EE511 Project 1 Summary

SilongHu   email:silonghu@usc.edu

This summary is divided into 3 parts: Implementation, Conclusion and Source Code

I. Implementation

I.I [Adding Coins ...]

#1 See Bernuolli.m in the source code.

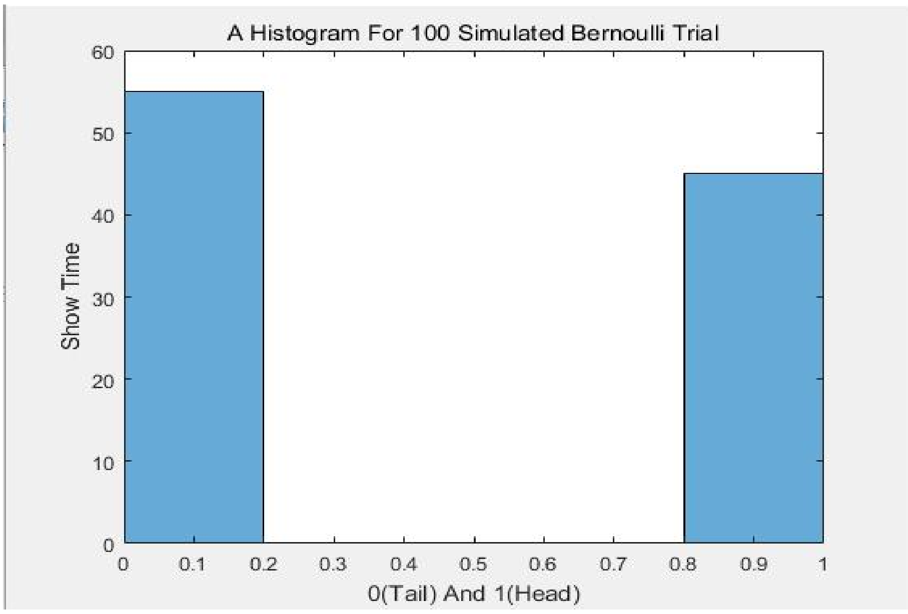


Figure 1. A histogram for 100 simulated Bernoulli trials

Each time executing this program, the frequency of 0 and 1 are different, but nearly 50.

#2 See Bernuolli2.m in the source code.

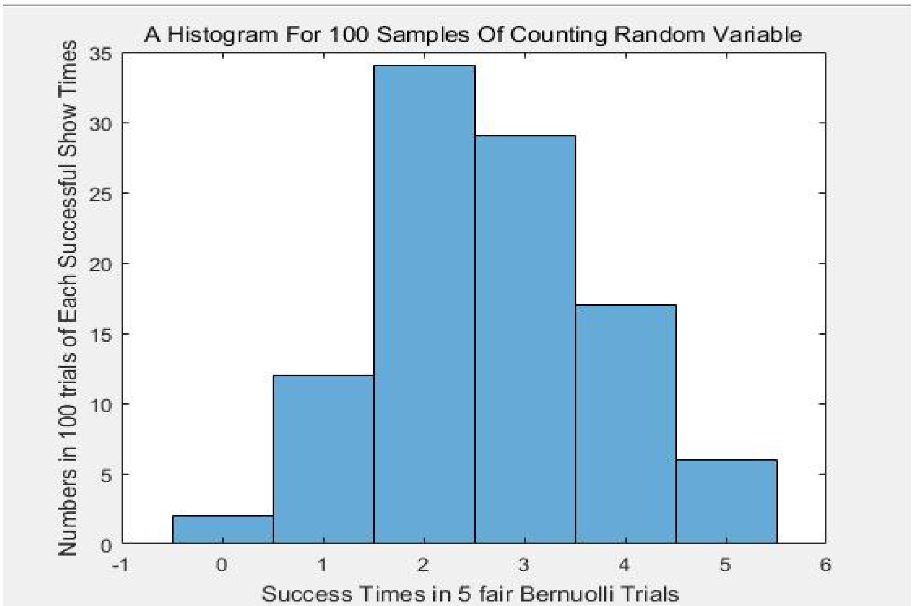


Figure 2. Histogram for 100 samples of Counting random variable

#3 See Bernuolli3.m in the source code

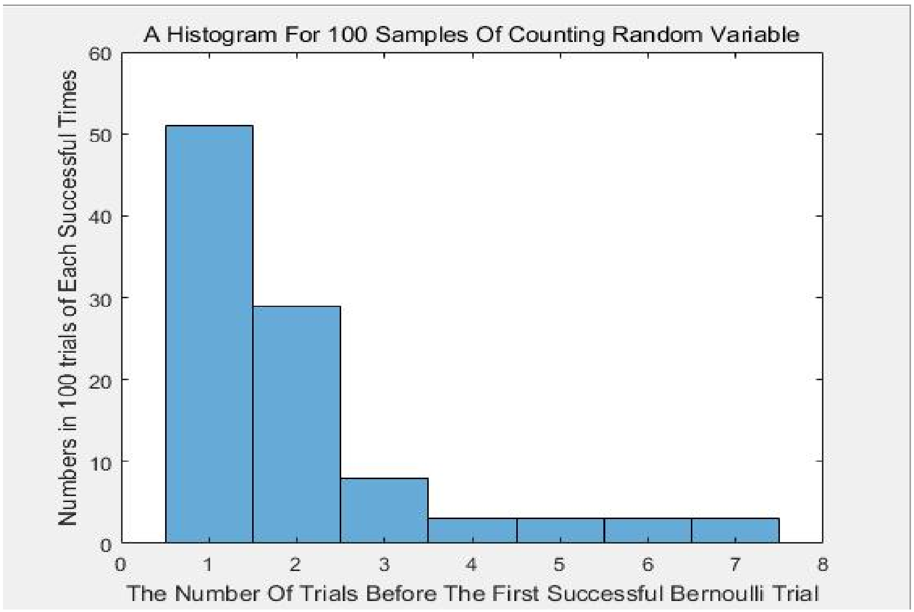
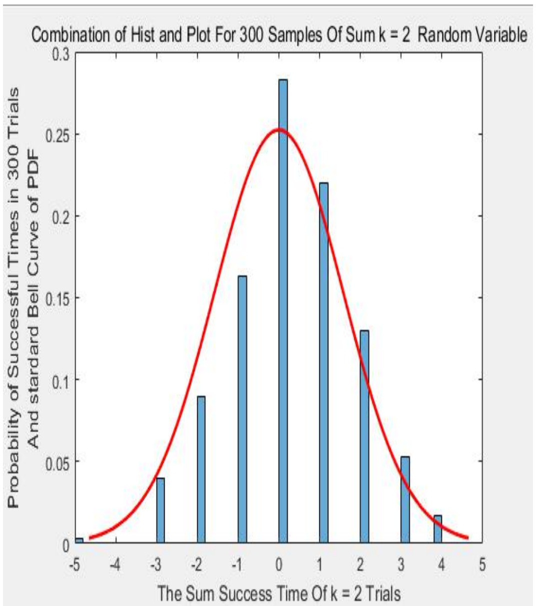


Figure 3. Histogram for 100 samples of Counting random variable until success

I.II. [Coin Limits]

In this section, 2 methods are taken: 1 for standard normal distribution (ND) (Using central limit theorem, see CLT.m in Source code) and the other for original data(see CLT_Original.m in Source code). Using TestSumSum.m to execute. (For the following pictures, each x-axis is different)

#1 Sum = 2



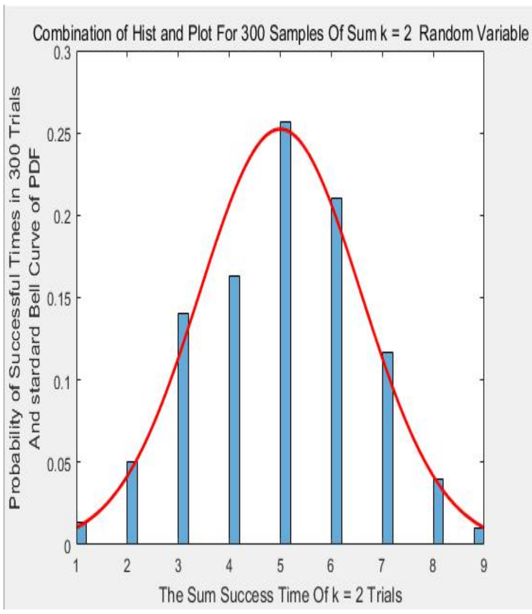
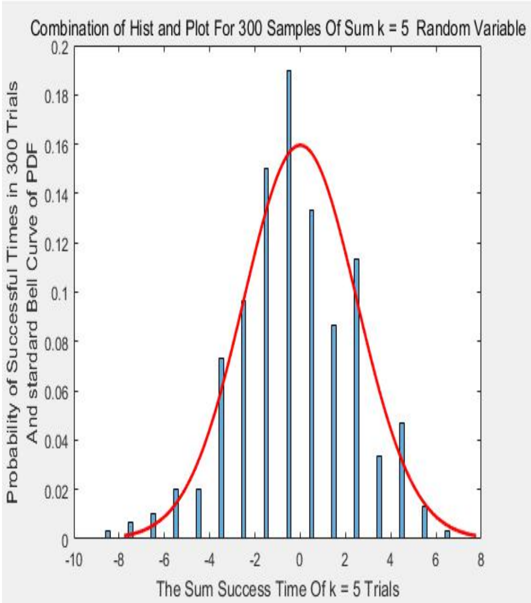


Figure 4.a Standard Normal Distribution Comparison   Figure 4.b Original Data Compare to ND

#2 Sum = 5



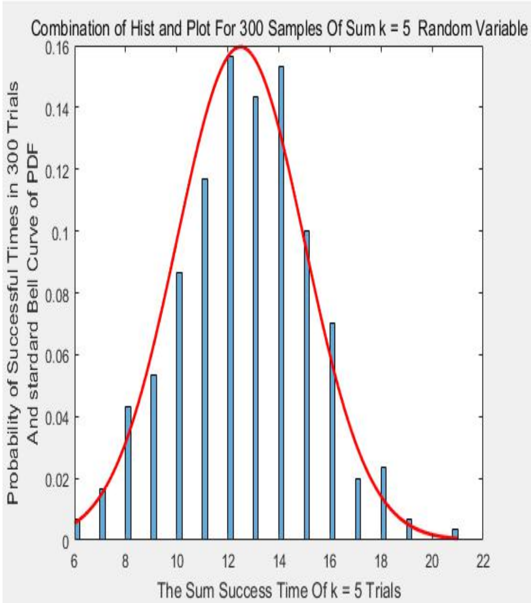
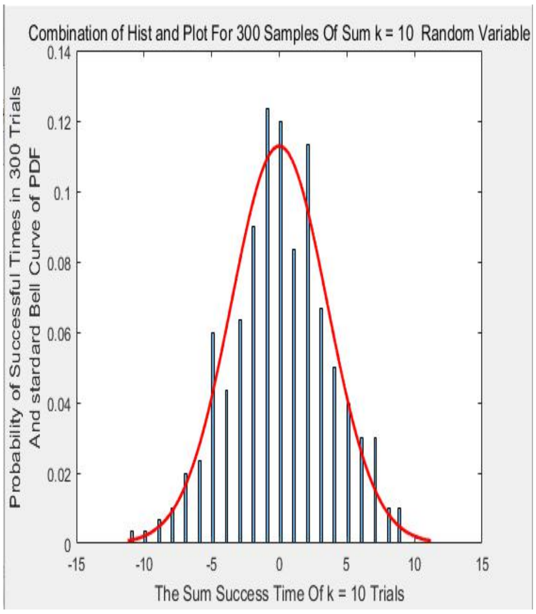


Figure 5.a Standard Normal Distribution Comparison Figure 5.b Original Data Compare to ND

#3 Sum = 10



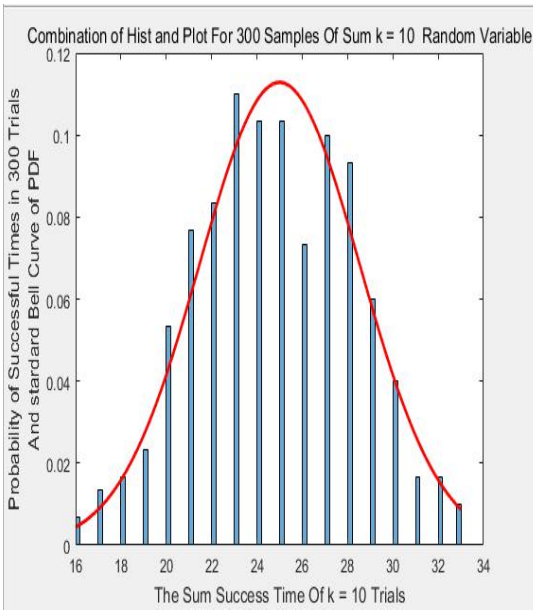
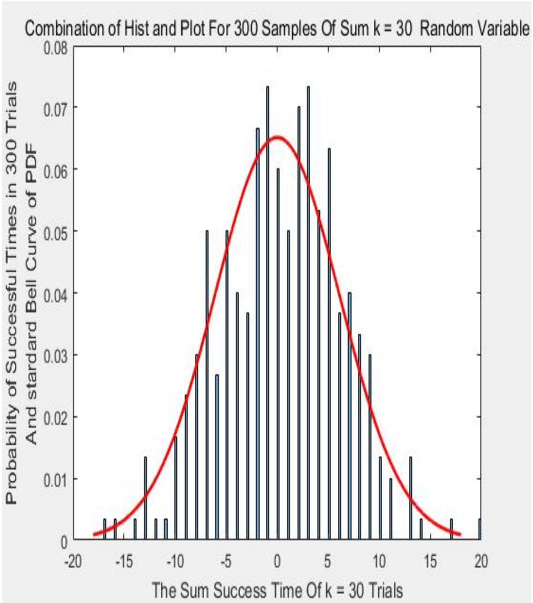


Figure 6.a Standard Normal Distribution Comparison Figure 6.b Original Data Compare to ND

#4 Sum = 30



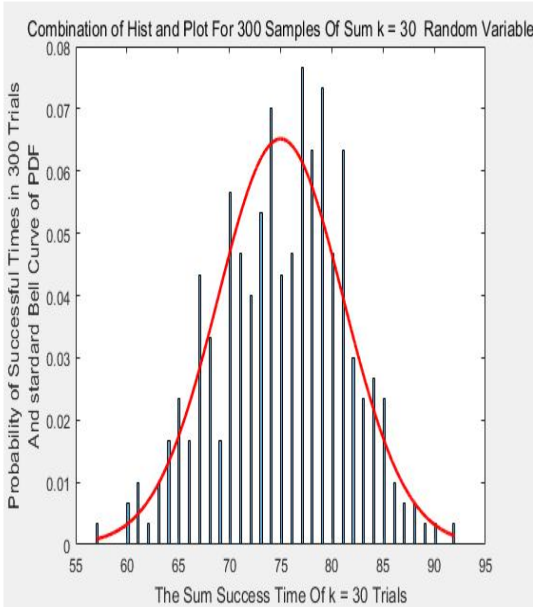
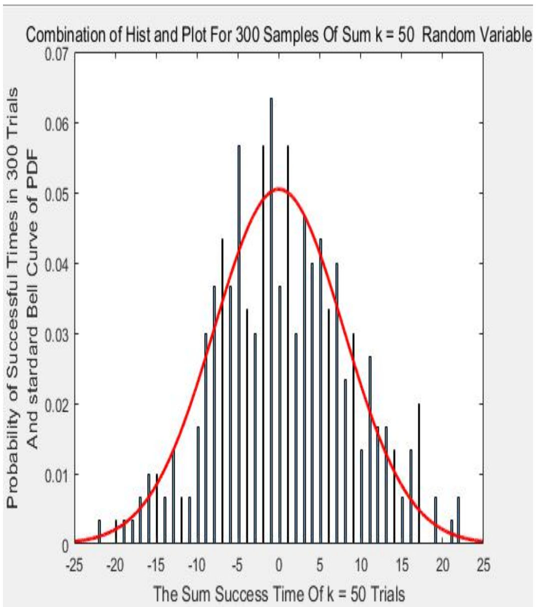


Figure 7.a Standard Normal Distribution Comparison Figure 7.b Original Data Compare to ND

#5 Sum = 50



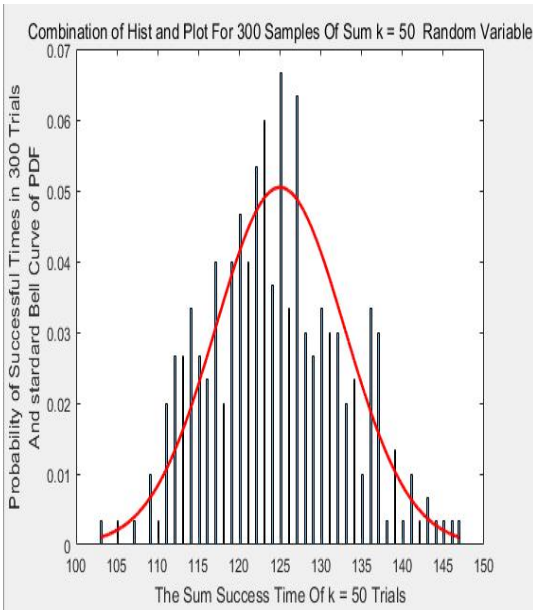


Figure 8.a Standard Normal Distribution Comparison Figure 8.b Original Data Compare to ND

I.III[Bootstrap]

Download the NJGAS data set from Blackboard and find the 95% bootstrap confidence interval for the mean of the data set.

In this part, 2 methods are used:

#1 Randomly choosing from the NJGAS every time with the sampling length of data. And the result turns out (taking 5 trials):

42.2500	165.8333
41.3333	161.5833
42.1667	165.1667
42.8333	165.8333
42.4167	164.4167

#2 Using the BOOTCI function in MATLAB and it turns out (taking 5 trials):

48.3333	179.9747
49.6667	177.8188
49.0000	180.1179
48.9167	179.0000
50.0000	179.4540

II. Conclusion

Identify and compare the distributions in "Adding Coins" section, Due to P = 0.5, in the first trial, 0 and 1 shows around same time which equals to 50. For the second trial, when there are 5 fair Bernoulli trials, the successful time is from 0 to 5, and the peak is around 2 and 3 just like figure 2 shows. As for the last trial, there are nearly 50 times that met the first success. The more numbers of trials before first success, the lower probability that it happens.

As for 'Limit Coin' question, from a bunch of figures varies from 4.a to 8.b, they show that whether using Central Limit Theorem $\sqrt{\mathrm{n}}\left(\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right)-\mu\right)$ to convert original data into N~(0, σ^2) Or just using the original data histogram compare to N~(μ, σ^2)the more samples are taken, the histogram is closer to bell curve, which is generated by *normpdf* using Binomial attributes B(n,p), whose Expectation is np, variance is np(1-p).

Finally, MATLAB provide its own function to evaluate the 95% confidence interval of the statistic. Furthermore, the re-sampling and calculate mean method is also available. Both the same amount of sampling in 2 methods. The error between 2 method is less than 20%.

III.Source Code

#1. Bernuolli.m

```
%Toss a coin 100 times
%1 represents HEAD,0 represents TAIL
a = zeros(1,100); %Store 100 trials' result
prob = 0.5;% the probability to get HEAD
for i=1:100
    a(i) = rand(1) < prob;%randomly generate a number from 0-1;
end
axis([0:1 0 100]);
histogram(a,'BinWidth',0.2,'Normalization','count','DisplayStyle','bar')
title('A Histogram For 100 Simulated Bernoulli Trial');
xlabel('0(Tail) And 1(Head)');
ylabel('Show Time');
```

#2.Bernuolli2.m

```
%Count the number of successes in 5 fair Bernoulli trials
%1 represents Success,0 represents Fail
result = zeros(1,100);
prob = 0.5;
% 100 trials
for i=1:100
    toss = 0;
    for j = 1:5 % Count to 5
        toss = 1 - (rand(1) < prob) + toss; % record the success time
    end
    result(i) = toss;
end

histogram(result,'Normalization','count','DisplayStyle','bar')
title('A Histogram For 100 Samples Of Counting Random Variable')
xlabel('Success Times in 5 fair Bernoulli Trials ')
ylabel('Numbers in 100 trials of Each Successful Show Times')
```

#3. Bernuolli3.m

```
%Count the number of trials before the first successful Bernoulli trial
result = zeros(1,100);%store the counting ramdon variable.
% 100 trials
for i=1:100
    toss = 0;
    count = 0;
    while(toss == 0)
        % if the trial is still fail, go ahead,count the number until success
```

```
        toss = 1 - (rand(1)<0.5);
        count = count + 1;
    end
    result(i) = count;
end

histogram(result,'Normalization','count','DisplayStyle','bar')
title('A Histogram For 100 Samples Of Counting Random Variable')
xlabel('The Number Of Trials Before The First Successful Bernoulli Trial')
ylabel('Numbers in 100 trials of Each Successful Times')
```

#4. CLT.m (function)

```
%Take your Bernoulli success counting routing
%1 represents Success,0 represents Fail
%In this function, the original data has been normalized
function result = CLT(k)
    %result = binornd(k,0.5,1,300);
    Prob = 0.5;
    test = zeros(k,300);
    result = zeros(1,300);
    for n = 1:k
        for i=1:300
            toss = 0;
            for p = 1:5
                toss = 1 - (rand(1)< Prob) + toss ;%if 1 shows less than 50, toss
= 0 (fail)
            end
            test(n,i) = toss;
        end
        test(n,:) =  test(n,:) - 2.5; % minus the mean,B(5,0.5) Ex = 2.5, leave
the std to bell curve
        result(1,:) = result(1,:) + test(n,:);
    end
    [mu,s] = normfit(result);

histogram(result,'BinWidth',0.2,'Normalization','probability','DisplayStyle',
'bar')
    hold on
    x = [-3*s:0.01:3*s];
    norm = normpdf(x,0,sqrt(k*1.25)); % Binomial Distribution, Ex = np, Var =
np(1-p)
    plot(x, norm,'linewidth', 2, 'color', 'r')
    str1 = sprintf('Combination of Hist and Plot For 300 Samples Of Sum k = %d
Random Variable',k);
```

```
    title(str1);
    str2 = sprintf('The Sum Success Time Of k = %d Trials',k);
    xlabel(str2);
    ylabel({'Probability of Successful Times in 300 Trials','And stardard Bell
Curve of PDF'});
```

#5. CLT_Original.m (function)

```
%Take your Bernoulli success counting routing
%1 represents Success,0 represents Fail
%In this function, the original data has been normalized
function result = CLT_original(k)
    %result = binornd(k,0.5,1,300);
    Prob = 0.5;
    test = zeros(k,300);
    result = zeros(1,300);
    for n = 1:k
        for i=1:300
            toss = 0;
            for p = 1:5
                toss = 1 - (rand(1)< Prob) + toss ;%if 1 shows less than 50, toss
= 0 (fail)
            end
            test(n,i) = toss;
        end
        result(1,:) = result(1,:) + test(n,:);
    end

histogram(result,'BinWidth',0.2,'Normalization','probability','DisplayStyle',
'bar')
    hold on
    x = [min(result):0.01:max(result)];
    norm = normpdf(x,2.5*k,sqrt(1.25*k)); % Binomial Distribution, Ex = np, Var
= np(1-p)
    plot(x, norm,'linewidth', 2, 'color', 'r')
    str1 = sprintf('Combination of Hist and Plot For 300 Samples Of Sum k = %d
Random Variable',k);
    title(str1);
    str2 = sprintf('The Sum Success Time Of k = %d Trials',k);
    xlabel(str2);
    ylabel({'Probability of Successful Times in 300 Trials','And stardard Bell
Curve of PDF'});
```

#6. TestSum.m

```
clear all;
clt1 = CLT(50);
%clt2 = CLT_original(50);
```

#7.bootstrap.m

```
A = importdata('NJGAS.dat'); %import the data and store it as a vector
sum = 0;
for i = 1:length(A)
    sum = sum+A(i);
end
avg = sum/length(A);

gather_mean = zeros(1,1000);%store 1000 times sampling mean
for i = 1:1000
    choose = randsample(A,length(A),true);
    %each time pick length(A) samples and count their mean
    sum_sample = 0;
    for j = 1:length(A)
        sum_sample = choose(j) + sum_sample;
    end
    gather_mean(i) = sum_sample/length(A);
end
gather_mean = sort(gather_mean);
CI = gather_mean(25:974);
interval1 = [CI(1),CI(950)];
ci = bootci(1000*length(A),@mean, A);%take the same amount of gether_mean taken
interval2 = ci;
disp(interval1)
disp(interval2)
```