Mathematics 3 File

Name: Ashish Gupta

Roll Number: 2K16/MC/023

Department of Applied Mathematics

Delhi Technological University

Content:

S. No	Practical	Signature
1		
2		
3		
4		
5		
6		
7		
8		
9		

Practical 1:

Write a program to determine the largest two Eigen values of the following matrix:

1	0	0	1	-1
0	2	3	5	0
-1	0	0	0	1
6	8	1	2	-2
1	1	1	1	1

Theory:

In linear algebra, an **eigenvector** or **characteristic vector** of a linear transformation is a non-zero vector that only changes by an overall scale when that linear transformation is applied to it. More formally, if T is a linear transformation from a vector space V over a field F into itself and $\mathbf{v}$ is a vector in V that is not the zero vector, then $\mathbf{v}$ is an eigenvector of T if $T(\mathbf{v})$ is a scalar multiple of $\mathbf{v}$. This condition can be written as the equation

Function Used:

Ans=eig(A), evalues all the eigen values in the given matrix and saves it as a list.

Matlab Code:

```
A=[1 0 0 -1 1; 0 2 3 5 0; -1 0 0 0 1; 1 8 1 2 -2 ; 1 1 1 1 1];

Ans=eig(A);

sort(Ans);

for i=1:2

   Ans(i,1)

end
```

Output:

```
ans =    8.1330

ans =   -4.2354
```

Practical 2:

W.A.P to show the consistency/ non consistency of the system of linear equations. If the system is consistent, then write a program to solve the given system of equations for unique/ infinite solutions:

System Given: AX=B

A=

1	**2**	**1**
2	**3**	**5**
7	**1**	**2**

B=

5
7
0

Function Used:

`RankofA=rank(A);` Evaluates the rank of matrix A
`RankOfAug=rank(Aug);` Evaluates the rank of matrix Aug(Augmented Matrix)
`X=inv(A);` Evaluates the inverse of matrix A

Matlab Code:

```
A=[1 2 1;2 3 5;7 1 2];
B=[5;7;0];
Aug=[A,B];
flag=0;
RankofA=rank(A);
RankOfAug=rank(Aug);
if(RankofA~=RankOfAug)
    fprintf('The given system of equations is
inconsistent')
    flag=1;
else
        fprintf('The given system is consistent\n')

end
clear X;
if(flag==0)
```

```
    % fprintf('\n')
    X=inv(A);

    Ans=(X*B)
End
```

Output:

>> Practical2_ConsistencyOfMatrix

The given system is consistent

Ans =

-0.3636

2.7273

-0.0909

Practical 3:

Using inbuilt ode solver 23 and ODE 45, find y(0.3), where y is solution of the following initial value problem abd hence compare this value to the original value:

System Given: dy/dx= y+x, y(0) = 1

Function Used:

Eqn(y,t) gets the differential equation to be input into the ode23 inbuilt solver.

ODE 23 and ODE 45 A function handle that evaluates the right side of the differential equations. All solvers solve systems of equations in the form $y' = f(t,y)$ or problems that involve a mass matrix, $M(t,y)y' = f(t,y)$. The ode23s solver can solve only equations with constant mass matrices. ode15s and ode23t can solve problems with a mass matrix that is singular, i.e., differential-algebraic equations (DAEs).

Tspan: Defines a range for ode solvers.

Matlab Code:

ODE-23 Solver:

```
function Value = eqn( y,t )

    Value=y+t;
end
syms x y t;

y0=1;
Value=eqn(x,y);
yspan=[-5,5];
[x,y]=ode23(value,yspan,y0);
%output
plot(x,y,'r*');
y1=y(3)
grid on;
title('using ode 23')
```

Output:

>> Practical2_ConsistencyOfMatrix

The given system is consistent

Ans =

-0.3636

2.7273

-0.0909

ODE-45 Solver:

```
function Value = eqn( y,t )

    Value=y+t;
end
syms x y t;

y0=1;
Value=eqn(x,y);
yspan=[-5,5];
[x,y]=ode23(value,yspan,y0);
%output
plot(x,y,'r*');
y1=y(3)
grid on;
title('using ode 23')
```

Output:

>> Practical2_ConsistencyOfMatrix

The given system is consistent

Ans =

-0.3636

2.7273

-0.0909

Practical 4:

Write a program to solve D^2(y) +4*y=Sec(x) by using the method of variation of parameters

Theory:

Variation of Parameters

Consider the differential equation,

$$y'' + q(t)\,y' + r(t)\,y = g(t)$$

$$y'' + q(t)\,y' + r(t)\,y = g(t)$$

Assume that $y_1(t)$ and $y_2(t)$ are a fundamental set of solutions for

$$y'' + q(t)\,y' + r(t)\,y = 0$$

$$y'' + q(t)\,y' + r(t)\,y = 0$$

Then a particular solution to the nonhomogeneous differential equation is,

$$Y_P(t) = -y_1 \int \frac{y_2 g(t)}{W(y_1, y_2)}\,dt + y_2 \int \frac{y_1 g(t)}{W(y_1, y_2)}\,dt$$

$$Y_P(t) = -y_1 \int \frac{y_2 g(t)}{W(y_1, y_2)}\,dt + y_2 \int \frac{y_1 g(t)}{W(y_1, y_2)}\,dt$$

Function Used:

Eqn(y,t) gets the differential equation to be input into the ode23 inbuilt solver.

ODE 23 and ODE 45 A function handle that evaluates the right side of the differential equations. All solvers solve systems of equations in the form $y' = f(t,y)$ or problems that involve a mass matrix, $M(t,y)y' = f(t,y)$. The ode23s solver can solve only equations with constant mass matrices. ode15s and ode23t can solve problems with a mass matrix that is singular, i.e., differential-algebraic equations (DAEs).

Tspan: Defines a range for ode solvers.

Matlab Code:

```
syms x t;
cf=dsolve('D2y+4*y=0');
y1dot=diff('cos(2*t)',t);
y2dot=diff('sin(2*t)',t);

w=[cos(2*t) sin(2*t) ;y1dot y2dot];
wdet=det(w);
w1=[0 sin(2*t); sec(t) y2dot];
w1det=det(w1);
w2=[cos(2*t) 0 ;y1dot sec(t)];
w2det=det(w2);
u=int(w1det/wdet);
v=int(w2det/wdet);
perticular_solution=u*cos(2*t)+v*sin(2*t);
Answer=cf+perticular_solution
```

Output:

```
>> method_VariationOfParameters

 Answer =

 cos(2*t)*cos(t) + C3*cos(2*t) + C4*sin(2*t) - sin(2*t)*(atanh(sin(t))/2 - sin(t))
```

Practical 5:

Graphically compare the function sin(x) and Taylor series expansion of sin(x) up to degree 10 in the neighbourhood of 1.

Theory:

The formula for the Taylor series expansion for sin(x) is :

$$\sin(x) = \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)!} x^{2k+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \ldots$$

In mathematics, a **Taylor series** is a representation of a function as an infinite sum of terms that are calculated from the values of the function's derivatives at a single point.

Function Used:

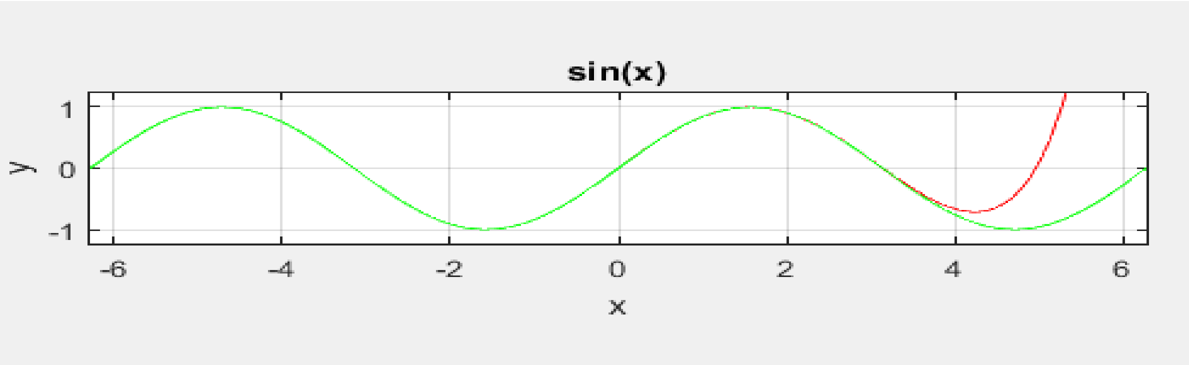
Taylor (`f`) computes the taylor's series expansion of 'f' up to the fifth order. The expansion point is 0.
Taylor (**`f,Name,Value`**) uses additional options specified by one or more `Name`, `Value` pair arguments.
Ezplot(x,f) plots the function's graph.

Matlab Code:

```
syms f x;
f=taylor(sin(x),'Order',11);% it represents the powers
from 0
%all the way upto 10
h=ezplot(x,f);
set(h,'color','r');
grid on;
hold on;
y=sin(x);
plot2=ezplot(y);
set(plot2,'color','g');
hold off;
```

Output:

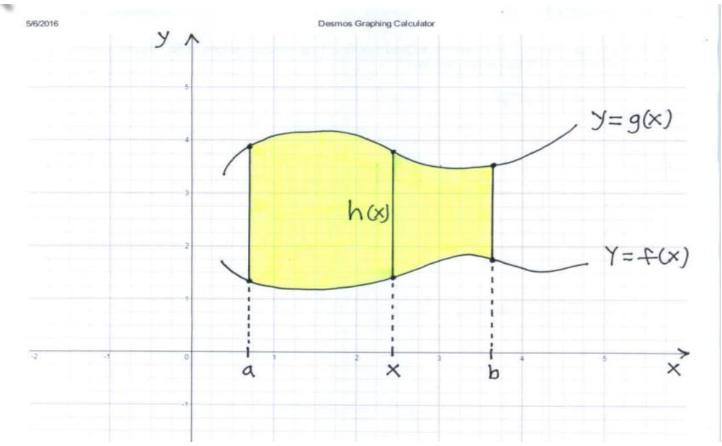
```
>> taylorSeries_Sinx

f =

x^9/362880 - x^7/5040 + x^5/120 - x^3/6 + x
```



Practical 6:

Sketch the area/ region enclosed by the curves f(x)= x^3 – 3*x^2 + 3*x and g(x)=x^2 and find the area of the enclosed region.

Theory:



The area of RR is given by:

$\boxed{\textbf{AREA}=\int \textbf{ba } h(x)dx=\int \textbf{ba } (g(x)-f(x))dx}$ (ba is integration limits from a to b)

Function Used:

Ezplot(x,f) plots the function's graph.
area1=int(f1-f2,0,1);
area2=int(f2-f1,1,3);
int=int(x) (integrate is used to integrate a function from limits initial to final)

Matlab Code:

```
syms x t;
f1=x^3- 3*x^2 +3*x;
f2=x^2;
plot1=ezplot(f1);
set(plot1,'color','r');
hold on;
plot2=ezplot(f2);
```

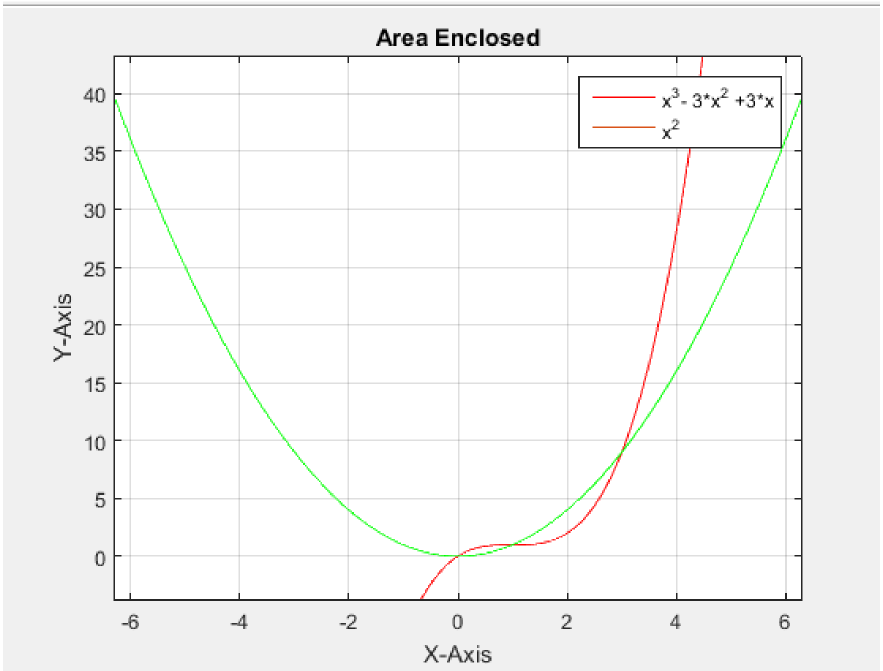
```
set(plot2,'color','g');
grid on;
legend('x^3- 3*x^2 +3*x','x^2');
xlabel('X-Axis');
ylabel('Y-Axis');
title('Area Enclosed');
area1=int(f1-f2,0,1);
area2=int(f2-f1,1,3);
area=area1+area2
```

Output:

>> area_Enclosed

area =

37/12



Practical 7:

To draw the tangent line at point on a given curve y=1+x2 at the point (2,5) and also find the Radius of curvature at that point..

Theory:

A **tangent line** is a line that touches a curve at a single point and does not cross through it. The point where the curve and the tangent meet is called the point of tangency. We know that for a line **y=m*x+c** its slope at any point is **m**. The same applies to a curve. When I say the slope of a curve, I mean the slope of tangent to the curve at a point.

Radius of curvature is :

$$\rho(t) = \frac{|1 + f'^2(t)|^{3/2}}{|f''(t)|}.$$

Function Used:

ezplot(FUN) plots the function FUN(X) over the default domain -2*PI < X < 2*PI, where FUN(X) is an explicitly defined function of X.

ezplot(FUN2) plots the implicitly defined function FUN2(X,Y) = 0 over the default domain -2*PI < X < 2*PI and -2*PI < Y < 2*PI.

sqrt(X) is the square root of the elements of X.

Matlab Code:

```
syms x t;
f1=1+x^2;
d(x)=diff(f1);
m=d(2);
c=5-m*2;
f2=m*x+c;
plot1=ezplot(f1);
set(plot1,'color','r');
hold on;
grid on;
plot2=ezplot(f2);
legend('1+x^2','y=m*x+c');
```

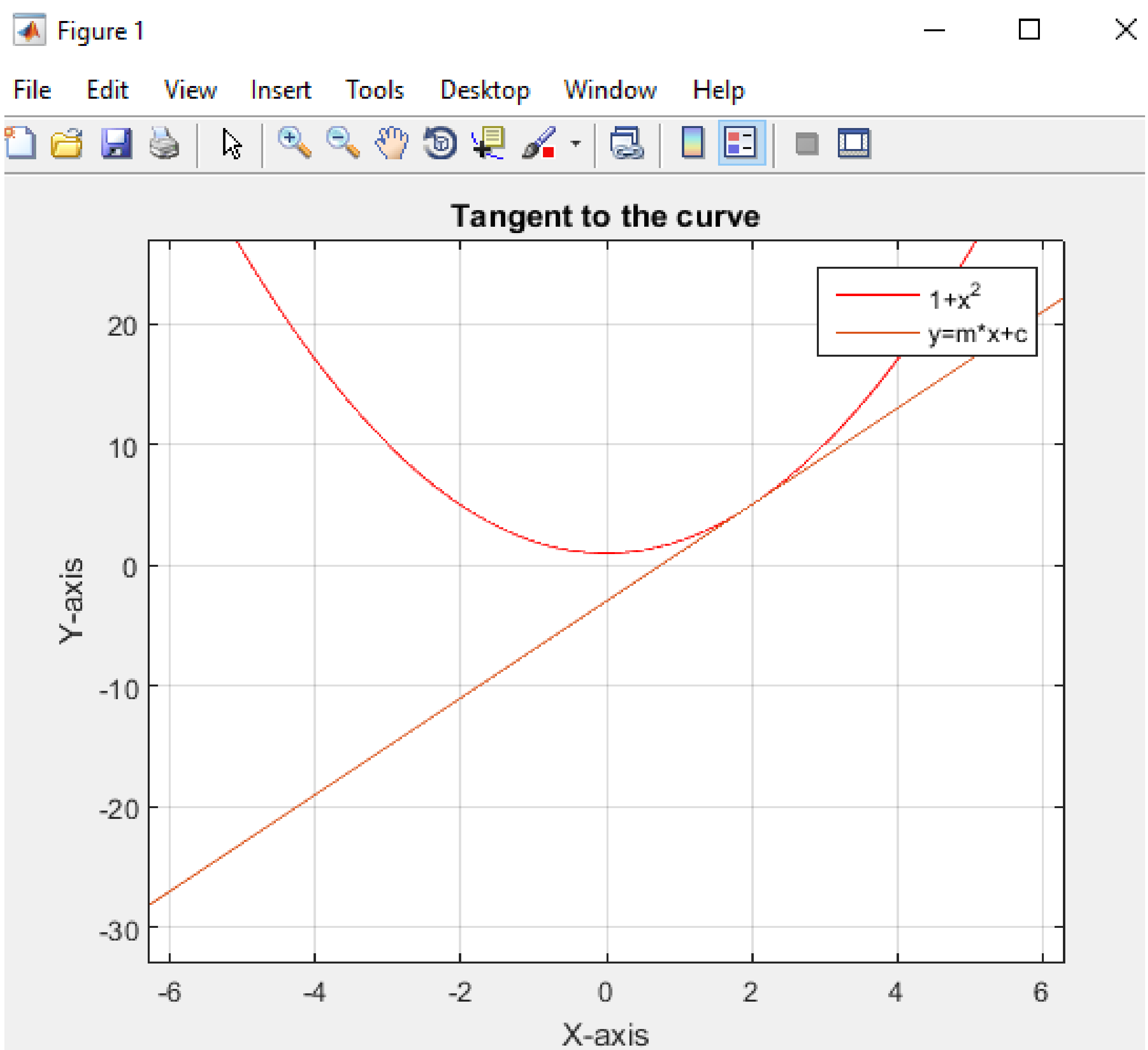
```
xlabel('X-axis');
ylabel('Y-axis');
title('Tangent to the curve');
ydot2=diff(diff(f1));
roc=sqrt((1+m^2)^3)/2
hold off;
```

Output:

>> tangent_to_a_curve

roc =

(17*17^(1/2))/2



Practical 8:

Plot the surface defined by the function $f(x, y) = -xye^{-2(x^2+y^2)}$ on the domain $-2 \le x \le 2$ and $-2 \le y \le 2$. Find the values and locations of the maxima and minima of the function.

Theory:

Maxima and Minima of Functions of Two Variables:

Locate relative maxima, minima and saddle points of functions of two variables. Several examples with detailed solutions are presented. 3-Dimensional graphs of functions are shown to confirm the existence of these points. More on Optimization Problems with Functions of Two Variables in this web site.

Theorem:

Let f be a function with two variables with continuous second order **partial derivatives** f_{xx}, f_{yy} and f_{xy} at a critical point (a,b). Let

$$D = f_{xx}(a,b)\, f_{yy}(a,b) - f_{xy}^2(a,b)$$

a. If $D > 0$ and $f_{xx}(a,b) > 0$, then f has a relative minimum at (a,b).
b. If $D > 0$ and $f_{xx}(a,b) < 0$, then f has a relative maximum at (a,b).
c. If $D < 0$, then f has a saddle point at (a,b).
d. If $D = 0$, then no conclusion can be drawn.

Function Used:

ezplot(FUN) plots the function FUN(X) over the default domain -2*PI < X < 2*PI, where FUN(X) is an explicitly defined function of X.

ezplot(FUN2) plots the implicitly defined function FUN2(X,Y) = 0 over the default domain -2*PI < X < 2*PI and -2*PI < Y < 2*PI.

Plot: Plots the graph of the desired function, with valid inputs

Meshgrid: Used to plot graph of 3D plots, functions of 2 independent variables.

Max & Min: Calculate max/min of a set of values.

Matlab Code:

```
clc;
clear;
syms x y;
[x,y]=meshgrid(-2:0.03:2,-2:0.03:2);
f=-x.*y.*exp(-2*(x.^2+y.^2));
figure(1)
mesh(x,y,f),xlabel('X'),ylabel('y'),grid
figure(2)
contour(x,y,f)
xlabel('X'),ylabel('y'),grid,hold on
fmax=max(max(f))
kmax=find(f==fmax)
pos=[x(kmax)  y(kmax)]
%pos = -0.5000 0.5000 0.5000 -0.5000

plot(x(kmax),y(kmax),'*')
text(x(kmax),y(kmax),'Maximum')
%plotting the maximum value on the graph
fmin=min(min(f))
kmin=find(f==fmin)
pos1=[x(kmin)  y(kmin)]
plot(x(kmin),y(kmin),'*')
% We are plotting the minimum value now
text(x(kmin),y(kmin),'Minimum')
```

Output:

fmax =

0.0919

kmax =

6784

11173

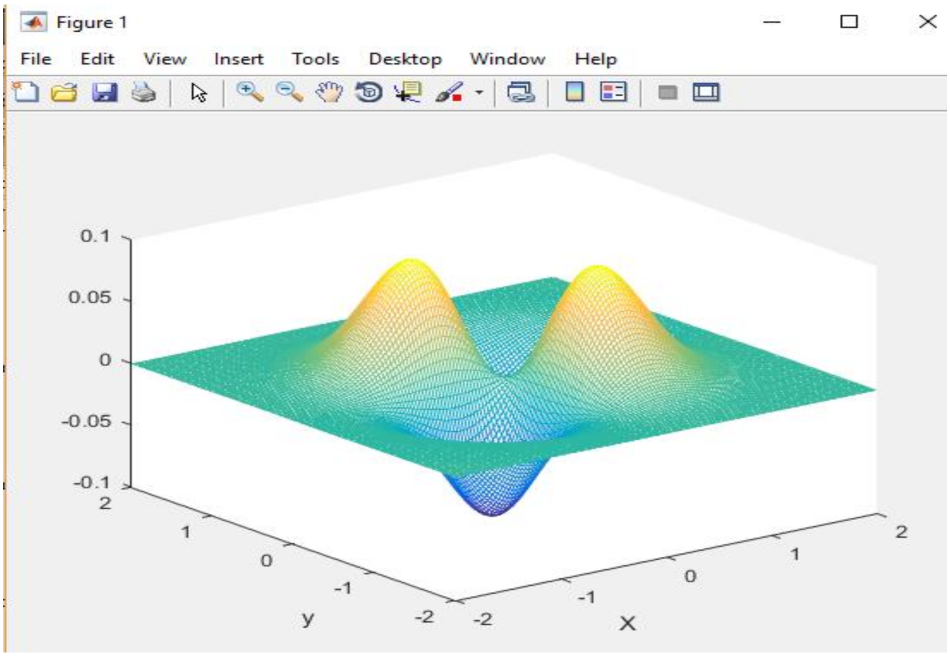
pos =

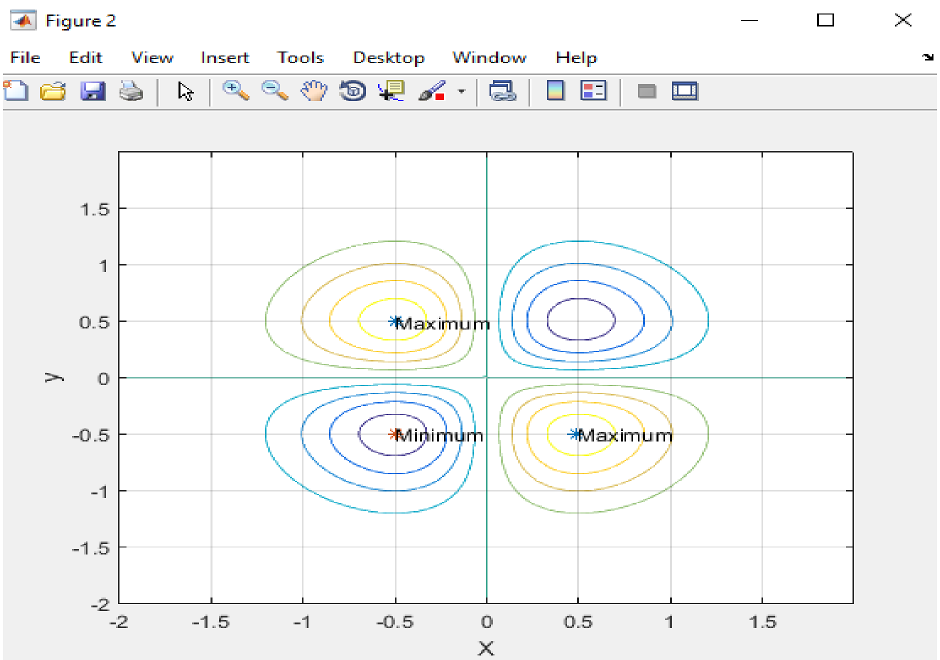
-0.5000 0.4900

0.4900 -0.5000

fmin =

-0.0920





Practical 9:

Determine the characteristic polynomial of a matrix evaluating the polynomial p(λ) at the nth points.

Theory:

In linear algebra, the **characteristic polynomial** of a square matrix is a polynomial which is invariant under matrix similarity and has the eigenvalues as roots. It has the determinant and the trace of the matrix as coefficients. The **characteristic polynomial** of an endomorphism of vector spaces of finite dimension is the characteristic polynomial of the matrix of the endomorphism over any base; it does not depend on the choice of a basis.

The **characteristic equation** is the equation obtained by equating to zero the characteristic polynomial.

Function Used:

ezplot(FUN) plots the function FUN(X) over the default domain -2*PI < X < 2*PI, where FUN(X) is an explicitly defined function of X.

ezplot(FUN2) plots the implicitly defined function FUN2(X,Y) = 0 over the default domain -2*PI < X < 2*PI and -2*PI < Y < 2*PI.

sqrt(X) is the square root of the elements of X.

Matlab Code:

```
function [ co ] = MyFunction( A )
[m n]=size(A) ;
if m~=n
  disp('It is not a Square Matrix')
  co=[] ;
  return
end
for i=1:(n+1)
    x(i)=(i-1)*pi/n;
    y(i)=det(A-x(i)*eye(n));
```

```
end
co=polyfit(x,y,n);

A=[1 2 3; 4 5 6 ; 7 8 9];
MyFunction(A)
z=length(ans);
syms x;
f=0;
i=4;
for y=1:1:z
    f=f+ans(y).*x.^(i-1);
    i=i-1;
end
f
```

Output:

f =

- x^3 + 15*x^2 + 18*x