Signals and Systems: Module 12

Suggested Reading: S&S 7.1-7.4

Back in the beginning of the course, we introduced **sampling** as the process of representing a continuous-time signal in discrete-time. This is of tremendous importance in practice, as virtually all computing devices are discrete-time systems that process and operate on digital signals. To apply continuous-time signals to such systems, we need to convert a continuous-time signal to discrete-time (i.e., sampling), process it in discrete-time, and then convert it back to continuous-time (referred to as reconstruction). This process is illustrated at a high-level in Figure 1 below.

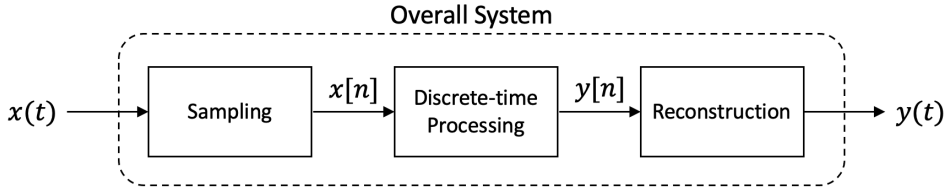


Figure 1: Processing continuous-time signals with discrete-time systems.

Sampling

We will first dive deeper into the process of sampling, before discussing discrete-time processing and reconstruction. In doing so, we will see that under certain conditions on how frequently we sample a continuous-time signal, we may be able to fully recover it from its so-called **samples**. Consider a movie, for example: a moving picture is fundamentally a sequence of individual frames that, when played back fast enough, appear to the human eye as a continuously changing scene. Each frame here is really a sample of the scene; how frequent do the samples need to be for it to look continuous?

Conceptual Sampling and the Sampling Theorem

Consider the three continuous-time signals $x_1(t), x_2(t), x_3(t)$ in Figure 2. While they are clearly distinct, if we tried to represent them as samples at integer multiples of T, they would appear identical: $x_1(kT) = x_2(kT) = x_3(kT)$. Said another way, there is really an infinite number of signals that can be generated from a given set of samples. How, then, could we ever fully recover a signal from its samples?

As we will see, if a signal is **band limited** – meaning its Fourier transform is zero outside of some finite band of frequencies – then it is possible to find samples that uniquely specify the original signal.

Impulse-train sampling. Procedurally, we can think of sampling a signal $x(t)$ as multiplying $x(t)$ by a periodic impulse train. In particular, in the time domain,

$$x_p(t) = x(t)p(t), \qquad p(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT)$$

where $p(t)$ is the **sampling function**, its period T is the **sampling period**, and its fundamental frequency $\omega_s = 2\pi/T$ is the **sampling frequency**. Since $x(t)\delta(t - t_0) = x(t_0)\delta(t - t_0)$, $x_p(t)$ is also an impulse train, with the weights as samples of $x(t)$, i.e.,

$$x_p(t) = \sum_{n=-\infty}^{\infty} x(nT)\delta(t - nT)$$

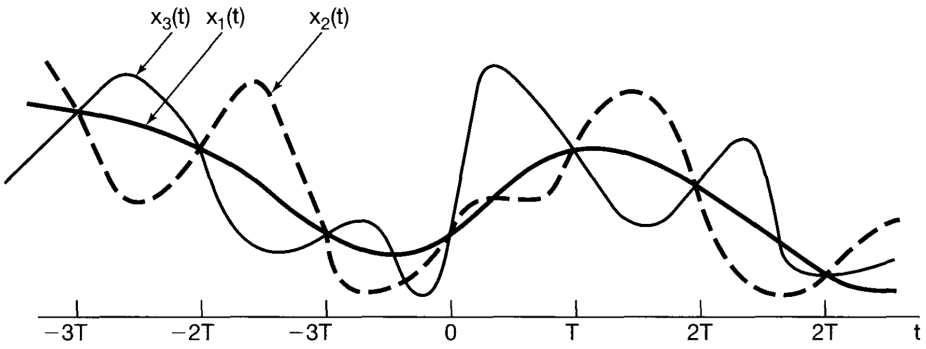
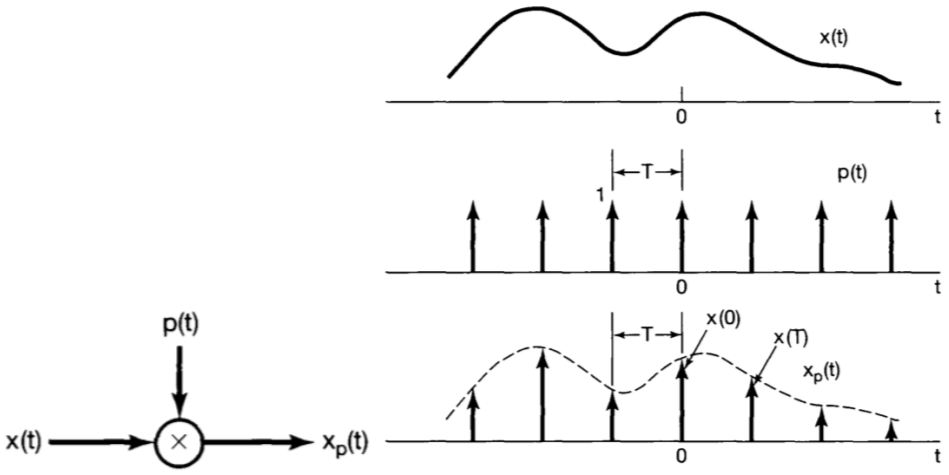


Figure 2: Three different continuous-time signals with the same sample values.



(a) Sampler (b) Result

Figure 3: Impulse-train sampling with the function $p(t)$ multiplied by $x(t)$ to obtain the sampled $x_p(t)$.

Note here that while n must be an integer, nT does not have to be. This impulse-train sampling process is illustrated in Figure 3.

Now we will consider the relationship between the frequency spectra of $x(t)$ and $x_p(t)$. From the multiplication property of the CTFT, we know that

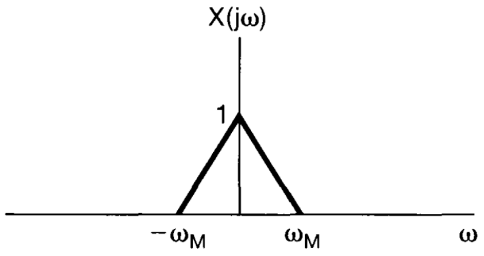
$$X_p(j\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\theta)P(j(\omega-\theta))d\theta = \frac{1}{2\pi}X(j\omega) * P(j\omega)$$

To obtain the Fourier transform of $p(t)$, recall that the spectrum of any periodic signal is determined by its Fourier series coefficients. Writing $p(t)$ as its Fourier series, we have the following transform pair:

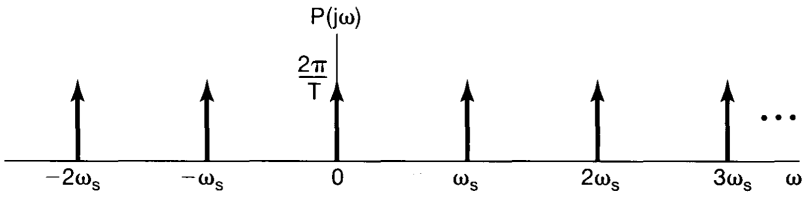
$$p(t) = \sum_{k=-\infty}^{\infty} a_k e^{jk\omega_0 t} \overset{\mathcal{F}}{\longleftrightarrow} P(j\omega) = \sum_{k=-\infty}^{\infty} 2\pi a_k \delta(\omega - k\omega_0)$$

We can determine the coefficients a_k by choosing the interval of integration for the Fourier series coefficients to be $[-T/2, T/2]$:

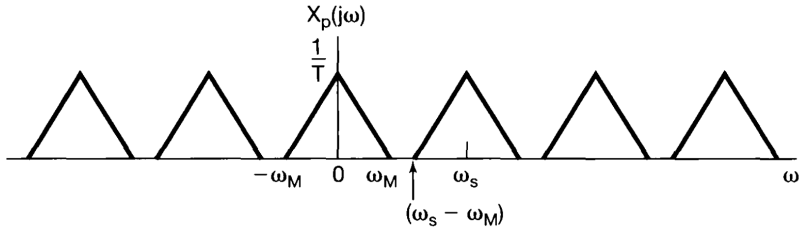
$$a_k = \frac{1}{T}\int_{-T/2}^{T/2} p(t)e^{-jk\omega_0 t}dt = \frac{1}{T}\int_{-T/2}^{T/2} \delta(t)e^{-jk\omega_0 t}dt = \frac{1}{T}e^{-jk\omega_0 \cdot 0} = \frac{1}{T}$$



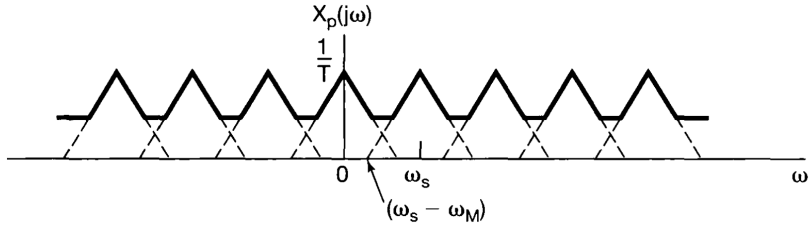
(a) Original signal



(b) Sampling function



(c) Sampled signal, $\omega_s > 2\omega_m$



(d) Sampled signal, $\omega_s < 2\omega_m$

Figure 4: Effect in the frequency domain of sampling in the time domain. If the sampling frequency is at least twice the signal bandwidth, nothing is lost.

Thus, the spectrum of the sampling function is

$$P(j\omega) = \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} \delta\left(\omega - \frac{2\pi k}{T}\right) = \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} \delta\left(\omega - k\omega_s\right)$$

which is also an impulse train. Now, we know that convolution with an impulse simply shifts the function: $X(j\omega) * \delta(\omega - k\omega_s) = X(j(\omega - k\omega_s))$. Therefore, $X_p(j\omega)$ is a summation of shifted versions of $X(j\omega)$ centered at multiples of the sampling frequency:

$$X_p(j\omega) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X(j(\omega - k\omega_s))$$

The implications of this are shown visually in Figure 4:

(a) gives the spectrum of a general band-limited signal $x(t)$, i.e., with all frequencies contained between $-\omega_m$ and ω_m.

(b) is the spectrum of a sampling function $p(t)$ with sampling frequency ω_s. Multiplying the two gives $X(j\omega)$ repeated every ω_s, for which there are two cases:

(c) if $\omega_s - \omega_M > \omega_M$, there will be no overlap between the shifted replicas. When there is no overlap, we can reconstruct $x(t)$ exactly from $x_p(t)$ using a lowpass filter (with a gain of T and a cutoff frequency between ω_M and $\omega_s - \omega_M$).

(d) otherwise, there will be overlaps between the shifted replicas.

This leads us to a famous result called the **sampling theorem**. Consider a band-limited signal $x(t)$ with bandwidth ω_M. If we sample it as $x(nT)$, for $n = 0, \pm 1, \pm 2, ...$ using a sampling frequency $\omega_s = 2\pi/T$ such that $\omega_s > 2\omega_M$, then $x(t)$ can be uniquely determined from its samples. In particular, given these samples, we can reconstruct $x(t)$ by generating a periodic impulse train in which successive impulses have amplitudes that are successive sample values. After passing this impulse train through a lowpass filter, we will recover $x(t)$ exactly.

The frequency $2\omega_M$ is commonly referred to as the **Nyquist rate**.

Practical challenges. Figure 5 summarizes the system we are describing for sampling a signal $x(t)$ and recovering $x_p(t)$, using a sampling function $p(t)$ and a lowpass filter $h(t)$. Even if we define $\omega_s > 2\omega_M$, we still face at least two challenges in building this system:

- *Lowpass filters are not ideal*: As we have discussed, a lowpass filter can only be approximated at best. So we need to build a filter that approximates the desired frequency characteristics of a lowpass filter, i.e.,

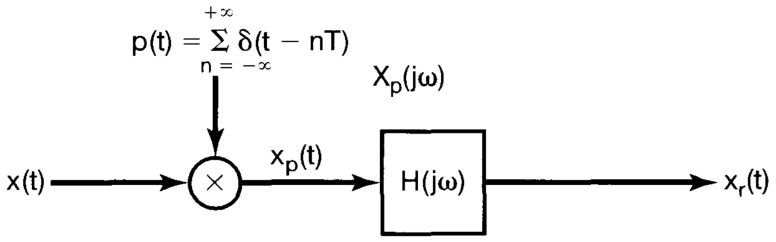


Figure 5: System for sampling $x(t)$ with a sampling function $p(t)$ and recovering $x_r(t)$ with a lowpass filter $h(t)$.

$H(j\omega) \approx 1$ for $|\omega| < \omega_M$ and $H(j\omega) \approx 0$ for $|\omega| > \omega_M$. Such approximations will lead to discrepancies between $X(j\omega)$ and $X_r(j\omega)$. The tolerable level of distortion will dictate how much care must be put into the design of this filter.

- *Impulses are hard to approximate*: The sampling function $p(t)$, consisting of a train of ideal impulses, must also be approximated. One possibility would be to use narrow, large-amplitude pulses that have small but finite width and high but finite height, but these are also relatively difficult to generate and transmit.

Practical Sampling and Interpolation

To address the second challenge mentioned above, one possibility is to generate the sampled signal in the form of a **zero-order hold**. Such a system will take a sample of $x(t)$ and then hold the sample until the next instant at which a new sample is taken, as shown in Figure 6 below. In other words, the signal value remains constant in-between the sampling points.

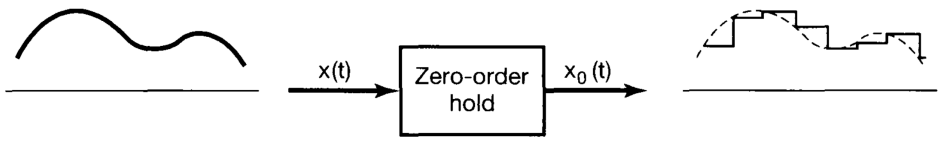


Figure 6: Sampling with a zero-order hold.

The zero-order hold is much easier to generate and transmit in practice. The challenge comes in the reconstruction step, as we can no longer use an ideal low-pass filter. To see this, consider Figure 7, which depicts the full

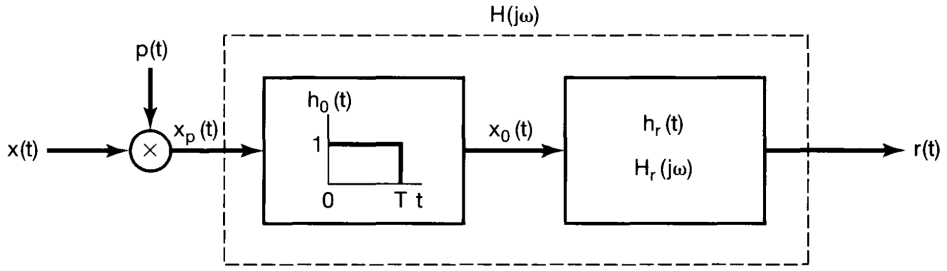


Figure 7: Reasoning about the reconstruction filter for zero-order hold sampling.

sampling and reconstruction process. Here, we are functionally representing the zero-order hold as an impulse sampling function cascaded with an LTI system that has impulse response $h_0(t) = u(t) - u(t-T)$. With $x_p(t)$ from Figure 3, $x_p(t) * h_0(t)$ will give us the zero-order hold in Figure 6 as it just becomes shifted versions of $h_0(t)$ scaled by the samples of $x(t)$.

The question boils down to what the reconstruction filter $h_r(t)$ must be. Overall, we need to design Figure 7 such that $r(t) = x(t)$. Therefore, the cascade frequency response $H(j\omega) = H_0(j\omega)H_r(j\omega)$ must be an ideal low-pass filter. We know from a common Fourier transform pair and the time-shifting property that

$$H_0(j\omega) = \left[\frac{2\sin(\omega T/2)}{\omega}\right] e^{-j\omega T/2}$$

Therefore, we need

$$H_r(j\omega) = \frac{\omega e^{j\omega T/2} H(j\omega)}{2\sin(\omega T/2)}$$

This frequency response also cannot be perfectly realized, and must be approximated in practice.

If we cannot actually implement the reconstruction filter, then what is the point of the zero-order hold in the first place? In fact, in many situations, the signal $x_0(t)$ *can already be considered an adequate approximation* of the original $x(t)$. In other words, if we can deal with the approximation being constant in-between the sampled points, we do not need to conduct any filtering at all!

Interpolation. The zero-order hold is the crudest form of interpolation, where we fit a continuous signal to a set of sample values. Interpolation gives

us a useful set of techniques for reconstructing a sampled signal without using a complex filter.

Perhaps the most common type is **linear interpolation**, whereby adjacent sample points are connected by a straight line. This is shown in Figure 8 below. More generally, we can connect the adjacent points by higher-order polynomials to approximate the original signal.

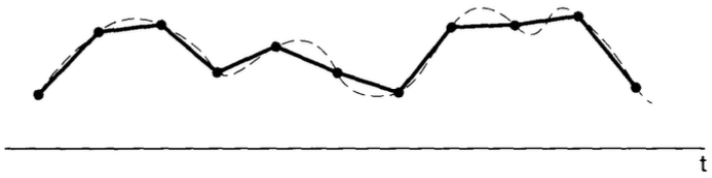


Figure 8: Linear interpolation.

To gain more insight into interpolation, suppose we take the sampled signal $x_p(t) = \sum_{n=-\infty}^{\infty} x(nT)\delta(t-nT)$ and pass it through a lowpass filter with impulse response $h(t)$. By the convolution property, we will obtain

$$x_r(t) = \sum_{n=-\infty}^{\infty} x(nT)h(t-nT)$$

which is, by definition, an interpolation formula, with $h(t)$ providing the interpolation function. If $h(t)$ is an ideal lowpass filter with cutoff frequency ω_c, then

$$h(t) = \frac{\omega_c T \sin(\omega_c t)}{\pi \omega_c t} \quad \rightarrow \quad x_r(t) = \sum_{n=-\infty}^{\infty} x(nT) \cdot \frac{\omega_c T}{\pi} \cdot \frac{\sin(\omega_c(t-nT))}{\omega_c(t-nT)}$$

This is a superposition of shifted sinc functions weighted by the sampled points. The result for when $\omega_c = \omega_s/2$ is illustrated in Fig. 9: the top is of the figure gives the original $x(t)$, and underneath are the superposition terms. When they are added together, they give $x_r(t)$, which will match $x(t)$ assuming ω_s satisfies the sampling theorem: as long as this is satisfied, we can, in theory, perfectly interpolate between the points.

Interpolation in this way, using an ideal lowpass filter, is commonly referred to as **band-limited interpolation**, since it implements exact reconstruction if $x(t)$ is band limited and the sampling frequency satisfies the conditions of the sampling theorem. However, as we have mentioned several times, we often desire using something simpler than an ideal lowpass filter

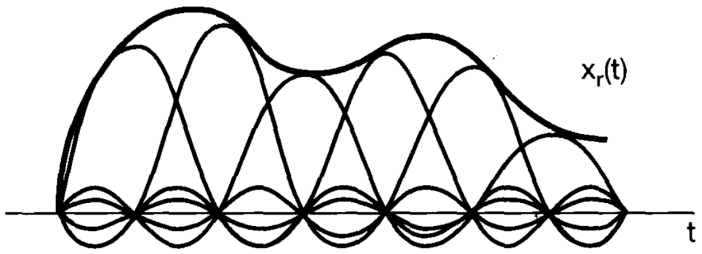


Figure 9: Ideal band-limited interpolation, where the sampled signal is passed through an ideal low-pass filter, becomes a superposition of sinc functions.

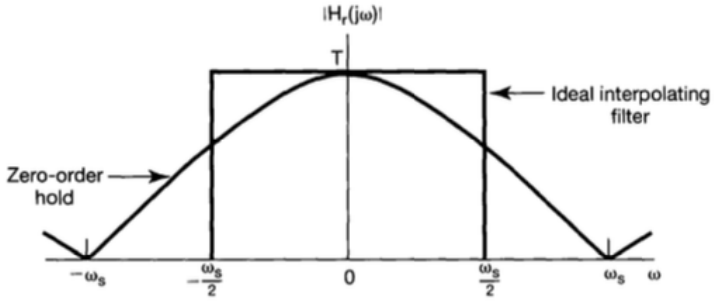


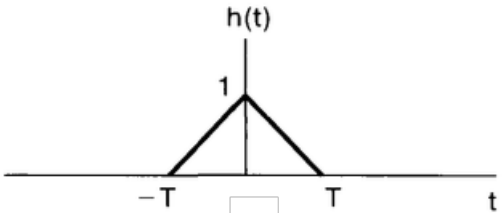
Figure 10: Ideal band-limited interpolation becomes a superposition of sinc functions.

(in the frequency domain). We can now see that this is equivalent to using a simpler interpolation function (in the time domain). For example, the zero-order hold has the impulse response shown in Figure 7, and the frequency response depicted in Figure 10; we can see that it is a very rough approximation of band-limited interpolation.

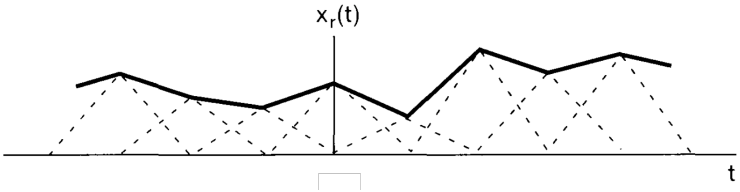
To create a more sophisticated interpolation, we can apply additional low-pass filtering to create a higher order filter. This will tend to improve the reconstruction result through a smoothening effect on the interpolation in the time domain. For instance, if we use a filter with the triangular shaped impulse response in Figure 11(a), this will have a frequency response

$$H(j\omega) = \frac{1}{T}\left[\frac{\sin(\omega T/2)}{\omega/2}\right]^2$$

which has a sharper (squaring effect) in the frequency domain. By the convolution property, we know that this is obtained by cascading two zero-order holds: in effect, this results in squaring the frequency response in



(a) Impulse response.



(b) Reconstruction.

Figure 11: Impulse response and illustration of reconstruction for linear interpolation (first-order hold).

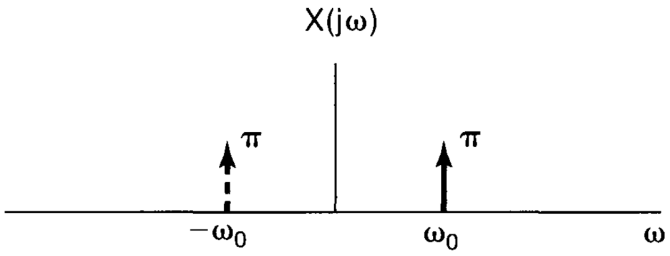
Figure 10 (with a phase shift), or convolving the impulse response in Figure 7 with itself (after a time shift). The result is called a **first-order hold**, which, in fact, corresponds to linear interpolation! This is shown in Figure 11(b).

We can keep applying this filter cascading logic to produce **higher-order holds** as well, which will give us a higher degree of polynomial interpolation.

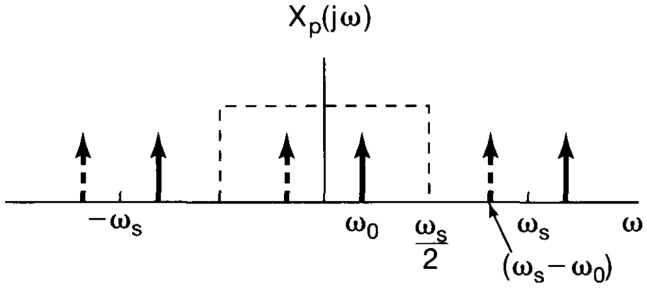
Aliasing

Returning back to the system in Figure 5, suppose we had the ability to construct a perfect impulse train and an ideal low-pass filter. What happens when we do not choose a large enough sampling frequency ω_s relative to the signal bandwidth ω_M?

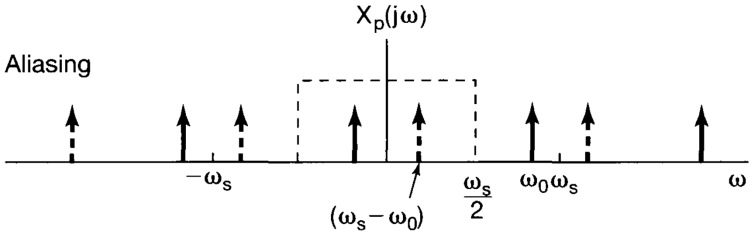
We saw in Figure 4 that when $\omega_s < 2\omega_M$, $X_p(j\omega)$ no longer contains exact replications of $X(j\omega)$ and thus is no longer exactly recoverable. This effect, in which the different bands of $X_p(j\omega)$ overlap, is referred to as **aliasing**. Note that even if $x(t)$ is not completely recoverable by $x_r(t)$, the two will always be equal at the sampling instants, i.e., $x_r(nT) = x(nT)$, $n = 0, \pm 1, \pm 2, ...$



(a) Sinusoidal signal



(b) Sampling with $\omega_0 < \omega_s/2$



(c) Sampling with $\omega_0 > \omega_s/2$

Figure 12: Effect of oversampling and undersampling a sinusoidal signal. With oversampling in (b), we recover $\cos\omega_0 t$ after lowpass filtering, while with undersampling in (c), we get $\cos(\omega_s - \omega_0)t$.

We can gain insight into the effect of aliasing in the time domain by considering the simple sinusoidal signal $x(t) = \cos\omega_0 t$. Figure 12(a) gives the spectrum, where we visually distinguish between the impulse at ω_0 and $-\omega_0$. Suppose we sample $x(t)$ at a frequency ω_s to obtain $x_p(t)$, and then apply an ideal lowpass filter with cutoff frequency $\omega_s/2$ to get $x_r(t)$. We consider two different intervals of the sampling frequency:

- Suppose $\omega_s > 2\omega_0$. The spectrum of the sampled signal is given in Figure 8(b). $X_p(j\omega)$ contains replicas of $X(j\omega)$ at $k\omega_s \pm \omega_0$ for all integer values of k, but all of these replicas fall outside of the passband of the lowpass filter. This is a case of **oversampling**, i.e., where ω_s is larger than what is needed, and $x_r(t) = \cos \omega_0 t$.
- Now suppose we decrease ω_s so that $\omega_s < 2\omega_0$. (We will further assume for convenience that $\omega_s > \omega_0$.) This is called **undersampling**. The spectrum of the sampled signal is shown in Figure 8(c), and leads to aliasing. The passband contains impulses at $\pm(\omega_s - \omega_0)$, so the original frequency ω_0 in $x(t)$ takes on the identity of a lower frequency $\omega_s - \omega_0$ in $x_r(t)$. In this case, $x_r(t) = \cos(\omega_s - \omega_0)t$.

It is important to note that the sampling theorem explicitly requires that the sampling frequency be *strictly greater than* twice the highest frequency in the signal, rather than $\omega_s \geq 2\omega_M$. Visually, we can see in Figure 12 that this would cause the impulses to all lie at odd multiples of $\omega_s/2$ and overlap, thereby not giving us $x(t)$ back. We illustrate the result of sampling a sinusoidal signal this way in the following example.

Example 1. *Suppose the sinusoidal signal $x(t) = \cos(2\pi t + \phi)$ is sampled using impulse sampling at exactly twice the frequency of the sinusoid, i.e., $\omega_s = 4\pi$, and then applied to an ideal lowpass filter with cutoff frequency $\omega_l = 2\pi$ to obtain $x_r(t)$. What is $x_r(t)$?*

Ans: By definition, the sampled signal $x_p(t)$ will be

$$x_p(t) = \sum_{n=-\infty}^{\infty} x(nT)\delta(t - nT)$$

We can use a trigonometric identity to write

$$x(t) = \cos(2\pi t + \phi) = \cos(2\pi t)\cos(\phi) - \sin(2\pi t)\sin(\phi)$$

The second term goes to zero in $x(nT)$ since $T = 2\pi/\omega_s = 1/2$: $\sin(n\pi)\sin(\phi) = 0$. So, the sampled $x_p(t)$ can be written as

$$x_p(t) = \sum_{n=-\infty}^{\infty} \cos(\pi n)\cos(\phi)\delta\left(t - \frac{n}{2}\right)$$

Since the lowpass filter satisfies the sampling theorem, this corresponds to band-limited interpolation. Thus, we get the exact reconstruction corresponding to $x_p(t)$, which is

$$x_r(t) = \cos(\phi)\cos(2\pi t)$$

As a consequence, we see that perfect reconstruction of $x(t)$ occurs only in the case in which the phase ϕ is zero or an integer multiple of 2π.

DT Processing of CT Signals

As stated, we often want to use discrete-time systems to process signals, even continuous-time signals. The high-level process for doing so is depicted in Figure 13, which is essentially a replica of Figure 1: starting with a continuous-time signal $x_c(t)$, we convert that to the discrete-time $x_d[n]$, pass that through a discrete-time system to obtain $y_d[n]$, and finally convert the output back to a continuous time $y_c(t)$.

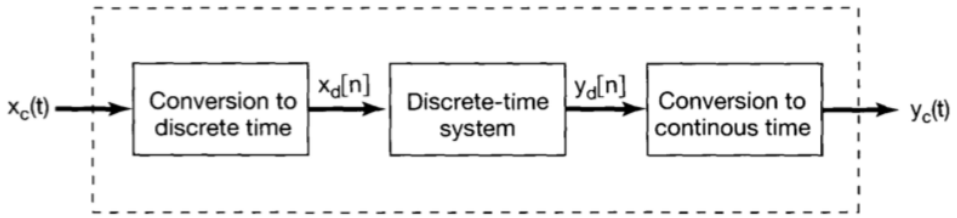


Figure 13: Processing continuous-time signals with discrete-time systems.

We want to examine the processes in Figure 13 in the frequency domain. If we use pulse-train sampling on $x_c(t)$ to obtain $x_p(t)$, we know that

$$x_p(t) = \sum_{n=-\infty}^{\infty} x_c(nT)\delta(t-nT) \overset{F}{\longleftrightarrow} X_p(j\omega) = \sum_{n=-\infty}^{\infty} x_c(nT)e^{-j\omega nT}$$

Now, consider the DTFT of $x_d[n]$. Using the definition and noting $x_d[n] = x_c(nT)$, we have

$$X_d(e^{j\Omega}) = \sum_{n=-\infty}^{\infty} x_d[n]e^{-j\Omega n} = \sum_{n=-\infty}^{\infty} x_c(nT)e^{-j\Omega n}$$

where we have used Ω in place of ω to distinguish between the continuous-time and discrete-time Fourier transform variables. We can see, then, that the transforms are related by

$$X_d(e^{j\Omega}) = X_p(j\Omega/T)$$

We also have a relationship between $X_p(j\omega)$ and $X_c(j\omega)$ that we derived from the properties of convolution. We can use this now to obtain a relationship between X_d and X_c:

$$X_p(j\omega) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_c(j(\omega - k\omega_s)) \quad \rightarrow \quad X_d(e^{j\omega}) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_c(j(\omega - 2\pi k)/T)$$

since $\omega_s = 2\pi/T$ and we only scale ω, not ω_s, by $1/T$.

To convert $y_d[n]$ to $y_c(t)$, we reverse the steps: generate a continuous-time impulse train $y_p(t)$ from $y_d[n]$, and then use a lowpass filter to recover $y_c(t)$. To get the "right" output, we must assume that the sampling frequency used in the first place at the input was sufficiently high. The overall system we are describing is given in Figure 14.

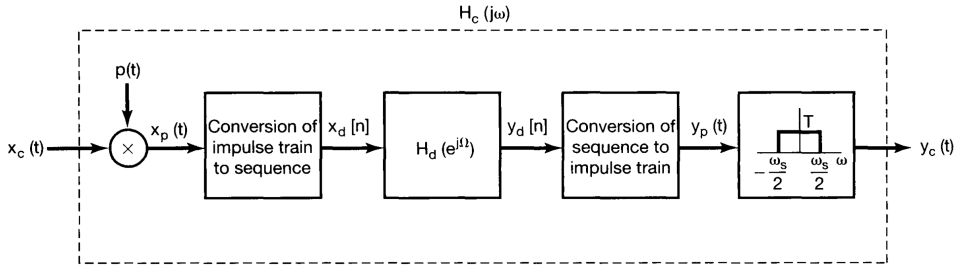


Figure 14: More formal version of the system depicted in Figure 13.

Tracing the spectra. To understand the dynamics for a general discrete-time system $H_d(e^{j\Omega})$ in the frequency domain, consider the series of Fourier transforms plotted in Figure 15.

- In (a), we have a generic band-limited signal $X_c(j\omega)$ that has been applied to the system.
- Sampling at a frequency $\omega_s > 2\omega_M$ gives the pulse train spectrum $X_p(j\omega)$ in (b), which is periodic in ω_s.
- When this pulse train is converted to a discrete-time signal, the spectrum becomes $X_d(e^{j\Omega}) = X_p(j\Omega/T)$ in (c), which is periodic in 2π.
- Then, $X_d(e^{j\Omega})$ is multiplied by the frequency response $H_d(e^{j\Omega})$ to obtain the spectrum $Y_d(e^{j\Omega})$ of $y_d[n]$: these two are overlaid in (d).

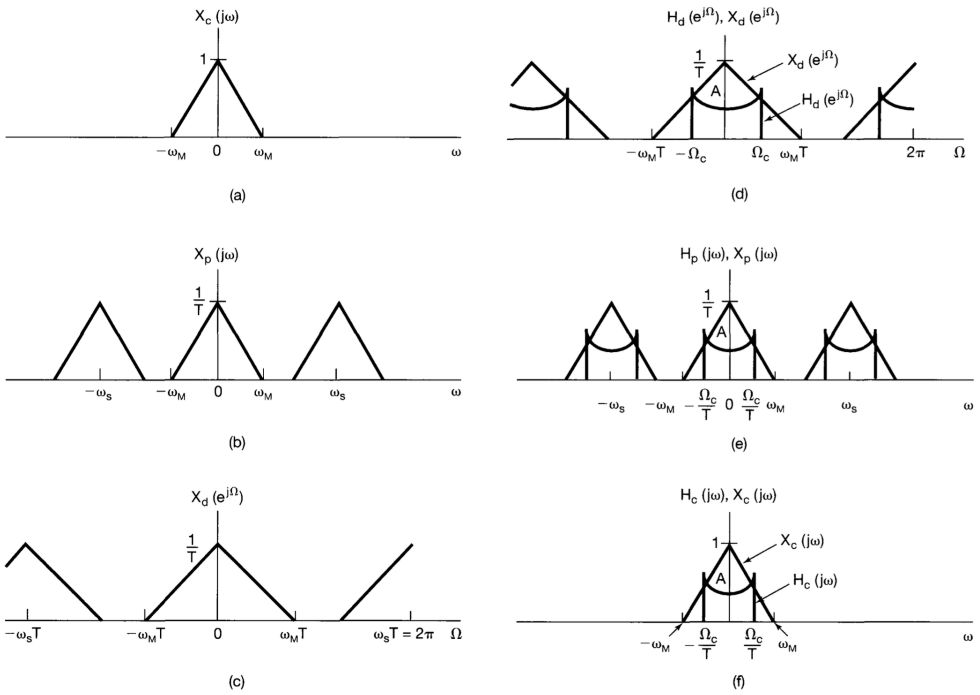


Figure 15: Visualization of the spectra at different steps of the system in Figure 14, for a generic CT signal $x_c(t)$ and DT LTI system $H_d(e^{j\Omega})$.

- The impulse train converting $y_d[n]$ to $y_p(t)$ is then a frequency scaling in the opposite direction: $Y_p(j\omega) = Y_d(e^{j\omega T})$, which is shown as the product of $H_p(j\omega)$ and $X_p(j\omega)$ in (e).
- Finally, lowpass filtering returns $Y_c(j\omega)$, given as the product of $H_c(j\omega)$ and $X_c(j\omega)$ in (f).

Equivalent LTI system. In comparing Figures 8(a)&(f), we see that the relationship between the output and input spectra can be written as

$$Y_c(j\omega) = X_c(j\omega)H_d(e^{j\omega T})$$

which is simply a frequency scaling by T applied to H_d. Consequently, for inputs that are sufficiently band-limited (i.e., so that $\omega_M < \omega_s/2$), the overall system in Figure 14 is equivalent to a continuous-time LTI system

with frequency response

$$H_c(j\omega) = \begin{cases} H_d(e^{j\omega T}) & |\omega| < \omega_s/2 \\ 0 & |\omega| > \omega_s/2 \end{cases}$$

It is important to note, however, that this only applies to inputs that are band-limited: otherwise the aliasing in the input would carry through to the output, and taking a single period of the discrete-time filter would not be equivalent.

This leads us to a three step design process for the digital filter component of Figure 14:

0. Start with $H(j\omega)$, the desired end-to-end frequency response in continuous-time. Calculate its corresponding impulse response $h(t)$ if needed.
1. For a given sampling period T and frequency ω_s, truncate $H(j\omega)$ outside of $(-\omega_s/2, \omega_s/2)$ to obtain the band-limited $H_c(j\omega)$ that will be implemented.
2. Apply frequency scaling to $H_c(j\omega)$ to obtain the discrete $H_d(e^{j\omega})$. $H_d(e^{j\omega})$ will be periodic outside of the interval $(-\pi, \pi)$.
3. Invert $H_d(e^{j\omega})$ to obtain the impulse response $h_d[n]$ that will be implemented.

We will next work through a few examples to illustrate this process.

Example 2. *Design a **digital differentiator**, which will implement the operation*

$$y(t) = \frac{dx(t)}{dt}$$

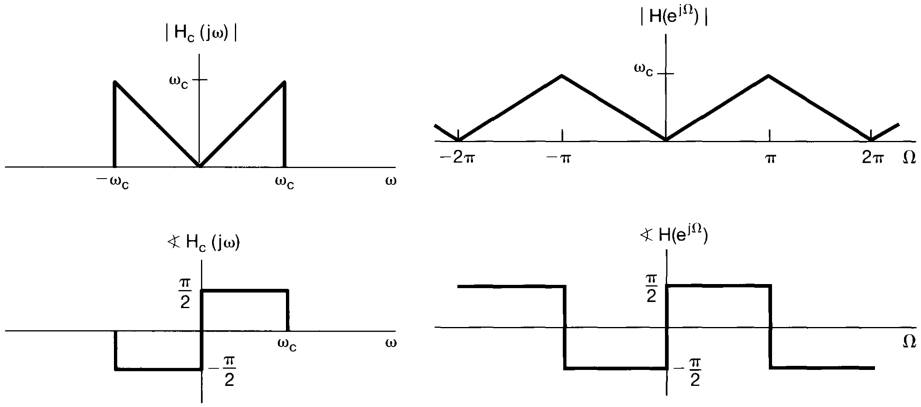
in discrete-time according to Figure 14.

Ans: We start by finding the desired frequency response. From the differentiation property, we know

$$H(j\omega) = \frac{Y(j\omega)}{X(j\omega)} = \frac{j\omega X(j\omega)}{X(j\omega)} = j\omega$$

Next, assuming a sampling frequency ω_s, we truncate (band-limit) $H(j\omega)$:

$$H_c(j\omega) = \begin{cases} j\omega & |\omega| < \omega_s/2 \\ 0 & else \end{cases}$$



(a) Band-limited $H_c(j\omega)$. (b) Discrete-time $H_d(e^{j\Omega})$.

Figure 16: Frequency response of (a) truncated continuous-time and (b) discrete-time filter used to implement a digital differentiator.

Applying frequency scaling, we get the frequency response of the discrete-time processor:

$$H_d(e^{j\omega}) = \begin{cases} j\omega/T & |\omega| < \pi \\ \textit{periodic with} & \textit{period } 2\pi \end{cases}$$

These frequency responses are plotted in Figure 16, assuming $\omega_s = 2\omega_c$.

Finally, we invert $H_d(e^{j\omega})$. *This requires integration by parts, for which we omit the details:*

$$h_d[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi}(j\omega/T)e^{j\omega n}d\omega = \begin{cases} \frac{1}{nT}(-1)^n & n \neq 0 \\ 0 & n = 0 \end{cases}$$

With the impulse response $h_d[n]$, we can implement our digital filter in Figure 14 by having it compute the convolution:

$$y_d[n] = x_d[n] * h_d[n]$$

Then, after passing through the ideal LPF, and scaling by T, we obtain our continuous-time output $y(t)$.

How good will this reconstruction be? Referring to Example 2, so long as $x(t)$ is bandlimited and ω_s is chosen large enough, we can get a perfect reconstruction, i.e., $y(t) = \frac{dx(t)}{dt}$ exactly. However, $h_d[n]$ in Example 2 technically

has an infinite number of non-zero samples. We can compare it to a "naive" approximation of the derivative as the slope between adjacent points:

$$y_d[n] = \frac{x_d[n-1] - x_d[n+1]}{2}$$

which has the impulse response

$$h_d[n] = \begin{cases} \frac{1}{2} & n = 1 \\ -\frac{1}{2} & n = -1 \\ 0 & \text{else} \end{cases}$$

This is less costly to implement (we only need to store two points), but gives us only a crude reconstruction.

Example 3. *Design a **digital time-shift** (delay), which will implement the operation*

$$y(t) = x(t - \Delta)$$

in discrete-time for a shift Δ.

Ans: From the time shift property, we know that

$$H(j\omega) = e^{-j\Delta\omega}$$

Truncating $H(j\omega)$, we have

$$H_c(j\omega) = \begin{cases} e^{-j\Delta\omega} & |\omega| < \omega_s/2 \\ 0 & else \end{cases}$$

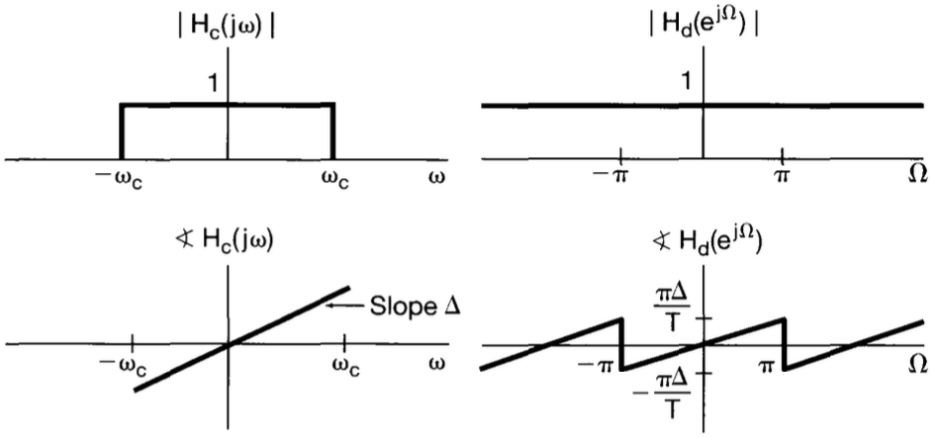
Applying frequency scaling,

$$H_d(e^{j\omega}) = \begin{cases} e^{-j\Delta\omega/T} & |\omega| < \pi \\ periodic\ with & period\ 2\pi \end{cases}$$

These frequency responses are visualized in Figure 17.

Finally, we invert $H_d(e^{j\omega})$:

$$h_d[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{-j\Delta\omega/T} e^{j\omega n} d\omega = \frac{1}{2\pi}\frac{e^{j\omega(n-\Delta/T)}}{j(n-\Delta/T)}\Big|_{-\pi}^{\pi} = \frac{\sin\left(\pi(n-\Delta/T)\right)}{\pi(n-\Delta/T)}$$



(a) Band-limited $H_c(j\omega)$. (b) Discrete-time $H_d(e^{j\Omega})$.

Figure 17: Frequency response of (a) truncated continuous-time and (b) discrete-time filter used to implement a digital time shift.

For appropriately band-limited inputs, the output of the system in Figure 14 using the digital filter in Example 3 will be a perfectly delayed replica of the input. If Δ/T is an integer, this is easy to see, as we can apply the time shifting property to $H_d(e^{j\omega})$ to obtain

$$y_d[n] = x_d[n - \Delta/T]$$

which just shifts the sample index n.

So then, why not just implement this shifting from $x_d[n]$ to $y_d[n]$ directly? Consider what happens when Δ/T is not an integer, which may very well happen as T does not even have to be an integer. In such cases, our discrete time shift has no meaning. Therefore, to accommodate a more general Δ/T, our only choice is to implement $x_d[n] * h_d[n]$. In fact, when $\Delta/T = 1/2$, we get a special case often referred to as a **half-sample delay**:

$$h_d[n] = \frac{(-1)^{n+1}}{\pi(n - \frac{1}{2})}$$