Lab 3

Nishant Velagapudi, Bryan Moore, Morris Burkhardt - Overall

March 19, 2018

E-Commerce Retail Sales as a Percentage of Total Sales

We have a time series that describes the percentage of sales attributed to e-commerce. Modelling this series will require three major steps. First, we will explore and visualize the series to understand potential transformations as well as AR and MA orders. Second, we will find the SARIMA specification with the optimal AIC value. Finally, we will compare model various model specifications (informed either through our EDA or our AutoArima results) on the basis of model performance in a hold-out data set. Data up until 2015 will be used for training, with 2015 and 2016 data being used for model selection. We generate a forecast for 2017 data using the best-performing specification trained over all data available.

Exploration and Visualization

Although it has been dictated that we are to create a SARIMA model, we will use exploratory analysis to show that this is suitable.

```r
# loading the data and making the train-test split
ecom = read.csv(file = "ECOMPCTNSA.csv")
ts_ecom = ts(ecom$ECOMPCTNSA, start = c(1999, 4), frequency = 4)
ts_ecom_train = ts(ts_ecom[time(ts_ecom) < 2015], start = c(1999, 4), freq = 4)
ts_ecom_test = ts(ts_ecom[time(ts_ecom) >= 2015], start = c(2015, 1), freq = 4)
head(ts_ecom_train, 13)
```

```
##      Qtr1 Qtr2 Qtr3 Qtr4
## 1999                 0.7
## 2000  0.8  0.8  0.9  1.1
## 2001  1.1  1.0  1.0  1.3
## 2002  1.3  1.3  1.4  1.7
```

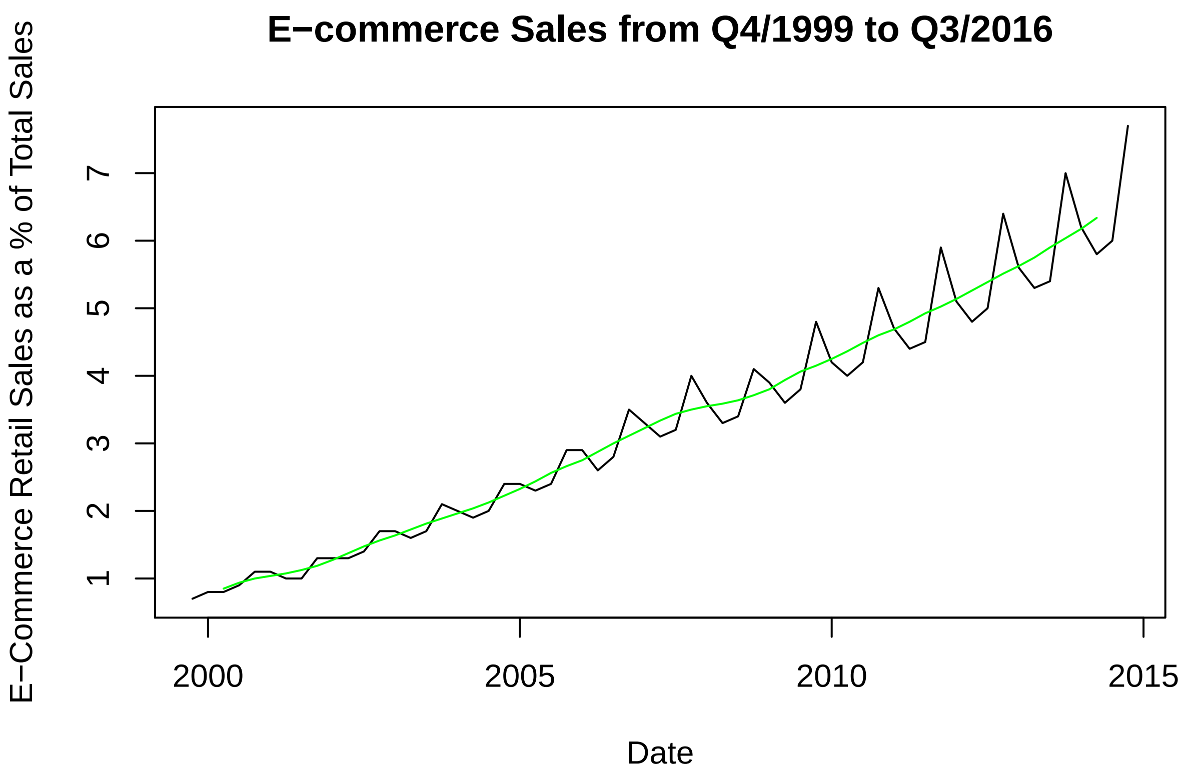
```r
#check for missing values
sum(is.na(ts_ecom_train))
```

```
## [1] 0
```

```r
#summarize the series
summary(ts_ecom_train)
```

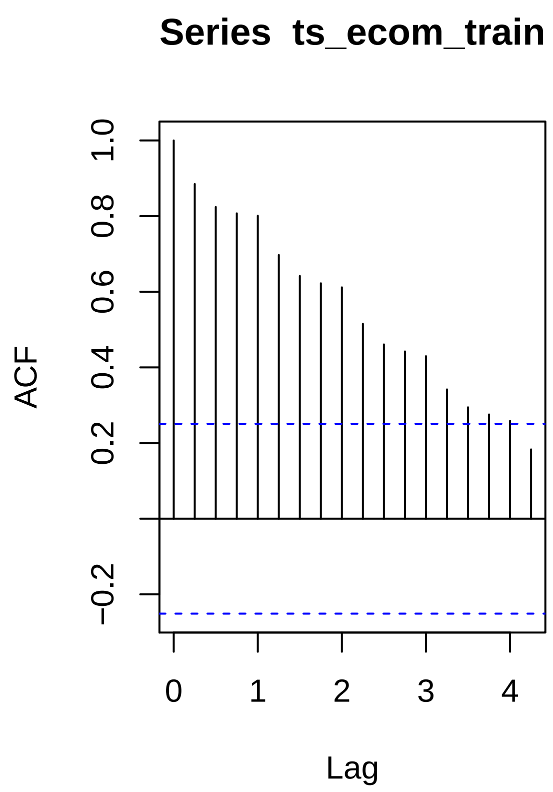
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.700   1.700   3.300   3.331   4.700   7.700
```

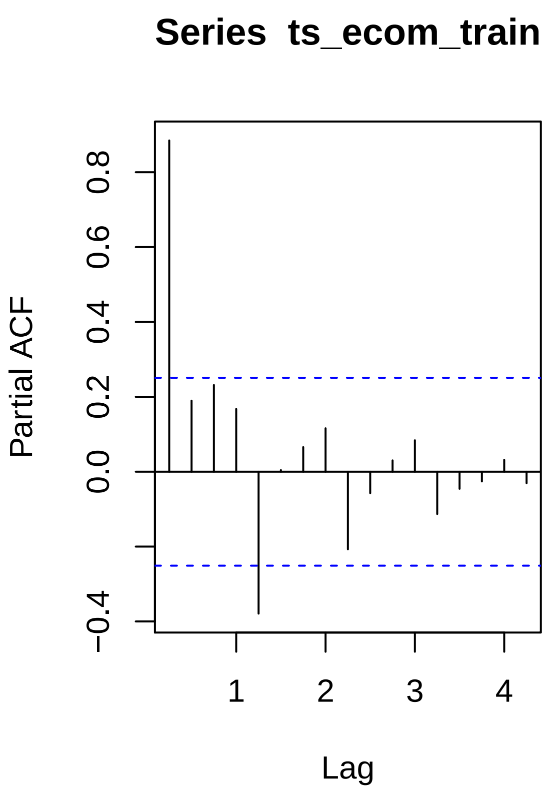
```r
#visualize the series
plot(ts_ecom_train, xlab = 'Date', ylab = 'E-Commerce Retail Sales as a % of Total Sales', main = 'E-co
lines(ma(ts_ecom_train, order = 4, centre = T), col = 'green')
```



We can see that we have no missing values in this series: the series ranges from 0.7 to 9.5 with a mean of 3.8 and median of 3.6. Plotting this series shows both a deterministic trend as well as strong seasonality. A MA(4) model of this series smooths out all variance: reinforcing the observation that we have annual seasonality in this series. We next look at autocorrelation and partial autocorrelation functions of the training series.

```
par(mfrow=c(1,2))
acf(ts_ecom_train)
pacf(ts_ecom_train)
```

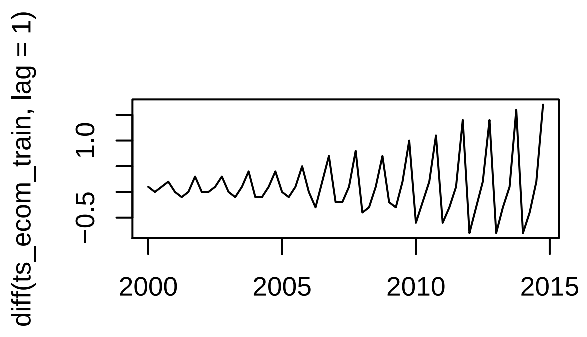


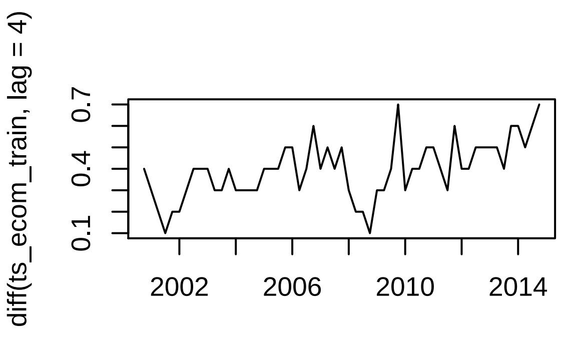


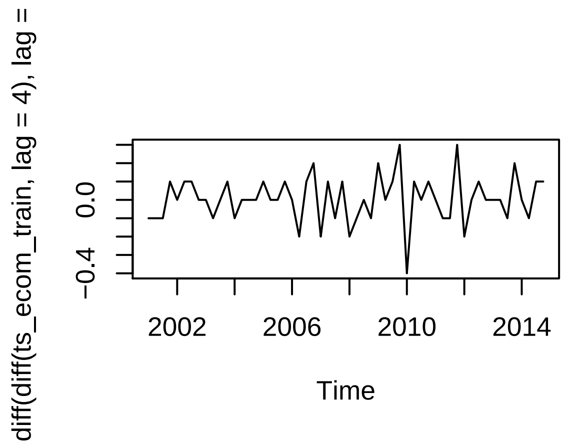
We see that the autocorrelations are significant for a large number of lag (16 quarters), further evidence of non-stationarity. The gradual decay without any spikes at seasonal intervals tells us that we will likely need a non-seasonal AR term p but may not need a seasonal AR term P.

Imposing stationarity will require differencing the series. We take first and fourth order differences as well as the first order difference of the fourth order differences.

```r
par(mfrow=c(2,2))
plot(diff(ts_ecom_train, lag = 1))
plot(diff(ts_ecom_train, lag = 4))
plot(diff(diff(ts_ecom_train, lag = 4),lag=1))
```

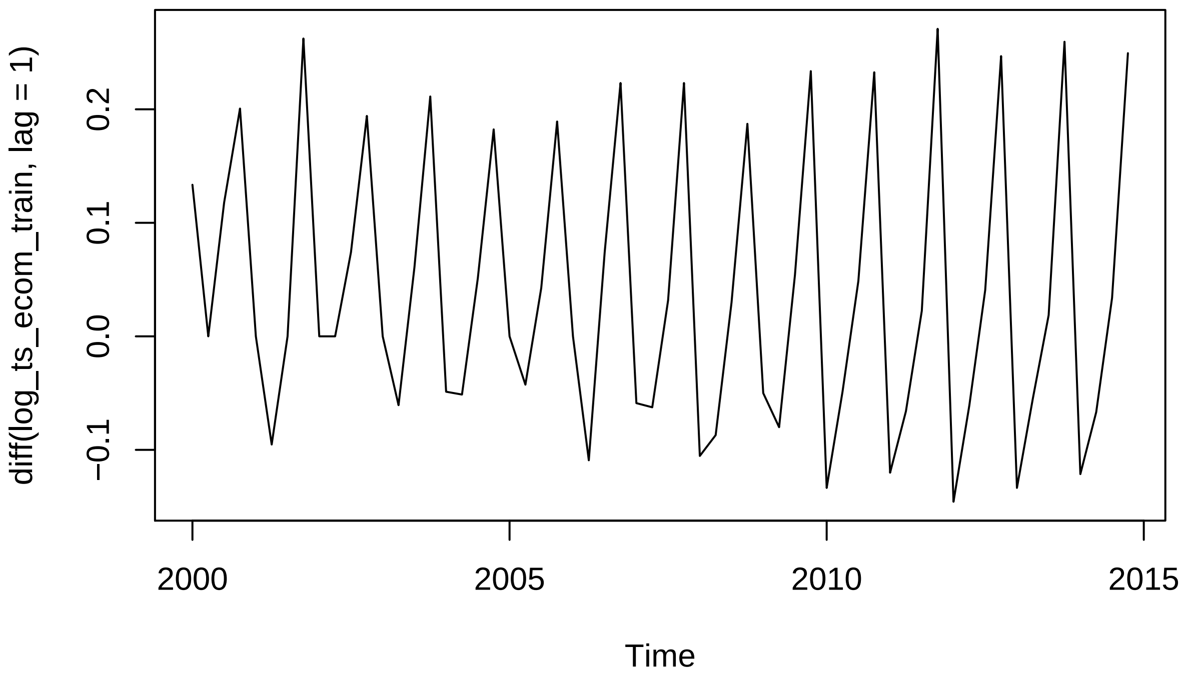




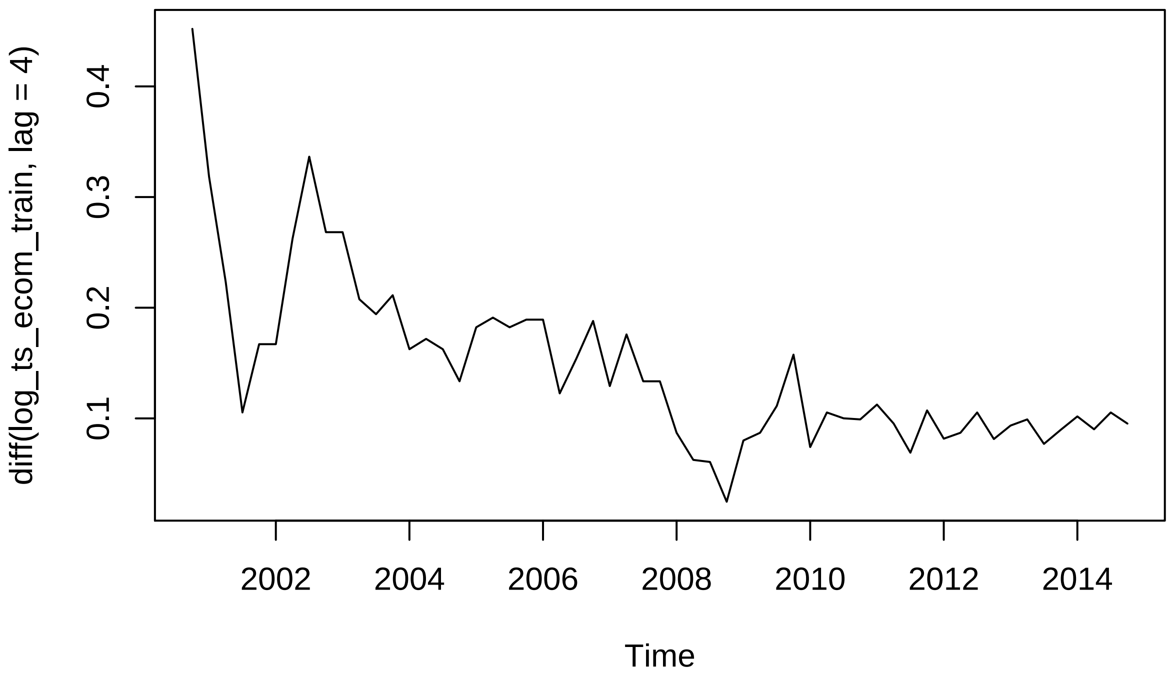


As this series increases in magnitude, the associated seasonal variance also increases, and none of the differences taken above are able to attain visually consistent variance. We thus examine the effect of taking the logarithmic transform of this series prior to differencing.

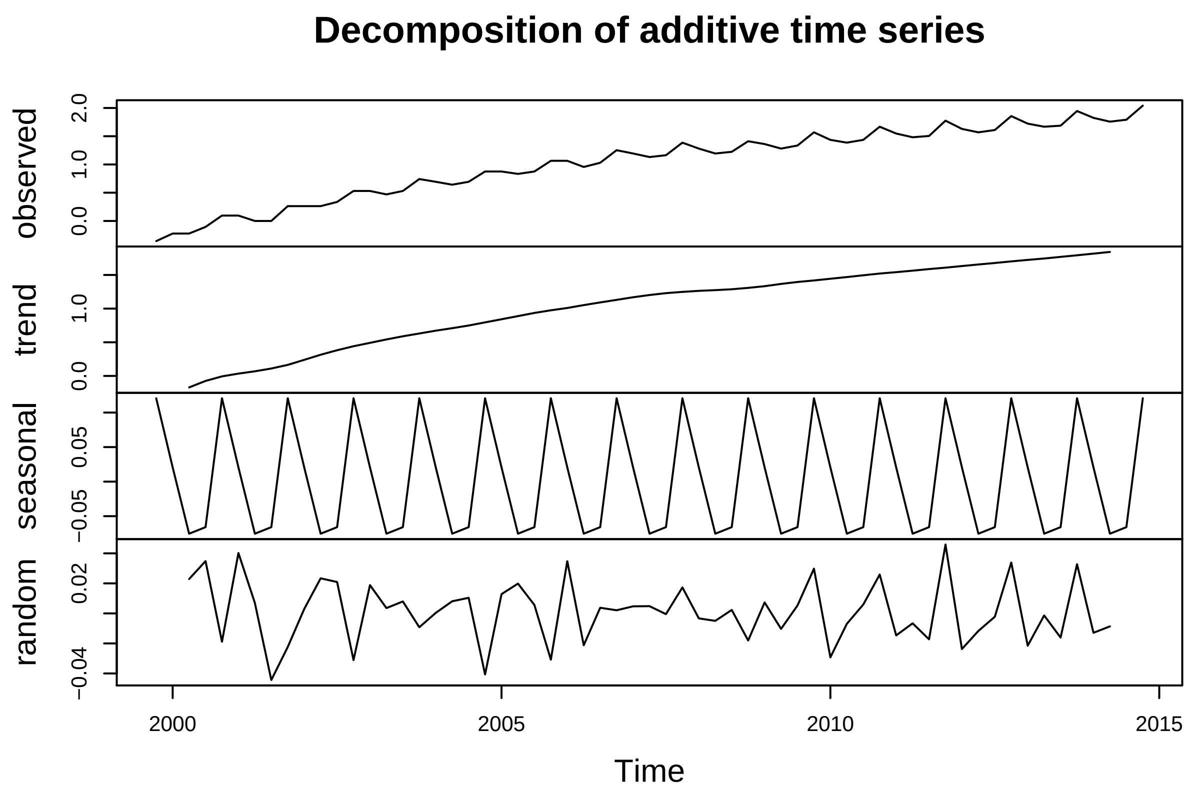
```r
log_ts_ecom_train = log(ts_ecom_train)
log_ts_ecom_test = log(ts_ecom_test)

plot(diff(log_ts_ecom_train, lag = 1))
```



```r
plot(diff(log_ts_ecom_train, lag = 4))
```



```
plot(decompose(log_ts_ecom_train))
```



We can see that the first order difference of the logarithm of the training series has near-constant variance.

Modelling

We will start by using the AutoArima function to identify the SARIMA parameters that minimize AIC. We will explore modelling both the log transformed time series (handled by the argument "lambda=0") and the non-transformed series.

```
log_model_auto <- auto.arima(ts_ecom_train, ic = 'aicc', lambda = 0)
model_auto <- auto.arima(ts_ecom_train, ic = 'aicc')

log_model_auto
```

```
## Series: ts_ecom_train
## ARIMA(0,1,0)(2,1,0)[4]
## Box Cox transformation: lambda= 0
##
## Coefficients:
##          sar1     sar2
##       -0.8012  -0.2491
## s.e.   0.1506   0.1618
##
## sigma^2 estimated as 0.001303:  log likelihood=106.25
## AIC=-206.5   AICc=-206.04   BIC=-200.42
```

```
model_auto
```

```
## Series: ts_ecom_train
## ARIMA(0,1,1)(0,1,0)[4]
##
## Coefficients:
##           ma1
##       -0.5680
## s.e.   0.1581
##
## sigma^2 estimated as 0.01302:  log likelihood=42.4
## AIC=-80.8   AICc=-80.57   BIC=-76.75
```

We can see that log transformed order is (0,1,0)(2,1,0)[4], while the non-transformed order is (0,1,1)(0,1,0).

We will now fit models using these orders as well as a specification purely informed by our previous EDA for both the logarithm transformed series as well as the raw series.

The spike in the ACF at a lag of 1 quarter suggests a nonseasonal MA(1) (q=1) component and the spikes at intervals of 4 quarters of lag suggest a seasonal MA(1) (Q=1). Additionally, the spike at at a lag of 1 quarter in the PACF and the spikes at intervals of 4 quarters of lag tells us that a a nonseasonal AR(1) (p=1) component and a seasonal AR(1) (P=1) component are appropriate for our initial model. Therefore, our initial model will be of the form $ARIMA(1,1,1)(1,1,1)_4$

In the logarithm case, without accounting for seasonality, we observe a somewhat sinusoidal ACF with a PACF function showing no significant spikes beyond lag 4 that has steady variance with a difference of one.

```r
manual_model_raw <- Arima(ts_ecom_train, order=c(1,1,1), seasonal=c(1,1,1))
manual_model_log <- Arima(ts_ecom_train, order = c(0,1,4), seasonal=c(0,0,0), lambda=0)
```

We check each of these four models for validity: we use the Shapiro-Wilk and Box-Ljung tests. The former tests for normality of residuals within the training set, while the latter tests for autocorrelation within these same residuals.

```r
log_mod_auto_sw <- shapiro.test(log_model_auto $resid)
log_mod_auto_box <-Box.test(log_model_auto $resid, type = "Ljung-Box")

mod_auto_sw <-shapiro.test(model_auto $resid)
mod_auto_box <-Box.test(model_auto $resid, type = "Ljung-Box")

man_mod_raw_sw <-shapiro.test(manual_model_raw $resid)
man_mod_raw_box <-Box.test(manual_model_raw $resid, type = "Ljung-Box")

log_man_mod_sw <-shapiro.test(manual_model_log $resid)
log_man_mod_box <-Box.test(manual_model_log $resid, type = "Ljung-Box")

TestResults <- data.frame(cbind(rbind(log_mod_auto_sw$p.value,mod_auto_sw$p.value,man_mod_raw_sw$p.valu

colnames(TestResults) <- c('Shapiro-Wilk', 'Box-Ljung')
rownames(TestResults) <- c('Log, Auto','Raw, Auto','Log, Manual', 'Raw, Manual')

TestResults
```

```
##             Shapiro-Wilk  Box-Ljung
## Log, Auto    0.003334123 0.09474961
## Raw, Auto    0.169793900 0.60900827
## Log, Manual  0.413933643 0.82470762
## Raw, Manual  0.103902258 0.43365721
```

We compare these models on the basis of root mean square error (RMSE) of predictions in the validation holdout. Visuals of predictions versus actuals help us understand the forecasts.

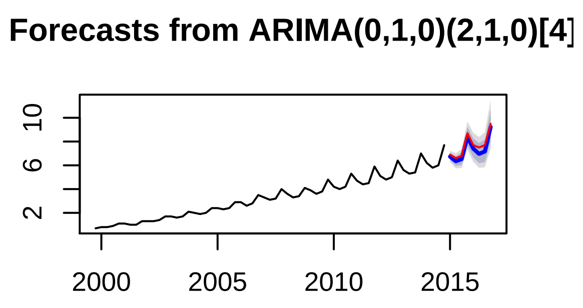
```r
calculate_rmse <- function(fcast, test){
  rmse <- sqrt(mean((fcast - test)^2))
}

par(mfrow=c(2,2))
forecast_log_auto <- forecast(log_model_auto, h = 8, lambda=0)
```

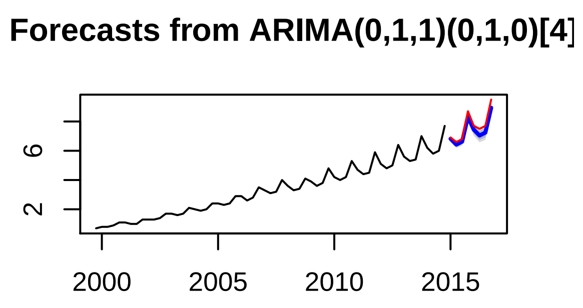
```
## Warning in InvBoxCox(pred$pred, lambda, biasadj, var(residuals(object), :
## biasadj information not found, defaulting to FALSE.
```

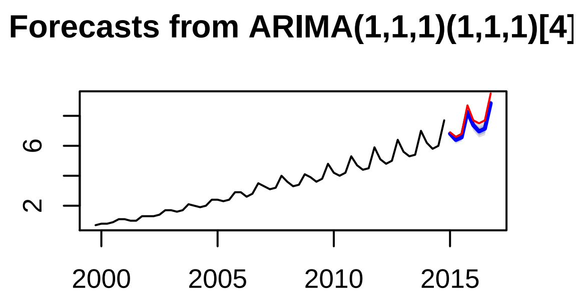
```r
plot(forecast_log_auto)
lines(ts_ecom_test, col='red')

forecast_auto <- forecast(model_auto, h = 8)
plot(forecast_auto)
lines(ts_ecom_test, col='red')

forecast_log_man <- forecast(manual_model_raw, h = 8)
plot(forecast_log_man)
lines(ts_ecom_test, col='red')

forecast_man <- forecast(manual_model_log, h = 8, lambda=0)
```

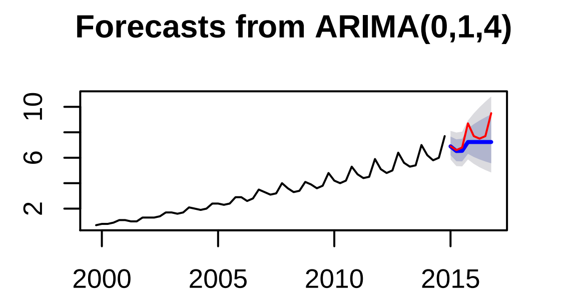
```
## Warning in InvBoxCox(pred$pred, lambda, biasadj, var(residuals(object), :
## biasadj information not found, defaulting to FALSE.
```

```r
plot(forecast_man)
lines(ts_ecom_test, col='red')
```









```r
forecast_df <- data.frame(cbind(forecast_log_auto$mean, forecast_auto$mean, forecast_log_man$mean, fore

print(calculate_rmse(forecast_df$forecast_log_auto.mean, forecast_df$ts_ecom_test))
```

```
## [1] 0.3492261
```

```r
print(calculate_rmse(forecast_df$forecast_auto.mean, forecast_df$ts_ecom_test))
```

```
## [1] 0.3562753
```

```r
print(calculate_rmse(forecast_df$forecast_log_man.mean, forecast_df$ts_ecom_test))
```

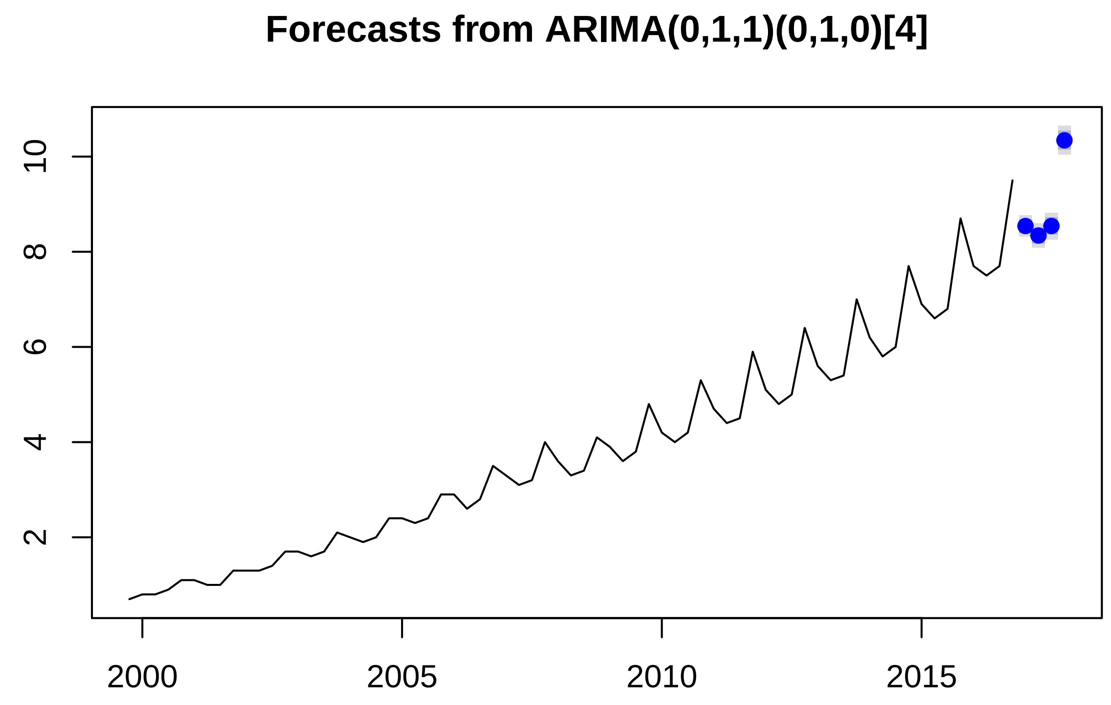
```
## [1] 0.4230131
```

```r
print(calculate_rmse(forecast_df$forecast_man.mean, forecast_df$ts_ecom_test))
```

```
## [1] 0.9901283
```

The AutoArima specification over the log transformed series has the best RMSE in out of sample comparisons. However, this specification also leads to non-normal residuals and potential autocorrelation in the residuals (as identified by the Shapiro-Wilk and Box-Ljung test in the previous section respectively). Thus, we will go forward with the AutoArima specification over the raw series, which has a comparable RMSE and non-significant results in both of these tests. We finally use this model, trained over all available data, to predict values for 2017.

```r
model_auto_all <- auto.arima(ts_ecom, ic = 'aicc')
plot(forecast(model_auto_all,h = 4))
```



# Question 2 ## EDA ## Modelling