Monte Carlo Simulation

CSE1010 Project 4, Fall 2012

Date: 10/1/2012

Name: William Dickson

Section: 009L

TA: Levon Nazaryan

Instructor: Jeffrey A. Meunier

1. **Introduction**

Monte Carlo is a statistical method using random sampling that is used to perform an estimation of something that is difficult or impossible to measure. This process was named after the city of Monte Carlo in Monaco, which is known for its casinos.

1. **Test runs**
   1. **Test run of field size 4, pond radius 1, 10 shots**

Enter the size of the field: 4

Enter the radius of the pond: 1

Enter the number of shots: 10

Number of shots = 10

Number of splashes = 3

Estimated pond area = 4.8

Estimated value of pi = 4.8

Error in estimation of pi = 52.789%

* 1. **Test run of field size 4, pond radius 1, 100 shots**

Enter the size of the field: 4

Enter the radius of the pond: 1

Enter the number of shots: 100

Number of shots = 100

Number of splashes = 24

Estimated pond area = 3.84

Estimated value of pi = 3.84

Error in estimation of pi = 22.231%

* 1. **Test run of field size 4, pond radius 1, 1000 shots**

Enter the size of the field: 4

Enter the radius of the pond: 1

Enter the number of shots: 1000

Number of shots = 1000

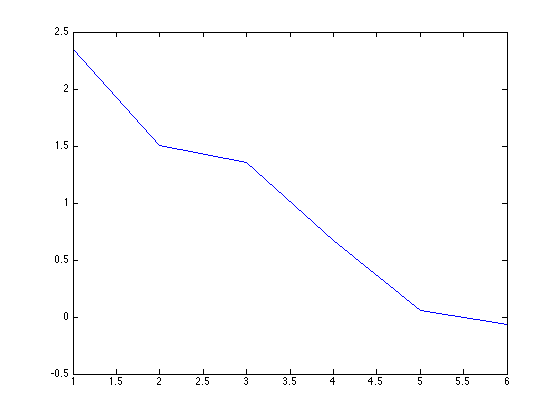
Number of splashes = 216

Estimated pond area = 3.456

Estimated value of pi = 3.456

Error in estimation of pi = 10.008%

1. **Plot of log(mean)**



This plot shows the log of the mean values of the errors in the calculations of pi. The reason why we use the graph of the log is because there is an exponential relationship between the number of shots we use and the accuracy of our calculation of the value of pi. The plot has a negative slope because as the number of shots increases, our error in calculating pi decreases.

1. **Spreadsheet data**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Run1** | **Run 2** | **Run 3** | **Run 4** | **Run 5** | **mean** | **log(mean)** |
| **1** | 100 | 100 | 100 | 409.3 | 409.3 | 223.72 | 2.349704811 |
| **10** | 100 | 52.789 | 1.8592 | 1.8592 | 1.8952 | 31.68052 | 1.500792301 |
| **100** | 3.2338 | 27.324 | 6.9521 | 28.699 | 47.696 | 22.78098 | 1.357572403 |
| **1000** | 4.9149 | 9.4986 | 0.84057 | 0.33128 | 7.9707 | 4.71121 | 0.673132463 |
| **10000** | 0.22895 | 0.27988 | 1.5531 | 1.6045 | 2.0624 | 1.145766 | 0.059095931 |
| **100000** | 1.1156 | 0.61092 | 1.0744 | 1.5175 | 0.0099521 | 0.86567442 | -0.062645415 |

This spreadsheet shows the values generated from monteCarlo.m with filedSize = 4, pondRadius = 1, and numShots = the values on the leftmost column (1, 10, 100, etc). From the data generated from 5 runs of each value for numShots, the means and log of the means were calculated and graphed on the plot above.

1. **Source Code**

% Monte Carlo Area calculation

% CSE1010 Project 4, Fall 2012

% William Dickson

% 10/1/2012

% TA: Levon Nazaryan

% Section: 009L

% Instructor: Jeffrey A. Meunier

clc % clear command window

clear % clear all variables

clf % clear plot area

rng('shuffle') % shuffle the random number generator

fieldSize = input('Enter the size of the field: ');

pondRadius = input('Enter the radius of the pond: ');

numShots = input('Enter the number of shots: ');

setupField(fieldSize);

plotPond(pondRadius);

numSplashes = fireShots(numShots,fieldSize,pondRadius);

fprintf('Number of shots = %g\n',numShots)

fprintf('Number of splashes = %g\n',numSplashes)

estimatedPondArea = estimatePondArea(numSplashes,numShots,fieldSize);

fprintf('Estimated pond area = %g\n',estimatedPondArea)

estimatedPi = estimatePi(estimatedPondArea,pondRadius);

fprintf('Estimated value of pi = %g\n',estimatedPi)

piErrorPercentage = piErrorPercent(estimatedPi);

fprintf('Error in estimation of pi = %4.5g%%\n',piErrorPercentage)

function estimatesPi = estimatePi(estimatedPondArea,pondRadius)

% Estimates the value of pi.

% Use: estimatePi(estimatedPondArea,pondRadius)

estimatesPi = estimatedPondArea/pondRadius^2;

end

function estimatedPondArea = estimatePondArea(numSplash,numShot,fieldSize)

% Estimates the area of the pond.

% Use: estimatePondArea(numSplash,numShot,fieldSize)

estimatedPondArea = numSplash/numShot\*fieldSize^2;

end

function plotShotResult = fireShot(fieldSize,pondRadius)

% Fires a single shot and returns either a 'thud' or 'splash' for missing

% or hitting the pond respectively.

% Use: fireShot(fieldSize,pondRadius)

xLoc = rand\*fieldSize-fieldSize/2;

yLoc = rand\*fieldSize-fieldSize/2;

plotShotResult = plotShot(pondRadius,xLoc,yLoc);

end

function splashCount = fireShots(numShots,fieldSize,pondRadius)

% Fires a number of shots into the pond using the fireShot function and

% returns the number of shots that hit the pond.

% Use: fireShots(numShots,fieldSize,pondRadius)

splashCount = 0;

while numShots>0

numShots=numShots-1;

plotShotResult = fireShot(fieldSize,pondRadius);

if strcmp(plotShotResult,'splash') == 1

splashCount = splashCount + 1;

end

end

end

function result = hitPond(pondRadius,xLoc,yLoc)

% Determines whether a point is inside or outside the pond and returns

% true of false respectively.

% Use: hitPond(pondRadius,xLoc,yLoc)

if sqrt(xLoc^2+yLoc^2)<pondRadius

result=true;

else

result=false;

end

end

function piErrorPercentResult = piErrorPercent(estimatePi)

% Determines and returns the percent error between the calculated value of pi and the

% actual value of pi.

% Use: piErrorPercent(estimatedPi)

piErrorPercentResult = 100\*abs((estimatePi-pi)/pi);

end

function plotPond(pondRadius)

% Plots the pond

% Use: plotPond(pondRadius)

theta=linspace(0,2\*pi,100);

x=pondRadius\*sin(theta);

y=pondRadius\*cos(theta);

plot(x,y)

end

function plotShotResult = plotShot(pondRadius,xLoc,yLoc)

% Plots a point on the pond and returns a splash or a thud for a point

% inside or outside the pond respectively.

% Use: plotShot(pondRadius,xLoc,yLoc)

if hitPond(pondRadius,xLoc,yLoc)==true

plot(xLoc,yLoc, 'b\*')

plotShotResult = 'splash';

else

plot(xLoc,yLoc, 'go')

plotShotResult = 'thud';

end

end

function setupField(fieldSize)

% Sets up the plot window for the simulation.

% Use: setupField(fieldSize)

title('Shots fired into a field with a round pond')

axis([-fieldSize/2, fieldSize/2, -fieldSize/2, fieldSize/2])

axis equal

hold on

end