DATA 556: Homework 4

Will Wright

October 25, 2018

```
# load packages
library(ggplot2)
set.seed(0) # set seed for reproducibility
```

General note

The encoding of certain math symbols like theta are not working as expected and display as '??'. This, however, will have no impact on the answers–just the quoated questions.

Problem 2a

A circle with a random radius R ~ Unif(0,1) is generated. Let A be its area. a.) Use simulations in R (the statistical programming language) to numerically estimate the mean and variance of A.

```
circleArea <- pi*runif(10000)^2 # simulate 10,000 circle areas
mean(circleArea)
```

```
## [1] 1.052201
```

```
var(circleArea)
```

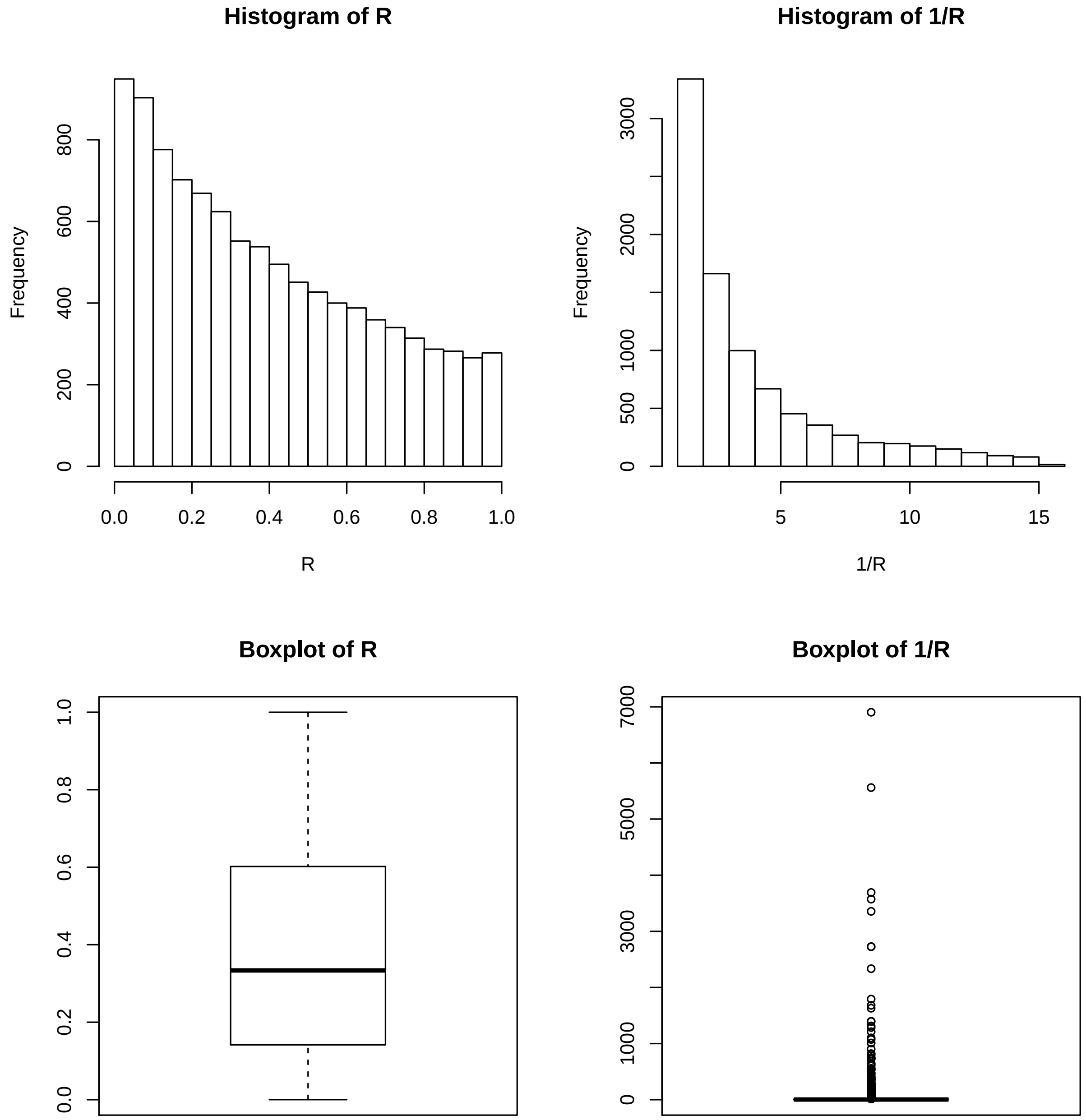
```
## [1] 0.8959733
```

Problem 3a

A stick of length 1 is broken at a uniformly random point, yielding two pieces. Let X and Y be the lengths of the shorter and longer pieces, respectively, and let R = X/Y be the ratio of the lengths X and Y. a.) Use simulations in R (the statistical programming language) to gain some understanding about the distribution of the random variable R. Numerically estimate the expected value of R and 1/R.

```
stickBreakPoints <- runif(10000) # simulate 10,000 stick breaks
# create empty dataframe to hold the X and Y lengths
xyLengths <- setNames(data.frame(matrix(nrow = length(stickBreakPoints), ncol = 2)), c("X","Y"))

# loop to assign X and Y for each breakpoint, based on which is longer
for(i in 1:length(stickBreakPoints)){
  ifelse(stickBreakPoints[i]>=0.5,
         xyLengths$Y[i] <- stickBreakPoints[i],
         xyLengths$X[i] <- stickBreakPoints[i])
}

# update the NAs with 1-(the other value)
xyLengths$X[which(is.na(xyLengths$X))] <- 1-xyLengths$Y[which(is.na(xyLengths$X))]
xyLengths$Y[which(is.na(xyLengths$Y))] <- 1-xyLengths$X[which(is.na(xyLengths$Y))]

# Calculate R and 1/R
R <- xyLengths$X/xyLengths$Y
Rreciprocal <- 1/R
```

```
par(mfrow = c(2,2))
hist(R)
# plot without the outliers to better see the distribution
hist(Rreciprocal[-which(Rreciprocal %in% boxplot.stats(Rreciprocal)$out)],
     main = "Histogram of 1/R",
     xlab = "1/R")
boxplot(R, main = "Boxplot of R")
boxplot(Rreciprocal, main = "Boxplot of 1/R")
```



```r
# calculate E(R) and E(1/R) via their means and show summary stats
summary(R)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
## 0.0001449 0.1415617 0.3336421 0.3866660 0.6018764 0.9999195
```

```r
summary(Rreciprocal)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
##    1.000    1.661    2.997   15.582    7.064 6903.237
```

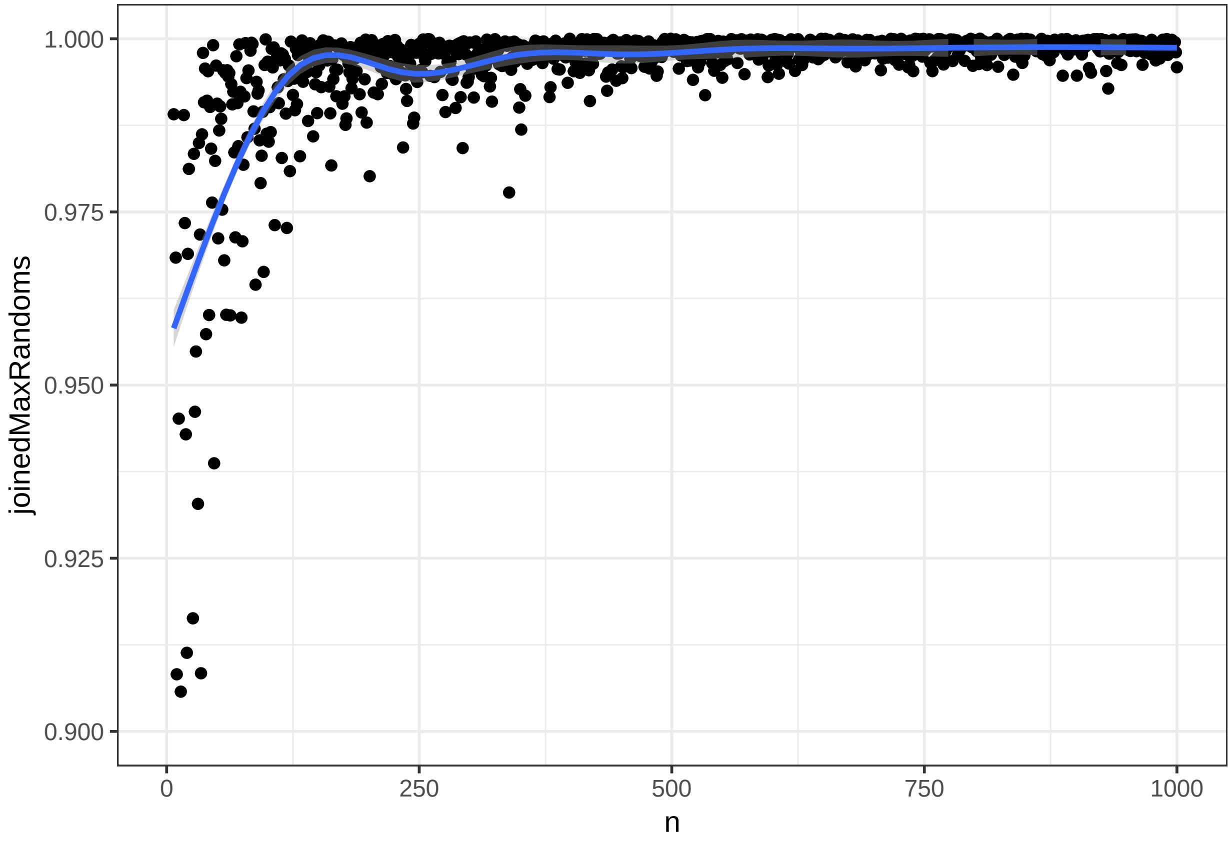
Problem 4c

Use simulations in R to numerically estimate E(X).

```r
# simulations for n = 1 to n = 1000
n <- 1:1000
joinedMaxRandoms <- c()

for(i in 1:length(n)){
  randoms <- runif(i)
  maxRandom <- max(randoms)
  joinedMaxRandoms <- c(joinedMaxRandoms, maxRandom)
}

mean(joinedMaxRandoms) # this is a bit misleading since n varies with this approach
```

```
## [1] 0.9917338
```

```r
# show convergence to n/(n+1) via plot
g <- ggplot(data.frame(joinedMaxRandoms), aes(x = n, y = joinedMaxRandoms))
g + geom_point() +
  geom_smooth() +
  ylim(0.9,1) +
  theme_bw()
```



```
# I can show for a particular n, but I'm hoping this is more interesting?
```

Problem 5b

Use simulations in R (the statistical programming language) to numerically estimate P(X < Y) for X ~ N(0, 1), Y ~ N(1, 5) with X and Y independent.

```
n <- 10000 #sample size

# draw samples for both X and Y
X <- rnorm(n)
Y <- rnorm(n, mean = 1, sd = sqrt(5))

# calculate how many times X < Y
count <- sum(X<Y)
# calculate the probability by dividing the cases where X < Y by n
prob <- count/n
prob
```

```
## [1] 0.6648
```

Problem 6b

Using simulations in R, calculate the Monte Carlo estimates of the mean and standard deviation of the distribution of x ~ y. Compare these estimates with the exact values of the mean and standard deviation.

```
# create empty vars to store the means from the simulations
maleMeans <- c()
femaleMeans <- c()
deltaMeans <- c()

# loop to do simulations
for(i in 1:500){
  x <- rnorm(1:100, 69.1, 2.9)
  y <- rnorm(1:100, 63.7, 2.7)
  maleMeans[i] = mean(x)
  femaleMeans[i] = mean(y)
  deltaMeans[i] = maleMeans[i]-femaleMeans[i]
}

mean(deltaMeans)
```

```
## [1] 5.368901
```

```
sd(deltaMeans)
```

```
## [1] 0.3985999
```

Problem 6c

What is the probability that a randomly sampled man is taller than a randomly sampled woman? Please do not answer this question by reporting a Monte Carlo estimate of this probability.

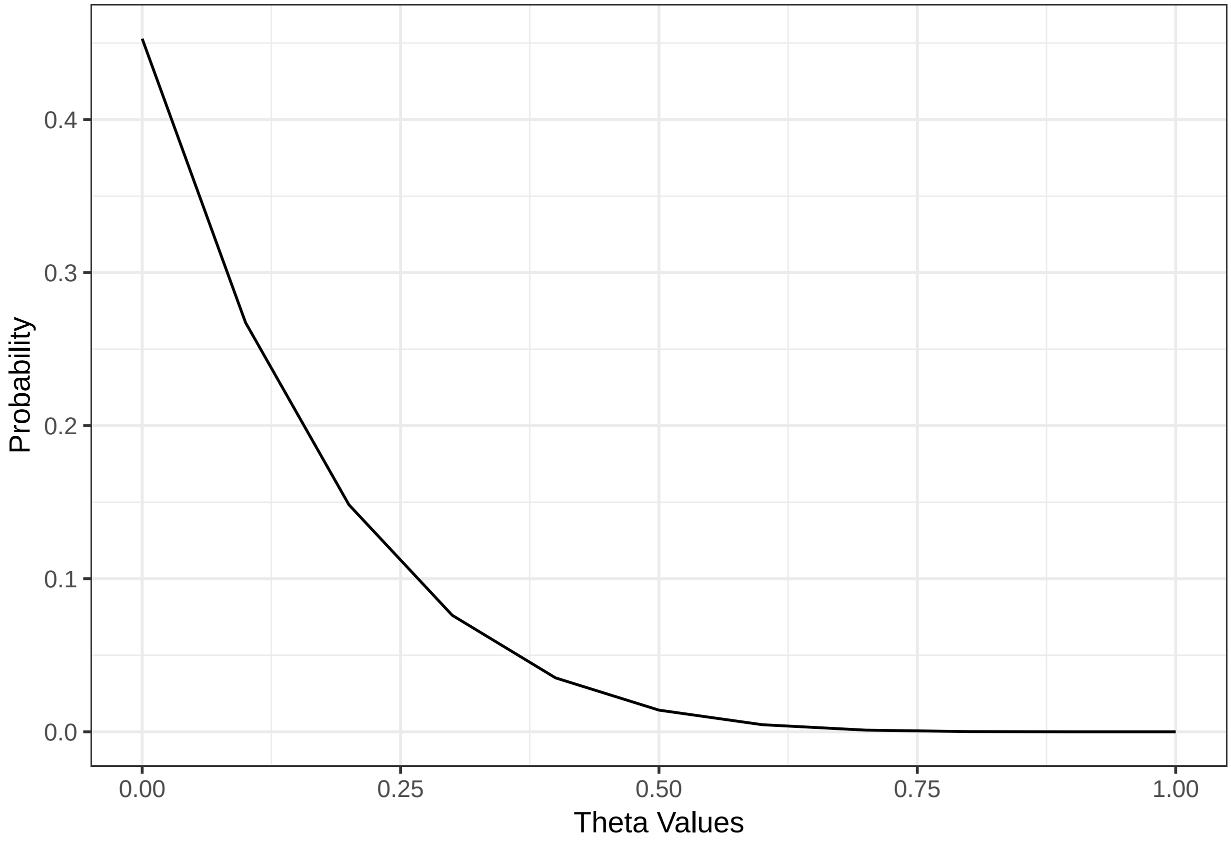
```
# calculate the exact probability
dist <- (69.1-63.7)/sqrt(2.9^2 + 2.7^2)
pnorm(dist, 0, 1)
```

```
## [1] 0.9135331
```

Problem 7b

For y = 0, make a plot for P(?? | y) for each ?? ??? ?? = {0.0, 0.1, . . . , 0.9, 1.0}. In other words, make a plot with the horizontal axis representing the 11 values of ?? and the vertical axis representing the corresponding values of P(?? | y).

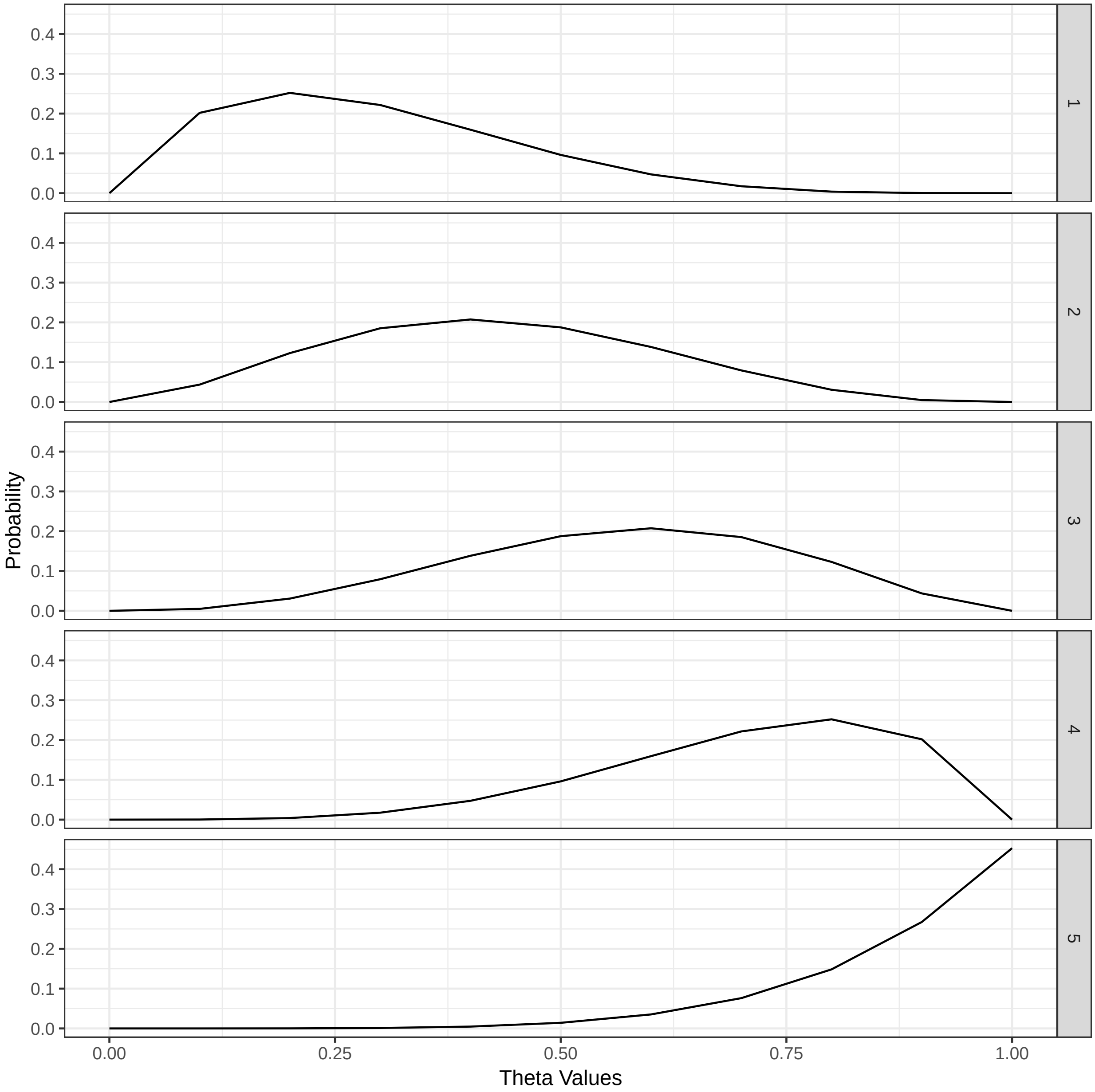
```
# vector to store theta values
thetaVals <- seq(0.0, 1.0, by = 0.1)

# function to evaluate probability of theta given Y
probabilizer <- function(Y) {
  numerators <- c()

  # use equation from 7a
  for(i in 1:length(thetaVals)) {
    numerators <- c(numerators, ((thetaVals[i]^Y)*(1 - thetaVals[i])^(5-Y)))
  }

  denominator <- sum(numerators)

  probs <- c()
  for(i in 1:length(thetaVals)) {
    probs <- c(probs, (numerators[i] / denominator))
  }
  return(probs)
}

# calculate the probs for y = 0
yCoords <- probabilizer(0)

# convert to data.frame for ggplot
plotData <- data.frame(x = thetaVals, y = yCoords)

g <- ggplot(plotData, aes(x = x, y = y))
g + geom_line() +
  theme_bw() +
  labs(x = "Theta Values", y = "Probability")
```



Problem 7c

Repeat (b) for each y ??? {1, 2, 3, 4, 5}, so in the end you have six plots (including the one in (b)). Describe what you see in your plots and discuss whether or not they make sense.

```
# build a dataframe that can be used with facet_wrap() by making a tall dataset
# this is terrible code, but I'm short on time so beauty is out the window!!
plotData <- data.frame(x = rep(thetaVals, 5),
                       y = c(probabilizer(1),
                             probabilizer(2),
                             probabilizer(3),
                             probabilizer(4),
                             probabilizer(5)),
                       givenY = as.character(c(rep(1,11),rep(2,11),rep(3,11),rep(4,11),rep(5,11))))

g <- ggplot(plotData, aes(x = x, y = y))
g + geom_line() +
  theme_bw() +
  labs(x = "Theta Values", y = "Probability") +
  facet_grid(givenY ~ .)
```



Problem 8b

Implement your sampling algorithm in R, and use your code to produce a Monte Carlo estimate of P(X ??? (2, 3)) where X is a random variable that has a Logistic distribution.

```
n <- 10000 #simulations
x <- runif(n, 0, 1) # build vector of random uniforms
y <- -log((1/x)-1) # equation from 8a

successes <- sum(2<y & y<3)
prob <- successes/n
prob
```

```
## [1] 0.0699
```