CHAPTER 5:

DISCRETE FOURIER TRANSFORM (DFT)

Lesson #19: DTFT of DT periodic signals

Lesson #20: DFT and Inverse DFT

Lesson #21: DFT properties

Lesson #22: Fast Fourier Transform (FFT)

Lesson #23: Applications of DFT/FFT

Lesson #24: Using of DFT/FFT

Lecture #19

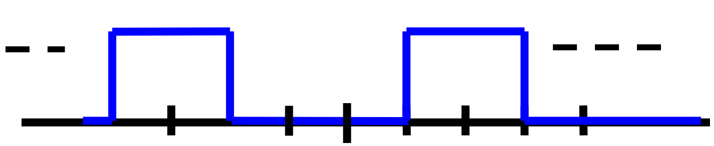
DTFT of periodic signals

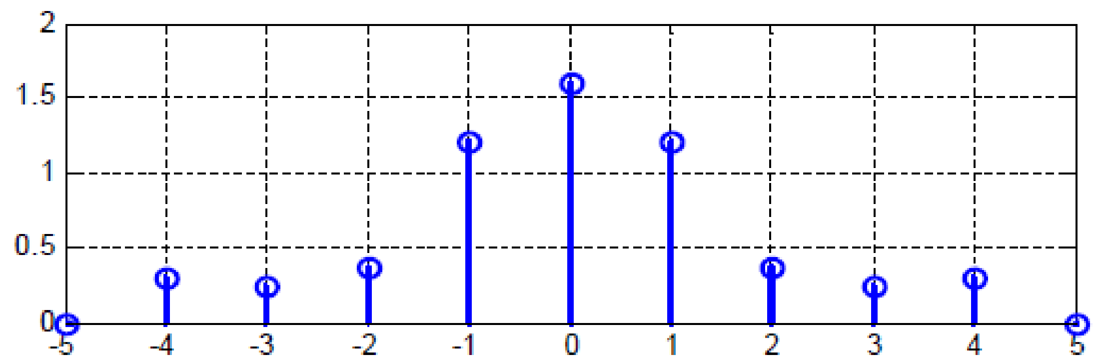
1. **Review of Fourier Transforms (FT)**
2. Fourier Series expansions
3. DTFT of periodic signals

FT of CT periodic signals

$$x(t)=\sum_{k=-\infty}^{\infty}a_k e^{jk\omega_0 t}; \quad a_k=\frac{1}{T}\int_T x(t)e^{-jk\omega_0 t}dt; \quad \omega_0=\frac{2\pi}{T}$$

$$X(\omega)=2\pi\sum_{k=-\infty}^{\infty}a_k\delta(\omega-k\omega_o)$$





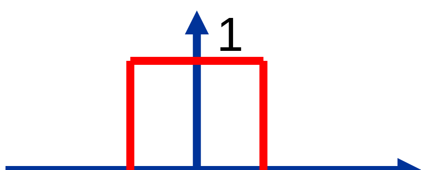
Continuous and periodic in time domain

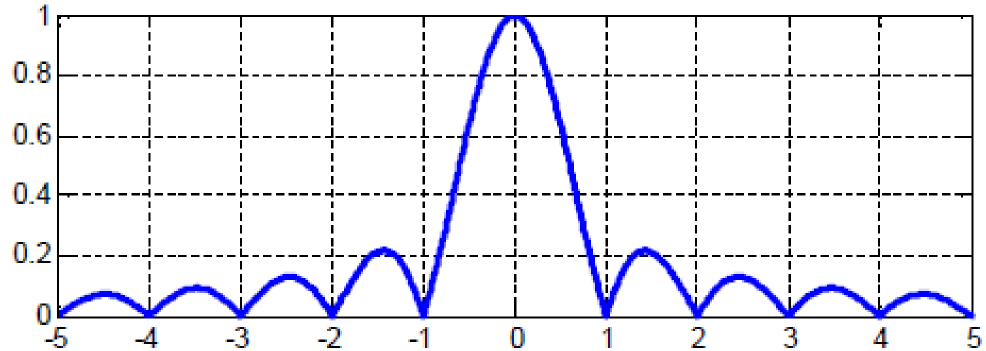
Discrete and aperiodic in frequency domain

FT of CT aperiodic signals

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt$$

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\omega)e^{j\omega t}d\omega$$





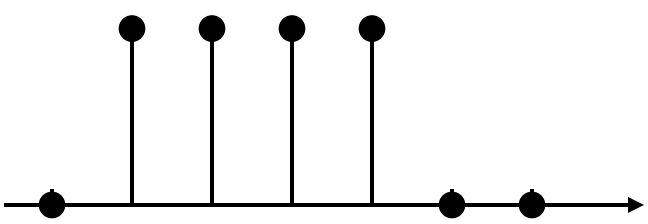
Continuous and aperiodic in time domain

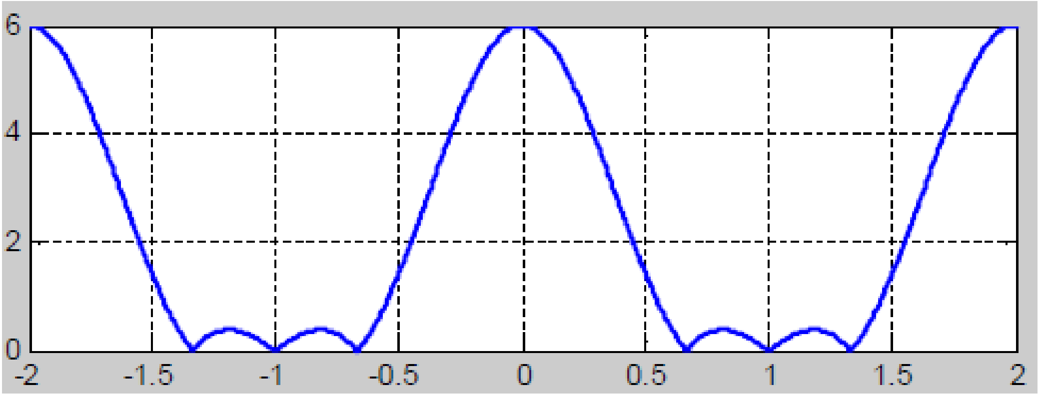
Continuous and aperiodic in frequency domain

FT of DT aperiodic signals

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\Omega n}$$

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega) e^{j\Omega n} d\Omega$$

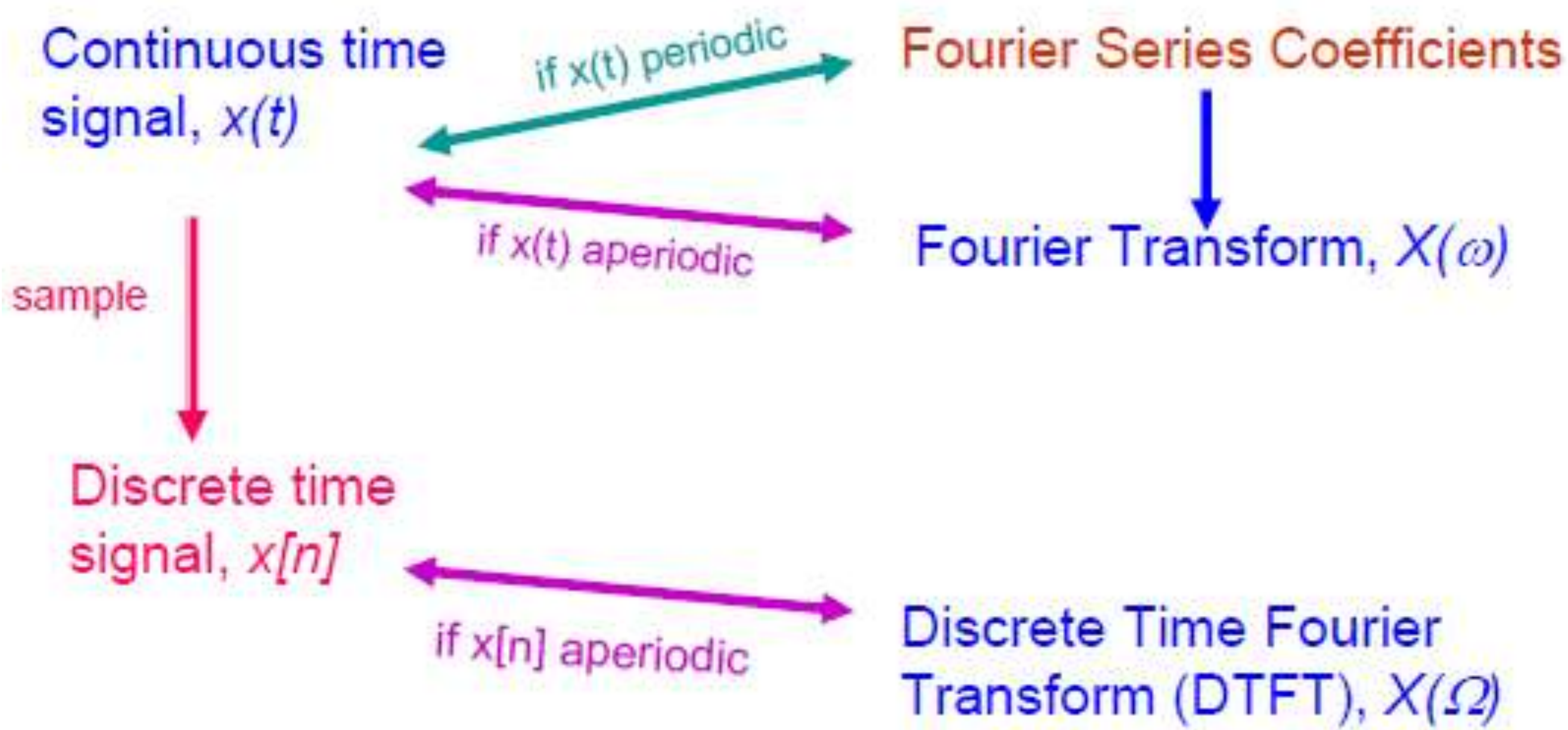




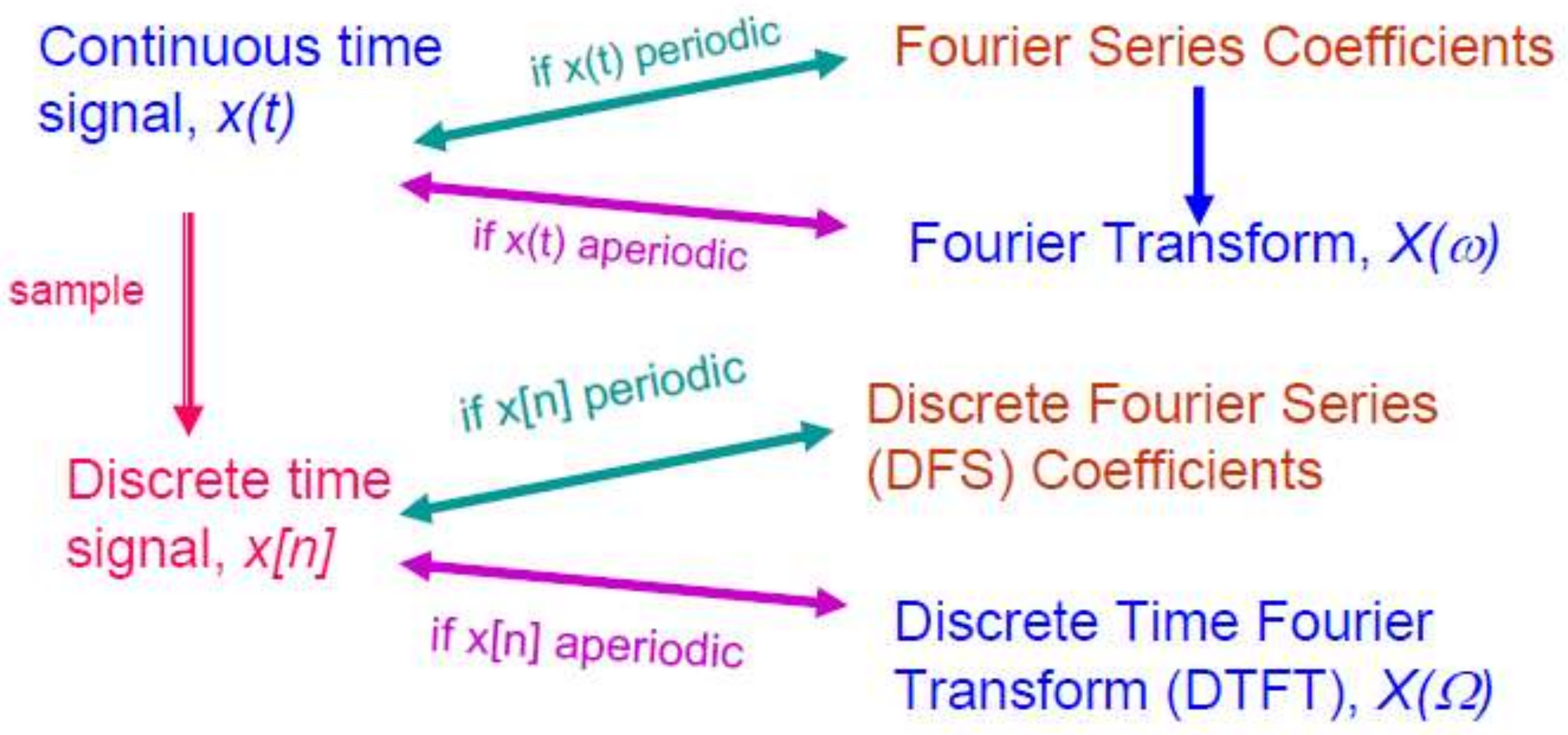
Discrete and aperiodic in time domain

Continuous and periodic in frequency with period 2п

Recapitulation



Can guess?



Continuous time signal, x(t)
if x(t) periodic
Fourier Series Coefficients
if x(t) aperiodic
Fourier Transform, X(ω)
sample
if x[n] periodic
Discrete Fourier Series (DFS) Coefficients
Discrete time signal, x[n]
if x[n] aperiodic
Discrete Time Fourier Transform (DTFT), X(Ω)

Lecture #19

DTFT of periodic signals

1. Review of Fourier Transforms (FT)

2. Fourier Series expansions

3. DTFT of periodic signals

Fourier Series expansion

- **CT periodic signals with period T:**

$$\boxed{x(t) = \sum_{k=-\infty}^{\infty} a_k e^{jk\omega_0 t}; \quad a_k = \frac{1}{T}\int_T x(t) e^{-jk\omega_0 t} dt} \qquad \omega_0 = \frac{2\pi}{T}$$

- **DT periodic signals with period N:**

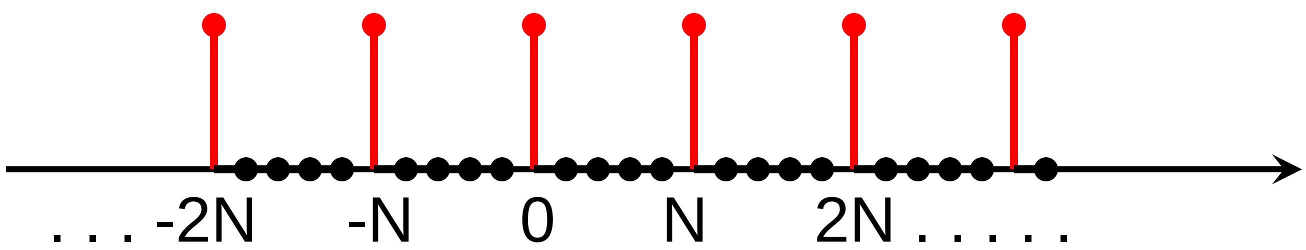
$$\boxed{x[n] = \sum_{k\in<N>} a_k e^{jk\Omega_0 n}; \quad a_k = \frac{1}{N}\sum_{n\in<N>} x[n] e^{-jk\Omega_0 n}} \qquad \Omega_0 = \frac{2\pi}{N}$$

Note: finite sums over an interval length of one periodic **N**

$$e^{jk\Omega_0 n} = e^{jk\frac{2\pi}{N}n} = e^{j(k+N)\frac{2\pi}{N}n} = e^{j(k+N)\Omega_0 n}$$

Example of Fourier Series expansion

Given a DT periodic signals with period N: $p[n] = \sum_{k=-\infty}^{\infty} \delta[n-kN]$



$$p[n] = \sum_{k=0}^{N-1} a_k e^{jk\frac{2\pi}{N}n}$$

$$a_k = \frac{1}{N} \sum_{n=0}^{N-1} p[n] e^{-jk2\pi n/N} = \frac{1}{N}$$

Lecture #19

DTFT of periodic signals

1. Review of Fourier Transforms (FT)
2. Fourier Series expansions
3. **DTFT of periodic signals**

DTFT of periodic signals

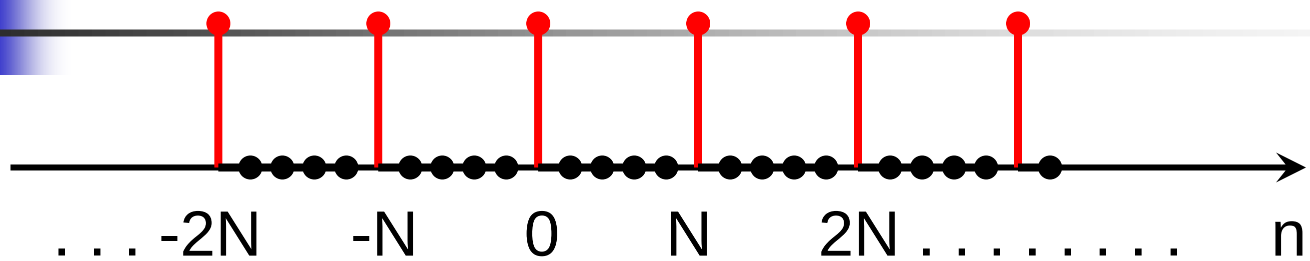
- CT periodic signals with period T:

$$x(t) = \sum_{k=-\infty}^{\infty} a_k e^{jk\omega_0 t} \stackrel{F}{\longleftrightarrow} X(\omega) = 2\pi \sum_{k=-\infty}^{\infty} a_k \delta(\omega - k\omega_0)$$

- DT periodic signals with period N:

$$\boxed{x[n] \stackrel{F}{\longleftrightarrow} X(\Omega) = 2\pi \sum_{k=-\infty}^{\infty} a_k \delta(\Omega - k\Omega_0)}$$

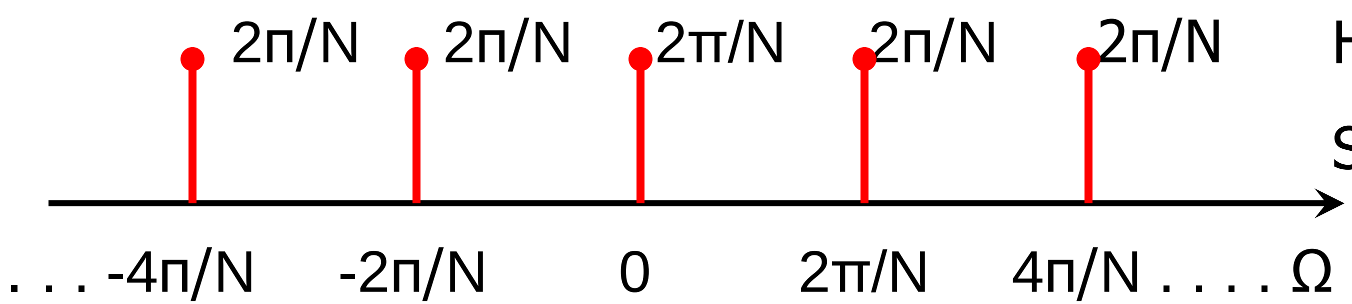
Example of calculating DTFT of periodic signals



Height = 1

Spacing = N

$$P(\Omega) = 2\pi \sum_{k=-\infty}^{\infty} a_k \delta(\Omega - k\Omega_0) = \frac{2\pi}{N} \sum_{k=-\infty}^{\infty} \delta\left(\Omega - k\frac{2\pi}{N}\right)$$

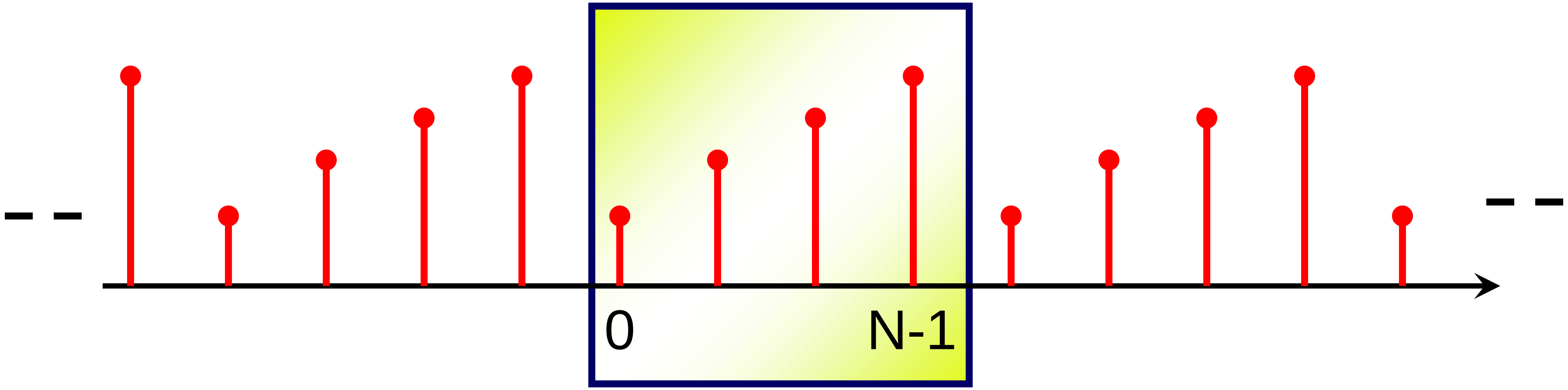


Height = 2п/N

Spacing = 2п/N

Another approach to get DTFT of periodic signals

$x(n)$ is periodic signal; $x_0(n)$ is a part of $x(n)$ that is repeated



$$x_0[n] = \begin{cases} x[n], & 0 \le n \le N-1 \\ 0, & \text{otherwise.} \end{cases}$$

Another approach (cont)

$$x_0[n] = \begin{cases} x[n], & 0 \le n \le N-1 \\ 0, & \text{otherwise.} \end{cases}$$

$$x[n] = \sum_{k=-\infty}^{\infty} x_0[n-kN] = \sum_{k=-\infty}^{\infty} x_0[n] * \delta[n-kN] = x_0[n] * \sum_{k=-\infty}^{\infty} \delta[n-kN]$$

p(n) in previous example

$$x[n] = x_0[n] * p[n] \xleftrightarrow{F} X_0(\Omega) P(\Omega) = X(\Omega)$$

Another approach (cont)

$$x[n] = x_0[n] * p[n] \overset{F}{\longleftrightarrow} X_0(\Omega)P(\Omega) = X(\Omega)$$

$$\boxed{\begin{aligned} X(\Omega) &= X_0(\Omega)\left(\frac{2\pi}{N}\sum_k \delta(\Omega - k\frac{2\pi}{N})\right) \\ &= \frac{2\pi}{N}\sum_k X_0(k\frac{2\pi}{N})\delta(\Omega - k\frac{2\pi}{N}) \end{aligned}}$$

N samples—you are sampling the DTFT at
N equal intervals around the unit circle

→ It has **N** distinct values at k = 0, 1, …, N-1

Inverse DTFT

$$DFT^{-1}\{X(\Omega)\} = x[n] = \frac{1}{2\pi}\int_0^{2\pi} X(\Omega)e^{j\Omega n}d\Omega$$

$$= \frac{1}{2\pi}\int_0^{2\pi}\left[\frac{2\pi}{N}\sum_{k=-\infty}^{\infty} X_0(k\frac{2\pi}{N})\delta(\Omega - k\frac{2\pi}{N})\right]e^{j\Omega n}d\Omega$$

$$= \frac{1}{N}\sum_{k=-\infty}^{\infty} X_0(k\frac{2\pi}{N})\int_0^{2\pi}\delta(\Omega - k\frac{2\pi}{N})e^{j\Omega n}d\Omega = \frac{1}{N}\sum_{k=0}^{N-1} X_0(k\frac{2\pi}{N})e^{j\frac{k2\pi n}{N}}$$

This is obtained from this property of the impulse:

$$\int_a^b f(t)\delta(t - t_0)dt = \begin{cases} f(t_0) & a \le t_0 < b \\ 0 & elsewhere \end{cases}$$

Summary

$$x[n] = x_0[n] * \sum_{k=-\infty}^{\infty} \delta[n - kN]$$

$$X_0(\Omega) = \sum_{n=0}^{N-1} x_0[n] e^{-j\Omega n}$$

$$X(\Omega) = \frac{2\pi}{N} \sum_{k=-\infty}^{\infty} X_0\left(\frac{2\pi k}{N}\right) \delta\left(\Omega - \frac{2\pi k}{N}\right)$$

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X_0\left(\frac{2\pi k}{N}\right) e^{\frac{j2\pi kn}{N}} = \sum_{k=0}^{N-1} a_k e^{\frac{j2\pi kn}{N}}$$

$$a_k = \frac{1}{N} X_0\left(\frac{2\pi k}{N}\right)$$

Procedure to calculate DTFT of periodic signals

Step 1:

Start with $\mathbf{x_0(n)}$ – one period of $\mathbf{x(n)}$**,** with zero everywhere else

Step 2:

Find the DTFT $\mathbf{X_0(\Omega)}$ of the signal $x_0[n]$ above

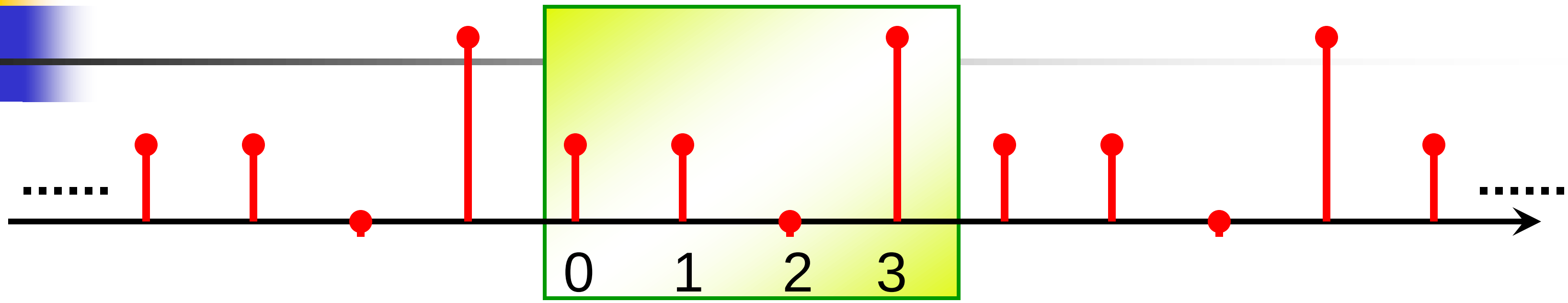
Step 3:

Find $X_0(\Omega)$ at **N** equally spacing frequency points $\mathbf{X_0(k2\pi/N)}$

Step 4:

Obtain the DTFT of x(n): $$\boxed{X(\Omega) = \frac{2\pi}{N}\sum_{k} X_0\left(k\frac{2\pi}{N}\right)\delta\left(\Omega - k\frac{2\pi}{N}\right)}$$

Example of calculating DTFT of periodic signals



$$X_0(\Omega) = \sum_{n=0}^{3} x_0(n) e^{-j\Omega n} = 1 + e^{-j\Omega} + 2e^{-j3\Omega}$$

$$X_o\left(\frac{2\pi k}{4}\right) = 1 + e^{-j\frac{2\pi k}{4}} + 2e^{-3j\frac{2\pi k}{4}} \qquad k = 0,1,2,3$$

k = 0 → $X_0(0)$ = 4; k = 1 → $X_0(1)$ = 1+j

k = 2 → $X_0(2)$ = -2; k = 3 → $X_0(3)$ = 1-j

Example (cont)

So we have $[4, 1+j, -2, 1-j]$

Test with the inverse transform formula

$$x[n] = \frac{1}{N}\sum_{0}^{N-1} X_o\left(\frac{2\pi k}{N}\right)e^{-j\frac{2\pi kn}{N}}$$

$$x[0] = \frac{1}{4}\left[X_o(0) + X_o\left(\frac{\pi}{2}\right) + X_o(\pi) + X_o\left(\frac{3\pi}{2}\right)\right]$$

$$= \frac{1}{4}[4 + 1 + j + -2 + 1 - j] = 1$$

Keep going for the rest—probably wise to use MATLAB to do

yields $[1, 1, 0, 2]$

Example (cont)

$$X(\Omega) = \frac{2\pi}{N} \sum_{k=-\infty}^{\infty} X_o(\frac{2\pi k}{N})\delta(\Omega - \frac{2\pi k}{N})$$

$$= \frac{2\pi}{4} \sum_{k=-\infty}^{\infty} X_o(\frac{2\pi k}{4})\delta(\Omega - \frac{2\pi k}{4})$$

For one period $0 \leq \Omega < 2\pi$

$$\frac{\pi}{2}\left\{4\delta(\Omega) + (1+j)\delta(\Omega - \frac{\pi}{2}) - 2\delta(\Omega - \pi) + (1-j)\delta(\Omega - \frac{3\pi}{2})\right\}$$

DTFT and DFS example

a) Draw the magnitude plot of the FT of a CT sinusoid *x(t),* that has a frequency of *24 Hz*

b) Sample this signal *x(t)* at *40 Hz*. Draw the magnitude plot of the FT of the sampled signal

c) Reconstruct a CT signal from the samples above with a LPF, it will look like a CT sinusoid of what frequency?

d) Consider DT signal that is obtained by the sampling *x(t)* at a sampling frequency of *40 Hz* is *x(n).* Is *x(n)* a periodic signal? If yes, what is its period?

e) Specify the DTFT of the DT signal *x(n).*

DTFT and DFS examples

a) Draw the magnitude plot of the FT of a CT sinusoid *x(t),* that has a frequency of *24 Hz*

DTFT and DFS examples

b) Sample this signal *x(t)* at *40 Hz*. Draw the magnitude plot of the FT of the sampled signal

DTFT and DFS examples

c) Reconstruct a CT signal from the samples above with a LPF, it will look like a CT sinusoid of what frequency?

DTFT and DFS examples

d) Consider DT signal that is obtained by the sampling *x(t)* at a sampling frequency of *40 Hz* is *x(n).* Is *x(n)* a periodic signal? If yes, what is its period?

DTFT and DFS examples

e) Specify the DTFT of the DT signal *x(n).*

Lecture #20

DFT and inverse DFT

1. DFT and inverse DFT formulas

2. DFT and inverse DFT examples

3. Frequency resolution of the DFT

DFT to the rescue!

Could we calculate the **frequency spectrum** of a signal using a **digital computer** with **CTFT/DTFT**?

- Both CTFT and DTFT produce continuous function of frequency → can't calculate an infinite continuum of frequencies using a computer

- Most real-world data is not in the form like $a^n u(n)$

DFT can be used as a FT approximation that can calculate a **finite set of discrete-frequency spectrum values** from a finite set of discrete-time samples of an analog signal

Building the DFT formula

Continuous time signal **x(t)**

sample

Discrete time signal **x(n)**

window

Discrete time signal **x_0(n)**

Finite length

"Window" x(n) is like multiplying the signal by the finite length rectangular window

$$w_R = \begin{cases} 1 & n = 0, 1, \cdots N-1 \\ 0, & otherwise \end{cases}$$

$$\boxed{x_o[n] = x[n]\, w_R[n]}$$

Building the DFT formula (cont)

Continuous time signal **x(t)**

sample

Discrete time signal **x(n)**

window

$$X_0(\Omega) = \sum_{n=-\infty}^{\infty} x_0[n]e^{-j\Omega n} = \sum_{n=0}^{N-1} x_0[n]e^{-j\Omega n}$$

Discrete time signal $\mathbf{x_0(n)}$

Finite length

DTFT

Discrete Time Fourier Transform (DTFT), $X_o(\Omega)$

(periodic over $[0,2\pi)$)

Building the DFT formula (cont)

Continuous time signal **x(t)**

Discrete Fourier Transform DFT **X(k)**

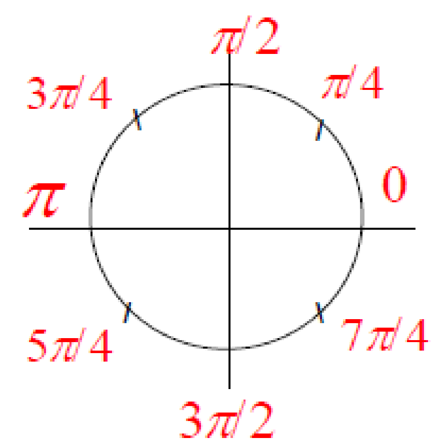
Discrete + periodic with period **N**

sample

Discrete time signal **x(n)**

DFT

Sample at N values around the unit circle



window

Discrete time signal **x_0(n)**

Finite length

DTFT

$X_0(\Omega)$

Continuous + periodic with period 2π

Notation conventions

$$W_N \equiv e^{-j\frac{2\pi}{N}}$$

"the Nth root of unity"

is often used to "simplify notation"

We see that $W_N^{\ N} \equiv e^{-j\frac{2\pi N}{N}} = 1$

Often just write this root as W, when the value of N is understood from the context of the discussion.

Recall the orthogonality property

$$\sum_{k=0}^{N-1} W_N^{k(p-n)} = \begin{cases} N, & p = n \\ 0, & p \neq n \end{cases}$$

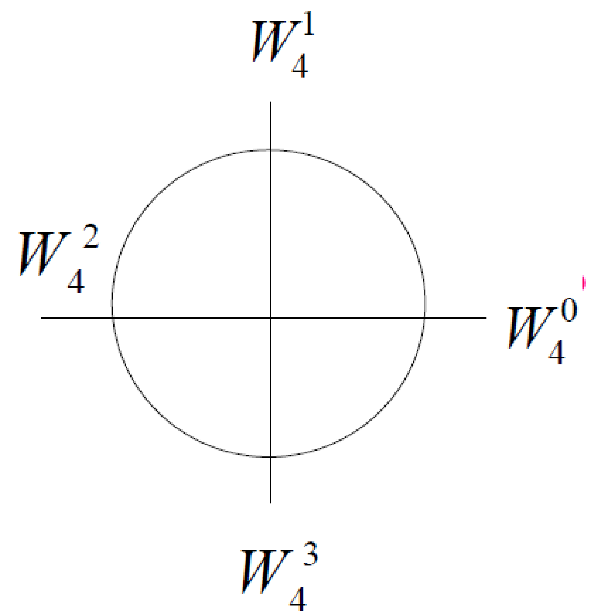
Notation conventions (cont)

$$W_N \equiv e^{-j\frac{2\pi}{N}} \qquad W_N^{\ N} = e^{-j\frac{2\pi N}{N}} = 1$$

To see that $$\sum_{k=0}^{N-1} W_N^{k(p-n)} = \begin{cases} N, & p = n \\ 0, & p \neq n \end{cases}$$

recall that

$$\sum_{i=0}^{N-1} a^i = \begin{cases} 0 & if \quad a = 0 \\ N & if \quad a = 1 \\ \dfrac{1-a^N}{1-a} & otherwise \end{cases}$$



DFT and inverse DFT formulas

Notation $$W_N = e^{-j\frac{2\pi}{N}}$$

$$X[k] = \sum_{n=0}^{N-1} x[n]\, W_N^{kn}$$

k = 0,1,...,N-1

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k]\, W_N^{-kn}$$

n = 0,1,...,N-1

Note that the DFT is a sequence of N numbers (in the frequency domain), just like $x_o[n]$ is a sequence of N numbers in the time domain

You only have to store N points

Lecture #20

DFT and inverse DFT

1. DFT and inverse DFT formulas

2. DFT and inverse DFT examples

3. Frequency resolution of the DFT

DFT and IDFT examples

Ex.1. Find the DFT of x(n) = 1, n = 0, 1, 2, ..., (N-1)

$$X[k]=\sum_{n=0}^{N-1}x[n]W_N^{kn}=\sum_{n=0}^{N-1}W^{kn}=\frac{1-W^{kN}}{1-W^{k}}$$

k = 0,1,...,N-1

k = 0 → X(k) = X(0) = N

k ≠ 0 → X(k) = 0

$$\Rightarrow \boxed{X[k]=N\delta[k]}$$

DFT and IDFT examples

Ex.2. Given $y(n) = \delta(n-2)$ and $N = 8$, find $Y(k)$

DFT and IDFT examples

Ex.3. Find the IDFT of X(k) = 1, k = 0, 1, ..., 7.

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k]\, W_N^{-kn}$$

$$\boxed{x[n] = \frac{1}{8} \sum_{k=0}^{7} W_8^{-kn} = \tfrac{1}{8} N\delta[n] = \delta[n]}$$

DFT and IDFT examples

Ex.4. Given $x(n) = \delta(n) + 2\delta(n-1) + 3\delta(n-2) + \delta(n-3)$ and $N = 4$. Find $X(k)$.

DFT and IDFT examples

Ex.5. Given $X(k) = 2\delta(k) + 2\delta(k-2)$ and $N = 4$. Find $x(n)$.

DFT in matrix forms

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

Let's define :

$$x_N = \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix} \qquad X_N = \begin{bmatrix} X[0] \\ X[1] \\ \vdots \\ X[N-1] \end{bmatrix}$$

$$W_N = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_N^1 & W_N^2 & W_N^{(N-1)} \\ \vdots & & & \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \ldots W_N^{(N-1)(N-1)} \end{bmatrix}$$

The N-point DFT may be expressed in matrix form as:

$$\boxed{\mathbf{X_N = W_N x_N}}$$

IDFT in matrix forms

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

$$Let's\ define:$$

$$x_N=\begin{bmatrix}x[0]\\x[1]\\\vdots\\x[N-1]\end{bmatrix}\qquad X_N=\begin{bmatrix}X[0]\\X[1]\\\vdots\\X[N-1]\end{bmatrix}$$

$$W_N=\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & W_N^1 & W_N^2 & W_N^{(N-1)}\\ \vdots & & & \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \ldots W_N^{(N-1)(N-1)}\end{bmatrix}$$

The N-point DFT may be expressed in matrix form as:

$$x_N=W_N^{-1}X_N=\frac{1}{N}W_N^*X_N$$

$$\Rightarrow W_NW_N^*=NI_N$$

$$I_N:identity\ \ matrix$$

Example of calculation of DFT in matrix form

Find DFT of the signal

x[n] =

δ[n-1] + 2δ[n-2] + 3δ[n-3]

$$Let's\ define:$$

$$x_4 = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix}$$

$$W_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^1 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^4 & W_4^6 \\ 1 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$Then\ X_4 = W_4 . x_4 = \begin{bmatrix} 6 \\ -2+2j \\ -2 \\ -2-2j \end{bmatrix}$$

Lecture #20

DFT and inverse DFT

1. DFT and inverse DFT formulas
2. DFT and inverse DFT examples
3. **Frequency resolution of the DFT**

Frequency resolution of the DFT

Discrete frequency spectrum computed from DFT has the spacing between frequency samples of:

$$\boxed{\Delta f = \frac{f_s}{N}} \qquad \boxed{\Delta\Omega = \frac{2\pi}{N}}$$

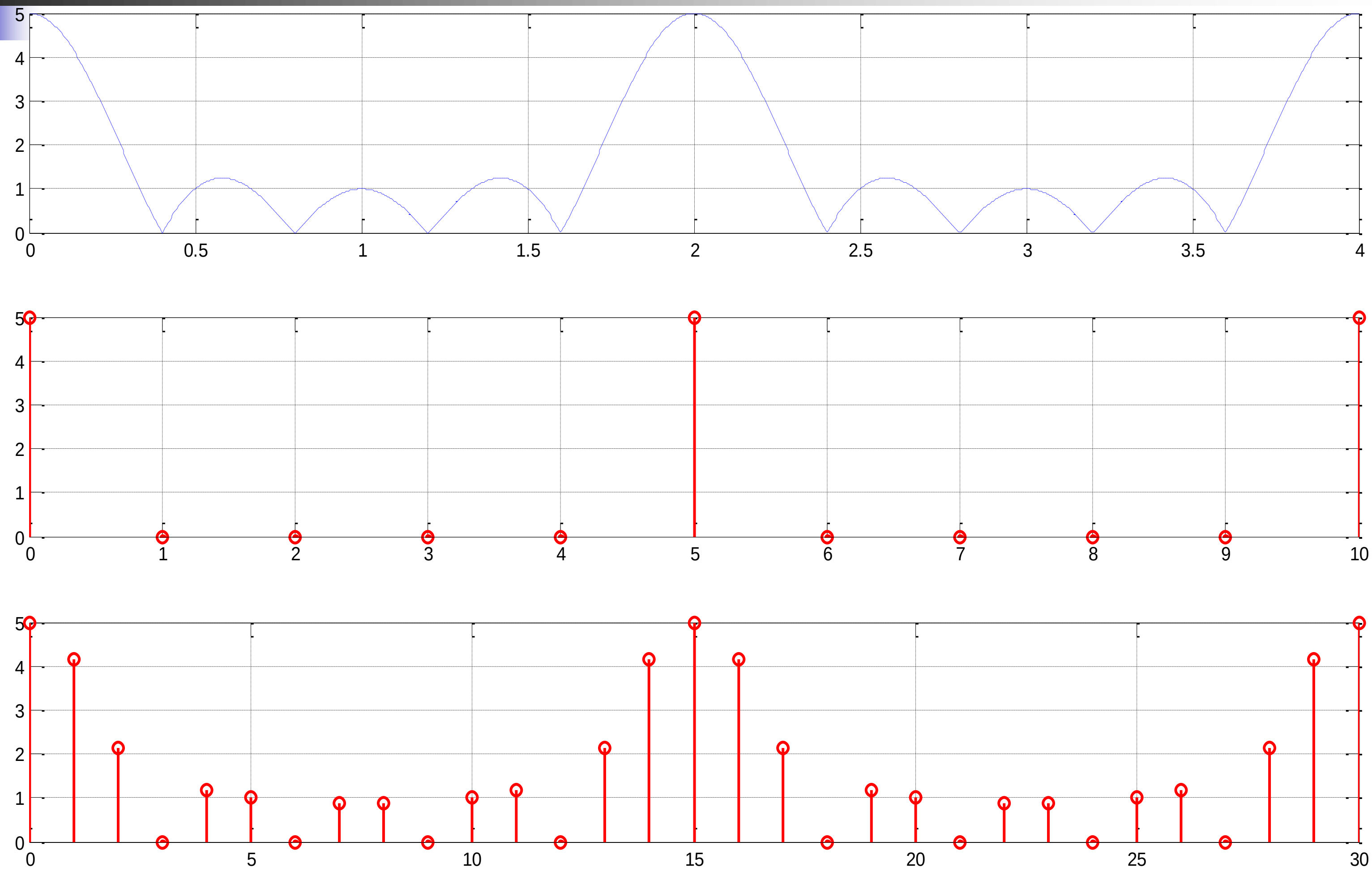
→ The choice of N determines the resolution of the frequency spectrum, or vice-versa

→ To obtain the adequate resolution:

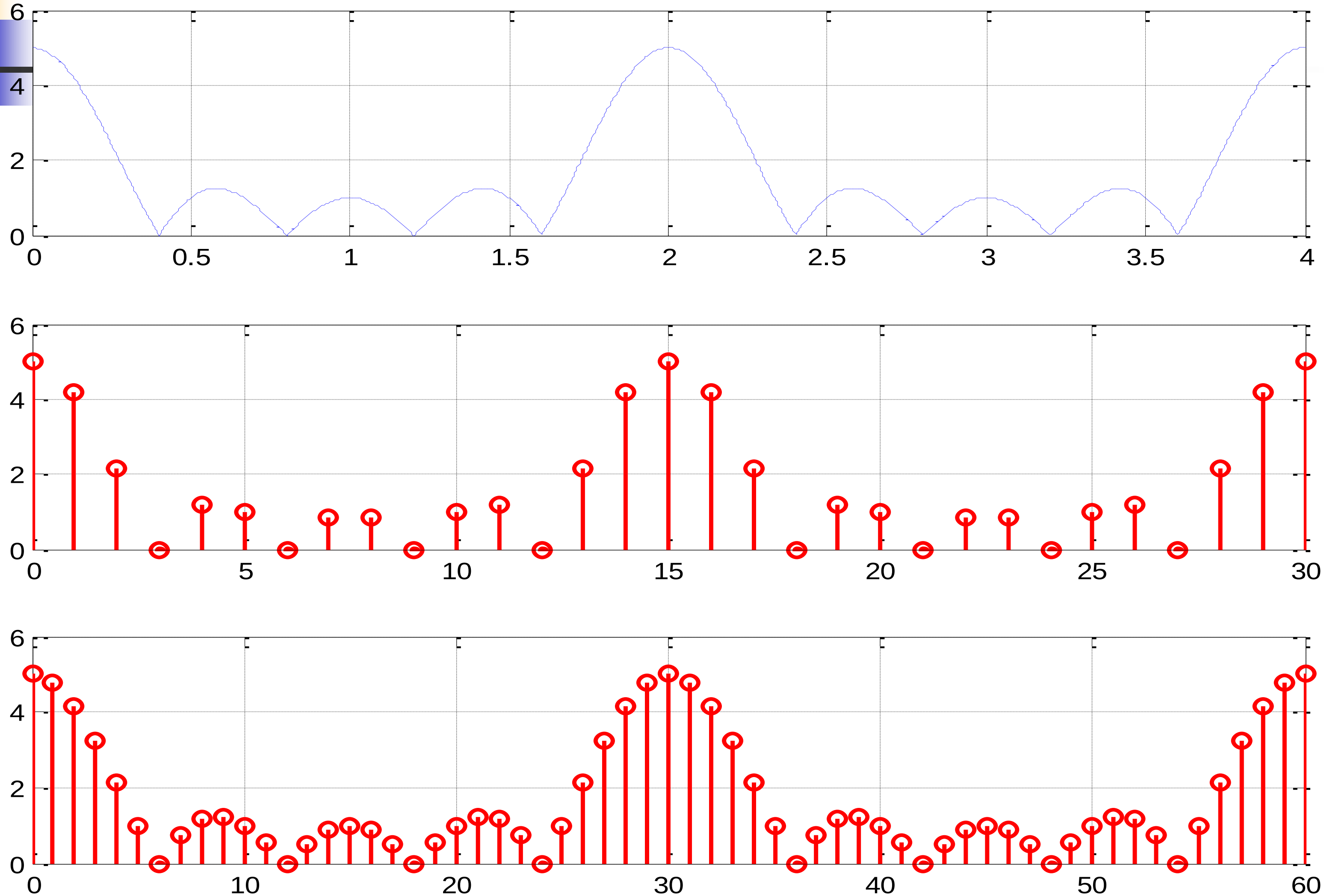
1. Increasing of the duration of data input to the DFT

2. *Zero padding*, which adds no new information, but effects a better interpolator

Examples of N = 5 and N = 15



Examples of N = 15 and N = 30

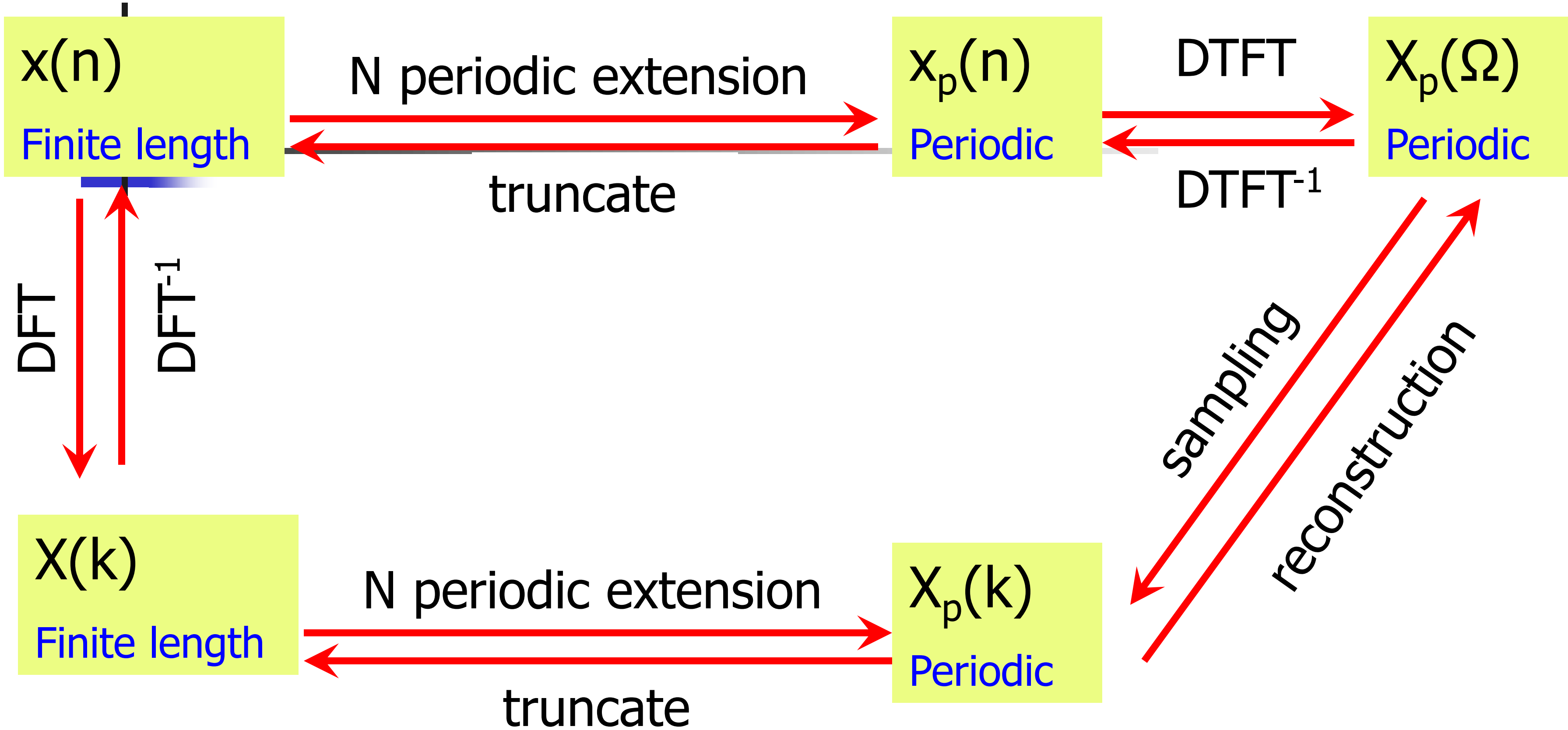


Lecture #21

DFT properties

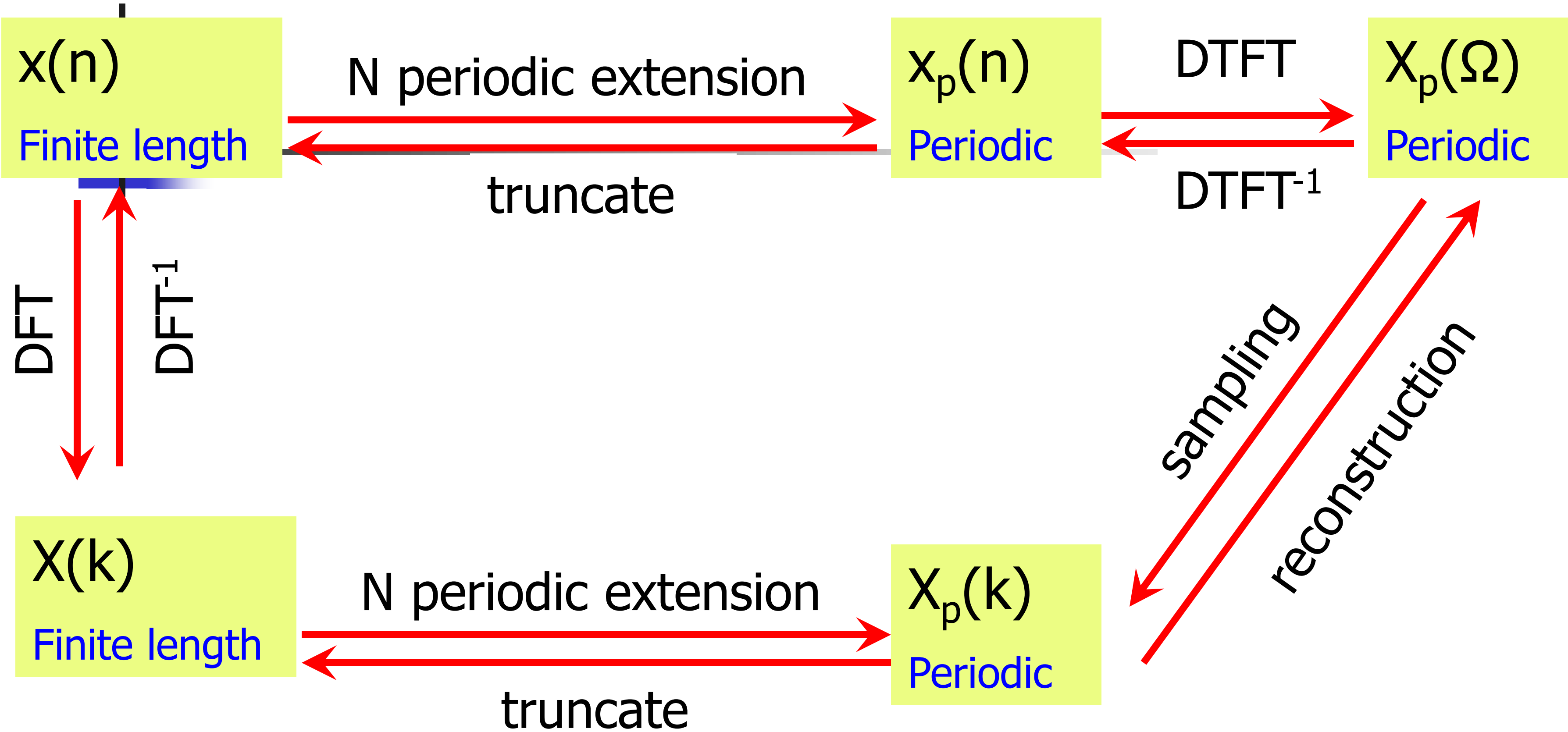
1. **Periodicity and Linearity**
2. Circular time shift
3. Circular frequency shift
4. Circular convolution
5. Multiplication
6. Parseval’s theorem

DFT properties



Most of the properties of the DFT are similar to other Fourier Transforms, but there are some key differences

DFT properties



All the differences are due to the fact that the DFT behaves like the underlying sequence is actually periodic with period N

Periodicity

If x[n] and X[k] are an N-point DFT pair, then

$x[n+N] = x[n]$ for all n

$X[k+N] = X[k]$ for all k

Linearity

$$a_1 x_1[n] + a_2 x_2[n] \overset{DFT}{\leftrightarrow} a_1 X_1[k] + a_2 X_2[k]$$

Note: The length of $x_1[n]$ is same with the length of $x_2[n]$

Proof:

Infer from the definition formula of DFT

Lecture #21

DFT properties

1. Periodicity and Linearity
2. **Circular time shift**
3. Circular frequency shift
4. Circular convolution
5. Multiplication
6. Parseval's theorem

Circular time shift property

$$\boxed{\mathrm{x}[\mathrm{n}-\mathrm{m}] \overset{\mathrm{DFT}}{\leftrightarrow} \mathrm{W}^{\mathrm{km}}\mathrm{X}[\mathrm{k}]}$$

Proof:

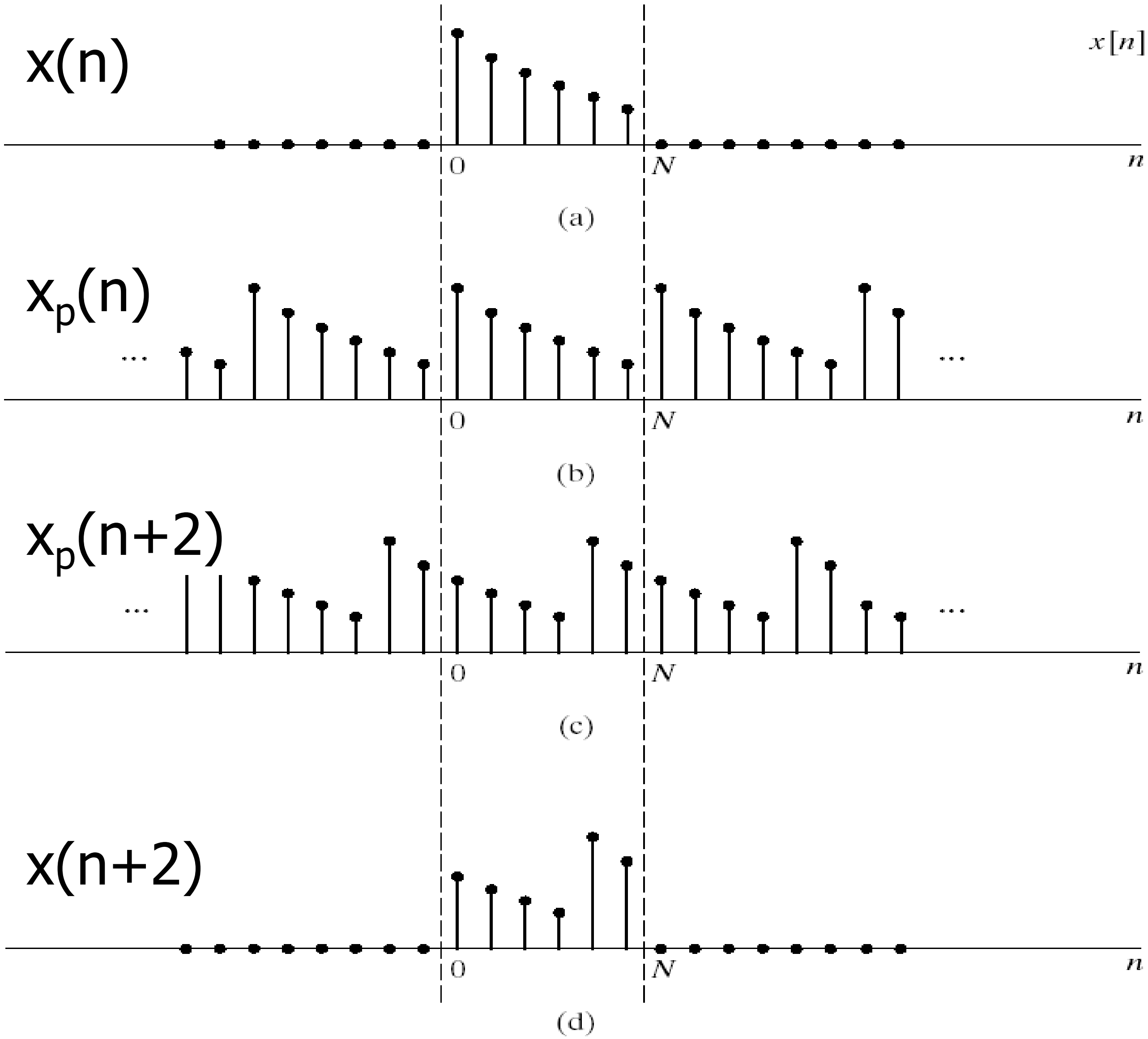
Infer from the relation between DFT and DTFT

$$x[n-m] \overset{DTFT}{\leftrightarrow} e^{-j\Omega m} X(\Omega)$$

$$\Omega \to k\frac{2\pi}{N}:$$

$$e^{-j\Omega m} X(\Omega) \to e^{-jk\frac{2\pi}{N}m} X[k] = W^{km} X[k]$$

Circular time shift



Circular shift by ***m*** is the same as a shift by ***m modulo N***

Lecture #21

DFT properties

1. Periodicity and Linearity
2. Circular time shift
3. **Circular frequency shift**
4. Circular convolution
5. Multiplication
6. Parseval's theorem

Circular frequency shift property

$$\boxed{x[n]W^{-\ln} \overset{DFT}{\longleftrightarrow} X[k-l]}$$

Proof:

Similar to the circular time shift property

Lecture #21

DFT properties

1. Periodicity and Linearity
2. Circular time shift
3. Circular frequency shift
4. **Circular convolution**
5. Multiplication
6. Parseval's theorem

Circular convolution property

$$X_1[k].X_2[k] \overset{DFT}{\longleftrightarrow} x_1[n] \otimes x_2[n] = \sum_{p=0}^{N-1} x_1[p] x_2[n-p]$$

- The non-zero length of $x_1(n)$ and $x_2(n)$ can be no longer than N
- The shift operation is circular shift
- The flip operation is circular flip

The circular convolution is not the linear convolution in chapter 2

Direct method to calculate circular convolution

1. Draw a circle with *N* values of *x(n)* with *N* equally spaced angles in a counterclockwise direction.

2. Draw a smaller radius circle with *N* values of *h(n)* with equally spaced angles in a clockwise direction. Superimpose the centers of 2 circles, and have *h(0)* in front of *x(0).*

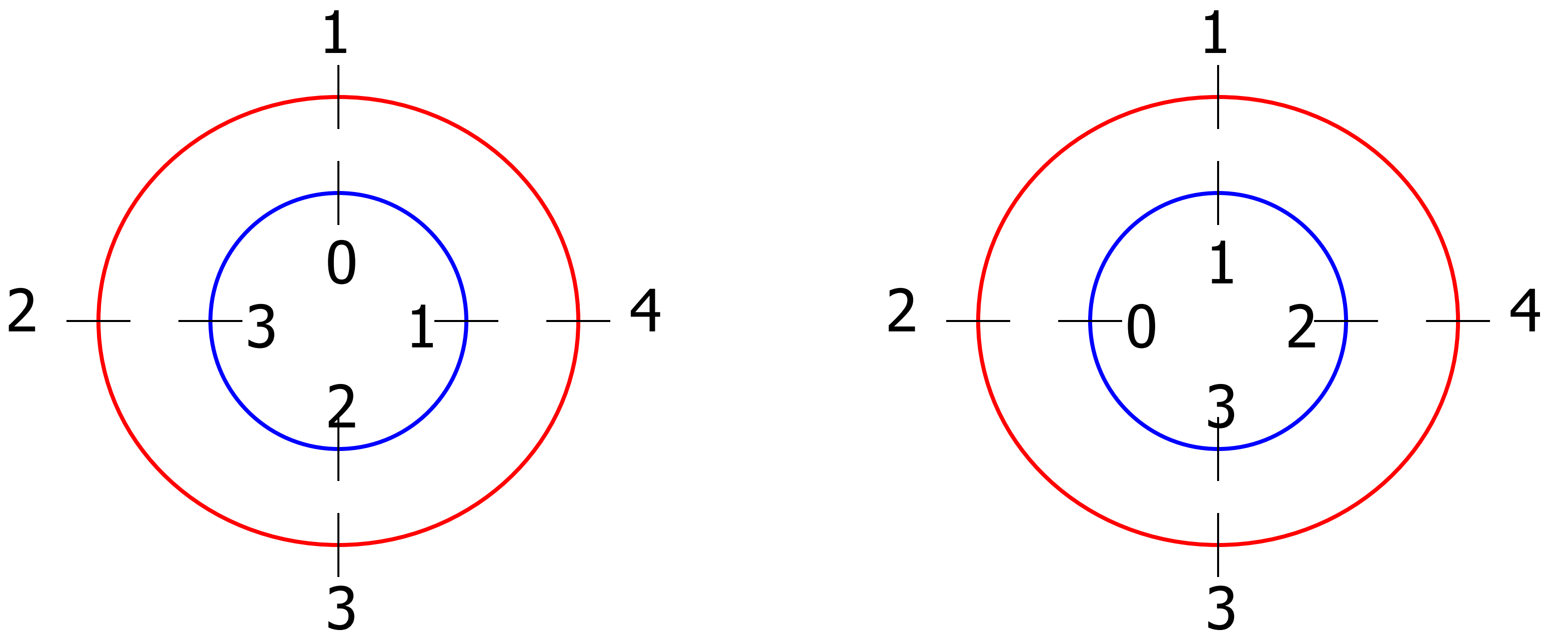
3. Calculate *y(0)* by multiplying the corresponding values on each radial line, and then adding the products.

4. Find succeeding values of *y(n)* in the same way after rotating the inner disk counterclockwise through the angle $2\pi k/N$

Example to calculate circular convolution

- Evaluate the circular convolution, *y(n)* of 2 signals:

$x_1(n)$ = [1 2 3 4]; $x_2(n)$ = [0 1 2 3]



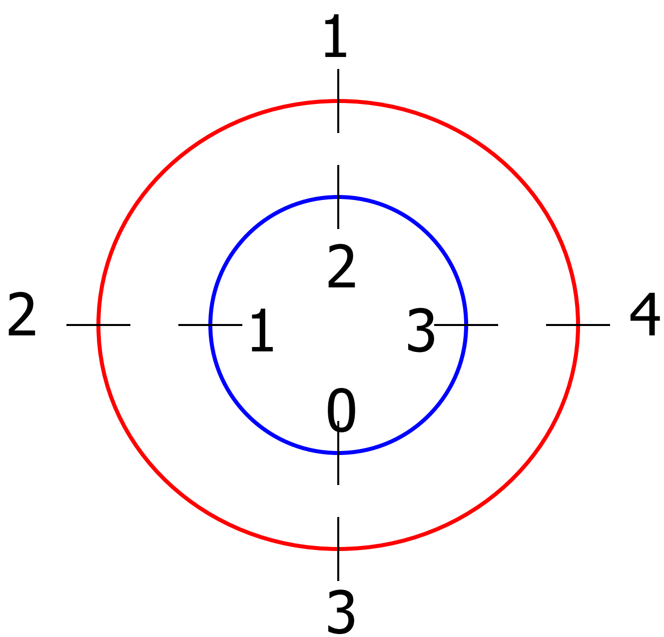
y(0) = 16

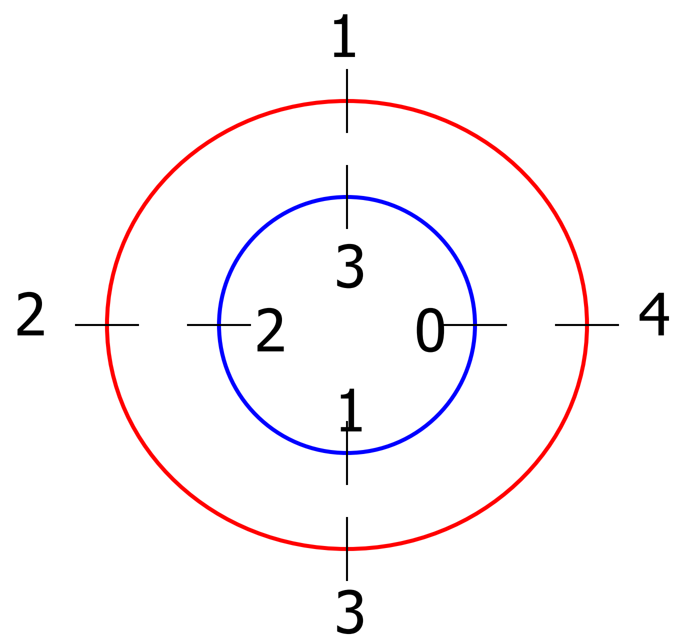
y(1) = 18

Example (cont)

- Evaluate the circular convolution, *y(n)* of 2 signals:

  $x_1(n) = [\ 1\ 2\ 3\ 4\ ]$; $x_2(n) = [\ 0\ 1\ 2\ 3\ ]$

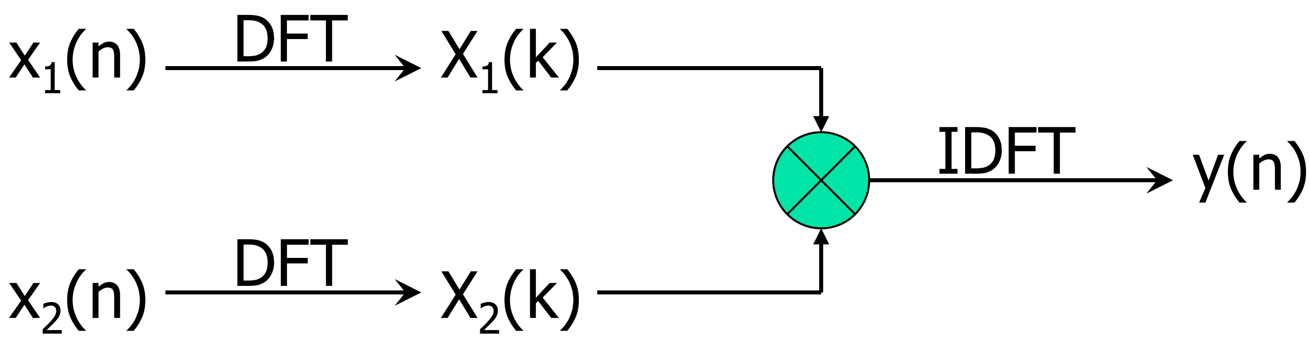




y(2) = 16

y(3) = 10

Another method to calculate circular convolution



Ex. $x_1(n) = [\ 1\ 2\ 3\ 4\ ]$; $x_2(n) = [\ 0\ 1\ 2\ 3\ ]$

$X_1(k) = [\ 10, -2+j2, -2, -2-j2\ ];$

$X_2(k) = [\ 6, -2+j2, -2, -2-j2\ ];$

$Y(k) = X_1(k).X_2(k) = [\ 60, -j8, 4, j8\ ]$

$y(n) = [\ 16, 18, 16, 10\ ]$

Lecture #21

DFT properties

1. Periodicity and Linearity
2. Circular time shift
3. Circular frequency shift
4. Circular convolution
5. **Multiplication**
6. Parseval's theorem

Multiplication

$$x_1[n].x_2[n] \overset{DFT}{\longleftrightarrow} \frac{1}{N} X_1[k] \otimes X_2[k] = \frac{1}{N} \sum_{l=0}^{N-1} X_1[l] X_2[k-l]$$

- The non-zero length of $X_1(k)$ and $X_2(k)$ can be no longer than N
- The shift operation is circular shift
- The flip operation is circular flip

Lecture #21

DFT properties

1. Periodicity and Linearity
2. Circular time shift
3. Circular frequency shift
4. Circular convolution
5. Multiplication
6. **Parseval's theorem**

Parseval's theorem

- The general form:

$$\boxed{\sum_{n=0}^{N-1} x[n]y^*[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]Y^*[k]}$$

- Special case:

$$\boxed{\sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N}\sum_{k=0}^{N-1} |X[k]|^2}$$

Parseval's theorem example

- Given x(n) = δ(n) + 2δ(n-1) +3δ(n-2) + δ(n-3) and N = 4.

$$X[0] = 7$$
$$X[1] = 1 - 2j - 3 + j = -2 - j$$
$$X[2] = 1 - 2 + 3 - 1 = 1$$
$$X[3] = 1 + 2j - 3 - j = -2 + j$$

$$\sum_{n=0}^{N-1} |x[n]|^2 = 1^2 + 2^2 + 3^2 + 1^2 = 15$$

$$\frac{1}{N}\sum_{k=0}^{N-1} |X[k]|^2 = \frac{1}{4}(7^2 + 2^2 + 1^2 + 1^2 + 2^2 + 1^2) = \frac{60}{4} = 15$$

Lecture #22

Fast Fourier Transform (FFT)

1. What is FFT?

2. The decomposition-in-time Fast Fourier Transform algorithm (DIT-FFT)

3. The decomposition-in-frequency Fast Fourier Transform algorithm (DIF-FFT)

Recall DFT and IDFT definition

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk} \qquad 0 \le k \le N-1$$

$$x[n] = \frac{1}{N} \sum_{n=0}^{N-1} X[k] W_N^{-nk} \qquad 0 \le n \le N-1$$

N^{th} complex root of unity:

$$W_N = e^{-j\frac{2\pi}{N}}$$

Using some "tricks" and choosing the appropriate N value, the DFT can be implemented in a very computationally efficient way – this is the Fast Fourier Transform (FFT) - which is the basis of lots of contemporary DSP hardware

Fast Fourier Transform (FFT)

- What does it take to compute DFT or IDFT?

  N^2 complex multiplications for all N points of $X(k)$ or $x[n]$

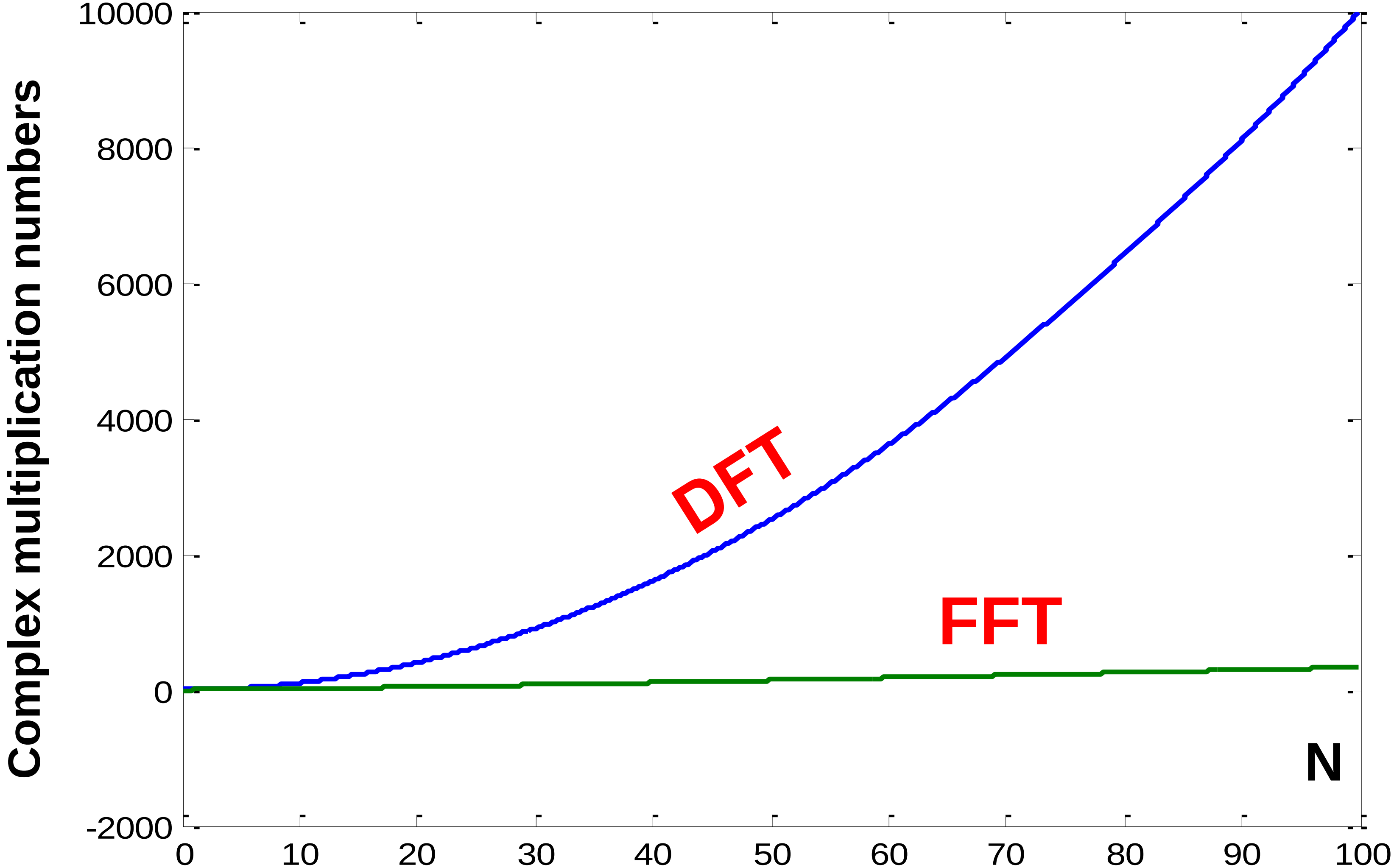
- **FFT:** the algorithm to optimize the computational process by breaking the N-point DFT into smaller DFTs

- **Ex.:** N is a power of 2: *split N-point DFT into two N/2-point DFTs, then split each of these into two of length N/4, etc., until we have N/2 subsequences of length 2*

- Cooley and Tukey (1965):

$$\boxed{N^2 \Rightarrow \frac{N}{2}\log_2 N \ \text{complex multiplications}}$$

Comparison of DFT and FFT efficiency



Comparison of DFT and FFT efficiency

Number of points N	Complex multiplications in direct DFT	Complex multiplications in FFT	Speed improvement factor
4	16	4	4.0
8	64	12	5.3
16	256	32	8.0
32	1,024	80	12.8
64	4,096	192	21.3
128	16,384	448	36.6
256	65,536	1,024	64.0
512	262,144	2,304	113.8
1024	1,048,576	5,120	204.8

FFT (cont)

- Some properties of $\left\{W_N^{nk}\right\}$ can be exploited in performing FFT:

$$W_N^{k(N-n)} = W_N^{-kn} = \left(W_N^{kn}\right)^*, \text{complex conjugate symmetry}$$

$$W_N^{kn} = W_N^{k(n+N)} = W_N^{(k+N)n}, \text{periodicity in n and k}$$

- Other useful properties:

$$W_N^{(k+\frac{N}{2})n} = W_N^{kn} W_N^{\frac{nN}{2}} = W_N^{kn} e^{-jn\pi} = \begin{cases} W_N^{kn}, & \text{if n even} \\ -W_N^{kn}, & \text{if n odd} \end{cases}$$

$$W_N^{2kn} = W_{\frac{N}{2}}^{kn}$$

Lecture #22

Fast Fourier Transform (FFT)

1. What is FFT?

2. The decomposition-in-time Fast Fourier Transform algorithm (DIT-FFT)

3. The decomposition-in-frequency Fast Fourier Transform algorithm (DIF-FFT)

DIT-FFT with N as a 2-radix number

- We divide X(k) into 2 parts:

$$X[k] = \sum_{n\,\mathrm{even}} x[n]W^{kn} + \sum_{n\,\mathrm{odd}} x[n]W^{kn}$$

$$\boxed{\mathrm{X(k)} = \mathrm{G(k)} + \mathrm{W}_{\mathrm{N}}^{\mathrm{k}}\mathrm{H(k)}, \quad \mathrm{k} = 0,1,\ldots,\mathrm{N}-1}$$

- *G(k)* is N/2 points DFT of the even numbered data: x(0), x(2), x(4), …., x(N-2).
- *H(k)* is the N/2 points DFT of the odd numbered data: x(1), x(3), …, x(N-1).

DIT-FFT (cont)

$$X[k] = \sum_{m=0}^{\frac{N}{2}-1} x[2m]W^{2mk} + \sum_{m=0}^{\frac{N}{2}-1} x[2m+1]W^{k(2m+1)}$$

$$= \sum_{m=0}^{\frac{N}{2}-1} x[2m](W^2)^{mk} + W^k \sum_{m=0}^{\frac{N}{2}-1} x[2m+1](W^2)^{mk}$$

Note: $\mathrm{W}_{\mathrm{N}}^{2} = \left(\mathrm{e}^{-\mathrm{j}2\pi/\mathrm{N}}\right)^{2} = \mathrm{e}^{-\mathrm{j}2\pi/(\mathrm{N}/2)} = \mathrm{W}_{\mathrm{N}/2}$

$$\boxed{X(k) = \sum_{m=0}^{\frac{N}{2}-1} g[m]W_{N/2}^{mk} + W_N^k \sum_{m=0}^{\frac{N}{2}-1} h[m]W_{N/2}^{mk} = G(k) + W_N^k H(k)}$$

G(k) and *H(k)* are of length N/2; *X(k)* is of length *N*

G(k)=G(k+N/2) and *H(k)=H(k+N/2)*

DIT-FFT of length N = 8

$$\boxed{X[k]_8 = G[k]_4 + W_8^k H[k]_4}$$

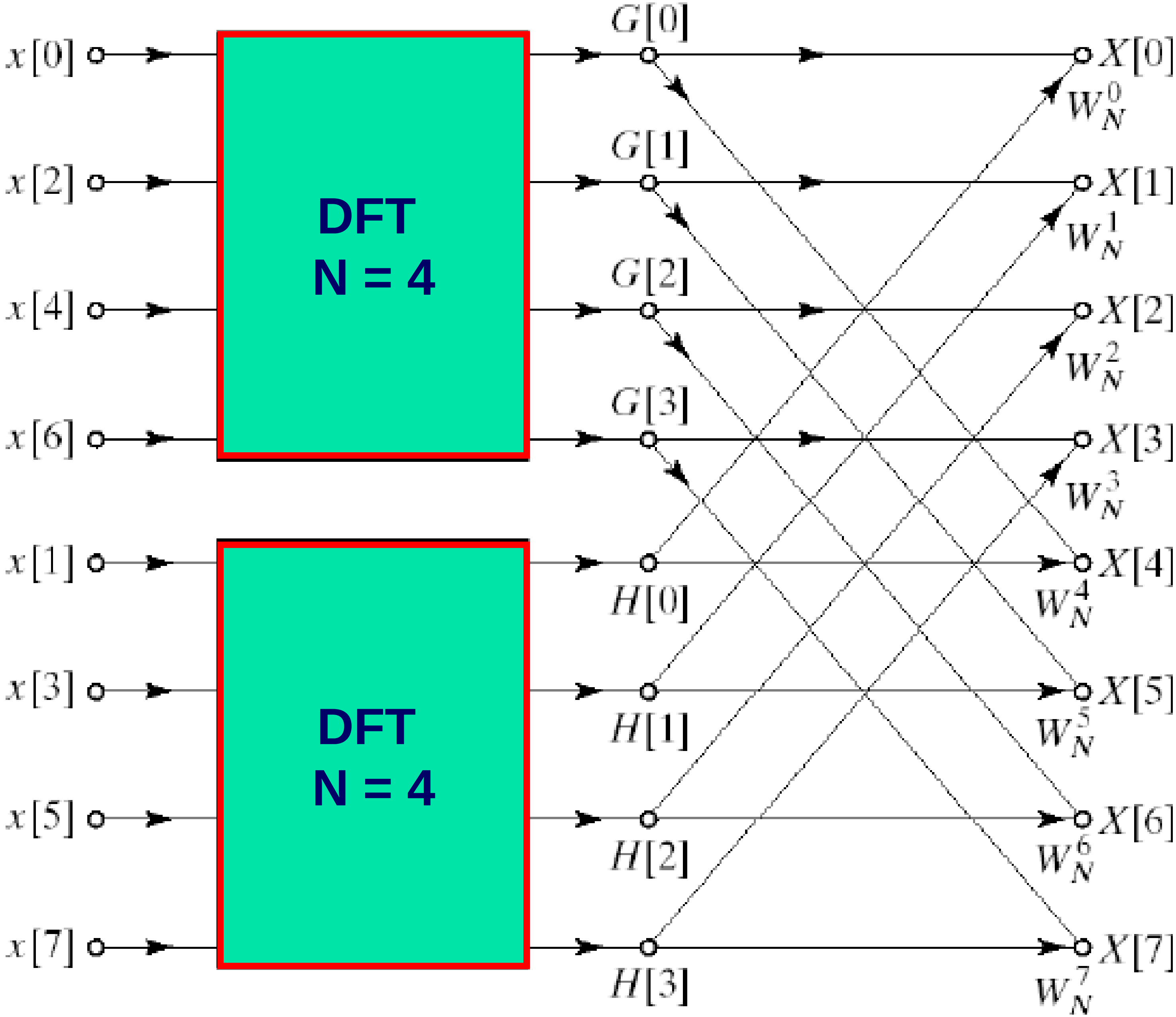
$$X[0] = G[0] + W_8^0 H[0] \qquad X[4] = G[0] + W_8^4 H[0]$$

$$X[1] = G[1] + W_8^1 H[1] \qquad X[5] = G[1] + W_8^5 H[1]$$

$$X[2] = G[2] + W_8^2 H[2] \qquad X[6] = G[2] + W_8^6 H[2]$$

$$X[3] = G[3] + W_8^3 H[3] \qquad X[7] = G[3] + W_8^7 H[3]$$

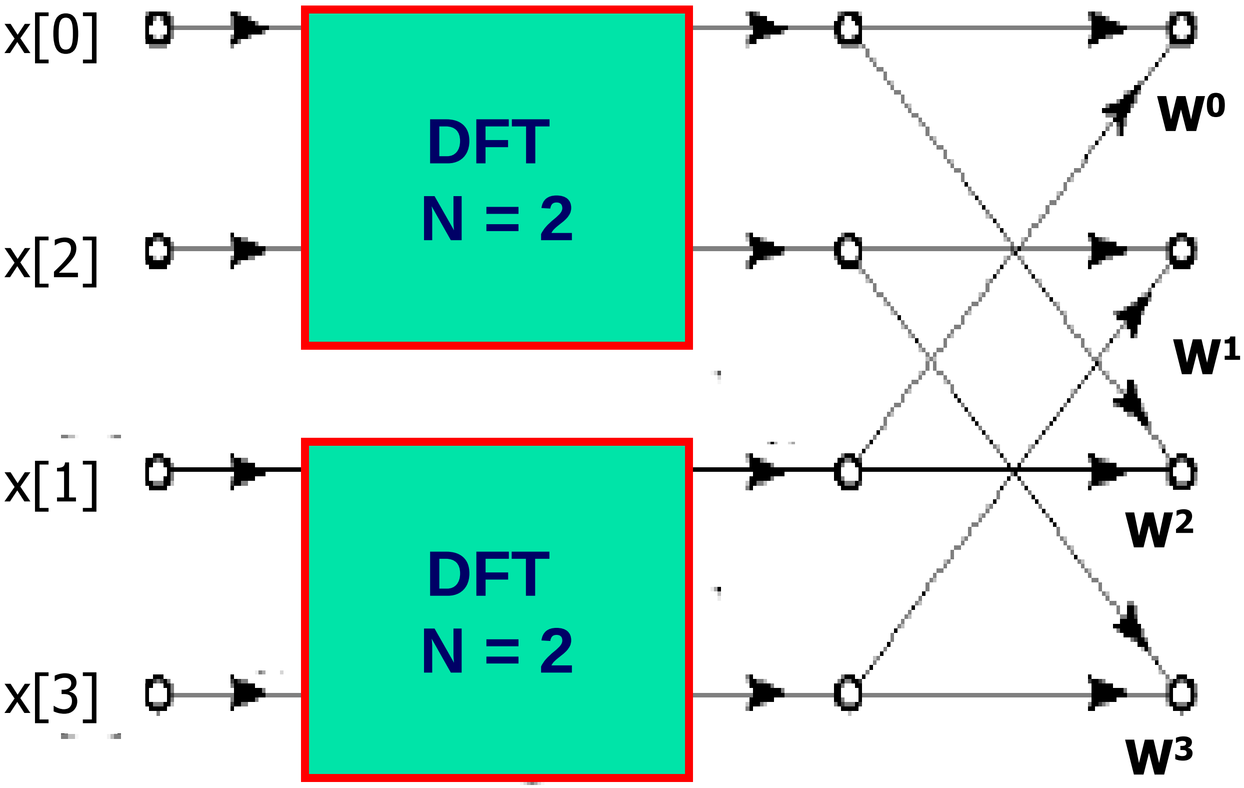
Signal flow graph for 8-point FFT



 # complex multiplications are reduced: $8^2 = 64 \rightarrow 2\text{x}(4)^2 + 8 = 40$

DIT-FFT of length N = 4

$$\boxed{X[k]_4 = G[k]_2 + W_4^k H[k]_2}$$



$$X[0] = G[0] + W_4^0 H[0]$$

$$X[1] = G[1] + W_4^1 H[1]$$

$$X[2] = G[0] + W_4^2 H[0]$$

$$X[3] = G[1] + W_4^3 H[1]$$

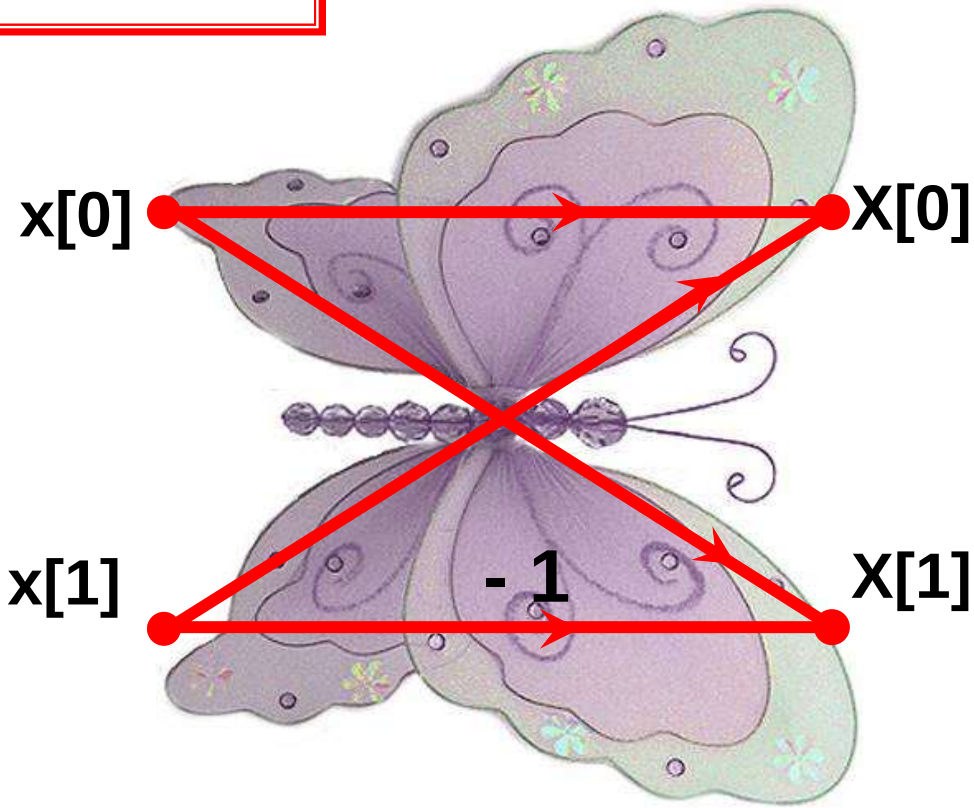
 # multiplications are reduced: $4^2 = 16 \rightarrow 2\text{x}(2)^2 + 4 = 12$

DIT-FFT of length N = 2

$$X[k]=\sum_{n=0}^{1}x[n]W^{nk},\quad 0\le k\le 1,\quad W=e^{-j\frac{2\pi}{2}}=-1$$

$$\Rightarrow \boxed{\begin{aligned} X[0]&=x[0]W^{0.0}+x[1]W^{1.0}=x[0]+x[1]\\ X[1]&=x[0]W^{0.1}+x[1]W^{1.1}=x[0]-x[1]\end{aligned}}$$

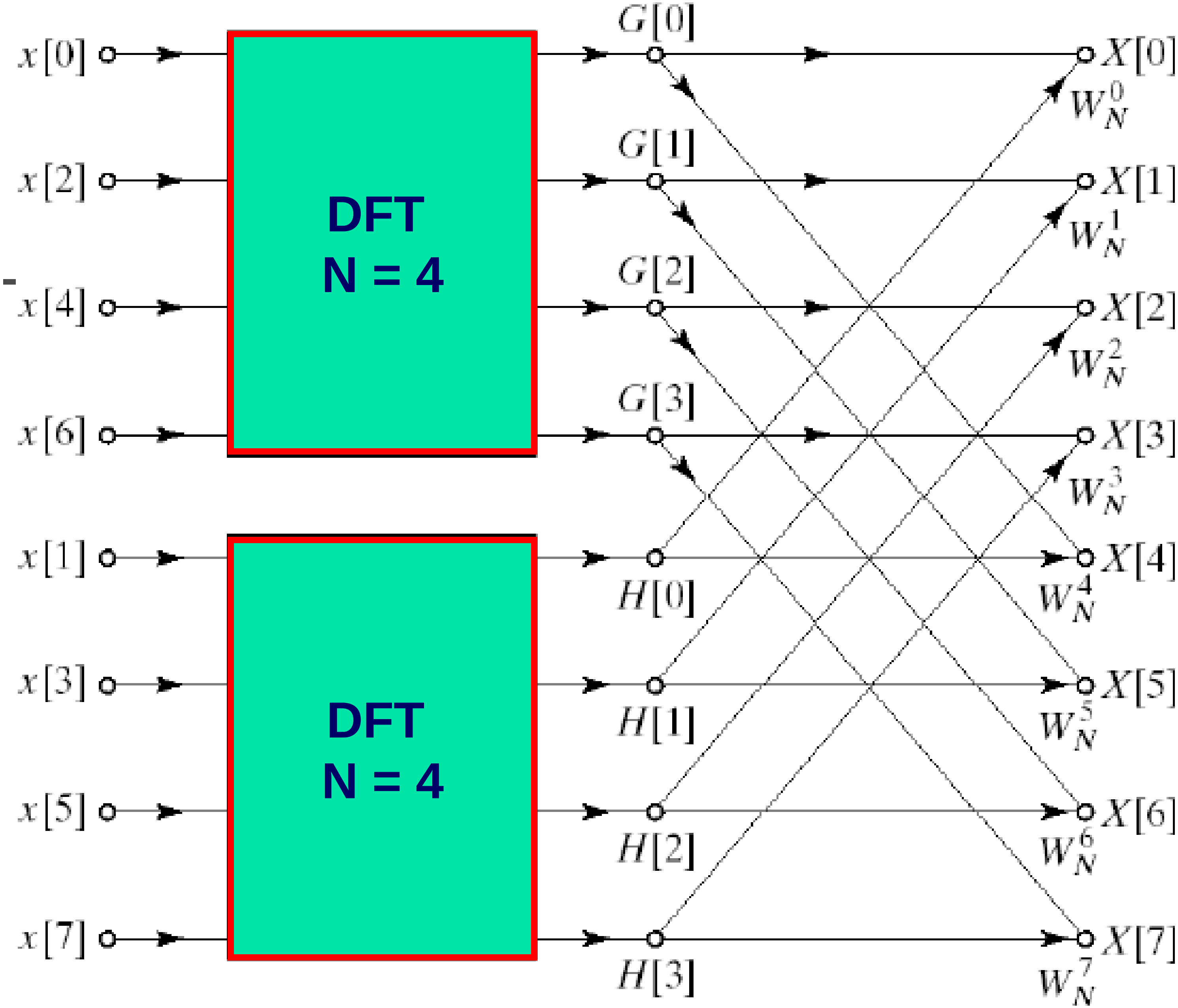
Butterfly diagram



 # multiplications are reduced: $2^2 = 4 \rightarrow 0$;

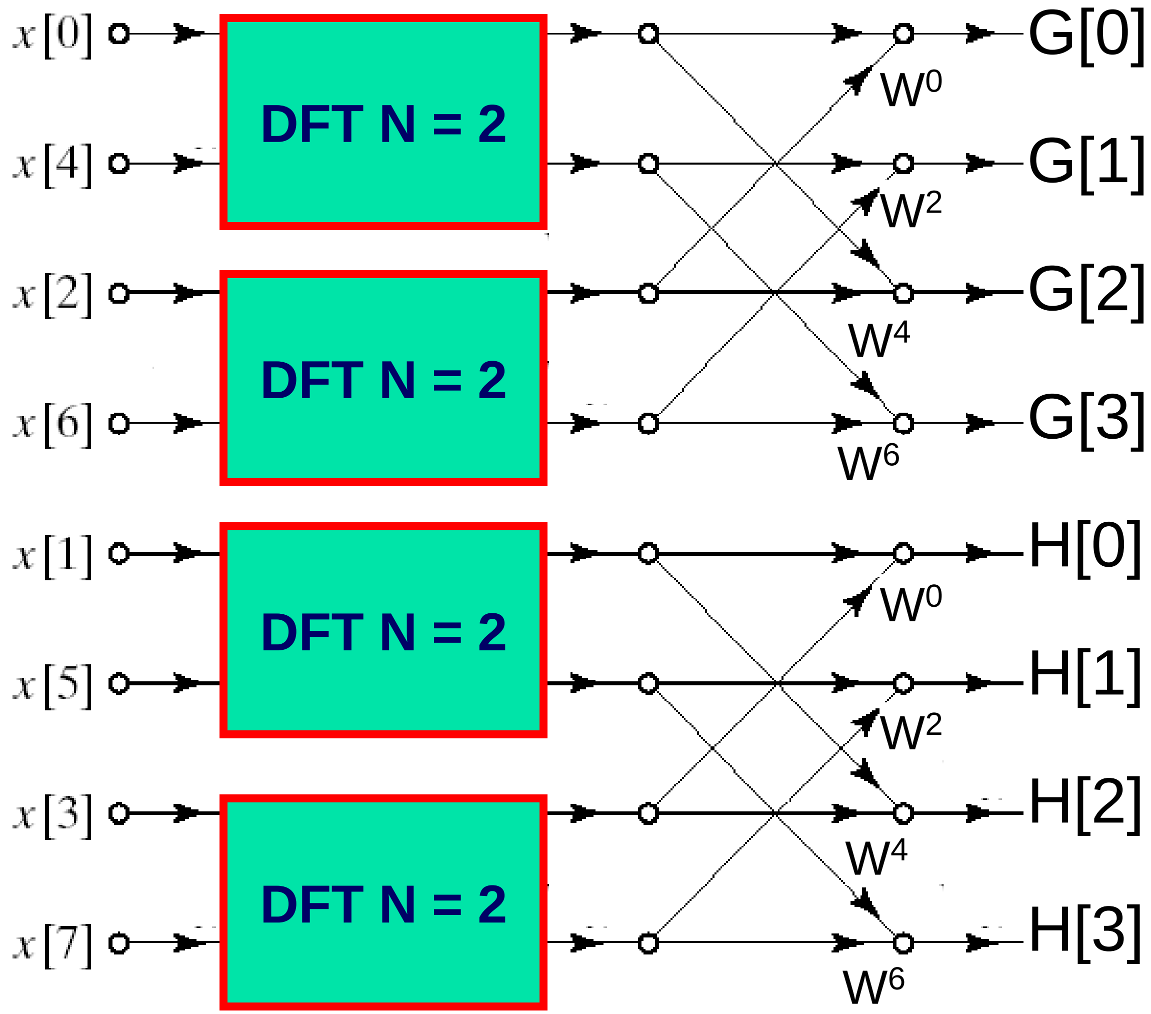
just 2 additions!!!

Example of 8-point FFT



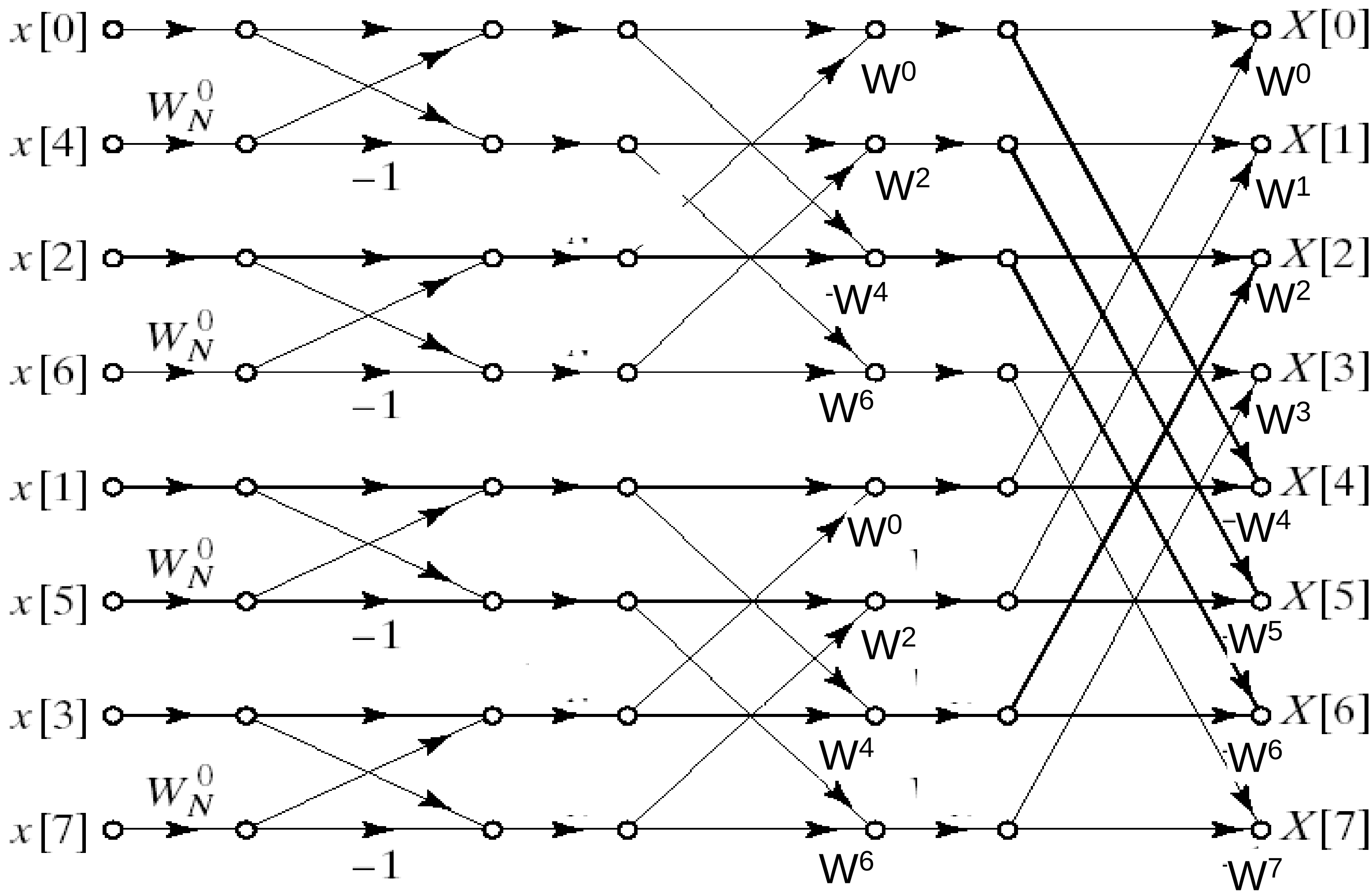
First step: split 8-point DFT into two 4-point DFTs

Example of 8-point FFT (cont)



Second step: split each of 4-point DFT into two 2-point DFTs

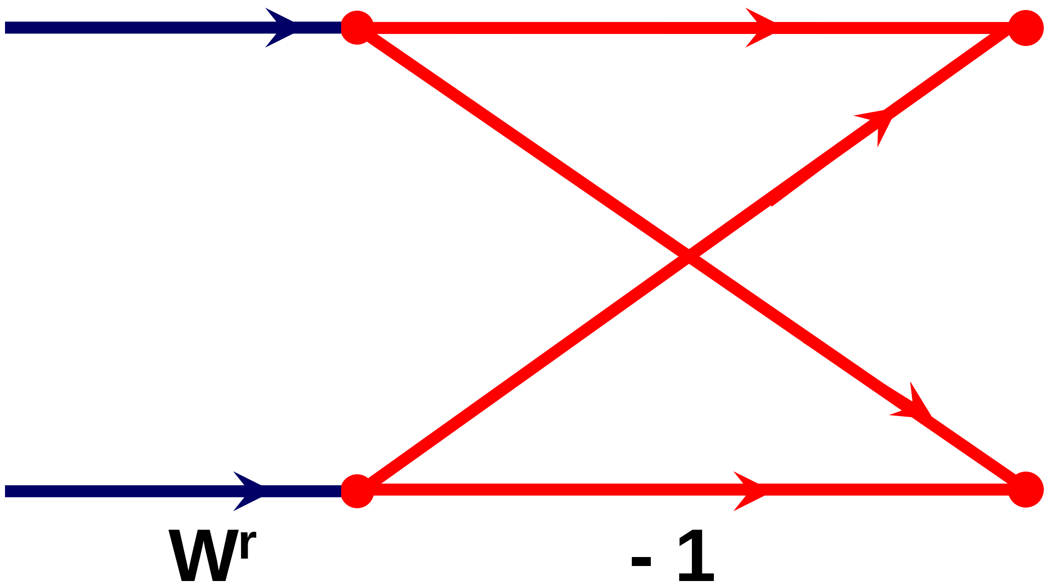
Example of 8-point FFT (cont)



Third step: combine all signal flow graphs of step 1 and step 2

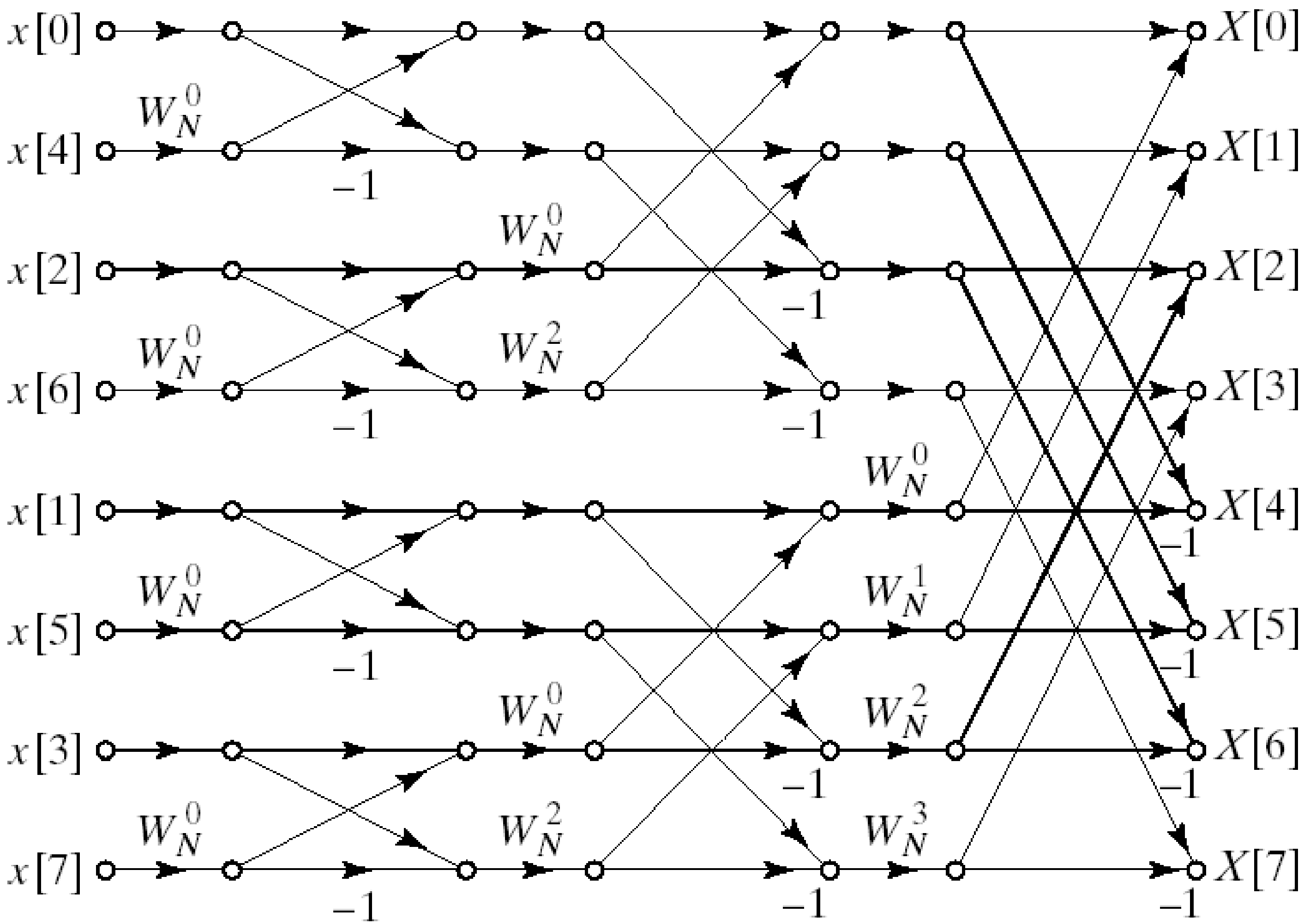
There are $3 = log_2 8$ stages; 4 butterfly diagrams in each stage

Butterfly diagram



$$\boxed{W^{r+N/2} = -W^r}$$

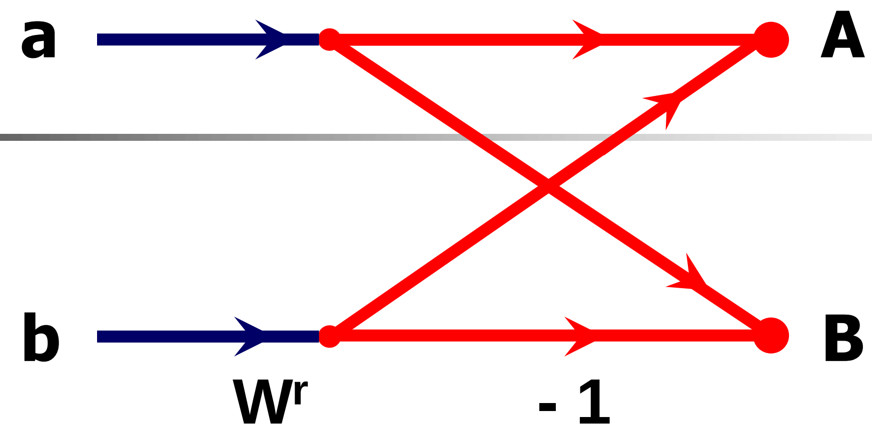
The overall signal flow graph for 8-point DIT-FFT



Now the overall computation is reduced to:

$$N^2 \Rightarrow \frac{N}{2}\log_2 N \text{ complex multiplications}$$

In-place computation



- Each butterfly takes *(a,b)* to produce *(A,B)* → no need to save *(a,b)* → can store *(A,B)* in the same locations as *(a,b)*
- We need *2N* store registers to store the results at each stage and these registers are also used throughout the computation of the N-point DFT
- The order of the input: bit-reversed order
- The order of the output: natural order

Lecture #22

Fast Fourier Transform (FFT)

1. What is FFT?

2. The decomposition-in-time Fast Fourier Transform algorithm (DIT-FFT)

3. The decomposition-in-frequency Fast Fourier Transform algorithm (DIF-FFT)

DIF-FFT with N as a 2-radix number

- We divide X(k) in half – first half and second half

$$X(k)=\sum_{n=0}^{(N/2)-1}x(n)W_N^{kn}+\sum_{n=(N/2)}^{N-1}x(n)W_N^{kn},\quad k=0,1,\ldots,N-1$$

- Rewrite the second half:

$$\sum_{n=(N/2)}^{N-1}x(n)W_N^{kn}=\sum_{n=0}^{(N/2)-1}x\left(n+\frac{N}{2}\right)W_N^{k\left(n+\frac{N}{2}\right)}=W_N^{k\frac{N}{2}}\sum_{n=0}^{(N/2)-1}x\left(n+\frac{N}{2}\right)W_N^{kn}$$

- Since: $$W_N^{k\frac{N}{2}}=e^{-j(k2\pi/N)(N/2)}=e^{-j\pi}=(-1)^k$$

then

$$\boxed{X(k)=\sum_{n=0}^{(N/2)-1}\left[x(n)+(-1)^k x(n+N/2)\right]W_N^{kn},\quad k=0,1,\ldots,N-1}$$

DIF-FFT (cont)

- Decompose *X(k)* into two frequency sequences, even and odd:

-
$$X(2k) = \sum_{n=0}^{(N/2)-1} \left[x(n) + x(n+N/2)\right] W_N^{2kn}, k = 0,1,\ldots,\frac{N}{2}-1$$

$$X(2k+1) = \sum_{n=0}^{(N/2)-1} \left[x(n) - x(n+N/2)\right] W_N^{(2k+1)n}, k = 0,1,\ldots,\frac{N}{2}-1$$

- Since: $W_N^2 = W_{N/2}^1$

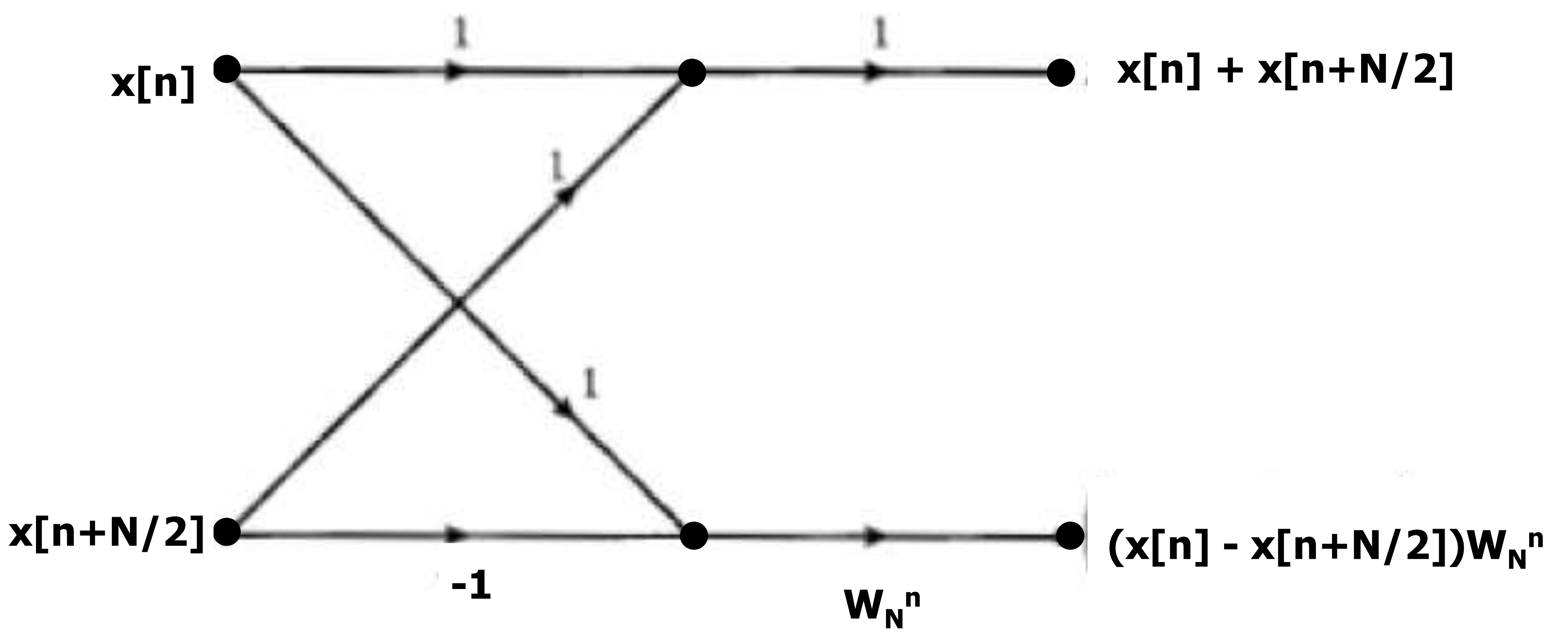
then,

$$\boxed{\begin{aligned} X(2k) &= \sum_{n=0}^{(N/2)-1} \left[x(n) + x(n+N/2)\right] W_{N/2}^{kn}, k = 0,1,\ldots,\frac{N}{2}-1 \\ X(2k+1) &= \sum_{n=0}^{(N/2)-1} \left\{\left[x(n) - x(n+N/2)\right] W_N^{n}\right\} W_{N/2}^{kn}, k = 0,1,\ldots,\frac{N}{2}-1 \end{aligned}}$$

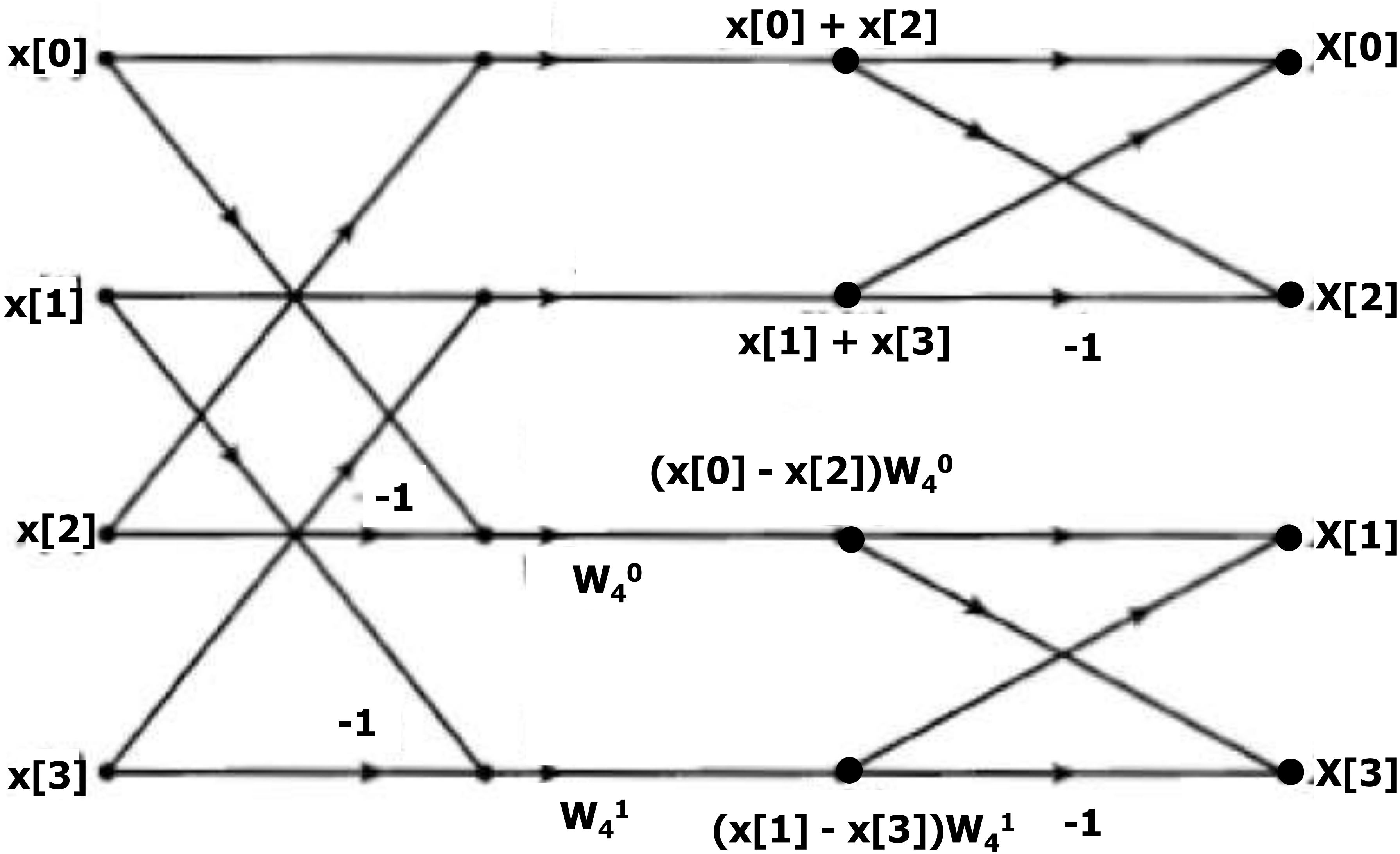
DIF-FFT butterfly

$$X[2k] = \frac{N}{2}\text{-point DFT of } [x[n] + x[n + N/2]];$$

$$X[2k+1] = \frac{N}{2}\text{-point DFT of } [x[n] - x[n + N/2]]W_N^n.$$



DIF-FFT of length N = 4



Lecture #23

DFT applications

1. Approximation of the Fourier Transform of analog signals

2. Linear convolution

Approximation of Fourier Transform

Using DFT as a discrete-frequency approximation of the DTFT and also an approximation of the CTFT

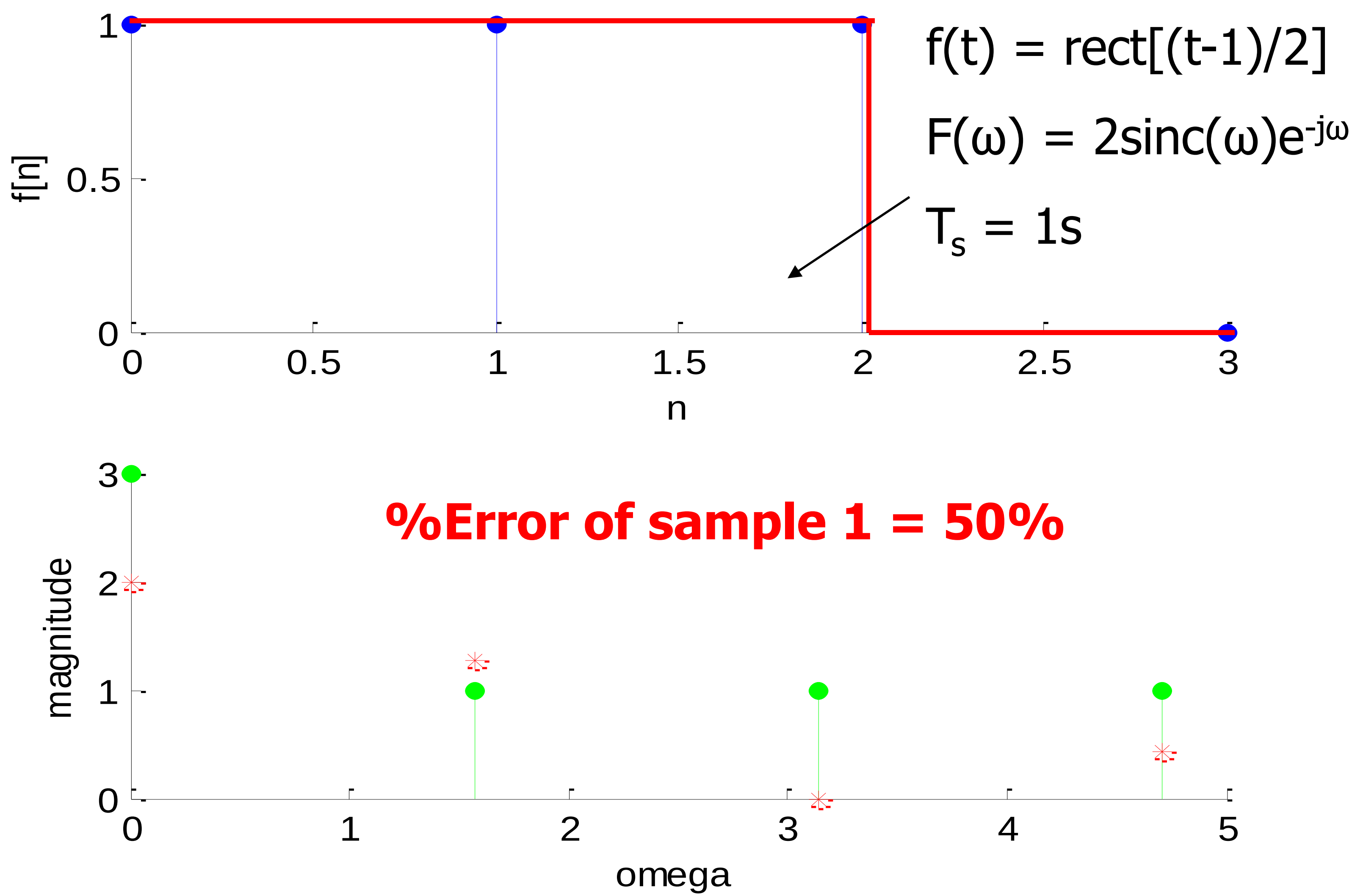
Step1. Determine the resolution $\Delta\Omega = 2\pi/N$ required for the DFT to be useful for its intended purpose → determine N (N often $= 2^n$)

Step 2. Determine the sampling frequency required to sample the analog signal so as to avoid aliasing $\omega_s \geq 2\omega_M$

Step 3. Accumulate N samples of the analog signal over $N.T$ seconds ($T = 2\pi/\omega_s$)

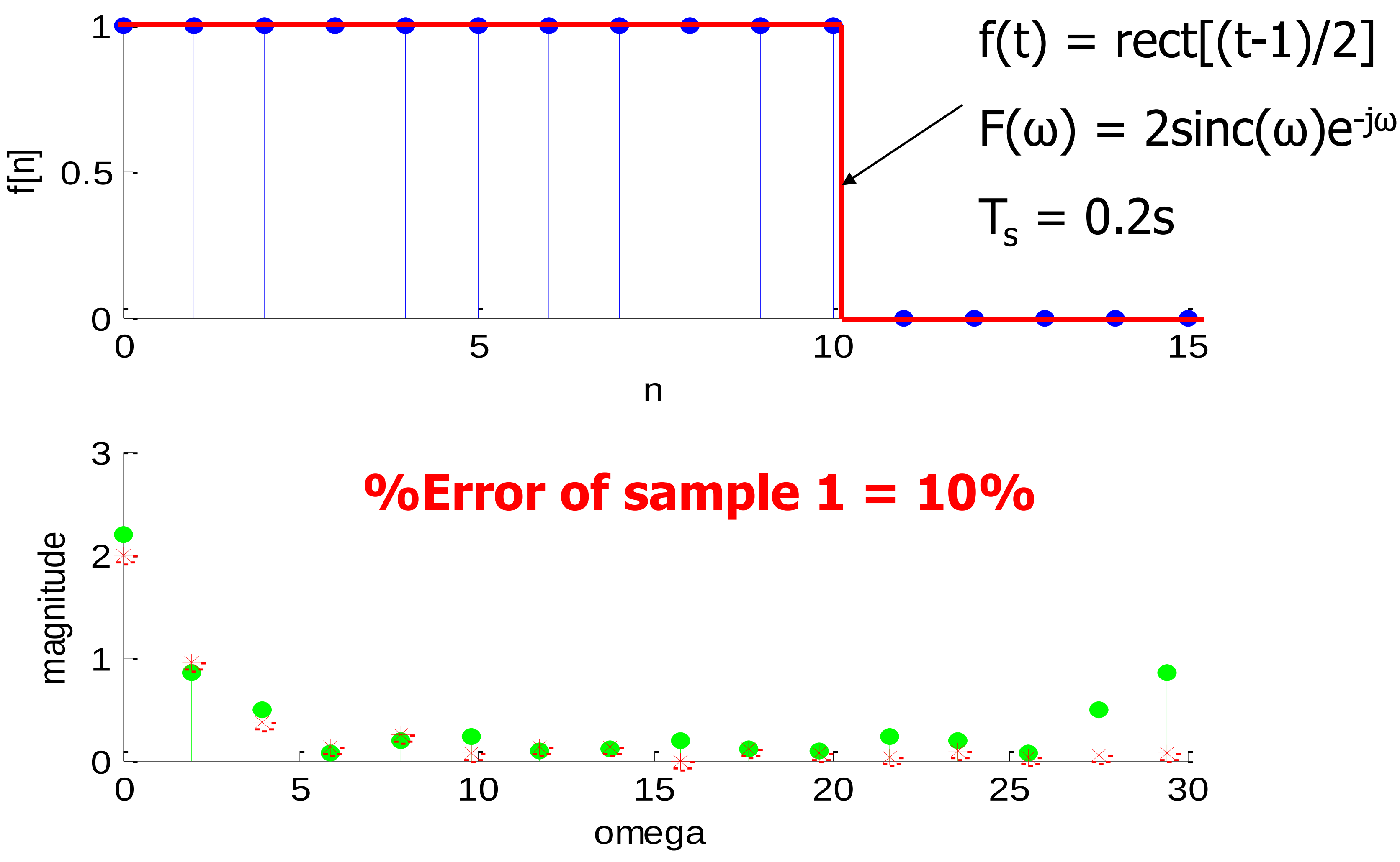
Step 4. Calculate DFT directly or using FFT algorithm

Example



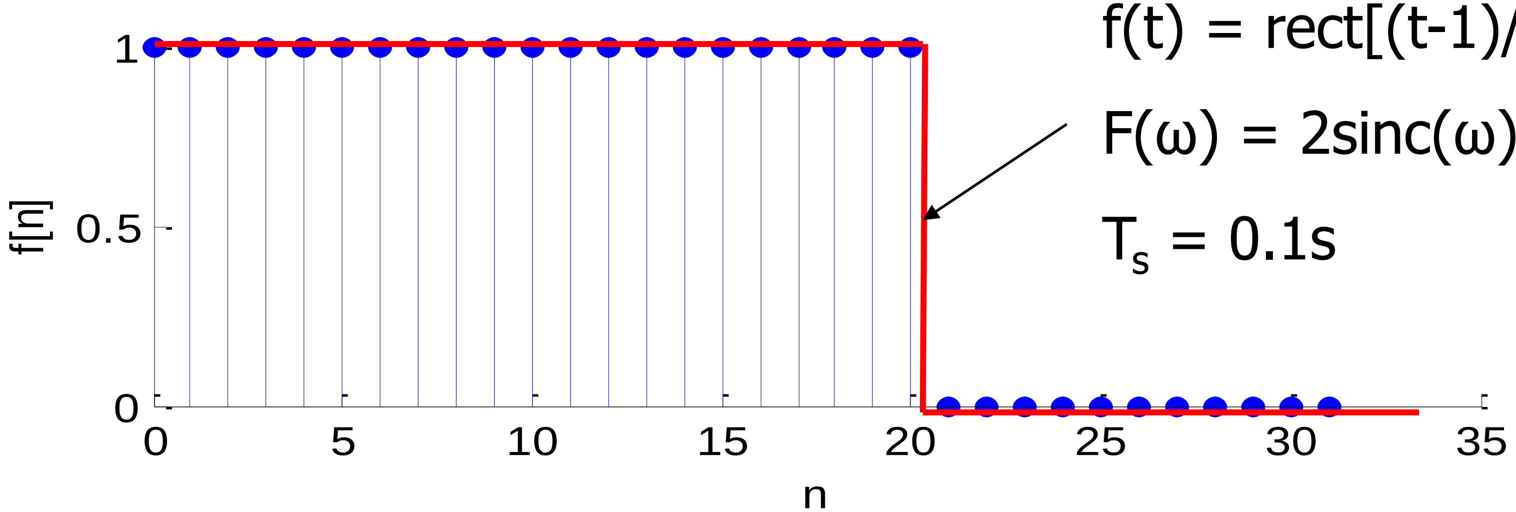
1
0.5
0
f[n]
0
0.5
1
1.5
2
2.5
3
n
$f(t) = rect[(t-1)/2]$
$F(\omega) = 2sinc(\omega)e^{-j\omega}$
$T_s = 1s$
3
2
1
0
magnitude
0
1
2
3
4
5
omega
%Error of sample 1 = 50%

Example



%Error of sample 1 = 10%

Example

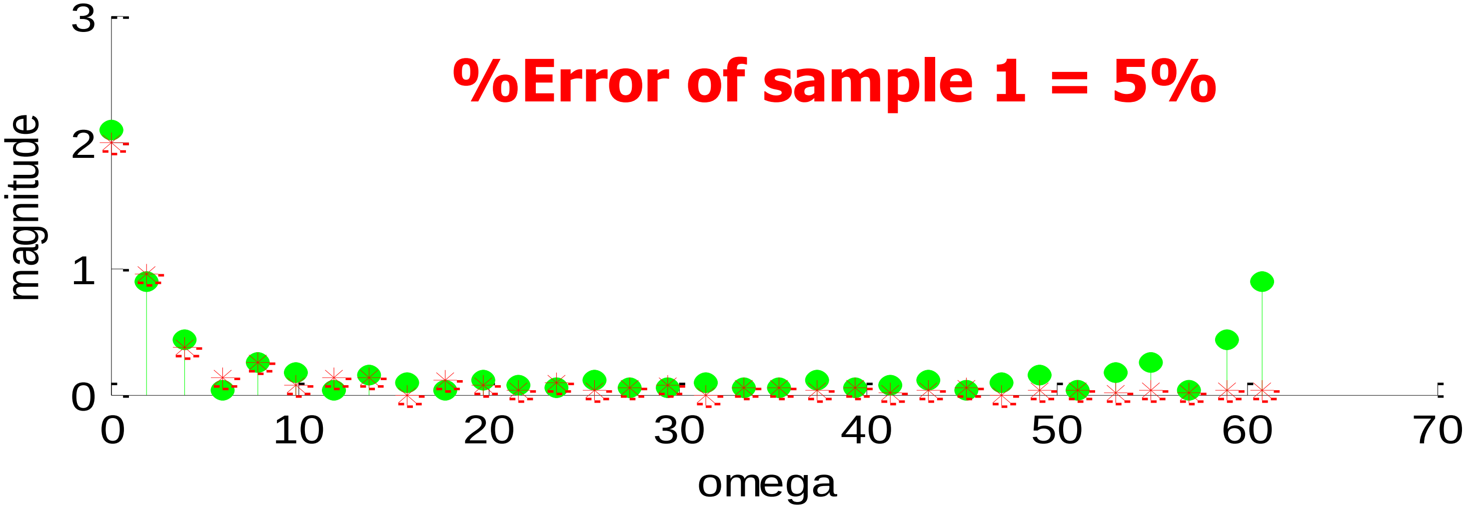


$f(t) = \text{rect}[(t-1)/2]$

$F(\omega) = 2\text{sinc}(\omega)e^{-j\omega}$

$T_s = 0.1s$

%Error of sample 1 = 5%



Windowing

- **Source of error:** truncation or "windowing" of the periodic extension of the DT sequence implied in the DFT development

- **Windowing** = multiplying the periodic extension of x[n] by a rectangular function with duration N.T → DFT involves a convolution

DFT = sampled(DTFT*sinc)

- This multiplication causes spectrum-leakage distortion

- To reduce the affect of spectrum-leakage distortion:

- Increase the sampling frequency
- Increase the number of samples
- Choose an appropriate windowing function (Hamming, Hanning)

Lecture #23

DFT applications

1. Approximation of the Fourier Transform of analog signals

2. Linear convolution

Recall linear convolution

$$y[n] = x_1[n] * x_2[n] = \sum_{p=-\infty}^{\infty} x_1[p] x_2[n-p]$$

- N_1: the non-zero length of $x_1(n)$; N_2: the non-zero length of $x_2(n)$; $N_y = N_1 + N_2 - 1$
- The shift operation is the regular shift
- The flip operation is the regular flip

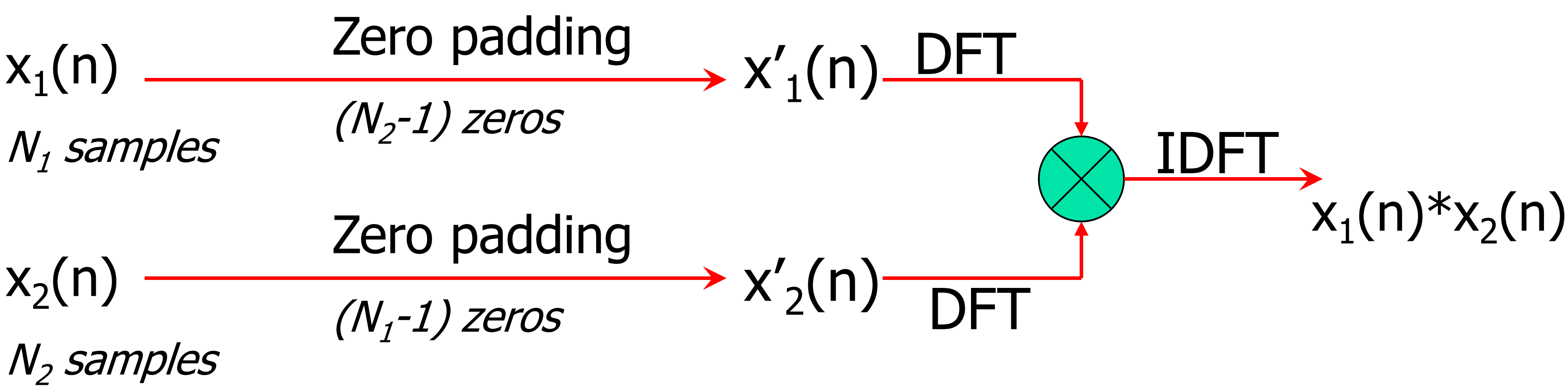
Recall circular convolution

$$y[n] = x_1[n] \otimes x_2[n] = \sum_{p=0}^{N-1} x_1[p] x_2[n-p]$$

- The non-zero length of $x_1(n)$ and $x_2(n)$ can be no longer than N
- The shift operation is circular shift
- The flip operation is circular flip

Calculation of the linear convolution

The circular convolution of 2 sequences of length N_1 and N_2 can be made **equal to** the linear convolution of 2 sequences by zero padding both sequences so that they both consists of N_1+N_2-1 samples.



Example of calculation of the linear convolution

$x_1(n) = [\ 1\ 2\ 3\ 4\ ]$; $x_2(n) = [\ 0\ 1\ 2\ 3\ ]$

$x'_1(n) = [\ 1\ 2\ 3\ 4\ 0\ 0\ 0\ ]$; $x'_2(n) = [\ 0\ 1\ 2\ 3\ 0\ 0\ 0\ ]$

$X'_1(k) = [\ 10, -2.0245-j6.2240, 0.3460+j2.4791, 0.1784-j2.4220, 0.1784+j2.4220, 0.3460-j2.4791, -2.0245-j6.2240\ ]$;

$X'_2(k) = [\ 6, -2.5245-j4.0333, -0.1540+j2.2383, -0.3216-j1.7950, -0.3216+j1.7950, -0.1540-j2.2383, -2.5245+j4.0333]$;

$Y'(k) = [\ 60, -19.9928+j23.8775, -5.6024+j0.3927, -5.8342-j0.8644, -4.4049+j0.4585, -5.6024-j0.3927, -19.9928+j23.8775\ ]$

$\mathrm{IDFT}\{Y'(k)\} = y'(n) = [\ 0\ 1\ 4\ 10\ 16\ 17\ 12\ ]$

Lecture #24

Using DFT/FFT in filtering

- The response, *y(n)* of a filter with the impulse response, *h(n)* to an input, *x(n)* can be calculated by using the DFT
- **Limitations:** All sample values of the signal must be accumulated before the process begins

→ Great memory

→ Long time delay

→ Not suitable for long duration input

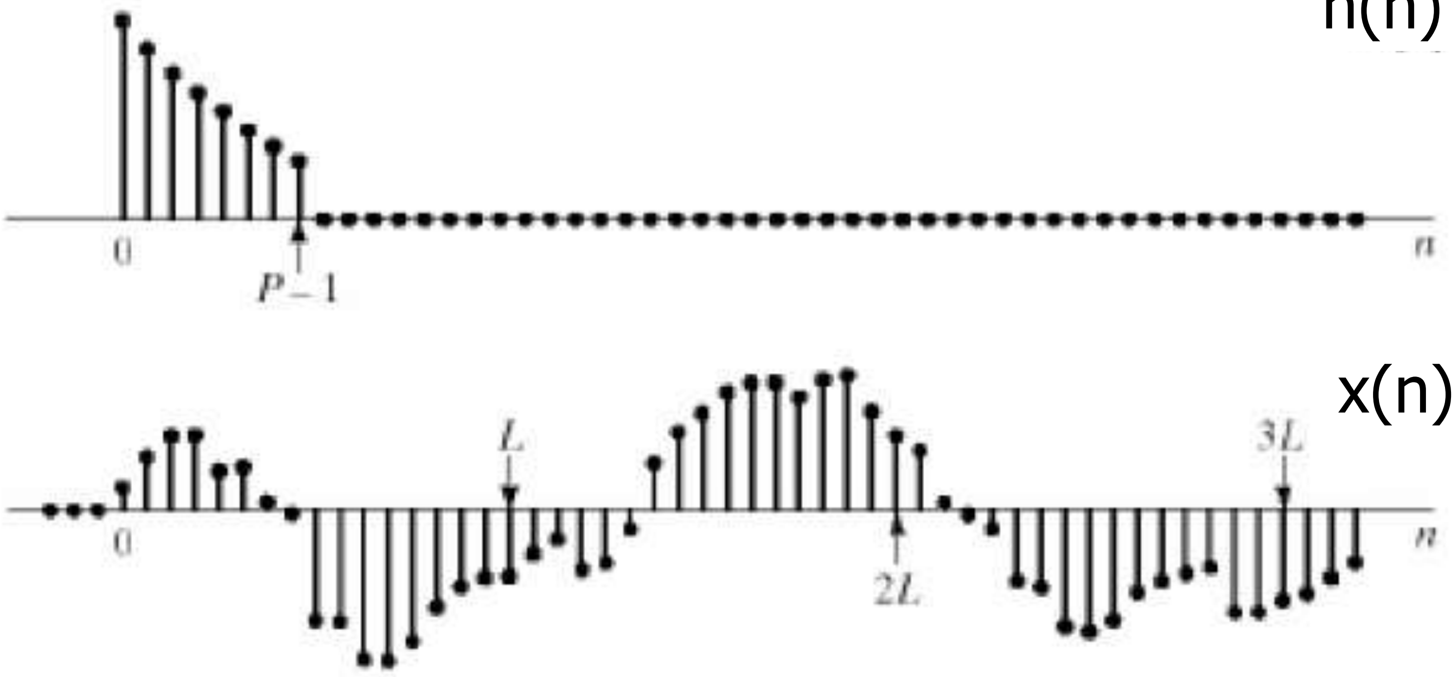
- **Solution:** block filtering

Overlap-add technique in block filtering

- Divide long input signal, $x(n)$ into non-overlapped blocks, $x_b(n)$ of appropriate length for FFT calculation.
- Convolve each block $x_b(n)$ and $h(n)$ to get the output, $y_b(n)$
- Overlap-add outputs, $y_b(n)$ together to form the output signal, $y(n)$

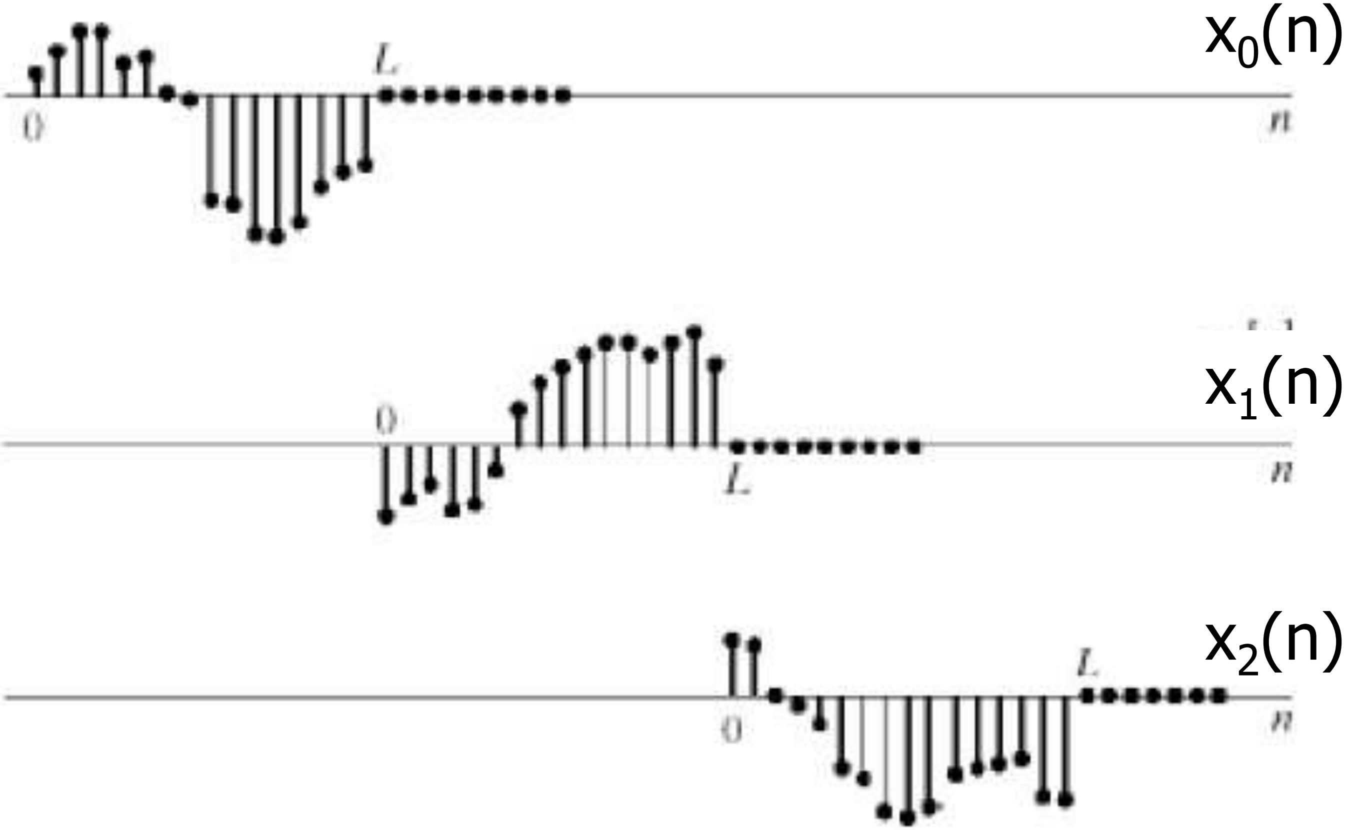
Example of overlap-add technique

h(n)



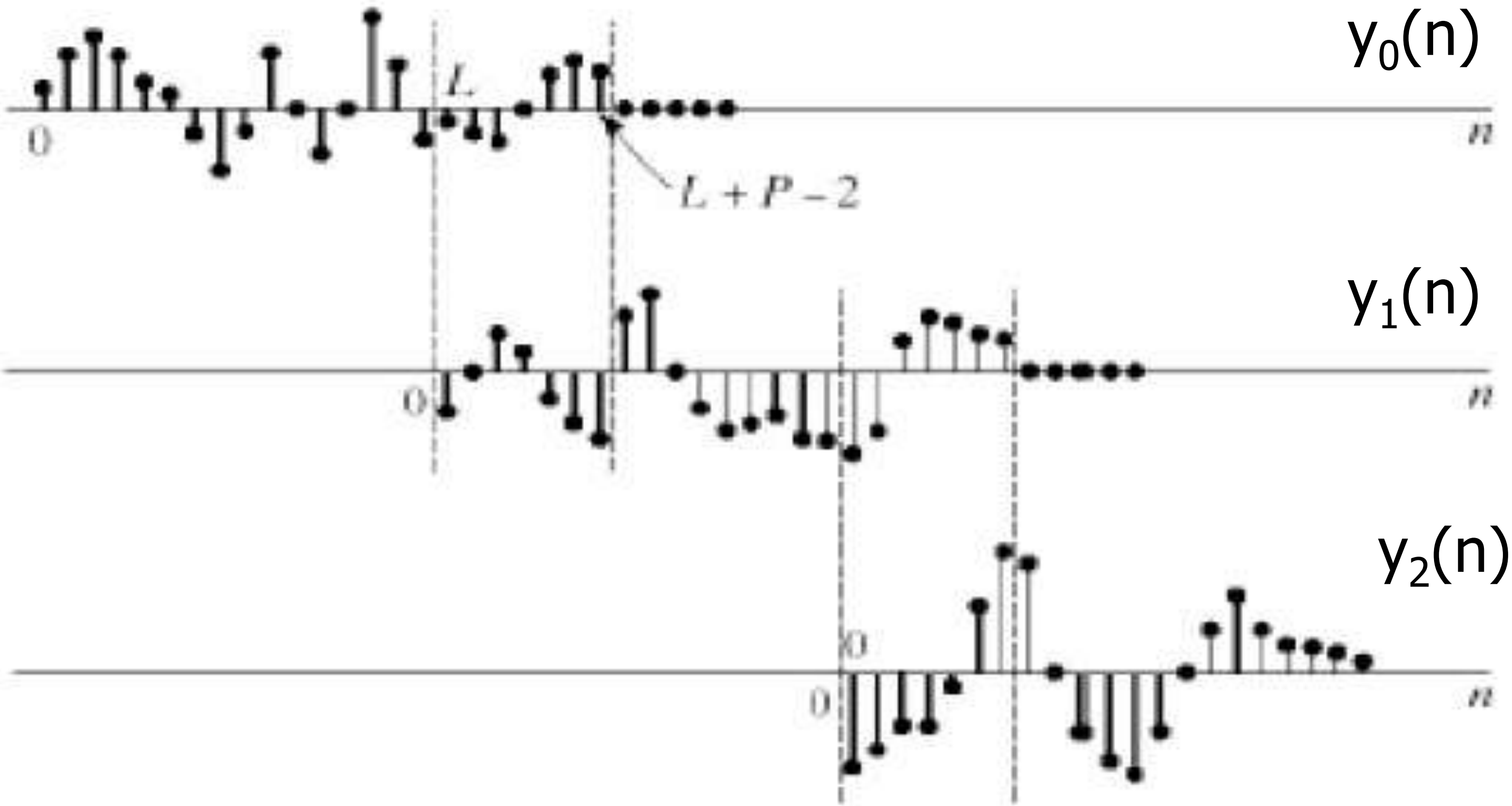
x(n)

Example of overlap-add technique



Break up x[n] into non-overlapping sections of length L

Example (cont)



Convolve each section with *h[n]*

Then add the results to get *y[n]*