Chapter 5B

Finite-Length Discrete Transforms

Part B

Discrete Fourier Transform Properties

DFT Properties

- **Circular Shift of a Sequence**
- **Circular Convolution**
- **Computation of the DFT of Real Sequences**
- **Linear Convolution Using the DFT**

DFT Properties

- Like the DTFT, the DFT also satisfies a number of properties that are useful in signal processing applications
- Some of these properties are essentially identical to those of the DTFT, while some others are somewhat different
- A summary of the DFT properties are given in table 5.3 on page 200

1. Circular Shift of a Sequence

- This property is analogous to the time-shifting property of the DTFT, but with a subtle difference
- Consider length-N sequences defined

$$0 \leq n \leq N-1$$

- The sample values of such sequences are equal to zero for values of $n < 0$ and $n \geq N$

1. Circular Shift of a Sequence

- If $x(n)$ is such a sequence, then for any non-zero arbitrary integer, the shifted sequence

$$x_1(n)=x_1(n-n_0)$$

is no longer defined for the range $0 \leq n \leq N-1$

- We thus need to define another type of a shift that will always keep the shifted sequence in the range $0 \leq n \leq N-1$

1. Circular Shift of a Sequence

- The desired shift, called the *circular shift*, is defined using a modulo operation:

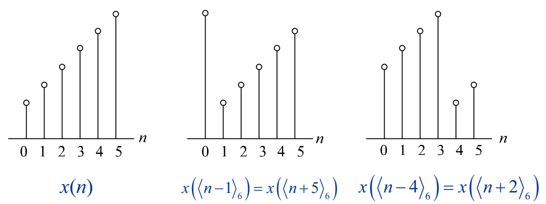
$$x_c(n) = x\left(\langle n-n_0\rangle_N\right)$$

- For $n_0>0$ (*right circular shift*), the above equation implies

$$x_c(n) = \begin{cases} x(n-n_0), & \text{for } n_0 \le n \le N-1 \\ x(N+n-n_0), & \text{for } 0 \le n < n_0 \end{cases}$$

1. Circular Shift of a Sequence

Illustration of the concept of a circular shift



1. Circular Shift of a Sequence

- As can be seen from the previous figure, a right circular shift by n_0 is equivalent to a left circular shift by $N-n_0$ sample periods.
- A circular shift by an integer number n_0 greater than N is equivalent to a circular shift by $\langle n_0\rangle_N$

1. Circular Shift of a Sequence

- DFT of the circular shift sequence

$$y(n) = x\left(\langle n+m\rangle_N\right)R_N\left(\langle n+m\rangle_N\right)$$
$$Y(k) = DFT\left[y(n)\right]$$
$$= \sum_{n=0}^{N-1} x\left(\langle n+m\rangle_N\right)R_N(n)W_N^{kn}$$
$$= \sum_{n=0}^{N-1} x\left(\langle n+m\rangle_N\right)W_N^{kn}$$

1. Circular Shift of a Sequence

$$Y(k) = \sum_{n'=m}^{N-1+m} x\left(\langle n'\rangle_N\right)W_N^{k(n'-m)}$$
$$= W_N^{-km}\sum_{n'=m}^{N-1+m} x\left(\langle n'\rangle_N\right)W_N^{kn'}$$
$$= W_N^{-km}\left(\sum_{n'=0}^{N-1}(.) - \sum_{n'=0}^{m-1}(.) + \sum_{n'=N}^{N-1+m}(.)\right)$$
$$= W_N^{-km}\sum_{n'=0}^{N-1} x\left(\langle n'\rangle_N\right)W_N^{kn'}$$
$$= W_N^{-km}\sum_{n'=0}^{N-1} x(n')W_N^{kn'} = W_N^{-km}X(k)$$

2. Circular Convolution

- Circular convolution is analogous to linear convolution, but with a subtle difference
- Comparison of linear convolution with circular convolution
- Consider two length-N sequences, $g(n)$ and $h(n)$ respectively

2. Circular Convolution

	linear convolution	circular convolution
Length of convolution	$2N-1$	*to be specified*
Convolution Formulas	$y_L(n)=\sum_{m=0}^{N-1} g(m)h(n-m)$	$y_C(n)=\sum_{m=0}^{N-1} g(m)h(\langle n-m\rangle_N)$
Convolution Signs	⊛ or $*$	Ⓝ
Condition of equivalence	?	

2. Circular Convolution

- To develop a convolution-like operation resulting in a length-N sequence $y_C(n)$, we need to define a circular time-reversal, and then apply a circular time-shift.
- Resulting operation, called a *circular convolution*, is defined by

$$y_C(n)=\sum_{m=0}^{N-1} g(m)h(\langle n-m\rangle_N),\quad 0\le n\le N-1$$

2. Circular Convolution

- Since the operation defined involves two length-N sequences, it is often referred to as an N-point circular convolution, denoted as

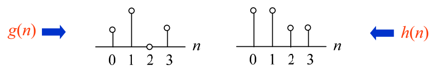
$$y_C(n)=g(n)\,Ⓝ\,h(n)$$

- The circular convolution is commutative, i.e.

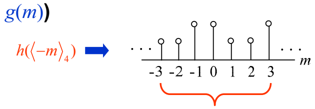
$$g(n)\,Ⓝ\,h(n)=h(n)\,Ⓝ\,g(n)$$

2. Circular Convolution

Example 1 *Length of Circular Convolution is* 4



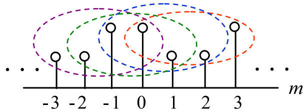
Step 1: Perform *Circular time-reversal operation* on $h(m)$ **(or** $g(m)$**)**



These seven samples are enough to calculate the circular convolution because of the periodicity of DFT

2. Circular Convolution

Step 2: Perform *Circular time-shift operation*



Red	$h(\langle -m\rangle_4)R_4(m)$	{2 1 1 2}
Blue	$h(\langle 1-m\rangle_4)R_4(m)$	{2 2 1 1}
Green	$h(\langle 2-m\rangle_4)R_4(m)$	{1 2 2 1}
Purple	$h(\langle 3-m\rangle_4)R_4(m)$	{1 1 2 2}

2. Circular Convolution

Step 3: Perform *multiplication and summation of sequences* over the region of $0\le m\le 3$ **for** $n=0, n=1, n=2$ **and** $n=3$ **respectively**

```
       1 2 0 1
       2 1 1 2
y(0)=  2+2+0+2= 6

       1 2 0 1
       2 2 1 1
y(1)=  2+4+0+1= 7

       1 2 0 1
       1 2 2 1
y(2)=  1+4+0+1= 6

       1 2 0 1
       1 1 2 2
y(3)=  1+2+0+2= 5
```

2. Circular Convolution

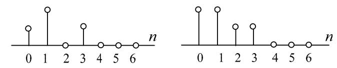
Example 2 *Length of Circular Convolution is 7*

- In order to develop the 7-point circular convolution on the sequences in the former example, we extended these two sequences to length 7 by appending each with 3 zero-valued samples, i.e.

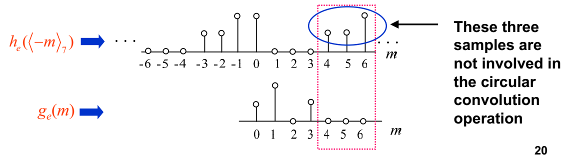
$$g_e(n)=\begin{cases} g(n), & 0\le n\le 3 \\ 0, & 4\le n\le 6 \end{cases}$$

$$h_e(n)=\begin{cases} h(n), & 0\le n\le 3 \\ 0, & 4\le n\le 6 \end{cases}$$

2. Circular Convolution



Perform *Circular time-reversal operation* on $h_e(m)$



2. Circular Convolution

In this case, hence, the procedure of circular convolution is equivalent to that of linear convolution over the region of principle value

Obviously, this conclusion always holds when the length of Circular Convolution is not less than 7

Summary

Provided that the length of Circular Convolution is not less than $N+M-1$ where N and M are the lengths of the two sequences involved, the procedure of circular convolution is equivalent to that of linear convolution

3. Classification of Finite-Length Sequences

- **Based on Conjugate Symmetry**

  It has been discussed in Ch.2

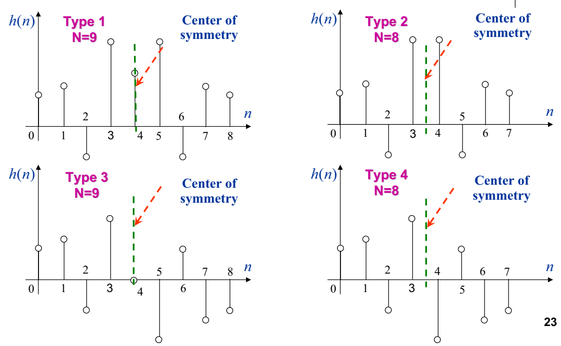
- **Based on Geometric Symmetry**

  A length-N ***symmetry*** sequence $x(n)$ satisfies the condition $x(n)=x(N-1-n)$

  A length-N ***antisymmetry*** sequence $x(n)$ satisfies the condition

$$x(n)=-x(N-1-n)$$

3. Classification of Finite-Length Sequences



4. Computation of the DFT of Real Sequences

- In most practical applications, sequences of interest are real
- In such cases, the symmetry properties of the DFT given in Table 5.2 can be exploited to make the DFT computations more efficient

4. Computation of the DFT of Real Sequences

- ***N*-Point DFTs of Two Real Sequences Using a Single *N*-Point DFT**
- **2*N*-Point DFTs of a Real Sequence Using a Single *N*-Point DFT**

4.1 *N*-Point DFTs of Two Real Sequences Using a Single *N*-Point DFT

- Let $g(n)$ and $h(n)$ be two length-N real sequences with $G(k)$ and $H(k)$ denoting their respective N-point DFTs
- These two N-point DFTs can be computed efficiently using a single N-point DFT
- Define a complex length-N sequence

$$x(n) = g(n) + j\,h(n)$$

- Hence, $g(n) = \text{Re}\{x(n)\}$ and $h(n) = \text{Im}\{x(n)\}$

4.1 *N*-Point DFTs of Two Real Sequences Using a Single *N*-Point DFT

- Let $X(k)$ denote the N-point DFT of $x(n)$
- Then, from Table 5.1 we arrive at

$$G(k) = \frac{1}{2}\{X(k) + X^*(\langle -k \rangle_N)\}$$

$$H(k) = \frac{1}{2j}\{X(k) - X^*(\langle -k \rangle_N)\}$$

- Note that

$$X^*(\langle -k \rangle_N) = X^*(\langle N-k \rangle_N)$$

4.1 *N*-Point DFTs of Two Real Sequences Using a Single *N*-Point DFT

Example

- We compute the 4-point DFTs of the two real sequences $g(n)$ and $h(n)$ given below

$$\{g(n)\} = \{\underset{\uparrow}{1} \;\; 2 \;\; 0 \;\; 1\}, \quad \{h(n)\} = \{\underset{\uparrow}{2} \;\; 2 \;\; 1 \;\; 1\}$$

- Then $\{x(n)\} = \{g(n)\} + j\{h(n)\}$ is given by

$$\{x(n)\} = \{1+j2 \;\; 2+j2 \;\; j \;\; 1+j\}$$

4.1 *N*-Point DFTs of Two Real Sequences Using a Single *N*-Point DFT

- We can work out the 4-point DFT of $x(n)$

$$\{X(k)\} = \{4+j6 \;\; 2 \;\; -2 \;\; j2\}$$

- From the above

$$\{X^*(k)\} = \{4-j6 \;\; 2 \;\; -2 \;\; -j2\}$$

- Hence

$$\{X^*(\langle N-k \rangle_N)\} = \{4-j6 \;\; -2j \;\; -2 \;\; 2\}$$

4.1 *N*-Point DFTs of Two Real Sequences Using a Single *N*-Point DFT

- Therefore

$$\{G(k)\} = \{4 \;\; 1-j \;\; -2 \;\; 1+j\}$$

$$\{H(k)\} = \{6 \;\; 1-j \;\; 0 \;\; 1+j\}$$

4.2 2*N*-Point DFT of a Real Sequence Using an *N*-Point DFT

- Let $v(n)$ be a length-$2N$ real sequence with an $2N$-point DFT $V(k)$
- Define two length-N real sequences $g(n)$ and $h(n)$ as follows:

  $g(n)=v(2n),\ h(n)=v(2n+1)\quad 0\leqslant n\leqslant N-1$
- Let $G(k)$ and $H(k)$ denote their respective N point DFTs

4.2 2*N*-Point DFT of a Real Sequence Using an *N*-Point DFT

$$
\begin{aligned}
V(k) &= \sum_{n=0}^{2N-1} v(n) W_{2N}^{nk} \\
&= \sum_{n=0}^{N-1} v(2n) W_{2N}^{2nk} + \sum_{n=0}^{N-1} v(2n+1) W_{2N}^{(2n+1)k} \\
&= \sum_{n=0}^{N-1} g(n) W_{N}^{nk} + \sum_{n=0}^{N-1} h(n) W_{N}^{nk} W_{2N}^{k} \\
&= \sum_{n=0}^{N-1} g(n) W_{N}^{nk} + W_{2N}^{k} \sum_{n=0}^{N-1} h(n) W_{N}^{nk},\ \ 0 \le k \le 2N-1
\end{aligned}
$$

4.2 2*N*-Point DFT of a Real Sequence Using an *N*-Point DFT

$$V(k)=\sum_{n=0}^{N-1} g(n) W_{N}^{nk} + W_{2N}^{k} \sum_{n=0}^{N-1} h(n) W_{N}^{nk},\ \ 0 \le k \le 2N-1$$

i.e.

$$V(k)=G(\langle k\rangle_N)+W_{2N}^{k} H(\langle k\rangle_N)\quad 0 \le k \le 2N-1$$

where the DFTs of $G(k)$ and $H(k)$ can be computed by means of the method discussed in 4.1

5. Linear Convolution Using the DFT

- *Linear convolution* is a key operation in many signal processing applications.
- Since a DFT can be efficiently implemented using FFT algorithms, it is of interest to develop methods for the implementation of linear convolution using the DFT.

5.1 Linear Convolution of Two Finite-Length Sequences

- Let $g(n)$ and $h(n)$ be two finite-length sequences of length N and M, respectively
- Denote $L=N+M-1$
- Define two length-L sequences

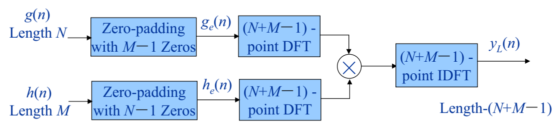
$$g_e(n)=\begin{cases} g(n), & 0 \le n \le N-1 \\ 0, & N \le n \le L-1 \end{cases}$$

$$h_e(n)=\begin{cases} h(n), & 0 \le n \le M-1 \\ 0, & M \le n \le L-1 \end{cases}$$

5.1 Linear Convolution of Two Finite-Length Sequences

- Then

  $y_L(n)=g(n)\circledast h(n)=g(n)\ \textcircled{N}\ h(n)$
- The corresponding implementation scheme is illustrated below



5.2 Linear Convolution of a Finite-Length Sequence with an Infinite-Length Sequence

- We next consider the DFT-based implementation of

$$y(n)=\sum_{l=0}^{M-1} h(l)x(n-l)=h(n)\circledast x(n)$$

where $h(n)$ is a finite-length sequence of length M and $x(n)$ is an infinite length (or a finite length sequence whose length is much greater than M)

5.2 Overlap-Add Method

- We first segment $x(n)$, assumed to be a causal sequence here without (any) loss of generality, into a set of contiguous finite-length subsequences of length N each:

$$x(n)=\sum_{m=0}^{\infty} x_m(n-mN)$$

where

$$x_m(n)=\begin{cases} x(n+mN), & 0\le n\le N-1 \\ 0, & \text{otherwise}\end{cases}$$

5.2 Overlap-Add Method

- Thus we can write

$$y(n)=h(n)\circledast x(n)=\sum_{m=0}^{\infty} y_m(n-mN)$$

where

$$y_m(n)=h(n)\circledast x_m(n)$$

- Since $h(n)$ is of length M and $x_m(n)$ is of length N, the linear convolution $y_m(n)=h(n)\circledast x_m(n)$ is of length $N+M-1$

5.2 Overlap-Add Method

- As a result, the desired linear convolution $y(n)=h(n)\circledast x(n)$ has been broken up into a sum of infinite number of short-length linear convolutions of length $N+M-1$ each:

$$y_m(n)=h(n)\circledast x_m(n)$$

- Each of these short convolutions can be implemented using the DFT-based method discussed earlier, where the DFTs (and the IDFT) are computed on the basis of $(N+M-1)$ points

5.2 Overlap-Add Method

- There is one more subtlety to take care of before we can implement

$$y(n)=\sum_{m=0}^{\infty} y_m(n-mN)$$

using the DFT-based approach

- Now the first convolution in the above sum, $y_0(n)=h(n)\circledast x_0(n)$ is of length $N+M-1$ and is defined for $0\leqslant n\leqslant N+M-2$

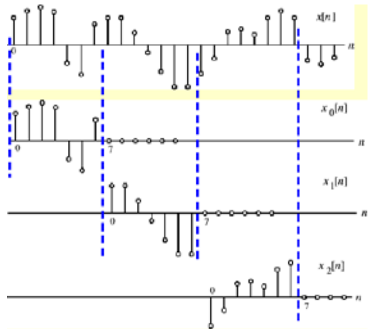
5.2 Overlap-Add Method

- The second short convolution $y_1(n)=h(n)\circledast x_1(n)$ is also of length $N+M-1$ but is defined for $N\leqslant n\leqslant 2N+M-2$
- ➡ There is an overlap of $M-1$ samples between these two short linear convolutions
- Likewise, the third short convolution $y_2(n)=h(n)\circledast x_2(n)$, is also of length $N+M-1$ but is defined for $2N\leqslant n\leqslant 3N+M-2$

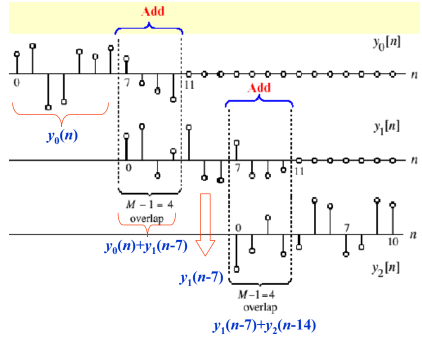
5.2 Overlap-Add Method

- Thus there is an overlap of $M-1$ samples between $h(n) \circledast x_1(n)$ and $h(n) \circledast x_2(n)$
- In general, there will be an overlap of $M-1$ samples between the samples of the short convolutions $h(n) \circledast x_{r-1}(n)$ and $h(n) \circledast x_r(n)$
- This process is illustrated in the figure on the next slide for $M = 5$ and $N = 7$.

5.2 Overlap-Add Method



5.2 Overlap-Add Method



5.2 Overlap-Add Method

- The above procedure is called the *overlap add method* since the results of the short linear convolutions overlap and the overlapped portions are added to get the correct final result.
- The function **`fftfilt`** can be used to implement the above method.
- Program 5_5 illustrates the use of **`fftfilt`** in the filtering of a noise-corrupted signal using a length-3 *moving average filter*