Lab 9

Sharleen Price spp2122

3/28/2018

Instructions

Make sure that you upload an RMarkdown file to the canvas page (this should have a .Rmd extension) as well as the PDF or HTML output after you have knitted the file. Note that since you have already knitted this file, you should see both a **Lab9_UNI.pdf** and a **Lab9_UNI.Rmd** file in your UN2102 folder. Click on the **Files** tab to the right to see this. The files you upload to the Canvas page should be updated with commands you provide to answer each of the questions below. You can edit this file directly to produce your final solutions. The lab is due 11:59pm on Wednesday, April 3rd.

Simulating Random Variables

The goal of this lab is to simulate a few different random variables (discrete and continuous) using the standard uniform distribution. Recall that the standard uniform distribution has probability density function

$$f(x) = 1, \quad 0 \leq x \leq 1.$$

The **R** functions used for the uniform distribution are respectability **dunif**, **punif**, **qunif**, and **runif**. In this lab, we will not use the base **R** functions for simulation, instead, students will create a Linear Congruential Generator function, similar to the LCG presented in the slides.

Perform the Following Tasks:

1) First let's study the modular arithmetic operator **%%**. For a full explanation, look up modular arithmetic on Wikipedia. In this lab, all that you need to understand is that $a \mod n$ is the remainder of the Euclidean division of a by n. To convince yourself, run the following code:

```
0 %% 3
```

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

```
1 %% 3
```

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

```
2 %% 3
```

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

```
3 %% 3
```

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

```
4 %% 3
```

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

```
5 %% 3
```

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

```r
6 %% 3
```

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

Notice that the sequence begins to repeat itself.

Uniform over [0,1]

2) Consider the **new.random** function defined in class. Define a **R** function called **my.runif** that uses the code below to simulate **n** random uniform variables over the unit interval. The **my.runif** function should have inputs **n**, **a**, **c** and **m**. Set the default values respectively to **n=1**, **a=1664545**, **c=1013904223**, and **m=2^32**. Set the seed to 10 in the global environment (not using the **set.seed** function). Once you created the function, simulate **n=20** standard uniform random draws.

```r
seed <- 10
new.random <- function(a = 1664545, c = 1013904223,  m = 2^32)
{
  out <- (a*seed + c) %% m
  seed <<- out
  return(out)
}

out.length <- 20
variants   <- rep(NA, out.length)
for (i in 1:out.length) {
  variants[i] <- new.random()
}
variants/2^(32)
```

```
##  [1] 0.23994354 0.06198592 0.58802936 0.56118427 0.71271264 0.49615883
##  [7] 0.93866800 0.35949191 0.70568234 0.25497190 0.43584239 0.51478071
## [13] 0.90072217 0.81668135 0.08857049 0.80120534 0.57149849 0.18884028
## [19] 0.37266710 0.39721732
```

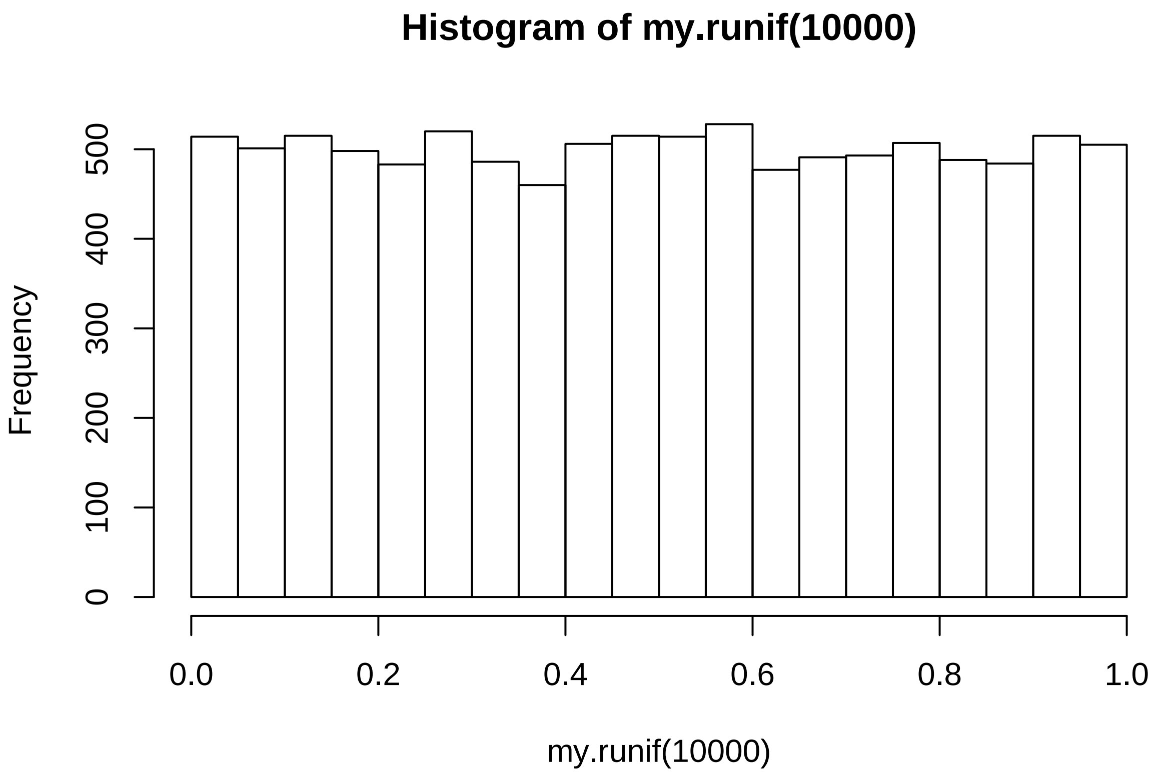
Solution

```r
seed <- 10
my.runif <- function(n = 1, a = 1664545, c = 1013904223, m = 2^32)
{

# fill in body of function
variants   <- rep(NA, n)
for (i in 1:n)
  {
  variants[i] <- new.random(a=a,c=c,m=m)
  }
return (variants/m)

}

my.runif(n=20)
```

```
##  [1] 0.23994354 0.06198592 0.58802936 0.56118427 0.71271264 0.49615883
##  [7] 0.93866800 0.35949191 0.70568234 0.25497190 0.43584239 0.51478071
## [13] 0.90072217 0.81668135 0.08857049 0.80120534 0.57149849 0.18884028
## [19] 0.37266710 0.39721732
```

3) Using **my.runif**, create a histogram of 10,000 simulated uniform draws over the unit interval. Set the breaks to 20.

```
# solution
hist(my.runif(10000), breaks = 20)
```



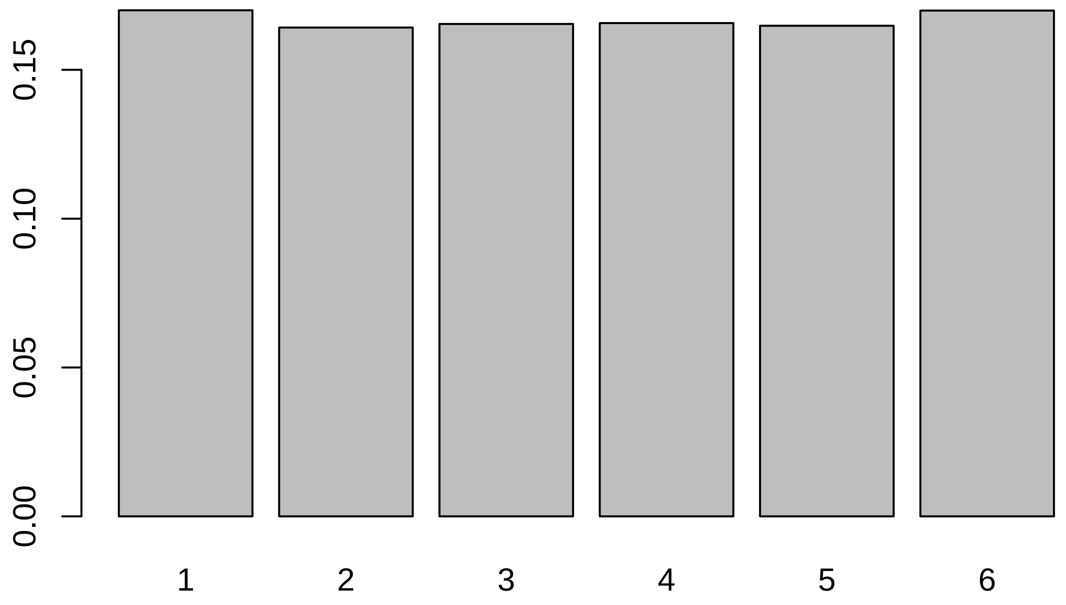
Rolling Dice

/ 4) Consider rolling a single fair six-sided die and recording its displayed face. Define this random variable as D_1. Using **my.runif**, simulate 10,000 draws of D_1. Display the empirical distribution as both a table and bargraph.

```
# solution
D_1<-ceiling(6*my.runif(10000))
table(D_1)/10000
```

```
## D_1
##      1      2      3      4      5      6
## 0.1700 0.1642 0.1654 0.1657 0.1648 0.1699
```

```
barplot(height=table(D_1)/10000)
```



5) Consider rolling two fair six-sided dice D_1, D_2 and recording the maximum of the two faces. Define the random variable

$$Y = \max\{D_1, D_2\}.$$

Compute the true probability mass function of the random variable Y. To complete this exercise, fill out the table displayed below:

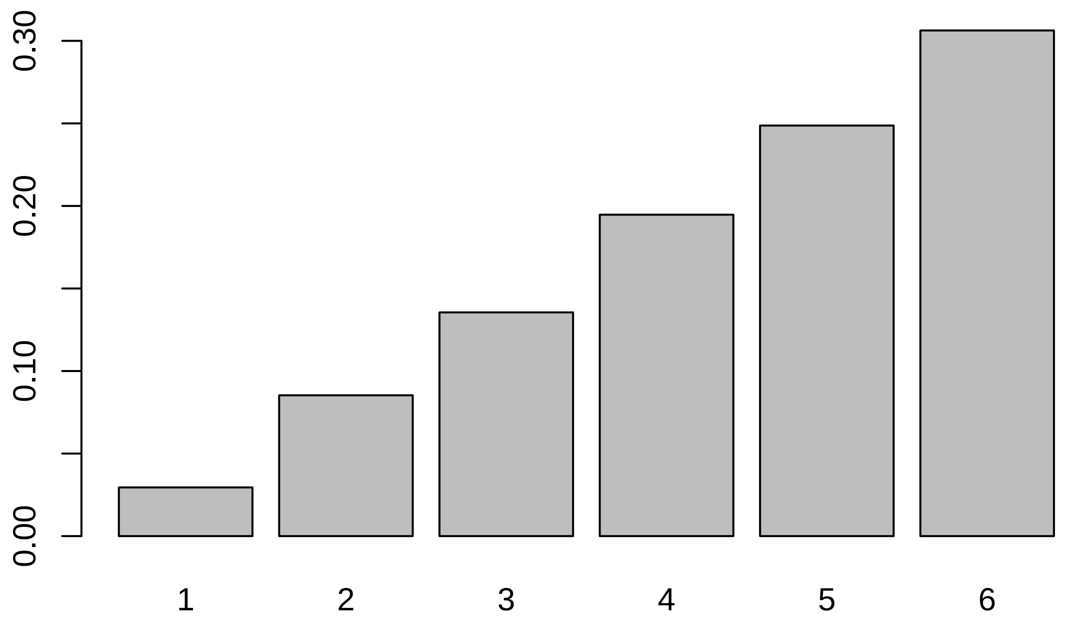
$Y = y$	1	2	3	4	5	6
$P(Y = y)$	1/36	3/36	5/36	7/36	9/36	11/36

6) Now using the function **my.runif**, simulate 10,000 draws of the random variable Y. Store the draws in the vector **Y**, display the empirical probability mass function of the simulated values, and display a bargraph of empirical distribution.

```r
# solution
D_1<-ceiling(6*my.runif(10000))
D_2<-ceiling(6*my.runif(10000))
y=c()
for (i in 1:10000){
  if (D_1[i]>D_2[i]){
    y<-append(y,D_1[i])
  }
  else if (D_1[i]<D_2[i]){
    y<-append(y,D_2[i])
  }
  else {
    y<-append(y,D_1[i]) #arbitrary if they are equal
  }
}
table(y)/10000
```

```
## y
##      1      2      3      4      5      6
## 0.0295 0.0853 0.1355 0.1947 0.2487 0.3063
```

```r
barplot(table(y)/10000)
```

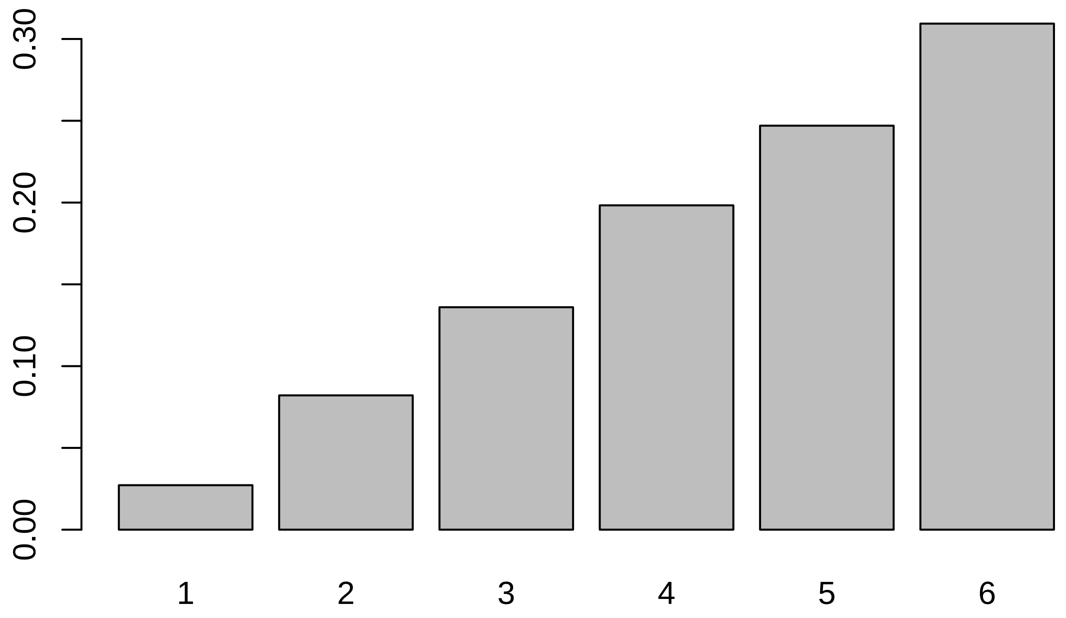


7) Perform the same task from Part (6) using the **sample** function.

```r
# solution
die1<-(sample(1:6,10000,replace=TRUE))
die2<-(sample(1:6,10000,replace=TRUE))
y=c()
for (i in 1:10000){
  if (die1[i]>die2[i]){
    y<-append(y,die1[i])
  }
  else if (die1[i]<die2[i]){
    y<-append(y,die2[i])
  }
  else {
    y<-append(y,die1[i]) #arbitrary if they are equal
  }
}
table(y)/10000
```

```
## y
##      1      2      3      4      5      6
## 0.0272 0.0821 0.1360 0.1983 0.2470 0.3094
```

```r
barplot(table(y)/10000)
```



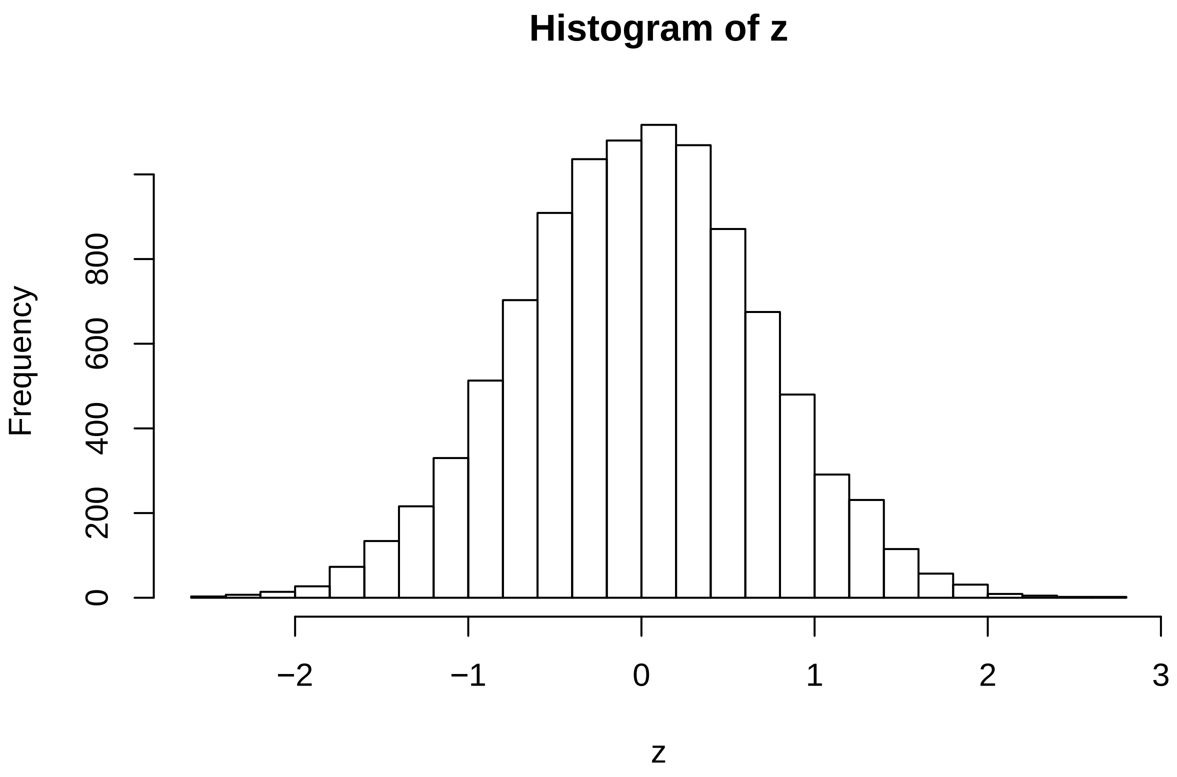
More Simulation

8) Consider two independent uniform random variables U_1 and U_2, each over the unit interval $[0, 1]$. Use the function **my.runif** to simulate 10,000 draws from U_1 and 10,000 draws from U_2. Then once the draws are simulated, construct a new random variable Z using the relation

$$Z = \cos(2\pi U_1)\sqrt{-2\log(U_2)}.$$

Create 10,000 draws of Z from U_1 and U_2 and construct a histogram of the empirical distribution. What does this distribution look like?

```
# solution
U.1 <- my.runif(n=10000)
U.2 <- my.runif(n=10000)
z <- cos(2*pi*U.1)*sqrt(-log(U.2))
hist(z, breaks = 30)
```



9) Compute the proportion of simulated Z values that fall between -1.96 and 1.96.

```
# solution
count = 0
for (i in 1:length(z)){
  if(z[i]>-1.96 & z[i]<1.96){
    count = count +1
  }
}
print(count/10000)
```

```
## [1] 0.9948
```