[10:30am] Announcements and takeaways from last lecture.

- Prelab quiz may contain duplicate questions. Answer as normally. The issue will be fixed for the next quiz.
- Bandwidth: Finite observation time and thermal noise lead to infinite bandwidth.
- Modulation: Multiply signal by sinusoid in the time domain. Shifts the signal in the frequency domain. Converts from a lowpass signal to a bandpass signal which is ideal for transmission.
- Demodulation: Modulate again, then apply a lowpass filter. Returns the signal to baseband (lowpass). The sinusoids used for modulation and demodulation must have the same phase; otherwise, a loss of amplitude will occur.
- Sine-modulated and cosine-modulated signals are orthogonal. Both can be transmitted in the same frequency band and perfectly recovered to double the efficiency.

[10:40am] Sinusoidal generation

- Three methods to generate a discrete-time sinusoid. Different tradeoffs in runtime implementation complexity vs signal quality. If possible, we prefer to perform some calculation offline to reduce the load at runtime.
  - Math library
  - Difference equation
  - Lookup table
- Goal: generate the discrete-time cosine

  $$x[n] = \cos(\omega_0 n)$$

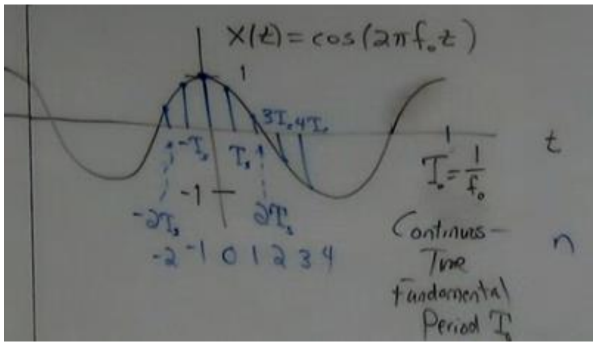
  Where ω_0 has units of radians per sample. Then, send to D/A converter to produce

  $$x(t) = \cos(2\pi f_0 t)$$

  Continuous-time cosine sampled at points in time uniformly spaced by $T_s = \frac{1}{f_s}$ sec.

  $$x[n] = \text{Sample}\{x(t)\} = x(t)|_{t=nT_s}, \text{ where } n \text{ is an integer.}$$

  $$\text{sample}\{\cos(2\pi f_0 t)\} = \cos(2\pi f_0 n T_s) = \cos \underbrace{\frac{2\pi f_0}{f_s}}_{\omega_0} n$$



- Example. $f_0 = 1200$. $f_s = 8000$ Hz (a common sampling rate for speech signals.)

$$\omega_0 = 2\pi \frac{1200 \frac{\text{cycles}}{\text{second}}}{8000 \frac{\text{samples}}{\text{second}}} = 2\pi \frac{3}{20} \frac{\text{radians}}{\text{sample}}$$

- Sampling theorem. If $f_s > 2f_{\max}$, then we can sample and reconstruct the signal. Otherwise, if $f_s \leq 2f_{\max}$ the recovered signal may not match the original signal.
- It is not realistic that $x(t)$ has no frequency content above $f_{\max}$ (thermal noise, finite observation time, etc). There will always be aliasing in practice. However, we can try to control this aliasing, for example with an analog lowpass filter.
  Sampling can be modeled by multiplication with $Ш_T(t) = \sum_{k=-\infty}^{\infty} \delta(t - kT)$, an impulse train with period T seconds. (Homework 0.3)

$$\text{sample}\{x(t)\} = x(t)Ш_T(t)$$

$$\mathcal{F}\{Ш_T(t)\}(f) = \frac{1}{T} Ш_{\frac{1}{T}}(f)$$

$$\mathcal{F}\{Ш_T(t)\}(\omega) = \frac{2\pi}{T} Ш_{2\pi/T}(\omega)$$

  If a signal is sampled in the time domain, it is equivalent to periodic replication in the frequency domain

$$\begin{aligned}\mathcal{F}\{x(t)Ш_T(t)\} &= X(f) * \frac{1}{T} Ш_{\frac{1}{T}}(f) \\ &= \frac{1}{T} \sum_{k=-\infty}^{\infty} X(f - kf_s)\end{aligned}$$

[11:15am] Discussion after break

- Sampling a signal from a microphone: In an ADC, there is typically an analog lowpass filter to enforce the condition that $f_s > 2f_{\max}$.
- In the DAC, there will also be a lowpass filter used for reconstruction.

[11:25] Sinusoidal Generation

- C math library cos function `gsl_sf_cos_e(double θ)`
  - Double precision accuracy is often overkill. Single precision is typically sufficient for most signal processing tasks.
  - Typical implementation: Approximate cosine using 11th order polynomial.
  - Use symmetry of cosine and sine. Only approximate the first 1/8th of cosine and first 1/8th of sine.
  - Horner's form can be used to simplify the calculation of the polynomial
  - High accuracy, but also high complexity (even using Horner's form)

- Difference Equation
    - Implement a discrete-time LTI filter that obeys the following difference equation (Homework 0.4)
    $$y[n] = 2\alpha y[n-1] - y[n-2] + x[n] - \alpha x[n-1]$$
    where the constant α is related to the desired frequency ω_0 by $\alpha = 2\cos\omega_0$.
    - The impulse response of this system is a one-sided cosine
    - Feed an impulse into the system, and it will generate a cosine!
    - Much lower complexity per sample than math library
    - Drawback: We must periodically "restart" the system by resetting the initial conditions and sending a new impulse to the input. Otherwise, accumulation of quantization error will gradually reduce the quality of the generated sinusoid.
- Lookup table
    - Precompute one discrete-time period of the cosine. (This may require several, or even an infinite number, of continuous-time periods)
    - Example: $f_0 = 1200$ Hz. $f_s = 8000$ Hz.
    $$\omega_0 = \frac{2\pi f_0}{f_s} = 2\pi\frac{1200\text{ Hz}}{8000\text{ Hz}} = 2\pi\frac{3}{20} = 2\pi\frac{N}{L}$$
    Requires lookup table of length $L = 20$. $N = 3$ Continuous-time periods will occur within one discrete-time period.
- Summary of methods:
    - Math library: high accuracy, high complexity that does not change based on ω_0
    - Difference equation: Must be periodically reset. low complexity that does not change based on ω_0
    - Lookup table: High quality, low computational complexity. Potentially high memory requirement that depends on ω_0.