YILDIZ TEKNİK ÜNİVERSİTESİ

İsim: Yusuf Taha TÜTEN
Bölüm: Bilgisayar Mühendisliği
Numara: 21011027
BLM2041 – Bilgisayar Mühendisleri İçin Sinyal ve Sistemler
Dersin Yürütücüsü: Ali Can KARACA

Ödevin Konusu: Konvülasyon algoritması kurgulama ve kodlama, konvüle edilen sinyalleri seslendirme ve grafik ile gösterme.

```python
import numpy as np
import matplotlib.pyplot as plt
import sounddevice as sd

# Question 1
# This is my own convolution function
def convoluted(x, n, y, m):

    result = [0]*(n+m-1)

    k = 0
    for i in range(len(x)):
        tmp = k
        for j in range(len(y)):
            result[k] = result[k] + y[j]*x[i]
            k += 1
        k = tmp + 1


    return result

```

Yukarıda görüldüğü üzere konvolüsyon fonksiyonum eklemeli olarak ilerlemektedir. Önce x dizisinin ilk elemanını y dizisinin sırasıyla tüm elemanlarıyla çarparak result dizisindeki 0. indisten itibaren yerleştirmeye başlar. Bu işlemi x dizisinin sonraki elemanlarına geçtikçe result dizisinin de sonraki elemanlarına geçmesi takip eder. Örn: x[0] için result dizisine eklemeyi result[0,1,2,3…] için yaparken x[1] için result[1,2,3…] olarak yapar. Dolayısıyla algoritma ilerledikçe yapılması gereken işlem sayısı azalır. Esasen konvolüsyon işleminin doğası incelendiğinde iki taraflı bir piramit şeklinde yükselip alçaldığı görülebilir.

N elemanlı iki sinyali birbiriyle konvüle ettiğimizi düşünelim. Bir adımdaki maks çarpım sayısı N olacaktır (küçük sinyalin uzunluğu kadar) fakat ilk adımlar N çarpım sayısına ulaşana kadar birer artarak gidecektir, son adımlar ise N'den itibaren 1 azalmaya başlayarak 0'a doğru gideceklerdir.

Örnek: x = [1,2,3], y = [1,2,3]

0. Adım = x[0]*y[0]

1. Adım = x[1]*y[0], x[0]*y[1]

2. Adım = x[2]*y[0], x[1]*y[1], x[0]*y[2]

3. Adım = x[2]*y[1], x[1]*y[2]

4. Adım = x[2]*y[2]

Görüldüğü üzere konvolüsyon işlemi piramit şeklinde ilerler. Hatta daha dikkatli bakılırsa dizi indislerinin toplamı o adımdaki adım sayısını sağlar. Algoritma bunun üzerine de kurulabilirdi fakat o zaman çok daha yavaş ve nonefektiv çalışırdı.

```python
n = int(input("Enter 'n' for x(n): "))
m = int(input("Enter 'm' for y(m): "))

x = input('Enter x(n) signal: ')
x = x.split()
for i in range(len(x)):
    x[i] = int(x[i])

y = input('Enter y(m) signal: ')
y = y.split()
for i in range(len(y)):
    y[i] = int(y[i])
```

Basit, kullanıcıdan bilgi alma işlemleri.

```
print("\nHere's convoluted signal by my convolution function: ")
print(sum)

pySum = np.convolve(x,y,'full')
print("\nHere's convoluted signal by Python's built-in convolution function: ")
print(pySum)

# Graphical shown of x(n),y(m),myConv,built-in function
xInd = int(input("\nGive the index of '0' point in signal x(n): "))
yInd = int(input("Give the index of '0' point in signal y(m): "))
xHorizontal = [1-xInd+i for i in range(n)]
yHorizontal = [1-yInd+i for i in range(m)]
sumHorizontal = [2-xInd-yInd+2*i for i in range(n+m-1)]
pySumHorizontal = [2-xInd-yInd+2*i for i in range(n+m-1)]

plt.stem(xHorizontal,x)
plt.title("first signal")
plt.ylabel("x(t)")
plt.xlabel("time")
plt.show()

plt.stem(yHorizontal,y)
plt.title("second signal")
plt.ylabel("y(t)")
plt.xlabel("time")
plt.show()

plt.stem(sumHorizontal,sum)
plt.title("convoluted signal by me")
plt.xlabel("time")
plt.show()

plt.stem(pySumHorizontal,pySum)
plt.title("convoluted signal by python")
plt.xlabel("time")
plt.show()
```

Burası sinyallerin grafik ve vektörel gösterimi kısmı. Vektörel gösterim için kullanıcıdan sinyalin 0 noktasını istiyorum ve girilen değerden ‘1-0noktası’ formülüyle sinyalin başlangıç noktasını buluyorum. Konvüle edilmiş fonksiyonun başlangıç fonksiyonu için ise iki sinyalin başlangıç noktalarını topluyorum.

Grafiksel gösterim için matplotlib.pyplot yardımıyla stem türünde tablolar oluşturuyorum. ‘stem’ fonksiyonlarının ilk parametresi olarak yatay eksendeki başlangıç noktalarını alıyorum.

```
# Question 3 - Recording audio
freq = 10000
duration = 5
sd.default.samplerate = freq # This code line allows sounddevice module to play sounds with giving frequency otherwise default is 44.1kHz
choice = int(input("Press 1 for 5 sec recording and 0 for 10 sec: "))
if(choice == 1):
    print("Start speaking.(5 sec)")
    x1 = sd.rec(int(duration*freq), samplerate = freq, channels = 1) # channels might differ according to computer, if it doesn't work plea
    sd.wait()
    print("End of Recording.")
    flat = np.array(x1).flatten()
elif(choice == 0):
    print("Start speaking.(10 sec)")
    x2 = sd.rec(int(2*duration*freq), samplerate = freq, channels = 1) # channels might differ according to computer, if it doesn't work pl
    sd.wait()
    print("End of Recording.")
    flat = np.array(x2).flatten()
else:
    print("That's not valid choice.")
```

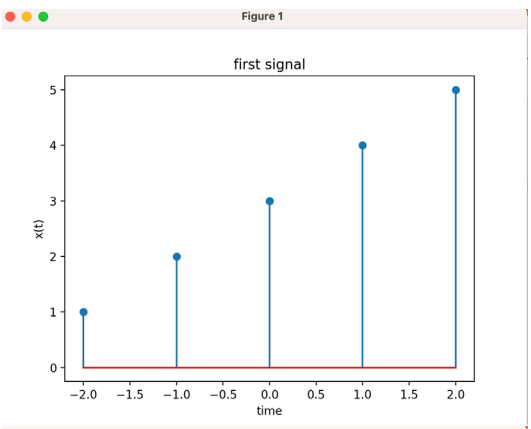
Ses kaydı için ‘sounddevice’ modülünü kullandım. Frekans değeri kolay olması bakımından 10000 olarak seçildi. Kaydedilen sesleri sonrasında konvüle edebilmek için ‘flatten’ fonksiyonu ile çok boyutlu formundan tek boyutlu dizi formuna çevirdim. Tercihe bağlı olarak 5 ya da 10 saniyelik ses kayıtları alınabiliyor.(sd.default.samplerate=freq kodu önemli yoksa ses süresinde kısalma yaşarız)
NOT: macOS’ta kodladığım için ‘channels = 1’ kullandım eğer hata veriyorsa ‘channels = 2’ deneyebilirsiniz.

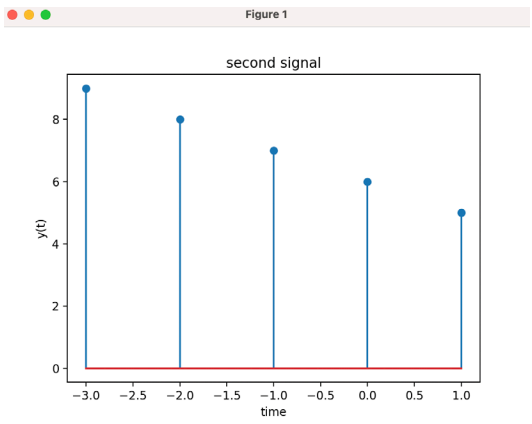
```
x(n):
[1, 2, 3, 4, 5]

y(m):
[9, 8, 7, 6, 5]

Here's convoluted signal by my convolution function:
[9, 26, 50, 80, 115, 96, 74, 50, 25]

Here's convoluted signal by Python's built-in convolution function:
[  9  26  50  80 115  96  74  50  25]
```

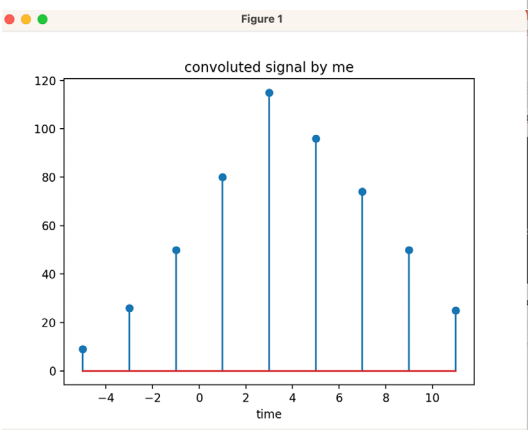
x = 5 ve y = 5 elemanlı iki dizi için yapılan vektörel gösterimi ve 0 noktalarından başlangıç noktasının hesaplanması.

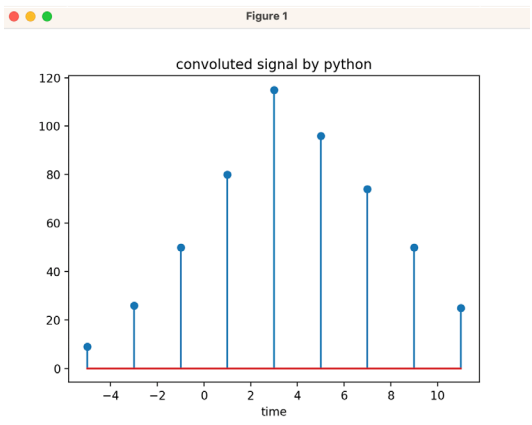
```
x(n):
[1, 2, 3, 4, 5]

y(m):
[9, 8, 7, 6, 5]

Here's convoluted signal by my convolution function:
[9, 26, 50, 80, 115, 96, 74, 50, 25]

Here's convoluted signal by Python's built-in convolution function:
[  9  26  50  80 115  96  74  50  25]

Give the index of '0' point in signal x(n): 3
Give the index of '0' point in signal y(m): 4
```

Görüldüğü üzere indisler hesaplanıp grafiklerde yerlerine konulmuştur.









```
flag = 0
while(flag == 0):
    A = 0.8
    M = int(input("Enter 'M' Value for sigma notation: "))
    h = [0]*(400*M)
    h[0] = 1
    h.append(0)
    for i in range(M):
        h[(i+1)*400] = A*(i+1)

    choice = int(input("Press 1 for my own convulation function and 0 for built-in function: "))
    if(choice == 1):
        convolved = convoluted(flat,len(flat),h,len(h))
    elif(choice == 0):
        convolved = np.convolve(flat,h)
    else:
        print("That's not valid choice.")

    sd.play(convolved)
    flag = int(input("To continue convolving press 0: "))

```

Yine tercihe bağlı olarak M değeri ve fonksiyon ile konvolüsyon işlemi gerçekleştirilebiliyor.

```
# Question 4

flag = 0
while(flag == 0):
    A = 0.8
    M = int(input("Enter 'M' Value for sigma notation: "))
    h = [0]*(400*M)
    h[0] = 1
    h.append(0)
    for i in range(M):
        h[(i+1)*400] = A*(i+1)

    choice = int(input("Press 1 for my own convulation function and 0 for built-in function: "))
    if(choice == 1):
        start = time.time()
        convolved = convoluted(flat,len(flat),h,len(h))
        stop = time.time()
    elif(choice == 0):
        start = time.time()
        convolved = np.convolve(flat,h)
        stop = time.time()
    else:
        print("That's not valid choice.")

    print("Convulation time: "f'{stop-start}')

    start = time.time()
    sd.play(convolved)
    sd.wait()
    stop = time.time()

    print("Sound time: "f'{stop-start}')

    flag = int(input("To continue convolving press 0: "))
```

Farklı bir dosya açarak fonksiyonların farklı veri setlerindeki hız farklarını ölçtüm ve bunları tablolaştırdım. (Üst kısımdaki resim ödev dosyasında değildir fakat time fonksiyonları çıkarılmış hali ödev dosyasındaki kod bloklarının birebir aynısıdır)

M	2	3	4
Python 5 sec	≈0.03sn	≈0.05sn	≈0.07sn
Python 10 sec	≈0.05sn	≈0.09sn	≈0.11sn
Own 5 sec	≈37sn	≈56sn	≈73sn
Own 10 sec	≈73sn	≈110sn	≈149.5sn

Konvüle sonucunda ses şiddetinde artma ve sesin kendisinde ekolanma oluştuğunu görüyoruz. M değeri arttıkça bu ekolanma ve şiddet artışı daha da belirgin hale geliyor. h[n] sinyalinin x[n] sinyali üzerine rastgele şiddetlerle rastgele frekans değerlerine eklenmesi sonucunda bu ekoların oluştuğunu ve seste –küçük de olsa - anlık genlik artışı ve düşüşü görülebilir.