Chapter 3

Image Transform: Frequency Domain Representation and Enhancement

Dr. Bhakti Palkar

Syllabus

Image Transform: Frequency Domain Representation and Enhancement (10-CO3)

3.1 Introduction , DFT and its properties, radix-2 algorithm(2- DFT), FFT algorithm: divide and conquer approach, Dissemination in Time(DIT)-FFT

3.2 Discrete Cosine Transform, Walsh Transform, Hadamard Transform, Haar Transform, Principal component Analysis(PCA/Hoteling Transform), Introduction to Wavelet Transform

3.3 Low Pass and High Pass Frequency domain filters: Ideal, Butterworth, Homomorphic filter

Self-Learning Topic: Discrete Sine Transform (DST)

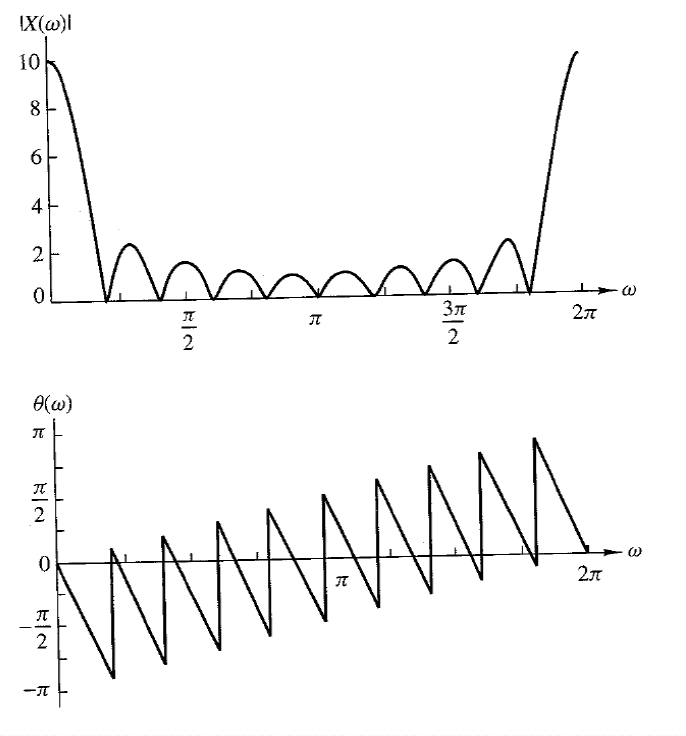
Discrete Fourier Transform

A finite duration sequence of length L is given by

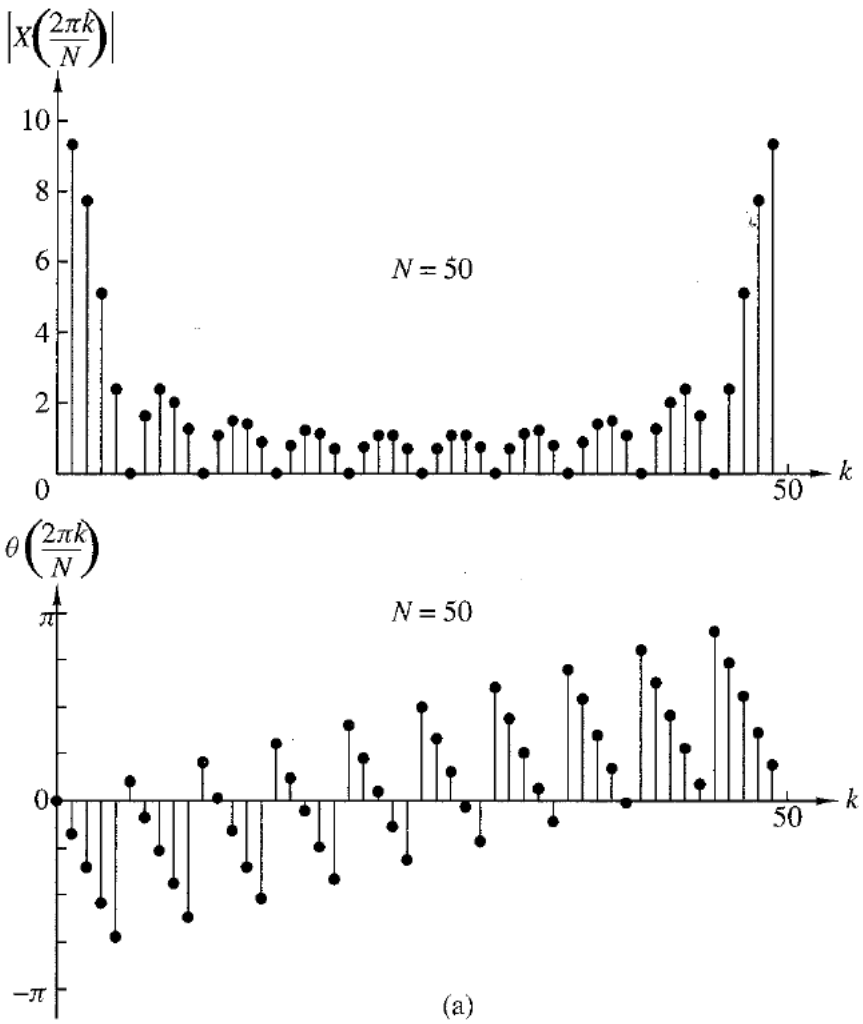
$$x(n) = \begin{cases} 1 & 0 < n < L-1 \\ 0 & otherwise \end{cases}$$

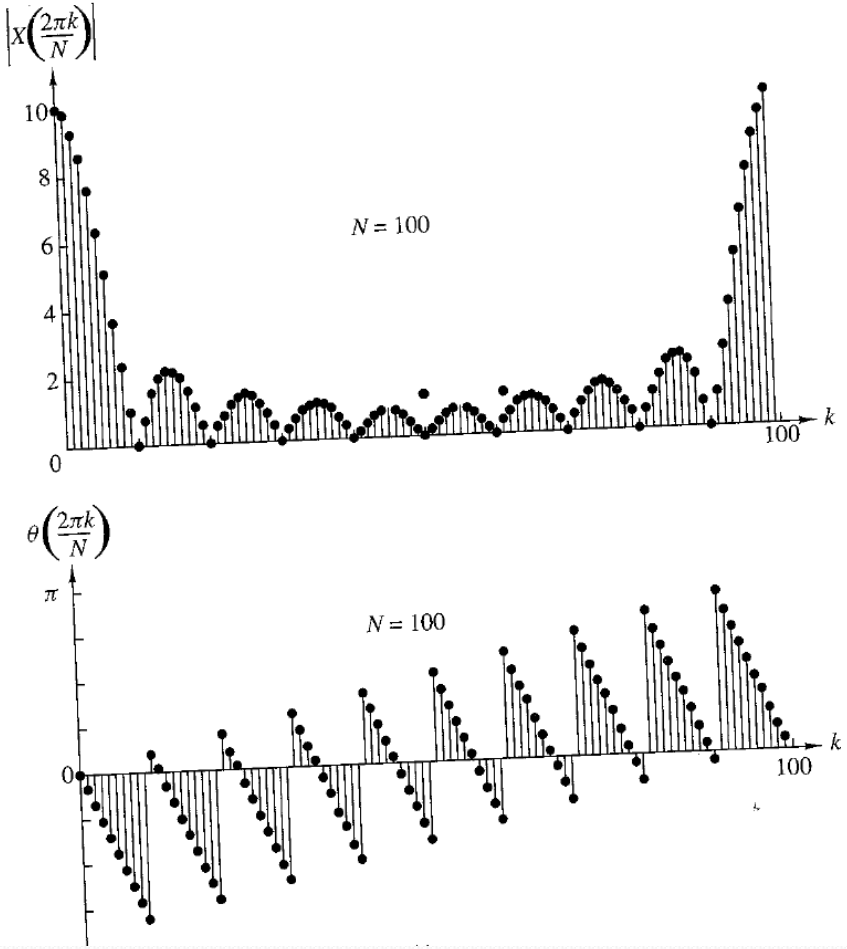
The DTFT was evaluated as

$$\frac{sin\,{\omega L}/{2}}{sin\omega} e^{-i\omega(L-1)/2}$$

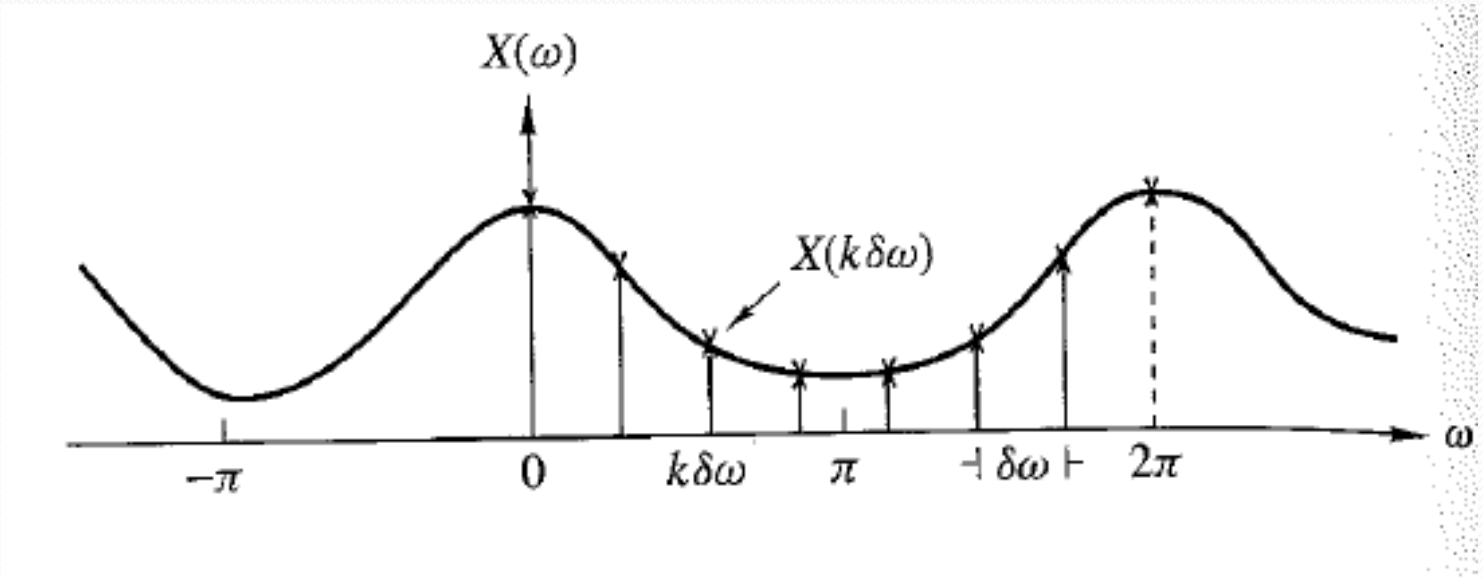


Discrete Fourier Transform





Discrete Fourier Transform



Discrete Fourier Transform

- Discrete fourier transform for a finite sequence is given by:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi \frac{kn}{N}}$$

- Evaluating summation for finite sequence for 0 to N-1.
- k/N corresponds to frequency F
- N corresponds to time for discrete signals

Discrete Fourier Transform

- Magnitude function
- $X(k) = \sqrt{Xr^2(k)+Xi^2(k)}$

- Phase function
- Angle $X(k)=\tan^{-1}[Xi(k)/Xr(k)]$

Inverse Discrete Fourier Transform

The inverse discrete Fourier transform of X(k) is defined as

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j2\pi kn/N} \quad 0 \leq n \leq N-1$$

Where K and n are in the range of 0 ,1,2.....N-1 For example, if N=4, K= 0,1,2,3: N=0,1,2,3

Properties of DFT-

- Periodicity

$$x(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X(k)$$

$$x(n+N) = x(n) \qquad \text{for all } n$$

$$X(k+N) = X(k) \qquad \text{for all } k$$

Properties of DFT

- Linearity

$$x_1(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X_1(k)$$

$$x_2(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X_2(k)$$

$$a_1 x_1(n) + a_2 x_2(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} a_1 X_1(k) + a_2 X_2(k)$$

Properties of DFT

- Circular Convolution

$$x_1(n) \overset{\underset{N}{\text{DFT}}}{\longleftrightarrow} X_1(k)$$

$$x_2(n) \overset{\underset{N}{\text{DFT}}}{\longleftrightarrow} X_2(k)$$

$$x_1(n)x_2(n) \overset{\underset{N}{\text{DFT}}}{\longleftrightarrow} \frac{1}{N} X_1(k) \textcircled{N} X_2(k)$$

$$x_1(n) \textcircled{N} x_2(n) \overset{\underset{N}{\text{DFT}}}{\longleftrightarrow} X_1(k)X_2(k)$$

Properties of DFT

- Time reversal

$$x(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X(k)$$

$$x((-n))_N = x(N-n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X((-k))_N = X(N-k)$$

- If the sequence is circularly folded , its DFT is also circularly folded.

Properties of DFT

- Time shift

$$x(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X(k)$$

$$x((n-l))_N \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X(k)e^{-j2\pi kl/N}$$

Properties of DFT

- Conjugation
- DFT of complex conjugate of any sequence is equal to complex conjugate of DFT of that sequence , with sequence delayed by k samples in frequency domain.
- DFT{x(n)}=X(k)
- $DFT\{X^*(n)\}=X^*(N-k)$

Unitary Transform A discrete linear transform is unitary if its transform matrix conforms to the unitary condition

$$A \times A^{H} = \mathrm{I} \tag{4.11}$$

where A = transformation matrix, A^{H} represents Hermitian matrix.

$$A^{H} = A^{*T}$$

I = identity matrix

When the transform matrix A is unitary, the defined transform is called unitary transform.

Orthogonal DFT

- If A is unitary and has only real elements , then it is orthogonal matrix

- A is orthogonal if $A.A' = I$

Image Transform

1) Check whether the DFT matrix is unitary or not.

Solution

Step 1 **Determination of the matrix *A***

Finding 4-point DFT (where $N = 4$)

The formula to compute a DFT matrix of order 4 is given below.

$$X(K) = \sum_{n=0}^{3} x(n) e^{-j\frac{2\pi}{4}kn} \quad \text{where } k = 0, 1..., 3$$

1. *Finding* $X(0)$

$$X(0) = \sum_{n=0}^{3} x(n) = x(0) + x(1) + x(2) + x(3)$$

2. *Finding* $X(1)$

$$X(1) = \sum_{n=0}^{3} x(n) e^{-j\frac{\pi}{2}n}$$

$$= x(0) + x(1) e^{-j\frac{\pi}{2}} + x(2) e^{-j\pi} + x(3) e^{-j\frac{3\pi}{2}}$$

$$X(1) = x(0) - jx(1) - x(2) + jx(3)$$

3. *Finding* $X(2)$

$$X(2) = \sum_{n=0}^{3} x(n) e^{-j\pi n}$$

$$= x(0) + x(1) e^{-j\pi} + x(2) e^{-j2\pi} + x(3) e^{-j3\pi}$$

$$X(2) = x(0) - x(1) + x(2) - x(3)$$

4. *Finding* $X(3)$

$$X(3)=\sum_{n=0}^{3}x(n)e^{-j\frac{3\pi}{2}n}$$

$$=x(0)+x(1)e^{-j\frac{3\pi}{2}}+x(2)e^{-j3\pi}+x(3)e^{-j\frac{9\pi}{2}}$$

$$X(3)=x(0)+jx(1)-x(2)-jx(3)$$

Collecting the coefficients of $X(0)$, $X(1)$, $X(2)$ and $X(3)$, we get

$$X[k]=A=\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & -j & -1 & j\\ 1 & -1 & 1 & -1\\ 1 & j & -1 & -j\end{bmatrix}$$

***Step 2* Computation of A^H**

To determine A^H, first determine the conjugate and then take its transpose.

$$A \xrightarrow{\text{Conjugate}} A^* \xrightarrow{\text{Transpose}} A^H$$

Step 2a Computation of conjugate A^*

$$A^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}$$

Step 2b Determination of transpose of A^*

$$\left(A^*\right)^T = A^{\mathrm{H}} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}$$

***Step 3* Determination of $A \times A^H$**

$$A \times A^{\mathrm{H}} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix} = 4 \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The result is the identity matrix, which shows that Fourier transform satisfies unitary condition.

Unitary Transform

- unitary transformation preserves the **signal energy**
- Most unitary transforms pack a **large fraction of the energy of the image into relatively few of the transform coefficients.**
- Few of the transform coefficients have significant values and these are the coefficients that are close to the origin

DFT of Image Matrix (2D)

Example 4.4 *Compute the 2D DFT of the 4 × 4 grayscale image given below.*

$$f[m, n] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Solution The 2D DFT of the image $f[m, n]$ is represented as $F[k, l]$.

$$F[k, l] = \text{kernel} \times f[m, n] \times (\text{kernel})^T$$

The kernel or basis of the Fourier transform for $N = 4$ is given by

The DFT basis for $N = 4$ is given by $\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$

DFT of Image Matrix (2D)

$$F(k,l) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$F(k,l) = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} = \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Example 4.5 *Compute the inverse 2D DFT of the transform coefficients given by*

$$F[k,l] = \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Solution The inverse 2D DFT of the Fourier coefficients $F[k, l]$ is given by $f[m, n]$ as

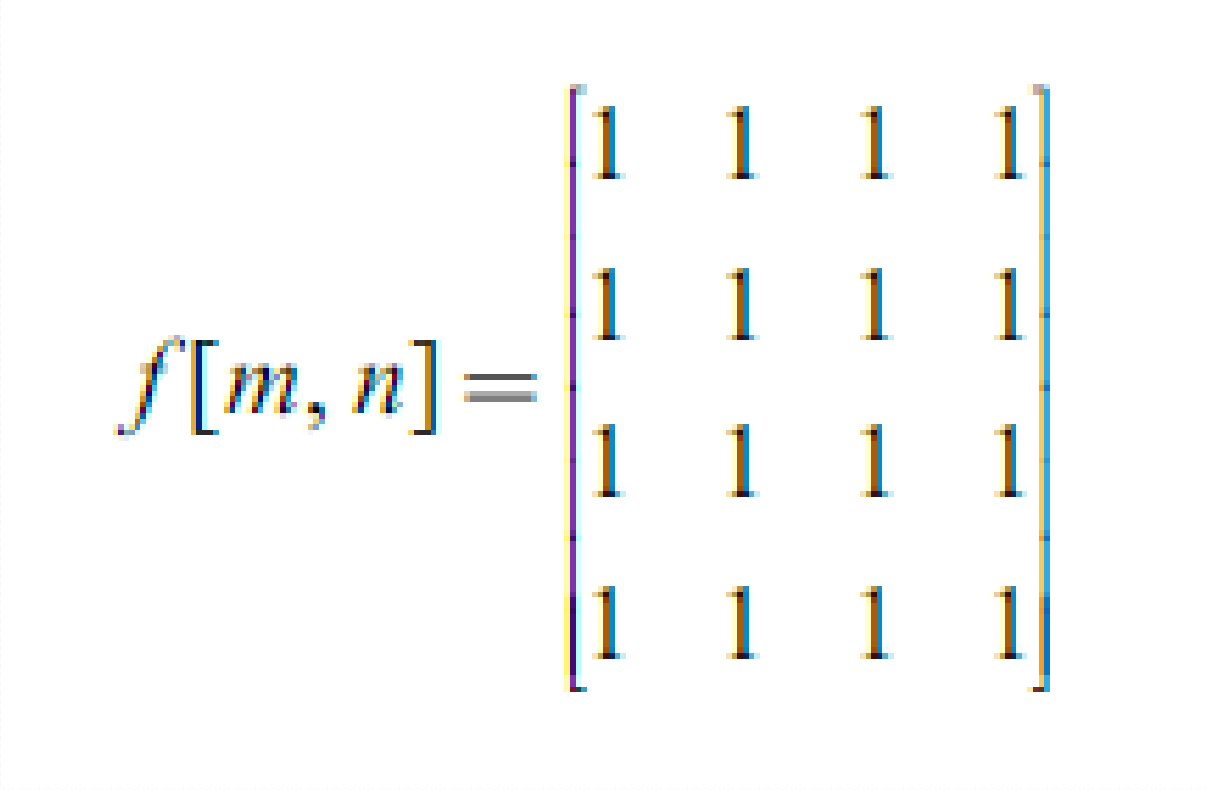
$$f[m,n] = \frac{1}{N^2} \times \text{kernel} \times F[k,l] \times (\text{kernel})^T$$

In this example, $N = 4$

$$f[m,n] = \frac{1}{16} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \times \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & \mathbf{0} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$f[m,n] = \frac{1}{16} \times \begin{bmatrix} 16 & 0 & 0 & 0 \\ 16 & 0 & 0 & 0 \\ 16 & 0 & 0 & 0 \\ 16 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & \mathbf{-1} & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

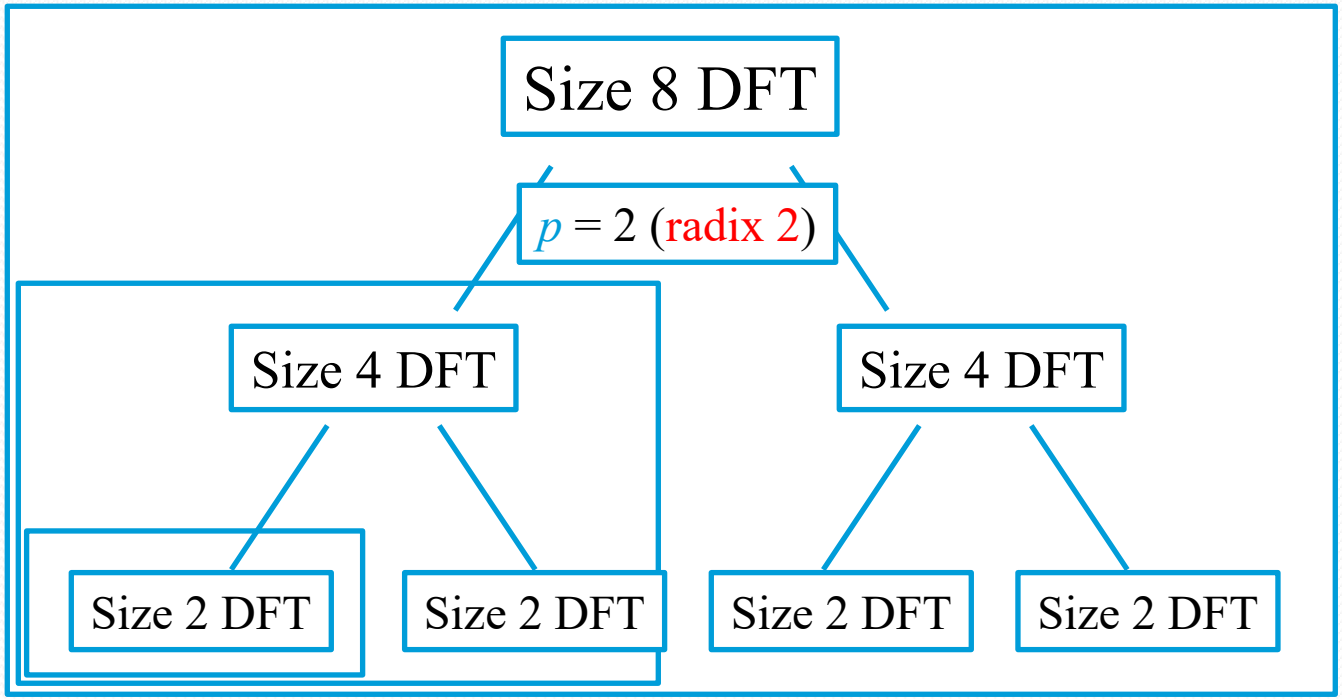
$$f[m,n] = \frac{1}{16} \times \begin{bmatrix} 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \end{bmatrix}$$


f[m, n] =
1 1 1 1
1 1 1 1
1 1 1 1
1 1 1 1

Fast Fourier Transform

- FFT is method of computing DFT with reduced number of calculations.
- Computational efficiency is achieved if we adopt a divide and conquer approach.
- Approach is based on decomposition of N point DFT into successively smaller DFTs.

FFT-Divide & Conquer



Radix-2 FFT

- Radix-x: here ‘x’ represents number of samples in each group made at the first stage. They are generally equal.

- N=2^m
- m is number of stages
- N point sequence is decimated into 2 point sequences, further from the results of 2 point DFTs, the 4 point DFTs can be computed and so on.

Radix -2

- Radix-2 is the first FFT algorithm. It was proposed by Cooley and Tukey in 1965.
- The algorithms appear either in

  (a) **D**ecimation **I**n **T**ime (DIT), or,

  (b) **D**ecimation **I**n **F**requency (DIF).
- DIT and DIF, both yield same complexity and results. They are complementary.
- We shall stress on 8 to radix 2 DIT FFT

Number of Calculations in N-point DFT

Let, X(k) be N-point DFT of an L-point discrete time sequence x(n), where $N \geq L$. Now, the N-point DFT is a sequence consisting of N-complex numbers. Each complex number of the sequence is calculated using the following equation (equation 5.2).

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi kn}{N}} \quad ; \quad \text{for } k = 0,1,2,\ldots\ldots,N-1$$

$$= x(0)\, e^{0} + x(1)\, e^{\frac{-j2\pi k}{N}} + x(2)\, e^{\frac{-j4\pi k}{N}} + x(3)\, e^{\frac{-j6\pi k}{N}} + \ldots\ldots\ldots\ldots + x(N-1)\, e^{\frac{-j2(N-1)\pi k}{N}}$$

$$\therefore X(k) = \underbrace{\underbrace{x(0)e^{0}}_{\text{Complex multiplication}} + \underbrace{x(1)e^{\frac{-j2\pi k}{N}}}_{\text{Complex multiplication}} + \underbrace{x(2)e^{\frac{-j4\pi k}{N}}}_{\text{Complex multiplication}} + \underbrace{x(3)e^{\frac{-j6\pi k}{N}}}_{\text{Complex multiplication}} + \ldots\ldots + \underbrace{x(N-1)e^{\frac{-j2(N-1)\pi k}{N}}}_{\text{Complex multiplication}}}_{N-1 \text{ Complex additions}}$$

The number of calculations to calculate X(k) for one value of k are,

N number of complex multiplications and

N – 1 number of complex additions.

The X(k) is a sequence consisting of N complex numbers.

Therefore, the number of calculations to calculate all the N complex numbers of the X(k) are,

$\mathbf{N \times N = N^2}$ number of complex multiplications and

$\mathbf{N \times (N-1) = N(N-1)}$ number of complex additions.

Hence, in direct computation of N-point DFT, the total number of complex additions are N(N – 1) and total number of complex multiplications are N^2.

Number of Calculations in Radix-2 FFT

In radix-2 FFT, $N = 2^m$, and so there will be m stages of computations, where $m = \log_2 N$, with each stage having N/2 butterflies. (Refer section 5.7.2 and 5.8.2).

The number of calculations in one butterfly are,

1 number of Complex multiplication and

2 number of Complex additions.

There are $\frac{N}{2}$ butterflies in each stage.

Therefore, number of calculations in one stage are,

$\frac{N}{2} \times 1 = \frac{N}{2}$ complex multiplications and

$\frac{N}{2} \times 2 = N$ complex additions.

The N-point DFT involves m stages of computations. Therefore, the number of calculations for m stages are,

$m \times \frac{N}{2} = \log_2 N \times \frac{N}{2} = \frac{N}{2} \log_2 N$ complex multiplications and

$m \times N = \log_2 N \times N = N \log_2 N$ complex additions.

Number of points N	Direct Computation		Radix-2 FFT	
	Complex additions $N(N-1)$	Complex Multiplications N^2	Complex additions $N\log_2 N$	Complex Multiplications $(N/2)\log_2 N$
$4 (=2^2)$	12	16	$4 \times \log_2 2^2 = 4 \times 2 = 8$	$\frac{4}{2} \times \log_2 2^2 = \frac{4}{2} \times 2 = 4$
$8 (=2^3)$	56	64	$8 \times \log_2 2^3 = 8 \times 3 = 24$	$\frac{8}{2} \times \log_2 2^3 = \frac{8}{2} \times 3 = 12$
$16 (=2^4)$	240	256	$16 \times \log_2 2^4 = 16 \times 4 = 64$	$\frac{16}{2} \times \log_2 2^4 = \frac{16}{2} \times 4 = 32$
$32 (=2^5)$	992	1,024	$32 \times \log_2 2^5 = 32 \times 5 = 160$	$\frac{32}{2} \times \log_2 2^5 = \frac{32}{2} \times 5 = 80$
$64 (=2^6)$	4,032	4,096	$64 \times \log_2 2^6 = 64 \times 6 = 384$	$\frac{64}{2} \times \log_2 2^6 = \frac{64}{2} \times 6 = 192$
$128 (=2^7)$	16,256	16,384	$128 \times \log_2 2^7 = 128 \times 7 = 896$	$\frac{128}{2} \times \log_2 2^7 = \frac{128}{2} \times 7 = 448$

Phase or Twiddle Factor

By the definition of DFT, the N-point DFT is given by,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}} \quad ; \text{for } k = 0, 1, 2, \ldots\ldots, N-1 \qquad \ldots\ldots(5.24)$$

To simplify the notation it is desirable to define the complex valued phase factor W_N (also called as twiddle factor) which is an N^{th} root of unity as,

$$W_N = e^{\frac{-j2\pi}{N}}$$

Here, W represents a complex number $1\angle -2\pi$. Hence the phase or argument of W is -2π. Therefore, when a number is multiplied by W, only its phase changes by -2π but magnitude remains same.

$$\therefore W = e^{-j2\pi}$$

The phase value -2π of W can be multiplied by any integer and it is represented as prefix in W. For example multiplying -2π by k can be represented as W^k.

$$\therefore e^{-j2\pi \times k} \Rightarrow W^k$$

The phase value -2π of W can be divided by any integer and it is represented as suffix in W. For example dividing -2π by N can be represented as W_N.

$$\therefore e^{\frac{-j2\pi}{N} \times N} = e^{-j2\pi \times \frac{1}{N}} \Rightarrow W_N$$

$$\therefore e^{\frac{-j2\pi nk}{N}} = \left(e^{-j2\pi}\right)^{\frac{nk}{N}} = W_N^{nk} \qquad \text{.....(5.25)}$$

Using equation (5.25) the equation (5.24) can be written as,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{kn} \quad ; \text{ for } k = 0, 1, 2, \ldots\ldots, N-1 \qquad \text{.....(5.26)}$$

The equation (5.26) is the definition of N-point DFT using phase factor, and this equation is popularly used in FFT.

Twiddle Factor

$$\text{DFT:}\quad X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}, \qquad k = 0, 1, 2, \ldots, N-1$$

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}, \qquad k = 0, 1, \ldots, N-1$$

$$\text{IDFT:}\quad x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j2\pi kn/N}, \qquad n = 0, 1, 2, \ldots, N-1$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-kn}, \qquad n = 0, 1, \ldots, N-1$$

Twiddle Factor

$\sum_{k=0}^{N-1} W_N^{nk}$ can be expanded as NXN **DFT matrix**

$$\sum_{k=0}^{N-1} W_N^{nk} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N^1 & W_N^2 & \cdots & W_N^{(N-1)} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{(N-1)} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)^2} \end{bmatrix}$$

DFT:

$$X(k) := \sum_{n=0}^{n-1} \left[x(n) \left(W_N \right)^{nk} \right]$$

For N of length 4,range of n, k = [0 1 2 3] each.

Hence $X(n) = x(0)W_N^{n.0} + x(1)W_N^{n.1} + x(2)W_N^{n.2} + x(3)W_N^{n.3}$

		x(0)	x(1)	x(2)	x(3)
X(0)	=	W_4^{0x0}	W_4^{0x1}	W_4^{0x2}	W_4^{0x3}
X(1)	=	W_4^{1x0}	W_4^{1x1}	W_4^{1x2}	W_4^{1x3}
X(2)	=	W_4^{2x0}	W_4^{2x1}	W_4^{2x2}	W_4^{2x3}
X(3)	=	W_4^{3x0}	W_4^{3x1}	W_4^{3x2}	W_4^{3x3}

		x(0)	x(1)	x(2)	x(3)
X(0)	=	W_4^{0x0}	W_4^{0x1}	W_4^{0x2}	W_4^{0x3}
X(1)	=	W_4^{1x0}	W_4^{1x1}	W_4^{1x2}	W_4^{1x3}
X(2)	=	W_4^{2x0}	W_4^{2x1}	W_4^{2x2}	W_4^{2x3}
X(3)	=	W_4^{3x0}	W_4^{3x1}	W_4^{3x2}	W_4^{3x3}

Can be rewritten as:

		x(0)	x(1)	x(2)	x(3)
X(0)	=	W_4^{0}	W_4^{0}	W_4^{0}	W_4^{0}
X(1)	=	W_4^{0}	W_4^{1}	W_4^{2}	W_4^{3}
X(2)	=	W_4^{0}	W_4^{2}	W_4^{4}	W_4^{6}
X(3)	=	W_4^{0}	W_4^{3}	W_4^{6}	W_4^{9}

		x(0)	x(1)	x(2)	x(3)
X(0)	=	W_4^0	W_4^0	W_4^0	W_4^0
X(1)	=	W_4^0	W_4^1	W_4^2	W_4^3
X(2)	=	W_4^0	W_4^2	W_4^4	W_4^6
X(3)	=	W_4^0	W_4^3	W_4^6	W_4^9

		x(0)	x(1)	x(2)	x(3)
X(0)	=	1	1	1	1
X(1)	=	1	-j	-1	j
X(2)	=	1	-1	1	-1
X(3)	=	1	j	-1	-j

$$\begin{pmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{pmatrix} := \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{pmatrix} \cdot \begin{pmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{pmatrix}$$

The effective determinant of above is 1/4.
Conversion in system matrices of DFT and IDFT obtained by replacing by j by –j.
Inversion of above matrix is:

$$\begin{pmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{pmatrix} := \left[\frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{pmatrix} \cdot \begin{pmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{pmatrix} \right]$$

FOR 8 point DFT: N=8

- The system Matrix is:

W_8^{0x0}	W_8^{0x1}	W_8^{0x2}	W_8^{0x3}	W_8^{0x4}	W_8^{0x5}	W_8^{0x6}	W_8^{0x7}
W_8^{1x0}	W_8^{1x1}	W_8^{1x2}	W_8^{1x3}	W_8^{1x4}	W_8^{1x5}	W_8^{1x6}	W_8^{1x7}
W_8^{2x0}	W_8^{2x1}	W_8^{2x2}	W_8^{2x3}	W_8^{2x4}	W_8^{2x5}	W_8^{2x6}	W_8^{2x7}
W_8^{3x0}	W_8^{3x1}	W_8^{3x2}	W_8^{3x3}	W_8^{3x4}	W_8^{3x5}	W_8^{3x6}	W_8^{3x7}
W_8^{4x0}	W_8^{4x1}	W_8^{4x2}	W_8^{4x3}	W_8^{4x4}	W_8^{4x5}	W_8^{4x6}	W_8^{4x7}
W_8^{5x0}	W_8^{5x1}	W_8^{5x2}	W_8^{5x3}	W_8^{5x4}	W_8^{5x5}	W_8^{5x6}	W_8^{5x7}
W_8^{6x0}	W_8^{6x1}	W_8^{6x2}	W_8^{6x3}	W_8^{6x4}	W_8^{6x5}	W_8^{6x6}	W_8^{6x7}
W_8^{7x0}	W_8^{7x1}	W_8^{7x2}	W_8^{7x0}	W_8^{7x4}	W_8^{7x5}	W_8^{7x6}	W_8^{7x7}

The characteristic matrix of 8-point DFT is:

$$\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & \frac{1-j}{\sqrt{2}} & -j & \frac{-(1+j)}{\sqrt{2}} & -1 & \frac{-(1-j)}{\sqrt{2}} & j & \frac{1+j}{\sqrt{2}} \\
1 & -j & -1 & j & 1 & -j & -1 & j \\
1 & \frac{-(1+j)}{\sqrt{2}} & j & \frac{1-j}{\sqrt{2}} & -1 & \frac{1+j}{\sqrt{2}} & -j & \frac{-(1-j)}{\sqrt{2}} \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & \frac{-(1-j)}{\sqrt{2}} & -j & \frac{1+j}{\sqrt{2}} & -1 & \frac{1-j}{\sqrt{2}} & j & \frac{-(1+j)}{\sqrt{2}} \\
1 & j & -1 & -j & 1 & j & -1 & -j \\
1 & \frac{1+j}{\sqrt{2}} & j & \frac{-(1-j)}{\sqrt{2}} & -1 & \frac{-(1+j)}{\sqrt{2}} & -j & \frac{(1-j)}{\sqrt{2}}
\end{bmatrix}$$

DFT and its Inverse

Conversion in system matrices of DFT and IDFT obtained by replacing by j by –j..

$$\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & \frac{1-j}{\sqrt{2}} & -j & \frac{-(1+j)}{\sqrt{2}} & -1 & \frac{-(1-j)}{\sqrt{2}} & j & \frac{1+j}{\sqrt{2}} \\
1 & -j & -1 & j & 1 & -j & -1 & j \\
1 & \frac{-(1+j)}{\sqrt{2}} & j & \frac{1-j}{\sqrt{2}} & -1 & \frac{1+j}{\sqrt{2}} & -j & \frac{-(1-j)}{\sqrt{2}} \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & \frac{-(1-j)}{\sqrt{2}} & -j & \frac{1+j}{\sqrt{2}} & -1 & \frac{1-j}{\sqrt{2}} & j & \frac{-(1+j)}{\sqrt{2}} \\
1 & j & -1 & -j & 1 & j & -1 & -j \\
1 & \frac{1+j}{\sqrt{2}} & j & \frac{-(1-j)}{\sqrt{2}} & -1 & \frac{-(1+j)}{\sqrt{2}} & -j & \frac{(1-j)}{\sqrt{2}}
\end{bmatrix}$$

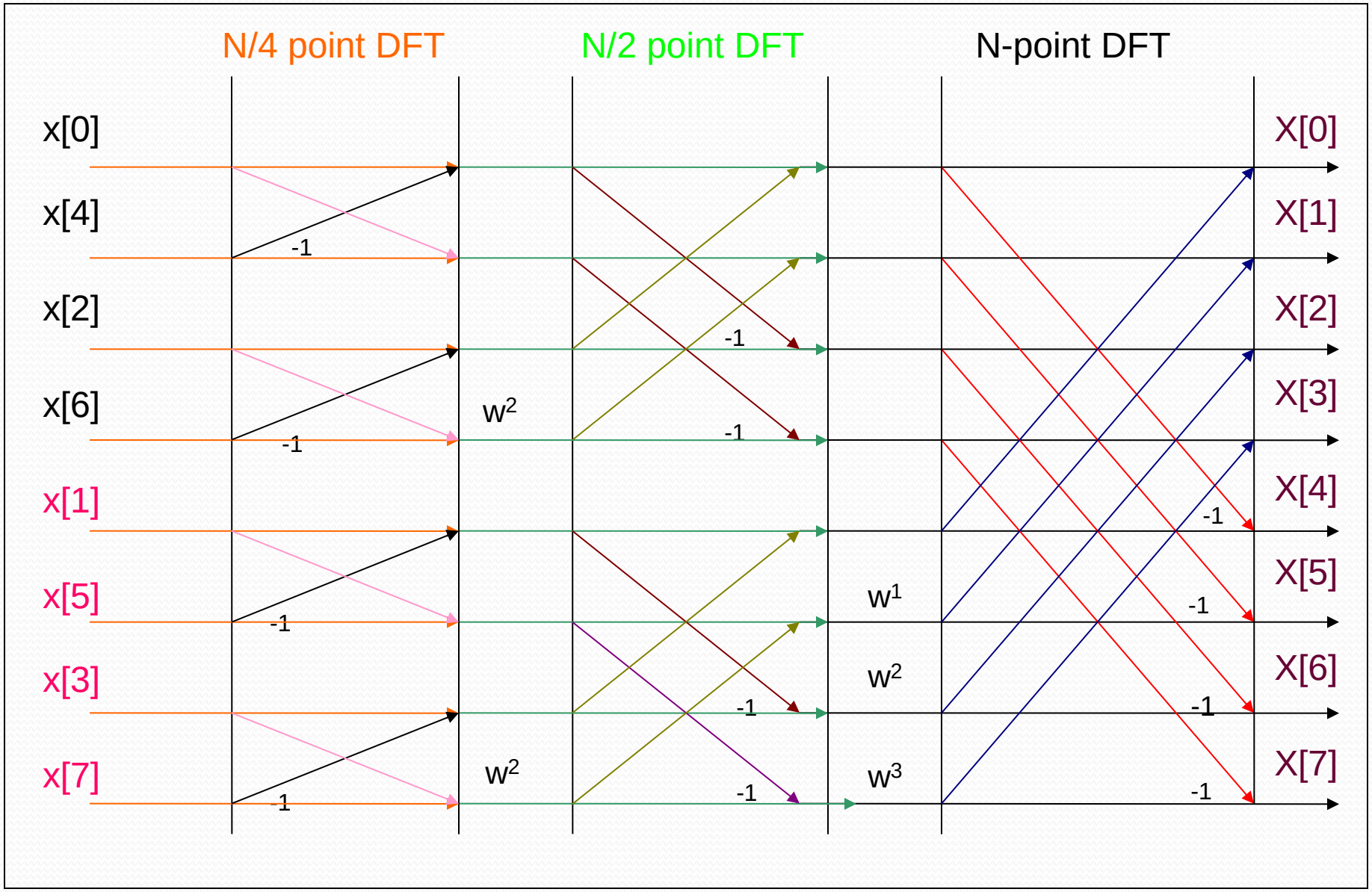
DFT

Effective determinant

$$\frac{1}{8}\cdot\left(\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & \frac{1+j}{\sqrt{2}} & j & -\frac{1-j}{\sqrt{2}} & -1 & -\frac{1+j}{\sqrt{2}} & -j & \frac{1-j}{\sqrt{2}} \\
1 & j & -1 & -j & 1 & j & -1 & -j \\
1 & -\frac{1-j}{\sqrt{2}} & -j & \frac{1+j}{\sqrt{2}} & -1 & \frac{1-j}{\sqrt{2}} & j & -\frac{1+j}{\sqrt{2}} \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & -\frac{1+j}{\sqrt{2}} & j & \frac{1-j}{\sqrt{2}} & -1 & \frac{1+j}{\sqrt{2}} & -j & -\frac{1-j}{\sqrt{2}} \\
1 & -j & -1 & j & 1 & -j & -1 & j \\
1 & \frac{1-j}{\sqrt{2}} & -j & -\frac{1+j}{\sqrt{2}} & -1 & -\frac{1-j}{\sqrt{2}} & j & \frac{1+j}{\sqrt{2}}
\end{bmatrix}\right)$$

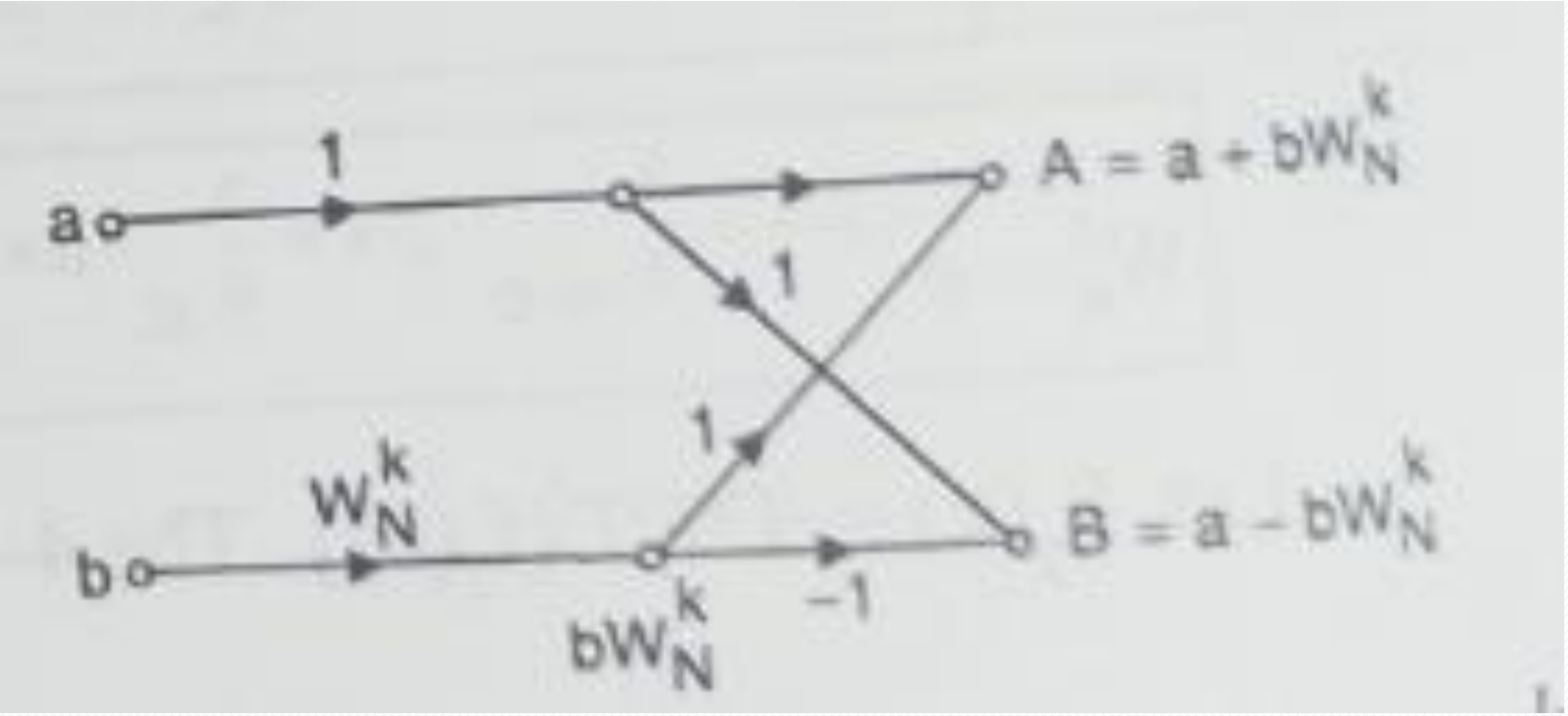
IDFT

N=8-point radix-2 DIT-FFT:



DIT-FFT

- Steps to find DFT using Radix 2 DIT FFT:



Ex: Draw the signal flow graph of the 4-point FFT, then use to it to find the spectrum of the sequence x(n)={1,-1,2,3}.

decimal	binary	binary	decimal
0	00	00	0
1	01	10	2
2	10	01	1
3	11	11	3

Mirror

The even numbered samples are x(0) and x(2)

The odd numbered samples are x(1) and x(3)

c $X(k)=G(k)+W_4^k H(k) \quad 3 \geq k \geq 0.$

$X(0)=G(0)+W_4^0 H(0)$

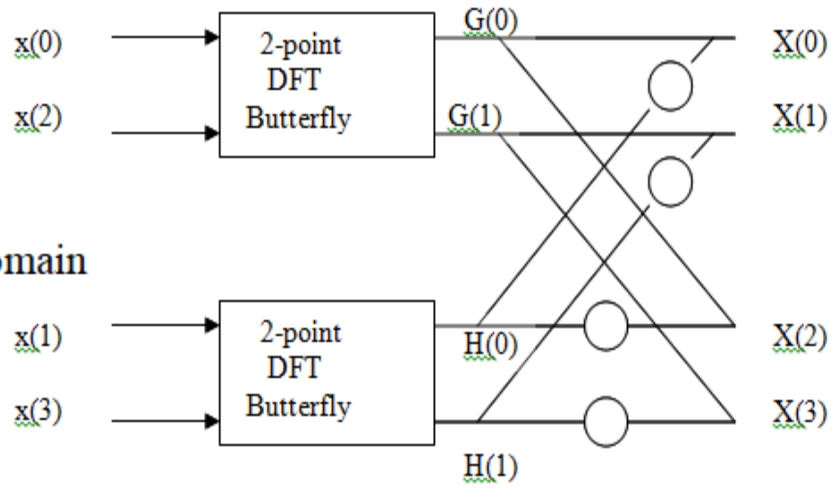
$X(1)=G(1)+W_4^1 H(1)$

$X(2)=G(0)+W_4^2 H(0)$

$X(3)=G(3)+W_4^3 H(3)$

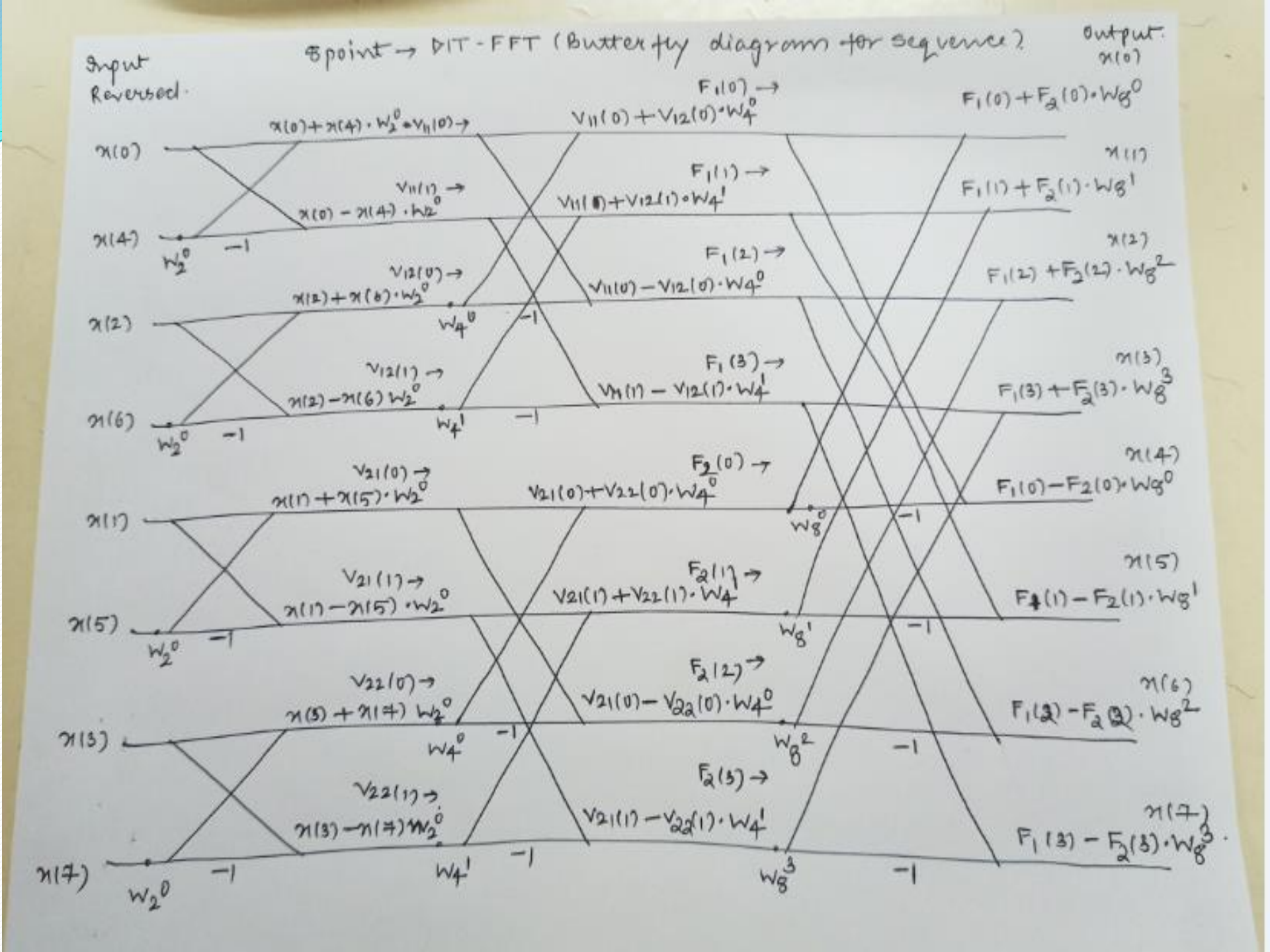
And again G(3)=G(1)and H(3)=H(1), ther

$X(3)=G(1)+W_4^3 H(1)$



$E=\sum x^2(n) = 1+4+1+9=15$ =energy from time domain

$\frac{1}{N}\sum |X(k)|^2 = (25+17+1+17)/4=15$ =energy from frequency domain


8point → DIT - FFT (Butterfly diagram for sequence)
Input Reversed
Output
x(0)
x(4)
x(2)
x(6)
x(1)
x(5)
x(3)
x(7)
W_2^0
W_4^0
W_4^1
W_8^0
W_8^1
W_8^2
W_8^3
-1
$x(0)+x(4)\cdot W_2^0 = V_{11}(0)$
$x(0)-x(4)\cdot W_2^0 = V_{11}(1)$
$x(2)+x(6)\cdot W_2^0 = V_{12}(0)$
$x(2)-x(6)\cdot W_2^0 = V_{12}(1)$
$x(1)+x(5)\cdot W_2^0 = V_{21}(0)$
$x(1)-x(5)\cdot W_2^0 = V_{21}(1)$
$x(3)+x(7)\cdot W_2^0 = V_{22}(0)$
$x(3)-x(7)\cdot W_2^0 = V_{22}(1)$
$F_1(0) = V_{11}(0)+V_{12}(0)\cdot W_4^0$
$F_1(1) = V_{11}(1)+V_{12}(1)\cdot W_4^1$
$F_1(2) = V_{11}(0)-V_{12}(0)\cdot W_4^0$
$F_1(3) = V_{11}(1)-V_{12}(1)\cdot W_4^1$
$F_2(0) = V_{21}(0)+V_{22}(0)\cdot W_4^0$
$F_2(1) = V_{21}(1)+V_{22}(1)\cdot W_4^1$
$F_2(2) = V_{21}(0)-V_{22}(0)\cdot W_4^0$
$F_2(3) = V_{21}(1)-V_{22}(1)\cdot W_4^1$
$x(0) = F_1(0)+F_2(0)\cdot W_8^0$
$x(1) = F_1(1)+F_2(1)\cdot W_8^1$
$x(2) = F_1(2)+F_2(2)\cdot W_8^2$
$x(3) = F_1(3)+F_2(3)\cdot W_8^3$
$x(4) = F_1(0)-F_2(0)\cdot W_8^0$
$x(5) = F_1(1)-F_2(1)\cdot W_8^1$
$x(6) = F_1(2)-F_2(2)\cdot W_8^2$
$x(7) = F_1(3)-F_2(3)\cdot W_8^3$

2 point 4 point 8

$W_2^0 = 1$ $W_4^0 = 1$

$W_4^1 = -j$

$$W_N^k = e^{\frac{-j2\pi k}{N}}$$

2.

$$W_2^0 = e^0 = 1$$

$$W_4^0 = e^0 = 1$$

$$W_4^1 = e^{\frac{-j2\pi(1)}{4}}$$

$e^{-j\pi/2}$ $\cos \pi/2 - j\sin \pi/2$

4 point

$W_4^0 = 1$

$W_4^1 = -j$

8 point

$W_8^0 = 1$

$W_8^1 = \frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$

$W_8^2 = -j$

$W_8^3 = -\frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$

Image Transforms

- Image transforms are extensively used in image processing and image analysis.
- Transform is basically a mathematical tool, which allows us to move from one domain to another domain (time domain to the frequency domain).
- migrate from one domain to another domain is to perform the task at hand in an easier manner.

Image Transforms

- Image transforms are useful for fast computation of convolution and correlation.
- The transforms do not change the information content present in the signal.
- Transforms play a significant role in various image-processing applications such as image analysis, image enhancement, image filtering and image compression.

NEED FOR TRANSFORM

(i) Mathematical Convenience

The complex convolution operation in time domain is equal to simple multiplication operation in the frequency domain.

NEED FOR TRANSFORM

(ii) *To Extract more Information*

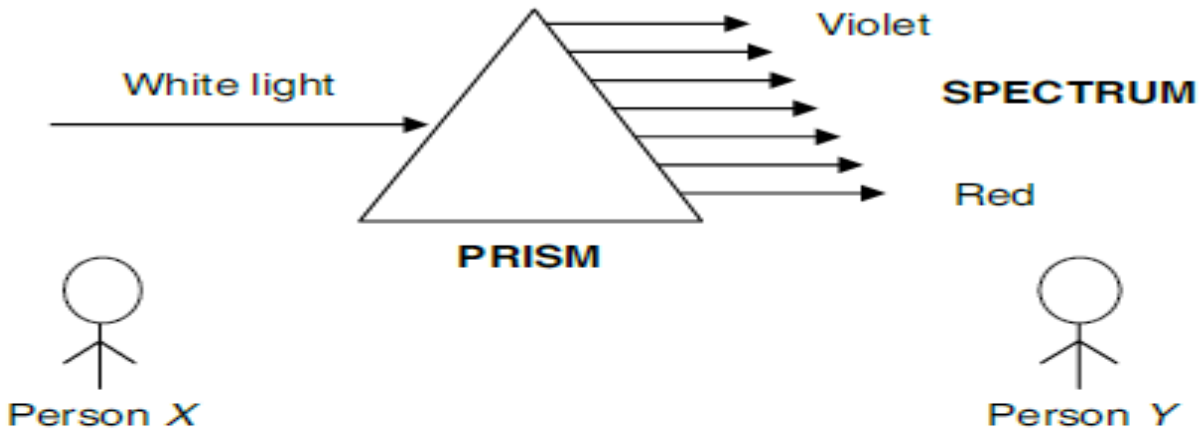


Fig. 4.1 *Spectrum of white light*

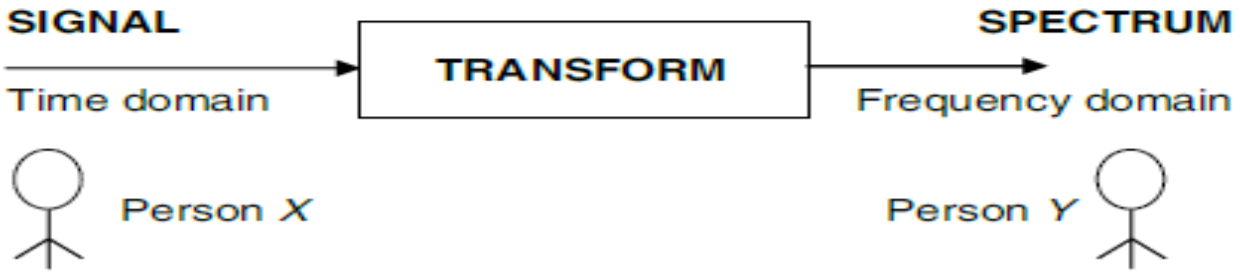


Fig. 4.2 *Concept of transformation*

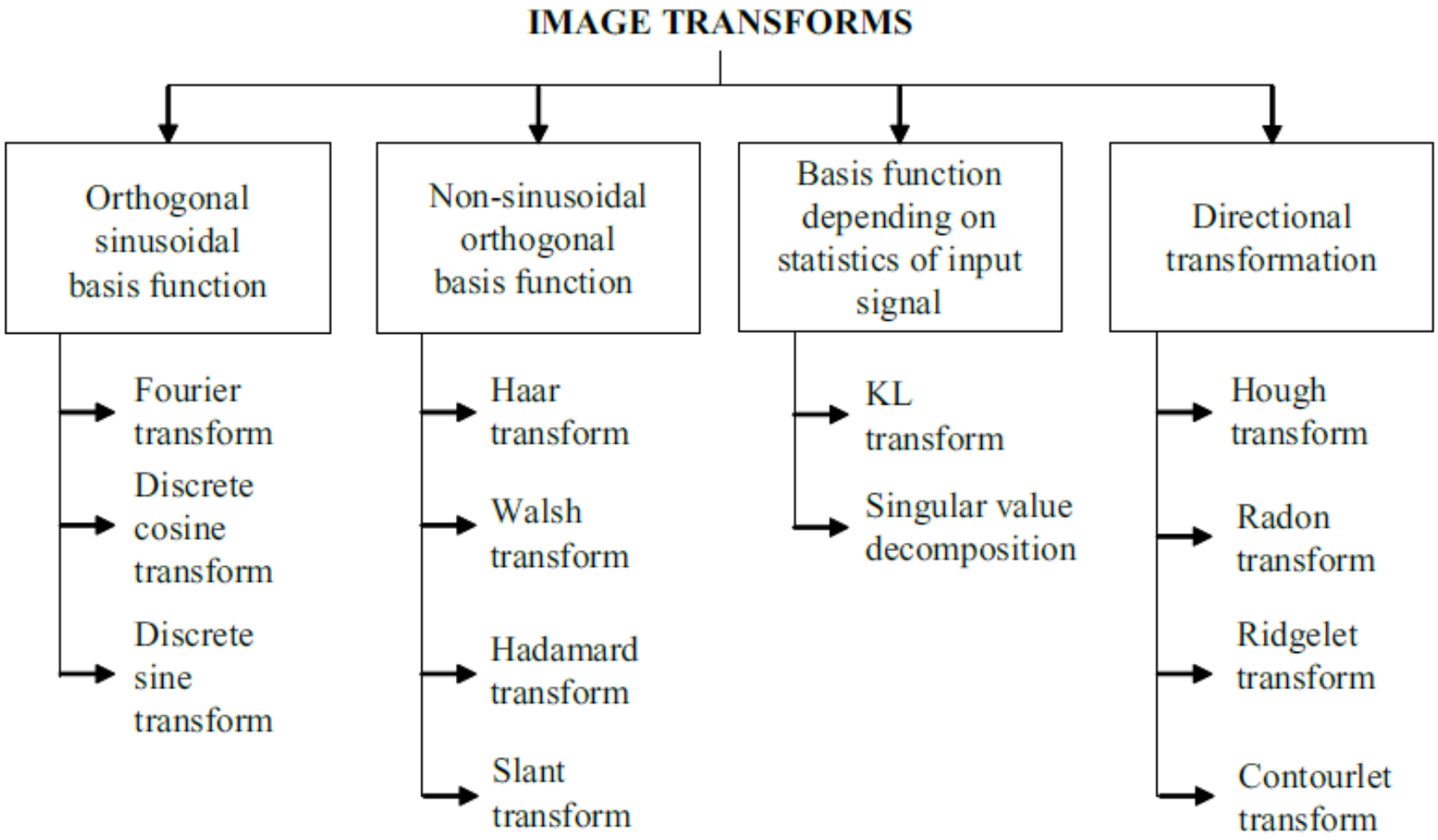


IMAGE TRANSFORMS
Orthogonal sinusoidal basis function
Fourier transform
Discrete cosine transform
Discrete sine transform
Non-sinusoidal orthogonal basis function
Haar transform
Walsh transform
Hadamard transform
Slant transform
Basis function depending on statistics of input signal
KL transform
Singular value decomposition
Directional transformation
Hough transform
Radon transform
Ridgelet transform
Contourlet transform

Image Transform

1) Orthogonal sinusoidal basis function

- One of the most powerful transforms with orthogonal sinusoidal basis function is the Fourier transform.
- The transform which is widely used in the field of **image compression** is discrete cosine transform.

Image Transform

2) Non-sinusoidal orthogonal basis function

- The transforms whose basis functions are non-sinusoidal in nature.

- One of the important advantages of wavelet transform is that signals (images) can be **represented in different resolutions.**

Image Transform

3) Basis function depending on statistics of input signal

- The transform whose basis function depends on the statistics of input signal are KL transform and singular value decomposition.

- The KL transform is considered to be the best among all linear transforms with respect to **energy compaction**.

Image Transform

4) Directional transformation

- The transforms whose basis functions are effective in **representing the directional information of a signal** include Hough transform, Radon transform, Ridgelet transform and Contourlet transform.

Need for transform

- The need for transform is most of the signals or images are time domain signal (ie) signals can be measured with a function of time.
- This representation is not always best.
- For most image processing applications anyone of the mathematical transformation are applied to the signal or images to obtain further information from that signal.

Discrete Cosine Transform (DCT)

- The discrete cosine transforms are the members of a family of real-valued discrete sinusoidal unitary transforms.
- A discrete cosine transform consists of a set of basis vectors that are sampled cosine functions.
- DCT is a technique for converting a signal into elementary frequency components and it is widely used in image compression.

Thus, the kernel of a one-dimensional discrete cosine transform is given by

$$X[k] = \alpha(k) \sum_{n=0}^{N-1} x[n] \cos\left\{ \frac{(2n+1)\pi k}{2N} \right\} \text{ , where } 0 \leq k \leq N-1$$

$$\alpha(k) = \sqrt{\frac{1}{N}} \text{ if } k = 0$$

$$\alpha(k) = \sqrt{\frac{2}{N}} \text{ if } k \neq 0$$

Example 4.10 *Compute the discrete cosine transform (DCT) matrix for N = 4.*

Solution The formula to compute the DCT matrix is given by

$$X[k] = \alpha(k) \sum_{n=0}^{N-1} x[n] \cos\left\{ \frac{(2n+1)\pi k}{2N} \right\}, \text{ where } 0 \leq k \leq N-1$$

where

$$\alpha(k) = \sqrt{\frac{1}{N}} \text{ if } k = 0$$

$$\alpha(k) = \sqrt{\frac{2}{N}} \text{ if } k \neq 0$$

In our case, the value of $N = 4$. Substituting $N = 4$ in the expression of $X[k]$ we get

$$X[k] = \alpha(k)\sum_{n=0}^{3} x[n]\cos\left[\frac{(2n+1)\pi k}{8}\right]$$

Substituting $k = 0$ in Eq. (4.82), we get

$$X[0] = \sqrt{\frac{1}{4}} \times \sum_{n=0}^{3} x[n]\cos\left[\frac{(2n+1)\pi \times 0}{8}\right]$$

$$= \frac{1}{2} \times \sum_{n=0}^{3} x[n]\cos(0) = \frac{1}{2} \times \sum_{n=0}^{3} x[n] \times 1$$

$$= \frac{1}{2} \times \sum_{n=0}^{3} x[n]$$

$$= \frac{1}{2} \times \{x(0) + x(1) + x(2) + x(3)\}$$

$$X[0] = \frac{1}{2}x(0) + \frac{1}{2}x(1) + \frac{1}{2}x(2) + \frac{1}{2}x(3)$$

Substituting $k = 1$ in Eq. (4.82), we get

$$X[1] = \sqrt{\frac{2}{4}} \times \sum_{n=0}^{3} x[n] \cos\left[\frac{(2n+1)\pi \times 1}{8}\right]$$

$$= \sqrt{\frac{1}{2}} \times \sum_{n=0}^{3} x[n] \cos\left[\frac{(2n+1)\pi}{8}\right]$$

$$= \sqrt{\frac{1}{2}} \times \left\{x(0) \times \cos\left(\frac{\pi}{8}\right) + x(1) \times \cos\left(\frac{3\pi}{8}\right) + x(2) \times \cos\left(\frac{5\pi}{8}\right) + x(3) \times \cos\left(\frac{7\pi}{8}\right)\right\}$$

$$= 0.707 \times \left\{x(0) \times 0.9239 + x(1) \times 0.3827 + x(2) \times (-0.3827) + x(3) \times (-0.9239)\right\}$$

$$X(1) = 0.6532x(0) + 0.2706x(1) - 0.2706x(2) - 0.6532x(3)$$

Substituting $k = 2$ in Eq. (4.82), we get

$$X(2) = \sqrt{\frac{2}{4}} \sum_{n=0}^{3} x[n] \cos\left[\frac{(2n+1)\pi \times 2}{8}\right]$$

$$= \sqrt{\frac{1}{2}} \times \sum_{n=0}^{3} x[n] \cos\left[\frac{(2n+1)\pi}{4}\right]$$

$$= \sqrt{\frac{1}{2}} \times \left\{x(0) \times \cos\left(\frac{\pi}{4}\right) + x(1) \times \cos\left(\frac{3\pi}{4}\right) + x(2) \times \cos\left(\frac{5\pi}{4}\right) + x(3) \times \cos\left(\frac{7\pi}{4}\right)\right\}$$

$$= 0.7071 \times \{x(0) \times 0.7071 + x(1) \times (-0.7071) + x(2) \times (-0.7071) + x(3) \times (0.7071)\}$$

$$X(2) = 0.5x(0) - 0.5x(1) - 0.5x(2) + 0.5x(3)$$

Substituting $k = 3$ in Eq. (4.82), we get

$$X(3) = \sqrt{\frac{2}{4}} \sum_{n=0}^{3} x[n] \cos\left[\frac{(2n+1)\pi \times 3}{8}\right]$$

$$= \sqrt{\frac{1}{2}} \times \sum_{n=0}^{3} x[n] \cos\left[\frac{(2n+1) \times 3\pi}{8}\right]$$

$$= \sqrt{\frac{1}{2}} \times \left\{x(0) \times \cos\left(\frac{3\pi}{8}\right) + x(1) \times \cos\left(\frac{9\pi}{8}\right) + x(2) \times \cos\left(\frac{15\pi}{8}\right) + x(3) \times \cos\left(\frac{21\pi}{8}\right)\right\}$$

$$= 0.7071 \times \{x(0) \times 0.3827 + x(1) \times (-0.9239) + x(2) \times (0.9239) + x(3) \times (-0.3827)\}$$

$$X(3) = 0.2706x(0) - 0.6533x(1) + 0.6533x(2) - 0.2706x(3)$$

Collecting the coefficients of $x(0), x(1), x(2)$ and $x(3)$ from $X(0), X(1), X(2)$ and $X(3)$, we get

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.6532 & 0.2706 & -0.2706 & -0.6532 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.2706 & -0.6533 & 0.6533 & -0.2706 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix}$$

IDCT

$$x[n] = \alpha(k) \sum_{k=0}^{N-1} X[k] \cos\left[\frac{(2n+1)\pi k}{2N}\right], 0 \leq n \leq N-1$$

Symmetric and Asymmetric

- **Transformation Matrix – T**
- **Image Matrix – f**
- **If Transformation matrix is symmetric:**
- **F = TfT**

- **If Transformation matrix is asymmetric:**
- **F = TfT'**

Hadamard Transform

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

The Hadamard matrix of order $2N$ can be generated by Kronecker product operation:

$$H_{2N} = \begin{bmatrix} H_N & H_N \\ H_N & -H_N \end{bmatrix}$$

Substituting $N = 2$ in Eq. (4.59), we get

$$H_4 = \begin{bmatrix} H_2 & H_2 \\ H_2 & -H_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

Similarly, substituting $N = 4$ in Eq. (4.59), we get

$$H_8 = \begin{bmatrix} H_4 & H_4 \\ H_4 & -H_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

Hadamard Transform

- Hadamard of Sequence

  $$X[n]=\{H(N).x(n)\}$$

- Inverse Hadamard of Sequence

  $$x[n] = 1/N\{H(N).X(n)\}$$

- Hadamard of image (2D)

  Hadamard is symmetric transform

  $$F = TfT$$

$$H_8 = \begin{bmatrix} H_4 & H_4 \\ H_4 & -H_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \begin{matrix} 0 \\ 7 \\ 3 \\ 4 \\ 1 \\ 6 \\ 2 \\ 5 \end{matrix}$$

$$W(8) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix} \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{matrix}$$

Walsh Transform

$$W(8) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix},$$

Haar Transform

Haar Transform

- For N=2 dan N=4:

$$\mathbf{Hr}_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad \mathbf{Hr}_4 = \frac{1}{\sqrt{4}}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$$

- Properties of Haar Transform:
  1. Real and orthogonal: $\mathbf{Hr} = \mathbf{Hr}^*$ dan $\mathbf{Hr}^{-1} = \mathbf{Hr}^T$
  2. Very fast transform : O(N) operation on Nx1 vector.
  3. Poor energy compaction for images

Haar Transform

$$\mathbf{H}_3 = \frac{1}{\sqrt{8}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{bmatrix} \begin{matrix} h_0 \\ h_1 \\ h_{2,1} \\ h_{2,2} \\ h_{3,1} \\ h_{3,2} \\ h_{3,3} \\ h_{3,4} \end{matrix}$$

IMAGE ENHANCEMENT IN THE FREQUENCY DOMAIN

- If we multiply each element of the Fourier coefficient by a suitably chosen weighting function then we can **accentuate certain frequency components and attenuate others.**
- This **selective enhancement or suppression of frequency components** is termed Fourier filtering or frequency domain filtering.
- The spatial representation of image data **describes the adjacency relationship between the pixels.**
- On the other hand, the frequency domain **representation clusters the image data according to their frequency distribution.**
- In frequency domain filtering, the **image data is dissected into various spectral bands**, where each band depicts a specific range of details within the image.
- The **process of selective frequency inclusion or exclusion is termed frequency domain filtering.**
- It is possible to perform filtering in the frequency domain by specifying the frequencies that should be kept and the frequencies that should be discarded.

IMAGE ENHANCEMENT IN THE FREQUENCY DOMAIN

Convolution in spatial domain = Multiplication in the frequency domain

$$\text{Filtering in spatial domain} = f(m, n) * h(m, n)$$

$$\text{Filtering in the frequency domain} = F(k, l) \times H(k, l)$$

f(m,n) – original image

F(k,l) – FT of original image

h(m,n) – Filtering mask

H(k,l) – FT of filtering mask

Low pass Filtering

- A low-pass filter attenuates high frequencies and retains low frequencies unchanged. The result in the spatial domain is equivalent to that of a smoothing filter; as the blocked high frequencies correspond to sharp intensity changes, *i.e.* to the fine-scale details and noise in the spatial domain image.

High Pass Filtering

- A high pass filter, on the other hand, yields edge enhancement or edge detection in the spatial domain, because edges contain many high frequencies. Areas of rather constant gray level consist of mainly low frequencies and are therefore suppressed.

Band Pass Filtering

- A bandpass attenuates very low and very high frequencies, but retains a middle range band of frequencies. Bandpass filtering can be used to enhance edges (suppressing low frequencies) while reducing the noise at the same time (attenuating high frequencies).

- Bandpass filters are a combination of both lowpass and highpass filters. They attenuate all frequencies smaller than a frequency Do and higher than a frequency D1 , while the frequencies between the two cut-offs remain in the resulting output image.

Low Pass (LP) Filters

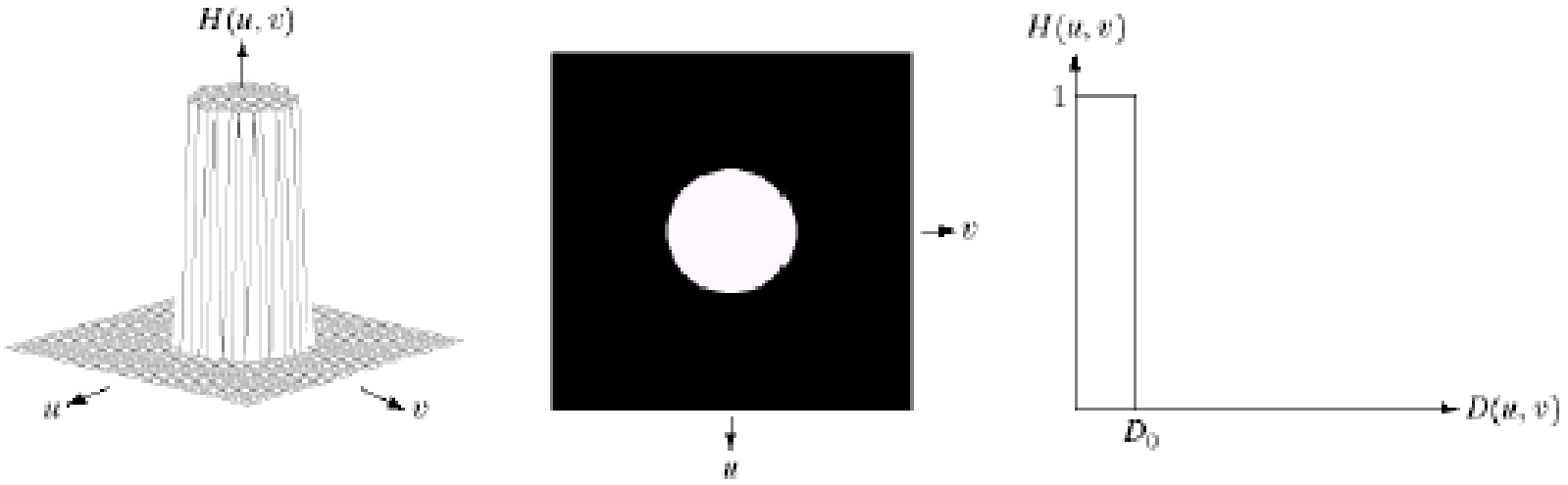
- Ideal low-pass filter (ILPF)
- Butterworth low-pass filter (BLPF)
- Gaussian low-pass filter (GLPF)

Ideal Low-pass (LP) filtering

$$H(u,v) = 1 \text{ ; if } D(u,v) \le D_0$$
$$= 0 \text{ ; if } D(u,v) > D_0$$

$$D(u,v) = \sqrt{(u - M/2)^2 + (v - N/2)^2}$$

D_0: cutoff frequency



a b c

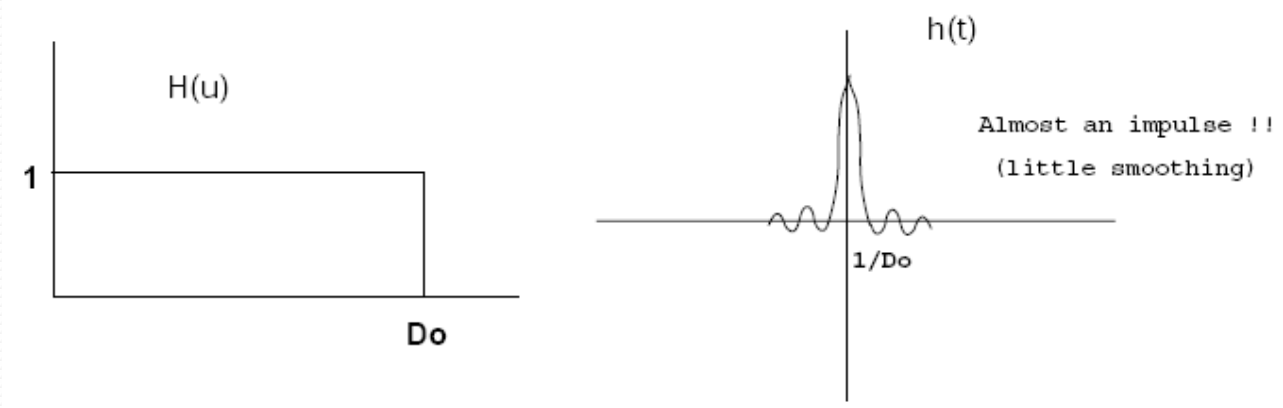
FIGURE 4.10 (a) Perspective plot of an ideal lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross section.

Ideal Low-pass (LP) filtering

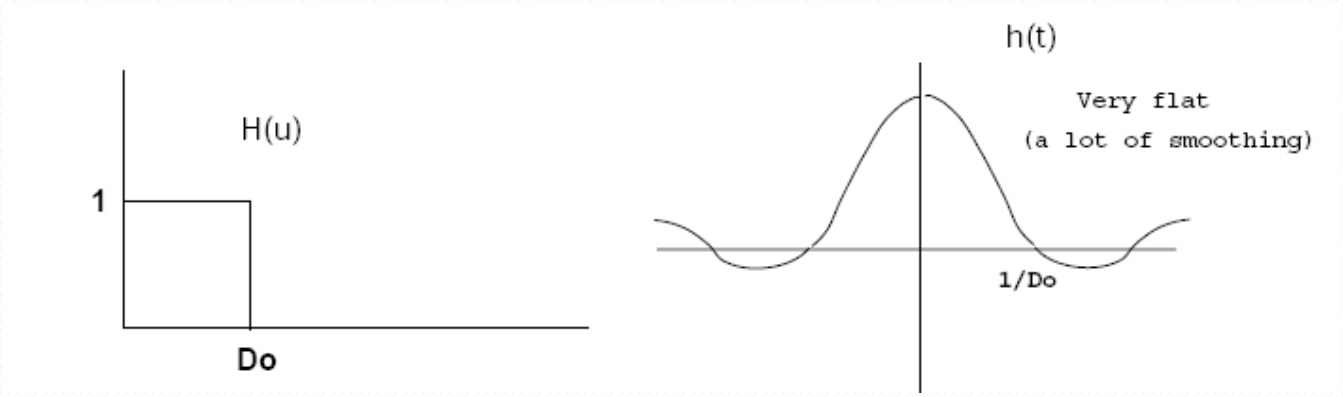
- Do is the cut-off frequency of the low-pass filter.
- The cut-off frequency determines the amount of frequency components passed by the filter.
- The smaller the value of Do , the greater the number of image components eliminated by the filter.
- The value of Do is chosen in such a way that the components of interest are passed through.

How does D_0 control smoothing? (cont'd)

- D_o controls the amount of blurring



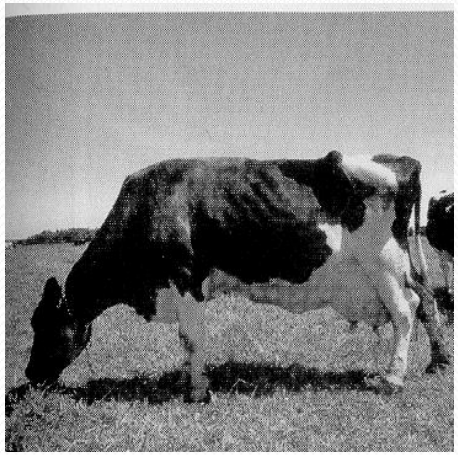
r=78 (99%)

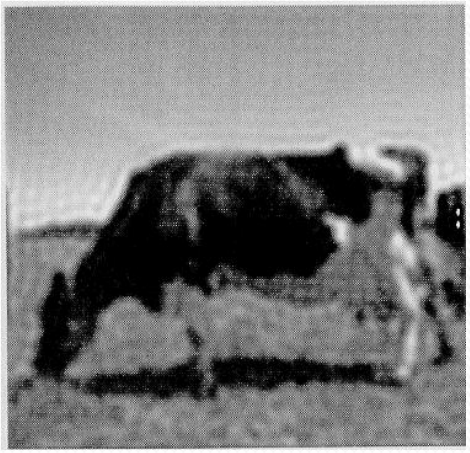


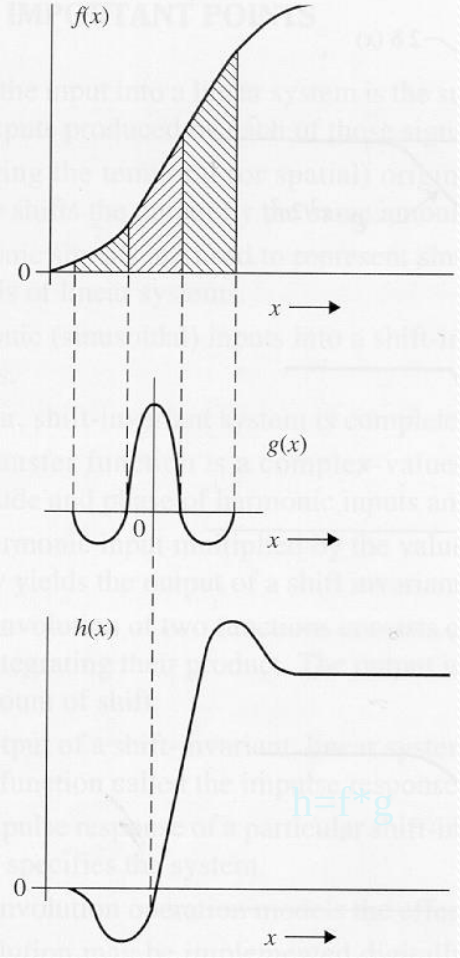
r=8 (90%)

Ringing Effect

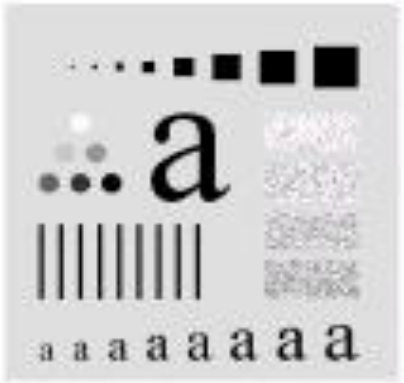
- Sharp cutoff frequencies produce an overshoot of image features whose frequency is close to the cutoff frequencies (**ringing effect**).





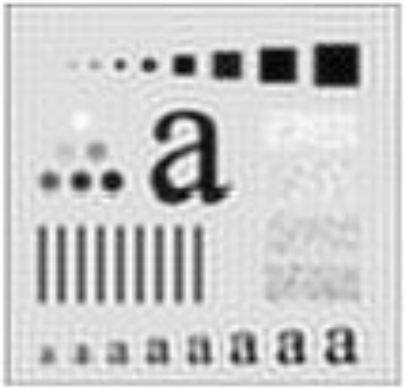


original



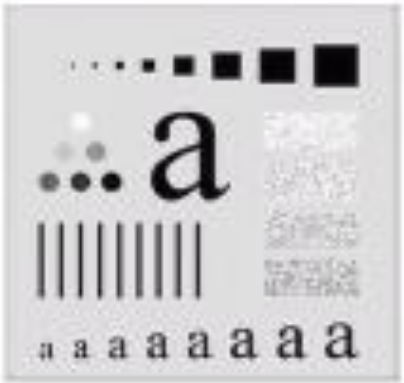
5 pixels
92.0%
Useless,
severe blurring
(8%→ detail)

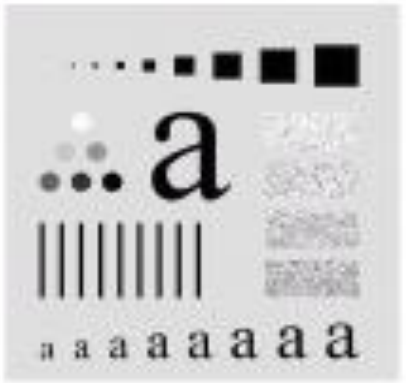
15 pixels
94.6%
Ringing effect
due to ideal filter
behavior



30 pixels
96.4%
Ringing effect
due to ideal filter
behavior

80 pixels
98.9%





230 pixels
99.5%

Butterworth LPF

Provide a smooth transition between low and high frequencies

Reduce the ringing effect caused by the ILPF

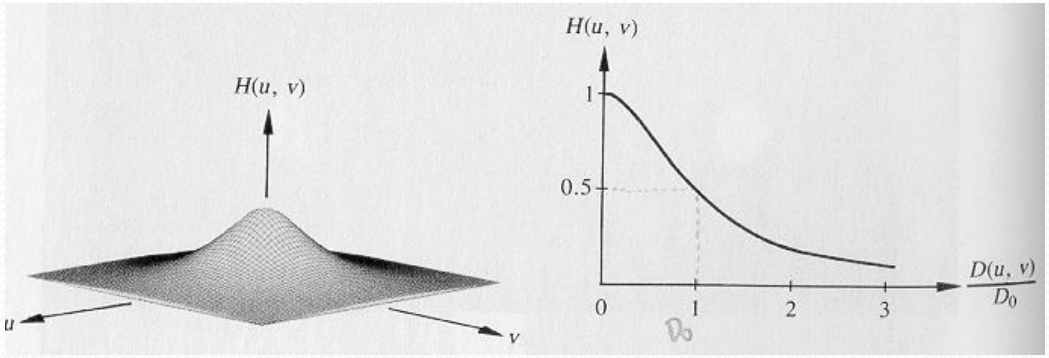
Transfer function of Butterworth LPF (order n, cutoff frequency D_0):

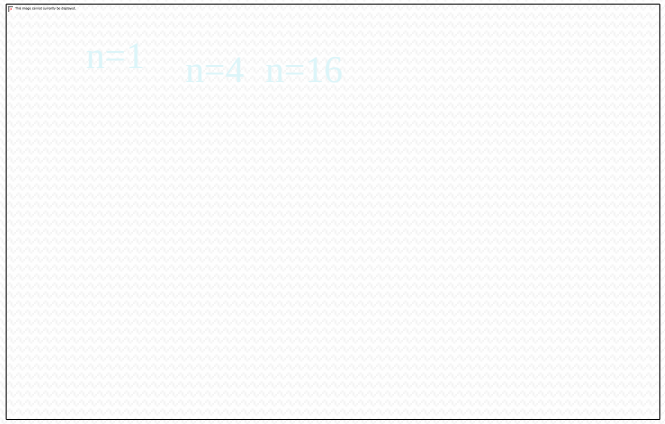
$$H(u,v) = \frac{1}{1+\left[\frac{D(u,v)}{D_0}\right]^{2n}} = \frac{1}{1+\left[\frac{(u-M/2)^2+(v-N/2)^2}{D_0^2}\right]^{n}}$$

Butterworth LP filter (BLPF)

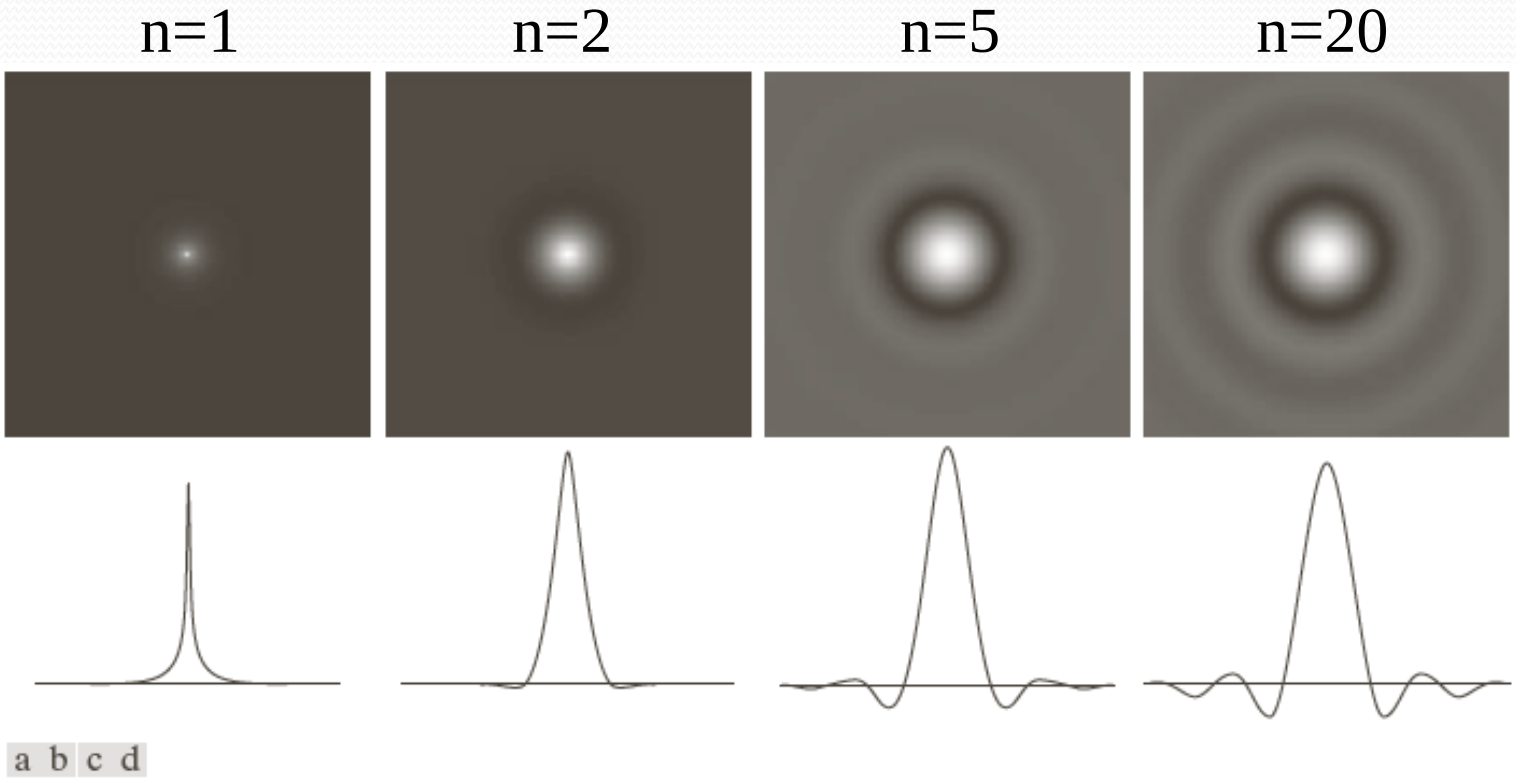
- In practice, we use filters that attenuate high frequencies smoothly (e.g., **Butterworth** LP filter) → less ringing effect

$$H(u,v) = \frac{1}{1 + [\sqrt{u^2 + v^2}/D_0]^{2n}}$$





Spatial Representation of BLPFs



a b c d

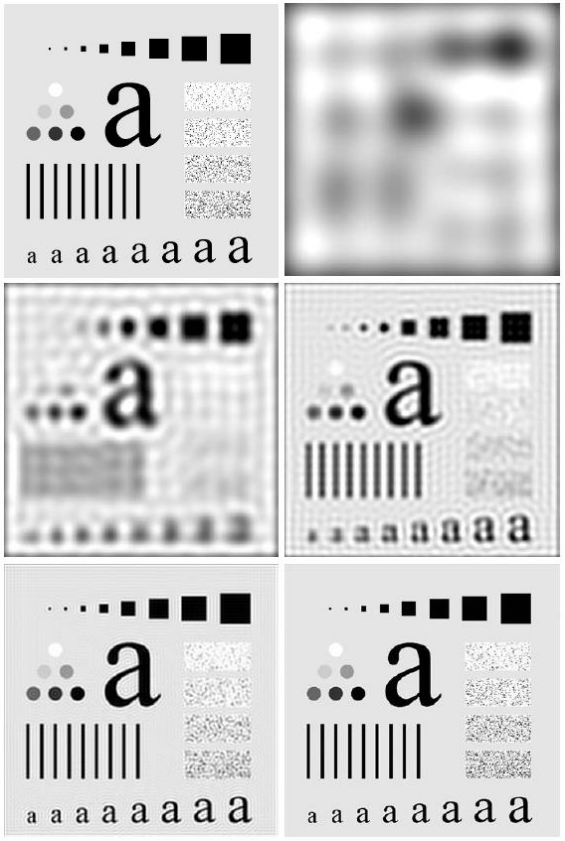
FIGURE 4.46 (a)–(d) Spatial representation of BLPFs of order 1, 2, 5, and 20, and corresponding intensity profiles through the center of the filters (the size in all cases is 1000 × 1000 and the cutoff frequency is 5). Observe how ringing increases as a function of filter order.

Comparison: Ideal LP and BLPF

ILPF

BLPF

D_0=10, 30, 60, 160, 460

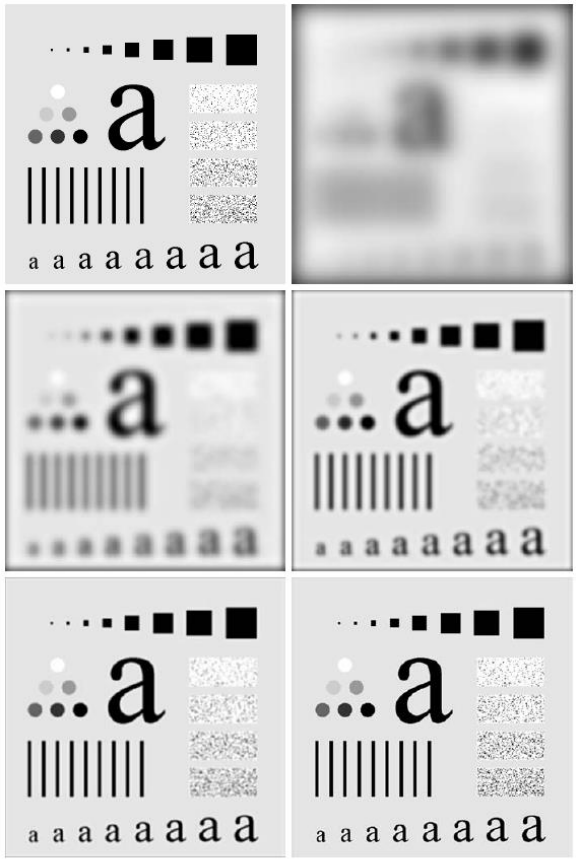


a b
c d
e f

FIGURE 4.42 (a) Original image. (b)–(f) Results of filtering using ILPFs with cutoff frequencies set at radii values 10, 30, 60, 160, and 460, as shown in Fig. 4.41(b). The power removed by these filters was 13, 6.9, 4.3, 2.2, and 0.8% of the total, respectively.

D_0=10, 30, 60, 160, 460

n=2



a b
c d
e f

FIGURE 4.45 (a) Original image. (b)–(f) Results of filtering using BLPFs of order 2, with cutoff frequencies at the radii shown in Fig. 4.41. Compare with Fig. 4.42.

Gaussian LP filter (GLPF)

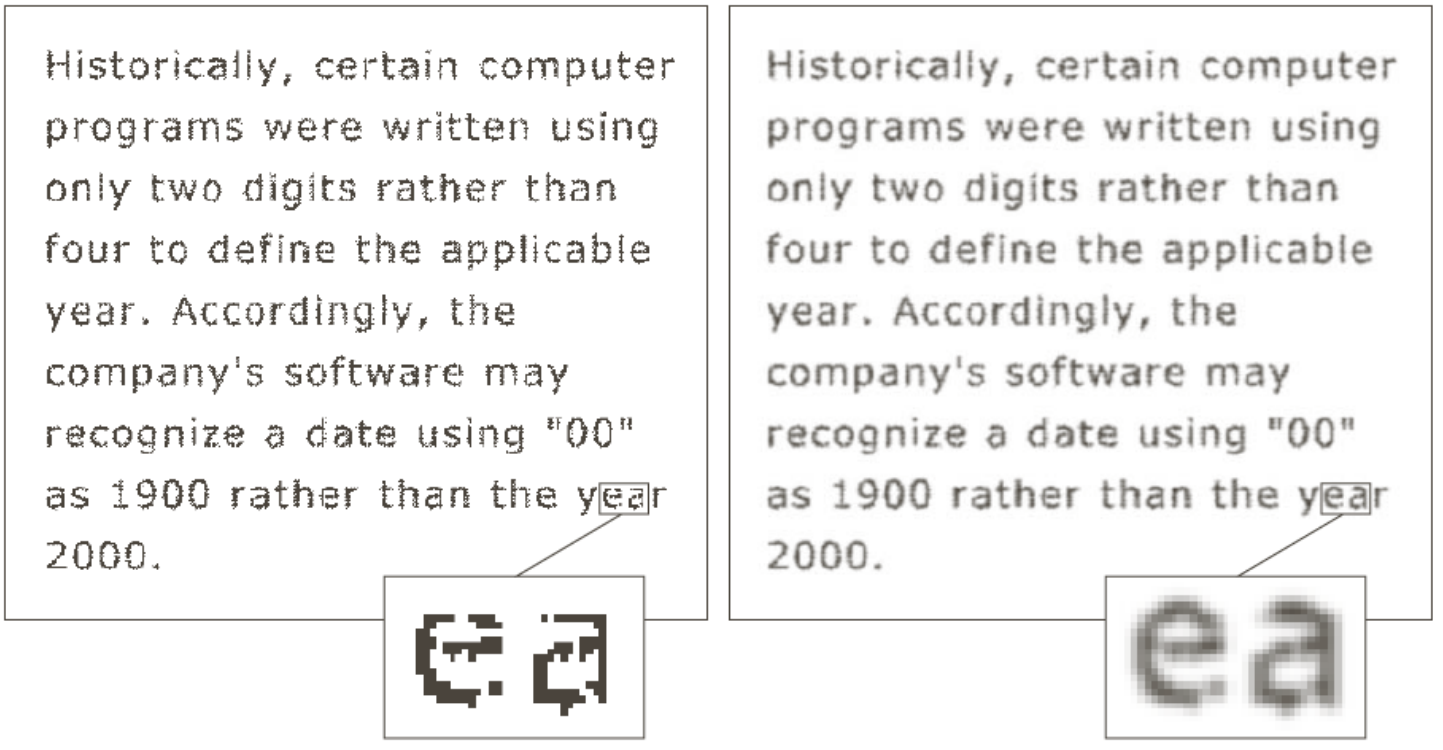
Gaussian Lowpass Filters (GLPF) in two dimensions is given

$$H(u,v) = e^{-(u^2+v^2)/2\sigma^2}$$

By letting $\sigma = D_0$

$$H(u,v) = e^{-(u^2+v^2)/2{D_0}^2}$$

Example: smoothing by GLPF (1)



a b

FIGURE 4.49
(a) Sample text of low resolution (note broken characters in magnified view). (b) Result of filtering with a GLPF (broken character segments were joined).

Examples of smoothing by GLPF (2)

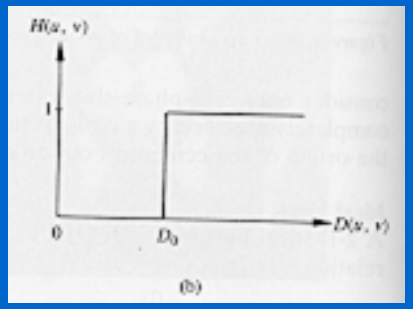
a b c

FIGURE 4.50 (a) Original image (784 × 732 pixels). (b) Result of filtering using a GLPF with $D_0 = 100$. (c) Result of filtering using a GLPF with $D_0 = 80$. Note the reduction in fine skin lines in the magnified sections in (b) and (c).

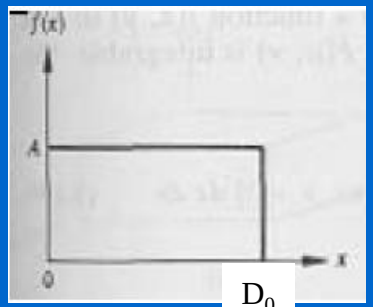
High-Pass filtering

- A high-pass filter can be obtained from a low-pass filter using:

$$H_{HP}(u,v) = 1 - H_{LP}(u,v)$$

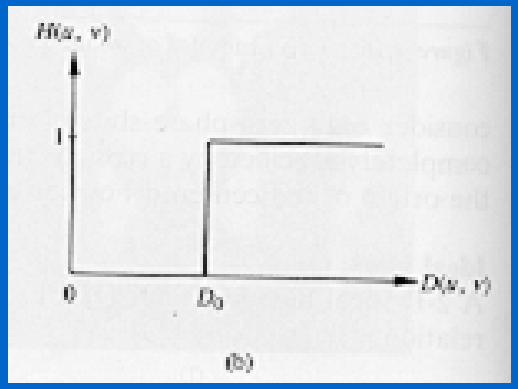


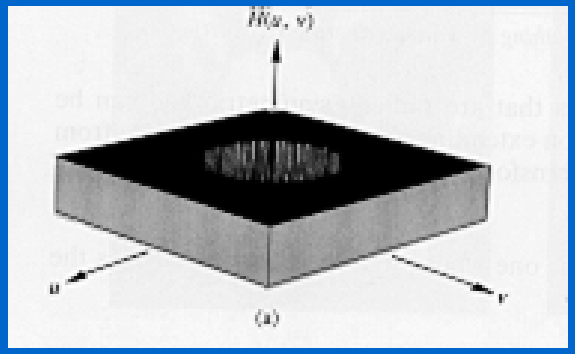
= 1 -



High-pass filtering (cont'd)

- Preserves high frequencies, attenuates low frequencies.





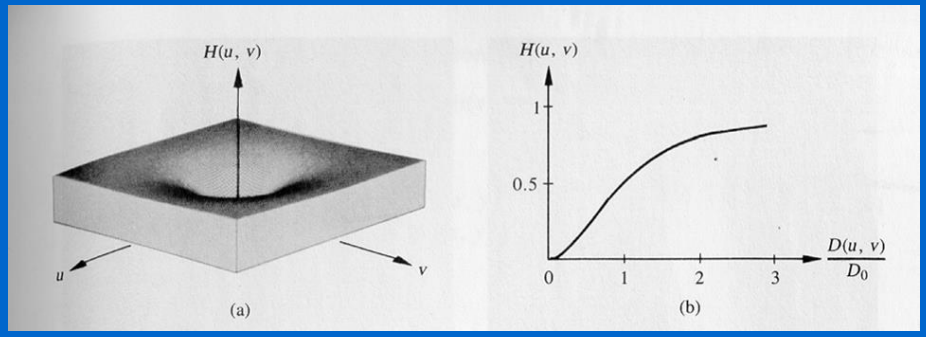
$$H(u,v) = \begin{cases} 1 & \text{if} u \geq D_0 \\ 0 & otherwise \end{cases}$$

$$H(u,v) = \begin{cases} 1 & \text{if} u^2 + v^2 \geq {D_0}^2 \\ 0 & otherwise \end{cases}$$

Butterworth high pass filter (BHPF)

- In practice, we use filters that attenuate low frequencies smoothly (e.g., **Butterworth** HP filter) → less ringing effect

$$H(u,v) = \frac{1}{1 + [D_0/\sqrt{u^2 + v^2}]^{2n}}$$



Comparison: IHPF and BHPF

IHPF

a b c

FIGURE 4.54 Results of highpass filtering the image in Fig. 4.41(a) using an IHPF with $D_0 = 30, 60$, and 160.

D_0=30,60,160

BHPF

a b c

FIGURE 4.55 Results of highpass filtering the image in Fig. 4.41(a) using a BHPF of order 2 with $D_0 = 30, 60$, and 160, corresponding to the circles in Fig. 4.41(b). These results are much smoother than those obtained with an IHPF.

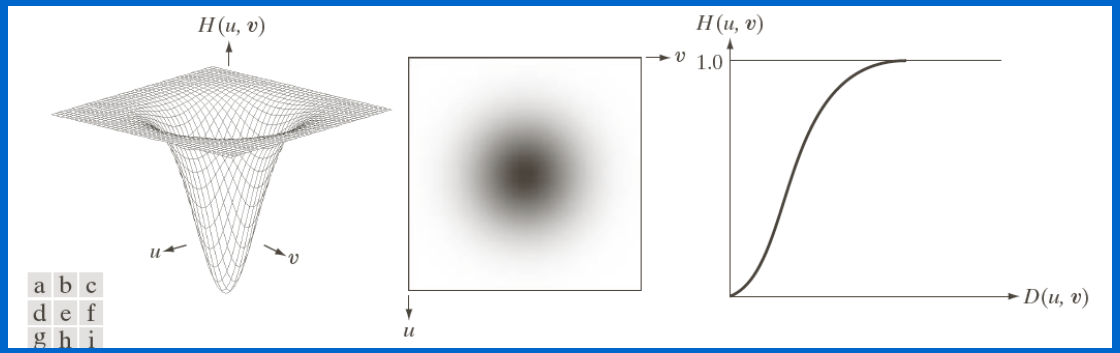
D_0=30,60,160

n=2

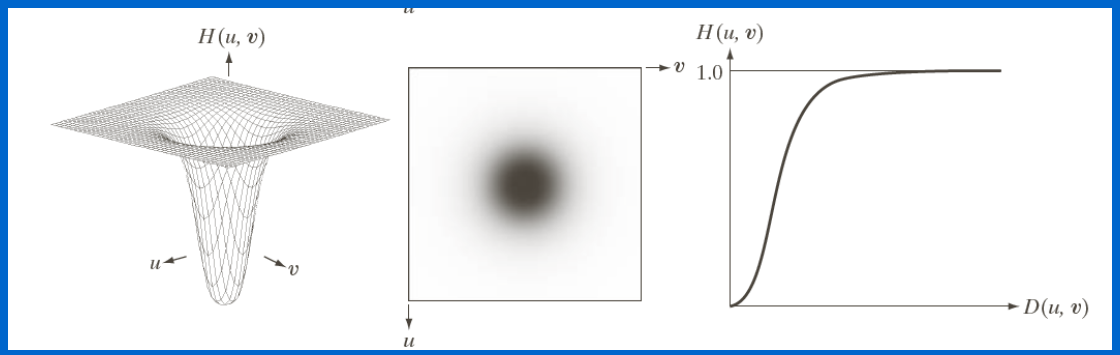
Gaussian HP filter

A 2-D Gaussian highpass filter (GHPL) is defined as

$$H(u,v) = 1 - e^{-(u^2+v^2)/2D_0^2}$$



GHPF



BHPF

Comparison: BHPF and GHPF

BHPF

a b c

FIGURE 4.55 Results of highpass filtering the image in Fig. 4.41(a) using a BHPF of order 2 with $D_0 = 30, 60,$ and 160, corresponding to the circles in Fig. 4.41(b). These results are much smoother than those obtained with an IHPF.

D_0=30,60,160

n=2

GHPF

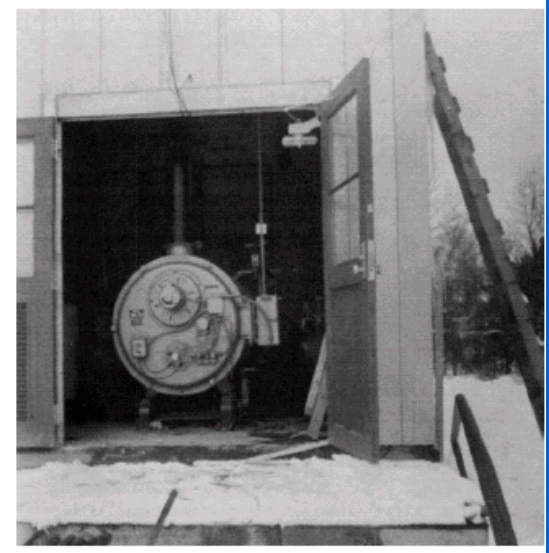
a b c

FIGURE 4.56 Results of highpass filtering the image in Fig. 4.41(a) using a GHPF with $D_0 = 30, 60,$ and 160, corresponding to the circles in Fig. 4.41(b). Compare with Figs. 4.54 and 4.55.

D_0=30,60,160

Homomorphic filtering

- Many times, we want to remove shading effects from an image (i.e., due to uneven illumination)
  - Enhance high frequencies
  - Attenuate low frequencies but preserve fine detail.



Homomorphic Filtering (cont'd)

- Consider the following model of image formation:

$$f(x, y) = i(x, y)\, r(x, y)$$

i(x,y): illumination
r(x,y): reflection

- In general, the illumination component i(x,y) varies **slowly** and affects **low** frequencies mostly.
- In general, the reflection component r(x,y) varies **faster** and affects **high** frequencies mostly.

IDEA: separate low frequencies due to i(x,y) from high frequencies due to r(x,y)

How are frequencies mixed together?

- Low and high frequencies from **i(x,y)** and **r(x,y)** are mixed together.

$$f(x,y) = i(x,y)\, r(x,y) \quad \Rightarrow \quad F(u,v) = I(u,v) * R(u,v)$$

- When applying filtering, it is difficult to handle low/high frequencies separately.

$$F(u,v)H(u,v) = [I(u,v) * R(u,v)]H(u,v)$$

Can we separate them?

- Idea:

Take the ln( ) of $f(x,y) = i(x,y)\, r(x,y)$

$$ln(f(x,y)) = ln(i(x,y)) + ln(r(x,y))$$

Steps of Homomorphic Filtering

(1) Take $ln(f(x,y)) = ln(i(x,y)) + ln(r(x,y))$

(2) Apply FT: $F(ln(f(x,y))) = F(ln(i(x,y))) + F(ln(r(x,y)))$

or $Z(u,v) = Illum(u,v) + Refl(u,v)$

(3) Apply H(u,v)

$$Z(u,v)H(u,v) = Illum(u,v)H(u,v) + Refl(u,v)H(u,v)$$

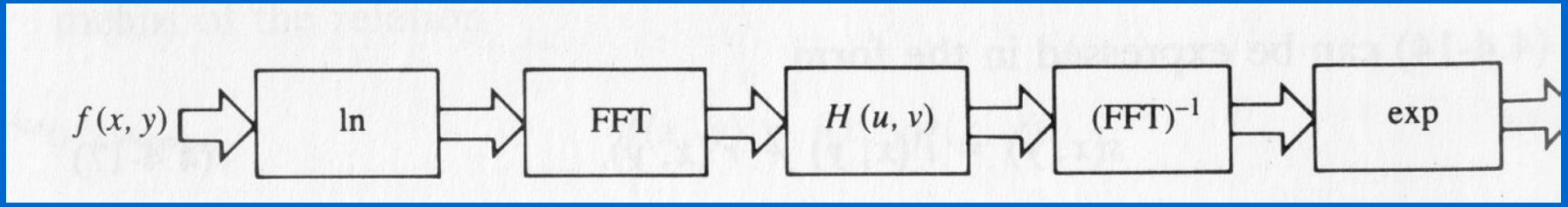
Steps of Homomorphic Filtering (cont'd)

(4) Take Inverse FT:

$$F^{-1}(Z(u,v)H(u,v)) = F^{-1}(Illum(u,v)H(u,v)) + F^{-1}(Refl(u,v)H(u,v))$$

or $$s(x,y) = i'(x,y) + r'(x,y)$$

(5) Take exp( ) $$e^{s(x,y)} = e^{i'(x,y)} e^{r'(x,y)}$$ or $$g(x,y) = i_0(x,y) r_0(x,y)$$



Example using high-frequency emphasis

$$H(u,v) = (\gamma_H - \gamma_L)\left[1 - e^{-c\left[(u^2+v^2)/D_0^2\right]}\right] + \gamma_L$$

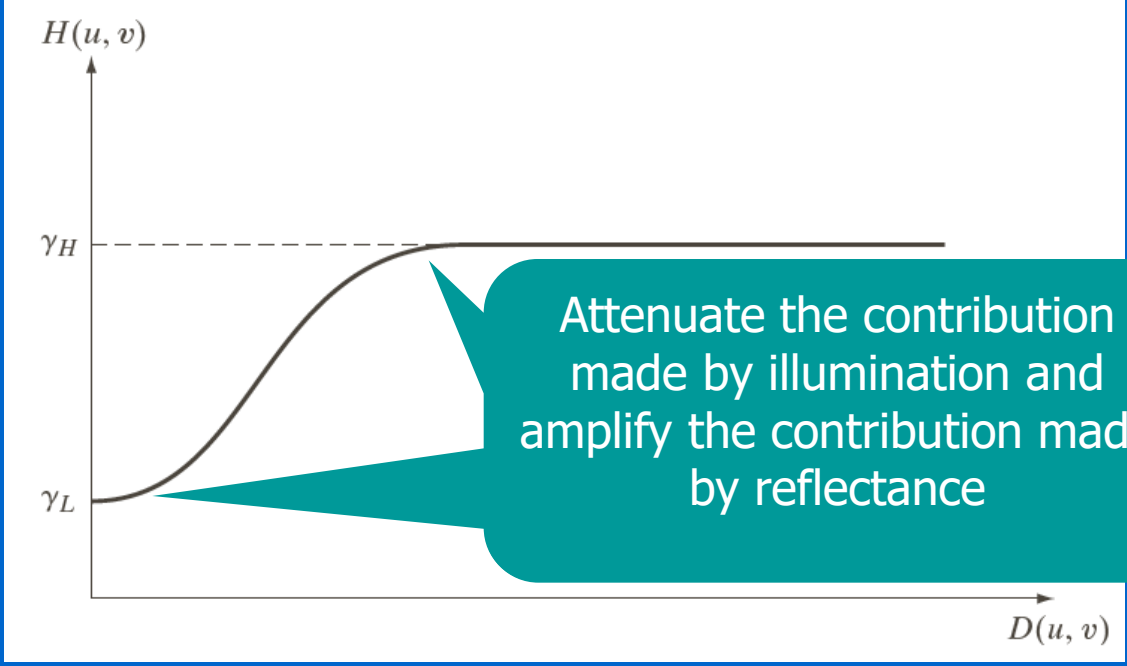
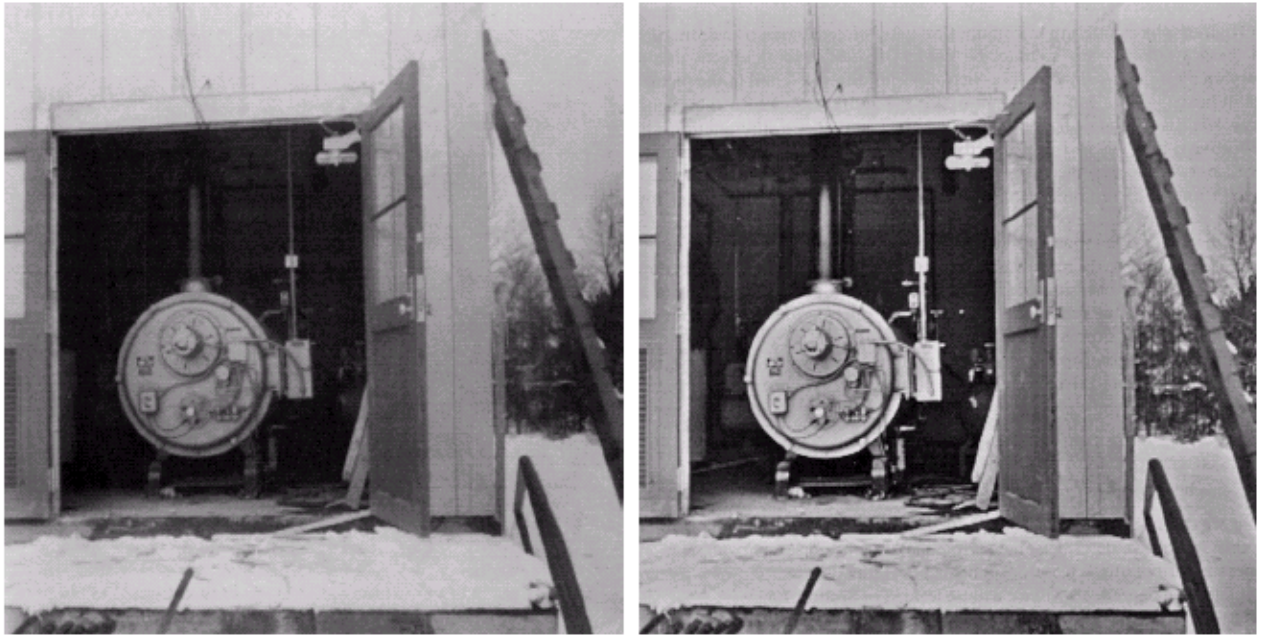


FIGURE 4.61 Radial cross section of a circularly symmetric homomorphic filter function. The vertical axis is at the center of the frequency rectangle and $D(u, v)$ is the distance from the center.

Homomorphic Filtering: Example

a b

FIGURE 4.33
(a) Original image. (b) Image processed by homomorphic filtering (note details inside shelter). (Stockham.)



Homomorphic Filtering: Example

$\gamma_L = 0.25$

$\gamma_H = 2$

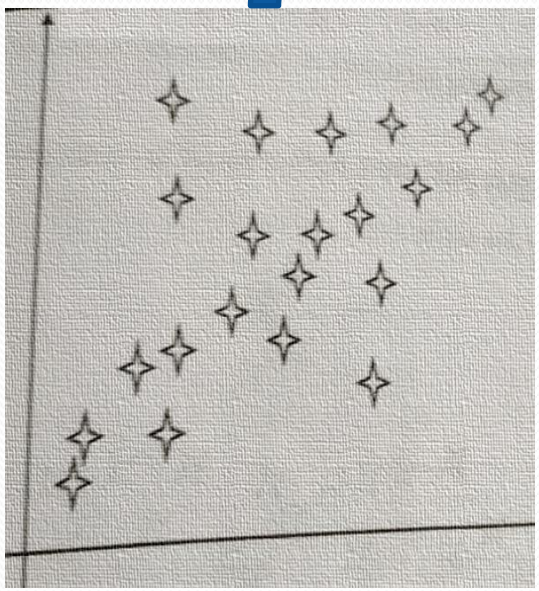
$c = 1$

$D_0 = 80$

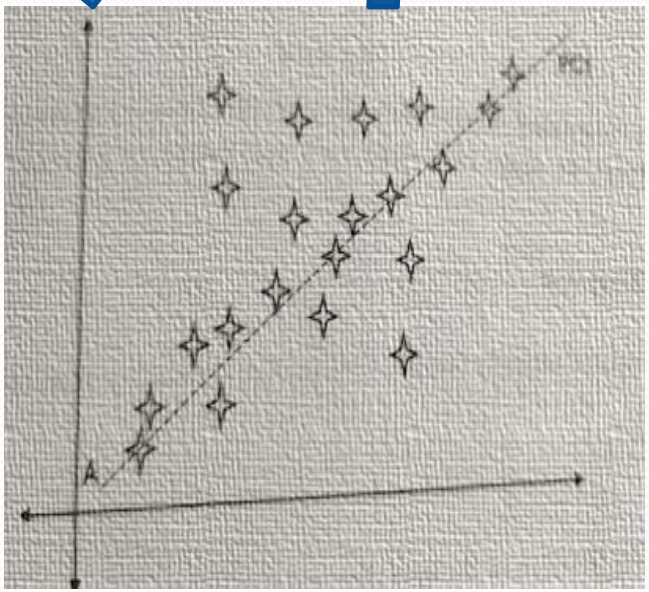
a b

FIGURE 4.62
(a) Full body PET scan. (b) Image enhanced using homomorphic filtering. (Original image courtesy of Dr. Michael E. Casey, CTI PET Systems.)

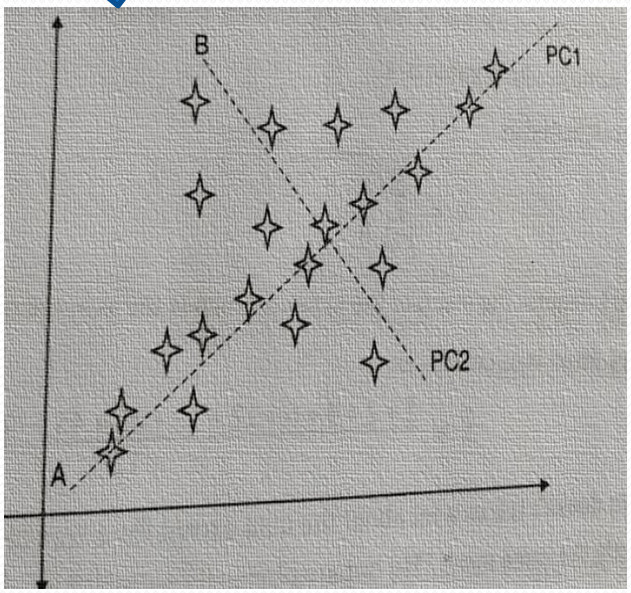
PCA Steps:



Centre the data by subtracting off the mean



Choose the direction with largest variation, place an axis in that direction



Look at the remaining variations, find another axis that is that is orthogonal to the first axis and covers maximum variations

Standard deviation

- *Standard deviation*, is simply the square root of the average square distance of data points to the mean. In the equation below, the numerator contains the sum of the differences between each datapoint and the mean, and the denominator is simply the number of data points (minus one), producing the average distance.

$$s = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)}}$$

Variance

- Variance is simply standard deviation squared, and is often expre

$$var(X) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(X_i - \bar{X})}{(n-1)}$$

- For both variance and standard deviation, squaring the differences between data points and the mean makes them positive, so that values above and below the mean don't cancel each other out.

Covariance

Variance and Covariance are a measure of the "spread" of a set of points around their center of mass (mean)

- Variance – measure of the deviation from the mean for points in one dimension e.g. heights
- Covariance as a measure of how much each of the dimensions vary from the mean with respect to each other.
- Covariance is measured between 2 dimensions to see if there is a relationship between the 2 dimensions e.g. number of hours studied & marks obtained.
- The covariance between one dimension and itself is the variance

Covariance

- So, if you had a 3-dimensional data set (x,y,z), then you could measure the covariance between the x and y dimensions, the y and z dimensions, and the x and z dimensions.
- Measuring the covariance between x and x , or y and y , or z and z would give you the variance of the x , y and z dimensions respectively

$$\text{covariance}(X,Y) = \frac{\sum_{i=1}^{n} (\bar{X}_i - X)(\bar{Y}_i - Y)}{(n-1)}$$

Covariance Matrix

Representing Covariance between dimensions as a matrix e.g. for 3 dimensions:

$$C = \begin{bmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{bmatrix}$$

- Diagonal is the variances of x, y and z
- cov(x,y) = cov(y,x) hence matrix is symmetrical about the diagonal
- N-dimensional data will result in NxN covariance matrix

Covariance

- Exact value is not as important as it's sign.
- (+) A positive value of covariance indicates both dimensions increase or decrease together e.g. as the number of hours studied increases, the marks in that subject increase.
- (-) A negative value indicates while one increases the other decreases, or vice-versa e.g. time spent on social media vs performance in exam.
- (zero) If covariance is zero: the two dimensions are independent of each other e.g. heights of students vs the marks obtained in a subject

Covariance

Why bother with calculating covariance when we could just plot the 2 values to see their relationship?

Ans:

Covariance calculations are used to find relationships between dimensions in high dimensional data sets (usually greater than 3) where visualization is difficult.

Eigen vectors

- $M = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 3 \end{bmatrix} = \begin{bmatrix} 11 \\ 5 \end{bmatrix}$ The resulting vector is not integer multiple of the original vector
- $N = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 12 \\ 8 \end{bmatrix} = 4 \times \begin{bmatrix} 3 \\ 2 \end{bmatrix}$ The resulting vector is exactly 4 times the original vector. This is called eigenvector.

- *We can transform and change matrices into new vectors by multiplying a matrix with a vector. The multiplication of the matrix by a vector computes a new vector. This is the transformed vector.*

Eigenvector

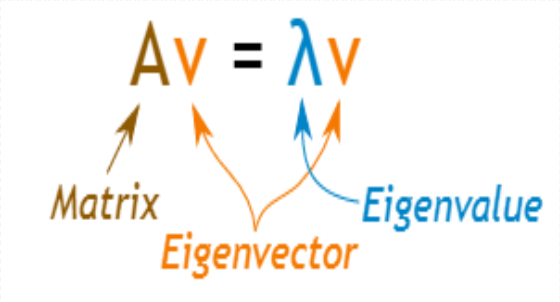
- If the new transformed vector is just a scaled form of the original vector then the original vector is known to be an **eigenvector** of the original matrix
- An eigenvector is a vector that does not change when a transformation is applied to it, except that it becomes a scaled version of the original vector.
- **Eigenvalue**—The scalar that is used to transform (stretch) an Eigenvector.

Eigenvector and Eigenvalue

- An eigenvector **does not change direction** in a transformation:
- In this shear mapping the red arrow changes direction, but the blue arrow does not. The blue arrow is an eigenvector of this shear mapping because it does not change direction, and since its length is unchanged, its eigenvalue is 1.
- **1** means no change,
- **2** means doubling in length,
- **–1** means pointing backwards along the eigenvalue's direction

The Mathematics

- For a square matrix **A**, an Eigenvector and Eigenvalue make this equation true:



Av gives us:

$$\begin{bmatrix} -6 & 3 \\ 4 & 5 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} = \begin{bmatrix} -6\times1+3\times4 \\ 4\times1+5\times4 \end{bmatrix} = \begin{bmatrix} 6 \\ 24 \end{bmatrix}$$

λv gives us :

$$6 \begin{bmatrix} 1 \\ 4 \end{bmatrix} = \begin{bmatrix} 6 \\ 24 \end{bmatrix}$$

Yes they are equal! So Av = λv as promised.

How do we find these eigen things?

We start by finding the **eigenvalue**:

we know this equation must be true: $Av = \lambda v$

Now let us put in an identity matrix so we are dealing with matrix-vs-matrix:

$$Av = \lambda Iv$$

Bring all to left hand side:

$$Av - \lambda Iv = 0$$

If **v** is non-zero then we can solve for λ using just the determinant:

$$| A - \lambda I | = 0$$

Let's try that equation on our previous example:

Example: Solve for λ:

Start with $|A - \lambda I| = 0$

$$\left|\begin{bmatrix} -6 & 3 \\ 4 & 5 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right| = 0$$

Which is:

$$\begin{vmatrix} -6-\lambda & 3 \\ 4 & 5-\lambda \end{vmatrix} = 0$$

Calculating that determinant gets:

$$(-6-\lambda)(5-\lambda) - 3 \times 4 = 0$$

Which then gets us this

$$\lambda^2 + \lambda - 42 = 0$$

$$\lambda = -7 \text{ or } 6$$

Example (continued): Find the Eigenvector for the Eigenvalue $\lambda = 6$:

Start with:

$$Av = \lambda v$$

Put in the values we know:

$$\begin{bmatrix} -6 & 3 \\ 4 & 5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 6 \begin{bmatrix} x \\ y \end{bmatrix}$$

After multiplying we get these two equations:

$$\begin{aligned} -6x + 3y &= 6x \\ 4x + 5y &= 6y \end{aligned}$$

Bringing all to left hand side:

$$\begin{aligned} -12x + 3y &= 0 \\ 4x - 1y &= 0 \end{aligned}$$

Either equation reveals that **y = 4x**, so the **eigenvector** is any non-zero multiple of this:

$$\begin{bmatrix} 1 \\ 4 \end{bmatrix}$$

And we get the solution shown at the top of the page:

$$\begin{bmatrix} -6 & 3 \\ 4 & 5 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} = \begin{bmatrix} -6\times1+3\times4 \\ 4\times1+5\times4 \end{bmatrix} = \begin{bmatrix} 6 \\ 24 \end{bmatrix}$$

... and also ...

$$6 \begin{bmatrix} 1 \\ 4 \end{bmatrix} = \begin{bmatrix} 6 \\ 24 \end{bmatrix}$$

So $Av = \lambda v$

PCA

Principal components analysis (PCA) is a technique that can be used to simplify a dataset

- It is a linear transformation that chooses a new coordinate system for the data set such that greatest variance by any projection of the data set comes to lie on the first axis (then called the first principal component),the second greatest variance on the second axis, and so on.
- PCA can be used for reducing dimensionality by eliminating the later principal components.

Steps Involved in the PCA

Step 1: Standardize the dataset.

Step 2: Calculate the covariance matrix for the features in the dataset.

Step 3: Calculate the eigenvalues and eigenvectors for the covariance matrix.

Step 4: Sort eigenvalues and their corresponding eigenvectors.

Step 5: Pick k eigenvalues and form a matrix of eigenvectors.

Step 6: Transform the original matrix.

Example

1. Standardize the Dataset

- Assume we have the below dataset which has 4 features and a total of 5 training examples.

f1	f2	f3	f4
1	2	3	4
5	5	6	7
1	4	2	3
5	3	2	1
8	1	2	2

Example

- First, we need to standardize the dataset and for that, we need to calculate the mean and standard deviation for each feature.

f1	f2	f3	f4
1	2	3	4
5	5	6	7
1	4	2	3
5	3	2	1
8	1	2	2

	f1	f2	f3	f4
μ =	4	3	3	3.4
σ =	3	1.58114	1.73205	2.30217

$$x_{new} = \frac{x - \mu}{\sigma}$$

After applying the formula for each feature in the dataset is transformed as

f1	f2	f3	f4
-1	-0.63246	0	0.26062
0.33333	1.26491	1.73205	1.56374
-1	0.63246	-0.57735	-0.17375
0.33333	0	-0.57735	-1.04249
1.33333	-1.26491	-0.57735	-0.60812

$$s = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)}}$$

Example

- 2. Calculate the covariance matrix for the whole dataset

The formula to calculate the covariance matrix:

-

For Population

$$Cov(x,y) = \frac{\sum (x_i - \bar{x}) * (y_i - \bar{y})}{N}$$

For Sample

$$Cov(x,y) = \frac{\sum (x_i - \bar{x}) * (y_i - \bar{y})}{(N-1)}$$

	f1	f2	f3	f4
f1	var(f1)	cov(f1,f2)	cov(f1,f3)	cov(f1,f4)
f2	cov(f2,f1)	var(f2)	cov(f2,f3)	cov(f2,f4)
f3	cov(f3,f1)	cov(f3,f2)	var(f3)	cov(f3,f4)
f4	cov(f4,f1)	cov(f4,f2)	cov(f4,f3)	var(f4)

*Since we have standardized the dataset, so the **mean for each feature is 0** and the standard deviation is 1.*

Example

f1	f2	f3	f4
-1	-0.63246	0	0.26062
0.33333	1.26491	1.73205	1.56374
-1	0.63246	-0.57735	-0.17375
0.33333	0	-0.57735	-1.04249
1.33333	-1.26491	-0.57735	-0.60812

	f1	f2	f3	f4
f1	var(f1)	cov(f1,f2)	cov(f1,f3)	cov(f1,f4)
f2	cov(f2,f1)	var(f2)	cov(f2,f3)	cov(f2,f4)
f3	cov(f3,f1)	cov(f3,f2)	var(f3)	cov(f3,f4)
f4	cov(f4,f1)	cov(f4,f2)	cov(f4,f3)	var(f4)

- $var(f_1) = ((-1.0-0)^2 + (0.33-0)^2 + (-1.0-0)^2 + (0.33-0)^2 + (1.33-0)^2)/5$
  var (f1) = 0.8
- $cov(f_1,f_2) = ((-1.0-0)*(-0.632456-0) +$
  $(0.33-0)*(1.264911-0) +$
  $(-1.0-0)*(0.632456-0) +$
  $(0.33-0)*(0.000000-0) +$
  $(1.33-0)*(-1.264911-0))/5$
  cov(f1,f2) = -0.25298

Example

- In the similar way be can calculate the other covariance [illegible] ow covariance

	f1	f2	f3	f4
f1	0.8	-0.25298	0.03849	-0.14479
f2	-0.25298	0.8	0.51121	0.4945
f3	0.03849	0.51121	0.8	0.75236
f4	-0.14479	0.4945	0.75236	0.8

Example

- 3. Calculate eigenvalues and eigen vectors.

$Av-\lambda Iv = 0$;

$(A-\lambda I)v = 0$

Since we already know v is a non- zero vector, only way this equation can be equal to zero, if

- $\det(A-\lambda I) = 0$

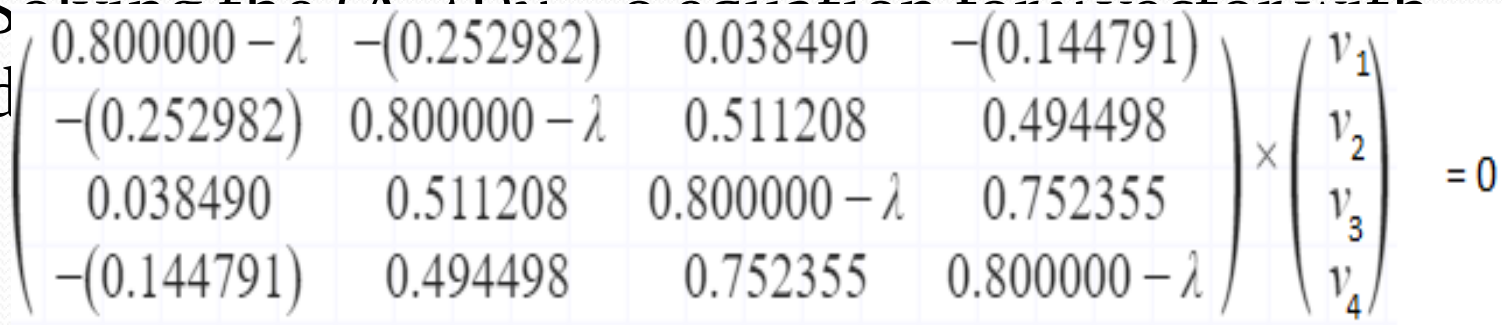
Example

$$\begin{bmatrix} 0.8 & -0.25298 & 0.03849 & -0.14479 \\ -0.25298 & 0.8 & 0.51121 & 0.4945 \\ 0.03849 & 0.51121 & 0.8 & 0.75236 \\ -0.14479 & 0.4945 & 0.75236 & 0.8 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = 0$$

	f1	f2	f3	f4
f1	0.8 - λ	-0.25298	0.03849	-0.14479
f2	-0.25298	0.8- λ	0.51121	0.4945
f3	0.03849	0.51121	0.8 - λ	0.75236
f4	-0.14479	0.4945	0.75236	0.8 - λ

- Solving the above equation $|M| = 0$
- ***λ = 2.51579324 , 1.0652885 , 0.39388704 , 0.02503121***

Example

- **Eigenvectors:**
- Solving the $(A-\lambda I)v = 0$ equation for vector with
- 

$$\begin{pmatrix} 0.800000-\lambda & -(0.252982) & 0.038490 & -(0.144791) \\ -(0.252982) & 0.800000-\lambda & 0.511208 & 0.494498 \\ 0.038490 & 0.511208 & 0.800000-\lambda & 0.752355 \\ -(0.144791) & 0.494498 & 0.752355 & 0.800000-\lambda \end{pmatrix} \times \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} = 0$$

For $\lambda = 2.51579324$, *solving the above equation using Cramer's rule, the values for v vector are*

v1 = 0.16195986

v2 = -0.52404813

v3 = -0.58589647

v4 = -0.59654663

Example

- Going by the same approach, we can calculate the eigen vectors for the other eigen values. We can from a matrix using the **eig**

e1	e2	e3	e4
0.161960	-0.917059	-0.307071	0.196162
-0.524048	0.206922	-0.817319	0.120610
-0.585896	-0.320539	0.188250	-0.720099
-0.596547	-0.115935	0.449733	0.654547

- ***4.* Sort eigenvalues and their corresponding eigenvectors.**
- Since eigenvalues are already sorted in this case so no need to sort them again.

Example

5. Pick k eigenvalues and form a matrix of eigenvectors

- If we choose the top 2 eigenvectors, the matrix will look like this:

e1	e2
0.161960	-0.917059
-0.524048	0.206922
-0.585896	-0.320539
-0.596547	-0.115935

6. Transform the original matrix.

- Fe Data
-

f1	f2	f3	f4
-1.000000	-0.632456	0.000000	0.260623
0.333333	1.264911	1.732051	1.563740
-1.000000	0.632456	-0.577350	-0.173749
0.333333	0.000000	-0.577350	-1.042493
1.333333	-1.264911	-0.577350	-0.608121

(5,4)

*

e1	e2
0.161960	-0.917059
-0.524048	0.206922
-0.585896	-0.320539
-0.596547	-0.115935

(4,2)

=

nf1	nf2
0.014003	0.755975
-2.556534	-0.780432
-0.051480	1.253135
1.014150	0.000239
1.579861	-1.228917

(5,2)

```python
import numpy as np
import pandas as pd
A = np.matrix([[1,2,3,4],
               [5,5,6,7],
               [1,4,2,3],
               [5,3,2,1],
               [8,1,2,2]])

df = pd.DataFrame(A,columns  = ['f1','f2','f3','f4'])
df_std  = (df - df.mean()) / (df.std())
n_components = 2
from sklearn.decomposition import PCA
pca = PCA(n_components=n_components)
principalComponents = pca.fit_transform(df_std)
principalDf = pd.DataFrame(data=principalComponents,columns=['nf'+str(i+1) for i in range(n_components)])
print(principalDf)
```

```
        nf1       nf2
0 -0.014003  0.755975
1  2.556534 -0.780432
2  0.051480  1.253135
3 -1.014150  0.000239
4 -1.579861 -1.228917
```

Wavelet Transform

- https://medium.com/@koushikc2000/2d-discrete-wavelet-transformation-and-its-applications-in-digital-image-processing-using-matlab-1f5c68672de3