Bios 6301: Assignment 3

Josh DeClercq

JC Grade: 43/50

Due Tuesday, 11 October, 1:00 PM

50 points total.

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

This assigment includes turning in the first two assignments. All three should include knitr files (named `homework1.rmd`, `homework2.rmd`, `homework3.rmd`) along with valid PDF output files. Inside each 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 properly name files or include author name may result in 5 points taken off.

Question 1

10 points

1. Use GitHub to turn in the first three homework assignments. Make sure the teacher (couthcommander) and TA (chipmanj) are collaborators. (5 points)
2. Commit each assignment individually. This means your repository should have at least three commits. (5 points)

Question 2

15 points

Write a simulation to calculate the power for the following study design. 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. 5 is the true treatment effect. Create a linear model for the outcome by the treatment group, and extract the p-value (hint: see assignment1). Test if the p-value is less than or equal to the alpha level, which should be set to 0.05.

Repeat this procedure 1000 times. The power is calculated by finding the percentage of times the p-value is less than or equal to the alpha level. Use the `set.seed` command so that the professor can reproduce your results.

```
getpower = function(patients, nsim) {
    p.test <- numeric(nsim)
    set.seed(12356)
    for (i in seq_along(p.test)) {
        outcome = rnorm(patients, 60, 20)
        treatment <- sample(patients)%%2
        p.test[i] = summary(lm(outcome ~ treatment))$coefficients[2, 4]
    }
    mean(p.test < 0.05)
}
ans1 <- getpower(patients = 100, nsim = 1000)
```

```
ans2 <- getpower(patients = 1000, nsim = 1000)
true.power <- power.t.test(n = 100, delta = 0.5, sd = 20, sig.level = 0.05,
    type = "one.sample")$power
```

1. Find the power when the sample size is 100 patients. (10 points)

The power when the sample size is 100 patients is 0.044.

JC Grading -5 for questions 1 and 2 Outcome for treatments should be 5 greater in group one. Consider something like the slightly modified code below:

```
getpower = function(patients, nsim) {
    p.test <- numeric(nsim)
    set.seed(12356)
    for (i in seq_along(p.test)) {
        outcome = rnorm(patients, 60, 20)
        treatment <- sample(patients)%%2
        outcome[treatment == 1] <- outcome[treatment == 1] + 5
        p.test[i] = summary(lm(outcome ~ treatment))$coefficients[2, 4]
    }
    mean(p.test < 0.05)
}
getpower(patients = 100, nsim = 1000)
```

```
## [1] 0.246
```

2. Find the power when the sample size is 1000 patients. (5 points)

The power when the sample size is 1000 patients is 0.043. The true power for this test is 0.0434136.

Question 3

15 points

Obtain a copy of the football-values lecture. Save the `2016/proj_wr16.csv` file in your working directory. Read in the data set and remove the first two columns.

1. Show the correlation matrix of this data set. (3 points)

```
#my.df <- read.csv('/Users/Bowie/Dropbox/Stat_Com/proj_wr16.csv')
my.df <- read.csv('proj_wr16.csv')
```

```
library(abind)
library(corrplot)
library(MASS)
```

```
head(my.df)
```

```
##          PlayerName Team rush_att rush_yds rush_tds rec_att rec_yds
## 1     Antonio Brown  PIT      3.1     17.0        0   123.6  1648.8
## 2       Julio Jones  ATL      0.3      1.6        0   116.6  1623.5
## 3 Odell Beckham Jr.  NYG      0.8      4.8        0    98.0  1439.5
## 4   DeAndre Hopkins  HOU      0.0      0.0        0   100.0  1423.2
## 5        Dez Bryant  DAL      0.0      0.0        0    85.2  1195.1
## 6        A.J. Green  CIN      0.0      0.1        0    87.4  1255.3
##   rec_tds fumbles  fpts
## 1    10.8     1.1 229.1
## 2     8.8     0.8 214.0
```

```
## 3    11.1     0.1 210.6
## 4     9.6     0.1 199.5
## 5    10.1     0.1 179.6
## 6     9.3     0.9 179.3
```

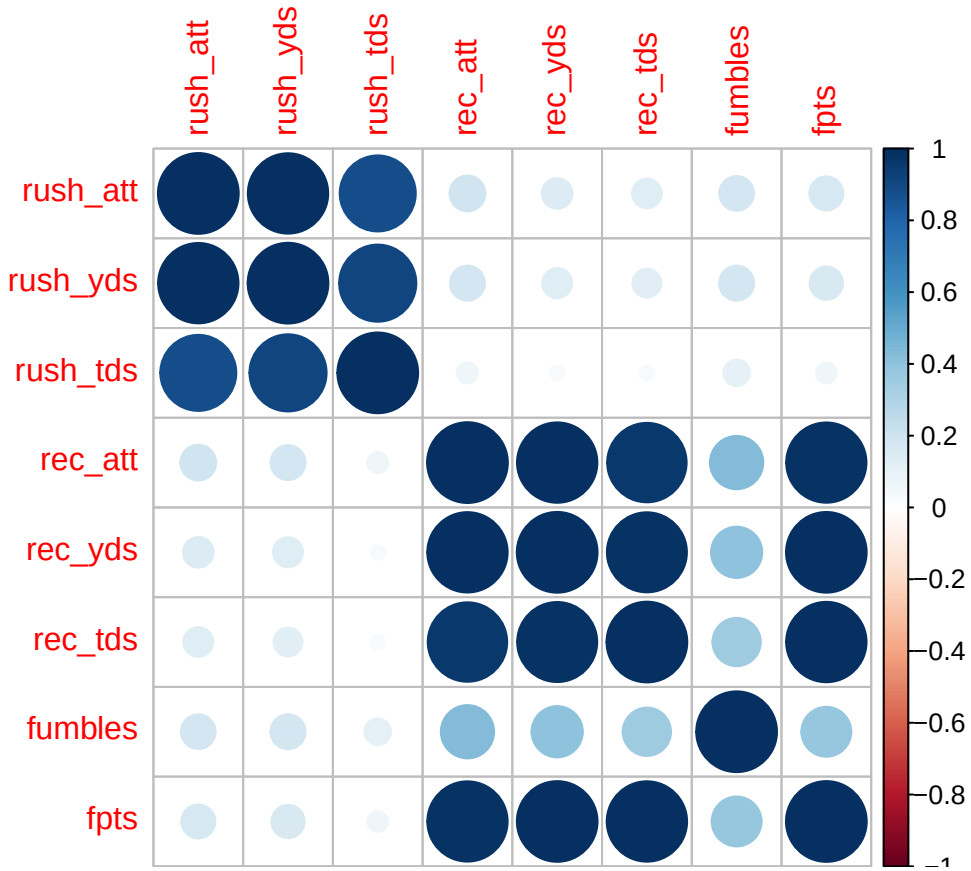
```
my.df <- my.df[ -c(1, 2) ]

head(my.df)
```

```
##   rush_att rush_yds rush_tds rec_att rec_yds rec_tds fumbles  fpts
## 1      3.1     17.0        0   123.6  1648.8    10.8     1.1 229.1
## 2      0.3      1.6        0   116.6  1623.5     8.8     0.8 214.0
## 3      0.8      4.8        0    98.0  1439.5    11.1     0.1 210.6
## 4      0.0      0.0        0   100.0  1423.2     9.6     0.1 199.5
## 5      0.0      0.0        0    85.2  1195.1    10.1     0.1 179.6
## 6      0.0      0.1        0    87.4  1255.3     9.3     0.9 179.3
```

```
df.cor <- cor(my.df)
df.cor
```

```
##           rush_att  rush_yds   rush_tds    rec_att    rec_yds    rec_tds
## rush_att 1.0000000 0.9906030 0.88608205 0.19706851 0.14473723 0.13548999
## rush_yds 0.9906030 1.0000000 0.91252627 0.18745520 0.13765791 0.12772327
## rush_tds 0.8860820 0.9125263 1.00000000 0.06914613 0.03114206 0.03163468
## rec_att  0.1970685 0.1874552 0.06914613 1.00000000 0.99002712 0.96757796
## rec_yds  0.1447372 0.1376579 0.03114206 0.99002712 1.00000000 0.98209522
## rec_tds  0.1354900 0.1277233 0.03163468 0.96757796 0.98209522 1.00000000
## fumbles  0.1844220 0.1881021 0.10845675 0.43577978 0.40349289 0.35852435
## fpts     0.1766540 0.1698501 0.06567865 0.98754942 0.99760259 0.99058639
##            fumbles       fpts
## rush_att 0.1844220 0.17665405
## rush_yds 0.1881021 0.16985010
## rush_tds 0.1084568 0.06567865
## rec_att  0.4357798 0.98754942
## rec_yds  0.4034929 0.99760259
## rec_tds  0.3585244 0.99058639
## fumbles  1.0000000 0.38269698
## fpts     0.3826970 1.00000000
```

```
corrplot(df.cor, method = "circle")
```




```
length(my.df$rush_att)
```

```
## [1] 243
```

2. Generate a data set with 30 rows that has a similar correlation structure. Repeat the procedure 10,000 times and return the mean correlation matrix. (10 points)

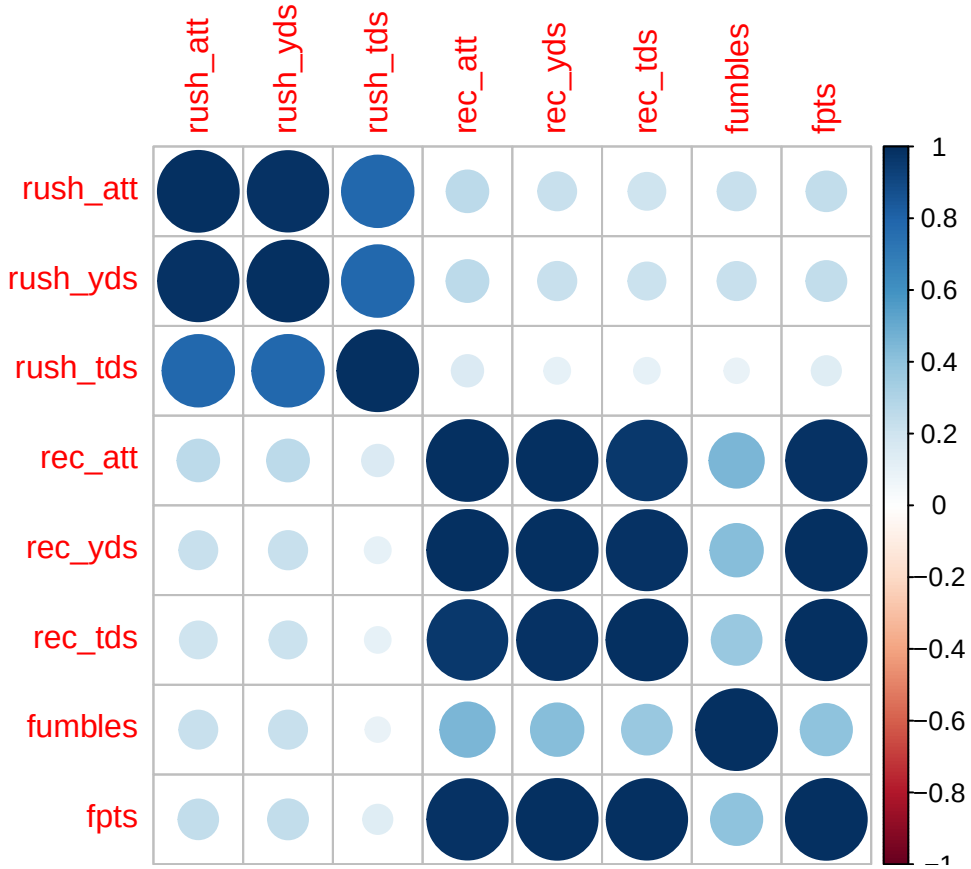
I really struggled with this one, and I had to download a new package (abind) to get it to work.

```
all.df <- list(replicate(10000, cor(my.df[sample(243,30,replace = FALSE),])))
all.matrix <- abind(all.df, along=3)
M <- apply(all.matrix, c(1,2),mean, na.rm = TRUE)
M
```

```
##           rush_att  rush_yds  rush_tds   rec_att   rec_yds   rec_tds
## rush_att 1.0000000 0.9897989 0.7829552 0.2657015 0.2206917 0.2079421
## rush_yds 0.9897989 1.0000000 0.7845247 0.2682600 0.2241008 0.2102626
## rush_tds 0.7829552 0.7845247 1.0000000 0.1507509 0.1029498 0.1012311
## rec_att  0.2657015 0.2682600 0.1507509 1.0000000 0.9906309 0.9696790
## rec_yds  0.2206917 0.2241008 0.1029498 0.9906309 1.0000000 0.9829910
## rec_tds  0.2079421 0.2102626 0.1012311 0.9696790 0.9829910 1.0000000
## fumbles  0.2210851 0.2221224 0.0922349 0.4520531 0.4211689 0.3794746
## fpts     0.2407492 0.2440134 0.1307024 0.9883720 0.9976327 0.9909887
##            fumbles      fpts
## rush_att 0.2210851 0.2407492
## rush_yds 0.2221224 0.2440134
## rush_tds 0.0922349 0.1307024
## rec_att  0.4520531 0.9883720
```

```
## rec_yds   0.4211689 0.9976327
## rec_tds   0.3794746 0.9909887
## fumbles   1.0000000 0.4016683
## fpts      0.4016683 1.0000000
```

```
corrplot(M, method = "circle")
```



3. Generate a data set with 30 rows that has the exact correlation structure as the original data set. (2 points)

I was not sure what is meant by 'exact correlation structure.' Here, I attempted to find 30 entries of the dataframe that make a correlation structure exactly equal to the correlation structure of all 243 entries. I think that this may be impossible to do.

```
x<- 0
repeat{
  x <- cor(my.df[sample(243,30,replace = FALSE),])
  if( all.equal(df.cor, x, tolerance = 0.01) == TRUE) break
}
x
```

I attempted to use some code I had in the lecture notes to achieve the same result. It works, but have to set a tolerance threshold in the test for equality.

```
vcov.df <- var(my.df)
means.df <- colMeans(my.df)

keep.1 <- 0
loops <- 1e4
```

```
for (i in seq(loops)){
df.sim <- mvrnorm(30, mu = means.df, Sigma = vcov.df, empirical = TRUE)
keep.1 <- keep.1 + cor(df.sim)/loops
}
keep.1
```

```
##           rush_att  rush_yds   rush_tds    rec_att    rec_yds    rec_tds
## rush_att 1.0000000 0.9906030 0.88608205 0.19706851 0.14473723 0.13548999
## rush_yds 0.9906030 1.0000000 0.91252627 0.18745520 0.13765791 0.12772327
## rush_tds 0.8860820 0.9125263 1.00000000 0.06914613 0.03114206 0.03163468
## rec_att  0.1970685 0.1874552 0.06914613 1.00000000 0.99002712 0.96757796
## rec_yds  0.1447372 0.1376579 0.03114206 0.99002712 1.00000000 0.98209522
## rec_tds  0.1354900 0.1277233 0.03163468 0.96757796 0.98209522 1.00000000
## fumbles  0.1844220 0.1881021 0.10845675 0.43577978 0.40349289 0.35852435
## fpts     0.1766540 0.1698501 0.06567865 0.98754942 0.99760259 0.99058639
##            fumbles       fpts
## rush_att 0.1844220 0.17665405
## rush_yds 0.1881021 0.16985010
## rush_tds 0.1084568 0.06567865
## rec_att  0.4357798 0.98754942
## rec_yds  0.4034929 0.99760259
## rec_tds  0.3585244 0.99058639
## fumbles  1.0000000 0.38269698
## fpts     0.3826970 1.00000000
```

```
 all.equal(df.cor, keep.1, tolerance = 0.0000000001)
```

```
## [1] TRUE
```

JC Grading -2 Looking for a single generated dataset (rather than running through loop).

Below are two attmpts I made to extract the 30 rows of data that would generate the correlation matrix, but I definitely did not figure this one out. When we went over simulation in lecture, I had a really hard time keeping up with copying the code and I hardly had a chance to understand what was happening nor did I have a chance to annotate my code so I could make sense of it later. I could not find anything like this in the class notes or github repository to help me along.

```
vcov.df <- var(my.df)
means.df <- colMeans(my.df)

df.sim <- list()
keep.1 <- 0
loops <- 1e4
for (i in seq(loops)){
df.sim[i] <- mvrnorm(30, mu = means.df, Sigma = vcov.df, empirical = TRUE)/loops
}
X.matrix <- abind(df.sim, along=3)
X <- apply(X.matrix, c(1,2),mean, na.rm = TRUE)
X
cor(X)
```

```
X.df <- list(replicate(10000, mvrnorm(30, mu = means.df, Sigma = vcov.df, empirical = TRUE)))
X.matrix <- abind(X.df, along = 3)
X <- apply(X.matrix, c(1, 2), mean, na.rm = TRUE)
X
cor(X)
```

Question 4

10 points

Use LaTeX to create the following expressions.
Note: These look better in html than pdf. I couldn't figure out how to make the functions work with the pdf output. Also, is it possible to load Latex packages into Rmarkdown?

1. Hint: `\Rightarrow` (4 points)

$$P(B) = \sum_j P(B|A_j)P(A_j), \Rightarrow P(A_i|B) = \frac{P(B|A_i)P(A_i)}{\sum_j P(B|A_j)P(A_j)}$$

2. Hint: `\zeta` (3 points)

$$\hat{f}(\zeta) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i x \zeta} dx$$

3. Hint: `\partial` (3 points)

$$\mathbf{J} = \frac{d\mathbf{f}}{d\mathbf{X}} = \begin{bmatrix} \frac{\partial \mathbf{f}}{\partial x_1} & \cdots & \frac{\partial \mathbf{f}}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

JC Comment I think the PDF output looks fine.