EELE 477
Digital Signal Processing

4

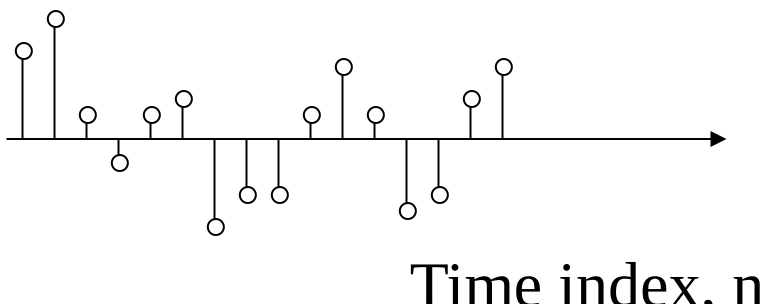
Sampling; Discrete-Time

Sampling a Continuous Signal

- Obtain a sequence of signal *samples* using a periodic instantaneous sampler:

$$x[n] = x(nT_s)$$

- Often plot discrete signals as dots or “lollypops”:



Sampling a Sinusoid

- Discrete time sinusoid via sampling:

$$x[n] = x(nT_s) = A\cos\left(\omega nT_s + \phi\right) = A\cos\left(\hat{\omega} n + \phi\right)$$

- Note that A, cos, ϕ are the same.
- Discrete-time radian frequency:

$$\hat{\omega} = \omega T_s$$

- Note that T_s cannot be deduce from x[n] alone!

Reconstruction??

- It *is* possible to reconstruct a continuous-time signal from its discrete-time samples, but with restrictions.
- The *sampling theorem* states that a signal can theoretically be reconstructed from its samples as long as

$$f_s = \frac{1}{T_s} \geq 2 f_{\max}$$

Sampling Rate

- In short, we must sample at a rate at least double the highest frequency component present in the continuous-time signal.
- This minimum sampling rate is called the *Nyquist rate*.
- Result: continuous-time signal must be *bandlimited* prior to sampling in order to allow perfect reconstruction.

Aliasing

- What happens if we don't obey Nyquist?
- Consider two signals:

$$x(t) = A\cos\left(2\pi f_0 t + \phi\right)$$
$$y(t) = A\cos\left(2\pi\left(f_0 + f_s\right)t + \phi\right)$$

(same amplitude and phase, different freq)

Aliasing (cont.)

- Now sample with period T_s:

$$x[n] = A\cos(2\pi f_0 n T_s + \phi)$$

$$y[n] = A\cos(2\pi(f_0 + f_s)nT_s + \phi)$$

$$= A\cos\left(2\pi f_0 n T_s + \underbrace{2\pi f_s n T_s}_{2\pi n} + \phi\right)$$

$$= A\cos(2\pi f_0 n T_s + \phi) = x(t)$$

Aliasing (cont.)

- Note that the <u>same</u> sampled sequence occurs for both x(t) and y(t) even though they have different frequencies: one signal is an *alias* of the other.
- Further, note that infinite number of aliases since same discrete-time sequence for: $f = f_0 \pm k f_s,\ k = 0,1,2,...$

Folding

- Also can find aliases corresponding to the *negative frequency* components:

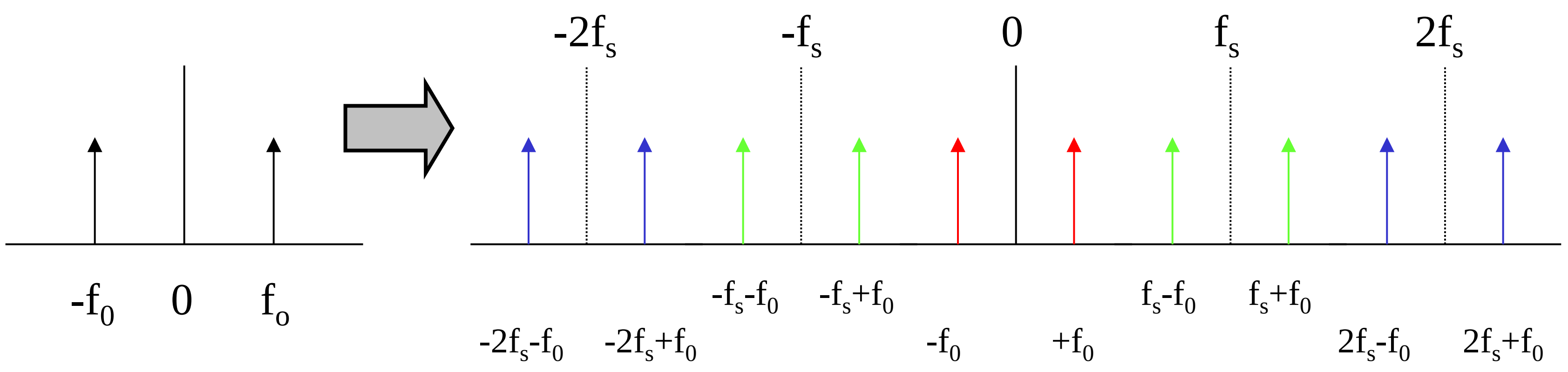
$$w(t) = A\cos\left(2\pi\left(-f_0 + kf_s\right)nT_s - \phi\right)$$

$$= A\cos\left(-2\pi f_0 nT_s + \underbrace{2\pi f_s nT_s}_{2\pi n} - \phi\right)$$

$$= A\cos\left(2\pi f_0 nT_s + \phi\right) = x(t)$$

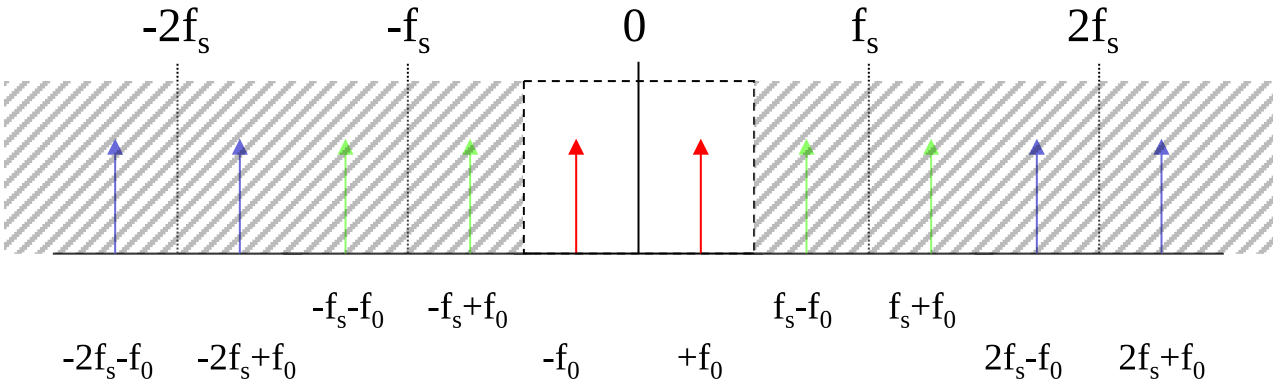
Spectral View of Sampling

- The effect of sampling is to create *images* of the continuous-time spectrum centered at multiples of the sampling frequency:



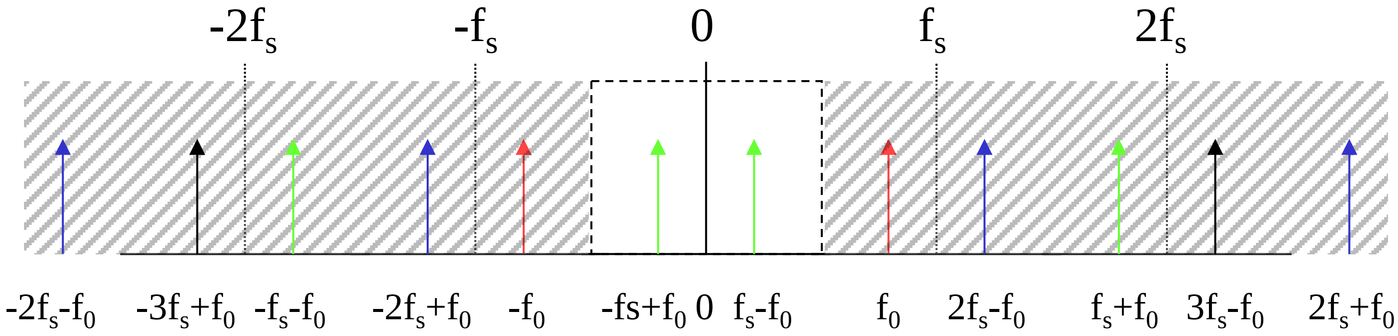
Spectral View (cont.)

- We can reconstruct the continuous signal by removing (filtering) the images and keeping the *baseband* image:



Aliasing

- What if $f_0 > f_s/2$? Sampling still creates images, but now the *baseband* image is *not* the expected original signal, but actually *aliases*.

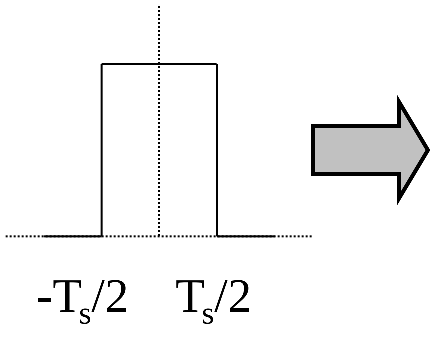


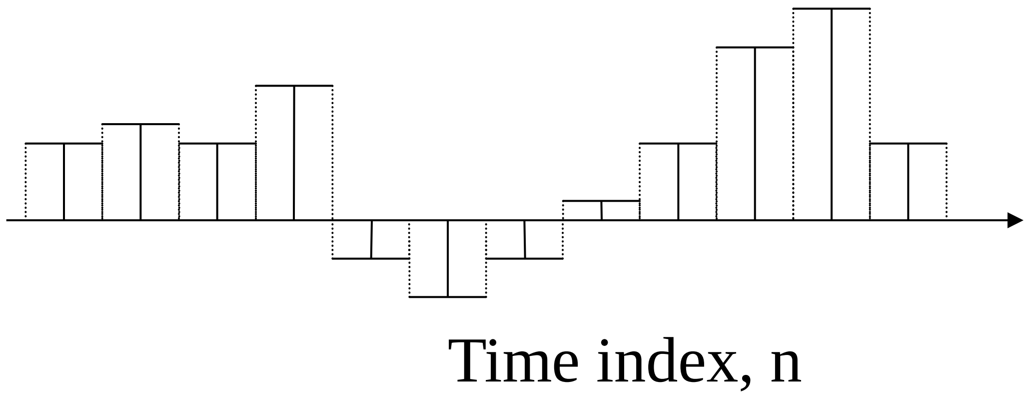
Reconstruction==Interpolation

- The reconstruction process can be thought of as *interpolating* between the discrete-time samples.
- Various interpolation approximations can be considered: “hold” last value, “connect the dots” (linear), fit a smooth polynomial curve, etc.
- Optimal reconstruction requires a process that retains only the baseband: a perfect *lowpass* filter.

Concept: Pulse-overlap Interpolation

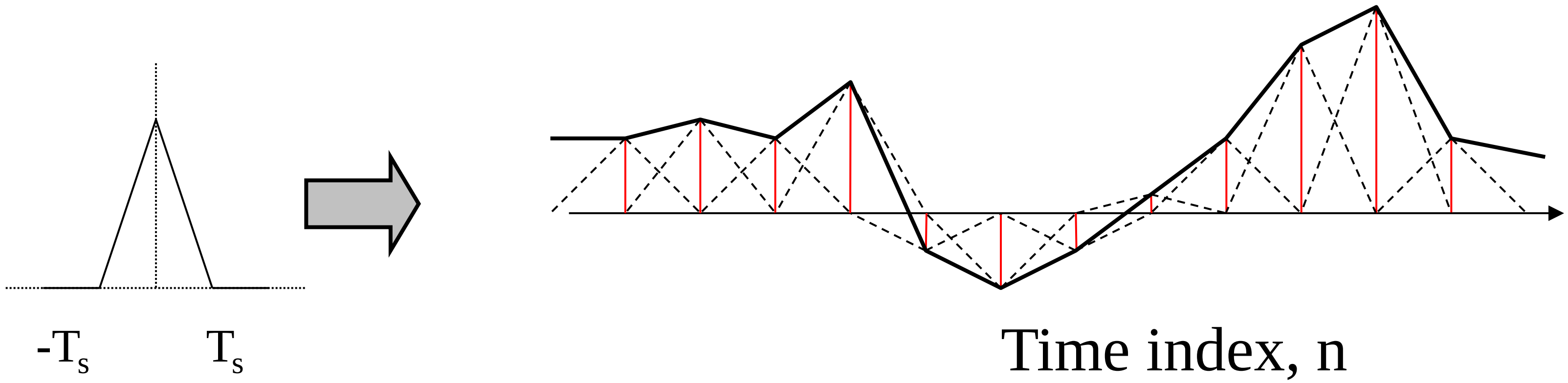
- Consider constructing the continuous waveform by shifting and scaling a set of pulses—one centered per discrete-time sample—then sum them all up.





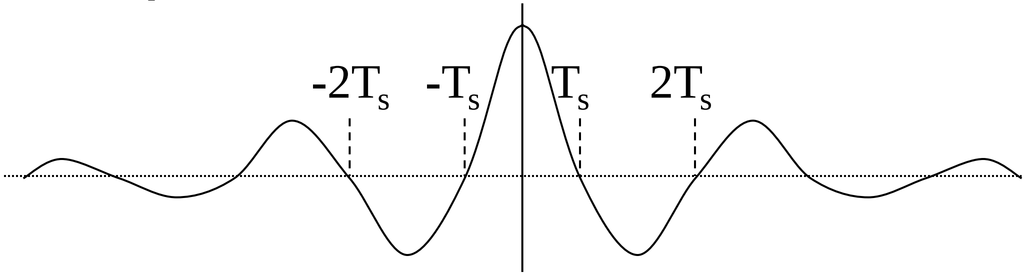
Pulse Overlap (cont.)

- Triangular pulse = linear interpolation
- Similar for higher-order interpolation



Reconstruction via Filtering

- The pulse overlap scheme implements time domain convolution.
- Time domain convolution is equivalent to frequency domain multiplication
- We want a perfect rectangle (low pass) in the frequency domain: this corresponds to a *sinc* pulse in time domain:



$$p_{ideal}(t) = \frac{\sin \frac{\pi}{T_s} t}{\frac{\pi}{T_s} t}$$

Oversampling

- Interpolation is easier of samples are close together: T_s is very small
- Small T_s means very high f_s
- From a spectral viewpoint, this *oversampling* means that $f_{max} << f_s$