Table of Contents

Lab-6: Discrete Fourier Transform

Name: Himanshu Sharma **Roll Number**: 1610110149 **Email**: hs583@snu.edu.in **Instructor**: Prof. Vijay K. Chakka


```
% define main namespace
function a = main()
```

IDFT Matrix Generator

```
function IDFTM = IDFT_Matrix(N)
   % initialise an empty matrix
   IDFTM = [];

   % iterate over rows and a nested iteration on columns.
   for k=1:N
      % initialise an empty row.
      row = [];
      % fill this row in N iterations
      for n=1:N
           row = [row exp(j*(2*pi/N)*(k-1)*(n-1))];
      end
      % add this row to the matrix
      IDFTM = [IDFTM; row];
   end
end
```

Question 2

```
function OutMatrix = question2(IDFT_Matrix)
    OutMatrix = [];
    % iterate over rows.
    for row=1:length(IDFT_Matrix)
        % initialize a column vector
```

```
            cvector = [];
            % compute the conjugate of this row.
            current_row = IDFT_Matrix(row,:);
            conjugate_row = conj(current_row);
            % multiply this conjugate row to the DFT Matrix.
            cvector = conjugate_row*IDFT_Matrix;
            % append the row vector
            OutMatrix = [OutMatrix; cvector];
        end
        % take the overall transpose now since we have a collection as
row vectors.
        OutMatrix = transpose(OutMatrix);
    end
```

Verification of $DD^H = NI$

```
    function OutMatrix = question3(N)
        % Compute the N pt IDFT Matrix.
        idftMatrix = IDFT_Matrix(N);
        % Take the hermation of the DFT Matrix
        dftMatrix = idftMatrix';
        % Display the results.
        disp(idftMatrix*dftMatrix);
    end
```

DFT Coefficients Generator

```
    function coeffs = DFT_Coeffs(signal_seq)
        % this function takes in a signal sequence and then
        % computes its DFT coefficients.

        % using the first function of this code, compute the IDFT
Matrix.
        N = length(signal_seq);
        idftm = IDFT_Matrix(N);

        % take the hermition of this dft Matrix.
        dftm = idftm';

        % take the transpose of the signal_seq, because its an array
        % and we need a column vector.
        signal_seq = transpose(signal_seq);

        % calculate the dft coeff.
        coeffs = dftm*signal_seq;
    end
```

IDFT Coefficients Generator

```
    function signal_coeffs = IDFT_Coeffs(dft_coeffs)
        % calculate IDFT Matrix from first function.
```

```
    N = length(dft_coeffs);
    idftm = IDFT_Matrix(N);

    % take transpose of dft_coeffs, as it is at present an array.
    dft_coeffs = transpose(dft_coeffs);

    signal_coeffs = idftm*dft_coeffs;
    % normalise the signal coefficients by 1/N.
    signal_coeffs = (1/N)*signal_coeffs;
end
```

Linear Convolution

```
function y = linear_convolution(x, h)
    % first construct a matrix of length(x) X length(x)+length(h)-1
    y = [];
    % add zeros to the end of h.
    zeros_needed = length(x)-1;
    h = [h zeros(1, zeros_needed)];

    h_matrix = [];
    for row = 1:length(x)
        h_matrix = [h_matrix; h];
        h = circshift(h, 1);
    end
    % take the transpose of h_matrix.
    h_matrix = transpose(h_matrix);

    % multiply the input with h_matrix to get the output.
    y = h_matrix*transpose(x);
    y = transpose(y);
end
```

Question 5: $x(n) = \{1, 2, 3, 4\}$ and $h(n) = \{0, 1, 0, 0\}$

```
function question5(x, h)

    % dft coeff of x(n) and h(n).
    X = DFT_Coeffs(x);
    H = DFT_Coeffs(h);

    % calculate Y(k).
    Y = X.*H;

    % calculate the linear convolution output.
    ylinear = linear_convolution(x, h);

    % calculate the 4 pt IDFT of Y(k).
    y = IDFT_Coeffs(transpose(Y));

    % display the results.
    disp('4 pt. DFT of x(n)');
```

```
    disp(X);

    disp('4 pt. DFT of h(n)');
    disp(H);

    disp('Output of 4 pt IDFT of X(k)H(k)');
    disp(y);

    figure;
    subplot(4,1,1);
    stem(x);
    title('$$x(n)=\{1,2,3,4\}$$', 'interpreter', 'latex');
    xlim([0, 5]);
    grid on;
    subplot(4,1,2);
    stem(h);
    xlim([0, 5]);
    title('$$h(n)=\{0,1,0,0\}$$', 'interpreter', 'latex');
    grid on;
    subplot(4,1,3);
    stem(ylinear);
    title('$$y(n)=x(n)*h(n)$$', 'interpreter', 'latex');
    xlim([0, 8]);
    grid on;
    subplot(4,1,4);
    stem(real(y));
    xlim([0, 5]);
    title('4 pt. IDFT');
    grid on;
end
```

Question 6

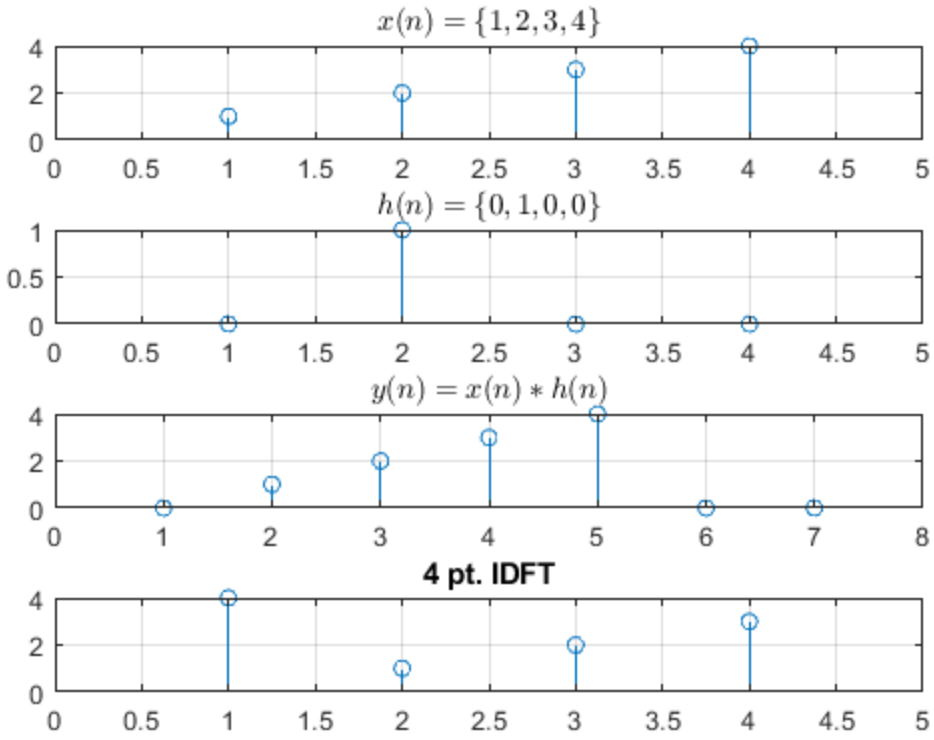
```
function question6()
    % load the input file.
    inp = load('inputData.mat', '-mat');
    inp = inp.inputData;

    % load the h1 impulse file.
    h1 = load('h1.mat', '-mat');
    h1 = h1.h1;

    % load the h2 impulse file.
    h2 = load('h2.mat', '-mat');
    h2 = h2.h2;

    % compute the DFT coefficients of input file.
    input_coeffs = DFT_Coeffs(inp);

    % compute the DFT coefficients of the h1 file.
    h1_coeffs = DFT_Coeffs(h1);

    % compute the DFT Coefficients of the h2 file.
```

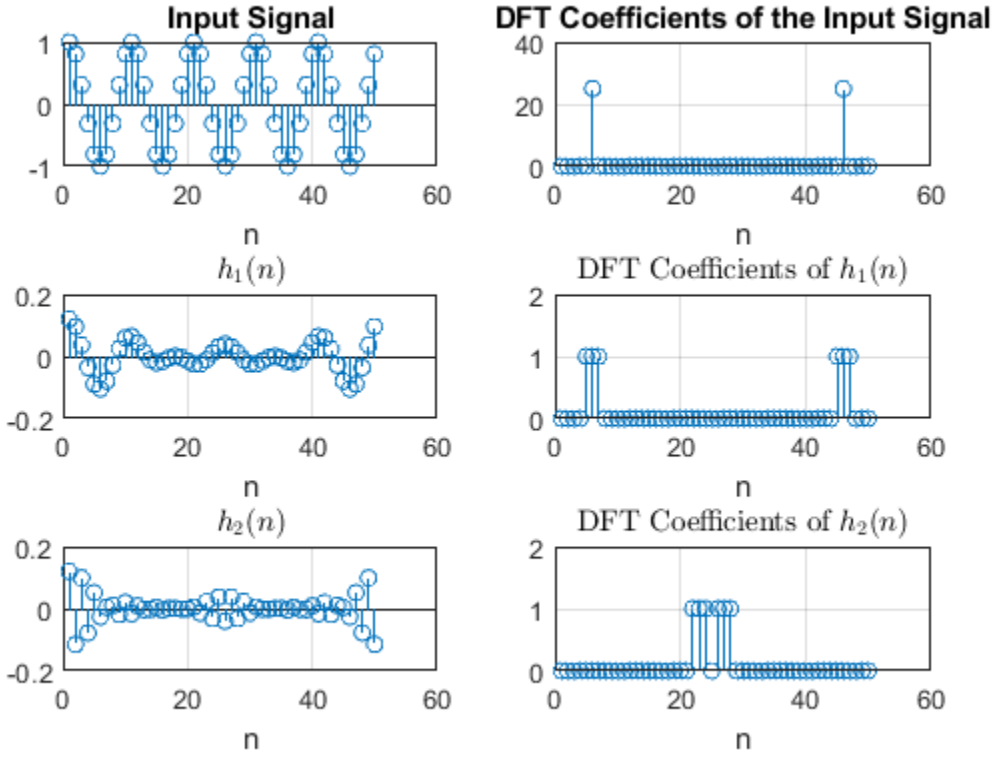
```
        h2_coeffs = DFT_Coeffs(h2);

        % Compute Y1[K] now.
        Y1 = input_coeffs.*h1_coeffs;

        % compute Y2[k].
        Y2 = input_coeffs.*h2_coeffs;

        % compute the inverse DFT coefficients of Y1.
        y1 = IDFT_Coeffs(transpose(Y1));
        % compute the IDFT coefficients for Y2.
        y2 = IDFT_Coeffs(transpose(Y2));

        figure;
        subplot(3,2,1);
        stem(inp);
        xlabel('n');
        title('Input Signal');
        grid on;
        subplot(3,2,2);
        stem(real(input_coeffs));
        title('DFT Coefficients of the Input Signal');
        xlabel('n');
        grid on;
        subplot(3,2,3);
        stem(h1);
        xlabel('n');
        title('$$h_{1}(n)$$', 'interpreter', 'latex');
        grid on;
        subplot(3,2,4);
        stem(real(h1_coeffs));
        xlabel('n');
        title('DFT Coefficients of $$h_{1}(n)$
$', 'interpreter', 'latex');
        grid on;
        subplot(3,2,5);
        stem(real(h2));
        xlabel('n');
        title('$$h_{2}(n)$$', 'interpreter', 'latex');
        grid on;
        subplot(3,2,6);
        stem(real(h2_coeffs));
        xlabel('n');
        title('DFT Coefficients of $$h_{2}(n)$
$', 'interpreter', 'latex');
        grid on;

        % figure for the outputs.
        figure;
        subplot(2,2,1);
        stem(real(Y1));
        xlabel('n');
        title('DFT Coefficients of $$y_{1}(n)$
$', 'interpreter', 'latex');
```

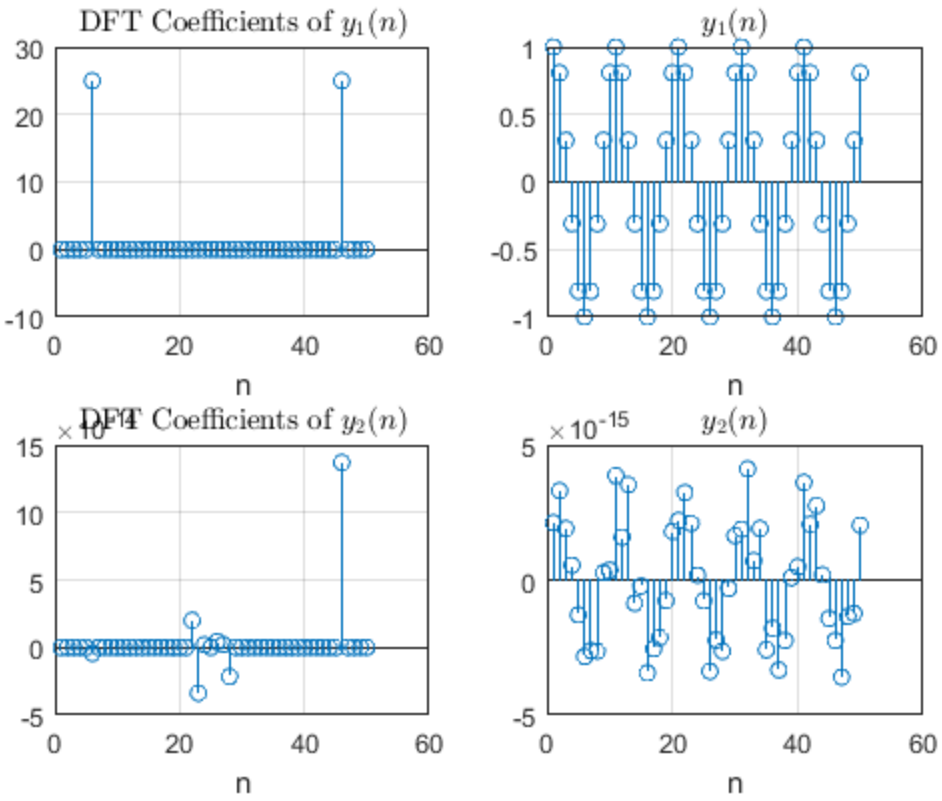
```
        grid on;
        subplot(2,2,2);
        stem(real(y1));
        xlabel('n');
        title('$$y_{1}(n)$$', 'interpreter', 'latex');
        grid on;
        subplot(2,2,3);
        stem(real(Y2));
        xlabel('n');
        title('DFT Coefficients of $$y_{2}(n)$
$', 'interpreter', 'latex');
        grid on;
        subplot(2,2,4);
        stem(real(y2));
        xlabel('n');
        title('$$y_{2}(n)$$', 'interpreter', 'latex');
        grid on;

    end
```

Code Application

```
    clc;
    dft = IDFT_Matrix(6);
    disp('A 6 pt IDFT Matrix');
    disp(dft);
    out = question2(dft);
    disp('Verification of Question 2 on the same 6 pt IDFT matrix
 calculated above.');
    disp(out);
    disp('Verification of Question 3 for a 5 pt DFT/IDFT Matrices.');
    question3(5);
    disp('Question 4 begins here.');
    disp('DFT Coefficients calculation.');
    disp('Input Sequence');
    disp([1,0,0,0,0,0.9]);
    coeffs_dft = DFT_Coeffs([1,0,0,0,0,0.9]);
    disp('DFT Coefficients');
    disp(coeffs_dft);
    disp('Calculation of signal coefficients for the above dft
 coefficients.');
    signal_coeffs = IDFT_Coeffs(transpose(coeffs_dft));
    disp(signal_coeffs);
    disp('Question5 begins here. Output Response using simple linear
 convolution. This is a. part.');
    disp(linear_convolution([1,2,3,4], [0,1,0,0]));
    disp('This is b. part now.');
    question5([1,2,3,4], [0,1,0,0]);
    question6();
```


$x(n) = \{1, 2, 3, 4\}$
$h(n) = \{0, 1, 0, 0\}$
$y(n) = x(n) * h(n)$
4 pt. IDFT


Input Signal
DFT Coefficients of the Input Signal
n
$h_1(n)$
DFT Coefficients of $h_1(n)$
$h_2(n)$
DFT Coefficients of $h_2(n)$




```
end
```

```
A 6 pt IDFT Matrix
  Columns 1 through 4

   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 +
 0.0000i
   1.0000 + 0.0000i   0.5000 + 0.8660i  -0.5000 + 0.8660i  -1.0000 +
 0.0000i
   1.0000 + 0.0000i  -0.5000 + 0.8660i  -0.5000 - 0.8660i   1.0000 -
 0.0000i
   1.0000 + 0.0000i  -1.0000 + 0.0000i   1.0000 - 0.0000i  -1.0000 +
 0.0000i
   1.0000 + 0.0000i  -0.5000 - 0.8660i  -0.5000 + 0.8660i   1.0000 -
 0.0000i
   1.0000 + 0.0000i   0.5000 - 0.8660i  -0.5000 - 0.8660i  -1.0000 +
 0.0000i

  Columns 5 through 6

   1.0000 + 0.0000i   1.0000 + 0.0000i
  -0.5000 - 0.8660i   0.5000 - 0.8660i
  -0.5000 + 0.8660i  -0.5000 - 0.8660i
   1.0000 - 0.0000i  -1.0000 + 0.0000i
  -0.5000 - 0.8660i  -0.5000 + 0.8660i
  -0.5000 + 0.8660i   0.5000 + 0.8660i
```

```
Verification of Question 2 on the same 6 pt IDFT matrix calculated
 above.
  Columns 1 through 4

   6.0000 + 0.0000i  -0.0000 + 0.0000i  -0.0000 - 0.0000i   0.0000 -
 0.0000i
  -0.0000 - 0.0000i   6.0000 + 0.0000i  -0.0000 + 0.0000i   0.0000 +
 0.0000i
  -0.0000 + 0.0000i  -0.0000 - 0.0000i   6.0000 + 0.0000i   0.0000 -
 0.0000i
   0.0000 + 0.0000i  -0.0000 + 0.0000i  -0.0000 - 0.0000i   6.0000 -
 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i  -0.0000 + 0.0000i  -0.0000 -
 0.0000i
   0.0000 + 0.0000i  -0.0000 + 0.0000i   0.0000 - 0.0000i   0.0000 +
 0.0000i

  Columns 5 through 6

   0.0000 - 0.0000i   0.0000 - 0.0000i
   0.0000 - 0.0000i  -0.0000 - 0.0000i
  -0.0000 - 0.0000i   0.0000 + 0.0000i
  -0.0000 - 0.0000i   0.0000 - 0.0000i
   6.0000 + 0.0000i   0.0000 - 0.0000i
   0.0000 + 0.0000i   6.0000 + 0.0000i

Verification of Question 3 for a 5 pt DFT/IDFT Matrices.
  Columns 1 through 4

   5.0000 + 0.0000i  -0.0000 - 0.0000i   0.0000 - 0.0000i   0.0000 -
 0.0000i
  -0.0000 + 0.0000i   5.0000 + 0.0000i  -0.0000 + 0.0000i   0.0000 -
 0.0000i
   0.0000 + 0.0000i  -0.0000 - 0.0000i   5.0000 + 0.0000i   0.0000 -
 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   5.0000 +
 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i  -0.0000 + 0.0000i  -0.0000 +
 0.0000i

  Column 5

   0.0000 - 0.0000i
  -0.0000 - 0.0000i
  -0.0000 - 0.0000i
  -0.0000 + 0.0000i
   5.0000 + 0.0000i

Question 4 begins here.
DFT Coefficients calculation.
Input Sequence
    1.0000         0         0         0         0    0.9000
```

```
DFT Coefficients
   1.9000 + 0.0000i
   1.4500 + 0.7794i
   0.5500 + 0.7794i
   0.1000 - 0.0000i
   0.5500 - 0.7794i
   1.4500 - 0.7794i

Calculation of signal coefficients for the above dft coefficients.
   1.0000 - 0.0000i
  -0.0000 - 0.0000i
  -0.0000 + 0.0000i
   0.0000 - 0.0000i
   0.0000 - 0.0000i
   0.9000 + 0.0000i

Question5 begins here. Output Response using simple linear
 convolution. This is a. part.
     0     1     2     3     4     0     0

This is b. part now.
4 pt. DFT of x(n)
  10.0000 + 0.0000i
  -2.0000 + 2.0000i
  -2.0000 - 0.0000i
  -2.0000 - 2.0000i

4 pt. DFT of h(n)
   1.0000 + 0.0000i
   0.0000 - 1.0000i
  -1.0000 - 0.0000i
  -0.0000 + 1.0000i

Output of 4 pt IDFT of X(k)H(k)
   4.0000 + 0.0000i
   1.0000 - 0.0000i
   2.0000 - 0.0000i
   3.0000 + 0.0000i
```

Published with MATLAB® R2017b