

Realization of Digital Filters

Chapter Intended Learning Outcomes:

- (i) Ability to implement finite impulse response (FIR) and infinite impulse response (IIR) filters using different structures in terms of block diagram and signal flow graph
- (ii) Ability to determine the system transfer function and difference equation given the corresponding block diagram or signal flow graph representation.

Filter Implementation

When causality is assumed, a LTI filter can be uniquely characterized by its transfer function $H(z)$:

$$H(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{\sum_{k=0}^N a_k z^{-k}} \quad (9.1)$$

or the corresponding difference equation:

$$\sum_{k=0}^N a_k y[n-k] = \sum_{k=0}^M b_k x[n-k] \quad (9.2)$$

where $x[n]$ and $y[n]$ are the system input and output

Assuming $a_0 \neq 0$, the output is:

$$y[n] = - \sum_{k=1}^N \frac{a_k}{a_0} y[n-k] + \sum_{k=0}^M \frac{b_k}{a_0} x[n-k] \quad (9.3)$$

Assigning $a_k/a_0 \rightarrow a_k$ and $b_k/b_0 \rightarrow b_k$ yields

$$y[n] = - \sum_{k=1}^N a_k y[n-k] + \sum_{k=0}^M b_k x[n-k] \quad (9.4)$$

Computing $y[n]$ involves $y[n-1], y[n-2], \dots, y[n-N]$, and $x[n], x[n-1], \dots, x[n-M]$. That is, we need

- **Delay** elements or storage
- **Multipliers**
- **Adders** (subtraction is considered as addition)

How many storage elements are needed?
How many multipliers are needed?
How many adders are needed?

Computations of $y[n]$ can be arranged in different ways to give the same difference equation, which leads to different **structures** for realization of discrete-time LTI systems

4 basic forms of implementations, namely, **direct form**, **canonic form**, **cascade form** and **parallel form** will be described

An implementation can be represented using either a **block diagram** or a **signal flow graph**

Block Diagram Representation

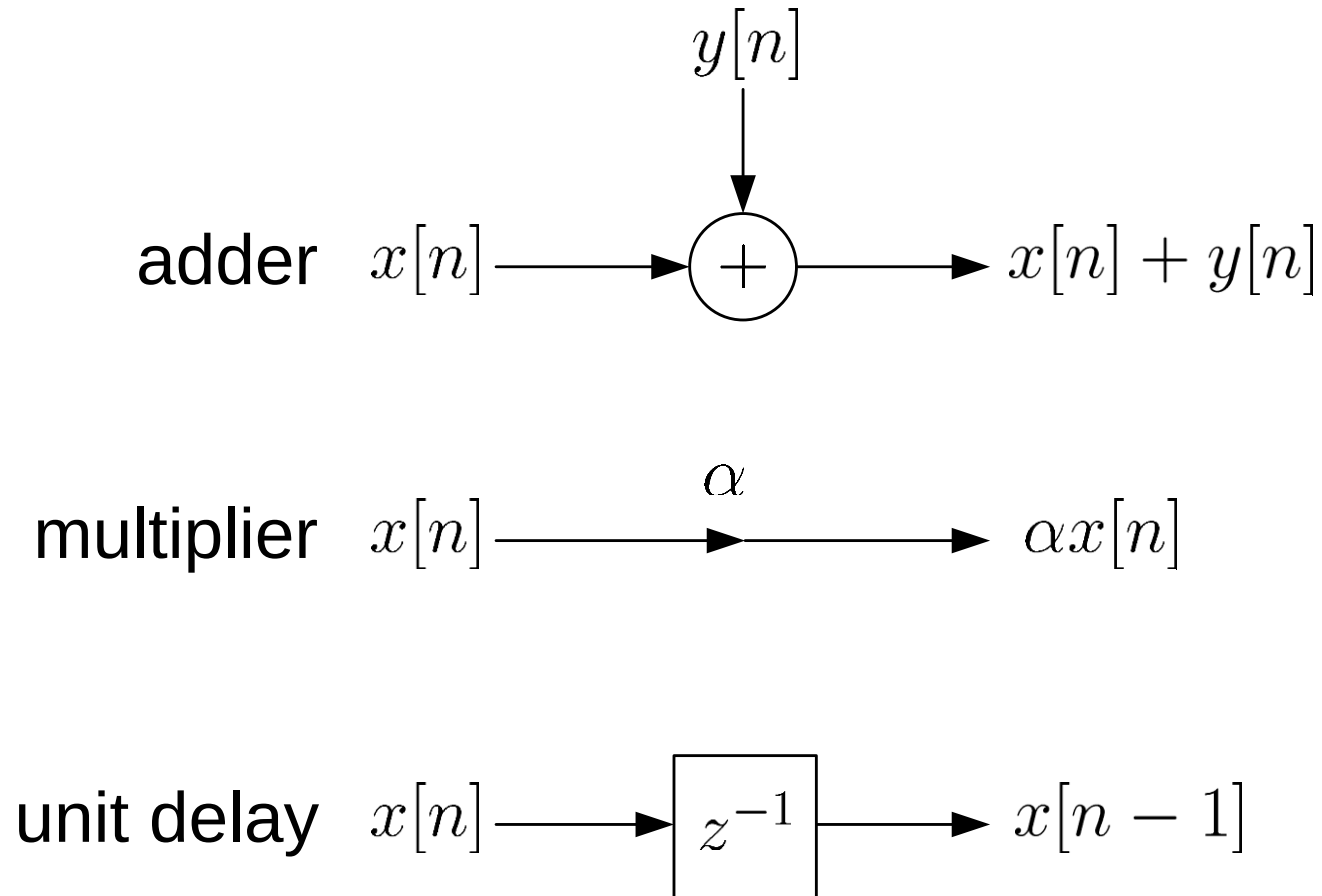


Fig.9.1: Basic operations in block diagram

Although an **adder** can generally deal with more than two sequences, here we consider two signals in order to align with practical implementation in microprocessors.

When $|\alpha| > 1$, it corresponds to signal amplification while the signal is attenuated for $|\alpha| < 1$. Note that a **multiplier** usually has the highest implementation or computational cost and thus it is desired to reduce the number of multipliers in different systems, if possible.

The transfer function z^{-1} corresponds to a unit **delay**. It can be implemented by providing a storage register for each unit delay in digital implementation. If the required number of samples of delay is $D > 1$, then the corresponding system function is z^{-D} .

Signal Flow Graph Representation

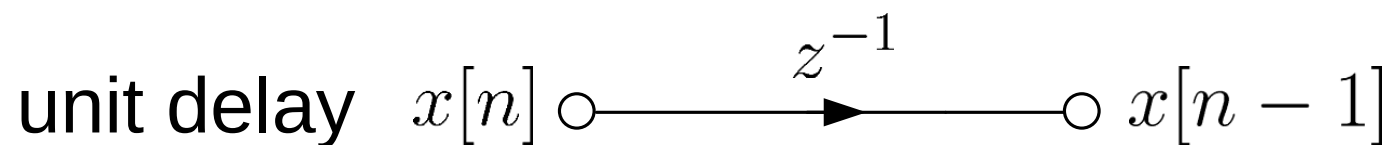
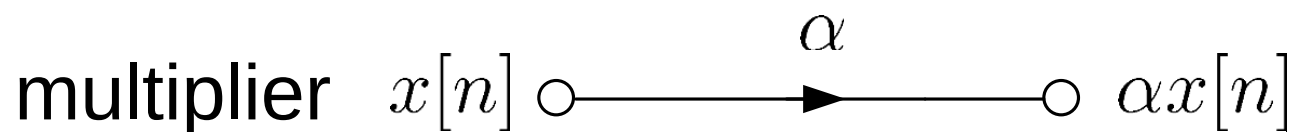
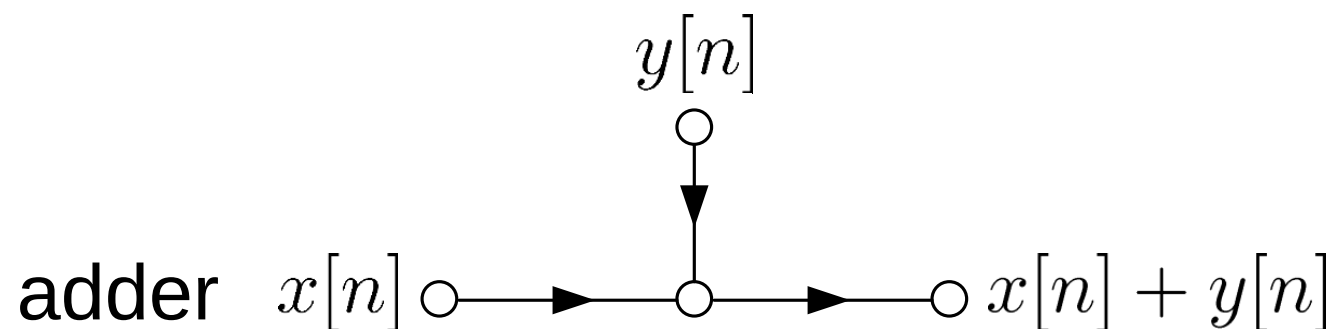


Figure 9.2: Basic operations in signal flow graph

Its basic elements are **branches** with directions, and **nodes**. That is, a signal flow graph is a set of directed branches that connect at nodes.

Signal at a node of a flow graph is equal to the **sum** of the signals from all branches connecting to the node.

Signal out of a branch is equal to the **branch gain** times the signal into the branch.

Branch gain can refer to a **scalar** or a **transfer function** of z^{-1} corresponding to **multiplication** or unit **delay** operation, respectively.

When the branch gain is unity, it is left unlabeled.

A **signal flow graph** provides an alternative but equivalent graphical representation to a **block diagram** structure

Example 9.1

Draw the block diagram and signal flow graph representations of a LTI system whose input $x[n]$ and output $y[n]$ satisfy the following difference equation:

$$y[n] = -a_1y[n-1] - a_2y[n-2] + b_0x[n]$$

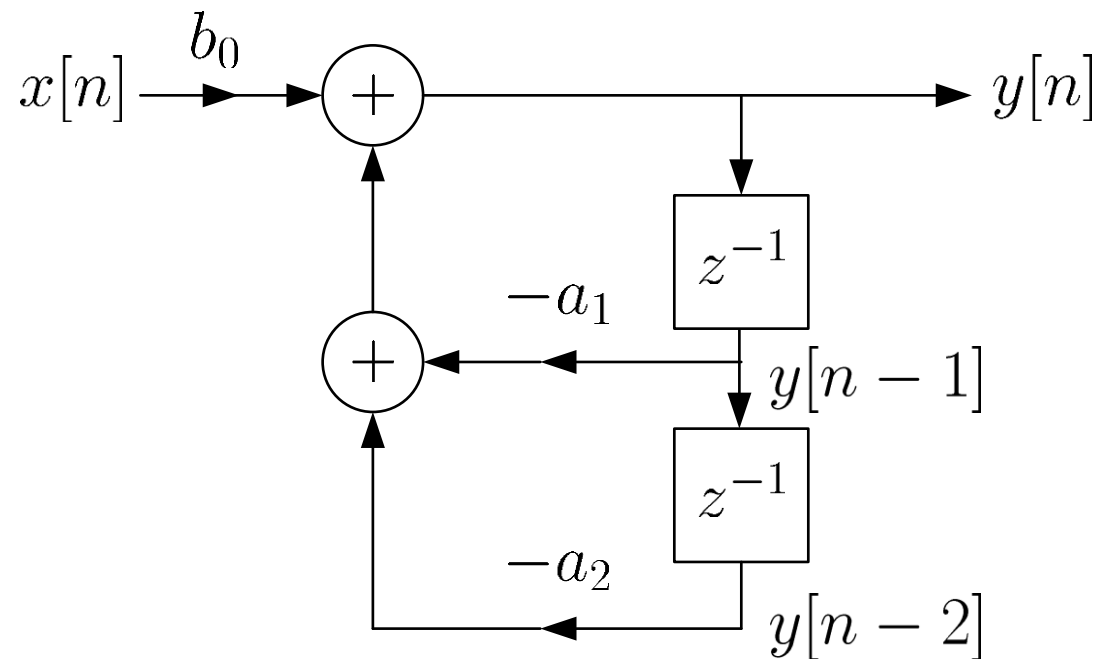


Fig.9.3: Illustration of block diagram

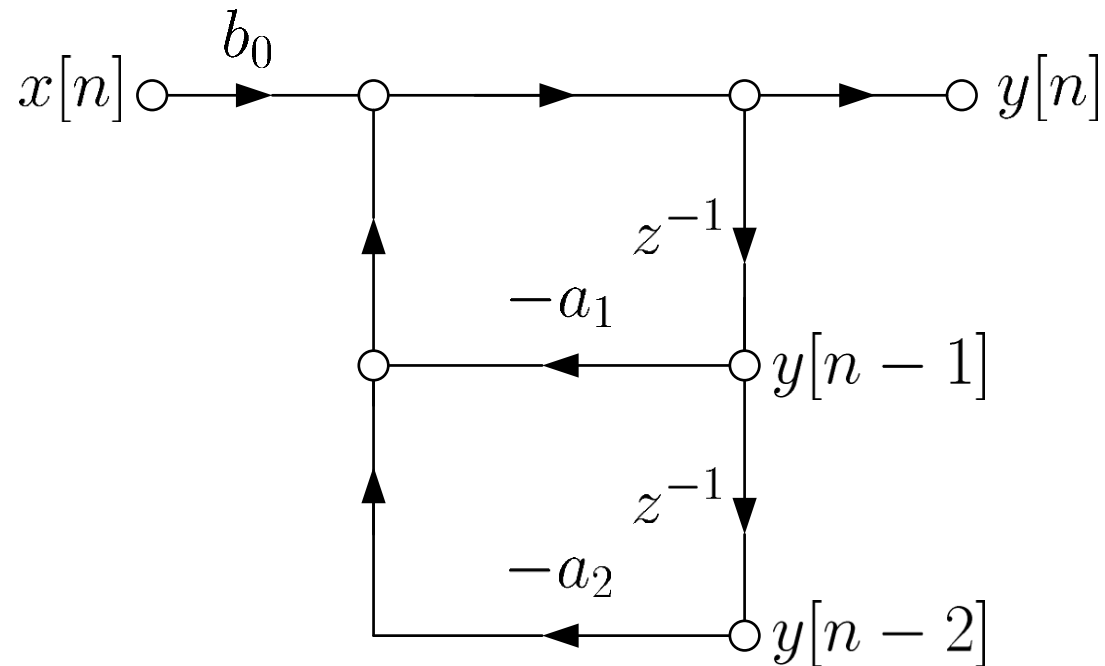


Fig.9.4: Illustration of signal flow graph

2 adders, 3 multipliers and 2 delay elements are required to implement the system.

Structures for FIR Filter

For FIR filter, its transfer function does not contain pole. That is, setting $a_0 = 1$ and $a_1 = a_2 = \cdots = a_N = 0$ in (9.1) yields a FIR system:

$$H(z) = \sum_{k=0}^M b_k z^{-k} \quad (9.5)$$

while the corresponding difference equation is:

$$y[n] = \sum_{k=0}^M b_k x[n - k] \quad (9.6)$$

1. Direct Form

Comparing (9.6) with the convolution formula of (3.18), $\{b_k\}$ correspond to the system impulse response $h[n]$:

$$h[n] = \begin{cases} b_n, & 0 \leq n \leq M \\ 0, & \text{otherwise} \end{cases} \quad (9.7)$$

(9.6) can also be written as:

$$y[n] = \sum_{k=0}^M h[k]x[n-k] = h[0]x[n] + h[1]x[n-1] + \cdots h[M]x[n-M] \quad (9.8)$$

The direct form follows straightforwardly from the difference equation.

The implementation needs M memory locations for storing M previous inputs of $x[n]$, $(M + 1)$ multiplications and M additions for computing each output value of $y[n]$.

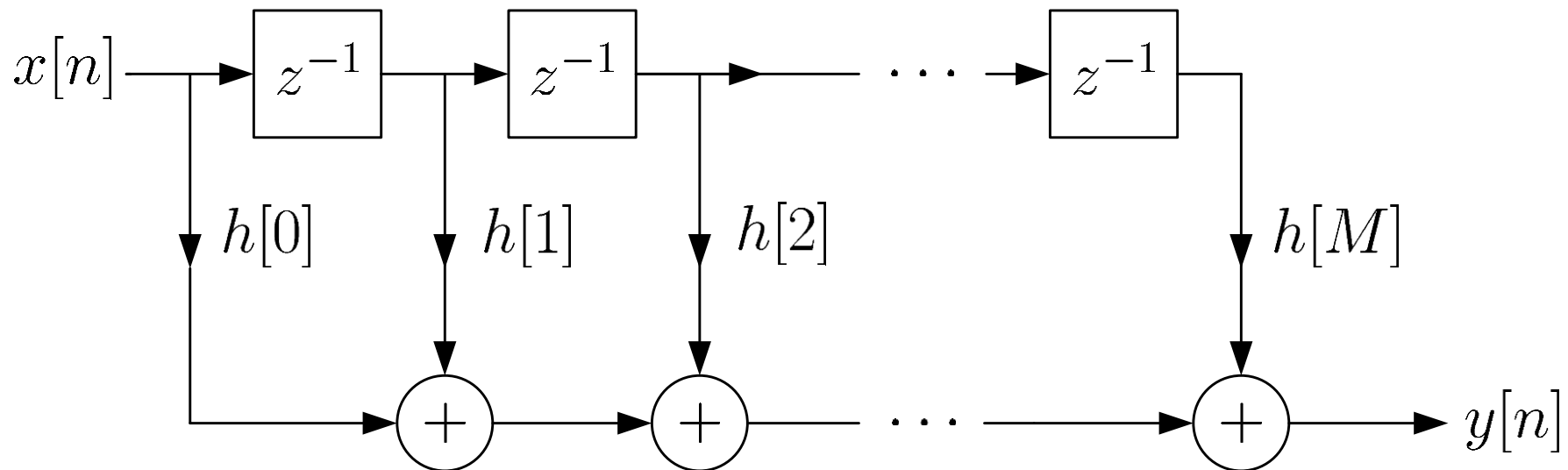


Fig.9.5: Direct form of FIR filter

2. Cascade Form

Expressing (9.5) as products of **second-order** polynomial system functions via factorization:

$$H(z) = \sum_{k=0}^M b_k z^{-k} = \prod_{k=1}^{M_c} (\beta_{0k} + \beta_{1k} z^{-1} + \beta_{2k} z^{-2}) \quad (9.9)$$

where $M_c = \lfloor (M+1)/2 \rfloor$ is the largest integer contained in $(M+1)/2$. Note that when M is odd, one of the $\{\beta_{2k}\}$ will be zero. Assuming that M is even, this implementation needs M storage elements, $3M/2$ multiplications and M additions, for computing each output value of $y[n]$.

Why second-order polynomial instead of first-order polynomial?

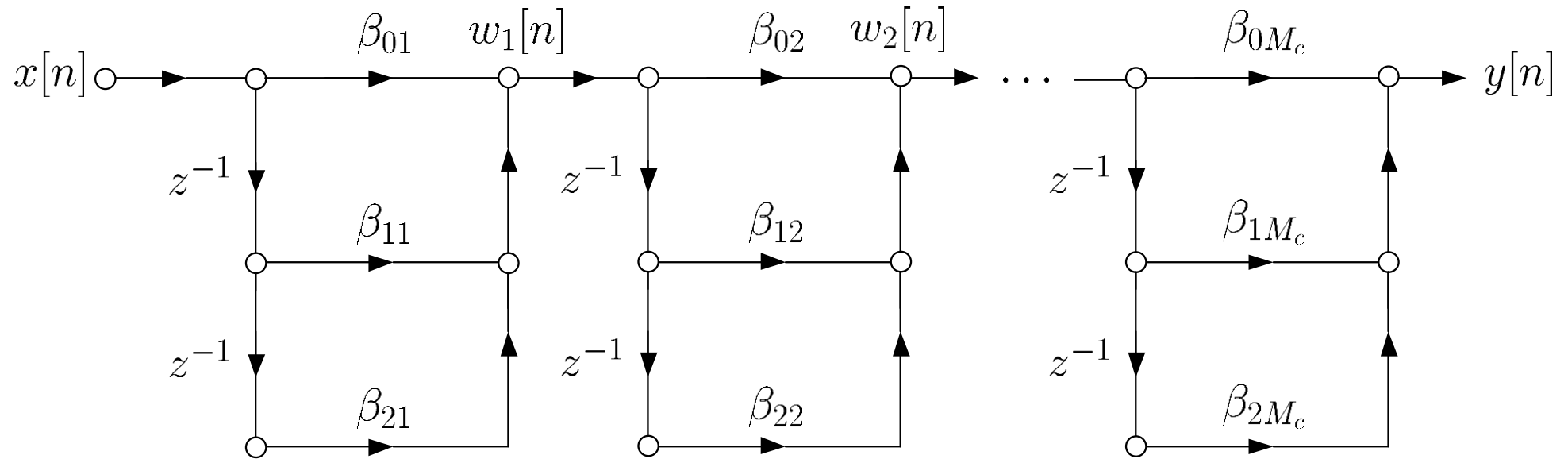


Fig.9.6: Cascade form of FIR filter

$$w_1[n] = \beta_{01}x[n] + \beta_{11}x[n - 1] + \beta_{21}x[n - 2] \quad (9.10)$$

and

$$w_2[n] = \beta_{02}w_1[n] + \beta_{12}w_1[n - 1] + \beta_{22}w_1[n - 2] \quad (9.11)$$

Taking z transform on (9.10) and (9.11):

$$\begin{aligned} W_1(z) &= \beta_{01}X(z) + \beta_{11}z^{-1}X(z) + \beta_{21}z^{-2}X(z) \\ &= (\beta_{01} + \beta_{11}z^{-1} + \beta_{21}z^{-2}) X(z) \end{aligned} \quad (9.12)$$

and

$$\begin{aligned} W_2(z) &= \beta_{02}W_1(z) + \beta_{12}z^{-1}W_1(z) + \beta_{22}z^{-2}W_1(z) \\ &= (\beta_{02} + \beta_{12}z^{-1} + \beta_{22}z^{-2}) W_1(z) \end{aligned} \quad (9.13)$$

Substituting (9.13) into (9.12) yields:

$$W_2(z) = (\beta_{01} + \beta_{11}z^{-1} + \beta_{21}z^{-2}) (\beta_{02} + \beta_{12}z^{-1} + \beta_{22}z^{-2}) X(z) \quad (9.14)$$

Extending (9.14) to $y[n]$, we finally get:

$$\begin{aligned} Y(z) &= (\beta_{01} + \beta_{11}z^{-1} + \beta_{21}z^{-2}) \cdots (\beta_{0M_c} + \beta_{1M_c}z^{-1} + \beta_{2M_c}z^{-2}) \\ &= X(z) \prod_{k=1}^{M_c} (\beta_{0k} + \beta_{1k}z^{-1} + \beta_{2k}z^{-2}) \end{aligned} \quad (9.15)$$

To save the computational complexity, we express (9.9) as:

$$H(z) = G \prod_{k=1}^{M_c} (1 + \beta'_{1k} z^{-1} + \beta'_{2k} z^{-2}) \quad (9.16)$$

where $G = \beta_{01}\beta_{02}\cdots\beta_{0M_c}$, $\beta'_{1k} = \beta_{1k}/\beta_{0k}$ and $\beta'_{2k} = \beta_{2k}/\beta_{0k}$, $k = 1, 2, \cdots, M_c$. That is, all $\{\beta_{0k}\}$ are normalized to 1.

Assuming that M is even, (9.16) needs M delay elements, $(M + 1)$ multiplications and M additions, for computing each output value of $y[n]$

Why (M+1) multiplications are needed?

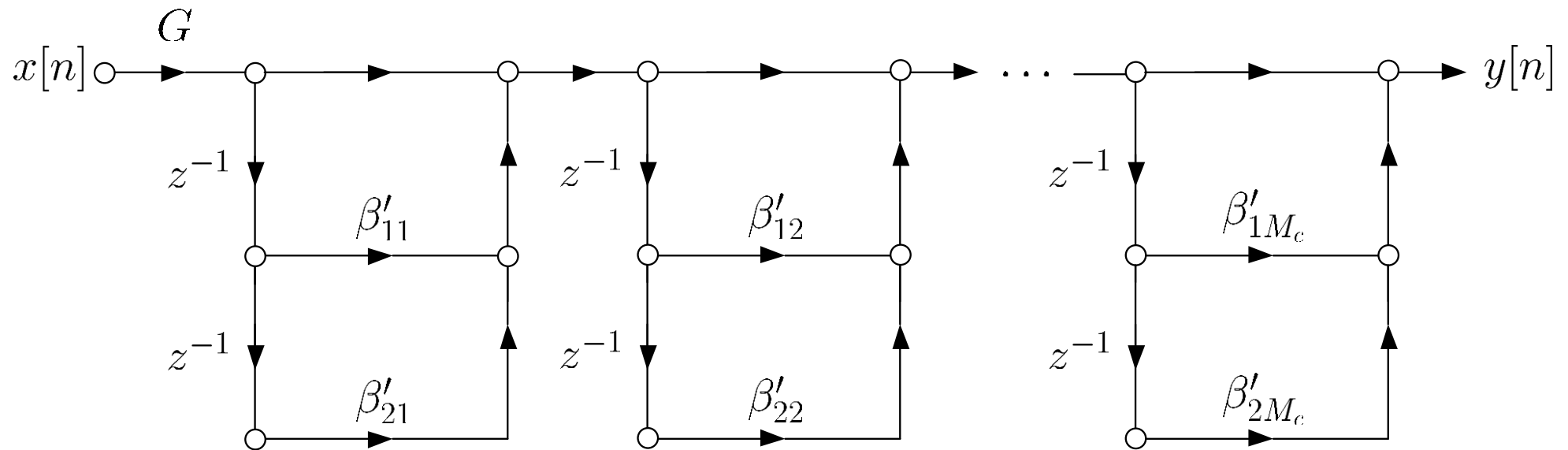


Fig.9.7: Cascade form of FIR filter with smaller complexity

Example 9.2

Draw the signal flow graph using the cascade form for the LTI system whose transfer function is:

$$H(z) = \sum_{k=0}^4 z^{-k} = 1 + z^{-1} + z^{-2} + z^{-3} + z^{-4}$$

To factorize $H(z)$, we use the MATLAB command `roots([1 1 1 1 1])` to solve for the roots:

```
0.3090 + 0.9511i  
0.3090 - 0.9511i  
-0.8090 + 0.5878i  
-0.8090 - 0.5878i
```

Hence $H(z)$ can be factorized as

$$H(z) = (1 - [0.309 + j0.9511]z^{-1}) (1 - [0.309 - j0.9511]z^{-1}) \times \\ (1 - [-0.809 + j0.5878]z^{-1}) (1 - [-0.809 - j0.5878]z^{-1})$$

Although it can be realized with first-order sections, complex coefficients are needed, which implies higher computational cost. To guarantee real-valued coefficients, we group the sections of complex conjugates together:

$$H(z) = (1 - 0.618z^{-1} + z^{-2})(1 + 1.618z^{-1} + z^{-2})$$

$$= (1 + 1.618z^{-1} + z^{-2})(1 - 0.618z^{-1} + z^{-2})$$

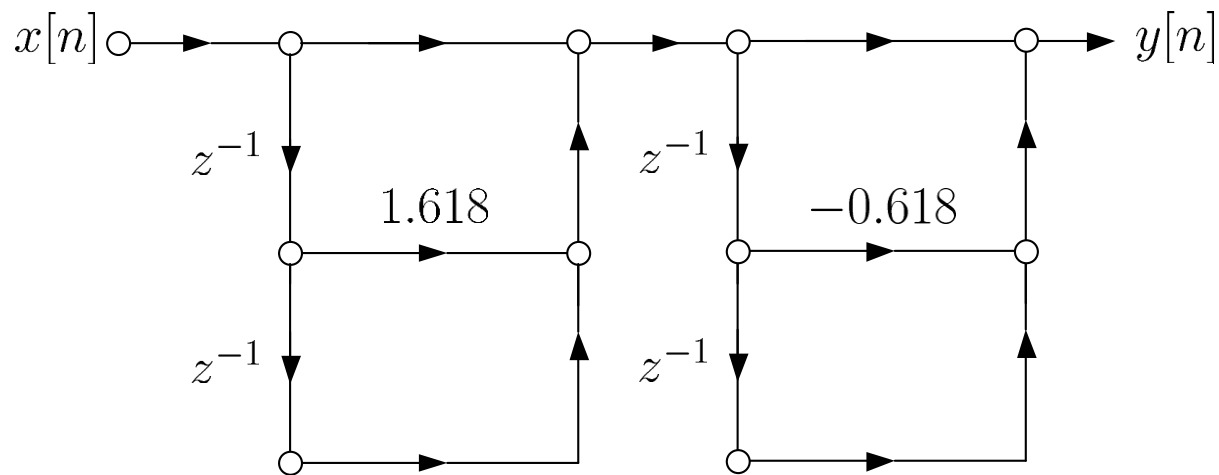
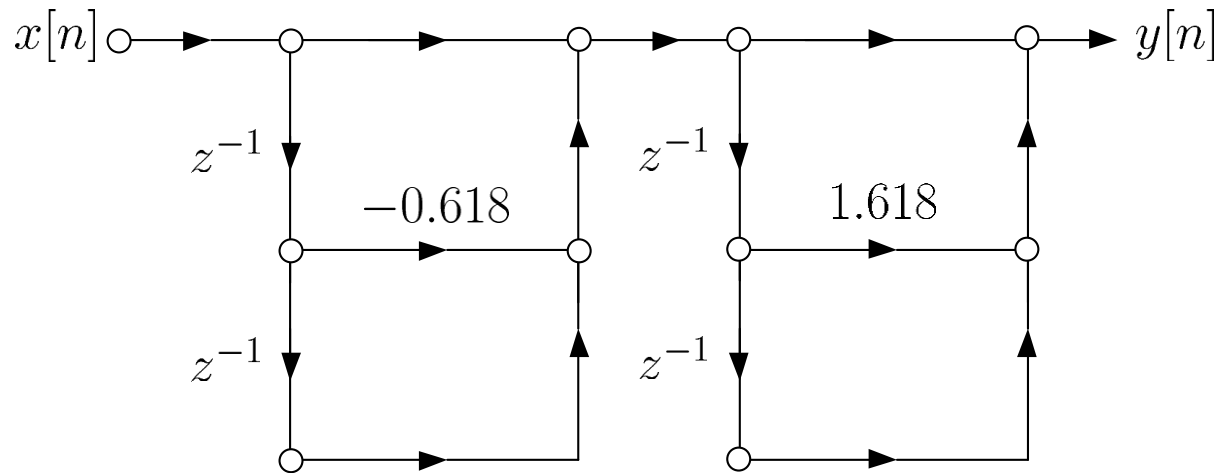


Fig.9.8: Two possible cascade forms for FIR filter

Structures for IIR Filter

When there is at least one pole in $H(z)$, it corresponds to an IIR filter. The corresponding transfer function is thus:

$$H(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{1 + \sum_{k=1}^N a_k z^{-k}} \quad (9.17)$$

That is, IIR filter is the general form of any discrete-time LTI system.

1. Direct Form

It realizes (9.4) in an explicit manner via decomposing it into a pair of difference equations:

$$v[n] = \sum_{k=0}^M b_k x[n - k] \quad (9.18)$$

and

$$y[n] = - \sum_{k=1}^N a_k y[n - k] + v[n] \quad (9.19)$$

The direct form can also be obtained by decomposing $H(z)$ into two transfer functions as:

$$H(z) = H_1(z) \cdot H_2(z) \quad (9.20)$$

where

$$H_1(z) = \sum_{k=0}^M b_k z^{-k} \quad (9.21)$$

and

$$H_2(z) = \frac{1}{1 + \sum_{k=1}^N a_k z^{-k}} \quad (9.22)$$

In the z transform domain, we have:

$$V(z) = X(z)H_1(z) = X(z) \sum_{k=0}^M b_k z^{-k} \quad (9.23)$$

and

$$Y(z) = V(z)H_2(z) = \frac{V(z)}{1 + \sum_{k=1}^N a_k z^{-k}} \quad (9.24)$$

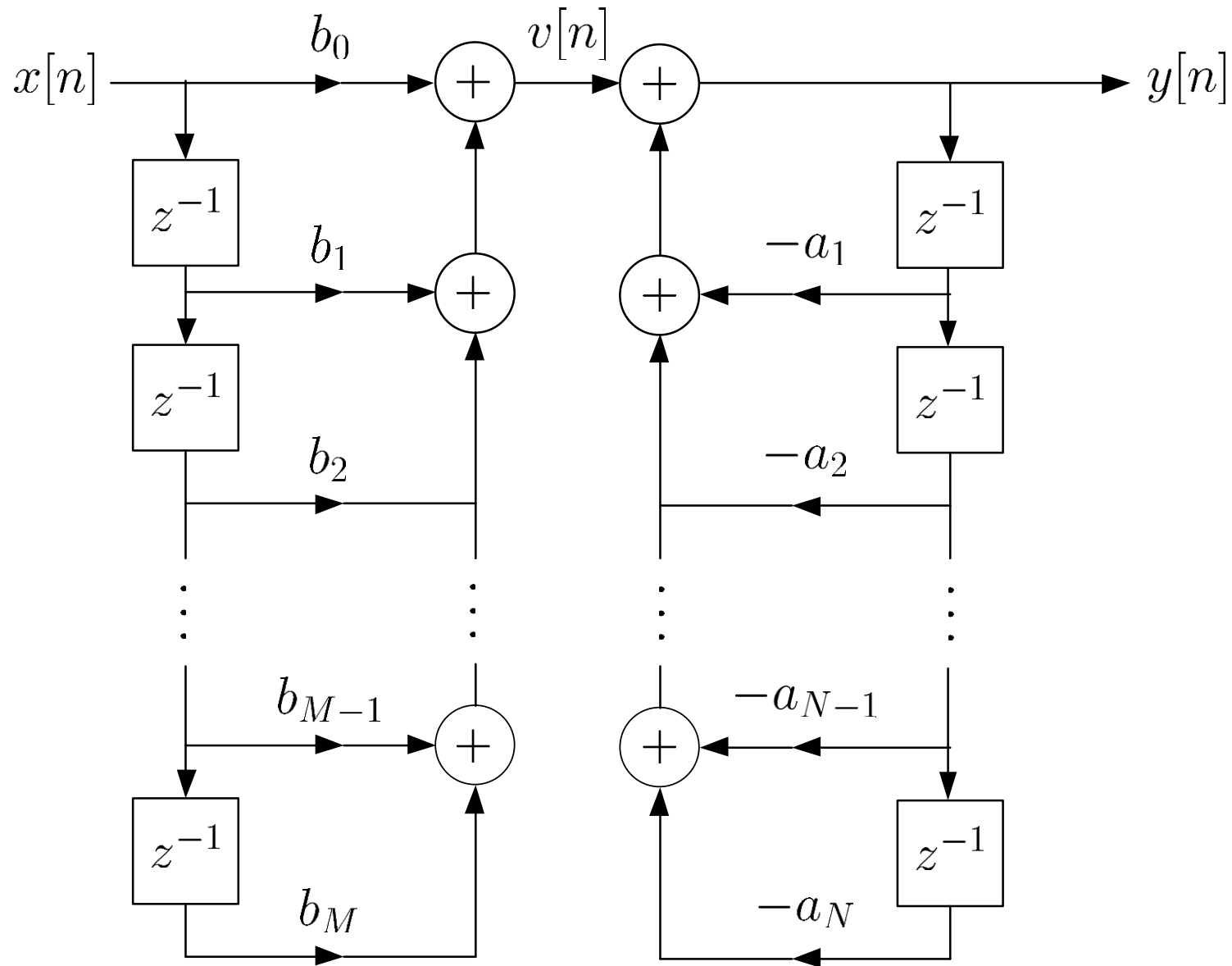


Fig.9.9: Direct form of IIR filter

The direct form implementation needs $(M + N)$ memory locations, $(M + N + 1)$ multiplications and $(M + N)$ additions, for computing each output value of $y[n]$.

2. Canonic Form

On the other hand, we can first pass $x[n]$ through the filter $H_2(z)$ to produce an intermediate signal $w[n]$. The $w[n]$ is then passed through the system $H_1(z)$ to give $y[n]$:

$$W(z) = X(z)H_2(z) = \frac{X(z)}{1 + \sum_{k=1}^N a_k z^{-k}} \quad (9.25)$$

and

$$Y(z) = W(z)H_1(z) = W(z) \sum_{k=0}^M b_k z^{-k} \quad (9.26)$$

Applying inverse z transform, we get:

$$w[n] = - \sum_{k=1}^N a_k w[n - k] + x[n] \quad (9.27)$$

and

$$y[n] = \sum_{k=0}^M b_k w[n - k] \quad (9.28)$$

which can be considered as an alternative direct form.

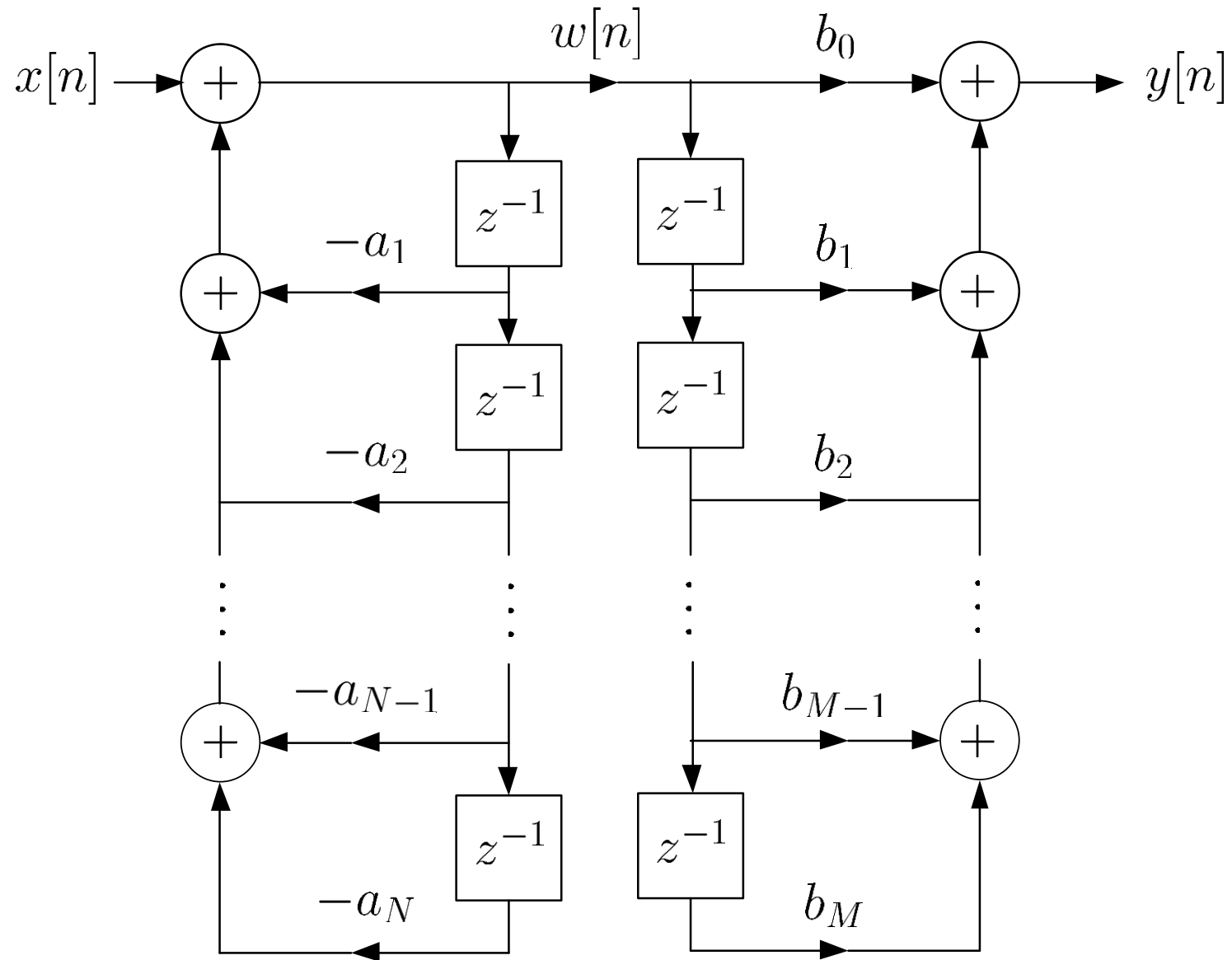


Fig.9.10: Alternative direct form of IIR filter

Assume $M = N$. Since the same signals $w[n], w[n - 1], \dots, w[n - N]$, are stored in the two chains of storage elements, they can be combined to reduce the memory requirement.

In general, the minimum number of delay elements required is $\max(M, N)$.

It is called canonic form because this implementation involves the **minimum number of storages**.

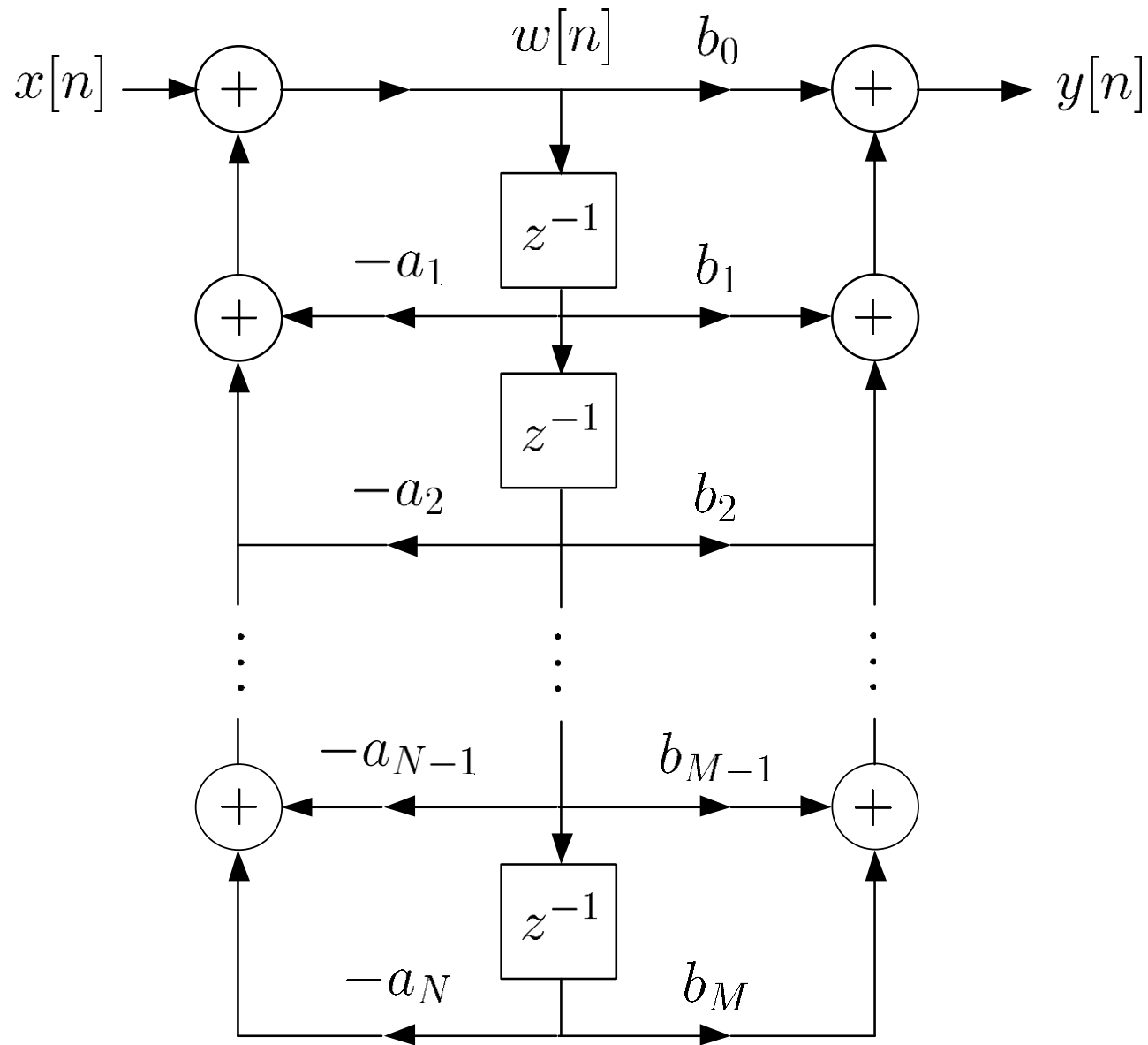


Fig.9.11: Canonic form of IIR filter

Example 9.3

Draw the block diagrams using the direct and canonic forms for the LTI system whose transfer function is:

$$H(z) = \frac{1 - 3z^{-1} + 2z^{-2}}{1 + 0.3z^{-1} - 0.1z^{-2}}$$

According to (9.18)-(9.19):

$$\begin{aligned}v[n] &= x[n] - 3x[n-1] + 2x[n-2] \\y[n] &= -0.3y[n-1] + 0.1y[n-2] + v[n]\end{aligned}$$

Based on (9.27)-(9.28):

$$\begin{aligned}w[n] &= -0.3w[n-1] + 0.1w[n-2] + x[n] \\y[n] &= w[n] - 3w[n-1] + 2w[n-2]\end{aligned}$$

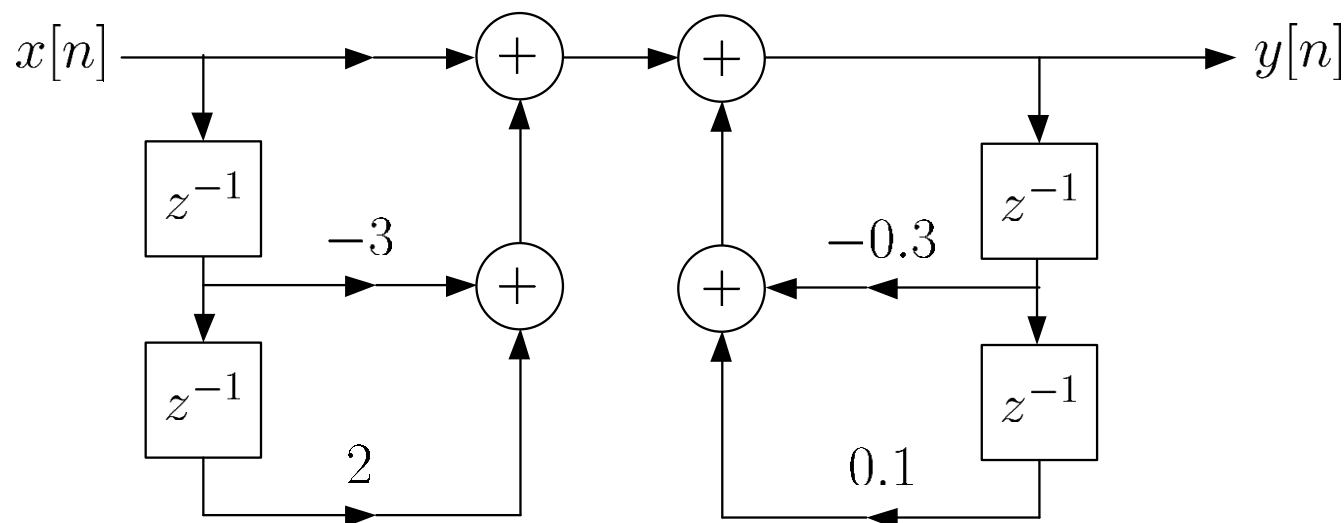


Fig.9.12: Direct form of second-order IIR filter

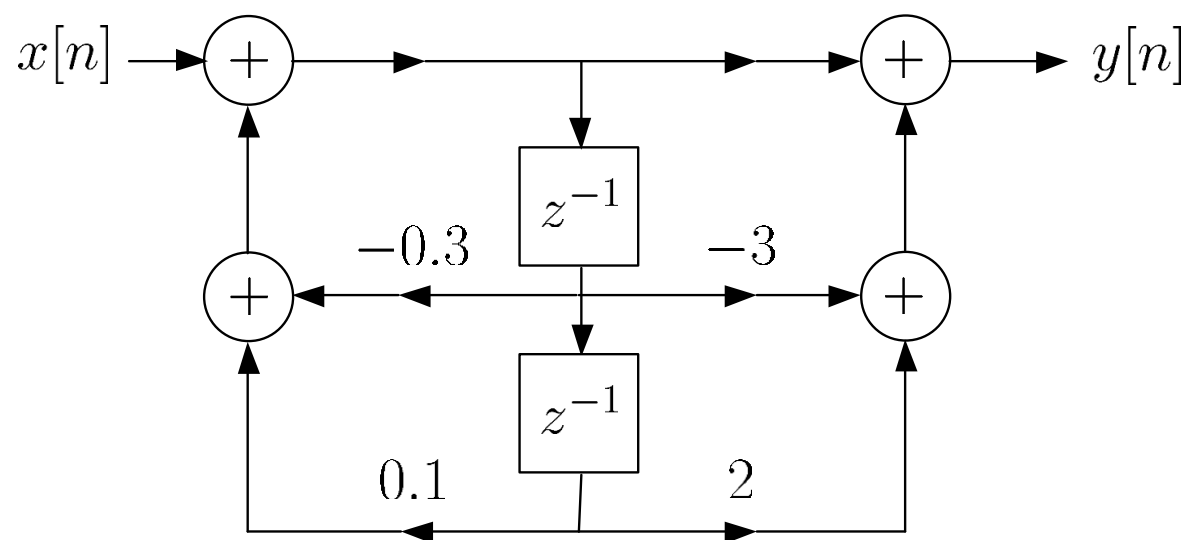


Fig.9.13: Canonic form of second-order IIR filter

3. Cascade Form

We factorize the numerator and denominator polynomials in terms of second-order polynomial system functions as:

$$H(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{1 + \sum_{k=1}^N a_k z^{-k}} = \prod_{k=1}^{N_c} \frac{\beta_{0k} + \beta_{1k} z^{-1} + \beta_{2k} z^{-2}}{1 + \alpha_{1k} z^{-1} + \alpha_{2k} z^{-2}} \quad (9.29)$$

Without loss of generality, it is assumed that $N \geq M$ so that $N_c = \lfloor (N+1)/2 \rfloor$. Note that when M or N is odd, one of the $\{\beta_{2k}\}$ or $\{\alpha_{2k}\}$ will be zero.

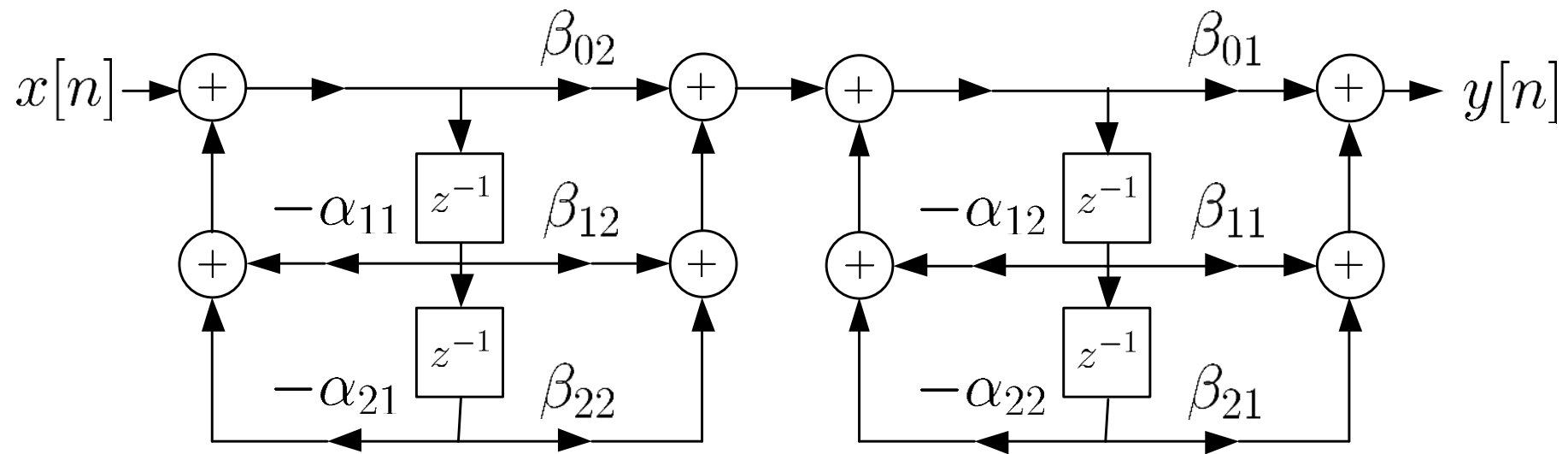
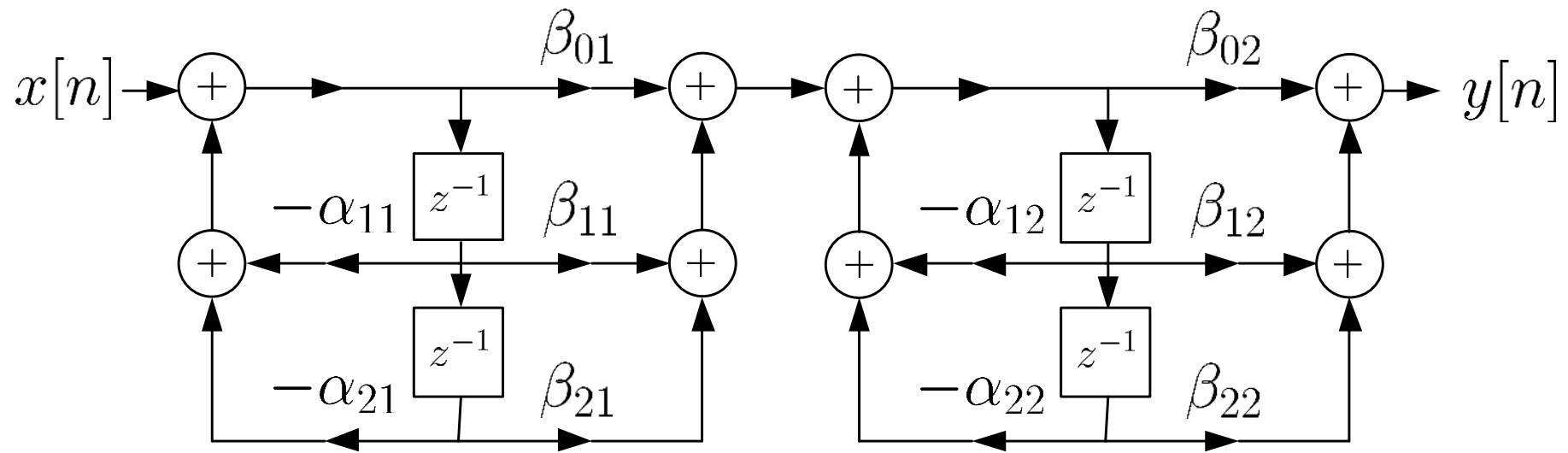
Each second-order subsystem

$$\frac{\beta_{0k} + \beta_{1k}z^{-1} + \beta_{2k}z^{-2}}{1 + \alpha_{1k}z^{-1} + \alpha_{2k}z^{-2}} \quad (9.30)$$

can be realized in either the direct or canonic form. Nevertheless, the canonic form is preferred because it requires the minimum number of delay elements.

In IIR filter implementation, we can group the numerator and denominator of (9.30) in different ways, leading to different pole and zero combinations in each of the second-order sections.

For example, there are four possible cascade realizations for a fourth-order IIR filter with $M = N = 4$.



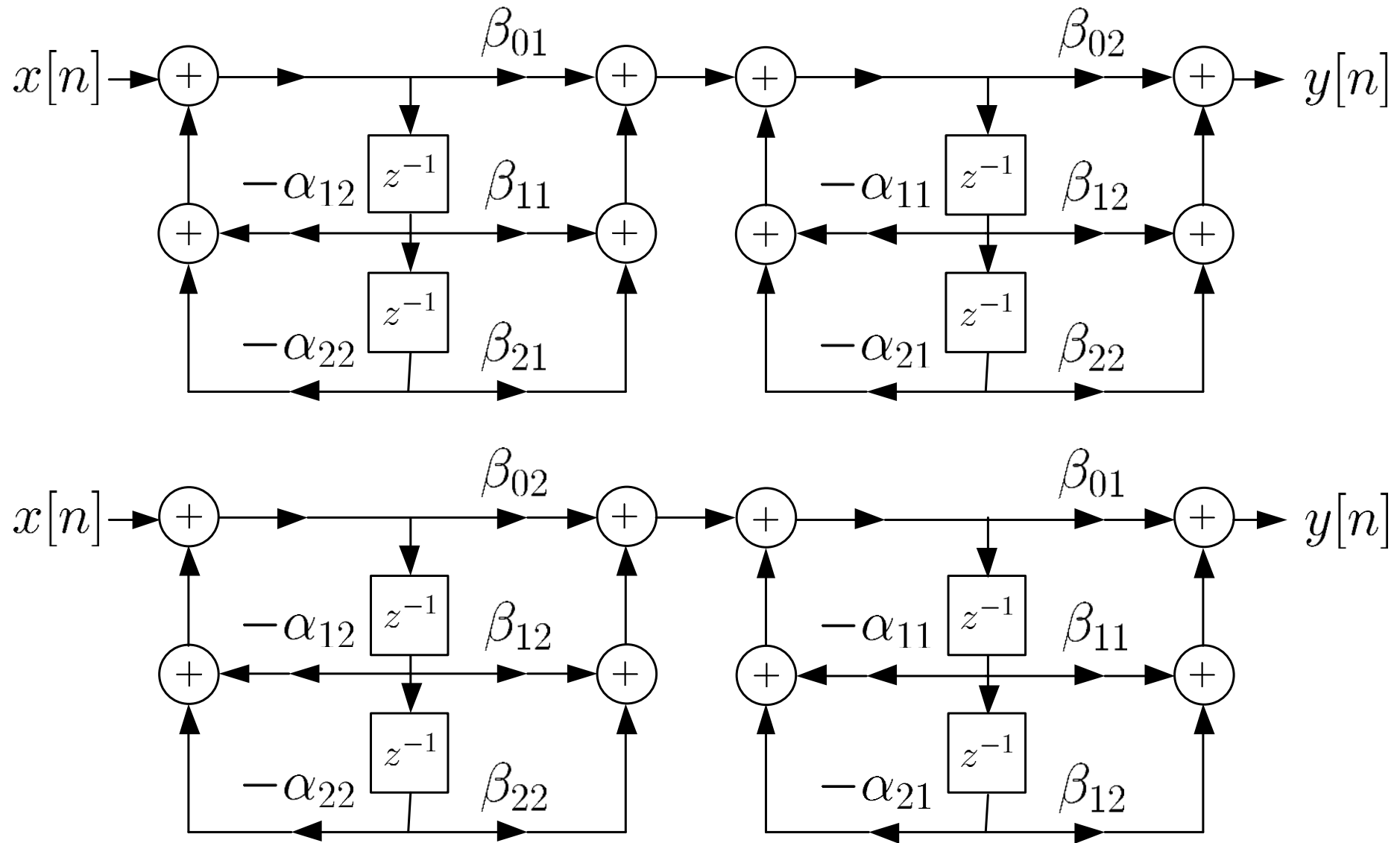


Fig.9.14: 4 possible cascade realizations for 4th-order IIR filter

To save the computational complexity, we express (9.29) as

$$H(z) = G \prod_{k=1}^{N_c} \frac{1 + \beta'_{1k} z^{-1} + \beta'_{2k} z^{-2}}{1 + \alpha_{1k} z^{-1} + \alpha_{2k} z^{-2}} \quad (9.31)$$

where $G = \beta_{01}\beta_{02} \cdots \beta_{0N_c}$, $\beta'_{1k} = \beta_{1k}/\beta_{0k}$ and $\beta'_{2k} = \beta_{2k}/\beta_{0k}$, $k = 1, 2, \cdots, N_c$.

Assuming that N is even with $N = M$, the cascade implementation of (9.31) needs N or $2N$ delay elements, $(2N + 1)$ multiplications and $2N$ additions, for computing each $y[n]$. That is, its memory and computational requirements are equal to those of the direct form.

Example 9.4

Draw the signal flow graph using the cascade form with first-order sections for the LTI system whose transfer function is:

$$H(z) = \frac{1 - 3z^{-1} + 2z^{-2}}{1 + 0.3z^{-1} - 0.1z^{-2}}$$

For each first-order section, canonic form is assumed.

Solving the quadratic equations of the numerator and denominator polynomials, we can factorize $H(z)$ as:

$$H(z) = \frac{(1 - 2z^{-1})(1 - z^{-1})}{(1 + 0.5z^{-1})(1 - 0.2z^{-1})}$$

There are four possible cascade forms for $H(z)$:

$$\begin{aligned}
 H(z) &= \frac{1 - 2z^{-1}}{1 + 0.5z^{-1}} \cdot \frac{1 - z^{-1}}{1 - 0.2z^{-1}} \\
 &= \frac{1 - z^{-1}}{1 + 0.5z^{-1}} \cdot \frac{1 - 2z^{-1}}{1 - 0.2z^{-1}} \\
 &= \frac{1 - 2z^{-1}}{1 - 0.2z^{-1}} \cdot \frac{1 - z^{-1}}{1 + 0.5z^{-1}} \\
 &= \frac{1 - z^{-1}}{1 - 0.2z^{-1}} \cdot \frac{1 - 2z^{-1}}{1 + 0.5z^{-1}}
 \end{aligned}$$

Note that although all four realizations are equivalent for infinite precision, they may differ in actual implementation when finite-precision numbers are employed.

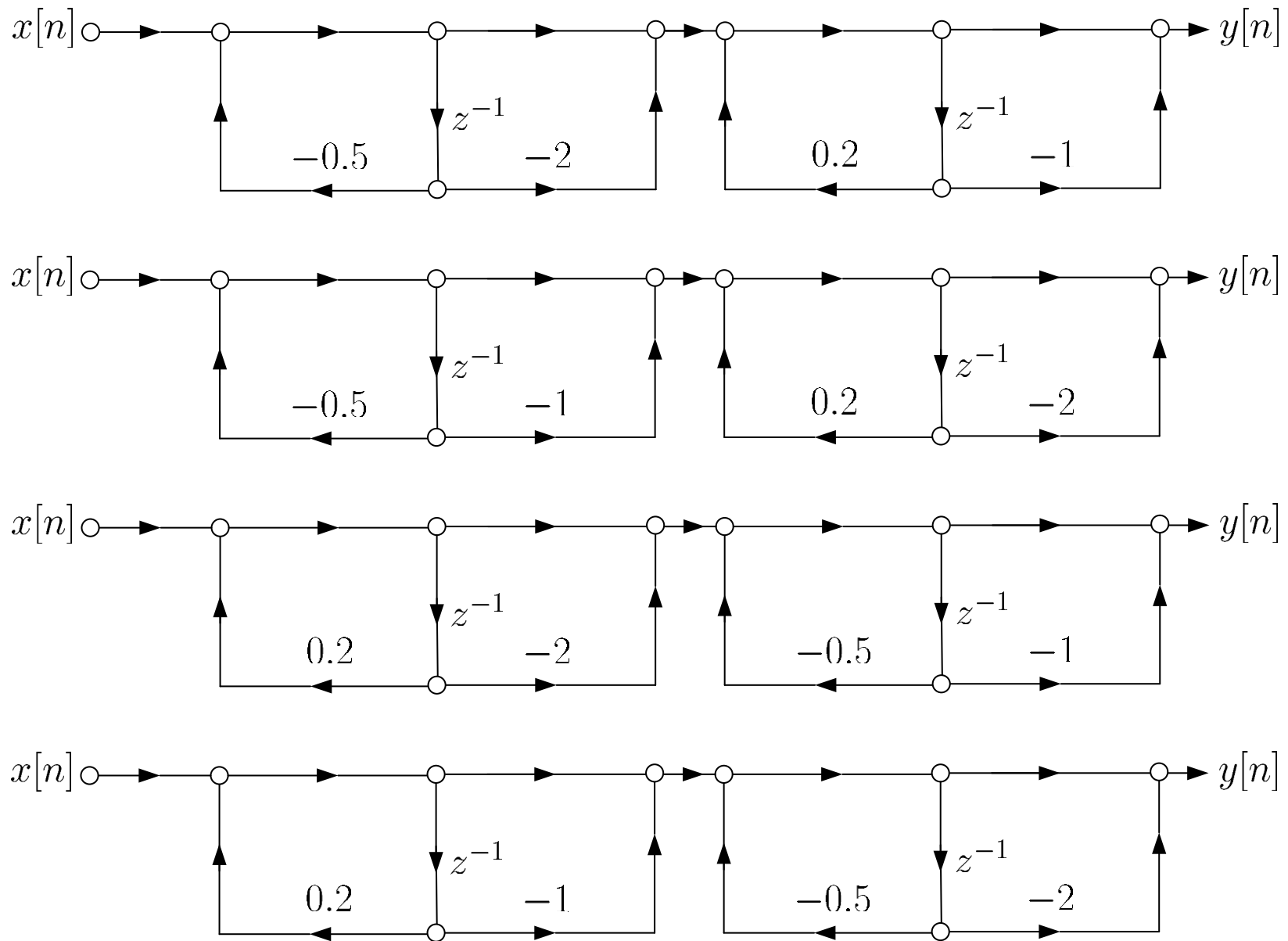


Fig.9.15: 4 possible cascade realizations with 1st-order sections

Example 9.5

Consider a LTI system whose transfer function $H(z)$ is:

$$H(z) = \frac{\left(1 - \frac{13}{10}z^{-1} + \frac{2}{5}z^{-2}\right) \left(1 - \frac{17}{12}z^{-1} + \frac{1}{2}z^{-2}\right)}{(1 - 3z^{-1} + 2z^{-2})(1 - 7z^{-1} + 12z^{-2})}$$

Find the number of possible combinations of cascade form with second-order sections. Determine if the system is stable or not.

Further factorizing $H(z)$ yields:

$$H(z) = \frac{\left(1 - \frac{1}{2}z^{-1}\right) \left(1 - \frac{2}{3}z^{-1}\right) \left(1 - \frac{3}{4}z^{-1}\right) \left(1 - \frac{4}{5}z^{-1}\right)}{(1 - z^{-1})(1 - 2z^{-1})(1 - 3z^{-1})(1 - 4z^{-1})}$$

For the numerator polynomials, there are 3 grouping possibilities in terms of second-order sections as follows:

$$\begin{array}{ll} \left(1 - \frac{1}{2}z^{-1}\right) \left(1 - \frac{2}{3}z^{-1}\right), & \left(1 - \frac{3}{4}z^{-1}\right) \left(1 - \frac{4}{5}z^{-1}\right) \\ \left(1 - \frac{1}{2}z^{-1}\right) \left(1 - \frac{3}{4}z^{-1}\right), & \left(1 - \frac{2}{3}z^{-1}\right) \left(1 - \frac{4}{5}z^{-1}\right) \\ \left(1 - \frac{1}{2}z^{-1}\right) \left(1 - \frac{4}{5}z^{-1}\right), & \left(1 - \frac{3}{4}z^{-1}\right) \left(1 - \frac{2}{3}z^{-1}\right) \end{array}$$

There are also 3 grouping possibilities for the denominator:

$$\begin{array}{ll} (1 - z^{-1}) (1 - 2z^{-1}), & (1 - 3z^{-1}) (1 - 4z^{-1}) \\ (1 - z^{-1}) (1 - 3z^{-1}), & (1 - 2z^{-1}) (1 - 4z^{-1}) \\ (1 - z^{-1}) (1 - 4z^{-1}), & (1 - 3z^{-1}) (1 - 2z^{-1}) \end{array}$$

Each fourth-order IIR filter can be realized in four possible cascade forms.

As a result, the number of possible cascade form combinations with second-order sections for $H(z)$ is $3 \cdot 3 \cdot 4 = 36$.

When implementing a filter, causality is always assumed because it is impossible to realize a noncausal system where the output depends on future input. It is clear that the region of convergence (ROC) for causal $H(z)$ is $|z| > 4$. Since the ROC does not include the unit circle, the system is not stable.

In summary, a causal system is stable when all poles are inside the unit circle.

4. Parallel Form

The idea of the parallel form is similar to the partial fraction expansion of z transform:

$$H(z) = \sum_{l=0}^{M-N} B_l z^{-l} + \sum_{k=1}^{N_c} \frac{\gamma_{0k} + \gamma_{1k} z^{-1}}{1 + \alpha_{1k} z^{-1} + \alpha_{2k} z^{-2}} \quad (9.32)$$

where $N_c = \lfloor (N+1)/2 \rfloor$. But now we use second-order sections in order to ensure all $\{\gamma_{0k}\}$, $\{\gamma_{1k}\}$, $\{\alpha_{1k}\}$ and $\{\alpha_{2k}\}$ are real.

Note that when $M < N$, the first summation term in (9.32) will not be included.

Example 9.6

Draw the block diagram using parallel form for a LTI system whose transfer function is:

$$H(z) = \frac{1 - 3z^{-1} + 2z^{-2}}{1 + 0.3z^{-1} - 0.1z^{-2}}$$

Following the long division as in Example 5.9, we obtain:

$$H(z) = -20 + \frac{21 + 3z^{-1}}{1 + 0.3z^{-1} - 0.1z^{-2}}$$

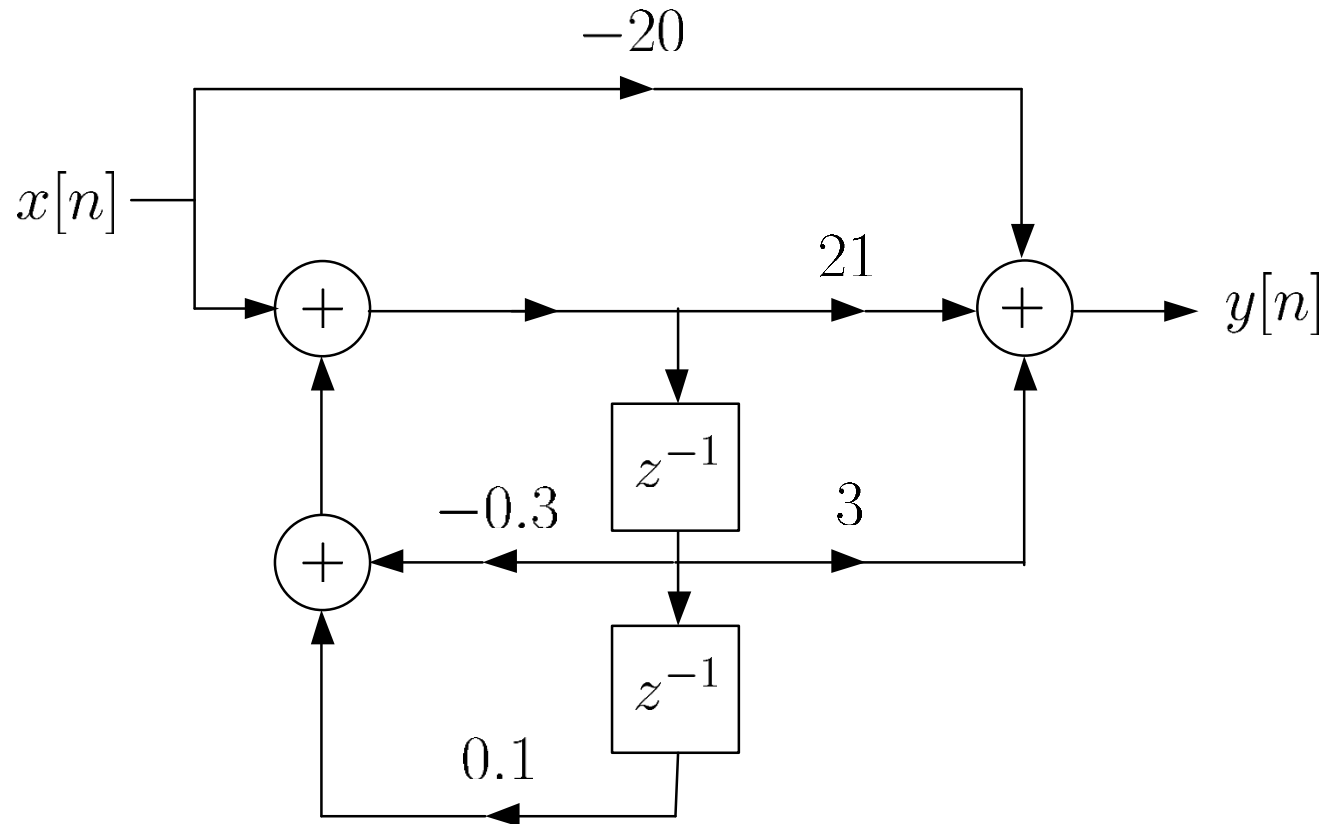


Fig.9.16: Parallel form with second-order section

As the poles of $H(z)$ are real, we can also express $H(z)$ in terms of first-order sections as:

$$H(z) = -20 + \frac{\frac{75}{7}}{1 + 0.5z^{-1}} + \frac{\frac{72}{7}}{1 - 0.2z^{-1}}$$

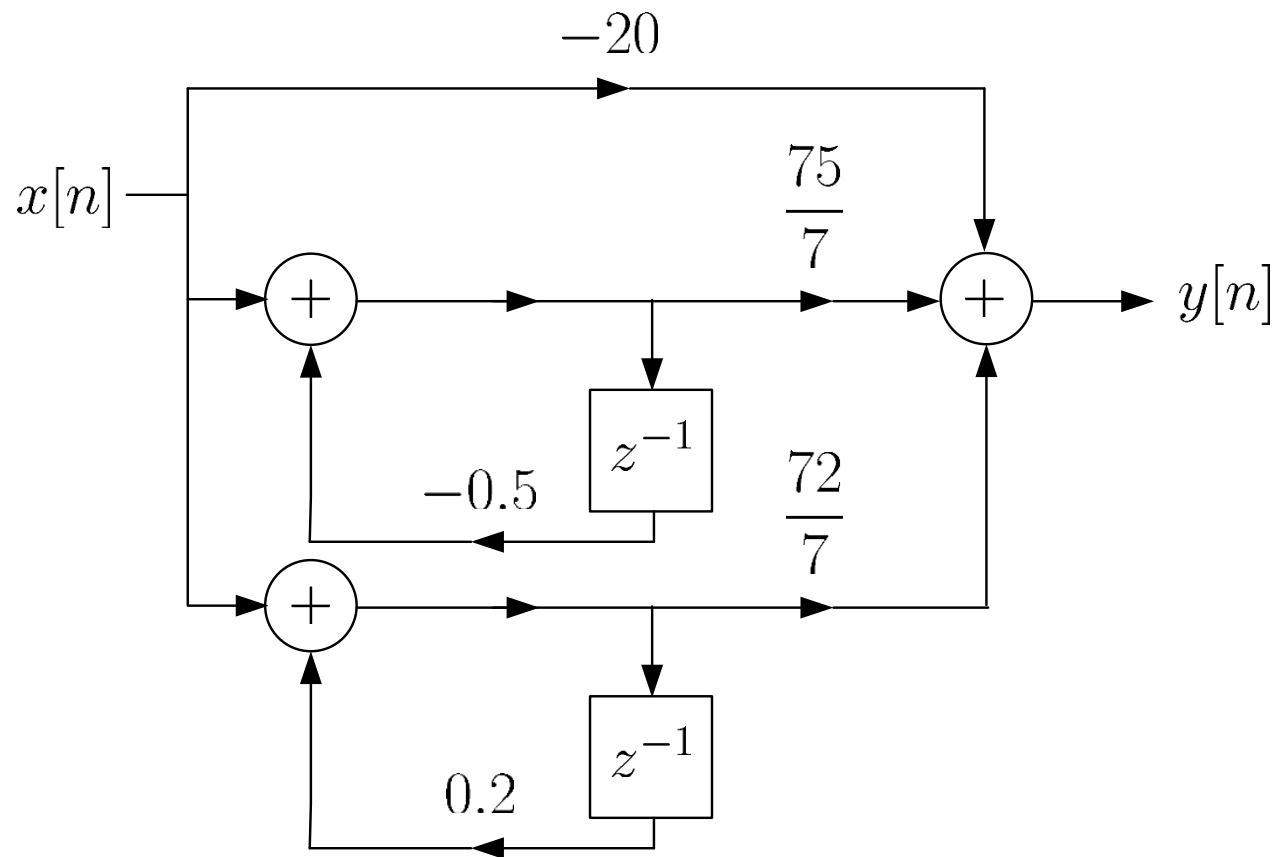


Fig.9.17: Parallel form with first-order sections

Example 9.7

Determine the transfer function $H(z)$ and the difference equation which relates $x[n]$ and $y[n]$ for

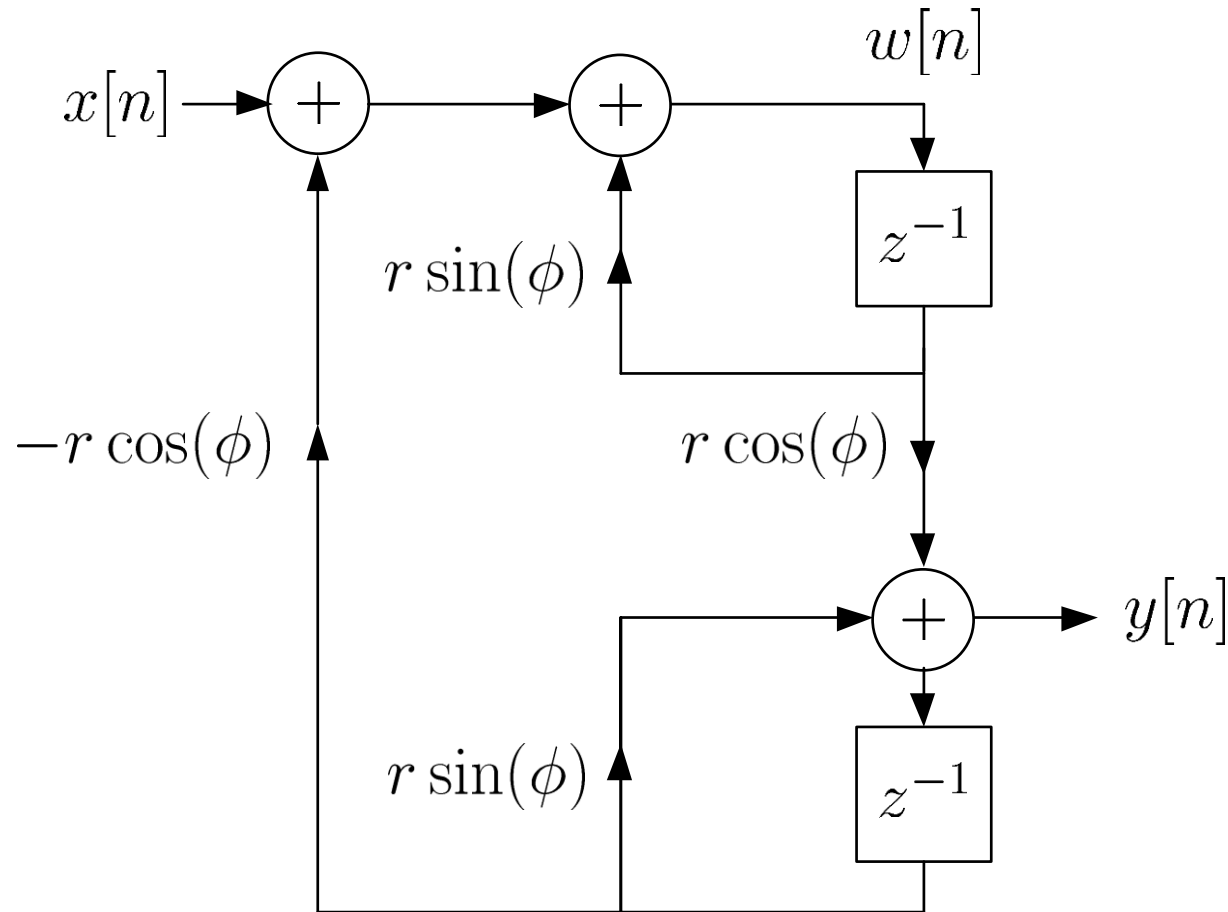


Fig.9.18: LTI system parameterized by r and ϕ

We first introduce an intermediate sequence $w[n]$ to relate $x[n]$ and $y[n]$. Then we can establish:

$$w[n] = w[n-1] \cdot r \sin(\phi) - y[n-1] \cdot r \cos(\phi) + x[n]$$

and

$$w[n-1] \cdot r \cos(\phi) + y[n-1] \cdot r \sin(\phi) = y[n]$$

Applying z transform yields:

$$W(z) = W(z)z^{-1}r \sin(\phi) - Y(z)z^{-1}r \cos(\phi) + X(z)$$

and

$$W(z)z^{-1}r \cos(\phi) + Y(z)z^{-1}r \sin(\phi) = Y(z)$$

From the second equation, we have:

$$W(z) = \frac{1 - z^{-1}r \sin(\phi)}{z^{-1}r \cos(\phi)} Y(z)$$

Substituting it into the first equation, we finally have:

$$\begin{aligned}\frac{1 - z^{-1}r \sin(\phi)}{z^{-1}r \cos(\phi)}Y(z) (1 - z^{-1}r \sin(\phi)) &= -Y(z)z^{-1}r \cos(\phi) + X(z) \\ \Rightarrow Y(z) (1 - 2z^{-1}r \sin(\phi) + z^{-2}r^2) &= X(z)z^{-1}r \cos(\phi) \\ \Rightarrow H(z) = \frac{Y(z)}{X(z)} &= \frac{z^{-1}r \cos(\phi)}{1 - 2z^{-1}r \sin(\phi) + z^{-2}r^2}\end{aligned}$$

Taking the inverse z transform gives:

$$y[n] - 2r \sin(\phi)y[n-1] + r^2y[n-2] = r \cos(\phi)x[n-1]$$

Comparison of Different Structures

The major factors that affect our choice of a specific realization are **computational complexity**, **memory requirement**, and **finite word-length effects**. Assuming that M is even with $M = N$:

Structure	Multiplication	Addition	Register
Direct form	$M + 1$	M	M
Cascade form	$M + 1$	M	M

Table 9.1: FIR filter structure comparison

Structure	Multiplication	Addition	Register
Direct form	$2M + 1$	$2M$	$2M$
Canonic form	$2M + 1$	$2M$	M
Cascade form	$2M + 1$	$2M$	M
Parallel form	$2M + 1$	$3M/2 + 1$	M

Table 9.2: IIR filter structure comparison

Computations of the direct form can be reduced if the FIR filter coefficients are symmetric or anti-symmetric.

When the filter coefficients are expressed using infinite precision numbers, all realizations are same. However, in practice, they are processed in registers which have finite word-lengths. In the presence of quantization errors, the cascade and parallel realizations are more robust than the direct and canonic forms, that is, they have frequency responses closer to the desired responses.

FIR filters are less sensitive than IIR filters to finite word-length effects.

For a feasible system, it should be causal and stable.

In cascade and parallel realizations of IIR filters, system stability can be easily monitored by checking pole locations.