ECE1371 Advanced Analog Circuits
Lecture 1

INTRODUCTION TO DELTA-SIGMA ADCS

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— a delta-sigma ADC**
- **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.

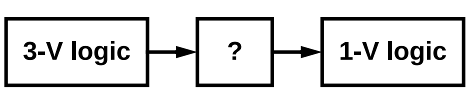
Logistics

- **Format:**
  Meet Thursdays 4:00-6:00 PM except Feb 23
  12 2-hr lectures
  plus proj. presentation
- **Grading:**
  50% homework
  50% project
- **References:**
  Schreier & Temes, "Understanding ΔΣ …"
  Chan-Carusone, Johns & Martin, "Analog IC …"
  Razavi, "Design of Analog CMOS ICs"
  Lecture Plan:

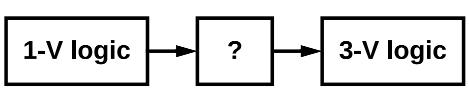
Date			Lecture	Ref	Homework
2012-01-12	RS	1	Introduction: MOD1	ST 2, A	1: MOD1 in Matlab
2012-01-19	RS	2	MOD2 & MOD*N*	ST 3, 4, B	2: MOD2 in Matlab
2012-01-26	RS	3	Example Design: Part 1	ST 9.1, CCJM 14	3: Sw.-level MOD2
2012-02-02	TC	4	SC Circuits	R 12, CCJM 14	4: SC Integrator
2012-02-09	TC	5	Amplifier Design		
2012-02-16	TC	6	Amplifier Design		5: SC Int w/ Amp
2012-02-23	Reading Week + ISSCC– No Lecture				
2012-03-01	RS	7	Example Design: Part 2	CCJM 18	Start Project
2012-03-08	RS	8	Comparator & Flash ADC	CCJM 10	
2012-03-15	TC	9	Noise in SC Circuits	ST C	
2012-03-22	TC	10	Matching & MM-Shaping	ST 6.3-6.5, +	
2012-03-29	RS	11	Advanced ΔΣ	ST 6.6, 9.4	
2012-04-05	TC	12	Pipeline and SAR ADCs	CCJM 15, 17	
2012-04-12	No Lecture				
2012-04-19	Project Presentation				

NLCOTD: Level Translator

VDD1 > VDD2, e.g.



- **VDD1 < VDD2, e.g.**



- **Constraints: CMOS**
  1-V and 3-V devices
  no static current

Highlights
(i.e. What you will learn today)

1. **1st-order modulator structure and theory of operation**
2. **Inherent linearity of binary modulators**
3. **Inherent anti-aliasing of continuous-time modulators**
4. **How to plot ΔΣ spectra**

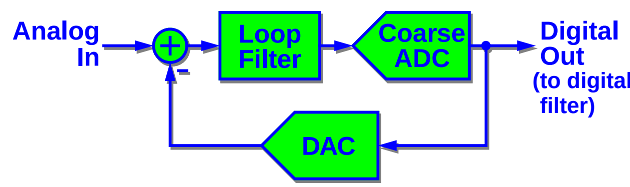
Background
(Stuff you already know)

- The 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
  - i.e. X → H(z) → Y $S_{yy}(f) = |H(e^{j2\pi f})|^2 \cdot S_{xx}(f)$
- The power in any frequency band is the integral of the PSD over that band

*. SQNR = Signal-to-Quantization-Noise Ratio

What is ΔΣ?

- ΔΣ is NOT a fraternity
- Simplified ΔΣ ADC structure:



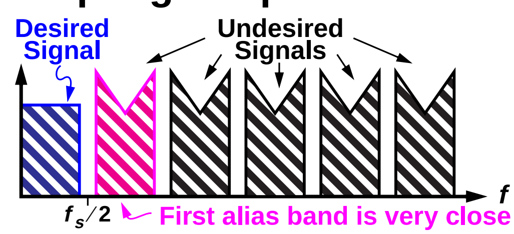
- Key features: coarse quantization, filtering, feedback and oversampling
  - Quantization is often *quite* coarse (1 bit!), but the effective resolution can still be as high as 22 bits.

What is Oversampling?

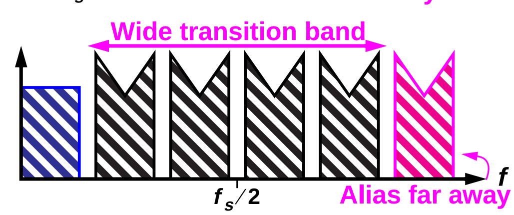
- Oversampling is sampling faster than required by the Nyquist criterion
  - For a lowpass signal containing energy in the frequency range $(0, f_B)$, the minimum sample rate required for perfect reconstruction is $f_s = 2f_B$
- The *oversampling ratio* is $OSR \equiv f_s/(2f_B)$
- For a regular ADC, $OSR \sim 2-3$
  - To make the anti-alias filter (AAF) feasible
- For a ΔΣ ADC, $OSR \sim 30$
  - To get adequate quantization noise suppression. Signals between f_B and $\sim f_s$ are removed digitally.

Oversampling Simplifies AAF

OSR ~ 1:

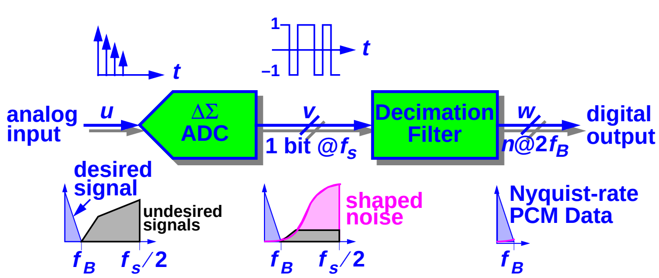


OSR = 3:

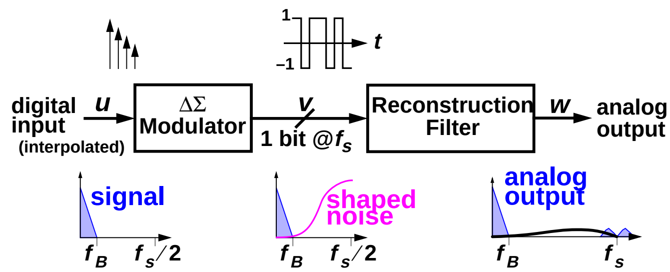


How Does A ΔΣ ADC Work?

- Coarse 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 that the out-of-band noise is handled by an analog reconstruction filter.

Why Do It The ΔΣ Way?

- ADC: Simplified Anti-Alias Filter

  Since the input is oversampled, only very high frequencies alias to the passband.
  A simple RC section often suffices.

  If a continuous-time loop filter is used, the anti-alias filter can often be eliminated altogether.

- DAC: Simplified Reconstruction Filter

  The nearby images present in Nyquist-rate reconstruction can be removed digitally.

+ Inherent Linearity

  Simple structures can yield very high SNR.

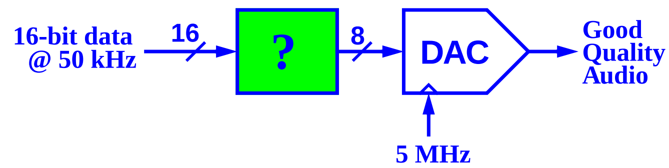
+ Robust Implementation

  ΔΣ tolerates sizable component errors.

Poor Man's ΔΣ DAC
[Not in Schreier & Temes]

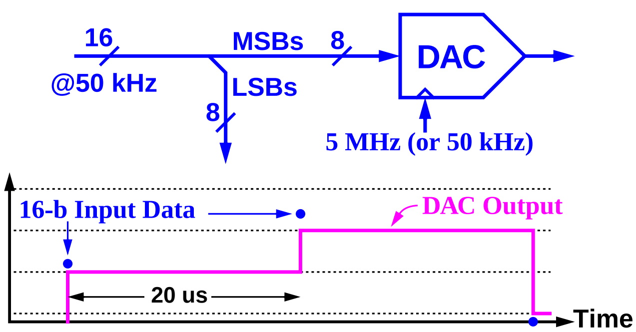
Suppose you have low-speed 16-bit data and a high-speed 8-bit DAC

- How can you get good analog performance?

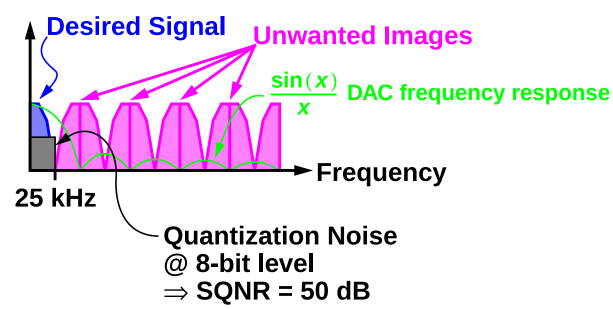


Simple (-Minded) Solution

- Connect only the MSBs; leave the LSBs hanging

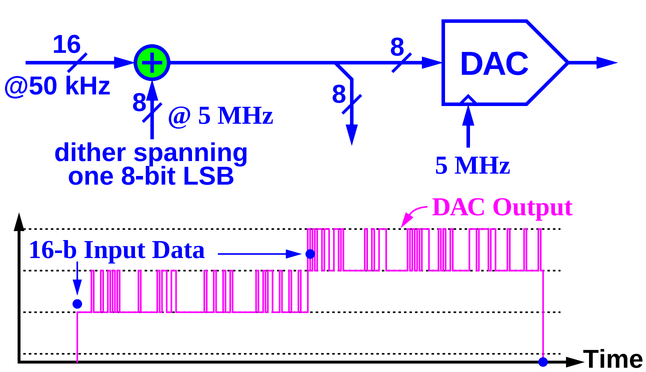


Spectral Implications



Better Solution

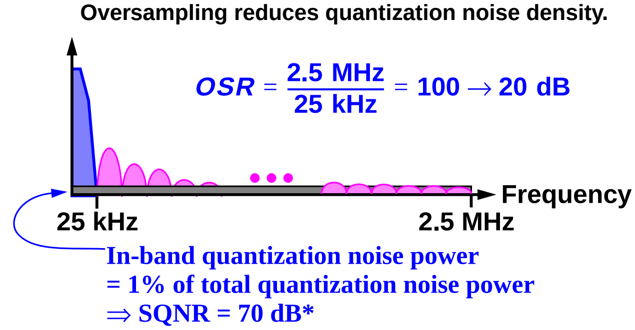
- Exploit oversampling: Clock fast and add dither



Spectral Implications

- Quantization noise is now spread over a broad frequency range

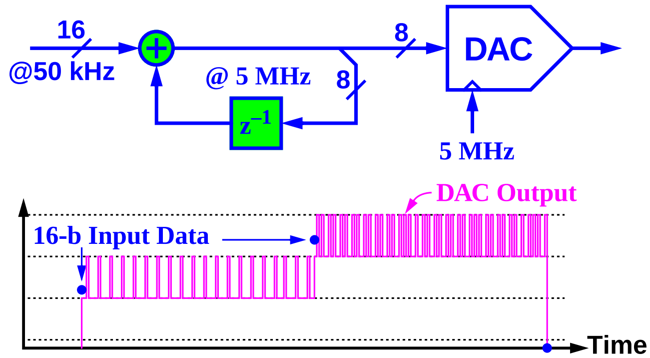
  Oversampling reduces quantization noise density.



*Actual SNR = 67 dB because dither power = quantization noise power

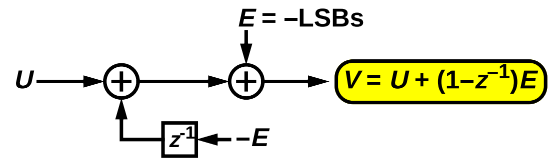
Even More Clever Method

- Add LSBs back into the input data



Mathematical Model

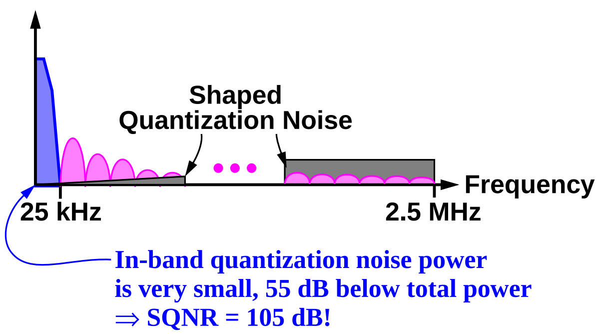
- Assume the DAC is ideal, model truncation as the addition of error:



- Hmm… Oversampling, coarse quantization and feedback. Noise-Shaping!
- Truncation noise is shaped by a $1–z^{–1}$ transfer function, which provides ~35 dB of attenuation in the 0-25 kHz frequency range

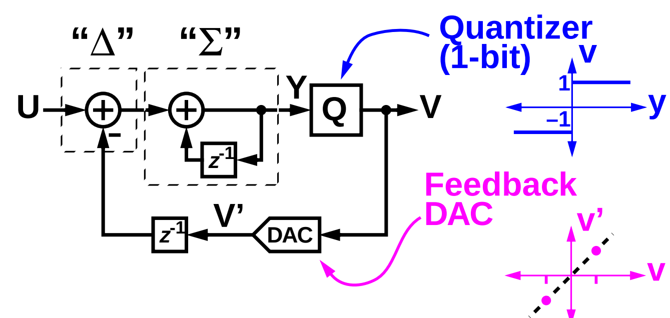
Spectral Implications

- Quantization noise is heavily attenuated at low frequencies



MOD1: 1st-Order ΔΣ Modulator

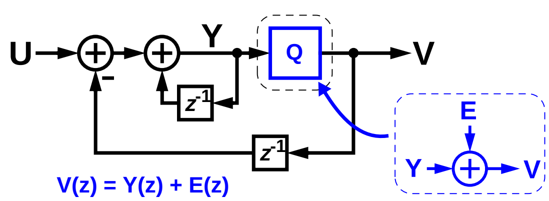
[Ch. 2 of Schreier & Temes]



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:



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

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

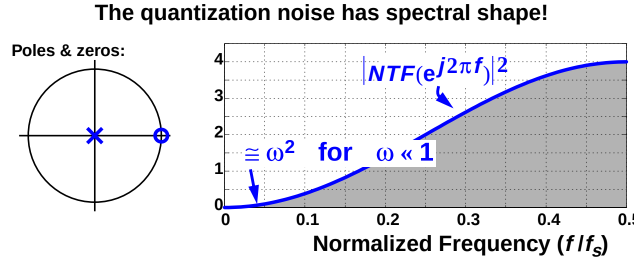
$\Rightarrow (1–z^{-1})\,V(z) = U(z) – z^{-1}V(z) + (1–z^{-1})E(z)$

$$V(z) = U(z) + (1–z^{–1})E(z)$$

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!



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

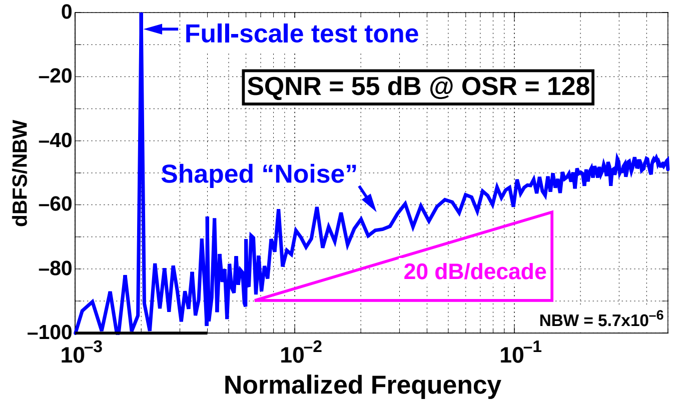
In-band Quant. Noise Power

- Assume that e is white with power σ_e^2
  i.e. $S_{ee}(\omega) = \sigma_e^2/\pi$
- The in-band quantization noise power is

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

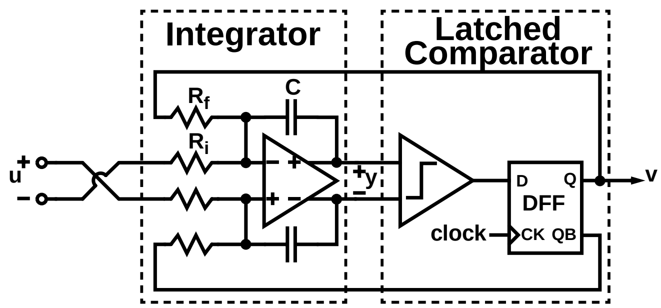
- Since $OSR \equiv \frac{\pi}{\omega_B}$, $IQNP = \frac{\pi^2 \sigma_e^2}{3} (OSR)^{-3}$
- For MOD1, an octave increase in *OSR* increases SQNR by 9 dB
  "1.5-bit/octave SQNR-OSR trade-off."

A Simulation of MOD1

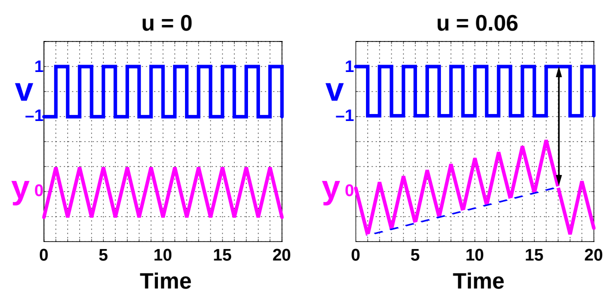


CT Implementation of MOD1

- R_i/R_f sets the full-scale; C is arbitrary
  Also observe that an input at f_s is rejected by the integrator— *inherent anti-aliasing*



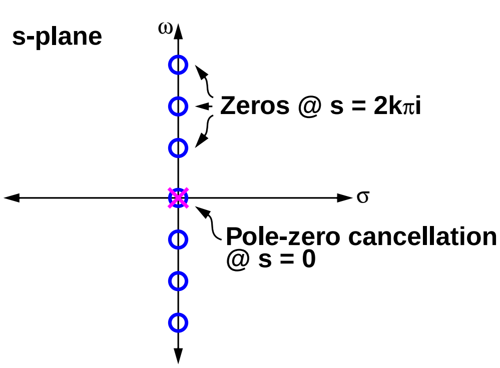
MOD1-CT Waveforms



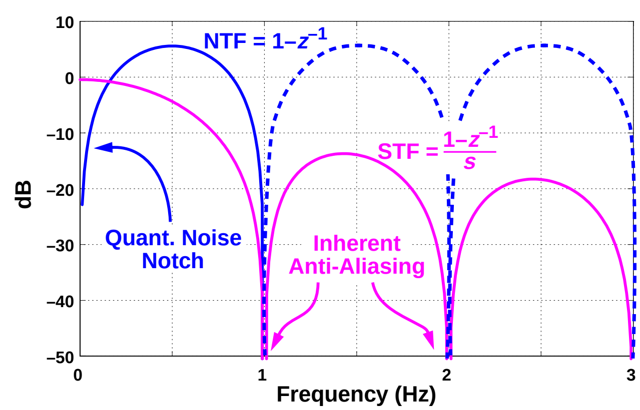
- With u=0, v alternates between +1 and –1
- With u>0, y drifts upwards; v contains consecutive +1s to counteract this drift

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

Recall $z = e^s$



MOD1-CT Frequency Responses

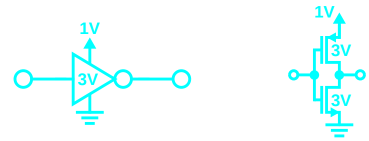


Summary

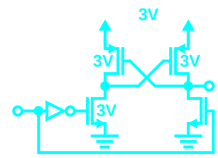
- $\Delta\Sigma$ works by spectrally separating the quantization noise from the signal
  Requires oversampling. $OSR \equiv f_s / (2 f_B)$.
- Noise-shaping is achieved by the use of *filtering* and *feedback*
- A binary DAC is *inherently linear,* and thus a binary $\Delta\Sigma$ modulator is too
- MOD1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

NLCOTD

3V $\rightarrow$ 1V:



1V $\rightarrow$ 3V:



Homework #1

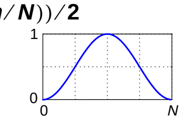
Create a Matlab function that computes MOD1's output sequence given a vector of input samples and exercise your function in the following ways.

1. Verify that the average of the output equals the input for a few DC inputs in [–1,1]. Approximately how many samples need to be averaged in order to guarantee the error is less than 0.001?
2. Plot the error $\bar{v} - u$ as a function of u using a 100-point rectangular average for $\bar{v}$.
3. Verify that with a sine-wave input the output consists of a sine wave plus shaped noise. Use good FFT practice and include the expected noise curve.

Good FFT Practice

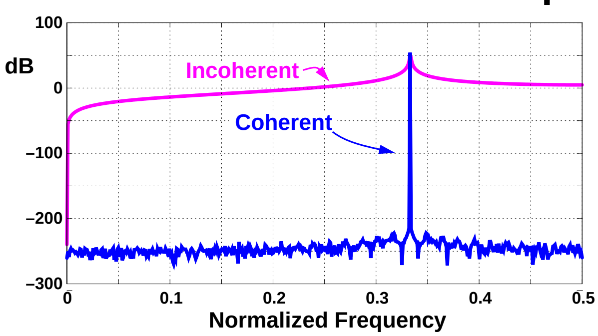
[Appendix A of Schreier & Temes]

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



- 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

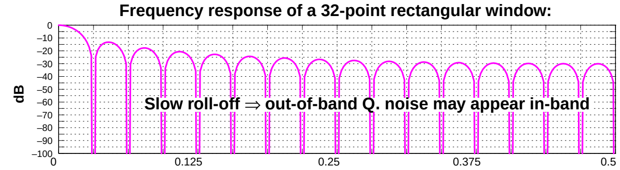


- Coherent sampling: only one non-zero FFT bin
- Incoherent sampling: "spectral leakage"

Windowing

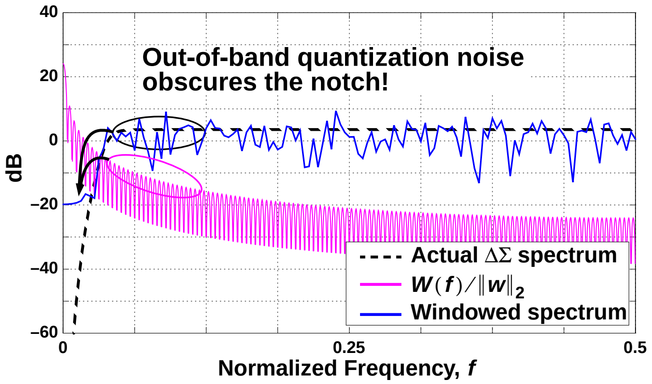
- $\Delta\Sigma$ data is usually not periodic
  Just because the input repeats does not mean that the output does too!
- A finite-length data record = an infinite record multiplied by a *rectangular window*:
  $w(n) = 1, 0 \le n < N$
  Windowing is unavoidable.
- "Multiplication in time is convolution in frequency"

Frequency response of a 32-point rectangular window:

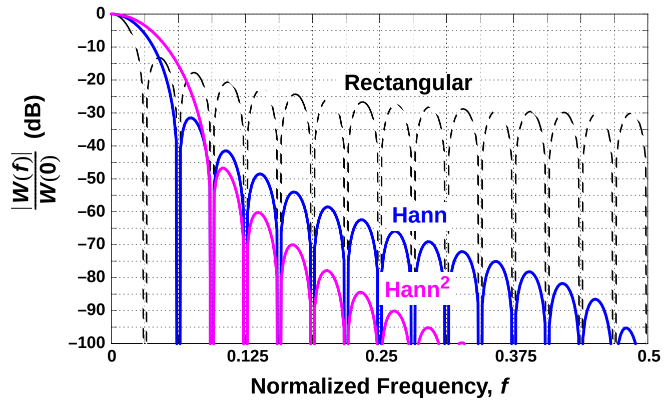


Example Spectral Disaster

Rectangular window, $N = 256$



Window Comparison ($N = 16$)



Window Properties

Window	Rectangular	Hann†	Hann²
$w(n)$, $n = 0, 1, \ldots, N-1$ ($w(n) = 0$ otherwise)	1	$\dfrac{1 - \cos\frac{2\pi n}{N}}{2}$	$\left(\dfrac{1 - \cos\frac{2\pi n}{N}}{2}\right)^2$
Number of non-zero FFT bins	1	3	5
$\Vert w\Vert_2^2 = \sum w(n)^2$	N	$3N/8$	$35N/128$
$W(0) = \sum w(n)$	N	$N/2$	$3N/8$
$NBW = \dfrac{\Vert w\Vert_2^2}{W(0)^2}$	$1/N$	$1.5/N$	$35/18N$

†. 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

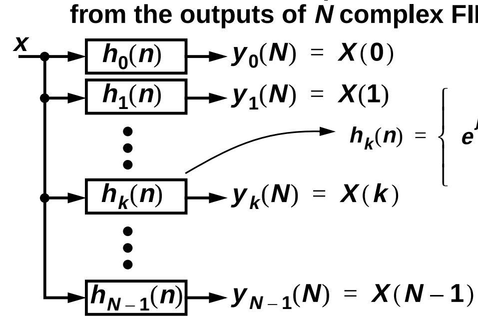
$$|X(k)| = \begin{cases} \dfrac{AN}{2} & , k = f \text{ or } N-f \\ 0 & , \text{otherwise} \end{cases}$$

⇒ Need to divide FFT by $(N/2)$ to get A.

†. f is an integer in $(0, N/2)$. I've defined $X(k) \equiv X_M(k+1)$, $x(n) \equiv x_M(n+1)$ since Matlab indexes from 1 rather than 0.

The Need For Smoothing

- The FFT can be interpreted as taking 1 sample from the outputs of N complex FIR filters:



$$h_k(n) = \begin{cases} e^{j\frac{2\pi k}{N}n} & , 0 \le n < N \\ 0 & , \text{otherwise} \end{cases}$$

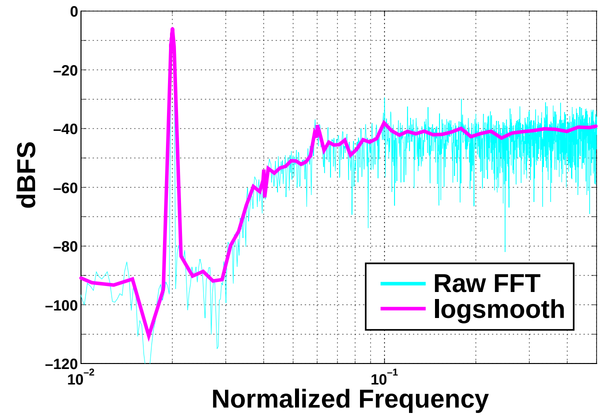
⇒ an FFT yields a high-variance spectral estimate

How To Do Smoothing

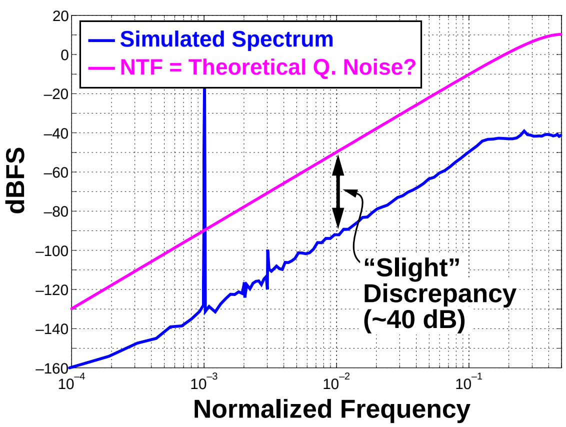
1. **Average multiple FFTs**
   Implemented by MATLAB's `psd()` function
2. **Take one big FFT and "filter" the spectrum**
   Implemented by the ΔΣ Toolbox's `logsmooth()` function

- **`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?

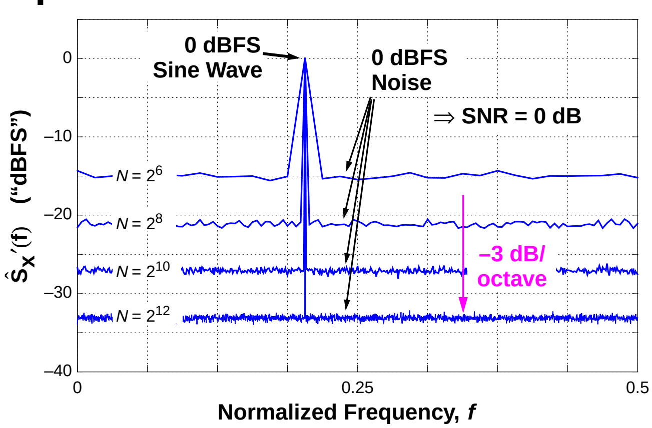
1. **We normalized the spectrum so that a full-scale sine wave (which has a power of 0.5) comes out at 0 dB (whence the "dBFS" units)**

   ⇒ **We need to do the same for the error signal.**
   i.e. use $S_{ee}(f) = 4/3$.

   But this makes the discrepancy 3 dB worse.
2. **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



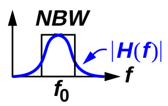
Observations

- **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.

So How Do We Handle Noise?

- Recall that 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

Rectangular Window

$$h_k(n) = \exp\left(j\frac{2\pi k}{N}n\right),\ H_k(f) = \sum_{n=0}^{N-1} h_k(n)\exp(-j2\pi fn)$$

$$f_0 = \frac{k}{N},\ H_k(f_0) = \sum_{n=0}^{N-1} 1 = N$$

$$\int |H_k(f)|^2 = \sum |h_k(n)|^2 = N \text{ [Parseval]}$$

$$\therefore NBW = \frac{\int |H_k(f)|^2 df}{H_k(f_0)^2} = \frac{N}{N^2} = \frac{1}{N}$$

Better Spectral Plot