上海大学

《信号分析与处理》
实验指导手册

上海大学中欧学院

上课时间：_____

学　　号：_____

姓　　名：_____

实验成绩：_____

目　录

前　言

“信号分析与处理”是一门兼具理论性与实用性的工科基础入门课程，是电子、通信类专业重要的技术基础课，为通信工程、电子工程、信息工程、电磁场与微波技术等专业开设。主要研究线性非时变系统的特性，信号通过线性系统的基本分析方法，以及某些典型信号通过某些典型系统引出的一些重要概念，为进一步学习专业课程奠定基础。

本课程针对确定性信号经线性时不变系统的传输与处理，分析其相关基本概念与基本分析方法，从时间域到变换域，从连续到离散，通过教学培养本科学生严肃认真的科学作风和严谨缜密的抽象思维能力；以及培养学生分析问题、解决问题及相关计算的能力。

为进一步匹配法国工程师教育体系，在课程中增加相关领域的最新发展动态，增设大量实践实验课程内容势在必行。本指导手册拟通过开展以大作业等前瞻性实践项目为驱动，以创新能力培养为核心的主动学习教学实践改革，从应试教育转向学生的能力锻炼和卓越工程师品质培育。

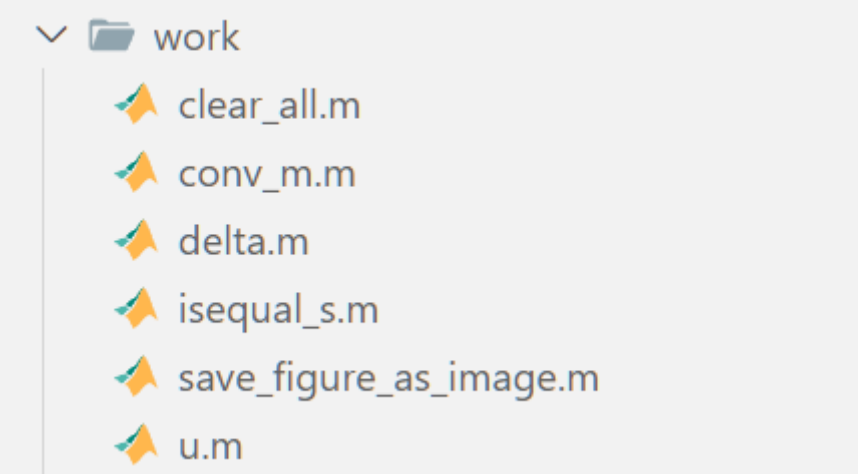
学生补充：本次实验所有代码均可以 Github 上查看所有源代码：

https://github.com/Cans518/MATLAB-Learning

仓库 main 分支对应于本次实验，如需下载使用，在确保您安装 Git 后运行：

```
git clone https://github.com/Cans518/MATLAB-Learning.git
```

对于后续实验中代码本人使用了以下自定义函数，请在验证实验代码前确保这些自定义函数处于 MATLAB 的搜索目录下：



实验一 信号的产生

一、实验目的

1. 熟悉 MATLAB 编程环境，掌握基本的绘图函数；
2. 熟悉和掌握常用的用于信号与系统时域仿真分析的 MATLAB 函数；掌握连续时间和离散时间信号的 MATLAB 产生；
3. 牢固掌握系统的单位冲激响应的概念。

二、实验设备

计算机，MATLAB 软件

三、MATLAB 编程环境

1 绘图函数 plot(x,y)，stem(k,y)

```
% plot(x,y)
x=0:0.01:2;
y=sin(2*pi*x);
plot(x,y)
```

```
% y(t)=sin(2t) + sin(5t)   -2pi ≤ t ≤ 2pi
t =-2*pi:0.02:2*pi;
y=sin(2*t) + sin(5*t);
 plot(t,y)
```

```
% stem(k,y)
k=0:50;
y=exp(-0.1*k);
stem(k,y)
```

四、实验原理

1 信号的时域表示方法

1.1 将信号表示成独立时间变量的函数

例如：x(t)=sin(ωt)和 $x[n]=n(0.5)^n u[n]$

分别表示一个连续时间信号和一个离散时间信号。在 MATLAB 中有许多内部函数，可以直接完成信号的这种表达，例如：

sin()：正弦信号

cos()：余弦信号

exp()：指数信号

sinc()：Sa 函数

1.2 用信号的波形图来描述信号

用函数曲线表示一个信号，图 1.1 就是一个连续时间信号和一个离散时间信号的波形图。

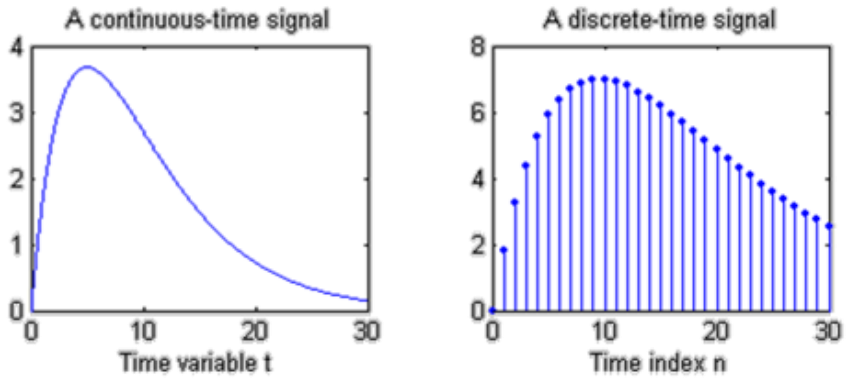


图 1.1 连续时间信号与离散时间信号的波形图

1.3 将信号用一个数据序列来表示

对于离散时间信号，还可以表示成一个数的序列，例如：

x[n]={...., 0.1, 1.1, -1.2, 0, 1.3, ….}

↑n=0

2 用 MATLAB 仿真连续时间信号和离散时间信号

在 MATLAB 中，无论是连续时间信号还是离散时间信号，MATLAB 都是用一个数字序列来表示信号，这个数字序列在 MATLAB 中叫做向量(vector)。通常的情况下，需要与时间

变量相对应。

如前所述，MATLAB 有很多内部数学函数可以用来产生这样的数字序列，例如 sin()、cos()、exp()等函数可以直接产生一个按照正弦、余弦或指数规律变化的数字序列。

2.1 连续时间信号的仿真

程序 Program1_1 是用 MATLAB 对一个正弦信号进行仿真的程序，请仔细阅读该程序，并在计算机上运行，观察所得图形。

```
% Program1_1
% This program is used to generate a sinusoidal signal and draw its plot
clear                   % Clear all variables
close all               % Close all figure windows
dt = 0.01;              % Specify the step of time variable
t = -2:dt:2;            % Specify the interval of time
x = sin(2*pi*t);        % Generate the signal
plot(t,x)               % Open a figure window and draw the plot of x(t)
title('Sinusoidal signal x(t)')
xlabel('Time t (sec)')
```

常用的图形控制函数:

axis([xmin,xmax,ymin,ymax])：图形显示区域控制函数，其中 xmin 为横轴的显示起点，xmax 为横轴的显示终点，ymin 为纵轴的显示起点，ymax 为纵轴的显示终点。

有时，为了使图形具有可读性，需要在所绘制的图形中，加上一些网格线来反映信号的幅度大小。MATLAB 中的 grid on/grid off 可以在图形中加删网格线。

grid on：在图形中加网格线。

grid off：取消图形中的网格线。

x = input('Type in signal x(t) in closed form:')

单位阶跃信号 u(t) 和单位冲激信号 δ(t)

单位阶跃信号 u(t) 和单位冲激信号 δ(t) 是两个非常有用的信号。它们的定义如下

$$\int_{t=-\infty}^{\infty}\delta(t)dt=1 \qquad \delta(t)=0,\quad t\neq 0 \tag{1.1(a)}$$

$$u(t)=\begin{cases}1, & t>0\\ 0, & t\leq 0\end{cases} \tag{1.1(b)}$$

这里分别给出相应的简单的产生单位冲激信号和单位阶跃信号的扩展函数。产生单位冲

激信号的扩展函数为：

```
% Unit impulse function
function y = delta(t)
    dt = 0.01;
    y = (u(t)-u(t-dt))/dt;
```

产生单位阶跃信号的扩展函数为：

```
% Unit step function
function y = u(t)
    y = (t>=0);   % y = 1 for t > 0, else y = 0
```

请将这两个 MATLAB 函数分别以 delta 和 u 为文件名保存在 work 文件夹中，以后，就可以像教材中的方法使用单位冲激信号 δ(t)和单位阶跃信号 u(t)。

2.2 离散时间信号的仿真

程序 Program1_2 用来产生离散时间信号 x[n]=sin(0.2πn)。

```
% Program1_2
% This program is used to generate a discrete-time sinusoidal signal and draw its plot
clear                 % Clear all variables
close all              % Close all figure windows
n = -10:10;             % Specify the interval of time
x = sin(0.2*pi*n);        % Generate the signal
stem(n,x)              % Open a figure window and draw the plot of x[n]
title ('Sinusoidal signal x[n]')
xlabel ('Time index n')
```

请仔细阅读该程序，比较程序 Program1_1 和 Program1_2 中的不同之处，以便自己编程时能够正确使用这种方法仿真连续时间信号和离散时间信号。

程序 Program1_3 用来仿真下面形式的离散时间信号：

x[n]={...., 0.1, 1.1, -1.2, 0, 1.3, ….}

↑n=0

```
% Program1_3
% This program is used to generate a discrete-time sequence
% and draw its plot
clear                 % Clear all variables
close all              % Close all figure windows
n = -5:5;               % Specify the interval of time, the number of points of n is 11.
x = [0, 0, 0, 0, 0.1, 1.1, -1.2, 0, 1.3, 0, 0];   % Generate the signal
stem(n,x,'.')            % Open a figure window and draw the plot of x[n]
grid on
title('A discrete-time sequence x[n]')
xlabel('Time index n')
```

由于在程序的 stem(n,x,'.') 语句中加有'.'选项，因此绘制的图形中每根棒条线的顶端是一个实心点。

如果需要在序列的前后补较多的零，可以利用函数 zeros()，其语法为：

zeros(1, N)：圆括号中的 1 和 N 表示该函数将产生一个一行 N 列的矩阵，矩阵中的所有元素均为零。利用这个矩阵与序列 x[n]进行组合，从而得到一个长度与 n 相等的向量。

例如，当 x[n]={0.1, 1.1, -1.2, 0, 1.3}时，为了得到程序 Program1_3 中的序列，

↑n=0

可以用这个 MATLAB 语句 x = [zeros(1,4) x zeros(1, 2)] 来实现。用这种方法编写的程序如下：

```
% Program1_4
% This program is used to generate a discrete-time sinusoidal signal and draw its plot
clear                   % Clear all variables
close all               % Close all figure windows
n = -5:5;               % Specify the interval of time
x = [zeros(1,4), 0.1, 1.1, -1.2, 0, 1.3, zeros(1,2)];     % Generate the sequence
stem (n,x,'.')             % Open a figure window and draw the plot of x[n]
grid on
title('A discrete-time sequence x[n]')
xlabel('Time index n')
```

离散时间单位样值序列 δ[n]和单位阶跃信号 u[n]定义为

$$\delta[n]=\begin{cases}1, & n=0\\ 0, & n\neq 0\end{cases} \qquad 1.2(a)$$

$$u[n]=\begin{cases}1, & n\geq 0\\ 0, & n<0\end{cases} \qquad 1.2(b)$$

离散时间单位样值序列 δ[n]，产生单位样值序列的扩展函数为：

```
% unit impulse sequence
k=-25:25;
delta=[zeros(1,25),1,zeros(1,25)];
stem(k,delta);
```

离散时间单位阶跃信号 u[n]可以直接用前面给出的扩展函数来产生，还可以利用 MATLAB 内部函数 ones(1,N)来实现。这个函数类似于 zeros(1,N)，所不同的是它产生的矩阵的所有元素都为 1。

值得注意的是，利用 ones(1,N) 来实现的单位阶跃序列并不是真正的单位阶跃序列，而是一个长度为 N 单位门(Gate)序列，也就是 u[n]-u[n-N]。但是在一个有限的图形窗口中，我

们看到的还是一个单位阶跃序列。

在绘制信号的波形图时，有时需要将若干个图形绘制在同一个图形窗口中，这就需要使用 MATLAB 的图形分割函数 subplot()，其用法是在绘图函数 stem 或 plot 之前，使用图形分割函数 subplot(n1,n2,n3)，其中的参数 n1，n2 和 n3 的含义是，该函数将把一个图形窗口分割成 n1 x n2 个子图，即将绘制的图形将绘制在第 n3 个子图中。

五、实验内容及步骤

实验前，首先阅读本实验原理，读懂所给出的全部范例程序。实验开始时，先在计算机上运行这些范例程序，观察所得到的信号的波形图。并结合范例程序应该完成的工作，进一步分析程序中各个语句的作用。

实验前，要针对下面的实验项目做好相应的实验准备工作，包括事先编写好相应的实验程序等事项。

1.修改程序 Program1_1，将 dt 改为 0.2，再执行该程序，保存图形，看看所得图形的效果如何？

```
%清除环境
clear_all;
dt = [0.01,0.2]; %指定时间步长
for i = 1:length(dt)
    t = -2:dt(i):2; %创建时间向量
    x = sin(2*pi*t);  % 生成正弦信号
    fig1 = figure('Units', 'inches', 'Position', [0 0 6 4]);  % 设置绘图窗口尺寸为6x4 英寸
    plot(t, x); % 绘制正弦信号
    xlabel('时间 t (秒)'); % 添加x 轴标签
    ylabel('振幅 x(t)'); % 添加y 轴标签
    save_figure_as_image(fig1, ['dt = ' ,num2str(dt(i))])
end
```

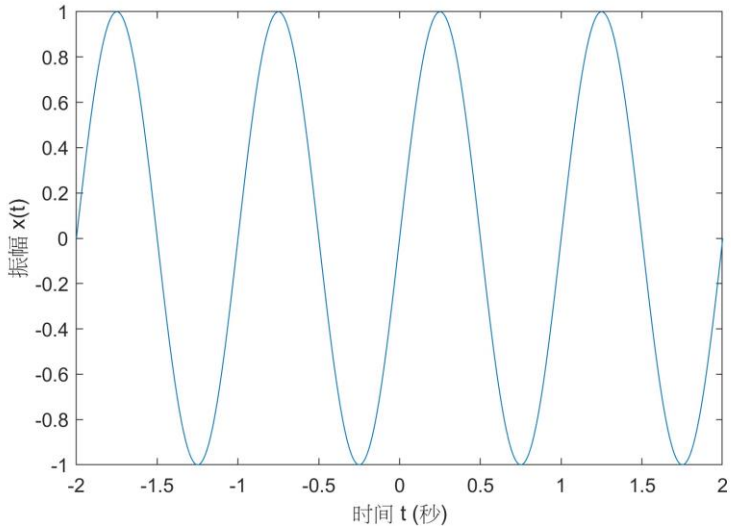


图 1-1. dt = 0.01 时的图像

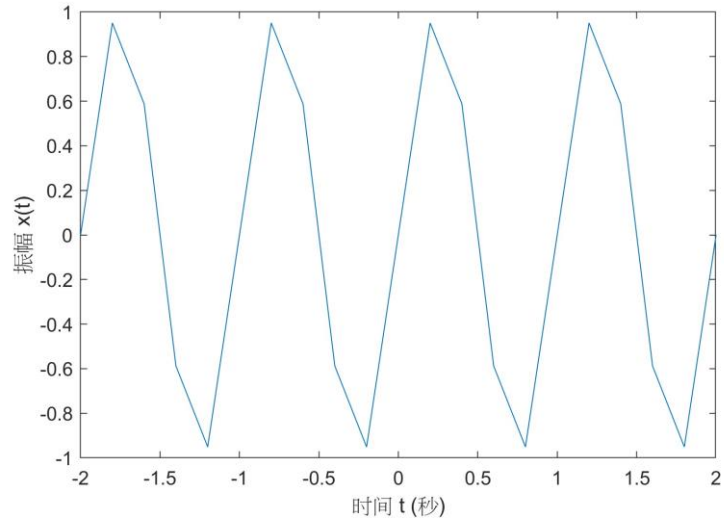


图 1-2. dt = 0.2 时的图像

这两幅图形有什么区别，哪一幅图形看起来与实际信号波形更像？

答：：dt=0.2 的图像的变化会比 dt=0.01 的图像变化更加陡峭，dt=0.01 的图像变化更加的平滑。 当 dt=0.01 时，采样的间隔非常小，因此采样的点非常密集。当 dt=0.2 时，采样的间隔更 大，采样的点相对稀疏。这可能会导致某些信号的细节被忽略或丢失。所以 dt=0.01 时的图 像会更加像实际的信号波图像。

2.修改程序 Program1_1，并存盘，产生实指数信号 $x(t)=e^{-2t}$。 要求在图形中加上网格线，并使用函数 axis()控制图形的时间范围在 0~2 秒之间。然后执行该程序，保存所得的图形。

修改 Program1_1 后得到的程序如下：

```
% 清除环境
clear_all;
t = 0:0.01:2;           % 时间范围为0 到2 秒，步长为0.01 秒
x = exp(-2*t);          % 生成实指数信号
fig1 = figure('Units', 'inches', 'Position', [0 0 6 4]);  % 设置绘图窗口尺寸为6x4 英寸
plot(t, x);             % 绘制图形
grid on;                % 打开网格线
axis([0 2 -0.2 1]);     % 控制坐标轴范围
xlabel('时间 (秒)');
ylabel('幅值');
save_figure_as_image(fig1,'e_minus_2t');
```

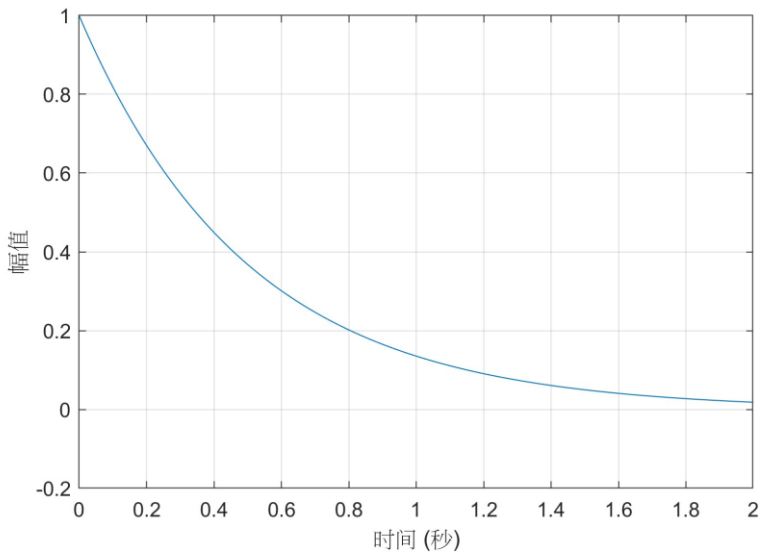


图 1-3. 信号 x(t)=e-2t 的波形图

3.修改程序 Program1_1，并存盘，使之能够仿真从键盘上任意输入的一个连续时间信号，并利用该程序仿真信号 $x(t)=e^{-0.5t}$。

修改 Program1_1 后得到的程序如下：

```
% 清除环境
clear_all;
t_start = input('请输入时间范围的起始值：');
t_end = input('请输入时间范围的结束值：');
t_step = input('请输入时间步长：');
t = t_start:t_step:t_end;  % 输入的时间范围和步长
x = exp(-0.5*t);           % 生成指数衰减信号
fig1 = figure('Units', 'inches', 'Position', [0 0 6 4]);  % 设置绘图窗口尺寸为6x4 英寸
plot(t, x)                 % 绘制图形
grid on                    % 打开网格线
titlename = ['指数衰减信号 x(t) = e^{-0.5t}, t = ', num2str(t_start), ' - ',
num2str(t_end), ', step = ', num2str(t_step)];
title(titlename)
xlabel('时间 (秒)')
ylabel('幅值')
save_figure_as_image(fig1,'e_minus_0_5t.jpg')
```

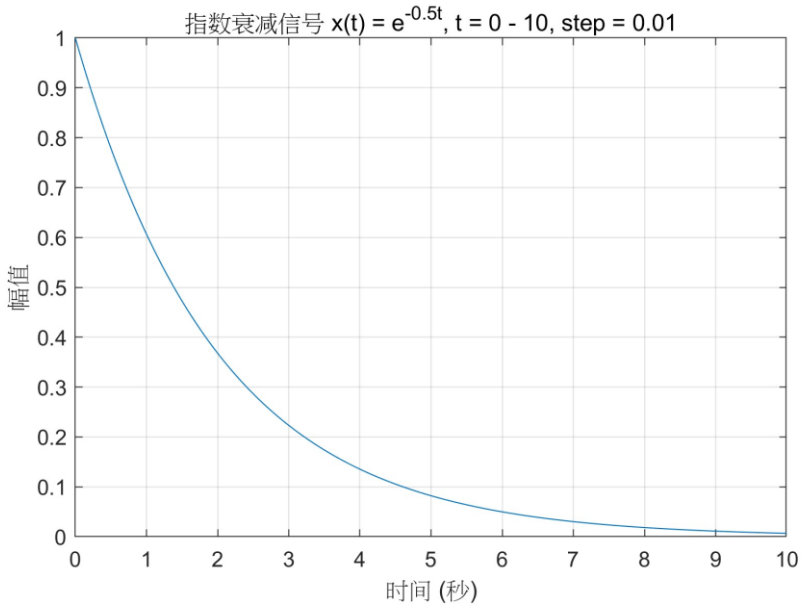


图 1-4. 信号 x(t)=e-0.5t 的波形图

4.将实验原理中所给的单位冲激信号和单位阶跃信号的函数文件在 MATLAB 文件编辑器中编写好，并分别以文件名 delta 和 u 存入 work 文件夹中以便于使用。

抄写函数文件 delta 如下：

```
function y = delta(t)
dt = 1;
y = (u(t)-u(t-dt))/dt;
```

抄写函数文件 u 如下：

```
function y = u(t)
y = (t>=0);   % y = 1 for t > 0, else y = 0
```

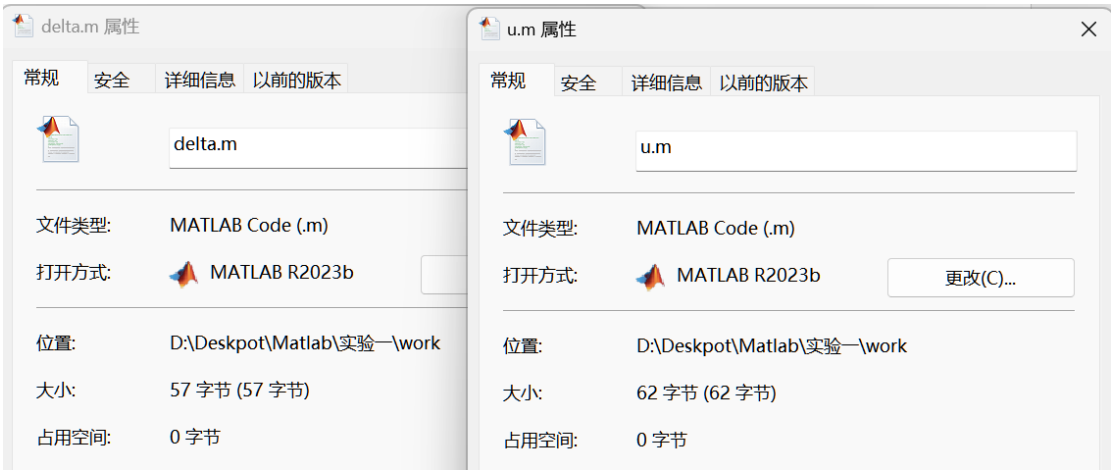


图 1-5. 两个文件路径

5.修改程序 Program1_1，并存盘，产生抽样函数 $y=\sin(\pi t)/(\pi t)$，（利用函数 sinc(t)）。要求在图形中加上网格线，然后执行该程序，保存所的图形。

修改 Program1_1 后得到的程序如下：

```
%清理环境
clear_all;
t = -10:0.01:10;        % 时间范围为-10 到10 秒，步长为0.01 秒
y = sinc(t);            % 生成抽样函数
fig = figure('Units', 'inches', 'Position', [0 0 6 4]);  % 设置绘图窗口尺寸为6x4 英寸
plot(t, y)              % 绘制图形
grid on                 % 打开网格线
xlabel('时间 (秒)')
ylabel('幅值');
save_figure_as_image(fig,'sinc_t.jpg');
```

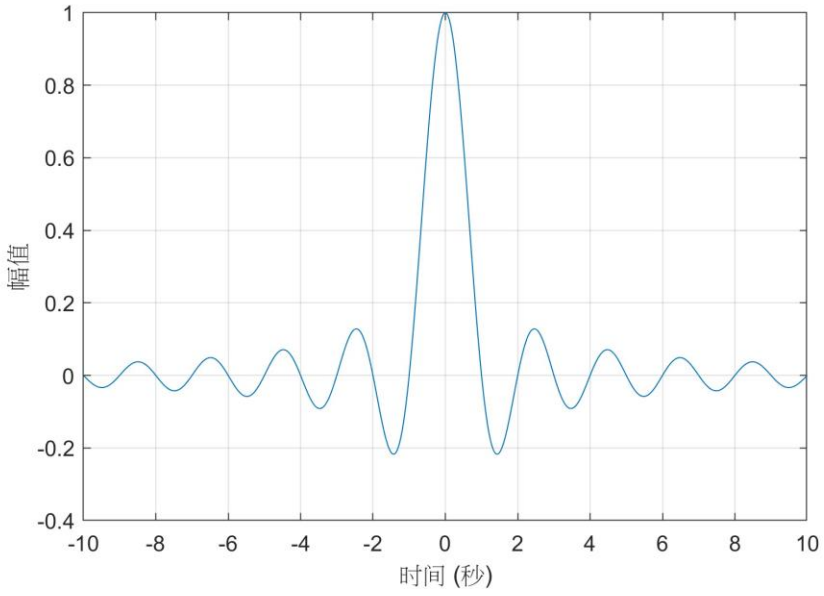


图 1-6. 信号 y(t)的波形图

6.修改程序 Program1_4，并存盘，利用 axis()函数，将图形窗口的横坐标范围改为-2≤n≤5，纵坐标范围改为-1.5≤ x ≤1.5。

修改 Program1_4 后得到的程序如下：

```
% 清除环境
clear_all;
n = -5:5;                  % 时间范围为-5 到 5
x = [zeros(1,4), 0.1, 1.1, -1.2, 0, 1.3, zeros(1,2)];     % 生成序列
fig1 = figure('Units', 'inches', 'Position', [0 0 6 4]);  % 设置绘图窗口尺寸为 6x4 英寸
stem(n, x, '.')              % 绘制图形
grid on                  % 打开网格线
title('离散时间序列 x[n]')
xlabel('时间索引 n')
ylabel('幅值')
axis([-2 5 -1.5 1.5])  % 设置坐标范围
save_figure_as_image(fig1,'Program1_4');
```

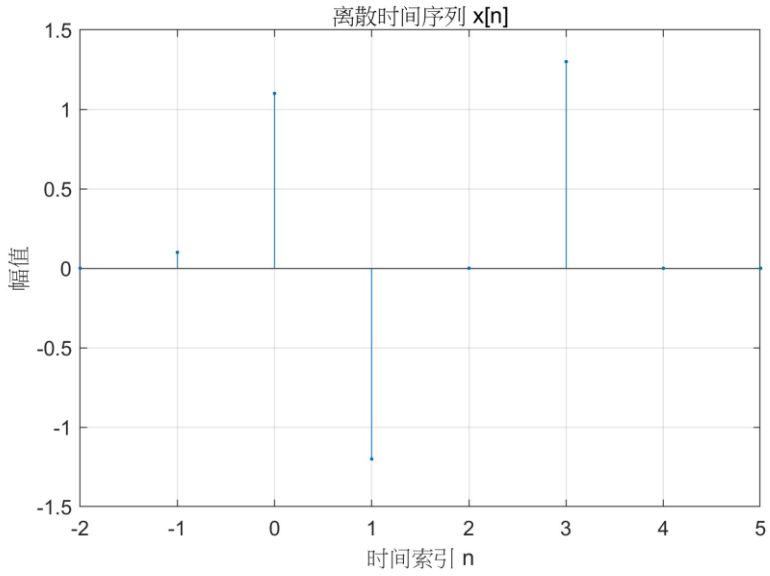


图 1-7. 时间序列图

本实验完成时间： 2024 年 3 月 19 日

实验二 信号的基本运算和波形变换

一、实验目的

1. 掌握基本的变量和矩阵的运算；
2. 熟悉和掌握常用的信号时域变换；
3. 掌握用周期延拓的方法将一个非周期信号进行周期信号延拓形成一个周期信号的 MATLAB 编程。

二、实验设备

计算机，MATLAB 软件

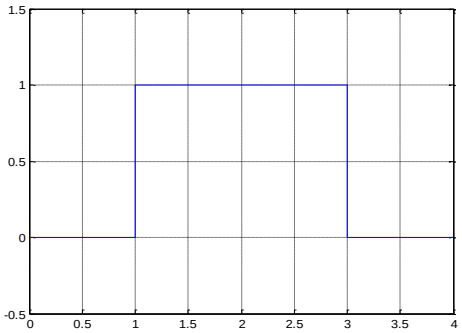
三、实验原理

1 信号的基本运算

1.1＋、－、×运算

两信号 $f1(\cdot)$和 $f2(\cdot)$的相+、－、×指同一时刻两信号之值对应相加、减、乘。

下面矩形信号的 MATLAB 程序表示，就采用了之前的扩展函数，设幅度 A=1，宽度为 W=2。




```
% Program2_1
% rectangular pulse signal
t=0:0.001:4;
ft=u(t-1)- u(t-3);
plot(t,ft);
grid on;
axis([0 4 -0.5 1.5]);
```

也可以用矩形函数表述：

```
% rectangular pulse signal
t=0:0.001:4;
T=1;
ft=rectpuls(t-2*T,2*T);
plot(t,ft);
grid on;
axis([0 4 -0.5 1.5]);
```

2 信号的时域变换

2.1 信号的时移

信号的时移可用下面的数学表达式来描述：设一个连续时间信号为 x(t)，时移 y(t)表示为：

$$y(t) = x(t - t_0) \qquad (2.1)$$

其中，t_0 为位移量。若 t_0 为正数，则 y(t)等于将 x(t)右移 t_0 秒之后的结果。反之，若 t_0 为负数，则 y(t)等于将 x(t)左移 t_0 秒之后的结果。在 MATLAB 中，时移运算与数学上习惯表达方法完全相同。程序 Program2_2 对给定一个连续时间信号 $x(t) = e^{-0.5t}u(t)$，对它分别左移 2 秒钟和右移 2 秒钟得到信号 $x_1(t) = e^{-0.5(t+2)}u(t+2)$和 $x_2(t) = e^{-0.5(t-2)}u(t-2)$。

```
% Program2_2
% This program is used to implement the time-shift operation
% on a continuous-time signal and to obtain and to draw their plots.
clear,close all,
t = -5:0.01:5;
x = exp(-0.5*t).*u(t);          % Generate the original signal x(t)
x1 = exp(-0.5*(t+2)).*u(t+2);   % Shift x(t) to the left by 2 second to get x1(t)
x2 = exp(-0.5*(t-2)).*u(t-2);    % Shift x(t) to the right by 2 second to get x2(t)
subplot(3,1,1)
plot(t,x)                   % Plot x(t)
grid on
title ('Original signal x(t)')
subplot (3,1,2)
plot (t,x1)                    % Plot x1(t)
grid on
title ('Left shifted version of x(t)')
subplot (3,1,3)
plot (t,x2)                    % Plot x2(t)
grid on
title ('Right shifted version of x(t)')
xlabel ('Time t (sec)')
```

2.2 信号的时域反褶

对一个信号 x[n]的反褶运算在数学上表示为

$$y[n] = x[-n] \qquad (2.2)$$

有多种方法可以实现信号的反褶运算：

方法一：修改绘图函数 plot(t,x)和 stem(n,x)中的时间变量 t 和 n，即用-t 和-n 替代原来的 t 和 n，这样绘制出来的图形，看起来就是原信号经时域反褶后的版本。

方法二：直接利用原信号与其反褶信号的数学关系式来实现。这种方法最符合信号反褶运算的实际意义。

方法三：使用 MATLAB 内部函数 fliplr()来实现信号的反褶运算。其用法如下：

y = fliplr(x)：其中 x 为原信号 x(t)或 x[n]，而 y 则为 x 的时域反褶。需要说明的是，函数 fliplr()对信号作时域反褶，仅仅将信号中各个元素的次序作了一个反褶，这种反褶处理是独立于时间变量 t 和 n 的。因此，如果信号与其时间变量能够用一个数学函数来表达的话，那么建议将时间变量 t 和 n 的范围指定在一个*正负对称*的时间区间即可。

2.3 信号的时域尺度变换

信号 x(t)的时域尺度变换在数学描述为：

$$y(t) = x(at), \qquad (2.3)$$

其中 a 为任意常数。根据 a 的不同取值，这种时域尺度变换对信号 x(t)具有非常不同的影响。

当 a = 1 时，y(t) = x(t)；

当 a = -1 时，y(t) = x(-t)，即 y(t)可以通过将 x(t)反褶运算而得到；

当 a > 1 时，y(t) = x(at)，y(t)是将 x(t)在时间轴上的压缩而得到；

当 0 < a < 1 时，y(t) = x(at)，y(t)是将 x(t)在时间轴上的扩展而得到；

当 -1 < a < 0 时，y(t) = x(at)，y(t)是将 x(t)在时间轴上的扩展同时翻转而得到；

当 a < -1 时，y(t) = x(at)，y(t)是将 x(t)在时间轴上的压缩同时翻转而得到；

由此可见，信号的时域尺度变换，除了对信号进行时域压缩或扩展外，还可能包括对信号的时域反褶运算。实际上，MATLAB 完成该运算，并不需要特殊的处理，按照数学上的常规方法即能完成。

已知 $f(t)$ 为三角函数，利用 MATLAB 画出 $f(2t)$ 和 $f(2-2t)$ 的波形。程序如下：

```
% Program2_3
%changed triangular pulse signal
t=-3:0.001:3;
ft1=tripuls(2*t,4,0.5);
subplot(2,1,1);
plot(t,ft1);
title('f(2t)');
grid on;
ft2=tripuls((2-2*t),4,0.5);
subplot(2,1,2);
plot(t,ft2);
title('f(2-2t)');
grid on
```

四、实验内容及步骤

实验前，必须首先阅读本实验原理，读懂所给出的全部范例程序。实验开始时，先在计算机上运行这些范例程序，观察所得到的信号的波形图。并结合范例程序应该完成的工作，进一步分析程序中各个语句的作用，从而真正理解这些程序。

1.仿照前面示例程序的编写方法，编写一个 MATLAB 程序，并存盘，使之能够在同一个图形窗口中的两个子图中分别绘制信号 $x[n]=0.5^{|n|}$ 和 $x(t)=\cos(2\pi t)[u(t)-u(t-3)]$。要求选择的时间窗能够表现出信号的主要部分（或特征）。

编写的程序如下：

```
% 清除环境
clear_all;
% 信号 x[n] = 0.5^|n|
n = -10:10; % 选择时间窗为 -10 <= n <= 10
x1 = 0.5 .^ abs(n); % 计算信号值
fig = figure;
subplot(2,1,1); % 在图形窗口的上半部分绘制子图
stem(n, x1, 'b', 'filled'); % 绘制离散信号
title('信号 x[n] = 0.5^{|n|}');
xlabel('n');
ylabel('x[n]');
grid on;
% 信号 x(t) = cos(2*pi*t)[u(t) - u(t-3)]
t = -0.5:0.01:3.5; % 选择时间窗为 -0.5 <= t <= 3.5
x2 = cos(2*pi*t) .* (u(t) - u(t-3)); % 计算信号值
subplot(2,1,2); % 在图形窗口的下半部分绘制子图
plot(t, x2, 'r'); % 绘制连续信号
```

```
title('信号 x(t) = cos(2πt)[u(t) - u(t-3)]');
xlabel('t');
ylabel('x(t)');
grid on;
save_figure_as_image(fig,'Program2_1');
```

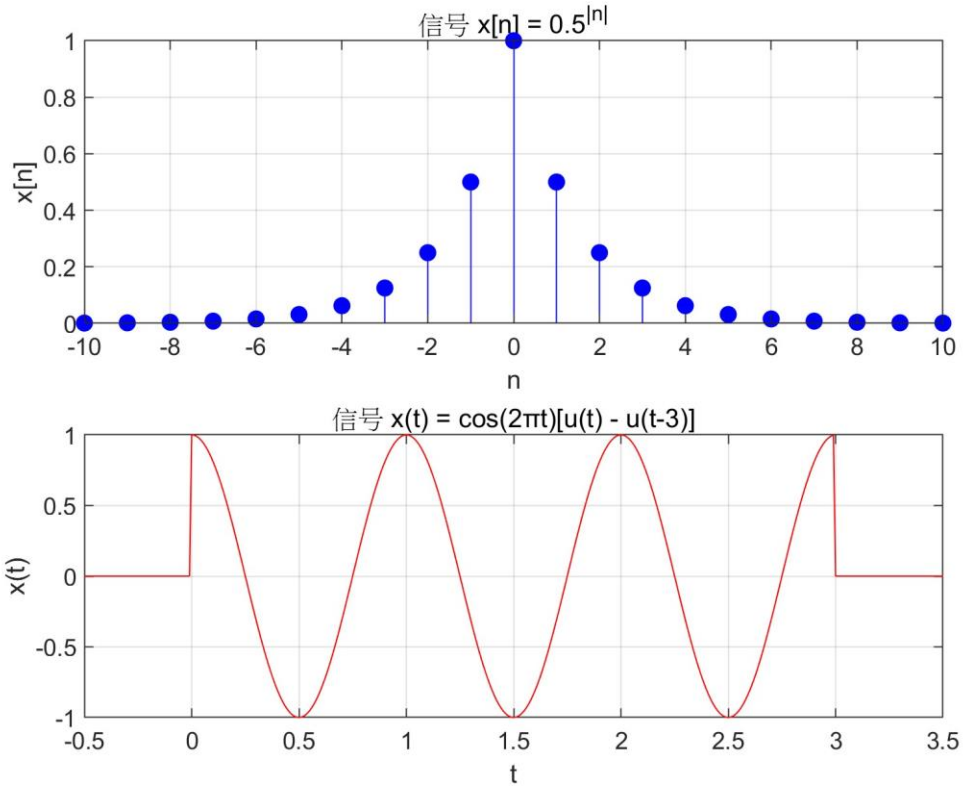


图 2-1. 信号 x[n]=0.5$^{|n|}$ 的波形图和信号 x(t)=cos(2πt)[u(t)-u(t-3)]的波形图

2.根据示例程序的编程方法，编写一个 MATLAB 程序，并存盘，由给定信号

$$x(t) = e^{-0.5t}u(t)$$

求信号 y(t) = x(1.5t+3)，并绘制出 x(t) 和 y(t)的图形。

编写的程序如下：

```
clear_all; % 清除环境
t = -4:0.01:10; % 定义时间范围
x = exp(-0.5*t) .* u(t); % 信号 x(t) = e^(-0.5t)u(t)
t_y = 1.5*t + 3; % 信号 y(t) = x(1.5t+3)
y = exp(-0.5*t_y) .* u(t_y);
fig = figure;
subplot(2,1,1); % 在图形窗口的上半部分绘制子图
plot(t, x, 'b');
title('x(t) = e^{-0.5t}u(t)');
xlabel('t');
ylabel('x(t)');
grid on;
subplot(2,1,2); % 在图形窗口的下半部分绘制子图
plot(t, y, 'r');
title('y(t) = x(1.5t+3)');
xlabel('t');
ylabel('y(t)');
grid on;
save_figure_as_image(fig,'Program2_2');
```

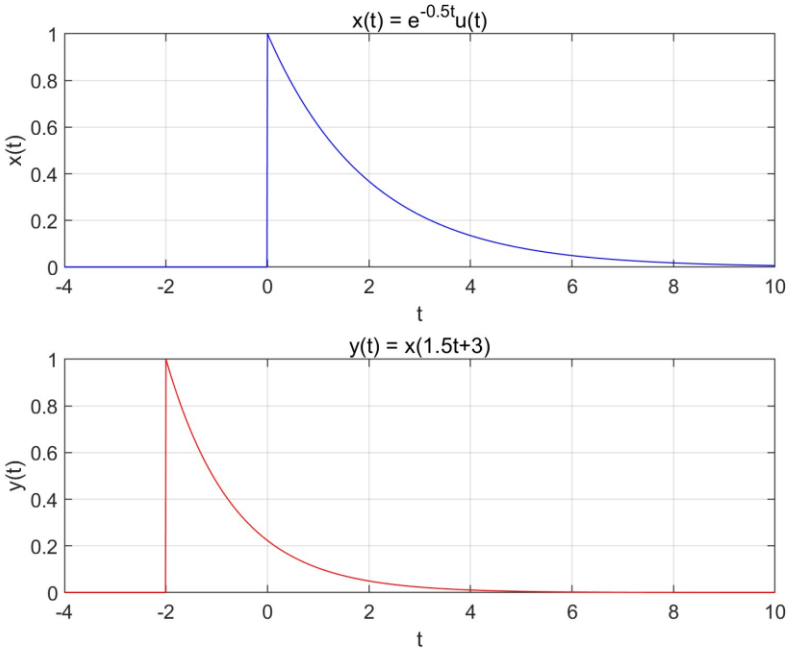


图 2-2. 信号 x(t)的波形图 和 信号 y(t) = x(1.5t+3) 的波形图

3.给定一个离散时间信号 x[n]＝u[n]– u[n-8]，仿照示例程序，编写程序，产生 x[n]的左移序列 $x_1[n]$＝x[n+6]和右移序列 $x_2[n]$＝x[n-6]，并在同一个图形窗口的三个子图中分别绘制这三个序列的图形。

编写的程序如下：

```
clear_all; % 清除环境
n = -10:20; % 定义时间范围
x = (n >= 0) - (n >= 8); % 信号 x[n] = u[n] - u[n-8]
x1 = (n+6 >= 0) - (n+6 >= 8); % 左移序列 x1[n] = x[n+6]
x2 = (n-6 >= 0) - (n-6 >= 8); % 右移序列 x2[n] = x[n-6]
fig = figure;
subplot(3,1,1); % 在图形窗口的上部分绘制子图
stem(n, x, 'b', 'filled');
title('x[n] = u[n] - u[n-8]');
xlabel('n');
ylabel('x[n]');
grid on;
subplot(3,1,2); % 在图形窗口的中部分绘制子图
stem(n, x1, 'r', 'filled');
title('x_{1}[n] = x[n+6]');
xlabel('n');
ylabel('x1[n]');
grid on
subplot(3,1,3); % 在图形窗口的下部分绘制子图
stem(n, x2, 'g', 'filled');
title('x_{2}[n] = x[n-6]');
xlabel('n');
ylabel('x2[n]');
grid on;
save_figure_as_image(fig,'Program2_3');
```

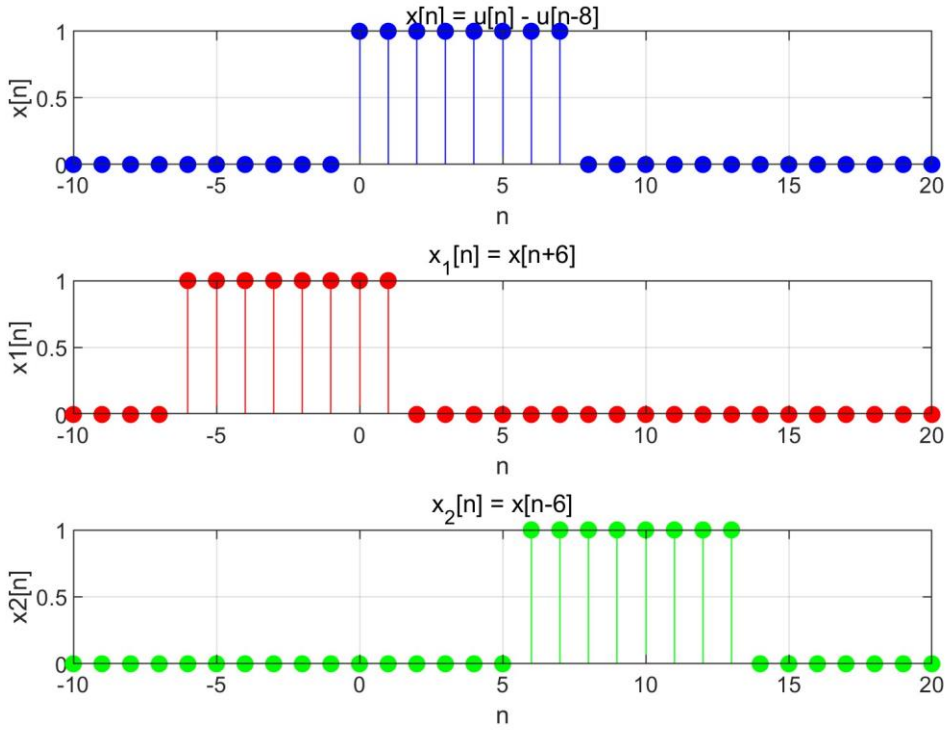


图 2-3. 信号波形图

本实验完成时间： 2024 年 3 月 20 日

实验三 连续时间系统时域分析的 MATLAB 实现

一、实验目的

掌握应用 MATLAB 实现对线性时不变连续时间系统的时域分析，即

1. 掌握连续系统及离散系统的零状态响应的概念；
2. 会由微分方程求连续系统的冲激响应、阶跃响应；
3. 会由差分方程求离散系统的序列响应、阶跃响应；
4. 熟悉并验证卷积的性质；
5. 利用卷积生成新的波形，建立波形间的联系。

二、主要设备

计算机，MATLAB 软件

三、实验原理

LTI 连续系统以常系数微分方程描述，系统的零状态响应可以通过求解初始状态为零的微分方程得到。在 MATLAB 中，控制系统工具箱提供了用于求解零初始状态微分方程数值解的函数 lsim，求解冲激响应的函数 impulse，求解阶跃响应的函数 step，调用形式

$$y = \mathrm{lsim}(sys, f, t)$$

$$y = impulse(sys, t)$$

$$y = step(sys, t)$$

式中，t 为响应的采样点向量，f 是输入信号向量，sys 是系统模型，用来表示微分方程、差分方程、状态方程。在求解微分方程时，sys 借助 tf 函数来获得，调用形式为：

$$sys = tf(b, a)$$

式中，b 和 a 分别为微分方程右端和左端的系数向量。

LTI 离散系统以常系数差分方程描述，差分方程实质是一个迭代方程，系统的响应可以通过编程进行迭代运算得到。在 MATLAB 中，控制系统工具箱提供了用于求解零初始状态差分方程解的函数 filter，求解单位序列响应的函数 impz，求解阶跃响应的函数 dstep，调用形式：

$$y = filter(b, a, f)$$

$$y = impz(b, a, k)$$

$$y = dstep(b, a, k)$$

式中，f 是输入序列，b 和 a 分别为差分方程右端和左端的系数向量。

信号的卷积一般用于求取信号通过某系统后的响应。在信号与系统中，我们通常求取某系统的单位序列响应，所求得的 h(k)可作为系统的时域表征。任意系统的系统响应可用卷积的方法求得：

$$y(k) = f(k) * h(k)$$

四、实验内容及步骤

1.已知系统的微分方程为

$$y''(t) + 2y'(t) + 100y(t) = f(t)$$

画出（1）冲激响应 h(t)的波形；

（2）阶跃响应 g(t)的波形；

（3）若 $f(t) = 10\sin(2\pi t)$，画出系统的零状态响应波形。

编写的程序如下：

```
% 清除环境
clear_all;
% 定义微分方程的系数
a = [1, 2, 100];
b = [0, 0, 1];
% 创建系统模型
sys = tf(b, a);
% 定义时间向量
t_start = 0;
t_end = 5;
t_step = 0.01;
t = t_start:t_step:t_end;
% 计算并画出冲激响应
fig = figure('Units', 'inches', 'Position', [0 0 6 4]);  % 设置绘图窗口尺寸为6x4 英寸
subplot(3, 1, 1);
impulse(sys, t);
grid on                % 打开网格线
title('Impulse response');
ylabel('Amplitude');
subplot(3, 1, 2);
step(sys, t);
grid on                % 打开网格线
title('Step response');
ylabel('Amplitude');
```

```
% 定义输入信号
f = @(t) 10*sin(2*pi*t);
% 定义微分方程
dydt = @(t, y) [y(2); f(t) - 2*y(2) - 100*y(1)];
% 定义初始条件
y0 = [0; 0];
% 使用 ode45 求解微分方程
[t, y] = ode45(dydt, t, y0);
subplot(3, 1, 3);
plot(t, y(:, 1));
grid on                 % 打开网格线
title('Zero-state response');
xlabel('Time (s)');
ylabel('Amplitude');
save_figure_as_image(fig, 'Program3_1');
```

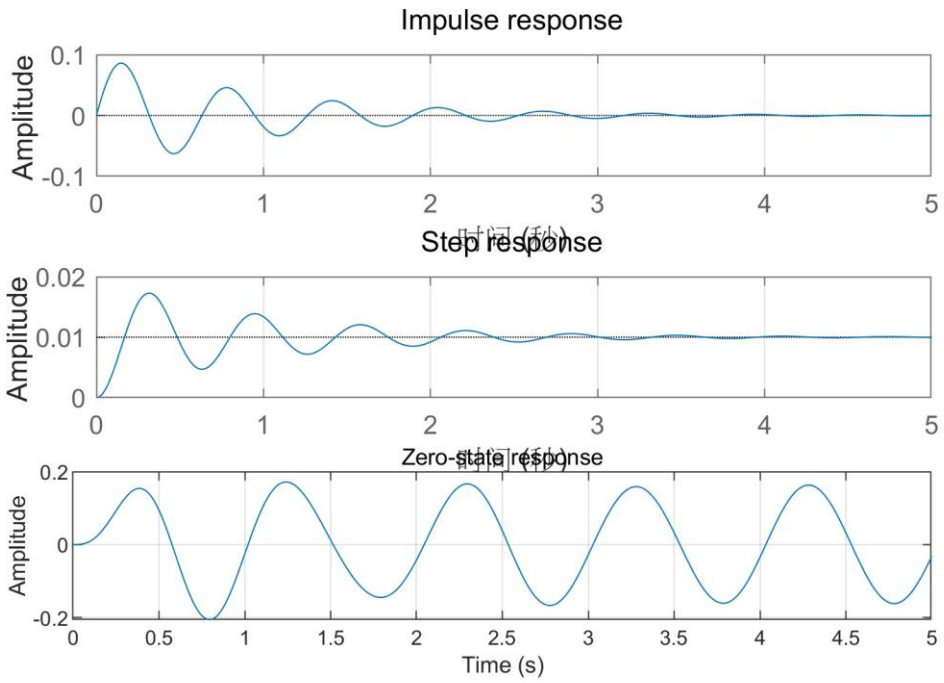


图 3-1. 信号分析图

2.利用 MATLAB 求离散系统：

$$y(k)-y(k-1)+0.9y(k-2)-0.5y(k-3)=f(k)$$

的单位序列响应，阶跃响应。

编写的程序如下：

```
clear_all;
% 定义系统函数的系数
a = [1 -1 0.9 -0.5];
b = 1;
% 定义时间向量
t_start = 0;
t_end = 30;
t_step = 1;
t = t_start:t_step:t_end;
% 生成单位脉冲信号和单位阶跃信号
impulse_signal = [1, zeros(1, t_end)];
step_signal = ones(1, t_end + 1);
% 计算单位脉冲响应和单位阶跃响应
impulse_response = filter(b, a, impulse_signal);
step_response = filter(b, a, step_signal);
% 画出单位脉冲响应
fig = figure('Units', 'inches', 'Position', [0 0 6 4]);  % 设置绘图窗口尺寸为6x4 英寸
subplot(2, 1, 1);
stem(t, impulse_response, 'filled');
grid on                % 打开网格线
title('Unit impulse response');
xlabel('Discrete time index k');
ylabel('Amplitude');
subplot(2, 1, 2);
stem(t, step_response, 'filled');
grid on                % 打开网格线
title('Unit step response');
xlabel('Discrete time index k');
ylabel('Amplitude');
save_figure_as_image(fig,'Program3_2');
```

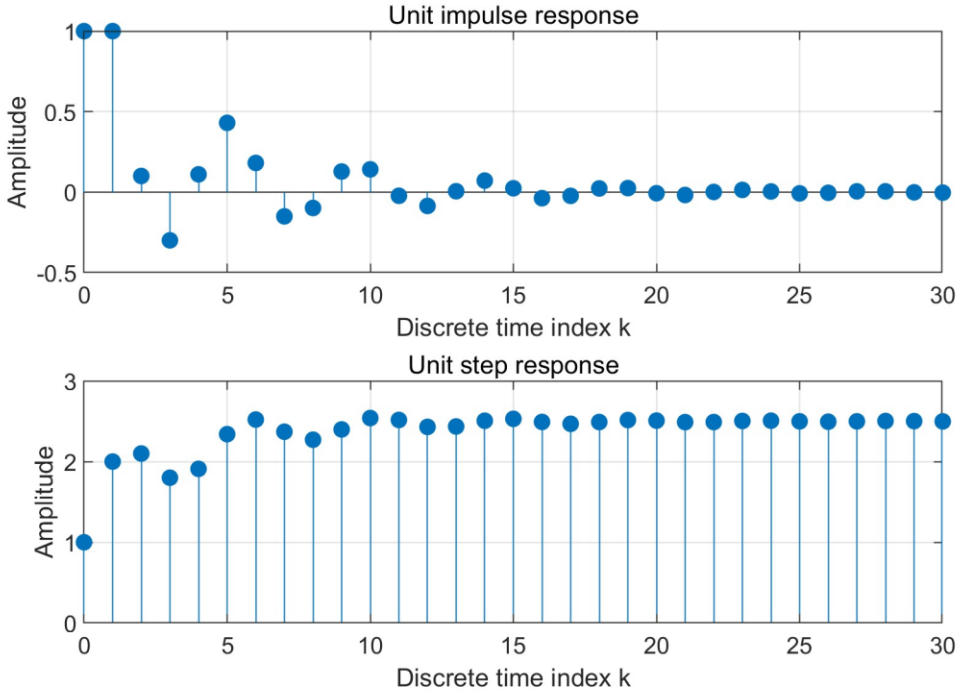

Unit impulse response
Amplitude
1
0.5
0
-0.5
0
5
10
15
20
25
30
Discrete time index k
Unit step response
Amplitude
3
2
1
0
0
5
10
15
20
25
30
Discrete time index k

图 3-2. 单位冲击信号

3.MATLAB 提供了一个内部函数 conv 来计算两个有限长序列的卷积。conv 函数假定两个序列都从 n=0 开始。给出序列

x=[3,11,7,　0,　-1,4,2]；h=[2,3,　0,　-5,2,1]；

↑n=0　　↑n=0

求两者的卷积 y。

将函数 conv 稍加扩展为函数 conv_m，它可以对任意基底的序列求卷积。格式如下：

function　[y,ny]=conv_m(x,nx,h,nh)

%　信号处理的改进卷积程序

%　[y,ny]=conv_m(x,nx,h,nh)

%　[y,ny]=卷积结果

%　[x,nx]=第一个信号，x 为序列值，nx 为序列值对应的时刻，二者都为数组

%　[h,nh]=第二个信号，h 为序列值，nh 为序列值对应的时刻，二者都为数组

编写的程序如下：

```
% 清除环境
clear_all;
% 定义序列
x = [3, 11, 7, 0, -1, 4, 2];
h = [2, 3, 0, -5, 2, 1];
% 计算卷积
y = conv(x, h);
% 输出结果
disp(y);
```

```
     6    31    47     6   -51    -5    41    18   -22    -3     8     2
```

图 3-3.　输出效果图

4.对下面三个序列，用 conv_m 函数，验证卷积特性（交换律、结合律、分配律、同一律）

$$f_1(n) * f_2(n) = f_2(n) * f_1(n)$$ 交换律

$$[f_1(n) * f_2(n)] * f_3(n) = f_1(n) * [f_2(n) * f_3(n)]$$ 结合律

$$[f_1(n) + f_2(n)] * f_3(n) = f_1(n) * f_3(n) + f_2(n) * f_3(n)$$ 分配律

$$f(n) * \delta(n - n_0) = f(n - n_0)$$ 同一律

其中：$f_1(n) = n\,[un+10) - u(n-20)]$

$f_2(n) = \cos(0.1\pi n)\,[u(n) - u(n-30)]$

$f_3(n) = (1.2)^n[u(n+5) - u(n-10)]$

编写的程序如下：

```
% 清除环境
clear_all;
% 定义时间向量
n = -30:30;
% 定义信号f1, f2, f3
f1 = n .* (u(n + 10) - u(n - 20));
f2 = cos(0.1 * pi * n) .* (u(n) - u(n - 30));
f3 = (1.2 .^ n) .* (u(n + 5) - u(n - 10));
d = delta(n);
tol = 1e-10;
% 验证交换律
[c1,nc1]=conv_m(f1,n,f2,n);
[c2,nc2]=conv_m(f2,n,f1,n);
if(isequal_s(c1,c2,tol))
    disp('交换律验证成功');
else
    disp('交换律验证失败');
end
% 验证结合律
[j1,nj1]=conv_m(c1,nc1,f3,n);
[t1,nt1]=conv_m(f3,n,f2,n);
[j2,nj2]=conv_m(t1,nt1,f1,n);
if(isequal_s(j1,j2,tol))
    disp('结合律验证成功');
else
    disp('结合律验证失败');
end
% 验证分配律
[d1,nd1]=conv_m(f1+f2,n,f3,n);
[d2,nd2]=conv_m(f3,n,f1,n);
d3 = d2 + t1;
if(isequal_s(d1,d3,tol))
```

```
    disp('分配律验证成功');
else
    disp('分配律验证失败');
end
% 验证同一律
[e1,ne1] = conv_m(d,n,f1,n);
e1 = e1(31:91);% 取e1的-30~30的部分
if (isequal_s(f1,e1,tol))
    disp('同一律验证成功');
else
    disp('同一律验证失败');
end
```

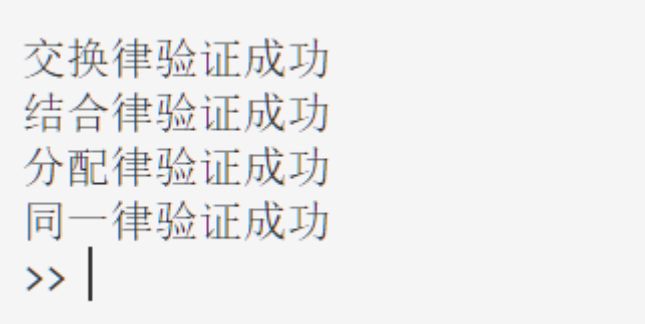


图 3-4. 运行效果图

5.已知信号 $x(t)=u(t)-u(t-2)$，$y(t)=u(t-3)-u(t-5)$，利用 MATLAB 计算 $x(t)*y(t)$，并画出卷积结果。

编写的程序如下：

```
% 定义时间范围
t = -10:0.01:10;
% 定义信号x(t)和y(t)
x = (t >= 0) - (t >= 2);
y = (t >= 3) - (t >= 5);
% 计算卷积
z = conv(x, y, 'same') * 0.01; % 'same'选项使得卷积结果的长度与x和y相同。乘以0.01是因为我们使用了0.01的时间步长。
% 画出卷积结果
figure;
plot(t, z);
title('Convolution of x(t) and y(t)');
xlabel('t');
ylabel('x(t) * y(t)');
save_figure_as_image(figure,'Program3_5');
```

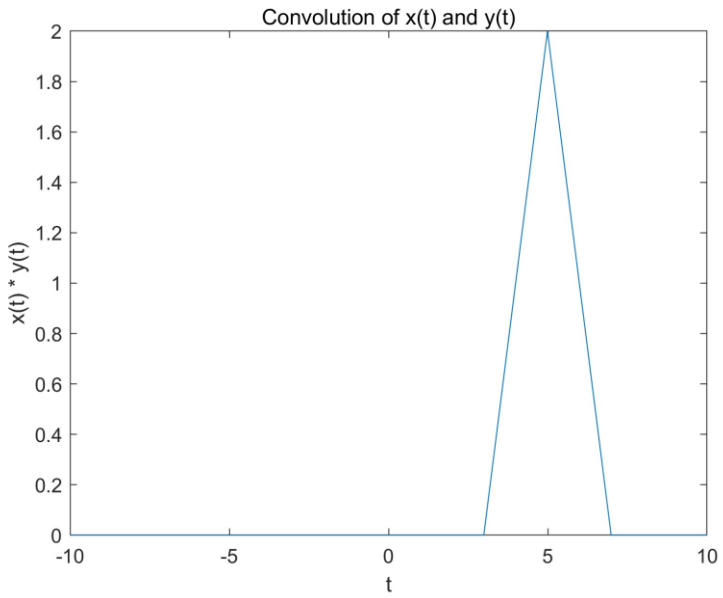


图 3-5. 卷积结果图

本实验完成时间： 2024 年 4 月 11 日

实验四 连续时间系统频域分析的 MATLAB 实现

一、实验目的

掌握应用 MATLAB 实现对线性时不变连续时间系统的频域分析，掌握应用 MATLAB 近似计算和绘制信号的频谱、连续时间系统的频率响应（幅频响应和相频响应），掌握连续信号的抽样定理。

三、主要设备

计算机，MATLAB 软件

四、实验原理

满足狄里赫利的周期信号可以展开为傅里叶级数，既可以用一系列不同频率正弦或复指数信号之和，傅里叶级数有三角形式和指数形式两种。

三角形式：

$$f(t)=\frac{a_0}{2}+\sum_{n=1}^{\infty}\left[a_n\cos\left(n\Omega t\right)+b_n\sin\left(n\Omega t\right)\right]$$

或 $f(t)=\frac{A_0}{2}+\sum_{n=1}^{\infty}A_n\cos\left(n\Omega t+\varphi_n\right)$

指数形式：$f(t)=\sum_{n=-\infty}^{\infty}F(n\Omega)\,\mathrm{e}^{\mathrm{j}n\Omega t}$

其中

$$a_n=\frac{2}{T}\int_{t_0}^{t_0+T}f(t)\cos\left(n\Omega t\right)\mathrm{d}t\,,$$

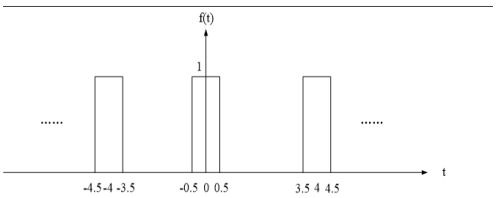
$$b_n=\frac{2}{T}\int_{t_0}^{t_0+T}f(t)\sin\left(n\Omega t\right)\mathrm{d}t\,,$$

$$F(n\Omega)=\frac{1}{2}A_n e^{j\varphi_n}=\frac{1}{2}(a_n-jb_n)=\frac{1}{T}\int_{-T/2}^{T/2}f(t)e^{-jn\Omega t}dt \qquad n=0,\pm1,\pm2,\mathrm{L}$$

一般来讲，傅里叶级数系数有无限个非零值，但对数值计算来讲，这是无法实现的。在实际应用中可以用有限项的傅里叶级数求和来逼近。当项数取得较多时，可以很好的近似原

信号，但会出现吉布斯现象。

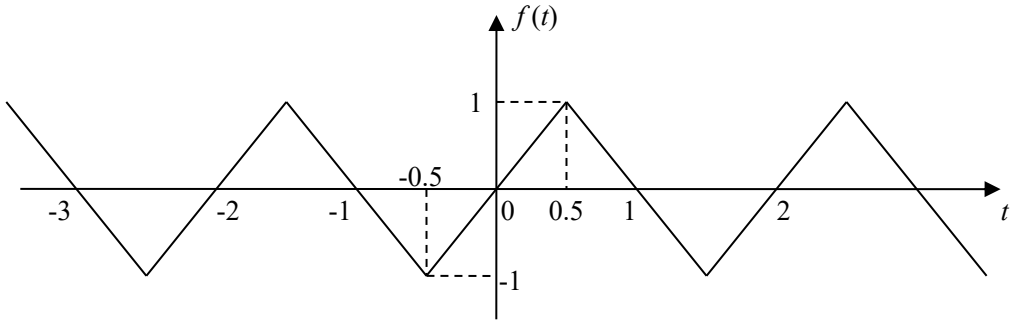
如图所示的周期矩形脉冲信号，求幅度谱。




```
% Program4_1
%幅度谱示例
clear, close all,
syms t n
T=4;A=1;tao=1;
f=A*exp(-1i*n*2*pi/T*t);
fn=int(f,t,-tao/2,tao/2)/T;   %计算傅里叶系数
fn=simplify(fn);          %简化
n1=[-20:-1,eps,1:20];   %给定频谱的整数自变量，eps 代表0
fn=subs(fn,n,n1);       %计算傅里叶系数对各个n 的值
fn = double(fn);
subplot(2,1,1),stem(n1,fn,'filled');        %绘制频谱
line([-20 20],[0 0]);
title('周期矩形脉冲的频谱');
subplot(2,1,2),stem(n1,abs(fn),'filled');
title('周期矩形脉冲的幅度谱');
axis([-20 20 0 0.3])
grid on
```

五、实验内容及步骤

1.利用 MATLAB 画出下图所示的周期三角波信号的频谱。经计算，该周期三角波信号的傅立叶级数系数为 $c(n)=\begin{cases}\dfrac{-4j}{n^2\pi^2}\sin(\dfrac{n\pi}{2}) & n\neq 0\\ 0 & n=0\end{cases}$



编写的程序如下：

```
% 清除环境
clear_all;
% 定义傅立叶系数
n1 = -10:-1;  % 定义频谱的整数自变量，不包括0
n2 = 1:10;
n = -10:10;
c1 = 4*1i*sin(n1*pi/2)/pi^2./n1.^2;
c0 = 0;
c2 = 4*1i*sin(n2*pi/2)/pi^2./n2.^2;
cn = [c1 c0 c2];  % 计算傅立叶系数
% 创建figure对象
fig = figure;
% 绘制频谱
subplot(2,1,1);
stem(n,abs(cn),'filled');        %绘制幅度谱
title('周期三角波的幅度谱');
xlabel('频率');
ylabel('幅度');
set(gca,'FontName','Microsoft YaHei');  % 设置字体
subplot(2,1,2),stem(n,angle(cn),'filled');        %绘制相位谱
title('周期三角波的相位谱');
xlabel('频率');
ylabel('相位 (弧度)');
set(gca,'FontName','Microsoft YaHei');  % 设置字体
grid on
% 保存图像
save_figure_as_image(fig,'Program4_1')
```

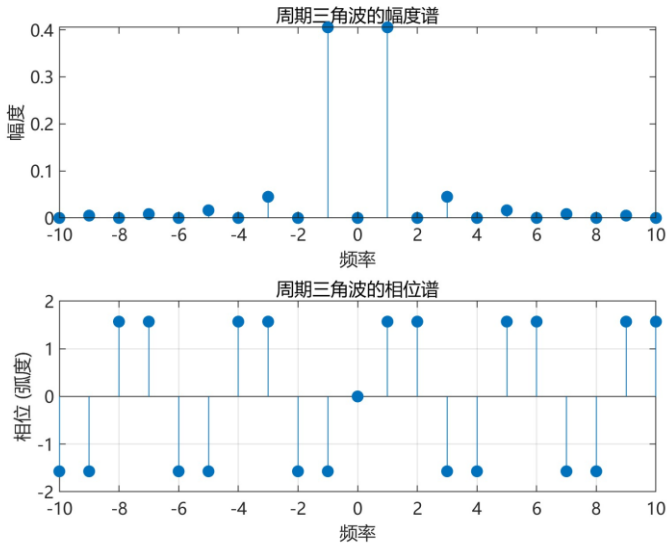


图 4-1. 周期三角波的分析

2.利用 MATLAB 采用数值方法近似计算三角波信号 $f(t)=\begin{cases}1-|t| & |t|\le 1\\ 0 & |t|>1\end{cases}$ 的频谱

编写的程序如下：

```
clear_all;
T = 2;  % 信号周期
N = 120;  % 采样点数
t = linspace(-T/2, T/2, N);
f = zeros(size(t));
f(abs(t) <= 1) = 1 - abs(t(abs(t) <= 1));
F = fftshift(fft(f)); % 计算傅立叶变换
Fs = N/T;  % 采样频率
frequencies = linspace(-Fs/2, Fs/2, length(F));
fig = figure;
subplot(2,1,1);
plot(frequencies,abs(F),"LineWidth",1.5);        %绘制幅度谱
title('三角波的幅度谱');
xlabel('频率');
ylabel('幅度');
set(gca,'FontName','Microsoft YaHei');  % 设置字体
subplot(2,1,2);
plot(frequencies,abs(F),"LineWidth",1.5);        %绘制相位谱
title('三角波的相位谱');
xlabel('频率');
ylabel('相位 (弧度)');
set(gca,'FontName','Microsoft YaHei');  % 设置字体
grid on
save_figure_as_image(fig, 'Program4_2')
```

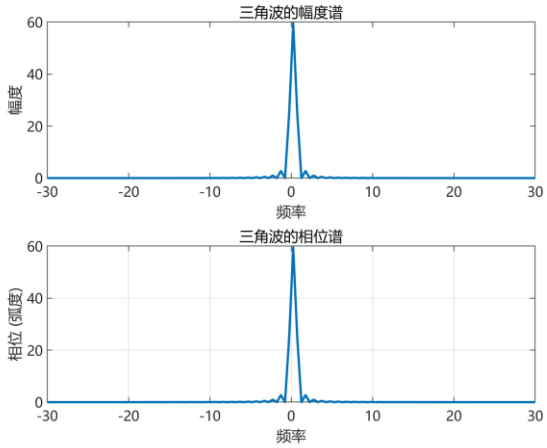


图 4-2. 三角波分析

3.利用 MATLAB 画出 $\alpha = \pm 0.9$ 时 $F(e^{j\Omega}) = \frac{1}{1-\alpha e^{-j\Omega}}$ 的幅度频谱。

编写的程序如下：

```
% 清除环境
clear_all;
% 定义不同的 alpha 值
alpha_values = [0.9, -0.9];
% 创建图形窗口
fig1 = figure;
% 对每个 alpha 值计算和绘制频响
for i = 1:length(alpha_values)
    alpha = alpha_values(i);
    % 定义系统函数
    b = 1;
    a = [1, -alpha];
    % 计算频响
    [H, W] = freqz(b, a, 'whole', 1000);
    % 绘制幅度频谱
    subplot(length(alpha_values), 1, i);
    plot(W/pi, abs(H),"LineWidth", 1);
    set(gca,'FontName','Microsoft YaHei');  % 设置字体
    title(['\alpha = ', num2str(alpha)]);
    xlabel('幅度谱 (\times\pi rad/sample)');
    ylabel('幅度');
end
save_figure_as_image(fig2,'Program4_3_2');
fig2 = figure;
for i = 1:length(alpha_values)
    alpha = alpha_values(i);
    n = -10:0.001:10;
    f = 1./(1-alpha.*exp(-1i*n));
    subplot(length(alpha_values), 1, i);
    plot(n, abs(f),"LineWidth", 1);
    set(gca,'FontName','Microsoft YaHei');
    title(['\alpha = ', num2str(alpha)]);
    xlabel('频率');
    ylabel('幅度');
    grid on;
end
save_figure_as_image(fig1,'Program4_3_1');
```

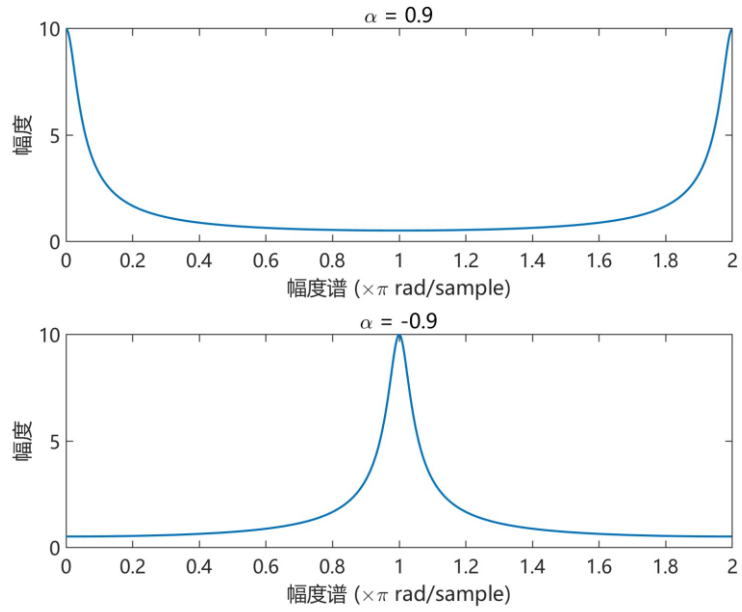


图 4-3. 弧度制下的幅度谱

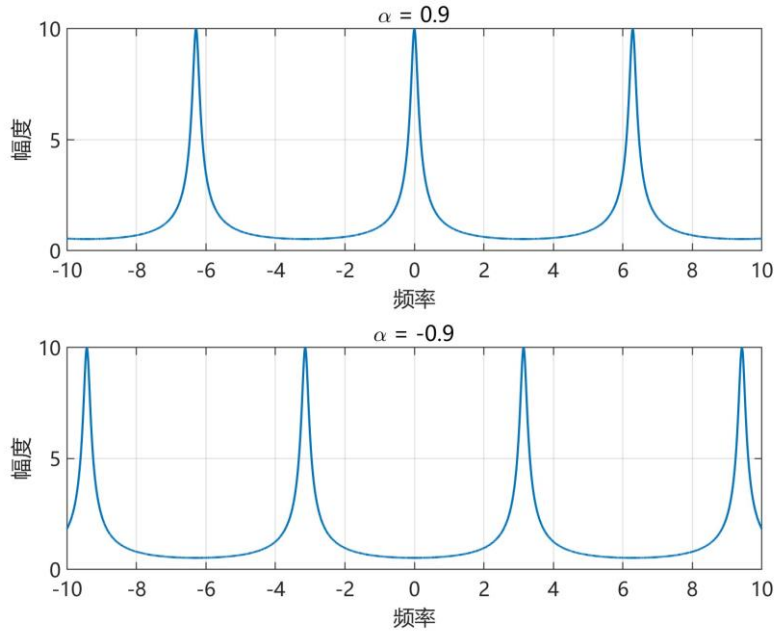


图 4-4. 频率分析

4.某连续线性系统的频率响应为：$H(j\omega)=\dfrac{1}{(j\omega)^3+2(j\omega)^2+2(j\omega)+1}$，利用 MATLAB 画出该系统的幅频响应$|H(j\omega)|$和相频响应$\phi(\omega)$。

编写的程序如下：

```
clear_all;
b = 1;  % 分子系数
a = [1, 2, 2, 1];  % 分母系数
[H , w]= freqs(b, a, linspace(0,5,200));
fig = figure;
subplot(2, 1, 1);
plot(w, abs(H),"LineWidth",1);
grid on;
set(gca,'FontName','Microsoft YaHei');  % 设置字体
title('幅度响应');
xlabel('频率 (rad/s)');
ylabel('幅度 (dB)');
% 绘制相位响应
subplot(2, 1, 2);
plot(w, angle(H),"LineWidth",1);
grid on;
set(gca,'FontName','Microsoft YaHei');  % 设置字体
title('相位响应');
xlabel('频率 (rad/s)');
ylabel('相位 (rad)');
% 保存图像
save_figure_as_image(fig,'Program4_4');
```

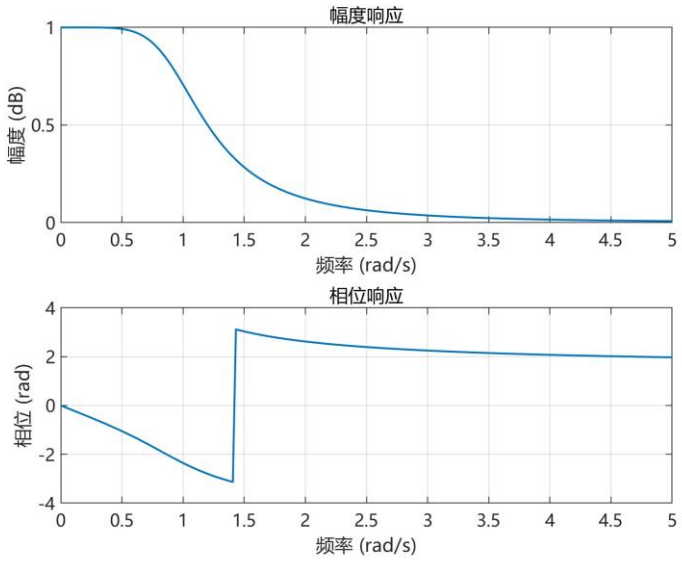


图 4-5. 响应图

5.信号为$\sin(2\pi 100t)$,分别画出采样速率 f_s=100，500 的时域波形，并求相应的频率，进行频域分析。

编写的程序如下：

```
clear_all;
fs_base = 10000;  % 一个足够高的采样频率，以模拟连续信号
t_base = 0:1/fs_base:10-1/fs_base;
x_base = sin(200*pi*t_base);
% 采样信号的采样频率
fs = [100, 500];
for i = 1:length(fs)
    t = 0:1/fs(i):10-1/fs(i);
    x = sin(200*pi*t);
    y = fft(x);
    n = length(x);
    f = (0:n-1)*(fs(i)/n);
    power = abs(y).^2/n;
    figure(i)  % 创建figure
    subplot(2,1,1);
    title(['fs = ', num2str(fs(i)), 'Hz']);
    plot(f,power);
    xlabel('frequency');
    ylabel('power');
    subplot(2,1,2);
    plot(t_base,x_base, 'k');  % 绘制基础信号
    hold on;
    plot(t,x, 'r');  % 绘制采样信号
    xlabel('time');
    ylabel('amplitude');
    save_figure_as_image(figure(i), ['Program4_5_fs_' num2str(fs(i)),'Hz']);
end
```

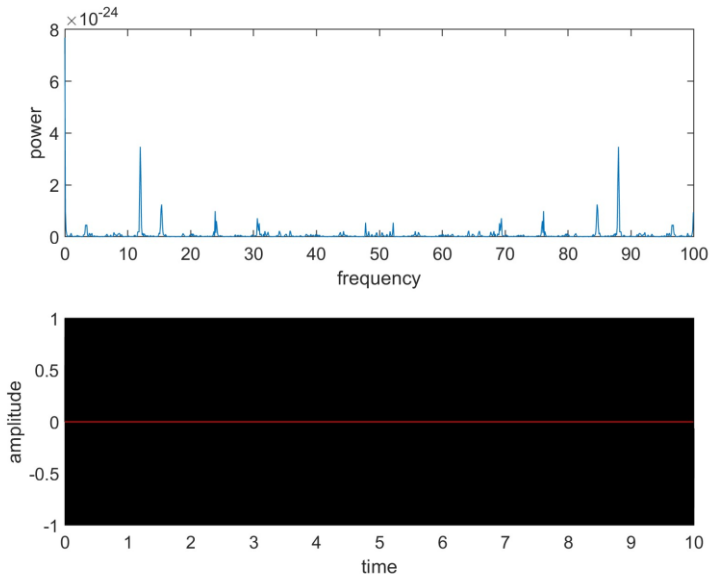


图 4-6. 100Hz 下的频域分析与采样信号

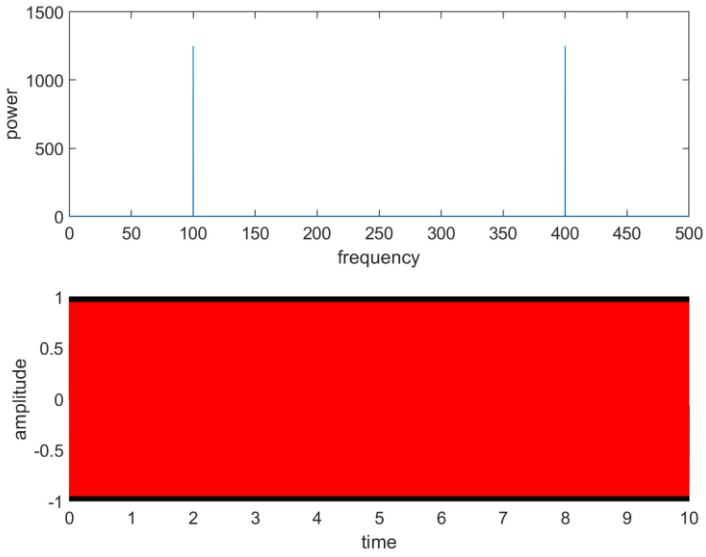


图 4-7. 500Hz 下的频域分析与采样信号

本实验完成时间： 2024 年 4 月 18 日

实验五 连续时间信号与系统的复频域分析

一、实验目的

掌握应用 MATLAB 实现对连续时间信号与系统的复频域分析，即

1. 掌握连续时间信号的拉普拉斯变换及其逆变换；
2. 掌握连续时间信号与系统的复频域分析，零、极点与单位冲击响应和稳定性。

二、主要设备

计算机，MATLAB 软件

三、实验原理

S 域分析是一种重要的复频域分析方法。复频域方法可以将微分方程转化为变换域中的代数方程，可以将卷积运算转化为变频域中的乘法运算，连续系统的 S 域分析在历史上一度是求解电路问题的首要方法。时至今日，变换域分析已经建立了一套完整的理论体系。从 $H(s)$ 研究系统的结构与参数，分析其零极点分布与系统的时域特性、频域特性的关系，讨论系统的稳定性情况，进行系统的改良、自动控制、不同结构的模拟与相互转换等。

1 Laplace 变换

Laplace 变换分为单边和双边两种形式。单边 Laplace 变换与反变换定义为：

$$\begin{cases} F(s)=\int_{0-}^{\infty} f(t)e^{-st}\mathrm{d}t \\ f(t)=\dfrac{1}{2\pi j}\int_{\sigma-j\infty}^{\sigma+j\infty} F(s)e^{st}\mathrm{d}s \end{cases} \tag{5-1}$$

MATLAB 内置了符号函数 laplace 和 ilaplace 函数对 Laplace 变换与反变化进行计算。laplace 和 ilaplace 函数使用形式如下：

```
Fs=laplace(ft,t,s);%求时域函数 ft 的 Laplace 变换 Fs
ft=ilaplace(Fs,s,t);%求频域函数 Fs 的 Laplace 反变换 ft
```

说明：t 是时域函数变量，s 是复频域函数的变量。ft 和 Fs 都是符号表达式。

5.1 利用符号函数计算常见信号 Laplace 变换。

```
% Program5_1
%符号计算常用的Laplace 变换
syms t s
syms a b positive  %限定常数为正值
Dt= str2sym('dirac(t-a)');
Ut= str2sym('heaviside(t-b)');
Mt=[Dt,Ut;exp(-a*t)*sin(b*t),t^2*exp(-t)];
MS=laplace(Mt,t,s)
```

运行结果：

MS =

[exp(-s*a), exp(-s*b)/s]

[1/b/((s+a)^2/b^2+1), 2/(s+1)^3]

5.2 利用 Laplace 变换计算 $f_1(t)=e^{-t}(t)$， $f_2=(t)te^{-1/2t}\varepsilon(t)$的符号卷积。

```
% Program5_2
%Laplace 变换求符号卷积
syms t,t=sym('t','positive');   %t 定义为"正的"符号变量
fs1=laplace(exp(-t));          %f1(t)的Laplace 变换
fs2=laplace(t*exp(-t/2));      %f2(t)的Laplace 变换
yt=simplify(ilaplace(fs1*fs2))   %利用Laplace 反变换求时域解
```

运行结果：

yt =

2*exp(-t)*(t*exp(t/2) - 2*exp(t/2) + 2)

2 零极点、稳定性与系统的频率特性

$H(s)$的极点位置决定系统的冲激响应形式和稳定性，零点位置影响增益和相位。

（1） $H(s)$的极点全在左半平面，系统稳定。

（2）$H(s)$的极点只要有一个极点在右半平面，或虚轴上有二阶及以上阶极点，系统不稳定。

（3） $H(s)$在虚轴上有一阶极点，其余极点全在左半平面，系统是临界稳定的。

从$H(s)$的零极点分布可以方便地描绘稳定系统的频率响应，可根据拉氏变换和傅氏变换的关系，由$H(s)$得到系统的频率响应特性$H(jw)$。

求系统函数的零点和极点可以用 MATLAB 的多项式求根函数 roots 来实现，调用函数 roots 的命令格式为：

```
P=roots(A)   %多项式求根
```

其中 A 为待求根的多项式的系数构成的行向量，返回向量 p 则是包含该多项式所有根位置的列向量。例如多项式 $A(s)=s^3+\frac{3}{4}s^2+\frac{1}{8}$，则求该多项式根的命令为：

```
A=[1 3/4 0 1/8];
p=roots(A)
```

运行结果为

p =

-0.9032

0.0766 + 0.3640i

0.0766 - 0.3640i

MATLAB 对于连续系统提供函数 pzmap 绘制零极点图。

```
pzmap(sys); %在 S 平面绘制系统函数的零极点图
[p,z]=pzmap(sys); %计算零极点（列向量），不绘零极图
```

系统 sys 可以是 tf、zpk、ss 等任一种系统表达式。

5.4 画出系统 $H(s)=\frac{s^2+4s+3}{s^4+3s^3+4s^2+6s+4}$ 的零极图，并判断系统的稳定性。

```
% Program5_4 系统稳定性判定与零极图程序
b=[1 4 3]; %分子系数，按降幂顺序排列
a=[1 3 4 6 4]; %分母系数，按降幂顺序排列
sys=tf(b,a)
pzmap(sys);sgrid;
azp=roots(a) %求出极点 azp  %根据参量 wd 的值判断稳定：1 表示稳定，0 表示不稳定
wd=1
for k = 1:length(azp)
      if real(azp(k))>-0.000001
          wd=0;
    end
    if wd==0
        title('不稳定系统');
    elseif wd==1
        title('稳定系统');
    end
end
```

结果：Transfer function:

```
          s^2 + 4 s + 3

  -------------------------------------

   s^4 + 3 s^3 + 4 s^2 + 6 s + 4

azp =

   0.0000 + 1.4142i

   0.0000 - 1.4142i

   -2.0000

   -1.0000
```

系统的零极图，如图 5.1 所示。

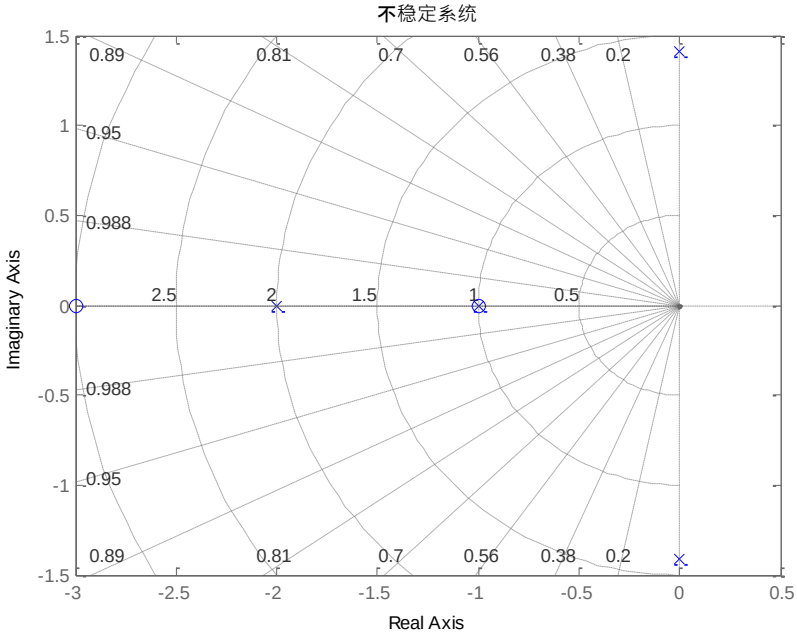


图 5.1 例 5-4 系统的零极图

说明：本系统有一对共轭虚根，是临界稳定。

如果系统函数的收敛域包含虚轴，那么令 $s = jw$，$H(s)$就成为$H(jw)$，反映系统的频率特性。MATLAB 提供 freqs 函数直接计算系统频率响应的值：

```
freqs(b,a,w); %直接绘制双对数坐标的幅频和相频特性曲线
H=freqs(b,a,w); %计算频率特性
[H,w]=freqs(b,a); %自动选频率区间的 200 个点计算频率特性
[H,w]=freqs(b,a,N); %在上述指令自选频率区间计算等分的 N 个点的频率特性
```

w 表示频率区间向量，b,a 分别表示 $H(s)$的分子和分母多项式系数。

5.5 绘制 $H(s)=\dfrac{s+1}{s^2+3s+2}$ 的幅频和相频特性曲线。

```
% Program5_5
w=0:0.01:10;%定义频率区间
b=[1 1];%分子系数，按降幂顺序排列
a=[1 3 2];%分母系数，按降幂顺序排列
H=freqresp(b,a,sqrt(-1)*w);%计算频率响应的值
subplot(2,1,1);
plot(w,abs(H));xlabel('w');ylabel('|H(jw))|');title('幅频特性');
subplot(2,1,2);
plot(w,angle(H));xlabel('w');ylabel('phase(w)|');title('相频特性');
```

系统的幅频特性和相频特性如图 5.2 所示。

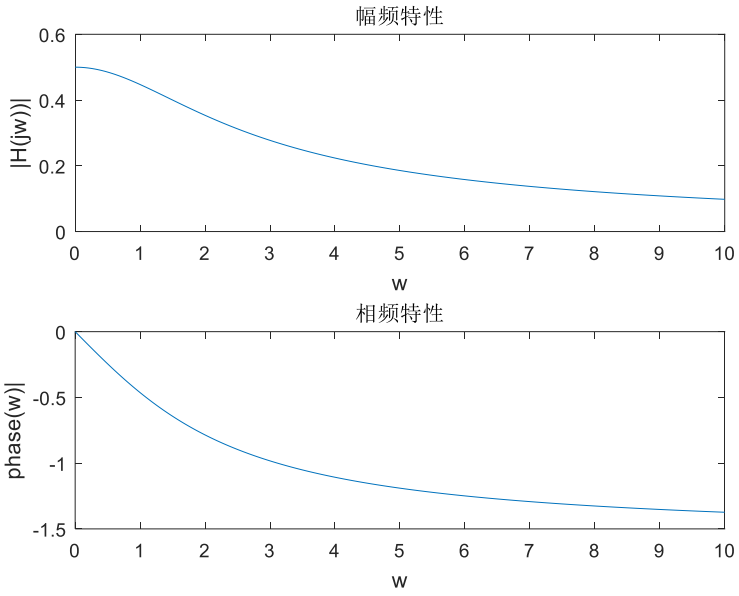


图 5.2 系统的幅频特性和相频特性

四、实验内容及步骤

1．系统 $H(s)=\dfrac{s+4}{s^2+3s+2}$，计算系统的零极点，利用函数 abs( )和 angle( )绘制系统的幅频和相频特性曲线。

编写的程序如下：

```
clear_all;
w = 0:0.01:10;
% 定义系统
b = [1 4]; % 分子系数，按降幂顺序排列
a = [1 3 2]; % 分母系数，按降幂顺序排列
H = tf(b, a);
% 计算零点和极点
zeros = zero(H);
poles = pole(H);
% 打印零点和极点
disp('零点:');
disp(zeros);
disp('极点:');
disp(poles);
% 计算频率响应
H_w = freqresp(H, w); % 计算频率响应的值
% 计算幅度和相位
mag = abs(squeeze(H_w));
phase = angle(squeeze(H_w));
fig = figure;
% 绘制幅频特性曲线
subplot(2,1,1);
plot(w, 20*log10(mag),"LineWidth",1); % dB scale
set(gca,'FontName','Microsoft YaHei');
xlabel('频率 (rad/s)');
ylabel('幅度 (dB)');
title('幅度响应');
% 绘制相频特性曲线
subplot(2,1,2);
set(gca,'FontName','Microsoft YaHei');
plot(w, unwrap(phase)*180/pi,"LineWidth",1); % degree scale
set(gca,'FontName','Microsoft YaHei');
xlabel('频率 (rad/s)');
ylabel('相位 (度)');
title('相位响应');
```

```
save_figure_as_image(fig, 'Program5_1');
```

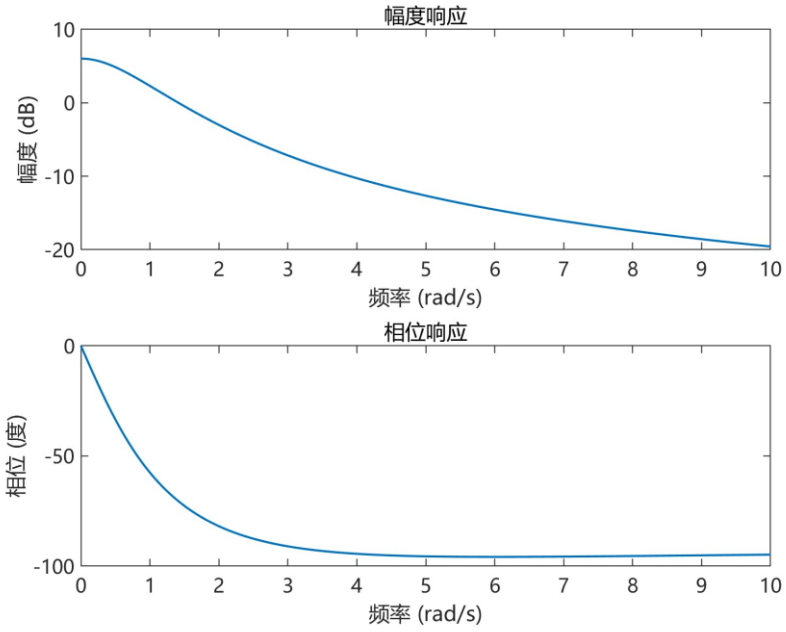


图 5-1. 系统的幅频和相频特性曲线

2．验证 Laplace 变换时域求导性质：$L[\frac{df(t)}{dt}]=sL[f(t)]-f(0)$。

编写的程序如下：

```
% 清除环境
clear_all;
tol = 1e-10;
syms t s f(t)
f(t) = sin(t);
F_s = laplace(f, t, s);
dfdt = diff(f, t);
DF_s = laplace(dfdt, t, s);
test = s*F_s - subs(f, t, 0);
fig = figure;
subplot(2, 1, 1)
ezplot(test, [0, 2*pi])
title_1 = ['test = s*F(s) - f(0)=', char(test)];
title(title_1)
subplot(2, 1, 2)
ezplot(DF_s, [0, 2*pi])
title_2 = ['DF(s)=', char(DF_s)];
title(title_2)
if(isequal_s(DF_s, test, tol))
    disp('验证通过')
end
save_figure_as_image(fig, 'Program5_2')
```

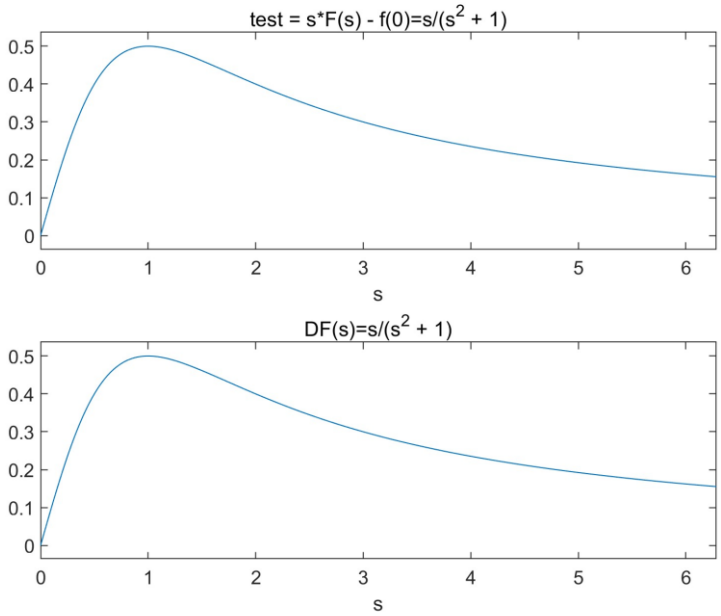


图 5-2. 验证图像与表达式

3．求 $F(s)=\dfrac{s+3}{s^3+3s^2+6s+4}$ 的 Laplace 反变换。

编写的程序如下：

```
% 清理环境
clear_all;
% 创建符号变量
syms s t
% 定义函数 F(s)
F = (s+3)/(s^3 + 3*s^2 + 6*s + 4);
% 计算 F(s) 的 Laplace 反变换
f = ilaplace(F);
disp(f)
fig = figure;
% 绘制 f(t) 的图像
fplot(f, [0, 10],"LineWidth",1)  % 绘制 t 在 [0, 10] 范围内的图像
% 添加包含 LaTeX 公式的标题
title(['$f(t) = ', latex(f), '$'], 'Interpreter', 'latex')
save_figure_as_image(fig, 'Program5_3')
```

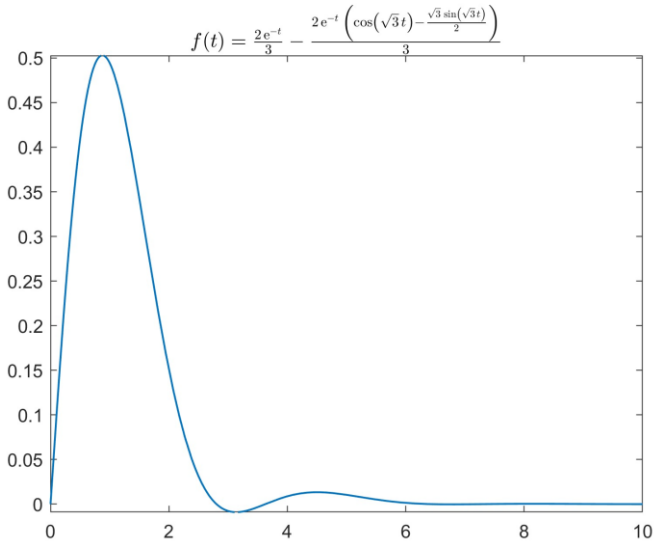


图 5-3. 所求函数图像与表达式

本实验完成时间：　　年　　月　　日

实验六 离散时间系统的时域分析的 MATLAB 实现

一、实验目的

掌握应用 MATLAB 实现对离散时间系统的时域分析，即

1. 掌握离散时间系统的时域分析差分方程迭代求解；
2. 掌握离散时间系统的单位脉冲响应、阶跃响应及系统响应。

二、主要设备

计算机，MATLAB 软件

三、实验原理

信号与系统时域分析主要研究 LTI 系统的差分方程模型、求解方法、响应的意义及分类等内容。本实验研究基于 MATLAB 的 LTI 系统的一般时域分析方法，包括系统的表示、方程求解、冲击响应与阶跃响应等。自然科学与社会科学中，很多动态的系统可以采用差分方程建模，而差分方程适合迭代求解。

1 差分方程迭代求解

线性时不变离散系统可以用如下所示的线性常系数差分方程来描述：

$$\sum_{i=0}^{N} a_i y(k-i) = \sum_{j=0}^{M} b_j f(k-i) \tag{6-1}$$

其中 $y(k)$ 为系统输出序列， $f(k)$ 为输入序列。其完全解包括零输入响应和零状态响应。

零输入响应：输入序列 $f(k)$ 为零时，由系统的初始状态引起的响应，系统等效为下式：

$$\sum_{i=0}^{N} a_i y(k-i) = 0 \tag{6-2}$$

零状态响应：系统的初始状态 $y(-1) = y(-2) = \Lambda\ = y(-N) = 0$，由输入 $f(k)$ 产生的响应。

系统的全响应为零输入响应与零状态响应的和。

利用系统的初始状态 $y(-1), y(-2), \Lambda\ , y(-N)$, 通过迭代可以分别求出系统有限长度的零输入响应，零状态响应和全响应。

6.1 差分方程：$y(k)-y(k-1)+0.5y(k-2)=0.4f(k)-2f(k-1)-f(k-2),$

$y(-2)=0.4, y(-1)=0.3,\ f(k)=\begin{cases}0 & k<0\\ \sin k & k\geq 0\end{cases}$，求系统响应 $y(k)\{0\leq k\leq 10\}$。

分析：由于 MATLAB 中数组的下标从 1 开始，因此，可以先将数据整体移位，使涉及的所有序列下标从大于等于 1 开始，计算出最终结果后进行反向移位。在序列 $y(k)$ 中，出现的最小负坐标为-2，假设 $y'(k)=y(k-3), f'(k)=f(k-3),$ 可得：

$y'(1)=y(-2),\ y'(2)=y(-1), y'(3)=y(0), f'(1)=0, f'(2)=0, f'(3)=\sin(3-3)=0$。

```
%差分方程的迭代求解
n=1:10+3;
F=sin(n-3);
F(1)=0;F(2)=0;                    %激励从零开始
Y(1)=0.4;Y(2)=0.3;                %输入初始条件
%迭代求解
for k=3:13
Y(k)=0.4*F(k)-2*F(k-1)-F(k-2)-0.5*Y(k-2)+Y(k-1);
end
n1=n-3;
stem(n1,Y);xlabel('k');ylabel('f(k)');grid on;
```

程序运行结果如图 6.1 所示。

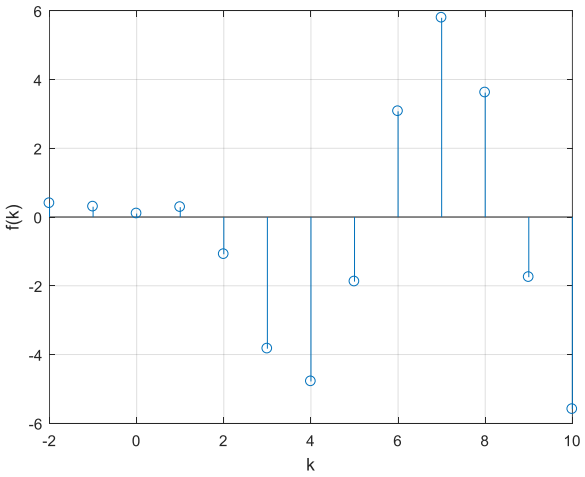


图 6.1 差分方程的迭代求解

2 冲激响应（单位样值响应）和阶跃响应

差分方程：

$$\sum_{i=0}^{N} a_i y(k-i) = \sum_{j=0}^{M} b_j f(k-j) \tag{6-3}$$

可以用算子方程描述为：

$$\sum_{i=0}^{N} a_i Y(z) z^{-i} = \sum_{j=0}^{M} b_j F(z) z^{-j} \tag{6-4}$$

式(6-4)通过对差分方程（零状态）作 Z 变换得到。

离散系统系统函数 $H(z)$ 为

$$H(z) = \frac{Y(z)}{F(z)} = \frac{\sum_{j=0}^{M} b_j z^j}{\sum_{i=0}^{N} a_i z^i} \tag{6-5}$$

单位样值响应 $h(k)$ 的定义为：输入 $f(k) = \delta(k)$ 时系统的零状态响应，表示为 $h(k)$，它和系统函数 $H(z)$ 组成 Z 变换对：

$$\begin{cases} H(z) = Z[h(k)] \\ h(k) = Z^{-1}[H(z)] \end{cases} \tag{6-6}$$

1.dstep——求单位序列响应

dstep 是离散系统计算单位序列响应（阶跃响应）的函数。用法为：

```
dstep(b,a);                %绘出单位序列响应
Y=dstep(b,a,N);            %计算单位序列响应并保存为Y
```

b,a 分别表示 $H(z)$ 分子与分母多项式系数；N 是可选项，代表计算结果长度。

为了避免编程时 b,a 被误用和提高程序可读性，后续部分 b,a 统一换为 num 和 den 表示。

6.2 给定离散系统 $y(k) - 0.5y(k-1) + 0.125y(k-2) = f(k) + f(k-1)$ 求系统的单位样值响应和阶跃响应，并画出前 50 个样值。

```
% Program6_2
%计算单位样值响应与阶跃响应
num=[1 1 0];den=[1 -0.5 0.125];N=20;                %输入系统参数
y=dimpulse(num,den,N);                               %求系统的单位样值响应
k=0:N-1;
subplot(3,1,1);
stem(k,y,'filled');                                  %画出单位样值响应
xlabel('k');ylabel('h(k)'),title('dimpulse');
subplot(3,1,2);
impz(num,den,N);                                     %求单位样值响应并绘图
xlabel('k');ylabel('h(k)'),title('impz');
subplot(3,1,3);
y1=dstep(num,den,N);
stem(k,y1,'filled');
xlabel('k');ylabel('g(k)'),title('dstep');       %画出阶跃响应
```

程序运行结果如图 6.2 所示。

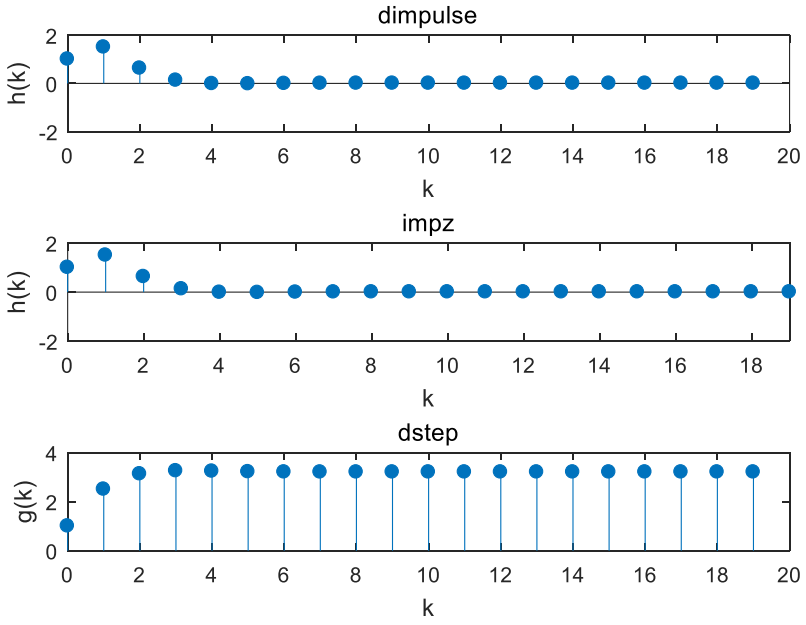


图 6.2　系统的单位样值响应与单位序列响应

3 系统分析

离散系统的输出可以分为零输入响应和零状态响应两部分，合起来成为全响应。系统分析可以通过差分方程迭代求解、符号微分方程求解、ode 指令集等。本实验分别求解零输入响应和零状态响应。零状态响应可以通过计算激励函数与冲激响应（单位样值响应）卷积的

方法求解；零输入响应具有齐次解的形式，由特征根的值形成解空间，结合初始条件求解系数即可。最后利用 lsim、dlsim、filter 等函数进行系统分析。

6.3 分析系统 $H(z)=\dfrac{1-0.5z^{-1}}{1+\dfrac{3}{4}z^{-1}+\dfrac{1}{8}z^{-2}}$ 在信号 $f(k)=0.5^k\,\varepsilon(k)$ 激励下的零状态响应。

```
% Program6_3
A=[1 3/4 1/8];B=[1 -0.5 0];
k=0:20;U=0.5.^k;                    %生成激励信号
Y1=dlsim(B,A,U);
subplot(2,1,1);stem(k,Y1,'filled');title('dlsim');
Y=filter(B,A,U);
subplot(2,1,2);stem(k,Y1,'filled');title('filter');
```

程序运行结果如图 6.3 所示。

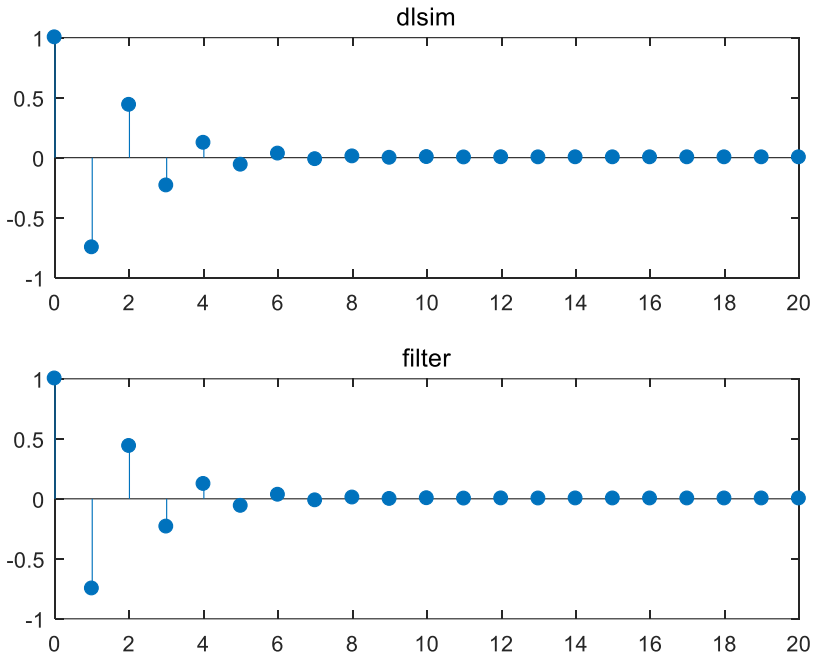


图 6.3　例 6-3 运行结果

四、实验内容及步骤

已知 $y(k)+2y(k-1)+2y(k-2)=f(k)+f(k-1), y(-1)=1, y(-2)=2, f(k)=u(k)$，求系统的 $h(k)$、$g(k)$、$y(k)$。

编写的程序如下：

```
clear_all;
% 定义系统参数
a = [1 2 2];
b = [1 1];
% 定义输入信号f(k)
t = 1:10;
f = u(t);
% 计算脉冲响应h(k)
h = dimpulse(b, a, t);
% 计算单位阶跃响应g(k)
g = dstep(b, a, t);
y(1) = 2;
y(2) = 1;
for k = 3:10
    y(k) = f(k)+f(k-1)-2*y(k-1)-2*y(k-2);
end
y = y(t);
% 绘制h(k), g(k)和y(k)
fig = figure;
subplot(2, 2, 3);
stem(h);
title('Impulse Response h(k)');
subplot(2, 2, 4);
stem(g);
title('Step Response g(k)');
subplot(2, 2, 2);
stem(t-3, y);
title('Output y(k)');
subplot(2, 2, 1);
stem(t, f);
title('Input f(k)');
save_figure_as_image(fig, 'Program6');
```

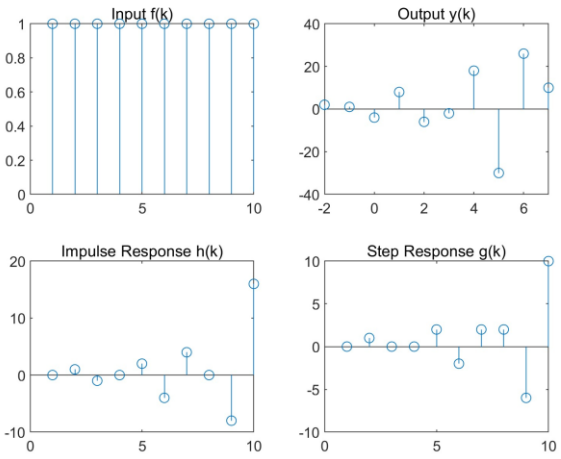


图 6-1. 系统信号图像

本实验完成时间：　　年　　月　　日

实验七 离散时间信号与系统的 Z 域分析

一、实验目的

掌握离散时间信号的 z 变换和逆 z 变换的实现方法，掌握离散时间系统的 z 域分析方法，熟悉 MATLAB 相应函数的调用格式和作用，掌握使用 MATLAB 来分析离散时间信号与系统的 z 域变换特性、实现离散系统的零、极点分析及稳定性分析的方法。

二、主要设备

计算机，MATLAB 软件

三、实验原理

1 *ztrans* 函数

ztrans 函数用来求离散序列的 z 变换。调用格式为：

$F = ztrans(f)$ 实现函数 $f(n)$ 的 z 变换，默认返回函数 F 是关于 z 的函数，即

$$f = f(n) \Rightarrow F = F(z)$$

2 *iztrans* 函数

iztrans 函数用来实现 z 逆变换。调用格式为：

$f = iztrans(F)$ 实现函数 $F(z)$ 的 z 逆变换，默认返回函数 f 是关于 n 的函数。

3 *residuez* 函数

residuez 函数用来进行 z 域的部分分式展开。
设离散系统的 z 域函数用下列有理分式来表示：

$$F(z) = \frac{B(z)}{A(z)} = \frac{b_m z^{-m} + b_{m-1} z^{-m+1} + \mathrm{L} + b_1 z^{-1} + b_0}{a_n z^{-n} + a_{n-1} z^{-n+1} + \mathrm{L} + a_1 z^{-1} + a_0}$$

可以将 $F(z)$ 展开成部分分式之和的形式，再对其求 z 逆变换。MATLAB 的信号处理工具箱提供了对 $F(z)$ 进行部分分式展开的函数 *residuez* 。

调用格式为：$[r,p,k]=residuez(b,a)$。其中，输入参量b为$F(z)$的分子多项式系数构成的行向量，a为$F(z)$的分母多项式系数构成的行向量，b和a都按z^{-1}升幂排列。函数将返回三个输出量r、p、k，其中r为$F(z)$部分分式展开系数的列向量，p为极点的列向量，k为多项式的系数的行向量，若为真分式$(m\langle n)$，则k返回为空阵。

利用*residuez*函数可以将有理分式展开为：

$$F(z)=\frac{B(z)}{A(z)}=\frac{r(1)}{1-p(1)z^{-1}}+\mathrm{L}\ +\frac{r(n)}{1-p(n)z^{-1}}+k(1)+k(2)z^{-1}+\mathrm{L}$$

如果$P(j)=\mathrm{L}\ P(j+m-1)$是$m$重极点，则展开式包含下面的结果：

$$\frac{r(j)}{1-p(j)z^{-1}}+\frac{r(j+1)}{(1-p(j)z^{-1})^2}+\mathrm{L}\ +\frac{r(j+m-1)}{(1-p(j)z^{-1})^m}$$

$[b,a]=residuez(r,p,k)$将部分分式展开式转换成B/A形式。

4 *zplane*函数

*zplane*函数用来绘制离散系统 z 平面的零、极点分布图。调用格式为：

$zplane(Z,P)$　以单位圆为参考圆绘制Z为零点列向量、P为极点列向量的零极点图。每个零点用'o'表示，每个极点用'x'表示。若用重复点，再重复点右上角以数字标出重数。

$zplane(B,A)$　B和A分别是传递函数$H(z)=B(z)/A(z)$按z^{-1}的升幂排列的分子分母系数行向量。注意，当B和A同为标量时，如B为零点，则A为极点。

四、实验内容及步骤

1 试利用 MATLAB 的符号运算实现下列序列的 z 变换。

①单边指数序列$f(n)=a^n u(n)$；

②阶跃序列$f(n)=u(n)$；

③单位样值序列$f(n)=\delta(n)$。

注意：在符号工具箱中，单位阶跃序列$u(n)$是用 heaviside(n)表示，而单位样值序列

$\delta(n)$是用 kroneckerDelta(n, 0)表示。

```
clear_all;
% 导入所有必要的符号函数
syms n z a
% 定义序列
f1 = a^n * heaviside(n);  % 单边指数序列
f2 = heaviside(n);  % 阶跃序列
f3 = kroneckerDelta(n, 0);  % 单位样值序列
% 计算 Z 变换
F1 = ztrans(f1, n, z);
F2 = ztrans(f2, n, z);
F3 = ztrans(f3, n, z);
% 显示结果
disp('Z-transform of f1(n):')
pretty(F1)
disp('Z-transform of f2(n):')
pretty(F2)
disp('Z-transform of f3(n):')
pretty(F3)
```

```
Z-transform of f1(n):
1     a
- - -----
2   a - z


Z-transform of f2(n):
  1     1
----- + -
z - 1   2


Z-transform of f3(n):
1
```

图 7-1. 运行结果截图

2 已知函数① $F(z)=1$，② $F_1(z)=\dfrac{1}{(z-2)(z-3)}$，③ $F_2(z)=\dfrac{z^2}{(z-2)(z-3)^3}$，利用 MATLAB 求单边 z 逆变换。

```
clear_all;
syms z n
F = 1;
f = iztrans(F, z, n);
disp('The inverse Z-transform of F(z) = 1 is:');
pretty(f)
F_1 = 1/((z-2)*(z-3));
f_1 = iztrans(F_1, z, n);
disp('The inverse Z-transform of F_1(z) = 1/((z-2)*(z-3)) is:');
pretty(f_1)
F_2 = z^2/((z-2)*(z-3)^3);
f_2 = iztrans(F_2, z, n);
disp('The inverse Z-transform of F_2(z) = z^2/((z-2)*(z-3)^3) is:');
pretty(f_2)
```

```
The inverse Z-transform of F(z) = 1 is:
kroneckerDelta(n, 0)


The inverse Z-transform of F_1(z) = 1/((z-2)*(z-3)) is:
 n    n
3    2    kroneckerDelta(n, 0)
-- - -- + --------------------
 3    2             6


The inverse Z-transform of F_2(z) = z^2/((z-2)*(z-3)^3) is:
 n / n - 1 \
3  |       |             n    n
   \   2   /      n   4 3    3  (n - 1)
------------ - 2 2  + ---- - ----------
      3                3         3
```

图 7-2. 运行结果截图

本实验完成时间：　　年　　月　　日