The Filter Wizard
issue 2: Alias, Damned Alias and Statistics
Kendall Castor-Perry

There have been some recent articles discussing aliasing. I thought I'd jump on the bandwagon – that one in the movies whose wheels seem to be rotating the wrong way. Is aliasing a problem? Does it need fixing, and does fixing it cause another problem elsewhere? Answering that is for a later column; today, a health warning on the arithmetic of aliasing. It's customary to demonstrate with the trigonometric identity

$$\sin(a \pm b) = \sin(a)\cos(b) \pm \cos(a)\sin(b)$$

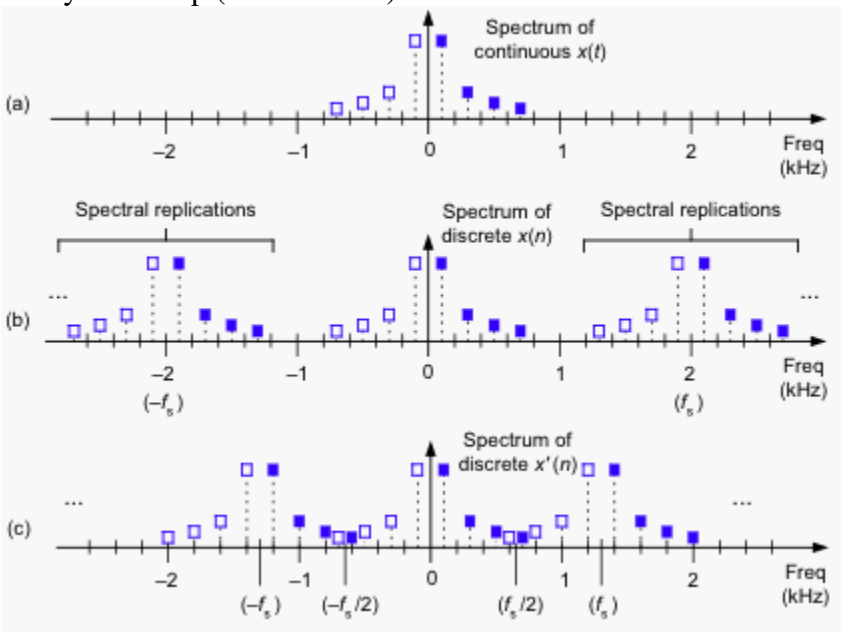
by substituting a = $2n\pi t/\tau$ and b = $2\pi ft$ (homework – try this!) that *if* it's possible to fit a set of points x(t) spaced at interval τ, with a sinewave at frequency f, *then* infinitely many sinusoidal solutions are possible, at frequencies $n/\tau \pm f$ for integer n. One of the solutions is the 'right' one and the rest are 'impostors' or 'aliases'. But which? Baseband guys assume the lowest-frequency solution is wanted when they process x(t), and incorrectly infer the presence of a baseband signal at f if x(t) was actually created by sampling one of the higher-frequency aliases. I spent many years designing filters – anti-aliasing filters – to keep those bad boys out (the signals, not the baseband guys).

Radio men vibrate to a higher harmonic; they usually reckon that one of the *higher* frequency solutions is the wanted one and that x(t) arose by sub-sampling the signal, at a frequency *below* the input frequency. They need anti-aliasing filters too; their filters need to have *bandpass* response because they must keep not only higher- but also *lower*-frequency solutions from getting into their ADCs.

What does the *spectrum* of that data set x(t) actually look like? In the real world, we play x(t) out through a DAC that pings out a pulse of value x(ti) at time ti, and check the output with a spectrum analyzer. Within the warm embrace of the virtual world, we perform an FFT on x(t). What we find is... *all* - that's right, *all* - of the frequencies that could have 'aliased down' (or up) to give a particular data set x(t) are present in the spectrum of x(t), whether or not they were present in the signal from which it was sampled.

I once heard an apps guy explain that aliasing happened when you put an input signal at a frequency greater than half the sampling frequency (usually called the Nyquist frequency) into an ADC and it "came out as" a signal at another frequency less than the Nyquist frequency, as if it had been reflected off an invisible mirror. This is a nice analogy but it obscures the truth. If you look at the spectrum of a sampled signal, *the original frequency component is always there*. Read that again; this point is not made clearly enough or often enough. It's just that the impostor frequencies are *also* present, and sometimes they turn up where our prejudice tells us that the 'actual' signal should be. Nothing's gone missing, we just picked up some hitchhikers.

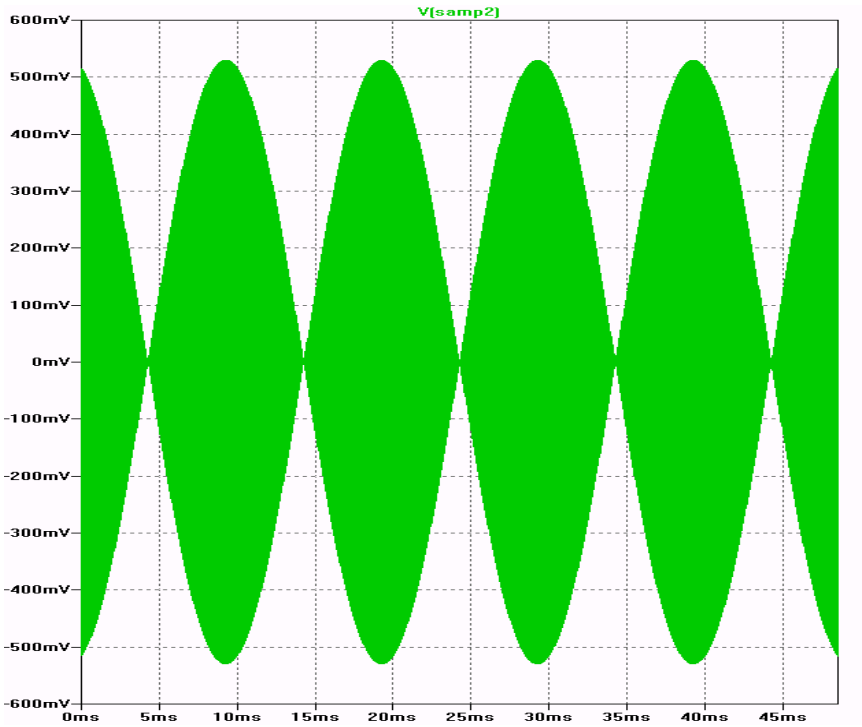
Perhaps the (German) apps guy was led astray by his *Muttersprache*. The German term for aliasing is *Rückfaltung*, or 'folding-back', and this reinforces the misconception that the signal frequency has been changed. Practitioners who work in the sampled domain know that the problem with aliasing that muddles things up is that the frequency spectrum of a complicated signal can end up *overlapping* with the extra spectrum formed through the sampling process. Once these signals have overlapped, it's generally not possible to disentangle them. All the original frequencies are still present, it's just that there's now a region where these frequencies are mixed up with each others' aliases. Here's a picture from a recent EETAsia article by Richard Lyons, showing stuff getting nicely mixed up (bottom trace):



(from http://www.eetasia.com/ART_8800561914_499489_NT_a9c51920.HTM).

This is visible in the reproduction of high frequency signals by digital audio systems. Audio DACs use filters to convert the digits into smooth (or even smoo-ooth, if you like that sort of music) signals by knocking out all the high frequency components. Well, *almost* all. To cut chip cost, a particular efficient digital filter design (called a half-band filter, and used in ADCs too) is often used. This filter doesn't have much rejection around half the sampling frequency – only 6dB, in fact.

Take a CD containing an x(t) sampled from a 22kHz sinewave at the standard 44.1kHz rate. 50ms of this signal (attenuated almost 6dB by the half-band filter) looks like this:



What does the spectrum of x(t) look like? Correct: *all* the possible sinewave solutions that fit x(t) will be there alongside the 22kHz one. The most worrying one is at 22.1kHz, which our low-cost digital filter lets through at just slightly lower amplitude than the 22kHz component. What happens when you sum a 22kHz sinewave and a 22.1kHz sinewave at nearly the same amplitude? Right again! You get something that looks like a 22.05kHz sinewave almost fully balanced-modulated with 50Hz – very like the plot of the sampled output above. If that hits any kind of nonlinearity... well, just think of how AM radio works. Your second homework assignment: lash this up in LTSpice or some other simulator.

And of course we don't know whether x(t) was the result of applying 22kHz or 22.1kHz data to the ADC, for the same reason. In other words, we can apply *either* 22kHz *or* 22.1kHz to the ADC and we'll get *both* of them at the DAC's output. And if *that's* accurate reproduction of the original signal, then I'm not at my editor's word count limit. So, let me know if the aliasing monster ever bit *you* when you weren't looking! - Kendall