



**本科实验报告**

|  |  |
| --- | --- |
| 姓名： | Awslasasd |
| 学院： |  |
| 系： |  |
| 专业： |  |
| 学号： |  |
| 指导教师： |  |

2024年 9 月 20 日

**实验报告**

装订线

课程名称： Mworks仿真实验 实验类型： 仿真实验

实验项目名称： 现代控制理论仿真实验(1)

学生姓名： Awslasasd 专业： 学号：

同组学生姓名： 指导老师：

实验地点： 实验日期： 年 月 日

1. 实验目的和要求（必填）

(1)安装好Mworks-sylab软件，并熟悉其基本功能；

(2)熟悉 Mworks 及其在模型表示方法，掌握用 Mworks 进行方块图的转化；

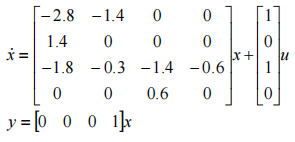
(3)熟悉 Mworks 的绘图方法，掌握用 Mworks进行控制系统时域分析的方法；

(4)熟悉 Mworks 中根轨迹的分析方法，掌握用 Mworks进行根轨迹的分析与设计。

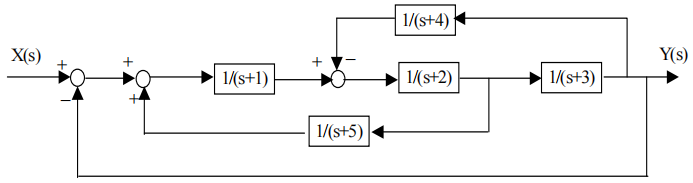
二、实验内容和原理（必填）

2.1实验内容

1.给定连续系统状态空间方程，求传递函数模型和零极点模型



2.根据系统方块图求输入输出传递函数（并与方框图法得到的传递函数进行比较）



1. 典型二阶系统 ，其中为自然频率（无阻尼振荡频率），ζ为相对阻尼系数，试绘制
2. 当=6时，ζ分别为 0.1,0.2,…,1.0,2.0 时的单位阶跃响应。
3. 当ζ=0.7时，分别为 2,4,6,8,10,12 时的单位阶跃响应。

4.计算二阶系统的时域指标（上升时间、超调量、峰值时间、稳态时间）。

装订线

5.非单位反馈控制系统的传递函数为： 绘制系统的根轨迹，确定具有最小阻尼比ζ的放大系数 A，并用零、极点、增益形式表示闭环传递函数

6. ，假设峰值 Mp=1.0948，确定满足Mp的ζ值对应的K值，并用零极点增益方式表示闭环传递函数。（计算精度±0.05）

2.2实验原理

1.稳定性判据：所有极点均在左半平面即认为稳定

2.求解时域指标：

1)求解系统的阶跃响应函数 ；

2)对C(s)进行拉普拉斯反变换，得到时间域的阶跃响应曲线 c(t)；

3)求解各项指标

上升时间 (Tr)：找到曲线从 10% 到 90% 的时间间隔。

超调量 (%OS)：找到曲线峰值与稳态值的差值，然后除以稳态值

峰值时间 (Tp)：找到曲线第一个峰值的时间。

稳态时间 (Ts)：找到曲线最终稳定在稳态值附近的时间。

3.利用Mworks-sylab中的各种函数实现各种模型之间的转换等

三、主要仪器设备（系统、软件或平台）

1.Windows电脑一台

2.Mworks-sylab 2024b 软件

四、操作方法与实验结果与分析

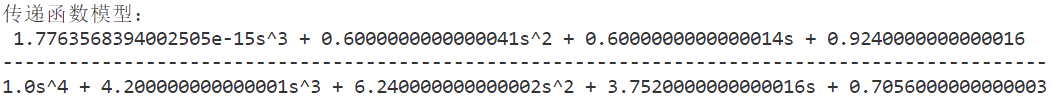
4.1 模型间的相互转换

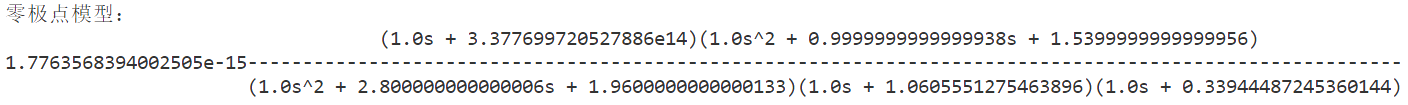
4.1.1 状态空间与传递函数

具体代码如下

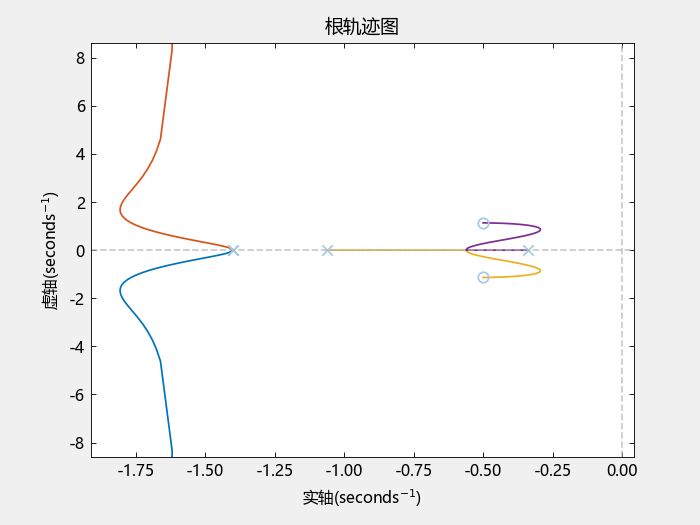
1. clc()
2. clear()
3. *# 定义状态空间方程*
4. a = [-2.8 -1.4 0 0;
5. 1.4 0 0 0;
6. -1.8 -0.3 -1.4 -0.6;
7. 0 0 0.6 0]
8. b = [1.0; 0; 1.0; 0]
9. c = [0 0 0 1.0]
10. d = 0
11. *# 方法1：利用ss函数*
12. println("传递函数模型： ")
13. G1 = ss(a,b,c,d)
14. sys\_tf = tf(G1)
15. println(sys\_tf)
16. println("零极点模型： ")
17. sys\_z = zpk(sys\_tf)
18. println(sys\_z)
19. rlocus(sys\_tf)

**方法一**得到结果如下图所示





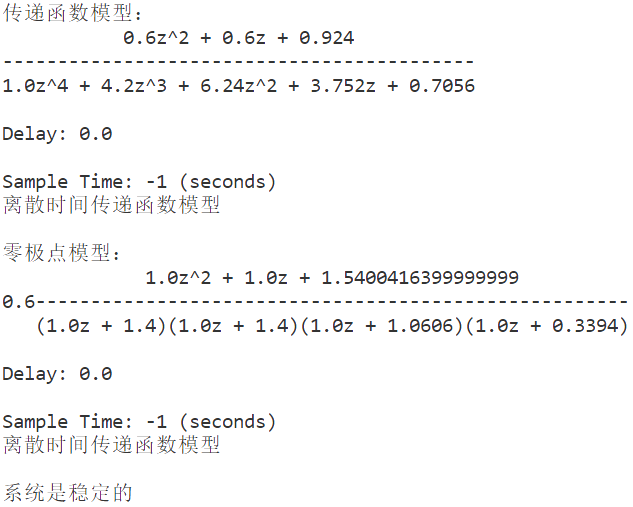
绘制得到的零极点图如下图所示



由根轨迹图可以看出，所有的极点均在左半平面，因此系统稳定。

1. *# 方法2：利用sst2f函数(不建议用)*
2. *# 计算传递函数的分子和分母*
3. num, den = ss2tf(a, b, c, d, 1)
4. *# 指定保留的小数位数*
5. digits=4
6. *# 对分子和分母进行保留小数处理*
7. num = round.(num;digits)
8. den = round.(den;digits)
9. *# 计算零点、极点和增益*
10. z, p, k = tf2zp(num, den)
11. *# 对零点、极点和增益保留小数*
12. z = round.(z;digits)
13. p = round.(p;digits)
14. k\_scalar = round(k[1];digits)
15. *# 创建传递函数模型和零极点增益模型*
16. sys\_tf = tf(num, den, -1)
17. sys\_zpk = zpk(z, p, k\_scalar, -1)
18. *# 判断系统稳定性*
19. stable = true
20. for pole in p
21. if real(pole) >= 0  *# 离散系统的极点判断*
22. stable = false
23. break
24. end
25. end
26. println("传递函数模型： ")
27. println(sys\_tf)
28. println("零极点模型： ")
29. println(sys\_zpk)
30. if stable
31. println("系统是稳定的")
32. else
33. println("系统是不稳定的")
34. end

**方法二**得到结果如下图所示

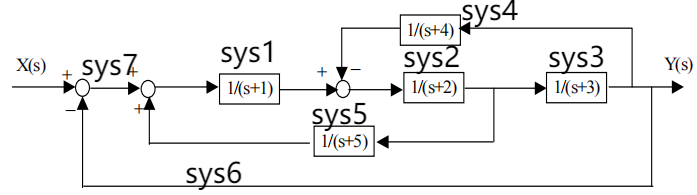


方法一是用ss()函数，再用tf()函数，而方法二是用sst2f(),再用tf()函数。在结果方面，由于Mworks精度很高，故出现了方法一的结果，但有些数量级的系数其实近乎为0。在方法二中，我对num、den、z、p、k进行了保留4位小数使得方法二得到的结果更加精简，但实际上两种方法在不保留小数时的结果是一致的。

**注**：sst2f()由于是信号库，容易出现数据格式问题，因此不常用该方法进行转换。

4.1.2 方块图与传递函数

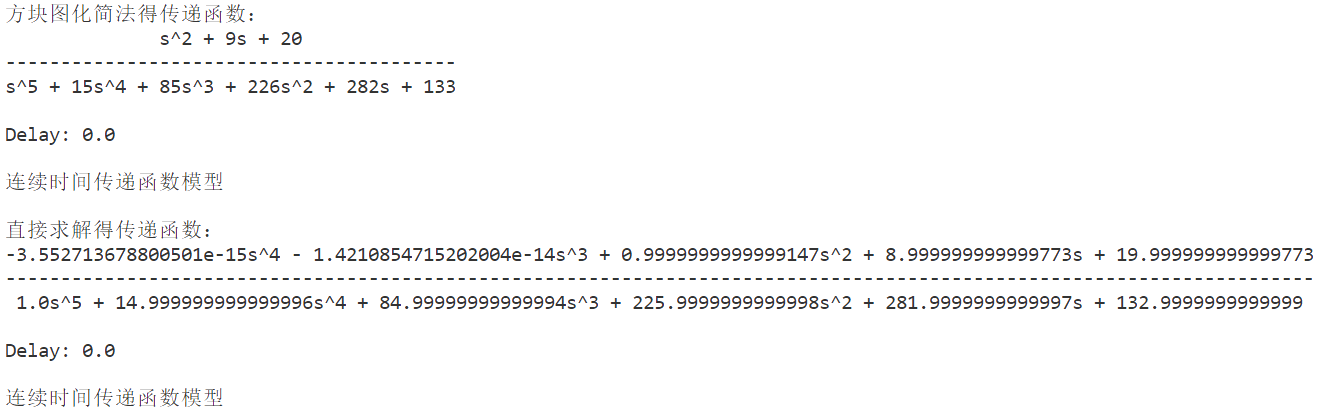
对其各项进行标注，如下图所示：



具体代码如下

1. clc()
2. clear()
3. sys1 = tf(1,[1;1])
4. sys2 = tf(1,[1;2])
5. sys3 = tf([1],[1;3])
6. sys4 = tf([1],[1;4])
7. sys5 = tf([1],[1;5])
8. sys6 = tf(-1,[1])
9. sys7 = tf(1,[1])
10. *# 化简方块图法*
11. println("方块图化简法得传递函数： ")
12. sys8 = series(sys2,sys3)
13. sys9 = feedback(sys8,sys4)
14. sys10 = series(sys9,sys1)
15. sys11 = feedback(sys10,-sys5/sys3)
16. sys12 = feedback(sys11,1) *# 传递函数模型*
17. println(sys12)
18. *# 直接利用connect()函数求解*
19. println("直接求解得传递函数： ")
20. blksys = append(sys1,sys2,sys3,sys4,sys5,sys6,sys7)
21. connections = [1 7 5;2 1 -4;3 2 0;4 3 0;5 2 0;6 3 0;7 6 0]
22. inputv = [7]
23. outputv = [3]
24. sys\_c = connect(ss(blksys),connections,inputv,outputv)
25. sys\_tf = tf(sys\_c)

运行结果如下图所示



从结果中我们可以看出，由方块图法得到的传递函数与直接利用connect()函数进行求解得到的传递函数在系数的精度上有一定的区别。

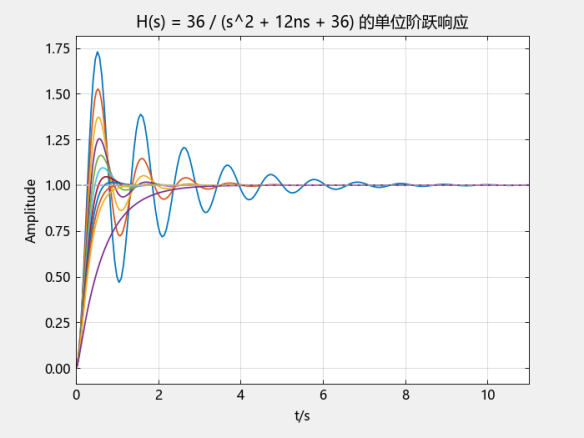
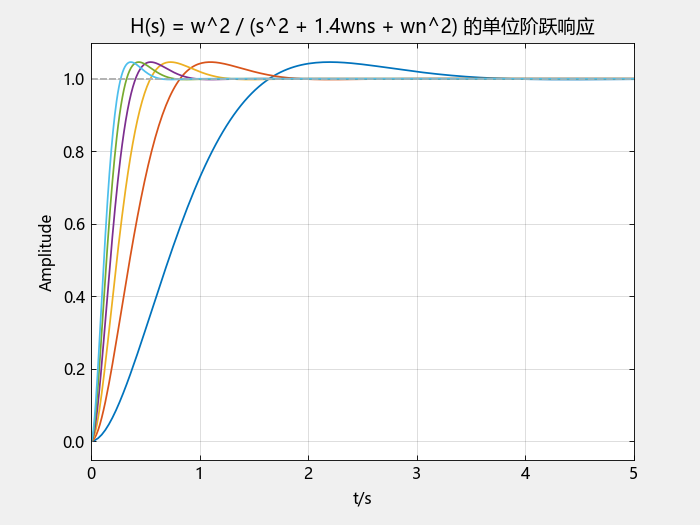
4.2 二阶系统时域分析

4.2.1 典型二阶系统的单位阶跃响应

具体代码如下

1. clc()
2. clear()
3. # 定义一个绘制阶跃响应的函数
4. function plot\_step\_response(num, den, figure\_num, x\_label, y\_label, plot\_title)
5. figure(figure\_num)
6. sys = tf(num,den)
7. step(sys)
8. grid("on")
9. hold("on")
10. xlabel(x\_label)
11. ylabel(y\_label)
12. title(plot\_title)
13. end
14. # 任务1: H(s) = 36 / (s^2 + 12ns + 36) 的单位阶跃响应
15. num1 = [0 0 36]
16. figure\_num = 1
17. x\_label = "t/s"
18. y\_label = "Amplitude"
19. plot\_title = "H(s) = 36 / (s^2 + 12ns + 36) 的单位阶跃响应"
20. for n = 0.1:0.1:1.0
21. local den = [1 12\*n 36]
22. plot\_step\_response(num1, den, figure\_num, x\_label, y\_label, plot\_title)
23. end
24. # 额外添加 n=2 的情况
25. den = [1 24 36];
26. plot\_step\_response(num1, den, figure\_num, x\_label, y\_label, plot\_title);
27. # 任务2: H(s) = w^2 / (s^2 + 1.4wns + wn^2) 的单位阶跃响应
28. figure\_num = 2
29. plot\_title = "H(s) = w^2 / (s^2 + 1.4wns + wn^2) 的单位阶跃响应"
30. for n = 2:2:12
31. local num2 = [0 0 n^2]
32. local den = [1 1.4\*n n^2]
33. plot\_step\_response(num2, den, figure\_num, x\_label, y\_label, plot\_title)
34. end

结果如下图所示

可以看出自然频率相同时，阻尼比ζ在 0-1 之间时，为欠阻尼状态，会发生振荡，ζ值越小振荡越明显；阻尼比ζ大于 1 后，系统为过阻尼，不发生振荡。

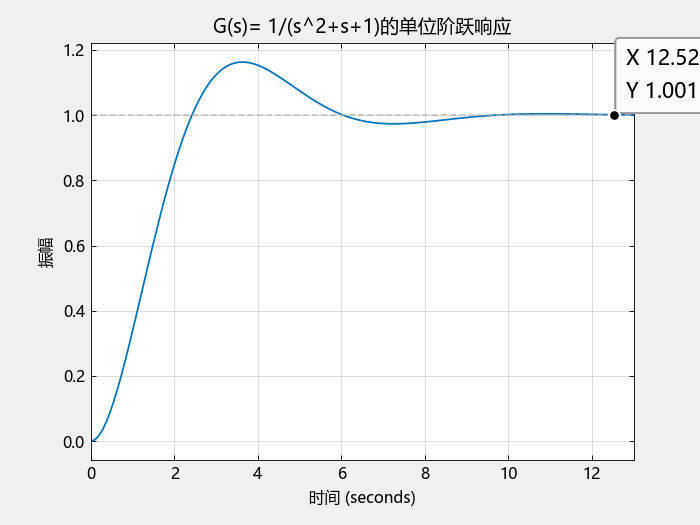
而阻尼比ζ相同时， 越大，响应速度越快，稳定时间越短

4.2.2 二阶系统的时域指标

代码如下：

1. clc()
2. clear()
3. *# 定义传递函数*
4. num = [0 0 1]
5. den = [1 1 1]
6. G1 = tf(num,den)
7. *# 绘制阶跃响应*
8. figure(1)
9. step(G1)
10. grid("on")
11. title("G(s)= 1/(s^2+s+1)的单位阶跃响应")
12. *# 计算响应信息*
13. global Mp = stepinfo(G1; RiseTimeLims = [0 1.0], SettlingTimeThreshold = 0.03)
14. *# 系统参数*
15. omega\_n = 1;
16. zeta = 0.5;
17. *# 计算与响应时间相关的公式*
18. beta = atan(sqrt(1-zeta^2)/zeta)
19. Tr = (pi-beta)/omega\_n/sqrt(1-zeta^2) *# 理论上升时间*
20. sigma = exp(-zeta\*pi/sqrt(1-zeta^2))  *# 超调百分比*
21. Tp = pi/omega\_n/sqrt(1-zeta^2)        *# 峰值时间*
22. Ts = 3/omega\_n/zeta                  *# 稳态时间*
23. *# 输出结果*
24. println("函数计算结果： ")
25. println("上升时间：")
26. println(Mp.RiseTime[1])
27. println("超调量: ")
28. println(Mp.Overshoot[1]/100)
29. println("峰值时间：")
30. println(Mp.PeakTime[1])
31. println("稳态时间: ")
32. println(Mp.SettlingTime[1])
33. println(" ")
34. println("公式计算结果： ")
35. println("上升时间: ")
36. println(Tr)
37. println("超调量: ")
38. println(sigma)
39. println("峰值时间：")
40. println(Tp)
41. println("稳态时间: ")
42. println(Ts)

结果如下图所示

从左图可以看到函数计算以及公式计算得到的不同结果，其中，上升时间、超调量和峰值时间差距不大，但是稳态时间存在较大的差别，经过查库可得，函数Mp.SettlingTime得到的是误差保持在2%以下的时间。

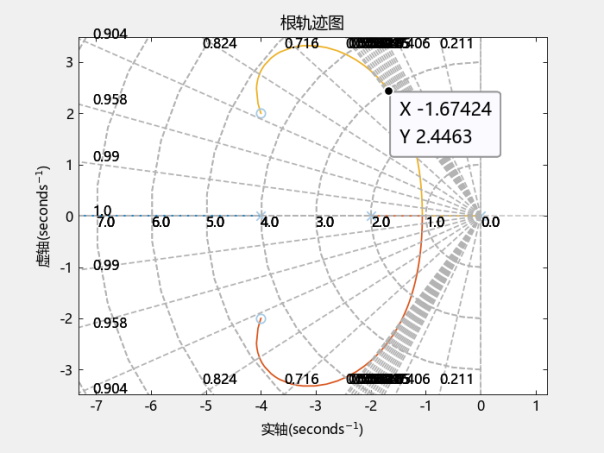
4.3 根轨迹的分析方法

4.3.1 根轨迹的绘制

代码如下：

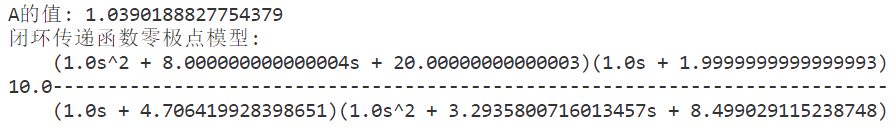
1. clear()
2. clc()
3. *# 创建传递函数模型sys1*
4. num1 = [10 80 200]
5. den1 = [1 4 0]
6. sys1 = tf(num1,den1)
7. *# 创建传递函数模型sys2*
8. num2 = [0.2]
9. den2 = [1 2]
10. sys2 = tf(num2,den2)
11. *# 绘制系统sys的根轨迹图*
12. sys = sys1 \* sys2
13. rlocus(sys)
14. sgrid("on")
15. *# 在根轨迹图上绘制指定阻尼比的直线*
16. zeta = 0.450:0.01:0.577
17. wn = [ ]
18. sgrid(zeta,wn)

绘制的根轨迹与阻尼比直线如下图所示，由图中可以找到最小阻尼比所在的点为



1. *# 计算A的值*
2. a = -1.67424 + 2.4463im
3. k = abs( a\*(a + 2)\*(a + 4)/(a^2 + 8\*a + 20)/2)
4. println("A的值: ",k)
5. *# 闭环传递函数的零极点模型*
6. sys3 = feedback(sys1,sys2)
7. println("闭环传递函数零极点模型: ")
8. zp = zpk(sys3)
9. println(zp)

然后再计算该点对应的A的值，并计算零极点传递函数，结果如下图所示

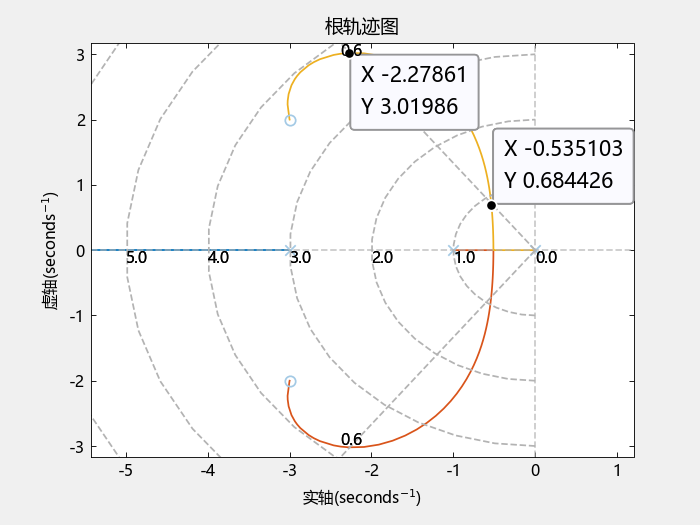


4.3.2 根轨迹的分析与设计

代码如下：

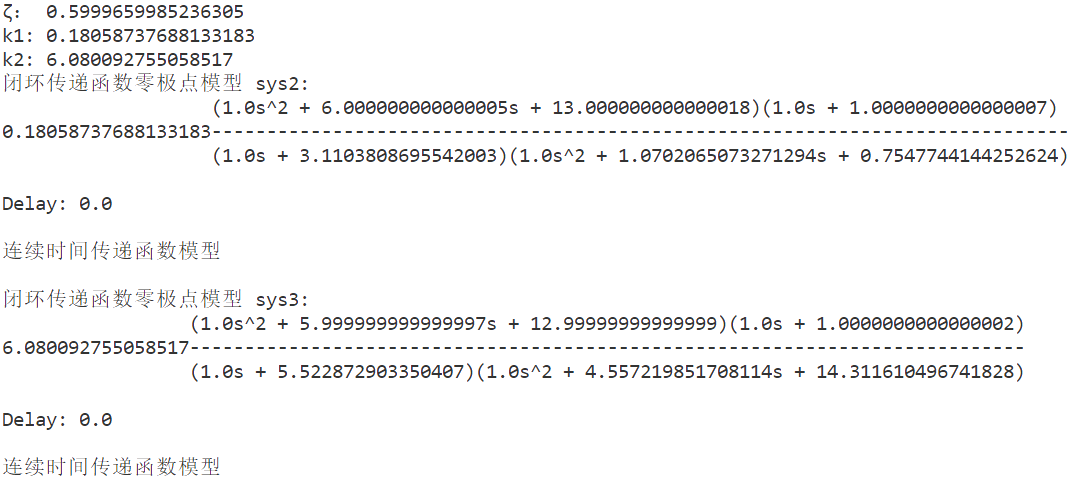
1. clear()
2. clc()
3. *# 计算阻尼比*
4. zeta = sqrt(log(0.0948)^2 / (log(0.0948)^2 + pi^2))
5. println("ζ： ", zeta)
6. *# 定义系统1和系统2的分子和分母多项式系数*
7. num1 = [1, 6, 13]
8. den1 = [1, 3, 0]
9. num2 = [1]
10. den2 = [1, 1]
11. *# 创建传递函数模型并串联*
12. sys = series(tf(num1, den1), tf(num2, den2))
13. *# 绘制根轨迹图*
14. rlocus(sys)
15. *# 在根轨迹图上绘制指定阻尼比的直线*
16. sgrid([zeta], [])

由根轨迹以及给定阻尼比可以绘制出下图，由阻尼比直线与根轨迹可以得到其交点分别为 -0.535103+0.684426j、-2.27861+3.01986j。



1. *# 由上根轨迹得到复数a和b*
2. a = -0.535103 + 0.684426im
3. b = -2.27861 + 3.01986im
4. *# 计算k1和k2*
5. k1 = abs(a \* (a + 1) \* (a + 3) / (a^2 + 6 \* a + 13))
6. k2 = abs(b \* (b + 1) \* (b + 3) / (b^2 + 6 \* b + 13))
7. println("k1: ",k1)
8. println("k2: ",k2)
9. *# 定义调整后的系统3和系统4的分子和分母多项式系数*
10. num3 = k1 \* [1, 6, 13]
11. den3 = [1, 3, 0]
12. num4 = k2 \* [1, 6, 13]
13. den4 = [1, 3, 0]
14. *# 创建闭环系统模型*
15. sys2 = feedback(tf(num3, den3), tf(num2, den2))
16. sys3 = feedback(tf(num4, den4), tf(num2, den2))
17. *# 打印闭环传递函数的零极点模型*
18. println("闭环传递函数零极点模型 sys2: ")
19. println(zpk(sys2))
20. println("闭环传递函数零极点模型 sys3: ")
21. println(zpk(sys3))

由交点可以分别计算出其对应的增益K的值，再构造零极点传递函数模型如下所示，K分别为0.18058737688133183和6.080092755058517



五、讨论与心得

5.1 讨论

一方面，这是我第一次接触Mworks，他的很多语法和函数其实和Matlab相似，在韩涛老师没有讲之前我就开始自己尝试对照着手册来做本次实验。Mworks的计算精度太高，导致很多结果在保留小数之后应该是另一形式。还有过程中也遇到了ss2ft( )关于数据格式的问题。在听完线上课后也是听到韩涛老师不建议用ss2ft（）的方法去计算。

另一方面，这次实验也增强了我对于根轨迹，系统传递函数与时域性能指标的知识的温习，在上面的题目中，不仅仅会简单的调用函数，还去用其公式求解结果，这更加加深了我对其的理解。

5.2 心得

通过这次实验，我深刻体会到了控制系统分析技术的重要性。在国家层面，这些技术是推动工业自动化、智能制造、航空航天等关键领域发展的基础。掌握先进的控制系统分析技术，对于提升国家的科技实力和产业竞争力具有重要意义。在实验过程中，我还感受到了理论与实践相结合的重要性。这不仅仅是一个技术训练，更是对分析问题和解决问题能力的培养。国家的发展需要大量具备实践能力和创新精神的人才，而大学教育在其中扮演着至关重要的角色。此外，在实验中，我尝试了不同的方法来解决问题，这体现了创新思维的重要性。对国家而言，创新是推动社会进步和经济发展的关键动力。通过鼓励创新思维和实践，可以促进新技术、新产品的开发，从而推动社会的整体进步。