Lab Experiments in Probability

James A. Reising

University of Evansville

Abstract

The subject of probability occurs not only in courses on probability and statistics, but also in courses covering stochastic processes. The concept of using probability models to describe real-life problems seems difficult for many students to grasp. Mathematical software, such as MATLAB, provides a useful tool in performing simulations using probability models.

To assist students in forming a conceptual link between the results of actual experiments and practical situations in which the outcome is predictable only in a probabilistic sense, several simple projects involving repeated trials of an experiment are used in a course in probability or random signals. MATLAB programs simulating the same experiment are assigned as part of each project.

This paper describes several such experiments and the associated MATLAB simulations. Students working in groups of three or four compare their experimental results with the MATLAB simulations and to the results of other groups in the class.

By comparing the actual and simulated results, students may develop some confidence in the use of computational software to simulate experiments for larger numbers of trials than they can realistically perform in practice.

Introduction

Random variables are a key concept in the study of probability and random processes. The expected value and variance of a random variable are key concepts in probability theory. These definitions can be extended to sums of random variables. Let $X_i$ represent one of a number of discrete random variables and $E[X_i]$ the expected value of $X_i$. For a sum of $n$ random variables,

$$E[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} E[X_i]$$

is generally true. If the $X_i$ are mutually independent,

$$Var[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} Var[X_i]$$

"Proceedings of the 2005 American Society for Engineering Education Annual Conference & Exposition
Copyright © 2005, American Society for Engineering Education"
can be used to determine the variance of the sum\(^2\). Furthermore, the Central Limit Theorem\(^3\), states that as \( n \to \infty \), the distribution of the sum approaches a Gaussian distribution having the same mean and variance. For finite values of \( n \), the Gaussian distribution may provide a reasonable approximation to the actual distribution. Two student exercises using standard and polyhedral dice can be used to illustrate the Central Limit Theorem. The Gaussian approximation may be a reasonable for values of \( n \) as small as five for several hundred repeated trials, although when five non-identical dice are used, a much greater number of trials is required for the approximation to seem reasonable. The use of MATLAB (The MathWorks, Inc.) to simulate rolling dice is described elsewhere\(^4\) for identical dice. The second dice experiment described below uses MATLAB to simulate rolling non-identical dice as well.

MATLAB was used for all calculations. The experimental data for the dice experiments were stored as m-files. In the case of the identical dice experiment, data were stored in an array of 200 rows and 5 columns, since data entry was easier in that format. The transpose of this array was then used in the calculations. Similarly, the data array for the non-identical dice experiment was 200 x 6.

To aid in the simulations, an m-file implementing a function returning an array of simulated dice rolls used the MATLAB random number generator. The numbers in the resulting array were used to create an array of the appropriate dimensions for further analysis.

In the case of random variables with a continuous range of values, the probability that a single random variable \( X \) has a value in the range \( a \leq X \leq b \) is

\[
P(a \leq X \leq b) = \int_a^b f(x)dx, \text{ where } a \leq b \text{ and } f(x) \text{ is the probability density function}^1.
\]

**Dice Experiments**

**Identical Dice**

- Teams of students roll five ordinary six-sided dice, recording the results.
- The teams share the data they have collected.
- Each team prepares a histogram of the sum of the spots on the dice.
- Each team compares the experimental results to theoretically predicted results.
- Each team produces a MATLAB program that simulates their experiment.

Theoretical predictions are derived as follows:

For a single die

\[
E[X_i] = \frac{1+2+3+4+5+6}{6} = \frac{7}{2}
\]

\[
Var[X_i] = E[X_i^2] - (E[X_i])^2
\]
\[ E[X_i^2] = \frac{1 + 4 + 9 + 16 + 25 + 36}{6} = \frac{91}{6} \]

giving \( \text{Var}[X_i] = \frac{35}{12} \).

For the sum of the numbers on five such dice

\[ E[\sum_{i=1}^{n} X_i] = 5E[X_i] = 5(\frac{7}{2}) = 17.5000 = \mu, \text{ using equation (1) above.} \]

\[ \text{Var}[\sum_{i=1}^{n} X_i] = 5\text{Var}[X_i] = 5(\frac{35}{12}) = 14.5833 \text{, using equation (2) above.} \]

\[ \sigma = \sqrt{\text{Var}[\sum_{i=1}^{n} X_i]} = 3.8188 \text{ is the standard deviation.} \]

The histogram and numerical results for the identical dice experiment and two MATLAB simulations for 200 trials are shown below. The y axis is normalized by the total number of tries. The solid curve is the normal probability distribution. The Central Limit Theorem may be illustrated by comparing the histograms with the normal probability density having the theoretical mean and variance.

![Figure 1](image1.png) ![Figure 2](image2.png) ![Figure 3](image3.png)

The plot of experimental data from 200 rolls in Figure 1 was produced using the MATLAB program shown as Listing 4 in the appendix. The plots of two simulations of 200 rolls are shown in Figures 2 and 3, produced using the MATLAB program shown as Listing 5 in the appendix.

The MATLAB program shown as Listing 6 in the appendix was used to produce the three simulations of rolling the identical dice 200,000 times shown in Figures 4, 5, and 6 below.
Non-identical Dice

- Teams of students roll a group of dice consisting of two ordinary six-sided dice, two eight-sided dice with faces numbered 1 through 8, and two ten-sided dice with faces numbered 0 through 9.
- The teams share the data they have collected.
- Each team prepares a histogram of the sum of the spots on the dice.
- Each team compares the experimental results to theoretically predicted results.
- Each team produces a MATLAB program that simulates their experiment.

Theoretical predictions are derived as follows:

For a single six-sided die

\[
E[X_i] = \frac{1 + 2 + 3 + 4 + 5 + 6}{6} = \frac{7}{2}
\]

\[
Var[X_i] = E[X_i^2] - (E[X_i])^2
\]

\[
E[X_i^2] = \frac{1 + 4 + 9 + 16 + 25 + 36}{6} = \frac{91}{6}
\]

giving \( Var[X_i] = \frac{35}{12}. \)

For a single eight-sided die

\[
E[X_i] = \frac{1 + 2 + 3 + 4 + 5 + 6 + 7 + 8}{8} = 4.5
\]

\[
Var[X_i] = E[X_i^2] - (E[X_i])^2
\]
\[ E[X_i^2] = \frac{1 + 4 + 9 + 16 + 25 + 36 + 49 + 64}{8} = 25.5 \]

giving \( Var[X_i] = 5.25 \).

For a single ten-sided die
\[ E[X_i] = \frac{0 + 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9}{10} = 4.5 \]

\[ Var[X_i] = E[X_i^2] - (E[X_i])^2 \]

\[ E[X_i^2] = \frac{0 + 1 + 4 + 9 + 16 + 25 + 36 + 49 + 64 + 81}{10} = 28.5 \]

giving \( Var[X_i] = 8.25 \).

For the sum of the numbers on two six-sided dice, two eight-sided dice, and two ten-sided dice
\[ E[\sum_{i=1}^{n} X_i] = 2(3.5) + 2(4.5) + 2(4.5) = 25.0000 \text{ } = \text{mu, using equation (1) above.} \]

\[ Var[\sum_{i=1}^{n} X_i] = 2\left(\frac{35}{12} + 5.25 + 8.25\right) = 32.8333 \text{ , using equation (2) above.} \]

\[ \sigma = \sqrt{Var[\sum_{i=1}^{n} X_i]} = 5.7300 \text{ is the standard deviation.} \]

The histogram and numerical results for the identical dice experiment and two MATLAB simulations for 200 trials are shown below. The y axis is normalized by the total number of tries. The solid curve is the normal probability distribution. The Central Limit Theorem may be illustrated by comparing the histograms with the normal probability density having the theoretical mean and variance.
The plot of experimental data from 200 rolls in Figure 7 was produced using the MATLAB program shown as Listing 7 in the appendix. The plots of two simulations of 200 rolls are shown in Figures 8 and 9, produced using the MATLAB program shown as Listing 8 in the appendix.

The MATLAB program shown as Listing 9 in the appendix was used to produce the three simulations of rolling the non-identical dice 200,000 times shown in Figures 10, 11, and 12 below.

![Figure 10](image1.png)  ![Figure 11](image2.png)  ![Figure 12](image3.png)

It is apparent that, compared to the identical dice experiment, a larger number of trials is necessary for the non-identical dice experiment to approach the normal distribution. The eight and ten-sided dice make greater contributions to the sums when faces with 7, 8, or 9 face up.

**Buffon's Needle Experiment**

A variation of the Buffon's Needle experiment is assigned to teams of students. Lines 2" apart are ruled parallel to the 22" edge of a 22" x 28" sheet of poster board. A 1" x 1/8" piece is cut from a thin sheet of plastic to serve as the "needle". The piece of plastic is tossed from a distance of 1 foot or more onto the poster board, giving it a spin when it is tossed. An observation consists of noting whether the "needle" touches or crosses a line when it lands. Data is recorded as either a zero (does not touch or cross a line) or a one (touches or crosses a line). The teams share the data they have collected.

Each team compares the collected data to the theoretical prediction of the fraction of the observations for which the "needle" touched or crossed a line, taking into account the non-zero width of the "needle".

A typical Buffon's Needle experiment involving 400 trials yielded 141 trials when the "needle" either touched or crossed a line. That is, the number of crossings was 141/400 or 0.3525 of the total number of trials. A probability model for the experiment can be developed as follows:

The figure represents a "needle" of length L and width W.
The two dark horizontal lines represent lines on the poster board spaced a distance D apart. Assume that theta is a random variable uniformly distributed between 0 and π and that r is a random variable uniformly distributed between 0 and D. The "needle" will cross a line if either

\[ r < \frac{L}{2} \sin(\theta) + \frac{W}{2} |\cos(\theta)| \quad \text{or} \quad \]
\[ r > D - \left( \frac{L}{2} \sin(\theta) + \frac{W}{2} |\cos(\theta)| \right) \]

Figure 13

Assuming that r and theta are independent, the joint probability density of r and theta is \( \frac{1}{\pi D} \).

Using this probability density in equation (3) and the fact that for mutually exclusive events the probability of both is the sum of their probabilities, the probability that the "needle" crosses a line is given by

\[
P(\text{cross}) = \int_0^\pi d\theta \int_0^{\frac{L}{2} \sin(\theta) + \frac{W}{2} |\cos(\theta)|} \frac{1}{\pi D} \, dr + \int_0^\pi d\theta \int_{D - \frac{L}{2} \sin(\theta) - \frac{W}{2} |\cos(\theta)|}^D \frac{1}{\pi D} = \frac{2(L + W)}{\pi D} \]

For the values of L and W used in the experiment above, the theoretical probability that the needle will cross a line is 0.3581, only about 1.6% higher than the "experimental value. Using the MATLAB program shown as Listing 10 in the appendix to simulate the experiment gives results such as those shown in Table 1 below:

<table>
<thead>
<tr>
<th># trials</th>
<th>Fraction of Line Crossings in each of three runs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>0.3370 0.3730 0.3510</td>
</tr>
<tr>
<td>10,000</td>
<td>0.3610 0.3530 0.3557</td>
</tr>
<tr>
<td>100,000</td>
<td>0.3555 0.3575 0.3574</td>
</tr>
<tr>
<td>1,000,000</td>
<td>0.3578 0.3576 0.3591</td>
</tr>
<tr>
<td>5,000,000</td>
<td>0.3584 0.3580 0.3580</td>
</tr>
</tbody>
</table>

Table 1

Larger numbers of trials in a simulation give results close to the value determined analytically.

**Conclusion**

The experiments show that even relatively small numbers of trials can be approximately described by theory and illustrate the nature of probability modeling of actual events. The students report that they enjoy "hands-on" projects that relate to the theory being studied.
Computer simulation produces results similar to the experimental observations, indicating that it can be used to model experiments with confidence. The computer also permits quick visualization of the effects of changing the number of trials and allows investigations using much larger numbers of trials than are feasible for a class assignment. Such computer-aided studies allow one to decide when a particular probability model is appropriate in a given situation.

An additional benefit of the computer programs is that students practice using mathematical software used in other courses. I used MATLAB because it is used in a number of courses taken by electrical and computer engineering students at the University of Evansville.

Appendix

MATLAB m-files used to produce quoted results are listed below.

Listing 1 – used to simulate the roll of a single six-sided die.

% Die6Roll.m
function G = Die6Roll(N)
% Simulates the rolling of a six-sided die and returns an N-element array
% of simulated rolls
% Usage: Die6Roll(N)
% First attempt to pick random startup state for rand
rand(‘state’, sum(100*clock))
% Define random array
X=rand(1,N);
% Form array of random integers 1 through 6
G=ceil(6*X);

Listing 2 – used to simulate the roll of an eight-sided die.

% Die8Roll.m
function G = Die8Roll(N)
% Simulates the rolling of an eight-sided die and returns an N-element array
% of simulated rolls
% Usage: Die8Roll(N)
% First attempt to pick random startup state for rand
rand(‘state’, sum(100*clock))
% Define random array
X=rand(1,N);
% Form array of random integers 1 through 6
G=ceil(8*X);

Listing 3 – used to simulate the roll of a ten-sided die.

% Die10Roll.m
function G = Die10Roll(N)
% Simulates the rolling of a ten-sided die and returns an N-element array
% of simulated rolls
% Usage: Die10Roll(N)
% First attempt to pick random startup state for rand
rand(‘state’, sum(100*clock))
% Define random array
X=rand(1,N);
% Form array of random integers 1 through 6
G=ceil(10*X);
function G = Die10Roll(N)
% Simulates the rolling of a ten-sided die and returns an N-element array
% of simulated rolls
% Usage: Die10Roll(N)
% First attempt to pick random startup state for rand
rand('state', sum(100*clock))
% Define random array
X=rand(1,N);
% Form array of random integers 1 through 6
G=floor(10*X);

Listing 4 – used to plot experimental data for five six-sided dice. Roll56 creates the data array Roll56d.

% IDiceEx.m
% Identical Dice Experimental Data
% Run data file
Roll56;
% Compute the sums
S = sum(Roll56d');
% Set up bin centers for histogram
bins = 5:31;
% Get frequency count and bin centers
[n,xout]=hist(S,bins);
% Normalize frequency count as fraction of total tries
n=n/sum(n);
% Create normalized histogram using bar plot
bar(xout,n)
% Add axes labels
xlabel('Sum of Spots')
ylabel('Fraction of Total Tries')
% Change default histogram appearance
h = findobj(gca,'Type','patch');
set(h,'FaceColor','w','EdgeColor','k')
hold on
% Now plot normal pdf for theoretical mu, sigma
x = 5:.1:30;
plot(x,normpdf(x,35/2,sqrt(175/12)))
% Add title
title('Experimental Data for 5 6-sided Dice')
hold off
% Finally compute mean and std dev of experimental data
mean(S)
std(S)

Listing 5 – used to simulate 200 rolls of five identical six-sided dice.
% IDiceSim.m
% Identical Dice Simulation
% Simulate 1000 rolls of dice
Sim = Die6Roll(1000);
% Reshape array
SimRoll = reshape(Sim,5,1000/5);
% Compute the sums
S = sum(SimRoll);
% Set up bin centers for histogram
bins = 5:31;
% Get frequency count and bin centers
[n,xout]=hist(S,bins);
% Normalize frequency count as fraction of total tries
n=n/sum(n);
% Create normalized histogram using bar plot
bar(xout,n)
% Add axes labels
xlabel('Sum of Spots')
ylabel('Fraction of Total Tries')
% Change default histogram appearance
h = findobj(gca,'Type','patch');
set(h,'FaceColor','w','EdgeColor','k')
hold on
% Now plot normal pdf for theoretical mu, sigma
x = 5:.1:30;
plot(x,normpdf(x,35/2,sqrt(175/12)))
% Add title
title('Simulation for 5 6-sided Dice')
hold off
% Finally compute mean and std dev of simulated data
mean(S)
std(S)

Listing 6 – used to simulate 200,000 rolls of five six-sided dice.

% IDiceSim200K.m
% Identical Dice Simulation
% Simulate 1000000 rolls of dice
Sim = Die6Roll(1000000);
% Reshape array
SimRoll = reshape(Sim,5,1000000/5);
% Compute the sums
S = sum(SimRoll);
% Set up bin centers for histogram
bins = 5:31;
% Get frequency count and bin centers
[n,xout]=hist(S,bins);
% Normalize frequency count as fraction of total tries
n=n/sum(n);
% Create normalized histogram using bar plot
bar(xout,n)
% Add axes labels
xlabel('Sum of Spots')
ylabel('Fraction of Total Tries')
% Change default histogram appearance
h = findobj(gca,'Type','patch');
set(h,'FaceColor','w','EdgeColor','k')
hold on
% Now plot normal pdf for theoretical mu, sigma
x = 5:.1:30;
plot(x,normpdf(x,35/2,sqrt(175/12)))
% Add title
title('Simulation for 5 6-sided Dice')
hold off
% Finally compute mean and std dev of simulated data
mean(S)
std(S)

Listing 7 – used to plot experimental data for six non-identical dice, two six-sided dice, two eigh-sided dice, and two ten-sided dice. Roll6mix creates the data array Roll6mixd

% NonIDiceEx.m
% Non-Identical Dice Experimental Data
% Run data file
Roll6mix;
% Compute the sums
S = sum(Roll6mixd');
% Set up bin centers for histogram
bins = 3:47;
% Get frequency count and bin centers
[n,xout]=hist(S,bins);
% Normalize frequency count as fraction of total tries
n=n/sum(n);
% Create normalized histogram using bar plot
bar(xout,n)
% Add axes labels
xlabel('Sum of Spots')
ylabel('Fraction of Total Tries')
% Change default histogram appearance
h = findobj(gca,'Type','patch');
set(h,'FaceColor','w','EdgeColor','k')
hold on
% Now plot normal pdf for theoretical mu, sigma
x = 3:.1:47;
plot(x,normpdf(x,25,sqrt(197/6)))
% Add title
title('Experimental Data for 6 Non-identical Dice')
hold off
% Finally compute mean and std dev of experimental data
mean(S)
std(S)

Listing 8 – used to simulate rolling the six non-identical dice 200 times.

% NonIDiceSim.m
% Non-Identical Dice Simulation
% Simulate 200 rolls of dice
R1 = Die6Roll(400);
R1 = reshape(R1,2,200);
R2 = Die8Roll(400);
R2 = reshape(R2,2,200);
R3 = Die10Roll(400);
R3 = reshape(R3,2,200);
SimRoll = [R1;R2;R3];
% Compute the sums
S = sum(SimRoll);
% Set up bin centers for histogram
bins = 3:47;
% Get frequency count and bin centers
[n,xout]=hist(S,bins);
% Normalize frequency count as fraction of total tries
n=n/sum(n);
% Create normalized histogram using bar plot
bar(xout,n)
% Add axes labels
xlabel('Sum of Spots')
ylabel('Fraction of Total Tries')
% Change default histogram appearance
h = findobj(gca,'Type','patch');
set(h,'FaceColor','w','EdgeColor','k')
hold on
% Now plot normal pdf for theoretical mu, sigma
x = 3:.1:47;
plot(x,normpdf(x,25,sqrt(197/6)))
% Add title
title('Simulation for 6 Non-Identical Dice')
hold off
% Finally compute mean and std dev of simulated data
mean(S)
std(S)

Listing 9 – used to simulate rolling the six non-identical dice 200,000 times.

% NonIDiceSim200K.m
% Non-Identical Dice Simulation
% Simulate 200000 rolls of dice
R1 = Die6Roll(400000);
R1 = reshape(R1,2,200000);
R2 = Die8Roll(400000);
R2 = reshape(R2,2,200000);
R3 = Die10Roll(400000);
R3 = reshape(R3,2,200000);
SimRoll = [R1;R2;R3];
% Compute the sums
S = sum(SimRoll);
% Set up bin centers for histogram
bins = 3:47;
% Get frequency count and bin centers
[n,xout]=hist(S,bins);
% Normalize frequency count as fraction of total tries
n=n/sum(n);
% Create normalized histogram using bar plot
bar(xout,n)
% Add axes labels
xlabel('Sum of Spots')
ylabel('Fraction of Total Tries')
% Change default histogram appearance
h = findobj(gca,'Type','patch');
set(h,'FaceColor','w','EdgeColor','k')
hold on
% Now plot normal pdf for theoretical mu, sigma
x = 3:.1:47;
plot(x,normpdf(x,25,sqrt(197/6)))
% Add title
title('Simulation for 6 Non-Identical Dice')
hold off
% Finally compute mean and std dev of simulated data
mean(S)
std(S)

Listing 10 – used to simulate the Buffon's Needle N times, N entered as part of command.
% Buffon.m
% This file implements a function to simulate a Buffon's Needle
% experiment using a "needle" of rectangular profile.
% D is the distance between lines.
% L is the length of the "needle".
% W is the width of the "needle".
% Usage: Buffon(N)
function XProb = Buffon(N)
rand('state', sum(100*clock))
D = 2;
L = 1;
W = .125;
% Define random array
X=rand(1,N);
Y=rand(1,N);
theta = pi*X;
r = D*Y;
lproj = L/2*sin(theta)+W/2*abs(cos(theta));
%ResultL = r < lproj;
%ResultG = r > (D-lproj);
%Result = ResultL | ResultG;
Result = r < lproj|r > (D-lproj);
Result = +Result;
XProb = mean(Result);

Bibliography


   <http://www.findarticles.com/p/articles/mi_qa4015/is_200401/ai_n9383821>
JAMES A. REISING is an Associate Professor of Electrical Engineering at the University of Evansville, Evansville, Indiana, where he has taught since 1980. Prior to that time he was employed by Eagle-Picher Industries at the Miami Research Laboratories and the Electro-Optic Materials Department. He is a senior member of IEEE.