```
%% test8_3_1
clear;clc
ng0 = [1];                                          %原系统开环传递函数分子
dg0 = conv([1,0.2],conv([1,1],[1,5]));              %原系统开环传递函数分母
g0 = tf(ng0,dg0)                                    %原系统开环传递函数
t = [0:0.01:100];                                   %时域范围
y = step(g0,t);                                     %系统的单位阶跃响应曲线
dy = diff(y);                                       %利用差分求得曲线各点的斜率
[k,x] = max(dy);
k = k * 100;                                        %找出斜率最大值及其位置
t_max = x * 0.01;
y_max = y(x);                                       %斜率最大值点的横纵坐标
b = y_max - k * t_max                               %计算截距
K = max(y);                                         %稳态值
tao = abs(b) / k;                                   %计算滞后时间tao
T = K / k;                                          %计算时间常数T
Kp = 0.9 * T / tao;
Ti = tao / 0.3;
PI = Kp * (1 + tf(1,[Ti,0]))                        %得到PI控制器
Kp = 1.2 * T / tao;
Ti = 2 * tao;
Td = 0.5 * tao;
PID = Kp * (1 + tf(1,[Ti,0]) + tf([Td,0],[1]))      %得到PID控制器

%% test8_3_2
%加入控制器后的单位阶跃响应
sysPI = feedback(PI * g0,1);
sysPID = feedback(PID * g0,1);
step(sysPI,t)
title('加入PI控制器后的单位阶跃响应')
figure();
step(sysPID,t)
title('加入PID控制器的单位阶跃响应')

%对PI控制器的参数进行调整
Kp = 0.65 * 0.9 * T / tao;
Ti = 3 * tao / 0.3;
PI = Kp * (1 + tf(1,[Ti,0]))                        %参数改变后的PI控制器
sysPI = feedback(PI * g0,1);
figure
step(sysPI,t)
title('加入改变参数的PI控制器的单位阶跃响应')
ss1=stepinfo(sysPI);     %改变后PI控制器的单位阶跃响应
fprintf('PI控制器调节参数后的超调量：%f\n',ss1.Overshoot)
```

```
%对PID控制器的参数进行调整
Kp = 1.25 * T / tao;
Ti = 2.55 * 2 * tao;
Td = 0.5 * tao;
PID = Kp * (1 + tf(1,[Ti,0]) + tf([Td,0],[1]))    %参数改变后的PI控制器
sysPID = feedback(PID * g0,1);
figure
step(sysPID,t)
title('加入改变参数的PID控制器的单位阶跃响应')
ss2=stepinfo(sysPID);    %改变后PID控制器的单位阶跃响应
fprintf('PID控制器调节参数后的超调量：%f\n',ss2.Overshoot)
fprintf('\n')

%% test8_3_3
t = 0:0.01:50;
%PI
g1 = sysPI * tf([1],[1,0,0]);                    %加入斜坡函数
y1 = impulse(g1,t);                              %单位斜坡响应
errorPI = t - y1';                               %计算误差
figure;
plot(t,errorPI)                                  %画出图像
title('PI的误差')
xlabel('t')
ylabel('errorPI')

n = length(t);
ess1 = errorPI(n);                               %得到稳态值
maxE1 = max(errorPI);                            %获得峰值
overshoot1 = 100 * (maxE1 - ess1) / ess1;        %计算超调量
for i = n : -1 : 1
    if errorPI(i) < 0.98 * ess1 || errorPI(i) > 1.02 * ess1
    ts1 = t(i);                                  %得到调节时间，误差带为2%
    break
    end
end
disp('加入PI调节器：')
fprintf('稳态误差：%f\n',ess1)
fprintf('超调量：%f\n',overshoot1)
fprintf('调节时间：%f\n',ts1)
fprintf('\n')

%PID
g2 = sysPID * tf([1],[1,0,0]);                   %加入斜坡函数
```

```
y2 = impulse(g2,t);                                    %单位斜坡响应
errorPID = t - y2';                                    %计算误差
figure;
plot(t,errorPID)                                       %画出图像
xlabel('t')
ylabel('errorPID')
title('PID的误差')

ess2 = errorPID(n);                                    %得到稳态值
maxE2 = max(errorPID);                                 %获得峰值
overshoot2 = 100 * (maxE2 - ess2) / ess2;              %计算超调量
for i = n : -1 : 1
    if errorPID(i) < 0.98 * ess2 || errorPID(i) > 1.02 * ess2
    ts2 = t(i);                                        %得到调节时间，误差带为2%
    break
    end
end
disp('加入PID调节器：')
fprintf('稳态误差：%f\n',ess2)
fprintf('超调量：%f\n',overshoot2)
fprintf('调节时间：%f\n',ts2)
```