APPLICATIONS OF FFT: FILTERING OF LONG SEQUENCES

1

Prof. Alina Karabchevsky, www.alinakarabchevsky.com

Introduction to Signal Processing,

School of ECE,

Ben-Gurion University

Reading: Ch. 7.3 by Proakis and Monolakis; Ch. 8.7 by Oppenheim and Shafer

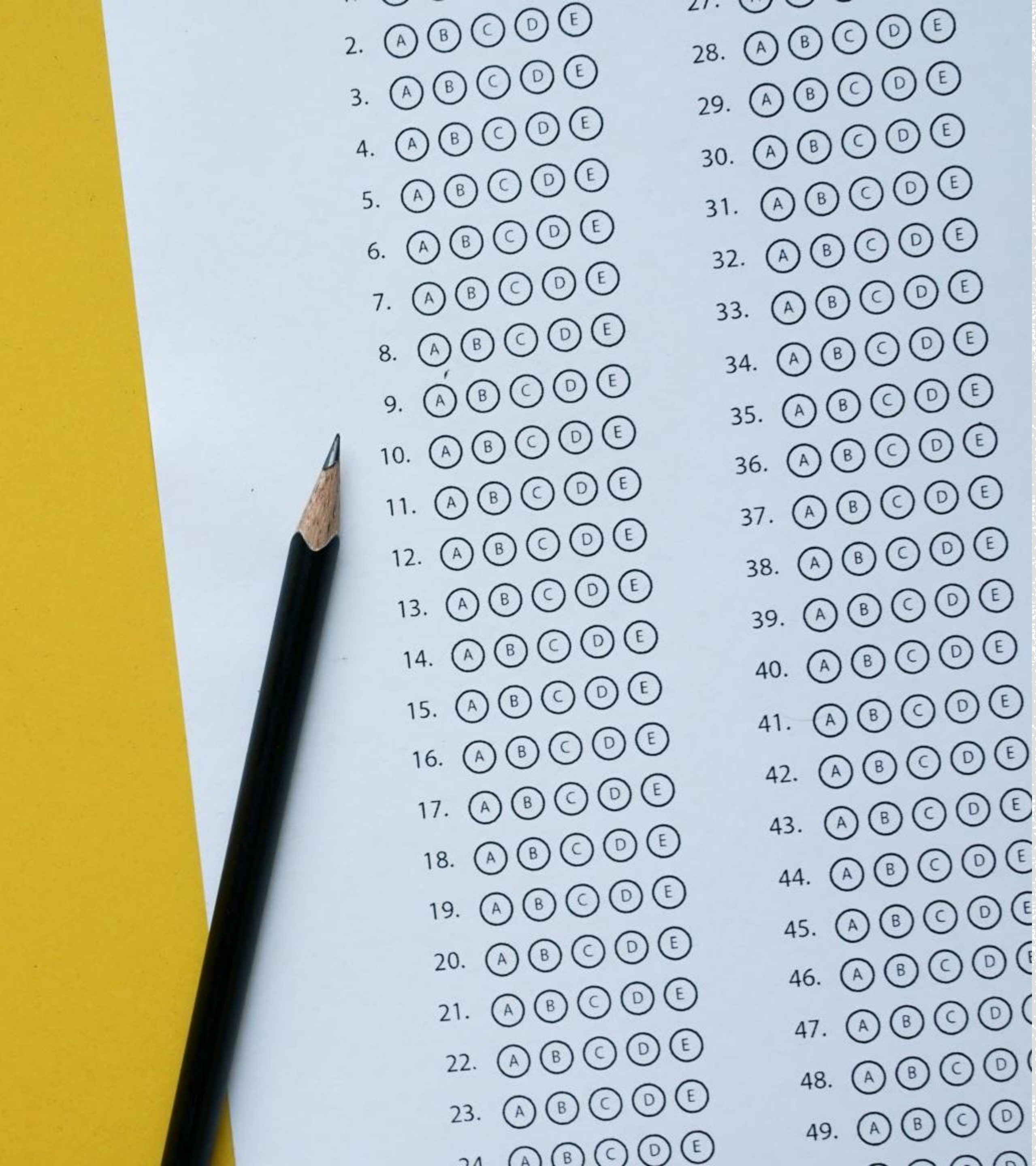
Streaming

OUTLINE

- Filtering Long duration sequences
- Overlap-Add and Overlap-Save method
- Approach: applying FFT on smaller sequences

INTRODUCTION

- An input sequence x[n] is of long duration (very long) or real-time series of unknown length (streaming) such as speech, telephone, TV, and is to be processed with a system having impulse response of finite duration by convolving two sequences. Because of the length of the input sequences, it's not practical to store it all before performing linear convolution.
- Therefore, input sequences are divided into blocks. Two methods commonly used for filtering the sectioned data and combining results are the:
- Overlap-save method
- Overlap-add method



FILTERING OF LONG SEQUENCES

- Let assume x[n] of length L and filter h[n] of length M. M<L
- What are the options to calculate y[n]?
- How many calculations are needed in each option?

- (I) Linear convolution in time
- (II) FFT and linear convolution after the zero-padding to the length of N=L+M-1
- (III) Segmentation of x[n] to blocks and calculation of convolution by performing FFT to each block.

(I) LINEAR CONVOLUTION IN TIME

h moves on x

$$y[n] = \sum_{m=0}^{M-1} h[m]x[n-m]\,, \qquad n = 0, \dots, M+L-2$$

Can be calculated in real time

Question: How many multiplications are needed to calculate $y[n]$?:

Answer: Length of $y[n]$ is L+M-1. For each $y[n]$ we need M multiplications, so in summary: (L+M-1)M

Real time: the calculation of each $y[n]$ needs to be before the next $x[n]$.

Assuming a suitable CPU, the delay will be of the calculation itself.

(II) FFT AND LINEAR CONVOLUTION AFTER THE ZERO-PADDING TO N=L+M-1

Assuming:

$x[n]$ length L

$h[n]$ length M

We zero-pad to the length of N=L+M-1

$$y[n] = \mathrm{IFFT}\{\underbrace{FFT\{x[n]\}}_{\text{Length N}}FFT\{h[n]\}\}$$

Number of multiplications: $3 \times \frac{N}{2}\log_2 N$

Disadvantages: 1) delay in output till receiving all L samples at the input. Can be very long and not suitable for many applications. 2) N can be large -> then FFT is too long.

Example: L=1024, M=1024

Direct computation: ~ $2\text{x}10^6$ computationally inefficient

FFT computation: N=2048-> $3 \times \frac{N}{2}\log_2 N = 3 \times \frac{2048}{2}\log_2 2048 = 33\text{x}10^3$

(III) SEGMENTATION OF X[N] TO BLOCKS

Segmentation of x[n] to blocks and calculation of convolution by performing FFT to each block, adding the results to obtain $y[n]$

- For this, we divide to blocks for enlarging the length of the FFT and the delay
- We will use FFT in each block for the computational efficiency

- There are two methods for this: overlap-save, overlap-add

OVERLAP-ADD METHOD

Let assume long $x[n]$, $h[n]$ length M

Calculate: $y[n] = h[n] * x[n]$

Divide $x[n]$ to blocks of length L

$$x_i[n] = \begin{cases} x[n], & L(i-1) \le n \le Li - 1 \\ 0, & \text{otherwise} \end{cases}$$

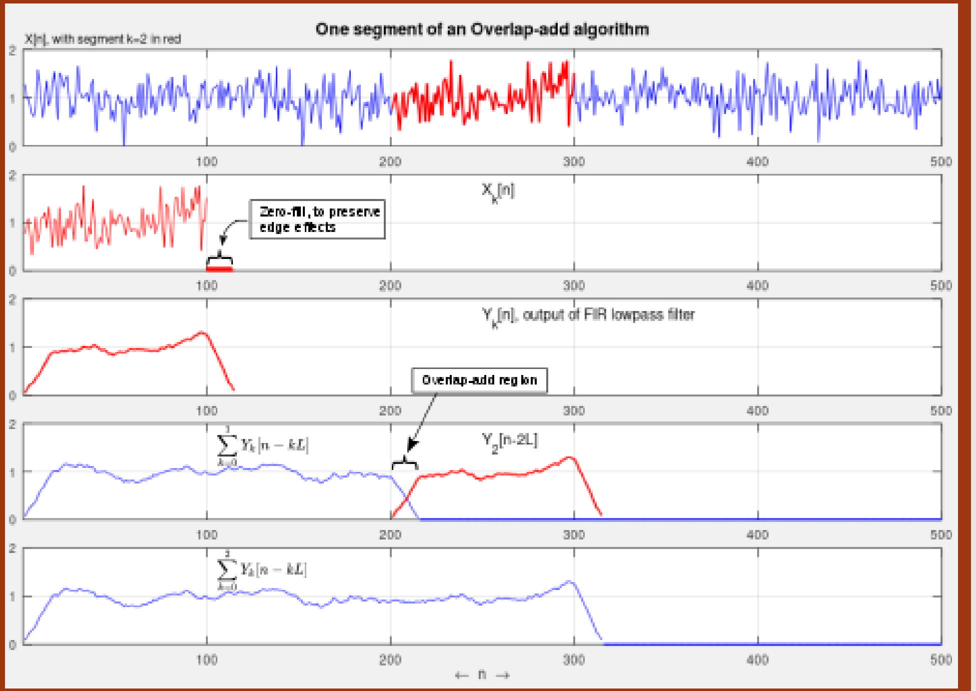
In a way that the location in time of each block is **preserved** and the signal beyond the block is nullified.

$$x[n] = \sum_i x_i[n]$$

Since the linear convolution is a linear operator, we obtain:

$$y[n] = x[n] * h[n] = \sum_i x_i[n] * h[n] = \sum_i y_i[n]$$

Therefore, $y[n]$ is given by the sum of the result of the convolution in each block.



OVERLAP-ADD METHOD

Calculation of each y_i will be performed by FFT after the zero-padding to the length of N=L+M-1

$$y_i[n] = \mathrm{IFFT}\{X_i[k] \cdot H[k]\}$$

Length N after the zero-padding

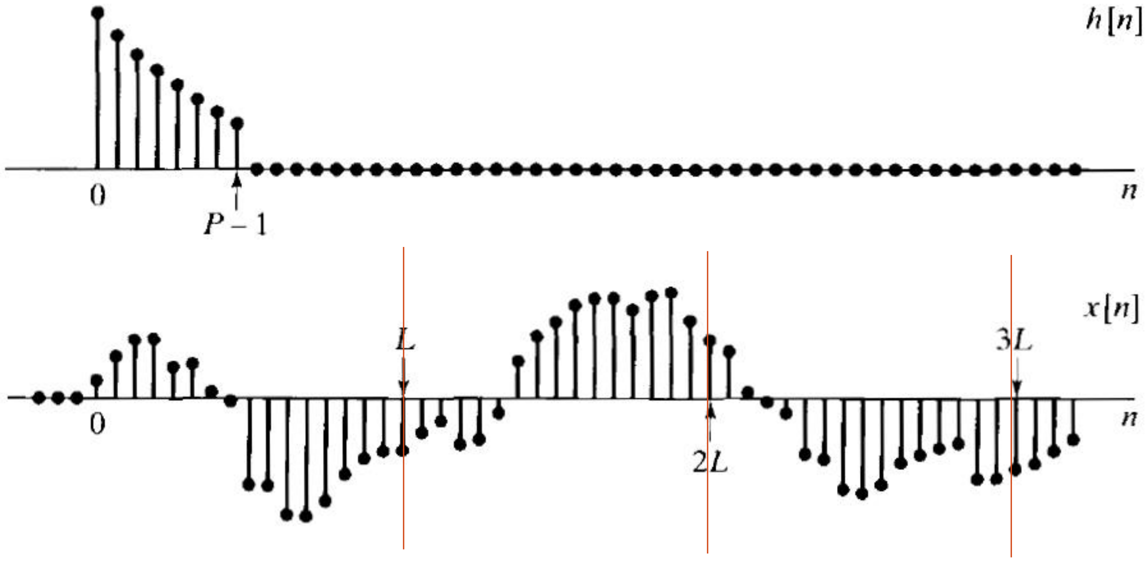
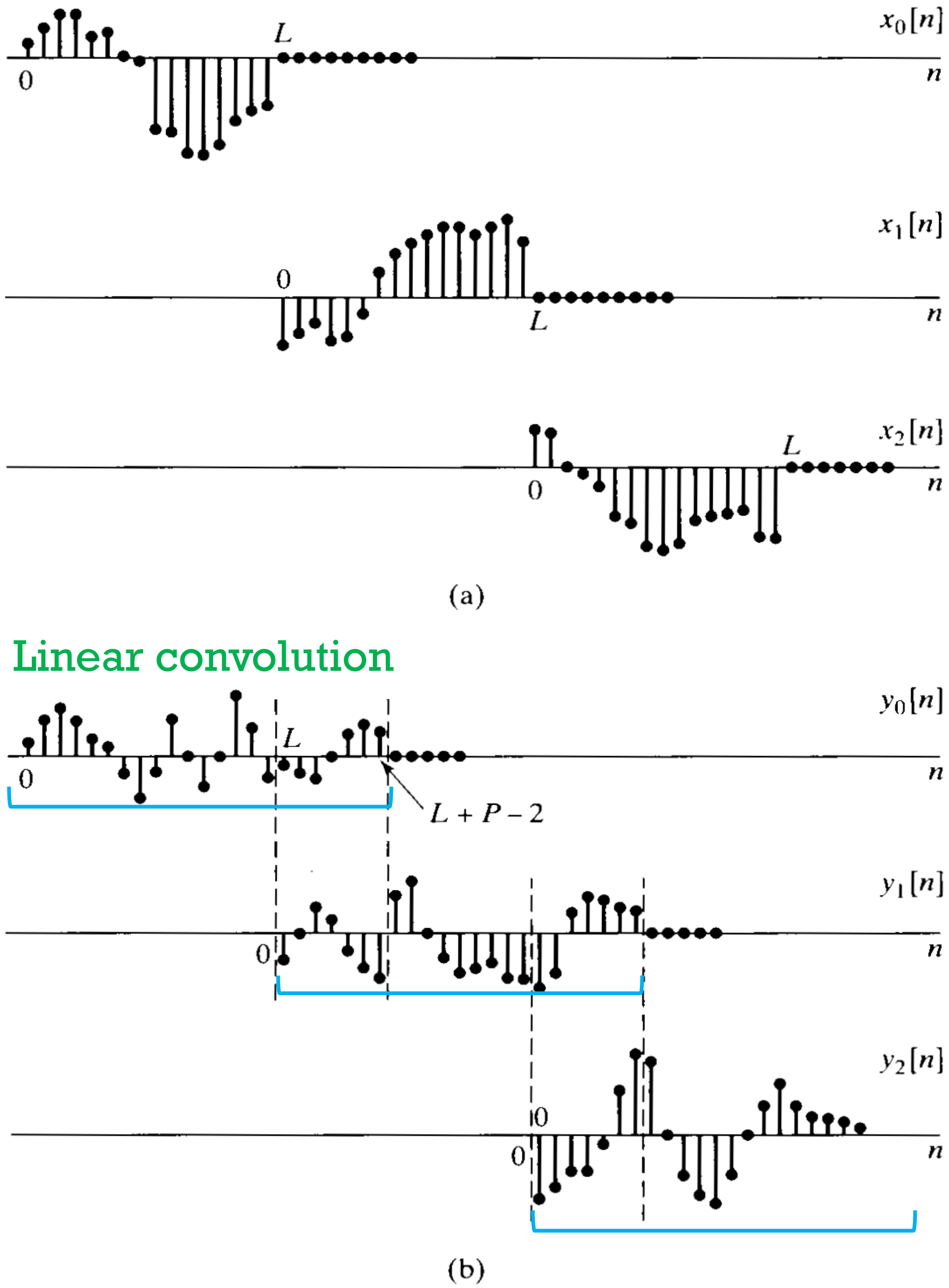


Figure 8.22 Finite-length impulse response $h[n]$ and indefinite-length signal $x[n]$ to be filtered.



OVERLAP-ADD METHOD

(a) Decomposition of x[n] into nonoverlapping sections of length L;

(b) Result of convolving each section each h[n].

LINEAR CONVOLUTION קונבולוציה לינארית

$x_1[n]=\{1,2,3,4\}$

$x_2[n]=\{2,1,1,3\}$

$N=4+4-1=7$

$x_1[n]$

Linear convolution

$x_2[n-m]$

			1	2	3	4			
3	1	1	2						
	3	1	1	2					
		3	1	1	2				
			3	1	1	2			
				3	1	1	2		
					3	1	1	2	
						3	1	1	2

y[0]=2

y[1]=5

y[2]=9

y[3]=16

y[4]=13

y[5]=13

y[6]=12

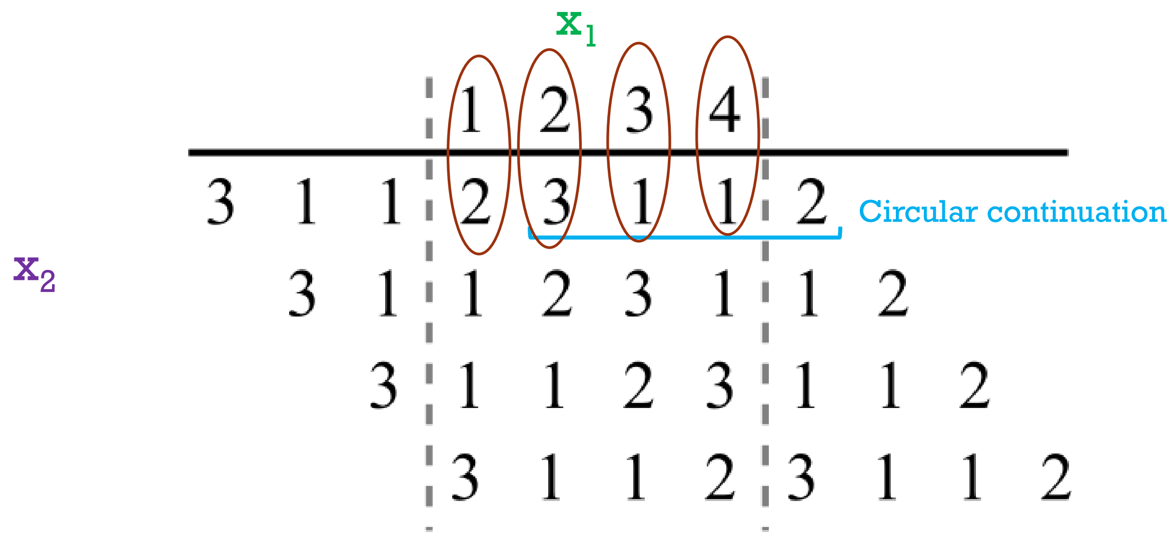
CIRCULAR CONVOLUTION קונבולוציה מעגלית

$x_1[n]=\{1,2,3,4\}$

$x_2[n]=\{2,1,1,3\}$

modulo 4 -> periodic in 4

Circular convolution



y[0]=15 ‣ 1*2+2*3+3*1+4*1=15

y[1]=18

y[2]=21

y[3]=16

OVERLAP-ADD METHOD: EXAMPLE

$x_1[n]=\{1,2,3,4\}$

$x_2[n]=\{2,1,1,3\}$

Linear convolution via circular convolution with zero padding

			x_1							
			1	2	3	4				
3	1	1	2							
	3	1	1	2						
		3	1	1	2					
			3	1	1	2				
				3	1	1	2			
					3	1	1	2		
						3	1	1	2	
							3	1	1	2

x_2

y[0]=2

y[1]=5

y[2]=9

y[3]=16

y[4]=13

y[5]=13

y[6]=12

y[7]=0

LINEAR CONVOLUTION VIA CIRCULAR CONVOLUTION WITH ZERO PADDING

$x_1[n]=\{1,2,3,4\}$

$x_2[n]=\{2,1,1,3\}$

$N_1+N_2-1=4+4-1=7$
modulo 7 -> periodic in 7

Linear convolution via circular convolution with zero padding

$x_1[n]$

							1	2	3	4	0	0	0				
	0	0	0	3	1	1	2	0	0	0	3	1	1	2	0	0	0	y[0]=2
	2	0	0	0	3	1	1	2	0	0	0	3	1	1	2	0	0	y[1]=5
	1	2	0	0	0	3	1	1	2	0	0	0	3	1	1	2	0	y[2]=9
x_2	1	1	2	0	0	0	3	1	1	2	0	0	0	3	1	1	2	y[3]=16
	3	1	1	2	0	0	0	3	1	1	2	0	0	0	3	1	1	y[4]=13
	0	3	1	1	2	0	0	0	3	1	1	2	0	0	0	3	1	y[5]=13
	0	0	3	1	1	2	0	0	0	3	1	1	2	0	0	0	3	y[6]=12
	0	0	0	3	1	1	2	0	0	0	3	1	1	2	0	0	0	2

n=8 the same as n=0 due to the periodicity

OVERLAP-ADD METHOD: EXAMPLE

$x[n] = \{1\ 2\ 3\ 4\ 1\ 2\ 2\ 1\ 1\ 3\ 1\ 2 \ldots\}$ L=4

$h[n] = \{1\ 1\ 3\}$ M=3

N=L+M-1=6

We take the second and third blocks :

$x_1[n]=\{1,2,2,1\}$

$x_2[n]=\{1,3,1,2\}$

Circular convolution on each block after the zero padding

	0	0	1	2	2	1	0	0	1	2	$x_1[n]$
	3	1	1	0	0	0	3	1	1	0	$h[n]$
	0	3	1	1	0	0	0	3	1	1	
	0	0	3	1	1	0	0	0	3	1	
	0	0	0	3	1	1	0	0	0	3	
	1	0	0	0	3	1	1	0	0	0	
	1	1	0	0	0	3	1	1	0	0	
…	7	3	1	3	7	9	7	3	1	3	… $y_1[n]$

0	0	1	3	1	2	0	0	1	3	$x_2[n]$
3	1	1	0	0	0	3	1	1	0	$h[n]$
0	3	1	1	0	0	0	3	1	1	
0	0	3	1	1	0	0	0	3	1	
0	0	0	3	1	1	0	0	0	3	
1	0	0	0	3	1	1	0	0	0	
1	1	0	0	0	3	1	1	0	0	
5	6	1	4	7	12	5	6	1	4	

$y[n]=y_1[n]+y_2[n]+\ldots$ 1 3 7 9 8 7 7 12 5 6

y_0+ precisely $+y_3$

OVERLAP-ADD METHOD: SUMMARY

- We have received the linear convolution in each segment
- At the edges, we had to perform the convolution with the rest of the signal x[n] but we performed the convolution with zero. Therefore, we will need to add the results of the convolution in third block till the correct overlap takes place.
- L-M+1 correct samples in each segment.

SURVEY: OVERLAP-ADD METHOD

- EasyPolls:

An engineer is instructed to implement an overlap-add algorithm with signal sections length L=8, filter length M=1017, and FFT length N=1024. Can the algorithm work?

- ○ Yes, N=L+M-1
- ○ No, L << M, none of the outputs will be correct
- ○ Not in general, but will work if the filter includes sufficient zeros

See results   vote

OVERLAP-SAVE METHOD

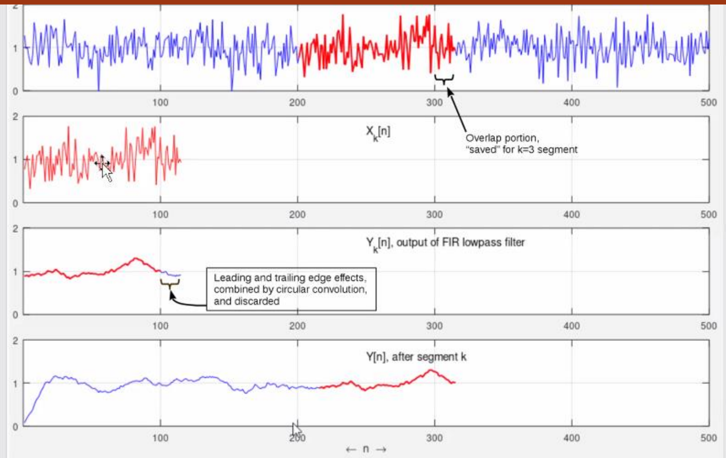


Fig 1: A sequence of 4 plots depicts one cycle of the overlap–save convolution algorithm. The 1st plot is a long sequence of data to be processed with a lowpass FIR filter. The 2nd plot is one segment of the data to be processed in piecewise fashion. The 3rd plot is the filtered segment, with the usable portion colored red. The 4th plot shows the filtered segment appended to the output stream.[A] The FIR filter is a boxcar lowpass with M=16 samples, the length of the segments is L=100 samples and the overlap is 15 samples.

- To avoid adding sequences of one block and adjacent blocks, we will be utilizing the similar method to overlap-add method.
- x$[n]$ is divided to blocks with length L, but FFT is not performed by zero-padding but rather by adding zeros at the end of the sequence $N=L+M-1$
- For block i:
- $X_i[k] = FFT\{[x[m]\ ;\ x[\text{n}]]\}$

  M-1 point L point

- $L(i-1) \le n \le Li - 1$ like in overlap-add method
- $L(i-1)-(M-1) \le m \le L(i-1)-1$ $M-1$ previous samples to block i
- $H[k] = \{[h[m]\ ;\ 0\ ;\ 0 \ldots 0]\}$

  M point L-1 zeros

- and then $y_i[n] = \text{IFFT}\,\{X_i[k] \cdot H[k]\}$

OVERLAP-SAVE METHOD EXAMPLE

- $x[n]=\{1\ 2\ 3\ 4\ 1\ 2\ 2\ 1\ 1\ 3\ 1\ 2 .....\}$
- $h[n]=\{1\ 1\ 3\}$
-
- We choose $x_1[n]=\{1\ 2\ 2\ 1\}$
- $x_2[n]=\{1\ 3\ 1\ 2\}$
- Circular convolution on each block after the zero-padding
- More correct values
- No summations of overlap-add

L=4

M=3

N=L+M-1=6

2	1	3	4	1	2	2	1	3	4	
3	1	1	0	0	0	3	1	1	0	
0	3	1	1	0	0	0	3	1	1	
0	0	3	1	1	0	0	0	3	1	
0	0	0	3	1	1	0	0	0	3	
1	0	0	0	3	1	1	0	0	0	
1	1	0	0	0	3	1	1	0	0	
7	9	10	10	14	15	7	9	10	10	$y_1[n]$
		x								

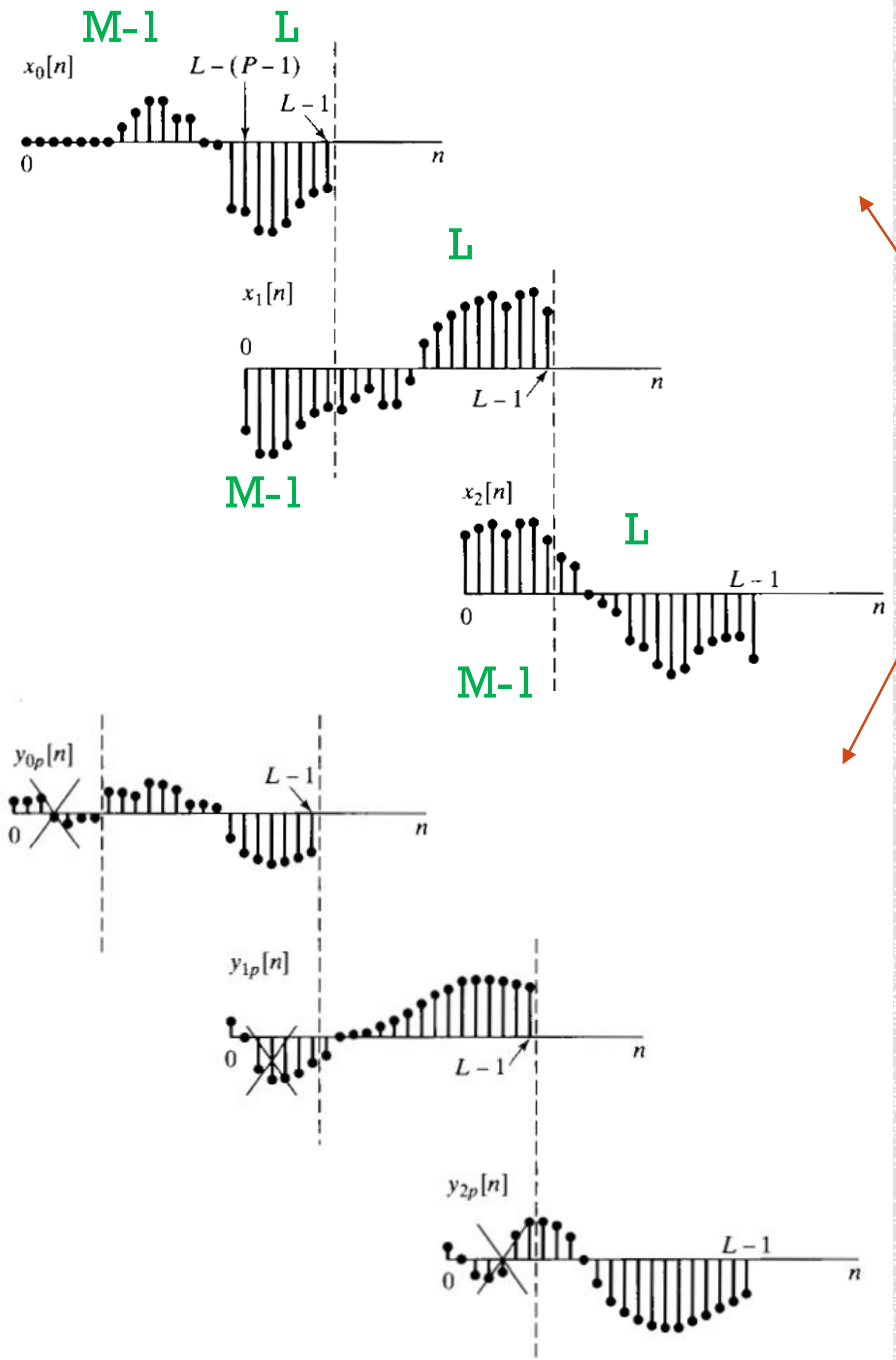
	1	2	2	1	1	3	1	2	2	1
	3	1	1	0	0	0	3	1	1	0
	0	3	1	1	0	0	0	3	1	1
	0	0	3	1	1	0	0	0	3	1
	0	0	0	3	1	1	0	0	0	3
	1	0	0	0	3	1	1	0	0	0
	1	1	0	0	0	3	1	1	0	0
$y_2[n]$	7	12	7	9	8	7	7	12	7	9
			x							

$y[n]$	x	x	14	15	7	9	8	7	7	12	x	x

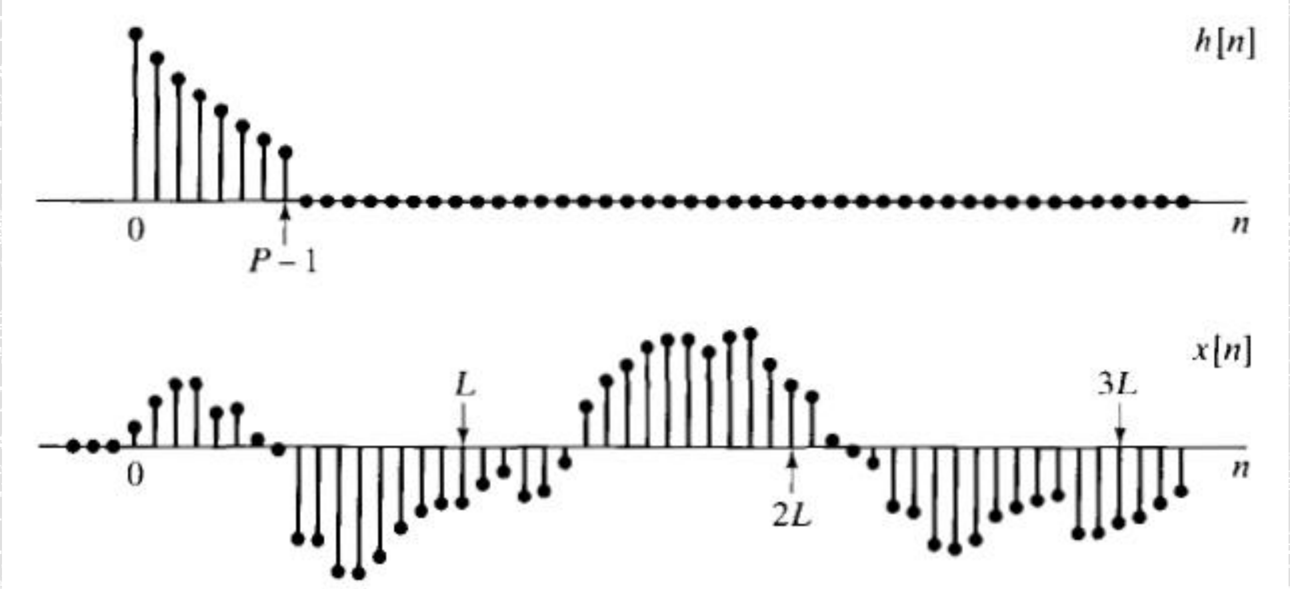
OVERLAP-SAVE METHOD: SUMMARY

- We noticed that $y[1]$, $y[0]$, and in general first (M-1) samples are not part of the linear convolution since they includes elements of the circular convolution.
- $y[2]$- $y[5]$ include the linear convolution
- Therefore, we can take last L samples and throw M-1 first samples
- For each new block with length of L at the input (x[n]), we will receive new block with length of L at the output (y[n]) without need of additional operations.
- Therefore, overlap-save method is preferable over overlap-add method.

OVERLAP-SAVE METHOD



- Top: decomposition of x[n] from the previous example (and see below) into overlapping sections of Length L.
- Bottom: Result of convolving each section with h[n]. The portions of each filtered section to be discarded in forming the linear convolution are indicated.



For *L* input samples we will receive in each block L output samples without a need of adding them -> therefore the method has an advantage over the overlap-add method.

OVERLAP-SAVE METHOD: NUMBER OF OPERATIONS

- The length of FFT is $N=L+M-1$
- Number of operations for each L elements at the output is:
- $\sim \underbrace{2 \times \frac{N}{2} \log_2 N}_{\substack{\text{FFT on x.} \\ \text{IFFT on \{HX\}.} \\ \text{Ignore FFT on h} \\ \text{since it is only once}}} + \underbrace{N}_{\substack{\text{Multiplication} \\ \text{Of HX}}}$

  FFT on x.
  IFFT on {HX}.
  Ignore FFT on h
  since it is only once

  Multiplication
  Of HX

- As compared to LxM in **direct computation**.
- When N is large, the computation efficiency will improve.
- For **overlap-add**, number of calculations is similar, but additional (M-1) operations are added in each block.

OVERLAP-SAVE METHOD: EXAMPLES

(i) L=9

M=8

N=L+M-1=16

Overlap-save: $2*8*\log_2 16+16=80$

Direct calculation: 9*8=72

(ii) L=1024

M=1024

N=L+M-1≅2048

Overlap-save: $2* 1024 *\log_2 2048+ 2048=24{,}576$

Direct calculation: LxM=1,048,576

Here overlap-save more efficient ~43 times

H.W.: check if fftfilt() command in Mtlaab is implemented via overlap-add or overlap-save mothd?

FILTERING LONG SEQUENCES

OVERLAP-ADD, OVERLAP-SAVE METHODS

FILTERS

SURVEY: OVERLAP-SAVE METHOD

- EasyPolls:

A signal x of length 320 samples is linearly-convolved with h of length 32 using the overlap-save method, using FFT of length 64. How many FFT/IFFT needs to be computed overall?

- 20
- 21
- 23
- 25

results vote

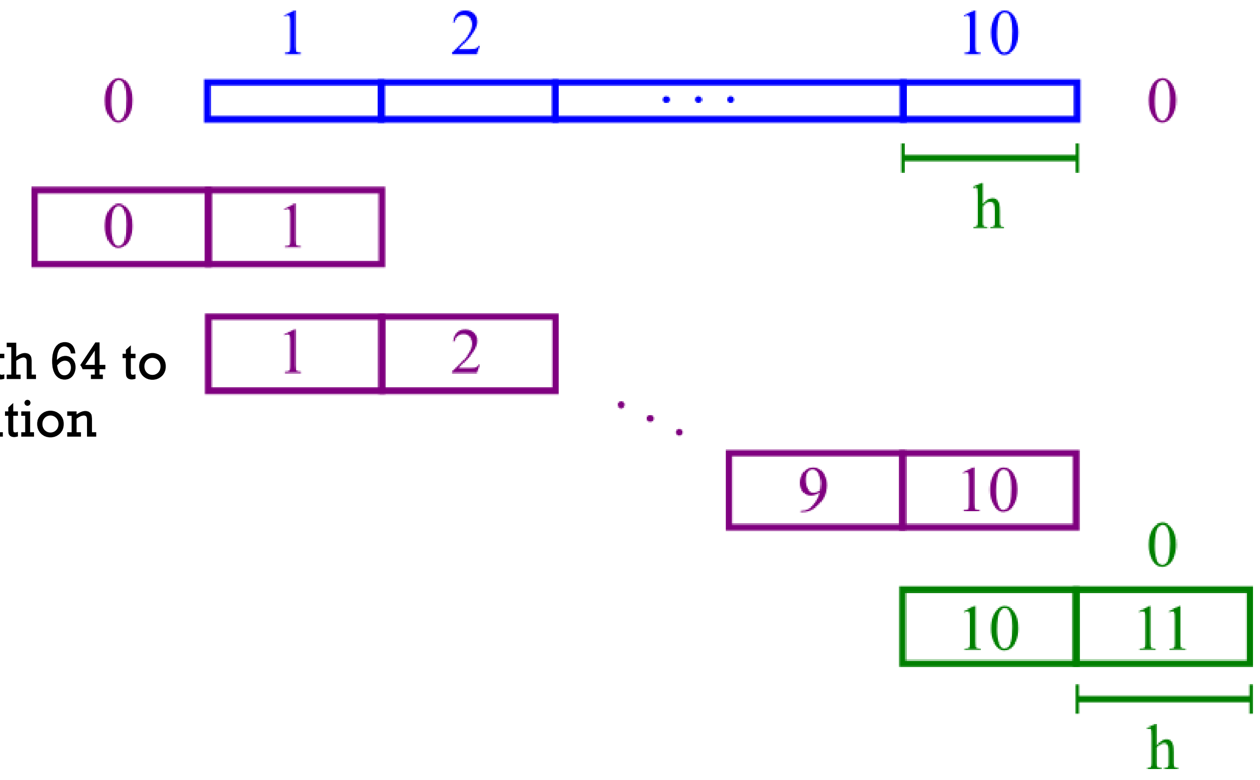
SURVEY: OVERLAP-SAVE METHOD- SOLUTION

L=32
M=32
N=63

- 11 blocks FFT{x}
- 11 blocks IFFT{y}
- 1 FFT on h
- Needed 11 blocks length 64 to calculate linear convolution including edges

x y h

- 11+11+1=23 right
- 10+10+1=21 wrong
- 10+10=20 wrong



OVERLAP-SAVE METHOD: SPEECH SPECTROGRAM

- The processing is made specifically on different sequences of different information

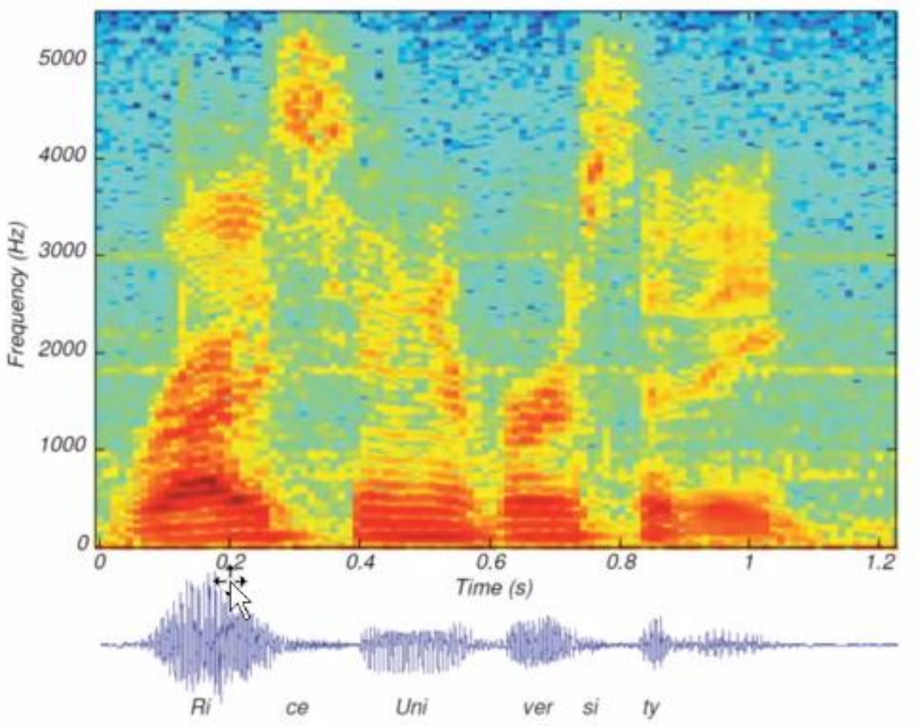


Figure 4.17: Displayed is the spectrogram of the author saying "Rice University." Blue indicates low energy portion of the spectrum, with red indicating the most energetic portions. Below the spectrogram is the time-domain speech signal, where the periodicities can be seen.

OVERLAP-SAVE METHOD: SPEECH SPECTROGRAM

- The separation to sequences of length L is performed via rectangular window filter which in turn add additional noise.