Digital Signal Processing

Ms. Ashwini

Department of Electronics and Communication.

Digital Signal Processing

Linear Filtering methods based on the DFT

Ms. Ashwini
Department of Electronics and Communication.

Overlap-Save Method

Deals with the following signal processing principles:

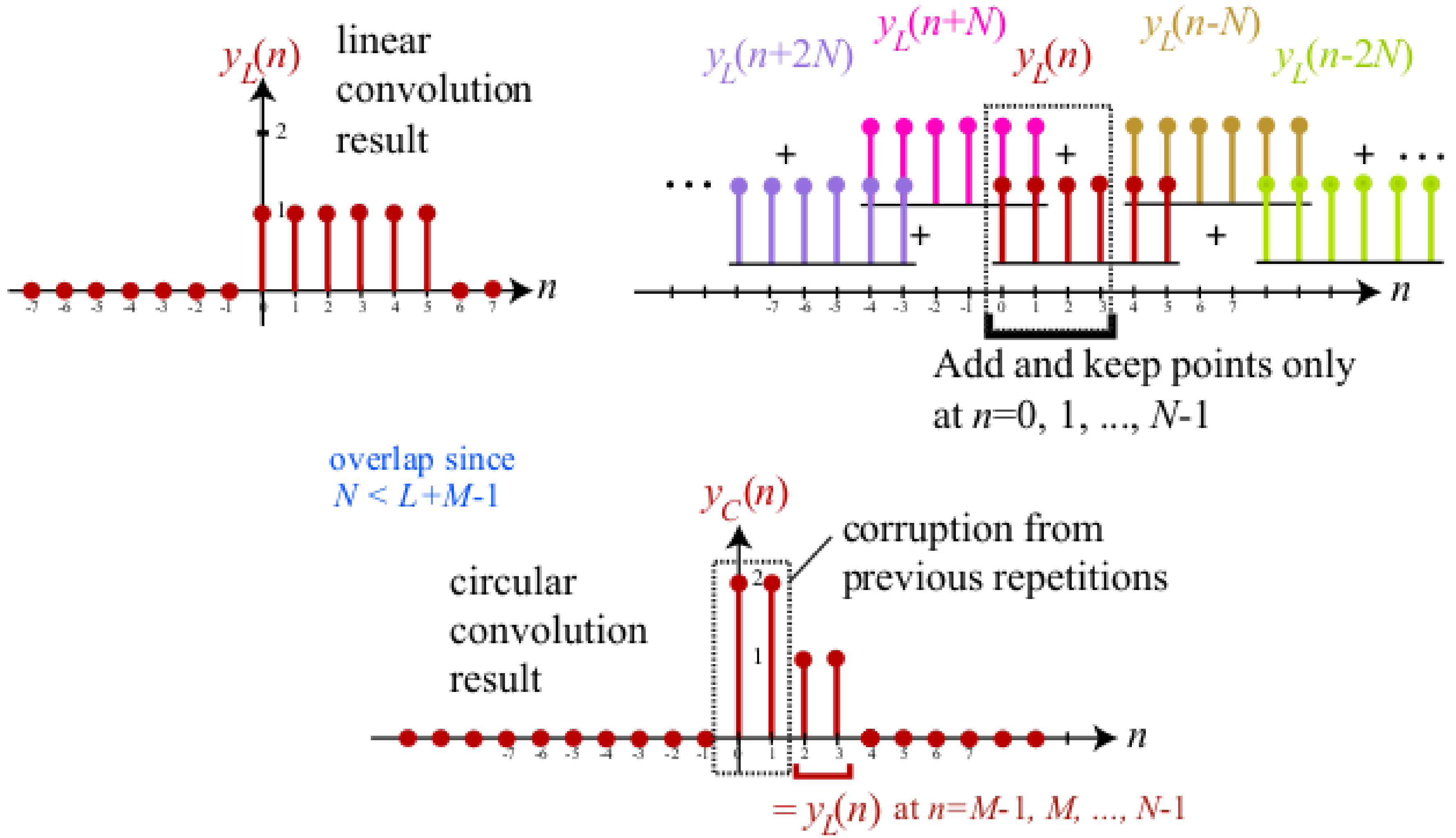
- The $N = (L + M - 1)$-circular convolution of a discrete-time signal of length N and a discrete-time signal of length M using an N-DFT and N-IDFT.

- Time-Domain Aliasing:

$$x_C(n) = \sum_{l=-\infty}^{\infty} \underbrace{x_L(n - lN)}_{\text{support}=M+N-1}, \qquad n = 0, 1, \ldots, N-1$$

Signals, Linear Filtering methods based on the DFT

Overlap-Save Method



Overlap-Save Method

- Convolution of $x_m(n)$ with support $n = 0, 1, \ldots, N-1$ and $h(n)$ with support $n = 0, 1, \ldots, M-1$ via the N-DFT will produce a result $y_{C,m}(n)$ such that:

$$y_{C,m}(n) = \begin{cases} \text{aliasing corruption} & n = 0, 1, \ldots, M-2 \\ y_{L,m}(n) & n = M-1, M, \ldots, N-1 \end{cases}$$

where $y_{L,m}(n) = x_m(n) * h(n)$ is the desired output.

- The first $M-1$ points of a the current filtered output block $y_m(n)$ must be discarded.
- The previous filtered block $y_{m-1}(n)$ must compensate by providing these output samples.

Overlap-Save Method

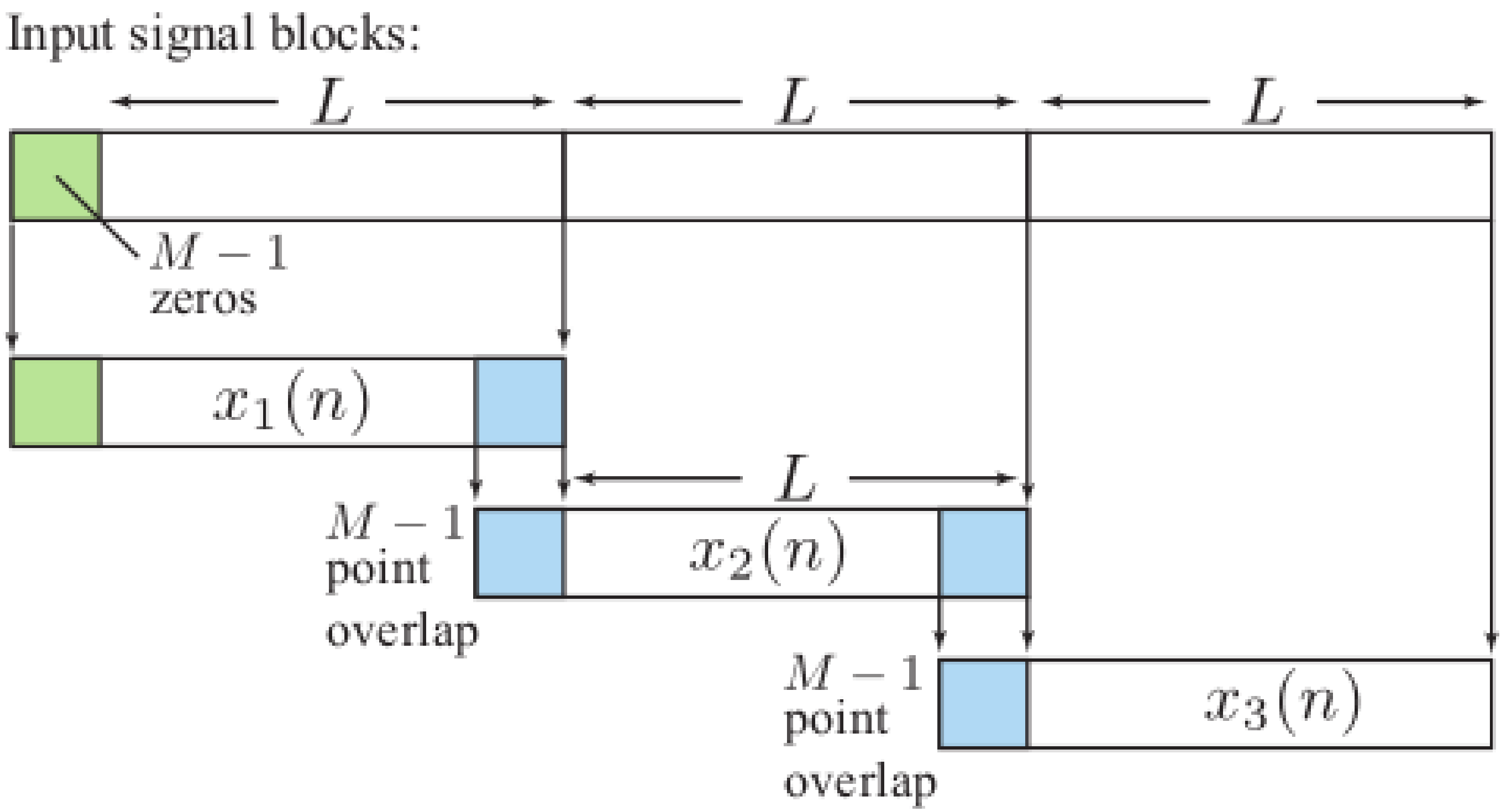
1. All input blocks $x_m(n)$ are of length $N = (L + M - 1)$ and contain sequential samples from $x(n)$.

2. Input block $x_m(n)$ for $m > 1$ overlaps containing the first $M - 1$ points of the previous block $x_{m-1}(n)$ to deal with aliasing corruption.

3. For $m = 1$, there is no previous block, so the first $M - 1$ points are zeros.

Signals, Linear Filtering methods based on the DFT

Overlap-Save Method



Overlap-Save Method

$$x_1(n) = \{\underbrace{0, \quad 0, \quad \ldots \quad 0}_{M-1 \text{ zeros}}, \; x(0), x(1), \ldots, x(L-1)\}$$

$$x_2(n) = \{\underbrace{x(L-M+1), \ldots x(L-1)}_{\text{last } M-1 \text{ points from } x_1(n)}, x(L), \ldots, x(2L-1)\}$$

$$x_3(n) = \{\underbrace{x(2L-M+1), \ldots x(2L-1)}_{\text{last } M-1 \text{ points from } x_2(n)}, x(2L), \ldots, x(3L-1)\}$$

$$\vdots$$

The last $M-1$ points from the previous input block must be saved for use in the current input block.

Signals, Linear Filtering methods based on the DFT

Overlap-Save Method

- makes use of the N-DFT and N-IDFT where: $N = L + M - 1$
  - Only a one-time zero-padding of $h(n)$ of length $M \ll L < N$ is required to give it support $n = 0, 1, \ldots, N-1$.
  - The input blocks $x_m(n)$ are of length N to start, so no zero-padding is necessary.
  - The actual implementation of the DFT/IDFT will use the FFT/IFFT for computational simplicity.

Overlap-Save Method

$N = L + M - 1$.

Let $x_m(n)$ have support $n = 0, 1, \ldots, N - 1$.

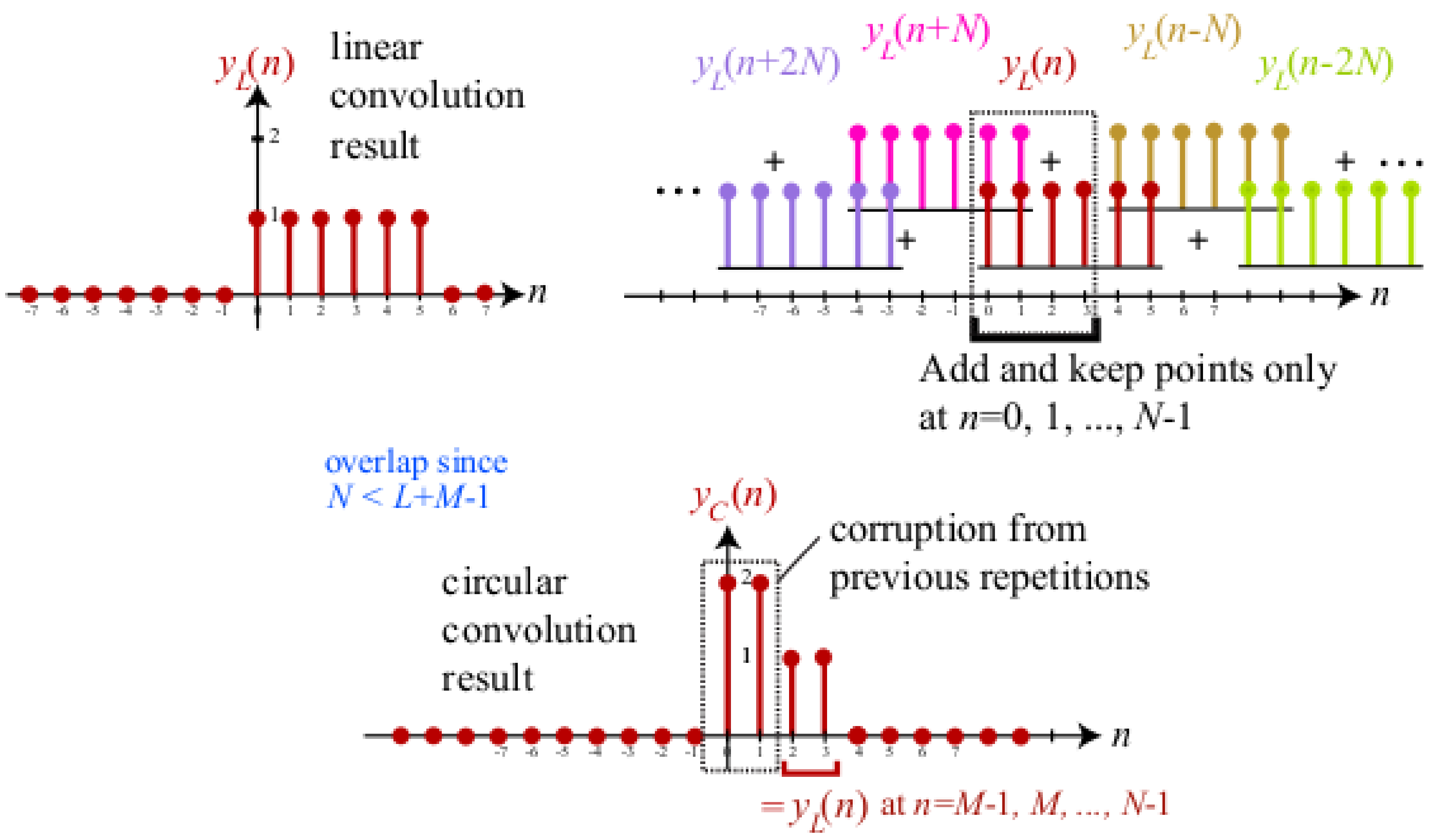
Let $h(n)$ have support $n = 0, 1, \ldots, M - 1$.

We zero pad $h(n)$ to have support $n = 0, 1, \ldots, N - 1$.

1. Take N-DFT of $x_m(n)$ to give $X_m(k)$, $k = 0, 1, \ldots, N - 1$.
2. Take N-DFT of $h(n)$ to give $H(k)$, $k = 0, 1, \ldots, N - 1$.
3. Multiply: $Y_m(k) = X_m(k) \cdot H(k)$, $k = 0, 1, \ldots, N - 1$.
4. Take N-IDFT of $Y_m(k)$ to give $y_{C,m}(n)$, $n = 0, 1, \ldots, N - 1$.

Signals, Linear Filtering methods based on the DFT

Overlap-Save Method



Signals, Linear Filtering methods based on the DFT

Overlap-Save Method

$$y_{C,m}(n) = \begin{cases} \text{aliasing} & n = 0, 1, \ldots, M-2 \\ y_{L,m}(n) & n = M-1, M, \ldots, N-1 \end{cases}$$

where $y_{L,m}(n) = x_m(n) * h(n)$ is the desired output.

Overlap-Save Method

$$y_1(n) = \{\underbrace{y_1(0), \quad y_1(1), \quad \ldots \quad y_1(M-2)}_{M-1 \text{ points corrupted from aliasing}}, \; y(0), \ldots, y(L-1)\}$$

$$y_2(n) = \{\underbrace{y_2(0), \quad y_2(1), \quad \ldots \quad y_2(M-2)}_{M-1 \text{ points corrupted from aliasing}}, y(L), \ldots, y(2L-1)\}$$

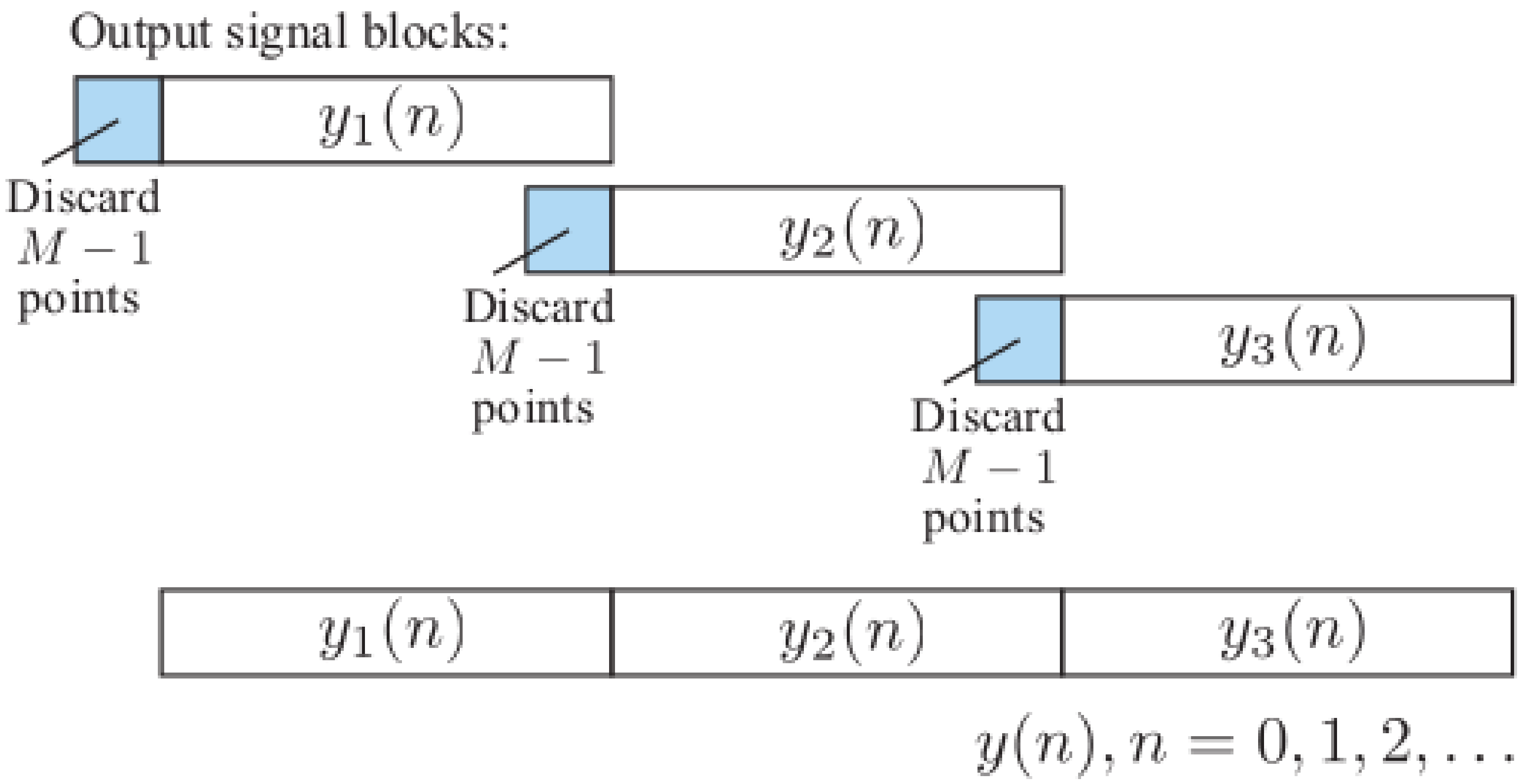
$$y_3(n) = \{\underbrace{y_3(0), \quad y_3(1), \quad \ldots \quad y_3(M-2)}_{M-1 \text{ points corrupted from aliasing}}, y(2L), \ldots, y(3L-1)\}$$

where $y(n) = x(n) * h(n)$ is the desired output.

The first $M-1$ points of each output block are discarded.

The remaining L points of each output block are appended to form $y(n)$.

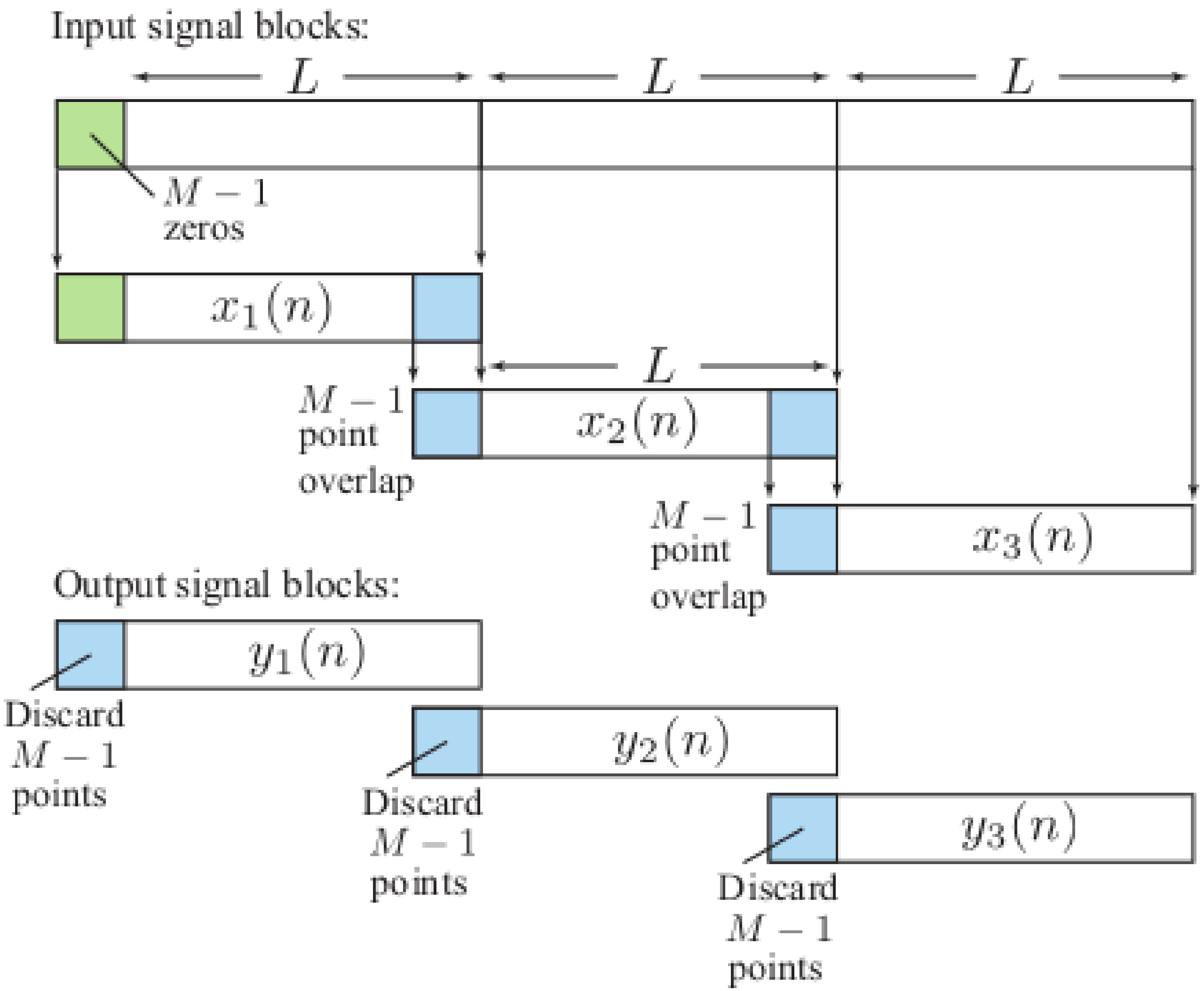
Overlap-Save Method



Overlap-Save Method

1. Insert $M-1$ zeros at the beginning of the input sequence $x(n)$.
2. Break the padded input signal into overlapping blocks $x_m(n)$ of length $N = L + M - 1$ where the overlap length is $M-1$.
3. Zero pad $h(n)$ to be of length $N = L + M - 1$.
4. Take N-DFT of $h(n)$ to give $H(k)$, $k = 0, 1, \ldots, N-1$.
5. For each block m:
   - 5.1 Take N-DFT of $x_m(n)$ to give $X_m(k)$, $k = 0, 1, \ldots, N-1$.
   - 5.2 Multiply: $Y_m(k) = X_m(k) \cdot H(k)$, $k = 0, 1, \ldots, N-1$.
   - 5.3 Take N-IDFT of $Y_m(k)$ to give $y_m(n)$, $n = 0, 1, \ldots, N-1$.
   - 5.4 Discard the first $M-1$ points of each output block $y_m(n)$.
6. Form $y(n)$ by appending the remaining (i.e., last) L samples of each block $y_m(n)$.

Overlap-Save Method



Signals, Linear Filtering methods based on the DFT

Overlap-Save Method

Signal x[n] (time domain):[3, -1, 0, 3, 2, 0, 1, 2, 1]

Filter h[n] (time domain): [1, -1, 1] M=3

If N=5

$N = L + M - 1 = L + 3 - 1 = 5$ Thus L=3

n	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11
x[n]	0	0	3	-1	0	3	2	0	1	2	1	0	0	0
x_0[n+2]	0	0	3	-1	0	0	0	0	0	0	0	0	0	0
x_1[n-1]	0	0	0	-1	0	3	2	0	0	0	0	0	0	0
x_2[n-4]	0	0	0	0	0	0	2	0	1	2	1	0	0	0
x_3[n-7]	0	0	0	0	0	0	0	0	0	2	1	0	0	0

Signals, Linear Filtering methods based on the DFT

Overlap-Save Method

Computing y_0[n] Using Method 1: Fourier Transform

x_0[n] = [0, 0, 3, -1, 0]

h[n] = [1, -1, 1, 0, 0]

y_0[n] = IDFT(DFT(x_0[n]) . DFT(h[n]))

X_0[n] = DFT(x_0[n]) = [2, -1.618 - 2.351i, 0.618 + 3.804i, 0.618 - 3.804i, -1.618 + 2.351i]

H[n] = DFT(h[n]) = [1, -0.118 + 0.363i, 2.118 + 1.539i, 2.118 - 1.539i, -0.118 - 0.363i]

H[n] . X_0[n] = [2, 1.045 - 0.31i, -4.545 + 9.009i, -4.545 - 9.009i, 1.045 + 0.31i]

y_0[n] = [-1, 0, 3, -4, 4]

Computing y_0[n] Using Method 2: Standard Convolution

x_0[n] = [0, 0, 3, -1, 0]

h[n] = [1, -1, 1, 0, 0]

$$y\_0[n] = \begin{vmatrix} 1 & 0 & 0 & 1 & -1 \\ -1 & 1 & 0 & 0 & 1 \\ 1 & -1 & 1 & 0 & 0 \\ 0 & 1 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 & 1 \end{vmatrix} . \begin{vmatrix} 0 \\ 0 \\ 3 \\ -1 \\ 0 \end{vmatrix} = \begin{vmatrix} -1 \\ 0 \\ 3 \\ -4 \\ 4 \end{vmatrix}$$

Signals, Linear Filtering methods based on the DFT

Overlap-Save Method

Computing y_1[n] Using Method 1: Fourier Transform

x_1[n] = [-1, 0, 3, 2, 0]

h[n] = [1, -1, 1, 0, 0]

y_1[n] = IDFT(DFT(x_1[n]) . DFT(h[n]))

X_1[n] = DFT(x_1[n]) = [4, -5.045 - 0.588i, 0.545 + 0.951i, 0.545 - 0.951i, -5.045 + 0.588i]

H[n] = DFT(h[n]) = [1, -0.118 + 0.363i, 2.118 + 1.539i, 2.118 - 1.539i, -0.118 - 0.363i]

H[n] . X_1[n] = [4, 0.809 - 1.763i, -0.309 + 2.853i, -0.309 - 2.853i, 0.809 + 1.763i]

y_1[n] = [1, 1, 2, -1, 1]

Computing y_1[n] Using Method 2: Standard Convolution

x_1[n] = [-1, 0, 3, 2, 0]

h[n] = [1, -1, 1, 0, 0]

$$
y\_1[n] = \begin{vmatrix} 1 & 0 & 0 & 1 & -1 \\ -1 & 1 & 0 & 0 & 1 \\ 1 & -1 & 1 & 0 & 0 \\ 0 & 1 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 & 1 \end{vmatrix} . \begin{vmatrix} -1 \\ 0 \\ 3 \\ 2 \\ 0 \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \\ 2 \\ -1 \\ 1 \end{vmatrix}
$$

Signals, Linear Filtering methods based on the DFT

Overlap-Save Method

Computing y_2[n] Using Method 1: Fourier Transform

x_2[n] = [2, 0, 1, 2, 1]

h[n] = [1, -1, 1, 0, 0]

y_2[n] = IDFT(DFT(x_2[n]) . DFT(h[n]))

X_2[n] = DFT(x_2[n]) = [6, -0.118 + 1.539i, 2.118 - 0.363i, 2.118 + 0.363i, -0.118 - 1.539i]

H[n] = DFT(h[n]) = [1, -0.118 + 0.363i, 2.118 + 1.539i, 2.118 - 1.539i, -0.118 - 0.363i]

H[n] . X_2[n] = [6, -0.545 - 0.225i, 5.045 + 2.49i, 5.045 - 2.49i, -0.545 + 0.225i]

y_2[n] = [3, -1, 3, 1, 0]

Computing y_2[n] Using Method 2: Standard Convolution

x_2[n] = [2, 0, 1, 2, 1]

h[n] = [1, -1, 1, 0, 0]

$$y\_2[n] = \begin{vmatrix} 1 & 0 & 0 & 1 & -1 \\ -1 & 1 & 0 & 0 & 1 \\ 1 & -1 & 1 & 0 & 0 \\ 0 & 1 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 & 1 \end{vmatrix} . \begin{vmatrix} 2 \\ 0 \\ 1 \\ 2 \\ 1 \end{vmatrix} = \begin{vmatrix} 3 \\ -1 \\ 3 \\ 1 \\ 0 \end{vmatrix}$$

Signals, Linear Filtering methods based on the DFT

Overlap-Save Method

Computing y_3[n] Using Method 1: Fourier Transform

x_3[n] = [2, 1, 0, 0, 0]

h[n] = [1, -1, 1, 0, 0]

y_3[n] = IDFT(DFT(x_3[n]) . DFT(h[n]))

X_3[n] = DFT(x_3[n]) = [3, 2.309 - 0.951i, 1.191 - 0.588i, 1.191 + 0.588i, 2.309 + 0.951i]

H[n] = DFT(h[n]) = [1, -0.118 + 0.363i, 2.118 + 1.539i, 2.118 - 1.539i, -0.118 - 0.363i]

H[n] . X_3[n] = [3, 0.073 + 0.951i, 3.427 + 0.588i, 3.427 - 0.588i, 0.073 - 0.951i]

y_3[n] = [2, -1, 1, 1, 0]

Computing y_3[n] Using Method 2: Standard Convolution

x_3[n] = [2, 1, 0, 0, 0]

h[n] = [1, -1, 1, 0, 0]

```
         |  1  0  0  1 -1 |   | 2 |   |  2 |
         | -1  1  0  0  1 |   | 1 |   | -1 |
y_3[n] = |  1 -1  1  0  0 | . | 0 | = |  1 |
         |  0  1 -1  1  0 |   | 0 |   |  1 |
         |  0  0  1 -1  1 |   | 0 |   |  0 |
```

Signals, Linear Filtering methods based on the DFT

Overlap-Save Method

n	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11
y_0[n]	~~-1~~ => 0	~~0~~ => 0	3	-4	4	0	0	0	0	0	0	0	0	0
y_1[n]	0	0	0	~~-1~~ => 0	~~-1~~ => 0	2	-1	1	0	0	0	0	0	0
y_2[n]	0	0	0	0	0	0	~~-3~~ => 0	~~-1~~ => 0	3	1	0	0	0	0
y_3[n]	0	0	0	0	0	0	0	0	0	~~-2~~ => 0	~~-1~~ => 0	1	1	0
y[n]	0	0	3	-4	4	2	-1	1	3	1	0	1	1	0

THANK YOU

Ms. Ashwini

Department of Electronics and Communication

ashwinib@pes.edu

+91 80 6666 3333
Ext 741