Homework 1

Leonel Carlen Stefan Ciric

October 15, 2019

Install and load tidyverse

```
#install.packages("tidyverse")
library("tidyverse")
```

```
## -- Attaching packages ---------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.2.0     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
## Parsed with column specification:
## cols(
##   season = col_character(),
##   size = col_character(),
##   speed = col_character(),
##   mxPH = col_double(),
##   mnO2 = col_double(),
##   Cl = col_double(),
##   NO3 = col_double(),
##   NH4 = col_double(),
##   oPO4 = col_double(),
##   PO4 = col_double(),
##   Chla = col_double(),
##   a1 = col_double(),
##   a2 = col_double(),
##   a3 = col_double(),
##   a4 = col_double(),
##   a5 = col_double(),
##   a6 = col_double(),
##   a7 = col_double()
## )
```

Problem 1

##1.a

```
algae%>%
  group_by(season)%>%
  summarize(n = n())
```

```
## # A tibble: 4 x 2
##   season     n
```

```
##   <chr>  <int>
## 1 autumn    40
## 2 spring    53
## 3 summer    45
## 4 winter    62
```

1.b

```
notNA <- algae %>%
  summarise_at(.funs=funs(sum(!is.na(.))), .vars = vars(mxPH:Chla))
```

```
## Warning: funs() is soft deprecated as of dplyr 0.8.0
## please use list() instead
##
##   # Before:
##   funs(name = f(.))
##
##   # After:
##   list(name = ~ f(.))
## This warning is displayed once per session.
```

```
#Compute mean of each chemical
chemMean <- algae %>%
  summarise_at(vars(mxPH:Chla), mean, na.rm=TRUE)
#Computer var of each chemical
chemVar <- algae %>%
  summarise_at(vars(mxPH:Chla), var, na.rm=TRUE)
Names <- c("Count", "Mean", "Variance")
cbind(Names, rbind(notNA, chemMean, chemVar))
```

```
##      Names        mxPH       mnO2         Cl        NO3          NH4
## 1    Count 199.0000000 198.000000  190.00000 198.000000     198.0000
## 2     Mean   8.0117337   9.117778   43.63628   3.282389     501.2958
## 3 Variance   0.3579693   5.718089 2193.17173  14.261756 3851584.6849
##        oPO4        PO4     Chla
## 1  198.0000   198.0000 188.0000
## 2   73.5906   137.8821  13.9712
## 3 8305.8499 16639.3845 420.0827
```

```
#NH4, oPO4 and PO4 have very large standard deviations, which indicate the mean isn't very useful.
```

1.c

```
# Compute median of each chemical
chemMed <- algae %>%
  dplyr::select(mxPH:Chla) %>%
  summarise_all(function(z) median(z, na.rm=TRUE))
# Compute mad of each chemical
chemMad <- algae %>%
  summarise_at(vars(mxPH:Chla), funs(mad), na.rm = TRUE)
rnames <- list("Med", "Mean", "Mad", "StD")
rbind(chemMed, chemMean, chemMad, sqrt(chemVar))
```

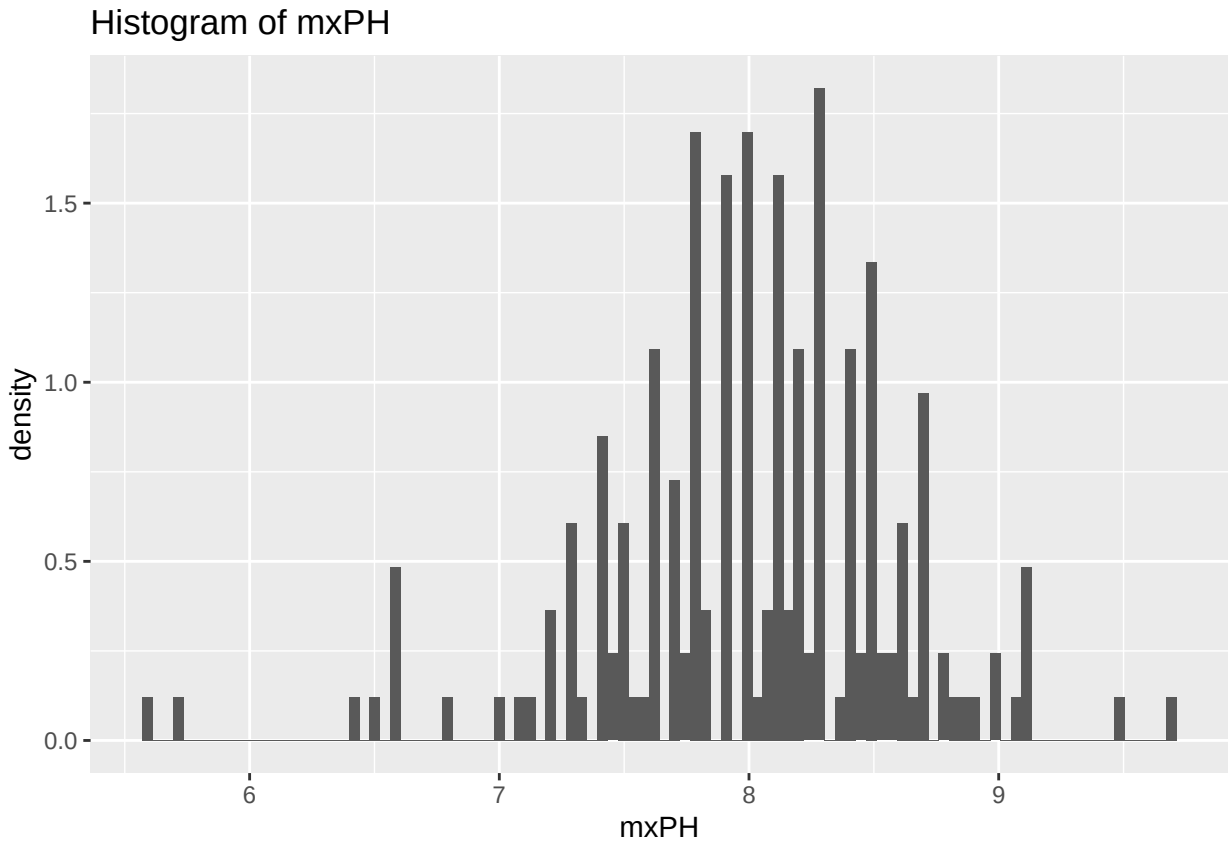
```
## # A tibble: 4 x 8
##    mxPH  mnO2    Cl   NO3   NH4  oPO4   PO4  Chla
```

```
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 8.06   9.8   32.7  2.68  103.  40.2  103.  5.48
## 2 8.01   9.12  43.6  3.28  501.  73.6  138. 14.0
## 3 0.504  2.05  33.2  2.17  112.  44.0  122.  6.67
## 4 0.598  2.39  46.8  3.78 1963.  91.1  129. 20.5
```

```
# The mean is generally higher than the median for each attribute.
# The standard deviation is larger than the mad.
#These differences imply that extreme high values skew the average
#and that the data is more broadly spread around the mean than it is around the median.
# The much smaller MAD vs. the StD indicates the presence of influential points, potentially outliers.
```

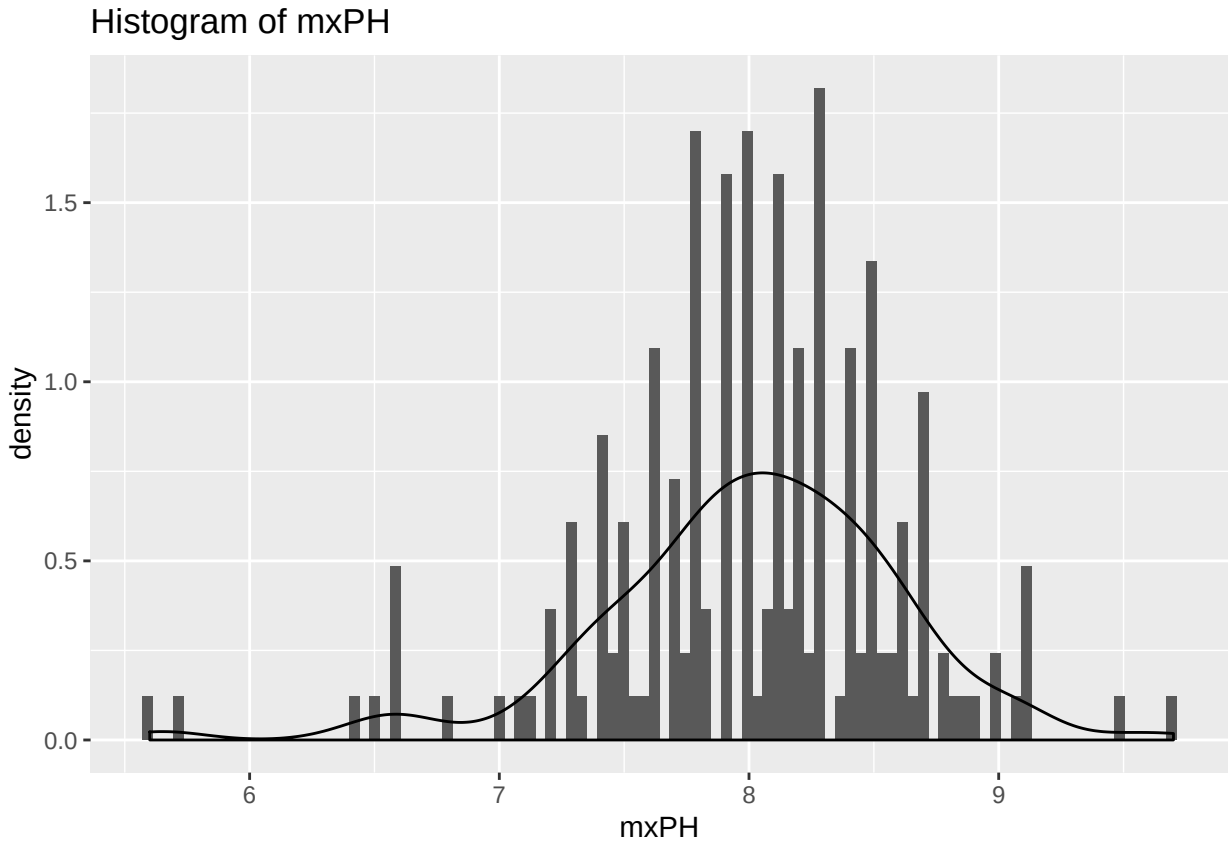
#2 ##2.a

```
mxphPlot <- algae%>%
  drop_na(mxPH)%>%
  ggplot(aes(mxPH,stat(density))) +
  geom_histogram(bins = 100) + ggtitle("Histogram of mxPH")
mxphPlot
```



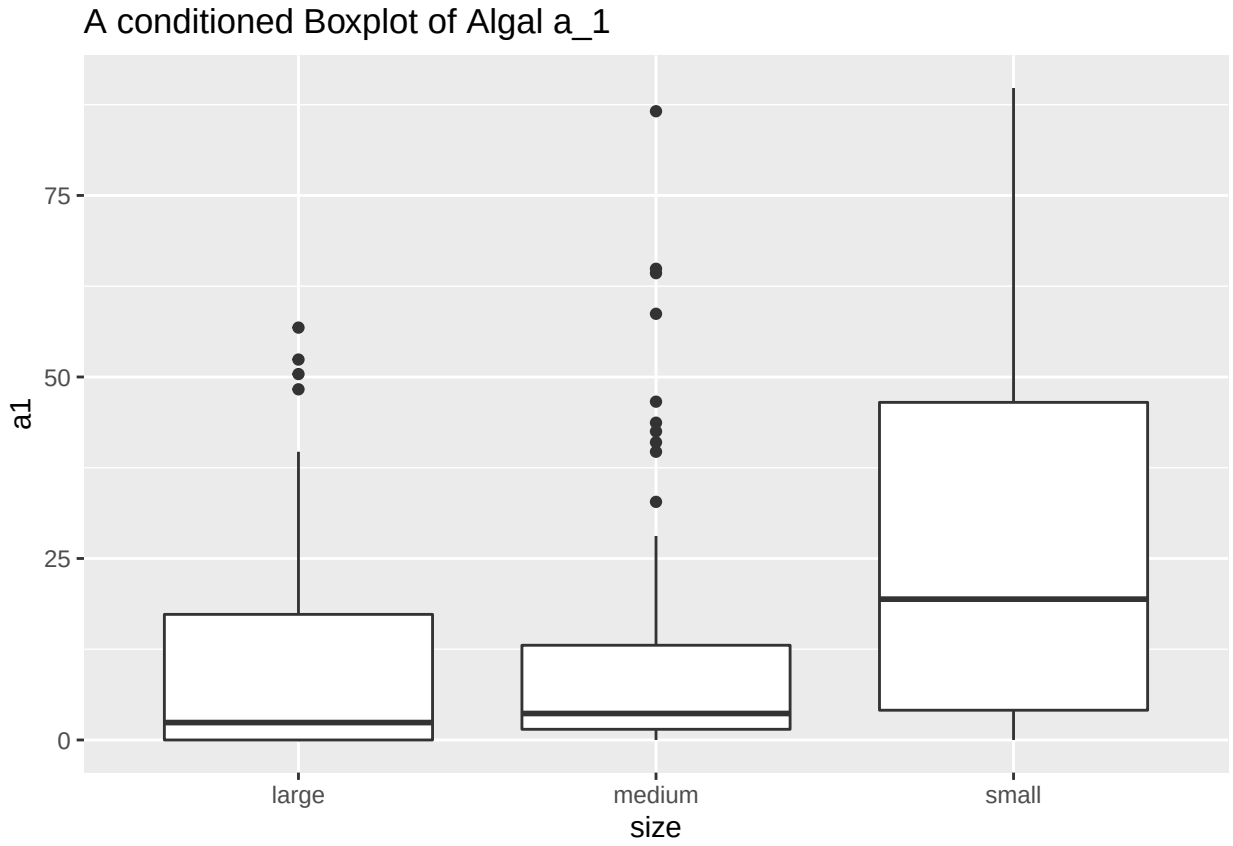
##2.b

```
algae%>%
  drop_na(mxPH)%>%
  ggplot(aes(mxPH,stat(density))) +
  geom_histogram(bins = 100) + ggtitle("Histogram of mxPH") + geom_density(inherit.aes = TRUE)
```



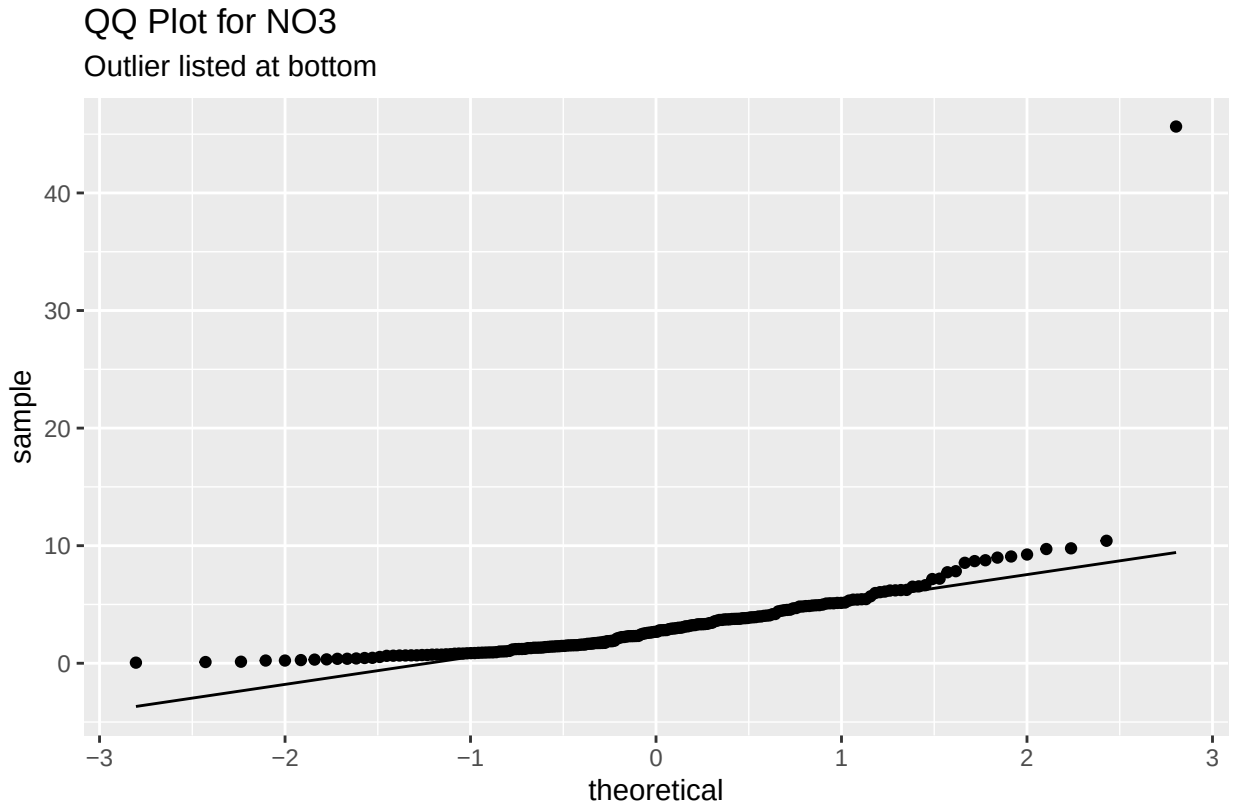
##2.c

```
a1Box <- ggplot() + geom_boxplot(data=algae, aes(y=a1, x=size)) + ggtitle('A conditioned Boxplot of Alg
a1Box
```



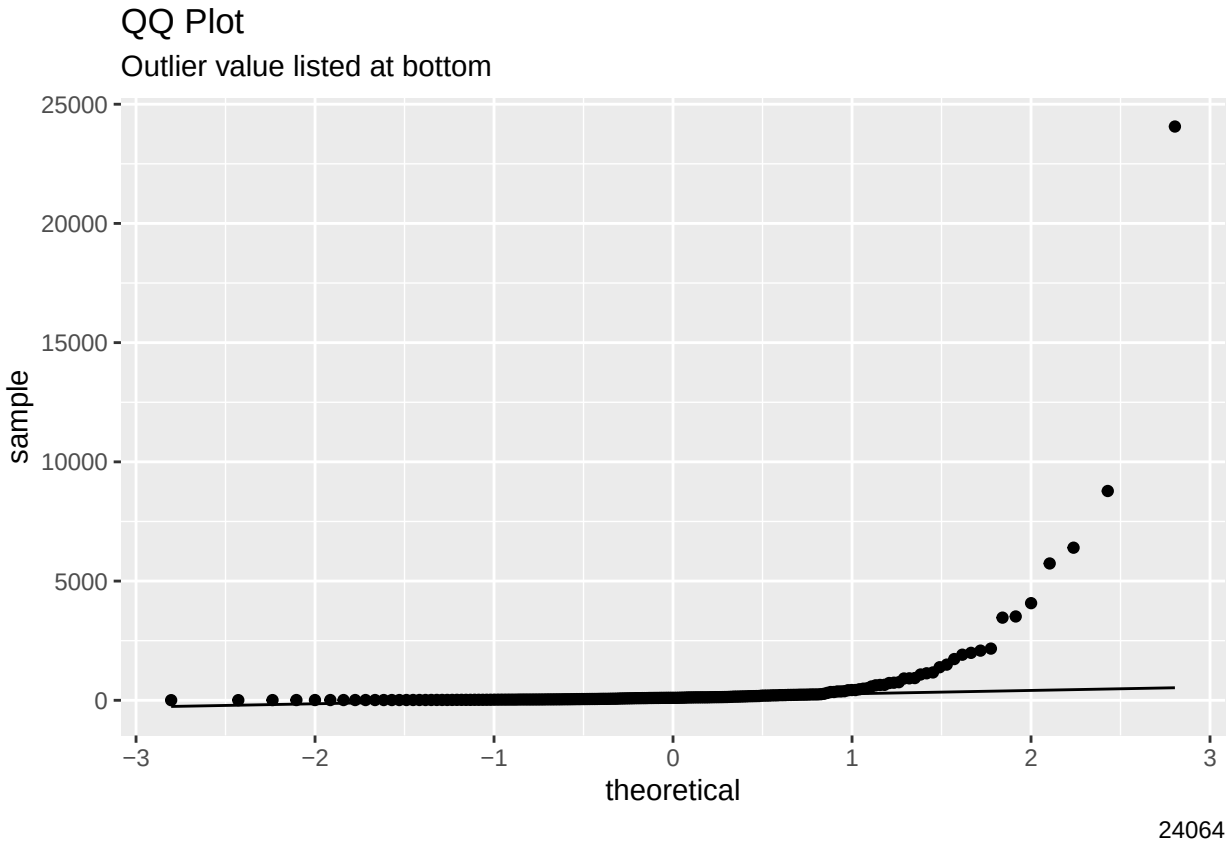
##2.d

```
# Use ggplot function stat_qq() and stat_qq_line to find outliers in NO3
outNO3 <- algae%>%
  drop_na(NO3)%>%
  ggplot(aes(sample = NO3)) + stat_qq()+stat_qq_line()+labs(title = "QQ Plot for NO3", subtitle = "Outli
outNO3
```



45.65

```
# Find outlier in NH4
outNO4 <- algae%>%
  drop_na(NH4)%>%
  ggplot(aes(sample = NH4)) + stat_qq()+stat_qq_line()+ labs(title = "QQ Plot", subtitle = "Outlier valu
outNO4
```



2.e

```
# Compute median of each chemical
medNO3NH4 <- algae%>%
  dplyr::select(NO3, NH4)%>%
  summarise_all(function(z) median(z, na.rm=TRUE))
# Compute mad of each chemical
madNO3NH4 <- algae%>%
  summarise_at(vars(NO3, NH4), funs(mad), na.rm = TRUE)
#Compute mean of each chemical
meanNO3NH4 <- algae%>%
  summarise_at(vars(NO3, NH4), mean, na.rm=TRUE)
#Computer var of each chemical
varNO3NH4 <- algae%>%
  summarise_at(vars(NO3, NH4), var, na.rm=TRUE)
myTable <- rbind(medNO3NH4,meanNO3NH4, madNO3NH4,sqrt(varNO3NH4))
Stat <- c("Med", "Mean", "Mad", "Std")
nTable <- cbind(Stat, myTable)
nTable
```

```
##   Stat      NO3       NH4
## 1  Med 2.675000  103.1665
## 2 Mean 3.282389  501.2958
## 3  Mad 2.172009  111.6175
## 4  Std 3.776474 1962.5455
```

```
# As expected, the mean is much higher than the median and the standard deviation is much higher than t
```

3

3.a

```
fAlgae <- filter(algae, is.na(mxPH)|is.na(mnO2)|is.na(Cl)|is.na(NO3)|is.na(NH4)|is.na(oPO4)|is.na(PO4)|

cat("The number of observations that contain one or more missing values is", nrow(fAlgae), "\n")
```

```
## The number of observations that contain one or more missing values is 16
```

```
isNA = notNA
for(i in 1:length(notNA))
  isNA[[i]] = 200 - notNA[[i]]
print("The number of missing values in each column is listed in the table below:")
```

```
## [1] "The number of missing values in each column is listed in the table below:"
```

```
isNA
```

```
## # A tibble: 1 x 8
##    mxPH  mnO2    Cl   NO3   NH4  oPO4   PO4  Chla
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     1     2    10     2     2     2     2    12
```

3.b

```
algae.del <- filter(algae, !is.na(mxPH)&!is.na(mnO2)&!is.na(Cl)&!is.na(NO3)&!is.na(NH4)&!is.na(oPO4)&!is
cat("There are",nrow(algae.del),"observations without missing values in the dataset.")
```

```
## There are 184 observations without missing values in the dataset.
```

3.c

```
algae.med <- algae%>%
  mutate_at(.vars = vars(4:11), .funs = funs(ifelse(is.na(.), median(., na.rm = TRUE), .)))
print("The number of observations in algae.med is")
```

```
## [1] "The number of observations in algae.med is"
```

```
nrow(algae.med)
```

```
## [1] 200
```

```
print("The chemicals for the 48th, 62nd, and 199th rows are displayed in the table below")
```

```
## [1] "The chemicals for the 48th, 62nd, and 199th rows are displayed in the table below"
```

```
Row <- c(48, 62, 199)
cbind(Row, rbind(algae.med[48,4:11], algae.med[62,4:11], algae.med[199,4:11]))
```

```
##   Row mxPH mnO2    Cl   NO3      NH4  oPO4      PO4  Chla
## 1  48 8.06 12.6  9.00 0.230  10.0000  5.00   6.0000 1.100
## 2  62 6.40  9.8 32.73 2.675 103.1665 40.15  14.0000 5.475
## 3 199 8.00  7.6 32.73 2.675 103.1665 40.15 103.2855 5.475
```

3.d

```
require(utils)
#pairs(algae[4:11])
x <- algae.del[4:11]
x.cor <- cor(x)
x.cor
```

```
##             mxPH        mnO2          Cl        NO3         NH4
## mxPH  1.00000000 -0.10269374  0.14709539 -0.1721302 -0.15429757
## mnO2 -0.10269374  1.00000000 -0.26324536  0.1179077 -0.07826816
## Cl    0.14709539 -0.26324536  1.00000000  0.2109583  0.06598336
## NO3  -0.17213024  0.11790769  0.21095831  1.0000000  0.72467766
## NH4  -0.15429757 -0.07826816  0.06598336  0.7246777  1.00000000
## oPO4  0.09022909 -0.39375269  0.37925596  0.1330145  0.21931121
## PO4   0.10132957 -0.46396073  0.44519118  0.1570297  0.19939575
## Chla  0.43182377 -0.13121671  0.14295776  0.1454929  0.09120406
##             oPO4        PO4        Chla
## mxPH  0.09022909  0.1013296  0.43182377
## mnO2 -0.39375269 -0.4639607 -0.13121671
## Cl    0.37925596  0.4451912  0.14295776
## NO3   0.13301452  0.1570297  0.14549290
## NH4   0.21931121  0.1993958  0.09120406
## oPO4  1.00000000  0.9119646  0.10691478
## PO4   0.91196460  1.0000000  0.24849223
## Chla  0.10691478  0.2484922  1.00000000
```

```
reg <- lm(algae$PO4~algae$oPO4)
algae$PO4[28] <- predict(reg)[28]
algae$PO4[28]
```

```
## [1] 76.51663
```

3.e

The case of the military plane example presented during lecture informs the need for human analysis to consider why missing values may be missing – it could be the result of an extreme, or some other situation that needs human intelligence to probe.

4

##4.a

```
id <- rep(1:5, each = 40)
id <- sample(id)
almed1 <- cbind(id, algae.med)
```

The recommended code didn't work for me and I couldn't get help on Piazza. * {r chunkids} set.seed(500) almed = algae.med %>% select(-c(season, size, speed)) cut(1:nrow(almed), 5, label=FALSE) %>% sample*

4.b

```
#do.chunk2 <- function(chunkid, chunkdef, dat)
 # Xtr = algae.med[almed1$id != 1, 1:11]
```

```
errors <- data.frame("fold"=NULL, "train.error"=NULL, "val.error"=NULL)
dat <- almed1
for(i in 1:5){
    Xtr = dat[dat$id != i,2:12]  # get training set
    Ytr = dat[dat$id != i,13]  # get true response values in trainig set
    Xvl = dat[!(dat$id != i),2:12]  # get validation set
    Yvl = dat[!(dat$id != i),13]  # get true response values in validation set
    lm.a1 <- lm(a1~., dat[dat$id!=i,2:13])
    predYtr = predict(lm.a1)  # predict training values
    predYtr
    predYvl = predict(lm.a1,Xvl)  # predict validation values
    output <- list(i, mean((predYtr - Ytr)^2), # compute and store training error
                mean((predYvl - Yvl)^2))   # compute and store test error
    errors <- rbind(errors, output)
}
colnames(errors) <- c("Fold", "Training Error", "Test Error")
errors
```

```
##   Fold Training Error Test Error
## 1    1       293.8760   288.0461
## 2    2       274.5309   369.2433
## 3    3       310.1495   433.9997
## 4    4       263.5081   398.5699
## 5    5       260.5010   421.9111
```

5

```
alTest <- read_table2('algaeTest.txt',
                      col_names=c('season','size','speed','mxPH','mnO2','Cl','NO3',
                                  'NH4','oPO4','PO4','Chla','a1'),
                      na=c('XXXXXXX'))
```

```
## Parsed with column specification:
## cols(
##   season = col_character(),
##   size = col_character(),
##   speed = col_character(),
##   mxPH = col_double(),
##   mnO2 = col_double(),
##   Cl = col_double(),
##   NO3 = col_double(),
##   NH4 = col_double(),
##   oPO4 = col_double(),
##   PO4 = col_double(),
##   Chla = col_double(),
##   a1 = col_double()
## )
```

##5.a

```
tSet = algae.med[,1:11]
vSet = alTest[,1:11]
vSet2 = alTest[,12]
```

```
lmAll <- lm(a1~., data = algae.med[,1:12])
predvSet = predict(lmAll, vSet)
sum((predvSet-vSet2)^2)/length(predvSet)
```

```
## [1] 250.1794
```

```
#MSE

#It definitely fits in with the values in part 4, however, I'm skeptical about my code because it didn'
```

6

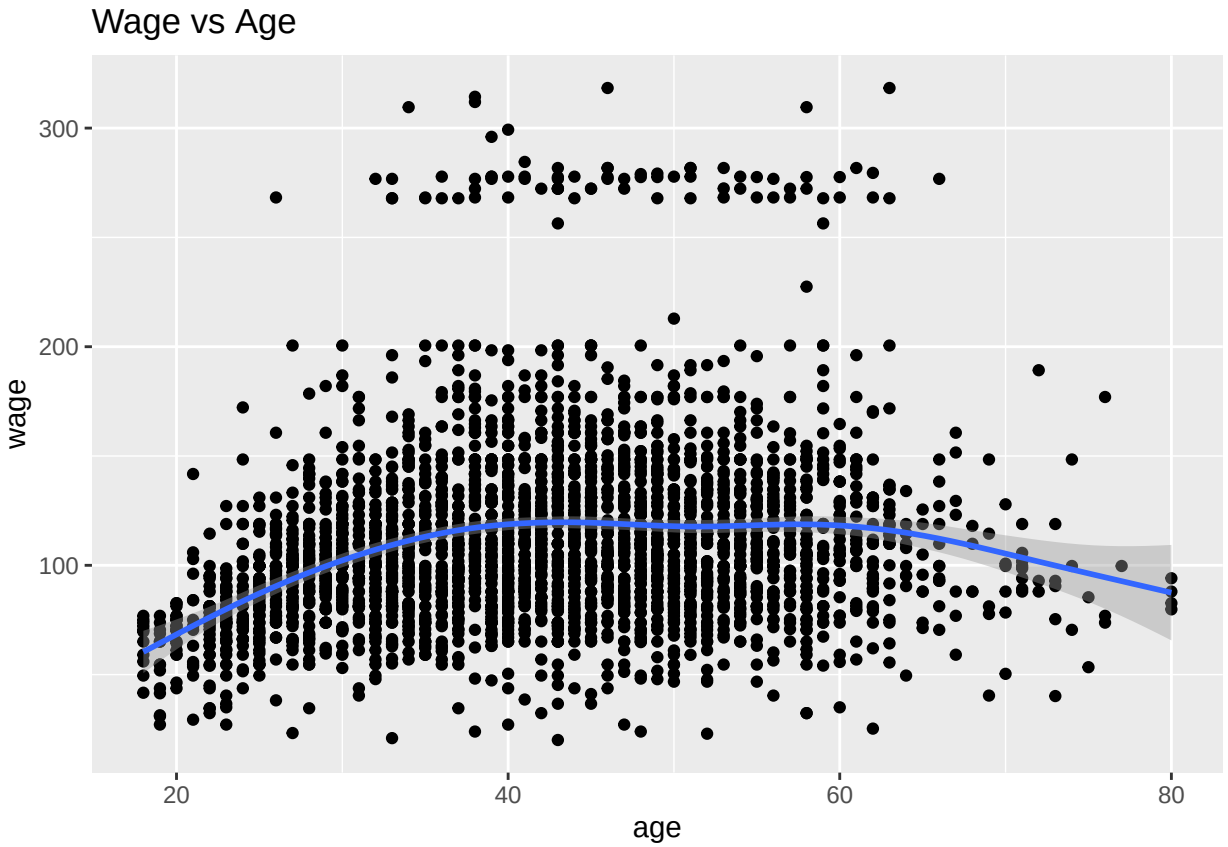
```
library(ISLR)
head(Wage)
```

```
##        year age           maritl     race       education
## 231655 2006  18 1. Never Married 1. White    1. < HS Grad
## 86582  2004  24 1. Never Married 1. White 4. College Grad
## 161300 2003  45       2. Married 1. White 3. Some College
## 155159 2003  43       2. Married 3. Asian 4. College Grad
## 11443  2005  50      4. Divorced 1. White      2. HS Grad
## 376662 2008  54       2. Married 1. White 4. College Grad
##                    region       jobclass          health health_ins
## 231655 2. Middle Atlantic  1. Industrial      1. <=Good      2. No
## 86582  2. Middle Atlantic 2. Information 2. >=Very Good      2. No
## 161300 2. Middle Atlantic  1. Industrial      1. <=Good     1. Yes
## 155159 2. Middle Atlantic 2. Information 2. >=Very Good     1. Yes
## 11443  2. Middle Atlantic 2. Information      1. <=Good     1. Yes
## 376662 2. Middle Atlantic 2. Information 2. >=Very Good     1. Yes
##         logwage      wage
## 231655 4.318063  75.04315
## 86582  4.255273  70.47602
## 161300 4.875061 130.98218
## 155159 5.041393 154.68529
## 11443  4.318063  75.04315
## 376662 4.845098 127.11574
```

6.a

```
 ExpSalary = ggplot(Wage, aes(x=age, y=wage))
    ExpSalary + geom_point() + geom_smooth()+ggtitle("Wage vs Age")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
    print("Apart from a few outliers, it seems that wages rise with age up till a peak point around 40-5
```

```
## [1] "Apart from a few outliers, it seems that wages rise with age up till a peak point around 40-50 y
```

6.b

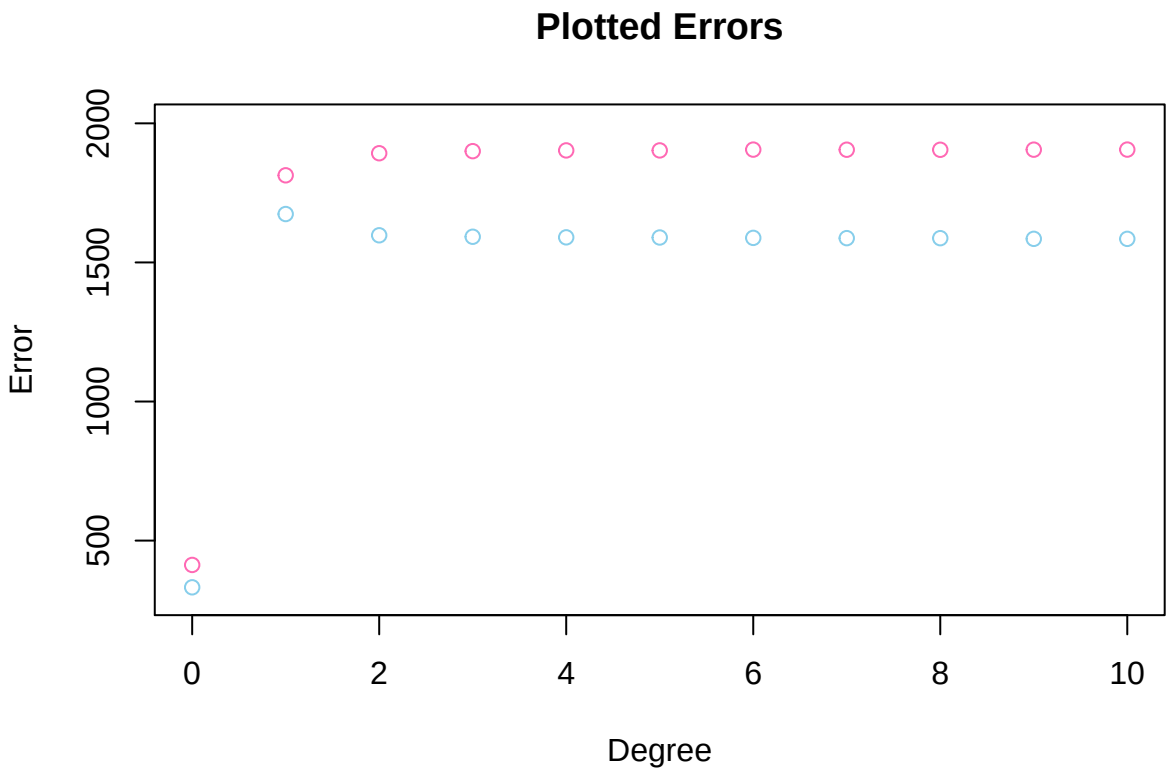
```
modelErrors <- data.frame("Model"=NULL, "Train Error"=NULL, "Test Error"=NULL)
nums <- rep(1:5, each = length(Wage)/5)
id <- sample(nums)
age <- Wage$age
wage <- Wage$wage
data <- data.frame("ID" = id, "AGE" = age, "WAGE" = wage)
for (i in 1:5){
  sumtrain = 0
  sumtest = 0
    inTrain = data[data[,1]!=i,2]
    outTrain = data[data[,1]!=i,3]
    inTest = data[!(data[,1]!=i),2:3]
    outTest = data[!(data[1]!=i),3]
    fit <- lm(WAGE~1, data[data[,1]!=i,2:3])
    length(data)
    pTrain = predict(fit)
    pTest = predict(fit, inTest)
    sumtrain <- sumtrain + mean((pTrain - outTrain)^2)
    sumtest <- sumtest + mean((pTest - outTest)^2)
}
modelErrors <- rbind(modelErrors, list(0, sumtrain/5, sumtest/5))
```

```
for(j in 1:10){
  sumtrain = 0
  sumtest = 0
  for (i in 1:5){
    inTrain = data[data[,1]!=i,2]
    outTrain = data[data[,1]!=i,3]
    inTest = data.frame(data[data[,1]!=i, 2])
    outTest = data[!(data[1]!=i),3]
    fit <- lm(data[data[,1]!=i, 3]~poly(data[data[,1]!=i, 2], j, raw = FALSE), data = data)
    pTrain = predict(fit)
    pTest = predict(fit, inTest)
    sumtrain <- sumtrain + mean((pTrain - outTrain)^2)
    sumtest <- sumtest + mean((pTest - outTest)^2)
  }
  modelErrors <- rbind(modelErrors, list(j, sumtrain/5, sumtest/5))
}
colnames(modelErrors) <- list("Degree", "Train Error", "Test Error")
modelErrors
```

```
##    Degree Train Error Test Error
## 1       0    332.1048   412.5337
## 2       1   1673.9714  1813.1836
## 3       2   1597.6285  1892.6123
## 4       3   1592.2990  1900.0750
## 5       4   1590.1683  1902.6895
## 6       5   1589.6595  1902.7625
## 7       6   1588.3078  1905.5763
## 8       7   1587.3546  1905.1819
## 9       8   1587.2329  1905.0550
## 10      9   1584.8404  1905.3800
## 11     10   1584.7766  1905.6661
```

##6.c

```
plot(modelErrors$Degree, modelErrors$`Train Error`, col = 'skyblue', ylab = "Error", xlab = "Degree", yl
points(modelErrors$Degree, modelErrors$`Test Error`, col = 'hotpink')
```



It looks like they are all equally bad except for the intercept-only mode. But I think that's because my code is bad.