


# Sampling: What Nyquist Didn't Say, and What to Do About It

Tim Wescott, Wescott Design Services

*The Nyquist-Shannon sampling theorem is useful, but often misused when engineers establish sampling rates or design anti-aliasing filters. This article explains how sampling affects a signal, and how to use this information to design a sampling system with known performance.*

June 20, 2016

# What Nyquist Did Say

The assertion made by the Nyquist-Shannon sampling theorem is simple: if you have a signal that is perfectly band limited to a bandwidth of $f_0$ then you can collect all the information there is in that signal by sampling it at discrete times, as long as your sample rate is greater than $2f_0$ . As theorems go this statement is delightfully short. Unfortunately, while the theorem is simple to state it can be very misleading when one tries to apply it in practice.

Often when I am teaching a seminar, working with a client, or reading a Usenet newsgroup, someone will say "Nyquist says", then follow this with an incorrect use of the Nyquist-Shannon theorem. This paper is about explaining what the Nyquist-Shannon sampling theorem really says, what it means, and how to use it.

It is a common misconception that the Nyquist-Shannon sampling theorem could be used to provide a simple, straight forward way to determine the correct minimum sample rate for a system. While the theorem does establish some bounds, it does not give easy answers. So before you decide the sampling rate for your system, you have to have a good understanding of the implications of the sampling, and of the information you really want to measure.

The difficulty with the Nyquist-Shannon sampling theorem is that it is based on the notion that the signal to be sampled must be perfectly band limited. This property of the theorem is unfortunate because no real world signal is truly and perfectly band limited. In fact, if a signal were to be perfectly band limited—if it were to have absolutely no energy outside of some finite frequency band—then it must extend infinitely in time.

What this means is that no system that samples data from the real world can do so perfectly—unless you're willing to wait an infinite amount of time for your results. If no system can sample data perfectly, however, why do we bother with sampled time systems? The answer, of course, is that while you can never be *perfect*, with a bit of work you can design sampled time systems that are *good enough*. Often, in fact, the advantages one gains by processing signals in sampled time far outweigh the disadvantages of sampling, making many sampled time systems superior to their continuous-time equivalents.

To understand how to make a sampled time system that's good enough we must understand what happens when a signal is sampled into the discrete-time domain, what happens when it is reconstructed in the continuous-time domain, and how these processes affect the quality of the signal.

# Sampling

So what is sampling, and what does it do? Sampling is the process by which continuous-time signals, such as voltages or water levels or altitudes, are turned into discrete time signals. This is usually done by translating the signal in question into a voltage, then using

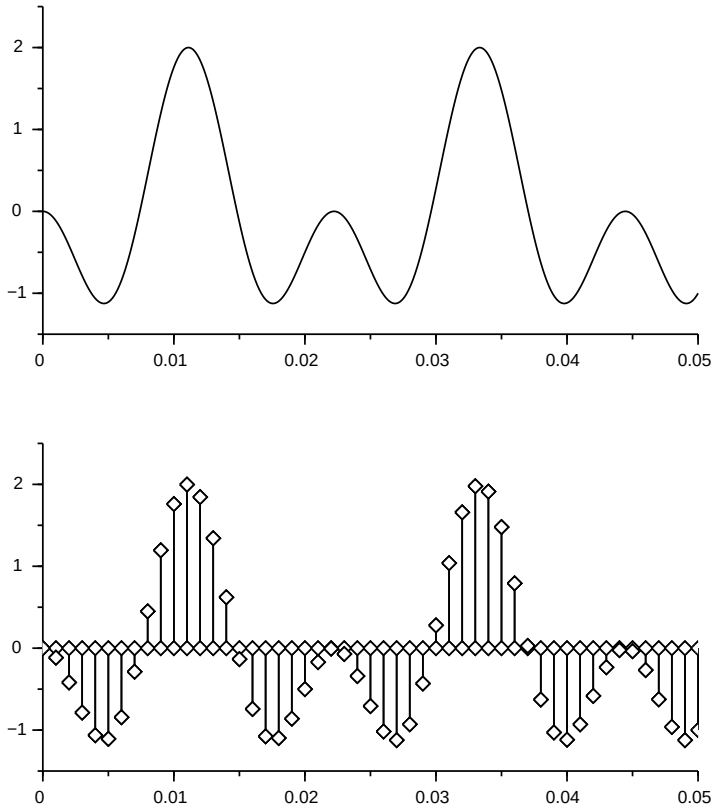


Figure 1: The results of Sampling

an analog to digital converter (ADC) to turn this continuous, analog signal into a discrete, digital one. The ADC both samples[1] the voltage and converts it to a digital signal.

The sampling process itself is easy to represent mathematically: given a continuous signal $x(t)$ to be sampled, and a sample interval $T$, the sampled version of $x$ is simply the continuous version of $x$ taken at integer intervals of $T$:

$$x_k = x\left(kT\right),\ k \in \mathbb{I} \tag{1}$$

Figure 1 shows the result of sampling a signal. The upper trace is the continuous-time signal, while the lower trace shows the signal after being sampled once per millisecond. You may wonder why the lower trace shows no signal between samples. This is because after sampling there is no signal between samples—all the information that existed between the samples in the original signal is irretrievably lost in the sampling process.

[1] Older ADCs and some very specialized systems may require external sampling circuitry, but nearly all newly designed ADC integrated circuits have their own sampling circuitry built in.

## Aliasing

By ignoring anything that goes on between samples the sampling process throws away information about the original signal[2]. This information loss must be taken into account during system design. Most of the time, when folks are designing systems they are doing their analysis in the frequency domain. When you are doing your design from this point of view you call this effect aliasing, and you can easily express it and model it as a frequency-domain phenomenon.

To understand aliasing, consider a signal that is a pure sinusoid, and look at its sampled version:

$$x_k = cos\left(\omega\, k\, T\right) \tag{2}$$

If you know the frequency of the original sine wave you'll be able to exactly predict the sampled signal. This is a concept is easy to grasp and apply. But the sampled signal won't necessarily seem to be at the same frequency as the original signal: there is an ambiguity in the signal frequency equal to the sampling rate. This can be seen if you consider two signals, one at frequency $f$ and one at frequency $f\ +\ {}^1\!/_T$. Using trigonometry, you can see that the sampled version of these two signals will be exactly the same:

$$\cos\left(2\pi\left(f+\frac{1}{T}\right)k\,T\right) = \cos\left(2\pi k + 2\pi\, f\, k\, T\right) = \cos\left(2\pi\, f\, k\, T\right) \tag{3}$$

This means that given a pair of sampled versions of the signals, one of the lower frequency sinusoid and one of the higher, you will have no way of distinguishing these signals from one another. This ambiguity between two signals of different frequencies (or two components of one signal) is aliasing, and it is happening all the time in the real world, anywhere that a real-world signal is being sampled.

Figure 2 on the next page shows an example of aliasing. Two possible input sine waves are shown: one has a frequency of 110Hz, the other has a frequency of 1110Hz. Both are sampled at 1000Hz. The dots show the value of the sine waves at the sampling instants. As indicated by (2) these two possible inputs both result in exactly the same output: after sampling you cannot tell these two signals apart.

It is rare, however, for real-world signals to resemble pure sine waves. In general, real-world continuous-time signals are more complex than than simple sine waves. But we can use what we learn about the system's response to pure sine wave input to predict the behavior of a system that is presented with a more complex signal. This is because more complex continuous-time signals can be represented as sums of collections of sine waves at different frequencies and amplitudes[3]. For many systems we can break the signal down

[2] If the signal is perfectly band limited then no real information is lost—but you can't tell that just by looking at the sampled signal, as we'll see later

[3] This can be expressed formally using the Fourier transform; I will restrict myself to an informal discussion here, but works such as [Opp83] give very good, formal discussions of the underlying mathematics. For a less formal, but still valid, treatment, see [Lyons04].

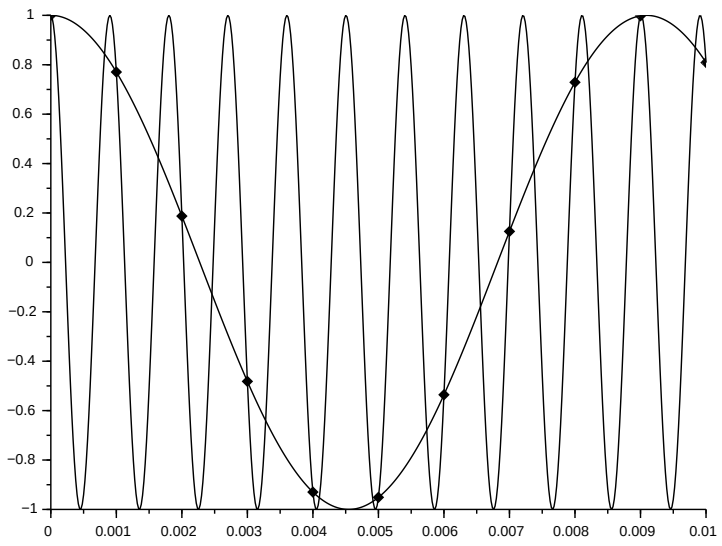


Figure 2: Aliasing of two sine waves.

into its component parts, analyze the system's response to each part separately, then add these responses back together to get the system's response[4].

When you break a signal down into its component sine waves, you see that the signal's energy is distributed as a function of frequency. This distribution of a signal's energy over frequency can be shown as a plot of spectral density vs. frequency, such as the solid plot in the center of Figure 3 on page 5.

When you have a signal such as the one mentioned above, and you sample it, aliasing will cause the signal components will be replicated endlessly. These replica signals are the signal's aliases. The spacing between these aliases will be even and equal to the sampling rate. These aliases are indistinguishable from real signals spaced an integer number of sampling rates away: there is no way, once the signal is sampled, to know which parts of it are real and which parts are aliased. To compound our trouble, any real-world signal will have a power spectrum that's symmetrical around zero frequency, with negative frequency components; after sampling these components of the original signal will appear at frequencies that are lower than the sample rate.

Figure 3 on page 5shows this effect. The central frequency density plot is the signal that's being sampled; the others are the signal's aliases in sampled time. If you sampled this signal as shown, then after sampling the signal energy would appear to "fold back" at $1/2$ the sampling rate. This can be used to demonstrate part of the Nyquist-Shannon sampling theorem: if the original signal were band limited to $1/2$ the sampling rate then after aliasing there would be no overlapping energy, and thus no ambiguity caused by aliasing.

[4]This is possible to do when a system is *linear*, i.e. when it obeys the property of superposition.

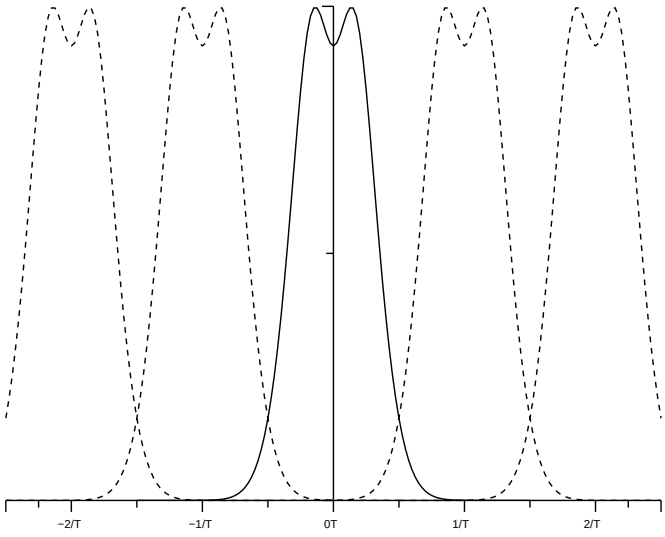


Figure 3: Aliasing of a signal's spectrum in the frequency domain.

## Reconstruction

The opposite process from sampling is reconstruction. The fundamental difference between continuous-time and sampled-time signals is that a continuous-time signal is defined everywhere in time, while a sampled-time signal is only defined at the sampling instants. Because a sampled-time signal isn't defined between samples it can't be used directly in a continuous-time system. To use a sampled-time signal in a continuous-time system it must be converted into a continuous-time signal before it can be useful. This conversion from sampled to continuous time is called reconstruction.

Reconstruction is done by interpolating the output signal. Interpolation is the process where the value of the continuous-time signals between samples is constructed from previous values of the discrete-time signal. In discrete-time control and signal processing systems the first step of this interpolation is almost universally done with digital-to-analog converters (DACs) which contain an intrinsic zero-order hold.

A zero order hold is simply a device which takes on the value of the most recent sample and holds it until the next sample time:

$$y\left(t\right) = y_{\left\lfloor \frac{t}{T} \right\rfloor} \tag{4}$$

where the function $\lfloor x \rfloor$ is simply the floor function as you might find in the C math library.

In a block diagram a zero-order hold is indicated by a block containing a picture of an interpolated signal, as shown in Figure 4 on page 6. This is exactly the function provided

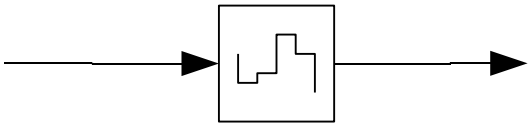

Figure 4: A sample-and-hold block.

by most digital-to-analog converters: the processor writes a number to the DAC, which drives a corresponding voltage (or current) output until the next number is written.

The result of using a zero-order hold is shown in Figure 5 on page 7. The sampled-time signal is at the top, and the interpolated signal is on the bottom. Note the "stair-step" nature of the interpolated signal, as well as the fact that on average it is delayed from the original signal[5].

Reconstruction causes a form of aliasing[6]: when a signal at a certain frequency is reconstructed the continuous-time signal will have components at the sampled-time signal frequency, as well as all multiples of the sample rate plus and minus the sampled-time signal frequency. For example, if the sampled-time signal is

$$y_k = \cos\left(2\pi\, f_0\, k\right) \tag{5}$$

(which only has two frequency components, at $\pm f_0$), then the reconstructed signal will have components at

$$f \in \pm f_0,\ \frac{1}{T_s} \pm f_0,\ \frac{2}{T_s} \pm f_0,\ \cdots \tag{6}$$

In addition to this effect, the zero-order hold interpolator acts as a low-pass filter for the signals going through it, with a frequency amplitude response of

$$A\left(f\right) = \operatorname{sinc}\left(\pi T_s, f\right) = \begin{cases} 1 & f = 0 \\ \frac{\sin(\pi T_s\, f)}{\pi T_s\, f} & \text{otherwise} \end{cases} \tag{7}$$

This means that while all of the components shown in (3) will be generated, the high-frequency components will be attenuated by the action of the zero order hold.

[5]Depending on your system requirements this little bit of delay may not matter at all, or it may break your system's error budget—but that discussion is beyond the scope of this article. See [Opp83, Wes06] for additional theory on this subject.

[6]Depending on your point of view, you can claim that reconstruction reflects the aliasing that was already there. This is the case if you unify the sampling and reconstruction processes with careful use of the Fourier transform and the Dirac delta functional. Such analysis is powerful, and can be a useful tool, but would require a lengthly digression (see [Opp83]). So for the purposes of this paper we will treat the sampled-time and continuous-time domains as complementary but separate, and we will treat reconstruction and sampling separately.

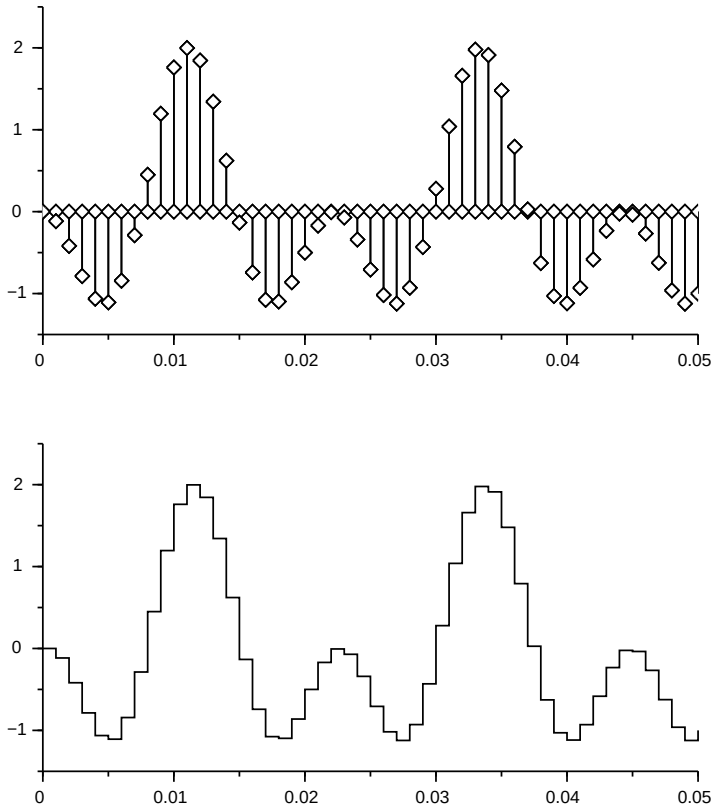


Figure 5: A signal, reconstructed.

# What Nyquist Didn't Say

The Nyquist-Shannon sampling theorem, with its limit on sampling rate vs. spectral content of the signal, does give you some clearly stated bounds to look for in your system design. But as we shall see, in practice these bounds aren't nearly as clear as they are in theory. So the Nyquist-Shannon theory appears on the surface to say things that in practice aren't true. What the Nyquist-Shannon sampling theorem—absolutely and positively—does not say, is that you can design your system to operate right at the Nyquist rate, at least not with any reasonable chance of success.

Here are some system design decisions that I have heard made using the Nyquist rate as a guiding light. Each one of these decisions is based on a faulty understanding of the Nyquist-Shannon sampling theorem. Some of these decisions are falsely optimistic, leading to a faulty system; some are falsely pessimistic, leading to an unnecessarily expensive system. Since it is a design engineer's job to make the most product out of the least stuff (and have the product work), neither of these design decisions end up being good ones. I'll list these design decisions here, then devote a section to each one explaining why it is based on faulty reasoning about the Nyquist limit.

1. "I am going to sample at 8kHz, so I need to use a filter with a 4kHz cutoff."
2. "I need to monitor the 60Hz power line, so I need to sample at 120Hz."
3. "My radio works at 5MHz, so I have to sample above 10MHz."
4. "We have a signal at 1kHz we need to detect, so I need to sample at 2kHz."
5. "What is the spectrum of an EKG signal? I want to figure out the Nyquist rate."
6. "My control loop samples at 8kHz, so I need to use a filter with a 1kHz cutoff."

## Nyquist and Filters

*"I am going to sample at 8kHz, so I need to use a filter with a 4kHz cutoff."*

Nyquist didn't say that if you are sampling at rate $N$ that you could use an anti-aliasing filter with a cutoff at frequency $f = {}^{N}/_{2}$.

Let's take a voice signal as an example. Assume that we want to sample the human voice, and furthermore that we want to sample it for a communications application—so while we care deeply about the intelligibility of the sampled signal, we aren't looking for the kind of quality you'd get in a musical recording.

If you were to start your effort with the grand old engineering tradition of copying something that works, you would be well served to copy the sampling rate and encoding that's used to transmit telephone conversations digitally. Land line telephone conversations are

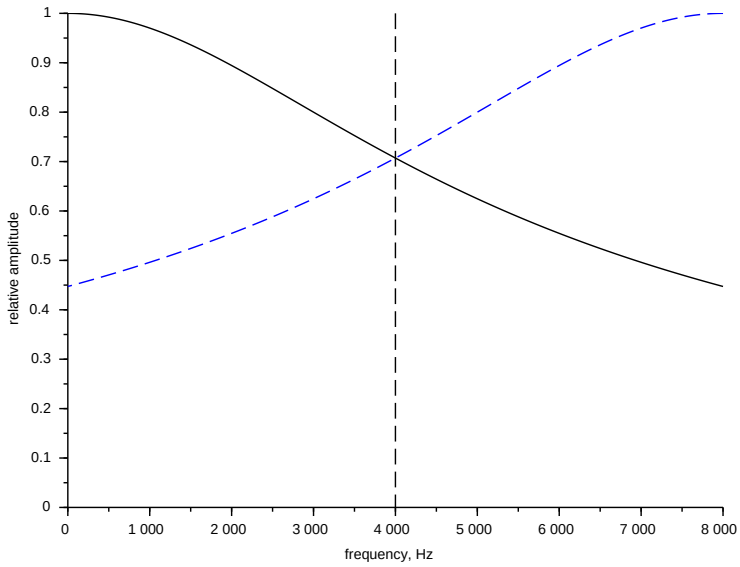


Figure 6: Sampling with a cutoff frequency of $1/T$.

sampled at 8000 samples per second, and are transmitted digitally with an effective precision of 12 bits[7] or so. We'd like to be able to sample the signal such that the energy in the aliased signal is insignificant. I'll choose 40dB (that's a factor of 1:10000 of energy) for an aliased signal energy, assuming that the spectrum of the sound hitting the microphone is even across all frequencies.

Possibly the worst thing you could do in this case would be to design a system where you go straight from a microphone to a sampler or ADC. In this case all of the energy of the incoming speech—and any background noise—that occurs above 4kHz will be aliased back into your sampled signal; the result of such a sound acquisition system would sound terrible.

So what does happen if you use a filter with a 4kHz cutoff, along with your 8kHz sampling rate? Figure 6 on page 9 shows what happens when we use a 4kHz 1st-order low pass filter. The solid curve shows the spectrum of the filtered audio to the ADC, while the dashed curve shows the aliased audio. You can see that this is not going to give good performance. In fact, if you take all the aliased energy into account the ratio between the correct signal and the aliased signal is exactly 1:1! That's not good at all.

Can you fix this problem by increasing the filter order? Not really. Figure 7 on page 10 shows the result of using a 6th-order Butterworth low pass filter instead of a 1st-order filter. We've improved the situation – we now have better than a 50:1 signal energy to alias ratio, and the aliasing is concentrated in the upper end of audio band where it presumably won't sound as bad. But the alias to signal ratio isn't even close to the 40dB that we'd like

[7] Capturing the meaning of "effective precision" gets smoky here—if you want to know how this operates in detail, search on "Mu-law companding" and "A-law companding".

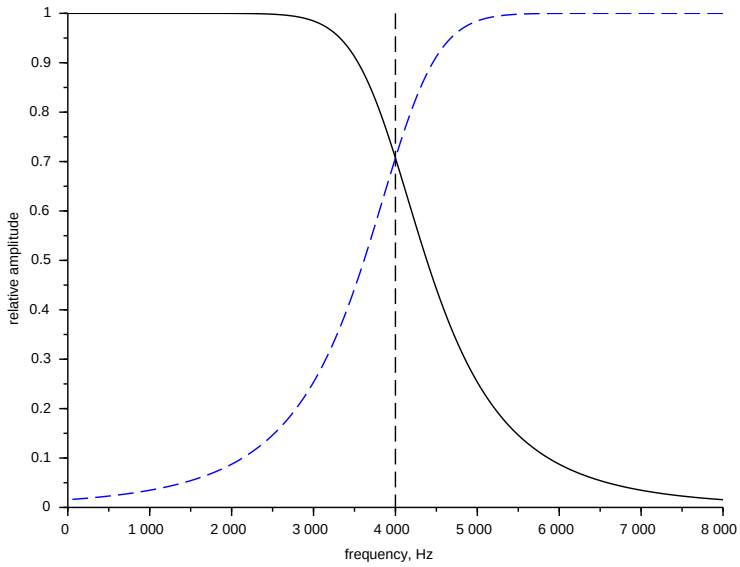


Figure 7: Sampling with a sharper filter.

to see. The situation does improve as the filter order gets higher, but you'd have to go to a 3500 pole filter to get a signal to alias ratio greater than 40dB. Clearly this is an absurdly large filter to realize in analog hardware.

What is wrong here? The biggest failure in our logic is that we read much too much into the term "cutoff frequency" of our filter. Analog filters are characterized by cutoff frequencies, but the response of the filter does not stop cleanly at the cutoff frequency. Instead, the term "cutoff frequency" usually means the frequency at which the filter is attenuating the signal by 3dB. This means that for all but the sharpest of filters there is a significant amount of filter response "left" outside of a filter's defined passband.

Since we're talking about telephone-quality communications, let's ask what the telephone guys do. A speech signal such as gets transmitted over a telephone doesn't have to extend all the way up to 4kHz. In fact, a good rule of thumb for intelligible speech transmission is that you need to transmit frequencies up to 3kHz. So let's continue to sample the system at 8kHz, but let's only try to keep the components of the audio signal that range in frequency from 0 to 3kHz. Figure 8 on page 11 shows the situation with this new sampling rate and the same 10-pole filter as in Figure 7 on page 10. This gets us a system with an alias to signal power ratio of over 16000:1.

So what does this mean?

When you're designing your anti-alias filter it means that you simply cannot set the cutoff frequency to the Nyquist rate. Moreover, unless you're working from a cookbook—and within that cookbook's limitations—you can't set it to any easy "rule of thumb" fraction of the Nyquist rate. Instead, you'll need to pay attention to the required precision of the conversion, and specify your filter accordingly.

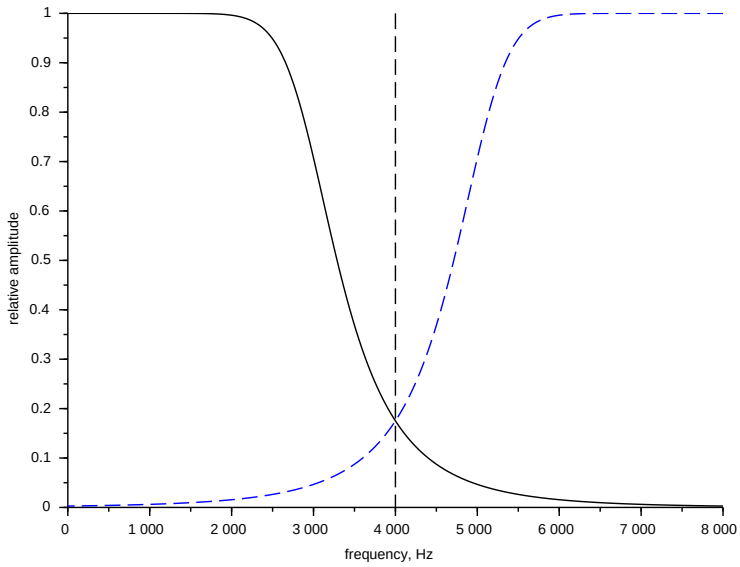


Figure 8: Filtering with a lower cutoff.

If you have the freedom to set the sampling rate, this filtering effect means that you'll need to make a trade-off between a low sampling rate and the complexity of your anti-alias filtering. It is not uncommon for systems to have sampling rates that are seemingly quite high, just to simplify the task of designing the anti-alias filters[8].

## Nyquist and Signal Content

*"I need to monitor the 60Hz power line, so I need to sample at 120Hz."*

*"We have a signal at 1kHz we need to detect, so I need to sample at 2kHz."*

Nyquist didn't say that a signal that repeats N times a second has a bandwidth of N Hertz. For example, you're designing a device to monitor a power line and collect statistics about the quality of the power being delivered. You know that the power grid in North America operates at 60Hz – this means that you can do an adequate job of monitoring your power line if you sample at 120Hz, right?

Wrong. Figure 9 on page 12 shows a snapshot that I collected of the line voltage at my house. It's probably typical of the power that you'll see in North America: it's nominally

[8] In fact, many systems today collect the information at one (high) sampling rate, then perform further antialias filtering in the digital domain. This allows the data to be further filtered in the digital domain (where sharp, accurate filters are easy), then sampled ("decimated") to an even lower sampling rate for storage or further processing.

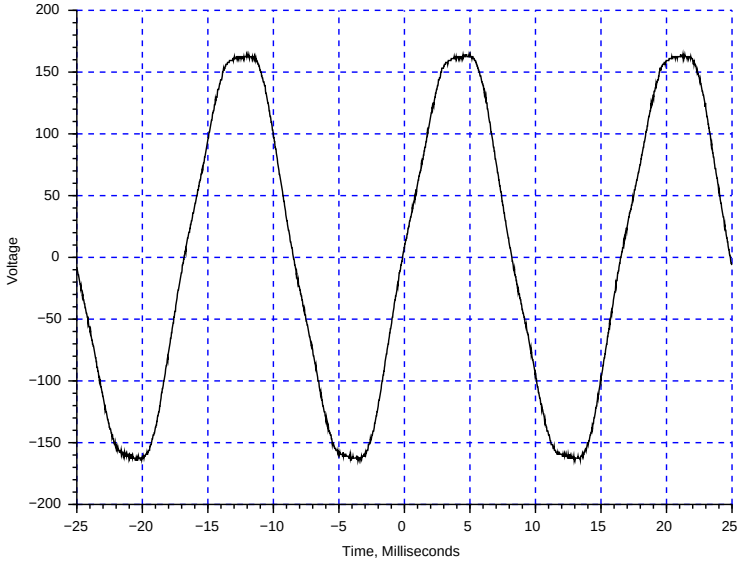


Figure 9: Example power line voltage.

60Hz 117V, but the waveform isn't a perfect sine wave. Looking at the waveform, you can see the distortion, with the slightly flattened peaks. It turns out that the waveform shown has significant harmonic content up to the 5th harmonic (300Hz) if you ignore the "fuzz" from noise. Moreover, I was having an exceptionally good day as far as power line noise goes[9]. So for this wave, 120Hz is *not* the Nyquist rate, and sampling at anything under 600Hz will cause you to miss the detail that was there that day, and even more detail on a bad day.

But can you catch that 5th harmonic by sampling at 600Hz? Figure 10 on page 13 shows two possible results of sampling a 300Hz signal at 600 Hz. The filled diamonds show what happens if you sample right at the peaks of the signal being sampled, while the open diamonds show the signal being sampled at its zero-crossings. What does this mean? It means that unless you happen to know the phase of the signal that you're sampling, you cannot sample right at the Nyquist rate and get a meaningful answer. In fact, this means that you couldn't have sampled a pure 60Hz sine wave at 120Hz!

What is necessary to sample this signal? Let's assume a few things about our signal, and see what we need to do to sample it adequately. First, we'll assume that the only interesting signal content is at the 5th harmonic and lower. Second we'll assume that there is no significant noise or signal above the 5th harmonic. Third, we'll assume that we want to make an accurate measurement in just one cycle.

[9]Normally, power line noise can extend far, far beyond the 5th harmonic. Had I plugged a lamp dimmer into the same outlet that I was monitoring, there would have been a huge spike in the waveform at 60 or 120Hz, with harmonics extending up to radio frequencies. Accurate power line measurement is not trivial.

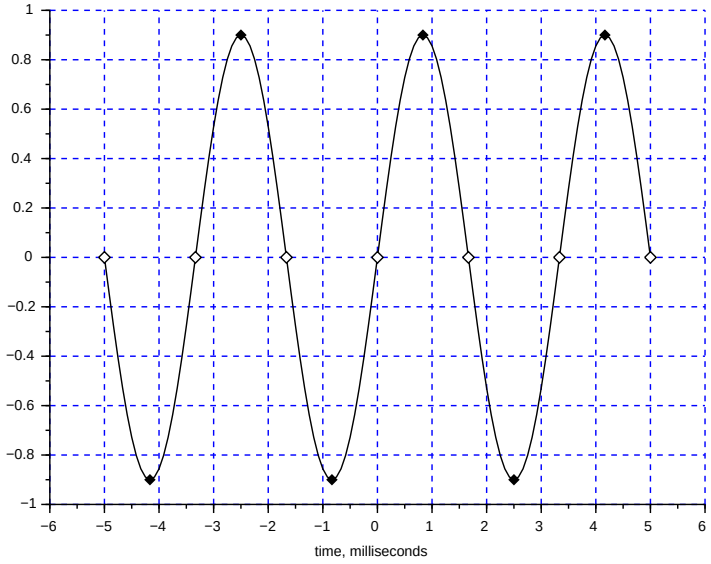


Figure 10: Sampling at exactly the Nyquist rate.

We know that we can't sample right at the Nyquist rate, that we have to sample higher. It turns out that to meet the requirement of sampling in just one cycle of our 60Hz wave, the minimum frequency that we could sample at would be 660Hz, or the harmonic's Nyquist rate plus the reciprocal of our observation interval. This rate would catch the 5th harmonic adequately for mathematicians. Because we're assuming a negligible amount of noise, then we can conclude that no anti-alias filter is necessary or wanted.

But the sampling would be barely adequate. Sampling at 660Hz would just give you the bare minimum of information you needed to reconstruct that 300Hz, 5th harmonic signal, but it would take some work, and you'd have make some strong assumptions about signal content above 300Hz. A quick look at Figure 11 on page 14 shows the signal you want to capture, and what you'd actually get. Moreover, the math would be weird (who wants to deal with all those factors of 11?). A better rate to use would be 900Hz, or even 1200, which gives you a 2-times overhead of your 5th harmonic.

## Nyquist and Repetitive Signals

*"OK Mr. Smarty-pants, I need to monitor the 60Hz power line, so I guess I need to sample at well over 120Hz."*

Nyquist doesn't say that a signal that repeats N times a second has a total occupied spectrum greater than N Hertz.

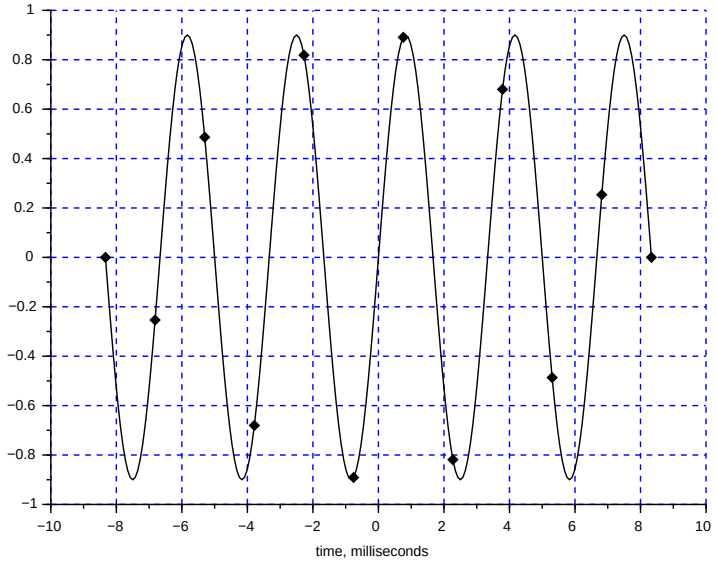


Figure 11: Barely adequate sampling.

The Nyquist-Shannon sampling theorem isn't always misinterpreted too optimistically. Sometimes it is misinterpreted too pessimistically. Indeed, there are times when a signal can be sampled slower—much slower—than a superficial reading of the Nyquist-Shannon sampling theorem would indicate. This doesn't happen as often as we'd like, but when it does you can make significant reductions in your system cost through intelligent use of sampling.

Let's look at our power line sampling problem again. Let's assume that we're using a really cost-effective processor, and add the constraint that we can't sample the wave any faster than 20Hz. Now what?

If the signal is at exactly 60Hz, and we sample it at exactly 20Hz, then we'll sample a point in the cycle, then wait two cycles, then sample the *same* point within the cycle, over and over again. What to do? Are we doomed? No—if we sample the signal at a frequency slightly *lower* than 20Hz, the signal phase within a cycle will advance a little bit with each new sample. With time, the effect will be a replica of the original signal, only slower.

In fact, you can almost arbitrarily choose how fast you want to sample, and how much detail you pick up. Let $M$, $P$, and $N$ be integers, with $M \geq 0$, $P \neq 0$, $N > 0$, and with $P$ and $N$ having no common factors. Then we can choose a sampling interval

$$T_s = (M + {}^{P}/_{N}) \frac{1}{F} \tag{8}$$

where $F$ is the frequency of our repeating signal (60Hz in this case). Doing so, we are guaranteed that after $N$ samples, we will have sampled $N$ evenly spaced points on the cycle. The only fly in the ointment is that for $P \neq 1$ the points will be out of order—but that's a problem that's reparable in software.

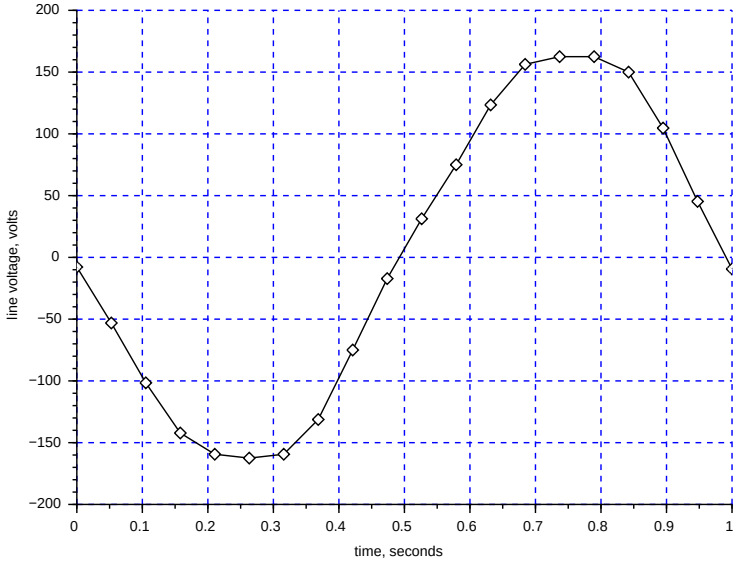


Figure 12: Example power line signal, sub-sampled.

Notice that (8) can represent "straight" sampling: by choosing $M = 0$ and $P = 1$, (8) becomes $T_s = 1/N F$, and we sample $N$ points in each cycle of our parent signal.

If we don't mind contending with odd frequencies, we can start by choosing $N$ for the number of samples we want in the waveform, then adjusting $M$ until we get a sampling rate that's an appropriate speed for our processor.

For instance, if we choose to sample the 60Hz waveform in Figure 9 on page 12 at 19Hz, then we can find that $M = 3$, $P = 3$, and $N = 19$. Thus, we should be able to build up a picture of the 60Hz power line voltage in 19 samples, or just under one second. The correct order of the $k^{th}$ sample in this sample set will be

$$n_k = (k\,P) \bmod N \tag{9}$$

or, for $k = [0, 1, 2, \cdots, 19]$, the indexes of the reordered vector should be

$$n_k = [0, 3, 6, 9, 12, 15, 18, 2, 5, 8, 11, 14, 17, 1, 4, 7, 10, 13, 16] \tag{10}$$

Figure 12 on page 15 shows the waveform in Figure 9 on page 12 after it's been sampled at 19Hz and reordered according to (9). The resulting waveform repeats at slightly less than 1Hz instead of 60, but it replicates the original waveform as faithfully as sampling it at 1140Hz ($19 \cdot 60\text{Hz}$) would have[10].

How can this work? Why is it that we can apparently violate Nyquist's rule so thoroughly, and still get good results? The reason is in the behavior of the power line voltage—it is

[10] It is interesting to note that when we are sampling at full speed our sampling rate still adheres to (8)—just with $M = 0$ and $P = 1$.

strictly cyclical, which means that we can count on it repeating over and over again. We've already described this in the time domain—how each new sample lands a bit farther ahead on the cycle where it falls than the previous sample did. If we try to think about this in the frequency domain we're still left with an apparent contradiction, however.

What is happening in the frequency domain is the result of a quirk in the Nyquist-Shannon sampling theorem, and a matching quirk in the frequency domain behavior of repetitive signals. These quirks work together to let you sub-sample truly repetitive signals and build up a picture of the waveform, even if your sampling rate is much lower than the fundamental frequency of the signal.

The quirk in the frequency domain is that a steady, repetitive signal has its energy concentrated in spikes around the fundamental frequency and its harmonics. The more steady the cyclical waveform is, the narrower these spikes are. So a signal that varies over a span of a few seconds will have spikes that are around 1Hz wide, while a signal that varies over a span of a few minutes will have spikes that are only a small fraction of 1Hz wide.

The quirk in the Nyquist-Shannon sampling theorem is that the spectrum of the signal doesn't have to be contiguous – as long as you can chop it up into unambiguous pieces, you can spread that Nyquist bandwidth into as many thin little pieces as you want. Sampling at some frequency that is equal to the repetition rate divided by a prime number will automatically stack these narrow bits of signal spectrum right up in the same order that they were in the original signal, only jammed much closer together in frequency – which is the roundabout frequency-domain way of saying that you can sample at just the right rate, and interpret the resulting signal as a slowed-down replica of the input waveform.

So in the end, we are taking advantage of these quirks to reconstruct the original waveform, even though our sampling is otherwise outrageously slow.

There is a huge caveat to the foregoing approach, however: this approach only works when the signal is steady, with a known and steady frequency. In the example I'm using it to analyze the voltage of the North American power grid. This works well because in North America, as in most of the developed world, you can count on the line frequency to be rock solid at the local standard frequency. This approach would not work well at all on any unstable power source, such as one might see out of a portable generator or from a power line in a less well developed area.

Even in a developed country, this approach has the drawback that it simply will not accurately catch transient events, or rapidly changing line levels. Transient events (such as someone switching on a hair dryer) have their own spectra that adds to the series of spikes of the basic power; rapidly changing line levels act to spread out the spikes in the power spectra, as does changing line frequency. Any of these phenomena can completely destroy the accuracy of the measurement.

The bottom line is that things aren't always as black as your reading may paint them—but you have to interpret your theorem carefully to make sure you don't fall into a trap.

## Nyquist and Band Limited Signals

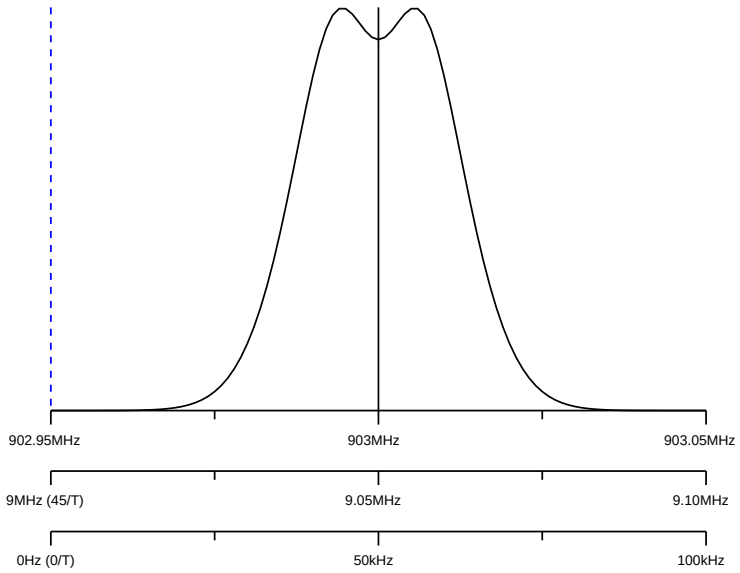


Figure 13: RF Signal, Translated.

*"My radio works at 5MHz, so I have to sample well above 10MHz."*

Nyquist didn't say that a signal centered around some high frequency has a bandwidth equal to that frequency. In a manner related to the power line sampling above, one can sample signals whose spectrum is centered around a frequency significantly higher than the sampling rate, as long as the signal in question is limited in bandwidth.

As an example, consider a radio receiver that must work with signals with a 50kHz bandwidth. A common architecture for such receivers is the superheterodyne, which translates them to some intermediate frequency in the megahertz region, filters them to the desired bandwidth, then sample them an adequate rate (200kHz at a minimum, in this case).

For example, consider a data link that generates a signal that is 50kHz wide, centered on 903MHz. This signal gets translated down by exactly 893.95MHz and filtered, so it is now centered on 9.05MHz and we can trust that there are no significant interfering signals. Then, the signal gets sampled at exactly 200kHz.

Figure 13 on page 17 shows how this signal as it occurs in each of the three places above—only the frequency axis changes. In the original, it is centered on 903MHz (top axis). Once it is translated down by 893.95MHz, it forms an intermediate frequency (IF) signal centered on 9.05MHz (middle axis). When this IF signal is sampled[11] at 200kHz the signal

[11] If you set out to build a sub-sampled receiver of this type, note that you may have to take on the responsibility of building the sampler: an ADC that samples at 200kHz may work fine with a signal at 50kHz, but the chip's internal sample-and-hold circuit may not be good enough to sample accurately at 9MHz.

will alias, and the content at 9.05MHz will show up at 50kHz, with the overall signal aliased down to a spectrum that runs from approximately 25kHz to 75kHz. This spectrum is well within the Nyquist rate of 100kHz, and the entire signal is easily encompassed by the available bandwidth of the sampled-time system.

# A Design Approach

So far in this discussion I have tried my best to destroy some commonly misused rules of thumb. I haven't left any rules of thumb in their wake. Why? Because solving problems in sampling and anti-alias filtering is not amenable to rules of thumb, at least not general ones. When you're solving sampling problems you're best off working from first principals and solving the problem by analysis.

In designing sampled-time systems, the variables that we need to juggle are signal accuracy (or fidelity) and various kinds of system cost (dollar cost, power consumption, size, etc.). Measured purely from a sample rate perspective, increasing the signal sample rate will always increase the signal fidelity. It will often decrease the cost of any analog anti-aliasing and reconstruction filters, but it will always increase the cost of the system digital hardware, which will not only have to do its computations faster, but which will need to operate on more data. Establishing a system sample rate[12] can be a complex juggling act, but you can go at it systematically.

In general when I am faced with a decision about sampling rate I try to estimate the minimum sampling rate that I will need while keeping my analog electronics within budget, then estimate the maximum sampling rate I can get away with while keeping the digital electronics within budget. If all goes well then the analog-imposed minimum sample rate will be lower than the digital-imposed maximum sample rate. If all doesn't go well then I need to revisit to my original assumptions about my requirements, or I will need to get clever about how I implement my system.

The common criteria for specifying a sampled-time system's response to an input signal are, in increasing order of difficulty: that the output signal's amplitude spectrum should match the input signal's spectrum to some desired accuracy over some frequency band; that the output signal's time-domain properties should match the input signal's time-domain properties to some desired accuracy, but with an allowable time shift; and that the output signal's time domain properties should match the input signal to some desired accuracy, in absolute real time.

## Designing for Amplitude Response

In many cases where one is sampling an audio signal one only needs to concern oneself with the amplitude response of the system over frequency. Indeed, this was the case with

[12] When you can—sometimes a system's sampling rate is determined by circumstances such as a sensor or channel that has its own natural sampling rate. In such cases you are reduced to making sure that the system will work at all given the limitations, and if it will, how you will make it do so.

the example presented in the section titled "Nyquist and Filters", above.

To design to an amplitude response specification you need to determine what range of frequencies you want to pass through your system, and how much amplitude variation you can stand from the original (this can vary quite a bit and still be acceptable). Then you must determine how much off-frequency signal you can stand being aliased down into your desired signal—you can usually take this to be equal to the least-significant bit of your ADC, although you can often allow more, and in rare cases you may need to allow much less. To determine how much attenuation you need at these off frequencies you must have an idea of how much signal is there to be filtered out—it is usually conservative to assume that the entire signal is spectrally flat, and plan your filter accordingly.

## Designing for Time-Shape Response

*"What is the spectrum of an EKG signal? I want to figure out the Nyquist rate."*

Nyquist didn't say that his theorem would find you the best solution in all cases. Sometimes you have to abandon the frequency-domain approach of the Nyquist-Shannon sampling theorem, and analyze things in the time domain.

For example, there is a problem that arises when one designs for a pure frequency vs. amplitude response. This problem is that the signal gets distorted in time.

For some applications this matters little, if at all. The human ear is remarkably insensitive to time distortions in a signal, as long as the overall frequency content gets passed correctly. Audiophiles may argue this point, but certainly for the purposes of verbal communication and casual home entertainment quite a bit of time distortion is acceptable.

For other applications it is important that the signal, after processing, appear to match the original signal as closely as possible. This requirement is particularly acute for applications that involve vision (human or machine) or that require pattern matching – in these cases a signal that is distorted in time can be critically wrong.

Figure 14 on page 20 shows what happens if you pass a rectangular pulse through the simple Butterworth anti-alias filter that we developed in the telephony example. This filter's frequency response is shown in Figure 8 on page 11. There are two salient things to note about this response: one, it is considerably delayed in time, and two, it shows moderately bad ringing. If you were building a system that needed to accurately portray the shape of a pulse this filter could be a disaster[13].

The filter in Figure 14 on page 20 isn't necessarily entirely as bad as it looks: often when we are storing information for later display we can absorb any filter delay into the overall delay of the system. This is shown in Figure 15 on page 20, where the filter response is shown artificially shifted back in time. This gives a better looking response and a much

[13]If you needed to transmit a pulse with significantly less delay it would be a disaster twice—we'll cover that case in a later section.

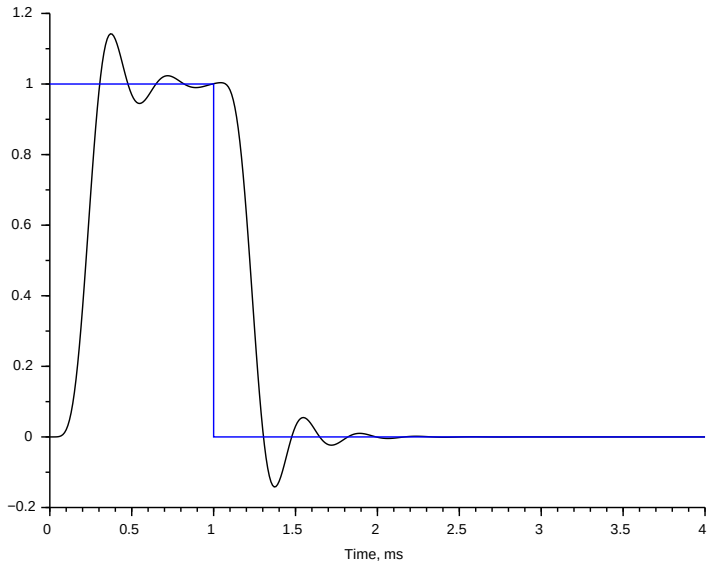


Figure 14: Butterworth Pulse Response.

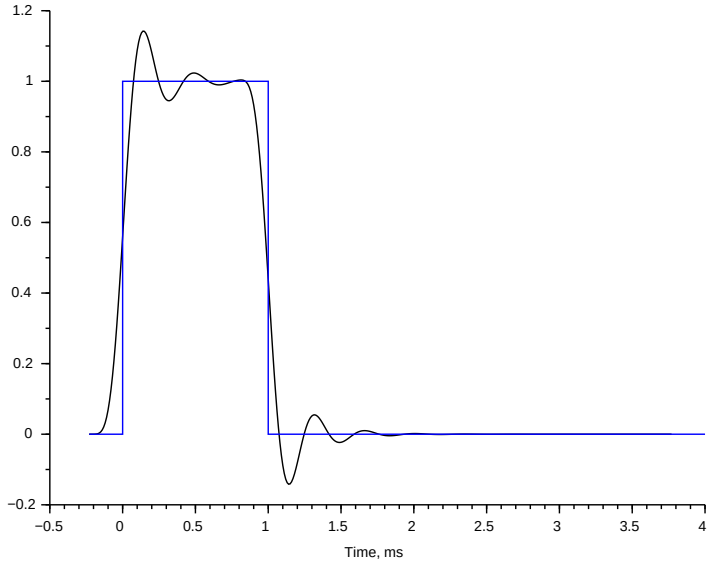


Figure 15: Delayed Butterworth Pulse Response.

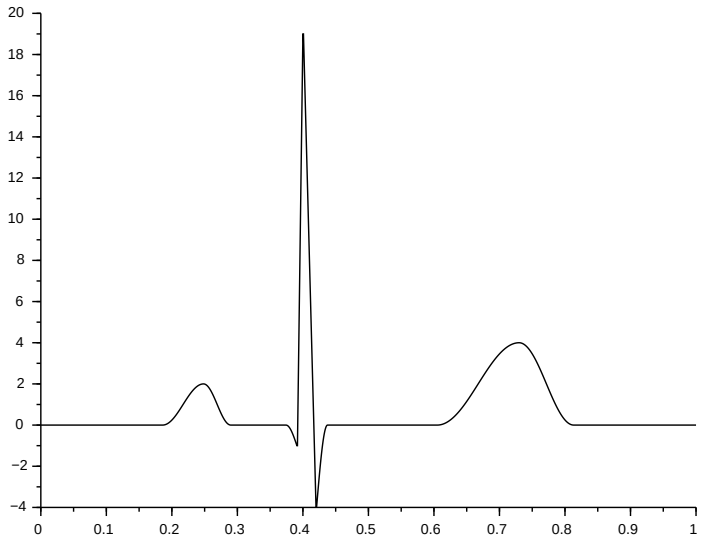


Figure 16: An EKG signal.

better fit. The ringing is still present and would still be objectionable in many applications, but for other applications this may be good enough.

The nature of the data being measured can make a tremendous difference to the necessary filter response, however.

For systems such as electrocardiogram displays, video systems, and pattern matching systems, it is very important that the filter response doesn't ring. It can be very difficult to tell the difference between ringing artifacts and real signal features. Figure 16 on page 21 shows an example EKG waveform[14], in an unfiltered state. If you were to take the Fourier transform of this signal, you would find that it has little energy above 100Hz. This would seem to indicate that it is quite safe to filter it with a nice sharp filter that has a cutoff above 100Hz. Figure 17 on page 22 shows the result of filtering the same signal through a rather aggressively designed filter[15] with a 150Hz cutoff frequency. The spike in the EKG is distorted—its amplitude has been enhanced, and it rings slightly. Clearly this filter has compromised the integrity of the data.

Why did this happen? The anti-alias filter in this case is designed to have a very aggressive frequency response, with the great attenuation with the least filter complexity. This can look good when you just consider the frequency-domain effects of the filter, but can have bad consequences when the time-domain features of the signal must be analyzed.

There are two effects at work to create this time-domain distortion. One of these is just the sharp transition in the frequency domain; the other effect arises from different components of the signal being delayed by different amounts. You'll see it referred to as "non-linear

[14] If you know how to read an EKG—relax, this one's fake.
[15] a $4^{th}$-order Chebychev filter with about 6dB of passband ripple

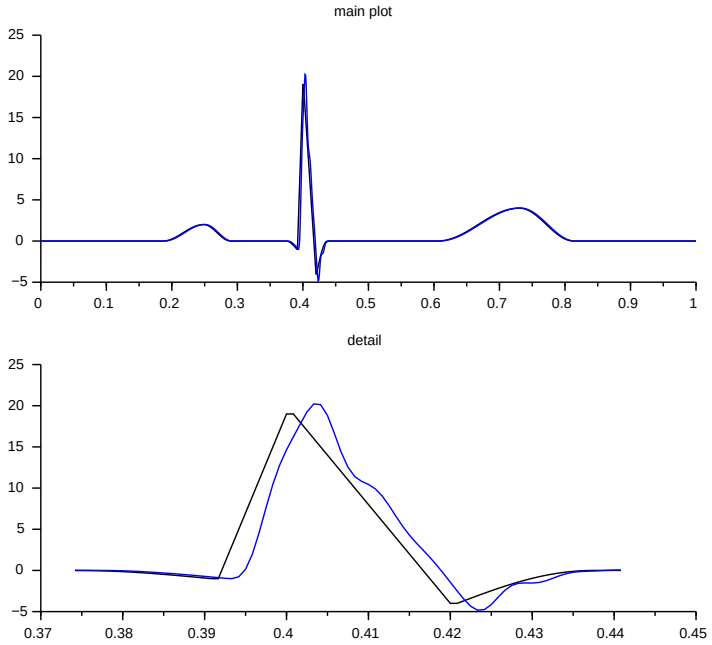


Figure 17: An EKG signal, filtered aggressively.

phase", or "non-linear phase delay". At any rate, to minimize this effect you need to select a filter that minimizes nonlinear phase delay effects.

Unfortunately, while digital filters with linear phase delay effects are easy to design such is not the case with analog filters. Simple analog filters that minimize nonlinear phase delay effects don't have good frequency-domain properties[16]. Analog filters can be designed with good frequency-domain properties and minimal nonlinear phase delay properties, but this requires special compensation methods that increase the filter complexity and cost[17]. Absent these special methods, the best analog filter for minimizing phase delay is the Bessel filter ([Bud74]).

Figure 18 on page 23 shows the results of filtering the sample EKG with a Bessel filter. The ringing is entirely eliminated, but you can see that the peak of the waveform is still somewhat attenuated. Such filtering may or may not be acceptable in this situation—there is obvious distortion of the waveform, but should there be noise on the signal it will be reduced.

What, then, do we do? One obvious solution is that if you are working with an application that cannot tolerate these effects is to sample your signal faster. This will allow you to use a more open anti-aliasing filter. Another solution, perhaps not so obvious, is to sample at the rate that you can stand, and increase the anti-alias filter bandwidth above what might seem "right".

[16]Techniques exist to compensate for analog domain phase effects in the digital domain, but their use is beyond the scope of this article – see [Pup06].

[17]Basically one incorporates unstable zeros into one's filter. Doing so allows for better phase linearity while preserving good amplitude characteristics, but at the cost of higher component count and increased sensitivity to component variations. See [WikiMin].

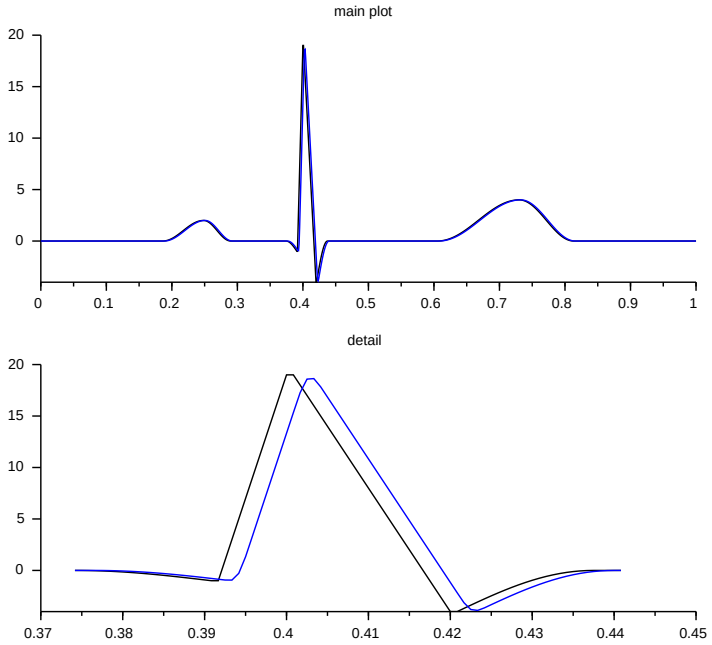


Figure 18: EKG filtered with a Bessel filter.

One last option—one that goes directly against the grain of the "DSP-101" knowledge base—is to leave the anti-aliasing filter off entirely. Yet, for many applications it is the best answer. To determine if you can ease up on your anti-alias filter requirements you need to ask yourself (a) if there is, indeed, high-frequency interference to your signal, or if it the signal itself has high-frequency components and (b) if such high-frequency components will cause more signal degradation than an anti-alias filter. If the answer to both of these questions is "no" then you should seriously consider not including a strict anti-alias filter in your design.

## Designing for Absolute Time Response

*"My control loop samples at 8kHz, so I need to use a filter with a 1kHz cutoff."*

Nyquist didn't say that aliasing was the worst thing that could happen to your system. The previous section was concerned with signals that must retain the general shape of the original after sampling, but which could stand an arbitrary amount of delay. Many recording and data-analysis tasks fall into that category.

Another category of signal sampling tasks, however, involves real-time control of systems[18]. In this case we are wrestling with the dynamics of some thing that we're trying to control, and chances are we're trying to speed up its response to commands while minimizing

[18]See [Wes06] for detail on sampled time control system design, including the effects of sampling on system modeling and performance.

its response to undesired input. In these cases not only must we be concerned with the shape of the signal after sampling, but we must be concerned with the amount of delay we introduce into the signal path.

When one is designing a controller one is very interested in the amount of phase delay one that controller introduces into the signal path. This phase delay is often the single most important factor that limits the overall performance of the control loop, so one generally puts a great deal of energy into minimizing it.

In any control loop that can be described using frequency-domain methods, we find that there is a critical frequency at the point where the phase shift due to the dynamics of the system being controlled, plus the dynamics of the controller, reaches 180 degrees. Should the gain of the system at this frequency be equal to one, then the system will oscillate, which will degrade performance if it doesn't destroy the system outright.

In a sampled-time control loop, the time delay due to sampling is approximately ½ the sampling rate. As a consequence, the phase delay due to sampling is approximately 360 f Ts/2 degrees, where f is the frequency of the signal and Ts is the sample interval. This means that the phase delay contribution from sampling alone is 180 degrees at f = Ts.

In my classes I try to present easy rules of thumb where I can. The rule of thumb for sampled-time control systems is to choose a sampling rate equal to ten or twenty times your desired system bandwidth. Using 10x oversampling the phase delay at the system bandwidth is

$$\theta_{delay} = (360)\,\frac{T_s}{20T_s} = 18^\circ \tag{11}$$

What about anti-alias filters in our control system? Any anti-aliasing filter you can use will add its own delay, and that delay can be considerable. Additionally, aliasing in a control system is generally benign, unless the resulting aliased signal has a frequency that is within the bandwidth of the control system. As long as the system doesn't put much noise on the feedback signal, the best anti-alias filter for a control system is the one you don't use.

If you do have noise in your system that could potentially alias down to a low frequency and cause you trouble, you should analyze your system to find out how much delay you can stand, and design your filter (and sampling rate) accordingly.

# Finally

The theme of this paper can be summed up to this: the Nyquist rate isn't a line in the sand that you can toe up to with complete safety. It is more like an electric fence or a hot poker; something that won't hurt you if you keep your distance, but never something you want to saunter up to and lean against.

So you should be aware of the Nyquist rate when you're designing systems. But to really determine an appropriate sampling rate for a system, or to determine the necessary anti-alias and reconstruction filters for a system, you have to understand aliasing and filtering. You have to know what aliasing is, how you can avoid it, and even whether avoiding it is the best answer for the system at hand.

## References

[Opp83] Alan V. Oppenheim, Alan S. Willsky, Ian T. Young: "Signals and Systems". Prentice-Hall, 1983

[Lyons04] Richard Lyons: "Understanding Digital Signal Processing". Prentice Hall, 2004

[Bud74] Aram Budak: "Passive and Active Network Analysis and Synthesis". Houghton Mifflin, 1974

[Wes06] Tim Wescott: "Applied Control Theory for Embedded Systems". Elsevier, 2006

[Pup06] "Group Delay and its Impact on Serial Data Transmission and Testing", Peter J. Pupalaikis, DesignCon 2006, <http://www.lecroy.com/tm/Library/WhitePapers/PDF/Group_Delay-DesignCon2006.pdf>.

[WikiMin] "Minimum Phase" Wikipedia 2010. Wikimedia Foundation, Inc.. 28 November 2010 <http://en.wikipedia.org/wiki/Minimum_phase>.

## About the Author

Tim Wescott has 20 years of experience in industry, designing and implementing algorithms and systems for digital signal processing and closed loop servo control. His extensive practical experience in translating concepts from the highly abstract domain of mathematical systems analysis into working hardware gives him a unique ability to make these concepts clear for the working engineer.

## Wescott Design Services

Wescott Design Services provides design and analysis help for companies with problems in control systems, communications systems, and embedded systems design. We specialize in solving difficult problems in embedded control and in detection and estimation problems. With our expertise in understanding the mathematics that underlie modern control and communications theory, and our ability in designing practical circuits and software, we can help you realize your goals quickly, efficiently, and successfully.

http://www.wescottdesign.com

## *Applied Control Theory for Embedded Systems*

This book is written for the practicing engineer who needs to develop working control systems without going back to school for years. It is aimed directly at software engineers who are learning control theory for the first time, however it has found favor engineers of all stripes when they are tasked with implementing control loops. It teaches theory, but it also covers many real-world issues in much greater depth than is taught in the sort of theory course you'd find in many universities.

*Applied Control Theory for Embedded Systems* is published by Elsevier. Learn more about the book an the author's own web page, http://www.wescottdesign.com/actfes/actfes.html, and from the publisher's page on the book:
http://www.elsevier.com/wps/find/bookdescription.cws_home/707797/description#description