7. Multirate DSP

(Deleted in 2007 Syllabus).

2007 Syllabus: Decimation, Interpolation, Sampling rate conversion, Filter design and Implementation of sampling rate conversion.

Contents:
7.1 Time and frequency scaling in continuous-time systems
7.2 Transformation of the independent variable
7.3 Down-sampling
7.4 Up-sampling
7.5 Cascading sampling rate converters
7.6 Identities
7.7 FIR implementation of sampling rate conversion
7.8 Polyphase structures
7.9 Polyphase structure for a decimator

Discrete-time systems with different sampling rates at various parts of the system are called multirate systems. They are linear and time-varying systems. Integer sampling rate converters change the sampling frequency by an integer factor and rational sampling rate converters by a rational number. Here is a sampling of sampling rates in commercial applications (Mitra):

<table>
<thead>
<tr>
<th>Sampling Rates</th>
<th>Digital Audio</th>
<th>Video</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broadcasting – 32 kHz</td>
<td></td>
<td>Composite Video Signal</td>
</tr>
<tr>
<td>CD – 44.1 kHz</td>
<td></td>
<td>• NTSC – 14.3181818 MHz</td>
</tr>
<tr>
<td>DAT – 48 kHz</td>
<td></td>
<td>• PAL – 17.734475 MHz</td>
</tr>
<tr>
<td>Digital Component Video Signal</td>
<td></td>
<td>• Luminance – 13.5 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Color difference – 6.75 MHz</td>
</tr>
</tbody>
</table>
7.1 Time and frequency scaling in continuous-time systems

Illustration An audio signal recorded on cassette tape at a certain speed could be played back at a higher speed than that at which it was recorded. This is called time scaling, in particular, compression in the time domain, and results in an inverse effect in the frequency domain, i.e., an expansion of the frequency spectrum. Similarly when the audio signal is played back at a slower speed than the recording speed we have expansion in the time domain resulting in a corresponding compression of the spectrum in the frequency domain.

Given the signal \( x(t) \) and its Fourier transform \( X(\Omega) \), represented notationally by

\[
x(t) \leftrightarrow X(\Omega)
\]

then time scaling results in

\[
x(at) \leftrightarrow \frac{1}{|a|} X(\Omega/a)
\]

If \( a > 1 \) the scaling corresponds to compression in time. If, for instance, \( a = 2 \), we may visualize a new signal \( y_1(t) = x(2t) \); with \( t = 1 \), for instance, the value of \( x \), that is, \( x(2) \) that occurred at 2 seconds occurs at 1 second in the case of \( y_1 \), that is, \( y_1(1) \) – which is compression in time.
If \( x(t) \) is an audio signal recorded on tape then \( x(2t) \) could be the signal \( x(t) \) played back at twice the speed at which \( x(t) \) was recorded. The signal \( x(2t) \) varies more rapidly than \( x(t) \) and the playback frequencies are higher.

If \( a < 1 \) the scaling corresponds to **expansion in time**. If, for instance, \( a = 1/2 \), then \( x(t/2) \) is the signal \( x(t) \) played back at half the speed at which \( x(t) \) was recorded. The signal \( x(t/2) \) varies slower than \( x(t) \) and the playback frequencies are lower. Again, we may visualize this as a new signal \( y_2(t) = x(t/2) \); the value of \( x(.) \) that occurred at \( t/2 \) occurs at \( t \) in the case of \( y_2(.) \) – which is expansion in time.

Time expansion and frequency compression is found in data transmission from space probes to receiving stations on earth. To reduce the amount of noise superposed on the signal, it is necessary to keep the bandwidth of the receiver as small as possible. One means of doing this is to reduce the bandwidth of the signal: store the data collected by the probe, and then transmit it at a slower rate. Since the time-scaling factor is known, the signal can be reproduced at the receiver.

The corresponding operations in the case of discrete-time systems are not quite so straightforward owing to

1. The need to band limit the continuous-time signal prior to sampling, and
2. The need to avoid aliasing in the process of sampling

**Example 1.1** Consider the 4 Hz signal \( x(t) = \cos 2\pi 4t \) which is obviously band-limited to \( F_{\text{max}} = 4 \) Hz. It is sufficient to sample it at 8 Hz. Alternatively, the signal can be sampled at, say, 16 Hz or 20 Hz etc. Suppose that it has been over-sampled by a factor of, say, 6 at \( F_s = 48 \) Hz to give \( x(n) = \cos 2\pi 4n(1/48) = \cos (\pi n/6) \).

(a) If it is desired subsequently to generate from \( x(n) \) another signal \( x_1(n) \) that is a discrete-time version of \( x(t) \) sampled at \( F_{s1} = 16 \) Hz (sampling rate reduced by a factor of 3), then can we do this by simply dropping two samples of \( x(n) \) for every sample that we keep? That is \( x_1(n) = x(3n) \). This is called **down-sampling**.

(b) How do we generate from \( x(n) \) another signal \( x_2(n) \) that is a discrete-time version of \( x(t) \) sampled at, say, \( F_{s2} = 96 \) Hz (sampling rate doubled)? This is called **up-sampling**.

(c) Can we generate from \( x(n) \) another signal \( x_3(n) \) that is a discrete-time version of \( x(t) \) sampled at \( F_{s3} = 6 \) Hz?

We pick up on this problem again after covering transformation of the independent variable.

### 7.2 Transformation of the independent variable

**Time scaling** (Refer also to Section 7.5 of Signals and Systems, Oppenheim and Willsky.) Given the sequence \( x(n) \), the sequence \( y(n) = x(2n) \) is obtained by skipping odd-numbered samples in \( x(n) \) and retaining the even-numbered ones. The extension to \( y(n) = x(Mn) \) means we retain sample numbers \( 0, M, 2M, 3M, \ldots \), and skip the intervening \( M-1 \) samples between those we keep. The original sequence \( x(n) \) is obtained by sampling a continuous signal \( x(t) \) at a certain rate (perhaps over-sampling). The signal \( y(n) = x(Mn) \) is then obtained by reducing the sampling rate by a factor of \( M \) on the continuous-time signal \( x(t) \). This is known as **down-sampling** or **decimation** or sampling rate compression.

Similarly the process of constructing the sequence \( y(n) = x(n/L) \) from the sequence \( x(n) \) means we derive \( y(n) \) by inserting \( (L-1) \) sequence points with zero value between points of \( x(n) \). This is called **up-sampling** or **interpolation** or sampling rate expansion. (Inserting \( (L-1) \) zeros is
just one way of interpolating. It is also possible for the up-sampler to be followed by a digital system that replaces the inserted zeros with more appropriate values based on a linear combination of the \( x(n) \) samples.

In general, the result of time scaling a discrete-time signal is not just a stretched or compressed version of the original but possibly a totally different sequence/waveform.

**Example 7.2.1** Given that \( x(t) = e^{-5t}u(t) \) is sampled at 50 Hz, find an expression for \( x(n) \). Plot \( x(t) \), \( x(n) \) and \( x(2n) \). Sketch the spectrum of \( x(n) \).

**Solution** The sampling time is \( T = 0.02 \) sec. Replacing \( t \) with \( nT \) we get \( x(nT) = e^{-5nT}u(nT) \), or \( x(n) = \left(e^{-5.01}\right)^n u(n) = (0.905)^n u(n) \).

We show below three plots: (1) The continuous-time signal \( x(t) \), (2) The sampled (at 50 Hz) version \( x(n) \), and (3) \( x(2n) \), the 2-fold down-sampled version of \( x(n) \); this is equivalent to sampling \( x(t) \) at 25 Hz.

```matlab
t = 0 : 1/512: 1; xt = exp (-5*t); \%x(t) evaluated at 512 points
subplot(3, 1, 1), plot(t, xt); legend ('x(t) = exp(-5t)');
xlabel ('time, sec.'), ylabel('x(t)'); grid; title ('x(t) – Continuous-time')
\%
t1 = 0 : 0.02: 1; xn = exp (-5*t1); \%Sampled at 50 Hz.
subplot(3, 1, 2), stem(t1, xn); legend ('x(n) at 50 Hz');
xlabel ('time, sec.'), ylabel('x(n)'); grid; title ('x(nT) at T = 0.02 sec')
\%
t2 = 0 : 0.04: 1; xt2 = exp (-5*t2); \%Sampled at 25 Hz
subplot(3, 1, 3), stem(t2, xt2); legend ('2-fold down-sampled');
xlabel ('time, sec.'), ylabel('x(2n)'); grid; title ('x(nT) at T = 0.04 sec.')
```

![Graphs of x(t), x(n), and x(2n)](image-url)
Note that $X(s) = \mathcal{L}(e^{-5t}u(t)) = 1/(s + 5)$. Shown below is the MATLAB plot of the magnitude spectrum $|X(j\Omega)|$ of the continuous-time signal $x(t)$ using the function `plot`. Omega is a vector, consequently we use “/.” instead of “/” etc. The main point to be made here is that $X(j\Omega)$ extends asymptotically to $\infty$, so, strictly speaking, $x(t)$ is not band-limited. Consequently, the spectrum $X(\omega)$ of the sampled signal $x(n)$ (shown later below) has some built-in aliasing.

$$
t = 0 : 1/512: 1; \text{xt = exp (-5*xt); } \%x(t) evaluated at 512 points
\text{subplot}(3, 1, 1), \text{plot}(t, \text{xt}); \text{legend ('x(t) = exp(-5t)' )};
\text{xlabel ('time'), ylable ('x(t)'); grid; title ('x(t) – Continuous-time')}
\%
\text{Omega = -6*pi: pi/256: 6*pi;} \text{X = 1./(5+j.*Omega);}
\text{subplot(3, 1, 2), plot(Omega, abs(X), 'k'); legend ('Spectrum of x(t)');}
\text{xlabel ('Omega, rad/sec'), ylable ('|X(Omega)|'); grid; title ('Magnitude')}
\%
\text{subplot(3, 1, 3), plot(Omega, angle(X), 'k'); legend ('Spectrum of x(t)');}
\text{xlabel ('Omega, rad/sec'), ylable ('Phase of X(Omega)'); grid; title ('Phase')}

Coming to the discrete-time signal, the spectrum of $x(n) = a^n u(n) = (0.905)^n u(n)$ is its DTFT

$$
X(e^{j\omega}) = \sum_{n=0}^{\infty} a^n e^{-j\omega n} = \frac{1}{1 - ae^{-j\omega}} = \frac{1}{1 - 0.905e^{-j\omega}}
$$

The MATLAB segment is

$$
t1 = 0 : 0.02: 1; \text{xn = exp (-5*t1); } \%Sampled at 50 Hz.
\text{subplot(3, 1, 1), stem(t1, xn); legend ('x(n) at 50 Hz');}
\text{xlabel ('time, sec.'), ylable ('x(n)'); grid; title ('x(nT) at T = 0.02 sec')}
\%$$
b = [1]; %Numerator coefficient
a = [1, -0.905]; %Denominator coefficients
w = -6*pi: pi/256: 6*pi; [Xw] = freqz(b, a, w);

Example 7.2.2 Given \( x(n) = e^{-n/2}u(n) \), find (a) \( x(5n/3) \), (b) \( x(2n) \), (c) \( x(n/2) \).

**Answer** The sequence

\[
x(n) = e^{-n/2}u(n) = (e^{-1/2})^n u(n) = a^n u(n)
\]

where \( a = e^{-1/2} = 0.606 \), is sketched below:
(a) With $y(n) = x(5n/3)$, we evaluate $y(n)$ for several values of $n$ (we have assumed here that $x(n)$ is zero if $n$ is not an integer):

$$
y(0) = x(5 \cdot 0/3) = x(0) = e^{-0/2} = 1$$
$$y(1) = x(5 \cdot 1/3) = x(5/3) = 0$$
$$y(2) = x(5 \cdot 2/3) = x(10/3) = 0$$
$$y(3) = x(5 \cdot 3/3) = x(5) = e^{-5/2} = a^5$$
$$\ldots$$
$$y(6) = x(5 \cdot 6/3) = x(10) = e^{-10/2} = a^{10}$$
$$\ldots$$

The general expression for $y(n)$ can be written as

$$y(n) = x(5n/3) = e^{-(5n/3)^2}, \quad n \text{ as specified below}$$

$$y(n) = \begin{cases} 
e^{-5n/6}, & n = 0, 3, 6, \ldots \\
0, & \text{otherwise} 
\end{cases}$$

The sequence is sketched below:

(b) With $y(n) = x(2n)$, we evaluate $y(.)$ for several values of $n$:

$$y(0) = x(2 \cdot 0) = x(0) = 1$$
$$y(1) = x(2 \cdot 1) = x(2) = a^2$$
$$y(2) = x(2 \cdot 2) = x(4) = a^4$$
$$y(3) = x(2 \cdot 3) = x(6) = a^6$$
$$\ldots$$

The general expression for $y(n)$ can be written as

$$y(n) = x(2n) = e^{-(2n)^2}, \quad n \text{ as specified below}$$

$$y(n) = \begin{cases} e^{-n}, & n \geq 0 \\
0, & \text{otherwise} 
\end{cases}$$
The sequence $y(.)$ is made up of every other sample of $x(.)$. This is **down-sampling** or **decimation** by a factor of 2 (or, compression in time). Note that some of the original sample values have disappeared. The sequence is sketched below.

(c) With $y(n) = x(n/2) = e^{-n/4}u(n)$, we evaluate $y(.)$ for several values of $n$ (again, we have assumed here that $x(n)$ is zero if $n$ is not an integer):

- $y(0) = x(0/2) = x(0) = 1$
- $y(1) = x(1/2) = x(0.5) = 0$
- $y(2) = x(2/2) = x(1) = a$
- $y(3) = x(3/2) = x(1.5) = 0$

The general expression for $y(n)$ can be written as

$$y(n) = x(n/2) = e^{-n/4}, \quad n \text{ as specified below}$$

$$\begin{cases} 
    e^{-n/4}, & n = 0, 2, 4, \ldots \\
    0, & \text{otherwise}
\end{cases}$$

The sequence $y(.)$ is constructed by inserting one zero between successive samples of $x(.)$. This is **up-sampling** or **interpolation** by a factor of 2 (or expansion in time). The sequence is sketched below:
To get back to the problem raised earlier, given the sequence \( x(n) \) obtained from \( x(t) \) at a rate \((1/T)\)

\[
x(t) \rightarrow x(nT) \rightarrow x(n), \text{ rate } (1/T)
\]

we want to obtain the sequence \( x'(n) \) which corresponds to a sampling rate \((1/T')\) where \( T \neq T'\)

\[
x(t) \rightarrow x(nT') \rightarrow x'(n), \text{ rate } (1/ T')
\]

There are two approaches to do this:

1. Convert \( x(n) \) to \( x(t) \) and resample at \((1/T')\) to generate \( x'(n) \). This is not ideal because of the imperfections in the \( A/D-H(z)-D/A \) originally involved in generating \( x(n) \). Or,
2. Change the sampling rate entirely with discrete-time operations.

**Example 7.2.3** Consider the 4 Hz signal \( x(t) = \cos 2\pi 4t \) which is obviously band-limited to \( F_{max} = 4 \) Hz. It is sufficient to sample it at 8 Hz. Suppose that it has been *over-sampled* by, say, a factor of 6 at \( F_s = 48 \) Hz to give \( x(n) = \cos 2\pi 4n(1/48) = \cos (\pi n/6) \).

If it is desired subsequently to generate from \( x(n) \) another signal \( x_1(n) \) that is a discrete-time version of \( x(t) \) sampled at \( F_{s1} = 16 \) Hz (sampling rate reduced or down-sampled by a factor of 3), then can we do this by dropping two samples of \( x(n) \) for every sample that we keep? In this specific example this is possible since a sampling rate of 16 Hz is clearly greater than \( 2F_{max} \) of 8 Hz. Thus the down-sampled version is obtained by replacing \( n \) in \( x(n) \) by \( 3n \)

\[
x_1(n) = x(3n) = \cos (\pi 3n/6) = \cos (\pi n/2) \quad \rightarrow (1)
\]

Let us compare this with what we would get if we were to sample \( x(t) = \cos 2\pi 4t \) directly at 16 Hz. We simply replace \( t \) by \( nT = n(1/16) \)

\[
x_1(n) = \cos 2\pi 4n(1/16) = \cos (\pi n/2) \quad \rightarrow (2)
\]

The results in (1) and (2) are the same. (QED)

We show below three plots: (1) The continuous-time signal \( x(t) \), (2) The sampled (at 48 Hz) version \( x(n) \), and (3) \( x(3n) \), the 3-fold down-sampled version of \( x(n) \); this is equivalent to sampling \( x(t) \) at 16 Hz.

```matlab
    t = 0 : 1/128: 0.5; xt = cos (2*pi*4*t); %x(t) evaluated at 128 points
    subplot(3, 1, 1), plot(t, xt); legend ('4-Hz Cosine');
    xlabel ('time, sec.'), ylabel('x(t)'); grid; title ('x(t) – Continuous-time')
    
    t1 = 0 : 1/48: 0.5; xn = cos (2*pi*4*t1); %Sampled at 48 Hz
    subplot(3, 1, 2), stem(t1, xn); legend ('x(n) at T = 1/48 = 0.020 sec.');
    xlabel ('time, sec.'), ylabel('x(n)'); grid; title ('x(nT) at T = 1/48 = 0.020 sec.')
    
    t3 = 0 : 1/16: 0.5; x1n = cos (2*pi*4*t3); %Sampled at 16 Hz
    subplot(3, 1, 3), stem(t3, x1n); xlabel ('time, sec.'), ylabel('x(3n)');
    grid; title ('3-fold down-sampled, x(nT) at T = 1/16 = 0.0625 sec.')
```
Alternatively, assuming $x(t)$ is not available, $x_1(n)$ could be obtained as follows:

1. Recover $x(t)$ by passing $x(n)$ through a DAC
2. Sample the resulting $x(t)$ at $F_{s1} = 16$ Hz

We take it, however, that this option is not desirable.

The above analysis assumes that we know the frequency content of the base band signal, $x(t)$. Generally this is not the case. Given the sequence $x(n)$ that was obtained by sampling at a rate, say $F_s$, we do not know what is the maximum frequency, $F_{\text{max}}$, contained in the underlying analog signal, $x(t)$. Assuming it was originally band-limited and properly sampled, it is safest to assume that the base band signal was band-limited to $F_s/2$ (or $F_{\text{max}}$) and not lower. In such a case simply dropping one or more samples of $x(n)$ for every sample we keep will not work. If we want to reduce the sampling rate by a factor of, say, $K$, then we would have to band-limit the precursor of $x(n)$ to $(F_s/2)/K = (F_{\text{max}}/K)$ and then sample it at the $K$-fold reduced sampling rate to achieve the required decimation. This amounts to down sampling $x(n)$ by a factor of $K$. (If the signal $x(t)$ originally actually contained a maximum frequency of $F_s/2$ then subsequent down sampling will result in unavoidable loss of information. But if it was band limited to significantly less than $F_s/2$ then down-sampling without loss of information is possible.)

The band-limiting mentioned above may be done either in the continuous-time domain or in the discrete-time domain. The procedure in the continuous time domain is as follows: Imagine that $x(t)$ is recovered from $x(n)$; $x(t)$ is then band-limited to $F_s/2K$ by passing it through an ideal low pass filter described by

$$H(F) = \begin{cases} 
1, & 0 \leq F < F_s/2K \\
0, & F_s/2K \leq F \leq F_s/2
\end{cases}$$

Dr. Ravi Billa
The band-limited signal, denoted \( x_1(t) \) is then sampled at the reduced rate of \( F_s/K \) to generate \( x_1(n) \). This method is generally undesirable because of all the imperfections inherent in originally generating \( x(n) \) from \( x(t) \) at a sampling rate of \( F_s \), converting \( x(n) \) back into \( x(t) \), then band-limiting \( x(t) \) to \( F_s/2 \) to generate \( x_1(t) \) and then sampling \( x_1(t) \) at a sampling rate of \( F_s/K \) to generate \( x_1(n) \).

**Sampling rate decimation** Reducing the sampling rate by an integer factor in the discrete-time domain is shown in the following block diagram. The down arrow in \( \downarrow K \) indicates down sampling by a factor of \( K \). The filter \( H(z) \) is a digital anti-aliasing filter whose output \( v(n) \) is a low pass filtered version of \( x(n) \).

If the filter \( H(z) \) is implemented as a linear phase FIR filter with \((M+1)\) coefficients specified as \( \{b_r, r = 0 \text{ to } M\} \), (some call it “\( M^{th} \) order”), then

\[
v(n) = \sum_{r=0}^{M} b_r x(n-r)
\]

We desire the output \( y(n) \) to be a down-sampled version of \( x(n) \), that is

\[
y(n) = v(Kn) = \sum_{r=0}^{M} b_r x(Kn-r)
\]

**Example 7.2.4** Consider the 4 Hz signal \( x(t) = \cos 2\pi 4t \) which is obviously band-limited to \( F_{\text{max}} = 4 \) Hz. It is sufficient to sample it at 8 Hz. Suppose instead that it has been over sampled, say, by a factor of 6 at \( F_s = 48 \) Hz to give \( x(n) = \cos 2\pi 4n(1/48) = \cos (\pi n/6) \).

Can we generate from \( x(n) \) another signal \( x_3(n) \) that is a discrete-time version of \( x(t) \) sampled at \( F_{s3} = F_s/8 = 6 \) Hz? This is down sampling by a factor of 8. We simply replace \( t \) by \( nT = n(1/6) \) to get

\[
x_3(n) = \cos 2\pi 4n(1/6) = \cos (8\pi n/6) = x(8n) = \{x(0), x(8), x(16), \ldots\}
\]

In other words, \( x_3(n) \) is made up of every 8th sample of \( x(n) \). For every sample value of \( x(n) \) we keep we discard the next 7 samples. We know, however, that a sampling frequency of 6 Hz does not satisfy the sampling theorem; in this case down sampling has been taken too far.

We show below three plots: (1) The sampled (at 48 Hz) version \( x(n) \) – this is repeated from above, (2) \( x(2n) \), the 2-fold down-sampled version of \( x(n) \); this is equivalent to sampling \( x(t) \) at 24 Hz, and (3) \( x(8n) \), the 8-fold down-sampled version of \( x(n) \); this is equivalent to sampling \( x(t) \) at the unacceptably low rate of 6 Hz.

```matlab
t1 = 0 : 1/48: 0.5; xn = cos (2*pi*4*t1); %Sampled at 48 Hz
subplot(3, 1, 1), stem(t1, xn); legend ('x(n) at 48 Hz'); xlabel ('time, sec.'), ylabel('x(n)'); grid; title ('x(nT) at T = 1/48')
%
t2 = 0 : 1/24: 0.5; xt2 = cos (2*pi*4*t2); %Sampled at 24 Hz
```
Example 7.2.5 To show visually a case of down sampling that is not satisfactory, consider $x_4(n)$ generated from $x(n)$ by down sampling by a factor of 12, i.e., $x_4(n) = x(12n)$. This is also obtained by sampling at $48/12 = 4$ Hz:

$$x_4(n) = x(nT) = \cos 2\pi 4n(1/4) = \cos (2\pi n) = \cos (12\pi n/6) = x(12n)$$

In this case $\cos (2\pi n) = 1$ for all $n$, so that

$$x_4(n) = 1$$

which has no resemblance to $x(n)$, making it visually obvious that down sampling has been taken too far. Depending on at what point in the cycle the samples are taken, $x_4(n)$ equals a constant (including 0), for all $n$.

7.3 Down-sampling
Assume that $x(n)$ is obtained from an underlying continuous-time signal $x(t)$ by sampling at $F_x$ Hz. Assume that $x(t)$ was originally band limited to $F_x/2$ Hz. On the digital frequency ($\omega$) scale this amounts to $x(n)$ being band limited to $\pi$.

We now wish to generate a signal $y(n)$ by down-sampling $x(n)$ by a factor of $M$, that is, we are reducing the sampling rate by a factor of $M$. This amounts to:

1. Converting $x(n)$ to $x(t)$ using a D/A converter.
2. Band limiting $x(t)$ to $F_x/2M$ Hz. Assume that no information is lost due to this band limiting.
3. Resampling $x(t)$ at $F_x/M$ Hz. to produce $y(n)$.

Equivalently the above task is accomplished entirely in the digital domain by

1. Band limiting $x(n)$ to $\pi/M$. Assume that no information is lost due to this step.
2. Down-sampling the above $x(n)$ by a factor of $M$ to produce $y(n)$.

We may view $y(n)$ as though it were generated by sampling an underlying analog signal $y(t)$ at a rate $F_y = F_x/M$ Hz.

Given the signal $x(n)$ that was obtained at a certain sampling rate the new signal $y(n)$, the down-sampled version of $x(n)$, with a sampling rate that is $(1/M)$ of that of $x(n)$, obtained from $x(n)$, is given by:

$$y(n) = x(Mn)$$

and is made up of every $M^{th}$ sample value of $x(n)$; the intervening $(M-1)$ sample values of $x(n)$ are dropped. This amounts to

$$y(0) = x(0), y(1) = x(M), y(2) = x(2M), y(3) = x(3M), \ldots$$

The time between samples of $y(.)$ is $M$ times that between samples of $x(.)$, or the sampling frequency of $y(.)$ is reduced by a factor of $M$ from that of $x(.)$. The block diagram of a down sampler is shown below.

![Down sampler block diagram](image-url)

**Example 7.3.1** As an example, if $x(n) = a^n u(n)$, $a < 1$, is the sequence:

$$
n = 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \\
x(n) = \{1 \quad a \quad a^2 \quad a^3 \quad a^4 \quad a^5 \quad a^6 \quad a^7 \quad a^8 \quad \ldots \}$$

then $y(n) = x(2n)$, with $M = 2$, is its 2-fold down-sampled version and is obtained by keeping every other sample of $x(n)$ and dropping the samples in between:

$$
n = 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \\
y(n) = \{1 \quad a^2 \quad a^4 \quad a^6 \quad a^8 \quad a^{10} \quad \ldots \}$$
In this example it is understood that the time between samples of $y(n)$ is twice that between samples of $x(n)$, or, the sampling rate of $y(n)$ is one-half of that of $x(n)$.

**Example 7.3.2** Test the system $y(n) = x(Mn)$, where $M$ is a constant, for time-invariance.

\[ y(n) = T[x(n)] = x(Mn) \]

**Solution** See also Unit I. For the input $x(n)$ the output is

\[ y(n) = T[x(n)] = x(Mn) \]

Delay this output by $n_0$ to get

\[ y(n-n_0) = x(M(n-n_0)) = x(Mn-Mn_0) \rightarrow (A) \]

Next, for the delayed input $x(n-n_0)$ the output is

\[ y(n, n_0) = T[x(n-n_0)] = x(Mn) = x(Mn-Mn_0) \rightarrow (B) \]

We see that $(A) \neq (B)$, that is, $y(n-n_0)$ and $y(n, n_0)$ are not equal. Delaying the input is not equivalent to delaying the output. So the system is not time-invariant. In other words the down-sampling operation is a time-varying system.

**Spectrum of a down-sampled signal** Given the signal $x(n)$ whose spectrum is $X(\omega)$ or $X(e^{j\omega})$ we want to find the spectrum of $y(n)$, the down-sampled version of $x(n)$, denoted by $y(n) \leftrightarrow Y(\omega)$.

Consider the periodic train of impulses, $p(n)$, with period $M$

\[
p(n) = \begin{cases} 
1, & n = 0, \pm M, \pm 2M, \ldots \\
0, & \text{otherwise}
\end{cases}
\]

The discrete Fourier series representation (see Example 1 in Unit II) of $p(n)$ is

\[ p(n) = \sum_{k=0}^{M-1} P_k e^{j2\pi k n/M}, \quad 0 \leq n \leq M-1 \]

The Fourier coefficients are given by

\[ P_k = \frac{1}{M} \sum_{n=0}^{M-1} p(n) e^{-j2\pi k n/M} = \frac{1}{M}, \quad 0 \leq k \leq M-1 \]
Thus the DFS for \( p(n) \) is
\[
p(n) = \frac{1}{M} \sum_{k=0}^{M-1} e^{j2\pi kn/M}, \quad 0 \leq n \leq M-1
\]

Define the signal \( x'(n) \)
\[
x'(n) = x(n) p(n) = \begin{cases} x(n), & n = 0, \pm M, \pm 2M, \ldots \\ 0, & \text{otherwise} \end{cases}
\]
The sequence \( x'(n) \) consists of values of \( x(n) \) whenever \( n = 0, \pm M, \pm 2M, \ldots \), and zeros in between those points.

Define the down-sampled version \( y(n) \)
\[
y(n) = x'(Mn) = x(Mn)
\]
The signal \( y(n) \) consists of values of \( x(Mn) \) at \( n = 0, \pm 1, \pm 2, \ldots \), but no zeros in between.

With \( y(n) = x'(Mn) = x(Mn) \) our objective is to find the spectrum \( Y(\omega) \). Keep in mind that \( X(\omega) \) periodic in \( \omega \) since \( x(n) \) is a discrete-time sequence; and the same is true of \( Y(\omega) \). Now the \( z \)-transform of \( y(n) \) is
\[
Y(z) = \sum_{n=-\infty}^{\infty} y(n)z^{-n} = \sum_{n=-\infty}^{\infty} x'(Mn)z^{-n}
\]
Set \( Mn = k \): then \( n = k/M \) and the summation limits \( n = \{ -\infty \text{ to } \infty \} \) become \( k = \{-\infty \text{ to } \infty \} \). Thus
\[
Y(z) = \sum_{k=-\infty}^{\infty} x'(k)z^{-k/M} = \sum_{n=-\infty}^{\infty} x'(n)z^{-n/M}
\]
Here \( x'(n) = 0 \) except when \( n \) is a multiple of \( M \). Substituting \( x(n) p(n) \) for \( x'(n) \) in the above equation,
\[
Y(z) = \sum_{n=-\infty}^{\infty} x(n) p(n)z^{-n/M}
\]
Substituting \( \frac{1}{M} \sum_{k=0}^{M-1} e^{j2\pi kn/M} \) for \( p(n) \) (from the DFS) in the above equation,
\[
Y(z) = \sum_{n=-\infty}^{\infty} x(n) \left[ \frac{1}{M} \sum_{k=0}^{M-1} e^{j2\pi kn/M} \right] z^{-n/M} = \frac{1}{M} \sum_{k=0}^{M-1} \sum_{n=-\infty}^{\infty} x(n) e^{j2\pi kn/M} z^{-n/M}
\]
\[
= \frac{1}{M} \sum_{k=0}^{M-1} \sum_{n=-\infty}^{\infty} x(n) \left( e^{-j2\pi k/M} z^{1/M} \right)^n
\]
\[
= X \left( e^{-j2\pi k/M} z^{1/M} \right)
\]
\[
= \frac{1}{M} \sum_{k=0}^{M-1} X\left( e^{-j2\pi k/M} z^{1/M} \right)
\]

Substituting \( z = e^{j\omega} \) gives us the DTFT, \( Y(\omega) \) or \( Y(e^{j\omega}) \),

\[
Y(\omega) = Y(z)|_{z = e^{j\omega}} = \frac{1}{M} \sum_{k=0}^{M-1} X\left( e^{-j2\pi k/M} e^{j\omega/M} \right) = \frac{1}{M} \sum_{k=0}^{M-1} X\left( e^{j(\omega-2\pi k)/M} \right)
\]

\[
= \frac{1}{M} \sum_{k=0}^{M-1} X\left( \frac{\omega - 2\pi k}{M} \right)
\]
MATLAB. To demonstrate the stretching and shifting of $X(\omega)$ to $X((\omega-2\pi k)/M)$ for $k = 1$ and $M = 2$, that is, $X((\omega-2\pi)/2)$. This is done in 3 steps: (1) $X(\omega)$, (2) $X(\omega/2)$, and (3) $X((\omega-2\pi)/2)$.

```matlab
w = -2*pi: pi/256: 2*pi;
subplot(3, 1, 1), plot(w, cos(w));
xlabel ('\omega, rad/sample'), ylabel('X(\omega)'); grid; title ('X(\omega)')

subplot(3, 1, 2), plot(w, cos(w/2));
xlabel ('\omega, rad/sample'), ylabel('X(\omega /2)'); grid; title ('Stretched by factor 2: X(\omega/2)')

subplot(3, 1, 3), plot(w, cos((w-2*pi)/2));
xlabel ('\omega, rad/sample'), ylabel('X((\omega – 2\pi)/2)'); grid; title ('And shifted by 2\pi: X((\omega – 2\pi)/2)')
```

where, for simplicity, we have used the notation $X(\omega)$ instead of $X(e^{j\omega})$. This expression for $Y(e^{j\omega})$ is a sum of $M$ terms. Note that the function $X(\omega-2\pi k)$ is a shifted (by $2\pi k$) version of $X(\omega)$ and $X(\omega/M)$ is a stretched (by a factor $M$) version of $X(\omega)$. Thus $Y(e^{j\omega})$ is the sum of $M$ uniformly shifted and stretched versions of $X(e^{j\omega})$ each scaled by the factor $(1/M)$. The shifting
in multiples of $2\pi$ corresponds to the factor $(\omega - 2\pi k)$ in the argument of $X(\cdot)$, and the stretching by the factor $M$ corresponds to the $M$ in $(\omega - 2\pi k)/M$. Note that the amount of shift is also affected by the factor $M$, that is, the amount of shift doesn’t stay at $2\pi k$ but ends up being $2\pi k/M$.

The expression for $Y(e^{j\omega})$ contains a total of $M$ versions of $X(e^{j\omega})$, one original and $(M-1)$ shifted replicas. Each of these is also stretched by a factor of $M$, so $X(e^{j\omega})$ should have been preshrunk, that is, band limited, to $\pi/M$ before undertaking the down-sampling. Writing out the expression for $Y(e^{j\omega})$ in full, we have

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

$$= \frac{1}{M} \left[ X\left(\frac{\omega}{M}\right) + X\left(\frac{\omega - 2\pi}{M}\right) + \ldots + X\left(\frac{\omega - 2\pi(M-1)}{M}\right) \right]$$

The first term that makes up $Y(e^{j\omega})$, that is, $\frac{1}{M} X\left(\frac{\omega}{M}\right)$, is shown in the figure below. The figure implicitly uses $M = 2$. In general there will be $(M-1)$ shifted replicas of this term.

In particular, for $M = 2$, we have

$$Y(e^{j\omega}) = \frac{1}{2} \sum_{k=0}^{2-1} X\left(\frac{\omega - 2\pi k}{2}\right) = \frac{1}{2} \left[ X\left(\frac{\omega}{2}\right) + X\left(\frac{\omega - 2\pi}{2}\right) \right]$$

$$= \frac{1}{2} \left[ X\left(\frac{\omega}{2}\right) + X\left(\frac{\omega}{2} - \pi\right) \right]$$

This is also written in the form

$$Y(e^{j\omega}) = \frac{1}{2} \sum_{k=0}^{2-1} X\left(e^{j(\omega - 2\pi k)/2}\right) = \frac{1}{2} \left[ X\left(e^{j\omega/2}\right) + X\left(e^{j(\omega-2\pi)/2}\right) \right]$$

DSP-7 (Multirate) 18 of 58 Dr. Ravi Billa
To recapitulate, before we decided to down sample \( X(\omega) \) was originally band limited to \( \pi \) on the digital frequency scale (that is, \( F_x/2 \) Hz). We then band limited it to \( \pi/M \) (that is, \( F_x/2M \) Hz) and down sampled by a factor of \( M \).

**Aliasing** Down-sampling by a factor of \( M \), in itself, is simply retaining every \( M^{th} \) sample while dropping all samples in between. If, therefore, prior to down-sampling, the signal \( x(n) \) is indeed band-limited to \( \pi/M \) then we generate the down-sampled version \( y(n) \) by simply taking every \( M^{th} \) sample of \( x(n) \). This process is shown below in block diagram fashion. If in this set-up \( x(n) \) is not band-limited as required then the spectrum of \( y(n) \) will contain overlapping spectral components of \( x(n) \) due to stretching, i.e., \( X(\omega/M) \) will overlap \( X(\omega-2\pi/M) \), etc. This results in **aliasing**.

Band-limiting \( x(n) \) to \( \pi/M \) (if not done already) is done by an **anti-aliasing filter** (digital low pass filter) with a cut-off frequency of \( \pi/M \). The general process of **decimation then consists of filtering followed by down sampling** shown in block diagram below.

Unlike an analog anti-aliasing filter associated with an ADC, the filter in this diagram is a **digital anti-aliasing filter** specified as

\[
H(\omega) = \begin{cases} 
1, & 0 \leq |\omega| < \pi/M \\
0, & \pi/M \leq |\omega| \leq \pi
\end{cases}
\]

Note that \( \pi \) corresponds to \( F_x/2 \) and \( \pi/M \) corresponds to \( F_x/2M \) where \( F_x \) is the sampling frequency of \( x(n) \).

Typically, in order to avoid (delay) distortion, the filter \( H(z) \) is a linear phase FIR filter with \((N+1)\) coefficients \({h(r), r = 0 \text{ to } N}\). The output, \( v(n) \), of the low pass filter is then given by convolution

\[
v(n) = \sum_{r=0}^{N} h(r) x(n-r)
\]

and the decimated signal is
\[ y(n) = v(nM) = \sum_{r=0}^{N} h(r) x(nM - r) \]

In summary, in order to down sample a signal by a factor of \( M \):
- The signal should have been originally over-sampled by a factor of \( M \) (that is originally band limited to \( \pi/M \) and over-sampled). In this case the signal is down-sampled straightaway; no pre-filter is needed. OR
- The signal, assumed originally band limited to \( \pi \), should be band-limited to \( \pi/M \) by a pre-filter; the signal is then down-sampled. In this case there will be some loss of information.

**Example 7.3.3** Consider the signal \( x(n) = a^n u(n), \quad a < 1 \).

a) Determine the spectrum \( X(\omega) \)

b) If \( x(n) \) is applied to a decimator that reduces the sampling rate by a factor of 2 determine the output spectrum

c) Show that the spectrum in part (b) is simply the Fourier transform of \( x(2n) \)

d) Plot the spectra of \( x(n) \) and \( x(2n) \) for \( a = 0.905 \)

**Solution** [See also Unit I]

a) The spectrum of \( x(n) \) is given by its DTFT

\[
X(\omega) = \sum_{n=-\infty}^{\infty} x(n) e^{-j\omega n} = \sum_{n=0}^{\infty} a^n e^{-j\omega n} = \sum_{n=0}^{\infty} (ae^{-j\omega})^n
\]

\[
= \frac{1}{1-ae^{-j\omega}}, \quad |ae^{-j\omega}| < 1
\]

This spectrum is not band-limited but we may pretend it is. This may also be obtained as \( X(\omega) = X(z)|_{z=e^{j\omega}} \).

b) The spectrum of \( y(n) = x(2n) \) is given by

\[
Y(\omega) = \frac{1}{M} \sum_{k=0}^{M-1} X\left(\frac{\omega - 2\pi k}{M}\right)
\]

which, with \( M = 2 \), becomes

\[
Y(\omega) = \frac{1}{2} \sum_{k=0}^{1} X\left(\frac{\omega - 2\pi k}{2}\right) = \frac{1}{2} \left[ X\left(\frac{\omega}{2}\right) + X\left(\frac{\omega}{2} - \pi\right) \right]
\]

\[
= \frac{1}{2} \left[ \frac{1}{1-ae^{-j\omega/2}} + \frac{1}{1-ae^{-j(\omega/2)-\pi}} \right] = \frac{1}{2} \left[ \frac{1}{1-ae^{-j\omega/2}} + \frac{1}{1-ae^{-j(\omega/2)}} \right]
\]

\[
= \frac{1}{2} \left[ \frac{1}{1-ae^{-j\omega/2}} + \frac{1}{1+ae^{-j\omega/2}} \right] = \frac{1}{1-a^2 e^{-j\omega}}
\]

C) The Fourier transform of \( y(n) = x(2n) = a^{2n} u(2n) = a^{2n} u(n) \) is

\[
Y(\omega) = \sum_{n=0}^{\infty} a^{2n} e^{-j\omega n} = \sum_{n=0}^{\infty} (a^2 e^{-j\omega})^n = \frac{1}{1-a^2 e^{-j\omega}}, \quad |a^2 e^{-j\omega}| < 1
\]

d) The spectra.

\[
b = [1]; \quad \text{%Numerator coefficient}
\]

\[
a_1 = [1, -0.905]; \quad a_2 = [1, -0.819]; \quad \text{%Denominator coefficients}
\]

\[
w = \text{pi/256}; \quad \text{pi; \%A total of 512 points}
\]

\[
[X1w] = \text{freqz(b, a1, w)}; \quad [X2w] = \text{freqz(b, a2, w)}
\]

\[
\text{subplot}(2, 1, 1), \quad \text{plot}(w, \text{abs(X1w)})\); \quad \text{legend ('Spectrum of x(n)'},
\]

Dr. Ravi Billa

DSP-7 (Multirate) 20 of 58 www.jntuworld.com
xlabel('Frequency \omega, rad/sample'), ylabel('Magnitude of X1(\omega)'); grid

subplot(2, 1, 2), plot(w, abs(X2w)); legend ('Spectrum of x(2n)');
xlabel('Frequency \omega, rad/sample'), ylabel('Magnitude of X2(\omega)'); grid
7.4 Up-sampling

Assume that $x(n)$ is obtained from the continuous-time signal $x(t)$ by sampling at $F_x$ Hz. We now wish to generate a signal $y(n)$ by up-sampling $x(n)$ by a factor of $L$, that is, we are increasing the sampling rate by a factor of $L$. This amounts to

1. Converting $x(n)$ to $x(t)$ using a D/A converter.
2. Resampling $x(t)$ at $LF_x$ Hz to produce $y(n)$.

We may view $y(n)$ as though it were generated by sampling an underlying analog signal $y(t)$ (or $x(t)$ for that matter) at a rate $F_y = LF_x$ Hz. As in the case of down-sampling we prefer to do this entirely in the digital domain.

Given the signal $x(n)$ that was obtained at a certain sampling rate we can obtain a new signal $y(n)$ from $x(n)$ with a sampling rate that is $L$ times that of $x(n)$. The signal $y(n)$, an up-sampled version of $x(n)$, is given by:

$$y(n) = \begin{cases} x(n/L), & n = 0, \pm L, \pm 2L, \ldots \\ 0, & \text{otherwise} \end{cases}$$

and is constructed by placing $(L-1)$ zeros between every pair of consecutive samples of $x(n)$. The time between samples of $y(n)$ is $(1/L)$ of that between samples of $x(n)$, or the sampling frequency of $y(n)$ is increased by a factor of $L$ from that of $x(n)$. The block diagram of an up-sampler is shown below.

### Example 7.4.1
As an example, if $x(n) = a^n u(n), a < 1$, is the sequence:

$$x(n) = \{1, a, a^2, a^3, a^4, a^5, a^6, a^7, a^8, \ldots \}$$

then $y(n) = x(n/2)$, with $L = 2$, is its 2-fold up-sampled version and is obtained by inserting a 0 between each pair of consecutive values in $x(n)$

$$y(n) = \{1, 0, a, 0, a^2, 0, a^3, 0, a^4, 0, a^5, \ldots \}$$

In this example it is understood that the time between samples of $y(n)$ is one half of that between samples of $x(n)$, or, the sampling rate of $y(n)$ is twice that of $x(n)$.

### Example 7.4.2
Test the system $y(n) = x(n/L)$, where $L$ is a constant, for time-invariance.
**Solution** See also Unit I. The system \( y(n) = x(n/L) \) in itself only partially defines an up-sampler. But the following goes to show that the up-sampling operation is a time-varying system. For the input \( x(n) \) the output is

\[
y(n) = T[x(n)] = x(n/L)
\]

Delay this output by \( n_0 \) to get

\[
y(n-n_0) = x((n-n_0)/L) = x((n/L)-(n_0/L)) \to (A)
\]

Next, for the delayed input \( x(n-n_0) \) the output is

\[
y(n, n_0) = T[x(n-n_0)] = x(n/L) = x((n/L)-n_0) \to (B)
\]

We see that \((A) \neq (B)\), that is, \( y(n-n_0) \) and \( y(n, n_0) \) are not equal. Delaying the input is not equivalent to delaying the output. So the system \( y(n) = x(n/L) \) is not time-invariant. Therefore the up sampler defined by

\[
y(n) = \begin{cases} x(n/L), & n = 0, \pm L, \pm 2L, \ldots \\ 0, & \text{otherwise} \end{cases}
\]

is not time-invariant; it is a time-varying system.

**Spectrum of an up-sampled signal** Given the signal \( x(n) \) whose spectrum is \( X(\omega) \) or \( X(e^{j\omega}) \) we want to find the spectrum of \( y(n) \), the up-sampled version of \( x(n) \), denoted by \( y(n) \leftrightarrow Y(\omega) \).

The signal \( y(n) \), with a sampling rate that is \( L \) times that of \( x(n) \), is given by:

\[
y(n) = \begin{cases} x(n/L), & n = 0, \pm L, \pm 2L, \ldots \\ 0, & \text{otherwise} \end{cases}
\]

We obtain the z-transform and from it the spectrum:

\[
Y(z) = \sum_{n=-\infty}^{\infty} y(n)z^{-n} = \sum_{n=0}^{\infty} x(n/L)z^{-n} + \sum_{n = \text{otherthan} kL, k = \text{all integers}}^{\infty} 0z^{-n}
\]

\[
= \sum_{n = 0, \pm L, \pm 2L, \ldots}^{\infty} x(n/L)z^{-n}
\]

Set \( n/L = k \): this leads to \( n = kL \), and the summation indices \( n = \{0, \pm L, \pm 2L, \pm 3L, \ldots\} \) become \( k = (-\infty \text{ to } \infty) \), so that

\[
Y(z) = \sum_{k = -\infty}^{\infty} x(k)z^{-kL} = \sum_{k = -\infty}^{\infty} x(k)(z^L)^k = X(z^L)
\]

Setting \( z = e^{j\omega} \) gives us the spectrum

\[
Y(e^{j\omega}) = Y(z) \bigg| z = e^{j\omega} = X(e^{jL\omega}) \quad \text{or} \quad Y(\omega) = X(\omega L)
\]

Thus \( Y(\omega) \) is an \( L \)-fold compressed version of \( X(\omega) \); the value of \( X(.) \) that occurred at \( \omega L \) occurs at \( \omega \), (that is, at \( \omega 0L/L \)) in the case of \( Y(.) \). In going from \( X \) to \( Y \) the frequency values are pushed in toward the origin by the factor \( L \). For example, the frequency \( \omega 0L \) is pushed to \( \omega 0L/L \), the frequency \( \pi \) is pushed to \( \pi/L \), \( 2\pi \) is pushed to \( 2\pi L \), etc.

Shown below are the spectra \( X(\omega) \) and \( Y(\omega) \) for 2-fold up-sampling, that is, \( L = 2 \). Note that \( X(\omega) \) is periodic to start with so that the frequency content of interest is in the base range \((-\pi \leq \omega \leq \pi)\) with replicas of this displaced by multiples of \( 2\pi \) from the origin on either side. Due to

DSP-7 (Multirate) 23 of 58 Dr. Ravi Billa
up-sampling the frequency content of \( X(\omega) \) in the range \((-\pi \leq \omega \leq \pi)\) is compressed into the range \((-\pi/L \leq \omega \leq \pi/L)\) of \( Y(\omega) \), that is, into \((-\pi/2 \leq \omega \leq \pi/2)\), centered at \( \omega = 0 \). The first replica of \( X(\omega) \) in the range \((\pi \leq \omega \leq 3\pi)\), centered at \( 2\pi \), is compressed to the range \((\pi/2 \leq \omega \leq 3\pi/2)\) of \( Y(\omega) \), centered at \( \pi \); its counterpart, in \((-3\pi \leq \omega \leq -\pi)\), centered at \(-2\pi\), is compressed to \((-3\pi/2 \leq \omega \leq -\pi/2)\), centered at \(-\pi\). If, for the purpose of discussion, we consider the range \((0, 2\pi)\) as one fundamental period then the replica in the range \((\pi/2, 3\pi/2)\) of \( Y(\omega) \) is an image (spectrum) and needs to be filtered out with a low pass filter (anti-imaging filter) of band-width \( \pi/2 \). With \( L = 2 \) this is the only image in \((0, 2\pi)\).

Furthermore, while the spectrum \( X(\omega) \) is periodic with a period = \( 2\pi \), the spectrum \( Y(\omega) \), on account of the image, is a 2-fold periodic repetition of the base spectrum in \((-\pi/2 \leq \omega \leq \pi/2)\); the image spectrum is actually spurious/unwanted; further the periodicity of \( Y(\omega) \) is still \( 2\pi \).

These observations can be extended to larger values of \( L \). For \( L = 3 \), for instance, there will be two image spectra (a 3-fold periodic repetition of the base spectrum in \((-\pi/3 \leq \omega \leq \pi/3)\), and the anti-imaging filter band width will be \( \pi/3 \).

In general, up-sampling of \( x(n) \) by a factor of \( L \) involves
- Inserting \( L-1 \) zeros between successive pairs of sample values of \( x(n) \).
- The spectrum \( Y(\omega) \) of the up-sampled signal is an \( L \)-fold compressed version of \( X(\omega) \). As a result \( Y(\omega) \) contains \( L-1 \) images and is an \( L \)-fold periodic repetition of the base spectrum in \((-\pi/L \leq \omega \leq \pi/L)\).
- The anti-imaging filter band width is \( \pi/L \).
The over-all scheme of up-sampling is shown in block diagram below. Unlike an analog anti-imaging filter associated with a DAC, the filter in this diagram is a digital anti-imaging filter.

![Block Diagram of Up-sampling](image)

In this diagram the pass band gain of the anti-imaging filter is shown as 1. This gain is actually chosen equal to \( L \) to compensate for the fact that the average value of \( y(n) \) is \( 1/L \) times the average value of \( x(n) \) due to the presence of the inserted zeros.

\[
H(\omega) = \begin{cases} 
L, & 0 \leq |\omega| < \pi/L \\
0, & \pi/L \leq |\omega| \leq \pi 
\end{cases}
\]

Note that \( \pi \) corresponds to \( F_x/2 \) and \( \pi/L \) corresponds to \( F_x/2L \) where \( F_x \) is the sampling frequency of \( x(n) \).

The output of the low pass filter is given by the convolution sum

\[
y(n) = \sum_{r=-\infty}^{\infty} h(n-r) v(r)
\]

where its input is

\[
v(r) = \begin{cases} 
x(r/L), & r = 0, \pm L, \pm 2L, \ldots \\
0, & \text{otherwise}
\end{cases}
\]

Now \( v(r) = 0 \) except at \( r = kL \), where \( k \) is all integers from \(-\infty\) to \( \infty \). Thus we have

\[
v(kL) = x(kL/L) = x(k)
\]

The convolution sum may be written as

\[
\sum_{r=-\infty}^{\infty} h(n-r) v(r) = \sum_{k=-\infty}^{\infty} h(n-kL) v(kL) = \sum_{k=-\infty}^{\infty} h(n-kL) x(k)
\]

so that the interpolated signal is

\[
y(n) = \sum_{k=-\infty}^{\infty} h(n-kL) x(k)
\]

**Illustration** Given the signal \( x(n) = \{1, a, a^2, a^3, a^4, a^5, a^6, a^7, a^8, a^9, a^{10}, \ldots\} \), its 2-fold up-sampled version is obtained by inserting a 0 between each pair of consecutive samples in \( x(n) \):

\[
y(n) = \{1, 0, a, 0, a^2, 0, a^3, 0, a^4, 0, a^5, 0, a^6, 0, a^7, 0, a^8, 0, a^9, 0, a^{10}, \ldots\}
\]
Intuitively, even visually, $y(n)$ contains higher (or, more) frequencies than $x(n)$ because of the inserted zeros. For instance, consider the first two or three samples in each sequence. In the case of $x(n)$ the changes from 1 to $a$ to $a^2$ are smoother than the fluctuations in $y(n)$ from 1 to 0 to $a$ to 0 to $a^2$; these latter fluctuations are the higher frequencies not originally contained in $x(n)$. It is these higher frequencies that are represented by the image in the spectrum of $y(n)$ prior to anti-imaging filtering. The anti-imaging filter removes or smoothes out the higher frequency fluctuations from the up-sampled version; this smoothing is manifested in the form of the interpolated zeros being replaced by nonzero values.

### 7.5 Cascading sampling rate converters

Given a discrete-time signal $x(n)$ we may want to convert its sampling rate by a non integer factor, in particular, by a rational number. For instance, we may be interested in $x(3n/2)$. This involves a 2-fold up-sampling and a 3-fold down-sampling for a net down sampling by a factor of 1.5 ($= 3/2$). The sequence $x(3n/5)$ involves a 5-fold up-sampling and a 3-fold down-sampling for a net up-sampling by a factor of 1.67 ($= 5/3$).

In general, in a cascade of an $M$-fold down-sampler and an $L$-fold up-sampler the positions of the two samplers are inter-changeable with no difference in the input-output behavior if and only if $M$ and $L$ are co-prime (relatively prime, that is, $M$ and $L$ do not have a common factor). The sequence $x(3n/2)$ may be generated by cascading the up-sampler and the down-sampler in either order, that is, down followed by up or vice versa. However, a cascade of a 6-fold down-sampler ($M = 6$) followed by a 4-fold up-sampler ($L = 4$) is not the same as a cascade of a 4-fold up-sampler followed by a 6-fold down-sampler even though in both cases $M/L = 6/4$. This is because $M$ and $L$ have a common factor, that is, the rational number $M/L$ is not in its reduced form. The ratio $M/L$ should be reduced to 3/2; then the 3-fold down-sampler and the 2-fold up-sampler are interchangeable in position.

**Example 7.5.1** Given $x(n) = e^{-n/2} u(n)$, find $x(5n/3)$.

**Answer** We borrow this from an earlier section. Our objective is to present the earlier solution and then reformulate it in the context of cascading up- and down-samplers. The sequence

$$x(n) = e^{-n/2} u(n) = (e^{-1/2})^n u(n) = (0.606)^n u(n) = a^n u(n)$$

where $a = e^{-1/2} = 0.606$, is sketched below:

![Diagram](image)

With $y(n) = x(5n/3)$, we evaluate $y(.)$ for several values of $n$ (we have assumed here that $x(n)$ is zero if $n$ is not an integer):
\[ y(0) = x(5 \cdot 0 / 3) = x(0) = e^{-0/2} = 1 \]
\[ y(1) = x(5 \cdot 1 / 3) = x(5 / 3) = 0 \]
\[ y(2) = x(5 \cdot 2 / 3) = x(10 / 3) = 0 \]
\[ y(3) = x(5 \cdot 3 / 3) = x(5) = e^{-5/2} = a^5 \]
\[ \ldots \]
\[ y(6) = x(5 \cdot 6 / 3) = x(10) = e^{-10/2} = a^{10} \]
\[ \ldots \]

The general expression for \( y(n) \) can be written as

\[
y(n) = x(5n/3) = e^{-5n/6}, \quad n \text{ as specified below}
\]

\[
\begin{cases} 
  e^{-5n/6}, & n = 0, 3, 6, \ldots \\
  0, & \text{otherwise}
\end{cases}
\]

\[ n = 0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \ 10 \ 11 \ 12 \ \ldots 
\]

\[ y(n) = \{1 \ 0 \ 0 \ a^2 \ 0 \ 0 \ a^5 \ 0 \ 0 \ a^{10} \ 0 \ 0 \ a^{15} \ \ldots \} \]

The sequence is sketched below:

We shall recast this problem in terms of cascading the up- and down-samplers. In the expression \( y(n) = x(5n/3) \) there is a 3-fold up-sampling and a 5-fold down-sampling. Since the numerator 5 is greater than the denominator 3 there is a net down-sampling by a factor of 1.67 (= 5/3). Let us first do a 3-fold up-sampling of \( x(n) \) followed by a 5-fold down-sampling of the resulting sequence. That is, given the sequence \( x(n) \)

\[
x(n) \rightarrow \uparrow 3 \rightarrow y_u(n) = x(n/3) \rightarrow \downarrow 5 \rightarrow y(n) = x(5n/3)
\]

\[ n = 0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \ 10 \ \ldots 
\]

\[ x(n) = \{1 \ a \ a^2 \ a^3 \ a^4 \ a^5 \ a^6 \ a^7 \ a^8 \ a^9 \ a^{10} \ \ldots \} \]
we define \( y_u(n) = x(n/3) \), and then \( y(n) = y_u(5n) = x(5n/3) \). The sequences \( y_u(n) \) and \( y(n) \) are given below.

\[
y_u(n) = x(n/3) = e^{-n/6} u(n/3) = \begin{cases} 
a^n/3 & n = 0, 3, 6, \\
0, & \text{otherwise}
\end{cases}
\]

\[
y(n) = y_u(5n) = x(5n/3) = e^{-5n/6} u(5n/3) = \begin{cases} 
a^{5n/3} & n = 0, 3, 6, \\
0, & \text{otherwise}
\end{cases}
\]

Alternatively, we may first do a 5-fold down sampling followed by a 3-fold up-sampling:

\[
x(n) \downarrow 5 \rightarrow y_u(n) = x(5n) \uparrow 3 \rightarrow y(n) = x(5n/3)
\]

\[
y_d(n) = x(5n) = \{1, a^5, a^{10}, a^{15}, a^{20}, \ldots \}
\]

\[
y(n) = y_u(5n/3) = x(5n/3) = \{1, 0, a^5, 0, 0, a^{10}, 0, 0, a^{15}, 0, 0, a^{20}, \ldots \}
\]

The net effect is that between the first two terms (1 and \( a^5 \)) of the final output \( y(.) \) we have dropped four original terms and inserted two zeros.

**Example 7.5.2** Given \( x(n) = e^{-n/2} u(n) \), find \( x(3n/5) \). Here there is a 5-fold up-sampling and a 3-fold down sampling. Since the denominator is bigger there is a *net up-sampling by a factor of 1.67*.

\[
x(n) = \{1, a, a^2, a^3, a^4, a^5, a^6, a^7, a^8, a^9, a^{10}, \ldots \}
\]

**Method A** Up-sampling followed by down sampling is given below. The 5-fold up-sampled signal, \( y_u(n) \), is obtained by inserting 4 zeros shown in bold face between every pair of consecutive samples in \( x(n) \)

\[
y_u(n) = x(n/5) = \{1, 0, 0, 0, a, 0, 0, 0, a^2, 0, 0, 0, a^3, 0, 0, 0, a^4, 0, 0, 0, 0, a^5, 0, 0, 0, a^{10}, 0, 0, 0, a^{15}, 0, 0, 0, a^{20}, \ldots \}
\]

The 3-fold down-sampled signal, \( y_f(n) \), is obtained by keeping every third sample in \( y_u(n) \) and discarding the rest (shown underlined)

\[
y_f(n) = \{1, 0, 0, 0, a, 0, 0, 0, 0, a^2, 0, 0, 0, a^3, 0, 0, 0, 0, a^4, 0, 0, 0, 0, a^5, 0, 0, 0, a^{10}, 0, 0, 0, a^{15}, 0, 0, 0, a^{20}, \ldots \}
\]
\[ y_1(n) = y_d(3n) = x(3n/5) = \{1, 0, 0, 0, a^3, 0, 0, 0, 0, a^6, 0, 0, 0, a^9, 0, \ldots \} \]

**Method B** Down-sampling followed by up-sampling is given below. The 3-fold down-sampled signal, \( y_d(n) \), is obtained by keeping every third sample in \( x(n) \) and discarding the rest (shown underlined)

\[ x(n) = \{1, a, a^2, a^3, a^4, a^5, a^6, a^7, a^8, a^9, a^{10}, a^{11}, \ldots \} \]

\[ y_d(n) = x(3n) = \{1, a^3, a^6, a^9, a^{12}, \ldots \} \]

The 5-fold up-sampled signal, \( y_2(n) \), is obtained by inserting 4 zeros shown in bold face between every pair of samples in \( y_d(n) \)

\[ y_2(n) = y_u(n/5) = x(3n/5) \]

\[ y_2(n) = \{1, 0, 0, 0, 0, a^3, 0, 0, 0, 0, a^6, 0, 0, 0, a^9, 0, 0, 0, a^{12}, \ldots \} = \{1, 0, 0, 0, 0, a^3, 0, 0, 0, 0, a^6, 0, 0, 0, a^9, 0, 0, 0, a^{12}, \ldots \} \]

It is seen that \( y_1(n) = y_2(n) \).

**Example 7.5.3** Given that \( x(n) = \{1, a, a^2, a^3, a^4, a^5, a^6, a^7, a^8, a^9, a^{10}, \ldots \} \) is the input,

1. Find the output \( y_1(n) \) of a cascade of a 2-fold up-sampler followed by a 4-fold down sampler.
2. Find the output \( y_2(n) \) of a cascade of a 4-fold down sampler followed by a 2-fold up-sampler.

**Solution** Note that the down-sampling factor \( M = 4 \) and the up-sampling factor \( L = 2 \) are not co-prime since they have a factor in common. The ratio \( M/L = 4/2 \), as given, is not in its reduced form. As a result we do not expect that \( y_1(n) \) and \( y_2(n) \) will be equal. Specifically, in the first case we have

\[ x(n) = \{1, a, a^2, a^3, a^4, a^5, a^6, a^7, a^8, a^9, a^{10}, \ldots \} \]

Up-sample by inserting a zero (shown bold face) between consecutive samples of \( x(n) \) resulting in \( y_u(n) \)

\[ y_u(n) = x(n/2) = \{1, 0, a, 0, a^2, 0, a^3, 0, a^4, 0, a^5, 0, a^6, 0, a^7, 0, a^8, 0, a^9, 0, a^{10}, \ldots \} \]

Down-sample by keeping every fourth sample of \( y_u(n) \) and discarding the three samples in between resulting in \( y_1(n) \)

\[ y_1(n) = y_u(4n) = x(4n/2) = \{1, a^2, a^4, a^6, a^8, a^{10}, \ldots \} \]

In the second case

\[ x(n) = \{1, a, a^2, a^3, a^4, a^5, a^6, a^7, a^8, a^9, a^{10}, \ldots \} \]

\[ y_d(n) = x(4n) = \{1, a^4, a^8, a^{12}, \ldots \} \]
\[ y_2(n) = y_d(n/2) = x(4n/2) = \{1, 0, a^4, 0, a^8, 0, a^{12}, \ldots\} \]

It is seen that \( y_1(n) \neq y_2(n) \).

**Sampling Rate Conversion by a Rational Factor \( L/M \)** Here the sampling rate is being converted by a non-integral factor such as 0.6 or 1.5. That is, given \( x(n) \) with a sampling rate of \( F_x \) we want to obtain \( y(n) \) with a sampling rate of \( F_y \) of, say, \( 0.6F_x \) (decimation) or \( 1.5F_x \) (interpolation).

Take, for instance \( L/M = 3/5 \). Here the basic approach is to first interpolate (up-sample) by a factor of \( L = 3 \) and then decimate (down-sample) by a factor of \( M = 5 \). The net effect of the cascade of interpolation followed by decimation is to change the sampling rate by a rational factor \( L/M \), that is,

\[
F_y = \left( \frac{L}{M} \right) F_x = \left( \frac{3}{5} \right) F_x = 0.6 F_x.
\]

The corresponding signal is given by \( y(n) = x(5n/3) \), ignoring the filters involved. (This can also be done by first down-sampling and then up-sampling).

The block diagram of the scheme where the interpolator precedes the decimator is shown below.

In general, if \( L < M \) we have a rational decimator and if \( L > M \) we have a rational interpolator. In this set-up interpolation is done before decimation in order to work at the higher sampling rate so as to preserve the original spectral characteristics of \( x(n) \). Recall that unless \( x(n) \) was originally over-sampled, decimation in itself or decimation prior to interpolation will modify the spectrum of \( x(n) \) irrecoverably.

The above configuration has an added benefit that the two filters \( H_u(z) \) and \( H_d(z) \) in series (which operate at the same sampling rate) can be combined into a single equivalent low pass filter with a frequency response of \( H(\omega) = H_u(\omega)H_d(\omega) \). The simplified configuration is shown below.
The bandwidth of the anti-imaging filter $H_u(z)$ is $\pi/L$ rad., and that of the anti-aliasing filter $H_d(z)$ is $\pi/M$ rad., so that the bandwidth of the composite anti-imaging and anti-aliasing filter $H(\omega)$ is

$$\omega_c = \min\left(\frac{\pi}{L}, \frac{\pi}{M}\right)$$

and the frequency response is given by

$$H(\omega) = \begin{cases} L, & 0 \leq |\omega| < \omega_c \\ 0, & \omega_c \leq |\omega| \leq \pi \end{cases}$$

In the time domain, the output of the up-sampler, $v(n)$, is given by

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

and the output of the linear time-invariant filter $H(z)$ is

$$w(n) = \sum_{k=-\infty}^{\infty} h(n-k)v(k)$$

Since $v(k) = 0$ except at $k = rL$, where $r$ is an integer between $-\infty$ to $\infty$, we set $k = rL$. As $k$ goes from $-\infty$ to $\infty$, $r$ goes from $-\infty$ to $\infty$, and $v(rL) = x(rL/L) = x(r)$. Therefore

$$w(n) = \sum_{r=-\infty}^{\infty} h(n-rL)v(rL) = \sum_{r=-\infty}^{\infty} h(n-rL)x(r) = \sum_{k=-\infty}^{\infty} h(n-kL)x(k)$$

Finally the output of the down sampler is

$$y(n) = w(Mn) = \sum_{k=-\infty}^{\infty} h(Mn-kL)x(k)$$

In summary, sampling rate conversion by the factor $L/M$ can be achieved by first increasing the sampling rate by $L$, accomplished by inserting $L-1$ zeros between successive samples of the input $x(n)$, followed by linear filtering of the resulting sequence to eliminate unwanted images of $X(\omega)$ and, finally, by down-sampling the filtered signal by the factor $M$ to get the output $y(n)$. The sampling rates are related by $F_y = (L/M)F_x$. If $F_y > F_x$, that is, $L > M$, the low pass filter acts as an anti-imaging post-filter to the up-sampler. If $F_y < F_x$, that is, $L < M$, the low pass filter acts as an anti-aliasing pre-filter to the down-sampler.
Example 7.5.4 The signal $x(t) = \cos 2\pi t + 0.8 \sin 2\pi 4t$ is sampled at 40 Hz to generate $x(n)$.

a) Give an expression for $x(n)$
b) Design a sampling rate converter to change the sampling frequency of $x(n)$ by a factor of 2.5. Give an expression for $y(n)$.
c) Design a sampling rate converter to change the sampling frequency of $x(n)$ by a factor of 0.4. Give an expression for $y(n)$.

Solution

a) $x(n) = \cos \frac{2\pi}{10} n + 0.8 \sin \frac{2\pi}{5} n$
b) The rate conversion factor is $L/M = 2.5 = 5/2$. We do this by first up-sampling by a factor of 5, then down-sampling by a factor of 2. Roughly speaking, $y(n) = x(2n/5)$.

c) The rate conversion factor is $L/M = 0.4 = 2/5$. We do this by first up-sampling by a factor of 2, then down-sampling by a factor of 5. Roughly speaking, $y(n) = x(5n/2)$. Band width of filter = $\pi/5$ rad., and gain = 2, once again determined by the up-sampler.

Write equations for $v(n)$, $w(n)$, and $y(n)$ based on the above diagram and assuming $H(z)$ is an FIR filter of $N$ coefficients.

c) The rate conversion factor is $L/M = 0.4 = 2/5$. We do this by first up-sampling by a factor of 2, then down-sampling by a factor of 5. Roughly speaking, $y(n) = x(5n/2)$. Band width of filter = $\pi/5$ rad., and gain = 2, once again determined by the up-sampler.

Multistage conversion The composite anti-imaging and anti-aliasing LP filter band width is $\pi/L$ or $\pi/M$ whichever is smaller. That is, the filter band width is determined by the larger number of $L$ and $M$. However, if either $L$ or $M$ is a very large number the filter bandwidth is very narrow.
Narrowband FIR (linear phase) filters can require a very large number of coefficients (see Unit VI, FIR Filters, Example 6). This can pose problems in

1. Increased storage space for coefficients,
2. Long computation time, and
3. Detrimental finite word length effects

The latter drawback is minimized by using a multistage sampling rate converter where the conversion ratio \( \frac{L}{M} \) is factored into the product of several ratios each of which has its own smaller \( L \) and \( M \) values. If, for instance, the ratio \( \frac{L}{M} \) is split into the product of two ratios

\[
\frac{L}{M} = \left( \frac{L_1}{M_1} \right) \left( \frac{L_2}{M_2} \right)
\]

where the \( L \)'s and \( M \)'s on the right hand side are smaller, we may implement the rate conversion in two stages as shown below

![Diagram of a multistage sampling rate converter]

**Example 7.5.5 [Rational sampling rate converter][CD, DAT]** Digital audio tape (DAT) used in sound recording studios has a sampling rate of 48 kHz, while a compact disc (CD) is recorded at a sampling rate of 44.1 kHz. Design a sampling rate converter that will convert the DAT signal \( x(n) \) to a signal \( y(n) \) for CD recording.

**Over-sampling analog-to-digital converter (ADC) [Ref. SKMitra, Sec 4.8.4]** A practical difficulty with analog to digital conversion is the need for a low pass analog anti-aliasing prefilter to band limit the signal to less than half of the sampling rate. High-order analog filters are expensive, and they are also difficult to keep in calibration. The combination of over-sampling followed by down-sampling can be used to transfer some of the anti-aliasing burden from the analog into the digital domain, and thereby use a simpler low order analog filter.

As an example, a typical compact disk encoding system may employ an over-sampling sigma-delta A/D converter which over-samples at 3175.2 kHz which is then brought down to the CD sampling rate of 44.1 kHz. This amounts to over-sampling by a factor of 72 (= 3175.2/44.1).

Suppose the range of frequencies of interest in the signal \( x(t) \) is \( 0 \leq |F| \leq F_M \). Normally we would band-limit \( x(t) \) to the maximum frequency \( F_M \) with a sharp cut-off analog low pass filter and sample it at a rate of \( F_s = 2F_M \) at least (strictly, \( F_s \geq 2F_M \)). Suppose that we instead over-sample \( x(t) \) by an integer factor \( M \), at \( F_s = M(2F_M) \). This significantly reduces the requirements for the anti-aliasing filter which may be specified more leniently as

\[
H_a(s) = \begin{cases} 
1, & 0 \leq |F| \leq F_M \\
0, & MF_M \leq |F| < \infty 
\end{cases}
\]
Though this is still an ideal low pass filter in the pass band and stop band, its transition band width is no longer zero and it may be approximated with an inexpensive first- or second-order Butterworth filter as shown below.

![Diagram of filter response](image)

We are paying the price of a higher sampling rate for the benefit of a cheaper analog anti-aliasing filter. The result is a discrete-time signal that is sampled at a much higher rate than $2F_M$. Following the sampling operation, we can reduce this sampling rate to the minimum value using a decimator. The resulting structure of the over-sampling ADC is shown in the block diagram below. There are two anti-aliasing filters, a low-order analog filter $H_a(s)$ with cut-off frequency $F_M$ rad/sec., and a high-order digital filter $H_d(z)$, with a cut-off frequency $(\pi/M)$ rad/sample.

**Over-sampling AD converter**

![Block diagram of over-sampling AD converter](image)

A second benefit of using the over-sampling ADC is the reduction in quantization noise. If $q$ is the quantization step size (precision), then the quantization noise in $x(n)$ is $\sigma_x^2 = q^2/12$, and the noise appearing in the output $y(n)$ in the above scheme is $\sigma_y^2 = \sigma_x^2 / M = q^2 / 12M$, a reduction by a factor of $M.$
7.6 Identities

A sampling rate converter (the \( \downarrow M \) or \( \uparrow L \) operation) is a linear time-varying system. On the other hand, the filters \( H_d(z) \) and \( H_u(z) \) are linear time-invariant systems. In general, the order of a sampling rate converter and a linear time-invariant system cannot be interchanged. We derive below several identities, two of which are known as noble identities (viz., identities 3 and 6, all the others being special cases), which help to swap the position of a filter with that of a down-sampler or up-sampler by properly modifying the filter.

Recall that the input-output description of a down-sampler is

\[
y(n) = x(Mn) \quad \leftrightarrow \quad Y(z) = \frac{1}{M} \sum_{k=0}^{M-1} X(e^{-j2\pi k/M} z^{1/M}) \quad \leftrightarrow \quad Y(e^{j\omega}) = \frac{1}{M} \sum_{k=0}^{M-1} X(e^{-j(\omega-2\pi k)/M})
\]

and the same for an up-sampler is

\[
y(n) = \begin{cases} x(n/L), & n = 0, \pm L, \pm 2L, \ldots \\ 0, & \text{otherwise} \end{cases} \quad \leftrightarrow \quad Y(z) = X(z^L) \quad \leftrightarrow \quad Y(e^{j\omega}) = X(e^{j\omega L})
\]

which we use in the following development.

Example 7.6.1 Show that the following systems are equivalent.

\[
\begin{array}{c}
x(n) \quad \downarrow \quad a \quad \downarrow \quad M \quad y(n) \\
\end{array}
\quad \leftrightarrow \quad \begin{array}{c}
x(n) \quad \downarrow \quad M \quad \uparrow \quad a \quad \uparrow \quad \downarrow \quad y(n) \\
\end{array}
\]

In the structure on the left the process of generating \( y(.) \) consists of multiplying every input sample \( x(.) \) by \( a \) and then, in the down-sampling process, dropping \((M-1)\) of these products for every \( M^{th} \) one we keep. The structure on the right is more efficient computationally: the \((M-1)\) samples are dropped first and every \( M^{th} \) is multiplied by \( a \), that is, only the samples that are retained are multiplied. The number of multiplications is reduced by \( \frac{100(M-1)}{M} \) %.

Example 7.6.2 Show that the following systems are equivalent.

\[
\begin{array}{c}
x(n) \quad \uparrow \quad \downarrow \quad a \quad \downarrow \quad y(n) \\
\end{array}
\quad \leftrightarrow \quad \begin{array}{c}
x(n) \quad \uparrow \quad a \quad \downarrow \quad \uparrow \quad \downarrow \quad y(n) \\
\end{array}
\]

In the structure on the left the process of generating \( y(.) \) consists of first up-sampling, that is, inserting zeros between consecutive points of \( x(n) \) and then multiplying by \( a \). In the process the \((L-1)\) zeros are also multiplied. The structure on the right is more efficient computationally: the sequence \( x(n) \) is first multiplied and then the zeros inserted. The number of multiplications is reduced by \( \frac{100(L-1)}{L} \) %.
Identity #1 If we use the notation $\downarrow M\{ . \}$ to mean the down-sampling of the signal in braces, then we have

$$\downarrow M\{ a x_1(n) + b x_2(n) \} = \downarrow M\{ a x_1(n) \} + \downarrow M\{ b x_2(n) \} \quad \rightarrow (1)$$

In words, the result of down-sampling the weighted sum of signals equals the weighted sum of the down-sampled signals. In other words, the two block diagrams below are equivalent.

In the diagram on the left the weighted sum of two inputs is down-sampled:

$$w(n) = a x_1(n) + b x_2(n) \quad \leftrightarrow \quad W(e^{j\omega}) = a X_1(e^{j\omega}) + b X_2(e^{j\omega})$$

The output and its spectrum are then given by

$$y(n) = w(Mn)$$
$$Y(e^{j\omega}) = \frac{1}{M} \sum_{k=0}^{M-1} W(e^{j(\omega - 2\pi k)/M})$$
$$= \frac{a}{M} \sum_{k=0}^{M-1} X_1(e^{j(\omega - 2\pi k)/M}) + \frac{b}{M} \sum_{k=0}^{M-1} X_2(e^{j(\omega - 2\pi k)/M}) \quad \rightarrow (A)$$

In the diagram on the right the weighted inputs are down-sampled and added to form the output.

$$y_1(n) = a x_1(Mn) \quad \leftrightarrow \quad Y_1(e^{j\omega}) = \frac{a}{M} \sum_{k=0}^{M-1} X_1(e^{j(\omega - 2\pi k)/M})$$
$$y_2(n) = b x_2(Mn) \quad \leftrightarrow \quad Y_2(e^{j\omega}) = \frac{b}{M} \sum_{k=0}^{M-1} X_2(e^{j(\omega - 2\pi k)/M})$$

The output and its spectrum are given by

$$y(n) = y_1(n) + y_2(n)$$
$$Y(e^{j\omega}) = Y_1(e^{j\omega}) + Y_2(e^{j\omega}) = \frac{a}{M} \sum_{k=0}^{M-1} X_1(e^{j(\omega - 2\pi k)/M}) + \frac{b}{M} \sum_{k=0}^{M-1} X_2(e^{j(\omega - 2\pi k)/M}) \quad \rightarrow (B)$$

Equations (A) and (B) are identical. QED.

Eventually, by virtue of Example 6.1, the down-samplers are moved to the upstream side of the multipliers which would correspond to Eq. (2).

Identity #2 A delay of $M$ sample periods before an $M$-fold down-sampler is the same as a delay of one sample period after the down-sampler.
In the first case (diagram on the left) we have

\[ y_1(n) = x(n-M) \quad \leftrightarrow \quad Y_1(z) = z^{-M} X(z) \quad \rightarrow (1) \]

and

\[ y(n) = y_1(Mn) \quad \leftrightarrow \quad Y(z) = \frac{1}{M} \sum_{k=0}^{M-1} Y_1(e^{-j2\pi k/M} z^{1/M}) \quad \rightarrow (2) \]

Note from (1) that

\[ Y_1(e^{-j2\pi k/M} z^{1/M}) = Y_1(z) \bigg|_{z=e^{-j2\pi k/M} z^{1/M}} = z^{-M} X(z) \bigg|_{z=e^{-j2\pi k/M} z^{1/M}} = (e^{-j2\pi k/M} z^{1/M})^{-M} X(e^{-j2\pi k/M} z^{1/M}) \]

Substituting this in (2) we have

\[ Y(z) = \frac{1}{M} \sum_{k=0}^{M-1} (e^{-j2\pi k/M} z^{1/M})^{-M} X(e^{-j2\pi k/M} z^{1/M}) \]

\[ = \frac{1}{M} \sum_{k=0}^{M-1} (e^{-j2\pi k/M} z^{1/M}) X(e^{-j2\pi k/M} z^{1/M}) = \frac{1}{M} \sum_{k=0}^{M-1} (z^{-1}) X(e^{-j2\pi k/M} z^{1/M}) \]

\[ = z^{-1} \frac{1}{M} \sum_{k=0}^{M-1} X(e^{-j2\pi k/M} z^{1/M}) \rightarrow (A) \]

In the second case (diagram on the right) we have

\[ y_2(n) = x(nM) \quad \leftrightarrow \quad Y_2(z) = \frac{1}{M} \sum_{k=0}^{M-1} X(e^{-2\pi k/M} z^{1/M}) \quad \rightarrow (3) \]

\[ y(n) = y_2(n-M) \quad \leftrightarrow \quad Y(z) = z^{-M} Y_2(z) \quad \rightarrow (4) \]

Substituting from (3) into (4) we have

\[ Y(z) = z^{-M} Y_2(z) = z^{-1} \frac{1}{M} \sum_{k=0}^{M-1} X(e^{-2\pi k/M} z^{1/M}) \rightarrow (B) \]

Equations (A) and (B) are identical. QED.

**Identity #3 (Noble identity)** An M-fold down-sampler followed by a linear time invariant filter \( H(z) \) is equivalent to a linear time invariant filter \( H(z^M) \) followed by an M-fold down-sampler. Note that the second identity is a special case of this identity with \( H(z) = z^{-I} \) and \( H(z^M) = z^{-M} \).

For the system consisting of the filter followed by the down-sampler we have

\[ Y_1(z) = H(z^M) X(z) \]

DSP-7 (Multirate) 37 of 58 Dr. Ravi Billa
\[
Y(z) = \frac{1}{M} \sum_{k=0}^{M-1} Y_1(e^{-j2\pi k/M} z^{1/M})
\]

Note that
\[
Y_1(e^{-j2\pi k/M} z^{1/M}) = Y_1(z) \bigg|_{z=e^{-j2\pi k/M} z^{1/M}} = H(z^M) X(z) \bigg|_{z=e^{-j2\pi k/M} z^{1/M}} = H(e^{-j2\pi k/M} z^{1/M}) X(e^{-j2\pi k/M} z^{1/M}) = H(1) X(e^{-j2\pi k/M} z^{1/M})
\]

Thus
\[
Y(z) = \frac{1}{M} \sum_{k=0}^{M-1} H(z) X(e^{-j2\pi k/M} z^{1/M}) = H(z) \frac{1}{M} \sum_{k=0}^{M-1} X(e^{-j2\pi k/M} z^{1/M}) \quad \rightarrow (A)
\]

For the system consisting of the down sampler followed by the filter we have
\[
y_2(n) = x(nM)
\]
\[
Y_2(z) = \frac{1}{M} \sum_{k=0}^{M-1} X(e^{-2\pi k/M} z^{1/M})
\]
\[
Y(z) = H(z) Y_2(z) = H(z) \frac{1}{M} \sum_{k=0}^{M-1} X(e^{-2\pi k/M} z^{1/M}) \quad \rightarrow (B)
\]

Equations (A) and (B) are identical. Thus the two systems are equivalent.
Identity #4 (This identity contains no summing junction as does identity #1.) Eventually, by virtue of Example 6.2, the up-samplers are moved to the downstream side of the multipliers.

Identity #5 A delay of one sample period before an \( L \)-fold up-sampler is the same as a delay of \( L \) sample periods after the up-sampler.

For the system consisting of the up-sampler preceded by \( z^{-1} \) we have

\[
Y_1(z) = z^{-1} X(z) \\
Y(z) = Y_1(z^L) = z^{-L} X(z^L) \rightarrow (A)
\]

For the system consisting of the up-sampler followed by \( z^{-L} \) we have

\[
Y_2(z) = X(z^L) \\
Y(z) = z^{L-1} Y_2(z) = z^{-L} X(z^L) \rightarrow (B)
\]

Since equations (A) and (B) are identical the two systems are equivalent.

Identity #6 (Noble identity) An \( L \)-fold up-sampler preceded by a linear time invariant filter \( H(z) \) is equivalent to a linear time invariant filter \( H(z^L) \) preceded by an \( L \)-fold up-sampler. Note that the fifth identity is a special case of this identity with \( H(z) = z^{-1} \) and \( H(z^L) = z^{-L} \).

For the system consisting of the filter followed by the up-sampler we have

\[
Y_1(z) = H(z) X(z) \\
Y(z) = Y_1(z^L) = H(z^L) X(z^L) \rightarrow (A)
\]
For the system consisting of the up-sampler followed by the filter we have

\[
\begin{align*}
Y_2(z) &= X(z^L) \\
Y(z) &= H(z^L) Y_2(z) = H(z^L) X(z^L) \rightarrow (B)
\end{align*}
\]

Since equations (A) and (B) are identical the two systems are equivalent.
7.7 FIR implementation of sampling rate conversion

The anti-aliasing filter in a decimator and the anti-imaging filter in an interpolator may each be either an FIR or an IIR filter, the former being preferred since it offers linear phase. We give here the FIR implementation.

Implementation of Decimator The process of decimation consists of an anti-aliasing (low pass) filter followed by a down-sampler. We repeat below the block diagram developed earlier.

Taking the filter $H(z)$ to be an FIR filter, the decimator is implemented as shown below, using a direct form structure for $H(z)$. Note that the coefficients $\{b_i, i = 0 to (N-1)\}$, used in earlier formulations, are the same as $\{h(i), i = 0 to (N-1)\}$ used in this diagram. Further, the FIR filter here is implemented with $N$ coefficients rather than $(N+1)$ coefficients.
The implementation equations which correspond to the above structure are

\[ v(n) = \sum_{r=0}^{N-1} h(r) x(n - r) \quad \text{and} \quad y(n) = v(Mn) = \sum_{r=0}^{N-1} h(r) x(Mn - r) \]

We first compute \( v(n) \) for all values of \( n \). Then \( y(n) \) is obtained by retaining every \( M^{th} \) value of \( v(.) \), dropping the intervening \((M-1)\) values. In other words there are \((M-1)\) computations of \( v(.) \) that could be avoided.

We may use identity #1 to move the down-sampler to the left of the adders and the result of Example 6.1 to move it to the upstream side of the multipliers as shown below. As a result the number of multiplications is reduced by \( \frac{100(M-1)}{M} \)%.

**Implementation of Interpolator** The process of interpolation consists of an up-sampler followed by an anti-imaging (low pass) filter. We repeat below the block diagram developed earlier.

![Block diagram of interpolator](image-url)
Taking the filter $H(z)$ to be an FIR filter, the interpolator is implemented as shown below, using a direct form structure for $H(z)$. Note that the coefficients $\{b_i, i = 0\text{ to } (N-1)\}$, used in earlier formulations, are the same as $\{h(i), i = 0\text{ to } (N-1)\}$ used in this diagram. Further, the FIR filter here is implemented with $N$ coefficients rather than $(N+1)$ coefficients.

We shall find it more convenient to use the transposed form of the FIR filter rather than the structure actually shown here, so here follows a digression on the transposed structure.
Start of Digression

Transposed Structure According to the transposition theorem the transposed form of a filter has the same transfer function as the filter. The transposed form of a given filter structure is found as follows:

1. Construct the signal flow graph of the filter.
2. Reverse the direction of arrow on every branch.
3. Interchange the inputs and outputs.
4. Reverse the roles of all nodes: an adder becomes a pick-off point and a pick-off point becomes an adder.

If we apply this procedure to the FIR structure in the above interpolator the result is the transpose shown below. The intermediate steps are omitted.

Start of Aside

As an aside note that the FIR structure is simple enough that the following algebraic manipulation can be used to proceed from the original FIR structure to the transpose structure. Let the system function be
\[
\frac{V(z)}{U(z)} = H(z) = h(0) + h(1)z^{-1} + h(2)z^{-2} + h(3)z^{-3} + h(4)z^{-4} + \ldots + h(N-1)z^{-(N-1)}
\]

This may be rearranged as

\[
V(z) = H(z)U(z)
\]

\[
= h(0)U(z) + h(1)z^{-1}U(z) + h(2)z^{-2}U(z) + h(3)z^{-3}U(z) + h(4)z^{-4}U(z) + \ldots \\
+ h(N-1)z^{-(N-1)}U(z)
\]

\[
= h(0)U(z) + z^{-1}(h(1)U(z) + z^{-1}(h(2)U(z) + z^{-1}(h(3)U(z) + \ldots \\
\ldots + z^{-1}(h(N-2)U(z) + z^{-1}h(N-1)U(z))))
\]

This last equation, proceeding from right to left, performs the following in the time domain:

- Multiply \(u(n)\) by \(h(N-1)\), giving \(h(N-1)u(n)\)
- Delay by 1 unit, giving \(h(N-1)u(n-1)\)
- Add to \(h(N-2)u(n)\), giving \(h(N-2)u(n) + h(N-1)u(n-1)\)
- Delay by 1 unit, giving \(h(N-2)u(n-1) + h(N-1)u(n-2)\)
- Add to \(h(N-3)u(n)\), giving \(h(N-3)u(n) + h(N-2)u(n-1) + h(N-1)u(n-2)\)
- Delay by 1 unit, giving \(h(N-3)u(n-1) + h(N-2)u(n-2) + h(N-1)u(n-3)\)
- \ldots
- Add to \(h(0)u(n)\)

all of which yields the implementation of the difference equation

\[
v(n) = h(0)u(n) + h(1)u(n-1) + h(2)u(n-2) + \ldots \\
\ldots + h(N-2)u(n-N-2) + h(N-1)u(n-N-1)
\]

Further, the equation

\[
V(z) = h(0)U(z) + z^{-1}(h(1)U(z) + z^{-1}(h(2)U(z) + z^{-1}(h(3)U(z) + \ldots \\
\ldots + z^{-1}(h(N-2)U(z) + z^{-1}h(N-1)U(z))))
\]

also suggests the transpose structure previously developed according to the rules.

End of Aside
End of Digression
Resumption of Implementation of Interpolator Using the transposed form of the FIR filter the structure of the interpolator appears as below:

\[ y(n) = \sum (h(0) u(n) + h(1) (u(n) - z^{-1} v(n)) + h(2) (u(n) - 2z^{-1} v(n)) + \ldots + h(N-2) (u(n) - (N-2)z^{-1} v(n)) + h(N-1) (u(n) - (N-1)z^{-1} v(n)) \)
We may use identity #4 and the result of Example 6.2 to move the up-sampler to the right of the multipliers as shown below. As a result the number of multiplications is reduced by \( \frac{100(L - 1)}{L} \)%.
7.8 Polyphase structures

The polyphase structure for FIR Filters was developed for the efficient implementation of sampling rate converters; however, it can be used in other applications. Further, the polyphase structure can be developed for any filter, FIR or IIR. We give below an introduction.

Polyphase Structure for FIR Filters The impulse response of the FIR filter \( h(n) \) is of finite length, \( N \). The system function with \( N \) coefficients is

\[
H(z) = \sum_{n=0}^{N-1} h(n) z^{-n} = h(0) + h(1) z^{-1} + h(2) z^{-2} + h(3) z^{-3} + h(4) z^{-4} + \ldots + h(N-1) z^{-(N-1)}
\]

We shall use another parameter \( M \): we shall divide the number of coefficients into \( M \) groups (or branches or phases), modulo \( M \). In other words the \( N \) terms in \( H(z) \) are arranged into \( M \) branches with each branch containing at most \( \left(\ln \left(\frac{N}{M}+1\right)\right) \) terms.

Type 1 polyphase decomposition For illustration, let \( N = 11 \) and \( M = 2 \) so that one group contains 6 coefficients and the other 5 as developed below:

\[
H(z) = \sum_{n=0}^{10} h(n) z^{-n} = h(0) + h(1) z^{-1} + h(2) z^{-2} + h(3) z^{-3} + h(4) z^{-4} + \ldots + h(10) z^{-10}
\]

\[
= h(0) + h(2) z^{-2} + h(4) z^{-4} + h(6) z^{-6} + h(8) z^{-8} + h(10) z^{-10} \quad \leftarrow \text{1st group}
\]

\[
+ h(1) z^{-1} + h(3) z^{-3} + h(5) z^{-5} + h(7) z^{-7} + h(9) z^{-9} \quad \leftarrow \text{2nd group}
\]

\[
= h(0) + h(2) z^{-2} + h(4) z^{-4} + h(6) z^{-6} + h(8) z^{-8} + h(10) z^{-10}
\]

\[
+ z^{-1} \{ h(1) + h(3) z^{-2} + h(5) z^{-4} + h(7) z^{-6} + h(9) z^{-8} \}
\]

Define

\[
\ln \left(\frac{N}{M}+1\right) = \text{integer part of the argument}
\]

\[
P_0(z) = h(0) + h(2) z^{-1} + h(4) z^{-2} + h(6) z^{-3} + h(8) z^{-4} + h(10) z^{-5} = \sum_{n=0}^{\lfloor \ln \left(\frac{N}{M}+1\right) \rfloor} h(2n+0) z^{-n}
\]

(More generally, \( P_0(z) = \sum_{n=0}^{\lfloor \ln \left(\frac{N}{M}+1\right) \rfloor} h(Mn+0) z^{-n} \))

\[
P_1(z) = h(1) + h(3) z^{-1} + h(5) z^{-2} + h(7) z^{-3} + h(9) z^{-4} = \sum_{n=0}^{\lfloor \ln \left(\frac{N}{M}+1\right) \rfloor} h(2n+1) z^{-n} \quad (h(11) = 0)
\]

(More generally, \( P_1(z) = \sum_{n=0}^{\lfloor \ln \left(\frac{N}{M}+1\right) \rfloor} h(Mn+1) z^{-n} \))

In this specific case we have

\[
\frac{Y(z)}{X(z)} = H(z) = P_0(z^2) + z^{-1} P_1(z^2) = \sum_{n=0}^{1} z^{-n} P_n(z^2)
\]
This decomposition of \( H(z) \) is known as **type 1 polyphase decomposition**. The corresponding structure is shown below. The functions \( P_0(z) \) and \( P_1(z) \) can each be implemented as a direct form.

By observing the expressions for \( P_0(z) \) and \( P_1(z) \) we can further generalize the functions \( P_m(z) \) for any \( m \) as

\[
P_m(z) = \sum_{n=0}^{\lfloor \frac{(N-1)/M} \rfloor} h(Mn+m)z^{-n}, \quad m = 0 \text{ to } (M-1)
\]

Further, the system function becomes

\[
\frac{Y(z)}{X(z)} = H(z) = P_0(z^M) + z^{-1}P_1(z^M) + \ldots + z^{-(M-1)}P_{M-1}(z^M) = \sum_{n=0}^{M-1} z^{-n}P_n(z^M)
\]

**Example 7.8.1** As another example, let \( N = 11 \) and \( M = 3 \).

\[
H(z) = \sum_{n=0}^{10} h(n)z^{-n} = h(0) + h(1)z^{-1} + h(2)z^{-2} + h(3)z^{-3} + h(4)z^{-4} + \ldots + h(10)z^{-10}
\]

\[
= h(0) + h(3)z^{-3} + h(6)z^{-6} + h(9)z^{-9} \quad - 1^{\text{st}} \text{ group}
\]
\[
+ h(1)z^{-1} + h(4)z^{-4} + h(7)z^{-7} + h(10)z^{-10} \quad - 2^{\text{nd}} \text{ group}
\]
\[
+ h(2)z^{-2} + h(5)z^{-5} + h(8)z^{-8} \quad - 3^{\text{rd}} \text{ group}
\]

\[
H(z) = h(0) + h(3)z^{-3} + h(6)z^{-6} + h(9)z^{-9}
\]
\[
+ z^{-1}\{h(1) + h(4)z^{-3} + h(7)z^{-6} + h(10)z^{-9}\}
\]
\[
+ z^{-2}\{h(2) + h(5)z^{-3} + h(8)z^{-6}\}
\]

Define

\[
\lfloor \frac{N-1}{M} \rfloor = \text{integer part of the argument}
\]

\[
P_0(z) = h(0) + h(3)z^{-1} + h(6)z^{-2} + h(9)z^{-3} = \sum_{n=0}^{\lfloor \frac{(N-1)/3} \rfloor} h(3n+0)z^{-n}
\]

(More generally, \( P_0(z) = \sum_{n=0}^{\lfloor \frac{(N-1)/M} \rfloor} h(Mn+0)z^{-n} \))

\[
P_1(z) = h(1) + h(4)z^{-1} + h(7)z^{-2} + h(10)z^{-3} = \sum_{n=0}^{\lfloor \frac{(N-1)/3} \rfloor} h(3n+1)z^{-n}
\]

(More generally, \( P_1(z) = \sum_{n=0}^{\lfloor \frac{(N-1)/M} \rfloor} h(Mn+1)z^{-n} \))
\[ P_2(z) = h(2) + h(5)z^{-1} + h(8)z^{-2} = \sum_{n=0}^{\lfloor \frac{N}{M} \rfloor - 1} h(3n)z^{-n} \quad (h(11) = 0) \]

(More generally, \( P_2(z) = \sum_{n=0}^{\lfloor \frac{MN}{N+1} \rfloor} h(Mn+1)z^{-n} \))

In this specific case we have

\[
\frac{Y(z)}{X(z)} = H(z) = P_0(z^3) + z^{-1}P_1(z^3) + z^{-2}P_2(z^3) = \sum_{m=0}^{2} z^{-m}P_m(z^3)
\]
**Generalization** As mentioned earlier, in the general case for an arbitrary $M (\leq N)$ we have

$$P_m(z) = \sum_{n=0}^{\lfloor\frac{N-1}{M}\rfloor} h(Mn+m)z^{-n}, \quad m = 0 \text{ to } (M-1)$$

and

$$\frac{Y(z)}{X(z)} = H(z) = P_0(z^M) + z^{-1} P_1(z^M) + \ldots + z^{-(M-1)} P_{M-1}(z^M) = \sum_{m=0}^{M-1} z^{-m} P_m(z^M)$$

This overall operation is known as polyphase filtering.
Type 2 polyphase decomposition Given  
\[ \frac{Y(z)}{X(z)} = H(z) = P_0(z^M) + z^{-1} P_1(z^M) + \ldots + z^{-(M-1)} P_{M-1}(z^M) = \sum_{m=0}^{M-1} z^{-m} P_m(z^M) \]
set \( m = M-1-k \): as \( m \) goes from 0 to \( M-1 \), \( k \) goes from \( M-1 \) to 0. Thus  
\[ H(z) = \sum_{k=M-1}^{0} z^{-(M-1-k)} P_{M-1-k}(z^M) \]
Let  
\[ Q_k(z^M) = z^{-(M-1-k)} P_{M-1-k}(z^M) \]
This gives us the type 2 polyphase decomposition  
\[ H(z) = \sum_{k=M-1}^{0} Q_k(z^M) = \sum_{k=0}^{M-1} Q_k(z^M) = Q_0(z^M) + Q_1(z^M) + \ldots + Q_{M-1}(z^M) \]

Type 3 polyphase decomposition Given  
\[ \frac{Y(z)}{X(z)} = H(z) = P_0(z^M) + z^{-1} P_1(z^M) + \ldots + z^{-(M-1)} P_{M-1}(z^M) = \sum_{m=0}^{M-1} z^{-m} P_m(z^M) \]
set \( m = -k \): as \( m \) goes from 0 to \( M-1 \), \( k \) goes from 0 to \(-(M-1)\) to 0. Thus  
\[ H(z) = \sum_{k=0}^{-(M-1)} z^{k} P_{-k}(z^M) \]
Let  
\[ R_k(z^M) = z^{k} P_{-k}(z^M) \]
Example 7.8.2 For the system
\[ H(z) = \sum_{n=0}^{k} h(n) z^{-n} = h(0) + h(1) z^{-1} + h(2) z^{-2} + h(3) z^{-3} + h(4) z^{-4} + h(5) z^{-5} + h(6) z^{-6} \]
\[ = \{ h_0 + h_2 z^{-2} + h_4 z^{-4} + h_6 z^{-6} \} + \{ h_1 z^{-1} + h_3 z^{-3} + h_5 z^{-5} \} \]
\[ = \{ h_0 + h_2 z^{-2} + h_4 z^{-4} + h_6 z^{-6} \} + z^{-1} \{ h_1 + h_3 z^{-2} + h_5 z^{-4} \} \]
\[ P_0(z) = h_0 + h_2 z^{-1} + h_4 z^{-2} + h_6 z^{-3} \]
\[ P_1(z) = h_1 + h_3 z^{-1} + h_5 z^{-2} \]
The structure is shown below (left). As mentioned earlier \( P_0(z^2) \) and \( P_1(z^2) \) can each be implemented as a direct form.

Polyphase realization

The structure shown on the right is obtained by moving the delay element \( z^{-1} \) to the right of \( P_1(z^2) \) these two being in series in the second phase. In this latter case the two systems \( P_0(z^2) \) and \( P_1(z^2) \) can share the same delay elements (that is, storage locations) even though each has its own set of coefficients, thus resulting in a canonical polyphase realization, shown below.
**Polyphase Structure for IIR Filters** The anti-aliasing filter in a decimator and the anti-imaging filter in an interpolator may each be either an FIR or an IIR filter. The polyphase structure can be developed for any filter, FIR or IIR, and any finite value of \( M \). We now proceed to the case where \( h(n) \) is an infinitely long sequence:

\[
H(z) = \sum_{n=-\infty}^{\infty} h(n) z^{-n} = \ldots + h(-M) z^M + \ldots + h(-2) z^2 + h(-1) z^1 + h(0) + h(1) z^{-1} + h(2) z^{-2} + \ldots + h(M-1) z^{-(M-1)} + h(M) z^{-M} + h(M+1) z^{-(M+1)} + \ldots
\]

Once again we arrange the terms into \( M \) groups or branches in a modulo-\( M \) fashion. Each branch contains infinitely many terms. The terms are arranged in tabular form below:

<table>
<thead>
<tr>
<th>Branch</th>
<th>Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>1(^{st})</td>
<td>( + h(-M) z^M ) \quad ( + h(0) ) \quad ( + h(M) z^{-M} )</td>
</tr>
<tr>
<td>2(^{nd})</td>
<td>( + h(-M+1) z^{M-1} ) \quad ( + h(1) z^{-1} ) \quad ( + h(M+1) z^{-M-1} )</td>
</tr>
<tr>
<td>(i)(^{th})</td>
<td>( + h(-M+i-1) z^{-(M+i-1)} ) \quad ( h(i-1) z^{-(i-1)} ) \quad ( h(M+i-1) z^{-(M+i-1)} )</td>
</tr>
<tr>
<td>(M)(^{th})</td>
<td>( + h(-1) z^1 ) \quad ( + h(M-1) z^{-(M-1)} ) \quad ( + h(M-2) z^{-(M-2)} )</td>
</tr>
</tbody>
</table>

\[
H(z) = [\ldots + h(-M) z^M + h(0) + h(M) z^{-M} + h(2M) z^{-2M} + \ldots] + [\ldots + h(-M+1) z^{M-1} + h(1) z^{-1} + h(M+1) z^{-M-1} + \ldots] + [\ldots + h(-M+i-1) z^{-(M+i-1)} + h(i-1) z^{-(i-1)} + h(M+i-1) z^{-(M+i-1)} + \ldots] + [\ldots + h(-2) z^2 + h(M-2) z^{-(M-2)} + \ldots] + [\ldots + h(-1) z^1 + h(M-1) z^{-(M-1)} + h(2M-1) z^{-(2M-1)} + \ldots] + \ldots
\]

We factor out \( z^0 \) from the first row (branch), \( z^{-1} \) from the 2\(^{nd}\) row, and, in general, \( z^{-(i-1)} \) from the \( i\)\(^{th}\) row to get:

\[
H(z) = [\ldots + h(-M) z^M + h(0) + h(M) z^{-M} + h(2M) z^{-2M} + \ldots] + z^{-1} [\ldots + h(-M+1) z^{M-1} + h(1) z^{-M-1} + \ldots] + z^{-2} [\ldots + h(-M+i-1) z^{-(M+i-1)} + h(i-1) z^{-M-1} + \ldots] + \ldots
\]

Define

\[
z^{-M-1} \left[ \ldots + h(-2) z^M + h(M-2) + h(2M-2) z^{-(M-2)} + \ldots \right] + z^{-M-1} \left[ \ldots + h(-1) z^M + h(M-1) + h(2M-1) z^{-M} + \ldots \right]
\]
\[
P_0(z^M) = \ldots + h(-M)z^M + h(0) + h(M)z^{-M} + h(2M)z^{-2M} + \ldots = \sum_{n=-\infty}^{\infty} h(nM)z^{-nM}
\]

so that
\[
P_0(z) = \ldots + h(-M)z + h(0) + h(M)z^{-1} + h(2M)z^{-2} + \ldots = \sum_{n=-\infty}^{\infty} h(nM)z^{-n}
\]

Similarly,
\[
P_1(z) = \ldots + h(-M+1)z + h(1) + h(M+1)z^{-1} + h(2M+1)z^{-2} + \ldots = \sum_{n=-\infty}^{\infty} h(nM+1)z^{-n}
\]

In general
\[
P_n(z) = \sum_{n=-\infty}^{\infty} h(nM+m)z^{-n} , \quad m = 0 \text{ to } (M-1)
\]

And the system function can now be written
\[
H(z) = \sum_{m=0}^{M-1} z^{-m} P_n(z^M)
\]

This is called the \(M\)-component polyphase decomposition of \(H(z)\). The \(M\) functions \(P_n(z)\) are the polyphase components of \(H(z)\). This overall operation is known as polyphase filtering.

As an example, for \(M = 3\), we have
\[
\frac{Y(z)}{X(z)} = H(z) = \sum_{n=0}^{2} z^{-n} P_n(z^3) = P_0(z^3) + z^{-1} P_1(z^3) + z^{-2} P_2(z^3)
\]

\[
Y(z) = P_0(z^3) X(z) + z^{-1} P_1(z^3) X(z) + z^{-2} P_2(z^3) X(z)
\]

This last equation leads to the structure below (left):

We may also rearrange the output equation as
\[
Y(z) = P_0(z^3) X(z) + z^{-1} P_1(z^3) X(z) + z^{-2} P_2(z^3) X(z)
\]

\[
= P_0(z^3) X(z) + z^{-1} \{ P_1(z^3) X(z) + z^{-1} P_2(z^3) X(z) \}
\]

This last equation leads to the polyphase structure shown above (right), known as the transpose polyphase structure because it is similar to the transpose FIR filter realization.
7.9 Polyphase structure for a decimator

The decimator block diagram is shown below: it consists of an anti-aliasing filter, \( H(z) \), which could be an FIR or an IIR filter, followed by an \( M \)-fold down sampler.

We replace the filter \( H(z) \) by its \( M \)-component polyphase decomposition

\[
H(z) = \sum_{m=0}^{M-1} z^{-m} P_m(z^M)
\]

The sub filters \( P_0(z^M), P_1(z^M), \ldots, P_{M-1}(z^M) \) could be FIR or IIR depending on \( H(z) \). The block diagram then appears as below.
We may use identity #1 to move the down-sampler to the immediate right of $P_m(z^M)$ in each branch, and then use identity #3 to move the down-sampler from the immediate right to the immediate left of $P_m(\cdot)$ while at the same time changing $P_m(z^M)$ to $P_m(z)$. The result appears as below. It can be seen from this diagram that in this structure the number of multiplications is reduced by a factor of $M$.

**** Continuing with the development, comparing

$$P_m(z) = \sum_{r=-\infty}^{\infty} h(rM + m) z^{-r}$$

with the defining equation of the $z$-transform

$$P_m(z) = \sum_{r=-\infty}^{\infty} p_m(r) z^{-r}$$

we identify

$$p_m(r) = h(rM + m)$$

Note that $M$ is a constant and $m$ is a parameter. Upon substituting for $P_m(z)$ in the system function

$$H(z) = \sum_{m=-M}^{M-1} z^{-m} P_m(z^M)$$

we have
\[
H(z) = \sum_{m=0}^{M-1} z^{-m} \sum_{r=-\infty}^{\infty} h(rM + m) \left( z^{M} \right)^{-r} = \sum_{m=0}^{M-1} \sum_{r=-\infty}^{\infty} h(rM + m) z^{-(rM + m)}
\]

The output transform is

\[
Y(z) = H(z)X(z) = \sum_{m=0}^{M-1} \sum_{r=-\infty}^{\infty} p_m(r) X(z)z^{-(rM + m)}
\]

The output is

\[
y(n) = z^{-1} \left\{ \sum_{m=0}^{M-1} \sum_{r=-\infty}^{\infty} p_m(r) X(z)z^{-(rM + m)} \right\} = \sum_{m=0}^{M-1} \sum_{r=-\infty}^{\infty} p_m(r) z^{-(n-rM + m)}
\]

Define

\[
x_m(r) = x(rM - m)
\]

Note that \(M\) is a constant and \(m\) is a parameter.

\[
x_m(-r) = x(-rM - m)
\]

Shifting the sequence by \(n\) units we get

\[
x_m(n-r) = x(n-rM - m)
\]

The output now may be written

\[
y(n) = \sum_{m=0}^{M-1} \sum_{r=-\infty}^{\infty} p_m(r) x_m(n-r)
\]

Define the operation of polyphase convolution as

\[
y(m) = p_m(n) * x_m(n) = \sum_{r=-\infty}^{\infty} p_m(r) x_m(n-r)
\]

Then

\[
y(n) = \sum_{m=0}^{M-1} p_m(n) * x_m(n) = \sum_{m=0}^{M-1} y_m(n)
\]

This overall operation is known as polyphase filtering.