Lecture 1
Introduction to Delta-Sigma ADCs

Trevor Caldwell
trevor.caldwell@awaveip.com

Course Goals

- **Deepen understanding of CMOS analog circuit design through a top-down study of a modern analog system – a delta-sigma ADC**
- **Develop circuit insight through brief peeks at some nifty little circuits of the day**
  - **The circuit world is filled with many little gems that every competent designer ought to know**

Logistics

- **Format:**
  - Meet Tuesdays 10:00-12:00 (no class Feb 18)
  - Twelve 2-hr Lectures + Exam + Project Presentations
- **Grading:**
  - 30% Homework (5%, 7.5%, 7.5%, 10%)
  - 40% Project
  - 30% Exam
- **References**
  - Pavan, Schreier & Temes, “Understanding $\Delta\Sigma$ …”
  - Chan Carusone, Johns & Martin, “Analog IC …”
  - Razavi, “Design of Analog CMOS ICs”

Lecture Plan

Date	Lecture (Wednesday 2-4pm)		Reference	Homework
2020-01-07	1	MOD1 & MOD2	PST 2, 3, A	1: Matlab MOD1&2
2020-01-14	2	MOD*N* + ΔΣ Toolbox	PST 4, B	2: ΔΣ Toolbox
2020-01-21	3	SC Circuits	R 12, CCJM 14	
2020-01-28	4	Comparator & Flash ADC	CCJM 10	3: Comparator
2020-02-04	5	Example Design 1	PST 7, CCJM 14	
2020-02-11	6	Example Design 2	CCJM 18	4: SC MOD2
2020-02-18	Reading Week / ISSCC			
2020-02-25	7	Amplifier Design 1		Project
2020-03-03	8	Amplifier Design 2		
2020-03-10	9	Noise in SC Circuits		
2020-03-17	10	Nyquist-Rate ADCs	CCJM 15, 17	
2020-03-24	11	Mismatch & MM-Shaping	PST 6	
2020-03-31	12	Continuous-Time ΔΣ	PST 8	
2020-04-07	Exam			
2020-04-21	Project Presentation (Project Report Due at start of class)			

What you will learn…

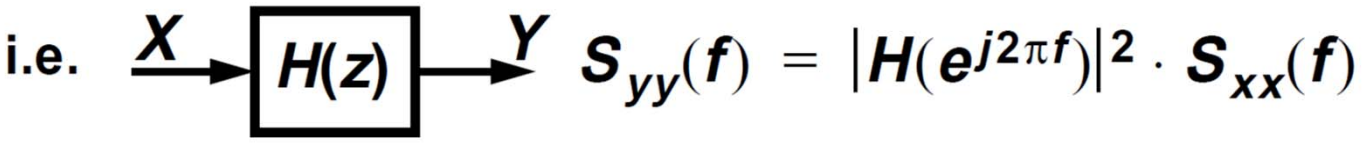
- **MOD1: 1st-order ΔΣ modulator**
  - **Structure and theory of operation**
- **Inherent linearity of binary modulators**
- **Inherent anti-aliasing of continuous-time modulators**
- **MOD2: 2nd-order ΔΣ modulator**
- **Good FFT practice**

Background

- **The Signal-to-Quantization Noise Ratio (SQNR) of an ideal *n*-bit ADC with a full-scale sine-wave input is (6.02*n* + 1.76) dB**

  "6 dB = 1 bit"

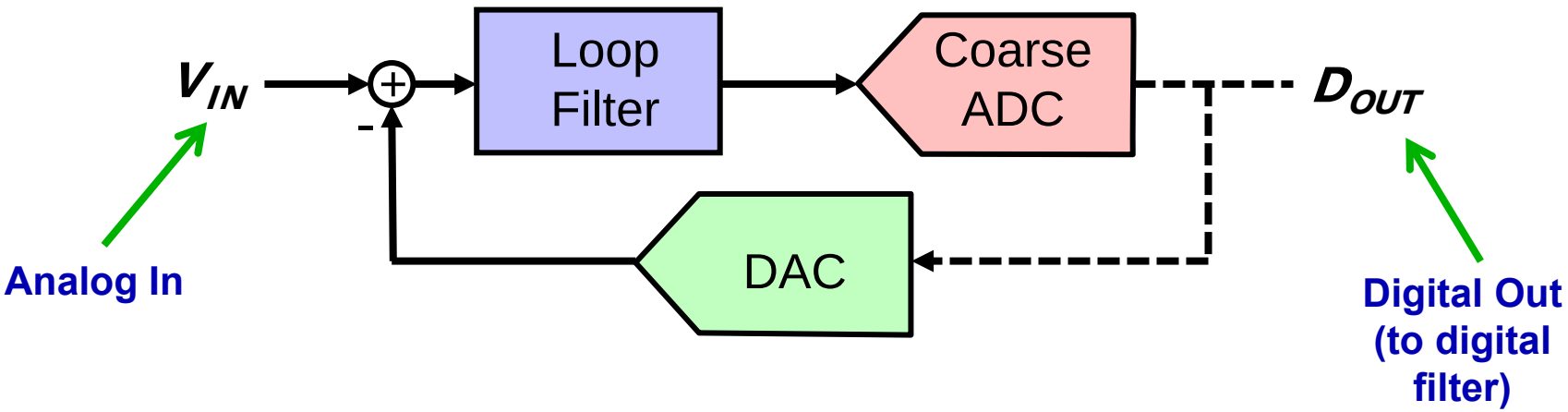
- **The PSD at the output of a linear system is the product of the input's PSD and the squared magnitude of the system's frequency response**



- **The power in any frequency band is the integral of the PSD over that band**

What is ΔΣ?

- **Simplified ΔΣ ADC structure**



- **Key features: course quantization, filtering, feedback and oversampling**

  Quantization is often quite course (as low as 1 bit), but the effective resolution can be as high as 22 bits

What is Oversampling?

- **Oversampling is sampling faster than required by the Nyquist criterion**

  For a lowpass signal containing energy in the frequency range (0, f_B), the minimum sample rate required for perfect reconstruction is $f_S = 2f_B$

- **Oversampling Ratio $OSR \equiv f_S/2f_B$**

- **For a regular ADC, OSR ~ 2-3**

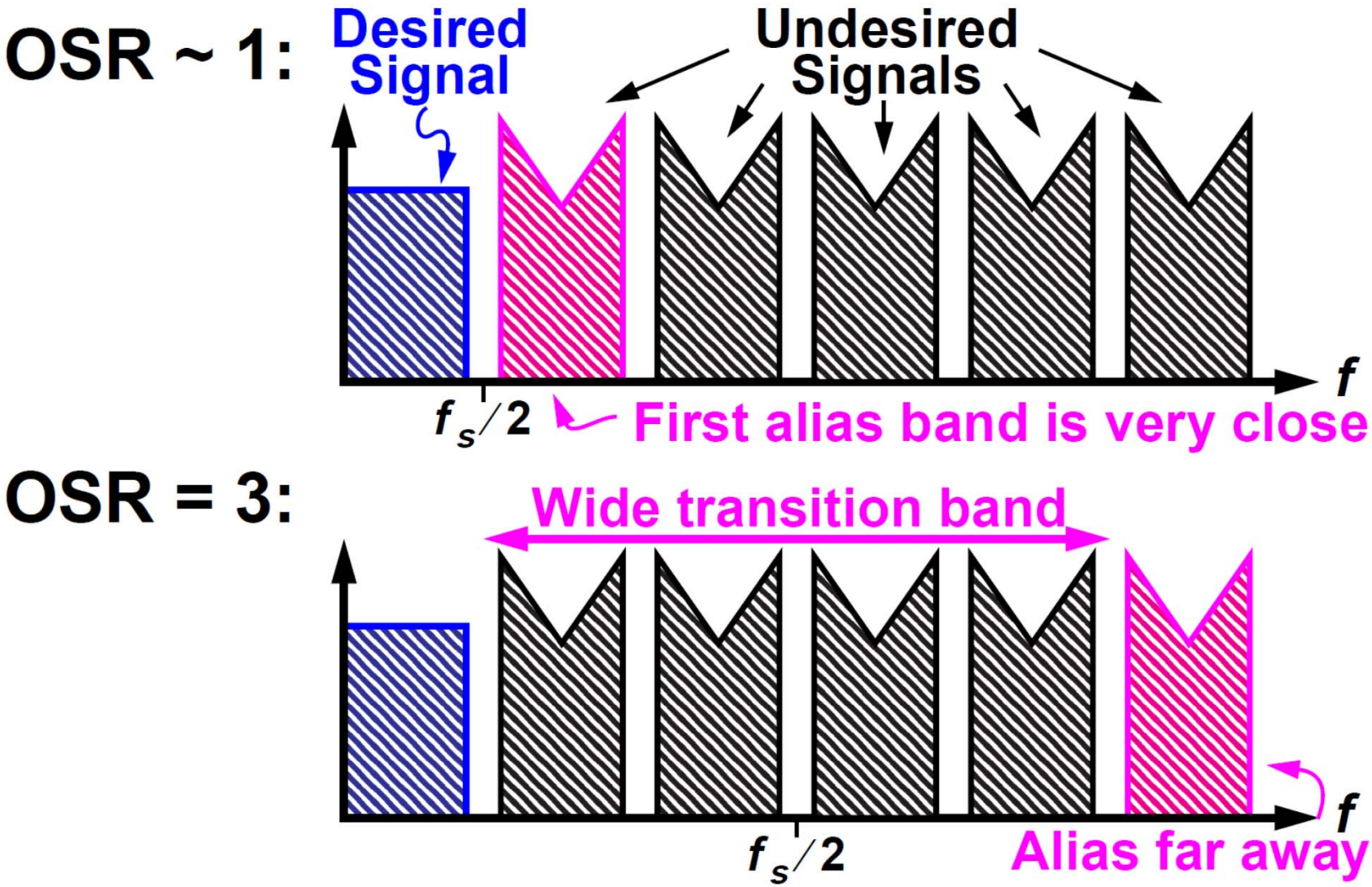
  Larger than 1 to make the anti-alias filter (AAF) feasible

- **For a $\Delta\Sigma$ ADC, OSR ~ 8 to 200**

  To get adequate quantization noise suppression

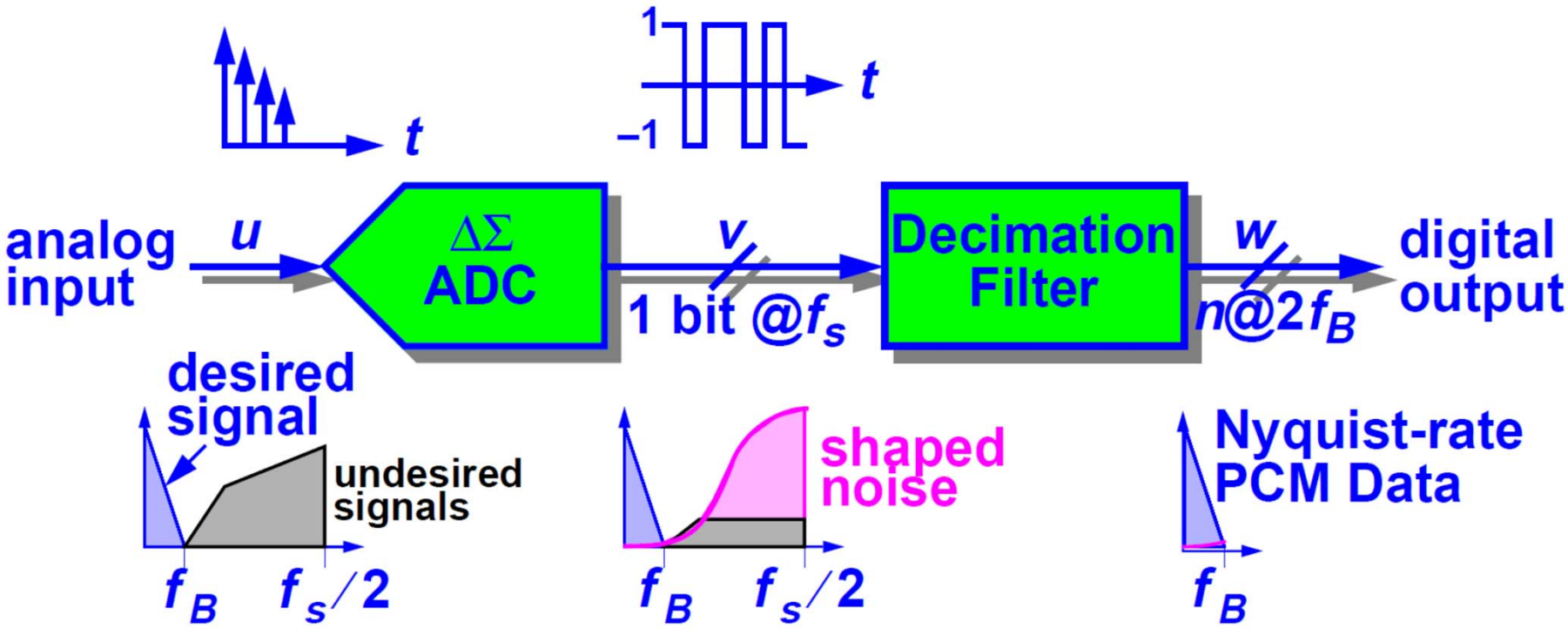
  Signals between f_B and ~f_S are removed digitally

Oversampling Simplifies AAF

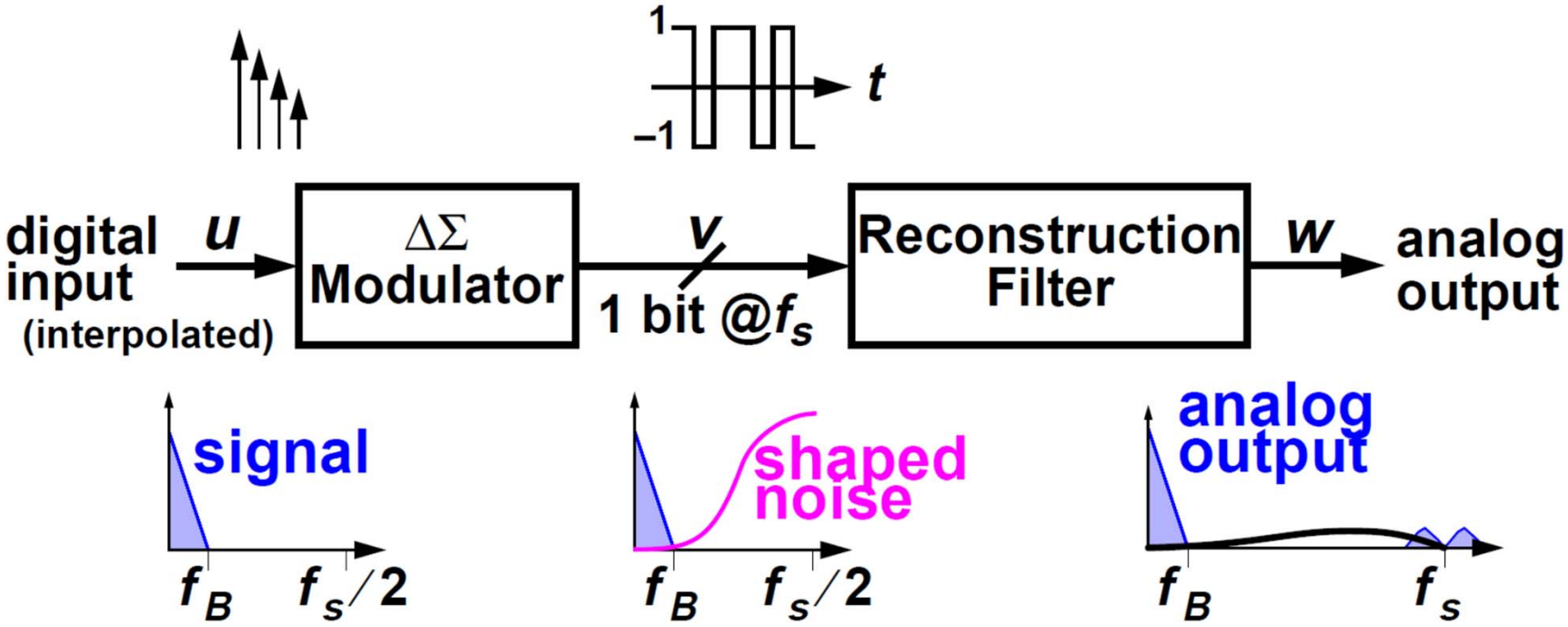


How The ΔΣ ADC Works

- Course quantization ➔ lots of quantization error
  So how can a ΔΣ ADC achieve 22-bit resolution?
- A ΔΣ ADC spectrally separates the quantization error from the signal through noise-shaping



A ΔΣ DAC System



- **Mathematically similar to an ADC system**

  Except that now the modulator is digital and drives a low-resolution DAC, and the out-of-band noise is handled by an analog reconstruction filter

Why Do It The ΔΣ Way?

- **Simplified Anti-Alias Filter in ADC**
  - **Since the input is oversampled, only very high frequencies alias to the passband**
  - **Simple RC filter is usually sufficient**
  - **If a continuous-time loop filter is used, the anti-alias filter can often be eliminated altogether**
  - **DAC: Simplified Reconstruction Filter**

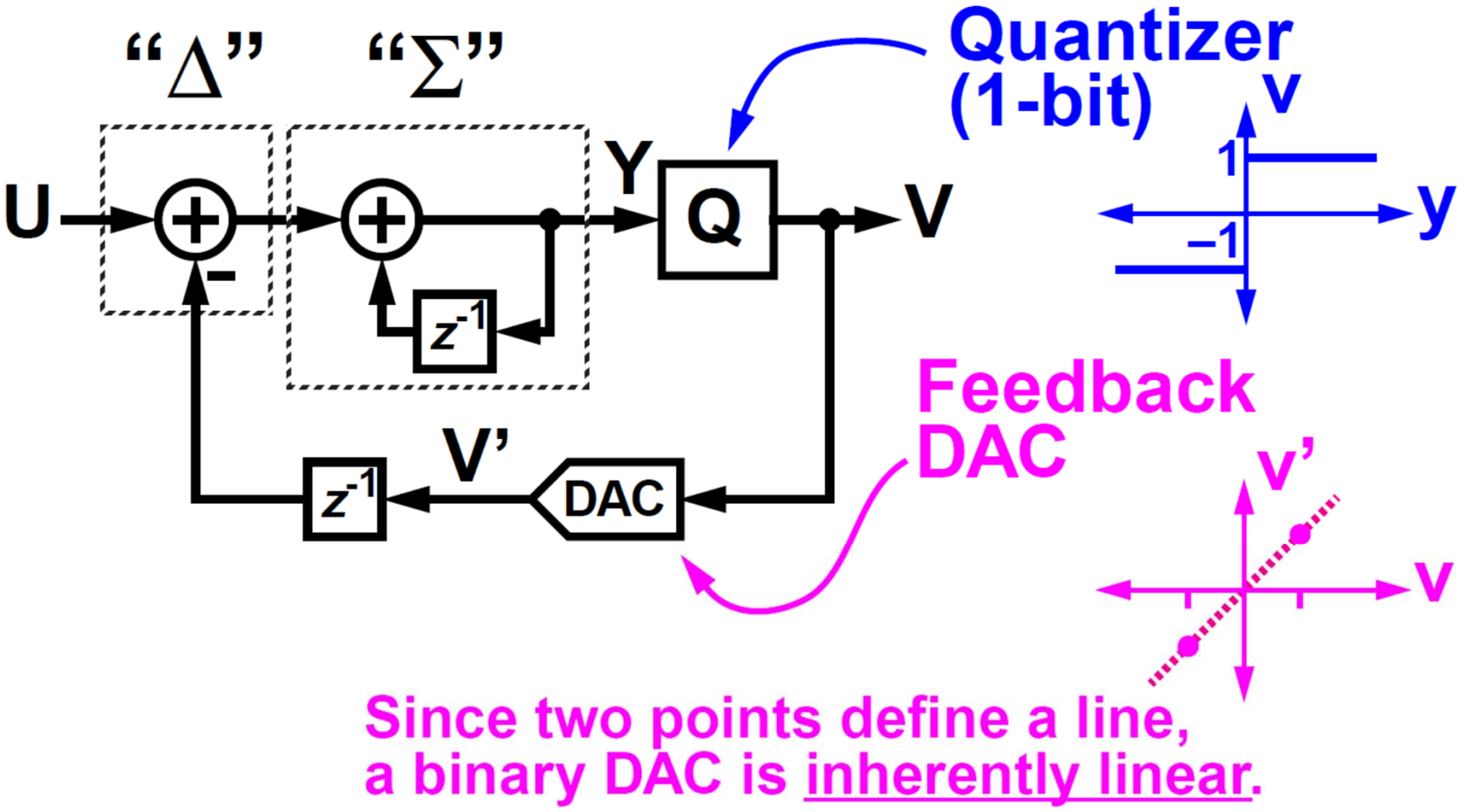
☺ **Inherent Linearity**

- **Simple structures can yield very high SNR**

☺ **Robust Implementation**

- **ΔΣ tolerates sizable component errors**

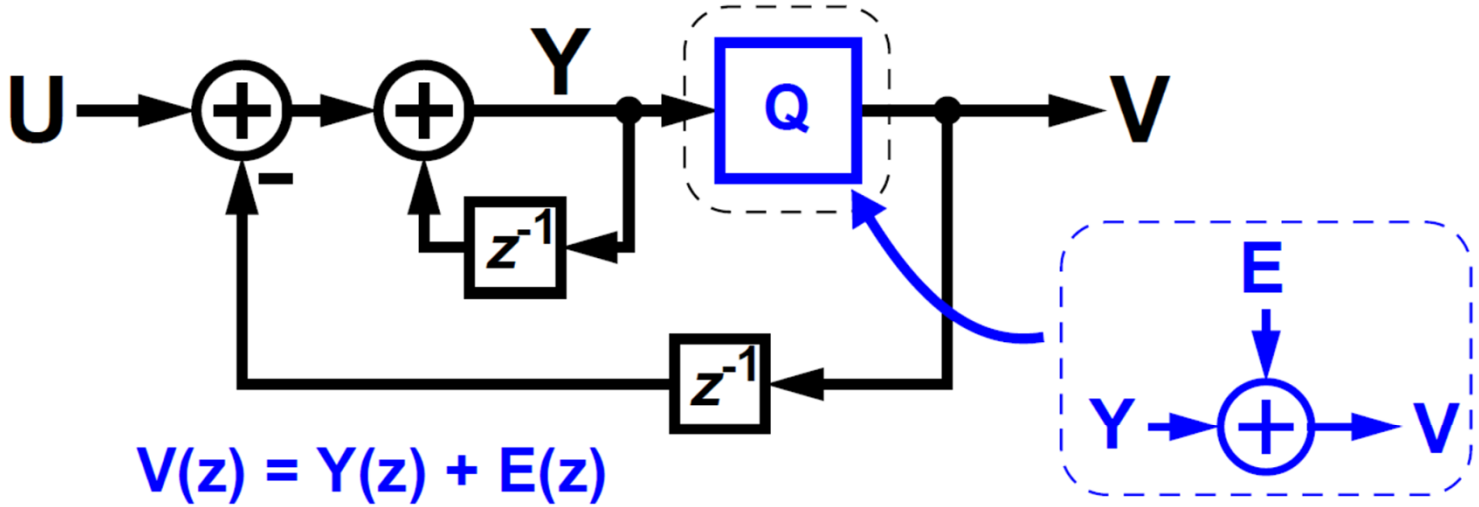
MOD1: 1st-Order ΔΣ Modulator



Since two points define a line,
a binary DAC is inherently linear.

MOD1 Analysis

- **Exact analysis is intractable for all but the simplest inputs, so treat the quantizer as an additive noise source:**



$$V(z) = Y(z) + E(z)$$

$$Y(z) = (U(z) - z^{-1}V(z)) / (1-z^{-1})$$

$$\Rightarrow (1-z^{-1})\, V(z) = U(z) - z^{-1}V(z) + (1-z^{-1})E(z)$$

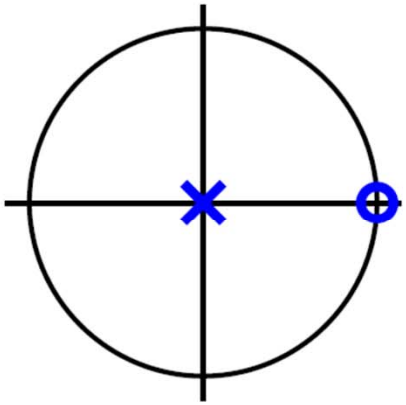
$$\boxed{V(z) = U(z) + (1-z^{-1})E(z)}$$

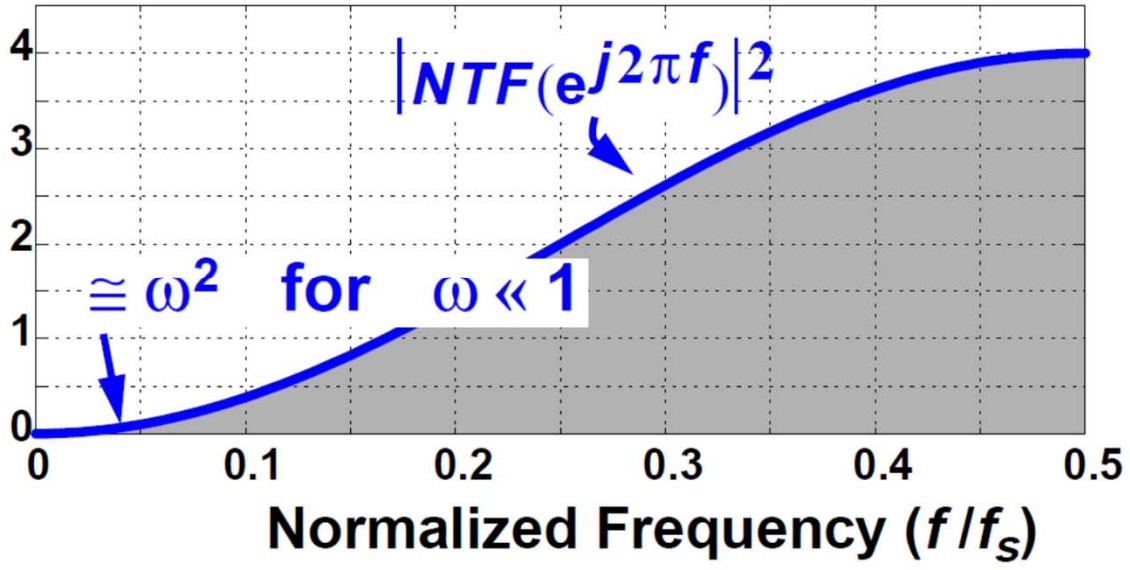
The Noise Transfer Function (NTF)

- In general, $V(z) = STF(z) \cdot U(z) + NTF(z) \cdot E(z)$
- For MOD1, $NTF(z) = 1 - z^{-1}$

  → The quantization noise has spectral shape!

Poles & zeros:





- The total noise power increases, but the noise power at low frequencies is reduced

In-band Quantization Noise Power

- **Assume the error is white with power σ_e^2**

  i.e. $S_{ee}(\omega) = \sigma_e^2/\pi$

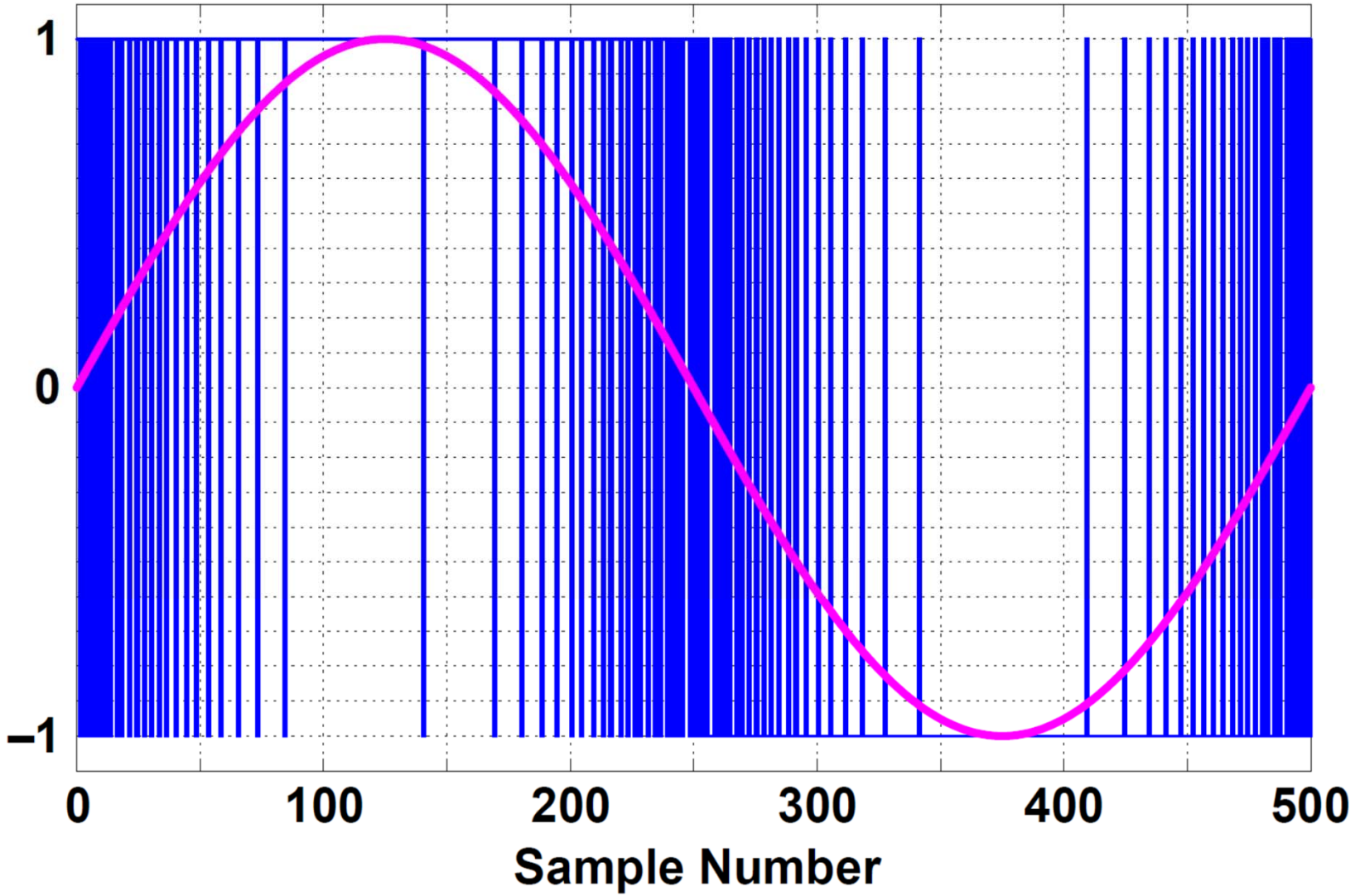
- **The in-band quatization noise power is**

$$IQNP = \int_0^{\omega_B} \left|H\left(e^{j\omega}\right)\right|^2 S_{ee}(\omega)d\omega \cong \frac{\sigma_e^2}{\pi}\int_0^{\omega_B} \omega^2\, d\omega$$

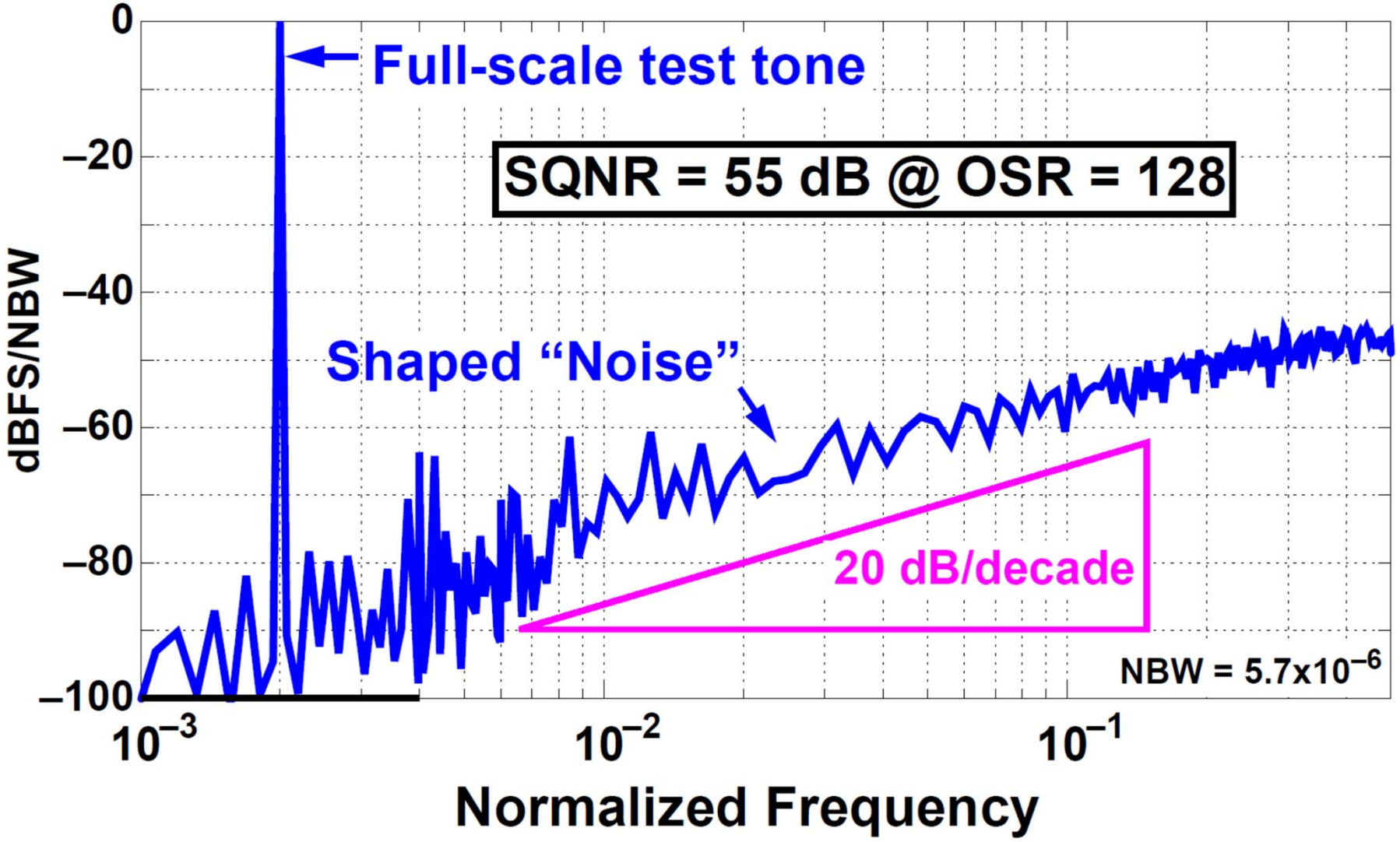
- **Since $OSR = \frac{\pi}{\omega_B}$, $IQNP = \frac{\pi^2\sigma_e^2}{3}OSR^{-3}$**
- **For MOD1, an octave increase in OSR increases SQNR by 9 dB**

  1.5-bit/octave SQNR-OSR trade-off

A Simulation of MOD1 (Time)

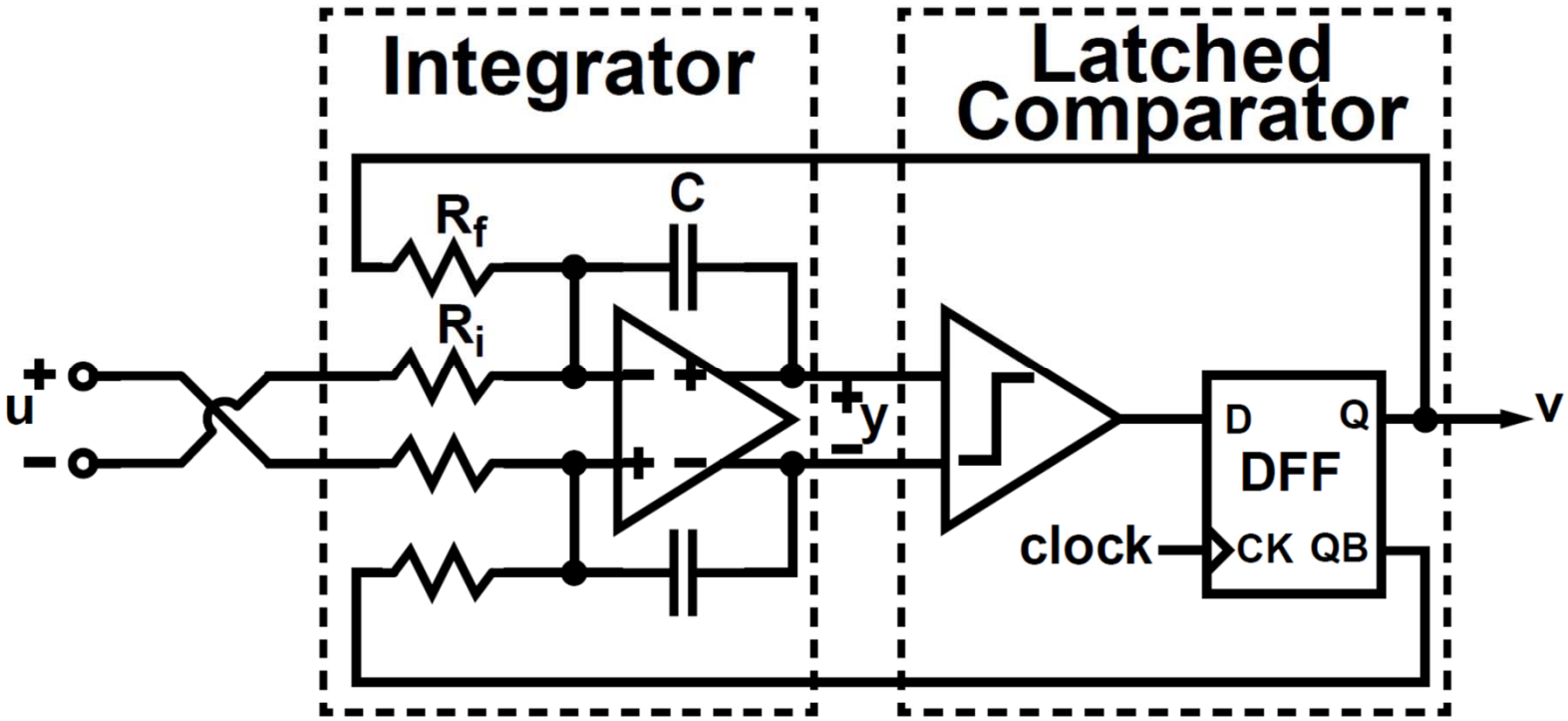


A Simulation of MOD1 (Freq)

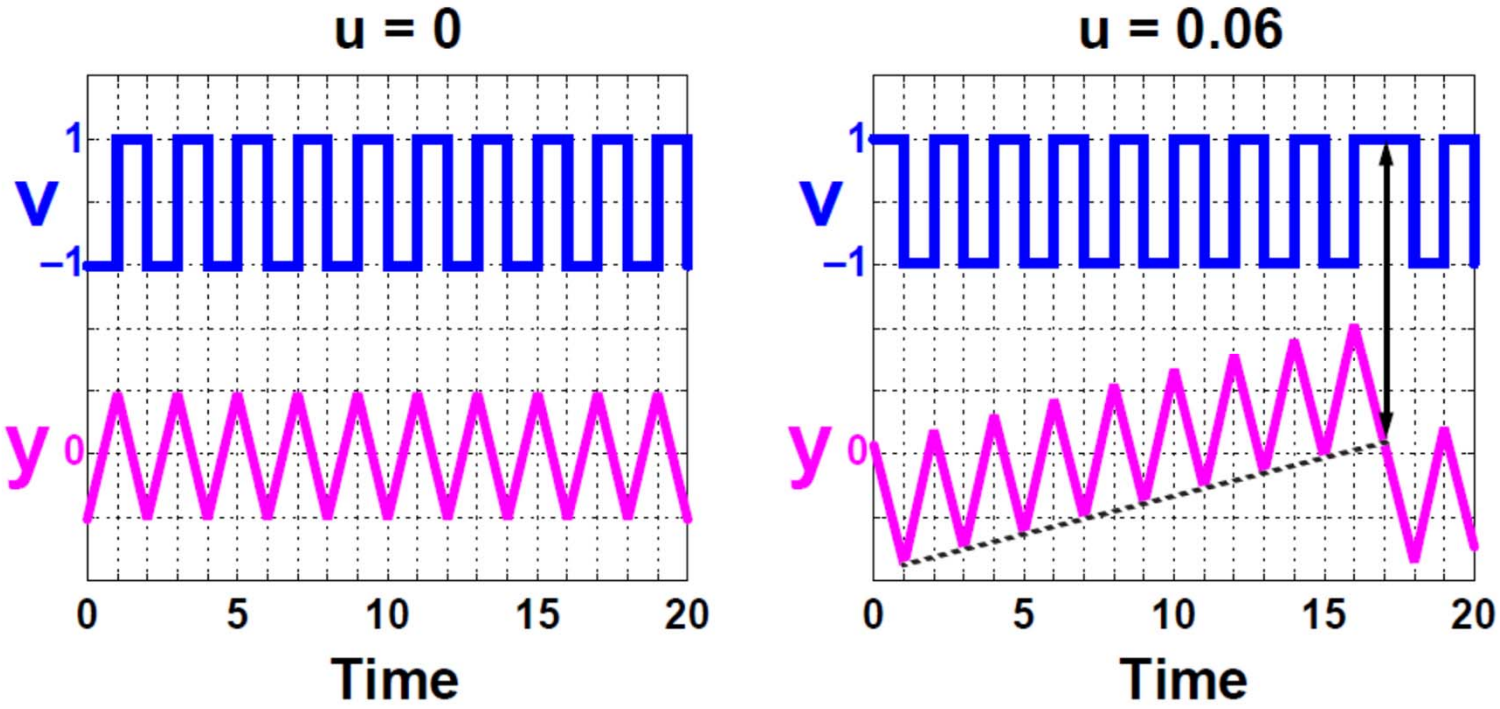


CT Implementation of MOD1

- **R_i/R_f sets the full-scale; C is arbitrary**
  - **R_i, R_f are typically sized based on noise**
  - **Also observe that an input at f_S is rejected by the integrator – *inherent anti-aliasing***



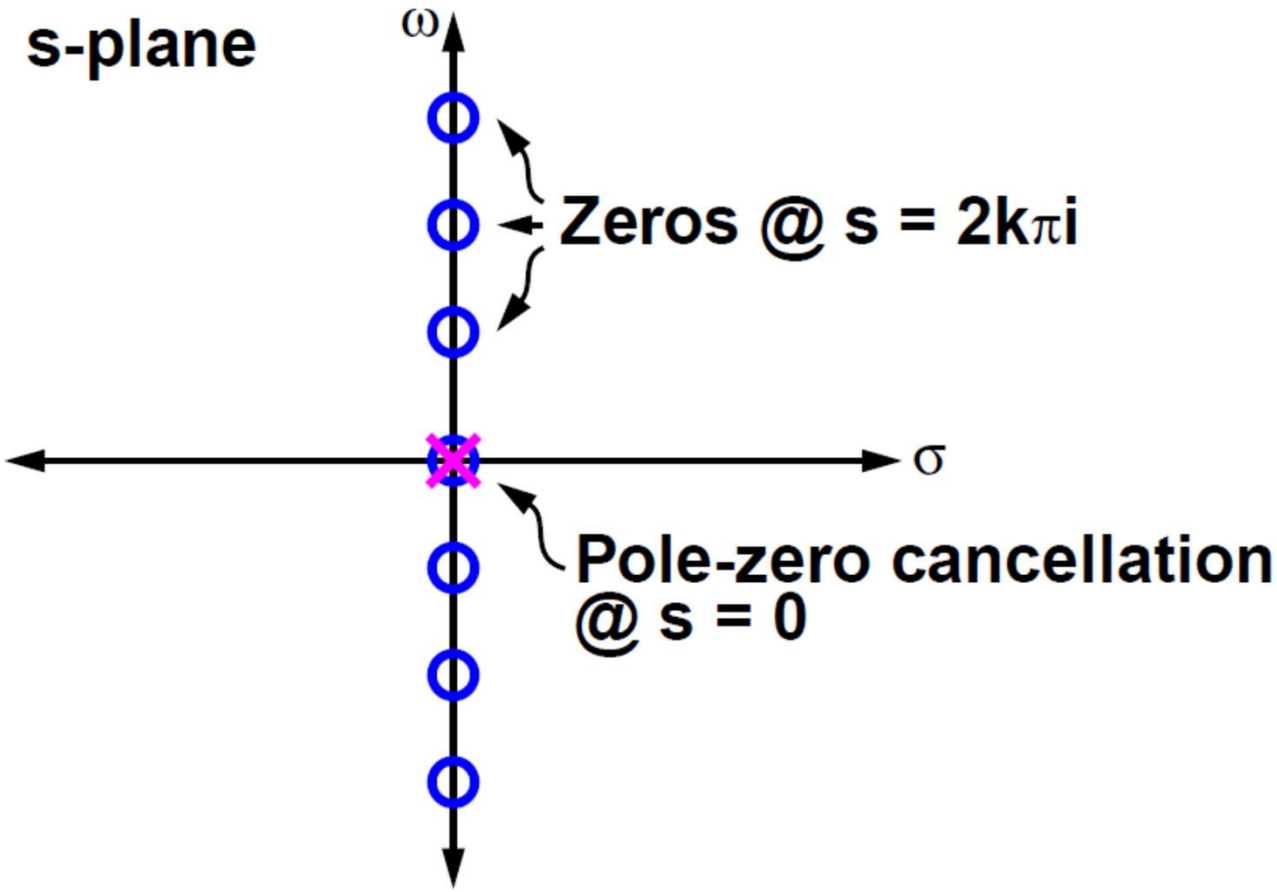
MOD1-CT Waveforms



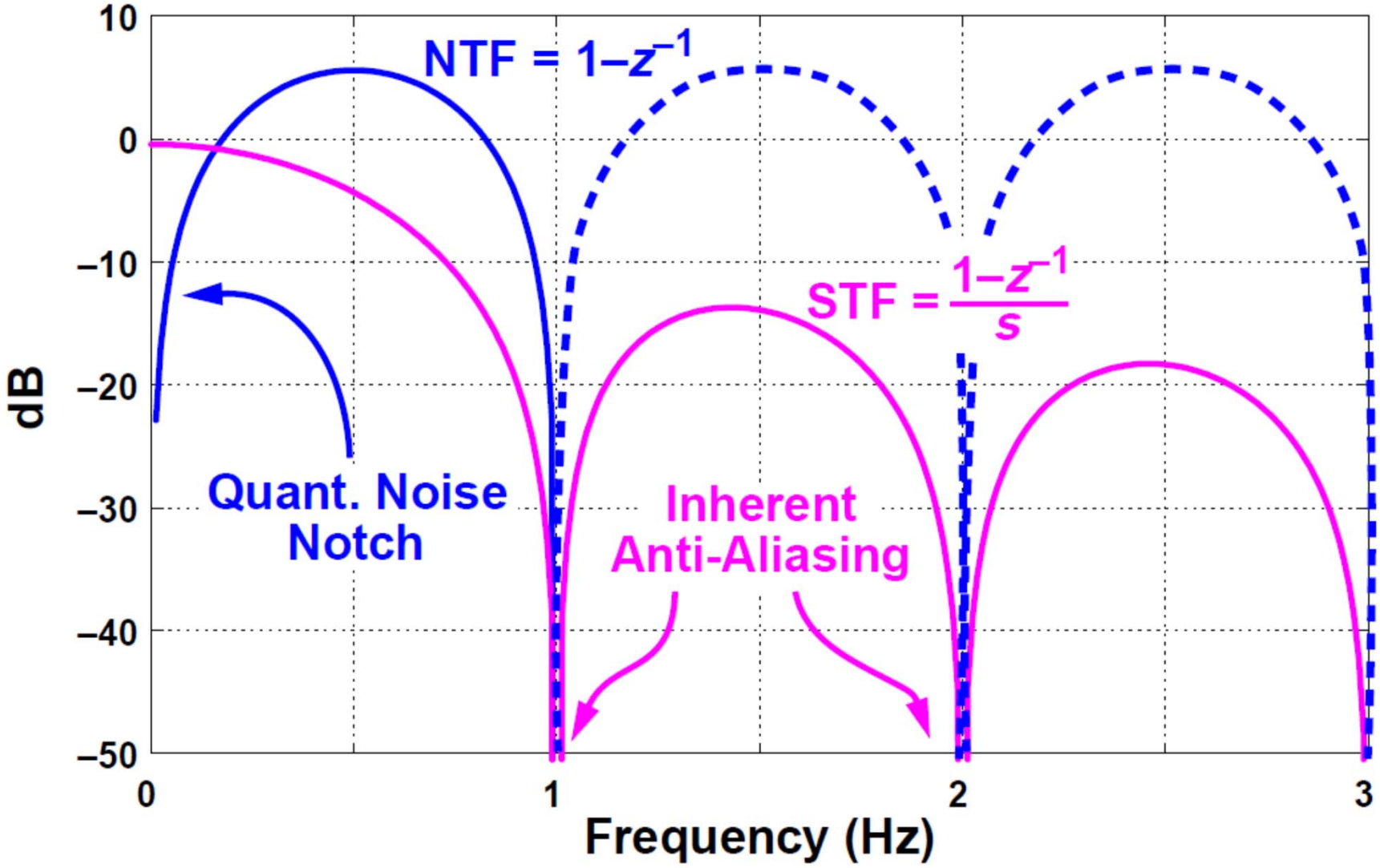
- With u=0, v alternates between +1 and -1
- With u>0, y drifts upwards; v contains consecutive +1s to counteract this drift

MOD1-CT STF

- $STF = \frac{1-z^{-1}}{s}$, recall $z = e^{s}$



MOD1-CT Frequency Responses

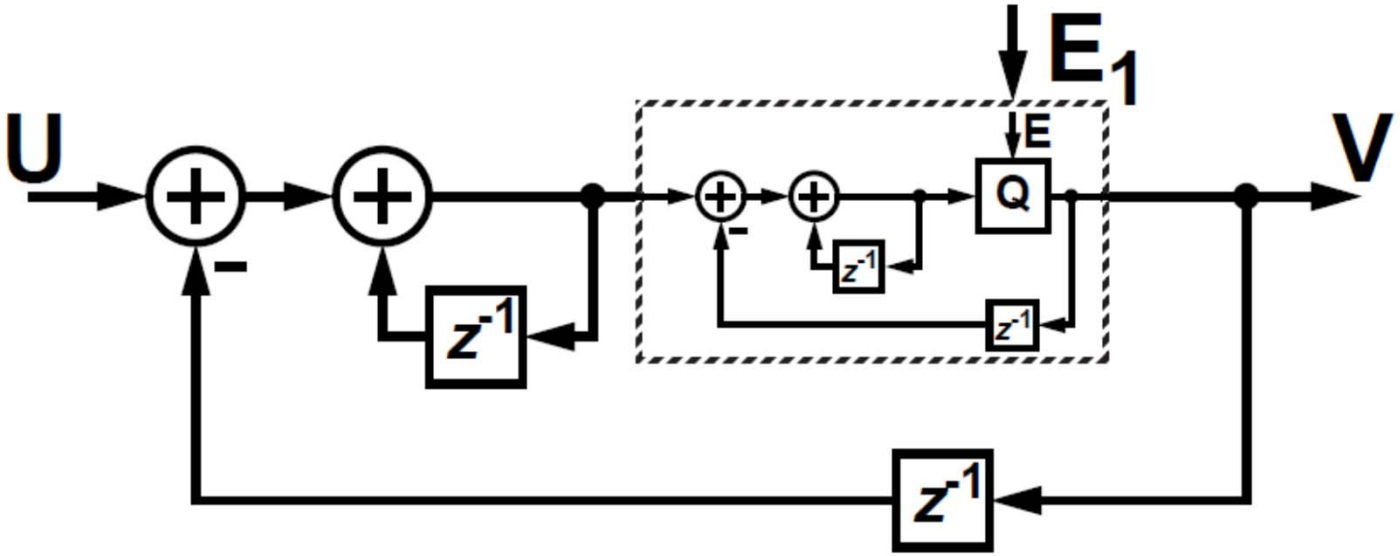


Summary

- **$\Delta\Sigma$ works by spectrally separating the quantization noise from the signal**
  - **Requires oversampling $OSR \equiv f_S/2f_B$**
- **Noise-shaping is achieved by the use of *filtering* and *feedback***
- **A binary DAC is *inherently linear*, and thus a binary $\Delta\Sigma$ modulator is too**
- **MOD1 has $NTF(z) = 1 - z^{-1}$**
  - **Arbitrary accuracy for DC inputs**
  - **1.5 bit/octave SQNR-OSR trade-off**
- **MOD1-CT has *inherent anti-aliasing***

MOD2: 2nd-Order ΔΣ Modulator

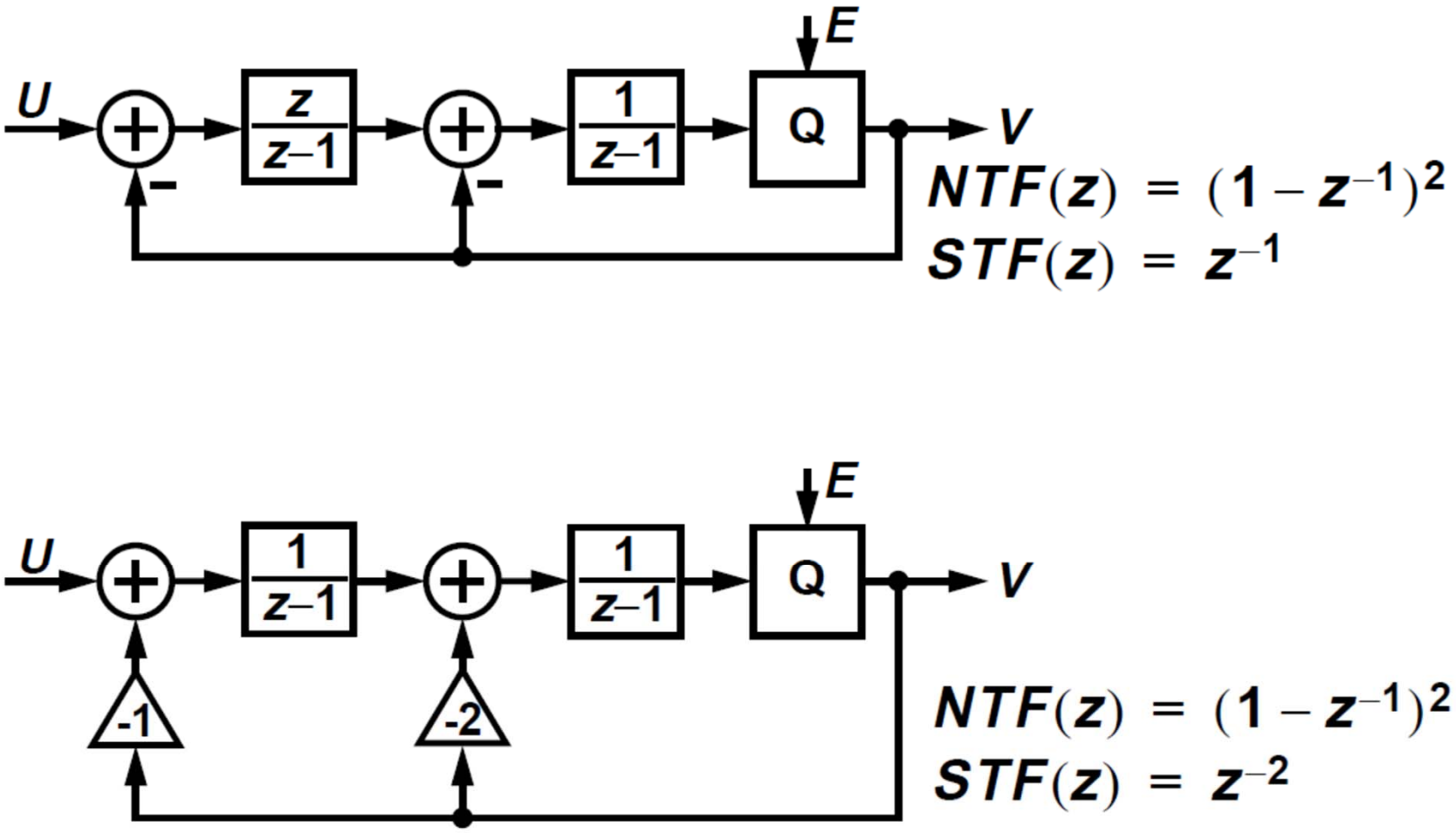
- **Replace the quantizer in MOD1 with another copy of MOD1 in a recursive fashion**



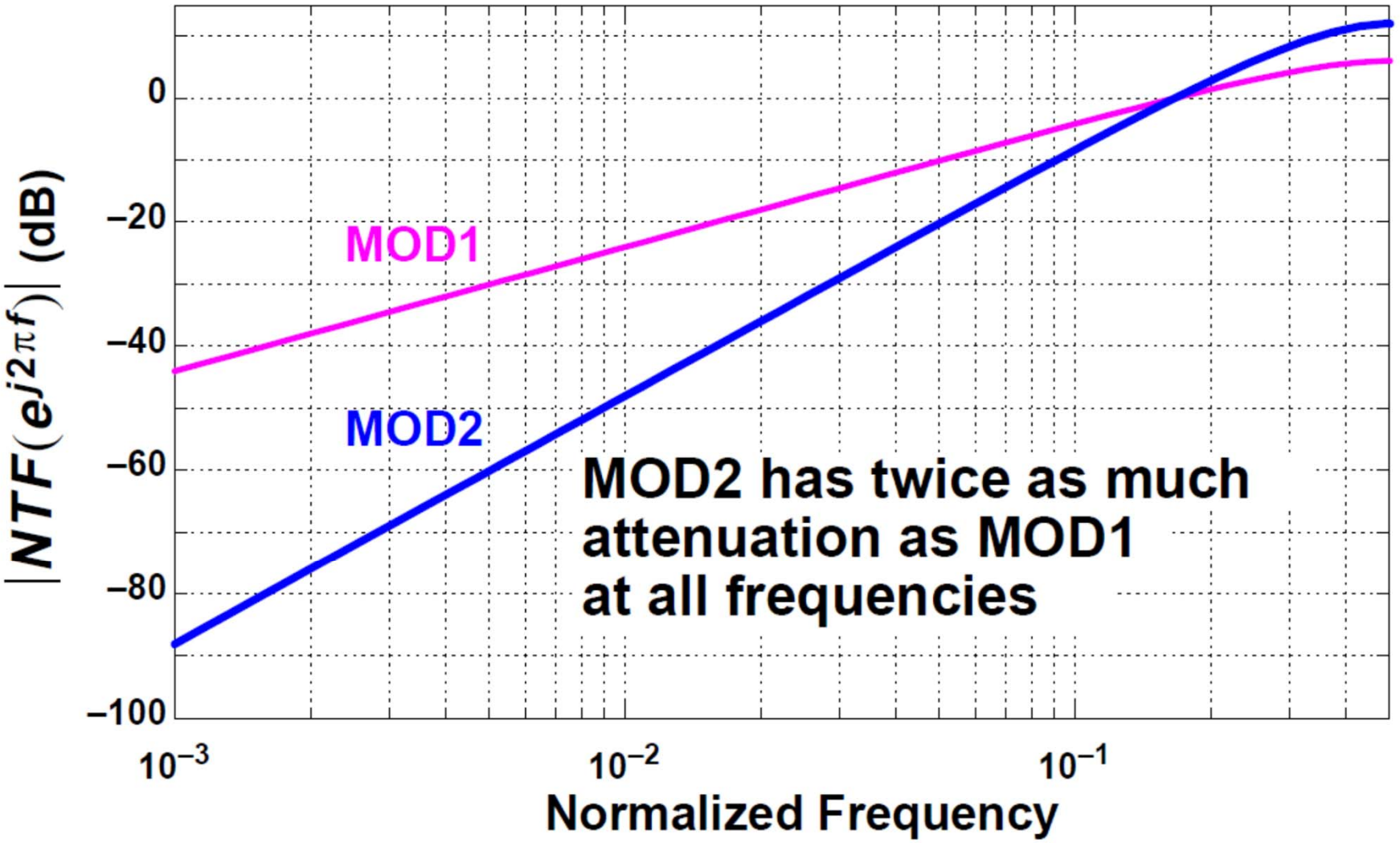
$V(z) = U(z) + (1-z^{-1})E_1(z)$, $E_1(z) = (1-z^{-1})E(z)$

→ $V(z) = U(z) + (1-z^{-1})^2E(z)$

Simplified Block Diagrams



NTF Comparison



In-Band Quantization Noise Power

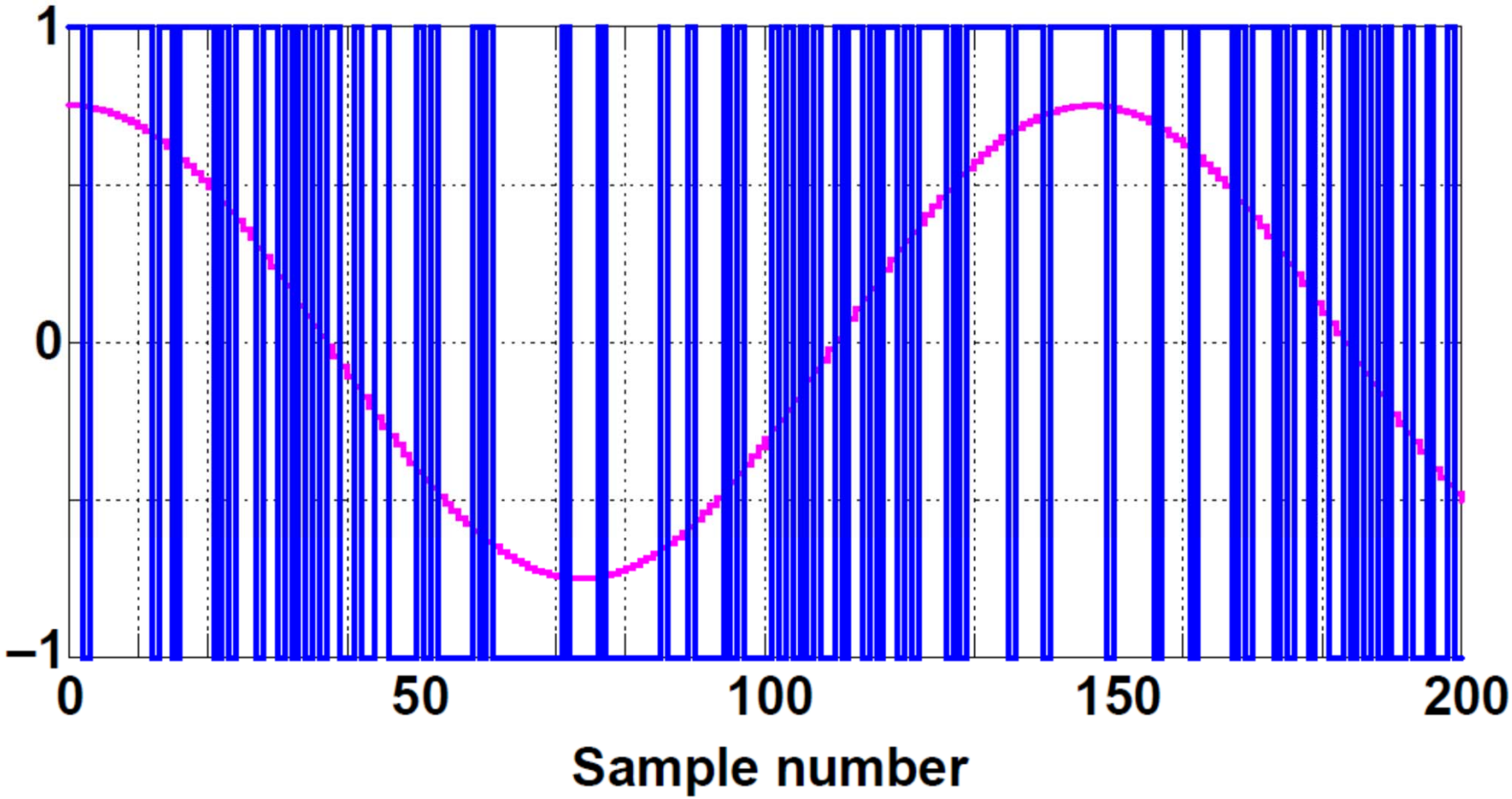
- For MOD2, $\left|H(e^{j\omega})\right|^2 \approx \omega^4$
- As before, $IQNP = \int_0^{\omega_B} \left|H\left(e^{j\omega}\right)\right|^2 S_{ee}(\omega)d\omega$ and $S_{ee}(\omega) = \sigma_e^2/\pi$
- So now $IQNP = \frac{\pi^4 \sigma_e^2}{5} OSR^{-5}$

  With binary quantization to +/- 1, $\Delta = 2$ and thus $\sigma_e^2 = \Delta^2/12 = 1/3$

- An octave increase in OSR increases MOD2's SQNR by 15dB (2.5 bits)

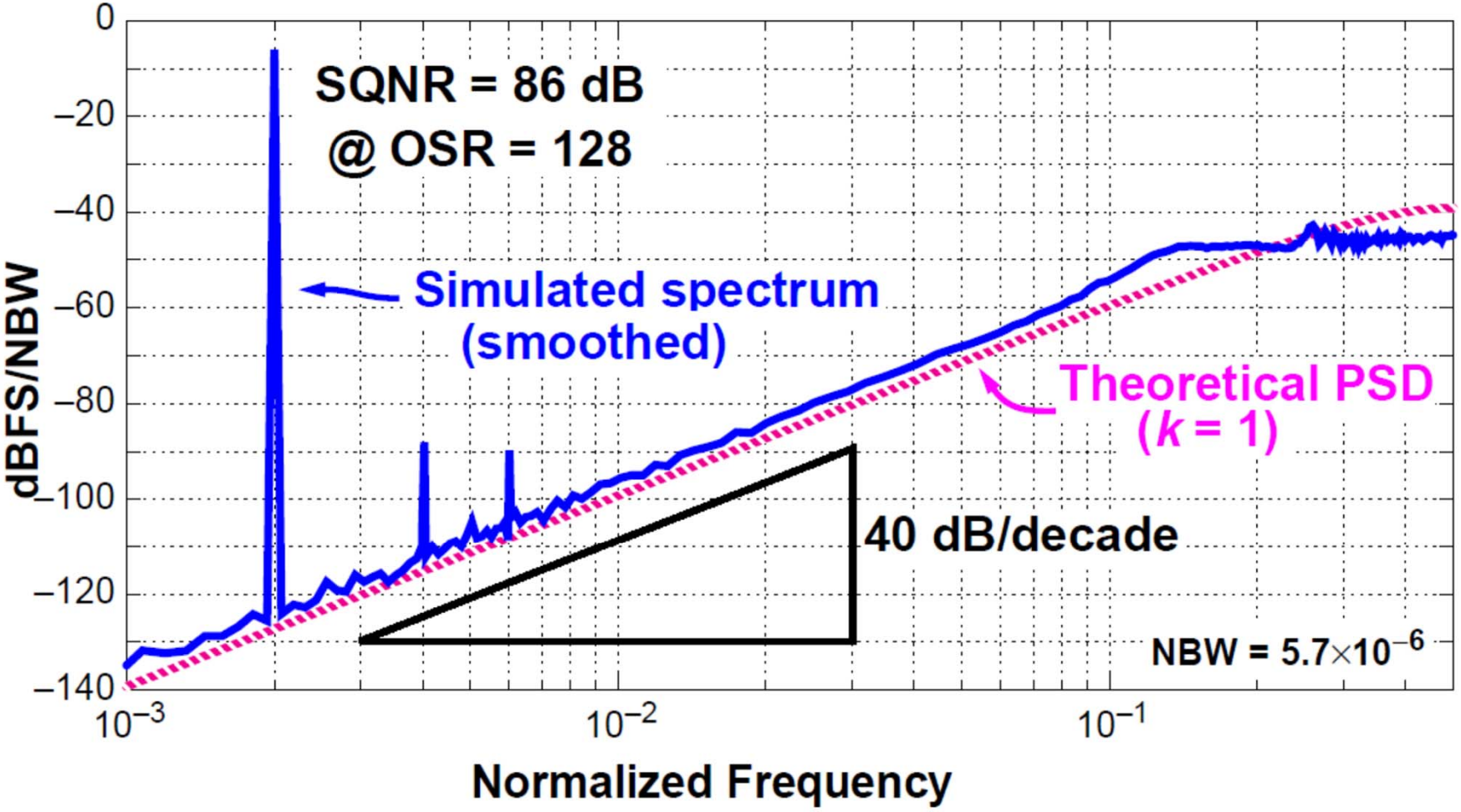
Simulation Example

- **Input at 75% of Full Scale**



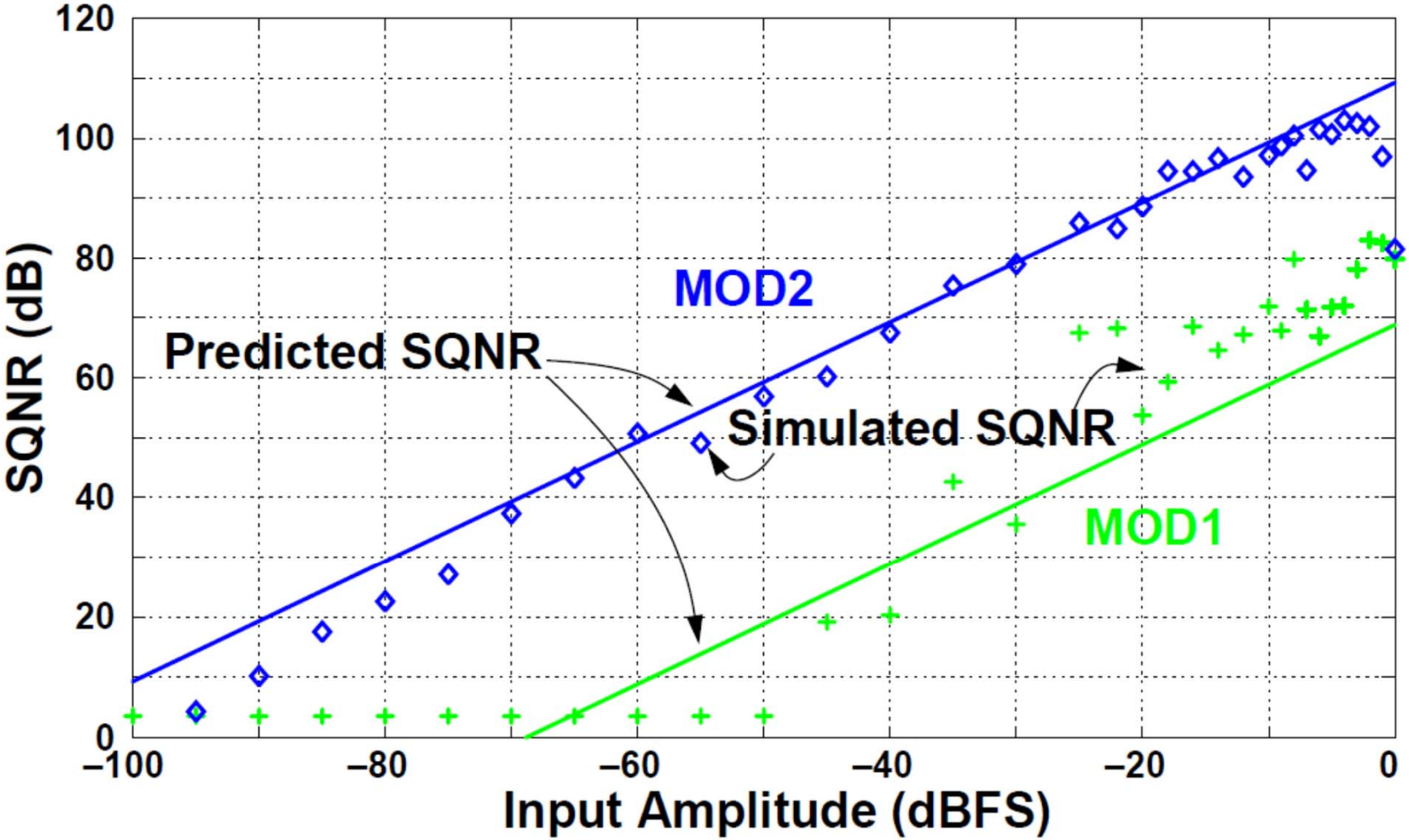
Simulated MOD2 PSD

- **Input at 50% of Full Scale**

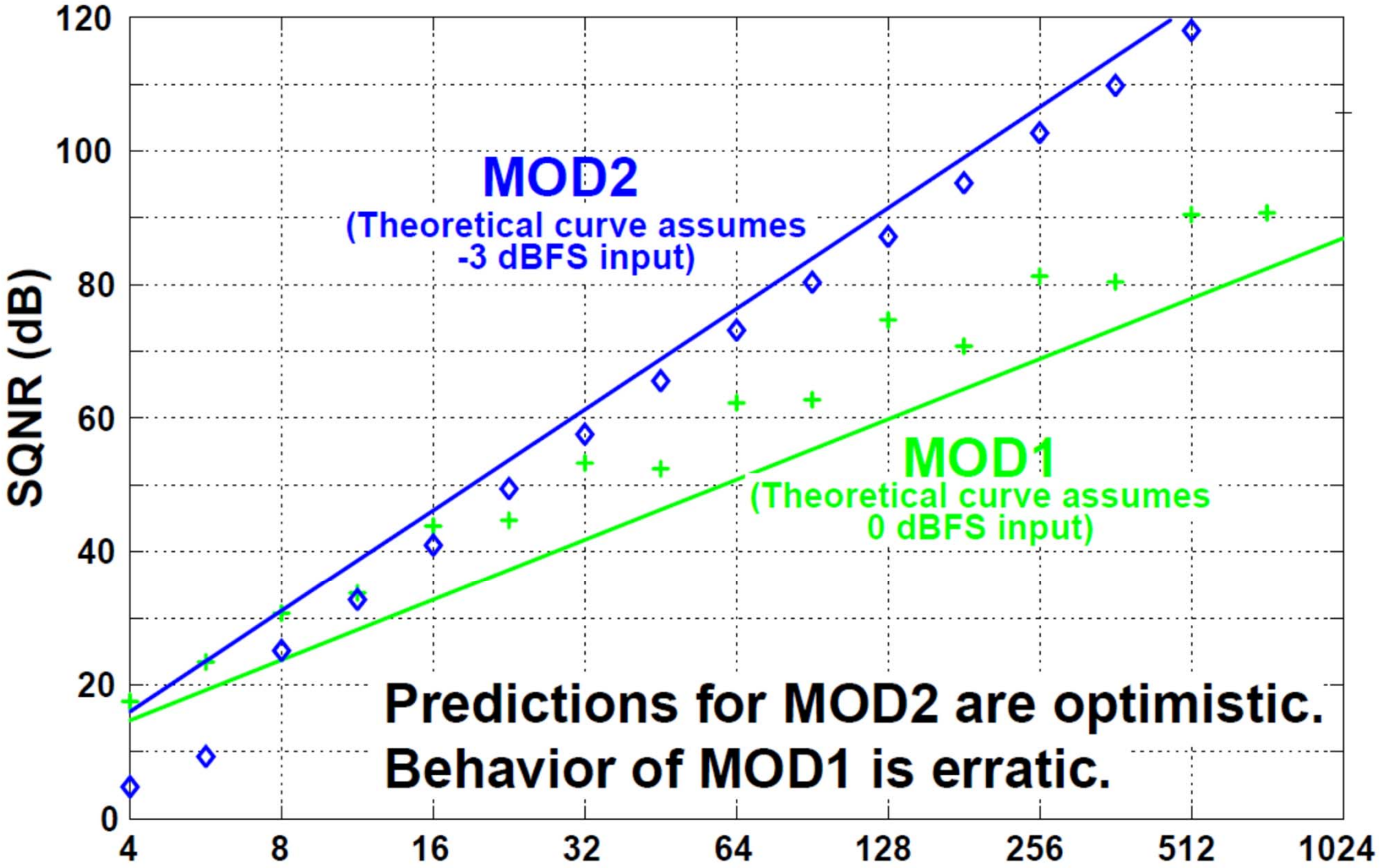


SQNR vs. Input Amplitude

- **MOD1 & MOD2 @ OSR = 256**

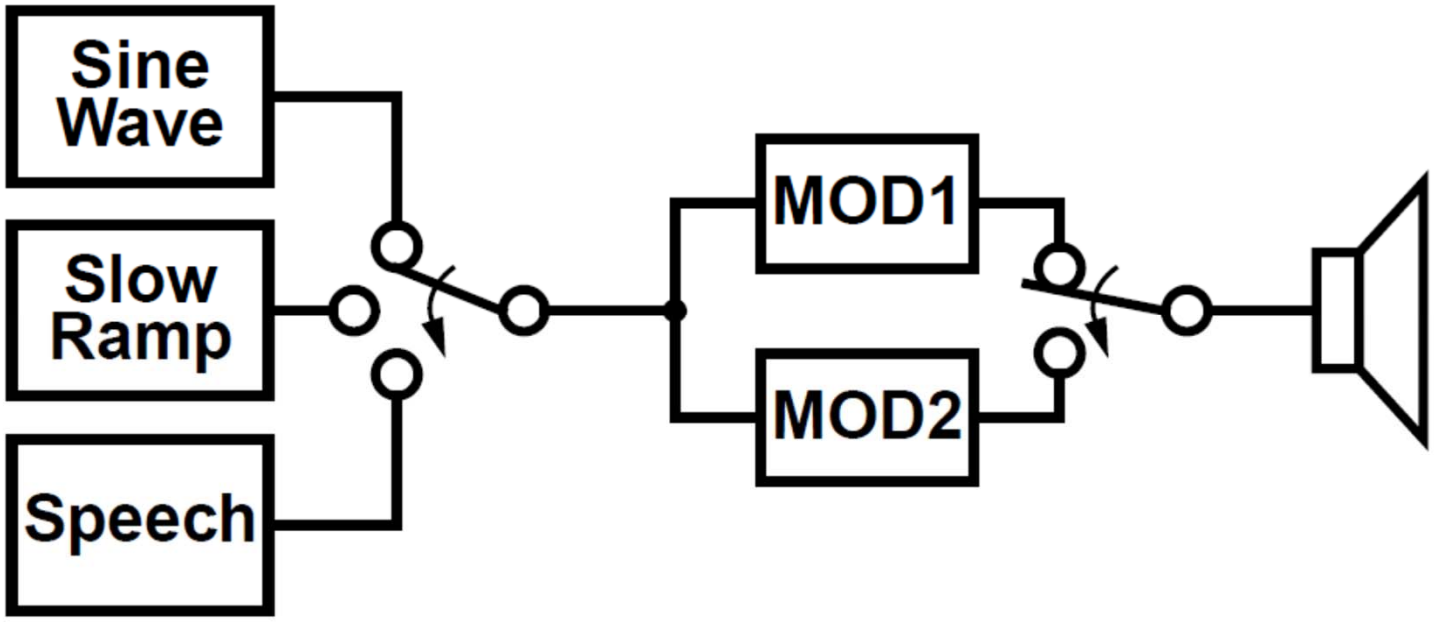


SQNR vs. OSR


SQNR (dB)
120
100
80
60
40
20
0
4
8
16
32
64
128
256
512
1024
MOD2
(Theoretical curve assumes
-3 dBFS input)
MOD1
(Theoretical curve assumes
0 dBFS input)
Predictions for MOD2 are optimistic.
Behavior of MOD1 is erratic.

Audio Demo: MOD1 vs. MOD2

- **dsdemo4 in Matlab ΔΣ Toolbox**

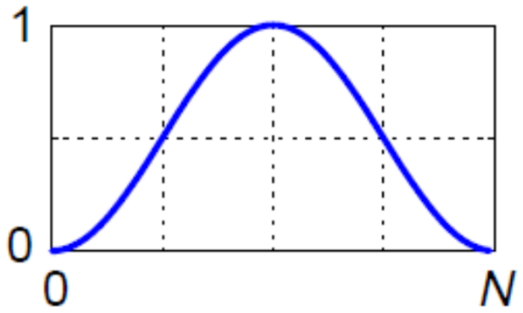


MOD1 and MOD2 Summary

- $\Delta\Sigma$ ADCs rely on filtering and feedback to achieve high SNR despite coarse quantization
  - They also rely on digital signal processing
  - $\Delta\Sigma$ ADCs need to be followed by a digital decimation filter and $\Delta\Sigma$ DACs need to be preceded by a digital interpolation filter
- Oversampling eases analog filtering requirements
  - Anti-alias filter in an ADC; image filter in a DAC
- Binary quantization yields inherent linearity
- MOD2 is better than MOD1
  - 15 dB/octave vs 9 dB/octave SQNR-OSR trade-off
  - Quantization noise more white
  - Higher-order modulators are even better

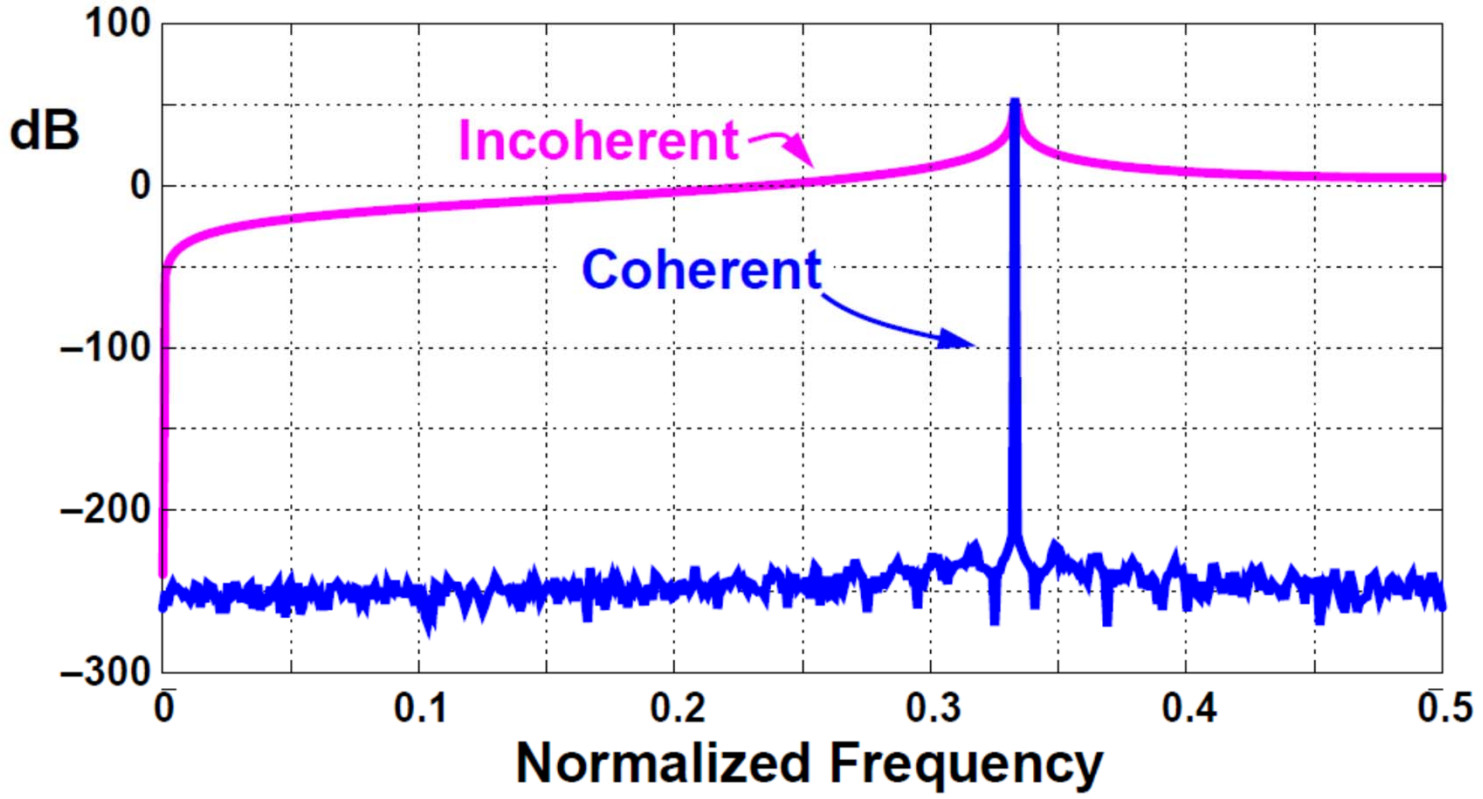
Good FFT Practice

- **Use coherent sampling**
  - Have an integer number of cycles in the record
- **Use windowing**
  - A Hann window works well
  - $w(n) = (1 - cos(2\pi n/N))/2$



- **Use enough points**
  - Recommend $N = 64 \cdot OSR$
- **Scale (and smooth) the spectrum**
  - A full-scale sine wave should yield a 0-dBFS peak
- **State the noise bandwidth**
  - For a Hann window, $NBW = 1.5/N$

Coherent vs Incoherent Sampling



- Coherent sampling: only one non-zero FFT bin
- Incoherent sampling: spectral leakage

Windowing

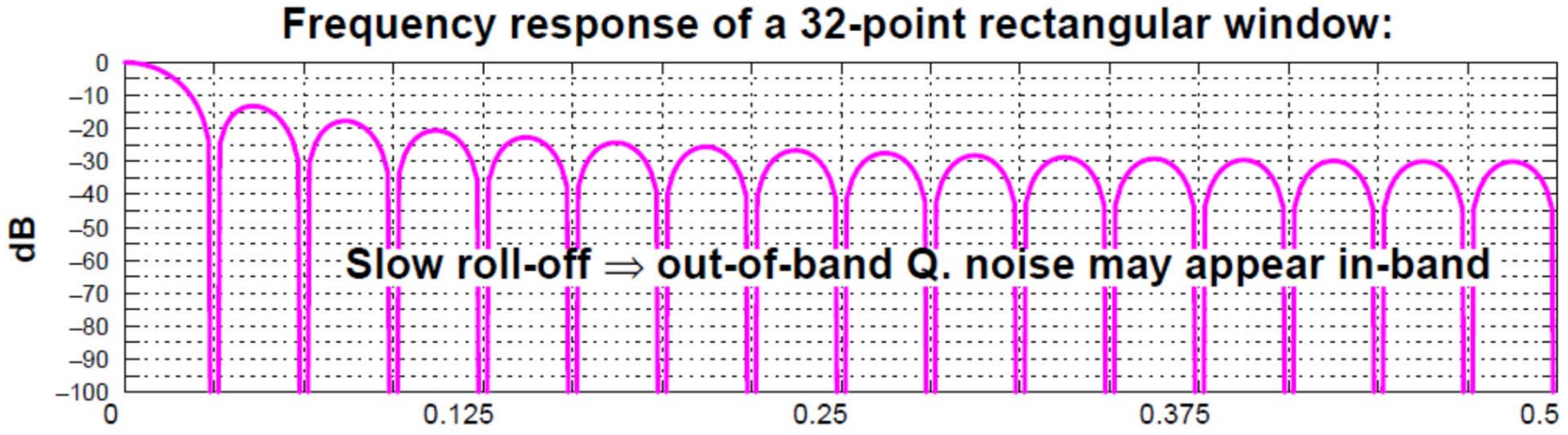
- $\Delta\Sigma$ data is usually not periodic

  Just because the input repeats does not mean the output does too!

- Windowing is unavoidable

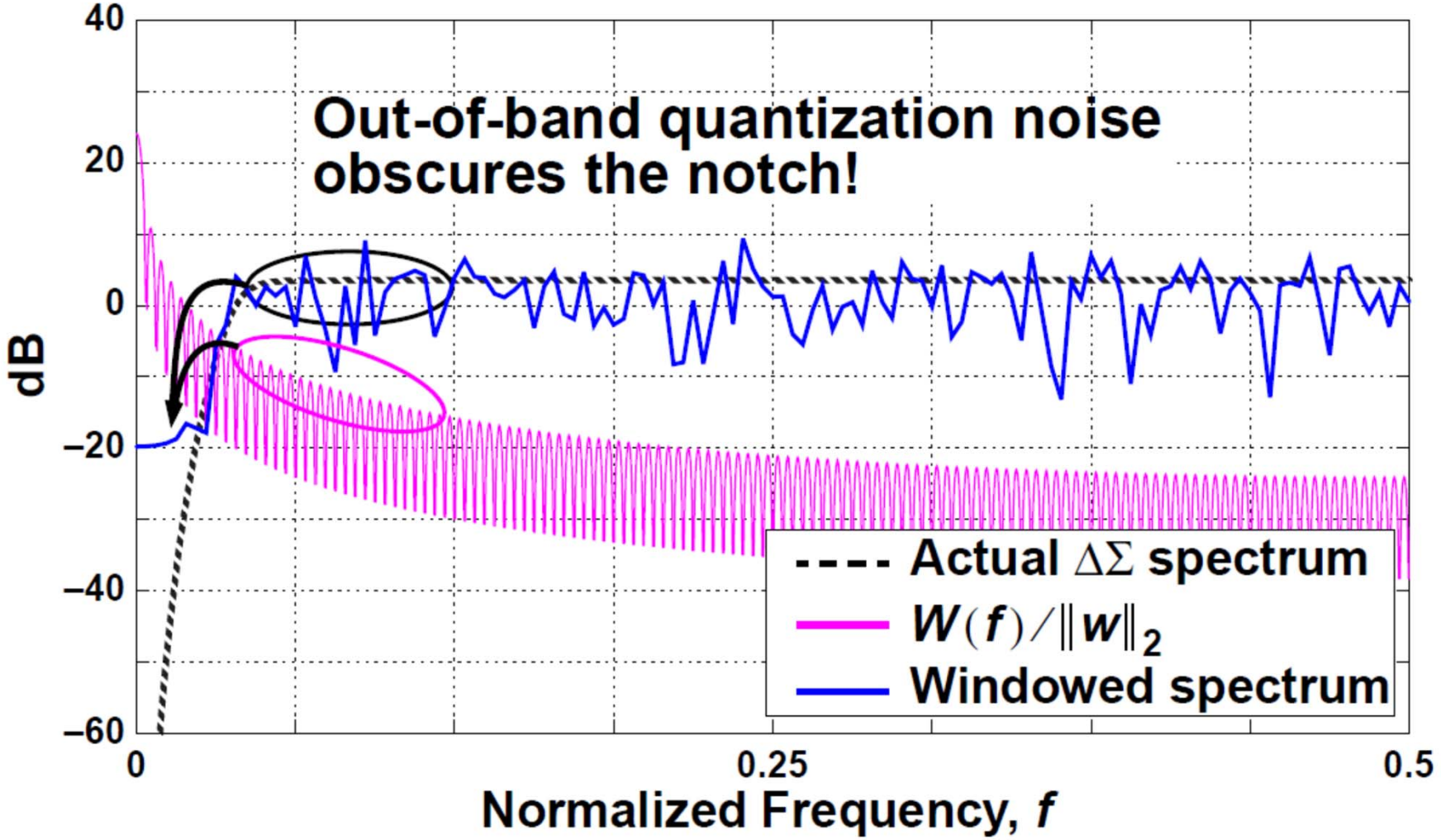
  A finite-length data record is equal to an infinite record multiplied by a *rectangular window* $w(n) = 1, 0 \leq n < N$

- Multiplication in time is convolution in frequency

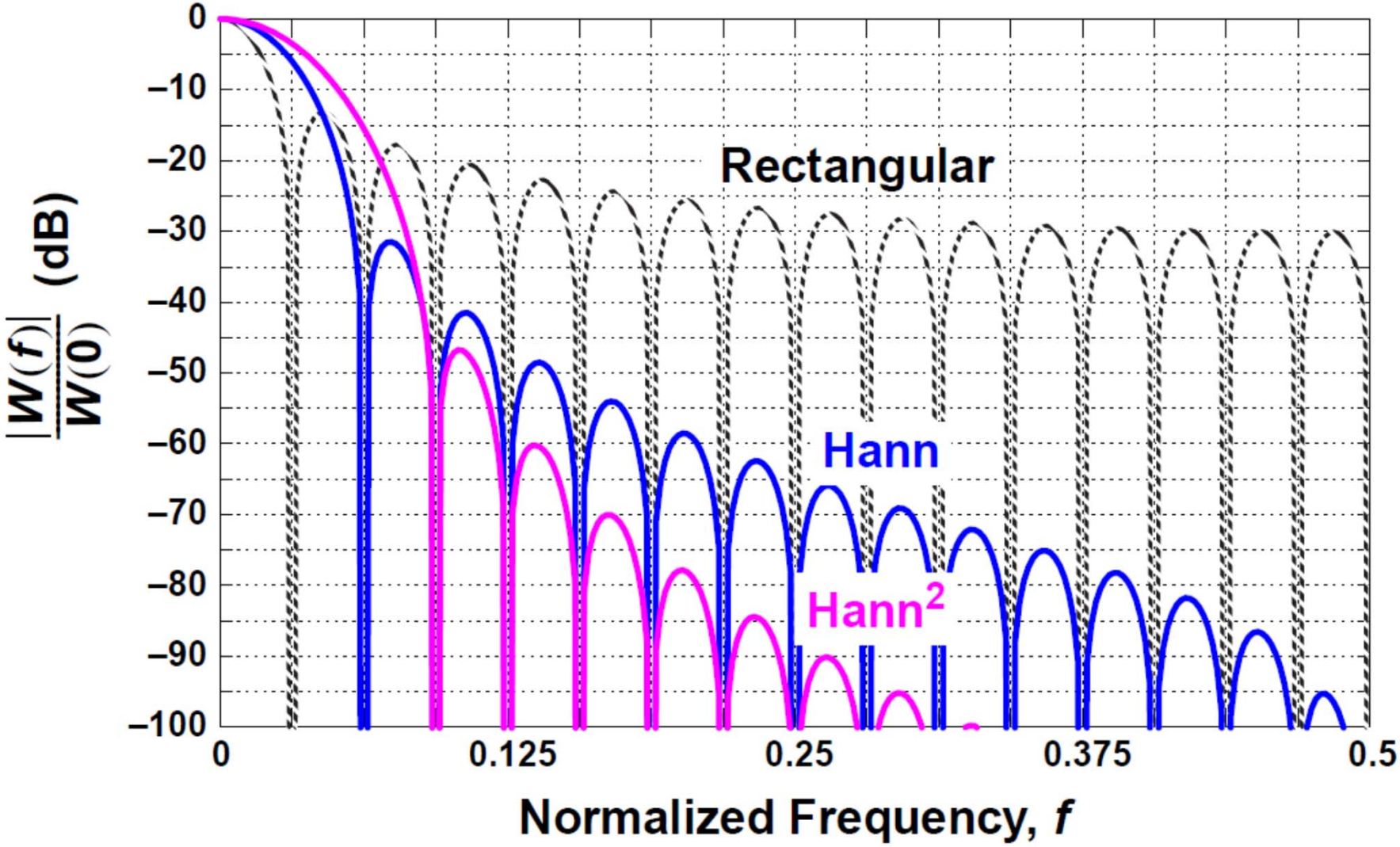


Example Spectral Disaster

- **Rectangular window, N=256**



Window Comparison (N=16)



Window Properties

Window	Rectangular	Hann†	Hann^2
$w(n)$, $n = 0, 1, \ldots, N-1$ ($w(n) = 0$ otherwise)	1	$\frac{1 - \cos\frac{2\pi n}{N}}{2}$	$\left(\frac{1 - \cos\frac{2\pi n}{N}}{2}\right)^2$
Number of non-zero FFT bins	1	3	5
$\|w\|_2^2 = \sum w(n)^2$	N	$3N/8$	$35N/128$
$W(0) = \sum w(n)$	N	$N/2$	$3N/8$
$NBW = \frac{\|w\|_2^2}{W(0)^2}$	$1/N$	$1.5/N$	$35/18N$

†. MATLAB's "hann" function causes spectral leakage of tones located in FFT bins unless you add the optional argument "periodic."

Window Length, *N*

- **Need to have enough in-band noise bins to**
  1. **Make the number of signal bins a small fraction of the total number of in-band bins**

     <20% signal bins → >15 in-band bins → $N > 30 \cdot OSR$
  2. **Make the SNR repeatable**

     $N = 30 \cdot OSR$ **yields std. dev. ~1.4 dB**

     $N = 64 \cdot OSR$ **yields std. dev. ~1.0 dB**

     $N = 256 \cdot OSR$ **yields std. dev. ~0.5 dB**

- $N = 64 \cdot OSR$ **is recommended**

FFT Scaling

- **The FFT implemented in MATLAB is**

$$X_M(k+1) = \sum_{n=0}^{N-1} x_M(n+1) e^{-j\frac{2\pi kn}{N}}$$

- **If** $x(n) = A\sin(2\pi fn/N)$**, then**

$$|X(k)| = \begin{cases} \frac{AN}{2} & k = f \text{ or } N-f \\ 0 & \text{otherwise} \end{cases}$$

➔ **Need to divide FFT by** $(N/2)$ **to get A**

Note: *f* is an integer in $(0, N/2)$**.** $X(k) \equiv X_M(k+1)$**,** $x(n) \equiv x_M(n+1)$ **since Matlab indexes from 1 rather than 0.**

How To Do Smoothing

1. **Average multiple FFTs**

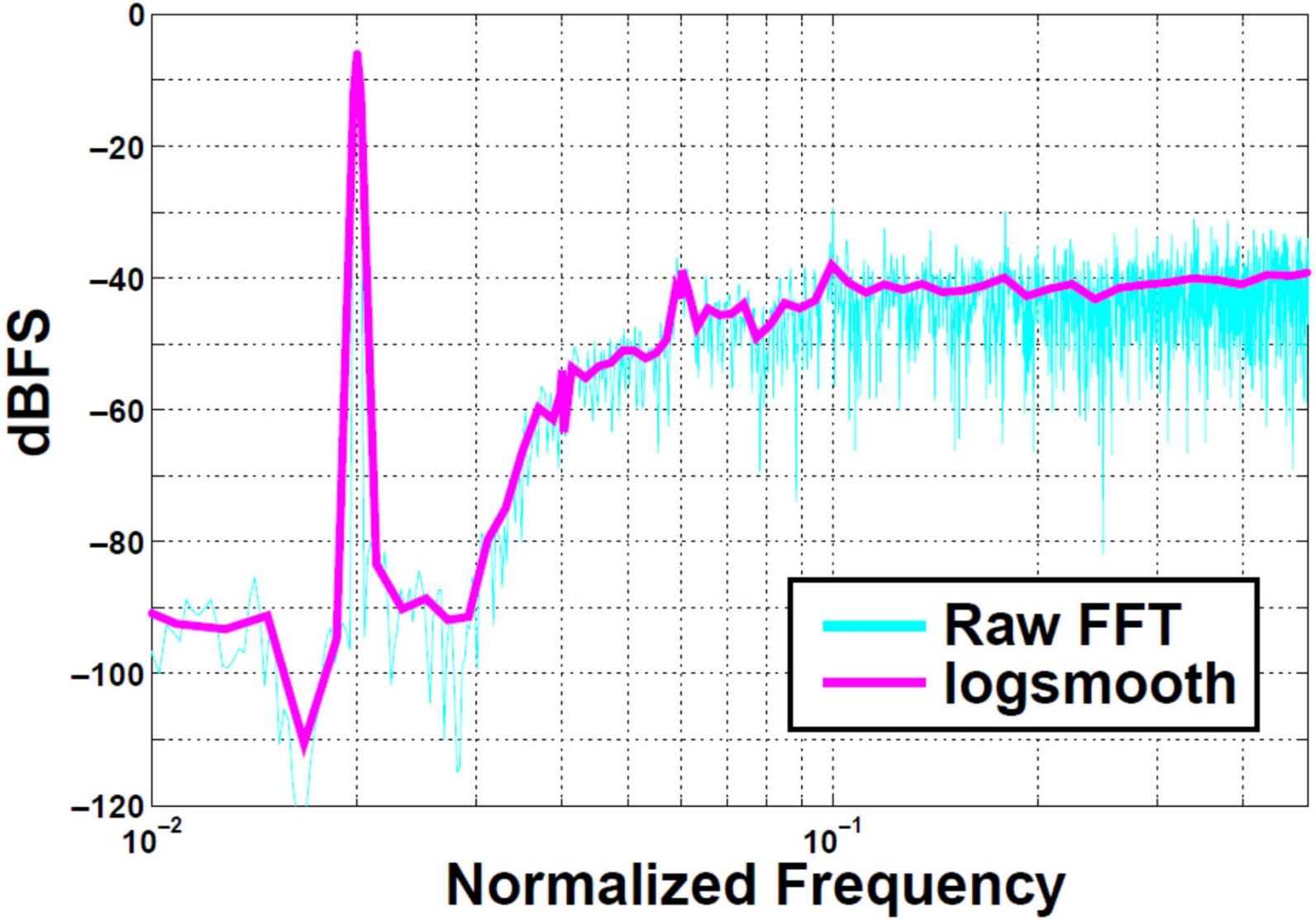
   Implemented by MATLAB's `psd()` function

2. **Take one big FFT and "filter" the spectrum**

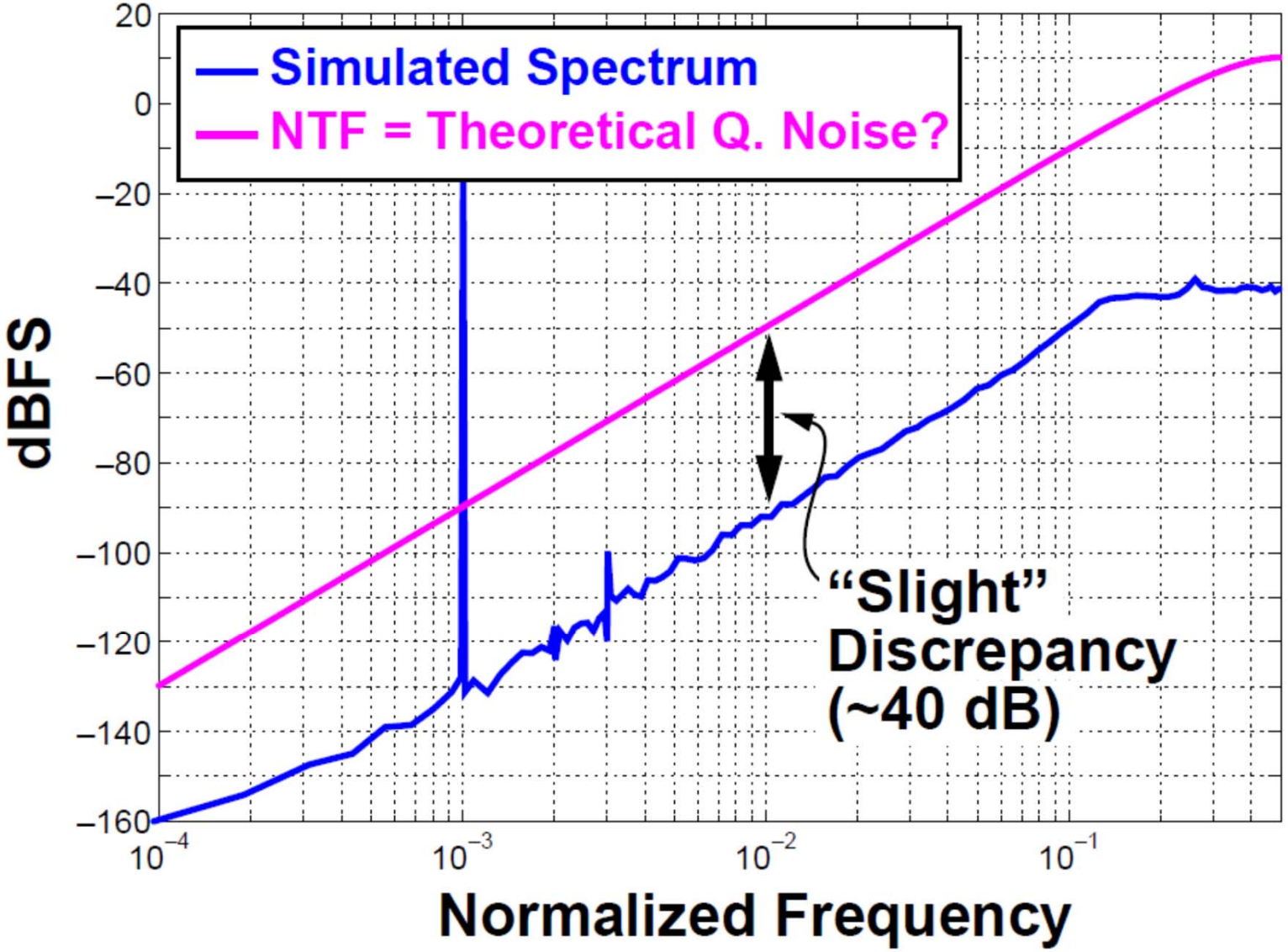
   Implemented by the ΔΣ Toolbox's `logsmooth()` function

   `logsmooth()` averages an exponentially-increasing number of bins in order to reduce the density of points in the high-frequency regime and make a nice log-frequency plot

Raw and Smoothed Spectra



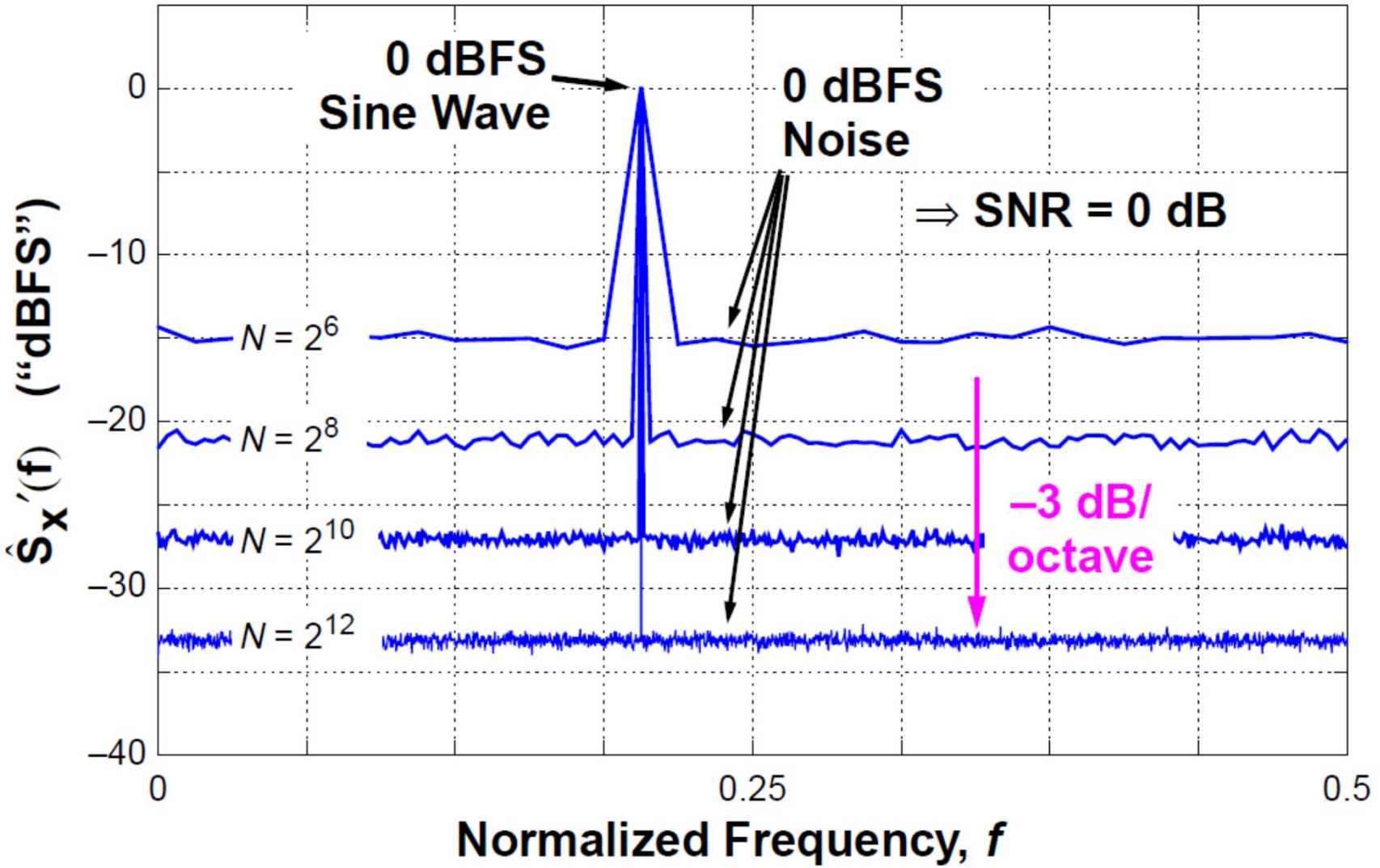
Simulation vs Theory (MOD2)



What Went Wrong?

- We normalized the spectrum so that a full-scale sine wave (which has a power of 0.5) comes out at 0 dB (hence the 'dBFS' units)
  - We do the same for the error signal, use $S_{ee}(f) = 4/3$
  - But this makes the discrepancy 3 dB worse
- We tried to plot a *power spectral density* together with something that we want to interpret as a *power spectrum*
- Sine-wave components are located in individual FFT bins, but broadband signals like noise have their power spread over all FFT bins
  - The 'noise floor' depends on the length of the FFT

Spectrum of a Sine Wave + Noise



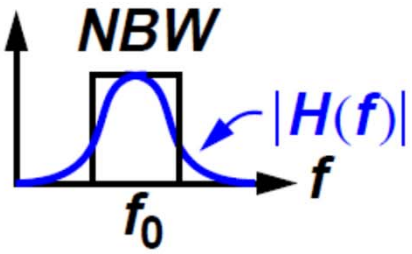
Observations

- **The power of the sine wave is given by the height of its spectral peak**
- **The power of the noise is spread over all bins**
  - **The greater the number of bins, the less power there is in any one bin**
- **Doubling *N* reduces the power per bin by a factor of 2 (i.e. 3 dB)**
  - **But the total integrated noise power does not change**

How Do We Handle Noise?

- An FFT is like a filter bank
- The longer the FFT, the narrower the bandwidth of each filter and thus the lower the power at each output
- We need to know the *noise bandwidth* (NBW) of the filters in order to convert the power in each bin (filter output) to a power density
- For a filter with frequency response *H(f)*,

$$NBW = \frac{\int |H(f)|^2 df}{H(f_0)^2}$$



FFT Noise Bandwidth

- **Alternatively, for $\|h\|_1$ as the L1-norm and $\|h\|_2$ as the L2-norm**

$$NBW = \frac{\int |H(f)|^2 df}{|H(0)|^2} = \left(\frac{\|h\|_2^2}{\|h\|_1^2}\right), f_0 = 0$$

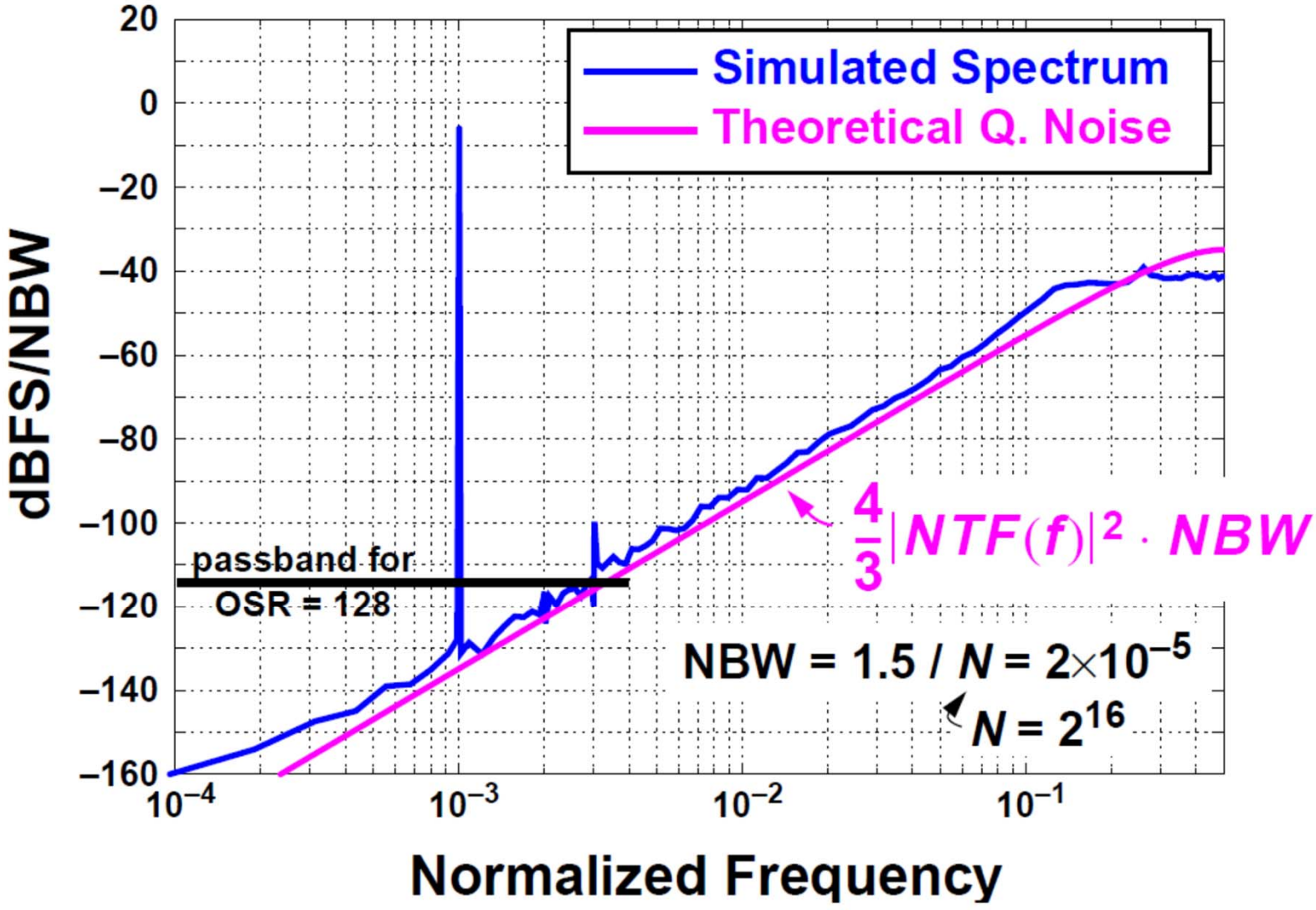
- **Parseval's theorem**

$$\int |H(f)|^2 df = \sum |h(n)|^2$$

- **If $h(n) \geq 0$**

$$|H(0)| = \sum |h(n)|$$

Better Spectral Plot



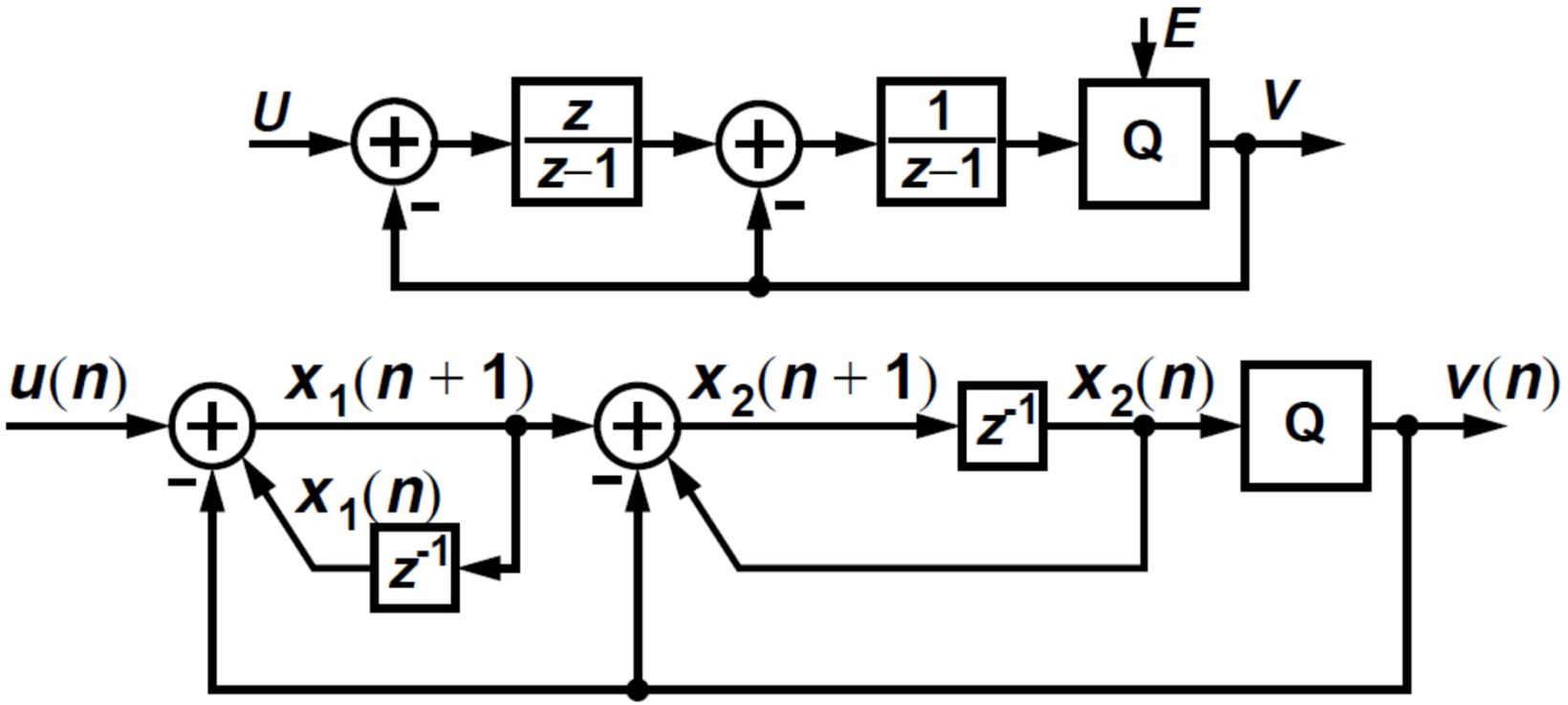
Homework #1 (Due Jan 14*)

A. Create a Matlab function that computes MOD1's output sequence given a vector of input samples and exercise your function in the following ways:

1. **Verify that $\overline{v} \approx u$ for a few random DC inputs in [-1,1]**
2. **Plot the output spectrum with a half-scale sine-wave input. Use good FFT practice. Include the theoretical quantization noise curve and list the theoretical and simulated SQNR for OSR = 128.**

B. Repeat with MOD2

MOD2 Expanded



Difference Equations:

$$v(n) = Q(x_2(n))$$

$$x_1(n+1) = x_1(n) - v(n) + u(n)$$

$$x_2(n+1) = x_2(n) - v(n) + x_1(n+1)$$

Example Matlab Code

```
function [v] = simulateMOD2(u)
      x1 = 0;
      x2 = 0;
      for i = 1:length(u)
            v(i) = quantize(x2);
            x1 = x1 + u(i) – v(i);
            x2 = x2 + x1 – v(i);
      end
return

function v = quantize(y)
      if y>=0
            v = 1;
      else
            v = -1;
      end
return
```

What You Learned Today

- **MOD1: 1st-order ΔΣ modulator**
  - **Structure and theory of operation**
- **Inherent linearity of binary modulators**
- **Inherent anti-aliasing of continuous-time modulators**
- **MOD2: 2nd-order ΔΣ modulator**
- **Good FFT practice**