DATA624 Homework 1

Esteban Aramayo, Coffy Andrews-Guo, LeTicia Cancel, Joseph Connolly, Ian Costello

5/31/2022

Table of Contents

library(ggfortify)  
library(openxlsx)  
library(fpp2)  
library(fma)  
library(gridExtra)  
library(seasonal)  
library(ggplot2)  
library(patchwork)  
library(caret)  
library(grid)

# Week 1

## HA 2.1

### Use the help function to explore what the series gold, woolyrnq and gas represent.

The fpp2 forecast package for “Forecasting: Principles and Practice” (2nd Edition) was loaded to explore three time series data library: gold, woolyrnq, and gas. The RStudio IDE help function is a comprehensive built-in system providing the following:

| Data Set | Description | Format | Source |
| --- | --- | --- | --- |
| gold | Daily morning gold prices in US dollars. 1 January 1985 - 31 March 1989. | Time series data | Not Available |
| woolyrnq | Quarterly production of woollen yarn in Australia: tonnes. Mar 1965 - Sep 1994 | Time series data | Time Series Data Library |
| gas | Australian monthly gas production: 1956-1995 | Time series data | Australian Bureau of Statistics |

#help function for each series using question mark "??"  
??gold

## starting httpd help server ... done

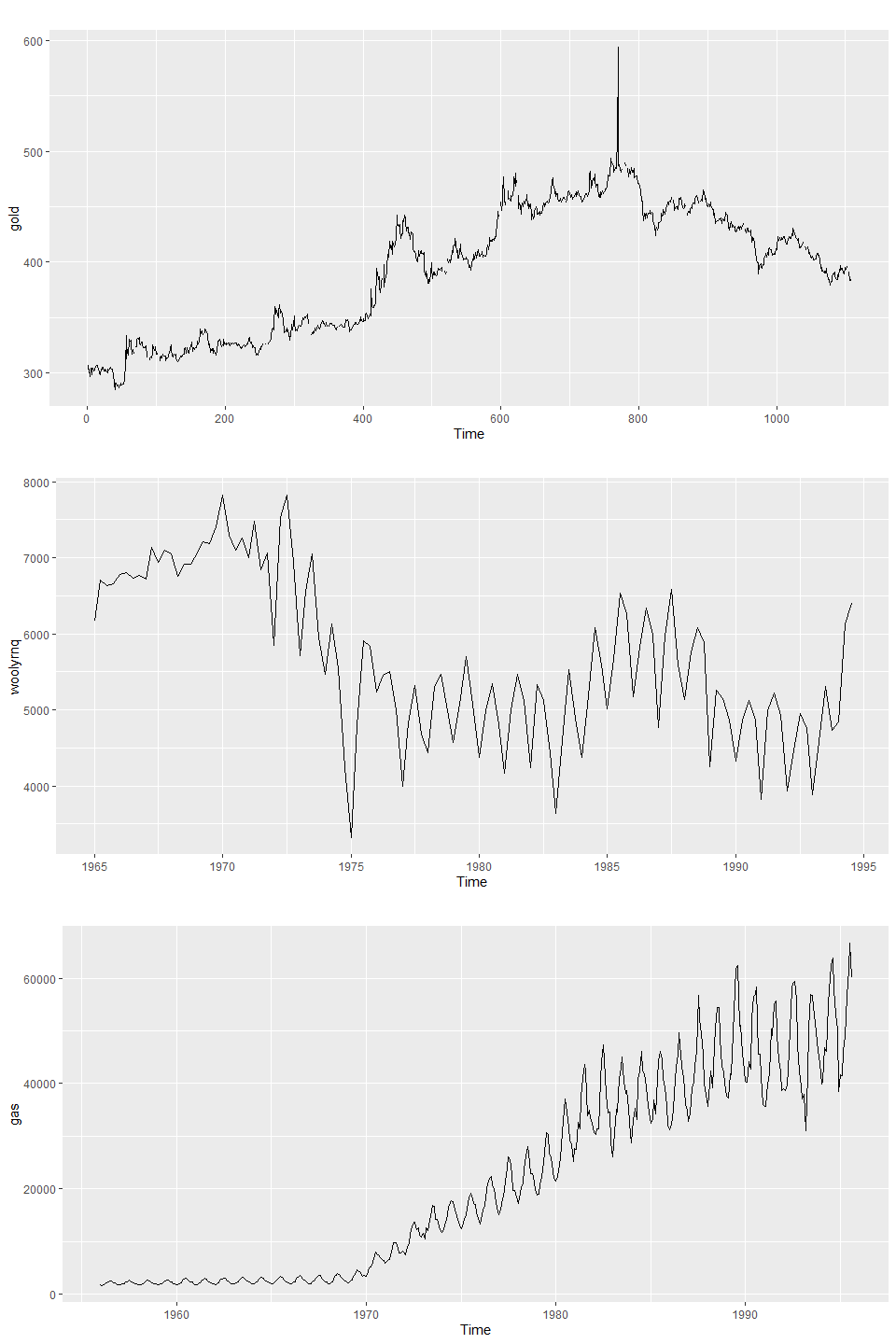
#help function for each series using question mark "??"  
??woolyrnq

#help function for each series using question mark "??"  
??gas

The help function may provide package code examples and hyperlink to additional research documentation.

### a. Use autoplot() to plot each of these in separate plots.

#autoplots of each series  
grid.arrange(autoplot(gold),autoplot(woolyrnq),autoplot(gas))



### b. What is the frequency of each series? Hint: apply the frequency() function.

* gold has a frequency of 1, meaning it is annual.
* Woolyrng has a frequency of 1, meaning it is quarterly.
* gas has a frequency of 12, meaning it is monthly.

#frequency of each series  
frequency(gold)

## [1] 1

frequency(woolyrnq)

## [1] 4

frequency(gas)

## [1] 12

### c. Use which.max() to spot the outlier in the gold series. Which observation was it?

The first maximum value in the gold series is located in the 475 position with a value of 6.66^{4}.

## HA 2.3

### Download some monthly Australian retail data from the book website. These represent retail sales in various categories for different Australian states, and are stored in a MS-Excel file.

### a. You can read the data into R with the following script:

retaildata <- read.xlsx("https://otexts.com/fpp2/extrafiles/retail.xlsx",startRow = 2)

### b. Select one of the time series as follows (but replace the column name with your own chosen column):

myts <- ts(retaildata[,"A3349873A"],  
 frequency=12, start=c(1982,4))  
#myts #code is silent because it represents an example - see "myts2"   
  
myts2 <- ts(retaildata[,"A3349791W"],  
 frequency=12, start=c(1982,4))

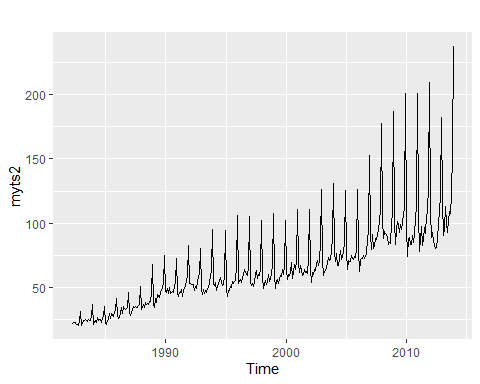
#### Explore your chosen retail time series using the following functions:

autoplot(), ggseasonplot(), ggsubseriesplot(), gglagplot(), ggAcf()

#### Can you spot any seasonality, cyclicity and trend? What do you learn about the series?

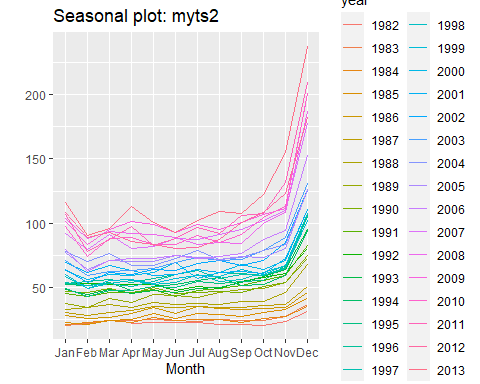
The first plot using autoplot() shows some seasonality for each year telling us that the price spikes at different points. There is also an upward trend showing that the price of gold continues to increase overtime. The length between the min and max for each of the seasonal spike grows longer each year telling us that the price difference between the minimum and maximum price spreads out more over time.

#time series qutoplot  
autoplot(myts2)



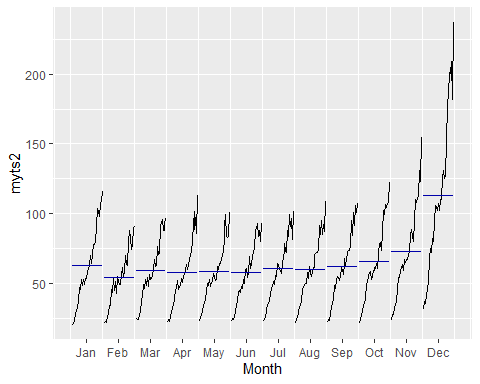
The seasonal plot below using ggseasonlot() gives us more clarity on when the seasonality takes place. Each year has a spike in December with the max prices increasing each year. We can take a guess as to why December is the month that spikes since this is when holiday shopping takes place.

#time series seasonal plot  
ggseasonplot(myts2)



The subseries plot confirms the seasonality shown in the previous plots with December being the seasonal month. This plot creates a better visualization for a couple of details better than the previous plots. The difference between the minimum and maximum over the years is clearer. The minimum has a minor increase in december but the maximum has a dramatic increase when compared to the other months. The mean also has a big increase in December but then levels out during the spring and summer months.

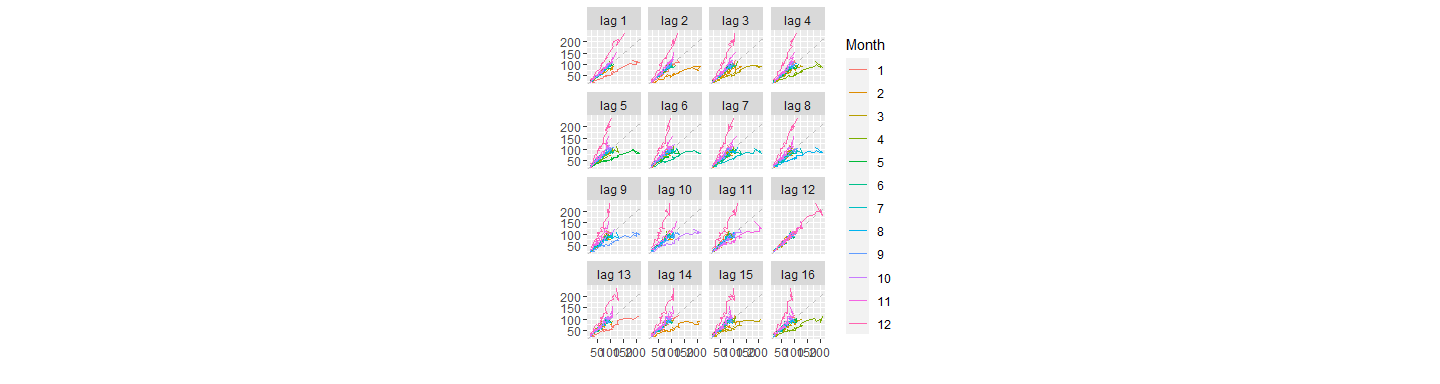
#time series sub series plot  
ggsubseriesplot(myts2)



The lag plot shows strong positive correlation for all months in lag 12.

**Alternative description** The scatterplots shows the lagged values of the time series. Each graph shows plotted against for different values of . The relationship is strongly positive at lag 16, reflecting the strong seasonality in the data.

#time series lag plot  
gglagplot(myts2)



The ACF plot supports the lag plot by showing a high autocorrelation coefficient for lag 12. There are also no negative correlations which was not as clear in the subseries plot. They are also all significantly different from zero.

**Additional description** The Acf graph shows the autocorrelation coefficients, correlogram, of the retail sales in various categories for different Australian states base on A3349791W time series. The dashed blue lines indicate whether the correlations are significantly different from zero.

#time series ACF  
ggAcf(myts2)

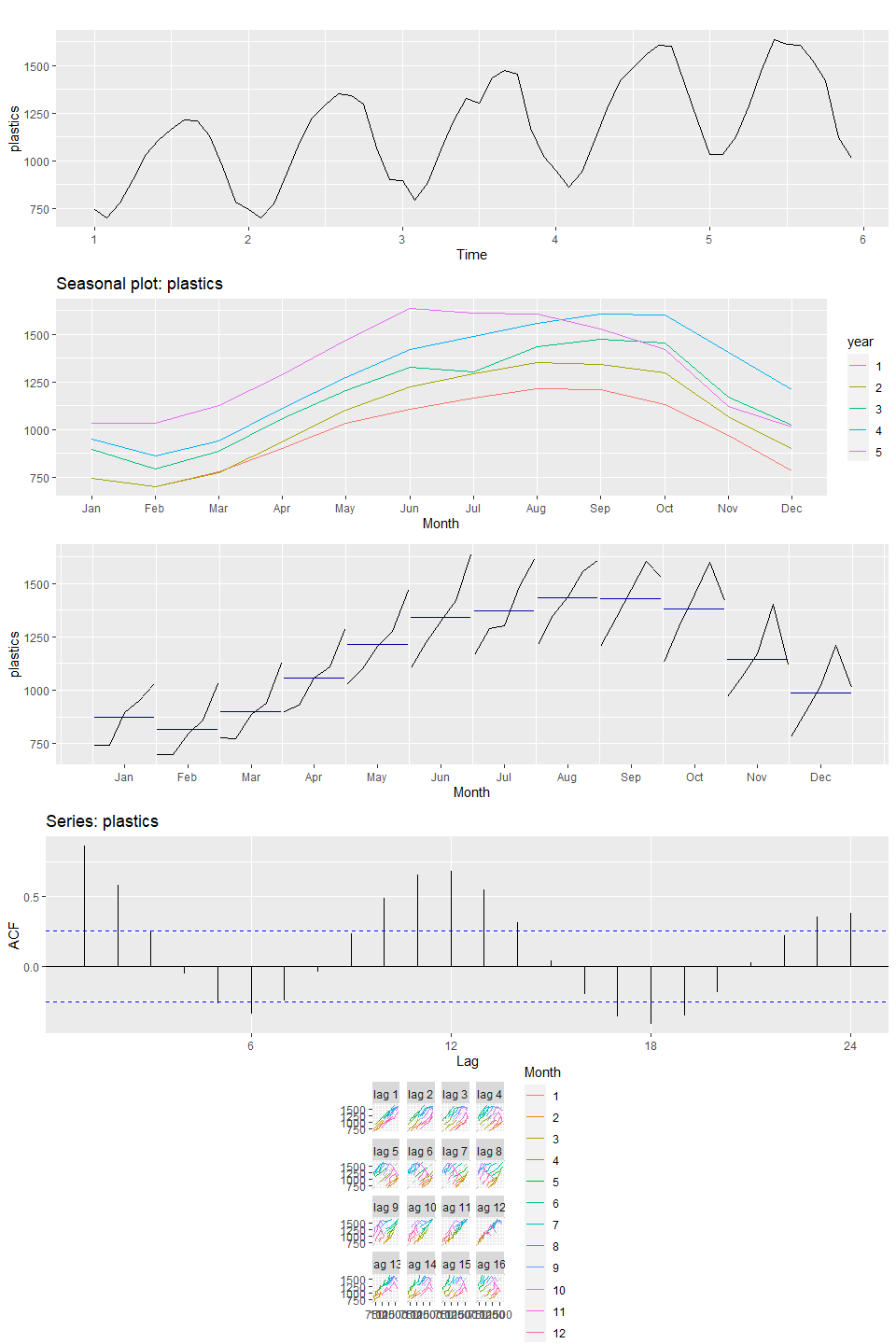


## HA 6.2

The plastics data set consists of the monthly sales (in thousands) of product A for a plastics manufacturer for five years.

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

#autoplot(), ggseasonplot(), ggsubseriesplot(), gglagplot(), ggAcf()  
p1 <- autoplot(plastics)  
p2 <- ggseasonplot(plastics)  
p3 <- ggsubseriesplot(plastics)  
p4 <- gglagplot(plastics)  
p5 <- ggAcf(plastics)  
grid.arrange(p1,p2,p3,p5,p4, ncol=1)

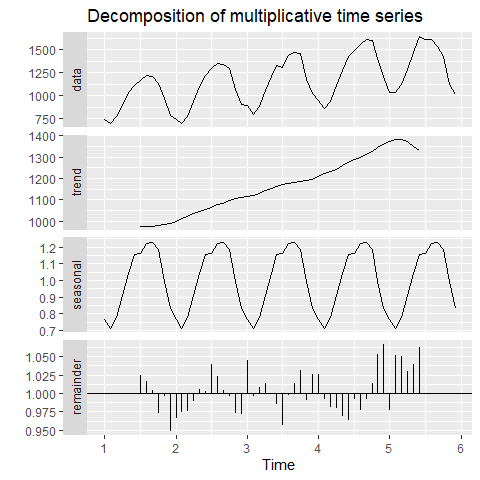


**Additional description** The time plot automatically shows the monthly sales (in thousands) of product A for a plastics manufacturer for five years and reveals: (1) there is an increasing trend; (2) there is a mild seasonal pattern that increases in size as the level of the series increases; and (3) the sudden drop at the year end/starting year is due to government subsidies for pollution control, such as [deposit-refund systems](https://media.rff.org/archive/files/sharepoint/WorkImages/Download/RFF-DP-11-47.pdf).

The season plot shows a seasonal pattern occurs from February to December for each year (five years) time series.

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

#multiplicative decomposition  
decomp\_plastics <- plastics %>%  
 decompose(type = "multiplicative")  
  
decomp\_plastics %>% autoplot()



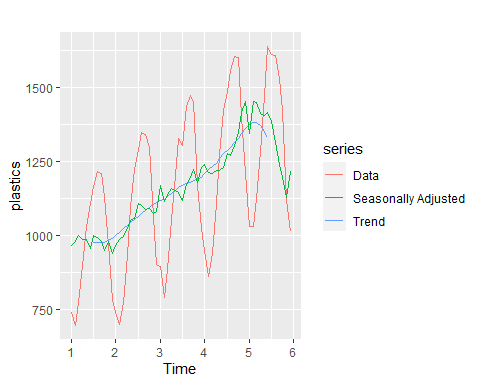
### c. Do the results support the graphical interpretation from part a?

The “Data” and “Seasonal” sections of the Decomposition shows similar results to the autoplot() chart. There is a steady seasonal trend that has a similar duration for each seasonal cycle. The “Trend” section of the Decomposition also supports the visuals from part A and shows a steady upward trend from years 1-5.

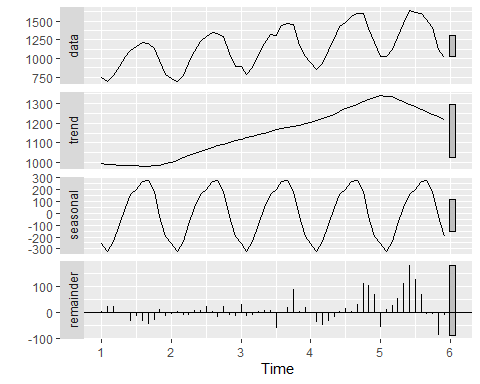
**Additional/Alternative description** The multiplicative decomposition plot supports Part (a) interpretation of a seasonal pattern. The seasonal pattern is unchanging, the remainder component has a lot of large values, and has an increasing trend with some missing observations from the beginning and the end of the data set.

### d. Compute and plot the seasonally adjusted data.

autoplot(plastics, series = "Data") +   
 autolayer(trendcycle(decomp\_plastics), series = "Trend") +  
 autolayer(seasadj(decomp\_plastics), series = "Seasonally Adjusted")

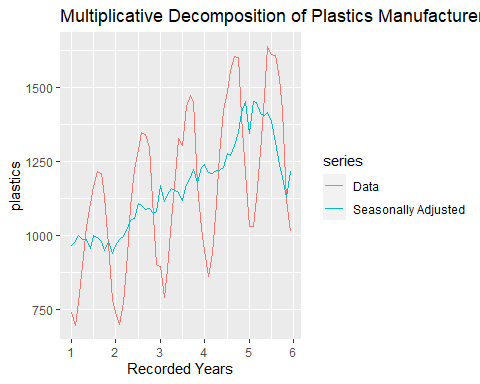


#fit <- seas(x = plastics, x11="")  
  
plastics %>%  
 stl(s.window = "periodic", robust = TRUE) %>%  
 autoplot()



**Alternative/Additional Plot**

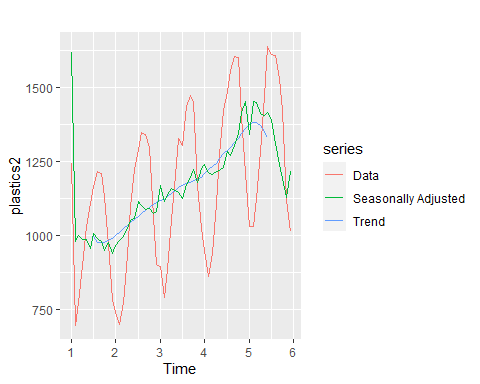
plastics %>% decompose(type = "multiplicative") -> fit  
autoplot(plastics, series = "Data") +   
 autolayer(seasadj(fit), series = "Seasonally Adjusted") +  
 xlab("Recorded Years") +  
 ggtitle("Multiplicative Decomposition of Plastics Manufacturer Sales Data")



### e. 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?

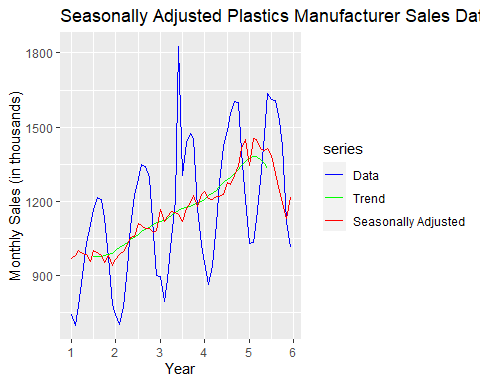
The outlier was applied to the first observation in the plastics data so the chart changed only in year 1. The outlier being in the beginning of the data had no impact on the rest of the plot.

#creating a copy of plastics and adding 500 to one observation  
plastics2 <- plastics  
plastics2[1] <- plastics2[1] + 500  
  
decomp\_plastics2 <- plastics2 %>%  
 decompose(type = "multiplicative")  
  
autoplot(plastics2, series = "Data") +   
 autolayer(trendcycle(decomp\_plastics2), series = "Trend") +  
 autolayer(seasadj(decomp\_plastics2), series = "Seasonally Adjusted")

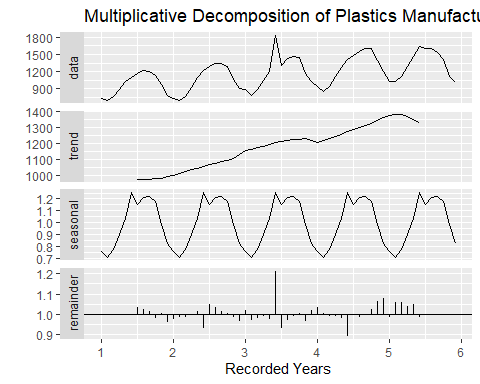


**Additional/Alternative Plots**

plasticsOutlier <- plastics  
plasticsOutlier[30] <- plasticsOutlier[30] + 500  
  
autoplot(plasticsOutlier, series = "Data") +  
 autolayer(trendcycle(fit), series = "Trend") +  
 autolayer(seasadj(fit), series = "Seasonally Adjusted") +   
 xlab("Year") + ylab("Monthly Sales (in thousands)") +  
 ggtitle("Seasonally Adjusted Plastics Manufacturer Sales Data") +  
 scale\_color\_manual(values = c("blue", "green", "red"),  
 breaks = c("Data", "Trend", "Seasonally Adjusted"))



plasticsOutlier %>% decompose(type="multiplicative") %>%  
 autoplot() + xlab("Recorded Years") +  
 ggtitle("Multiplicative Decomposition of Plastics Manufacturer Sales Data")

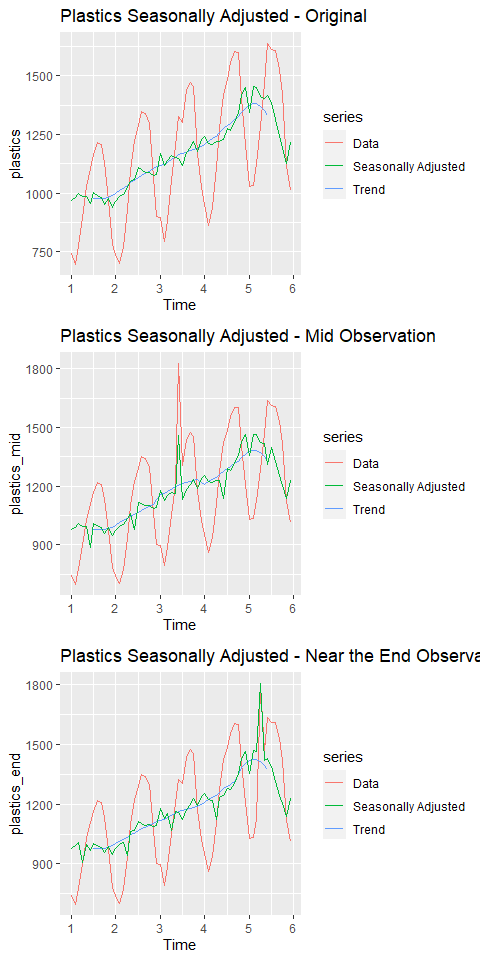


The effects of the outlier is noticeable and still supports Part (a) interpretation of a seasonal pattern. The difference is shown in the data component with the changed observation creates a high peak, then change direction and continues along a seasonal pattern.

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

The middle outlier has a greater impact on the Seasonally adjusted line than the end outlier when compared to the original plot. When the last observation is the outlier the Data and Seasonally Adjusted lines trend upward and all other areas on the chart are not impacted. When a middle observation is the outlier we see a different seasonally adusted line when compared to the original chart. The downward and upward spikes around the outlier are sharper on the middle outlier chart while the original chart has subtle changes with a fairly even upward trend.

plastics\_mid <- plastics  
plastics\_end <- plastics   
plastics\_mid[30] <- plastics\_mid[30] + 500 #outlier in the middle  
plastics\_end[52] <- plastics\_end[52] + 500 #outlier to reflect "near the end"  
  
decomp\_mid <- plastics\_mid %>%  
 decompose(type = "multiplicative")  
  
p <- autoplot(plastics, series = "Data") +   
 autolayer(trendcycle(decomp\_plastics), series = "Trend") +  
 autolayer(seasadj(decomp\_plastics), series = "Seasonally Adjusted") +  
 ggtitle("Plastics Seasonally Adjusted - Original")  
  
p1 <- autoplot(plastics\_mid, series = "Data") +   
 autolayer(trendcycle(decomp\_mid), series = "Trend") +  
 autolayer(seasadj(decomp\_mid), series = "Seasonally Adjusted") +  
 ggtitle("Plastics Seasonally Adjusted - Mid Observation")  
  
decomp\_end <- plastics\_end %>%  
 decompose(type = "multiplicative")  
  
p2 <- autoplot(plastics\_end, series = "Data") +   
 autolayer(trendcycle(decomp\_end), series = "Trend") +  
 autolayer(seasadj(decomp\_end), series = "Seasonally Adjusted") +  
 ggtitle("Plastics Seasonally Adjusted - Near the End Observation")  
  
grid.arrange(p, p1,p2)



# Week 2

## KJ 3.1

The UC Irvine Machine Learning Repository contains a data set related to glass identification. The data consist of 214 glass samples labeled as one of seven class categories. There are nine predictors, including the refractive index and percentages of eight elements: Na, Mg, Al, Si, K, Ca, Ba, and Fe. The data can be accessed via:

library(mlbench)  
data(Glass)  
str(Glass)

## 'data.frame': 214 obs. of 10 variables:  
## $ RI : num 1.52 1.52 1.52 1.52 1.52 ...  
## $ Na : num 13.6 13.9 13.5 13.2 13.3 ...  
## $ Mg : num 4.49 3.6 3.55 3.69 3.62 3.61 3.6 3.61 3.58 3.6 ...  
## $ Al : num 1.1 1.36 1.54 1.29 1.24 1.62 1.14 1.05 1.37 1.36 ...  
## $ Si : num 71.8 72.7 73 72.6 73.1 ...  
## $ K : num 0.06 0.48 0.39 0.57 0.55 0.64 0.58 0.57 0.56 0.57 ...  
## $ Ca : num 8.75 7.83 7.78 8.22 8.07 8.07 8.17 8.24 8.3 8.4 ...  
## $ Ba : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Fe : num 0 0 0 0 0 0.26 0 0 0 0.11 ...  
## $ Type: Factor w/ 6 levels "1","2","3","5",..: 1 1 1 1 1 1 1 1 1 1 ...

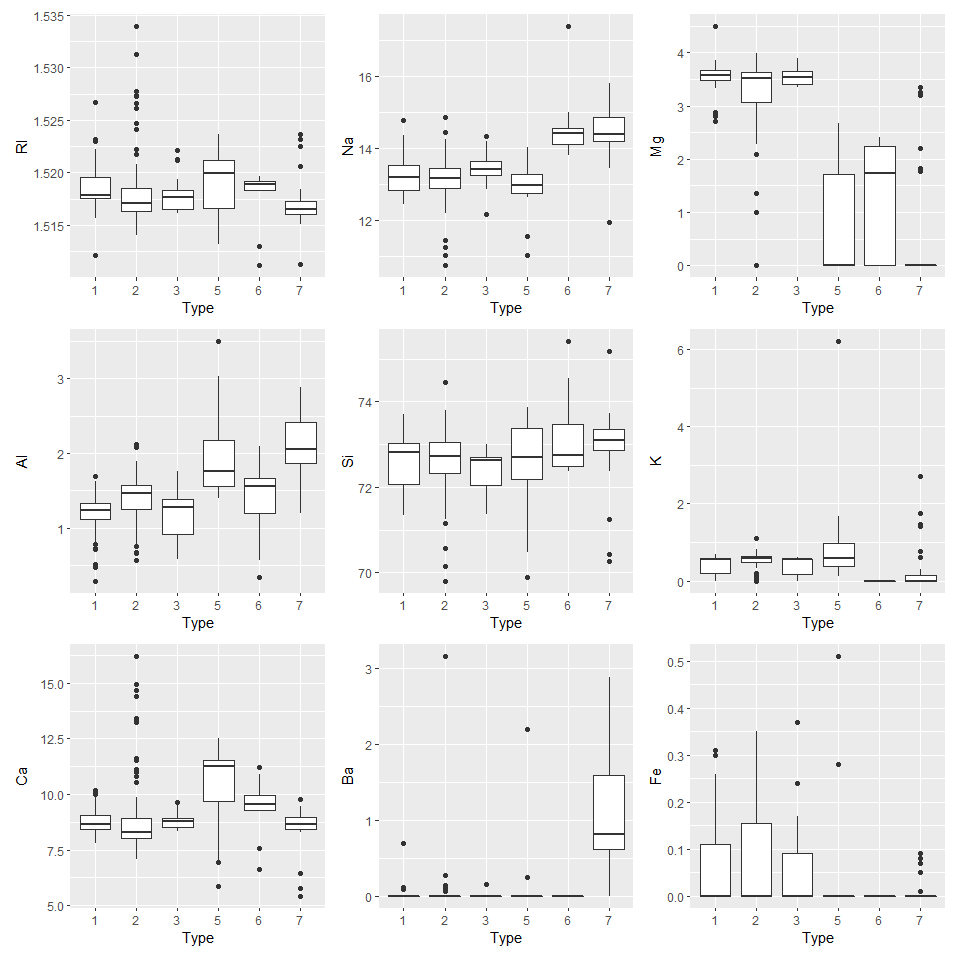
### a. Using visualizations, explore the predictor variables to understand their distributions as well as the relationships between predictors.

When we explore the Glass dataset using summary we see that there are no NA values for any of the predictors. There are also no negative values with the minimum value being 0.00 (zero) for some predictors.

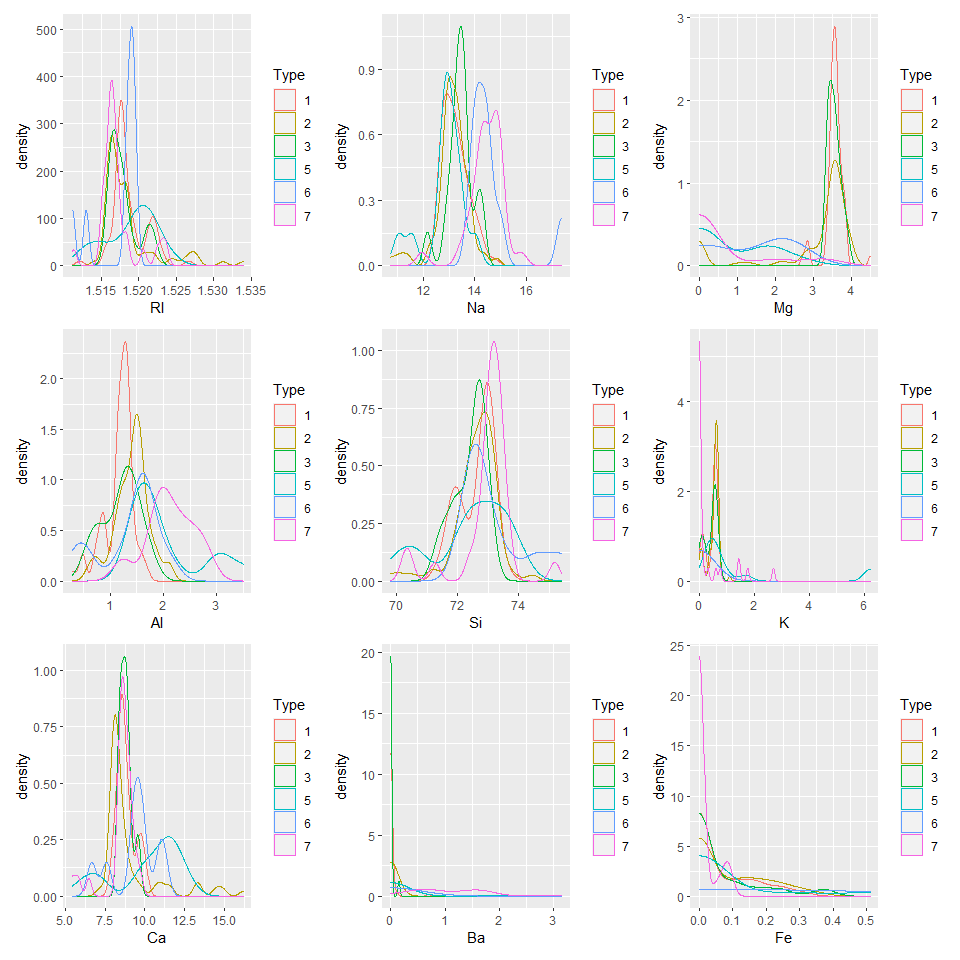
summary(Glass)

## RI Na Mg Al   
## Min. :1.511 Min. :10.73 Min. :0.000 Min. :0.290   
## 1st Qu.:1.517 1st Qu.:12.91 1st Qu.:2.115 1st Qu.:1.190   
## Median :1.518 Median :13.30 Median :3.480 Median :1.360   
## Mean :1.518 Mean :13.41 Mean :2.685 Mean :1.445   
## 3rd Qu.:1.519 3rd Qu.:13.82 3rd Qu.:3.600 3rd Qu.:1.630   
## Max. :1.534 Max. :17.38 Max. :4.490 Max. :3.500   
## Si K Ca Ba   
## Min. :69.81 Min. :0.0000 Min. : 5.430 Min. :0.000   
## 1st Qu.:72.28 1st Qu.:0.1225 1st Qu.: 8.240 1st Qu.:0.000   
## Median :72.79 Median :0.5550 Median : 8.600 Median :0.000   
## Mean :72.65 Mean :0.4971 Mean : 8.957 Mean :0.175   
## 3rd Qu.:73.09 3rd Qu.:0.6100 3rd Qu.: 9.172 3rd Qu.:0.000   
## Max. :75.41 Max. :6.2100 Max. :16.190 Max. :3.150   
## Fe Type   
## Min. :0.00000 1:70   
## 1st Qu.:0.00000 2:76   
## Median :0.00000 3:17   
## Mean :0.05701 5:13   
## 3rd Qu.:0.10000 6: 9   
## Max. :0.51000 7:29

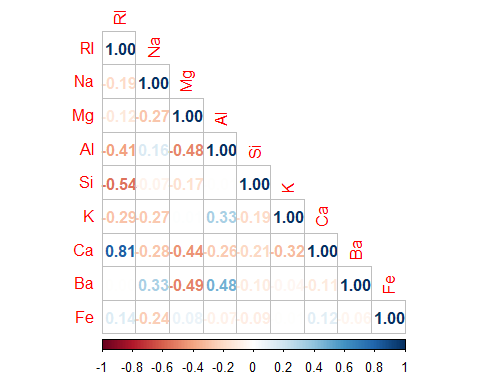
ri\_b <- ggplot(Glass, aes(x = Type, y = RI)) +  
 geom\_boxplot()  
  
na\_b <- ggplot(Glass, aes(x = Type, y = Na)) +  
 geom\_boxplot()  
  
mg\_b <- ggplot(Glass, aes(x = Type, y = Mg)) +  
 geom\_boxplot()  
  
al\_b <- ggplot(Glass, aes(x = Type, y = Al)) +  
 geom\_boxplot()  
  
si\_b <- ggplot(Glass, aes(x = Type, y = Si)) +  
 geom\_boxplot()  
  
k\_b <- ggplot(Glass, aes(x = Type, y = K)) +  
 geom\_boxplot()  
  
ca\_b <- ggplot(Glass, aes(x = Type, y = Ca)) +  
 geom\_boxplot()  
  
ba\_b <- ggplot(Glass, aes(x = Type, y = Ba)) +  
 geom\_boxplot()  
  
fe\_b <- ggplot(Glass, aes(x = Type, y = Fe)) +  
 geom\_boxplot()  
  
  
ri\_b+na\_b+mg\_b+al\_b+si\_b+k\_b+ca\_b+ba\_b+fe\_b+  
 plot\_layout(ncol=3)



ri\_d <- ggplot(Glass, aes(RI, color=Type)) + geom\_density()  
na\_d <- ggplot(Glass, aes(Na, color=Type)) + geom\_density()  
mg\_d <- ggplot(Glass, aes(Mg, color=Type)) + geom\_density()  
al\_d <- ggplot(Glass, aes(Al, color=Type)) + geom\_density()  
si\_d <- ggplot(Glass, aes(Si, color=Type)) + geom\_density()  
k\_d <- ggplot(Glass, aes(K, color=Type)) + geom\_density()  
ca\_d <- ggplot(Glass, aes(Ca, color=Type)) + geom\_density()  
ba\_d <- ggplot(Glass, aes(Ba, color=Type)) + geom\_density()  
fe\_d <- ggplot(Glass, aes(Fe, color=Type)) + geom\_density()  
  
ri\_d+na\_d+mg\_d+al\_d+si\_d+k\_d+ca\_d+ba\_d+fe\_d+  
 plot\_layout(ncol=3)



corrplot::corrplot(cor(Glass[c("RI", "Na", "Mg", "Al", "Si", "K", "Ca", "Ba", "Fe")]), method = 'number', type = 'lower')



### b. Do there appear to be any outliers in the data? Are any predictors skewed?

The boxplots help us see the outliers for each type for each predictor. Type 2 has outliers for the Ri, Mg, and Ca predictors. The Ba predictor has more data for Type 7 with the rest of the types being scattered. The mean for Si is consistent between all types. When we explore the distribution plots we see that Si is closest to a normal distribution. K, Ba, and Fa are left skewed and Mg is right skewed. The other predictors are slightly skewed.

### c. Are there any relevant transformations of one or more predictors that might improve the classification model?

Experimenting with BoxCox transformations for Ba and Fe since both predictors are heavily skewed to the left

## KJ 3.2

The soybean data can also be found at the UC Irvine Machine Learning Repository. Data were collected to predict disease in 683 soybeans. The 35 predictors are mostly categorical and include information on the environmental conditions (e.g., temperature, precipitation) and plant conditions (e.g., left spots, mold growth). The outcome labels consist of 19 distinct classes. The data can be loaded via:

library(mlbench)  
data(Soybean)  
head(Soybean)

## Class date plant.stand precip temp hail crop.hist area.dam  
## 1 diaporthe-stem-canker 6 0 2 1 0 1 1  
## 2 diaporthe-stem-canker 4 0 2 1 0 2 0  
## 3 diaporthe-stem-canker 3 0 2 1 0 1 0  
## 4 diaporthe-stem-canker 3 0 2 1 0 1 0  
## 5 diaporthe-stem-canker 6 0 2 1 0 2 0  
## 6 diaporthe-stem-canker 5 0 2 1 0 3 0  
## sever seed.tmt germ plant.growth leaves leaf.halo leaf.marg leaf.size  
## 1 1 0 0 1 1 0 2 2  
## 2 2 1 1 1 1 0 2 2  
## 3 2 1 2 1 1 0 2 2  
## 4 2 0 1 1 1 0 2 2  
## 5 1 0 2 1 1 0 2 2  
## 6 1 0 1 1 1 0 2 2  
## leaf.shread leaf.malf leaf.mild stem lodging stem.cankers canker.lesion  
## 1 0 0 0 1 1 3 1  
## 2 0 0 0 1 0 3 1  
## 3 0 0 0 1 0 3 0  
## 4 0 0 0 1 0 3 0  
## 5 0 0 0 1 0 3 1  
## 6 0 0 0 1 0 3 0  
## fruiting.bodies ext.decay mycelium int.discolor sclerotia fruit.pods  
## 1 1 1 0 0 0 0  
## 2 1 1 0 0 0 0  
## 3 1 1 0 0 0 0  
## 4 1 1 0 0 0 0  
## 5 1 1 0 0 0 0  
## 6 1 1 0 0 0 0  
## fruit.spots seed mold.growth seed.discolor seed.size shriveling roots  
## 1 4 0 0 0 0 0 0  
## 2 4 0 0 0 0 0 0  
## 3 4 0 0 0 0 0 0  
## 4 4 0 0 0 0 0 0  
## 5 4 0 0 0 0 0 0  
## 6 4 0 0 0 0 0 0

summary(Soybean)

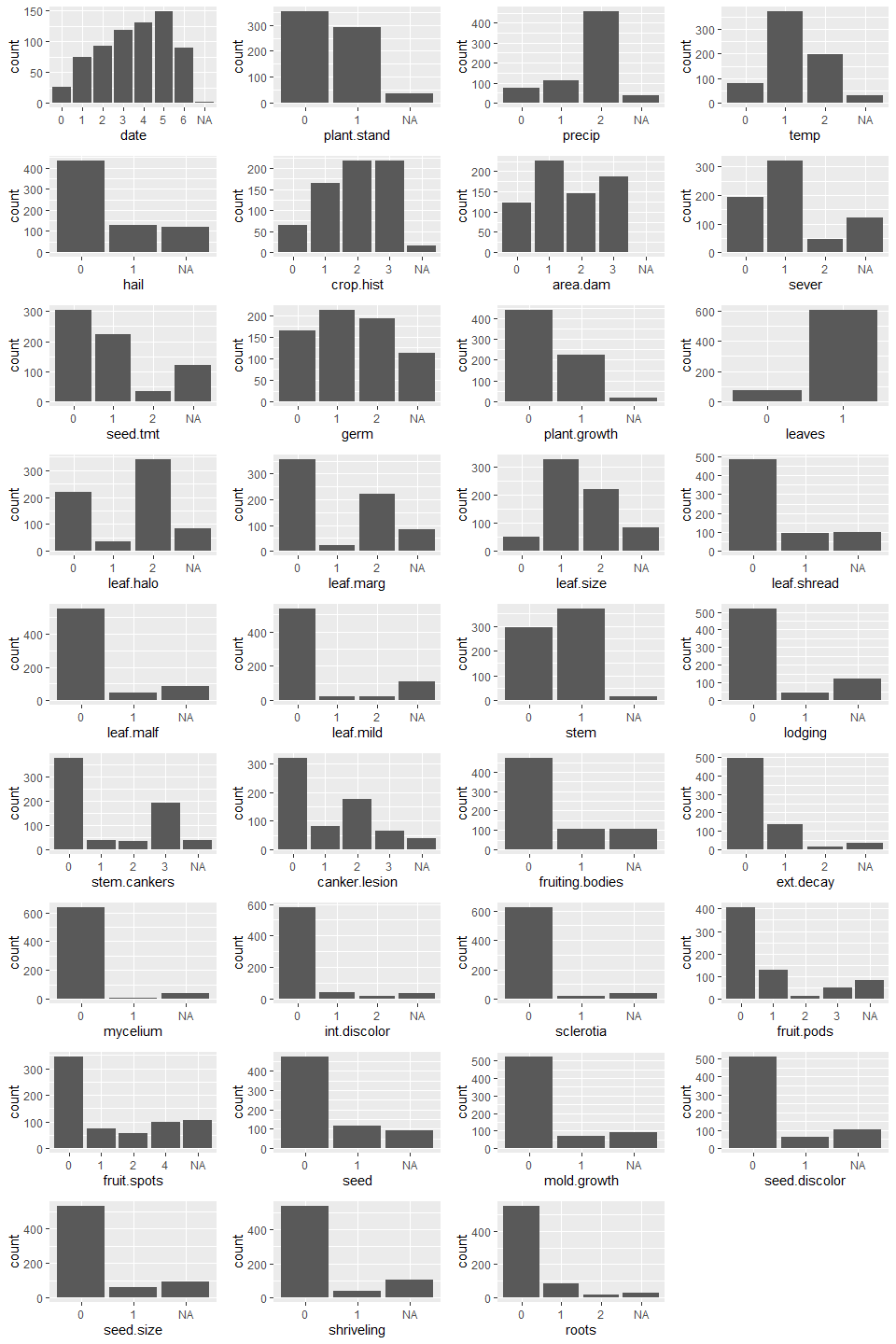
## Class date plant.stand precip temp   
## brown-spot : 92 5 :149 0 :354 0 : 74 0 : 80   
## alternarialeaf-spot: 91 4 :131 1 :293 1 :112 1 :374   
## frog-eye-leaf-spot : 91 3 :118 NA's: 36 2 :459 2 :199   
## phytophthora-rot : 88 2 : 93 NA's: 38 NA's: 30   
## anthracnose : 44 6 : 90   
## brown-stem-rot : 44 (Other):101   
## (Other) :233 NA's : 1   
## hail crop.hist area.dam sever seed.tmt germ plant.growth  
## 0 :435 0 : 65 0 :123 0 :195 0 :305 0 :165 0 :441   
## 1 :127 1 :165 1 :227 1 :322 1 :222 1 :213 1 :226   
## NA's:121 2 :219 2 :145 2 : 45 2 : 35 2 :193 NA's: 16   
## 3 :218 3 :187 NA's:121 NA's:121 NA's:112   
## NA's: 16 NA's: 1   
##   
##   
## leaves leaf.halo leaf.marg leaf.size leaf.shread leaf.malf leaf.mild   
## 0: 77 0 :221 0 :357 0 : 51 0 :487 0 :554 0 :535   
## 1:606 1 : 36 1 : 21 1 :327 1 : 96 1 : 45 1 : 20   
## 2 :342 2 :221 2 :221 NA's:100 NA's: 84 2 : 20   
## NA's: 84 NA's: 84 NA's: 84 NA's:108   
##   
##   
##   
## stem lodging stem.cankers canker.lesion fruiting.bodies ext.decay   
## 0 :296 0 :520 0 :379 0 :320 0 :473 0 :497   
## 1 :371 1 : 42 1 : 39 1 : 83 1 :104 1 :135   
## NA's: 16 NA's:121 2 : 36 2 :177 NA's:106 2 : 13   
## 3 :191 3 : 65 NA's: 38   
## NA's: 38 NA's: 38   
##   
##   
## mycelium int.discolor sclerotia fruit.pods fruit.spots seed   
## 0 :639 0 :581 0 :625 0 :407 0 :345 0 :476   
## 1 : 6 1 : 44 1 : 20 1 :130 1 : 75 1 :115   
## NA's: 38 2 : 20 NA's: 38 2 : 14 2 : 57 NA's: 92   
## NA's: 38 3 : 48 4 :100   
## NA's: 84 NA's:106   
##   
##   
## mold.growth seed.discolor seed.size shriveling roots   
## 0 :524 0 :513 0 :532 0 :539 0 :551   
## 1 : 67 1 : 64 1 : 59 1 : 38 1 : 86   
## NA's: 92 NA's:106 NA's: 92 NA's:106 2 : 15   
## NA's: 31   
##   
##   
##

## See ?Soybean for details

### a. Investigate the frequency distributions for the categorical predictors. Are any of the distributions degenerate in the ways discussed earlier in this chapter?

All but one predictor contain NA values and can be evaluated to see if it is a degenerate distribution. There are no predictors where NA outnumber real values but there are some with a large number of NA. ‘hail’, ‘sever’, ‘seed.tmt’, and ‘germ’ have a large amount of NA values so we will explore how that impacts the data later.

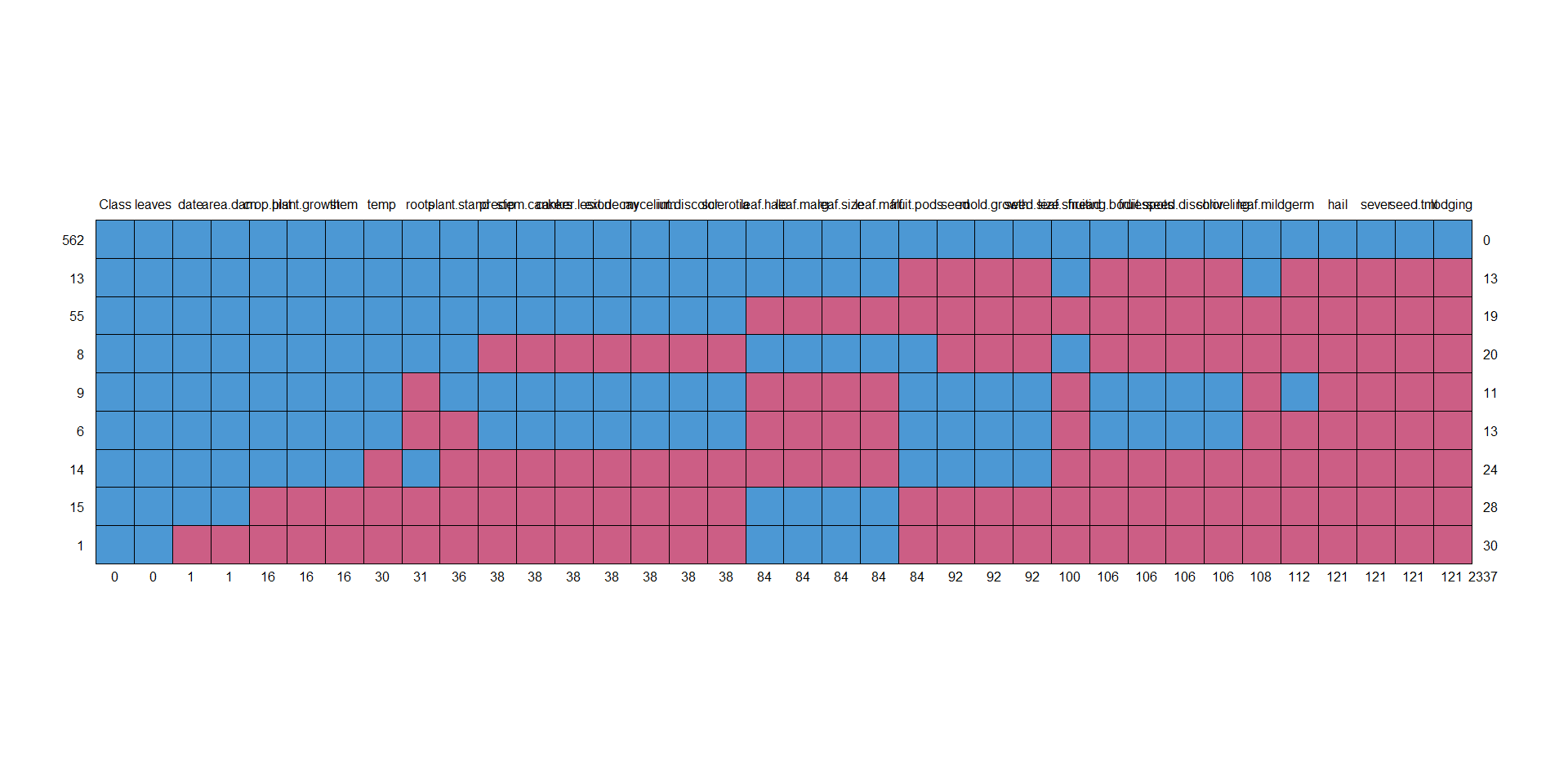
col\_names <- colnames(Soybean[-1])  
  
plot\_list <- list()  
  
for (i in col\_names){  
 plot <- ggplot(Soybean, aes\_string(Soybean[,i])) +  
 geom\_bar() +  
 xlab(colnames(Soybean[i]))   
 plot\_list[[i]] <- plot  
 #print(plot)  
}  
  
grid.arrange(grobs=plot\_list, ncol=4)



### b. Roughly 18 % of the data are missing. Are there particular predictors that are more likely to be missing? Is the pattern of missing data related to the classes?

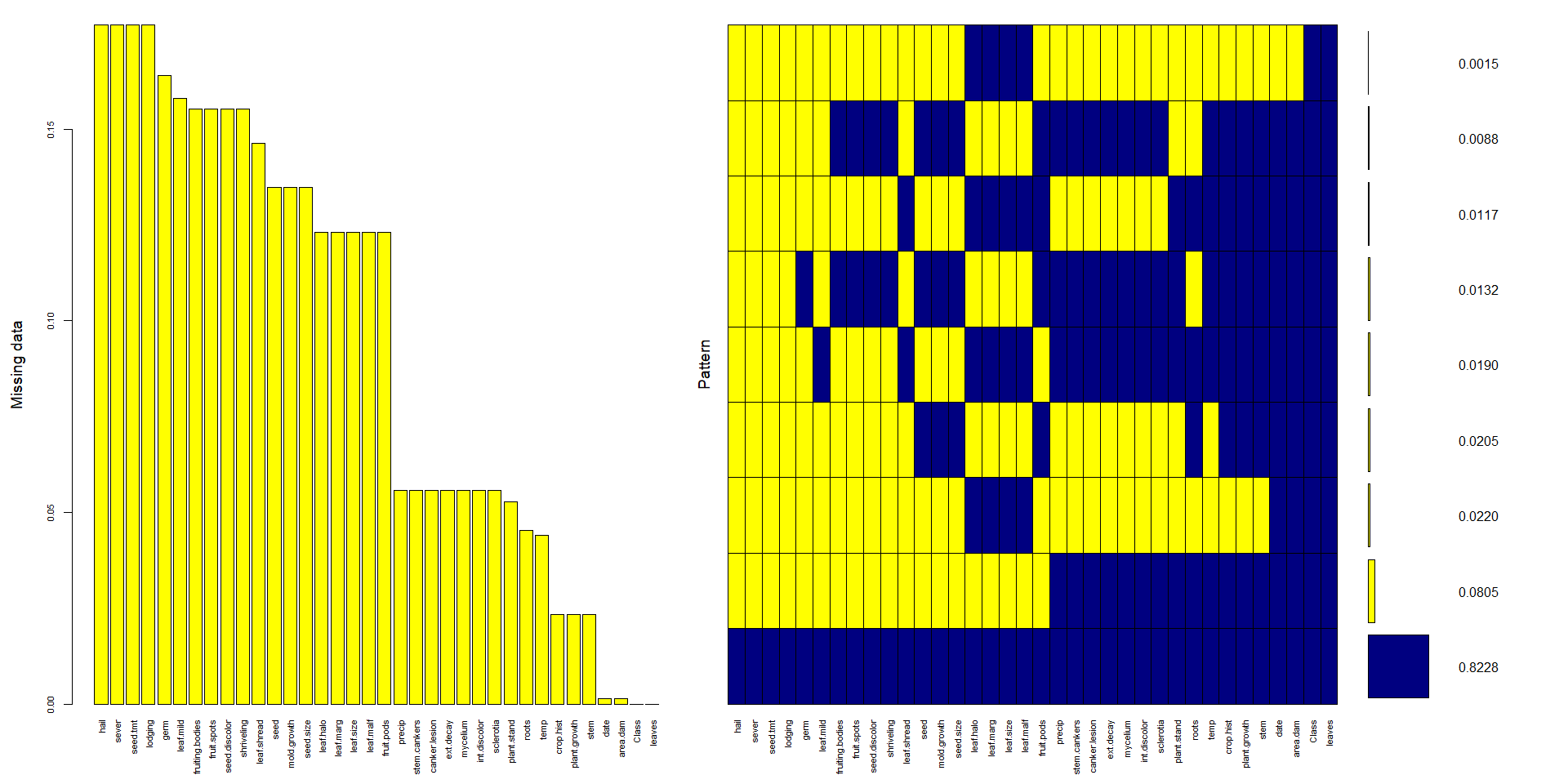
Of the 18% of missing data, 8% come from 19 out of 36 columns.

library(mice)  
library(VIM)  
md.pattern(Soybean)



## Class leaves date area.dam crop.hist plant.growth stem temp roots  
## 562 1 1 1 1 1 1 1 1 1  
## 13 1 1 1 1 1 1 1 1 1  
## 55 1 1 1 1 1 1 1 1 1  
## 8 1 1 1 1 1 1 1 1 1  
## 9 1 1 1 1 1 1 1 1 0  
## 6 1 1 1 1 1 1 1 1 0  
## 14 1 1 1 1 1 1 1 0 1  
## 15 1 1 1 1 0 0 0 0 0  
## 1 1 1 0 0 0 0 0 0 0  
## 0 0 1 1 16 16 16 30 31  
## plant.stand precip stem.cankers canker.lesion ext.decay mycelium  
## 562 1 1 1 1 1 1  
## 13 1 1 1 1 1 1  
## 55 1 1 1 1 1 1  
## 8 1 0 0 0 0 0  
## 9 1 1 1 1 1 1  
## 6 0 1 1 1 1 1  
## 14 0 0 0 0 0 0  
## 15 0 0 0 0 0 0  
## 1 0 0 0 0 0 0  
## 36 38 38 38 38 38  
## int.discolor sclerotia leaf.halo leaf.marg leaf.size leaf.malf fruit.pods  
## 562 1 1 1 1 1 1 1  
## 13 1 1 1 1 1 1 0  
## 55 1 1 0 0 0 0 0  
## 8 0 0 1 1 1 1 1  
## 9 1 1 0 0 0 0 1  
## 6 1 1 0 0 0 0 1  
## 14 0 0 0 0 0 0 1  
## 15 0 0 1 1 1 1 0  
## 1 0 0 1 1 1 1 0  
## 38 38 84 84 84 84 84  
## seed mold.growth seed.size leaf.shread fruiting.bodies fruit.spots  
## 562 1 1 1 1 1 1  
## 13 0 0 0 1 0 0  
## 55 0 0 0 0 0 0  
## 8 0 0 0 1 0 0  
## 9 1 1 1 0 1 1  
## 6 1 1 1 0 1 1  
## 14 1 1 1 0 0 0  
## 15 0 0 0 0 0 0  
## 1 0 0 0 0 0 0  
## 92 92 92 100 106 106  
## seed.discolor shriveling leaf.mild germ hail sever seed.tmt lodging   
## 562 1 1 1 1 1 1 1 1 0  
## 13 0 0 1 0 0 0 0 0 13  
## 55 0 0 0 0 0 0 0 0 19  
## 8 0 0 0 0 0 0 0 0 20  
## 9 1 1 0 1 0 0 0 0 11  
## 6 1 1 0 0 0 0 0 0 13  
## 14 0 0 0 0 0 0 0 0 24  
## 15 0 0 0 0 0 0 0 0 28  
## 1 0 0 0 0 0 0 0 0 30  
## 106 106 108 112 121 121 121 121 2337

aggr(Soybean, col=c('navyblue','yellow'),  
 numbers=TRUE, sortVars=TRUE,  
 labels=names(Soybean), cex.axis=.7,  
 gap=3, ylab=c("Missing data","Pattern"))



##   
## Variables sorted by number of missings:   
## Variable Count  
## hail 0.177159590  
## sever 0.177159590  
## seed.tmt 0.177159590  
## lodging 0.177159590  
## germ 0.163982430  
## leaf.mild 0.158125915  
## fruiting.bodies 0.155197657  
## fruit.spots 0.155197657  
## seed.discolor 0.155197657  
## shriveling 0.155197657  
## leaf.shread 0.146412884  
## seed 0.134699854  
## mold.growth 0.134699854  
## seed.size 0.134699854  
## leaf.halo 0.122986823  
## leaf.marg 0.122986823  
## leaf.size 0.122986823  
## leaf.malf 0.122986823  
## fruit.pods 0.122986823  
## precip 0.055636896  
## stem.cankers 0.055636896  
## canker.lesion 0.055636896  
## ext.decay 0.055636896  
## mycelium 0.055636896  
## int.discolor 0.055636896  
## sclerotia 0.055636896  
## plant.stand 0.052708638  
## roots 0.045387994  
## temp 0.043923865  
## crop.hist 0.023426061  
## plant.growth 0.023426061  
## stem 0.023426061  
## date 0.001464129  
## area.dam 0.001464129  
## Class 0.000000000  
## leaves 0.000000000

### c. Develop a strategy for handling missing data, either by eliminating predictors or imputation.

I chose to impute the data instead of eliminating predictors altogether. Predictors with NA values still had enough data where I didn’t feel that complete removal was necessary. Below is a comparison of the summary for the original dataset and the imputed dataset. After the imputation, the numbers do not differ too much but we would need to continue this analysis to see how much the imputation impacted the dataset.

impute\_soybean <- parlmice(Soybean, maxit = 5, m = 1, printFlag = FALSE, seed = 500, cluster.seed = 500)  
Soybean\_2 <- complete(impute\_soybean,1)

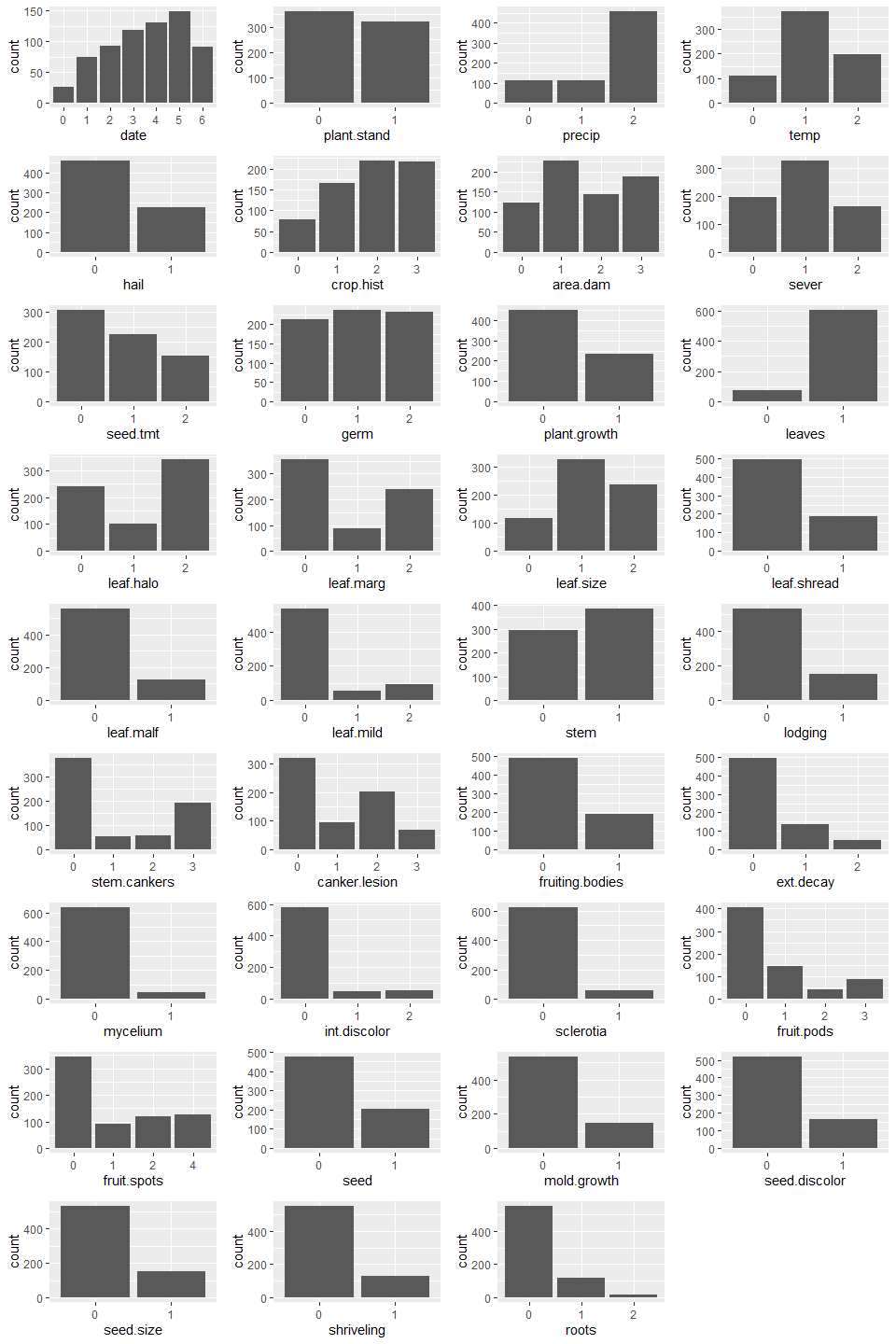
#original Soybean data  
summary(Soybean)

## Class date plant.stand precip temp   
## brown-spot : 92 5 :149 0 :354 0 : 74 0 : 80   
## alternarialeaf-spot: 91 4 :131 1 :293 1 :112 1 :374   
## frog-eye-leaf-spot : 91 3 :118 NA's: 36 2 :459 2 :199   
## phytophthora-rot : 88 2 : 93 NA's: 38 NA's: 30   
## anthracnose : 44 6 : 90   
## brown-stem-rot : 44 (Other):101   
## (Other) :233 NA's : 1   
## hail crop.hist area.dam sever seed.tmt germ plant.growth  
## 0 :435 0 : 65 0 :123 0 :195 0 :305 0 :165 0 :441   
## 1 :127 1 :165 1 :227 1 :322 1 :222 1 :213 1 :226   
## NA's:121 2 :219 2 :145 2 : 45 2 : 35 2 :193 NA's: 16   
## 3 :218 3 :187 NA's:121 NA's:121 NA's:112   
## NA's: 16 NA's: 1   
##   
##   
## leaves leaf.halo leaf.marg leaf.size leaf.shread leaf.malf leaf.mild   
## 0: 77 0 :221 0 :357 0 : 51 0 :487 0 :554 0 :535   
## 1:606 1 : 36 1 : 21 1 :327 1 : 96 1 : 45 1 : 20   
## 2 :342 2 :221 2 :221 NA's:100 NA's: 84 2 : 20   
## NA's: 84 NA's: 84 NA's: 84 NA's:108   
##   
##   
##   
## stem lodging stem.cankers canker.lesion fruiting.bodies ext.decay   
## 0 :296 0 :520 0 :379 0 :320 0 :473 0 :497   
## 1 :371 1 : 42 1 : 39 1 : 83 1 :104 1 :135   
## NA's: 16 NA's:121 2 : 36 2 :177 NA's:106 2 : 13   
## 3 :191 3 : 65 NA's: 38   
## NA's: 38 NA's: 38   
##   
##   
## mycelium int.discolor sclerotia fruit.pods fruit.spots seed   
## 0 :639 0 :581 0 :625 0 :407 0 :345 0 :476   
## 1 : 6 1 : 44 1 : 20 1 :130 1 : 75 1 :115   
## NA's: 38 2 : 20 NA's: 38 2 : 14 2 : 57 NA's: 92   
## NA's: 38 3 : 48 4 :100   
## NA's: 84 NA's:106   
##   
##   
## mold.growth seed.discolor seed.size shriveling roots   
## 0 :524 0 :513 0 :532 0 :539 0 :551   
## 1 : 67 1 : 64 1 : 59 1 : 38 1 : 86   
## NA's: 92 NA's:106 NA's: 92 NA's:106 2 : 15   
## NA's: 31   
##   
##   
##

#imputed soybean data  
summary(Soybean\_2)

## Class date plant.stand precip temp hail   
## brown-spot : 92 0: 26 0:362 0:112 0:110 0:460   
## alternarialeaf-spot: 91 1: 75 1:321 1:112 1:374 1:223   
## frog-eye-leaf-spot : 91 2: 93 2:459 2:199   
## phytophthora-rot : 88 3:118   
## anthracnose : 44 4:131   
## brown-stem-rot : 44 5:149   
## (Other) :233 6: 91   
## crop.hist area.dam sever seed.tmt germ plant.growth leaves leaf.halo  
## 0: 79 0:123 0:195 0:306 0:212 0:451 0: 77 0:239   
## 1:166 1:228 1:326 1:224 1:238 1:232 1:606 1:102   
## 2:220 2:145 2:162 2:153 2:233 2:342   
## 3:218 3:187   
##   
##   
##   
## leaf.marg leaf.size leaf.shread leaf.malf leaf.mild stem lodging  
## 0:357 0:118 0:497 0:559 0:536 0:296 0:533   
## 1: 88 1:328 1:186 1:124 1: 54 1:387 1:150   
## 2:238 2:237 2: 93   
##   
##   
##   
##   
## stem.cankers canker.lesion fruiting.bodies ext.decay mycelium int.discolor  
## 0:379 0:320 0:492 0:497 0:639 0:581   
## 1: 53 1: 94 1:191 1:136 1: 44 1: 46   
## 2: 58 2:202 2: 50 2: 56   
## 3:193 3: 67   
##   
##   
##   
## sclerotia fruit.pods fruit.spots seed mold.growth seed.discolor seed.size  
## 0:625 0:407 0:345 0:477 0:537 0:520 0:532   
## 1: 58 1:146 1: 92 1:206 1:146 1:163 1:151   
## 2: 42 2:121   
## 3: 88 4:125   
##   
##   
##   
## shriveling roots   
## 0:552 0:551   
## 1:131 1:116   
## 2: 16   
##   
##   
##   
##

col\_names <- colnames(Soybean\_2[-1])  
  
plot\_list <- list()  
  
for (i in col\_names){  
 plot <- ggplot(Soybean\_2, aes\_string(Soybean\_2[,i])) +  
 geom\_bar() +  
 xlab(colnames(Soybean\_2[i]))   
 plot\_list[[i]] <- plot  
 #print(plot)  
}  
  
grid.arrange(grobs=plot\_list, ncol=4)



## KJ 635

## HA 7.1 - Exponential Smoothing

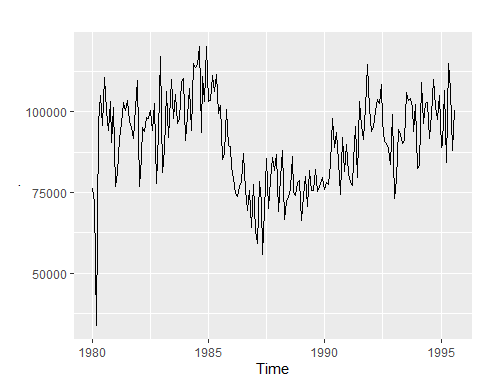
### Consider the pigs series — the number of pigs slaughtered in Victoria each month.

View the data set

pigs

## Jan Feb Mar Apr May Jun Jul Aug Sep Oct  
## 1980 76378 71947 33873 96428 105084 95741 110647 100331 94133 103055  
## 1981 76889 81291 91643 96228 102736 100264 103491 97027 95240 91680  
## 1982 76892 85773 95210 93771 98202 97906 100306 94089 102680 77919  
## 1983 81225 88357 106175 91922 104114 109959 97880 105386 96479 97580  
## 1984 90974 98981 107188 94177 115097 113696 114532 120110 93607 110925  
## 1985 103069 103351 111331 106161 111590 99447 101987 85333 86970 100561  
## 1986 82719 79498 74846 73819 77029 78446 86978 75878 69571 75722  
## 1987 63292 59380 78332 72381 55971 69750 85472 70133 79125 85805  
## 1988 69069 79556 88174 66698 72258 73445 76131 86082 75443 73969  
## 1989 66269 73776 80034 70694 81823 75640 75540 82229 75345 77034  
## 1990 75982 78074 77588 84100 97966 89051 93503 84747 74531 91900  
## 1991 81022 78265 77271 85043 95418 79568 103283 95770 91297 101244  
## 1992 93866 95171 100183 103926 102643 108387 97077 90901 90336 88732  
## 1993 73292 78943 94399 92937 90130 91055 106062 103560 104075 101783  
## 1994 82413 83534 109011 96499 102430 103002 91815 99067 110067 101599  
## 1995 88905 89936 106723 84307 114896 106749 87892 100506   
## Nov Dec  
## 1980 90595 101457  
## 1981 101259 109564  
## 1982 93561 117062  
## 1983 109490 110191  
## 1984 103312 120184  
## 1985 89543 89265  
## 1986 64182 77357  
## 1987 81778 86852  
## 1988 78139 78646  
## 1989 78589 79769  
## 1990 81635 89797  
## 1991 114525 101139  
## 1992 83759 99267  
## 1993 93791 102313  
## 1994 97646 104930  
## 1995

help(pigs)  
pigs %>% autoplot()

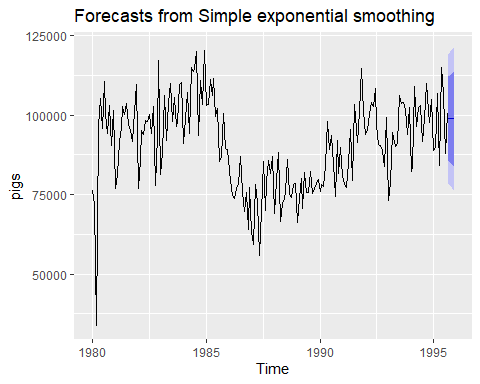


**Alternative Description** The Simple Exponential Method (“SES”) method is suitable to forecast time series with no clear trend or seasonality. The Pigs data does not have a clear pattern and therefore we will use SES.

### a. Use the ses() function in R to find the optimal values of and and generate forecasts for the next four months.

Using the summary() function and .

#estimate parameter - the next four months  
fc<- ses(pigs, h=4)  
  
#timeseries plot with autoplot() function  
fc %>%  
 autoplot()



summary(fc)

##   
## Forecast method: Simple exponential smoothing  
##   
## Model Information:  
## Simple exponential smoothing   
##   
## Call:  
## ses(y = pigs, h = 4)   
##   
## Smoothing parameters:  
## alpha = 0.2971   
##   
## Initial states:  
## l = 77260.0561   
##   
## sigma: 10308.58  
##   
## AIC AICc BIC   
## 4462.955 4463.086 4472.665   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE ACF1  
## Training set 385.8721 10253.6 7961.383 -0.922652 9.274016 0.7966249 0.01282239  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## Sep 1995 98816.41 85605.43 112027.4 78611.97 119020.8  
## Oct 1995 98816.41 85034.52 112598.3 77738.83 119894.0  
## Nov 1995 98816.41 84486.34 113146.5 76900.46 120732.4  
## Dec 1995 98816.41 83958.37 113674.4 76092.99 121539.8

#### b. Compute a 95% prediction interval for the first forecast using where is the standard deviation of the residuals. Compare your interval with the interval produced by R.

From chapter 3.5. Prediction intervals are calculated as . The multiplier for 95% interval is 1.96 and the residuals are also equal to the RMSE. Using the SES model formula from part a, the $ values are stored in a vector as are the residuals. Then, the values are subbed in for the formula.

The computed intervals vs. the predicted intervals are quite close. The September 1995 low interval at 95% is 78611.97, compared to the computed 78679.97 only a difference of 68. The high values for the predicted interval at 95% is 119020.80, compared to 118952.84 again differing by 68.

y\_hat <- c(1.96, -1.96)  
s <- sd(residuals(fc))  
  
ses(pigs, h=4)$mean[1]+(y\_hat\*s)

## [1] 118952.84 78679.97

## HA 7.2

#### Write your own function to implement simple exponential smoothing. The function should take arguments y (the time series), alpha (the smoothing parameter α) and level (the initial level ℓ0). It should return the forecast of the next observation in the series. Does it give the same forecast as ses()?

## HA 7.3

#### Modify your function from the previous exercise to return the sum of squared errors rather than the forecast of the next observation. Then use the optim() function to find the optimal values of α and ℓ0. Do you get the same values as the ses() function?

## HA 8.1

## HA 8.2

## HA 8.6

## HA 8.8