Multirate Digital Signal Processing

Textbook pages 768-784

Dr D. Laurenson

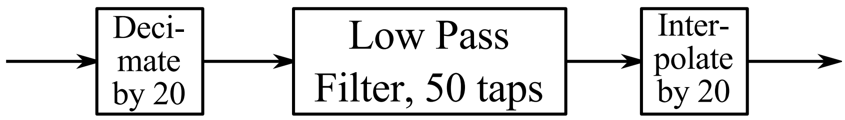
10th November 2020

Introduction

Motivation

Sampled data systems represent data at a fixed sampling rate. However, when interacting with other sampled data systems, or when carrying out particular processing tasks, it is often necessary to change the rate with which data is sampled. Consider, for example, applying a low pass filter with a narrow transition band. The filter length will be long, involving many multiplications, and the output of the filter is contained solely at low frequencies. If the data rate is lowered, taking care to avoid aliasing of the filter output, then the number of multiplications required in the filter design will be reduced. Following this processing, subsequent systems may expect the input to be presented at the original sampling rate, so the filter output should be converted to this higher sampling rate. An example of this is shown below where the sampling rate is lowered by a factor of 20 prior to low pass filtering the signal.

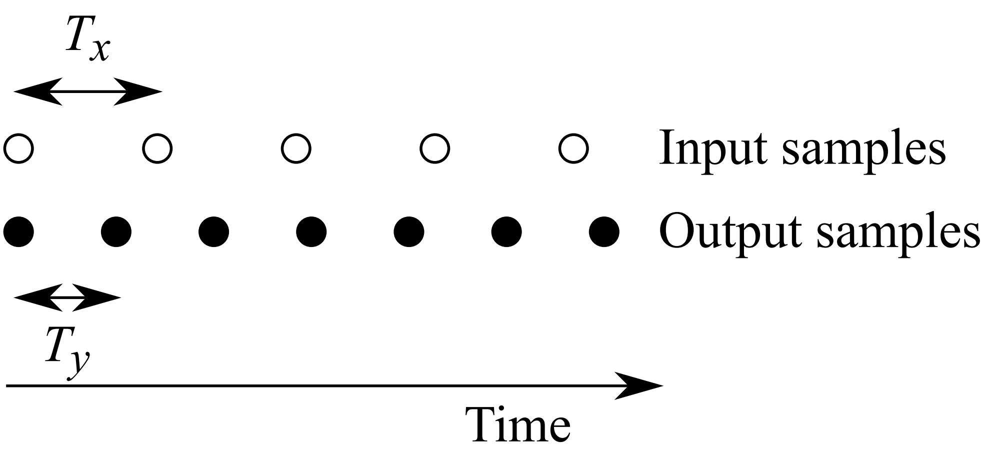
System without sample rate conversion



System using sample rate conversion

Resampling the original analogue signal at a different rate would be costly, and in many cases would not be possible, particularly if any processing has been carried out on the signal in the digital domain.

To deal with such problems, multirate techniques are required. These techniques generate a signal, with a new sampling rate, that represents the original signal.



Note that particularly when increasing the sampling rate (upsampling) this is not necessarily the same as resampling the original analogue signal as the initial sampling may have excluded frequency content due to the anti-aliasing filter.

Changing sampling rate of a signal can be considered as a process of signal reconstruction, followed by a sampling of the reconstructed signal. In practice the two stages are combined into one. A signal sampled at a rate of F_x is periodic in the frequency domain with period F_x. If the periodic repetitions are removed in the frequency domain, then the signal is no longer sampled in the time domain.

Therefore, to reconstruct the signal from the samples, apply the ideal filter:

$$G(F) = \begin{cases} 1/F_x & ;|F| \leq F_x/2 \\ 0 & ;\text{otherwise} \end{cases}$$

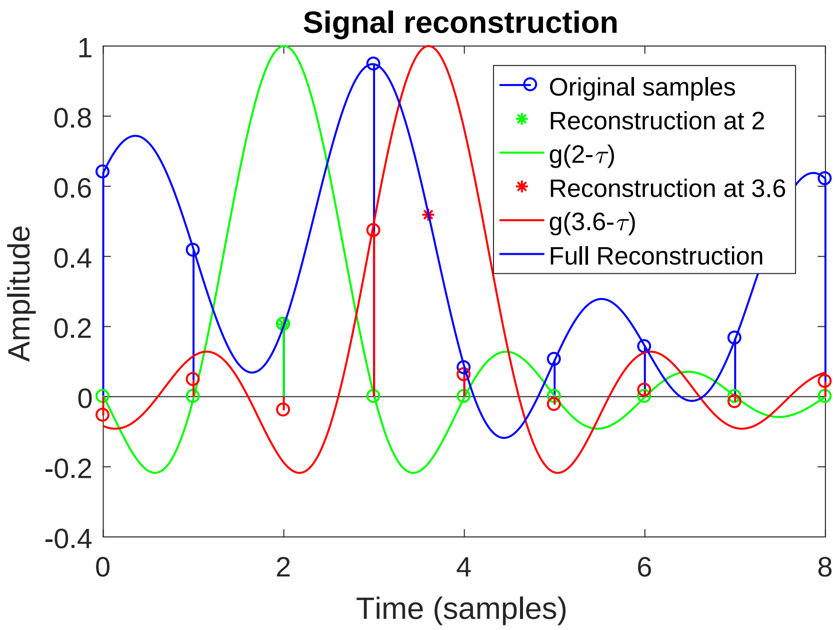
This is equivalent to convolving the sampled signal with

$$g(t) = \frac{\sin(\pi t F_x)}{\pi t F_x}$$

Note that this is not the same as applying a "hold" on the sampled values, where the continuous time representation is simply the sampled values extended across the sampling interval. Instead, the ideal filter removes all of the periodic repetitions in the frequency domain that correspond to the sampling process in the time domain. Removal of these periodic images results in a continuous time domain that corresponds to the frequency content in the range $(-F_x/2, F_x/2)$.

The ideal filter to reconstruct would be a brick-wall filter, $G(\omega)$, with its corresponding impulse response, $g(t)$. Letting $T_x = 1/F_x$, the perfect reconstruction is given by

$$y(t) = \sum_{n=-\infty}^{\infty} x(nT_x)g(t - nT_x)$$



Reconstruction of a signal is the first step in defining the process to allow us to re-sampled a signal at a new sampling rate F_y. This forms the basis of multi-rate systems.

Resampling

Let $T_y = 1/F_y$, then:

$$y(mT_y) = \sum_{n=-\infty}^{\infty} x(nT_x)g(mT_y - nT_x)$$

Note that to avoid aliasing, if $F_y < F_x$, $G(F)$ should be changed, such that

$$g(t) = \frac{\sin(\pi t F_y)}{\pi t F_y}$$

This last step is necessary when downsampling (i.e. $F_y < F_x$), as otherwise the re-sampled signal will contain aliased components, with frequency terms above F_y appearing as different frequencies below F_y. Thus, potentially, information contained in the original sampled signal will be lost when it is resampled.

Unfortunately, for arbitrary F_x and F_y, the process is a time varying one as $g()$ depends on the fractional spacing of samples from one sampling rate to the other. Without any constraints on the ratio of the two frequencies, an infinite number of filter designs will be required for $g()$, each design being used only once for a single output value.

To see why, consider the sampling points of $y(t)$ for an irrational ratio of sampling rates. Each new sample point is formed from a sum of original sampled values, weighted by a unique set of coefficients, $g(t - nT_x)$. The textbook lays out this derivation, but it is not required to be able to reproduce this as part of this course.

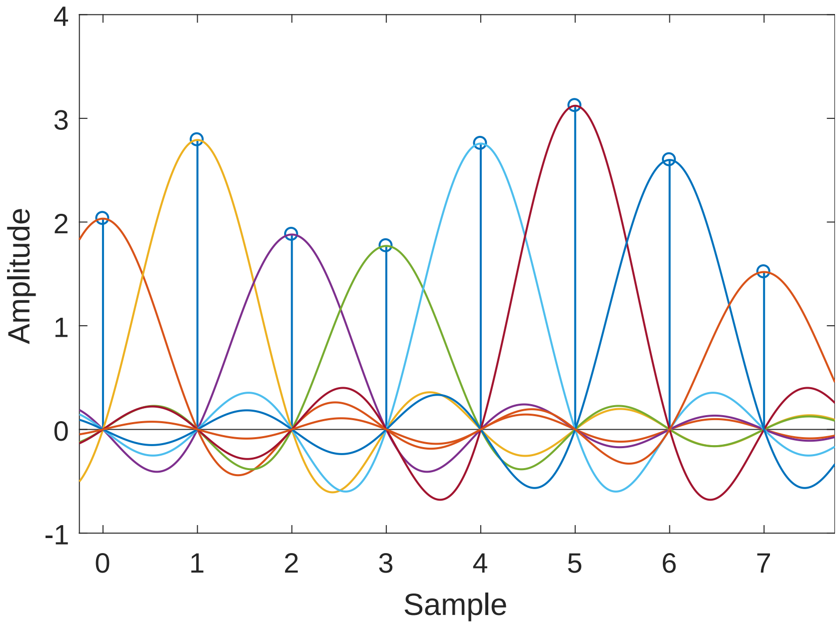
Instead, in practice where m and n represent a finite number of values, it would be much better to have a finite number of possible values for $mT_y - nT_x$. Then only a finite number of $g()$ coefficients are required, and one or more filters can be designed to represent all possible combinations. When

$$\frac{T_y}{T_x} = \frac{F_x}{F_y} = \frac{D}{I}$$

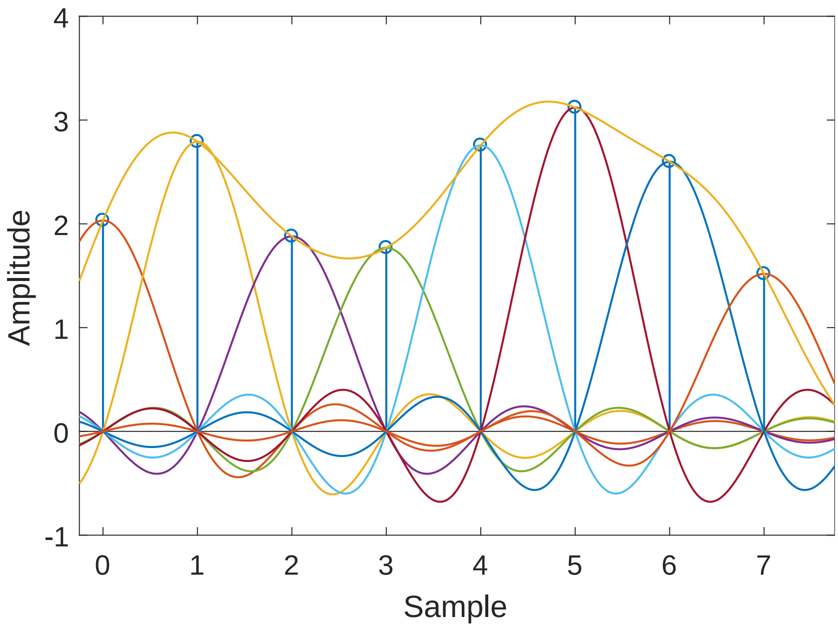
and D and I are integers, then it is possible to define a linear system, with I interpolation functions, $g_m()$, used in turn periodically. The following example illustrates the case where $\frac{D}{I} = \frac{3}{2}$.

Example

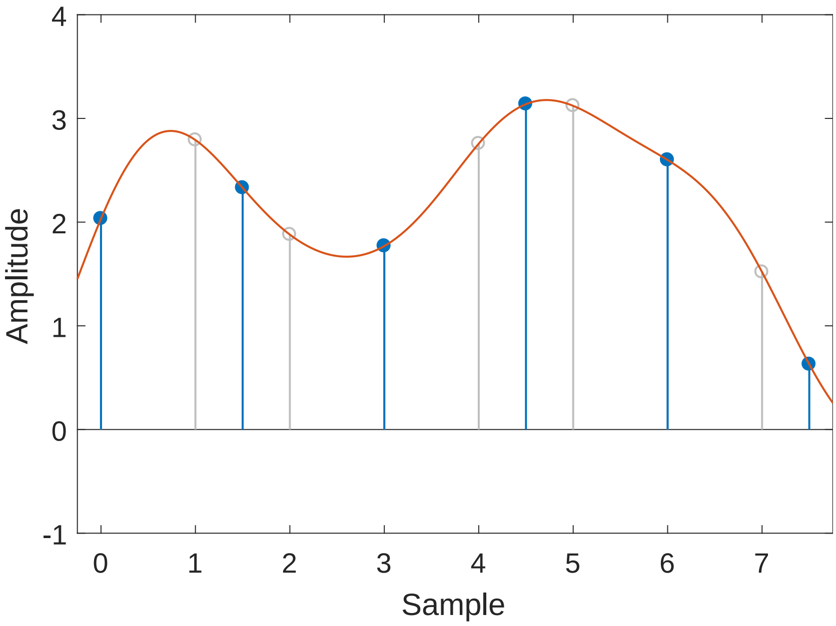
First the interpolation filter is convolved with each of the samples of the data in turn:



Then the reconstruction is the sum of all of the interpolation filter responses:



As our resampling rate is 3/2, the first, fourth and seventh samples (n=0, 3 and 6) will remain as they currently are. All interpolation filter responses are zero at these samples, except for the filter that corresponds to the sample itself. So when summed, the same value is returned. The other samples are at positions n=1.5 and 4.5. It is clear that for these positions, the interpolation filter values at $g((m+0.5)T_x)$ are required. The textbook describes this as the problem requiring two filters, used alternately for each successive output sample. Finally, the reconstruction can be resampled:



Unfortunately, the construction of the ideal filter is not possible:

- An infinite number of samples of $x(n)$ are required
- The process requires future values of the sampled signal
- Only frequencies below $F_x/2$ will be present

Despite the issues with this process, the technique can be made practical by using a non-ideal filter for $G(\omega)$. The design techniques of module 6 may be used to generate the filters, $g()$, so this will not be covered again in this module. We will now consider the specific cases of downsampling only ($I=1$), and upsampling only ($D=1$) in turn before combining them.

Decimation by a factor, D

Decimation is the term used to describe reducing the data rate, or downsampling, by a factor. Here we consider only positive integer values of D.

Decimation by a factor of D results in:

$$y(mT_y) = y(mDT_x) = \sum_{k=-\infty}^{\infty} x(kT_x)g((mD-k)T_x) \tag{11.1.14}$$

As D is an integer, $mD-k$ is an integer. $g(nTx)$ is zero for all integer n except for $n = 0$ where it has the value 1. Thus the equation implies that $y(mT_y) = x(mDT_x)$, however this is an oversimplification of the problem.

As $F_y < F_x$, it is not possible to just take every D^{th} sample as aliasing would occur. A digital anti-alias filter is required:

$$H_D(\omega) = \begin{cases} 1 & |\omega| \le \pi/D \\ 0 & \text{otherwise} \end{cases} \tag{11.2.1}$$

The filter can be implemented as an FIR, $h(n)$:[1]

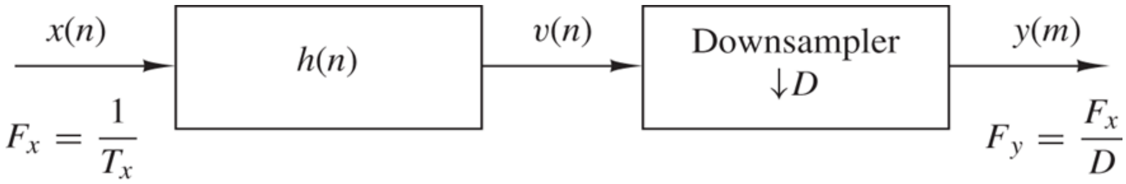


Figure 11.2.1 Decimation by a factor D.

The output of the filter, $v(n)$, is $h(n)$ convolved with $x(n)$:

$$v(n) = \sum_{k=0}^{\infty} h(k)x(n-k) \tag{11.2.2}$$

then downsampling gives:

$$y(m) = v(mD) = \sum_{k=0}^{\infty} h(k)x(mD-k) \tag{11.2.3}$$

Following the anti-alias filter stage, the downsampling can be viewed as a two step process - first, setting every sample, except for every D^{th} sample, to zero to obtain $\tilde{v}(n)$, and then subsampling the resulting signal by a factor of D.

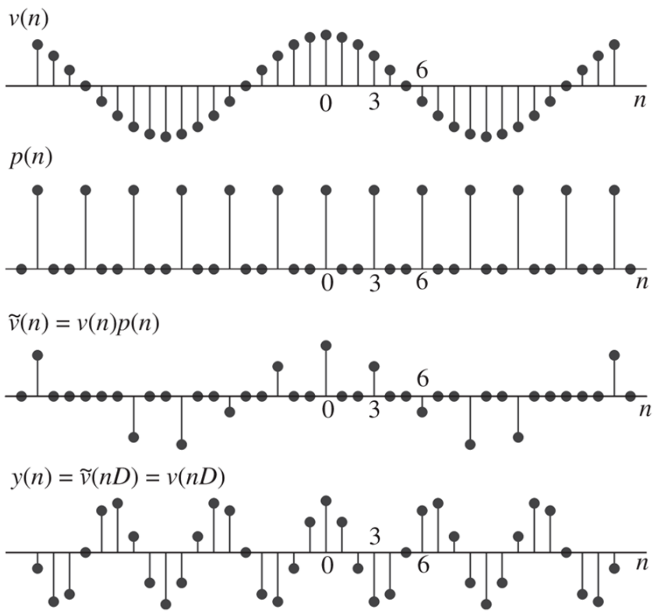
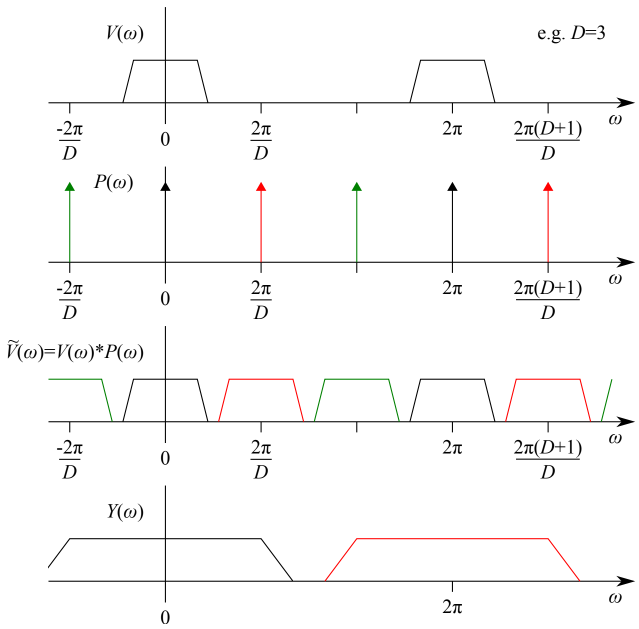


Figure 11.2.2 Steps required to facilitate the mathematical description of downsampling by a factor D, using a sinusoidal sequence for illustration.

[1] Selected figures taken from "Digital Signal Processing, New International Edition/4th", Proakis & Manolakis, ©Pearson Education Limited, 2014. ISBN: 978-1-29202-573-5

It is also helpful to view the signals in the frequency domain:



Demonstrating that the transform of a time-domain periodic pulse train is an impulse train in the frequency domain is simplified if the inverse transform is considered. Starting with the impulse train in the frequency domain:

$$
\begin{aligned}
P(\omega) &= \sum_k 2\pi\delta\left(\omega - \frac{2\pi k}{D}\right) \\
x(n) &= \frac{1}{2\pi}\int_0^{2\pi} X(\omega)e^{jn\omega}d\omega \\
&= \sum_k \int_0^{2\pi} \delta\left(\omega - \frac{2\pi k}{D}\right)e^{jn\omega}d\omega \\
&= \sum_{k=0}^{D-1} e^{j\frac{2\pi kn}{D}} = \frac{1-e^{j2\pi n}}{1-e^{\frac{j2\pi n}{D}}}
\end{aligned}
$$

As $e^{j2\pi n} = 1$ for integer n, we need to use De L'Hôpital's rule for $n = 0$:

$$
x(n) = \begin{cases} D & n = 0 \\ 0 & \text{otherwise} \end{cases}
$$

where $x(n)$ is periodic with period D.

The spectrum is altered, with the new sampling rate causing the frequencies between $\pm\pi/D$ to be expanded to cover the range $\pm\pi$.

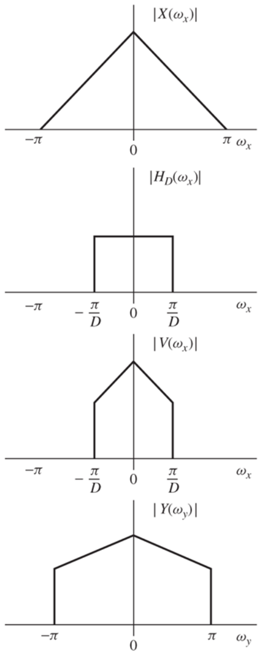
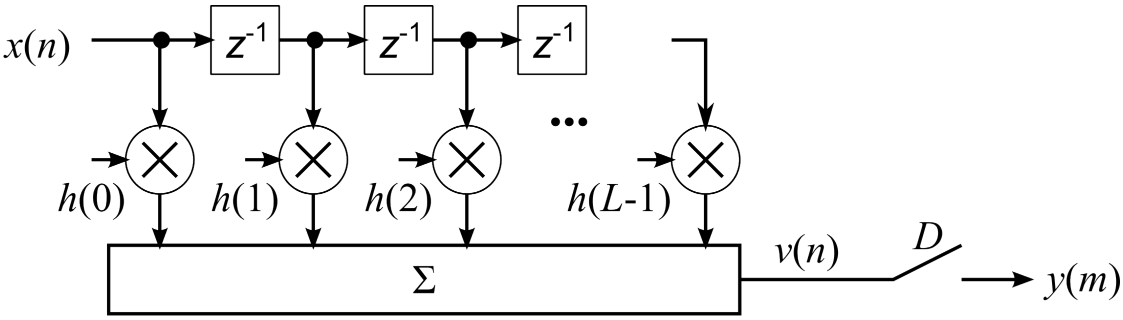
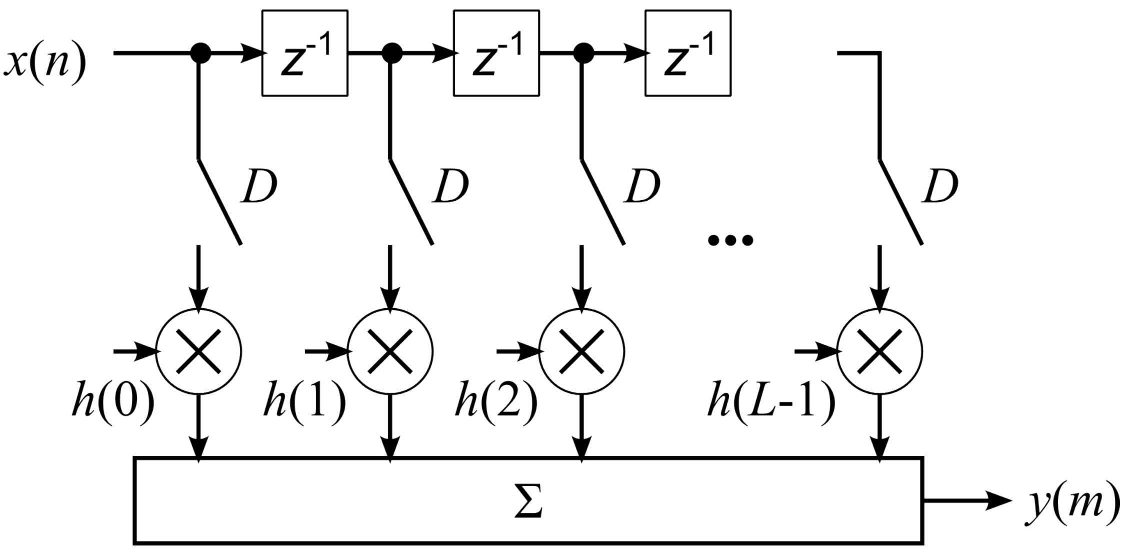


Figure 11.2.3 Spectra of signals in the decimation of $x(n)$ by a factor D.

In practice, an ideal filter cannot be constructed, and an approximation needs to be made to this. As many signal processing tasks are phase sensitive, a linear phase finite impulse response (FIR) filter is used to implement the anti-alias filter in the digital domain. The decimator can be constructed efficiently by noting that only one in D outputs is required. The structure of the decimator, using an FIR with L coefficients (taps), could be viewed as:



This is a direct representation of (11.2.2) followed by a sub sampling process. However, by re-expressing the process, as in (11.2.3), it is also possible to sample the output of the tapped delay line. The tapped delay line outputs form a vector, $\{x(mD-k)\}$ for $k \in \{0,1,\ldots L-1\}$, which is only required at the lower sampling rate. This is then multiplied by $h(k)$ and summed which results in the multiplication and summation operations needing to be performed at the lower rate of the output.



This is a much more efficient structure as it reduces the number of multiplication operations required by a factor of D, yet yields exactly the same result.

Interpolation by a Factor I

Interpolation is best understood by examining the problem from the frequency domain. Increasing sampling rate by a factor of I we can get:

$$v(m) = \begin{cases} x(m/I) & ; m = 0, \pm I, \pm 2I, \ldots \\ 0 & ; \text{otherwise} \end{cases} \tag{11.3.1}$$

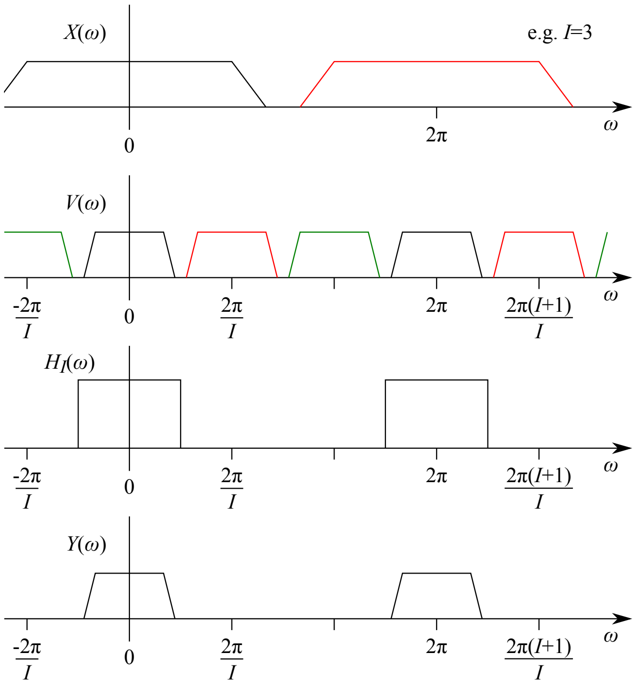
The missing samples are set to 0 initially. The spectrum of $v(m)$ is a compressed version of the spectrum of $x(m)$:

$$V(\omega_y) = X(\omega_y I) \tag{1}$$

It is periodic, as is $X(\omega)$, however it has a period of $2\pi/I$. This can be explained as follows: although there are more samples in the time domain, the new samples are all zeros. Thus, the frequency spectrum of the signals before and after adding the zeros, when measured in Hz, or rad/s, is the same. However, as we are using a normalised frequency axis, and the sampling rate has increased, then the periodic repetitions of the original signal's frequency content are compressed in the new spectrum.

We only want the spectrum in the range $\pm\pi/I$. The reason is that this period contains the frequency content of $x(m)$ within its Nyquist rate. Periodic repetitions in $X(\omega)$ correspond to the original sampling rate.

At the new rate, the effect of the old sampling rate can be seen in the zero values of $v(m)$, as these represent the spaces between the samples of $x(m)$. By filtering out the *extra* $I-1$ repetitions within $\pm\pi$ of $V(\omega)$, the zero values of $v(m)$ then alter to represent the band-limited signal.



To preserve energy, the interpolation filter includes a scale factor:

$$H_I(\omega_y) = \begin{cases} I & ; 0 \leq |\omega_y| \leq \pi/I \\ 0 & ; \text{otherwise} \end{cases}$$

Then:

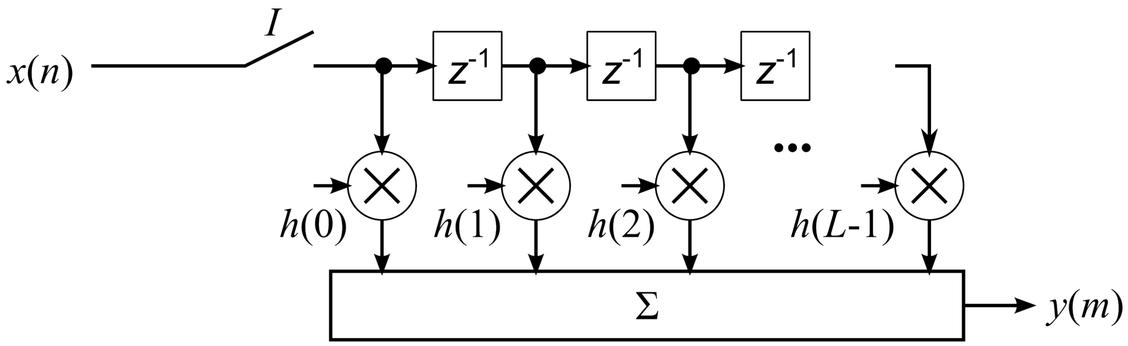
$$y(m) = \sum_{-\infty}^{\infty} h(m-k)v(k) \tag{11.3.9}$$

or:

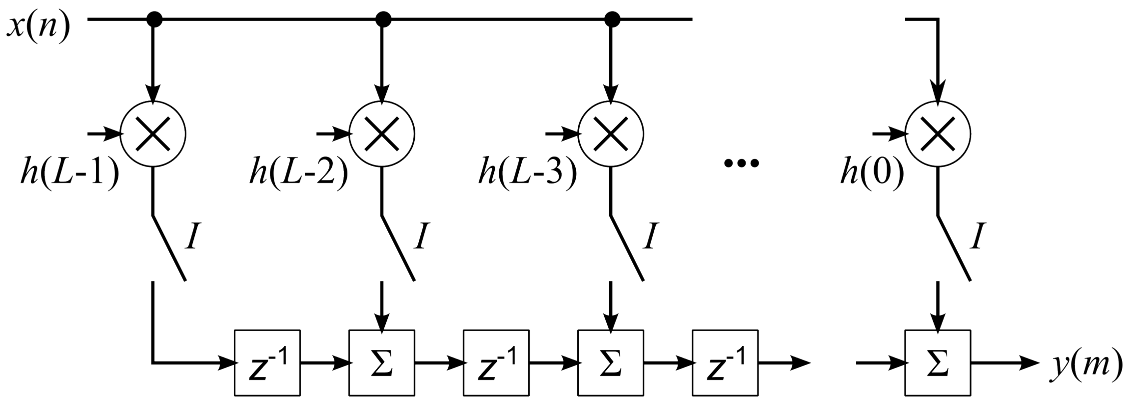
$$y(m) = \sum_{-\infty}^{\infty} h(m-kI)x(k) \tag{11.3.10}$$

Interpolator structure

By examining the equations carefully, an efficient implementation of the upsampler can also be created. First, as specified in (11.3.9):



Then by re-ordering the filter operations and resampling, (11.3.10) is obtained, and can be constructed as:



Here, as for the downsampler, the order of the rate changing component and the filtering are switched. In this case it is because $I-1$ out of every I samples is zero, so there is no point in performing the multiplication. Instead, the multiplications can be carried out at the lower rate, and then the delay line structure used to combine them appropriately.

1 Rational factor I/D

Resampling by a fractional sampling rate, I/D, where I and D are positive integers, is a combination of interpolation and decimation. It is important to interpolate first to avoid causing unnecessary aliasing that would occur if decimation were placed first.

Fractional rate resampling

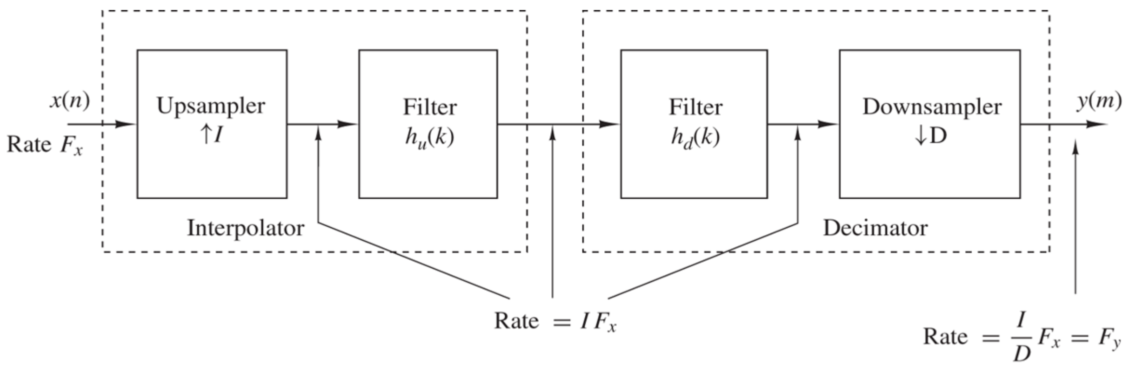


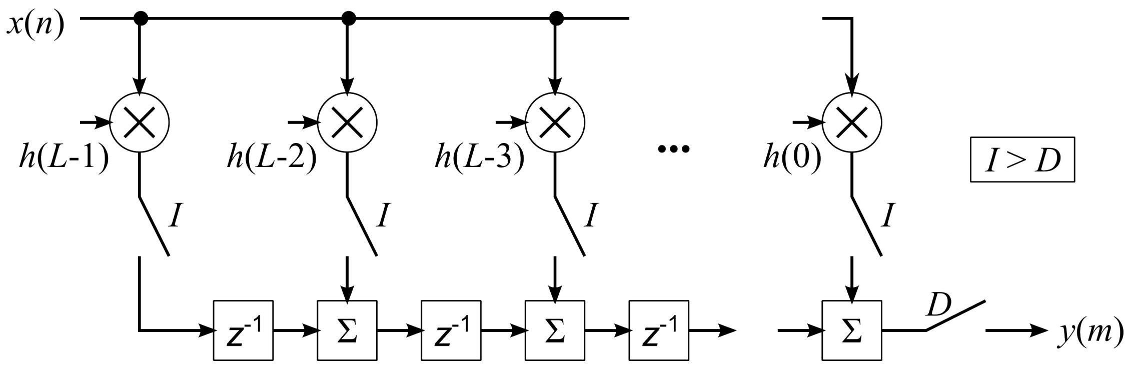
Figure 11.4.1 Method for sampling rate conversion by a factor I/D.

Both interpolation and decimation involve filtering - the first to remove images and the second to avoid aliasing. Viewed in another way, the two are essentially performing the same task of bandlimiting the signal. This implies that they can be performed by a single filtering operation. $h_u(k)$ and $h_d(k)$ can be performed by a single filter $h(k)$ defined as an approximation to:

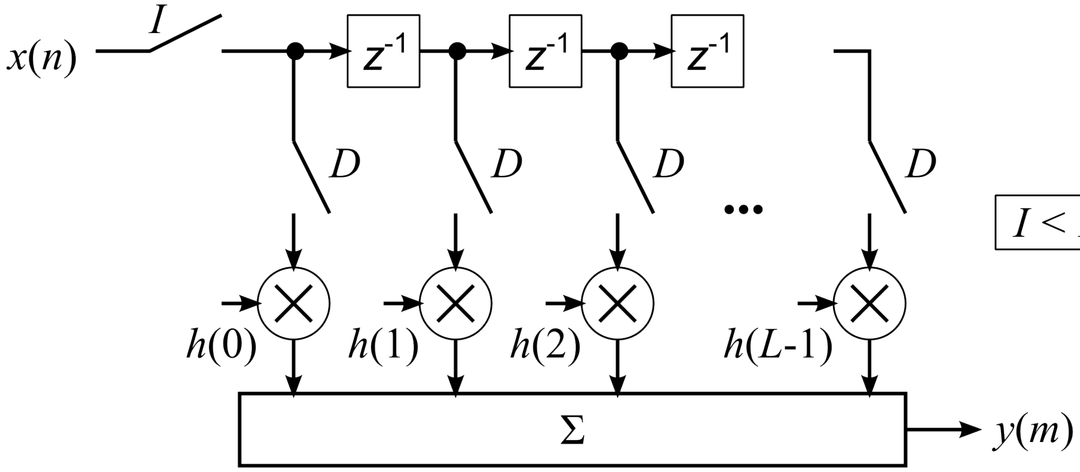
$$H(\omega_v) = \begin{cases} I & ;0 \le |\omega_v| \le \min(\pi/D, \pi/I) \\ 0 & ;\text{otherwise} \end{cases} \tag{11.4.1}$$

The filter is designed for a sampling frequency of IF_x, but will be operated at a lower rate.

This gives rise to two different structures, dependent upon whether the target rate is higher or lower than the original rate. In both structures, the aim is to operate the filter at the lowest of the two rates:



In this first instance, as the ratio results in an upconversion, unwanted images are removed by the filter, and its location corresponds to the interpolating filter structure.



In this second instance, the ratio results in a downconversion, so here the purpose of the filter is to avoid aliasing. This is located in the same place as for a decimator.