

# DELHI TECHNOLOGICAL UNIVERSITY



## MATHEMATICAL MODELLING AND SIMULATION LAB FILE

Submitted To:

**Dr. Nilam Rathi**  
*Assistant Professor*

**Ms. Mridulla**

Submitted by -

**Sachin Duhan**  
2K17/MC/087

# **DELHI TECHNOLOGICAL UNIVERSITY**

## **VISION**

TO BE A WORLD CLASS UNIVERSITY THROUGH EDUCATION, INNOVATION, AND  
RESEARCH FOR THE SERVICE OF HUMANITY.

## **MISSION**

- ◆ TO ESTABLISH CENTRES OF EXCELLENCE IN EMERGING AREAS OF SCIENCE, ENGINEERING, TECHNOLOGY, MANAGEMENT AND ALLIED AREAS.
- ◆ TO FOSTER AN ECOSYSTEM FOR INCUBATION, PRODUCT DEVELOPMENT, TRANSFER OF TECHNOLOGY AND ENTREPRENEURSHIP.
- ◆ TO CREATE ENVIRONMENT OF COLLABORATION, EXPERIMENTATION, IMAGINATION AND CREATIVITY.
- ◆ TO DEVELOP HUMAN POTENTIAL WITH ANALYTICAL ABILITIES, ETHICS AND INTEGRITY.
- ◆ TO PROVIDE ENVIRONMENT FRIENDLY, REASONABLE AND SUSTAINABLE SOLUTIONS FOR LOCAL AND GLOBAL NEEDS.

# **DEPARTMENT OF APPLIED MATHEMATICS**

## **VISION**

TO EMERGE AS A CENTRE OF EXCELLENCE AND EMINENCE BY IMPARTING FUTURISTIC TECHNICAL EDUCATION WITH SOLID MATHEMATICAL BACKGROUND IN KEEPING WITH GLOBAL STANDARDS, MAKING OUR STUDENTS TECHNOLOGICALLY AND MATHEMATICALLY COMPETENT AND ETHICALLY STRONG SO THAT THEY CAN READILY CONTRIBUTE TO THE RAPID ADVANCEMENT OF SOCIETY AND MANKIND

## **MISSION**

- TO ACHIEVE ACADEMIC EXCELLENCE THROUGH INNOVATIVE TEACHING AND LEARNING PRACTICES.
- TO IMPROVE THE RESEARCH COMPETENCE TO ADDRESS SOCIAL NEEDS.
- TO INCULCATE A CULTURE THAT SUPPORTS AND REINFORCES ETHICAL, PROFESSIONAL BEHAVIOURS FOR A HARMONIOUS AND PROSPEROUS SOCIETY.
- STRIVE TO MAKE STUDENTS TO UNDERSTAND, APPRECIATE AND GAIN MATHEMATICAL SKILLS AND DEVELOP LOGIC, SO THAT THEY ARE ABLE TO CONTRIBUTE INTELLIGENTLY IN DECISION MAKING WHICH CHARACTERISES OUR SCIENTIFIC AND TECHNOLOGICAL AGE.

# **PROGRAMME EDUCATIONAL OUTCOMES**

- TO PREPARE GRADUATES WITH A SOLID FOUNDATION IN ENGINEERING, MATHEMATICAL SCIENCE AND TECHNOLOGY FOR A SUCCESSFUL CAREER IN MATHEMATICS AND COMPUTING / FINANCE / COMPUTER ENGINEERING FIELDS.
- TO PREPARE GRADUATES TO BECOME EFFECTIVE COLLABORATORS / INNOVATORS, WHO COULD ABLY ADDRESS TOMORROW'S SOCIAL, TECHNICAL AND ENGINEERING CHALLENGES.
- TO ENRICH GRADUATES WITH INTEGRITY AND ETHICAL VALUES SO THAT THEY BECOME RESPONSIBLE ENGINEERS.

# PROGRAMME OUTCOMES

1. ENGINEERING KNOWLEDGE: THE GRADUATE OF MATHEMATICS & COMPUTING MUST HAVE AN ABILITY TO APPLY KNOWLEDGE OF MATHEMATICS, BASIC SCIENCE AND COMPUTER SCIENCE TO SOLVE ENGINEERING AND RELATED PROBLEMS.
2. PROBLEM ANALYSIS: AN ABILITY TO IDENTIFY, ANALYZE AND FORMULATE COMPLEX ENGINEERING PROBLEMS TO REACH LOGICAL CONCLUSION.
3. DESIGN/DEVELOPMENT OF SOLUTION: AN ABILITY TO DESIGN AND CONDUCT EXPERIMENTS, ANALYZE AND INTERPRET THE DATA.
4. CONDUCT INVESTIGATIONS OF COMPLEX PROBLEMS: AN ABILITY TO USE RESEARCH-BASED KNOWLEDGE AND APPLY RESEARCH METHODS TO PROVIDE VALID CONCLUSION.
5. MODERN TOOL USAGES: AN ABILITY TO CREATE, SELECT AND IMPLEMENT APPROPRIATE TECHNIQUES, SUCH AS ARTIFICIAL INTELLIGENCE, NEURAL NETWORK TO MODEL COMPLEX COMPUTER ENGINEERING ACTIVITY.
6. THE ENGINEER AND SOCIETY: AN ABILITY TO EXPLORE THE IMPACT OF ENGINEERING SOLUTIONS ON THE SOCIETY AND ALSO ON CONTEMPORARY ISSUES ON SOCIETAL AND ENVIRONMENTAL CONTEXT.
7. ENVIRONMENT AND SUSTAINABILITY: AN ABILITY TO DESIGN A FEASIBLE SYSTEM, COMPONENT OR PROCESS WITHOUT VIOLATING NORMS FOR PUBLIC HEALTH AND SAFETY, CULTURAL, SOCIAL AND ENVIRONMENTAL ISSUES.
8. ETHICS: AN ABILITY TO UNDERSTAND AND PRACTICE PROFESSIONAL AND ETHICAL RESPONSIBILITIES. INDIVIDUAL AND TEAM WORKS: AN ABILITY TO FUNCTION EFFECTIVELY AS AN INTEGRAL MEMBER OR A LEADER IN A MULTIDISCIPLINARY TEAM.
9. COMMUNICATION: AN ABILITY TO COMMUNICATE EFFECTIVELY IN BOTH ORAL AND WRITTEN FORM FOR EFFECTIVE TECHNICAL DECISION MAKING, REPORT MAKING AND PRESENTATION.
10. PROJECT MANAGEMENT AND FINANCE: AN ABILITY TO DEMONSTRATE PRINCIPLE OF MANAGEMENT AND APPLY THEM TO SUITABLE PROJECTS.
11. LIFE LONG LEARNING: AN ABILITY TO RECOGNIZE THE NEED FOR AND TO BE READY FOR LIFE LONG LEARNING TO KEEP UPDATED ON TECHNOLOGICAL CHANGES.

# INDEX

[illegible]

# Practical 1

**AIM - SOLVE THE GIVEN SET OF DIFFERENTIAL EQUATION USING OCTAVE**

$$2x^2 \frac{d^2 y}{dx^2} + 3x \frac{dy}{dx} + y = e^x$$

$$5x \frac{dy}{dx} + 2y = \sin 2x$$

**SOLUTION -**

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% FILE 1 FOR EXPERIMENT 1 - MS LAB
syms y(x)
eqn=diff(y,2) == 1/2*x^2*(exp(x) - 3*diff(y)-y)
S=dsolve(eqn)
```

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% FILE 2 FOR EXPERIMENT 1
syms y(x)
eqn = diff(y,x) == (sin(2^x) - 2*y)/5*x
S = dsolve(eqn)
```

```
>> a1
S =
x*exp(-(3*x^4)/8)*hypergeom(11/12, 5/4, (3*x^4)/8)*int((15*x^2*exp((9*x^4)/16 + x)*whittakerM(-7/24, -1/8, (3*x^4)/8))/((3*x^4)/8)^(3/8)*(30*exp((3*x^4)/16)*hyperg
f> >> |
```

```

Command Window
New to MATLAB? See resources for Getting Started
>> a1
S =
x*exp(-(3*x^4)/8)*hypergeom(11/12, 5/4, (3*x^4)/8)*int((15*x^2*exp((9*x^4)/16 + x)*whittakerM(-7/24, -1/8, (3*x^4)/8))/(((3*x^4)/8)^(3/8))*((30*exp((3*x^4)/16)*hyperg
>> clear
>> a2
S =
exp(-x^2/5)*int((x*sin(2*x)*exp(x^2/5))/5, x, 'IgnoreSpecialCases', true, 'IgnoreAnalyticConstraints', true) + C1*exp(-x^2/5)

```



# Practical 2

**AIM - SOLVE THE GIVEN SYSTEM OF DIFFERENTIAL EQUATION USING OCTAVE**

a.)

$$\begin{aligned} 3x + 2y - z &= 1 \\ 2x - 2y + 4z &= -2 \\ -x + \frac{1}{2}y - z &= 0 \end{aligned}$$

b.)

$$\begin{aligned} 3x - y + 4z &= 2 \\ 17x + 2y + z &= 14 \\ x + 12y - 77z &= 54 \end{aligned}$$

**SOLUTION -**

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% DIFFERENTIAL EQUATION 1 FOR EXPERIMENT 2

sysm x y z
e1 = 3*x + 2*y - z == 1;
e2 = 2*x - 2*y + 4*z == -2;
e3 = -x+(1/2)*y-z == 0;

solution = solve([e1,e2,e3],[x,y,z]);
xsol = solution.x
ysol = solution.y
zsol = solution.z
```

```
>> b1
xsol =
1
ysol =
-2
zsol =
-2
```

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% DIFFERENTIAL EQUATION 2 FOR EXPERIMENT 2

syms x y z
e1 = 3*x + y - 4*z == 2;
e2 = 17*x - 2*y + z == 14;
e3 = x + (12)*y + 7*z == 54;

solution = solve([e1,e2,e3],[x,y,z]);
xsol = solution.x
ysol = solution.y
zsol = solution.z
```

```
>> D2
xsol =
    72/233
ysol =
    578/233
zsol =
    382/233
```

## CODE -

```
%% DIFFERENTIAL EQUATION 1 FOR EXPERIMENT 2
```

```
syms x y z
```

```
e1 = 3*x + 2*y - z == 1;
```

```
e2 = 2*x - 2*y + 4*z == -2;
```

```
e3 = -x+(1/2)*y-z == 0;
```

```
solution = solve([e1,e2,e3],[x,y,z]);
```

```
xsol = solution.x
```

```
ysol = solution.y
```

```
zsol = solution.z
```

```
%% DIFFERENTIAL EQUATION 1 FOR EXPERIMENT 2
```

```
sysm x y z
```

```
e1 = 3*x + 2*y - z == 1;
```

```
e2 = 2*x - 2*y + 4*z == -2;
```

```
e3 = -x+(1/2)*y-z == 0;
```

```
solution = solve([e1,e2,e3],[x,y,z]);
```

```
xsol = solution.x
```

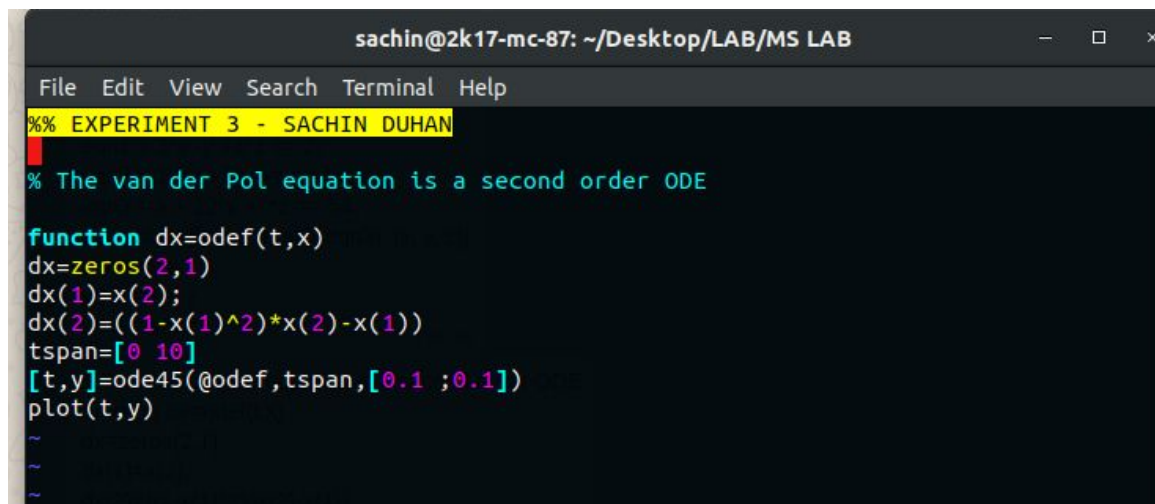
```
ysol = solution.y
```

```
zsol = solution.z
```

## Practical 3

**AIM** - Consider any real-life problem and write a differential equation for that. Further, solve it.

$$y_1'' - \mu (1 - y_1^2) y_1' + y_1 = 0,$$



```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% EXPERIMENT 3 - SACHIN DUHAN
% The van der Pol equation is a second order ODE

function dx=odef(t,x)
dx=zeros(2,1)
dx(1)=x(2);
dx(2)=((1-x(1)^2)*x(2)-x(1))
tspan=[0 10]
[t,y]=ode45(@odef,tspan,[0.1 ;0.1])
plot(t,y)
~
~
~
```

### CODE -

% The van der Pol equation is a second order ODE

```
function dx=odef(t,x)
```

```
dx=zeros(2,1)
```

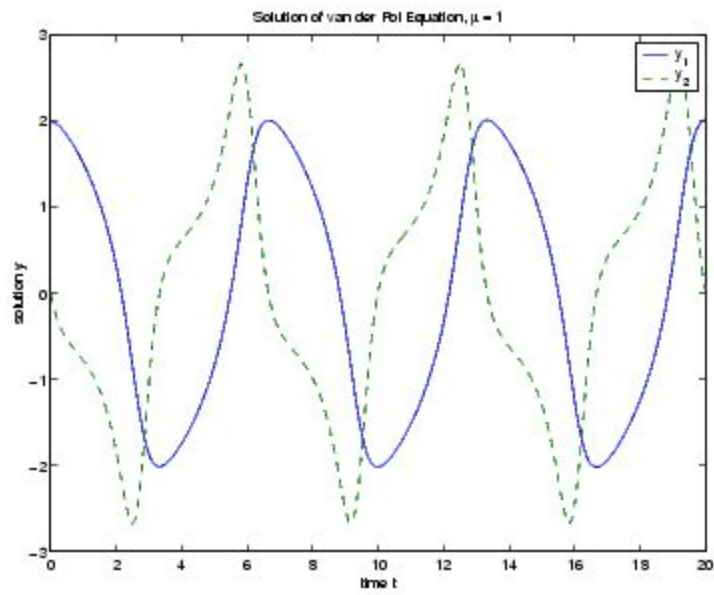
```
dx(1)=x(2);
```

```
dx(2)=((1-x(1)^2)*x(2)-x(1))
```

```
tspan=[0 10]
```

```
[t,y]=ode45(@odef,tspan,[0.1 ;0.1])
```

```
plot(t,y)
```

**OUTPUT -**

# Practical 4

**AIM** - Consider any real-life problem and write a differential equation for that. Further, solve it.

$$\frac{dx}{dt} = x + 2y$$

$$\frac{dy}{dt} = y + 2x$$

CODE -

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% SOLUTION OF THE PRACTICAL 4 EQUATION - 1
syms
eq1 = diff(x) == x + 2*y;
eq2 = diff(y) == 2*x + y;

odes = [eq1; eq2];
ans = dsolve(odes)
xSol(t) = ans.x
ySol(t) = ans.y
[xSol(t), ySol(t)] = sdolve(odes)
cond1 = x(0) == 0
cond2 = y(0) == 1
conds = [cond1; cond2];
[xSol(t), ySol(t)] = dsolve(odes,conds);
fplot(xSol)
hold on
fplot(ySol)
grid on
legend('xSol','ySol','Location','best');
```

**OUTPUT -**

```
odes(t) =

diff(x(t), t) == x(t) + 2*y(t)
diff(y(t), t) == 2*x(t) + y(t)
```

```
S =
```

```
struct with fields:
```

```
  y: [1×1 sym]
  x: [1×1 sym]
```

```
xSol(t) =
```

```
C6*exp(3*t) - C5*exp(-t)
```

```
ySol(t) =
```

```
C5*exp(-t) + C6*exp(3*t)
```

```
xSol(t) =
```

```
C6*exp(3*t) - C5*exp(-t)
```

```
ySol(t) =
```

```
C5*exp(-t) + C6*exp(3*t)
```

```
xSol(t) =
```

```
C6*exp(3*t) - C5*exp(-t)
```

```
ySol(t) =
```

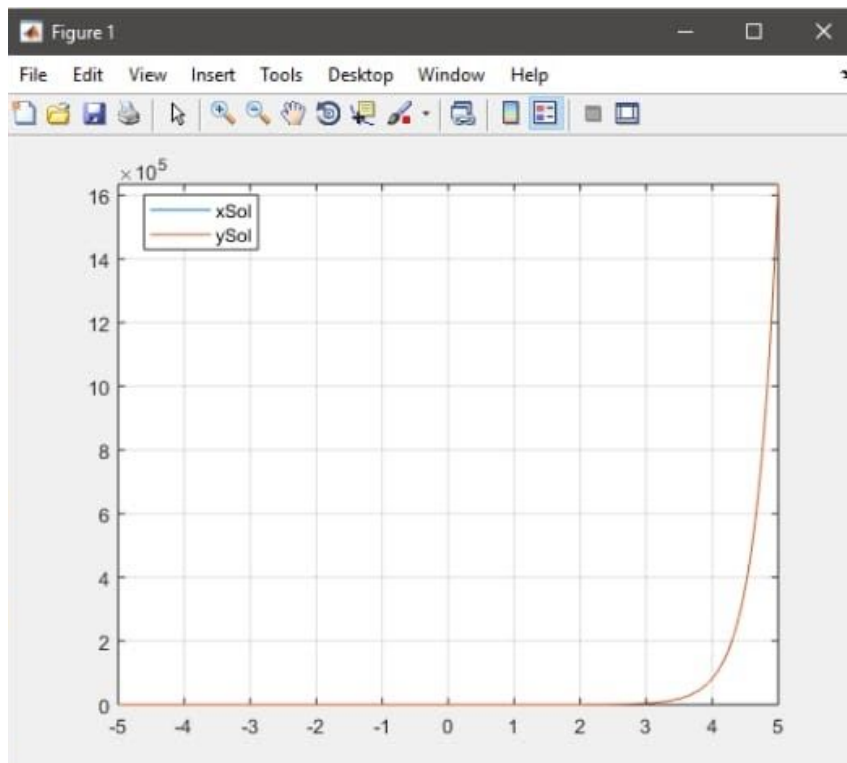
```
C5*exp(-t) + C6*exp(3*t)
```

```
xSol(t) =
```

```
exp(3*t)/2 - exp(-t)/2
```

```
ySol(t) =
```

```
exp(-t)/2 + exp(3*t)/2
```

**PLOT -**



# Practical 5

**AIM** - Write a program to linear fit the curve using Matlab.

**CODE** -

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% EXPERIMENT 5 FOR MS LAB

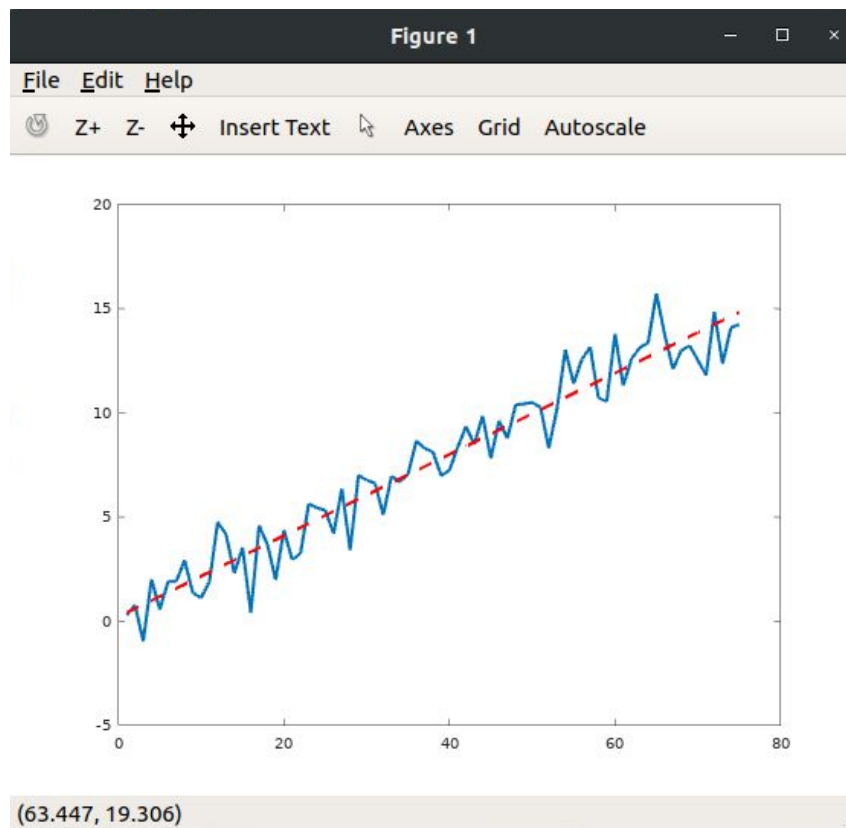
% Create and Plot Raw Data
x = 1:100;
y = 0.25*x + randn(1,100);
plot(x,y,'LineWidth',2)

% Fit line to data using polyfit
c = polyfit(x,y,1);

% Display evaluated equation y = m*x + b
disp(['Equation is y = ' num2str(c(1)) '*x + ' num2str(c(2))])

% Evaluate fit equation using polyval
y_est = polyval(c,x);

% Add trend line to plot
hold on
plot(x,y_est,'r--','LineWidth',2)
hold off
```

**PLOT -**

# Practical 6

**AIM** - Write a MATLAB Program for the following -

1. Write a program for the quadratic fit of the curve.
2. Write a program to the fin cubic fit of the curve.

**CODE** -

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% EXPERIMENT 6.1
%% FITTING THE QUADRATIC EQUATION
x = rand(1,100)*5
delta = rand(1,100)
y = 4*(x-3).^2 + 5+6*delta

figure
plot(x,y,'b*')
grid on

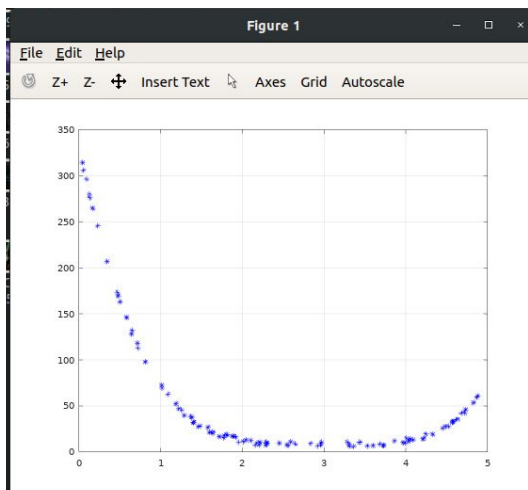
constant = lsqcurvefit(@f1,[0;0;0],x,y)

a = constant(1)
b = constant(2)
c = constant(3)

xfit = 0:0.1:10
yfit = f1(constant,xfit)

figure
plot(x,y,'b*')
hold on
plot(xfit,yfit,'r','linewidth',2)
grid on

-- INSERT -- 3,1 Top
```



## 2. CUBIC FIT

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% EXPERIMENT 6.2
%% FITTING THE CUBIC EQUATION

x = rand(1,100)*10
delta = rand(1,100)
y = 4*(x-3).^3 + 4+5+6*delta

figure
plot(x,y,'b*')
grid on

constant = lsqcurvefit(@f1,[0;0;0],x,y)

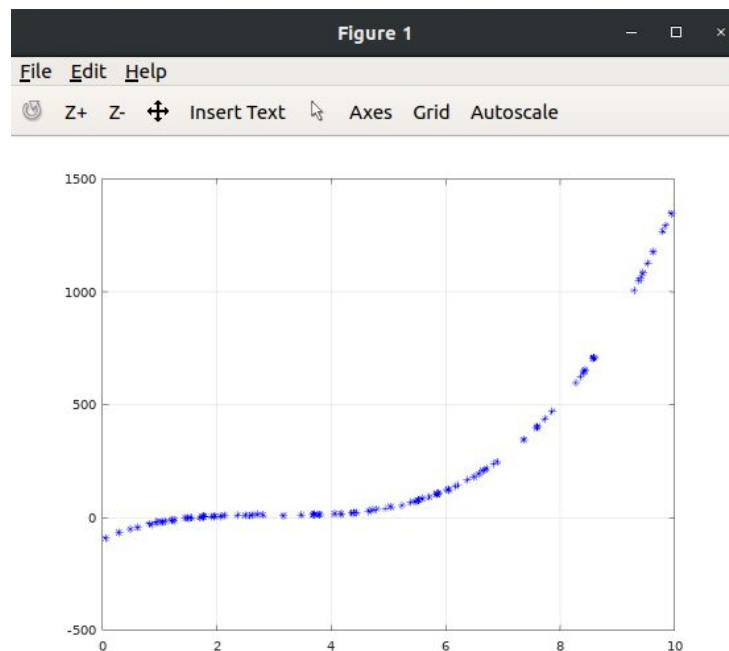
a = constant(1)
b = constant(2)
c = constant(3)

xfit = 0:0.1:10
yfit = f1(constant,xfit)

figure

plot(x,y,'b*')
hold on
plot(xfit,yfit,'r','linewidth',2)
grid on

1,17 All
```



# Practical 7

**AIM** - Write a MATLAB Program to simulate the SIR Model.

**CODE** -

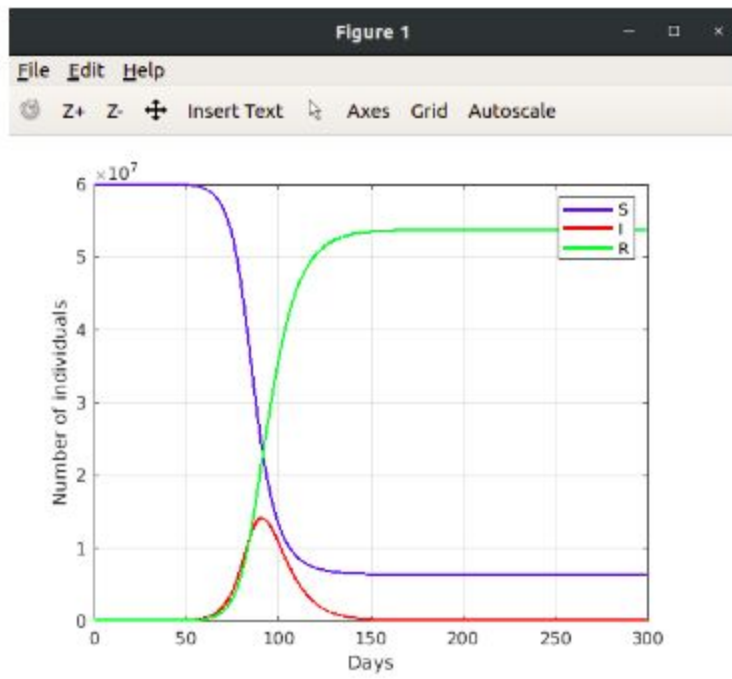
```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
% EXPERIMENT 7 - SIMULATION OF SIR MODEL

% Model parameters
beta = 5*10^-9; % rate of infection
gamma = 0.12; % rate of recovery (try also 0.07)
delta = 0.0; % rate of immunity loss
N = 6*10^7; % Total population N = S + I + R
I0 = 10; % initial number of infected
T = 300; % period of 300 days
dt = 1/4; % time interval of 6 hours (1/4 of a day)
fprintf('Value of parameter R0 is %.2f',N*beta/gamma)
```

```
error: called from
    /home/sachin/Desktop/LAB/MS LAB/7.m at line 12 column 1
>>

Value of parameter R0 is 2.50>> |
```

```
% Calculate the model
[S,I,R] = sir_model(beta,gamma,delta,N,I0,T,dt);
% Plots that display the epidemic outbreak
tt = 0:dt:T-dt;
% Curve
plot(tt,S,'b',tt,I,'r',tt,R,'g','LineWidth',2); grid on;
xlabel('Days'); ylabel('Number of individuals');
legend('S','I','R');
```



# Practical - 8

**AIM** - Write a program to find a cubic spline of data.

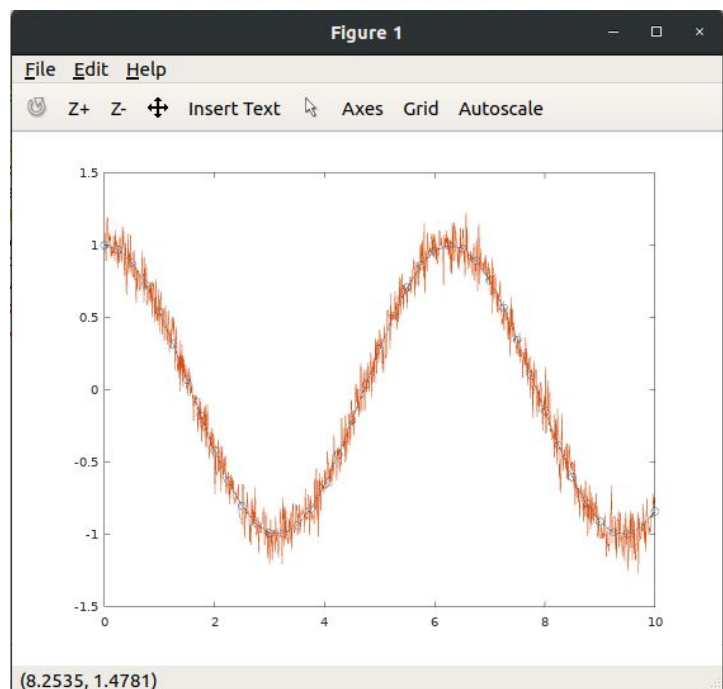
**CODE** -

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% EXPERIMENT 8 - find cubic spline matlab

x = 0:.25:10; y = cos(x);
xval=[0:.01:10];
yval=cos(xval)+0.1*randn(size(xval));
figure(1)
plot(x,y,'o-',xval,yval);
_B= spap2(6,4,xval,yval);
yfit=fnval(_B,xval);
hold on;
plot(xd,yfit,'r');

"8.m" [noeol] 11L, 236C 5,21 All
```

**OUTPUT** -



# Practical - 9

**AIM** - Multiple Linear Regression using MATLAB code

**CODE** -

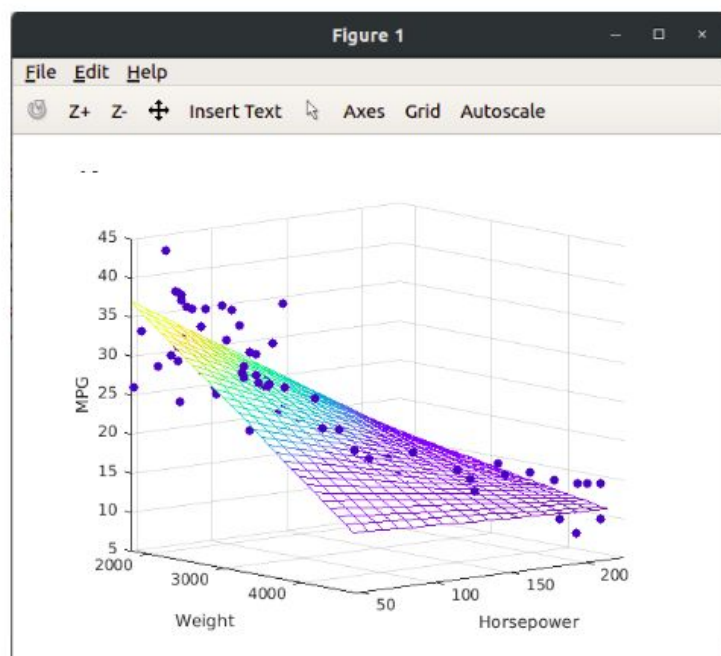
```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% working on the experiment 9 - multiple lineaeer regression

load carsmall
x1 = Weight;
x2 = Horsepower;    % Contains NaN data
y = MPG;

%Compute the regression coefficients for a linear model with an interaction term.
X = [ones(size(x1)) x1 x2 x1.*x2];
b = regress(y,X)    % Removes NaN data

%Plot the data and the model.
scatter3(x1,x2,y,'filled')
hold on
x1fit = min(x1):100:max(x1);
x2fit = min(x2):10:max(x2);
[X1FIT,X2FIT] = meshgrid(x1fit,x2fit);
YFIT = b(1) + b(2)*X1FIT + b(3)*X2FIT + b(4)*X1FIT.*X2FIT;
mesh(X1FIT,X2FIT,YFIT)
xlabel('Weight')
ylabel('Horsepower')
zlabel('MPG')
view(50,10)
hold off
```

**OUTPUT** -





# Practical - 10

**AIM** - Multiple Linear Regression using MATLAB code

**CODE** -

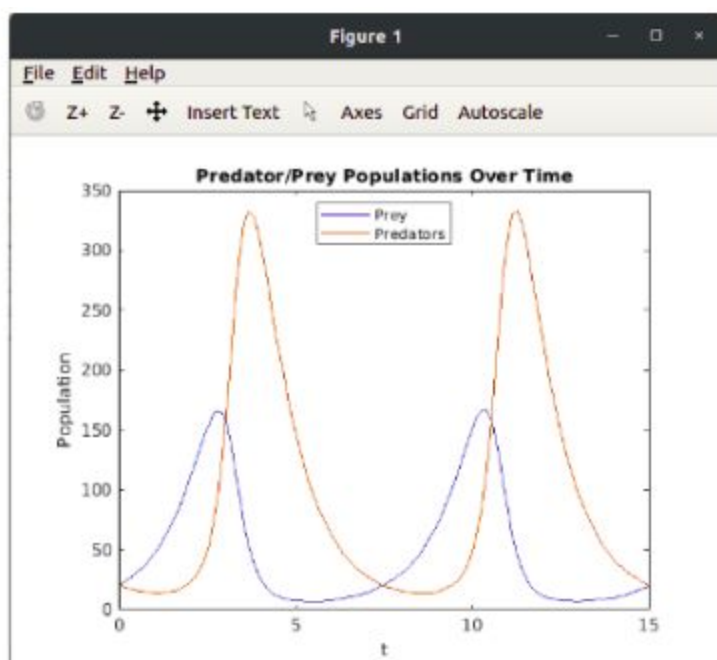
```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% dram some trajectory for the predator-prey problem!
type lotka
plot(t,y)
title('Predator/Prey Populations Over Time')
xlabel('t')
ylabel('Population')
legend('Prey','Predators','Location','North')

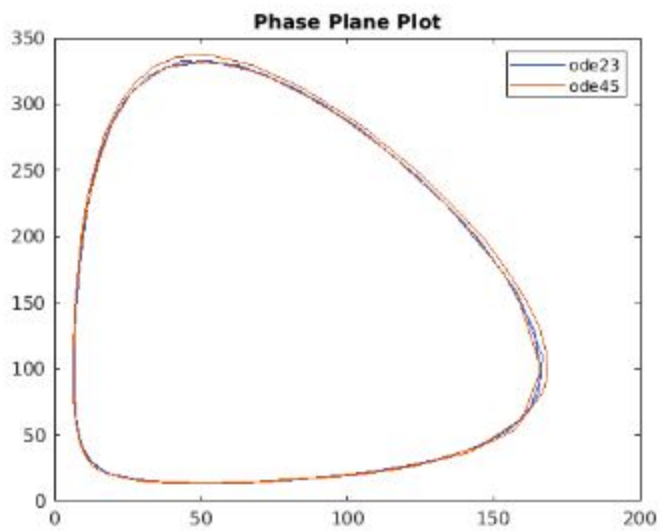
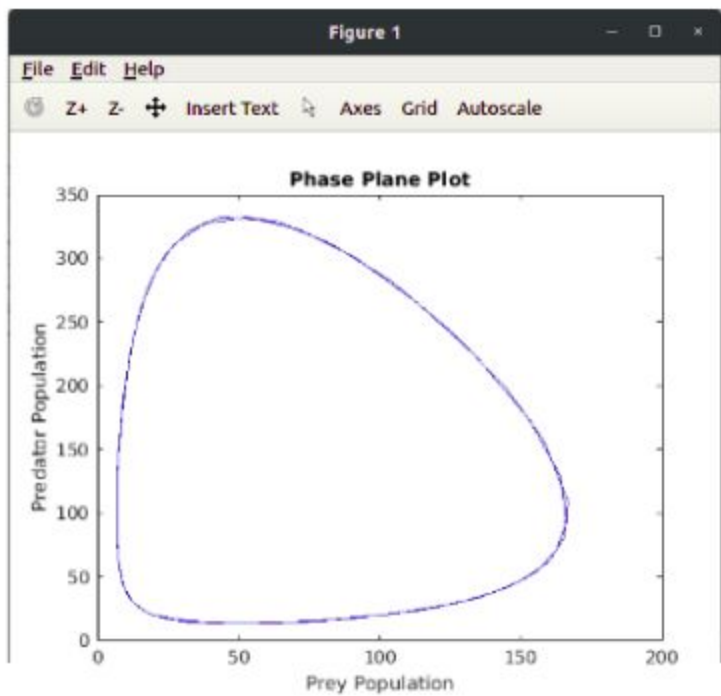
plot(y(:,1),y(:,2))
title('Phase Plane Plot')
xlabel('Prey Population')
ylabel('Predator Population')

[T,Y] = ode45(@lotka,[t0 tfinal],y0);

plot(y(:,1),y(:,2),'- ',Y(:,1),Y(:,2),'- ');
title('Phase Plane Plot')
legend('ode23','ode45')

~
-- INSERT --
```





## 2. Statistical data analysis using MATLAB

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
%% experiment 3
%% standard deviation!
A = [1 5; 3 7; -9 2];
w = [1 1 0.5];
S = std(A,w)

%% finding mean
A = [0 1 1; 2 3 2; 3 0 1; 1 2 3]

M = mean(A,2)
```

```
A = [1 5; 3 7; -9 2];
w = [1 1 0.5];
S = std(A,w)
```

```
S = 1x2
```

```
4.4900    1.8330
```

```
A = [0 1 1; 2 3 2; 3 0 1; 1 2 3]
```

```
A = 4×3
```

0	1	1
2	3	2
3	0	1
1	2	3

```
M = mean(A,2)
```

```
M = 4×1
```

0.6667
2.3333
1.3333
2.0000

# Practical - 11

**AIM** - Program to use the Monte Carlo method for approximate integration.

**CODE** -

```
sachin@2k17-mc-87: ~/Desktop/LAB/MS LAB
File Edit View Search Terminal Help
% EXPERIMENT 11

%% Program to use Monte Carlo method for approximate integration of f(x) over
%% [0,1] interval; taking f(x)=x ,x^2 ,cos(pi*x)

clc;
n=70;
x=rand(n,1);
gav=zeros(n,3);
gavvar=NaN(n,3);
gav(1,1)=x(1,1);
gav(1,2)=x(1,1)^2;
gav(1,3)=cos(pi*x(1,1));
for i=2:n
gav(i,1)=sum(x(1:i))/i; gav(i,2)=sum(x(1:i).^2)/i; gav(i,3)=sum(cos(pi*x(1:i)))/i;
gavvar(i,1)=var(x(1:i));
gavvar(i,2)=var(x(1:i).^2);
gavvar(i,3)=var(cos(pi*x(1:i)));
end
% Visualization
figure(1);
subplot(3,1,1);
plot(gav(:,1)); line((1:n),ones(n,1)/2,'color','red');
legend('Empirical Average','Theoretical Mean','Location','NorthEastOutside');
title('f(x)=x');
subplot(3,1,2);
plot(gav(:,2)); line((1:n),ones(n,1)/3,'color','red');
legend('Empirical Average','Theoretical Mean','Location','NorthEastOutside');
title('f(x)=x^2');
subplot(3,1,3);
plot(gav(:,3)); line((1:n),ones(n,1)*0,'color','red');
legend('Empirical Average','Theoretical Mean','Location','NorthEastOutside');
title('f(x)=cos(pi x)');
~
~
```

**OUTPUT** -

