Changing the Sampling Rate in DSP

Jose Krause Perin

Stanford University

July 11, 2017

Last lecture

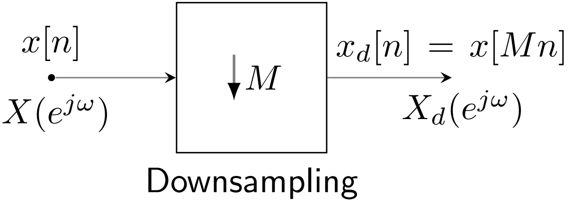
- Sampling a continuous-time signal results in replicas of the spectrum at multiples of the sampling frequency Ω_s (or 2π of the normalized frequency ω)
- A band-limited signal has highest frequency Ω_N $(X_c(j\Omega) = 0, |\Omega| > \Omega_N)$
- If a band-limited signal is oversampled $(\Omega_s > 2\Omega_N)$ there'll be gaps between the spectrum replicas
- If the signal is undersampled $(\Omega_s < 2\Omega_N)$ the spectrum replicas will overlap resulting in aliasing distortion
- We can perfectly reconstruct a signal from its samples, provided that there is no aliasing and that we use the ideal lowpass filter as reconstruction filter
- In practice, we use different reconstruction filters, since the ideal lowpass filter is unfeasible.
- Oversampling relaxes the reconstruction filter specifications
- In theory, we can perform any LTI continuous-time filtering in discrete-time (in DSP), provided that there is no aliasing and that we use the ideal reconstruction filter

Today's lecture

- Downsampling and decimation
- Upsampling and interpolation
- Noninteger rate change
- Multi-rate processing

Downsampling

Downsampling by an integer factor M is equivalent to sampling the discrete-time signal $x[n]$ with sampling period M.



To understand what happens in the frequency domain, we can think that we're sampling the original continuous-time signal $x_c(t)$ with sampling period $T_d = MT$:

$$x_d[n] = x[Mn] = x_c(nMT)$$

Downsampling: frequency domain interpretation

Sampling $x_c(t)$ with sampling period $T_d = MT$ results in

$$X_d(e^{j\omega}) = \frac{1}{T_d}\sum_{k=-\infty}^{\infty} X_c\left[j\left(\frac{\omega}{T_d} - \frac{2\pi k}{T_d}\right)\right] \qquad \text{(spectrum replicas appear with period } \Omega_s = 2\pi/T_d\text{)}$$

$$= \frac{1}{MT}\sum_{k=-\infty}^{\infty} X_c\left[j\left(\frac{\omega}{MT} - \frac{2\pi k}{MT}\right)\right] \qquad (T_d = MT)$$

$$= \frac{1}{MT}\sum_{m=0}^{M-1}\sum_{l=-\infty}^{\infty} X_c\left(j\left(\frac{\omega}{MT} - \frac{2\pi l}{T} - \frac{2\pi m}{MT}\right)\right) \qquad \text{(change of variables: } k = m + lM\text{)}$$

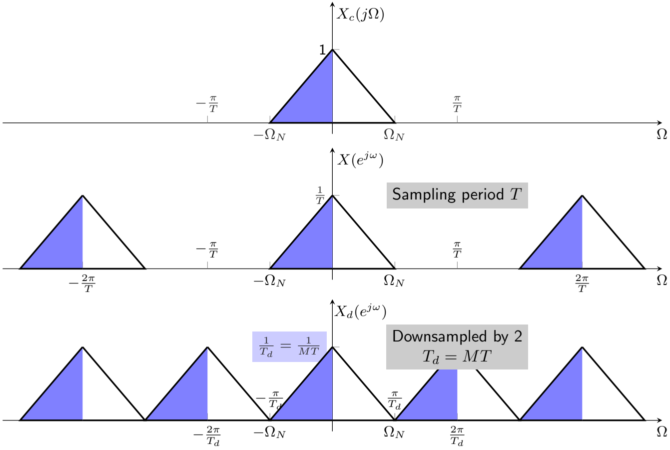
$$= \frac{1}{M}\sum_{m=0}^{M-1} \boxed{\frac{1}{T}\sum_{l=-\infty}^{\infty} X_c\left(j\left(\frac{\omega - 2\pi m}{MT} - \frac{2\pi l}{T}\right)\right)} \qquad \text{(rearranging)}$$

$$= \frac{1}{M}\sum_{m=0}^{M-1} X(e^{j\omega'})\Big|_{\omega' = \frac{\omega - 2\pi m}{M}} \qquad (\square \text{ is equivalent to } X(e^{j\omega'}) \text{ for } \omega' = \tfrac{\omega - 2\pi m}{M})$$

Conclusion: $X(e^{j\omega})$ is stretched by a factor of M (ω/M), and there will be replicas of the spectrum with period $2\pi/M$.

Downsampling: frequency domain interpretation

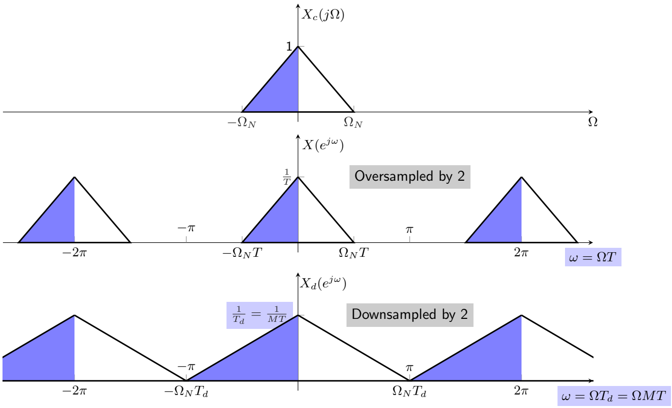
Example of downsampling with $M = 2$ i.e., $T_d = 2T$.
Impulse sampling interpretation:



After downsampling spectrum replicas appear with period $2\pi/T_d$

Downsampling: frequency domain interpretation

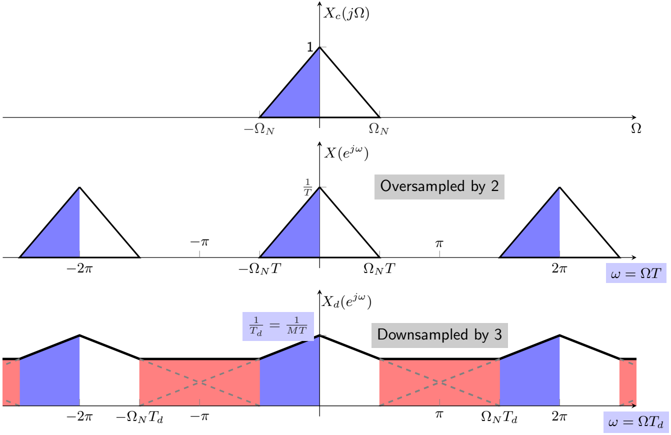
Same example of downsampling with $M = 2$ i.e., $T_d = 2T$.
Discrete-time interpretation:



To obtain the spectrum we discrete time, we just need to use the change of variables $\omega = \Omega T$ and $\omega = \Omega T_d$

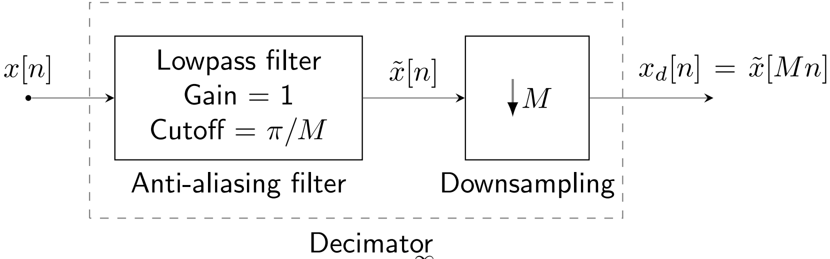
Downsampling: frequency domain interpretation

Downsampling may lead to spectrum overlapping (**aliasing distortion**).
Example of downsampling with $M = 3$ i.e., $T_d = 3T$.



Decimation

Similarly to sampling, it is common to employ an **anti-aliasing filter** before downsampling in order to minimize aliasing.
Pre-filtering followed by downsampling is called **decimation**.



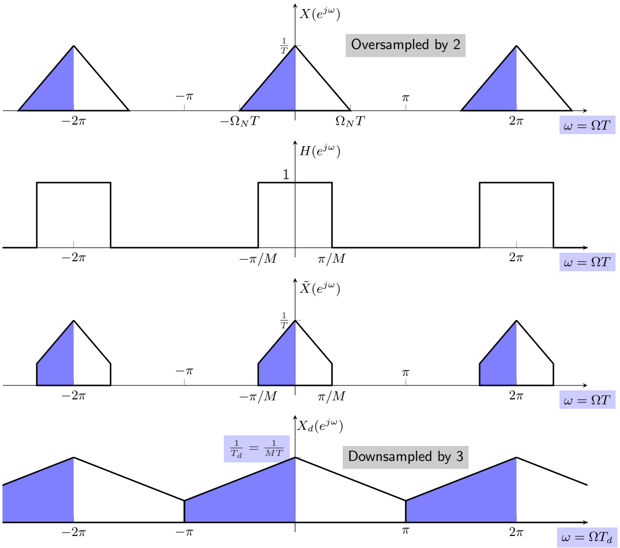
$$x[n] \Longleftrightarrow X(e^{j\Omega T}) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_c\left(j\left(\Omega - \frac{2\pi k}{T}\right)\right) \qquad \text{(from sampling)}$$

$$\tilde{x}[n] \Longleftrightarrow \tilde{X}(e^{j\Omega T}) = H(e^{j\Omega T})X(e^{j\Omega T}) \qquad \text{(LTI output)}$$

$$x_d[n] = \tilde{x}[nM] \Longleftrightarrow \frac{1}{M}\sum_{m=0}^{M-1} \tilde{X}(e^{j(\Omega T - 2\pi T)/M}) \qquad \text{(downsampling by } M\text{)}$$

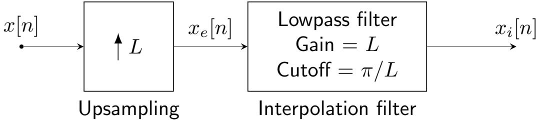
Decimation: frequency domain interpretation

Decimation with $M = 3$ i.e., $T_d = 3T$.



Interpolation

Interpolation is used to increase the sampling rate by an integer factor L.



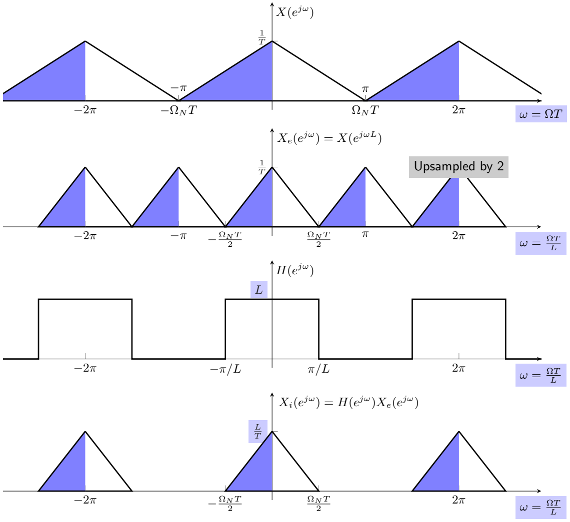
$$x_e[n] = \begin{cases} x[n/L], & 0, \pm L, \pm 2L, \ldots \\ 0, & \text{otherwise} \end{cases} \qquad \text{(after upsampling)}$$

In the frequency domain:

$$\begin{aligned} X_e(e^{j\omega}) &= \sum_{n=-\infty}^{\infty} x_e[n] e^{-j\omega n} && \text{(definition of DTFT)} \\ &= \sum_{n=0,\pm L,\ldots} x[n/L] e^{-j\omega n} && \text{(from } x_e[n] \text{ equation above)} \\ &= \sum_{k=-\infty}^{\infty} x[k] e^{-j\omega k L} && \text{(change of variable: } k = n/L) \\ &= X(e^{j\omega L}) && \text{(from DTFT equation with } \omega L) \end{aligned}$$

Interpolation: frequency domain interpretation

Example of interpolation with $L = 2$

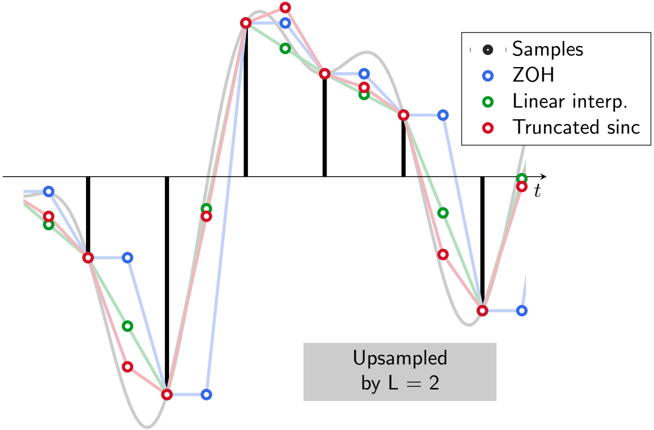


Practical interpolation filters

- Similarly to what we saw in reconstruction (D-to-C), the ideal lowpass filter is not practical. Hence, we must use practical interpolation filters such as ZOH, linear interpolator, or cubic spline.
- **One important difference:** The interpolation filter used in reconstruction to convert from discrete-time to continuous-time was an analog filter (a continuous-time filter). The interpolation filter used for upsampling is realized in discrete-time (in DSP). Therefore, we have more flexibility.

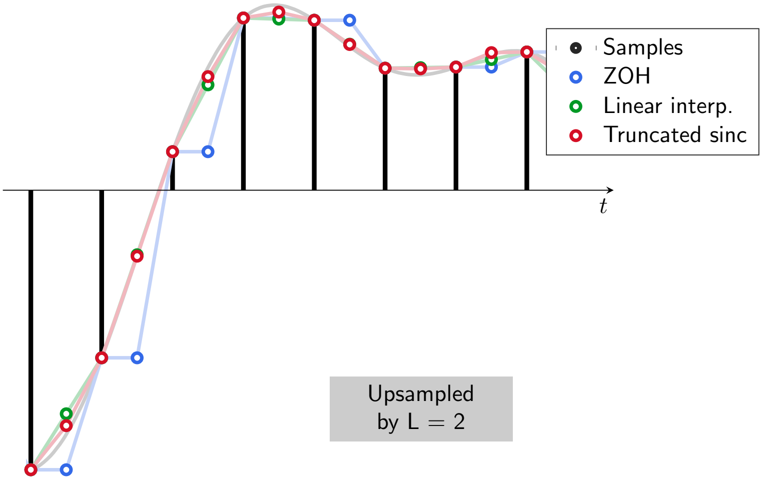
Example of interpolation

- The original signal has maximum frequency $\Omega_N = 1$ rad/s.
- From the Nyquist-Shannon theorem, we need $\Omega_s > 2\Omega_N$ in order to be able to achieve perfect reconstruction. Or equivalently, $T < \pi$ s.
- Sampling period is $T = \pi$



Example of interpolation

Same example as before, but now sampling period is $T = 0.4\pi$.



Truncated sinc filter had 11 coefficients, while ZOH had 2 and linear interpolator had 3.

- In the first example the continuous-time signal was sampled at the Nyquist rate, whereas in the second example the continuous-time signal was oversampled by 2.5.
- In both cases, we upsample by a factor of 2 and use practical reconstruction filters: ZOH, a linear interpolator, and a truncated sinc with 11 samples.
- The ZOH filter only has two coefficients, the linear interpolator has three coefficents, and the truncated sinc has 11 coefficients.
- Although we use the same filters in both examples, the interpolated sequences are much closer to the original continuous-time signal in the second example. This illustrates how oversampling can help the interpolation filters.
- In these examples, the linear interpolator offers that best performance vs complexity trade-off, as it achieves performance close to the truncated sinc, but only uses 3 samples.
- Even with high oversampling, we see that the truncated sinc filter didn't achieve perfect reconstruction. This is a consequence of the **Gibbs phenomenon**, discussed in lecture 1. Recall that a truncated sinc will produce a DTFT that is different from the ideal lowpass filter. Specifically, the DTFT of the truncated sequence will have oscillations, which will affect the signal and will not suppress the spectrum replicas centered at multiples of 2π. Using an even larger sequence (sinc with more samples) would not help much, since the ripples would only become more rapid, but their amplitude would not decrease.

In Matlab

Sampled signal with period T

```
>> T = 0.5*pi
>> t = -20:T:20
>> x = cos(t/2) - sin(t) + cos(t/2-pi/4) - sin(t/4-deg2rad(154)); % Sampled
signal
```

Upsample

```
>> xu = upsample(x, L) % Upsample
```

Interpolation filters

```
>> hZOH = [1 1] % ZOH
>> hlin = [1/2 1 1/2] % Linear interpolator
>> hsinc = sinc(-(5:5)/2) % truncated sinc with 11 samples
```

Interpolate

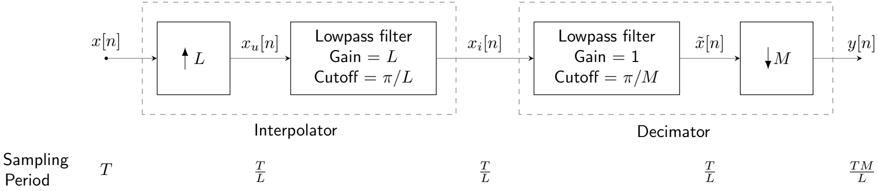
```
>> yzoh = filter(hZOH, 1, xu)
>> ylin = filter(hlin, 1, xu)
>> ysinc = filter(hsinc, 1, xu)
% since these filters are FIR we could also have used the conv command
```

Before plotting we need to remove the group delay introduced by the filters (more on this next week)

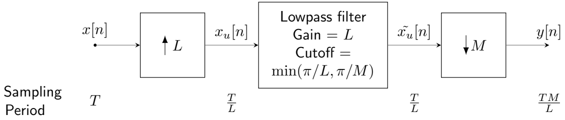
Interpolation/Decimation by a non-integer factor

- We have seen how to increase the sampling period by an integer factor M and how to decrease the sampling period by an integer factor L
- By cascading interpolation and decimation we can change the sampling period by a non-integer factor M/L.

Cascading interpolation and decimation

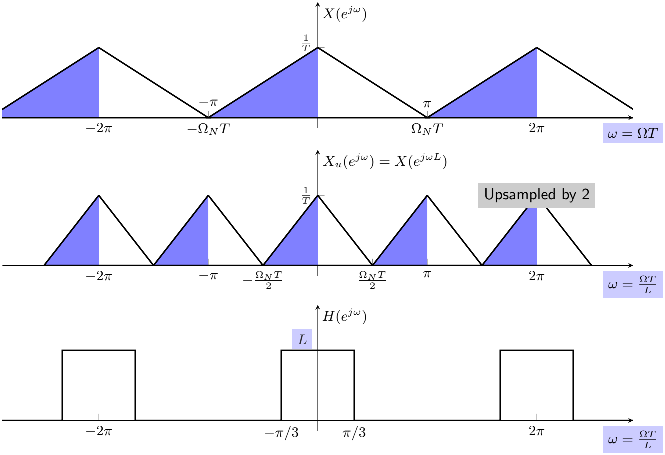


Equivalent diagram



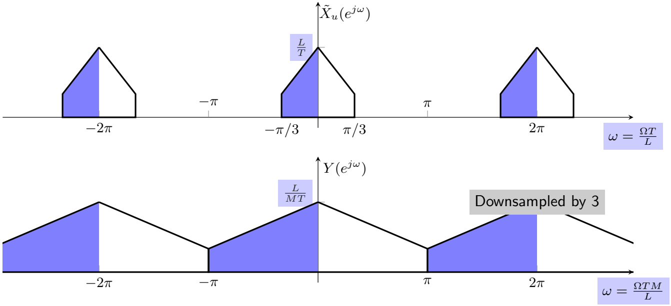
Example

- Let's combine the examples we saw earlier with $L = 2$ and $M = 3$. Recall that with $M = 3$, there would be aliasing if we didn't use the anti-aliasing filter.
- The filter cutoff is $\min(\pi/2, \pi/3) = \pi/3$.



Example

continuing...



The resulting signal has sampling period $MT/L = 3T/2$. Note that aliasing was prevented by selecting the cutoff frequency $\min(\pi/2, \pi/3) = \pi/3$.

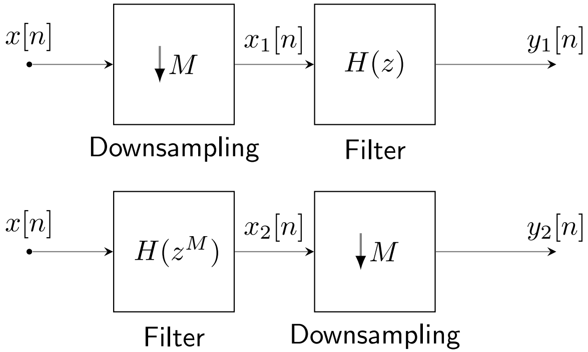
Multirate processing

In practice, it is common to have parts of the system operating at one sampling rate and other parts operating at a different sampling rate.

- Interchanging filtering and downsampling
- Interchanging filtering and upsampling
- Multi-stage decimation
- Multi-stage interpolation
- Polyphase decomposition

Interchanging filtering and downsampling

These two systems are equivalent i.e., $y_1[n] = y_2[n]$



To move the filter before downsampling by M, we must stretch its frequency response by a factor of M: $H(z^M) \xrightarrow{z=e^{j\omega}} H(e^{j\omega M})$.

Proof:
Staring with the second system: $X_2(e^{j\omega}) = H(e^{j\omega M})X(e^{j\omega})$. Now we can apply the equation for downsampling to obtain $Y_2(e^{j\omega})$

$$
\begin{aligned}
Y_2(e^{j\omega}) &= \frac{1}{M}\sum_{m=0}^{M-1} X_2(e^{j(\omega/M - 2\pi m)}) \\
&= \frac{1}{M}\sum_{m=0}^{M-1} X(e^{j(\omega/M - 2\pi m)})H(e^{j(\omega - 2\pi m)}) \\
&= H(e^{j\omega})\frac{1}{M}\sum_{m=0}^{M-1} X(e^{j(\omega/M - 2\pi m)}) \\
&\qquad \text{(periodicity of the DTFT} \implies H(e^{j(\omega-2\pi m)}) = H(e^{j\omega})) \\
&= H(e^{j\omega})\boxed{\frac{1}{M}\sum_{m=0}^{M-1} X(e^{j(\omega/M - 2\pi m)})} \\
&= H(e^{j\omega})\boxed{X_1(e^{j\omega})} = Y_1(e^{j\omega})
\end{aligned}
$$

Comments on stability:

Note that if $H(z)$ is a rational z-transform with poles $\{p_1, p_2, \ldots, p_N\}$ and zeros $\{z_1, z_2, \ldots, z_R\}$:

$$H(z) = \frac{b_0}{a_0} z^{N-R} \frac{(z - z_1)(z - z_2)\ldots(z - z_R)}{(z - p_1)(z - p_2)\ldots(z - p_N)}$$

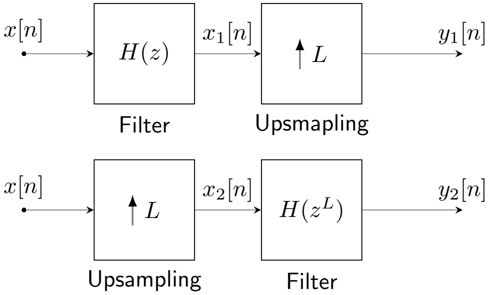
Then $H(z^M)$ will be

$$H(z^M) = \frac{b_0}{a_0} z^{M(N-R)} \frac{(z^M - z_1)(z^M - z_2)\ldots(z^M - z_R)}{(z^M - p_1)(z^M - p_2)\ldots(z^M - p_N)},$$

with poles $\{ \sqrt[M]{p_1}, \sqrt[M]{p_N}, \ldots, \sqrt[M]{p_N}\}$, and zeros $\{ \sqrt[M]{z_1}, \sqrt[M]{z_N}, \ldots, \sqrt[M]{z_R}\}$.
If $H(z)$ is stable and **causal**, the poles of $H(z^M)$ will lie inside the unit circle, and therefore $H(z^M)$ will also be stable.
Similarly, if $H(z)$ is **anti-causal** and stable, the poles of $H(z^M)$ will lie outside the unit circle, and therefore $H(z^M)$ will also be stable.

Interchanging filtering and interpolation

These two systems are equivalent i.e., $y_1[n] = y_2[n]$



Proof:
Top diagram:

$$Y_1(e^{j\omega}) = X_1(e^{j\omega L}) = X(e^{j\omega L})H(e^{j\omega L})$$

Bottom diagram:

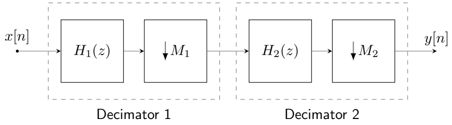
$$Y_2(e^{j\omega}) = X_2(e^{j\omega})H(e^{j\omega L}) = X(e^{j\omega L})H(e^{j\omega L})$$

Multi-stage decimation

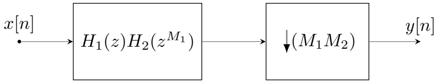
Suppose we want to decimate by a factor $M = 20$. The cutoff frequency of the lowpass filter would be $\pi/20$.

Sharp filter $\implies$ long impulse response $\implies$ higher complexity, higher cost, higher power consumption

It's more efficient to use several decimation stages



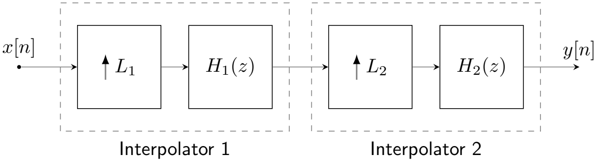
Interchanging filter and downsampling results in the equivalent system:



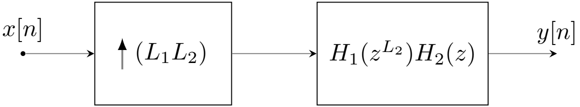
- The equivalent downsampling factor is $M = M_1 M_2$.
- Design $H_1(z)$ and $H_2(z)$ so that $H_1(z)H_2(z^{M_1})$ has the desired frequency response.

Multi-stage interpolation

The same rationale applies to interpolation



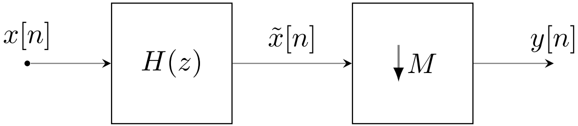
Interchanging filter and downsampling results in the equivalent system:



- The equivalent upsampling factor is $L = L_1 L_2$.
- Design $H_1(z)$ and $H_2(z)$ so that $H_1(z^{L_2})H_2(z)$ has the desired frequency response.

Polyphase decomposition

What if the filter is placed before downsampling?



- To interchange filter and downsampling in this case, we'd need to express $H(z)$ as some $G(z^M)$. Generally not easy.
- **Practical problem:** this implementation wastes computation. All samples of the output of $H(z)$ are calculated, but only 1 out of M is used after downsampling.
- If $H(z)$ is FIR of length N, there are N multiplications per sample. Downsampling by M discards $M - 1$ samples every M samples.

Polyphase decomposition

We can decompose any given sequence $h[n]$ into M subsequences such that

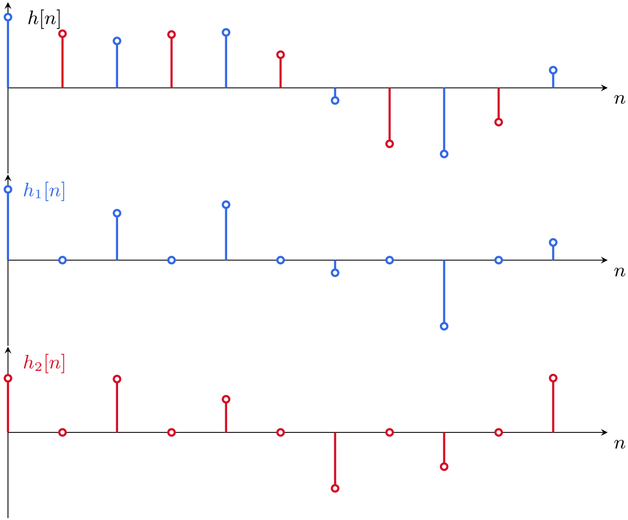
$$h_k[n] = \begin{cases} h[n+k], & n \text{ integer multiple of } M \\ 0, & \text{otherwise} \end{cases}, k = 0, 1, \ldots, M-1$$

It follows that

$$h[n] = \sum_{k=0}^{M-1} h_k[n-k] \Longleftrightarrow H(z) = \sum_{k=0}^{M-1} H_k(z) z^{-k}$$

Polyphase decomposition

Example of decomposition with $M = 2$

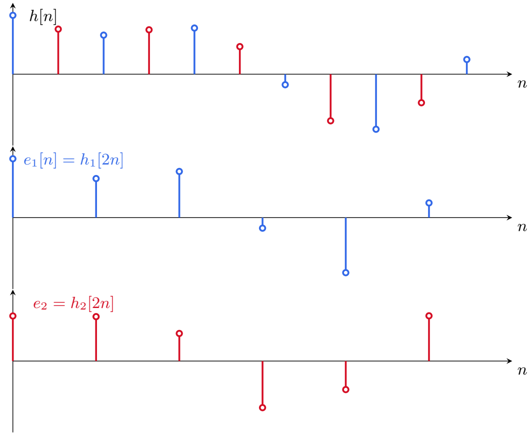


Polyphase decomposition

We can downsample $h_k[n]$ in order to discard the zero samples

$$e_k[n] = h_k[Mn] \Longleftrightarrow E_k(z^M) = H_k(z) \qquad \text{(upsamling by M)}$$

The subsequences $e_k[n]$ are called the **polyphase components** of $h[n]$



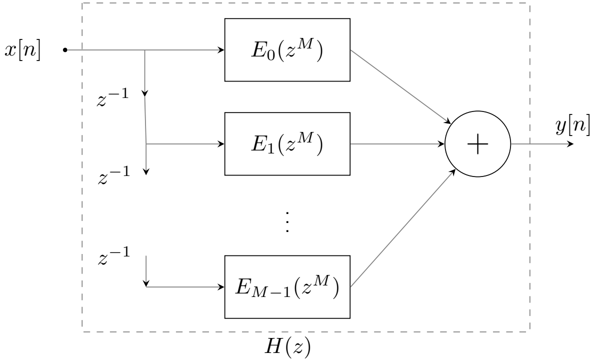
Polyphase decomposition

How to recover $h[n]$ from $e_0[n], \ldots, e_{M-1}[n]$?

1. Upsample $e_k[n]$ by M, and we're back with $h_k[n]$
2. Delay by k and add

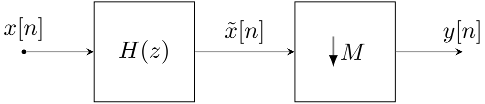
In terms of the z-transform:

$$H(z) = \sum_{k=0}^{M-1} E_k(z^M) z^{-k}$$

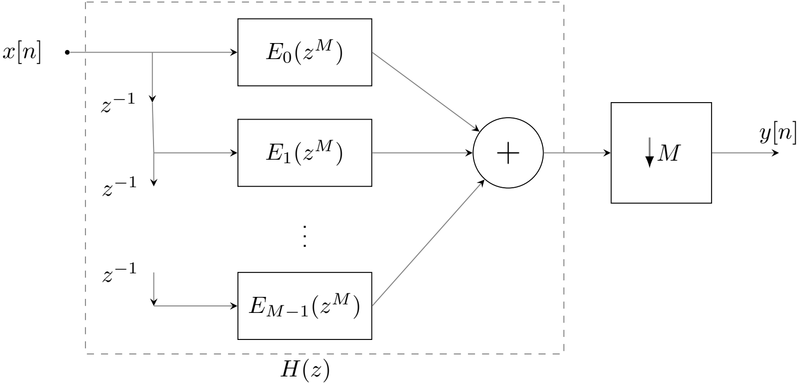


Polyphase decimation

Back to the original problem: how to interchange filter and downsampling?

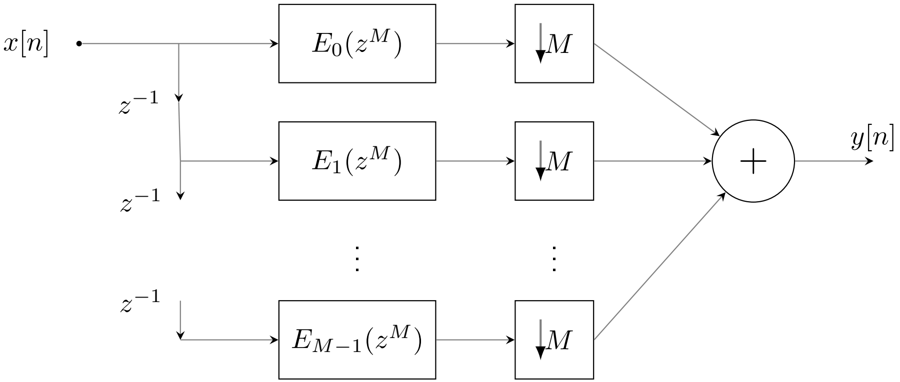


Using the polyphase decomposition of $H(z)$



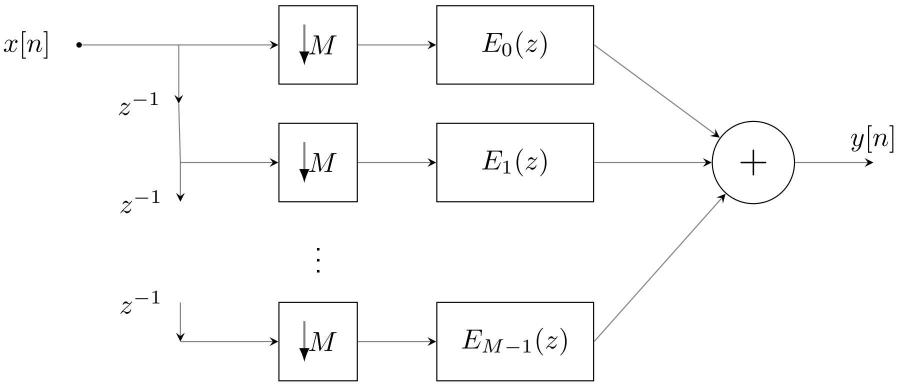
Polyphase decimation

First interchange sum and downsampling:



Polyphase decimation

Now it is easy to interchange $E_k(z^M)$ with downsampling resulting in the filters $E_k(z)$



Computation: Each polyphase filter $E_k(z)$ requires N/M multiplications, which are realized at the lower rate (higher sampling period) TM.
Similarly for polyphase interpolation (Textbook section 4.7.5)

Summary

- Downsampling by an integer factor M stretches the discrete-time spectrum by a factor M and causes replicas of the spectrum to appear at $2\pi/M$. The amplitude of the spectrum is attenuated by M
- It's often easier to think of downsampling as sampling the original continuous-time signal with a sampling period $T_d = MT$
- Anti-aliasing filtering followed by downsampling is called decimation
- Upsampling by an integer factor L compresses the discrete-time spectrum by a factor L. The interpolation filter is assumed to have gain L, so the spectrum amplitude is scaled by L
- We can achieve non-integer sampling rate changes by cascading interpolation and decimation stages
- For large downsampling/uspsampling factors, it's generally more efficient to realized multistage decimation/interpolation
- Polyphase decomposition allows efficient implementation of filtering followed by downsampling and upsampling followed by filtering.