第 3 章　专家 PID 控制和模糊 PID 控制

3.1　专家 PID 控制

3.1.1　专家 PID 控制原理

专家控制（Expert Control）的实质是基于受控对象和控制规律的各种知识，并以智能的方式利用这些知识来设计控制器。利用专家经验来设计 PID 参数便构成专家 PID 控制。

典型的二阶系统单位阶跃响应误差曲线如图 3-1 所示。对于典型的二阶系统阶跃响应过程作如下分析。

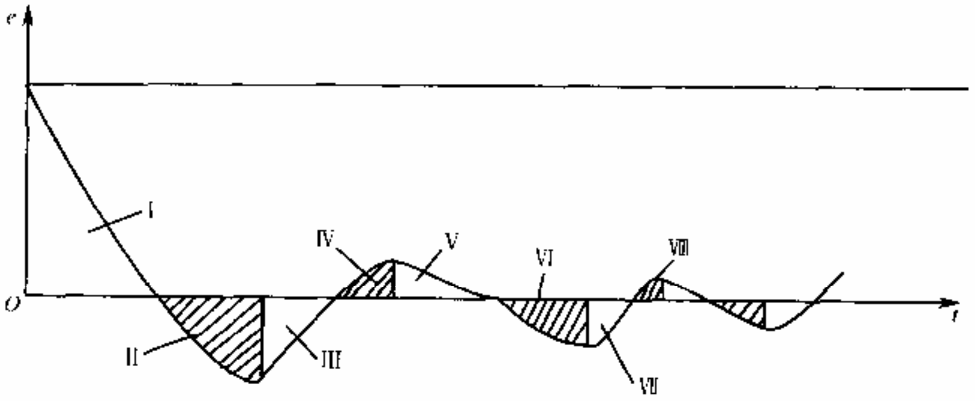

图 3-1　典型二阶系统单位阶跃响应误差曲线

令 $e(k)$表示离散化的当前采样时刻的误差值，$e(k-1)$、$e(k-2)$分别表示前一个和前两个采样时刻的误差值，则有

$$\begin{aligned}\Delta e(k)&=e(k)-e(k-1)\\ \Delta e(k-1)&=e(k-1)-e(k-2)\end{aligned}\tag{3.1}$$

根据误差及其变化，可设计专家 PID 控制器，该控制器可分为以下五种情况进行设计：

（1）当$|e(k)|>M_1$时，说明误差的绝对值已经很大。不论误差变化趋势如何，都应考虑控制器的输出应按最大（或最小）输出，以达到迅速调整误差，使误差绝对值以最大速度减小。此时，它相当于实施开环控制。

（2）当$e(k)\Delta e(k)>0$时，说明误差在朝误差绝对值增大方向变化，或误差为某一常值，未发生变化。此时，如果$|e(k)|\geq M_2$，说明误差也较大，可考虑由控制器实施较强的控制作用，以达到扭转误差绝对值朝减小方向变化，并迅速减小误差的绝对值，控制器输出可为

$$u(k)=u(k-1)+k_1\{k_p[e(k)-e(k-1)]+k_i e(k)+k_d[e(k)-2e(k-1)+e(k-2)]\}\tag{3.2}$$

此时，如果$|e(k)|<M_2$，说明尽管误差朝绝对值增大方向变化，但误差绝对值本身并不

很大，可考虑控制器实施一般的控制作用，只要扭转误差的变化趋势，使其朝误差绝对值减小方向变化，控制器输出为

$$u(k)=u(k-1)+k_p[e(k)-e(k-1)]+k_i e(k)+k_d[e(k)-2e(k-1)+e(k-2)] \tag{3.3}$$

（3）当$e(k)\Delta e(k)<0$、$\Delta e(k)\Delta e(k-1)>0$或者$e(k)=0$时，说明误差的绝对值朝减小的方向变化，或者已经达到平衡状态。此时，可考虑采取保持控制器输出不变。

（4）当$e(k)\Delta e(k)<0$、$\Delta e(k)\Delta e(k-1)<0$时，说明误差处于极值状态。如果此时误差的绝对值较大，即$|e(k)|\geq M_2$，可考虑实施较强的控制作用

$$u(k)=u(k-1)+k_1 k_p e_m(k) \tag{3.4}$$

如果此时误差的绝对值较小，即$|e(k)|<M_2$，可考虑实施较弱的控制作用

$$u(k)=u(k-1)+k_2 k_p e_m(k) \tag{3.5}$$

（5）当$|e(k)|\leq\varepsilon$时，说明误差的绝对值很小，此时加入积分，减少稳态误差。

式中，$e_m(k)$——误差e的第k个极值；

$u(k)$——第k次控制器的输出；

$u(k-1)$——第$k-1$次控制器的输出；

k_1——增益放大系数，$k_1>1$；

k_2——抑制系数，$0<k_2<1$；

M_1，M_2——设定的误差界限，$M_1>M_2$；

k——控制周期的序号(自然数)；

ε——任意小的正实数。

图 3-1 中，Ⅰ、Ⅲ、Ⅴ、Ⅶ、…区域，误差朝绝对值减小的方向变化。此时，可采取保持等待措施，相当于实施开环控制；Ⅱ、Ⅳ、Ⅵ、Ⅷ、…区域，误差绝对值朝增大的方向变化。此时，可根据误差的大小分别实施较强或一般的控制作用，以抑制动态误差。

3.1.2 仿真程序及分析

仿真实例

求三阶传递函数的阶跃响应

$$G_p(s)=\frac{523500}{s^3+87.35s^2+10470s}$$

其中对象采样时间为1ms。

采用专家 PID 设计控制器。在仿真过程中，ε取 0.001，程序中的五条规则与控制算法的五种情况相对应，其结果如图 3-2 至图 3-3 所示。

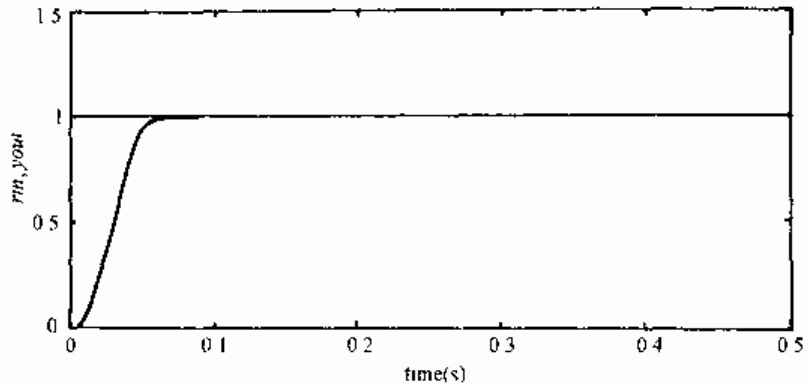

图 3-2　专家 PID 控制阶跃响应曲线

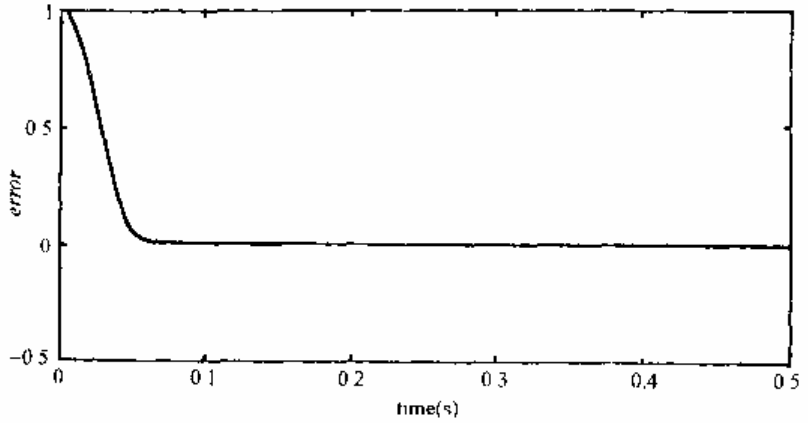

图 3-3　误差响应曲线

仿真程序：chap3_1.m

```
%Expert PID Controller
clear all;
close all;
ts=0.001;

sys=tf(5.235e005,[1,87.35,1.047e004,0]);
dsys=c2d(sys,ts,'z');
[num,den]=tfdata(dsys,'v');

u_1=0.0;u_2=0.0;u_3=0.0;
y_1=0;y_2=0;y_3=0;

x=[0,0,0]';
x2_1=0;

kp=0.6;
ki=0.03;
kd=0.01;
```

```
error_1=0;
for k=1:1:500
time(k)=k*ts;

rin(k)=1.0;                    %Tracing Jieyue Signal

u(k)=kp*x(1)+kd*x(2)+ki*x(3); %PID Controller

%Expert control rule
if abs(x(1))>0.8      %Rule1:Unclosed control firstly
   u(k)=0.45;
elseif abs(x(1))>0.40
   u(k)=0.40;
elseif abs(x(1))>0.20
   u(k)=0.12;
elseif abs(x(1))>0.01
   u(k)=0.10;
end

if x(1)*x(2)>0|(x(2)==0)       %Rule2
   if abs(x(1))>=0.05
      u(k)=u_1+2*kp*x(1);
   else
      u(k)=u_1+0.4*kp*x(1);
   end
end

if (x(1)*x(2)<0&x(2)*x2_1>0)|(x(1)==0)   %Rule3
    u(k)=u(k);
end

if x(1)*x(2)<0&x(2)*x2_1<0   %Rule4
   if abs(x(1))>=0.05
      u(k)=u_1+2*kp*error_1;
   else
      u(k)=u_1+0.6*kp*error_1;
   end
end
```

```
    if abs(x(1))<=0.001   %Rule5:Integration separation PI control
      u(k)=0.5*x(1)+0.010*x(3);
    end

    %Restricting the output of controller
    if u(k)>=10
      u(k)=10;
    end
    if u(k)<= 10
      u(k)=-10;
    end

    %Linear model
    yout(k)=-den(2)*y_1-den(3)*y_2-
den(4)*y_3+num(1)*u(k)+num(2)*u_1+num(3)*u_2+num(4)*u_3;
    error(k)=rin(k)-yout(k);

    %  - ------Return of PID parameters-- ---------%
    u_3=u_2;u_2=u_1;u_1=u(k);
    y_3=y_2;y_2=y_1;y_1=yout(k);

    x(1)=error(k);              % Calculating P
    x2_1=x(2);
    x(2)=(error(k)-error_1)/ts;   % Calculating D
    x(3)=x(3)+error(k)*ts;        % Calculating I

    error_1=error(k);
    end
    figure(1);
    plot(time,rin,'b',time,yout,'r');
    xlabel('time(s)');ylabel('rin,yout');
    figure(2);
    plot(time,rin-yout,'r');
    xlabel('time(s)');ylabel('error');
```

3.2 模糊自适应整定 PID 控制

3.2.1 模糊自适应整定 PID 控制原理

在工业生产过程中，许多被控对象随着负荷变化或干扰因素影响，其对象特性参数或

结构发生改变。自适应控制运用现代控制理论在线辨识对象特征参数，实时改变其控制策略，使控制系统品质指标保持在最佳范围内，但其控制效果的好坏取决于辨识模型的精确度，这对于复杂系统是非常困难的。因此，在工业生产过程中，大量采用的仍然是 PID 算法，PID 参数的整定方法很多，但大多数都以对象特性为基础。

随着计算机技术的发展，人们利用人工智能的方法将操作人员的调整经验作为知识存入计算机中，根据现场实际情况，计算机能自动调整 PID 参数，这样就出现了智能 PID 控制器。这种控制器把古典的 PID 控制与先进的专家系统相结合，实现系统的最佳控制。这种控制必须精确地确定对象模型，首先将操作人员（专家）长期实践积累的经验知识用控制规则模型化，然后运用推理便可对 PID 参数实现最佳调整。

由于操作者经验不易精确描述，控制过程中各种信号量以及评价指标不易定量表示，模糊理论是解决这一问题的有效途径，所以人们运用模糊数学的基本理论和方法，把规则的条件、操作用模糊集表示，并把这些模糊控制规则以及有关信息(如评价指标、初始 PID 参数等)作为知识存入计算机知识库中，然后计算机根据控制系统的实际响应情况（即专家系统的输入条件），运用模糊推理，即可自动实现对 PID 参数的最佳调整，这就是模糊自适应 PID 控制。模糊自适应 PID 控制器目前有多种结构形式，但其工作原理基本一致。

自适应模糊 PID 控制器以误差 e 和误差变化 ec 作为输入，可以满足不同时刻的 e 和 ec 对 PID 参数自整定的要求。利用模糊控制规则在线对 PID 参数进行修改，便构成了自适应模糊 PID 控制器，其结构如图 3-4 所示。

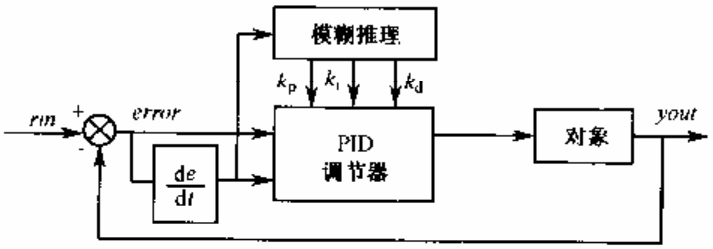

图 3-4　自适应模糊控制器结构

PID 参数模糊自整定是找出 PID 三个参数与 e 和 ec 之间的模糊关系，在运行中通过不断检测 e 和 ec，根据模糊控制原理来对 3 个参数进行在线修改，以满足不同 e 和 ec 时对控制参数的不同要求，而使被控对象有良好的动、静态性能。

从系统的稳定性、响应速度、超调量和稳态精度等各方面来考虑，k_p, k_i, k_d 的作用如下：

（1）比例系数 k_p 的作用是加快系统的响应速度，提高系统的调节精度。k_p 越大，系统的响应速度越快，系统的调节精度越高，但易产生超调，甚至会导致系统不稳定。k_p 取值过小，则会降低调节精度，使响应速度缓慢，从而延长调节时间，使系统静态、动态特性变坏

（2）积分作用系数 k_i 的作用是消除系统的稳态误差。k_i 越大，系统的静态误差消除越快，但 k_i 过大，在响应过程的初期会产生积分饱和现象，从而引起响应过程的较大超调。若 k_i 过小，将使系统静态误差难以消除，影响系统的调节精度。

（3）微分作用系数 k_d 的作用是改善系统的动态特性，其作用主要是在响应过程中抑制

偏差向任何方向的变化，对偏差变化进行提前预报。但 k_d 过大，会使响应过程提前制动，从而延长调节时间，而且会降低系统的抗干扰性能。

PID 参数的整定必须考虑到在不同时刻三个参数的作用以及相互之间的互联关系。

在线实时模糊自整定 PID 控制器控制方案原理如图 3-4 所示。

模糊自整定 PID 是在 PID 算法的基础上，通过计算当前系统误差 e 和误差变化率 ec,利用模糊规则进行模糊推理，查询模糊矩阵表进行参数调整。

模糊控制设计的核心是总结工程设计人员的技术知识和实际操作经验，建立合适的模糊规则表，得到针对 k_p，k_i，k_d 三个参数分别整定的模糊控制表。

(1) k_p 的模糊规则表（见表 3-1）

表 3-1　k_p 的模糊规则表

ec / Δkp / e	NB	NM	NS	ZO	PS	PM	PB
NB	PB	PB	PM	PM	PS	ZO	ZO
NM	PB	PB	PM	PS	PS	ZO	NS
NS	PM	PM	PM	PS	ZO	NS	NS
ZO	PM	PM	PS	ZO	NS	NM	NM
PS	PS	PS	ZO	NS	NS	NM	NM
PM	PS	ZO	NS	NM	NM	NM	NB
PB	ZO	ZO	NM	NM	NM	NB	NB

（2）k_i 的模糊规则表（见表 3-2）

表 3-2　k_i 的模糊规则表

ec / Δki / e	NB	NM	NS	ZO	PS	PM	PB
NB	NB	NB	NM	NM	NS	ZO	ZO
NM	NB	NB	NM	NS	NS	ZO	ZO
NS	NB	NM	NS	NS	ZO	PS	PS
ZO	NM	NM	NS	ZO	PS	PM	PM
PS	NM	NS	ZO	PS	PS	PM	PB
PM	ZO	ZO	PS	PS	PM	PB	PB
PB	ZO	ZO	PS	PM	PM	PB	PB

（3）k_d 的模糊控制规则表（见表 3-3）

表 3-3　k_d 的模糊控制规则表

Δk_d \ ec / e	NB	NM	NS	ZO	PS	PM	PB
NB	PS	NS	NB	NB	NB	NM	PS
NM	PS	NS	NB	NM	NM	NS	ZO
NS	ZO	NS	NM	NM	NS	NS	ZO
ZO	ZO	NS	NS	NS	NS	NS	ZO
PS	ZO	ZO	ZO	ZO	ZO	ZO	ZO
PM	PB	NS	PS	PS	PS	PS	PB
PB	PB	PM	PM	PM	PS	PS	PB

k_p，k_i，k_d 的模糊控制规则表建立好后，可根据如下方法进行 k_p，k_i，k_d 的自适应校正。

将系统误差 e 和误差变化率 ec 变化范围定义为模糊集上的论域。

$$e,ec=\{-5,-4,-3,-2,-1,0,1,2,3,4,5\} \tag{3.6}$$

其模糊子集为 $e,ec=\{NB,NM,NS,O,PS,PM,PB\}$，子集中元素分别代表负大，负中，负小，零，正小，正中，正大。设 e,ec 和 k_p，k_i，k_d 均服从正态分布，因此可得出各模糊子集的隶属度，根据各模糊子集的隶属度赋值表和各参数模糊控制模型，应用模糊合成推理设计 PID 参数的模糊矩阵表，查出修正参数代入下式计算

$$\begin{aligned} k_p &= k_p' + \{e_i, ec_i\}_p \\ k_i &= k_i' + \{e_i, ec_i\}_i \\ k_d &= k_d' + \{e_i, ec_i\}_d \end{aligned} \tag{3.7}$$

在线运行过程中，控制系统通过对模糊逻辑规则的结果处理、查表和运算，完成对 PID 参数的在线自校正。其工作流程图如图 3-5 所示。

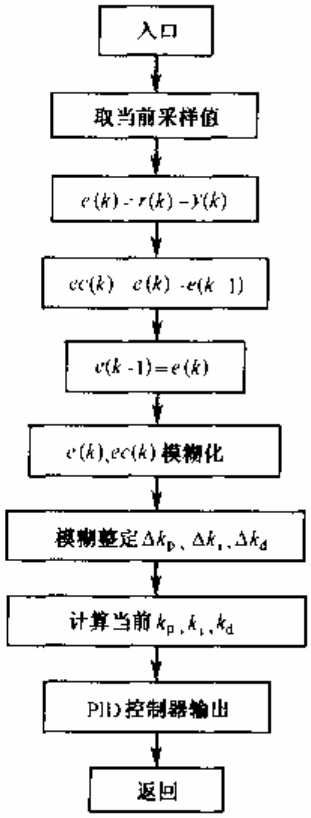

图 3-5　在线自校正工作流程图

3.2.2　仿真程序及分析

仿真实例

被控对象为

$$G_p(s)=\frac{523500}{s^3+87.35s^2+10470s}$$

采样时间为 1ms，采用模糊 PID 控制进行阶跃响应，在第 300 个采样时间时控制器输出加 1.0 的干扰，相应的响应结果如图 3-6 至 3-11 所示。

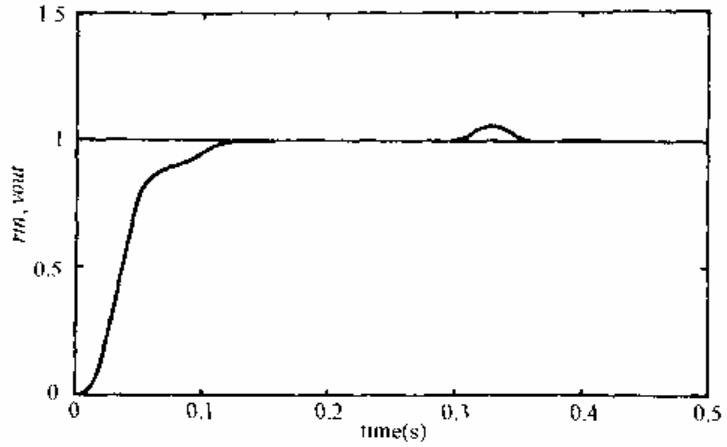

图 3-6　模糊 PID 控制阶跃响应

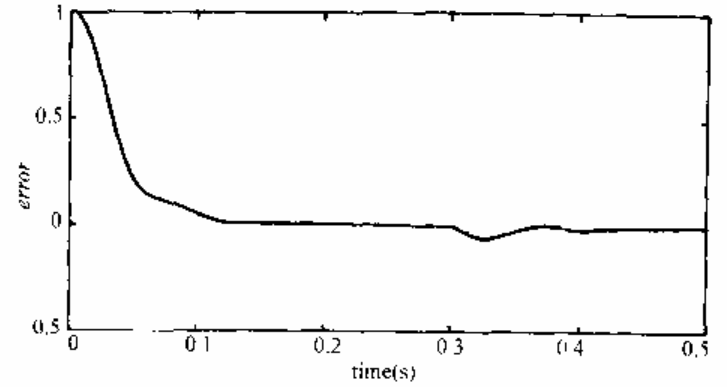

图 3-7　模糊 PID 控制误差响应

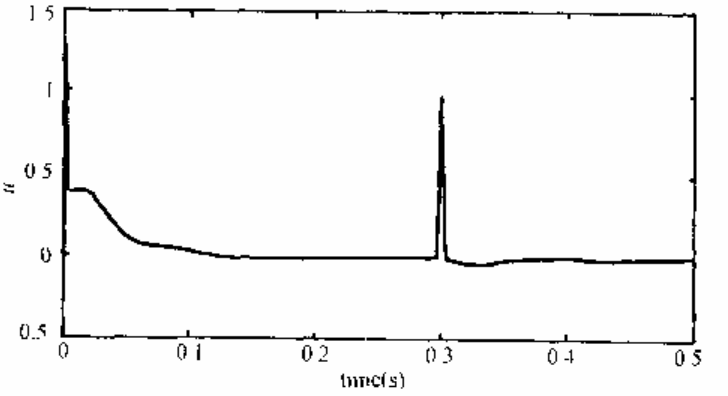

图 3-8　控制器输出 u

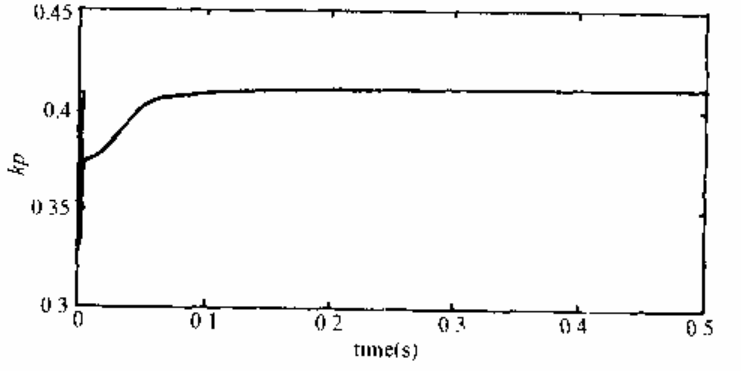

图 3-9　k_p 的自适应调整

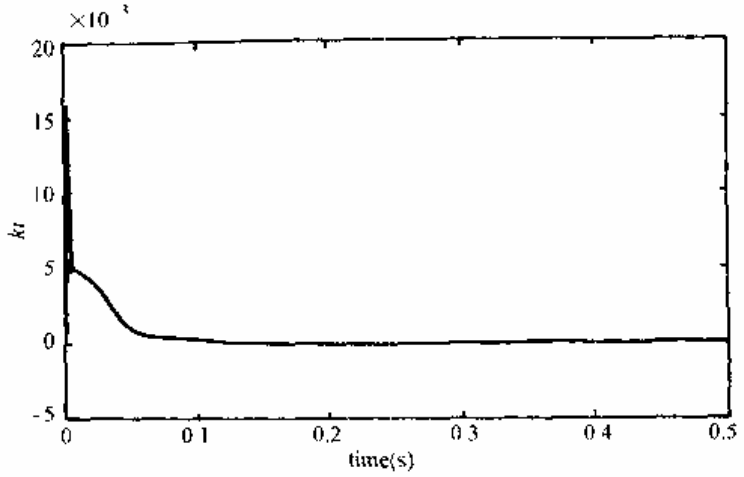

图 3-10　k_i 的自适应调整

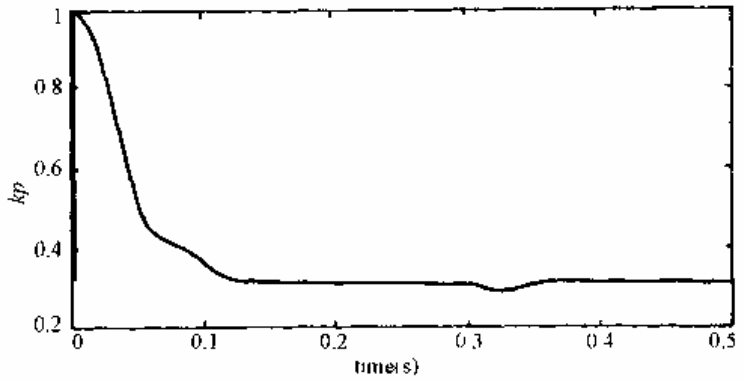

图 3-11　k_d 的自适应调整

仿真程序：chap3_2.m

```
%Fuzzy Tunning PID Control
clear all;
close all;

a=newfis('fuzzpid');

a=addvar(a,'input','e',[-3,3]);                    %Parameter e
a=addmf(a,'input',1,'NB','zmf',[-3,-1]);
a=addmf(a,'input',1,'NM','trimf',[-3,-2,0]);
a=addmf(a,'input',1,'NS','trimf',[-3,-1,1]);
a=addmf(a,'input',1,'Z','trimf',[-2,0,2]);
a=addmf(a,'input',1,'PS','trimf',[-1,1,3]);
a=addmf(a,'input',1,'PM','trimf',[0,2,3]);
a=addmf(a,'input',1,'PB','smf',[1,3]);

a=addvar(a,'input','ec',[-3,3]);                   %Parameter ec
a=addmf(a,'input',2,'NB','zmf',[-3,-1]);
```

```
a=addmf(a,'input',2,'NM','trimf',[-3,-2,0]);
a=addmf(a,'input',2,'NS','trimf',[-3,-1,1]);
a=addmf(a,'input',2,'Z','trimf',[ 2,0,2]);
a=addmf(a,'input',2,'PS','trimf',[-1,1,3]);
a=addmf(a,'input ,2,'PM','trimf ,[0,2,3]);
a=addmf(a,'input',2,'PB','smf',[1,3]);

a=addvar(a,'output','kp',[ 0.3,0.3]);                    %Parameter kp
a=addmf(a,'output',1,'NB','zmf',[-0.3,-0.1]);
a=addmf(a,'output',1,'NM','trimf',[-0.3,-0.2,0]);
a=addmf(a,'output',1,'NS','trimf',[-0.3, 0.1,0.1]);
a=addmf(a,'output',1,'Z','trimf',[-0.2,0,0.2]);
a=addmf(a,'output',1,'PS','trimf',[ 0.1,0.1,0.3]);
a=addmf(a,'output',1,'PM','trimf',[0,0.2,0.3]);
a=addmf(a,'output',1,'PB','smf',[0.1,0.3]);

a=addvar(a,'output','ki ,[-0.06,0.06]);                  %Parameter ki
a=addmf(a,'output',2,'NB','zmf',[-0.06,-0.02]);
a=addmf(a,'output',2,'NM','trimf',[-0.06,-0.04,0]);
a=addmf(a,'output',2,'NS','trimf',[-0.06,-0.02,0.02]);
a=addmf(a,'output',2,'Z','trimf',[-0.04,0,0.04_);
a=addmf(a,'output',2,'PS','trimf',[-0.02,0.02,0.06]);
a=addmf(a,'output',2,'PM','trimf','0,0.04,0.06]);
a=addmf(a,'output',2,'PB','smf',[0.02,0.06]);

a=addvar(a,'output','kd',[-3,3]);                        %Parameter kp
a=addmf(a,'output',3,'NB','zmf',[ 3, 1]);
a=addmf(a,'output',3,'NM','trimf',[ 3,-2,0]);
a=addmf(a,'output',3,'NS','trimf',[ 3,-1,1]);
a=addmf(a,'output',3,'Z','trimf',[-2,0,2]);
a=addmf(a,'output',3,'PS','trimf',[ 1,1,3]);
a=addmf(a,'output',3,'PM','trimf',[0,2,3]);
a=addmf(a,'output',3,'PB','smf',[1,3]);

rulelist=[1 1 7 1 5 1 1;
          1 2 7 1 3 1 1;
        1 3 6 2 1 1 1;
        1 4 6 2 1 1 1;
        1 5 5 3 1 1 1;
        1 6 4 4 2 1 1;
```

```
1 7 4 4 5 1 1;

2 1 7 1 5 1 1;
2 2 7 1 3 1 1;
2 3 6 2 1 1 1;
2 4 5 3 2 1 1;
2 5 5 3 2 1 1;
2 6 4 4 3 1 1;
2 7 3 4 4 1 1;

3 1 6 1 4 1 1;
3 2 6 2 3 1 1;
3 3 6 3 2 1 1;
3 4 5 3 2 1 1;
3 5 4 4 3 1 1;
3 6 3 5 3 1 1;
3 7 3 5 4 1 1;

4 1 6 2 4 1 1;
4 2 6 2 3 1 1;
4 3 5 3 3 1 1;
4 4 4 4 3 1 1;
4 5 3 5 3 1 1;
4 6 2 6 3 1 1;
4 7 2 6 4 1 1;

5 1 5 2 4 1 1;
5 2 5 3 4 1 1;
5 3 4 4 4 1 1;
5 4 3 5 4 1 1;
5 5 3 5 4 1 1;
5 6 2 6 4 1 1;
5 7 2 7 4 1 1;

6 1 5 4 7 1 1;
6 2 4 4 5 1 1;
6 3 3 5 5 1 1;
6 4 2 5 5 1 1;
6 5 2 6 5 1 1;
6 6 2 7 5 1 1;
```

```
        6 7 1 7 7 1 1;

        7 1 4 4 7 1 1;
        7 2 4 4 6 1 1;
        7 3 2 5 6 1 1;
        7 4 2 6 6 1 1;
        7 5 2 6 5 1 1;
        7 6 1 7 5 1 1;
        7 7 1 7 7 1 1];

a=addrule(a,rulelist);
a=setfis(a,'DefuzzMethod','mom');
writefis(a,'fuzzpid');

a=readfis('fuzzpid');

%PID Controller
ts=0.001;
sys=tf(5.235e005,[1,87.35,1.047e004,0]);
dsys=c2d(sys,ts,'tustin');
[num,den]=tfdata(dsys,'v');

u_1=0.0;u_2=0.0;u_3=0.0;

y_1=0;y_2=0;y_3=0;

x=[0,0,0]';

error_1=0;
e_1=0.0;
ec_1=0.0;

kp0=0.40;
kd0=1.0;
ki0=0.0;

for k=1:1:500
time(k)=k*ts;

rin(k)=1;
```

```
    %Using fuzzy inference to tunning PID
    k_pid=evalfis([e_1,ec_1],a);
    kp(k)=kp0+k_pid(1);
    ki(k)=ki0+k_pid(2);
    kd(k)=kd0+k_pid(3);
    u(k)=kp(k)*x(1)+kd(k)*x(2)+ki(k)*x(3);

    if k==300    % Adding disturbance(1.0v at time 0.3s)
      u(k)=u(k)+1.0;
    end
    if u(k)>=10
      u(k)=10;
    end
    if u(k)<=-10
      u(k)=-10;
    end

    yout(k)=-den(2)*y_1-den(3)*y_2-
den(4)*y_3+num(1)*u(k)+num(2)*u_1+num(3)*u_2+num(4)*u_3;
    error(k)=rin(k)-yout(k);
    %%%%%%%%%%%%%%Return of PID parameters%%%%%%%%%%%%%%
      u_3=u_2;
      u_2=u_1;
      u_1=u(k);

      y_3=y_2;
      y_2=y_1;
      y_1=yout(k);

      x(1)=error(k);              % Calculating P
      x(2)=error(k)-error_1;       % Calculating D
      x(3)=x(3)+error(k);          % Calculating I

      e_1=x(1);
      ec_1=x(2);

      error_2=error_1;
      error_1=error(k);
    end
    showrule(a)
```

```
figure(1);plot(time,rin,'b',time,yout,'r');
xlabel('time(s)');ylabel('rin,yout');
figure(2);plot(time,error,'r');
xlabel('time(s)');ylabel('error');
figure(3);plot(time,u,'r');
xlabel('time(s)');ylabel('u');
figure(4);plot(time,kp,'r');
xlabel('time(s)');ylabel('kp');
figure(5);plot(time,ki,'r');
xlabel('time(s)');ylabel('ki');
figure(6);plot(time,kd,'r');
xlabel('time(s)');ylabel('kd');
figure(7);plotmf(a,'input',1);
figure(8);plotmf(a,'input',2);
figure(9);plotmf(a,'output',1);
figure(10);plotmf(a,'output',2);
figure(11);plotmf(a,'output',3);
plotfis(a);
fuzzy fuzzpid.fis
```

通过仿真，还可以得到以下几个结果：

（1）在 MATLAB 下运行 plotmf(a,'input', 1)可得到模糊系统第一个输入 e 的隶属函数，同理可得到 de，k_p，k_i，k_d 的隶属函数，如图 3-12 至 3-16 所示。

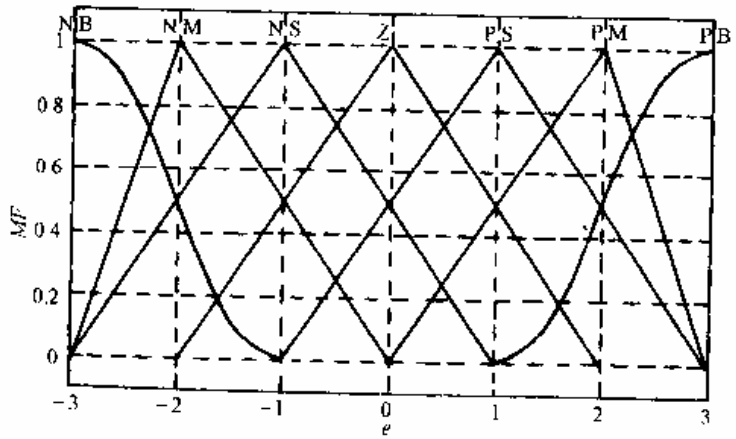

图 3-12　误差的隶属函数

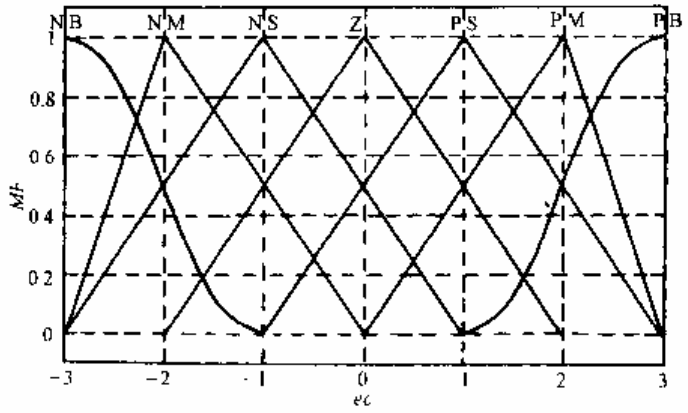

图 3-13　误差变化率的隶属函数

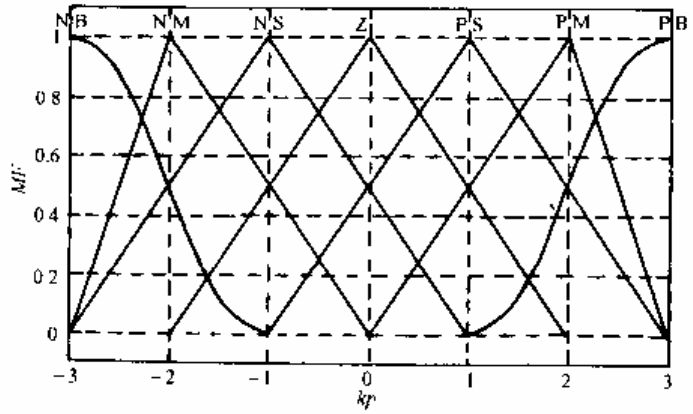

图 3-14　k_p 的隶属函数

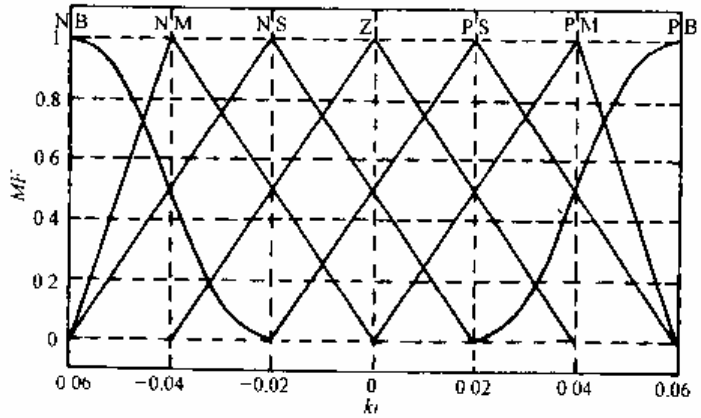

图 3-15　k_i 的隶属函数

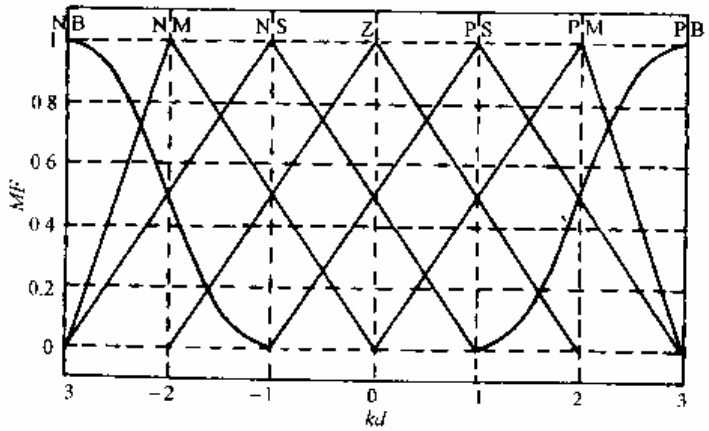

图 3-16　k_d 的隶属函数

（2）在 MATLAB 下运行 plotfis（a）可观察模糊控制系统的构成，如图 3-17 所示。

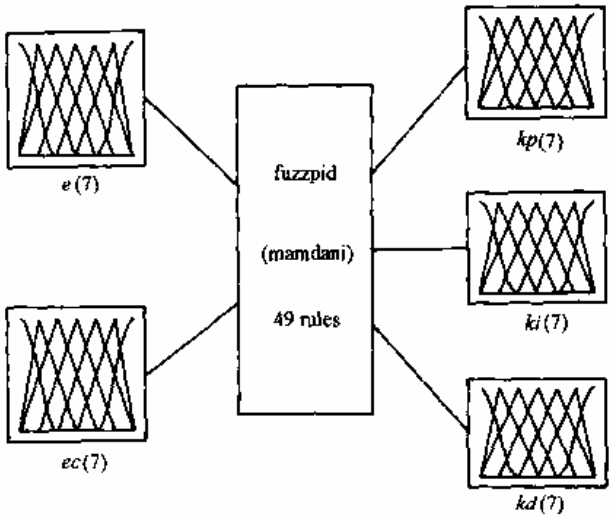

图 3-17　模糊 PID 控制系统构成

（3）在 MATLAB 下运行 fuzzy fuzzpid.fis 可进入 MATLAB 动态仿真工具箱动态仿真环境，如图 3-18 所示。

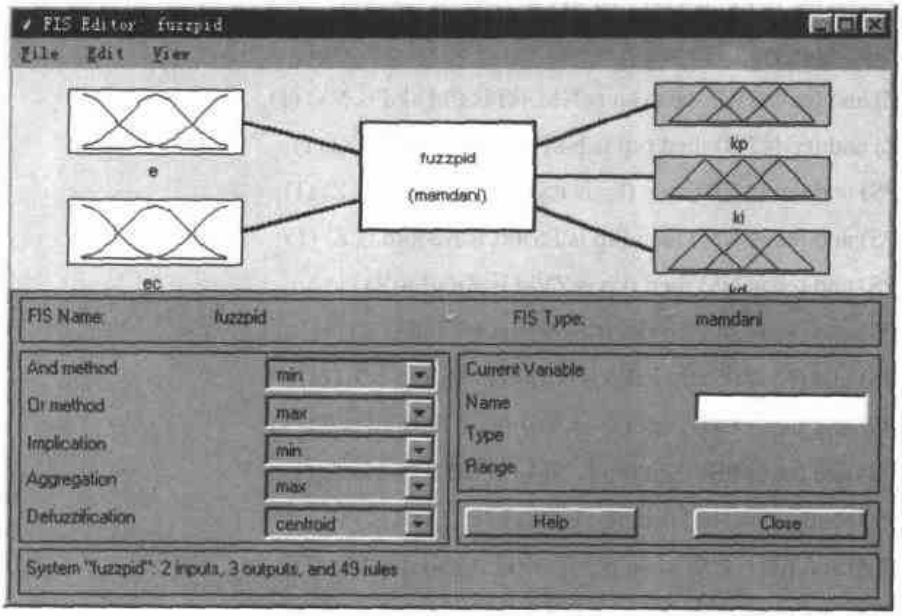

图 3-18　模糊 PID 动态仿真环境

（4）MATLAB 下运行 showrule(a)，可得到以下 49 条模糊规则：

1. If (e is NB) and (ec is NB) then (kp is PB)(ki is NB)(kd is PS) (1)
2. If (e is NB) and (ec is NM) then (kp is PB)(ki is NB)(kd is NS) (1)
3. If (e is NB) and (ec is NS) then (kp is PM)(ki is NM)(kd is NB) (1)
4. If (e is NB) and (ec is Z) then (kp is PM)(ki is NM)(kd is NB) (1)
5. If (e is NB) and (ec is PS) then (kp is PS)(ki is NS)(kd is NB) (1)
6. If (e is NB) and (ec is PM) then (kp is Z)(ki is Z)(kd is NM) (1)
7. If (e is NB) and (ec is PB) then (kp is Z)(ki is Z)(kd is PS) (1)
8. If (e is NM) and (ec is NB) then (kp is PB)(ki is NB)(kd is PS) (1)

9. If (e is NM) and (ec is NM) then (kp is PB)(ki is NB)(kd is NS) (1)
10. If (e is NM) and (ec is NS) then (kp is PM)(ki is NM)(kd is NB) (1)
11. If (e is NM) and (ec is Z) then (kp is PS)(ki is NS)(kd is NM) (1)
12. If (e is NM) and (ec is PS) then (kp is PS)(ki is NS)(kd is NM) (1)
13. If (e is NM) and (ec is PM) then (kp is Z)(ki is Z)(kd is NS) (1)
14. If (e is NM) and (ec is PB) then (kp is NS)(ki is Z)(kd is Z) (1)
15. If (e is NS) and (ec is NB) then (kp is PM)(ki is NB)(kd is Z) (1)
16. If (e is NS) and (ec is NM) then (kp is PM)(ki is NM)(kd is NS) (1)
17. If (e is NS) and (ec is NS) then (kp is PM)(ki is NS)(kd is NM) (1)
18. If (e is NS) and (ec is Z) then (kp is PS)(ki is NS)(kd is NM) (1)
19. If (e is NS) and (ec is PS) then (kp is Z)(ki is Z)(kd is NS) (1)
20. If (e is NS) and (ec is PM) then (kp is NS)(ki is PS)(kd is NS) (1)
21. If (e is NS) and (ec is PB) then (kp is NS)(ki is PS)(kd is Z) (1)
22. If (e is Z) and (ec is NB) then (kp is PM)(ki is NM)(kd is Z) (1)
23. If (e is Z) and (ec is NM) then (kp is PM)(ki is NM)(kd is NS) (1)
24. If (e is Z) and (ec is NS) then (kp is PS)(ki is NS)(kd is NS) (1)
25. If (e is Z) and (ec is Z) then (kp is Z)(ki is Z)(kd is NS) (1)
26. If (e is Z) and (ec is PS) then (kp is NS)(ki is PS)(kd is NS) (1)
27. If (e is Z) and (ec is PM) then (kp is NM)(ki is PM)(kd is NS) (1)
28. If (e is Z) and (ec is PB) then (kp is NM)(ki is PM)(kd is Z) (1)
29. If (e is PS) and (ec is NB) then (kp is PS)(ki is NM)(kd is Z) (1)
30. If (e is PS) and (ec is NM) then (kp is PS)(ki is NS)(kd is Z) (1)
31. If (e is PS) and (ec is NS) then (kp is Z)(ki is Z)(kd is Z) (1)
32. If (e is PS) and (ec is Z) then (kp is NS)(ki is PS)(kd is Z) (1)
33. If (e is PS) and (ec is PS) then (kp is NS)(ki is PS)(kd is Z) (1)
34. If (e is PS) and (ec is PM) then (kp is NM)(ki is PM)(kd is Z) (1)
35. If (e is PS) and (ec is PB) then (kp is NM)(ki is PB)(kd is Z) (1)
36. If (e is PM) and (ec is NB) then (kp is PS)(ki is Z)(kd is PB) (1)
37. If (e is PM) and (ec is NM) then (kp is Z)(ki is Z)(kd is PS) (1)
38. If (e is PM) and (ec is NS) then (kp is NS)(ki is PS)(kd is PS) (1)
39. If (e is PM) and (ec is Z) then (kp is NM)(ki is PS)(kd is PS) (1)
40. If (e is PM) and (ec is PS) then (kp is NM)(ki is PM)(kd is PS) (1)
41. If (e is PM) and (ec is PM) then (kp is NM)(ki is PB)(kd is PS) (1)
42. If (e is PM) and (ec is PB) then (kp is NB)(ki is PB)(kd is PB) (1)
43. If (e is PB) and (ec is NB) then (kp is Z)(ki is Z)(kd is PB) (1)
44. If (e is PB) and (ec is NM) then (kp is Z)(ki is Z)(kd is PM) (1)
45. If (e is PB) and (ec is NS) then (kp is NM)(ki is PS)(kd is PM) (1)
46. If (e is PB) and (ec is Z) then (kp is NM)(ki is PM)(kd is PM) (1)
47. If (e is PB) and (ec is PS) then (kp is NM)(ki is PM)(kd is PS) (1)
48. If (e is PB) and (ec is PM) then (kp is NB)(ki is PB)(kd is PS) (1)

49. If (e is PB) and (ec is PB) then (kp is NB)(ki is PB)(kd is PB) (1)

3.3 模糊免疫 PID 控制算法

3.3.1 模糊免疫 PID 控制算法原理

常规增量式 PID 控制器离散形式如下

$$\begin{aligned} u(k)&=u(k-1)+k_p(e(k)-e(k-1))+k_i e(k)+k_d(e(k)-2e(k-1)+e(k-2)) \\ &=u(k-1)+k_p((e(k)-e(k-1))+\frac{k_i}{k_p}e(k)+\frac{k_d}{k_p}(e(k)-2e(k-1)+e(k-2))) \end{aligned} \tag{3.8}$$

式中，k_p, k_i, k_d 分别为比例、积分和微分系数

P 控制器的控制算法为

$$u(k)=k_p e(k) \tag{3.9}$$

免疫 PID 控制器是借鉴生物系统的免疫机理而设计出的一种非线性控制器。免疫是生物体的一种特性生理反应。生物的免疫系统对于外来侵犯的抗原，可产生相应的抗体来抵御。抗原和抗体结合后，会产生一系列的反应，通过吞噬作用或产生特殊酶的作用而毁坏抗原。生物的免疫系统由淋巴细胞和抗体分子组成，淋巴细胞又由胸腺产生的 T 细胞（分别为辅助细胞 T_H 和抑制细胞 T_S）和骨髓产生的 B 细胞组成。当抗原侵入机体并经周围细胞消化后，将信息传递给 T 细胞，即传递给 T_H 细胞和 T_S 细胞，然后刺激 B 细胞。B 细胞产生抗体以消除抗原。当抗原较多时，机体内的 T_H 细胞也较多，而 T_S 细胞却较少，从而会产生较多的 B 细胞。随着抗原的减少，体内 T_S 细胞增多，它抑制了 T_H 细胞的产生，则 B 细胞也随着减少。经过一段时间间隔后，免疫反馈系统便趋于平衡。抑制机理和主反馈机理之间的相互协作，是通过免疫反馈机理对抗原的快速反应和稳定免疫系统完成的。

免疫系统虽然十分复杂，但其抗御抗原的自适应能力却是十分明显的。生物信息系统的这些智能行为，为科学和工程领域提供了各种理论参考和技术方法。基于上述免疫反馈原理，提出了免疫 PID 控制器：假设第 k 代的抗原数量为 $\varepsilon(k)$，由抗原刺激的 T_H 细胞的输出为 $T_H(k)$，T_S 细胞对 B 细胞的影响为 $T_S(k)$，则 B 细胞接收的总刺激为

$$S(k)=T_H(k)-T_S(k) \tag{3.10}$$

式中，$T_H(k)=k_1\varepsilon(k), T_S(k)=k_2 f(\Delta s(k))\varepsilon(k)$

若以抗原的数量 $\varepsilon(k)$ 作为偏差 $e(k)$，B 细胞接收的总刺激 $S(k)$ 作为控制输入 $u(k)$，则有如下的反馈控制规律

$$u(k)=K(1-\eta f(u(k),\Delta u(k)))e(k)=k_{p1}e(k) \tag{3.11}$$

式中，$k_{p1}=K(1-\eta f(u(k),\Delta u(k))$，$K=k_1$ 为控制反应速度，$\eta=\frac{k_2}{k_1}$ 为控制稳定效果，$f(\cdot)$ 为一选定的非线性函数。

利用模糊控制器可逼近非线性函数 $f(\cdot)$：每个输入变量被三个模糊集模糊化，分别是“正”（P）和“负”（N）；输出变量被三个模糊集模糊化，分别是“正”（P）、零“Z”和负（N）。以上隶属度函数都定义在整个 $(+\infty,-\infty)$ 区间。模糊控制器可采用以下四条规则：

（1）If u is P and △u is P then f(u, △u) is N（1）

（2）If u is P and △u is N then f(u, △u) is Z（1）

（3）If u is N and △u is P then f(u, △u) is Z（1）

（4）If u is N and △u is N then f(u, △u) is P（1）

各规则中，使用Zadeh的模糊逻辑 AND 操作，并采用常用的 mom 反模糊化方法得到模糊控制器的输出 $f(\cdot)$。

基于免疫反馈原理的控制器实际上就是一个非线性 P 控制器，其比例系数 $k_{p1}=K(1-\eta f(u(k),\Delta u(k))$ 随控制器输出的变化而变化，其中 K 为增益，则免疫 PID 控制器的输出为

$$\begin{aligned} u(k)&=u(k-1)+k_{p1}((e(k)-e(k-1))+\frac{k_i}{k_p}e(k)+\frac{k_d}{k_p}(e(k)-2e(k-1)+e(k-2))) \\ &=u(k-1)+k_{p1}((e(k)-e(k-1))+k_i'e(k)+k_d'(e(k)-2e(k-1)+e(k-2))) \end{aligned} \tag{3.12}$$

3.3.2 仿真程序及分析

仿真实例

设被控对象为一延迟系统

$$G_p(s)=\frac{1}{60s+1}e^{-80s}$$

采样时间为 20s，采用免疫 PID 控制器式(3.12)，取 $K=0.30$，$\eta=0.80$，$k_i'=0.30, k_d'=0.30$。输入的指令信号为1.0sgn(sin(3πt))，仿真时间为 1000 个采样点。为了测试控制器的鲁棒性，在第500个采样时间时加入一个干扰，免役PID控制方波跟踪结果如图3-19和图3-20所示。仿真结果表明，免疫 PID 控制具有很好的控制效果和较高的鲁棒性。非线性函数 $f(\cdot)$ 为模糊控制的输出，模糊控制输入、输出隶属函数如图3-21至图3-23所示。

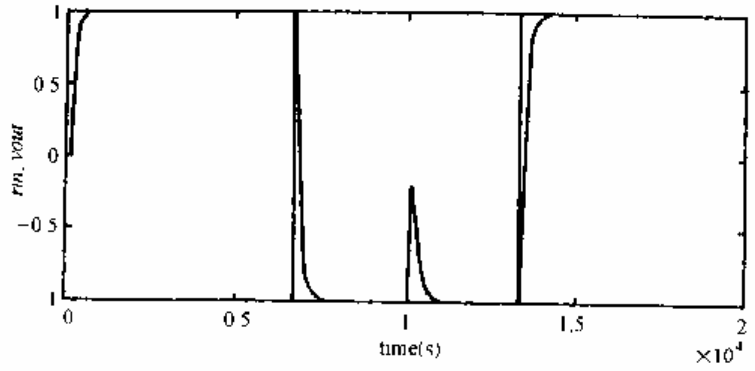

图 3-19 免疫 PID 控制方波跟踪

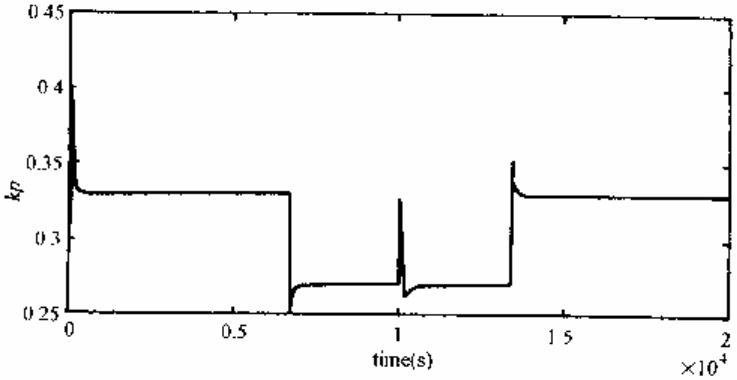

图 3-20 免疫 PID 控制 k_{p1} 的变化

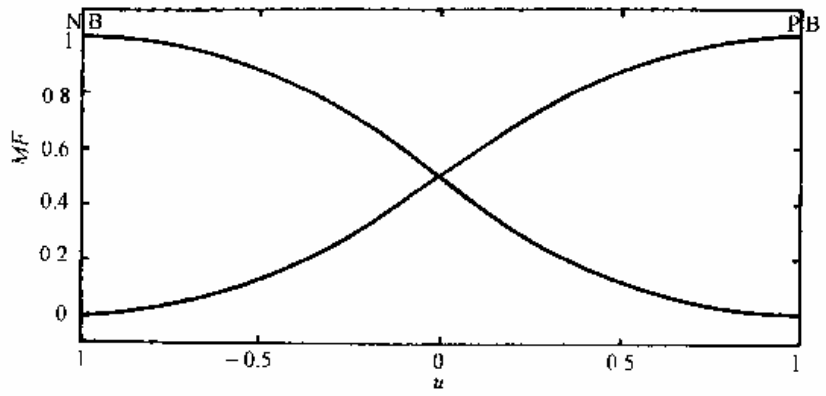

图 3-21　免疫 PID 控制 u 隶属函数

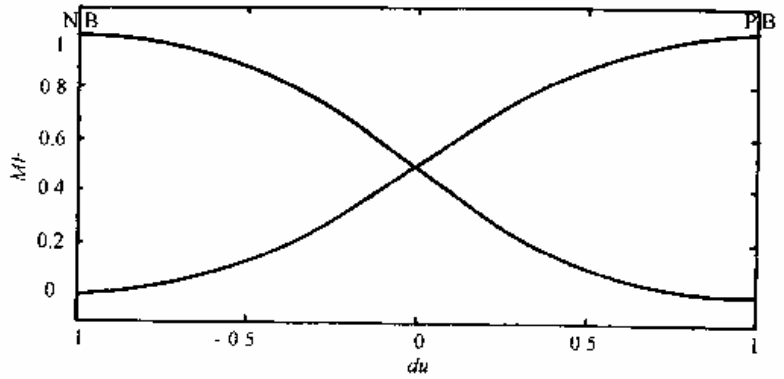

图 3-22　免疫 PID 控制 du 隶属函数

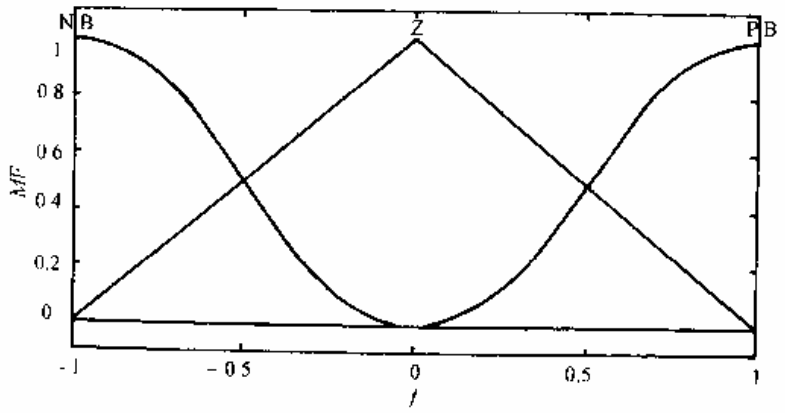

图 3-23　免疫 PID 控制 $f(\cdot)$ 隶属函数

仿真程序：chap3_3.m

```
%Fuzzy Immune PID Control
clear all;
close all;

a=newfis('fuzz_ljk');

f1=1.0;
a=addvar(a,'input','u',[-f1*1,f1*1]);          %Parameter e
```

```
a=addmf(a,'input',1,'NB','zmf',[-f1*1,f1*1]);
a=addmf(a,'input',1,'PB','smf',[-f1*1,f1*1]);

f2=1.0;
a=addvar(a,'input','du',[-f2*1,f2*1]);          %Parameter ec
a=addmf(a,'input',2,'NB','zmf',[-f2*1,f2*1]);
a=addmf(a,'input',2,'PB','smf',[-f2*1,f2*1]);

f3=1.0;
a=addvar(a,'output','f',[-f3*1,f3*1]);          %Parameter u
a=addmf(a,'output',1,'NB','zmf',[-f3*1,0]);
a=addmf(a,'output',1,'Z','trimf',[-f3*1,0,f3*1]);
a=addmf(a,'output',1,'PB','smf',[0,f3*1]);

rulelist=[2 2 1 1 1;    % Edit rule base
          2 1 2 1 1;
          1 2 2 1 1;
          1 1 3 1 1;];

a=addrule(a,rulelist);
%showrule(a)            % Show fuzzy rule base

a1=setfis(a,'DefuzzMethod','mom');  % Defuzzy
writefis(a1,'ljk');     % save to fuzzy file "ljk.fis" which can be
                        % simulated with fuzzy tool
a2=readfis('ljk');
%plotfis(a2);

%%%%%%%%%%%%%%%%% Using Fuzzy Controller%%%%%%%%%%%%%%%%
ts=20;
sys=tf([1],[60,1],'inputdelay',80);
dsys=c2d(sys,ts,'zoh');
[num,den]=tfdata(dsys,'v');

u_1=0;u_2=0;u_3=0;u_4=0;u_5=0;
y_1=0;
e_1=0;e_2=0;
for k=1:1:1000
time(k)=k*ts;
```

```
rin(k)=1.0*sign(sin(3*pi*k*0.001));

%Linear model
yout(k)=-den(2)*y_1+num(2)*u_5;

e(k)=rin(k)-yout(k);
ec(k)=e(k)-e_1;

f(k)=evalfis([u_1 u_1-u_2],a2);

K=0.30;
xite=0.80;
Kp(k)=K*(1-xite*f(k));

u(k)=u_1+Kp(k)*((e(k)-e_1)+0.3*e(k)+0.3*(e(k)-2*e_1+e_2));

if k==500
   u(k)=u(k)+1.0;
end

%Return of parameters
u_5=u_4;u_4=u_3;u_3=u_2;u_2=u_1;u_1=u(k);
y_1=yout(k);
e_2=e_1;
e_1=e(k);
end
figure(1);
plot(time,rin,'b',time,yout,'r');
xlabel('time(s)');ylabel('rin,yout');
figure(2);
plot(time,e,'r');
xlabel('time(s)');ylabel('error');
figure(3);
plot(time,u,'r');
xlabel('time(s)');ylabel('u');
figure(4);
plot(time,Kp,'r');
xlabel('time(s)');ylabel('Kp');
figure(5);
plotmf(a,'input',1);
```

```
figure(6);
plotmf(a,'input',2);
figure(7);
plotmf(a,'output',1);
```