ASSIGNMENT - 2

21AIE212-MATHEMATICS FOR INTELLIGENT SYSTEMS-4

BATCH - A, GROUP - 1

GROUP MEMBERS:

R SRIVISWA, CB.EN.U4AIE21046
AMAN SIROHI, CB.EN.U4AIE21003
VIKHYAT BANSAL, CB.EN.U4AIE21076
RAKHIL ML, CB.EN.U4AIE21048

श्रद्धावान् लभते ज्ञानम्

AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

1. How the Eigenvalues and Eigenvectors of shift and circulant matrices are connected to each other.

ANS:

With N = 4, the equation $Px = \lambda x$ leads directly to four eigenvalues and eigenvectors:

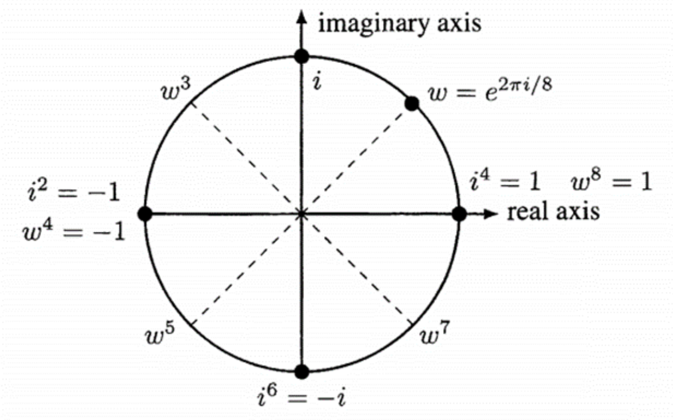
$$Px = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ x_4 \\ x_1 \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \quad \textbf{gives} \quad \begin{matrix} x_2 = \lambda x_1 \\ x_3 = \lambda x_2 \\ x_4 = \lambda x_3 \\ x_1 = \lambda x_4 \end{matrix}$$

Start with the last equation $x_1 = \lambda x_4$ and work upwards :

$X_1 = \lambda x_4 = \lambda^2 x_3 = \lambda^3 x_2 = \lambda^4 x$ leading to $\lambda^4 = 1_1$

The eigenvalues of P are the fourth roots of 1. They are all the powers i, i^2, i^3, 1 of $\omega = i$

The eigenvalues **i**, **-1**, **-i**, **1** are ***equally spaced*** around the unit circle in the complex plane.



All the four eigenvalues will **add to zero (**0**).**

$$\boxed{\textbf{The solutions to } z^N = 1 \textbf{ are } \lambda = w, w^2, \ldots, w^{N-1}, 1 \textbf{ with } w = e^{2\pi i/N}.}$$

In the complex plane, the first eigenvalue **ω** is $e^{i\Theta}$ = cos**Θ** + i sin**Θ** and the angle **Θ** is 2***π***/N.

The angles for the other eigenvalues are 2**Θ**, 3**Θ**, ., N**Θ**. Since **Θ** is 2***π***/N, that last angle is N**Θ** = 2***π*** and that eigenvalue is **λ** = $e^{2\pi i}$ which is cos 2***π*** + i sin 2***π*** = 1.

Knowing the N eigenvalues **λ** = 1, **ω**, …. , **ω**N-1 of PN, we quickly find N eigenvectors. Set the first component of q to 1. The other components of q are **λ** and **λ**2 and **λ**3 :

Eigenvectors for $\boldsymbol{\lambda = 1, i, i^2, i^3}$

$$q_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad q_1 = \begin{bmatrix} 1 \\ i \\ i^2 \\ i^3 \end{bmatrix} \quad q_2 = \begin{bmatrix} 1 \\ i^2 \\ i^4 \\ i^6 \end{bmatrix} \quad q_3 = \begin{bmatrix} 1 \\ i^3 \\ i^6 \\ i^9 \end{bmatrix}$$

Eigenvector matrix
$N = 4$
Fourier matrix

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & i^2 & i^3 \\ 1 & i^2 & i^4 & i^6 \\ 1 & i^3 & i^6 & i^9 \end{bmatrix}$$

Fourier Matrix (made up of Eigenvectors of P as Columns)

$$F_N = \begin{bmatrix} 1 & 1 & 1 & . & 1 \\ 1 & \omega & \omega^2 & . & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & . & \omega^{2(N-1)} \\ . & . & . & . & . \\ . & . & . & . & . \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & . & \omega^{(N-1)(N-1)} \end{bmatrix}$$

EIGENVALUES AND EIGENVECTORS OF CIRCULANT MATRIX

The eigenvectors of a ***circulant matrix* C** can be computed quite easily. **Those eigenvectors are the same as the eigenvectors of the permutation/shift P.** So they are the columns $\mathbf{q_0}$ **,** $\mathbf{q_1}$**, ... ,** $\mathbf{q_{N-1}}$ of the same Fourier matrix **F.** Here is $\mathbf{Cq_k = \lambda\, q_k}$ for the $\mathbf{k^{th}}$ eigenvector and eigenvalue :

$$\left(c_0 I + c_1 P + \cdots + c_{N-1} P^{N-1}\right) \boldsymbol{q}_k = \left(c_0 + c_1 \lambda_k + \cdots + c_{N-1} \lambda_k^{N-1}\right) \boldsymbol{q}_k .$$

$$\begin{bmatrix} \lambda_0(C) \\ \lambda_1(C) \\ \lambda_2(C) \\ \cdot \\ \cdot \\ \lambda_{N-1}(C) \end{bmatrix} = \begin{bmatrix} c_0 + c_1 + \ldots + c_{N-1} \\ c_0 + c_1\omega + \ldots + c_{N-1}\,\omega^{N-1} \\ c_0 + c_1\omega^2 + \ldots + c_{N-1}\,\omega^{2(N-1)} \\ \cdot \\ \cdot \\ c_0 + c_1\omega^{N-1} + \ldots + c_{N-1}\omega^{(N-1)(N-1)} \end{bmatrix} = F \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \cdot \\ \cdot \\ c_{N-1} \end{bmatrix} = Fc$$

$$\boxed{\textbf{The } N \textbf{ eigenvalues of } C \textbf{ are the components of } F\boldsymbol{c} = \textbf{inverse Fourier transform of } \boldsymbol{c}.}$$

For any N, the permutation P is a circulant matrix C with $\boldsymbol{c} = (0, 1, 0, \ldots, 0)$.
The eigenvalues of P are in the column vector $F\boldsymbol{c}$ with this $\boldsymbol{c}$.
That is the column $(1, w, w^2, \ldots, w^{N-1})$ of the Fourier matrix F.
This agrees with the eigenvalues $1, i, i^2, i^3$ of P in equation (8), for $N = 4$.

CODE:

```
% Define the first row of the circulant matrix
c = [1 2 3 4];

% Construct the shift matrix with first row [0 1 0 ... 0]
shift_mat = diag(ones(1, length(c) - 1), -1);
shift_mat(end, :) = [0 ones(1, length(c) - 1)];


% Compute the eigenvalues and eigenvectors of the shift
matrix
[V_shift, D_shift] = eig(shift_mat);


% Compute the Fourier matrix
F = fft(eye(length(c)));
```

```
% Compute the eigenvectors of the circulant matrix by
multiplying the
% eigenvectors of the shift matrix by the Fourier matrix
V_circ = F * V_shift;

% Compute the eigenvalues of the circulant matrix by
raising the eigenvalues
% of the shift matrix to the appropriate powers
lam_shift = diag(D_shift);
n = length(c);
lam_circ = zeros(n, 1);
for k = 1:n
lam_circ(k) = lam_shift(1)^((k-1) / n);
end

% Display the eigenvalues and eigenvectors
disp('Eigenvalues:');
disp(lam_circ);
disp('Eigenvectors:');
disp(V_circ);
```

OUTPUT:

```
Eigenvalues:

  10.0000 + 0.0000i  -2.0000 + 2.0000i  -2.0000 + 0.0000i  -2.0000 - 2.0000i


Eigenvectors:

  -0.5000 + 0.0000i  -0.5000 - 0.0000i  -0.5000 + 0.0000i  -0.5000 + 0.0000i
  -0.5000 + 0.0000i   0.0000 - 0.5000i   0.0000 + 0.5000i   0.5000 + 0.0000i
  -0.5000 + 0.0000i   0.5000 + 0.0000i   0.5000 + 0.0000i  -0.5000 + 0.0000i
  -0.5000 + 0.0000i  -0.0000 + 0.5000i  -0.0000 - 0.5000i   0.5000 + 0.0000i
```

2. How the Fourier matrix F and DFT matrix Ω are connected to each other.

The matrices F and Ω have the same columns. So F and Ω are connected by a permutation matrix. That permutation P leaves the zeroth columns alone: the column of 1's in both matrices. Then P exchanges the next column (1, w, w^2 ,...,w^{N-1}) of F for its last column (1, w, w^2 ,...,w^{N-1}). After the 1 in its zeroth and column, P contains the reverse identity matrix J (with 1's on the antidiagonal):

$$\boldsymbol{P} = \begin{bmatrix} \mathbf{1} & 0 \\ 0 & \boldsymbol{J} \end{bmatrix} = \begin{bmatrix} \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & \mathbf{1} & 0 \\ 0 & \mathbf{1} & 0 & 0 \end{bmatrix} \qquad \begin{aligned} &\boldsymbol{P^2 = I} \\ &\boldsymbol{\Omega = FP} \\ &\boldsymbol{\Omega P = FP^2 = F} \end{aligned}$$

Here are the full matrices for Ω = FP when N = 4 :

$$\boldsymbol{\Omega} = \begin{bmatrix} 1 & \mathbf{1} & 1 & 1 \\ 1 & \boldsymbol{-i} & (-i)^2 & (-i)^3 \\ 1 & \boldsymbol{(-i)^2} & (-i)^4 & (-i)^6 \\ 1 & \boldsymbol{(-i)^3} & (-i)^6 & (-i)^9 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \mathbf{1} \\ 1 & i & i^2 & \boldsymbol{i^3} \\ 1 & i^2 & i^4 & \boldsymbol{i^6} \\ 1 & i^3 & i^6 & \boldsymbol{i^9} \end{bmatrix} \begin{bmatrix} \mathbf{1} & & & \\ & & & \mathbf{1} \\ & & \mathbf{1} & \\ & \mathbf{1} & & \end{bmatrix} \qquad \boxed{\begin{aligned} 1 &= 1 \\ -i &= i^3 \\ (-i)^2 &= i^2 \\ (-i)^3 &= i \end{aligned}}$$

These matrix identities lead to the remarkable fact that $F^4 = \Omega^4 = N^2 I$. Four transforms bring back the original vector (times N^2). Just combine FΩ = NI and FP = Ω with P^2 = I :

$$\boldsymbol{F^2 P = F\Omega = NI} \quad \text{so} \quad \boldsymbol{PF^2 = NI} \quad \text{and} \quad \boldsymbol{F^4 = F^2 PPF^2 = N^2 I}$$

From $F^4 = N^2 I$, it follows that the Fourier matrix F and the DFT matrix Ω have only four possible eigenvalues! They are the numbers λ = √N and i√N and -√N and -i√N that solve $\lambda^4 = N^2$ For sizes N > 4 there must be and will be repeated λ's. The eigenvectors of F are not so easy to find.

Start with any N-dimensional vector $f=(f_0 ,\dots,f_{N-1})$. The Discrete Fourier Transform expresses ƒ as a combination of the Fourier basis vectors. Those basis vectors are the columns b (containing powers of w) in the Fourier matrix F_N :

$$\begin{bmatrix} f_0 \\ \vdots \\ f_{N-1} \end{bmatrix} = \begin{bmatrix} b_0 & \cdots & b_{N-1} \end{bmatrix} \begin{bmatrix} c_0 \\ \vdots \\ c_{N-1} \end{bmatrix}$$

$$\begin{aligned} \boldsymbol{f} &= F_N \boldsymbol{c} \\ \boldsymbol{c} &= F_N^{-1} \boldsymbol{f} \\ \boldsymbol{c} &= \frac{1}{N} \Omega_N \boldsymbol{f} \end{aligned}$$

The forward transform c = fft (f) multiplies f by the DFT matrix Ω (and divides by N). That is the analysis step, to separate f into N orthogonal pieces. The DFT finds the coefficients in a finite Fourier series.

The synthesis step is the inverse transform f = ifft (c) = Fc. It starts with those coefficients c=(c_0 ,...,c_{N-1}). It carries out the matrix-vector multiplication Fc to recover f.

Thus ifft (fft(f)) ≈ f.

Therefore, we can say that the DFT matrix Ω is a subset of the Fourier matrix F, with each column of Ω being a scaled version of a corresponding column of F.

CODE :

```
N = 4;
F = dftmtx(N)/sqrt(N);
omega = exp(-2*pi*1i/N); %DFT matrix Ω
% Compute the DFT matrix Ω_N
Omega_N = zeros(N);
for k = 0:N-1
    for n = 0:N-1
        Omega_N(k+1,n+1) = omega^(k*n);
    end
end
f = [1; 2; 3; 4]; %test vector f
% Computing the coefficients using the Fourier matrix
c = F*f;
```

```
% Computing the coefficients using the DFT matrix
d = (1/sqrt(N))*Omega_N*f;


% Comparing the coefficients
if isequal(round(c,8),round(d,8))
    disp('The coefficients are equal');
else
    disp('The coefficients are not equal');
end
disp('Coefficients c:');
disp(c);
disp('Coefficients d:');
disp(d);
```

OUTPUT :

```
The coefficients are equal

Coefficients c:
   5.0000 + 0.0000i
  -1.0000 + 1.0000i
  -1.0000 + 0.0000i
  -1.0000 - 1.0000i
Coefficients d:
   5.0000 + 0.0000i
  -1.0000 + 1.0000i
  -1.0000 - 0.0000i
  -1.0000 - 1.0000i
```

3. Obtain the Eigenvalues and Eigenvectors of a circulant matrix 'C' of size N without doing the Eigen decomposition. Assume 'c' as the first column of 'C'.

ANS:

The eigenvalues and eigenvectors can be derived with the help of Shift Matrix 'P' as the circulant Matrix C is just formed using the Linear Combination of the Shift Matrix 'P'.

We know that the Eigenvector matrix for the Shift Matrix is the **Fourier Matrix**. (Derived in Question 1)

Fourier matrix

Eigenvectors of P

$$F_N = \begin{bmatrix} 1 & 1 & 1 & \cdot & 1 \\ 1 & w & w^2 & \cdot & w^{N-1} \\ 1 & w^2 & w^4 & \cdot & w^{2(N-1)} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & w^{N-1} & w^{2(N-1)} & \cdot & w^{(N-1)(N-1)} \end{bmatrix}.$$

Therefore, the eigenvectors of a Circulant Matrix 'C' are the columns q_0 , q_1, ... , q_{N-1} of the same Fourier matrix F. We know the Eigen Relation for A (nonzero) vector **v** of dimension *N* is an eigenvector of a square *N* × *N* matrix **A** if it satisfies a linear equation of the form:

$$Av = \lambda v$$

Similarly, for a circulant matrix C the Eigen Relation can be given as:

$$C\mathbf{q_k} = \lambda \mathbf{q_k}$$

Therefore, for the kth Eigenvector and Eigenvalue, the relation can be given as:

$$(c_0 I + c_1 P + \cdots + c_{N-1} P^{N-1}) q_k = (c_0 + c_1 \lambda_k + \cdots + c_{N-1} \lambda_k^{N-1}) q_k.$$

We know that $\lambda^k=\omega^k=e^{2\pi ik/N}$ is the kth eigenvalue of P. Those numbers are in the Fourier matrix F. Then the eigenvalues of C in the above equation gives us the formula: **Multiply F times the vector c in the top row of C to find the eigenvalues.**

$$\begin{bmatrix} \lambda_0(C) \\ \lambda_1(C) \\ \lambda_2(C) \\ \cdot \\ \cdot \\ \lambda_{N-1}(C) \end{bmatrix} = \begin{bmatrix} c_0 + c_1 + \ldots. + c_{N-1} \\ c_0 + c_1\omega + \ldots. + c_{N-1}\,\omega^{N-1} \\ c_0 + c_1\omega^2 + \ldots. + c_{N-1}\,\omega^{2(N-1)} \\ \cdot \\ \cdot \\ c_0 + c_1\omega^{N-1} + \ldots. + c_{N-1}\omega^{(N-1)(N-1)} \end{bmatrix} = F \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \cdot \\ \cdot \\ c_{N-1} \end{bmatrix} = Fc$$

The N eigenvalues of C are components of Fc = Inverse Fourier Transform of C

For any N, the permutation P is a circulant matrix C with c = (0, 1, 0, ... , 0).
The eigenvalues of P are in the column vector Fc with this c.
That is the column $(1, \omega, \omega^2, \bullet.., \omega^{N-1})$ of the Fourier matrix F.

CODE:

```
C =[1 2 3 4];
[lam, V] = circulant_eig_decomp(c);
disp('Eigenvalues:');
disp(lam);
disp('Eigenvectors:');
disp(V);

function [lam, V] = circulant_eig_decomp(c)
% Compute the eigenvalues and eigenvectors of a circulant
matrix with first row c using eigen decomposition.

n = length(c);
% Construct the circulant matrix
C = zeros(n);
for i = 1:n
for j = 1:n
C(i,j) = c(mod(i-j,n)+1);
end
end
```

```
% Compute the eigenvalues and eigenvectors using eigen
decomposition
[V, D] = eig(C);
lam = diag(D);
% Return the eigenvalues and eigenvectors
end
```

COMPARING THE OUTPUTS

The code for computing the eigenvectors and eigenvalues of a Circular Matrix has been given in Q1. Comparing those results with the code for finding the eigenvalues and eigenvectors using Eigen decomposition.

```
Eigenvalues:

  10.0000 + 0.0000i
  -2.0000 + 2.0000i
  -2.0000 - 2.0000i
  -2.0000 + 0.0000i
```

```
Eigenvectors:

  -0.5000 + 0.0000i  -0.5000 - 0.0000i  -0.5000 + 0.0000i  -0.5000 + 0.0000i
  -0.5000 + 0.0000i   0.0000 - 0.5000i   0.0000 + 0.5000i   0.5000 + 0.0000i
  -0.5000 + 0.0000i   0.5000 + 0.0000i   0.5000 + 0.0000i  -0.5000 + 0.0000i
  -0.5000 + 0.0000i  -0.0000 + 0.5000i  -0.0000 - 0.5000i   0.5000 + 0.0000i
```

We can see that both the outputs are same.

To summarize,

1. Compute the Fourier transform of the first row of the circulant matrix to obtain its eigenvalues.
2. Compute the inverse Fourier transform of the vectors [1, ωk, ωk^2, ..., ωk^(n-1)] for each eigenvalue to obtain the corresponding eigenvectors.

4. DFT of 2-D data utilizing the concept of Kronecker product.

ANS:

We will construct 2–dimensional DFT matrix of size N2 using Fourier matrices F and Ω
This construction uses the Kronecker products F x F and Ω x Ω. The earlier word was tensor product. The MATLAB command is kron(F, F) and kron(Ω, Ω).

The first thing to know about Kronecker products is the size of $A \otimes B = \mathrm{kron}(A, B)$:

1 If A and B are n by n, then $\boldsymbol{A \otimes B}$ **is** $\boldsymbol{n^2}$ **by** $\boldsymbol{n^2}$.

2 If A is m by n and B is M by N, then $\boldsymbol{A \otimes B}$ **has** $\boldsymbol{mM}$ **rows and** $\boldsymbol{nN}$ **columns.**

The entries of $A \otimes B$ are (all mn entries of A) times (all MN entries of B).

The next fact is the position of those products in the large matrix. The rule is to **multiply each entry of A times the whole matrix B.** Then $A \otimes B$ is a block matrix. Every block is a multiple of B:

$$\boxed{\textbf{Kronecker product} \qquad \boldsymbol{A \otimes B} = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}.} \qquad (1)$$

2-D DFT

We intend to apply a 2D Discrete Fourier Transform to f. The result will be a 2D vector c.
You could think of this process in two steps:
Row by row: Apply the l-D DFT to each row of pixels separately.
Column by column: Rearrange the output by columns and transform each column.
The matrix for each step is N^2 by N^2.
First think of the N^2 pixels a row at a time and multiply each row in that long vector by the one-dimensional DFT matrix.

$$\Omega_{\text{row}} f = \begin{bmatrix} \Omega_N & & & \\ & \Omega_N & & \\ & & \cdot & \\ & & & \Omega_N \end{bmatrix} \begin{bmatrix} \text{row } 1 \\ \text{row } 2 \\ \text{row } 3 \\ \text{row } 4 \end{bmatrix} \qquad (f \text{ and } \Omega_{\text{row}} f \text{ have length } N^2)$$

That matrix is $\Omega_{\text{row}} = I_N \otimes \Omega_N$. It is a Kronecker product of size N^2.

Now the output $\Omega_{row}f$ is (mentally not electronically) rearranged into columns. The second step of the 2D transform multiplies each column of that "halfway" image $\Omega_{row}f$ by Ω_N . Again we are multiplying by a matrix Ω_{column} of size N^2.

The full 2D transform Is Ω_N x Ω_N.

That matrix Ω_{column} is the Kronecker product $\Omega_N \otimes I_N$.

The 2D transform puts the row and column steps together into $\Omega_{N \times N}$.

$$\Omega_{N \times N} = \Omega_{\text{column}}\, \Omega_{\text{row}} = (\Omega_N \otimes I_N)(I_N \otimes \Omega_N) = \Omega_N \otimes \Omega_N.$$

$$Let \quad Data \;\; \mathrm{X}= \begin{pmatrix} a_1 & a_2 & a_3 & a_4 \\ b_1 & b_2 & b_3 & b_4 \\ c_1 & c_2 & c_3 & c_4 \\ d_1 & d_2 & d_3 & d_4 \end{pmatrix} = \begin{pmatrix} -\boldsymbol{a}^T - \\ -\boldsymbol{b}^T - \\ -\boldsymbol{c}^T - \\ -\boldsymbol{d}^T - \end{pmatrix}$$

$2D$ DFT can be visualized as follows.

Let $\Omega_{4\times 4}$ 1D DFT transform matrix

Then we Apply DFT on each Row of X.

Let the output be as follows

$$\Omega \boldsymbol{a} = \hat{\boldsymbol{a}}\,; \quad \Omega \boldsymbol{b} = \hat{\boldsymbol{b}}\,; \; \Omega \boldsymbol{c} = \hat{\boldsymbol{c}}\,; \Omega \boldsymbol{d} = \hat{\boldsymbol{d}}\,;$$

Re*arrange back as* row vectors

$$\hat{\mathrm{X}}= \begin{pmatrix} -\hat{\boldsymbol{a}}^T - \\ -\hat{\boldsymbol{b}}^T - \\ -\hat{\boldsymbol{c}}^T - \\ -\hat{\boldsymbol{d}}^T - \end{pmatrix} = \begin{pmatrix} \hat{a}_1 & \hat{a}_2 & \hat{a}_3 & \hat{a}_4 \\ \hat{b}_1 & \hat{b}_2 & \hat{b}_3 & \hat{b}_4 \\ \hat{c}_1 & \hat{c}_2 & \hat{c}_3 & \hat{c}_4 \\ \hat{d}_1 & \hat{d}_2 & \hat{d}_3 & \hat{d}_4 \end{pmatrix}$$

$$Let \quad Data \quad \mathbf{X}=\begin{pmatrix} a_1 & a_2 & a_3 & a_4 \\ b_1 & b_2 & b_3 & b_4 \\ c_1 & c_2 & c_3 & c_4 \\ d_1 & d_2 & d_3 & d_4 \end{pmatrix} = \begin{pmatrix} -\boldsymbol{a}^T & - \\ -\boldsymbol{b}^T & - \\ -\boldsymbol{c}^T & - \\ -\boldsymbol{d}^T & - \end{pmatrix}$$

$$\hat{X}_{r\_vec} = \begin{pmatrix} \boldsymbol{\Omega}_4 & & & \\ & \boldsymbol{\Omega}_4 & & \\ & & \boldsymbol{\Omega}_4 & \\ & & & \boldsymbol{\Omega}_4 \end{pmatrix}_{16\times 16} \underbrace{\begin{pmatrix} \boldsymbol{a} \\ \boldsymbol{b} \\ \boldsymbol{c} \\ \boldsymbol{d} \end{pmatrix}}_{X_{r\_vec}} \rightarrow \begin{pmatrix} \hat{\boldsymbol{a}} \\ \hat{\boldsymbol{b}} \\ \hat{\boldsymbol{c}} \\ \hat{\boldsymbol{d}} \end{pmatrix}_{16\times 1} \quad ;$$

$$= \left(I_4 \otimes \boldsymbol{\Omega}_4\right) X_{r\_vec}$$

$$\begin{pmatrix} I & I & I & I \\ I & \omega I & \omega^2 I & \omega^3 I \\ I & \omega^2 I & \omega^4 I & \omega^6 I \\ I & \omega^3 I & \omega^6 I & \omega^9 I \end{pmatrix} \hat{X}_{r\_vec} = \begin{array}{c} \begin{matrix} & & & & & 5 & & & & 9 & & & & 13 & & & \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 1 \\ 2 \\ 3 \\ 4 \\ 1 \\ 2 \\ 3 \\ 4 \\ 1 \\ 2 \\ 3 \\ 4 \end{matrix} \begin{pmatrix} 1 & . & . & . & 1 & . & . & . & 1 & . & . & . & 1 & . & . & . \\ & 1 & & & & 1 & & & & 1 & & & & 1 & & \\ & & 1 & & & & 1 & & & & 1 & & & & 1 & \\ & & & 1 & & & & 1 & & & & 1 & & & & 1 \\ 1 & & & & \omega & & & & \omega^2 & & & & \omega^3 & & & \\ & 1 & & & & \omega & & & & \omega^2 & & & & \omega^3 & & \\ & & 1 & & & & \omega & & & & \omega^2 & & & & \omega^3 & \\ & & & 1 & & & & \omega & & & & \omega^2 & & & & \omega^3 \\ 1 & & & & \omega^2 & & & & \omega^4 & & & & \omega^6 & & & \\ & 1 & & & & \omega^2 & & & & \omega^4 & & & & \omega^6 & & \\ & & 1 & & & & \omega^2 & & & & \omega^4 & & & & \omega^6 & \\ & & & 1 & & & & \omega^2 & & & & \omega^4 & & & & \omega^6 \\ 1 & & & & \omega^3 & & & & \omega^6 & & & & \omega^9 & & & \\ & 1 & & & & \omega^3 & & & & \omega^6 & & & & \omega^9 & & \\ & & 1 & & & & \omega^3 & & & & \omega^6 & & & & \omega^9 & \\ & & & 1 & & & & \omega^3 & & & & \omega^6 & & & & \omega^9 \end{pmatrix} \end{array} \begin{pmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \\ \hat{a}_4 \\ \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \\ \hat{b}_4 \\ \hat{c}_1 \\ \hat{c}_2 \\ \hat{c}_3 \\ \hat{c}_4 \\ \hat{d}_1 \\ \hat{d}_2 \\ \hat{d}_3 \\ \hat{d}_4 \end{pmatrix}$$

$$\Omega_4 \otimes \Omega_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & (-i)^1 & (-i)^2 & (-i)^3 \\ 1 & (-i)^2 & (-i)^4 & (-i)^6 \\ 1 & (-i)^3 & (-i)^6 & (-i)^9 \end{pmatrix} \otimes \Omega_4 = \begin{pmatrix} \Omega_4 & \Omega_4 & \Omega_4 & \Omega_4 \\ \Omega_4 & (-i)\Omega_4 & (-i)^2\Omega_4 & (-i)^3\Omega_4 \\ \Omega_4 & (-i)^2\Omega_4 & (-i)^4\Omega_4 & (-i)^6\Omega_4 \\ \Omega_4 & (-i)^3\Omega_4 & (-i)^6\Omega_4 & (-i)^9\Omega_4 \end{pmatrix}$$

$Putting$ all together

$$\hat{X}_{r\_vec} = (I_4 \otimes \Omega_4) X_{r\_vec}$$

$$\tilde{\hat{X}}_{r\_vec} = (\Omega_4 \otimes I_4)\hat{X}_{r\_vec} = (\Omega_4 \otimes I_4)(I_4 \otimes \Omega_4) X_{r\_vec}$$

$$\tilde{\hat{X}}_{r\_vec} = (\Omega_4 \otimes \Omega_4) X_{r\_vec}$$

$$\tilde{\hat{X}} = \left(reshape\left(\tilde{\hat{X}}_{r\_vec}, 4{,}4 \right) \right)$$

For generic $\mathrm{N} \times N$ $matrix$ X

$$\tilde{\hat{X}}_{r\_vec} = (\Omega_N \otimes I_N)(I_N \otimes \Omega_N) X_{r\_vec}$$

$$\tilde{\hat{X}} = \left(reshape\left(\tilde{\hat{X}}_{r\_vec}, N, N \right) \right)^T$$

Code:

```
function [x_dd_matrix] = dft2d_kron(x)


% 2D DFT using Kronecker product
% x: input matrix
% coefficient: output matrix


x_vec = vec(transpose(x));  % convert the matrix into a vector
[M,N] = size(x);            % Getting the value of M x N
I_1 = eye(M);      % Creating identity matrix of size M an d N
I_2 = eye(N);
Omega_1 = dftmtx(M); % Creating DFT matrix of size M and N
Omega_2 = dftmtx(N);
```

```
kron_1 = kron(I_1,Omega_1); % Doing Kronecker product
kron_2 = kron(Omega_2,I_2);
x_vec_dd = kron_2*kron_1*x_vec; % DFT_coeff. after applying DFT
x_dd_matrix = transpose(reshape(x_vec_dd,M,N)); %Getting matrix
end


function [original_matrix] = idft2d_kron(y)

% 2D IDFT using Kronecker product
% coefficients: input matrix
% image: output matrix


y_vec = vec(transpose(y));  % convert the matrix into a vector
[M,N] = size(y);            % Getting the value of M x N
I_1 = eye(M);      % Creating identity matrix of size M an d N
I_2 = eye(N);
Omega_1 = dftmtx(M);  % Creating DFT matrix of size M and N
Omega_2 = dftmtx(N);
kron_1 = kron(I_1,Omega_1); % Doing Kronecker product
kron_2 = kron(Omega_2,I_2);
original_vec_matrix = inv((1/M*N)*(kron_2*kron_1))*y_vec;
original_matrix = transpose(reshape(original_vec_matrix,M,N));
end

% Load image
data_matrix = double(imread('doll.png'))

coefficients = dft2d_kron(data_matrix)
```

Output:

```
data_matrix = 96x96                                                          Rows 3:12 | Columns 44:62
     11    16    13    10    26    26    38    13    14    16     1    11    12     3     0     0     0     0     0
     59    65     7     7    72     7     7     7    50    46    55    26    13    16    11     3     5     0     0
      7    72     7     7    72     7     7    72     7     7    72     6    32    47    15    14     1     3     0
     72    72   125   115   151   102   112    72    72    72    72    72    82    82    50    47    15    14    11
    165   141   140   143   174   136   173   165   141   173   127   125     7     6    82     6    32    17    26
    122    72     7     7    72     7     7    72    72   107   180   165   173   151    82     6     6    45    47
    200   211   224   190   123   134   153   132   189   184   122    72   102   173   138   112    83    83    74
    181   206   179   169   193   193   235   204   192    93   106   146   151     7   102   141   212    83    78
    198    77    85    85    77    85    85    77    77   172   203   194    93   158   112    72   147   143   116
     77    77    77    77    77    77    77    77    77    77    77   168   179   134   134   199    83   127   146

coefficients = 96x96 complex
10^5  x
   4.4275 + 0.0000i  -2.6232 - 0.1343i  -0.0357 + 0.0170i   0.4787 + 0.0767i   0.0465 - 0.0046i  ...
  -0.6824 + 0.3574i   0.3758 + 0.1517i   0.0335 - 0.4482i   0.0779 - 0.1571i  -0.1573 + 0.3307i
  -0.4313 - 0.2394i   0.0760 + 0.1218i   0.3260 - 0.0661i  -0.1902 + 0.2005i  -0.0325 - 0.0634i
   0.1192 + 0.2496i  -0.0166 - 0.1776i  -0.0426 - 0.0126i  -0.0653 + 0.1290i   0.0584 - 0.0340i
  -0.2597 + 0.2808i   0.2451 - 0.0897i  -0.1391 - 0.1636i  -0.0468 + 0.1540i   0.1307 + 0.0138i
  -0.3293 + 0.2026i   0.1809 - 0.1098i   0.0777 - 0.0417i  -0.1070 + 0.0900i   0.0040 - 0.0332i
  -0.2964 + 0.0933i   0.2410 - 0.0909i  -0.1081 + 0.0722i   0.0089 + 0.0256i   0.0303 - 0.1061i
  -0.1267 + 0.0924i   0.1149 - 0.0619i  -0.0957 - 0.0008i   0.0751 + 0.0829i  -0.0478 - 0.1137i
  -0.1810 - 0.0141i   0.1161 + 0.0033i  -0.0111 + 0.0165i  -0.0386 - 0.0209i  -0.0242 + 0.0141i
  -0.1081 - 0.0047i   0.1026 - 0.0157i  -0.0397 + 0.0966i  -0.0043 - 0.1386i  -0.0115 + 0.0442i
       :
```

```
original_image = idft2d_kron(coefficients)
imshow(original_image)
```

Output:

```
original_image = 96x96 complex                                          Rows 1:10 | Columns 45:50
10^2  x
  -0.0000 + 0.0000i  -0.0000 - 0.0000i   0.0000 - 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0500 - 0.0000i   0.0500 + 0.0000i   0.0300 + 0.0000i   0.0300 + 0.0000i   0.0300 + 0.0000i   0.0300 + 0.0000i
   0.1600 - 0.0000i   0.1300 + 0.0000i   0.1000 + 0.0000i   0.2600 + 0.0000i   0.2600 - 0.0000i   0.3800 - 0.0000i
   0.6500 + 0.0000i   0.0700 + 0.0000i   0.0700 + 0.0000i   0.7200 - 0.0000i   0.0700 - 0.0000i   0.0700 - 0.0000i
   0.7200 + 0.0000i   0.0700 + 0.0000i   0.0700 - 0.0000i   0.7200 - 0.0000i   0.0700 + 0.0000i   0.0700 + 0.0000i
   0.7200 + 0.0000i   1.2500 - 0.0000i   1.1500 - 0.0000i   1.5100 + 0.0000i   1.0200 + 0.0000i   1.1200 - 0.0000i
   1.4100 + 0.0000i   1.4000 - 0.0000i   1.4300 + 0.0000i   1.7400 + 0.0000i   1.3600 + 0.0000i   1.7300 - 0.0000i
   0.7200 - 0.0000i   0.0700 + 0.0000i   0.0700 - 0.0000i   0.7200 + 0.0000i   0.0700 - 0.0000i   0.0700 - 0.0000i
   2.1100 - 0.0000i   2.2400 - 0.0000i   1.9000 + 0.0000i   1.2300 - 0.0000i   1.3400 + 0.0000i   1.5300 - 0.0000i
   2.0600 - 0.0000i   1.7900 - 0.0000i   1.6900 + 0.0000i   1.9300 + 0.0000i   1.9300 + 0.0000i   2.3500 - 0.0000i
```

THANK YOU !!