实验三 控制理论方法分析与建模

1. 实验目的

**1.** 熟悉线性系统建模

**2.** 掌握控制理论方法分析

**3.** 学习基于Simulink的非线性系统仿真

**4.** 了解传统控制理论方法分析与建模

1. 实验平台

Windows10、MATLAB 2022b、Office2010软件

三、演练

**1.**时域响应分析

**1.1** 系统的传递函数为,试求其闭环传递函数，并绘制出输出量阶跃响应曲线和脉冲响应曲线。

**step()**函数、**impulse()**函数分别是求阶跃响应和脉冲响应的函数。

求解过程：在MATLAB 命令行窗口下键入“edit”或选择“新建脚本”，进入MATLAB 的编辑器，编辑M文件“sysresponse.m”

% 关闭所有图形窗口并清除内存变量

close all;

clear;

clc;

% 系统开环传递函数初始化

numo = [0 0 0 0 200];

deno = [1 10 100 400 484];

% 求解系统的闭环传递函数

numc = numo;

n = length(deno);

denc = zeros(1,n);

denc = numo + deno;

% 结果显示

disp('System Closed Loop Transfer Function is:')

numc

denc

% 系统仿真数据初始化

t = 0:0.05:3;

% 系统输出量的阶跃响应

y = step(numc,denc,t);

% 系统输出量的脉冲响应

yy = impulse(numc,denc,t);

% 输出量阶跃响应曲线绘制

subplot(2,1,1); % 第一个子图，阶跃响应

plot(t,y);

title('System Step Response');

xlabel('Time-sec');

ylabel('Response-value');

grid;

% 输出量脉冲响应曲线绘制

subplot(2,1,2); % 第二个子图，脉冲响应

plot(t,yy);

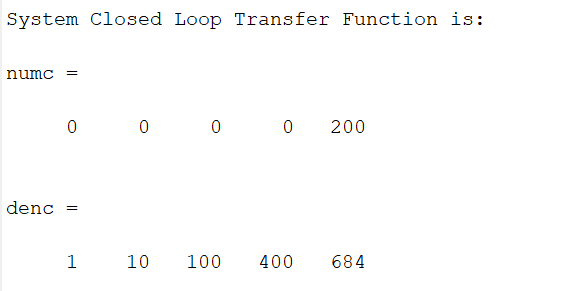
title('System Impulse Response');

xlabel('Time-sec');

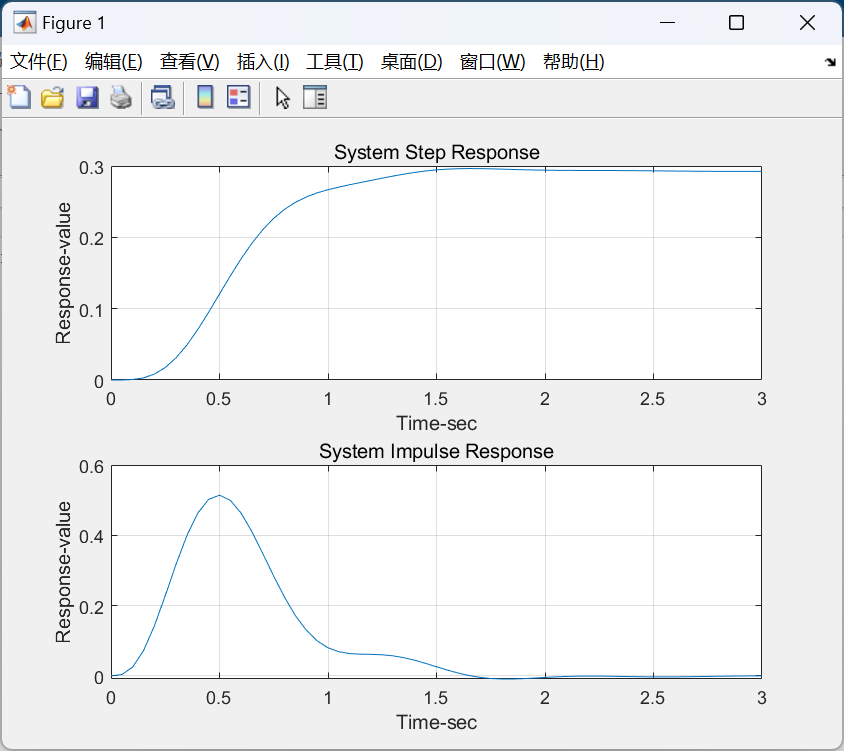
ylabel('Response-value');

grid;

点击“保存”按钮，保存文件名为“sysresponse.m”。选择“编辑器”菜单的“运行按选项或直接在MATLAB的命令行窗口下键入文件名sysresponse，在MATLAB的命令行窗口下查看运行的结果。



图一.闭环传递函数结果



图二.阶跃响应与脉冲响应图

**1.2** 求传递函数为的控制系统在输入函数u=sint的时间响应。

Matlab中提供了一个**lsim()**函数可以求解和绘制任意输入函数的系统时间响应，其具体用法可以在MATLAB的命令控制行中键入查看。

求解过程：在MATLAB 命令行窗口下键入“edit”或选择“新建脚本”，进入MATLAB 的编辑器，编辑M文件“sinresponse.m”

% 关闭所有图形窗口并清除内存变量

close all;

clear;

clc;

num = [0 0 1 1];

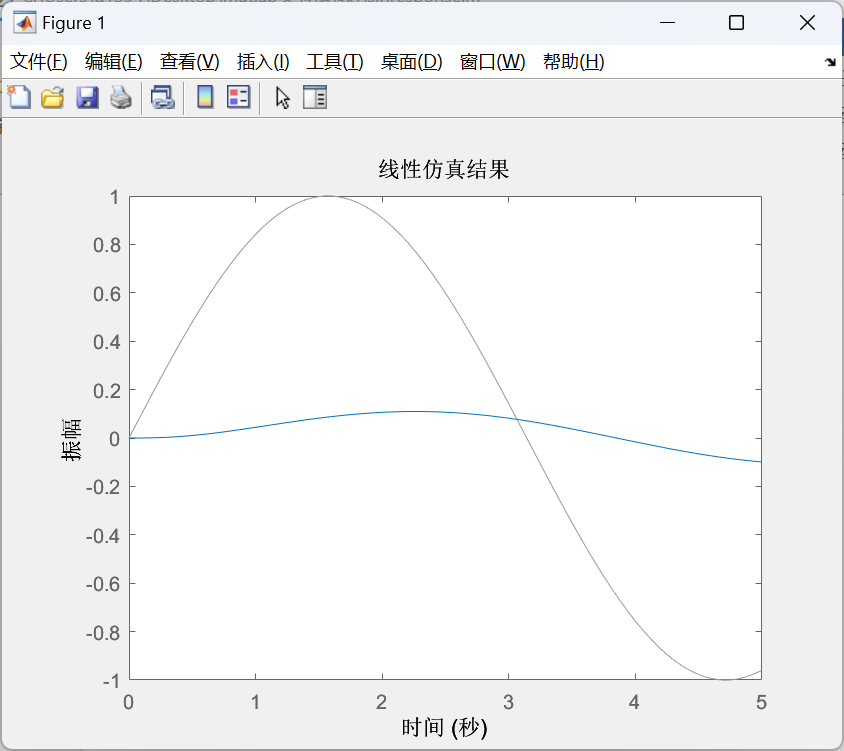
den = [1 7 14 8];

t = 0:0.05:5;

u = sin(t);

lsim(num,den,u,t)

点击“保存”按钮，保存文件名为“sinresponse.m”。选择“编辑器”菜单的“运行按选项或直接在MATLAB的命令行窗口下键入文件名sinresponse，在MATLAB的命令行窗口下查看运行的结果。



图三. 正弦激励响应图

其中lsim 函数的输出包含两个部分：

灰色线代表的是系统的总响应，蓝色线代表的是系统的零状态响应

**1.3** 求解离散控制系统的单位阶跃响应和单位脉冲响应。

对于离散时间的控制系统(或称采样控制系统),MATLAB同样提供了相应的**dstep()**、**dimpulse()**、**dlsim()**函数求解其单位阶跃响应、单位脉冲响应和任意函数的激励响应。

求解过程：在MATLAB 命令行窗口下键入“edit”或选择“新建脚本”，进入MATLAB 的编辑器，编辑M文件“dsysresponse.m”

% 关闭所有图形窗口并清除内存变量

close all;

clear;

clc;

num = [1.6 -1 0];

den = [1 -0.8 0.5];

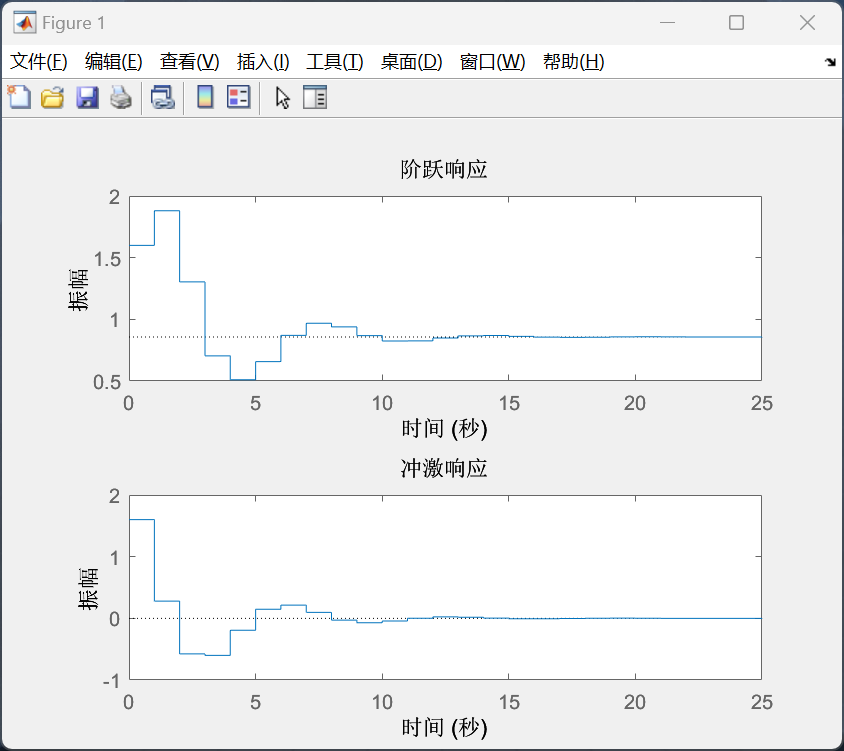
subplot(2,1,1);

dstep(num,den)

subplot(2,1,2);

dimpulse(num,den)

点击“保存”按钮，保存文件名为“dsysresponse.m”。选择“编辑器”菜单的“运行按选项或直接在MATLAB的命令行窗口下键入文件名dsysresponse，在MATLAB的命令行窗口下查看运行的结果。



图四.离散控制系统响应图

**2.** 频率响应分析

在MATLAB环境下求解系统的幅频特性和相频特性时，经常要用到**polyval()**函数。其基本调用个数为：

Y=polyval(P,X)

其中P是多项式系数，X是自变量，Y是求得的返回值。

**2.1** 某控制系统的频率特性函数为,试求解其幅频特性曲线和相频特性曲线，并与系统的阶跃响应曲线对比。

求解过程：在MATLAB 命令行窗口下键入“edit”或选择“新建脚本”，进入MATLAB 的编辑器，编辑M文件“freqresponse.m”：

% 关闭所有窗口并清除所有内存变量

clear all

close all

% 系统的频率特性函数描述

num = [0 0 0 10];

den = [1 5 10 12];

% 仿真时间和频率初始化

t = 0:0.05:4;

wt = 0:0.02:4;

% 系统的阶跃响应

y = step(num,den,t);

% 系统的频率特性

G = polyval(num,sqrt(-1)\*wt)./polyval(den,sqrt(-1)\*wt);

% 幅频特性

mag = abs(G);

% 相频特性

theta = angle(G);

% 绘图

subplot(2,1,1);

plot(t,y);

title('System Step Response');

xlabel('Time-sec');

ylabel('Response-value');

grid;

subplot(2,2,3);

plot(wt,mag);

title('System Frequency Response-Amplitude');

xlabel('Frequency-rad');

ylabel('Amplitude-value');

grid;

subplot(2,2,4);

plot(wt,theta);

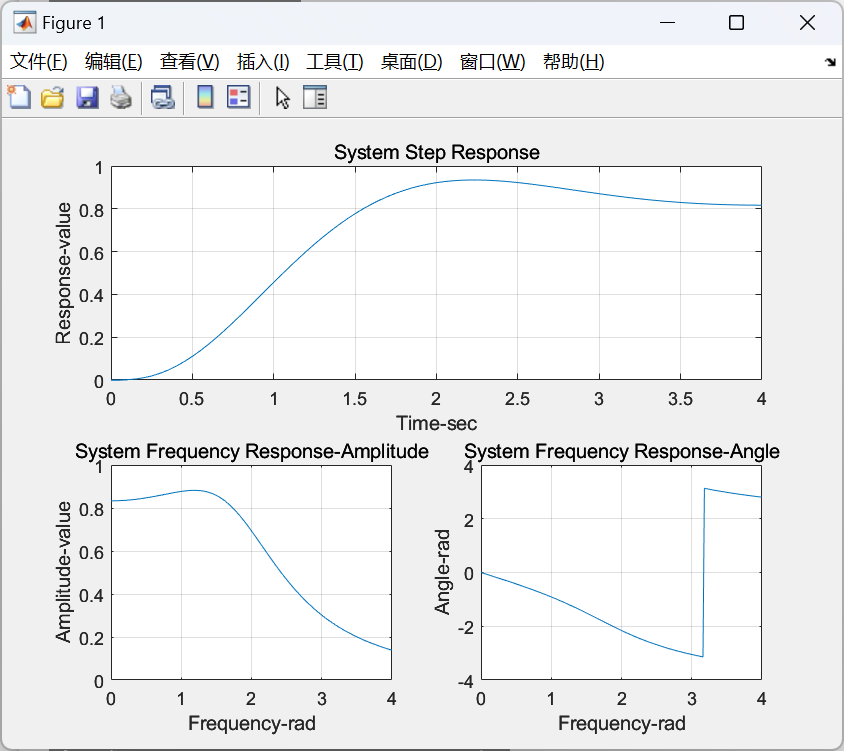
title('System Frequency Response-Angle');

xlabel('Frequency-rad');

ylabel('Angle-rad');

grid;

点击“保存”按钮，保存文件名为“freqresponse.m”。选择“编辑器”菜单的“运行按选项或直接在MATLAB的命令行窗口下键入文件名freqresponse，在MATLAB的命令行窗口下查看运行的结果。



图五.幅频特性曲线、相频特性曲线、系统的阶跃响应图

图四的上半部分是系统的阶跃响应曲线，横坐标是时间，纵坐标是阶跃响应的值；下半部分是系统的频率响应曲线，横坐标都是频率；其中左边是相频特性曲线，右边是幅频特性曲线。

**2.2** Bode图绘制

某控制系统的开环传递函数为。试绘制其Bode图和增益裕量及相角裕量，并于系统的闭环阶跃响应曲线比较。

MATLAB提供了bode()函数可以求解和绘制系统的Bode图，以及提供了一条函数margin()可以求解系统的增益裕量和相角裕量，其具体用法可以在MATLAB的命令控制行内键入“help bode” 和“help margin”查看。

求解过程：在MATLAB 命令行窗口下键入“edit”或选择“新建脚本”，进入MATLAB 的编辑器，编辑M文件“sysbode.m” (主要不要直接取名为bode.m，否则会冲掉MATLAB系统原有的bode.m文件)：

% 关闭所有图形窗口并清除内存变量

close all;

clear all;

% 系统开环传递函数描述

numo = [0 0 0 0 0.86];

den1 = [1 0];

den2 = [0.36 1];

den3 = [0.3906 0.75 1];

deno = conv(den1,conv(den2,den3));

% 系统闭环传递函数描述

numc = numo;

denc = deno + numo;

% 系统仿真时间及频率初始化

t = 0:0.01:20;

wt = logspace(-1,1);

% 系统频率响应数据

[mag,phase] = bode(numo,deno,wt);

% 系统频率增益裕量及相角裕量

[Gm,Pm,Wcg,Wcm] = margin(numo,deno);

% 显示结果

disp('System Gain Margin and its associated frequency are:');

Gm

Wcg

disp('System Phase Margin and its associated frequency are:');

Pm

Wcm

% 系统的闭环阶跃响应

y = step(numc,denc,t);

% 图形绘制

subplot(2,1,1);

% 对数幅频特性函数

amp = 20\*log10(mag);

semilogx(wt,amp)

title('System Bode Frequency Response-Amplitude');

xlabel('Frequency-rad');

ylabel('Amplitude-dB');

grid;

subplot(2,1,2);

semilogx(wt,phase)

title('System Bode Frequency Response-Phase');

xlabel('Frequency-rad');

ylabel('Phase-degree');

grid;

figure; % 创建新的窗口

plot(t,y);

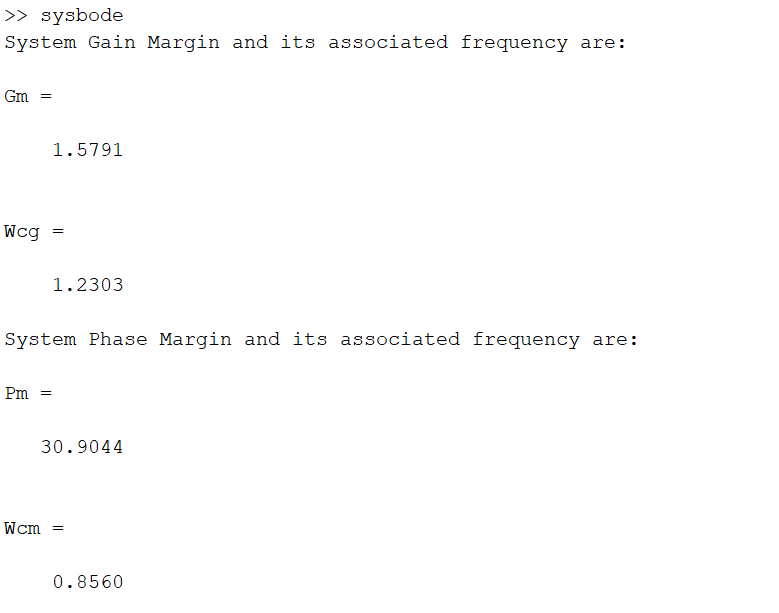
title('System Time Response');

xlabel('Time-sec');

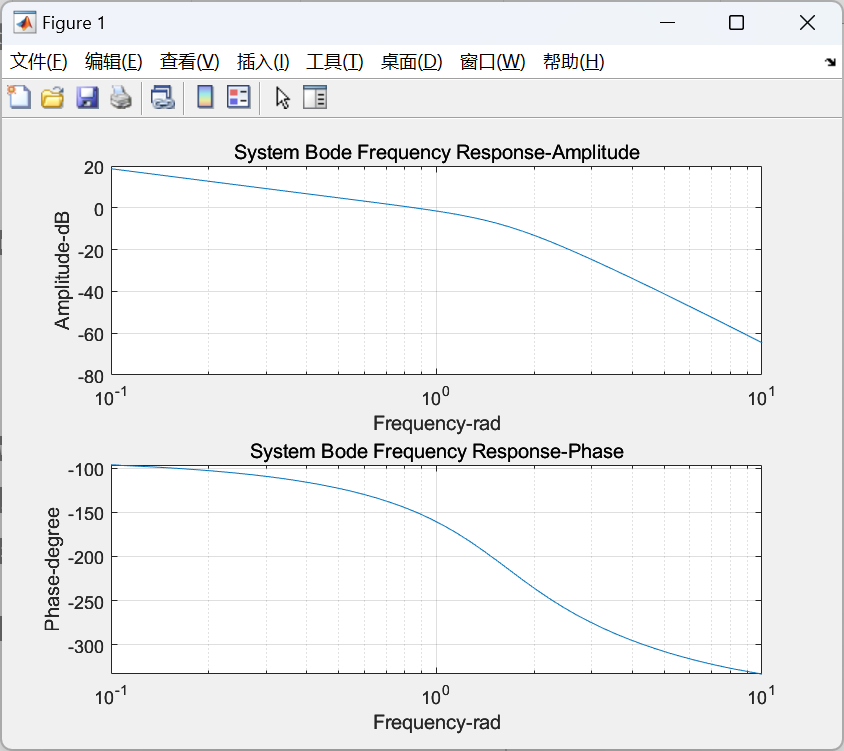
ylabel('Response-value');

grid;

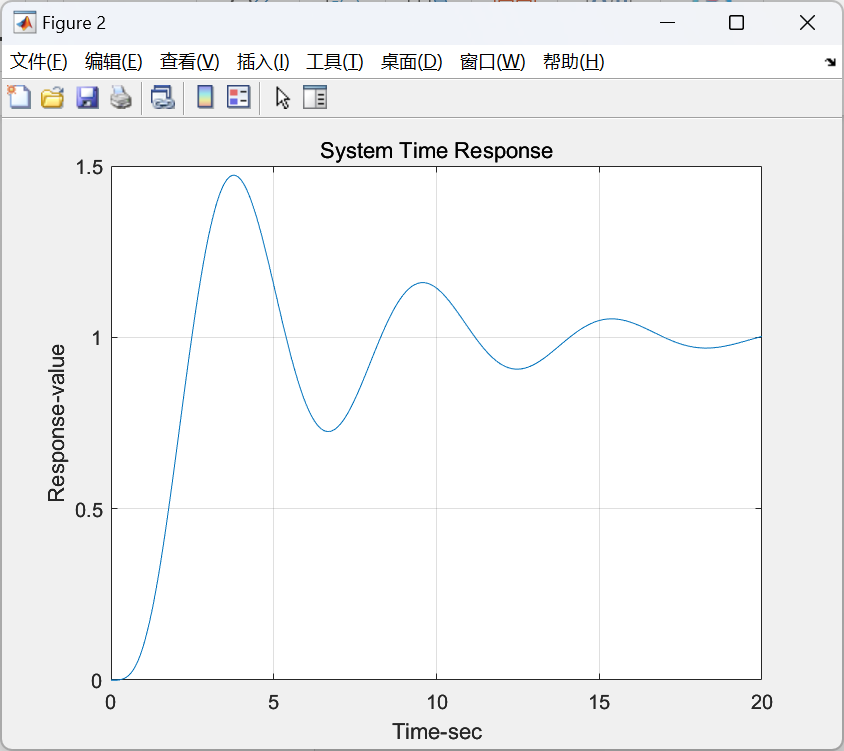
点击“保存”按钮，保存文件名为“sysbode.m”。选择“编辑器”菜单的“运行按选项或直接在MATLAB的命令行窗口下键入文件名sysbode，在MATLAB的命令行窗口下查看运行的结果。



图六.系统的增益裕量和相角裕量



图七.四阶系统Bode图



图八.四阶系统闭环阶跃响应曲线

**2.3** Nyquist图绘制

控制系统的开环传递函数为。试绘制当K = 2和K = 20时系统的Nyquist图。

求解过程：在MATLAB 命令行窗口下键入“edit”或选择“新建脚本”，进入MATLAB 的编辑器，编辑M文件“sysnyquist.m” (主要不要直接取名为nyquist.m，否则会冲掉MATLAB系统原有的nyquist.m文件)：

% 关闭所有图像窗口并清除内存变量

close all;

clear all;

% K = 2时系统的开环频率特性函数

numo1 = [0 0 0 2];

den1 = [1 0];

den2 = [1 1];

den3 = [0.1 1];

deno1 = conv(den1,conv(den2,den3));

% K = 20时系统的开环频率特性函数

numo2 = [0 0 0 20];

deno2 = deno1;

% 系统的闭环频率特性函数

numc1 = numo1;

denc1 = deno1 + numo1;

numc2 = numo2;

denc2 = deno2 + numo2;

% 仿真时间及角频率初始化

t = 0:0.1:15;

wt = -10:1:10;

% K = 2时系统的开环Nyquist曲线

subplot(1,2,1);

nyquist(numo1,deno1,wt);

title('System Nyquist Charts with K=2');

% K = 20时系统的开环Nyquist曲线

subplot(1,2,2);

nyquist(numo2,deno2,wt);

title('System Nyquist Charts with K=20');

% 系统的闭环阶跃响应

y1 = step(numc1,denc1,t);

y2 = step(numc2,denc2,t);

% 绘图

figure;

subplot(2,1,1);

plot(t,y1);

title('System Time Response With K=2');

xlabel('Time-sec');

ylabel('Response-value');

grid;

subplot(2,1,2);

plot(t,y2);

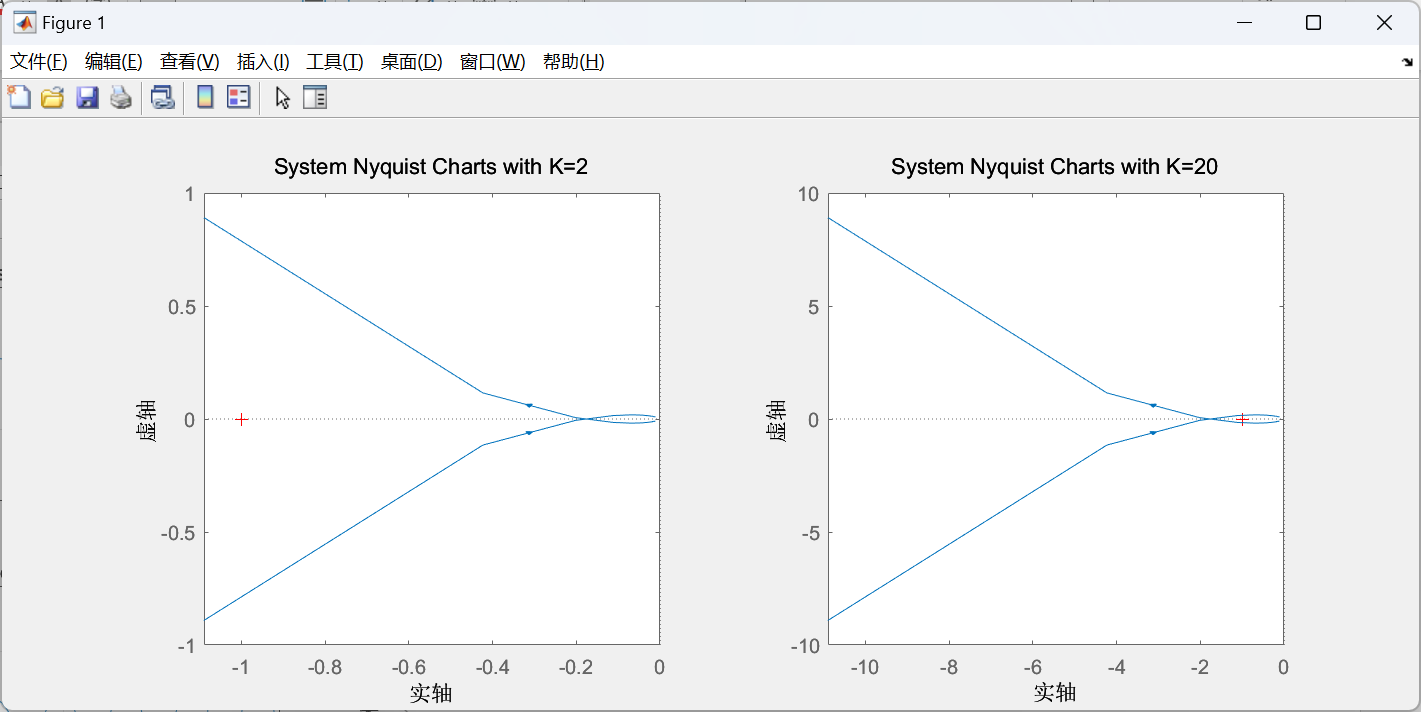
title('System Time Response With K=20');

xlabel('Time-sec');

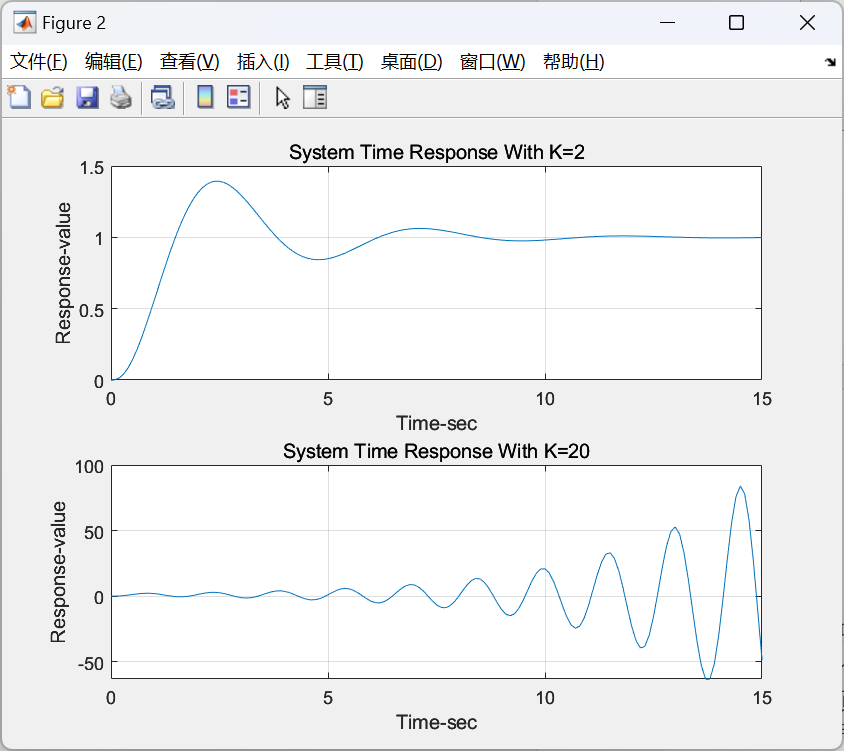
ylabel('Response-value');

grid;

点击“保存”按钮，保存文件名为“sysnyquist.m”。选择“编辑器”菜单的“运行按选项或直接在MATLAB的命令行窗口下键入文件名sysnyquist，在MATLAB的命令行窗口下查看运行的结果。



图九.K=2(左)及K=20(右)系统的Nyquist曲线



图十.K=2 (上)和K=20(下)时系统的闭环阶跃响应曲线

**2.4** 根轨迹的绘制

某开环系统的开环传递函数为,试绘制系统的闭环根轨迹。寻找系统临界稳定时的增益K，并绘制此时的系统阶跃响应作为验证。

求解过程：在MATLAB 命令行窗口下键入“edit”或选择“新建脚本”，进入MATLAB 的编辑器，编辑M文件“sysrlocus.m” (主要不要直接取名为rlocus.m，否则会冲掉MATLAB系统原有的rlocus.m文件)：

% 关闭所有窗口并清除所有内存变量

clear all;

close all;

clc;

% 系统开环传递函数

num = 1;

den1 = [1 1];

den2 = [1 3-sqrt(-1)];

den3 = [1 3+sqrt(-1)];

den = conv(den1,conv(den2,den3));

% 增益相邻初始化

K = 0:1:200;

% 求解根轨迹数据

r = rlocus(num,den,K);

% 寻找临界稳定点

[m,n] = size(r);

for i = 1:m

if real(r(i,2)) > 0

break

end

end

% 显示结果

disp('System critical stable-point is:')

K(i)

% 根轨迹绘图

figure;

plot(r,'.');

title('System Roots Locus');

xlabel('Re-axis');

ylabel('Im-axis');

grid;

% 临界稳定点系统闭环传递函数

numc = zeros(1,length(den));

numc(length(den)) = K(i);

denc = den + numc;

% 临界稳定时系统的阶跃响应曲线

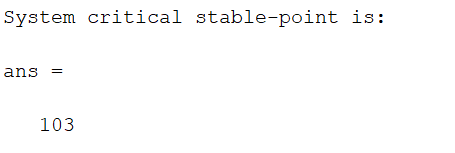
figure;

t = 0:0.01:15;

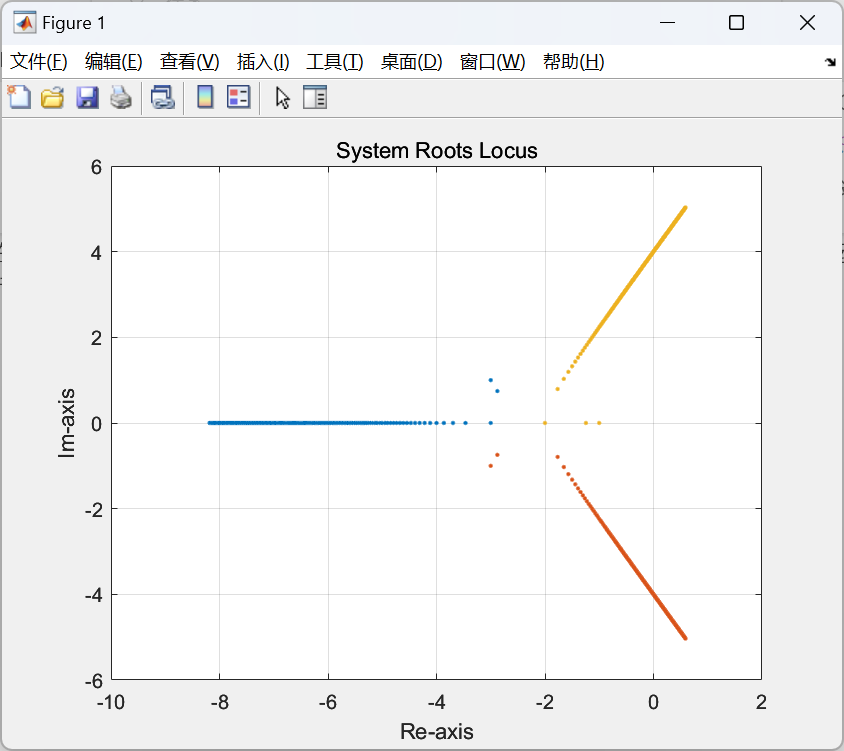
step(numc,denc,t);

axis([0 15 -1 2]); % 设置横坐标范围为 0 到 15

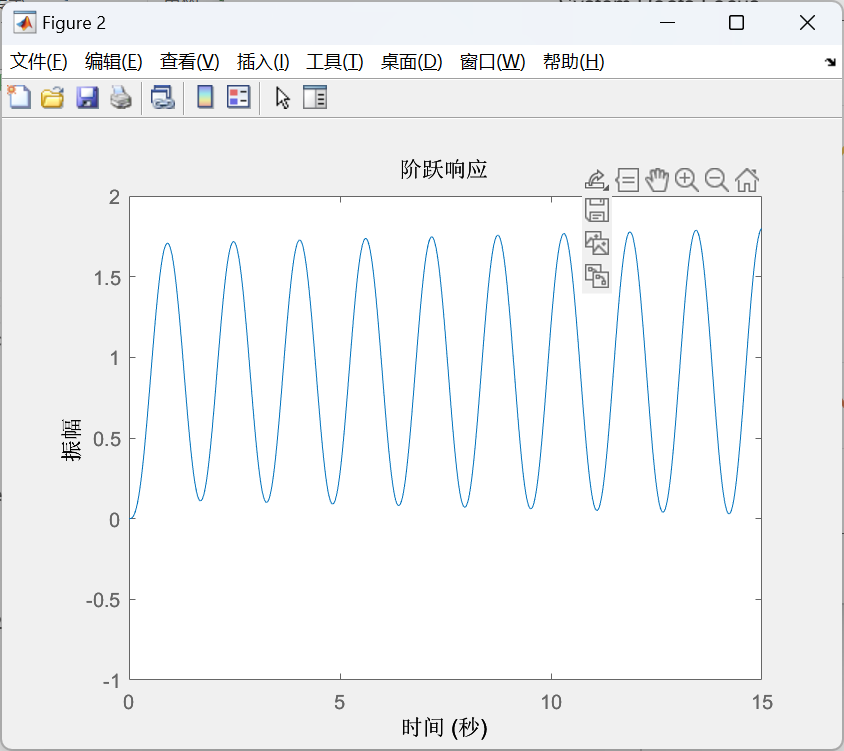
点击“保存”按钮，保存文件名为“sysrlocus.m”。选择“编辑器”菜单的“运行按选项或直接在MATLAB的命令行窗口下键入文件名sysrlocus，在MATLAB的命令行窗口下查看运行的结果。



图十一.系统临界稳定结果图

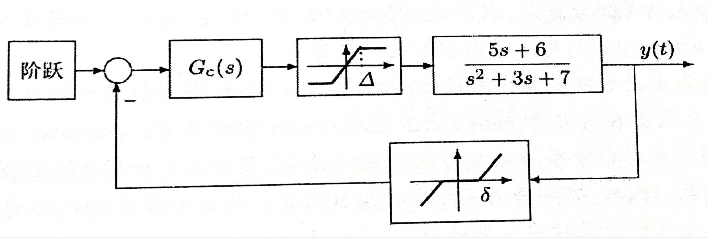


图十二.系统闭环根轨迹图



图十三.系统临界稳定时的响应曲线

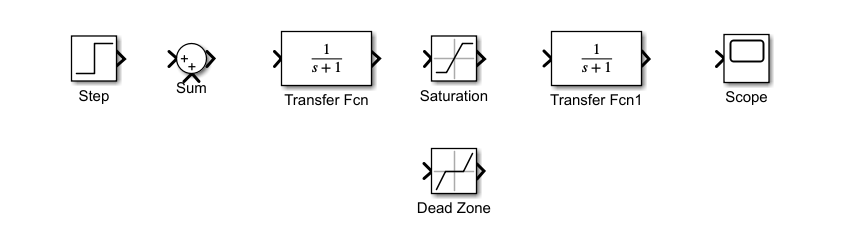
1. Simulink的非线性系统仿真
   1. 考虑图十四中给出的典型非线性反馈系统框图，其中控制器为PI控制器，其模型为,且Kp=3，Ki=10，饱和非线性中的Δ=2，死区非线性的死区宽度为 δ=0.1。试建立其Simulink仿真模型并用示波器观察输出。



图十四.非线性系统

步骤一、**打开模型编辑窗口**。首先打开一个空白模型编辑窗口，这可以直接在命令控制行内键入“simulink”或者单击主页菜单下的“simulink”菜单项实现，之后选择新建空白模型。

步骤二、**复制相关模块**。点击库浏览器，将相关的模块组中的模块拖动到此窗口，例如将Sources组中的Step模块拖动到此窗口中，将Math组中的加法器拖动到此窗口中等，这样就可以将如图十五所示的一些模块复制到模型编辑窗口中



图十五.编辑窗口

步骤三、**修改模块参数**。通过观察发现，其中很多模块的参数和要求的不一致，如受控源对象模型、控制器模块、加法器模块等。双击加法器模块，将打开如图十六所示的对话框，其中符号列表 (List of Signs)栏目描述加法器各输入的符号，其中|表示该路没有信号，所以用|+-取代原来的符号，就可以得出反馈系统中所需的减法器模块了。



图十六.常用模块参数对话框

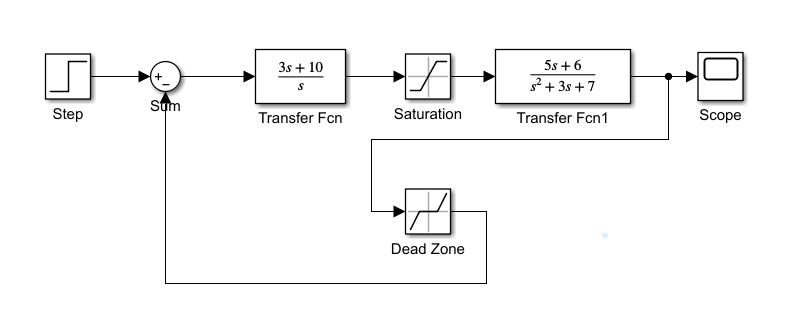
传递函数参数也可以相应地修改，双击控制器模块，则将打开如图十七所示的对话框，用户只需在其分子和分母栏目分别填写系统的分子多项式和分母多项式系数，就可以正确输入这两个模块了



图十七.传递函数模块参数设置对话框

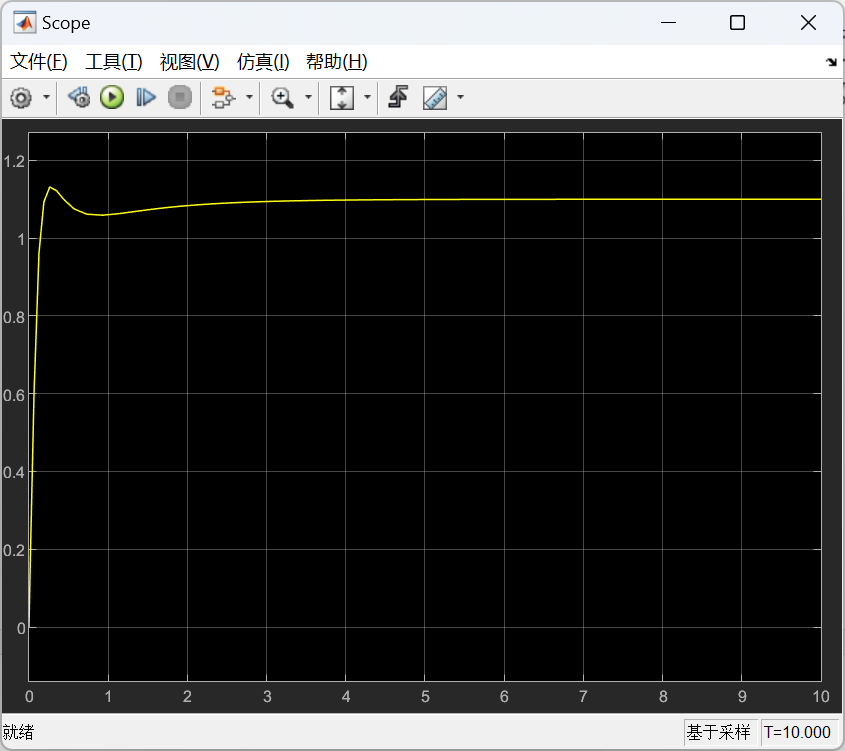
模型中还需要修改的参数如下：阶跃输入模块将阶跃时刻(Step time)默认的1修改为0；饱和非线性模块的饱和上界(Upper limit)和下界(Lower limit)参数分别设置为2和-2；死区非线性模块的死区起止值(Start of dead zone 和 End of zone)分别设置为-0.1和0.1。

步骤四、**模块连接**。将有关的模块直接连接起来，具体的方法是用鼠标单击某模块的输出端，拖动鼠标到另一模块的输入端处再释放，则可以将这两个连接起来。完成模块连接后，就可以得到如图十八所示的系统模型。



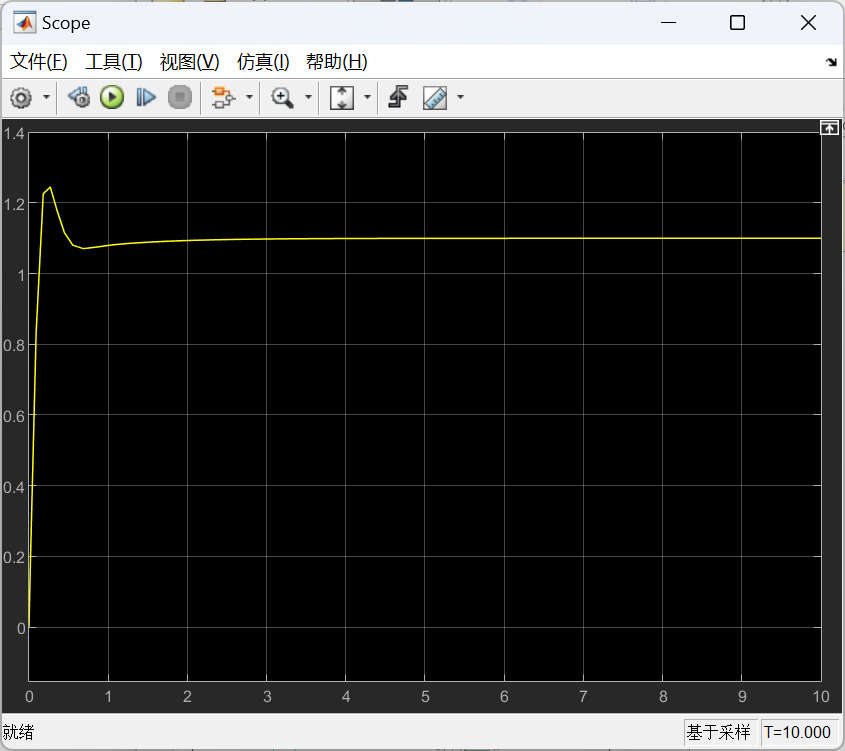
图十八.模块连接后的系统模型

步骤五、**系统仿真研究**。单击运行(Run)按钮，则可以启动仿真过程，这样双击示波器模块就可以显示仿真结果了，如图十九所示。



图十九.直接仿真结果

从仿真结果看，跟踪速度较慢。根据PI控制器设计经验。如果能加大Ki的值将有望加快系统响应速度，用手动调节的方法将Ki设置为20，则可以得出如图二十所示的仿真结果。



图二十.修改控制器后的结果