

8.2B Digital-to-Analog Conversion

D/A = Passing $x_s(t)$ through an analog low-pass filter. Mathematically:

$$\hat{X}(s) = X_s(s)H(s) \Leftrightarrow \begin{aligned} \hat{x}(t) &= \int_{-\infty}^{\infty} x_s(\tau)h(t-\tau)d\tau \\ &= \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x(nT)\delta(\tau-nT)h(t-\tau)d\tau \\ &= \sum_{n=-\infty}^{\infty} x(nT)h(t-nT) \end{aligned}$$

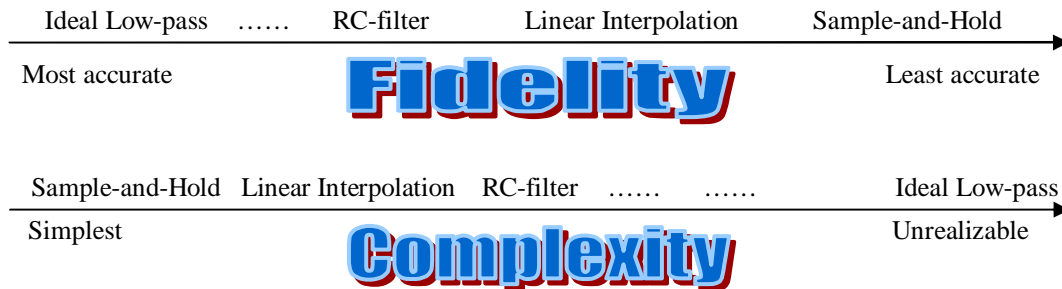
The estimate $\hat{x}(t)$ at time t is based on a linear combination of all samples $x(nT)$.

Two factors:

1. How accurate can the continuous-time signal be reconstructed?
2. How complex is it to realize?

We will look at four methods briefly

1. Ideal low-pass
2. Sample-and-Hold
3. Linear Interpolation
4. General low-pass filter (e.g. RC filter)



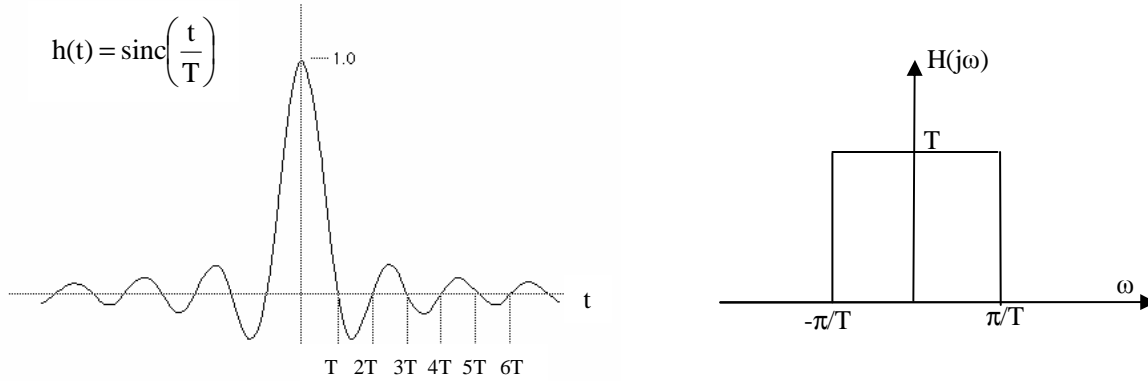
1. Ideal Low-Pass filter

Idea: In frequency domain, multiply by a brick wall filter

$$H(j\omega) = \begin{cases} T & |\omega| < 0.5\omega_s \\ 0 & \text{Otherwise} \end{cases}$$

is the same as convolving with a *sinc filter* in the time domain

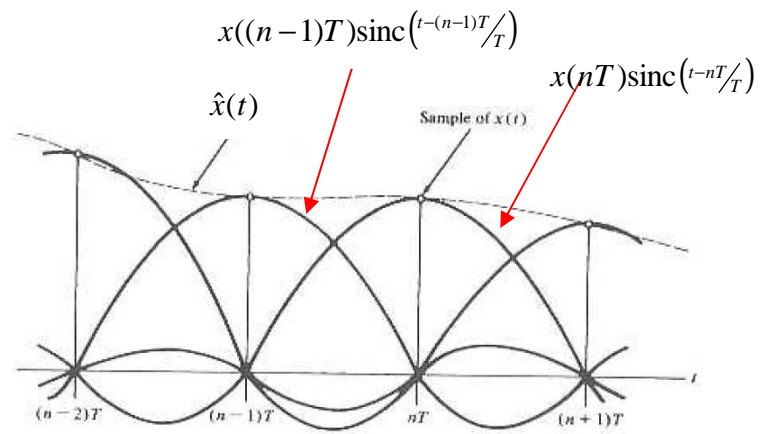
$$h(t) = \frac{\sin(\omega_s t / 2)}{(\omega_s t / 2)} = \text{sinc}\left(\frac{\omega_s}{2\pi} t\right) = \text{sinc}\left(\frac{\omega_s}{2\pi} t\right)$$



Interpretation in time-domain:

$$\hat{x}(t) = \sum_{n=0}^{\infty} x(nT) \text{sinc}\left(\frac{t-nT}{T}\right)$$

Diagram on right shows the contributions from each term (only four terms are shown)
 Notice:
 1. At integral number of T (i.e. (n-1)T, nT), only one term has non-zero contribution.
 2. For all other t, $\hat{x}(t)$ has contributions from all other samples.

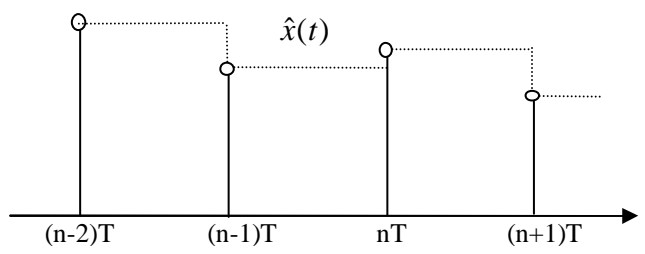


Why is it impossible to implement “real-time” sinc filter?

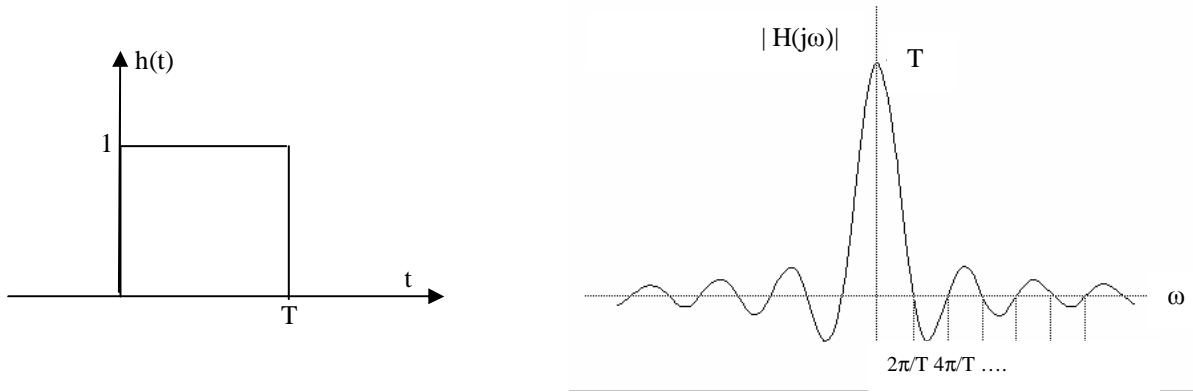
It goes from $-\infty$ to $+\infty$, which means we need to know $x(nT)$ from $-\infty$ to $+\infty$. Not possible unless the signal is finite duration and stored.

2. Sample-and-Hold

Idea:



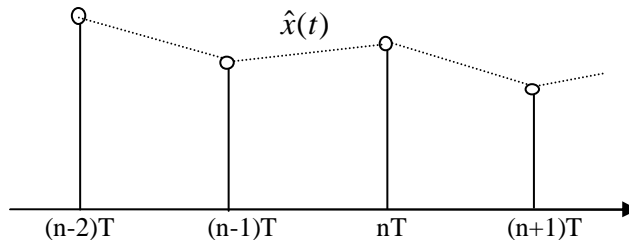
It is easy to see that the interpolation filter must look like:



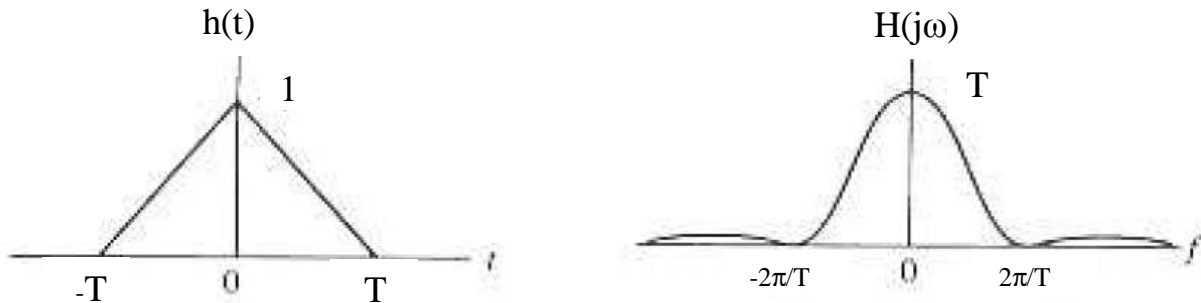
As the frequency spectrum of the sample-and-hold is a sinc function, many high frequency components remains after interpolation. Though very simple to implement, this is not a very effect interpolation filter.

3. Linear Interpolation

Idea:

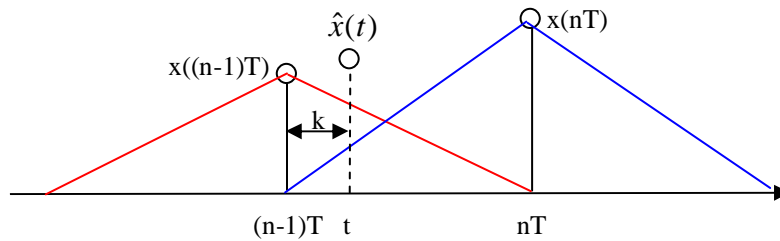


The interpolation filter looks like:



It is a better filter than sample-and-hold as less high-frequency components can pass through. There is still some low-frequency distortion.

Time-domain interpolation: let's say $t = (n-1)T + k$ with $0 < k < T$



Contribution from $x((n-1)T) = x((n-1)T) \frac{T-k}{T}$ (Red part)

Contribution from $x(nT) = x(nT) \frac{k}{T}$ (Blue part)

and there are no contribution from any other samples. Thus

$$x((n-1)T + k) = x((n-1)T) \frac{T-k}{T} + x(nT) \frac{k}{T}$$

- a linear combination of the closest two samples, weighted based on how close they are to the point of interpolation.

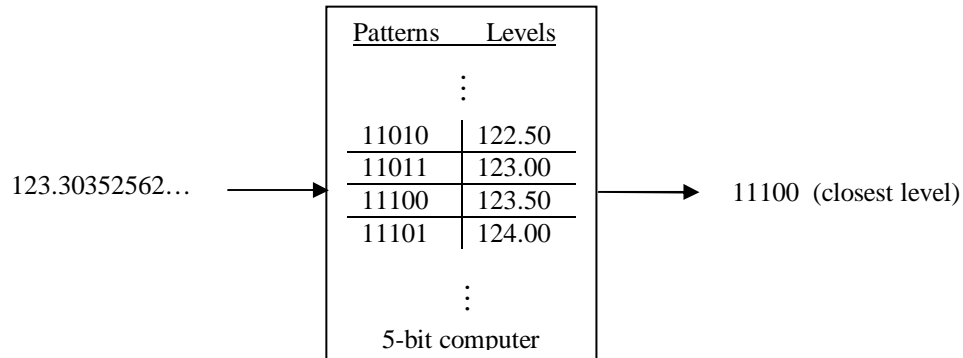
4. General filter

In practice, we can design a better filter using proper analog filter design.

To ensure that the high frequency components are removed, the pass-band frequency ω_p must be smaller than half of the Nyquist rate or $\frac{1}{2}\omega_s$.

The text (page 382) lists the first order Butterworth filter but you can use anyone, depending on the desired fidelity and complexity.

8.2C Quantization



Quantization represents a continuous real number (ex. 123.30352562...) using the discrete level, among 2^n (ex. $n=5$) different levels, closest to the input (ex. 123.50).

Unlike sampling, quantization almost always induces loss in precision. If the input value is very large, we may not care about a loss in the 10^{th} decimal place.

To quantify this “relative” loss, we use the Signal-to-(quantization)-Noise Ratio (SNR) defined as follows:

$$SNR = \frac{\text{Average signal power}}{\text{Average power of quantization error}} \Big|_{\text{in dB}}$$

Obviously, a large SNR indicates high fidelity.

Quantization Error, $e = 123.30352562 - 123.50 = -0.19647438$

This error must be between $-\Delta/2$ and $\Delta/2$, where Δ is the interval between successive levels (ex $\Delta=0.5$) Δ depends on two parameters:

1. Number of levels, i.e. 2^n
2. The dynamic range (max – min) of the input signal. Let’s denote as D .

Then,
$$\Delta = \frac{D}{2^n} = D2^{-n}$$

Assume the quantization error e is uniformly distributed between $-\Delta/2$ and $\Delta/2$.

$$\begin{aligned} & \text{Average power of quantization error} \\ &= E[e^2] \\ &= \int_{-\Delta/2}^{\Delta/2} e^2 \frac{1}{\Delta} de \\ &= \frac{\Delta^2}{12} = \frac{D^2 2^{-2n}}{12} \end{aligned}$$

If the signal power is P_s , then

$$\begin{aligned} SNR &= 10 \log \left(\frac{P_s}{D^2 2^{-2n} / 12} \right) \\ &= 10 \log 12 + 10 \log P_s - 20 \log D + 20n \log 2 \end{aligned}$$

Thus, SNR can be improved by

- raising the signal power, P_s
- reducing the signal dynamic range, D
- increase the number of bits, n

“6-dB rule” : it’s convenient to remember that every additional bit will improve the SNR by $20 \log 2 \approx 6$ dB