Project #1 - Coin flip

Li Yicheng[*]

USCID:7827077047

email: l.y.c.liyicheng@gmail.com

USC Viterbi of Engineering

I. Simulation of the Bernoulli Outcomes —- Q1

1.Requirement: tossing a fair coin 50 times. Count the number of heads. Record the longest run of heads. Generate a histogram for the Bernoulli outcomes.

2.Tool: Matlab (Version: R2014b Platform: Ubuntu 16.04 LTS)

3.Experiment: For further experiment convenience, I generated a function for the experiment trial. In the function, a coin (can be either fair or biased) is tossed several times. The amount of trial is determined by the input parameter eval_budget and the coin property(fair or not) is determined by the parameter p_thre. The output is the head number and the longest run of the head. This function also plot the histogram showing the number of heads and tails plus a figure showing the relative frequency througout the experiment.

4.core code:

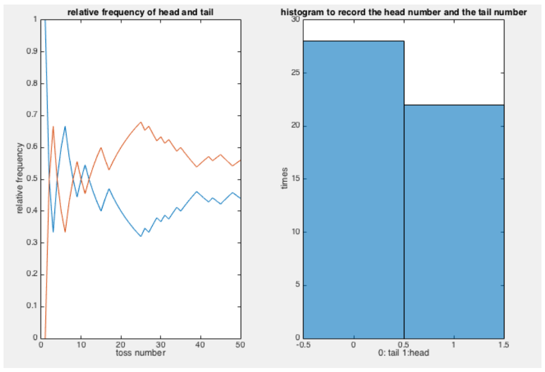
```
function [head_number,longest_run] = tossing_coin(eval_budget,p_thre)
% usage: function [head_number,longest_run] = tossing_coin(eval_budget,p_thre)
%tossing eval_budget times
%Author : Li Yicheng

%define variables
result = zeros(eval_budget,1);
length = 0;
longest_run = 0;
head_number = 0;
tail_number = 0;
head_rela_fre = zeros(1,eval_budget);
tail_rela_fre = zeros(1,eval_budget);
for i = 1 : eval_budget
    coin = rand(1) < p_thre;
    if coin > 0 % a head
        head_number = head_number + 1;
        length = length + 1; %add length
        result(i) = 1;
       %the state now is a head
    else % a tail
        tail_number= tail_number + 1;
        result(i) = 0;
        if length > longest_run
            longest_run = length; %refresh the new longest run
        end
        length = 0; %reset the length
```

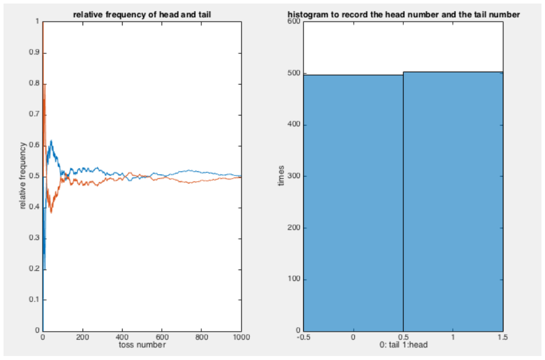
[*]github link: https://github.com/IAMLYCHEE/EE511-PROJECT1

```
        %the state now is a tail
    end
        head_rela_fre(i) = head_number / i;
        tail_rela_fre(i) = tail_number / i;
end
```

5.result: Executing query: tossing_coin(50,0.5)



(a) *toss 50 times*



(b) *toss 1000 times*

Figure 1: *left: change of relative frequency , right: histogram of the Bernoulli Outcome*

from the figure1(a), the experiment ends with 28 heads and 22 tails and let's toss the coin 1000 times by calling tossing_coin(1000,0.5), and we got the figure1(b). And from the result we can say with a far more larger amount of trials given, the relative frequency of heads and tail both have a convergency to 0.5. In this experiment, the longest run of heads is 6.

II. Number of Heads Experiment —- Q1.a

1.Requirement: Repeat the first experiment 20,100,200, and 1000 times. Count the number of heads in 50 flips. Generate a histogram showing the phenomenon.

2.Tool: Matlab (Version: R2014b Platform: Ubuntu 16.04 LTS)

3.Experiment: Since this time we only care about the result of the amount of heads, we only need the sum of the result vector (0 is tail and 1 is head, therefore the sum of the vector is the number of the heads in 50 flips). Then we generate a histogram showing what is the most frequent number of head we would get.

4.core code:

```
function repeat_trial(trial_time)
%usage: repeat_trial(trail_time)
%Li Yicheng
%repeat the 50 tossing coin trial several times
for i = 1 : trial_time % repeat trial_time
    result  = rand(1,50); %generate the result
    exp_trial(i) = sum(result);%count the number of heads and store the amount
```

```
end
histogram(exp_trial,50,'BinLimits',[0,50]) %plot the result of the experiment
title('exp_trial times')
xlabel('number of heads')
ylabel('times')
```

5.result:

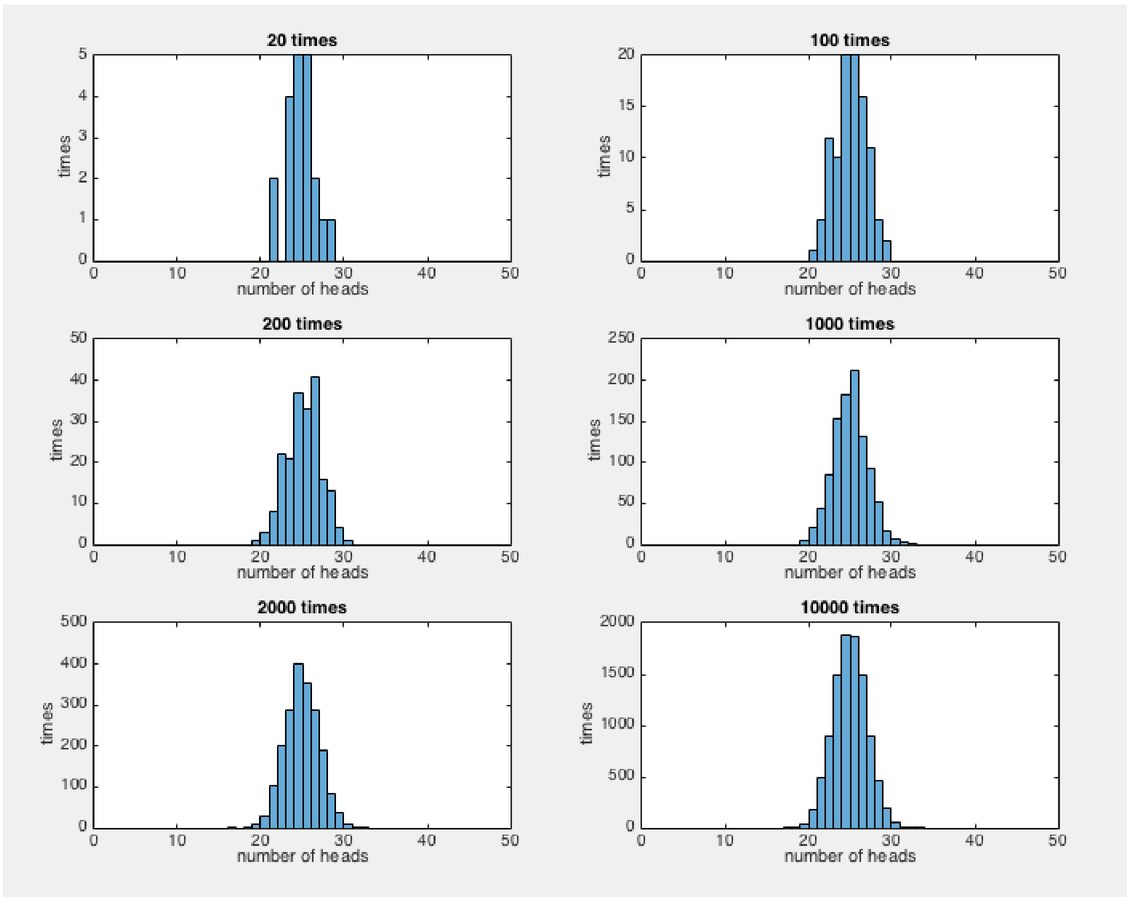


Figure 2: *number of heads histogram*

Observation: we can see that we got four 'mountains' in figure2. when the trial time increase to 1000, the peak of the mountain is somewhere near 25 and the number of heads has the highest probability to vary between 23 to 27. This is quite easy to get for a fair coin, because there is no biase on head over tail. When we assigned a large number 10000 to this experiment we saw a fairly symmetric mountain. Both side decline in a regular way. Mathematically, this situation can be computed. The probability to get 25 heads is $\binom{50}{25}(0.5)^{25} \cdot 0.5^{25}$, the 26 heads probability is $\binom{50}{26}(0.5)^{26} \cdot (0.5)^{24}$ and $P(25heads)/P(26heads) = 26/25$ generally $\binom{n}{k}/\binom{n}{k+1} = (k+1)/(n-k)$ so it will decrease faster and faster.

III. Biased Coin Experiment —- Q2

1.Requirement: Simulate tossing a biased coin 200 times where P(HEAD) = 0.8. Count the number of heads. Record the longest run of heads. Generate a histogram for the outcomes.

2.Tool: Matlab (Version: R2014b Platform: Ubuntu 16.04 LTS)

3.Experiment: Using the tossing_coin function again in Q1, query tossing_coin(200,0.8). Two figures are derived one is the tendency of the relative frequency and the other is the histogram showing the number of heads and tails, the same as the first experiment.

4.result:

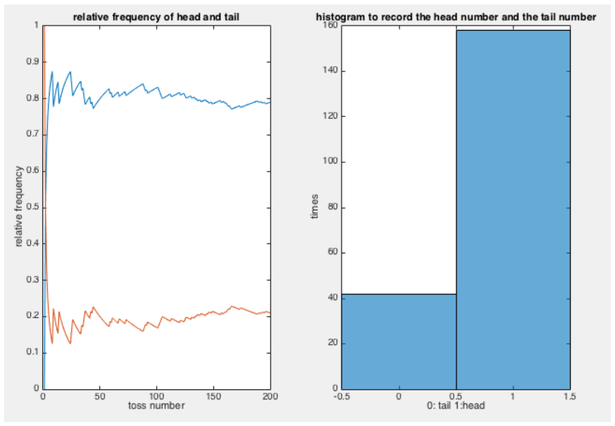


Figure 3: *number of heads histogram*

Figure3 shows the result of tossing a biased coin, the relative frequency of heads has a convergency to 0.8, the number of heads is nearly 160 which is the 80% of 200.

IV. Head Run Length Experiment —- Q3

1.Requirement: Simulating tossing a fair coin 100 times. Generate a histogram showing the heads run length.

2.Tool: Matlab (Version: R2014b Platform: Ubuntu 16.04 LTS)

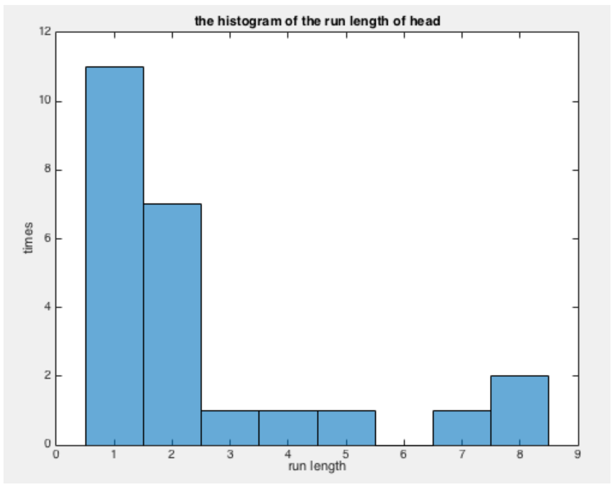
3.Experiment: Construct a function called head_run with the input parameter tossing times named eval_budget. To do this experiment by calling head_run(100).

4.Core Code:

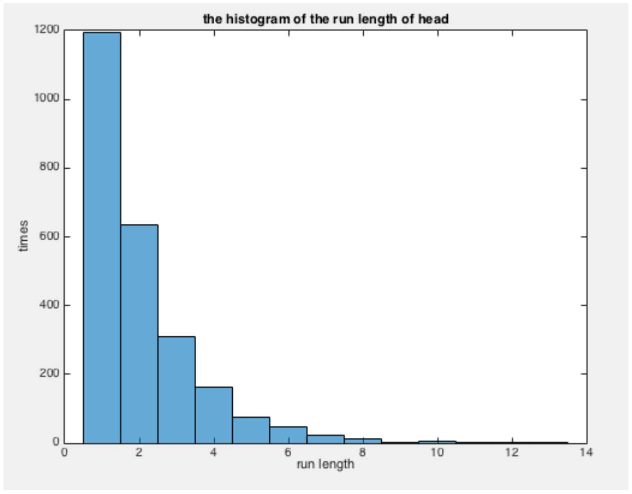
```
function head_run(eval_budget)
% usage: function head_run(eval_budget)
% Li Yicheng
% record the heads run
head_number = 0;
length = 0; %to store the number of heads run length
j = 1; %store the index of the record, every time a continous series of heads end, j plus
    one to its self for another record.
```

```
for i = 1 : eval_budget %toss eval_budget times
    coin = rand(1) > 0.5; % a fair coin
    if coin > 0 %a head
        head_number = head_number + 1;
        length = length + 1; %add length
    else % a tail
        if length > 0
            record(j) = length;  %record the head run
            j = j + 1;
        end
        length = 0; %reset the length
    end
end
histogram(record')
title('histogram for the outcome')
title('the histogram of the run length of head')
xlabel('run length')
ylabel('times')
```

5.result:



(a) *tossing 100 times*



(b) *tossing 10000 times*

Figure 4: *number of heads histogram*

From the figure4(a) we can see that the run length has the largest posiibility to be one or two and the longest run length is 8. Let's make an assumption that this is also true when we tossing a coin 10000 times. let's excuting a query of head_run(10000). Not surprisingly, we have the longest run length of only 13 and we have the most frequent length of 1, 2 and 3. Therefore, the conclusion is that it is hard for the head condition can not maintain for a long time. Moreover, if we take a look at the figure 5, we saw a 'regular' downstair. If we use the knowledge of probability to describe what we saw in figure 4(b), every toss is a i.i.d.(independent and identically distributed) experiment. So we get the run length of 1 with the probabilty 0.5, and 2 with 0.25, 3 with $1/2 \cdot 1/2 \cdot 1/2 = 1/8$. The figure 4(b) seems to arrive at the similar result.

V. Reach Specified amount of Heads —- Q4

1.Requirement: Simulate tossing a fair coin and count the number of tosses until reaching a user-specified possitive number of heads.

2.Tool: Matlab (Version: R2014b Platform: Ubuntu 16.04 LTS)

3.Experiment: Just use the while loop in the code design and the loop break until the requirement is met. Use a table to record the data. Query the function toss_until(p_thre)

4.Core Code:

```
function num_tosses = toss_until(p_thre)
%usage: function num_tosses = toss_until(p_thre)
%Author: Li Yicheng
num_wanted = input('the number of heads you want:');
head_number = 0;
num_tosses = 0;
while (head_number < num_wanted)
    num_tosses = num_tosses + 1;
    if rand(1) > p_thre %a head occurs
        head_number = head_number + 1;
    end
end
```

5.result:

heads query	trial amount
20	45
50	110
100	189
200	407
500	1024
1000	1984

The results show that we need nearly twice the number of the expected amount of heads trials. And for a fair coin, this result is rational because the probabilty to get a head is 50%.