ECE-700 Multirate Notes

Phil Schniter

March 27, 2006

1 Fundamentals of Multirate Signal Processing

- *Upsampling:* The operation of "upsampling" by factor $L \in \mathbb{N}$ describes the insertion of $L-1$ zeros between every sample of the input signal. This is denoted by "$\uparrow L$" in block diagrams, as below.

$$\begin{tikzcd} x[m] \arrow[r] & \boxed{\uparrow L} \arrow[r] & y[n] \end{tikzcd}$$

Formally, upsampling can be expressed in the time domain as

$$y[n] \;=\; \begin{cases} x\left[\frac{n}{L}\right] & \text{when } \frac{n}{L} \in \mathbb{Z} \\ 0 & \text{else.} \end{cases}$$

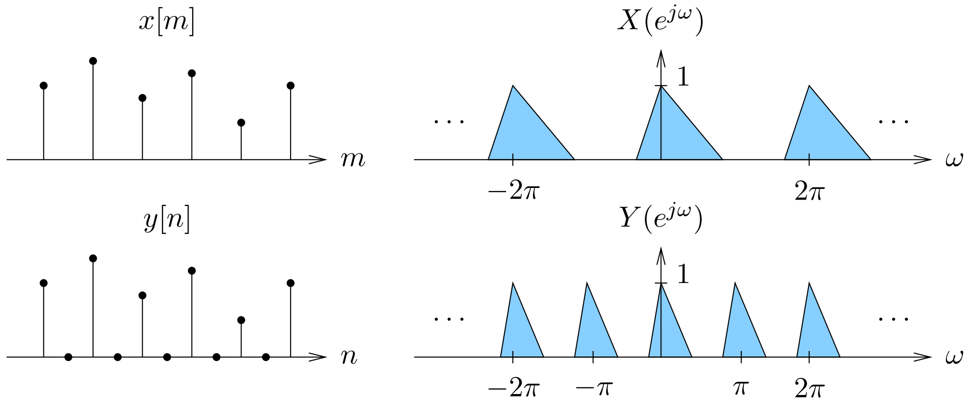
In the z-domain,

$$Y(z) \;=\; \sum_n y[n]z^{-n} \;=\; \sum_{n:\frac{n}{L}\in\mathbb{Z}} x\left[\tfrac{n}{L}\right]z^{-n} \;=\; \sum_k x[k]z^{-kL} \;=\; X\big(z^L\big),$$

and substituting $z = e^{j\omega}$ for the DTFT,

$$Y(e^{j\omega}) \;=\; X(e^{j\omega L}). \tag{1}$$

As shown below, upsampling compresses the DTFT by a factor of L along the ω axis.



- *Downsampling:* The operation of "downsampling" by factor $M \in \mathbb{N}$ describes the process of keeping every M^{th} sample and discarding the rest. This is denoted by "$\downarrow M$" in block diagrams, as below.

$$\begin{tikzcd} x[m] \arrow[r] & \boxed{\downarrow M} \arrow[r] & y[n] \end{tikzcd}$$

Formally, downsampling can be written as

$$y[n] \;=\; x[nM].$$

In the z domain,

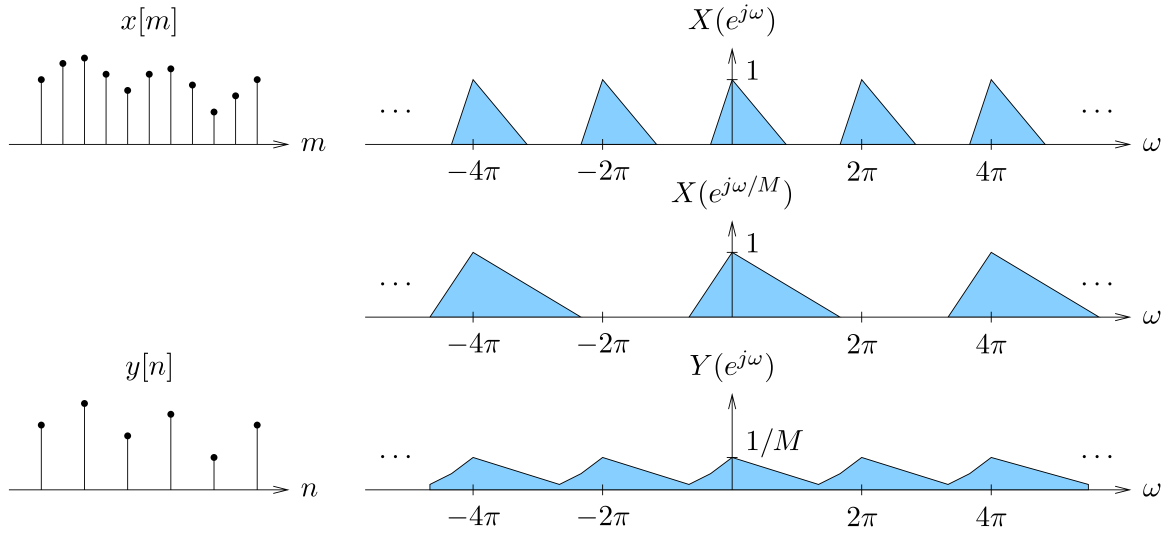
$$\begin{aligned}
Y(z) &= \sum_n y[n] z^{-n} \\
&= \sum_n x[nM] z^{-n} \\
&= \sum_m x[m] \underbrace{\left(\frac{1}{M} \sum_{p=0}^{M-1} e^{j\frac{2\pi}{M}pm} \right)}_{= \begin{cases} 1 & \text{when } m \text{ is a multiple of } M \\ 0 & \text{else} \end{cases}} z^{-m/M} \\
&= \frac{1}{M} \sum_{p=0}^{M-1} \sum_m x[m] \big(e^{-j\frac{2\pi}{M}p} z^{1/M} \big)^{-m} \\
&= \frac{1}{M} \sum_{p=0}^{M-1} X\big(e^{-j\frac{2\pi}{M}p} z^{1/M} \big).
\end{aligned}$$

Translating to the frequency domain,

$$Y(e^{j\omega}) \;=\; \frac{1}{M} \sum_{p=0}^{M-1} X\left(e^{j\frac{\omega - 2\pi p}{M}} \right). \tag{2}$$

As shown below, downsampling expands each 2π-periodic repetition of $X(e^{j\omega})$ by a factor of M along the ω axis, and reduces the gain by a factor of M. If $x[m]$ is not bandlimited to π/M, aliasing may result from spectral overlap.

Note: When performing a frequency-domain analysis of systems with up/downsamplers, it is strongly recommended to carry out the analysis in the z-domain until the last step, as done above. Working directly in the $e^{j\omega}$-domain can easily lead to errors.



- <u>*Interpolation:*</u> Interpolation is the process of upsampling and filtering a signal to increase its effective sampling rate. To be more specific, say that $x[m]$ is an (unaliased) T-sampled version of $x_c(t)$ and $v[n]$ is an L-upsampled version of $x[m]$. If we filter $v[n]$ with an ideal π/L-bandwidth lowpass filter (with DC gain L) to obtain $y[n]$, then $y[n]$ will be a T/L-sampled version of $x_c(t)$. This process is illustrated below.

$$\begin{tikzcd} x[m] \arrow[r] & \boxed{\uparrow L} \arrow[r, "v[n]"] & \overset{\text{DC gain } L}{\boxed{\text{LPF } \frac{\pi}{L}}} \arrow[r] & y[n] \end{tikzcd}$$

We justify our claims about interpolation using frequency-domain arguments. From the sampling theorem, we know that T-sampling $x_c(t)$ to create $x[n]$ yields

$$X(e^{j\omega}) \;=\; \frac{1}{T}\sum_k X_c\left(j\Big(\frac{\omega - 2\pi k}{T}\Big)\right)$$

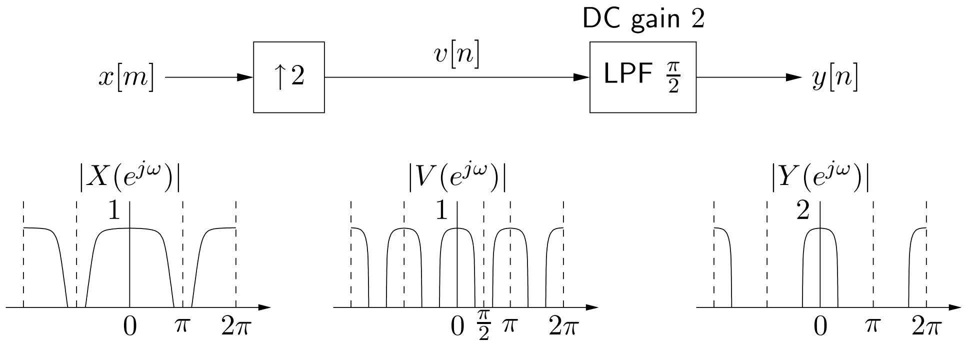
After upsampling by factor L, (1) implies

$$V(e^{j\omega}) \;=\; \frac{1}{T}\sum_k X_c\left(j\Big(\frac{\omega L - 2\pi k}{T}\Big)\right) \;=\; \frac{1}{T}\sum_k X_c\left(j\Big(\frac{\omega - \frac{2\pi}{L}k}{T/L}\Big)\right)$$

Lowpass filtering with cutoff $\frac{\pi}{L}$ and gain L yields

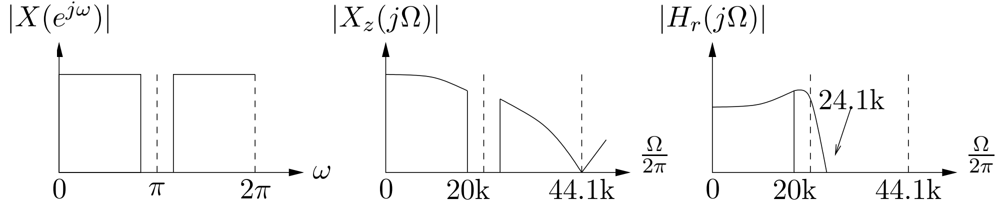
$$Y(e^{j\omega}) \;=\; \frac{L}{T}\sum_{k:\frac{k}{L}\in\mathbb{Z}} X_c\left(j\Big(\frac{\omega - \frac{2\pi}{L}k}{T/L}\Big)\right) \;=\; \frac{L}{T}\sum_l X_c\left(j\Big(\frac{\omega - 2\pi l}{T/L}\Big)\right)$$

since spectral copies with indices other than $k = lL$ (for $l \in \mathbb{Z}$) are removed. Clearly, this process yields a T/L-sampled version of $x_c(t)$. The figure below illustrates these frequency-domain arguments for $L = 2$.

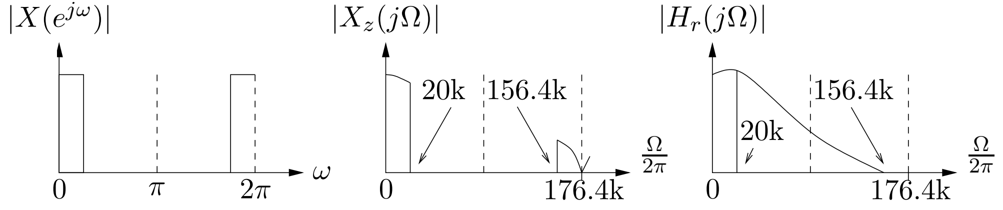


- *Application of Interpolation—Oversampling in CD Players:*

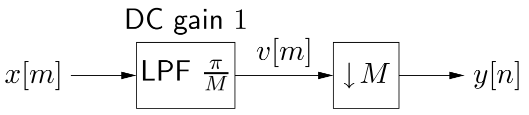
  The digital audio signal on a CD is a 44.1 kHz sampled representation of a continuous signal with bandwidth 20 kHz. With a standard ZOH-DAC, the analog reconstruction filter would have passband edge at 20 kHz and stopband edge at 24.1 kHz. (See the figure below.) With such a narrow transition band, this would be a difficult (and expensive) filter to build.



  If digital interpolation is used prior to reconstruction, the effective sampling rate can be increased and the reconstruction filter's transition band can be made much wider, resulting in a much simpler (and cheaper) analog filter. The figure below illustrates the case of interpolation by 4. The reconstruction filter has passband edge at 20 kHz and stopband edge at 156.4 kHz, resulting in a much wider transition band and therefore an easier filter design.



- *Decimation:* Decimation is the processing of filtering and downsampling a signal to decrease its effective sampling rate, as illustrated below. The filtering is employed to prevent aliasing that might otherwise result from downsampling.



  To be more specific, say that

$$x_c(t) = x_l(t) + x_b(t),$$

  where $x_l(t)$ is a lowpass component bandlimited to $\frac{1}{2MT}$ Hz and $x_b(t)$ is a bandpass component with energy between $\frac{1}{2MT}$ and $\frac{1}{2T}$ Hz. If sampling $x_c(t)$ with interval T yields an unaliased discrete representation $x[m]$, then decimating $x[m]$ by factor M will yield $y[n]$, an unaliased MT-sampled representation of lowpass component $x_l(t)$.

We offer the following justification of the previously described decimation procedure. From the sampling theorem, we have

$$X(e^{j\omega}) \;=\; \frac{1}{T}\sum_k X_l\left(j\Big(\frac{\omega - 2\pi k}{T}\Big)\right) + \frac{1}{T}\sum_k X_b\left(j\Big(\frac{\omega - 2\pi k}{T}\Big)\right)$$

The bandpass component $X_b(j\Omega)$ is then removed by π/M-lowpass filtering, giving

$$V(e^{j\omega}) \;=\; \frac{1}{T}\sum_k X_l\left(j\Big(\frac{\omega - 2\pi k}{T}\Big)\right)$$

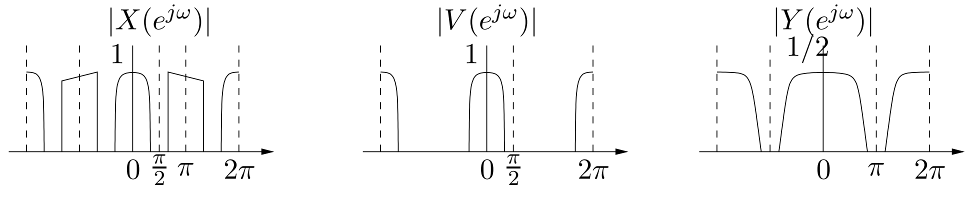
Finally, downsampling yields

$$\begin{aligned}
Y(e^{j\omega}) \;&=\; \frac{1}{MT}\sum_{p=0}^{M-1}\sum_k X_l\left(j\Big(\frac{\frac{\omega-2\pi p}{M} - 2\pi k}{T}\Big)\right)\\
&=\; \frac{1}{MT}\sum_{p=0}^{M-1}\sum_k X_l\left(j\Big(\frac{\omega - 2\pi(kM+p)}{MT}\Big)\right)\\
&=\; \frac{1}{MT}\sum_l X_l\left(j\Big(\frac{\omega - 2\pi l}{MT}\Big)\right)
\end{aligned}$$

which is clearly a MT-sampled version of $x_l(t)$. A frequency-domain illustration for $M = 2$ appears below.

DC gain 1

$$\begin{tikzcd}
x[m] \arrow[r] & \boxed{\text{LPF } \frac{\pi}{2}} \arrow[r, "v[m]"] & \boxed{\downarrow 2} \arrow[r] & y[n]
\end{tikzcd}$$



- *Resampling with Rational Factor:* Interpolation by L and decimation by M can be combined to change the effective sampling rate of a signal by the rational factor $\frac{L}{M}$. This process is called "resampling" or "sample-rate conversion". Rather than cascading an anti-imaging filter for interpolation with an anti-aliasing filter for decimation, we implement one filter with the minimum of the two cutoffs ($\frac{\pi}{L}, \frac{\pi}{M}$) and the multiplication of the two DC gains (L and 1), as illustrated below.

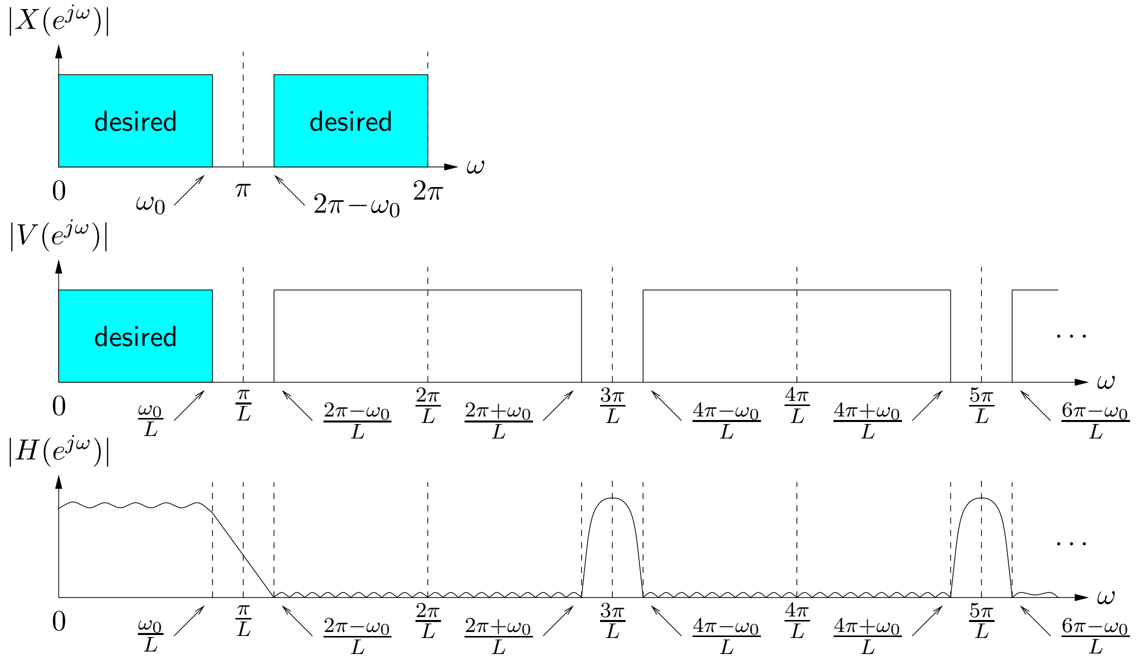
DC gain L

$$\begin{tikzcd}
{} \arrow[r] & \boxed{\uparrow L} \arrow[r] & \boxed{\text{LPF } \min\{\frac{\pi}{L}, \frac{\pi}{M}\}} \arrow[r] & \boxed{\downarrow M} \arrow[r] & {}
\end{tikzcd}$$

- *Digital Filter Design for Interpolation and Decimation:* First we treat filter design for interpolation. Consider an input signal $x[n]$ that is ω_0-bandlimited in the DTFT domain. If we upsample by factor L to get $v[m]$, the desired portion of $V(e^{j\omega})$ is the spectrum in $\left[-\frac{\pi}{L}, \frac{\pi}{L}\right)$, while the undesired portion is the remainder of $[-\pi, \pi)$. Noting from the figure below that $V(e^{j\omega})$ has zero energy in the regions

$$\left[\frac{2k\pi + \omega_0}{L}, \frac{2(k+1)\pi - \omega_0}{L}\right), \quad k \in \mathbb{Z}$$

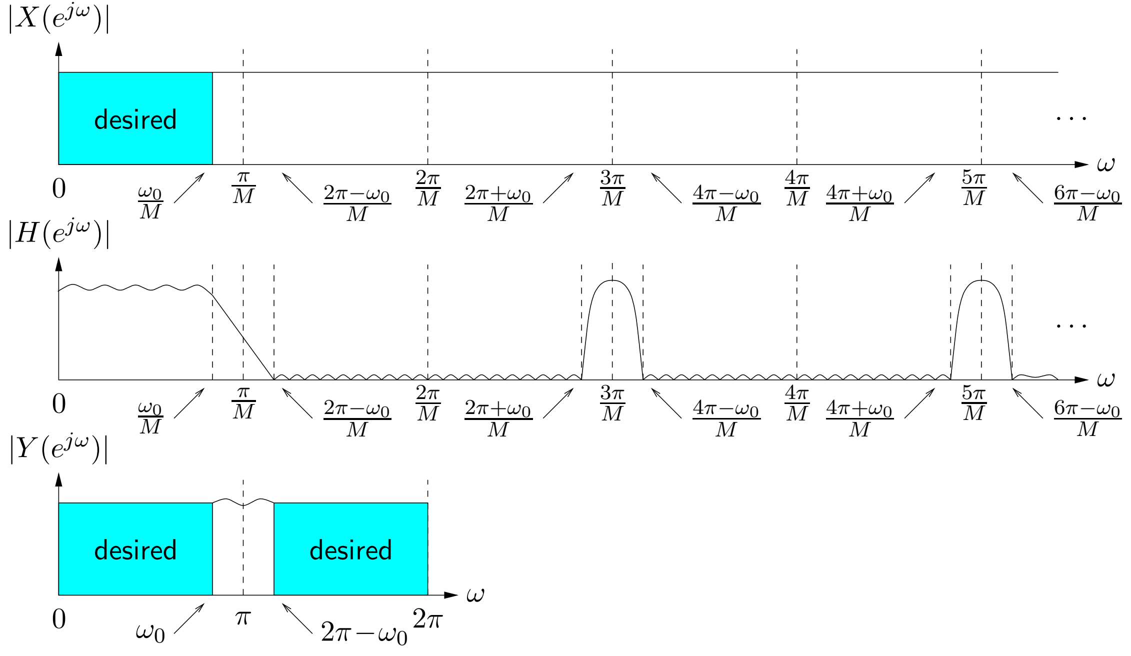
the anti-imaging filter can be designed with transition bands in these regions (rather than passbands or stopbands). For a given number of taps, the additional degrees of freedom offered by these transition bands allows for better responses in the passbands and stopbands. The resulting filter design specifications are shown in the bottom subplot below.



Next we treat filter design for decimation. Say that the *desired* spectral component of the input signal is bandlimited to $\frac{\omega_0}{M} < \frac{\pi}{M}$ and we have decided to downsample by M. The goal is to minimally distort the input spectrum over $\left[-\frac{\omega_0}{M}, \frac{\omega_0}{M}\right)$, i.e., the post-decimation spectrum over $[-\omega_0, \omega_0)$. Thus, we must not allow any aliased signals to enter $[-\omega_0, \omega_0)$. To allow for extra degrees of freedom in the filter design, we *do* allow aliasing to enter the post-decimation spectrum outside of $[-\omega_0, \omega_0)$ within $[-\pi, \pi)$. Since the input spectral regions which alias outside of $[-\omega_0, \omega_0)$ are given by

$$\left[\frac{2k\pi + \omega_0}{M}, \frac{2(k+1)\pi - \omega_0}{M}\right), \quad k \in \mathbb{Z}$$

(as shown in the figure below), we can treat these regions as transition bands in the filter design. The resulting filter design specifications are illustrated in the middle subplot below.



We now consider the effect of the passband and stopband ripples in $H(e^{j\omega})$. If we assume that $|X(e^{j\omega})| = 1$, as in the figure above, then, in the worst case, a height-δ_s ripple from each of the $M-1$ stopbands could alias onto a height-δ_p ripple from the passband. Remembering that decimation results in an amplitude reduction of $\frac{1}{M}$, this yields a worst-case post-decimation ripple of

$$\frac{1}{M}\big(\delta_p + (M-1)\delta_s\big).$$

Thus, it might be advantageous to choose $\delta_s \ll \delta_p$ when M is large. When $\delta_s = \delta_p = \delta$, however, the post-decimation ripple reduces to δ.

- *The Noble Identities:* The Noble identities (illustrated in the two block diagrams below) describe when it is possible to reverse the order of upsampling/downsampling and filtering. We prove the Noble identities showing the equivalence of each pair of block diagrams.

  The Noble identity for interpolation can be depicted as follows:

$$x[n] \rightarrow \boxed{\uparrow L} \xrightarrow{v_1[m]} \boxed{H(z^L)} \rightarrow y[m] \quad \Leftrightarrow \quad x[n] \rightarrow \boxed{H(z)} \xrightarrow{v_2[n]} \boxed{\uparrow L} \rightarrow y[m]$$

  For the left side of the preceding diagram, we have

$$\begin{aligned} Y(z) &= H(z^L)V_1(z) \qquad \text{where} \quad V_1(z) = X(z^L) \\ &= H(z^L)X(z^L) \end{aligned}$$

  while for the right side,

$$\begin{aligned} Y(z) &= V_2(z^L) \qquad \text{where} \quad V_2(z) = H(z)X(z) \\ &= H(z^L)X(z^L) \end{aligned}$$

  Thus we have established the Noble identity for interpolation.

The Noble identity for decimation can be depicted as follows:

$$\begin{tikzcd}
x[n] \arrow[r] & \boxed{\downarrow M} \arrow[r, "v_1{[m]}"] & \boxed{H(z)} \arrow[r] & y[m] & \Leftrightarrow & x[n] \arrow[r] & \boxed{H(z^M)} \arrow[r, "v_2{[n]}"] & \boxed{\downarrow M} \arrow[r] & y[m]
\end{tikzcd}$$

For the left side of the preceding diagram, we have

$$\begin{aligned}
Y(z) &= H(z)V_1(z) \\
&= H(z)\frac{1}{M}\sum_{k=0}^{M-1} X(e^{-j\frac{2\pi}{M}k}z^{\frac{1}{M}})
\end{aligned}$$

while for the right side,

$$\begin{aligned}
Y(z) &= \frac{1}{M}\sum_{k=0}^{M-1} V_2(e^{-j\frac{2\pi}{M}k}z^{\frac{1}{M}}) \qquad \text{where} \quad V_2(z) = X(z)H(z^M) \\
&= \frac{1}{M}\sum_{k=0}^{M-1} X(e^{-j\frac{2\pi}{M}k}z^{\frac{1}{M}})H(e^{-j\frac{2\pi}{M}Mk}z^{\frac{M}{M}}) \\
&= H(z)\frac{1}{M}\sum_{k=0}^{M-1} X(e^{-j\frac{2\pi}{M}k}z^{\frac{1}{M}})
\end{aligned}$$

Thus we have established the Noble identity for decimation. Note that the impulse response of $H(z^L)$ is the L-upsampled impulse response of $H(z)$.

- *Polyphase Interpolation:* Recall the standard interpolation procedure illustrated below.

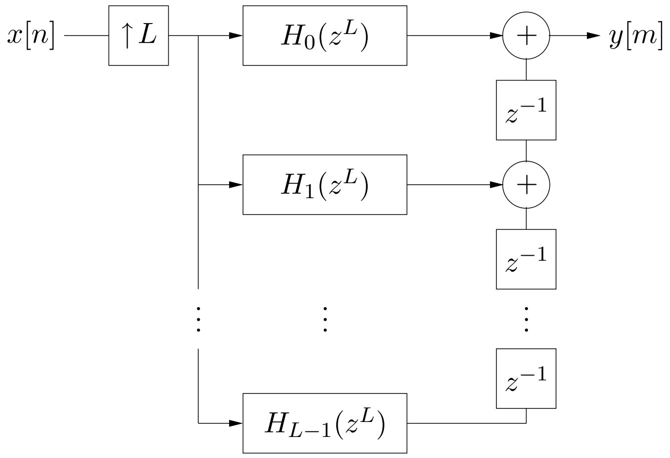
$$\begin{tikzcd}
x[n] \arrow[r] & \boxed{\uparrow L} \arrow[r] & \boxed{H(z)} \arrow[r] & y[m]
\end{tikzcd}$$

Note that this procedure is computationally inefficient because the lowpass filter operates on a sequence that is mostly composed of zeros. Through the use of the Noble identities, it is possible to rearrange the preceding block diagram so that operations on zero-valued samples are avoided.

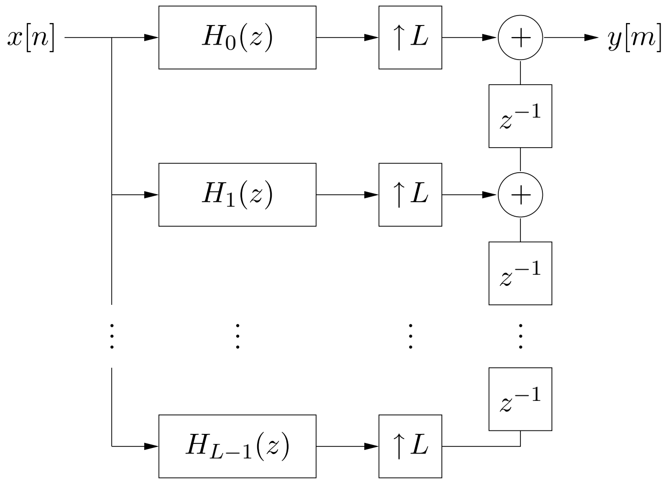
In order to apply the Noble identity for interpolation, we must transform $H(z)$ into its upsampled polyphase components $H_p(z^L),\ p = 0, \dots, L-1$.

$$\begin{aligned}
H(z) &= \sum_n h[n]z^{-n} \\
&= \sum_k \sum_{p=0}^{L-1} h[kL+p]z^{-(kL+p)} \quad \text{via} \quad k := \lfloor \tfrac{n}{L} \rfloor,\ p := \langle n \rangle_L \\
&= \sum_{p=0}^{L-1} \left(\sum_k h_p[k]z^{-kL} \right) z^{-p} \quad \text{via} \quad h_p[k] := h[kL+p] \\
&= \sum_{p=0}^{L-1} H_p(z^L)z^{-p}
\end{aligned}$$

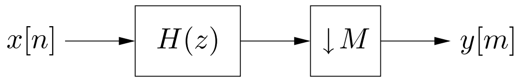
Above, $\lfloor \cdot \rfloor$ denotes the floor operator and $\langle \cdot \rangle_M$ the modulo-M operator. Note that the p^{th} polyphase filter $h_p[k]$ is constructed by downsampling the "master filter" $h[n]$ at offset p. Using the upsampled polyphase components, the preceding block diagram can be redrawn as below.



Applying the Noble identity for interpolation to the figure above yields the figure below. The ladder of upsamplers and delays on the right below accomplishes a form of parallel-to-serial conversion.



- *Polyphase Decimation:* Recall the standard decimation method illustrated below.

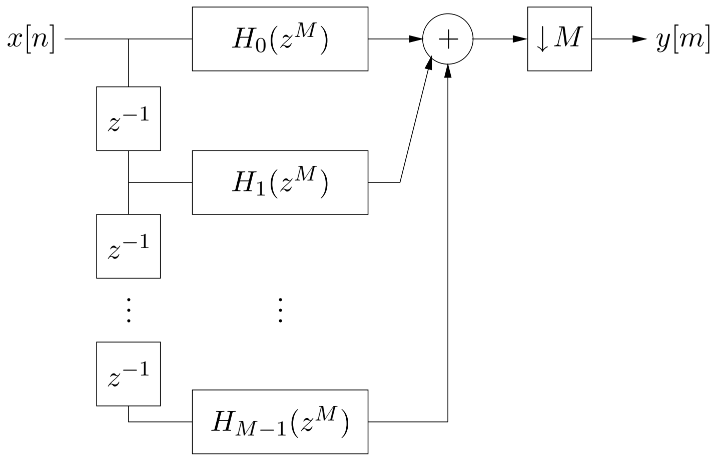


Note that this procedure is computationally inefficient because it discards the majority of the computed filter outputs. Through the use of the Noble identities, it is possible to rearrange the block diagram above so that filter outputs are not discarded.

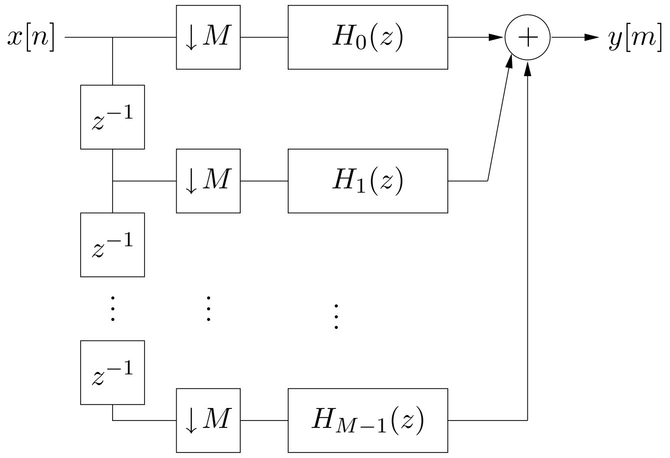
In order to apply the Noble identity for decimation, we must transform $H(z)$ into its upsampled polyphase components $H_p(z^M), p = 0, \ldots, M-1$, defined previously in the context of polyphase interpolation.

$$\begin{aligned}
H(z) &= \sum_n h[n] z^{-n} \\
&= \sum_k \sum_{p=0}^{M-1} h[kM+p] z^{-kM-p} \quad \text{via} \quad k := \lfloor \tfrac{n}{M} \rfloor,\ p := \langle n \rangle_M \\
&= \sum_{p=0}^{M-1} \left(\sum_k h_p[k] z^{-kM} \right) z^{-p} \quad \text{via} \quad h_p[k] := h[kM+p] \\
&= \sum_{p=0}^{M-1} H_p(z^M) z^{-p}
\end{aligned}$$

Using these upsampled polyphase components, the preceding block diagram can be redrawn as below.



Applying the Noble identity for decimation to the figure above yields the figure below. The ladder of delays and downsamplers on the left below accomplishes a form of serial-to-parallel conversion.



- *Computational Savings of Polyphase Interpolation/Decimation:* Assume that we design FIR LPF $H(z)$ with N taps, requiring N multiplies per output. For standard decimation by factor M, we have N multiplies per intermediate sample and M intermediate samples per output, giving NM multiplies per output.

  For polyphase decimation, we have $\frac{N}{M}$ multiplies per branch and M branches, giving a total of N multiplies per output. The assumption of $\frac{N}{M}$ multiplies per branch follows from the fact that $h[n]$ is downsampled by M to create each polyphase filter. Thus, we conclude that the standard implementation requires M times as many operations as its polyphase counterpart. (For decimation we count multiplies per output, rather than per input, to avoid confusion, since only every M^{th} input produces an output.)

  From this result, it appears that the the number of multiplications required by polyphase decimation is independent of the decimation rate M. However, it should be remembered that the length N of the $\frac{\pi}{M}$-lowpass FIR filter $H(z)$ will typically be proportional to M. This is suggested by, e.g., the Kaiser FIR-length approximation formula

$$N \approx \frac{-10\log_{10}(\delta_p \delta_s) - 13}{2.324\Delta\omega}$$

  where $\Delta\omega$ is the transition bandwidth in radians, and δ_p and δ_s are the passband and stopband ripple levels. Recall that, to preserve a fixed signal bandwidth, the transition bandwidth $\Delta\omega$ will be linearly proportional to the cutoff $\frac{\pi}{M}$, so that N will be linearly proportional to M. In summary, polyphase decimation by factor M requires N multiplies per output, where N is the filter length, and where N is linearly proportional to M.

  Using similar arguments for polyphase interpolation, we could find essentially the same result. Polyphase interpolation by factor L requires N multiplies per input, where N is the filter length, and where N is linearly proportional to the interpolation factor L. (For interpolation we count multiplies per input, rather than per output, to avoid confusion, since M outputs are generated in parallel.)

- *A Group-Delay Interpretation of Polyphase Filters:* Previously, polyphase interpolation and decimation were derived from the Noble identities and motivated for reasons of computational efficiency. Here we present a different interpretation of the (ideal) polyphase filter.

  Assume that $H(z)$ is an ideal lowpass filter with gain L, cutoff $\frac{\pi}{L}$, and constant group delay of d:

$$H(e^{j\omega}) = \begin{cases} Le^{-jd\omega} & \omega \in [-\frac{\pi}{L}, \frac{\pi}{L}) \\ 0 & \omega \in [-\pi, -\frac{\pi}{L}) \cup [\frac{\pi}{L}, \pi) \end{cases}$$

  Recall that the polyphase filters are defined as

$$h_p[k] = h[kL+p] \quad \text{for} \quad p = 0, \dots, L-1.$$

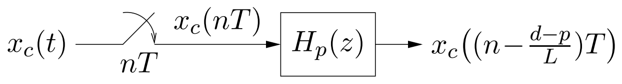
  In other words, $h_p[k]$ is an advanced (by p samples) and downsampled (by factor L) version of $h[n]$.

$$\begin{tikzcd} h[n] \arrow[r] & \boxed{z^p} \arrow[r, "{v[n]}"] & \boxed{\downarrow L} \arrow[r] & h_p[k] \end{tikzcd}$$

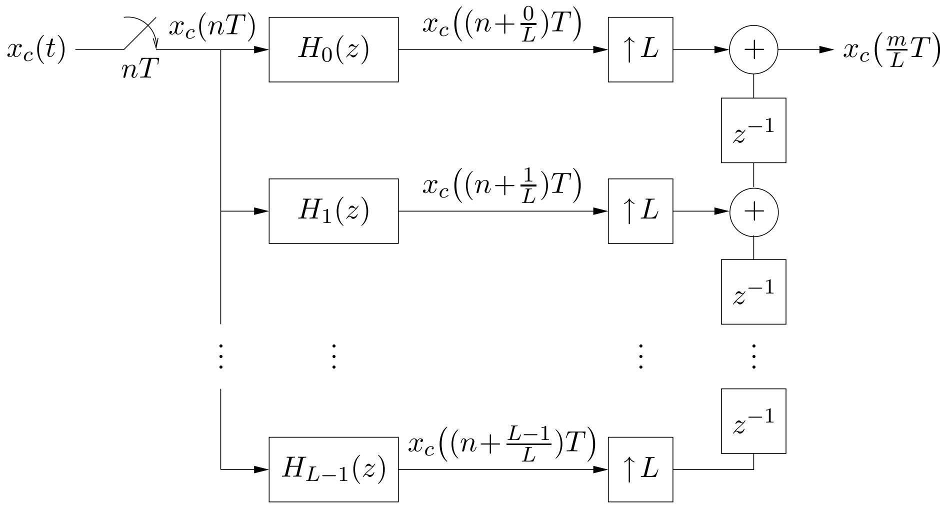
The DTFT of the p^{th} polyphase filter impulse response is then

$$\begin{aligned}
H_p(z) &= \frac{1}{L}\sum_{l=0}^{L-1} V(e^{-j\frac{2\pi}{L}l}z^{\frac{1}{L}}) \qquad \text{where} \quad V(z) = H(z)z^p \\
&= \frac{1}{L}\sum_{l=0}^{L-1} e^{-j\frac{2\pi}{L}lp}z^{\frac{p}{L}}H(e^{-j\frac{2\pi}{L}l}z^{\frac{1}{L}}) \\
H_p(e^{j\omega}) &= \frac{1}{L}\sum_{l=0}^{L-1} e^{j(\frac{\omega-2\pi l}{L})p}H(e^{j\frac{\omega-2\pi l}{L}}) \\
&= \frac{1}{L}e^{j\frac{\omega}{L}p}H(e^{j\frac{\omega}{L}}) \quad \text{for} \quad |\omega| \le \pi \\
&= e^{-j\frac{d-p}{L}\omega} \quad \text{for} \quad |\omega| \le \pi
\end{aligned}$$

Thus, the ideal p^{th} polyphase filter has a constant magnitude response of one and a constant group delay of $\frac{d-p}{L}$ samples. The implication is that if the input to the p^{th} polyphase filter is the unaliased T-sampled representation $x[n] = x_c(nT)$, then the output of the filter would be the unaliased T-sampled representation $y_p[n] = x_c\big((n-\frac{d-p}{L})T\big)$.

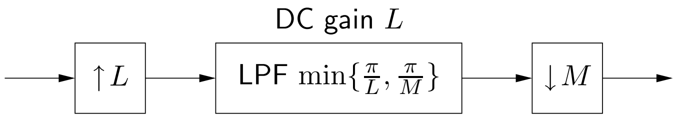


The block diagram below shows the role of polyphase interpolation filters assuming zero-delay (i.e., $d = 0$) processing for simplicity. Essentially, the p^{th} filter interpolates the waveform $\frac{p}{L}$-way between consecutive input samples. The L polyphase outputs are then interleaved to create the output stream. Assuming that $x_c(t)$ is bandlimited to $\frac{1}{2T}$ Hz, perfect polyphase filtering yields a perfectly interpolated output. In practice, we use causal FIR approximations of the polyphase filters $h_p[k]$ (which correspond to some causal FIR approximation of the master filter $h[n]$).



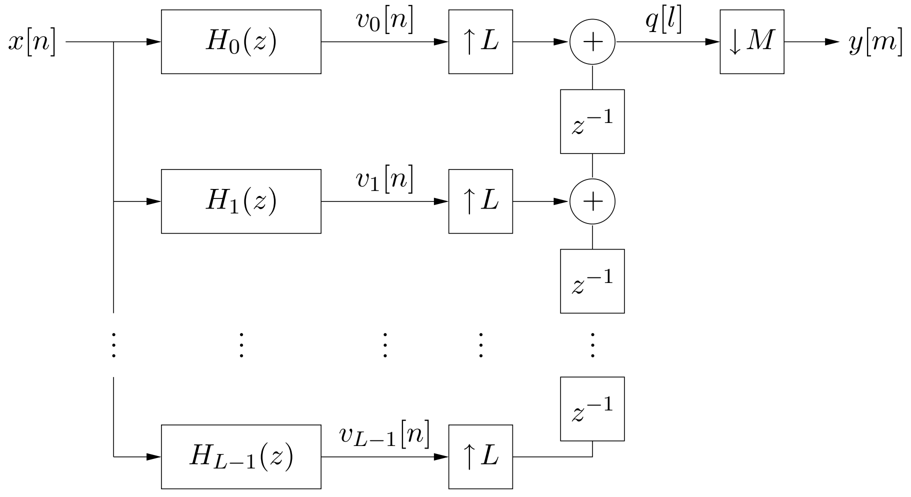
$x_c(t)$
nT
$x_c(nT)$
$H_0(z)$
$x_c\big((n+\frac{0}{L})T\big)$
$\uparrow L$
$+$
$x_c\big(\frac{m}{L}T\big)$
z^{-1}
$H_1(z)$
$x_c\big((n+\frac{1}{L})T\big)$
$\uparrow L$
$+$
z^{-1}
z^{-1}
$H_{L-1}(z)$
$x_c\big((n+\frac{L-1}{L})T\big)$
$\uparrow L$

- *Polyphase Resampling with a Rational Factor:* Recall that resampling by rational rate $\frac{L}{M}$ can be accomplished through the following three-stage process.



If we implemented the upsampler/LPF pair with a polyphase filterbank, we would still waste computations due to eventual downsampling by M. Alternatively, if we implemented the LPF/downsampler pair with a polyphase filterbank, we would waste computations by feeding it the (mostly-zeros) upsampler output. Thus, we need to examine this problem in more detail.

Assume for the moment that we implemented the upsampler/LPF pair with a polyphase filterbank, giving the architecture below.



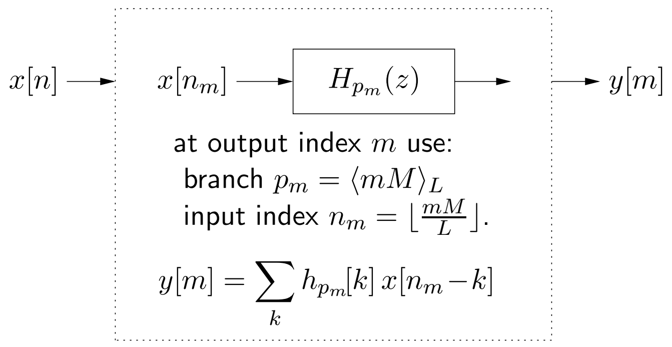
Keeping the "parallel-to-serial" interpretation of the upsampler/delay ladder in mind, the input sequence to the decimator $q[l]$ has the form

$$\begin{aligned} \ldots,\quad & v_0[0], v_1[0], v_2[0], \ldots, v_{L-1}[0], \\ & v_0[1], v_1[1], v_2[1], \ldots, v_{L-1}[1], \\ & v_0[2], v_1[2], v_2[2], \ldots, v_{L-1}[2], \ldots \end{aligned}$$

leading to the observation that

$$\begin{aligned} q[l] &= v_{\langle l \rangle_L}\big[\lfloor \tfrac{l}{L} \rfloor\big] \\ y[m] &= q[mM] \\ &= v_{\langle mM \rangle_L}\big[\lfloor \tfrac{mM}{L} \rfloor\big] \\ &= \sum_k h_{\langle mM \rangle_L}[k]\; x\big[\lfloor \tfrac{mM}{L} \rfloor - k\big] \end{aligned}$$

Thus, to calculate the resampled output at output index m, we should calculate only the output of branch number $\langle mM \rangle_L$ at input time index $\lfloor \frac{mM}{L} \rfloor$. No other branch outputs are calculated, so that no computations are wasted. The resulting structure is depicted below.



$x[n]$
$x[n_m]$
$H_{p_m}(z)$
$y[m]$
at output index m use:
branch $p_m = \langle mM \rangle_L$
input index $n_m = \lfloor \frac{mM}{L} \rfloor$.
$y[m] = \sum_k h_{p_m}[k]\, x[n_m - k]$

An equally-efficient structure could be obtained if we implemented the LPF/downsampler using an M-branch polyphase decimator which was fed with the proper sequence of input samples. However, this structure is not as elegant: rather than computing the output of *one* particular polyphase branch per output sample, we would need to add *all* branch outputs, but where each branch output was calculated using a particular *subset* of polyphase taps.

- *Computational Savings of Polyphase Resampling:* Recall the standard (non-polyphase) resampler below.

$$\begin{tikzcd} {} \arrow[r] & \boxed{\uparrow L} \arrow[r] & \boxed{\text{LPF } \min\{\frac{\pi}{L}, \frac{\pi}{M}\}} \arrow[r] & \boxed{\downarrow M} \arrow[r] & {} \end{tikzcd}$$

  For simplicity, assume that $L > M$. Since the length of an FIR filter is inversely proportional to the transition bandwidth (recalling Kaiser's formula), and the transition bandwidth is directly proportional to the cutoff frequency, we model the lowpass filter length as $N = RL$, where R is a constant that determines the filter's (and thus the resampler's) performance (independent of L and M). To compute one output point, we require M filter outputs, each requiring $N = RL$ multiplies, giving a total of MRL multiplies per output.

  In the polyphase implementation, calculation of one output point requires the computation of only one polyphase filter output. With $N = RL$ master filter taps and L branches, the polyphase filter length is R, so that only R multiplies are required per output. Thus, the polyphase implementation saves a factor of LM multiplies over the standard implementation!

- *Example: CD to DAT rate conversion:* Digital audio signals stored on compact digital discs (CDs) are sampled at 44.1 kHz, while those stored on digital audio tapes (DATs) are sampled at 48 kHz. Conversion from CD to DAT requires a rate change of

$$\frac{L}{M} = \frac{48000}{44100} = \frac{160}{147}$$

  Assuming that the audio signal is bandlimited to 20 kHz, we design our master lowpass filter with transition bands

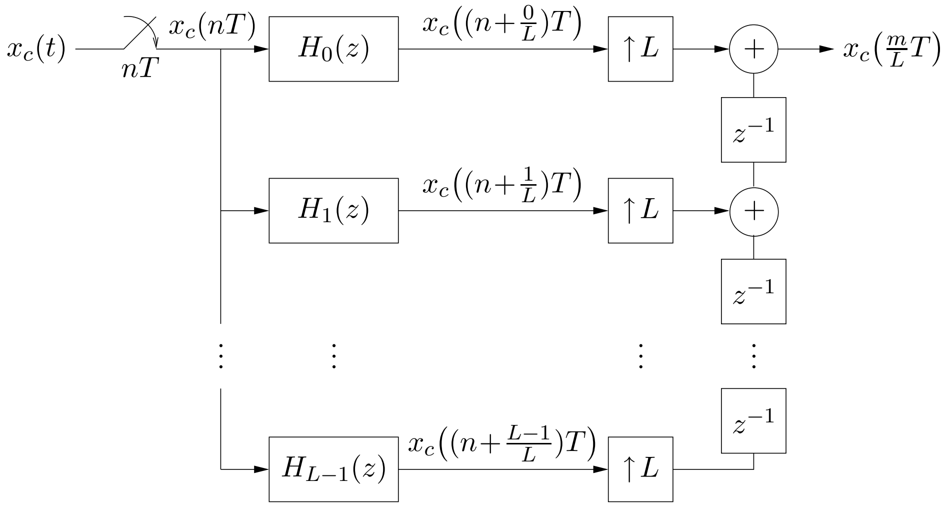
$$\left[\left(2k + \frac{20}{22.05}\right)\frac{\pi}{160}, \left(2(k+1) - \frac{20}{22.05}\right)\frac{\pi}{160}\right), \quad k \in \mathbb{Z}$$

  Keeping the passband and stopband ripple levels below -96 dB requires a filter with a length $N \approx 10000$, implying that the standard polyphase resampler will require about

$NM = 1.5$ million multiplications per output, or 70 billion multiplications per second! If the equivalent polyphase implementation is used instead, we require only $N/L \approx 62$ multiplies per output, or 3 million multiplications per second.

- *Polyphase Resampling with Arbitrary (Non-Rational or Time-Varying) Rate:* Though we have derived a computationally efficient polyphase resampler for rational factors $Q = \frac{L}{M}$, the structure will not be practical to implement for large L, such as might occur when the desired resampling factor Q is not well approximated by a ratio of two small integers. Furthermore, we may encounter applications in which Q is chosen on-the-fly, so that the number L of polyphase branches cannot be chosen a priori. Fortunately, a slight modification of our existing structure will allow us to handle both of these cases.

  Say that our goal is to produce the $\frac{Q}{T}$-rate samples $x_c(m\frac{T}{Q})$ given the $\frac{1}{T}$-rate samples $x_c(nT)$, where we assume that $x_c(t)$ is bandlimited to $\frac{1}{2T}$ and Q can be any positive real number. Consider, for a moment, the outputs of polyphase filters in an ideal zero-delay L-branch polyphase interpolation bank (as in the block diagram below).



$x_c(t)$
nT
$x_c(nT)$
$H_0(z)$
$x_c\big((n+\frac{0}{L})T\big)$
$\uparrow L$
$+$
$x_c\big(\frac{m}{L}T\big)$
z^{-1}
$H_1(z)$
$x_c\big((n+\frac{1}{L})T\big)$
$\uparrow L$
$+$
z^{-1}
z^{-1}
$H_{L-1}(z)$
$x_c\big((n+\frac{L-1}{L})T\big)$
$\uparrow L$

  We know that, at time index n, the p^{th} and $(p+1)^{th}$ filter outputs equal

$$x_c\big((n+\tfrac{p}{L})T\big) \quad \text{and} \quad x_c\big((n+\tfrac{p+1}{L})T\big)$$

  respectively. Because the highest frequency in $x_c(t)$ is limited to $\frac{1}{2T}$, the waveform cannot not change abruptly, and therefore cannot change significantly over a very small time interval. In fact, when L is large, the waveform is nearly linear in the time interval between $t = (n+\frac{p}{L})T$ and $t = (n+\frac{p+1}{L})T$, so that, for any $\alpha \in [0,1)$,

$$\begin{aligned} x_c\Big((n+\tfrac{p+\alpha}{L})T\Big) &= x_c\Big((1-\alpha)(n+\tfrac{p}{L})T + \alpha(n+\tfrac{p+1}{L})T\Big) \\ &\approx (1-\alpha)x_c\Big((n+\tfrac{p}{L})T\Big) + \alpha\, x_c\Big((n+\tfrac{p+1}{L})T\Big) \end{aligned}$$

  This suggests that we can closely approximate $x_c(t)$ at any $t \in \mathbb{R}$ by linearly interpolating adjacent-branch outputs of a polyphase filterbank with a large-enough L. The details are worked out below.

  Assume an ideal L-branch polyphase filterbank with d-delay master filter and T-sampled input, giving access to $x_c\big((n+\frac{p-d}{L})T)\big)$ for $n \in \mathbb{Z}$ and $p \in 0 \ldots L-1$. By linearly interpolating branch outputs p and $p+1$ at time n, we are able to closely approximate

$x_c\big((n+\frac{p-d+\alpha}{L})T)\big)$ for any $\alpha \in [0,1)$. We would like to approximate $y[m] = x_c(m\frac{T}{Q} - d\frac{T}{L})$ in this manner—note the inclusion of the master filter delay. So, for a particular m, Q, d, and L, we would like to find $n \in \mathbb{Z}$, $p \in 0 \ldots L-1$, and $\alpha \in [0,1)$ such that

$$\begin{aligned}
\left(n + \frac{p-d+\alpha}{L}\right)T &= m\frac{T}{Q} - d\frac{T}{L} \\
nL + p + \alpha &= \frac{mL}{Q} \\
&= \left(\frac{m}{Q}\right)L \\
&= \left(\left\lfloor\frac{m}{Q}\right\rfloor + \left\langle\frac{m}{Q}\right\rangle_1\right)L \\
&= \left\lfloor\frac{m}{Q}\right\rfloor L + \left\lfloor\left\langle\frac{m}{Q}\right\rangle_1 L\right\rfloor + \left\langle\left\langle\frac{m}{Q}\right\rangle_1 L\right\rangle_1 \\
&= \underbrace{\left\lfloor\frac{m}{Q}\right\rfloor}_{\in\,\mathbb{Z}} L + \underbrace{\left\lfloor\left\langle\frac{m}{Q}\right\rangle_1 L\right\rfloor}_{\in\, 0\ldots L-1} + \underbrace{\left\langle\frac{mL}{Q}\right\rangle_1}_{\in\,[0,1)}
\end{aligned}$$

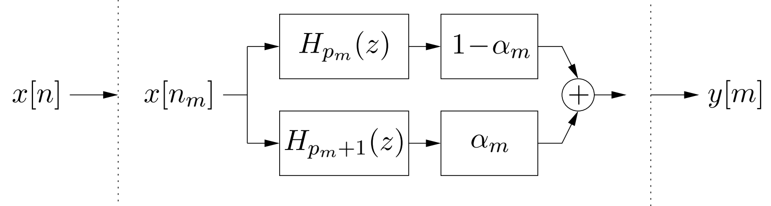
Thus, we have found suitable n, p, and α. Making clear the dependence on output time index m, we write

$$\begin{aligned}
n_m &= \left\lfloor\frac{m}{Q}\right\rfloor \\
p_m &= \left\lfloor\left\langle\frac{m}{Q}\right\rangle_1 L\right\rfloor \\
\alpha_m &= \left\langle\frac{mL}{Q}\right\rangle_1
\end{aligned}$$

and generate output $y[m] \approx x_c(m\frac{T}{Q} - d\frac{T}{L})$ via

$$y[m] \;=\; (1-\alpha_m)\sum_k h_{p_m}[k]\,x[n_m-k] + \alpha_m \sum_k h_{p_m+1}[k]\,x[n_m-k]$$

The arbitrary rate polyphase resampling structure is summarized in the diagram below.



at output index m, use
branch $p_m = \lfloor\langle\frac{m}{Q}\rangle_1 L\rfloor$
input index $n_m = \lfloor\frac{m}{Q}\rfloor$
weight $\alpha_m = \langle\frac{mL}{Q}\rangle_1$.

$$\begin{aligned}
y[m] \;=\;& (1-\alpha_m)\textstyle\sum_k h_{p_m}[k]\,x[n_m-k] \\
&+\alpha_m \textstyle\sum_k h_{p_m+1}[k]\,x[n_m-k]
\end{aligned}$$

Note that our structure refers to polyphase filters $H_{p_m}(z)$ and $H_{p_m+1}(z)$ for $p_m \in 0 \dots L–1$. This specifies the standard polyphase bank $\{H_0(z), \dots, H_{L-1}(z)\}$ plus the additional filter $H_L(z)$. Ideally the p^{th} filter has group delay $\frac{d-p}{L}$, so that $H_L(z)$ should advance the input one full sample relative to $H_0(z)$, i.e., $H_L(z) = zH_0(z)$. There are a number of ways to design/implement the additional filter.

1. Design a master filter of length $LR + 1$ (where R is the polyphase filter length), and then construct
$$h_p[k] = h[kL + p] \quad \text{for} \quad p \in 0 \dots L.$$
Note that $h_L[k] = h_0[k + 1]$ for $0 \leq k \leq R - 2$.
2. Set $H_L(z) = H_0(z)$ and advance the input stream to the last filter by one sample (relative to the other filters).

In certain applications the rate of resampling needs to be adjusted on-the-fly. The arbitrary rate resampler easily facilitates this requirement by replacing Q with Q_m in the definitions for n_m, p_m, and α_m.

Polyphase filter design follows either an *indirect* approach, where a master filter with DC gain L and cutoff frequency $\min\{\frac{\pi}{L}, \frac{Q\pi}{L}\}$ is designed and later split into polyphase filters, or a *direct* approach, where each filter is independantly designed to approximate an ideal fractional-delay filter. For the direct approach, we require that $Q \geq 1$ (or $Q \approx 1$) since the otherwise the filterbank would not perform the necessary anti-aliasing function.

Finally, it should be emphasized that L, the number of branches, must be chosen large enough so that the errors due to linear interpolation are negligible. (As such, it is not a function of Q.) If we instead use a more sophisticated interpolation method, e.g., Lagrange interpolation involving more than two branch outputs, then fewer polyphase filters would be necessary for the same overall performance, reducing the storage requirements for polyphase filter taps. On the other hand, combining the outputs of more branches requires more computations per output point. In the end, we have a tradeoff between storage and computation.

- *Multi-stage Interpolation:* In the single-stage interpolation structure illustrated below, the required impulse response of $H(z)$ can be very long for large L.

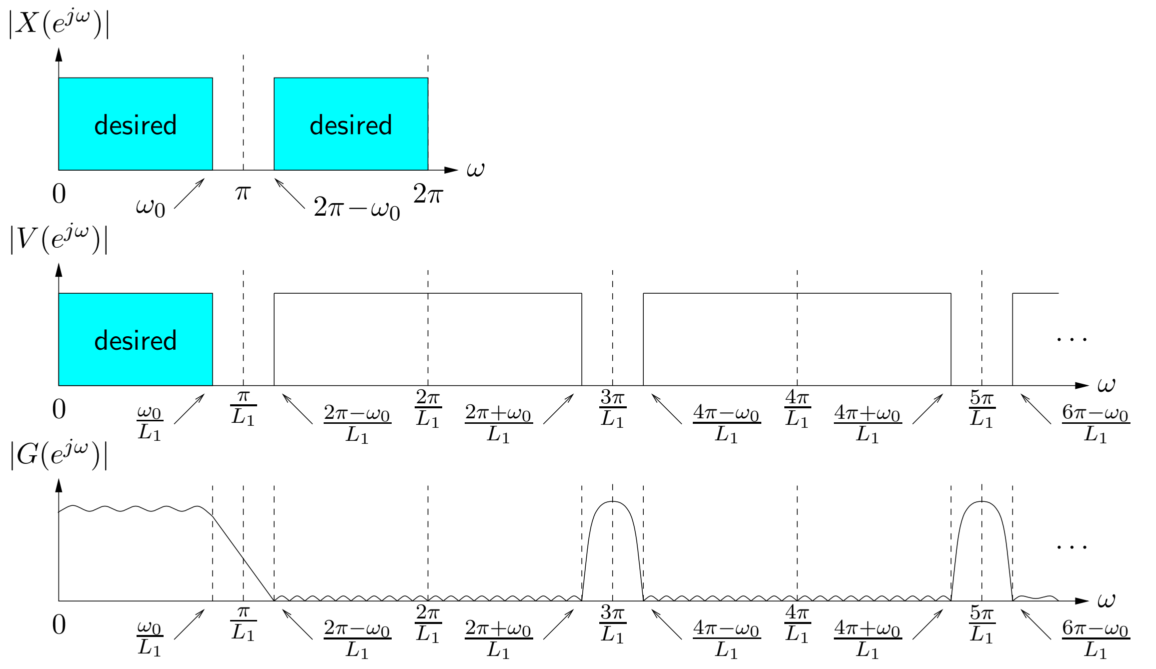
$$\begin{tikzcd}
x[n] \arrow[r] & \boxed{\uparrow L} \arrow[r] & \boxed{H(z)} \arrow[r] & y[m]
\end{tikzcd}$$

Significant computational savings may result from breaking the interpolation into multiple stages. In two-stage interpolation (illustrated below), we choose interpolation factors L_1 and L_2 such that $L_1L_2 = L$ and the filter pair $\{G(z), F(z)\}$ so that the overall performance meets some ripple specifications.

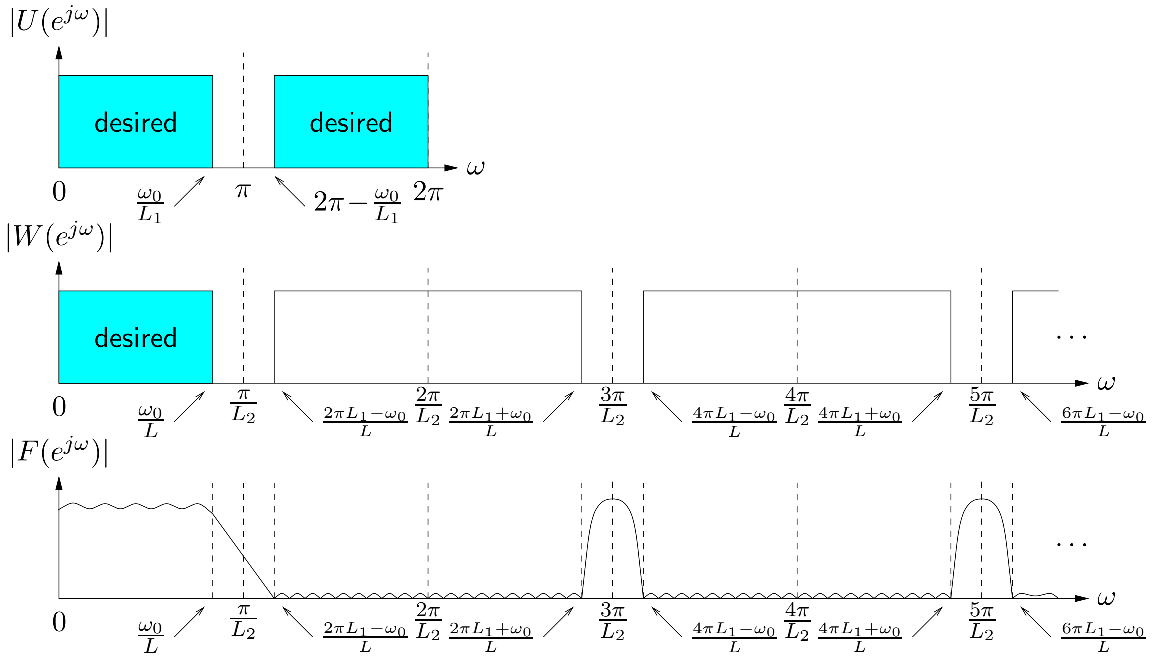
$$\begin{tikzcd}
x[n] \arrow[r] & \boxed{\uparrow L_1} \arrow[r, "v[l]"] & \boxed{G(z)} \arrow[r, "u[l]"] & \boxed{\uparrow L_2} \arrow[r, "w[m]"] & \boxed{F(z)} \arrow[r] & y[m]
\end{tikzcd}$$

Below we describe a general approach to the design of two-stage interpolators. Multi-stage interpolators with more than two stages can be designed using similar principles.

Consider the role of $G(z)$ to be the suppression of unwanted spectral images caused by L_1-upsampling. Assuming that $G(z)$ is designed correctly, $F(z)$ needs only to suppress the images caused by the L_2-upsampling. Assuming an input signal bandlimited to ω_0, design of $G(z)$ proceeds in accordance with the figure below. Note that this is a straightforward filter design for interpolation factor L_1.



For design of $F(z)$, consider that $u[l]$ has been upsampled by factor L_2 to produce $w[m]$; we want to filter out the unwanted spectral images in $w[m]$. The design procedure for $F(z)$ is very similar to the design procedure for $G(z)$ except that the upsampler input ($u[l]$) now has bandwidth $L_2\frac{\omega_0}{L} = \frac{\omega_0}{L_1}$. This procedure is illustrated in the next figure, were we used the fact that $L = L_1L_2$.



We now address the ripple specifications on $F(z)$ and $G(z)$ relative to those on $H(z)$. Because ripple from the passbands of $G(z)$ and $F(z)$ could add constructively, we set the passband ripple requirements for each filter equal to half of the total allowable passband ripple δ_p, i.e., that of $H(z)$. It is sufficient to set the stopband ripple requirements for each filter equal to the total allowable stopband ripple δ_s, i.e., that of $H(z)$, if we assume that the none of the transition-band gains exceed the passband gain.

In multi-stage interpolation, it is common to choose a small value for L_1 (e.g., $L_1 = 2$). This choice will become more clear later, when we consider the computational savings of multi-stage interpolation.

- *Example of Multi-stage Interpolation:* Let us first examine the computational savings of multi-stage interpolation through a simple example. Say that we have input signal with bandwidth $\omega_0 = 0.9\pi$ radians, and that we require an interpolation factor $L = 30$ with maximum passband ripple $\delta_p = 0.002$ and stopband ripple $\delta_s = 0.001$.

  With the one-stage interpolator below,

$$\begin{tikzcd} x[n] \arrow[r] & \boxed{\uparrow 30} \arrow[r] & \boxed{H(z)} \arrow[r] & y[m] \end{tikzcd}$$

  Kaiser's formula approximates the required FIR filter length to be

$$N_h \;\approx\; \frac{-10\log_{10}(\delta_p\delta_s) - 13}{2.3\Delta\omega} \;\approx\; 900,$$

  where we have chosen $\Delta\omega = \frac{2\pi - 2\omega_0}{L}$ as the width of the first transition band (i.e., ignoring the other transition bands for this rough approximation). Thus, a polyphase implementation of this one-stage interpolator would cost about 900 multiplies per input sample.

  With the two-stage interpolator below,

$$\begin{tikzcd} x[n] \arrow[r] & \boxed{\uparrow 2} \arrow[r] & \boxed{G(z)} \arrow[r] & \boxed{\uparrow 15} \arrow[r] & \boxed{F(z)} \arrow[r] & y[m] \end{tikzcd}$$

  the transition band in $G(z)$ has center $\omega = \frac{\pi}{2}$ and width $\frac{2\pi - 2\omega_0}{2} = 0.1\pi$ rad. Likewise, the transition bands in $F(z)$ have width $\frac{4\pi - 2\omega_0}{30} = \frac{2.2}{30}\pi$ rad. Plugging these specifications into the Kaiser length approximation, we obtain

$$N_g \;\approx\; 64 \quad \text{and} \quad N_f \;\approx\; 88.$$

  Already we see that it will be computationally easier to *design* two filters of lengths 64 and 88 than it would be to design one length-900 filter. Note that the filter-length reductions of this structure result from the fact that the transition bands in both $F(z)$ and $G(z)$ are much wider than the transition band in $H(z)$.

  The computational savings also carry over to the *operation* of the two-stage structure. Using a cascade of two single-stage polyphase interpolators to implement the two-stage scheme, we find that the first interpolator would require $N_g \approx 64$ multiplies per input point $x[n]$, while the second would require $N_f \approx 88$ multiplies per output of $G(z)$. Since $G(z)$ outputs two points per input $x[n]$, the two-stage structure would require a total of

$$\approx 64 + 2 \cdot 88 = 240$$

multiplies per input $x[n]$. Clearly this is a significant savings over the 900 multiplies required by the one-stage polyphase structure. Note that it was advantageous to choose the first upsampling ratio (L_1) as a small number, else the second stage of interpolation would need to operate at higher rate and the total number of multiplications per *input* point $x[n]$ would increase.

- *Computational Savings of Multistage Interpolation:* Let us now ask the general question: How much computation does a two-stage interpolator require relative to a one-stage interpolator as a function of input signal bandwidth ω_0, overall interpolation factor L, and first-stage interpolation factor L_1? Answering this question would tell us exactly how to pick the two-stage factors $\{L_1, L_2\}$ and when to use a single stage instead.

  For the two structures below (where $L = L_1L_2$),

$$\begin{tikzcd} x[n] \arrow[r] & \boxed{\uparrow L} \arrow[r] & \boxed{H(z)} \arrow[r] & y[m] \end{tikzcd}$$

$$\begin{tikzcd} x[n] \arrow[r] & \boxed{\uparrow L_1} \arrow[r] & \boxed{G(z)} \arrow[r] & \boxed{\uparrow L_2} \arrow[r] & \boxed{F(z)} \arrow[r] & y[m] \end{tikzcd}$$

  recall the first-transition-band specifications:

filter	band center	band width Δ
$H(z)$	$\frac{\pi}{L}$	$\frac{2\pi-2\omega_0}{L}$
$G(z)$	$\frac{\pi}{L_1}$	$\frac{2\pi-2\omega_0}{L_1}$
$F(z)$	$\frac{\pi}{L_2}$	$\frac{2\pi L_1-2\omega_0}{L}$

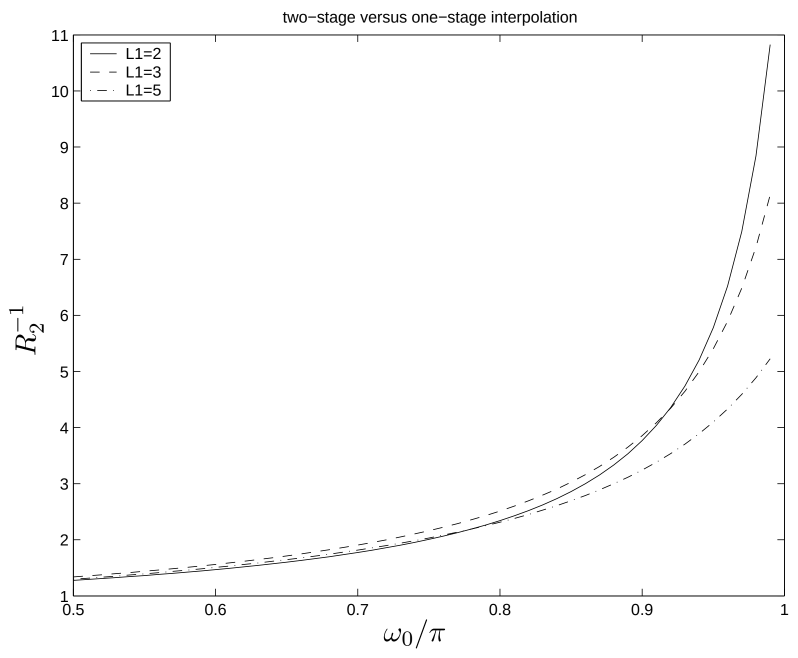
  Also recall that the passband ripple for G and F must be half that of H, say δ_p, while the stopband ripple is the same as that for H, say δ_s. Kaiser's formula approximates the required lengths of H, G and F to be

$$\begin{aligned} N_h &\approx \frac{-10\log_{10}(\delta_s\delta_p)-13}{2.3\Delta_h} = \frac{\beta}{2.3(2\pi-2\omega_0)}L \\ N_g &\approx \frac{-10\log_{10}(\delta_s\delta_p/2)-13}{2.3\Delta_g} = \frac{\beta+3}{2.3(2\pi-2\omega_0)}L_1 \\ N_f &\approx \frac{-10\log_{10}(\delta_s\delta_p/2)-13}{2.3\Delta_f} = \frac{\beta+3}{2.3(2\pi L_1-2\omega_0)}L \end{aligned}$$

  using $\beta := -10\log_{10}(\delta_s\delta_p) - 13$ for brevity. Assuming a polyphase implementation of each interpolation stage, the one-stage structure would require N_h multiplies per input point, while the two-stage structure would require $N_g + L_1N_f$ multiplies per input point. This yields the ratio

$$\begin{aligned} R_2(\omega_0, L, L_1, \beta) &:= \frac{\text{two-stage mults per input}}{\text{one-stage mults per input}} \\ &= \frac{\frac{\beta+3}{2.3(2\pi-2\omega_0)}L_1 + \frac{\beta+3}{2.3(2\pi L_1-2\omega_0)}LL_1}{\frac{\beta}{2.3(2\pi-2\omega_0)}L} \\ &= \frac{\beta+3}{\beta}\left(\frac{1}{L} + \frac{\pi-\omega_0}{\pi L_1-\omega_0}\right)L_1 \end{aligned}$$

Below we plot $R_2^{-1}(\omega_0, L, L_1, \beta)$ as a function of ω_0 for $L = 30$, $\beta = 44$ (from $\delta_p = 0.002$ and $\delta_s = 0.001$), and different choices of L_1. (Note: $R_2^{-1} > 1$ favors a two-stage implementation.) Note that the choice $L_1 = 2$ is optimal for high-bandwidth signals, while the choice $L_1 = 3$ is slightly better for low-bandwidth signals. If we were forced to choose a single value, we would be safe with $L_1 = 2$. For all ω_0 tested, the two-stage structure has lower implementation complexity than the one-stage structure.



- *Multi-stage Decimation:* In the single-stage decimation structure illustrated below, the required impulse response of $H(z)$ can be very long for large M.

$$\begin{tikzcd} x[n] \arrow[r] & \boxed{H(z)} \arrow[r] & \boxed{\downarrow M} \arrow[r] & y[m] \end{tikzcd}$$

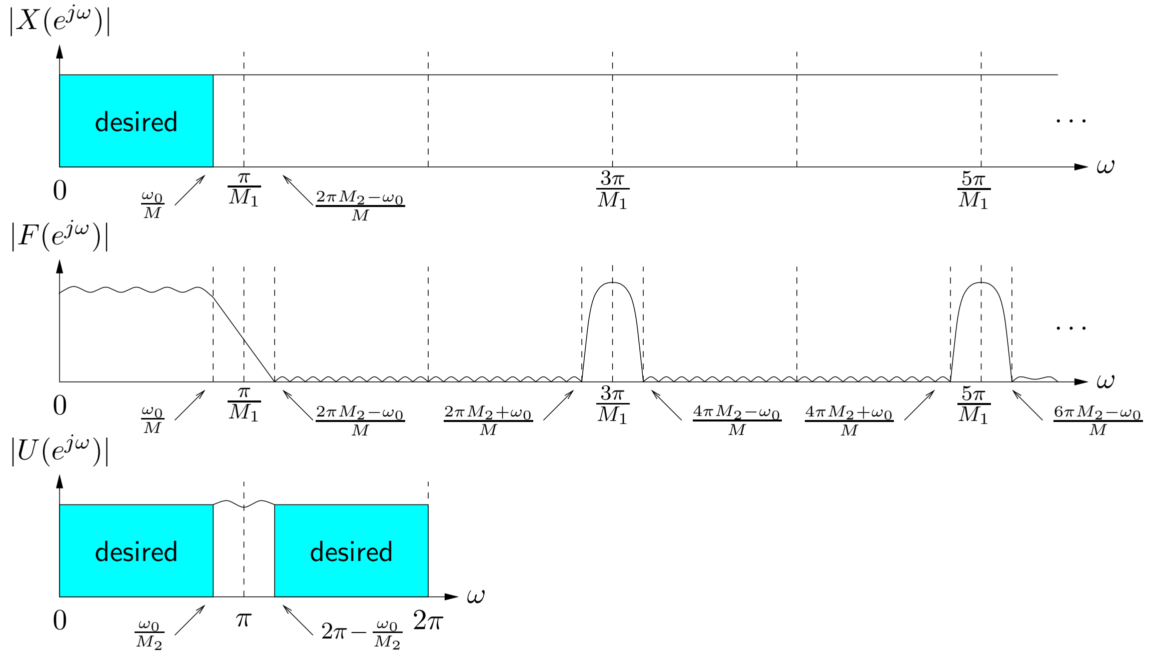
Significant computational savings may result from breaking the decimation into multiple stages. In two-stage decimation (illustrated below), we choose decimation factors M_1 and M_2 such that $M_1M_2 = M$ and the filter pair $\{F(z), G(z)\}$ so that the overall performance meets some ripple specifications.

$$\begin{tikzcd} x[n] \arrow[r] & \boxed{F(z)} \arrow[r, "{w[n]}"] & \boxed{\downarrow M_1} \arrow[r, "{u[l]}"] & \boxed{G(z)} \arrow[r, "{v[l]}"] & \boxed{\downarrow M_2} \arrow[r] & y[m] \end{tikzcd}$$

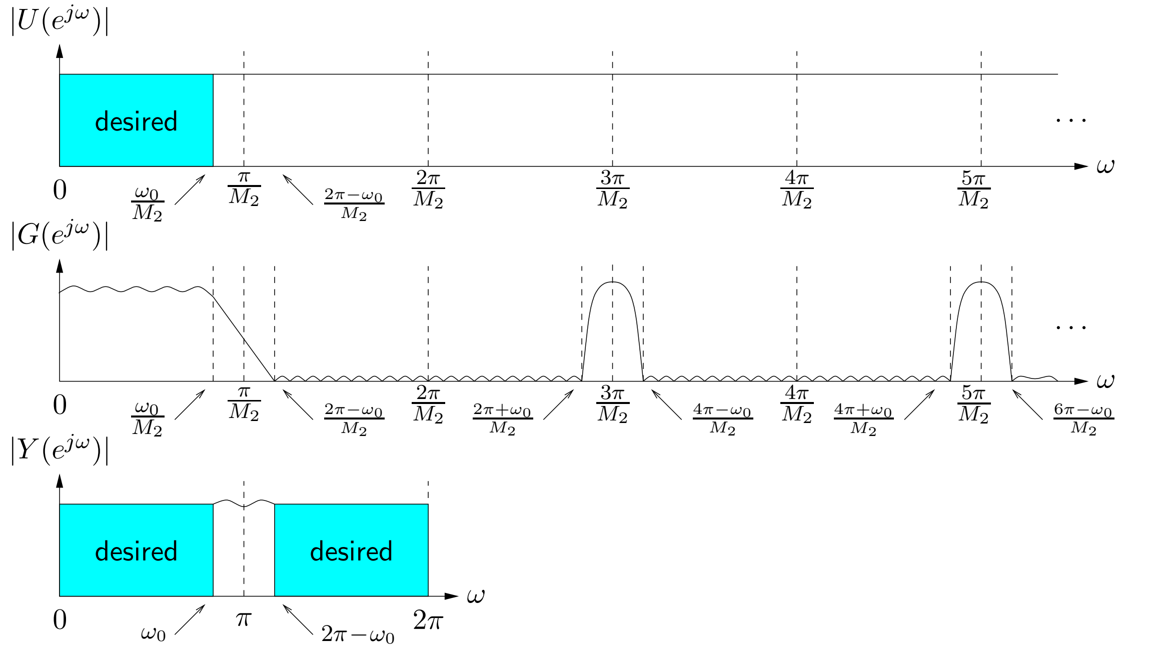
Below we describe a general approach to the design of two-stage decimators for the case where the input signal has equal energy at all frequencies, and where we intend that the input signal components at $\omega \in [-\frac{\omega_0}{M}, \frac{\omega_0}{M})$ are to be preserved through decimation (manifesting at $\omega \in [-\omega_0, \omega_0)$ in the decimated output signal). This approach can be easily extended to design multi-stage decimators with more than two stages.

Consider that $F(z)$ is an anti-aliasing filter for M_1-downsampling of the full-bandwidth signal $x[n]$. Assuming that $F(z)$ has been designed properly, $G(z)$ then functions as an anti-aliasing filter for M_2-downsampling of the full-bandwidth signal $u[l]$. Hence, the multi-stage design problem can be decoupled into two single-stage design problems.

Assuming that we want to preserve the input signal components in $\omega \in [-\frac{\omega_0}{M}, \frac{\omega_0}{M})$, design of $F(z)$ proceeds in accordance with the following figure. Note that this is a straightforward filter design for decimation factor M_1.



The objective of $G(z)$ is to anti-alias filter the signal $u[l]$. The design procedure for $G(z)$ is identical to the design procedure for $F(z)$ except that the input signal ($u[l]$) now has bandwidth $\frac{\omega_0}{M_2}$. As illustrated in the following figure, this is a straightforward filter design for decimation factor M_2.



We now address the ripple specifications on $F(z)$ and $G(z)$. Recall that, after the first decimation stage, the input signal components at frequency $\omega \in [-\frac{\omega_0}{M}, \frac{\omega_0}{M})$ will be corrupted by worst-case ripples of

$$\frac{1}{M_1}\big(\delta_{f,p} + (M_1 - 1)\delta_{f,s}\big),$$

where $\delta_{f,p}$ and $\delta_{f,s}$ denote the passband and stopband ripples of $F(z)$. Since this spectral region will occupy the passband of $G(z)$, the first-stage output ripples will, in the worst case, (approximately) add to the passband ripples of $G(z)$. If we assume that the signal components within the stopband of $G(z)$ have equal energy as the signal components in the passband, then after two stages of decimation, the input signal components at frequencies $\omega \in [-\frac{\omega_0}{M}, \frac{\omega_0}{M})$ will be corrupted by worst-case ripples of

$$\frac{1}{M_2}\left(\frac{1}{M_1}\big(\delta_{f,p} + (M_1 - 1)\delta_{f,s}\big) + \delta_{g,p} + (M_2 - 1)\delta_{g,s}\right) \;\approx\; \frac{\delta_{f,p}}{M} + \frac{\delta_{f,s} + \delta_{g,p}}{M_2} + \frac{(M_2 - 1)\delta_{g,s}}{M_2},$$

where the approximation holds when $M_1 \gg 1$, and where $\delta_{g,p}$ and $\delta_{g,s}$ denote the passband and stopband ripples of $G(z)$. The previous expression suggests that the ripples in $G(z)$ have a greater effect than those in $F(z)$, and that stopband ripples have a greater effect than passband ripples.

In multi-stage decimation, it is common to choose a small value for M_2 (e.g., $M_2 = 2$) since, by putting the bulk of the decimation in the first stage, the second stage can operate at a very low rate. As with the case of interpolation, one could analyze the computational savings precisely for various choices of M_2.