1. Tạo xung lực đơn vị dn (Một xung lực đơn vị u[n] có chiều dài N được tạo bằng câu lệnh: un = signal.unit\_impulse(N). Đoạn chương trình sau tạo ra một xung lực đơn vị có số mẫu là 8)

Chart, histogram

Description automatically generated**import matplotlib.pyplot as plt**

**from scipy import signal**

**import numpy as np**

un = signal.unit\_impulse(8) #Hàm unit\_impulse được lấy từ thư viện signal 🡪 8 mẫu 0 đến 7

un2 = signal.unit\_impulse(8,2) # xung lực 8 đơn vị delay 2 mẫu

plt.stem(un) #vẽ rời rạc

n = np.arange (5) # tạo 1 mảng 5 phần tử 0 🡪4

x = np.ones(n,x) # tạo một mảng 5 phần tử có giá trị 1

plt.stem(n,x)

Chart, histogram

Description automatically generated

x = np.concatenate([np.zeros(5), np.ones(15)] ) #Viết chương trình tạo ra bậc đơn vị chiều dài 20, delay 5 mẫu u(n-5)

n = np.arange(0,20)

plt.stem(n,x)

t = np.linspace(0, 1, 500, endpoint=False) #Tạo 1 dải tuyến tính từ 0 -> 1, số mẫu là 500, không lấy mẫu cuối cùng

s = signal.square(2 \* np.pi \* 5 \* t, duty = 0.5) #Tạo sóng vuông tần số 5Hz, với tần số lấy mẫu là 500Hz

plt.plot(t, s)

plt.ylim(-2, 2)  #Giới hạn trục y từ -2 đến 2

Chart, histogram

Description automatically generated

fs = 100 # Tốc độ lấy mẫu (Hz)

f = 2 # Tần số của tín hiệu

N = 100 # Số mẫu

n = np.arange(N)/fs #Tạo ra 100 mẫu với chu kỳ lấy mẫu là Ts = 1/fs

y = 2\*np.sin(2\*np.pi\*f \* n) #Tính biên độ của sóng sin

plt.stem(n,y, 'r' )

t = 4

fs = 100

n = np.linspace(0, t, **fs\*t**, endpoint=False)

**Đáp ứng xung dùng lfilter()**

n = np.arange(10)

x = [1.5, -4, 6, 2.5, -3, 0, 0, 0, 0, 0]

b = [1.2, -0.85, 1] #Các hệ số của x

a = [ 1] #Các hệ số của y, lưu ý đến dấu của các hệ số

'''

a, b là hệ số của phương trình vào ra:

a[0]\*y[n] = b[0]\*x[n] + b[1]\*x[n-1] + ... + b[M]\*x[n-M]

                      - a[1]\*y[n-1] - ... - a[N]\*y[n-N]

Hoặc viết dưới dạng hàm chuyển biến đổi z: b là các hệ số của tử số, a là các hệ số của mẫu số

                    -1              -M

        b[0] + b[1]z  + ... + b[M] z

Y(z) = -------------------------------- X(z)

                    -1              -N

        a[0] + a[1]z  + ... + a[N] z

'''

**y = signal.lfilter(b, a, x)**

######## **NGƯỢC LẠI VỚI lfilter () dùng convolve()**

n = np.arange(10)

x = [1.5, -4, 6, 2.5, -3, 0, 0, 0, 0, 0]

h = [ 1.2, -0.85, 1, 0, 0, 0, 0, 0, 0, 0]

**y = signal.convolve(x, h, mode='full')**

y = y[:10]y

print(y) # Xem kết quả

**LỌC FIR**

**Signal.firwin (**BacLoc, TanSoCatChuanHoa, **window**=DangCuaSoBoLoc**)**

**Wc = fc/nyquistRate (fc là tần số cắt)**

**VD1:** **Thiết kế và vẽ đáp ứng tần số của bộ lọc FIR với các yêu cầu sau Lọc thấp qua N = 65, fc = 15KHz, của sổ hann, Biết: fs = 44.1KHz**

nSamples=400 #số mẫu

sampleRate = 44100 #fs tần số lấy mẫu

nyquistRate = fs/2 #tần số nyquist

fc =15000 #tần số cắt

wC = fc/nyquistRate

b=signal.firwin(65,wC,window="hann") #**thấp qua**  🡺 b=signal.firwin(65,wC,window="hamming",pass\_zero=False) #**cao qua cửa sổ hamming**

w,H=signal.freqz(b,1,worN=1024)

plt.plot((w/np.pi)\*nyquistRate,abs(H),linewidth=2) #vẽ đáp ứng tần số của bộ lọc

plt.title('Đáp ứng tần số')

plt.xlabel('Tần số (Hz)')

plt.ylabel('Biên độ')

**VD2:** **Thiết kế và vẽ đáp ứng tần số của bộ lọc FIR với các yêu cầu sau Lọc dải qua N = 23, trong khoảng tần số 4khz🡪8khz, Biết: fs = 44.1KHz**

fs = 44100

nyquistRate = fs/2

f1 = 4000

f2= 8000

w1 = f1/nyquistRate

w2 = f2/nyquistRate

wC = [w1 ,w2 ]

b=signal.firwin(23,wC,pass\_zero=False) # **lọc dải qa 🡺** b=signal.firwin(23,wC) #**lọc dải chặn**

w, H = signal.freqz(b,1,worN=1024)

plt.plot((w/np.pi)\*nyquistRate,abs(H))

plt.title('Đáp ứng tần số')

plt.xlabel('Tần số (Hz)')

plt.ylabel('Biên độ')

**VD3: Cho tín hiệu X ||a) Thực hiện mạch lọc loại bỏ tín hiệu 0.5Hz và 2.5Hz || b) Thực hiện mạch lọc chỉ giữ lại tần số 15.3Hz (dùng dải qua )**

1. sampleRate = 100 #Tần số lấy mẫu

nSamples = 400   #Số mẫu

**t = np.arange(nSamples)/sampleRate #miền thời gian**

x = np.cos(2\*np.pi\*0.5\*t) + 0.2\*np.sin(2\*np.pi\*2.5\*t+0.1) + np.sin(2\*np.pi\*15.3\*t) + 0.1\*np.sin(2\*np.pi\*18.7\*t + 0.1) + 0.1\*np.sin(2\*np.pi\*23.45\*t+.8)

**#Viết tiếp chương trình ở đây**

w, X = signal.freqz(x, 1, worN=1024)

fc=10 #loại bỏ nên dùng cao qua tần số cắt 10 để bỏ 0.5 và 2.5

N=39

wC=fc/50 # w = fc/nyquistRate

nyquistRate = sampleRate/2

**b = signal.firwin(N, wc, pass\_zero=False)**

w, H = signal.freqz(b, 1, worN=1024)

**y = signal.lfilter(b, 1, x)**

w, Y = signal.freqz(y, 1, worN=1024)

plt.subplot(2,1,1)

plt.plot(w/np.pi\*nyquistRate, abs(H), linewidth=2)

plt.title('Đáp ứng tần số')

plt.xlabel('Tần số (Hz)')

plt.ylabel('Biên độ')

plt.subplot(2,1,2)

**plt.plot((w/np.pi)\*nyquistRate, abs(X),'b') # tín hiệu trc khi qua lọc theo miền tần số >< plt.plot(t,x) # theo miền thời gian**

**plt.plot((w/np.pi)\*nyquistRate, abs(Y),'r') #tín hiệu sau khi qua lọc theo miền tần số >< plt.plot(t,y) # theo miền thời gian**

plt.title('phổ biên độ')

plt.xlabel('tần số')

plt.ylabel('Biên độ')

plt.subplots\_adjust(top=1.5, hspace=0.5)

**LỌC IIR**

**b, a = iirfilter(N, Wn, rp=None, rs=None, btype='band', analog=False,  ftype='butter', output='ba')**

**rp**: Độ dợn sóng tối đa dải qua (dB) (Đối với lọc  Chebyshev và elliptic) || **rs**: Độ dợn sóng tối thiểu ở dải chặn (dB) (Đối với lọc Chebyshev và elliptic)

**Butterworth** : ‘butter’>> \* **Chebyshev** I : ‘cheby1’>> \* **Chebyshev II** : ‘cheby2’>> \* **Cauer/elliptic**: ‘ellip’>> \* **Bessel/Thomson**: ‘bessel’

**VD1: Thiết kế và vẽ đáp ứng tần số của mạch lọc IIR Butterworth thấp qua với các thông số sau: <br>Tần số cắt 8KHz, Bậc lọc N = 4. Biết: fs = 44.1KHz**

fc = 8000

N = 4

fs = 44100 # tần số lấy mẫu

nyquistRate = fs/2

wC = fc/nyquistRate

**b,a = signal.iirfilter(N,wC,rs = 60, btype="lowpass",analog=False,ftype="butter") # cao qua: ‘highpass’ #dải qua: ‘band’ #dải chặn: ‘stop’**

**w,H = signal.freqz(b,a,worN=512)**

**plt.plot((w/np.pi)\*nyquistRate,abs(H)) # nếu theo rad/sample thì 20\*np.log10(abs(H))**

plt.title('Dap ung tan so')

plt.xlabel('TanSo Hz')

plt.ylabel('BienDo')

**## lưu ý : TH tần số lớn 🡪 đồ thị hiện đoạn cắt ko rõ thì dùng plt.xlim(0,soGanTanSoCat) để hiển thị**

**Vấn đề subplot: dùng plt.subplots\_adjust(top=1.5, hspace=0.5)**