Introduction to Digital Filters

Yogananda Isukapalli

Digital Filters

- Introduction

- Types of filters
  1. FIR filters
  2. IIR filters

- Choosing between FIR and IIR filters

- Filter design steps

Introduction

- A digital filter is a mathematical algorithm implemented in hardware and/or software that operates on a digital input signal to produce a digital output signal for the purpose of achieving a filtering objective.

- *Digital filters operate on digitized analog signals or just numbers, representing some variable, stored in a computer memory.*

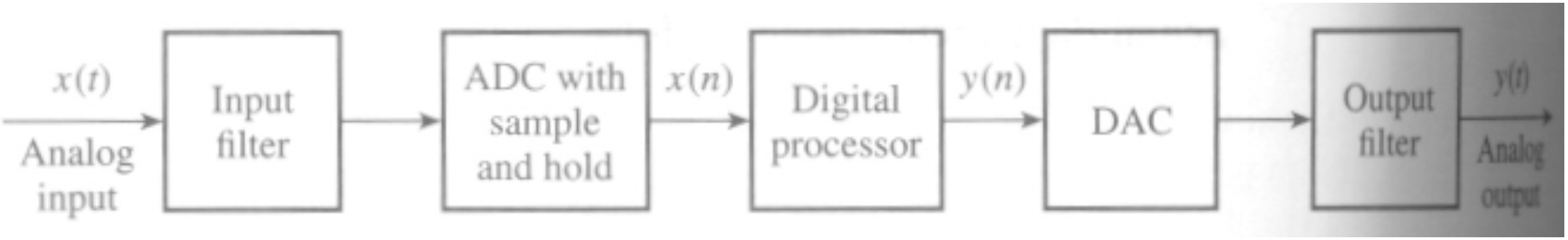


Fig 1: A simplified block diagram of a real-time digital filter with analog input and output signals[1]

- Digital filters are used in applications like data compression, biomedical signal processing, speech and image processing, data transmission, digital audio, telephone echo cancellation etc.

- Some advantages of digital filters over analog filters

  1. Truly linear phase response.
  2. Performance does not vary with environmental changes, for example thermal variations.
  3. The frequency response can be automatically adjusted if the filter is implemented using a programmable processor.
  4. Digital filters can be used at very low frequencies.

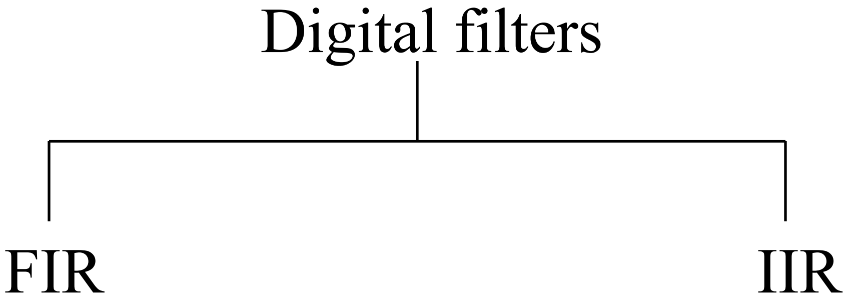
- Main disadvantages of digital filters over analog filters

  1. *Speed limitation:* The maximum bandwidth of signals that digital filters can handle, in real time, is much lower than for analog signals.

  2. *Finite wordlength effects:* Digital filters are subject to noises like ADC noise and roundoff noise that could lead to instability.

  3. *Long design and development times:* The design and development times for digital filters, especially hardware development, can be much longer than for analog filters.

Types of filters


Digital filters

FIR

IIR

- A digital filter in general can be represented by its impulse response, h(k) (k=0,1,..) as in the following figure:

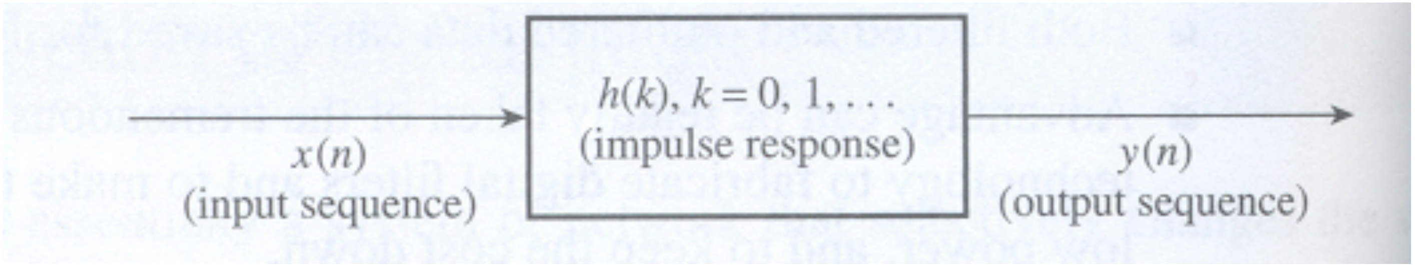


Fig 2: A conceptual representation of a digital filter[1]

- The input and output signals to the filter are related by the convolution sum:

$$y(n)=\sum_{k=0}^{\infty}h(k)x(n-k) \qquad \text{IIR} \qquad (1)$$

$$y(n)=\sum_{k=0}^{N-1}h(k)x(n-k) \qquad \text{FIR} \qquad (2)$$

- Because of the infinite length of the impulse response of the IIR filter, the IIR filtering equation can be expressed in a recursive form as:

$$y(n)=\sum_{k=0}^{\infty}h(k)x(n-k)=\sum_{k=0}^{N}b_k x(n-k)-\sum_{k=1}^{M}a_k y(n-k) \qquad (3)$$

where a_k and b_k are the coefficients of the filter.

- We note that in eqn (3), the current output sample, y(n), is a function of past outputs as well as present and past input samples, *that is the IIR is a feedback system of some sort*.

- In the FIR equation, the current output sample, y(n), is a function only of past and present values of the input.

- The transfer functions of FIR and IIR filters:

$$H(z) = \sum_{k=0}^{N-1} h(k) z^{-k} \qquad \text{FIR} \qquad (4)$$

$$H(z) = \frac{\sum_{k=0}^{N} b_k z^{-k}}{1 + \sum_{k=1}^{M} a_k z^{-k}} \qquad \text{IIR} \qquad (5)$$

Choosing between FIR and IIR filters

	FIR filters	IIR filters
1.	Can have exactly linear phase response, i.e. do not distort the phase of the signal.	The phase responses are nonlinear, especially at the band edges.
2.	FIR filters realized non-recursively are always stable.	The stability of IIR filters cannot be guaranteed.
3.	The effects of using limited number of bits to implement filters are less severe.	The effects like roundoff noise and coefficient quantization errors are more severe.
4.	Require more coefficients for sharp cutoff filters.	Less coefficients and thus less processing time and storage.
5.	No analog counterpart, but, easier to synthesize filters with arbitrary frequency responses.	Analog filters can be readily transformed into equivalent IIR digital filters.
6.	Algebraically more difficult to synthesize if CAD support is not available.	Less difficult to synthesize.

- Thus, a broad guideline on when to use FIR or IIR would be as follows:

  - **Use IIR** when the only important requirements are *sharp cutoff filters* and *high throughput*, as IIR filters will give fewer coefficients than FIR.

  - **Use FIR** if the number of *filter coefficients is not too large* and, in particular, if *little or no phase distortion* is desired.

Example 1)

The following transfer functions represent two different filters meeting identica amplitude–frequency response specifications:

(1) $$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

where

$b_0 = 0.498\ 181\ 9$

$b_1 = 0.927\ 477\ 7$

$b_2 = 0.498\ 181\ 9$

$a_1 = -0.674\ 487\ 8$

$a_2 = -0.363\ 348\ 2$

(2) $$H(z) = \sum_{k=0}^{11} h(k) z^{-k}$$

where

$h(0) = \ \ 0.546\ 032\ 80 \times 10^{-2} = h(11)$

$h(1) = -0.450\ 687\ 50 \times 10^{-1} = h(10)$

$h(2) = \ \ 0.691\ 694\ 20 \times 10^{-1} = h(9)$

$h(3) = -0.553\ 843\ 70 \times 10^{-1} = h(8)$

$h(4) = -0.634\ 284\ 10 \times 10^{-1} = h(7)$

$h(5) = \ \ 0.578\ 924\ 00 \times 10^{0} = h(6)$

For each filter,

(a) state whether it is an FIR or IIR filter,

(b) represent the filtering operation in a block diagram form and write down the difference equation, and

(c) determine and comment on the computational and storage requirements.

Solution)

(a) Filters (1) and (2) are IIR and FIR respectively.

(b) The block diagram for filter (1) is given in Figure 3(a). The corresponding set of difference equations is

$$w(n) = x(n) - a_1 w(n-1) - a_2 w(n-2)$$

$$y(n) = b_0 w(n) + b_1 w(n-1) + b_2 w(n-2)$$

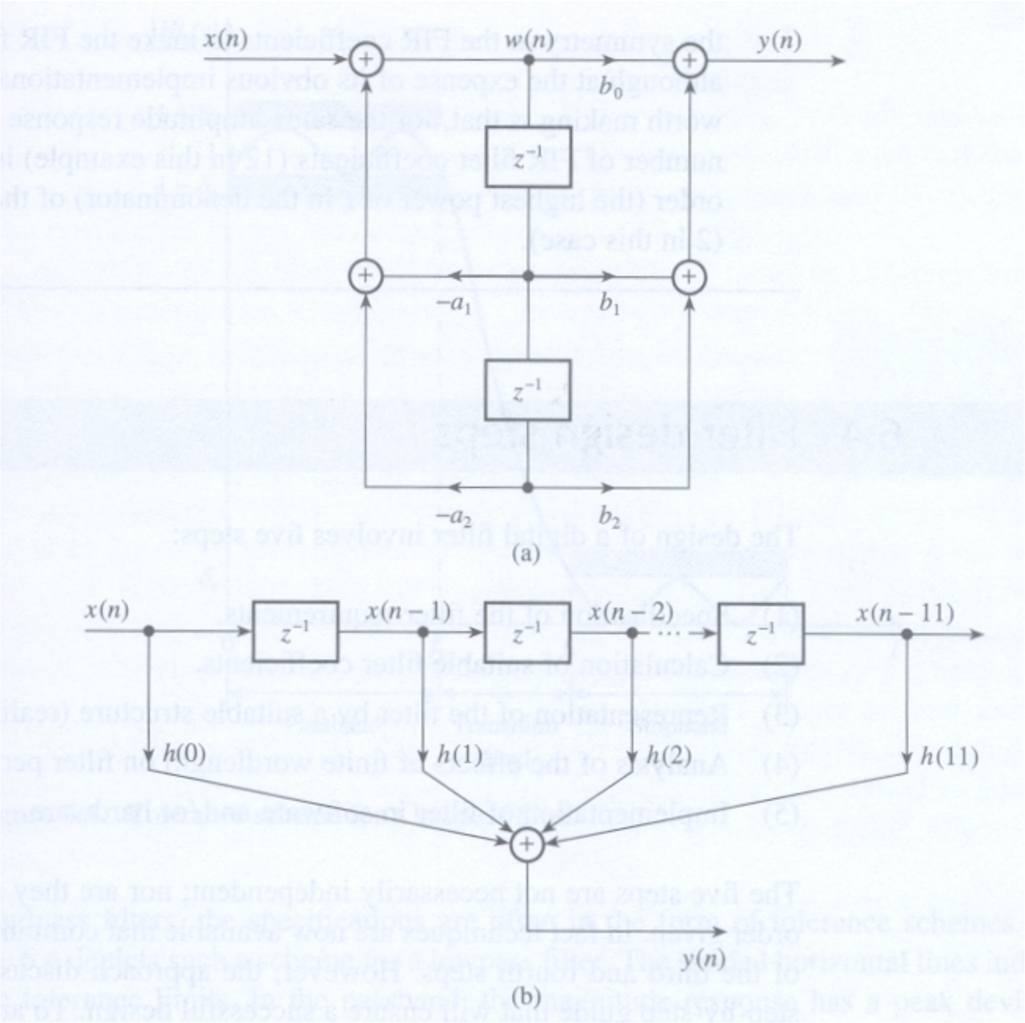
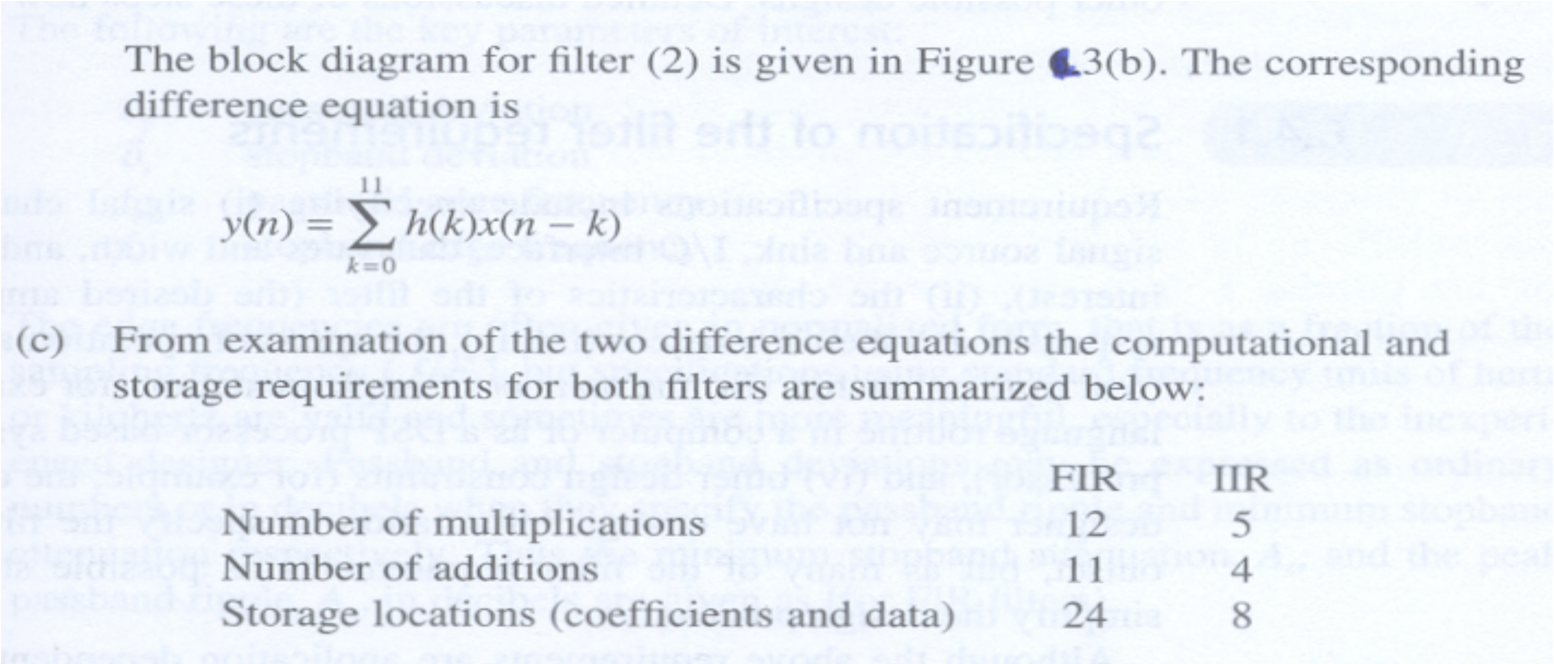


Fig 3: (a) Block diagram representation of the IIR filter of example 1. (b) Block diagram representation of the FIR filter of example 1.[1]



The block diagram for filter (2) is given in Figure 6.3(b). The corresponding difference equation is

$$y(n) = \sum_{k=0}^{11} h(k)x(n-k)$$

(c) From examination of the two difference equations the computational and storage requirements for both filters are summarized below:

	FIR	IIR
Number of multiplications	12	5
Number of additions	11	4
Storage locations (coefficients and data)	24	8

- It is evident that the IIR filter is more economical in both computational and storage requirements than the FIR filter for the above example.

Filter design steps

The design of a digital filter involves five steps:

1. Specification of the filter requirements.
2. Calculation of suitable filter coefficients.
3. Representation of the filter by a suitable structure (realization).
4. Analysis of the effects of finite wordlength on filter performance.
5. Implementation of filter in software and/or hardware.

1. Specification of the filter requirements

Requirement specifications include specifying the following:

(i) **Signal characteristics** like data rates and width, the highest frequency of interest etc.

(ii) **Filter characteristics** like the desired amplitude and/or phase responses and their tolerances, speed of operation etc.

(iii) **The manner of implementation**, for example as a high-level language on a computer or as a DSP processor-based system.

(iv) **Other design constraints** like cost of the filter etc.

- The characteristics of digital filters are often specified in the frequency domain.

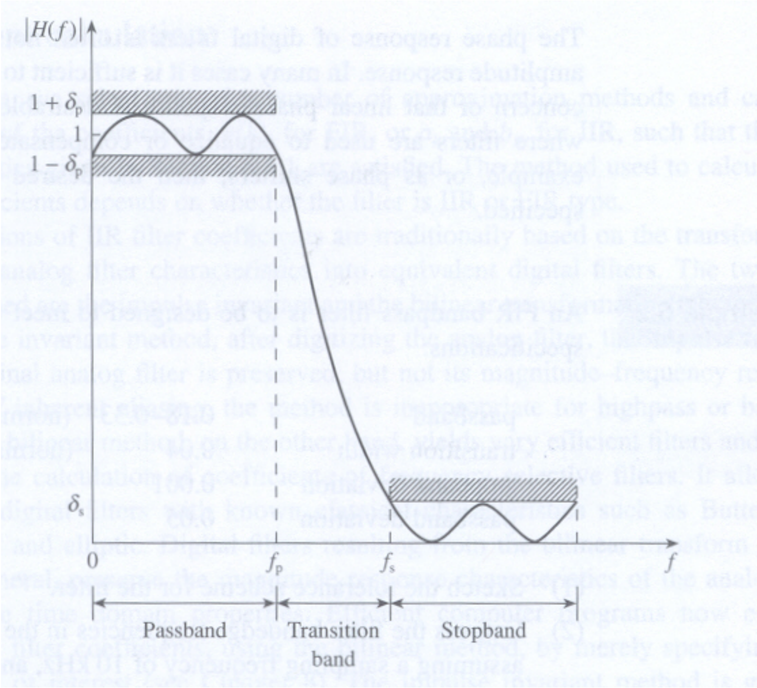


Fig 4: Tolerance scheme for a low pass filter[1]

- The width of the transition band specifies how sharp the filter is.

The following are the key parameters of interest:

δ_p	passband deviation
δ_s	stopband deviation
f_p	passband edge frequency
f_s	stopband edge frequency

A_s (stopband attenuation) = $-20\log_{10}\delta_s$

A_p (passband ripple) = $20\log_{10}(1+\delta_p)$

2. Coefficient calculation

- **For IIR filters** (calculating a_k and b_k)

• Calculations of IIR filter coefficients are traditionally based on the transformation of known analog filter characteristics into equivalent digital filters.

• The two basic methods used are **impulse invariant** and **bilinear transformation** methods.

• The impulse invariant method is good for simulating analog systems, but the bilinear method is best for frequency selective IIR filters.

• The **pole-zero placement** method offers an alternative approach and is an easy way of calculating the coefficients of very simple filters.

- **For FIR filters** (calculating h(k))

• The three methods generally followed are **window**, **frequency sampling**, and **optimal (Parks-McClellan algorithm)** methods.

• The window method offers a very simple and flexible way of computing FIR filter coefficients, but it does not allow the designer adequate control over the filter parameters.

• Frequency sampling method allows a recursive realization of FIR filters which can be computationally very efficient, but however lacks flexibility in specifying or controlling filter parameters.

• The optimal method is now widely used in the industry, with the availability of an efficient and easy-to-use program, and for most applications, will yield the desired FIR filters.

3. Representation of a filter by a suitable structure (realization)

- Realization involves converting a given transfer function, H(z), into a suitable filter structure that is generally depicted using block or flow diagrams and shows the computational procedure for implementing the digital filter.

- Commonly used realization structures for FIR and IIR filters:

FIR

- transversal (direct)
- frequency sampling
- fast convolution
- lattice

IIR

- direct form
- cascade
- parallel
- lattice

4. Analysis of finite wordlength effects

- In actual implementations, it is necessary to represent the filter coefficients using a limited number of bits, typically 8 to 16 bits.

- The arithmetic operations indicated in the difference equations are performed using finite precision arithmetic.

- As a result, there is a degradation in the performance of the filters from the following sources:

  1. Input/output signal quantization
  2. Coefficient quantization
  3. Arithmetic roundoff errors
  4. Overflow

- The extent of filter degradation depends on:

  (i) the wordlength and type of arithmetic used to perform the filtering operation

  (ii) the method used to quantize filter coefficients and variables to the chosen wordlengths

  (iii) the filter structure

- With a knowledge of these factors, the designer can assess the effects of finite wordlength on the filter performance and take remedial action if necessary.

5. Implementation of a filter

- The computation of the filter output y(n) involves only multiplications, additions/subtractions and delays.

- Thus to implement the filter, we need the following basic blocks:

  - memory (for example ROM) for storing filter coefficients;

  - memory (such as RAM) for storing the present and past inputs and outputs;

  - hardware or software multiplier(s);

  - adder or arithmetic logic unit;

- The designer provides these basic blocks and also ensures that they are suitably configured for the application.

- For *batch (non-real time) processing*, the entire data is already available in some memory device and the filter is often implemented in a high level language and runs in a general-purpose computer.

- *Real-time filtering* may require fast and dedicated hardware if the sample rate is very high or if the filter is of a high order.

- For most audio-frequency work, DSP processors such as the DSP56000 (by Motorola) and the TMS320C25 (by Texas Instruments) will be adequate and offer considerable flexibility.

References

1. “Digital Signal Processing – A Practical Approach” - Emmanuel C. Ifeachor and Barrie W. Jervis
   Second Edition