Discrete - Time Signals and Systems

FIR Filters-I

Yogananda Isukapalli
Since a discrete-time signal is a sequence of numbers, the operator $\tau$ can be described by a mathematical formula. It is just a computational process.
An Evolution of computing machines

For the price of a small house, you could have one of these
TMS32010, 1983: First PC plug-in board from Atlanta Signal Proc
Digital Cell Phone (ca. 2000)
Example 1: A Simple Discrete System

\[ x[n] \xrightarrow{\tau\{x[n]\}} y[n] = |x[n]| \]

Output is the absolute value of input

Fig. 8.6
Fig. 8.7
Example 2: 3-point averaging method

Consider an input sequence $x[n]$

Take the average of any three points in a sequence

- $\frac{2 + 4 + 6}{3} = 4$
- $\frac{4 + 6 + 4}{3} = \frac{14}{3}$
- $\frac{6 + 4 + 2}{3} = 4$
- $\frac{4 + 2 + 0}{3} = 2$

$x[0] = 2 \quad x[3] = 4$
$x[1] = 4 \quad x[4] = 2$
$x[2] = 6 \quad x[5] = 0$
Notation for the output is arbitrary, following the notation shown below leads to the output:

\[ y[0] = \frac{x[0] + x[1] + x[2]}{3} \quad y[1] = \frac{x[1] + x[2] + x[3]}{3} \]

<table>
<thead>
<tr>
<th>( n )</th>
<th>( n &lt; -2 )</th>
<th>(-2)</th>
<th>(-1)</th>
<th>(0)</th>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
<th>(5)</th>
<th>( n &gt; 5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x[n] )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( y[n] )</td>
<td>0</td>
<td>( \frac{2}{3} )</td>
<td>2</td>
<td>4</td>
<td>( \frac{14}{3} )</td>
<td>4</td>
<td>2</td>
<td>( \frac{2}{3} )</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ n=0 \quad y[0] = \frac{1}{3} (x[0] + x[1] + x[2]) \]
\[ n=1 \quad y[1] = \frac{1}{3} (x[1] + x[2] + x[3]) \]

Fig.8.9
The above process (3-point average) generalizes to an important input-output equation known as a difference equation:

\[ y[n] = \frac{1}{3} (x[n] + x[n+1] + x[n+2]) \]

The above equation describes a very important class of discrete-time systems called ‘FIR filters’.
However, the equation $y[n] = \frac{1}{3} (x[n] + x[n + 1] + x[n + 2])$ doesn’t seem practical as we need two future samples to calculate present output.

depends only on past values, a causal system
Difference equations

Recursive equation

\[ y[n] = \sum_{l=1}^{N} a_l y[n-l] + \sum_{k=0}^{M} b_k x[n-k], \quad \text{Recursive equation} \]

Represents a class of filters known as IIR filters

Non-Recursive part

\[ y[n] = \sum_{k=0}^{M} b_k x[n-k], \quad \text{Non-Recursive part} \]

FIR filters
First order discrete-time system

\[ y[n] = a_1 y[n-1] + b_0 x[n] \]

a Recursive equation

Solution:

\[ n = 1 \quad y[1] = a_1 y[0] + b_0 x[1] \]
\[ n = 2 \quad y[2] = a_1 y[1] + b_0 x[2] \]
\[ = a_1 \left( a_1 y[0] + b_0 x[1] \right) + b_0 x[2] \]
\[ = a_1^2 y[0] + a_1 b_0 x[1] + b_0 x[2] \]
\[ y[3] = a_1^3 y[0] + a_1^2 b_0 x[1] + a_1 b_0 x[2] + b_0 x[3] \]

Generalizing the 1\(^{st}\) order discrete-time system, \(y[r]\) becomes:

\[ y[r] = a_1^r y[0] + b_0 \left[ a_1^{r-1} x[1] + a_1^{r-2} x[2] + \cdots + a_1^0 x[r] \right] \]

For a causal system, \(y[0] = a_1 y[-1] + b_0 x[0] = b_0 x[0]\)

\[ y_c[r] = a_1^r b_0 x[0] + b_0 \sum_{m=1}^{r} a_1^{r-m} x[m] \]

\[ y_c[r] = b_0 \sum_{m=0}^{r} a_1^{r-m} x[m] \]
Example

Let \( x[n] = 1 \) for all \( n \), \( n \geq 0 \)

\[
y[r] = a_1^r y[0] + b_0 \left[ a_1^{r-1} x[1] + a_1^{r-2} x[2] + \ldots + a_1^0 x[r] \right]
\]

\[
= a_1^r y[0] + b_0 \left[ a_1^{r-1} + a_1^{r-2} + \ldots + 1 \right]
\]

\[
y[r] = a_1^r y[0] + b_0 \frac{a_1^r - 1}{a_1 - 1} \quad a_1 \neq 1
\]
\[ y[r] = y[0] + b_0 r \quad a_1 = 1 \]

\[ \lim_{a_1 \to 1} \frac{a_1^r - 1}{a_1 - 1} = \lim_{a_1 \to 1} \frac{d}{da_1} \left( a_1^r - 1 \right) = \lim_{a_1 \to 1} \frac{d}{da_1} (a_1 - 1) = r \]

\[ = \lim_{a_1 \to 1} \frac{r a_1^{r-1}}{1} = r 1^{r-1} = r \]

Let \( a_1 = 1/2, \ b_0 = 2, \ y[0] = 3 \)

\[ y[n] = (1/2)^n 3 + 2 \left( \frac{(1/2)^n - 1}{(1/2) - 1} \right) = (1/2)^n 3 - 4 \left[ (1/2)^n - 1 \right] = 4 - (1/2)^n \quad n \geq 0 \]
The General FIR Filter

Non-Recursive part of difference equation represents a general FIR filter

\[ y[n] = \sum_{k=0}^{M} b_k x[n - k], \]

The above equation doesn’t involve any past samples, so the system is a causal one. The moving average problem discussed earlier is an FIR filter.
Filter Order = $M$: No. of memory blocks required in the filter implementation

Filter Length, $L = M + 1$: Total No. of samples required in calculating the output, $M$ from memory (past) and one present sample

Filter coefficients $\{b_k\}$: Completely define an FIR filter. All the properties of the filter can be understood through the coefficients
Graphical view of a general FIR filter

\[ y[n] = \sum_{k=0}^{M} b_k x[n - k] \]

Fig. 8.13

Mth order Causal filter

Running onto the signal

Weighted sum of M+1 points

Running off the signal

Zero output

Present: \( \ell = n \)
Block Diagrams: An Implementation view of FIR filters

Building blocks required

\[ y[n] = \beta x[n] \]

\[ y[n] = x_1[n] + x_2[n] \]

\[ y[n] = x[n-1] \]
direct form of block diagram

Difference equation

\[ y[n] = \left( \left( b_0 x[n] + b_1 x[n-1] \right) + b_2 x[n-2] \right) + b_3 x[n-3] \]
transposed form of block diagram

\[
\begin{align*}
    y[n] &= b_0 x[n] + v_1[n - 1] \\
    v_1[n] &= b_1 x[n] + v_2[n - 1] \\
    v_2[n] &= b_2 x[n] + v_3[n - 1] \\
    v_3[n] &= b_3 x[n]
\end{align*}
\]

\[
y[n] = \left( \left( (b_0 x[n] + b_1 x[n - 1]) + b_2 x[n - 2] \right) + b_3 x[n - 3] \right)
\]

Same Result!!!
Example: FIR filter application

Averaging of a sequence with different filter lengths

\[ x[n] = (1.02)^n + \cos\left(\frac{2\pi n}{8} + \frac{\pi}{4}\right) \quad \text{for } 0 \leq n \leq 40 \]

Signal  \hspace{1cm} \text{Noise, cosine part}

Fig. 8.17
$$y_3[n] = \sum_{k=0}^{2} \left( \frac{1}{3} \right) x[n - k]$$

Notice that the output is smoother or its noise level is low than input.

Output goes till n=42

1st 2 samples

Output of 3-Point Running-Average Filter

Fig.8.18
Notice that the output is much smoother than 3-point averaging method, noise level is low.
Example: DJIA signal

similar approach, averaging

\[ y[n] = \left( \frac{1}{51} \right) \sum_{k=0}^{50} x[n-k] \]
Compensating for delay,

\[
\tilde{y}[n] = \left(\frac{1}{51}\right) \sum_{k=-25}^{25} x[n-k]
\]

In these two examples, FIR filters are shown to remove rapid fluctuations.
Discrete-Time Unit Impulse Sequence

Unit Impulse is the simplest sequence with only one nonzero value at \( n=0 \)

\[ \delta[n] = \begin{cases} 
1 & n = 0 \\
0 & n \neq 0 
\end{cases} \]

\( \delta[n] \) is known as Kronecker delta function
**Tabular form and a shifted version of unit impulse**

<table>
<thead>
<tr>
<th>$n$</th>
<th>...</th>
<th>$-2$</th>
<th>$-1$</th>
<th>$0$</th>
<th>$1$</th>
<th>$2$</th>
<th>$3$</th>
<th>$4$</th>
<th>$5$</th>
<th>$6$</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta[n]$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$\delta[n-3]$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**$\delta[n]$ is NON-ZERO When its argument is equal to ZERO**

![Fig. 8.24](image.png)

**Shifted impulse sequence, $\delta[n-3]$.**

$n = 3$
Unit Impulse Response Sequence

The response of an FIR filter to a unit impulse sequence is called as unit impulse response or simply ‘impulse response’
General FIR equation

\[ y[n] = \sum_{k=0}^{M} b_k x[n - k] \]

Impulse Response

\[ x[n] = \delta[n], \quad y[n] = h[n] \]

\[ h[n] = \sum_{k=0}^{M} b_k \delta[n - k] = \begin{cases} b_n & n = 0, 1, 2, \ldots, M \\ 0 & \text{otherwise} \end{cases} \]

The sum evaluates to a single term for each value of \( n \), as \( \delta[n - k] \) is nonzero only when \( n = k \)
### Tabular form for ‘Impulse Response’ equation

<table>
<thead>
<tr>
<th>$n$</th>
<th>$n &lt; 0$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>$\ldots$</th>
<th>$M$</th>
<th>$M+1$</th>
<th>$n &gt; M + 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x[n] = \delta[n]$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$y[n] = h[n]$</td>
<td>0</td>
<td>$b_0$</td>
<td>$b_1$</td>
<td>$b_2$</td>
<td>$b_3$</td>
<td>$\ldots$</td>
<td>$b_M$</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

In the above table $h[n] = 0$ for $n < 0$ and $n > M$, The length of impulse response sequence is finite. This is why the system is called a **Finite Impulse Response (FIR) system**

Fig.8.26
**Example 1: 3-point average filter**

\[ y_3[n] = \left( \frac{1}{3} \right) \sum_{k=0}^{2} x[n-k] \] *compare with standard equation*

\[ y[n] = \sum_{k=0}^{M} b_k x[n-k], \quad h[0] = b_0, \quad h[1] = b_1, \quad h[2] = b_2 \]

\[ \therefore h[0] = \frac{1}{3}, \quad h[1] = \frac{1}{3}, \quad h[2] = \frac{1}{3} \]

Fig. 8.27
Example 2

Find the difference equation governing the input–output relation with FIR filter coefficients \{3, -1, 2, 1\}

FIR filter coefficients \( h[n] = \{3, -1, 2, 1\} \)

\( h[n] = b_k \) for \( k = 0, 1 \ldots N \)

\( y[n] = \sum_{k=0}^{M} b_k x[n - k] \)

\( = \sum_{k=0}^{3} b_k x[n - k] \)

\( = b_0 x[n] + b_1 x[n - 1] + b_2 x[n - 2] + b_3 x[n - 3] \)

\( = 3x[n] - x[n - 1] + 2x[n - 2] + x[n - 3] \) difference equation
Representation of a general sequence $x[n]$

Any sequence can be obtained by adding shifted impulses

$$x[n] = 2\delta[n] + 4\delta[n - 1] + 6\delta[n - 2] + 4\delta[n - 3] + 2\delta[n - 4]$$
### Tabular form: Breaking a sequence into shifted impulses

<table>
<thead>
<tr>
<th>(n)</th>
<th>(\ldots)</th>
<th>(-2)</th>
<th>(-1)</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>(\ldots)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(2\delta[n])</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(4\delta[n - 1])</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(6\delta[n - 2])</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(4\delta[n - 3])</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(2\delta[n - 4])</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(x[n])</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\[
x[n] = \sum_k x[k]\delta[n - k]
\]

\[
= \ldots + x[-1]\delta[n + 1] + x[0]\delta[n] + x[1]\delta[n - 1] + \ldots
\]

For any signal

Fig.8.29
Discrete-Time Convolution Sum

General Discrete-System

\[
\begin{align*}
  y[n] &= T\{x[n]\} \\
  T\{\cdot\} &= \text{Discrete-Time System}
\end{align*}
\]

Fig. 8.30
From the previous figure,
\[ x[0] \delta[n] = x[0]h[n] \]
\[ x[0] \delta[n-1] = x[0]h[n-1] \]
\[ x[0] \delta[n-k] = x[0]h[n-k] \]

As shown previously using superposition,
\[ x[n] = \sum_{k=-\infty}^{\infty} x[k] \delta[n-k] \]
\[ y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k], \text{ Convolution Sum} \]
**Example 1: FIR from Convolution**

\[ y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k], \quad \text{If } h[n] \text{ is non-zero} \]

only in the interval \( 0 \leq n \leq M \) then,

\[ y[n] = \sum_{k=n-M}^{n} x[k]h[n-k], \]

which is a classic FIR filter

About limits: \( 0 \leq (n-k) \leq M \)

\[ \therefore (n-M) \leq k \leq n \]
Example 2: Computing the output

\[ x[n] = \{2, 4, 6, 4, 2\}, \quad h[n] = \{3, -1, 2, 1\} \]

Convolve \( x[n] \) and \( h[n] \) to get \( y[n] \)

- Write out the signals \( x[n] \) and \( y[n] \) on separate rows
- The output is to be computed as sum of shifted rows
- Each shifted row is to be produced by multiplying the \( x[n] \) row by one of the \( h[k] \) values and,
- By shifting the result to the right so that it lines up with \( h[k] \) position
Numerical convolution done through the above process is also called as synthetic polynomial multiplication

Tabular form describing the convolution

<table>
<thead>
<tr>
<th></th>
<th>$n &lt; 0$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>$n &gt; 7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x[n]$</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$h[n]$</td>
<td>0</td>
<td>3</td>
<td>-1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$h[0]x[n]$</td>
<td>0</td>
<td>6</td>
<td>12</td>
<td>18</td>
<td>12</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$h[1]x[n-1]$</td>
<td>0</td>
<td>0</td>
<td>-2</td>
<td>-4</td>
<td>-6</td>
<td>-4</td>
<td>-2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$h[2]x[n-2]$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>8</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$h[3]x[n-3]$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$y[n]$</td>
<td>0</td>
<td>6</td>
<td>10</td>
<td>18</td>
<td>16</td>
<td>18</td>
<td>12</td>
<td>8</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>
Try the demo on your CD

Signal
Flipped Signal

Multiplication

Linear Convolution

Signal Axis:
- $o = x[k]$
- $o = h[n-k]$

Multiplication Axis:
- $x[k]h[n-k]$

Convolution Axis:
- $y[n] = \sum x[k]h[n-k]$
Reference