LABSHEET 2
1D CONVOLUTION

NumPy is a very potent library offered by Python that enables us to carry out intricate and computationally intensive computations. One such computation is called Convolution.

In general, convolution means the formation of the third function by any sort of combination of two functions. It is a very common and useful topic when it comes to signal analysis/processing. In Signal Processing, the operation of combining two signals from a third signal is called a convolution operation.

The **NumPy** library offers a function convolve(), which allows us to find the discrete and linear convolution of two one-dimensional arrays/vectors.

Syntax for NumPy convolve()

The syntax for **NumPy convolve()** function is:

```
numpy.convolve (a1, a2, mode)
```

Parameters for NumPy convolve()

The parameters that **NumPy convolve()** functions takes are:

- **a1**: This mandatory parameter represents the first one-dimensional vector.
- **a2**: This mandatory parameter represents the second one-dimensional vector.
- **mode**: This optional parameter represents the different modes for the convolution operation. These modes are **full**, **same**, **valid**.
- **full**: This returns the convolution at each point of overlap, with an output shape of (M+N-1); M and N are the dimensions of the first and second array, respectively.
- **same**: This returns the output sequence of length max(M, N)
- **valid**: This returns the output sequence is of length max(M,N) – min(M,N) + 1.

Exercises

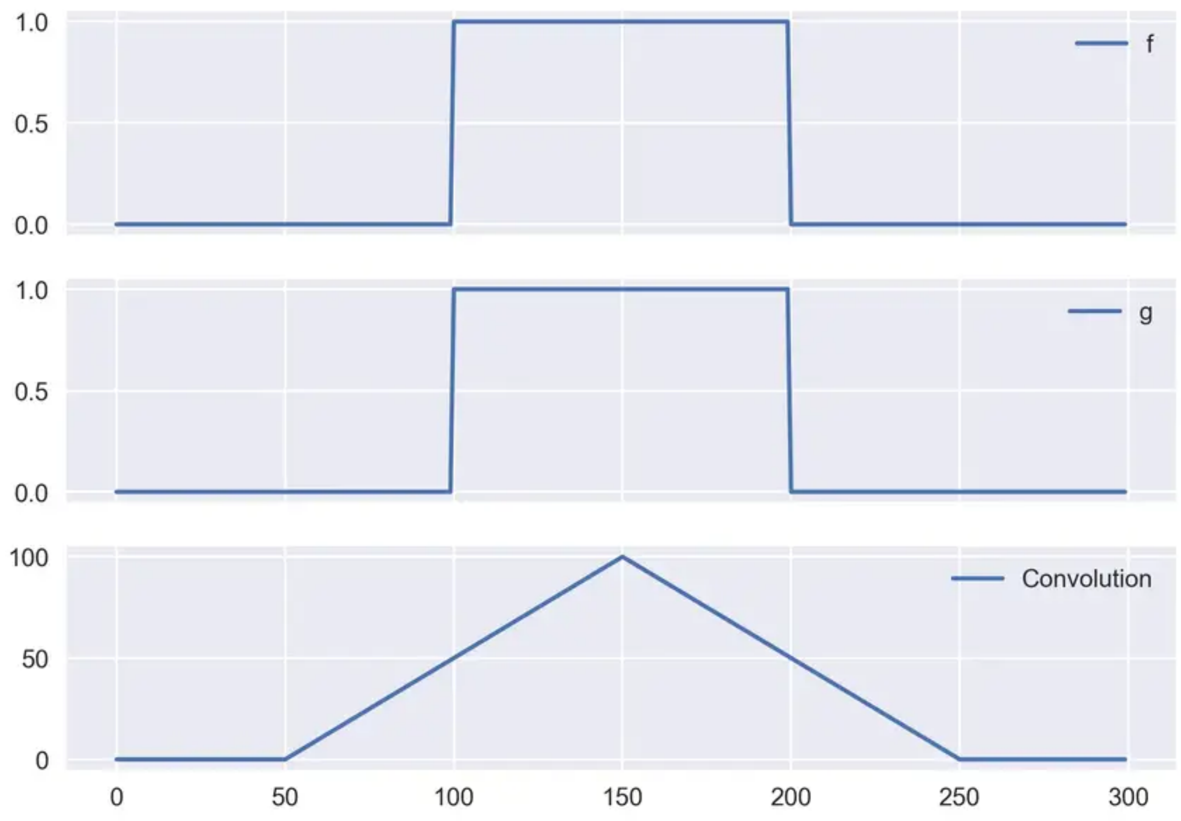
1) Perform the convolution of 1D digital signal with np.convolve function.

```
# Import NumPy
import numpy as np

# First input array
a = np.array([3, 7])
print("First vector: ", a)

# Second input array
v = np.array([1, 2, 5, 7])
print("Second vector: ", v)

```

```
•    print("Convolution using full mode:")
•    # Using convolve() function in "full" mode:
•    print(np.convolve(a, v))
```

2) Try the above example with 'same' and 'valid' modes. Write down the difference.
3) Perform 1d convolution of two square signal with same width.

```
import numpy as np
from scipy import signal
import matplotlib.pyplot as plt
plt.style.use('seaborn')

sig1 = np.repeat([0., 1., 0.], 100)
sig2 = np.repeat([0., 1., 0.], 100)

filtered = np.convolve(sig1, sig2, mode='same')

fig, ax = plt.subplots(3,1, sharex=True)
ax[0].plot(sig1, label='f')
ax[1].plot(sig2, label='g')
ax[2].plot(filtered, label = 'Convolution')

for axx in ax:
   axx.legend()

plt.savefig('convolvesigs.png',bbox_inches='tight', dpi=300)
```



4) Perform 1D convolution of two square signal where one signal width is twice as that of the first one. What is the shape of the resultant signal?