ECE503: Digital Signal Processing Lecture 6

D. Richard Brown III

WPI

20-February-2012

Lecture 6 Topics

1. Filter structures overview
2. FIR filter structures
   - Direct form
   - Direct form transposed
   - Symmetry tricks
   - Cascaded
   - Polyphase/parallel
3. IIR filter structures
   - Direct form I
   - Direct form II
   - DFI/DFII transposed
   - Cascaded
   - Parallel
4. Cascaded Lattice Structures
5. Computational Complexity

Digital Filter Structures

The problem we now consider is **how to actually implement discrete-time filters/systems**. Some possibilities:

1. Compute output by convolving input buffer with impulse response?

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

2. Directly compute input-output difference equation?

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

3. Other methods?

Our scope will be restricted to LTI causal real-coefficient FIR and IIR discrete-time filters.

Motivation

All filter structures give the same output when we have infinite-precision math. There can be big differences, however, when implementing a filter with finite precision (Chapter 12).

Example: Causal IIR bandpass filter

$y[n] = 0.01x[n] - 0.01x[n-2] + 0.0031y[n-1] - 0.98y[n-2]$

$$H(z) = \frac{0.01(1-z^{-2})}{1 - 0.0031z^{-1} + 0.98z^{-2}}$$

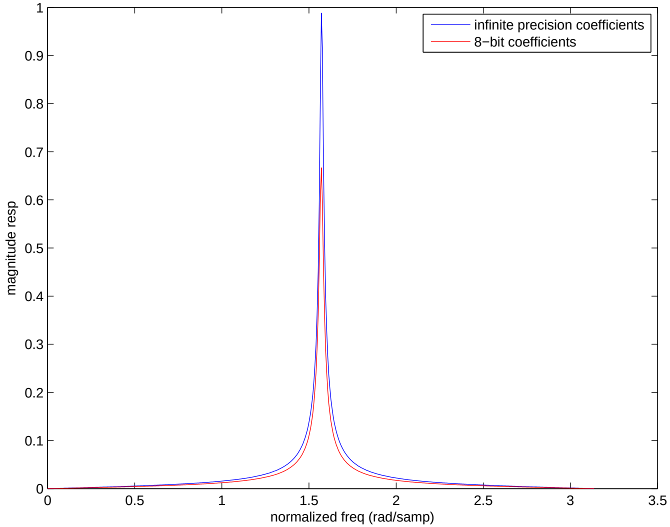
If we implement this filter on a small microcontroller with 8-bit integer datatypes, the quantized difference equation will be

$$y[n] = \frac{1}{128}\left\{x[n] - x[n-2] + 0y[n-1] - 125y[n-2]\right\}$$

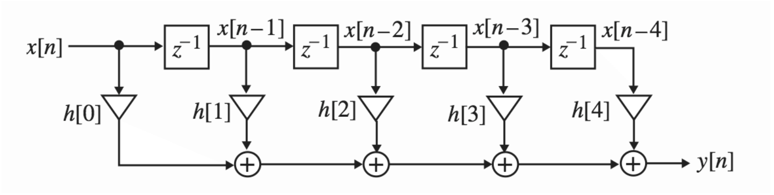
which gives a transfer function of

$$H_q(z) = \frac{0.078(1-z^{-2})}{1 + 0.9766z^{-2}}$$

Finite-Precision IIR BPF



FIR Direct Form



This is called "direct form" because it is a direct implementation of the convolution operation. The number of delays is equal to the order of the filter, hence this structure is **canonic**.

Note the fundamental building blocks:

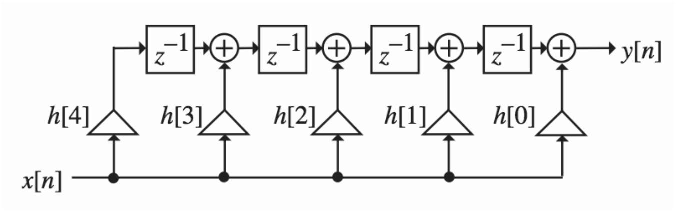
- Unit delay z^{-1}
- Multiplier/gain
- Accumulator/summer/adder
- Pick-off node

FIR Direct Form Transposed

In general (not just for FIR filters), we can form an equivalent structure by performing the **transpose operation**, which is defined as follows:

1. Reverse all paths
2. Replace pick-off nodes with accumulators, and vice-versa
3. Interchange input and output

If we perform this transpose operation on a direct form FIR filter, we get the "direct form transposed" FIR filter structure:



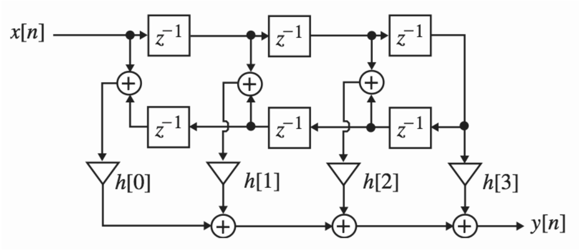
Not difficult to confirm this realizes the same difference equation as the direct form structure. Also canonic.

Linear Phase FIR Symmetry Tricks

Recall the various types of linear phase FIR filters, each having either symmetric or anti-symmetric impulse response. For example:

$$\begin{aligned} h[n] =& h[0]\delta[n] + h[1]\delta[n-1] + h[2]\delta[n-2] \\ &+ h[3]\delta[n-3] + h[2]\delta[n-4] + h[1]\delta[n-5] + h[0]\delta[n-6] \\ =& h[0](\delta[n] + \delta[n-6]) + h[1](\delta[n-1] + \delta[n-5]) \\ &+ h[2](\delta[n-2] + \delta[n-4]) + h[3]\delta[n-3] \end{aligned}$$

This symmetry can be exploited to reduce the number of multipliers.

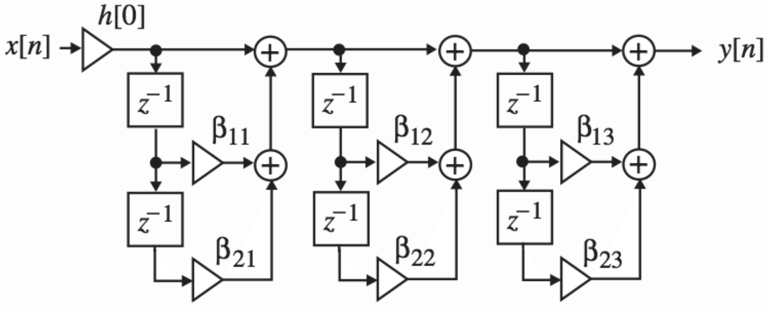


FIR Cascaded

Idea: Factor transfer function into product of smaller transfer functions.

Example:

$$\begin{aligned} H(z) &= h[0] + h[1]z^{-1} + h[2]z^{-2} + h[3]z^{-3} + h[4]z^{-4} + h[5]z^{-5} + h[6]z^{-6} \\ &= h[0](1 + \beta_{1,1}z^{-1} + \beta_{2,1}z^{-2})(1 + \beta_{1,2}z^{-1} + \beta_{2,2}z^{-2})(1 + \beta_{1,3}z^{-1} + \beta_{2,3}z^{-2}) \end{aligned}$$



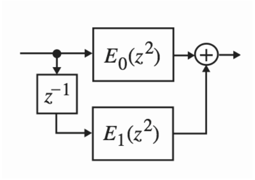
Why do we factor into second-order rather than first-order sections?

Canonic?

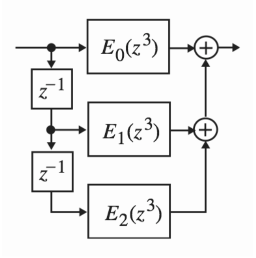
FIR Polyphase

We can also develop equivalent structures by putting smaller filters in parallel. One way to do this is called "polyphase realization":

2-branch polyphase decomposition



3-branch polyphase decomposition



This approach is not canonic unless the subfilters share delays (see Figure 8.8 in your textbook).

These structures are useful for computationally efficient realizations of multirate signal processing systems.

FIR Polyphase Examples

Example (2-branch):

$$\begin{aligned} H(z) &= h[0] + h[1]z^{-1} + h[2]z^{-2} + h[3]z^{-3} + h[4]z^{-4} + h[5]z^{-5} \\ &= (h[0] + h[2]z^{-2} + h[4]z^{-4}) + z^{-1}(h[1] + h[3]z^{-2} + h[5]z^{-4}) \\ &= E_0(z^2) + z^{-1}E_1(z^2) \end{aligned}$$

where $E_0(z) = h[0] + h[2]z^{-1} + h[4]z^{-2}$ and $E_1(z) = h[1] + h[3]z^{-1} + h[5]z^{-2}$.

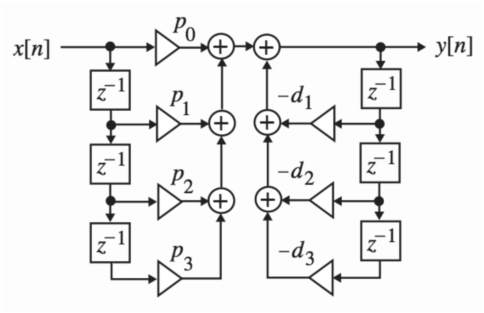
Same example except 3-branch:

$$\begin{aligned} H(z) &= h[0] + h[1]z^{-1} + h[2]z^{-2} + h[3]z^{-3} + h[4]z^{-4} + h[5]z^{-5} \\ &= (h[0] + h[3]z^{-3}) + z^{-1}(h[1] + h[4]z^{-3}) + z^{-2}(h[2] + h[5]z^{-3}) \\ &= E_0(z^3) + z^{-1}E_1(z^3) + z^{-2}E_2(z^3) \end{aligned}$$

where $E_0(z) = h[0] + h[3]z^{-1}$, and $E_1(z) = h[1] + h[4]z^{-1}$, and $E_2(z) = h[2] + h[5]z^{-1}$.

IIR Direct Form I

If $H(z) = \frac{P(z)}{Q(z)}$, we can implement this directly as a cascade of the feedforward part $P(z)$ and the feedback part $Q(z)$.

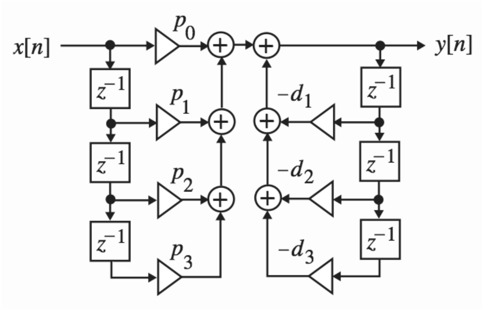


Note textbook uses $p_0, \ldots, p_{M-1}$ for numerator coefficients and $d_0, \ldots, d_{N-1}$ for denominator coefficients. We typically use $b_0, \ldots, b_{M-1}$ for numerator coefficients and $a_0, \ldots, a_{N-1}$ for denominator coefficients (consistent with Matlab). Is this structure canonic?

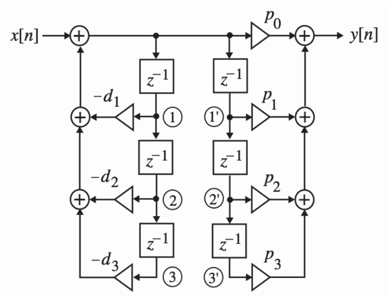
Changing the Order of the IIR DFI Sections

To get equivalent structures, we can perform a transpose or we can reverse the order of the sections (do feedback first, feedforward second):

IIR DFI



IIR-DFI swapped sections



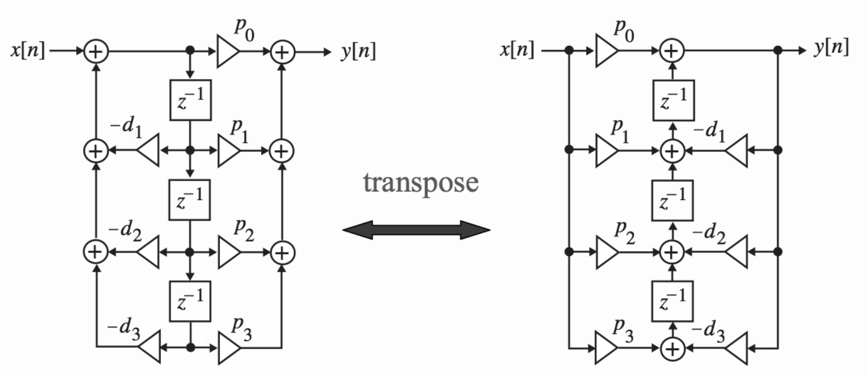
Canonic?

Note the delays are no longer storing past inputs and past outputs.

What can we say about the contents at pickoff node ℓ with respect to the contents at pickoff node ℓ'?

IIR Direct Form II

By sharing the delays in our swapped sections structure, we arrive at a structure called IIR Direct Form II (its transpose is also shown below).



Canonic?

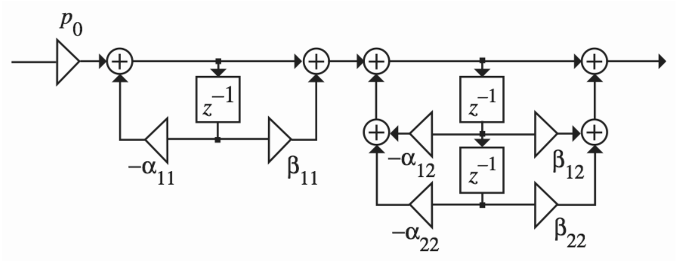
Pseudocode Implementation of IIR Direct Form II

```
<read in new input x[n] from ADC>

<shift intermediate values down, starting at bottom
  u[n-M+1] = u[n-M+2]
  u[n-M+2] = u[n-M+3]
  ...
>

<compute new intermediate value
  u[n] = x[n] - d[1]*u[n-1] - d[n]*u[n-2] - ...
>

<compute new output
  y[n] = p[0]*u[n] + p[1]*u[n-1] + p[2]*u[n-2] + ...
>

<write new ouput y[n] to DAC>
```

IIR Cascaded Direct Form II

Same idea as before: Factor transfer function and implement as a series/product of smaller cascaded transfer functions. Example:

$$H(z) = \frac{p_0 + p_1 z^{-1} + p_2 z^{-2} + p_3 z^{-3}}{1 + d_1 z^{-1} + d_2 z^{-2} + d_3 z^{-3}}$$
$$= \left(\frac{p_0(1+\beta_{1,1} z^{-1})}{1+\alpha_{1,1} z^{-1}}\right)\left(\frac{1+\beta_{1,2} z^{-1} + \beta_{2,2} z^{-2}}{1+\alpha_{1,2} z^{-1} + \alpha_{2,2} z^{-2}}\right)$$



Both sections are implemented here as DF II. Canonic?

IIR Cascaded Direct Form II

Suppose we have the factorization

$$H(z) = \frac{P(z)}{D(z)} = \frac{P_1(z)P_2(z)P_3(z)}{D_1(z)D_2(z)D_3(z)}$$

Section ordering and pole-zero pairings can affect finite-precision behavior of system. We'll look at this more closely when we cover Chapter 12.

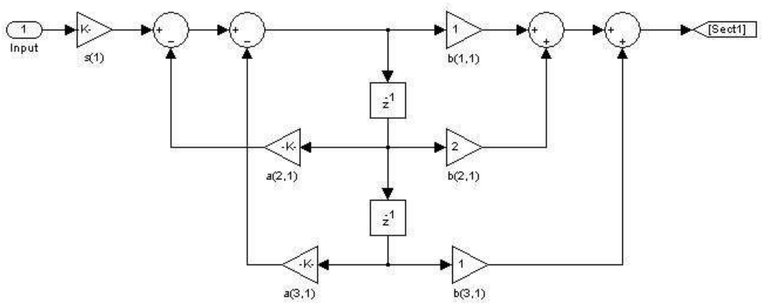
$$\begin{tikzcd}
{} \arrow[r] & \boxed{\frac{P_1(z)}{D_1(z)}} \arrow[r] & \boxed{\frac{P_2(z)}{D_2(z)}} \arrow[r] & \boxed{\frac{P_3(z)}{D_3(z)}} \arrow[r] & {} & {} \arrow[r] & \boxed{\frac{P_1(z)}{D_2(z)}} \arrow[r] & \boxed{\frac{P_2(z)}{D_3(z)}} \arrow[r] & \boxed{\frac{P_3(z)}{D_1(z)}} \arrow[r] & {} \\
{} \arrow[r] & \boxed{\frac{P_1(z)}{D_3(z)}} \arrow[r] & \boxed{\frac{P_2(z)}{D_1(z)}} \arrow[r] & \boxed{\frac{P_3(z)}{D_2(z)}} \arrow[r] & {} & {} \arrow[r] & \boxed{\frac{P_1(z)}{D_2(z)}} \arrow[r] & \boxed{\frac{P_2(z)}{D_1(z)}} \arrow[r] & \boxed{\frac{P_3(z)}{D_3(z)}} \arrow[r] & {} \\
{} \arrow[r] & \boxed{\frac{P_1(z)}{D_1(z)}} \arrow[r] & \boxed{\frac{P_2(z)}{D_3(z)}} \arrow[r] & \boxed{\frac{P_3(z)}{D_2(z)}} \arrow[r] & {} & {} \arrow[r] & \boxed{\frac{P_1(z)}{D_3(z)}} \arrow[r] & \boxed{\frac{P_2(z)}{D_2(z)}} \arrow[r] & \boxed{\frac{P_3(z)}{D_1(z)}} \arrow[r] & {}
\end{tikzcd}$$

$$\begin{tikzcd}
{} \arrow[r] & \boxed{\frac{P_1(z)}{D_1(z)}} \arrow[r] & \boxed{\frac{P_2(z)}{D_2(z)}} \arrow[r] & \boxed{\frac{P_3(z)}{D_3(z)}} \arrow[r] & {} & {} \arrow[r] & \boxed{\frac{P_1(z)}{D_1(z)}} \arrow[r] & \boxed{\frac{P_3(z)}{D_3(z)}} \arrow[r] & \boxed{\frac{P_2(z)}{D_2(z)}} \arrow[r] & {} \\
{} \arrow[r] & \boxed{\frac{P_2(z)}{D_2(z)}} \arrow[r] & \boxed{\frac{P_1(z)}{D_1(z)}} \arrow[r] & \boxed{\frac{P_3(z)}{D_3(z)}} \arrow[r] & {} & {} \arrow[r] & \boxed{\frac{P_2(z)}{D_2(z)}} \arrow[r] & \boxed{\frac{P_3(z)}{D_3(z)}} \arrow[r] & \boxed{\frac{P_1(z)}{D_1(z)}} \arrow[r] & {} \\
{} \arrow[r] & \boxed{\frac{P_3(z)}{D_3(z)}} \arrow[r] & \boxed{\frac{P_1(z)}{D_1(z)}} \arrow[r] & \boxed{\frac{P_2(z)}{D_2(z)}} \arrow[r] & {} & {} \arrow[r] & \boxed{\frac{P_3(z)}{D_3(z)}} \arrow[r] & \boxed{\frac{P_2(z)}{D_2(z)}} \arrow[r] & \boxed{\frac{P_1(z)}{D_1(z)}} \arrow[r] & {}
\end{tikzcd}$$

IIR Direct Form II Second Order Sections

When creating IIR filters in Matlab with `fdatool` this is the default filter structure. Why?

- Canonic (minimum number of delays/memory)
- Computationally efficient (minimum number of multiply/accumulates)
- Robust to coefficient quantization

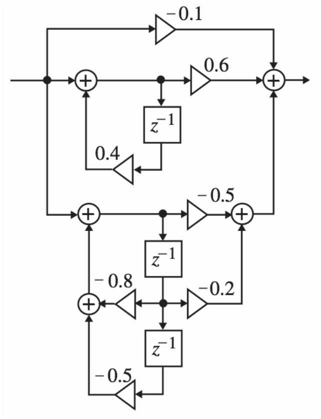


Some Matlab functions for working with DFII-SOS filter structures: `tf2sos`, `sos2tf`, `zp2sos`, `sos2zp`, and `fdatool`.

IIR Parallel Realization

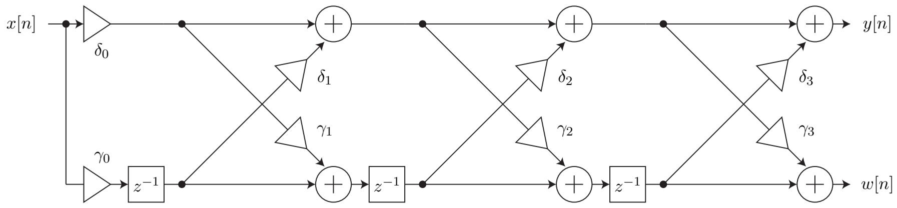
One way to realize IIR filters in parallel is to do a partial fraction expansion. Example:

$$H(z) = \frac{0.44z^{-1} + 0.362z^{-2} + 0.02z^{-3}}{1 + 0.4z^{1-} + 0.18z^{-2} - 0.2z^{-3}} = -0.1 + \frac{0.6}{1 - 0.4z^{-1}} + \frac{-0.5 - 0.2z^{-1}}{1 + 0.8z^{-1} + 0.5z^{-2}}$$



Each subfilter is DFII. Can be useful for dividing computation across processors/cores.

FIR Cascaded Lattice Structures

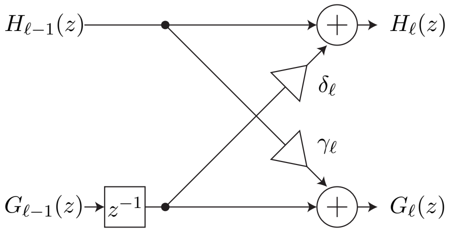


This approach can realize two FIR transfer functions $H(z) = \frac{Y(z)}{X(z)}$ and $G(z) = \frac{W(z)}{X(z)}$.

Given a lattice structure block diagram, how do we determine $H(z)$ and $G(z)$?

Given a desired FIR $H(z)$ and $G(z)$ (or $h[n]$ and $g[n]$), how do we determine the lattice coefficients?

FIR Cascaded Lattice Structures (Lattice → TF)



Analysis of single lattice segment:

$$
\begin{aligned}
H_\ell(z) &= H_{\ell-1}(z) + z^{-1}\delta_\ell G_{\ell-1}(z) \\
G_\ell(z) &= \gamma_\ell H_{\ell-1}(z) + z^{-1} G_{\ell-1}(z)
\end{aligned}
$$

Since $H_0(z) = \delta_0 X(z)$ and $G_0(z) = \gamma_0 X(z)$, this recursion can be applied for $\ell = 1, 2, \dots, N$ to determine the transfer functions $H(z)$ and $G(z)$ from the lattice coefficients.

FIR Cascaded Lattice Structures (TF $\rightarrow$ Lattice)

To get a recursion for computing lattice coefficients given $H(z)$ and $G(z)$, we can redo our analysis, solving instead for $H_{\ell-1}(z)$ and $G_{\ell-1}(z)$, to get

$$\begin{aligned} H_{\ell-1}(z) &= K_\ell \left[H_\ell(z) - \delta_\ell G_\ell(z) \right] \\ G_{\ell-1}(z) &= K_\ell z \left[G_\ell(z) - \gamma_\ell H_\ell(z) \right] \end{aligned}$$

where $K_\ell = \frac{1}{1-\delta_\ell \gamma_\ell}$.

Suppose, as an example, $\ell = 2$ with $H_\ell(z) = h_0 + h_1 z^{-1} + h_2 z^{-2}$ and $G_\ell(z) = g_0 + g_1 z^{-1} + g_2 z^{-2}$. Then

$$\begin{aligned} H_1(z) &= K_2 \left[h_0 + h_1 z^{-1} + h_2 z^{-2} - \delta_2 (g_0 + g_1 z^{-1} + g_2 z^{-2}) \right] \\ G_1(z) &= K_2 z \left[g_0 + g_1 z^{-1} + g_2 z^{-2} - \gamma_2 (h_0 + h_1 z^{-1} + h_2 z^{-2}) \right] \end{aligned}$$

Note that $H_1(z)$ and $G_1(z)$ can only be order 1 transfer functions. Hence we need the z^{-2} and z^{+1} terms to vanish, which happens when

$$\begin{aligned} z^{-2} \text{ term in } H_1(z) \text{ vanishes} \quad &\Leftrightarrow \quad h_2 - \delta_2 g_2 = 0 \quad \Leftrightarrow \quad \delta_2 = \frac{h_2}{g_2} \\ z^{+1} \text{ term in } G_1(z) \text{ vanishes} \quad &\Leftrightarrow \quad g_0 - \gamma_2 h_0 = 0 \quad \Leftrightarrow \quad \gamma_2 = \frac{g_0}{h_0} \end{aligned}$$

FIR Cascaded Lattice Structures (TF $\rightarrow$ Lattice)

Outline of procedure:

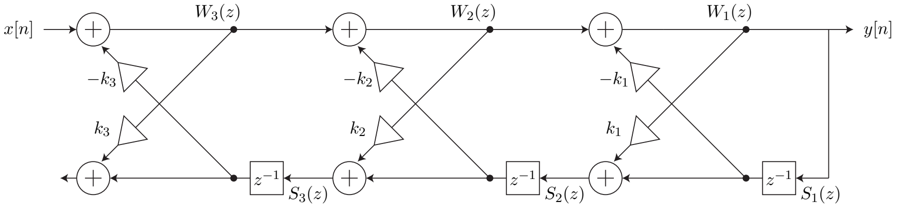
1. Set L equal to the number of lattice sections.
2. Set $H_L(z) = H(z)$ and $G_L(z) = G(z)$.
3. Set $\ell = L$.
4. Solve for δ_ℓ and γ_ℓ.
5. Compute $K_\ell = \frac{1}{1-\delta_\ell\gamma_\ell}$.
6. Compute $H_{\ell-1}(z)$ and $G_{\ell-1}(z)$. Both transfer functions should be order $\ell - 1$. Note coefficients are affected by K_ℓ (you can't just say $H_{\ell-1}(z)$ is equal to the first $\ell - 1$ coefficients of $H_\ell(z)$ or that $G_{\ell-1}(z)$ is equal to the first $\ell - 1$ coefficients of $G_\ell(z)$).
7. Decrement ℓ.
8. If $\ell \geq 1$ go to step 4.
9. The final gains at the input are simply $\delta_0 = H_0(z)$ and $\gamma_0 = G_0(z)$ (these are both zero-order transfer functions).

FIR Cascaded Lattice Structures (TF $\rightarrow$ Lattice)

Remarks and extensions:

1. There are some special cases where this recursion breaks down. See your textbook pp. 454-455 for workarounds.
2. The lattice structure is natural for creating complementary filter pairs. For example, if $H(z)$ and $G(z)$ are power complementary transfer functions, then $\delta_\ell = (-1)^\ell \gamma_\ell$.
3. There are other FIR lattice structures. For example, you can formulate a single output version with $\delta_0 = \gamma_0 = 1$ and symmetric lattice coefficients ($\delta_\ell = \gamma_\ell$). Similar recursions can be used to determine the lattice coefficients.
4. Lattice structures have nice properties for adaptive filtering (ECE630) and also usually have good robustness to coefficient quantization (Chap 12).

All-Pole IIR Lattice Structures (1 of 2)



Analysis of single section:

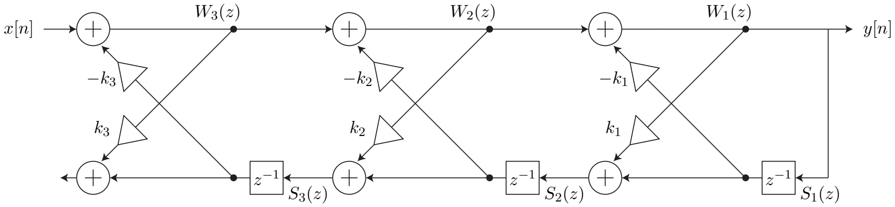
$$W_i(z) = W_{i+1}(z) - k_i z^{-1} S_i(z)$$
$$S_{i+1}(z) = k_i W_i(z) + z^{-1} S_i(z)$$

We can rearrange this to write

$$\begin{bmatrix} W_{i+1}(z) \\ S_{i+1}(z) \end{bmatrix} = \begin{bmatrix} 1 & k_i z^{-1} \\ k_i & z^{-1} \end{bmatrix} \begin{bmatrix} W_i(z) \\ S_i(z) \end{bmatrix}$$

Note $X(z) = W_4(z)$ and $Y(z) = W_1(z) = S_1(z)$. Continued

All-Pole IIR Lattice Structures (2 of 2)



We can apply the result from the previous slide to write for this example

$$\begin{bmatrix} X(z) \\ S_4(z) \end{bmatrix} = \begin{bmatrix} 1 & k_3 z^{-1} \\ k_3 & z^{-1} \end{bmatrix} \begin{bmatrix} 1 & k_2 z^{-1} \\ k_2 & z^{-1} \end{bmatrix} \begin{bmatrix} 1 & k_1 z^{-1} \\ k_1 & z^{-1} \end{bmatrix} \begin{bmatrix} Y(z) \\ Y(z) \end{bmatrix}.$$

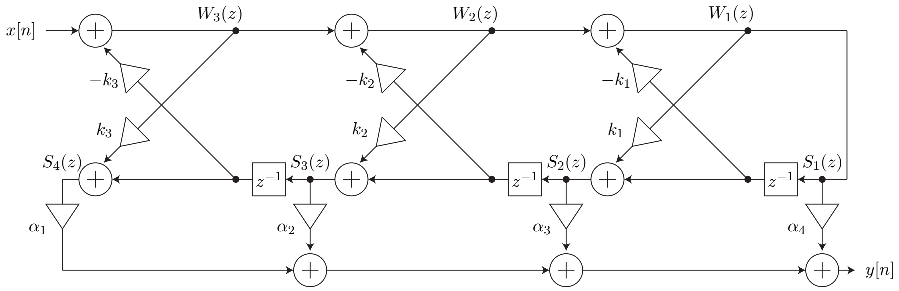
Since we only care about the relationship between $X(z)$ and $Y(z)$, we can do a little bit of algebra to write

$$X(z) = \left(1 + [k_1(1+k_2) + k_2 k_3]z^{-1} + [k_2 + k_1 k_3(1+k_2)]z^{-2} + k_3 z^{-3}\right) Y(z)$$

Hence $H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1+d_1 z^{-1} + d_2 z^{-2} + d_3 z^{-3}}$ is an all-pole transfer function.

A recursion similar to the FIR case can be found to determine the lattice coefficients from an all-pole transfer function (see Section 8.8.1 of your textbook).

General IIR Lattice Structures



Note $Y(z) = \alpha_1 S_4(z) + \alpha_2 S_3(z) + \alpha_3 S_2(z) + \alpha_4 S_1(z)$.

Similar analysis as before can be applied to determine the TF from the lattice. See also Matlab function `latc2tf`.

To get the lattice coefficients from the TF, you can use the Gray-Markel method (see Section 8.8.2) of your textbook. See also Matlab function `tf2latc`.

Conclusions

1. This concludes Chapter 8. You are responsible for all of the material in this chapter except Sections 8.7 (Parametrically Tunable Low-Order IIR Digital Filter Pairs) and 8.11 (Tunable High-Order Digital Filter Pairs), even if it wasn't covered in lecture.
2. Please read Chapter 9 before the next lecture and have some questions prepared.
3. The next lecture is on Monday 12-March-2012 at 6pm.
4. The midterm exam is scheduled for Monday 27-Feb-2012 and is based on Chapters 1-7 of your textbook. No homework is due on Monday 27-Feb-2012 or Monday 12-March-2012.