EE 508
Lecture 24

Sensitivity Functions

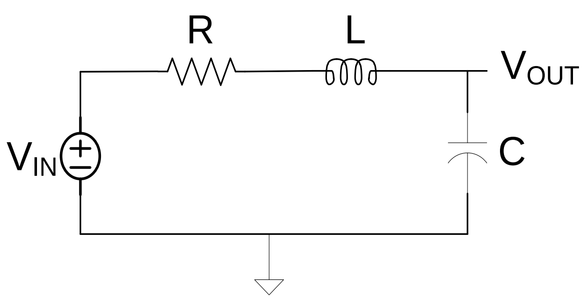
- Predistortion and Calibration

Review from last time

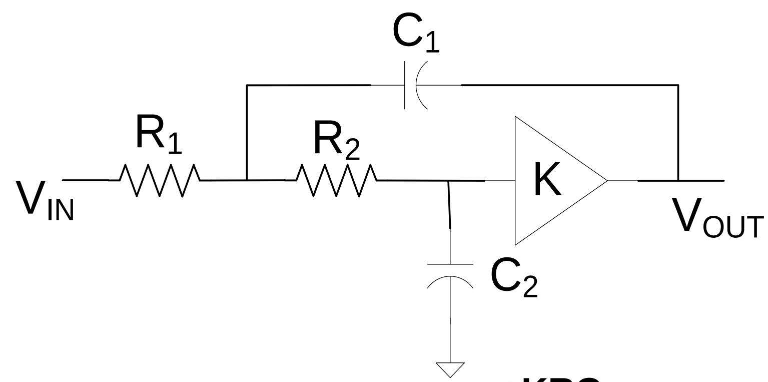
Sensitivity Comparisons

Consider 5 second-order lowpass filters

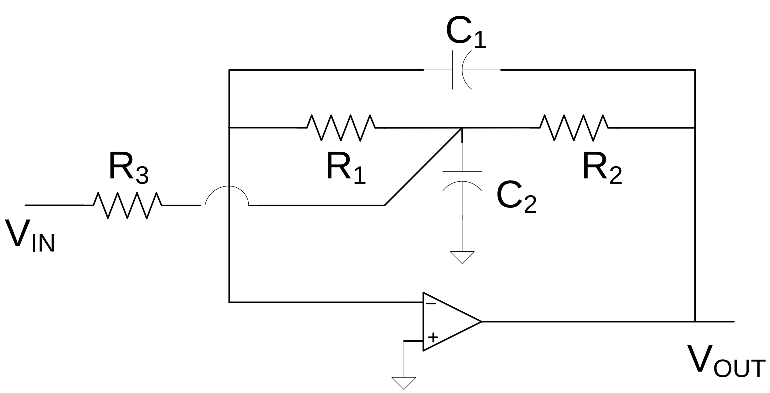
(all can realize same T(s) within a gain factor)



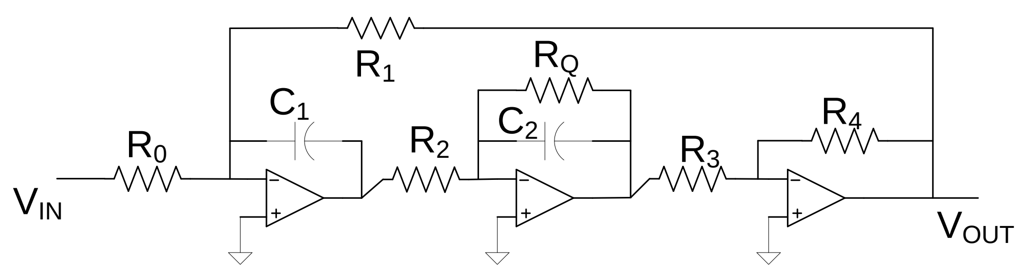
Passive RLC



+KRC



Bridged-T Feedback



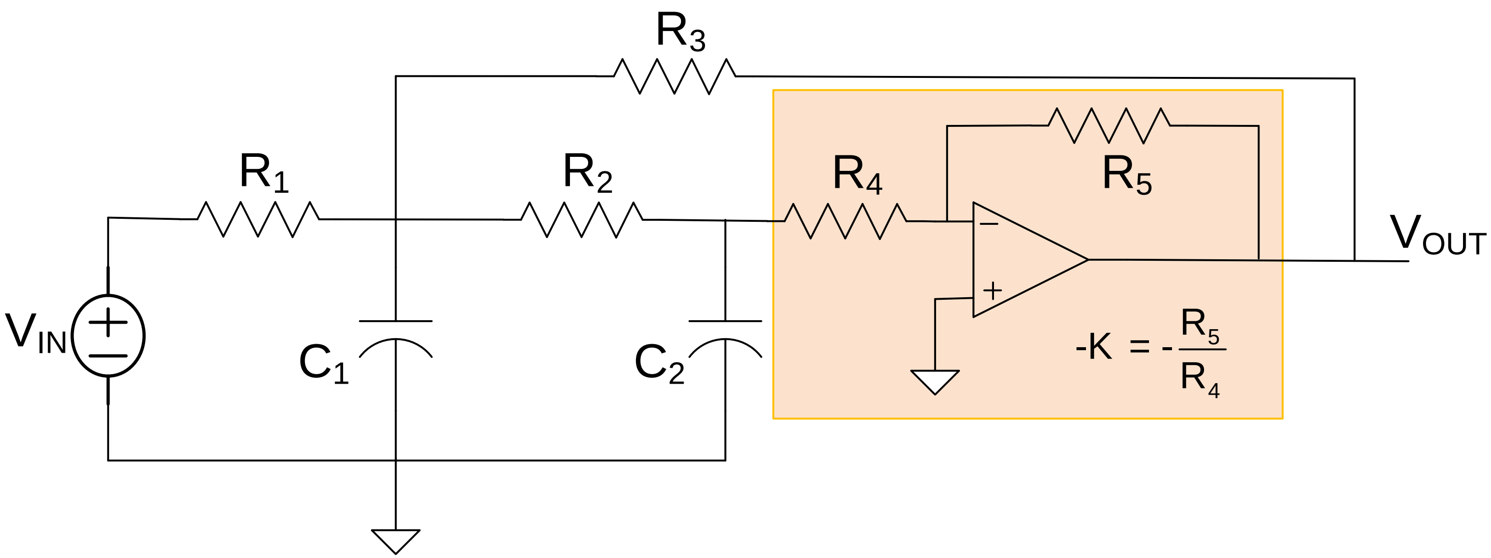
Two-Integrator Loop

Review from last time

Sensitivity Comparisons

Consider 5 second-order lowpass filters

(all can realize same T(s) within a gain factor)



-KRC Lowpass

Review from last time

Passive Sensitivity Comparisons

	$\left\|S_x^{\omega_0}\right\|$	$\left\|S_x^{Q}\right\|$
Passive RLC	$\leq 1/2$	**1,1/2**
+KRC		
Equal R, Equal C (K=3-1/Q)	**0,1/2**	**Q, 2Q, 3Q**
Equal R, K=1 ($C_1=4Q^2C_2$)	**0,1/2**	**0,1/2, $2Q^2$**
Bridged-T Feedback	**0,1/2**	**1/3,1/2, 1/6**
Two-Integrator Loop	**0,1/2**	**1,1/2, 0**
-KRC	**less than or equal to 1/2**	**less than or equal to 1/2**

Substantial Differences Between (or in) Architectures

Review from last time

Where we are at with sensitivity analysis:

Considered a group of five second-order filters

Passive Sensitivity Analysis

- Closed form expressions were obtained for ω_0 and Q
- Tedious but straightforward calculations provided passive sensitivities directly from the closed form expressions ???

Active Sensitivity Analysis

- Closed form expressions for ω_0 and Q are very difficult or impossible to obtain

If we consider higher-order filters

Passive Sensitivity Analysis

- Closed form expressions for ω_0 and Q are very difficult or impossible to obtain for many useful structures

Active Sensitivity Analysis

- Closed form expressions for ω_0 and Q are very difficult or impossible to obtain

Need some better method for obtaining sensitivities when closed-form expressions are difficult or impractical to obtain or manipulate !!

Review from last time

Relationship between active pole sensitivities and ω_0 and Q sensitivities

Define $D(s)=D_0(s)+t\,D_1(s)$ (from bilinear form of T(s))

Recall: $$\mathcal{S}_{\tau}^{p} = \frac{-D_1(p)}{\left.\frac{\partial D(s)}{\partial s}\right|_{s=p,\tau=0}}$$

Theorem: $$\Delta p \simeq \tau \mathcal{S}_{\tau}^{p}$$

Theorem: $$\Delta\alpha \simeq \tau \,\text{Re}\left(\mathcal{S}_{\tau}^{p}\right)$$ $$\Delta\beta \simeq \tau \,\text{Im}\left(\mathcal{S}_{\tau}^{p}\right)$$

Theorem:

$$\frac{\Delta\omega_0}{\omega_0} \simeq \frac{1}{2Q}\frac{\Delta\alpha}{\omega_0} + \sqrt{1-\frac{1}{4Q^2}}\frac{\Delta\beta}{\omega_0}$$

$$\frac{\Delta Q}{Q} \simeq -2Q\left(1-\frac{1}{4Q^2}\right)\frac{\Delta\alpha}{\omega_0} + \sqrt{1-\frac{1}{4Q^2}}\frac{\Delta\beta}{\omega_0}$$

Claim: These theorems, with straightforward modification, also apply to other parameters (R, C, L, K, …) where, $D_0(s)$ and $D_1(s)$ will change since the parameter is different

Review from last time

Active Sensitivity Comparisons

	$\frac{\Delta\omega_0}{\omega_0}$	$\frac{\Delta Q}{Q}$
Passive RLC		
+KRC		
Equal R, Equal C (K=3-1/Q)	$-\frac{1}{2}\left(3-\frac{1}{Q}\right)^2\tau\omega_0$	$-\frac{1}{2}\left(3-\frac{1}{Q}\right)^2\tau\omega_0$
Equal R, K=1 ($C_1=4Q^2C_2$)	$-Q\tau\omega_0$	$Q\tau\omega_0$
Bridged-T Feedback	$-\frac{3}{2}Q\tau\omega_0$	$\frac{1}{2}Q\tau\omega_0$
Two-Integrator Loop	$-\tau\omega_0$	$4Q\tau\omega_0$
-KRC	$\frac{5}{2}Q\tau\omega_0$	$25Q^3\tau\omega_0$

Substantial Differences Between Architectures

Review from last time

Are these passive sensitivities acceptable?

	$\left|S_x^{\omega_0}\right|$	$\left|S_x^{Q}\right|$
Passive RLC	$\leq 1/2$	**1,1/2**
+KRC		
Equal R, Equal C (K=3-1/Q)	**0,1/2**	**Q, 2Q, 3Q**
Equal R, K=1 ($C_1=4Q^2C_2$)	**0,1/2**	**0,1/2, $2Q^2$**
Bridged-T Feedback	**0,1/2**	**1/3,1/2, 1/6**
Two-Integrator Loop	**0,1/2**	**1,1/2, 0**
-KRC	**less than or equal to 1/2**	**less than or equal to 1/2**

Review from last time

Are these sensitivities acceptable?

Passive Sensitivities:

$$\frac{\Delta\omega_0}{\omega_0} \simeq S_x^{\omega_0} \frac{\Delta x}{x}$$

In integrated circuits, Δ R/R and Δ C/C due to process variations can be K 30% or larger due to process variations

Many applications require $\Delta\omega_0/\omega_0 < .001$ or smaller and similar requirements on ΔQ/Q

Even if sensitivity is around ½ or 1, variability is often orders of magnitude too large

Active Sensitivities:

All are proportional to $\tau\omega_0$

Some architectures much more sensitive than others

Can reduce $\tau\omega_0$ by making GB large but this is at the expense of increased power and even if power is not of concern, process presents fundamental limits on how large GB can be made

What can be done to address these problems?

1. Predistortion

Design circuit so that <u>after</u> component shift, correct pole locations are obtained

Predistortion is generally used in integrated circuits to remove the bias associated with inadequate amplifier bandwidth

Tedious process after fabrication since depends on individual components

Temperature dependence may not track

Difficult to maintain over time and temperature

Over-ordering will adversely affect performance

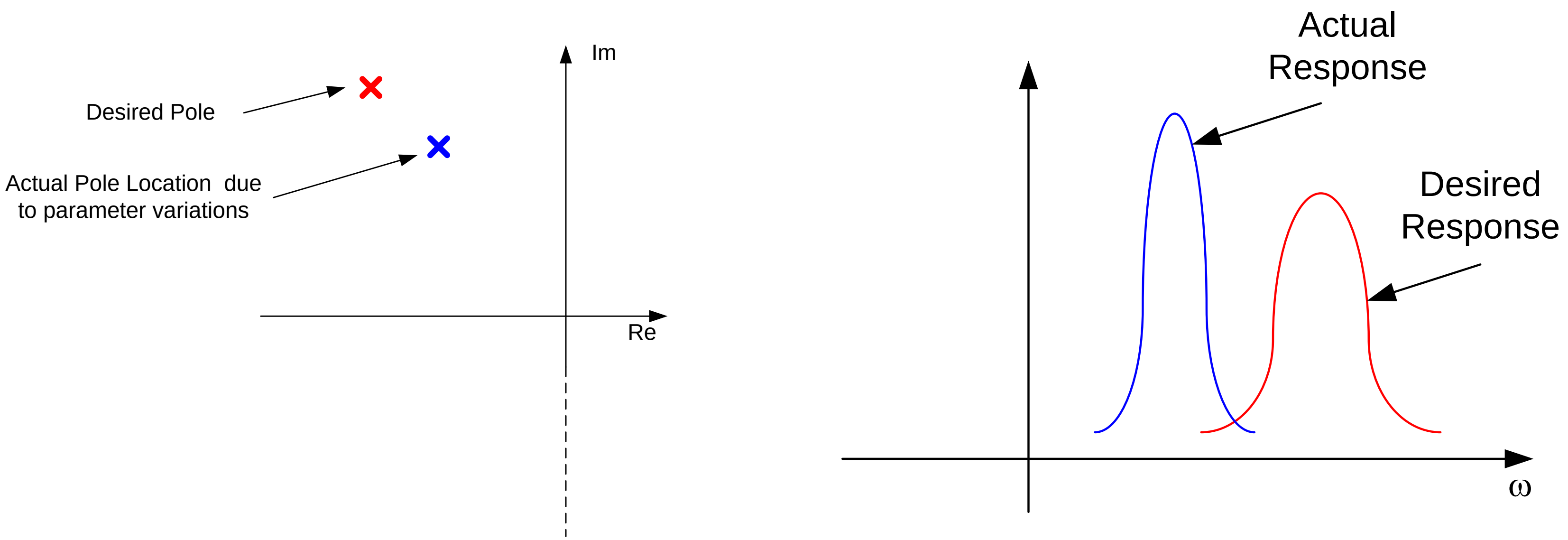
Seldom will predistortion alone be adequate to obtain acceptable performance

Bell Labs did to this in high-volume production (STAR Biquad)

What can be done to address these problems?

1. Predistortion

Design circuit so that <u>after</u> component shift, correct pole locations are obtained

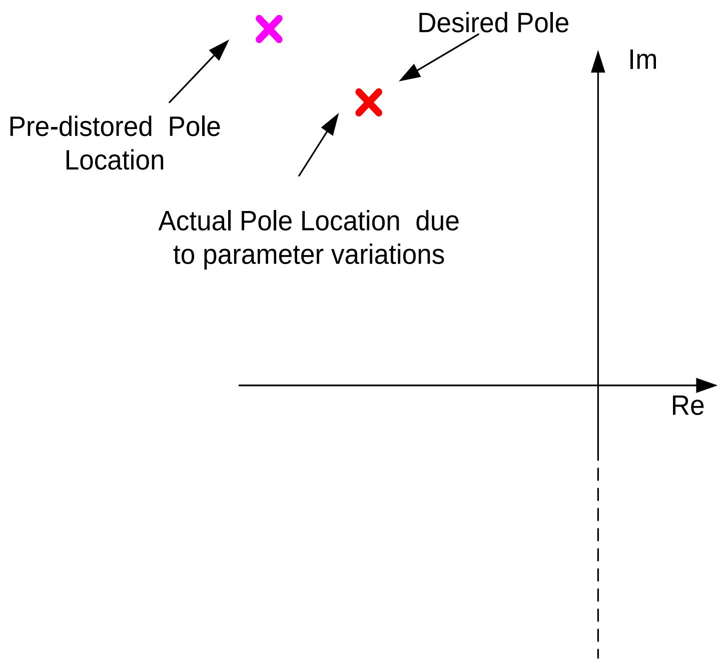


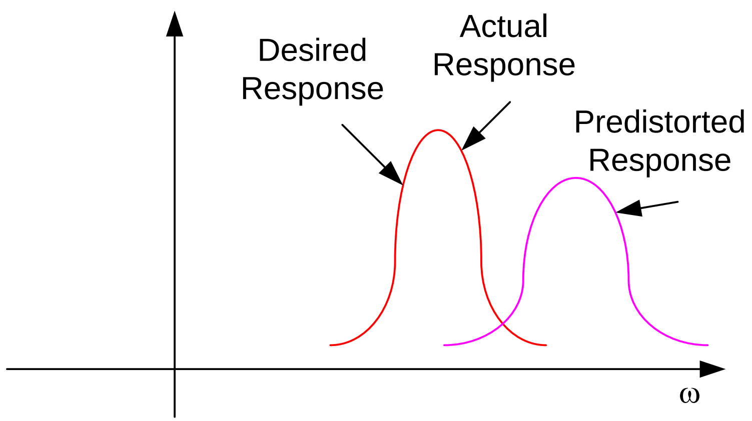
Pole shift due to parametric variations (e.g. inadequate GB)

What can be done to address these problems?

1. Predistortion

Design circuit so that after component shift, correct pole locations are obtained



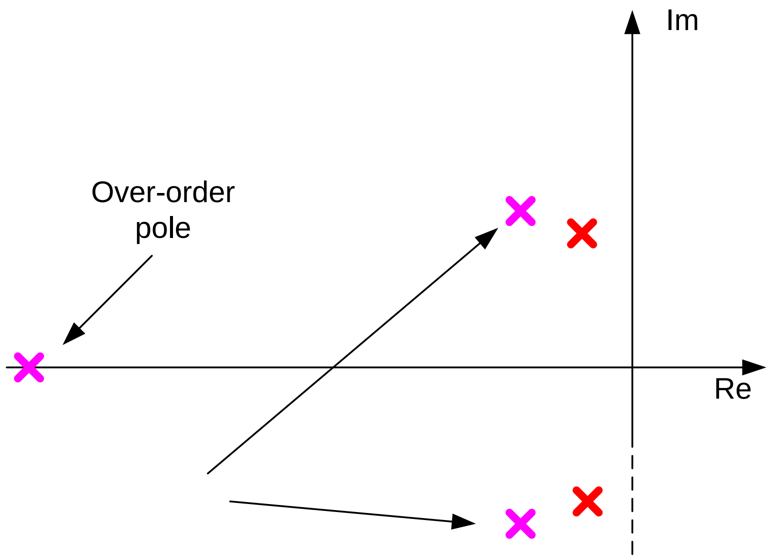


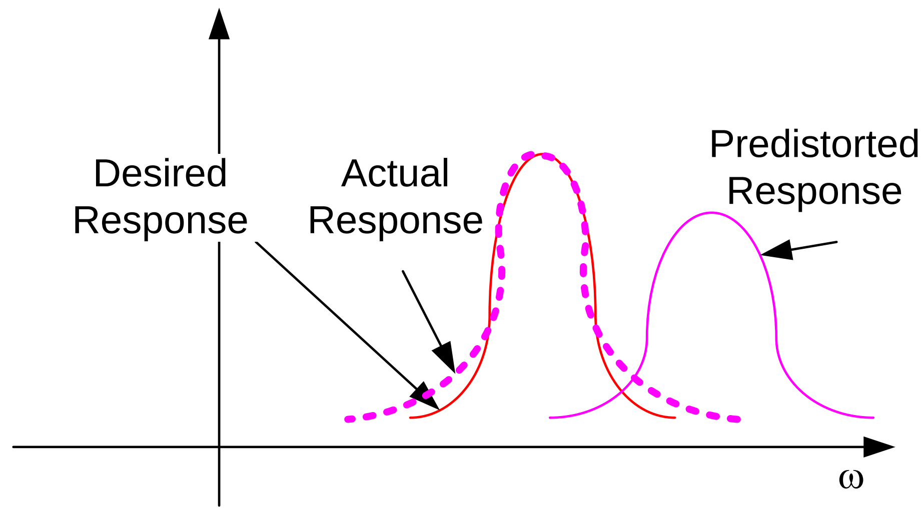
Pre-distortion concept

What can be done to address these problems?

1. Predistortion

Design circuit so that <u>after</u> component shift, correct pole locations are obtained





Over-ordering Limitations with Pre-distortion

Parasitic Pole Affects Response

Predistortion almost always done even if benefits only modest

Not effective is significant deviations exist before predistortion

What can be done to address these problems?

2. Trimming

a) Functional Trimming

- trim parameters of actual filter based upon measurements
- difficult to implement in many structures
- manageable for cascaded biquads

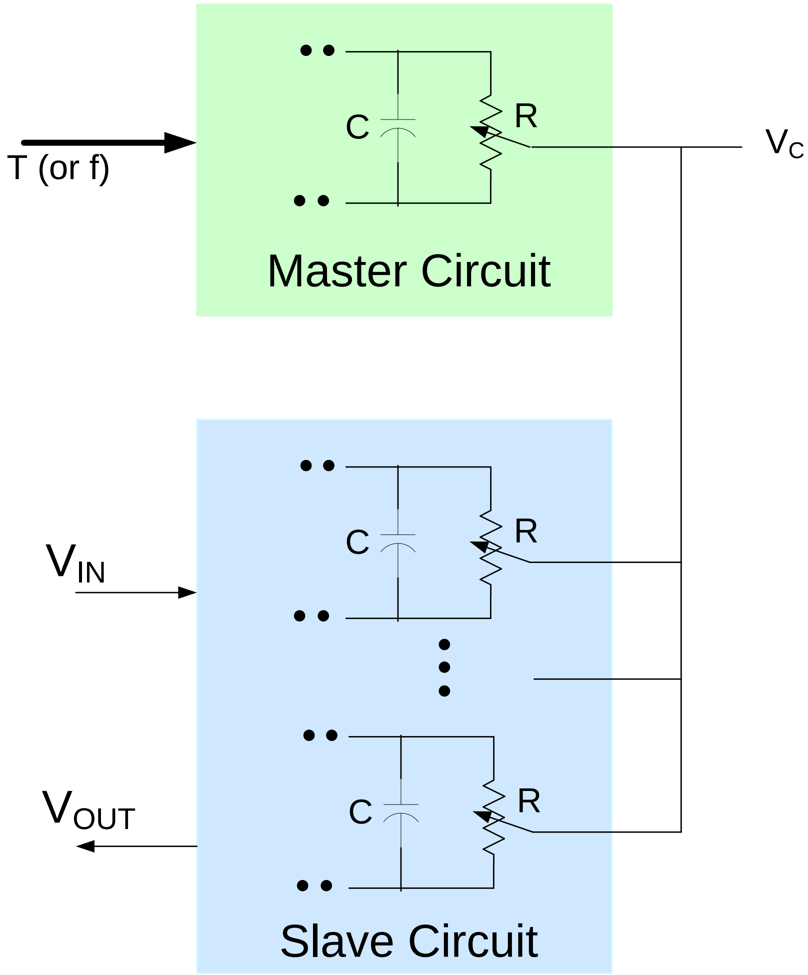
b) Deterministic Trimming (much preferred)

- Trim component values to their ideal value
  - Continuous-trims of resistors possible in some special processes
  - Continuous-trim of capacitors is more challenging
  - Link trimming of Rs or Cs is possible with either metal or switches
- If all components are ideal, the filter should also be ideal
  - R-trimming algorithms easy to implement
  - Limited to unidirectional trim
  - Trim generally done at wafer level for laser trimming, package for link trims
- Filter shifts occur due to stress in packaging and heat cycling

c) Master-slave reference control (depends upon matching in a process)

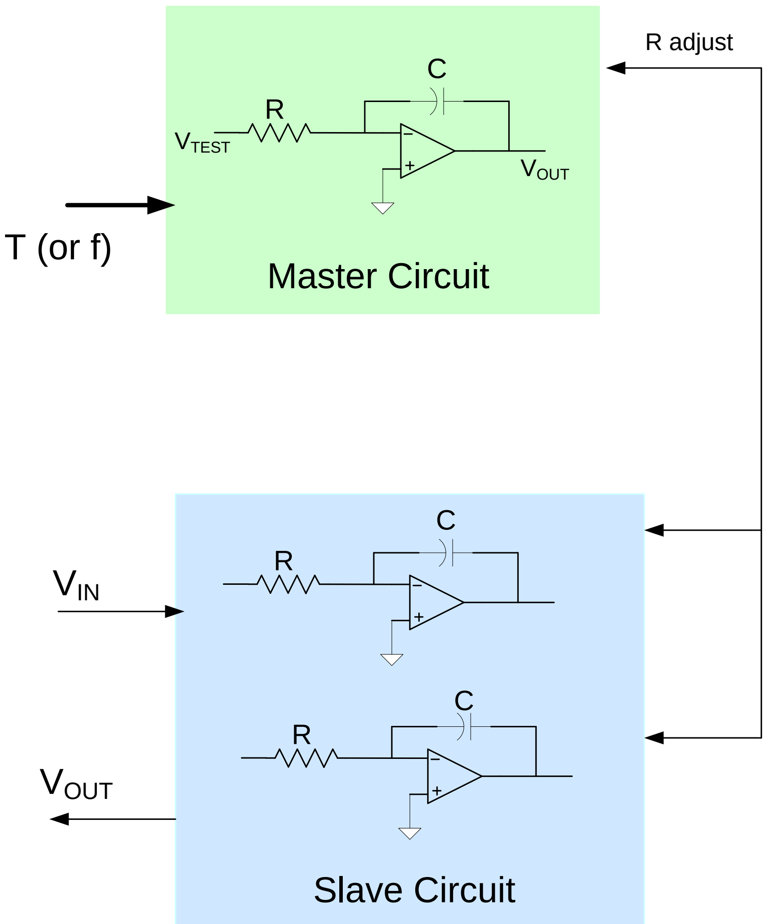
- Can be implemented in discrete or integrated structures
- Master typically frequency or period referenced
- Most effective in integrated form since good matching possible
- Widely used in integrated form

Master-slave Control (depends upon matching in a process)



- Automatically adjust R in the Master Circuit to match RC to T
- Rely on matching to match RC products in Slave Circuit to T
- Matching can be very good (1% or 0.1% or better)

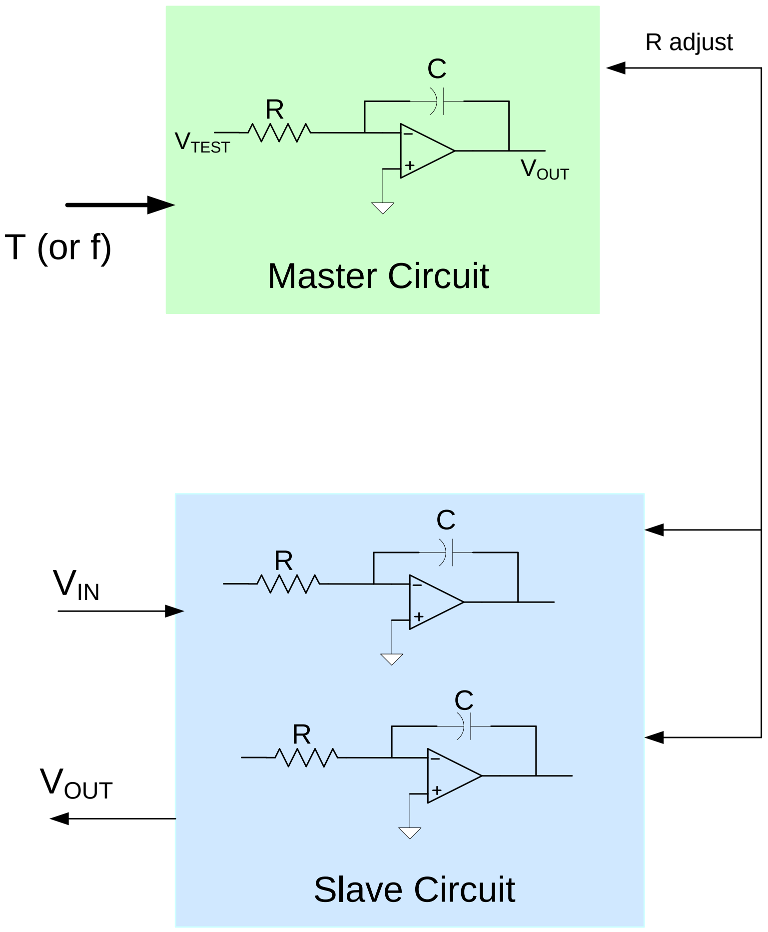
Master-slave Example:



$$T(s) = \frac{V_{OUT}}{V_{TEST}} = -\frac{1}{RCs}$$

- Key parameter of integrator is unity gain frequency $I_0 = 1/RC$
- Adjust R in Master Circuit so that $I_0 = 1$ at the input frequency f
- With matching, unity gain frequency of all integrators in Slave Circuit will also be 1

Master-slave Example:



$$T(s)=\frac{V_{OUT}}{V_{TEST}}=-\frac{1}{RCs}$$

$$T_{ACT}(s)=\frac{V_{OUT}}{V_{TEST}}=-\frac{1}{RCs+\tau\left(s+RCs^2\right)}$$

- Over-ordering will limit accuracy of master-slave approach even if unity gain frequency of master circuit is precisely obtained
- Technique is often used to maintain good control of effective RC products

What can be done to address these problems?

3. Select Appropriate Architecture

Helps a lot

Best architectures are not known

Performance of good architectures often not good enough

What can be done to address these problems?

4. Different Approach for Filter Implementation

- **Frequency Referenced Filters**
  Switched-Capacitor Filters
- **DSP- Based Filter Implementation**
- **Other Niche Methods**

Summary of Sensitivity Observations

- Sensitivity varies substantially from one implementation to another

- Variability too high, even with low sensitivity, for more demanding applications

- Methods of managing high variability
  - Select good structures
  - Trimming
    - Functional
    - Deterministic
  - Predistortion
    - In particular, for active sensitivities
    - Useful but not a total solution
  - Frequency Referenced Techniques
    - Master-Slave Control
      - Depends upon matching
      - Can self-trim or self-compensate
    - Switched-Capacitor Filters
    - AD/digital filter/D/A
  - Alternate Design Approach
    - Other methods

Filter Design Process

Establish Specifications

- possibly $T_D(s)$ or $H_D(z)$
- magnitude and phase characteristics or restrictions
- time domain requirements

↓

Approximation

- obtain acceptable transfer functions $T_A(s)$ or $H_A(z)$
- possibly acceptable realizable time-domain responses

↓

Where we are at →

Synthesis

- build circuit or implement algorithm that has response close to $T_A(s)$ or $H_A(z)$
- actually realize $T_R(s)$ or $H_R(z)$

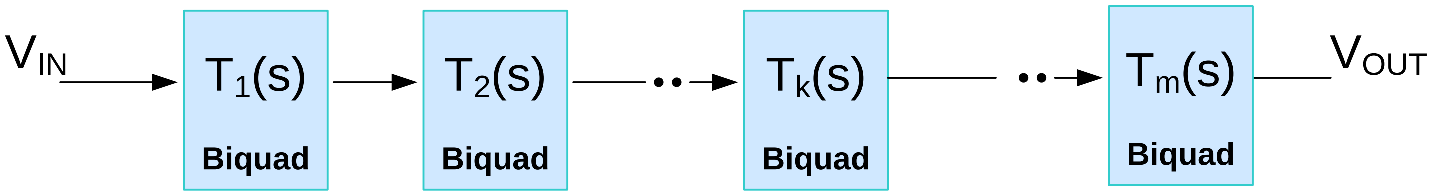
↓

Filter

Filter Design/Synthesis Considerations

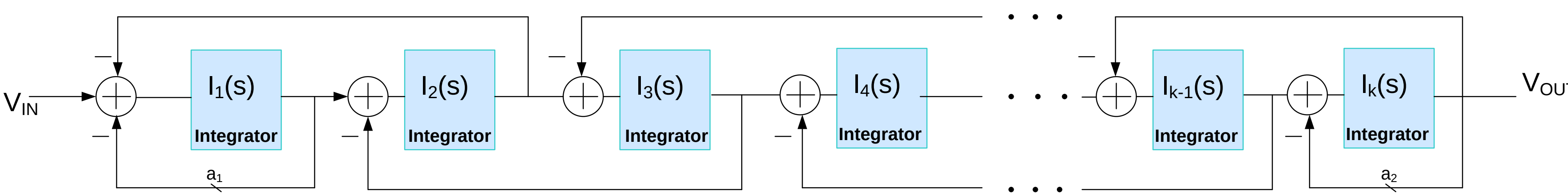
Most designs today use one of the following three basic architectures

Cascaded Biquads

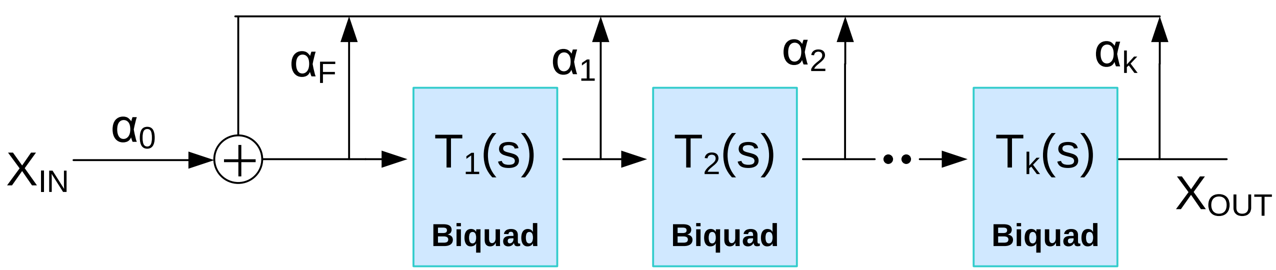


$$T(s) = T_1 T_2 \bullet\bullet T_m$$

Leapfrog

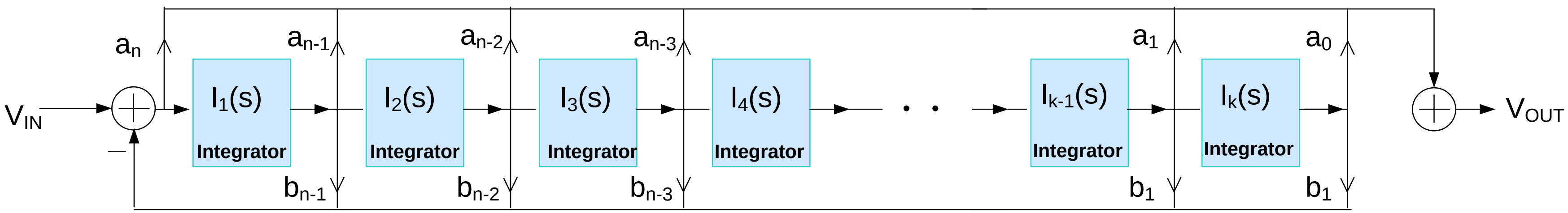


Multiple-loop Feedback – One type shown (less popular)



Filter Design/Synthesis Considerations

Multiple-loop Feedback – Another type



$$X = V_{IN} - X \bullet \sum_{k=1}^{n} b_{n-k} \left(\frac{I_0}{s}\right)^k$$

$$V_{OUT} = X \bullet \sum_{k=0}^{n} a_{n-k} \left(\frac{I_0}{s}\right)^k$$

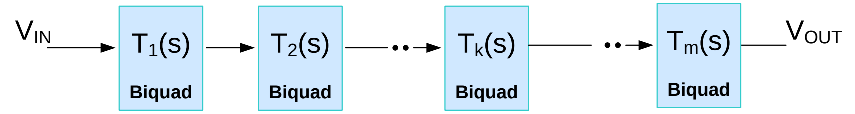
$$T(s) = \frac{\sum_{k=0}^{n} a_{n-k} \left(\frac{I_0}{s}\right)^k}{1 + \sum_{k=1}^{n} b_{n-k} \left(\frac{I_0}{s}\right)^k}$$

$$T(s) = \frac{\sum_{k=0}^{n} a_{n-k} I_0^k s^{n-k}}{s^n + \sum_{k=1}^{n} b_{n-k} I_0^k s^{n-k}}$$

- Termed the direct synthesis method
- Directly implements the coefficients in the numerator and denominator
- Approach followed in the Analog Computers
- Not particularly attractive from an overall performance viewpoint

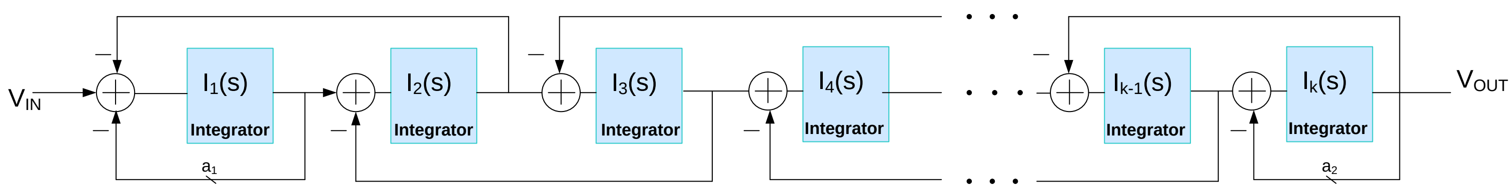
Filter Design/Synthesis Considerations

Cascaded Biquads

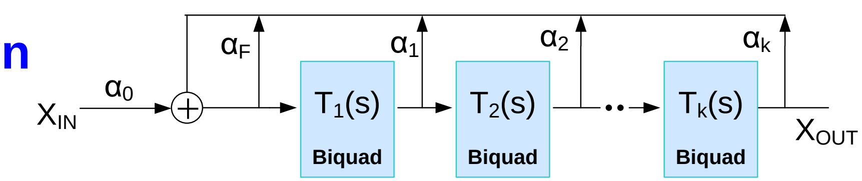


$$T(s) = T_1 T_2 \bullet\bullet T_m$$

Leapfrog



Multiple-loop Feedback – One type shown



Will study details of all three types of architectures later

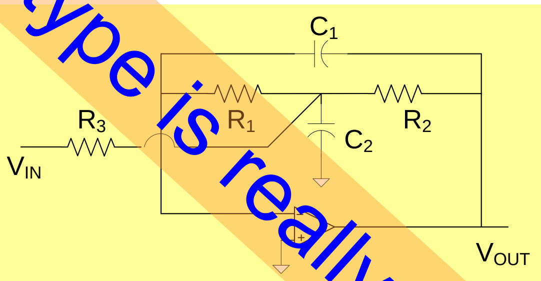
Observation: All filters are comprised of summers, biquads and integrators

Consider now the biquads

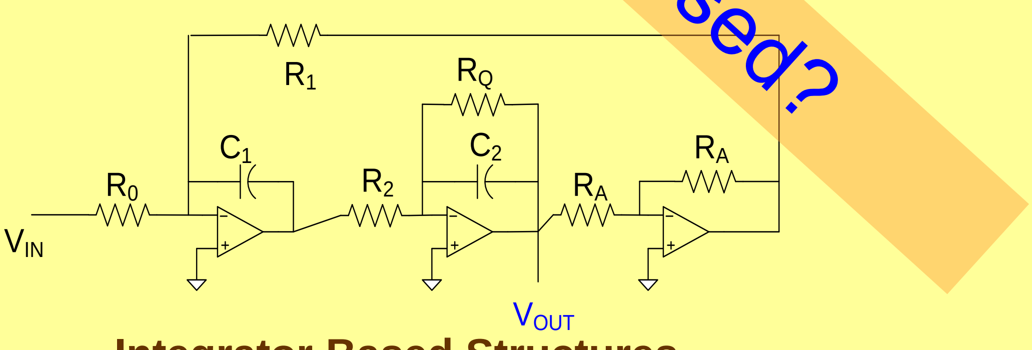
Biquad Filters Design Considerations

Several different Biquads were considered and other implementations exist

Sallen-Key Type (Dependent Sources)



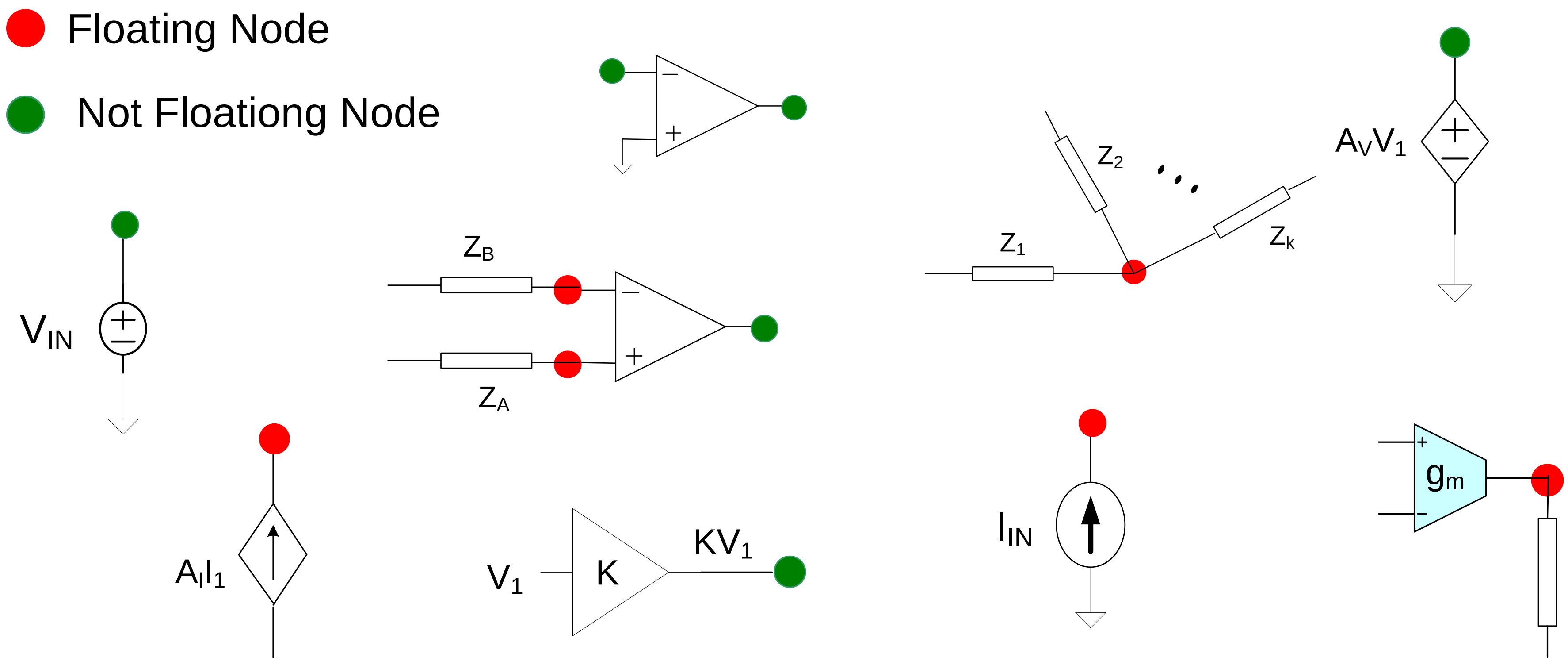
Infinite Gain Amplifiers



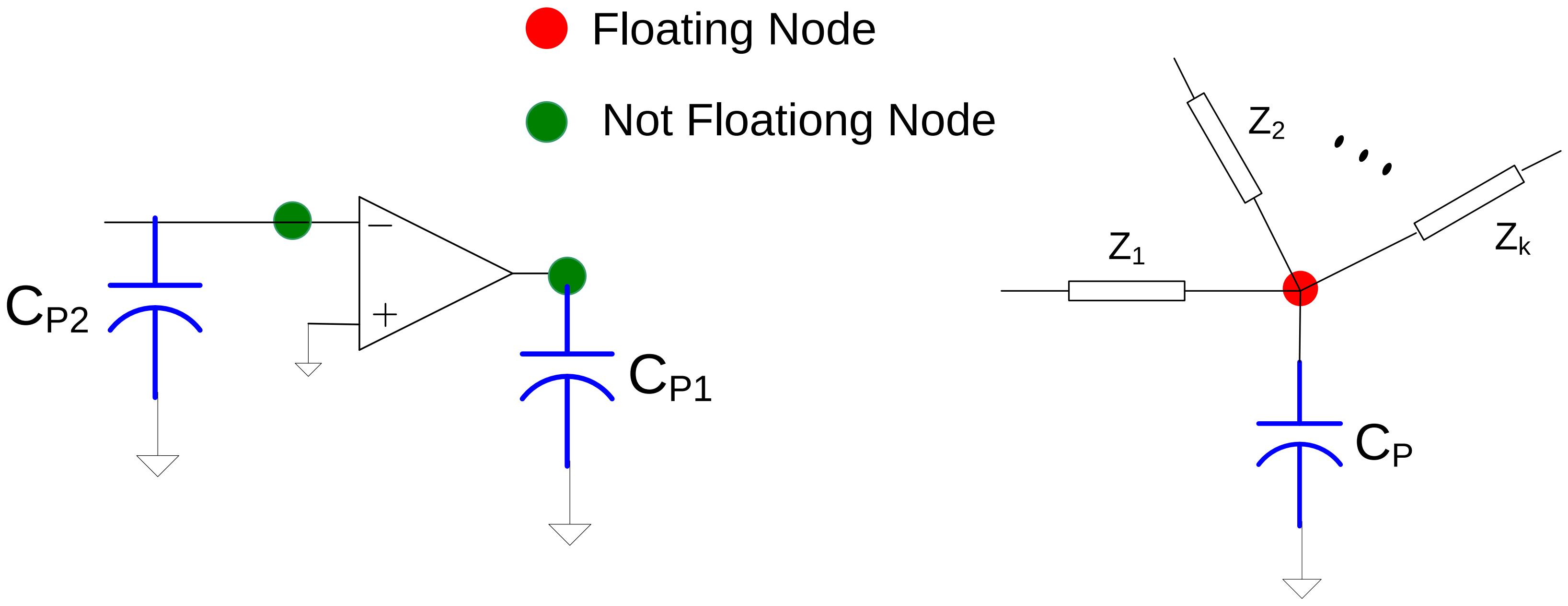
Integrator Based Structures

Floating Nodes

A node in a circuit is termed a floating node if it is not an output node of a ground-referenced voltage-output amplifier (dependent or independent), not connected to a ground-referenced voltage source, or not connected to a ground-referenced null-port



Parasitic Capacitances on Floating Nodes



Parasitic capacitances ideally have no affect on filter when on a non-floating node but directly affect transfer function when they appear on a floating node

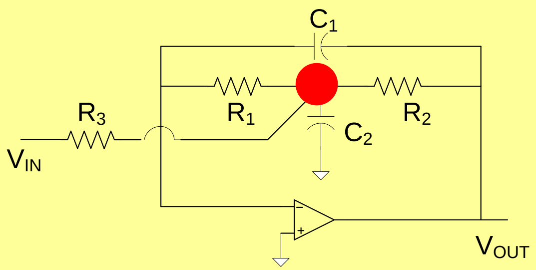
Parasitic capacitances are invariably large, nonlinear, and highly process dependent in integrated filters. Thus, it is difficult to build accurate integrated filters if floating nodes are present

Generally avoid floating nodes, if possible, in integrated filters

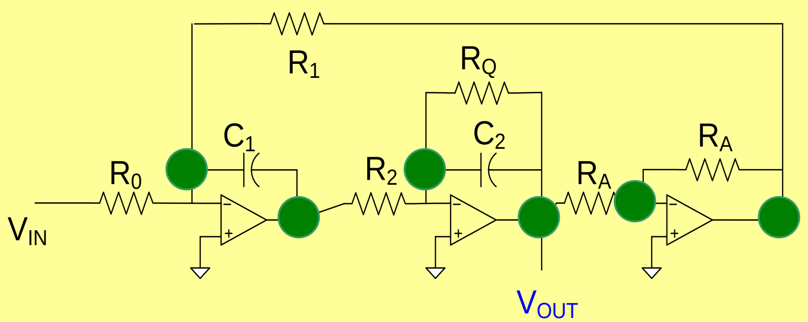
Which type of Biquad is really used?

Not Floationg Node   Floating Node

Sallen-Key Type (Dependent Sources)



Infinite Gain Amplifiers



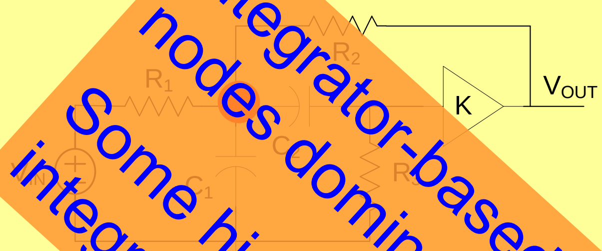
Integrator Based Structures

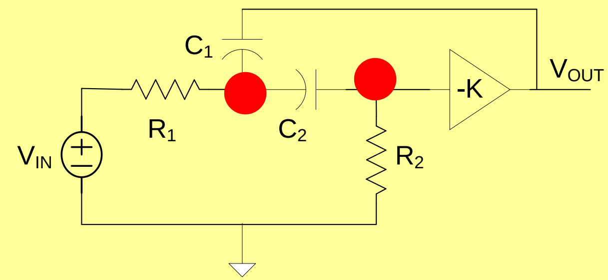
Which type of Biquad is really used?

● Not Floatong Node ● Floating Node

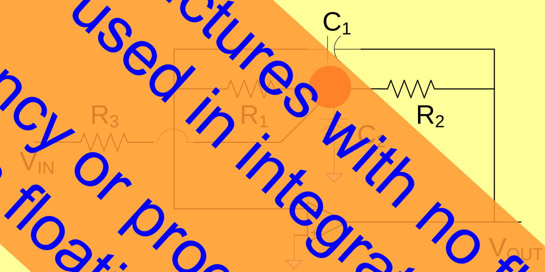
Integrator-based structures with no floating nodes dominantly used in integrated filters

Some high-frequency or programmable integrated filters with floating nodes are used

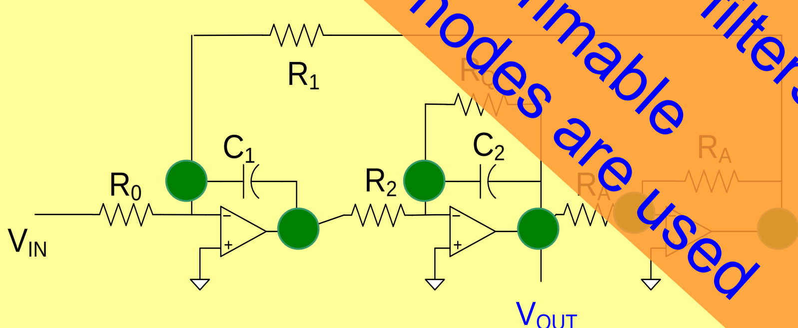




Sallen-Key type (Dependent Sources)

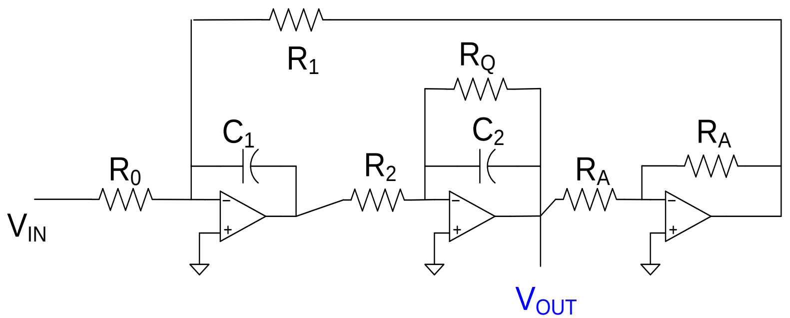


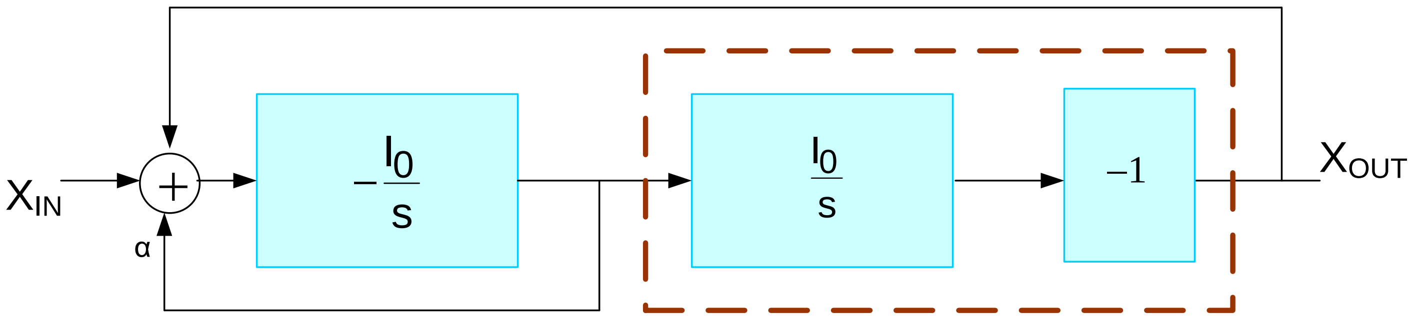
Infinite Gain Amplifiers

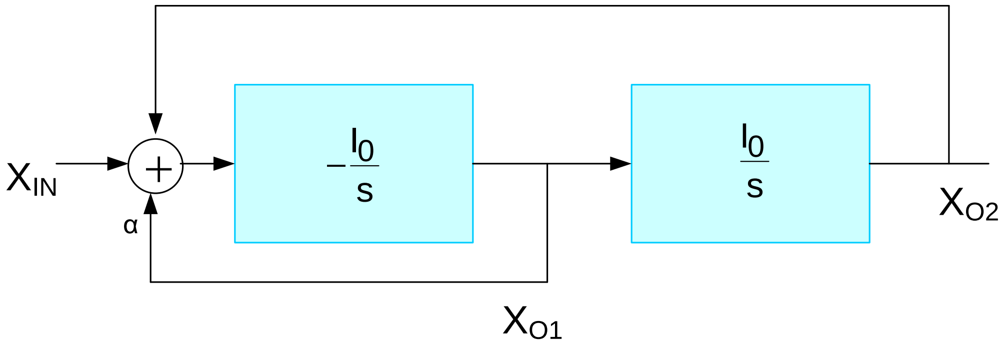


Integrator Based Structures

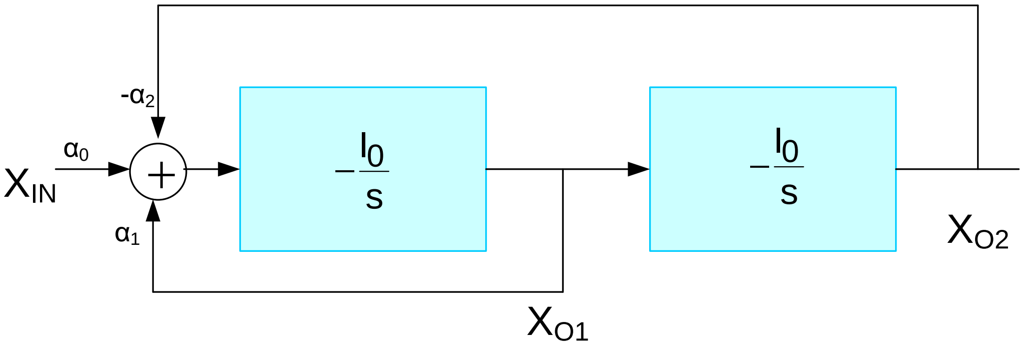
Integrator-based Biquads



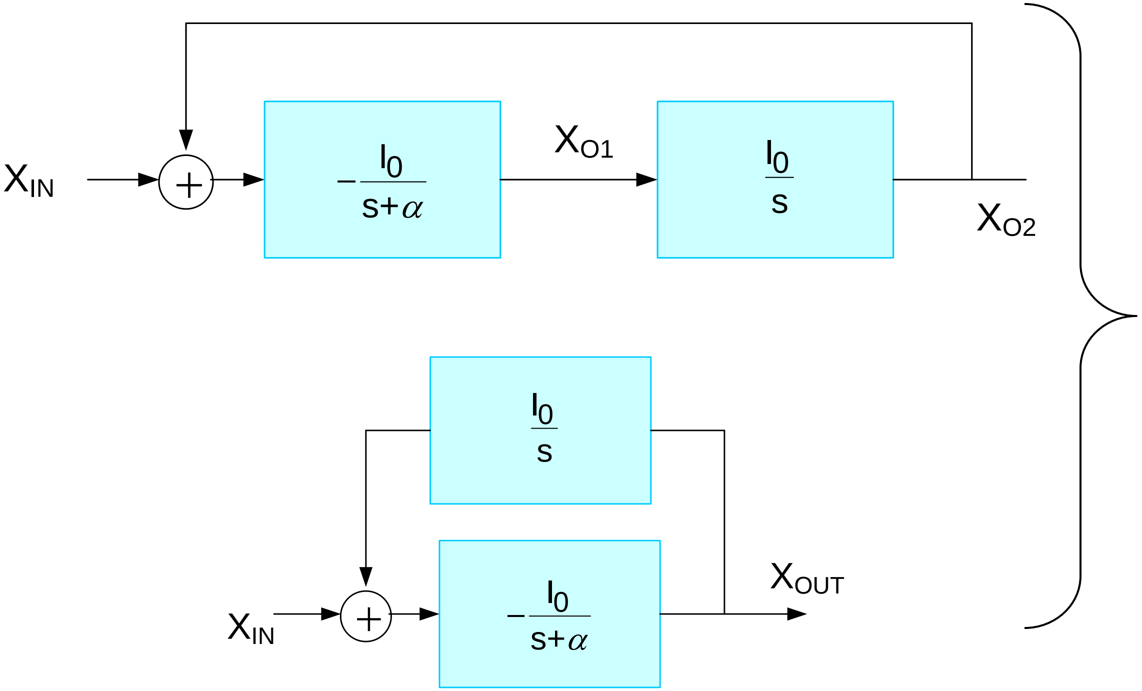




Integrator-based Biquads

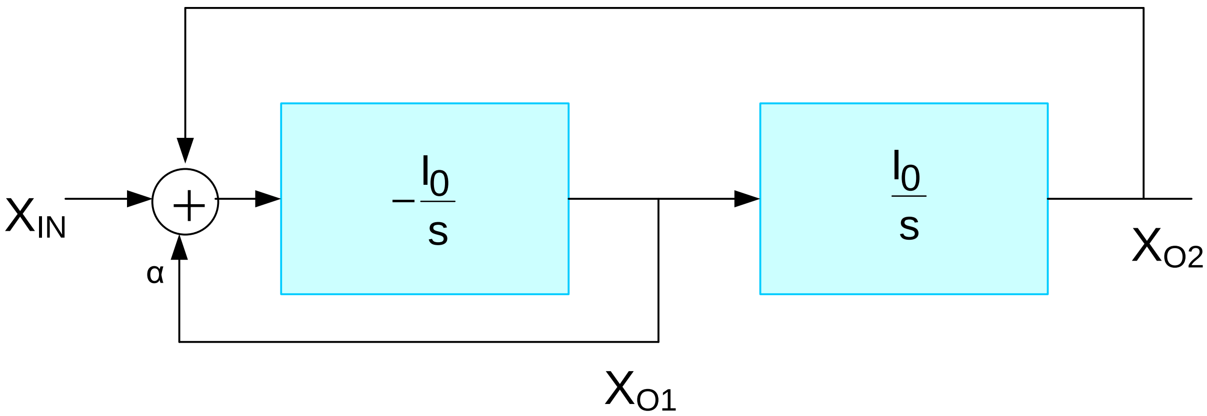


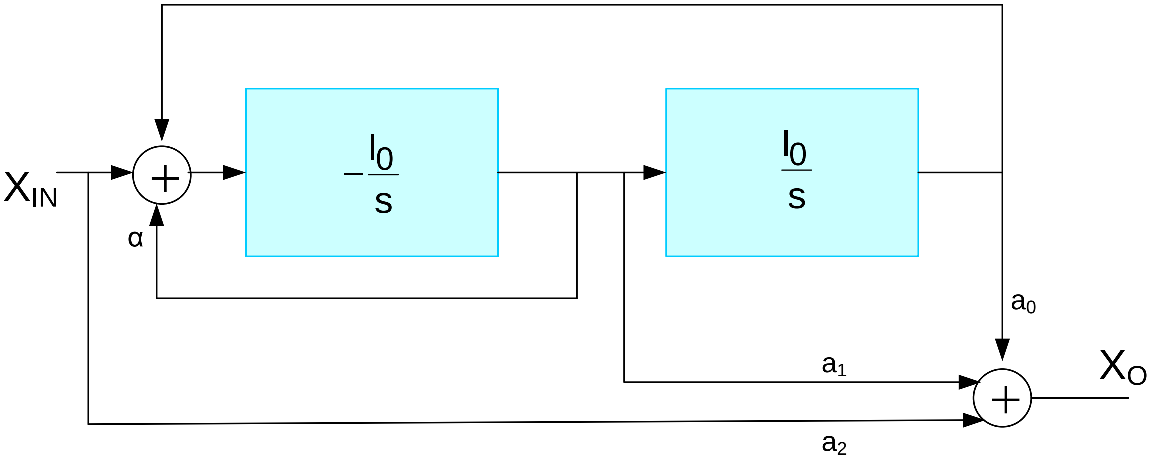
State Variable Biquad
(Alt KHN Biquad)



Integrator and lossy integrator in a loop

Integrator-based Biquads

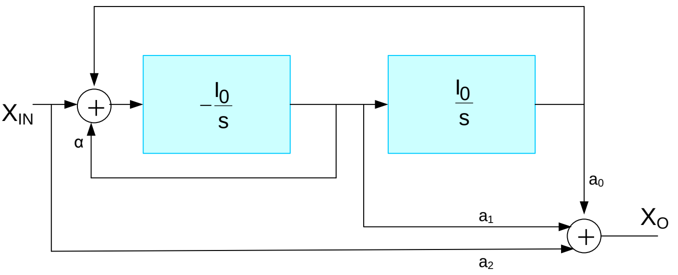


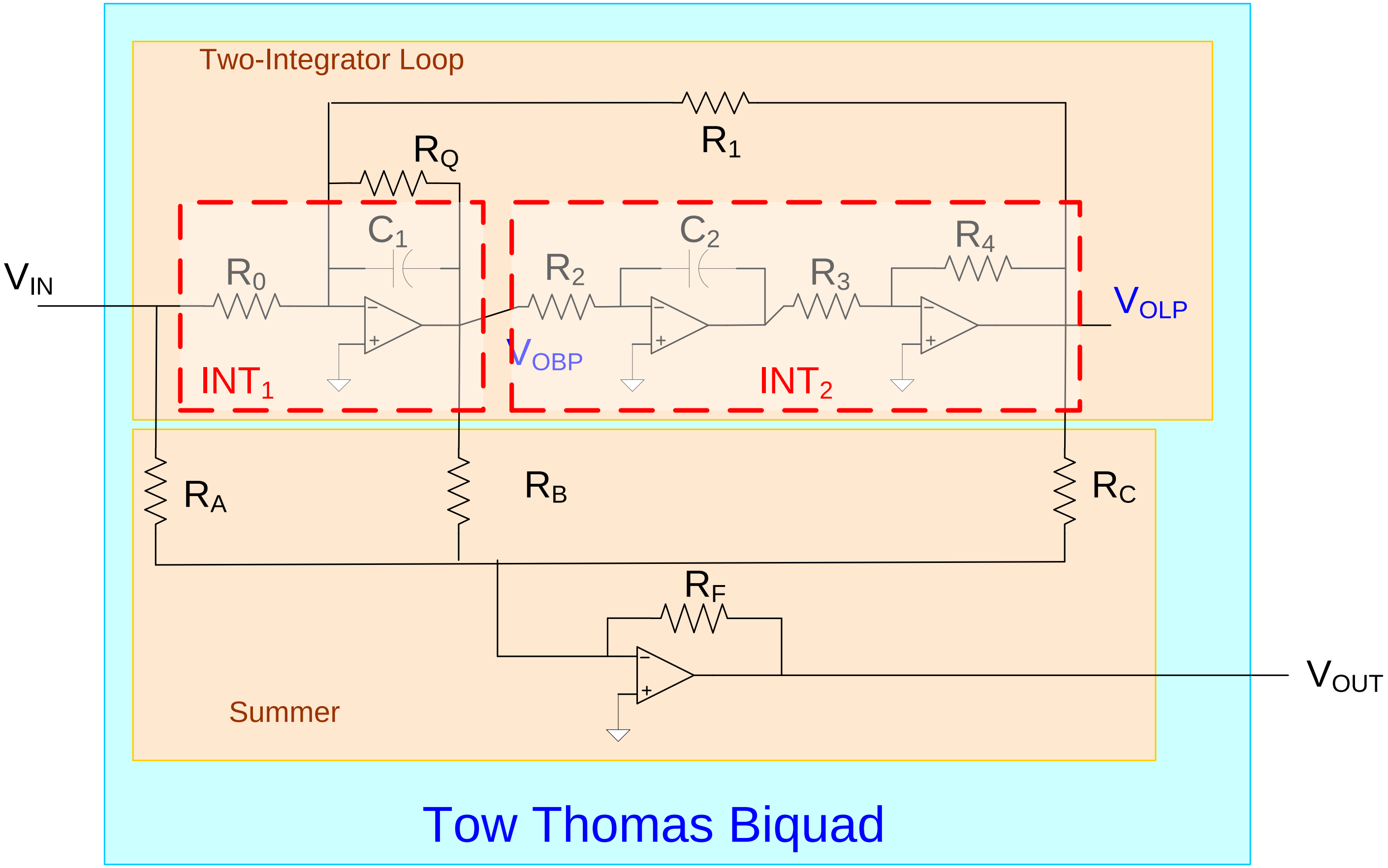


With arbitrary zero locations

Tow-Thomas Biquad

Integrator-based Biquads



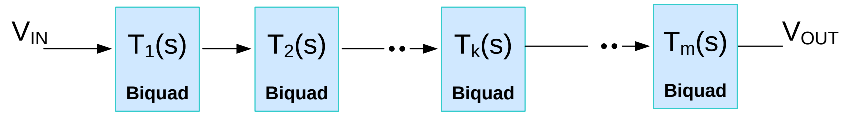


Integrator-based Biquads

- Integrator-based biquads all involve two integrators in a loop
- All integrator-based biquads discussed have no floating nodes
- Most biquads in integrated filters are based upon two integrator loop structures
- The summers are usually included as summing inputs on the integrators
- The loss can be combined with the integrator to form a lossy integrator
- Performance of the minor variants of the two integrator loop structures are comparable

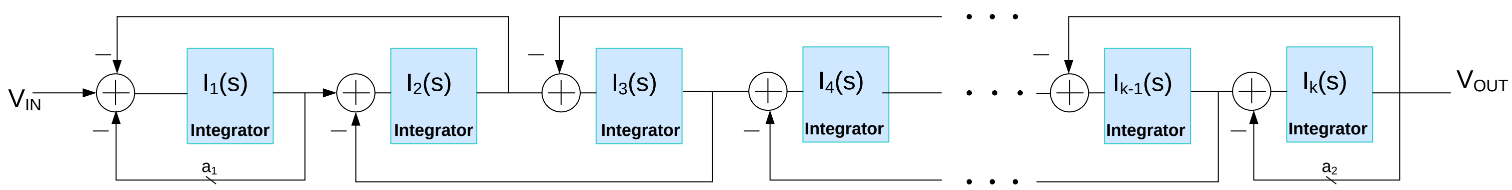
Filter Design/Synthesis Considerations

Cascaded Biquads

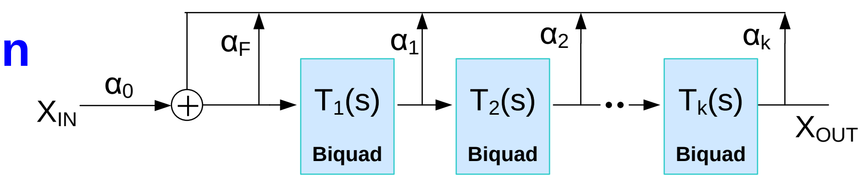


$$T(s) = T_1 T_2 \bullet\bullet T_m$$

Leapfrog



Multiple-loop Feedback – One type shown



Observation: All filters are comprised of summers, biquads and integrators

And biquads usually made with summers and integrators

Integrated filter design generally focused on design of integrators, summers, and amplifiers (Op Amps)

Will now focus on the design of integrators, summers, and op amps

End of Lecture 24