HW 4 markdown

Steve Harms

September 24, 2017

Exercise 1

a)

```
#Read in the data
votes <- read.table(file = "http://maitra.public.iastate.edu/stat579/datasets/senate-109.txt", sep = "\t", header = TRUE)

#Give some names to our columns
names(votes$bill_type_bill_name_bill_ID)= "type.name.ID"
names(votes$missing_votes)= "missing"
#Make sure we have the right format for our first column
votes$bill_type_bill_name_bill_ID <- as.character(votes$bill_type_bill_name_bill_ID)
```

b)

i)

```
#Use sub() to get everything before the first _, combine it all into one vector
billtype <- cbind(sub("_.*$", "", votes$bill_type_bill_name_bill_ID))
colnames(billtype) = "billtype"
```

ii)

```
#A frequency table of the types of bills (it's a pretty big list)
table(billtype)
```

```
## billtype
##                      Abortion Issues                   Agriculture Issues 
##                                   17                                    6 
##                       Appropriations                  Arts and Humanities 
##                                   93                                    1 
##           Budget, Spending and Taxes               Business and Consumers 
##                                   69                                    7 
## Campaign Finance and Election Issues                      Civil Liberties 
##                                   13                                    7 
##                Congressional Affairs                         Crime Issues 
##                                    9                                    3 
##                              Defense                          Drug Issues 
##                                   24                                    2 
##                            Education    Employment and Affirmative Action 
##                                   12                                    1 
##                        Energy Issues                 Environmental Issues 
##                                   13                                    7 
##                     Executive Branch           Family and Children Issues 
##                                   26                                    3 
##        Foreign Aid and Policy Issues                           Gun Issues 
##                                   17                                    4 
##                        Health Issues                          Immigration 
##                                   18                                   16 
##                                Labor                         Legal Issues 
##                                    7                                    6 
##                      Military Issues             National Security Issues 
##                                    4                                   10 
##                    Regulatory Issues         Science and Medical Research 
##                                    3                                    1 
##    Senior and Social Security Issues                        Social Issues 
##                                    6                                    2 
##         Technology and Communication                         Trade Issues 
##                                    7                                    9 
##                Transportation Issues                      Veterans Issues 
##                                   11                                    2 
##                  Welfare and Poverty 
##                                    5
```

c)

```
#first, create a matrix with only the actual votes and exclude the bills and missing votes columns
votesonly <- as.matrix(votes[,-(1:2)])
#Get XX', then get the values of its diagonal
votecount1 <- votesonly%*%t(votesonly)
votecount <- diag(votecount1)

#Compare our recorded votes and missing votes with our total number of senators to see if any were not accounted for
matches <- votecount + votes$missing_votes == 100
sum(matches)
```

```
## [1] 441
```

We have 441 matches for 441 bills, so there are no discrepancies. All votes have been accounted for.

d)

```
#First, create the subset where Senator Frist cast a vote (either 1 or -1)
relative1 <- subset(cbind(billtype,votes), votes$William.H..Bill.Frist..TN. != 0)

#Then, recode (votes only) relative to Frist's votes. "Indifferent" means they did
not cast a vote
relative <- relative1[,-(1:3)]
relative[relative == relative$William.H..Bill.Frist..TN.] <- "With"
relative[relative != relative$William.H..Bill.Frist..TN. & relative != 0] <- "Again
st"
relative[relative == 0] <- "Indifferent"

#And now we have a matrix of votes relative to Senator Frist's votes
```

e)

```
#I'm using reshape to get my data into an easier to use format
require(reshape2)
```

```
## Loading required package: reshape2
```

```
## Warning: package 'reshape2' was built under R version 3.2.5
```

```
#recombine the bill type vector from earlier with our new votes matrix
party <- cbind(relative1[,1], relative)
names(party)[1] <- "billtype"

#Then melt the data (using reshape2) so that For/Against/Indifferent is its own col
umn
party2 <- melt(party, id.vars = "billtype", variable_name = "Senator", value.name
= "ForAgainst")

#Create a frequency table of votes by bill type
table1 <- table(party2$billtype, party2$ForAgainst)
#Then use prop.table to get percentages of the totals
percentages <- prop.table(table1,1)
percentages
```

```
##
##                                           Against Indifferent       With
##   Abortion Issues                      0.30588235  0.35000000 0.34411765
##   Agriculture Issues                   0.21833333  0.39166667 0.39000000
##   Appropriations                       0.15608696  0.24641304 0.59750000
##   Arts and Humanities                  0.07000000  0.34000000 0.59000000
##   Budget, Spending and Taxes           0.32159420  0.23565217 0.44275362
##   Business and Consumers               0.41428571  0.06142857 0.52428571
##   Campaign Finance and Election Issues 0.29307692  0.28230769 0.42461538
##   Civil Liberties                      0.23571429  0.34428571 0.42000000
##   Congressional Affairs                0.22444444  0.24888889 0.52666667
##   Crime Issues                         0.30333333  0.32333333 0.37333333
##   Defense                              0.20458333  0.31166667 0.48375000
##   Drug Issues                          0.16000000  0.37500000 0.46500000
##   Education                            0.20833333  0.29250000 0.49916667
##   Employment and Affirmative Action    0.39000000  0.38000000 0.23000000
##   Energy Issues                        0.40384615  0.12000000 0.47615385
##   Environmental Issues                 0.21571429  0.26428571 0.52000000
##   Executive Branch                     0.11560000  0.30080000 0.58360000
##   Family and Children Issues           0.27333333  0.08666667 0.64000000
##   Foreign Aid and Policy Issues        0.22411765  0.34294118 0.43294118
##   Gun Issues                           0.27250000  0.16750000 0.56000000
##   Health Issues                        0.30222222  0.27277778 0.42500000
##   Immigration                          0.26312500  0.30437500 0.43250000
##   Labor                                0.35714286  0.30428571 0.33857143
##   Legal Issues                         0.16833333  0.47500000 0.35666667
##   Military Issues                      0.22750000  0.04500000 0.72750000
##   National Security Issues             0.25200000  0.16000000 0.58800000
##   Regulatory Issues                    0.16333333  0.46333333 0.37333333
##   Science and Medical Research         0.37000000  0.00000000 0.63000000
##   Senior and Social Security Issues    0.32166667  0.14000000 0.53833333
##   Social Issues                        0.22500000  0.27500000 0.50000000
##   Technology and Communication         0.10285714  0.30142857 0.59571429
##   Trade Issues                         0.26000000  0.21333333 0.52666667
##   Transportation Issues                0.13272727  0.35727273 0.51000000
##   Veterans Issues                      0.45500000  0.01500000 0.53000000
##   Welfare and Poverty                  0.12750000  0.48750000 0.38500000
```

Exercise 2

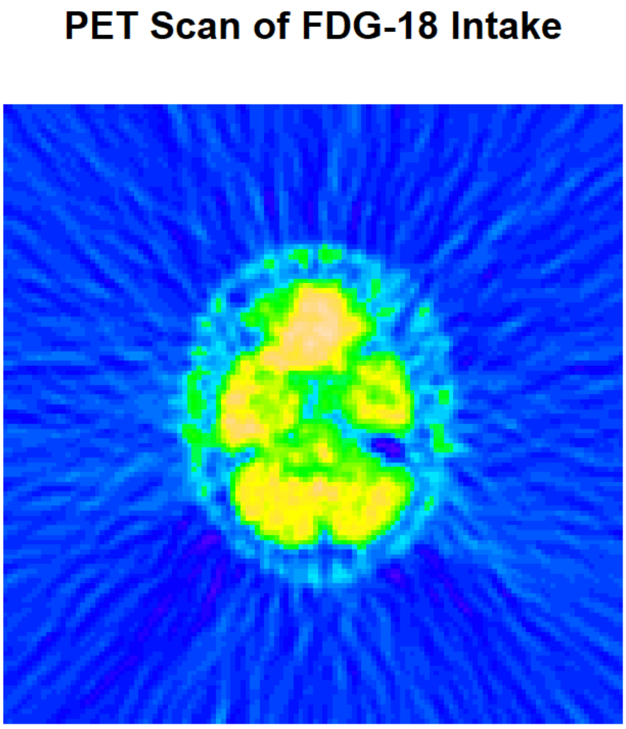
a) **See note about reversing columns/rows**

```
#Read the data into a matrix
pet <- matrix(scan(file = "http://maitra.public.iastate.edu/stat579/datasets/fbp-img.dat"), nrow = 128, ncol = 128, byrow = TRUE)

#Reverse the order of the columns so image() reads it in correctly. In class we reversed columns,
#but based on the information given in the assignment it appears we want to reverse row order instead?
#I can't tell which brain image looks correct
brain <- pet[,128:1]
```

b)

```
require(grDevices); require(graphics);
#An image of our PET brain scan
image(z=brain, col = topo.colors(40), main = "PET Scan of FDG-18 Intake", axes = F
, asp = 1)
```



c)

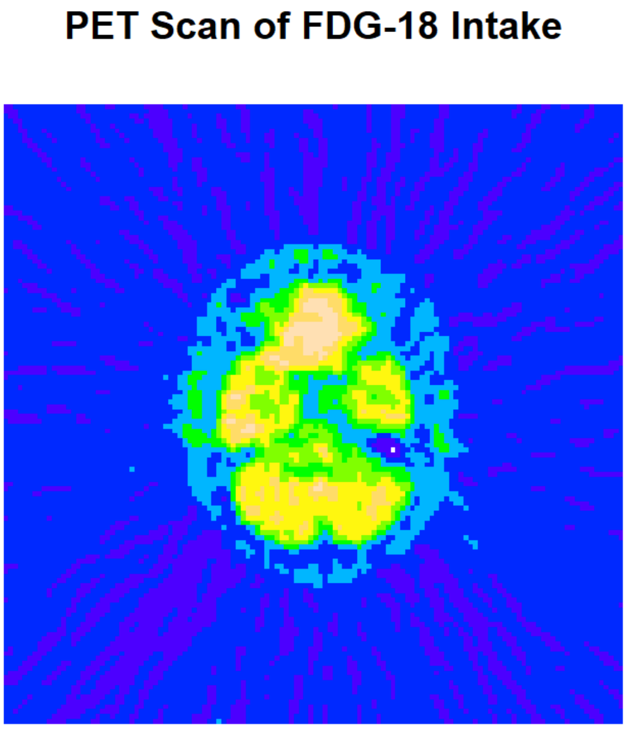
i

```
#First we find the range of our brain data
range(brain)
```

```
## [1] -7418.18 42976.96
```

```
#Then create 8 equally spaced bins
binwidth <-  (max(brain) - min(brain))/8
#Create a sequence of midpoints for each bin
midpts <- seq(from= min(brain)+ binwidth/2, by = binwidth, length = 8)
#bin the data using apply() on each column of the data. cut() seperates the data into 8 bins
binned1 <- apply(brain, 2, cut, seq(from=min(brain), to=max(brain), length = 9), labels = midpts)
#change our bin labels to numeric
binned2 <- apply(binned1, 2, as.numeric)

#an image of our new binned data
image(z=binned2, col = topo.colors(40), main = "PET Scan of FDG-18 Intake", axes = F, asp = 1)
```

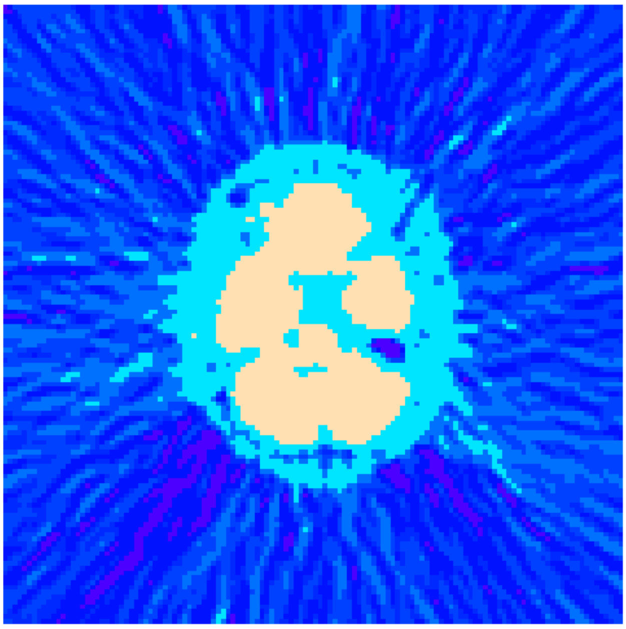


ii

```
#A function to find quantiles based on the methods discussed in lecture
#Easier than the way I did it
bin <- function(b, n){
  #Find quantiles and the mid points of those quantile bins
  b.qt <- quantile(b, probs = seq(0, 1, length = n))
  b.qt.mid <- (b.qt[-1] + b.qt[-length(b.qt)])/2
  #stack data matrix and midpoint n times on top of each other
  b.arr <- array(b, dim = c(dim(b), length(b.qt.mid)))
  mid.arr <- array(rep(b.qt.mid,
                       each = prod(dim(b))), dim = dim(b.arr))
  #Find squared difference between each point and the possible mid points
  sq.diff.arr <- (b.arr - mid.arr)^2
  #Create new matrices that pick the smallest distance between the points and the mid points
  min.idb1 <- apply(X = sq.diff.arr, MAR = c(1,2), FUN = which.min)
  min.idb2 <- apply(X = sq.diff.arr, MAR = c(1,2), FUN = order, decreasing = F)[1,,]
  #Recode the data to be the binned data
  b.quant <- array(b.qt.mid[min.idb1], dim = dim(b))
  return(b.quant)
}
#Then Just run the function with n = 9 (for 8 bins)
brain.quant <- bin(brain, 9)

#A third plot of our brain scan
image(z=brain.quant, col = topo.colors(40), main = "PET Scan of FDG-18 Intake", axes = F, asp = 1)
```

PET Scan of FDG-18 Intake



iii

When we bin the data based on quantiles, it is not a linear transformation of the data, so the coloring is distorted based on the distribution. Since the distribution of the values is not uniform over the range, the bins will be unevenly spaced and sized based on the distribution, which is different from the simple linear binning from the previous part. Note that the first raw image is similar to the raw data, but shows a slightly sharper contrast for the high values.

Exercise 3

a)

```
#Read in the data
auto <- read.table(file= "auto.txt", header=T, sep="")
```

b)

```
#Create our data matrices, with a column of 1s for the intercept term
x <- cbind(rep(1, times = length(auto$horsepower)), auto$horsepower)
y <- auto$mpg

#Estimate our least squares coefficients using standard matrix operations.
b0b1 <- solve(t(x)%*%x)%*%(t(x))%*%y
#The result is a 2x1 matrix with the first entry being our intercept and the second
our slope
b0b1
```

```
##             [,1]
## [1,] 39.9358610
## [2,] -0.1578447
```

c)

```
#Now estimate using the lm() function
model <- lm(auto$mpg~auto$horsepower)
model
```

```
## 
## Call:
## lm(formula = auto$mpg ~ auto$horsepower)
## 
## Coefficients:
##     (Intercept)  auto$horsepower  
##         39.9359          -0.1578
```

As should be expected for this simple model, the regression coefficients are the same for each method.