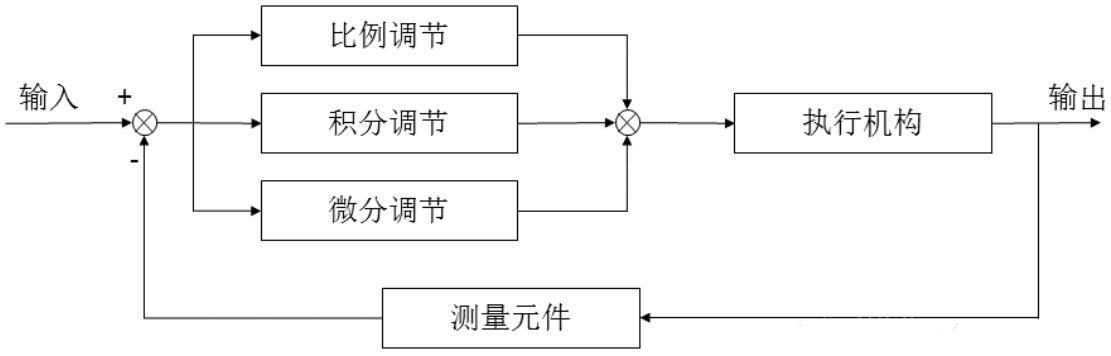
智能控制 专家 PID 系统

一、PID 算法介绍

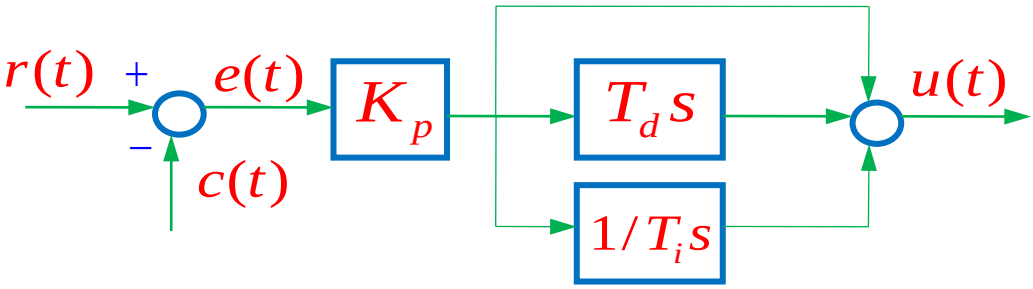
PID(Proportional Integral Derivative)控制是最早发展起来的控制策略之一，由于其算法简单、鲁棒性好和可靠性高，被广泛应用于工业过程控制，尤其适用于可建立精确数学模型的确定性控制系统。

在工程实际中，应用最为广泛的调节器控制规律为比例、积分、微分控制，简称 PID 控制，又称 PID 调节，它实际上是一种算法。PID 控制器问世至今已有近 70 年历史，它以其结构简单、稳定性好、工作可靠、调整方便而成为工业控制的主要技术之一。当被控对象的结构和参数不能完全掌握，或得不到精确的数学模型时，控制理论的其它技术难以采用时，系统控制器的结构和参数必须依靠经验和现场调试来确定，这时应用 PID 控制技术最为方便。即当我们不完全了解一个系统和被控对象，或不能通过有效的测量手段来获得系统参数时，最适合用 PID 控制技术。PID 控制器就是根据系统的误差，利用比例、积分、微分计算出控制量进行控制的。

从信号变换的角度而言，超前校正、滞后校正、滞后 - 超前校正可以总结为比例、积分、微分三种运算及其组合。



具体计算公式如下:



$$u(t) = K_p[e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{d}{dt}e(t)]$$

其中:

K_p 是 PID 控制器的比例系数;

T_i 是 PID 控制器的积分系数;

T_d 是 PID 控制器的微分系数。

1.1 比例环节(P): $K_p * e(t)$

成比例地反映控制系统的偏差信号$e(t)$，偏差一旦产生，控制器立即产生控制作用，产生相应的控制量$u(t)$，以减小偏差。控制作用的强弱取决于比例系数K_p，K_p越大，控制作用越强，则过渡过程越快，控制过程的静态偏差也就越小,但是K_p越大，也越容易产生振荡，增加系统的超调量，系统的稳定性会变差。

优点：调整系统的开环比例系数，提高系统的稳态精度，加快响应速度。

缺点：仅用 P 控制器,过大的开环比例系数不仅会使系统的超调量增大，而且会使系统稳定度变小，甚至不稳定。

1.2 积分环节(I): $\frac{K_p}{T_i}\int_0^t e(t)dt$

只要偏差$e(t)$存在，积分控制作用就会就不断的增加(条件是控制器没有饱和)，偏差$e(t)$就不断减小，当偏差$e(t) = 0$时，积分控制作用才会停止。

优点：积分环节可以消除系统的稳态误差。

缺点：积分控制同时也会降低系统的响应速度，积分作用太强会增加系统的超调量，系统的稳定性会变差。

1.3 微分环节(D)：$K_p * T_d \frac{d}{dt} e(t)$

微分环节反映偏差信号的变化趋势，并能在偏差信号$e(t)$变得太大之前，在系统中引入一个有效的早期修正信号，从而加快系统的动作速度，减少调节时间。在微分控制中，控制器的输出与输入误差信号的微分（即误差的变化率）成正比关系。

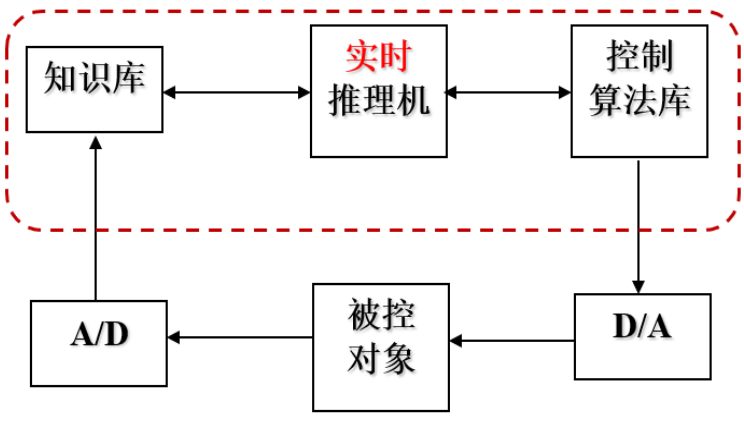
优点：微分作用的引入，将有助于减小超调量，克服振荡，使系统趋于稳定，它加快了系统的跟踪速度，减少调节时间。

缺点：微分作用对输入信号的噪声很敏感，对那些噪声较大的系统一般不用微分，或在微分之前先对输入信号进行滤波。

二、专家 PID 控制介绍

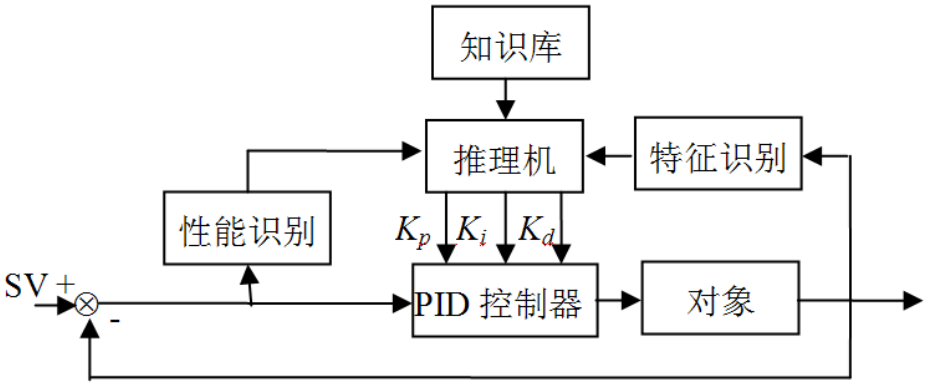
2.1 专家控制基本思想

专家控制试图在传统控制的基础上“加入”一个富有经验的控制工程师，实现控制的功能，它由知识库和推理机构构成主体框架，通过对控制领域知识 (先验经验、动态信息、目标等) 的获取与组织，按某种策略及时地选用恰当的规则进行推理输出，实现对实际对象的控制。



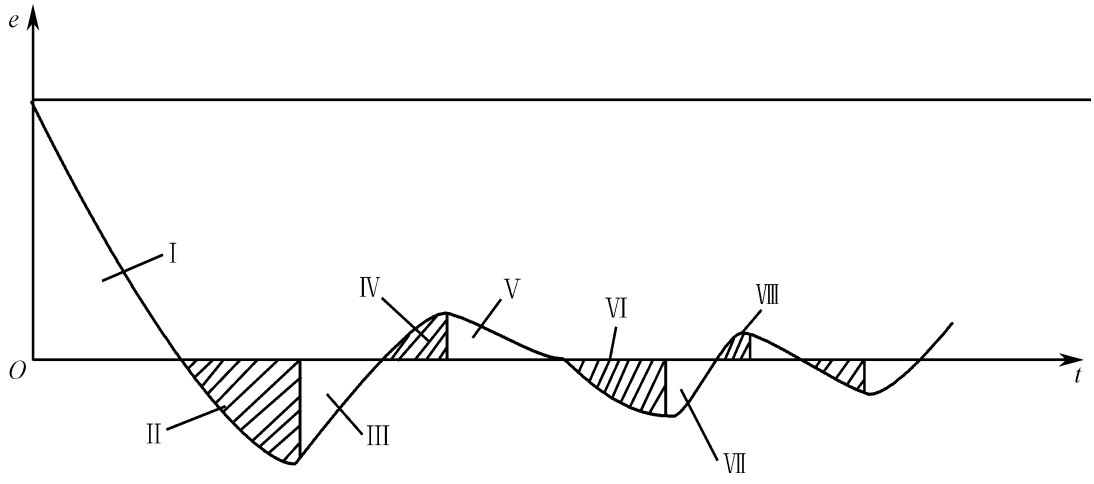
2.2 专家 PID 控制

PID 专家控制的实质：基于受控对象和控制规律的各种知识，无需知道被控对象的精确模型，利用专家经验来设计 PID 参数。专家 PID 控制是一种直接型专家控制器。



专家 PID 的核心：根据偏差和偏差的变化率，结合经验触发规则 1---规则 5，针对当下不同的偏差和偏差变化率选用不同的控制规则，以实现控制器的最佳输出。

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



增量式 PID 控制算法公式：

$$\Delta u_k = u_k - u_{k-1} = K_P[e_k - e_{k-1} + \frac{T}{T_1}e_k + \frac{T_D}{T}(e_k - 2e_{k-1} + e_{k-2})]$$

当前误差：$e(k)$

误差变化率：$\Delta e(k) = e(k) - e(k-1)$

设定偏差的一个极大值，记为M_{max}；

设定一个偏差较大的中间值，记为M_{mid}；

设定一个偏差的极小值，记为M_{min}。

规则 1：误差值本身特别大

当$|e(k)| > M_{max}$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输出，以达到迅速调整误差，使误差绝对值以最大速度减小。 这种情况下其实相当于实施开环控制，是一种对偏差出现极限情况的快速响应。即：

$$u(k) = \begin{cases} u_{max}\text{，偏差为正} \\ u_{min}\text{， 偏差为负} \end{cases}$$

规则 2：误差趋于增大

当$e(k)\Delta e(k) > 0$或$\Delta e(k) = 0$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。

（1）此时，**如果$|e(k)| \geq M_{mid}$**，说明误差也较大，可考虑由控制器实施较强的控制作用（加大增益快速控制），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：

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

(2)此时，**如果$|e(k)| < M_{mid}$**，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（按原系数做控制，不做调整），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：

$$u(k)=u(k-1)+k_{\mathrm{p}}[e(k)-e(k-1)]+k_{\mathrm{i}}e(k)+k_{\mathrm{d}}[e(k)-2e(k-1)+e(k-2)]$$

规则 3：误差趋于减小

当$e(k)\Delta e(k)<0$、$e(k)\Delta e(k-1)>0$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（不加 PID）。 即：

$$u(k)=u(k-1)$$

规则 4：峰值点

当$e(k)\Delta e(k)<0$、$e(k)\Delta e(k-1)<0$时，说明误差处于极值状态。

（1）此时，**如果$|e(k)|\geq M_{mid}$**，说明误差的绝对值较大，可考虑实施较强的控制作用。即：

$$u(k)=u(k-1)+k_1k_pe(k)$$

（2）此时，**如果$|e(k)|<M_{mid}$**，说明误差的绝对值较小，可考虑实施较弱的控制作用。即：

$$u(k)=u(k-1)+k_2k_pe(k)$$

规则 5：误差值本身特别小

当$|e(k)|>M_{min}$时，说明误差的绝对值很小，此时加入积分，减少稳态误差。即：

$$u(k)=u(k-1)+k_p[e(k)-e(k-1)]+k_ie(k)$$

三、仿真内容设置

控制一个三阶传递函数的阶跃响应：

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

采用专家 PID 设计控制器，在仿真过程中，采样时间取 0.001s，程序中的五条规则与控制算法的五种情况相对应。

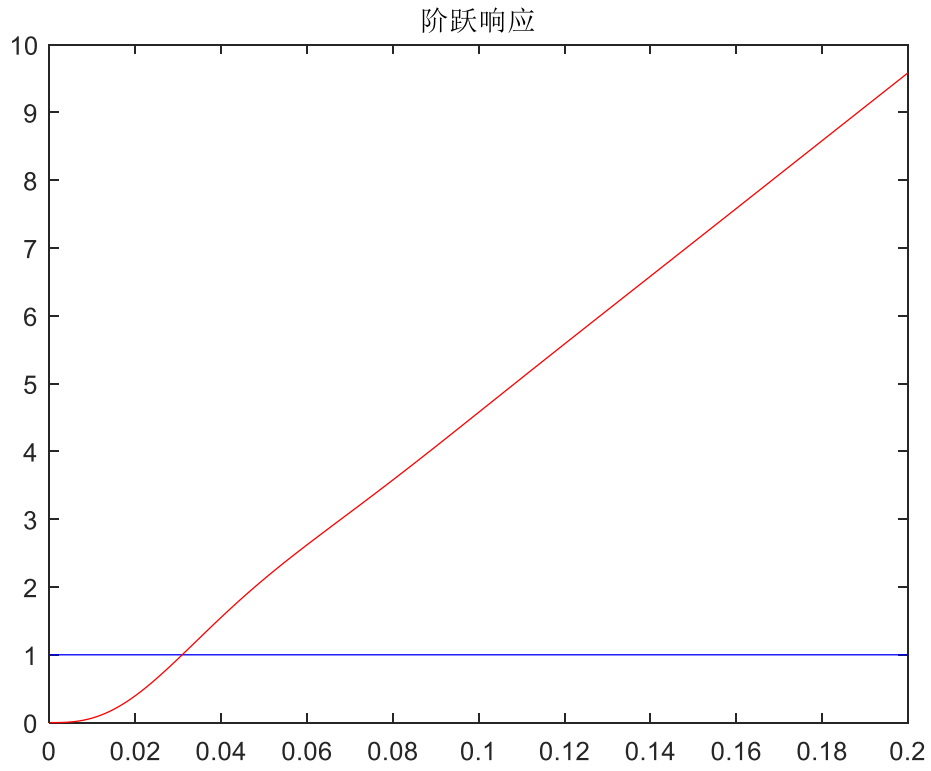
（1） 编程实现上述系统的单位阶跃响应，画输出波形；

（2） 编程实现传统 PID 对上述系统的矫正结果（设置 10 组参数）；

（3） 编程实现专家 PID 对上述系统的矫正结果。

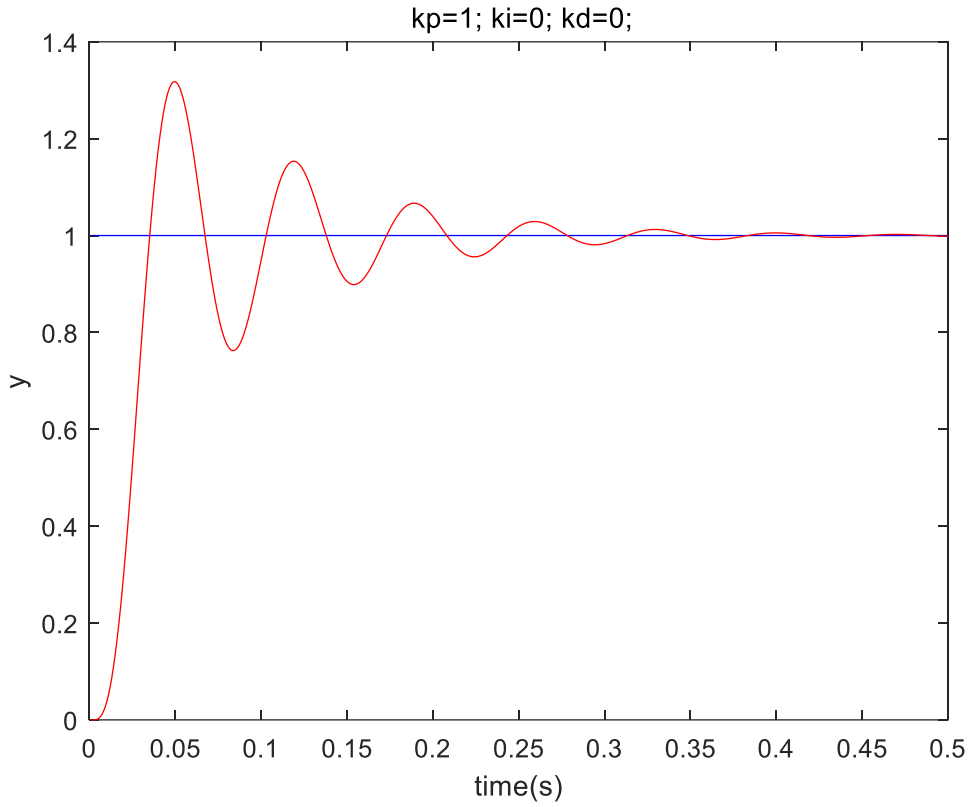
四、仿真结果展示

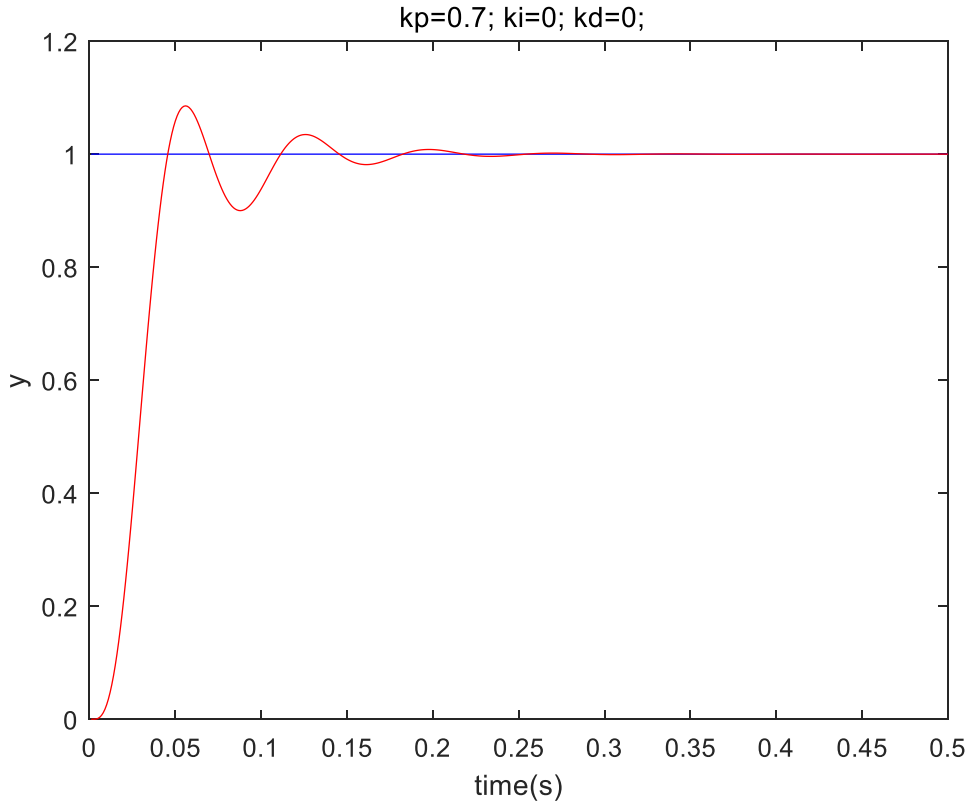
3.1 阶跃响应



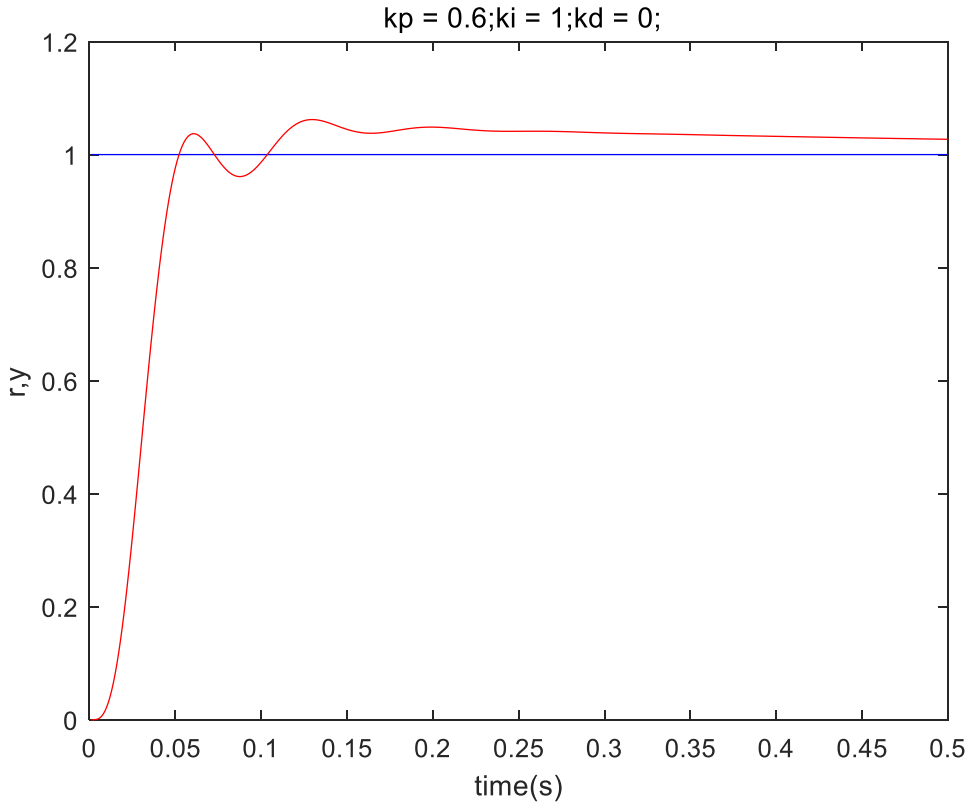
3.2 传统 PID

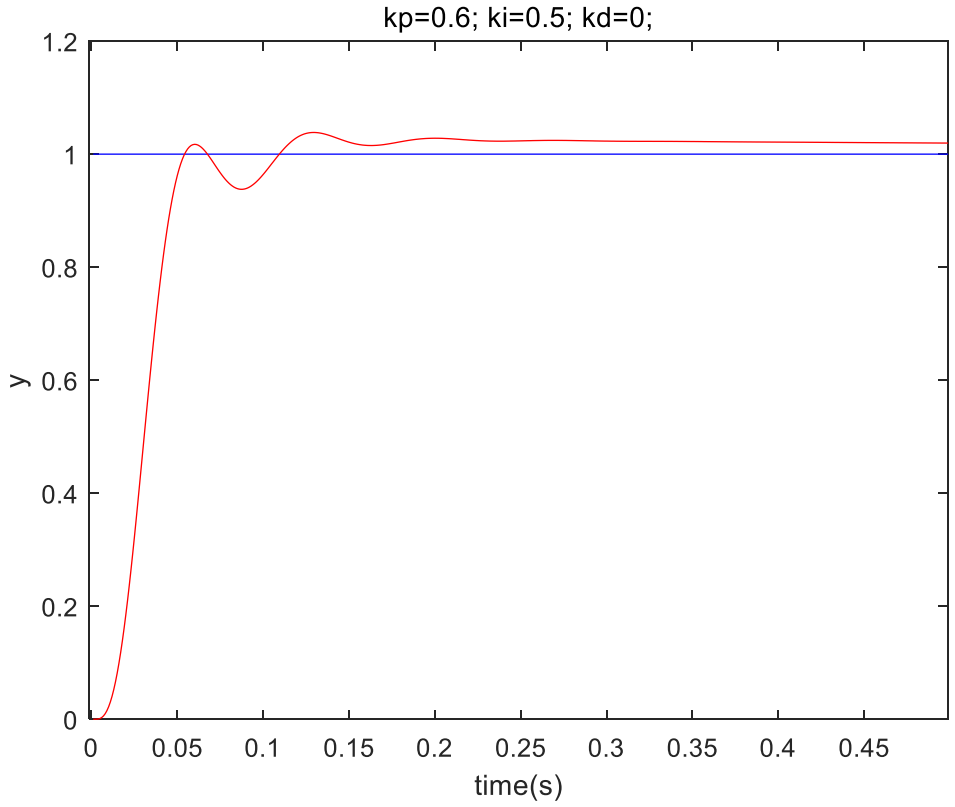
首先，将 I、D 设为 0，调节 P：

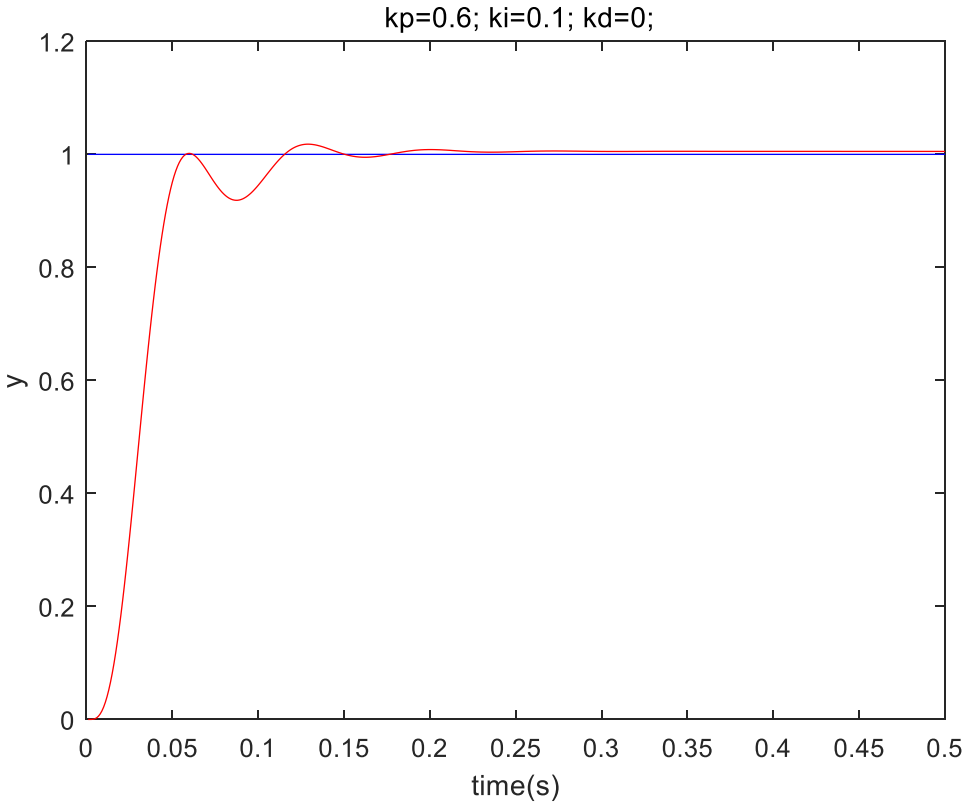


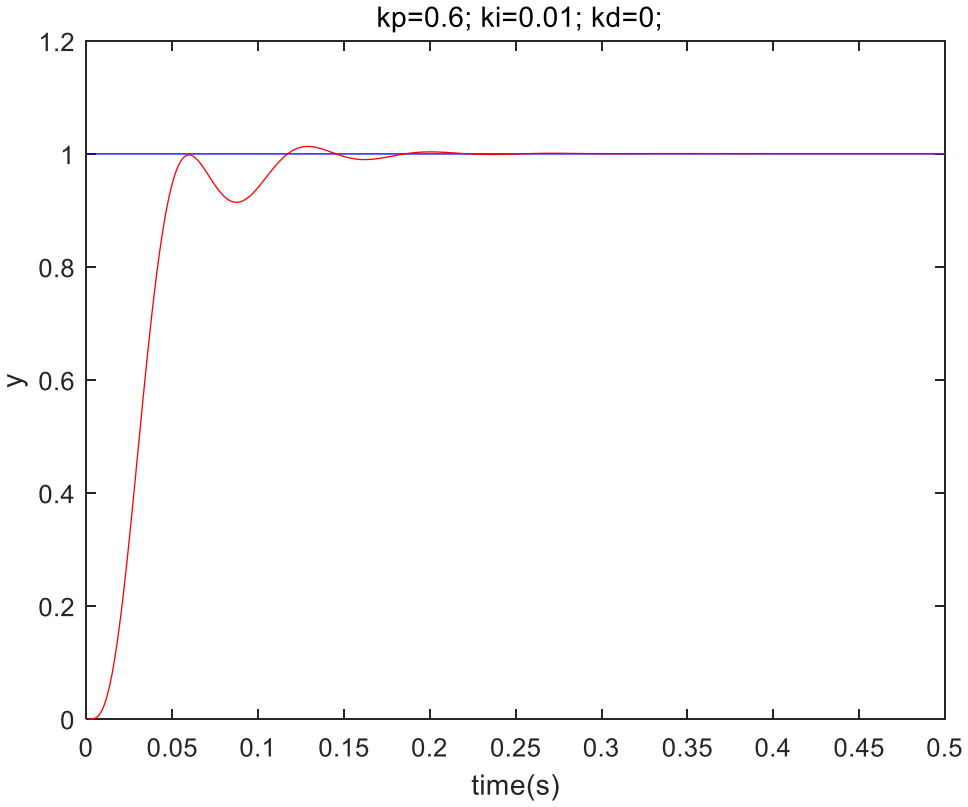


P 调节满意后，将 D 设为 0，调节 I：

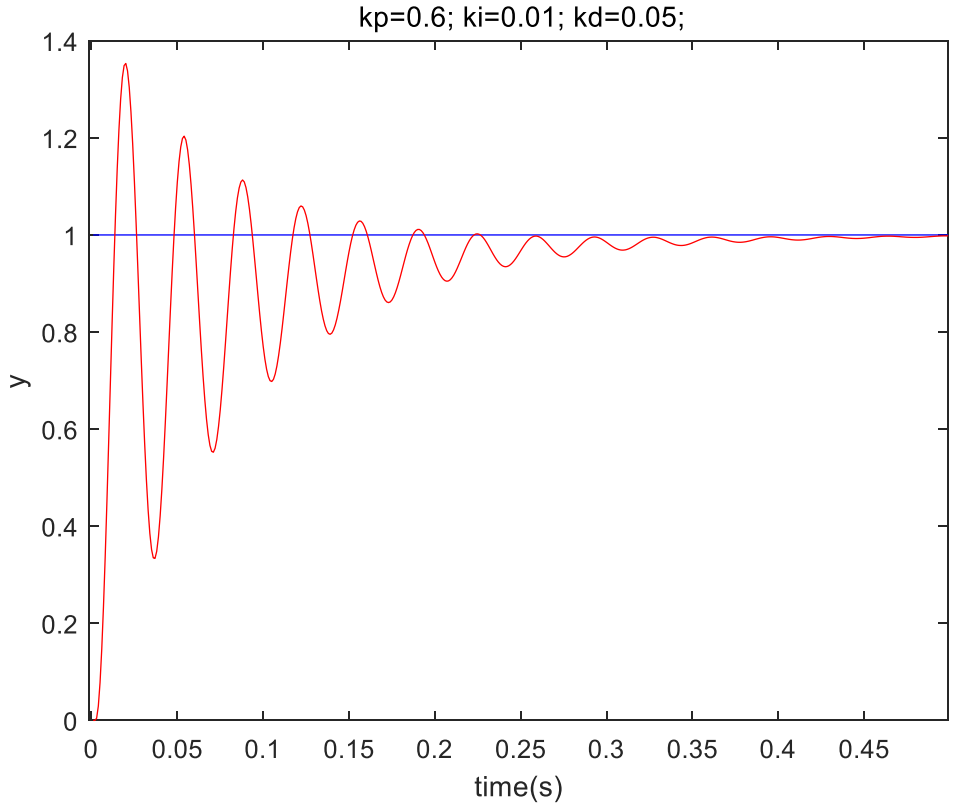


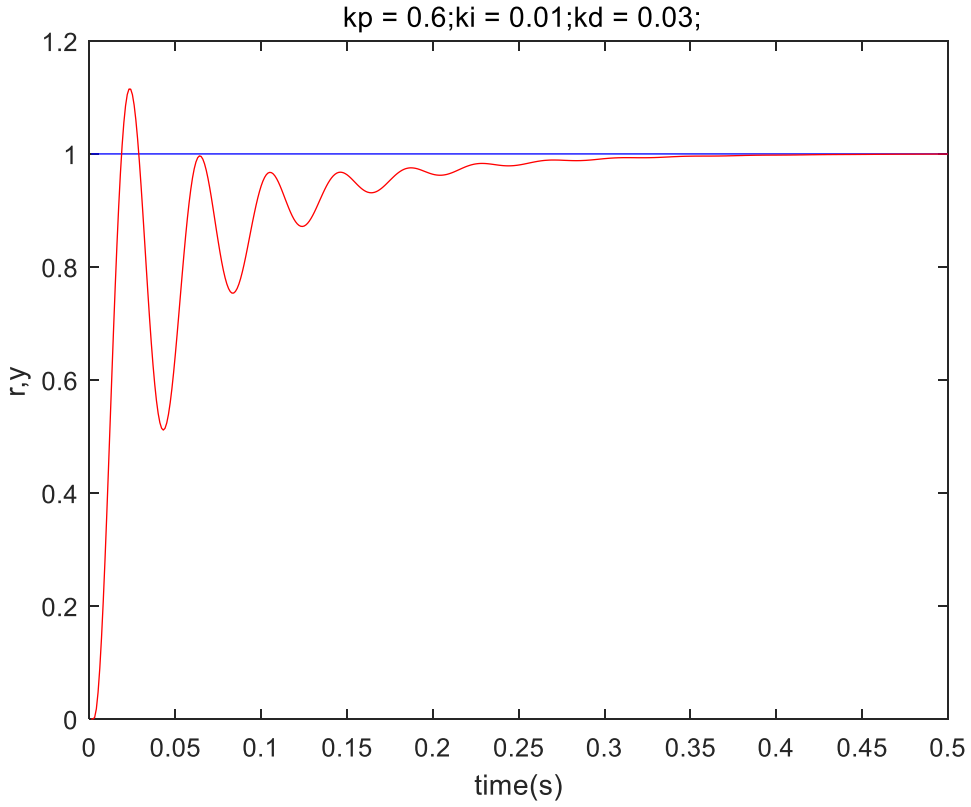


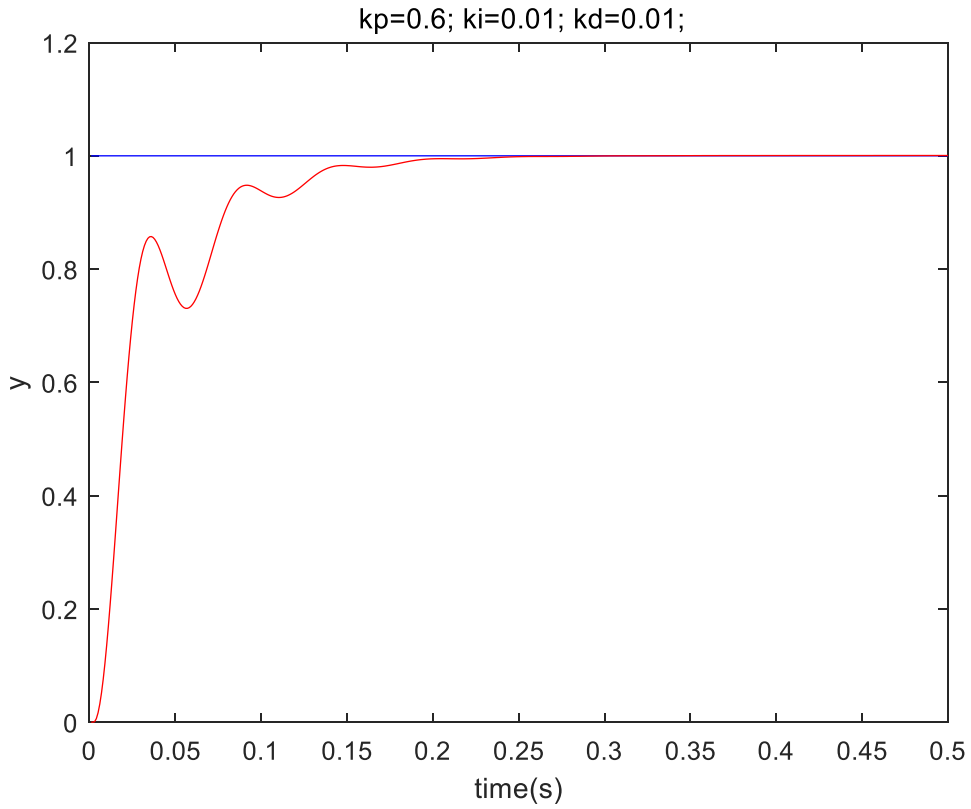

kp=0.6; ki=0.1; kd=0;
1.2
1
0.8
0.6
0.4
0.2
0
y
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
time(s)

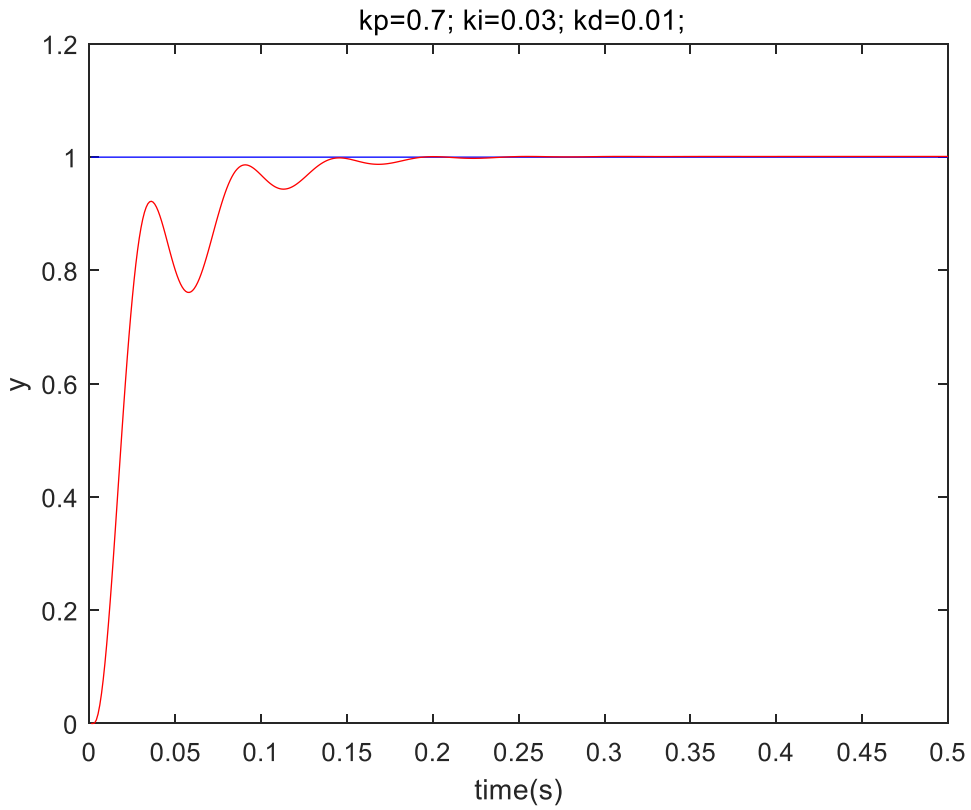

kp=0.6; ki=0.01; kd=0;
1.2
1
0.8
0.6
0.4
0.2
0
y
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
time(s)

P、I 调节满意后，调节 D：


kp=0.6; ki=0.01; kd=0.05;
y
time(s)
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0.05
0.1
0.15
0.25
0.3
0.35
0.45


kp = 0.6;ki = 0.01;kd = 0.03;
r,y
time(s)
0
0.2
0.4
0.6
0.8
1
1.2
0.05
0.1
0.15
0.25
0.3
0.35
0.45
0.5

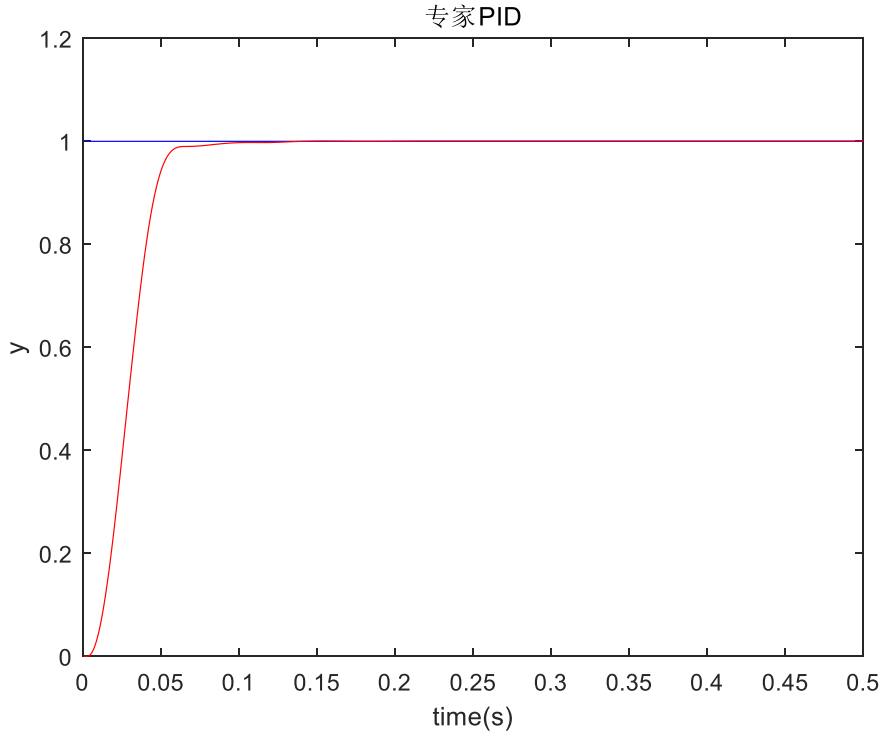


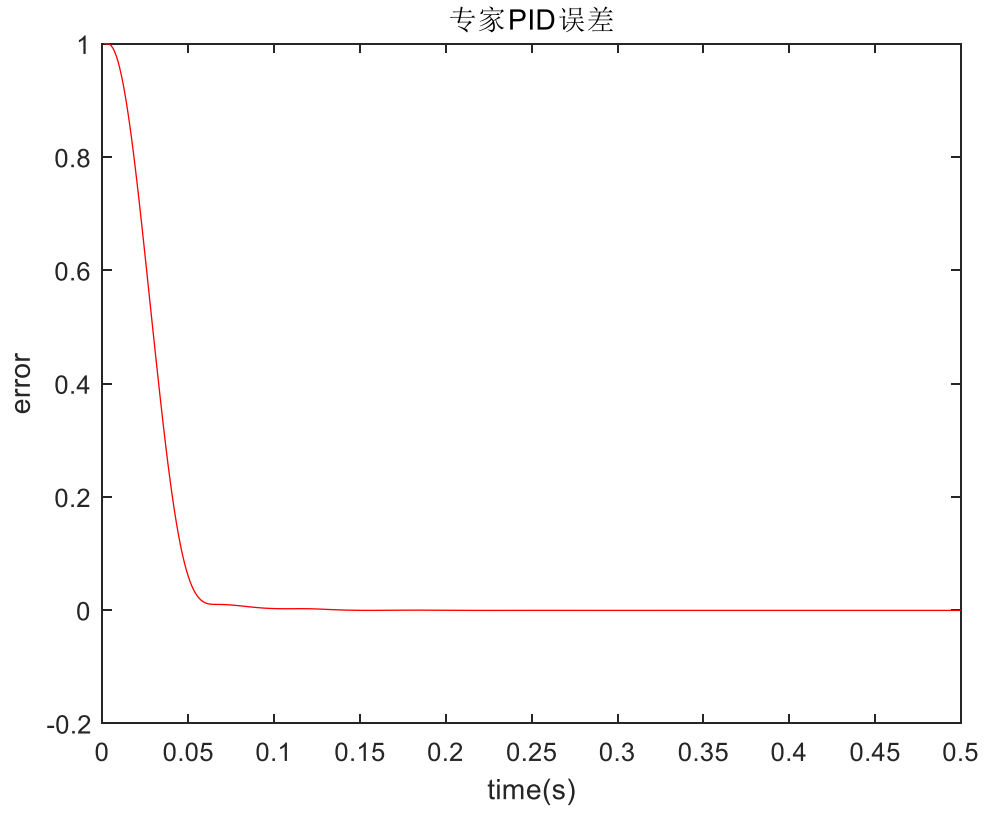


经过反复调节 PID 后，发现 P=0.7，I=0.03，D=0.01 时效果最好。

3.3 专家 PID

调节 P=0.7, I=0.03, D=0.01, 加入专家控制的 5 条规则, 显示输出和误差。





五、仿真分析与心得体会

由于系统的原因，比例调节部分对系统影响最大，积分微分部分只在很小程度上起辅助作用，调节过程中作用太强会导致系统出现明显波动。参数调节具有一定的技巧性，需要多次逐一参数调节，不断试凑。引入专家控制规则后，系统可在更短的时间内稳定到期望值，达到理想效果。

通过这次专家 PID 控制的仿真，让我更加清楚了 PID 经典控制算法和专家控制算法，对调节 PID 参数的试凑法有了初步掌握，并进一步了解了 Matlab 在信号、控制方面的应用。

六、Matlab 代码

```
%智能控制:专家 PID
clc                 %清除命令行窗口的内容
clear               %清除工作空间的所有变量
close               %关闭当前的 figure 窗口

ts = 0.001;         %采样时间

sys = tf(5.235e005,[1,87.35,1.047e004,0]); %传递函数
dsys = c2d(sys,ts,'z');                    %将连续的时间模型转换成离散的时间模型,z 变换
[num,den] = tfdata(dsys,'v');              %将传递函数的分子分母分别放入 num,den 中

[y0,t,x] = step(dsys,0.2);                 %计算系统的阶跃响应(0-0.5s),返回输入响应 y,模拟时间向量 t,状态轨迹 x

%绘制该传递函数的阶跃响应
figure('name','阶跃响应');
title('阶跃响应');
plot([0 0.2],[1 1],'b',t,y0,'r');

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;

%手动设定 PID
kp=0.6; ki=0.01; kd=0.01;

error_1 = 0;
for k = 1:1:500
   time(k) = k*ts;
   r(k)= 1.0;      %期望值
   u(k)= kp*x(1)+kd*x(2)+ki*x(3);          %PID 控制输出

   %专家 PID 控制规则
   if abs(x(1))>0.8                        %规则 1：误差值本身特别大
       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)               %规则 2：误差趋于增大
       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)%规则 3：误差趋于减小
       u(k)=u(k);
   end

   if x(1)*x(2)<0&&x(2)*x2_1<0              %规则 4：峰值点
       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                         %规则 5：误差值本身特别小
        u(k)=0.5*x(1)+0.010*x(3);
    end

    %计算输出和误差
    y(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) = r(k)-y(k);

    %参数更新
    u_3 =u_2;
    u_2 =u_1;
    u_1 =u(k);
    y_3 =y_2;
    y_2 =y_1;
    y_1 =y(k);

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

    error_1 = error(k);
end
%绘制输出曲线
figure(2);
plot(time,r,'b',time,y,'r');
xlabel('time(s)');ylabel('y');
%绘制误差曲线
figure(4);
plot(time,r-y,'r');
xlabel('time(s)');ylabel('error');
```