a DAC to hit a desired voltage with minimal error, the INL and gain-error terms will dominate, and you don't care about monotonicity. If, however, you're closing a control loop, it's the exact opposite: the servoing action of the loop will remove the INL, but large DNL could cause hidden zones of instability, which are particularly hard to debug.

A DAC's architecture influences the mix of INL/DNL. Consider two good 16-bit DACs, the DAC8564 and the AD5544. The first uses a string of resistors, so you have to work pretty hard to end up with a DNL greater than 1 LSB. And you are *guaranteed* monotonicity. However, nothing is controlling the INL except the statistical distribution of the resistor values, so it's not surprising that the INL is ± 8 LSBs, and that's for the expensive part. It's 12 LSBs in the bleachers.

In contrast, in the *R*–2*R* architecture a large INL will often translate to a large DNL; the same process that keeps the DNL under control also keeps the INL down, to some extent, so the INL spec of the AD5544 is ± 4 LSBs, with a DNL of 1.5 LSBs. So all other specs being equal (which they aren't), one would choose the AD5544 for precisely setting voltages, and the DAC8564 for use in a control loop.

And, while we're issuing warnings, beware, Beware, *BEWARE* using audio DACs in non-audio applications. If a DAC doesn't provide a DNL spec, it's because it's embarrassingly large. This is often acceptable in audio, but not for either control loop use or voltage setting. Likewise, gain-drift specs for audio DACs are often too large for use in voltage-setting applications.

13.5 Analog-to-digital converters

Look back at the earlier section on "Preliminaries" (§13.1) to remind yourself of some of the things to think about when choosing a converter (whether DAC or ADC). At the top level you're concerned much less with the details of how the thing actually does its conversion, and much more with the major questions of (a) performance (speed, accuracy, etc.), (b) digital interface (parallel or serial; single-ended or LVDS; etc) and (c) integration (single or multiple units; stand-alone, or integrated into a microcontroller or other complex function). In most cases you'll use a commercial ADC chip or module rather than building your own. But it's important to know about the inner workings of the various A/D conversion methods, in order not to be caught unaware by their idiosyncrasies.

13.5.1 Digitizing: aliasing, sampling rate, and sampling depth

We'll get into the nitty-gritty of analog-to-digital conversion soon enough, but first a short riff on the business of *sampling*, which will come up again and again as we visit various ADC methods.

When you convert an analog signal (e.g., an audio waveform) to a series of digital quantities (i.e., numbers corresponding to the instantaneous voltage at successive moments in time), you need to choose both the precision of the voltage measurements (the sampling *depth*) and the rate at which such samples are taken (the sampling *rate*). We saw this briefly in Chapter 6, in connection with anti-alias low-pass filters (§6.3.7A); let's look a bit more deeply here, in the context of ADC sampling of analog waveforms.

A. Sampling depth

Let's look first at the effects of bit depth (because they are the more easily understood): sampling to n equally spaced bits quantizes the waveform samples to 2^n levels, effectively limiting the dynamic range to $6n$ dB. A waveform so sampled, when properly scaled to exploit the full conversion range, will also exhibit quantization distortion, to the tune of 2^{-n} (i.e., $100/2^n$ percent).

As an example, 16-bit quantization of audio (the standard used in CD audio) has a dynamic range limited to 96 dB, and minimum distortion of 0.0015%. Of course, the signal itself is typically limited in both dynamic range and distortion; a well designed digitizing system should have sufficient bit depth (and sampling rate) so that it does not degrade the signal's quality.

At a deeper level there is more to the story than mere bit depth: nonlinearity (even nonmonotonicity!), noise, spurs, etc., all contribute to the fidelity of the digitized signal. A common metric that captures much of this is "ENOB" (effective number of bits); we'll see more of this later (see for example Figure 13.56).

B. Sampling rate and filtering

The story here is more subtle (and more interesting). Contrary to intuition, a waveform that is perfectly sampled at a rate at least twice that of the highest frequency component present suffers *no loss of information whatsoever*. Nothing is lost in the unsampled portion of the waveform in between the samples; this is the Nyquist sampling theorem (which has its cadre of unbelievers, who swear that digital audio removes the very soul of music).[28]

[28] It can be shown mathematically that the original signal (excluding a set of pathological waveforms) is perfectly recovered:

Readers harboring contempt for authority may wonder what happens if one violates statute by *under*sampling. Easy enough to try: look at Figure 13.23, where we've sampled a 100 Hz sinewave (requiring $f_{\text{samp}} \geq 200$ sps) at 90 sps, in serious violation of Nyquist's prescription. The sampled points trace out a false signal, in this case 10 Hz. This is called an "alias," and most of the time it's something you don't want.[29] Put simply, for a given sampling rate f_{samp}, the analog input signal must be lowpass filtered (with an "anti-alias filter") such that no significant signal remains above $f_{\text{samp}}/2$. Conversely, for an analog signal that extends to some maximum frequency f_{max}, the minimum sampling rate is $2f_{\text{max}}$. (You can, of course, sample more rapidly than the Nyquist limit $f_{\text{samp}} > 2f_{\text{max}}$, and in fact it's wise to do a modest degree of "oversampling" because that permits more relaxed lowpass filtering of the analog signal, as we'll see presently.)

It's helpful to look at the business of aliasing in the *frequency* domain. In Figure 13.24A we've attached a wimpy 2-section *RC* filter to a broadband signal, putting the -3 dB point of each section at the Nyquist limit ($f_{\text{samp}}/2$). Frequency components in the forbidden region get falsely digitized as shown,[30] contaminating the intended signal band; they cannot be removed later by filtering – in the digitized output they are now "in-band."

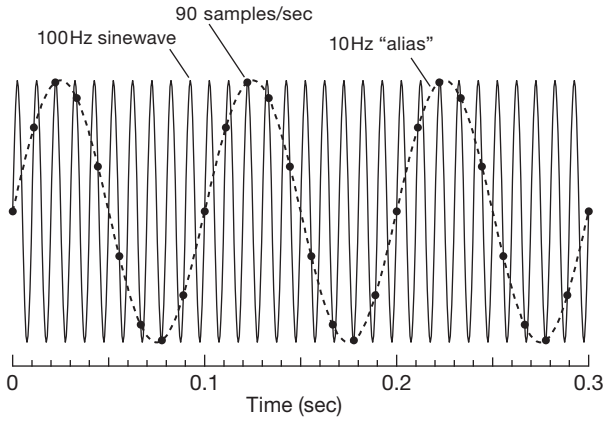

Figure 13.23. Digitizing at less than the Nyquist rate produces "aliases." A 100 Hz sinewave (solid line) sampled at 90 sps (far below the 200 sps Nyquist rate) produces a 10 Hz alias (dots, connected by a dashed line).

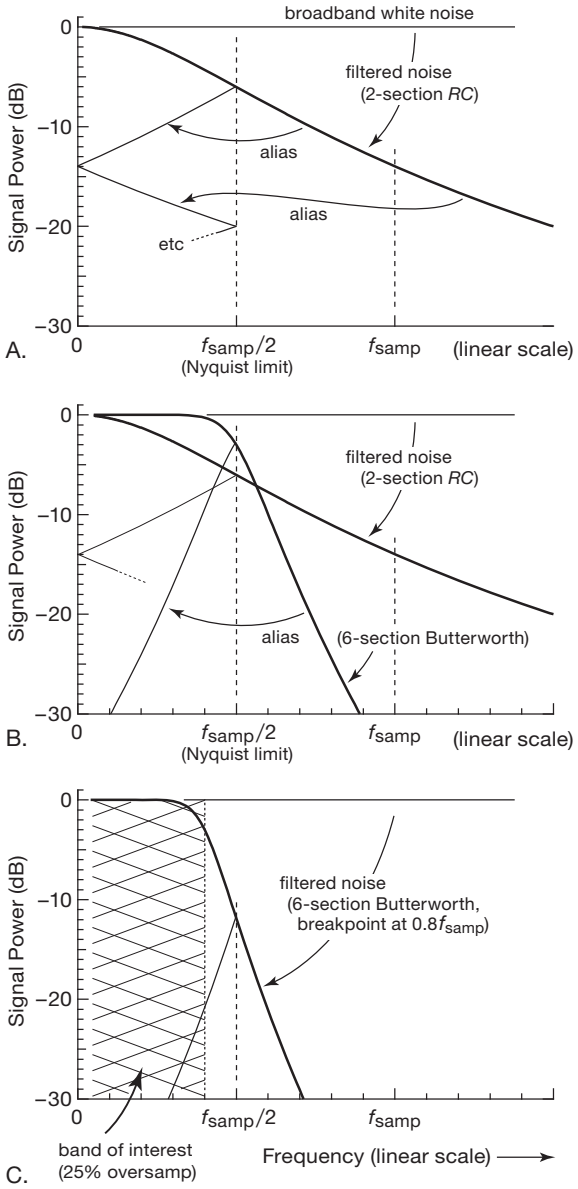

Figure 13.24. Undersampling, oversampling, and aliasing. A. Sampling a signal with frequency components above the Nyquist limit ($f_{\text{samp}}/2$) produces aliased digitized versions that fall within the properly-sampled signal; here a gentle 2-section *RC* rolloff allows significant aliased out-of-band signal energy to contaminate the intended signal. B. A steeper filter is more effective; but aliasing still contaminates the Nyquist band edge. C. Oversampling (setting the Nyquist frequency above the band of interest, here by 25%) gives the anti-alias filter a guard band in which to roll off, greatly reducing aliasing.

$v(t) = \sum v_i \operatorname{sinc} \pi(f_s t - i)$, where f_s is the sampling rate and v_i is the amplitude of the ith sample.

[29] See §13.6.3 for an important exception.

[30] To draw these "elevation sketches," just mirror the contours seen past the multiples of the Nyquist frequency.

A steeper anti-alias filter does a better job, as seen in Figure 13.24B, where we've included a 6-pole Butterworth whose −3 dB point is set at $f_{samp}/2$. But the situation is still far from ideal, with plenty of aliased signal present, especially at the high frequency end.

What you do, then, is to run the sample clock somewhat faster than the Nyquist minimum, as seen in Figure 13.24C, illustrating 25% oversampling relative to the signal band we care about. That gives the anti-alias filter a guard band in which to transition from passband to stopband. Note that we've set the filter's −3 dB point at the band edge, not at the Nyquist frequency.

This is how it's done, in applications where signal purity is important. Using again the example of CD audio, the 20 kHz audio band, if limited by a perfect brickwall lowpass filter at 20 kHz, could be sampled at the Nyquist limit of f_{samp}=40 ksps; but the CD standard sets the rate at 44.1 ksps (10% oversampling), allowing for a 20% filter guard band.[31] Later, in Figure 13.60, we'll use a frequency-domain view of aliasing to understand some benefits of delta–sigma conversion.

Note that there are compromises involved in anti-alias filter design. For example, an analog multi-section filter with steeper transition to cutoff (e.g., a Chebyshev filter) exhibits poorer performance in the time domain (overshoot and ringing, poor phase characteristics, sensitivity to component values, etc.) – see Figures 6.25 and 6.26. For much more on filter types and characteristics see the extensive discussion in Chapter 6 (particularly §6.2.5). And, while you're paging through earlier chapters in search of wisdom, be mindful always of the degrading effects of noise (Chapter 8).

13.5.2 ADC Technologies

There are half a dozen basic techniques of A/D conversion, each with its peculiar advantages and limitations. In the following subsections we'll take each in turn, along with some application examples. Here, in outline form, is a compact summary of these techniques.

Flash, or "parallel" (§13.6) The analog input voltage is compared with a set of fixed reference voltages, most simply by driving an array of 2^n analog comparators to generate an n-bit result. Variations on this theme include pipelined or folded architectures, in which the conversion is done in several steps, each of which converts the "residue" of the previous low-resolution conversion.

Successive approximation (§13.7) Internal logic generates successive trial codes, which are converted to voltages by an internal DAC and compared with the analog input voltage. It requires just n such steps to do an n-bit conversion. The internal DAC can be implemented as a conventional n-stage R–$2R$ resistor ladder or, interestingly, as a set of 2^n binary-scaled capacitors; the latter method is known as a *charge-redistribution* DAC.

Voltage-to-frequency (§13.8.1) The output is a pulse train (or other waveform) whose frequency is accurately proportional to the analog input voltage. In an *asynchronous* V/F the oscillator is internal and free running. By contrast, a *synchronous* V/F requires an external source of clock pulses, gating a fraction of them through such that the *average* output frequency is proportional to the analog input.

Single-slope integration (§13.8.2) The time required for an internally generated analog ramp (capacitor charged by a current source) to go from zero volts to the analog input voltage is proportional to the value of the analog input. That time is converted to an output number by gating a fast fixed-frequency clock, and counting the number of clock pulses. Note that pulse-width modulation employs the same ramp-comparator scheme as single-slope integration to generate the ON time of each cycle.

Dual-slope and multislope integration (§§13.8.3–13.8.4, 13.8.6) These are variations on single-slope integration, effectively eliminating errors from comparator offsets and component stability. In *dual-slope integration* the capacitor is ramped up for a fixed time with a current proportional to the input signal, and ramped back down with a fixed current; the latter time interval is proportional to the analog input. In *quad-slope integration* the input is held at zero while a second such "auto-zero" cycle is done. The so-called *multislope* technique is somewhat different, with a single conversion consisting of a succession of fast dual-slope cycles (in which the input is integrated continuously, combined with subtractive fixed-current cycles), and with a correction based upon the partial-cycle residue at both ends. In some aspects it's a close cousin of the delta–sigma method.

Delta–sigma (§13.9) There are two parts: a *modulator* converts the analog input voltage to a serial *bitstream*; then a digital lowpass filter accepts that bitstream as input, producing the final n-bit digital output. Most simply (and it's never very simple!), the modulator consists of an integrator acting on the difference between the

[31] Much higher oversampling – at *many* times Nyquist rate – is exploited in the technique of delta–sigma conversion, see §13.9.