**ELEC4543 Assignment 4**

***Fuzzy System Implementation of function approximation***

**Name: Wang Yuli(Rainy)**

**U.no: 3035028946**

**Program: BEng(ElecE), Year5**

**Abstract**

This assignment aims at implementing a simple fuzzy system using MATLAB Fuzzy Logic Toolbox and observing the results. The subject of the experiment is to do an approximation of function through 9 sample data points. To achieve the goal, we choose to implement a fuzzy system through fuzzification, inference engine processing and defuzzification. Besides, the experiment is conducted by implementing the fuzzy system by hand and comparing with results produced by actual function. From observation, it can be shown that the fuzzy system is relatively accurate for the 9 given data points. The error for predicting unknown data points are also generally lower than 0.07, which can generalize the behavior of function but still not accurate enough. Certain methods like increasing size of known data samples may improve our accuracy score.

**Objective & Implementation**

* **Objective**

The objective of this assignment is to approximate function as accurate as possible, through implementation of a fuzzy system via MATLAB Fuzzy Logic Toolbox. As illustrated in Section B, Chapter 11 of this course, we use the following method to construct the system:

1. Fuzzification: triangular fuzzifier by triangular membership function
2. Inference Engine: product inference engine
3. Defuzzification: center-of-area (CoA) defuzzifier \*

\*: the assignment instruction advises us to use center-average fuzzifier. However, since Fuzzy Logic Toolbox provides centroid (i.e. CoA) defuzzifier and center-average is to approximate CoA, we use CoA directly in this assignment.

* **Implementation**

Instead of using pure GUI tool as a shortcut, in this assignment, we utilize the functions provided by MATLAB Fuzzy Logic Toolbox to construct our model step-by-step.

The way of implementation is following these steps:

1. Initialize the model by specify methods for Inference Engine & Defuzzification

To initialize the fuzzy system using our selected inference Engine & Defuzzification, we need to set the parameters as follows for MATLAB function newfis:

* System type: Mamdani
* AND Method: t-norm -> algebraic product
* OR Method: s-norm -> max
* Implication Method: Mamdani product implication
* Aggregation (i.e. composition of individual rules): Max, i.e. Union
* Defuzzifier: centroid

1. Membership function setup with given data points

To set up the membership function, we add member functions to our fuzzy system for “input” and “output” respectively. To summarize our given 9 data points, i.e. (0, 0), (0.125,0.7071), (0.25,1), (0.375,0.7071), (0.5, 0), (0.625, -0.7071), (0.75, -1), (0.875, -0.707), (1,0). We create 9 triangles and 5 triangles on input & output spectrum respectively. The details are shown in the code

1. Rules & save the system

Add inference rules to the fuzzy system, following syntax of provided functions. The save the system for future use. The details are shown in the code.

1. Test & Evaluation

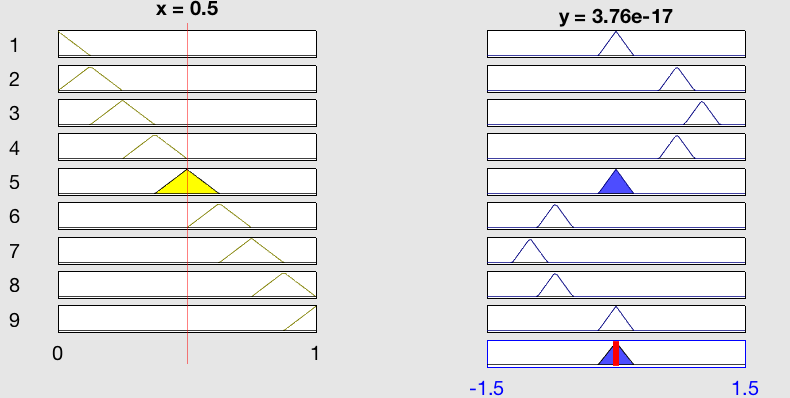
Plot the model architecture and use 101 sample inputs to see performance of the fuzzy system. Analyze overall & maximum difference between predicted & desired output.

**Results & Evaluation**

* Member functions:



* Rules & system structure



* System output vs original function & Error plot



Hence we can know that the system can generalize the behavior of function . From the above plot, we know that the error is generally lower than approximately 0.07.

Therefore, by extracting our information of interest, we found:

* Maximum difference between fuzzy system & actual function: 0.0701
* When using 101 inputs, the value of x at the point of maximum difference (i.e. 0.0701):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Value of x** | **0.19** | **0.31** | **0.69** | **0.81** |
| **Actual function result** | 0.9298 | 0.9298 | -0.9298 | -0.9298 |
| **Fuzzy system result** | 0.8597 | 0.8597 | -0.8597 | -0.8597 |
| **Difference (absolute)** | 0.0701 | 0.0701 | 0.0701 | 0.0701 |

The maximum difference happened close to the local minimum/maximum of the function, which shows that it is more difficult for a fuzzy system to approximate a function at the points of decreasing slopes (derivatives). Therefore, we can suggest:

With more data points used closer to such points, for constructing the fuzzy system, the accuracy is likely to be improved.

**Code**

% This fuzzy system is constructed to approximate function sin(2pix) using

% 9 data points, with method of:

% 1. triangular fuzzifier

% 2. product inference engine

% 3. center of area defuzzifier

%

% Experiments are conducted to observe the accuracy of the system

%

% data points we use to estimate g(x) = sin(2pix)

% (0, 0), (0.125,0.7071), (0.25,1),

% (0.375,0.7071), (0.5, 0), (0.625, -0.7071),

% (0.75, -1), (0.875,-0.707), (1,0)

clear all;

clear all;

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% \*\*\*\*\*\*\*\* Step1: system model \*\*\*\*\*\*

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% create fuzzy system model

% a = newfis ( fisName, fisType, and, or, imp, agg, defuzz)

fuzzySys = newfis('sin2pix','mamdani', 'prod', 'max', 'prod', 'max', 'centroid');

% note: the aggMethod is max, representing the 'union' of individual rules

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% \*\*\* Step2: membership functions \*\*\*

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% add input membership functions

fuzzySys = addvar(fuzzySys, 'input', 'x', [0 1]);

num = 1;

for ak = 0:0.125:1

fuzzySys = addmf(fuzzySys, 'input', 1, strcat('A', num2str(num)), 'trimf', [ak-0.125 ak, ak+0.125]);

num = num + 1;

end

% add output membership functions

fuzzySys = addvar (fuzzySys, 'output', 'y', [-1.5 1.5]);

fuzzySys = addmf(fuzzySys, 'output', 1, 'B1', 'trimf', [-1.2071 -1 -0.7929]);

fuzzySys = addmf(fuzzySys, 'output', 1, 'B2', 'trimf', [-0.9142 -0.7071 -0.5]);

fuzzySys = addmf(fuzzySys, 'output', 1, 'B3', 'trimf', [-0.2071 0 0.2071]);

fuzzySys = addmf(fuzzySys, 'output', 1, 'B4', 'trimf', [0.5 0.7071 0.9142]);

fuzzySys = addmf(fuzzySys, 'output', 1, 'B5', 'trimf', [0.7929 1 1.2071]);

plotmf(fuzzySys,'input',1);

title('input member functions');

plotmf(fuzzySys,'output',1);

title('output member functions');

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% \*\*\*\*\*\*\*\*\* Step3: Rules \*\*\*\*\*\*\*\*\*\*\*\*

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

w = 1; % weights: each rule are weighted 1

c = 1; % rules are composited by AND method

r1 = [1 3 w c];

r2 = [2 4 w c];

r3 = [3 5 w c];

r4 = [4 4 w c];

r5 = [5 3 w c];

r6 = [6 2 w c];

r7 = [7 1 w c];

r8 = [8 2 w c];

r9 = [9 3 w c];

ruleList = [r1;r2;r3;r4;r5;r6;r7;r8;r9];

fuzzySys = addrule(fuzzySys, ruleList);

showrule(fuzzySys);

ruleview(fuzzySys);

writefis(fuzzySys, 'fuzzy\_sin2pi'); % save the system

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% \*\*\*\* Step4: test & evaluation \*\*\*\*\*

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% view the whole system

plotfis(fuzzySys);

% view the plotted surface

surfview(fuzzySys);

% compare output with actual function to obtain the error score plot

% 1. error within original 9 data points

input = 0:0.125:1;

out = evalfis(input, fuzzySys);

expected = sin(2\*pi\*input);

error = expected' - out;

plot(input, abs(error)), grid on;

title('error between fuzzy system and function sin(2pix)');

xlabel('x');

ylabel('error (abs)');

% 2. error in 101 data points in range [0,1]

% gensurf(fuzzySys);

input = 0:0.01:1;

out = evalfis(input, fuzzySys);

expected = sin(2\*pi\*input);

% plotting estimated & expected output for comparison

plot(input, out, input, expected);

title('estimated & expected output comparison');

xlabel('x (101 data points)');

ylabel('y estimated/expected')

% plotting difference between estimated & expected output

error = expected' - out;

plot(input, abs(error)), grid on;

title('error between fuzzy system and function sin(2pix)');

xlabel('x (101 data points)');

ylabel('error (abs)');

% find data points with max error value

max\_error = max(error);

max\_error\_occur = abs(abs(error) - max\_error) < 0.0001; % whether the error is the maximum error

indices = find(max\_error\_occur == 1); % the indices of inputs that result in maximum error

max\_error\_inputs = input(indices); % the input(x) values that result in maximum error

sin(2\*pi\*max\_error\_inputs) % Actual function result

evalfis(max\_error\_inputs, fuzzySys) % Fuzzy system result