W271 - Lab 3

Beau Kramer, Kathryn Papandrew

11/24/2018

Question 1: Forecasting using SARIMA models

1.1 Introduction

The Federal Bank of St. Louis collects data on the E-Commerce Retail Sales as a Percent of Total Sales each quarter. The data collection began at the genesis of online shopping in 1999 and has continued since. After performing cursory raw data checks & any necessary scrubbing, we will conduct a thorough Exploratory Time Series Data Analysis (ETSDA) on all data and perform requisite data transformations to move to modeling. Then, using data from 1999 to the end of 2014, we will fit a model for percentage of e-commerce retail sales over total sales, test the model using the 2015 and 2016 data, and ultimately forecast what the figures would look like for 2017.

1.2 Collect & Cleanse Data

```
# Read in raw data

raw.df <- read.csv('https://docs.google.com/spreadsheets/d/e/2PACX-1vQDzamLw2GdlS5aj3Pb3Sip8Iv
```

```
describe(raw.df)
```

```
## raw.df
##
##  2  Variables      69  Observations
## ---------------------------------------------------------------------------
## DATE
##        n  missing distinct
##       69        0       69
##
## lowest : 1999-10-01 2000-01-01 2000-04-01 2000-07-01 2000-10-01
## highest: 2015-10-01 2016-01-01 2016-04-01 2016-07-01 2016-10-01
## ---------------------------------------------------------------------------
## ECOMPCTNSA
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       69        0       50        1    3.835    2.524     0.94     1.10
##      .25      .50      .75      .90      .95
##     2.00     3.60     5.30     6.92     7.70
##
## lowest : 0.7 0.8 0.9 1.0 1.1, highest: 7.0 7.5 7.7 8.7 9.5
## ---------------------------------------------------------------------------
```

This summary reveals that the data is recorded quarterly between October 1999 and October 2016 (17 years). All quarters in that time range are accounted for and intuitively, the range of numbers makes sense. For example, there are no figures greater than 100 (i.e. greater than 100% of total sales) as that would not make sense. We do not see the need to cleanse/scrub the raw data ourselves, so we will continue into the Exploratory Time Series Data Analysis (ETSDA).

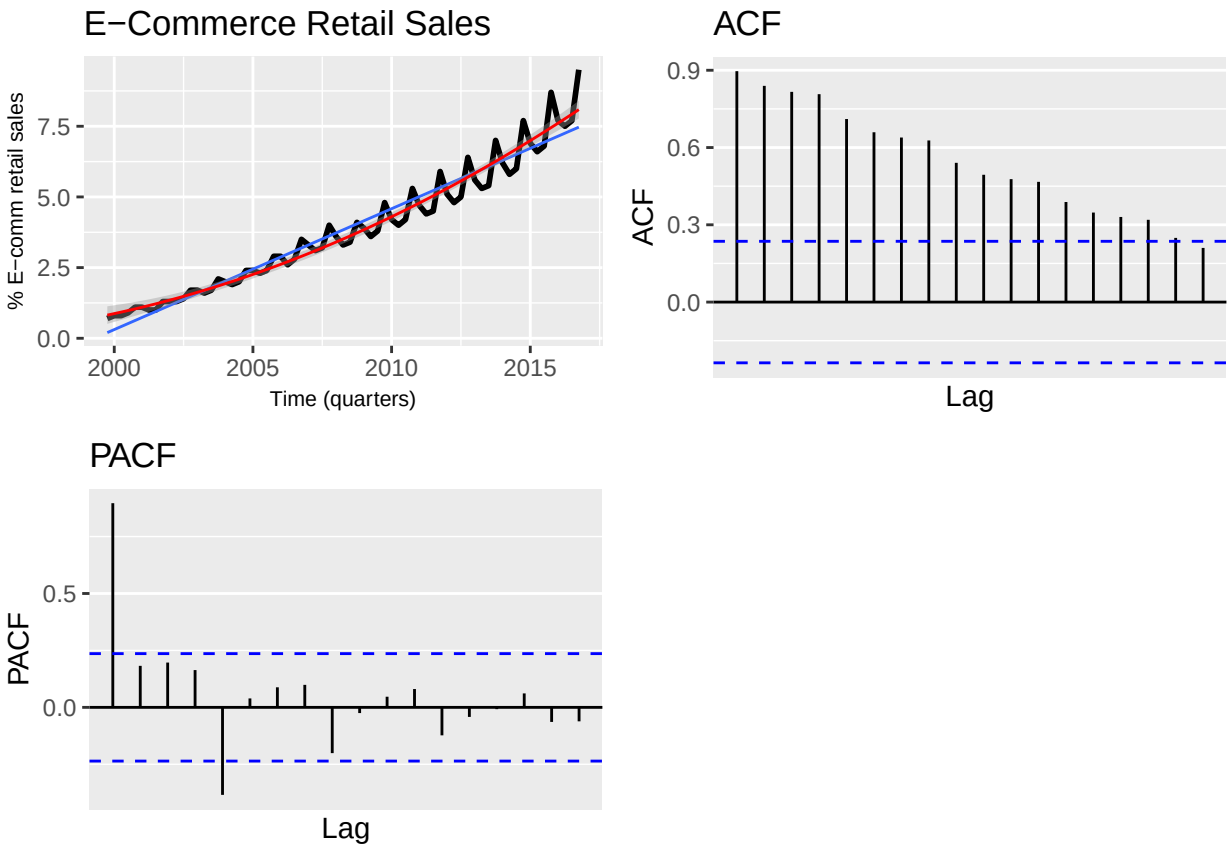
1.3 Exploratory Time Series Data Analysis (ETSDA)

Conduct exploratory time series data analysis (ETSDA) by plotting the series, and examine the main patterns and atypical observations of the graph, after collecting and cleaning the data

1.3.1 Trend Analysis

In this section, we will be examining any trend and fluctuations around the trend. We plot the data over time, the autocorrelation function (ACF), and partial autocorrelation function (PACF) to guide our analysis of trend.

```
# Load data into time series object

sales <- ts(raw.df[,2], frequency=4, start=c(1999,4))

# Plot 1 - E-Commerce Retail Sales over Time (quarterly)
p1 = ggplot(sales, aes(x=time(sales), y=sales)) +
    geom_line(colour = "black", size = 1) +
    ggtitle("E-Commerce Retail Sales") +
    theme(axis.title = element_text(size = rel(0.75))) +
    geom_smooth(method = "lm", se=FALSE, size = 0.5) +   # Linear
    stat_smooth(method = "lm", formula = y ~ x + I(x^2), size = 0.5, color = "red") + # Quadra
    xlab("Time (quarters)") +
    ylab("% E-comm retail sales")


# Plot 2 - Sales Autocorrelation Function
p2 = autoplot(acf(sales, plot = FALSE)) +
  ggtitle("ACF")

# Plot 3 - Sales Partial Autocorrelation Function
p3 = autoplot(pacf(sales, plot = FALSE)) +
  ggtitle("PACF")

grid.arrange(p1, p2, p3, ncol=2)
```



The top left graph (proportion of e-commerce sales over time) clearly indicates an upward trend over time. In blue, we have plotted the linear trend and in red, the quadratic trend. This trend can also be seen in the slow decay in the ACF correlogram. Additionally, we see that the variance is directly proportional with time; the variance increases as time does.

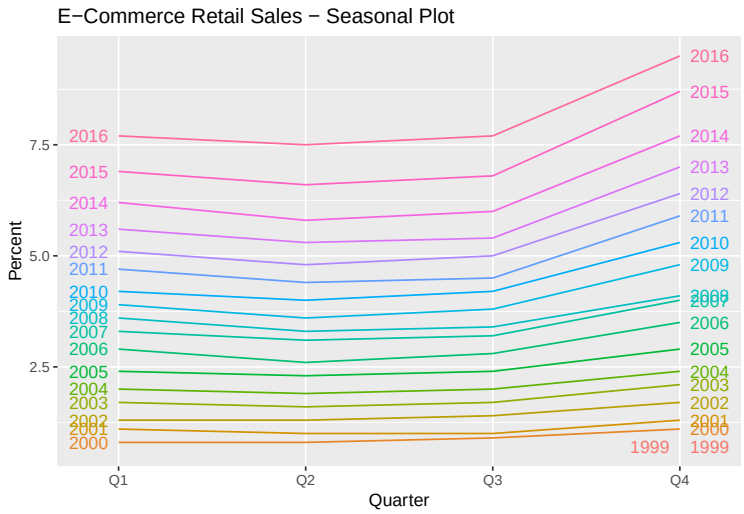
From the e-commerce sales over time and the PACF, we see that there appears to be seasonality, especially given the significance at lag = 4 and the shape sinusoidal decay where there are spikes every four (4) periods. We can also see hint of seasonality in the ACF plot where there are little increases in correlogram every four (4) periods, as well. The presence of seasonality will be confirmed in next section (`1.3.2 Seasonal Variation (or Seasonality)`)

Finally, with an eye toward modeling, we see that the ACF shows a typical autocorrelation structure and the original series may be modeled with an autoregressive process. However, we still must address trend and seasonality in the data which would change this initial observation.

1.3.2 Seasonal Variation (or Seasonality)

This section focuses on testing for seasonality in the data. Below, we use a season plot to show how the e-commerce sales slope over each quarter by year.

```
ggseasonplot(sales, year.labels=TRUE, year.labels.left=TRUE) +
  ylab("Percent") +
  ggtitle("E-Commerce Retail Sales - Seasonal Plot")
```



There seems to be seasonality in that between quarter 3 and quarter 4, the proportion of e-commerce go up and then slowly decline through to quarter 1.

1.3.3 Cyclical variations

Based on the e-commerce retail sales over time figure in `1.3.1 Trend Analysis`, we see no cyclical variation detected beyond the seasonal variation (quarter 2 of each year).

1.3.4 Sharp change in behavior

Based on the e-commerce retail sales over time figure in `1.3.1 Trend Analysis`, there appears to be no structural changes or jumps in the data.

1.3.5 Outliers

Based on raw data analysis and the e-commerce retail sales over time figure in `1.3.1 Trend Analysis`, there appear to be no outliers in the data.

1.4 Stationarity

This section focuses on analyzing the stationarity of the data, including taking into account the trend and seasonality aspects found in Section `1.3 Exploratory Time Series Data Analysis (ETSDA)` by decomposing the data using the multiplicative model as the trend increases over time. In the figures below, the top graph shows the random component over time, the bottom left shows the autocorrelation function (ACF), and bottom right shows the partial autocorrelation function (PACF).

```
sales.decom <- decompose(sales, "multiplicative")
random.component <- sales.decom$random[!is.na(sales.decom$random)]
random.component %>% ggtsdisplay(main = "E-Commerce Retail Sales - Decomposed Data")
```

A visual inspection of the random component in the top graph shows that it is not quite white noise because the variance changes over time. In addition, the ACF should have only one significant lag at zero and the PACF should have no significant lags. Instead, we see that both have several significant lags. We should consider transforming the data so that it is closer to white noise.

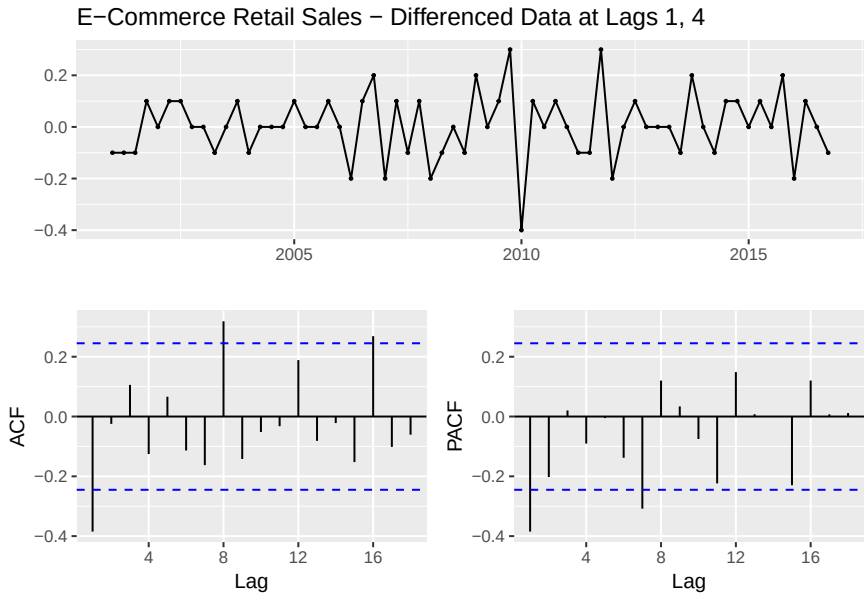
1.5 Transformations to Stationarity

Because we see above that the data is not stationary (the ACF and PACF have multiple significant lags), we decide to perform transformations to try to acheive stationarity in the data to proceed with model building.

1.5.1 Differencing

We experimented with several differences and found that a lag of 1 combined with a seasonal lag of 4 produced the process closest to stationarity. This differenced process appears to be closer to a moving average process than an autoregressive one. We chose these numbers based on the trend and significant lag at 4 in the PACF observed in `1.3.1 Trend Analysis`. The plots below display the random component over time, the ACF, and the PACF for the differenced data.

```
sales %>% diff(lag=1) %>% diff(lag=4) %>% ggtsdisplay(main = "E-Commerce Retail Sales - Differe
```



We use the following takeaways from our EDA to guide our modeling. The initial series has a trend and annual seasonality. It appears to be autoregressive. By differencing with a lag of 1 and 4 we detrend the series and remove the seasonality. After differencing the series appears to be a moving average process. Given these observations we would expect a model of the form $ARIMA(0,1,q)(0,1,0)_4$ to be best model. In the next section, we will determine the best value for p and q in the non-seasonal term which will be used for our model.

1.6 Modeling

1.6.3 Seasonal ARIMA (SARIMA) Model

We now subset the data so we can have one set of data to fit the model and another set to test. We exclude all data for 2015 and 2016 and fit the model with the remaining, earlier data.

```
sales.train <- window(sales, end=c(2014,4))
sales.test <- window(sales, start=c(2015,1), end=c(2016, 4))
```

```
for (q in 0:3){
      for(p in 0:3){
          mod <- Arima(sales.train, order = c(p,1,q),
                       seasonal = list(order = c(0,1,0),4),
                       method = "ML")
          print(c(p, q, mod$aic, mod$bic))
      }
    }
```

```
## [1]   0.00000   0.00000 -71.02447 -68.99912
## [1]   1.0000   0.0000 -76.8255 -72.7748
## [1]   2.00000   0.00000 -78.26241 -72.18636
## [1]   3.00000   0.00000 -76.28983 -68.18843
## [1]   0.00000   1.00000 -80.79782 -76.74711
```

```
## [1]   1.00000   1.00000 -80.38921 -74.31316
## [1]   2.00000   1.00000 -78.70735 -70.60594
## [1]   3.00000   1.00000 -77.12660 -66.99985
## [1]   0.00000   2.00000 -79.95362 -73.87756
## [1]   1.00000   2.00000 -78.80362 -70.70221
## [1]   2.00000   2.00000 -76.38921 -66.26245
## [1]   3.00000   2.00000 -75.59927 -63.44716
## [1]   0.00000   3.00000 -78.10283 -70.00142
## [1]   1.00000   3.00000 -76.81611 -66.68935
## [1]   2.00000   3.00000 -75.00099 -62.84888
## [1]   3.00000   3.00000 -76.66482 -62.48736
```

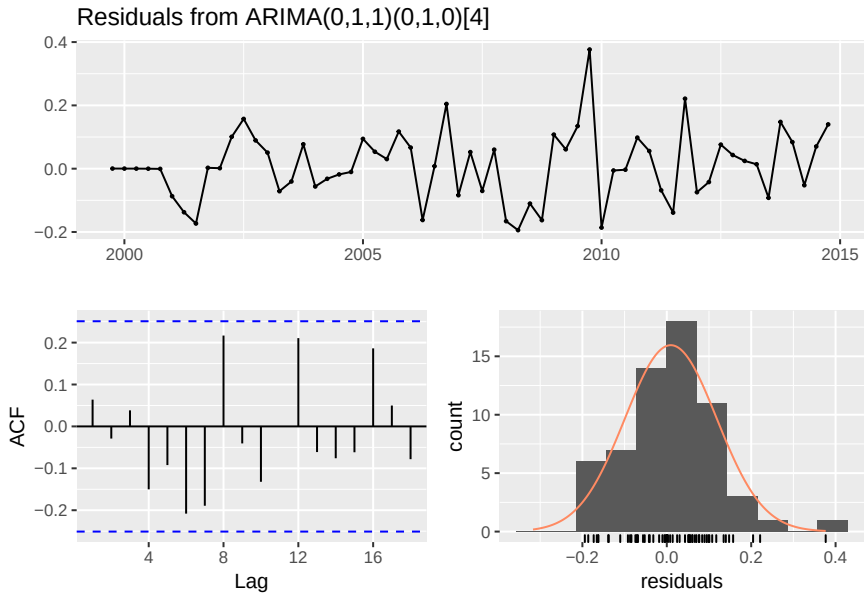
```
# AIC and BIC minimized with Arima(0,1,1)
```

We considered values for p and q from 0 through 10 and show 0 through 3 here for brevity. The best model appears to be $ARIMA(0,1,1)(0,1,0)_4$ with an AIC of *-80.79* and a BIC of *-76.75*. We will now examine this model in more detail below.

```
model <- Arima(sales.train, order = c(0,1,1),
                            seasonal = list(order = c(0,1,0),4),
                            method = "ML")
summary(model)
```

```
## Series: sales.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
##
## Training set error measures:
##                       ME      RMSE        MAE        MPE     MAPE
## Training set 0.009538021 0.1083595 0.08296349 -0.1336688 2.969044
##                    MASE       ACF1
## Training set 0.2101742 0.06391302
```

```
checkresiduals(model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1)(0,1,0)[4]
## Q* = 11.505, df = 7, p-value = 0.118
##
## Model df: 1.   Total lags used: 8
```

The equation for this model is $(1 - B^4)(1 - B)x_t = (1 + \phi_1 B)\omega_t$, or in the fully-expanded form $x_t = x_{t-1} + x_{t-4} - x_{t-5} + \omega_t + \phi_1\omega_{t-1}$ where $\phi_1 = -0.5680$.

The single coefficient, ϕ_1 appears significant.

The residuals appear to be white noise and the ACF of the residuals shows no significant lags. The Ljung-Box test for the model has a p-value of 0.118. Thus we fail to reject the null hypothesis that the data are identically distributed.

1.6.3.1 Underlying Model Assumption Validation

Our model is confirmed non-stationary by looking at the equation. Because there exists a term $(1 - B)$ (as well as term $(1 - B^4)$), there is a unit root $= 1$ hence not exceeding unity.

1.7 Statistical Inference and Forecasting

With our model estimated, we are now ready to test our model on out-of-sample data. We forecast and compare against the held out data for 2015 and 2016. Then, we generate quarterly forecasts for 2017.

```
forecast.os <- forecast(model, h=8)
accuracy(forecast.os, sales.test)[,c(2,3,6)]
```

```
##                      RMSE        MAE      MASE
## Training set 0.1083595 0.08296349 0.2101742
## Test set     0.3562760 0.31920725 0.8086584
```

1.7.1 Model Performance

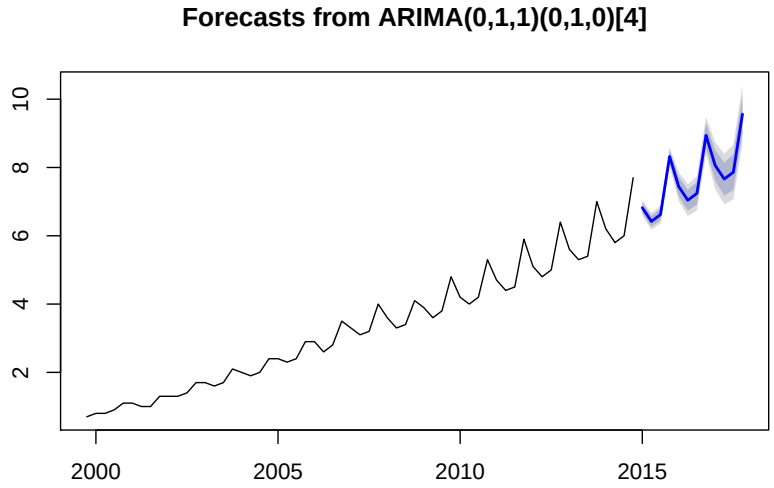
In sample, the model has a root mean squared error of 0.1084 and a mean absolute error of 0.0830. For scaled errors the model has a mean absolute scaled error of 0.2102. That this value is less than 1 is encouraging because it implies that the model's predictive power is better the naive method assumed in the denominator of the measure.

Out of sample, the model's forecast has a root mean squared error of 0.3563 and a mean absolute error of 0.3192. These are both worse than the in sample figures. This is expected because the in sample data is used to fit the model. The out of sample forecast has a mean absolute scaled error of 0.8087. Again, this is worse than the in sample performance. However, it is also less than 1 so the model's out of sample performance is superior to the naive model.

1.7.2 2017 Forecast

We now generate quarterly forecasts for 2017.

```
forecast.17 <- forecast(model, h=12)
plot(forecast.17)
```



Question 2: xts demonstration

2.1 Read AMAZ.csv and UMCSENT.csv into R as R DataFrames

```
# Read in AMZN data
amaz.df <- read.csv("https://docs.google.com/spreadsheets/d/e/2PACX-1vQmSApFqLbQhHFmKZREXZghhb

# Read in UMCSENT data
umcsent.df <- read.csv('https://docs.google.com/spreadsheets/d/e/2PACX-1vTYvnhW-AxvEiVcoFCV9lP
```

2.2 Convert them to xts objects

```
amaz.idx <- amaz.df$Index
amaz.df <- amaz.df[,2:6]

umcsent.idx <- umcsent.df$Index
umcsent.df <- umcsent.df[,2]
```

```
amaz  <- xts(amaz.df, order.by = as.Date(amaz.idx))
umcsent <- xts(umcsent.df, order.by = as.Date(umcsent.idx))
```

2.3 Merge the two set of series together, perserving all of the obserbvations in both set of series

```
merged_data <- merge(amaz, umcsent, join = "outer", fill = 9999)
merged_data['2007-03-29/2007-04-05']
```

```
##            AMAZ.Open AMAZ.High AMAZ.Low AMAZ.Close AMAZ.Volume umcsent
## 2007-03-29      22.4      22.8     22.4       22.8         381  9999.0
## 2007-03-30      22.8      22.8     22.8       22.8         131  9999.0
## 2007-04-01    9999.0    9999.0   9999.0     9999.0        9999    87.1
## 2007-04-04      23.2      23.2     22.8       22.8         190  9999.0
## 2007-04-05      22.8      22.8     22.4       22.4         138  9999.0
```

Note: The insctructions below suggest that these operations only be applied to the UMCSENT portion of the merged dataset. We have proceeded with this interpreation and show only the effects on the UMCSENT series.

2.3.1 fill all of the missing values of the UMCSENT series with -9999

```
UMCSENT01 <- merged_data$umcsent
UMCSENT01[UMCSENT01 == 9999] <- -9999
UMCSENT01['2007-03-29/2007-04-05']
```

```
##            umcsent
```

```
## 2007-03-29 -9999.0
## 2007-03-30 -9999.0
## 2007-04-01    87.1
## 2007-04-04 -9999.0
## 2007-04-05 -9999.0
```

2.3.2 then create a new series, named UMCSENT02, from the original UMCSENT series replace all of the -9999 with NAs

```
UMCSENT02 <- UMCSENT01
UMCSENT02[UMCSENT02 <= -9999] <- NA
UMCSENT02['2007-03-29/2007-04-05']
```

```
##            umcsent
## 2007-03-29      NA
## 2007-03-30      NA
## 2007-04-01    87.1
## 2007-04-04      NA
## 2007-04-05      NA
```

2.3.3 then create a new series, named UMCSENT03, and replace the NAs with the last observation

```
UMCSENT03 <- na.locf(UMCSENT02, na.rm = FALSE, fromLast = FALSE)
UMCSENT03['2007-03-29/2007-04-05']
```

```
##            umcsent
## 2007-03-29    88.4
## 2007-03-30    88.4
## 2007-04-01    87.1
## 2007-04-04    87.1
## 2007-04-05    87.1
```

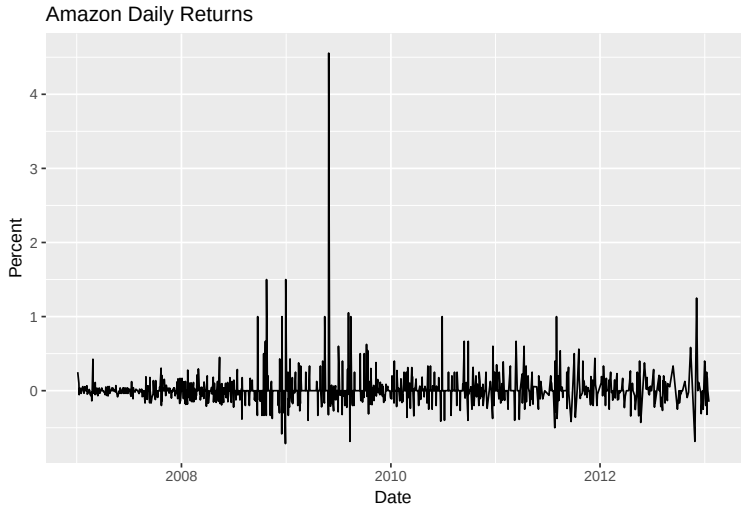
2.3.4 then create a new series, named UMCSENT04, and replace the NAs using linear interpolation.

```
UMCSENT04 <- na.approx(UMCSENT02)
UMCSENT04['2007-03-29/2007-04-05']
```

```
##             umcsent
## 2007-03-29 87.22581
## 2007-03-30 87.18387
## 2007-04-01 87.10000
## 2007-04-04 87.22000
## 2007-04-05 87.26000
```

2.4 Calculate the daily return of the Amazon closing price (AMAZ.close), where daily return is defined as $(x(t) - x(t-1))/x(t-1)$. Plot the daily return series.

```
amaz.ret <- diff(amaz$AMAZ.Close, lag = 1, difference = 1, log = FALSE, na.pad = TRUE)/lag(amaz
amaz.ret <- xts(amaz.ret, order.by = as.Date(amaz.idx))
autoplot(amaz.ret) + ggtitle("Amazon Daily Returns") + xlab("Date") + ylab("Percent")
```



2.5 Create a 20-day and a 50-day rolling mean series from the AMAZ.close series.

```
amaz20ma <- rollapply(amaz$AMAZ.Close, 20, FUN = mean, na.rm = TRUE)
amaz50ma <- rollapply(amaz$AMAZ.Close, 50, FUN = mean, na.rm = TRUE)
```

```
autoplot(ts(cbind(amaz$AMAZ.Close,amaz20ma, amaz50ma) , start = c(2007,1,3), frequency = 252),
scale_colour_discrete(labels=c("AMAZ Close","AMAZ 20-day MA","AMAZ 50-day MA"))
```