Structures for Discrete-Time Systems

Chia-Ping Chen

Department of Computer Science and Engineering
National Sun Yat-sen University
Kaohsiung, Taiwan 800

Spring 2012

Contents

Introduction

At this point, we know that difference equations, impulse responses, and system functions are equivalent representations for LTI systems.

Consider the system described by the *system function*

$$H(z) = \frac{b_0 + b_1 z^{-1}}{1 - az^{-1}}, \quad |z| > a.$$

The *impulse response* is

$$h[n] = b_0 a^n u[n] + b_1 a^{n-1} u[n-1].$$

The first-order *difference equation* is

$$y[n] - ay[n-1] = b_0 x[n] + b_1 x[n-1].$$

Implementation

Since the system is IIR, it is impossible to implement the system with discrete convolution.

Instead, to implement this system, the difference equation can be employed as

$$y[n] = ay[n-1] + b_0 x[n] + b_1 x[n-1],$$

and the output can be recursively computed given certain initial condition, such as *initial rest* (i.e. $x[n], y[n]$ are causal).

Divide and Conquer

The implementation of a linear difference equation, expressed as

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

is based on 3 types of basic operations:

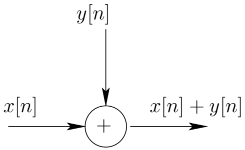
- addition of sequences
- multiplication of a sequence by a constant
- delay

We are going to learn how such a linear difference equation can be implemented through certain very basic structures.

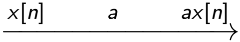
Basic Elements

We are going to represent a system represented as a linear difference equation by a diagram, so we first introduce the basic elements.

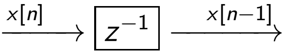
- addition of two sequences



- multiplication of a sequence by a constant



- delay

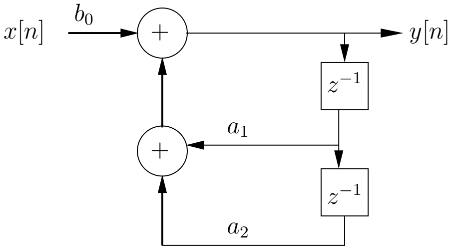


Block Diagram Representation

Consider a system with a second-order linear difference equation

$$y[n] = a_1 y[n-1] + a_2 y[n-2] + b_0 x[n].$$

The block diagram representation is as follows



Auxiliary Signal

A linear difference equation of order N can be expressed as

$$\begin{aligned} y[n] &= \sum_{k=1}^{N} a_k y[n-k] + \sum_{k=0}^{M} b_k x[n-k] \\ &= \sum_{k=1}^{N} a_k y[n-k] + v[n], \end{aligned}$$

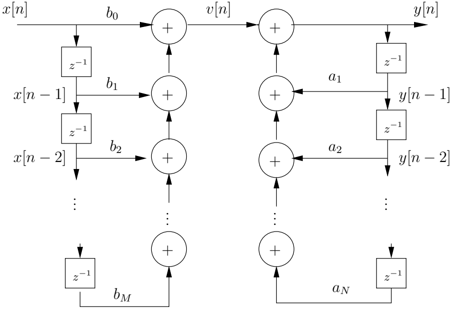
where $v[n]$ is called the **auxiliary signal** defined as

$$v[n] \triangleq \sum_{k=0}^{M} b_k x[n-k].$$

Repeating the equations as follows,

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

we see the overall system can be represented as a block diagram as follows:



System Function

In the z-domain, we have

$$V(z) = \left(\sum_{k=0}^{M} b_k z^{-k} \right) X(z) = H_1(z) X(z),$$

and

$$Y(z) = \left(\frac{1}{1 - \sum_{k=1}^{N} a_k z^{-k}} \right) V(z) = H_2(z) V(z),$$

where

$$H_1(z) = \sum_{k=0}^{M} b_k z^{-k}, \quad H_2(z) = \frac{1}{1 - \sum_{k=1}^{N} a_k z^{-k}}.$$

Alternative System Function

The z-transforms can also be expressed as

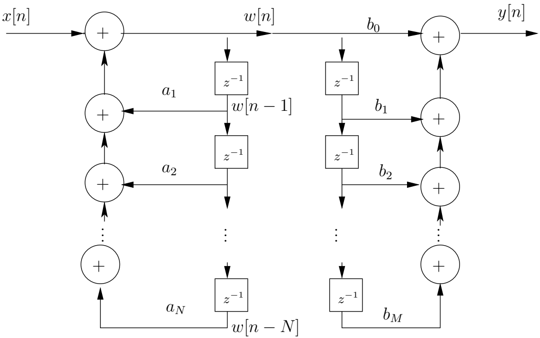
$$\begin{aligned} Y(z) &= H_2(z)(H_1(z)X(z)) \\ &= H_1(z)\left(H_2(z)X(z)\right) \\ &= H_1(z)W(z), \qquad W(z) \triangleq H_2(z)X(z). \end{aligned}$$

The corresponding difference equations are

$$\begin{aligned} w[n] &= \sum_{k=1}^{N} a_k w[n-k] + x[n], \\ y[n] &= \sum_{k=0}^{M} b_k w[n-k]. \end{aligned}$$

Block Diagram in Alternative Form

The alternative form suggests the following block diagram for the same input $x[n]$ and output $y[n]$, with an alternative auxiliary $w[n]$, as follows.



Basically, the two modules are swapped.

Direct Forms and Canonic Forms

The above implementations of a difference equation are called **direct forms**, as the coefficients are used directly in the block diagram.

- The first implementation, where the numerator (upstairs) is implemented before the denominator (downstairs), is called **direct form I**.
- The second implementation, where the denominator is implemented before the numerator, is called **direct form II**.

A **canonic form** implementation means that the minimum number of delay elements is used in the implementation. Therefore, the **direct form II** implementation is also called **canonic direct form**.

Example 6.2

Consider the system function

$$H(z) = \frac{1 + 2z^{-1}}{1 - 1.5z^{-1} + 0.9z^{-2}}.$$

Draw the block diagrams for direct form I and direct form II implementation.

Convert the system function to a linear difference equation

$$y[n] = 1.5y[n-1] - 0.9y[n-2] + x[n] + 2x[n-1],$$

and the rest is easy to see.

Signal Flow Graph

Definition (signal flow graph)

A **signal flow graph** is a network with the following semantics.

- Each node, say node k, is associated with a value, say $w_k[n]$;
- The branches (edges) are directed; branch (j, k) originates at node j and ends at node k; each branch has an input value and an output value; the input value of branch (j, k) is $w_j[n]$ and the output value depends on the transmittance of the branch (can be $aw_j[n]$ or $w_j[n-1]$);
- The value of node k is the sum of the branch output values entering node k.

Signal Flow Graph and Difference Equation

A signal flow graph is used to represent a difference equation.

- The input signal corresponds to a **source node**, which is a node without incoming edges.
- The output signal corresponds to a **sink node**, which is a node without outgoing edges.

Figure 6.9 gives an example. Note that

- Each node induces an equation, except for the source node;
- The number of equations is one less than the number of variables (node values);
- Thus, in principle, the output signal can be expressed as a function of the input signal.

Example

Consider Figure 6.10 (a), the block diagram representation of a first-order system.

The signal flow graph is depicted in (b). According to the semantics of signal flow graph, we have

$$\begin{cases} w_1[n] = x[n] + a w_4[n] \\ w_2[n] = w_1[n] \\ w_3[n] = b_0 w_2[n] + b_1 w_4[n] \\ w_4[n] = w_2[n-1] \\ y[n] = w_3[n] \end{cases} \xrightarrow{\mathcal{Z}} \begin{cases} W_1(z) = X(z) + a W_4(z) \\ W_2(z) = W_1(z) \\ W_3(z) = b_0 W_2(z) + b_1 W_4(z) \\ W_4(z) = z^{-1} W_2(z) \\ Y(z) = W_3(z) \end{cases}$$

It follows that

$$Y(z) = \frac{b_0 + b_1 z^{-1}}{1 - a z^{-1}} X(z).$$

System Function from Signal Flow Graph

Consider Example 6.3. A system not in a direct form is provided (Figure 6.12). What is the system function?

Although we can not write down system function immediately, we can go a round-about way guided by the network semantics. The equations at the nodes are

$$\begin{cases} w_1[n] = w_4[n] - x[n] \\ w_2[n] = \alpha w_1[n] \\ w_3[n] = w_2[n] + x[n] \\ w_4[n] = w_3[n-1] \\ y[n] = w_2[n] + w_4[n] \end{cases} \xrightarrow{\mathcal{Z}} \begin{cases} W_1(z) = W_4(z) - X(z) \\ W_2(z) = \alpha W_1(z) \\ W_3(z) = W_2(z) + X(z) \\ W_4(z) = z^{-1} W_3(z) \\ Y(z) = W_2(z) + W_4(z). \end{cases}$$

We can eliminate $W_1(z)$ and $W_3(z)$ and then solve for $W_2(z)$ and $W_4(z)$ in terms of $X(z)$. Finally, we have

$$Y(z) = \frac{z^{-1} - \alpha}{1 - \alpha z^{-1}} X(z).$$

(all-pass filter)

With this rational system function, the implementations in direct forms are not difficult to see, e.g. Figure 6.13.

Structures for IIR Systems

An IIR system is described by

$$y[n] - \sum_{k=1}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k],$$

where there $\exists$ k such that $a_k \neq 0$. It follows that

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

We henceforth use *signal flow graphs* to represent systems described by difference equations. In addition to **direct forms**, we introduce **cascade forms** and **parallel forms**.

Direct Forms

The direct forms represented by signal flow graphs for an IIR system are depicted in Figure 6.14 and Figure 6.15.

Using signal flow graphs is more succinct than using block diagrams.

Figure 6.16 and Figure 6.17 are the graphs of an instance of second-order LCCDE.

Factorization of System Functions

A rational system function with real coefficients can be factorized as

$$H(z) = A \frac{\prod_{k=1}^{M_1}(1 - f_k z^{-1}) \prod_{k=1}^{M_2}(1 - g_k z^{-1})(1 - g_k^* z^{-1})}{\prod_{k=1}^{N_1}(1 - c_k z^{-1}) \prod_{k=1}^{N_2}(1 - d_k z^{-1})(1 - d_k^* z^{-1})},$$

where $M = M_1 + 2M_2$ and $N = N_1 + 2N_2$ are the orders of the numerator (moving-average part) and of the denominator (auto-regression part) respectively.

c_k (resp. f_k) is a real pole (resp. zero), d_k, d_k^* (resp. g_k, g_k^*) are complex conjugate poles (resp. zeros).

Cascade Forms

The equation can be written as

$$H(z) = \prod_{k=1}^{N_s} \frac{b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2}}{1 - a_{1k} z^{-1} - a_{2k} z^{-2}},$$

where we assume $M \leq N$ and $N_s = \lfloor (N+1)/2 \rfloor$.

The multiplicative form of system function suggests a cascade structure of subsystems which are simple first-order and second-order systems. Such an implementation is called **cascade form**.

Figure 6.18 provides an example of a sixth-order system in the cascade structure of second-order subsystems in direct forms.

Implementation

Given a rational system function with N_s second-order factors, there are $(N_s!)^2$ equivalent implementations in the cascade forms.

- One $N_s!$ comes from the pairing of poles and zeros;
- The other $N_s!$ comes from the ordering of the subsystems.

These implementations are not equivalent in practice due to finite precision.

Trade-off

We have

$$H(z) = \prod_{k=1}^{N_s} \frac{b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2}}{1 - a_{1k} z^{-1} - a_{2k} z^{-2}} = b_0 \prod_{k=1}^{N_s} \frac{1 + \tilde{b}_{1k} z^{-1} + \tilde{b}_{2k} z^{-2}}{1 - a_{1k} z^{-1} - a_{2k} z^{-2}}.$$

Here we can see a trade-off

- With *fixed-point* arithmetic, it is a good idea to use 5-multiplier sections (left) to distribute the gains to reduce the accumulation of errors.
- With *floating-point* arithmetic, it is a good idea to use 4-multiplier sections (right) to reduce the number of multiplications.

Partial Fraction Expansion

A rational system function with real coefficients can be expressed as a partial fraction expansion

$$H(z) = \sum_{k=0}^{N_p} C_k z^{-k} + \sum_{k=1}^{N_1} \frac{A_k}{1 - c_k z^{-1}} + \sum_{k=1}^{N_2} \frac{B_k(1 - e_k z^{-1})}{(1 - d_k z^{-1})(1 - d_k^* z^{-1})},$$

where $N = N_1 + 2N_2, \quad N_p = M - N.$

Pairing the real poles, we have

$$H(z) = \sum_{k=0}^{N_p} C_k z^{-k} + \sum_{k=1}^{N_s} \frac{e_{0k} + e_{1k} z^{-1}}{1 - a_{1k} z^{-1} - a_{2k} z^{-2}}.$$

Note that the coefficents are real.

Parallel Forms

The additive form of system function suggests a parallel structure of subsystems which are simple first-order and second-order systems. Such an implementation is called **parallel form**.

A few examples are helpful.

- Figure 6.20 is an illustration of a sixth-order system in the parallel structure of second-order subsystems in direct forms.
- Example 6.6 is an instance of a second-order system which is represented in the parallel structures of first-order or second-order subsystems.

Feedback Loops

A **feedback loop** in a signal flow graph is a closed loop that begin with a node and return to the node by traversing branches only in the direction of the arrowheads.

An instance of feedback loop is given in Figure 6.23(a).The signal flow graph corresponds to the difference equation

$$y[n] = ay[n-1] + x[n].$$

For a system to be IIR, there must be feedback loop(s) in its signal flow graph.

Non-Computable Network

A network is **non-computable** if there exists a feedback loop without delay.

Figure 6.23(c) gives a simple example with

$$y[n] = ay[n] + x[n].$$

A non-computable network does not mean that the corresponding input/output difference equation cannot be solved. It simply means that the node variables cannot be solved successively.

Transposed Forms

Definition (transposition)

The **transposition**, or **flow graph reversal**, of a signal flow graph is the signal flow graph obtained as follows.

- Reverse the directions of all branches;
- Keep the branch transmittances;
- Reverse the roles of input and output;

Transposition Theorem

Theorem (transposition)

For a single-input, single-output system, a signal flow graph and its transposition have the same system function.

A few examples are helpful.

- The signal flow graph and its transposition of a first-order IIR system,

$$H(z) = \frac{1}{1 - az^{-1}},$$

  as well as their equivalency, are provided in Figures 6.24 – 6.26.
- In Figure 6.27 and Figure 6.28, the direct form II of a second-order system and its transposition (called transposed direct form II) are illustrated.

FIR Systems

For an FIR system, the coefficients a_k's vanish,

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

It is the discrete convolution of $x[n]$ and the impulse response

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

Direct Forms

The *direct form* of an FIR system is illustrated in Figure 6.31.

The structure is also called *tapped delay line structure* due to the chain of delay elements at the top.

The *transposed direct form* is illustrated in Figure 6.32.

Cascade Forms

The system function of an FIR system can be factorized as follows:

$$H(z) = \sum_{n=0}^{M} h[n]z^{-n} = \prod_{k=1}^{M_s} \left(b_{0k} + b_{1k}z^{-1} + b_{2k}z^{-2}\right),$$

where $M_s = \lfloor (M+1)/2 \rfloor$.

It suggests a *cascade structure*, as illustrated in Figure 6.33.

FIR Systems with Generalized Linear Phases

For an FIR system with generalized linear phase, we have

$$h[n] = \pm h[M-n], \quad n = 0, 1, \ldots, M.$$

Due to symmetry, the implementation can be simplified.

- The direct form of Type I is shown in Figure 6.34.
- The direct form of Type II is shown in Figure 6.35.
- Similarly for Type III and Type IV.