Audio Special Effects

Professor Deepa Kundur

University of Toronto

Audio Effects

- **Q:** What is an audio effect?
  - **A:** artificially enhanced sound or sound processes used to emphasize artistic content in films, television, shows, live performance, animation, video, games, music or other media.

Common Audio Special Effects

Two common types:

- Delay-based special effects
  - simple echo
  - reverberation
  - flanging
  - chorus
- Rate-conversion special effects
  - downsampling (decimation)
  - upsampling
  - voice gender changers

Delay-Based Special Effects

Delay Effects

- **Q:** What is a delay effect?
  - **A:** audio effect which records an input signal to an audio storage medium and then plays it back (possibly multiple times) into the recording again to create the sound of a repeating decaying echo.
- **Q:** What is this so popular?
  - **A:** easy to achieve even before the use of computers while adding an attractive texture to the music.

Analog and Digital Delays

- Analog delay
  - created by recording in a naturally reverberant space
  - achieved using tape loops improvised on reel-to-reel magnetic recording systems
    - signal is recorded on analog tape and played back from same piece of tape through the use of two different record and replay heads
    - adjusting loop length and distance between the read and write heads enables control over delayed echo
- Digital delay
  - first introduced in 1984 by Boss Corporation
  - provides great flexibility, portability and programmability

Examples of Delay Effects

Delay-based special effects:

- simple echo
- reverberation
- flanging
- chorus

Note: Check out course website on Handouts page for an example of a simple echo.

Single Echo

- **Q:** How can we achieve a single echo from a given sound signal $x(n)$?
  - **A:** add a delayed and attenuated version of $x(n)$ to itself.

$$y(n) = x(n) + \alpha x(n - n_0)$$

Note: The audio example available on the course web page was generated using $\alpha = 0.35$ and $n_0 = 20000$ with $F_s = 44kHz$. Thus the echo delay is $20000/44000 = 0.45$ sec.

Single Echo

- **Q:** How can we characterize this single echo generation system? *Hint:* The system is linear time-invariant?
  - **A:** impulse response and frequency response.

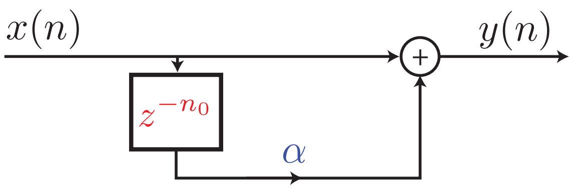
Single Echo: Impulse Response

$$y(n) = x(n) + \alpha x(n - n_0)$$

Let $x(n) = \delta(n)$ to give $y(n) = h(n)$.

$$\therefore h(n) = \delta(n) + \alpha\delta(n - n_0).$$

Single Echo: Filter Implementation



LTI Single-zero system

Note: This is also called a delay line in audio applications and is characterized by n_0 and α.

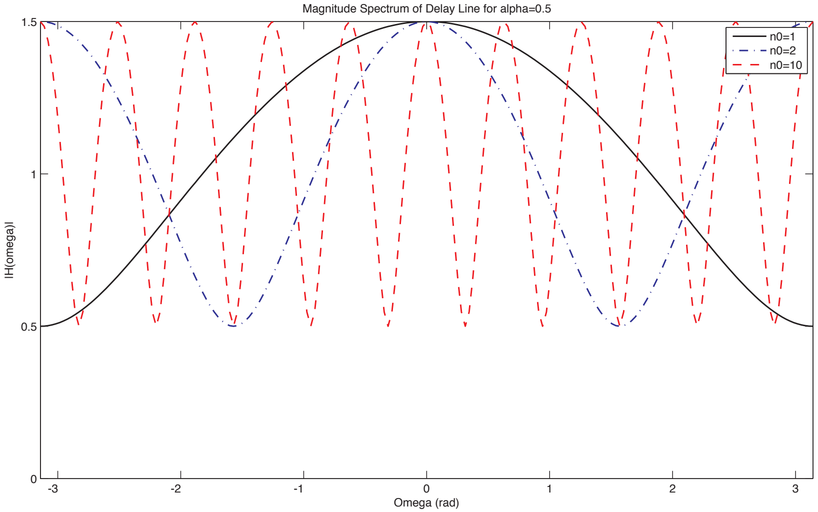
Single Echo: Frequency Response

$$\begin{aligned} h(n) &= \delta(n) + \alpha\delta(n - n_0) \qquad \text{FIR system} \\ H(\omega) &= \sum_{n=-\infty}^{\infty} h(n) e^{-j\omega n} \\ &= \sum_{n=-\infty}^{\infty} [\delta(n) + \alpha\delta(n - n_0)]\, e^{-j\omega n} = 1 + \alpha e^{-j\omega n_0} \\ |H(\omega)| &= \sqrt{1 + \alpha^2 + 2\alpha\cos(\omega n_0)} \end{aligned}$$

Note: $1 - \alpha \leq |H(\omega)| \leq 1 + \alpha$.

Single Echo: Frequency Response

Note: $1-\alpha \leq |H(\omega)| \leq 1+\alpha$; $\alpha = 0.5$ in example.



Extended Echo: Impuse Response

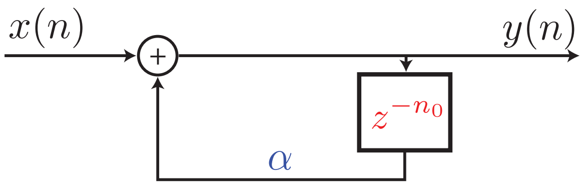
Consider an infinite series of echos geometrically decaying in amplitude and with equally spaced delays:

$$y(n) = x(n) + \alpha x(n-n_0) + \alpha^2 x(n-2n_0) + \cdots$$

Let $x(n) = \delta(n)$ to give $y(n) = h(n)$.

$$\therefore h(n) = \delta(n) + \alpha\delta(n-n_0) + \alpha^2\delta(n-2n_0) + \cdots = \sum_{k=0}^{\infty} \alpha^k \delta(n - k n_0)$$

Extended Echo: Filter Implementation



LTI Single-pole system

Extended Echo: Frequency Response

$$h(n) = \sum_{k=0}^{\infty} \alpha^k \delta(n-kn_0) \qquad \text{IIR system}$$

$$\begin{aligned} H(\omega) &= \sum_{n=-\infty}^{\infty} h(n) e^{-j\omega n} \\ &= \sum_{n=-\infty}^{\infty} \left[\sum_{k=0}^{\infty} \alpha^k \delta(n-kn_0) \right] e^{-j\omega n} \\ &= \sum_{k=0}^{\infty} \sum_{n=-\infty}^{\infty} \alpha^k e^{-j\omega n} \delta(n-kn_0) \\ &= \sum_{k=0}^{\infty} \alpha^k e^{-j\omega k n_0} = \sum_{k=0}^{\infty} (\alpha e^{-j\omega n_0})^k = \frac{1}{1-(\alpha e^{-j\omega n_0})} \end{aligned}$$

for $|\alpha| < 1$. Instability occurs for $\alpha > 1$.

Extended Echo as Reverberation

- Consider an original sound source $x(n)$ of finite duration in the order of a few seconds.
- Specifically, let its time duration be T_d sec and its sample duration be $N_d = \lfloor \frac{T_d}{T} \rfloor = \lfloor T_d \cdot F_s \rfloor$ samples.
- Let the echo generation parameters be $|\alpha| < 1$ and n_0 "small" such that

$$n_0 \cdot T = \frac{n_0}{F_s} \ll 1 \qquad \text{(normally in the order of 0.01 - 1 msec)}$$

Extended Echo as Reverberation

- When the original sound source is present, the echoes overlap first building up the overall sound effect.
  - For a source that is T_d sec in duration,

$$\text{No. Overlapping Echoes} = \left\lfloor T_d \frac{F_s}{n_0} \right\rfloor = \left\lfloor \frac{N_d}{n_0} \right\rfloor \gg 1$$

- After the original source has stopped, the overall sound decays due to the echo reflections that eventually die out due to $\alpha < 1$; sounds like you are in a music hall.

This overall process is a type of reverberation.

Reverberation

Good examples at:

http://www.youtube.com/watch?v=cGBn7sU6m3k

Reverberation

Recall,

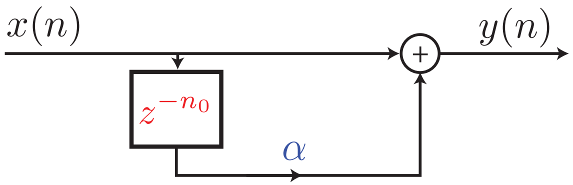
- First the echoes overlap with the original source signal building up the sound effect.
- When the original source has stopped, the sound may temporarily persist and then eventually die out.

There are other ways to achieve a "richer" reverberation than our prior example . . .

Reverberation

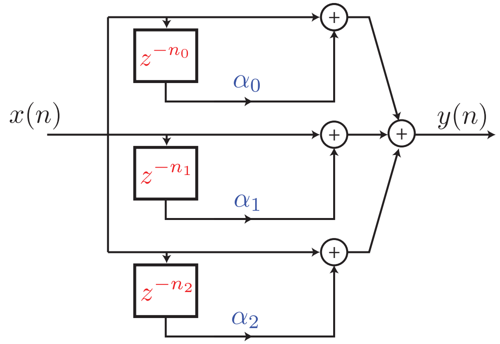
Example: More realistic reverb using multiple delay lines

- Use multiple delay lines with delays that are relatively prime, so that the echoes emanating from each lines do not ever overlap giving a richer sound.
- Single delay line:



Reverberation

Three delay line example:

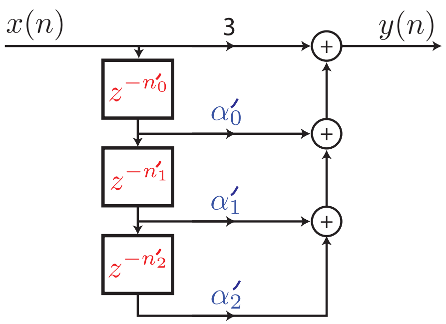


where

- $n_0 > n_1 > n_2$ are relatively prime

Reverberation

Note: the three delay line is equivalent to the following:



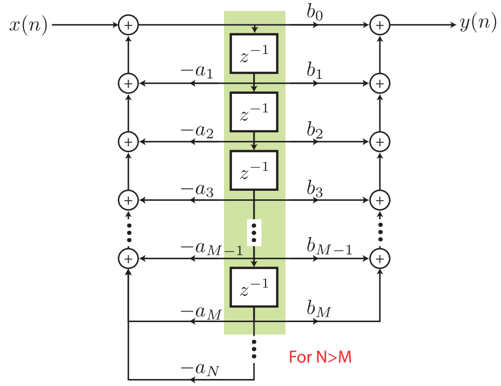
LTI All-zero system

where

- $\alpha_i = \alpha'_i$ for $i = 0, 1, 2$
- $n'_0 = n_0$
- $n'_1 = n_1 - n_0$
  $n'_2 = n_2 - n_1 - n_0$

Reverberation

For a more realistic reverb:



where

- feedforward and feedback present
- poles and zeros provide a more all-pass spectrum for realism
- more parameters to tune or experimentally estimate

Flanging

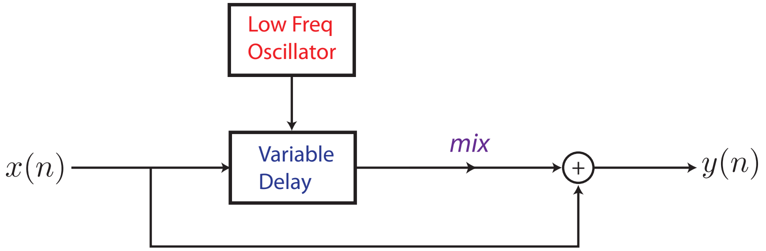
- process of mixing two signal together that are nearly identical such that one signal is a slightly variably delayed version of the other
- manifests like a "swooshing" sound
- a variation of this sound often occurs when instruments are trying to tune to a tuning fork

Flanging

Good examples at:

http://www.youtube.com/watch?v=NAqQvs_WXs8&feature=related

Flanging



- The low frequency oscillator (LFO) controls the delay of $x(n)$ which may change from block to block or even sample to sample.
- The scalar constant *mix* determines the proportion of the delayed signal that is added back to the original source.

Flanging

- Let $d(n)$ be the variable delay for $x(n)$ controlled by the LFO.
- Let the LFO provide the following sinusoidal signal:

$$
\begin{aligned}
d(n) &= \text{round}(\alpha \sin(2\pi f_0 n) + \beta) \\
y(n) &= x(n) + mix \cdot x(n + d(n)) \\
&= x(n) + mix \cdot x(n + \text{round}(\alpha \sin(2\pi f_0 n) + \beta))
\end{aligned}
$$

Flanging

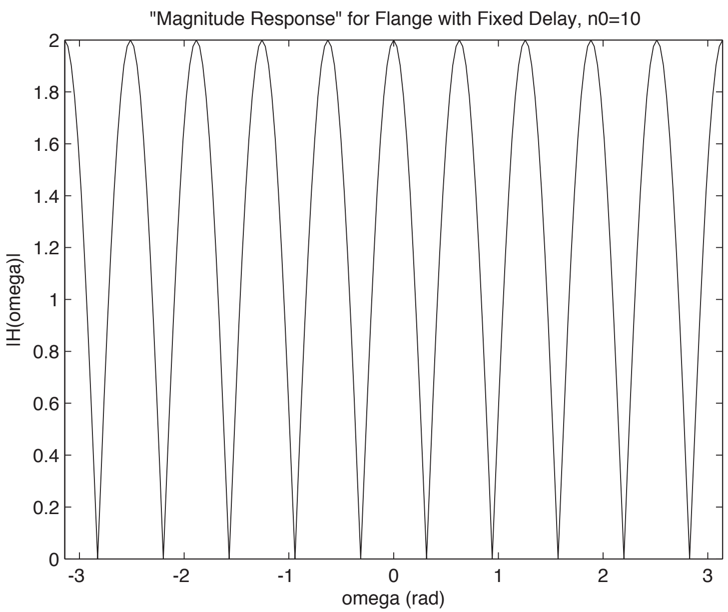
$$y(n) = x(n) + mix \cdot x(n + \text{round}(\alpha \sin(2\pi f_0 n) + \beta))$$

- rate is given by f_0 and is generally small; typically $f_0 \cdot F_s$ should be 0.7 Hz (classical flange sound) up to 6 Hz (slight whammy effect) or even 20 Hz (mechanistic warble effect).
- sweep depth is given by 2α; α should be selected so that the temporal (i.e., refers to seconds not samples) sweep depth is around a couple of milliseconds.
- delay is given by $\beta - \alpha$ and represents the minimum delay reached by the LFO; typically β should be set so that the delay is 1-10 milliseconds; note: human ear will perceive an echo (not flange) if the delay is more than 50-70 milliseconds!

Flanging: Instantaneous "Frequency Response"

Consider fixed delay n_0 and $mix = 1$:

$$\begin{aligned} y(n) &= x(n) + x(n - n_0) \\ Y(\omega) &= X(\omega) + e^{-j\omega n_0} X(\omega) \\ H(\omega) &= \frac{Y(\omega)}{X(\omega)} = 1 + e^{-j\omega n_0} \\ &= 2e^{-j\omega n_0/2} \cos(\omega n_0/2) \\ \therefore |H(\omega)| &= 2|\cos(\omega n_0/2)| \end{aligned}$$

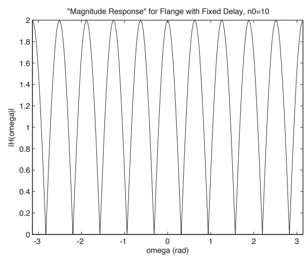


- spectrum nulls occur when argument of the cosine is an odd multiple of π:

$$\omega \frac{n_0}{2} = (2k+1)\pi \quad \text{or} \quad \omega = \frac{2(2k+1)\pi}{n_0}$$

for $k = 0, 1, 2, \ldots$

- If the delay n_0 varies, then so do the spectrum nulls.



Flanging: Instantaneous "Frequency Response"

Thus, one can envision flanging as being the result of changing the position of the nulls of the frequency response.

A cautionary note: the flanging system is not LTI therefore, it's frequency response does not fully characterize it, or we may say it has no frequency response!

Thus, this analysis is just a tool to intuitively explain the flange effect.

From Flange to Chorus

- Overall a classic flange has a delay ranging between 1 - 10 milliseconds.
- To create a chorus effect, this delay range must be between 30 - 50 milliseconds
- A delay above 50 milliseconds will be perceived as an echo.

Chorus

- A chorus effect sounds likes more than one instrument is playing.
- Good examples at:

  http://www.youtube.com/watch?v=ZSL1w9UeSgc

Rate-Conversion Special Effects

Rate-Conversion Special Effects

- Shifting, stretching and/or expanding spectral information across frequency bands can provide interesting effects especially for voice signals.
- Roughly speaking moving spectral content to lower frequencies adds base making a voice sound more male. Similarly, moving spectral content to higher frequency adds treble making a voice sound more female.
- One way to achieve spectral shifts, stretches and expansions is through sampling rate conversion.

Sampling Rate Conversion

Reference:

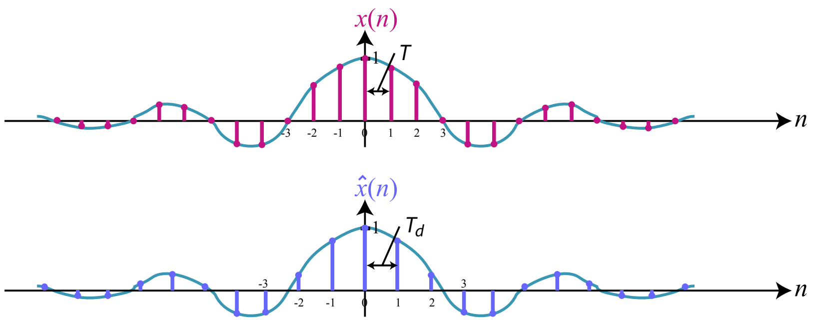
Sections 11.2, 11.3 and 11.4 of

John G. Proakis and Dimitris G. Manolakis, *Digital Signal Processing: Principles, Algorithms, and Applications*, 4th edition, 2007.

Sampling Rate Conversion

- **Goal:** Given a discrete-time signal $x(n)$ sampled at period T from an underlying continuous-time signal $x_a(t)$, determine a new sequence $\hat{x}(n)$ that is a sampled version of $x_a(t)$ at a different sampling rate T_d.

$$x(n) = x_a(nT) \qquad \hat{x}(n) = x_a(nT_d)$$



Sampling Rate Conversion for Audio Effects

Two fundamental questions for use in audio effects applications:

- What does sampling rate conversion do to the frequency spectrum of a signal?
- How is it best to implement sampling rate conversion?

Sampling Rate Conversion

- **One Interpretation:**
  1. Reconstruct the underlying continuous-time signal $x_a(t)$ from samples $x(n) = x_a(nT)$.
  2. Resample at the desired sampling rate: $\hat{x}(n) = x_a(nT_d)$.

- If

$$\frac{T}{T_d} = \text{rational number}$$

then sampling rate conversion becomes equivalent to sampling and/or interpolation of discrete-time signals.

Sampling and Interpolation of Discrete-Time Signals

Let $D, I \in \{1, 2, 3, 4, \ldots\}$

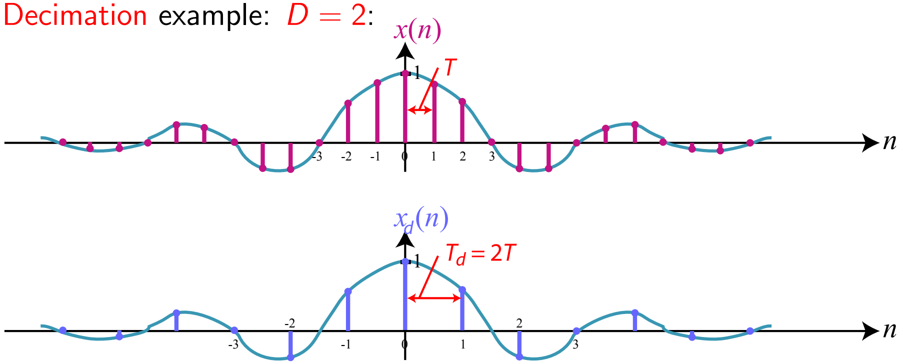
- For $T_d = DT$: called decimation or downsampling
- For $T_d = \frac{T}{I}$: called interpolation or upsampling

Sampling of Discrete-Time Signals

Suppose a discrete-time signal $x(n)$ is sampled by taking every Dth sample as follows:

$$x_d(n) = x(nD), \qquad \text{for all } n$$

Decimation example: $D = 2$:



Sampling of Discrete-Time Signals

Q: What happens to the signal spectrum during decimation?

Q: What is the relationship between $X(F)$ and $X_d(F)$?

Sampling of Discrete-Time Signals

Recall when we sample a continuous-time signal $x(t)$ to produce $x(n)$, we have the following relationships:

$$x(n) = x_a(nT) \overset{\mathcal{F}}{\longleftrightarrow} X(F) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_a\left(F - \frac{k}{T}\right)$$

$$\text{sampling} \overset{\mathcal{F}}{\longleftrightarrow} \text{periodic extension}$$

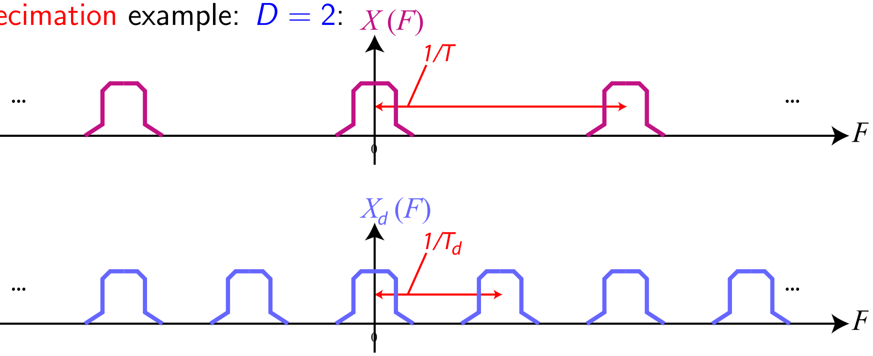
Suppose

$$x_d(n) = x(nD) = x_a(\underbrace{nD}\, T)$$
$$x(n) = x_a(nT)$$

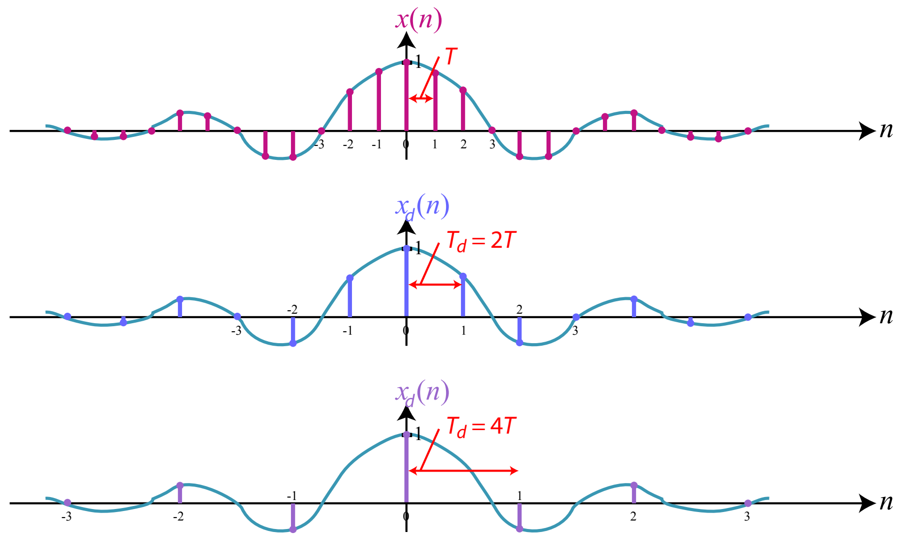
$$x(n) = x_a(nT)$$
$$X(F) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_a\left(F - \frac{k}{T}\right)$$
$$x_d(n) = x_a(nDT)$$
$$X_d(F) = \frac{1}{DT}\sum_{k=-\infty}^{\infty} X_a\left(F - \frac{k}{DT}\right)$$

$$X(F) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_a\left(F - \frac{k}{T}\right)$$
$$X_d(F) = \frac{1}{DT}\sum_{k=-\infty}^{\infty} X_a\left(F - \frac{k}{DT}\right)$$

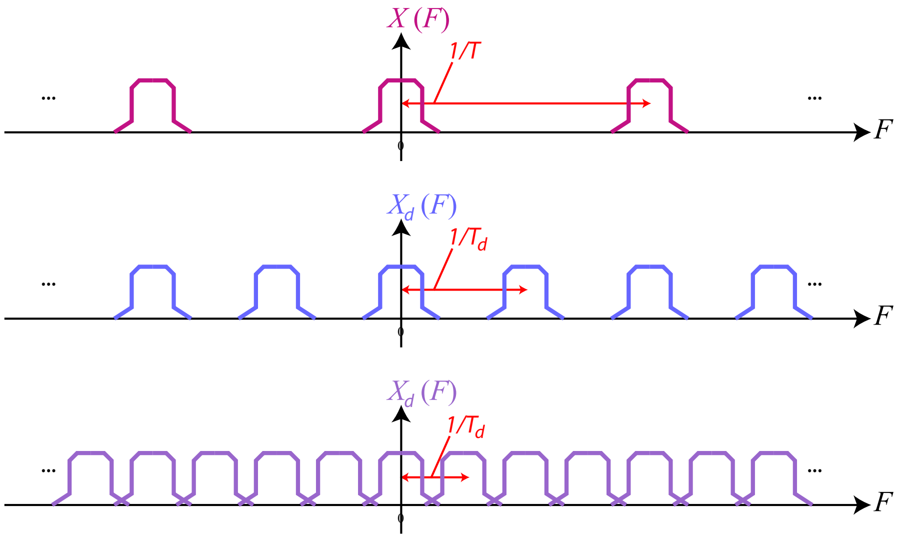
Decimation example: $D = 2$:



Decimation example: $D = 2, 4$:



Decimation example: $D = 2, 4$:



Sampling of Discrete-Time Signals

Therefore, from

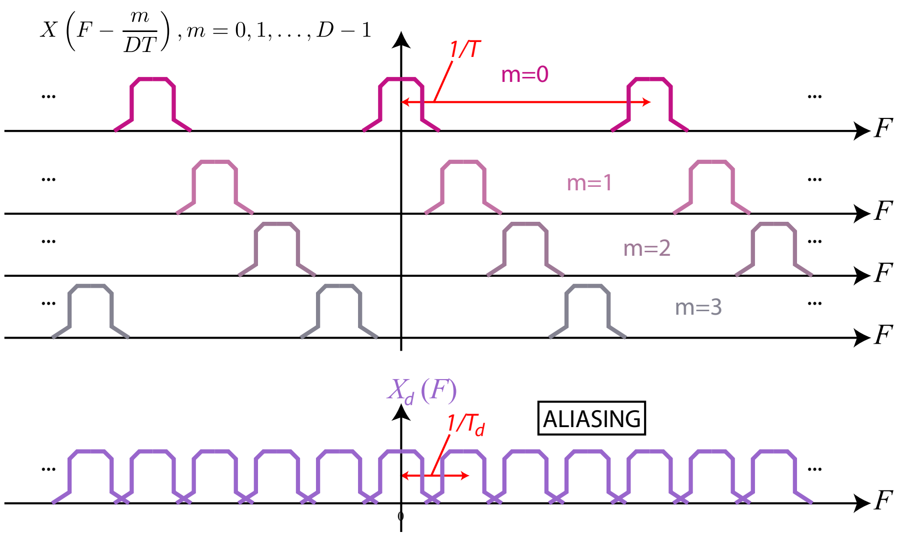
$$X(F) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X_a\left(F - \frac{k}{T}\right)$$

$$X_d(F) = \frac{1}{DT} \sum_{k=-\infty}^{\infty} X_a\left(F - \frac{k}{DT}\right)$$

By inspection, we have:

$$X_d(F) = \frac{1}{D} \sum_{m=0}^{D-1} X\left(F - \frac{m}{DT}\right)$$

Decimation example: $D = 4$:



Aliasing from Decimation

Thus,

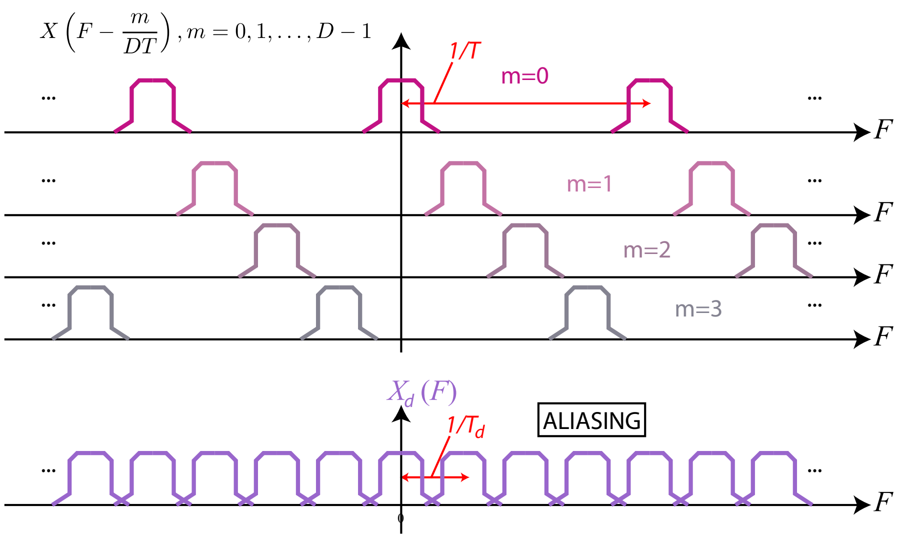
Cts-time Sampling	$\Longleftrightarrow$	$X_a(F)$ repeated infinite times
Dst-time Sampling	$\Longleftrightarrow$	$X(F)$ repeated finite times

To avoid aliasing when decimating via factor D:

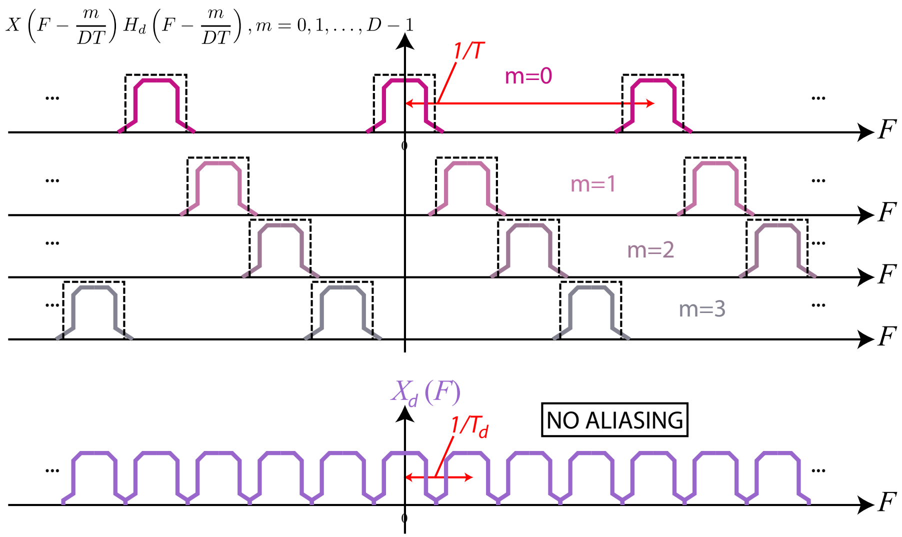
$$\text{Maximum Frequency} \leq \frac{1}{2DT}$$

Thus an anti-aliasing filter is applied prior to decimation.

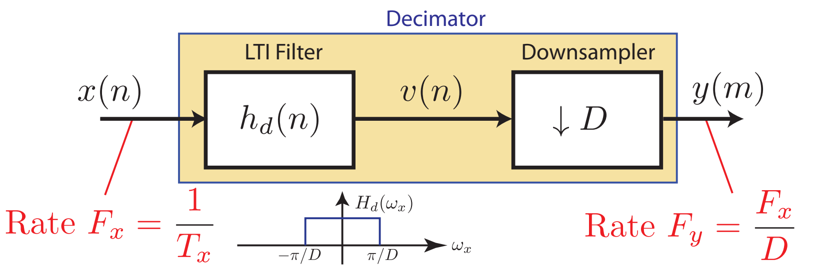
Decimation example: $D = 4$: no anti-aliasing filter



Decimation example: $D = 4$: anti-aliasing filter



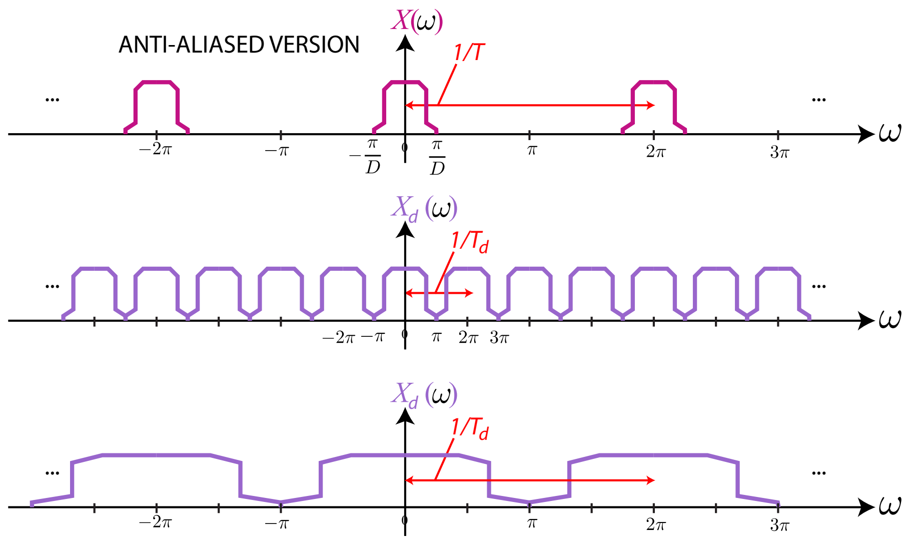
Downsampling with Anti-Alaising Filter



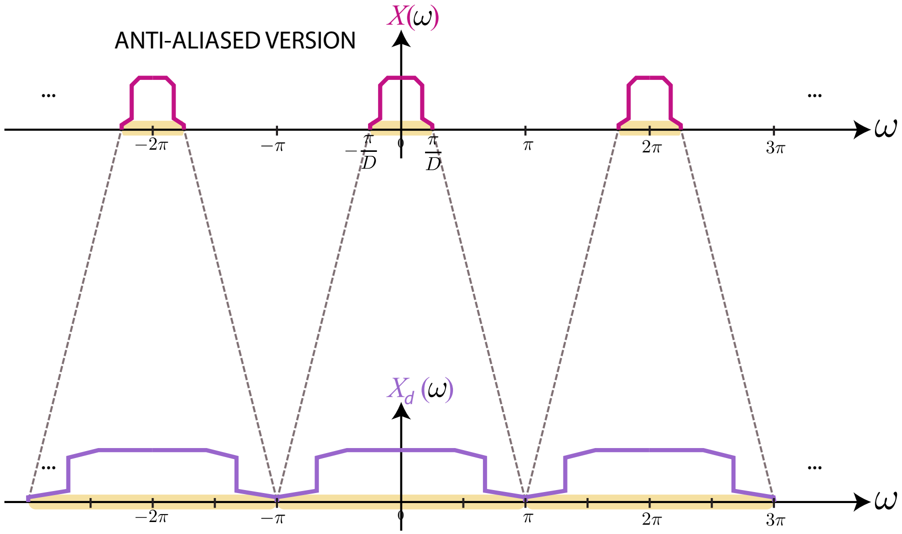
- The anti-aliasing filter $H_d(\omega)$ should have effective continuous-time frequency cutoff of $F_0 = \frac{1}{2DT}$ Hz, which is equivalent to a normalized cutoff of:

$$f_0 = \frac{F_0}{F_s} = \frac{1}{2DT} \cdot \frac{1}{F_s} = \frac{1}{2D} \quad \text{or} \quad \omega_0 = 2\pi \frac{1}{2D} = \frac{\pi}{D}$$

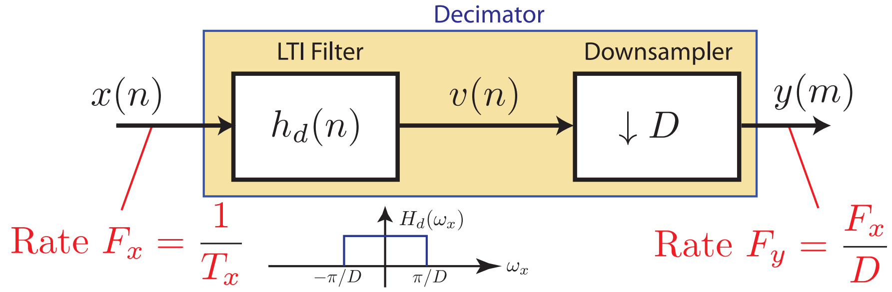
$-\pi/D \leq \omega \leq \pi/D$ is expanded into $-\pi \leq \omega \leq \pi$



$-\pi/D \le \omega \le \pi/D$ is expanded into $-\pi \le \omega \le \pi$

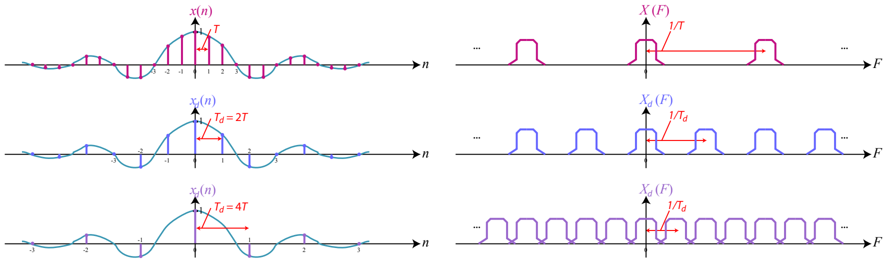


Interpolation by a Factor I



- Decimation keeps every Dth point giving a higher rate of change to the signal.
- The decimation process stretches an anti-aliased signal such that it contains higher frequency components.
- Thus, decimation generally speeds up an audio signal, making it appear to have higher tonal characteristics.

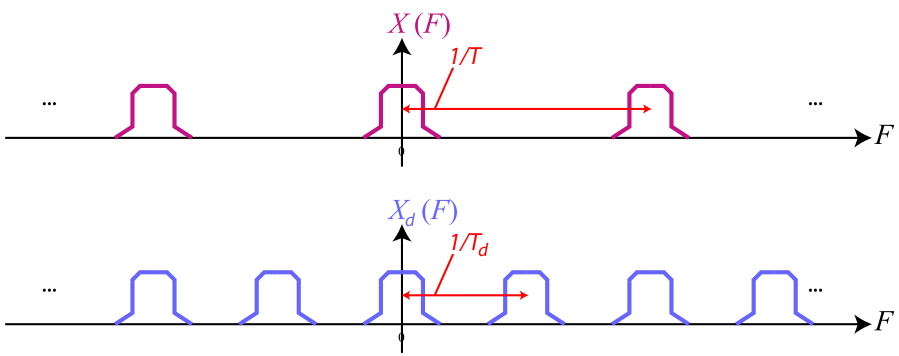
Interpolation of Discrete-time Signals



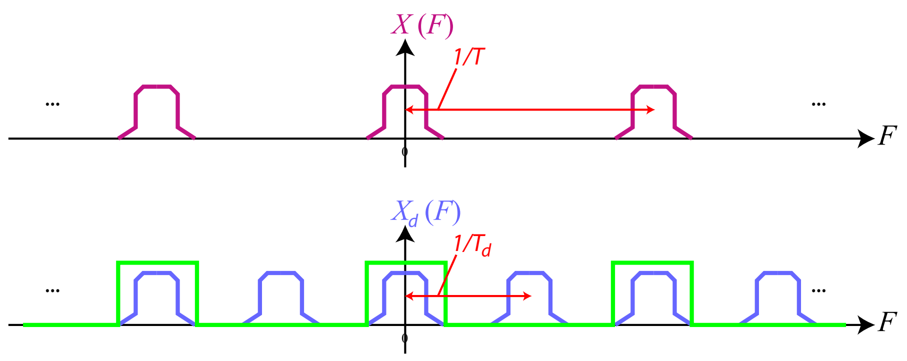
- Interpolation for $T_d = \frac{T}{D}$ is possible if no aliasing exists in the signal to be interpolated.

Note: We will later change D to I to distinguish between the decimation and interpolation factors. We use D here for simplicity as interpolation is being described, in part, as the *reverse process* of decimation.

Interpolation of Discrete-time Signals



Interpolation of Discrete-time Signals



Interpolation of Discrete-time Signals

Analysis Strategy:

- We consider the process of discrete-time interpolation; i.e., obtaining $x(n)$ from its decimated version $x_d(n) = x(nD)$.
- We will assume that no aliasing resulted from the decimation process.
- We will determine a relationship between $x(n)$ and $x_d(n)$ in the following way:
  1. Let us mathematically reconstruct $x_a(t)$ from $x_d(n)$ assuming a sampling period of DT.
  2. Let us then sample $x_a(t)$ with a sampling period of T to construct $x(n)$.

Interpolation of Discrete-time Signals

Step 1: $x_a(t)$ can be reconstructed from $x_d(n)$ as follows:

$$x_a(t) = \sum_{m=-\infty}^{\infty} x_d(m) \frac{\sin \frac{\pi}{DT}(t - mDT)}{\frac{\pi}{DT}(t - mDT)}$$

Step 2: Sample $x_a(t)$ to produce $x(n)$:

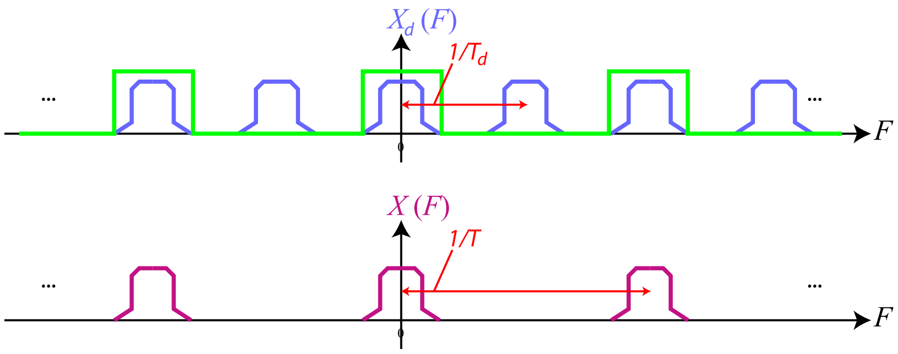
$$\begin{aligned} x(n) = x_a(nT) &= \sum_{m=-\infty}^{\infty} x_d(m) \frac{\sin \frac{\pi}{DT}(nT - mDT)}{\frac{\pi}{DT}(nT - mDT)} \\ &= \sum_{m=-\infty}^{\infty} x_d(m) \frac{\sin \frac{\pi}{D}(n - mD)}{\frac{\pi}{D}(n - mD)} \end{aligned}$$

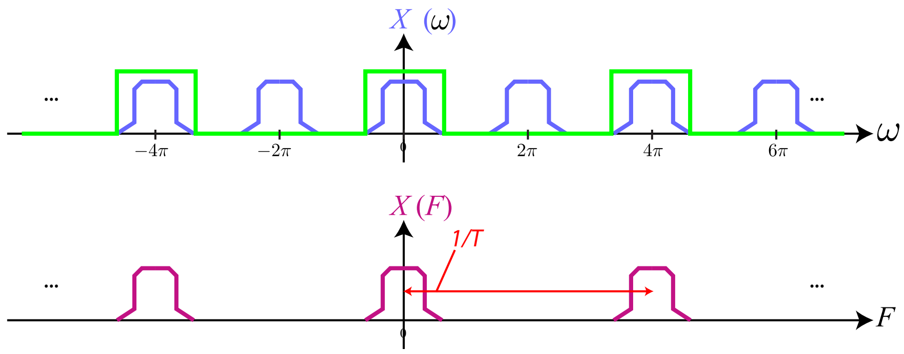
Interpolation of Discrete-time Signals

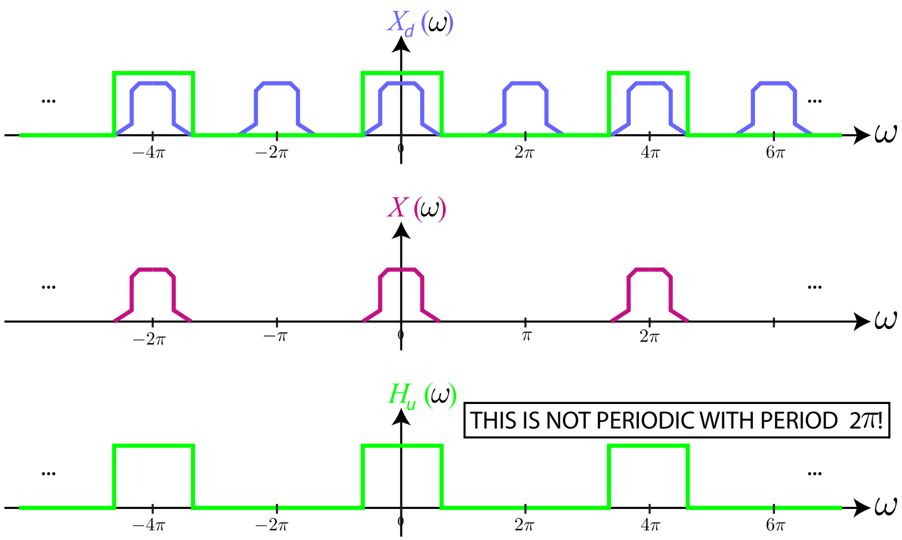
$$\begin{aligned} x(n) &= \sum_{m=-\infty}^{\infty} x_d(m) \left[\frac{\sin \frac{\pi}{D}(n - mD)}{\frac{\pi}{D}(n - mD)} \right] \\ &= \sum_{m=-\infty}^{\infty} x_d(m) g_{BL}(n - mD) \end{aligned}$$

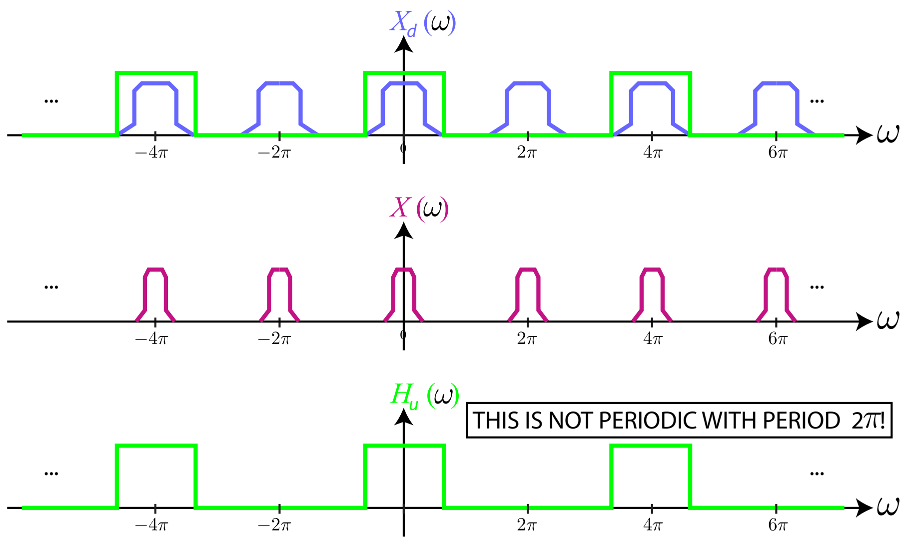
where

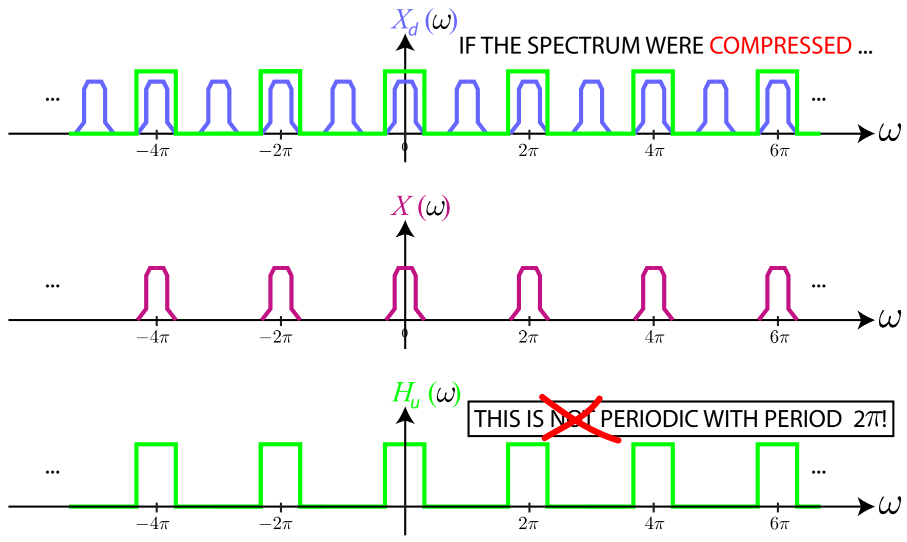
$$g_{BL}(n) = D \frac{\sin(\pi/D)n}{\pi n} \overset{\mathcal{F}}{\longleftrightarrow} G_{BL}(\omega) = \begin{cases} D & |\omega| \leq \frac{\pi}{D} \\ 0 & \frac{\pi}{D} < |\omega| \leq \pi \end{cases}$$


$X_d(F)$
$1/T_d$
...
...
F
$X(F)$
$1/T$
F


$X(\omega)$
-4π
-2π
2π
4π
6π
ω
$X(F)$
$1/T$
F


$X_d(\omega)$
-4π
-2π
2π
4π
6π
ω
$X(\omega)$
-2π
$-\pi$
π
2π
$H_u(\omega)$
THIS IS NOT PERIODIC WITH PERIOD 2π!


$X_d(\omega)$
-4π
-2π
2π
4π
6π
ω
$X(\omega)$
$H_u(\omega)$
THIS IS NOT PERIODIC WITH PERIOD 2π!



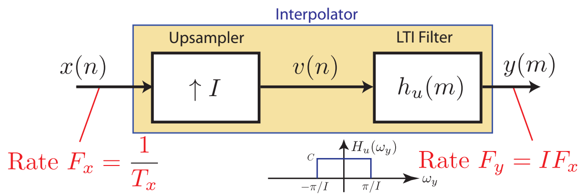
Interpolation of Discrete-time Signals

To achieve this, consider a two-stage process:

- Stage 1: Upsample to appropriately compress the spectrum.
- Stage 2: Then filter with an appropriate lowpass filter.

- We will consider upsampling by a factor of I.
  - Note: we change here the interpolation factor from D to I to distinguish our results from decimation.

Interpolation of Discrete-time Signals

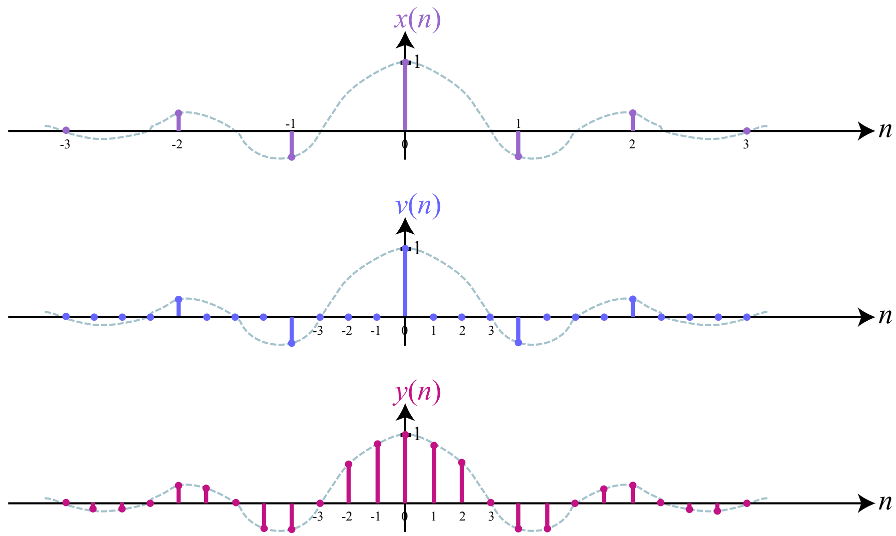


- Upsampling (without filtering) can be represented as:

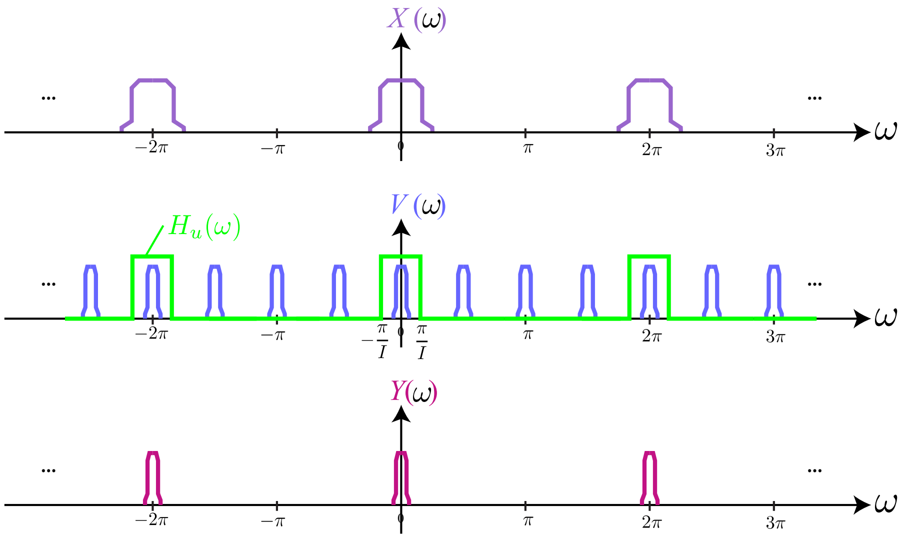
$$v(m) = \begin{cases} x(m/I) & m = 0, \pm I, \pm 2I, \ldots \\ 0 & \text{otherwise} \end{cases}$$

$$V(\omega) = X(\omega I)$$

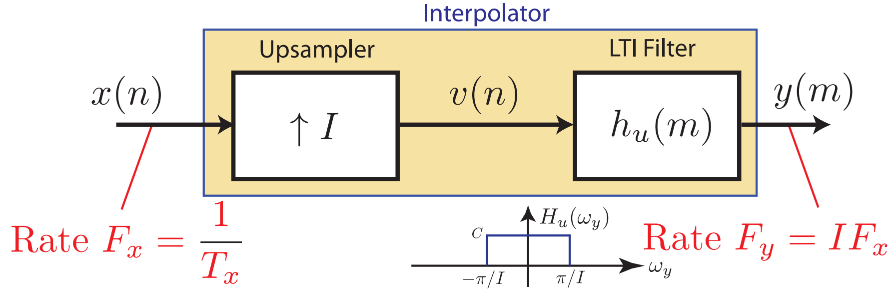
Interpolation example: $I = 4$: upsampling + lowpass filtering



Interpolation example: $I = 4$: upsampling + lowpass filtering



Interpolation by a Factor I



- Interpolation only increases the visible resolution of the signal. No new information is gained.
- Interpolation generally slows down an audio signal, making it appear to have lower tonal characteristics.

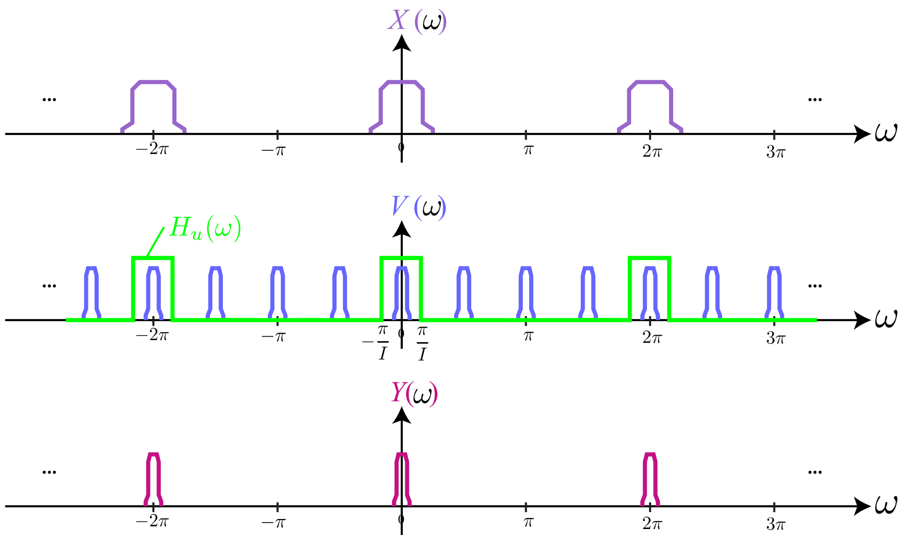
Overall,

$$V(\omega) = X(\omega I)$$
$$H_u(\omega) = \begin{cases} I & 0 \le |\omega| \le \pi/I \\ 0 & \text{otherwise} \end{cases}$$
$$Y(\omega) = H_u(\omega)V(\omega) = \begin{cases} IX(\omega I) & 0 \le |\omega| \le \pi/I \\ 0 & \text{otherwise} \end{cases}$$

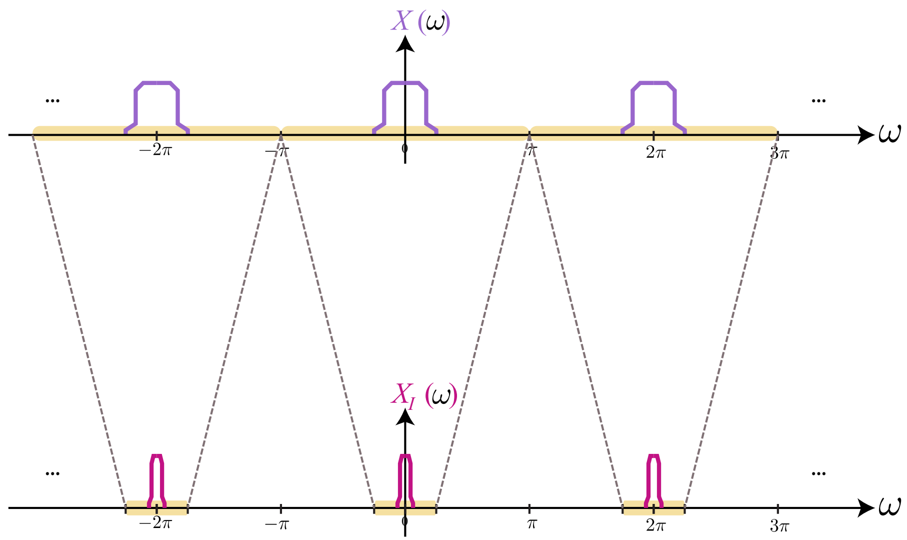
$$\boxed{Y(\omega) = \begin{cases} IX(\omega I) & 0 \le |\omega| \le \pi/I \\ 0 & \text{otherwise} \end{cases}}$$

$-\pi \le \omega \le \pi$ is compressed into $-\pi/I \le \omega \le \pi/I$

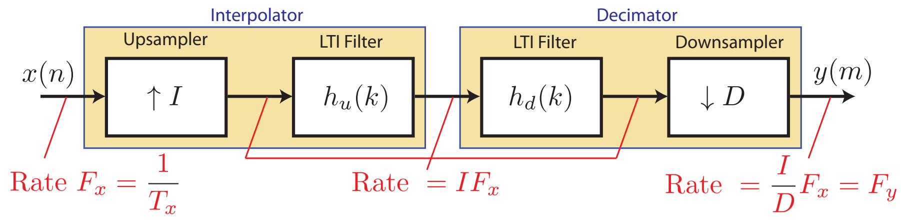
$-\pi \le \omega \le \pi$ is compressed into $-\pi/I \le \omega \le \pi/I$



$-\pi \leq \omega \leq \pi$ is compressed into $-\pi/I \leq \omega \leq \pi/I$

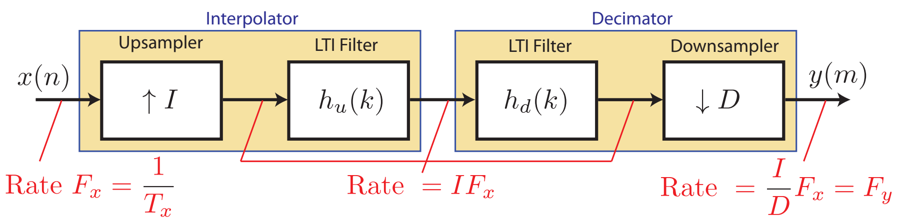


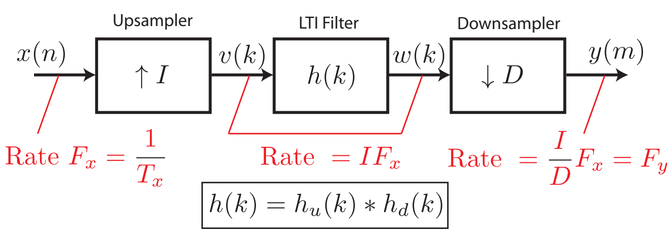
Sampling Rate Conversion by I/D

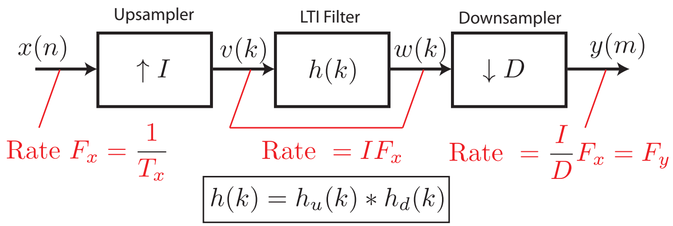


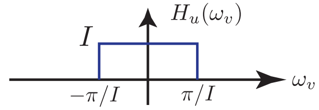
- $x(n)$: original samples at sampling rate F_x
- $y(n)$: new samples at sampling rate F_y

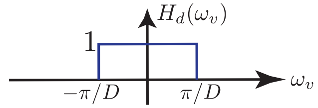
Sampling Rate Conversion by I/D

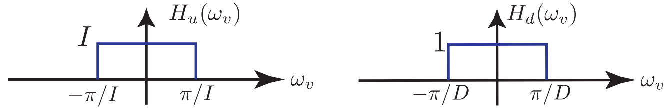






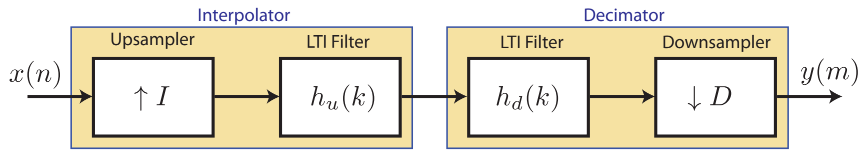


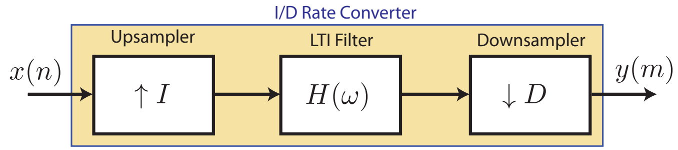


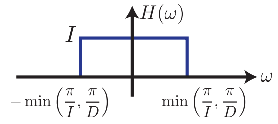


$$H(\omega) = H_u(\omega)H_d(\omega) = \begin{cases} I & 0 \le |\omega| \le \min(\pi/D, \pi/I) \\ 0 & \text{otherwise} \end{cases}$$

Sampling Rate Conversion by I/D







- Thus, a wide variety of sound speed conversions is possible through a combination of upsampling, LTI filtering and downsampling.

■