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

N-Point DFTs of Two Length-*N* Real Sequences

- 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, $g[n] = \mathcal{R}e\{x[n]\}$ and $h[n] = \mathcal{I}m\{x[n]\}$

N-Point DFTs of Two Length-*N* Real Sequences

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

$$G[k] = \tfrac{1}{2}\{X[k] + X^*[\langle -k\rangle_N]\}$$

$$H[k] = \tfrac{1}{2j}\{X[k] - X^*[\langle -k\rangle_N]\}$$

- Note that for $0 \le k \le N-1$,

$$X^*[\langle -k\rangle_N] = X^*[\langle N-k\rangle_N]$$

N-Point DFTs of Two Length-*N* Real Sequences

- Example - We compute the 4-point DFTs of the two real sequences $g[n]$ and $h[n]$ given below

$$\{g[n]\} = \{\underset{\uparrow}{1} \quad 2 \quad 0 \quad 1\}, \quad \{h[n]\} = \{\underset{\uparrow}{2} \quad 2 \quad 1 \quad 1\}$$

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

$$\{x[n]\} = \{\underset{\uparrow}{1+j2} \quad 2+j2 \quad j \quad 1+j\}$$

N-Point DFTs of Two Length-*N* Real Sequences

- Its DFT $X[k]$ is

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1+j2 \\ 2+j2 \\ j \\ 1+j \end{bmatrix} = \begin{bmatrix} 4+j6 \\ 2 \\ -2 \\ j2 \end{bmatrix}$$

- From the above

$$X^*[k] = [4-j6 \quad 2 \quad -2 \quad -j2]$$

- Hence

$$X^*[\langle 4-k\rangle_4] = [4-j6 \quad -j2 \quad -2 \quad 2]$$

N-Point DFTs of Two Length-*N* Real Sequences

- Therefore

$$\{G[k]\} = \{4 \quad 1-j \quad -2 \quad 1+j\}$$

$$\{H[k]\} = \{6 \quad 1-j \quad 0 \quad 1+j\}$$

verifying the results derived earlier

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:

$$g[n] = v[2n], \quad h[n] = v[2n+1], \quad 0 \le n \le N$$

- Let $G[k]$ and $H[k]$ denote their respective N-point DFTs

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

- Define a length-N complex sequence

$$\{x[n]\} = \{g[n]\} + j\{h[n]\}$$

with an N-point DFT $X[k]$

- Then as shown earlier

$$G[k] = \tfrac{1}{2}\{X[k] + X^*[\langle -k\rangle_N]\}$$

$$H[k] = \tfrac{1}{2j}\{X[k] - X^*[\langle -k\rangle_N]\}$$

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

- Now $V[k] = \sum_{n=0}^{2N-1} v[n] W_{2N}^{nk}$

$$= \sum_{n=0}^{N-1} v[2n] W_{2N}^{2nk} + \sum_{n=0}^{N-1} v[2n+1] W_{2N}^{(2n+1)k}$$

$$= \sum_{n=0}^{N-1} g[n] W_N^{nk} + \sum_{n=0}^{N-1} h[n] W_N^{nk} W_{2N}^{k}$$

$$= \sum_{n=0}^{N-1} g[n] W_N^{nk} + W_{2N}^{k} \sum_{n=0}^{N-1} h[n] W_N^{nk}, \ 0 \le k \le 2N-1$$

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

- i.e.,

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

- Example - Let us determine the 8-point DFT V[k] of the length-8 real sequence

$$\{v[n]\} = \{\underset{\uparrow}{1} \quad 2 \quad 2 \quad 2 \quad 0 \quad 1 \quad 1 \quad 1\}$$

- We form two length-4 real sequences as follows

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

$$\{g[n]\} = \{v[2n]\} = \{\underset{\uparrow}{1} \quad 2 \quad 0 \quad 1\}$$

$$\{h[n]\} = \{v[2n+1]\} = \{\underset{\uparrow}{2} \quad 2 \quad 1 \quad 1\}$$

- Now

$$V[k] = G[\langle k\rangle_4] + W_8^{k} H[\langle k\rangle_4], \quad 0 \le k \le 7$$

- Substituting the values of the 4-point DFTs $G[k]$ and $H[k]$ computed earlier we get

---

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

$$V[0] = G[0] + H[0] = 4 + 6 = 10$$

$$V[1] = G[1] + W_8^{1} H[1]$$

$$= (1 - j) + e^{-j\pi/4}(1 - j) = 1 - j2.4142$$

$$V[2] = G[2] + W_8^{2} H[2] = -2 + e^{-j\pi/2} \cdot 0 = -2$$

$$V[3] = G[3] + W_8^{3} H[3]$$

$$= (1 + j) + e^{-j3\pi/4}(1 + j) = 1 - j0.4142$$

$$V[4] = G[0] + W_8^{4} H[0] = 4 + e^{-j\pi} \cdot 6 = -2$$

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

$$V[5] = G[1] + W_8^5 H[1]$$
$$= (1-j) + e^{-j5\pi/4}(1-j) = 1 + j0.4142$$

$$V[6] = G[2] + W_8^6 H[2] = -2 + e^{-j3\pi/2} \cdot 0 = -2$$

$$V[7] = G[3] + W_8^7 H[3]$$
$$= (1+j) + e^{-j7\pi/4}(1+j) = 1 + j2.4142$$

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

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

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

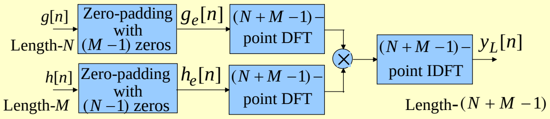
$$h_e[n] = \begin{cases} h[n], & 0 \le n \le M-1 \\ 0, & M \le n \le L-1 \end{cases}$$

Linear Convolution of Two Finite-Length Sequences

- Then

$$y_L[n] = g[n] \circledast h[n] = y_C[n] = g[n] \, Ⓛ \, h[n]$$

- The corresponding implementation scheme is illustrated below



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

- We next consider the DFT-based implementation of

$$y[n] = \sum_{\ell=0}^{M-1} h[\ell]\, x[n-\ell] = h[n] \circledast x[n]$$

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

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 $x_m[n]$ of length N each:

$$x[n] = \sum_{m=0}^{\infty} x_m[n - mN]$$

where

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

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 $h[n] \circledast x_m[n]$ is of length $N + M - 1$

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: $y_m[n] = x_m[n] \circledast h[n]$
- Each of these short convolutions can be implemented using the DFT-based method discussed earlier, where now the DFTs (and the IDFT) are computed on the basis of $(N + M - 1)$ points

---

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 $0 \le n \le N + M - 2$

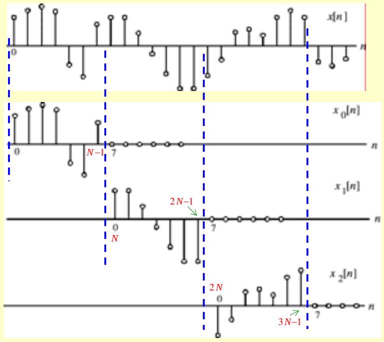
Overlap-Add Method

- The second short convolution $y_1[n] = h[n] \circledast x_1[n]$, is also of length $N + M - 1$ but is defined for $N \le n \le 2N + M - 2$
- ➡ 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 $2N \le n \le 3N + M - 2$

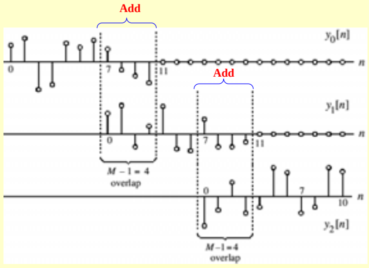
Overlap-Add Method

- 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 $h[n] \circledast x_r[n]$ for $(r - 1)N \le n \le rN + M - 2$
- This process is illustrated in the figure on the next slide for $M = 5$ and $N = 7$

Overlap-Add Method



Overlap-Add Method



Overlap-Add Method

- Therefore, $y[n]$ obtained by a linear convolution of $x[n]$ and $h[n]$ is given by

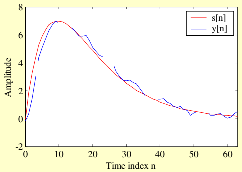
$$\begin{aligned} y[n] &= y_0[n], & 0 \le n \le 6 \\ y[n] &= y_0[n] + y_1[n-7], & 7 \le n \le 10 \\ y[n] &= y_1[n-7], & 11 \le n \le 13 \\ y[n] &= y_1[n-7] + y_2[n-14], & 14 \le n \le 17 \\ y[n] &= y_2[n-14], & 18 \le n \le 20 \\ &\vdots \end{aligned}$$

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

Overlap-Add Method

- Program 5_5 illustrates the use of `fftfilt` in the filtering of a noise-corrupted signal using a length-3 moving average filter
- The plots generated by running this program is shown below



Overlap-Save Method

- In implementing the overlap-add method using the DFT, we need to compute two $(N+M-1)$-point DFTs and one $(N+M-1)$-point IDFT since the overall linear convolution was expressed as a sum of short-length linear convolutions of length $(N+M-1)$ each
- It is possible to implement the overall linear convolution by performing instead circular convolution of length shorter than $(N+M-1)$

Overlap-Save Method

- To this end, it is necessary to segment $x[n]$ into overlapping blocks $x_m[n]$, keep the terms of the circular convolution of $h[n]$ with $x_m[n]$ that corresponds to the terms obtained by a linear convolution of $h[n]$ and $x_m[n]$, and throw away the other parts of the circular convolution

Overlap-Save Method

- To understand the correspondence between the linear and circular convolutions, consider a length-4 sequence $x[n]$ and a length-3 sequence $h[n]$
- Let $y_L[n]$ denote the result of a linear convolution of $x[n]$ with $h[n]$
- The six samples of $y_L[n]$ are given by

Overlap-Save Method

$$y_L[0] = h[0]x[0]$$
$$y_L[1] = h[0]x[1] + h[1]x[0]$$
$$y_L[2] = h[0]x[2] + h[1]x[1] + h[2]x[0]$$
$$y_L[3] = h[0]x[3] + h[1]x[2] + h[2]x[1]$$
$$y_L[4] = h[1]x[3] + h[2]x[2]$$
$$y_L[5] = h[2]x[3]$$

Overlap-Save Method

- If we append $h[n]$ with a single zero-valued sample and convert it into a length-4 sequence $h_e[n]$, the 4-point circular convolution $y_C[n]$ of $h_e[n]$ and $x[n]$ is given by

$$y_C[0] = h[0]x[0] + h[1]x[3] + h[2]x[2]$$
$$y_C[1] = h[0]x[1] + h[1]x[0] + h[2]x[3]$$
$$y_C[2] = h[0]x[2] + h[1]x[1] + h[2]x[0]$$
$$y_C[3] = h[0]x[3] + h[1]x[2] + h[2]x[1]$$

Overlap-Save Method

- If we compare the expressions for the samples of $y_L[n]$ with the samples of $y_C[n]$, we observe that the first 2 terms of $y_C[n]$ do not correspond to the first 2 terms of $y_L[n]$, whereas the last 2 terms of $y_C[n]$ are precisely the same as the 3rd and 4th terms of $y_L[n]$, i.e.,

$$y_L[0] \neq y_C[0], \qquad y_L[1] \neq y_C[1]$$
$$y_L[2] = y_C[2], \qquad y_L[3] = y_C[3]$$

Overlap-Save Method

- General case: N-point circular convolution of a length-M sequence $h[n]$ with a length-N sequence $x[n]$ with $N > M$
- First $M-1$ samples of the circular convolution are incorrect and are rejected
- Remaining $N-M+1$ samples correspond to the correct samples of the linear convolution of $h[n]$ with $x[n]$

Overlap-Save Method

- Now, consider an infinitely long or very long sequence $x[n]$
- Break it up as a collection of smaller length (length-4) overlapping sequences $x_m[n]$ as

$$x_m[n] = x[n+2m], \qquad 0 \le n \le 3, \quad 0 \le m \le \infty$$

- Next, form

$$w_m[n] = h[n] \textcircled{4} x_m[n]$$

Overlap-Save Method

- Or, equivalently,

$$w_m[0] = h[0]x_m[0] + h[1]x_m[3] + h[2]x_m[2]$$
$$w_m[1] = h[0]x_m[1] + h[1]x_m[0] + h[2]x_m[3]$$
$$w_m[2] = h[0]x_m[2] + h[1]x_m[1] + h[2]x_m[0]$$
$$w_m[3] = h[0]x_m[3] + h[1]x_m[2] + h[2]x_m[1]$$

- Computing the above for m = 0, 1, 2, 3, . . . , and substituting the values of $x_m[n]$ we arrive at

---

Overlap-Save Method

$w_0[0] = h[0]x[0] + h[1]x[3] + h[2]x[2]$ ← Reject

$w_0[1] = h[0]x[1] + h[1]x[0] + h[2]x[3]$ ← Reject

$w_0[2] = h[0]x[2] + h[1]x[1] + h[2]x[0] = y[2]$ ← Save

$w_0[3] = h[0]x[3] + h[1]x[2] + h[2]x[1] = y[3]$ ← Save

$w_1[0] = h[0]x[2] + h[1]x[5] + h[2]x[4]$ ← Reject

$w_1[1] = h[0]x[3] + h[1]x[2] + h[2]x[5]$ ← Reject

$w_1[2] = h[0]x[4] + h[1]x[3] + h[2]x[2] = y[4]$ ← Save

$w_1[3] = h[0]x[5] + h[1]x[4] + h[2]x[3] = y[5]$ ← Save

Overlap-Save Method

$w_2[0] = h[0]x[4] + h[1]x[5] + h[2]x[6]$ ← Reject

$w_2[1] = h[0]x[5] + h[1]x[4] + h[2]x[7]$ ← Reject

$w_2[2] = h[0]x[6] + h[1]x[5] + h[2]x[4] = y[6]$ ← Save

$w_2[3] = h[0]x[7] + h[1]x[6] + h[2]x[5] = y[7]$ ← Save

Overlap-Save Method

- It should be noted that to determine $y[0]$ and $y[1]$, we need to form $x_{-1}[n]$:

$$x_{-1}[0] = 0, \quad x_{-1}[1] = 0,$$
$$x_{-1}[2] = x[0], \quad x_{-1}[3] = x[1]$$

  and compute $w_{-1}[n] = h[n] \textcircled{4} x_{-1}[n]$ for $0 \le n \le 3$ reject $w_{-1}[0]$ and $w_{-1}[1]$, and save $w_{-1}[2] = y[0]$ and $w_{-1}[3] = y[1]$

Overlap-Save Method

- General Case: Let $h[n]$ be a length-N sequence
- Let $x_m[n]$ denote the m-th section of an infinitely long sequence $x[n]$ of length N and defined by

$$x_m[n] = x[n + m(N - m + 1)], \quad 0 \le n \le N - 1$$

  with $M < N$

Overlap-Save Method

- Let $w_m[n] = h[n] \textcircled{N} x_m[n]$
- Then, we reject the first $M - 1$ samples of $w_m[n]$ and "abut" the remaining $N - M + 1$ samples of $w_m[n]$ to form $y_L[n]$, the linear convolution of $h[n]$ and $x[n]$
- If $y_m[n]$ denotes the saved portion of $w_m[n]$, i.e.

$$y_m[n] = \begin{cases} 0, & 0 \le n \le M - 2 \\ w_m[n], & M - 1 \le n \le N - 2 \end{cases}$$

Overlap-Save Method

- Then

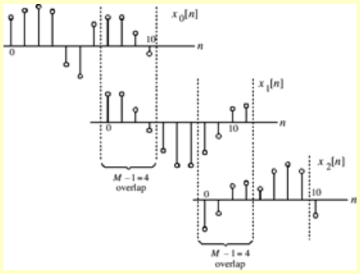
$$y_L[n+m(N-M+1)] = y_m[n], \qquad M-1 \le n \le N-1$$

- The approach is called **overlap-save method** since the input is segmented into overlapping sections and parts of the results of the circular convolutions are saved and abutted to determine the linear convolution result

---

Overlap-Save Method

- Process is illustrated next



Overlap-Save Method