What: Varying the sampling rate without resampling continuous-time signal.
Why: D/A conversion, digital filter design, audio processing (see below).
How: Downsampling=Decimation=reduces sampling rate & increases freqs.
Upsampling=Interpolation=increases sampling rate & decreases freqs.

**Downsampling:** \( x[n] \rightarrow \downarrow 2 \rightarrow y[n] \)
This doubles frequencies of \( x[n] \).

Matlab: 
\[ L=\text{length}(X); Y=X(1:2:L); \]
Takes every other sample; halves length.
Exactly as if the original sampling rate had been reduced by a half.

**Spectrum:**
\( x[n]=\cos(\omega_0 n) \rightarrow y[n]=x[2n]=\cos((2\omega_0)n). \)
Doubles all frequencies.

\[
Y(e^{j\omega}) = \sum x[2n]e^{-j\omega(2n)/2} = \sum x[n] \frac{1}{2}[1+(-1)^n]e^{-j\omega n/2} = \frac{1}{2}X(e^{j\omega/2}) + \frac{1}{2}X(e^{j(2\pi + \pi/2)})
\]
Need: \( \omega_0 < \frac{1}{4}2\pi \) (sampling rate) \( \rightarrow 2\omega_0 < \frac{1}{4}2\pi \) (sampling rate) or get aliasing.
Note: Downsample by \( N \rightarrow Y(e^{j\omega}) \) period=2\pi N. Add copies to make 2\pi.

**Upsampling:** \( x[n] \rightarrow \uparrow 2 \rightarrow y[n] \)
This halves frequencies of \( x[n] \).

Matlab: 
\[ XX=[X;zeros(1,L)]; Y=XX(:); \]
Inserts zeros between samples.

**Upsample and Interpolate:** \( x[n] \rightarrow \uparrow 2 \rightarrow y[n] \rightarrow \text{LPF} \rightarrow z[n] \)

Matlab: 
\[ F=\text{fft}(Y); Z=\text{real}(\text{ifft}(F([1:L/4+1 \ zeros(1,L/2-1) \ 3*L/4+1:L]))); \]

**Spectrum:**
\( y[n] = \begin{cases} x[\frac{n}{2}] & \text{for } n \text{ even} \\ 0 & \text{for } n \text{ odd} \end{cases} \)
\( x[n]=\cos(\omega_0 n) \rightarrow y[n]=\frac{1}{2} \cos(\omega_0 n) + \frac{1}{2} \cos((\pi - \omega_0) n) \) (modulation)

\[
Y(e^{j\omega}) = \sum_{n \text{ even}} x[\frac{n}{2}]e^{-j\omega 2n/2} = \sum x[n']e^{-j\omega 2n'} = X(e^{j2\omega}) \quad \text{where } n' = \frac{n}{2}.
\]
Note: If Upsample by \( N \rightarrow Y(e^{j\omega}) \) periodic with period=2\pi N < 2\pi.
So: Always LPF (LowPass Filter) to reject \( \omega > \frac{\pi}{N} \) after upsampling by \( N \).

**Multirate filtering:** \( x[n] \rightarrow \uparrow 3 \rightarrow \text{LPF} \rightarrow \downarrow 2 \rightarrow y[n] \)

What: Reduces all frequencies to 2/3 previous values. Using \( f=(k-1)F/N \):
Matlab: 
\[ F=\text{fft}(X); Y=\text{real}(\text{ifft}(F([1:L/2+1 \ zeros(1,L/2-1) \ L/2+1:L]))); \]
Note: Upsample by 3 first; then downsample by 2. Otherwise: aliasing.
Note: If increasing frequency, need ensure resulting frequency unaliased.

**EX#1:** 600 Hz → A/D → \( \uparrow 3 \) → D/A → ?. Sampling rate=2400 Sampler/Second.
Soln: \( \frac{1}{3}\{600,2400–600,2400+600\} = \{200,600,1000\} \) Hz. Need LPF for 200.

**EX#2:** 300 Hz → A/D → \( \uparrow 3 \) → D/A → ?. Sampling rate=2400 Sampler/Second.
Soln: 3(300)=900 Hz < \( \frac{1}{2} \) (2400) so no aliasing. 900 Hz only. LPF unneeded.

**EX#3:** 500 Hz → \( \downarrow 3 \) → \( \uparrow 2 \) → LPF → ?. Sampling rate=2400 Sampler/Second.
Soln: 500→1500 aliased to 900→ \( \frac{1}{2} \{900,2400–900\} = \{450,750\} \) Hz. No good.

**EX#4:** 500 Hz → \( \uparrow 2 \) → LPF → \( \downarrow 3 \) → ?. Sampling rate=2400 Sampler/Second.
Soln: \( \frac{1}{2}\{500,2400–500\} = \{250,950\} \). LPF leaves only 250→750 only. OK!
Note: Upsample+LPF+downsample in that order for multirate filtering.
APPLICATIONS OF MULTIRATE FILTERING

EX#1: Audio: Alter the pitch (i.e., period or frequency) of music or speech. Generate all 12 semitones from a single snippet using Circle of Fifths.

EX#2: D/A: Given Nyquist-sampled, reconstruct from samples (A/D). But: Need ideal analog LPF for interpolation (D/A)! (no can do).

Idea: Upsample & interpolate sampled signal→now its oversampled.
Then: Analog LPF for interpolation (D/A) can be lousy (realistic).
Point: Instead of ideal analog LPF, use ideal digital LPF (can do).

CD: $x[n] \rightarrow \uparrow 4 \rightarrow h[n] = \frac{\sin(\pi n/4)}{\pi n} \rightarrow \text{0-ORDER HOLD INTERPOLATOR} \rightarrow x(t)$

Point: This effectively increases the sampling rate to $4(44.1kHz)=176.4$ Hz.
Thus: “4x oversampling” (audio term) → can use 0-order hold for D/A.
Also: 3/4 of upsampled $x[n]$ are zeros→$h[n]$ convolved with mostly zeros.

DIGITAL FILTER DESIGN USING MULTIRATE FILTERING

Goal: Digital LPF: $H_D(e^{i\omega})=1$ for $\omega < 0.0095\pi$; $H_D(e^{i\omega})=0$ for $\omega > 0.01\pi$.
Problem: Sharp transition; the equiripple FIR filter has very long duration.
Idea: Use 3 shorter-duration FIR filters and downsample in between them.

Use: $x[n] \rightarrow \boxed{H_1(e^{i\omega})} \rightarrow \downarrow 10 \rightarrow z[n] \rightarrow \boxed{H_2(e^{i\omega})} \rightarrow \uparrow 10 \rightarrow \boxed{H_3(e^{i\omega})} \rightarrow y[n]$

where: $H_1 = \begin{cases} 1 & \omega < 0.0095\pi \\ 0 & \omega > 0.1\pi \end{cases}$; $H_2 = \begin{cases} 1 & \omega < 0.095\pi \\ 0 & \omega > 0.1\pi \end{cases}$; $H_3 = \begin{cases} 1 & \omega < 0.01\pi \\ 0 & \omega > 0.1\pi \end{cases}$.

Why: Sharp LPF with large stopband replaced with dull LPF which becomes a sharp large-stopband LPF after upsampling compresses spectrum.

How: 1st $H_1(e^{i\omega})$ prevents aliasing after $\downarrow 10$ is really an antialiasing filter.
2nd $H_2(e^{i\omega})$ becomes $H_D(e^{i\omega})$ after $\uparrow 10$ compresses spectrum of $z[n]$.
3rd $H_3(e^{i\omega})$ interpolates following $\uparrow 10$ upsample and interpolate $z[n]$. 