Table of Contents

```
%ENG006 A07
%Chengeng Xiao
%SID: 913186040
%These programs solves 4 problems in HW3
```

Problem 1 Rolling of a six-sided dice:

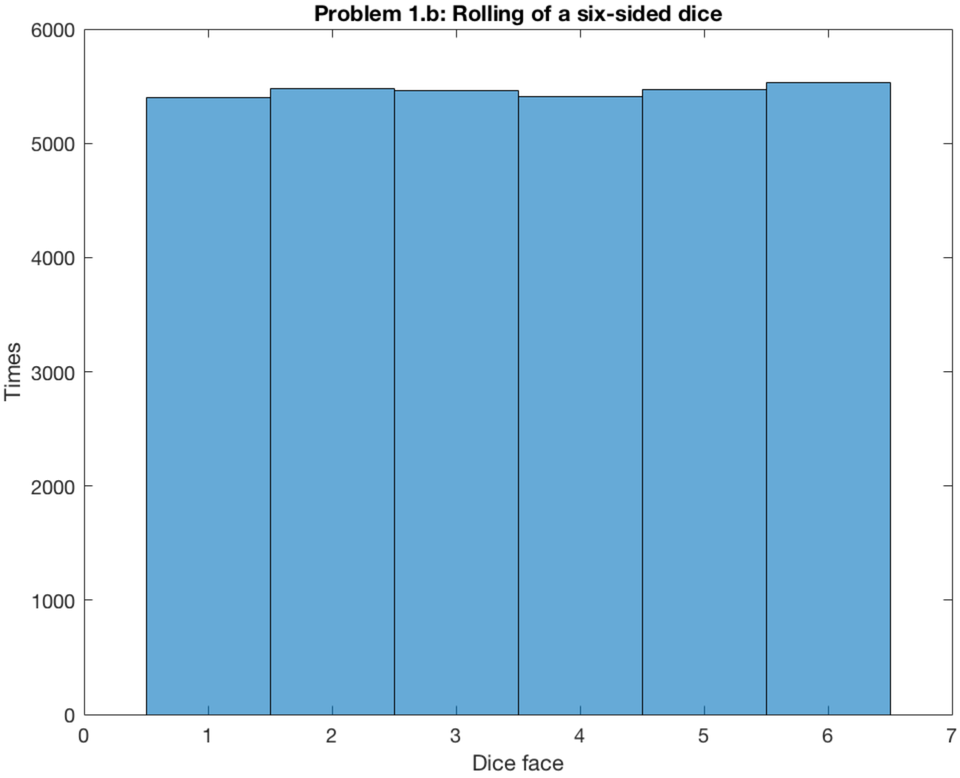
```
%a.
roll = randi(6, 1, 32767); % roll dice
s1 = sum(roll == 1); % side 1s
s2 = sum(roll == 2); % side 2s
s3 = sum(roll == 3); % side 3s
s4 = sum(roll == 4); % side 4s
s5 = sum(roll == 5); % side 5s
s6 = sum(roll == 6); % side 6s
fprintf('Problem 1.a):\n1s showed %d times.\n2s showed %d times.\n3s
 showed %d times.\n4s showed %d times.\n5s showed %d times.\n6s showed
 %d times.\n',s1,s2,s3,s4,s5,s6); % print with format

%b.
figure(1); % Set new figure
histogram(roll); % Draw histogram
title('Problem 1.b: Rolling of a six-sided dice'); % Set title
xlabel('Dice face'); % Set label x
ylabel('Times'); % Set label y

%c.
fprintf('Problem 1.c):\nI can learn from the histogram that each face
 has a similar probability to show.\nThe the probability to get the
 number of 6 is %f.\n',s6 / 32767);
```

```
Problem 1.a):
1s showed 5402 times.
2s showed 5480 times.
3s showed 5464 times.
4s showed 5413 times.
5s showed 5474 times.
6s showed 5534 times.
Problem 1.c):
I can learn from the histogram that each face has a similar
 probability to show.
```

```
The the probability to get the number of 6 is 0.168889.
```



Problem 2 Differentiation and plot:

```
%a.
x = [0:0.001:2]; % Create a vector x starts at 0 and stop at 2 with a
 step of 0.001.

%b.
y2b = 2 .* sin(x) .* cos(x); % Plot the vector y where y =
 2sin(x)cos(x).
figure(2); % Set new figure
plot(x, y2b); % Draw plot
title('Problem 2.b: plot of y = 2sin(x)cos(x)'); % Set title
xlabel('x'); % Set label x
ylabel('y'); % Set label y
legend('y = 2sin(x)cos(x)'); % Set legend

%c.
z = diff(y2b); % Differentiate the vector y numerically using the
 Matlab function diff and assign the output to z.

%d.
zVector = z / 0.001; % zVector is (z/step), the vector created by
 dividing z by the step.
zVector(end + 1) = 0;
```

```
y2d = 2 .* cos(2 .* x); % y = 2cos(2x)
figure(3); % Set new figure
plot(x(1:end-1), zVector(1:end-1),'r', x(1:end-1),
 y2d(1:end-1), 'b'); % Draw plot get rid of the last one because after
 diff z vector is one shorten
title('Problem 3.d: plot of vector z and y = 2cos(2x)'); % Set title
xlabel('x'); % Set label x
ylabel('y'); % Set label y
legend('z in vector', 'y = 2cos(2x)'); % Set legend

%e.
fprintf('Problem 2.e:\nFrom the figure in d, I learned that the
 differentiation y = 2sin(x)cos(x) of is y = 2cos(2x) because their
 figure is the same.\n'); % Print with format
```

```
Subscript indices must either be real positive integers or logicals.

Error in HW3 (line 39)
z = diff(y2b); % Differentiate the vector y numerically using the
 Matlab function diff and assign the output to z.
```

Problem 3 Integration:

```
%a.
a = 3.01 * (10 ^ 9); % Set a
b = 2.74 * (10 ^ 9); % Set b
k = sqrt((a ^ 2) - (b ^ 2)) / a; % Set k
dtheta1 = pi / 2 / 100; % Set d? with 100 discrete point
dtheta2 = pi / 2 / 1000;% Set d? with 1000 discrete point
theta1 = 0 : dtheta1 : pi / 2; % Set ? with 100 discrete point
theta2 = 0 : dtheta2 : pi / 2; % Set ? with 1000 discrete point
P1 = 4 * a * sum(sqrt(1 - ((k ^ 2) * (sin(theta1) .^ 2)))) *
 dtheta1; % Calculate P1
P2 = 4 * a * sum(sqrt(1 - ((k ^ 2) * (sin(theta2) .^ 2)))) *
 dtheta2; % Calculate P2
fprintf('Problem 3.a:\nThe distance Uranus travels in one orbit
 calculated with 100 discrete point is %f, and calculated in 1000
 discrete point is %f.\n', P1, P2); % Print with format

%b.
fun = @(theta) sqrt(1 - ((k ^ 2) * (sin(theta).^2))); % Set fun for
 integral
PIntegral = 4 * a * integral(fun, 0, (pi/2)); % Calculate integral
error1 = abs(P1 - PIntegral) / P1 * 100; % Error of P1 in percentage
error2 = abs(P2 - PIntegral) / P2 * 100; % Error of P2 in percentage
fprintf('Problem 3.b:\nThe distance Uranus travels in one orbit
 calculated with integreal is %f, the percentage error calculated with
 100 discrete point is %f%%, and the percentage error calculated with
 1000 discrete point is %f%%.\n',PIntegral, error1, error2); % Print
 with format

%c.
time = 84.3 * 365 * 24 * 60 * 60; % Convert time in to second
```

```
speed1 = P1 / time; % Calculate with P1
speed2 = P2 / time; % Calculate with P2
fprintf('Problem 3.c:\nThe speed calculated with 100 discrete point is
 %fkm/s, and the speed calculated with 1000 discrete point is %fkm/s.
\n', speed1, speed2); % Print with format
```

Problem 4 Interpolations

```
%a.
v = [5, 10, 15, 25, 35, 40, 50, 65, 80]; % Set list v
Fe = [47, 72, 82, 80, 71, 66, 55, 42, 31]; % Set list Fe
p = polyfit(v, Fe, 3); % Set polynomial curving fit curve p
x4a = 1 : 1 : 85; % Set x
y4a = polyval(p, x4a); % Set y
figure(4); % Set new figure
plot(v, Fe, 'o', x4a, y4a); % Draw plot
title('Problem 4.a: Curve-fit the data with a third-order
 polynomial.'); % Set title
xlabel('v'); % Set label x
ylabel('Fe'); % Set label y

%b.
Fe60 = polyval(p, 60); % Get polynomial to estimate the fuel
 efficiency at 60mi/hr
p2 = polyfit(Fe, v, 3); % Set polynomial curving fit curve p2 Fe
 against v
v50 = polyval(p2, 50); % Get velocities were fuel efficiency, FE ?
 50mpg
fprintf('Problme 4.b:\nThe fuel efficiency at 60mi/hr is %f.
 Also,velocities were fuel efficiency, FE ? 50mpg is %f.\n',Fe60,
 v50); % Print with format

%c.
s = spline(v, Fe, x); % Set cubic spline data interpolation curve s
i = interp1(v, Fe, x); % Set 1-D data interpolation curve i
figure(5); % Set new figure
plot(v, Fe, 'o', x, s, 'b', x, i, 'r'); % Draw plot
title('Problem 4.c: Fit the data with linear and spline
 interpolations.'); % Set title
xlabel('v'); % Set label x
ylabel('Fe'); % Set label y

%d.
diff = abs(Fe60 - i) / Fe60 * 100; % Calculate difference in
 percentage
fprintf('Problem 4.d:\nThe relative difference or error between the
 fuel efficiency prediction at 60mi/hr in part (b) with respect to the
 estimates in part (c) is %f%%.\n', diff); % Print with format
```

Published with MATLAB® R2016a