# BLM6105-2 Uzaktan Algılama Dersi Ödev 1

# 24501100 – Aleyna Nil Uzunoğlu

## **Giriş**

Bu ödevde ayrık zamanlı lineer zamanla değişmeyen sistemlerin analizinde kullanılan konvolüsyon fonksiyonu üzerine çalışılmıştır. Çalışmanın amacı, belirli bir uzunlukta tanımlanan x[n] ve h[n] dizileri için konvolüsyon işlemini Python programlama dili üzerinde manuel olarak kodlamak ve oluşturulan bu fonksiyonun sonuçlarını NumPy kütüphanesinde bulunan convolve() fonksiyonu ile doğrulamaktır. Karşılaştırma 3 farklı dizi üzerinden gerçekleştirilmiş ve sonuçlar grafiksel olarak görselleştirilmiştir.

## **Uygulama**

Konvolüsyon işlemini gerçekleştirmek için my\_conv (x\_n, h\_n) adında bir Python fonksiyonu tanımlanmıştır. Bu fonksiyon girdi olarak x[n] ve h[n] dizilerini alır. Çıktı olarak y[n] dizisini verir. Çalıştırılması durumunda sırasıyla

* x[n], h[n] dizileri için değer aralıkları olan x\_low , x\_high, h\_low, h\_high değerlerini hesaplar.
* Bu değerler yardımıyla y[n] sonuç dizisinin değer aralığını bulur ve y\_low, y\_high değişkenlerine atar.
* Sonuç dizisi y[n]’i low ve high değerleri arasında 0’lardan oluşan bir dizi olarak ilklendirir. (np.zeros() yardımıyla)
* İç içe for döngüsü kullanarak konvolüsyon toplamı hesaplamasını gerçekleştirir.
  + Dış döngü y[n] dizisindeki her bir indis için çalışır.
  + İç döngü ise her bir x[k] sinyali için h[n-k] ile çarpım işlemini gerçekleştirir.
* y[n] dizisini döndürür.

## **Örnekler ve Sonuçlar**

Üç farklı x[n] ve h[n] çifti için konvolüsyon değerleri hesaplanmış ve sonuçlar karşılaştırılmıştır. Her bir örnek için; giriş sinyali x[n], dürtü yanıtı h[n], my\_conv() fonksiyonunun çıktısı olan y[n] ve NumPy yardımıyla hesaplanan çıkış dizisi y[n] yan yana çizdirilmiştir.

Örnek 1:

* x1 = [1, 2, 3, 4] ve h1 = [0.2, 0.5, 0.2] için grafikleri çizdiriniz.

metin, diyagram, çizgi, paralel içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

Grafik 1: x[n] ve h[n] için fonksiyon karşılaştırmaları

Örnek 2:

* x2 = [1, 2, 1, 0, -1] ve h2 = [1, -1, 2] için grafikleri çizdiriniz.

metin, diyagram, çizgi, paralel içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

Grafik 2: x[n] ve h[n] için fonksiyon karşılaştırmaları

Örnek 3:

* x3 = [1,1,1,1,1] ve h3 = [0, 1, 0.5] için grafikleri çizdiriniz.

metin, diyagram, paralel, çizgi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

Grafik 3: x[n] ve h[n] için fonksiyon karşılaştırmaları

**Sonuç**

Bu çalışmada, ayrık zamanlı konvolüsyon işlemi için manuel bir Python fonksiyonu başarıyla geliştirilmiştir. Fonksiyonun doğruluğu, farklı özelliklere sahip üç sinyal çifti üzerinde NumPy'ın gömülü convolve fonksiyonu ile karşılaştırılarak doğrulanmıştır. Manuel uygulamanın, konvolüsyonun temel mekanizmasını anlamak açısından faydalı olduğu değerlendirilmiştir. Elde edilen grafiksel sonuçlarda, fonksiyonun doğru uygulandığı görülmüştür. Ayrıca grafiksel sonuçlar, konvolüsyonun sinyal şeklini nasıl değiştirdiğini (filtreleme, yumuşatma, değişimleri vurgulama vb.) görsel olarak ortaya koymaktadır.

**Ek: main.py**

import numpy as np

import matplotlib.pyplot as plt

def find\_low\_and\_high(k\_array):

low = np.min(k\_array)

high = np.max(k\_array)

return low, high

def my\_convolution(x\_n, h\_n):

x\_low, x\_high = find\_low\_and\_high(x\_n)

h\_low, h\_high = find\_low\_and\_high(h\_n)

y\_low = x\_low + h\_low

y\_high = x\_high + h\_high

print("x[n] dizisi aralığı: ", x\_low, " ile ", x\_high)

print("h[n] dizisi aralığı: ", h\_low, " ile ", h\_high)

print("y[n] dizisi aralığı: ", y\_low, " ile ", y\_high)

len\_y = len(x\_n) + len(h\_n) - 1

y = np.zeros(len\_y)

# Convolution işlemi

for n in range(len\_y):

for k in range(len(x\_n)):

if 0 <= n - k < len(h\_n):

y[n] += x\_n[k] \* h\_n[n - k]

return y

def draw\_plot(location, input, title):

plt.subplot(location)

plt.stem(input)

plt.title(title)

if \_\_name\_\_ == "\_\_main\_\_":

# ------------------------------------------------- #

# x[n] ve h[n] dizileri

x1 = np.array([1, 2, 3, 4])

h1 = np.array([0.2, 0.5, 0.2])

y1\_manual = my\_convolution(x1, h1)

y1\_numpy = np.convolve(x1, h1, mode='full')

plt.figure(figsize=(6, 6))

draw\_plot(221, x1, 'x[n]')

draw\_plot(222, h1, 'h[n]')

draw\_plot(223, y1\_manual, 'My Convolution Function')

draw\_plot(224, y1\_numpy, 'Numpy Convolve')

plt.tight\_layout()

plt.savefig("ornek1.png")

plt.show()

# ------------------------------------------------- #

# x[n] ve h[n] dizileri

x2 = np.array([1, 2, 1, 0, -1])

h2 = np.array([1, -1, 2])

y2\_manual = my\_convolution(x2, h2)

y2\_numpy = np.convolve(x2, h2, mode='full')

plt.figure(figsize=(6, 6))

draw\_plot(221, x2, 'x[n]')

draw\_plot(222, h2, 'h[n]')

draw\_plot(223, y2\_manual, 'My Convolution Function')

draw\_plot(224, y2\_numpy, 'Numpy Convolve')

plt.tight\_layout()

plt.savefig("ornek2.png")

plt.show()

# ------------------------------------------------- #

# x[n] ve h[n] dizileri

x3 = np.array([1, 1, 1, 1, 1])

h3 = np.array([0, 1, 0.5])

y3\_manual = my\_convolution(x3, h3)

y3\_numpy = np.convolve(x3, h3, mode='full')

plt.figure(figsize=(6, 6))

draw\_plot(221, x3, 'x[n]')

draw\_plot(222, h3, 'h[n]')

draw\_plot(223, y3\_manual, 'My Convolution Function')

draw\_plot(224, y3\_numpy, 'Numpy Convolve')

plt.tight\_layout()

plt.savefig("ornek3.png")

plt.show()