EE321 DSP LAB

EXPERIMENT NO: 04

(DFT)

DISCRETE FOURIER TRANSFORM

Date: 28th Feb'24

Lab Group No: 12

Submitted By:

1) Aviral Srivastava-[2101EE88]
2) Divyansh Gupta-[2101EE89]
3) Ayush Tripathi-[2101EE90]

Aim 1: Find the DFT and IDFT of the given sequence using definitions. Say, x[n] = {0, 1, 2, 3}

MATLAB Code:

```
clc ;
clear all ;
x = [0,1,2,3] ;
N = length(x) ;
Wn = exp((-1i * 2*pi)/N) ;
dftt= zeros(1,N) ;
for k = 1 : N
    for n = 1 : N
        dftt(k) = dftt(k) + x(n) * (Wn)^((n-1)*(k-1)) ;
    end
end

disp(dftt)
subplot(2,1,1)
stem(real(dftt),'Marker','none','Color',[1 0 0], 'LineWidth',2)
ylabel('Real x[k]')
xlabel('k')
title('DFT plot of x(n)')
subplot(2,1,2)
stem(imag(dftt),'Marker','none','Color',[0 1 1], 'LineWidth',2)
ylabel('Imag x[k]')
xlabel('k')

idft=zeros(1,N) ;
for n = 1 : N
    for k = 1 : N
        idft(n) = idft(n) + dftt(k) * (1/Wn)^((n-1)*(k-1)) ;
    end
    idft(n)=(idft(n))/N;
end

disp(idft)
%subplot(2,1,1)
%(idft,'Marker','none','Color',[0 0 1], 'LineWidth',2)
%ylabel('x(n)')
%xlabel('n')
```

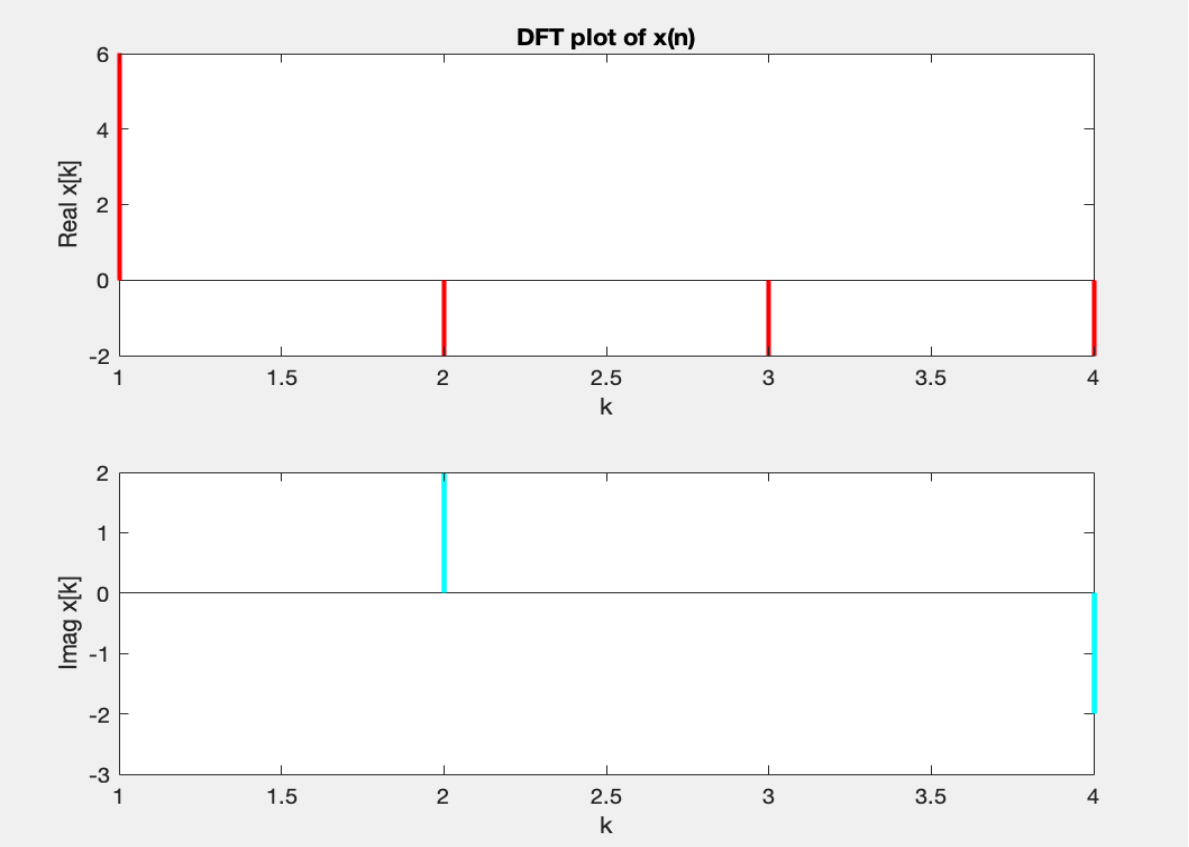
DFT of x(n):

X[k]= [6.0000 + 0.0000i, -2.0000 + 2.0000i, -2.0000 - 0.0000i, -2.0000 - 2.0000i]

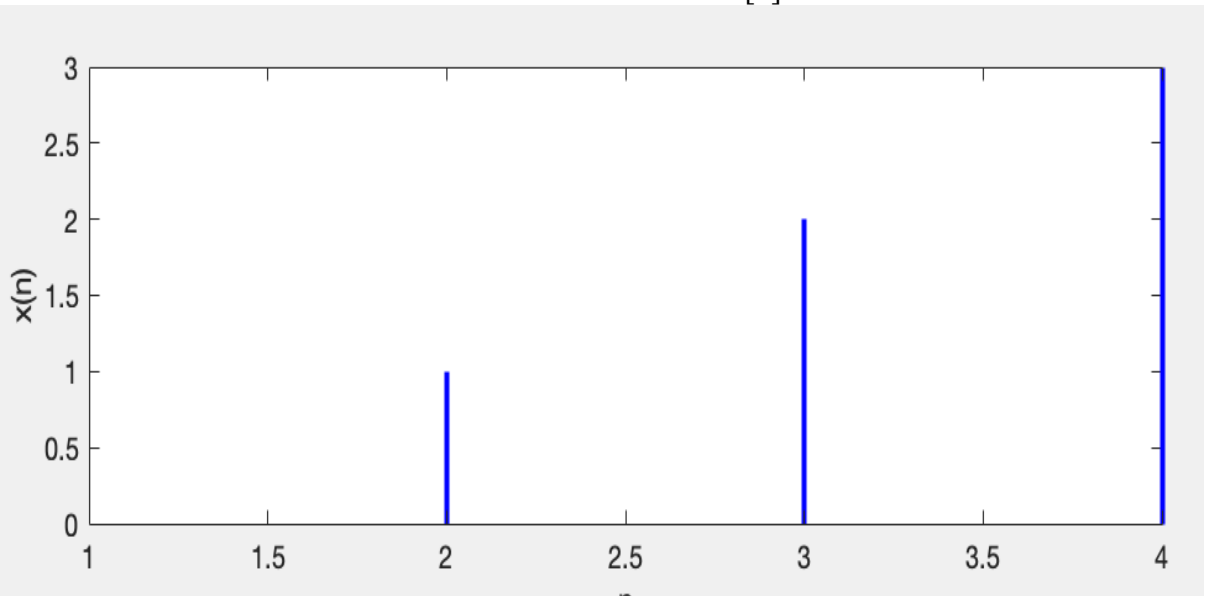
IDFT of x[k]:

X(n)= [0.0000 - 0.0000i , 1.0000 - 0.0000i, 2.0000 - 0.0000i, 3.0000 + 0.0000i]

Output:



IDFT of x[k]



Aim 2: Repeat **Aim 1** using **COMPACT MATRIX representation.**

MATLAB Code:

```
clc ;
clear all ;
x = [0,1,2,3] ;
N = length(x) ;
Wn = exp((-1i * 2*pi)/N) ;
Dn = zeros(N) ;
for row = 1:N
    for col = 1:N
        Dn(row,col) = (Wn)^((row-1)*(col-1)) ;
    end
end
disp(Dn)
dft = Dn * transpose(x) ;
disp(dft)
%Method 1 for IDFT with complexity O(N^2)
Idft1 = (conj(Dn) * dft)/N ;
disp(idft)
%Method 2 for IDFT with complexity O(N^3)
idft2 = (inv(Dn) * dft);
disp(idft2)
```

Output:

```
Dn matrix:  (4*4)
 1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i
 1.0000 + 0.0000i   0.0000 - 1.0000i  -1.0000 - 0.0000i  -0.0000 +1.0000i
 1.0000 + 0.0000i  -1.0000 - 0.0000i   1.0000 + 0.0000i  -1.0000 - 0.0000i
 1.0000 + 0.0000i  -0.0000 + 1.0000i  -1.0000 - 0.0000i   0.0000 - 1.0000i

DFT matrix: (4*1)
   6.0000 + 0.0000i
  -2.0000 + 2.0000i
  -2.0000 - 0.0000i
  -2.0000 - 2.0000i
```

IDFT1 matrix:

0.0000 - 0.0000i

1.0000 - 0.0000i

2.0000 - 0.0000i

3.0000 + 0.0000i

IDFT2 matrix:

-0.0000 - 0.0000i

1.0000 + 0.0000i

2.0000 + 0.0000i

3.0000 - 0.0000i

Aim 3: Consider **Aim 1** and **Aim 2** and write "C" program to verify using Code Composer Studio (CCS-7.4) IDE and the DSP TMS 320C6713 DSK Kit.

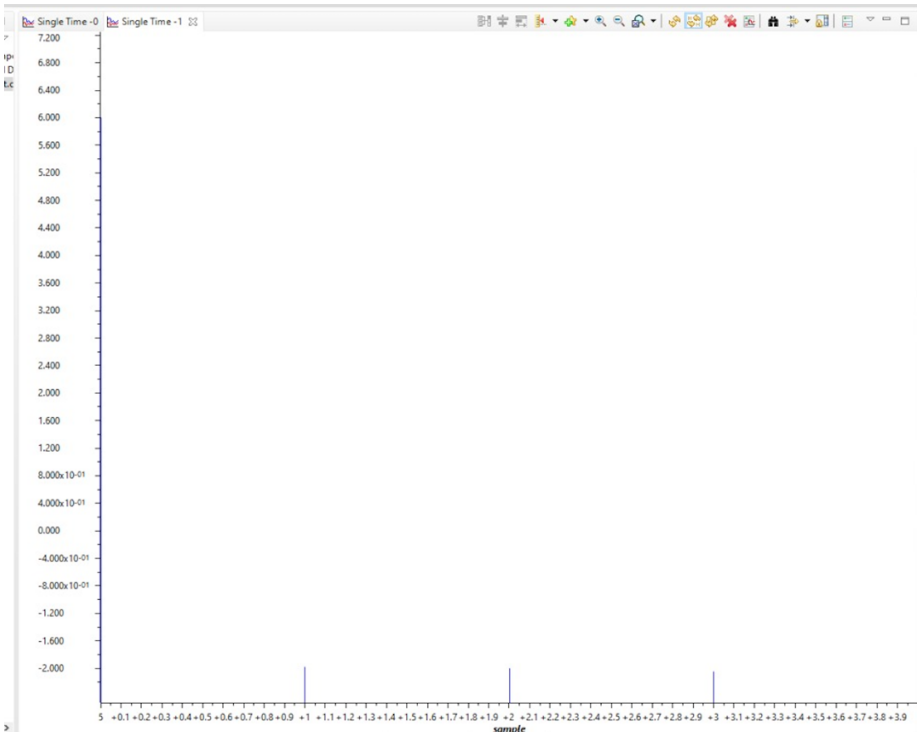
Code For DFT:

main.c exit.c

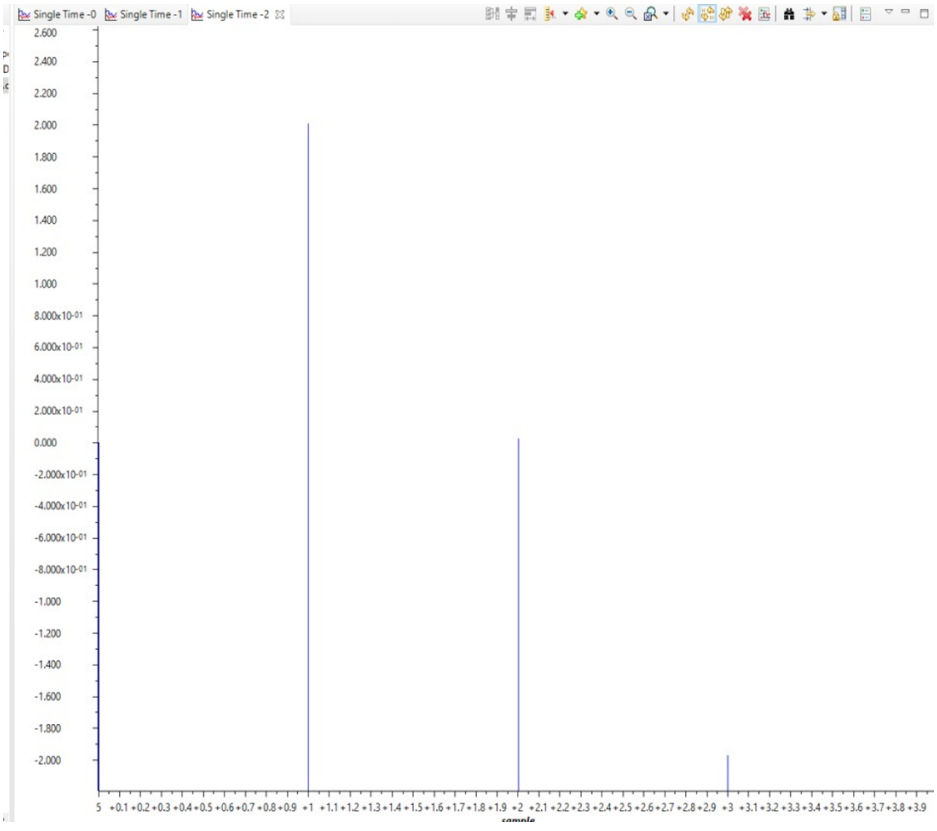
```c

/**
 * main.c
 */
#include<math.h>

float yReal[4] = {0,0,0,0} ;
float yImg[4] = {0,0,0,0} ;
int Xn[4] = {0,1,2,3} ;
int i , j;
float DnReal[4][4] ;
float DnImg[4][4] ;


int main(void)
{
    for(i = 0 ; i < 4 ; i++){
        for(j = 0 ; j < 4 ; j++){
            DnImg[i][j] = -1*sin((2*3.146*i*j)/4) ;
            DnReal[i][j] = cos((2*3.146*i*j)/4) ;
        }
    }



    for(i = 0 ; i < 4 ; i++){
        for(j = 0 ; j < 4 ; j++){
            yReal[i] += Xn[j] * DnReal[i][j] ;
            yImg[i] += Xn[j] * DnImg[i][j] ;
        }
    }




    return 0;
}

```

Output of Real and Imaginary Parts of DFT:

Real part of x[k]:



Imaginary part of x[k]:

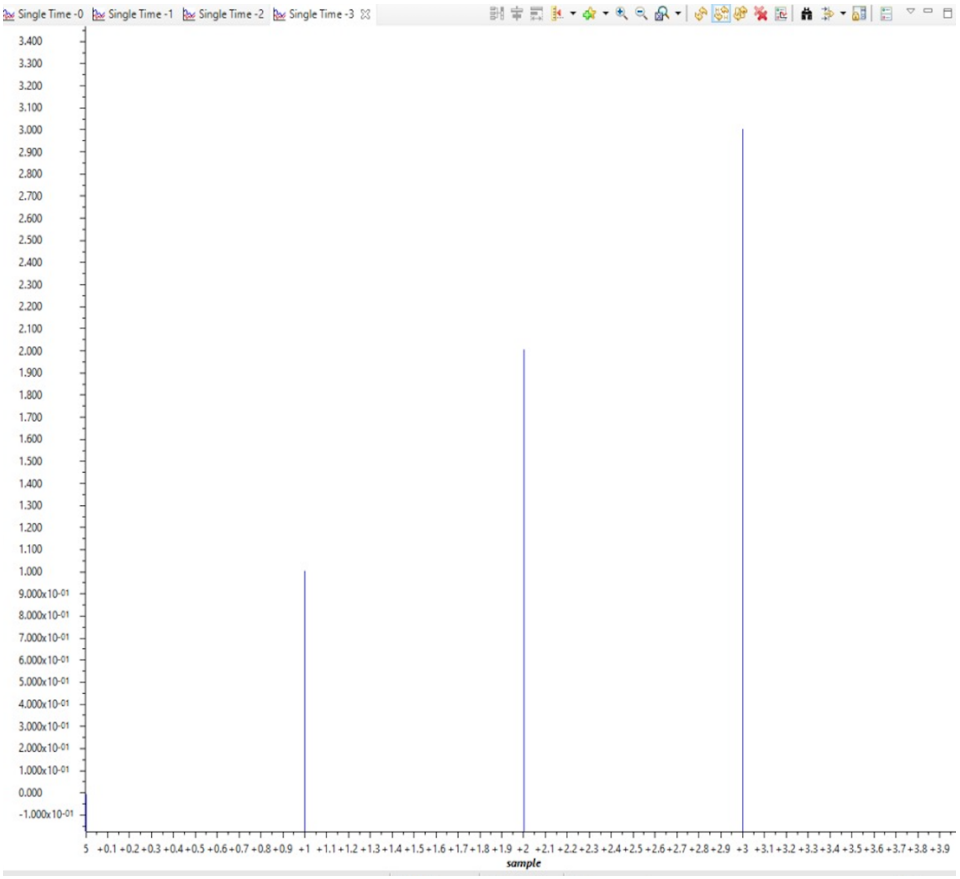


C Code for IDFT:

```c

/**
 * main.c
 */
#include<math.h>

float yReal[4] = {0,0,0,0} ;
float yImg[4] = {0,0,0,0} ;
int Xn[4] = {0,1,2,3} ;
int i , j;
float DnReal[4][4] ;
float DnImg[4][4] ;
float xReal[4] = {0,0,0,0} ;
float xImg[4] = {0,0,0,0} ;

int main(void)
{
    for(i = 0 ; i < 4 ; i++){
        for(j = 0 ; j < 4 ; j++){
            DnImg[i][j] = -1*sin((2*3.146*i*j)/4) ;
            DnReal[i][j] = cos((2*3.146*i*j)/4) ;
        }
    }

    for(i = 0 ; i < 4 ; i++){
        for(j = 0 ; j < 4 ; j++){
            yReal[i] += Xn[j] * DnReal[i][j] ;
            yImg[i] += Xn[j] * DnImg[i][j] ;
        }
    }

    for(i = 0 ; i < 4 ; i++){
        for(j = 0 ; j < 4 ; j++){
            DnImg[i][j] = -0.25*DnImg[i][j] ;
            DnReal[i][j] = 0.25*DnReal[i][j] ;
        }
    }

    for(i = 0 ; i < 4 ; i++){
        for(j = 0 ; j < 4 ; j++){
                xReal[i] += (yReal[j] * DnReal[i][j] -yImg[j]*DnImg[i][j])
                xImg[i] += (yImg[j] * DnReal[i][j] + yReal[j]*DnImg[i][j])
        }
    }




    return 0;
}
```

Output of IDFT:

Real Part of x(n):



Imaginary Part of x(n):