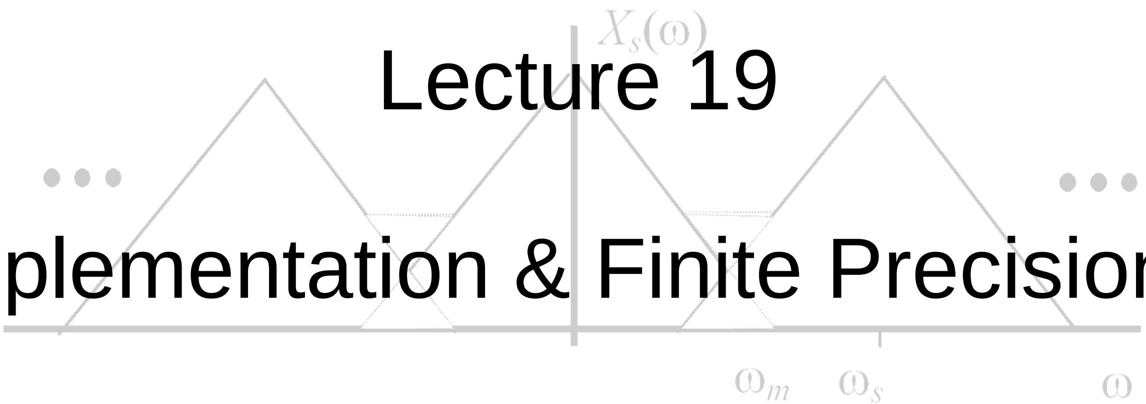
Lecture 19

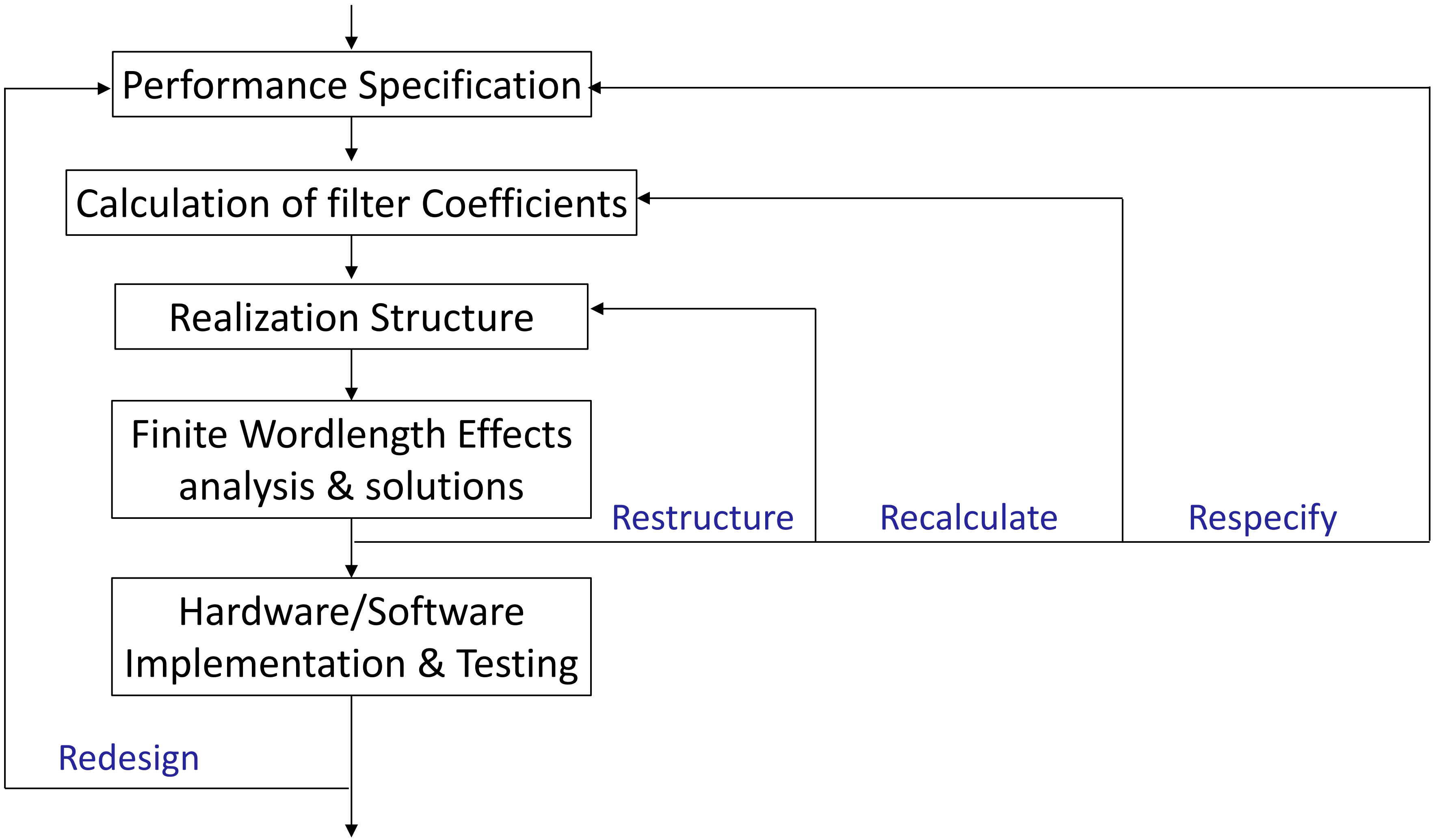
Filter Implementation & Finite Precision Effects



Dr. Dimitris Agrafiotis, Dept. Of Electrical & Electronic Engineering

Digital Filter Design Procedure

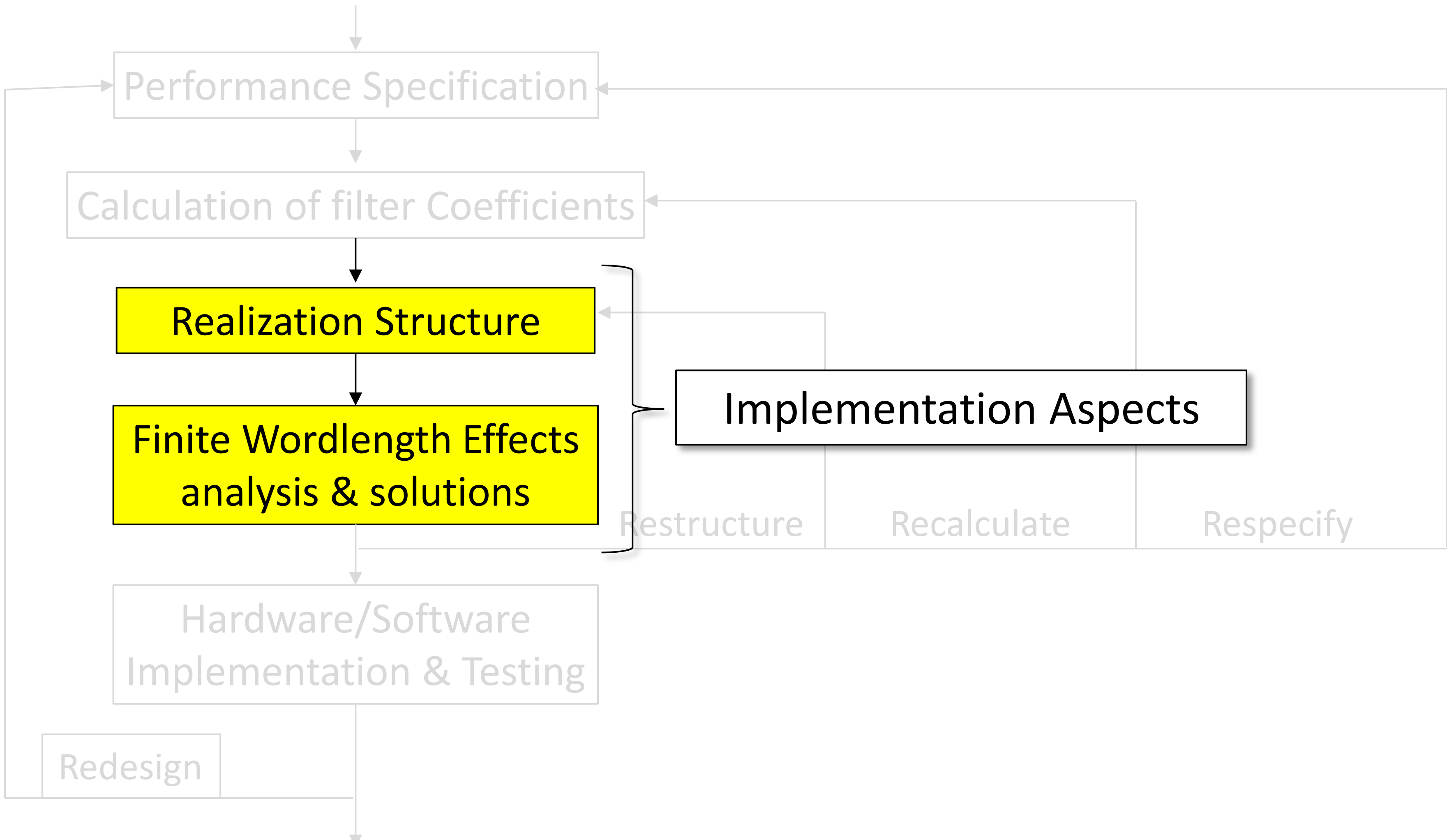
Design Stages for Digital Filters



Finite Precision Effects

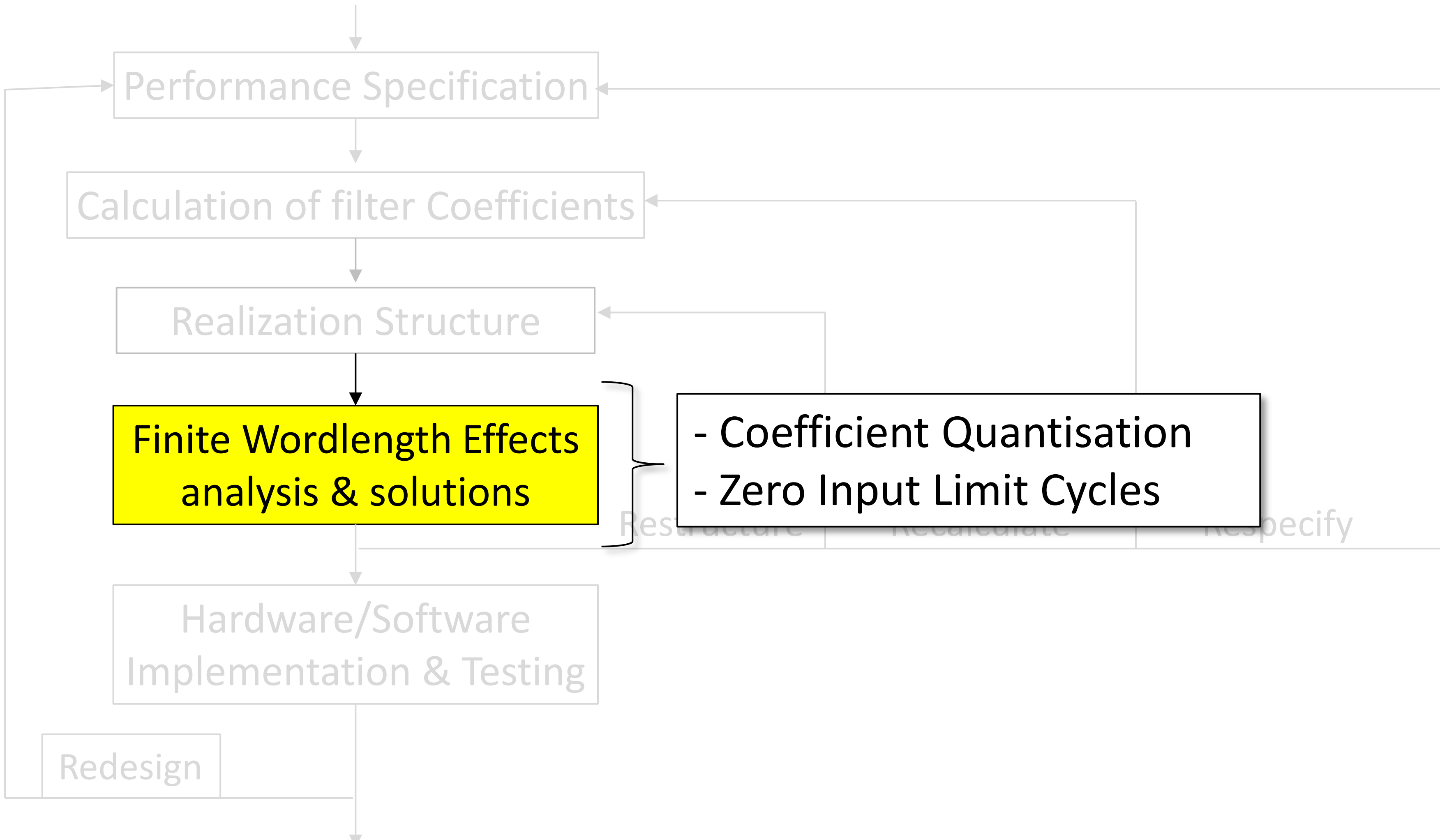
Digital Filter Design Procedure

Design Stages for Digital Filters



Digital Filter Design Procedure

Design Stages for Digital Filters



Finite Precision Effects

Number Representation

Fixed Point & Floating Point Arithmetic

- Implementing digital filters requires **representation of numbers** (coefficients, intermediate results and output) **with finite precision**.
- **Quantisation** is applied to map numbers to a fixed number of bits
- With **floating point arithmetic** and 32 or 64 bit wordlengths (e.g. general processors) accuracy is very high - **quantisation** issues are **of little or no concern**
- With **fixed point arithmetic** and smaller wordlengths (e.g. many embedded processors) **quantisation becomes an issue**
- Quantisation is a non-linear operation

Fixed Point Arithmetic

Quantisation Error & Dynamic Range

2s Complement Binary Representation

Infinite number of bits

Infinite precision representation: $$x = X_m \left(-b_0 + \sum_{i=1}^{\infty} b_i 2^{-i} \right)$$

Scaling factor

Sign Bit

$$if \quad b_0 = 0 \quad 0 \le x \le X_m$$

$$if \quad b_0 = 1 \quad -X_m \le x < 0$$

Finite number of bits

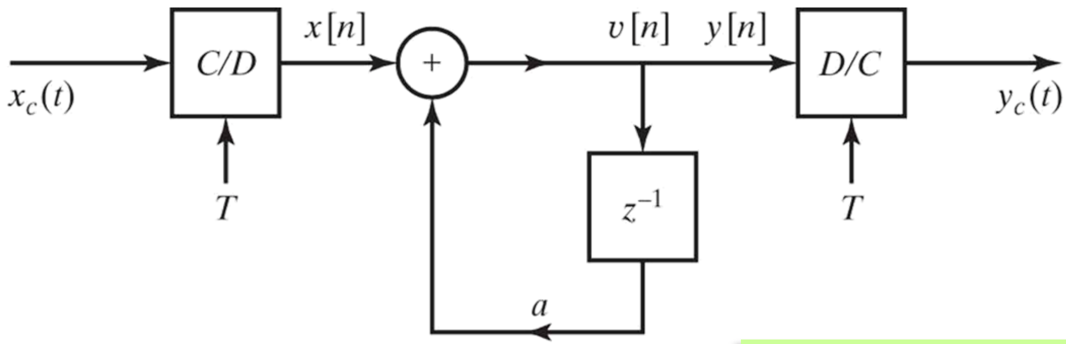
Finite precision representation: $$\hat{x} = Q_B[x] = X_m \left(-b_0 + \sum_{i=1}^{B} b_i 2^{-i} \right)$$

Quantisation Step: $$\Delta = X_m 2^{-B}$$

Quantisation Error: $$e = Q_B[x] - x$$

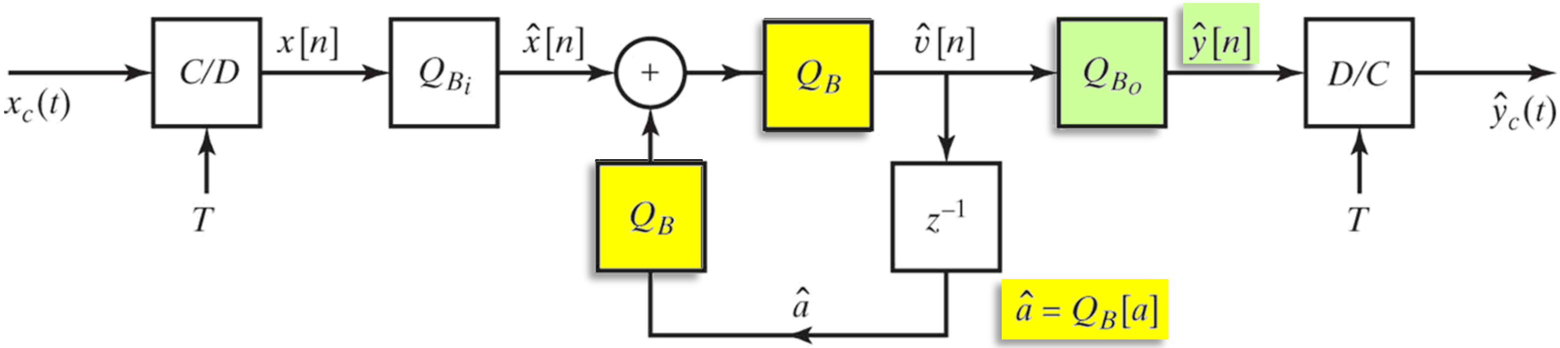
Digital Filter Implementation

Infinite vs. Finite Precision Implementation



(a)

Non-linearity and Limit Cycles



$\hat{a} = Q_B[a]$

(b)

Quantisation of Filter Coefficients

Digital Filter Implementation

Quantisation of Filter Coefficients

How close is the performance of the implemented filter with system function $\hat{H}(z)$ to the designed one with system function $H(z)$

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

Poles occur at slightly different location due to quantisation of filter coefficients

Finite Precision Effects

Effects of Coefficient Quantisation

IIR Digital Filters

- Poles and zeros change location due to coefficient quantisation
- The filter's frequency response will change and may not meet the specification
- The filter may even become unstable if the poles move outside the unit circle

$$\hat{H}(z) = \sum_{k=0}^{M} \hat{b}_k z^{-k} \Big/ 1 - \sum_{k=1}^{N} \hat{a}_k z^{-k}$$

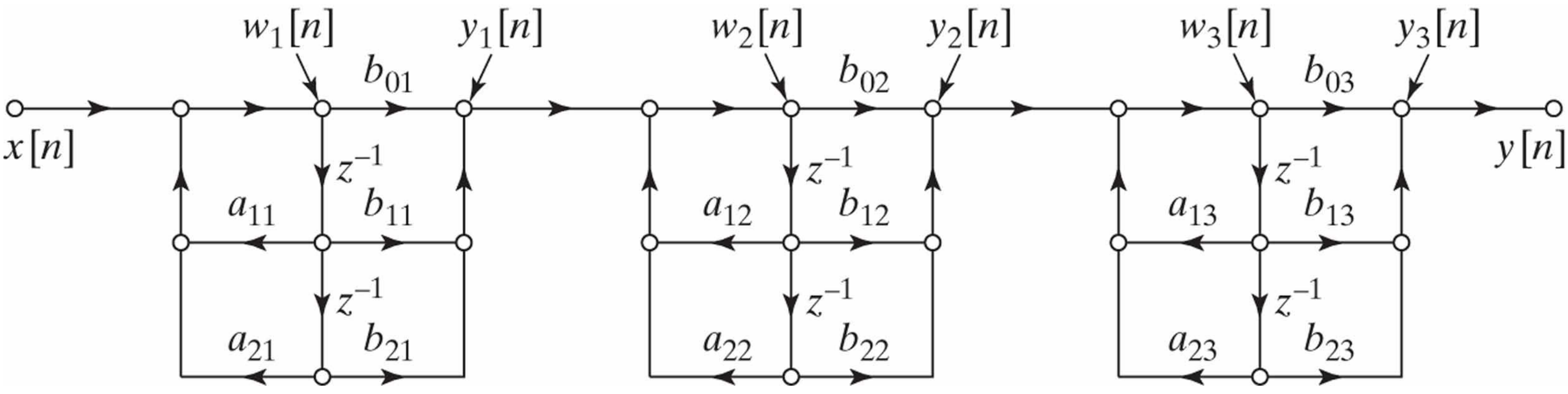
- The roots of the polynomials (denominator or numerator) are affected by all the coefficients.
- Each pole and zero will be affected by all the quantization errors in the denominator and numerator polynomial respectively
- If roots are tightly clustered then large shifts can happen with the direct form

Effects of Coefficient Quantisation

IIR Digital Filters – Cascade and Parallel Forms

- Combinations of 2nd order direct form systems
- Pairs of poles and zeros (the latter only for cascade form) are realized independently of other poles
- Parallel and especially cascade form less sensitive to coefficient quantisation

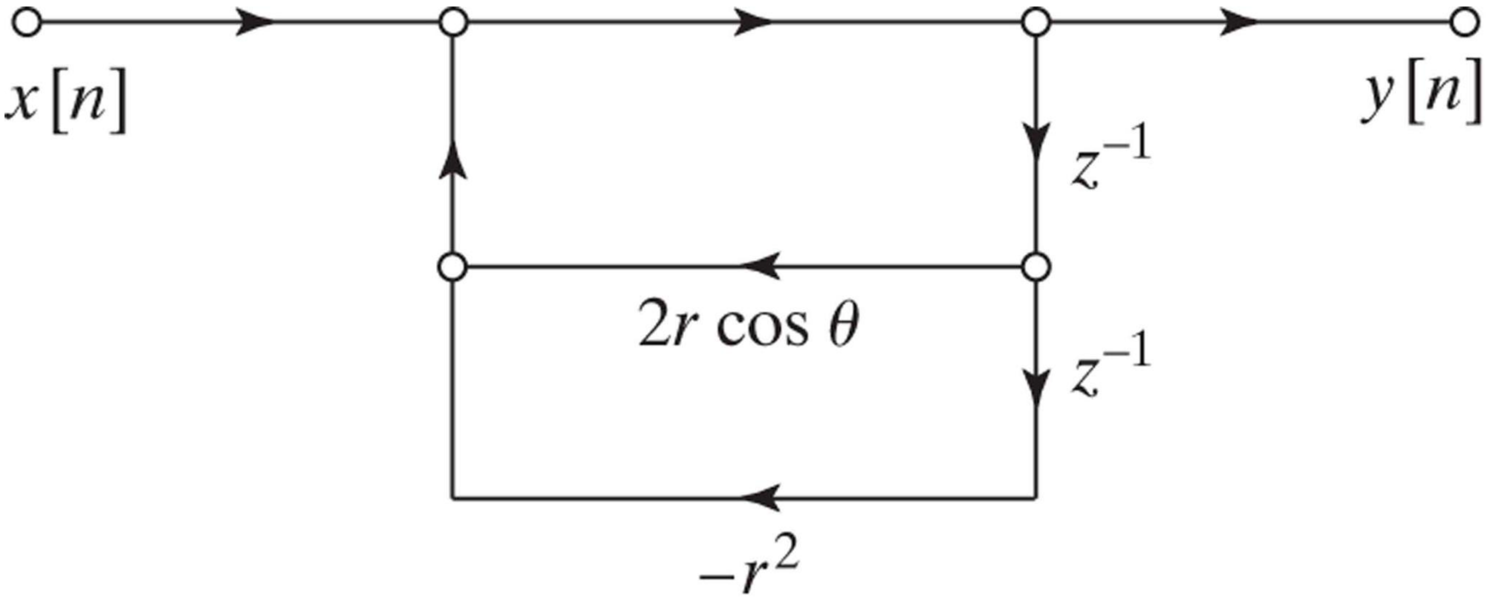
$$H(z)=\prod_{k=1}^{N_s} H_k(z)=\prod_{k=1}^{N_s}\frac{b_{0k}+b_{1k}z^{-1}+b_{2k}z^{-2}}{1-a_k z^{-1}-a_{2k}z^{-2}}$$



Effects of Coefficient Quantisation

IIR Digital Filters - Example

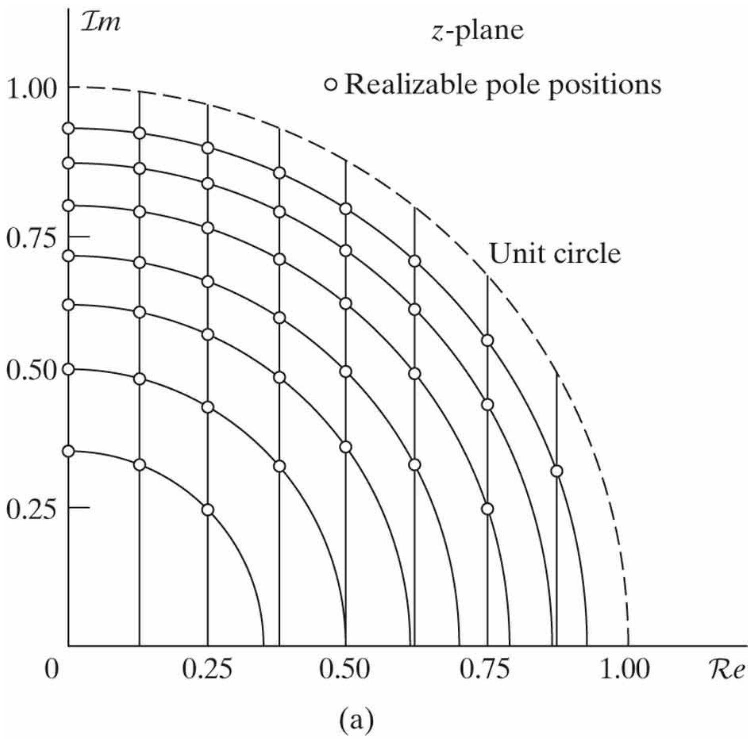
- 2^{nd} order resonator filter $H(z) = \dfrac{1}{1 - 2r\cos(\theta) z^{-1} + r^2 z^{-2}}$

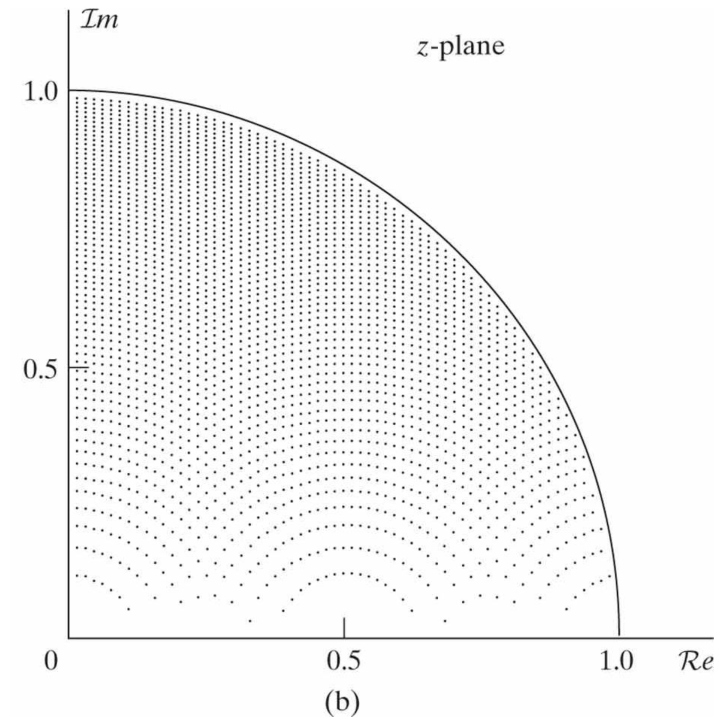


Effects of Coefficient Quantisation

IIR Digital Filters - Example

- 2nd order resonator filter $H(z) = \dfrac{1}{1 - 2r\cos(\theta)z^{-1} + r^2 z^{-2}}$





Pole-locations for 2nd-order IIR direct form resonator filter with
(a) four-bit quantization of coefficients, (b) seven-bit quantization.

Finite Precision Effects

Effects of Coefficient Quantisation

IIR Digital Filters - Example

- 2[nd] order resonator filter $H(z) = \dfrac{1}{1 - 2r\cos(\theta) z^{-1} + r^2 z^{-2}}$

- $r = 0.99 \qquad \theta = \dfrac{\pi}{4} \qquad H(z) = \dfrac{1}{1 - 1.4001 z^{-1} + 0.9801 z^{-2}}$

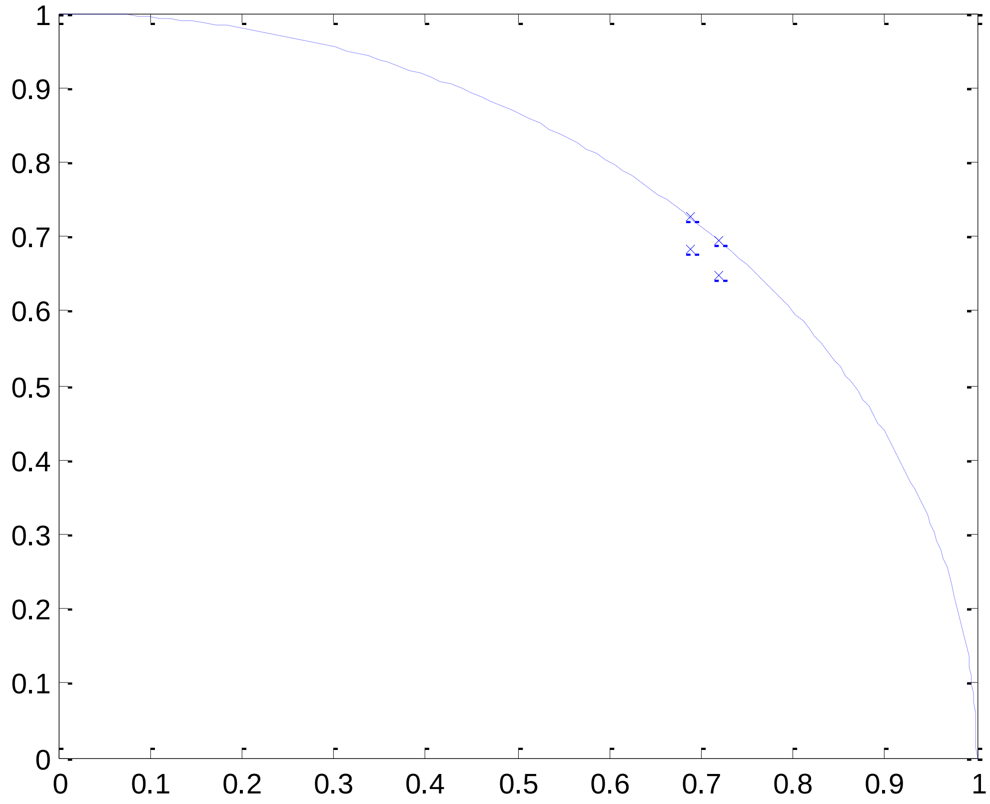
- Quantize coefficients to nearest multiple of 1/16 $\hat{H}(z) = \dfrac{1}{1 - 1.375 z^{-1} + z^{-2}}$

- **Unstable due to poles on unit circle**

Effects of Coefficient Quantisation

IIR Digital Filters - Example

- 2nd order resonator filter $\hat{H}(z) = \dfrac{1}{1 - 1.375z^{-1} + z^{-2}}$
- Rounding up or down to a multiple of 1/16 gives four choices for the location of the poles

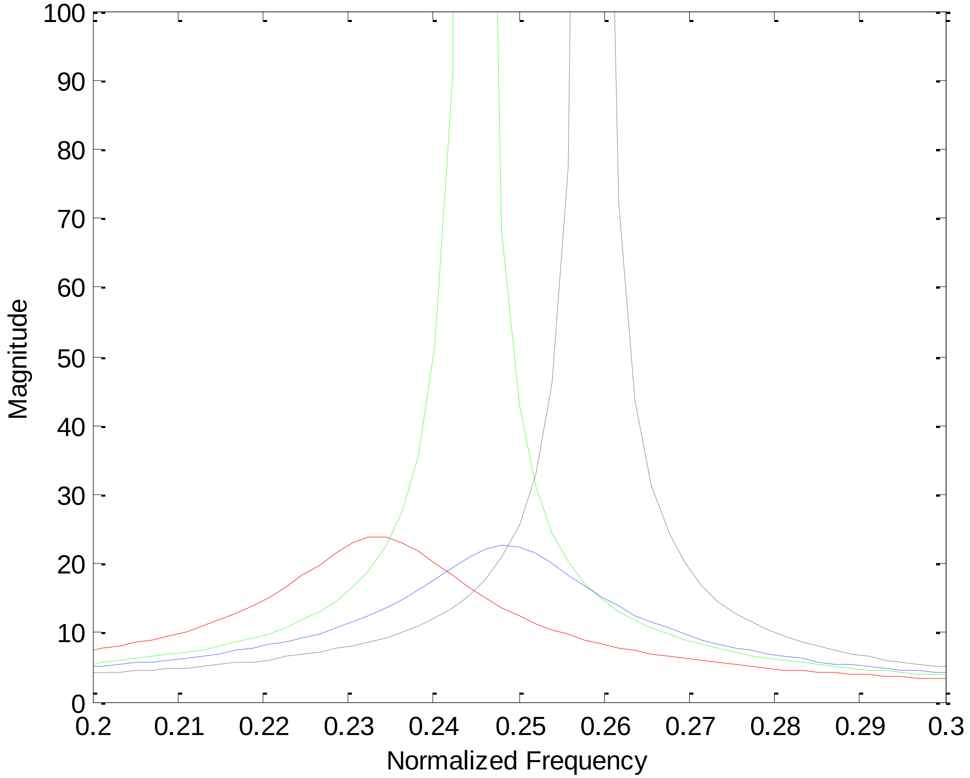


Finite Precision Effects

Effects of Coefficient Quantisation

IIR Digital Filters - Example

- 2nd order resonator filter $\hat{H}(z) = \dfrac{1}{1 - 1.375z^{-1} + z^{-2}}$
- Rounding up or down to a multiple of 1/16 gives four frequency responses



Finite Precision Effects

Zero Input Limit Cycles

Fixed Point Realisation of IIR Digital Filters

- With infinite precision arithmetic if the input becomes zero the output will decay to zero after a certain number of samples (assuming the filter is stable)

- With fixed point arithmetic the output may continue to oscillate indefinitely with a periodic pattern while the input remains zero

- Successive rounding-off or truncation of products in an iterated difference equation can create such repeating patterns

- Rounding of data stored in feedback path is a non-linear effect

Zero Input Limit Cycles

Example

$$H(z) = \frac{1}{1+0.75z^{-1}} \qquad y[n] = x[n] - 0.75y[n-1]$$

- Impulse response
  {1, -0.75, 0.5625, -0.4219 , 0.3164…}

Decays to zero

- Rounding y[n] to nearest multiple of 0.25 gives
  {1, -0.75, 0.5, -0.25, 0.25, -0.25, 0.25, …}

Oscillation

Finite Precision Effects

Zero Input Limit Cycles

Does it matter ?

- Suppose that a speech signal is sampled, filtered by a digital filter, and then converted back to an acoustic signal using a D/A converter
- If the filter suffers from periodic limit cycles whenever the input is zero an audible tone would be present (due to the oscillating output)

Implementation Complexity

How to reduce it

1. Reduce the filter order.
2. Quantize the coefficients to fixed word-length.
3. Replace multipliers by additions of power-of-two shifted values. Note that power-of-two multipliers are simple bit shifts. e.g. $45x = 32x + 8x + 4x + x$
4. Replace multipliers with signed power-of-two terms. e.g. $31x = 32x - x$
5. Using other representations allows further savings. e.g. $45x = (8+1)(4+1)x$
6. For FIR filters, the transpose structure allows all the multipliers to be implemented together in one multiplier block. This can offer further savings.
7. Designing filters as a cascade and/or sum of simple sections can also help reduce complexity.

FIR vs. IIR

FIR Advantages relative to IIR

- FIR filters may be linear phase
- FIR filters have guaranteed stability, while IIR filters require poles in the unit-circle
- FIR filters are less sensitive to coefficient quantisation (IIR filters can become unstable)
- FIR filters do not suffer from limit cycle problems

FIR Disadvantages relative to IIR

- FIR filters need higher filter order (complexity) for comparable performance