CHAPTER 7

Digital music

7.1. Digital signals

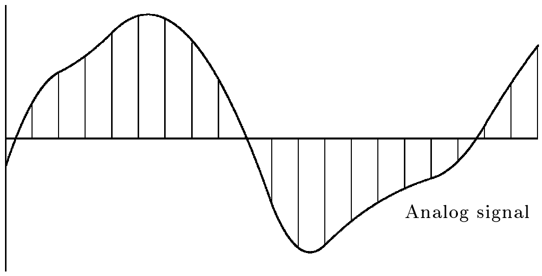
The commonest method of digital representation of sound is about as simple minded as you can get. To digitize an analog signal, the signal is sampled a large number of times a second, and a binary number represents the height of the signal at each sample time. Both of these processes are sometimes referred to as *quantization* (don't worry, there's no quantum mechanics involved here), but it is important to realize that the processes are separate, and need to be understood separately.

For example, the Compact Disc is based on a sample rate of 44.1 KHz, or 44,100 sample points per second.[1] At each sample point, a sixteen digit binary number represents the height of the waveform at that point. The computer WAV file format is another example, which we shall describe in detail in §7.3.

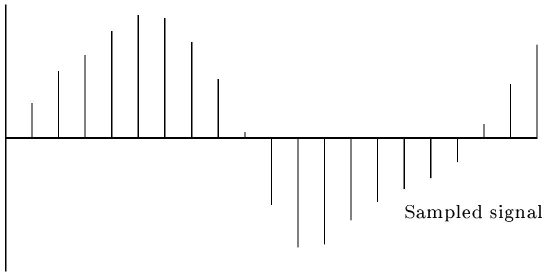
The following diagrams illustrate the process. The first picture shows the original analog signal.

[1]It is annoying that the default sample rate for DAT (Digital Audio Tape) is 48 KHz, thereby making it difficult to make a digital copy on CD directly from DAT. This seems to be the result of industry paranoia at the idea that anyone might make a digital copy of music from a CD (DAT was originally designed as a consumer format, but never took off except among the music business professionals). The excuse that the higher sample rate for DAT gives a higher cutoff frequency and therefore better audio fidelity is easily seen through in light of the fact that the improvement is about three quarters of a tone, which is essentially insignificant.

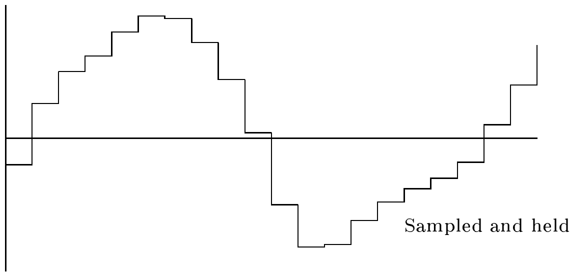
Fortunately, the ratio $48,000/44,100$ can be written as a product of small fractions, $4/3 \times 8/7 \times 5/7$, which suggests an easy method of digital convertion. To multiply the sample rate by $4/3$, for example, we use linear interpolation to quadruple the sample rate and then omit two out of every three sample points. This gives much better fidelity than converting to an analog signal and then back to digital.



Next we have the sampled signal, but still with continuously variable amplitudes.

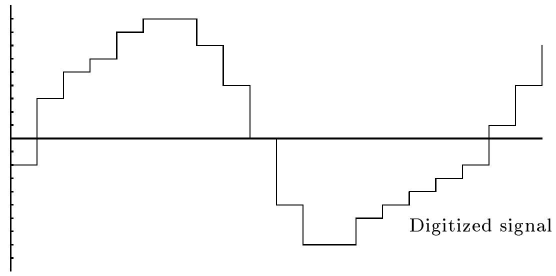


If we apply a "sample and hold" to the signal, we obtain a staircase waveform.



We shall see at the end of §7.6 that provided that the original analog signal has no frequency components at half the sample rate or above (this is achieved with a low pass filter), it may be reconstructed exactly from this sampled signal. This rather extraordinary statement is called the *sampling theorem*, and understanding it requires an understanding of what sampling does to the Fourier spectrum of a signal. We shall do this by systematically making use of Dirac delta functions, starting in §7.5.

Finally, if we digitize the samples, each sample value gets adjusted to the nearest allowed value.



This part of the process of turning an analog signal into a digital signal does entail some loss of information, even if the original signal contains no frequency component above half the sample rate. To see this, think what happens to a very low level signal. It will simply get reported as zero. There is a method for overcoming this limitation to a certain extent, called *dithering*, and it is described in §7.2.

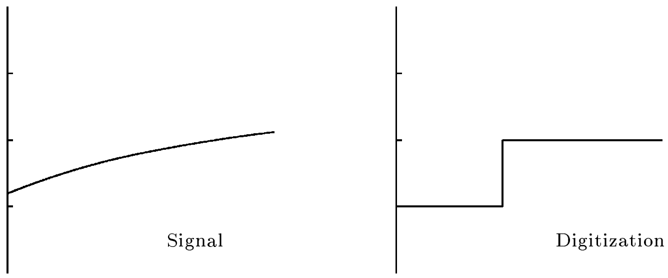
Further reading:

Ken C. Pohlmann, *The compact disc handbook, 2nd edition*, A-R Editions, Inc., Madison, Wisconsin, 1992.

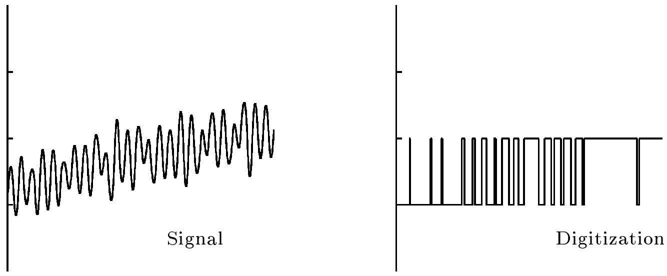
7.2. Dithering

Dithering is a method of decreasing the distortion of a low level signal due to digitization of signal level. This is based on the audacious proposition that adding a low level source of random noise to a signal can increase the signal resolution. This works best when the sample rate is high in comparison with the rate at which the signal is changing.

To see how this works, consider a slowly varying signal and its digitization.



Now if we add noise to the original signal at amplitude roughly one half the step size in the digitization process, here's what the signal looks like.



If the digitized signal is put through a resistor-capacitor circuit to smooth it out, some reasonable approximation to the original signal can be recovered. There is no theoretical limit to the accuracy possible with this method, as long as the sampling rate is high enough.

Further reading:

J. Vanderkooy and S. Lipshitz, *Resolution below the least significant bit in digital audio systems with dither*, J. Audio Eng. Soc. 32 (3) (1984), 106–113; *Correction*, J. Audio Eng. Soc. 32 (11) (1984), 889.

7.3. WAV and MP3 files

A common format for digital sound files on a computer is the WAV format. This is an example of Resource Interchange File Format (RIFF) for multimedia files; another example of RIFF is the AVI movie format. Here is what a WAV file looks like. The file begins with some header information, which comes in a 12 byte RIFF chunk and a 24 byte FORMAT chunk, and then the actual wave data, which comes in a DATA chunk occupying the rest of the file.

The binary numbers in a WAV file are always *little endian*, which means that the least significant byte comes first, so that the bytes are in the reverse of what might be thought of as the normal order. We shall represent binary numbers using *hexadecimal*, format, or base 16. Each hexadecimal digit encodes four binary digits, so that there are two hexadecimal digits in a byte. The sixteen symbols used are 0–9 and A–F. So for example in little endian format, 4E 02 00 00 would represent the hexadecimal number 24E, which is the binary number 0010 0100 1110, or in decimal, 590.

The first 12 bytes are called the RIFF chunk. Bytes 0–3 are 52 49 46 46, the ascii characters “RIFF”. Bytes 4–7 give the total number of bytes in the remaining part of the entire WAV file (byte 8 onward), in little endian format as described above. Bytes 8–11 are 57 41 56 45, the ascii characters “WAVE” to indicate the RIFF file type.

The next 24 bytes are called the FORMAT chunk. Bytes 0–3 are 66 6D 74 20, the ascii characters “fmt ”. Bytes 4–7 give the length of the remainder of the FORMAT chunk, which for a WAV file will always be 10 00 00 00 to indicate 16 bytes. Bytes 8–9 are always 01 00, don’t ask me why. Bytes 10–11 indicate the number of channels, 01 00 for Mono and 02 00 for Stereo. Bytes 12–15 give the sample rate, which is measured in Hz. So for example 44,100 Hz comes out as 44 AC 00 00. Bytes 16–19 give the number of bytes per second. This can be found by multiplying the sample rate with the number of bytes representing each sample. Bytes 20–21 give the number of bytes per sample, so 01 00 for 8-bit mono, 02 00 for 8-bit stereo or 16-bit mono, and 04 00 for 16-bit stereo. Finally, bytes 22–23 give the number of bits per sample, which is 8 times as big as bytes 20–21.

So for 16-bit CD quality stereo audio, the number of bytes per second is 44,100 $\times$ 2 $\times$2 = 176,400, which in hexadecimal is 10 B1 02 00. So the RIFF and FORMAT chunk would be as follows.

```
52 49 46 46 xx xx xx xx-57 41 56 45 66 6D 74 20
10 00 00 00 01 00 02 00-44 AC 00 00 10 B1 02 00
04 00 20 00
```

Here, `xx xx xx xx` represents the total length of the WAV file after the first eight bytes.

Finally, for the DATA chunk, bytes 0–3 are 64 52 74 61 for ascii "data". Bytes 4–7 give the length of the remainder of the DATA chunk, in bytes. Bytes 8 onwards are the actual data samples, in little endian binary as always. The data come in pieces called *sample frames*, each representing the data to be played at a particular point in time. So for example for a 16-bit stereo signal, each sample frame would consist of two bytes for the left channel followed by two bytes for the right channel. Since both positive and negative numbers are to be encoded in the binary data, the format used is *two's complement*. So positive numbers from 0 to $32,767$ begin with a binary digit zero, and negative numbers from $-32,768$ to -1 are represented by adding $65,536$, so that they begin with a binary digit one. For example, the number -1 is represented by the bytes FF FF, $-32,768$ is represented by 00 80, and $32,767$ is represented by FF 7F.

Unfortunately, two's complement is only used when the samples are more than 8 bits long; 8-bit samples are represented using the numbers from 0 to 255, with no negative numbers. So 128 is the neutral position for the signal.

Other digital audio formats similar in nature to the WAV file include the AIFF format (Audio Interchange File Format), commonly used on Macintosh computers, and the AU format, developed by Sun Microsystems and commonly used on UNIX computers.

The MP3 format[2] is different from the WAV format in that it uses *data compression*. The file needs to be uncompressed as it is played.

There are two kinds of compression: lossless and lossy. Lossless compression gives rise to a shorter file from which the original may be reconstructed exactly. For example, the ZIP file format is a lossless compression format. This can only work with non-random data. The more random the

[2]MP3 stands for "MPEG I/II Layer 3." MPEG is itself an acronym for "Motion Picture Experts Group," which is a family of standards for encoding audio-visual information such as movies and music.

data, the less it can be compressed. For example, if the data to be compressed consists of 10,000 consecutive copies of the binary string 01001000, then that information can be imparted in a lot less than 10,000 bytes. In information theory, this is captured by the concept of *entropy*. The entropy of a signal is defined to be the logarithm to base two of the number of different possibilities for the signal. The less random the signal, the fewer possibilities are allowed for the data in the signal, and hence the smaller the entropy. The entropy measures the smallest number of binary bits the signal could be compressed into.

Lossy compression retains the most essential features of the file, and allows some degeneration of the data. The kind of degeneration allowed must always depend on the context. For an audio file, for example, we can try to decide which aspects of the signal make little difference to the perception of the sound, and allow these aspects to change. This is precisely what the MP3 format does.

The actual algorithm is very complicated, and makes use of some subtle psychoacoustics. Here are some of the techniques used for encoding MP3 formats.

(i) The threshold of hearing depends on frequency, and the ear is most sensitive in the middle of the audio frequency range. This is described using the Fletcher–Munson curves, as explained on page 9. So low amplitude sounds at the extremes of the frequency range can be ignored unless there is no other sound present.

(ii) The phenomenon of masking means that some sounds will be present but will not be perceived because of the existence of some other component of the sound. These masked sounds are omitted from the compressed signal.

(iii) A system of borrowing is used, so that a passage which needs more bytes to represent the sound in a perceptually accurate way can use them at the expense of using fewer bytes to represent perceptually simpler passages.

(iv) A stereo signal often does not contain much more information than each channel alone, and joint stereo encoding makes use of this.

(v) The MP3 format also makes use of Huffman coding, in which strings of information which occur with higher probability are coded using a shorter string of bits.

7.4. MIDI

Most synthesizers these days talk to each other and to computers via MIDI cables. MIDI stands for “Musical Instrument Digital Interface”. It is an internationally agreed data transmission protocol, introduced in 1982, for the transmission of musical information between different digital devices. It is important to realise that in general there is no waveform information present in MIDI, unless the message is a “sample dump”. Instead, most MIDI messages give a short list of abstract parameters for an event.

There are three basic types of MIDI message: note messages, controller messages, and system exclusive messages. Note messages carry information about the starting time and stopping time of notes, which patch (or voice) should be used, the volume level, and so on. Controller messages change parameters like chorus, reverb, panning, master volume, etc. System exclusive messages are for transmitting information specific to a given instrument or device. They start with an identifier for the device, and the body can contain any kind of information in a format proprietary to that device. The commonest kind of system exclusive messages are for transmitting the data for setting up a patch on a synthesizer.

The MIDI standard also includes some hardware specifications. It specifies a baud rate of 31.25 KBaud. For modern machines this is very slow, but for the moment we are stuck with this standard. One of the results of this is that systems often suffer from MIDI "bottlenecks," which can cause audibly bad timing. The problem is especially bad with MIDI data involving continually changing values of a control variable such as volume or pitch.

Further reading:

S. de Furia and J. Scacciaferro, *MIDI programmer's handbook* [**36**].

F. R. Moore, *The dysfunctions of MIDI*, Computer Music Journal 12 (1) (1988), 19–28.

J. Rothstein, *MIDI, A comprehensive introduction* [**114**].

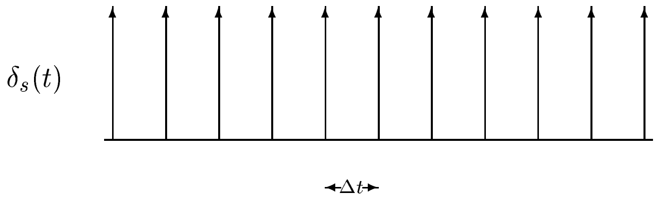
Eleanor Selfridge-Field (Editor), Donald Byrd (Contributor), David Bainbridge (Contributor), *Beyond MIDI: The Handbook of Musical Codes*, M. I. T. Press (1997).

7.5. Delta functions and sampling

One way to represent the process of sampling a signal is as multiplication by a stream of Dirac delta functions (see §2.15). Let N denote the sample rate, measured in samples per second, and let $\Delta t = 1/N$ denote the interval between sample times. So for example for compact disc recording we want $N = 44,100$ samples per second, and $\Delta t = 1/44,100$ seconds. We define the *sampling function* with spacing Δt to be

$$\delta_s(t) = \Delta t \sum_{n=-\infty}^{\infty} \delta(t - n\Delta t).$$

The reason for the factor of Δt in front of the summation is so that the integral of this function over an interval of time approximates the length of the interval.



If $f(t)$ represents an analog signal, then

$$f(t)\delta_s(t) = \Delta t \sum_{n=-\infty}^{\infty} f(t)\delta(t - n\Delta t) = \Delta t \sum_{n=-\infty}^{\infty} f(n\Delta t)\delta(t - n\Delta t)$$

represents the sampled signal. This has been digitized with respect to time, but not with respect to signal amplitude. The integral of the digitized signal $f(t)\delta_s(t)$ over any period of time approximates the integral of the analog signal $f(t)$ over the same time interval.

One of the keys to understanding the digitized signal is Poisson's summation formula from Fourier analysis.

THEOREM 7.5.1.

$$\Delta t \sum_{n=-\infty}^{\infty} f(n\Delta t) = \sum_{n=-\infty}^{\infty} \hat{f}\left(\frac{n}{\Delta t}\right). \tag{7.5.1}$$

PROOF. This follows from the Poisson summation formula (2.14.1), using Exercise 3 of §2.13. □

COROLLARY 7.5.2. *The Fourier transform of the sampling function $\delta_s(t)$ is another sampling function in the frequency domain,*

$$\widehat{\delta_s}(\nu) = \sum_{n=-\infty}^{\infty} \delta\left(\nu - \frac{n}{\Delta t}\right).$$

PROOF. If $f(t)$ is a test function, then the definition of $\delta_s(t)$ gives

$$\int_{-\infty}^{\infty} f(t)\delta_s(t)\,dt = \Delta t \sum_{n=-\infty}^{\infty} f(n\Delta t).$$

Applying Parseval's formula (2.13.4) to the left hand side (and noting that the sampling function is real, so that $\delta_s(t) = \overline{\delta_s(t)}$) and applying formula (7.5.1) to the right hand side, we obtain

$$\int_{-\infty}^{\infty} \hat{f}(\nu)\overline{\widehat{\delta_s}(\nu)}\,d\nu = \sum_{n=-\infty}^{\infty} \hat{f}\left(\frac{n}{\Delta t}\right).$$

The required formula for $\widehat{\delta_s}(\nu)$ follows. □

COROLLARY 7.5.3. *The Fourier transform of a digital signal $f(t)\delta_s(t)$ is*

$$\widehat{f\delta_s}(\nu) = \sum_{n=-\infty}^{\infty} \hat{f}\left(\nu - \frac{n}{\Delta t}\right)$$

which is periodic in the frequency domain, with period equal to the sampling frequency $1/\Delta t$.

PROOF. By Theorem 2.16.1(ii), we have

$$\widehat{f\delta_s}(\nu) = (\hat{f} * \hat{\delta}_s)(\nu),$$

and by Corollary 7.5.2, this is equal to

$$\int_{-\infty}^{\infty} \hat{f}(u) \sum_{n=-\infty}^{\infty} \delta\left(\nu - \frac{n}{\Delta t} - u\right) du = \sum_{n=-\infty}^{\infty} \hat{f}\left(\nu - \frac{n}{\Delta t}\right). \qquad \square$$

7.6. Nyquist's theorem

Nyquist's theorem[3] states that the maximum frequency that can be represented when digitizing an analog signal is exactly half the sampling rate. Frequencies above this limit will give rise to unwanted frequencies below the *Nyquist frequency* of half the sampling rate. What happens to signals at exactly the Nyquist frequency depends on the phase. If the entire frequency spectrum of the signal lies below the Nyquist frequency, then the *sampling theorem* states that the signal can be reconstructed exactly from its digitization.

To explain the reason for Nyquist's theorem, consider a pure sinusoidal wave with frequency ν, for example

$$f(t) = A\cos(2\pi\nu t).$$

Given a sample rate of $N = 1/\Delta t$ samples per second, the height of the function at the Mth sample is given by

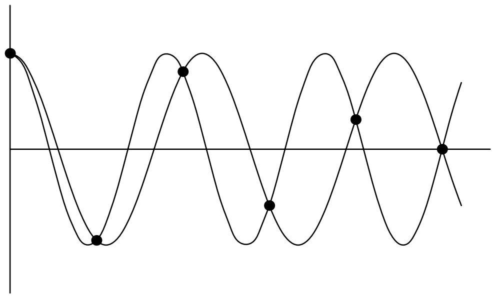
$$f(M/N) = A\cos(2\pi\nu M/N).$$

If ν is greater than $N/2$, say $\nu = N/2 + \alpha$, then

$$\begin{aligned} f(M/N) &= A\cos(2\pi(N/2+\alpha)M/N) \\ &= A\cos(\pi M + 2\pi\alpha M/N) \\ &= (-1)^M A\cos(2\pi\alpha M/N). \end{aligned}$$

Changing the sign of α makes no difference to the outcome of this calculation, so this gives exactly the same answer as the waveform with $\nu = N/2-\alpha$ instead of $\nu = N/2 + \alpha$. To put it another way, the sample points in this calculation are exactly the points where the graphs of the functions $A\cos(2\pi(N/2+\alpha)t)$ and $A\cos(2\pi(N/2-\alpha)t)$ cross.

[3] Harold Nyquist, *Certain topics in telegraph transmission theory*, Transactions of the American Institute of Electrical Engineers, April 1928. Nyquist retired from Bell Labs in 1954 with about 150 patents to his name. He was renowned for his ability to take a complex problem and produce a simple minded solution that was far superior to other, more complicated approaches.



The result of this is that a frequency which is greater than half the sample frequency gets reflected through half the sample frequency, so that it sounds like a frequency of the corresponding amount less than half. This phenomenon is called *aliasing.* In the above diagram, the sample points are represented by black dots. The two waves have frequency slightly more and slightly less than half the sample frequency. It is easy to see from the diagram why the sample values are equal. Namely, the sample points are simply the points where the two graphs cross.

For waves at exactly half the sampling frequency, something interesting occurs. Cosine waves survive intact, but sine waves disappear altogether. This means that phase information is lost, and amplitude information is skewed.

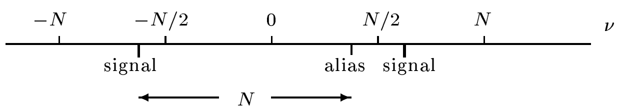
The upshot of Nyquist's theorem is that before digitizing an analog signal, it is essential to pass it through a low pass filter to cut off frequencies above half the sample frequency. Otherwise, each frequency will come paired with its reflection.

In the case of digital compact discs, the cutoff frequency is half of 44.1 KHz, or 22.05 KHz. Since the limit of human perception is usually below 20 KHz, this may be considered satisfactory, but only by a small margin.

We can also explain Nyquist's theorem in terms of Corollary 7.5.3. Namely, the Fourier transform

$$\widehat{f\delta_s}(\nu) = \sum_{n=-\infty}^{\infty} \hat{f}\left(\nu - \frac{n}{\Delta t}\right)$$

is periodic with period equal to the sampling frequency $N = 1/\Delta t$. The term with $n = 0$ in this sum is the Fourier transform of $f(t)$. The remaining terms with $n \neq 0$ appear as aliased artifacts, consisting of frequency components shifted in frequency by multiples of the sampling frequency $N = 1/\Delta t$. If $f(t)$ has a nonzero part of its spectrum at frequency greater than $N/2$, then its Fourier transform will be nonzero at plus and minus this quantity. Then adding or subtracting N will result in an artifact at the corresponding amount less than $N/2$, the other side of the origin.



Another remarkable fact comes out of Corollary 7.5.3, namely the *sampling theorem.* Provided the original signal $f(t)$ satisfies $\hat{f}(\nu) = 0$ for $\nu \geq N/2$, in other words, provided that the entire spectrum lies below the Nyquist frequency, the original signal can be reconstructed exactly from the sampled signal, without any loss of information. To reconstruct $\hat{f}(\nu)$, we begin by by truncating $\widehat{f\delta_s}(\nu)$, and then $f(t)$ is reconstructed using the inverse Fourier transform. Carrying this out in practice is a different matter, and requires very accurate analog filters.

7.7. The z-transform

For digital signals, it is often more convenient to use the z-transform instead of the Fourier transform. The point is that by Corollary 7.5.3, the Fourier transform of a digital signal is periodic, with period equal to the sampling frequency. So it contains a lot of redundant information. The idea of the z-transform is to wrap the Fourier transform round the unit circle in the complex plane. This is achieved by setting

$$\boxed{z = e^{2\pi i\nu\Delta t}}$$

so that as ν changes in value by $1/\Delta t$, z goes exactly once round the unit circle in the complex plane. Any periodic function of ν with period $1/\Delta t$ can then be written as a function of z. The Fourier transform of the sampled signal $f(t)\delta_s(t)$ is then

$$\begin{aligned}\int_{-\infty}^{\infty} f(t)\delta_s(t)e^{-2\pi i\nu t}\,dt &= \int_{-\infty}^{\infty}\left(\sum_{n=-\infty}^{\infty} \Delta t\, f(t)\delta(t-n\Delta t)\right) z^{-t/\Delta t}\,dt \\ &= \sum_{n=-\infty}^{\infty} \Delta t\, f(n\Delta t)z^{-n}.\end{aligned}$$

The factor of Δt is just an annoying constant, and so the z-transform of the digitized signal is simply defined as

$$\boxed{F(z) = \sum_{n=-\infty}^{\infty} f(n\Delta t)z^{-n}.} \tag{7.7.1}$$

The Fourier transform may be recovered as

$$\widehat{f\delta_s}(\nu) = \Delta t\, F(e^{2\pi i\nu\,\Delta t}).$$

Warning. It is necessary to exercise caution when manipulating expressions like equation (7.7.1), because of *Euler's joke.* Here's the joke. Consider a signal which is constant over all time,

$$\begin{aligned}F(z) &= \cdots + z^2 + z + 1 + z^{-1} + z^{-2} + \ldots \\ &= \sum_{n=-\infty}^{\infty} z^n.\end{aligned}$$

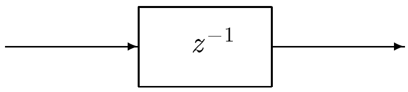
Divide this infinite sum up into two parts, and sum them separately.

$$\begin{aligned} F(z) &= (\dots + z^2 + z + 1) + (z^{-1} + z^{-2} + \dots) \\ &= \frac{1}{1-z} + \frac{z^{-1}}{1-z^{-1}} \\ &= \frac{1}{1-z} + \frac{1}{z-1} \\ &= 0. \end{aligned}$$

This is clearly nonsense. The problem is that the first parenthesized sum only converges for $|z| > 1$, while the second sum only converges for $|z| < 1$. So there is no value of z for which both sums make sense simultaneously.

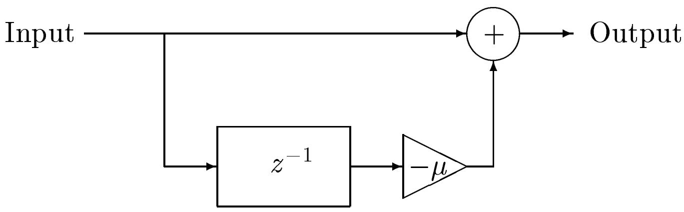
The resolution of this problem is only to allow signals with some finite starting point. So we assume that $f(n\Delta t) = 0$ for all large enough negative values of n.

In terms of the z-transform, delaying the signal by one sample corresponds to multiplication by z^{-1}. So in the literature, you will see the block diagram for such a digital delay drawn as follows. We shall use the same convention.



7.8. Digital filters

The subject of digital filters has a vast literature. We shall only touch the surface, in order to illustrate how the z-transform enters the picture. Let us begin with an example. Consider the following diagram.



If $f(n\Delta t)$ is the input and $g(n\Delta t)$ is the output, then the relation represented by the above diagram is

$$g(n\Delta t) = f(n\Delta t) - \mu f((n-1)\Delta t). \tag{7.8.1}$$

So the relation between the z-transforms is

$$G(z) = F(z) - \mu z^{-1}F(z) = (1 - \mu z^{-1})F(z).$$

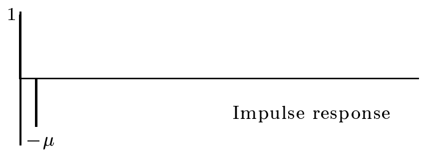
This tells us about the frequency response of the filter. A given frequency ν corresponds to the points $z = e^{\pm 2\pi i\nu\Delta t}$ on the unit circle in the complex plane, with half the sampling frequency corresponding to $e^{\pi i} = -1$.

At a particular point on the unit circle, the value of $1 - \mu z^{-1}$ gives the frequency response. Namely, the amplification is $|1 - \mu z^{-1}|$, and the phase shift is given by the argument of $1 - \mu z^{-1}$.

More generally, if the relationship between the z-transforms of the input and output signal, $F(z)$ and $G(z)$, is given by

$$G(z) = H(z)F(z)$$

then the function $H(z)$ is called the *transfer function* of the filter. The interpretation of the transfer function, for example $1 - \mu z^{-1}$ in the above filter, is that it is the z-transform of the *impulse response* of the filter.



The impulse response is defined to be the output resulting from an input which is zero except at the one sample point $t = 0$, where its value is one, namely

$$f(n\Delta t) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0. \end{cases}$$

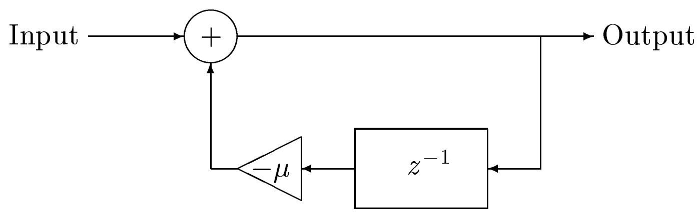
The sampled function $f\delta_s$ is then a Dirac delta function.

For digital signals, the convolution of f_1 and f_2 is defined to be

$$(f_1 * f_2)(n\Delta t) = \sum_{m=-\infty}^{\infty} f_1((n-m)\Delta t)f_2(m\Delta t).$$

Multiplication of z-transforms corresponds to convolution of the original signals. This is easy to see in terms of how power series in z^{-1} multiply. So in the above example, the impulse response is: 1 at $n = 0$, $-\mu$ at $n = 1$, and zero for $n \neq 0, 1$. Convolution of the input signal $f(n\Delta t)$ with the impulse response gives the output signal $g(n\Delta t)$ according to equation (7.8.1).

As a second example, consider a filter with feedback.



The relation between the input $f(n\Delta t)$ and the output $g(n\Delta t)$ is now given by

$$g(n\Delta t) = f(n\Delta t) - \mu g((n-1)\Delta t).$$

This time, the relation between the z-transforms is

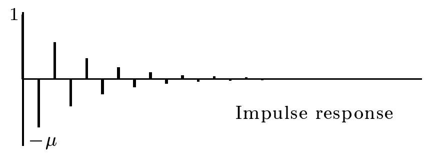
$$G(z) = F(z) - \mu z^{-1} G(z),$$

or

$$G(z) = \frac{1}{1+\mu z^{-1}} F(z).$$

Notice that this is unstable when $|\mu| > 1$, in the sense that the signal grows without bound. Even when $|\mu| = 1$, the signal never dies away, so we say that this filter is *stable* provided $|\mu| < 1$. This is easiest to see in terms of the impulse response of this filter, which is

$$\frac{1}{1+\mu z^{-1}} = 1 - \mu z^{-1} + \mu^2 z^{-2} - \mu^3 z^{-3} + \dots$$



Filters are usually designed in such a way that the output $g(n\Delta t)$ depends linearly on $f((n-m)\Delta t)$ for a finite set of values of $m \geq 0$ and on $g((n-m)\Delta t)$ for a finite set of values of $m > 0$. For such a filter, the z-transform of the impulse response is a rational function of z, which means that it is a ratio of two polynomials

$$\frac{p(z)}{q(z)} = a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots$$

The coefficients $a_0, a_1, a_2, \dots$ are the values of the impulse response at $t = 0$, $t = \Delta t$, $t = 2\Delta t$, ...

The coefficients a_n tend to zero as n tends to infinity, if and only if the poles μ of $p(z)/q(z)$ satisfy $|\mu| < 1$. This can be seen in terms of the complex partial fraction expansion of the function $p(z)/q(z)$.

The location of the poles inside the unit circle has a great deal of effect on the frequency response of the filter. If there is a pole near the boundary, it will cause a local maximum in the frequency response, which is called a *resonance*. The frequency is given in terms of the argument of the position of the pole by

$$\nu = (\text{sample rate}) \times (\text{argument})/2\pi.$$

Decay time. The decay time of a filter for a particular frequency is defined to be the time it takes for the amplitude of that frequency component to reach $1/e$ of its initial value. To understand the effect of the location of a pole on the decay time, we examine the transfer function

$$H(z) = \frac{1}{z-a} = \frac{z^{-1}}{1-az^{-1}} = z^{-1} + az^{-2} + a^2z^{-3} + \dots.$$

So in a period of n sample times, the amplitude is multiplied by a factor of a^n. So we want $|a|^n = 1/e$, or $n = -1/\ln|a|$. So the formula for decay time is

$$\boxed{\text{Decay time} = \frac{-\Delta t}{\ln|a|} = \frac{-1}{N\ln|a|}} \tag{7.8.2}$$

where $N = 1/\Delta t$ is the sample rate. So the decay time is inversely proportional to the logarithm of the absolute value of the location of the pole. The further the pole is inside the unit circle, the smaller the decay time, and the faster the decay. A pole near the unit circle gives rise to a slow decay.

Exercises

1. (a) Design a digital filter whose transfer function is $z^2/(z^2 + z + \frac{1}{2})$, using the symbol z^{-1} in a box to denote a delay of one sample time, as above.

(b) Compute the frequency response of this filter. Let N denote the number of sample points per second, so that the answer should be a function of ν for $-N/2 < \nu < N/2$.

(c) Is this filter stable?

Further reading:

R. W. Hamming, *Digital filters* [**45**].

Bernard Mulgrew, Peter Grant and John Thompson, *Digital signal processing* [**83**].

7.9. The discrete Fourier transform

The discrete Fourier transform describes the frequency components of a digitized signal of finite length. If the length of the signal is N, then the discrete Fourier transform is given by

$$F(k) = \frac{1}{N}\sum_{n=0}^{N-1} f(n)e^{-2\pi ink/N}$$

$$f(n) = \sum_{k=0}^{N-1} F(k) e^{2\pi i k n/N}.$$

For a long signal, the usual process is to choose for N a number that is used as a window size for a moving window in the signal. So the discrete Fourier transform is really a digitized version of the windowed Fourier transform.

The fast Fourier transform is a way to compute the discrete Fourier transform using $2N \log_2 N$ operations rather than N^2. The number of sample points N has to be a power of two for it to be this efficient, but the algorithm works for any highly composite value of N.

Further reading:

G. D. Bergland, *A guided tour of the fast Fourier transform*, IEEE Spectrum 6 (1969), 41–52.

James W. Cooley and John W. Tukey, *An algorithm for the machine calculation of complex Fourier series*, Math. of Computation 19 (1965), 297–301. This is usually regarded as the original article announcing the fast Fourier transform as a practical algorithm.