Table of Contents

```
%Roshan Jaiswal-Ferri
%Section - 03
%Aero 300 Lab 1 - Review of the MATLAB Environment: 4/5/24
```

```
close all;       %Clears all
clear all;       %Clears Workspace
clc;             %Clears Command Window
```

PART 1: Plotting

```
o = -2*pi; %Setting bounds to variables
p = 2*pi;


theta = linspace(o,p,130); %Creating vector with proper bounds and stepping

y = pi*sin(theta/2); %Given Function

g = theta/2; %Other given function

figure; %Creating a figure with overlayed functions
plot(theta, y)
hold on
plot(theta, g)
grid on;
title('Graph of πSin(θ/2) & θ/2')

figure; %Creating a figure where both functions recieve their own subplot
subplot(1,2,1)
plot(theta, y)
hold on
title('Graph of πSin(θ/2)')
grid on;
subplot(1,2,2)
plot(theta, g)
grid on;
title('Graph of θ/2')
```

PART 2: Creating Vectors

```
y=1;
x = linspace(0,10,100000); %Creating a vector from 1 to 10 with 100000 steps
```

```
r1 = zeros(1,100000); %Creating an r1 zero vector
r2 = zeros(1,100000); %Creating an r2 zero vector

tic %Start stopwatch
for i = 1:100000 %looping 100000 times and substituting each value in from
their corresponding vector positions
    u = x(1,i);
    r1(1,i) = 3*(u^3)-(u^2)-1;
end

g = toc; %End timer
%disp(['R1: ', num2str(r1(1,1:9))]); %idk if we were supposed to print the
100000 long vectors but here they are
disp(['Time to Calculate R1 using for loop: ', num2str(g)])

tic %Start new timer
r2 = 3*x.^3-x.^2-1; %the period tells matlab that it is working with a matrix
v = toc; %End timer
%disp(['R2: ', num2str(r2)]);
disp(['Time to Calculate R2 not using for loop: ', num2str(v)])

%There are fewer lines of code for matlab to process when you are not using
%the for loop, which means the process is better optimized for matlab and
%will run faster than a for loop which has more lines of code to process
```

PART 3: Average of a List

```
x = rand(1,500); %Creating 500 unit long vector with random numbers

l = length(x); %taking length of vector

[avgx] = avg(x,l); %calling function and inputing vector and vector length
to calculate avg value

disp(['Using my avg function: ',num2str(avgx)])

disp(['Using matlabs mean function: ',num2str(mean(x))])
```

PART 4: Calculating Pi

```
k = 0; %Assigning iteration to k
Pisum = 0; %Creating Pisum
a = zeros(1,20); %creating a vector of zeros to more efficiently store the
first 20 calculations
e = zeros(1,10000); %creating a 10000 unit long vector to store percent error
m = linspace(1,20,20); %Creating a 20 unit long vector with steps of 1 for
subplot axis

while abs(pi-(4*Pisum)) > 1e-4
    k = k + 1;
    temp = 0;
    temp = ((-1)^(k+1))/(2*k-1); %calculating
    Pisum = Pisum + temp; %adding k value to total sum
    e(1,k) = abs((((Pisum*4)-pi)/(pi))*100); %Calculating Percent error
    if k <= 20 %Stores the first 20 calculations in vector a
        a(1,k) = Pisum*4;
    end

end
Pisum = 4*Pisum;

l = linspace(1,k,k); %Creating a vector with length k (iterations) for scale
on plot

disp(['Estimated Value of Pi in ', num2str(k), ' iterations: ',
num2str(Pisum)])

figure;
subplot(1,2,1) %Creating sub plots with gridlines
plot(m, a)
hold on
title('π Estimation vs Iteration')
xlabel('Iteration #')
ylabel('π Estimate')
grid on;
subplot(1,2,2)
semilogx(l,e) %Log scale on x axis
grid on;
title('Percent Error vs Iteration')
xlabel('Iteration # (Log)')
ylabel('Percent Error')
```

Published with MATLAB® R2023b