In the name of God

Producer:

Mehrab Atighi

Subject :

What we do to estimate the pi value with montcarlo method in R

Date:

11/11/2020

Supervisor:

Dr. Seyed Noorullah Mousavi

Issue:

A) Approximate the Pi number with the Montcarlo method and display the graphs of this method.

B) Repeat this approximation 200 times and obtain a graph of the averages of these approximations at points for different values of N, and then plot its standard deviation as a line on either side of the same point next to each mean.

Solve:

we do this to estimate the pi value with montcarlo method.

First to consider this we have to form a square on the side of a unit (0,1) then we form a circle to the center of the coordinate origin, it is clear that a quarter of the area of the circle is inside the square

Next, we examine the ratio of the number of points in the circle to the total points:

```
> rm(list = ls())
> x<-runif(2^20,0,1)
> y<-runif(2^20,0,1)
> d<-(x^2+y^2<1)
> plot(x,y,col=d+1 , pch=10)
```

Now we can see the chart above:

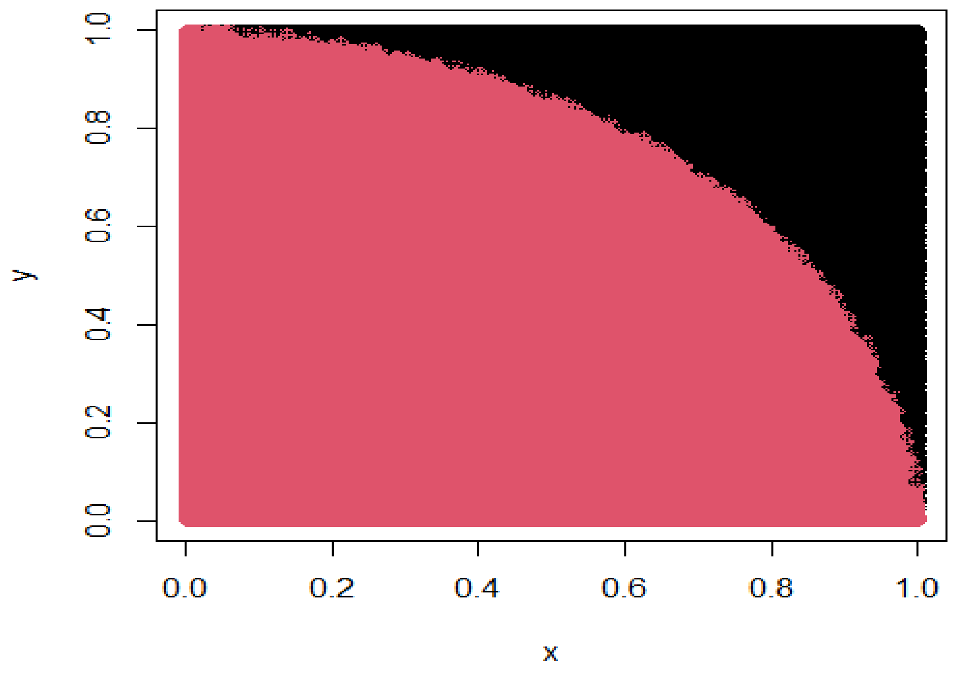

Figure.1-3

In the diagram above, the red part is the quarter of the circle.

Now to store all 2^N states we have to save a matrix in a row each time we run the method, and finally we have a matrix with the number of rows and columns A_{200*2^N}

```
q<-c(1:(2^20*200))
> A<-matrix(q, nrow = 200)
> for (j in 1:200) {
+   a<-0
+   b<-1
+   N<-20
+   l<-0
+   num<-2^N
+   x<-runif(num,a,b)
+   y<-runif(num,a,b)
+   k<-c()
+   for(i in 1:num){
+     if(x[i]^2+y[i]^2<1){
+       l<-l+1
+       k[i]<-l}
+     else
+     {
+       k[i]<-l
+     }
+     A[j,i]<-4*(k[i]/i)
+   }
+ }
```

Now by dividing the total number of points by the total number of points that were placed in the quadrilateral, we get the approximate value of the area of the quadrilateral and multiply by 4 to approximate the number.

```
> print("the pi value estiamte is:")
[1] "the pi value estiamte is:"
> (4*(k[i]/num))
[1] 3.142189
```

And we draw four diagrams of these 200 states:

```
> for(j in 1:4){
+   par(new= "TRUE")
+   for (i in 1:N) {
+   pi[i]<-A[j,2^i]}
+   plot(n,pi , col= 85*j , type = "o" , ylab = "the pi estimate value"
, xlab = "2^N" ,ylim =c(0,6), new= "True")}
> abline(h=3.141593  , col= "150")
```

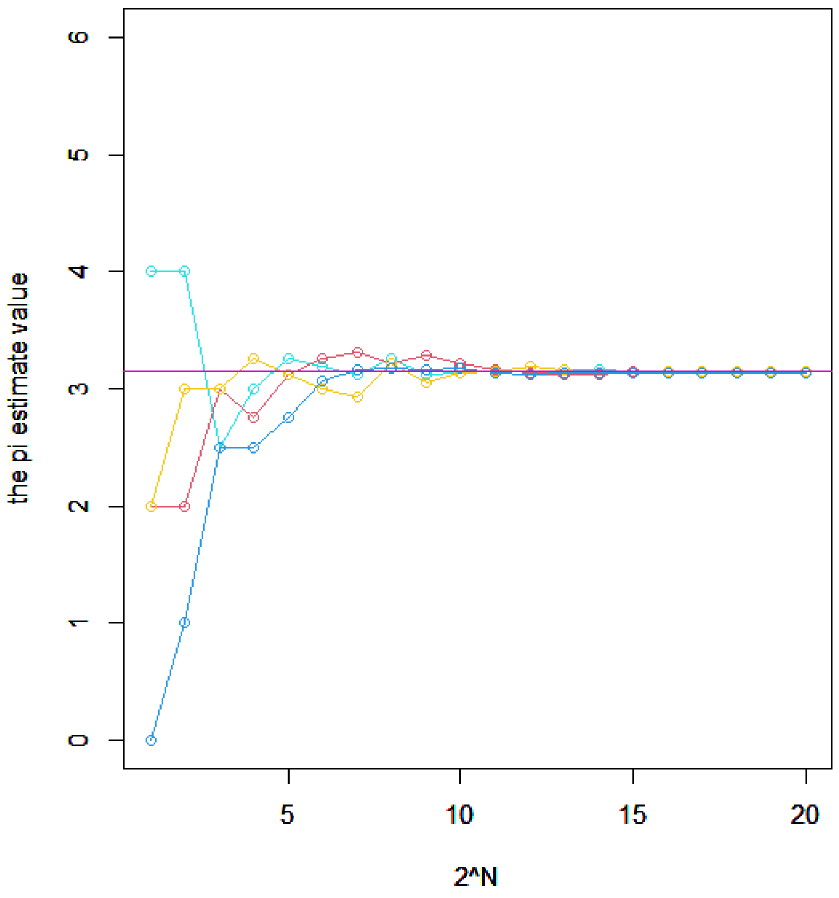

Figure.1-2

Now for twenty points from these 2^N to the average point we get the standard repetition and deviation of 200 times:

```
> Mean<-c()
> a<-c()
> a<-apply(A, 2,mean)
> for (i in 1:N){
+   Mean[i]<-a[2^i]
+ }
> sigma<-c()
> b<-c()
> b<-apply(A, 2,sd)
> for (i in 1:N){
+   sigma[i]<-b[2^i]
+ }
```

Now we draw a graph so that the points represent our averages at these twenty points, and the vertical lines near these points represent the standard deviations of the approximated points for the foundation at that point:

```
 > par(mfrow= c(1,1))
 > plot(Mean ,type="p" , col="red" ,xlab= "2^N" , ylab = "Mean of pi value estimates" )
 > for(i in 1:N){
 +   segments(n[i],Mean[i]-sigma[i],x1=n[i] , y1 = Mean[i]+sigma[i] , col = "85")}
```

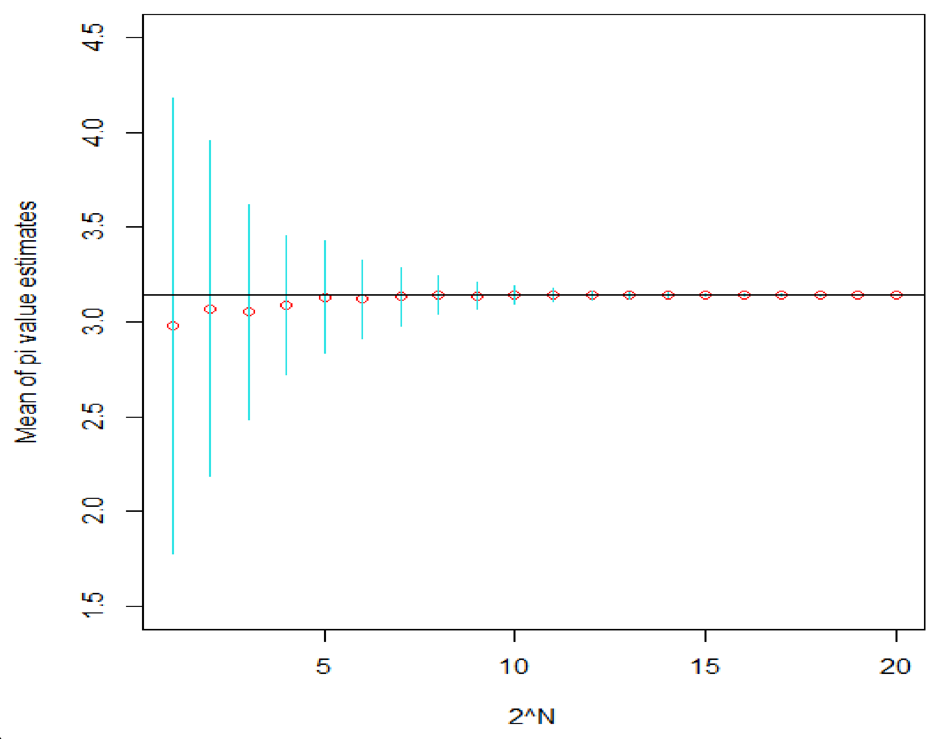

Figure.1-3

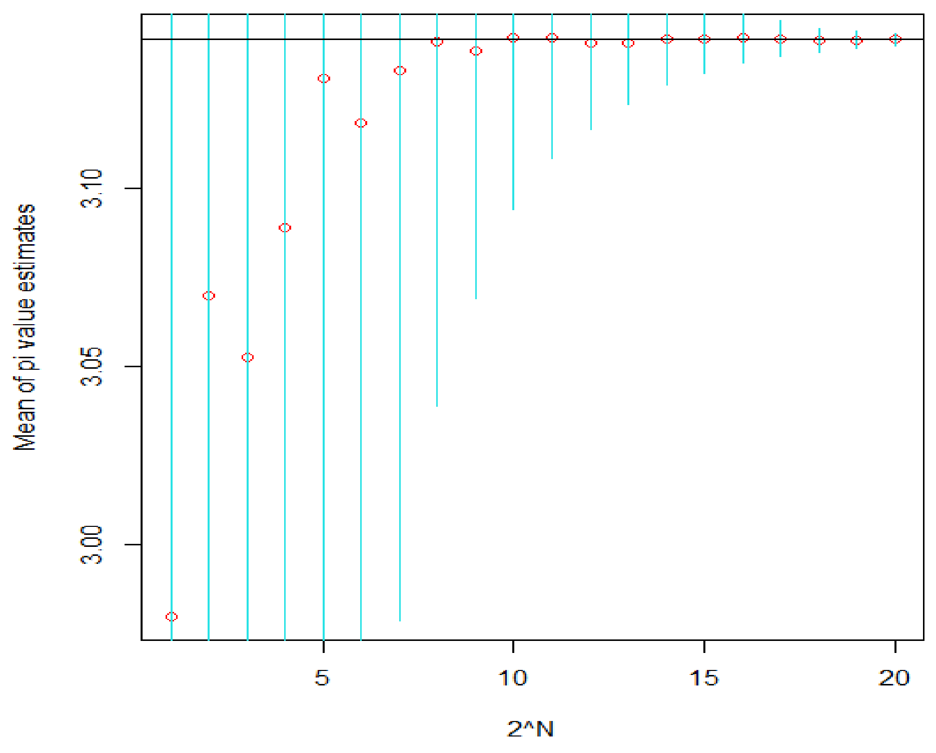

Figure1-4

Conclusion :
As you can see in the diagrams1-2, 1-3the more random samples we have, the more accurate the answer will be.

And the more we have, the less our standard deviations and the more accurate the average.

I hope these commands and solving this example with them have solved your problem.

Thanks.