1.

import numpy as np

import matplotlib.pyplot as plt

def downsampling( x, method = 1 ):

    N = int( len( x ) / 2 )

    y = np.zeros( N )

    if method == 1:             # Decimation

        for n in range( N ):

            y[n] = x[2\*n]

    else:                       # Average

        for n in range( N ):

            y[n] = ( x[2\*n] + x[2\*n+1] ) / 2

    return y

def upsampling( x, method = 1 ):

    N = len( x ) \* 2

    y = np.zeros( N )

    if method == 1:             # Zero-Order Hold 零階保持

        for n in range( N ):

            y[n] = x[int( n / 2 )]

    else:                       # Linear Interpolation 線性內插

        for n in range( N ):

            if int( n / 2 ) == n / 2:

                y[n] = x[int( n / 2 )]

            else:

                n1 = int( n / 2 )

                n2 = n1 + 1

                if n2 < len( x ):

                    y[n] = ( x[n1] + x[n2] ) / 2

                else:

                    y[n] = x[n1] / 2

    return y

def main( ):

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

    y1 = downsampling( x, 1 )

    y2 = upsampling(x, 2)

    plt.figure( 1 )

    plt.stem( x )

    plt.xlabel( 'input' )

    plt.ylabel( 'intensity' )

    plt.figure( 2 )

    plt.stem( y1 )

    plt.xlabel( 'Decimation' )

    plt.ylabel( 'Intensity' )

    plt.figure( 3 )

    plt.stem( y2 )

    plt.show()

main( )

依題目以2下取樣為結果,所以每隔一個元素取樣 輸入序列:{1,2,4,3,2,1,2,1} 以2下取樣= {1,4,2,2}

依題目用線性內插法以2上取樣,在每個原始樣本x[n]之間插入中間值(x[n] + x[n+1]) / 2, 上取樣結果為{1, 1.5, 2, 3,4, 3.5, 3,2.5, 2,1.5, 1, 1.5, 2, 1.5, 1,0.5 }

2.

import numpy as np

x = np.array( [ 1, 2, 4, 3, 2, 1, 1 ] ) # 數位訊號 N=7

h = np.array([1,2,3,1,1]) # 脈衝響應 M=5 ，Zero-padding M-1=4

y = np.convolve( x, h, 'full' ) #全卷積 M+N-1= 5+7-1=11

y1 = np.convolve( x, h, 'same' ) #同卷積 n=7

print( "x =", x )

print( "h =", h )

print( "Full Convolution y =", y )

print( "same Convolution y =", y1 )

Zero-padding M-1=4 兩邊補上4個0

1 2 3 1 1 轉180度 1 1 3 2 1

|  |  |
| --- | --- |
| X | h |
| 0 0 0 0 1 2 4 3 2 1 1 0 0 0 0 | 1 2 3 1 1 |

1 1 3 2 1

y[0] = 0\*1+0\*1+0\*3+0\*2+1\*1=1

|  |  |
| --- | --- |
| X | h |
| 0 0 0 0 1 2 4 3 2 1 1 0 0 0 0 | 1 2 3 1 1 |

1 1 3 2 1

y[1] = 0\*1+0\*1+0\*3+1\*2+2\*1=4

全卷積 M+N-1= 5+7-1= 11

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| y[n] | 1 | 4 | 11 | 18 | 23 | 20 | 16 | 10 | 6 | 2 | 1 |

同卷積

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| n | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| x[n] | 1 | 2 | 4 | 3 | 2 | 1 | 1 |
| y[n] | 11 | 18 | 23 | 20 | 16 | 10 | 6 |

3.

import numpy as np

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

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

y = np.correlate( x, h, 'full' )

y1 = np.correlate( x, h, 'same' )

print( "x =", x )

print( "h =", h )

print( "Full Correlation y =", y )

print( "Correlation y =", y1 )

M=5, M-1=4 , 兩邊補上4個0

|  |  |
| --- | --- |
| X | h |
| 0 0 0 0 1 2 4 3 2 1 1 0 0 0 0 | 1 2 3 1 1 |

1 2 3 1 1

y[0] = 0\*1+0\*2+0\*3+0\*1+1\*1=1

|  |  |
| --- | --- |
| X | h |
| 0 0 0 0 1 2 4 3 2 1 1 0 0 0 0 | 1 2 3 1 1 |

1 2 3 1 1

y[1] = 0\*1+0\*2+0\*3+1\*1+2\*1=3

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| y[n] | 1 | 3 | 9 | 15 | 22 | 22 | 18 | 11 | 7 | 3 | 1 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| n | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| x[n] | 1 | 2 | 4 | 3 | 2 | 1 | 1 |
| y[n] | 9 | 15 | 22 | 22 | 18 | 11 | 7 |

4.

import numpy as np

import matplotlib.pyplot as plt

N = eval( input( "Please enter number of terms for partial sum: " ) )

t = np.linspace( -1, 1, 1000 )  # 定義時間陣列t,範圍為-1到1,並包含1000個點

x = np.zeros( 1000 )            # 一個長度為1000的零陣列,儲存方波的傅立葉級數的和

for n in range( 1, N + 1 ):

    x += 2 / ( n \* np.pi ) \* ( 1 - np.power( -1, n ) ) \* np.sin( n \* np.pi \* t )

#迴圈從 n= 1 到 N , 計算震幅 '2 / ( n \* np.pi ) \* ( 1 - np.power( -1, n ) )' , 計算正弦波'np.sin( n \* np.pi \* t )',將震幅跟正弦波加到X陣列

plt.plot( t, x )            # 繪製時間 t 與振幅 x 的曲線

plt.xlabel( 't (second)' )  # 設定 x 軸的標籤為 't (second)'

plt.ylabel( 'Amplitude' )   # 設定 y 軸的標籤為 'Amplitude'

plt.show( )

5.

import numpy as np

from numpy.fft import fft, fftshift, fftfreq

import matplotlib.pyplot as plt

t = np.linspace( 0, 1, 1000, endpoint = False ) #定義時間陣列t,從0 ~ 1秒,包含1000個樣本

x = np.cos( 2 \* np.pi \* 100 \* t )  #頻率為100HZ的餘弦信號的陣列x

f = fftshift( fftfreq( 1000, 0.001) ) #計算頻率軸的值,1000是樣本,0.001是時間間隔

X = fftshift( fft( x ) )    # fft(x) 對信號 x 做傅立葉變換，得到頻譜, fftshift 對頻譜和頻率軸進行位移，將零頻率移到頻譜中心

Xm = abs( X )   #取頻譜的絕對值，得到頻譜的振幅

plt.plot( f, Xm )   #  繪製頻譜圖, x軸為f、y軸為Xm

plt.xlabel( 'f' )   # x軸標籤為'f'

plt.ylabel( 'Magnitude' )   # y軸標籤為'Magnitude'

plt.show( )