Topics: Simulation, MATLAB wrap-up

Reading: -

Simulation of systems

Simulation is the application of mathematical and computer models that imitate the behavior of a system. Simulation is a useful tool for design, training, and games!

Simple dice game

Simulate the rolling of a fair die. The function below allows the user to specify the number of rolls. Be careful about using the random number generator for generating integers *with equal probability*.

```
function freq = rollDice(rolls)
% Simulate rolling of fair 6-sided die
% Usage:  freq = rollDice(rolls)
%   ROLLS is the number of times to roll die
%   FREQ is vector of frequencies of possible outcomes

SIDES = 6;              % number of sides on die
freq = zeros(1,SIDES);

% Roll FAIR die
allRolls = ceil(rand(1,rolls)*SIDES);

% Count outcomes
for i = 1:rolls
    freq(allRolls(i)) = freq(allRolls(i)) + 1;
end

% Show histogram of outcome
% YOU ARE NOT RESPONSIBLE FOR LEARNING hist
hist(allRolls,1:SIDES);
title(['Outcomes from ' num2str(rolls) ...
       ' rolls of fair die']);
xlabel('Outcome');  ylabel('Frequency');
```

Estimate Pi

The mathematical “constant” π can be approximated in many ways. One method is to use Monte Carlo simulations of dart throwing!
Let N be the number of darts thrown randomly over a square domain of area $L \times L$. The largest circle that can fit inside this domain has a diameter of L and an area of $\pi L^2/4$.
Let the number of darts N be the area of the square domain:

$$N = L \times L. \tag{1}$$

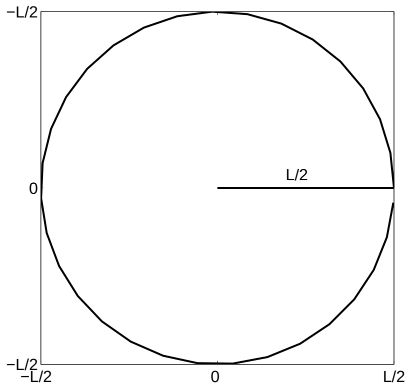
Then the number of darts that fall inside the circle, N_{in}, is the area of the circle:

$$N_{in} = \frac{\pi L^2}{4}. \tag{2}$$

Substitute equation (1) into (2) to get π:

$$\pi = \frac{4N_{in}}{N} \tag{3}$$

The following function performs Monte Carlo simulations of dart throwing. The function argument is the number of darts to be thrown.




```
function myPi = approxPi(nDarts)
% Approximate Pi using Monte Carlo simulations
% Usage:  myPi = approxPi(nDarts)
%   NDARTS = number of "darts" thrown
%   myPi = Monte Carlo approximation of Pi

L = 10;  % length of square

% Throw darts in L-by-L area, centered at 0
throws = L*rand(nDarts,2) - L/2;
x = throws(:,1);  % x-coordinates of darts
y = throws(:,2);  % y-coordinates of darts

% Location of darts relative to center
  dist = sqrt(x.^2+y.^2);  % distance from center
  nIn = sum(dist <= L/2);  % #darts inside circle

myPi = 4*nIn/nDarts;

% Plot darts in domain
% YOU ARE NOT RESPONSIBLE FOR LEARNING AXIS FORMATS
  % Circle data
    theta = 0:0.2:2*pi;
    xcircle = cos(theta)*L/2;
    ycircle = sin(theta)*L/2;
plot(xcircle,ycircle,'r',x,y,'*','linewidth',2)
axis([-L/2 L/2 -L/2 L/2]);  axis('square');
title(['Pi = ' num2str(myPi)]);
```