Homework 6_Kim_Jaiyool

Due Wednesday Oct 9, 9am

2019-10-16

For each assignment, turn in by the due date/time. Late assignments must be arranged prior to submission. In every case, assignments are to be typed neatly using proper English in Markdown.

This week, we spoke about the apply family of functions. We can use these functions to simplify our code (ie our job) if we can create functions. Ultimately, our goal is to find deficiencies and explore relationships in data and quantify these relationships. Efficiently. So, functions and methods to use these functions could be helpful in some scenarios.

Problem 1

Work through the Swirl "R_programming_E" lesson parts 10 and 11, and perhaps 12 if you need some help with things important to Chris' class (there is also a set of swirl lessons on probability. . .).

My ans : I did it.

Problem 2

As in the last homework, create a new R Markdown file (file–>new–>R Markdown–>save as.

The filename should be: HWXX_lastname_firstname, i.e. for me it would be HWXX_Settlage_Bob

You will use this new R Markdown file to solve the following problems:

My ans : Yes. I will use this R Markdown file named HWXX_Kim_Jaiyool.

Problem 3

a.) Create a function that computes the proportion of successes in a vector. Use good programming practices.

```
Proportion.of.success.calculating.function <- function(y){

  success.prob <- sum(y) / length(y) ## sum(y) : sum of the Bernoulli Random Variables.

  return(success.prob)

}
```

b.) Create a matrix to simulate 10 flips of a coin with varying degrees of "fairness" (columns = probability) as follows:

```
    set.seed(12345)
    P4b_data <- matrix(rbinom(10, 1, prob = (30:40)/100), nrow = 10, ncol = 10, byrow = FALSE)
```

c.) Use your function in conjunction with apply to compute the proportion of success in P4b_data by column and then by row. What do you observe? What is going on?

```
## Use apply function.

Proportion.of.success.by.column <- apply(P4b_data, 2, Proportion.of.success.calculating.function)
Proportion.of.success.by.column
```

```
##  [1] 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6
```

```
Proportion.of.success.by.row <- apply(P4b_data, 1, Proportion.of.success.calculating.function)
Proportion.of.success.by.row
```

```
##  [1] 1 1 1 1 0 0 0 0 1 1
```

d.) You are to fix the above matrix by creating a function whose input is a probability and output is a vector whose elements are the outcomes of 10 flips of a coin. Now create a vector of the desired probabilities. Using the appropriate apply family function, create the matrix we really wanted above. Prove this has worked by using the function created in part a) to compute and tabulate the appropriate marginal successes.

```
Printing.output.vector.function <- function(probability){
  outcomes.of.10flips.of.a.coin <- c(rbinom(10, 1, prob = probability))
  return(outcomes.of.10flips.of.a.coin)
}

## To get the tabulate of the appropriate marginal successes, we can use 'sapply' function
Calculating.Proportion.of.success.matrix <- sapply((30:40)/100, Printing.output.vector.function)
Calculating.Proportion.of.success.matrix
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
##  [1,]    0    0    1    1    1    1    1    1    1     0     1
##  [2,]    0    0    0    0    1    0    0    0    1     1     0
##  [3,]    1    1    0    1    0    1    0    0    0     1     1
##  [4,]    0    1    1    1    0    1    0    0    0     1     0
##  [5,]    0    0    0    0    1    1    0    0    1     0     1
##  [6,]    0    0    0    0    0    0    0    0    0     1     1
##  [7,]    0    1    1    0    1    1    1    1    1     1     1
##  [8,]    0    0    1    0    0    0    1    0    1     1     0
##  [9,]    0    0    0    0    0    0    0    1    0     0     0
## [10,]    1    0    0    0    0    1    0    0    0     0     0
```

Problem 4

In Homework 4, we had a dataset we were to compute some summary statistics from. The description of the data was given as "a dataset which has multiple repeated measurements from two devices by thirteen Observers", where the device measurements were in columns "dev1" and "dev2". Reimport that dataset, change the names of "dev1" and "dev2" to x and y and do the following:

1. create a function that accepts a dataframe of values, title, and x/y labels and creates a scatter plot
2. use this function to create:
   (a) a single scatter plot of the entire dataset
   (b) a seperate scatter plot for each observer (using the apply function)

4.1 create a function that accepts a dataframe of values, title, and x/y labels and creates a scatter plot

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
```

```
##     intersect, setdiff, setequal, union
```

```
library(knitr)

Hw4.data<-readRDS("HW4_data.rds")
colnames(Hw4.data)<- c("Observer", "X", "Y")

kable(Hw4.data[c(1,143,285,427,569,711,853,1000,1137,1279,1427,1578,1705),],
      caption = "Homework 4 data set (13 observers exist in this data set).")
```

Table 1: Homework 4 data set (13 observers exist in this data set).

	Observer	X	Y
1	4	55.38460	97.17950
143	1	32.33111	61.41110
285	6	53.36657	90.20803
427	11	50.48151	93.22270
569	13	38.33776	92.47272
711	10	58.21361	91.88189
853	7	57.61323	83.90517
1000	5	50.28853	82.97525
1137	3	55.99303	79.27726
1279	2	51.20389	83.33978
1427	9	30.12333	81.14430
1578	8	20.93200	51.64624
1705	12	65.81554	95.58837

```
kable(summary(Hw4.data[,(2:3)]),caption="Homework 4 dataset summary for dev1 (X) and dev2 (Y).")
```

Table 2: Homework 4 dataset summary for dev1 (X) and dev2 (Y).

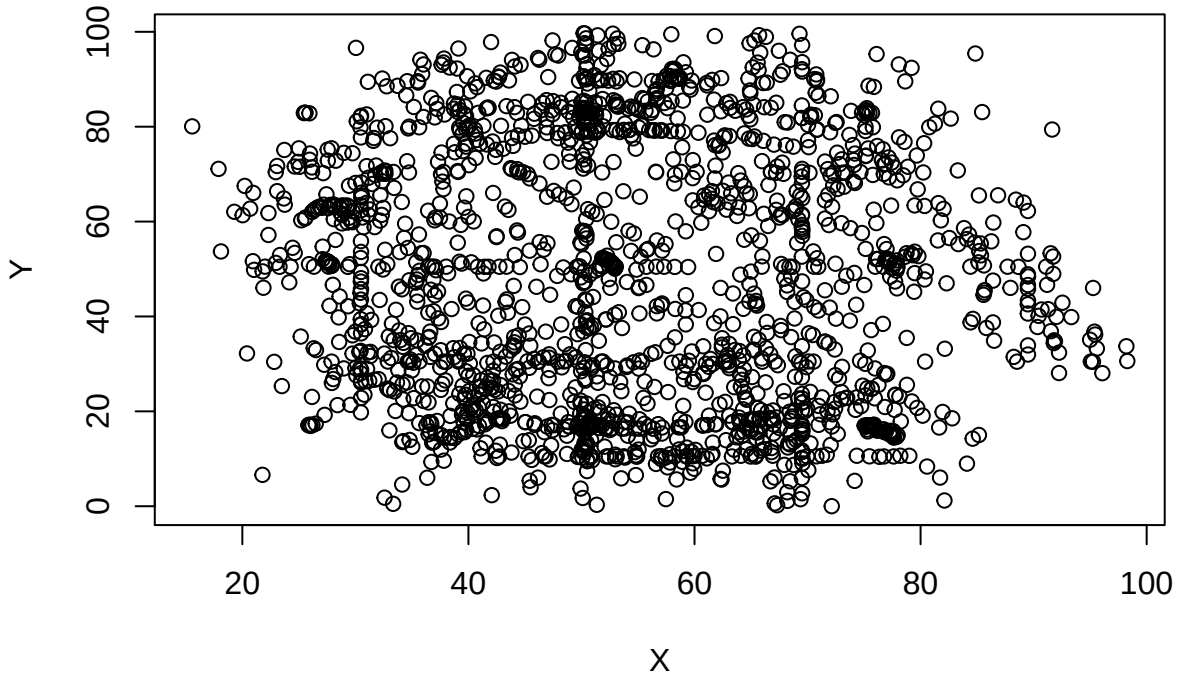
X	Y
Min. :15.56	Min. : 0.01512
1st Qu.:41.07	1st Qu.:22.56107
Median :52.59	Median :47.59445
Mean :54.27	Mean :47.83510
3rd Qu.:67.28	3rd Qu.:71.81078
Max. :98.29	Max. :99.69468

```
## Weave the function that can draw the scatter plot when the given data is inputted.

scatter.plotting.function <- function(givendata){
  scatter.plot <- plot(Y ~ X , data = givendata)
  return(scatter.plot)
  }
```

4.2 Use this function to create:

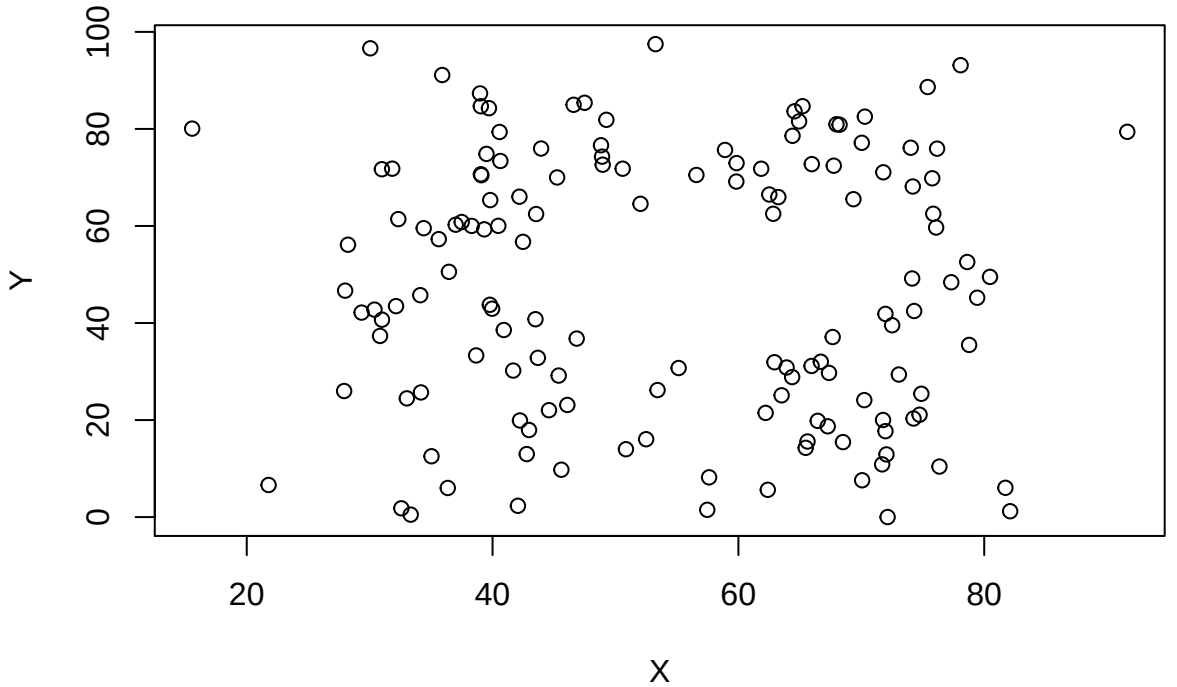
1. a) single scatter plot of the entire dataset
2. b) seperate scatter plot for each observer (using the apply function)

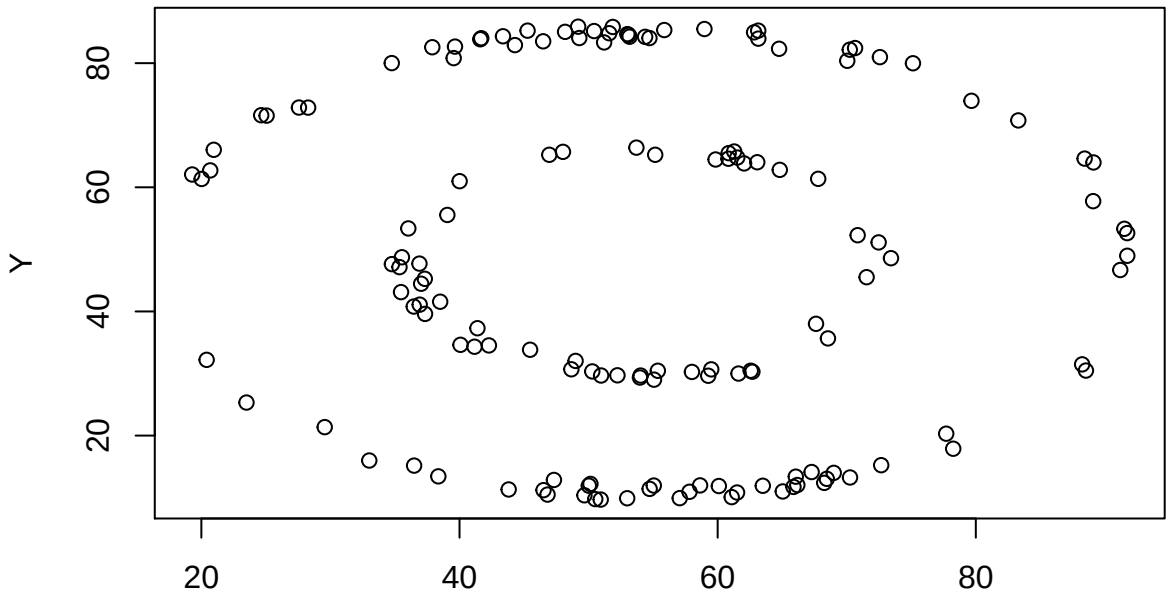
```
## (a) single scatter plot of the entire dataset
scatter.plotting.function(Hw4.data)
```

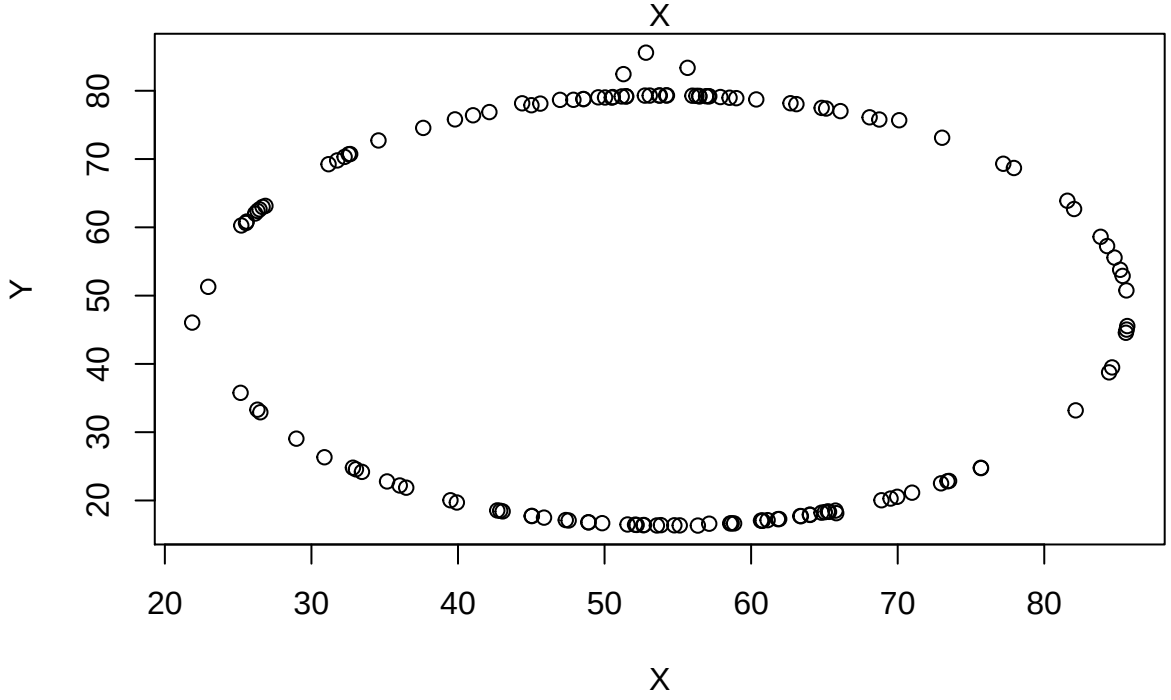


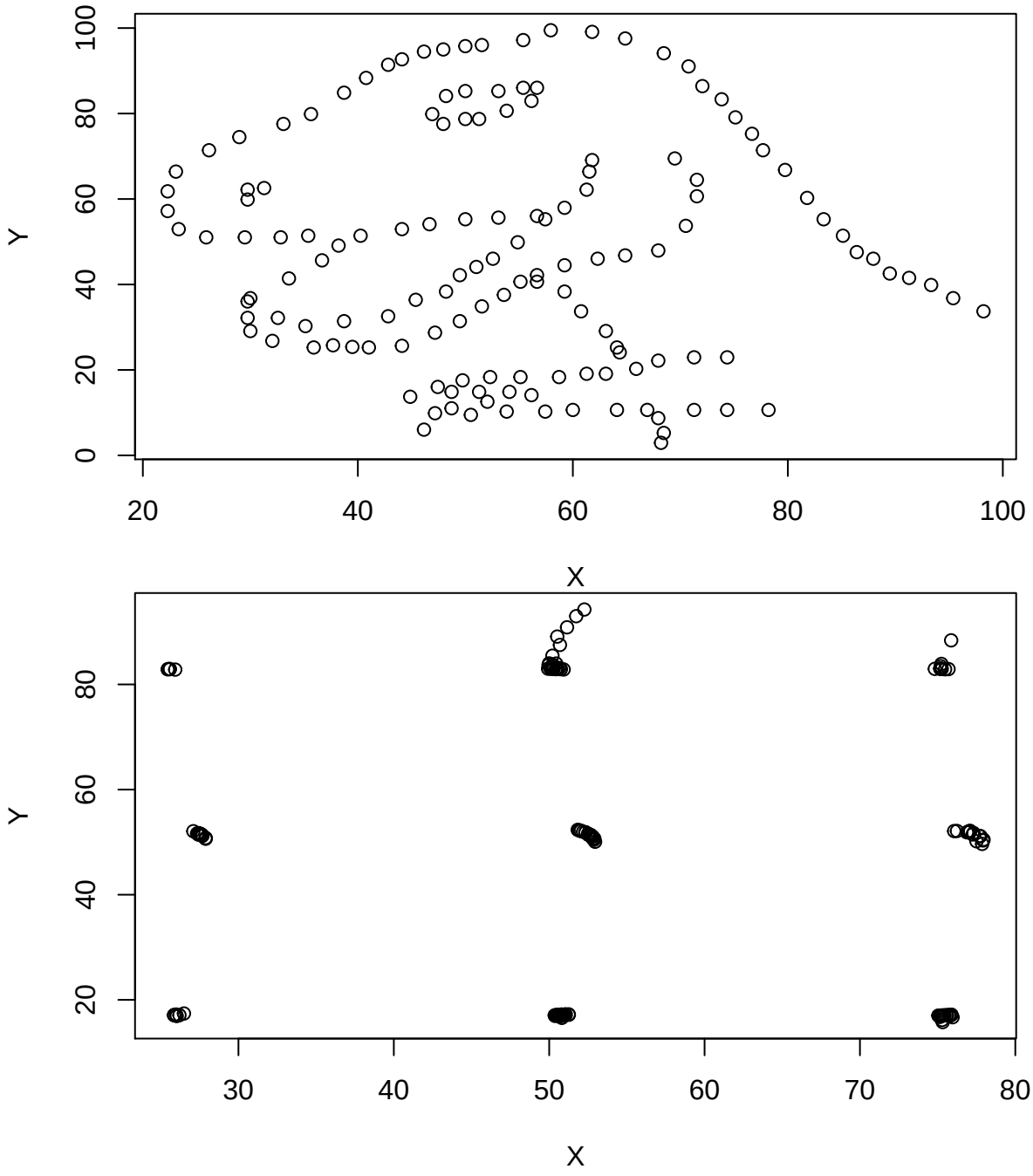

```
## NULL
```

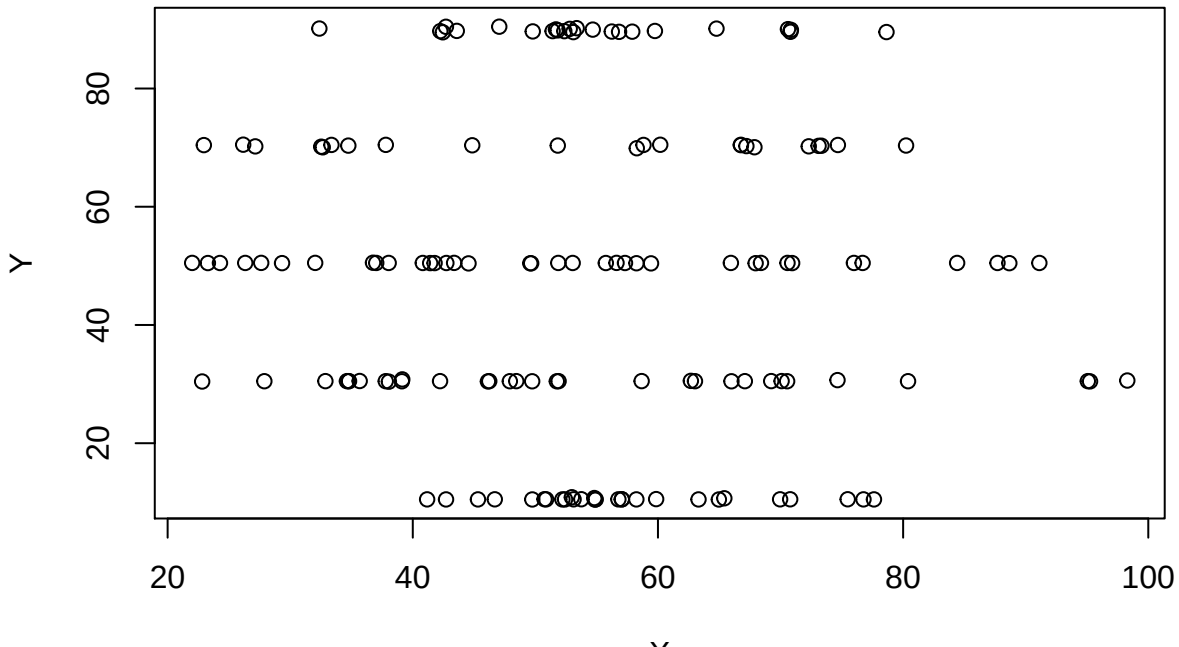
```
## (b) seperate scatter plot for each observer (using the apply function)
Scatter_Plots_for.each.observer <- by(Hw4.data, Hw4.data$Observer, scatter.plotting.function)
```

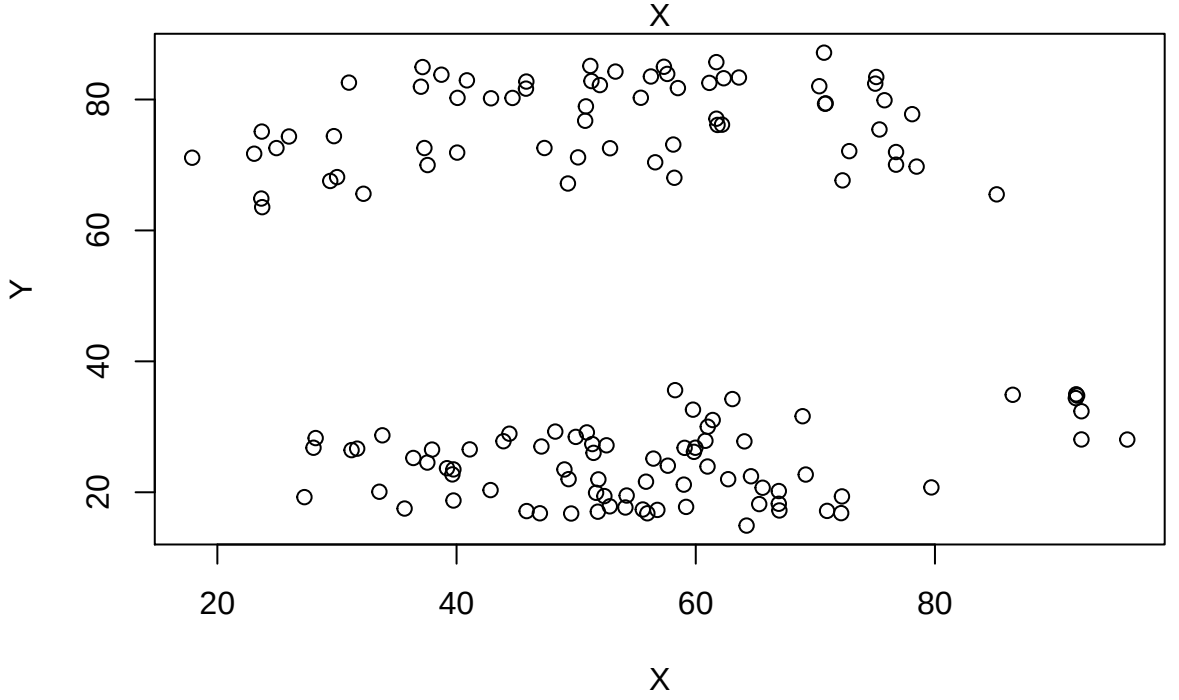


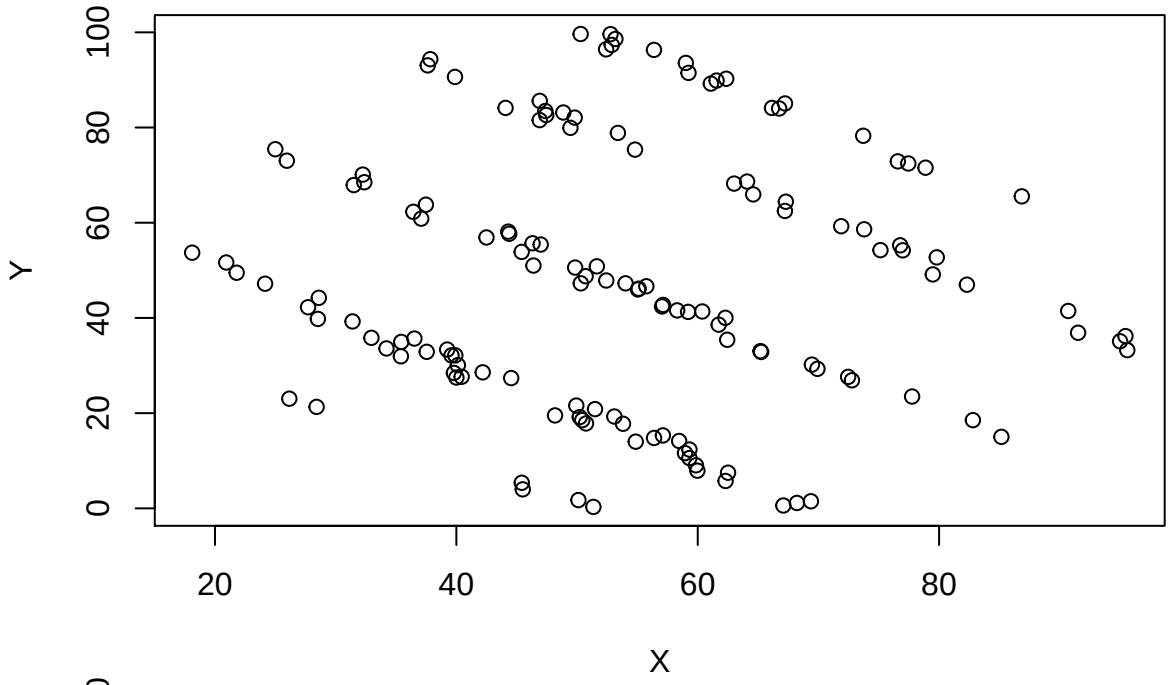

Y
20
40
60
80
20
40
60
80
X

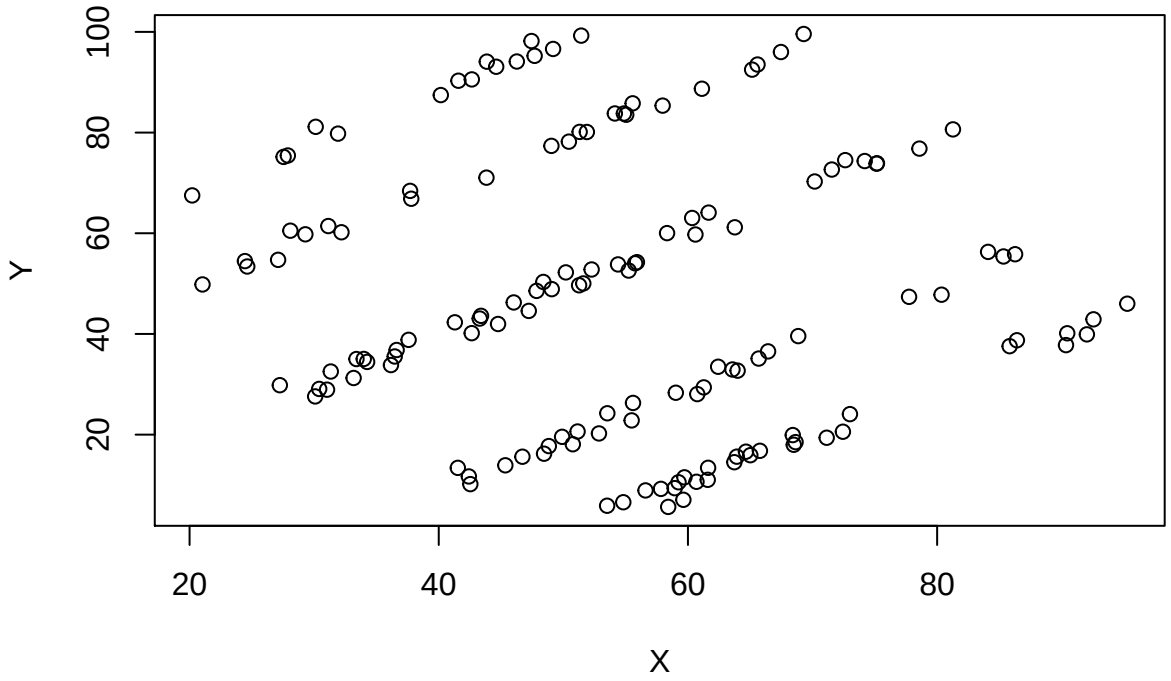

Y
20
30
40
50
60
70
80
20
30
40
50
60
70
80
X

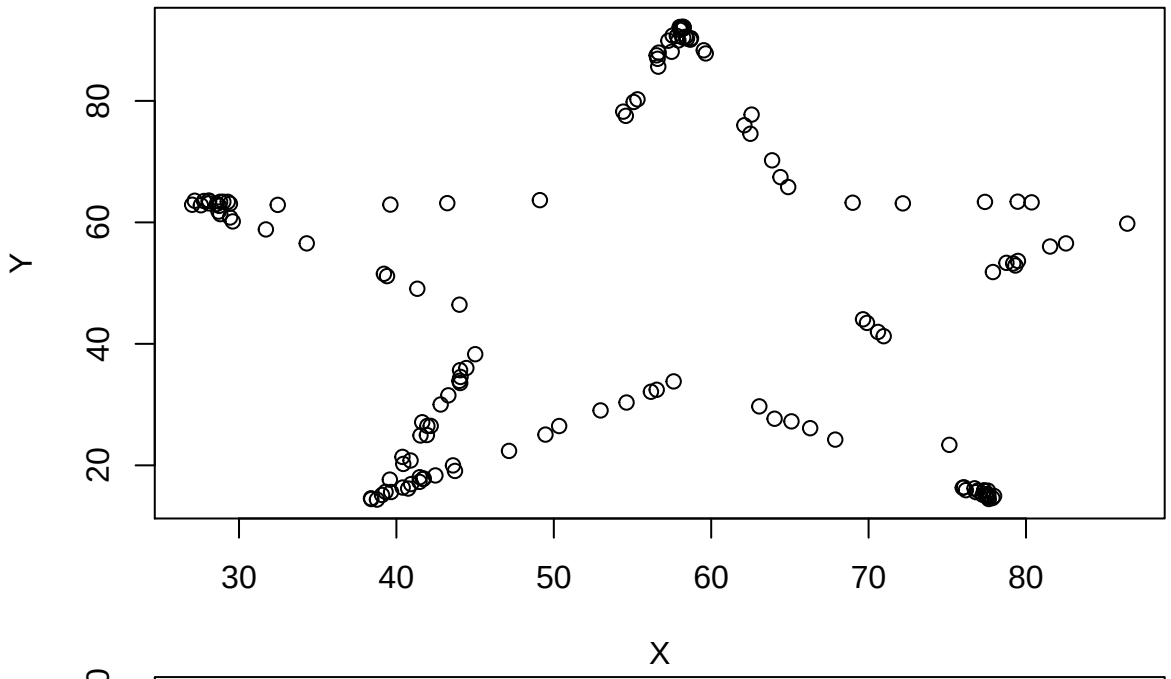

Y
0
20
40
60
80
100
20
40
60
80
100
X
Y
20
40
60
80
30
40
50
60
70
80
X

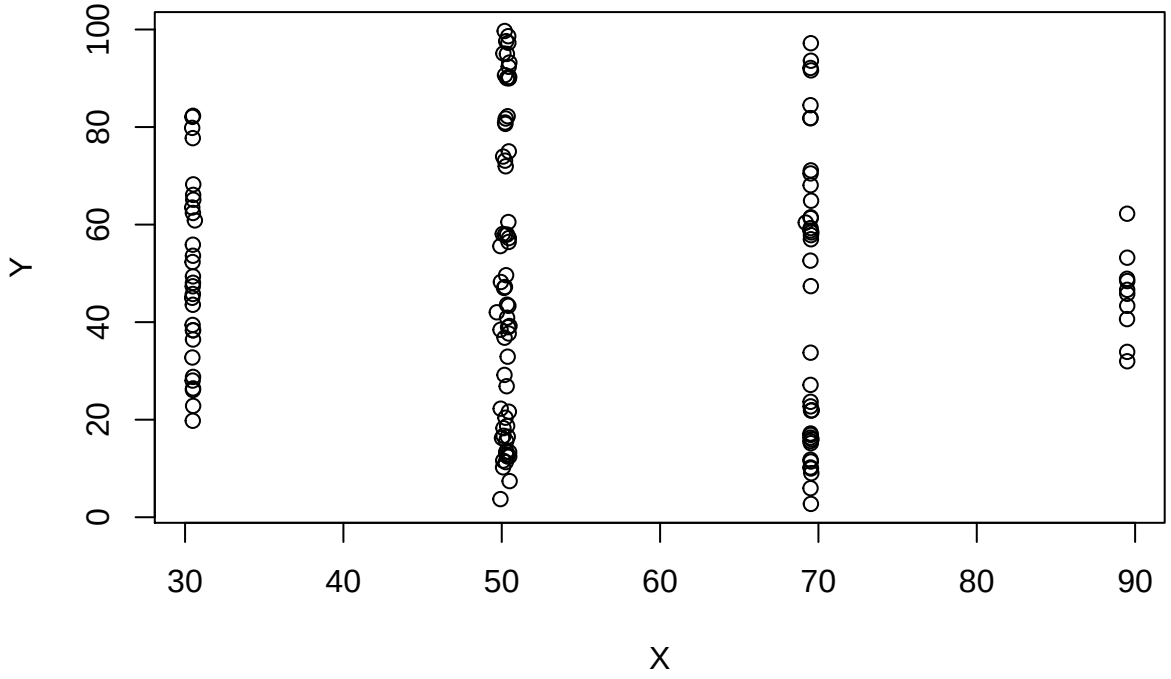

Y
20
40
60
80
20
40
60
80
100
X

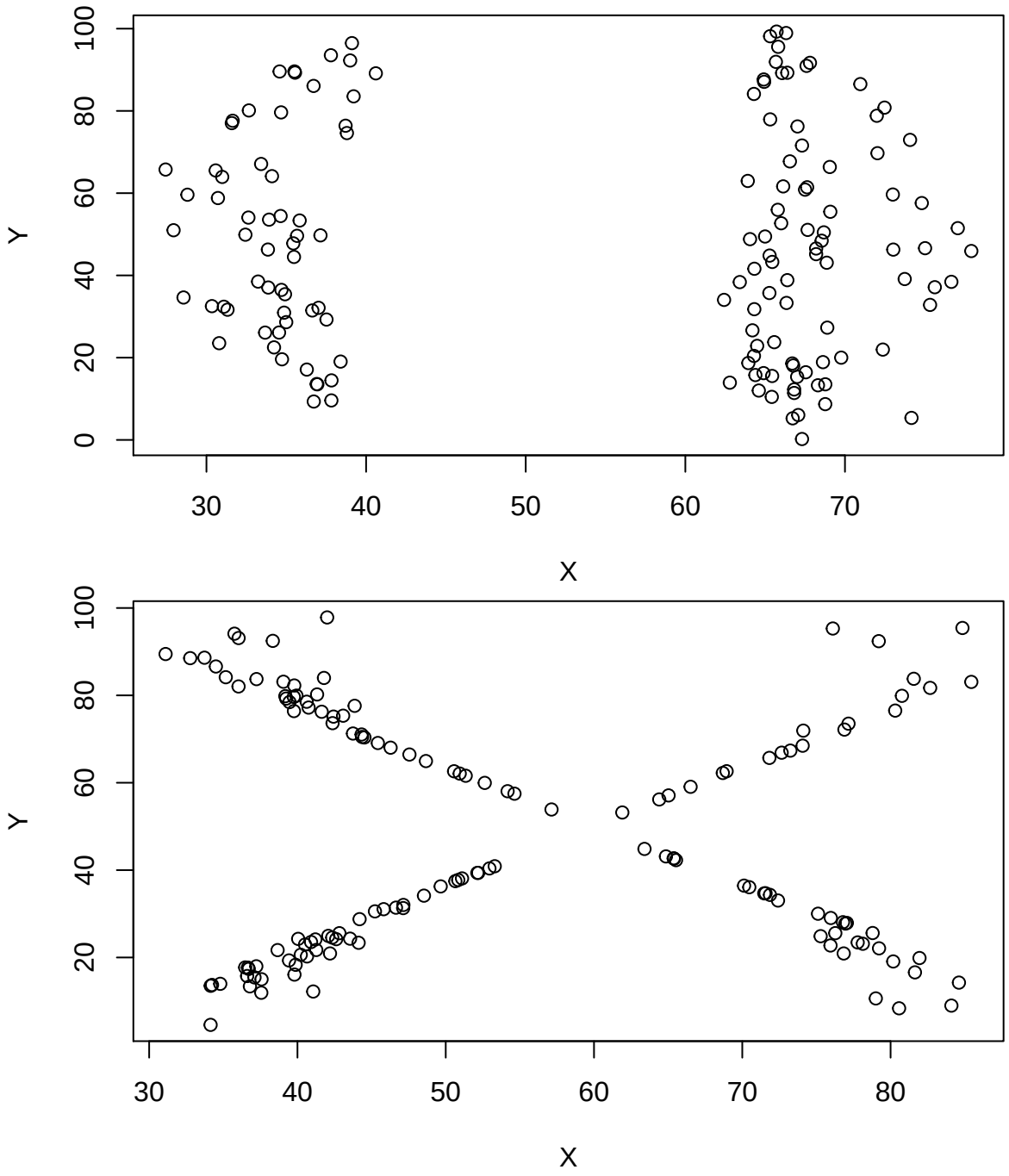

Y
20
40
60
80
20
40
60
80
X


Y
0
20
40
60
80
100
20
40
60
80
X


Y
20
40
60
80
100
20
40
60
80
X


Y
20
40
60
80
30
40
50
60
70
80
X


Y
0
20
40
60
80
100
30
40
50
60
70
80
90
X



Problem 5

Our ultimate goal in this problem is to create an annotated map of the US. I am giving you the code to create said map, you will need to customize it to include the annotations.

Part a.) Get and import a database of US cities and states. Here is some R code to help:

```
# We are grabbing a SQL set from here
# http://www.farinspace.com/wp-content/uploads/us_cities_and_states.zip

# Download the files, looks like it is a .zip
```

```
    library(downloader)

    download("http://www.farinspace.com/wp-content/uploads/us_cities_and_states.zip",dest="us_cities_sta
    unzip("us_cities_states.zip", exdir="./")

    # Read in data, looks like sql dump, blah

    library(data.table)
    states <- fread(input = "./us_cities_and_states/states.sql",skip = 23,sep = "'", sep2 = ",", header

    ### YOU do the CITIES
    ### I suggest the cities_extended.sql may have everything you need
    ### can you figure out how to limit this to the 50?

    nrow(states) # 51
```

Part b.) Create a summary table of the number of cities included by state.

```
library(dplyr)
library(fiftystater)
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
The_cities_extended.sql <- fread(input = "./us_cities_and_states/cities_extended.sql",
                                 skip = 23,sep = "'", sep2 = ",", header = F,select = c(2,4))


The_cities_extended.sql
```

```
##                   V2 V4
##     1:    Holtsville NY
##     2:    Holtsville NY
##     3:      Adjuntas PR
##     4:        Aguada PR
##     5:     Aguadilla PR
##    ---
## 41751:          Reno NV
## 41752:        Duarte CA
## 41753:     Oceanside CA
## 41754: Discovery Bay CA
## 41755:    Sacramento CA
```

```
## Use pipelines and group_by function to create a summary table the problem requires us.

Summary.table.the.number.of.cities.by.state <- The_cities_extended.sql %>%
  group_by(V4) %>%
  tally()
```

Part c.) Create a function that counts the number of occurances of a letter in a string. The input to the function should be "letter" and "state_name". The output should be a scalar with the count for that letter.

```
## Make the function that counts the number of occurances of a letter

 String.count.function<-function(strings, pattern){

  Counted.Number<-NULL

  for(i in 1:length(strings)){

    Counted.Number[i]<-length(attr(gregexpr(pattern,strings[i])[[1]],
                            "match.length")[attr(gregexpr(pattern,strings[i])[[1]], "match.length")>0])
  }

  return(Counted.Number)

 }
```

Part d.)

Create 2 maps to finalize this. Map 1 should be colored by count of cities on our list within the state. Map 2 should highlight only those states that have more than 3 occurances of ANY letter in their name.

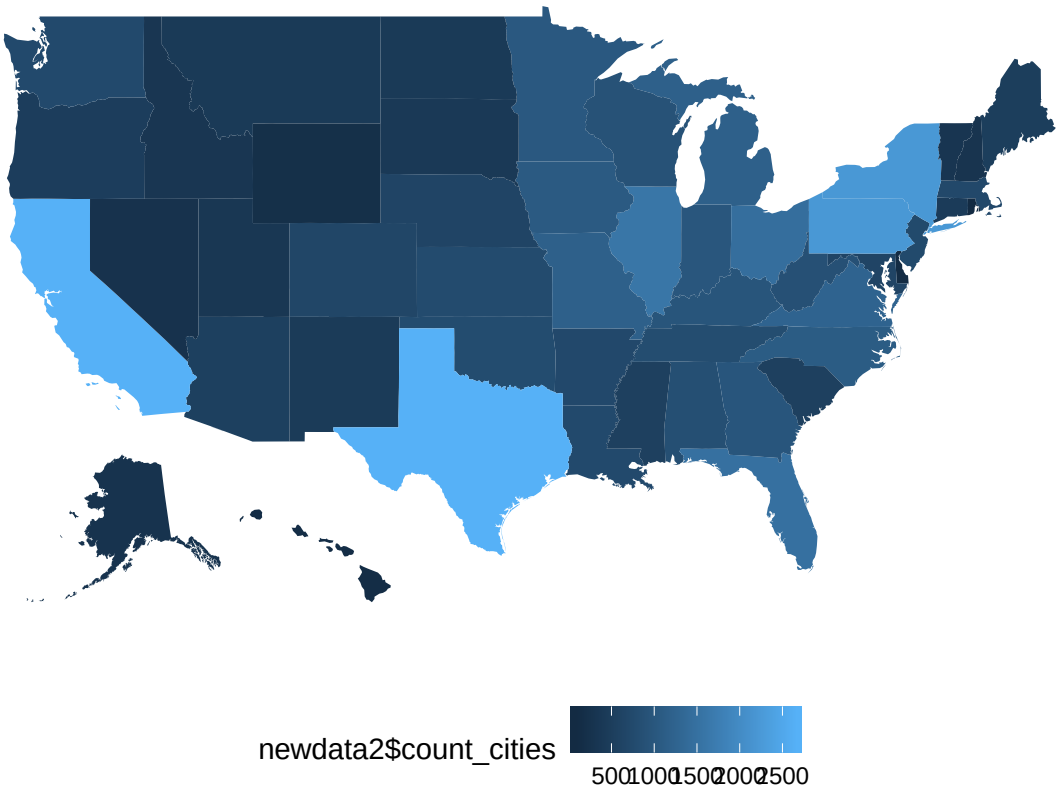
Quick and not so dirty map:

```
states <- fread(input = "./us_cities_and_states/states.sql",skip = 23,sep = "'", sep2 = ",", header = F

 ## First, we need to do alphabet ordering

Alphbet_split_by_alphabet <- strsplit("abcdefghijklmnopqrstuvwxyz","")[[1]]

given.string<-c(Alphbet_split_by_alphabet)

 ##  Matrix counting alphabet in states

letter_count <- data.frame(matrix(NA,nrow=51, ncol=26))


 # https://cran.r-project.org/web/packages/fiftystater/vignettes/fiftystater.html

    library(ggplot2)
    library(fiftystater)
    library(mapproj)
```

```
## Loading required package: maps
```

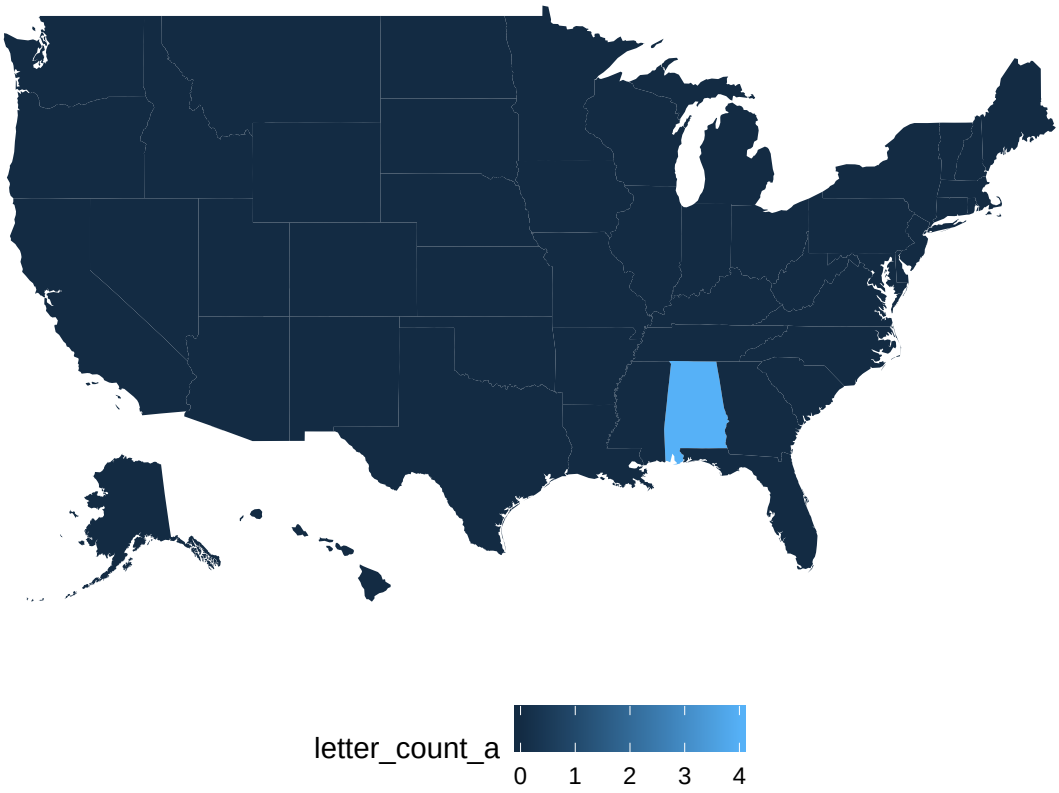
```
    data("fifty_states") # this line is optional due to lazy data loading

    crimes <- data.frame(state = tolower(rownames(USArrests)), USArrests)

    ## Organize data to fit the given the mapping code.

    count_cities <- Summary.table.the.number.of.cities.by.state[-c(8,40),]
    statesam <- states[-8,]
    count_mer_cities <-merge(count_cities, statesam,by="V4")
    newdata <- count_mer_cities
    newdata2 <- newdata[order(newdata$V2),]
    colnames(newdata2) <- c("V4","count_cities" ,"state")
```

```
newdata3 <-cbind(crimes, newdata2$count_cities)


#### Map 1 : should be colored by count of cities on our list within the state.
Map1 <- ggplot(newdata3, aes(map_id = state)) +
  # map points to the fifty_states shape data
  geom_map(aes(fill = newdata2$count_cities), map = fifty_states) +
  expand_limits(x = fifty_states$long, y = fifty_states$lat) +
  coord_map() +
  scale_x_continuous(breaks = NULL) +
  scale_y_continuous(breaks = NULL) +
  labs(x = "", y = "") +
  theme(legend.position = "bottom",
        panel.background = element_blank())

Map1
```




```
#### Map 2 : highlight only those states that have more than 3 occurances of ANY letter in their nam
```

```r
   letter_count_a <- vector()

   letter_count_a <- String.count.function(crimes$state, "a")

   ### Highlight only those states that have more than 3 occurances of "a" letter

   letter_count_a[letter_count_a < 4] <-0
   crimes2 <- cbind(crimes, letter_count_a)

## Map 2 drawing

   Map2 <- ggplot(crimes2, aes(map_id = state)) +

     # map points to the fifty_states shape data

     geom_map(aes(fill = letter_count_a), map = fifty_states) +

     expand_limits(x = fifty_states$long, y = fifty_states$lat) +

     coord_map() +

     scale_x_continuous(breaks = NULL) +

     scale_y_continuous(breaks = NULL) +

     labs(x = "", y = "") +

     theme(legend.position = "bottom",
           panel.background = element_blank())

   Map2
```



Problem 6

Push your homework to submit.

My ans : I did it.