|  |
| --- |
| 吉林大学 |
| 《通信系统仿真》实验报告 |
| 实验：通信系统仿真实验 |

|  |
| --- |
| 姓名：杨宇烨 学号：20210924  2024-5-10 |

# 实验一　Matlab语言初识和数值计算

## 实验内容

1.练习数据和符号的输入方式，将前面的命令在命令窗口中执行通过；

2.输入A=[7 1 5；2 5 6；3 1 5]，B=[1 1 1; 2 2 2; 3 3 3]，在命令窗口中执行下列表达式，掌握其含义：

A(2,3)、A(:,2)、A(3,:)、A(:,1:2:3)、A(:,3).\*B(:,2)、A(:,3)\*B(2,:)、A\*B、A.\*B、A^2、A.^2、B/A、B./A

3.输入C=1:2:20，则C（i）表示什么？其中i=1,2,3,…,10；

4.查找已创建变量的信息，删除无用的变量；

5.在命令窗口中键入表达式，并求时的值。

6.做出下列函数的图像：

（1），（分别用plot、fplot）

（2）（用参数方程）

（3），，（绘网格图、二维和三维等高线图）

(4) 在同一图形窗口中，画出四幅不同图形（用subplot命令）（）：

，，，

7.在同一坐标系中画出的图像。

## 实验过程（代码）

%% 1-2

A=[7 1 5;2 5 6;3 1 5],B=[1 1 1; 2 2 2; 3 3 3]

% 获取第2行第3列的元素

result1 = A(2,3) % 输出：6

% 获取矩阵A的第2列

result2 = A(:,2) % 输出：[1; 5; 1]

% 获取矩阵A的第3行

result3 = A(3,:) % 输出：[3 1 5]

% 获取矩阵A的第1和第3列

result4 = A(:,1:2:3) % 输出：[7 5; 2 6; 3 5]

% 点乘操作，A的第3列和B的第2列

result5 = A(:,3) .\* B(:,2) % 输出：[5; 12; 15]

% 矩阵乘法，A的第3列乘以B的第2行

result6 = A(:,3) \* B(2,:) % 输出：[10 10 10; 12 12 12; 15 15 15]

% 矩阵乘法，A与B

result7 = A \* B % 输出：[22 22 22; 35 35 35; 23 23 23]

% 点乘操作，A与B

result8 = A .\* B % 输出：[7 1 5; 4 10 12; 9 3 15]

% 矩阵幂运算，A与自身相乘

result9 = A^2 % 输出：[58 38 70; 38 43 67; 29 23 48]

% 元素幂运算，A的每个元素平方

result10 = A.^2 % 输出：[49 1 25; 4 25 36; 9 1 25]

% 矩阵除法，B除以A

result11 = B / A % 输出：[0.2361 0.2778 -0.1667; 0.4444 0.3889 0.1667; 0.3194 0.3333 -0.0556]

% 元素除法，B的每个元素除以A的对应元素

result12 = B ./ A % 输出：[0.1429 1 0.2; 1 0.4 0.3333; 1 3 0.2]

%% 1-3

C = 1:2:20; % 生成 [1 3 5 7 9 11 13 15 17 19]

%% 1-4

% 创建一些变量

A = [1, 2, 3];

B = magic(3);

C = 'Hello, World!';

% 查看所有变量信息

whos % 显示当前工作区中所有变量的详细信息

% 假设我们不需要变量 C

clear C % 删除变量 C

% 查看剩余变量信息

whos % 显示剩余的变量信息

%% 1-5

% 定义变量 x 和 y

x = 2;

y = 4;

% 计算表达式

z = x^2 + exp(x + y) - y \* log(x) - 3;

% 显示结果

disp("z 的值是: " + z);

%% 1-6-1

x = -2:0.01:2; % 定义x的范围

y = x.^2 .\* sin(x.^2 - x - 2); % 计算y(x)

% 使用plot绘制图像

figure;

plot(x, y);

title('y(x) = x^2 \* sin(x^2 - x - 2) (Using plot)');

xlabel('x');

ylabel('y');

% 使用fplot绘制图像

figure;

fplot(@(x) x.^2 .\* sin(x.^2 - x - 2), [-2, 2]);

title('y(x) = x^2 \* sin(x^2 - x - 2) (Using fplot)');

xlabel('x');

ylabel('y');

%% 1-6-2

% 定义参数

t = linspace(0, 2\*pi, 100);

% 椭圆参数方程

x = 2 \* cos(t);

y = 4 \* sin(t);

% 绘制椭圆

figure;

plot(x, y);

title('x^2/4 + y^2/16 = 1');

xlabel('x');

ylabel('y');

axis equal; % 确保轴的比例相同

%% 1-6-3

% 定义x和y的范围

x = -3:0.1:3;

y = -2:0.1:2;

% 创建网格

[X, Y] = meshgrid(x, y);

% 定义函数z

Z = (X.^2 - 2 \* X) .\* exp(-(X.^2 + Y.^2 + X .\* Y));

% 绘制3D网格图

figure;

mesh(X, Y, Z);

title('3D Mesh Grid of z=(x^2-2x) \* exp(-(x^2 + y^2 + xy))');

xlabel('x');

ylabel('y');

zlabel('z');

% 绘制二维等高线图

figure;

contour(X, Y, Z, 20); % 20个等高线

title('2D Contour of z=(x^2-2x) \* exp(-(x^2 + y^2 + xy))');

xlabel('x');

ylabel('y');

% 绘制三维等高线图

figure;

contour3(X, Y, Z, 20); % 20个等高线

title('3D Contour of z=(x^2-2x) \* exp(-(x^2 + y^2 + xy))');

xlabel('x');

ylabel('y');

zlabel('z');

%% 1-6-4

x = linspace(0, 2\*pi, 100);

% 定义不同的函数

y1 = cos(x);

y2 = sin(x - pi/2);

y3 = x.^2 .\* cos(x - pi);

y4 = exp(sin(x));

% 创建子图

figure;

% 子图1

subplot(2, 2, 1); % 2x2 网格，位置 1

plot(x, y1);

title('y1 = cos(x)');

% 子图2

subplot(2, 2, 2); % 2x2 网格，位置 2

plot(x, y2);

title('y2 = sin(x - pi/2)');

% 子图3

subplot(2, 2, 3); % 2x2 网格，位置 3

plot(x, y3);

title('y3 = x^2 \* cos(x - pi)');

% 子图4

subplot(2, 2, 4); % 2x2 网格，位置 4

plot(x, y4);

title('y4 = exp(sin(x))');

%% 1-7

% 定义x的范围

x = linspace(0, 2, 100); % 从0到2，100个点

% 计算y的不同函数

y1 = sqrt(x); % y = √x

y2 = x.^2; % y = x^2

y3 = nthroot(x, 3); % y = ∛x

y4 = x.^3; % y = x^3

y5 = x; % y = x

% 绘制图像

figure;

hold on; % 确保在同一图中叠加绘制

plot(x, y1, 'b-', 'LineWidth', 1.5); % 蓝色实线

plot(x, y2, 'r--', 'LineWidth', 1.5); % 红色虚线

plot(x, y3, 'g-.', 'LineWidth', 1.5); % 绿色点划线

plot(x, y4, 'm:', 'LineWidth', 1.5); % 紫色点点线

plot(x, y5, 'k-', 'LineWidth', 1.5); % 黑色实线

% 设置图表标题和轴标签

title('不同函数的图像');

xlabel('x');

ylabel('y');

% 添加图例

legend('y = √x', 'y = x^2', 'y = ∛x', 'y = x^3', 'y = x');

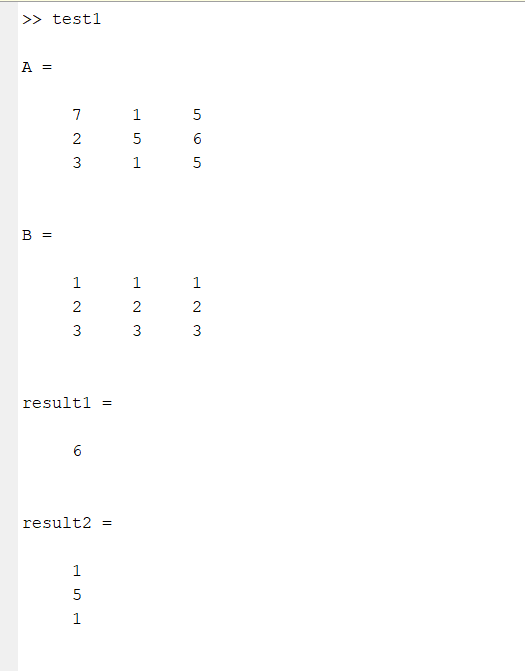
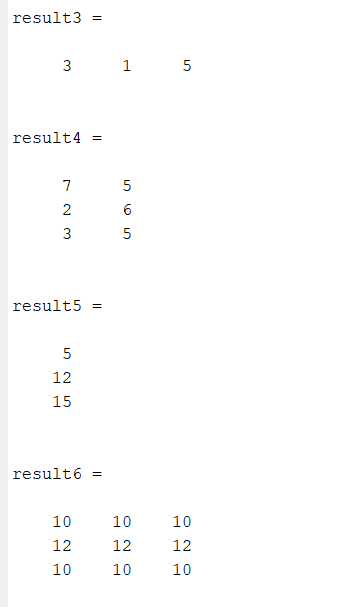
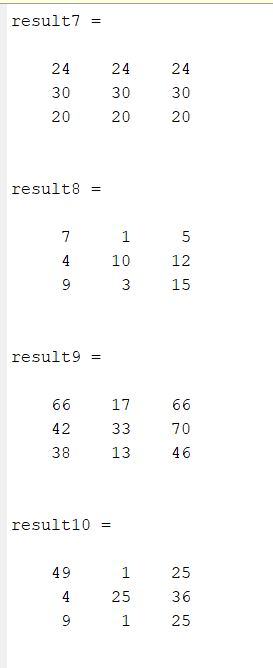
% 显示网格

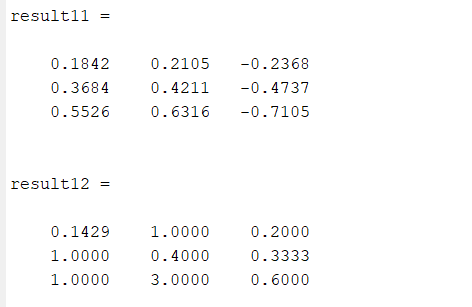
grid on;

hold off; % 结束绘制

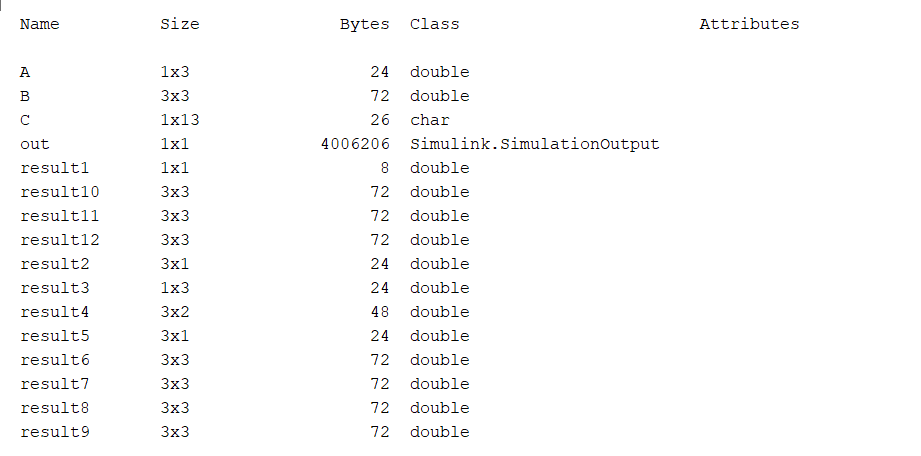
## 实验结果

1. 矩阵查看

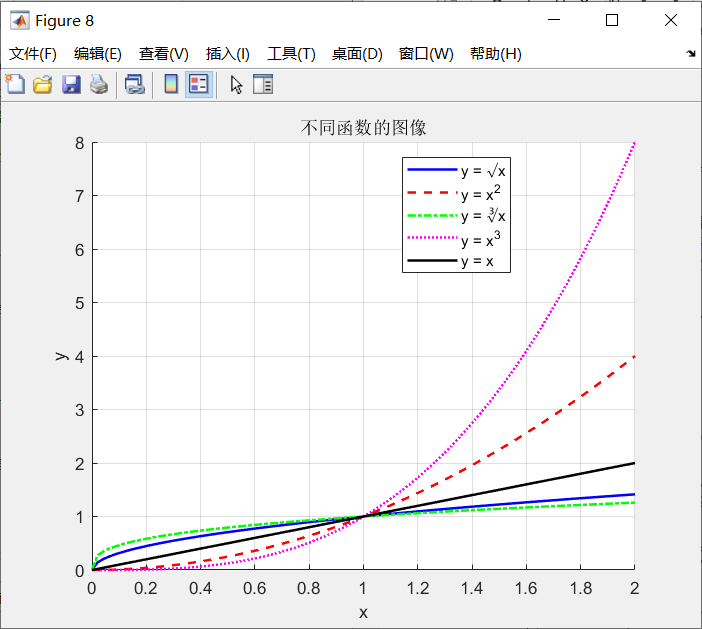
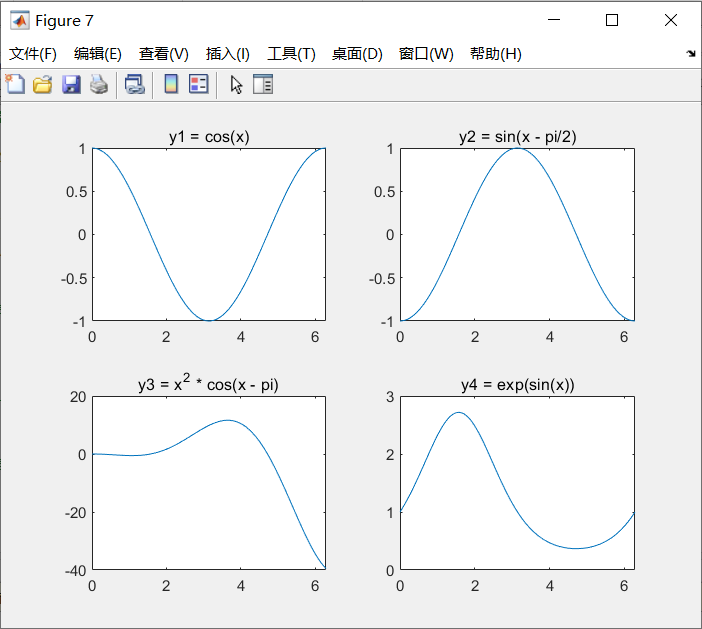
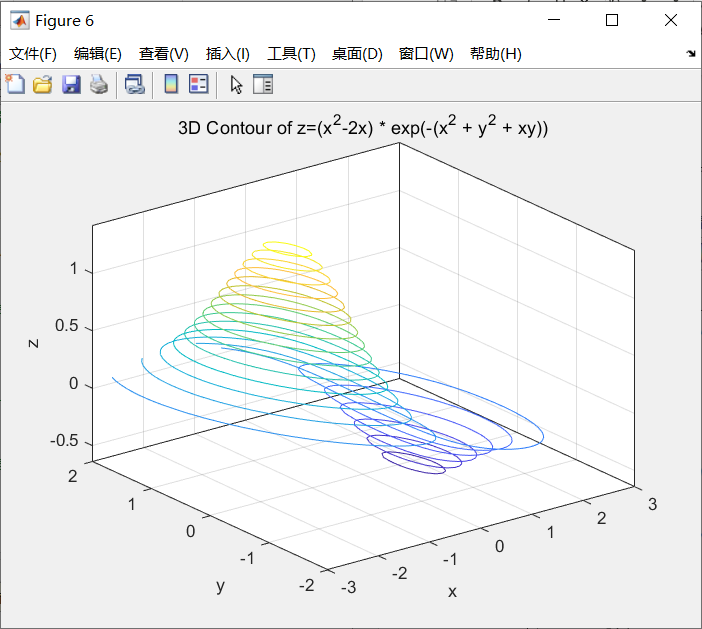
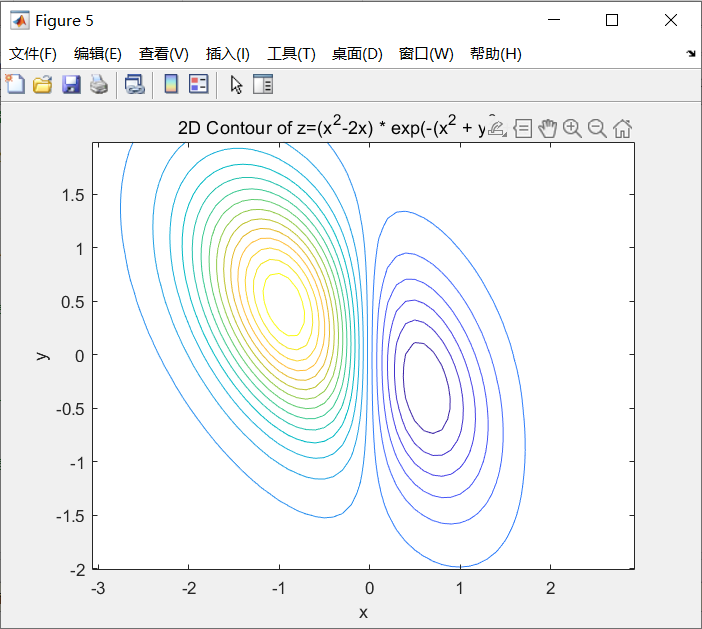
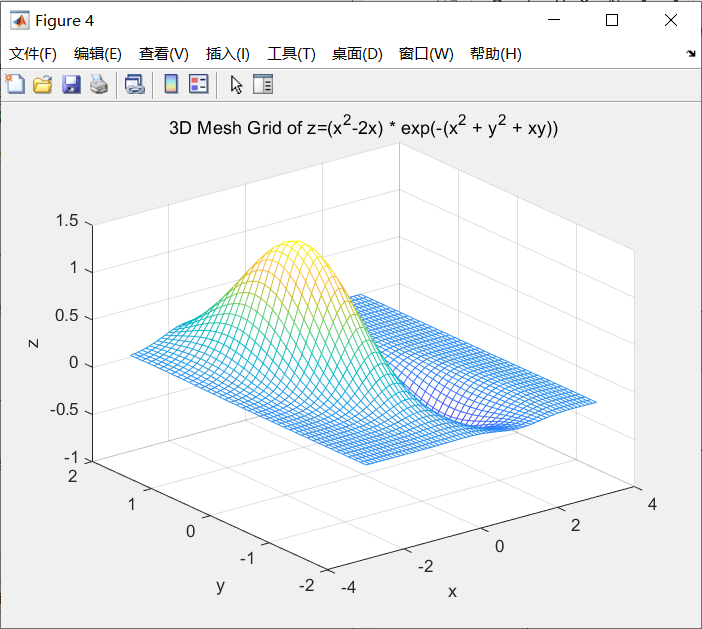
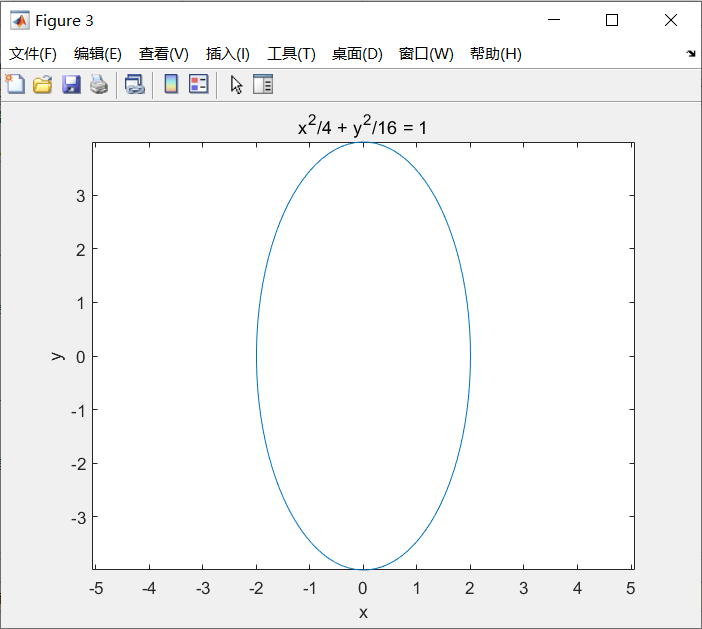
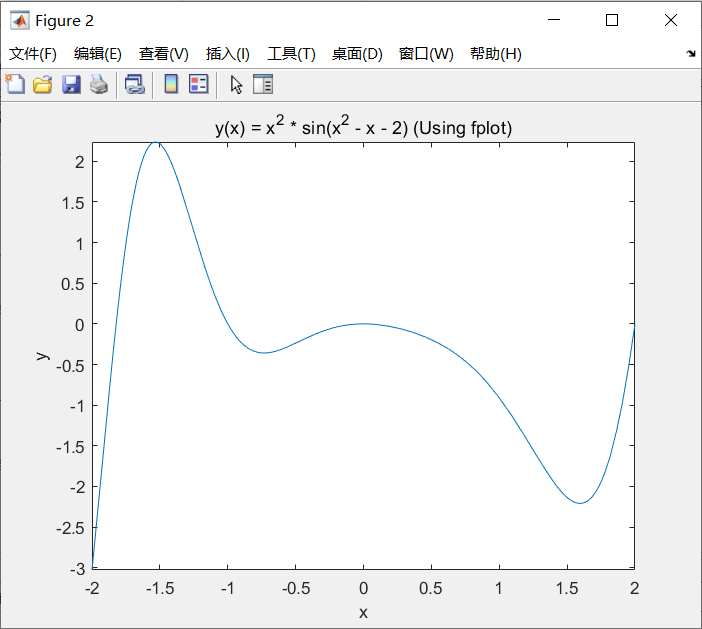
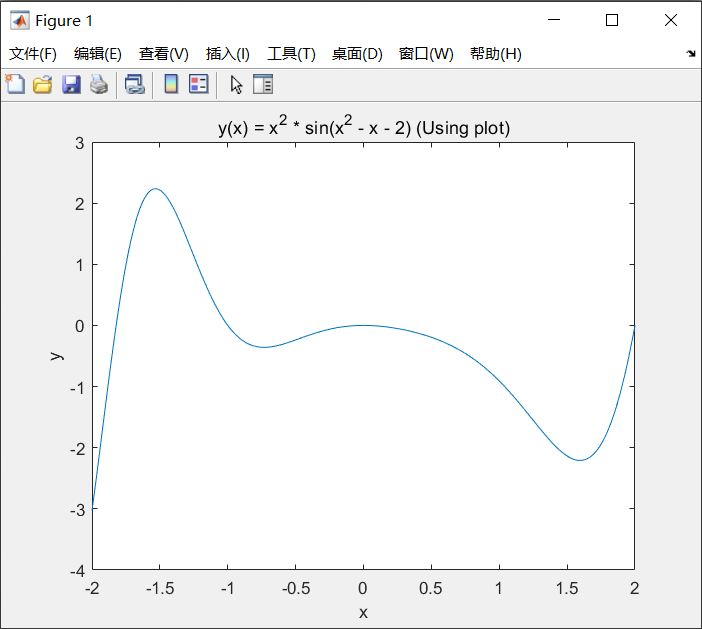
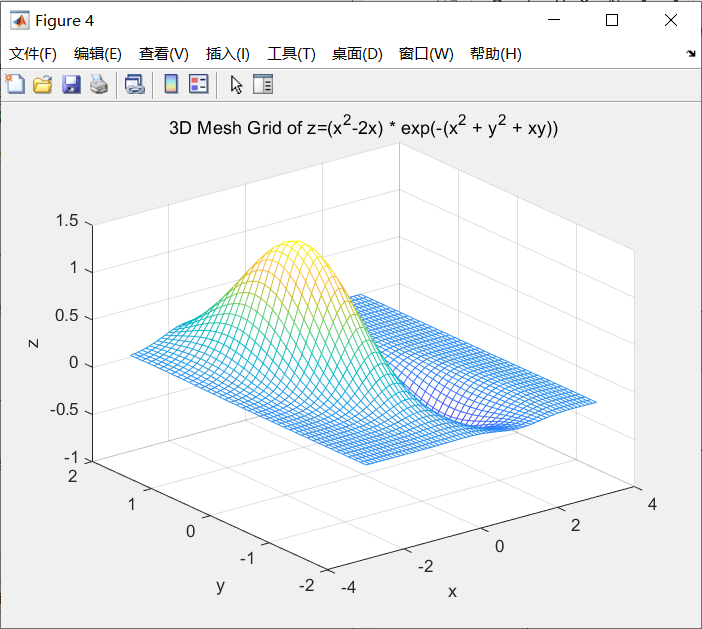
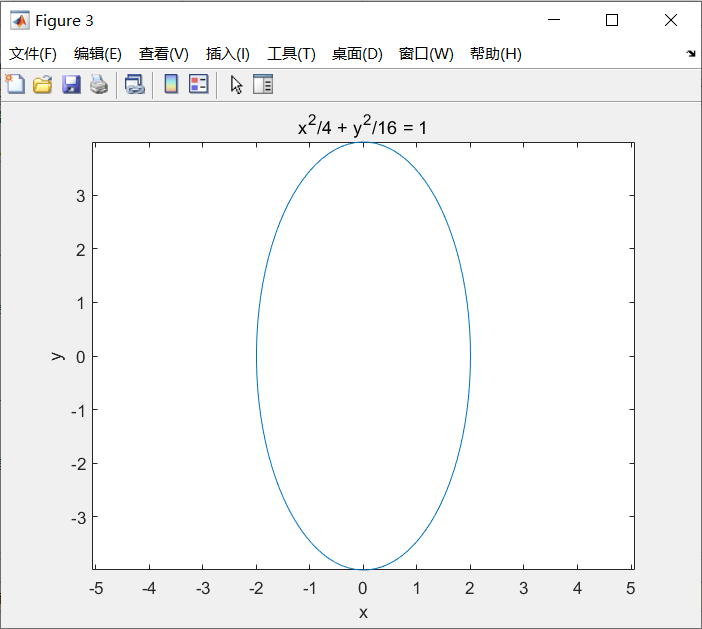
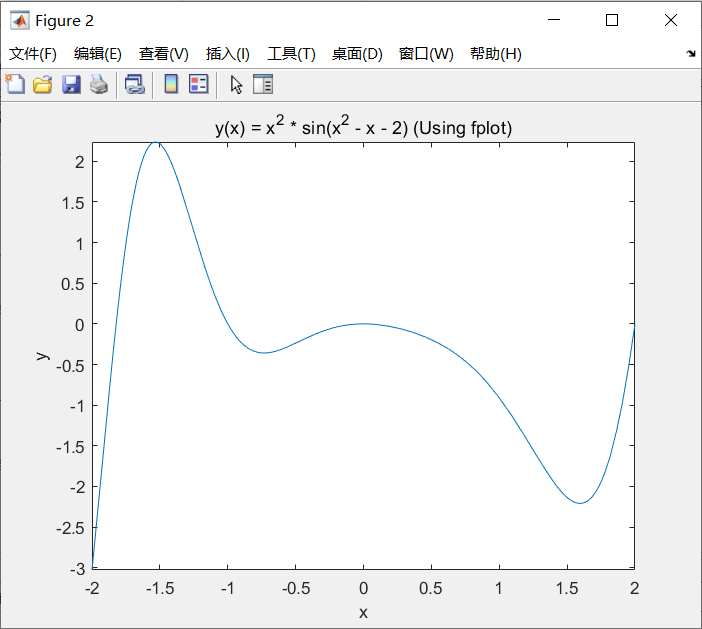
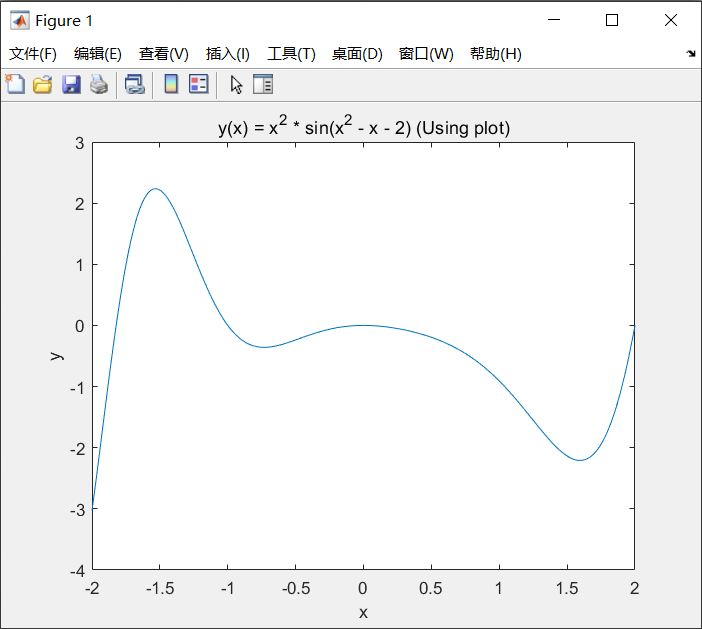
  



1. 查找已创建变量的信息



3.函数绘图



## 结果分析

完成实验

## 心得体会

学习了matlab的入门用法

# 实验二　Matlab程序设计

## 一、实验内容

1.编写程序，计算1+3+5+7+…+(2n+1)的值（用input语句输入n 值）。

2.编写分段函数 的函数文件，存放于文件ff.m中，计算出，，的值。

3.利用矩阵的初等变换及函数rank、函数inv求下列矩阵的秩：

(1) ，求 Rank(A)=?

4.计算

（1）A=求

(2) 求

5.用循环语句编写函数M文件计算的值，其中为输入变量（本实验取），的近似值可用下式表示：。

6.已知多项式，，求：

（1）的根；

(2) 在闭区间[-1,2]上的最小值；

（3），和；

（4）的导数。

7.程序设计：相传古代印度国王要褒奖他的聪明能干的宰相达依尔（国际象棋发明者），问他要什么？达依尔回答：“陛下只要在国际象棋棋盘的第一个格子上放一粒麦子，第二个格子上放二粒麦子，以后每个格子的麦子数都按前一格的两倍计算。如果陛下按此法给我64格的麦子，就感激不尽，其他什么也不要了。”国王想：“这还不容易！”让人扛了一袋麦子，但很快用光了，再扛出一袋还不够，请你为国王算一下共要给达依尔多少小麦？（1袋小麦约1.4×108 粒）。

8．程序设计：公元前五世纪我国古代数学家张丘建在《算经》一书中提出了“百鸡问题”：鸡翁一值钱五，鸡母一值钱三，鸡雏三值钱一。百钱买百鸡，问鸡翁、母、雏各几何？

## 二、实验过程（代码）

%% 2-1

% 输入 n 的值

n = input('请输入 n 的值: ');

% 计算等差数列的和

% 首项 a = 1，公差 d = 2，项数 m = n + 1

% 求和公式: S = m/2 \* (2 \* a + (m - 1) \* d)

m = n + 1; % 项数

a = 1; % 首项

d = 2; % 公差

S = m / 2 \* (2 \* a + (m - 1) \* d); % 求和

% 输出结果

disp(['1 + 3 + 5 + ... + (2 \* ', num2str(n), ' + 1) = ', num2str(S)]);

%% 2-2

% 计算 f(-3)

result1 = ff(-3);

disp(['f(-3) = ', num2str(result1)]);

% 计算 f(√2)

result2 = ff(sqrt(2));

disp(['f(√2) = ', num2str(result2)]);

% 计算 f(∞)

result3 = ff(Inf);

disp(['f(∞) = ', num2str(result3)]);

%% 2-3

% 定义矩阵 A

A = [1 -6 3 2; 3 -5 4 0; -1 -11 2 4];

% 求矩阵 A 的秩

rank\_A = rank(A); % 使用 MATLAB 的 rank 函数

% 输出结果

disp(['Rank(A) = ', num2str(rank\_A)]);

%% 2-4-1

% 输入旋转角度

t = input('请输入旋转角度 t（弧度）: ');

% 构建旋转矩阵

A = [cos(t), -sin(t); sin(t), cos(t)];

% 计算五次幂 A^5

theta = 5 \* t; % 旋转角度乘以 5

A\_5 = [cos(theta), -sin(theta); sin(theta), cos(theta)];

% 输出结果

disp('A^5 =');

disp(A\_5);

%% 2-4-2

% 定义矩阵 B

B = [

1, 2, 1, 0;

6, 2, 4, 1;

0, 2, 1, 0;

3, 1, 4, 1

];

% 求逆矩阵 B^(-1)

B\_inv = inv(B); % 使用 inv 函数

% 输出结果

disp('B^(-1) =');

disp(B\_inv);

%% 2-5

% 输入变量

x = 1;

n = 100;

% 计算 e^x 的近似值

e\_approx = approx\_exp(x, n);

% 输出结果

disp(['e^', num2str(x), ' 的近似值是：', num2str(e\_approx)]);

%% 2-6-1

% 多项式 f(x) 的系数

f\_coeff = [3, -1, 2, 1, 1, 3];

% 求 f(x) 的根

f\_roots = roots(f\_coeff);

% 输出根

disp('f(x) 的根是:');

disp(f\_roots);

%% 2-6-2

% 多项式 g(x) 的系数

g\_coeff = [1/3, 1, -3, -1];

% 定义多项式 g(x)

g = @(x) polyval(g\_coeff, x);

% 求 g(x) 在区间 [-1, 2] 上的最小值

[x\_min, g\_min] = fminbnd(g, -1, 2);

% 输出最小值

disp(['g(x) 在 [-1, 2] 上的最小值是: ', num2str(g\_min), ', 在 x = ', num2str(x\_min), ' 处。']);

%% 2-6-3

% 定义多项式 f(x) 和 g(x) 的系数

f\_coeff = [3, -1, 2, 1, 1, 3];

g\_coeff = [1/3, 1, -3, -1];

% 确保系数长度一致，通过填充 0

len\_f = length(f\_coeff);

len\_g = length(g\_coeff);

if len\_f > len\_g

g\_coeff = [zeros(1, len\_f - len\_g), g\_coeff];

elseif len\_g > len\_f

f\_coeff = [zeros(1, len\_g - len\_f), f\_coeff];

end

% 求 f(x) + g(x)

fg\_sum = f\_coeff + g\_coeff; % 多项式加法

% 输出结果

disp('f(x) + g(x) 的系数是:');

disp(fg\_sum);

% 计算 f(x) ⋅ g(x)

fg\_prod = conv(f\_coeff, g\_coeff);

% 输出结果

disp('f(x) ⋅ g(x) 的系数是:');

disp(fg\_prod);

% 计算 f(x) / g(x)

% 确保 g\_coeff 的首项非零

if g\_coeff(1) == 0

% 找到第一个非零系数

idx = find(g\_coeff ~= 0, 1);

% 去掉前面的零

g\_coeff = g\_coeff(idx:end);

end

% 使用 deconv 进行多项式除法

[fg\_quotient, fg\_remainder] = deconv(f\_coeff, g\_coeff);

% 输出结果

disp('f(x) / g(x) 的商是:');

disp(fg\_quotient);

disp('f(x) / g(x) 的余数是:');

disp(fg\_remainder);

% 输出结果

disp('f(x) / g(x) 的商是:');

disp(fg\_quotient);

disp('f(x) / g(x) 的余数是:');

disp(fg\_remainder);

%% 2-6-4

% 求 f(x) 的导数

f\_derivative = polyder(f\_coeff);

% 输出 f(x) 的导数

disp('f(x) 的导数是:');

disp(f\_derivative);

%% 2-7

% 计算64个格子的麦子总数

total\_grains = 2^64 - 1;

% 每袋小麦约含 1.4 × 10^8 粒

grains\_per\_bag = 1.4e8;

% 计算需要多少袋小麦

bags\_needed = total\_grains / grains\_per\_bag;

% 输出结果

disp(['总共需要小麦数量: ', num2str(total\_grains)]);

disp(['需要多少袋小麦: ', num2str(bags\_needed)]);

%% 2-8

% 程序计算满足条件的鸡翁、鸡母、鸡雏的数量

% 鸡翁5钱一只，鸡母3钱一只，鸡雏三只1钱

% 为了满足"百钱买百鸡"，我们可以分别枚举鸡翁和鸡母的数量

% 鸡翁数量最大不能超过20，因为20只公鸡要花100钱

for rooster = 0:20

% 鸡母数量最大不能超过33，因为33只母鸡刚好100钱

for hen = 0:33

% 根据鸡翁和鸡母数量计算鸡雏数量

chick = 100 - rooster - hen; % 剩余的鸡数

% 根据鸡雏数量计算所需的费用

total\_cost = rooster \* 5 + hen \* 3 + (chick / 3);

% 检查费用是否正好100钱，并且鸡雏数量是3的倍数

if total\_cost == 100 && mod(chick, 3) == 0

fprintf('鸡翁: %d, 鸡母: %d, 鸡雏: %d\n', rooster, hen, chick);

end

end

end

%% 2-2

function y = ff(x)

% 分段函数 f(x)={■(x&0≤x<1@2-x&1≤x≤2@0&其它)┤

if x >= 0 && x < 1

y = x;

elseif x >= 1 && x <= 2

y = 2 - x;

else

y = 0; % 对于其它情况，返回 0

end

end

%% 2-5

function result = approx\_exp(x, n)

% 计算 e^x 的近似值，使用泰勒级数展开

% 输入 x 为自变量，n 为展开的最大阶数

% 初始化结果

result = 1; % 第一项是 1

term = 1; % 用于计算每一项的中间变量

% 循环计算每一项，并累加

for k = 1:n

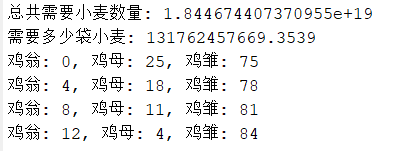
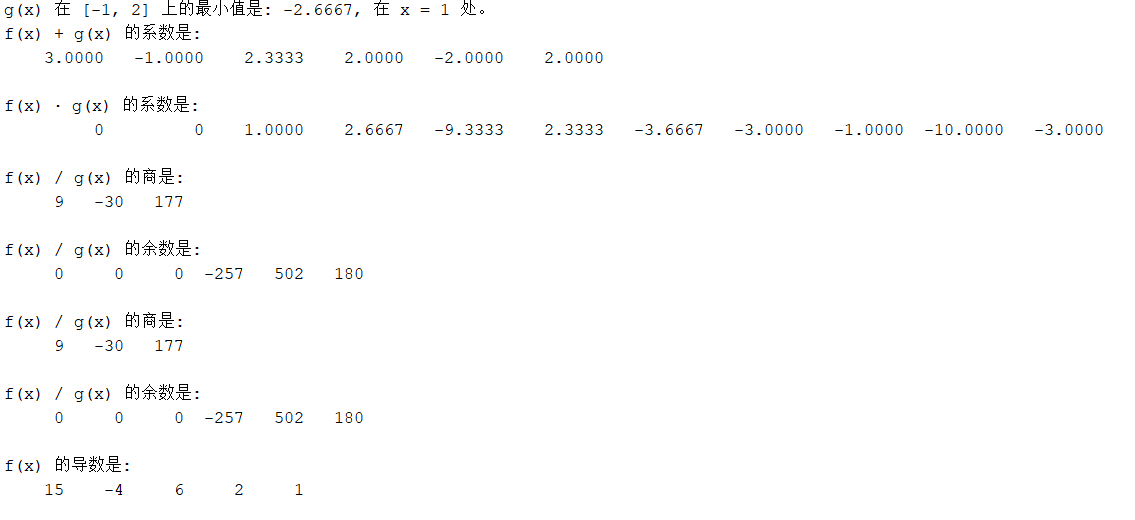
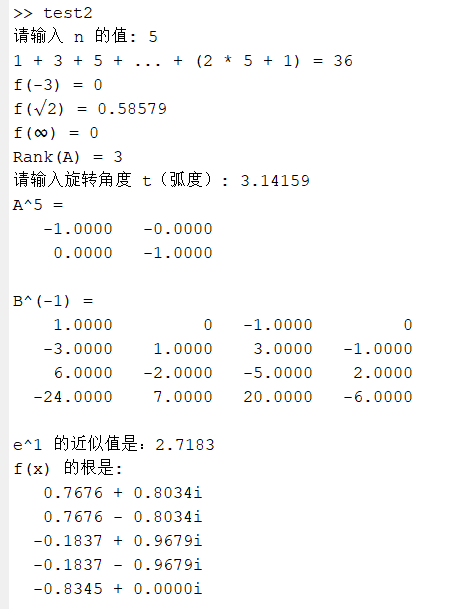
term = term \* x / k; % 计算第 k 项

result = result + term; % 累加到结果

end

end

## 三、实验结果



## 四、结果分析

符号计算是MATLAB的一项强大功能，可以处理符号表达式和进行代数运算。这在处理数学问题和工程计算时非常有用。

## 五、心得体会

通过本次实验，我掌握了MATLAB的流程控制结构，包括**if-else**、**for**、**while**和**switch-case**等。这些结构为程序提供了多样化的控制流程，可以满足不同的逻辑需求。此外，我了解了MATLAB中的函数调用和参数传递过程，并创建了简单的函数。函数的使用增强了代码的模块化和可复用性。

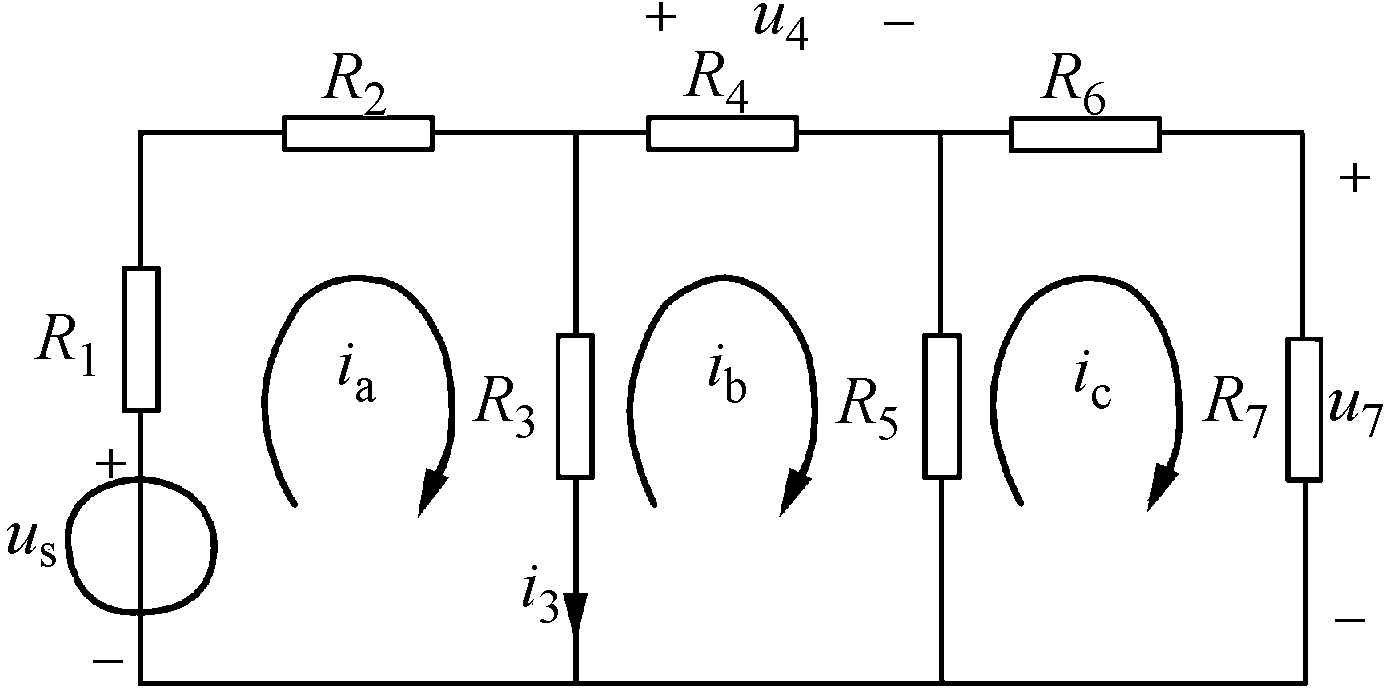
# 实验三　在电路中Matlab的应用

## 一、实验内容

1、在如图所示的梯形直流电路中，已知：,问：

（1）如果，求；

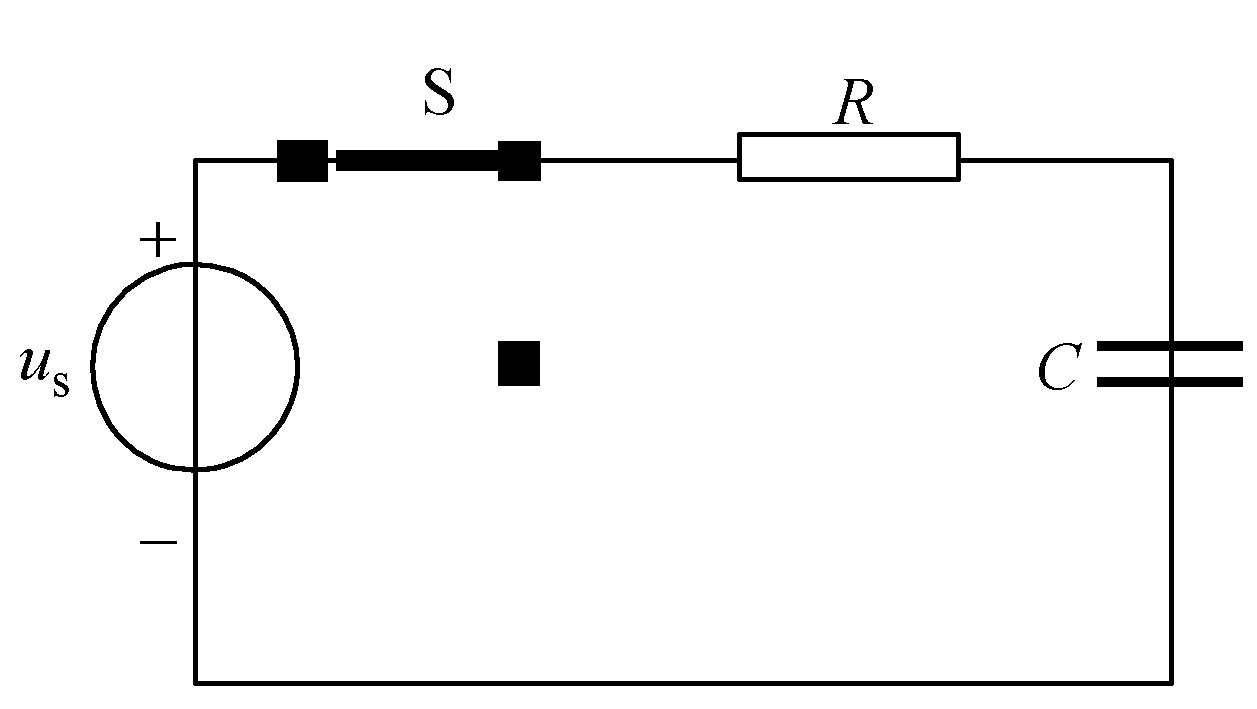
（2）如果，求。



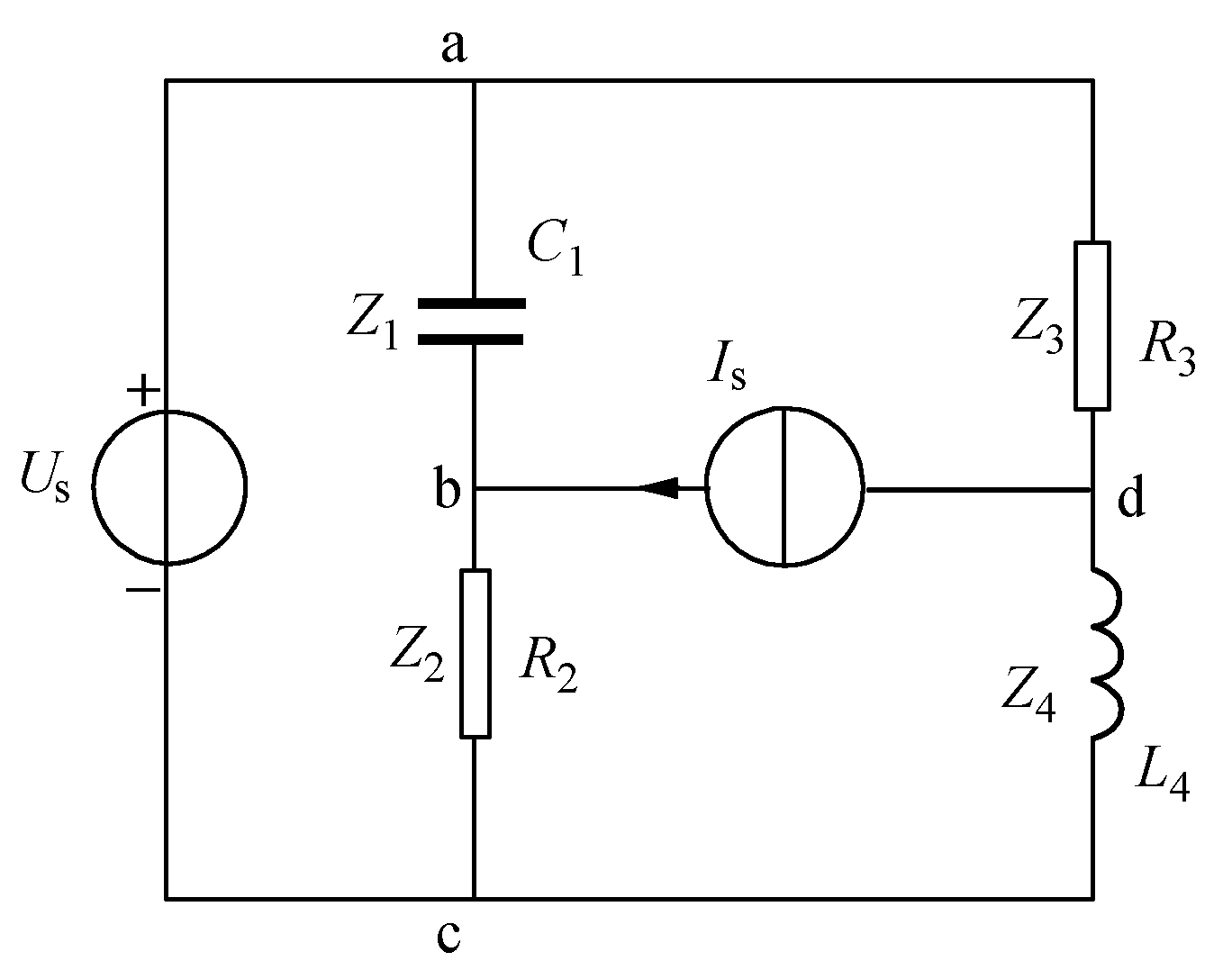
2、如下图所示的电路，在t<0时，开关S位于“1”，电路已处于稳态，t=0时，开关S闭合到“2”，求和的响应，并画出它们的波形。



3、已知：R＝2Ω，C＝0.5F，电容初始电压，激励的正弦电压,其中。当t=0时，开关s闭合，求电容电压的全响应，区别其暂态响应和稳态响应，并画出波形。



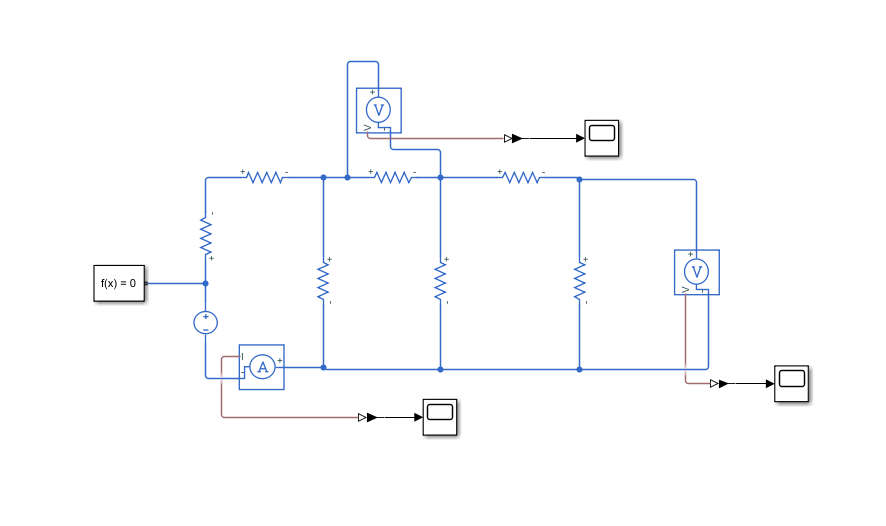
4、如下图所示电路，已知；，，求b，d两点之间的电压U (t)。

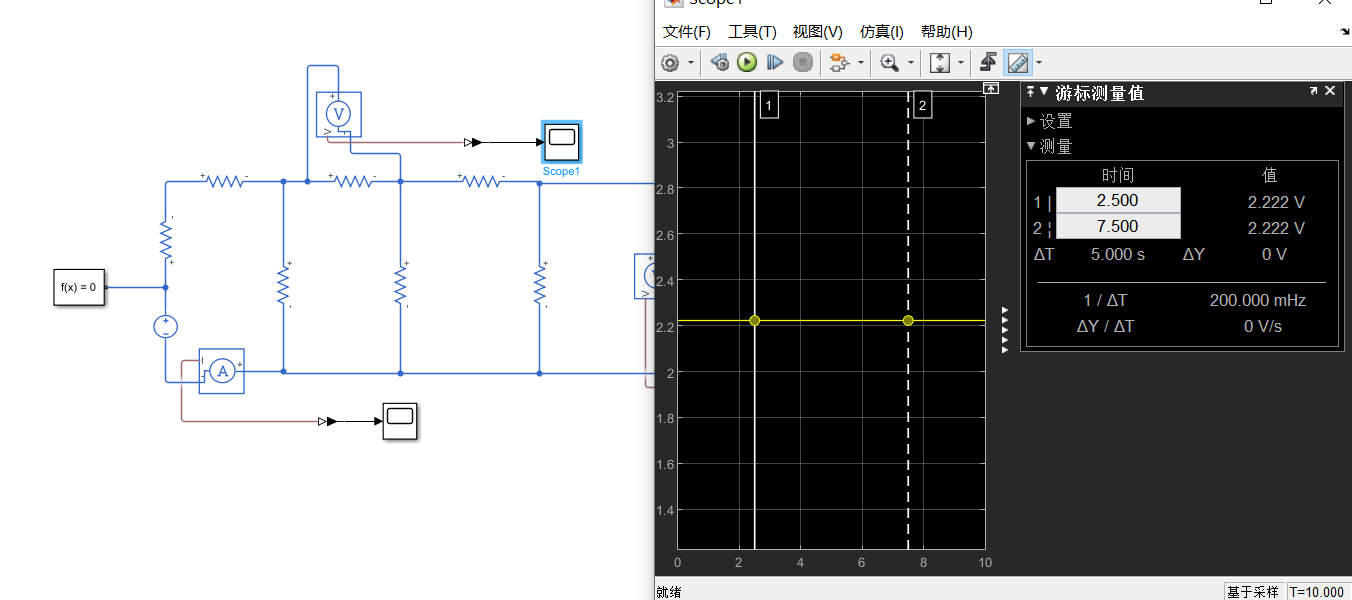


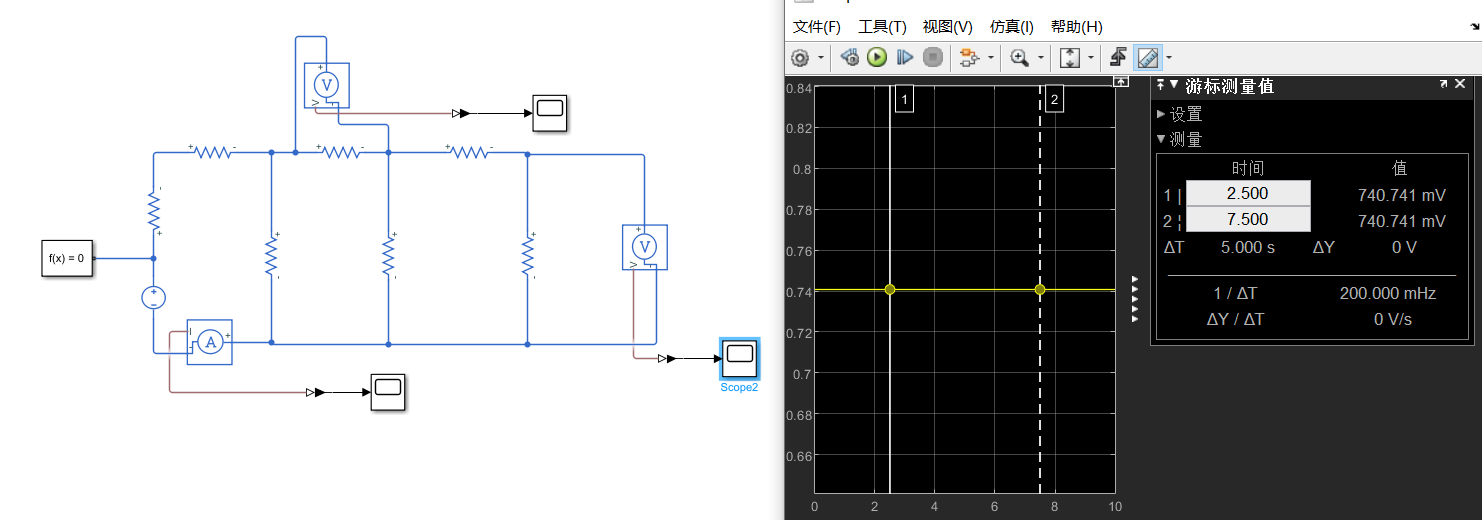
## 二、实验电路以及结果

### （1-1）

电路（1）搭建：



U4: 

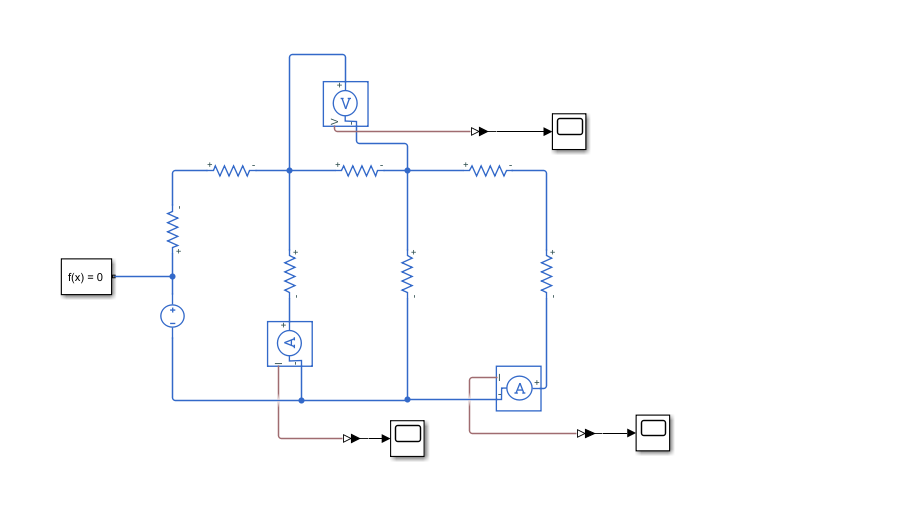
U7：

Is:

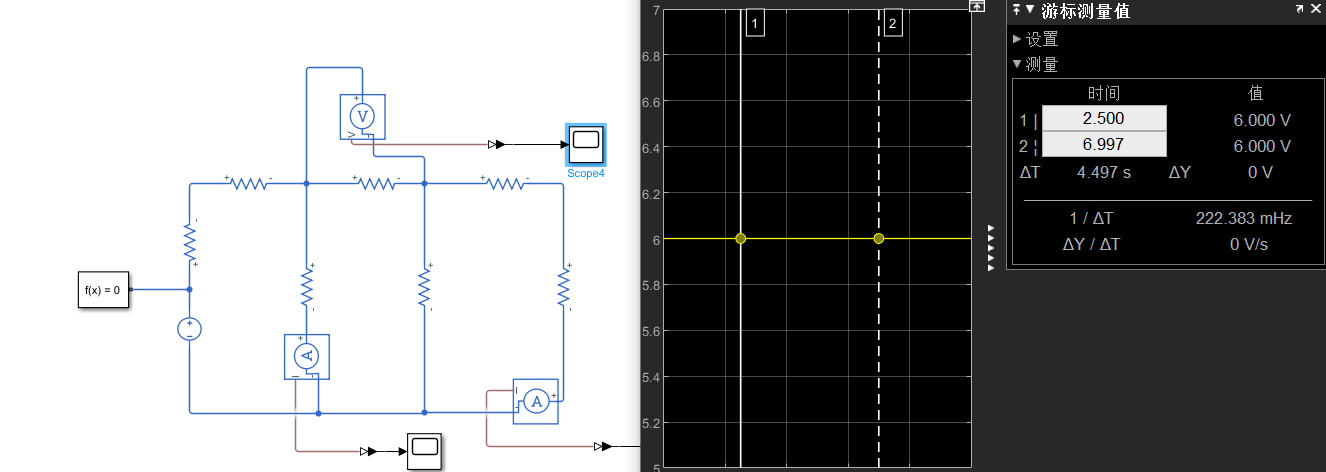


### （1-2）

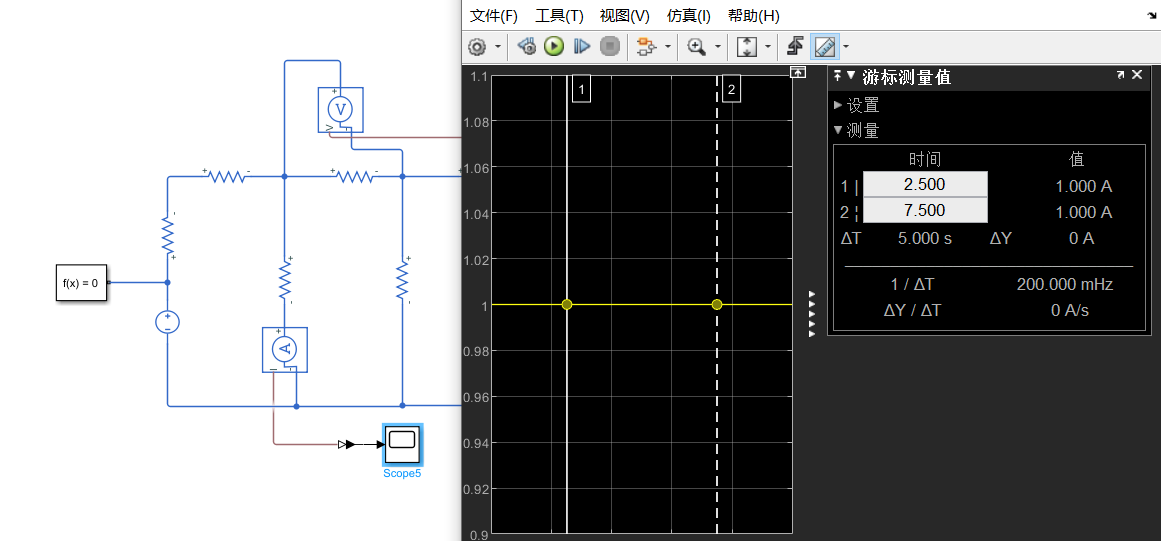
电路（2）搭建：



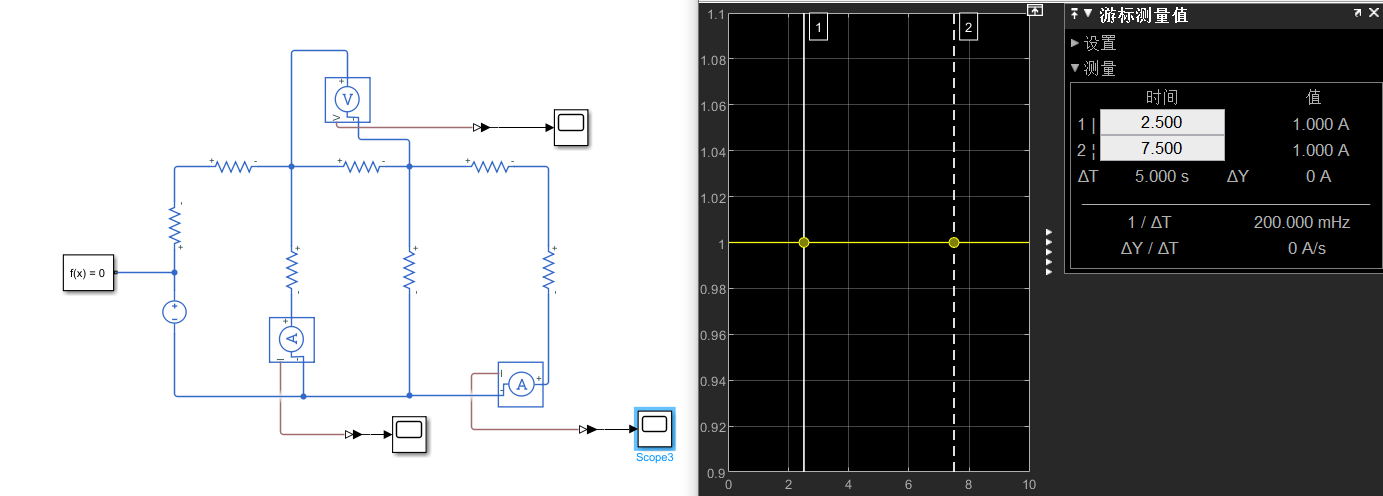
Us:



I3:

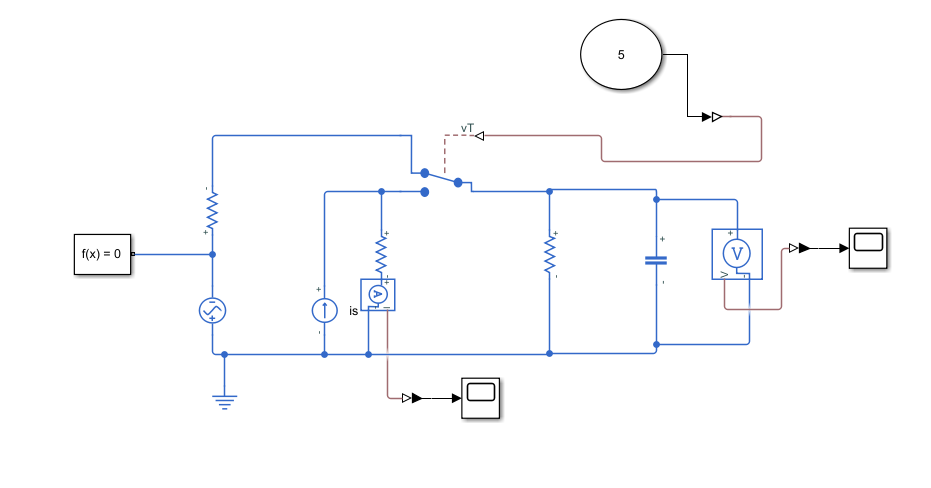


I7:

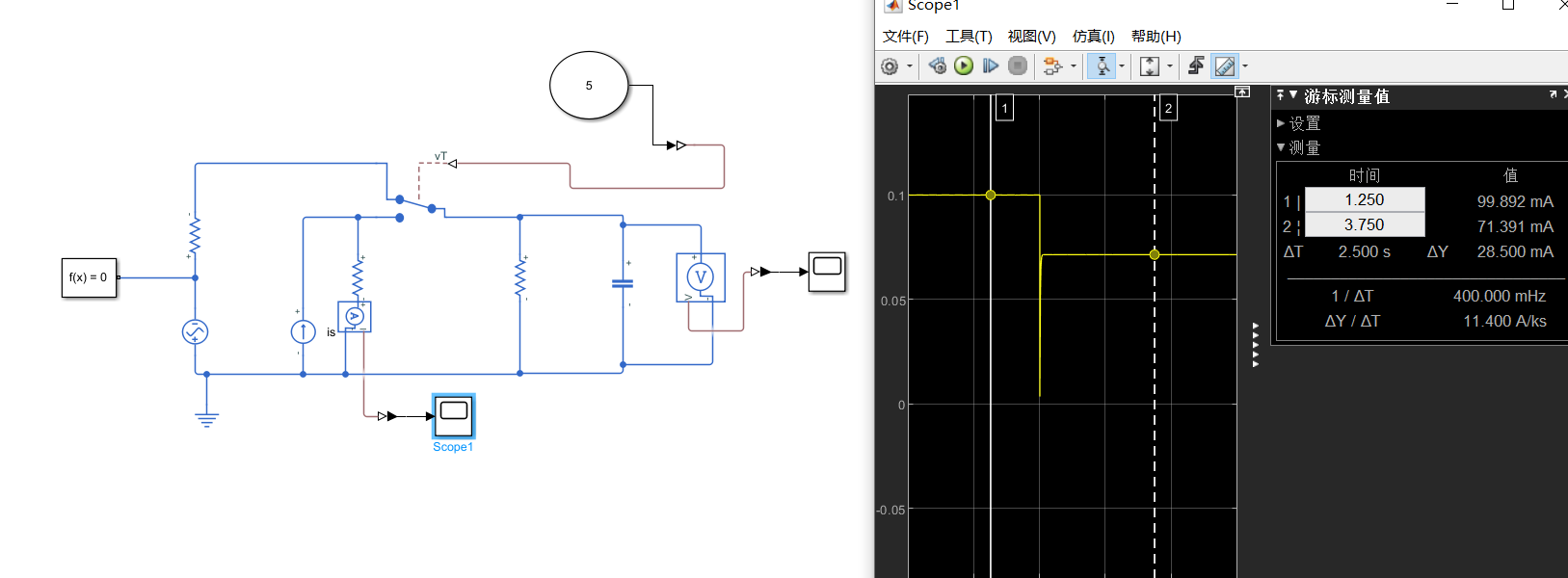


### （2）

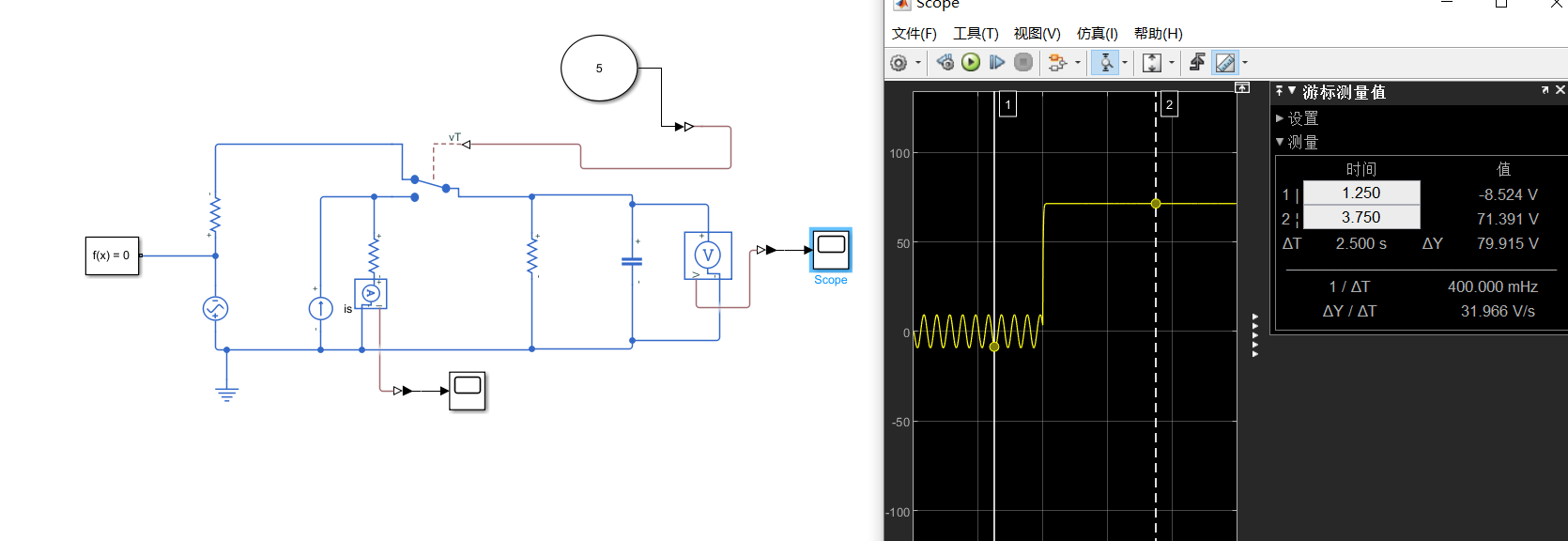
电路搭建：



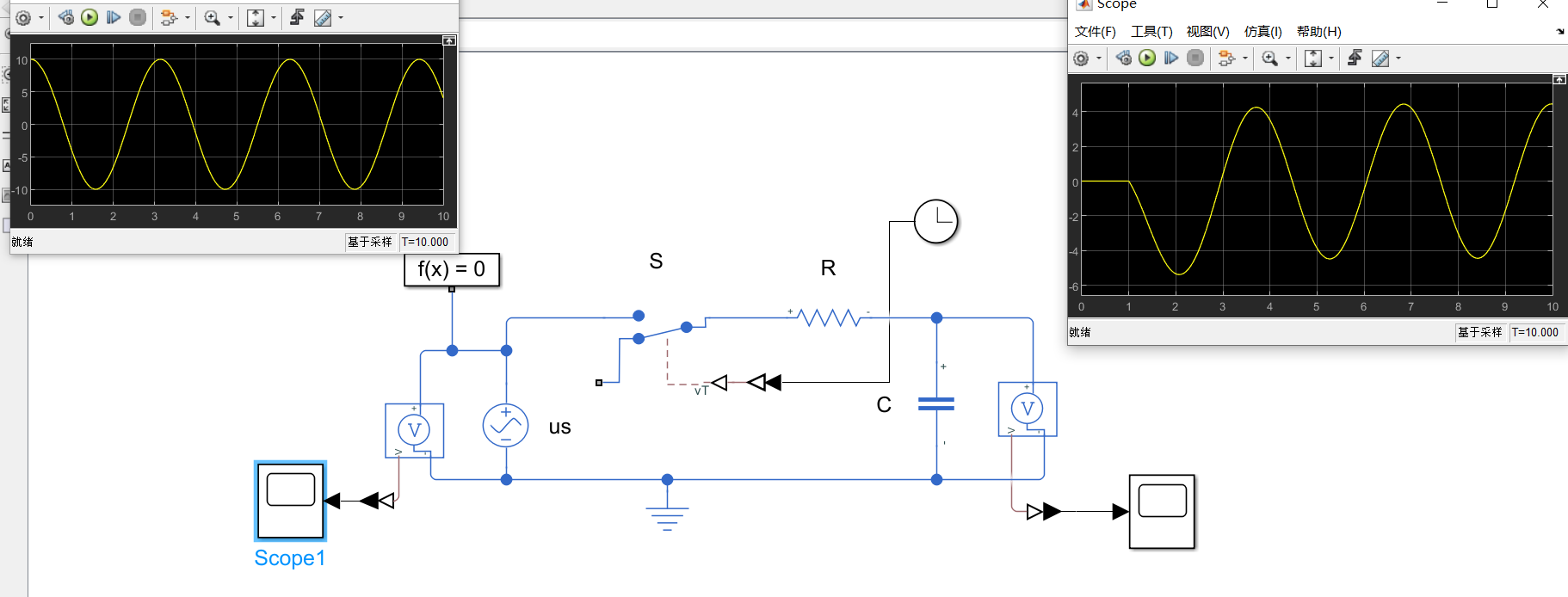
Ir2:



Uc:

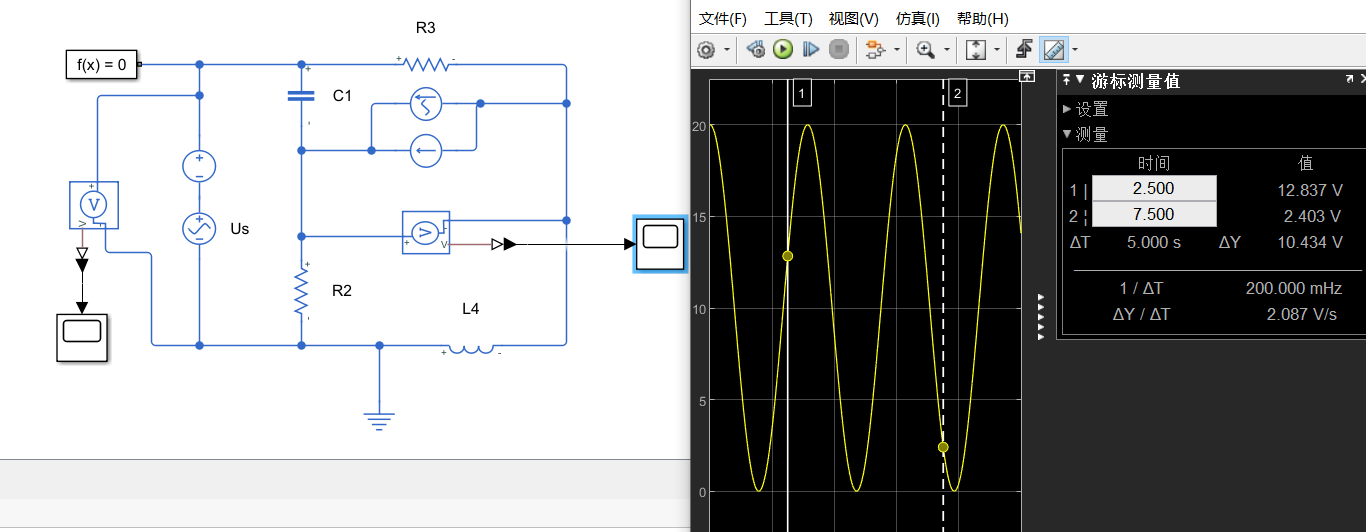


### （3）



### （4）

Ut:



## 三、结果分析

电路模拟与仿真：使用MATLAB的Simulink模块，您可以构建电路模型并模拟电路行为。通过建立电路的方程式，MATLAB可以模拟其时间响应、频率响应，以及其他动态特性。

信号处理：MATLAB提供了广泛的信号处理工具，可以用于分析和处理电路信号。例如，使用傅里叶变换来分析电路中的频率成分，使用滤波器来处理信号。

## 四、心得体会

总体而言，MATLAB是电路分析和设计中的强大工具。通过合理应用MATLAB的功能，我们可以更有效地设计、模拟和优化电路系统。

# 实验四　在信号与系统中Matlab的应用

## 一、实验内容

1、画信号的波形

2、求系统的零状态响应，已知。

3、求系统的零状态响应，已知。

4、求系统的单位脉冲响应。

5、试用MATLAB画出图示周期三角波信号的频谱。

6、试用数值方法近似计算三角波信号的频谱。



7、周期方波通过RC系统的响应。





8、画的幅度响应曲线。

## 二、实验过程（代码）

%% 4-1

% 定义信号参数

A = 1; % 振幅

a = 0.5; % 指数因子

omega\_0 = 2 \* pi; % 基频

phi = pi / 4; % 相位

% 定义时间范围

t = linspace(0, 10, 1000); % 从0到10，生成1000个点

% 定义信号

y = A \* exp(a \* t) .\* cos(omega\_0 \* t + phi);

% 绘制波形

figure;

plot(t, y);

title('信号 Ae^{at} \\cos(\\omega\_0 t + \\phi) 的波形'); % 修复了标题的特殊字符

xlabel('时间 (s)');

ylabel('幅度');

grid on;

%% 4-2

s = tf('s'); % 定义拉普拉斯变量

% 定义传递函数

H = 10 / (s^2 + 2 \* s + 100); % 系统的传递函数

% 输入信号的拉普拉斯变换

F\_s = 2 \* pi / (s^2 + (2 \* pi)^2); % sin(2 \* pi \* t) 的拉普拉斯变换

% 输出信号的拉普拉斯变换

Y\_s = H \* F\_s; % 传递函数与输入信号拉普拉斯变换的乘积

% 计算零状态响应的时间域

t = 0:0.01:10; % 定义时间范围

y\_t = impulse(Y\_s, t); % 使用冲激响应函数求逆拉普拉斯变换

% 绘制零状态响应

figure;

plot(t, y\_t);

title('系统零状态响应');

xlabel('时间 (s)');

ylabel('响应');

grid on;

%% 4-3

s = tf('s'); % 定义拉普拉斯变量

% 定义系统的传递函数

H = 10 / (s^2 + 2 \* s + 100); % 系统的传递函数

% 计算冲激响应

t = 0:0.01:10; % 时间范围

y\_t = impulse(H, t); % 计算冲激响应

% 绘制冲激响应波形

figure;

plot(t, y\_t);

title('系统冲激响应');

xlabel('时间 (s)');

ylabel('响应幅度');

grid on; % 显示网格线

%% 4-4

% 定义 Z 变量

z = tf('z', 1); % 定义 Z 变换变量，采样时间为1

% 定义系统传递函数

H = 10 / (1 + 3/z + 2/(z^2)); % 系统的传递函数

% 计算单位脉冲响应

N = 10; % 定义计算的长度

y\_impulse = impulse(H, N); % 计算单位脉冲响应

% 重新定义 X 轴

x\_values = 0:(N-1); % X 轴数据

% 确保两者的长度相等

max\_length = max(length(x\_values), length(y\_impulse)); % 找到最大长度

x\_values = [x\_values, (length(x\_values):(max\_length-1))]; % 填充 x\_values

y\_impulse = [y\_impulse; zeros(max\_length - length(y\_impulse), 1)]; % 填充 y\_impulse

% 绘制单位脉冲响应

figure;

stem(x\_values, y\_impulse); % 确保 X 和 Y 长度一致

title('系统单位脉冲响应');

xlabel('k');

ylabel('响应幅度');

grid on;

%% 4-5

% 定义时间范围

t=-6\*pi:0.0001:6\*pi;

% 定义周期为 2 的三角波，幅度为 -1 到 1

tri\_wave=-sawtooth(pi\*(t-0.5),0.5);

% 绘制三角波

figure;

plot(t, tri\_wave);

title('周期为 2 的三角波信号');

xlabel('时间 (s)');

ylabel('幅度');

grid on;

% 计算 FFT

N = length(t); % 信号的长度

Fs = N / (t(end) - t(1)); % 采样频率

tri\_wave\_fft = fft(tri\_wave); % 计算 FFT

% 计算频率

f = (0:(N/2 - 1)) \* (Fs / N); % 频率范围，只看正频率部分

% 计算 FFT 的幅度

fft\_magnitude = abs(tri\_wave\_fft / N); % 归一化 FFT 幅度

fft\_magnitude = fft\_magnitude(1:(N/2)); % 只看正频率部分

% 绘制频谱

figure;

plot(f, fft\_magnitude);

title('三角波信号的频谱');

xlabel('频率 (Hz)');

ylabel('幅度');

grid on; % 显示网格

%% 4-6

% 定义时间范围和采样点数

T = 2; % 信号周期

Fs = 100; % 采样频率

t = 0:1/Fs:4\*T; % 时间范围，生成足够多的点

% 生成三角波信号，周期为 2，幅度 ±1

tri\_wave = 2 \* sawtooth(2 \* pi / T \* (t-1), 0.5) - 1; % 幅度调整到 -1 到 1

% 绘制三角波信号

figure;

plot(t, tri\_wave);

title('三角波信号');

xlabel('时间 (s)');

ylabel('幅度');

grid on;

% 计算 FFT

N = length(tri\_wave); % 信号的长度

fft\_tri\_wave = fft(tri\_wave); % 应用 FFT

% 计算频率轴

f = (0:N-1) \* (Fs / N); % 频率范围

% 计算幅度谱

fft\_magnitude = abs(fft\_tri\_wave / N); % 归一化 FFT 幅度

% 绘制频谱图，只显示正频率部分

figure;

plot(f(1:floor(N/2)), fft\_magnitude(1:floor(N/2)));

title('三角波信号的频谱');

xlabel('频率 (Hz)');

ylabel('幅度');

grid on;

%% 4-7

% 定义时间范围

t = linspace(0, 10, 1000); % 时间范围，生成 1000 个点

% 定义周期方波信号

Vin = square(2\*pi\*t); % 周期方波信号

% 定义RC电路参数

R = 1; % 电阻（Ω）

C = 1; % 电容（F）

RC = R \* C; % RC 时间常数

% 计算系统的频率响应

s = tf('s'); % 创建传递函数变量

H = 1 / (RC\*s + 1); % RC系统的传递函数

% 计算周期方波信号通过RC系统的响应

Vout = lsim(H, Vin, t); % 使用 lsim 函数计算响应

% 绘制输入输出信号

figure;

plot(t, Vin, 'b', t, Vout, 'r');

title('周期方波通过RC系统的响应');

xlabel('时间 (s)');

ylabel('电压 (V)');

legend('输入信号', '输出信号');

grid on;

%% 4-8

% 定义参数

alpha = 0.5; % 可调整 alpha 值

% 定义传递函数的分子和分母

b = 1; % 分子系数

a = [1, -alpha]; % 分母系数

% 使用 freqz 计算频率响应

[H\_freq, w] = freqz(b, a, 512); % 512 是计算点数

% 绘制幅度响应曲线

figure;

plot(w, abs(H\_freq)); % 绘制幅度

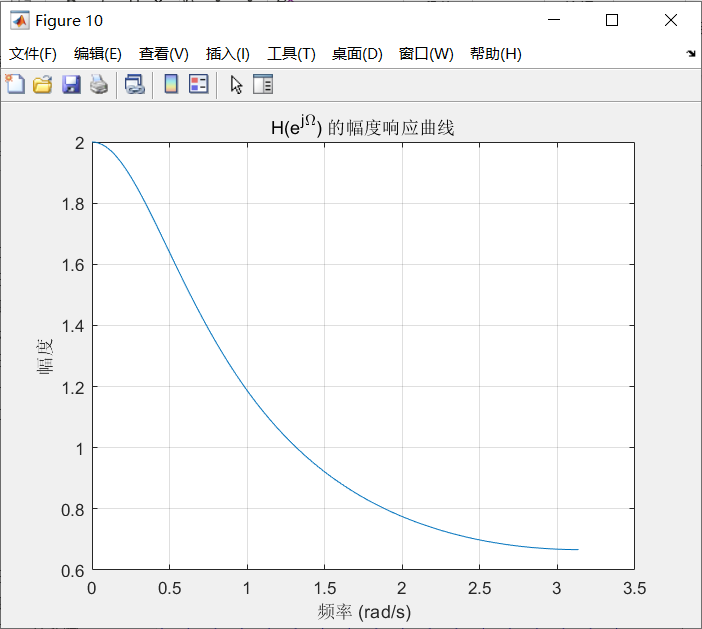
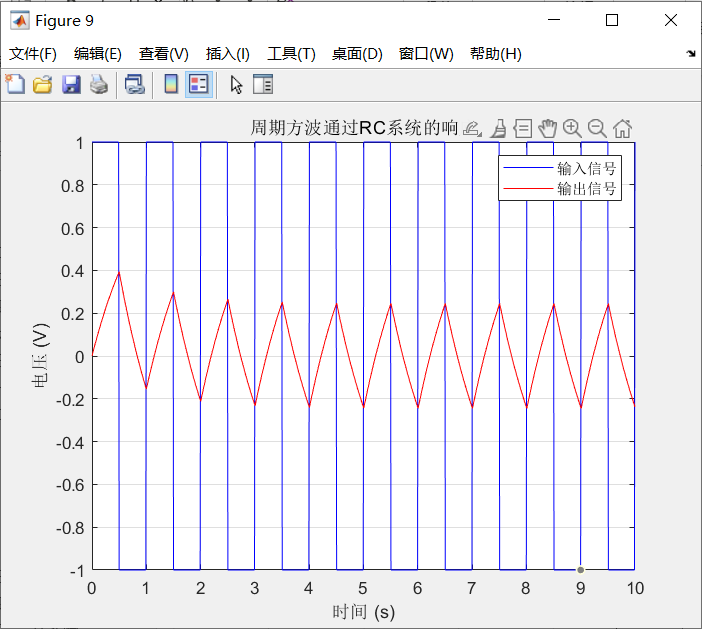
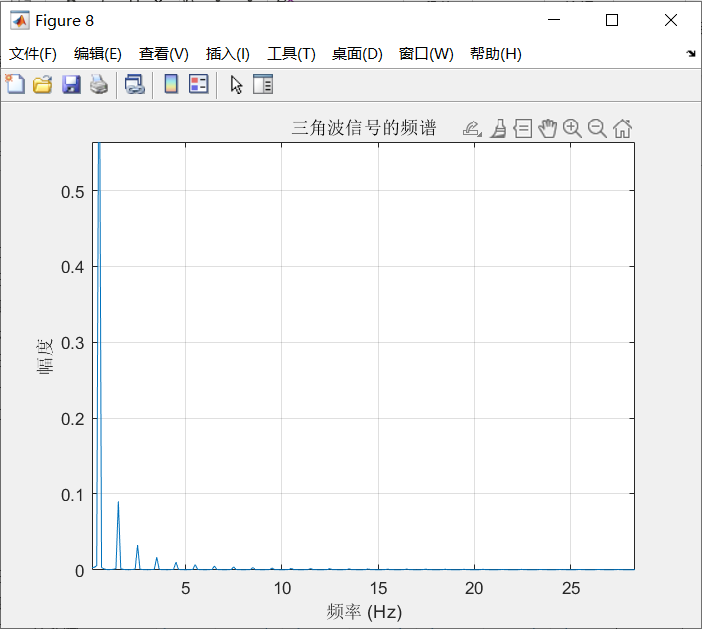
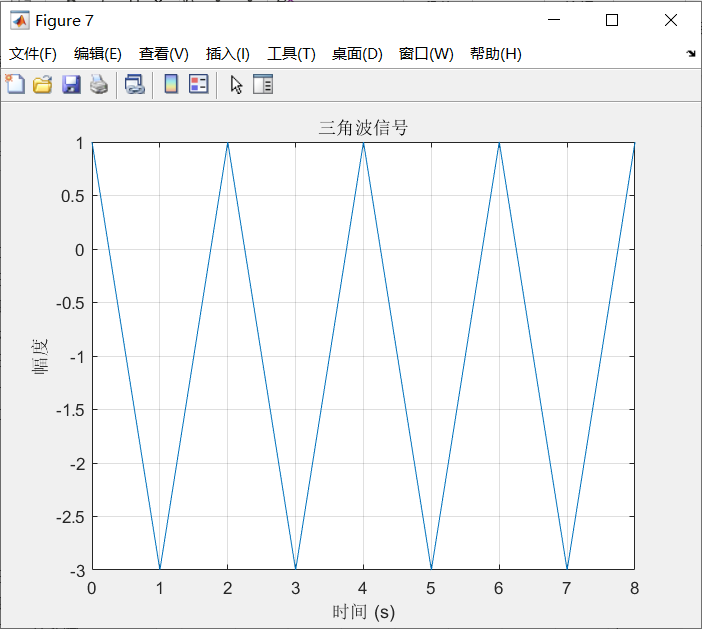
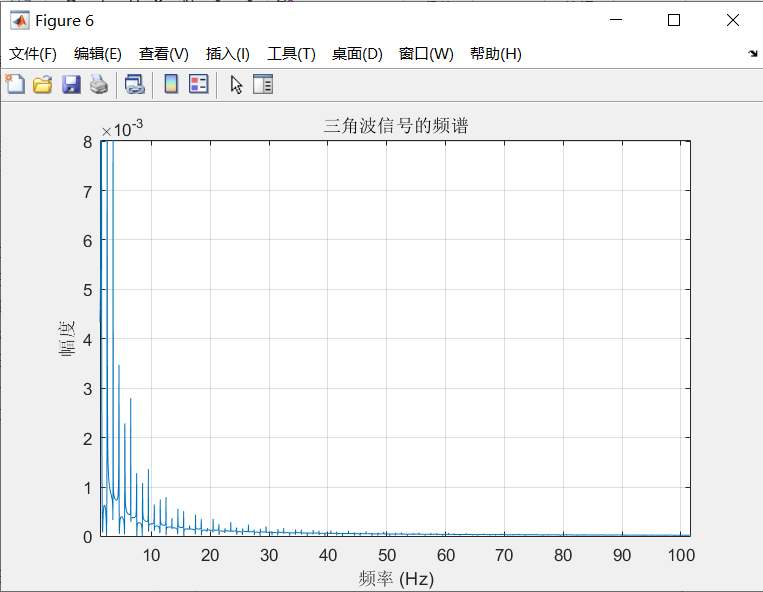
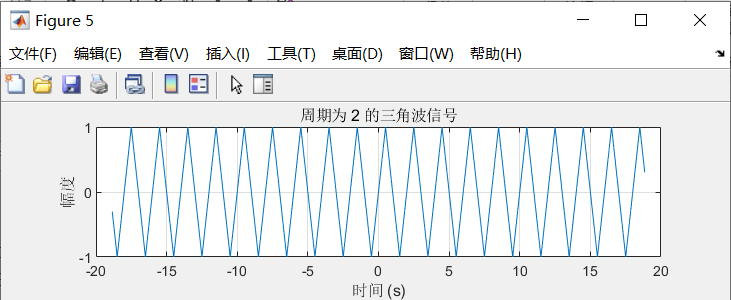
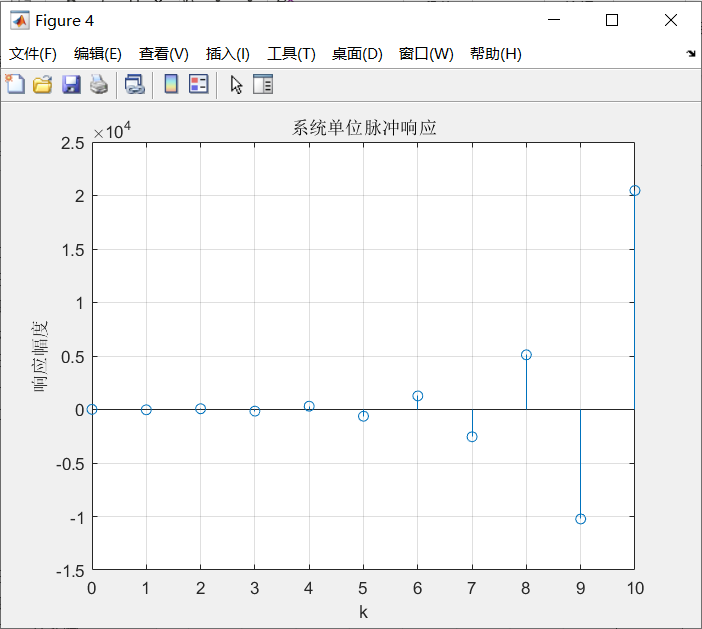
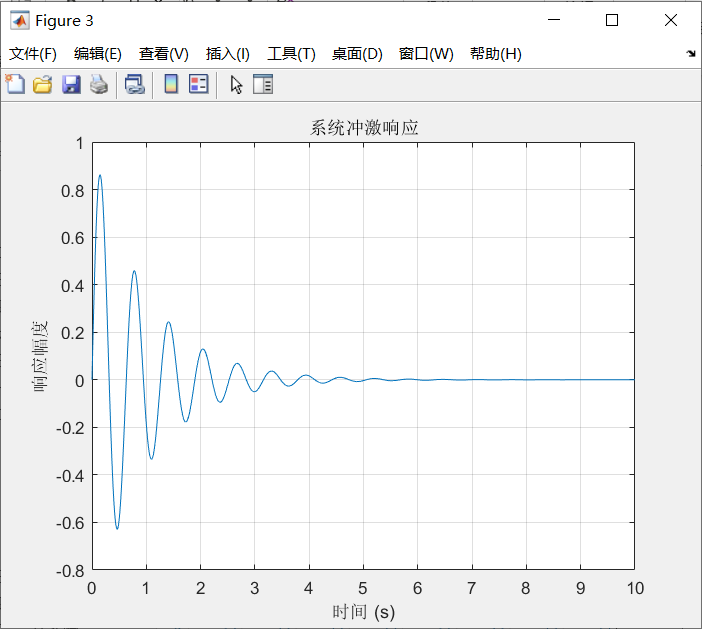
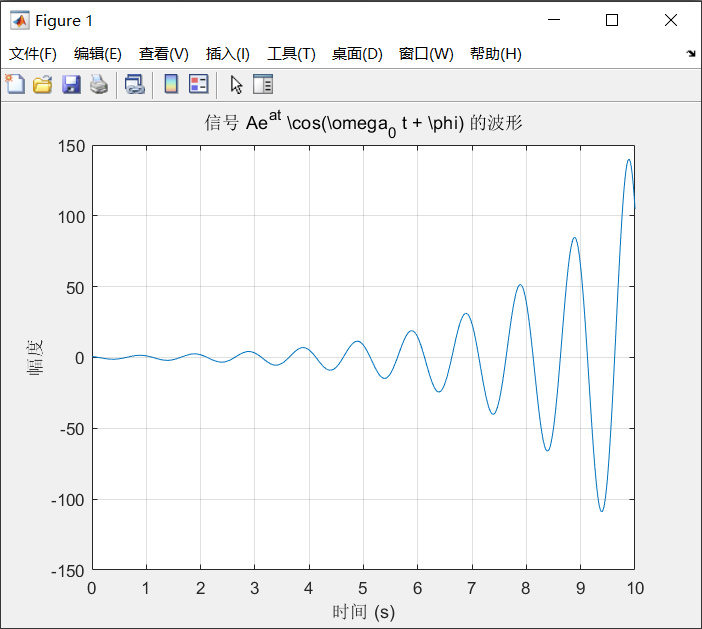
title('H(e^{j\Omega}) 的幅度响应曲线');

xlabel('频率 (rad/s)');

ylabel('幅度');

grid on;

## 三、实验结果



## 四、结果分析

**连续信号到离散信号：**

在MATLAB中，可以使用linspace和sampling函数对连续时间信号进行离散化。通过调整取样率，观察不同频率下的离散信号结果。

**时域取样定理验证：**

使用正弦信号生成连续信号，然后根据不同的取样率进行离散化，分析不同取样率对信号的影响。通过频谱分析，可以观察到低取样率下混叠现象。

**离散信号到连续信号：**

使用MATLAB的interp1或resample等函数进行信号重构。尝试不同的插值方法来观察其对信号恢复的影响。

**傅里叶分析：**

通过MATLAB的fft函数，对连续信号进行频谱分析，观察其频谱特性。同时，可以利用ifft进行反向变换，验证傅里叶分析的准确性。

**离散信号频谱分析：**

使用MATLAB的fft对离散信号进行DFT，分析其频谱特性。验证不同取样率下的离散信号的频谱特点。

**系统响应分析：**

使用MATLAB的tf、lsim等函数，分析连续系统的冲激响应、阶跃响应等。观察系统的零极点对系统稳定性和频率响应的影响。

## 五、心得体会

**时域取样定理是信号处理的基础：**

了解取样率与信号频率的关系，确保不会产生混叠。过低的取样率会导致信号丢失信息，导致无法准确重构。

**傅里叶分析提供了强大的信号分析工具：**

通过傅里叶变换，能够有效分析信号的频谱成分，并验证信号处理过程的正确性。

**MATLAB是进行信号处理的强大工具：**

MATLAB提供了多种功能，可以用于模拟、分析和重构信号。这包括从取样、傅里叶分析到系统响应等。

**稳定性与系统响应的理解：**

通过MATLAB分析连续系统的响应，深入理解系统的稳定性和频率响应。这对控制系统和电信号处理非常重要。

# 实验五　在数字信号处理中Matlab的应用

## 一、实验内容

1、观察高斯序列的时域和幅频特性，固定信号中参数，改变的值，使分别等于2，4，8，观察它们的时域和幅频特性，了解当取不同值时，对信号序列的时域幅频特性的影响；固定，改变，使分别等于8，13，14，观察参数变化对信号序列的时域及幅频特性的影响，观察等于多少时，会发生明显的泄漏现象，混叠是否也随之出现？记录实验中观察到的现象，绘出相应的时域序列和幅频特性曲线。

2、观察衰减正弦序列的时域和幅频特性，，检查谱峰出现位置是否正确，注意频谱的形状，绘出幅频特性曲线，改变，使分别等于和，观察这两种情况下，频谱的形状和谱峰出现位置，有无混叠和泄漏现象？说明产生现象的原因。

3、观察三角波和反三角波序列的时域和幅频特性，用点FFT分析信号序列和的幅频特性，观察两者的序列形状和频谱曲线有什么异同？绘出两序列及其幅频特性曲线。

在和末尾补零，用点FFT分析这两个信号的幅频特性，观察幅频特性发生了什么变化？两情况的FFT频谱还有相同之处吗？这些变化说明了什么？

4、一个连续信号含两个频率分量，经采样得

已知，分别为和，观察其频谱；当时，不变，其结果有何不同，为什么？

 5、用FFT分别实现和 的16点圆周卷积和线性卷积。

6、产生一512点的随机序列，并用和作线性卷积，观察卷积前后频谱的变化。要求将分成8段，分别采用重叠相加法和重叠保留法。

7、分别用脉冲响应不变法及双线性变换法设计一个Butterworth数字低通滤波器，观察所设计数字滤波器的幅频特性曲线，记录带宽和衰减量，检查是否满足要求。比较这两种方法的优缺点。

## 二、实验过程（代码）

%% 5-1

% Define parameters

n = 0:15; % Time indices

% Parameters for Gaussian sequence

p\_default = 8;

q\_values = [2, 4, 8];

p\_values = [8, 13, 14];

a = 0.1; % For decaying sine

f = 0.1; % Frequency

% Generate Gaussian sequences with varying q

for q = q\_values

x\_a = exp(-((n - p\_default).^2) / q);

plot\_time\_freq(x\_a, ['Gaussian sequence with q=', num2str(q)]);

end

% Generate Gaussian sequences with varying p

for p = p\_values

x\_a = exp(-((n - p).^2) / q\_values(end)); % Use the last q as default

plot\_time\_freq(x\_a, ['Gaussian sequence with p=', num2str(p)]);

end

%% 5-2

% 参数定义

n = 0:15; % 定义时间范围

a = 0.2; % 衰减因子

f1 = 0.1; % 正弦频率 1

f2 = 0.5; % 正弦频率 2

% 衰减正弦序列

x\_b1 = exp(-a \* n) .\* sin(2 \* pi \* f1 \* n); % 频率为f1

x\_b2 = exp(-a \* n) .\* sin(2 \* pi \* f2 \* n); % 频率为f2

% 绘制时域信号

figure;

subplot(2, 1, 1);

stem(n, x\_b1);

title('时域信号 - 衰减正弦序列 (f = 0.1)');

xlabel('n');

ylabel('x\_b(n)');

grid on;

subplot(2, 1, 2);

stem(n, x\_b2);

title('时域信号 - 衰减正弦序列 (f = 0.5)');

xlabel('n');

ylabel('x\_b(n)');

grid on;

% 计算并绘制频谱

X\_b1 = fft(x\_b1, 256); % 零填充至256点FFT

X\_b2 = fft(x\_b2, 256); % 零填充至256点FFT

f = (0:127) / 256; % 频率范围

figure;

subplot(2, 1, 1);

plot(f, abs(X\_b1(1:128))); % 绘制正频率范围

title('频域信号 - 衰减正弦序列 (f = 0.1)');

xlabel('频率');

ylabel('|X\_b(f)|');

grid on;

subplot(2, 1, 2);

plot(f, abs(X\_b2(1:128))); % 绘制正频率范围

title('频域信号 - 衰减正弦序列 (f = 0.5)');

xlabel('频率');

ylabel('|X\_b(f)|');

grid on;

%% 5-3

% 定义序列范围

n = 0:15; % 序列长度

% 三角波序列

x\_c = [n(1:4) + 1, 8 - n(5:8)]; % 三角波序列

x\_c = [x\_c, zeros(1, 8)]; % 在末尾补零

% 反三角波序列

x\_d = [4 - n(1:4), n(5:8) - 3]; % 反三角波序列

x\_d = [x\_d, zeros(1, 8)]; % 在末尾补零

% 绘制时域信号

figure;

subplot(2, 1, 1);

stem(0:15, x\_c(1:16));

title('三角波序列 - 时域');

xlabel('n');

ylabel('x\_c(n)');

grid on;

subplot(2, 1, 2);

stem(0:15, x\_d(1:16));

title('反三角波序列 - 时域');

xlabel('n');

ylabel('x\_d(n)');

grid on;

% 使用256点FFT进行频域分析

N = 256; % FFT的点数

X\_c = fft(x\_c, N); % 计算三角波序列的FFT

X\_d = fft(x\_d, N); % 计算反三角波序列的FFT

% 绘制频域信号

f = (0:(N/2 - 1)) \* (1 / N); % 正频率范围

figure;

subplot(2, 1, 1);

plot(f, abs(X\_c(1:N/2)));

title('三角波序列 - 幅频特性');

xlabel('频率 (单位: 周期/采样点数)');

ylabel('|X\_c(f)|');

grid on;

subplot(2, 1, 2);

plot(f, abs(X\_d(1:N/2)));

title('反三角波序列 - 幅频特性');

xlabel('频率 (单位: 周期/采样点数)');

ylabel('|X\_d(f)|');

grid on;

%% 5-4

% 参数定义

N1 = 16; % 第一个样本数

N2 = 128; % 第二个样本数

f\_base = 0.125; % 基础频率

% Δf 分别为 1/16 和 1/64

delta\_f1 = 1/16; % 第一个 Δf

delta\_f2 = 1/64; % 第二个 Δf

% 生成信号

n1 = 0:(N1 - 1); % 时间索引

x1 = sin(2 \* pi \* f\_base \* n1) + cos(2 \* pi \* (f\_base + delta\_f1) \* n1);

n2 = 0:(N2 - 1); % 时间索引

x2 = sin(2 \* pi \* f\_base \* n2) + cos(2 \* pi \* (f\_base + delta\_f2) \* n2);

% 计算频谱

X1 = fft(x1); % 对第一个信号进行FFT

X2 = fft(x2); % 对第二个信号进行FFT

% 频率轴

f1 = (0:(N1 - 1)) / N1; % 对应N1的频率轴

f2 = (0:(N2 - 1)) / N2; % 对应N2的频率轴

% 绘制频谱

figure;

subplot(2, 1, 1);

plot(f1, abs(X1)); % 绘制第一个信号的频谱

title('频谱 - N = 16, Δf = 1/16');

xlabel('归一化频率');

ylabel('|X(f)|');

grid on;

subplot(2, 1, 2);

plot(f2, abs(X2)); % 绘制第二个信号的频谱

title('频谱 - N = 128, Δf = 1/64');

xlabel('归一化频率');

ylabel('|X(f)|');

grid on;

%% 5-5

% 定义参数

p = 8; % Gaussian序列的均值

q = 2; % Gaussian序列的方差

a = 0.1; % 衰减正弦序列的衰减因子

f = 0.0625; % 衰减正弦序列的频率

% 定义序列长度

n = 0:15; % 时间范围

% 定义Gaussian序列

x\_a = exp(-((n - p).^2) / q); % Gaussian序列

% 定义衰减正弦序列

x\_b = exp(-a \* n) .\* sin(2 \* pi \* f \* n); % 衰减正弦序列

% 圆周卷积（16点FFT）

X\_a = fft(x\_a, 16); % 对x\_a进行16点FFT

X\_b = fft(x\_b, 16); % 对x\_b进行16点FFT

% FFT的乘积实现圆周卷积

circ\_conv\_result = ifft(X\_a .\* X\_b); % 圆周卷积

% 线性卷积（需要零填充以避免循环效应）

X\_a\_padded = fft(x\_a, 31); % 零填充至31点，保证线性卷积长度

X\_b\_padded = fft(x\_b, 31); % 零填充至31点

% FFT的乘积实现线性卷积

linear\_conv\_result = ifft(X\_a\_padded .\* X\_b\_padded); % 线性卷积

% 绘制结果

figure;

subplot(2, 1, 1);

stem(0:15, circ\_conv\_result);

title('圆周卷积 - x\_a(n) 和 x\_b(n)');

xlabel('n');

ylabel('结果');

grid on;

subplot(2, 1, 2);

stem(0:30, linear\_conv\_result);

title('线性卷积 - x\_a(n) 和 x\_b(n)');

xlabel('n');

ylabel('结果');

grid on;

%% 5-6

% 生成高斯序列 x\_a

p = 7.5; % 中心位置

q = 2; % 标准差

n = 0:15; % 序列索引

x\_a = exp(-((n - p).^2) / q);

% 生成衰减正弦序列 x\_b

a\_n = 0.3; % 衰减因子

f\_n = 0.2; % 正弦频率

x\_b = exp(-a\_n \* n) .\* sin(2 \* pi \* f\_n \* n);

% 生成三角波序列 x\_c

x\_c = [n(1:4) + 1, 8 - n(5:8)];

% 生成反三角波序列 x\_d

x\_d = [4 - n(1:4), n(5:8) - 3];

% 产生 512 点随机序列 x\_rand

x\_rand = randn(1, 512);

% 对 x\_rand 与 x\_c 进行卷积

y\_conv\_c = conv(x\_rand, x\_c);

% 对 x\_rand 与 x\_d 进行卷积

y\_conv\_d = conv(x\_rand, x\_d);

% 重叠相加法

segment\_size = 64; % 512/8 = 64

y\_overlap\_add = zeros(1, 512 + length(x\_c) - 1); % 创建足够大的结果数组

% 遍历 8 段

for i = 0:7

start\_index = i \* (segment\_size - (length(x\_c) - 1)); % 注意重叠部分

segment = x\_rand(start\_index + 1 : start\_index + segment\_size); % 获取分段

conv\_segment = conv(segment, x\_c); % 进行卷积

% 确保索引范围在允许范围内

end\_index = start\_index + length(conv\_segment);

if end\_index <= length(y\_overlap\_add)

% 重叠相加

y\_overlap\_add(start\_index + 1 : end\_index) = ...

y\_overlap\_add(start\_index + 1 : end\_index) + conv\_segment;

end

end

% 绘制重叠相加法的结果

figure;

plot(y\_overlap\_add);

title('重叠相加法的结果');

xlabel('样本');

ylabel('幅度');

%% 5-7

% 定义滤波器规格

f\_p = 0.2e3; % 通带频率 (0.2 kHz)

f\_r = 0.3e3; % 阻带频率 (0.3 kHz)

A\_p = 1; % 通带衰减 (1 dB)

A\_r = 25; % 阻带衰减 (25 dB)

T = 1e-3; % 采样时间 (1 ms)

% 转换频率至归一化角频率 (rad/s)

omega\_p = 2 \* pi \* f\_p; % 通带角频率

omega\_r = 2 \* pi \* f\_r; % 阻带角频率

% 计算Butterworth滤波器的阶数和截止频率

[N, Wn] = buttord(omega\_p, omega\_r, A\_p, A\_r, 's'); % 计算模拟滤波器的阶数和截止频率

[b, a] = butter(N, Wn, 's'); % 模拟低通Butterworth滤波器的传递函数

% 脉冲响应不变法

[b\_ii, a\_ii] = impinvar(b, a, 1 / T); % 通过脉冲响应不变法转换至数字滤波器

% 双线性变换法

[b\_bl, a\_bl] = bilinear(b, a, 1 / T); % 通过双线性变换转换至数字滤波器

% 绘制幅频特性

fvtool(b\_ii, a\_ii, b\_bl, a\_bl); % 使用滤波器可视化工具比较两个滤波器

legend('脉冲响应不变法', '双线性变换法');

%% 5-1

% Helper function to plot time domain and frequency domain

function plot\_time\_freq(seq, title\_str)

% Plot the time-domain sequence

figure;

subplot(2, 1, 1);

stem(seq, 'filled');

title(['Time-domain: ', title\_str]);

xlabel('n'); ylabel('Amplitude');

% Plot the frequency-domain sequence (magnitude spectrum)

subplot(2, 1, 2);

mag\_spectrum = abs(fft(seq, 256)); % FFT with zero-padding

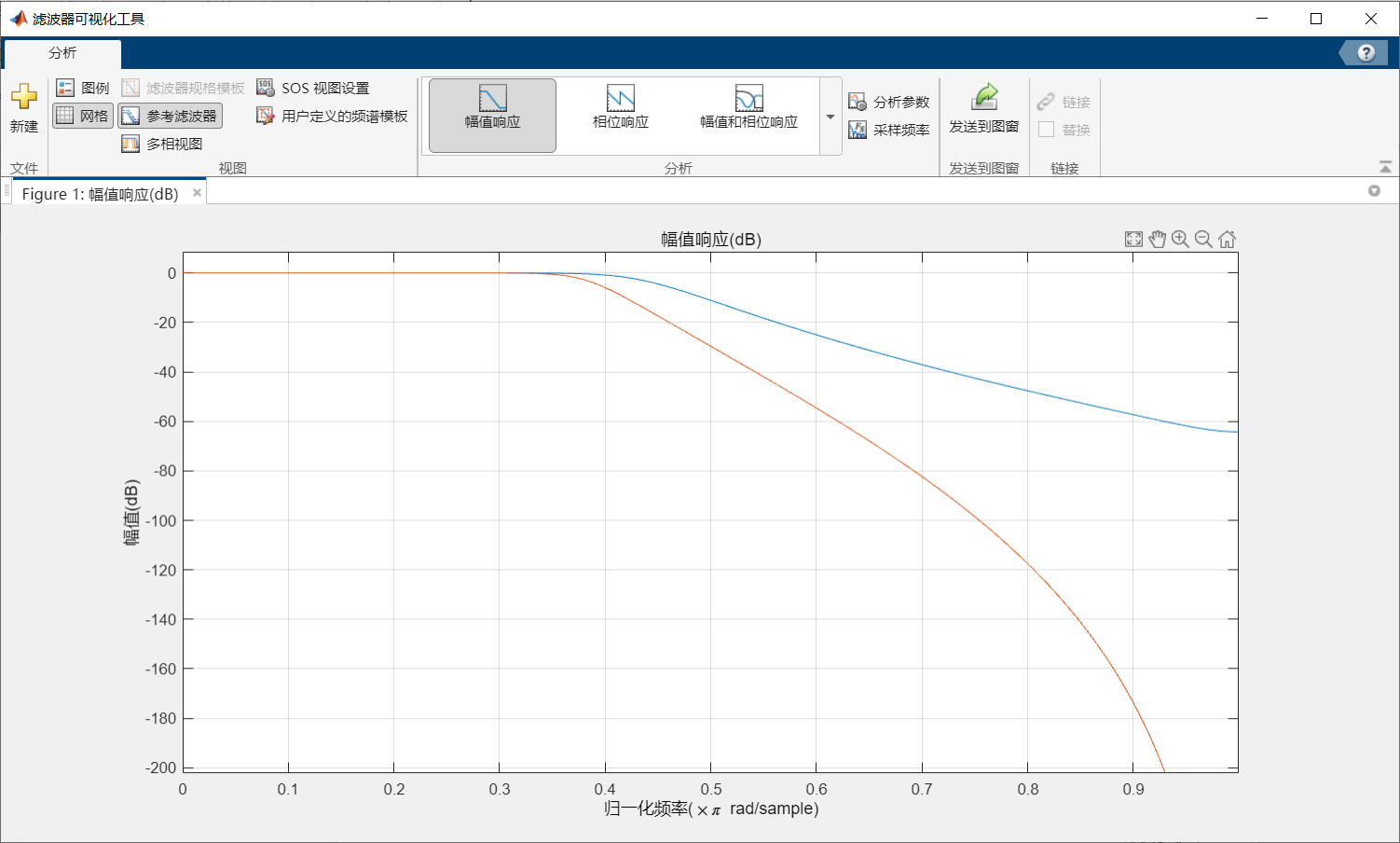
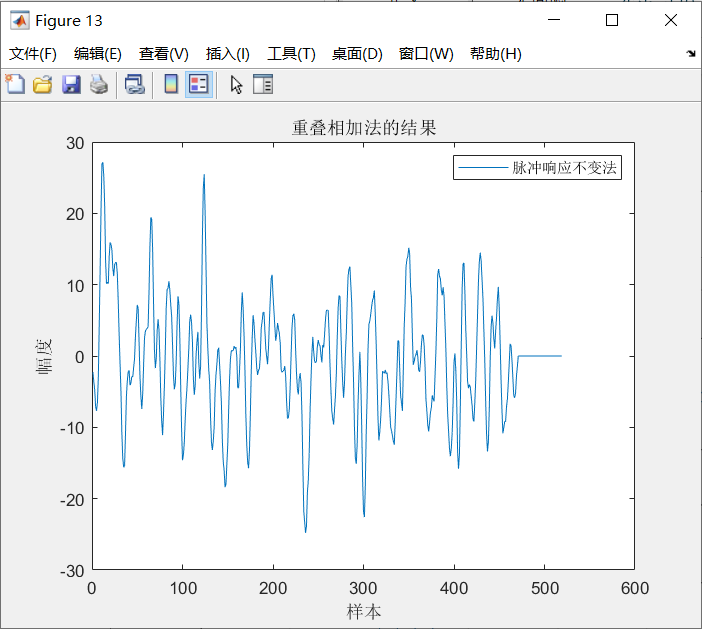
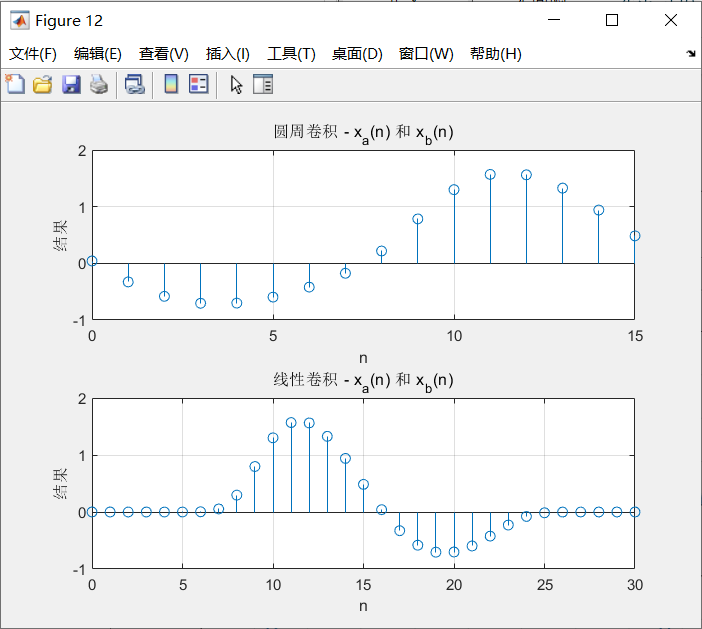
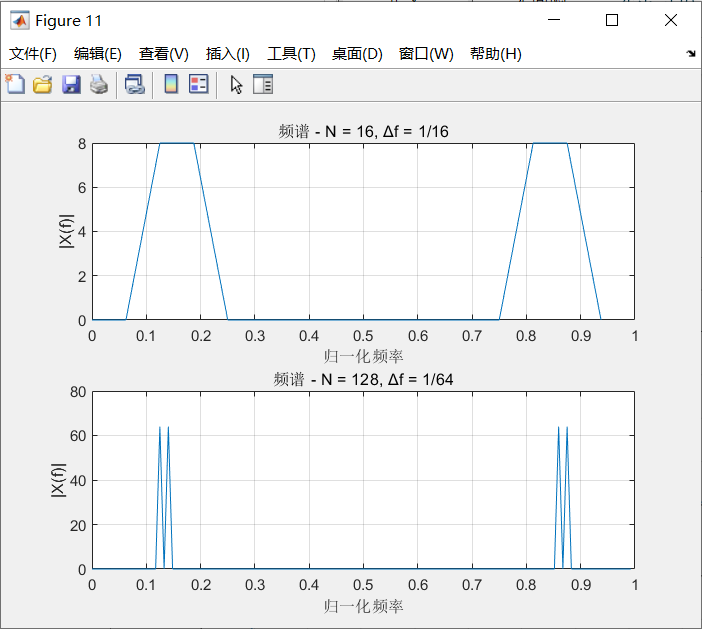
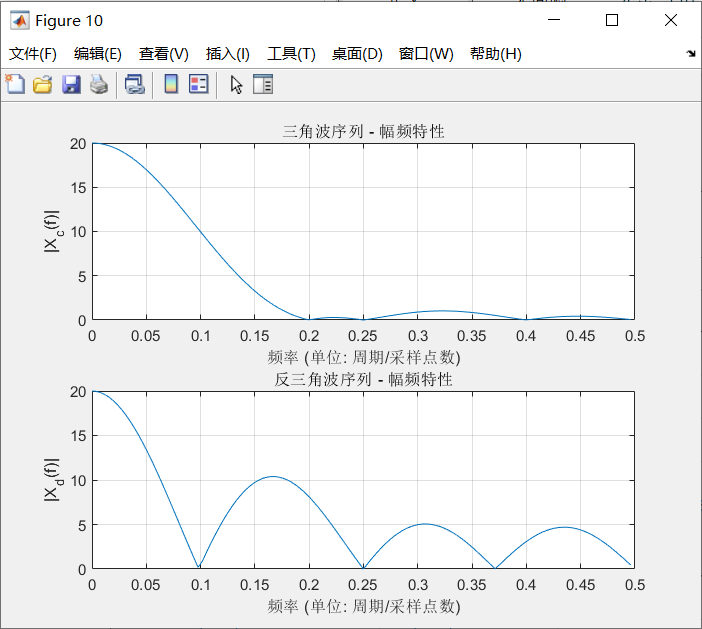
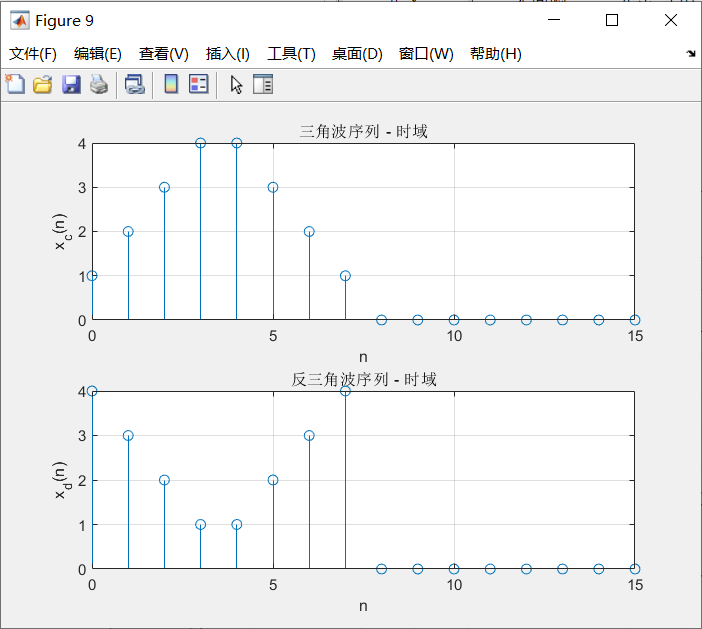
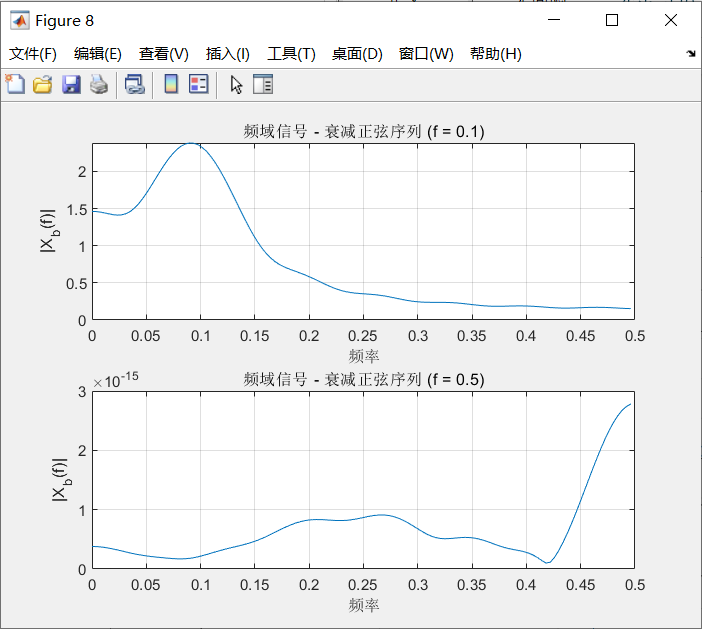
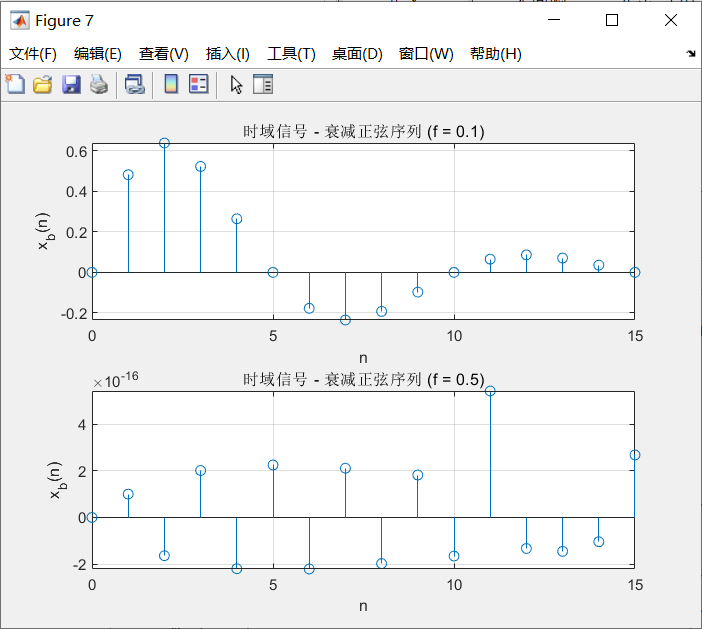
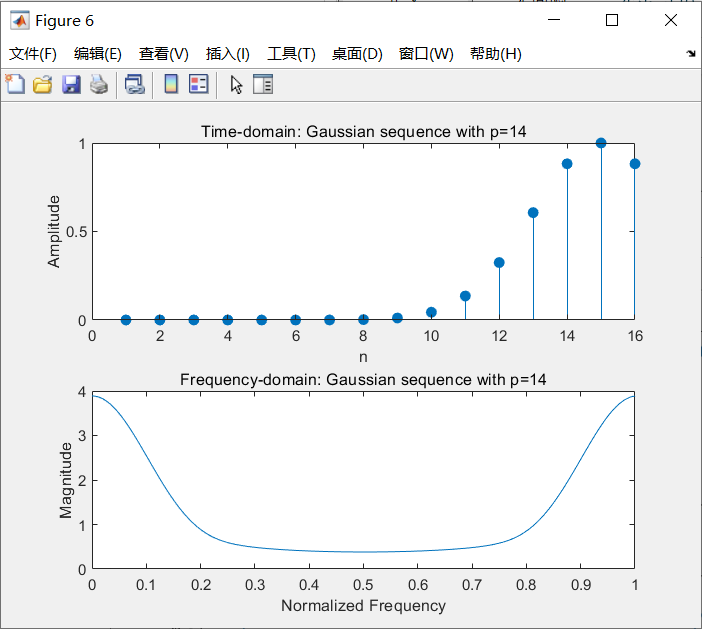
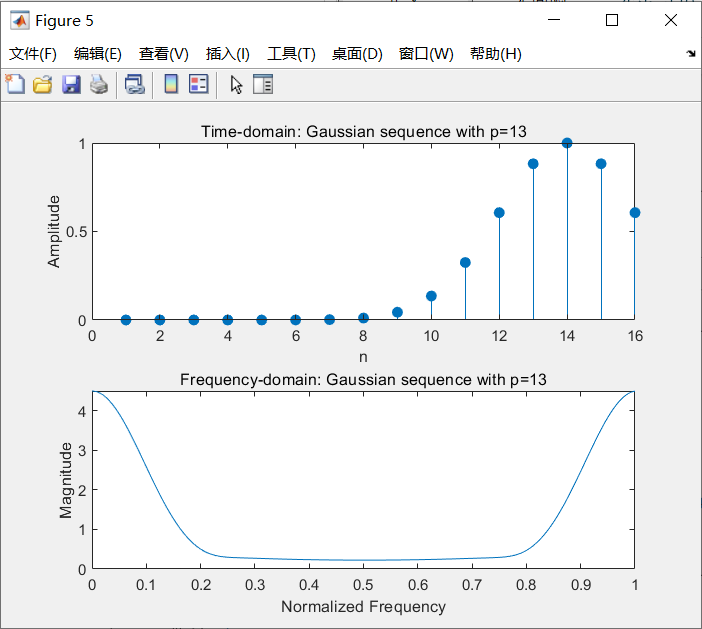
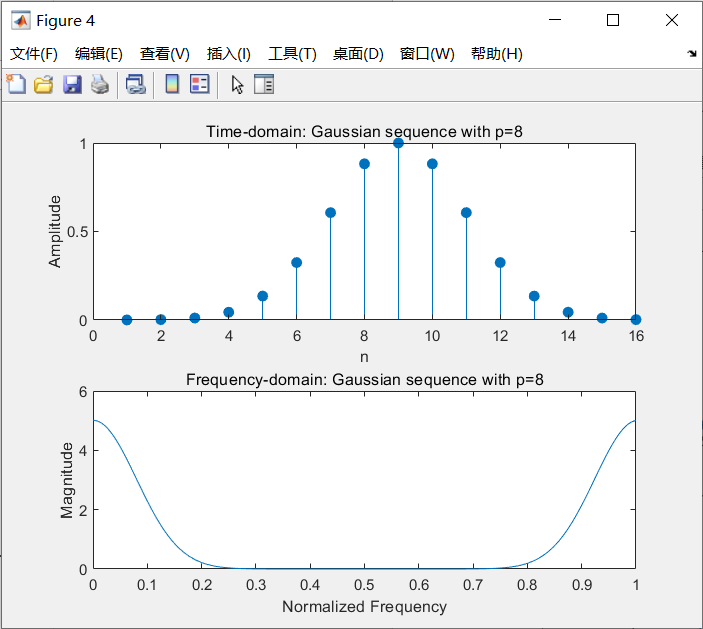
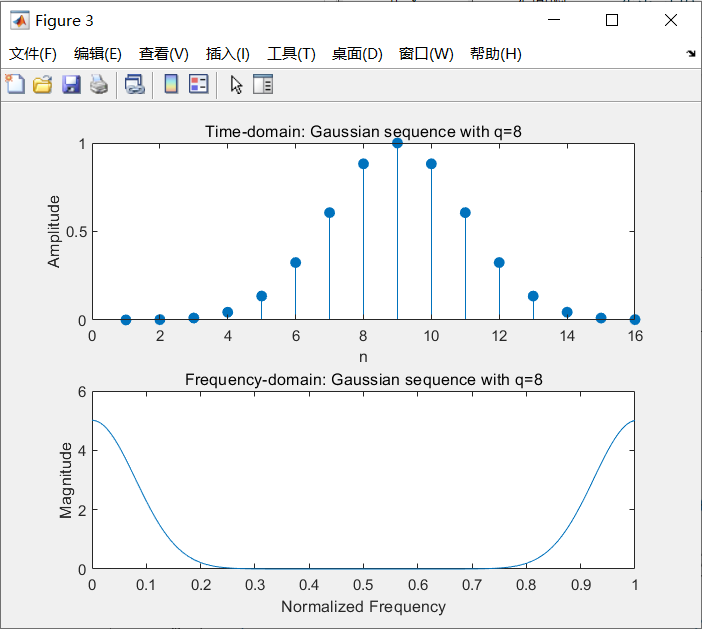
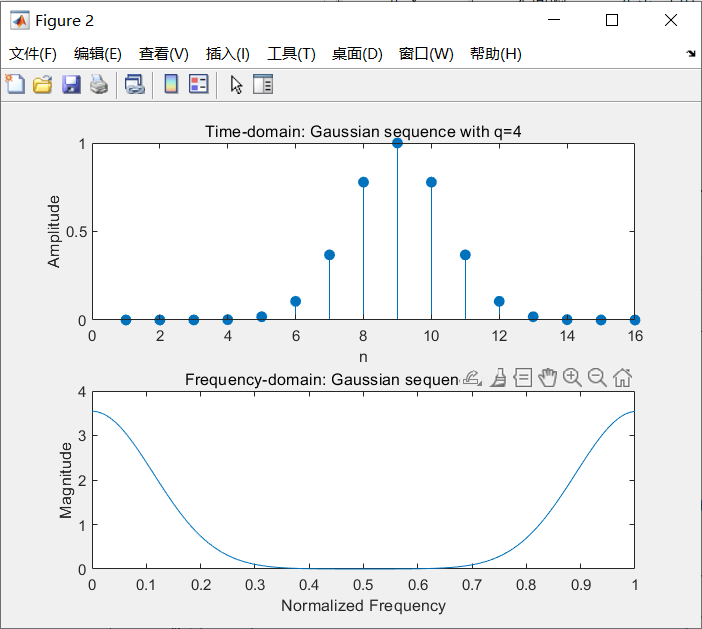
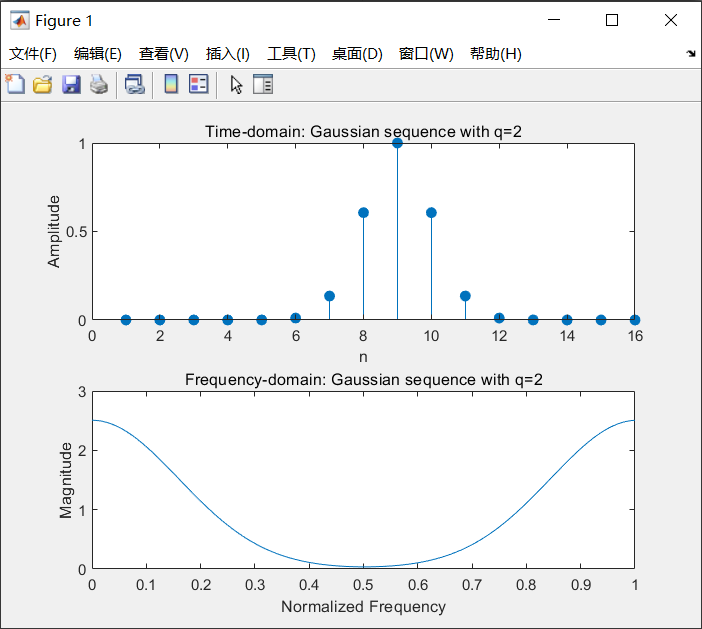
plot(linspace(0, 1, 256), mag\_spectrum);

title(['Frequency-domain: ', title\_str]);

xlabel('Normalized Frequency'); ylabel('Magnitude');

end

## 三、实验结果



## 四、结果分析

**典型信号频谱分析：**

使用MATLAB的fft函数，对正弦波、方波等典型信号进行频谱分析。可以观察到不同信号在频域的特征，如正弦波产生单一频率成分，方波包含多个谐波。

**FFT潜在问题：**

在频谱分析中，若信号的长度不是2的幂次，可能导致泄漏。通过增加零填充来匹配FFT长度，可以减少这种现象。避免混叠则需要确保信号在合适的取样率下采样。

**FFT卷积：**

使用MATLAB的fft和ifft函数，实现两个序列的卷积。对比传统卷积方法，FFT-based卷积更为高效，特别是处理长序列时。

**滤波器设计：**

使用MATLAB设计低通、高通和带通IIR滤波器。可以观察到不同设计方法的滤波器频率响应特点，以及设计过程中的参数选择。

**频域特性：**

使用MATLAB绘制滤波器的幅度和相位响应。比较Butterworth、Chebyshev和椭圆滤波器，观察各自的频率特性。例如，Butterworth滤波器具有平滑的幅度响应，Chebyshev滤波器有更陡峭的过渡区，椭圆滤波器则提供最陡峭的过渡和最少的通带波动。

## 五、心得体会

**FFT是高效的频谱分析工具：**

FFT在信号处理和频谱分析中非常重要，但需要注意潜在问题，如混叠和泄漏。在实际应用中，通过适当调整取样率和使用零填充等方法，可以减少这些问题。

**滤波器设计方法多样：**

双线性变换法和脉冲响应不变法提供了设计IIR数字滤波器的不同途径。通过实验，我学会了如何利用MATLAB设计和分析这些滤波器。

**不同滤波器的特性各异：**

各种滤波器，如Butterworth、Chebyshev和椭圆滤波器，各自具有不同的频域特性。在设计时，需要根据具体需求选择合适的滤波器类型。

**MATLAB在信号处理中的强大功能：**

MATLAB提供了丰富的工具，可以用于信号处理、频谱分析、滤波器设计等。通过实践，我熟悉了这些工具的使用方法，并学会了如何利用它们进行实验和分析。

# 实验六 GUI图形用户界面设计

## 一、实验内容

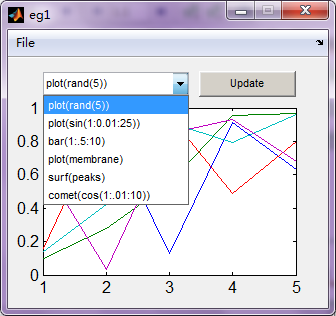
1、利用MATLAB环境制作一个简单的GUI程序。

启动GUI开发环境窗口，创建一个新的GUI工程文件，选择“GUI with Axes and Menu”选项，在默认生成的窗口上布置坐标轴控件、按钮和菜单，然后单击按钮“OK”，进入GUI开发环境。

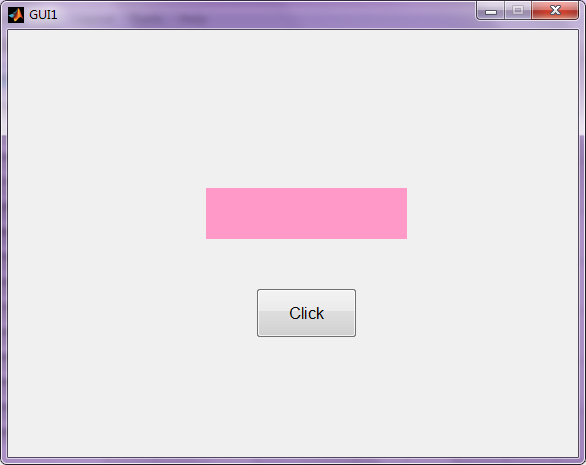
运行GUI程序，点击工具栏上的绿色方向按键，根据提示保存文件。

通过查看系统自动生成的M文件相应代码，了解GUI设计的基本流程，掌握GUI中主要的函数结构。

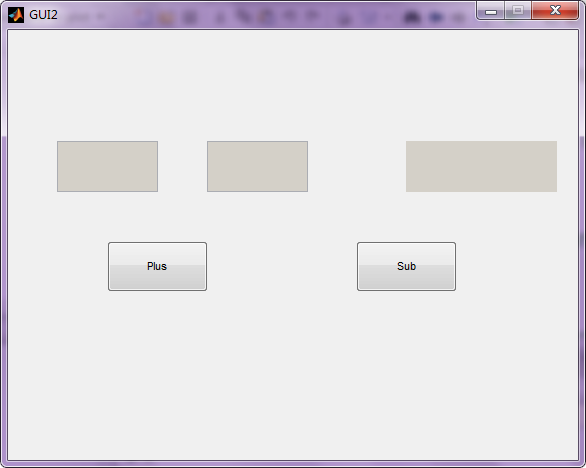
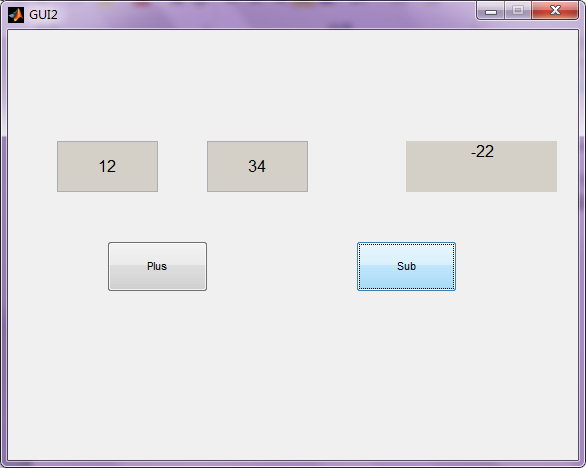
程序运行界面如下图所示：



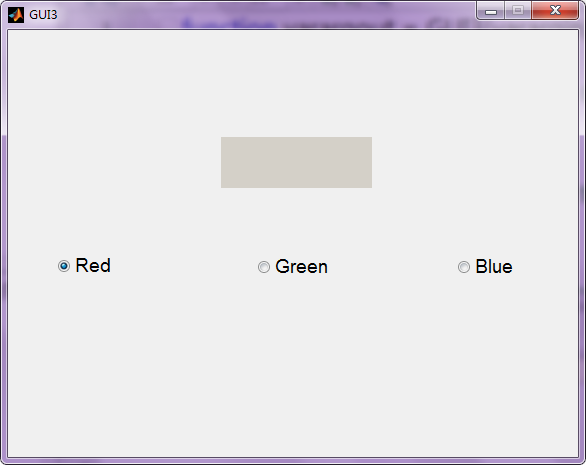
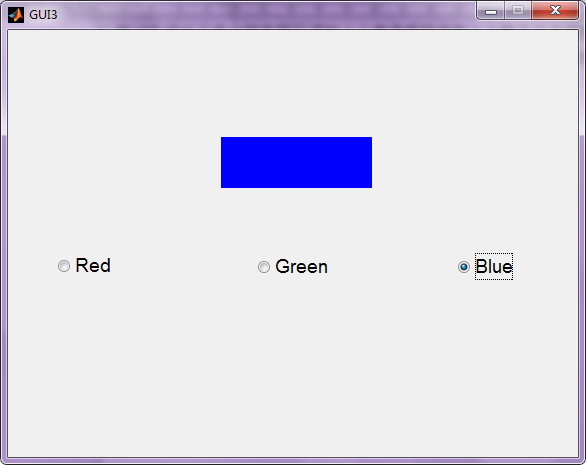
2、在GUI设计工作台的布局区添加静态文本框控件和按钮控件，并适当修改控件对象的属性。编写程序，实现功能：单击Push Button按钮，在静态文本框中显示单击按钮的次数。程序界面如下左图所示，运行结果如下右图所示。

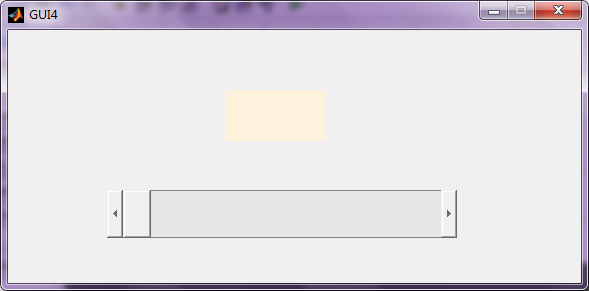
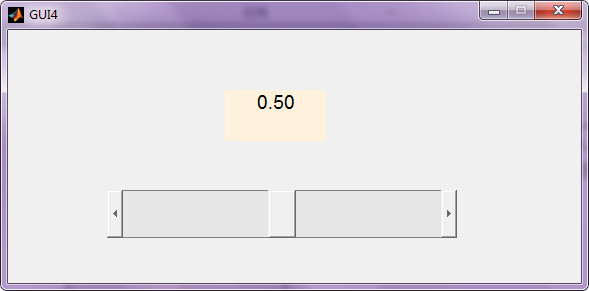
3、在GUI设计工作台的布局区添加2个文本编辑器控件、1个静态文本框控件和2个按钮控件，通过编程实现简易的加减法计算器功能。程序界面如下左图所示，运行结果如下右图所示。

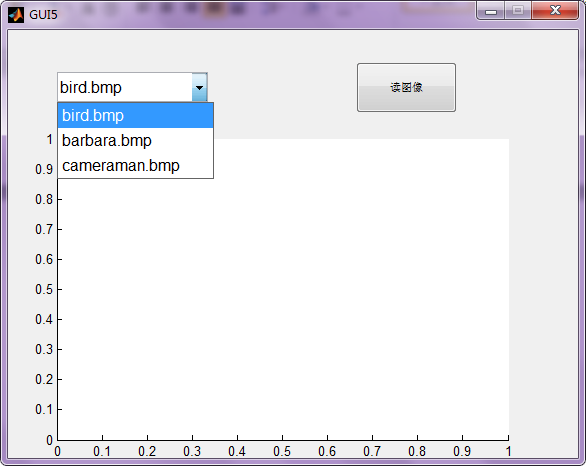
4、在GUI设计工作台的布局区添加3个单选框控件和1个静态文本框控件，通过编程实现使用3个单选按钮控制静态文本框的背景颜色。程序界面如下左图所示，运行结果如下右图所示。

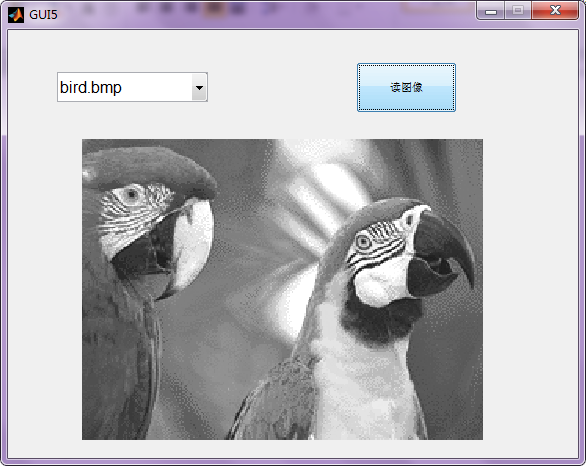
 

5、在GUI设计工作台的布局区添加1个静态文本框控件和1个滚动条控件，通过编程实现拖动滚动条时在静态文本框中显示对应滑块的位置变化，变化范围在0-1之间。程序界面如下左图所示，运行结果如下右图所示。

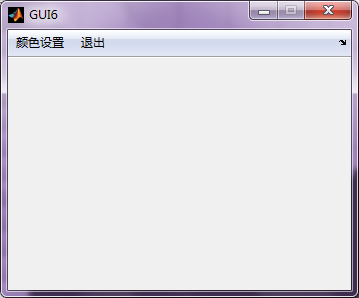
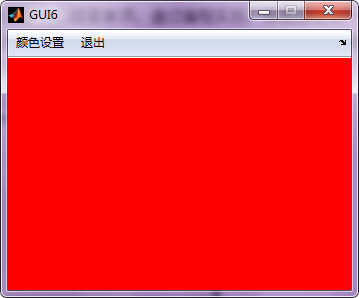
 

6、在GUI设计工作台的布局区添加1个弹出式菜单控件、1个坐标轴控件和1个按钮控件，通过编程实现：当选择弹出式菜单控件的其中一个图像名称时，并单击按钮控件，在坐标轴控件上显示相应的图像。程序界面如下左图所示，运行结果如下右图所示。





7、在GUI设计工作台的布局区打开菜单编辑器，添加“颜色设置”和“退出”2个一级菜单项，再在“颜色设置”一级菜单项下添加“红色”、“绿色”、“蓝色”3个二级菜单项，通过编程实现：当选择其中一个菜单项时，能够完成窗口颜色的相应变换或直接退出运行程序。程序界面如下左图所示，运行结果如下右图所示。

## 二、实验过程（代码）与实验结果

%%% 6-1

classdef test6\_1 < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

UIFigure matlab.ui.Figure

UpdateButton matlab.ui.control.Button

DropDown matlab.ui.control.DropDown

Label matlab.ui.control.Label

UIAxes matlab.ui.control.UIAxes

end

% Callbacks that handle component events

methods (Access = private)

% Button pushed function: UpdateButton

function UpdateButtonPushed(app, event)

% Get the selected function from DropDown

selectedFunction = app.DropDown.Value;

% Define x range for the plot

x = linspace(-2\*pi, 2\*pi, 100); % From -2π to 2π

% Depending on the selected function, plot the appropriate graph

switch selectedFunction

case 'Sin(x)'

y = sin(x);

case 'Cos(x)'

y = cos(x);

case 'x'

y = x;

case 'arcTan(x)'

y = atan(x); % Note: arctan values should be between -1 and 1

end

% Plot on UIAxes

plot(app.UIAxes, x, y);

title(app.UIAxes, ['Graph of ', selectedFunction]);

end

end

% Component initialization

methods (Access = private)

% Create UIFigure and components

function createComponents(app)

% Create UIFigure and hide until all components are created

app.UIFigure = uifigure('Visible', 'off');

app.UIFigure.Position = [100 100 640 480];

app.UIFigure.Name = 'MATLAB App';

% Create UIAxes

app.UIAxes = uiaxes(app.UIFigure);

title(app.UIAxes, '图片')

xlabel(app.UIAxes, 'X')

ylabel(app.UIAxes, 'Y')

zlabel(app.UIAxes, 'Z')

app.UIAxes.Position = [170 159 302 191];

% Create Label

app.Label = uilabel(app.UIFigure);

app.Label.HorizontalAlignment = 'right';

app.Label.Position = [201 362 41 22];

app.Label.Text = '函数名';

% Create DropDown

app.DropDown = uidropdown(app.UIFigure);

app.DropDown.Items = {'Sin(x)', 'Cos(x)', 'x', 'arcTan(x)'};

app.DropDown.Position = [257 362 100 22];

app.DropDown.Value = 'Sin(x)';

% Create UpdateButton

app.UpdateButton = uibutton(app.UIFigure, 'push');

app.UpdateButton.ButtonPushedFcn = createCallbackFcn(app, @UpdateButtonPushed, true);

app.UpdateButton.Position = [369 361 100 23];

app.UpdateButton.Text = 'Update';

% Show the figure after all components are created

app.UIFigure.Visible = 'on';

end

end

% App creation and deletion

methods (Access = public)

% Construct app

function app = test6\_1

% Create UIFigure and components

createComponents(app)

% Register the app with App Designer

registerApp(app, app.UIFigure)

if nargout == 0

clear app

end

end

% Code that executes before app deletion

function delete(app)

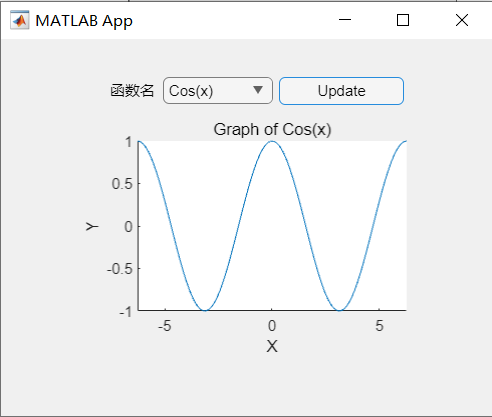
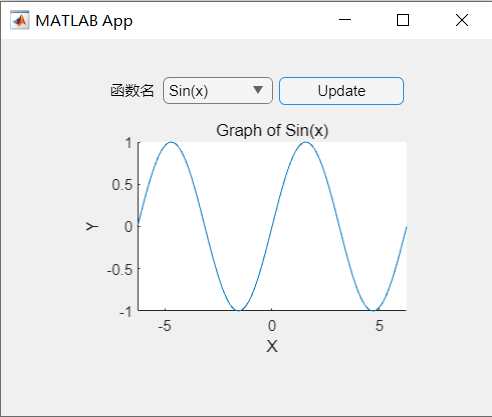
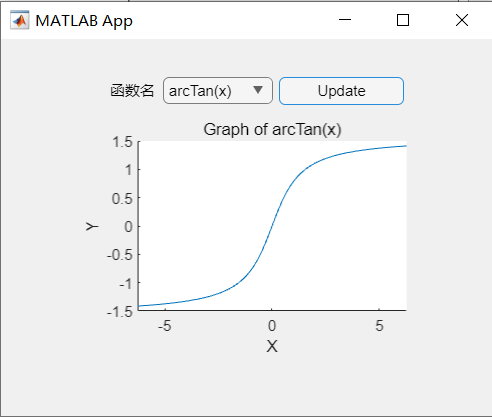
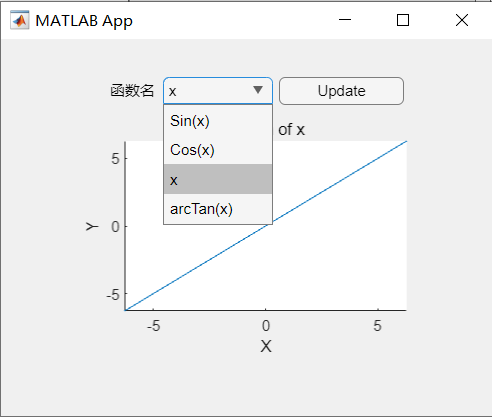
% Delete UIFigure when app is deleted

delete(app.UIFigure)

end

end

end

%%%6-2

classdef test6\_2 < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

UIFigure matlab.ui.Figure

Gauge matlab.ui.control.NinetyDegreeGauge

Label\_2 matlab.ui.control.Label

ClickButton matlab.ui.control.Button

TextArea matlab.ui.control.TextArea

Label matlab.ui.control.Label

end

properties (Access = public)

ClickCount = 0 % Initialize click count

end

% Callbacks that handle component events

methods (Access = private)

% Button pushed function: ClickButton

function ClickButtonPushed(app, event)

% Increment the click count

app.ClickCount = app.ClickCount + 1

% Update the TextArea with the new count

app.TextArea.Value = num2str(app.ClickCount)

% Update the Gauge to reflect the click frequency or count

app.Gauge.Value = app.ClickCount % This can be adjusted as needed

if(app.ClickCount == 100)

app.ClickCount = 0

app.TextArea.Value = "你真无聊！"

end

end

end

% Component initialization

methods (Access = private)

% Create UIFigure and components

function createComponents(app)

% Create UIFigure and hide until all components are created

app.UIFigure = uifigure('Visible', 'off');

app.UIFigure.Position = [100 100 640 480];

app.UIFigure.Name = 'MATLAB App';

% Create Label

app.Label = uilabel(app.UIFigure);

app.Label.HorizontalAlignment = 'right';

app.Label.Position = [242 327 53 22];

app.Label.Text = '按键次数';

% Create TextArea

app.TextArea = uitextarea(app.UIFigure);

app.TextArea.Position = [247 215 48 113];

% Create ClickButton

app.ClickButton = uibutton(app.UIFigure, 'push');

app.ClickButton.ButtonPushedFcn = createCallbackFcn(app, @ClickButtonPushed, true);

app.ClickButton.Position = [302 337 100 23];

app.ClickButton.Text = 'Click';

% Create Label\_2

app.Label\_2 = uilabel(app.UIFigure);

app.Label\_2.HorizontalAlignment = 'center';

app.Label\_2.Position = [326 178 53 22];

app.Label\_2.Text = '按键频率';

% Create Gauge

app.Gauge = uigauge(app.UIFigure, 'ninetydegree');

app.Gauge.Position = [307 215 90 90];

% Show the figure after all components are created

app.UIFigure.Visible = 'on';

end

end

% App creation and deletion

methods (Access = public)

% Construct app

function app = test6\_2

% Create UIFigure and components

createComponents(app)

% Register the app with App Designer

registerApp(app, app.UIFigure)

if nargout == 0

clear app

end

end

% Code that executes before app deletion

function delete(app)

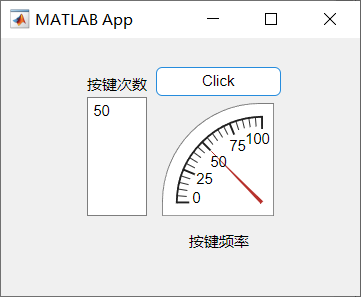
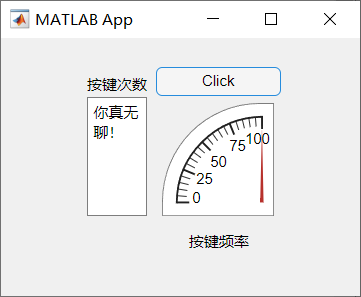
% Delete UIFigure when app is deleted

delete(app.UIFigure)

end

end

end

%%% 6-3

classdef test6\_3 < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

UIFigure matlab.ui.Figure

Label\_2 matlab.ui.control.Label

EditField\_3 matlab.ui.control.EditField

Label matlab.ui.control.Label

EditField\_2 matlab.ui.control.EditField

EditField matlab.ui.control.EditField

SubButton matlab.ui.control.Button

PlusButton matlab.ui.control.Button

end

% Callbacks that handle component events

methods (Access = private)

% Button pushed function: SubButton

function SubButtonPushed(app, event)

% 假设 app.EditField 是一个数值类型（例如 uitextfield）

% 如果要在 Label\_2 中显示字符串

app.Label\_2.Text = "-";

% 如果要从 EditField 和 EditField\_2 中得到数值并计算差异

value1 = str2double(app.EditField.Value);

value2 = str2double(app.EditField\_2.Value);

% 确保计算后的结果是数值，然后将其转换为字符串进行显示

result = value1 - value2;

% 如果 EditField\_3 是用于显示结果的文本框

app.EditField\_3.Value = num2str(result);

end

% Button pushed function: PlusButton

function PlusButtonPushed(app, event)

% 假设 app.EditField 是一个数值类型（例如 uitextfield）

% 如果要在 Label\_2 中显示字符串

app.Label\_2.Text = "+";

% 如果要从 EditField 和 EditField\_2 中得到数值并计算差异

value1 = str2double(app.EditField.Value);

value2 = str2double(app.EditField\_2.Value);

% 确保计算后的结果是数值，然后将其转换为字符串进行显示

result = value1 + value2;

% 如果 EditField\_3 是用于显示结果的文本框

app.EditField\_3.Value = num2str(result);

end

end

% Component initialization

methods (Access = private)

% Create UIFigure and components

function createComponents(app)

% Create UIFigure and hide until all components are created

app.UIFigure = uifigure('Visible', 'off');

app.UIFigure.Position = [100 100 640 480];

app.UIFigure.Name = 'MATLAB App';

% Create PlusButton

app.PlusButton = uibutton(app.UIFigure, 'push');

app.PlusButton.ButtonPushedFcn = createCallbackFcn(app, @PlusButtonPushed, true);

app.PlusButton.Position = [179 180 63 41];

app.PlusButton.Text = 'Plus';

% Create SubButton

app.SubButton = uibutton(app.UIFigure, 'push');

app.SubButton.ButtonPushedFcn = createCallbackFcn(app, @SubButtonPushed, true);

app.SubButton.Position = [306 180 63 41];

app.SubButton.Text = 'Sub';

% Create EditField

app.EditField = uieditfield(app.UIFigure, 'text');

app.EditField.Position = [157 245 51 50];

% Create EditField\_2

app.EditField\_2 = uieditfield(app.UIFigure, 'text');

app.EditField\_2.Position = [259 245 51 50];

% Create Label

app.Label = uilabel(app.UIFigure);

app.Label.FontSize = 48;

app.Label.Position = [321 239 33 63];

app.Label.Text = '=';

% Create EditField\_3

app.EditField\_3 = uieditfield(app.UIFigure, 'text');

app.EditField\_3.Position = [353 245 51 50];

% Create Label\_2

app.Label\_2 = uilabel(app.UIFigure);

app.Label\_2.FontSize = 48;

app.Label\_2.Position = [217 239 31 63];

app.Label\_2.Text = '±';

% Show the figure after all components are created

app.UIFigure.Visible = 'on';

end

end

% App creation and deletion

methods (Access = public)

% Construct app

function app = test6\_3

% Create UIFigure and components

createComponents(app)

% Register the app with App Designer

registerApp(app, app.UIFigure)

if nargout == 0

clear app

end

end

% Code that executes before app deletion

function delete(app)

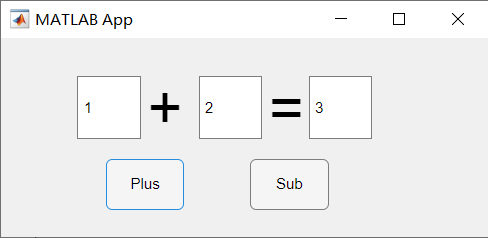
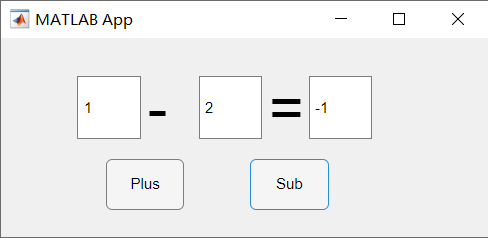
% Delete UIFigure when app is deleted

delete(app.UIFigure)

end

end

end

%%% 6-4

classdef test6\_4 < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

UIFigure matlab.ui.Figure

ButtonGroup matlab.ui.container.ButtonGroup

Button\_4 matlab.ui.control.RadioButton

Image matlab.ui.control.Image

Button\_3 matlab.ui.control.RadioButton

Button\_2 matlab.ui.control.RadioButton

Button matlab.ui.control.RadioButton

end

% Callbacks that handle component events

methods (Access = private)

% Key press function: UIFigure

function UIFigureKeyPress(app, event)

% 获取选择的单选按钮

selectedButton = app.ButtonGroup.SelectedObject.Text;

% 根据选择的按钮改变 Image 的背景颜色

switch selectedButton

case '红'

% 将背景颜色设置为红色

app.Image.BackgroundColor = [1 0 0];

case '绿'

% 将背景颜色设置为绿色

app.Image.BackgroundColor = [0 1 0];

case '蓝'

% 将背景颜色设置为蓝色

app.Image.BackgroundColor = [0 0 1];

otherwise

% 默认颜色（黄色）

app.Image.BackgroundColor = [1 1 0];

end

end

% Button down function: UIFigure

function UIFigureButtonDown(app, event)

end

% Selection changed function: ButtonGroup

function ButtonGroupSelectionChanged(app, event)

% 获取选择的单选按钮

selectedButton = app.ButtonGroup.SelectedObject.Text;

% 根据选择的按钮改变 Image 的背景颜色

switch selectedButton

case '红'

% 将背景颜色设置为红色

app.Image.BackgroundColor = [1 0 0];

app.Image.ImageSource=""

case '绿'

% 将背景颜色设置为绿色

app.Image.BackgroundColor = [0 1 0];

app.Image.ImageSource=""

case '蓝'

% 将背景颜色设置为蓝色

app.Image.BackgroundColor = [0 0 1];

app.Image.ImageSource=""

case '图片'

app.Image.BackgroundColor = [0 0 0];

app.Image.ImageSource="./1.jpg"

otherwise

% 默认颜色（黄色）

app.Image.BackgroundColor = [1 1 0];

end

end

end

% Component initialization

methods (Access = private)

% Create UIFigure and components

function createComponents(app)

% Create UIFigure and hide until all components are created

app.UIFigure = uifigure('Visible', 'off');

app.UIFigure.Position = [100 100 640 480];

app.UIFigure.Name = 'MATLAB App';

app.UIFigure.ButtonDownFcn = createCallbackFcn(app, @UIFigureButtonDown, true);

app.UIFigure.KeyPressFcn = createCallbackFcn(app, @UIFigureKeyPress, true);

% Create ButtonGroup

app.ButtonGroup = uibuttongroup(app.UIFigure);

app.ButtonGroup.SelectionChangedFcn = createCallbackFcn(app, @ButtonGroupSelectionChanged, true);

app.ButtonGroup.Title = '颜色';

app.ButtonGroup.Position = [177 170 286 114];

% Create Button

app.Button = uiradiobutton(app.ButtonGroup);

app.Button.Text = '红';

app.Button.Position = [11 68 58 22];

app.Button.Value = true;

% Create Button\_2

app.Button\_2 = uiradiobutton(app.ButtonGroup);

app.Button\_2.Text = '绿';

app.Button\_2.Position = [11 46 65 22];

% Create Button\_3

app.Button\_3 = uiradiobutton(app.ButtonGroup);

app.Button\_3.Text = '蓝';

app.Button\_3.Position = [11 24 65 22];

% Create Image

app.Image = uiimage(app.ButtonGroup);

app.Image.BackgroundColor = [1 1 0];

app.Image.Position = [109 3 150 85];

% Create Button\_4

app.Button\_4 = uiradiobutton(app.ButtonGroup);

app.Button\_4.Text = '图片';

app.Button\_4.Position = [11 3 65 22];

% Show the figure after all components are created

app.UIFigure.Visible = 'on';

end

end

% App creation and deletion

methods (Access = public)

% Construct app

function app = test6\_4

% Create UIFigure and components

createComponents(app)

% Register the app with App Designer

registerApp(app, app.UIFigure)

if nargout == 0

clear app

end

end

% Code that executes before app deletion

function delete(app)

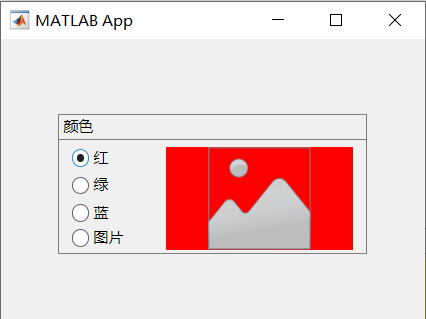
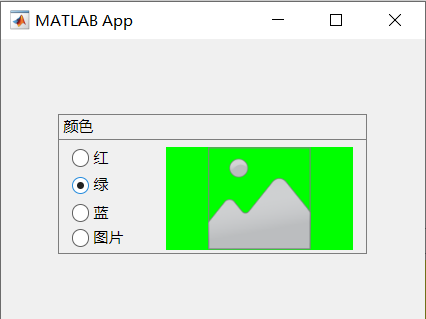
% Delete UIFigure when app is deleted

delete(app.UIFigure)

end

end

end

%%% 6-5

classdef test6\_5 < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

UIFigure matlab.ui.Figure

TextArea matlab.ui.control.TextArea

Label matlab.ui.control.Label

Slider matlab.ui.control.Slider

SliderLabel matlab.ui.control.Label

end

% Callbacks that handle component events

methods (Access = private)

% Value changed function: Slider

function SliderValueChanged(app, event)

value = app.Slider.Value;

app.TextArea.Value = string(value/100);

end

end

% Component initialization

methods (Access = private)

% Create UIFigure and components

function createComponents(app)

% Create UIFigure and hide until all components are created

app.UIFigure = uifigure('Visible', 'off');

app.UIFigure.Position = [100 100 640 480];

app.UIFigure.Name = 'MATLAB App';

% Create SliderLabel

app.SliderLabel = uilabel(app.UIFigure);

app.SliderLabel.HorizontalAlignment = 'right';

app.SliderLabel.Position = [172 191 36 22];

app.SliderLabel.Text = 'Slider';

% Create Slider

app.Slider = uislider(app.UIFigure);

app.Slider.ValueChangedFcn = createCallbackFcn(app, @SliderValueChanged, true);

app.Slider.Position = [229 200 150 3];

% Create Label

app.Label = uilabel(app.UIFigure);

app.Label.HorizontalAlignment = 'right';

app.Label.Position = [185 272 29 22];

app.Label.Text = '数字';

% Create TextArea

app.TextArea = uitextarea(app.UIFigure);

app.TextArea.Position = [229 236 150 60];

% Show the figure after all components are created

app.UIFigure.Visible = 'on';

end

end

% App creation and deletion

methods (Access = public)

% Construct app

function app = test6\_5

% Create UIFigure and components

createComponents(app)

% Register the app with App Designer

registerApp(app, app.UIFigure)

if nargout == 0

clear app

end

end

% Code that executes before app deletion

function delete(app)

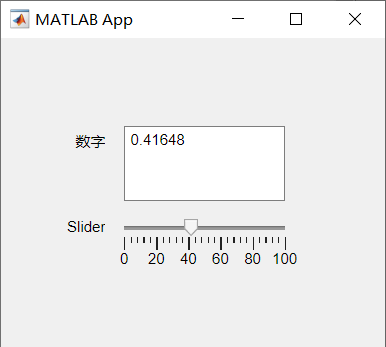
% Delete UIFigure when app is deleted

delete(app.UIFigure)

end

end

end



%%% 6-6

classdef test6\_6 < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

UIFigure matlab.ui.Figure

Button matlab.ui.control.Button

DropDown matlab.ui.control.DropDown

DropDownLabel matlab.ui.control.Label

UIAxes matlab.ui.control.UIAxes

end

% Callbacks that handle component events

methods (Access = private)

% Button pushed function: Button

function ButtonPushed(app, event)

% Get the selected item from the dropdown

selectedItem = app.DropDown.Value;

% Load and display the corresponding image

switch selectedItem

case '1'

image(app.UIAxes, imread('./1.jpg')); % 显示图像1

case '2'

image(app.UIAxes, imread('./2.jpg')); % 显示图像2

case '3'

image(app.UIAxes, imread('./3.jpg')); % 显示图像3

end

end

end

% Component initialization

methods (Access = private)

% Create UIFigure and components

function createComponents(app)

% Create UIFigure and hide until all components are created

app.UIFigure = uifigure('Visible', 'off');

app.UIFigure.Position = [100 100 640 480];

app.UIFigure.Name = 'MATLAB App';

% Create UIAxes

app.UIAxes = uiaxes(app.UIFigure);

app.UIAxes.Position = [161 92 300 185];

% Create DropDownLabel

app.DropDownLabel = uilabel(app.UIFigure);

app.DropDownLabel.HorizontalAlignment = 'right';

app.DropDownLabel.Position = [182 276 65 22];

app.DropDownLabel.Text = 'Drop Down';

% Create DropDown

app.DropDown = uidropdown(app.UIFigure);

app.DropDown.Items = {'1', '2', '3'};

app.DropDown.Position = [262 276 83 22];

app.DropDown.Value = '1';

% Create Button

app.Button = uibutton(app.UIFigure, 'push');

app.Button.ButtonPushedFcn = createCallbackFcn(app, @ButtonPushed, true);

app.Button.Position = [375 276 74 23];

app.Button.Text = '显示';

% Show the figure after all components are created

app.UIFigure.Visible = 'on';

end

end

% App creation and deletion

methods (Access = public)

% Construct app

function app = test6\_6

% Create UIFigure and components

createComponents(app)

% Register the app with App Designer

registerApp(app, app.UIFigure)

if nargout == 0

clear app

end

end

% Code that executes before app deletion

function delete(app)

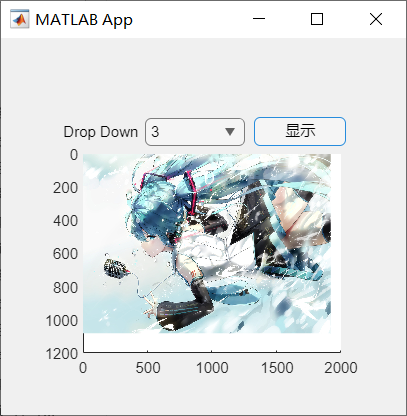
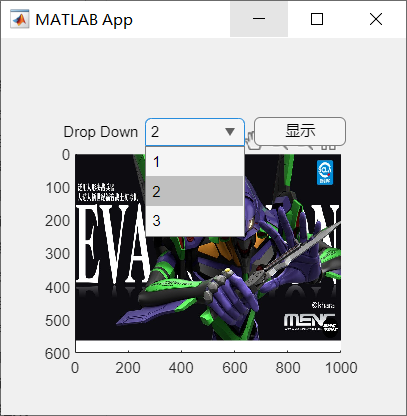
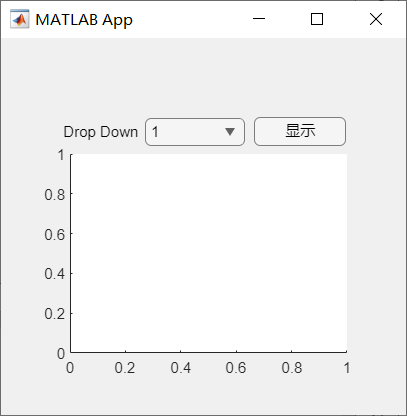
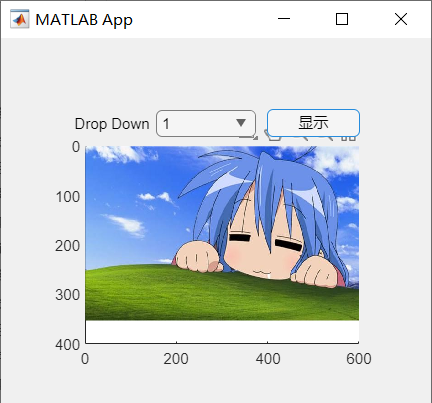
% Delete UIFigure when app is deleted

delete(app.UIFigure)

end

end

end

%%% 6-7

classdef test6\_7 < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

UIFigure matlab.ui.Figure

Menu matlab.ui.container.Menu

Menu\_3 matlab.ui.container.Menu

Menu\_4 matlab.ui.container.Menu

Menu\_5 matlab.ui.container.Menu

Menu\_2 matlab.ui.container.Menu

end

% Callbacks that handle component events

methods (Access = private)

% Menu selected function: Menu\_3

function Menu\_3Selected(app, event)

app.UIFigure.Color = [1,0,0]

end

% Menu selected function: Menu\_4

function Menu\_4Selected(app, event)

app.UIFigure.Color = [0,1,0]

end

% Menu selected function: Menu\_5

function Menu\_5Selected(app, event)

app.UIFigure.Color = [0,0,1]

end

% Menu selected function: Menu\_2

function Menu\_2Selected(app, event)

close(app.UIFigure)

end

end

% Component initialization

methods (Access = private)

% Create UIFigure and components

function createComponents(app)

% Create UIFigure and hide until all components are created

app.UIFigure = uifigure('Visible', 'off');

app.UIFigure.Color = [1 1 1];

app.UIFigure.Position = [100 100 640 480];

app.UIFigure.Name = 'MATLAB App';

% Create Menu

app.Menu = uimenu(app.UIFigure);

app.Menu.Text = '颜色设置';

% Create Menu\_3

app.Menu\_3 = uimenu(app.Menu);

app.Menu\_3.MenuSelectedFcn = createCallbackFcn(app, @Menu\_3Selected, true);

app.Menu\_3.Text = '红';

% Create Menu\_4

app.Menu\_4 = uimenu(app.Menu);

app.Menu\_4.MenuSelectedFcn = createCallbackFcn(app, @Menu\_4Selected, true);

app.Menu\_4.Text = '绿';

% Create Menu\_5

app.Menu\_5 = uimenu(app.Menu);

app.Menu\_5.MenuSelectedFcn = createCallbackFcn(app, @Menu\_5Selected, true);

app.Menu\_5.Text = '蓝';

% Create Menu\_2

app.Menu\_2 = uimenu(app.UIFigure);

app.Menu\_2.MenuSelectedFcn = createCallbackFcn(app, @Menu\_2Selected, true);

app.Menu\_2.Text = '退出';

% Show the figure after all components are created

app.UIFigure.Visible = 'on';

end

end

% App creation and deletion

methods (Access = public)

% Construct app

function app = test6\_7

% Create UIFigure and components

createComponents(app)

% Register the app with App Designer

registerApp(app, app.UIFigure)

if nargout == 0

clear app

end

end

% Code that executes before app deletion

function delete(app)

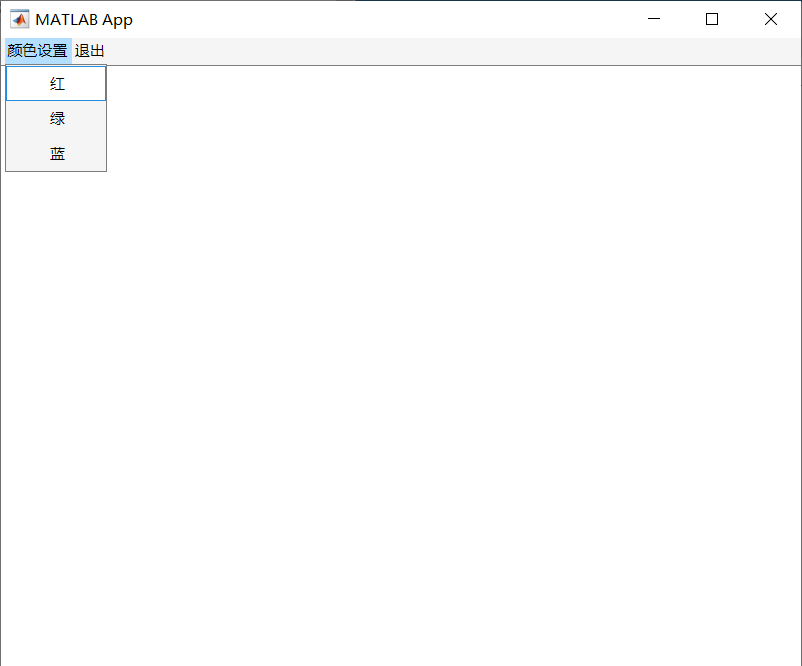
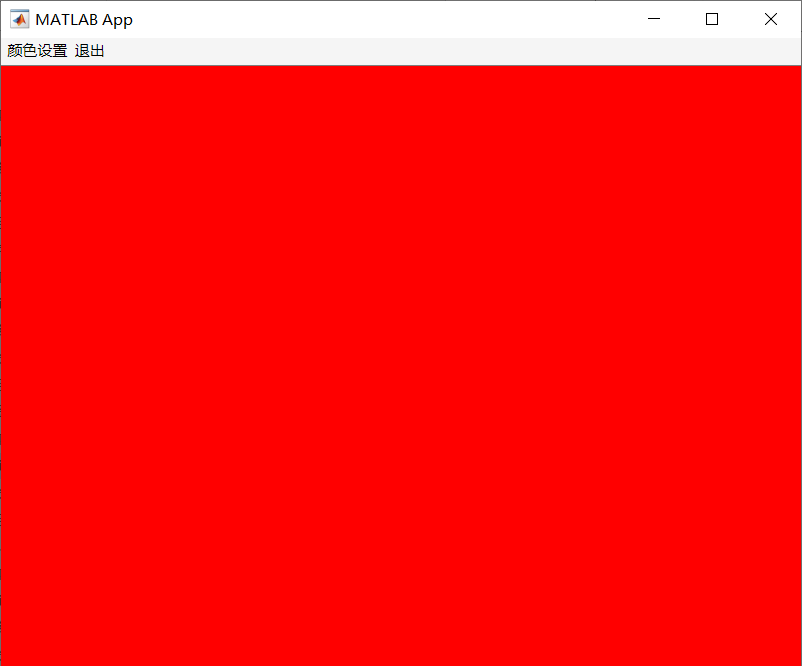
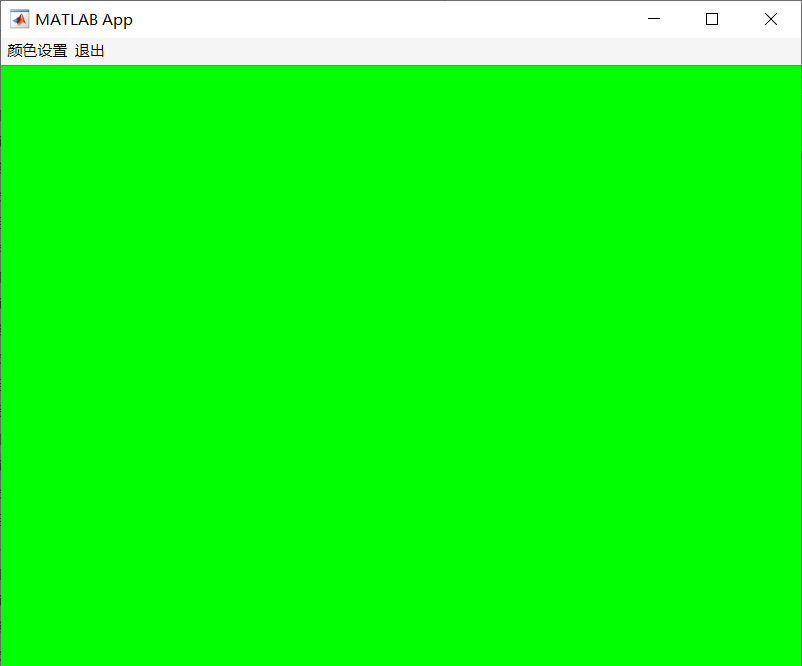
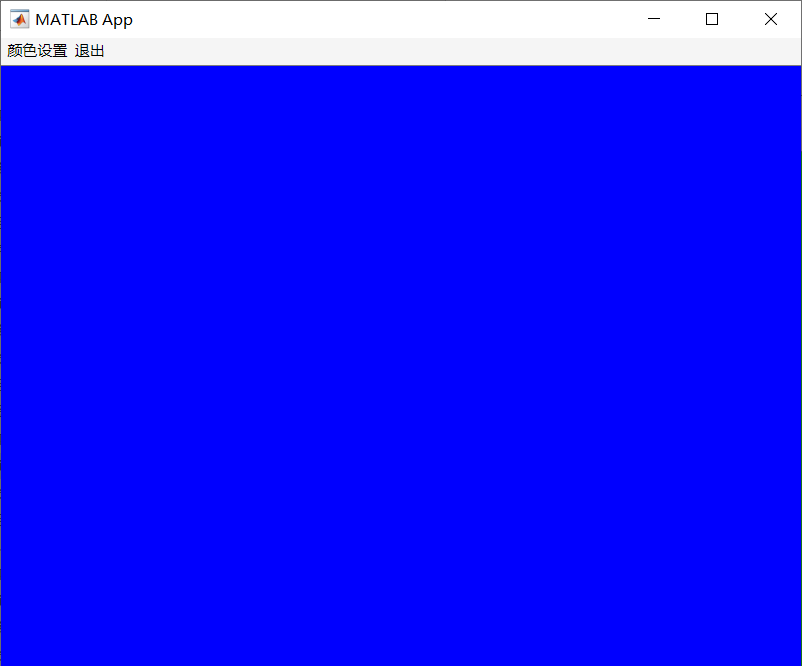
% Delete UIFigure when app is deleted

delete(app.UIFigure)

end

end

end

## 三、结果分析

代码正常运行

## 四、心得体会

App Designer是MATLAB中用于创建图形用户界面(GUI)的现代化工具，它比传统的GUIDE更灵活、功能更强大，支持拖放控件并提供了更简洁的代码组织方式。

# 实验七 系统的频域与时域分析

## 一、实验内容

1、表示下列传递函数模型,并转化成其他的数学模型

(1)

(2)

(3))

(4)

2、一个单位负反馈开环传递函数为，试绘出系统闭环的根轨迹图；并在跟轨迹图上任选一点，试计算该点的增益及其所有极点的位置。

3、已知某负反馈系统的前向通路传递函数为，反馈通路传递函数为。绘制系统的单位阶跃响应曲线，并计算上升时间，峰值时间，超调量，延迟时间。

## 二、实验过程（代码）

%% 7-1-1

% 定义传递函数

% 简化分子多项式

num = [4, 8, 4, 96, 144, 144]; % 降低分子阶数

den = [1, 3, 5, 6, 2, 5];

G = tf(num, den);

% 零极点图

figure;

pzmap(G);

title('零极点图');

% 频率响应 - Bode 图

figure;

bode(G);

title('Bode 图');

% 时间响应 - 阶跃响应

figure;

step(G);

title('阶跃响应');

%% 7-1-2

% 定义传递函数

num = [4, -2];

den = [1, 0, 2, 5];

G = tf(num, den);

% 零极点图

figure;

pzmap(G);

title('零极点图');

% 频率响应 - Bode 图

figure;

bode(G);

title('Bode 图');

% 时间响应 - 阶跃响应

figure;

step(G);

title('阶跃响应');

%% 7-1-3

% 定义传递函数

num = [1];

den = [2, 5, 2];

G = tf(num, den);

% 零极点图

figure;

pzmap(G);

title('零极点图');

% 频率响应 - Bode 图

figure;

bode(G);

title('Bode 图');

% 时间响应 - 阶跃响应

figure;

step(G);

title('阶跃响应');

%% 7-1-4

% 定义状态空间模型

A = [0, 1; -1, -2];

B = [0; 1];

C = [0, 1];

D = [0];

% 转换为传递函数

[num, den] = ss2tf(A, B, C, D);

G = tf(num, den);

% 零极点图

figure;

pzmap(G);

title('零极点图');

% 频率响应 - Bode 图

figure;

bode(G);

title('Bode 图');

% 时间响应 - 阶跃响应

figure;

step(G);

title('阶跃响应');

%% 7-2

% 定义开环传递函数

num = [1]; % k 在根轨迹图上是增益，通常不指定

den = [0.5, 1, 0, 4, 1];

G = tf(num, den);

% 绘制根轨迹图

figure;

rlocus(G);

title('根轨迹图');

%% 7-3

% 定义前向通路传递函数

G = tf(10, [1, 0, 1]);

% 定义反馈通路传递函数

H\_f = tf([0.3, 1], 1);

% 计算闭环传递函数

T = feedback(G, H\_f); % 负反馈系统

% 绘制单位阶跃响应曲线

figure;

[response, time] = step(T); % 获取阶跃响应

plot(time, response);

title('单位阶跃响应');

xlabel('时间 (秒)');

ylabel('响应');

grid on;

% 计算上升时间、峰值时间、超调量

S = stepinfo(T);

% 上升时间、峰值时间、超调量

disp('上升时间: ');

disp(S.RiseTime);

disp('峰值时间: ');

disp(S.PeakTime);

disp('超调量: ');

disp(S.Overshoot);

% 计算延迟时间

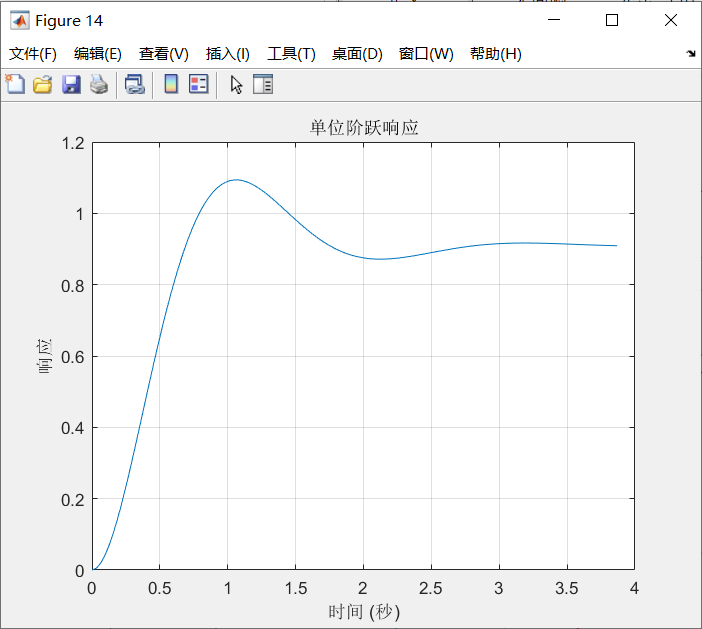
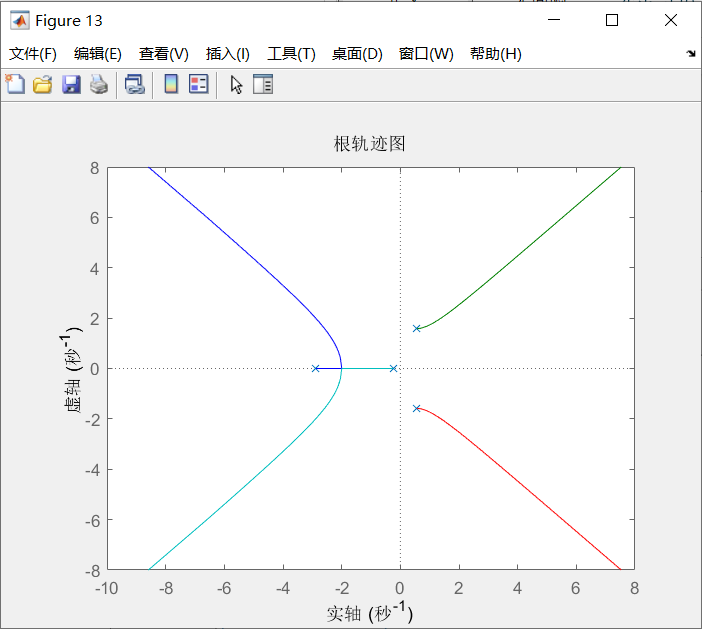
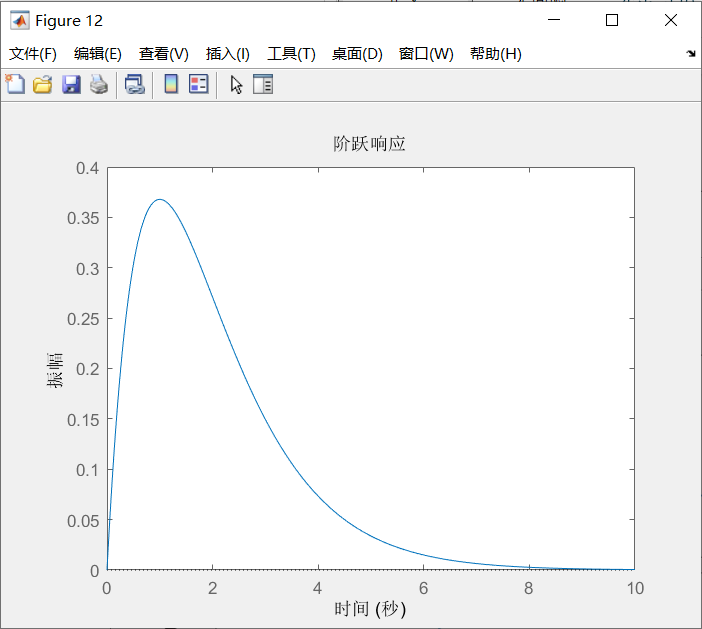
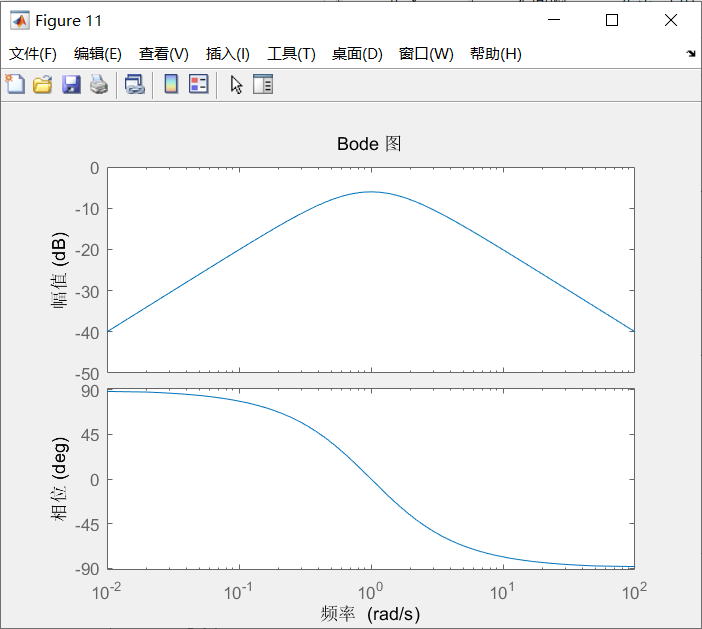
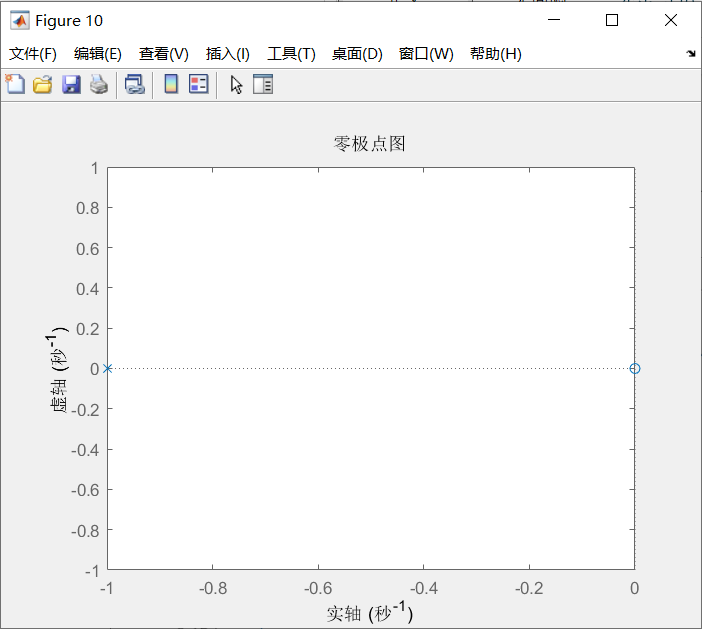
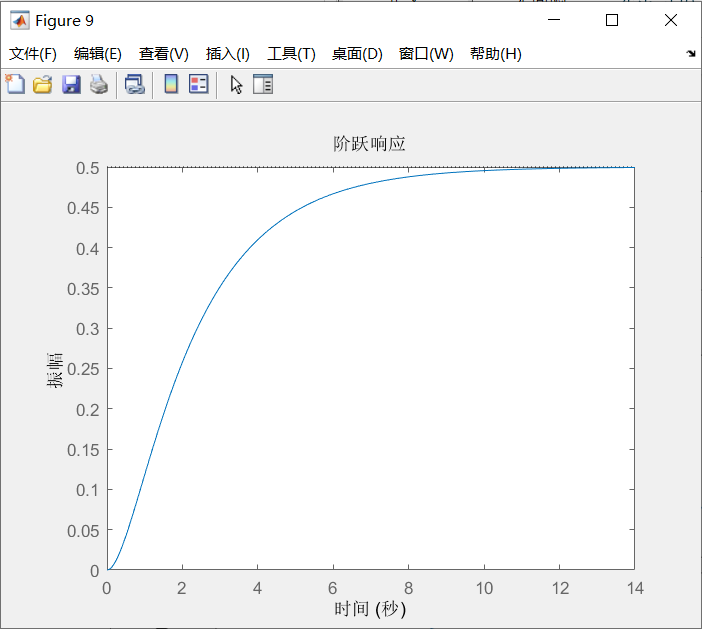
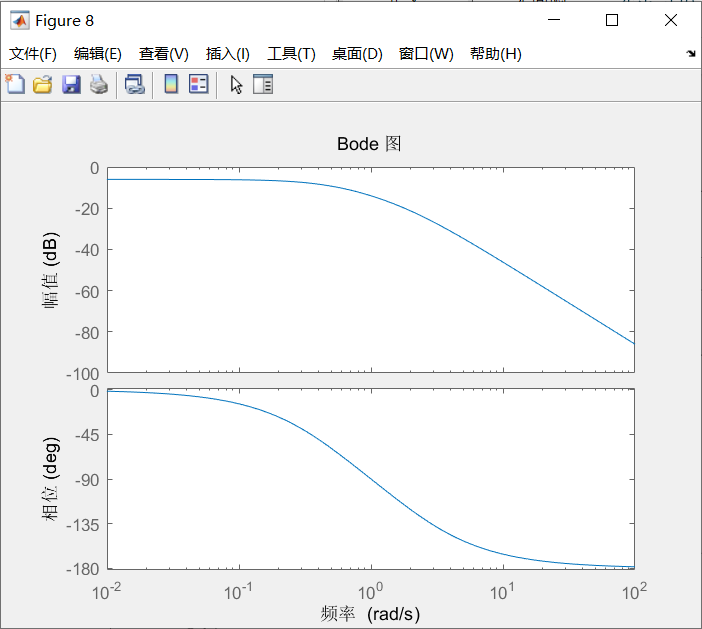
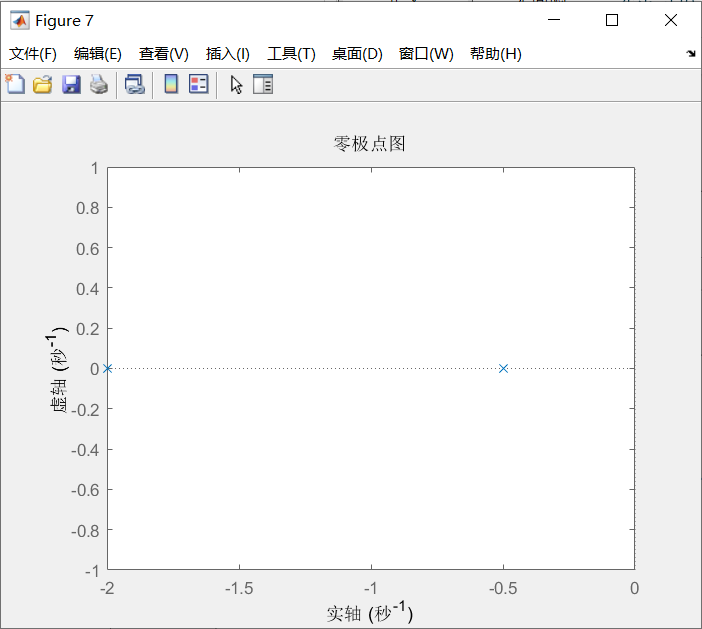
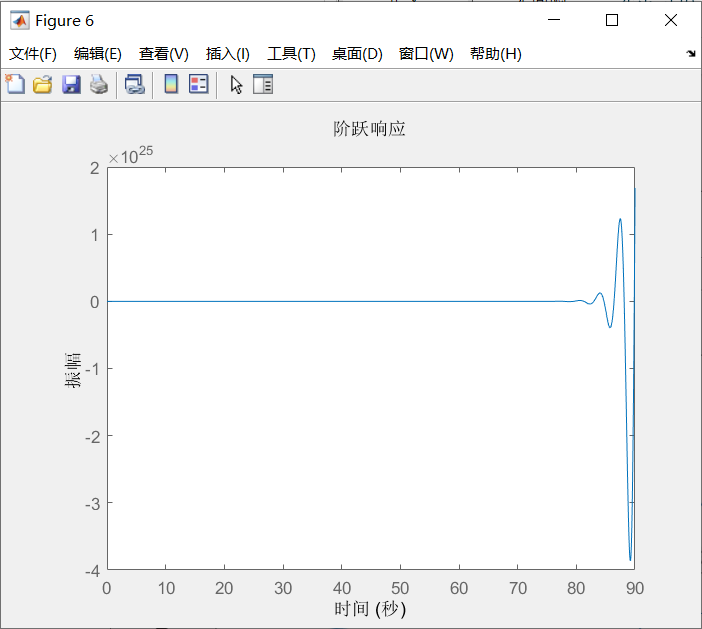
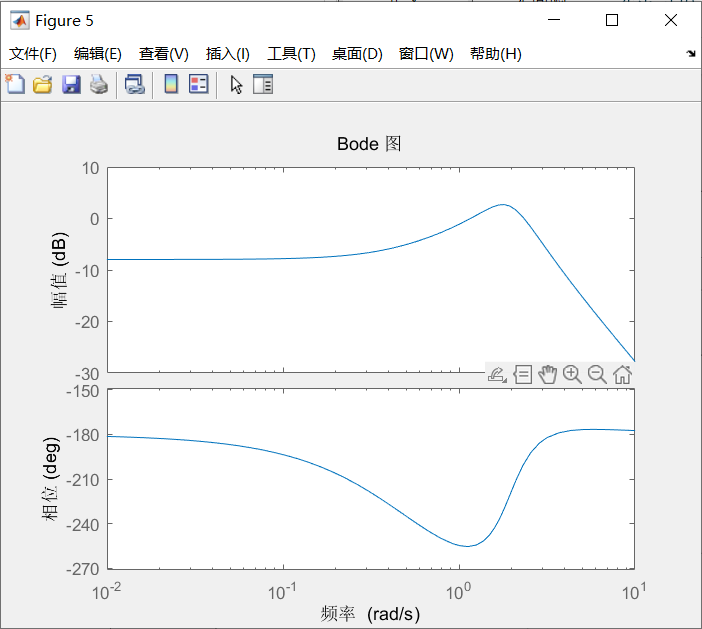
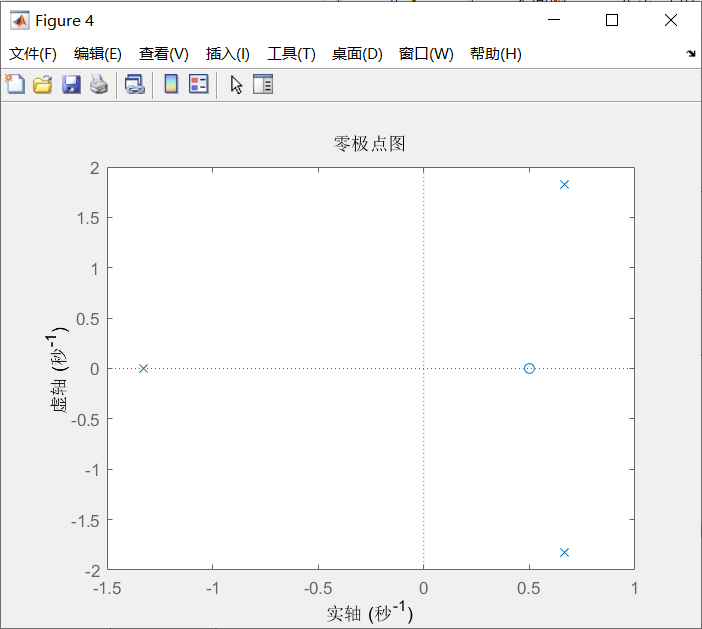
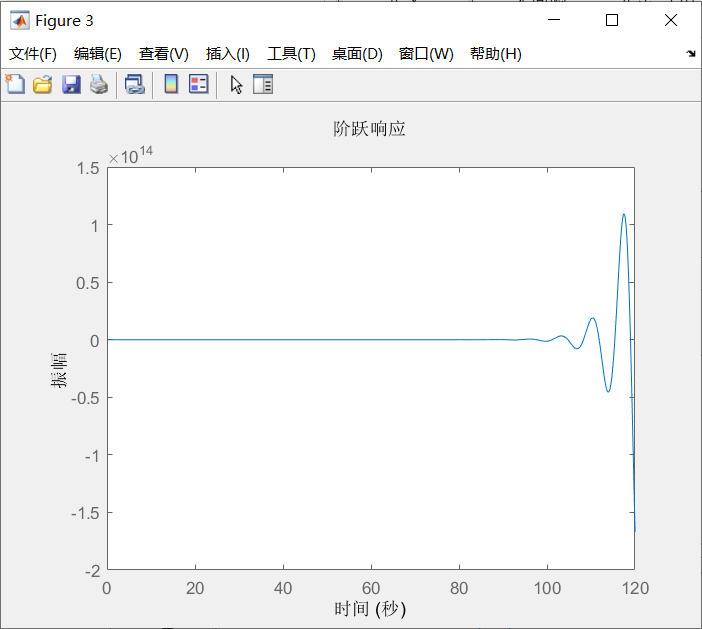
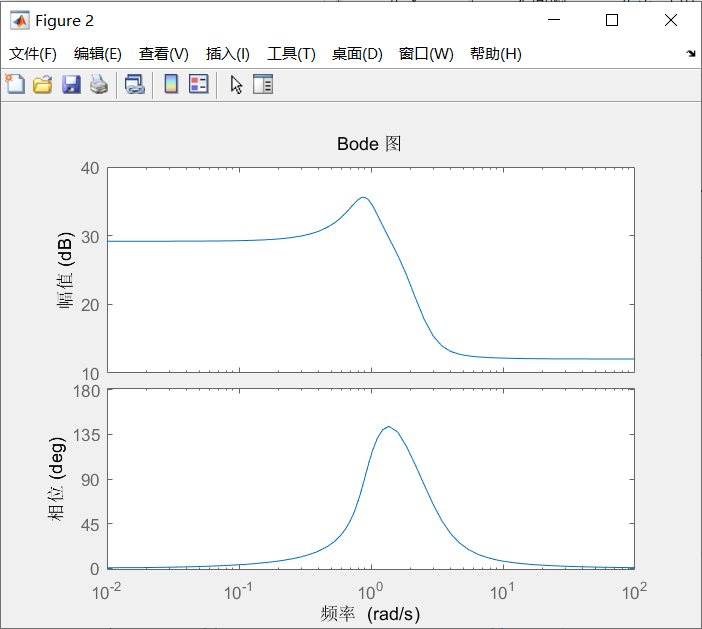
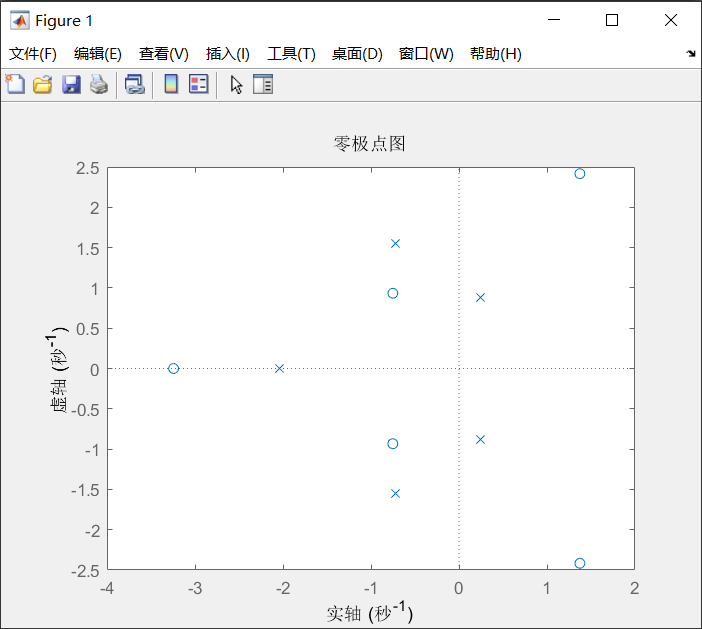
threshold = 0.1 \* response(end); % 阈值设为稳态的10%

delay\_time = time(find(response >= threshold, 1)); % 首次超过阈值的时间

disp('延迟时间: ');

disp(delay\_time);

## 三、实验结果



## 四、结果分析

在系统分析中，频域和时域是两个重要的视角。通过实验，可以理解通信系统的数学模型描述，学习系统稳定性分析的方法，并掌握在频域和时域中分析系统的基本方法。

## 五、心得体会

**频域和时域是互补的分析方法：**

在系统分析中，频域和时域提供了不同的视角。频域分析侧重于系统的频率响应，而时域分析关注系统的时间响应。理解两者的互补性，有助于更全面地分析系统。

**MATLAB是系统分析的强大工具：**

MATLAB提供了丰富的工具用于系统分析，包括传递函数、状态空间模型、频域分析和时域分析。通过MATLAB，可以快速构建模型并进行各种分析。

**稳定性分析至关重要:**

通信系统的稳定性是确保其正常运行的关键。通过零极点分析、Routh-Hurwitz判据、根轨迹图等，可以有效评估系统的稳定性。

**系统响应分析提供了直观的洞察:**

通过观察系统的阶跃响应和冲激响应，可以直观地了解系统在时间域的行为。这有助于评估系统的性能，并指导系统的设计与调整。

# 实验八　Simulink在通信系统中的应用

## 一、实验内容

1、信号平方运算。系统的功能是对输入的信号进行平方运算。现要求建立系统的Simulink模型并进行简单的仿真分析。具体要求如下：

（1）系统输入信号源：幅值为2的正弦波。

（2）使用Scope 显示原始信号和结果信号。

（3）生成系统运算部分的子系统

（4）生成信号标签并传递。

对于如下的一个系统

其中为系统的输入变量，为时间变量，*y*为系统的输出变量。输入变量。很显然，此系统服从简单系统的条件，为一简单系统。系统输出仅由系统当前时刻的输入决定。

2、调用MATLAB工作空间中的信号矩阵作为模型输入。所需的输入为

3、在仿真启动时，积分器从零开始对0.5t进行积分。当复位口信号t-5=0瞬间，积分器被重置为零。此后，再对0.5（t-5）进行积分。

4、假设从实际自然界（力学、电学、生态等）或社会中，抽象出有初始状态为0的二阶微分程，是单位阶跃函数。求如何用积分器直接构搭求解该微分方程的模型。

5、对二阶微分程进行拉氏变换**：**

6、建立数字滤波器系统模型

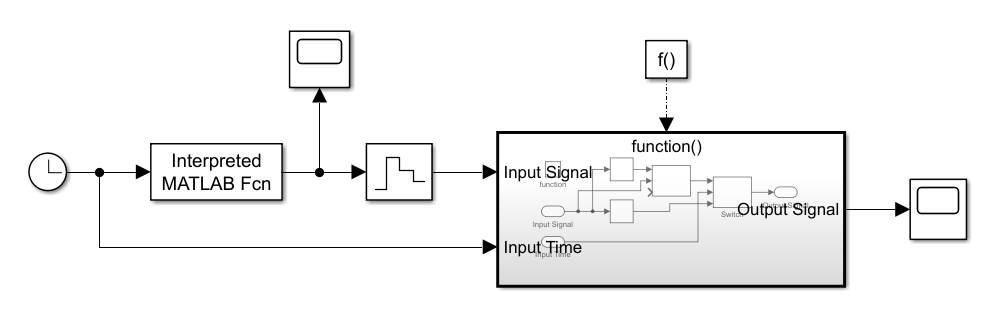
这里使用简单的通信系统说明低通数字滤波器的功能。在此系统中，发送方首先使用高频正弦波对一低频锯齿波进行幅度调制，然后在无损信道中传递此幅度调制信号；接收方在接受到幅度调制信号后，首先对其进行解调，然后使用低通数字滤波器对解调后的信号进行滤波以获得低频锯齿波信号。

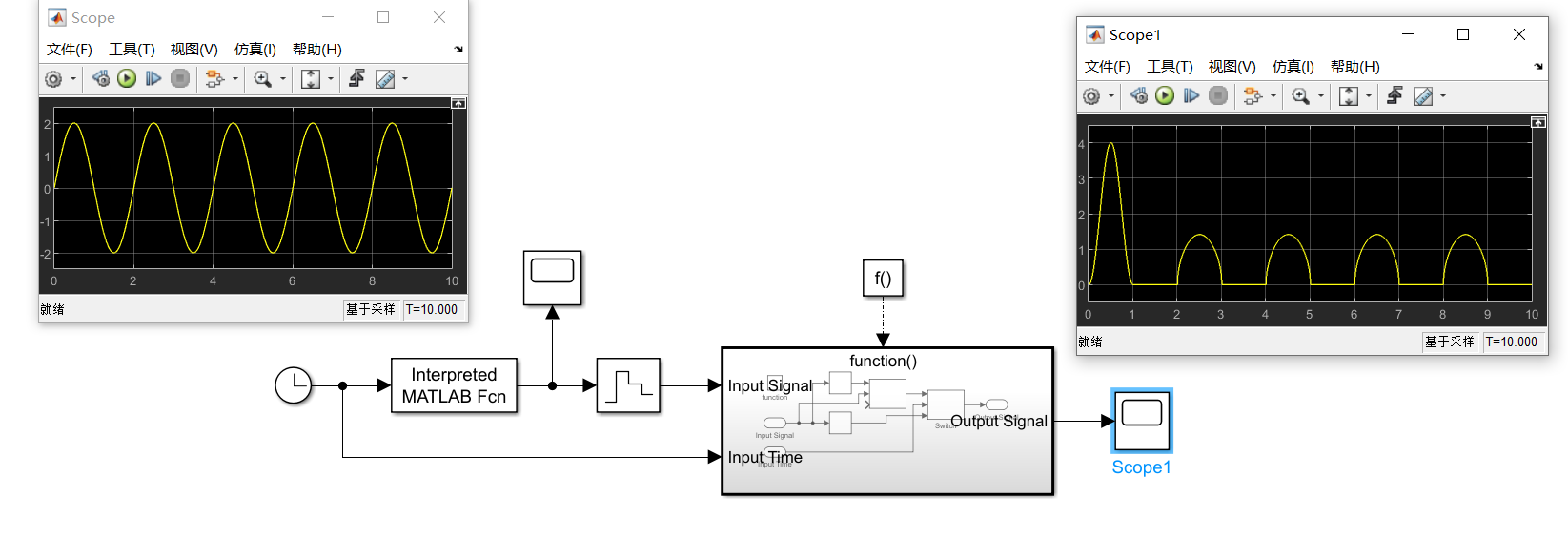
7.利用SIMULINK仿真标准AM调制解调过程，即实现，并使用示波器观察原始信号、调制信号以及解调信号的波形。设，，。参考仿真模型如下，自行利用MATLAB帮助系统查阅关于参考模型中各个模块的有关说明，以及设置适当的低通滤波器参数。



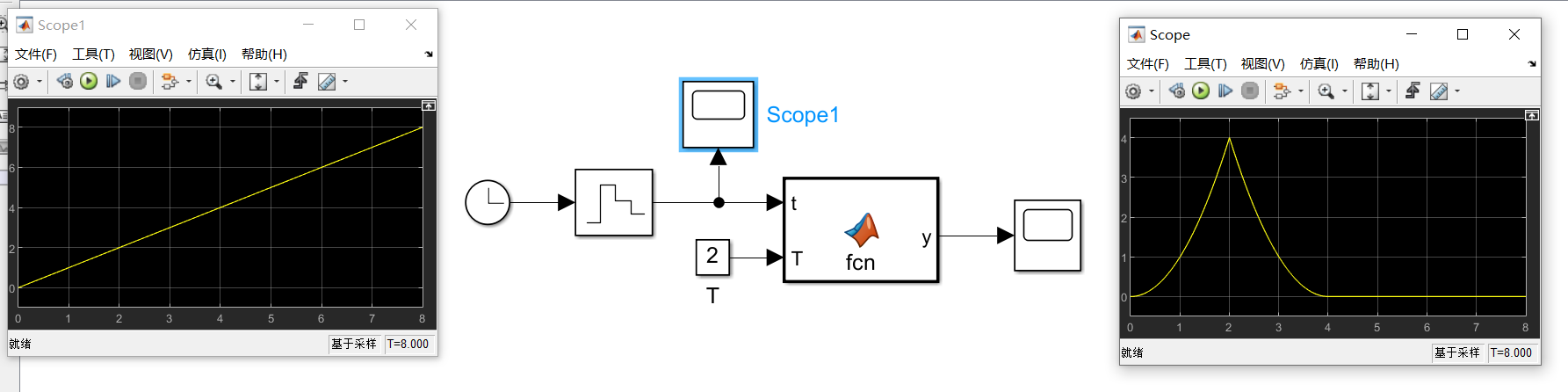
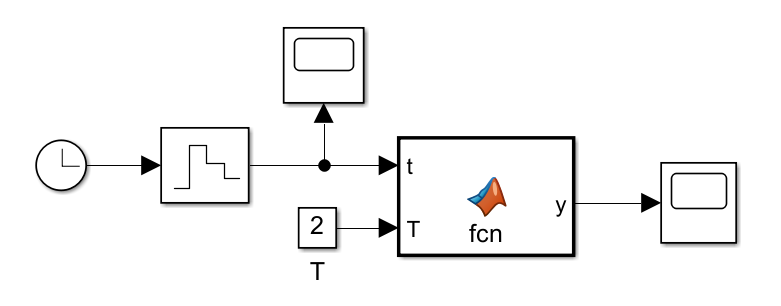
## 二、实验电路及结果

### （1）信号平方运算

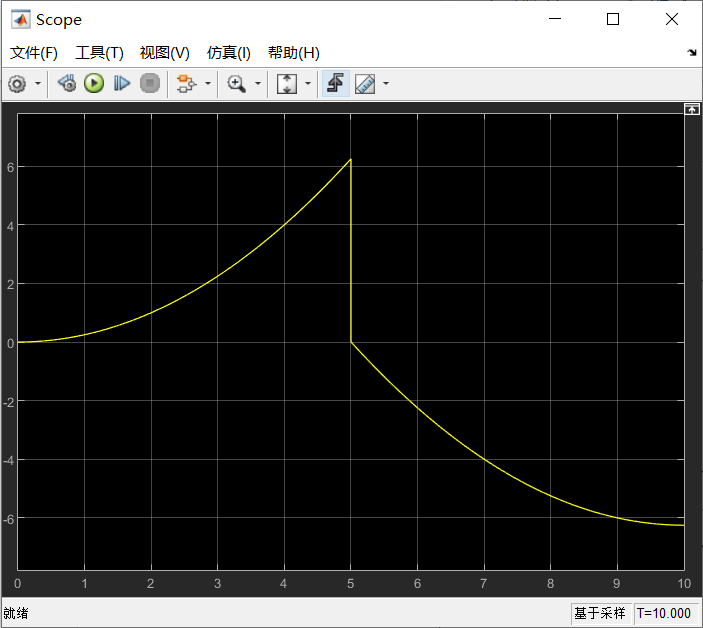




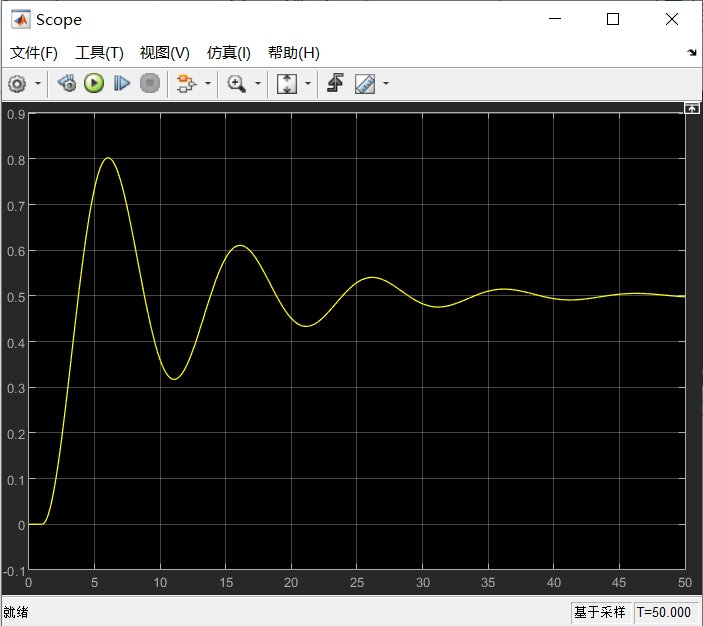
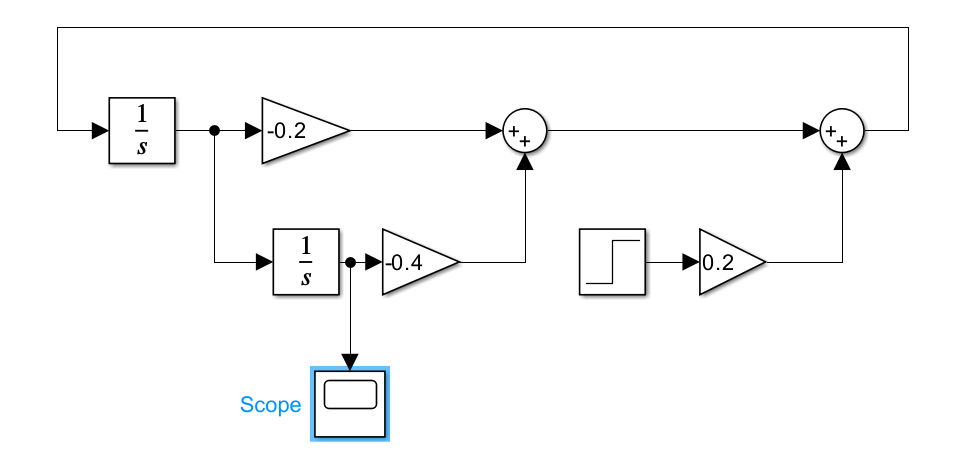
### （2）调用MATLAB工作空间中的信号矩阵作为模型输入



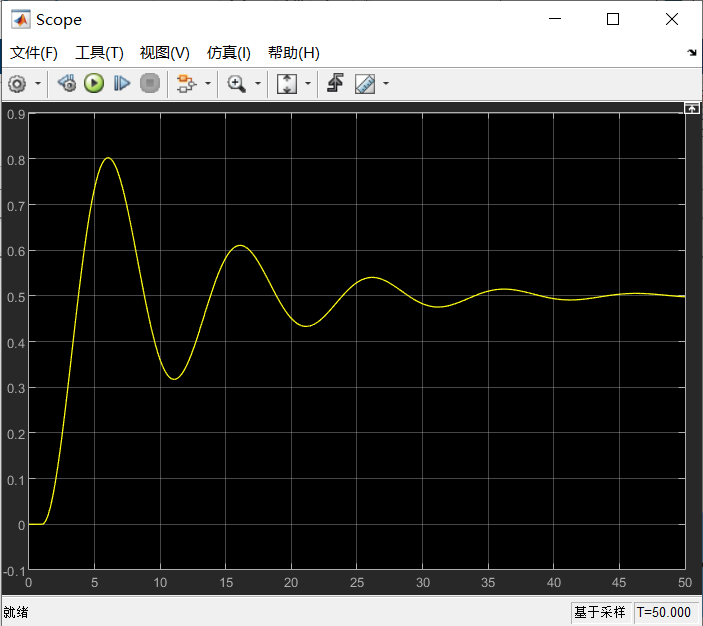
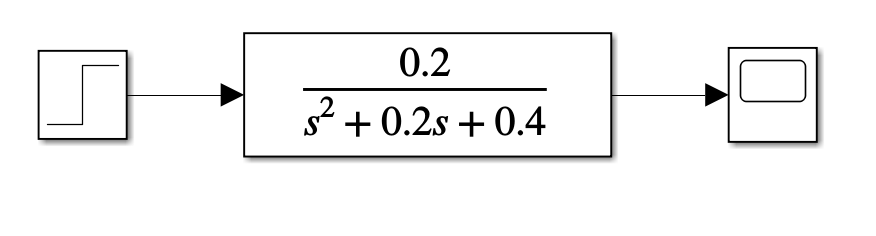
### （3）积分器从零开始对0.5t进行积分



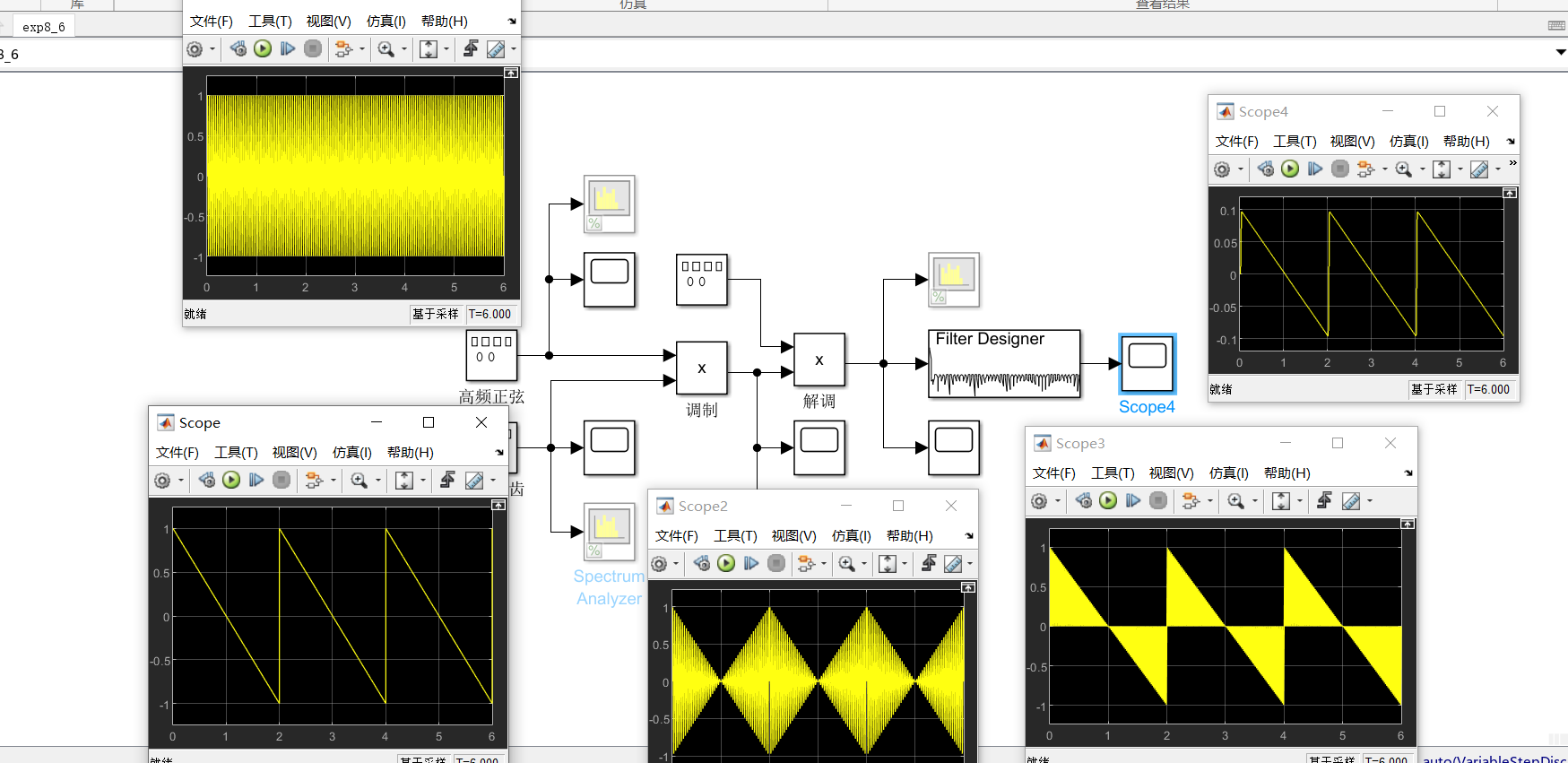
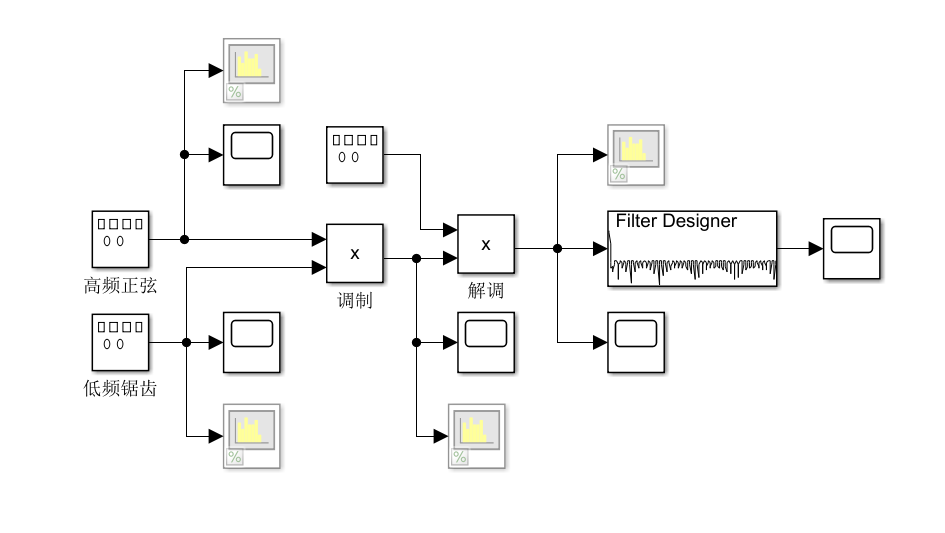
### （4）用积分器直接构搭求解该微分方程的模型



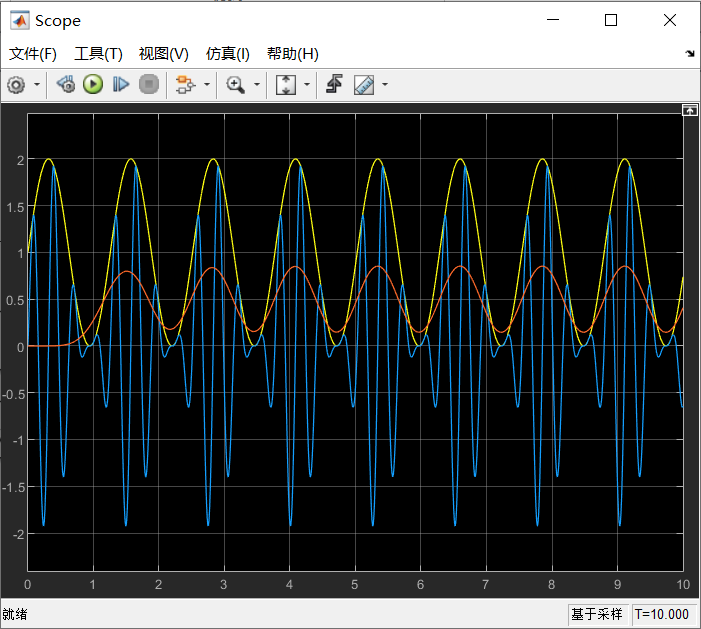
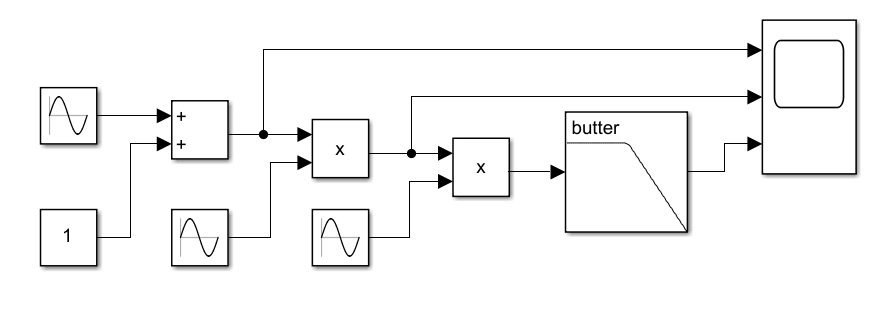
### （5）对二阶微分程进行拉氏变换



### （6）建立数字滤波器系统模型



### （7）利用SIMULINK仿真标准AM调制解调过程



## 三、结果分析

Simulink是MATLAB的一个强大工具，主要用于模拟、仿真和分析动态系统。通过实验，我们学习了Simulink的基本操作，了解了系统仿真和参数设置的基本方法，并熟悉了子系统和封装的概念。

## 四、心得体会

**Simulink提供了强大的图形化仿真能力：**

Simulink的图形化界面使得系统仿真直观且易于操作。通过拖放模块和调整连接，可以快速构建复杂的系统模型。

**求解器和步长的选择是仿真成功的关键：**

仿真的准确性取决于求解器的选择和步长的设置。可变步长适用于连续系统，固定步长适用于离散系统。在选择求解器时，要根据系统的特点进行选择。

**子系统和封装简化了复杂系统的构建：**

子系统可以将一组模块封装在一起，便于管理和复用。封装技术允许创建参考模型和库，方便在不同项目之间共享模块。

**Simulink与MATLAB的无缝集成：**

Simulink与MATLAB紧密集成，仿真结果可以导出到MATLAB工作空间，便于进一步分析和可视化。通过这种集成，可以实现从仿真到分析的顺畅工作流程。