MEE406 Intelligent Control Systems
Fuzzy Control Demo Lecture

Mertcan Koçak

17.04.2019

Content

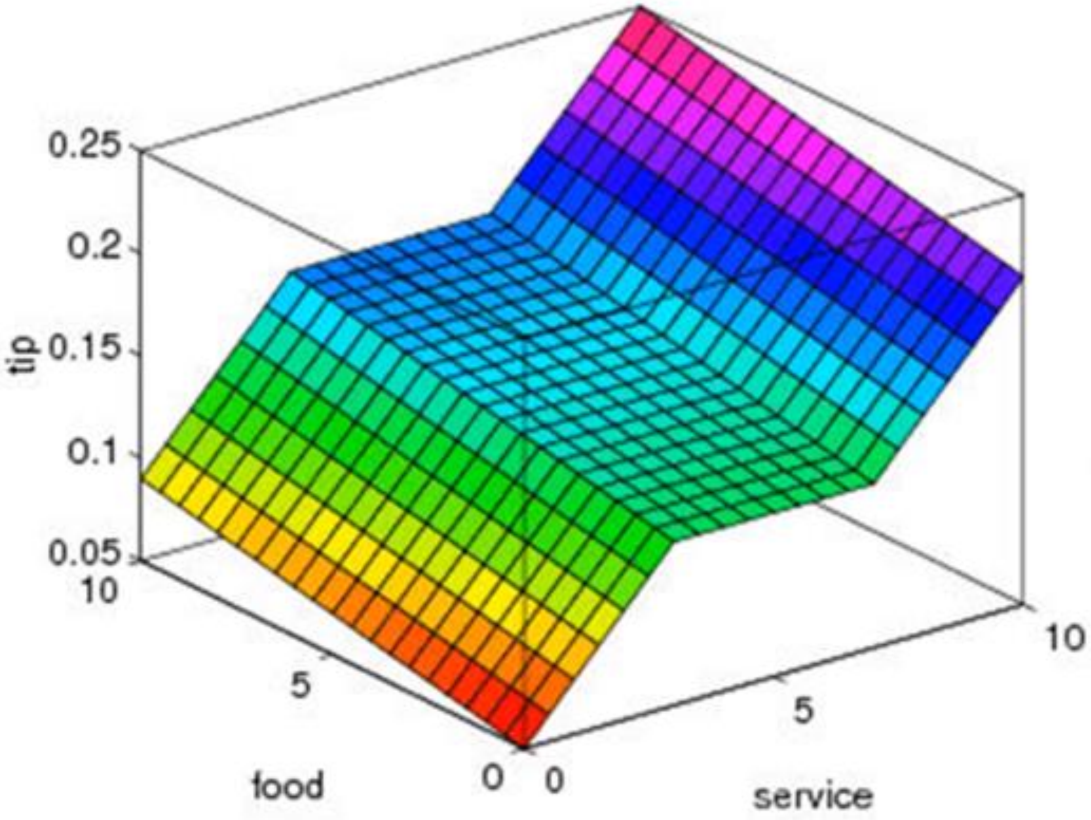
- Why Use Fuzzy Logic Toolbox
- Fuzzy Example by using Fuzzy Inference System (FIS)
- Defuzzification In MATLAB
- Algorithm: Fuzzy Controller Design Procedure
- Implementation of Fuzzy PID Controller in Simulink (PID, Fuzzy PID, Lookup Table)

Why Use Fuzzy Logic Toolbox

- Easy-to-use interface for applying modern fuzzy logic techniques.
- Perform fuzzy logic control and identification.
- Easily integrated into Model-Based design with Simulink blocks.
  - Possible to be used with other appropriate control techniques.
  - Possible to generate code for various uses.
- Supplies fuzzy inference engine.

Why Use Fuzzy Logic Toolbox

- Nonfuzzy Approach



Nonfuzzy approach

```
% Establish constants
lowTip=0.05; averTip=0.15; highTip=0.25;
tipRange=highTip-lowTip;
badService=0; okayService=3;
goodService=7; greatService=10;
serviceRange=greatService-badService;
badFood=0; greatFood=10;
foodRange=greatFood-badFood;

% If service is poor or food is rancid, tip is cheap
if service<okayService,
    tip=(((averTip-lowTip)/(okayService-badService)) ...
        *service+lowTip)*servRatio + ...
        (1-servRatio)*(tipRange/foodRange*food+lowTip);
% If service is good, tip is average
elseif service<goodService,
    tip=averTip*servRatio + (1-servRatio)* ...
        (tipRange/foodRange*food+lowTip);
% If service is excellent or food is delicious, tip is generous
else
    tip=(((highTip-averTip)/ ...
        (greatService-goodService))* ...
        (service-goodService)+averTip)*servRatio + ...
        (1-servRatio)*(tipRange/foodRange*food+lowTip);
end
```

Fuzzy Example by using Fuzzy Inference System (FIS)

- At restaurant, what should be the tip according to the quality of the service and the quality of the food?
- Two sets [0 10];
  - Quality of service
  - Quality of food
- What should be the tip ?

Fuzzy Example by using Fuzzy Inference System (FIS)

- Rules;
  - If service is poor or food is rancid, tip is cheap.
  - If service is good, tip is average.
  - If service is excellent or food is delicious, tip is genereous.
- Service (Input) Membership Function;
  - Poor (Gaussian) => (1.5, 0)
  - Good (Gaussian) => (1.5, 5)
  - Excellent (Gaussian) => (1.5, 10)
- Food (Input) Membership Function;
  - Rancid (Trapezoidal) => (0, 0, 1, 3)
  - Delicious (Trapezoidal) => (7, 8, 10, 10)
- Tip (Output) Membership Function;
  - Cheap (Triangular) => (0, 5, 10)
  - Average (Triangular) => (10, 15, 20)
  - Generous (Triangular) => (20, 25, 30)

Defuzzification in MATLAB

```matlab
%%
clc
clear all
close all

x = -10:0.1:10;

mf1 = trapmf(x,[-10 -8 -2 2]);
mf2 = trapmf(x,[-5 -3 2 4]);
mf3 = trapmf(x,[2 3 8 9]);
mf1 = max(0.5*mf2,max(0.9*mf1,0.1*mf3));

figure('Tag','defuzz');
plot(x,mf1,'LineWidth',3);
h_gca = gca;
h_gca.YTick = [0 .5 1] ;
ylim([-1 1]);

% CENTEROID
x1 = defuzz(x,mf1,'centroid');
h1 = line([x1 x1],[-0.2 1.2],'Color','k');
t1 = text(x1,-0.2,' centroid','FontWeight','bold');
```

```matlab
% BISECTOR
x2 =  defuzz(x,mf1,'bisector');

gray = 0.7*[1 1 1];
h1.Color = gray;
t1.Color = gray;
h2 = line([x2 x2],[-0.4 1.2],'Color','k');
t2 = text(x2,-0.4,' bisector','FontWeight','bold');

% MIDDLE, SMALLEST AND LARGEST OF MAXIMUM
x3 = defuzz(x,mf1,'mom');

x4 = defuzz(x,mf1,'som');

x5 = defuzz(x,mf1,'lom');

h2.Color = gray;
t2.Color = gray;

h3 = line([x3 x3],[-0.7 1.2],'Color','k');
t3 = text(x3,-0.7,' MOM','FontWeight','bold');
h4 = line([x4 x4],[-0.8 1.2],'Color','k');
t4 = text(x4,-0.8,' SOM','FontWeight','bold');
h5 = line([x5 x5],[-0.6 1.2],'Color','k');
t5 = text(x5,-0.6,' LOM','FontWeight','bold');
```

Algorithm: Fuzzy Controller Design Procedure

- Design a crisp PID controller. Use a PID tuning method to find Kp, 1/Ti and Td.
- Replace it with a linear fuzzy controller. (Theorem 3.1 in reference book).
- Make it nonlinear. Use a standard nonlinear control surface to begin with. Modify the rules by trial and error to reshape the control surface.
- Fine tune it. Use hand tuning: use GE to improve the rise time, use GCE tp dampen overshoot and use GIE to remove any steady state error.

Implementation of Fuzzy PID Controller in Simulink (PID, Fuzzy PID, Lookup Table)

- open_system(‘sllookuptable’) [https://www.mathworks.com/help/fuzzy/implement-fuzzy-pid-controller-in-simulink-using-lookup-table.html]
- Design Conventional PID Controller
- Design Equivalent Linear Fuzzy PID Controller
- Implement Fuzzy Inference System Using 2D Lookup Table
- Design Fuzzy PID Controller with Nonlinear Control Surface

Design Equivalent Linear Fuzzy PID Controller

- E (Input) Membership Function [-10 10];
  - Negative (Triangular) => (-20, -10, 0)
  - Zero (Triangular) => (-10, 0, 10)
  - Positive (Triangular) => (0, 10, 20)
- CE (Input) Membership Function [-10 10];
  - Negative (Triangular) => (-20, -10, 0)
  - Zero (Triangular) => (-10, 0, 10)
  - Positive (Triangular) => (0, 10, 20)
- u (Output) Membership Function [-20 20];
  - LargeNegative (Constant) => (-20)
  - SmallNegative (Constant) => (-10)
  - Zero (Constant) => (0)
  - SmallPositive (Constant) => (10)
  - LargePositive (Constant) => (20)

Design Equivalent Linear Fuzzy PID Controller

- Rules;
  - If E is negative and CE is negative, then u is -20.
  - If E is negative and CE is zero, then u is -10.
  - If E is negative and CE is positive, then u is 0.
  - If E is zero and CE is negative, then u is -10.
  - If E is zero and CE is zero, then u is 0.
  - If E is zero and CE is positive, then u is 10.
  - If E is positive and CE is negative, then u is 0.
  - If E is positive and CE is zero, then u is 10.
  - If E is positive and CE is positive, then u is 20.

Implement Fuzzy Inference System Using 2D Lookup Table

- The fuzzy controller block has two inputs and one output. Therefore, you can replace the fuzzy system using a 2-D lookup table.
- Use function ‘evalfis’.
- Since the control surface is linear, you can use a few sample points for each input variable.

```
Step = 10;
E = -10:Step:10;
CE = -10:Step:10;
N = length(E);
LookUpTableData = zeros(N);
for i=1:N
   for j=1:N
      % Compute output u for each combination of sample points.
      LookUpTableData(i,j) = evalfis(FIS,[E(i) CE(j)]);
   end
end
```

Design Fuzzy PID Controller with Nonlinear Control Surface

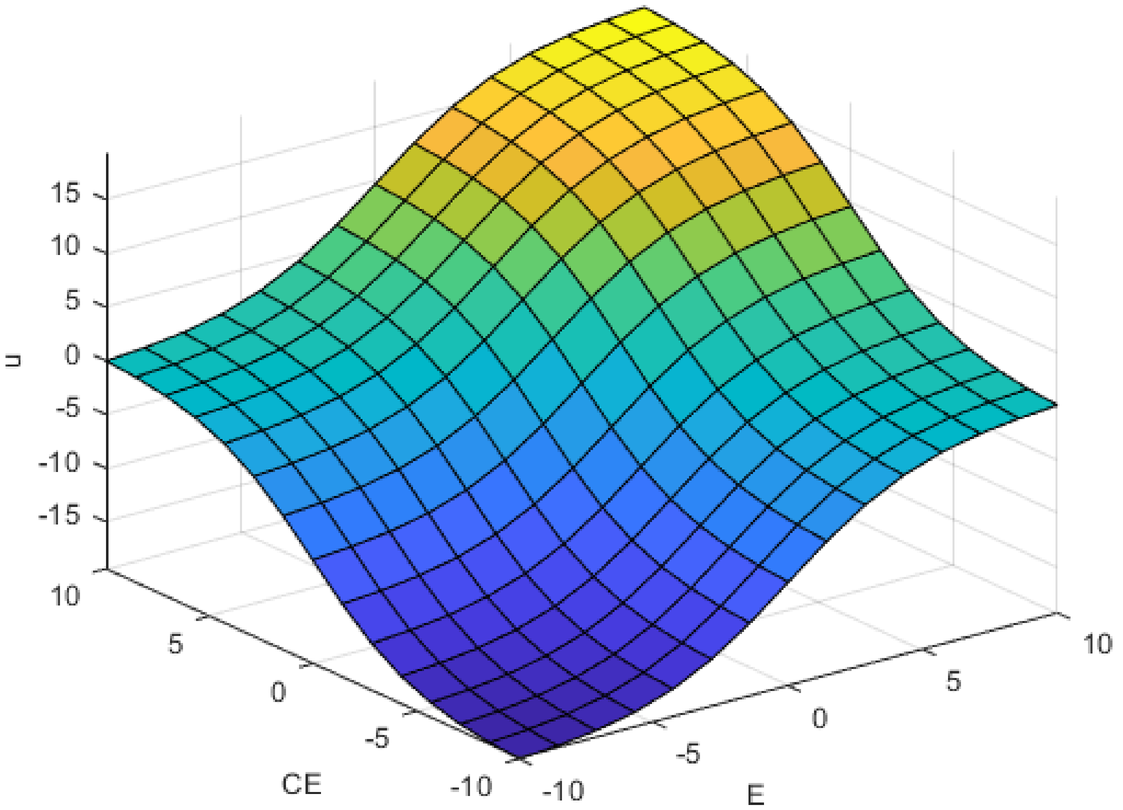
- You can obtain a nonlinear control surface by adjusting your FIS settings, such as its style, membership functions and rule base.
- E (Input) Membership Function [-10 10];
  - Negative (Gaussian) => (7, -10)
  - Positive (Gaussian) => (7, 10)
- CE (Input) Membership Function [-10 10];
  - Negative (Gaussian) => (7, -10)
  - Positive (Gaussian) => (7, 10)
- u (Output) Membership Function [-20 20];
  - Min (Constant) => (-20)
  - Zero (Constant) => (0)
  - Max (Constant) => (20)

Design Fuzzy PID Controller with Nonlinear Control Surface

- Rules;
  - If E is negative and CE is negative, then u is -20.
  - If E is negative and CE is positive, then u is 0.
  - If E is positive and CE is negative, then u is 0.
  - If E is positive and CE is positive, then u is 20.

Design Fuzzy PID Controller with Nonlinear Control Surface

- This surface has a higher gain near the center of the E and CE plane than the linear surface has.
- When the error is large, the controller becomes less agressive to avoid possible saturation.



Design Fuzzy PID Controller with Nonlinear Control Surface

- Update the lookup table with the new control surface data.
- Since the surface is nonlinear, to obtain a sufficient approximation, add more sample points.

```
Step = 1;
E = -10:Step:10;
CE = -10:Step:10;
N = length(E);
LookUpTableData = zeros(N);
for i=1:N
   for j=1:N
      % Compute output u for each combination of sample points.
      LookUpTableData(i,j) = evalfis(FIS,[E(i) CE(j)]);
   end
end
```

Design Fuzzy PID Controller with Nonlinear Control Surface

- Compared with the traditional linear PID controller, the nonlinear fuzzy PID controller reduces the overshoot by 50%.