Approximating Pi with Monte Carlo Simulation

Boyie Chen

10/25/2019

Monte Carlo Simulation

simulate the dice-roll game

The expected value of the number come up after rolling a faire dice is:`sum(1:6*(1/6))`, which is 3.5 What we consider is how come the theoretical mean is 3.5?

```
sum(1:6*(1/6)) # 理論值 -> 但實際值不總是理論值 -> 如何模擬出來？
```

```
## [1] 3.5
```

Let's see some randomly result of rolling a dice n times.

```
#roll a fair dice n times
sample(1:6, size = 1, replace = T)
```

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

```
sample(1:6, size = 10, replace = T)
```

```
##  [1] 2 2 4 3 3 1 2 1 6 4
```

```
mean(sample(1:6, size = 10, replace = T))
```

```
## [1] 3.3
```

```
mean(sample(1:6, size = 1000, replace = T))
```

```
## [1] 3.528
```

```
mean(sample(1:6, size = 100000, replace = T)) #getting closer and closer to 3.5
```

```
## [1] 3.50059
```

```
mean(replicate(100000, sample(1:6, size = 1, replace = T)))
```

```
## [1] 3.51122
```

```
#note that this is an equivalent expression, but minor difference in performance
```

Under what situation that we can use simulation?

Some values are stationary if we try to repeat the process of getting that value for nearly infinity times. We have not yet talked about the theory behind it, let's just take this kind of phenomenon as granted.

Approximating Pi

If we want to know π... Recall the definition of π: the ratio of a circle's circumference to its diameter We can derive π from some crazing things such as Polygon approximation, or some infinite series or fractions... These are indeed the true formula with infinite terms for calculating π. Given all the truths above, we get our formula for areas of circle: πr^2

Let's mess up this formula with probability… Say we have a square and a quarter circle inside it. Ref: https://helloacm.com/r-programming-tutorial-how-to-compute-pi-using-monte-carlo-in-r/

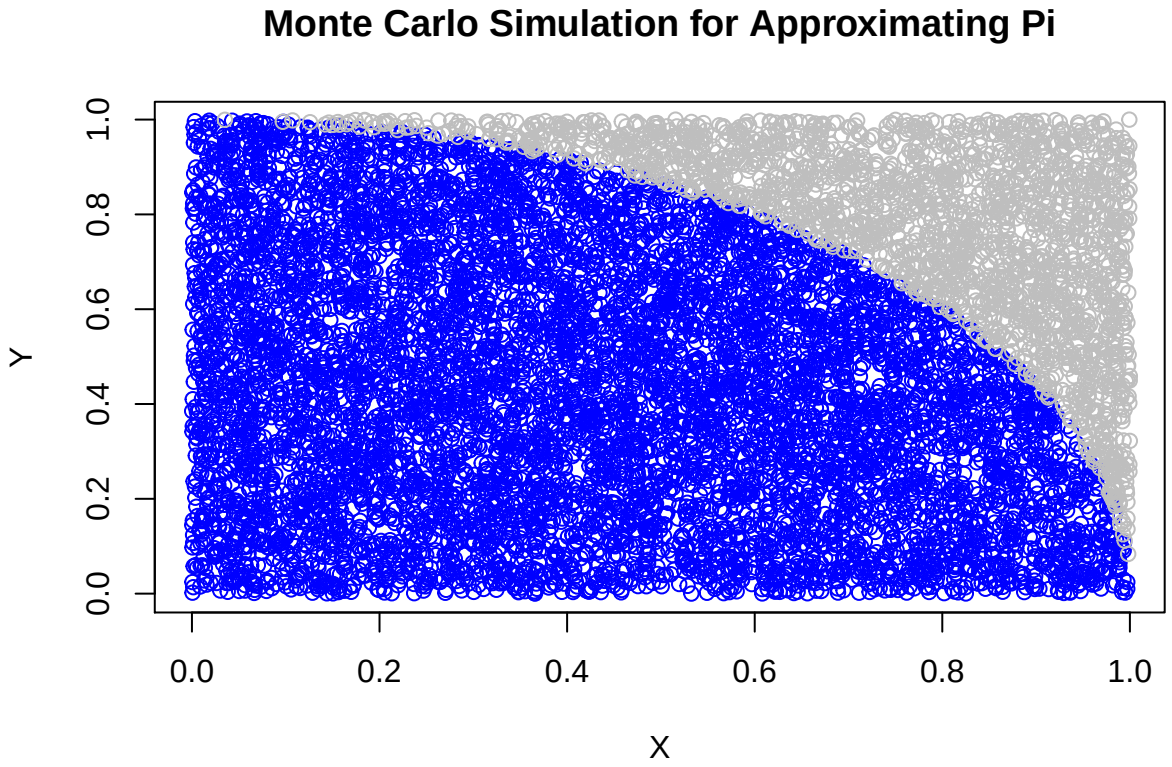
```
# both x and y contains now 100000 random numbers
x=runif(10000) #We just randomly distribute the points inside the square on X-Y plane
y=runif(10000)
z=sqrt(x^2+y^2)
# The following returns the indices vector when z is less or equal than 1
head(which(z<=1))
```

```
## [1] 1 2 3 4 6 8
```

```
length(which(z<=1))*4/length(z) #appoximation of pi
```

```
## [1] 3.1356
```

```
plot(x[which(z<=1)],y[which(z<=1)],xlab="X",ylab="Y",main="Monte Carlo Simulation for Approximating Pi"
points(x[which(z>1)],y[which(z>1)],col='gray')
```



The above is the case of 1/4 circle. We can draw a whole circle, and the mechanism is the same.

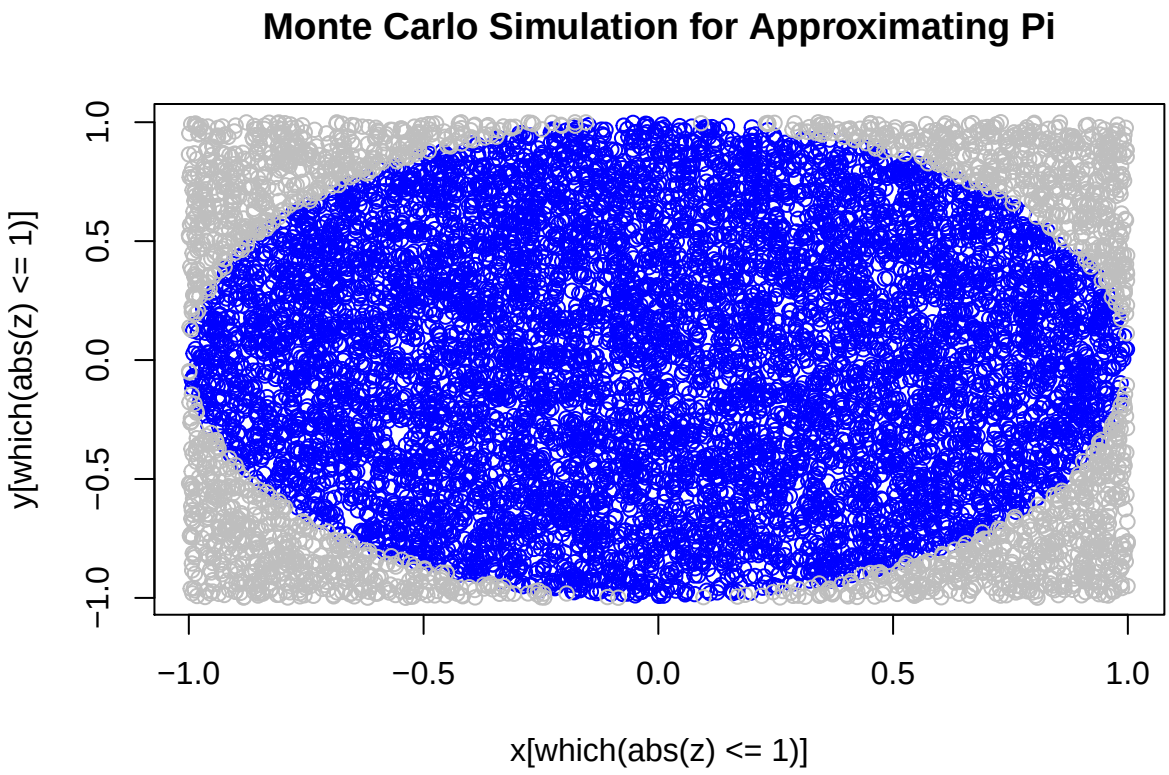
```
#Same thing for the whole circle
x = runif(10000, -1, 1)
y = runif(10000, -1, 1)
z=sqrt(x^2+y^2)
head(abs(z) <= 1)
```

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

```
length(which(abs(z) <= 1))*4/length(z) #appoximation of pi
```

```
## [1] 3.1216
```

```r
plot(x[which(abs(z)<=1)], y[which(abs(z)<=1)], main = 'Monte Carlo Simulation for Approximating Pi', ty
points(x[which(abs(z)>1)], y[which(abs(z)>1)], col = 'gray')
```



Further Example of simulation

In Class Practices (Are Left in Optional Lecture Videos)

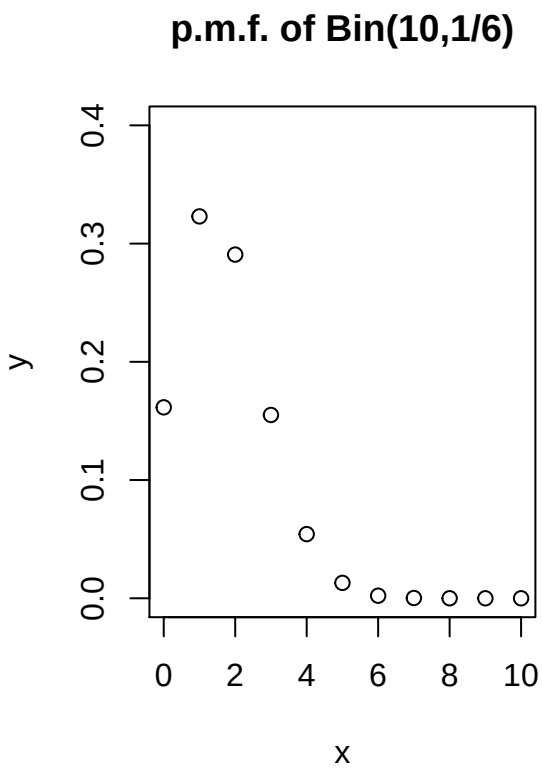
Back to rolling a dice: define a function that give us the number after rolling a n sided dice once

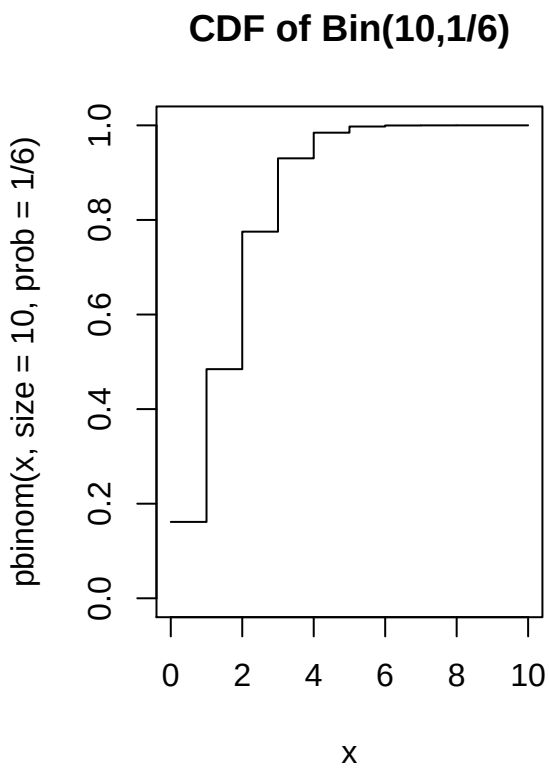
```r
dice = function(sides = numeric(), times = numeric()){
  sample(1:sides, size = times, replace = T)
}
```

令 X 為一隨機變數，表示擲一枚公正骰子十次且點數為 5 的次數 * How does X distribute? * What is E(X)?

In L1, we know that $X \sim Bin(n = 10, p = 1/6)$, so we can response the above question theoretically. $E(X) = np = 10/6 \approx 1.6667$

```r
x = 0:10 # 可能沒有出現半個 5，也可能 10 次都是 5
y = dbinom(x, size = 10, prob = 1/6) #pmf of binomial
par(mfrow = c(1,2))
plot(x, y, main = 'p.m.f. of Bin(10,1/6)', ylim = c(0,.4)) # 只有一次是 5 的機率最高
plot(x, pbinom(x, size = 10, prob = 1/6), type = 's', main = 'CDF of Bin(10,1/6)', ylim = c(0,1))
```






```
#E(X) = n*p = 10/6 = 1.6667
# 也就是平均而言，每擲十次骰子，會有 1.6667 次出現 5
```

If we don't know the theoretical mean...Use simulation! 那我就擲好幾組十次骰子，看看每組出現 5 的次數是多少，有幾組就有幾個次數

```
dice_num = dice(6,10)
head(dice_num == 5)
```

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

```
sum(dice_num == 5) # 得到一個「次數」
```

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

```
# 重複以上三行很多次，以得到進似的「期望次數」
```

重複一千次擲十次骰子的試驗，所以有 1000 組數據，每組數據有 10 個數字

```
mat = replicate(n = 1000, dice(6,10)) #a matrix
count = c() #create a vector that save the times of number that equals to 5
for(i in 1:1000){
  count = rbind(count, sum(mat[,i] == 5))
}
mean(count) # 給了我平均而言，擲 10 次骰子會有這樣多次出現 5
```

```
## [1] 1.691
```

Another Example: If I don't know what kind of the distribution it follows

令 X 為一隨機變數，表示擲一枚公正骰子一次的點數 * How does X distribute? * What is E(X)? Does $X \sim Binomial$ anymore? No! If we don't know the theoretical mean...Use simulation! 那我就擲好幾次骰子，看看每次出現的點數是多少

```r
dice_num = dice(6,1)
dice_num # 重複很多次就可以告訴我 E(X)：平均而言，擲 1 次骰子會出現 `dice_num`點
```

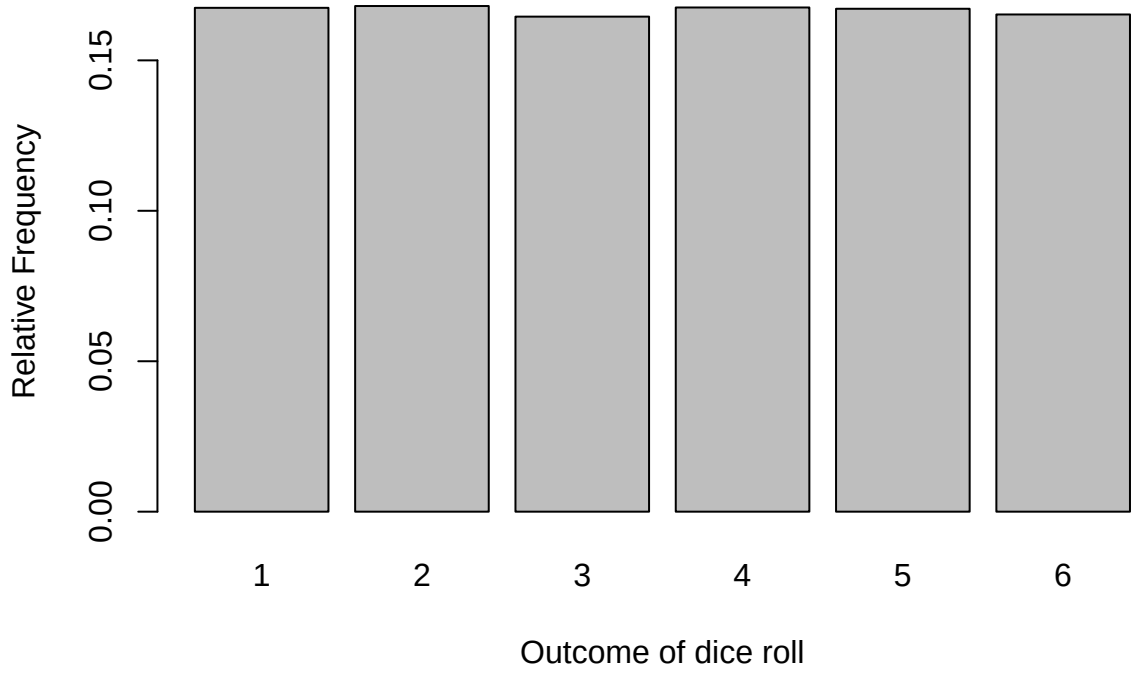
```
## [1] 4
```

重複十萬次擲一次骰子的試驗，看看這十萬個點數有幾個 1 點、幾個 2 點，...到幾個 6 點用古典機率的定義：機率為樣本空間的大小分之事件的大小

```r
set.seed(1234)
dice_num = replicate(n = 100000, expr = dice(6,1))
count = c()
for(i in 1:6){
  count = c(count, sum(dice_num == i))
}
count
```

```
## [1] 16745 16806 16453 16757 16714 16525
```

```r
p_dice = count/100000
barplot(p_dice, names.arg = c(1,2,3,4,5,6), ylab = 'Relative Frequency', xlab = 'Outcome of dice roll')
```



Calculating expected value: $E(X) = \sum_{x=1}^{6} xP(X = x)$

```r
expected_dice = 1*p_dice[1]+2*p_dice[2] +3*p_dice[3]+4*p_dice[4]+5*p_dice[5]+6*p_dice[6]
expected_dice
```

```
## [1] 3.49464
```

#When the mechanism behind the random process is discrete uniform, then I can use `mean()`

```r
mean(dice_num) #Tells me E(X) = 3.5
```

```
## [1] 3.49464
```

```r
expected_dice-mean(dice_num)==0 #there is minor difference when the time of rolling is not large enough
```

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

Try the following… Let X be a r.v. represents the sum of the number of rolling a fair dice 10 times * What is E(X)?

```
mean(replicate(n = 1000, sum(dice(6,10)))) #simply use the 'disrete uniform' approach
```

```
## [1] 35.102
```

```
#Why it equals to 10 times the above result?
```