ECE1371 Advanced Analog Circuits
Lecture 5

ADVANCED $\Delta\Sigma$

Richard Schreier
`richard.schreier@analog.com`

Trevor Caldwell
`trevor.caldwell@utoronto.ca`

Course Goals

- **Deepen understanding of CMOS analog circuit design through a top-down study of a modern analog system**

  The lectures will focus on Delta-Sigma ADCs, but you may do your project on another analog system.

- **Develop circuit insight through brief peeks at some nifty little circuits**

  The circuit world is filled with many little gems that every competent designer ought to know.

Date	Lecture			Ref	Homework
2008-01-07	RS	1	Introduction: MOD1 & MOD2	S&T 2-3, A	Matlab MOD2
2008-01-14	RS	2	Example Design: Part 1	S&T 9.1, J&M 10	Switch-level sim
2008-01-21	RS	3	Example Design: Part 2	J&M 14	Q-level sim
2008-01-28	TC	4	Pipeline and SAR ADCs		Arch. Comp.
2008-02-04	ISSCC– No Lecture				
2008-02-11	RS	5	Advanced ΔΣ	S&T 4, 6.6, 9.4, B	ΔΣ Toolbox; Proj.
2008-02-18	Reading Week– No Lecture				
2008-02-25	RS	6	Comparator & Flash ADC	J&M 7	
2008-03-03	TC	7	SC Circuits	J&M 10	
2008-03-10	TC	8	Amplifier Design		
2008-03-17	TC	9	Amplifier Design		
2008-03-24	TC	10	Noise in SC Circuits	S&T C	
2008-03-31	Project Presentation				
2008-04-07	TC	11	Matching & MM-Shaping		Project Report
2008-04-14	RS	12	Switching Regulator		Q-level sim

Highlights

(i.e. What you will learn today)

1 **High-Order ΔΣ Modulation**

   NTF, SQNR, instability, $\|H\|_\infty$, $\|h\|_1$

2 **Feedback, Feedforward and Generic ΔΣ Topologies**

   NTF selection

3 **ΔΣ Toolbox**

   Demos: NTF synthesis and simulation

   Examples: Lowpass and Bandpass, MASH

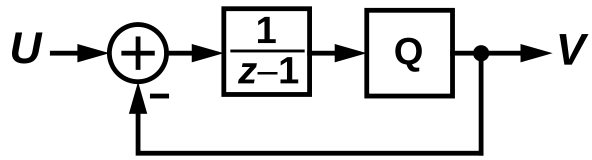
   ABCD: State-space representation

4 **Continuous-Time ΔΣ**

   Inherent anti-aliasing

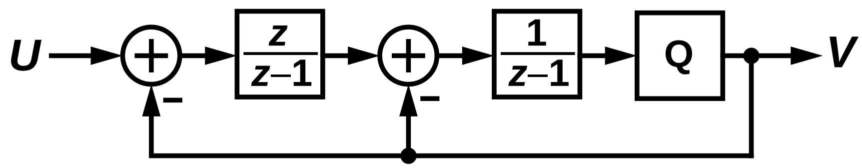
Review: MOD1 & MOD2

MOD1:



$$STF(z) = z^{-1}$$
$$NTF(z) = 1 - z^{-1}$$

MOD2:



$$STF(z) = z^{-1}$$
$$NTF(z) = (1 - z^{-1})^2$$

Review: SQNR vs. OSR

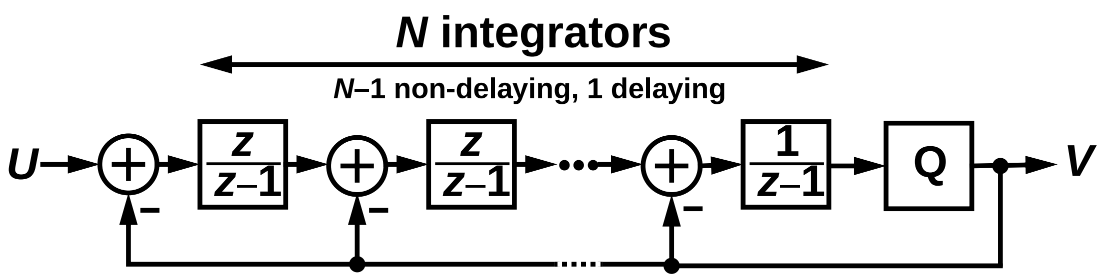
- **The in-band quantization noise power is:**

$$IQNP = \int_0^{\omega_B} |H(e^{j\omega})|^2 S_{ee}(\omega)\, d\omega$$

$$= \begin{cases} \dfrac{\pi^2\sigma_e^2}{3}(OSR)^{-3} & \text{for MOD1} \\ \dfrac{\pi^4\sigma_e^2}{5}(OSR)^{-5} & \text{for MOD2} \end{cases}$$

$\Rightarrow$ **9 dB/octave SQNR-OSR trade-off for MOD1; 15 dB/octave for MOD2**

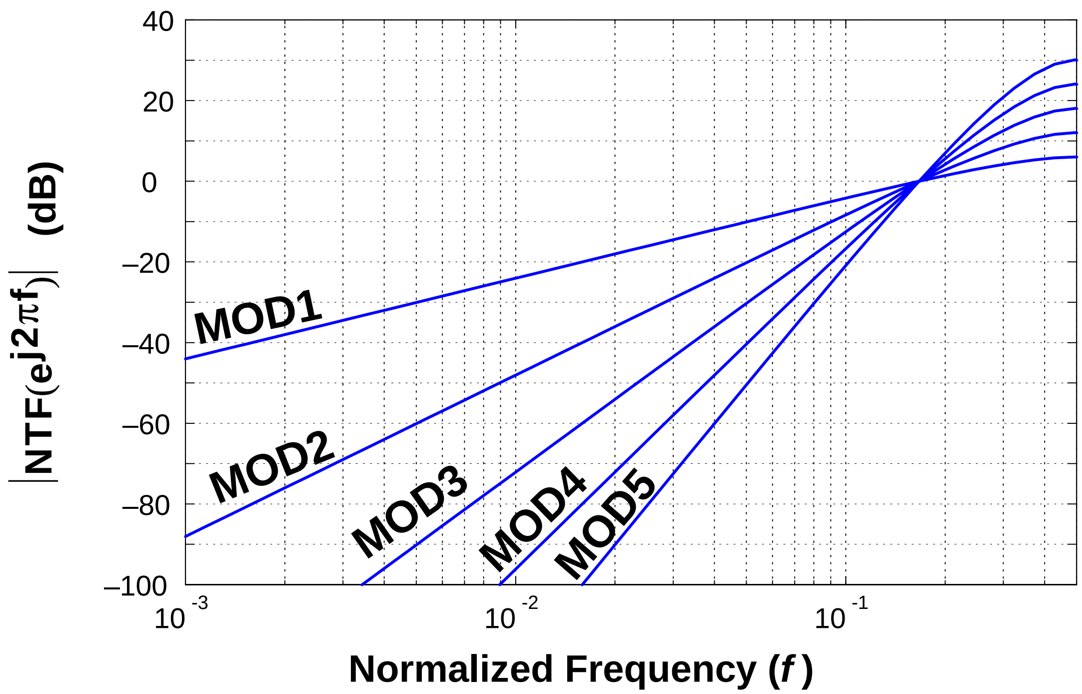
1. MOD*N*



$$STF(z) = z^{-1}$$
$$NTF(z) = (1 - z^{-1})^{N}$$

- **MOD*N*'s NTF is the *N*th power of MOD1's NTF**

NTF Comparison



Predicted Performance

- **In-band quantization noise power**

$$IQNP = \int_0^{0.5/OSR} \left|NTF(e^{j2\pi f})\right|^2 \cdot S_{ee}(f)\, df$$

$$\approx \int_0^{0.5/OSR} (2\pi f)^{2N} \cdot 2\sigma_e^2 df$$

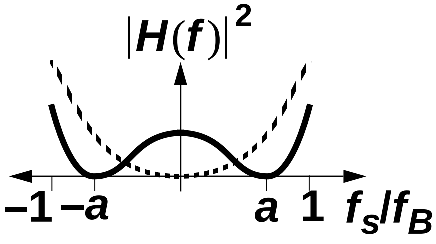
$$= \frac{\pi^{2N}}{(2N+1)(OSR)^{2N+1}}\sigma_e^2$$

- **Quantization noise drops as the $(2N+1)^{th}$ power of OSR— (6*N*+3) dB/octave SQNR-OSR trade-off**

Improving NTF Performance– NTF Zero Optimization

- **Minimize the integral of $|NTF|^2$ over the passband**

  Normalize passband edge to 1 for ease of calculation:



Need to find the a_i which minimize the integral

$$\int_{-1}^{1}(x^2 - a_1^2)^2 dx\,, \qquad n = 2$$

$$\int_{-1}^{1}x^2(x^2 - a_1^2)^2 dx\,, \qquad n = 3$$

$$\int_{-1}^{1}(x^2 - a_1^2)^2(x^2 - a_2^2)^2 dx\,, \qquad n = 4$$

$$\vdots$$

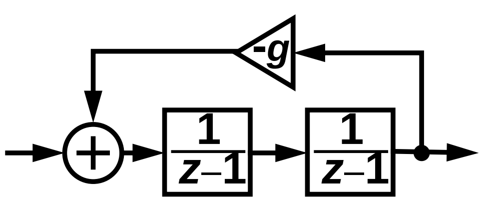
Solutions Up to Order = 8

Order	Optimal Zero Placement Relative to f_B	SQNR Improvement
1	0	0 dB
2	$\pm 1/\sqrt{3}$	3.5 dB
3	$0, \pm\sqrt{3/5}$	8 dB
4	$\pm\sqrt{3/7 \pm \sqrt{(3/7)^2 - 3/35}}$	13 dB
5	$0, \pm\sqrt{5/9 \pm \sqrt{(5/9)^2 - 5/21}}$ [Y. Yang]	18 dB
6	$\pm 0.23862, \pm 0.66121, \pm 0.93247$	23 dB
7	$0, \pm 0.40585, \pm 0.74153, \pm 0.94911$	28 dB
8	$\pm 0.18343, \pm 0.52553, \pm 0.79667, \pm 0.96029$	34 dB

Topological Implication

- **Feedback around pairs integrators:**

2 Delaying Integrators



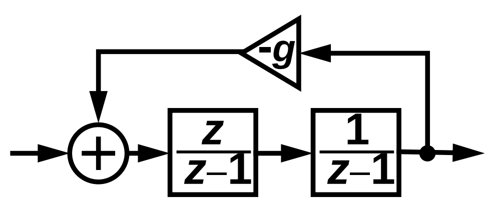
Poles are the roots of

$$1 + g\left(\frac{1}{z-1}\right)^2 = 0$$

i.e. $z = 1 \pm j\sqrt{g}$

Not quite on the unit circle, but fairly close if *g*<<1.

Non-delaying + Delaying Integrators (LDI Loop)



Poles are the roots of

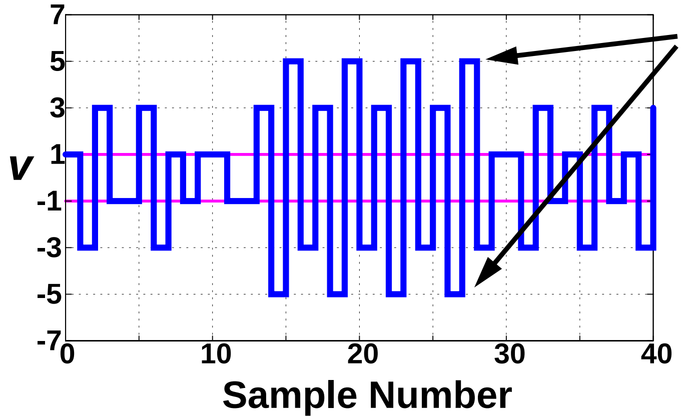
$$1 + \frac{gz}{(z-1)^2} = 0$$

i.e. $z = e^{\pm j\theta}$, $\cos\theta = 1 - g/2$

Precisely on the unit circle, regardless of the value of *g*.

Problem: A High-Order Modulator Wants a Multi-bit Quantizer

E.g. MOD3 with an Infinite Quantizer and Zero Input

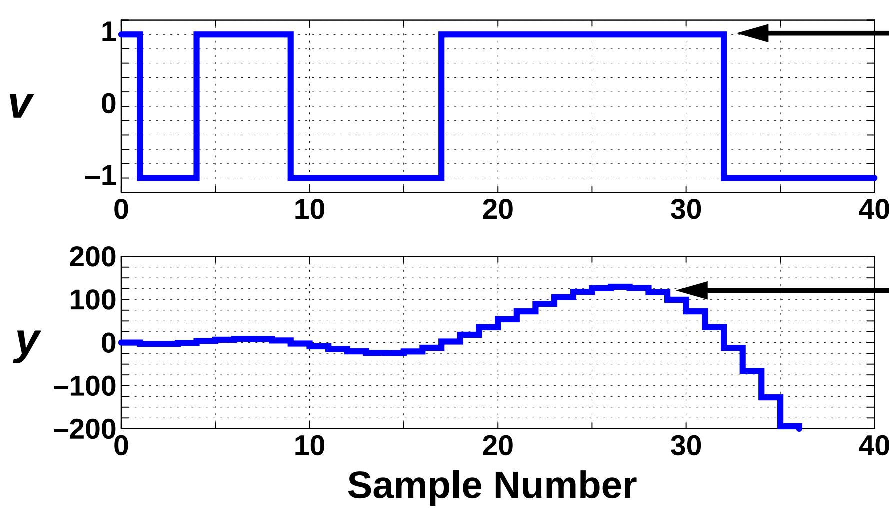


Quantizer input gets large, even if the input is small.

6 quantizer levels are used by a small input.

Simulation of MOD3-1b

(MOD3 with a Binary Quantizer)



Long strings of +1/–1

HUGE!

- **MOD3-1b is unstable, even with zero input**

Solutions to the Stability Problem
Historical Order

1 **Multi-bit quantization**

Initially considered undesirable because we lose the inherent linearity of a 1-bit DAC.

2 **More general NTF (not pure differentiation)**

Lower the NTF gain so that quantization error is amplified less.

Unfortunately, reducing the NTF gain reduces the amount by which quantization noise is attenuated.

3 **Multi-stage (MASH) architecture**

- **Combinations of the above are possible**

Multi-bit Quantization

A modulator with *NTF* = *H* and *STF* = 1 is guaranteed to be stable if $|u| < u_{max}$ at all times, where $u_{max} = nlev + 1 - \|h\|_1$ and $\|h\|_1 = \sum_{i=0}^{\infty} |h(i)|$

- **In MOD*N* $H(z) = (1 - z^{-1})^N$, so $h(n) = \{1, -a_1, a_2, -a_3, \ldots (-1)^N a_N, 0 \ldots\}$, $a_i > 0$ and thus $\|h\|_1 = H(-1) = 2^N$**
- **$nlev = 2^N$ implies $u_{max} = nlev + 1 - \|h\|_1 = 1$**

  MOD*N* is guaranteed to be stable with an *N*-bit quantizer if the input magnitude is less than $\Delta/2 = 1$.

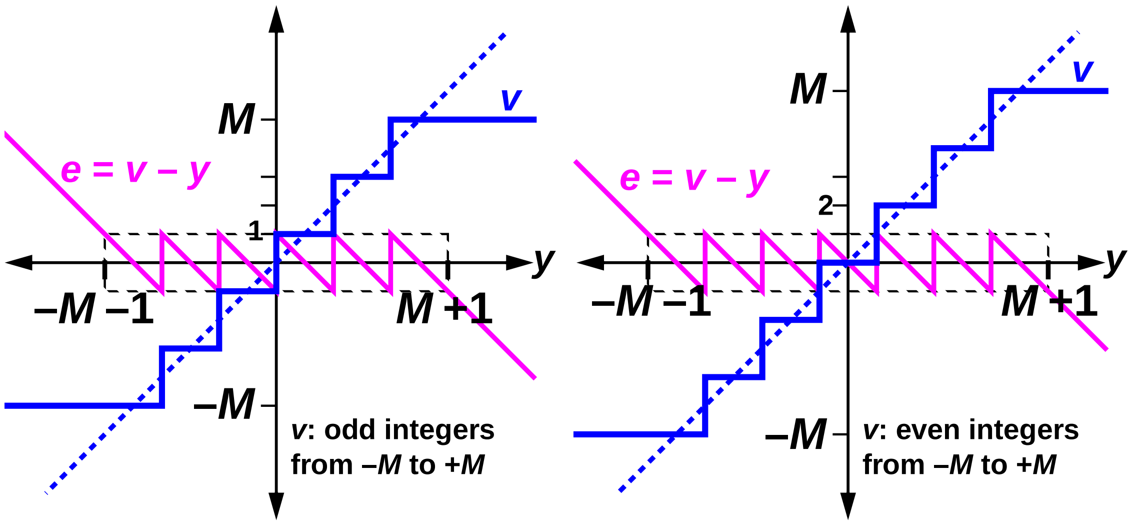
  This result is quite conservative.

- **Similarly, $nlev = 2^{N+1}$ guarantees that MOD*N* is stable for inputs up to 50% of full-scale**

M-Step Symmetric Quantizer

$\Delta = 2$, ($nlev = M + 1$)

M odd: mid-rise **M even: mid-tread**



- **No-overload range:** $|y| \le nlev \Rightarrow |e| \le \Delta/2 = 1$

Inductive Proof of $\|h\|_1$ Criterion

- **Assume STF = 1 and** $(\forall n)(|u(n)| \le u_{max})$
- **Assume** $|e(i)| \le 1$ **for** $i < n$.**[Induction Hypothesis]**

$$
\begin{aligned}
|y(n)| &= \left| u(n) + \textstyle\sum_{i=1}^{\infty} h(i)e(n-i) \right| \\
&\le u_{max} + \textstyle\sum_{i=1}^{\infty} |h(i)||e(n-i)| \\
&\le u_{max} + \textstyle\sum_{i=1}^{\infty} |h(i)| = u_{max} + \|h\|_1 - 1
\end{aligned}
$$

Then $u_{max} = nlev + 1 - \|h\|_1$

$\Rightarrow \quad |y(n)| \le nlev$

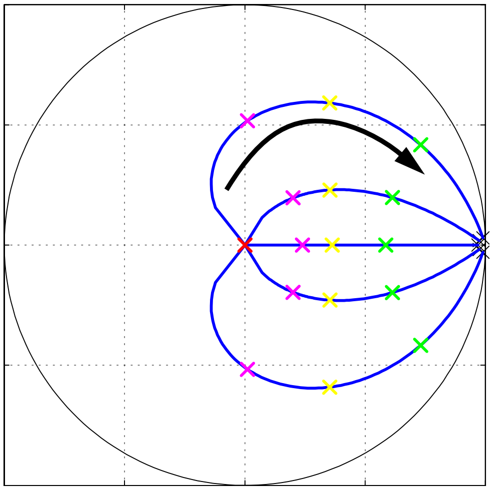
$\Rightarrow \quad |e(n)| \le 1$

- **So by induction** $|e(i)| \le 1$ **for all** $i > 0$

More General NTF

- **Instead of $NTF(z) = A(z)/B(z)$ with $B(z) = z^n$, use a more general $B(z)$**

  Roots of *B* are the poles of the NTF and must be inside the unit circle.



Moving the poles away from $z = 1$ toward $z = 0$ makes the gain of the NTF approach unity.

The Lee Criterion for Stability in a 1-bit Modulator: $\|H\|_\infty \le 2$

[Wai Lee, 1987]

- **The measure of the "gain" of *H* is the maximum magnitude of *H* over frequency, aka the *infinity-norm* of *H*:** $\|H\|_\infty \equiv \max_{\omega \in [0, 2\pi]} (H(e^{j\omega}))$

Q: Is the Lee criterion <u>necessary</u> for stability?

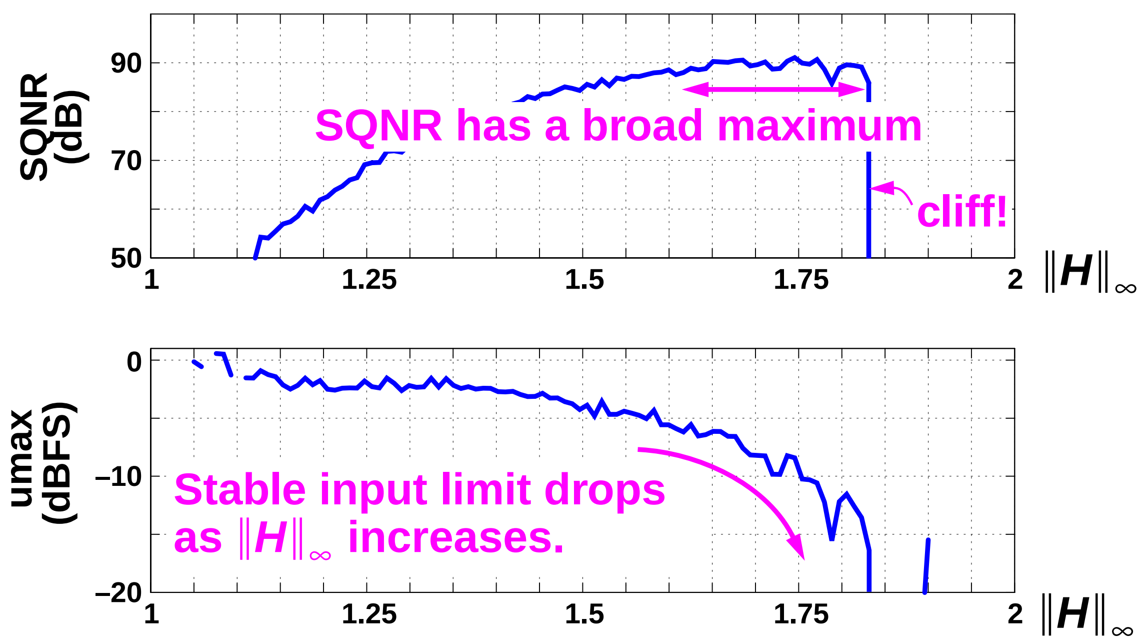
No. MOD2 is stable (for DC inputs less than FS) but for MOD2 $\|H\|_\infty = 4$.

Q: Is the Lee criterion <u>sufficient</u> to ensure stability?

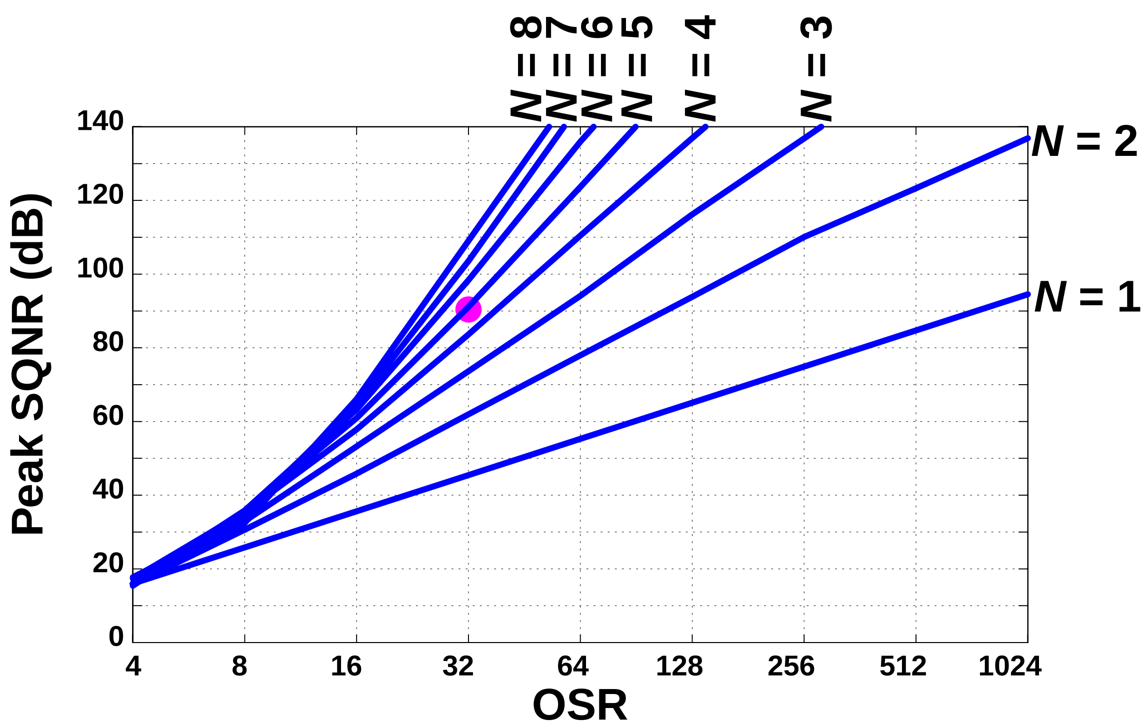
No. There are lots of counter-examples, but $\|H\|_\infty \le 1.5$ often works.

Simulated SQNR vs. $\|H\|_\infty$

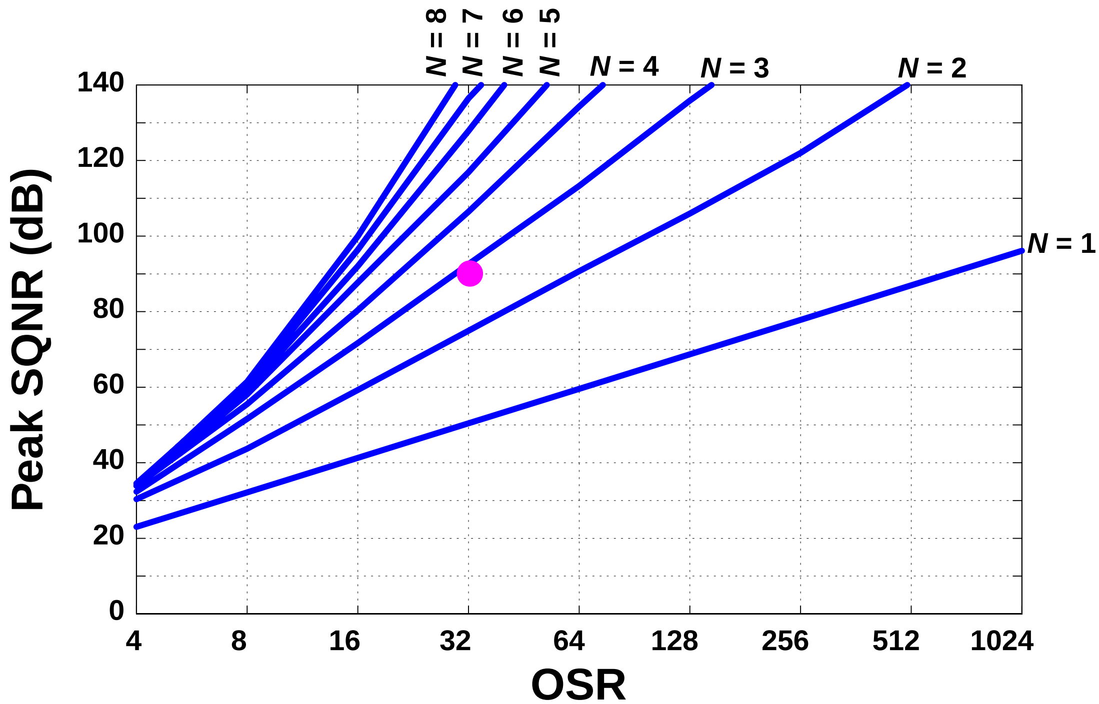
5^{th}-Order NTFs, 1-b Quant., OSR = 32



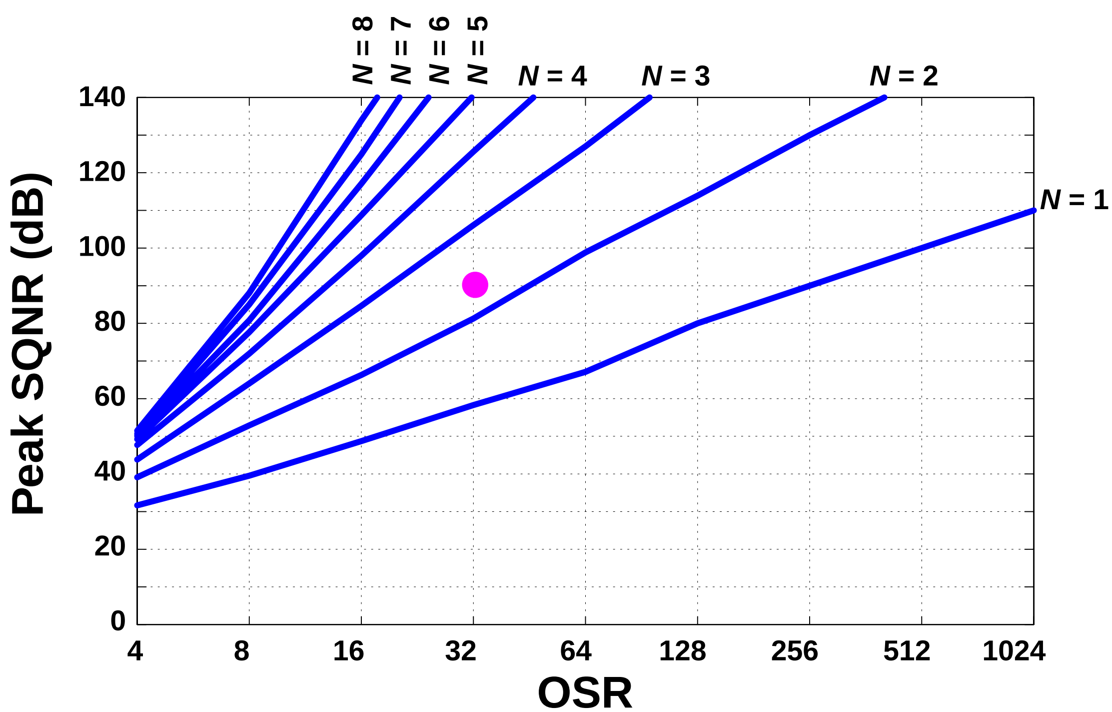
SQNR Limits— 1-bit Modulation



SQNR Limits for 2-bit Modulators

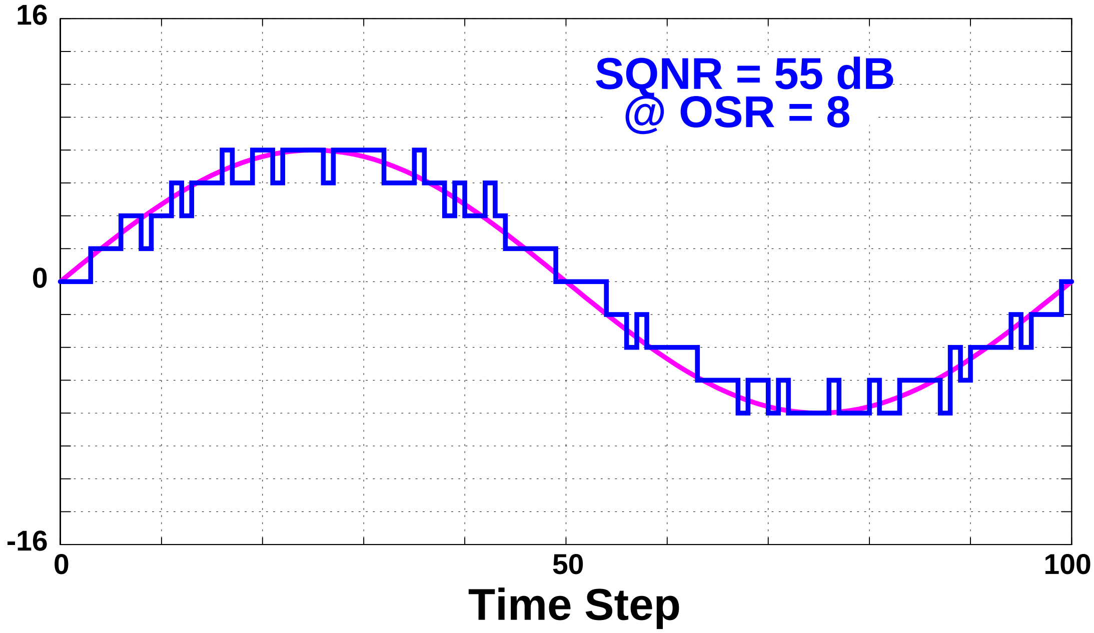


SQNR Limits for 3-bit Modulators



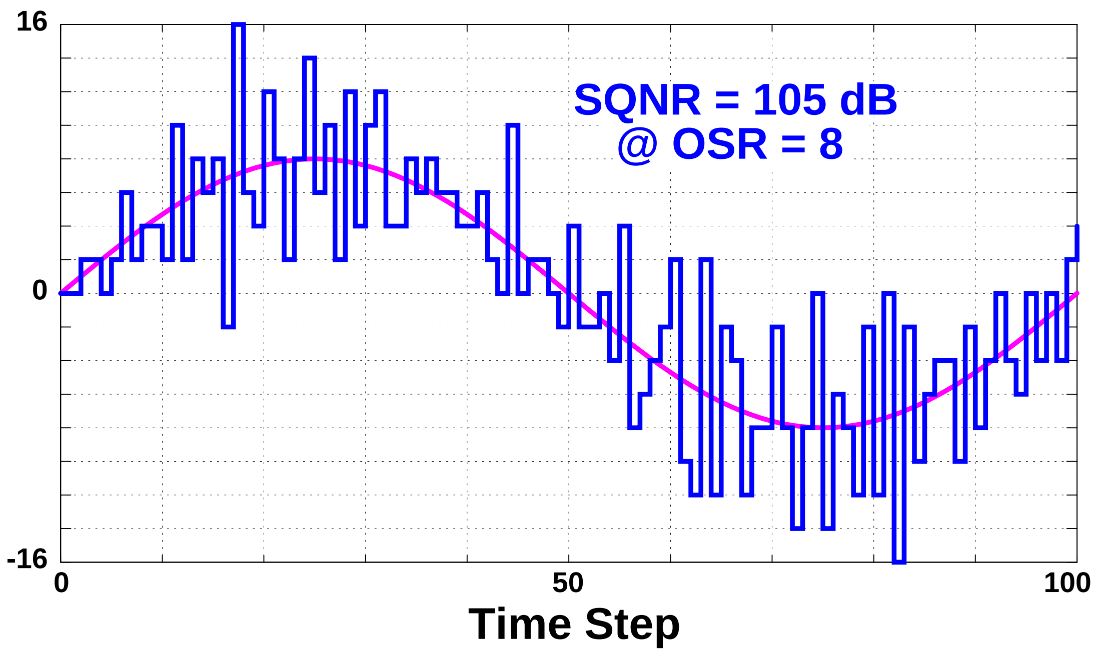
Example Waveforms

7^{th}-order 17-level modulator, $\|H\|_\infty = 2$

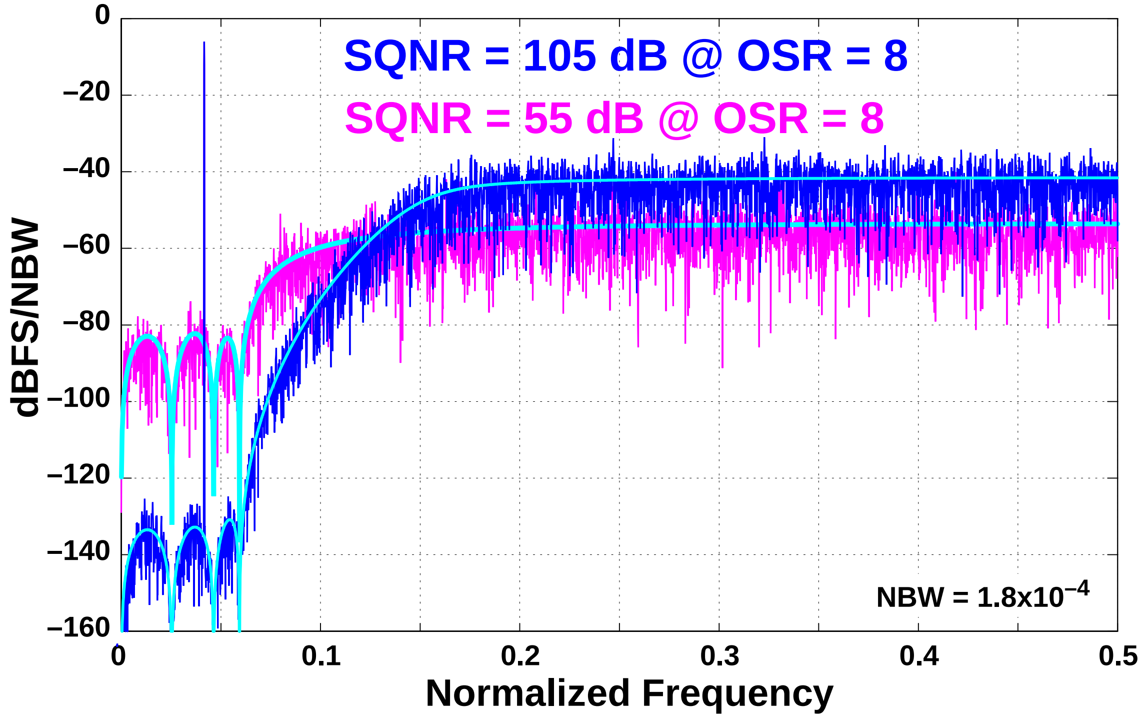


Example Waveforms

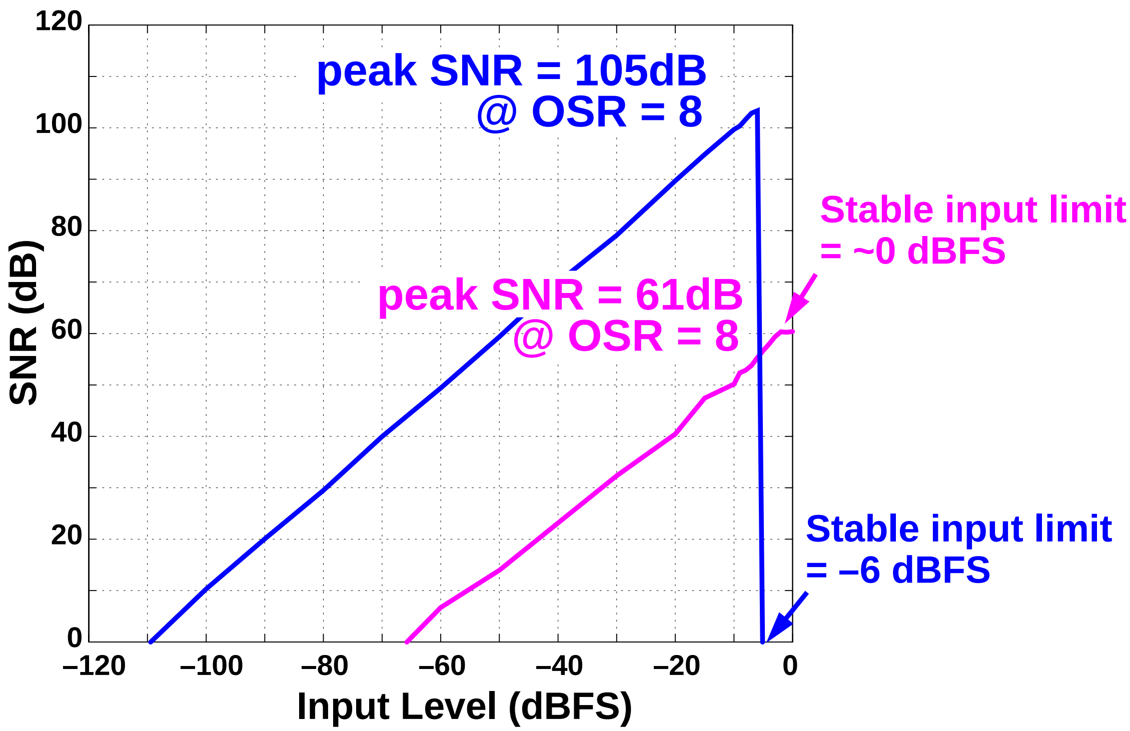
7^{th}-order 17-level modulator, $\|H\|_\infty = 8$



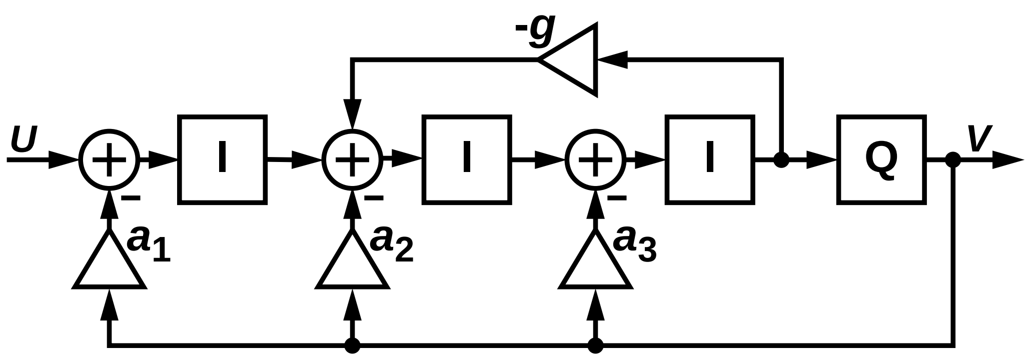
Simulated Spectra



SQNR Curves

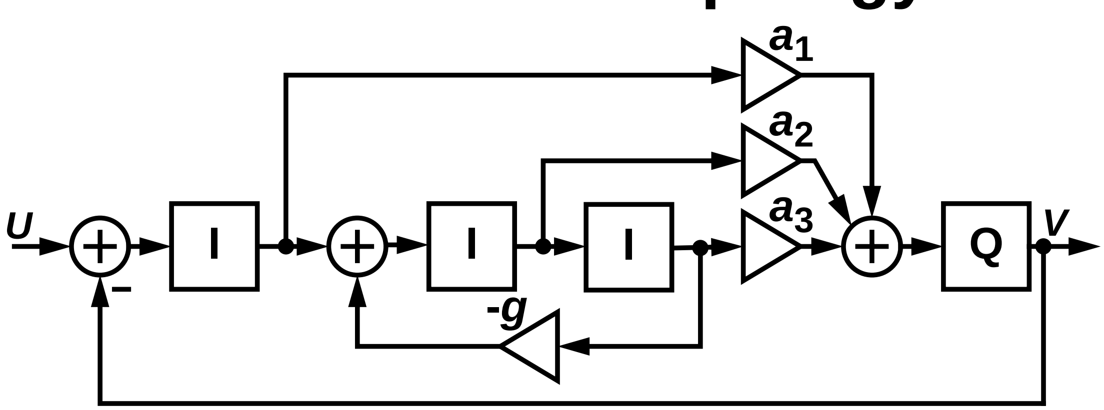


2. Feedback Topology



- *N* integrators precede the quantizer
- Feedback from the quantizer to the input of each integrator (via a DAC)
- Local feedback around pairs of integrators to control NTF zeros
- Multiple input feed-in branches are possible

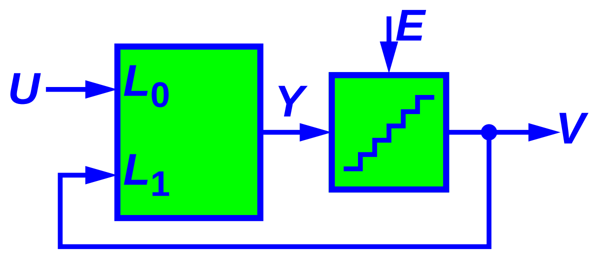
Feedforward Topology



- *N* integrators in a row
- Each integrator output is fed forward to the quantizer
- Local feedback around pairs of integrators to control NTF zeros
- Multiple input feed-in branches also possible

Generic Single-Loop ΔΣ ADC

- **Linear Loop Filter + Nonlinear Quantizer:**



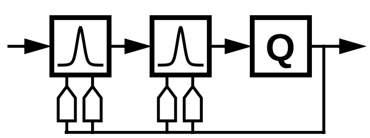
$$\begin{aligned} Y &= L_0 U + L_1 V \\ V &= Y + E \end{aligned} \Rightarrow \boxed{V = STF \cdot U + NTF \cdot E}\text{, where}$$

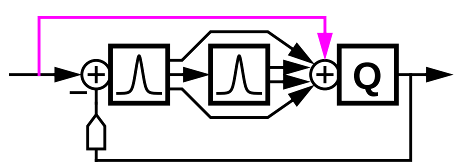
$$NTF = \frac{1}{1 - L_1} \quad \& \quad STF = L_0 \cdot NTF$$

Inverse Relations:

$L_1 = 1 - 1/NTF$, $L_0 = STF\ /\ NTF$

Feedback vs. Feedforward STF





$$L_0(z) = \frac{N_0(z)}{D(z)} \qquad L_1(z) = \frac{N_1(z)}{D(z)}$$

$$NTF(z) = \frac{1}{1 - L_1(z)} = \frac{D(z)}{N_1(z) - D(z)}$$

poles of LF are zeros of NTF

$$STF(z) = \frac{L_0(z)}{1 - L_1(z)} = \frac{N_0(z)}{N_1(z) - D(z)}$$

same poles as NTF

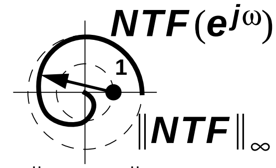
zeros = zeros of L_0

STF often has no zeros, only poles.

$$L_0(z) = -L_1(z) = L(z)$$

$$NTF(z) = \frac{1}{1 - L_1(z)} = \frac{1}{1 + L(z)}$$

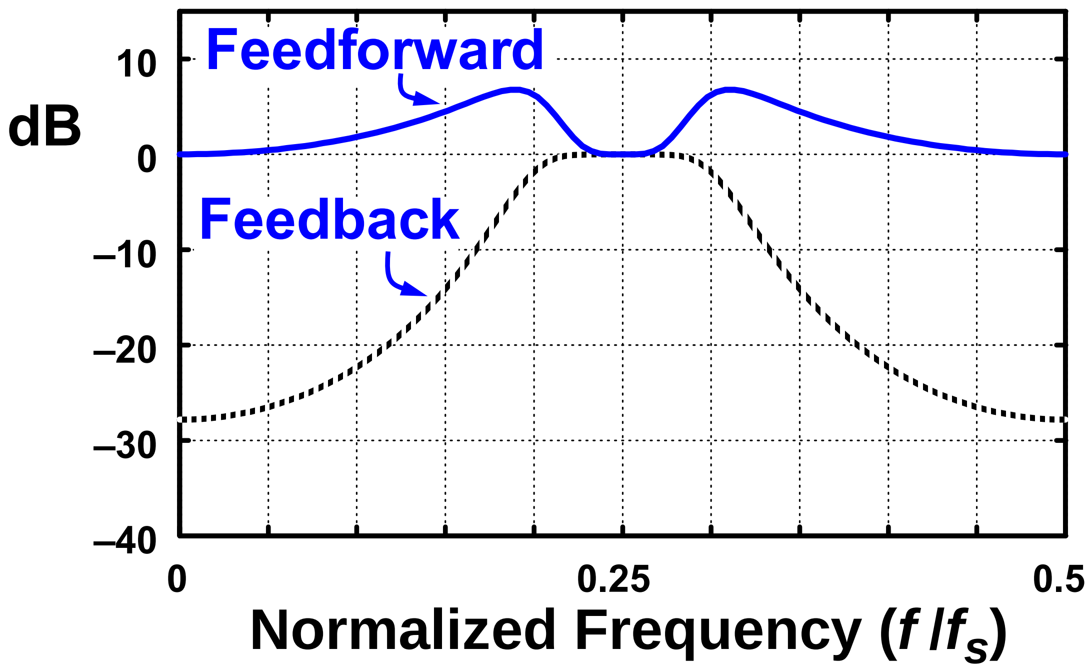
$$STF(z) = \frac{L(z)}{1 + L(z)} = 1 - H(z)$$



$$\|STF\|_\infty \approx \|NTF\|_\infty + 1$$

With extra feed-in to Q, $G(z) = 1$.

STF Comparison



Feedforward vs. Feedback

- **FF has relaxed dynamic range requirements**

  “All stages except the last have attenuated signal components.”

- **FB has better STF and, in CT modulators, a better AAF**

  In a discrete-time modulator, the STF of FF can be made unity by adding a signal feedforward term to the input of the quantizer.

- **FB needs many DACs;**
  FF needs a summation block

  Can do partial summation before the last integrator.

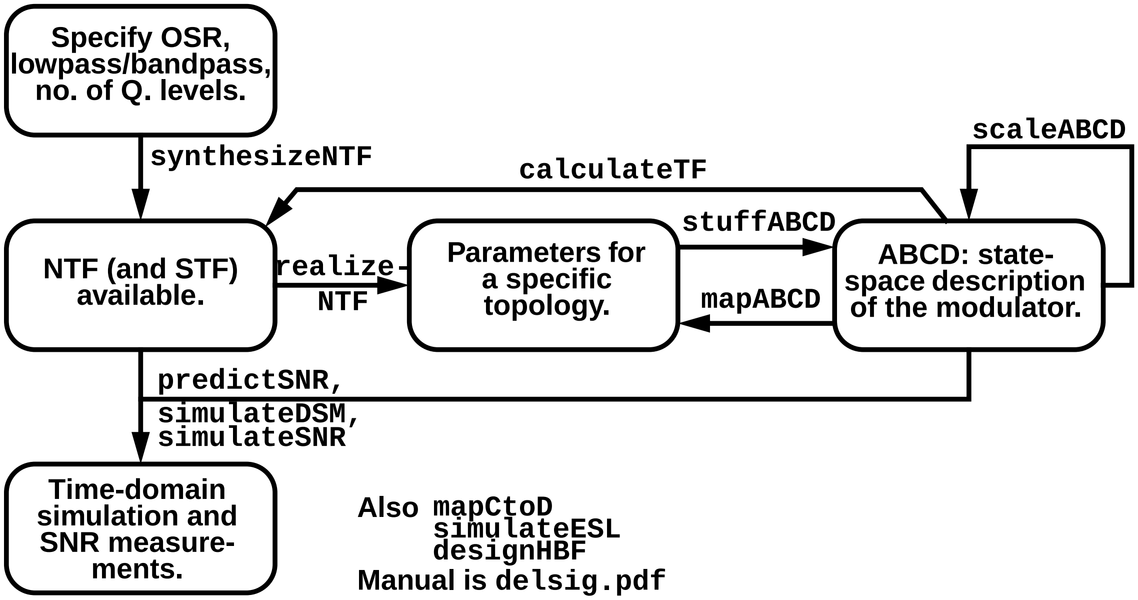
- **FF timing can be tricky**

  Need to quantize *u* and feed it back in zero time.

3. ΔΣ Toolbox

`http://www.mathworks.com/matlabcentral/fileexchange`

Search for “delsig” or “Delta Sigma Toolbox”

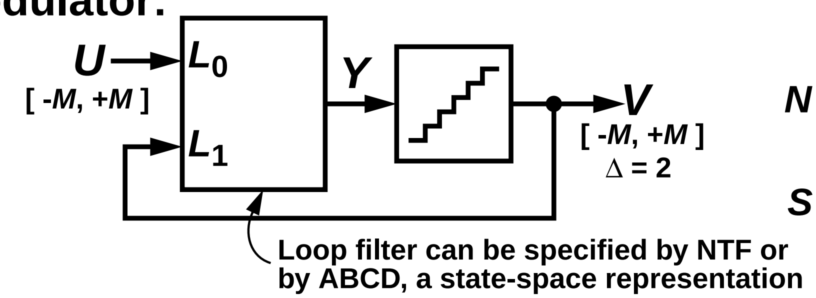


Also `mapCtoD`
`simulateESL`
`designHBF`

Manual is `delsig.pdf`

ΔΣ Toolbox Modulator Model

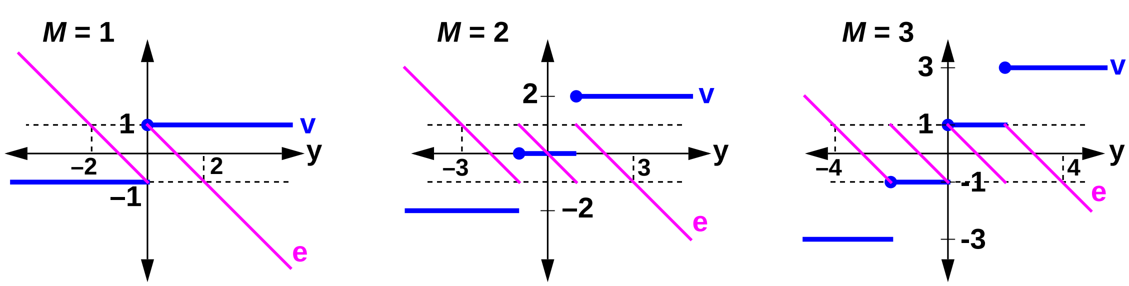
Modulator:



$$NTF = \frac{1}{1 - L_1}$$

$$STF = \frac{L_0}{1 - L_1}$$

Quantizer:



Mid-tread quantizer;
v: even integers [–*M*,+*M*]

Mid-rise quantizer;
v: odd integers [–*M*,+*M*]

NTF Synthesis

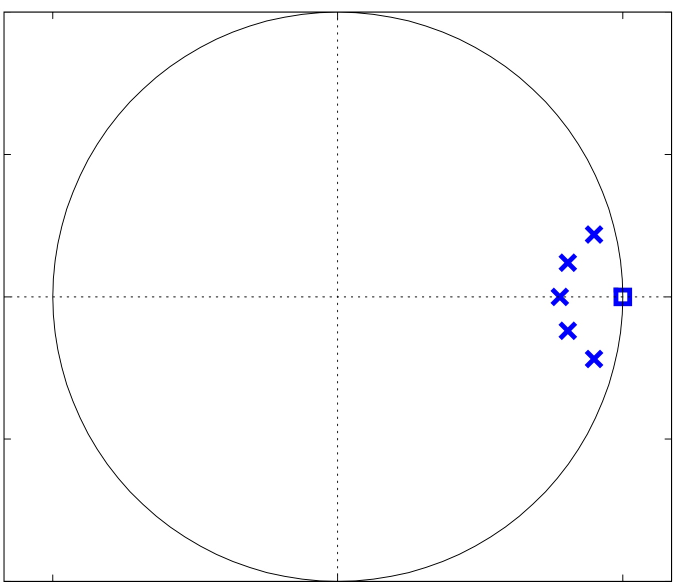
`synthesizeNTF`

- **Not all NTFs are realizable**
  Causality requires $h(0) = 1$, or, in the frequency domain, $H(\infty) = 1$. Recall $H(z) = h(0)z^0 + h(1)z^{-1} + \ldots$
- **Not all NTFs yield stable modulators**
  Rule of thumb for single-bit modulators: $\|H\|_\infty < 1.5$ [Lee].
- **Can optimize NTF zeros to minimize the mean-square value of *H* in the passband**
- **The NTF and STF share poles, and in some modulator topologies the STF zeros are not arbitrary**
  Restrict the NTF such that an all-pole STF is maximally flat. (Almost the same as Butterworth poles.)

Lowpass Example [dsdemo1]

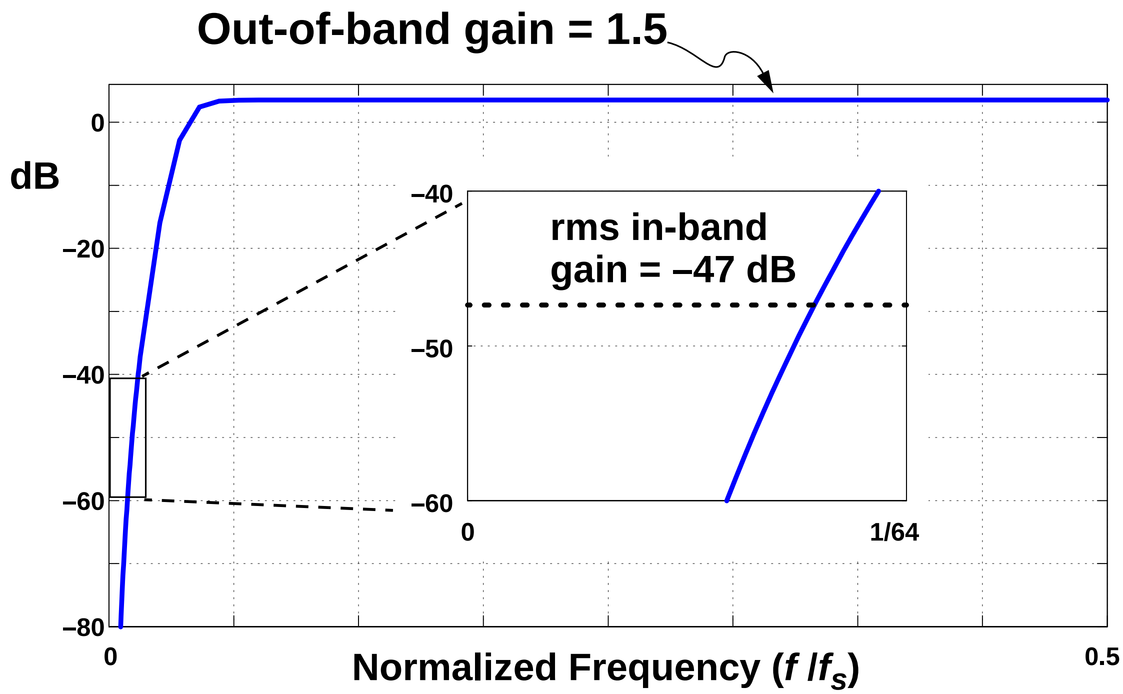
5^{th}-order NTF, all zeros at DC

- **Pole/Zero diagram:**



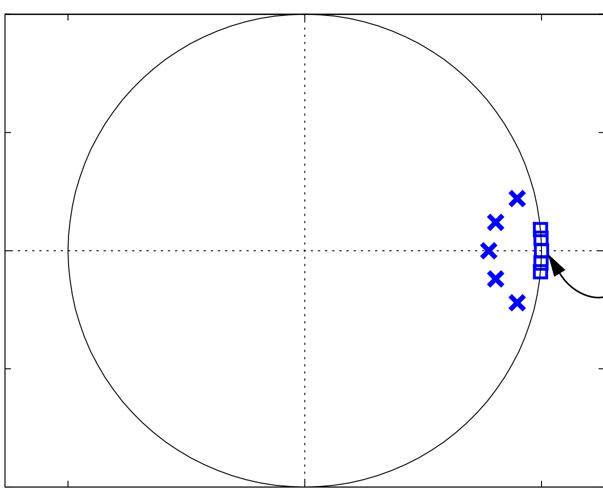
```
OSR = 32;
H = synthesizeNTF(5);
plotPZ(H);

f = linspace(0,0.5);
z = exp(2i*pi*f);
H_z = evalTF(H,z);
plot(f,dbv(H_z));
g = rmsGain(H,0,0.5/OSR)
```

Lowpass NTF



Improved 5th-Order Lowpass NTF

Zeros optimized for OSR=32

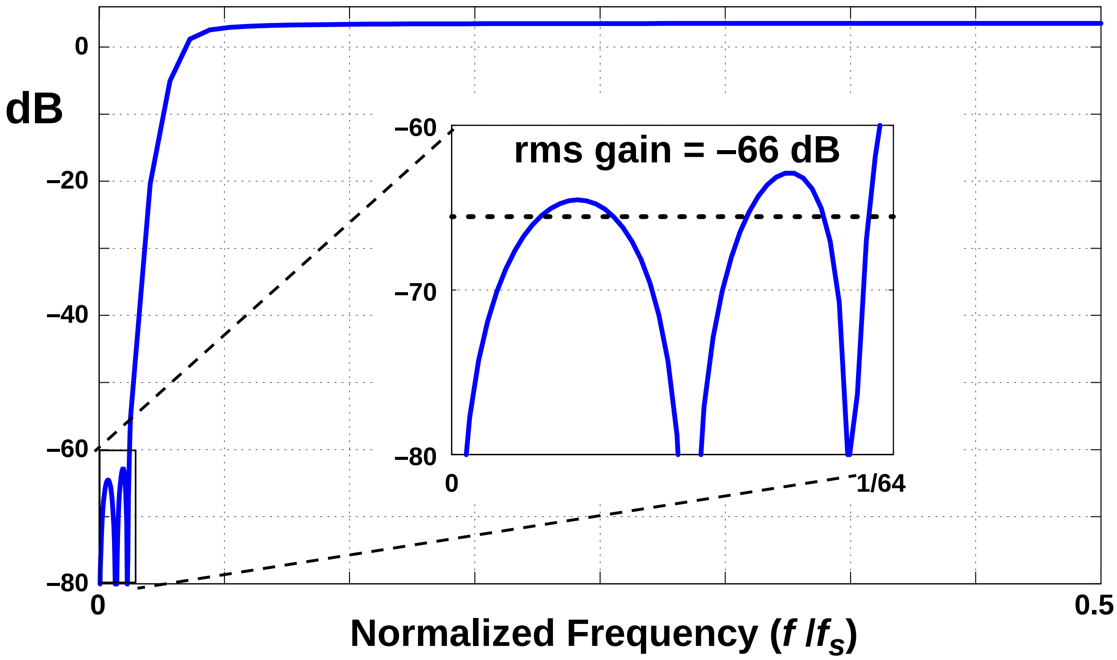


optimization flag

```
OSR = 32;
H = synthesizeNTF(5,OSR,1);
...
```

Zeros spread across the band-of-interest to minimize the rms value of the NTF.

Improved NTF

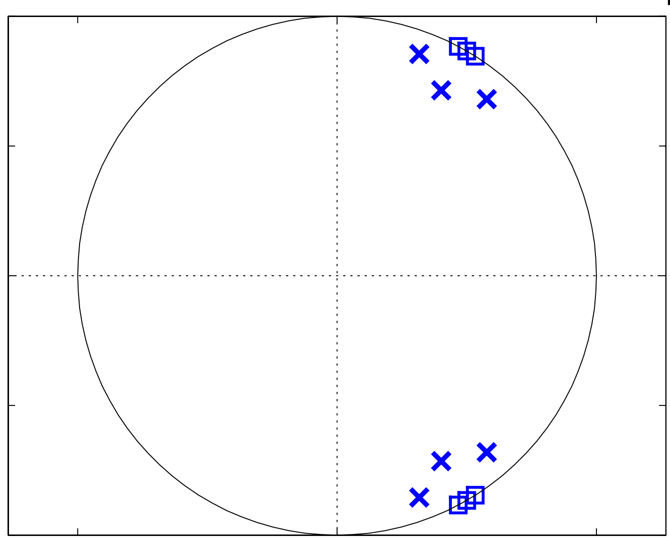


Bandpass Example

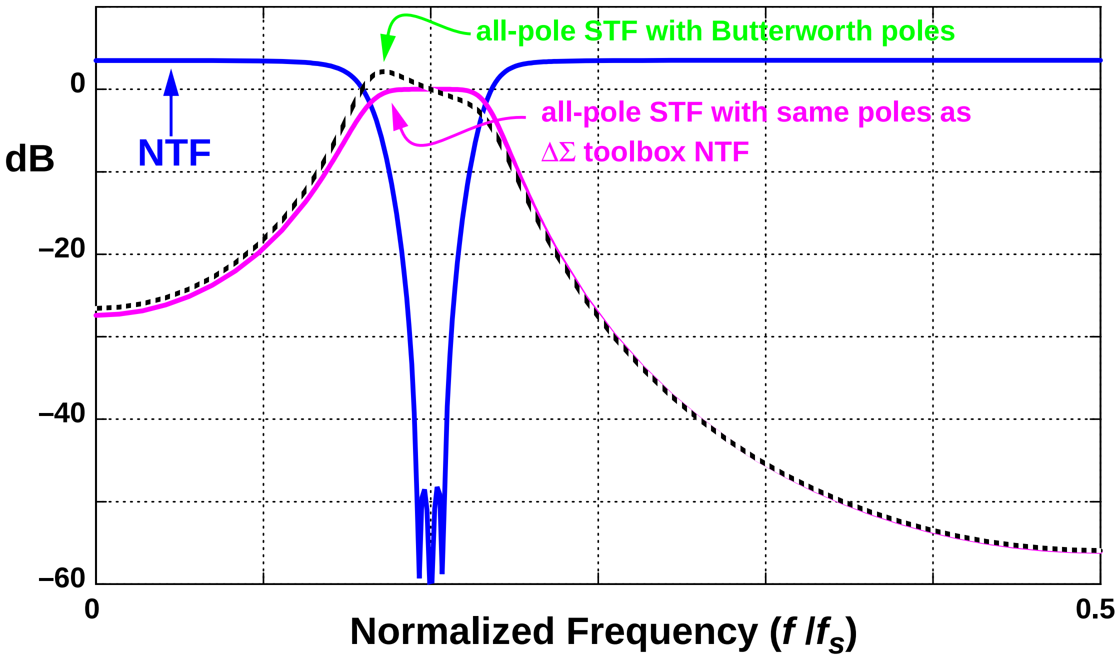
```
OSR = 64;
f0 = 1/6;
H=synthesizeNTF(6,OSR,1,[],f0);...
```

center frequency

[] or NaN means use default value, i.e. Hinf = 1.5



Bandpass NTF and STF



Summary: NTF Selection

- **If OSR is high, consider using a single-bit modulator**

- **To improve SQNR,**

  Optimize zeros,

  Increase $\|H\|_\infty$, or

  Increase order.

- **If SQNR is insufficient, must use a multi-bit design**

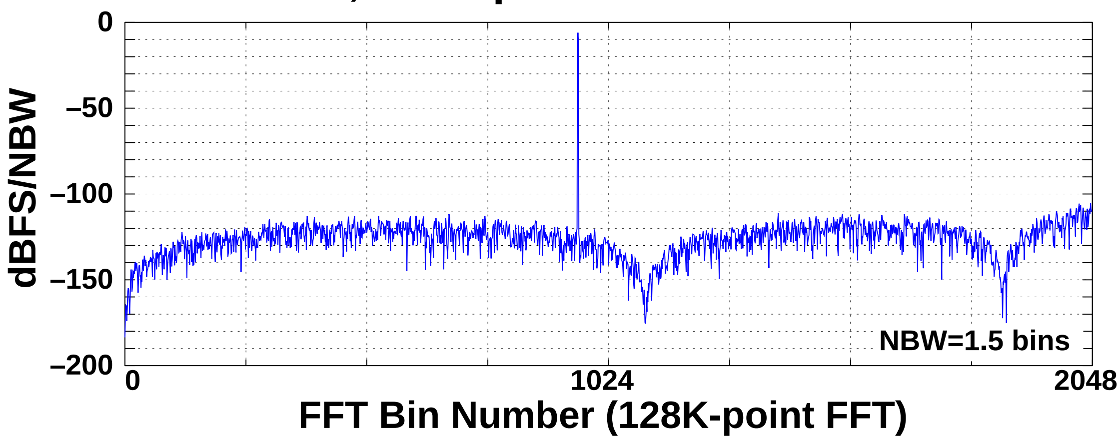
  Can turn all the above knobs to enhance performance.

- **Feedback DAC assumed to be ideal**

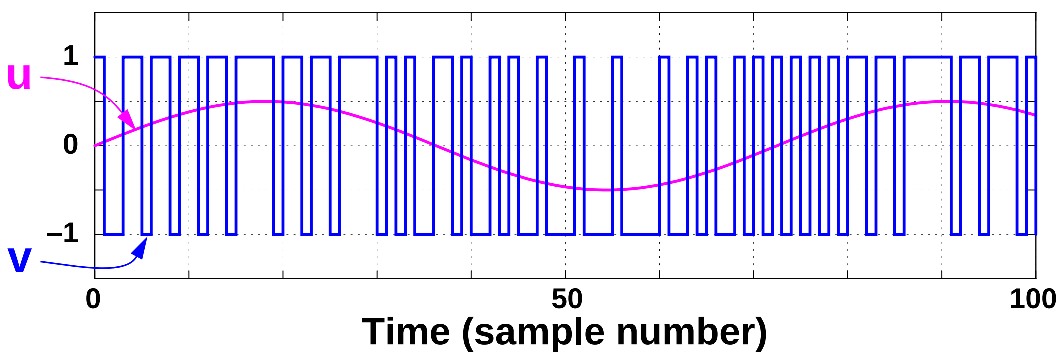
NTF-Based Simulation [dsdemo2]

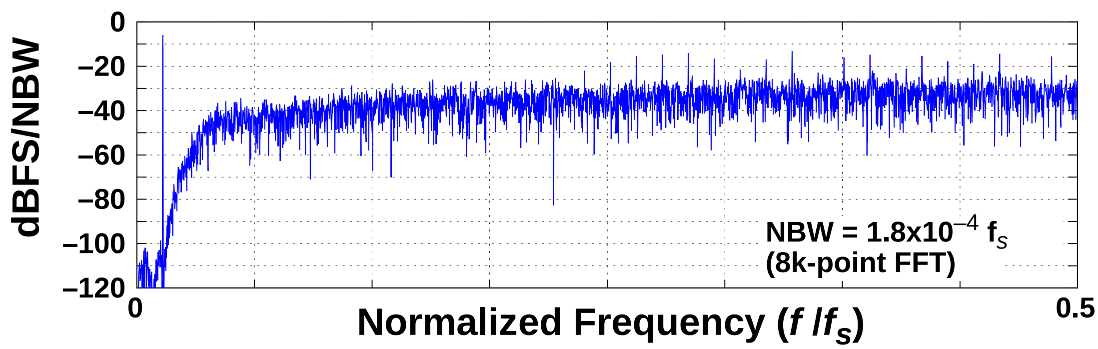
```
order=5; OSR=32;
ntf = synthesizeNTF(order,OSR,1);
N=2^17; fbin=959; A=0.5; % 128K points
input = A*sin(2*pi*fbin/N*[0:N-1]);
output = simulateDSM(input,ntf);
spec = fft(output.*ds_hann(N)/(N/4));
plot(dbv(spec(1:N/(2*OSR))));
```

- In mex form; 128K points in < 0.1 sec



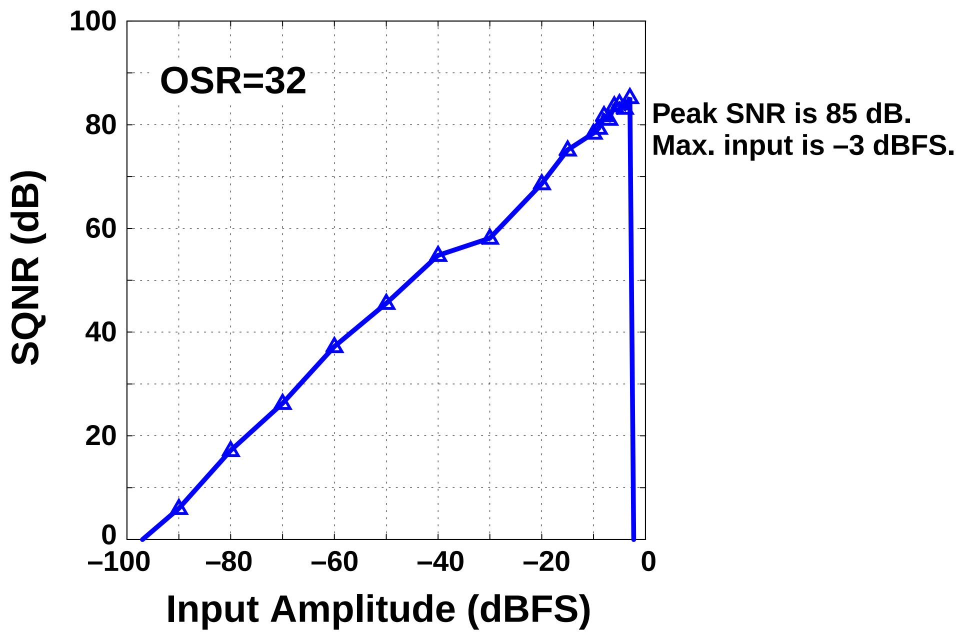
Simulation Example Cont'd





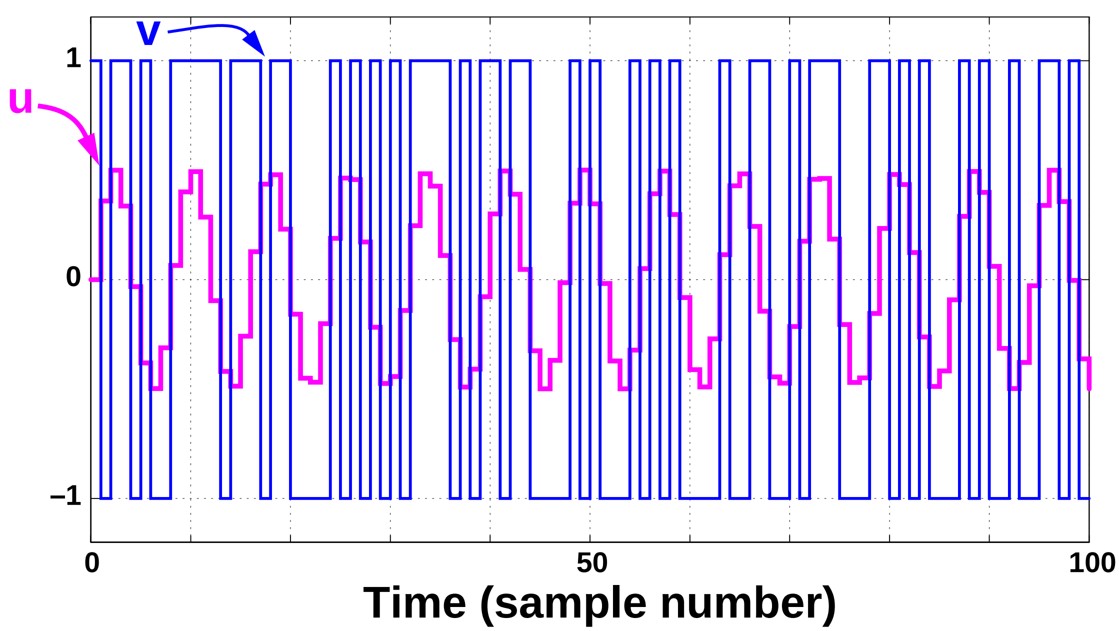
SNR vs. Amplitude: `simulateSNR`

```
[snr amp] = simulateSNR(ntf,OSR);
plot(amp,snr,'b-^');
```

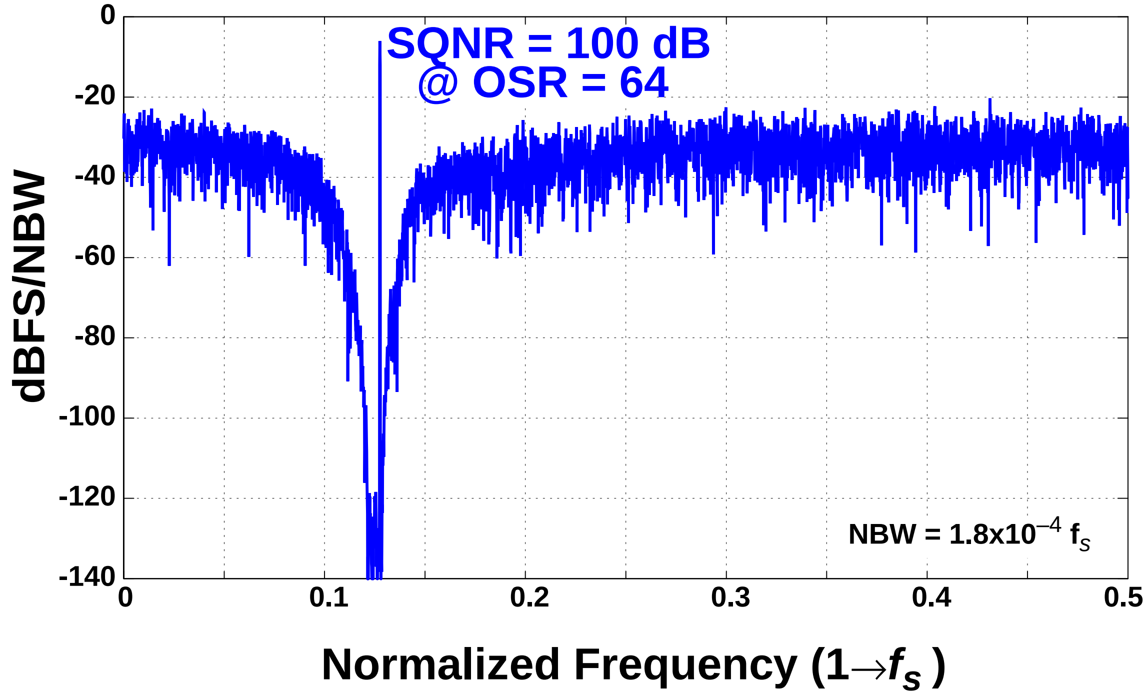


Bandpass Simulation Example

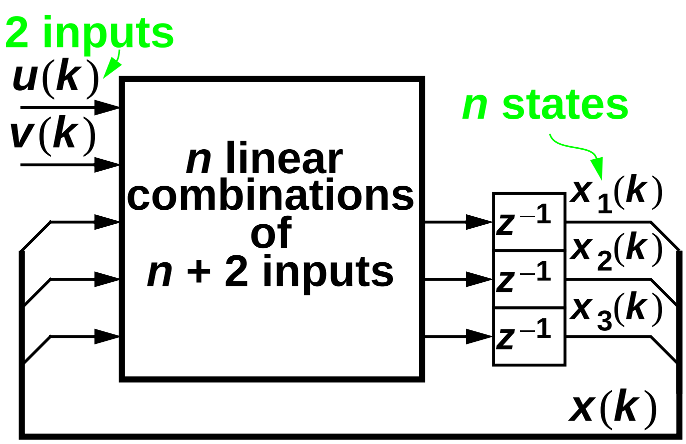
8^{th}-order, $f_0 = f_s/8$

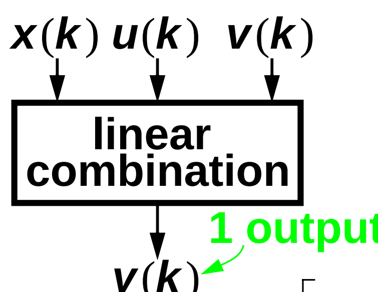


Example Bandpass Spectrum



ABCD: A *State-Space* Representation of the Loop Filter



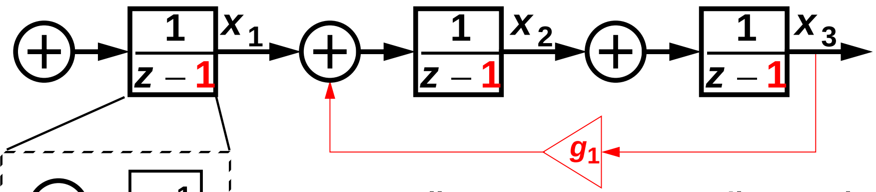


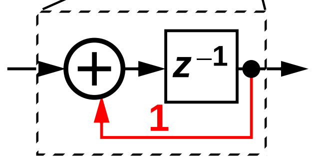
$$y(k) = \underbrace{C}_{1\times n}x(k) + \underbrace{D}_{1\times 2}\begin{bmatrix} u(k) \\ v(k) \end{bmatrix}$$

$$x(k+1) = \underbrace{A}_{n\times n}x(k) + \underbrace{B}_{n\times 2}\begin{bmatrix} u(k) \\ v(k) \end{bmatrix}$$

$$ABCD = \begin{array}{c|cc} & n & 2 \\ \hline n & A & B \\ 1 & C & D \end{array}$$

Ex.: Cascade of Integrators Feedback (CIFB) Topology

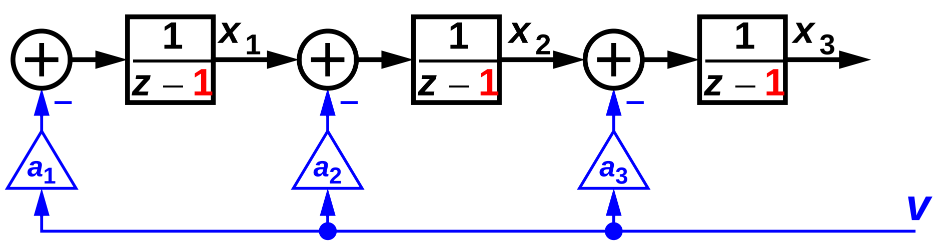




Recall: NTF zeros = Loop Filter's poles.
Loop Filter poles determined by A matrix.

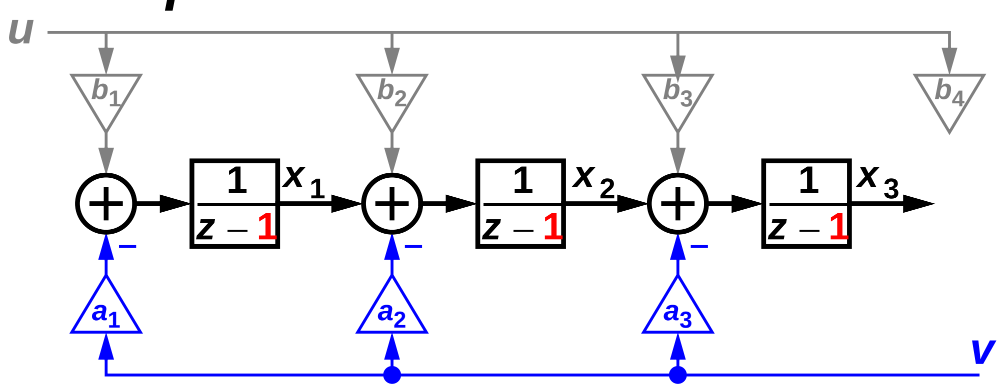
$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \\ y(k) \end{bmatrix} = \left[\begin{array}{ccc|cc} 1 & 0 & 0 & - & - \\ 1 & 1 & g_1 & - & - \\ 0 & 1 & 1 & - & - \\ \hline 0 & 0 & 1 & - & - \end{array}\right] \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ u(k) \\ v(k) \end{bmatrix}$$

CIFB cont'd: a_i Control NTF & STF Poles



$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \\ y(k) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & - & -a_1 \\ 1 & 1 & 0 & - & -a_2 \\ 0 & 1 & 1 & - & -a_3 \\ 0 & 0 & 1 & - & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ u(k) \\ v(k) \end{bmatrix}$$

b_i Control STF Zeros



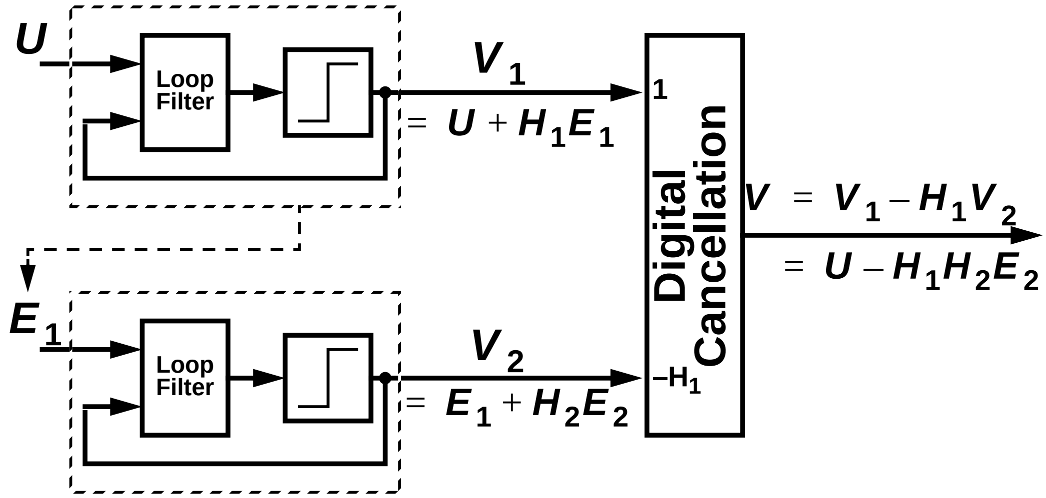
$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \\ y(k) \end{bmatrix} = \left[\begin{array}{ccc|cc} 1 & 0 & 0 & b_1 & -a_1 \\ 1 & 1 & 0 & b_2 & -a_2 \\ 0 & 1 & 1 & b_3 & -a_3 \\ \hline 0 & 0 & 1 & b_4 & 0 \end{array}\right] \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ u(k) \\ v(k) \end{bmatrix}$$

ABCD and the Toolbox

- `simulateDSM` simulates a modulator given an ABCD description of its loop filter
- `realizeNTF` gives (unscaled) coefficients for any of the supported topologies
- `stuffABCD` produces an ABCD matrix given the coefficients for one of the supported topologies
  - `mapABCD` performs the inverse operation.
- `scaleABCD` does dynamic range scaling on any ABCD matrix
- `calculateTF` calculates the NTF and STF from ABCD
  - Useful for checking implementation of new topologies.

Cascade (MASH) Modulators

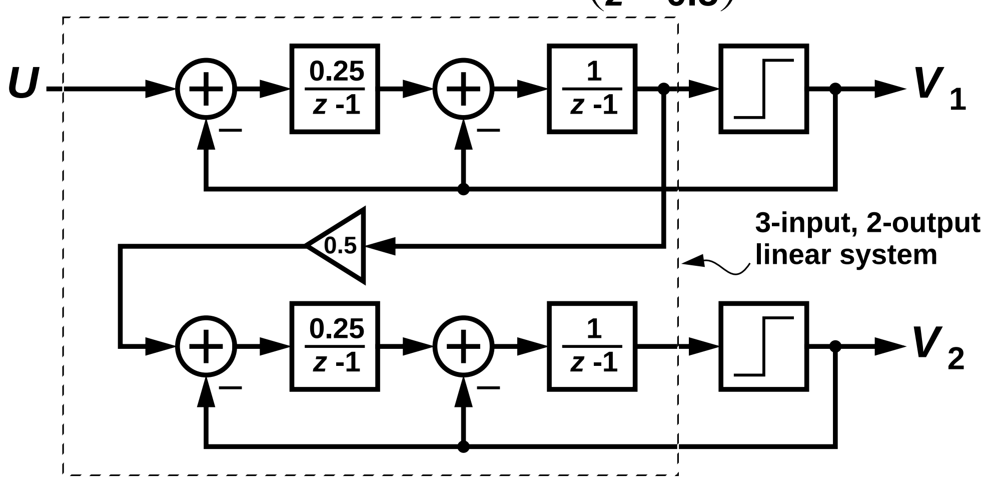
- **Put two (or more) modulators in "series"**



- **The resulting NTF is the *product* of the individual NTFs**

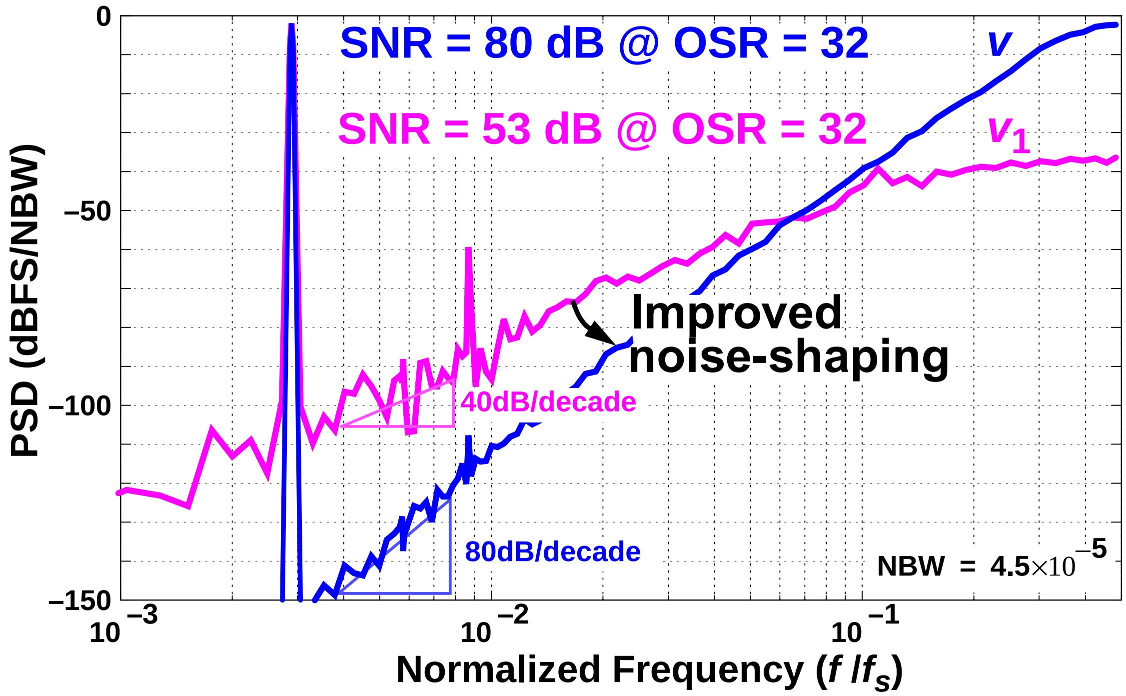
Example: 2-2 Cascade

- **Use Two MOD2b:** $H(z) = \left(\frac{z-1}{z-0.5}\right)^2$

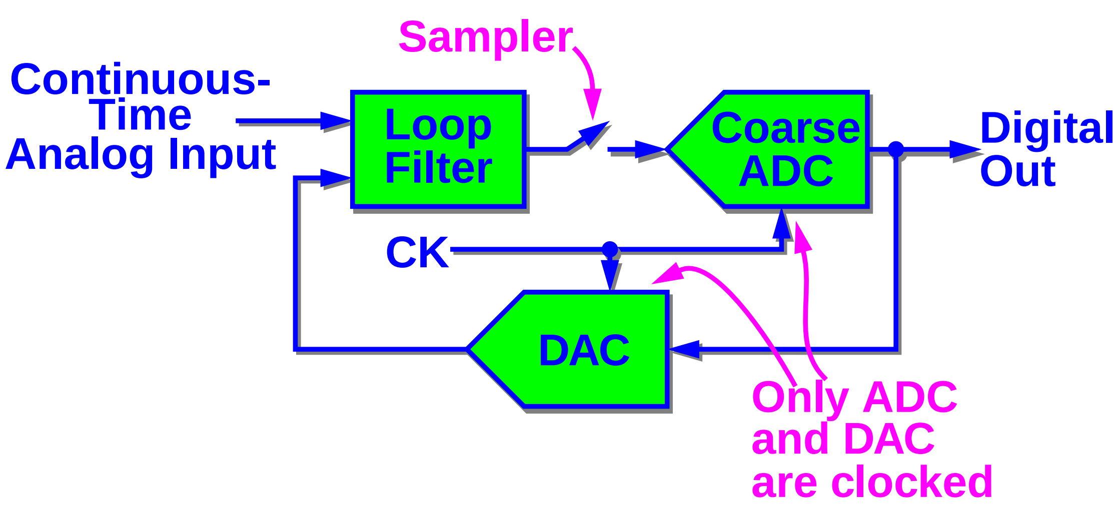


$$V = \frac{1}{z^3}V_1 + \frac{8(z-1)^2(z-0.5)^2}{z^3(z-0.75)}V_2\Bigg) \Rightarrow \left(H(z) = \frac{8(z-1)^4}{(z-0.75)}\right)$$

Example MASH Spectra



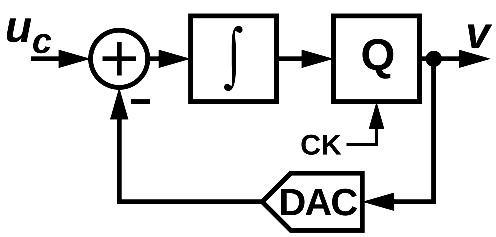
4. A Continuous-Time ΔΣ ADC



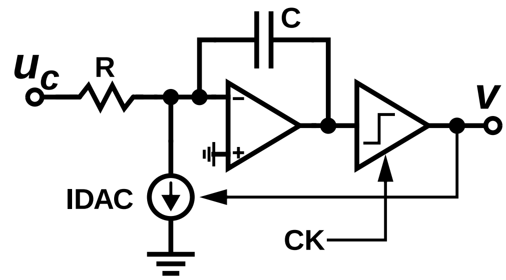
- Loop filter implemented with continuous-time circuitry
- Sampling occurs after the loop filter

Example: MOD1-CT

Block Diagram



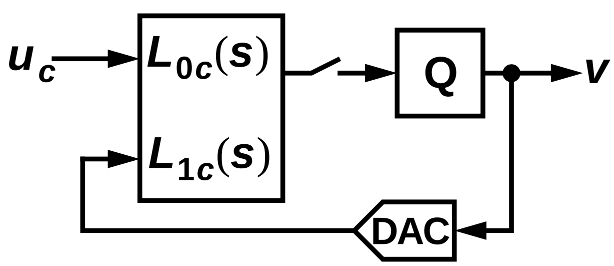
Schematic



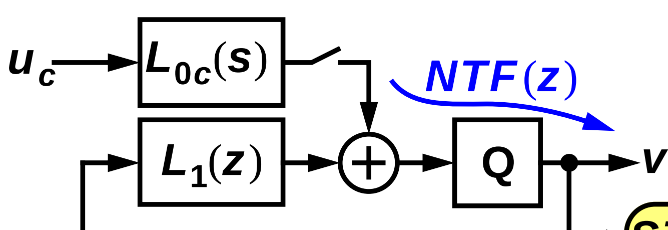
- **Note: Input is a simple resistor, *not* a switched capacitor**
  - **CT ADCs are easier to drive than DT ADCs**

Inherent Anti-Aliasing

- **ΔΣ ADC with CT Loop Filter**



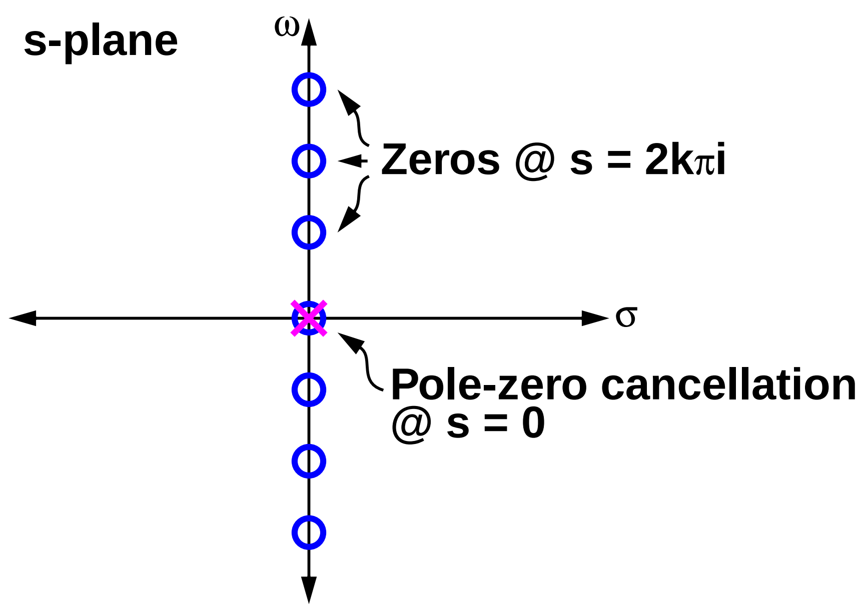
- **Equivalent system with DT feedback path**



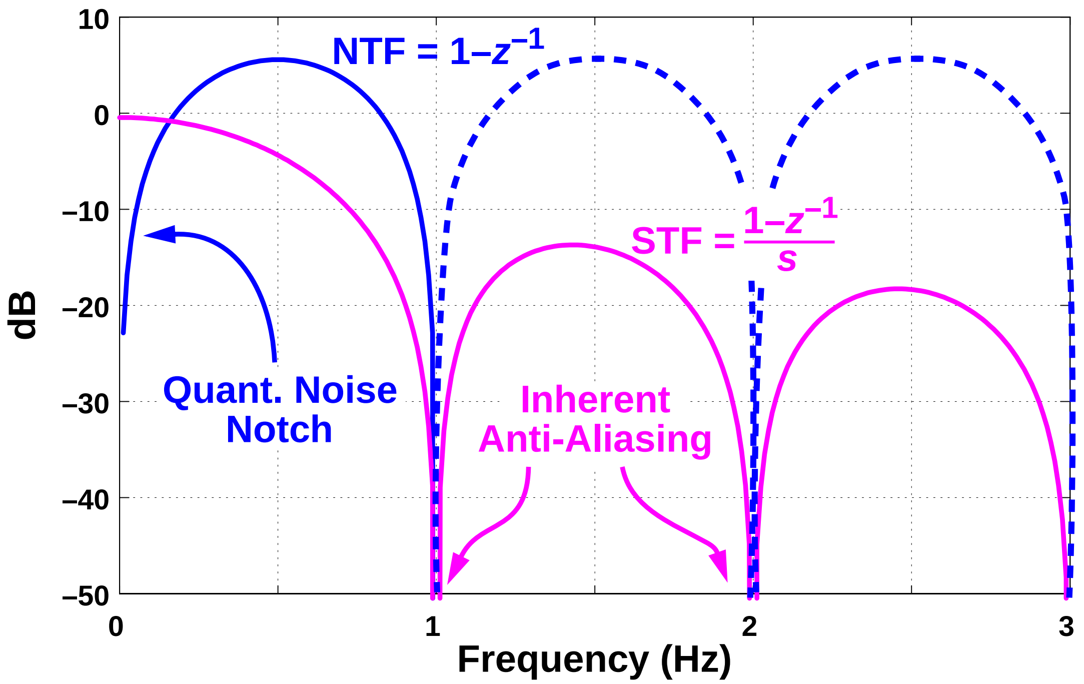
$$\Rightarrow \boxed{STF = L_{0c}(s) NTF(z)}$$

Example: MOD1-CT $STF = \frac{1 - z^{-1}}{s}$

Recall $z = e^s$



MOD1-CT Frequency Responses



Inherent AAF Summary

$$STF = L_{0c}(s) NTF(z)$$

- **STF contains the zeros of the NTF**
- **Any frequency which aliases to the passband is attenuated by at least as much as the quantization noise**

  Anti-alias performance tracks modulator order.
  Also true for MASH systems.
- **The effective anti-alias filter is**

$$AAF(f) = \frac{STF(f)}{STF(f_{alias})} = \frac{L_{0c}(f)}{L_{0c}(f_{alias})}$$

What You Learned Today

And what the homework should solidify

1. **High-Order ΔΣ Modulation**

   NTF, SQNR, instability, $\|H\|_\infty$, $\|h\|_1$
2. **Feedback, Feedforward and Generic ΔΣ Topologies**

   NTF selection
3. **ΔΣ Toolbox**

   Demos: NTF synthesis and simulation

   Examples: Lowpass and Bandpass, MASH

   ABCD: State-space representation
4. **Continuous-Time ΔΣ**

   Inherent anti-aliasing

Homework #5

1. **Verify that with ABCD set to** $ABCD = \begin{bmatrix} 1 & 0 & 1 & -1 \\ 1 & 1 & 1 & -2 \\ 0 & 1 & 0 & 0 \end{bmatrix}$ **`simulateDSM` is the same as your `simulateMOD2` from Homework #1.**

   One simulation is sufficient, just be sure to compare the internal states as well as the bit-stream output.

2. **Obtain an NTF for MOD2b using `synthesizeNTF` with** $\|H\|_\infty = 2.5$**, OSR = 128 and `opt=1`.**

   Plot the NTF's poles and zeros as well as its frequency response. Include those of MOD2.

   Compare MOD2b's SQNR-vs.-input-amplitude curve with MOD2's using `simulateSNR`.

3. **Construct the block diagram of a 4th-order single-loop 9-level ΔΣ ADC employing the CRFB topology that achieves SQNR > 100 dB at OSR = 32 and verify it.**

   Choose $\|H\|_\infty$ **such that the stable input range is maximized.**

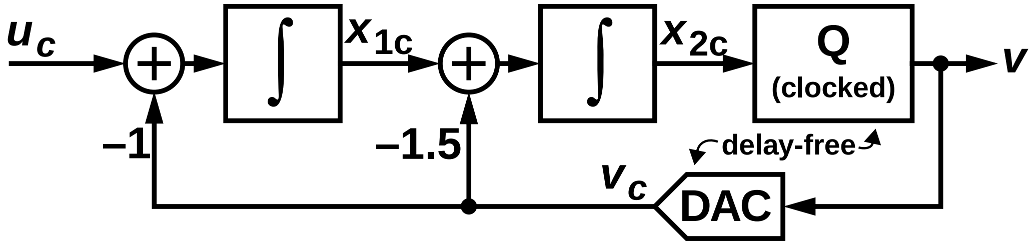
   Plot the SQNR-versus-amplitude curve and note the value of u_{max}**. Compare with the** $\|h\|_1$ **criterion.**

   Realize your NTF with a CRFB-style modulator having a single input feed-in. Ensure STF(dc) = 1.

   Do dynamic range scaling for $u_{max} = -1$ **dBFS and state maxima less than unity. You may use the ΔΣ toolbox function `scaleABCD`. Plot the state maxima as functions of the input amplitude.**

   Implement your block diagram with C/Matlab code and verify that your coded modulator behaves the same as your MATLAB modulator.

4 Verify that the following is a continuous-time implementation of MOD2



It is sufficient to verify that the sampled pulse response of the loop filter is the same as MOD2's.

Show that the STF is $\left(\frac{1-z^{-1}}{s}\right)^2$

If this modulator is used at an oversampling ratio of 64, what is its minimum alias attenuation?