Dr. Norbert Cheung's Series in Electrical Engineering

Level 1 Topic no: 03-l

Introduction to Filters Design and Implementation

Contents

Reference:

"Modern Digital Control Systems, 2nd edition" Raymond G. Jacquot, Longman.

Email: norbert.cheung@polyu.edu.hk **Web Site:** www.ncheung.com

1. Using prototypes to design analogue filters

One good prototype filter is the Butterworth filter, which has minimum ripple in the passband and stopband, although it does not make the transition from passband to stopband as crisply as other possible filters. The best way to start the process is to begin with a unity-gain, unity-bandwidth lowpass Butterworth filter of a fixed order. Typical low-order unity bandwidth (1 rad/sec) Butterworth transfer functions are given in Table 5.2.

Table 5.2. Butterworth Unity-Gain, Unity-Cutoff Frequency Transfer

Order	$H_B(s)$
2	$\dfrac{1}{s^2 + 1.414s + 1}$
3	$\dfrac{1}{s^3 + 2s^2 + 2s + 1}$
4	$\dfrac{1}{s^4 + 2.6133s^3 + 3.414s^2 + 2.6133s + 1}$

For a lowpass filter with arbitrary cutoff frequency ω_0, we simply employ the lowpass transformation whereby s is replaced by s/ω_0, which essentially frequency scales the lowpass filter:

$$s \rightarrow \frac{s}{\omega_0} \qquad (5.9.1)$$

For a bandpass filter with the passband centered at ω_0 and a bandwidth of BW rad/s we employ the bandpass transformation, which is

$$s \rightarrow \frac{1}{\mathrm{BW}}\left(\frac{s^2 + \omega_0^2}{s}\right) \qquad (5.9.2)$$

If a highpass filter is desired, we use the highpass transformation, which is

$$s \rightarrow \frac{\omega_0}{s} \qquad (5.9.3)$$

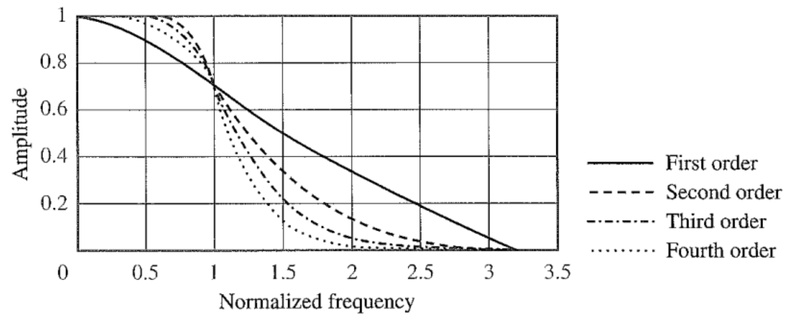


Figure 15.20 Butterworth low-pass filter frequency response

Table 15.3 Butterworth polynomials in quadratic form

Order n	Quadratic factors
1	$s+1$
2	$s^2+\sqrt{2}s+1$
3	$(s+1)(s^2+s+1)$
4	$(s^2+0.7654s+1)(s^2+1.8478s+1)$
5	$(s+1)(s^2+0.6180s+1)(s^2+1.6180s+1)$

Example 5.1. Consider a second-order bandpass filter with a bandpass frequency of 10 rad/s and a Q of 5 with the continuous-time transfer function

$$H(s) = \frac{Y(s)}{U(s)} = \frac{2s}{s^2 + 2s + 100} \qquad \text{(a)}$$

If we cross-multiply the transfer function, we get

$$(s^2 + 2s + 10)Y(s) = 2sU(s)$$

Inversion of the indicated transform yields a differential equation description

$$\frac{d^2y}{dt^2} + 2\frac{dy}{dt} + 100y = 2\frac{du}{dt} \qquad \text{(b)}$$

An alternative description is that of the impulse response function, which is the inverse Laplace transform of the transfer function of (a). This inversion can be accomplished by adding a constant term to the numerator of (a) and then subtracting a term of the same size upon completing the square in the denominator

$$H(s) = \frac{2(s+1)}{(s+1)^2 + 99} - \frac{2}{\sqrt{99}}\frac{\sqrt{99}}{(s+1)^2 + 99}$$

Using the appropriate table entries in Appendix A, the impulse response function is

$$h(t) = \begin{cases} 2e^{-t}\cos\sqrt{99}t - \dfrac{2}{\sqrt{99}}e^{-t}\sin\sqrt{99}\,t & t \ge 0 \\ 0 & t < 0 \end{cases} \qquad \text{(c)}$$

The frequency response of this filter is

$$H(j\omega) = H(s)\big|_{s=j\omega} = \frac{2j\omega}{-\omega^2 + 2j\omega + 100} \qquad \text{(d)}$$

2. Conventional design techniques

General filter function in s-domain

If the filter described by relation (5.2.1) has numerator and denominator polynomials of equal order, long division must be carried out to give a constant plus a remainder proper polynomial ratio. The result is of the form

$$H(s) = b_n + \frac{c_{n-1}s^{n-1} + \cdots + c_1 s + c_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0} = \frac{Y(s)}{U(s)} \qquad (5.2.4)$$

If we now define a new response variable $W(s)$ by the transfer function relation

$$\frac{W(s)}{U(s)} = \frac{1}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0} \qquad (5.2.5)$$

and thus the output relation is

$$Y(s) = b_n U(s) + (c_{n-1}s^{n-1} + \cdots + c_1 s + c_0)W(s) \qquad (5.2.6)$$

Relation (5.2.5) could be realized as a cascade of integrators with feedback to create the denominator dynamics as illustrated in the lower half of Fig. 5.1. Relation (5.2.6) can be thought of as a feedforward of the input $U(s)$ and the outputs of the integrators as illustrated in the upper portion of Fig. 5.1. One common way to convert the continuous-time filtering of Fig. 5.1 to a digital filter would be to change the signals in the block diagram to z-domain signals and to replace the integrator ($1/s$) blocks by approximate discrete-time integrators. In the following section we explore several approximate integration algorithms.

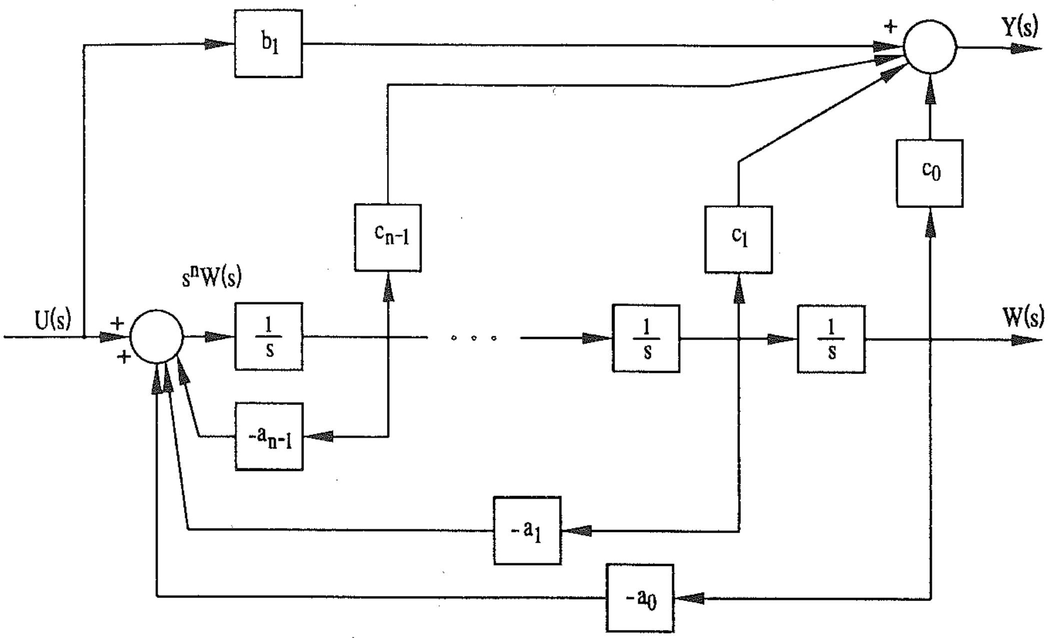


Figure 5.1. Phase-variable filter realization.

Example 5.2. The filter examined in Example 5.1 had a transfer function

$$H(s) = \frac{2s}{s^2 + 2s + 100} = \frac{Y(s)}{U(s)}$$

From relation (5.2.5) the forward dynamics are described by

$$\frac{W(s)}{U(s)} = \frac{1}{s^2 + 2s + 100}$$

and the output equation from (5.2.6) is $Y(s) = 2sW(s)$. These dynamics are represented in Fig. 5.2.

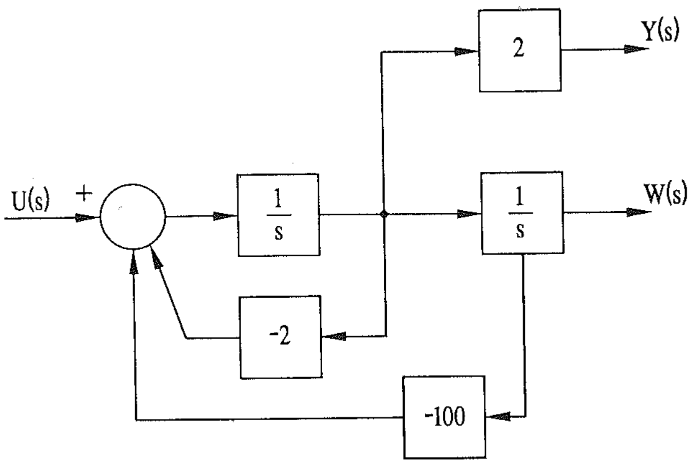


Figure 5.2. Bandpass filter realization.

3. Approximate numerical integration techniques

In this section we explore several techniques for numerical integration with the purpose in mind of approximation of filter transfer functions. The function we wish to approximate is represented in Fig. 5.3. The input–output relation is

$$y(t) = \int_0^t x(\tau)\, d\tau \tag{5.3.1}$$

which can be thought of the area under the $x(\tau)$ curve between $\tau = 0$ and $\tau = t$.

Forward Rectangular Integration

In Fig. 5.4 we note that the integral at time kT will be denoted as $y(kT)$ and that at time $(k - 1)T$ will be denoted as $y((k - 1)T)$.

The area accumulated on the interval $((k - 1)T, kT)$ is then $x((k - 1)T) \cdot T$. Then the area at time kT can be thought of as the area at $(k - 1)T$ plus the rectangular area shown in Fig. 5.4. The integration algorithm is then

$$y(kT) = y((k - 1)T) + x((k - 1)T) \cdot T \tag{5.3.2}$$

This difference equation algorithm for numerical integration can be transformed to give a z-domain transfer function

$$H(z) = \frac{Y(z)}{X(z)} = \frac{T}{z - 1} \tag{5.3.3}$$

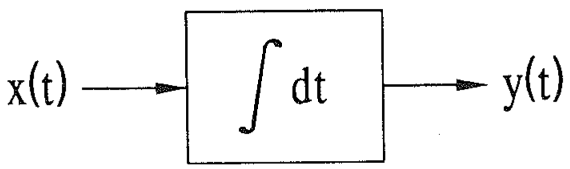


Figure 5.3. Continuous time-domain integrator.

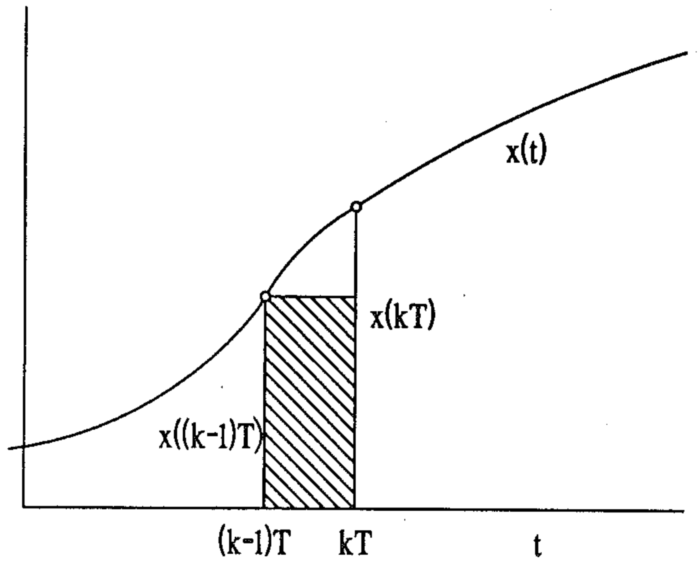


Figure 5.4. Forward rectangular integration.

Backward Rectangular Integration

Consider now an alternative definition for the integral approximation illustrated in Fig. 5.5. With this definition the integration algorithm becomes

$$y(kT) = y((k-1)T) + x(kT) \cdot T \tag{5.3.4}$$

Upon z-transformation the transfer function for this discrete-time integra-

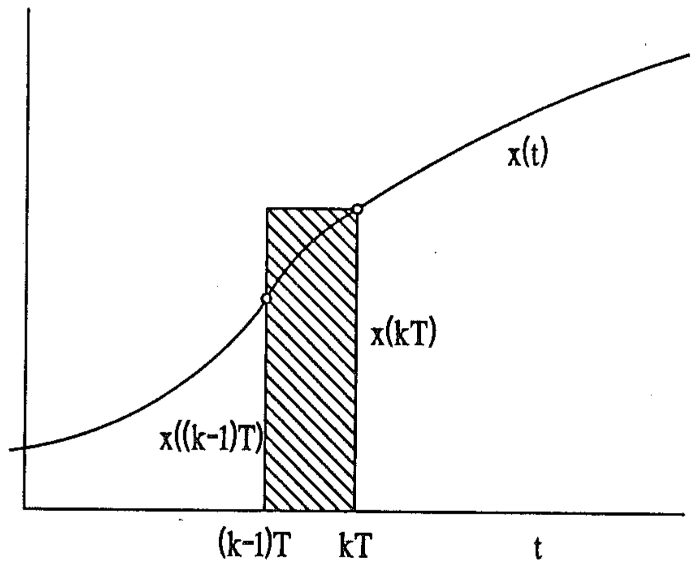


Figure 5.5. Backward rectangular integration.

tor is

$$H(z) = \frac{Y(z)}{X(z)} = \frac{Tz}{z - 1} \qquad (5.3.5)$$

which differs in the numerator from that given by forward rectangular integration.

Trapezoidal Integration

Perhaps a better approximation to the integral can be obtained by using both samples of the $x(t)$ function in computation of the additional accumulated area by using the trapezoidal area illustrated in Fig. 5.6.

For this approximation the additional accumulated area is the area of the trapezoid or the algorithm becomes

$$y(kT) = y((k - 1)T) + \frac{T}{2}[x((k - 1)T) + x(kT)] \qquad (5.3.6)$$

and the associated z-domain transfer function is

$$H(z) = \frac{T}{2}\left(\frac{z + 1}{z - 1}\right) \qquad (5.3.7)$$

This is commonly referred to as the bilinear transformation, and the method of filter synthesis using this method is sometimes called Tustin's method

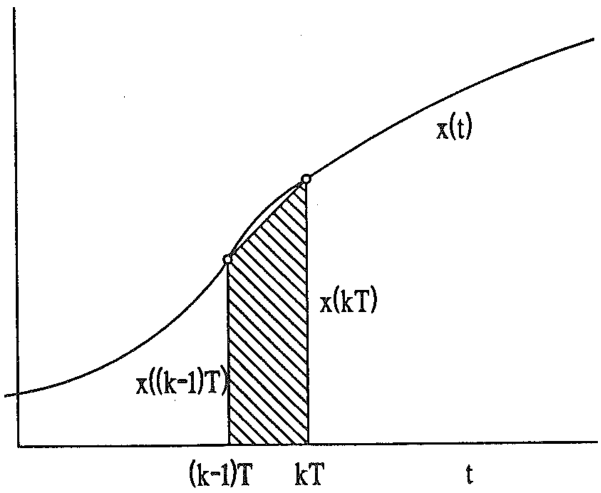


Figure 5.6. Trapezoidal integration.

Table 5.1. Substitutions for Various Integration Methods

Method	
Forward rectangular	$s = (z - 1)/T$
Backward rectangular	$s = (z - 1)/Tz$
Trapezoidal (bilinear transformation)	$s = 2(z - 1)/T(z + 1)$

Example 5.3. Consider the second-order bandpass filter with the passband centered at 10 rad/s and a quality factor of $Q = 5$, which was illustrated in Example 5.1. The s-domain transfer function is

$$H(s) = \frac{Y(s)}{U(s)} = \frac{2s}{s^2 + 2s + 100} \qquad (*)$$

Synthesize an "equivalent" discrete-time filter using the backward rectangular integration method. Since the critical frequency is at 10 rad/s (1.59 Hz), a reasonable sampling frequency would be 10 Hz or the sampling period T will be 0.1 s. The appropriate substitution from Table 5.1 would be

$$s = \frac{10(z - 1)}{z}$$

Substitution of this relation into relation (*) yields

$$H(z) = \frac{2\left[\frac{10(z-1)}{z}\right]}{\left[\frac{10(z-1)}{z}\right]^2 + 2\left[\frac{10(z-1)}{z}\right] + 100}$$

Cleaning up the algebra yields the transfer function

$$H(z) = \frac{0.0909z(z - 1)}{z^2 - 1.0z + 0.4545}$$

This filter has zeros at $z = 0$ and $z = 1$, while there is a complex pair of poles at $z = 0.5 \pm j0.4522$.

The difference equation, which is equivalent to the transfer function, is

$$y(k + 2) = y(k + 1) - 0.4545y(h) + 0.0909u(k + 2) - 0.0909u(k + 1)$$

Example 5.4. Consider the bandpass filter considered in Example 5.1 with transfer function

$$H(s) = \frac{2s}{s^2 + 2s + 100}$$

Synthesize an "equivalent" digital filter employing the bilinear transformation method and a sampling integral of $T = 0.1$ s.

The appropriate bilinear transformation is

$$s = \frac{20(z - 1)}{z + 1}$$

Substitution of this relation into the original s-domain transfer function yields

$$H(z) = \frac{\dfrac{40(z-1)}{z+1}}{400\left(\dfrac{z-1}{z+1}\right)^2 + (2)(20)\left(\dfrac{z-1}{z+1}\right) + 100}$$

Upon cleaning up the algebra yields a z-domain transfer function is

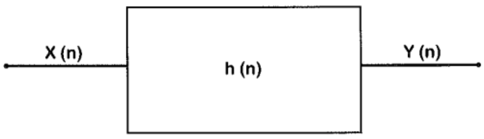
$$H(z) = \frac{0.0740(z - 1)(z + 1)}{z^2 - 1.111z + 0.8519}$$

This filter has zeros at $z = 1$ and $z = -1$ and poles at $z = 0.5556 \pm j0.737$. The difference equation algorithm for real-time filtering is

$$y(k + 2) = 1.111y(k + 1) - 0.8519y(k) + 0.0740u(k + 2) - 0.0740u(k)$$

4. FIR digital filters

The FIR filters are also called *non-recursive* filters. The recursive filters use present and past values to perform computations. The past values used are the previous inputs and outputs. Since digital filters use microprocessors, the past values are stored in the memory. Non-recursive filters use a finite number of the past input samples, and only have feedforward circuits and no feedback circuitry. The best way to understand the concepts of an FIR filter is by representing it either in a graphical or mathematical form. There are several ways an FIR filter can be represented. These differ-



An expression for the FIR filter is given as follows:

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

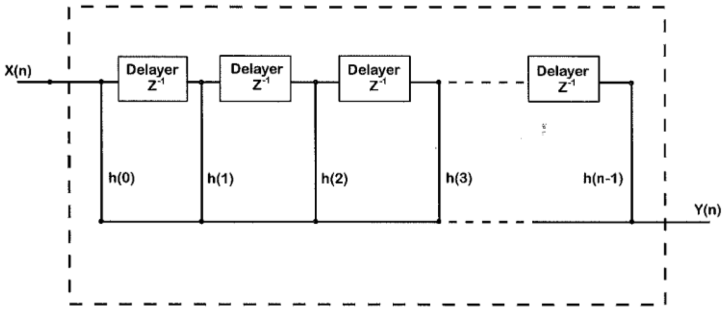
To find the transfer function, we do the following:

$$h(n) = \frac{y(n)}{x(n)}$$

Taking the z-transform of this expression, we get:

$$H(z) = \frac{Y(z)}{X(z)} = \sum_{k=0}^{M-1} h(k) z^{-k}$$

The term z^{-1} represents a delayer. To represent the FIR filter in a direct form, we use the above expressions to derive the structure as shown in *Figure 8-6*.



This is called a *direct-form* FIR filter. The above figure shows a filter of *n* length, where n is a finite number.

For example, for a filter of length 3, we write the expression as follows:

$$y(n) = \sum_{k=0}^{2} h(k) x(n-k)$$

Example 6.1: Direct Form

Given the transfer function of an FIR filter as $H(z) = \sum_{n=0}^{M} h(n)z^{-n}$, let us consider its equivalent algorithm for the output, for example, when $M = 4$:

$$\begin{aligned} y(n) = h(0)x(n) + h(1)x(n-1) + h(2)x(n-2) \\ + h(3)x(n-3) + h(4)x(n-4) \end{aligned} \tag{6.5}$$

We have already discussed one structure employed to implement this algorithm in Chapter 5, and because the coefficients of the multipliers in it are directly available as the coefficients $h(n)$ in $H(z)$, it is called the *direct form I structure* and is shown in Figure 6.1.

Whenever we have a structure to implement an FIR or an IIR filter, an equivalent structure can be obtained as its transpose by the following operations:

1. Interchanging the input and the output nodes
2. Replacing adders by pickoff nodes and vice versa
3. Reversing all paths

Using these operations, we get the transpose of the structure of Figure 6.1 as Figure 6.2. This is known as *direct form II structure*; remember that this (direct form II) structure will be called *direct form I transposed structure* in the next chapter.

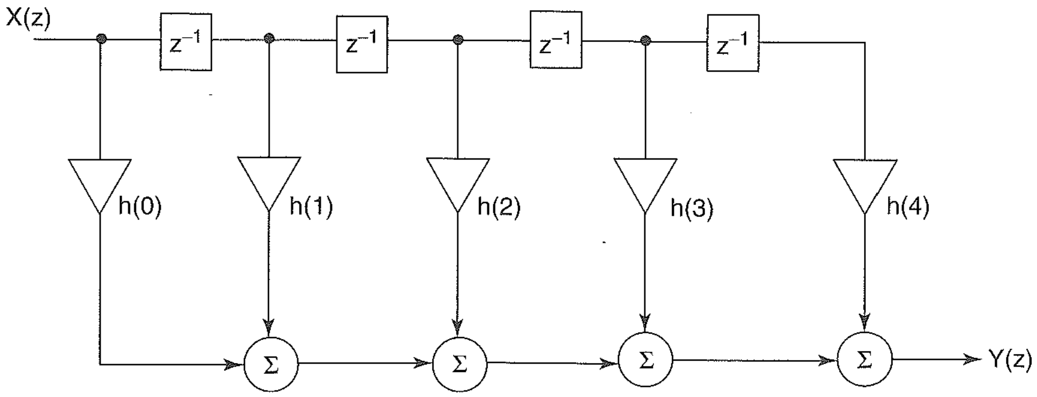
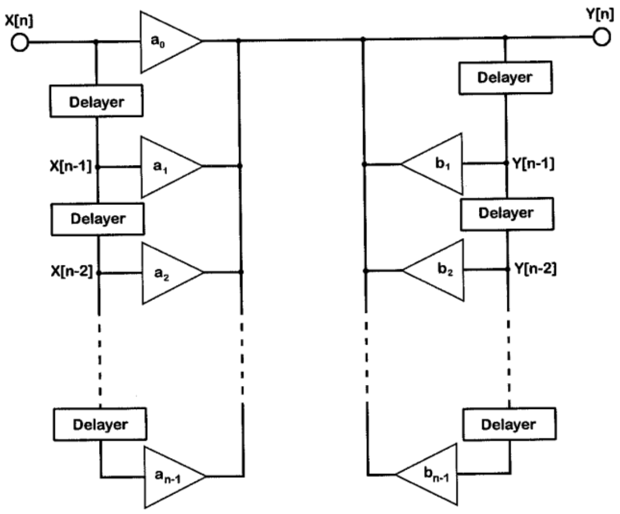


Figure 6.1 Direct form I of an FIR filter.

5. IIR Digital Filters

IIR (Infinite Impulse Response) filters are recursive filters. Unlike non-recursive filters, these filters use feedback to obtain the past values of outputs and inputs. An expression for an IIR filter is:

$$y(n)=\sum_{k=0}^{M-1}a(k)x(n-k)+\sum_{k+1}^{N-1}b(k)y(n-k)$$



An example of realizing low-pass characteristics using an IIR filter is given as follows:

y(n) = a y(n-1) + x(n)

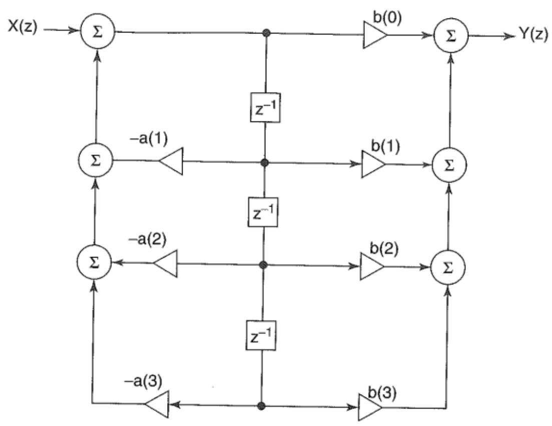


Figure 6.12 Direct form II structure of an IIR filter.

the same as those at the output of the three delay elements of filter $H_2(z)$. Hence we let the two circuits share one set of three delay elements, thereby reducing the number of delay elements. The result of merging the two circuits is shown in Figure 6.12 and is identified as the direct form II realization of the IIR filter. Its transpose is shown in Figure 6.13. Both of them use the minimum number of delay elements equal to the order of the IIR filter and hence are canonic realizations.

5. Glossary – English/Chinese Translation

English	Chinese
analogue filter and digital filter	模拟滤波器和数字滤波器
implementation	实现
approximate numerical integration technique	近似数值积分技术
FIR and IIR digital filters	FIR 和 IIR 数字滤波器
butterworth filter	巴特沃斯过滤器
low pass filter	低通滤波器
high pass filter	高通滤波器
bandpass filter	带通滤波器
numerical integration	数值积分
bilinear transformation	双线性变换
Tustin's method	塔斯汀的方法
finite impulse response	有限脉冲响应
infinite impulse response	无限脉冲响应