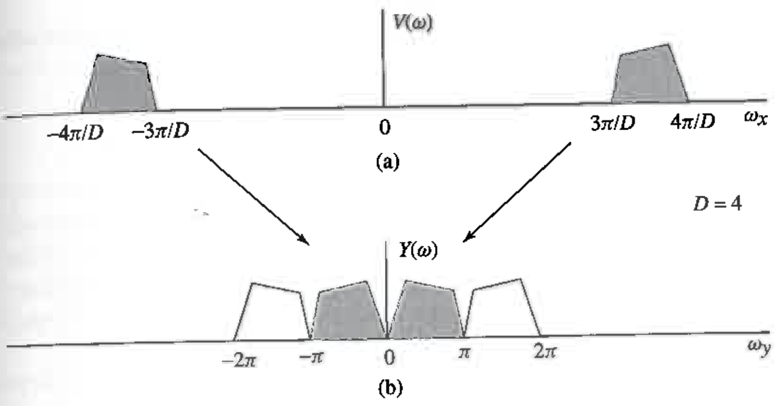

Figure 11.7.2 Spectral interpretation of bandpass decimation for integer-band positioning (even integer positioning).

interest. Note that this approach provides us with a modulation-free method for achieving frequency translation of a signal by selecting $D = I$.

11.8 Sampling Rate Conversion by an Arbitrary Factor

Efficient implementation of sampling rate conversion by a polyphase structure requires that the rates F_x and F_y are fixed and related by a rational factor I/D. In some applications, it is either inefficient or sometimes impossible to use such an exact rate conversion scheme.

For example, suppose we need to perform rate conversion by the rational number I/D, where I is a large integer (e.g., $I/D = 1023/511$). Although we can achieve exact rate conversion by this number, we would need a polyphase filter with 1023 subfilters. Such an implementation is obviously inefficient in memory usage because we need to store a large number of filter coefficients.

In some applications, the exact conversion rate is not known when we design the rate converter, or the rate is continuously changing during the conversion process. For example, we may encounter the situation where the input and output samples are controlled by two independent clocks. Even though it is still possible to define a nominal conversion rate that is a rational number, the actual rate would be slightly different, depending on the frequency difference between the two clocks. Obviously, it is not possible to design an exact converter in this case.

In principle, we can convert from any rate F_x to any rate F_y (fixed or variable) using formula (11.1.9), which we repeat here for convenience:

$$y(mT_y) = \sum_{k=K_1}^{K_2} g(kT_x + \Delta_m T_x)x((k_m - k)T_x) \tag{11.8.1}$$

This requires the computation of a new impulse response $p_m(k) = g(kT_x + \Delta_m T_x)$ for each output sample. However, if Δ_m is measured with finite accuracy, there is only a finite set of impulse responses, which may be precomputed and loaded from memory as needed. We next discuss two practical approaches for sampling rate conversion by an arbitrary factor.

11.8.1 Arbitrary Resampling with Polyphase Interpolators

If we use a polyphase interpolator with I subfilters we can generate samples with spacing T_x/I. Therefore, the number I of stages determines the granularity of the interpolating process. If T_x/I is sufficiently small so that successive values of the signal do not change significantly or the change is less than the quantization step, we can determine the value at any location $t = nT_x + \Delta T_x$, $0 \le \Delta \le 1$, using the value of the nearest neighbor (zero-order hold interpolation).

Additional improvement can be obtained using two-point linear interpolation

$$y(nT_x + \Delta T_x) = (1-\Delta)x(n) + \Delta x(n+1) \tag{11.8.2}$$

The performance of these interpolation techniques has been discussed in Section 6.5 by analyzing their frequency-domain characteristics. Additional practical details can be found in Ramstad (1984).

11.8.2 Arbitrary Resampling with Farrow Filter Structures

In practice, we typically implement polyphase sampling rate converters by a rational factor using causal FIR lowpass filters. If we use an FIR filter with $M = KI$ coefficients, the coefficients of the polyphase filters are obtained by the mapping

$$p_i(n) = h(nI + i), \quad i = 0, 1, \ldots, I-1 \tag{11.8.3}$$

This mapping can be easily visualized as mapping the one-dimensional sequence $h(n)$ into a two-dimensional array with I rows and K columns by filling successive columns in natural order as follows:

$$\begin{array}{llllll}
p_0(k) & \mapsto h(0) & h(I) & \ldots & h((K-1)I) & \\
p_1(k) & \mapsto h(1) & h(I+1) & \ldots & h((K-1)I+1) & \\
 & \vdots & & & & \\
p_i(k) & \mapsto h(i) & h(I+i) & \ldots & h((K-1)I+i) & (11.8.4) \\
p_{i+1}(k) & \mapsto h(i+1) & h(I+i+1) & \ldots & & \\
 & \vdots & & & & \\
p_{I-1}(k) & \mapsto h(I-1) & h(2I-1) & \ldots & h(KI-1) &
\end{array}$$

The polyphase filters $p_i(n)$ are used to compute samples at I equidistant locations $t = nT_x + i(T_x/I)$, $i = 0, 1, \ldots, I-1$ covering each input sampling interval. Suppose now that we wish to compute a sample at $t = nT_x + \Delta T_x$, where $\Delta \ne i/I$ and $0 \le \Delta \le 1$. This requires a nonexisting polyphase subfilter, denoted as $p_\Delta(k)$, which would "fall" between two existing subfilters, say, $p_i(k)$ and $p_{i+1}(k)$. This set of coefficients would create a row between the rows with indexes i and $i+1$. We note that each column of (11.8.4) consists of a segment of I consecutive samples of

the impulse response $h(n)$ and covers one sampling interval T_x. Suppose next that we can approximate the coefficient set in each column by an L-degree polynomial

$$B_k(\Delta) = \sum_{\ell=0}^{L} b_\ell^{(k)} \Delta^\ell, \quad k = 0, 1, \ldots, K-1 \tag{11.8.5}$$

We note that evaluating (11.8.5) at $\Delta = i/I$ will provide the coefficients of $p_i(k)$ polyphase subfilter. The type of the polynomial (Lagrange, Chebyschev, etc.) and the order L can be chosen to avoid any performance degradation compared to the original filter $h(n)$. The sample at location $t = nT_x + \Delta T_x$ is determined by

$$y((n+\Delta)T_x) = \sum_{k=0}^{K-1} B_k(\Delta)x((n-k)T_x), \quad 0 \le \Delta \le 1 \tag{11.8.6}$$

where the required filter coefficients are computed using (11.8.5). If we substitute the polynomials (11.8.5) into the filtering formula (11.8.6) and we change the order of summations, we obtain

$$\begin{aligned}
y((n+\Delta)T_x) &= \sum_{k=0}^{K-1} \sum_{\ell=0}^{L} b_\ell^{(k)} \Delta^\ell x((n-k)T_x) \\
&= \sum_{\ell=0}^{L} \Delta^\ell \sum_{k=0}^{K-1} b_\ell^{(k)} x((n-k)T_x)
\end{aligned}$$

The last equation can be written as

$$y((n+\Delta)T_x) = \sum_{\ell=0}^{L} v(\ell)\Delta^\ell \tag{11.8.7}$$

where

$$v(\ell) = \sum_{k=0}^{K-1} b_\ell^{(k)} x((n-k)T_x), \quad \ell = 0, 1, \ldots, L \tag{11.8.8}$$

Equation (11.8.7) can be interpreted as a Taylor series representation of the output sequence, where the terms $v(\ell)$ are successive local derivatives determined from the input sequence. Relation (11.8.8) can be implemented using FIR filtering structures with system functions

$$H_\ell(z) = \sum_{k=0}^{K-1} b_\ell^{(k)} z^{-k} \tag{11.8.9}$$

The most efficient computation of polynomial (11.8.7) can be done using the nested Horner's rule, which is illustrated next for $L = 4$:

$$\begin{aligned}
y(\Delta) &= c_0 + c_1\Delta + c_2\Delta^2 + c_3\Delta^3 + c_4\Delta^4 \\
&= c_0 + \Delta(c_1 + \Delta(c_2 + \Delta(c_3 + \Delta c_4)))
\end{aligned} \tag{11.8.10}$$

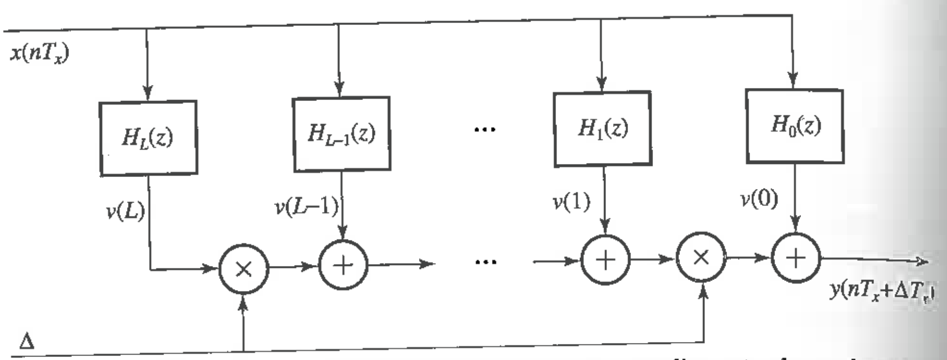

Figure 11.8.1 Block diagram of the Farrow structure for sampling rate change by an arbitrary factor.

This approach leads to the block diagram realization shown in Figure 11.8.1, which is known as Farrow structure (Farrow 1988). Basically, the Farrow structure performs interpolation between signal values by interpolating between filter coefficients. More details can be found in Gardner (1993), Erup et al. (1993), Ramstad (1984), Harris (1997), and Laakso et al. (1996).

11.9 Applications of Multirate Signal Processing

There are numerous practical applications of multirate signal processing. In this section we describe a few of these applications.

11.9.1 Design of Phase Shifters

Suppose that we wish to design a network that delays the signal $x(n)$ by a fraction of a sample. Let us assume that the delay is a rational fraction of a sampling interval T_x [i.e., $d = (k/I)T_x$, where k and I are relatively prime positive integers]. In the frequency domain, the delay corresponds to a linear phase shift of the form

$$\Theta(\omega) = -\frac{k\omega}{I} \tag{11.9.1}$$

The design of an all-pass linear-phase filter is relatively difficult. However, we can use the methods of sample-rate conversion to achieve a delay of $(k/I)T_x$, exactly, without introducing any significant distortion in the signal. To be specific, let us consider the system shown in Fig. 11.9.1. The sampling rate is increased by a factor I using a standard interpolator. The lowpass filter eliminates the images in the spectrum of the interpolated signal, and its output is delayed by k samples at the sampling rate IF_x. The delayed signal is decimated by a factor $D = I$. Thus we have achieved the desired delay of $(k/I)T_x$.

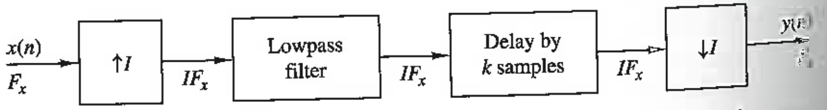

Figure 11.9.1 Method for generating a delay in a discrete-time signal.