LAB MANUAL LAB

DRONACHARYA
College of Engineering

LAB MANUAL

Computer Science & Engineering

SOFT COMPUTING LAB

FACULTY NAME:

MRS. KSHITIJA POL

ASSOCIATE PROFESSOR

CSE DEPARTMENT

INDEX

Sr. No.	Program	Remarks
1.	To perform Union,Intersection and Complement operations.	
2.	To implement De-Morgan's Law.	
3.	To plot various membership functions.	
4.	To implement FIS Editor. Use Fuzzy toolbox to model tip value that is given after a dinner based on quality ans service.	
5.	To implement FIS Editor.	
6.	Generate ANDNOT function using McCulloch-Pitts neural net.	
7.	Generate XOR function using McCulloch-Pitts neural net.	
8.	Hebb Net to classify two dimensional input patterns in bipolar with given targets.	
9.	Perceptron net for an AND function with bipolar inputs and targets.	
10.	To calculate the weights for given patterns using hetero-associative neural net.	
11.	To store vector in an auto-associative net.Find weight matrix & test the net with input	

12.	**To store the vector ,find the weight matrix with no self-connection.Test this using a discrete Hopfield net.**	

Program No. 1

Write a program in MATLAB to perform Union,Intersection and Complement operations.

```
%Enter Data
u=input('Enter First Matrix');
v=input('Enter Second Matrix');
%To Perform Operations
w=max(u,v);
p=min(u,v);
q1=1-u;
q2=1-v;
%Display Output
display('Union Of  Two Matrices');
display(w);
display('Intersection Of  Two Matrices');
display(p);
display('Complement Of  First Matrix');
display(q1);
display('Complement Of  Second Matrix');
display(q2);
```

Output:-

Enter First Matrix [0.3 0.4]

Enter Second Matrix [0.1 0.7]

Union Of Two Matrices

w =0.3000 0.7000

Intersection Of Two Matrices

p = 0.1000 0.4000

Complement Of First Matrix

q1 =0.7000 0.6000

Complement Of Second Matrix

q2 =0.9000 0.3000

Program No. 2

Write a program in MATLAB to implement De-Morgan's Law.

De-Morgan's Law c(i(u,v)) = max(c(u),c(v))

c(u(u,v)) = min(c(u),c(v))

```
%Enter Data
u=input('Enter First Matrix');
v=input('Enter Second Matrix');
%To Perform Operations
w=max(u,v);
p=min(u,v);
q1=1-u;
q2=1-v;
x1=1-w;
x2=min(q1,q2);
y1=1-p;
y2=max(q1,q2);
%Display Output
display('Union Of  Two Matrices');
display(w);
display('Intersection  Of  Two Matrices');
display(p);
```

```
display('Complement Of  First Matrix');
display(q1);
display('Complement Of  Second Matrix');
display(q2);
display('De-Morgans Law');
display('LHS');
display(x1);
display('RHS');
display(x2);
display('LHS1');
display(y1);
display('RHS1');
display(y2);
```

Output:-

Enter First Matrix [0.3 0.4]

Enter Second Matrix [0.2 0.5]

Union Of Two Matrices

w =0.3000 0.5000

Intersection Of Two Matrices

p =0.2000 0.4000

Complement Of First Matrix

q1 =0.7000 0.6000

Complement Of Second Matrix

q2 =0.8000 0.5000

De-Morgans Law

LHS

x1 = 0.7000 0.5000

RHS

x2 = 0.7000 0.5000

LHS1

y1 =0.8000 0.6000

RHS1

y2 = 0.8000 0.6000

Program No. 3

Write a program in MATLAB to plot various membership functions.

```
%Triangular Membership Function
x=(0.0:1.0:10.0)';
y1=trimf(x, [1 3 5]);
subplot(311)
plot(x,[y1]);


%Trapezoidal Membership Function
x=(0.0:1.0:10.0)';
y1=trapmf(x, [1 3 5 7]);
subplot(312)
plot(x,[y1]);


%Bell-Shaped Membership Function
x=(0.0:0.2:10.0)';
y1=gbellmf(x, [1 2 5]);
subplot(313)
plot(x,[y1]);
```

Output:-

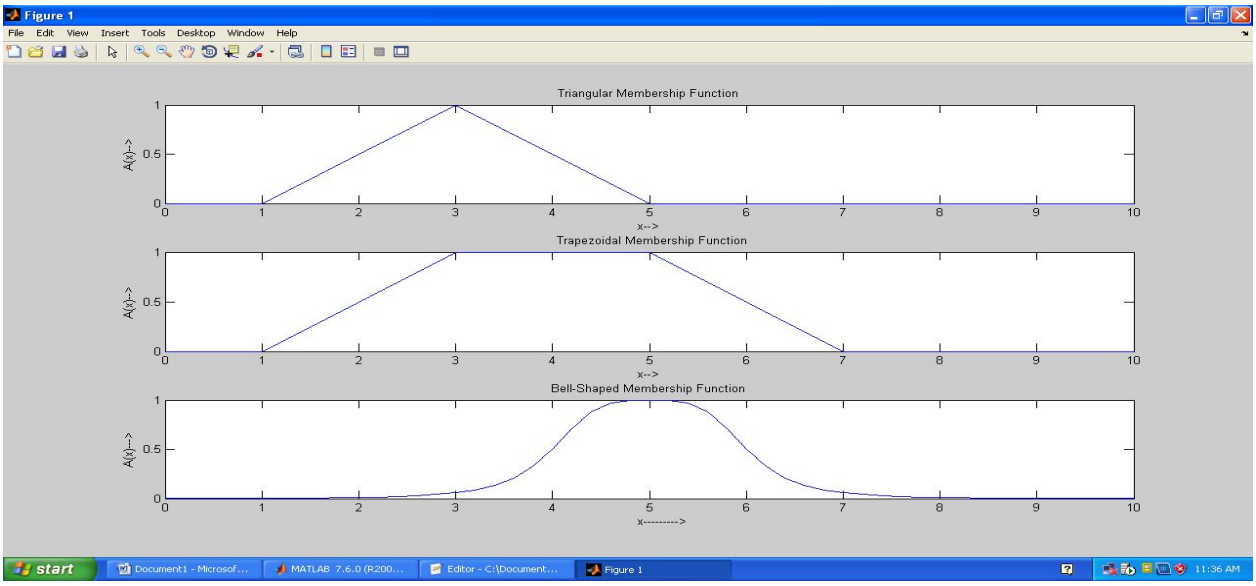


Figure 1
File Edit View Insert Tools Desktop Window Help
Triangular Membership Function
A(x)-->
x-->
Trapezoidal Membership Function
A(x)-->
x-->
Bell-Shaped Membership Function
A(x)-->
x--------->
start
Document1 - Microsof...
MATLAB 7.6.0 (R200...
Editor - C:\Document...
Figure 1
11:36 AM

Program No. 4

Use Fuzzy toolbox to model tip value that is given after a dinner which can be-not good,satisfying,good and delightful and service which is poor,average or good and the tip value will range from Rs. 10 to 100.

We are given the linguistic variables quality of food and sevice as input variables which can be written as:

Quality(not good,satisfying,good,delightful)

Service(poor,average,good)

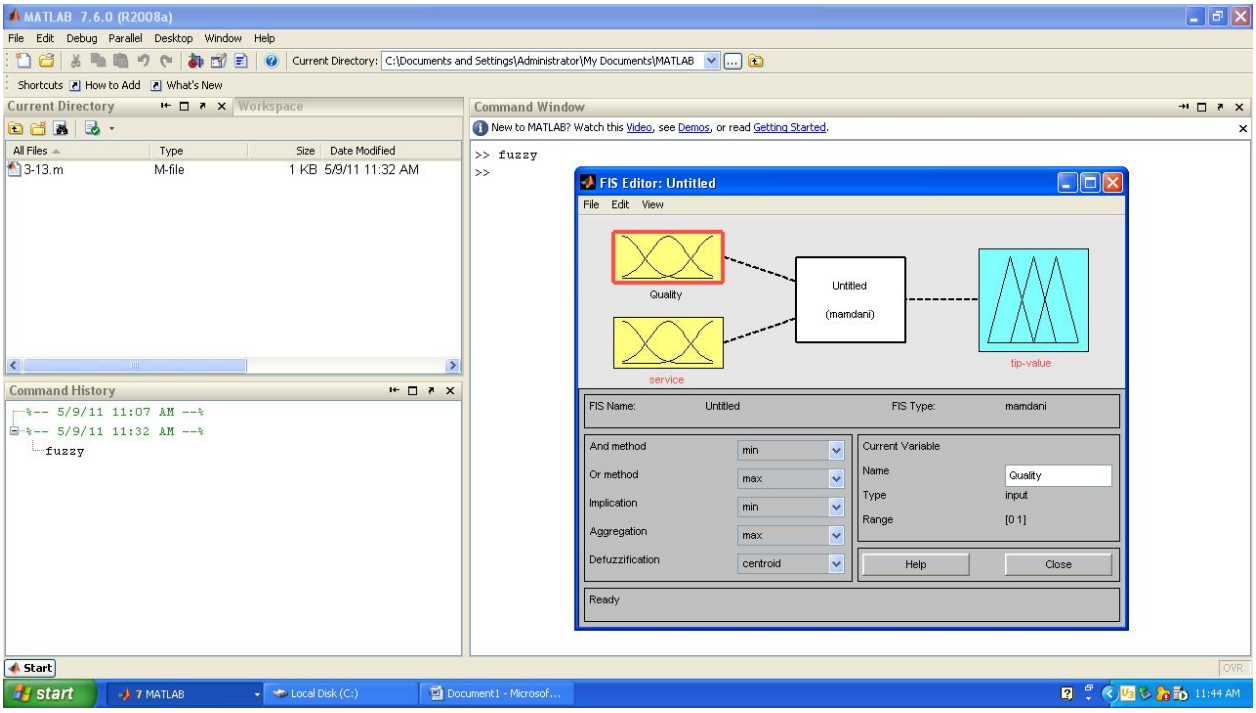
Similarly Output variable is Tip_value which may range from Rs. 10 to 100.

A Fuzzy system comprises the following modules:-

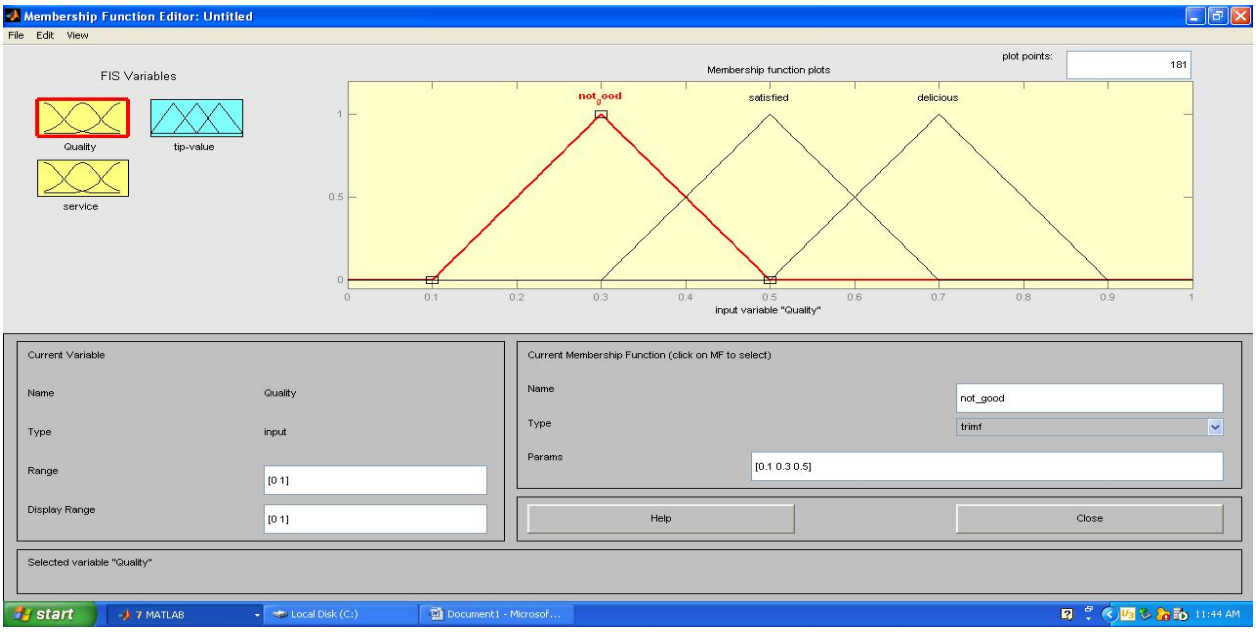
1. Fuzzification Interface
2. Fuzzy Inference Engine
3. Deffuzification Interface

Fuzzy sets are defined on each of the universe of discourse:-

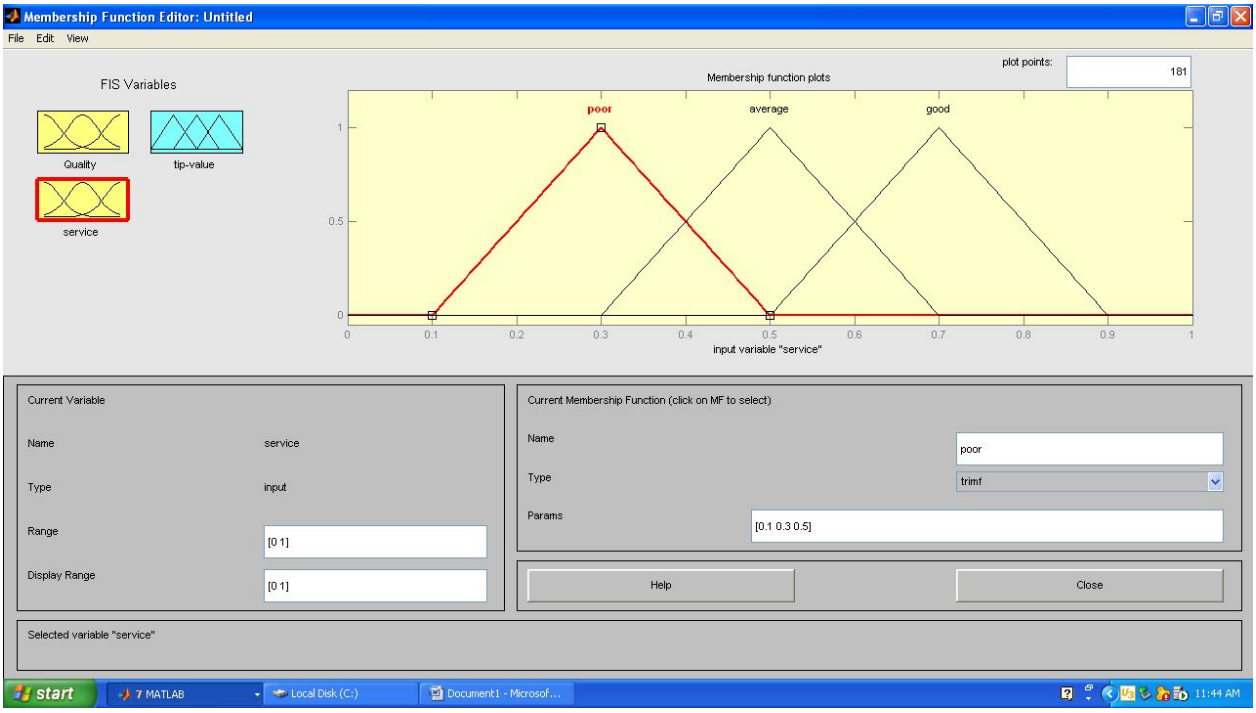
Quality,service and tip value.

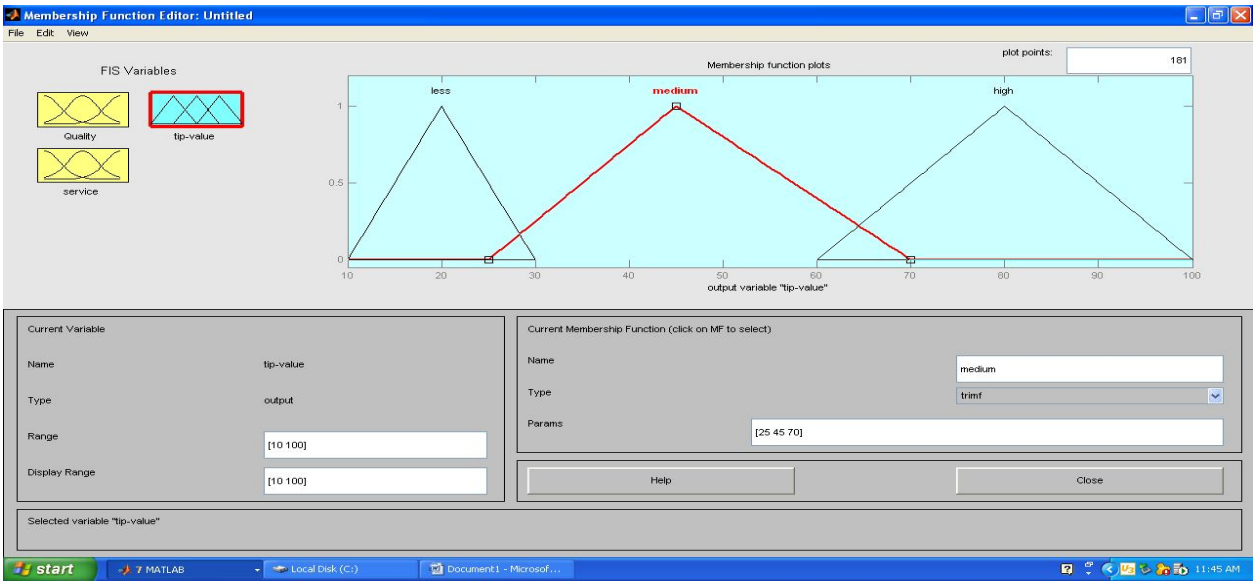


The values for Quality variable are selected for their respective ranges:-



Similarly values for Service variable are selected for their respective ranges :-





In general a compositional rule for inference involves the following procedure:=-

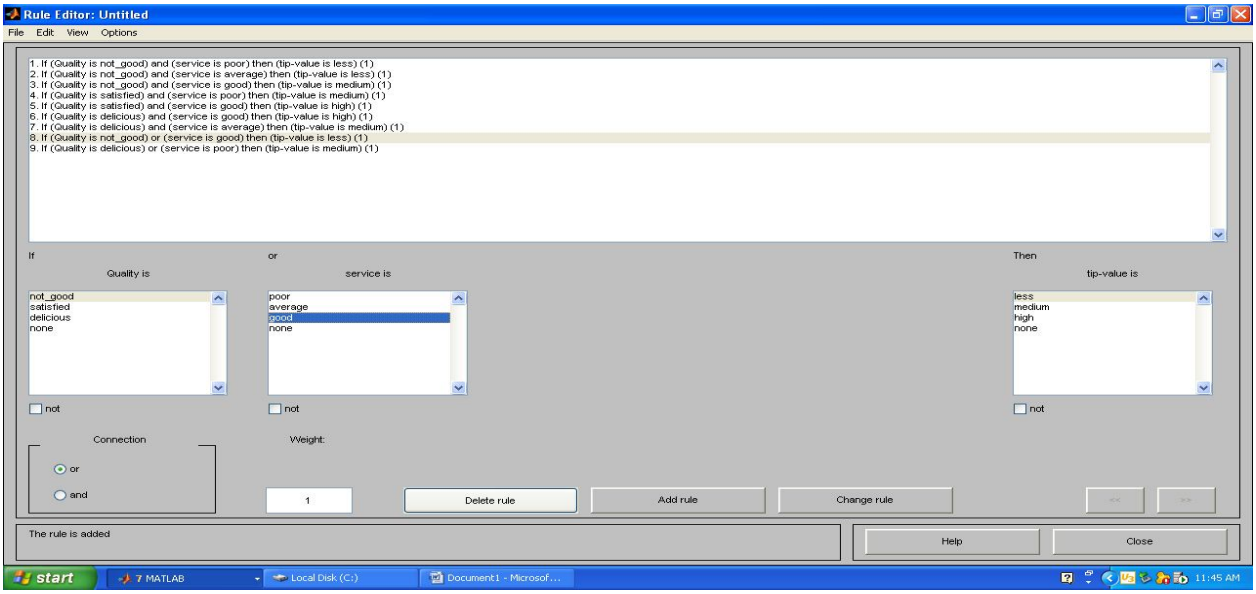
1. Compute memberships of current inputs in the relevant antecedent fuzzy set of rule.

2. If the antecedents are in conjunctive form,the AND operation is replaced by a minimum,if OR then by Maximum and similarly other operations are performed.

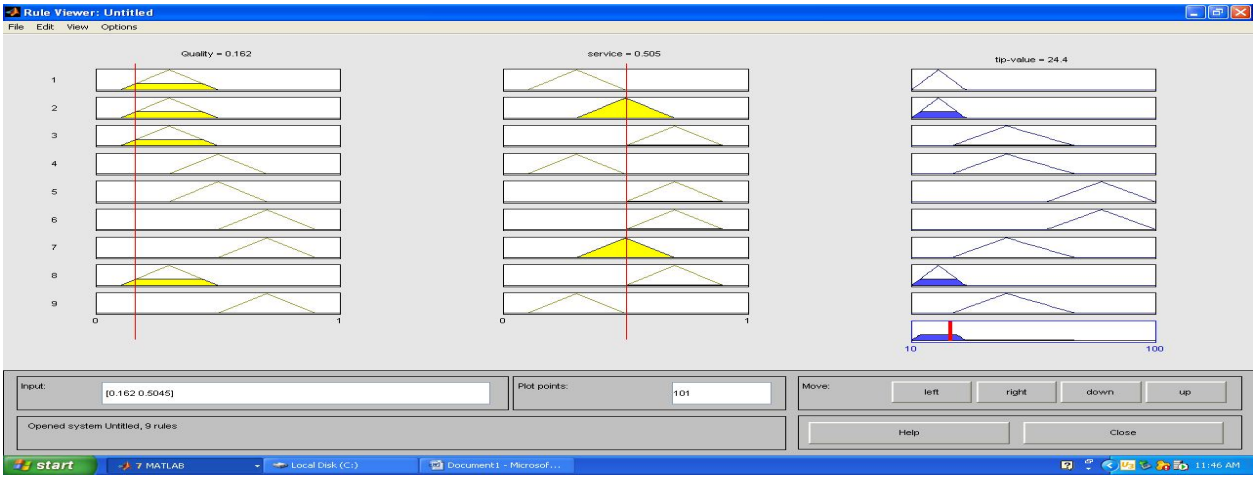
3. Scale or clip the consequent fuzzy set of the rule by a minimum value found in step 2 since this gives the smallest degree to which the rule must fire.

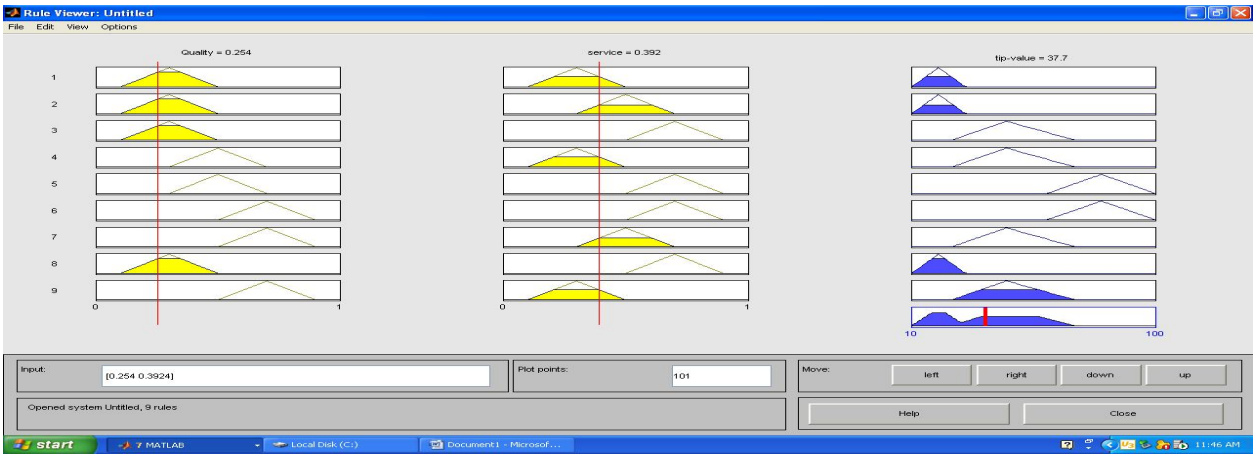
4. Repeat steps 1-3 for each rule in the rule base.

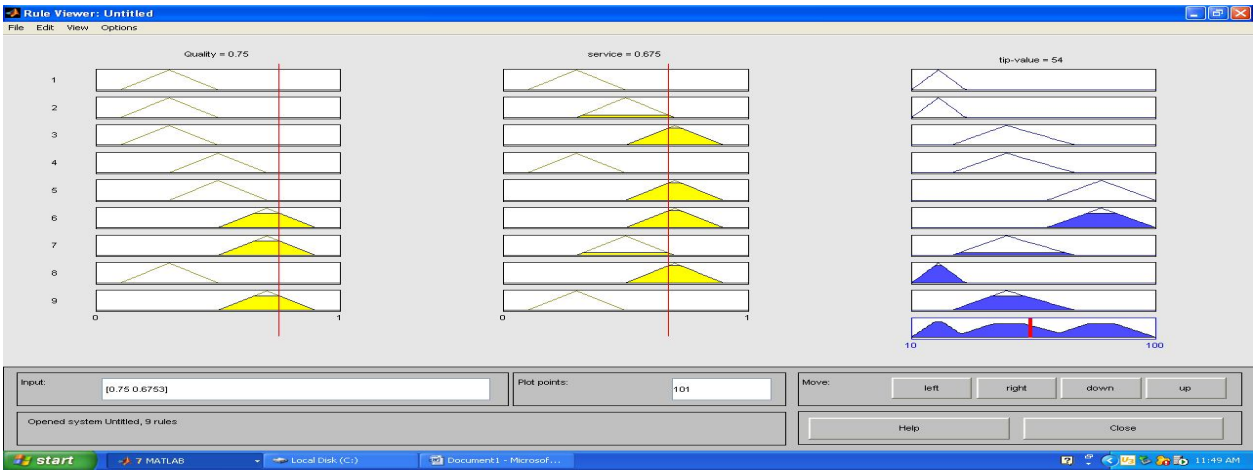
Superpose the scaled or clipped consequent fuzzy sets formed by such a superposition.There are numerous variants of the defuzzifications.

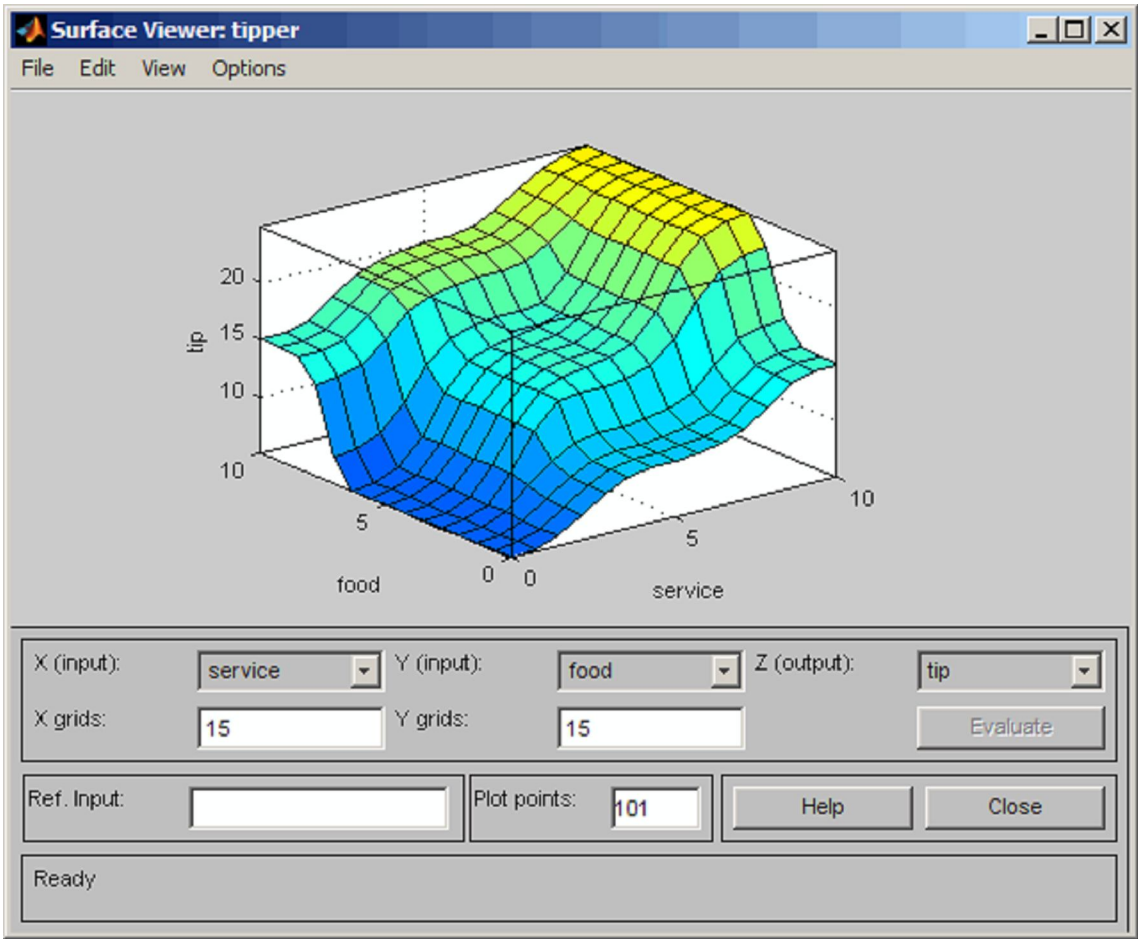


The output will be displayed as :-








Surface Viewer: tipper
File Edit View Options
tip
20
15
10
10
5
0
food
0
5
10
service
X (input):
service
Y (input):
food
Z (output):
tip
X grids:
15
Y grids:
15
Evaluate
Ref. Input:
Plot points:
101
Help
Close
Ready

Program No. 5

To implement FIS Editor.

FIS stands for Fuzzy Inference System.In FIS fuzzy rules are used for approximate reasoning.It is the logical framework that allows us to design reasoning systems based on fuzzy set theory.

To illustrate these concepts we use example of Water Tank:-

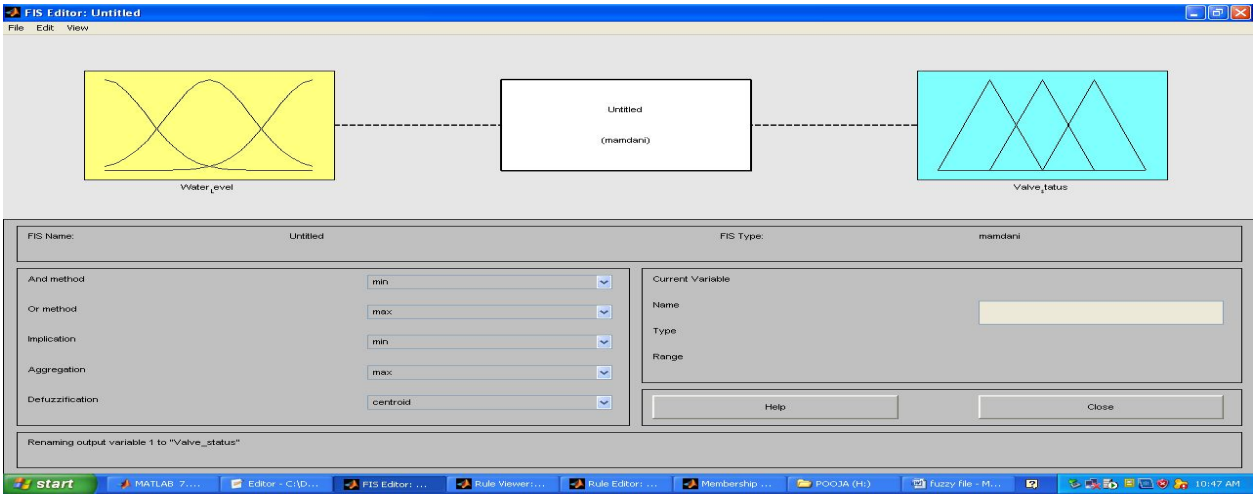
FIS editor consists of following units:-

i) Input

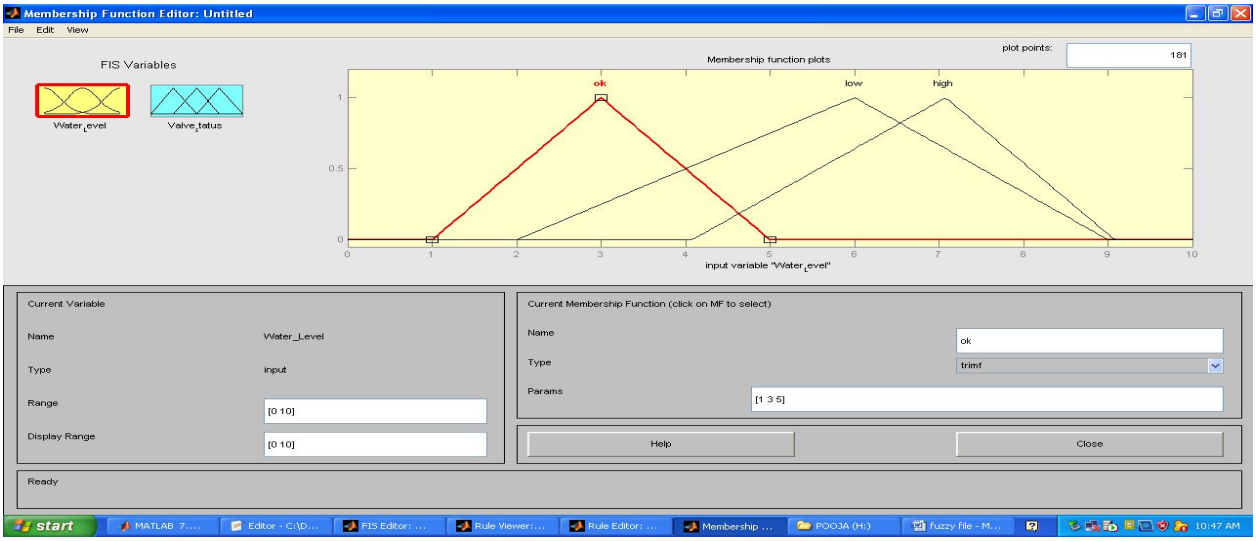
ii) Inference System

iii) Output

The Water Level is considered as the Input variable and Valve status is taken as Output Variable.



The Input-Output Variable's Membership functions should be plotted along with their ranges:-



The following screen appearance is obtained by clicking on the FIS Rule system indicator:-

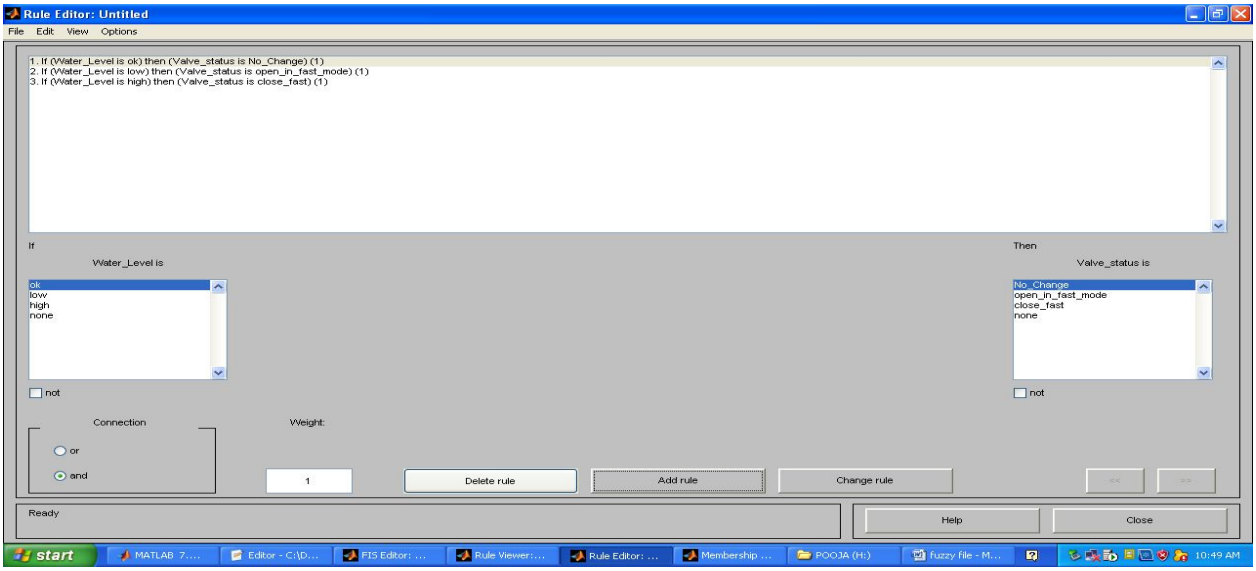
Rules are added by selecting variable's values and clicking on add rule menu each time a new rule is added.

The fuzzy Rules defined for water tank are:-

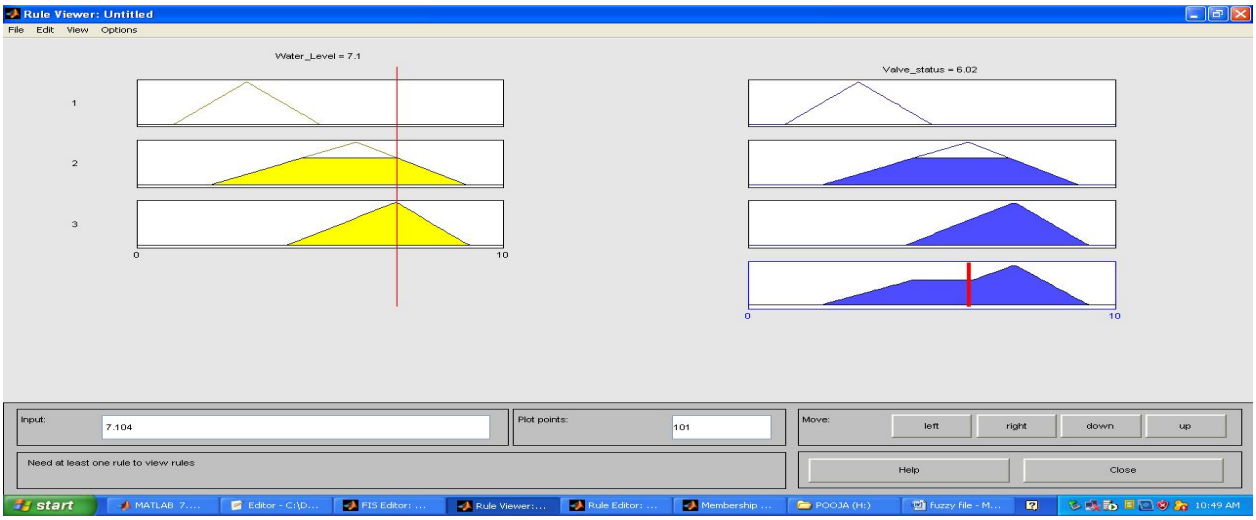
IF level is ok,THEN there is no change in valve.

IF level is low,THEN valve is open in fast mode.

IF level is high,THEN valve is closed in fast mode.



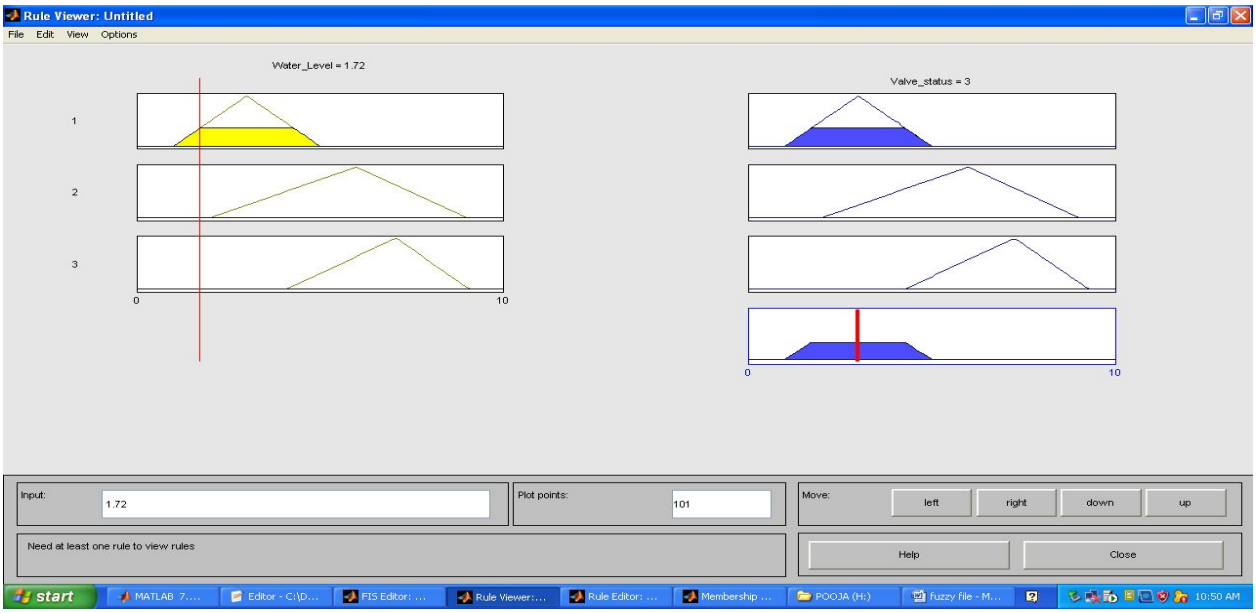
The result is displayed as plots of input-output membership functions :-

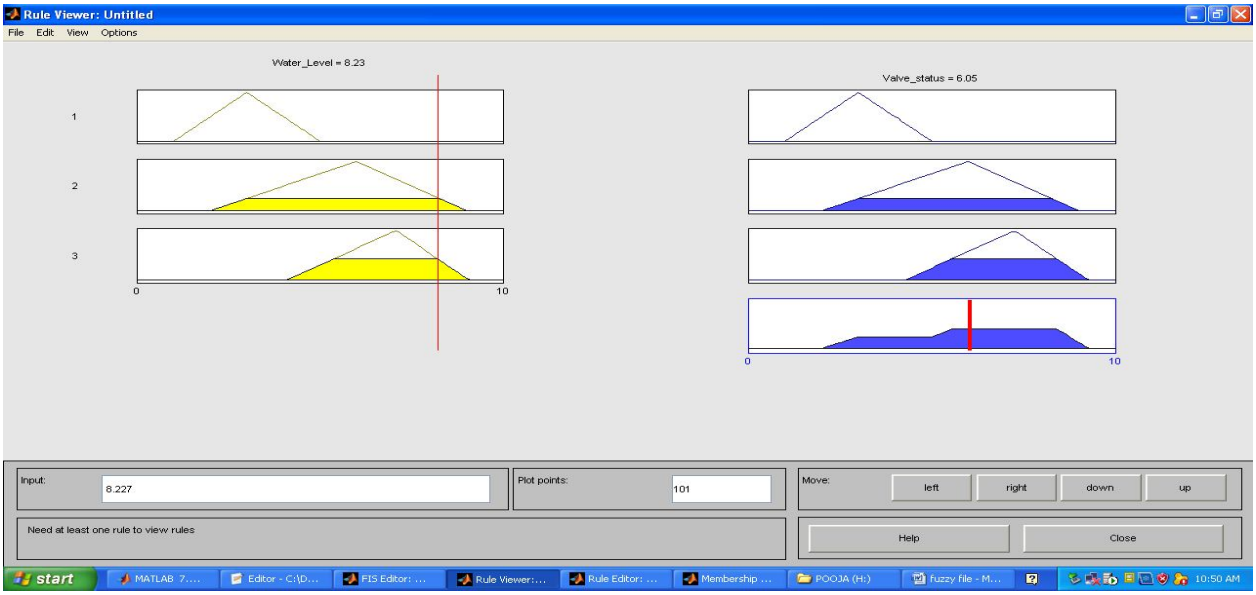


Water Level(ok,low,high)

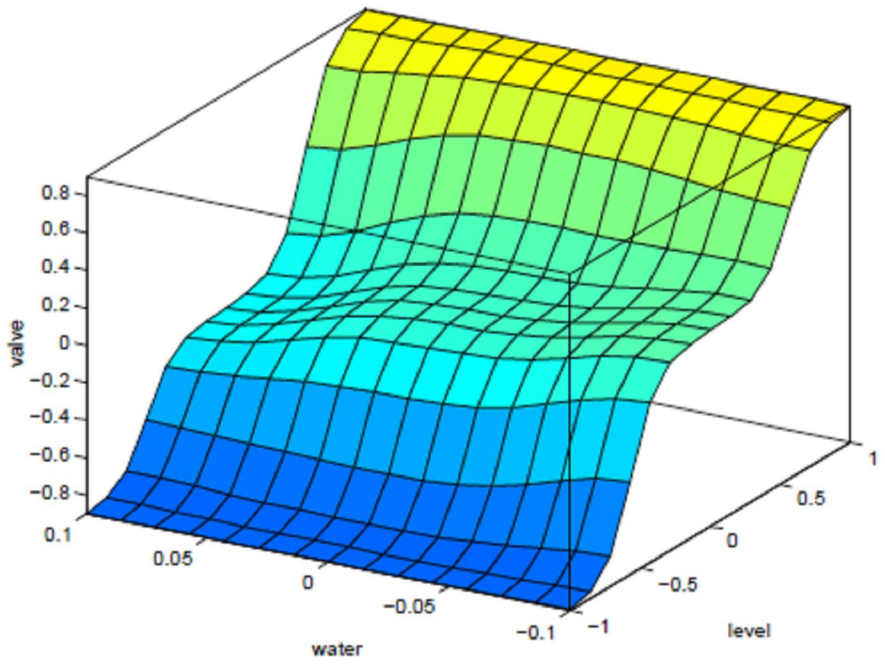
Valve Status(no change,open fast,closed fast)

The output in accordance with the input and rules provided by user is shown as(view-rule viewer):-


Rule Viewer: Untitled
File Edit View Options
Water_Level = 1.72
Valve_status = 3
1
2
3
0
10
0
10
Input:
1.72
Plot points:
101
Move:
left
right
down
up
Need at least one rule to view rules
Help
Close
start
MATLAB 7....
Editor - C:\D...
FIS Editor: ...
Rule Viewer:...
Rule Editor: ...
Membership ...
POOJA (H:)
fuzzy file - M...
10:50 AM


Rule Viewer: Untitled
File Edit View Options
Water_Level = 8.23
Valve_status = 6.05
1
2
3
0
10
0
10
Input:
8.227
Plot points:
101
Move:
left
right
down
up
Need at least one rule to view rules
Help
Close
start
MATLAB 7....
Editor - C:\D...
FIS Editor: ...
Rule Viewer:...
Rule Editor: ...
Membership ...
POOJA (H:)
fuzzy file - M...
10:50 AM

Output


valve
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
0.1
0.05
0
−0.05
−0.1
water
−1
−0.5
0
0.5
1
level

Program No. 6

Generate ANDNOT function using McCulloch-Pitts neural net by MATLAB program.

```
%ANDNOT function using McCulloch-Pitts neuron
clear;
clc;
% Getting weights and threshold value
disp('Enter the weights');
w1=input('Weight w1=');
w2=input('Weight w2=');
disp('Enter threshold value');
theta=input('theta=');
y=[0 0 0 0];
x1=[0 0 1 1];
x2=[0 1 0 1];
z=[0 0 1 0];
con=1;
while con
   zin = x1*w1+x2*w2;
   for i=1:4
      if zin(i)>=theta
```

```
        y(i)=1;
    else y(i)=0;
    end
  end
  disp('Output of net=');
  disp(y);
  if y==z
    con=0;
  else
    disp('Net is not learning Enter another set of weights and threshold value');
    w1=input('Weight w1=');
w2=input('Weight w2=');
thete=input('theta=');
  end
end
disp('McCulloch Pitts Net for ANDNOT function');
disp('Weights of neuron');
disp(w1);
disp(w2);
disp('Threshold value=');
disp(theta);
```

Output :-

Enter the weights

Weight w1=1

Weight w2=1

Enter threshold value

theta=1

Output of net= 0 1 1 1

Net is not learning Enter another set of weights and threshold value

Weight w1=1

Weight w2=-1

theta=1

Output of net=0 0 1 0

McCulloch Pitts Net for ANDNOT function

Weights of neuron

1

-1

Threshold value=

1

Program No. 7

Generate XOR function using McCulloch-Pitts neural net by MATLAB program.

```
% XOR function using McCulloch-Pitts neuron
clear;
clc;
% Getting weights and threshold value
disp('Enter the weights');
w11=input('Weight w11=');
w12=input('Weight w12=');
w21=input('Weight w21=');
w22=input('Weight w22=');
v1=input('Weight v1=');
v2=input('Weight v2=');
disp('Enter threshold value');
theta=input('theta=');
 x1=[0 0 1 1];
x2=[0 1 0 1];
z=[0 1 1 0];
con=1;
while con
    zin1 = x1*w11+x2*w21;
```

```
 zin2 = x1*w21+x2*w22;
    for i=1:4
        if zin1(i)>=theta
            y1(i)=1;
        else y1(i)=0;
        end
if zin2(i)>=theta
            y2(i)=1;
        else y2(i)=0;
    end
end
yin=y1*v1+y2*v2;
for i=1:4
if yin(i)>=theta;
y(i)=1;
else
y(i)=0;
end
end
    disp('Output of net=');
    disp(y);
    if y==z
        con=0;
```

```
    else
        disp('Net is not learning Enter another set of weights and threshold value');
        w11=input('Weight w11=');
        w12=input('Weight w12=');
        w21=input('Weight w21=');
        w22=input('Weight w22=');
        v1=input('Weight v1=');
        v2=input('Weight v2=');
theta=input('theta=');
end
end
disp('McCulloch Pitts Net for XOR function');
disp('Weights of neuron Z1');
disp(w11);
disp(w21);
disp('Weights of neuron Z2');
disp(w12);
disp(w22);
disp('Weights of neuron Y');
disp(v1);
disp(v2);
disp('Threshold value=');
disp(theta);
```

Output :-

Enter the weights

Weight w11=1

Weight w12=-1

Weight w21=-1

Weight w22=1

Weight v1=1

Weight v2=1

Enter threshold value

theta=1

Output of net= 0 1 1 0

McCulloch Pitts Net for XOR function

Weights of neuron z1

1

-1

Weights of neuron z2

-1

1

Weights of neuron y

1

1

Threshold value= 1

Program No. 8

Write a MATLAB program for Hebb Net to classify two dimensional input patterns in bipolar with their targets given below:

'*' indicates a '+' and '.' Indicates '-'

*****	*****
*....	*....
*****	*****
*....	*....
*****	*

```
% Hebb Net to classify Two -Dimensional input patterns.
clear;
clc;
%Input Pattern
E=[1 1 1 1 1 -1 -1 -1 1 1 1 1 1 -1 -1 -1 1 1 1 1];
F=[1 1 1 1 1 -1 -1 -1 1 1 1 1 1 -1 -1 -1 1 -1 -1 -1];
X(1,1:20)=E;
X(2,1:20)=F;
w(1:20)=0;
t=[1 -1];
b=0;
for i=1:2
```

```
w=w+X(i,1:20)*t(i);
b=b+t(i);
end
disp('Weight Matrix');
disp(w);
disp('Bias');
disp(b);
```

Output :-

Weight Matrix

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2

Bias

0

Program No. 9

Write a MATLAB program for Perceptron net for an AND function with bipolar inputs and targets.

```
% Perceptron for AND Function
clear;
clc;
x=[1 1 -1 -1;1 -1 1 -1];
t=[1 -1 -1 -1];
w=[0 0];
b=0;
alpha=input('Enter Learning rate=');
theta=input('Enter Threshold Value=');
con=1;
epoch=0;
while con
con=0;
for i=1:4
yin=b+x(1,i)*w(1)+x(2,i)*w(2);
if yin>theta
y=1;
end
if yin<=theta & yin>=-theta
```

```
y=0;
end
if yin<-theta
y=-1;
end
if y-t(i)
con=1;
for j=1:2
w(j)=w(j)+alpha*t(i)*x(j,i);
end
b=b+alpha*t(i);
end
end
epoch=epoch+1;
end
disp('Perceptron for AND Function');
disp('Final Weight Matrix');
disp(w);
disp('Final Bias');
disp(b);
```

Output :-

Enter Learning rate=1

Enter Threshold Value=0.5

Perceptron for AND Function

Final Weight Matrix

1 1

Final Bias

-1

Program No. 10

Write a M-file to calculate the weights for the following patterns using hetero-associative neural net for mapping four input vectors to two output vectors

S1	S2	S3	S4	t1	t2
1	1	0	0	1	0
1	0	1	0	1	0
1	1	1	0	0	1
0	1	1	0	0	1

```
% Hetero-associative neural net for mapping input vectors to output vectors.
clear;
clc;
x=[1 1 0 0;1 0 1 0;1 1 1 0;0 1 1 0];
t=[1 0;1 0;0 1;0 1];
w=zeros(4,2);
for i=1:4

w=w+x(i,1:4)'*t(i,1:2);
end
disp('Weight Matrix');
disp(w);
```

Output:-

Weight Matrix

2	1
1	2
1	2
0	0

Program No. 11

Write an M-file to store vector[-1 -1 -1 -1] and [-1 -1 1 1] in an auto-associative net.Find weight matrix.Test the net with [1 1 1 1] as input.

```
% Auto-association problem
clc;
clear;
x=[-1 -1 -1 -1;-1 -1 1 1];
t=[1 1 1  1];
w=zeros(4,4);
for i=1:2
w=w+x(i,1:4)'*x(i,1:4);
end
yin=t*w;
for i=1:4
if yin(i)>0
 y(i)=1;
else
y(i)=-1;
end
end
disp('The calculated Weight Matrix');
disp(w);
```

```
if x(1,1:4)==y(1:4)| x(2,1:4)==y(1:4)
disp('The Vector is a Known vector');
else
disp('The Vector is a UnKnown vector');
end
```

Output :-

The calculated Weight Matrix

2	2	0	0
2	2	0	0
0	0	2	2
0	0	2	2

The Vector is a UnKnown vector

Program No. 12

Write a MATLAB program to store the vector (1 1 1 -1).Find the weight matrix with no self-connection.Test this using a discrete Hopfield net with mistakes in first and second component of stored vector i.e (0 0 1 0).Also the given pattern in binary form is[1 1 1 0].

```
% Discrete Hopfield Net
clc;
clear;
x=[1 1 1 0];
tx=[0 0 1  0];
w=(2*x'-1)*(2*x-1);
for i=1:4
w(i,i)=0;
end
con=1;
y=[0 0 1 0];
while con
up=[4 2 1 3]
for i=1:4
yin(up(i))=tx(up(i))+y*w(1:4,up(i));
if yin(up(i))>0
```

```
y(up(i))=1;
end
end
if y==x
disp('Convergence has been obtained');
disp('The Converged Output');
disp(y);
con=0;
end
end
```

Output:-

```
up =     4     2     1     3


Convergence has been obtained

The Converged Output

    1     1     1
```