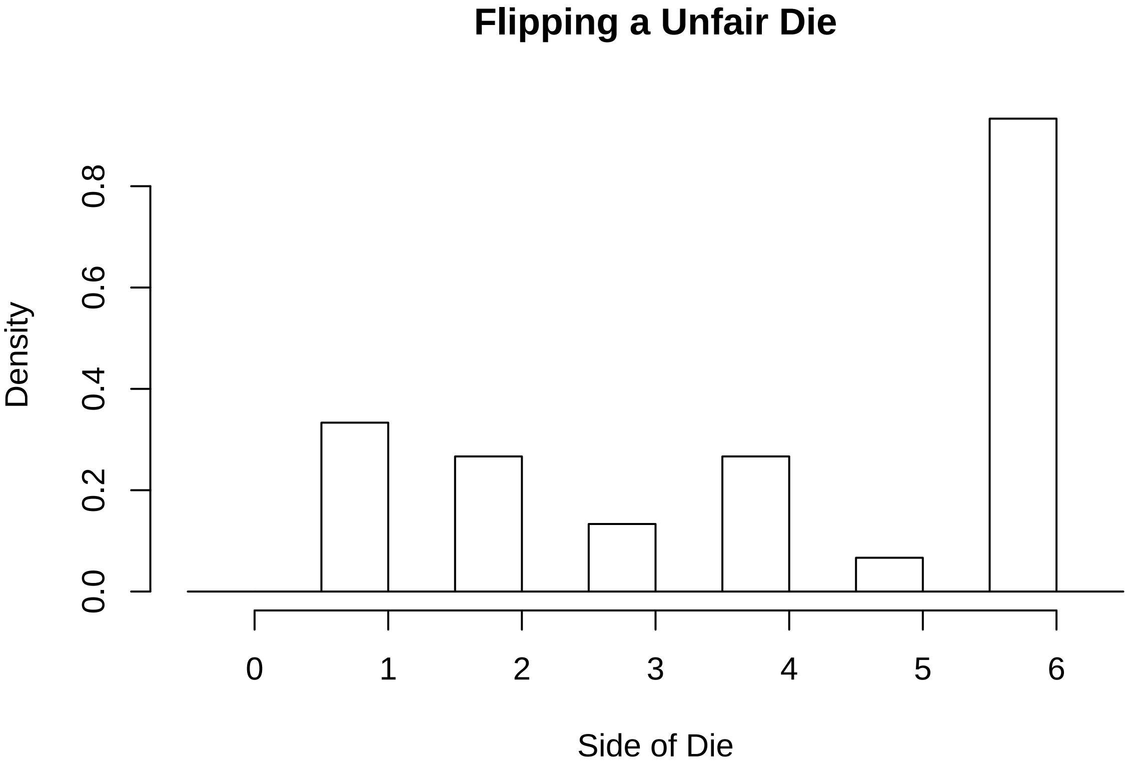
IS457_HW6_127

Part 1: Unfair Dice Simulation

A die is not necessarily fair, in which case the probabilities for 6 sides are different. We will look at a way to simulate unfair dice rolls in R

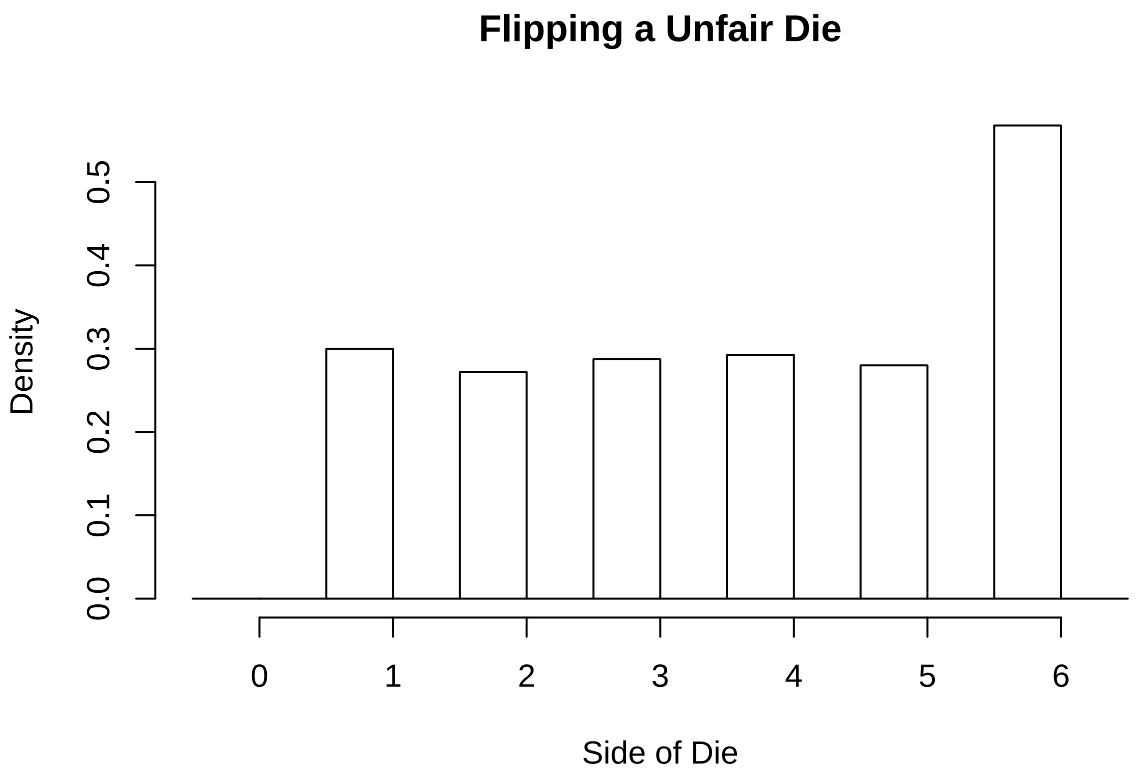
(1) Draw independently from a 6-side die with probability 2/7 for a six and 1/7 for others 30 times, and save your result in a vector called roll1, make a histogram for the empirical density. (2 pt)

```r
set.seed(457)
proba<-c(rep(1/7,5),2/7)
proba<-as.vector(proba,mode = "any")
roll1<-sample(6,size = 30,prob = proba,replace = T)
hist(roll1,breaks=c(seq(-0.5,6.5,by =0.5)),freq=F,
     main = "Flipping a Unfair Die",
     xlab = "Side of Die")
```



(2) Now, draw independently from a 6-side die with probability 2/7 for a six and 1/7 for others 3000 times, and save your result in vector roll2, make a histogram for the empirical density. (2 pt)

```r
set.seed(457)
roll2<-sample(1:6,size = 3000,prob = proba,replace = T)
hist(roll2,breaks=c(seq(-0.5,6.5,by =0.5)),freq=F,
     main = "Flipping a Unfair Die",
     xlab = "Side of Die")
```



(3) What do you conclude from comparing these two plots? (2 pts)

More simulations we run, the empirical density we get closer to the true density.

Part 2: Monte Carlo Simulation

We will use the simulation techniques (Monte Carlo) introduced in class to generate confidence intervals for our estimates of the distribution mean.

(1) As we will generate random numbers, to ensure reproducibility, please set the seed as 457.(1 pt) NOTE: make sure you run the seed command EVERY time you sample something

```
set.seed(457)
```

(2) For this simulation problem, we will sample data from the binomial distribution with parameters n and p.

First, we will estimate an individual experiment.

(a) Generate 100 observations of test data from the binomial distribution, with 20 trials and 0.8 probability and name it test_sample. (1 pt)

```
set.seed(457)
test_sample<-rbinom(100,20,0.8)
```

(b) What is your estimate of the mean for the test data? call your estimate X_hat. What is the exact mean (use the formula to calculate mean for a binomial dist)? are they close? what does this say about our random generation?(4 pts)

```
X_hat<-mean(test_sample)
True_mean<-20*0.8
X_hat
```

```
## [1] 16.24
```

```
True_mean
```

```
## [1] 16
```

Yes, they are close to each other. We can say that our random generation makes sense, it provides a good simulation to the reality.

(c) What is the 95% confidence interval for X_hat? (2 pts)

```
sd<-sd(test_sample)
confidence_interval<-c(X_hat-(1.96*sd)/sqrt(length(test_sample)),
                       X_hat+(1.96*sd)/sqrt(length(test_sample)))
confidence_interval
```

```
## [1] 15.89528 16.58472
```

(3) Now use simulation technique to estimate the distribution of X_hat and create confidence intervals for it.

(a) Form a set of X_hat's by repeating B = 1000 times the individual experiment. (2 pts) HINT: You may want to create a matrix to save those values.

```
set.seed(457)
B <- 1000
sample<-matrix(nrow = 100,ncol = 1000)
for (i in 1:1000){
    sample[,i]<-as.vector(rbinom(100,20,0.8))
}
```

100行；1000列；
一次100个，来1000次

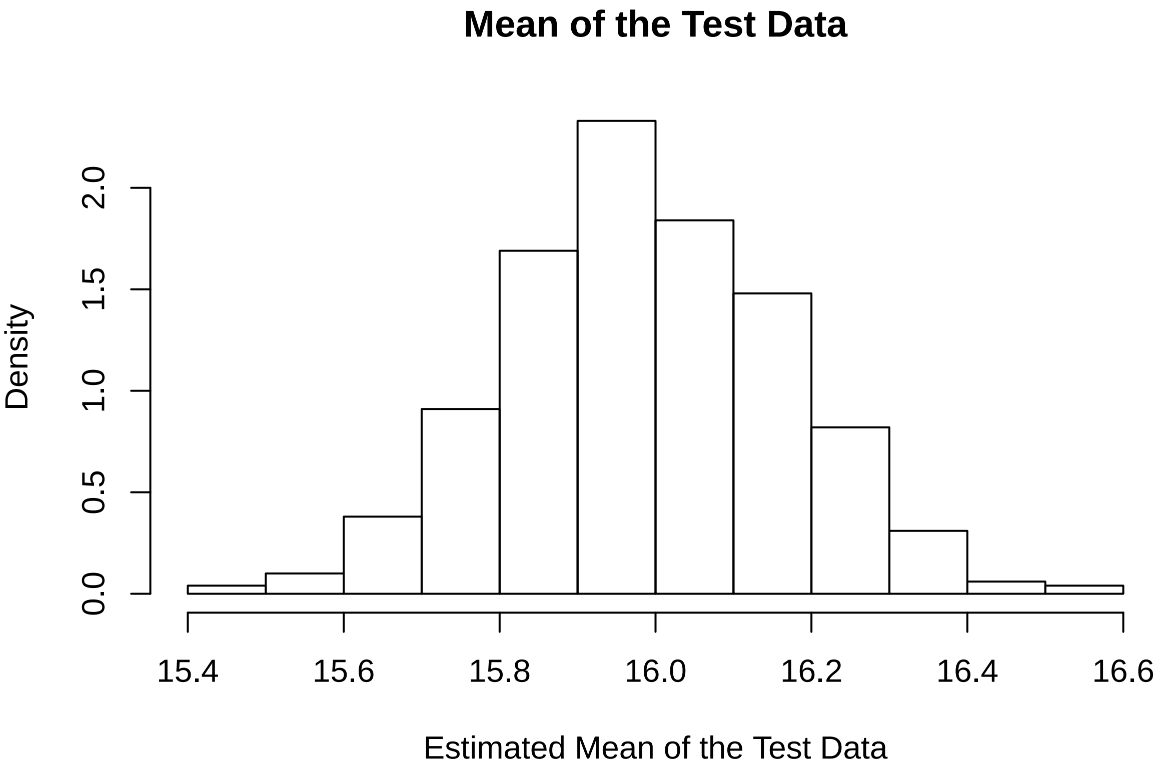
(b) Get an estimate for the mean of the X_hat's for each experiment in (3)(a) and save it to a vector X_hat_estimate (length B vector).(1 pt)

```
X_hat_estimate<-apply(sample, 2, mean)
head(X_hat_estimate)
```

```
## [1] 16.24 16.01 15.88 15.88 16.10 16.25
```

(c) Now use X_hat_estimate to create a "sampling distribution" for X_hat, and create a histogram to show the distribution. Does the distribution look normal (what are the essential elements of normal dist)? how can you tell? if yes, what does it say about our random generation? (4 pts)

```
hist(X_hat_estimate, freq=F,
     main = "Mean of the Test Data",
     xlab = "Estimated Mean of the Test Data")
```



As we can see from above, the histogram of `X_hat_estimate` is symmetric and bell-shaped, so that it looks like follow normal distribution. Besides, the histogram centers around the true mean so that our random generation provides a good simulation to eastimate the mean of a test data which follows binomial distribution.

(d) Now as we have a simulated sampling distribution of X_hat, we could empirically calculate the standard error using the X_hat_estimate. What is your 95% confidence interval?(2 pts) Notice here the standard error is indeed the standard deviation

```
se<-sd(X_hat_estimate)
mean<-mean(X_hat_estimate)
confidence_interval_x_hat<-c(mean-1.96*se,
                             mean+1.96*se)
confidence_interval_x_hat
```

```
## [1] 15.64166 16.34426
```

(4) We made some decisions when we used the simulation above that we can now question.

Repeat the above creation of a confidence interval in (3) for a range of settings (we had our sample size fixed at 100) and a range of B values (we had B fixed at 1000). Suppose the sample size varies (100, 200, 300, ... , 1000) and B varies (1000, 2000, . . . , 10000). You will likely find it useful to write functions to carry out these calculations. Your final output should be upper and lower pairs for the confidence intervals produced using the bootstrap method for each value of sample size and B.

(a) Generalize (3) into a function, and vary inputs of sample size and B as we did above. (5 pts)

```
X_hat_Estimate_CI_mean<-function(sample_size, B){
  X_hat_estimate1<-vector(length = length(B))
  se1<-vector(length = length(B))
  X_hat_estimate1_mean<-vector(length = length(B))
  X_Hat_Estimate_CI<-list()
  for(j in 1:length(B)){
 sample<-matrix(nrow = sample_size[j],ncol = B[j])
  for (i in 1:B[j]){
    sample[,i]<-as.vector(rbinom(sample_size[j],20,0.8))
  }
  X_hat_estimate1<- apply(sample, 2, mean)
  se1<-sd(X_hat_estimate1)
  X_hat_estimate1_mean[j]<-mean(X_hat_estimate1)
  X_Hat_Estimate_CI[[j]]<-c(X_hat_estimate1_mean[j]-1.96*se1,
                            X_hat_estimate1_mean[j]+1.96*se1)
  }
  c( CI=list(X_Hat_Estimate_CI),mean=list(X_hat_estimate1_mean))
}

B<-c(seq(1000,10000,by=1000))
sample_size<-c(seq(100,1000,by=100))
set.seed(457)
X_hat_Estimate_CI_means<-X_hat_Estimate_CI_mean(sample_size,B)
X_hat_Estimate_CI_means
```

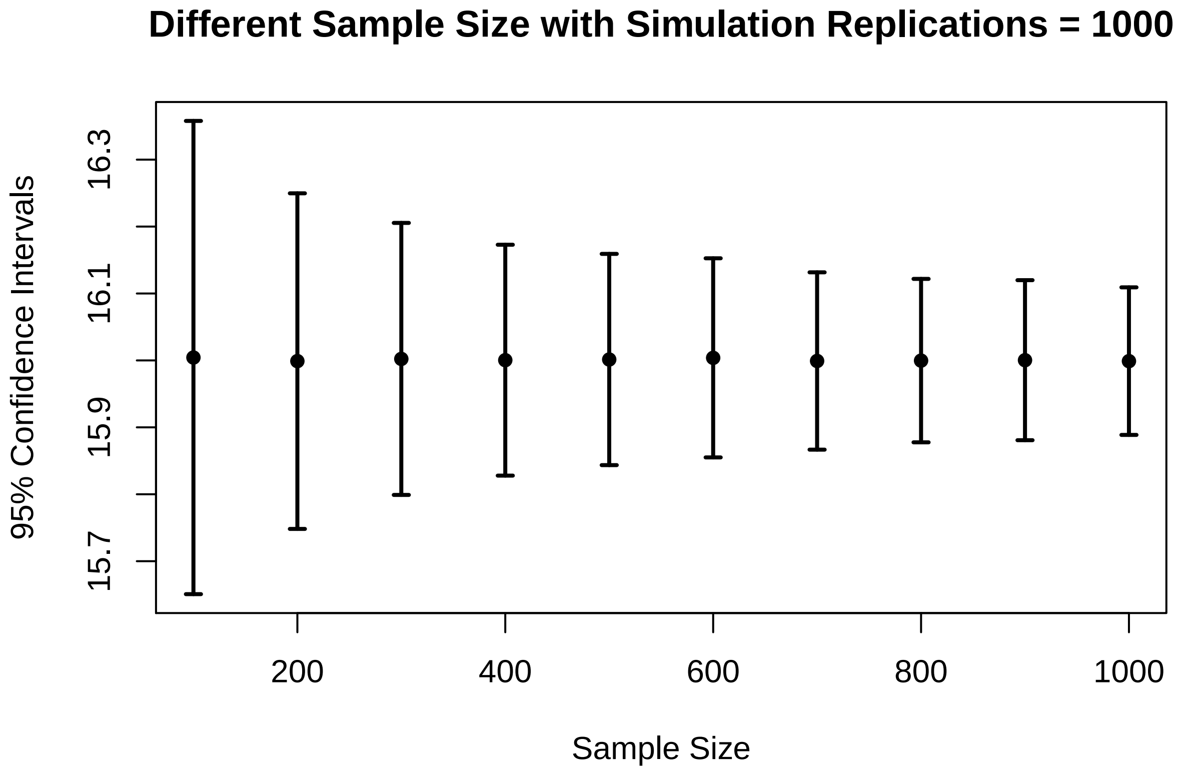
```
## $CI
## $CI[[1]]
## [1] 15.64166 16.34426
##
## $CI[[2]]
## [1] 15.74781 16.25046
##
## $CI[[3]]
## [1] 15.80531 16.19833
##
## $CI[[4]]
## [1] 15.82997 16.17334
##
## $CI[[5]]
## [1] 15.83946 16.15660
##
## $CI[[6]]
## [1] 15.85767 16.14538
##
## $CI[[7]]
## [1] 15.86813 16.13113
##
## $CI[[8]]
## [1] 15.87874 16.12227
##
## $CI[[9]]
## [1] 15.88301 16.11579
##
## $CI[[10]]
## [1] 15.88882 16.11207
##
##
## $mean
##  [1] 15.99296 15.99914 16.00182 16.00165 15.99803 16.00153 15.99963
##  [8] 16.00050 15.99940 16.00045
```

(5) Use the function errbar() in Hmisc package.

Plot your confidence interval limits to compare the effect of changing the sample size and changing the number of simulation replications B (10 pts). What do you conclude? (4 pts)

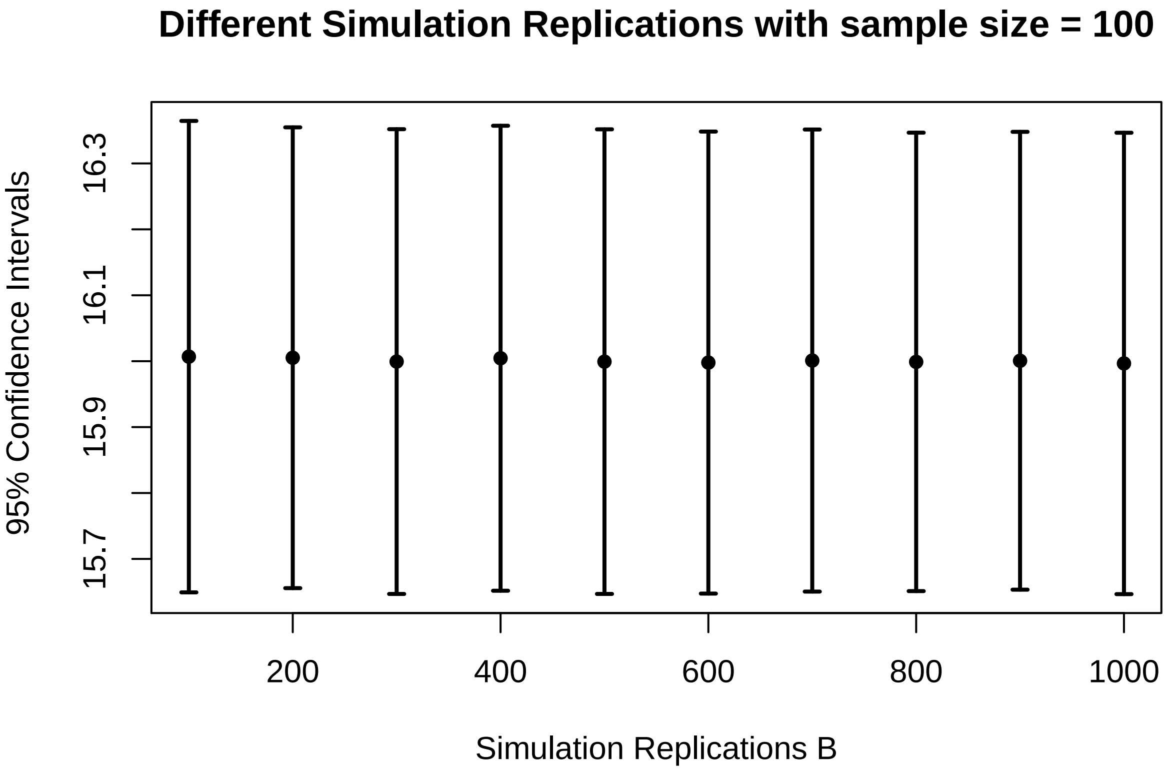
```r
#Effect of changing the sample size
B1<-c(rep(1000,10))
sample_size<-c(seq(100,1000,by=100))
X_hat_Estimate_CI_means1<-X_hat_Estimate_CI_mean(sample_size,B1)
CI1<-X_hat_Estimate_CI_means1$CI
CI1_lower<-vector(length = length(B1))
for (i in 1:length(CI1))(
  CI1_lower[i]<-CI1[[i]][1]
)
CI1_higher<-vector(length = length(B1))
for (i in 1:length(CI1))(
  CI1_higher[i]<-CI1[[i]][2]
)
means1<-X_hat_Estimate_CI_means1$mean

library(Hmisc)
errbar(x=c(seq(100,1000,by=100)),
       yplus = CI1_higher,
       yminus = CI1_lower,
       y=means1,
       xlab = "Sample Size",
       ylab = "95% Confidence Intervals",
       lwd=2)
title(main = "Different Sample Size with Simulation Replications = 1000")
```



```
#Effect of changing the number of simulation replications B
B1<-c(seq(1000,10000,by=1000))
sample_size<-c(rep(100,10))
X_hat_Estimate_CI_means1<-X_hat_Estimate_CI_mean(sample_size,B1)
CI1<-X_hat_Estimate_CI_means1$CI
CI1_lower<-vector(length = length(B1))
for (i in 1:length(CI1))(
  CI1_lower[i]<-CI1[[i]][1]
)
CI1_higher<-vector(length = length(B1))
for (i in 1:length(CI1))(
  CI1_higher[i]<-CI1[[i]][2]
)
means1<-X_hat_Estimate_CI_means1$mean

library(Hmisc)
errbar(x=c(seq(100,1000,by=100)),
       yplus = CI1_higher,
       yminus = CI1_lower,
       y=means1,
       xlab = "Simulation Replications B",
       ylab = " 95% Confidence Intervals",
       lwd=2)
title(main = "Different Simulation Replications with sample size = 100")
```



According to the result above, sample size have influence on confidence intervals, since as the sample size increases, the confidence intervals shrink towards the true value(center). And changing the number of simulation replications B seems not affect confidence intervals.