University of California - Los Angeles

ECE 131A

Probability and Statistics

MATLAB PROJECT

Student: Nha Do

UID: 405314461

Professor: Lara Dolecek

TAs: Lev Tauz & Debarnab Mitra

Winter 2021

1 Tossing a Fair and Unfair Dice

a) The simulation of tossing a 5-sided fair dice for t = 10, 50, 100, 500 and 1000 tosses.

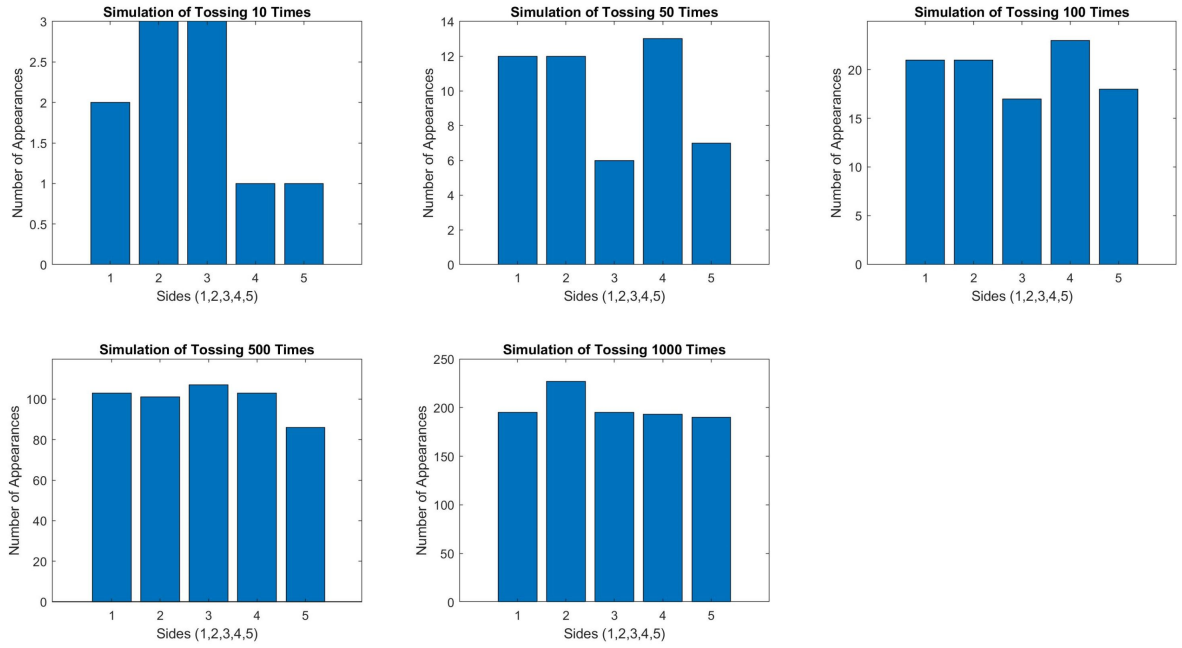

Figure 1: Simulation of Tossing 5-Sided Fair Dice

Based on number of appearances simulated above, we have the estimated probability of odd number below:

t	Odd	Probability(odd)
10	6	0.6
50	25	0.5
100	56	0.56
500	296	0.592
1000	580	0.58

b)

$$\begin{aligned} \text{P(X has odd value)} &= \text{P(X=1)} + \text{P(X=3)} + \text{P(X=5)} \\ &= \frac{1}{5} + \frac{1}{5} + \frac{1}{5} = \frac{3}{5} = 0.6 \end{aligned}$$

c)Since they are random and the real probability cannot be exactly the same as mathematical analysis, yet the results in part (a) are very close to part (b). Hence, they agree with the theoretical results and are acceptable.

d) We must have:

P(X=1) + P(X=2) + P(X=3) + P(X=4) + P(X=5) = 1

Based on the problem, we also have:

$$\begin{cases} 2P(X=1)+3P(X=5)=1 \\ P(X=1)=2P(X=5) \end{cases} \Rightarrow \begin{cases} P(X=1)=\frac{2}{7} \\ P(X=5)=\frac{1}{7} \end{cases}$$

Matlab simulation:

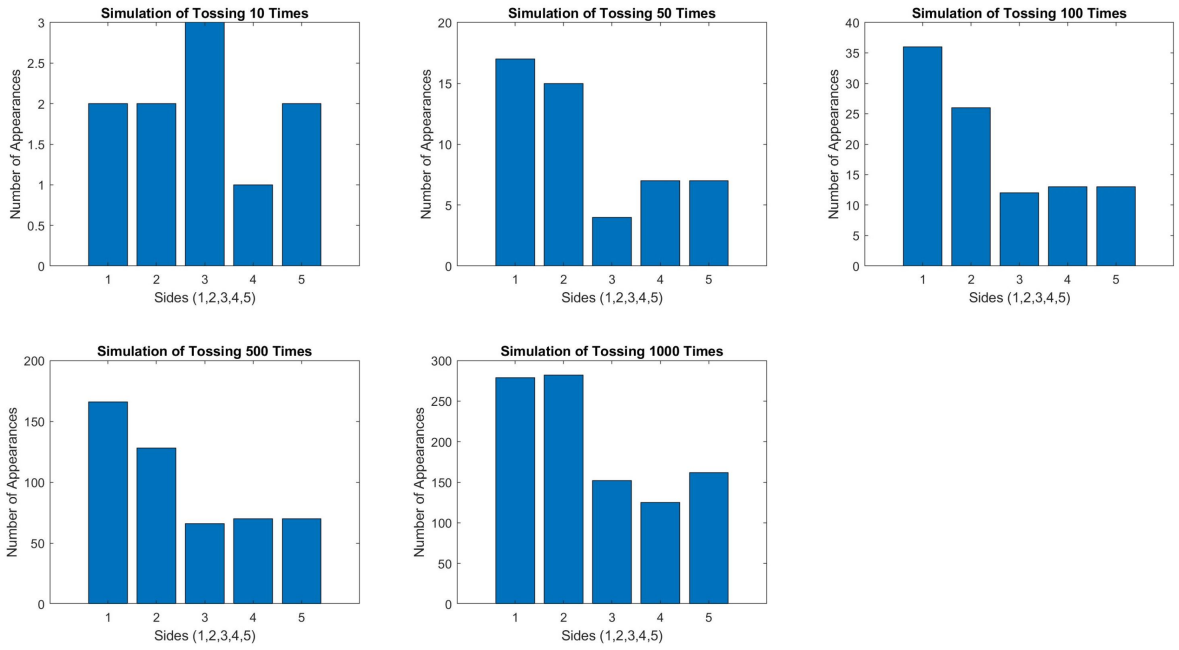

Figure 2: Simulation of Tossing 5-Sided Unfair Dice

Based on number of appearances simulated above, we have the estimated probability of odd number below:

t	Odd	Probability(odd)
10	7	0.7
50	28	0.56
100	61	0.61
500	302	0.604
1000	593	0.593

The exact probability is calculated as:

$$\begin{aligned} \text{P(X has odd value)} &= \text{P(X=1)} + \text{P(X=3)} + \text{P(X=5)} \\ &= \frac{2}{7} + \frac{1}{7} + \frac{1}{7} = \frac{4}{7} = 0.57 \end{aligned}$$

Increase the number of tossing makes the probability closer to mathematical analysis, other are also somewhat close. Hence, they agree with the theoretical results and are acceptable.

2 Coding for BEC

a) Matlab simulation for N-repetition code with N = [3 4 5] and p_e = 0.125:0.025:0.40. For a given N, simulate 100000 instances of sending a bit through the BEC.

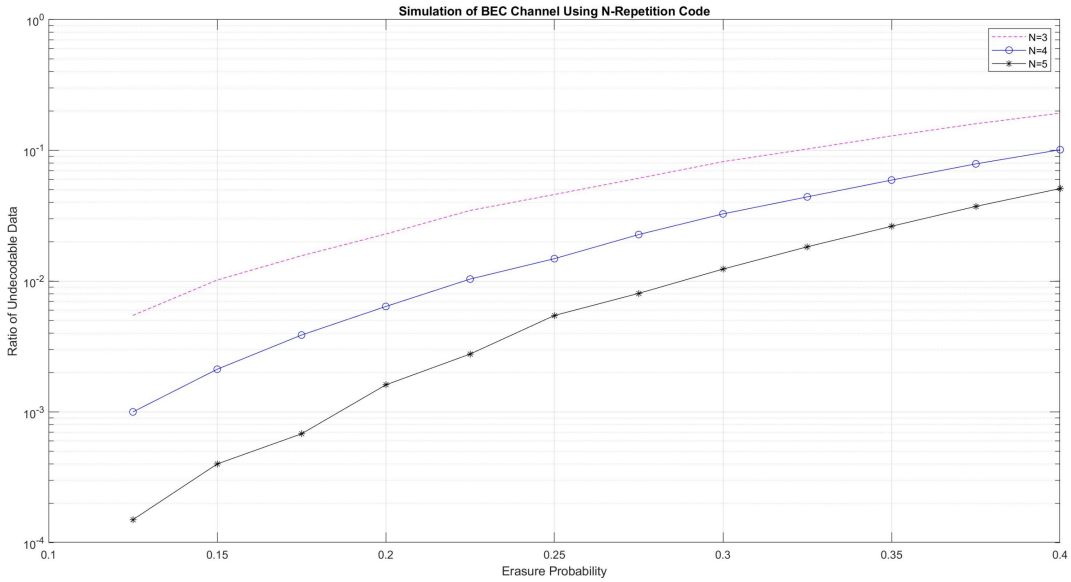

Figure 3: Simulation of BEC Using N-Repetition Code

b) The exact probability in each case is calculated below. It is parameterized by N and p_e:

$$P(undecodable) = Np_e^N$$

Matlab simulation for exact probability:

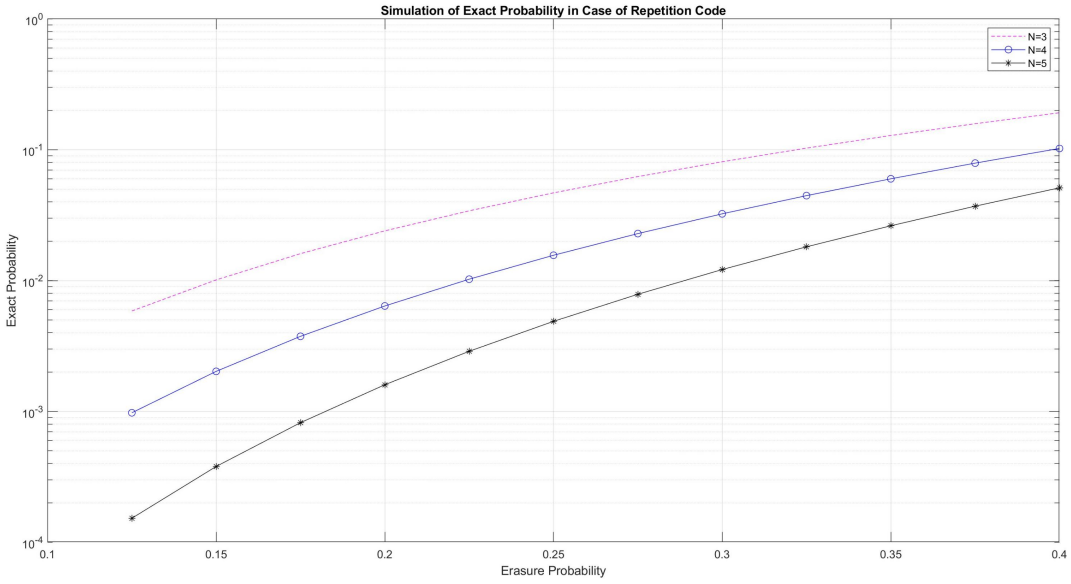

Figure 4: Simulation of Exact Probability for N-Repetition Code

c)Matlab simulation for N-single partiy code with N = [3 4 5] and p_e = 0.125:0.025:0.40. For

a given N, simulate 100000 instances of sending a bit through the BEC.

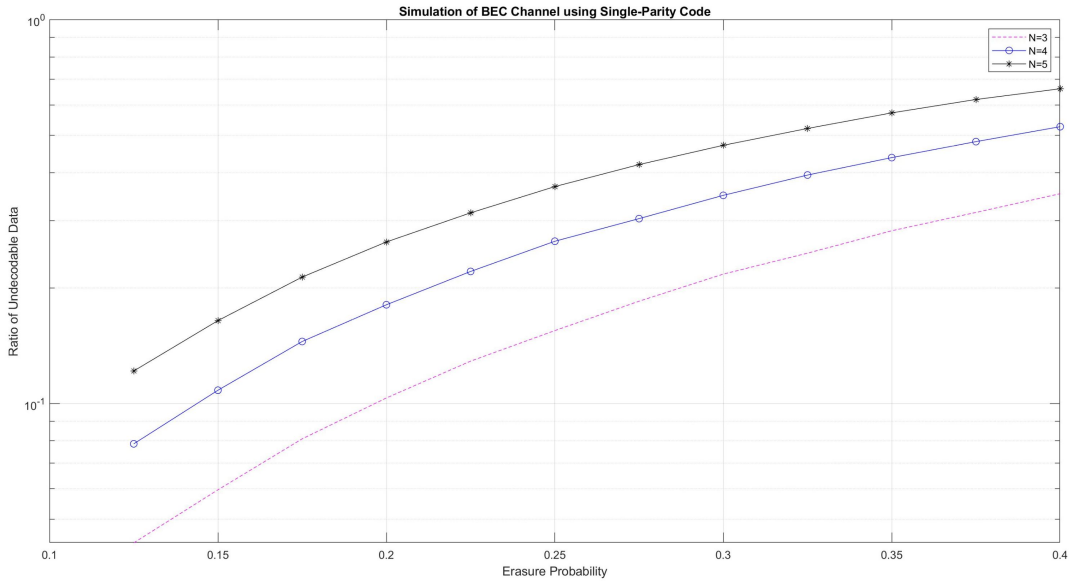

Figure 5: Simulation of BEC Using N-Single-Parity Code

The exact probability in each case is calculated below. It is parameterized by N and p_e:

$$1 - (1 - p_e)^N - N.p_e.(1 - p_e)^{N-1}$$

Matlab simulation for exact probability:

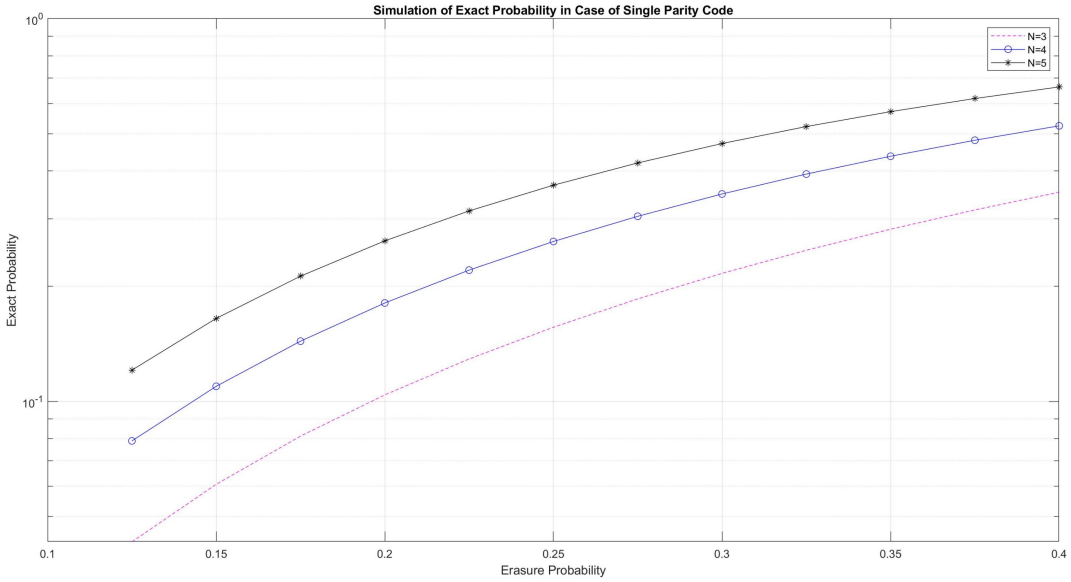

Figure 6: Simulation of Exact Probability for N-Single-Parity Code

d) Compare the plots of N-repetition code and N-single parity code, we can see that the probability of being undecodable in case of N-single parity code is larger. Since, in N-single parity code, a sequence of bit will become undecodable if at least 2 bits are erased. Therefore, when the length of bit sequence becomes larger, the probability of being undecodable also becomes

bigger.
Compare to N-repetition code, we are splitting each bit in the sequence and repeat it N times before transmitting. As long as the received bit sequence has at least 1 bit not erased, we can still receive the correct original bit sequence.

e) When the length of bit sequence is small, such as 1 or 2, we should use N-single parity code. Because the advantage of this kind of encoding is the receiver is capable of reconstructing the signal as long as no more than 1 bit is erased. The speed of this encoding is also faster because we can transmit a whole sequence at one time.
If the length of bit sequence becomes larger, we should use N-repetition code. In this case, the probability of being undecodable is small since the probability of all bits are erased when N large is rare and not likely to happen often. However, for N-repetition code, we have to repeat each bit in a sequence before transmitting, it will slow down the speed. Another disadvantage of N-repetition code compared to N-single parity code is the capable of reconstructing the signal.

3 Naiive Bayes Classifer

a) Bought: 0 for did not buy and 1 for did buy:

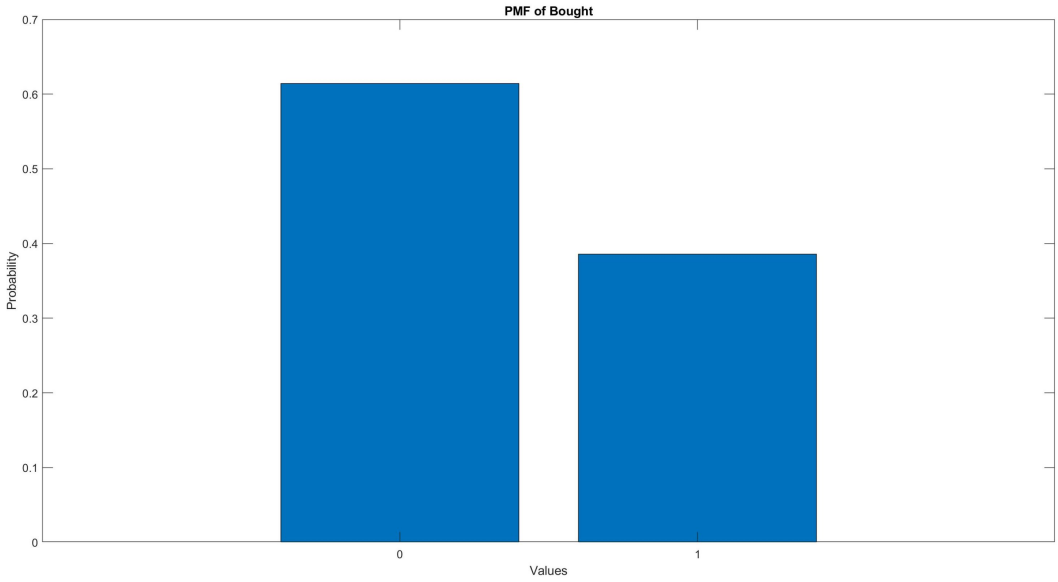

Figure 7: PMF of Bought

Type of spender: 1 for larger spender, 2 for medium spender, 3 for small spender

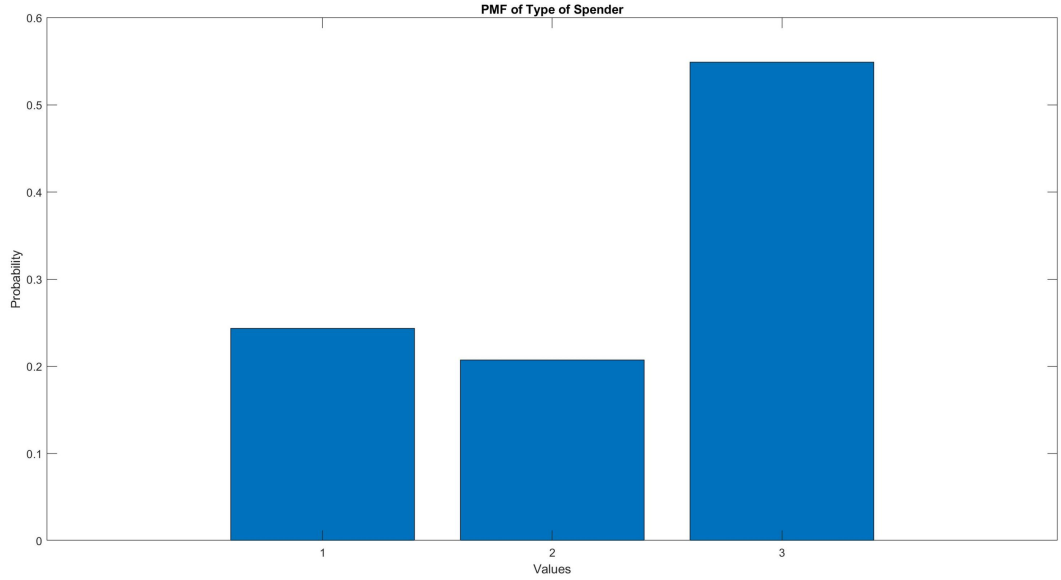

Figure 8: PMF of Type of Spender

Sex of user: 1 for Male and 0 for Female

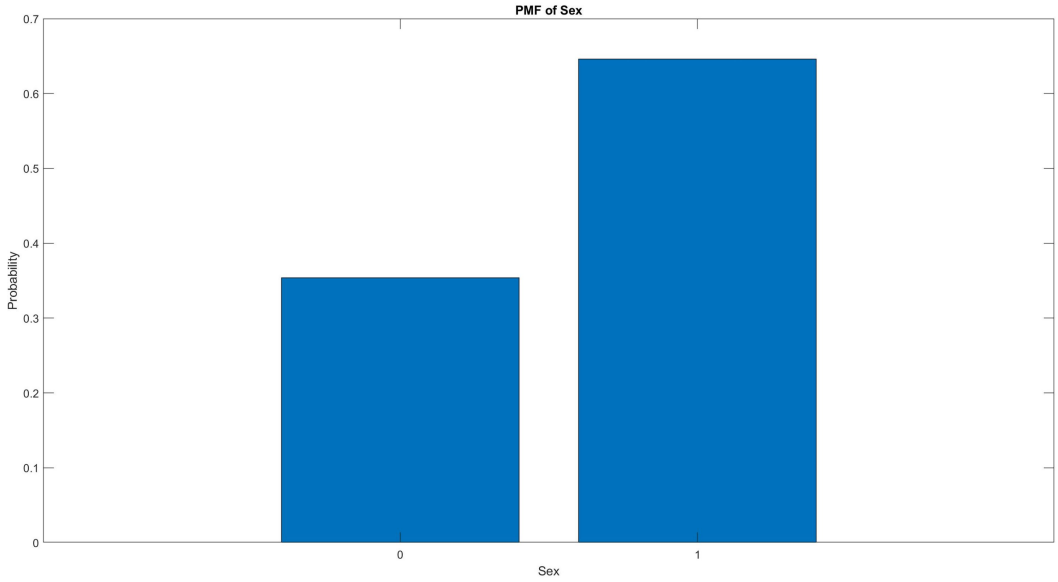

Figure 9: PMF of Sex of User

Age of user:

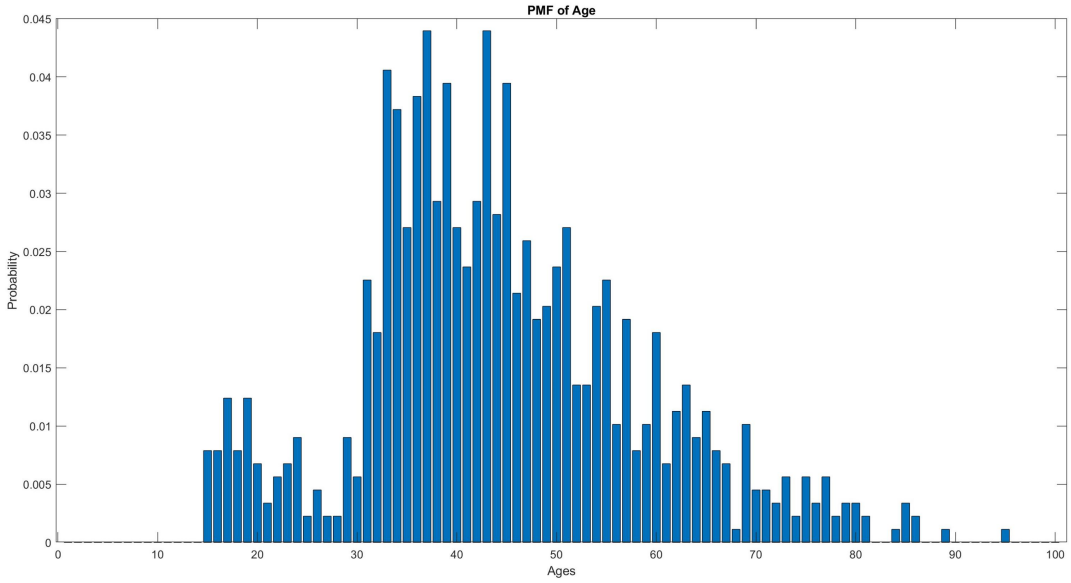

Figure 10: PMF of Age of User

b) PMF of T given B:

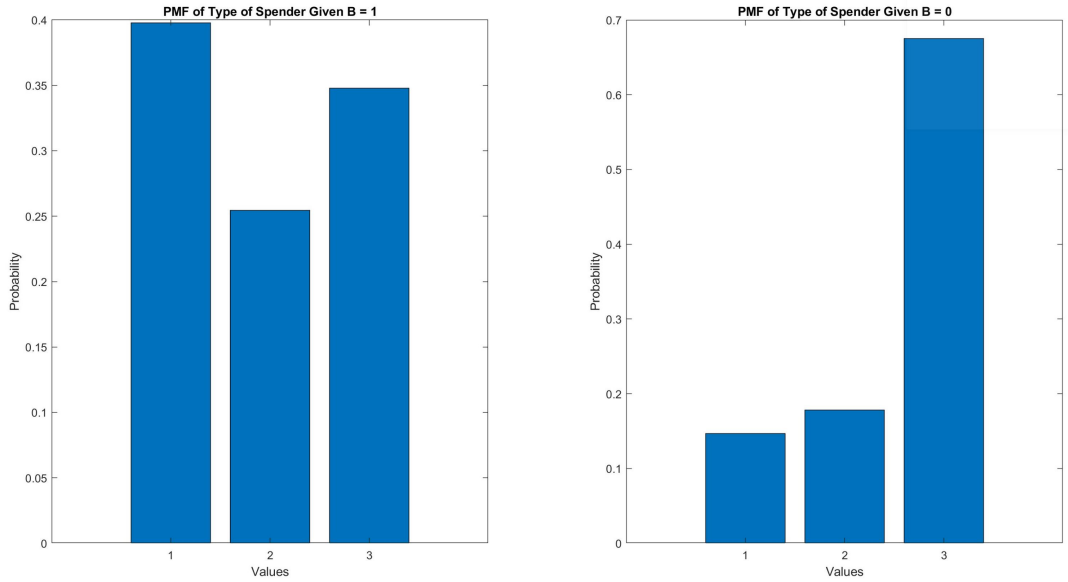

Figure 11: PMF of T given B = 1(left) and B = 0(right)

PMF of S given B:

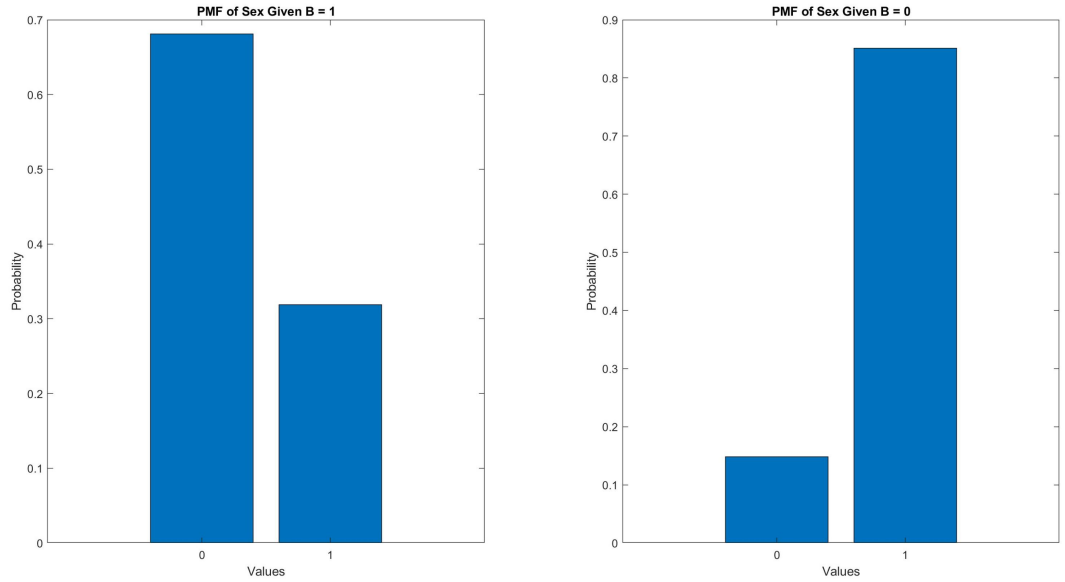

Figure 12: PMF of S given B = 1(left) and B = 0 (right)

PMF of A given B:

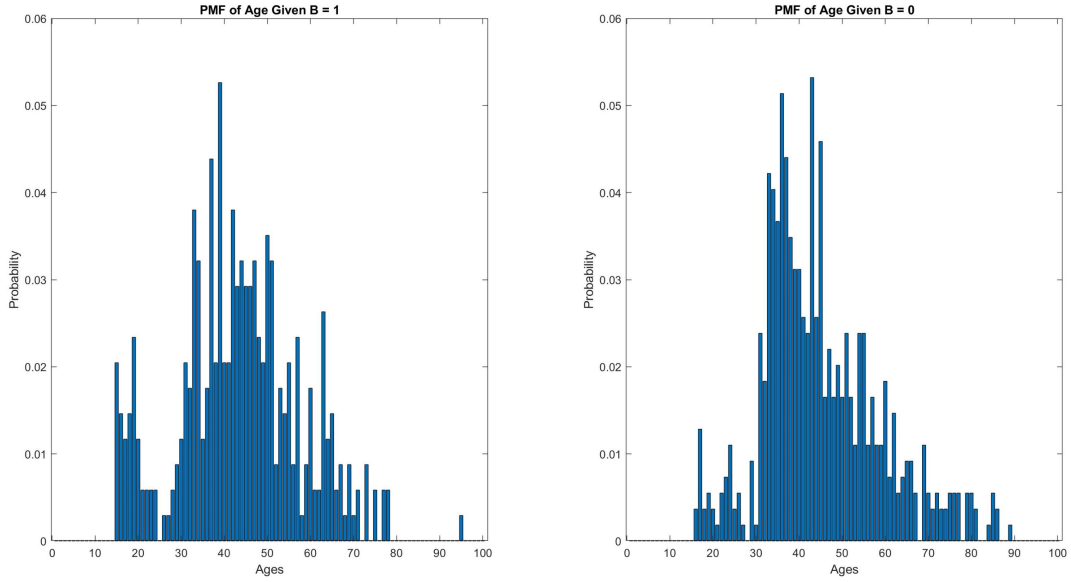

Figure 13: PMF of A given B = 1(left) and B = 0(right)

c)Based on results in question (a) and (b). I wrote the code to generate the values of some conditional probability. The code is attached in Appendix.

$P(T = 1 \mid B = 0)$	0.146789
$P(S = 0 \mid B = 0)$	0.148624
$P(A \leq 55 \mid B = 0)$	0.7963303
$P(T = 1 \mid B = 1)$	0.397661
$P(S = 0 \mid B = 1)$	0.681287
$P(A \leq 55 \mid B = 1)$	0.8099415

Hence:

$$P(T = 1, S = 0, A \leq 55 \mid B = 0) = P(T = 1 \mid B = 0).P(S = 0 \mid B = 0).P(A \leq 55 \mid B = 0)$$
$$= (0.146789).(0.148624).(0.7963303) = 0.01737$$
$$P(T = 1, S = 0, A \leq 55 \mid B = 1) = P(T = 1 \mid B = 1).P(S = 0 \mid B = 1).P(A \leq 55 \mid B = 1)$$
$$= (0.397661).(0.681287).(0.8099415) = 0.21943$$
$$P(B = 0, T = 1, S = 0, A \leq 55) = P(T = 1, S = 0, A \leq 55 \mid B = 0).P(B = 0)$$
$$= (0.01737).(0.6144307) = 0.01067$$
$$P(B = 1, T = 1, S = 0, A \leq 55) = P(T = 1, S = 0, A \leq 55 \mid B = 1).P(B = 1)$$
$$= (0.21943).(0.3855693) = 0.08461$$

d)

$$\begin{aligned} P(T=1,S=0,A\leq 55) &= P(T=1,S=0,A\leq 55 \mid B=0).P(B=0) \\ &\quad + P(T=1,S=0,A\leq 55 \mid B=1).P(B=1) \\ &= (0.01737).(0.6144307) + (0.21943).(0.3855693) = 0.095 \end{aligned}$$

Hence, using Bayes rule to make prediction:

$$P(B=0 \mid T=1,S=0,A\leq 55) = \frac{P(T=1,S=0,A\leq 55|B=0).P(B=0)}{P(T=1,S=0,A\leq 55)} = \frac{(0.01737).(0.6144307)}{0.095} = 0.11$$

$$P(B=1 \mid T=1,S=0,A\leq 55) = \frac{P(T=1,S=0,A\leq 55|B=1).P(B=1)}{P(T=1,S=0,A\leq 55)} = \frac{(0.21943).(0.3855693)}{0.095} = 0.89$$

Therefore, a female whose age is below 55 and who is a large spender will likely buy this product with probability of 0.89.

4 Central Limit Theorem

a) X_i be a uniform continuous random variable in the interval (3,7). The result below is generated by Matlab with n = 1,2,3,10,30,100 and number of samples is 10000.

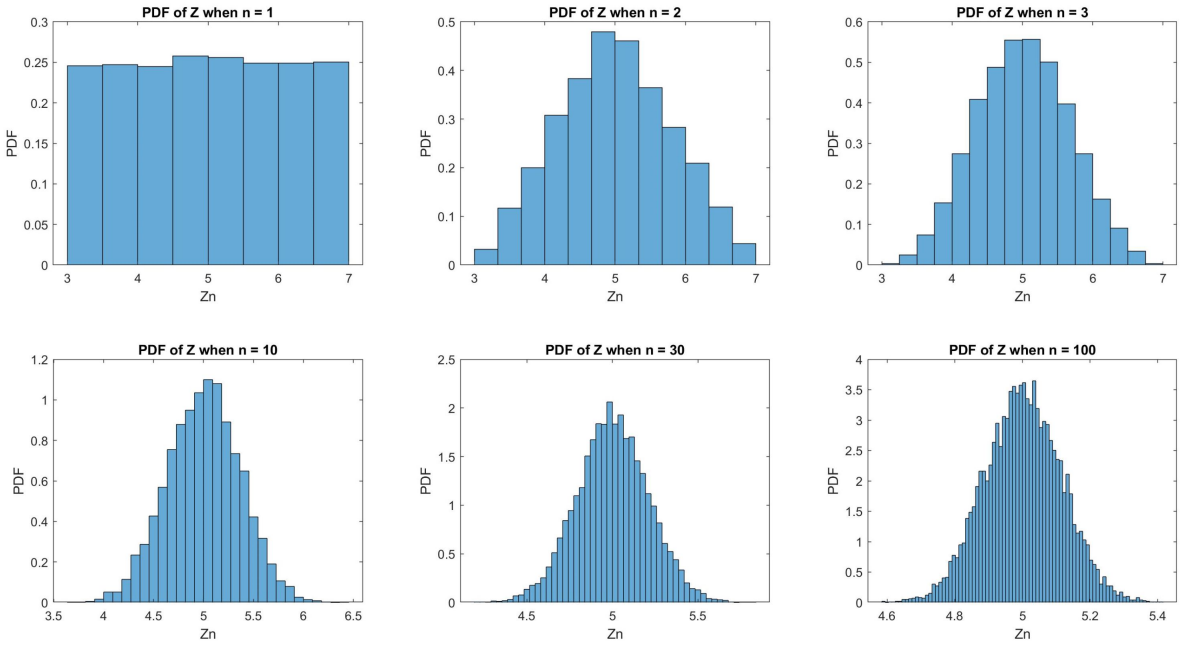

Figure 14: PDF of Z in different n

Observation: The larger n, the closer shape of PDF to the normal distribution (the bell curve).

b) Since Xi is a uniform continuous random variable in the interval (3,7), so:

$$E[X_i] = \frac{3+7}{2} = 5$$

$$VAR(X_i) = \frac{a^2+ab+b^2}{3} - \frac{(a+b)^2}{4} = \frac{9+21+49}{3} - \frac{10^2}{4} = \frac{4}{3}$$

$$\Rightarrow E[Z_n] = \frac{E[X_1]+E[X_2]+...+E[X_n]}{n} = \frac{nE[X_1]}{n} = E[X_1] = 5$$

$$\Rightarrow VAR(Z_n) = \frac{\sigma^2}{n} = \frac{4}{3n}$$

c) Based on mean and variance of Z in question (b), I generated corresponding Gaussian distribution on each plots in (a) using normpdf. When n is high enough, we see that it comes closer to the Gaussian distribution.

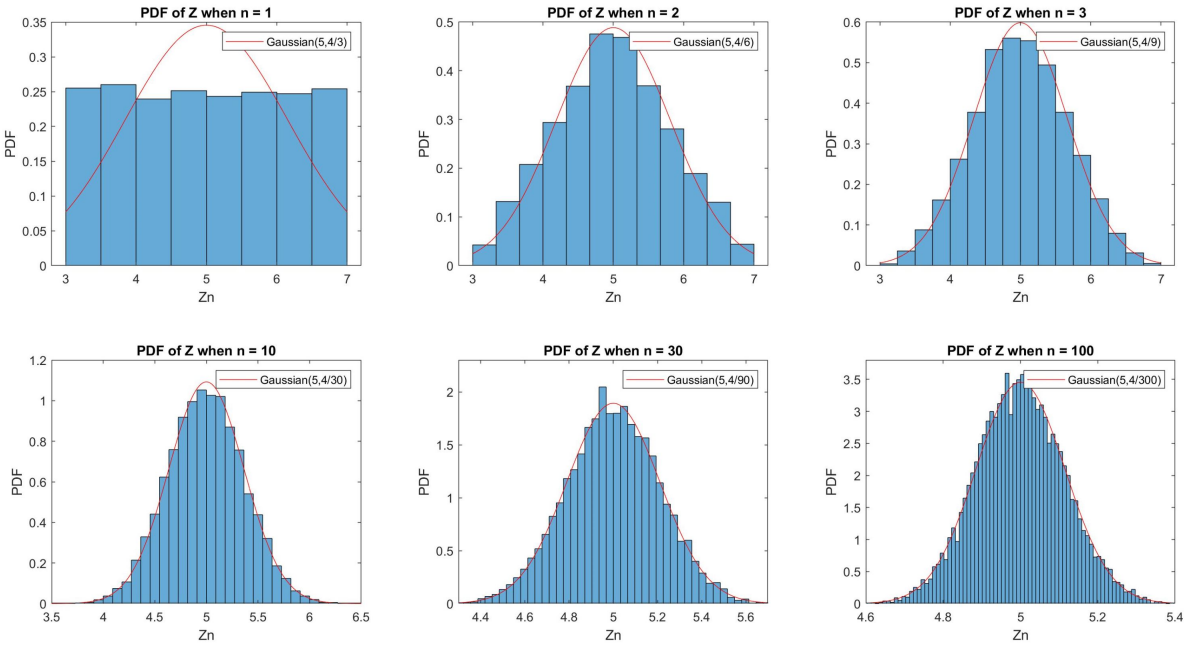

Figure 15: Superimpose Gaussian distribution on the previous plots

d) Now, Xi representing a toss of 5-sided unfair dice in question 1, number of samples = 10000. Again, we have:

$$P(X = 1) = \frac{2}{7}, P(X = 5) = \frac{1}{7}$$

X	1	2	3	4	5
P(X)	2/7	2/7	1/7	1/7	1/7

$$E[X] = \frac{2}{7} + \frac{4}{7} + \frac{3}{7} + \frac{4}{7} + \frac{5}{7} = \frac{18}{7}$$

$$E[X^2] = \frac{2}{7} + \frac{8}{7} + \frac{9}{7} + \frac{16}{7} + \frac{25}{7} = \frac{60}{7}$$

$$VAR(X) = E[X^2] - (E[X])^2 = \frac{60}{7} - (\frac{18}{7})^2 = \frac{96}{49}$$

$$\Rightarrow E[Z_n] = E[X] = \frac{18}{7}$$

$$\Rightarrow VAR(Z_n) = \frac{VAR(X)}{n} = \frac{96}{49n}$$

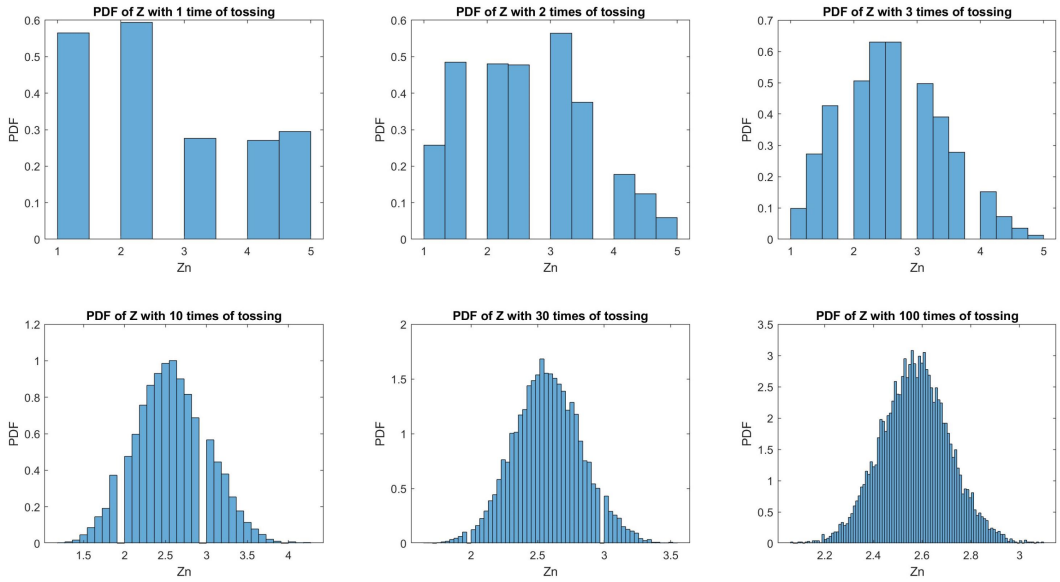

Figure 16: Histogram of Z with different n

Observation: Again, although the dice is unfair, when the number of tossing increases, the distribution comes closer to the normal distribution.

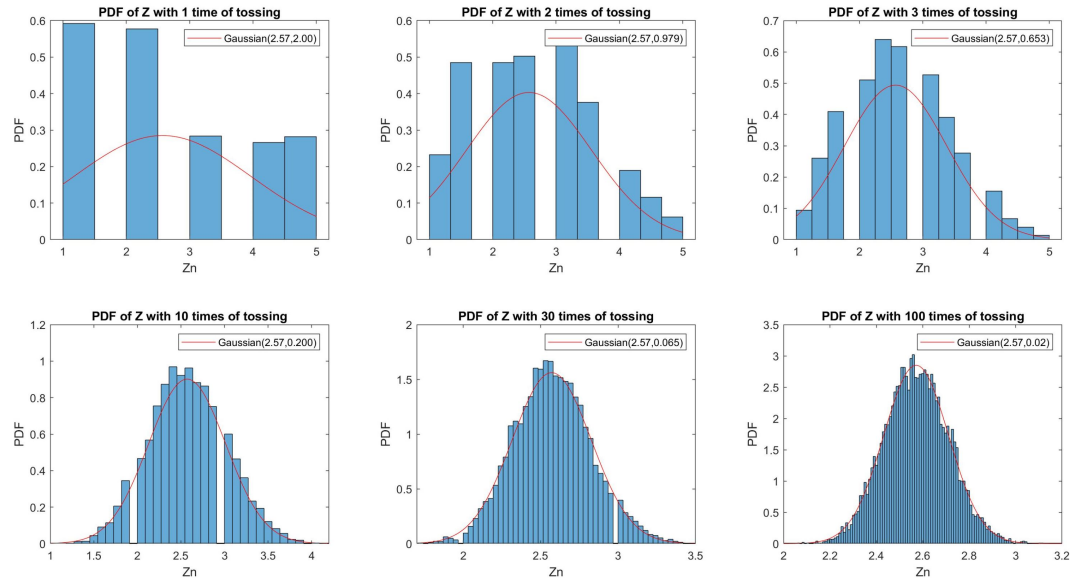

Figure 17: Superimpose Gaussian distribution on the previous plots

APPENDIX

MATLAB CODE

Question1a

Tossing Dice Function:

```
%This function is to simulate the tossing of 5-sided die by breaking down
%rand(0,1) into 5 equal interval and assign them to their respective side.
function [side] = tossing_die

result = rand();

if(result ≤ 0.20)
    side = 1;
elseif(result ≤ 0.40)
    side = 2;
elseif(result ≤ 0.60)
    side = 3;
elseif(result ≤ 0.80)
    side = 4;
else
    side = 5;
end

end
```

Main Function:

```
%Code Written by Nha Do
%The program is to simulate the event of tossing 5-sided fair dice
%n = 10, 50, 100, 500 and 1000

clc;
clear all;
```

```
7
8  result1 = zeros(1,5);
9  %Tossing 10 times
10 for t=1:10
11     side = tossing_die();
12     result1(side) = result1(side) + 1;
13 end
14
15 subplot(2,3,1);
16 bar(result1);
17 title("Simulation of Tossing 10 Times");
18 xlabel("Sides (1,2,3,4,5)");
19 ylabel("Number of Appearances");
20
21 result2 = zeros(1,5);
22 %Tossing 50 times
23 for t=1:50
24     side = tossing_die();
25     result2(side) = result2(side) + 1;
26 end
27
28 subplot(2,3,2);
29 bar(result2);
30 title("Simulation of Tossing 50 Times");
31 xlabel("Sides (1,2,3,4,5)");
32 ylabel("Number of Appearances");
33
34 result3 = zeros(1,5);
35 %Tossing 100 times
36 for t=1:100
37     side = tossing_die();
38     result3(side) = result3(side) + 1;
39 end
40
41 subplot(2,3,3);
42 bar(result3);
43 title("Simulation of Tossing 100 Times");
```

```
44 xlabel("Sides (1,2,3,4,5)");
45 ylabel("Number of Appearances");
46
47 result4 = zeros(1,5);
48 %Tossing 500 times
49 for t=1:500
50     side = tossing_die();
51     result4(side) = result4(side) + 1;
52 end
53
54 subplot(2,3,4);
55 bar(result4);
56 title("Simulation of Tossing 500 Times");
57 xlabel("Sides (1,2,3,4,5)");
58 ylabel("Number of Appearances");
59
60 result5 = zeros(1,5);
61 %Tossing 1000 times
62 for t=1:1000
63     side = tossing_die();
64     result5(side) = result5(side) + 1;
65 end
66
67 subplot(2,3,5);
68 bar(result5);
69 title("Simulation of Tossing 1000 Times");
70 xlabel("Sides (1,2,3,4,5)");
71 ylabel("Number of Appearances");
```

Question1d

Tossing Dice Function:

```
1 %This function is to simulate the tossing of 5-sided die by breaking down
2 %rand(0,1) into 5 non-equal interval and assign them to their respective ...
```

```
    side.
%This function is created for question 1d
function [side] = tossing_die_1d

result = rand();

if(result ≤ 2/7)
    side = 1;
elseif(result ≤ 4/7)
    side = 2;
elseif(result ≤ 5/7)
    side = 3;
elseif(result ≤ 6/7)
    side = 4;
else
    side = 5;
end

end
```

Main Function:

```
%Code Written by Nha Do
%The program is to simulate the event of tossing 5-sided unfair dice
%n = 10, 50, 100, 500 and 1000

clc;
clear all;

result1 = zeros(1,5);
%Tossing 10 times
for t=1:10
    side = tossing_die_1d();
    result1(side) = result1(side) + 1;
end

```

```
15 subplot(2,3,1);
16 bar(result1);
17 title("Simulation of Tossing 10 Times");
18 xlabel("Sides (1,2,3,4,5)");
19 ylabel("Number of Appearances");
20
21 result2 = zeros(1,5);
22 %Tossing 50 times
23 for t=1:50
24     side = tossing_die_1d();
25     result2(side) = result2(side) + 1;
26 end
27
28 subplot(2,3,2);
29 bar(result2);
30 title("Simulation of Tossing 50 Times");
31 xlabel("Sides (1,2,3,4,5)");
32 ylabel("Number of Appearances");
33
34 result3 = zeros(1,5);
35 %Tossing 100 times
36 for t=1:100
37     side = tossing_die_1d();
38     result3(side) = result3(side) + 1;
39 end
40
41 subplot(2,3,3);
42 bar(result3);
43 title("Simulation of Tossing 100 Times");
44 xlabel("Sides (1,2,3,4,5)");
45 ylabel("Number of Appearances");
46
47 result4 = zeros(1,5);
48 %Tossing 500 times
49 for t=1:500
50     side = tossing_die_1d();
51     result4(side) = result4(side) + 1;
```

```
end

subplot(2,3,4);
bar(result4);
title("Simulation of Tossing 500 Times");
xlabel("Sides (1,2,3,4,5)");
ylabel("Number of Appearances");

result5 = zeros(1,5);
%Tossing 1000 times
for t=1:1000
    side = tossing_die_1d();
    result5(side) = result5(side) + 1;
end

subplot(2,3,5);
bar(result5);
title("Simulation of Tossing 1000 Times");
xlabel("Sides (1,2,3,4,5)");
ylabel("Number of Appearances");
```

Question2a

Encoding N-Repetition Code:

```
%This function is to encode the input bit using N-repetition code
%The output bit will be a sequence in which the input bit is repeated N
%times

function bitSequence = encoding(bit, N)

bitSequence = ones(1,N);

bitSequence = bit.*bitSequence;

```

```
11 end
```

Decoding N-Repetition Code:

```
1  %This function will decode the encoded sequence bit and output
2  %respective value of bit
3
4  function outputBit = decoding(sequence)
5
6  countBitOne = 0;
7  countBitZero = 0;
8  countError = 0;
9
10 for i=1:length(sequence)
11     if(sequence(i) == 1)
12         countBitOne = countBitOne + 1;
13     elseif(sequence(i) == 0)
14         countBitZero = countBitZero + 1;
15     else
16         countError = countError + 1;
17     end
18 end
19
20 if(countBitOne ≥ 1)
21     outputBit = 1;
22 elseif(countBitZero ≥ 1)
23     outputBit = 0;
24 else
25     outputBit = 'The encoded bit is undecodable';
26 end
27
28 end
```

Transmission (Channel Simulation):

```
1  %This function simulates a BEC channel
```

```
2
3  function sendingBit = transmission(input, probability)
4
5  for k=1:length(input)
6      result = rand();
7
8      if(result < probability)
9          input(k) = 2; %Assume that when bit is erased, it will be 2
10     end
11 end
12
13 sendingBit = input;
14 end
```

Main Function:

```
1  %Code Written by Nha Do
2  %This program simulates the transmission of a sequence of bit using
3  %N-repetition code
4
5  clc;
6  clear all;
7  N = [3,4,5]; %Number of N
8  probability = 0.125:0.025:0.40; %pe
9  error = zeros(length(N),length(probability));
10 P_rep = zeros(length(N),length(probability));
11 t = 100000; %sample
12 for n=1:3
13     for i=1:length(probability)
14         for j=1:t
15             input_rep = randi(2,N(n),1)-1; %Generate a sequence of 0 and 1
16             for k=1:length(input_rep)
17                 bitSequence = encoding(input_rep(k),N(n)); %Encoding ...
                       input bit as a sequence of 1 or 0
18                 sendingBit = transmission(bitSequence, probability(i)); ...
                       %Transmit data
```

```
                outputBit = decoding(sendingBit); %Decoding
                if(outputBit ≠ input_rep(k))
                    error(n,i) = error(n,i) + 1; %Calculate error
                    P_rep(n,i) = error(n,i)/100000; %Probability of ...
                        undecodable
                end
            end
        end
    end
end

semilogy(probability, P_rep(1,:),'m--',probability, P_rep(2,:),'-bo', ...
    probability, P_rep(3,:),'-k*');
grid on;
hold on;
title("Simulation of BEC Channel");
xlabel("Erasure Probability");
ylabel("Ratio of Undecodable Data");
legend('N=3', 'N=4', 'N=5');
```

Question2b

Exact Probability:

```
%Code Written by Nha Do
%This program generates exact probability of N-repetition code
%Used to compare with question 2a

clc;
pe = 0.125:0.025:0.40; %Erasure probability
P_exact_3 = zeros(1,length(pe));
P_exact_4 = zeros(1,length(pe));
P_exact_5 = zeros(1,length(pe));
for i=1:length(pe)
    P_exact_3(1,i) = 3*(pe(i).^3);
```

```
    P_exact_4(1,i) = 4*(pe(i).^4);
    P_exact_5(1,i) = 5*(pe(i).^5);
end

semilogy(pe, P_exact_3,'m--', pe, P_exact_4,'-bo', pe, P_exact_5,'-k*');
grid on
hold on
title("Simulation of Exact Probability in Case of Repetition Code");
xlabel("Erasure Probability");
ylabel("Exact Probability");
legend('N=3', 'N=4', 'N=5');
```

Question2c

Encoding N-Single Parity Code:

```
%This function simulates encoding a bit sequence using N single parity code

function encodedSequence = encoding_single_parity(bitSequence)
sum = 0;

for i=1:length(bitSequence)
    sum = sum + bitSequence(i);
end

appendedBit = mod(sum,2); %Calculate appended bit
encodedSequence = [bitSequence appendedBit]; %Append the calculated bit ...
    at the end of sequence
end
```

Decoding N-Single Parity Code:

```
%This function simulates decoding single parity code

function  decodedSequence = decoding_single_parity(bitSequence)
```

```
count = 0;
l = length(bitSequence);
sum1 = 0;
sum2 = 0;
originalSequence = zeros(1,l-1);
for i=1:l
   if(bitSequence(i) == 2) %%Assume that when bit is earased, it will be 2
       temp = i;
       count = count + 1;
   end
end

%Note that N-single-parity code can only reconstruct 1 erasured bit
if(count == 1)
    %Find the position of erasured bit and reconstruct
    if(bitSequence(l) == 2)
        for k=1:length(bitSequence)-1
            originalSequence(k) = bitSequence(k);
        end
        decodedSequence = originalSequence;
    else
        for m=1:temp-1
            sum1 = sum1 + bitSequence(m);
        end

        for n=temp+1:l-1
            sum2 = sum2 + bitSequence(n);
        end
        bitSequence(temp) = mod(bitSequence(l)+sum1+sum2,2);
        tempSequence = bitSequence;

        for k=1:length(tempSequence)-1
            originalSequence(k) = tempSequence(k);
        end
        decodedSequence = originalSequence;
    end
elseif(count == 0) %If there is no erasued bit, output the same sequence ...
```

```
    except the last bit
    for k=1:length(bitSequence)-1
        originalSequence(k) = bitSequence(k);
    end
        decodedSequence = originalSequence;
else %If more than 1 bit are erasured, it will be undecodable
    decodedSequence = -1;
end

end
```

Main Function:

```
%Code Written by Nha Do
%This program simulates the transmission of a sequence of bit using
%N-single parity code

clc;
clear all;
N = [3,4,5];
probability = 0.125:0.025:0.40; %Erasure Probability
error = zeros(length(N),length(probability));
P_sin = zeros(length(N),length(probability));
t = 100000; %sample

for n=1:length(N)
    for i=1:length(probability)
        for j=1:t
            input = randi(2,N(n)-1,1)-1; %Generate a sequence of 0 and 1
            encodedSequence = encoding_single_parity(input'); %Encoding input
            sendingBit = transmission(encodedSequence, probability(i)); ...
                %Transmit data
            decodedSequence = decoding_single_parity(sendingBit); %Decoding
            if(decodedSequence == -1)
                error(n,i) = error(n,i) + 1; %Calculate error
                P_sin(n,i) = error(n,i)/100000; %Probability of undecodable
```

```
            end
        end
    end
end

semilogy(probability, P_sin(1,:),'m--',probability, P_sin(2,:),'-bo', ...
    probability, P_sin(3,:),'-k*');
grid on;
hold on;
title("Simulation of BEC Channel using Single-Parity Code");
xlabel("Erasure Probability");
ylabel("Ratio of Undecodable Data");
legend('N=3', 'N=4', 'N=5');
```

Exact Probability for N-Single Parity Code:

```
%Code Written by Nha Do
%This program generates exact probability of N-Single-Parity code

clc;
clear all;
pe = 0.125:0.025:0.40; %Erasure Probability
P_sin_exact_3 = zeros(1,length(pe));
P_sin_exact_4 = zeros(1,length(pe));
P_sin_exact_5 = zeros(1,length(pe));
for i=1:length(pe)
    P_sin_exact_3(1,i)= 1 - (1-pe(i)).^3 - 3*pe(i)*(1-pe(i)).^2;
    P_sin_exact_4(1,i)= 1 - (1-pe(i)).^4 - 4*pe(i)*(1-pe(i)).^3;
    P_sin_exact_5(1,i)= 1 - (1-pe(i)).^5 - 5*pe(i)*(1-pe(i)).^4;
end

semilogy(pe, P_sin_exact_3,'m--', pe, P_sin_exact_4,'-bo', pe, ...
    P_sin_exact_5,'-k*');
grid on
hold on
title("Simulation of Exact Probability in Case of Single Parity Code");
```

```
xlabel("Erasure Probability");
ylabel("Exact Probability");
legend('N=3', 'N=4', 'N=5');
```

Question3a

PMF of Bought and print out Number and Probability of 0 and 1:

```
%Code Written by Nha Do
%This program plots the PMF of Bought in user_data.csv

clc;
clear all;
count1 = 0;
count2 = 0;
filename = 'user_data.csv';
bought = readtable(filename,'Range','A1:A888'); %Read the file and store ...
    in bought
for i=1:height(bought)
    if(bought{i,1} == 0)
        count1 = count1 + 1; %Count did not buy
    else
        count2 = count2 + 1; %Count did buy
    end
end

fprintf("Number of 0: %d \n", count1);
fprintf("P(X=0) = %d \n", count1/height(bought));
fprintf("Number of 1: %d \n", count2);
fprintf("P(X=1) = %d \n", count2/height(bought));

values = [0 1];
probability = [count1/height(bought) count2/height(bought)];
bar(values, probability);
title("PMF of Bought");
```

```
xlabel("Values");
ylabel("Probability");
```

PMF of Type of Spender and print out Number and Probability of 1,2 and 3:

```
%Code Written by Nha Do
%This program plots the PMF of Type of Spender in user_data.csv

clc;
clear all;
count1 = 0;
count2 = 0;
count3 = 0;
filename = 'user_data.csv';
type_of_spender = readtable(filename,'Range','B1:B888'); %Read the file ...
    and store in type_of_spender
for i=1:height(type_of_spender)
    if(type_of_spender{i,1} == 1)
        count1 = count1 + 1; %Count large spender
    elseif(type_of_spender{i,1} == 2)
        count2 = count2 + 1; %Count medium spender
    else
        count3 = count3 + 1; %Count small spender
    end
end

fprintf("Number of 1: %d \n", count1);
fprintf("Number of 2: %d \n", count2);
fprintf("Number of 3: %d \n", count3);
fprintf("P(X=1) = %d \n", count1/height(type_of_spender));
fprintf("P(X=2) = %d \n", count2/height(type_of_spender));
fprintf("P(X=3) = %d \n", count3/height(type_of_spender));

probability = [count1/height(type_of_spender) ...
    count2/height(type_of_spender) count3/height(type_of_spender)];
bar( probability);
```

```
30 title("PMF of Type of Spender");
31 xlabel("Values");
32 ylabel("Probability");
```

PMF of Sex and print out Number and Probability of 0 and 1:

```
1  %Code Written by Nha Do
2  %This program plots the PMF of Sex of user in user_data.csv
3
4  clc;
5  clear all;
6  count1 = 0;
7  count2 = 0;
8  filename = 'user_data.csv';
9  sex = readtable(filename,'Range','C1:C888'); %Read the file and store in sex
10 for i=1:height(sex)
11     if(sex{i,1} == 0)
12         count1 = count1 + 1; %Count number of female
13     else
14         count2 = count2 + 1; %Count number of male
15     end
16 end
17
18 fprintf("Number of 0: %d \n", count1);
19 fprintf("Number of 1: %d \n", count2);
20 fprintf("P(X=0) = %d \n", count1/height(sex));
21 fprintf("P(X=1) = %d \n", count2/height(sex));
22
23 values = [0 1];
24 probability = [count1/height(sex) count2/height(sex)];
25 bar(values, probability);
26 title("PMF of Sex");
27 xlabel("Sex");
28 ylabel("Probability");
```

PMF of Age and print out Number of people whose age is less than 55:

```
%Code Written by Nha Do
%This program plots the PMF of Age of user in user_data.csv
%floor is used to round down the age

clc;
clear all;
count = zeros(1,100);
probability = zeros(1,100);
result_55 = 0;
filename = 'user_data.csv';
age = readtable(filename,'Range','D1:D888'); %Read the file and store in age
for i=1:height(age)
    for j=1:100
        if(floor(age{i,1}) == j) %Round down age of user
            count(j) = count(j) + 1; %Count age
        end
    end
end

for k=1:length(count)
    probability(k) = count(k)./height(age);
end

for l=1:55
    result_55 = result_55 + count(l);
end

fprintf("Number of people whose age ≤55 is: %d \n", result_55);

bar(probability);
title("PMF of Age");
xlabel("Ages");
ylabel("Probability");
```

Question3b

Conditional PMF for Type of Spender (2 plots):

```
%Code Written by Nha Do
%This program esimates and plots the conditional PMF of T
%given B = 1 and B = 0

clc;
clear all;
count1 = 0;
count2 = 0;
count3 = 0;
count4 = 0;
count5 = 0;
count6 = 0;
filename = 'user_data.csv';
bought = readtable(filename,'Range','A1:A888');
type_of_spender = readtable(filename,'Range','B1:B888');
for i=1:height(bought)
    if(bought{i,1} == 1)
        if(type_of_spender{i,1} == 1)
            count1 = count1 + 1; %Large spender bought product
        elseif(type_of_spender{i,1} == 2)
            count2 = count2 + 1; %Medium spender bought product
        else
            count3 = count3 + 1; %Small spender bought product
        end
    else
        if(type_of_spender{i,1} == 1)
            count4 = count4 + 1; %Large spender did not buy product
        elseif(type_of_spender{i,1} == 2)
            count5 = count5 + 1; %Medium spender did not buy product
        else
            count6 = count6 + 1; %Small spender did not buy product
        end
```

```
33      end
34  end
35
36  probability_1 = [count1/342 count2/342 count3/342];
37  probability_0 = [count4/545 count5/545 count6/545];
38
39  subplot(1,2,1);
40  bar(probability_1);
41  title("PMF of Type of Spender Given B = 1");
42  xlabel("Values");
43  ylabel("Probability");
44
45  subplot(1,2,2);
46  bar(probability_0);
47  title("PMF of Type of Spender Given B = 0");
48  xlabel("Values");
49  ylabel("Probability");
```

Conditional PMF for Sex (2 plots):

```
1  %Code Written by Nha Do
2  %This program esimates and plots the conditional PMF of S
3  %given B = 1 and B = 0
4
5  clc;
6  clear all;
7  count1 = 0;
8  count2 = 0;
9  count3 = 0;
10 count4 = 0;
11 filename = 'user_data.csv';
12 bought = readtable(filename,'Range','A1:A888');
13 sex = readtable(filename,'Range','C1:C888');
14 for i=1:height(bought)
15     if(bought{i,1} == 1)
16         if(sex{i,1} == 1)
```

```
            count1 = count1 + 1; %Male bought product
        else
            count2 = count2 + 1; %Female bought product
        end
    else
        if(sex{i,1} == 1)
            count3 = count3 + 1; %Male did not buy product
        else
            count4 = count4 + 1; %Female did not buy product
        end
     end
end

values = [1 0];
probability_1 = [count1/342 count2/342];
probability_0 = [count3/545 count4/545];

subplot(1,2,1);
bar(values, probability_1);
title("PMF of Sex Given B = 1");
xlabel("Values");
ylabel("Probability");

subplot(1,2,2);
bar(values, probability_0);
title("PMF of Sex Given B = 0");
xlabel("Values");
ylabel("Probability");
```

Conditional PMF for Age (2 plots) then calculate and print out Probability of people whose age is less than 55 but did buy or did not buy product:

```
%Code Written by Nha Do
%This program esimates and plots the conditional PMF of A
%given B = 1 and B = 0

```

```
clc;
clear all;
count_1 = zeros(1,100);
count_0 = zeros(1,100);
probability_1 = zeros(1,100);
probability_0 = zeros(1,100);
filename = 'user_data.csv';
bought = readtable(filename,'Range','A1:A888');
age = readtable(filename,'Range','D1:D888');
for i=1:height(bought)
    if(bought{i,1} == 1)
        for j=1:100
            if(floor(age{i,1}) == j)
                count_1(j) = count_1(j) + 1; %Age j bought product
            end
        end
    else
        for k=1:100
            if(floor(age{i,1}) == k)
                count_0(k) = count_0(k) + 1; %Age k did not buy product
            end
        end
    end
end

for n=1:length(count_1)
    probability_1(n) = count_1(n)/342;
    probability_0(n) = count_0(n)/545;
end

subplot(1,2,1);
bar(probability_1);
title("PMF of Age Given B = 1");
xlabel("Ages");
ylabel("Probability");

subplot(1,2,2);
```

```
bar(probability_0);
title("PMF of Age Given B = 0");
xlabel("Ages");
ylabel("Probability");

result_0 = 0;
result_1 = 0;
for a=1:55
    result_0 = result_0 + probability_0(a);
    result_1 = result_1 + probability_1(a);
end

fprintf("Probability for age ≤ 55 given B = 0 is %d \n", result_0);
fprintf("Probability for age ≤ 55 given B = 1 is %d\n", result_1);
```

Question4a

PDF of Z_n:

```
%Code Written by Nha Do
%This program plots and compares the PDF in different number of n
%To plot the PDF at n's position, the cumsum is used.
%Z(n) = (X1 + X2 + ... + Xn)/n
clc;
clear all;
N = [1 2 3 10 30 100]; %Create n
%Create lower and upper bound for Xi
lower_bound = 3;
upper_bound = 7;
t = 10000; %Number of samples
X = lower_bound + (upper_bound - lower_bound).*rand(max(N),t); %Generates ...
    Uniform RV
Z = cumsum(X); %Cumulative sum

subplot(2,3,1);
```

```
16 histogram(Z(1,:)./1, 'Normalization', 'pdf', 'Binwidth', 1/2);
17 title("PDF of Z when n = 1");
18 xlabel("Zn");
19 ylabel("PDF");
20
21 subplot(2,3,2);
22 histogram(Z(2,:)./2, 'Normalization', 'pdf', 'Binwidth', 1/3);
23 title("PDF of Z when n = 2");
24 xlabel("Zn");
25 ylabel("PDF");
26
27 subplot(2,3,3);
28 histogram(Z(3,:)./3, 'Normalization', 'pdf', 'Binwidth', 1/4);
29 title("PDF of Z when n = 3");
30 xlabel("Zn");
31 ylabel("PDF");
32
33 subplot(2,3,4);
34 histogram(Z(10,:)./10, 'Normalization', 'pdf', 'Binwidth', 1/11);
35 title("PDF of Z when n = 10");
36 xlabel("Zn");
37 ylabel("PDF");
38
39 subplot(2,3,5);
40 histogram(Z(30,:)./30, 'Normalization', 'pdf', 'Binwidth', 1/31);
41 title("PDF of Z when n = 30");
42 xlabel("Zn");
43 ylabel("PDF");
44
45 subplot(2,3,6);
46 histogram(Z(100,:)./100, 'Normalization', 'pdf', 'Binwidth', 1/101);
47 title("PDF of Z when n = 100");
48 xlabel("Zn");
49 ylabel("PDF");
```

Question4c

Generates and Superimposes Gaussian Distribution:

```
%Code Written by Nha Do
%Generate Gaussian random variable and apply to previous plots
%Mean = 5, Variance = 4/(3n)
clc;
clear all;
N = [1 2 3 10 30 100]; %Create n
%Create lower and upper bound for Xi
lower_bound = 3;
upper_bound = 7;
t = 10000; %Number of samples

X = lower_bound + (upper_bound - lower_bound).*rand(max(N),t); %Generates ...
    Uniform RV
Z = cumsum(X); %Cumulative sum

x = 3:0.001:7; %Create range for Gaussian Distribution

subplot(2,3,1);
h1 = histogram(Z(1,:)./1, 'Normalization', 'pdf', 'Binwidth', 1/2);
h1.Annotation.LegendInformation.IconDisplayStyle = 'off';
title("PDF of Z when n = 1");
xlabel("Zn");
ylabel("PDF");
hold on;
y1 = normpdf(x,5,sqrt(4/3)); %Gaussian Distribution with n = 1;
plot(x,y1,'r');
legend("Gaussian(5,4/3)");

subplot(2,3,2);
h2 = histogram(Z(2,:)./2, 'Normalization', 'pdf', 'Binwidth', 1/3);
h2.Annotation.LegendInformation.IconDisplayStyle = 'off';
title("PDF of Z when n = 2");
```

```
xlabel("Zn");
ylabel("PDF");
hold on;
y2 = normpdf(x,5,sqrt(4/(3*2))); %Gaussian Distribution with n = 2;
plot(x,y2,'r');
legend("Gaussian(5,4/6)");

subplot(2,3,3);
h3 = histogram(Z(3,:)./3, 'Normalization', 'pdf', 'Binwidth', 1/4);
h3.Annotation.LegendInformation.IconDisplayStyle = 'off';
title("PDF of Z when n = 3");
xlabel("Zn");
ylabel("PDF");
hold on;
y3 = normpdf(x,5,sqrt(4/(3*3))); %Gaussian Distribution with n = 3;
plot(x,y3,'r');
legend("Gaussian(5,4/9)");

subplot(2,3,4);
h4 = histogram(Z(10,:)./10, 'Normalization', 'pdf', 'Binwidth', 1/11);
h4.Annotation.LegendInformation.IconDisplayStyle = 'off';
title("PDF of Z when n = 10");
xlabel("Zn");
ylabel("PDF");
hold on;
axis([3.5 6.5 0 1.2]);
y10 = normpdf(x,5,sqrt(4/(3*10))); %Gaussian Distribution with n = 10;
plot(x,y10,'r');
legend("Gaussian(5,4/30)");

subplot(2,3,5);
h5 = histogram(Z(30,:)./30, 'Normalization', 'pdf', 'Binwidth', 1/31);
h5.Annotation.LegendInformation.IconDisplayStyle = 'off';
title("PDF of Z when n = 30");
xlabel("Zn");
ylabel("PDF");
hold on;
```

```
69 axis([4.3 5.7 0 2.3]);
70 y30 = normpdf(x,5,sqrt(4/(3*30))); %Gaussian Distribution with n = 30;
71 plot(x,y30,'r');
72 legend("Gaussian(5,4/90)");
73
74 subplot(2,3,6);
75 h6 = histogram(Z(100,:)./100, 'Normalization', 'pdf', 'Binwidth', 1/101);
76 h6.Annotation.LegendInformation.IconDisplayStyle = 'off';
77 title("PDF of Z when n = 100");
78 xlabel("Zn");
79 ylabel("PDF");
80 hold on;
81 axis([4.6 5.4 0 3.8]);
82 y100 = normpdf(x,5,sqrt(4/(3*100))); %Gaussian Distribution with n = 100;
83 plot(x,y100,'r');
84 legend("Gaussian(5,4/300)");
```

Question4d

PDF of Z_n:

```
1  %Code Written by Nha Do
2  %This program simulates the pdf of Zn in different number of tossing
3  %Z(n) = (X1 + X2 + ... + Xn)/n
4  %In this case, X(i) representing a toss of a 5-sided dice
5  %Note that this dice is unfair (each side has different probability)
6  clc;
7  clear all;
8  N = [1 2 3 10 30 100]; %Number of tosses
9  t = 10000; % Number of samples
10
11 Z = zeros(length(N),t);
12 for i=1:length(N)
13     for j=1:t
14         for k=1:N(i)
```

```
15              side = tossing_die_1d(); %Tossing dice
16              Z(i,j) = Z(i,j) + side; %Add value to the corresponding position
17          end
18          Z(i,j) = Z(i,j)./N(i); %Calculate the average
19      end
20  end
21
22  subplot(2,3,1);
23  histogram(Z(1,:), 'Normalization', 'pdf', 'Binwidth', 1/2);
24  title("PDF of Z with 1 time of tossing");
25  xlabel("Zn");
26  ylabel("PDF");
27
28  subplot(2,3,2);
29  histogram(Z(2,:), 'Normalization', 'pdf', 'Binwidth', 1/3);
30  title("PDF of Z with 2 times of tossing");
31  xlabel("Zn");
32  ylabel("PDF");
33
34  subplot(2,3,3);
35  histogram(Z(3,:), 'Normalization', 'pdf', 'Binwidth', 1/4);
36  title("PDF of Z with 3 times of tossing");
37  xlabel("Zn");
38  ylabel("PDF");
39
40  subplot(2,3,4);
41  histogram(Z(4,:), 'Normalization', 'pdf', 'Binwidth', 1/11);
42  title("PDF of Z with 10 times of tossing");
43  xlabel("Zn");
44  ylabel("PDF");
45
46  subplot(2,3,5);
47  histogram(Z(5,:), 'Normalization', 'pdf', 'Binwidth', 1/31);
48  title("PDF of Z with 30 times of tossing");
49  xlabel("Zn");
50  ylabel("PDF");
51
```

```
subplot(2,3,6);
histogram(Z(6,:), 'Normalization', 'pdf', 'Binwidth', 1/101);
title("PDF of Z with 100 times of tossing");
xlabel("Zn");
ylabel("PDF");
```

Generates and Superimposes Gaussian Distribution:

```
%Code Written by Nha Do
%Generate Gaussian random variable and apply to previous plots
%Mean = 18/7, Variance = 96/(49n)
clc;
clear all;
N = [1 2 3 10 30 100]; %Number of tosses
t = 10000; % Number of samples

Z = zeros(length(N),t);
for i=1:length(N)
    for j=1:t
        for k=1:N(i)
            side = tossing_die_1d(); %Tossing dice
            Z(i,j) = Z(i,j) + side; %Add value to the corresponding position
        end
        Z(i,j) = Z(i,j)./N(i); %Calculate the average
    end
end

x = 1:0.001:5; %Range of the Gaussian distribution

subplot(2,3,1);
h1 = histogram(Z(1,:), 'Normalization', 'pdf', 'Binwidth', 1/2);
h1.Annotation.LegendInformation.IconDisplayStyle = 'off';
title("PDF of Z with 1 time of tossing");
xlabel("Zn");
ylabel("PDF");
hold on;
```

```
y1 = normpdf(x,18/7,sqrt(96/(49*1))); %Gaussian Distribution with n = 1;
plot(x,y1,'r');
legend("Gaussian(2.57,2.00)");

subplot(2,3,2);
h2 = histogram(Z(2,:), 'Normalization', 'pdf', 'Binwidth', 1/3);
h2.Annotation.LegendInformation.IconDisplayStyle = 'off';
title("PDF of Z with 2 times of tossing");
xlabel("Zn");
ylabel("PDF");
hold on;
y2 = normpdf(x,18/7,sqrt(96/(49*2))); %Gaussian Distribution with n = 2;
plot(x,y2,'r');
legend("Gaussian(2.57,0.979)");

subplot(2,3,3);
h3 = histogram(Z(3,:), 'Normalization', 'pdf', 'Binwidth', 1/4);
h3.Annotation.LegendInformation.IconDisplayStyle = 'off';
title("PDF of Z with 3 times of tossing");
xlabel("Zn");
ylabel("PDF");
hold on;
y3 = normpdf(x,18/7,sqrt(96/(49*3))); %Gaussian Distribution with n = 3;
plot(x,y3,'r');
legend("Gaussian(2.57,0.653)");

subplot(2,3,4);
h4 = histogram(Z(4,:), 'Normalization', 'pdf', 'Binwidth', 1/11);
h4.Annotation.LegendInformation.IconDisplayStyle = 'off';
title("PDF of Z with 10 times of tossing");
xlabel("Zn");
ylabel("PDF");
hold on;
axis([1 4.2 0 1.2]);
y4 = normpdf(x,18/7,sqrt(96/(49*10))); %Gaussian Distribution with n = 10;
plot(x,y4,'r');
legend("Gaussian(2.57,0.200)");
```

```
66
67 subplot(2,3,5);
68 h5 = histogram(Z(5,:), 'Normalization', 'pdf', 'Binwidth', 1/31);
69 h5.Annotation.LegendInformation.IconDisplayStyle = 'off';
70 title("PDF of Z with 30 times of tossing");
71 xlabel("Zn");
72 ylabel("PDF");
73 hold on;
74 axis([1.7 3.5 0 2]);
75 y5 = normpdf(x,18/7,sqrt(96/(49*30))); %Gaussian Distribution with n = 30;
76 plot(x,y5,'r');
77 legend("Gaussian(2.57,0.065)");
78
79 subplot(2,3,6);
80 h5 = histogram(Z(6,:), 'Normalization', 'pdf', 'Binwidth', 1/101);
81 h5.Annotation.LegendInformation.IconDisplayStyle = 'off';
82 title("PDF of Z with 100 times of tossing");
83 xlabel("Zn");
84 ylabel("PDF");
85 hold on;
86 axis([2 3.2 0 3.5]);
87 y5 = normpdf(x,18/7,sqrt(96/(49*100))); %Gaussian Distribution with n = 100;
88 plot(x,y5,'r');
89 legend("Gaussian(2.57,0.02)");
```