中山大学计算机学院
信号与系统
实验二 信号的卷积

（2024 学年春季学期）

课程名称：Signal and System

教学班级	202320353	专业（方向）	计算机科学与技术
学号	22320107	姓名	饶鉴晟
班级	计算机 3 班	日期	2024/4/11

一、 实验题目

1. 验证程序实例中的相关程序

2. 编写程序，绘制下列信号的卷积波形。

A. 已知$x_1(t) = tu(t)$，$x_2(t) = e^{-t}u(t)$，求 $x_1(t) * x_2(t)$ （要求：抽样频率$fs = 1000$；时间 $t = -1.1 \sim 2.1$）

B. 已 知 $x[n] = [3, 2, 1, -2, 1, 0, 4, 0, 3; n = 0.8]$； $h[n] = [1, -2, 3, -4, 3, 2, 1; n = 0{:}6]$；求$x[n] * h[n]$.

二、 实验内容

1. 实验原理

针对实验二，我们需要理解的相关原理包括信号卷积的概念和相关的数学表达式。

（1） 实验原理

卷积是信号与系统领域中一个重要的数学运算，它描述了两个函数通过某种方式结合生成第三个函数的过程。在信号处理中，卷积运算经常用于描述线性时不变系统（LTI）对输入信号的响应。连续时间信号的卷积定义为两个信号乘积的积分，反映了一个信号在另一个信号上的“加权滑动”效果。离散时间信号的卷积则是两个信号乘积的累加。

卷积的直观理解是，将一个信号与另一个信号的翻转版本进行“滑动”相乘，然后对结果进行积分（连续信号）或求和（离散信号），得到的结果即为卷积结果。在物理或工程问题中，卷积经常用来描述系统对输入的动态反应。

（2） 核心公式

对于连续信号 $x_1(t)$ 和 $x_2(t)$，它们的卷积 $y(t)$ 定义为：

$$y(t) = (x_1 * x_2)(t) = \int_{-\infty}^{+\infty} x_1(\tau) \cdot x_2(t-\tau) d\tau$$

对于离散信号 $x_1[n]$ 和 $x_2[n]$，它们的卷积 $y[n]$ 定义为：

$$y[n] = (x_1 * x_2)[n] = \sum_{k=-\infty}^{+\infty} x_1[k] \cdot x_2[n-k]$$

在实验二中，如果需要计算两个离散时间信号的卷积，通常采用 $NumPy$ 或 $Scipy$ 中的函数（如 $np.convolve$ 或 $sg.convolve$），它们能够自动处理上述的离散信号卷积计算。

实验二中还需要利用到卷积性质，例如卷积定理表明，在傅里叶变换的帮助下，两个信号的卷积可以转换为它们的傅里叶变换的乘积，这在计算上可能更加高效。

在理解以上实验原理的基础上，需要编写相应的$Python$程序来实现信号的卷积计算，这是实验二中需要完成的关键步骤。

2. 伪代码

任务 A:

Algorithm 1: Convolution of two signals

Input: Sampling rate fs, time vectors $t1$ and $t2$, signals $x1$ and $x2$
Output: Convolved signal $y1$, and time vector tt

```
Initialize t1 as an array of time points from -1100 to 2100, with
  step size 1/fs;
Initialize x1 as an array, where x1[t] = t if t ≥ 0, and 0 otherwise;
Initialize t2 as an array of time points from -1100 to 2100, with
  step size 1/fs;
Initialize x2 as an array, where x2[t] = e^{-t} if t ≥ 0, and 0
  otherwise;
y1 ← convolve(x1, x2)/fs;
n ← length(y1);
tt ← linspace(−2200, 4201, n)/fs;
for i ← 1 to 2 do
    for j ← 1 to 2 do
        if i == 1 j == 1 then
            Plot x1 against t1;
            Add grid;
            Set title to 'x1(t)';
        end
        else if i == 1 j == 2 then
            Plot x2 against t2;
            Add grid;
            Set title to 'x2(t)';
        end
        else
            Plot y1 against tt;
            Add grid;
            Set title to 'conv(x1,x2)';
        end
    end
end
Show the plot;
```

图 1

任务 B：

Algorithm 1: Convolution of two discrete-time signals

```
Input: Discrete-time signals x1 and x2
Output: Convolved signal y, and time index vector n3
Initialize n1 as an array of time indices from 0 to 8;
Initialize x1 as an array of signal samples: [3, 2, 1, −2, 1, 0, 4, 0, 3];
Create a figure with 2 rows and 2 columns;
for i ← 1 to 2 do
    for j ← 1 to 2 do
        if i == 1 j == 1 then
            Plot x1 against n1 using stem plot;
            Add grid;
            Set title to 'x[n]';
        end
        else if i == 1 j == 2 then
            Initialize n2 as an array of time indices from 0 to 6;
            Initialize x2 as an array of signal samples:
             [1, −2, 3, −4, 3, 2, 1];
            Plot x2 against n2 using stem plot;
            Add grid;
            Set xticks to integer values from 0 to 6;
            Set title to 'h[n]';
        end
        else
            y ← convolve(x1, x2, 'full');
            n3 ← linspace(0, 14, 15);
            Plot y against n3 using stem plot;
            Add grid;
            Set title to 'Conv Sum y[n]';
            Set x-axis label to 'Time index n';
        end
    end
end
Adjust the subplots spacing;
Show the plot;
```

图 2

3. 关键代码展示（带注释）

任务 A：

```python
# 导入必要的库
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sg

# 定义采样频率
fs = 1000

# 创建时间序列 t1 和 t2
t1 = np.array([t/fs for t in range(-1100, 2101)])
t2 = np.array([t/fs for t in range(-1100, 2101)])

# 生成信号 x1(t) 和 x2(t)
x1 = np.array([t if t >= 0 else 0 for t in t1])
x2 = np.array([np.exp(-1 * t) if t >= 0 else 0 for t in t2])

# 计算 x1 和 x2 的卷积并除以采样频率以获得正确的振幅
y1 = sg.convolve(x1, x2) / fs

n = len(y1)
tt = np.linspace(-2200, 4201, n) / fs
fig, axs = plt.subplots(2, 2, figsize=(10, 10))
plt.subplots_adjust(wspace=0.2, hspace=0.2)

# 绘制 x1(t)
plt.subplot(221)
plt.plot(t1, x1)
plt.grid()
plt.title('x1(t)')
plt.subplot(222)
plt.plot(t2, x2)
plt.grid()
plt.title('x2(t)')
plt.subplot(212)
plt.plot(tt, y1)
plt.grid()
plt.title('conv(x1, x2)')

# 显示所有图像
plt.show()
```

图 3

任务 B：

```python
# 导入所需的库
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sg

# 创建第一个信号 x[n]
n1 = np.linspace(0, 8, 9) # 生成一个0到8的9个整数序列，代表离散时间点
x1 = [3, 2, 1, -2, 1, 0, 4, 0, 3] # 定义信号 x[n] 的值

fig, axs = plt.subplots(2, 2, figsize=(10, 10))

# 绘制第一个信号 x[n]
plt.subplot(221)
plt.stem(n1, x1, '-', use_line_collection=True)
plt.grid(True)
plt.title('x[n]')

# 创建第二个信号 h[n]
n2 = np.linspace(0, 6, 7)
x2 = [1, -2, 3, -4, 3, 2, 1]
plt.subplot(222)
plt.stem(n2, x2, '-', use_line_collection=True)
plt.grid(True)
plt.xticks(np.arange(0, 7, step=1.0))
plt.title('h[n]')

# 计算两个信号的卷积
plt.subplot(212)
y = sg.convolve(x1, x2, 'full') x1 和 x2 的卷积
n3 = np.linspace(0, 14, 15)
plt.stem(n3, y, '-', use_line_collection=True)
plt.grid(True)
plt.title('Conv Sum y[n]')

# 设置子图之间的间距和整体画布的布局
plt.xlabel('Time index n')
plt.subplots_adjust(top=1, wspace=0.2, hspace=0.2)
plt.show()
```

图 4

三、 实验结果及分析

1. 实验结果展示示例

前序任务（验证程序实例）：

样例 A:

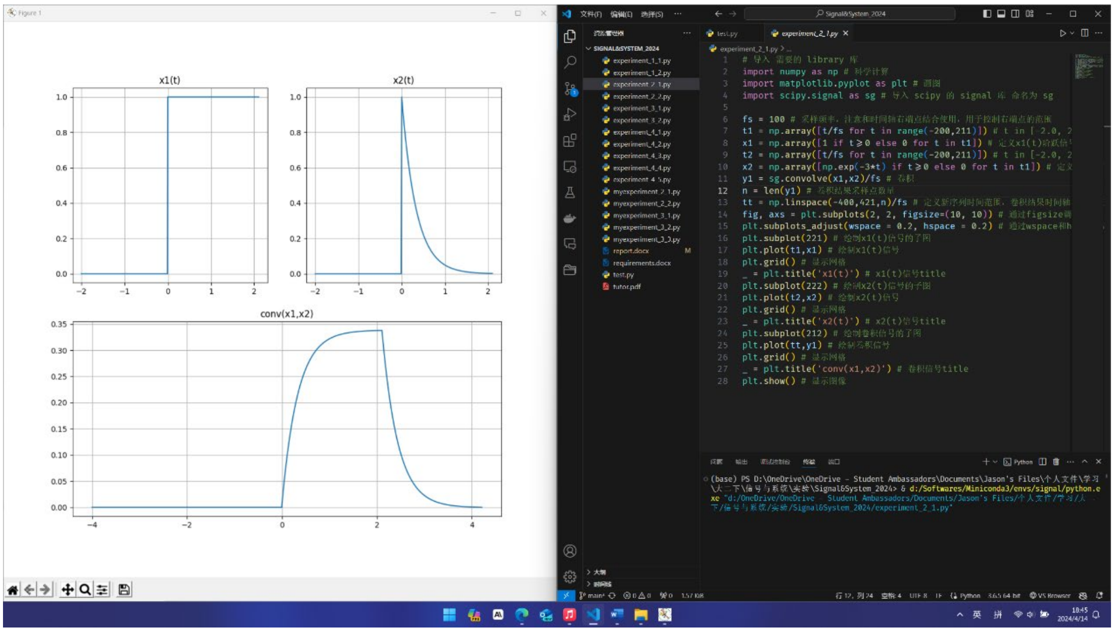

图 5

样例 B:

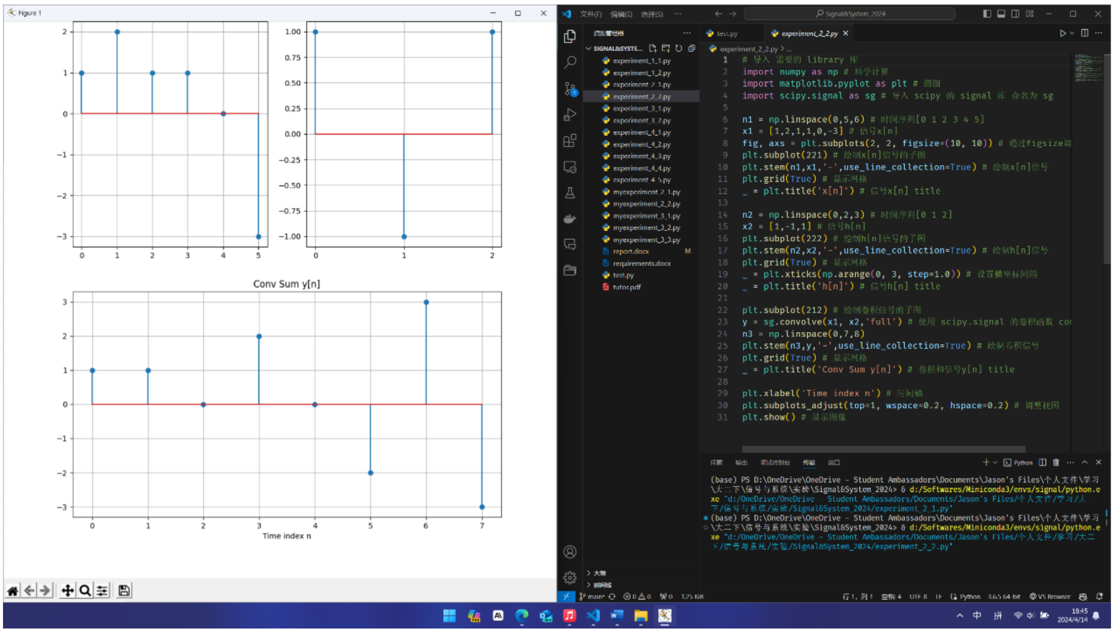

图 6

如图 5 和图 6 所示，程序正确显示了连续时间信号和离散时间信号及其卷积的波形图，和示例的图片一致。

任务 A:

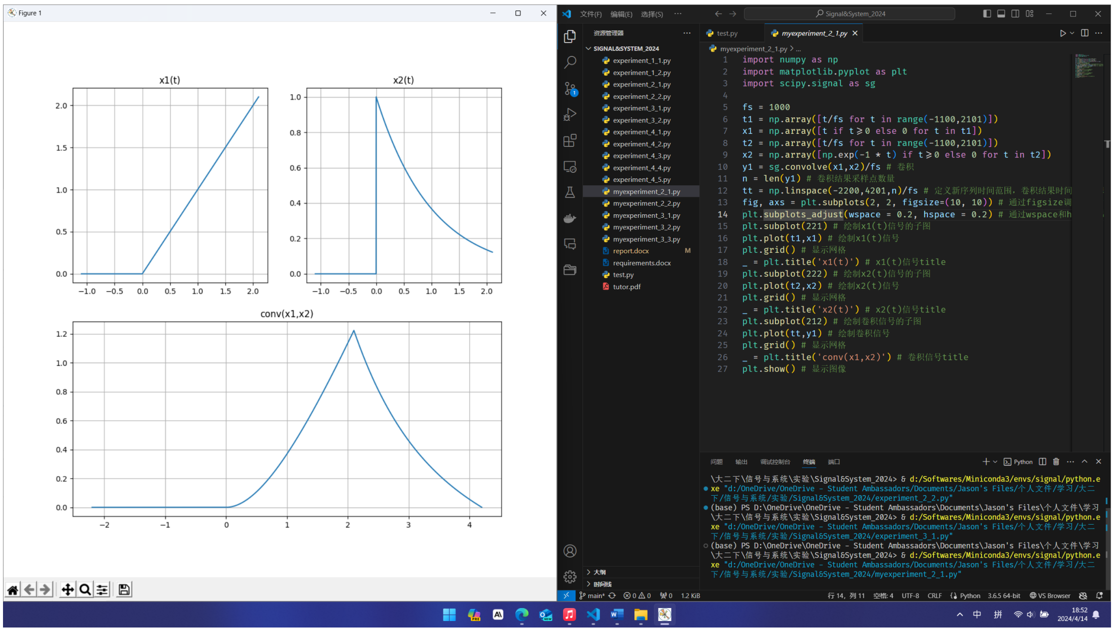

图 7

任务 B:

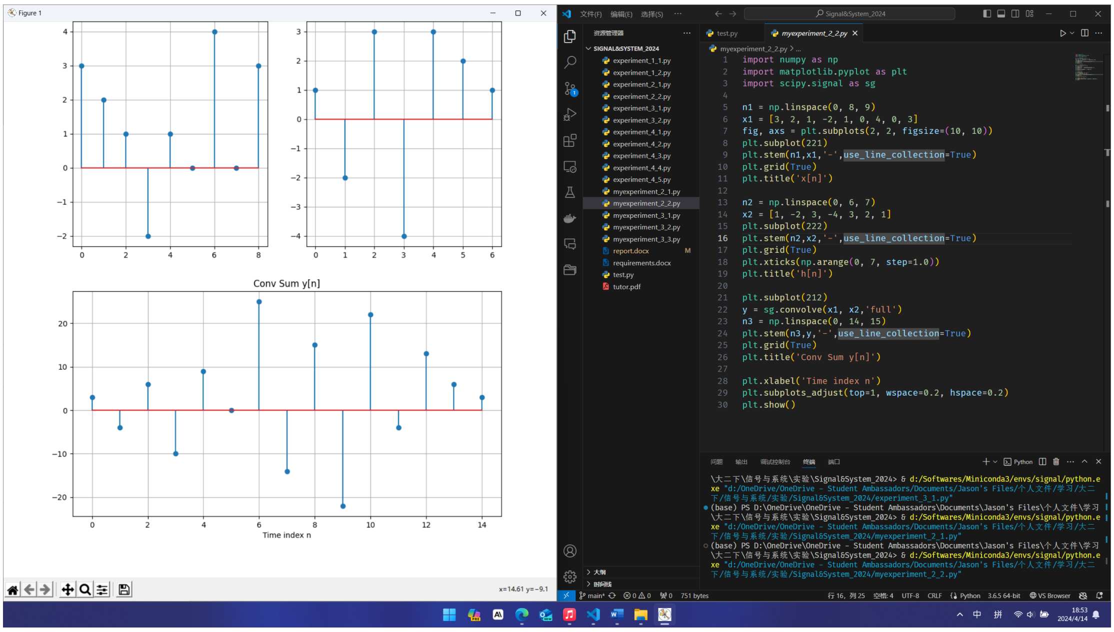

图 8

如图 7 和图 8 所示，程序正确显示了连续时间信号和离散时间信号及其卷积的波形图。

四、 思考题

1. 连续时间与离散时间信号的卷积定义分别是什么？卷积的作用是什么？conv 函数只输出了卷积的结果，没有输出对于的时间向量，如何使得时间向量和卷积的结果对应起来？

（1）

对于连续信号 $x_1(t)$ 和 $x_2(t)$，它们的卷积 $y(t)$ 定义为：

$$y(t) = (x_1 * x_2)(t) = \int_{-\infty}^{+\infty} x_1(\tau) \cdot x_2(t-\tau) d\tau$$

对于离散信号 $x_1[n]$ 和 $x_2[n]$，它们的卷积 $y[n]$ 定义为：

$$y[n] = (x_1 * x_2)[n] = \sum_{k=-\infty}^{+\infty} x_1[k] \cdot x_2[n-k]$$

（2）卷积的作用：卷积在信号处理中有很多应用，包括但不限于：滤波器设计、系统响应计算、图像处理等。在系统分析中，如果知道系统的冲激响应，那么可以通过与输入信号卷积来计算系统的输出。

（3）对于$scipy.signal.convolve$ 函数，它只返回卷积的结果，不返回对应的时间向量。但是，我们可以根据输入信号的长度来计算出输出的时间向量。如果输入信号的长度分别为N和M，那么输出信号的长度将为$N+M-1$。

2. 两个离散时间信号进行卷积和所得新序列的时域区间与原来的两个序列具有什么关系？

两个离散时间信号进行卷积后，所得新序列的时域区间（长度）是原来两个序列时域区间（长度）之和减一。

具体来说，如果原来的两个离散时间序列的长度分别为N和M，那么它们的卷积结果的长度将为$N+M-1$。

五、 参考资料

[1] NumPy 开发团队. (n.d.). NumPy 官方参考文献. 取自 https://www.numpy.org.cn/reference/. Accessed April 10, 2024.
[2] SciPy 开发团队. (n.d.). SciPy 官方参考文档. 取自 https://docs.scipy.org/doc/scipy/index.html. Accessed April 10, 2024.
[3] Matplotlib 开发团队. (n.d.). Matplotlib 官方参考文档. 取自 https://www.matplotlib.net/stable/index.html. Accessed April 10, 2024.