Implementation of discrete-time systems

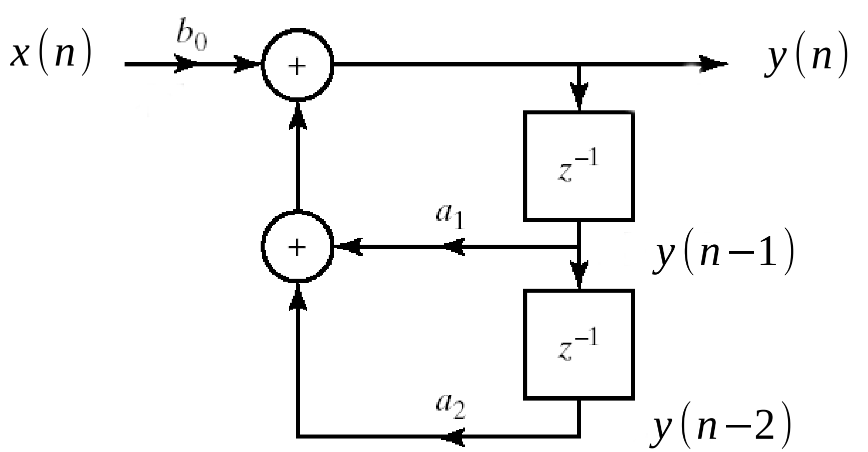
- Block diagram representations of LTI discrete systems
- Direct form realizations of LTI discrete systems
- Basic structures for IIR systems: cascade form
- Basic structures for IIR systems: parallel form
- Comparison of structures
- Transposing
- Second order modules for discrete-time systems
- Basic structures of FIR systems: direct forms
- Basic structures of FIR systems: cascade form
- (Additional materials)

Block diagram representation of LTI discrete systems

- Transfer function and difference equation are equivalent descriptions of an linear time-invariant discrete system
- Example

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

$$y(n) = a_1 y(n-1) + a_0 y(n-2) + b_0 x(n)$$



Block diagram representation of LTI discrete systems

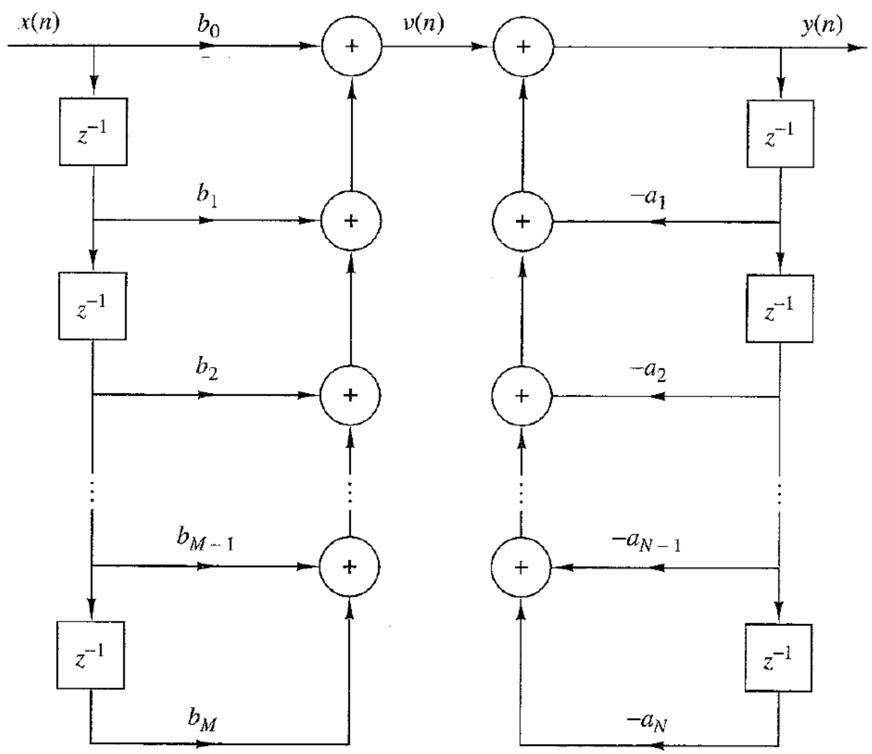
- Direct-form realizations of discrete systems
  - Direct form I
  - Direct form II
- Indirect-form realizations of discrete systems
  - Cascade form
  - Parallel form

Direct form realizations of LTI discrete systems

- Direct form I
- The cascade of a non-recursive system and a recursive system

It requires
M+N memory cells
M+N additions
M+N+1 multiplications

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



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

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

[Proakis, Manolakis]

Direct form realizations of LTI discrete systems

- Direct form II

  (delay lines merge)

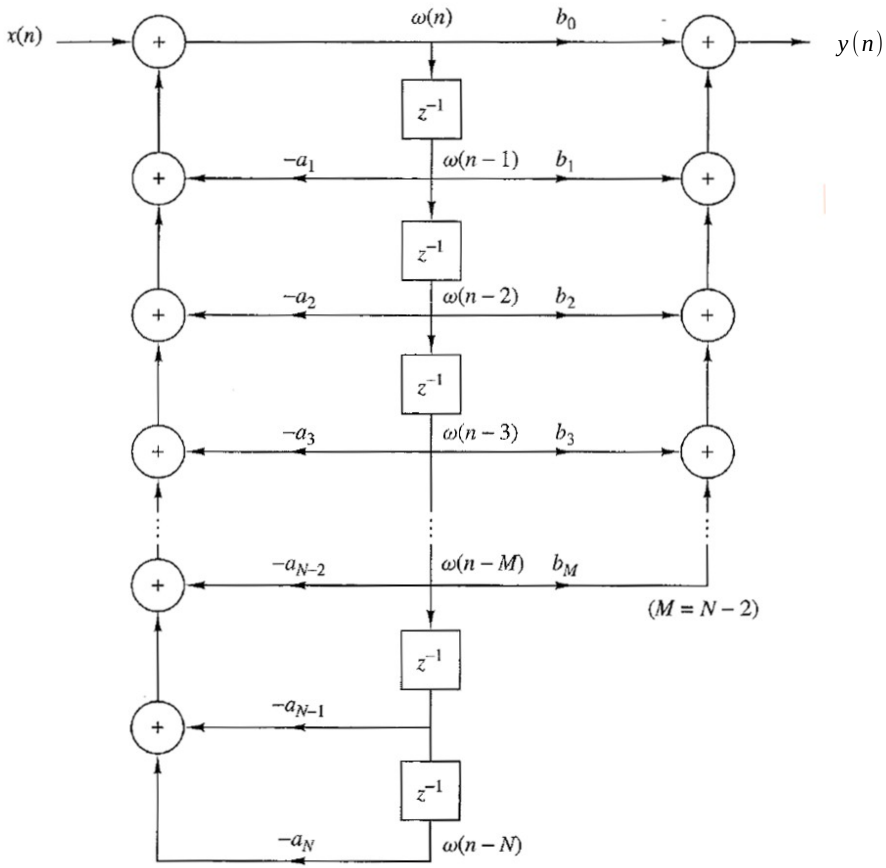
- The cascade of a recursive system and a non-recursive system

  - It requires
    max(*N*,*M*) memory cells
    N+*M* additions
    N+*M*+1 multiplications

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

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

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



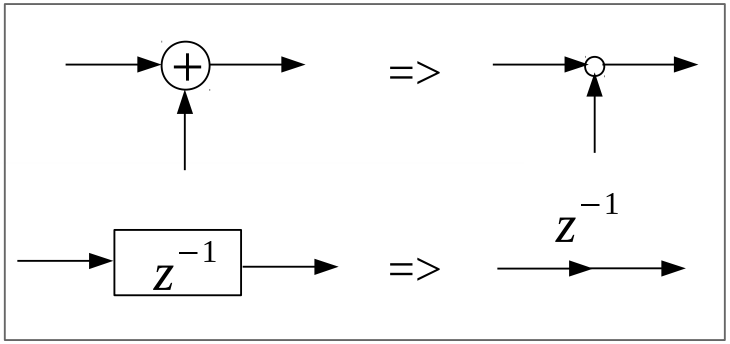
[Proakis, Manolakis]

Direct form realizations of LTI discrete systems

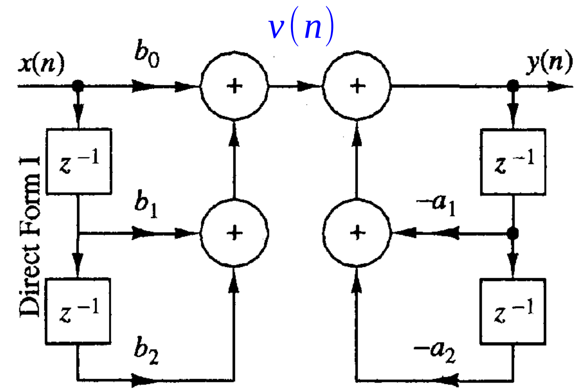
- Theoretically there is no difference between direct form I and II
- Both are
  - Simple
  - Visible from the difference equation
- Direct form I
  - Sensitive to quantization error (less in comparison to the form II since it implements the non-recursive part prior to the recursive part)
  - Number of elements is not minimum
- Direct form II
  - Sensitive to quantization error (more in comparison the the form I since it implements the recursive part prior to the non-recursive part)
  - Number of memory cells is minimum (canonical form)

Direct form realizations of LTI discrete systems

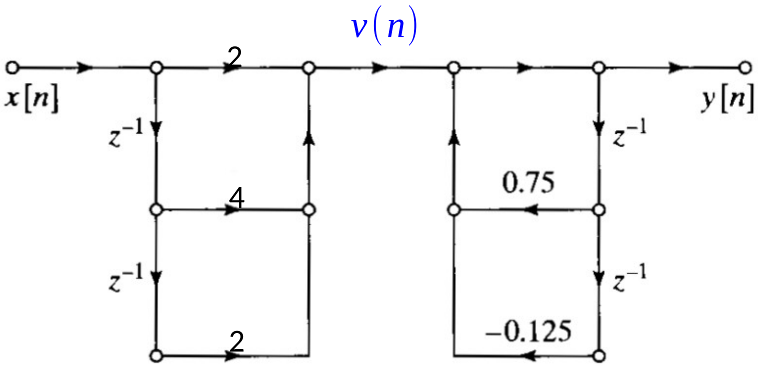
- Example, direct form I



$$H(z) = \frac{2 + 4z^{-1} + 2z^{-2}}{1 - 0.75z^{-1} + 0.125z^{-2}}$$



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



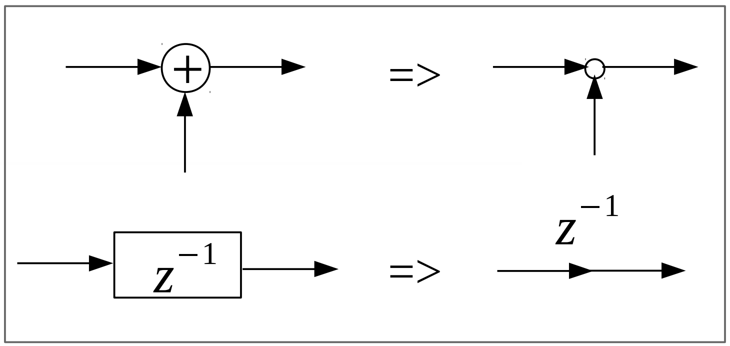
$$v(n) = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2)$$
$$y(n) = -a_1 y(n-1) - a_2 y(n-2) + v(n)$$

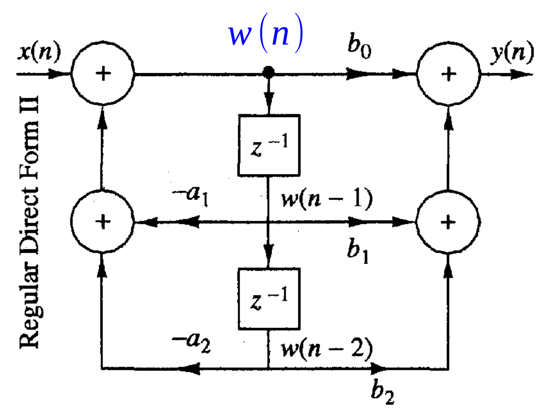
$$y(n) = 0.75\, y(n-1) - 0.125\, y(n-2) + \\ + 2x(n) + 4x(n-1) + 2x(n-2)$$

Direct form realizations of LTI discrete systems

- Example, direct form II

$$H(z)=\frac{2+4z^{-1}+2z^{-2}}{1-0.75z^{-1}+0.125z^{-2}}$$

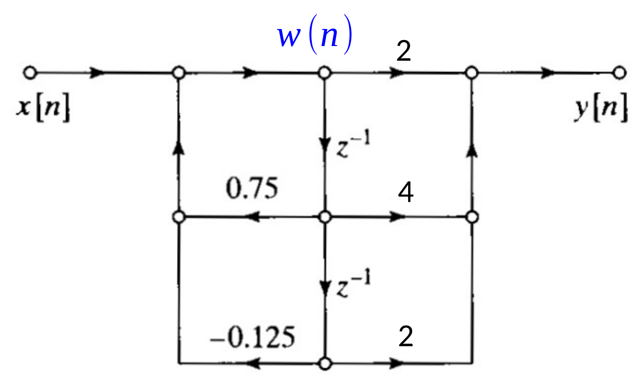




$$H(z)=\frac{b_0+b_1z^{-1}+b_2z^{-2}}{1+a_1z^{-1}+a_2z^{-2}}$$

$$w(n)=-a_1w(n-1)-a_2w(n-2)+x(n)$$
$$y(n)=b_0w(n)+b_1w(n-1)+b_2w(n-2)$$

$$w(n)=0.75w(n-1)-0.125w(n-2)+x(n)$$
$$y(n)=2w(n)+4w(n-1)+2w(n-2)$$



Direct form realizations of LTI discrete systems

- Homework

Draw the direct form I and direct form II realization for the following systems

$$y(n) - 2.5y(n-1) = -y(n-2) + x(n) - 5x(n-1) + 6x(n-2)$$

$$y(n) = 4y(n-1) - 4y(n-2) + 2x(n-1)$$

$$y(n) = 0.5y(n-1) + 2x(n)$$

Direct form realizations of LTI discrete systems

- Direct form II implementation of IIR systems requires minimum number of delay elements and minimum number of multiplications
- However, direct form II is error-prone IIR filter implementation
- How to realize large systems?
- To minimize the errors (overflow, quantization) associated with finite-word-length width implement
  - Cascade form of low order systems
  - Parallel form of low order systems

Basic structures for IIR systems: cascade form

- Cascade form
  - Practical form for cascade implementation using 2nd order sections
  - Base for each 2nd order section is again the direct form II

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 + \sum_{k=1}^{N} a_k z^{-k}} = \prod_{k=1}^{N_C} \frac{b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2}}{1 + a_{1k} z^{-1} + a_{2k} z^{-2}} = H_1(z).H_2(z)\ldots H_{N_c}(z)$$

- Systems (filters) with real outputs have conjugate symmetric roots

$$H(z) = \frac{1}{(1 - (2r\cos\theta) z^{-1} + r^2 z^{-2})} \qquad H(z) = \frac{1}{(1 - (\alpha + j\beta) z^{-1})(1 - (\alpha - j\beta) z^{-1})}$$

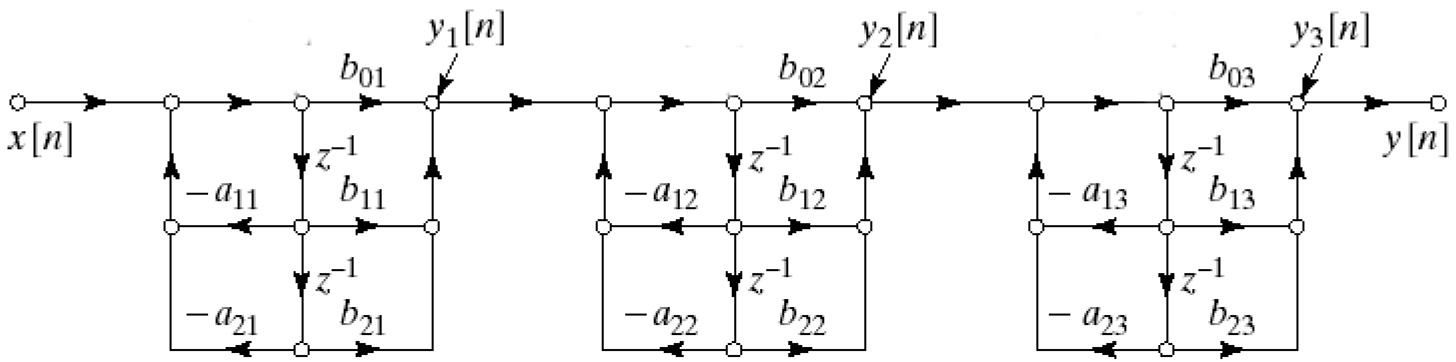
- Can always be grouped into 2nd order terms with real coefficients

$$(1 - 2\alpha z^{-1} + (\alpha^2 + \beta^2) z^{-2})$$

Basic structures for IIR systems: cascade form

- Cascade form
  - Base for each 2nd order section is again the direct form II

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 + \sum_{k=1}^{N} a_k z^{-k}} = \prod_{k=1}^{N_C} \frac{b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2}}{1 + a_{1k} z^{-1} + a_{2k} z^{-2}} = H_1(z).H_2(z)\ldots H_{N_C}(z)$$

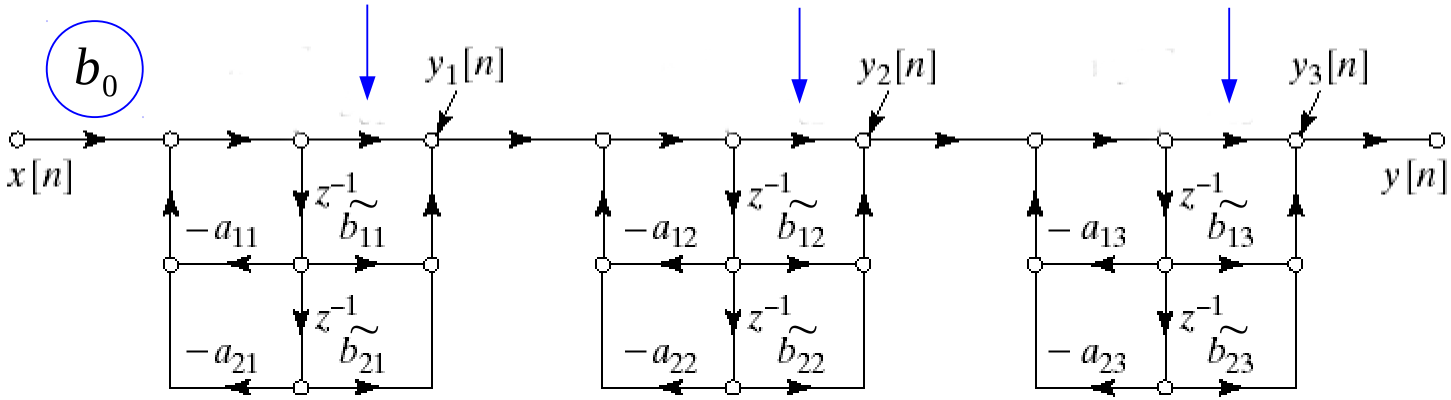


Basic structures for IIR systems: cascade form

- How to further reduce number of multiplications?

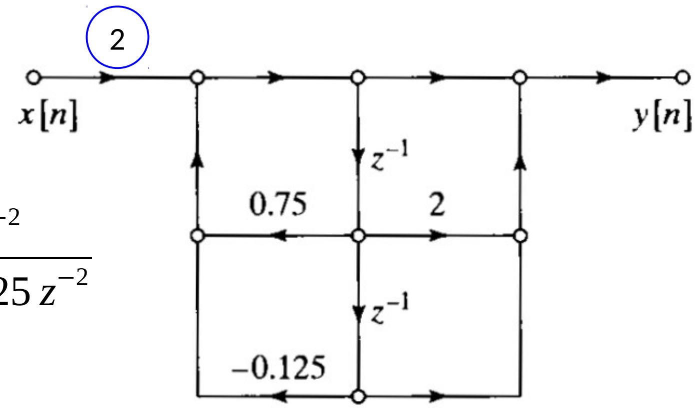
$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 + \sum_{k=1}^{N} a_k z^{-k}} = b_0 \prod_{k=1}^{N_C} \frac{1 + \widetilde{b}_{1k} z^{-1} + \widetilde{b}_{2k} z^{-2}}{1 + a_{1k} z^{-1} + a_{2k} z^{-2}} = b_0 \,.\, \widetilde{H}_1 . \widetilde{H}_2 ... \widetilde{H}_{N_C}$$

→ Forward gain factored out !



Basic structures for IIR systems: cascade form

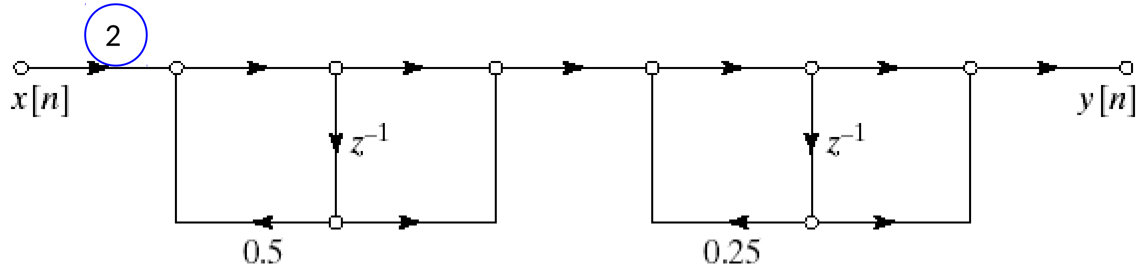
- Previous example (2nd order system)



$$H(z) = \frac{2 + 4z^{-1} + 2z^{-2}}{1 - 0.75z^{-1} + 0.125z^{-2}} = 2\frac{1 + 2z^{-1} + 1z^{-2}}{1 - 0.75z^{-1} + 0.125z^{-2}}$$

- Even more simplified (cascade of two 1st order sections, base: the direct form II)

$$H(z) = \frac{2 + 4z^{-1} + 2z^{-2}}{1 - 0.75z^{-1} + 0.125z^{-2}} = 2\frac{(1 + 1z^{-1})}{(1 - 0.5z^{-1})}\frac{(1 + z^{-1})}{(1 - 0.25z^{-1})}$$



 [Oppenheim, Schafer]

Basic structures for IIR systems: cascade form

- How to pick pairs of roots for the 2nd order sections to optimizenumerical properties (overflow, quantization)?
  - Avoid very large values (overflow) and very small values (quantization)
  - Use Matlab's zp2sos() or tf2sos() functions which convert a transfer function into 2nd order sections, and returns the coefficients of each section and the gain
  - The functions select the poles farthest from the unit circle for the front 2nd order sections (or vice versa)

Basic structures for IIR systems: cascade form

- Laboratory

Given the following system

$$H(z)=\frac{10+\frac{25}{3}z^{-1}-20z^{-2}+\frac{20}{3}z^{-3}}{1-\frac{15}{8}z^{-1}+\frac{47}{32}z^{-2}-\frac{17}{32}z^{-3}+\frac{3}{64}z^{-4}}$$

- Realize it as a cascade of direct form II 2nd order sections and draw the cascade

- Write transfer function of each of the resulting 2nd order sections

- MATLAB

```
>> [sos, g] = zp2sos(Z, P, K);  % Converts a discrete-time zero-pole-gain representation
>> [sos, g] = tf2sos(b, a);     % Converts a transfer function representation (b, a)
>>                              % → to an equivalent second-order section representation
>>                              % sos – matrix of coefficients of second order sections
>>                              % g – gain (b0)
>> % Recall also
>> [Z, P, K] = tf2zpk(b, a);
```

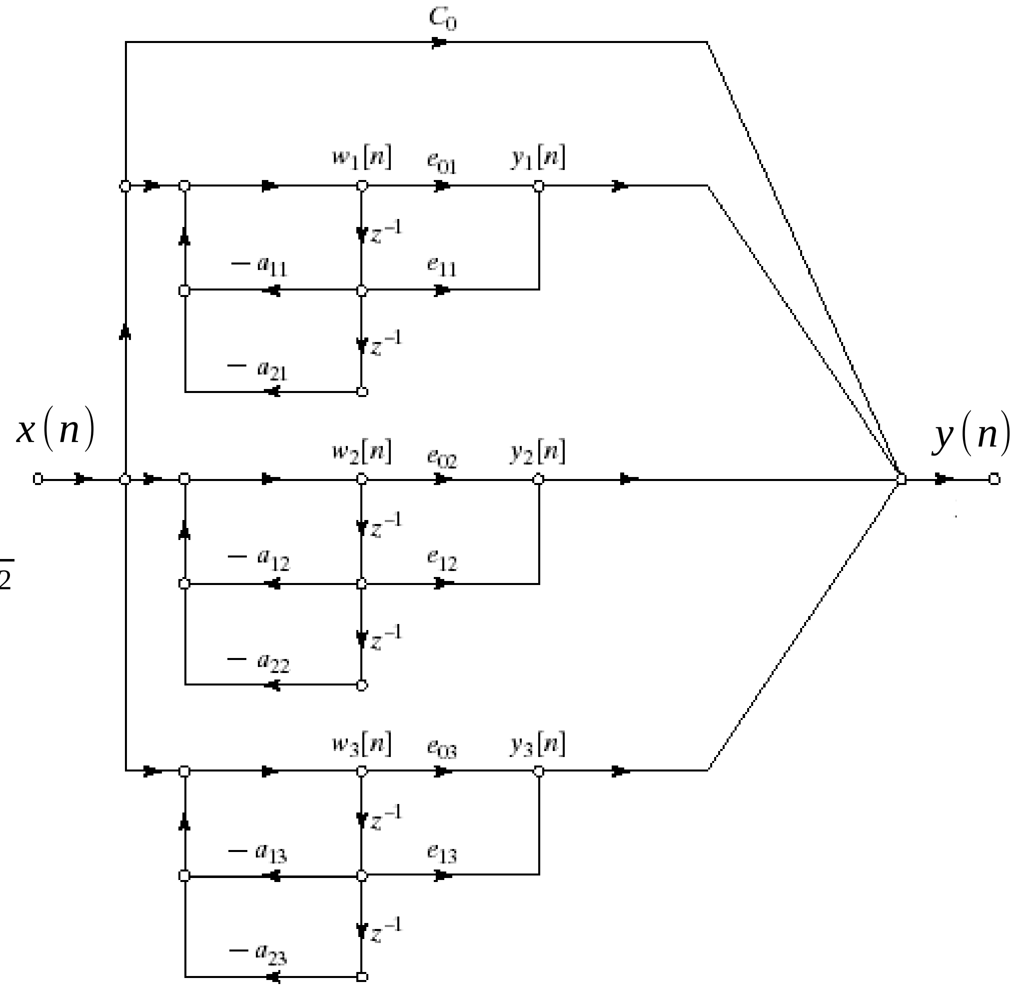
$$H(z)=\frac{\sum_{k=0}^{M} b_k z^{-k}}{1+\sum_{k=1}^{N} a_k z^{-k}}$$

Basic structures for IIR systems: parallel form

- Parallel form
  - Practical form for parallel implementation by grouping the real-valued poles in pairs
  - Base for each term under the sum is the direct form II

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

$$H(z) = \sum_{k=0}^{N_P} C_k z^{-k} + \sum_{k=1}^{N_S} H_k(z)$$



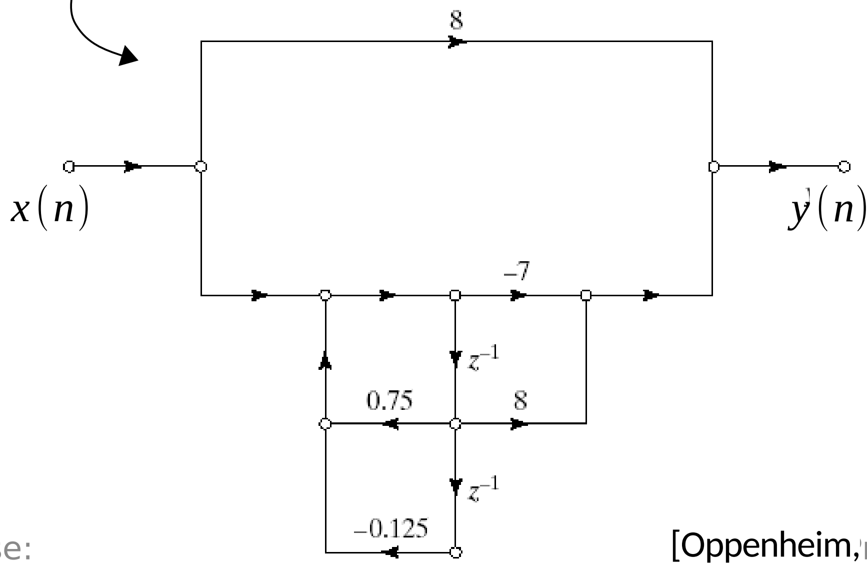
Basic structures for IIR systems: parallel form

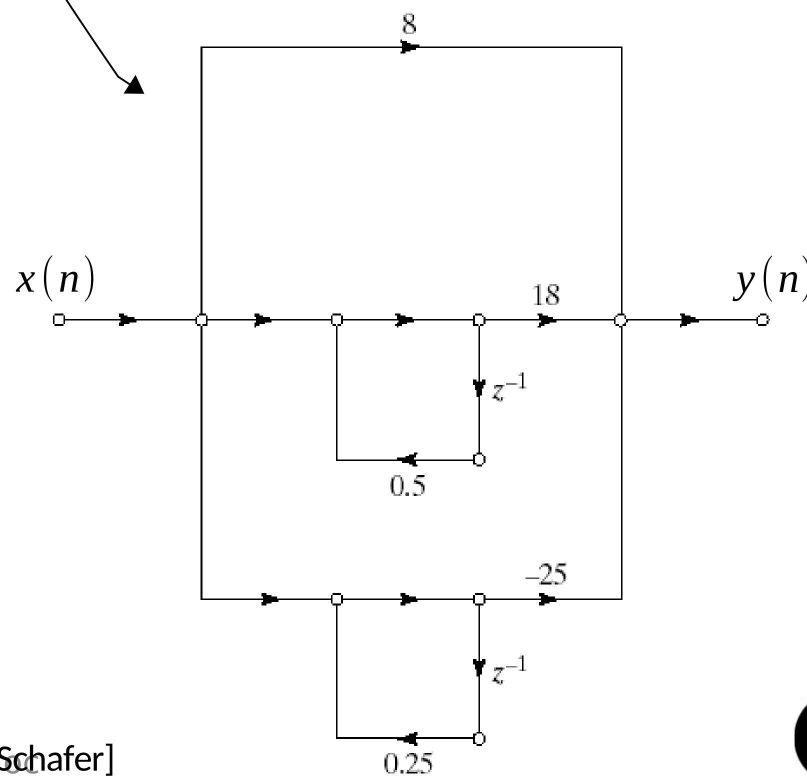
- Example, partial fraction expansion

$$H(z) = \frac{1 + 2z^{-1} + z^{-2}}{1 - 0.75z^{-1} + 0.125z^{-2}} = 8 + \frac{18}{(1-0.5z^{-1})} - \frac{25}{(1-0.25z^{-1})}$$

- After combining poles

$$H(z) = 8 + \frac{-7 + 8z^{-1}}{1 - 0.75z^{-1} + 0.125z^{-2}}$$





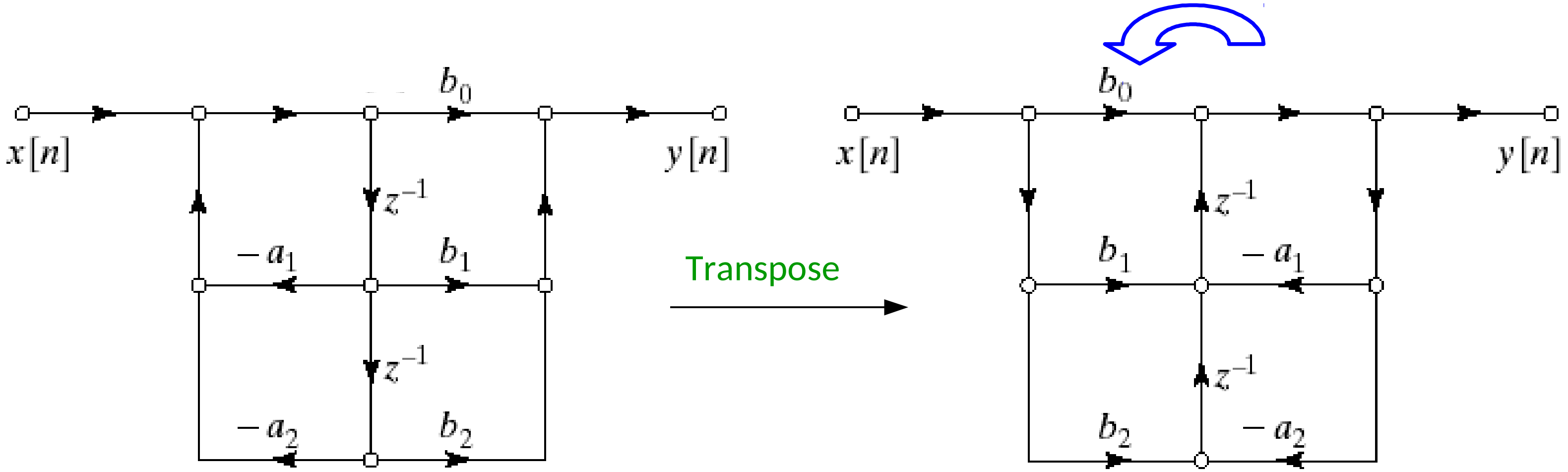
Comparison of structures

- Direct form I and II
  - Simple implementation, visible from difference equation
  - Quantization sensitive, larger number of bits
- Cascade or parallel form
  - Less quantization sensitive, can use lower number of bits
  - Can reuse and connects existing systems
  - Difficult to combine (find proper) pairs of roots
- Cascade form
  - Good for pipeline computation, high through-put
  - Can compose bandpass and bandstop filters using lowpass and highpass filters
- Parallel form
  - Fast parallel computation, no noise amplification (more robust)
  - Can compose filter banks, spectral analyzers

Transposing

- Transposing does not change the input- output relation
- Transposing (the transposition theorem)

  1) Reverse the directions of all branches; 2) Interchange input and output nodes
- Example, 2^{nd} order section, direct form II and its transpose

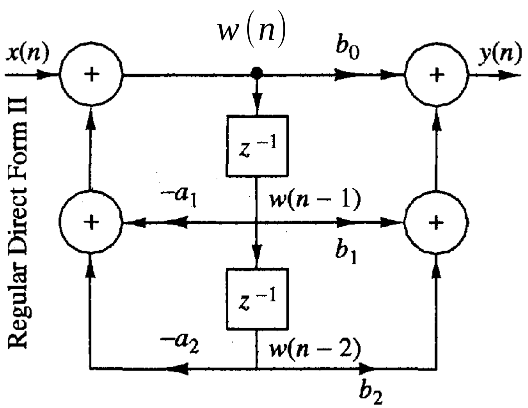


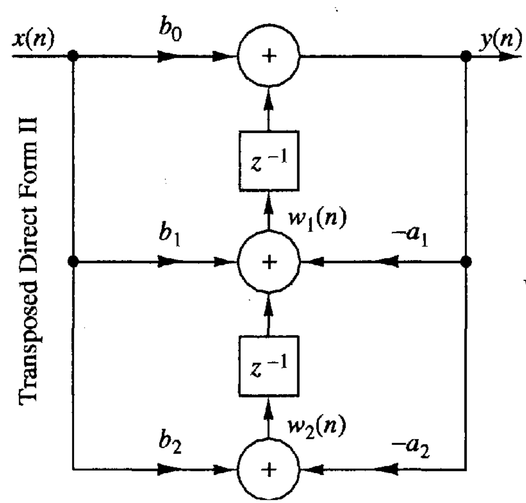
- The transposed form is less susceptible to the errors due to finite precision arithmetic in comparison to the direct form II since it implements non-recursive part prior to the recursive one.
- In addition, factors out forward gain

 [Oppenheim, Schafer]

Second order modules for discrete-time systems

- Structures and transfer functions
- Regular direct form II, and Transposed direct form II
- Verify their transfer functions !





$$y(n) = b_0 x(n) + w_1(n-1)$$

$$w_1(n) = b_1 x(n) - a_1 y(n) + w_2(n-1)$$

$$w_2(n) = b_2 x(n) - a_2 y(n)$$

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

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

$$H_{\mathrm{II}}(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} = H_{\mathrm{T}}(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

[Proakis, Manolakis]

Second order modules for discrete-time systems

- Homework
  - Draw the direct form I and direct form II realization for the following systems
  - Transpose the direct form II realizations
  - Verify transfer functions of direct form II and transposed form

$$y(n) - 2.5\,y(n-1) = -y(n-2) + x(n) - 5\,x(n-1) + 6\,x(n-2)$$

$$y(n) = 4\,y(n-1) - 4\,y(n-2) + 2\,x(n-1)$$

$$y(n) = 0.5\,y(n-1) + 2\,x(n)$$

Second order modules for discrete-time systems

- Homework

The transfer function of a 2nd order system is given by

$$H(z) = \frac{-r\sin\theta}{z^2 - 2r\cos\theta . z + r^2}$$

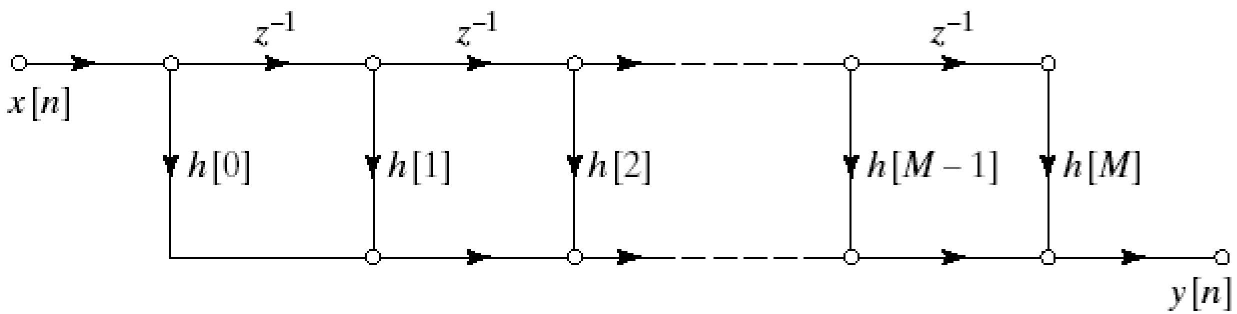
- Draw the direct form II realization of this system

- What is the condition for this system to be stable?

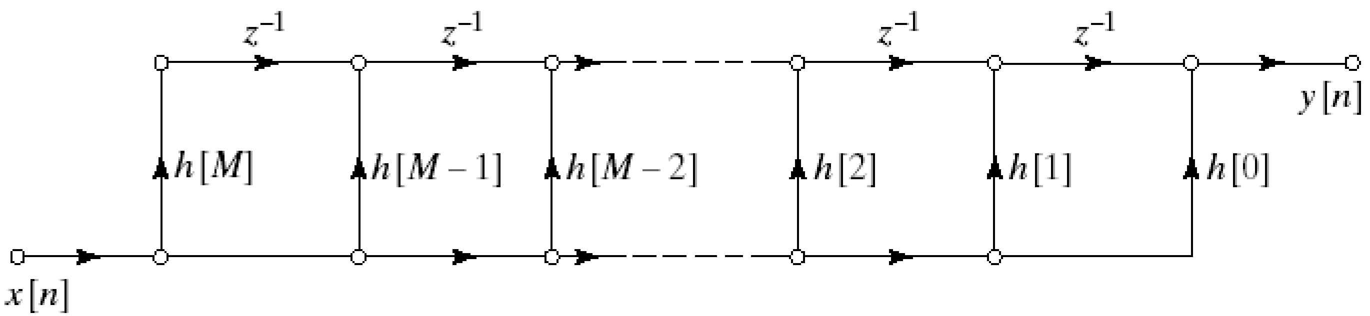
Basic structures for FIR systems: direct forms

- Direct form I – Tapped Delay Line

$$H(z) = \sum_{k=0}^{M} h_k z^{-k}$$



- Transpose of direct form I gives direct form II



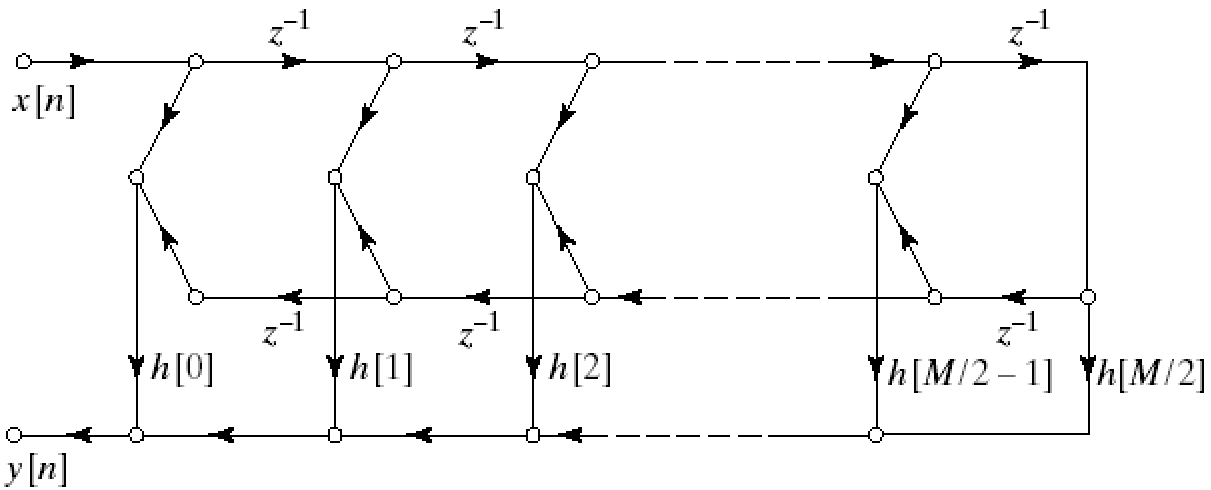
[Oppenheim, Schafer]

Basic structures for FIR systems: direct forms

- Direct form realization of linear-phase FIR system (M+1 odd)
- The structure takes advantage of the symmetry of the impulse response
- Half the number of multiplications

$$h(n) = h(-n), \quad n = 0,1,\dots,M$$

$$y(n) = \sum_{k=0}^{M} h(k)x(n-k) = \sum_{k=0}^{M/2-1} h(k)\left(x(n-k) + x(n-M+k)\right) + h(M/2)x(n-M/2)$$

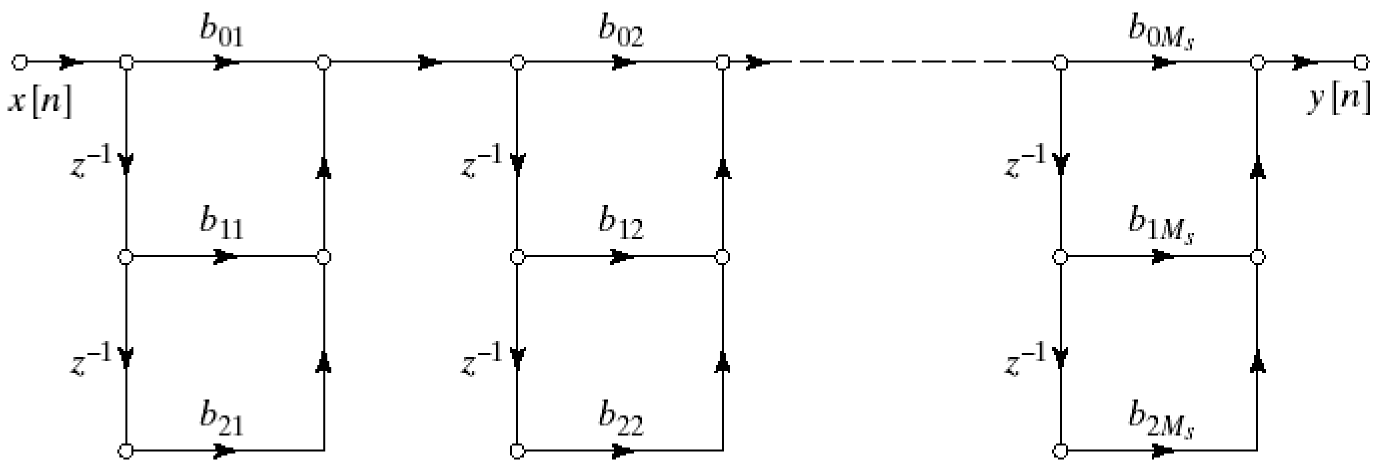


[Oppenheim, Schafer]

Basic structures for FIR systems: cascade form

- Factoring the polynomial systems function

$$H(z) = \sum_{k=0}^{M} h_k z^{-k} = \prod_{k=1}^{M_S} \left(b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2}\right)$$



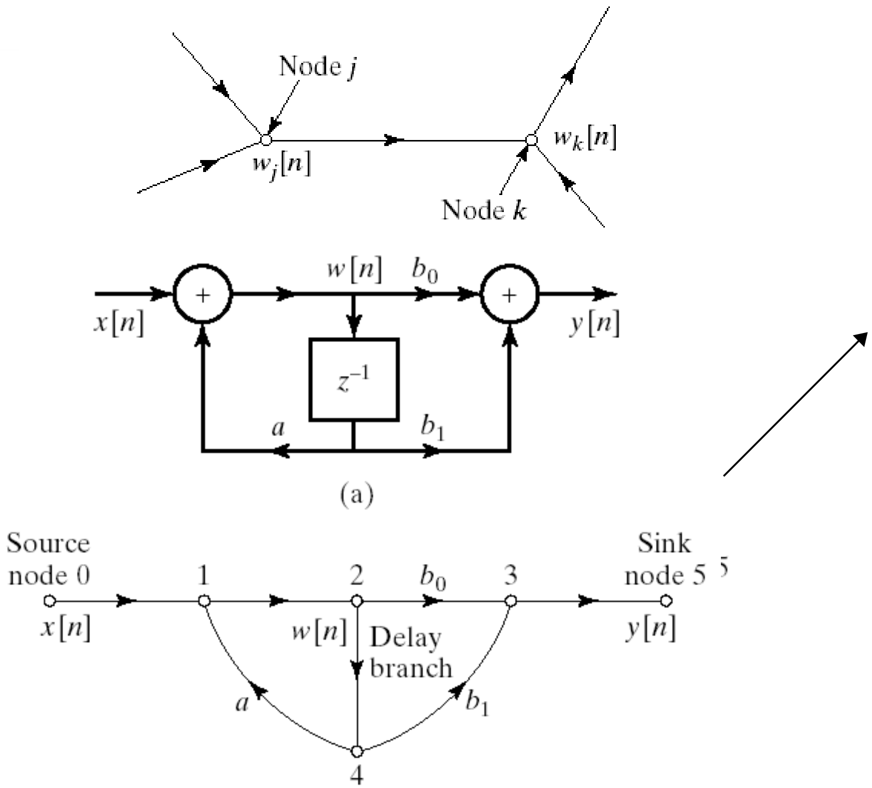
[Oppenheim, Schafer]

(Additional materials)

- Signal flow graph representation
- Second order modules for discrete-time systems

Signal flow graph representation

- Similar to block diagrams
- A network of directed branches connected at nodes
- Example, 1st order section, representation of direct form II as flow graph



$$
\begin{aligned}
w_1(n) &= a w_4(n) + x(n) \\
w_2(n) &= w_1(n) \\
w_3(n) &= b_0 w_2(n) + b_1 w_4(n) \\
w_4(n) &= w_2(n-1) \\
y(n) &= w_3(n)
\end{aligned}
$$

$$
\begin{aligned}
w_1(n) &= a w_1(n-1) + x(n) \\
y(n) &= b_0 w_1(n) + b_1 w_1(n-1)
\end{aligned}
$$

[Oppenheim, Schafer]

Second order modules for discrete-time systems

- Additional task

The transfer function of a digital filter is given as follows

$$H(z) = \frac{0.1669 + 0.3207\,z^{-1} + 0.1943\,z^{-2}}{1 + 1.7935\,z^{-1} + 0.8410\,z^{-2}}$$

- Discuss the stability of this filter
- Find the amplitude values of the frequency response at $\omega = 0$, $\pi/4$, $\pi/2$, $3\pi/4$, and π
- Draw the frequency response
- Sketch the block diagram of the direct form II realization of this filter