HW4_sliang

Liang Shi

10/11/2020

Problem 2

```
set.seed(1256)
theta <- as.matrix(c(1,2),nrow=2)
X <- cbind(1,rep(1:10,10))
h <- X%*%theta+rnorm(100,0,0.2)

tol <- 0.00001
alpha <- 0.01
theta_ <- matrix(c(0,0), nrow=2)
theta_h <- matrix(c(1,2), nrow=2)
while(abs(theta_h[1] - theta_[1] > tol) & abs(theta_h[2] - theta_[2] > tol)){
  theta_ <- theta_h
  theta_h[1] <- theta_h[1] - alpha * mean(X %*% theta_h - h)
  theta_h[2] <- theta_h[2] - alpha * mean((X %*% theta_h - h) * X[,2])
}
print(theta_h)
```

```
##           [,1]
## [1,] 0.9997817
## [2,] 1.9989402
```

```
lm(h~0+X)
```

```
##
## Call:
## lm(formula = h ~ 0 + X)
##
## Coefficients:
##     X1      X2
## 0.9696  2.0016
```

Problem 3

Problem 4

We know that $X'X\hat{\beta} = X'\underline{y}$, so we can use solve function to get $\hat{\beta}$ beta_hat = solve(t(X)%%X, *t(X)* %% y)

Problem 5

```
set.seed(12456)
G <- matrix(sample(c(0,0.5,1),size=16000,replace=T),ncol=10)
R <- cor(G) # R: 10 * 10 correlation matrix of G
C <- kronecker(R, diag(1600)) # C is a 16000 * 16000 block diagonal matrix
id <- sample(1:16000,size=932,replace=F)
q <- sample(c(0,0.5,1),size=15068,replace=T) # vector of length 15068
A <- C[id, -id] # matrix of dimension 932 * 15068
B <- C[-id, -id] # matrix of dimension 15068 * 15068
p <- runif(932,0,1)
r <- runif(15068,0,1)
C<-NULL #save some memory space
```

Question a

From the environment, we know that the size of A B is 107.1MB and 1.7Gb respectively. However, my mac is not powerful enough to run this code. (I tried to run it, but after 15 min wait, it's still pending).

```
# system.time(y <- p+A%*%solve(B, q-r))
```

Question b

```
# system.time(y <- p+A%*%solve(B, q-r))
# system.time(y <- p + A %*% solve(B) %*% (q - r))
```

Question C

Problem 3

Question a

```
proportion <- function(x, mar){
  return(apply(x, mar, sum) / apply(x, mar, length))
}
```

Question b

```
set.seed(12345)
P4b_data <- matrix(rbinom(10, 1, prob = (31:40)/100), nrow = 10, ncol = 10, byrow = FALSE)
```

Question c

```
# by column
proportion(P4b_data, 2)
```

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

```
# by row
proportion(P4b_data, 1)
```

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

We can find that columns of the data are the same. This means the probability of random variable in the generated matrix are the same.

Question d

```
P4b_data <- matrix(0, nrow = 10, ncol = 10, byrow = F)
prob_ = (31:40)/100
get_coin <- function(data, prob_){
  for (i in 1:10){
    data[i,] <- rbinom(10, 1, prob = prob_[i])
  }
  return(data)
}
P4b_data <- get_coin(P4b_data, prob_)
proportion(P4b_data, 2)
```

```
##  [1] 0.7 0.3 0.5 0.5 0.3 0.1 0.8 0.4 0.1 0.2
```

```
proportion(P4b_data, 1)
```

```
##  [1] 0.2 0.3 0.4 0.3 0.4 0.6 0.3 0.3 0.5 0.6
```

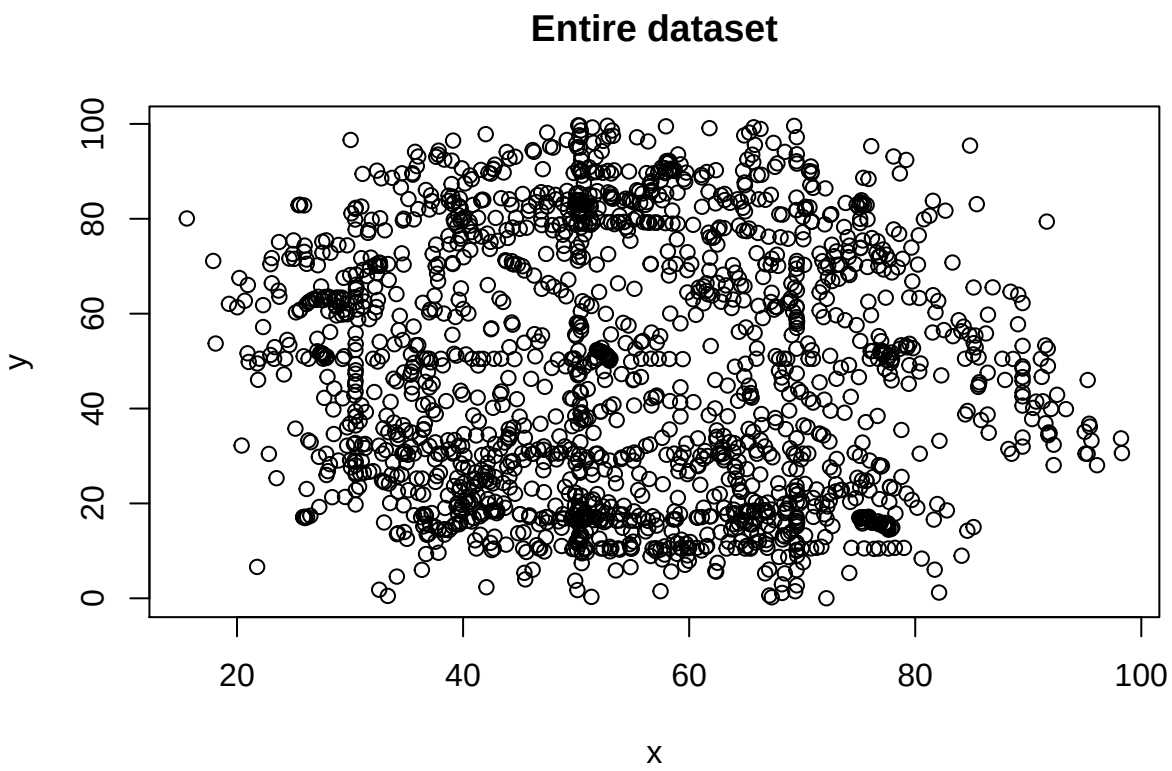
Problem 4

Question a

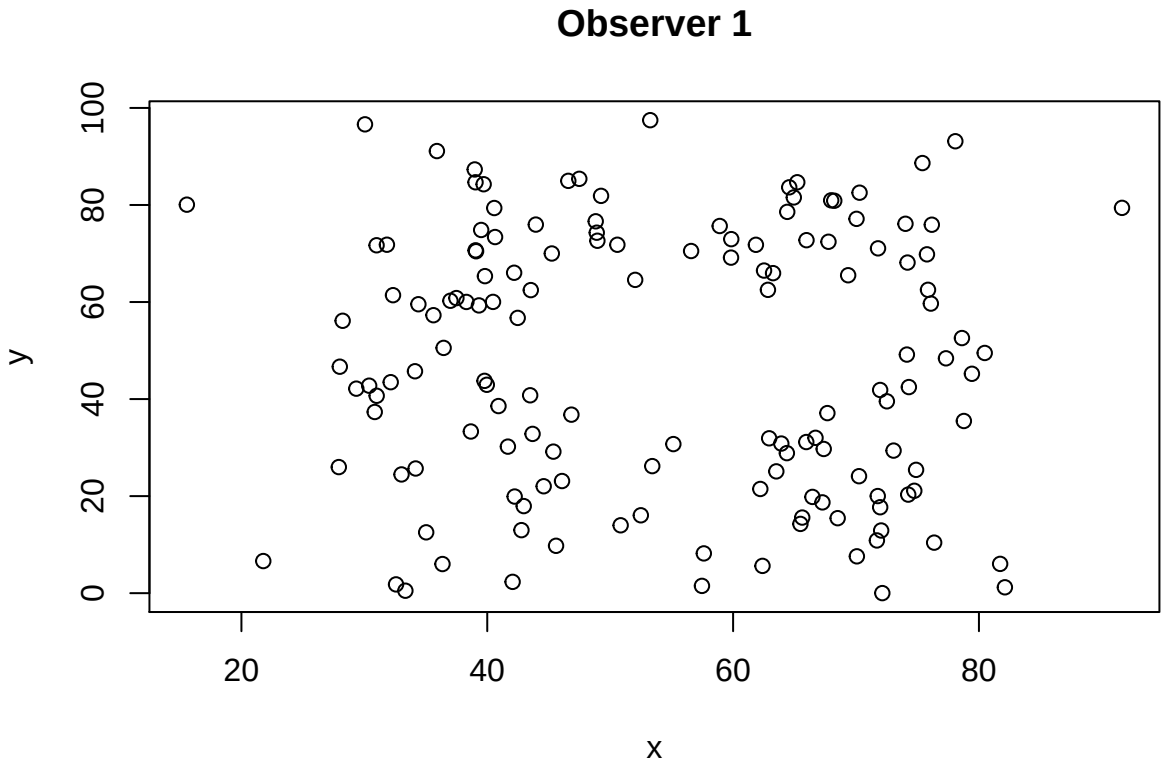
```
df <- readRDS('HW3_data.rds')
colnames(df) <- c('Observer', 'x', 'y')

fun_plot <- function(i){
  if (i == 0)
    plot(df$x, df$y, main = "Entire dataset", xlab = "x", ylab = "y")
  else
    plot(df[which(df$Observer == i),]$x, df[which(df$Observer == i),]$y, main = paste("Observer", i), xl
}
```

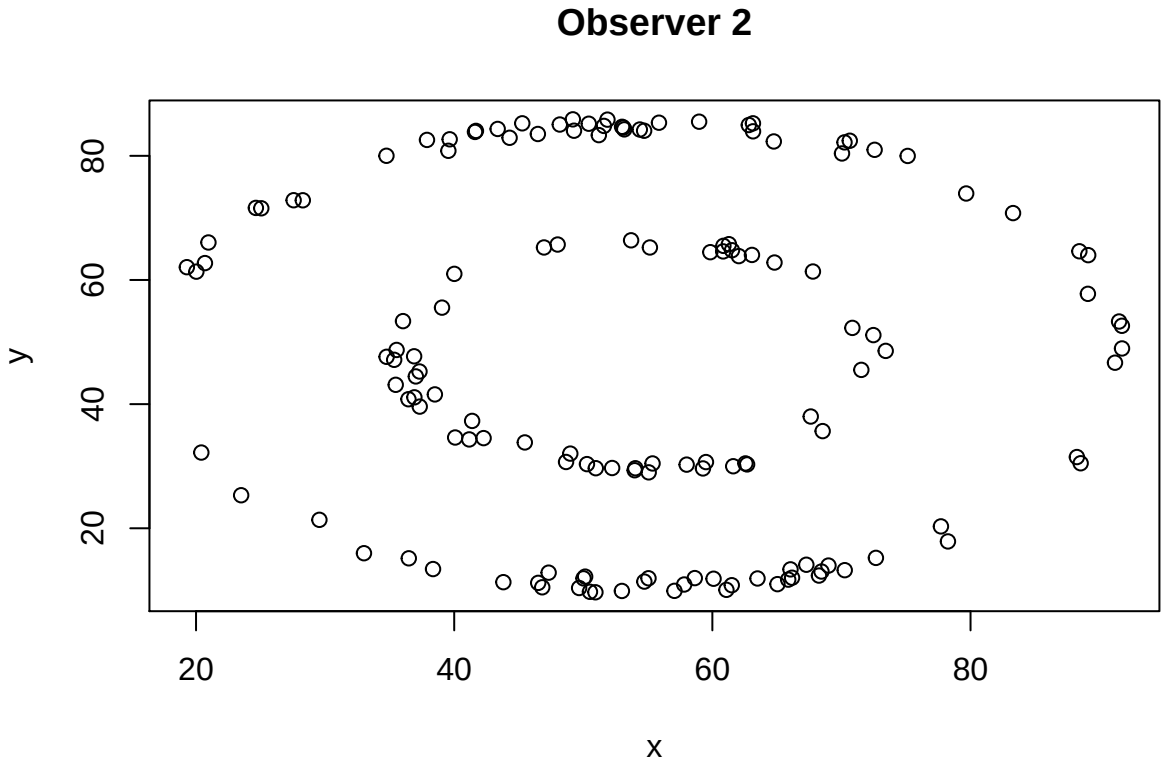
Question b

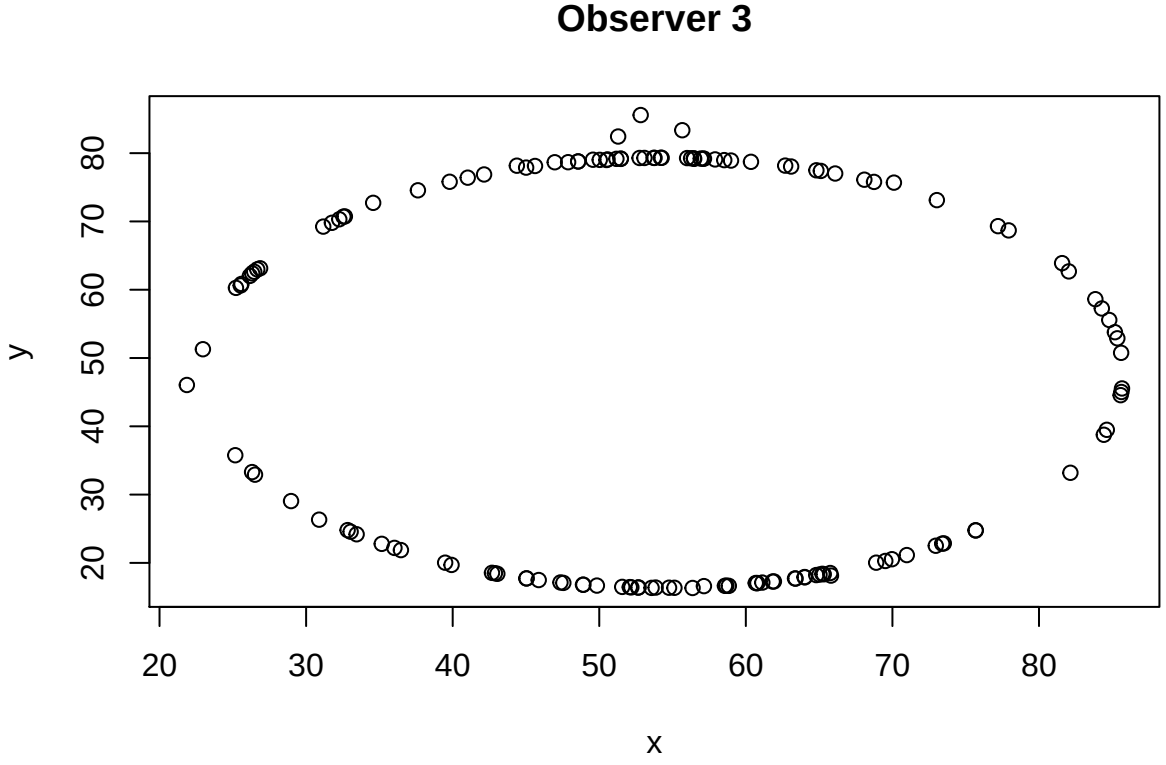
```
fun_plot(0)
```

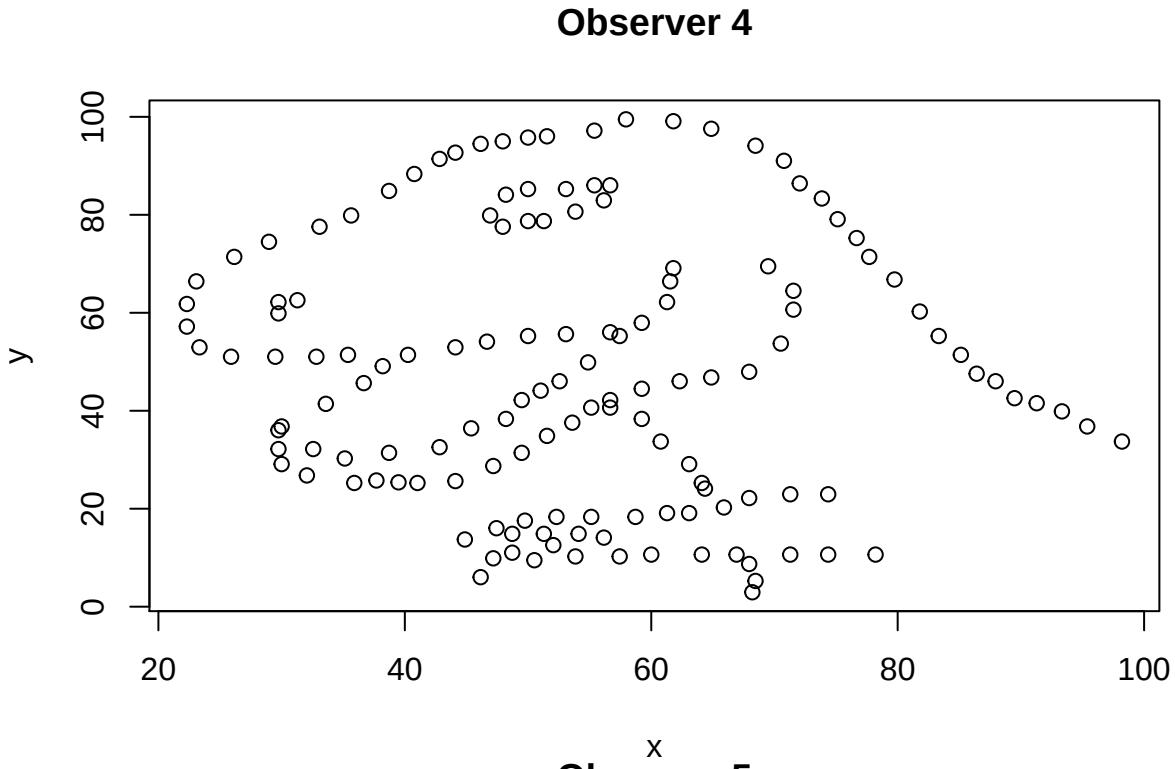


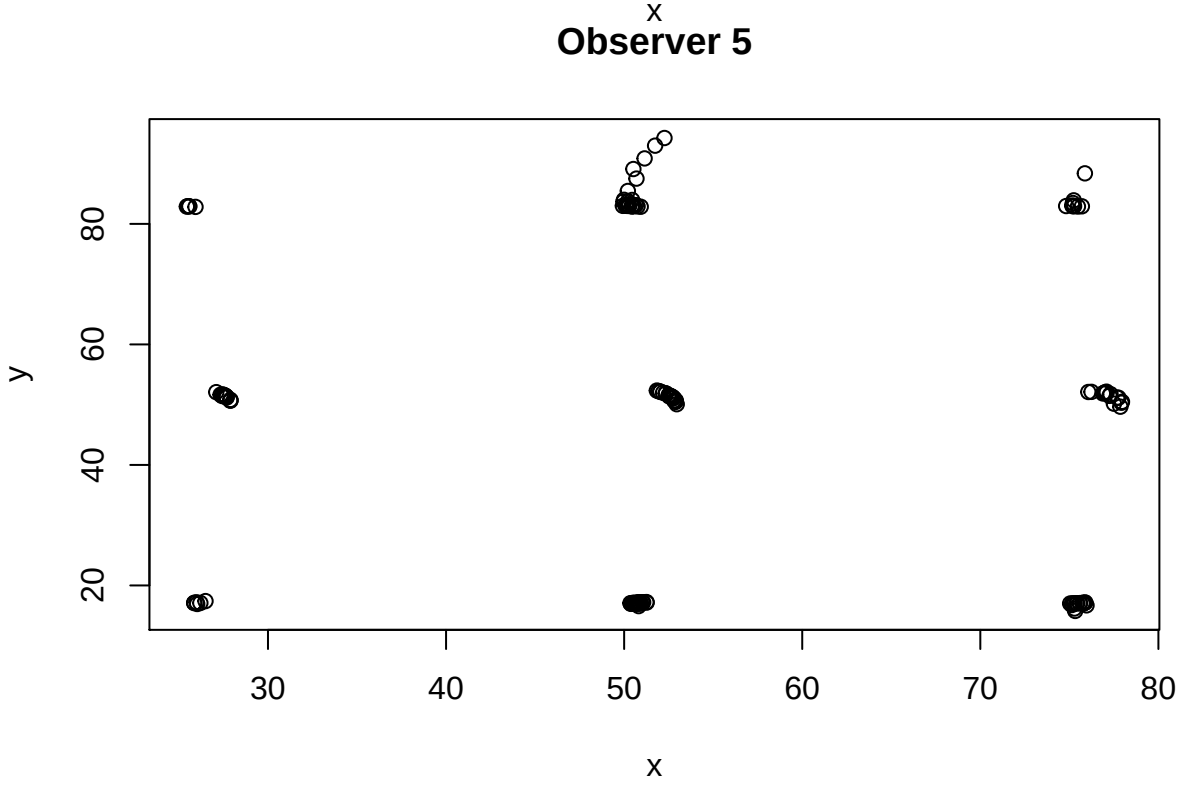

```
lapply(1:13, fun_plot)
```

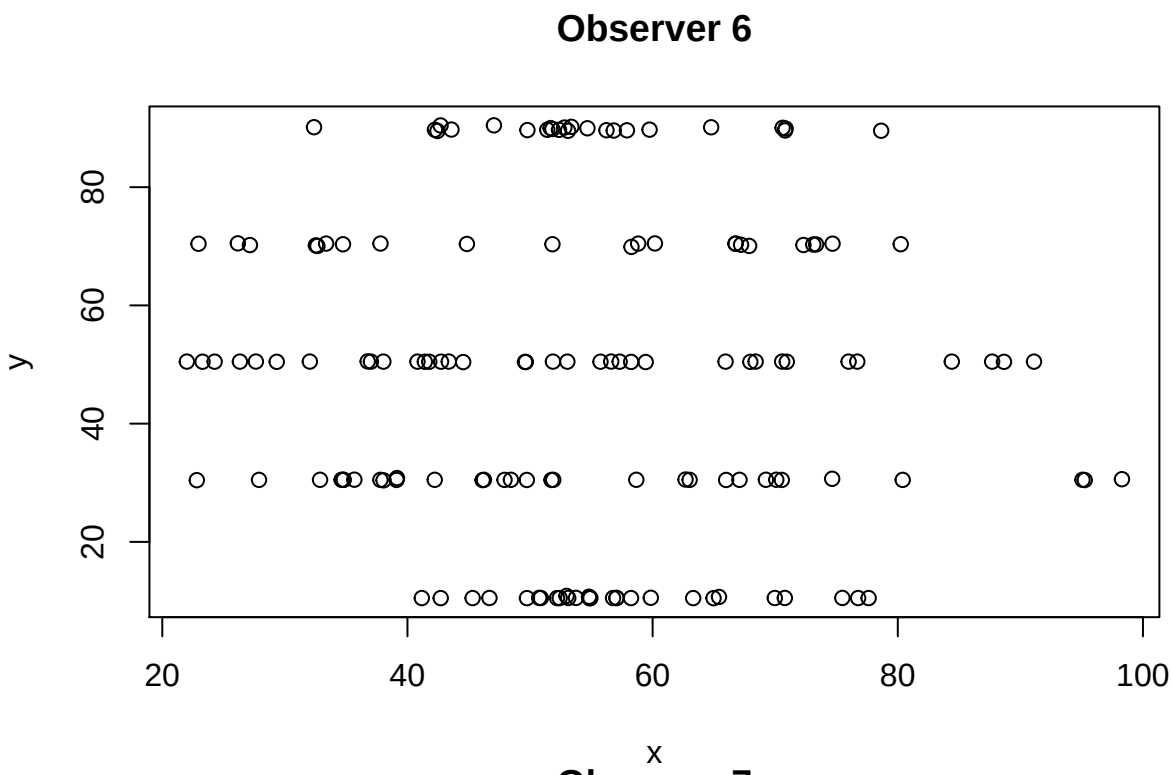


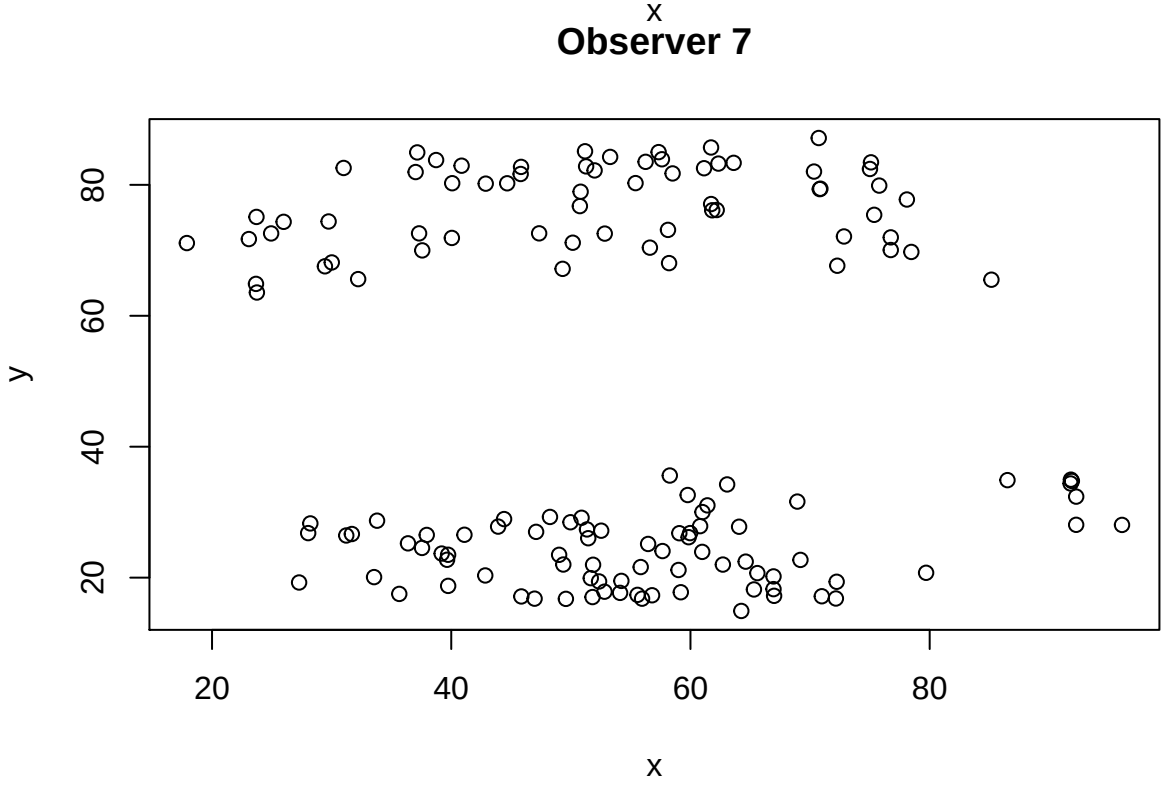

Observer 2
y
x
20
40
60
80

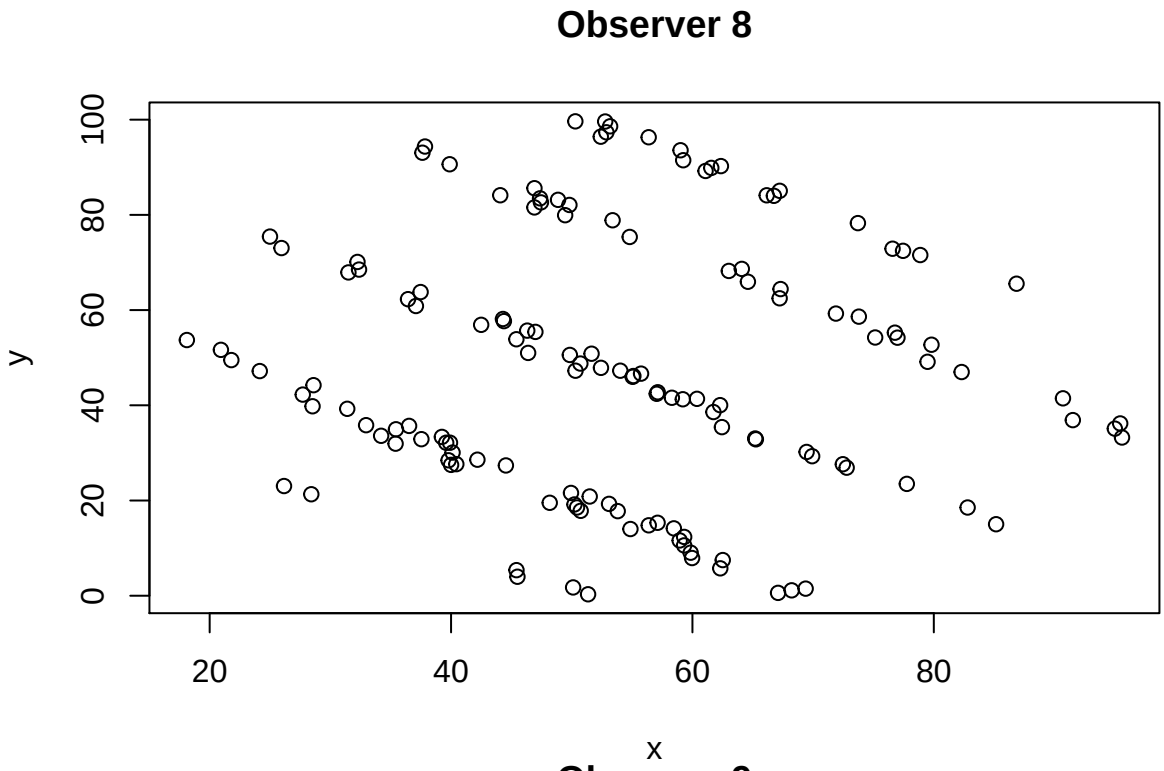

Observer 3
y
x
20
30
40
50
60
70
80

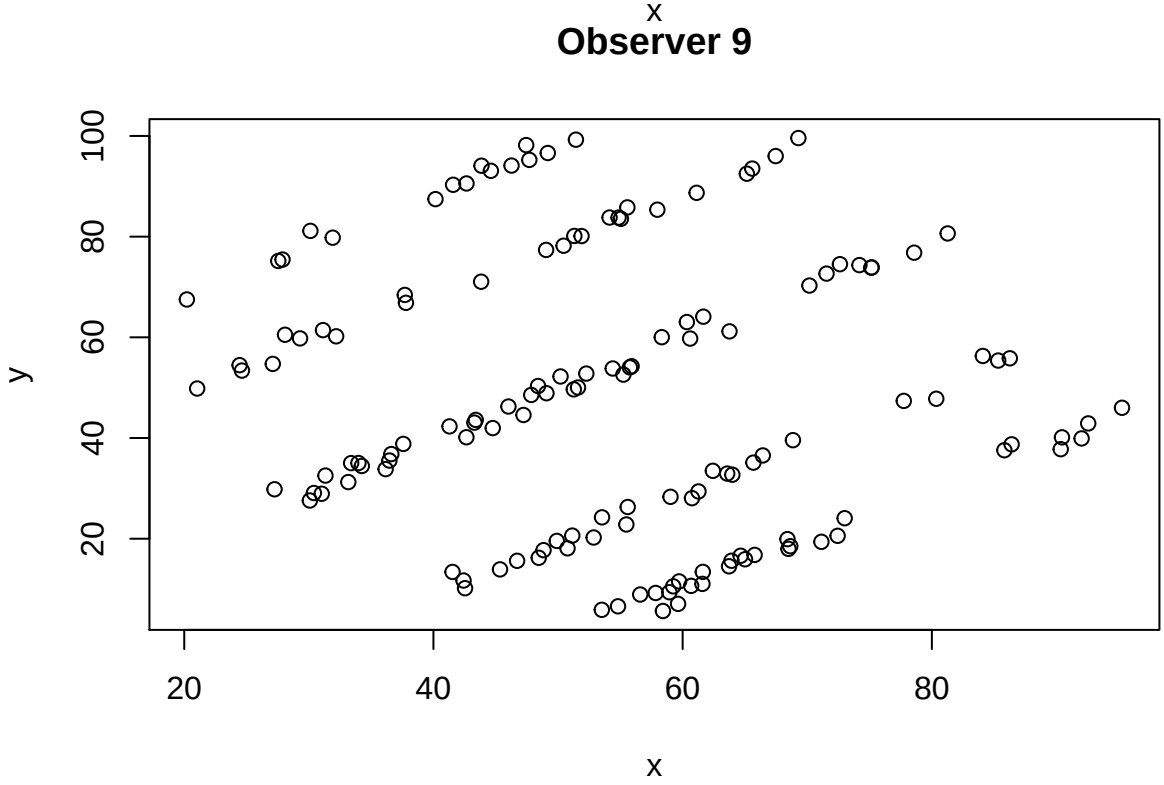

Observer 4
y
x
0
20
40
60
80
100
20
40
60
80
100

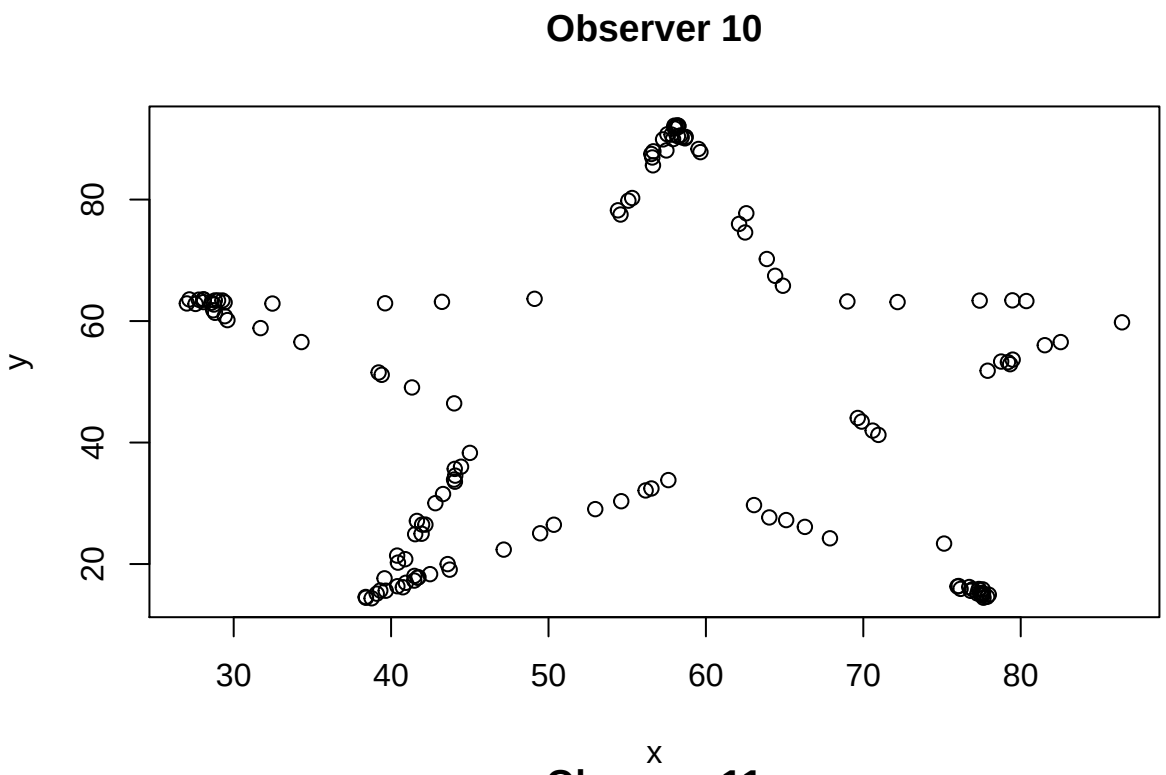

Observer 5
y
x
20
40
60
80
30
40
50
60
70
80

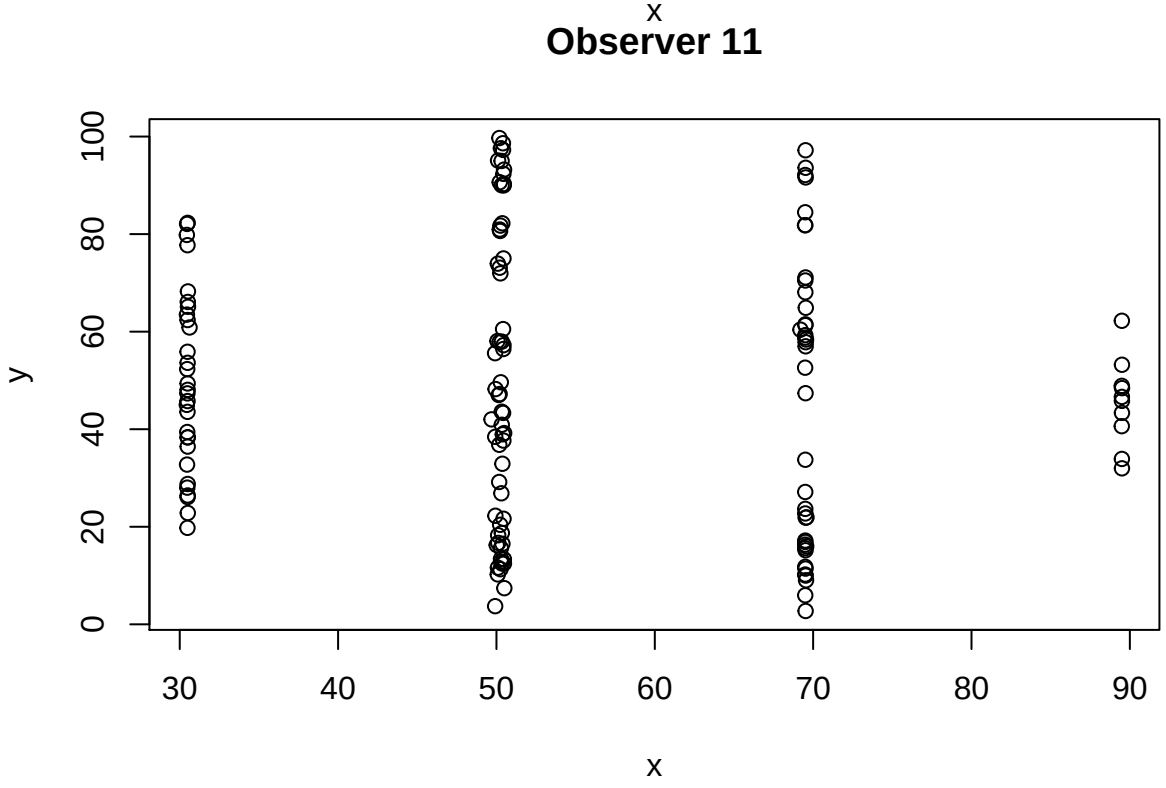

Observer 6
y
x
20
40
60
80
100

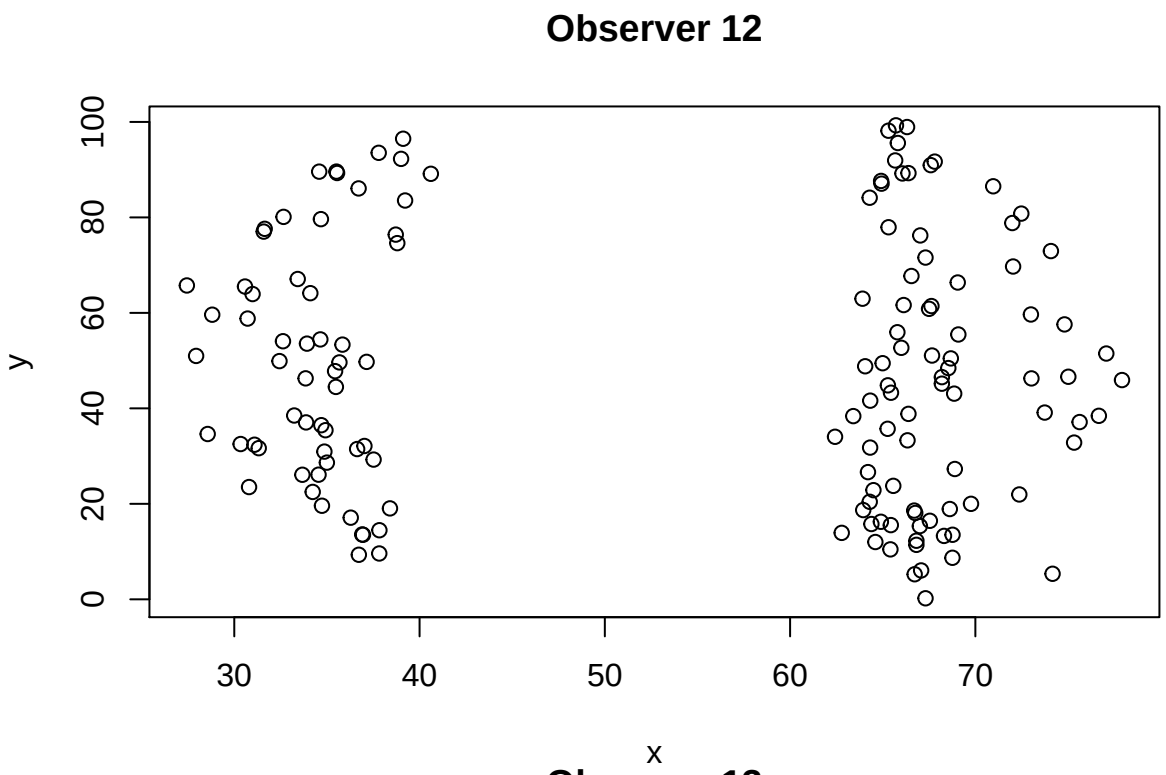

Observer 7
y
x
20
40
60
80

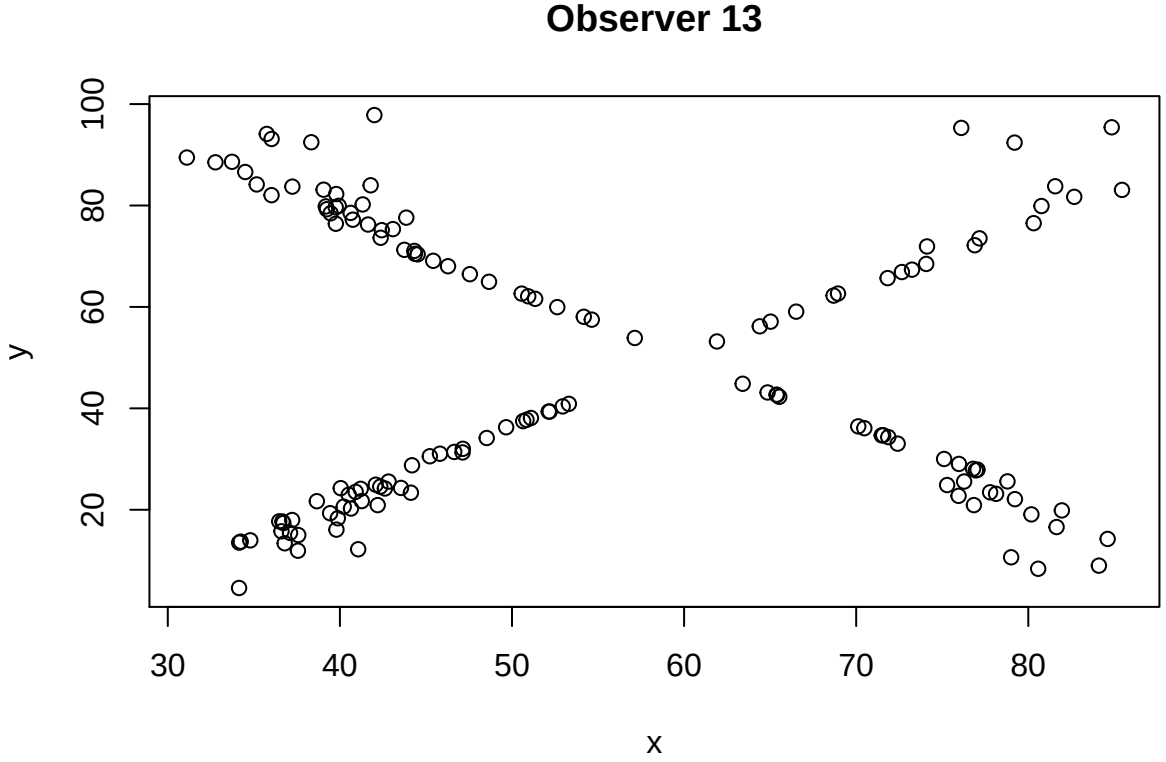

Observer 8
y
x
0
20
40
60
80
100
20
40
60
80


Observer 9
y
x
20
40
60
80
100
20
40
60
80


Observer 10
y
x
20
40
60
80
30
40
50
60
70
80


Observer 11
y
x
0
20
40
60
80
100
30
40
50
60
70
80
90


Observer 12
y
x
0
20
40
60
80
100
30
40
50
60
70


Observer 13
y
x
20
40
60
80
100
30
40
50
60
70
80

```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
```

```
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
##
## [[6]]
## NULL
##
## [[7]]
## NULL
##
## [[8]]
## NULL
##
## [[9]]
## NULL
##
## [[10]]
## NULL
##
## [[11]]
## NULL
##
## [[12]]
## NULL
##
## [[13]]
## NULL
```

Problem 5

Part a

```
library(downloader)
download("http://www.farinspace.com/wp-content/uploads/us_cities_and_states.zip",dest="us_cities_states
unzip("us_cities_states.zip", exdir="./")
library(data.table)
states <- fread(input = "./us_cities_and_states/states.sql",skip = 23,sep = "'", sep2 = ",", header = F
```

Part b

```
library(knitr)
library(dplyr)
```

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

```
## The following objects are masked from 'package:data.table':
##
```

```
##     between, first, last

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

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
cities <- fread(input = "./us_cities_and_states/cities_extended.sql",skip = 23,sep = "'", sep2 = ",", he
cities <- unique(cities)
num_city <- data.frame(table(cities$V4))
colnames(num_city) <- c("State", "# of cities")
kable(num_city)
```

State	# of cities
AK	229
AL	579
AR	605
AZ	264
CA	1239
CO	400
CT	269
DC	3
DE	57
FL	524
GA	629
HI	92
IA	937
ID	266
IL	1287
IN	738
KS	634
KY	803
LA	479
MA	511
MD	430
ME	461
MI	885
MN	810
MO	942
MS	440
MT	360
NC	762
ND	373
NE	528
NH	255
NJ	579
NM	346
NV	99
NY	1612
OH	1069
OK	585
OR	379

State	# of cities
PA	1802
PR	99
RI	70
SC	377
SD	364
TN	548
TX	1466
UT	250
VA	839
VT	288
WA	493
WI	753
WV	753
WY	176

Part c

```
letter_count <- data.frame(matrix(NA,nrow=51, ncol=26))
getCount <- function(state, l){
  state <- tolower(state)
  dif <- gsub(l, "", state)
  count <- nchar(state) - nchar(dif)
  return(count)
}
for (i in 1:51){
  for(j in 1:26){
    letter_count[i,j] <- getCount(states$V2[i], letters[j])
  }
}
colnames(letter_count) <- letters
rownames(letter_count) <- states$V2
kable(letter_count)
```

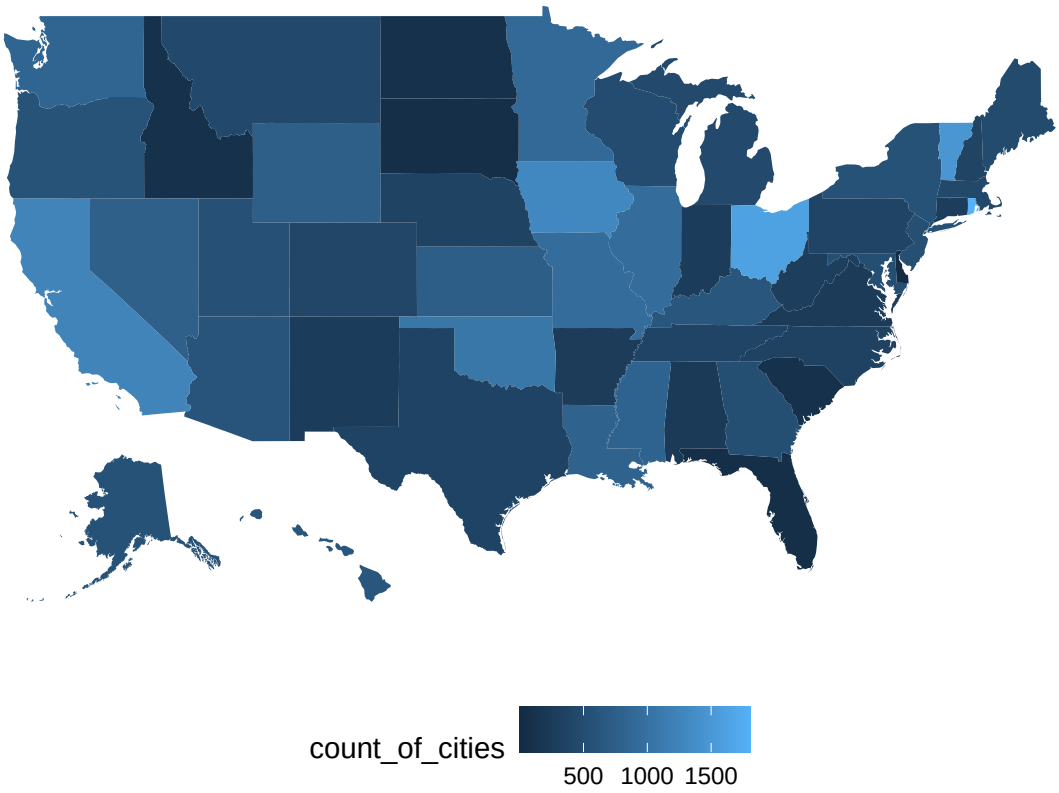
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Alaska	3	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
Alabama	4	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Arkansas	3	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	2	0	0	0	0	0	0	0
Arizona	2	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	1
California	2	0	1	0	0	1	0	0	2	0	0	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0
Colorado	1	0	1	1	0	0	0	0	0	0	0	1	0	0	3	0	0	1	0	0	0	0	0	0	0	0
Connecticut	0	0	3	0	1	0	0	0	1	0	0	0	0	2	1	0	0	0	0	2	1	0	0	0	0	0
District of Columbia	1	1	2	1	0	1	0	0	3	0	0	1	1	0	2	0	0	1	1	2	1	0	0	0	0	0
Delaware	2	0	0	1	2	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0
Florida	1	0	0	1	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0
Georgia	1	0	0	0	1	0	2	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
Hawaii	2	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Iowa	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0
Idaho	1	0	0	1	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Illinois	0	0	0	0	0	0	0	0	3	0	0	2	0	1	1	0	0	0	1	0	0	0	0	0	0	0
Indiana	2	0	0	1	0	0	0	0	2	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
Kansas	2	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Kentucky	0	0	1	0	1	0	0	0	0	0	2	0	0	1	0	0	0	0	0	1	1	0	0	0	1	0
Louisiana	2	0	0	0	0	0	0	0	2	0	0	1	0	1	1	0	0	0	1	0	1	0	0	0	0	0
Massachusetts	2	0	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	4	2	1	0	0	0	0	0
Maryland	2	0	0	1	0	0	0	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	1	0
Maine	1	0	0	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
Michigan	1	0	1	0	0	0	1	1	2	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
Minnesota	1	0	0	0	1	0	0	0	1	0	0	0	1	2	1	0	0	0	1	1	0	0	0	0	0	0
Missouri	0	0	0	0	0	0	0	0	2	0	0	0	1	0	1	0	0	1	2	0	1	0	0	0	0	0
Mississippi	0	0	0	0	0	0	0	0	4	0	0	0	1	0	0	2	0	0	4	0	0	0	0	0	0	0
Montana	2	0	0	0	0	0	0	0	0	0	0	0	1	2	1	0	0	0	0	1	0	0	0	0	0	0
North Carolina	2	0	1	0	0	0	0	1	1	0	0	1	0	2	2	0	0	2	0	1	0	0	0	0	0	0
North Dakota	2	0	0	1	0	0	0	1	0	0	1	0	0	1	2	0	0	1	0	2	0	0	0	0	0	0
Nebraska	2	1	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0
New Hampshire	1	0	0	0	2	0	0	2	1	0	0	0	1	1	0	1	0	1	1	0	0	0	1	0	0	0
New Jersey	0	0	0	0	3	0	0	0	0	1	0	0	0	1	0	0	0	1	1	0	0	0	1	0	1	0
New Mexico	0	0	1	0	2	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	1	1	0	0
Nevada	2	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
New York	0	0	0	0	1	0	0	0	0	0	1	0	0	1	1	0	0	1	0	0	0	0	1	0	1	0
Ohio	0	0	0	0	0	0	0	1	1	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
Oklahoma	2	0	0	0	0	0	0	1	0	0	1	1	1	0	2	0	0	0	0	0	0	0	0	0	0	0
Oregon	0	0	0	0	1	0	1	0	0	0	0	0	0	1	2	0	0	1	0	0	0	0	0	0	0	0
Pennsylvania	2	0	0	0	1	0	0	0	1	0	0	1	0	3	0	1	0	0	1	0	0	1	0	0	1	0
Rhode Island	1	0	0	2	1	0	0	1	1	0	0	1	0	1	1	0	0	1	1	0	0	0	0	0	0	0
South Carolina	2	0	1	0	0	0	0	1	1	0	0	1	0	1	2	0	0	1	1	1	1	0	0	0	0	0
South Dakota	2	0	0	1	0	0	0	1	0	0	1	0	0	0	2	0	0	0	1	2	1	0	0	0	0	0
Tennessee	0	0	0	0	4	0	0	0	0	0	0	0	0	2	0	0	0	0	2	1	0	0	0	0	0	0
Texas	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0
Utah	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
Virginia	1	0	0	0	0	0	1	0	3	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0
Vermont	0	0	0	0	1	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1	0	1	0	0	0	0
Washington	1	0	0	0	0	0	1	1	1	0	0	0	0	2	1	0	0	0	1	1	0	0	1	0	0	0
Wisconsin	0	0	1	0	0	0	0	0	2	0	0	0	0	2	1	0	0	0	2	0	0	0	1	0	0	0
West Virginia	1	0	0	0	1	0	1	0	3	0	0	0	0	1	0	0	0	1	1	1	0	1	1	0	0	0
Wyoming	0	0	0	0	0	0	1	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	1	0	1	0

Part d

```
#https://cran.r-project.org/web/packages/fiftystater/vignettes/fiftystater.html
library(ggplot2)
library(fiftystater)
data("fifty_states") # this line is optional due to lazy data loading
crimes <- data.frame(state = tolower(rownames(USArrests)), USArrests)
crimes$count_of_cities <- num_city$`# of cities`[1:50]
# map_id creates the aesthetic mapping to the state name column in your data
p <- ggplot(crimes, aes(map_id = state)) +
# map points to the fifty_states shape data
geom_map(aes(fill = count_of_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())
p
```



Problem 2

```
library(quantreg)
```

```
## Loading required package: SparseM
```

```
##
## Attaching package: 'SparseM'
```

```
## The following object is masked from 'package:base':
##
##     backsolve
```

```
library(quantmod)
```

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
##
```

```
## Attaching package: 'xts'

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

## The following objects are masked from 'package:data.table':
##
##     first, last

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```
#1)fetch data from Yahoo
#AAPL prices
apple08 <- getSymbols('AAPL', auto.assign = FALSE, from = '2008-1-1', to =
"2008-12-31")[,6]
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
```

```
#market proxy
rm08<-getSymbols('^ixic', auto.assign = FALSE, from = '2008-1-1', to =
"2008-12-31")[,6]

#log returns of AAPL and market
logapple08<- na.omit(ROC(apple08)*100)
logrm08<-na.omit(ROC(rm08)*100)

#OLS for beta estimation
beta_AAPL_08<-summary(lm(logapple08~logrm08))$coefficients[2,1]

#create df from AAPL returns and market returns
df08<-cbind(logapple08,logrm08)
set.seed(666)
Boot=1000
sd.boot=rep(0,Boot)
for(i in 1:Boot){
# nonparametric bootstrap
bootdata=df08[sample(nrow(df08), size = 251, replace = TRUE),]
sd.boot[i]= coef(summary(lm(AAPL.Adjusted~IXIC.Adjusted, data = bootdata)))[2,2]
}
head(sd.boot)
```

```
## [1] 0.06861697 0.05623268 0.05940469 0.07227114 0.04946579 0.06534476
```

The problem is the formular in lm is wrong. I have already correct them.

Problem 3

```
url_sen<-"https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"
df_sen<-read.table(url_sen, skip=1, fill=TRUE, header=TRUE)

cur_item = 1
align_data <- function(row_){
    if (is.na(row_['X5'])){
      row_[2:6] <- row_[1:5]
      row_[1] <- cur_item
    }else{
      cur_item <- row_[1]
    }
    return(row_)
}
df_sen_align<-data.table(t(apply(df_sen, 1, align_data)))
set.seed(666)
Boot=1000
beta.boot=rep(0,Boot)
for(i in 1:Boot){
# nonparametric bootstrap
  bootdata=df_sen_align[sample(nrow(df_sen_align), size = nrow(df_sen_align), replace = TRUE),]
  beta.boot[i]= coef(summary(lm(Item~., data = bootdata)))[2,2]
}
head(beta.boot)
```

```
## [1] 0.7970865 0.7471297 0.7815065 0.6511281 0.5847815 1.0509479
```

Part c

```
library(parallel)
library(foreach)
library(doParallel)
```

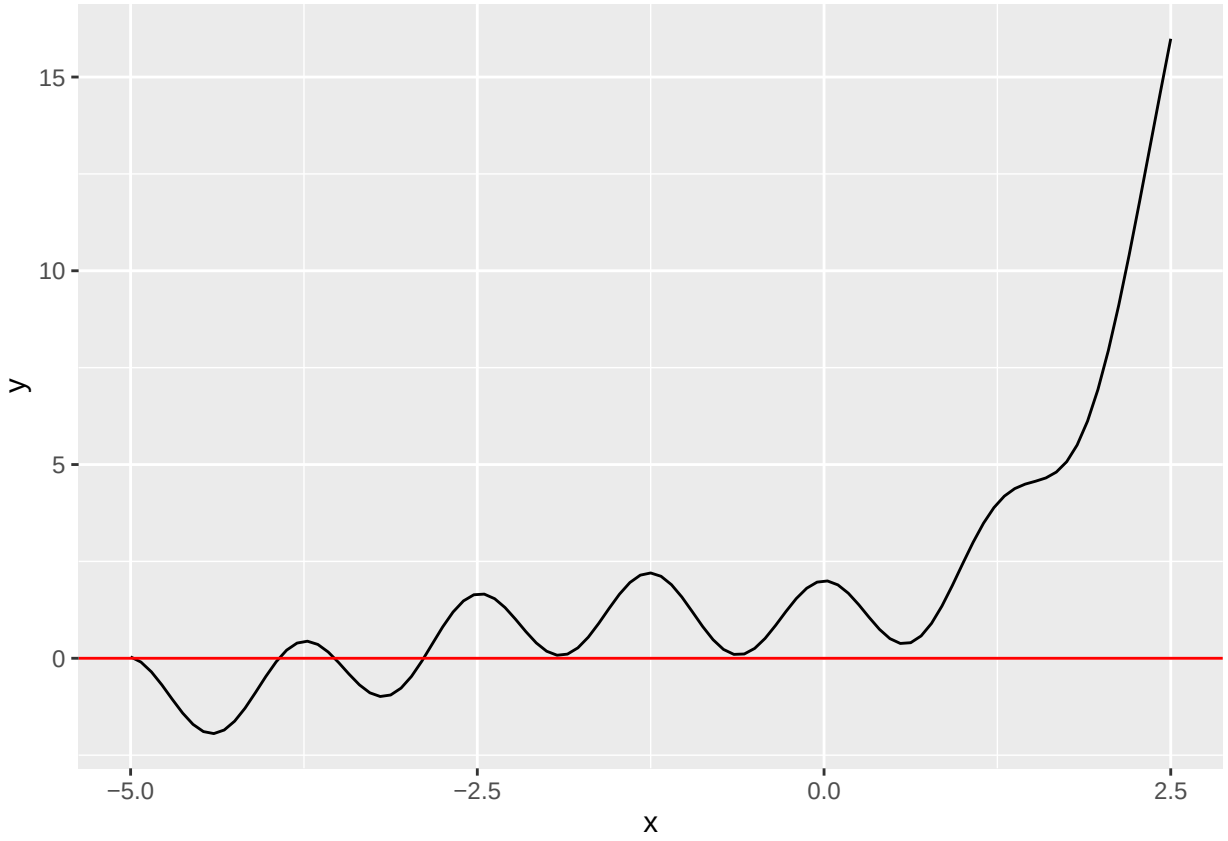
```
## Loading required package: iterators
```

```
cores <- detectCores()
cl <- makeCluster(cores - 1)
# system.time({
#   para <- function(){
#     bootdata=df_sen_align[sample(nrow(df_sen_align), size = nrow(df_sen_align), replace = TRUE),]
#     beta.boot[i]= coef(summary(lm(Item~., data = bootdata)))[2,2]
#     return(beta.boot)
#   }
#
#   Boot <- matrix(1:100, ncol = 100)
#   clusterExport(cl = cl, varlist = c("Boot", "para", "df_sen_align"), envir = environment())
#   beta <- t(parSapply(cl, Boot, para))
# })
stopCluster(cl)
```

Problem 3

Part a

```
fx <- function(x){
  y <- 3^x - sin(x) + cos(5*x)
}

fdx <- function(x){
  y <- 3^x*log(3) - cos(x) - 5*sin(5*x)
}

ggplot(data = data.frame(x = 0,y = 0), mapping = aes(x = x)) +
  stat_function(fun = fx) +
  xlim(-5, 2.5) +
  geom_abline(intercept = 0, slope = 0, colour = "red")
```




```
x <- -2.5
iter <- function(x){
  while (abs(fx(x)-0) > 1e-6) {
    x <- x - fx(x)/fdx(x)
  }
  return(x)
}
iter(x)
```

```
## [1] -3.930114
```

```r
system.time(s <- lapply(-999:0, iter))
```

```
##    user  system elapsed
##   0.008   0.000   0.008
```

Part b

```r
system.time(s <- mclapply(-999:0, iter, mc.cores = 8))
```

```
##    user  system elapsed
##   0.042   0.064   0.056
```