Acknowledgements: I worked with Sejal Dua on problems 2-4 and Matt Manfre on scripting problem 5.

Problem 1

MATLAB code below.

```
%Sook-Hee Evans
%EE24 Compute Set Part 1

function [ss,probs] = computeHist(ds, varargin)
% function varargout = plotData(varargin
% computes probabilities of a set and plots a histogram
% Inputs:
%   variable: can be one or two, can pass in just a data set ds or
%   a data set ds and a sample space ss.
% Output:
%   ordered set ss and probabilities of each

rng('shuffle')

%% if the data set and sample space are passed in, produce the sample space
%% as output
if nargin == 2
    ss = varargin;
%% if only the data set is passed in, produce the ordered unique set
elseif nargin == 1
    ss = unique(ds);
    fprintf("Number of unique values in dataset: %i\n", length(ss));
end

h = histogram(ds, length(ss), 'EdgeColor', [.5 .6 1], 'FaceAlpha', 0.6,...
'FaceColor',[1 .509 .67]);
title('Occurences vs. Length of Tweets','FontSize',20);
ylabel('Number of occurences');
xlabel('Length of tweet in sample space');

%This variable is used to track the number of tweets with 100 or more
%characters for problem 4.
moreThan100 = 0;

for a = 1:length(ss)
    if h.Values(a) == 0
        continue;
    else
        if a >= 100
            moreThan100 = moreThan100 + 1;
        end
        probs(ss(a)) = h.Values(a) / numel(ds);
```

```
43          fprintf("Probability of tweet with length %i: %.4f \n", ss(a),probs(ss
                (a)));
44      end
45  end
46
47  fprintf("The number of tweets with more than 100 chars is: %i\n", moreThan100)
        ;
48  fprintf("The sample mean of the tweet lengths is: %1.4f\n", mean(ss));
49
50  return
```

Problem 2

By running my computeHist function with N = 500 random integers and using fprintf to print the mean of the set at the end of the function, I found that the average, as expected, was about 70.5 characters. I ran five trials, and the mean lengths for each of those trials were 70.17, 71.03, 71.18, 70.71, and 70.35. Because the lengths of the tweets are integers distributed randomly from 1-140, it makes sense that the average number of characters would hover around 70.5 for an experiment based on 500 tweets.

```
Probability of tweet with length 119: 0.0080
Probability of tweet with length 120: 0.0120
Probability of tweet with length 121: 0.0020
Probability of tweet with length 122: 0.0040
Probability of tweet with length 123: 0.0040
Probability of tweet with length 126: 0.0040
Probability of tweet with length 127: 0.0080
Probability of tweet with length 128: 0.0060
Probability of tweet with length 129: 0.0060
Probability of tweet with length 130: 0.0080
Probability of tweet with length 131: 0.0160
Probability of tweet with length 132: 0.0060
Probability of tweet with length 133: 0.0020
Probability of tweet with length 134: 0.0080
Probability of tweet with length 135: 0.0120
Probability of tweet with length 136: 0.0080
Probability of tweet with length 137: 0.0100
Probability of tweet with length 138: 0.0120
Probability of tweet with length 139: 0.0080
Probability of tweet with length 140: 0.0060
The number of tweets with more than 100 chars is: 37
The sample mean of the tweet lengths is: 70.4234
```

Figure 1: Partial output for 500 tweets.

Problem 3

Compared to a histogram with 100 tweets, the histogram with 500 tweets is much more even. This makes sense, since as the number of elements of the data set go up, the sample variance

goes down. Because a set with less than at least 140 random integers will always have at least one bin with zero occurrences, small sets have gaps in the histogram that a larger set will have much less of or even possibly lack completely.

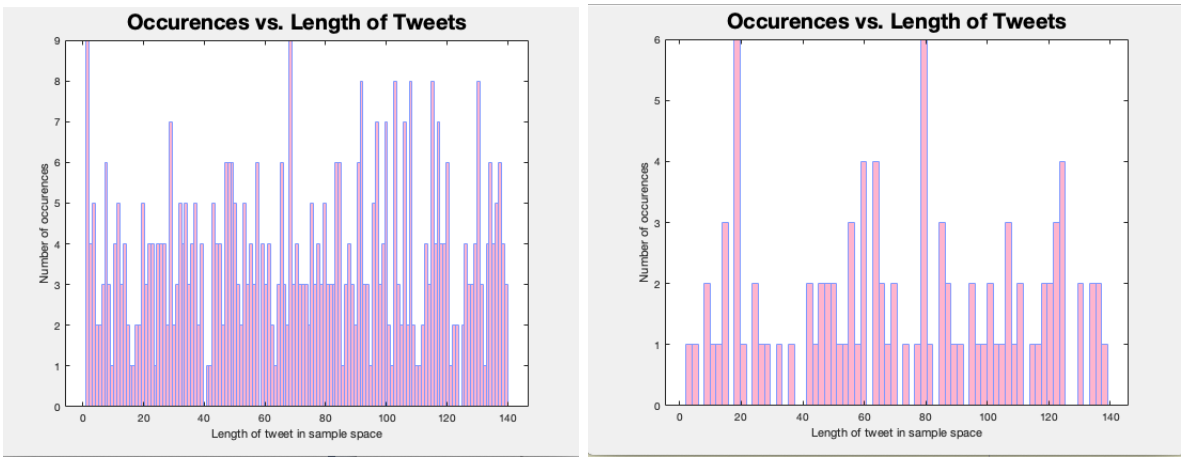


Figure 2 and 3. On the left, a histogram of 500 tweets and on the right, a histogram of 100 tweets. As number of tweets increases, the histogram will become more and more even.

Problem 4

In theory, about 41 in 140 tweets should be greater than or equal to 100 characters, since 100 is included‘. As the number of tweets increases, the number of tweets greater than or equal to 100 characters will become closer and closer to that fraction as the sample variance decreases. As seen above, I used a counter to count the total number of tweets greater than or equal to 100 characters and printed that to the terminal each run. The values for 5 trials are below:

$$\frac{39}{140}$$
$$\frac{38}{140}$$
$$\frac{37}{140}$$
$$\frac{40}{140}$$
$$\frac{43}{140}$$

These five values average to about 0.281, and 41 in 140 is about 0.293. These values are expected.

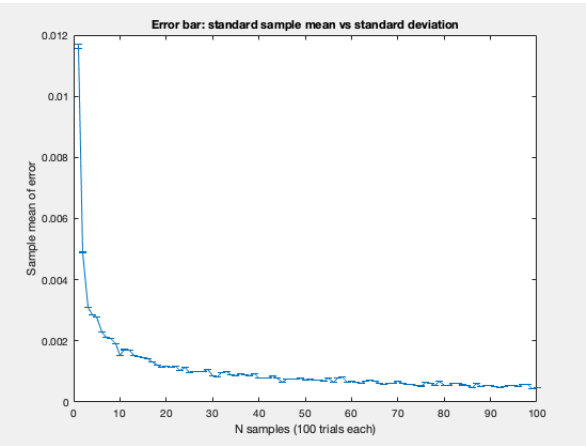
Problem 5

MATLAB function and script below.

```
%Sook–Hee Evans
%EE24 Compute Set Part 1

function count = moreThan100(ds)
    num = 0;
    totalLen = length(ds);
    for i = 1:totalLen
        %% Increments counter to count number of tweets above 100 chars
        if ds(i) >= 100
            num = num + 1;
        end
    end
    %% Return fraction of tweets longer or equal to 100 chars
    count = num / totalLen;

return
```

```
%Sook–Hee Evans
%EE24 Compute Set Part 1

theoreticallyTrue = 41 / 140;
count = 1;

% Monte Carlo Simulation
for i = 1000:5000:500000
    %Run 100 times
    for numTrial = 1:100
        data = randi([1 140],1,i);
        trialError = moreThan100(data);
        diff = trialError - theoreticallyTrue;
        error(numTrial) = abs(diff);
    end
    [sampMean(count), sampVar(count)] = computeData(error);

    fprintf("Average error in N = %i is %1.8f and var is %1.8f\n", i, sampMean
        (count), sampVar(count));
    count = count + 1;


end

%Errorbar plot of sample mean and standard dev
figure
errorbar(sampMean, sampVar);
ylabel('Sample mean of error');
xlabel('N samples (100 trials each)');
```

```
29 title('Error bar: standard sample mean vs standard deviation')
30
31 %Log–Log figure of sample mean of error v. N
32 figure
33 loglog(1000:5000:500000, sampMean);
34 ylabel('Sample mean of error');
35 xlabel('N samples (100 trials each)');
36 title('Log–log scale of sample mean of error versus N number samples')
37 x = log(1000:5000:500000);
38 y = log(sampMean);
39 coefficients = polyfit(x, y, 1);
40 % Now get the slope, which is the first coefficient in the array:
41 slope = coefficients(1);
42 fprintf("Slope is: %1.4f\n", slope);
```



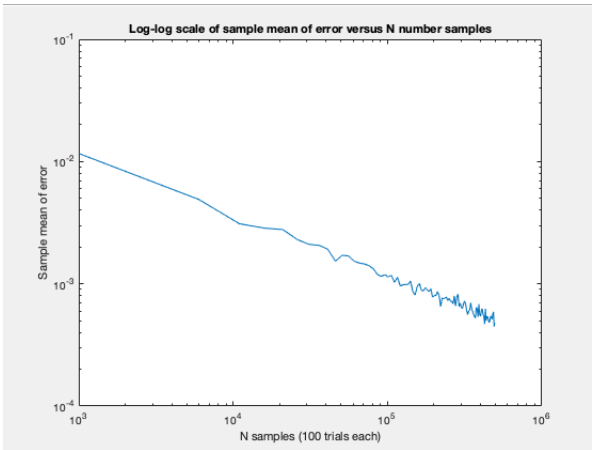


Figure 4 and 5. On the left, the error bar plot with the sample standard mean. On the right, the log-log scale plot of the sample mean of the error versus N. The slope of the log-log plot was about -0.507.

```
Average error in N = 441000 is 0.00058522 and var is 0.00000023
Average error in N = 446000 is 0.00059415 and var is 0.00000016
Average error in N = 451000 is 0.00053710 and var is 0.00000015
Average error in N = 456000 is 0.00055397 and var is 0.00000020
Average error in N = 461000 is 0.00050949 and var is 0.00000013
Average error in N = 466000 is 0.00050070 and var is 0.00000015
Average error in N = 471000 is 0.00047423 and var is 0.00000013
Average error in N = 476000 is 0.00053412 and var is 0.00000016
Average error in N = 481000 is 0.00049714 and var is 0.00000013
Average error in N = 486000 is 0.00047939 and var is 0.00000015
Average error in N = 491000 is 0.00051503 and var is 0.00000015
Average error in N = 496000 is 0.00046633 and var is 0.00000013
Slope is: -0.5071
```

Output of histScript showing the slope of the log-log plot. Sample var showed a large decrease from the first N samples to the last shown above.

To test what factor increase of the size of the data set would cut the average error in half, I solved the line equation given by the slope of the log plot, which came out to be log(y) = -0.5(log(x)) - 1.04. To half log(y), the right side of the equation should be doubled to be log(y) = -(log(x)) - 2.08. By using a system of equations, 4 is found because it halves the error. This was tested in histScript and verified that the average error was halved by quadrupling the size of the test data..