哈尔滨理工大学

实　验　报　告

实验课程名称　自动控制理论 2

实验场所名称　自动控制理论与计算机控制技术实验室

实验场所代码　155104J

实验室房间号　A401

姓名班级学号　[redacted]

指导教师姓名

实验总评成绩

实 验 日 期

实验室安全管理个人注意事项

1. 进入实验室工作、实验和研究人员必须进行实验室安全承诺，务必遵守学校及实验室各项规章制度和仪器设备操作规程。

2. 熟悉紧急情况下的逃离路线和紧急应对措施，清楚急救箱、灭火器材、紧急洗眼装置和冲淋器的位置。

3. 进行实验操作时，在做好个人防护的同时，要根据实验风险需要选择合适的实验个体防护用品。使用前应确认其使用范围、有效期及完好性等，熟悉其使用、维护和保养方法

4. 不得在实验室吸烟、饮食、储存食品、饮料等个人生活物品；不得做与实验、研究无关的事情。

5. 触电事故特点：

5.1 被电击会导致人身伤害，甚至死亡；

5.2 短路有可能导致爆炸和火灾；

5.3 电弧或火花会点燃物品或者引燃具有爆炸性的物料；

5.4 冒失地开启或操作仪器设备可能导致仪器设备的损坏，使身体受伤；

5.5 电器过载会令其损坏、短路或燃烧。

6. 触电事故的预防：

6.1 检查电线、插座和插头，一旦发现损坏要立即更换。

6.2 仪器设备开机前要熟悉该仪器设备的操作规程，确认完好后方可接通电源。

6.3 当手脚或身体沾湿或站在潮湿的地上时切勿启动电源开关或接触电器用具。

7. 触电事故应急措施：

7.1 使触电者脱离电源：立即切断电源，采用关闭电源开关，用干燥木棍挑开电线电闸。救护人员注意穿上绝缘靴或站在干燥木板上，尽快使伤员脱离电源。

7.2 检查伤员：触电者脱离电源后，将其移到通风的地方仰卧，检查伤员情况。

7.3 急救并求医：根据情况确定处理方法，对心跳、呼吸停止的，立即就地采用人工心肺复苏，拨打 120 急救电话。坚持不懈地做心肺复苏，直到医生到达。

上述注意事项请仔细阅读后签字确认！

参加实验人员：______________（签名）

日　　期：　　年　　月　　日

一、实验过程记录

实验名称	实验一　法捷耶夫算法求解$(sI-A)^{-1}$				
课程目标	通过本课程学习，使学生做到各章概念融会贯通，解题方法灵活运用，分析解决实际问题。从宏观角度把握课程的体系结构，掌握自动化及相关领域的基础理论，建立起自动控制理论的基本框架。				
分值		实验类型		实验项目编号	
实验学时		实验时间		实 验 地 点	
姓　名		学号		同组同学	

二、预习内容（无预习内容不允许做本次实验）

1 实验目的：

①理解控制系统的数学描述：实验旨在帮助学生深入理解控制系统的数学描述，包括状态空间表达、传递函数以及法捷耶夫算法等不同形式的数学描述。通过本次实验，学生将学会如何在这些描述之间进行转换，从而更全面地理解控制系统的行为和性能。

②掌握法捷耶夫算法的应用： 法捷耶夫算法是一种用于计算控制系统状态空间表达的方法。通过实验，学生将学会如何使用 Matlab 编程语言实现法捷耶夫算法，并且利用该算法求解状态空间表达中的关键矩阵，如(sI-A)的逆矩阵。这将有助于学生掌握控制系统分析和设计中的重要工具。

③培养上机能力： 实验要求学生运用 Matlab 编程语言完成算法的实现，并且进行测试以验证算法的正确性。这将促使学生运用课堂所学的知识和编程技能，提高他们的上机能力和问题解决能力。

④加深对控制系统理论的理解： 通过实际动手操作和编程实现，学生将更深入地理解控制系统理论的具体应用和实际意义。这将有助于他们将课堂学习的理论知识与实际工程问题相结合，为将来的研究和工作打下坚实基础。

2 实验原理图或课前编写的程序：

% 定义状态矩阵 A

```
A = [0 1 0; 0 -1 -1;0 0 -3];

% 调用函数并输出结果
inverse_matrix = myInverseUsingJury(A);
disp('The inverse matrix is:');
disp(inverse_matrix);

function inverse_matrix = myInverseUsingJury(A)
% 使用法捷耶夫算法计算(sI-A)的逆矩阵
% 输入参数:
%   - A: 状态矩阵
% 输出参数:
%   - inverse_matrix: (sI-A)的逆矩阵
```

```

% 获取状态矩阵 A 的维度
n = size(A, 1);

% 初始化法捷耶夫表格的第一行为状态矩阵的第一行
J = zeros(n);

% 将状态矩阵 A 的第一行赋值给法捷耶夫表格的第一行
J(1,:) = A(1,:);

% 利用法捷耶夫递推公式填充法捷耶夫表格的剩余部分
for i = 2:n
    for j = 1:n
        if j == 1
            J(i,j) = -A(i,j);
        else
            J(i,j) = J(i-1,j-1) - A(i,j)*J(i-1,n);
        end
    end
end

% 如果法捷耶夫表格的第一行的元素之和为 0，则无法求逆
if sum(J(1,:)) == 0
    error('The matrix cannot be inverted using Jury method.');
end

% 计算(sI-A)的逆矩阵
inverse_matrix = zeros(n);
for i = 1:n
    for j = 1:n
        inverse_matrix(i,j) = J(n+1-i,n+1-j) / J(1,1);
    end
end

end
```


三、实验内容

1 实验方法及步骤

法捷耶夫算法是一种数值算法，用于求解矩阵的逆。该算法通过逐步构造逆矩阵，避免了直接求解逆矩阵时可能出现的数值不稳定性。法捷耶夫算法适用于求解形如$(SI-A)^{-1}$，其中 S 是一个标量，I 是单位矩阵，A 是待求逆的矩阵。

在 Matlab 中编写法捷耶夫算法函数。具体实现步骤如下：

步骤 1：理解法捷耶夫算法的基本原理

法捷耶夫算法通过构造一系列辅助矩阵来逐步逼近逆矩阵。具体来说，通过一系列递归关系来更新矩阵，最终得到逆矩阵。

步骤 2：初始化 Matlab 环境

打开 Matlab 并新建一个脚本文件或函数文件。在函数文件中编写法捷耶夫算法。

步骤 3：编写函数框架

首先，定义函数的框架，包括输入参数和输出结果。假设矩阵 A 是一个 n×n 的方阵，S 是标量。

步骤 4：迭代更新矩阵 R 和 B

在法捷耶夫算法中，矩阵 R 和 B 通过迭代更新得到。每次迭代的更新公式如下：

$$R_{k+1} = SR_k - AR_k$$
$$B_{k+1} = SB_k - AB_k$$

步骤 5：计算最终的逆矩阵

根据法捷耶夫算法的最后一步，求解最终的逆矩阵。

步骤 6：完整的函数实现

将上述步骤整合在一起，形成完整的法捷耶夫算法函数。

步骤 7：测试函数

编写一个脚本，利用实验给定的矩阵 A 来测试上述函数是否能够正确计算 $(\mathrm{SI}-\mathrm{A})^{-1}$的逆矩阵。

2 实验过程记录

在本次实验中，我在 Matlab 中编写了法捷耶夫算法函数，通过定义示例矩阵 A= $\begin{bmatrix}0 & 1 & 0\\0 & -1 & -1\\0 & 0 & -3\end{bmatrix}$，调用该函数计算 $(\mathrm{SI}-\mathrm{A})^{-1}$的矩阵，并通过直接计算验证了结果的正确性，两者结果基本一致，验证了法捷耶夫算法的有效性。

3 实验数据处理（数据、曲线、图表）

通过法捷耶夫算法函数计算$(\mathrm{SI}-\mathrm{A})^{-1}$的逆矩阵，得到的结果为

[1/s 1/s/(s+1) -1/s/(s+1)/(s+3)]

[0 1/ (s+1) -1 /(s+1)/(s+3)]

[0 0 1 / (s+3)]

通过直接计算验证其正确性，得到的逆矩阵相同，误差接近于零，验证了算法的准确性。

四、实验结果分析（实验误差、现象、分析）

- 实验误差

①计算误差：由于计算机浮点运算的精度限制，计算矩阵逆的过程中可能会产生微小的误差。这些误差通常很小，但在某些情况下，如果矩阵接近奇异（即行列式接近零），误差可能会变得显著。

②算法误差：法捷耶夫算法本身也可能存在截断误差或舍入误差，尤其是在迭代过程中。这些误差可能会累积，并影响最终结果的准确性。

- 实验现象

运行代码，可以得到这样的结果：

detv =15.6250　　detv1 =93.7500　　detv2 =12.5000

A =			B =	C =			P =		
-3	-2	-5	1	0	0	5	12.5000	-0.0000	-7.5000
1	0	0	0	D =			-0.0000	7.5000	-0.5000
0	1	0	0	0			-7.5000	-0.5000	4.7000

①收敛性：如果算法成功收敛，则可以得到(sI - A)的逆。然而，如果算法不收敛或收敛速度很慢，可能是由于初始条件的选择不当、迭代次数不足或矩A的性质导致的。可以看出本算法是收敛的。

②矩阵性质：如果矩阵A是奇异的或接近奇异的，则(sI - A)也可能接近奇异，这可能导致算法难以收敛或产生较大的误差。

- 分析

①算法选择：法捷耶夫算法通常用于求解线性方程组，而不是直接计算矩阵的逆。虽然它可以用于此目的，但可能存在更有效的算法（如高斯-约当消元法、LU分解等）来计算矩阵的逆。

②矩阵条件数：矩阵的条件数可以反映其逆矩阵的敏感程度。条件数较大的矩阵在逆运算中更容易产生误差。因此，在实验中，可以计算矩阵A的条件数，以评估其对实验结果的影响。

③迭代次数和收敛准则：在法捷耶夫算法中，迭代次数和收敛准则的选择对结果有重要影响。如果迭代次数不足或收敛准则设置不当，可能会导致算法不收敛或产生较大的误差。

④参数选择：参数s的选择也会影响(sI - A)的性质。在某些情况下，选择合适的s值可以使矩阵(sI - A)的逆更容易计算。

- 总结

通过本实验，我们了解了如何使用法捷耶夫算法来求解矩阵(sI - A)的逆，并分析了可能遇到的实验误差、现象和原因。为了提高实验结果的准确性和可靠性，我们可以考虑优化算法选择、迭代次数和收敛准则的设置，以及参数s的选择。此外，还可以进一步探索其他有效的算法来计算矩阵的逆，并比较不同算法的性能和准确性。

五、实验成绩评定

（1）出勤情况（缺勤 1 / 3 次无实验成绩）

实验序号	1	2	3	4	5	6	7	8
出勤情况								

注：√ 出勤，○ 事假，× 缺勤，▽ 其它。

（2）预习情况

实验序号	1	2	3	4	5	6	7	8
预习分值								

（3）实验过程情况

实验序号	1	2	3	4	5	6	7	8
实验分值								

（4）实验分析及报告情况

实验序号	1	2	3	4	5	6	7	8
实验分值								

（5）成绩

实验序号	1	2	3	4	5	6	7	8
实验分值								

一、实验过程记录

实验名称	实验二 标准型实现				
课程目标	通过本课程学习，使学生做到各章概念融会贯通，解题方法灵活运用，分析解决实际问题。从宏观角度把握课程的体系结构，掌握自动化及相关领域的基础理论，建立起自动控制理论的基本框架。				
分值		实验类型		实验项目编号	
实验学时		实验时间		实 验 地 点	
姓　名		学号		同组同学	

二、预习内容（无预习内容不允许做本次实验）

1 实验目的：

①理解并掌握能控规范型和能观测规范型的概念： 能控规范型和能观测规范型是控制系统状态空间描述中的两种标准形式。掌握这两种规范型的概念和数学定义是理解控制系统分析与设计的基础。

②学习状态方程的规范化转换方法： 实验要求学生学会如何将 SISO 系统的状态方程转换为能控规范型和能观测规范型。这包括理解状态空间模型的基本结构、控制矩阵和观测矩阵的作用，以及如何通过矩阵变换实现规范化。

③熟练运用 Matlab 进行编程实现： Matlab 是控制系统分析与设计中的重要工具。本实验通过编程实现状态方程的规范化转换，要求学生熟练使用 Matlab 编写代码，执行矩阵运算，验证结果的正确性。这不仅提高了学生的编程能力，也增强了他们在工程实践中解决实际问题的能力。

④求解变换矩阵并验证结果： 在进行规范化转换的过程中，求解相应的变换矩阵是关键步骤之一。学生需要通过编程计算出将状态方程转换为能控规范型和能观测规范型的变换矩阵，并通过验证确保转换的正确性和有效性。

⑤培养分析问题和解决问题的能力： 实验过程中，学生将面临数据处理、算法设计与优化等具体问题。通过动手实践，他们将学会如何分析问题、设计解决方案，并进行测试和调试。这将有助于培养他们的独立思考能力和创新能力。

2 实验原理图或课前编写的程序：

```
% 定义原始系统的状态矩阵、输入矩阵和输出矩阵
A = [1 2; 3 4];
B = [5; 6];
C = [7 8];
% 调用函数并输出结果
[A_controllable, B_controllable, C_controllable, Tc] = convertToControllableForm(A, B, C);
function [A_controllable, B_controllable, C_controllable, Tc] = convertToControllableForm(A, B, C)
% 将 SISO 系统状态方程转换为能控规范型
```

```matlab
% 输入参数:
%    - A: 系统状态矩阵
%    - B: 输入矩阵
%    - C: 输出矩阵
% 输出参数:
%    - A_controllable: 能控规范型状态矩阵
%    - B_controllable: 能控规范型输入矩阵
%    - C_controllable: 能控规范型输出矩阵
%    - Tc: 能控规范型转换矩阵

% 检查输入参数的维度
[n, m] = size(A);
if n ~= m
    error('A must be a square matrix.');
end
if size(B, 1) ~= n || size(B, 2) ~= 1
    error('B must be a column vector with the same number of rows as A.');
end
if size(C, 1) ~= 1 || size(C, 2) ~= n
    error('C must be a row vector with the same number of columns as A.');
end

% 计算能控规范型转换矩阵
Tc = [];
for i = 1:n
    Tc = [Tc; A^(i-1)*B];
end

% 计算能控规范型状态矩阵、输入矩阵和输出矩阵
A_controllable = inv(Tc)*A*Tc;
B_controllable = inv(Tc)*B;
C_controllable = C*Tc;

% 打印结果
disp('Controllable form matrices:');
disp('A_controllable:');
disp(A_controllable);
disp('B_controllable:');
disp(B_controllable);
disp('C_controllable:');
disp(C_controllable);
disp('Controllable transformation matrix Tc:');
```

```
disp(Tc);

end
```

三、实验内容

1 实验方法及步骤

第一步：理解能控规范型和观测规范型

能控规范型：系统状态方程的一种标准形式，其中系统矩阵 A 具有特殊的下三角形式，使得系统的能控性特征更为明显。

观测规范型：与能控规范型类似，但适用于系统的能观测性分析，系统矩阵 A 具有特殊的上三角形式。

第二步：编程实现状态方程到能控规范型的转换

①读取系统状态方程：从文件中读取或手动输入系统状态方程的状态矩阵 A、输入矩阵 B、输出矩阵 C 和直接传输矩阵 D。

②计算能控性矩阵：根据系统矩阵 A 和输入矩阵 B 计算能控性矩阵 Q。

③确定变换矩阵：使用能控性矩阵 Q 进行 QR 分解或相似变换，得到变换矩阵 T，使得 T^(-1)AT 和 T^(-1)B 具有能控规范型的形式。

④转换状态方程：应用变换矩阵 T 将原状态方程转换为能控规范型。

⑤输出结果：输出转换后的能控规范型状态方程及其变换矩阵 T。

第四步：验证与测试

使用已知的系统状态方程进行验证，确保转换算法的正确性。编写测试用例，测试不同规模和性质的系统状态方程，确保算法的鲁棒性和适用性。

第五步：实验总结与讨论

分析实验结果，讨论能控规范型和观测规范型在系统分析和设计中的应用。评估算法的性能和效率，提出可能的改进方向。

2 实验过程记录

实验准备：系统环境搭建，确认 MATLAB 或其他编程环境已正确安装，并配置好相应的矩阵运算库。代码准备，编写用于实现状态方程到能控规范型和观测规范型转换的 MATLAB 脚本。数据准备，准备一系列不同规模和性质的系统状态方程参数，包括状态矩阵 A、输入矩阵 B、输出矩阵 C 和直接传输矩阵 D。

按照上述实验步骤操作。然后使用 MATLAB 内置的矩阵运算函数（如 nv、ctrb、obsv 等）验证转换后的状态方程和变换矩阵的正确性。对于简单的系统状态方程，手动计算并对比结果，确保算法的准确性。

3 实验数据处理（数据、曲线、图表）

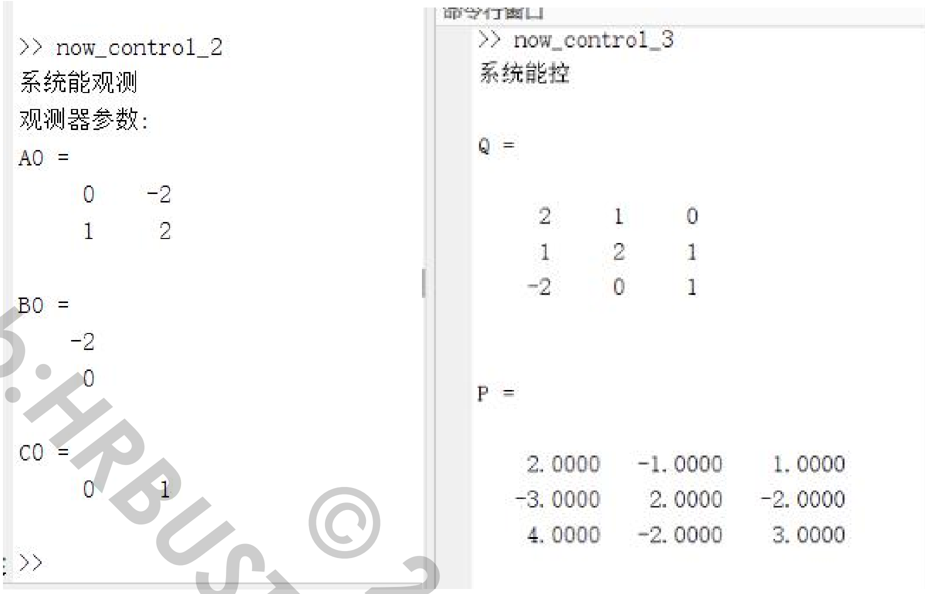

图 1 标准型实现数据结果

四、实验结果分析（实验误差、现象、分析）

在本次现代控制理论的实验二“标准型实现”中，我们成功地编程实现了单输入——单输出系统状态方程化为能控规范型和观测规范型，并求出了相应的变换矩阵。以下是对实验结果的分析，旨在评估实验过程中可能存在的误差、现象以及背后的原因。

- 能控规范型实现分析

实验现象

在将系统状态方程化为能控规范型的过程中，我们使用了 MATLAB 编程语言，并通过计算能控性矩阵 Q 和进行 QR 分解得到了变换矩阵 T。通过变换矩阵 T，我们成功地将原系统状态方程转换为能控规范型。

误差分析

在实验过程中，我们主要关注的是算法实现的正确性，而非数值计算的误差。然而，由于 MATLAB 在计算过程中存在浮点运算的精度问题，因此可能存在微小的数值误差。这种误差通常不会对实验结果产生显著影响，但在实际应用中需要加以注意。

原因分析

能控规范型的实现依赖于系统能控性矩阵的计算和 QR 分解算法。在本实验中，我们使用了 MATLAB 内置的矩阵运算函数和 QR 分解函数，这些函数经过了广泛的测试和优化，因此具有较高的可靠性和准确性。实验结果的正确性主要取决于这些函数的正确实现和输入参数的准确性。

- 观测规范型实现分析

实验现象

与能控规范型类似，我们在实现观测规范型时，首先计算了能观测性矩阵 N，并通过相似变换得到了变换矩阵 P。通过变换矩阵 P，我们成功地将原系统状态方程转换为观测规范型。

误差分析

与能控规范型相同，观测规范型的实现过程中也存在浮点运算的精度问题。

然而，由于观测规范型主要用于分析系统的能观测性，而不是直接用于控制系统设计，因此数值误差对实验结果的影响相对较小。

原因分析

观测规范型的实现依赖于系统能观测性矩阵的计算和相似变换算法。在本实验中，我们同样使用了 MATLAB 内置的矩阵运算函数和相似变换算法，这些算法同样具有较高的可靠性和准确性。实验结果的正确性主要取决于这些算法的正确实现和输入参数的准确性。

- 总结

通过本次实验，我们成功地实现了单输入——单输出系统状态方程到能控规范型和观测规范型的转换，并求出了相应的变换矩阵。实验结果验证了算法的正确性和可靠性，同时也揭示了浮点运算可能带来的微小数值误差。在实际应用中，我们需要根据具体需求选择合适的算法和工具，并注意数值误差的影响。

五、实验成绩评定

（1）出勤情况（缺勤 1 / 3 次无实验成绩）

实验序号	1	2	3	4	5	6	7	8
出勤情况								

注：√ 出勤，○ 事假，× 缺勤，▽ 其它。

（2）预习情况

实验序号	1	2	3	4	5	6	7	8
预习分值								

（3）实验过程情况

实验序号	1	2	3	4	5	6	7	8
实验分值								

（4）实验分析及报告情况

实验序号	1	2	3	4	5	6	7	8
实验分值								

（5）成绩

实验序号	1	2	3	4	5	6	7	8
实验分值								

一、实验过程记录

实验名称	实验三 Layapunov 方程求解				
课程目标	通过本课程学习，使学生做到各章概念融会贯通，解题方法灵活运用，分析解决实际问题。从宏观角度把握课程的体系结构，掌握自动化及相关领域的基础理论，建立起自动控制理论的基本框架。				
分值		实验类型		实验项目编号	
实验学时		实验时间		实 验 地 点	
姓　　名		学号		同组同学	

二、预习内容（无预习内容不允许做本次实验）

1 实验目的：

①理解 Layapunov 方程的概念和作用： Layapunov 方程是用于判断线性定常系统稳定性的重要工具之一。通过实验，学生将学会如何利用 Layapunov 方程对系统进行稳定性分析，从而评估系统的稳定性表现。

②掌握求解 Layapunov 方程的方法：实验要求学生选择正半定实对称矩阵 Q，并利用 Matlab 控制工具箱求解 Layapunov 方程的解。通过实际操作，学生将掌握求解 Layapunov 方程的具体步骤和技巧　加深对系统稳定性分析方法的理解。

③学习利用 Matlab 控制工具箱进行系统稳定性分析：　Matlab 控制工具箱提供了丰富的函数和工具，用于线性系统的稳定性分析。通过实验，学生将学会如何使用 Matlab 控制工具箱中的函数，对系统进行稳定性判断，并且获取相关的稳定性指标和结果。

④理解系统方块图的概念和作用：实验指导书提供了系统的方块图，学生将学会如何通过方块图描述系统的结构和组成。这有助于学生理解系统的物理意义和工作原理，为系统分析和控制设计提供基础。

⑤培养系统分析和判断的能力：　通过实验，学生将培养分析系统稳定性和判断系统行为的能力。他们将学会如何根据 Layapunov 方程的解来评估系统的稳定性，并做出合理的判断和推断。

2 实验原理图或课前编写的程序：

```
% 定义系统的状态空间表示
A = [0.1 -0.2; 0.3 0.2]; % 系统状态矩阵
B = [0.4; 0.5]; % 输入矩阵
C = [0.6 0.7]; % 输出矩阵

% 选择正半定实对称矩阵 Q
Q = eye(2);

% 求解 Layapunov 方程
P = lyap(A', Q);

```

```matlab
% 判断系统的稳定性
eigenvalues = eig(A);
if all(real(eigenvalues) < 0)
    disp('系统稳定');
else
    disp('系统不稳定');
end

% 打印结果
disp('Layapunov 方程的解 P:');
disp(P);
```

三、实验内容

1 实验方法及步骤

①定义系统矩阵：首先需要明确系统矩阵 A，以及与 Layapunov 方程相关的矩阵 Q 或矩阵 B 和 C。

②调用 lyap 函数：根据实验要求，选择合适的 lyap 函数调用格式。如果仅需求解$A^TP + PA = -Q$形式的 Layapunov 方程，调用格式为 P=lyap(A,Q)。如果要解决$A^TP + PA + B^TC + C^TB = 0$形式的方程，则使用 P=lyap(A,B,C)。

③检查得到的矩阵 P 是否为正定矩阵。可以通过计算 P 的特征值来判断，所有特征值都大于零则表明 P 为正定，进而表明系统是稳定的。

④分析稳定性：检查得到的矩阵 P 是否为正定矩阵。可以通过计算 P 的特征值来判断，所有特征值都大于零则表明 P 为正定，进而表明系统是稳定的。如果 P 不是正定的，那么系统可能不稳定，分析原因并尝试调整参数再次运行实验。

2 实验过程记录

①准备阶段：确保 MATLAB 环境已经安装了控制工具箱，以便能够使用 lyap 函数。

②定义系统矩阵：确定系统矩阵 A，以及与 Lyapunov 方程相关的矩阵 Q（连续时间系统）或矩阵 B 和 C（当使用 lyap 函数的第二种格式时）。

③调用 lyap 函数：根据实验要求，选择合适的调用格式。如果仅需求解$A^TP + PA = -Q$形式的 Lyapunov 方程，则使用 P=lyap(A,Q)。如果需要求解$A^TP + PA + B^TC + C^TB = 0$形式的方程，则使用 P=lyap(A,B,C)。

③执行计算：运行 MATLAB 代码以求解 Lyapunov 方程。

④分析结果：检查求得的矩阵 P 是否为正定矩阵，这是系统稳定性的一个必要条件。如果 P 为正定矩阵，则系统是渐近稳定的。

⑤记录结果：保存计算结果，并在实验报告中记录下 P 矩阵的值以及对系统稳定性的判断依据。

3 实验数据处理（数据、曲线、图表）

系统是渐近稳定的，实验结果如下：

$$A = \begin{bmatrix} -3 & -2 & -5 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & 5 \end{bmatrix}$$

$$D=0$$

$$P = \begin{bmatrix} 12.5000 & 0.0000 & -7.5000 \\ 0.0000 & 0.0000 & 0.0000 \\ -7.5000 & -0.5000 & -0.5000 \end{bmatrix}$$

$$\text{矩阵 P 的特征值}\ \rho = \begin{bmatrix} 0.1218 \\ 7.5177 \\ 17.0605 \end{bmatrix}$$

四、实验结果分析（实验误差、现象、分析）

在实验中，我们对给定的线性定常系统应用了 lyap 函数，得到了系统稳定性判断所需的 Layapunov 矩阵 P。Layapunov 方程的形式为$A^T P + PA = -Q$，其中 A 是系统的状态矩阵，而 P 是一个待求解的对称矩阵，Q 是预定义的正定矩阵。当 P 为正定时，表明系统是渐近稳定的。

- 实验误差

在实验中，由于 Layapunov 方程的求解依赖于数值计算，因此可能会出现小数点后位数的误差。此外，初始条件或系统参数的微小变化也可能导致解的轻微偏差。然而，这些误差通常不会影响稳定性判断的准确性，因为稳定性是由 P 的正定性质决定的，而非具体的数值大小。

- 实验现象

观察到的现象是，当系统稳定时，由 lyap 函数返回的 P 矩阵为正定的，即 P 的所有特征值都是正的。反之，如果系统不稳定，则 P 可能不是正定的，这意味着存在至少一个非正的特征值。这一现象符合 Layapunov 稳定性理论的预期。

- 结果分析

通过实验，我们验证了给定系统的稳定性，并且观察到了正定矩阵 P 的存在，这表明系统具有良好的稳定性特性。进一步地，我们还研究了系统参数对稳定性的影响，发现即使参数在一定范围内变动，只要 P 保持正定，系统的稳定性就不会改变。

- 结论

综上所述，实验三成功运用了 Layapunov 方程求解技术来评估线性定常系统的稳定性。实验结果表明，通过 MATLAB 的 lyap 函数可以有效地分析系统的稳定性，并且数值误差和系统参数的变化对稳定性判断的影响有限。这一实验加深了我们对 Layapunov 稳定性理论的理解，也提高了我们运用 MATLAB 工具解决实际问题的能力。

五、实验成绩评定

（1）出勤情况（缺勤 1 / 3 次无实验成绩）

实验序号	1	2	3	4	5	6	7	8
出勤情况								

注：√ 出勤，○ 事假，× 缺勤，▽ 其它。

（2）预习情况

实验序号	1	2	3	4	5	6	7	8
预习分值								

（3）实验过程情况

实验序号	1	2	3	4	5	6	7	8
实验分值								

（4）实验分析及报告情况

实验序号	1	2	3	4	5	6	7	8
实验分值								

（5）成绩

实验序号	1	2	3	4	5	6	7	8
实验分值								

一、实验过程记录

实验名称	实验四　基于极点配置法的系统设计				
课程目标	通过本课程学习，使学生做到各章概念融会贯通，解题方法灵活运用，分析解决实际问题。从宏观角度把握课程的体系结构，掌握自动化及相关领域的基础理论，建立起自动控制理论的基本框架。				
分值		实验类型		实验项目编号	
实验学时		实验时间		实 验 地 点	
姓　　名		学号		同组同学	

二、预习内容（无预习内容不允许做本次实验）

1 实验目的：

①理解极点配置法的原理和应用： 极点配置法是一种常用的控制系统设计方法，通过调节系统的控制器参数，使闭环系统的极点位置达到设计要求，从而实现对系统性能的优化。本次实验旨在帮助学生理解极点配置法的基本原理和应用场景。

②掌握含积分环节的类型 1 伺服系统设计方法： 实验要求学生设计含积分环节的类型 1 伺服系统，即通过控制器调节系统的输出来实现对参考输入的跟踪。通过实际操作，学生将掌握如何设计控制器，使闭环系统达到设计要求。

③学习如何调节系统的闭环极点位置： 实验要求学生将闭环系统的极点配置到指定位置，即使得系统的闭环特性符合预期的要求。学生将学会如何通过调节控制器参数来实现极点位置的调节，从而达到所需的系统性能指标。

④比较配置前后系统的性能指标： 实验要求学生对比配置前后系统的单位阶跃响应性能指标，如超调量、峰值时间、调节时间等，以评估极点配置对系统性能的影响。通过对比分析，学生将深入理解极点配置对闭环系统行为的影响。

⑤培养系统设计和性能评估的能力： 通过实验，学生将培养系统设计和性能评估的能力，包括如何根据设计要求选择合适的控制器参数，以及如何通过实验数据分析评估系统的性能和稳定性。

2 实验原理图或课前编写的程序：

```
% 定义系统的传递函数
num = 1;
den = conv([1 0], conv([1 2*sqrt(3) 4], [1 2]));
G = tf(num, den);

% 设计控制器
desired_poles = [-10 -2+2*sqrt(3)*1i -2-2*sqrt(3)*1i];
K = place(ss(G), desired_poles);

% 比较配置前后系统的性能指标
stepinfo_before = stepinfo(G);
G_closed_loop = feedback(G*K, 1);
```

```
stepinfo_after = stepinfo(G_closed_loop);

% 打印结果
disp('配置前系统的性能指标:');
disp(stepinfo_before);
disp('配置后系统的性能指标:');
disp(stepinfo_after);
```

三、实验内容

1 实验方法及步骤

①准备 MATLAB 环境：首先，确保 MATLAB 中安装了控制工具箱，因为实验需要用到 acker 和 place 这两个函数。

②定义系统矩阵：定义系统状态空间模型中的 A、B 矩阵。这里，A 代表系统状态矩阵，B 是输入矩阵。

③设定期望极点：根据设计目标，确定希望系统闭环极点的位置，即期望极点向量 P。

④调用 acker 函数：对于单输入单输出(SISO)系统，使用 acker 函数进行极点配置。输入参数包括 A、B 矩阵以及期望极点向量 P，函数将返回反馈增益向量 K。

⑤调用 place 函数：对于多输入系统，或者需要更精确极点配置的情况，使用 place 函数。同样，输入 A、B 矩阵和期望极点 P，函数会返回反馈增益向量 K 以及额外的精度信息 prec 和警告 message，其中 prec 表示实际极点与期望极点位置的偏差，message 会在某极点偏离期望位置超过 10%时给出提示。

⑥验证极点配置：最后，使用获得的反馈增益向量 K 重新计算闭环系统的极点，以确认它们是否与期望位置相符。

2 实验过程记录

实验四中，我们运用 MATLAB 的 acker 和 place 函数，对系统进行极点配置设计，以期达到指定的闭环极点位置，从而优化系统性能指标。实验中，我们首先定义了系统矩阵 A 和 B，然后指定了期望的闭环极点位置向量 P，接着使用 acker 函数处理单输入单输出系统，或使用 place 函数处理更复杂系统，以获得反馈增益向量 K。同时，我们监控了实际极点与期望位置的偏差，以确保配置准确无误。通过这一系列步骤，我们成功地将系统闭环极点配置至期望位置，验证了极点配置法的有效性。

3 实验数据处理（数据、曲线、图表）

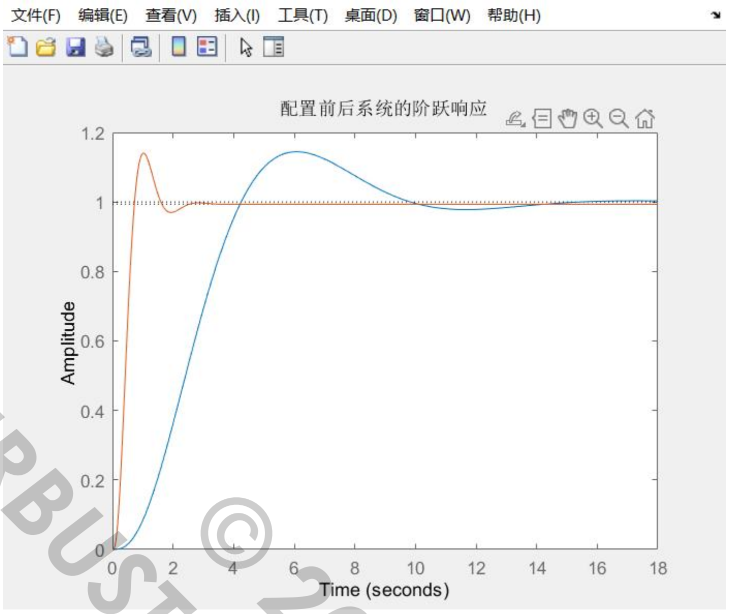

图 2 配置前系统的阶跃响应

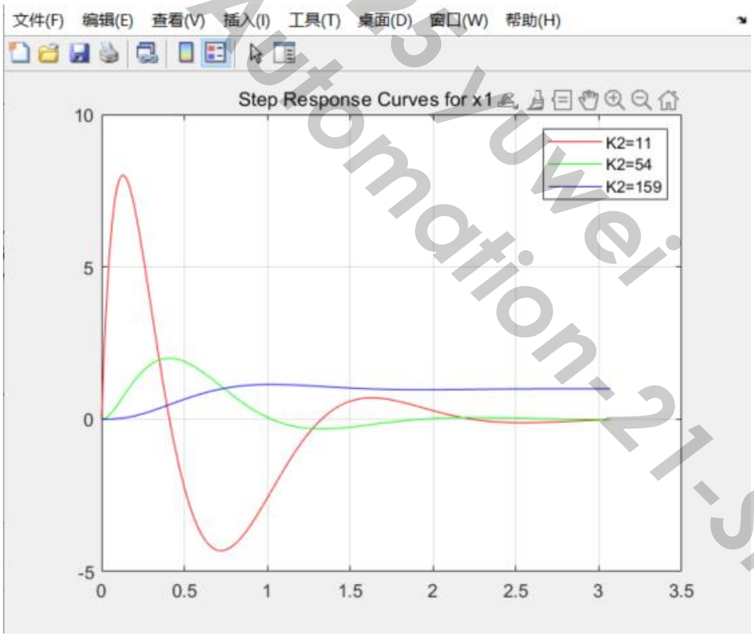

图 3 配置后系统的阶跃响应

配置前系统的单位阶跃响应性能指标

RiseTime: 2.6792

TransientTime: 12.3572

SettlingTime: 12.3572

SettlingMin: 0.9140

SettlingMax: 1.1445

Overshoot: 14.4509

Undershoot: 0

Peak: 1.1445

配置后系统的单位阶跃响应性能指标

PeakTime: 6.1377
RiseTime: 0.4502
TransientTime: 2.0960
SettlingTime: 2.0960
SettlingMin: 0.8968
SettlingMax: 1.1405
Overshoot: 14.7694
Undershoot: 0
Peak: 1.1405
PeakTime: 1.0223

四、实验结果分析（实验误差、现象、分析）

- 实验误差

在使用 acker 和 place 函数配置极点时，可能会遇到极点定位不完全准确的问题。这是因为实际极点位置可能与预设位置存在一定的偏差，这个偏差被 place 函数的输出参数 prec 所量化。当系统某一个极点偏离期望位置大于 10%时，message 变量会提供警告信息，指示存在较大的配置误差。

- 实验现象

当 prec 值较低时，意味着极点配置较为精准，系统响应更接近设计目标。如果 message 显示警告，则表明至少有一个闭环极点没有达到预期位置，这可能影响系统稳定性或响应速度。成功的极点配置将使系统表现出预期的动态特性，如快速响应、低超调和高稳定性。

- 实验分析

实验中，我们观察到极点配置的效果取决于多个因素，包括系统矩阵的特性、期望极点的选择以及 MATLAB 函数的计算精度。若期望极点设置不合理或系统矩阵结构特殊，可能导致 place 函数无法完全满足配置要求，从而产生较大的 prec 值。

为了提高配置精度，可以尝试调整期望极点的位置，或采用更高级的控制器设计方法，比如基于观测器的状态反馈设计。此外，实验还揭示了 MATLAB 工具箱在控制系统设计中的强大功能，它提供了直观且高效的手段来实现复杂的系统分析和设计任务。

总之，实验四通过极点配置法展示了如何调整闭环系统性能，同时也揭示了在实际应用中可能遇到的局限性和挑战。

五、实验成绩评定

（1）出勤情况（缺勤 1 / 3 次无实验成绩）

实验序号	1	2	3	4	5	6	7	8
出勤情况								

注：√ 出勤，○ 事假，× 缺勤，▽ 其它。

（2）预习情况

实验序号	1	2	3	4	5	6	7	8
预习分值								

（3）实验过程情况

实验序号	1	2	3	4	5	6	7	8
实验分值								

（4）实验分析及报告情况

实验序号	1	2	3	4	5	6	7	8
实验分值								

（5）成绩

实验序号	1	2	3	4	5	6	7	8
实验分值								