AE 260 - HW 02

Total Points: / 100

Submission Instruction: Please submit this homework on Canvas in a single pdf format. The filename should be "HWXX_FullName_RedID.pdf" (ex. HW02_JamesGault_12345678.pdf).
Please copy your Matlab code in the given box. Adjust the box size as needed.
Please also submit all your m files separately. **Don't zip them**.

Full Name: Evan Peres

Red ID: 129964468

Email address: eperes2481@sdsu.edu

Iteration (Write your code in the box.)

1. Write a function with header [M] = myNMax(A,N) where M is an array consisting of the N largest elements of A. You may use MATLAB's max function. You may also assume that N is less than the length of M, that A is a one-dimensional array with no duplicate entries, and that N is a strictly positive integer smaller than the length of A. (/ 10)

Test Cases:

```
>> x = [7,9,10,5,8,3,4,6,2,1]
>> M = myNMax(x,3)
M=
    10    9    8
```

```
function [M] = myNMax(A,N)
   M=[];
   for k=1:N
      M = [M,max(A)];
      A(A == max(A))=[];
   end
end
```

2. The interest, i, on a principle, P0, is a payment for allowing the bank to use your money. Compound interest is accumulated according to the formula $P_n = (1 + i)P_{n-1}$, where n is the compounding period, usually in months or years. Write a function with header [years] = mySavingPlan(P0, i, goal) where years is the number of years it will take P0 to become goal at i% interest compounded annually. (/ 10)

Test Cases:

```
>> y = mySavingPlan(1000, 0.05, 2000)
```

```
y = 15
>> y = mySavingPlan(1000, 0.07, 2000)
y = 11
>> y = mySavingPlan(500, 0.07, 2000)
y = 21
```

```
function [years] = mySavingPlan(P0, i, goal)
   P = P0;
   years = 0;
   while P < goal
      years = years + 1;
      P = P*(1+i);
   end
end
```

3. A number is prime if it is divisible without remainder only by itself and 1. The number 1 is not prime. Write a function with header [out] = myIsPrime(n) where out is 1 if n is prime and 0 otherwise. You may assume that n is a strictly positive integer. Hint: Use pre-defined function “rem(a, b)” that returns the reminder after division of a by b. For example, the result “rem(6.5, 3)” is 0.5 (/ 10)

```
function [out] = myIsPrime(n)
   if n > 1
      for i = 2:((n/2)+1)
         if (rem(n,i) == 0) & (n/i ~= 1)
            out = 0;
            break
         else
            out = 1;
         end
      end
   else
      out = 0;
   end
end
```

1D array (Write your code in the box.)

4. Write a function with header [TemC] = myF2C(TemF) to convert fahrenheit to celsius. This function should work for one number, or an array of numbers to be converted. (/ 10)

```
function [TemC] = myF2C(TemF)
   TemC=[]
   for k=1:length(TemF)
      TemC = [TemC,(TemF-32)*(5/9)];
   end
end
```

5. Write a function with header[indices] = myWithinTolerance(A, a, tol) where indices is an array of the indices in A such that |A − a| < tol. You may assume that A is a one-dimensional double array and that a and tol are 1 × 1 doubles. (/ 10)

Test Cases:

```
>> I = myWithinTolerance([0 1 2 3], 1.5, .75)
I =
          2          3
>> I = myWithinTolerance(0: .01 : 1, 0.5, .03)
I =
     48   49  50  51  52   53
```

```
function [indices] = myWithinTolerance(A, a, tol)
   indices = []
   for k = 1:length(A)
      if abs(A(k)-a) < tol
         indices = [indices,k];
      end
   end
end
```

6. Write a function with header [boundedA] = myBoundingArray(A, top, bottom) where boundedA is equal to the array A wherever bottom < A < top, boundedA is equal to bottom wherever A <= bottom, and boundedA is equal to top wherever A >= top. You may assume that A is a one-dimensional double array and that top and bottom are 1 × 1 doubles. (/ 10)

Test Cases:

```
>> A = myBoundingArray(−5:5, 3, −3)

A =
          −3 −3 −3 −2 −1 0 1 2 3 3 3
```

```
>> x = linspace(0,2*pi,10)
>> A = myBoundingArray(sin(x), .5, −.5)

I =
     0    0.5000    0.5000    0.5000    0.3420   -0.3420  -0.5000   -0.5000   -0.5000   -0.0000
```

```
function [boundedA] = myBoundingArray(A, top, bottom)
   boundedA = []
   for k = 1:length(A)
      if A(k) < bottom
         boundedA = [boundedA,bottom];
      elseif A(k) > top
         boundedA = [boundedA,top];
      elseif (A(k) >= bottom) && (A(k) <= top)
         boundedA = [boundedA,A(k)];
      end
   end
end
```

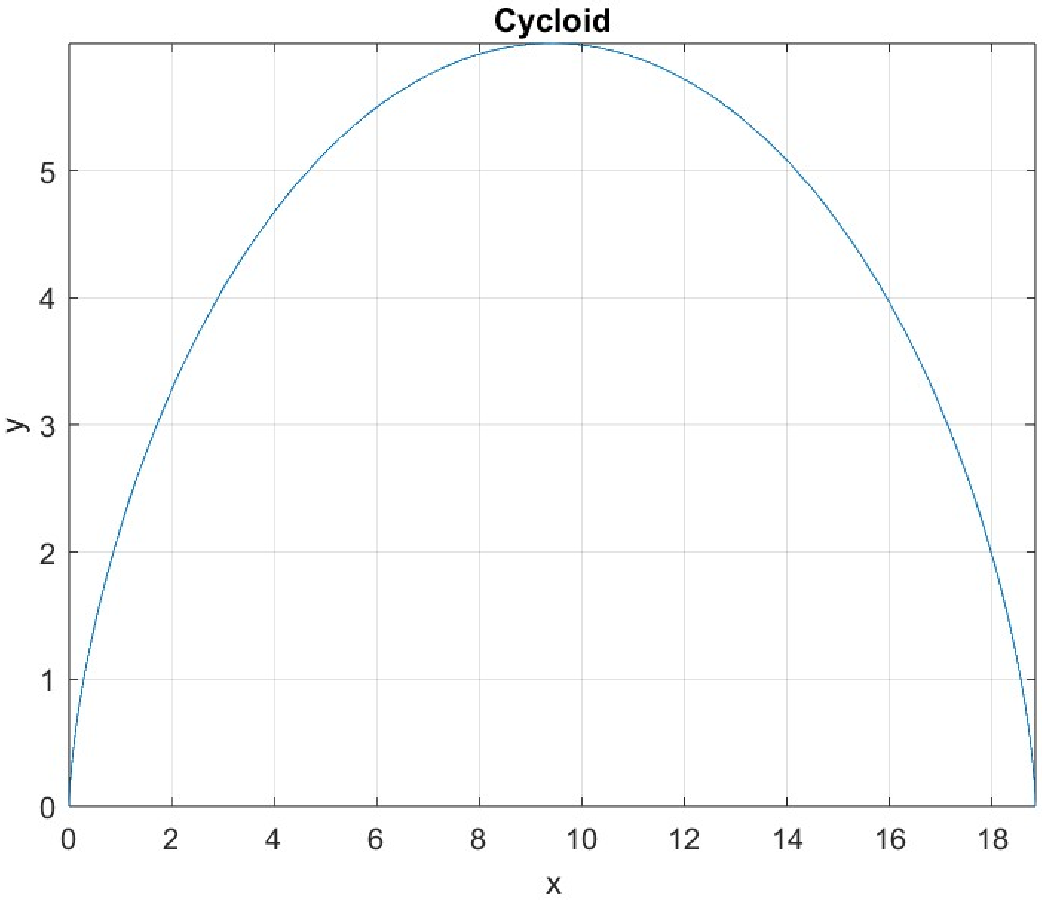
Plotting (Write your code in the box.)

7. A cycloid is the curve traced by a point located on the edge of a wheel rolling along a flat surface. The (x,y) coordinates of a cycloid generated from a wheel with radius, r, can be described by the parametric equations:

$$x = r(\emptyset - sin\emptyset)$$
$$y = r(1 - cos\emptyset)$$

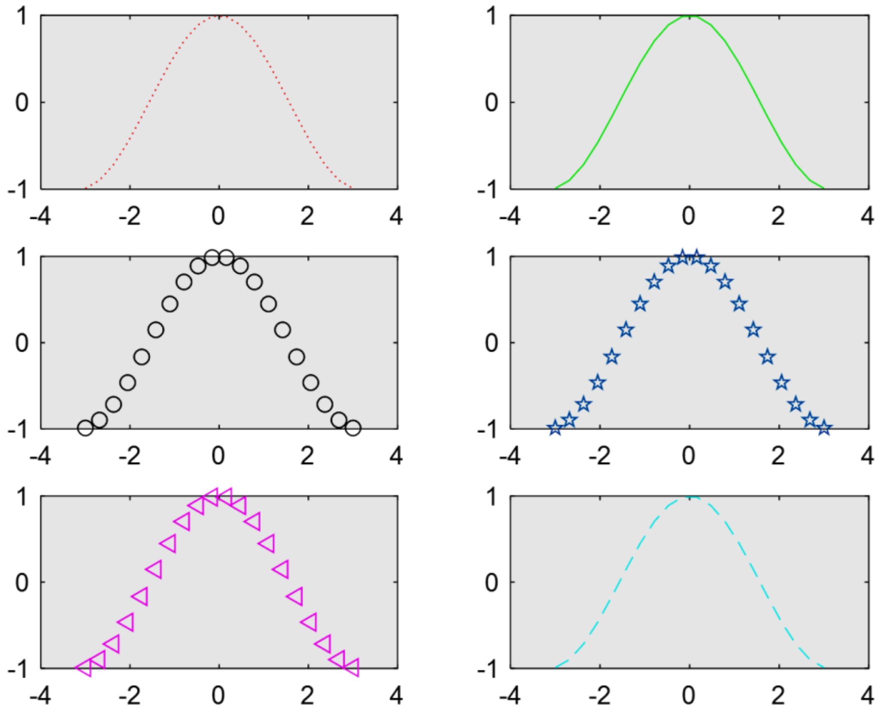
where ∅ is the number of radians that the wheel has rolled through.

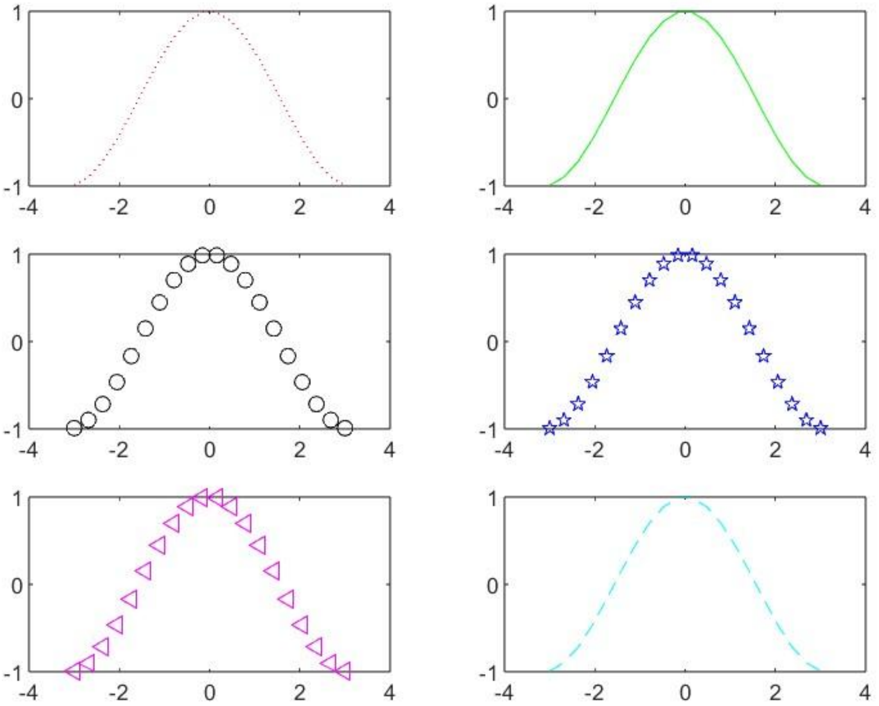
Generate a plot of the cycloid for 0 ≤ ∅ ≤ 2π using 1000 increments and r = 3. Give your plot a title 'Cycloid' and label the x-axis and y-axis with "x" and "y," respectively. Turn the grid on and modify the axis limits to make the plot neat. (/ 10)



```
r=3;
theta = linspace(0,2*pi,1000);
x = r*(theta-sin(theta));
y = r*(1-cos(theta));
plot(x,y)
title('Cycloid');
xlabel('x');
ylabel('y');
grid on;
xlim([min(x) max(x)]);
ylim([min(y) max(y)]);
```

8. Provide your scripts to create the following plot. All the subplots are y = cos(x) on the interval [-3,3] with 20 values in between. (/ 10)



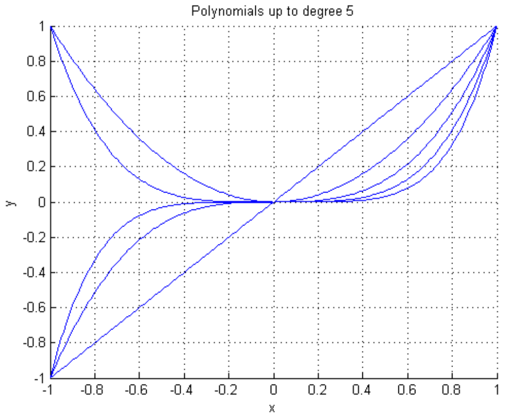


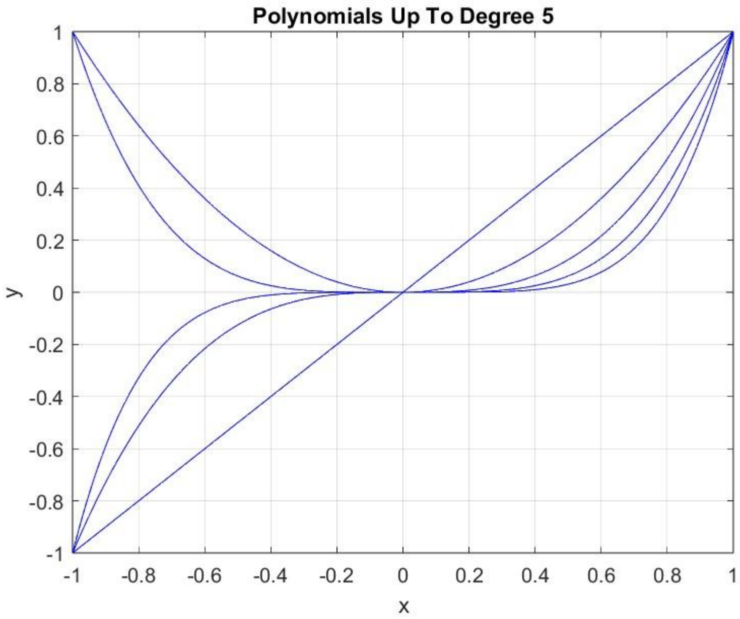
```
x=linspace(-3,3,20)
y=cos(x)

% red doted line
subplot(3,2,1);xlim([-4,4]);ylim([-1,1]);plot(x,y,":r")
% green line
subplot(3,2,2);xlim([-4,4]);ylim([-1,1]);plot(x,y,"g")
% black circle
subplot(3,2,3);xlim([-4,4]);ylim([-1,1]);plot(x,y,"ko")
% blue stars
subplot(3,2,4);xlim([-4,4]);ylim([-1,1]);plot(x,y,"b pentagram")
% magenta triangles
subplot(3,2,5);xlim([-4,4]);ylim([-1,1]);plot(x,y,"m<")
% cyan dashed line
subplot(3,2,6);xlim([-4,4]);ylim([-1,1]);plot(x,y,"--c")
```

9. Write a function with header [] = myPolyPlotter(n,x) that plots the polynomials $p_k(x) = x^k$ for k = 1,··· ,n. Make sure your plot has axis labels and a title., Hint: use hold on (/ 10)

Test Cases:

```
>> myPolyPlotter(5, -1: .01: 1 )
```



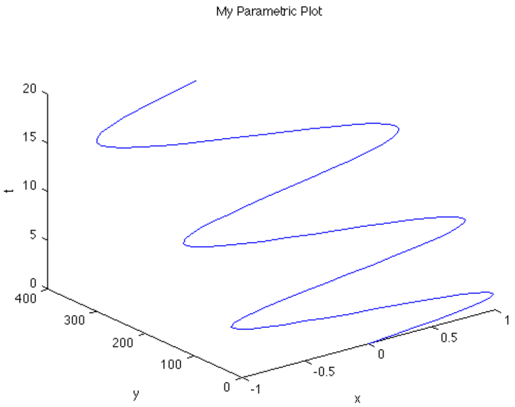


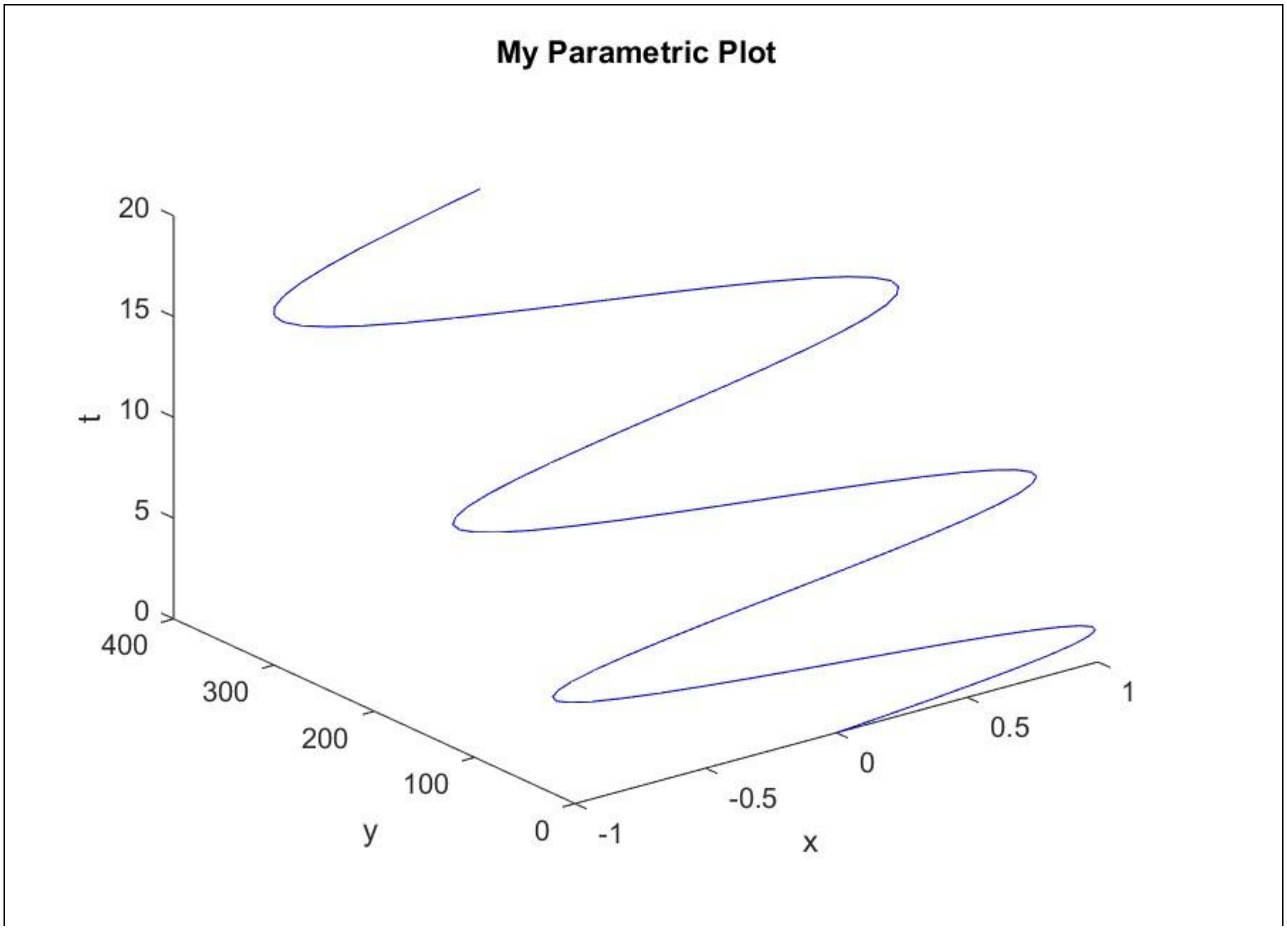
```
function [] = myPolyPlotter(n,x)
   y=[];
   for k = 1:n
      y = x.^k;
      plot(x, y,"b");
      hold on;
   end
   hold off;
   title(sprintf("Polynomials Up To Degree %d",n));
   xlabel('x');
   ylabel('y');
   grid on;
   xlim([min(x) max(x)]);
   ylim([min(y) max(y)]);
   xticks(min(x):.2:max(x));
   yticks(min(y):.2:max(y));
end
```

10. Write a function with header [ ] = myParametricPlotter(x,y,t) where x and y are handles to the functions x(t) and y(t), respectively, and t is a one-dimensional array. The function myParametricPlotter should produce the curve (x(t),y(t),t) in a three-dimensional plot. Be sure to give your plot a title and axis labels. (/ 10)

Test Cases:

```
>> f = @(t) sin(t);
>> g = @(t) t.^2;
>> myParametricPlotter(f,g,linspace(0,6*pi,100))
```





```
function [] = myParametricPlotter(x,y,t)

    X = x(t);

    Y = y(t);

    plot3(X,Y,t,'b')

    title('My Parametric Plot');

    xlabel('x');

    ylabel('y');

    zlabel('t');

end
```