EE 511
Fall 2018
Prof. John Silvester

Project 2 - Monte Carlo Methods

Pavan Athreya Narasimha Murthy
USC ID: 9129210968
Section: Wednesday 9:00 AM
E-mail: pavanatn@usc.edu

1. Estimate π by the area method including confidence intervals on your estimate. Draw a graph of the successive values of the estimator as the number of samples increases.

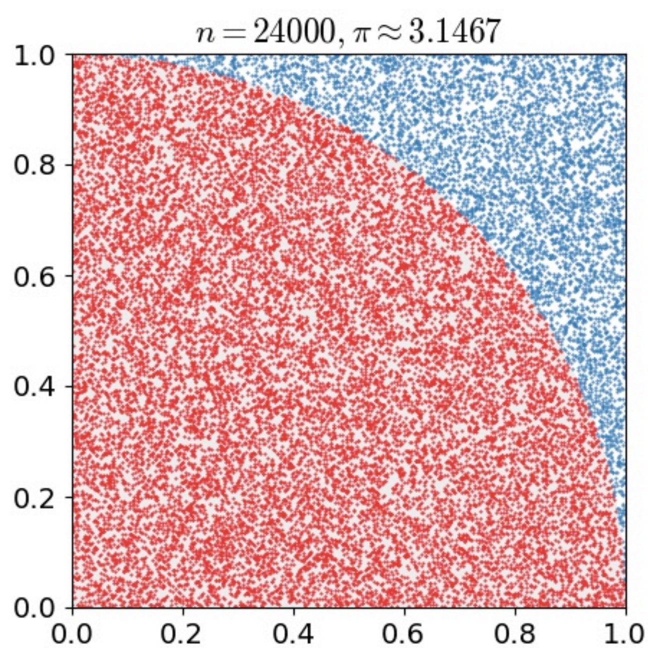
How many points do you need to use for your estimate to be within $\pm 1\%$ of the true value of π (with probability 0.95)?

Problem Statement:

To estimate the value of pi with 95% confidence interval and to estimate the number of points needed to get the true value of pi with probability of 95% (0.95)

Theory/Analysis:

Theory is explained in the next page by hand.



References:

- Class notes
- Wikipedia - https://en.wikipedia.org/wiki/Monte_Carlo_method
- Spring 2018 EE503 notes on pi estimate and Monte Carlo estimations

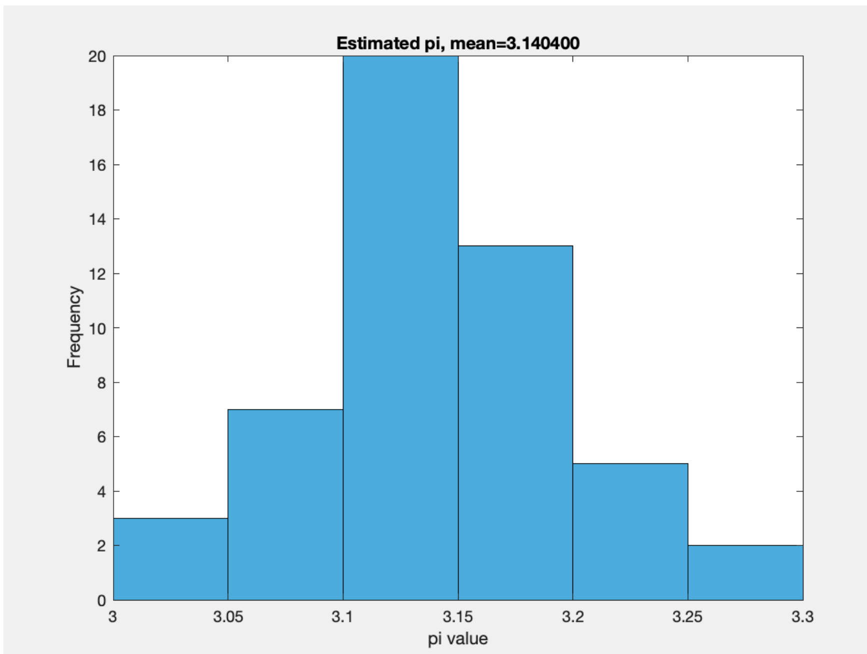
Simulation Methodology:

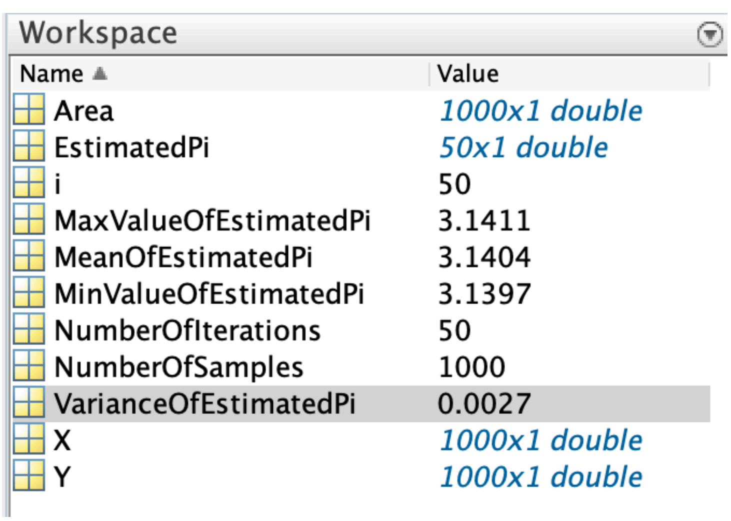
A 1000 random samples are chosen and 50 iterations are performed to check whether the given random sample falls inside the circle. This is done by verifying whether the area formed by the point is less than one.

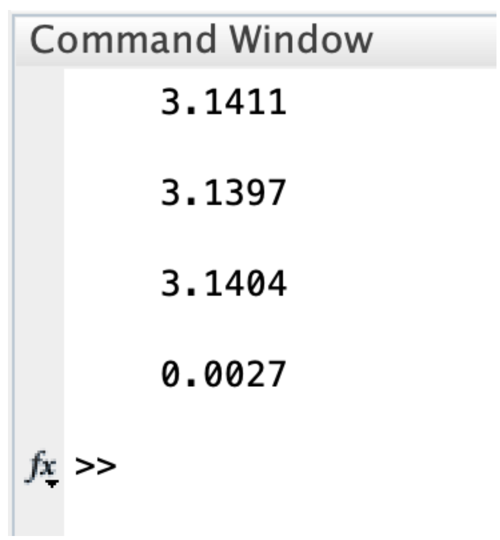
Following that, the confidence interval concept is applied to check the max estimated and min estimated value with 95% confidence interval.

Finally a histogram is presented to show the estimates of pi

Results:







Source Code:

```
%Name: Pavan Athreya Narasimha Murthy
%USC ID: 9129210968
%E-mail: pavanatn@usc.edu
%Ph: +1(323)-684 5715
%Term: Fall 2018
%Course: EE511
%Professor: John Silvester

%Clear the Workspace variables and command window for every run
clear all;
clc;

%First part of Project 2
%Estimation of Pi value
NumberOfSamples = 1000;
NumberOfIterations = 50;
EstimatedPi = zeros(NumberOfIterations,1);

%Running the iterations to check the points of the circle
for i = 1:NumberOfIterations
   X = rand([NumberOfSamples 1]);
   Y = rand([NumberOfSamples 1]);
   Area = X.^2 + Y.^2;
   EstimatedPi(i,1) = 4 * length(find(Area<1))/NumberOfSamples;
end

%Plotting the histogram
figure;
histogram(EstimatedPi);
title(sprintf('Estimated pi, mean=%f',sum(EstimatedPi)/length(EstimatedPi)));
xlabel("pi value");
ylabel("Frequency");

%Variance
VarianceOfEstimatedPi = var(EstimatedPi);
MeanOfEstimatedPi = mean(EstimatedPi);

%Confidence interval of 95%
MaxValueOfEstimatedPi = ((MeanOfEstimatedPi/4)+((1.96*VarianceOfEstimatedPi)/
sqrt(NumberOfSamples)))*4;
MinValueOfEstimatedPi = ((MeanOfEstimatedPi/4)-((1.96*VarianceOfEstimatedPi)/
sqrt(NumberOfSamples)))*4;

%Display the results
disp(MaxValueOfEstimatedPi);
disp(MinValueOfEstimatedPi);
disp(MeanOfEstimatedPi);
disp(VarianceOfEstimatedPi);
```

2. Consider a deck of cards (for simplicity numbered 1.. N). Use a uniform random number generator to pick a card and record what card it is (if you were using actual cards, you would replace the card back into the deck – that is not necessary here since we never really take the card out of the deck). Repeat this N times, recording the number of times that each of the cards is selected. Some cards may not show up (actually, it is very likely that several card numbers will not show up) and some will show up more than once. You can use this data to estimate the following probabilities:

$$p_j = \Pr\{\text{a card will be selected } j \text{ times in the } N \text{ selections}\}$$

It is unlikely that any card will show up more than about 10 times. Run this for $N = 10,\ N = 52,\ N = 100,\ N = 1{,}000, N = 10{,}000$ and verify that $p_0 \simeq 1/e$. Can you also find values for the other p_j based on a mathematical analysis?

Problem Statement:

Count the number of times a card shows up when we pick them from the deck with replacement.

Finally to calculate the probability that a card will be selected j times in N selections

Also to verify that the probability of the card not getting selected in the deck is approximately equal to 1/e

Theory/Analysis:

Theory is hand written in next page. Mathematical analysis for Pj is also explain the in the write out.

Simulation Methodology:

Number of trials and the number to be selected form the deck of cards to calculate the probability is taken as input.
A random number between 1 and 52 is created, based on which the card is picked from the deck.
Then for every trial, the count for each picked card is incremented.
At the end, the probability for the chosen card is calculated.

Results:

N = 10

Command Window

```
Enter the card number:4
Probability of the card will be selected j times in N selections
      0

p0 for the given user given input card
    0.3487

>>
```

N = 52

Command Window

```
Enter the card number:6
Probability of the card will be selected j times in N selections
      0

p0 for the given user given input card
    0.3643

>>
```

N = 100

Command Window

```
Enter the card number:8
Probability of the card will be selected j times in N selections
    0.0100

p0 for the given user given input card
    0.3660

>>
```

N = 1000

```
Command Window
Enter the card number:2
Probability of the card will be selected j times in N selections
    0.0210

p0 for the given user given input card
    0.3677

>>
```

N = 10000

```
Command Window
Enter the card number:9
Probability of the card will be selected j times in N selections
    0.0191

p0 for the given user given input card
    0.3679

>>
```

References:

Class notes

Source Code:

```
%Name: Pavan Athreya Narasimha Murthy
%USC ID: 9129210968
%E-mail: pavanatn@usc.edu
%Ph: +1(323)-684 5715
%Term: Fall 2018
%Course: EE511
%Professor: John Silvester

%Clear the Workspace variables and command window for every run
clear all;
clc;

%Second part of Project 2
NumberOfTrials = 10;
```

```
CardChoosen = input('Enter the card number:');
NumberOfTimesCardArray = zeros(52, 1);

for i = 1:NumberOfTrials
    RandomCard = randi([1 52], 1, 1);
    NumberOfTimesCardArray(RandomCard, 1) = NumberOfTimesCardArray(RandomCard, 1) +
1;
end

ProbabilityOfCardSelected = NumberOfTimesCardArray(CardChoosen, 1)/NumberOfTrials;
disp('Probability of the card will be selected j times in N selections');
disp(ProbabilityOfCardSelected);

disp('p0 for the given user given input card');
p0 = (1 - (1/NumberOfTrials)).^NumberOfTrials;
disp(p0);
```

3. Use the method discussed in class to find $\hat{y}$, an estimate for Y and find a 95% confidence interval for the value of the integral.

$$Y = \int_0^{\pi} \frac{\sin(x)}{x}\,dx$$

Problem Statement:

To estimate the value of the sinc function over 0 to pi using the monte carlo integration method taught in class.
In addition to that we should also find the 95% confidence interval for the integral.

Theory/Analysis:

Written part is attached.

Simulation Methodology:

1. The limit of integral is set using two variables x1 and x2.
2. The value of the function at limits is calculated
3. The rand function is used to calculate the X and Y values, later the area is calculated
4. All the X, Y and Area values are used to calculate integral value and then the mean value is obtained
5. The 95% confidence interval for the integral value is calculated

Results:

```
Command Window
Mean estimate of the sinc integral
    1.5416

95% confidence interval values
Max value
    1.5419

Min value
    1.5414

>>
```

Source Code:

```
%Name: Pavan Athreya Narasimha Murthy
%USC ID: 9129210968
%E-mail: pavanatn@usc.edu
%Ph: +1(323)-684 5715
%Term: Fall 2018
%Course: EE511
%Professor: John Silvester

%Clear the Workspace variables and command window for every run
clear all;
clc;

%Third part of Project 2
NumberOfSamples = 1000;
NumberOfIterations = 100;
IntegralValue = zeros(NumberOfIterations, 1);

x1 = 0.0;
x2 = pi;
y1 = sinc(x1);
y2 = sinc(x2);
```

```
for i = 1:NumberOfIterations
    X = (x2-x1).*rand(NumberOfSamples, 1) + x1;
    Y = (y2-y1).*rand(NumberOfSamples, 1) + y1;
    Area = sinc(X);
    IntegralValue(i,1) = abs(x2-x1)*abs(y2-y1)*length(find(Area>Y))/NumberOfSamples;
end

MeanValue = mean(IntegralValue);
VarianceValue = var(IntegralValue);
disp('Mean estimate of the sinc integral');
disp(MeanValue);

%Confidence Interval
MaxValueOfIntegralValue = ((MeanValue) + ((1.96*VarianceValue)/sqrt(NumberOfSamples)));
MinValueOfIntegralValue = ((MeanValue) - ((1.96*VarianceValue)/sqrt(NumberOfSamples)));

disp('95% confidence interval values');
disp('Max value');
disp(MaxValueOfIntegralValue);
disp('Min value');
disp(MinValueOfIntegralValue);
```