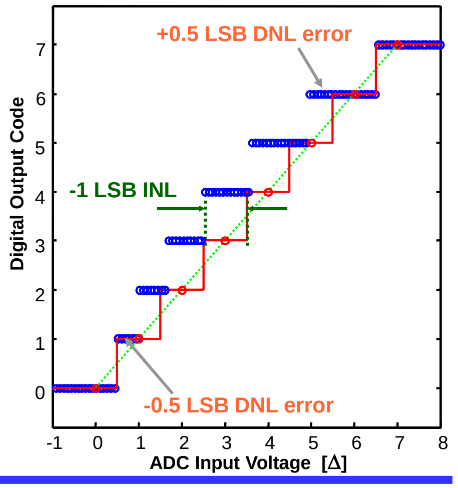
EE247
Lecture 12

- Data Converters
  - Data converter testing (continued)
    - Measuring DNL & INL
      - Servo-loop
      - Code density testing (histogram testing)
    - Dynamic tests
      - Spectral testing→ Reveals ADC errors associated with dynamic behavior i.e. ADC performance as a function of frequency
        - Direct Discrete Fourier Transform (DFT) based measurements utilizing sinusoidal signals
        - DFT measurements including windowing
    - Relationship between: DNL & SNR, INL & SFDR
    - Effective number of bits (ENOB)

Summary
ADC Differential Nonlinearity & Integral Nonlinearity
End-Point

1. Endpoints connected
2. Ideal characteristics derived eliminating offset & full-scale error (same as for DNL)
3. DNL → deviation of code width from Δ (1LSB)
4. INL→ deviation of code transition from ideal

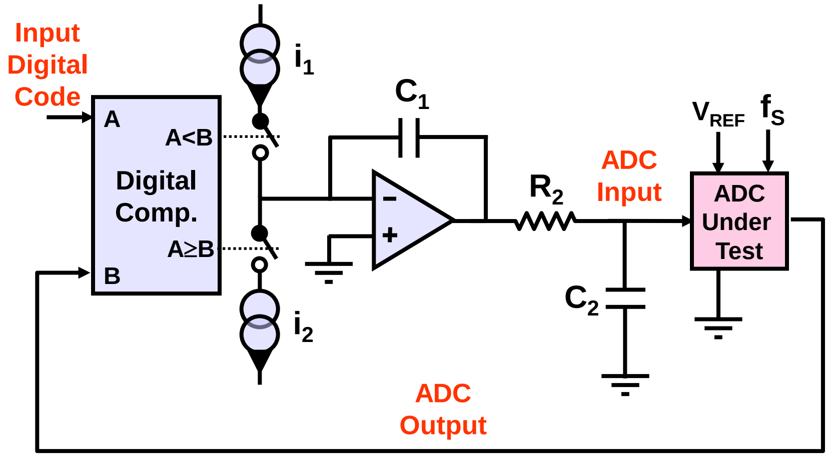


How to measure DNL/INL?

- DAC:
  - Simply apply digital codes and use a good voltmeter to measure corresponding analog output
- ADC
  - Not as simple as DAC→ need to find "decision levels", i.e. input voltages at all code boundaries
    - One way: Adjust voltage source to find exact code trip points "code boundary servo"
    - More versatile: Histogram testing

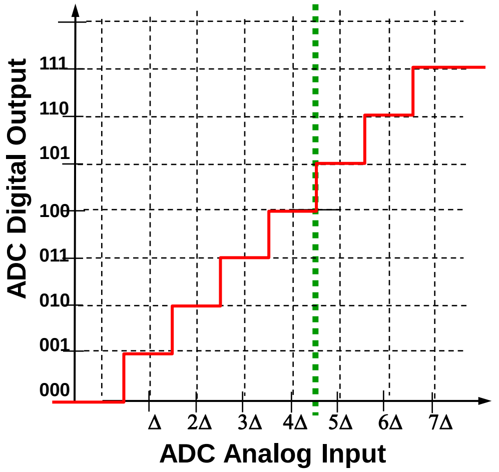
      →Apply a signal with known amplitude distribution and analyze digital code distribution at ADC output

Code Boundary Servo

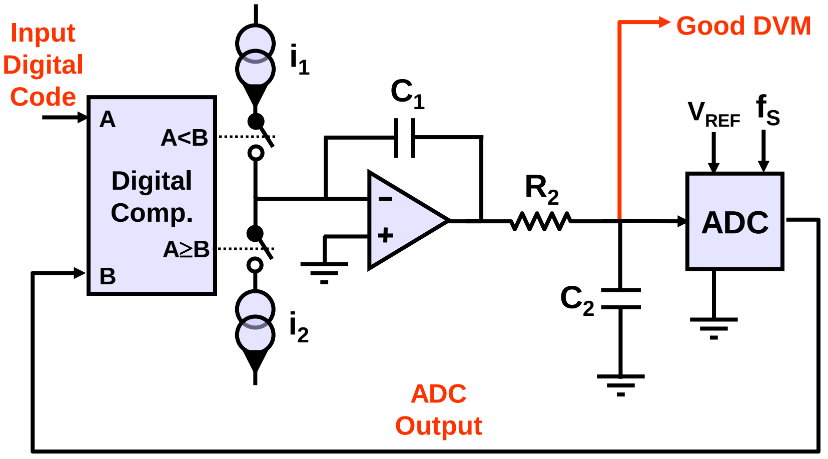


Code Boundary Servo

- i1 and i2 are small, and C1 is large ($\Delta V = it/C1$), so the ADC analog input moves a small fraction of an LSB (e.g. 0.1LSB) each sampling period
- For a code input of 101, the ADC analog input settles to the code boundary shown



Code Boundary Servo



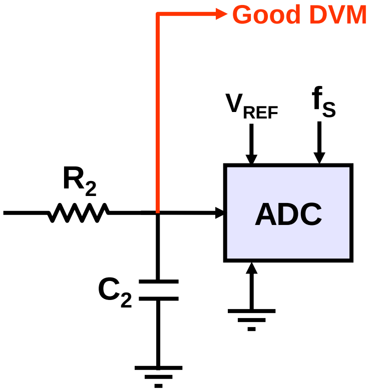
Code Boundary Servo

- A very good digital voltmeter (DVM) measures the analog input voltage corresponding to the desired code boundary
- DVMs have some interesting properties
  - They can have very high resolutions (8½ decimal digit meters are inexpensive)
  - To achieve stable readings, DVMs average voltage measurements over multiple 60Hz ac line cycles to filter out pickup in the measurement loop

Code Boundary Servo

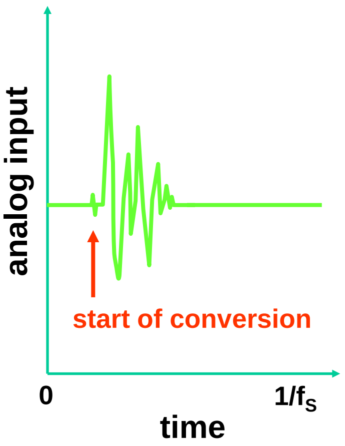
- ADCs of all kinds are notorious for kicking back high-frequency, signal-dependent glitches to their analog inputs

- A magnified view of an analog input glitch follows …



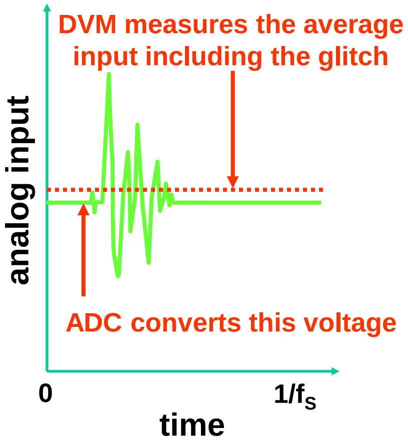
Code Boundary Servo

- Just before the input is sampled and conversion starts, the analog input is pretty quiet
- As the converter begins to quantize the signal, it kicks back charge



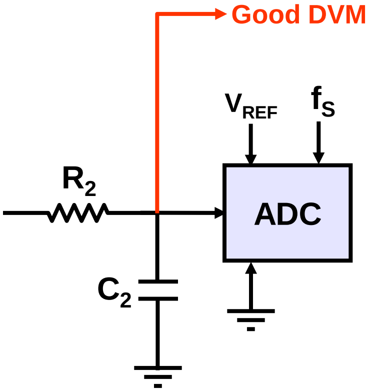
Code Boundary Servo

- The difference between what the ADC measures and what the DVM measures is not ADC INL, it's error in the INL measurement
- How do we control this error?



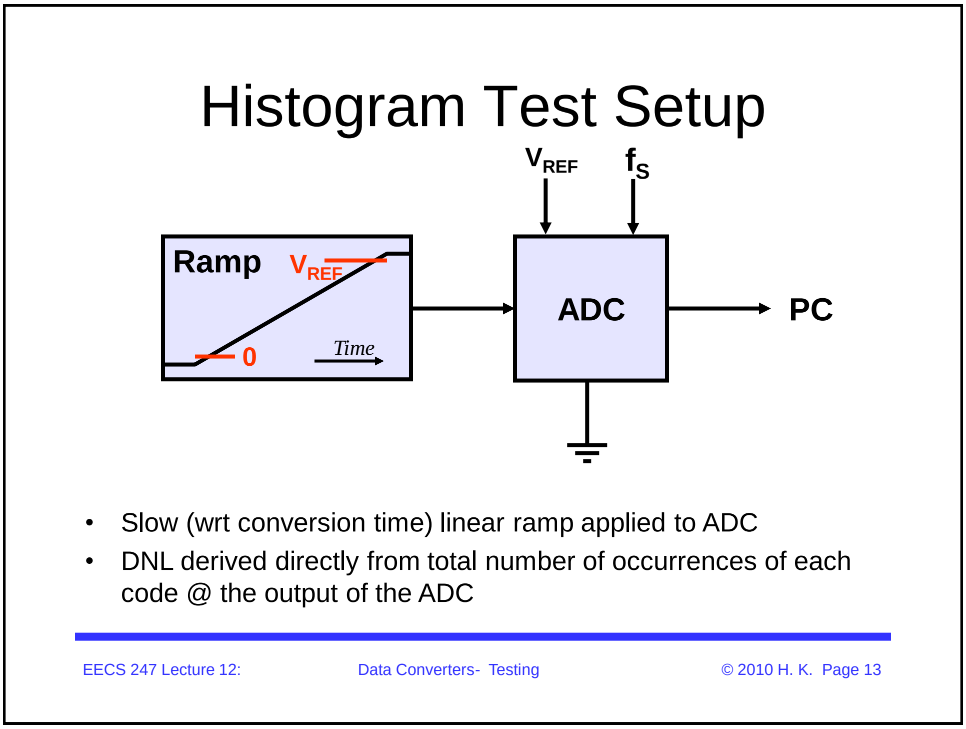
Code Boundary Servo

- A large C_2 reduces the effect of kick-back
- At the expense of longer measurement time



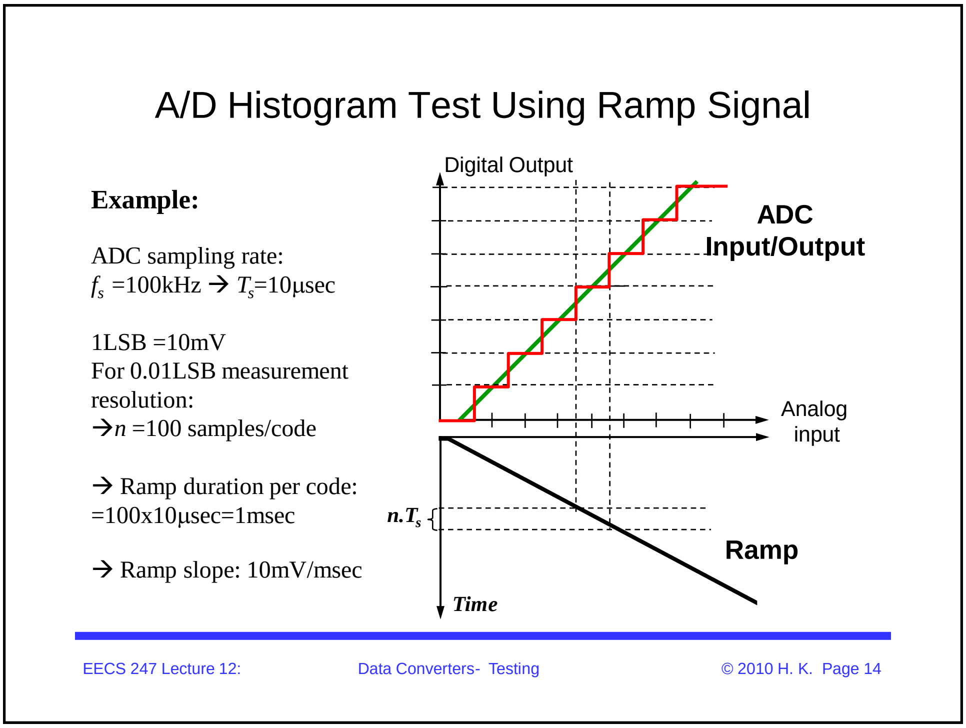
Histogram Testing

- Code boundary measurements are slow
  - Long testing time
- Histogram testing
  - Apply input with known pdf (e.g. ramp or sinusoid) & quantize
  - Measure output pdf
  - Derive INL and DNL from deviation of measured pdf from expected result



Histogram Test Setup

- Slow (wrt conversion time) linear ramp applied to ADC
- DNL derived directly from total number of occurrences of each code @ the output of the ADC



A/D Histogram Test Using Ramp Signal

Example:

ADC sampling rate:
f_s =100kHz → T_s=10µsec

1LSB =10mV
For 0.01LSB measurement resolution:
→n =100 samples/code

→ Ramp duration per code:
=100x10µsec=1msec

→ Ramp slope: 10mV/msec

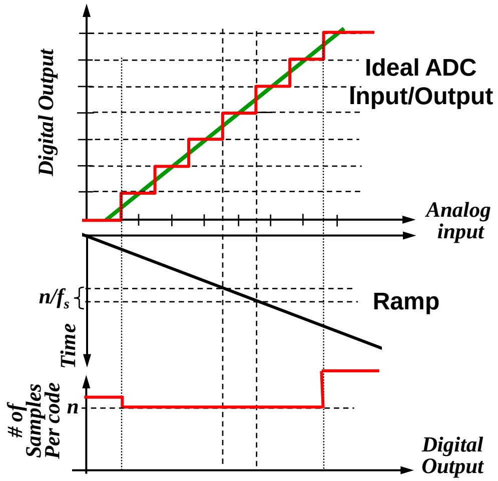
A/D Histogram Test Using Ramp Signal

Example:

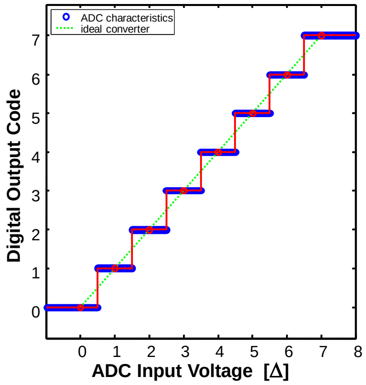
Ramp slope: 10mV/msec
1LSB =10mV
Each ADC code→1msec

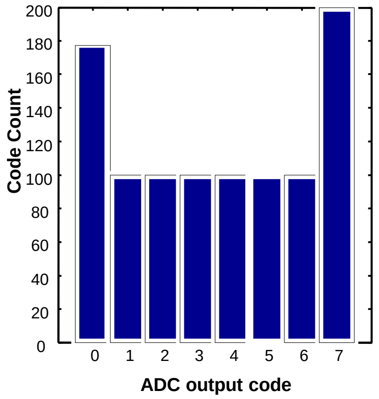
f_s =100kHz → T_s=10μsec

→n =100 samples/code

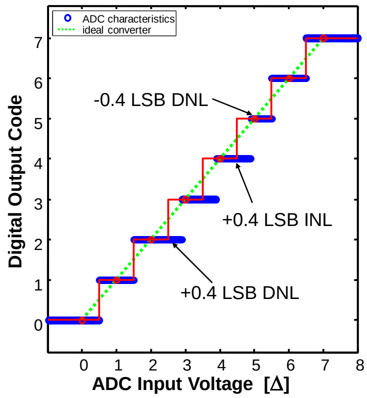


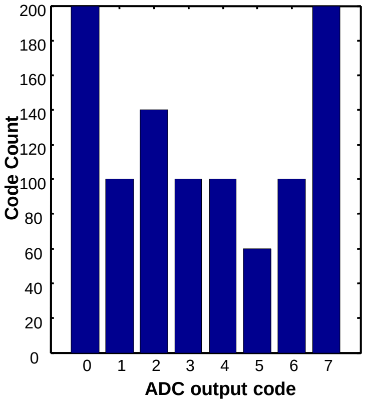
Ramp Histogram
Example: Ideal 3-Bit ADC





Ramp Histogram
Example: Real 3-Bit ADC Including Non-Idealities

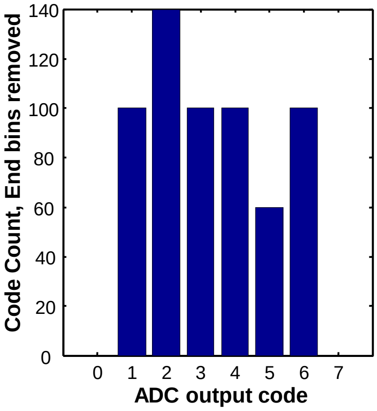




Example: 3 Bit ADC
DNL Extracted from Histogram

1- Remove “Over-range bins” (0 and full-scale)

2- Compute average count/bin (600/6=100 in this case)



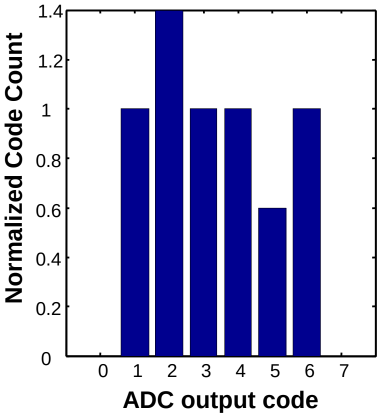
Example: 3 Bit ADC
Process of Extracting from Histogram

3- Normalize:

- Divide histogram by average count/bin

→ ideal bins have exactly the average count, which, after normalization, would be 1

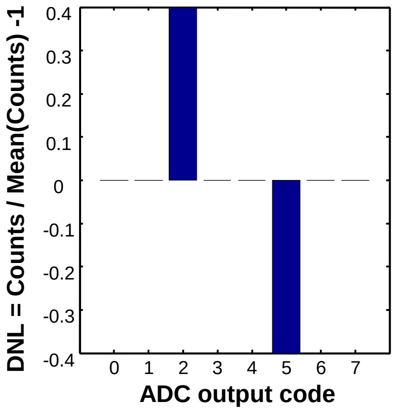
→ Non-ideal bins would have a normalized value greater or smaller than 1



Example: 3 Bit ADC
DNL Extracted from Histogram

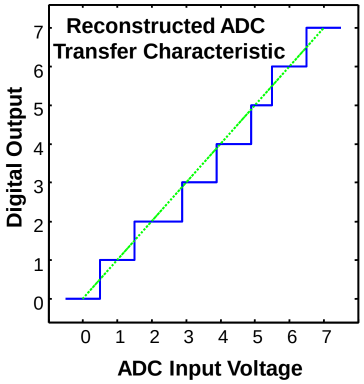
4- Subtract ***1*** from the normalized code count

5- Result → DNL (+-0.4LSB in this case)

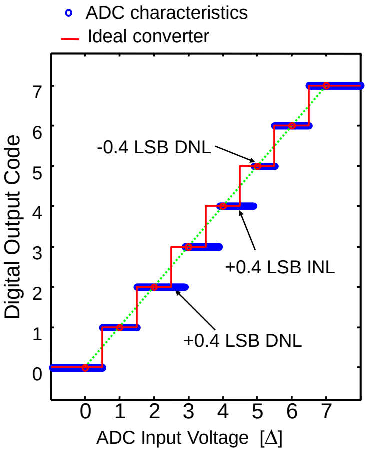


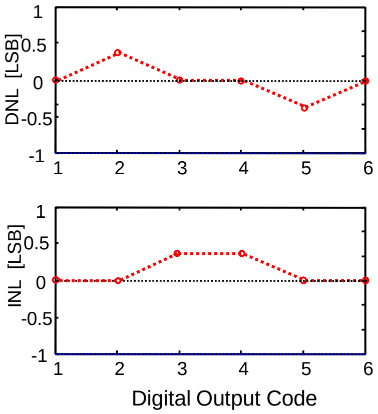
Example: 3-Bit ADC
Static Characteristics Extracted from Histogram

- DNL histogram → used to reconstruct the exact converter characteristic (having measured only the histogram)
- Width of all codes derived from measured DNL (Code=DNL + 1LSB)
- INL→(deviation from a straight line through the end points)- is found



Example: 3 Bit ADC
DNL & INL Extracted from Histogram





Measuring DNL

- Ramp speed is adjusted to provide large number of output/code - e.g. an average of 100 outputs of each ADC code (for 1/100 LSB resolution)

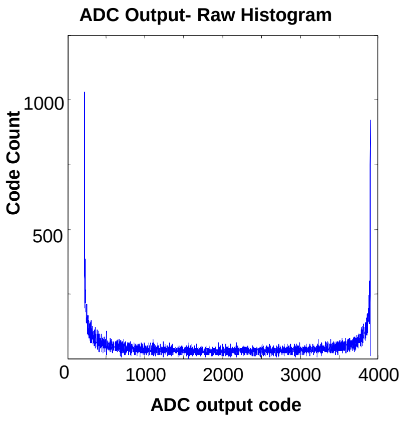
- Ramp test can be quite slow for high resolution ADCs
- Example:

  16bit ADC & 100conversions/code @100kHz sampling rate

$$\frac{\textbf{(2}^{16}\textbf{or 65,536 codes)(100 conversions/code)}}{\textbf{100,000 conversions/sec}} \textbf{ = 65.6 sec}$$

ADC Histogram Testing
Sinusoidal Inputs

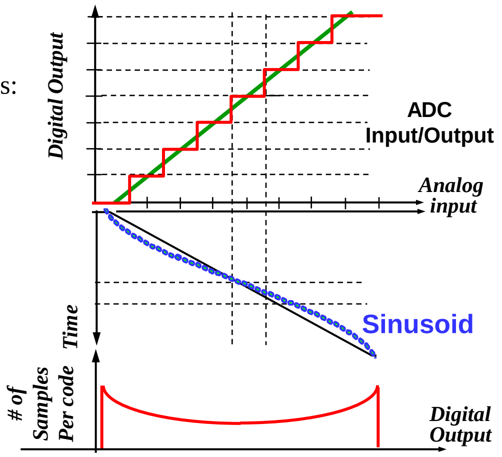
- Ramp signal generators linear to only 8 to10bits & thus only good for testing ADCs <10bit res.
  - → Need to find input signal with better purity for testing higher res. ADCs
- Solution:
  - →Use sinusoidal test signal (may need to filter out harmonics)
- Problem: Ideal ADC histogram not flat but has “bath-tub shape”



ADC Histogram Test Using Sinusoidal Signals

At sinusoid midpoint crossings:
dv/dt → max.
→ least # of samples

At sinusoid amplitude peaks:
dv/dt → min.
→ highest # of samples



Histogram Testing
Correction for Sinusoidal PDF

- Is it necessary to know the exact amplitude and offset of sinusoidal input? No!

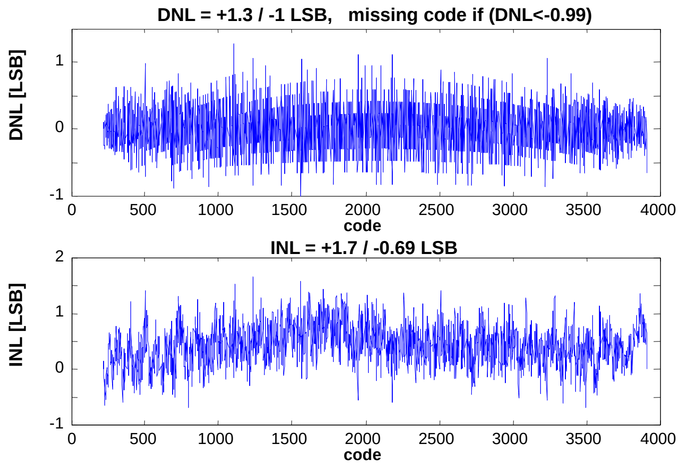
- References:
  - [1] M. V. Bossche, J. Schoukens, and J. Renneboog, “Dynamic Testing and Diagnostics of A/D Converters,” IEEE Transactions on Circuits and Systems, vol. CAS-33, no. 8, Aug. 1986.
  - [2] IEEE Standard 1057

DNL/INL Extraction Matlab Program Sinusoidal Histogram

```
function [dnl,inl] = dnl_inl_sin(y);
%DNL_INL_SIN
% dnl and inl ADC output
% input y contains the ADC output
% vector obtained from quantizing a
% sinusoid

% Boris Murmann, Aug 2002
% Bernhard Boser, Sept 2002

% histogram boundaries
minbin=min(y);
maxbin=max(y);

% histogram
h = hist(y, minbin:maxbin);

% cumulative histogram
ch = cumsum(h);
```

```
% transition levels found by:
T = -cos(pi*ch/sum(h));

% linearized histogram
hlin = T(2:end) - T(1:end-1);

% truncate at least first and last
% bin, more if input did not clip ADC
trunc=2;
hlin_trunc = hlin(1+trunc:end-trunc);

% calculate lsb size and dnl
lsb= sum(hlin_trunc) / (length(hlin_trunc));
dnl= [0 hlin_trunc/lsb-1];
misscodes = length(find(dnl<-0.99));

% calculate inl
inl= cumsum(dnl);
```

Example: Test Results for DNL & INL Using Sinusoidal Histogram

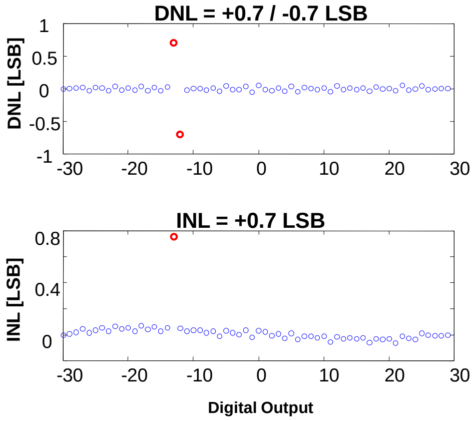


DNL = +1.3 / -1 LSB, missing code if (DNL<-0.99)

INL = +1.7 / -0.69 LSB

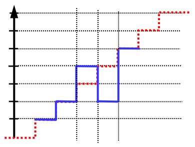
Example: Matlab ADC Model
DNL/INL Code Test

```
% converter model
B = 6;                 % bits
range = 2^(B-1) - 1;
% thresholds (ideal converter)
th = -range:range; % ideal thresholds
th(20) = th(20)+0.7; % error

fs = 1e6;
fx = 494e3 + pi;       % try fs/10!
C  = round(100 * 2^B / (fs / fx));

t = 0:1/fs:C/fx;
x = (range+1) * sin(2*pi*fx.*t);
y = adc(x, th) - 2^(B-1);

hist(y, min(y):max(y));

dnl_inl_sin(y);
```



Histogram Testing Limitations

- The histogram (as any ADC test, of course) characterizes one particular converter. Test many devices to get valid statistics.
- Histogram testing assumes monotonicity
  E.g. “code flips” will not be detected.



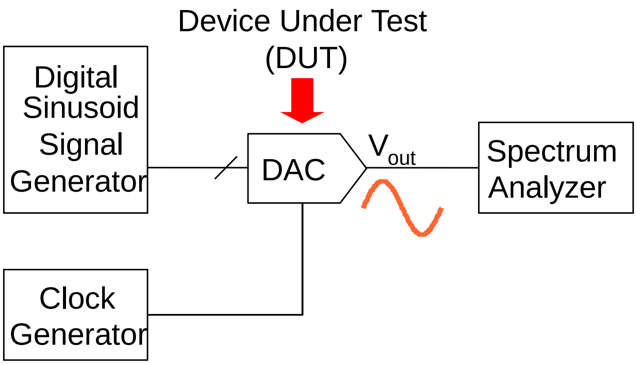
- Dynamic sparkle codes produce only minor DNL/INL errors
  E.g. 123, 123, …, 123, 0, 124, 124, … → look at ADC output to detect
- Noise not detected & averaged out
  E.g. 9, 9, 9, 10, 9, 9, 9, 10, 9, 10, 10, 10, …

Ref: B. Ginetti and P. Jespers, “Reliability of Code Density Test for High Resolution ADCs,” Electron. Lett., vol. 27, pp. 2231-3, Nov. 1991.

Why Additional Tests/Metrics?

- Static testing does not tell the full story
  - E.g. no info about "noise“ or high frequency effects
- Frequency dependence (f_s and f_{in}) ?
  - In principle we can vary f_s and f_{in} when performing histogram tests
  - Result of such sweeps is usually not very useful
  - Hard to separate error sources, ambiguity
  - Typically we use $f_s = f_{sNOM}$ and $f_{in} << f_s/2$ for histogram tests
- For additional info regarding higher frequency operation → Spectral testing

DAC Spectural Test or Simulation

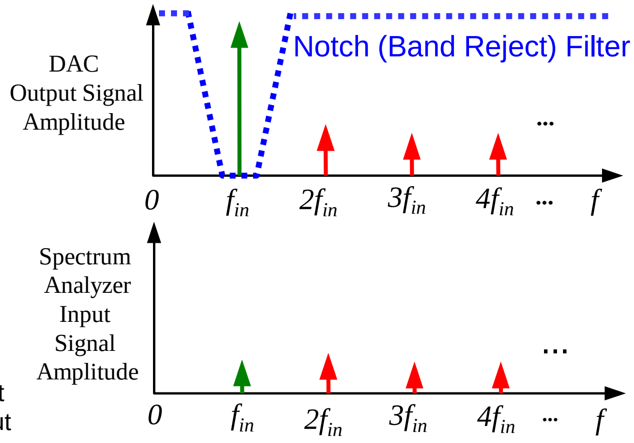


- Input sinusoid → Need to have significantly better purity compared to DAC linearity
- Spectrum analyzer need to have better linearity than DUT
- Typcally, test performed at several different input signal frequencies

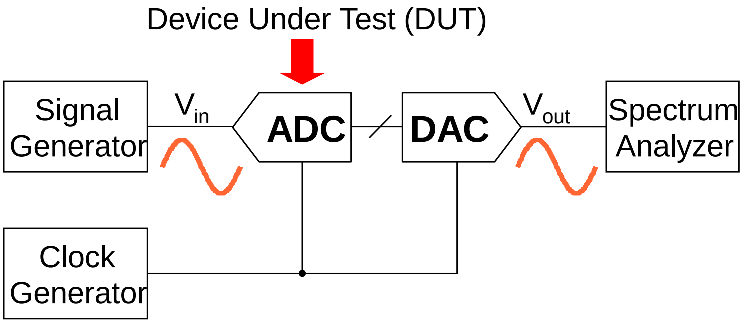
Filtering Input to Spectrum Analyzer

Prevent Signal Distortion Incurred by Spec. Analyzer

1. Measure fundamental signal level
2. Notch out fundamental signal so that Spec. Analyzer input signal becomes small enough not to drive S.A. input into non-linear region
3. Measure the harmonic content of the DAC output

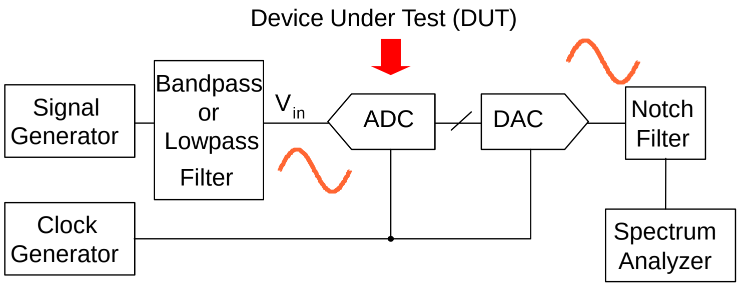


Direct ADC Spectral Test via DAC



- Need DAC with much better performance compared to ADC under test
- Beware of DAC output sinx/x frequency shaping
- Good way to "get started"...

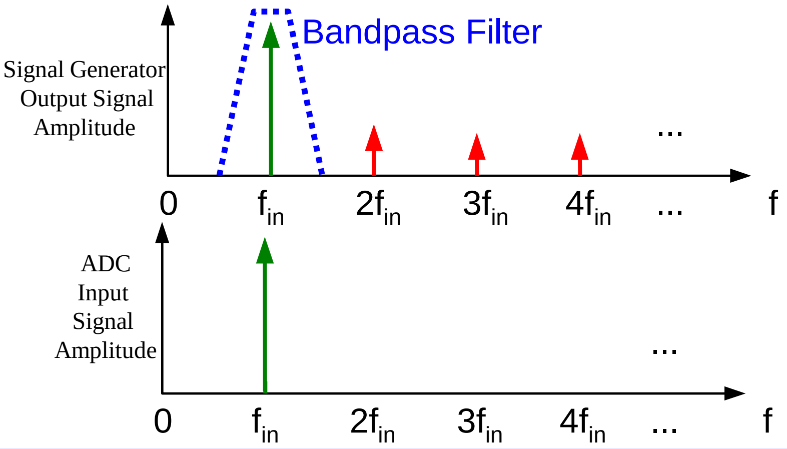
Direct ADC-DAC Test



- Issues to beware of:
  - Linearity of the signal generator output has to be much better than ADC linearity
  - Spectrum analyzer nonlinearities

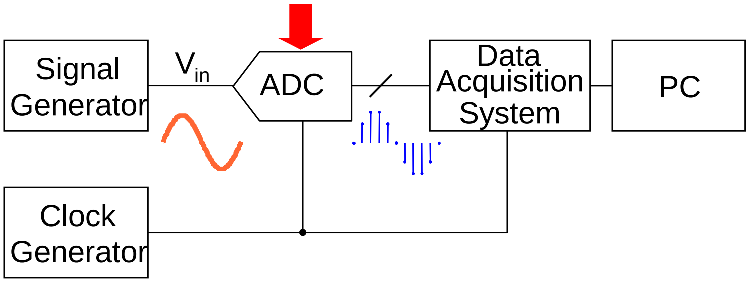
    → May need to build/purchase filters to address one or both above problems
  - Clock generator signal jitter

Filtering ADC Input Signal



ADC Spectral Test via Data Acquisition Sytem

Device Under Test (DUT)



Signal Generator

V_{in}

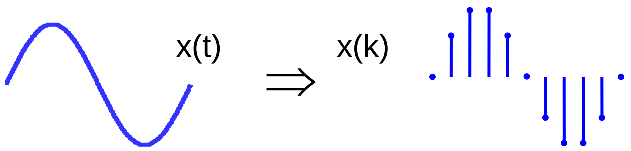
ADC

Data Acquisition System

PC

Clock Generator

Analyzing ADC Outputs via Discrete Fourier Transform (DFT)



- Sinusoidal waveform has all its power at one single frequency
- An ideal, infinite resolution ADC would preserve ideal, single tone spectrum
- DFT used as a vehicle to reveal ADC deviations from ideality

Discrete Fourier Transform (DFT) Properties

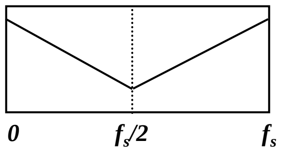
- DFT of N samples spaced $T_s = 1/f_s$ seconds:
  - N frequency bins from DC to f_s
  - Num of bins → N & each bin has width= f_s/N
  - Bin # m represents frequencies at $m * f_s/N$ $[Hz]$

- DFT frequency resolution:
  - Proportional to f_s/N in [Hz/bin]

- DFT with $N = 2^k$ (k is an integer) can be found using a computationally more efficient algorithm named:
  - FFT → Fast Fourier Transform

DFT Magnitude Plots

- Because magnitudes of DFT bins (A_m) are symmetric around $f_S/2$, it is redundant to plot $|A_m|$'s for $m > N/2$

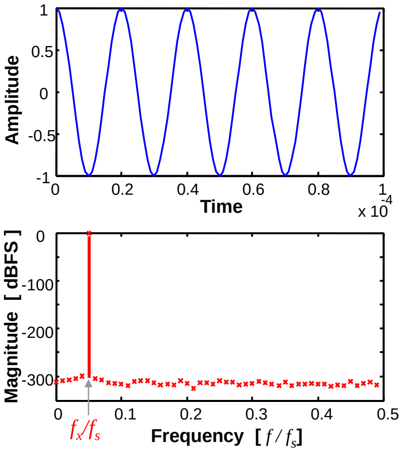


- Usually magnitudes are plotted on a log scale normalized so that a full scale sinusoidal waveform with *rms* value a_{FS} yields a peak bin of *0dBFS*:

$$|A_m|\ [dBFS] = 20\log_{10}\frac{|A_m|}{a_{FS}.N/2}$$

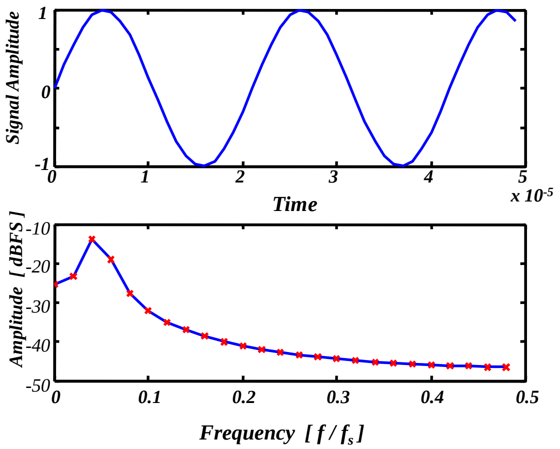
Matlab Example
Normalized DFT

```
fs  = 1e6;
fx  = 50e3;
Afs = 1;
N   = 100;

% time vector
t = linspace(0, (N-1)/fs, N);
% input signal
y = Afs * cos(2*pi*fx*t);
% spectrum
s = 20 * log10(abs(dft(y)/N/Afs*2));
% drop redundant half
s = s(1:N/2);
% frequency vector (normalized to fs)
f = (0:length(s)-1) / N;
```



Note: Where does the -300dBFS noise floor come from?

"Another" Example …

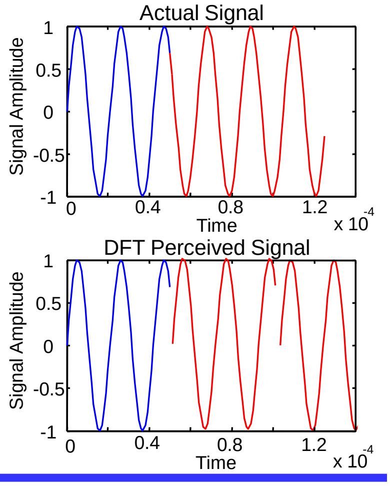


Even though the input signal is a pure sinusoidal waveform note that the DFT results does not look like the spectrum of a sinusoid …

Seems that the signal is distributed among several bins

DFT Periodicity

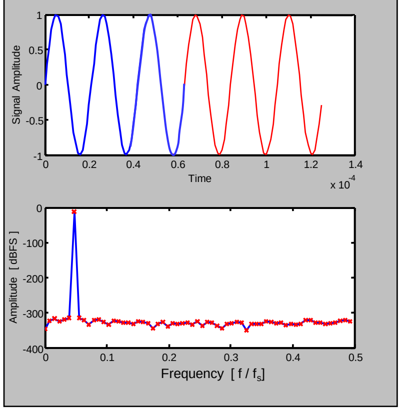
- The DFT implicitly assumes that time sample blocks repeat every N samples
- With a non-integer number of signal periods within the observation window, the input yields significant amplitude/phase discontinuity at the block boundary
- This energy spreads into other frequency bins as “spectral leakage”
- Spectral leakage can be eliminated by either
  1. Choice of integer number of sinusoids in each block
  2. Windowing



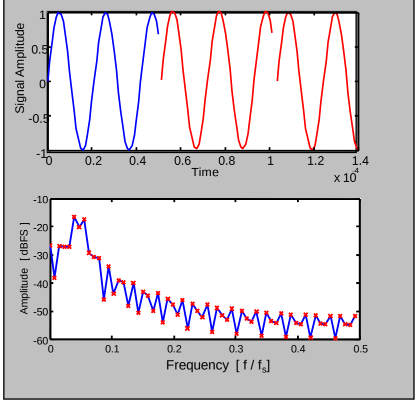
Frequency Spectrum

Integer # of Cycles versus Non-Integer # of Cycles

Integer number of cycles



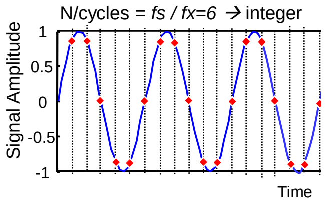
Non-integer number of cycles

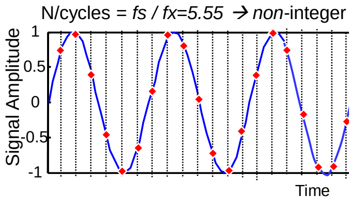


Choice of Number of Cycles & Number of Samples

To overcome frequency spectrum leakage problem:

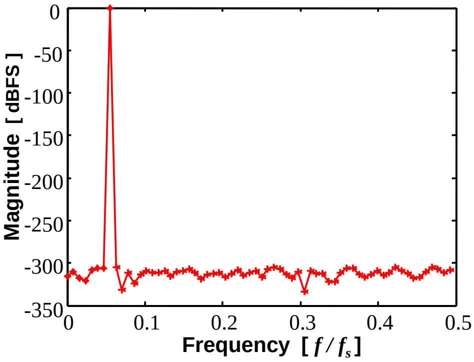
- Number of Cycles → integer
- N/cycles = f_s / f_x → non-integer (choose prime # of cycles) otherwise quant. noise → periodic and non-random
- Preferable to have N: → power of 2 (FFT instead of DFT)





Example: Integer Number of Cycles

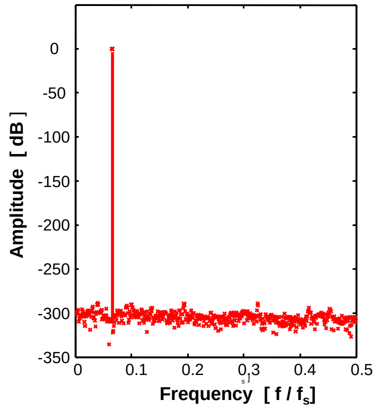
```
fs = 1e6;
% Number of cycles in test
cycles = 67;

%Make N/cycles non-
   integer!
 accomplished by choosing
   cycles→ prime #

%N=power of 2 speeds up
   analysis

N = 2^10;

%signal frequency
fx = fs*cycles/N
y = Afs * cos(2*pi*fx*t);
s = 20 * log10(abs(fft(y)/N/Afs*2));
```



Notice: Range of test signals limited to *[(cycles)x f_s/N]*

Example: Integer Number of Cycles

- Fundamental falls into a single DFT bin
- Noise (this example numerical quantization noise) occupies all other bins
- “integer number of cycles” constrains signal frequency f_x
- Alternative: windowing →

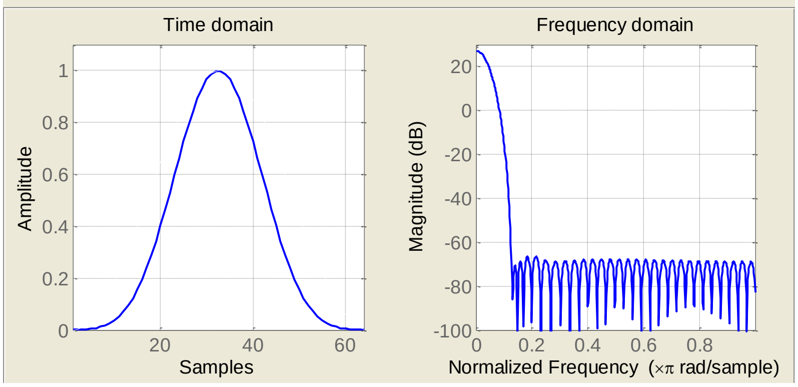


Windowing

- Spectral leakage can be attenuated by “windowing” time samples prior to the DFT
  - Windows taper smoothly down to zero at the beginning and the end of the observation window
  - Time samples are multiplied by window coefficients on a sample-by-sample basis

    → Convolution in frequency domain
- Large number choices of various windows
  - Tradeoff: attenuation versus fundamental signal spreading to number of adjacent bins
- Window examples: Nuttall versus Hann

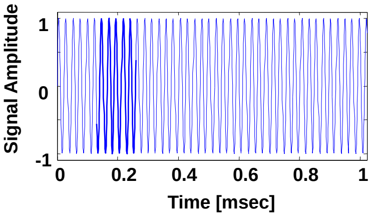
Example: Nuttall Window



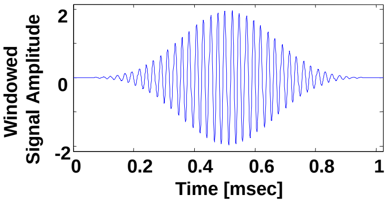
- Time samples are multiplied by window coefficients on a sample-by-sample basis
- Multiplication in the time domain → convolution in the frequency domain

Windowed Data

- Signal before windowing

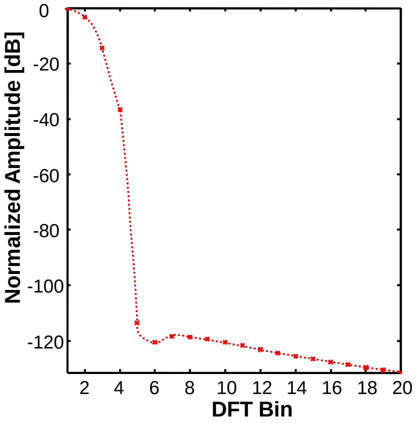


- Time samples are multiplied by window coefficients on a sample-by-sample basis
- Signal after windowing
  - Windowing removes the discontinuity at block boundaries



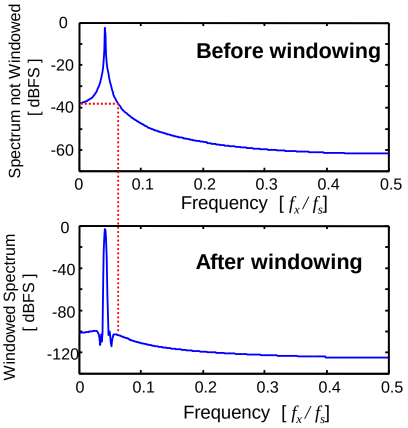
Nuttall Window DFT

- Only first 20 bins shown
- Response attenuated by -120dB for bins > 5
- Lots of windows to choose from (go by name of inventor- Blackman, Harris, Nutall…)
- Various window trade-off attenuation versus width (smearing of sinusoids)

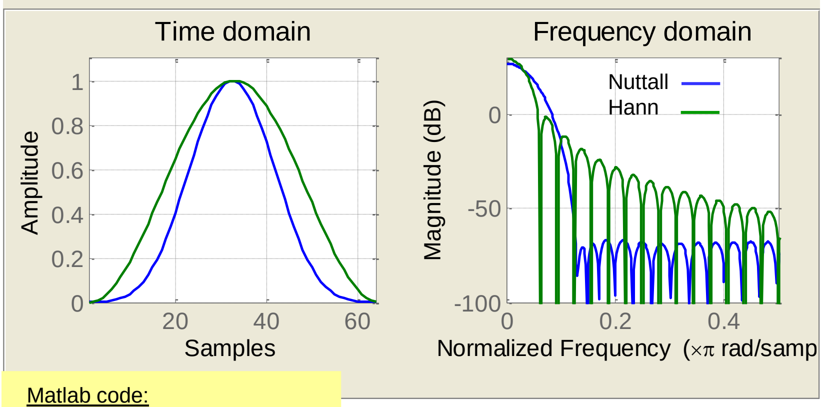


DFT of Windowed Signal Spectrum Before/After Windowing

- Windowing results in ~ 100dB attenuation of sidelobes
- Signal energy “smeared” over several (approximately 10) bins



Window
Nuttall versus Hann



Matlab code:

```
N=64;
wvtool(nuttallwin(N),hann(N));
```

Integer Cycles versus Windowing

- Integer number of cycles
  - Signal energy for a single sinusoid falls into single DFT bin
  - Requires careful choice of f_x
  - Ideal for simulations
  - Measurements → need to lock f_x to f_s (PLL)- not always possible
- Windowing
  - No restrictions on f_x → no need to have the signal locked to f_s → Good for measurements w/o having the capability to lock f_x to f_s or cases where input is not periodic
  - Signal energy and its harmonics distributed over several DFT bins – handle smeared-out harmonics with care!
  - Requires more samples for a given accuracy
  - Note that no windowing is equal to windowing with a rectangular window!

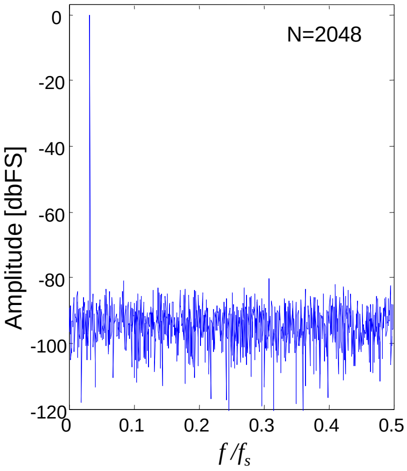
Example: ADC Spectral Testing

- ADC with B bits
- Full scale input level=2

```
B = 10;
delta = 2/2^B;
%sampled sinusoid
y = cos(2*pi*fx/fs*[0:N-1]);
%quantize samples to delta=1LSB
y=round(y/delta)*delta;
s = abs(fft(y)/N*2);
f = (0:length(s)-1) / N;
```

ADC Output Spectrum

- Input signal bin:
  - Bx @ bin # ($N * f_x/f_s + 1$) (Matlab arrays start at 1)
  - A_{signal} = 0dBFS
- What is the SNR?

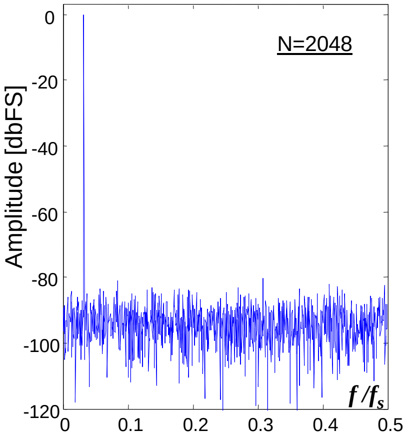


Simulated ADC Output Spectrum

- Noise bins: all except signal bin

```
bx = N*fx/fs + 1;
As = 20*log10(s(bx))
%set signal bin to 0
s(bx) = 0;
An = 10*log10(sum(s.^2))
SNR = As - An
```

- Matlab→SNR = 62dB (10 bits)
- Computed SQNR = 6.02xN+1.76dB=61.96dB



Note: In a real circuit including thermal/flicker noise → the measured total noise is the sum of quantization & noise associated with the circuit

Why is Noise Floor Not @ -62dB ?

- DFT bins act like an analog spectrum analyzer with bandwidth per bin of f_s/N
- Assuming noise is uniformly distributed, noise per bin:

  (Total noise)/N/2

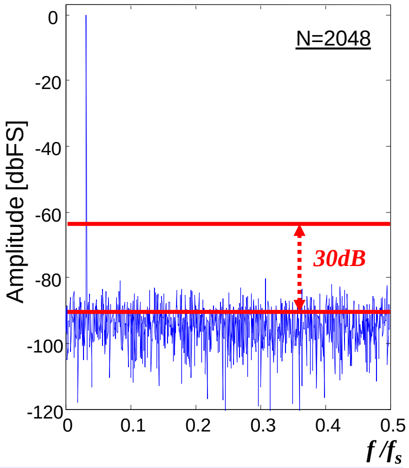
  →The DFT noise floor wrt total noise:

  $-10\log_{10}(N/2)$ [dB]

  below the actual noise floor

- For N=2048:

  $-10\log_{10}(N/2) = -30$ [dB]

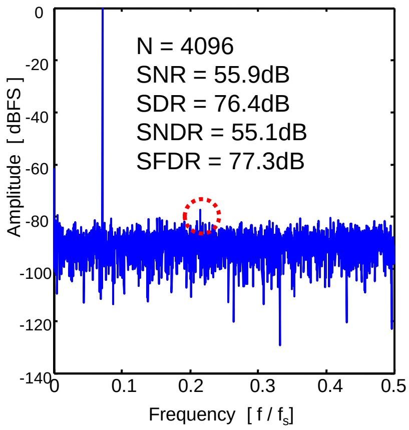


DFT Plot Annotation

- Need to annotate DFT plot such that actual noise floor can be readily computed by one of these 3 ways:
  1. Specify how many DFT points (N) are used
  2. Shift DFT noise floor by $10\log_{10}(N/2)$ [dB]
  3. Normalize to "noise power in 1Hz bandwidth“ then noise is in the form of power spectral density

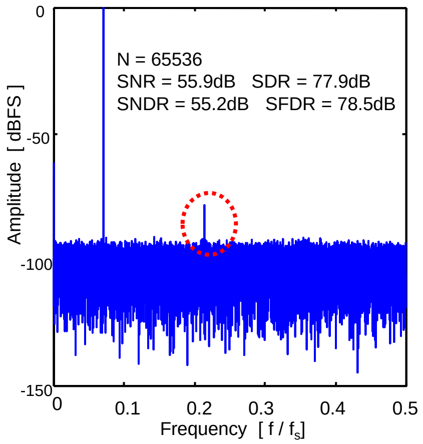
Example:10Bit ADC FFT

- For a real 10bit ADC spectral test results:
- SNR=55.9dB
- A 3rd harmonic is barely visible
- Is better view of distortion component possible?


N = 4096
SNR = 55.9dB
SDR = 76.4dB
SNDR = 55.1dB
SFDR = 77.3dB

Example:10Bit ADC FFT

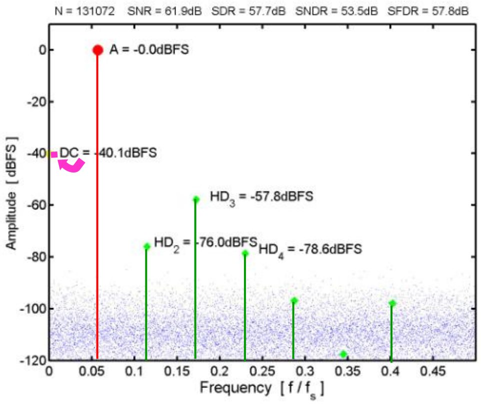
- Increasing N, the number of samples (and hence the measurement or simulation time) distributes the noise over larger # of bins
- Larger # of bins → less noise power per bin (total noise stays constant)
- Note the 3rd harmonic is clearly visible when N is increased



Spectral Performance Metrics
ADC Including Non-Idealities

- Signal S
- DC
- Distortion D
- Noise N

- Ideal ADC adds:
  - Quantization noise
- Real ADC typically adds:
  - Thermal and flicker noise
  - Harmonic distortion associated with circuit nonlinearities

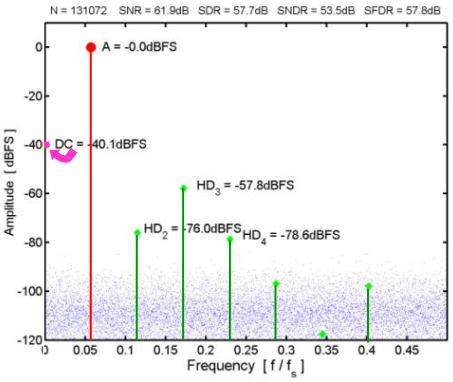


ADC Spectral Performance Metrics
SNR

- Signal S
- DC
- Distortion D
- Noise N

- Signal-to-noise ratio
  SNR = 10log[(Signal Power) /
  (Noise Power)]

- In Matlab: Noise power includes power associated with all bins except:
  - DC
  - Signal
  - Signal harmonics

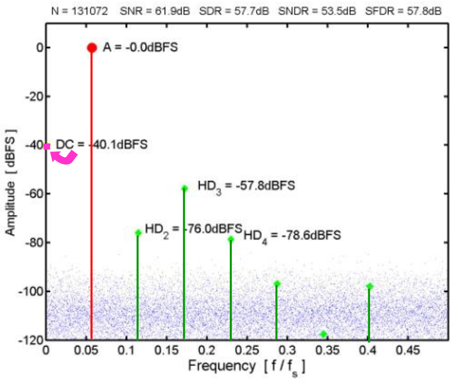


ADC Spectral Performance Metrics
SDR & SNDR & SFDR

- SDR→ Signal-to-distortion ratio
  = 10log[(Signal Power) /
  (Total Distortion Power)]

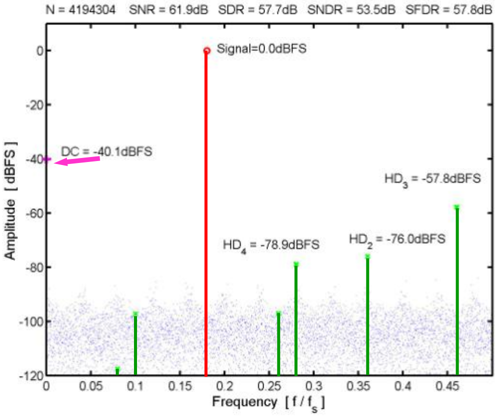
- SNDR→ Signal-to-(noise+distor
  = 10log[S / (N+D)]

- SFDR→ Spurious-free dynamic range
  = 10log[(Signal)/
  (Largest Harmonic)]
  → Typically SFDR > SDR



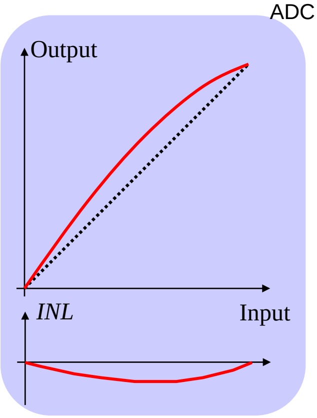
Harmonic Components

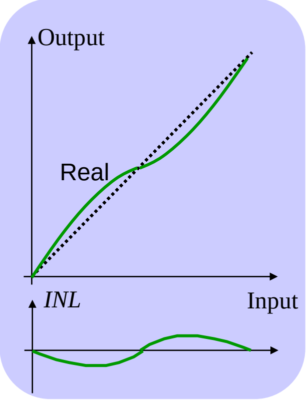
- At multiples of f_x
- Aliasing:
  - $f_{signal} = f_x = 0.18\ f_s$
  - $f_2 = 2\ f_0 = 0.36\ f_s$
  - $f_3 = 3\ f_0 = 0.54\ f_s$ → $0.46\ f_s$
  - $f_4 = 4\ f_0 = 0.72\ f_s$ → $0.28\ f_s$
  - $f_5 = 5\ f_0 = 0.90\ f_s$ → $0.10\ f_s$
  - $f_6 = 6\ f_0 = 1.08\ f_s$ → $0.08\ f_s$



Relationship INL & SFDR/SNDR

ADC Transfer Curve

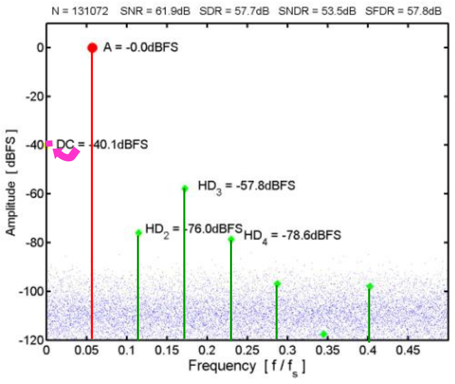


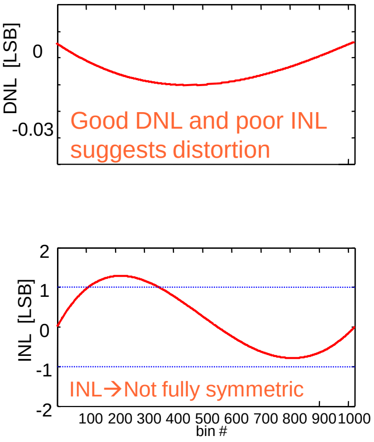


Quadratic shaped transfer function:
→ Gives rise to **even** order harmonics

Cubic shaped transfer function:
→ Gives rise to **odd** order harmonics

Frequency Spectrum versus INL & DNL





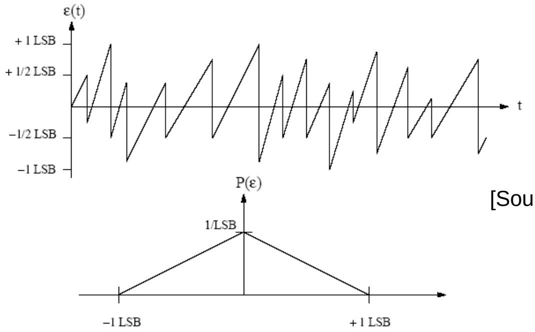
Good DNL and poor INL suggests distortion

INL→Not fully symmetric

Relationship INL & SFDR/SNDR

- Nature of harmonics depend on "shape" of INL curve
- Rule of Thumb: SFDR ≅ 20log(2^B/INL)
  – E.g. 1LSB INL, 10b→ SFDR≅60dB
- Beware, this is of course only true under the same conditions at which the INL was taken, i.e. typically low input signal frequency

SNR Degradation due to DNL



[Source: Ion Opris]

- Uniform quantization error pdf was assumed for ideal quantizer over the range of: +/- Δ/2
- Let's now add uniform DNL over +/- Δ/2 and repeat math...
  - Joint pdf for two uniform pdfs → Triangular shape

SNR Degradation due to DNL

- To find total noise → Integrate triangular pdf:

$$\overline{e^2} = 2\int_{0}^{+\Delta}(1-e)\frac{e^2}{\Delta}de = \frac{\Delta^2}{6} \qquad \Rightarrow SNR = 6.02\cdot N - 1.25\ [\mathrm{dB}]$$

3dB

- Compare to ideal quantizer:

$$\overline{e^2} = \int_{-\Delta/2}^{+\Delta/2}\frac{e^2}{\Delta}de = \frac{\Delta^2}{12} \qquad \Rightarrow SNR = 6.02\cdot N + 1.76\ [\mathrm{dB}]$$

→Error associated with DNL reduces overall SNR

SNR Degradation due to DNL

- More general case:
  - – Uniform quantization error (ideal) ±0.5Δ
  - – Uniform DNL error ± DNL [LSB]
  - – Convolution yields trapezoid shaped joint pdf
  - – SQNR becomes:

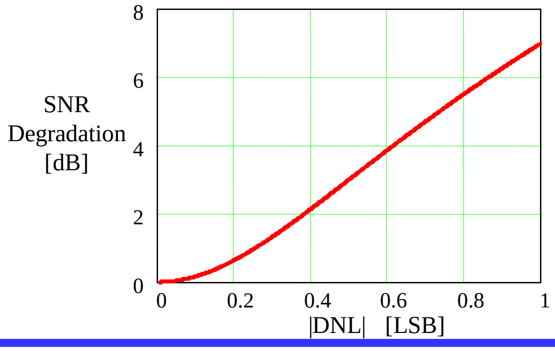
$$\boxed{SQNR = \frac{\frac{1}{2}\left(\frac{2^N \Delta}{2}\right)^2}{\frac{\Delta^2}{12} + \frac{DNL^2}{3}}}$$

SNR Degradation due to DNL

- Degradation in dB:

$$SQNR\_\deg = 1.76 - 10\log\left[\frac{\frac{1}{8}}{\frac{1}{12} + \frac{DNL^2}{3}}\right]$$

⇦ Valid only for cases where no missing codes



Summary
INL & SFDR - DNL & SNR

INL & SFDR

- Type of distortion depends on "shape" of INL
- Rule of Thumb:

$$\boxed{SFDR \cong 20\log(2^B/INL)}$$

  - E.g. 1LSB INL, 10b → SFDR≅60dB

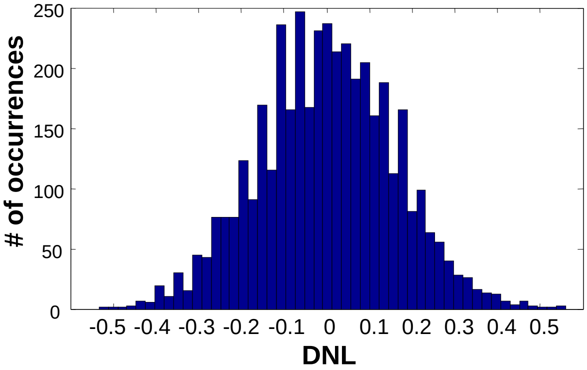
DNL & SNR

Assumptions:

- DNL pdf →uniform
- No missing codes

$$\boxed{SQNR = \frac{\frac{1}{2}\left(\frac{2^N\Delta}{2}\right)^2}{\frac{\Delta^2}{12}+\frac{DNL^2}{3}}}$$

Uniform DNL?



- DNL distribution of 12-bit ADC test chip
- Not quite uniform...

Effective Number of Bits (ENOB)

- Is a 12-Bit converter with 68dB SNDR really a 12-Bit converter?
- Effective Number of Bits ($ENOB$)→ # of bit of an ideal ADC with the same SQNR as the SNDR of the non-ideal ADC

$$\boxed{ENOB = \frac{SNDR - 1.76\text{dB}}{6.02\text{dB}}}$$

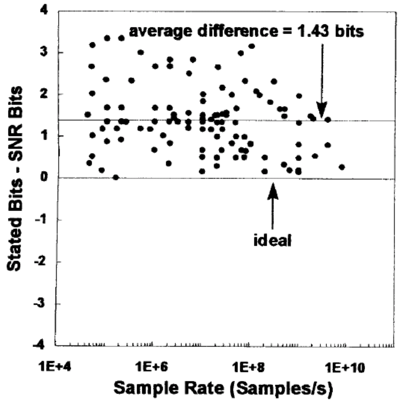
$$= \frac{68 - 1.76}{6.02} = 11.0\text{Bits}$$

→ Above ADC is a 12bit ADC with ENOB=11bits

ENOB

- At best, we get "ideal" ENOB only for negligible thermal noise, DNL, INL
- Low noise design is costly → 4x penalty in power per (ENOB-) bit or 6dB extra SNDR
- Rule of thumb for good performance /power tradeoff: ENOB < N-1

ENOB Survey



R. H. Walden, "Analog-to-digital converter survey and analysis," *IEEE J. on Selected Areas in Communications*, pp. 539-50, April 1999