 -0.33029186 0.64367563
## Naive test             13.021695  2.3803297  0.19264832 0.84145774
## Holt ES test           14.453496  2.6413068 -0.31884764 0.80392378
## ES test                13.914024  2.8378548  0.19264832 1.14636916
## Mean training          95.673845  3.3955784  0.85309991         NA
## linear trend test      18.782943  3.7326224 -0.01283008 1.39812722
## linear t+s test        20.053057  3.8915823  0.16214924 1.33288097
## S. Naive test          21.136412  4.0204793  0.71120897 1.29259597
## Mean test              63.096308 11.8291359  0.19264832 3.74256145
```

```
format(round(a.table, 3), nsmall = 3)
```

```
##                              ME     RMSE      MAE    MPE   MAPE  MASE
## ARIMA training (log)   -139.232 1175.772  770.783 -0.751  2.483 0.153
## Holt Winters training   150.584 1271.044  846.293  0.176  3.049 0.168
## ARIMA training           98.038 1405.355  937.034  0.175  3.027 0.186
## ETS training            225.465 1430.024  967.259  0.423  2.932 0.192
## TBATS training         -140.856 1431.166 1016.180 -0.217  3.205 0.201
## Hybrid training         174.820 1505.001 1034.339  0.883  4.114 0.205
## ARIMA test (log)        934.475 1587.035 1066.962  0.963  1.135 0.211
## ANN training              6.198 1938.935 1266.151 -0.402  3.952 0.251
## ARIMA test             -324.354 1674.208 1471.478 -0.572  1.562 0.291
## Hybrid test             -90.463 2012.242 1696.276 -0.376  1.771 0.336
## ETS test               1038.503 3134.330 2559.121  0.734  2.668 0.507
## STL training              0.000 4478.085 2961.554 -1.545 13.668 0.586
## ANN test               1033.519 4586.521 3014.339  0.561  2.929 0.597
## Holt linear training    652.398 5718.061 3526.302 -0.031  9.402 0.698
## Holt Winters test      3406.901 4484.507 3559.968  3.385  3.591 0.705
## Holt dampled training   792.326 5735.240 3571.035  0.544  9.533 0.707
## linear t+s traning        0.000 5033.456 3620.770  5.900 18.377 0.717
## ES training            2583.205 6737.880 3678.506  6.984  9.902 0.728
## Holt ES training       -145.627 5779.713 3923.074 -1.214 10.471 0.777
```

```
## linear trend training     0.000  6380.753  4524.342   4.070 18.174  0.896
## Naive training         1501.780  7689.305  4750.017   3.436 12.383  0.941
## TBATS test            -4943.455  5410.614  4943.455  -5.298  5.298  0.979
## S. Naive training      4840.607  5659.009  5050.393  15.395 15.990  1.000
## STL test               -684.940  8754.818  7309.310  -2.115  7.923  1.447
## Holt damped test        982.686 10438.301  7812.198  -0.288  7.938  1.547
## Holt linear test      -1704.881 10276.939  9039.205  -3.084  9.451  1.790
## Naive test             -688.000 14291.829 12021.600  -3.005 13.022  2.380
## Holt ES test          -9072.866 13709.343 13339.637 -10.703 14.453  2.641
## ES test               12103.204 18715.517 14332.282  10.987 13.914  2.838
## Mean training             0.000 21215.070 17149.005 -67.886 95.674  3.396
## linear trend test     18851.209 22400.418 18851.209  18.783 18.783  3.733
## linear t+s test       19654.019 21699.250 19654.019  20.053 20.053  3.892
## S. Naive test         20305.000 22373.200 20305.000  21.136 21.136  4.020
## Mean test             59741.783 61423.641 59741.783  63.096 63.096 11.829
##                         ACF1 Theil's U
## ARIMA training (log)   0.077        NA
## Holt Winters training  0.322        NA
## ARIMA training        -0.015        NA
## ETS training          -0.212        NA
## TBATS training         0.144        NA
## Hybrid training        0.088        NA
## ARIMA test (log)       0.022     0.095
## ANN training           0.694        NA
## ARIMA test            -0.214     0.099
## Hybrid test           -0.170     0.121
## ETS test               0.150     0.175
## STL training          -0.416        NA
## ANN test              -0.238     0.279
## Holt linear training  -0.129        NA
## Holt Winters test      0.500     0.263
## Holt dampled training -0.124        NA
## linear t+s traning     0.467        NA
## ES training           -0.234        NA
```

```
## Holt ES training      -0.101        NA
## linear trend training  0.166        NA
## Naive training        -0.421        NA
## TBATS test             0.456     0.300
## S. Naive training      0.800        NA
## STL test               0.121     0.513
## Holt damped test      -0.275     0.665
## Holt linear test      -0.330     0.644
## Naive test             0.193     0.841
## Holt ES test          -0.319     0.804
## ES test                0.193     1.146
## Mean training          0.853        NA
## linear trend test     -0.013     1.398
## linear t+s test        0.162     1.333
## S. Naive test          0.711     1.293
## Mean test              0.193     3.743
```

Univariate Model Forecasts

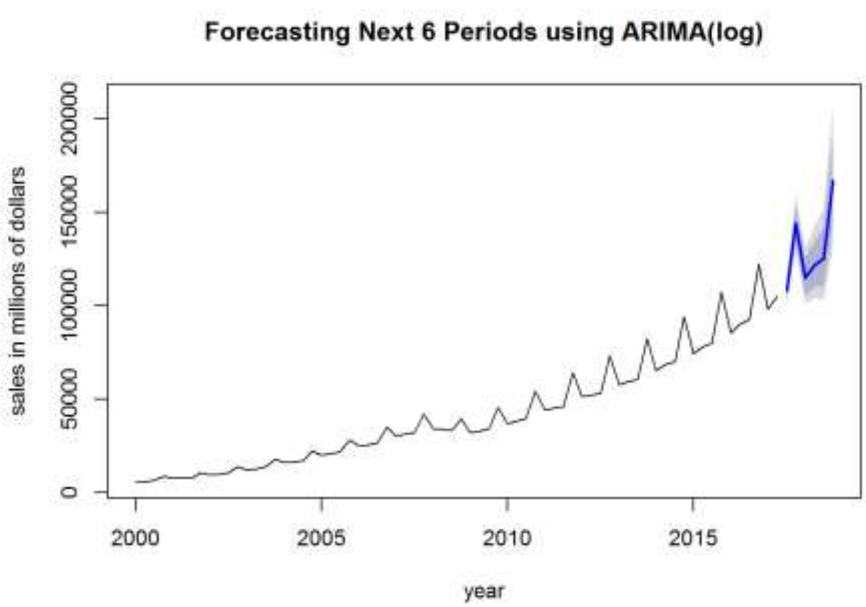
The forecasts from log transformed ARIMA are shown below. The actual Q3 2017 data is $107,002 million of dollars (obtained from FRED). This is similar to the Q3 sales (107,980.4) predicted by the log transformed ARIMA model. The ecommerce retail sales will continue to increase; we will continue to shop online.

```
f.arima.lambdaf <- auto.arima(esales,lambda=0)
f <- forecast(f.arima.lambdaf, h=6)
summary(f)
```

```
##
## Forecast method: ARIMA(1,1,0)(0,1,1)[4]
##
## Model Information:
## Series: esales
## ARIMA(1,1,0)(0,1,1)[4]
## Box Cox transformation: lambda= 0
##
## Coefficients:
##          ar1     sma1
##       0.2009  -0.4844
```

```
## s.e.   0.1269   0.1093
##
## sigma^2 estimated as 0.001143:  log likelihood=128.56
## AIC=-251.12   AICc=-250.72   BIC=-244.59
##
## Error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -64.06663 1152.068 802.6072 -0.4801543 2.267987 0.1315465
##                    ACF1
## Training set -0.1040567
##
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2017 Q3       107980.4 103402.4 112761.1 101058.0 115376.9
## 2017 Q4       144125.7 134691.1 154221.1 129949.4 159848.4
## 2018 Q1       115111.8 105577.4 125507.2 100854.1 131385.0
## 2018 Q2       122051.8 110215.9 135158.7 104422.3 142657.6
## 2018 Q3       125190.3 110201.6 142217.6 103007.8 152149.7
## 2018 Q4       167039.8 143615.6 194284.6 132576.1 210462.5
```

```
plot(f, main="Forecasting Next 6 Periods using ARIMA(log)", xlab="year", ylab="sale
s in millions of dollars")
```



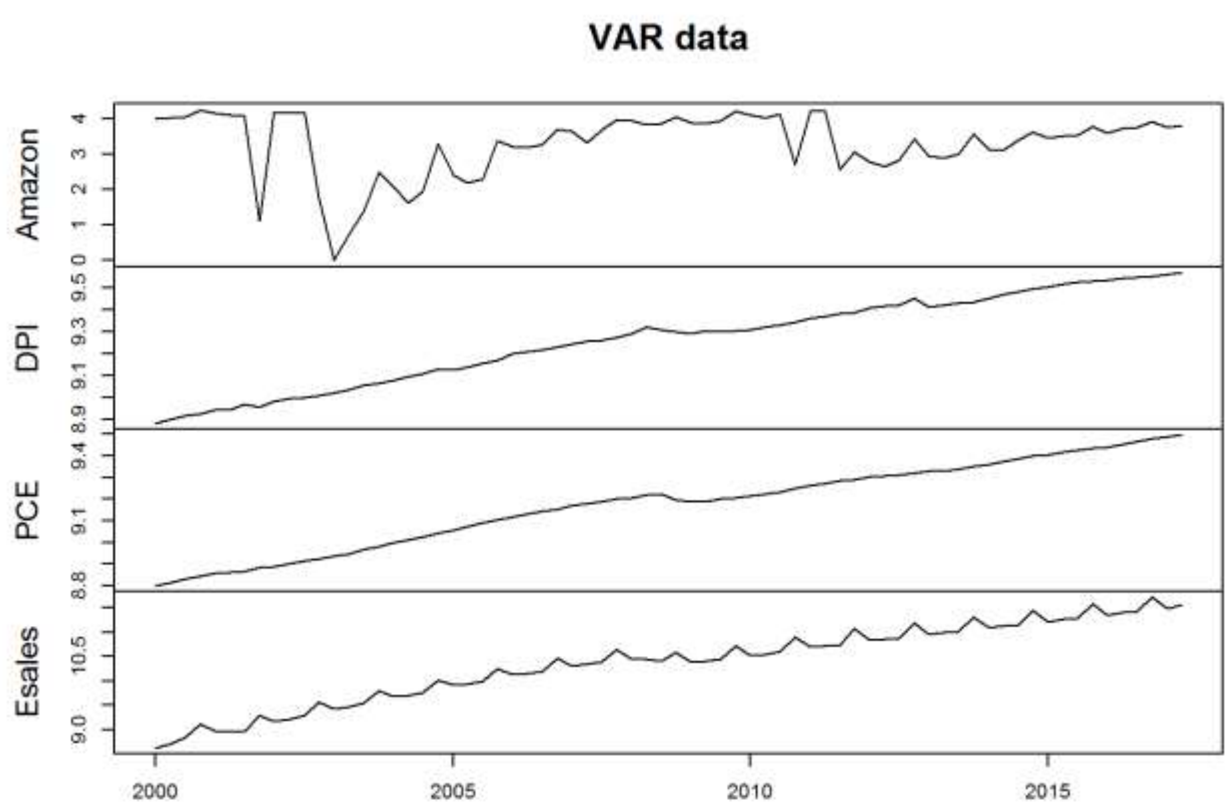
Multivariate Analysis

VAR Estimation

I built a VAR model for a collection of related variables. The explanatory variables I used are macroeconomics measures including US Personal Consumption Expenditure and US Disposable Personal Income obtained from FRED, and the ecommerce giant, Amazon's revenue from Bloomberg. I was planning to use number of internet users, and Facebook and Google's revenues as well; however, I was unable to find the complete 17 years of quarterly data for these three variables. GDP is too broad and thus I decided to narrow it down to consumption.

Ecommerce retail sales and the other 3 variables are plotted below:

```
vardata = log (sales[,c(6,5,4,2)])
colnames(vardata) = c( "Amazon","DPI", "PCE","Esales")
nrow(vardata)
## [1] 70
plot(vardata, main = "VAR data", xlab = "")
```



The R out below shows the lag length selected by each of the information criteria available. They suggested different lags. I decided to go with HQ, because when I tried the other three, the hypothesis of no serial correlation would be rejected.

```
vs = VARselect(vardata, lag.max =9, season =4)
vs
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      9      2      1      7
##
## $criteria
##                    1             2             3             4
## AIC(n) -2.772685e+01 -2.808661e+01 -2.786191e+01 -2.788128e+01
## HQ(n)  -2.729288e+01 -2.743565e+01 -2.699395e+01 -2.679634e+01
## SC(n)  -2.661951e+01 -2.642560e+01 -2.564722e+01 -2.511293e+01
## FPE(n)  9.141557e-13  6.473896e-13  8.347057e-13  8.609613e-13
##                    5             6             7             8
## AIC(n) -2.816773e+01 -2.833733e+01 -2.855106e+01 -2.855146e+01
## HQ(n)  -2.686580e+01 -2.681841e+01 -2.681515e+01 -2.659855e+01
## SC(n)  -2.484570e+01 -2.446163e+01 -2.412169e+01 -2.356841e+01
## FPE(n)  6.995480e-13  6.637000e-13  6.341015e-13  8.037983e-13
##                    9
## AIC(n) -2.879606e+01
## HQ(n)  -2.662617e+01
## SC(n)  -2.325934e+01
## FPE(n)  8.793724e-13
vs$selection[2]
## HQ(n)
##     2
```

Summary of the model is shown. Under “Estimation results for equation Esales”, we see a high r-square and small p-value. The roots are less than 1. These suggest the model is a good fit. On the correlation matrix of residuals, we see that the errors of ecommerce sales and personal consumption expenditures are correlated with R-square equal to 0.6974. It is strange to see that ecommerce sales correlates negatively with Amazon’s revenue, given that Amazon is one of the largest ecommerce retailer in US.

```
var.1 = VAR(vardata, p=vs$selection[2], season =4)
```

```
summary(var.1)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: Amazon, DPI, PCE, Esales
## Deterministic variables: const
## Sample size: 68
## Log Likelihood: 591.519
## Roots of the characteristic polynomial:
## 0.9814 0.8742 0.8742 0.3484 0.3484 0.3084 0.3084 0.08982
## Call:
## VAR(y = vardata, p = vs$selection[2], season = 4L)
##
##
## Estimation results for equation Amazon:
## =========================================
## Amazon = Amazon.l1 + DPI.l1 + PCE.l1 + Esales.l1 + Amazon.l2 + DPI.l2 + PCE.l2 + Esales.l2 + const + sd1 + sd2 + sd3
##
##            Estimate Std. Error t value Pr(>|t|)
## Amazon.l1    0.5757     0.1371   4.198 9.74e-05 ***
## DPI.l1     -17.3633    11.0216  -1.575   0.1208
## PCE.l1      22.2663    19.0869   1.167   0.2483
## Esales.l1    0.4808     3.2942   0.146   0.8845
## Amazon.l2   -0.1242     0.1397  -0.889   0.3776
## DPI.l2      12.5259    10.5416   1.188   0.2398
## PCE.l2       2.4764    20.3156   0.122   0.9034
## Esales.l2   -5.0995     3.1150  -1.637   0.1072
## const     -132.5360    60.3704  -2.195   0.0323 *
## sd1         -0.1962     0.8587  -0.229   0.8201
## sd2          1.1296     0.7142   1.582   0.1194
## sd3          0.1441     0.2548   0.566   0.5740
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
##
## Residual standard error: 0.7249 on 56 degrees of freedom
## Multiple R-Squared: 0.4754,  Adjusted R-squared: 0.3723
## F-statistic: 4.613 on 11 and 56 DF,  p-value: 5.686e-05
##
##
## Estimation results for equation DPI:
## ====================================
## DPI = Amazon.l1 + DPI.l1 + PCE.l1 + Esales.l1 + Amazon.l2 + DPI.l2 + PCE.l2 + Es
ales.l2 + const + sd1 + sd2 + sd3
##
##             Estimate Std. Error t value Pr(>|t|)
## Amazon.l1 -0.0009319  0.0017965  -0.519   0.6060
## DPI.l1     0.4612035  0.1443667   3.195   0.0023 **
## PCE.l1     0.4450503  0.2500107   1.780   0.0805 .
## Esales.l1  0.0186013  0.0431493   0.431   0.6681
## Amazon.l2 -0.0012417  0.0018294  -0.679   0.5001
## DPI.l2     0.1952570  0.1380797   1.414   0.1629
## PCE.l2    -0.1179817  0.2661056  -0.443   0.6592
## Esales.l2 -0.0149260  0.0408017  -0.366   0.7159
## const      0.1609905  0.7907671   0.204   0.8394
## sd1       -0.0038340  0.0112475  -0.341   0.7345
## sd2        0.0072041  0.0093554   0.770   0.4445
## sd3        0.0020600  0.0033373   0.617   0.5396
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.009495 on 56 degrees of freedom
## Multiple R-Squared: 0.998,   Adjusted R-squared: 0.9976
## F-statistic:  2486 on 11 and 56 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation PCE:
## ====================================
```

```
## PCE = Amazon.l1 + DPI.l1 + PCE.l1 + Esales.l1 + Amazon.l2 + DPI.l2 + PCE.l2 + Es
ales.l2 + const + sd1 + sd2 + sd3
##
##              Estimate Std. Error t value Pr(>|t|)
## Amazon.l1  0.0002946  0.0010121   0.291   0.7721
## DPI.l1     0.1104666  0.0813348   1.358   0.1799
## PCE.l1     1.2224659  0.1408537   8.679 5.98e-12 ***
## Esales.l1  0.0366273  0.0243099   1.507   0.1375
## Amazon.l2 -0.0006372  0.0010307  -0.618   0.5389
## DPI.l2    -0.1757898  0.0777928  -2.260   0.0277 *
## PCE.l2    -0.2700679  0.1499214  -1.801   0.0770 .
## Esales.l2 -0.0093775  0.0229873  -0.408   0.6849
## const      0.7654407  0.4455106   1.718   0.0913 .
## sd1       -0.0080720  0.0063367  -1.274   0.2080
## sd2        0.0031129  0.0052708   0.591   0.5572
## sd3        0.0016000  0.0018802   0.851   0.3984
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.005349 on 56 degrees of freedom
## Multiple R-Squared: 0.9993,  Adjusted R-squared: 0.9992
## F-statistic:  7601 on 11 and 56 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation Esales:
## =======================================
## Esales = Amazon.l1 + DPI.l1 + PCE.l1 + Esales.l1 + Amazon.l2 + DPI.l2 + PCE.l2 +
 Esales.l2 + const + sd1 + sd2 + sd3
##
##             Estimate Std. Error t value Pr(>|t|)
## Amazon.l1 -0.008244   0.006870  -1.200  0.23517
## DPI.l1     0.207576   0.552089   0.376  0.70835
## PCE.l1     1.550454   0.956094   1.622  0.11049
## Esales.l1  0.587046   0.165012   3.558  0.00077 ***
```

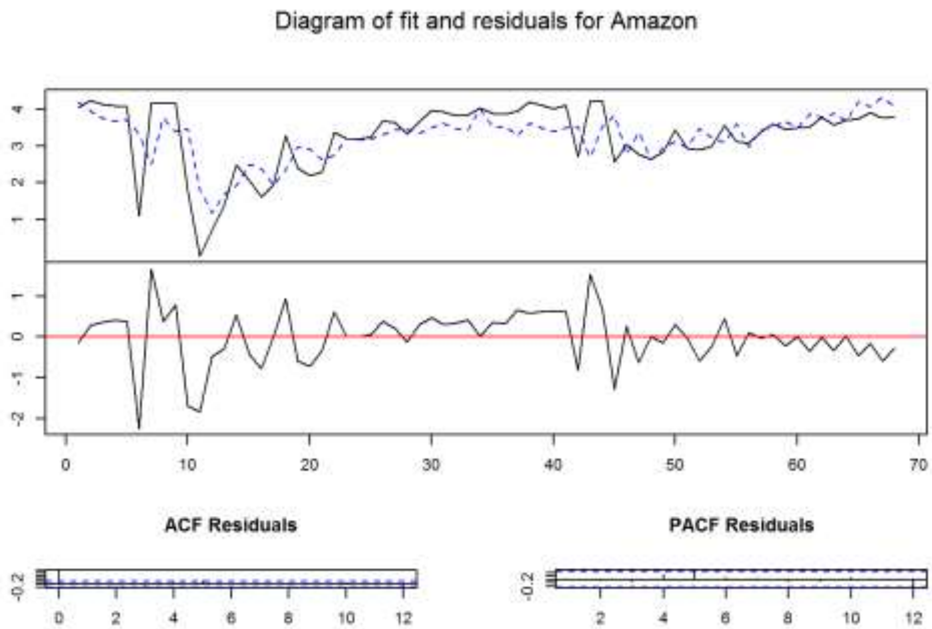
```
## Amazon.l2 -0.001380   0.006996  -0.197  0.84437
## DPI.l2     0.008538   0.528046   0.016  0.98716
## PCE.l2    -1.299916   1.017645  -1.277  0.20674
## Esales.l2  0.281637   0.156034   1.805  0.07646 .
## const     -2.865904   3.024062  -0.948  0.34735
## sd1       -0.354526   0.043013  -8.242 3.09e-11 ***
## sd2       -0.300972   0.035777  -8.412 1.63e-11 ***
## sd3       -0.243175   0.012763 -19.054  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.03631 on 56 degrees of freedom
## Multiple R-Squared: 0.9981,  Adjusted R-squared: 0.9978
## F-statistic:  2734 on 11 and 56 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##             Amazon       DPI        PCE     Esales
## Amazon  0.5254741 1.348e-03 -7.453e-04 -3.521e-03
## DPI     0.0013477 9.016e-05  1.788e-05  8.862e-05
## PCE    -0.0007453 1.788e-05  2.862e-05  1.355e-04
## Esales -0.0035205 8.862e-05  1.355e-04  1.319e-03
##
## Correlation matrix of residuals:
##         Amazon    DPI     PCE  Esales
## Amazon  1.0000 0.1958 -0.1922 -0.1337
## DPI     0.1958 1.0000  0.3521  0.2570
## PCE    -0.1922 0.3521  1.0000  0.6974
## Esales -0.1337 0.2570  0.6974  1.0000
```

```r
roots(var.1)
```

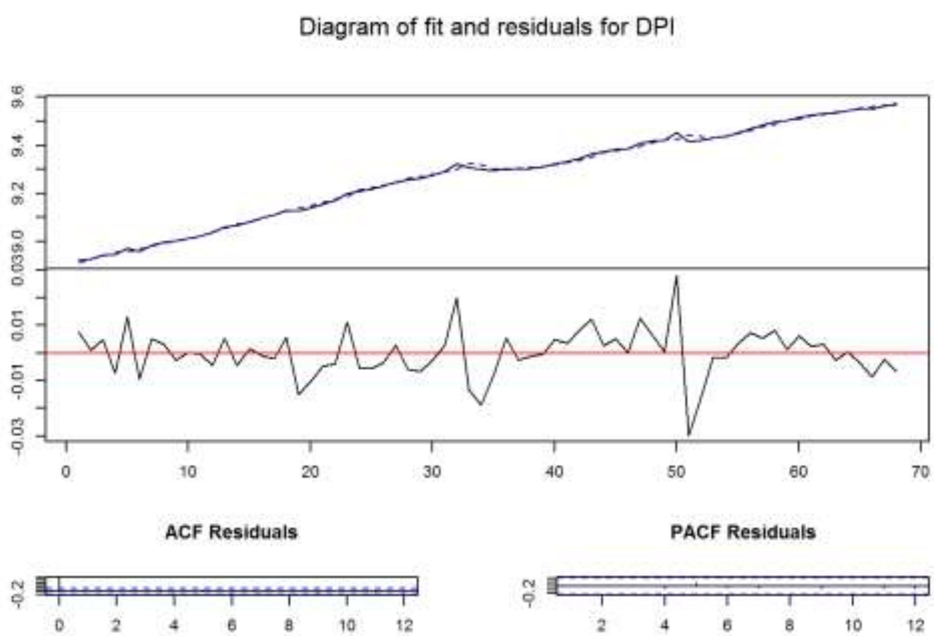
```
## [1] 0.98141910 0.87415083 0.87415083 0.34844881 0.34844881 0.30841276
## [7] 0.30841276 0.08982099
```

The variables are plotted below.

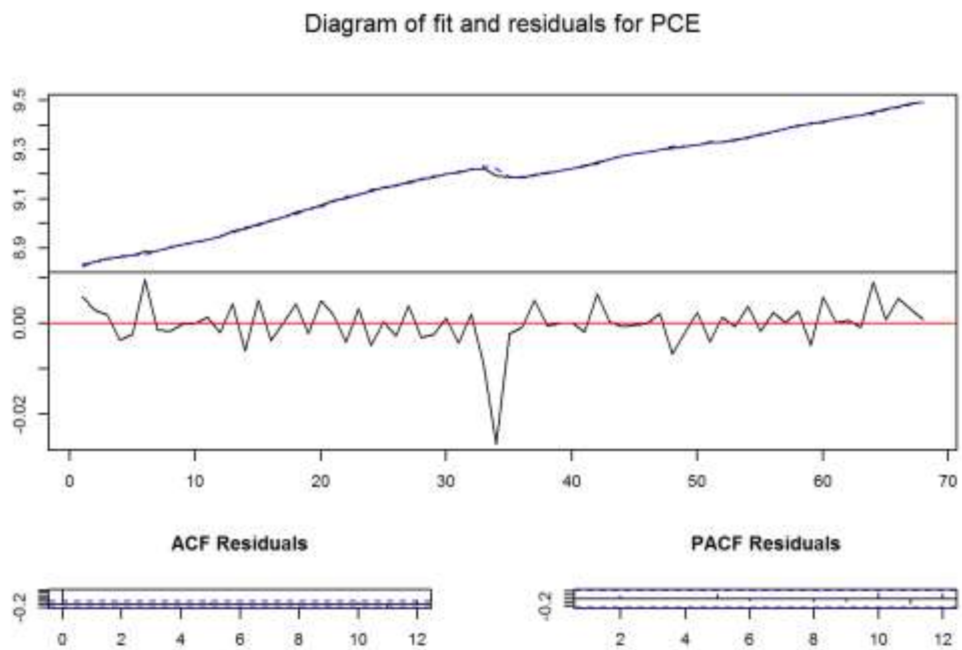
```
plot(var.1,names = "Amazon")
```



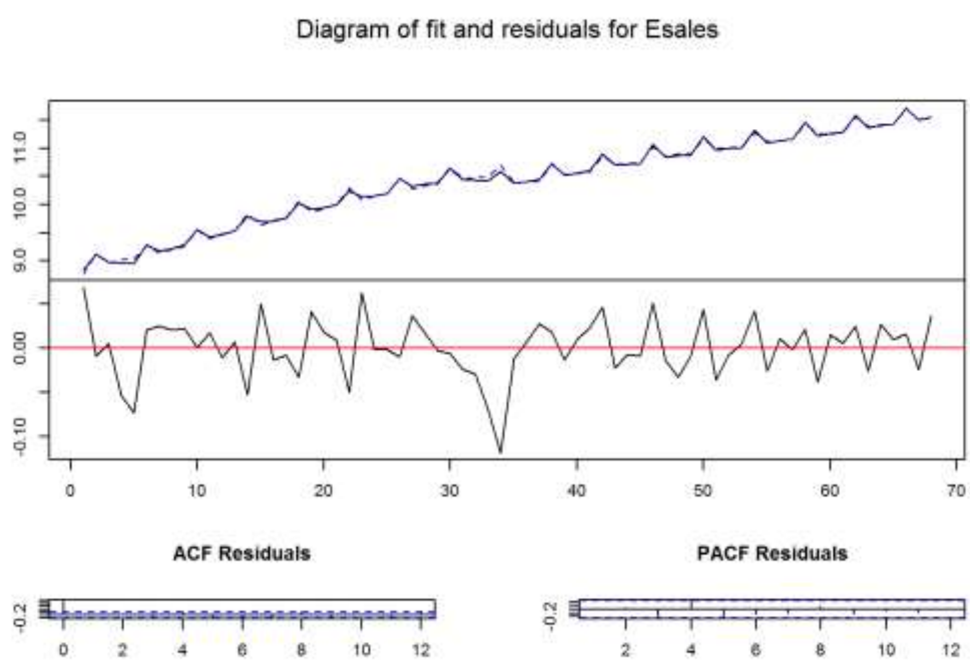

```
plot(var.1, names =  "DPI")
```



```
plot(var.1, names =  "PCE")
```

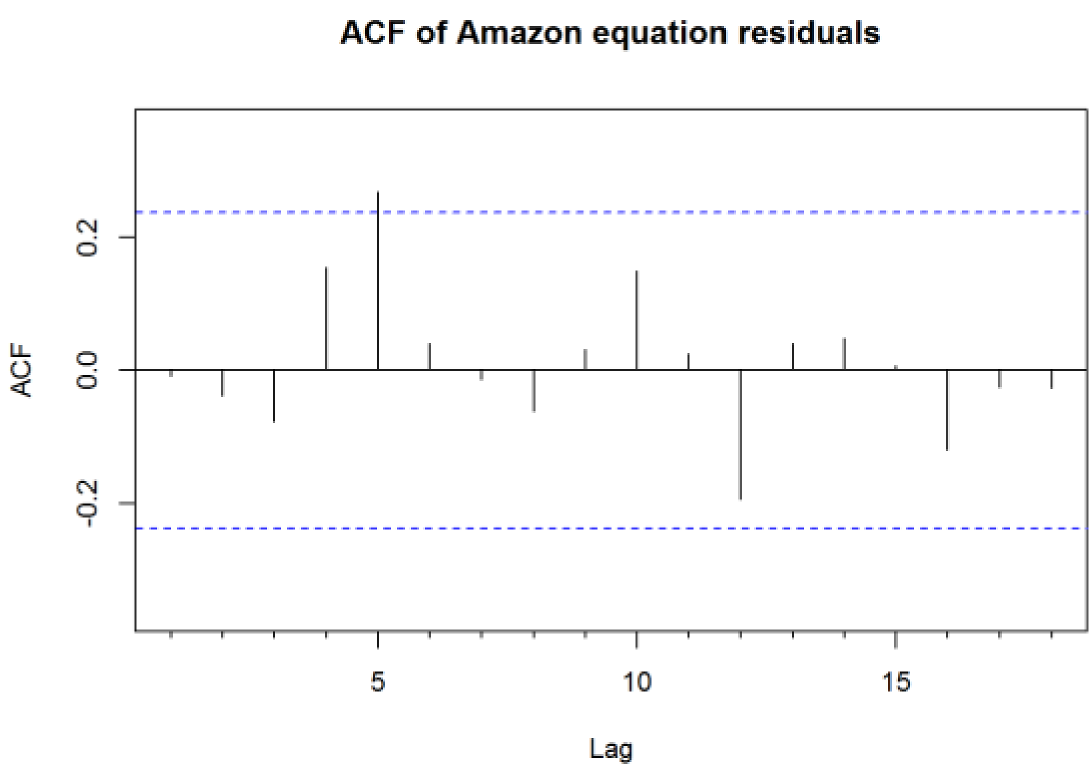



```
plot(var.1, names = "Esales")
```

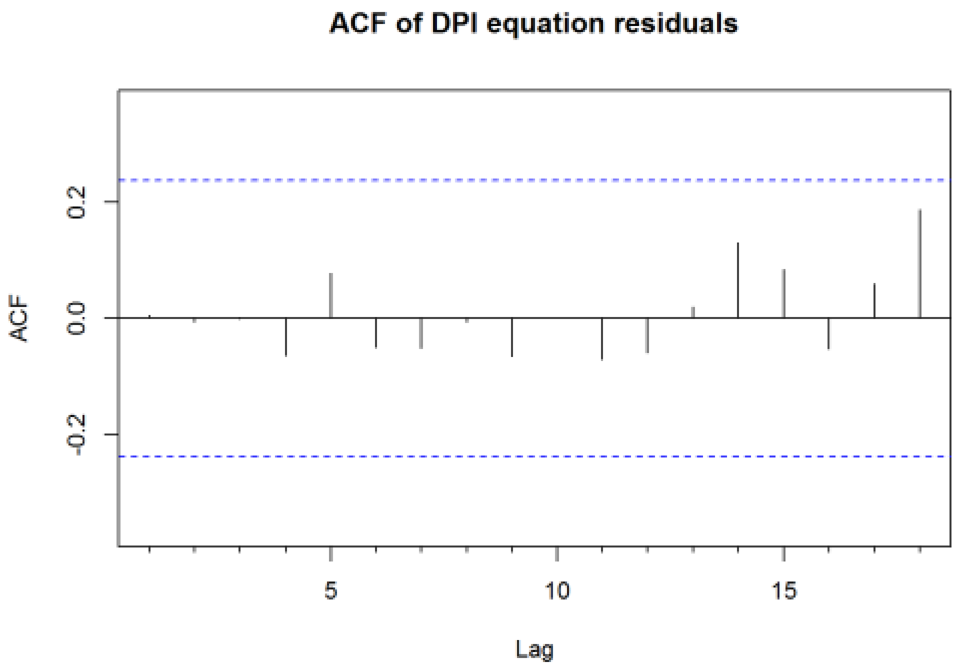


The null hypothesis of no serial correlation in the residuals is not rejected with a p-value slightly larger than 0.05. The ACF residuals of the four variables show that they are random and the autocorrelation is insignificant as there is no significant spike (although there is one small spike at lag=5 for Amazon), suggesting that the model is a good fit.

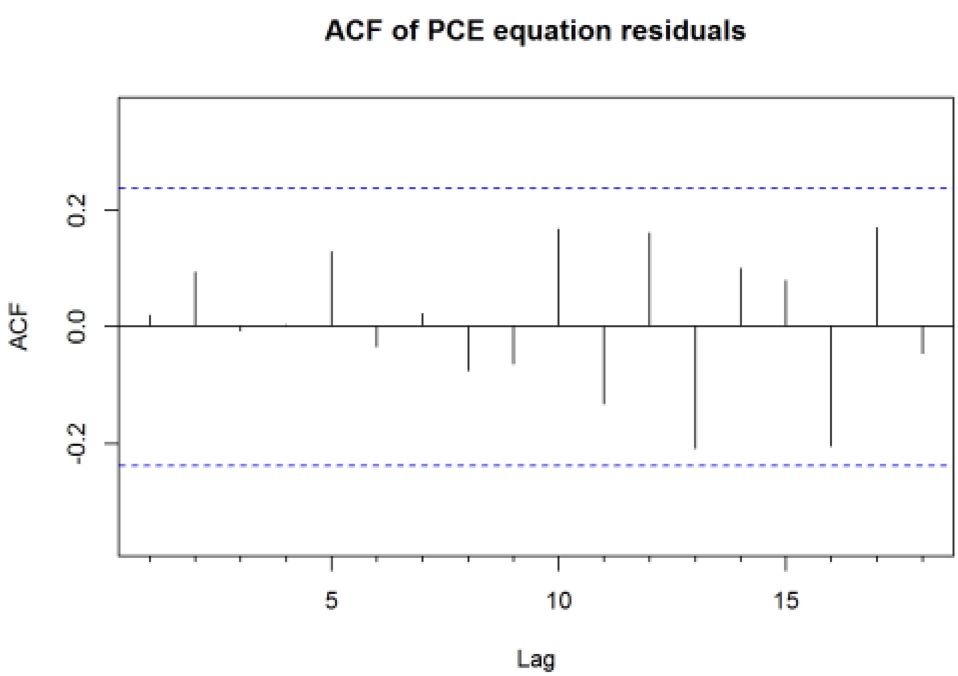
```
serial.test(var.1, lags.pt = 16, type = "PT.adjusted")
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var.1
## Chi-squared = 259.68, df = 224, p-value = 0.05105
Acf(residuals(var.1)[,1], main="ACF of Amazon equation residuals")
```



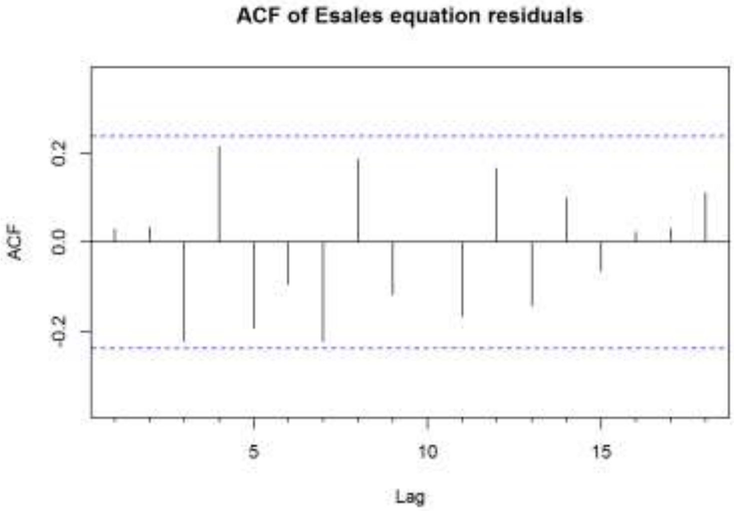

```
Acf(residuals(var.1)[,2], main="ACF of DPI equation residuals")
```




```
Acf(residuals(var.1)[,3], main="ACF of PCE equation residuals")
```



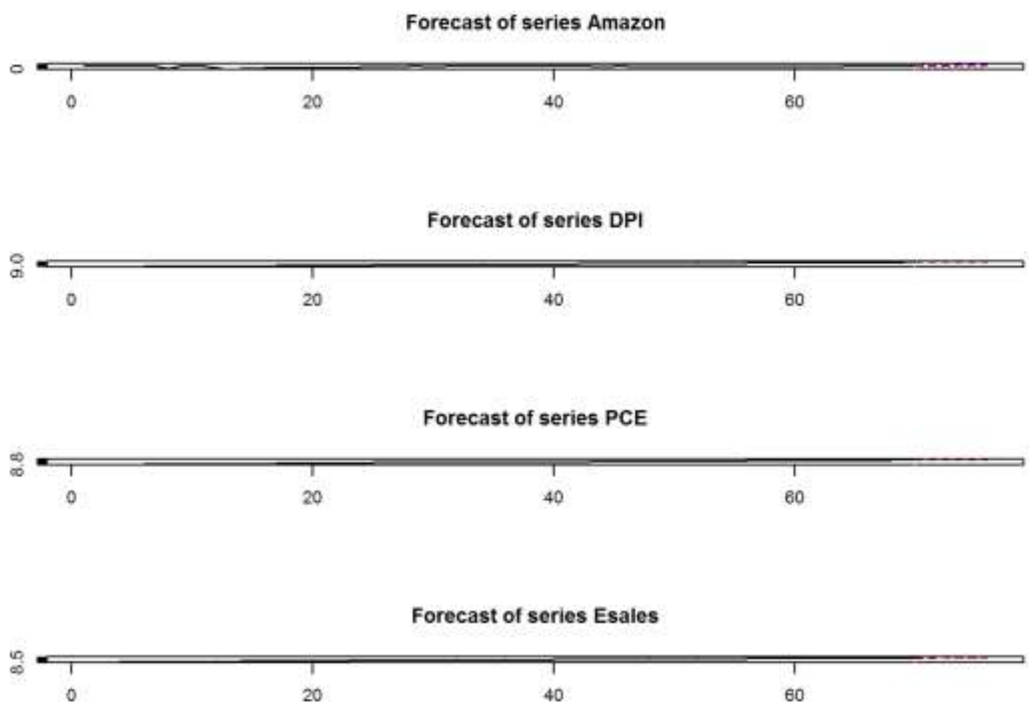

```
Acf(residuals(var.1)[,4], main="ACF of Esales equation residuals")
```



VAR Forecasts

The actual Q3 2017 data is $107,002 million of dollars (obtained from FRED). VAR predicted 105,144.1, which is much lower. Comparing to the forecasts made by log transformed ARIMA, VAR's forecasts seem more conservative.

```
var.fc = predict(var.1, n.ahead= 6)
plot(var.fc)
```



```r
exp(var.fc$fcst$Esales)
```

```
##           fcst     lower    upper       CI
## [1,] 105144.1  97921.17 112899.8 1.073763
## [2,] 136567.1 124547.48 149746.7 1.096506
## [3,] 111670.0  99641.62 125150.4 1.120716
## [4,] 112735.7  98812.04 128621.3 1.140910
## [5,] 113925.0  98257.06 132091.4 1.159459
## [6,] 146884.3 124878.93 172767.2 1.176213
```

Causality

Consumption, income, and Amazon revenues do not granger-cause ecommerce sales based on the results shown below - p-value is larger than 0.05, failing to reject the null hypothesis. However, there is instantaneous causality between consumption, income, Amazon revenues, and ecommerce sales, as indicated by a very small p-value.

```r
causality(var.1, cause= c("Amazon","DPI","PCE"))
```

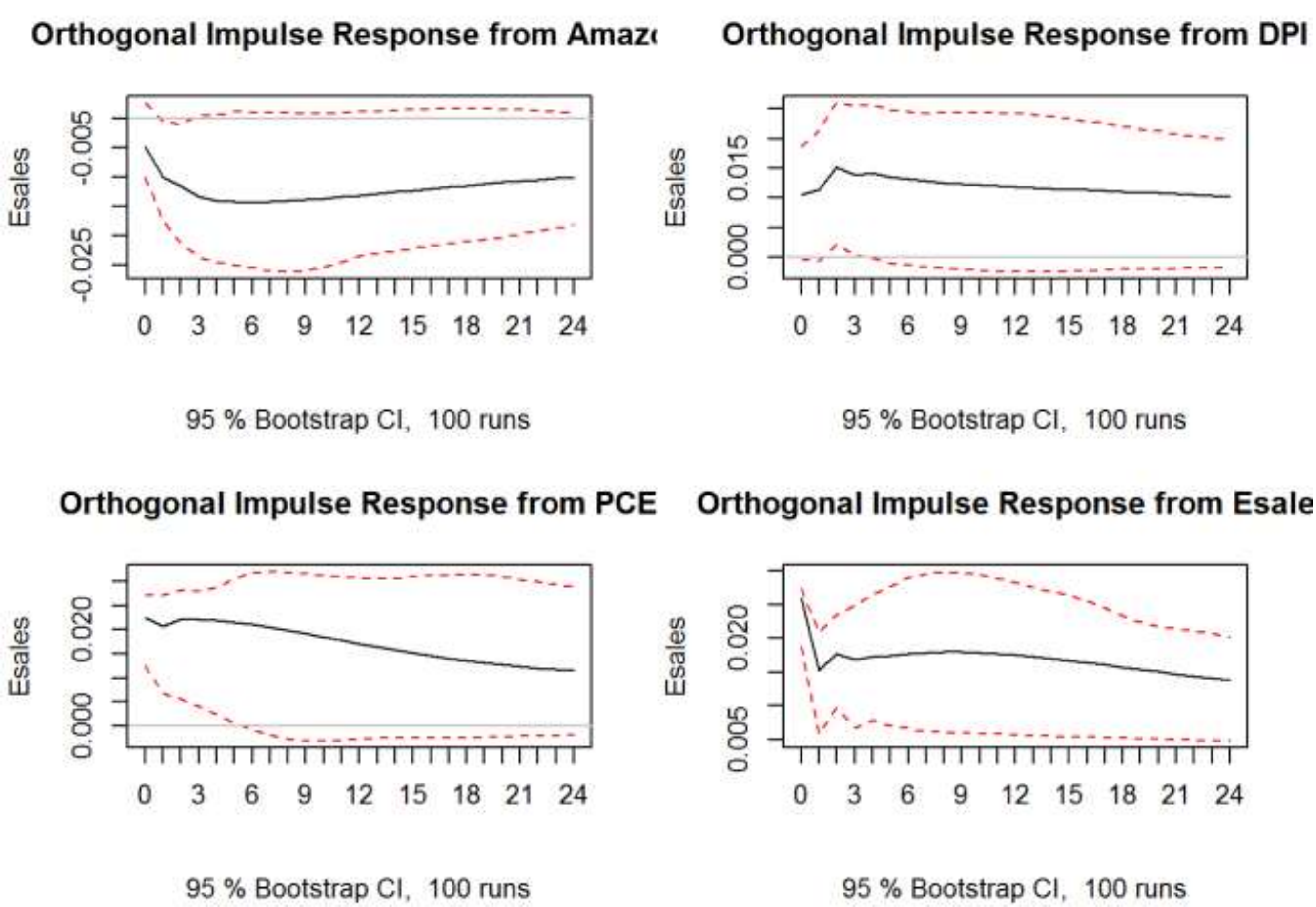
```
## $Granger
##
##  Granger causality H0: Amazon DPI PCE do not Granger-cause Esales
##
## data:  VAR object var.1
## F-Test = 0.90423, df1 = 6, df2 = 224, p-value = 0.4926
##
##
## $Instant
##
##  H0: No instantaneous causality between: Amazon DPI PCE and Esales
##
## data:  VAR object var.1
## Chi-squared = 22.254, df = 3, p-value = 5.775e-05
```

Impulse Response Functions

Next I traced out how a shock to one variable would affect the response of the other variables. We can see that a shock to Amazon's revenue negatively affects the response of ecommerce sales; it is insignificant, however, with the zero inside the 95% confidence bands. A shock to income positively affects the response of ecommerce sales; it is only significant at lag 2. And a shock to consumption leads to a positive response on ecommerce sales, with the confidence bands above zero before Lag 6. The magnitudes of the impulses are all small.

```
par(mfrow=c(2,2))

plot(irf(var.1, response = "Esales", n.ahead = 24, boot = TRUE) , plot.type = "single")
```




```
par(mfrow=c(1,1))
```

Forecast Error Variance Decompositions

Lastly, I computed the forecast error variance decompositions. The variation of Amazon and income are highly explained by themselves. The variation of consumption is mostly explained by itself, with a very small portion being explained by the income. The variation of ecommerce retail sales is explained by both itself and the personal consumption expenditure; for example, at lag=4, a large portion of 42.33% is explained by consumption. This makes sense as ecommerce expenditure is part of the consumption expenditure.

```
fevd(var.1, n.ahead = 16)
```

```
## $Amazon
##          Amazon        DPI        PCE       Esales
##  [1,] 1.0000000 0.00000000 0.00000000 0.0000000000
##  [2,] 0.9618770 0.01754305 0.02035004 0.0002299161
##  [3,] 0.9396097 0.01667716 0.02805145 0.0156616528
##  [4,] 0.9154541 0.01668555 0.03967888 0.0281815156
##  [5,] 0.8920317 0.01644062 0.05377909 0.0377485970
##  [6,] 0.8708834 0.01607398 0.07040152 0.0426410985
##  [7,] 0.8515823 0.01574713 0.08808548 0.0445850779
##  [8,] 0.8341567 0.01556491 0.10546606 0.0448123591
##  [9,] 0.8186916 0.01554163 0.12149889 0.0442678399
## [10,] 0.8052835 0.01563183 0.13555481 0.0435299092
## [11,] 0.7939206 0.01578630 0.14737541 0.0429177154
## [12,] 0.7844992 0.01595933 0.15696659 0.0425748362
## [13,] 0.7768424 0.01611902 0.16450314 0.0425354491
## [14,] 0.7707323 0.01624649 0.17024923 0.0427719442
## [15,] 0.7659364 0.01633425 0.17450241 0.0432269782
## [16,] 0.7622263 0.01638313 0.17755658 0.0438339920
##
## $DPI
##           Amazon       DPI        PCE      Esales
##  [1,] 0.03833722 0.9616628 0.00000000 0.000000000
##  [2,] 0.02835938 0.9182104 0.05157968 0.001850536
##  [3,] 0.03262559 0.8576469 0.10682024 0.002907267
##  [4,] 0.04527822 0.7718915 0.17543467 0.007395586
##  [5,] 0.05890127 0.6863196 0.24003358 0.014745534
##  [6,] 0.07140970 0.6074497 0.29626701 0.024873619
```

```
##  [7,] 0.08231361 0.5394015 0.34151903 0.036765902
##  [8,] 0.09166291 0.4825243 0.37612476 0.049688079
##  [9,] 0.09967360 0.4358320 0.40146466 0.063029740
## [10,] 0.10656520 0.3978196 0.41923581 0.076379425
## [11,] 0.11253089 0.3669672 0.43104685 0.089455065
## [12,] 0.11772654 0.3419295 0.43827409 0.102069841
## [13,] 0.12227577 0.3215848 0.44203871 0.114100669
## [14,] 0.12627604 0.3050248 0.44323003 0.125469138
## [15,] 0.12980491 0.2915230 0.44254319 0.136128902
## [16,] 0.13292514 0.2805008 0.44051647 0.146057626
##
## $PCE
##           Amazon        DPI       PCE      Esales
##  [1,] 0.03694178 0.15795210 0.8051061 0.000000000
##  [2,] 0.02284437 0.22541364 0.7418090 0.009932945
##  [3,] 0.03588912 0.19583572 0.7491021 0.019173076
##  [4,] 0.05325254 0.17498815 0.7429958 0.028763460
##  [5,] 0.06914097 0.15608664 0.7347362 0.040036141
##  [6,] 0.08212536 0.14085012 0.7243757 0.052648779
##  [7,] 0.09257458 0.12873643 0.7125754 0.066113579
##  [8,] 0.10109375 0.11933142 0.6996761 0.079898690
##  [9,] 0.10817922 0.11214343 0.6860700 0.093607330
## [10,] 0.11417898 0.10675189 0.6721163 0.106952809
## [11,] 0.11932850 0.10280994 0.6581213 0.119740264
## [12,] 0.12378968 0.10004109 0.6443290 0.131840224
## [13,] 0.12767792 0.09822579 0.6309246 0.143171719
## [14,] 0.13107945 0.09718837 0.6180419 0.153690266
## [15,] 0.13406183 0.09678620 0.6057723 0.163379708
## [16,] 0.13668025 0.09690155 0.5941721 0.172246084
##
## $Esales
##           Amazon        DPI       PCE    Esales
##  [1,] 0.01788888 0.08341273 0.3851677 0.5135307
##  [2,] 0.05625717 0.10844597 0.4232420 0.4120549
```

```
##  [3,] 0.07604234 0.13922077 0.4220703 0.3626666
##  [4,] 0.09629960 0.14654719 0.4233638 0.3337894
##  [5,] 0.11054697 0.15168607 0.4203615 0.3174055
##  [6,] 0.12142936 0.15284570 0.4173621 0.3083629
##  [7,] 0.12954075 0.15300223 0.4137063 0.3037507
##  [8,] 0.13582897 0.15255218 0.4097417 0.3018771
##  [9,] 0.14081288 0.15202475 0.4054683 0.3016941
## [10,] 0.14486420 0.15158438 0.4009930 0.3025584
## [11,] 0.14821959 0.15132791 0.3963993 0.3040532
## [12,] 0.15104152 0.15128085 0.3917716 0.3059061
## [13,] 0.15344305 0.15144140 0.3871809 0.3079346
## [14,] 0.15550621 0.15179123 0.3826857 0.3100168
## [15,] 0.15729225 0.15230507 0.3783313 0.3120714
## [16,] 0.15884825 0.15295505 0.3741513 0.3140454
```

In conclusion, the three explanatory variables do not cause and interact with ecommerce retail sales in a significant way. It is surprising to see that Amazon’s revenue and ecommerce sales do not affect each other. I think this might be caused by the volatility exhibited by the Amazon’s revenue data; while the ecommerce retail sales overall have been increasing in the past 17 years, Amazon’s business has been much more volatile and it does not have a tremendous influence in the industry as I expected it would. The ecommerce retail sales may include sales from a wide range of companies, and it is possible that it consists mostly of sales from small to medium-sized retailers.