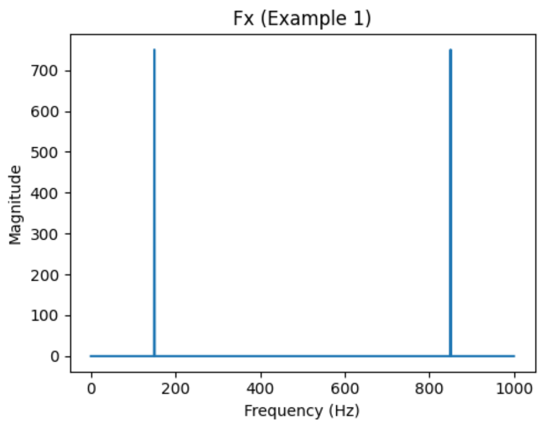
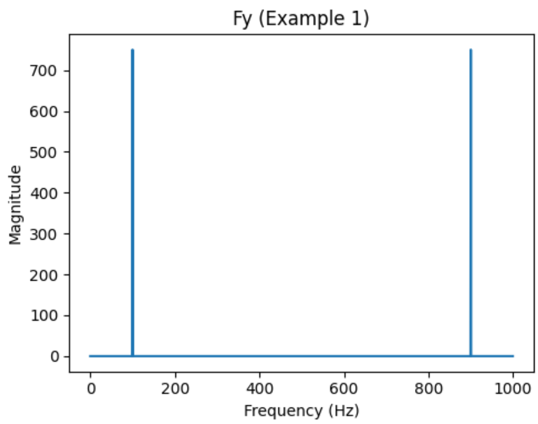
(1) Write the Matlab or Python code to compute the FFT of two N-point real signals x and y using only one N-point FFT. (20 scores)

$$[Fx, Fy] = \text{fftreal}(x, y)$$

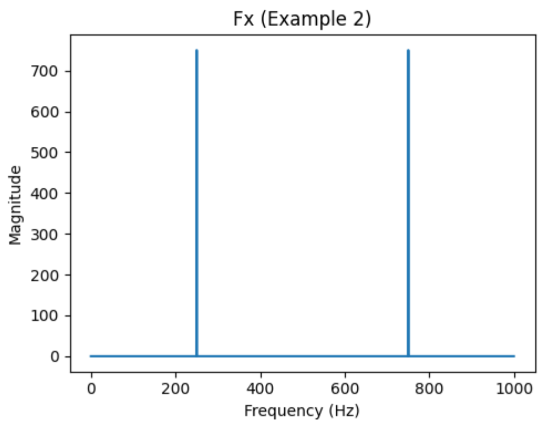
The code should be handed out by NTUCool.

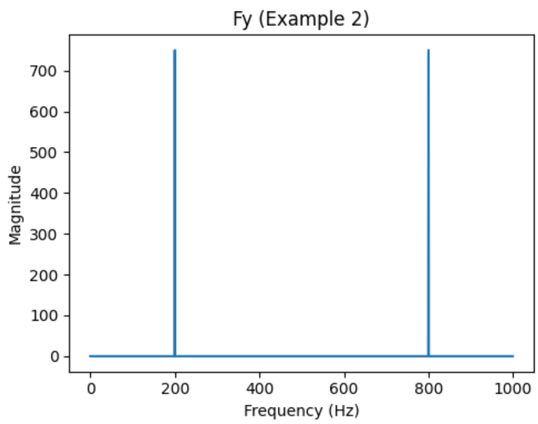
```python
# Example data for first signal pair
Fs = 1000  # Sampling frequency
T = 1 / Fs  # Sampling period
L = 1500  # Length of signal
t = np.arange(0, L) * T  # Time vector
f = Fs * np.arange(0, L) / L  # Frequency vector

# Example signals
f1_example1 = np.cos(2 * np.pi * 150 * t)
f2_example1 = np.sin(2 * np.pi * 100 * t)
```






```python
# Example data for second signal pair
f1_example2 = np.cos(2 * np.pi * 250 * t)
f2_example2 = np.sin(2 * np.pi * 200 * t)
```





(2) Suppose that length($x[n]$) = 1200. What is the best way to implement the convolution of $x[n]$ and $y[n]$ if

(a) length($y[n]$) = 300, (b) length($y[n]$) = 30,

(c) length($y[n]$) = 8, and (d) length($y[n]$) = 2?

Please show (i) the calculation method (direct, non-sectioned convolution, or sectioned convolution), (ii) the number of points of the FFT, (iii) and the number of real multiplications for the best implementation method. Also, consider the general case where $x[n]$ and $y[n]$ are complex sequences and the FFT of $y[n]$ can be computed in prior. (25 scores)

Sol:

length(x[n]) = 1200 = N

a.

length(y[n]) = 300 = M

1.Direct => 3 X M X N = 1080000

2.IFFT(FFT(x)FFT(h))

P ≥ M + N -1 = 1499

If P = 1680

Number of real multiplications = 2 X MUL_p + 3 X P = 2X10420+ 3 X

1680 =25880

3.Sectioned convolution

If L = 373

P ≥ L + M -1 = 672

S = 4

Number of real multiplications = 2S X MUL_p + 3S X P =4(2x3496 +

3X672) = 36032

If L = 261

P $\geq$ L + M -1 = 560

S = 5

Number of real multiplications = 2S X MUL_p + 3S X P =5(2x3100 +

3X560) = 39400

If L = 37

P $\geq$ L + M -1 = 336

S = 33

Number of real multiplications = 2S X MUL_p + 3S X P =33(2x1412 +

3X336) = 126456

Best way is IFFT(FFT(x)FFT(h)).

b.

length(y[n]) = 30 = M

1.Direct => 3 X M X N = 108000

2.IFFT(FFT(x)FFT(h))

P $\geq$ M + N -1 = 1229

If P = 1260

Number of real multiplications = 2 X MUL_p + 3 X P = 2X7640+ 3 X 1260

=19060

3.Sectioned convolution

If L = 211

P $\geq$ L + M -1 = 240

S = 6

Number of real multiplications = 2S X MUL_p + 3S X P =6(2x940 + 3X240)

= 15600

If L = 139

P $\geq$ L + M -1 = 168

S = 9

Number of real multiplications = 2S X MUL_p + 3S X P =9(2x580 + 3X168)

= 14976

If L = 163

P $\geq$ L + M -1 = 192

S = 8

Number of real multiplications = 2S X MUL_p + 3S X P =8(2x752 + 3X192)

= 16640

Best way is Sectioned convolution

c.

length(y[n]) = 8 = M

1.Direct => 3 X M X N = 28800

2.IFFT(FFT(x)FFT(h))

P $\geq$ M + N -1 = 1207

If P = 1260

Number of real multiplications = 2 X MUL_p + 3 X P = 2X7640+ 3 X 1260

=19060

3.Sectioned convolution

If L = 29

P $\geq$ L + M -1 = 36

S = 42

Number of real multiplications = 2S X MUL_p + 3S X P =42(2x64 + 3X36)

= 9912

If L = 32

P $\geq$ L + M -1 = 39

S = 38

Number of real multiplications = 2S X MUL_p + 3S X P =38(2x182 + 3X39)

= 18278

If L = 17

P $\geq$ L + M -1 = 24

S = 71

Number of real multiplications = 2S X MUL_p + 3S X P =71(2x28 + 3X24)

= 9088

Best way is Sectioned convolution

d.

length(y[n]) = 2 = M

1.Direct => 3 X M X N = 7200

2.IFFT(FFT(x)FFT(h))

P $\geq$ M + N -1 = 1201

If P = 1260

Number of real multiplications = 2 X MUL_p + 3 X P = 2X7640+ 3 X 1260

=19060

3.Sectioned convolution

If L = 3

P $\geq$ L + M -1 = 4

S = 400

Number of real multiplications = 2S X MUL_p + 3S X P =400(2x0 + 3X4) =

4800

If L = 1

P $\geq$ L + M -1 = 2

S = 1200

Number of real multiplications = 2S X MUL_p + 3S X P =1200(2x0 + 3X2)

= 7200

Best way is Sectioned convolution

(3) (a) What are the number of entries equal to 1 and -1 for the 2^k-point Walsh transform? (b) What are the number of entries equal to 1, 0, and -1 for the 2^k-point Haar transform? (c) What is the most important application of the Walsh transform nowadays? (d) What is the most important advantage of the Haar transform nowadays? (20 scores)

Sol:

a.

$$\mathbf{W_2} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

當$\mathrm{N} = 2^1$，則有 3 個 1，1 個-1

$$\mathbf{W_4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

當$\mathrm{N} = 2^2$，則有 10 個 1，6 個-1

除了第一 col.都是 1，後面每次都是 1 跟-1 各半，

所以推出

1 為$2^k + \frac{2^k}{2}(2^k - 1)$

-1 為$\frac{2^k}{2}(2^k - 1)$

b.

$$N=2 \qquad \mathbf{H}_2=\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$$

當$N=2^1$，則有 3 個 1，1 個-1，0 個 0，且可以分為兩組

$$N=4 \qquad \mathbf{H}_4=\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & 1 & -1 & -1\\ 1 & -1 & 0 & 0\\ 0 & 0 & 1 & -1\end{bmatrix}$$

當$N=2^2$，則有 8 個 1，4 個-1，4 個 0，且可以分為三組

$$N=8 \qquad \mathbf{H}_8=\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1\\ 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1\\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1\end{bmatrix}$$

當$N=2^3$，則有 20 個 1，12 個-1，32 個 0，且可以分為 4 組

$N = 16$

$$\mathbf{H}_{16} = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\
1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1
\end{bmatrix}$$

當$N = 2^4$，則有 48 個 1，32 個-1，176 個 0，且可以分為 5 組

根據上面得知一些規律，

第一組 col 都為 1

第二組 N/2 個 1，N/2 個-1，然後 0 會有 0 個

第三組 N/2 個 1，N/2 個-1，然後 0 會有 N 個

第四組 N/2 個 1，N/2 個-1，然後 0 會有 3N 個

第五組 N/2 個 1，N/2 個-1，然後 0 會有 7N 個

最後一組會有 N/2 個 1，N/2 個-1，然後 0 會有$N(2^{k-1} - 1)$個

共會有 k+1 組

所以推導出

1 為$\mathrm{N}+\frac{\mathrm{N}}{2}k$=> 2^k+k2^{k-1}

-1 為$\frac{\mathrm{N}}{2}k$=> $k2^{k-1}$

0 為$\mathrm{N}(2^{k-1}-1)+\mathrm{N}(2^{k-2}-1)+\mathrm{N}(2^{k-3}-1)+\ldots+\mathrm{N}(2^{0}-1)$

=> $\mathrm{N}[(2^{k-1}+2^{k-2}+\ldots+2^{0})-k]$

=>$2^k(\frac{(2^k-1)1}{2-1})-\mathrm{k}2^k$

=>$2^{2k}-\mathrm{k}2^k-2^k$

c.

CDMA (code division multiple access)

using the basis(rows) of the walsh transform to perform modulation.

其中 modulation: using some man-made waveform to represent a data.

d.

Analysis of the local high frequency component. (The wavelet transform is a generalization of the Haar transform)

其中 local high frequency component.(edge of different locations and scales)

(4) (a) What is the results of CDMA if there are three data [1 0 1], [1 1 0], [0 1 1] and these three data are modulated by the 1st, 4th, and 10th rows of the 16-point Walsh transform? (The beginning row is the 1st row). (10 scores)

(b) In (a), if the 7th and the 19th entries of the CDMA results are missed, can we recover the original data? Why? (5 scores)

Sol:

a.

[1 0 1] => [1 -1 1]

[1 1 0] => [1 1 -1]

[0 1 1] => [-1 1 1]

$$V_{16} = \begin{bmatrix} W_8 & W_8 \\ W_8 & -W_8 \end{bmatrix} = \begin{array}{c} \\ \left[\begin{array}{cccccccccccccccc}
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\
1&1&-1&-1&1&1&-1&-1&1&1&-1&-1&1&1&-1&-1\\
1&-1&-1&1&1&-1&-1&1&1&-1&-1&1&1&-1&-1&1\\
1&-1&1&-1&1&-1&1&-1&1&-1&1&-1&1&-1&1&-1\\
1&1&1&1&-1&-1&-1&-1&1&1&1&1&-1&-1&-1&-1\\
1&1&-1&-1&-1&-1&1&1&1&1&-1&-1&-1&-1&1&1\\
1&-1&-1&1&-1&1&1&-1&1&-1&-1&1&-1&1&1&-1\\
1&-1&1&-1&-1&1&-1&1&1&-1&1&-1&-1&1&-1&1\\
1&1&1&1&1&1&1&1&-1&-1&-1&-1&-1&-1&-1&-1\\
1&1&-1&-1&1&1&-1&-1&-1&-1&1&1&-1&-1&1&1\\
1&-1&-1&1&1&-1&-1&1&-1&1&1&-1&-1&1&1&-1\\
1&-1&1&-1&1&-1&1&-1&-1&1&-1&1&-1&1&-1&1\\
1&1&1&1&-1&-1&-1&-1&-1&-1&-1&-1&1&1&1&1\\
1&1&-1&-1&-1&-1&1&1&-1&-1&1&1&1&1&-1&-1\\
1&-1&-1&1&-1&1&1&-1&-1&1&1&-1&1&-1&-1&1\\
1&-1&1&-1&-1&1&-1&1&-1&1&-1&1&1&-1&1&-1
\end{array}\right] \end{array}
\begin{array}{c} \text{sign changes} \\ 0\\7\\8\\15\\3\\4\\11\\12\\1\\6\\9\\14\\2\\5\\10\\13 \end{array}$$

$$W_{16} = \left[\begin{array}{cccccccccccccccc}
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&-1&-1&-1&-1&-1&-1&-1&-1\\
1&1&1&1&-1&-1&-1&-1&-1&-1&-1&-1&1&1&1&1\\
1&1&1&1&-1&-1&-1&-1&1&1&1&1&-1&-1&-1&-1\\
1&1&-1&-1&-1&-1&1&1&1&1&-1&-1&-1&-1&1&1\\
1&1&-1&-1&-1&-1&1&1&-1&-1&1&1&1&1&-1&-1\\
1&1&-1&-1&1&1&-1&-1&-1&-1&1&1&-1&-1&1&1\\
1&1&-1&-1&1&1&-1&-1&1&1&-1&-1&1&1&-1&-1\\
1&-1&-1&1&1&-1&-1&1&1&-1&-1&1&1&-1&-1&1\\
1&-1&-1&1&1&-1&-1&1&-1&1&1&-1&-1&1&1&-1\\
1&-1&-1&1&-1&1&1&-1&-1&1&1&-1&1&-1&-1&1\\
1&-1&-1&1&-1&1&1&-1&1&-1&-1&1&-1&1&1&-1\\
1&-1&1&-1&-1&1&-1&1&1&-1&1&-1&-1&1&-1&1\\
1&-1&1&-1&-1&1&-1&1&-1&1&-1&1&1&-1&1&-1\\
1&-1&1&-1&1&-1&1&-1&-1&1&-1&1&-1&1&-1&1\\
1&-1&1&-1&1&-1&1&-1&1&-1&1&-1&1&-1&1&-1
\end{array}\right]
\begin{array}{c} 0\\1\\2\\3\\4\\5\\6\\7\\8\\9\\10\\11\\12\\13\\14\\15 \end{array}$$

v1 = [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

[1 -1 1] modulated by V1 =>[v1 -v1 v1]

=>

[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

v2 = [1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1]

[1 1 -1] modulated by V1 =>[v2 v2 -v2]

=>

[1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1

1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1

-1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1]

v3 = [1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1]

[-1 1 1] modulated by V1 =>[-v3 v3 v3]

=>

[-1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1

1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1

1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1]

合成:

[1 3 3 1 -1 1 1 -1 1 3 3 1 -1 1 1 -1

1 -1 -1 1 -1 -3 -3 -1 1 -1 -1 1 -1 -3 -3 -1

1 -1 -1 1 3 1 1 3 1 -1 -1 1 3 1 1 3]

b.

[1 3 3 1 -1 1 0 -1 1 3 3 1 -1 1 1 -1

1 -1 0 1 -1 -3 -3 -1 1 -1 -1 1 -1 -3 -3 -1

1 -1 -1 1 3 1 1 3 1 -1 -1 1 3 1 1 3]

=>

[1 3 3 1 -1 1 0 -1 1 3 3 1 -1 1 1 -1]

[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

做內積=>16

[1 -1 0 1 -1 -3 -3 -1 1 -1 -1 1 -1 -3 -3 -1]

[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

做內積=>-16

[1 -1 -1 1 3 1 1 3 1 -1 -1 1 3 1 1 3]

[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

做內積=>16

16/N=1 -16/N=-1 16/N=1

[1 -1 1] => [1 0 1]

[1 3 3 1 -1 1 0 -1 1 3 3 1 -1 1 1 -1]

[1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1]

做內積=>16

[1 -1 0 1 -1 -3 -3 -1 1 -1 -1 1 -1 -3 -3 -1]

[1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1]

做內積=>16

[1 -1 -1 1 3 1 1 3 1 -1 -1 1 3 1 1 3]

[1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1]

做內積=>-16

16/N=1 16/N=1 -16/N=-1

[1 1 -1] => [1 1 0]

[1 3 3 1 -1 1 0 -1 1 3 3 1 -1 1 1 -1]

[1 -1 -1 1 1 -1 -1 1 1,-1 -1 1 1 -1 -1 1]

做內積=>-16

[1 -1 0 1 -1 -3 -3 -1 1 -1 -1 1 -1 -3 -3 -1]

[1 -1 -1 1 1 -1 -1 1 1,-1 -1 1 1 -1 -1 1]

做內積=>16

[1 -1 -1 1 3 1 1 3 1 -1 -1 1 3 1 1 3]

[1 -1 -1 1 1 -1 -1 1 1,-1 -1 1 1 -1 -1 1]

做內積=>16

-16/N=-1 16/N=1 16/N=1

[-1 1 1] => [0 1 1]

原因:

Walsh 矩陣的稀疏性：Walsh 矩陣的行和列是相互正交的，因此可以通過少量的數據點進行恢復。

冗餘性：CDMA 系統設計通常會有一定的冗餘，使得即使某些數據點缺失，還是可以通過剩餘的數據進行恢復。

而且經過計算的確可以 recover

(5) Ramanujan's Sum in NTT

Given $M = 11$, $\alpha = 8+6i$, and $N = 12$. Please determine the complex number theoretic transform (CNT) of $\mathbf{x}$ if

$$\mathbf{x} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Hint: fft(x) is as follows, which is Ramanujan's Sum

$$fft(\mathbf{x}) = \begin{bmatrix} 4 & 0 & 2 & 0 & -2 & 0 & -4 & 0 & -2 & 0 & 2 & 0 \end{bmatrix} \qquad \text{(8 scores)}$$

Sol:

FFT Matrix:

```
[[ 1.         +0.j          1.         +0.j          1.         +0.j
   1.         +0.j          1.         +0.j          1.         +0.j
   1.         +0.j          1.         +0.j          1.         +0.j
   1.         +0.j          1.         +0.j          1.         +0.j        ]
 [ 1.         +0.j          0.8660254 +0.5j         0.5        +0.8660254j
   0.         +1.j         -0.5        +0.8660254j -0.8660254 +0.5j
  -1.         +0.j         -0.8660254 -0.5j        -0.5        -0.8660254j
   0.         -1.j          0.5        -0.8660254j  0.8660254 -0.5j       ]
 [ 1.         +0.j          0.5        +0.8660254j -0.5        +0.8660254j
  -1.         +0.j         -0.5        -0.8660254j  0.5        -0.8660254j
   1.         +0.j          0.5        +0.8660254j -0.5        +0.8660254j
  -1.         +0.j         -0.5        -0.8660254j  0.5        -0.8660254j]
 [ 1.         +0.j          0.         +1.j         -1.         +0.j
   0.         -1.j          1.         +0.j          0.         +1.j
  -1.         +0.j          0.         -1.j          1.         +0.j
   0.         +1.j         -1.         +0.j          0.         -1.j        ]
 [ 1.         +0.j         -0.5        +0.8660254j -0.5        -0.8660254j
   1.         +0.j         -0.5        +0.8660254j -0.5        -0.8660254j
   1.         +0.j         -0.5        +0.8660254j -0.5        -0.8660254j
   1.         +0.j         -0.5        +0.8660254j -0.5        -0.8660254j]
 [ 1.         +0.j         -0.8660254 +0.5j         0.5        -0.8660254j
   0.         +1.j          0.5        +0.8660254j -0.8660254 +0.5j
  -1.         +0.j         -0.8660254 -0.5j         0.5        -0.8660254j
   0.         -1.j          0.5        -0.8660254j -0.8660254 -0.5j       ]
 [ 1.         +0.j         -1.         +0.j          1.         +0.j
  -1.         +0.j          1.         +0.j         -1.         +0.j
   1.         +0.j         -1.         +0.j          1.         +0.j
  -1.         +0.j          1.         +0.j         -1.         +0.j        ]
 [ 1.         +0.j         -0.8660254 -0.5j         0.5        +0.8660254j
```

0. -1.j 0.5 -0.8660254j -0.8660254 -0.5j
-1. +0.j -0.8660254 +0.5j 0.5 -0.8660254j
0. +1.j 0.5 +0.8660254j -0.8660254 +0.5j]
[1. +0.j -0.5 -0.8660254j -0.5 +0.8660254j
1. +0.j -0.5 -0.8660254j -0.5 +0.8660254j
1. +0.j -0.5 -0.8660254j -0.5 +0.8660254j
1. +0.j -0.5 -0.8660254j -0.5 +0.8660254j]
[1. +0.j 0. -1.j -1. +0.j
0. +1.j 1. +0.j 0. -1.j
-1. +0.j 0. +1.j 1. +0.j
0. -1.j -1. +0.j 0. +1.j]
[1. +0.j 0.5 -0.8660254j -0.5 -0.8660254j
-1. +0.j -0.5 +0.8660254j 0.5 +0.8660254j
1. +0.j 0.5 -0.8660254j -0.5 -0.8660254j
-1. +0.j -0.5 +0.8660254j 0.5 +0.8660254j]
[1. +0.j 0.8660254 -0.5j 0.5 -0.8660254j
0. +1.j -0.5 -0.8660254j -0.8660254 -0.5j
-1. +0.j -0.8660254 +0.5j -0.5 -0.8660254j
0. -1.j 0.5 +0.8660254j 0.8660254 -0.5j]]

則 fft(x) = FFT matrix X [0 1 0 0 0 1 0 1 0 0 0 1]^T
=[4 0 2 0 -2 0 -4 0 -2 0 2 0]^T

(6) (a) Please determine

$$3^{2049} \bmod 103 \qquad \text{(Hint: 費馬小定理)}$$

Sol:

by費馬小定理

$3^{102} \equiv 1 \pmod{103}$

$3^{1020} \equiv 1 \pmod{103}$

$3^{2040} \equiv 1 \pmod{103}$

$3^{2040} \cdot 3^{9} \equiv 3^{9} \pmod{103}$

$\therefore 3^{9} \bmod 103$

$\Rightarrow 3^{1} \equiv 3 \quad 3^{2} \equiv 9 \quad 3^{3} \equiv 27 \quad 3^{4} \equiv 81 \pmod{103}$

$\Rightarrow (3^{3})^{3} \equiv 27^{3} \equiv 10 \pmod{103}$

$\therefore 3^{2049} \equiv 10 \pmod{103}$

(b) Suppose that $x \bmod 43 = 2$ *and* $x \bmod 67 = 13$

Please Determine

$x \bmod 2881.$ (Hint: Chinese Remainder Theorem)

Sol:

$x \equiv 2 \bmod 43$

$x \equiv 13 \bmod 67$

$r_1 = 2 \quad r_2 = 13$

$n_1 = 43 \quad n_2 = 67$

$n = 43 \times 67 = 2881$

$N_1 = \frac{n}{n_1} = 67 \quad N_2 = \frac{n}{n_2} = 43$

$M_1 \equiv 67^{-1} \equiv 9 \pmod{43}$

$M_2 \equiv 43^{-1} \equiv 53 \pmod{67}$

$x \equiv r_1 M_1 N_1 + r_2 M_2 N_2 \equiv 2 \times 9 \times 67 + 13 \times 53 \times 43$

$\equiv 30833 \equiv 2023 \pmod{2881}$

$\therefore x \bmod 2881 \equiv 2023$

(c) $n! = n(n-1)(n-2) \ldots. 1$. Please determine 39! mod 43

(Hint: Wilson's Theorem)

Sol:

$(43-1)! \equiv -1 \pmod{43}$

$42! \equiv -1 \pmod{43}$

$\downarrow$

$41! \equiv 1 \pmod{43}$

$41 \times 40! \equiv 1 \pmod{43}$

$41 \times 40 \times 39! \equiv 1 \pmod{43}$

$-2 \times -3 \times 39! \equiv 1 \pmod{43}$

$\Rightarrow 6y \equiv 1 \pmod{43}$

$y = 36$

$\therefore 36 \times 6 \times 39! \equiv 36 \pmod{43}$

$\Rightarrow 39! \equiv 36 \pmod{43}$