Dayananda Sagar College of Engineering

Department of Electronics and Communication Engineering

Shavige Malleshwara Hills, Kumaraswamy Layout, Bengaluru – 560 078.

(An Autonomous Institute affiliated to VTU, Approved by AICTE & ISO 9001:2008 Certified)

Accredited by National Assessment and Accreditation Council (NAAC) with 'A' grade

Assignment

Program: B.E. Branch: ECE

Course: Programming in Python Semester : 5

Course Code: 18EC5DEBPP Date:18/12/2020

A Report on

Python program for linear and circular convolution of two finite duration sequences.

Submitted by

USN:1DS19EC415 NAME: HARSHITH K C

USN:1DS19EC422 NAME: NITHIN TM

USN:1DS19EC425 NAME: RAHUL KUMAR

Faculty In-charge

Prof. Deepa N P

Signature of Faculty In-charge

OVERVIEW

WHAT IS CONVOLUTION?

Convolution is a mathematical operation that expresses a relationship between an input signal, the output signal, and the impulse response of a linear-time invariant system.

An impulse response is the response of any system when an impulse signal (a signal that contains all possible frequencies) is applied to it.

As we have seen earlier in this digital signal processing, a linear time-invariant system is a system that a) behaves linearly, and b) is time-invariant (a shift in time at the input causes a corresponding shift in time in the output).

Formula for Convolution of a continuous-time system

$$y(t) = x(t)*h(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$$

Formula for Convolution for a discrete-time system

$$y(n) = x(n)*h(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k)$$

LINEAR CONVOLUTION:

LINEAR CONVOLUTION is a mathematical operation done to calculate the output of any Linear-Time Invariant (LTI) system given its input and impulse response.

It is applicable for both continuous and discrete-time signals.

We can represent Linear Convolution as
y(n)=x(n)*h(n)

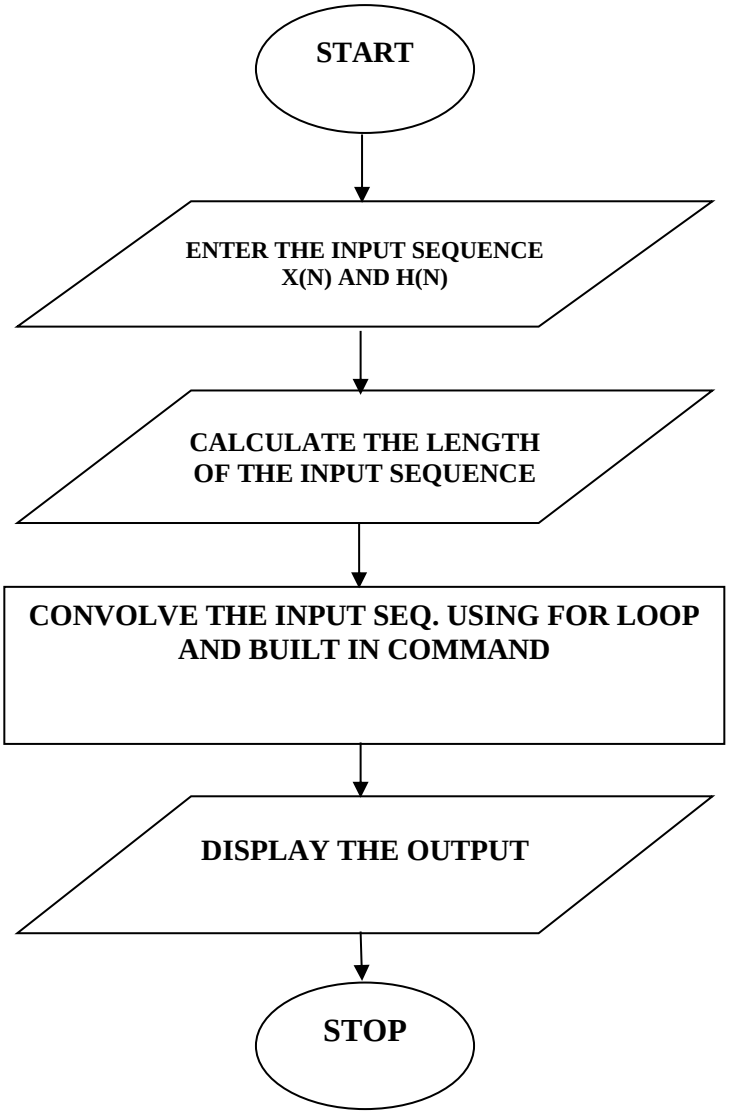
Here, y(n) is the output (also known as convolution sum). x(n) is the input signal, and h(n) is the impulse response of the LTI system.

In linear convolution, both the sequences (input and impulse response) may or may not be of equal sizes. That is, they may or may not have the same number of samples. Thus, the output, too, may or may not have the same number of samples as any of the inputs.

Graphically, when we perform linear convolution, there is a linear shift taking place. Check out the formula for a convolution.
$\sum_{-\infty}^{\infty}$x(k)h(n-k)
In python we are using this formula for convolution: y[n] = y[n]+x[n-k]*h[k]
It is possible to find the response of a filter using linear convolution. Linear convolution may or may not result in a periodic output signal.

Circular convolution, also known as **cyclic convolution**, is a special case of **periodic convolution**, which is the convolution of two periodic functions that have the same period. Periodic convolution arisesCircular Convolution is used to study the interaction of two signals that are periodic.

FLOW CHART:



ALGORITHM:

Step 1: Start
Step 2: Read the first sequence
Step 3: Read the second sequence
Step 4: calculate the length of the sequence
Step 5: Perform convolution for both the sequences.
Step 6: Display the output sequence
Step 7: Stop

PYTHON PROGRAM FOR LINEAR AND CIRCULAR CONVOLUTION:

LINEAR CONVOLUTION:

```
#Program to perform linear convolution
import numpy as np
#defining x[n] naming it as x
x=[]
x = [int(item) for item in input("Enter the list items : ").split(',')]
#defining h[n] naming it as h
h=[]
h = [int(item) for item in input("Enter the list items : ").split(',')]
print (x)
print (h)
N1 = len(x)
N2 = len(h)
N = N1+N2-1 # Total length -1
y = np.zeros(N)
#Linear convolution using built-in function in NumPy
y1 = np.convolve(x,h) #inbuilt function
m = N-N1
n = N-N2
#Padding zeros to x and h to make their length to N
x =np.pad(x,(0,m),'constant')
h =np.pad(h,(0,n),'constant')
#Linear convolution using convolution sum formula
for n in range (N):
   for k in range (N):
      if n >= k:
         y[n] = y[n]+x[n-k]*h[k]

print('Linear convolution using convolution sum formula output response y =\n',y)
print('Linear convolution using NumPy built-in function output response y=\n',y1)
```

RESULT:

Example 1:

Enter the list items : 1,2,3,4,5,6,7

Enter the list items : 1,2,3,4,5

[1, 2, 3, 4, 5, 6, 7]

[1, 2, 3, 4, 5]

Linear convolution using convolution sum formula output response y =

[1. 4. 10. 20. 35. 50. 65. 72. 70. 58. 35.]

Linear convolution using NumPy built-in function output response y=

[1 4 10 20 35 50 65 72 70 58 35]

```
Enter the list items : 1,2,3,4,5,6,7
Enter the list items : 1,2,3,4,5
[1, 2, 3, 4, 5, 6, 7]
[1, 2, 3, 4, 5]
Linear convolution using convolution sum formula output response y =
 [ 1.  4. 10. 20. 35. 50. 65. 72. 70. 58. 35.]
Linear convolution using NumPy built-in function output response y=
 [ 1  4 10 20 35 50 65 72 70 58 35]
```

Example 2 :

Enter the list items : 12,18,21,24

Enter the list items : 14,18,22,32

[12, 18, 21, 24]

[14, 18, 22, 32]

Linear convolution using convolution sum formula output response y =

[168. 468. 882. 1494. 1470. 1200. 768.]

Linear convolution using NumPy built-in function output response y=

[168 468 882 1494 1470 1200 768]

```
Enter the list items : 12,18,21,24
Enter the list items : 14,18,22,32
[12, 18, 21, 24]
[14, 18, 22, 32]
Linear convolution using convolution sum formula output response y =
 [ 168.  468.  882. 1494. 1470. 1200.  768.]
Linear convolution using NumPy built-in function output response y=
 [ 168  468  882 1494 1470 1200  768]
```

CIRCULAR CONVOLUTION:

#Program to perform circular convolution

```
import numpy as np
#defining x[n] naming it as x
x=[]
x = [int(item) for item in input("Enter the list items : ").split(',')]
#defining h[n] naming it as h
h=[]
h = [int(item) for item in input("Enter the list items : ").split(',')]
print (x)
print (h)
N1 = len(x)
N2 = len(h)
N = max (N1, N2)
y = np.zeros(N)
m = N-N1
n = N-N2
#Padding zeros to x and h to make their length to N
x =np.pad(x,(0,m),'constant')
h =np.pad(h,(0,n),'constant')

#circular convolution using convolution sum formula
for n in range (N):
   for k in range (N):
         y[n] = y[n]+x[n-k]*h[k]

print('circular convolution using convolution sum formula output response y =\n',y)
```

RESULT:

Example 1:

Enter the list items : 6,2,1,3
Enter the list items : 6,2,1,3
[6, 2, 1, 3]
[6, 2, 1, 3]
circular convolution using convolution sum formula output response y =
[49. 30. 25. 40.]

```
Enter the list items : 6,2,1,3
Enter the list items : 6,2,1,3
[6, 2, 1, 3]
[6, 2, 1, 3]
circular convolution using convolution sum formula output response y =
 [49. 30. 25. 40.]
```

Example 2 :

Enter the list items : 11,13,16

Enter the list items : 16,19,12,22,24

[11, 13, 16]

[16, 19, 12, 22, 24]

circular convolution using convolution sum formula output response y =

[840. 801. 635. 702. 742.]

```
Enter the list items : 11,13,16
Enter the list items : 16,19,12,22,24
[11, 13, 16]
[16, 19, 12, 22, 24]
circular convolution using convolution sum formula output response y =
 [840. 801. 635. 702. 742.]
```

REFERENCE:

- **GEEKSFORGEEKS : https://www.geeksforgeeks.org/circular-convolution-using-matrix-method/ , https://www.geeksforgeeks.org/introduction-to-convolutions-using-python/**
- **GITHUB : https://github.com/vrajvtec/DTSP-**
- **WIKIPIDEA : https://en.wikipedia.org/wiki/Convolution#:~:text=In%20mathematics%20(in%20particular%2C%20functional,the%20process%20of%20computing%20it.**