Lab.4 &5. Introduction to FIR Filters

Draft Handouts

4.1. Definition

In signal processing, a finite impulse response (FIR) filter is a filter whose impulse response (or response to any finite length input) is of *finite* duration, because it settles to zero in finite time. This is in contrast to infinite impulse response (IIR) filters, which may have internal feedback and may continue to respond indefinitely (usually decaying).

The impulse response of an Nth-order discrete-time FIR filter (i.e., with a Kronecker delta impulse input) lasts for N + 1 samples, and then settles to zero.

FIR filters can be discrete-time or continuous-time, and digital or analog.

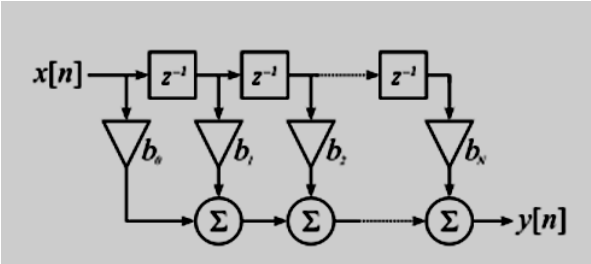

Fig. FIR filter block diagram

A discrete-time FIR filter of order *N*. The top part is an *N*-stage delay line with *N* + 1 taps. Each *unit delay* is a z^{-1} operator in the Z-transform notation. The output *y* of a linear time invariant system is determined by convolving its input signal *x* with its impulse response *b*. For a discrete-time FIR filter, the output is a weighted sum of the current and a finite number of previous values of the input. The operation is described by the following equation, which defines the output sequence *y[n]* in terms of its input sequence *x[n]*:

$$\begin{aligned} y[n] &= b_0 x[n] + b_1 x[n-1] + \cdots + b_N x[n-N] \\ &= \sum_{i=0}^{N} b_i x[n-i] \end{aligned}$$

where:

- $x[n]$ is the input signal,
- $y[n]$ is the output signal,

- b_i are the *filter coefficients*, also known as *tap weights*, that make up the impulse response,
- N is the filter order; an Nth-order filter has $(N+1)$ terms on the right-hand side. The $x[n-i]$ in these terms are commonly referred to as *taps*, based on the structure of a tapped delay line that in many implementations or block diagrams provides the delayed inputs to the multiplication operations. One may speak of a *5th order/6-tap filter*, for instance.

4.2. FIR Properties

An FIR filter has a number of useful properties which sometimes make it preferable to an infinite impulse response (IIR) filter. FIR filters:

- ***Require no feedback.*** This means that any rounding errors are not compounded by summed iterations. The same relative error occurs in each calculation. This also makes implementation simpler.
- ***Inherent stability.*** This is due to the fact that, because there is no required feedback, all the poles are located at the origin and thus are located within the unit circle (the required condition for stability in a Z transformed system).
- ***Phase Issue:*** can easily be designed to be ***linear phase*** by making the coefficient sequence symmetric; linear phase, or phase change proportional to frequency, corresponds to equal delay at all frequencies. This property is sometimes desired for ***phase-sensitive applications***, for example data communications, crossover filters, and mastering.

The main disadvantage of FIR filters is that ***considerably more computation power*** in a general purpose processor is required compared to an IIR filter with similar sharpness or selectivity, especially when low frequency (relative to the sample rate) cutoffs are needed. However many digital signal processors provide specialized hardware features to make FIR filters approximately as efficient as IIR for many applications.

4.3. FIR Impulse response

The impulse response $h[n]$ can be calculated if we set $x[n] = \delta[n]$ in the above relation, where $\delta[n]$ is the Kronecker delta impulse. The impulse response for an FIR filter then becomes the set of coefficients b_n, as follows

$$h[n] = \sum_{i=0}^{N} b_i \delta[n-i] = b_n$$

The Z-transform of the impulse response yields the transfer function of the FIR filter

$$\begin{aligned} H(z) &= Z\{h[n]\} \\ &= \sum_{n=-\infty}^{\infty} h[n] z^{-n} \\ &= \sum_{n=0}^{N} b_n \, z^{-n} \end{aligned}$$

FIR filters are clearly *bounded-input bounded-output* (BIBO) stable, since the output is a sum of a finite number of finite multiples of the input values, so can be no greater than $\sum |b_i|$ times the largest value appearing in the input.

4.4. Filter design

To design a filter means to select the coefficients such that the system has specific characteristics. The required characteristics are stated in filter specifications. Most of the time filter specifications refer to the frequency response of the filter. There are different methods to find the coefficients from frequency specifications:

1. Window design method
2. Frequency Sampling method
3. Weighted least squares design
4. Parks-McClellan method (also known as the Equiripple, Optimal, or Minimax method). The Remez exchange algorithm is commonly used to find an optimal equiripple set of coefficients. Here the user specifies a desired frequency response, a weighting function for errors from this response, and a filter order *N*. The algorithm then finds the set of $(N+1)$coefficients that minimize the maximum deviation from the ideal. Intuitively, this finds the filter that is as close as you can get to the desired response given that you can use only $(N+1)$coefficients. This method is particularly easy in practice since at least one text[1] includes a program that takes the desired filter and *N*, and returns the optimum coefficients.
5. Equiripple FIR filters can be designed using the FFT algorithms as well[2]. The algorithm is iterative in nature. You simply compute the DFT of an initial filter design

that you have using the FFT algorithm (if you don't have an initial estimate you can start with h[n]=delta[n]). In the Fourier domain or FFT domain you correct the frequency response according to your desired specs and compute the inverse FFT. In time-domain you retain only N of the coefficients (force the other coefficients to zero). Compute the FFT once again. Correct the frequency response according to specs.

Software packages like MATLAB, GNU Octave, Scilab, and SciPy provide convenient ways to apply these different methods.

Some filter specifications refer to the time-domain shape of the input signal the filter is expected to "recognize". The optimum matched filter for separating any waveform from white noise is obtained by sampling that shape and using those samples in reverse order as the coefficients of the filter — giving the filter an impulse response that is the time-reverse of the expected input signal.

4.5. Window design method

In the Window Design Method, one designs *an ideal IIR filter*, then applies a *window function* to it – in the time domain, multiplying the infinite impulse by the window function. This results in the frequency response of the IIR being convolved with the frequency response of the window function. If the ideal response is sufficiently simple, such as rectangular, the result of the convolution can be relatively easy to determine. In fact one usually specifies the desired result first and works backward to determine the appropriate window function parameter(s). Kaiser windows are particularly well-suited for this method because of their closed form specifications.

Moving average example

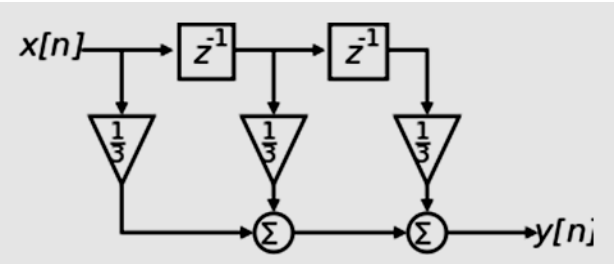

Fig. Block diagram of a simple FIR filter
(2nd-order/3-tap filter in this case, implementing a moving average)

A moving average filter is a very simple FIR filter. It is sometimes called a boxcar filter, especially when followed by decimation. The filter coefficients, $b_0, \ldots, b_N$, are found via the following equation:

$$b_i - \frac{1}{N \mid 1}$$

To provide a more specific example, we select the filter order $N = 2$. The impulse response of the resulting filter is:

$$h[n] = \frac{1}{3}\delta[n] - \frac{1}{3}\delta[n-1] + \frac{1}{3}\delta[n-2]$$

In Fig. above, the block diagram of a 2nd-order moving-average filter is shown. The frequency response $H(z)$ using the z-transform is to be:

$$H(z) = \frac{1}{3} + \frac{1}{3}z^{-1} + \frac{1}{3}z^{-2} = \frac{1}{3}\frac{z^2 \quad z \mid 1}{z^2}$$

Fig. below shows the pole-zero diagram of the filter. Zero frequency (DC) corresponds to (1,0), positive frequencies advancing counterclockwise around the circle to (-1,0) at half the sample frequency. Two poles are located at the origin, and two zeros are located at: $z_1 - -\frac{1}{2} + j\frac{\sqrt{3}}{2}$ and $z_2 - -\frac{1}{2} - j\frac{\sqrt{3}}{2}$. The frequency response, for frequency ω in radians per sample, is:

$$H\left(e^{j\omega}\right) = \frac{1}{3} + \frac{1}{3}e^{-j\omega} + \frac{1}{3}e^{-j2\omega}$$

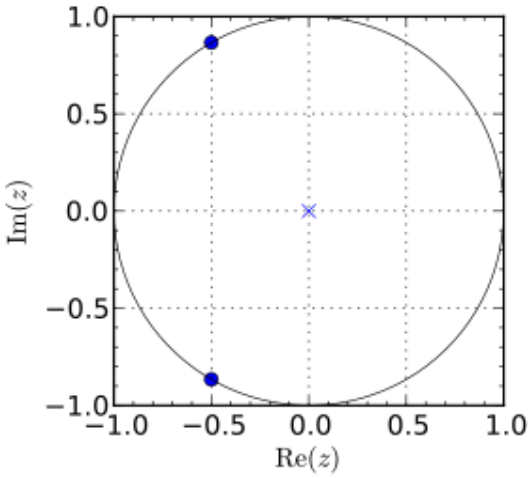

Fig. Pole-Zero Diagram

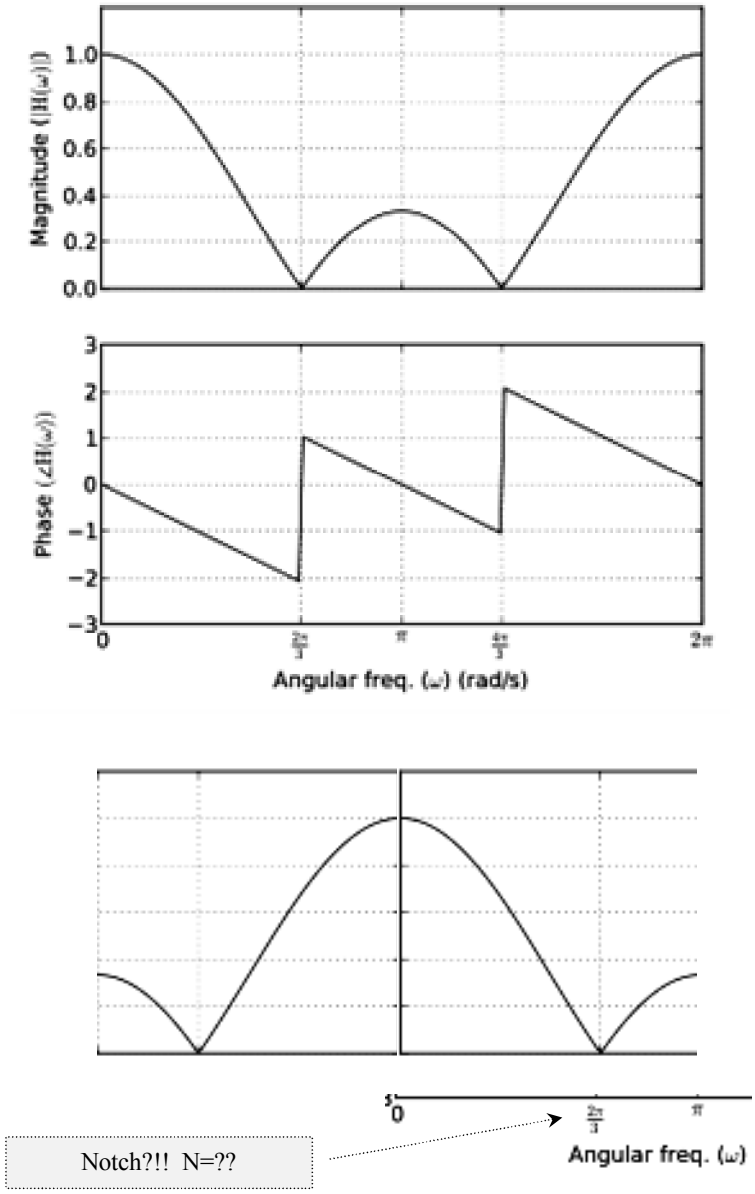

Fig. Amplitude and phase responses

In Fig. above, the magnitude and phase plots of the frequency response are depicted. Clearly, the moving-average filter passes low frequencies with a gain near 1, and attenuates high frequencies. This is a typical low-pass filter characteristic. Frequencies above π are aliases of the frequencies below π, and are generally ignored or filtered out if reconstructing a continuous-time signal. The following figure shows the phase response. Since the phase always follows a straight line except where it has been reduced modulo π radians (should be 2π), the linear phase property is demonstrated.

4.6. How to practically sketch the spectrum?!

1- ***Direct method:*** use a signal generator and an oscilloscope or spectrum analyzer to measure its gain at different individual frequencies. By using this method, it is straightforward to identify the distinct notches in the magnitude frequency response. Then, construct a table to assess the magnitude frequency response of the filter

2- ***Indirect method:*** to use wideband noise as an input signal. A pseudorandom binary sequence (PRBS) is generated and applied to the filter. The filtered noise can be viewed on a spectrum analyzer and whereas the frequency content of the PRBS input is uniform across all frequencies, the frequency content of the filtered noise will reflect the frequency response of the filter.

4.7. Application: Audi Scrambler

The scrambling method used is commonly referred to as frequency inversion. It takes an audio range, in this case 300 Hz to 3 kHz, and " folds " it about a 3.3 – kHz carrier signal. The frequency inversion is achieved by multiplying (modulating) the audio input by a carrier signal, causing a shift in the frequency spectrum with upper and lower sidebands. In the lower sideband that represents the audible speech range, the low tones are high tones, and vice versa. Fig. below demonstrates the block diagram of the scrambling scheme. At point A we have an input signal, band limited to 3 kHz. At point B we have a double - sideband signal with suppressed carrier. At point C the upper sideband and the section of the lower sideband between 3 and 3.3 kHz are filtered out.

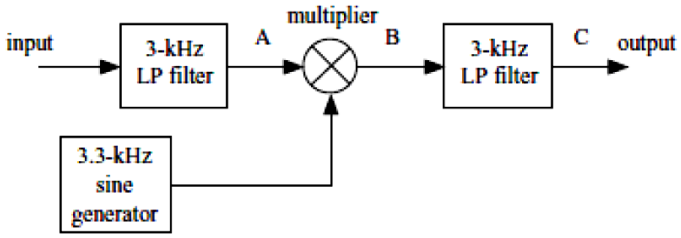

Fig. block diagram of the scrambling scheme