

**宁波理工学院**

**信号大作业**



题 目 卷积实现过程

姓 名 毛佳雁，祝立群，李易潼

学 号3170406034，3170406036，3170406037

专业班级 电信172

指导教师 王一刚

学 院 信息科学与工程学院

完成日期 2018年12月25日

摘 要

信号的卷积运算是信号处理领域中最重要的运算之一。随着对信号与系统理论研究的深入,特别是计算机技术的不断发展,使卷积方法在很多领域得到了很广泛的应用，但是通过直接计算卷积的运算量非常大，它与序列长度的平方成反比，因此制约了卷积的运用。然而卷积定理指出：傅里叶变换可以化复杂的卷积运算为简单的乘积运算，从而提供了计算卷积的一种简单手段。这不仅仅减少了运算量，而且节约了时间。通过分析我们可以在Python中编程实现。

**作业展示网址：**

<https://github.com/DEisreal/DEisreal.github.io/tree/master/VSS%20CODE/5_卷积实现过程_3170406037_李易潼_3170406034_毛佳雁_3170406036_祝立群>

**关键词：**卷积；傅里叶变换；Python。

目 录

[1.前言 3](#_Toc535590290)

[1.1背景与意义 3](#_Toc535590291)

[1.2基本原理 4](#_Toc535590292)

[1.3成员组成和任务分工 7](#_Toc535590293)

[2.选题说明 8](#_Toc535590294)

[2.1 选题任务描述 8](#_Toc535590295)

[2.2 设计思路描述 8](#_Toc535590296)

[2.3 程序运行进度 10](#_Toc535590297)

[2.4最终定稿： 12](#_Toc535590298)

[3.进度安排 17](#_Toc535590299)

[4.总结与设计感悟 17](#_Toc535590300)

[参考文献 19](#_Toc535590301)

# 1.前言

## 1.1背景与意义

背景：

卷积是在信号与线性系统的基础上或背景中出现的，脱离这个背景单独谈卷积是没有任何意义的，除了那个所谓褶反公式上的数学意义和积分（或求和，离散情况下）。

信号与线性系统，讨论的就是信号经过一个线性系统以后发生的变化（就是输入输出和所经过的所谓系统，这三者之间的数学关系）。所谓线性系统的含义，就是，这个所谓的系统，带来的输出信号与输入信号的数学关系式之间是线性的运算关系。

因此，实际上，都是要根据我们需要待处理的信号形式，来设计所谓的系统传递函数，那么这个系统的传递函数和输入信号，在数学上的形式就是所谓的卷积关系。

卷积关系最重要的一种情况，就是在信号与线性系统或数字信号处理中的卷积定理。利用该定理，可以将时间域或空间域中的卷积运算等价为频率域的相乘运算，从而利用FFT等快速算法，实现有效的计算，节省运算代价。

意义：

卷积在工程和数学上都有很多应用：

统计学中，加权的滑动平均是一种卷积。

概率论中，两个统计独立变量X与Y的和的概率密度函数是X与Y的概率密度函数的卷积。

声学中，回声可以用源声与一个反映各种反射效应的函数的卷积表示。

电子工程与信号处理中，任一个线性系统的输出都可以通过将输入信号与系统函数（系统的冲激响应）做卷积获得。

物理学中，任何一个线性系统（符合叠加原理）都存在卷积。

## 1.2基本原理

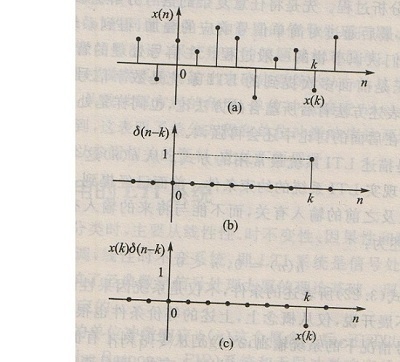
**连续时间LTI系统的卷积；卷积积分的图示法和性质；Python的基础编程。**

函数卷积的[傅里叶变换](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=412184&ss_c=ssc.citiao.link)是函数傅里叶变换的乘积。即，一个域中的卷积相当于另一个域中的乘积，例如时域中的卷积就对应于频域中的乘积。

单位冲激信号仅在n=0时，取得值1，其他位置皆为0。是一类最简单的信号，我们可以将任意一个复杂的信号分解为多个类似这样简单的信号。例如，对于一个任意的离散信号x(n)，如下图(a)所示，当n=k时，其取值为x(k)，用单位冲激信号来表示的话，可写为：

==

式中，表示单位冲激信号的延时，如图b。这样，将s(n)与相乘之后，所得到的的信号除了在n=k处取值为x(k)而不为零外，其他各点均为零。



如果重复对x(n)和相乘，其中m是另一个延时(m≠k)，则所得到的的信号仅在n=m时不为零，其值为x(m)，而在其余各处均为零。这说明，信号x(n)与单位冲激信号的某个延时相乘，实际上就是将信号x(n)在n=m处的单个值x(m)挑选出来。因此，如果在所有可能的延时处，即−∞<m<∞，都重复这样的动作，然后把得到的结果相加，即可得到x(n)的另外一种表达式：

这样，就将任意的一个信号分解为多个冲激信号的叠加。

单位冲激响应是输入信号为单位冲激信号时所对应的系统输出，常用来表示。对于线程时不变系统，如果知道了单位冲激信号的输出，根据叠加定理，就可知道任意复杂信号的输出。可得：

上式就是即为线性卷积，通常称为卷积，可以简写为：

对于连续信号，同理，任意信号可用冲激信号的组合表示，由冲激响应可得：

卷积方法的原理就是将信号分解为冲激信号之和。

连续卷积定义:f(x)与g(x)的卷积是f(t-x)g(x)在t从负无穷到正无穷的积分值。t-x要在f(x)定义域内，所以看上去很大的积分实际上还是在一定范围的。 实际的过程就是f(x)

先做一个Y轴的反转，然后再沿X轴平移t就是f(t-x)，然后再把g(x)拿来，两者乘积的值再积分。想象一下如果g(x)或者f(x)是个单位的阶越函数。 那么就是f(t-x)与g(x)相交部分的面积。这就是卷积了。

离散卷积定义：把积分符号换成求和。那么在图像中卷积是什么意思呢，就是图像f(x)，模板是g(x)，然后将模版g(x)在模版中移动，每到一个位置，就把f(x)与g(x)的定义域相交的元素卷积定义上是线性系统分析经常用到的。线性系统就是一个系统的输入和输出的关系是线性关系。就是说整个系统可以分解成N多的无关独立变化，整个系统就是这些变化的累加。如：

x1->y1，x2->y2;那么A\*x1+B\*x2->A\*y1+B\*y2这就是线性系统。表示一个线性系统可以用积分的形式，就是f(t,x)表示的是AB之类的线性系数。看上去很像卷积呀，对，如果f(t,x)=F(t-x)不就是了吗。从f(t,x)变成F(t-x)实际上是说明f(t,x)是个线性移不变，就是说变量的差不变化的时候， 那么函数的值不变化。 实际上说明一个事情，就是说线性移不变系统的输出，可以通过输入和表示系统线性特征的函数卷积得到。

## 1.3成员组成和任务分工

|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | **角色** | **任务分工** | **说明** |
| 李易潼 | 组长 | （1）总负责，分组大作业总体推进。  （2）负责大作业选题和老师沟通等。  （3）大作业设计。  （4）程序编写。 | 答辩参与人 |
| 毛佳雁 | 组员 | （1）参与大作业选题。  （2）大作业设计。  （3）程序编写。  （4）书写大作业实现报告。 | 主要答辩汇报人 |
| 祝立群 | 组员 | （1）参与大作业选题。  （2）大作业设计。  （3）程序编写。  （4）书写大作业实现报告。 | 答辩参与人 |

# 2.选题说明

## 2.1 选题任务描述

初始条件：

1. python3.7
2. 大作业设计辅助资料：python语言基础及使用入门、python基础教程、python在电子信息课程中的应用等
3. 修读课程：信号与系统

本大作业需要实现一个卷积实现动画，其要求如下：

包含连续卷积和离散卷积（生成图片）

用动画模式说明卷积实现过程

用多种方式说明卷积过程（普通函数计算；通过傅里叶变换）

PYTHON语言实现，设计对应GUI；

## 2.2 设计思路描述

（1）采用面向对象的程序设计方法。使用Python语言实现。

（2）如何生成matplotlib动画

首先新建了图片、坐标和一条空白的线作为全局变量。然后init方法是一个初始化的方法，什么都不干。animate方法中的参数i表示当前帧数，通过函数接受i生成了坐标集合，并且更新到线条中去。接下来新建了anim对象，几个参数的名称都很好懂，最后一个blit方法是告诉matplotlib记得在每帧之前擦除init方法返回的那些图元：

# call the animator. blit=true means only re-draw the parts that have changed.

anim1 = animation.FuncAnimation(fig, animate, init\_func=init, frames=30, interval=1000, blit=True)

（3）建立一个fig对象，和一个 axis1对象，并获取ax的图例：

fig, ax = plt.subplots()

ax = plt.gca()

（4）隐藏坐标轴的上边和右边：

ax.spines['right'].set\_color('none')

ax.spines['top'].set\_color('none')

（5）创建def函数animate，画出移动的矩形窗，并用if语句控制可变化的卷积和的形状：

def animate(i):

x2 = np.linspace(-1 + i, 1 + i, 1000)

y2 = x1/x1 #控制卷积函数

y2[0] = 0

y2[-1] = 0

line.set\_data(x2,y2)

if i == 0:

···

if i>=-3 and i < 1:

···

elif i >= 1:

···

return line, line1,

return line,

（6）通过ImageMagick导出GIF

1.需要安装PythonMagick-0.9.19和ImageMagick-6.2.7-6-Q16-windows-dll

2.通过此条寻找本地安装的ImageMagick

plt.rcParams['animation.convert\_path'] = 'D:\RouseWeiser\ImageMagick-6.2.7-Q16\convert.exe

3.保存gif

anim1.save('exxxo.gif', fps=2, writer='imagemagick')

（7）目前还做不成让系统自动算卷积，只能手动画，所以固定函数（红色）为x(t)，移动函数（黑色）h(t)，卷积和为（黄色）

## 2.3 程序运行进度

12月4日

# -\*- coding: UTF-8 -\*-

#coding=utf-8

importmatplotlib.pyplotasplt

importnumpyas np

def u(t,t0):

n=0

foriin t:

if(i<t0):

t[n]=0

n+=1

else:

t[n]=1

n+=1

return t

x0=-1

x1=np.pi

ω=np.pi

t0=1

#x(t)

x=np.linspace(x0,x1,100)

a=np.linspace(x0,x1,100)

y=np.sin(ω\*x)\*u(a,0)

plt.subplot(1,2,1)

plt.title("x(t)=sin(ωt)u(t)")

plt.plot(x,y)

#h(t)

z=np.linspace(x0,x1,100)

y1=np.sin(z)

plt.subplot(1,2,2)

plt.title("h(t)=sin(t)")

plt.plot(z,y1)

#时域的卷积=频域的乘积,所以用傅里叶变换快速实现卷积\_fft

#scipy.signal.fftconvolve

#使用快速傅里叶变换卷积函数。

#scipy.linalg.toeplitz

#可用于构造卷积运算符（Used to construct the convolution operator.）。

#polymul

#多项式乘法，可以同本函数获得相同的输出，但是还可以接受poly1d对象作为输入。

b=np.convolve(y,y1)

plt.figure("卷积")

plt.title("y(t)——Y(jw)；X(jw)\*Y(jw)——x(t)·h(t)")

plt.plot(b)

plt.show()

12月11日

# -\*- coding: UTF-8 -\*-

#coding=utf-8

importmatplotlib.pyplotasplt

importnumpyas np

from matplotlib import animation

fig, ax = plt.subplots()

ax = plt.gca()

plt.rcParams['font.sans-serif']=['SimHei']

plt.rcParams['axes.unicode\_minus']=False

ax.set(xlim=[-15, 15], ylim=[0, 6], title='卷积信号动画演示',ylabel='Y-轴', xlabel='X-轴')

ax.spines['right'].set\_color('none')

ax.spines['top'].set\_color('none')

ax.xaxis.set\_ticks\_position('bottom')

ax.spines['bottom'].set\_position(('data', 0))

ax.yaxis.set\_ticks\_position('left')

ax.spines['left'].set\_position(('data', 0))

ω=np.pi/4

x = np.linspace(-10, 0, 1000)

x1 = np.linspace(-13, -8, 1000)

x\_data, y\_data = [], []

y = 4\*np.sin(ω\*x) #控制固定函数

y[0] = 0

y[-1] = 0

ax.plot(x, y, "black")

definit():

y1 = 2\*np.sin(x1)

line.set\_data(x1,y1)

return line,

defanimate(i):

x2 = np.linspace(-13 + i, -8 + i, 1000)

y2 = x1/x1 #控制移动函数

y2[0] = 0

y2[-1] = 0

line.set\_data(x2,y2)

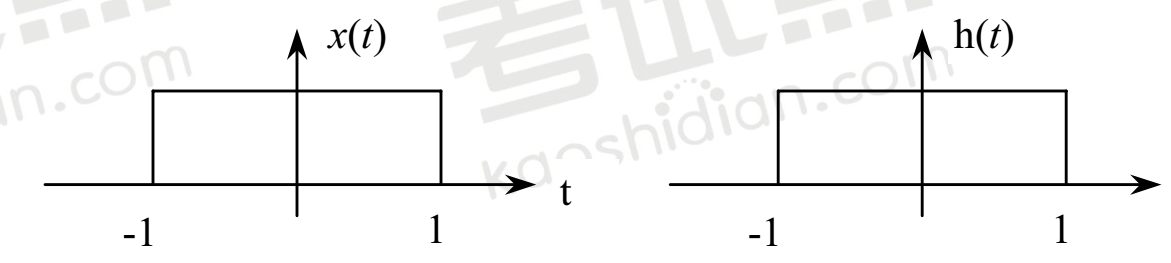
return line,

plt.show()

## 2.4最终定稿：

**原题：**

求图所示信号 x(t)与 h(t)的卷积，并用图解的方法画出 x(t)\* h(t)的波形。



解：（1）当|t|≥2时x(t)\* h(t)=0

**代码：**

# -\*- coding: UTF-8 -\*-

#coding=utf-8

import matplotlib.pyplot as plt

import numpy as np

from matplotlib import animation

plt.rcParams['animation.convert\_path'] = 'D:\RouseWeiser\ImageMagick-6.2.7-Q16\convert.exe'

fig, ax = plt.subplots()

#建立一个fig对象，建立一个axis1对象

ax = plt.gca()

#获取ax图例Convolution signal process

plt.rcParams['font.sans-serif']=['SimHei']

plt.rcParams['axes.unicode\_minus']=False

ax.set(xlim=[-5, 5], ylim=[0, 2.5], title='Envolved signal animation presentation',

ylabel='Y-axis', xlabel='X-axis')

#设置x轴y轴范围及标题

ax.spines['right'].set\_color('none')

ax.spines['top'].set\_color('none')

#隐藏轴上边和右边

ax.xaxis.set\_ticks\_position('bottom')

ax.spines['bottom'].set\_position(('data', 0))

ax.yaxis.set\_ticks\_position('left')

ax.spines['left'].set\_position(('data', 0))

#获取指定的x区间的数据

x = np.linspace(-1, 1, 1000)

x1 = np.linspace(-8, 1, 1000)

x\_data, y\_data = [], []

y = x1/x1 #控制固定函数

y[0] = 0

y[-1] = 0

ax.plot(x, y, "red")

def init():

y1 = x1/x1

line.set\_data(x1,y1)

return line,

def animate(i):

x2 = np.linspace(-4 + i, -2 + i, 1000)

y2 = x1/x1 #控制卷积函数

y2[0] = 0

y2[-1] = 0

line.set\_data(x2,y2)

if i == 0:

global x\_data, y\_data

x\_data, y\_data = [], []

line1, = ax.plot(x\_data, y\_data)

if i>=-3 and i < 1:

x\_data.append(i - 4)

y\_data.append(2+(i - 4))

line1, = ax.plot(x\_data, y\_data, "yellow")

return line, line1,

elif i >= 1:

x\_data.append(i - 1)

y\_data.append(2-(i - 1))

line1, = ax.plot(x\_data, y\_data, "yellow")

return line, line1,

return line,

y3 = x1/x1

y3[0] = 2

y3[-2] = 0

line, = ax.plot(x1, y3, "black") #控制移动函数

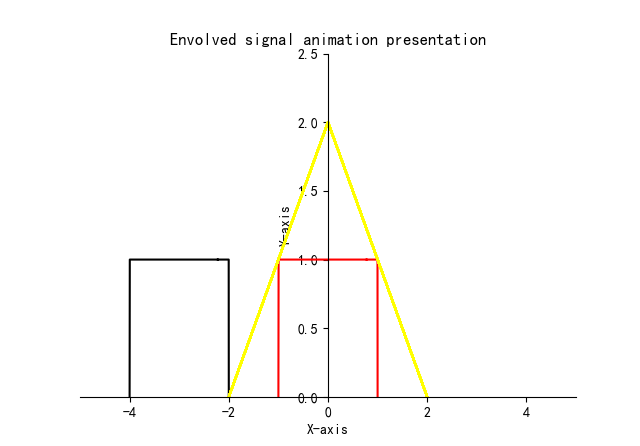
# call the animator. blit=true means only re-draw the parts that have changed.

anim1=animation.FuncAnimation(fig, animate, init\_func=init, frames=30, interval=1000, blit=True)

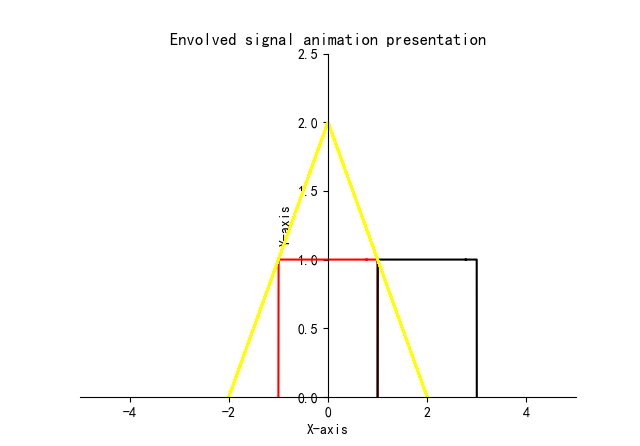
anim1.save('exxxo.gif', fps=2, writer='imagemagick')

plt.show()

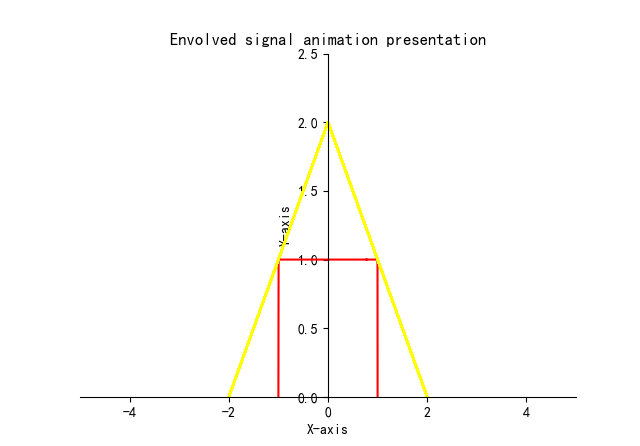
**图例：**



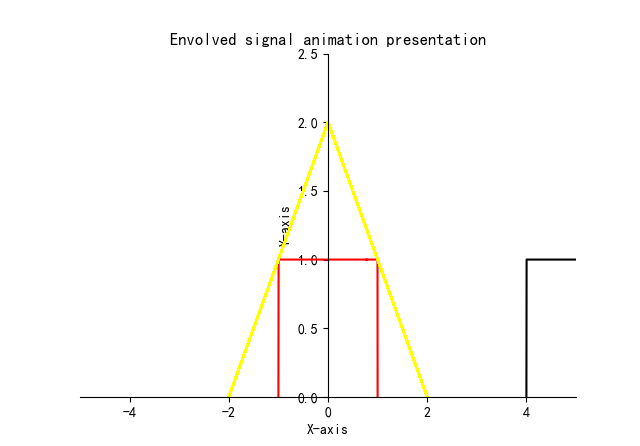
图例2-4- a



图例2-4- b



图例2-4- c



图例2-4- d

# 3.进度安排

11月3日基本GUI环境搭建

11月8日连续信号和离散信号测试

11月12日-11月18日学习python和matlab的区别

11月19日-11月26日期中考期间暂停

11月27日学习python卷积代码

11月27日-12月17日

**编写**

连续卷积和离散卷积（生成图片）

用动画模式说明卷积实现过程

用多种方式说明卷积过程（普通函数计算；通过傅里叶变换）

12月18日（或12月20日）答辩

# 4.总结与设计感悟

**总结：**

在选择了这次的大作业后，我们就开始准备资料了，先是每个人去网上查阅相关资料，做个大概的了解，接着我们去图书馆借阅相关的书籍，发现绝大部分只涉及到matlab而非python，但是很幸运在CDSN上有着相关的代码可以借鉴，在操作的过程中我们也发现了许多问题，对于整个程序也进行了许多改进，但还没此结果都不一样，不过我们也从中发现了规律，最后我们的整个大作业过程和最终结果如上。

**感悟：**

（1）本次大作业的收获很大，虽然技术有待提高，但可以说是很用心了，我们对程序编程有了新的认识，并且在编写这个程学的过程中提高了自身的水平。

（2）通过这次大作业对计算机这门学科、对Python这个软件有了许多新的认识，学习书本的知识是远远不够的，要懂得活学活用，程序的编写也是要经过数次的分析、思考和修正的，我们利用所学的知识在实践中得以运用。

（3）这次大作业是以小组的方式进行的，组内各个成员分工明确，团结一致，有条不紊地工作，锻炼了我们团结合作精神。

（4）由于我们都是初学者，自己进行探索。原定的通过FFF变换快速卷积和离散卷积都出了一定的误差，没有得到很好的结果，因此本文只放连续信号卷积动画代码。不断地深入研究，所实现的动画相对简单，有很多可以提升的地方，希望我们在未来的学习道路上可以不断进步，携手前行。

# 参考文献

1. Python编程从入门到实践. [美]埃里克·马瑟斯人民.邮电出版社.2016-07
2. 流畅的Python.[巴西] Luciano Ramalho.人民邮电出版社.2017-05-01
3. Python 基础教程

<http://www.runoob.com/python/python-tutorial.html>

1. 用Python做科学计算.作者(HYRY Studio)

<http://bigsec.net/b52/scipydoc/index.html>

1. MATLAB commands in numerical Python. Vidar Bronken Gundersen. 2