Bios 6301: Assignment 6

Rui Wang

Grade 50/50

Due Thursday, 1 December, 1:00 PM

$5^{n=day}$ points taken off for each day late.

50 points total.

Submit a single knitr file (named `homework6.rmd`), along with a valid PDF output file. Inside the file, clearly indicate which parts of your responses go with which problems (you may use the original homework document as a template). Add your name as `author` to the file's metadata section. Raw R code/output or word processor files are not acceptable.

Failure to name file `homework6.rmd` or include author name may result in 5 points taken off.

Question 1

15 points

Consider the following very simple genetic model (*very* simple – don't worry if you're not a geneticist!). A population consists of equal numbers of two sexes: male and female. At each generation men and women are paired at random, and each pair produces exactly two offspring, one male and one female. We are interested in the distribution of height from one generation to the next. Suppose that the height of both children is just the average of the height of their parents, how will the distribution of height change across generations?

Represent the heights of the current generation as a dataframe with two variables, m and f, for the two sexes. We can use `rnorm` to randomly generate the population at generation 1:

```
pop <- data.frame(m = rnorm(100, 160, 20), f = rnorm(100, 160, 20))
```

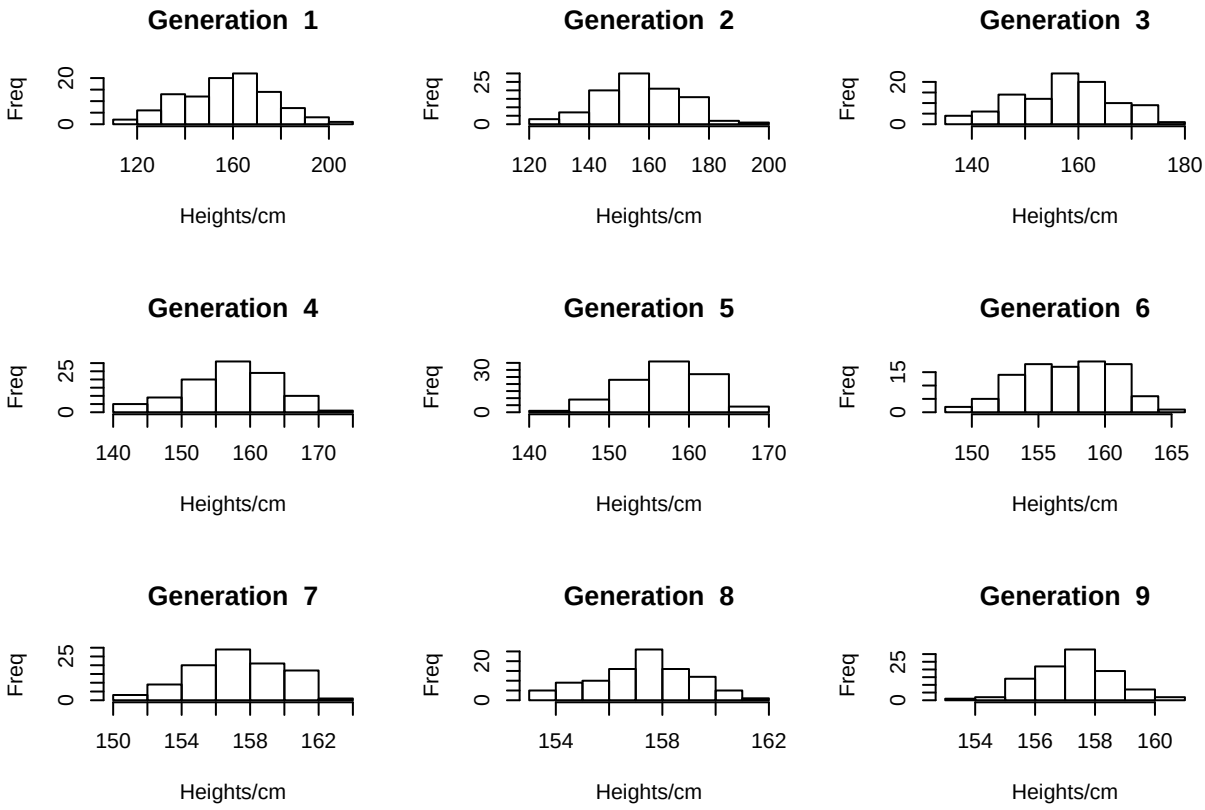
The following function takes the data frame `pop` and randomly permutes the ordering of the men. Men and women are then paired according to rows, and heights for the next generation are calculated by taking the mean of each row. The function returns a data frame with the same structure, giving the heights of the next generation.

```
next_gen <- function(pop) {
    pop$m <- sample(pop$m)
    pop$m <- rowMeans(pop)
    pop$f <- pop$m
    pop
}
```

Use the function `next_gen` to generate nine generations (you already have the first), then use the function `hist` to plot the distribution of male heights in each generation (this will require multiple calls to `hist`). The phenomenon you see is called regression to the mean. Provide (at least) minimal decorations such as title and x-axis labels.

```
pop_all <- data.frame()
# use pop_temp to storage the data in each generation
# combine all data in one data.frame
pop_temp <- pop
pop_temp$generation <- 1
pop_all <- pop_temp
```

```
for (i in 2:9) {
  pop <- next_gen(pop)
  pop_temp <- pop
  pop_temp$generation <- i
  pop_all <- rbind(pop_all, pop_temp)
}
# set up a 3*3 graphical chart
par(mfrow = c(3,3))
for (i in 1:9) {
  # plot the histogram of male heights in each generation
  hist(pop_all[which(pop_all$generation == i), ]$m, xlab = c('Heights/cm'), ylab = c('Freq'), main = pas
}
```



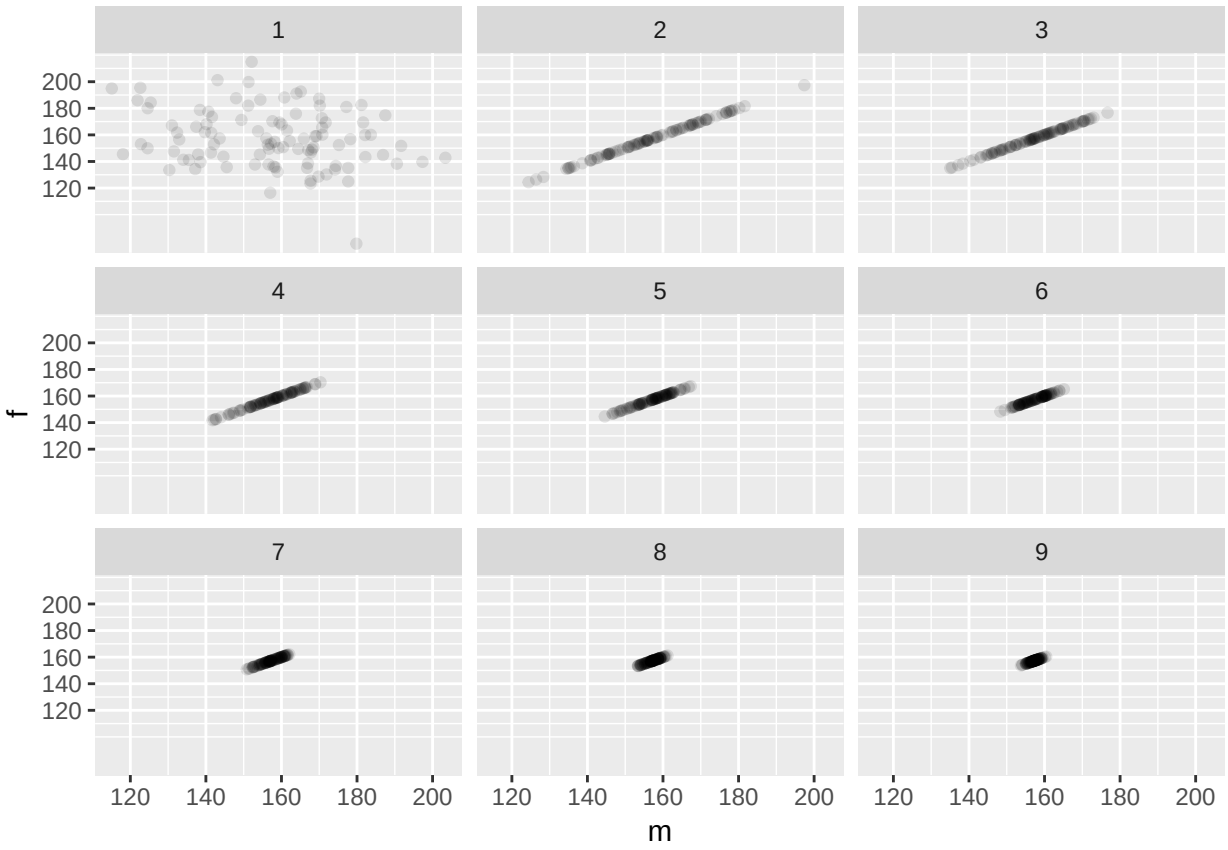
Clearly, the heights of male is regressed to the mean value (160) as we generate from the parent genetation.

Question 2

10 points

Use the simulated results from question 1 to reproduce (as closely as possible) the following plot in ggplot2.

```
library(testthat)
library(ggplot2)
ggplot(pop_all, aes(m, f)) + geom_point(alpha=1/10) + scale_x_continuous(breaks=seq(100,220,20)) + scal
```



Question 3

10 points

You calculated the power of a study design in question #2 of assignment 3. The study has two variables, treatment group and outcome. There are two treatment groups (0, 1) and they should be assigned randomly with equal probability. The outcome should be a random normal variable with a mean of 60 and standard deviation of 20. If a patient is in the treatment group, add 5 to the outcome.

Starting with a sample size of 250, create a 95% bootstrap percentile interval for the mean of each group. Then create a new bootstrap interval by increasing the sample size by 250 until the sample is 2500. Thus you will create a total of 10 bootstrap intervals. Each bootstrap should create 1000 bootstrap samples. (4 points)

```
# initial setting
t_mean <- c()
t_mean_ub <- c()
t_mean_lb <- c()
c_mean <- c()
c_mean_ub <- c()
c_mean_lb <- c()
# create a new bootstrap interval by increasing the sample size by 250 until the sample is 2500
for(i in seq(1:10)){
  # set sample size
  n <- i*250
  treat <- rbinom(n,1,0.5)
  # add 5 to the outcome if the treat is 1
  outcome <- rnorm(n, 60, 20) + treat * 5
```

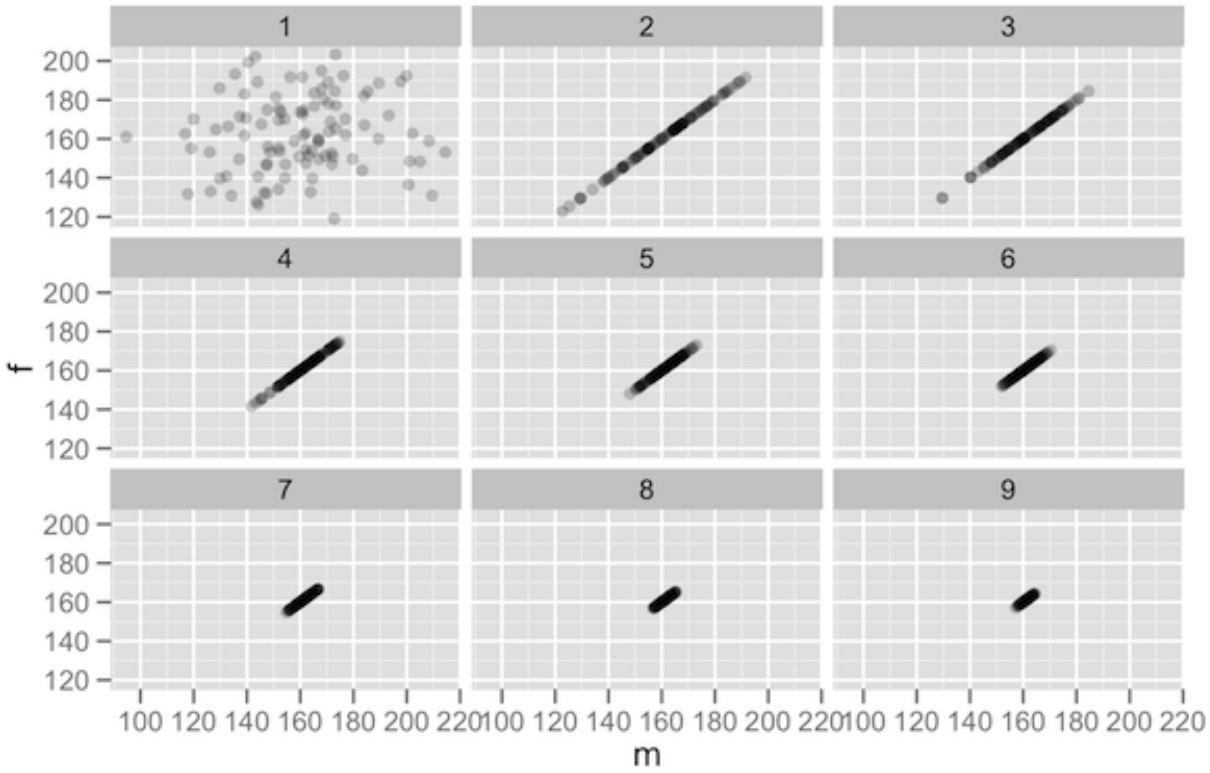
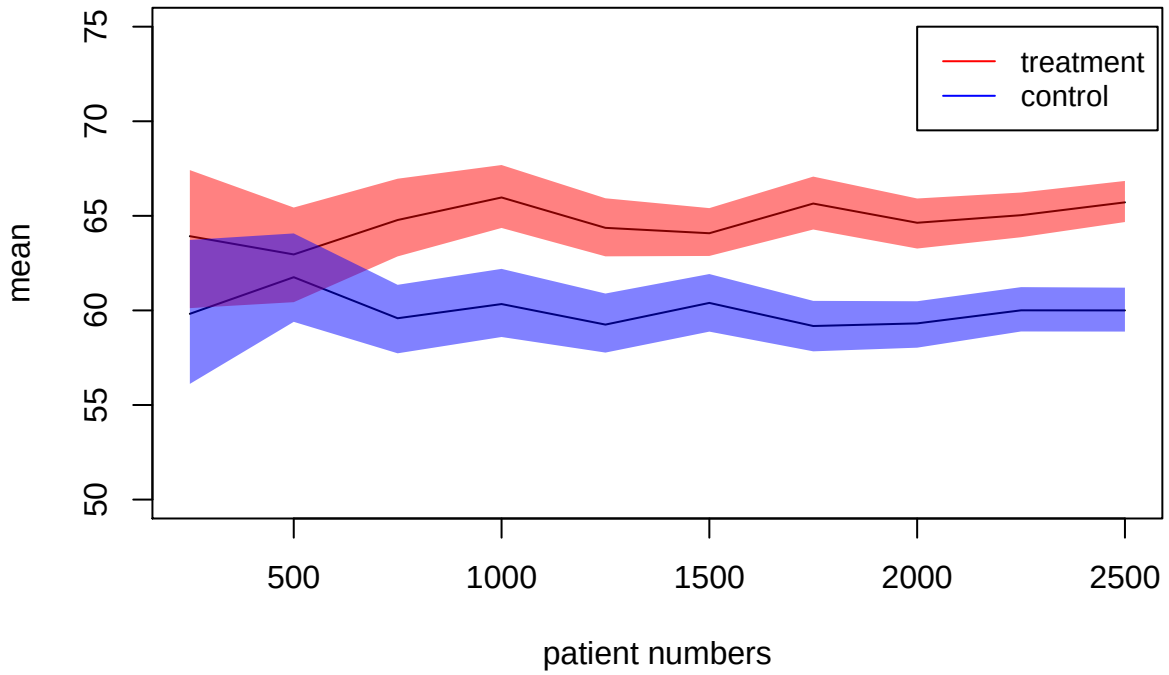


Figure 1: generations plot

```
  # sort the outcome by treat results into t and c
  t <- outcome[treat == 1]
  c <- outcome[treat == 0]
  # Each bootstrap should create 1000 bootstrap samples
  t_bootstrap <- replicate(1000, mean(sample(t, size = length(t), replace = TRUE)))
  c_bootstrap <- replicate(1000, mean(sample(c, size = length(c), replace = TRUE)))
  # calculate the mean, ub and ub for each bootstrap
  t_mean <- c(t_mean,mean(t))
  t_mean_ub <- c(t_mean_ub, quantile(t_bootstrap, 0.975))
  t_mean_lb <- c(t_mean_lb, quantile(t_bootstrap, 0.025))
  c_mean <- c(c_mean,mean(c))
  c_mean_ub <- c(c_mean_ub, quantile(c_bootstrap, 0.975))
  c_mean_lb <- c(c_mean_lb, quantile(c_bootstrap, 0.025))
}
```

Produce a line chart that includes the bootstrapped mean and lower and upper percentile intervals for each group. Add appropriate labels and a legend. (6 points)

```
makeTransparent = function(..., alpha=0.5) {
  if(alpha<0 | alpha>1) stop("alpha must be between 0 and 1")
  alpha = floor(255*alpha)
  newColor = col2rgb(col=unlist(list(...)), alpha=FALSE)
  .makeTransparent = function(col, alpha) {
    rgb(red=col[1], green=col[2], blue=col[3], alpha=alpha, maxColorValue=255)
  }
  newColor = apply(newColor, 2, .makeTransparent, alpha=alpha)
  return(newColor)
}
plot(x = seq(250,2500,250),y = t_mean, type = "l", xlab ="patient numbers", ylab = "mean", xlim = c(250
lines(x = seq(250,2500,250),y = c_mean)
# use rev
polygon(x=c(seq(250, 2500, length.out = 10),seq(2500, 250, length.out = 10)),y = c(t_mean_lb, rev(t_mea
polygon(x=c(seq(250, 2500, length.out = 10),seq(2500, 250, length.out = 10)),y = c(c_mean_lb, rev(c_mea
legend(2000,75,c("treatment","control"),lty = c(1,1,1),lwd = c(1,1,1),col = c("red","blue"), cex = 0.9)
```



You may use base graphics or ggplot2. It should look similar to this (in base).

Here's an example of how you could create transparent shaded areas.

```
makeTransparent = function(..., alpha=0.5) {
  if(alpha<0 | alpha>1) stop("alpha must be between 0 and 1")
  alpha = floor(255*alpha)
  newColor = col2rgb(col=unlist(list(...)), alpha=FALSE)
  .makeTransparent = function(col, alpha) {
    rgb(red=col[1], green=col[2], blue=col[3], alpha=alpha, maxColorValue=255)
  }
  newColor = apply(newColor, 2, .makeTransparent, alpha=alpha)
  return(newColor)
}

par(new=FALSE)
plot(NULL,
  xlim=c(-1, 1),
  ylim=c(-1, 1),
  xlab="",
  ylab=""
)

polygon(x=c(seq(-0.75, 0.25, length.out=100), seq(0.25, -0.75, length.out=100)),
        y=c(rep(-0.25, 100), rep(0.75, 100)), border=NA, col=makeTransparent('blue',alpha=0.5))
polygon(x=c(seq(-0.25, 0.75, length.out=100), seq(0.75, -0.25, length.out=100)),
        y=c(rep(-0.75, 100), rep(0.25, 100)), border=NA, col=makeTransparent('red',alpha=0.5))
```

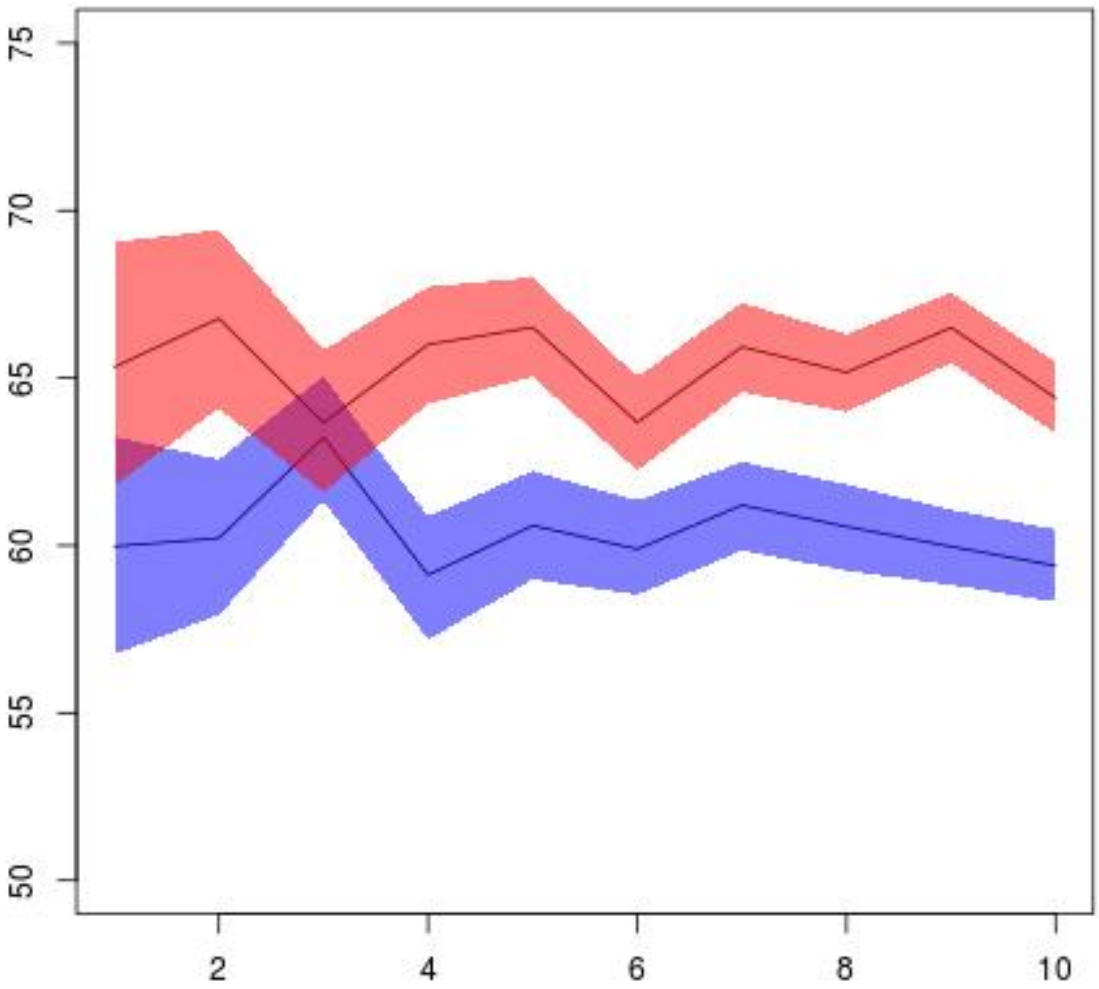
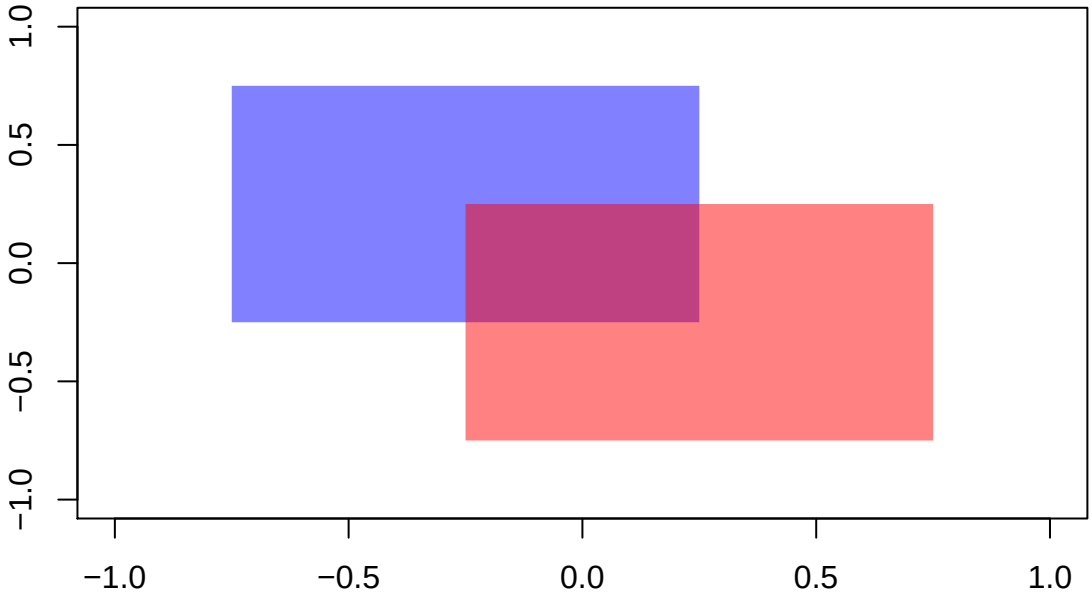


Figure 2: bp interval plot



Question 4

15 points

Programming with classes. The following function will generate random patient information.

```
makePatient <- function() {
  vowel <- grep("[aeiou]", letters)
  cons <- grep("[^aeiou]", letters)
  name <- paste(sample(LETTERS[cons], 1), sample(letters[vowel], 1), sample(letters[cons], 1), sep='')
  gender <- factor(sample(0:1, 1), levels=0:1, labels=c('female','male'))
  dob <- as.Date(sample(7500, 1), origin="1970-01-01")
  n <- sample(6, 1)
  doa <- as.Date(sample(1500, n), origin="2010-01-01")
  pulse <- round(rnorm(n, 80, 10))
  temp <- round(rnorm(n, 98.4, 0.3), 2)
  fluid <- round(runif(n), 2)
  list(name, gender, dob, doa, pulse, temp, fluid)
}
```

1. Create an S3 class `medicalRecord` for objects that are a list with the named elements `name`, `gender`, `date_of_birth`, `date_of_admission`, `pulse`, `temperature`, `fluid_intake`. Note that an individual patient may have multiple measurements for some measurements. Set the RNG seed to `8` and create a medical record by taking the output of `makePatient`. Print the medical record, and print the class of the medical record. (5 points)

```
# change some code above
makePatient <- function() {
  vowel <- grep("[aeiou]", letters)
  cons <- grep("[^aeiou]", letters)
  name <- paste(sample(LETTERS[cons], 1), sample(letters[vowel], 1), sample(letters[cons], 1), sep='')
  gender <- factor(sample(0:1, 1), levels=0:1, labels=c('female','male'))
  dob <- as.Date(sample(7500, 1), origin="1970-01-01")
  n <- sample(6, 1)
  doa <- as.Date(sample(1500, n), origin="2010-01-01")
  pulse <- round(rnorm(n, 80, 10))
```

```
  temp <- round(rnorm(n, 98.4, 0.3), 2)
  fluid <- round(runif(n), 2)
  list(name = name, gender = gender, date_of_birth = dob, date_of_admission = doa, pulse = pulse, tempe
}

set.seed(8)
# create a medical record by taking the output of `makePatient`
output1 <- makePatient()
output1
```

```
## $name
## [1] "Mev"
##
## $gender
## [1] male
## Levels: female male
##
## $date_of_birth
## [1] "1976-08-09"
##
## $date_of_admission
## [1] "2011-03-14" "2013-10-30" "2013-02-27" "2012-08-23" "2011-11-16"
##
## $pulse
## [1] 67 81 95 74 81
##
## $temperature
## [1] 98.33 98.16 99.00 98.49 98.67
##
## $fluid_intake
## [1] 0.62 0.93 0.18 0.39 0.34
```

```
# Create an S3 class `medicalRecord`
class(output1) <- "medicalRecord"
class(output1)
```

```
## [1] "medicalRecord"
```

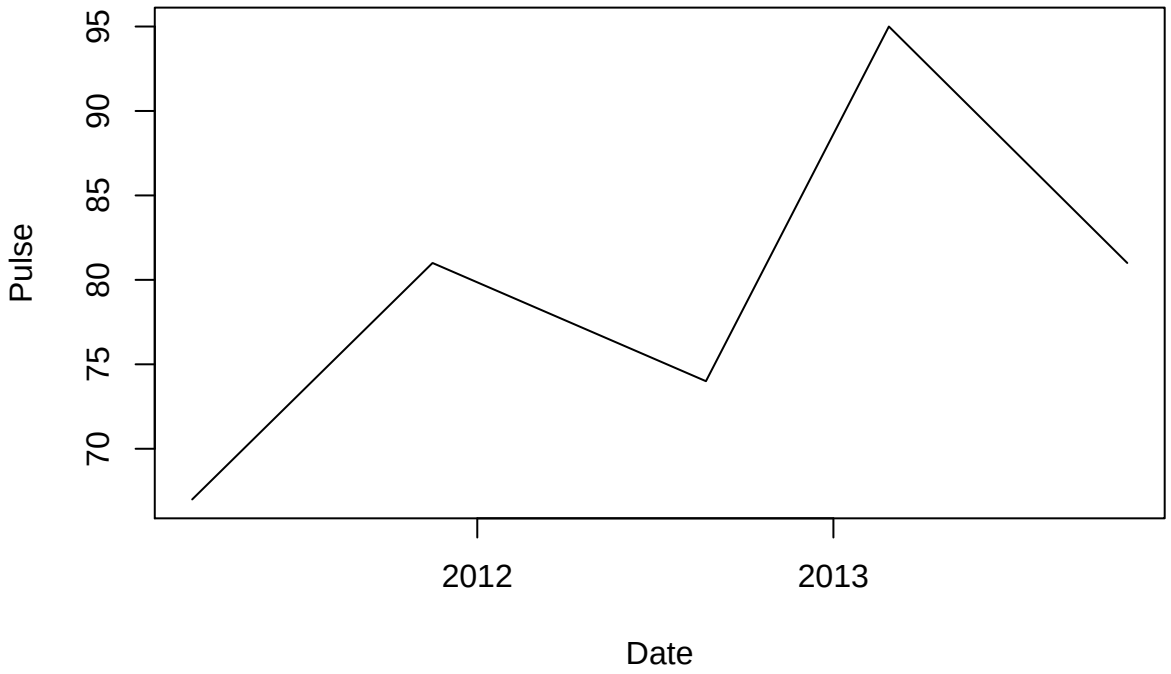
2. Write a `medicalRecord` method for the generic function `mean`, which returns averages for pulse, temperature and fluids. Also write a `medicalRecord` method for `print`, which employs some nice formatting, perhaps arranging measurements by date, and `plot`, that generates a composite plot of measurements over time. Call each function for the medical record created in part 1. (5 points)

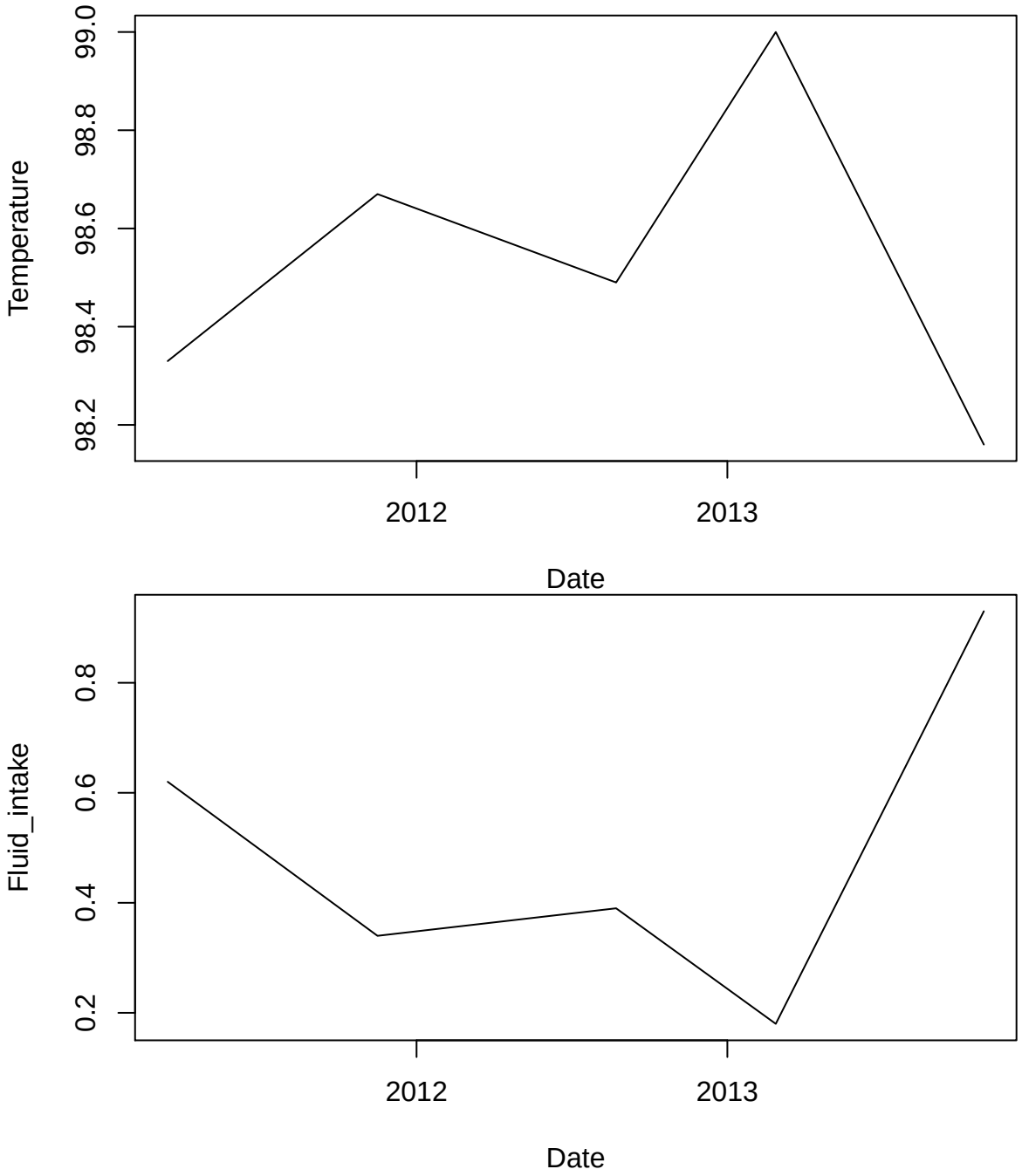
```
mean.medicalRecord <- function(p){
    list(pulse_ave = mean(p$pulse), temp_ave = mean(p$temperature),fluids_ave = mean(p$fluid_intake))
}
mean(output1)
```

```
## $pulse_ave
## [1] 79.6
##
## $temp_ave
## [1] 98.53
##
## $fluids_ave
## [1] 0.492
```

```r
print.medicalRecord <- function(p){
  dat <- data.frame(date_of_admission = p$date_of_admission, pulse = p$pulse, temperature = p$temperatu
  dat <- dat[order(dat$date_of_admission),]
}
print(output1)


plot.medicalRecord <- function(p){
  dat <- data.frame(date_of_admission = p$date_of_admission, pulse = p$pulse, temperature = p$temperatu
  dat <- dat[order(dat$date_of_admission),]
  plot(x=dat$date_of_admission, y=dat$pulse, type = 'l', xlab = "Date", ylab = 'Pulse')
  plot(x=dat$date_of_admission, y=dat$temperature, type = 'l', xlab = "Date", ylab = 'Temperature')
  plot(x=dat$date_of_admission, y=dat$fluid_intake, type = 'l', xlab = "Date", ylab = 'Fluid_intake')
}
plot(output1)
```





3. Create a further class for a cohort (group) of patients, and write methods for `mean` and `print` which, when applied to a cohort, apply mean or print to each patient contained in the cohort. Hint: think of this as a "container" for patients. Reset the RNG seed to 8 and create a cohort of ten patients, then show the output for `mean` and `print`. (5 points)

```
set.seed(8)
output3 <- lapply(seq(10), function(p) {makePatient()})
class(output3) <- "cohort"

mean.cohort <- function(p){
    for(q in p) {
        class(q) <- "medicalRecord"
```

```
        print(mean(q))
    }
}
mean(output3)
```

```
## $pulse_ave
## [1] 79.6
##
## $temp_ave
## [1] 98.53
##
## $fluids_ave
## [1] 0.492
##
## $pulse_ave
## [1] 78
##
## $temp_ave
## [1] 98.495
##
## $fluids_ave
## [1] 0.245
##
## $pulse_ave
## [1] 81.5
##
## $temp_ave
## [1] 98.44
##
## $fluids_ave
## [1] 0.4033333
##
## $pulse_ave
## [1] 78
##
## $temp_ave
## [1] 98.6
##
## $fluids_ave
## [1] 0.65
##
## $pulse_ave
## [1] 88.33333
##
## $temp_ave
## [1] 98.05
##
## $fluids_ave
## [1] 0.5866667
##
## $pulse_ave
## [1] 83.5
##
## $temp_ave
```

```
## [1] 98.45
##
## $fluids_ave
## [1] 0.4525
##
## $pulse_ave
## [1] 83
##
## $temp_ave
## [1] 98.01
##
## $fluids_ave
## [1] 0.97
##
## $pulse_ave
## [1] 77.5
##
## $temp_ave
## [1] 98.14833
##
## $fluids_ave
## [1] 0.3366667
##
## $pulse_ave
## [1] 77
##
## $temp_ave
## [1] 98.83
##
## $fluids_ave
## [1] 0.445
##
## $pulse_ave
## [1] 79.33333
##
## $temp_ave
## [1] 98.3
##
## $fluids_ave
## [1] 0.6583333
```

```
print.cohort <- function(p){
    for(q in p) {
        class(q) <- "medicalRecord"
        print(print(q))
    }
}
print(output3)
```

```
##   date_of_admission pulse temperature fluid_intake
## 1        2011-03-14    67       98.33         0.62
## 5        2011-11-16    81       98.67         0.34
## 4        2012-08-23    74       98.49         0.39
## 3        2013-02-27    95       99.00         0.18
## 2        2013-10-30    81       98.16         0.93
```

```
##   date_of_admission pulse temperature fluid_intake
## 1        2012-01-16    76       98.92         0.14
## 2        2013-08-07    80       98.07         0.35
##   date_of_admission pulse temperature fluid_intake
## 6        2010-03-21    79       98.58         0.22
## 5        2010-04-01    73       98.32         0.61
## 4        2012-08-29    88       98.47         0.59
## 3        2013-06-01    84       98.22         0.25
## 1        2013-11-03    72       98.54         0.03
## 2        2014-02-05    93       98.51         0.72
##   date_of_admission pulse temperature fluid_intake
## 1        2011-06-22    78        98.6         0.65
##   date_of_admission pulse temperature fluid_intake
## 3        2010-04-12    76       98.05         0.65
## 1        2011-02-16    93       98.26         0.97
## 2        2012-04-12    96       97.84         0.14
##   date_of_admission pulse temperature fluid_intake
## 4        2010-03-10    81       99.11         0.66
## 1        2010-03-25    90       98.58         0.26
## 3        2010-04-18    75       98.58         0.60
## 2        2010-06-10    88       97.53         0.29
##   date_of_admission pulse temperature fluid_intake
## 1        2010-03-12    83       98.01         0.97
##   date_of_admission pulse temperature fluid_intake
## 5        2011-04-07    80       97.87         0.36
## 4        2011-04-14    83       97.91         0.00
## 2        2011-08-16    66       98.49         0.13
## 1        2013-03-15    74       98.38         0.31
## 6        2013-06-20    74       98.41         0.49
## 3        2013-11-12    88       97.83         0.73
##   date_of_admission pulse temperature fluid_intake
## 1        2010-10-30    85       98.84         0.60
## 2        2012-05-10    69       98.82         0.29
##   date_of_admission pulse temperature fluid_intake
## 4        2010-01-28    63       97.95         0.94
## 3        2010-03-06    81       98.45         0.67
## 1        2010-07-10    98       98.65         0.79
## 6        2010-08-27    66       97.68         0.36
## 5        2011-06-18    83       98.00         0.69
## 2        2013-01-06    85       99.07         0.50
```