Quantizing a random signal - N1AL 6/5/2009

Compared to a periodic waveform like a sine wave, a voice signal is much more random. We will simulate a voice signal with random noise filtered to a 1 kHz bandwidth. Then we will quantize it with 8 bits of resolution and compare the spectra of the continuous and the quantized signals.

$fs := 10000$ Sample rate $f := 1000$ Bandwidth $bits := 8$ Resolution

$n := 10000$ Number of samples $i := 0 .. n - 1$ Sample index

$A_i := rnd(1) - 0.5$ 10,000 samples of unfiltered random noise

$fwidth := 500$ Filter width $j := 0 .. n - fwidth - 1$ Fewer filtered samples to account for filter width

$k := 0 .. fwidth - 1$ Filter index

$$sinc_k := \frac{\sin\left[\pi\cdot\left(k - \frac{fwidth}{2} + 0.5\right)\cdot\frac{2\cdot f}{fs}\right]}{\pi\cdot\left(k - \frac{fwidth}{2} + 0.5\right)\cdot\frac{2\cdot f}{fs}}$$

An FIR filter with an impulse response in the shape of a sinc function theoretically results in a "brick wall" ideal low-pass filter.

$$hann_k := \frac{\left[1 + \cos\left[\pi\cdot\left(k - \frac{fwidth}{2} + 0.5\right)\cdot\frac{2}{fwidth}\right]\right]}{2}$$

The impulse response must be windowed to reduce passband and stopband ripple due to the finite impulse response length. We will use a Hann (Hanning) window.

$$Afilt_j := \sum_k A_{j+k}\cdot sinc_k\cdot hann_k$$

Convolve the windowed filter impulse response with the signal.

$$RMS := \sqrt{\frac{\sum_j \left(Afilt_j\right)^2}{n - fwidth}}$$

Calculate the RMS value of the filtered signal.

And normalize by the RMS value: $Afilt_j := \frac{Afilt_j}{RMS}$

$scale := 2^{bits-1} - 1$ $scale := \frac{scale}{max(Afilt)}$

Scale the signal so that the peak is at full scale of the analog-to-digital conversion.

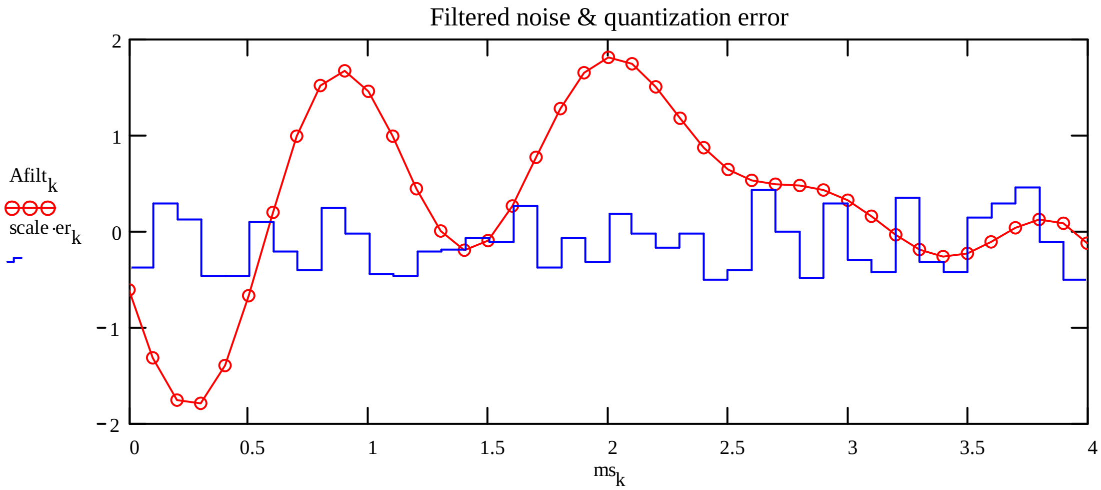
$$Adig_j := \frac{floor\left(scale\cdot Afilt_j + 0.5\right)}{scale}$$

Digitize the signal = convert from analog to digital.

$er_j := Adig_j - Afilt_j$ The error signal is the difference between the original signal and the quantized signal

$start := 70$

$k := start .. start + 40$ Counter for points to plot. $ms_k := (k - start)\cdot\frac{1000}{fs}$ Time in millisconds



Now let's plot the frequency spectra of the original and quantized filtered-noise signals.
First, we'll need to window the entire 9500-point sequence before doing the FFT.
Let's use a Hann window again:

$$hann_j := \frac{\left[1 + \cos\left[\pi \cdot \left(j - \frac{n - fwidth}{2} + 0.5\right) \cdot \frac{2}{n - fwidth}\right]\right]}{2}$$

Window the original signal:

$$Afilt_j := Afilt_j \cdot hann_j$$

Window the digitized signal:

$$Adig_j := Adig_j \cdot hann_j$$

The regular FFT functions in Mathcad require that the input be a real sequence of length 2^n.
Our sequence is real but of the wrong length so we have to use the "complex FFT" cfft():

$S := cfft(Afilt)$ Frequency spectrum of undigitized signal.

$Sd := cfft(Adig)$ Frequency spectrum of digitized signal.

$$m := ceil\left(\frac{n}{2}\right)$$

$$j := 0 .. m$$

The Fourier transform of a real signal gives a symmetrical frequency spectrum. That is, the spectrum from 0 to fs/2 is the mirror image of the spectrum from -fs/2 to 0 (which is the same as the spectrum from fs/2 to fs). So the output of the normal FFT function has only half the samples since the other half is the same anyway. The complex FFT gives the entire spectrum but we will only use half of it since we actually have a real input.

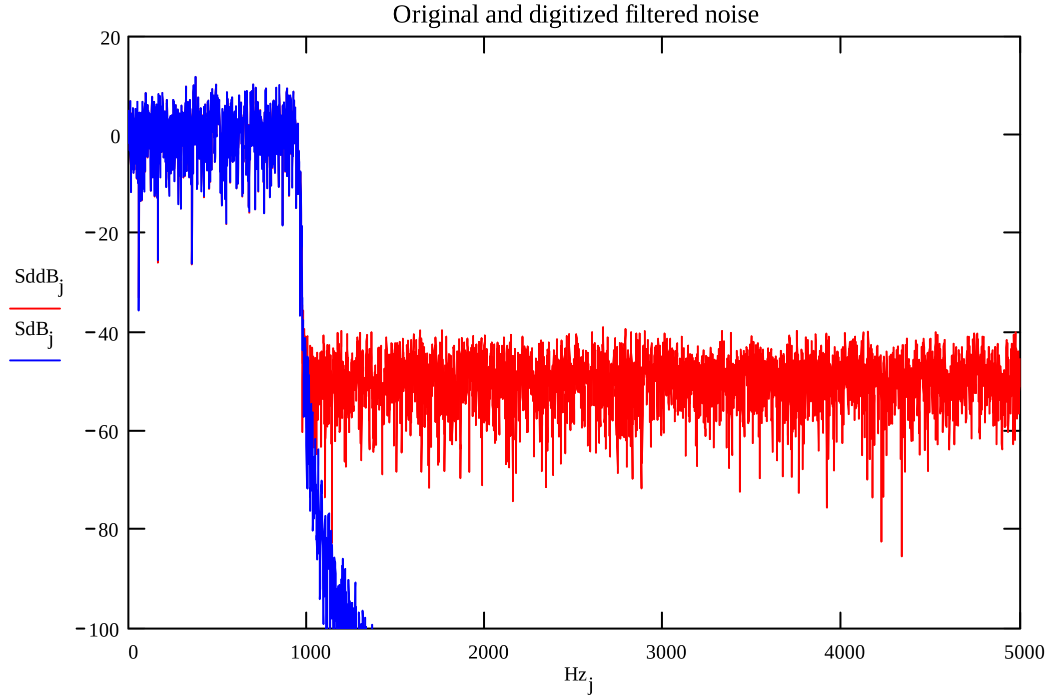
$$Hz_j := j \cdot \frac{fs}{n}$$

Frequency in Hz as a function of FFT sample.

$$SdB_j := 20 \cdot \log\left(\left|S_j\right| + 0.000001\right)$$

$$SddB_j := 20 \cdot \log\left(\left|Sd_j\right| + 0.000001\right)$$

Convert frequency spectra into dB.
The 0.000001 fudge factor is to prevent taking the log of zero in case some sample is zero.



With an 8-bit quantizer, the quantization noise should theoretically be
SNR_{eff} = 1.76 + 8 * 6.02 + 10 log(10,000/(2*1000)) = 56.9 dB.

However, that assumes a sine-wave signal whose RMS power is only 3 dB below the peak. With a noise signal, the peak-to-RMS ratio is greater, so SNR_{eff} is somewhat worse.