实验一 序列计算机

姓名：丁彦添

学号：1911406

日期：2021.12.02

一、实验平台

实验开发环境 Anaconda

Python 环境配置相关选项

```
name: dsp
channels:
  - defaults
dependencies:
  - jupyter=1.0.0
  - python=3.7.3
  - psutil=5.6.1
```

Python 导入的包

```
import copy
```

二、实验目的

完成序列计算机 v1.0 版本，要求

1. 定义一个信号序列
    - 该序列长度有限
    - 该序列起始位置可以不为0
    - 能够读取、写入该序列任意位置
2. 单序列基本操作
    - 满足前、后补零操作
    - 满足序列延迟、提前操作
    - 满足序列反转操作
    - 满足序列拉伸、压缩操作（上采样、下采样）
    - 满足序列差分、累加操作
3. 多序列操作
    - 满足加法操作
    - 满足乘法操作
    - 满足卷积操作
        - 线性卷积
        - 圆周卷积
    - 满足序列相似性比对操作
        - 滑动窗的相似性比对
        - 归一化的相似性比对

三、核心公式/问题

本次实验仅实现数字序列的基本操作，没有特殊的公式。

卷积公式$x[n] * y[n] = \sum_{k=-\infty}^{\infty} x[k] \times h[n-k]$

相似性比对公式$r_{xy}[l] = \sum_{-\infty}^{\infty} x[n] \times y[n-l]$

四、实验设计

类似于命令行控制的方式。

代码大致思路

- 先初始化两条有限的数字序列表示两个离散的数字信号。输入起始下标、并输入初始的值。
- 显示命令菜单，一个数字对应一个操作，操作包含：补零；延迟/提前；反转；拉伸/压缩；一阶差分/累加；序列间加法；序列间乘法；卷积；相似性比对
- 如果是单序列操作，需要输入对哪一条序列进行操作。
- 输出操作后得到的结果。

五、代码及结果展示和分析

本次实验基本没有调库，全部操作均由手工实现。

核心代码在于卷积运算和相似性比对运算。均严格按照对应的公式进行计算。注意零均值和归一化操作，方法还有其它的，但此处仅采用上课的PPT上的归一方法

$$K = \sqrt{\sum A_i^2},\ A_i{}' = \frac{A_i}{K}$$

$$\sum (A'_i)^2 = \sum \frac{(A_i)^2}{K^2} = \frac{1}{K^2} \sum A_i^2$$

$$= \frac{1}{\sqrt{(\sum A_i^2)}^2} \sum A_i^2 = 1$$

核心代码

```
# 卷积
def convolution(self, kern, mode='L'):
    kern.reverse()# 反转
    l = len(kern)
    if mode == 'L':# 线性卷积
        self.extension(self.start-(l-1), self.tail+(l-1))
        li = copy.deepcopy(self.seq)
        for i in range(self.start, self.tail-(l-1)+1):
            sum = 0
            for j in range(l):
                sum += kern[j] * self[i+j]
            print(sum)
            li[i-self.start+l-1]=sum
        self.start += (l-1)
        self.seq = copy.deepcopy(li[1:])
    elif mode == 'C':# 圆周卷积
        self.seq = self.seq[-l+1:]+self.seq+self[:self.start+l-1]
        self.start -= l-1
```

```
        self.tail += l-1
        li = copy.deepcopy(self.seq)
        for i in range(self.start,self.tail-(l-1)+1):
            sum = 0
            for j in range(l):
                sum += kern[j] * self[i + j]
            li[i - self.start + l - 1] = sum
        self.start+=l-1
        self.seq = copy.deepcopy(li[1:])
    return self.seq
# 相似性比对
def comp(self, k, mode='N'):
    li = []
    if mode == 'W':
        sum = 0
        if k == '1':
            for i in range(-4,5):
                x2.extension(x1.start-i, x1.tail+i)
                l=len(x2.seq)
                sum = 0
                for j in range(self.start, self.tail + 1):
                    sum += x2[j-i] * self[j]
                li.append(sum)
        if k == '2':
            for i in range(-4,5):
                x1.extension(x2.start-i, x2.tail+i)
                l=len(x1.seq)
                sum = 0
                for j in range(self.start, self.tail + 1):
                    sum += x1[j-i] * self[j]
                li.append(sum)
    elif mode == 'N':
        sum = 0
        if k == '1':
            for i in range(-4, 5):
                x1.norm()
                x2.norm()
                x2.extension(x1.start - i, x1.tail + i)
                l = len(x2.seq)
                sum = 0
                for j in range(self.start, self.tail + 1):
                    sum += x2[j - i] * self[j]
                li.append(sum)
        if k == '2':
            for i in range(-4, 5):
                x1.norm()
                x2.norm()
                x1.extension(x2.start - i, x2.tail + i)
                l = len(x1.seq)
                sum = 0
                for j in range(self.start, self.tail + 1):
                    sum += x1[j - i] * self[j]
                li.append(sum)
    return li
# 零均值与归一化
def norm(self):
    mean=0
    for i in range(len(self.seq)):
```

```
            mean += self.seq[i]
        mean /= len(self.seq)
        for i in range(len(self.seq)):
            self.seq[i] -= mean  # 零均值
        mean=0
        for i in range(len(self.seq)):
            mean += (self.seq[i] * self.seq[i])  # 归一化
        import math
        mean = math.sqrt(mean)
        for i in range(len(self.seq)):
            self.seq[i] = self.seq[i] / mean
```

结果展示和分析

运行平台是 Anaconda 环境下的 jupyter notebook。

初始化两个序列

```
--------序列计算机v1.0! --------
>> 说明：本计算机对原始序列x进行变换，但操作需要时也可输入新的序列
>> 准备就绪，输入'q'退出使用
>> 初始化序列x1，请输入起止下标（以空格分隔）：
2 6
>> 请输入序列内容(以空格分隔)：
1 2 3 4 5
>> 初始化序列x2，请输入起止下标（以空格分隔）：
-1 4
>> 请输入序列内容(以空格分隔)：
3 2 4 1 2 6
```

补零操作

```
>> 输入以下数字之一选择操作：
        0：查看某序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；5：一阶差分/累
加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对
1
>> 输入待补零序列编号（1或2）：1
>> 将x1相对于序列x2补零，结果为：
[0, 0, 0, 1, 2, 3, 4, 5]
>> 输入以下数字之一选择操作：
        0：初始化序列以及查看该序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；
5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对；q：退出使用
1
>> 输入待补零序列编号（1或2）：2
>> 将x2相对于序列x1补零，结果为：
[3, 2, 4, 1, 2, 6, 0, 0]
```

延迟和提前

```
>> 输入以下数字之一选择操作：
        0：初始化序列以及查看该序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；
5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对；q：退出使用
2
>> 输入序列编号（1或2）：1
>> 输入延迟时长（正数时延迟）：3
[0, 0, 0, 1, 2, 3, 4, 5]
index: start=2, end=9
>> 输入以下数字之一选择操作：
        0：初始化序列以及查看该序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；
5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对；q：退出使用
2
>> 输入序列编号（1或2）：1
>> 输入延迟时长（正数时延迟）：-3
[0, 0, 0, 1, 2, 3, 4, 5]
index: start=-1, end=6
```

反转

```
>> 输入以下数字之一选择操作：
        0：初始化序列以及查看该序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；
5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对；q：退出使用
3
>> 输入序列编号（1或2）：1
[5, 4, 3, 2, 1, 0, 0, 0]
index: start=-1, end=6
>> 输入以下数字之一选择操作：
        0：初始化序列以及查看该序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；
5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对；q：退出使用
3
>> 输入序列编号（1或2）：1
[0, 0, 0, 1, 2, 3, 4, 5]
index: start=-1, end=6
```

上采样

```
>> 初始化序列x1，请输入起止下标（以空格分隔）：
2 6
>> 请输入序列内容(以空格分隔)：
1 2 3 4 5 6
>> 初始化序列x2，请输入起止下标（以空格分隔）：
2 6
>> 请输入序列内容(以空格分隔)：
1 2 3 4 5 6
>> 输入以下数字之一选择操作：
        0：查看某序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；5：一阶差分/累
加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对
4
>> 输入序列编号（1或2）：1
>> 输入采样间距：2
>> 输入'u'/'d'选择上/下采样：u
[0, 0, 1, 0, 2]
index: start=2, end=6
```

下采样

```
>> 输入以下数字之一选择操作：
        0：初始化序列以及查看该序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对；q：退出使用
4
>> 输入序列编号（1或2）：2
>> 输入采样间距：2
>> 输入'u'/'d'选择上/下采样：d
[3, 5]
index: start=2, end=6
```

一阶差分

```
>> 初始化序列x1，请输入起止下标（以空格分隔）：
2 6
>> 请输入序列内容(以空格分隔)：
1 2 3 4 5
>> 初始化序列x2，请输入起止下标（以空格分隔）：
2 6
>> 请输入序列内容(以空格分隔)：
1 2 3 4 5
>> 输入以下数字之一选择操作：
        0：查看某序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对
5
>> 输入序列编号（1或2）：1
>> 输入'd'/'a'选择差分/累加：d
[1, 1, 1, 1]
index: start=2, end=6
```

一阶累加

```
>> 初始化序列x2，请输入起止下标（以空格分隔）：
2 6
>> 请输入序列内容(以空格分隔)：
1 2 3 4 5
>> 输入以下数字之一选择操作：
        0：查看某序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对
5
>> 输入序列编号（1或2）：2
>> 输入'd'/'a'选择差分/累加：a
[1, 3, 6, 10, 15]
index: start=2, end=6
```

序列间加法

```
        0：初始化序列以及查看该序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；
5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对；q：退出使用
2
>> 输入序列编号（1或2）：1
>> 输入延迟时长（正数时延迟）：0
[0, 0, 0, 1, 2, 3, 4, 5]
index: start=-1, end=6
>> 输入以下数字之一选择操作：
        0：初始化序列以及查看该序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；
5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对；q：退出使用
2
>> 输入序列编号（1或2）：2
>> 输入延迟时长（正数时延迟）：0
[3, 2, 4, 1, 2, 6, 0, 0]
index: start=-1, end=6
>> 输入以下数字之一选择操作：
        0：初始化序列以及查看该序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；
5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对；q：退出使用
6
[3, 2, 4, 2, 4, 9, 4, 5]
index: start=-1, end=6
```

序列间乘法

```
>> 初始化序列x1，请输入起止下标（以空格分隔）：
-1 6
>> 请输入序列内容(以空格分隔)：
0 0 0 1 2 3 4 5
>> 初始化序列x2，请输入起止下标（以空格分隔）：
-1 6
>> 请输入序列内容(以空格分隔)：
3 2 4 1 2 6 0 0
>> 输入以下数字之一选择操作：
        0：查看某序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；5：一阶差分
/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对
7
[0, 0, 0, 1, 4, 18, 0, 0]
index: start=-1, end=6
```

卷积

线性卷积

```
        0：查看某序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似
性比对
8
>> 初始化卷积核
>> 输入卷积核内容(以空格分隔)：
1 2 3
[1, 2, 3]
>> 输入序列编号（1或2）：2
>> 输入'L'/'C'选择线性/循环卷积：L
3
8
17
15
16
13
18
18
[0, 3, 8, 17, 15, 16, 13, 18, 18]
index: start=-1, end=6
```

循环卷积

```
>> 初始化序列x2，请输入起止下标（以空格分隔）：
2 6
>> 请输入序列内容(以空格分隔)：
1 2 3 4 5 6
>> 输入以下数字之一选择操作：
        0：查看某序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对
8
>> 初始化卷积核
>> 输入卷积核内容(以空格分隔)：
1 2 3
[1, 2, 3]
>> 输入序列编号（1或2）：2
>> 输入'L'/'C'选择线性/循环卷积：C
[6, 28, 22, 10, 16, 22, 28, 28, 2, 3, 4]
index: start=2, end=8
```

相似性比对

滑动窗口

```
>> 输入序列编号（1或2）：1
>> 输入延迟时长（正数时延迟）：0
[1, 2, 3, 4, 5]
index: start=2, end=6
>> 输入以下数字之一选择操作：
        0：初始化序列以及查看该序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对；q：退出使用
9
>> 输入序列编号（1或2）：1
>> 输入'W'/'N'选择滑动窗/归一化：W
[0, 0, 6, 14, 23, 36, 51, 33, 33]
index: start=2, end=6
```

归一化

```
>> 初始化序列x2，请输入起止下标（以空格分隔）：
2 6
>> 请输入序列内容(以空格分隔)：
1 2 3 4 5
>> 输入以下数字之一选择操作：
        0：查看某序列内容；1：补零；2：延迟/提前；3：反转；4：拉伸/压缩；5：一阶差分/累加；6：序列间加法；7：序列间乘法；8：卷积；9：相似性比对
9
>> 输入序列编号（1或2）：2
>> 输入'W'/'N'选择滑动窗/归一化：N
[-0.4, -0.4, -0.1, 0.4, 1.0, 0.4, -0.1, -0.4, -0.4]
index: start=2, end=6
```

经检验，计算结果无误。