**实验四 二阶模糊逻辑控制系统仿真**

一、实验目的

1．熟悉在MATLAB下，模糊逻辑推理系统的创建、输入、输出变量设置及其论域、隶属度函数的选取设置、模糊逻辑规则的建立以及输入输出特性曲面浏览；

2．熟悉利用模糊逻辑控制工具箱或利用模糊推理系统的基本函数来建立模糊逻辑推理系统；

3．了解模糊推理系统在控制系统中的应用。

二、实验内容

假设某一工业过程可等效成以下二阶系统：

设计一个模糊控制器，使其能自动建立模糊规则库，保证控制规则如表1所示，这种规则可表示为：

(1)

式(1)中，fix为取整函数；E为误差的模糊集；DE为误差导数的模糊集；为常数。

表1 模糊控制规则

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| u e  de | NB | NS | ZR | PS | PB |
| NB | PB | PB | PS | PS | ZR |
| NS | PB | PS | PS | ZR | ZR |
| ZR | PS | PS | ZR | ZR | NS |
| PS | PS | ZR | ZR | NS | NS |
| PB | ZR | ZR | NS | NS | NB |

这样表示的模糊控制系统可以通过改变值方便地修改如表1所示的模糊控制规则，从而自动建立系统的模糊规则库。

设计模糊控制器，使其能自动建立模糊规则库，保证系统输出尽快跟随系统输入。

要求建立脚本实现上述模糊逻辑控制系统。采样时间T=0.01秒；系统输入r(t)=1.0。观察不同取值时，阶跃响应的变化情况，找到匹配较佳动态性能的值，并进行详细的实验分析。

参考代码

clear all;

%被控系统建模

num=20;den=[8 6 1];

[A,b,c,d]=tf2ss(num,den);

%系统参数

T=0.01;h=T;

N=500;R=1.0\*ones(1,N);

uu=zeros(1,N); yy=zeros(3,N);

ka=1;

for alpha=[0.45 0.75 0.90];

%定义输入/输出变量及其隶属度函数

fisMat=newfis('n4');

fisMat=addvar(fisMat,'input','e',[-6,6]);

fisMat=addvar(fisMat,'input','de',[-6,6]);

fisMat=addvar(fisMat,'output','u',[-6,6]);

fisMat=addmf(fisMat,'input',1, 'NB','trapmf',[-6 -6 -5 -3]);

fisMat=addmf(fisMat,'input',1,'NS','trapmf',[-5 -3 -2 0]);

fisMat=addmf(fisMat,'input',1,'ZR','trimf',[-2 0 2]);

fisMat=addmf(fisMat,'input',1,'PS','trapmf',[0 2 3 5]);

fisMat=addmf(fisMat,'input',1,'PB','trapmf',[3 5 6 6]);

fisMat=addmf(fisMat,'input',2,'NB','trapmf',[-6 -6 -5 -3]);

fisMat=addmf(fisMat,'input',2,'NS','trapmf',[-5 -3 -2 0]);

fisMat=addmf(fisMat,'input',2,'ZR','trimf',[-2 0 2]);

fisMat= addmf(fisMat,'input',2,'PS','trapmf',[0 2 3 5]);

fisMat=addmf(fisMat,'input',2,'PB','trapmf',[3 5 6 6]);

fisMat=addmf(fisMat,'output',1,'NB','trapmf',[-6 -6 -5 -3]);

fisMat=addmf(fisMat,'output',1,'NS','trapmf',[-5 -3 -2 0]);

fisMat=addmf(fisMat,'output',1,'ZR','trimf',[-2 0 2]);

fisMat=addmf(fisMat,'output',1,'PS','trapmf',[0 2 3 5]);

fisMat=addmf(fisMat,'output',1,'PB','trapmf',[3 5 6 6]);

%模糊规则矩阵

for i=1:5

for j=1:5

rr(i,j)=round(alpha\*i+(1-alpha)\*j);

end

end

rr=6-rr;

r1=zeros(prod(size(rr)),3);

k=1;

for i=1:size(rr,1)

for j=1:size(rr,2)

r1(k,:)=[i,j,rr(i,j)];

k=k+1;

end

end

[r,s]=size(r1);

r2=ones(r,2);

rulelist=[r1 r2];

fisMat=addrule(fisMat,rulelist);

%模糊控制系统仿真

Ke=30;Kd=0.2;

Ku=1.0;x=[0;0];

e=0;de=0;

for k=1:N

e1=Ke\*e;

de1=Kd\*de;

%将模糊控制器的输入变量变换到论域

if e1>=6

e1=6;

elseif e1<=-6

e1=-6;

end

if de1>=6;

de1=6;

elseif de1<=-6

de1=-6;

end

%计算模糊控制器的输出

in=[e1 de1];

uu(1,k)=Ku\*evalfis(in,fisMat);

u=uu(1,k);

%利用四阶龙格-库塔法计算系统输出

K1=A\*x+b\*u;

K2=A\*(x+h\*K1/2)+b\*u;

K3=A\*(x+h\*K2/2)+b\*u;

K4=A\*(x+h\*K3)+b\*u;

x=x+(K1+2\*K2+2\*K3+K4)\*h/6;

y=c\*x+d\*u;

yy(ka,k)=y;

%计算误差和误差微分

e1=e;e=y-R(1,k);

de=(e-e1)/T;

end

ka=ka+1;

end

%绘制结果曲线

kk=[1:N]\*T;

plot(kk,yy(1,:),'r:',kk,yy(2,:),'r-.',kk,yy(3,:),'k--',kk,R,'m');

xlabel('时间');ylabel('输出');

legend('alpha=0.45','alpha=0.75','alpha=0.90');

grid on;

**四、实验过程及分析**

1. **实验代码及分析**

首先是对被控系统进行建模，由二阶系统可知，传递函数分子的num为20，den为传递函数分母的系数。利用tf2ss()函数将传递函数转换为状态空间表达式。状态空间的标准表达式为：

其中，x为n维向量，u为p维向量，y为q维向量，A为的矩阵，B为的矩阵，C为的矩阵，D为的矩阵。在本次实验中，u(t)为模糊控制器的输出量。

num = 20; *% 传递函数的分子*

den = [8 6 1]; *% 传递函数分母的系数*

[A, b, c, d] = tf2ss(num, den); *% 将传递函数转换为状态空间表达式*

设置采样时间为0.01秒，系统的输入，即稳态值为1。

T = 0.01; *% 采样时间*

h = T;

N = 1000;

R = 1.0 \* ones(1, N); *% 稳态为1*

uu = zeros(1, N);

yy = zeros(3, N);

rr = zeros(5, 5);

ka = 1;

随后，定义一个名为“M11\_6”的模糊推理系统，使用addvar()函数向模糊推理系统添加语言变量函数，包含2个输入，分别是学生的e和de，以及一个输出u，同时为其指定论域范围。然后为语言变量添加相应的隶属度函数，例如为“NB”、“NS”、“PS”和“PB”添加了梯形隶属度函数，为“ZR”添加了三角形隶属度函数。

取为不同的值，依据每个的值，通过公式(1)会得到不同的模糊规则，再根据这些模糊规则来进行模糊控制系统的仿真，得到模糊控制器的输出。

利用四阶龙格-库塔法对状态空间标准式计算系统输出，因为已知方程导数和初值信息，所以可以省去求解微分方程的复杂过程，在本次实验中得到的方程如下：

其中：

随后得到当前值下的系统输出。

**for** alpha = [0.45 0.75 0.90] *% 计算alpha为不同值时系统的输出*

*% 定义输入/输出变量及其隶属函数*

fisMat = newfis('M11\_6'); *% 定义模糊控制系统*

fisMat = addvar(fisMat, 'input', 'e', [-6, 6]);

fisMat = addvar(fisMat, 'input', 'de', [-6, 6]);

fisMat = addvar(fisMat, 'output', 'u', [-6, 6]);

fisMat = addmf(fisMat, 'input', 1, 'NB', 'trapmf', [-6 -6 -5 -3]);

fisMat = addmf(fisMat, 'input', 1, 'NS', 'trapmf', [-5 -3 -2 0]);

fisMat = addmf(fisMat, 'input', 1, 'ZR', 'trimf', [-2 0 2]);

fisMat = addmf(fisMat, 'input', 1, 'PS', 'trapmf', [0 2 3 5]);

fisMat = addmf(fisMat, 'input', 1, 'PB', 'trapmf', [3 5 6 6]);

fisMat = addmf(fisMat, 'input', 2, 'NB', 'trapmf', [-6 -6 -5 -3]);

fisMat = addmf(fisMat, 'input', 2, 'NS', 'trapmf', [-5 -3 -2 0]);

fisMat = addmf(fisMat, 'input', 2, 'ZR', 'trimf', [-2 0 2]);

fisMat = addmf(fisMat, 'input', 2, 'PS', 'trapmf', [0 2 3 5]);

fisMat = addmf(fisMat, 'input', 2, 'PB', 'trapmf', [3 5 6 6]);

fisMat = addmf(fisMat, 'output', 1, 'NB', 'trapmf', [-6 -6 -5 -3]);

fisMat = addmf(fisMat, 'output', 1, 'NS', 'trapmf', [-5 -3 -2 0]);

fisMat = addmf(fisMat, 'output', 1, 'ZR', 'trimf', [-2 0 2]);

fisMat = addmf(fisMat, 'output', 1, 'PS', 'trapmf', [0 2 3 5]);

fisMat = addmf(fisMat, 'output', 1, 'PB', 'trapmf', [3 5 6 6]);

*%模糊规则矩阵*

**for** i = 1 : 5

**for** j = 1 : 5

rr(i, j) = round(alpha \* i + (1 - alpha) \* j);

**end**

**end**

rr = 6 - rr;

r1 = zeros(numel(rr), 3); *% 25 \* 3的矩阵*

k = 1;

**for** i = 1 : size(rr, 1) *% 取rr的行*

**for** j = 1 : size(rr, 2) *% 取rr的列*

r1(k, :) = [i, j, rr(i, j)]; *% 得到各输入语言变量的语言值*

k = k + 1;

**end**

**end**

[r, s] = size(r1);

r2 = ones(r, 2); *% 设置规则权重为1，以及模糊规则前件的各语言变量是"与"的关系*

rulelist = [r1 r2];

fisMat = addRule(fisMat, rulelist);

*% 模糊控制系统仿真*

Ke = 30;

Kd = 0.2;

Ku = 1.0;

ki = 0.03;

x = [0; 0];

e = 0;

de = 0;

ie = 0; *% 用于消除稳态误差*

**for** k = 1 : N

e1 = Ke \* e;

de1 = Kd \* de;

*% 将模糊控制器的输入变量变换到论域*

**if** e1 >= 6

e1 = 6;

**elseif** e1 <= -6

e1 = -6;

**end**

**if** de1 >= 6

de1 = 6;

**elseif** de1 <= -6

de1 = -6;

**end**

*% 计算模糊控制器的输出*

in = [e1 de1]; *% 输入向量*

uu(1, k) = Ku \* evalfis(in, fisMat) - ki \* ie; *% 计算输出*

u = uu(1, k);

*% 利用**四阶龙格-库塔法对状态空间标准式计算系统输出*

K1 = A \* x + b \* u;

K2 = A \* (x + h \* K1 / 2) + b \* u;

K3 = A \* (x + h \* K2 / 2) + b \* u;

K4 = A \* (x + h \* K3) + b \* u;

x = x + (K1 + 2 \* K2 + 2 \* K3 + K4) \* h / 6;

y = c \* x + d \* u;

yy(ka, k) = y; *% 记录当前 alpha 值下的输出*

*% 计算误差和误差微分*

e1 = e;

e = y - R(1, k);

de = (e - e1) / T;

ie = ie + e \* T;

**end**

ka = ka + 1;

**end**

最后，绘制在不同的取值下，模糊控制系统的输出曲线。

figure

linewidth = 2;

kk = [1 : N] \* T;

hold on

plot(kk, yy(1, :), 'r:', LineWidth = linewidth);

plot(kk, yy(2, :), 'b-', LineWidth = linewidth);

plot(kk, yy(3, :), 'k--', LineWidth = linewidth);

plot(kk, R, 'm', LineWidth = linewidth);

hold off

xlabel('时间');

ylabel('输出');

legend('alpha = 0.45', 'alpha = 0.75', 'alpha = 0.90');

grid on;

1. **实验结果及分析**

通过对的不同取值，由图4.1可以看到，当和时，系统对阶跃响应的超调量都比较大，且经过多次振荡之后才能逐渐接近稳态值，但也始终在稳态值的附近振荡。当

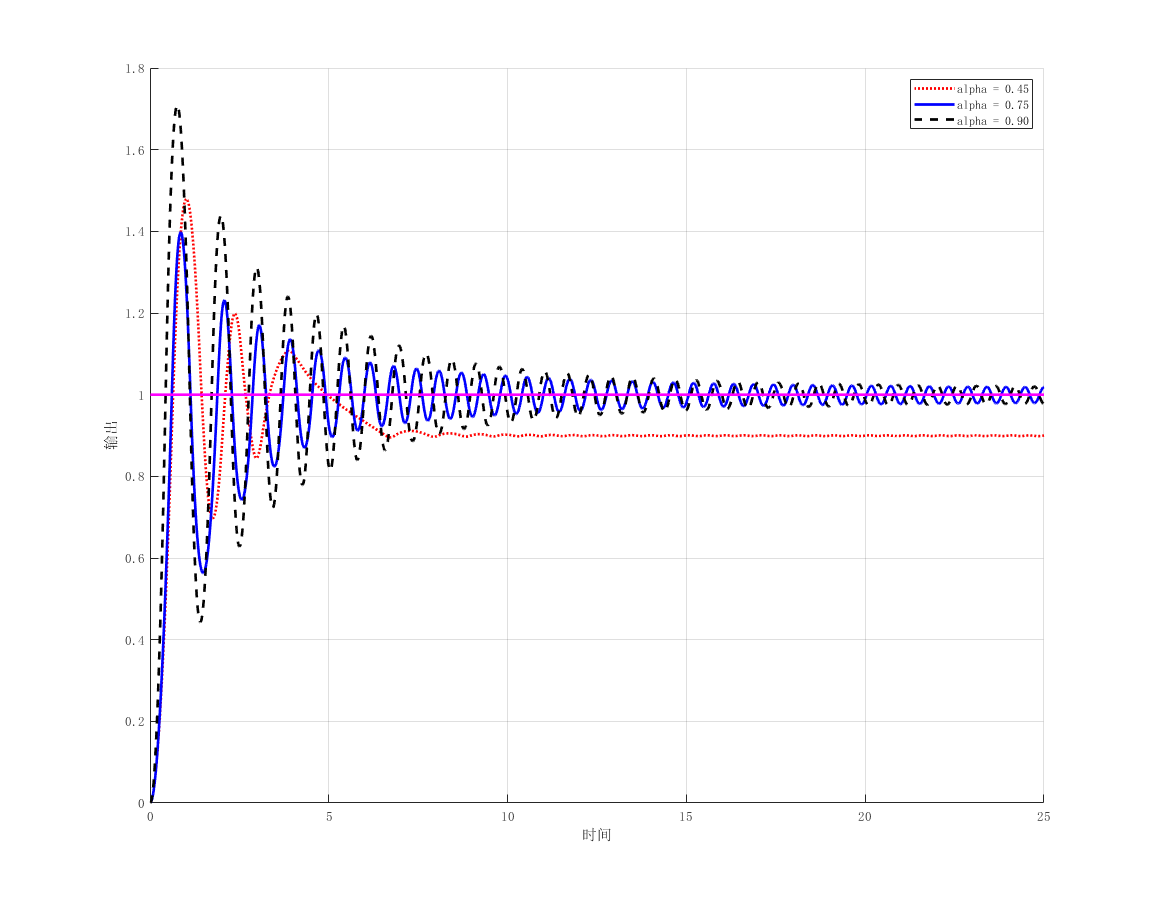
时超调量依旧比较大，但是能够趋于稳定，然而有一定的稳态误差。

图4.1 实验结果1

为了消除时的稳态误差，我引入了误差e的积分ie用来消除稳态误差，在原来的循环体中加入以下代码：

ki = 0.03;

ie = 0;

uu(1, k) = Ku \* evalfis(in, fisMat) - ki \* ie; *% 计算输出*

ie = ie + e \* T;

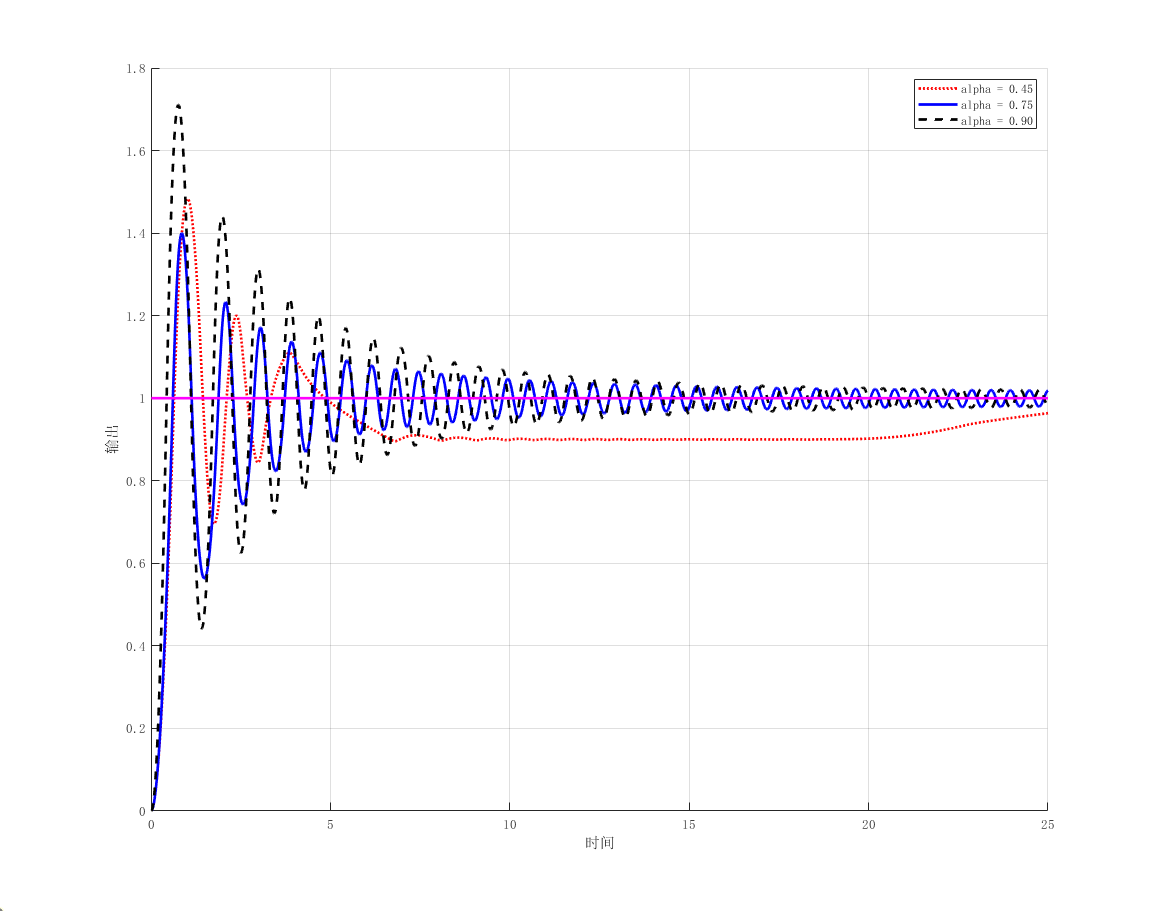
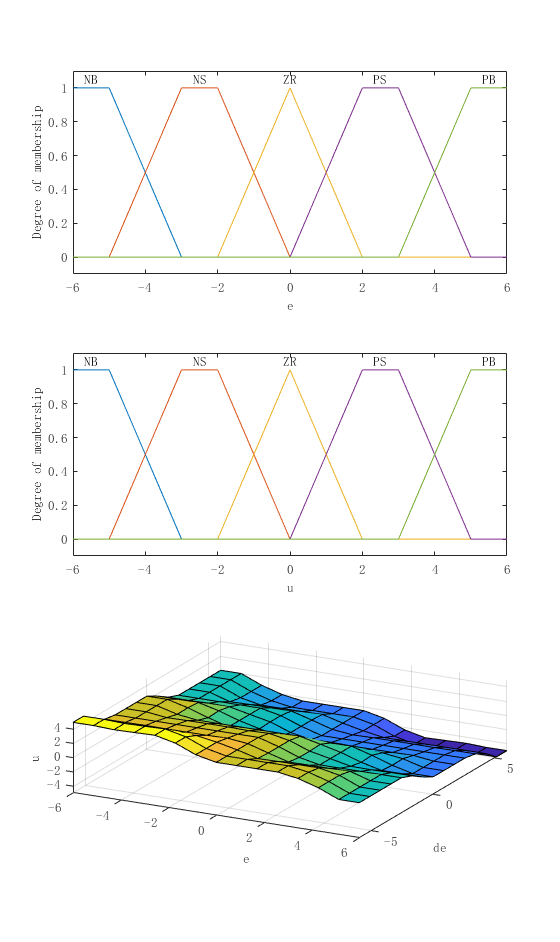
 最后得到改进后的结果如图4.2所示。

图4.2 实验结果2

此时，系模糊控制系统的隶属度函数图像和输入/输出特性曲面如图4.3所示。

图4.3 隶属度函数图像和输入/输出特性曲面