

实验内容：

(1) 生成连续信号 $f(t) = (2 + e^{-t})u(t + 1)$ ，在自变量范围 $(-2, 4)$ 内绘图。

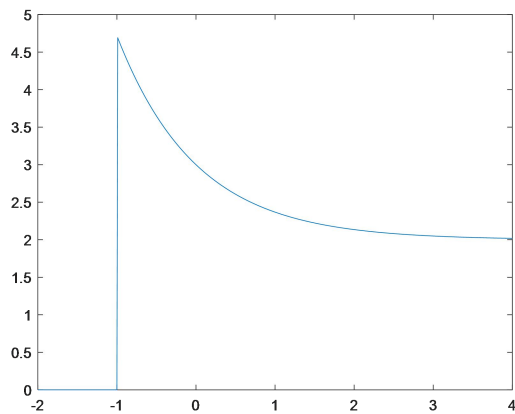
代码：

```
t=-2:0.01:4;
```

```
f=(2+exp(-t)).*u(t+1);
```

```
plot(x,f)
```

图像：



(2) 生成连续信号 $f(t) = \cos(t) u(\sin(t) + 0.3)$ ，在自变量范围 $(-10, 10)$ 内绘图

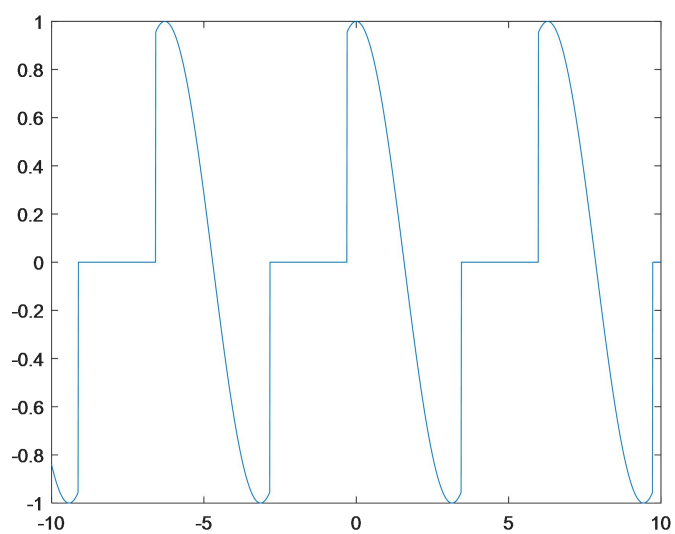
代码：

```
t=-10:0.01:10
```

```
f=cos(t).*u(sin(t)+0.3);
```

```
plot(t,f)
```

图像：



(3) 生成离散信号 $f[n] = (2 - 0.8n)u[n]$ ，在自变量范围 $(-2, 4)$ 内绘图。

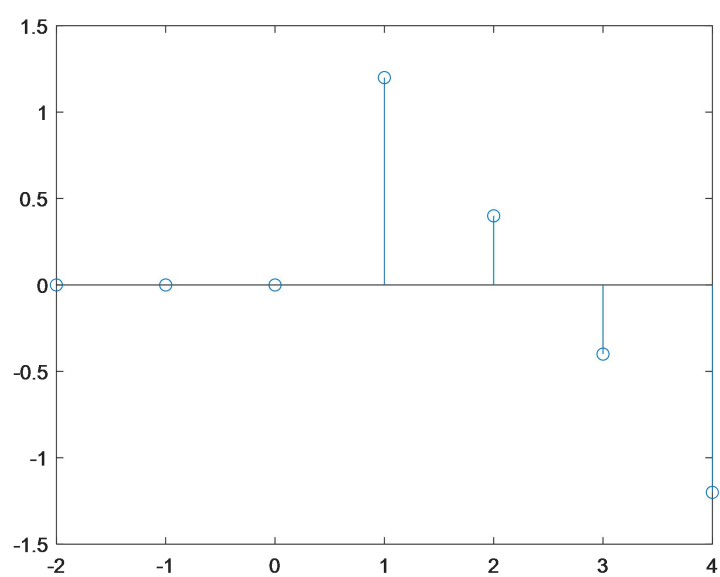
代码：

```
t=-2:4
```

```
f=(2-0.8*t).*u(t);
```

```
stem(t,f);
```

图像：



(4) 在多张子图上绘制以下信号（提示，使用 subplot 函数）

代码：

```
t1=linspace(-5,5,100)
```

```
t2=linspace(-15,15,100);
```

```
f1=4*sin(3*t1+pi/2);
```

```
f2=4*sin(pi/6*t2);
```

```
subplot(2,1,1);
```

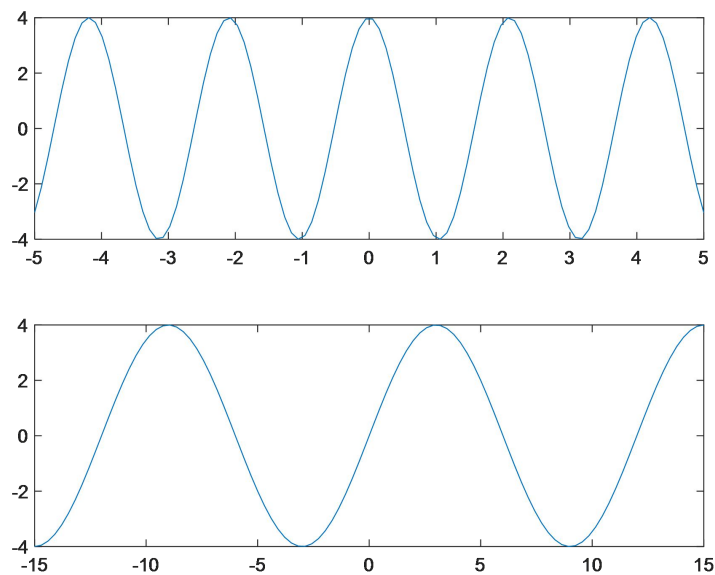
```
plot(t1,f1)
```

```
hold
```

```
subplot(2,1,2);
```

```
plot(t2,f2);
```

图像：



(5) 在多张子图上绘制以下信号（提示，使用 subplot 函数）

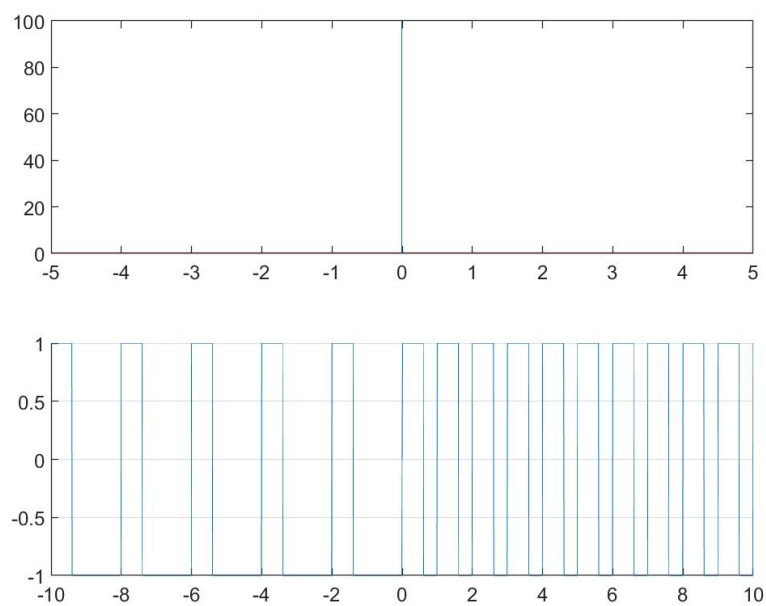
代码 1:

```
>> t=-5:0.01:5;  
  
>>n=length(t)  
  
>>f=zeros(1001)  
  
>>f(500)=1/0.01  
  
>> subplot(2,1,1);  
  
>> stairs(t,f)
```

代码 2:

```
>> t=-10:0.01:10;  
  
>> f1=square(t*pi,30).*u(-t)+square(t*2*pi,60).*u(t);  
  
>> plot(t,f1);  
  
>> axis([-10,10,-1.2,1.2])
```

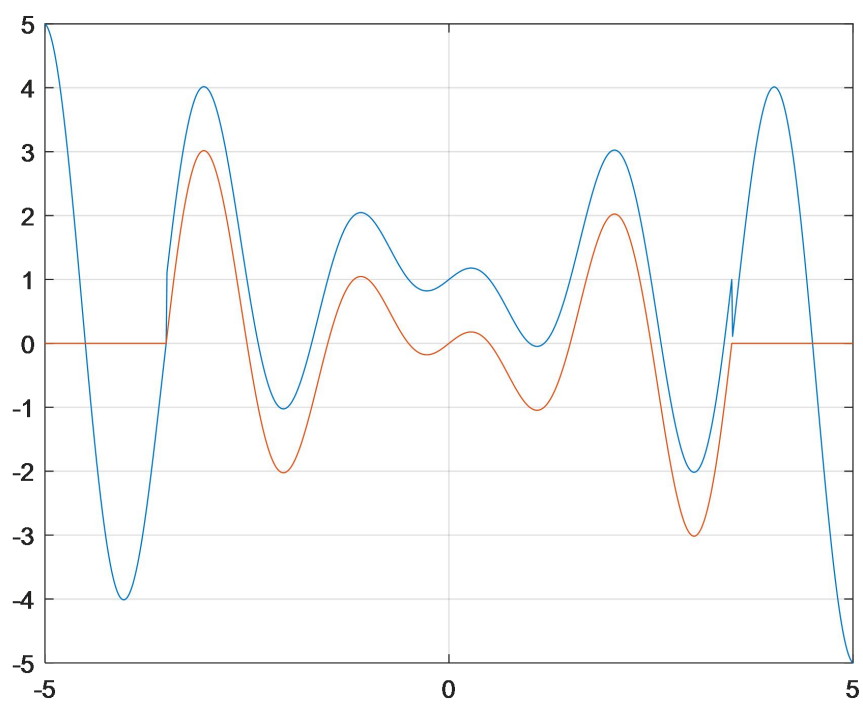
图像



代码 1:

```
f1=u(t+3.5)-u(t-3.5);  
  
f2=t.*cos(pi*t);  
  
>> plot(t,f1+f2);  
  
>> grid  
  
>> hold on  
  
>> plot(t,f1.*f2);  
  
>> legend('f1+f2','f1*f2');
```

图像 1:



代码 2:

微分的代码

```
t=-5:0.01:5;  
  
y=diff(f1.*f2)/0.01;
```

```
t=-5:0.01:4.99;
```

```
subplot(2,1,1);
```

```
plot(t,y)
```

积分的代码：

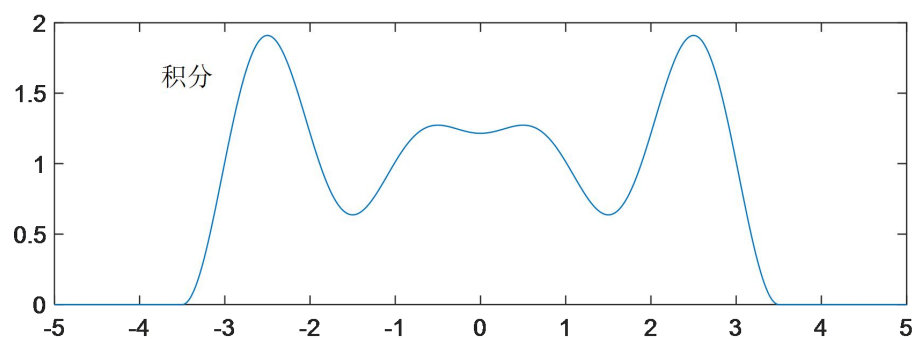
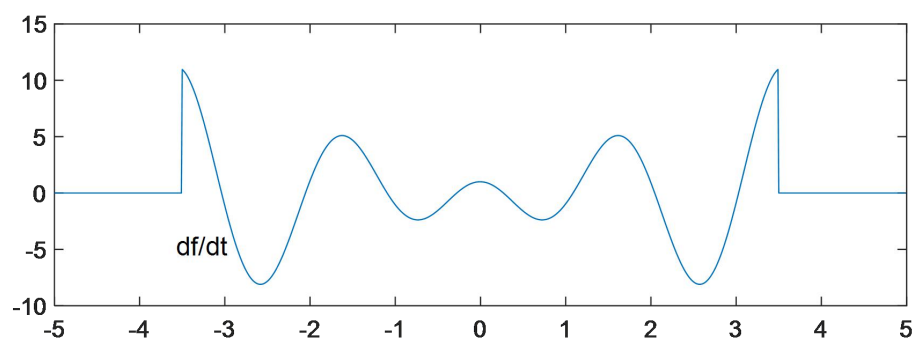
```
>> for x=1:length(t)
```

```
y(x)=quad('f4(t)',-5,t(x));
```

```
end
```

```
>> plot(t,y);
```

```
function y=f4(t)
    y=( ((t+3.5)>0)-((t-3.5)>0) ).*t.*cos(pi*t);
end
```



代码：

$f(t-1)$

```
>> t=-5:0.01:5;
```

```
>> f=t.*(u(t)-u(t-1));
```

```
>> x1=t+1;
```

```
>> plot(x1,f)
```

```
>> axis([-5,5,-0.2,1.2])
```

```
>> grid
```

$f(t + 2)$

```
>> subplot(2,2,2)
```

```
>> plot(x2,f)
```

```
>> axis([-5,5,-0.2,1.2]);
```

```
>> grid
```

$f(0.5t)$

```
>> subplot(2,2,3)
```

```
>> x3=2*t;
```

```
>> plot(x3,f);
```

```
>> grid
```

```
>> axis([-5,5,-0.2,1.2]);
```

$f(2t)$

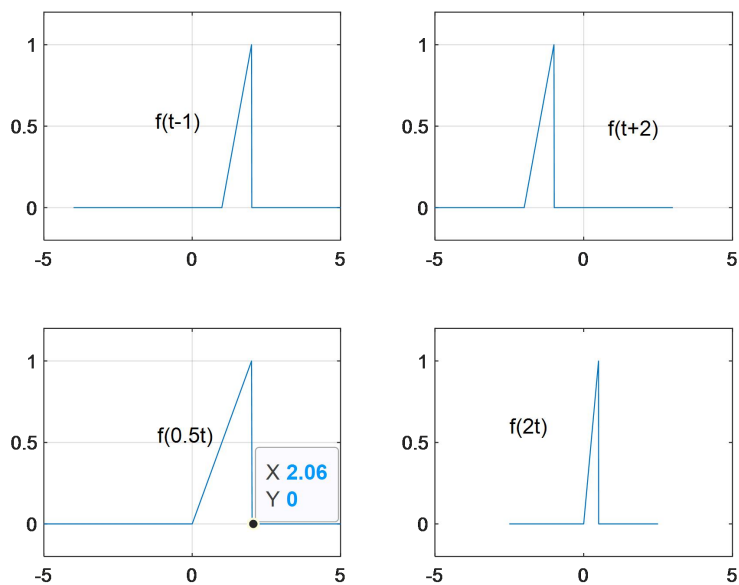
$x_4=0.5t;$

```
>> x4=0.5*t;
```

```
>> plot(x4,f);
```

```
>> axis([-5,5,-0.2,1.2]);
```

图像：



思考题：

- (1) 对于离散时间信号，单位冲激函数是一个信号，在 0 点值为 1，其他点处为 0 的信号；对于连续时间来说，单位冲激函数是一种理想化信号，是由

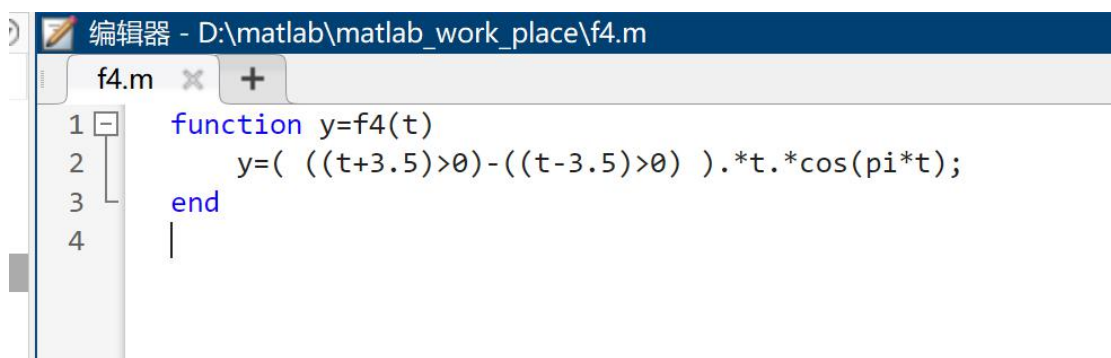
$u_{\Delta}(t)$ 信号的微分取极限得到的一种理想信号，本身是一个奇异函数。

Matlab 中，离散时间的单位冲激信号很好产生，对与连续时间的单位冲

激的函数，我们使用 $\delta_{\Delta}(t)$ 来近似表达单位冲激函数。

- (2) 好处就是模块化调用，写好之后就可以反复调用，减少每次都要数学相应的时间。缺点就是（不知道是不是我操作问题），定义的函数不能在里面调用之前自己定义的函数，所以定义函数的时候很麻烦，需要将所有函数重写。

定义函数的方法就是这样，写一个文档，然后将它保存在工作目录，就可以直接以调用函数的形式来调用了。注意定义函数的名字要和保存的文件同名。



The image shows a MATLAB editor window titled '编辑器 - D:\matlab\matlab_work_place\f4.m'. The window contains a function definition for 'f4.m'. The code is as follows:

```
1 function y=f4(t)
2     y=( ((t+3.5)>0)-((t-3.5)>0) ).*t.*cos(pi*t);
3 end
4 |
```

- (3) 下面是求微分的一次代码, y_1 是原函数在 t 对应的函数值，因为 t 是以 0.01 的单位来进行分割的, 那么 y_1 在 t_0 时刻的导数就近似 $y_1(t_0+0.01)-y_1(t_0)/(0.01)$, 所以我们只用调用 $y=\text{diff}(y_1)/0.01$, 就可以生成对应函数的导数数组了，注意一点，本来 y_1 数组长度是 l , 那么 y 的数组长度就是 $l-1$, 使用 plot 函数时要注意这一点。

```
t=-5:0.01:5;
```

```
y1=cos (t)
```

```
y=diff(y1)/0.01;
```

收获和感想

- (1) 首先对通过上机，对 matlab 的基础操作有了一定认识，知道了 matlab 一些刻画函数的方法。
- (2) 学习了如何创建自己的函数和对函数进行积分和微分。
- (3) 了解 matlab 的绘画图像的功能，学会之后可以对学习有很好的帮助。
- (4) 对于函数的基本变换，比如题（7）中，对函数的移动和伸缩变换，并不是很好操作，如果自己将函数带入计算，十分麻烦，这是我们只需要对自变量进行相反的变换，就可以很好的达到对应的效果，比如 x , $f(x)$ 本来时函数的自变量和因变量，我们求的 $f(x-1)$ ，我们只需要将 $t=x+1$, $\text{plot}(t,f)$ 就可以了。
- (5) 最后就是如何在函数的标签，比如说 `xlabel`，中插入 latex 的表达式，因为普通表达式不美观，插入 latex 公式看的很舒服。用下下面的命令方式就行了。

标签显示 latex 语句

matnworks 中国。

例子

现在以 `xlabel` 函数和 `ylabel` 函数为例，介绍使用方法。

```
1 x = [1 2 3 4 5 6];
2 y = [1 2 3 4 5 6];
3 p = plot(x, y);
4 txt = xlabel('$P_t/\sigma_0^2$');
5 set(txt, 'Interpreter', 'latex');
6 txt = ylabel('$\bar{R}_s$');
7 set(txt, 'Interpreter', 'latex');
```