Data 624 - Homework Part 1

Libraries used

library(fma)  
library(ggfortify)  
library(tidyverse)  
library(psych)  
library(knitr)

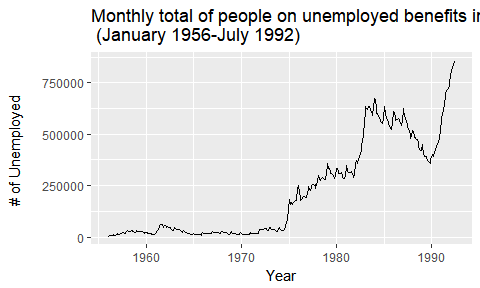
## Week 1

2.1 For each of the following series (from the fma package), make a graph of the data. If transforming seems appropriate, do so and describe the effect.

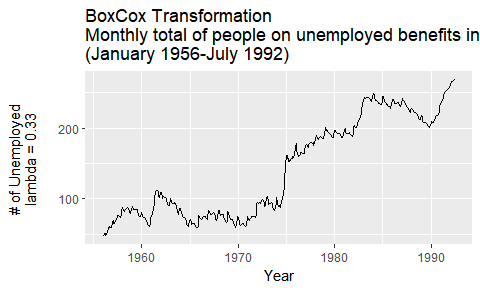
1. Monthly total of people on unemployed benefits in Australia (January 1956-July 1992).

The monthly total of people is graphed below. Caution should be exercised when plotting # totals as it doesn't account for the proportional change. In this case, it could be that the population of Australia also grew and the change in total unemployed is not as dramatic as the chart suggests.

library(fma)  
library(ggfortify)  
data(dole)  
autoplot(dole, xlab = "Year",   
 ylab = "# of Unemployed \n",   
 main = "Monthly total of people on unemployed benefits in Australia \n (January 1956-July 1992)")

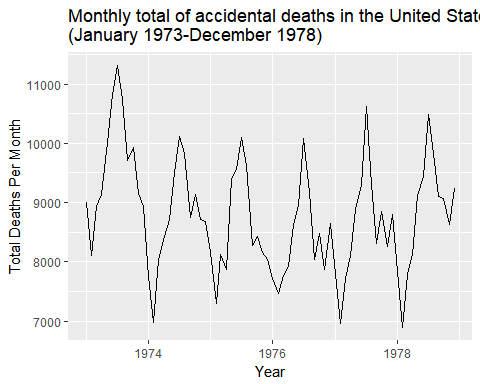


autoplot(BoxCox(dole, BoxCox.lambda(dole)),   
 xlab = "Year",   
 ylab = paste0("# of Unemployed \n", "lambda = ", round(BoxCox.lambda(dole),2), " \n"),  
 main = "BoxCox Transformation \nMonthly total of people on unemployed benefits in Australia \n(January 1956-July 1992)")



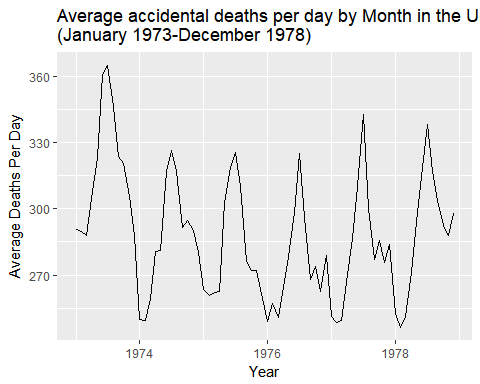
1. Monthly total of accidental deaths in the United States (January 1973-December 1978).

data(usdeaths)  
autoplot(usdeaths,  
 ylab="Total Deaths Per Month",  
 xlab="Year",  
 main="Monthly total of accidental deaths in the United States \n(January 1973-December 1978)")



While the chart is for each month, the days per month are variable so lets adjust for the deaths per day by month to have a more normalized time series.

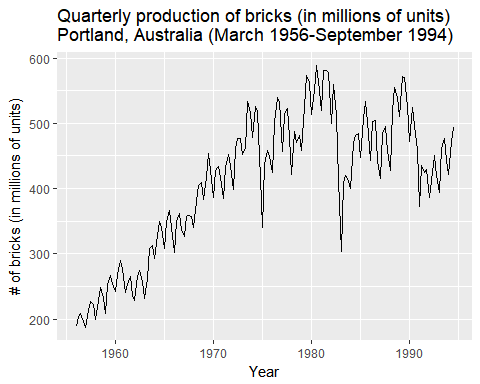
monthdays <- rep(c(31,28,31,30,31,30,31,31,30,31,30,31),6)  
monthdays[38] <- 29 #Leap year adjustment  
  
autoplot(usdeaths/monthdays, ylab="Average Deaths Per Day",  
 xlab="Year",  
 main="Average accidental deaths per day by Month in the United States \n(January 1973-December 1978)")



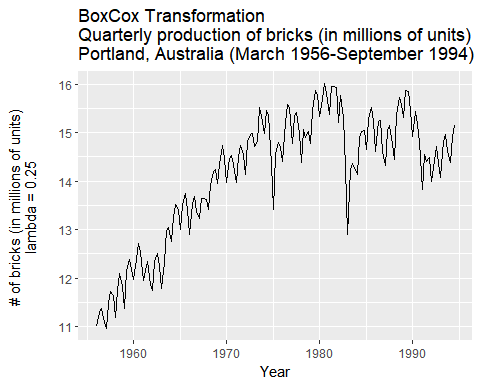
Interestingly, there is some smoothing seen in the earlier years of the time series but more dramatic changes later in the series. Since we adjusted for changes in days per month we can further assume that this is a properly normalized data set month over month.

1. Quarterly production of bricks (in millions of units) at Portland, Australia (March 1956-September 1994).

data(bricksq)  
autoplot(bricksq,  
 xlab = "Year",   
 ylab = paste0("# of bricks (in millions of units)"),  
 main = "Quarterly production of bricks (in millions of units) \nPortland, Australia (March 1956-September 1994)")



autoplot(BoxCox(bricksq, BoxCox.lambda(bricksq)),   
 xlab = "Year",   
 ylab = paste0("# of bricks (in millions of units) \n", "lambda = ", round(BoxCox.lambda(bricksq),2), " \n"),  
 main = "BoxCox Transformation \nQuarterly production of bricks (in millions of units) \nPortland, Australia (March 1956-September 1994)")



Hints: data(package="fma") will give a list of the available data. To plot a transformed data set, use plot(BoxCox(x,0.5)) where x is the name of the data set and 0.5 is the Box-Cox parameter.

2.3 Consider the daily closing IBM stock prices (data set ibmclose).

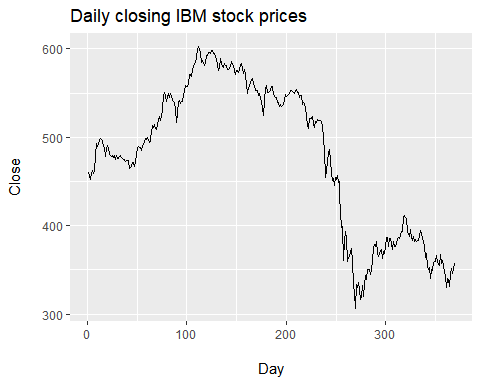
1. Produce some plots of the data in order to become familiar with it.

The sharp downward trend will be challenging for our predictions, it appears to be a shock event and not a normal/seasonal change. We will have to be cautious about our interpretations and predictions when including the shock event.

library(psych)  
library(knitr)  
data(ibmclose)  
kable(describe(ibmclose))

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | vars | n | mean | sd | median | trimmed | mad | min | max | range | skew | kurtosis | se |
| X1 | 1 | 369 | 478.4688 | 84.21924 | 494 | 481.9663 | 93.4038 | 306 | 603 | 297 | -0.3784213 | -1.263461 | 4.384278 |

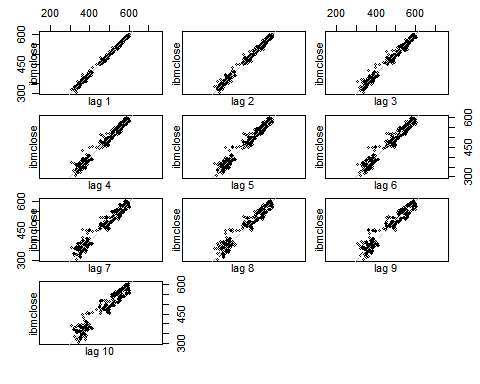
autoplot(ibmclose,   
 xlab = "\nDay",   
 ylab = "Close\n",  
 main = "Daily closing IBM stock prices")



Lagged Plot

The lagged plot shows the correlation over lagged periods, it's not entirely surprisingly that previous day close has more impact on the next day's close than several days later. However, it is useful to visualize the impact when attempting to further model.

lag.plot(ibmclose, lags= 10, do.lines=FALSE)



1. Split the data into a training set of 300 observations and a test set of 69 observations.

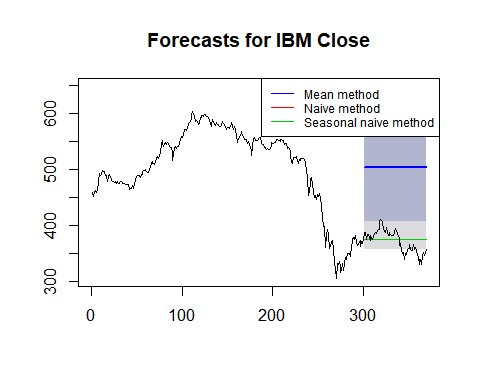
Splitting the data set using the window function for the first 300 observation for training and 69 for the test set.

ibmclose\_train <- window(ibmclose, start=1, end=300)  
ibmclose\_test <- window(ibmclose, start=301)

1. Try various benchmark methods to forecast the training set and compare the results on the test set. Which method did best?

The mean method appears to have performed the worse, which intuitively make sense with the higher performance of the stock before the shock loss event. The Naive and Seasonal Naive performed exactly the same but performed much better looking at the MAE score compared across all three methods. My prefernce would be for naive method based on our analysis of the lag method and how I understand stock performances. It may not be completely appropriate to assume that the stock performs in a seasonal fashion but performance on a short term scale may be indictative of some of the future performance.

ibmclosefit1 <- meanf(ibmclose\_train, h = 69)  
ibmclosefit2 <- rwf(ibmclose\_train, h = 69)  
ibmclosefit3 <- snaive(ibmclose\_train, h = 69)  
  
plot(ibmclosefit1, col = 1, main="Forecasts for IBM Close")  
lines(ibmclosefit2$mean, col=2)  
lines(ibmclosefit3$mean, col=3)  
lines(ibmclose)  
legend("topright",   
 cex = .75,  
 lty=1,   
 col=c(4,2,3),   
 legend=c("Mean method",   
 "Naive method",   
 "Seasonal naive method"))



Mean Method

kable(accuracy(ibmclosefit1, ibmclose\_test))

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ME | RMSE | MAE | MPE | MAPE | MASE | ACF1 | Theil's U |
| Training set | 0.000 | 73.61532 | 58.72231 | -2.642058 | 13.03019 | 11.52098 | 0.9895779 | NA |
| Test set | -130.618 | 132.12557 | 130.61797 | -35.478819 | 35.47882 | 25.62649 | 0.9314689 | 19.05515 |

Naive method

kable(accuracy(ibmclosefit2, ibmclose\_test))

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ME | RMSE | MAE | MPE | MAPE | MASE | ACF1 | Theil's U |
| Training set | -0.2809365 | 7.302815 | 5.09699 | -0.0826287 | 1.115844 | 1.000000 | 0.1351052 | NA |
| Test set | -3.7246377 | 20.248099 | 17.02899 | -1.2939174 | 4.668186 | 3.340989 | 0.9314689 | 2.973486 |

Seasonal naive method

kable(accuracy(ibmclosefit3, ibmclose\_test))

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ME | RMSE | MAE | MPE | MAPE | MASE | ACF1 | Theil's U |
| Training set | -0.2809365 | 7.302815 | 5.09699 | -0.0826287 | 1.115844 | 1.000000 | 0.1351052 | NA |
| Test set | -3.7246377 | 20.248099 | 17.02899 | -1.2939174 | 4.668186 | 3.340989 | 0.9314689 | 2.973486 |

## Week 2

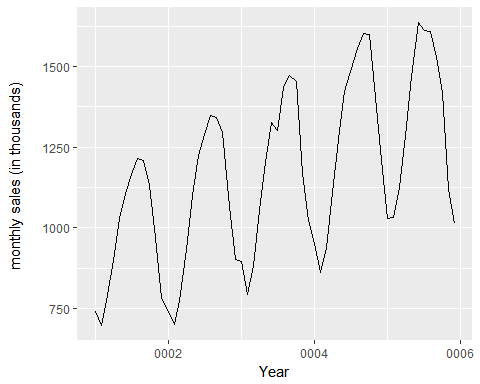
The data below represent the monthly sales (in thousands) of product A for a plastics manufacturer for years 1 through 5 (data set plastics).

library(fma)  
plastics

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

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

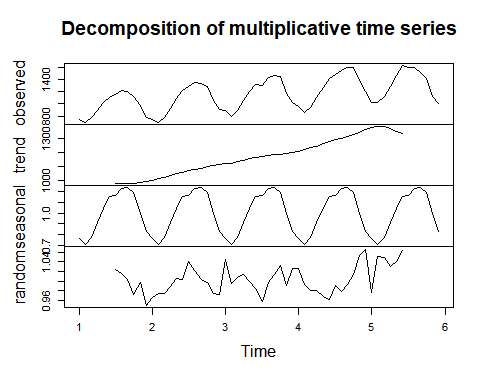
library(ggfortify)  
autoplot(plastics, xlab = "Year", ylab = "monthly sales (in thousands) \n")



There is clearly seasonal fluctuations here, it shows a trend towards high volume in summer and a sharp dip in winter for lower volume.

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

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

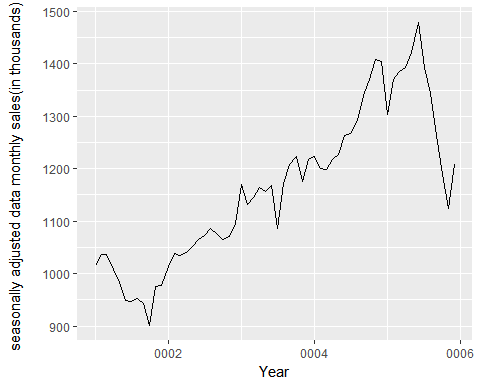


1. Do the results support the graphical interpretation from part (a)?

The results do support the seasonality trend assumption from part (a), it is useful to see the seasonal trend accounting for upward trend in the observed values to highlight the consistent seasonality.

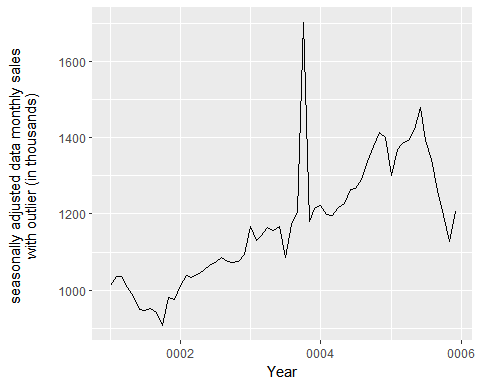
1. Compute and plot the seasonally adjusted data.

fit <- stl(plastics, t.window = 7, s.window="periodic", robust=TRUE)  
autoplot(seasadj(fit), xlab = "Year", ylab = "seasonally adjusted data monthly sales(in thousands) \n")



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

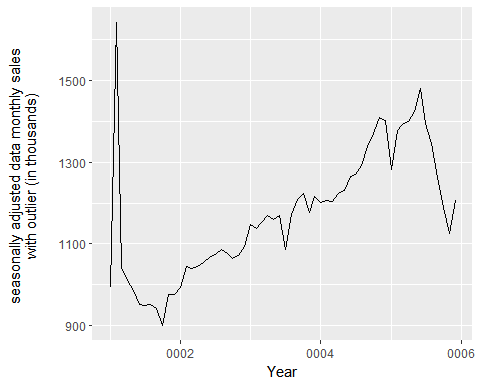
outlier\_plastics <- plastics  
outlier\_plastics[34] <- outlier\_plastics[30] + 600  
fit <- stl(outlier\_plastics, t.window = 7, s.window="periodic", robust=TRUE)  
autoplot(seasadj(fit), xlab = "Year", ylab = "seasonally adjusted data monthly sales \n with outlier (in thousands) \n")



Surpisingly, the effect of an outlier has very little impact on the overall trend of the line but does show considerable impact on the line at the point of the outlier.

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

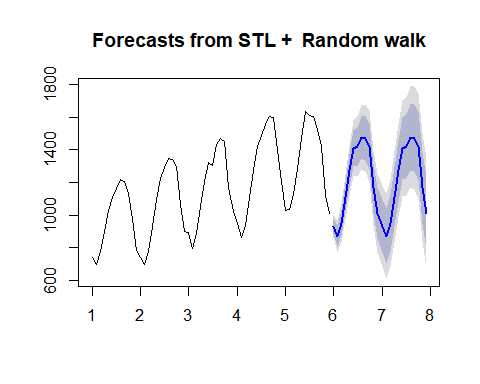
outlier\_plastics <- plastics  
outlier\_plastics[2] <- outlier\_plastics[2] + 600  
fit <- stl(outlier\_plastics, t.window = 7, s.window="periodic", robust=TRUE)  
autoplot(seasadj(fit), xlab = "Year", ylab = "seasonally adjusted data monthly sales \n with outlier (in thousands) \n")



It does appear that there is an initial smoothing of the line from the outlier at the beginning of the series. The line does return to a similar seasonally adjusted line from the original data which makes sense once the line is further out from the outlier. It does seem intiuitive that with an outlier at the beginning of the time series there is less surrounding data to account for the outlier.

1. Use a random walk with drift to produce forecasts of the seasonally adjusted data.

fc <- stlf(plastics, method="naive")  
plot(fc)



1. Reseasonalize the results to give forecasts on the original scale.

fit <- stl(plastics, s.window="periodic", robust=TRUE)  
fitadj <- seasadj(fit)  
plot(naive(fitadj))

