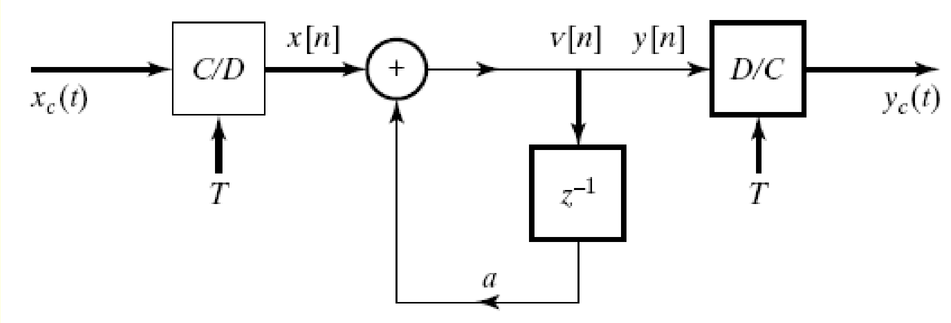
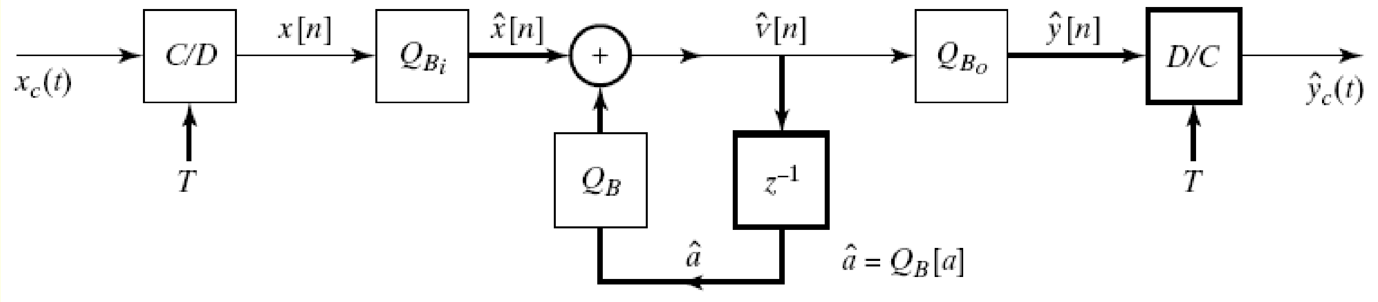
Quantization in Implementing Systems

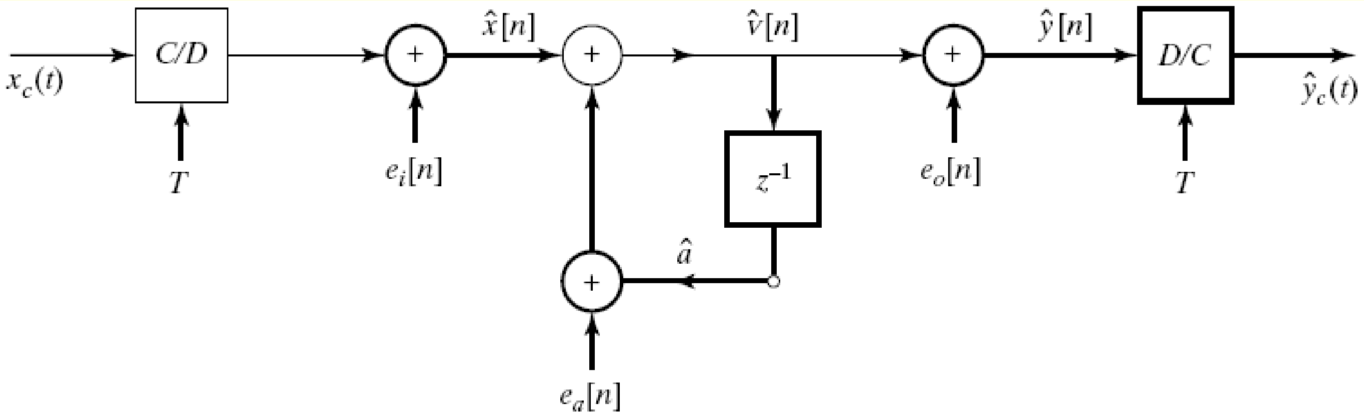
- Consider the following system



- A more realistic model would be



- In order to analyze it we would prefer



Effects of Coefficient Quantization in IIR Systems

- When the parameters of a rational system are quantized
  - The poles and zeros of the system function move
- If the system structure of the system is sensitive to perturbation of coefficients
  - The resulting system may no longer be stable
  - The resulting system may no longer meet the original specs
- We need to do a detailed sensitivity analysis
  - Quantize the coefficients and analyze frequency response
  - Compare frequency response to original response
- We would like to have a general sense of the effect of quantization

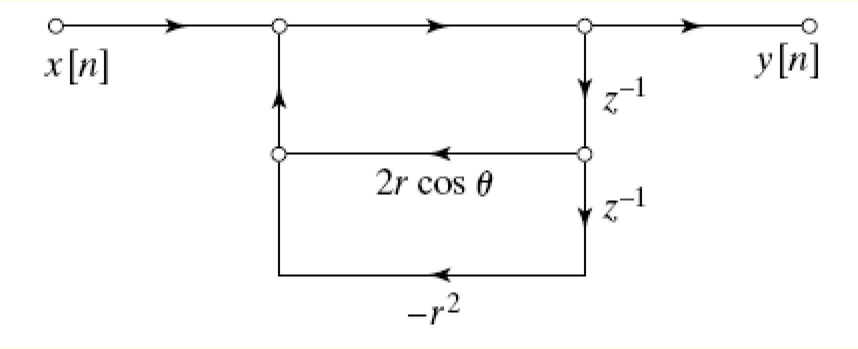
Effects on Roots

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 - \sum_{k=1}^{N} a_k z^{-k}} \xrightarrow{\text{Quantization}} \hat{H}(z) = \frac{\sum_{k=0}^{M} \hat{b}_k z^{-k}}{1 - \sum_{k=1}^{N} \hat{a}_k z^{-k}}$$

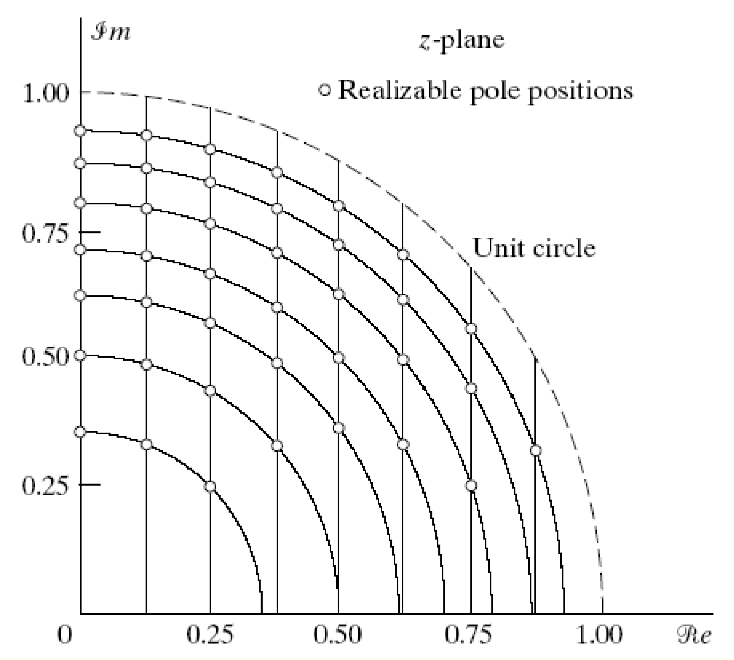
- Each root is affected by quantization errors in ALL coefficient
- Tightly clustered roots can be significantly effected
  - Narrow-bandwidth lowpass or bandpass filters can be very sensitive to quantization noise
- The larger the number of roots in a cluster the more sensitive it becomes
- This is the reason why second order cascade structures are less sensitive to quantization error than higher order system
  - Each second order system is independent from each other

Poles of Quantized Second-Order Sections

- Consider a 2nd order system with complex-conjugate pole pair

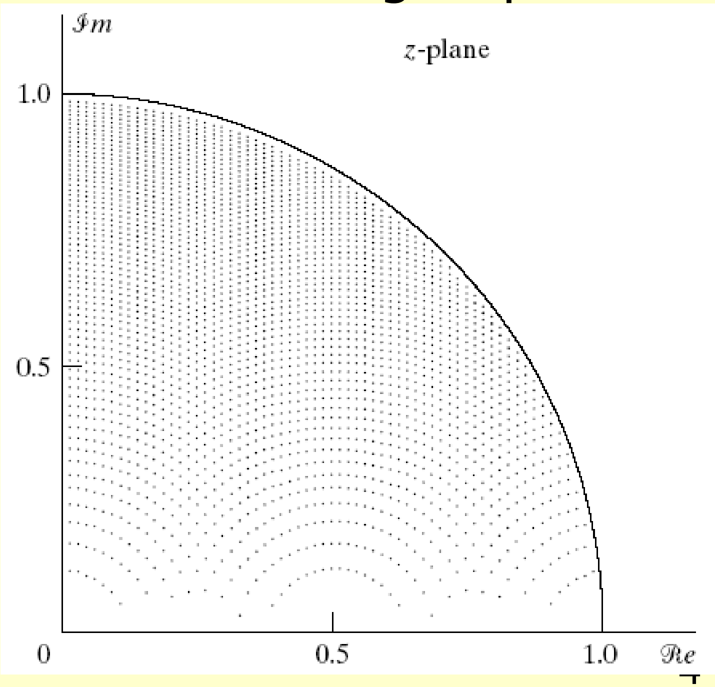


- The pole locations after quantization will be on the grid point



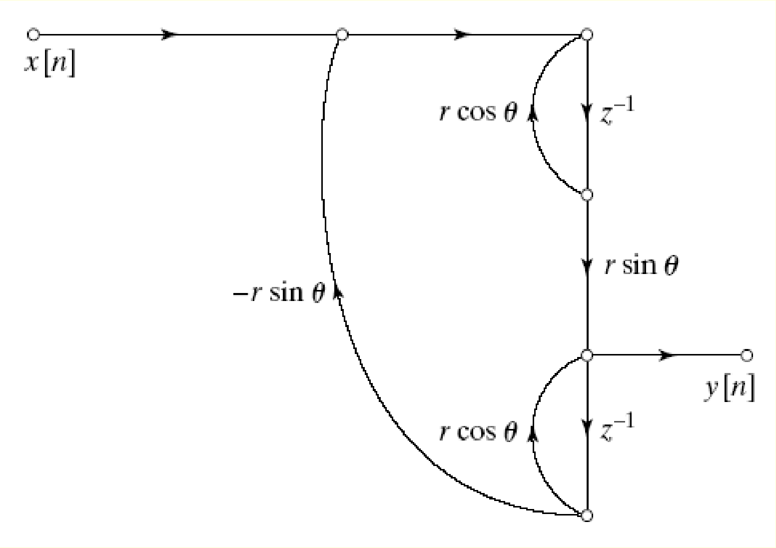
← 3-bits

7-bits →

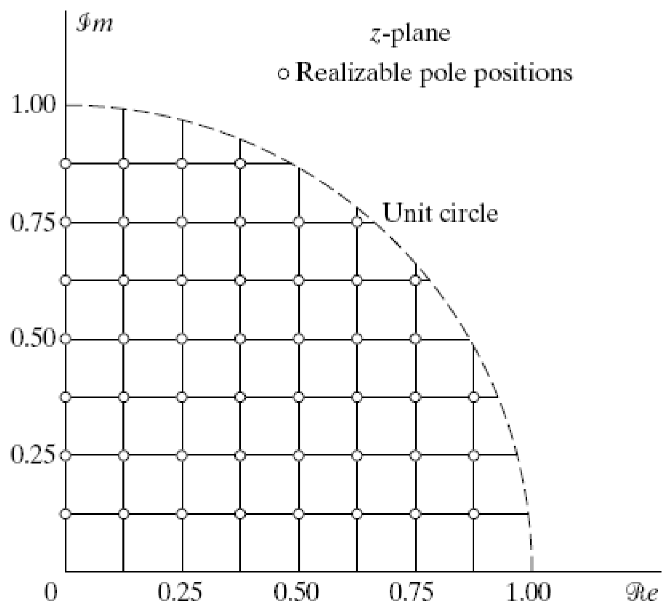


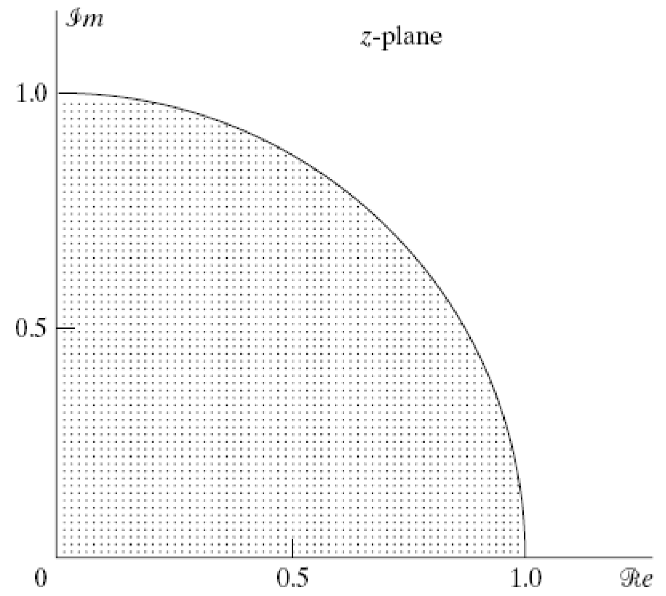
Coupled-Form Implementation of Complex-Conjugate Pair

- Equivalent implementation of the second order system



- But the quantization grid this time is





Effects of Coefficient Quantization in FIR Systems

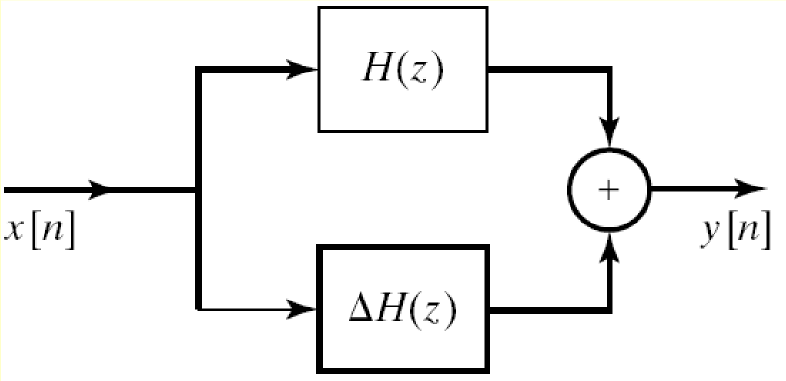
- No poles to worry about only zeros
- Direct form is commonly used for FIR systems

$$H(z) = \sum_{n=0}^{M} h[n]z^{-n}$$

- Suppose the coefficients are quantized

$$\hat{H}(z) = \sum_{n=0}^{M} \hat{h}[n]z^{-n} = H(z) + \Delta H(z) \qquad \Delta H(z) = \sum_{n=0}^{M} \Delta h[n]z^{-n}$$

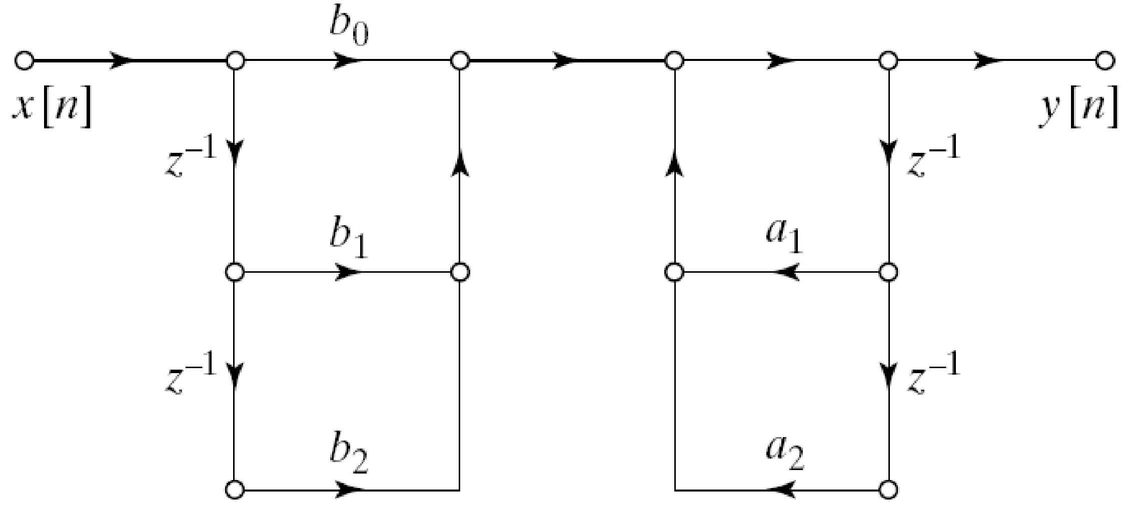
- Quantized system is linearly related to the quantization error

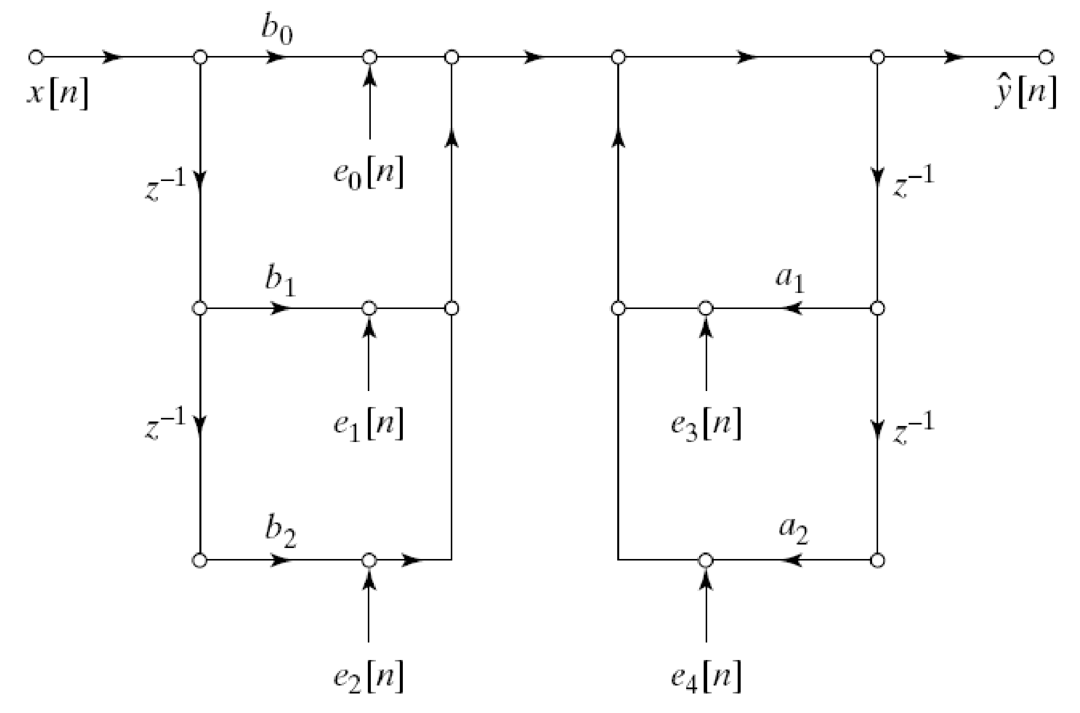


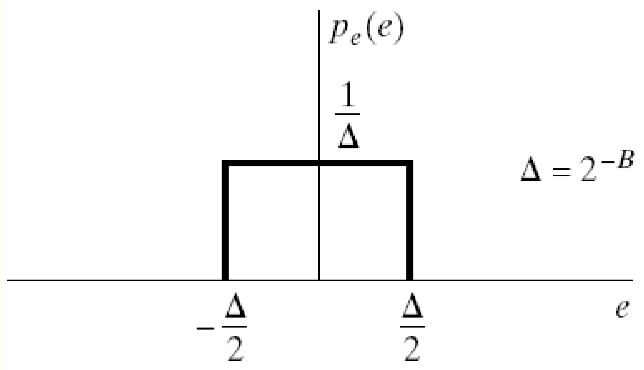
- Again quantization noise is higher for clustered zeros
- However, most FIR filters have spread zeros

Round-Off Noise in Digital Filters

- Difference equations implemented with finite-precision arithmetic are non-linear systems
- Second order direct form I system
- Model with quantization effect
- Density function error terms for rounding

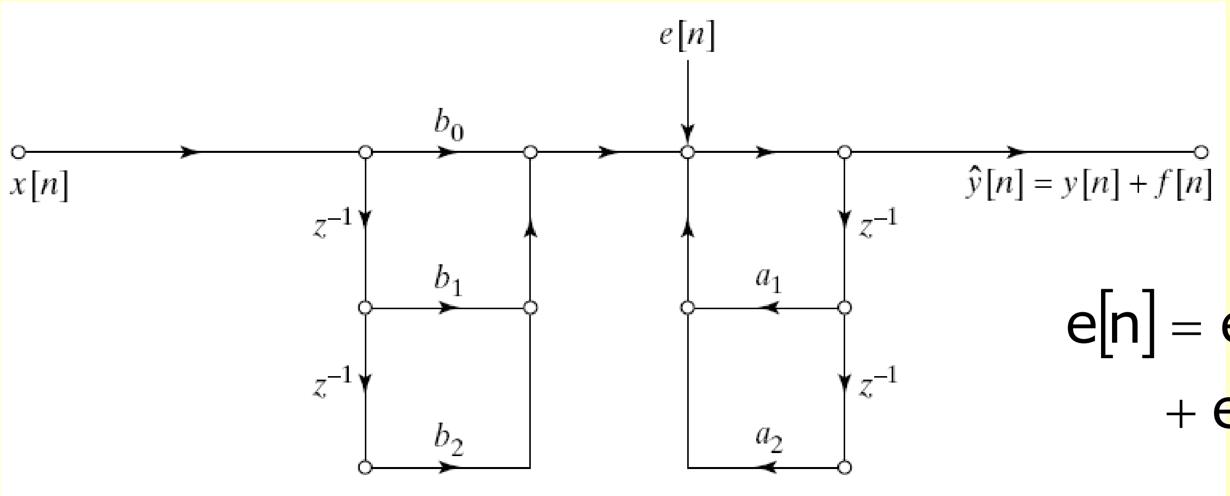






Analysis of Quantization Error

- Combine all error terms to single location to get



$$e[n] = e_0[n] + e_1[n] + e_2[n] + e_3[n] + e_4[n]$$

- The variance of e[n] in the general case is $\sigma_e^2 = (M + 1 + N)\dfrac{2^{-2B}}{12}$

- The contribution of e[n] to the output is $f[n] = \sum_{k=1}^{N} a_k f[n-k] + e[n]$

- The variance of the output error term f[n] is

$$\sigma_f^2 = (M + 1 + N)\frac{2^{-2B}}{12}\sum_{n=-\infty}^{\infty}|h_{ef}[n]|^2 \qquad H_{ef}(z) = 1/A(z)$$

Round-Off Noise in a First-Order System

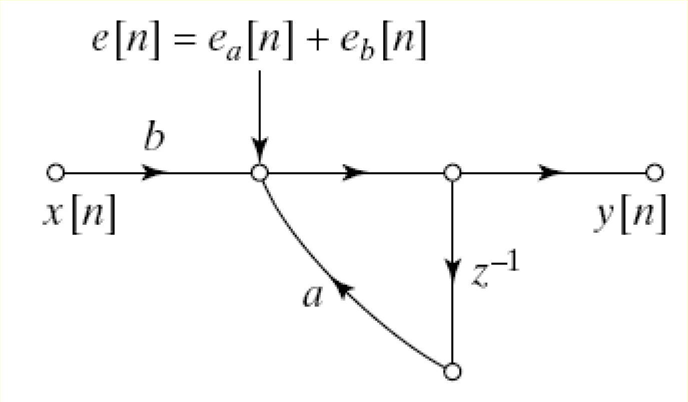
- Suppose we want to implement the following stable system

$$H(z) = \frac{b}{1 - az^{-1}} \qquad |a| < 1$$

- The quantization error noise variance is

$$\sigma_f^2 = (M + 1 + N)\frac{2^{-2B}}{12}\sum_{n=-\infty}^{\infty}|h_{ef}[n]|^2 = 2\frac{2^{-2B}}{12}\sum_{n=0}^{\infty}|a|^{2n} = 2\frac{2^{-2B}}{12}\left(\frac{1}{1-|a|^2}\right)$$

- Noise variance increases as |a| gets closer to the unit circle
- As |a| gets closer to 1 we have to use more bits to compensate for the increasing error

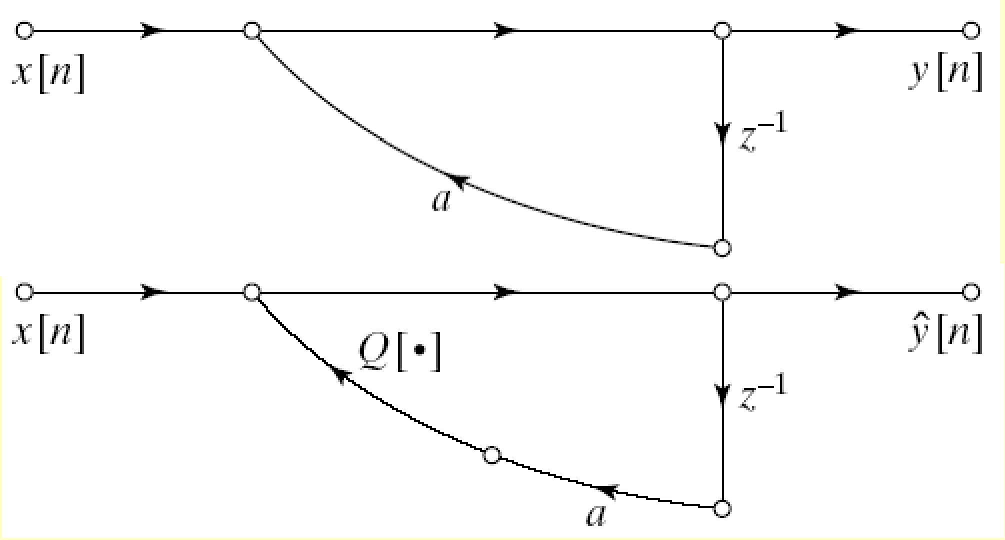


Zero-Input Limit Cycles in Fixed-Point Realization of IIR Filters

- For stable IIR systems the output will decay to zero when the input becomes zero
- A finite-precision implementation, however, may continue to oscillate indefinitely
- Nonlinear behaviour very difficult to analyze so we sill study by example
- Example: Limite Cycle Behavior in First-Order Systems

$$y[n] = ay[n-1] + x[n] \qquad\qquad |a| < 1$$

- Assume x[n] and y[n-1] are implemented by 4 bit registers



Example Cont'd

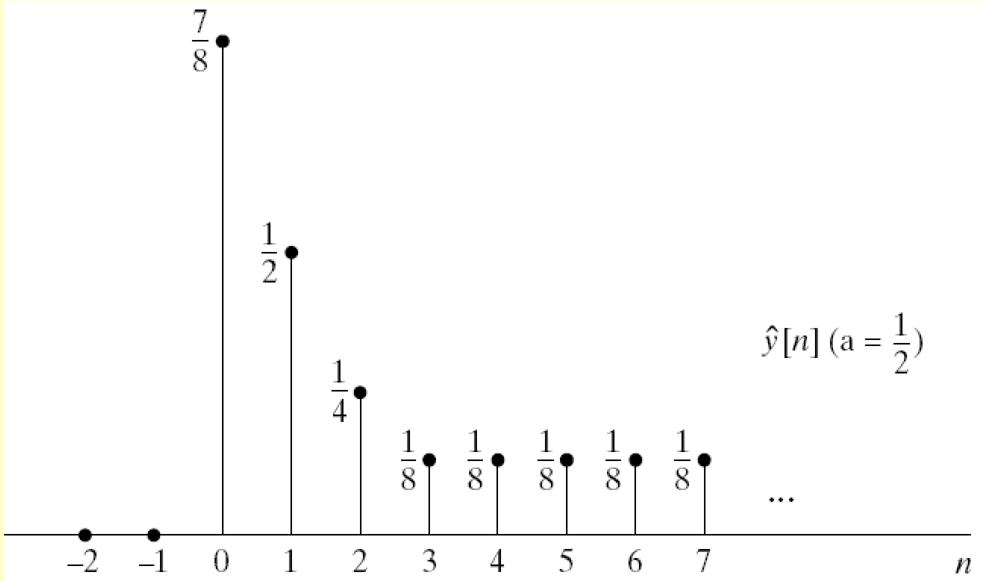
$$y[n] = ay[n-1] + x[n] \qquad |a| < 1$$

- Assume that a=1/2=0.100b and the input is

$$x[n] = \frac{7}{8}\delta[n] = (0.111b)\delta[n]$$

- If we calculate the output for values of n

n	y[n]	Q(y[n])
0	7/8=0.111b	7/8=0.111b
1	7/16=0.011100b	1/2=0.100b
2	1/4=0.010000b	1/4=0.010b
3	1/8=0.001000b	1/8=0.001b
4	1/16=0.00010b	1/8=0.001b



- A finite input caused an oscilation with period 1

Example: Limite Cycles due to Overflow

- Consider a second-order system realized by
$$\hat{y}[n] = x[n] + Q(a_1\hat{y}[n-1]) + Q(a_2\hat{y}[n-2])$$
  - Where Q() represents two's complement rounding
  - Word length is chosen to be 4 bits
- Assume a_1=3/4=0.110b and a_2=-3/4=1.010b
- Also assume
$$\hat{y}[-1] = 3/4 = 0.110b \text{ and } \hat{y}[-2] = -3/4 = 1.010b$$
- The output at sample n=0 is
$$\hat{y}[0] = 0.110b \times 0.110b + 1.010b \times 1.010b$$
$$= 0.100100b + 0.100100b$$
- After rounding up we get
$$\hat{y}[0] = 0.101b + 0.101b = 1.010b = \text{-}3/4$$
- Binary carry overflows into the sign bit changing the sign
- When repeated for n=1
$$\hat{y}[0] = 1.010b + 1.010b = 0.110 = 3/4$$

Avoiding Limite Cycles

- Desirable to get zero output for zero input: Avoid limit-cycles
- Generally adding more bits would avoid overflow
- Using double-length accumulators at addition points would decrease likelihood of limit cycles
- Trade-off between limit-cycle avoidance and complexity
- FIR systems cannot support zero-input limit cycles