3.3 Maximally Flat FD FIR Filter: Lagrange Interpolation

A useful FIR filter approximation for the fractional delay (FD) is obtained by setting the error function and its N derivatives to zero at zero frequency. This is the *maximally flat* (MF) design at $\omega = 0$. It is interesting to notice that the FIR filter coefficients obtained by this method are the same as the weighting coefficients in the classical *Lagrange interpolation.*

It appears that already in the 1950s Lagrange interpolation was used for recovering an analog bandlimited signal from its samples (see Jerri, 1977). For some reason this approach never gained much popularity and it is not described in basic text books. Later, Lagrange interpolation has been used for increasing the sampling rate of signals and systems (see, e.g., Schafer and Rabiner, 1973; Oetken, 1979).

To our knowledge, Lagrange interpolation was first used for fractional delay approximation by Strube (1975) who derived it using the Taylor series approach. He did not, however, notice that he was actually using the Lagrangian interpolation technique. Laine (1988) applied Lagrange interpolation for FD approximation and observed its maximally flat property but he did not give the mathematical derivation. The MF design of an FD FIR filter has been independently proposed by Ko and Lim (1988) and Sivanand *et al.* (1991). Liu and Wei (1990, 1992) derived the FD FIR filter approximation letting an Nth-order polynomial pass through $N + 1$ equidistant signal values. This approach is often used for deriving the classical Lagrange interpolation formula (see Section 3.3.2), but it does not reflect the frequency-domain properties of the technique. Liu and Wei give the solution for even-order Lagrange interpolation only.

Recently, Hermanowicz (1992) pointed out the equivalence of the MF FD approximation and Lagrange interpolation. This fact was already stated by Oetken in 1979, but since his paper addressed the design of FIR interpolators for sampling-rate conversion, it was supposedly not known among those who studied FD approximation. Minocha, Dutta Roy, and Kumar (1993) demonstrated that the Lagrange interpolation formula may also be derived by truncating the Taylor expansion of the error function. Kootsookos and Williamson (1995) have shown that Lagrange interpolation is also obtained by windowing the impulse response of the ideal bandlimited interpolator, i.e., the sinc function, using a *scaled binomial window* (see Section 3.3.5).

Below we derive the explicit formula for the coefficients of the MF approximation in a manner that is familiar in digital filter design and describe the relationship of this technique to the basic idea applied in Lagrange interpolation. We also discuss the properties of this FD filter in the frequency domain and introduce a novel structure for implementing Lagrange interpolation. Two related polynomial approximation techniques are considered as potential solutions for the FD approximation problem.

3.3.1 Derivation of the Maximally Flat Fractional Delay FIR Filter

The error function E and its N derivatives are first set to zero at a frequency ω_0, or

$$\left.\frac{d^k E(e^{j\omega})}{d\omega^k}\right|_{\omega=\omega_0} = 0 \quad \text{for } k = 0,1,2,\ldots,N \tag{3.54}$$

where $E(e^{j\omega})$ is the complex error function defined by Eq. (3.27). Equation (3.54) can thus be written as

$$\frac{d^k}{d\omega^k}\left[\sum_{n=0}^{N} h(n)e^{-j\omega n} - e^{-j\omega D}\right]\Bigg|_{\omega=\omega_0} = 0 \quad \text{for } k = 0,1,2,\ldots,N \tag{3.55}$$

We proceed by differentiating and setting $\omega_0 = 0$. For $k = 0$ this yields

$$\sum_{n=0}^{N} h(n) - 1 = 0 \quad \Leftrightarrow \quad \sum_{n=0}^{N} h(n) = 1 \tag{3.56}$$

This is the requirement that the coefficients of the FIR filter must sum up to unity, i.e., the magnitude response at $\omega = 0$ has to be equal to 1. For $k = 1$, Eq. (3.55) yields

$$-\sum_{n=0}^{N} jnh(n) + jD = 0 \quad \Leftrightarrow \quad \sum_{n=0}^{N} nh(n) = D \tag{3.57}$$

for $k = 2$

$$-\sum_{n=0}^{N} n^2 h(n) + D^2 = 0 \quad \Leftrightarrow \quad \sum_{n=0}^{N} n^2 h(n) = D^2 \tag{3.58}$$

and so on until $k = N$.

The $N + 1$ requirements of Eq. (3.55) can be collected together as

$$\sum_{n=0}^{N} n^k h(n) = D^k \quad \text{for } k = 0,1,2,\ldots,N \tag{3.59}$$

This is a set of $N + 1$ linear equations and may be rewritten in the matrix form as

$$\mathbf{Vh} = \mathbf{v} \tag{3.60}$$

where $\mathbf{V}$ is an $L \times L$ Vandermonde matrix ($L = N + 1$)

$$\mathbf{V} = \begin{bmatrix} 0^0 & 1^0 & 2^0 & \cdots & N^0 \\ 0^1 & 1^1 & 2^1 & & N^1 \\ 0^2 & 1^2 & 2^2 & & N^2 \\ \vdots & & & \ddots & \vdots \\ 0^N & 1^N & 2^N & \cdots & N^N \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 0 & 1 & 2 & & N \\ 0 & 1 & 4 & & N^2 \\ \vdots & & & \ddots & \vdots \\ 0 & 1 & 2^N & \cdots & N^N \end{bmatrix} \tag{3.61a}$$

$\mathbf{h}$ is the coefficient vector of the FIR filter

$$\mathbf{h} = [h(0) \quad h(1) \quad h(2) \quad \ldots \quad h(N)]^T \tag{3.61b}$$

and

$$\mathbf{v} = [1 \quad D \quad D^2 \quad \ldots \quad D^N]^T \tag{3.61c}$$

The Vandermonde matrix is known to be nonsingular. Thus it has an inverse matrix $\mathbf{V}^{-1}$ and the solution of Eq. (3.60) can be expressed as

$$\mathbf{h} = \mathbf{V}^{-1}\mathbf{v} \tag{3.62}$$

where $\mathbf{V}^{-1}$ may be evaluated applying *Cramer's rule* (see, e.g., Hildebrand, 1974, p. 541). The solution is given in an explicit form as

$$h(n) = \prod_{\substack{k=0 \\ k \neq n}}^{N} \frac{D-k}{n-k} \quad \text{for } n = 0,1,2,\ldots N \tag{3.63}$$

where D is the fractional delay and N is the order of the FIR filter. The coefficients $h(n)$ can be seen to be equal to those of the *Lagrange interpolation formula* for equally spaced abscissas (see, e.g., Hildebrand, 1974, pp. 89–91).

Ko and Lim (1988) and Hermanowicz (1992) have proposed a more general maximally flat FD approximation where the frequency ω_0 of maximal flatness is arbitrarily chosen from the interval [0, π]. The closed-form solution is (Hermanowicz, 1992)

$$h(n) = e^{j\omega_0(n-D)} \prod_{\substack{k=0 \\ k \neq n}}^{N} \frac{D-k}{n-k} \quad \text{for } n = 0,1,2,\ldots N \tag{3.64}$$

Note that $\omega_0 = 0$ results in Eq. (3.63). If $\omega_0 = \pi$ the filter coefficients are real and they are the same as the Lagrange interpolation coefficients except that the sign of every second one of them has been toggled. This implies that the filter has been modulated to the Nyquist frequency, i.e., its frequency response has been shifted by π.

In the other cases, that is $0 < \omega_0 < \pi$, the coefficients $h(n)$ will be complex-valued. Complex FIR filters are computationally much more expensive than the real-valued FIR filters of the same order. We prefer the real-valued FIR interpolators because they are computationally efficient and conceptually simple, and also because in audio signal processing it is normally appropriate to have the smallest approximation error at low frequencies.

3.3.2 Classical Approach

The Lagrange interpolation method is based on the well-known result in polynomial algebra that using an Nth-order polynomial it is possible to match $N+1$ given arbitrary points (see, e.g., Davis, 1963, pp. 24–26). The uniformly spaced Lagrange interpolation formula can be derived in many different ways. The simplest of them is to approximate a function $x(t)$, $t \in \mathbb{R}$, known at $N+1$ integral points $n = 0, 1, 2, ..., N$ using Nth-order polynomials $h(n, D)$. The polynomial approximation $\hat{x}(t)$ in the range $[0, N]$ is given by

$$\hat{x}(D) = \sum_{n=0}^{N} h(n, D)x(n) \tag{3.65}$$

where $D \in \mathbb{R}$ $(0 \leq D \leq N)$ is the distance from the point $n = 0$ (analogously to fractional delay) and $h(n, D)$, $n = 0, 1, 2, ..., N$ are polynomials in D of order N or less which take on the value 1 when $n = D$ and 0 otherwise. This means that the polynomial $h(n, D)$ may be expressed by means of the Kronecker delta

$$h(n,D) = \delta(n-D) = \begin{cases} 1 & \text{if } n = D \\ 0 & \text{if } n \neq D \end{cases} \tag{3.66}$$

It is easy to write an Nth-order polynomial that vanishes at points $n = 0, 1, 2, ..., n-1, n+1, ..., N$:

$$h(n,D) = C_n\left[D(D-1)\cdots(D-n+1)(D-n-1)\cdots(D-N+1)(D-N)\right] \tag{3.67}$$

The requirement that $h(n,D) = 1$ when $n = D$ is then fulfilled by the scaling constant

$$C_n = \frac{1}{n(n-1)\cdots 1(-1)\cdots(n-N+1)(n-N)} \tag{3.68}$$

Substitution of Eq. (3.68) into (3.67) yields Eq. (3.63), the Lagrange interpolation formula. This approach is a simple, but purely mathematical way to derive the solution. It does not throw light upon the signal processing aspects of the technique like the derivation given in Section 3.3.1.

3.3.3 Symmetry of the Lagrange Interpolation Coefficients

One remarkable feature of Lagrange interpolation is that the coefficients $h(n)$ for the fractional delay $N-D$ are the same as those for D, but in reverse order, that is

$$h(n,D) = h(N-n, N-D) \quad \text{for } n = 0,1,2,\ldots,N \tag{3.69}$$

This property of Lagrange interpolation is easily proved by writing

$$\begin{aligned} h(n,D) &= \prod_{k=0,\, k\neq n}^{N} \frac{D-k}{n-k} \\ &= \frac{D(D-1)\cdots(D-n+1)(D-n-1)\cdots(D-N+1)(D-N)}{n(n-1)\cdots(1)(-1)\cdots(n-N+1)(n-N)} \end{aligned} \tag{3.70}$$

and similarly,

$$\begin{aligned} h(N-n,N-D) &= \prod_{k=0,\, k\neq N-n}^{N} \frac{N-D-k}{N-n-k} \\ &= \frac{(N-D)(N-D-1)\cdots(-D+n+1)(-D+n-1)\cdots(-D+1)(-D)}{(N-n)(N-n-1)\cdots(1)(-1)\cdots(-n+1)(-n)} \\ &= \frac{D(D-1)\cdots(D-n+1)(D-n-1)\cdots(D-N+1)(D-N)}{n(n-1)\cdots(1)(-1)\cdots(n-N+1)(n-N)} \end{aligned} \tag{3.71}$$

The equality in the second and third row of Eq. (3.71) is obtained by changing the sign of every term in the denominator and numerator of the second row. This is valid, since there is the same number of terms (N) in both the denominator and the numerator. It is seen that the last row of Eq. (3.71) is equivalent to that of Eq. (3.52).

The symmetry of Lagrange interpolation coefficients implies that the complementary fractional delay $\Delta = N - D$ can be approximated by the same coefficients $h(n)$ as the delay D. The order of coefficients just has to be reversed. This property is generally true for FD FIR filters. In Chapter 4 it is shown that this feature is advantageous when considering the operation of nonrecursive deinterpolation in digital waveguide systems.

3.3.4 Computation of Coefficients for a Lagrange Interpolator

The first-order Lagrange interpolator corresponds to the well-known *linear interpolation*. The coefficients are obtained from Eq. (3.63) with $N = 1$ as

$$h(0) = 1 - D\,, \quad h(1) = D \tag{3.72}$$

Note that in the first-order case $D = d$ since the integer part of D is zero. The formulas for the filter coefficients of the Lagrange interpolator with $N = 2$ are

$$h(0) = \frac{1}{2}(D-1)(D-2)\,, \quad h(1) = -D(D-2)\,, \quad h(2) = \frac{1}{2}D(D-1) \tag{3.73}$$

and with $N = 3$,

$$\begin{aligned} h(0) &= -\frac{1}{6}(D-1)(D-2)(D-3)\,, & h(1) &= \frac{1}{2}D(D-2)(D-3)\,, \\ h(2) &= -\frac{1}{2}D(D-1)(D-3)\,, & h(3) &= \frac{1}{6}D(D-1)(D-2) \end{aligned} \tag{3.74}$$

In general, the coefficients $h(n)$ of the Lagrange interpolator are determined by an Nth-order polynomial in D. This implies that N additions and N multiplications per coefficient are required to evaluate its value for a given D. Altogether, it takes $N(N+1) = N^2 + N$ additions and $N^2 + N$ multiplications to update all the N coefficients. An efficient procedure to update the coefficients of a Lagrange interpolator was introduced in Välimäki (1992, Appendix A):

1) evaluate the differences $D - 1, D - 2, ..., D - N$;
2) evaluate the products $D(D - 1), (D - 2)(D - 3), ..., (D - N + 1)(D - N)$;
3) evaluate the coefficients $h(0), h(1), ..., h(N)$ by multiplying the results of Steps 1 and 2 and constant coefficients.

Using this technique, the computational burden of evaluating the coefficients of an Nth-order interpolator consists of N additions (subtractions) and less than $N^2 + N$ multiplications.

For example, computing the values of the four coefficients of a third-order Lagrange interpolator using this method requires 3 additions and 10 multiplications. This is a remarkable saving when compared to 12 additions and 12 multiplications that would be required were the straightforward polynomial evaluation used.

The most efficient practical solution to the updating of interpolating coefficients is to use *table lookup*. In software implementations this is usually the fastest way, and thus most recommendable, provided that the required memory space is available.

3.3.5 Windowing Approach to Lagrange Interpolation

Kootsookos and Williams (1995) have proved that the Lagrange interpolation coefficients can also be obtained by windowing the shifted and sampled sinc function with a scaled binomial window. They only discuss the even-order Lagrange interpolator. It is straightforward to extend this result for odd-order Lagrange filters. Below we repeat the derivation of Kootsookos and Williams (1995) with our notation and give the odd-order extension.

The Lagrange interpolation coefficients can be expressed in the following way:

$$h(n,D) = \prod_{k=0,\, k\neq n}^{N} \frac{D-k}{n-k} = (-1)^{N-1-n}\binom{D}{n}\binom{D-n-1}{L-n-1} \quad \text{for } n = 0,1,2,\ldots,N \qquad (3.75a)$$

where $L = N + 1$ and

$$\binom{D}{n} = \frac{\Gamma(D+1)}{n!\,\Gamma(D-n+1)!} \qquad (3.75b)$$

where $\Gamma(\cdot)$ is the gamma function. (The binomial coefficient must be evaluated using the gamma function when one of its parameters is real or non-positive). It is easy to verify Eq. (3.75) by examining the form in Eq. (3.70). This result is valid for both even and odd N.

When N is *even*, the samples of the sinc function under the window of length $L = N + 1$ can be expressed as (Kootsookos and Williams, 1995)

$$\begin{aligned} h_{\text{id}}(n) &= \text{sinc}(n-D) = \frac{\sin[\pi(n-D)]}{\pi(n-D)} \\ &= \frac{(-1)^{N-n+1}\sin(\pi D)}{\pi(n-D)} \quad \text{for } n = 0, 1, 2, \ldots, N \ (N \text{ even}) \end{aligned} \qquad (3.76)$$

Now the formula of the Lagrange interpolation coefficients can be manipulated in the following way (Kootsookos and Williams, 1995):

$$\begin{aligned} h(n,D) &= (-1)^{N-n}\binom{D}{n}\binom{D-n-1}{N-n} = (-1)^{N-n}\binom{D}{n}\binom{D-n}{N-n}\frac{1}{D-n} \\ &= (-1)^{N-n}\binom{D}{N}\binom{N}{n}\frac{1}{D-n} = (-1)^{N-n}\binom{D}{N+1}\binom{N}{n}\frac{N+1}{D-n} \\ &= \frac{(-1)^{N-n+1}\sin(\pi D)}{\pi(n-D)}\binom{D}{N+1}\binom{N}{n}\frac{\pi(N+1)}{\sin(\pi D)} \\ &= \frac{\pi(N+1)}{\sin(\pi D)}\binom{D}{N+1}\binom{N}{n}h_{\text{id}}(n) = C_{\text{bin,e}}(D,N)w_{\text{bin}}(n)h_{\text{id}}(n) \end{aligned} \qquad (3.77)$$

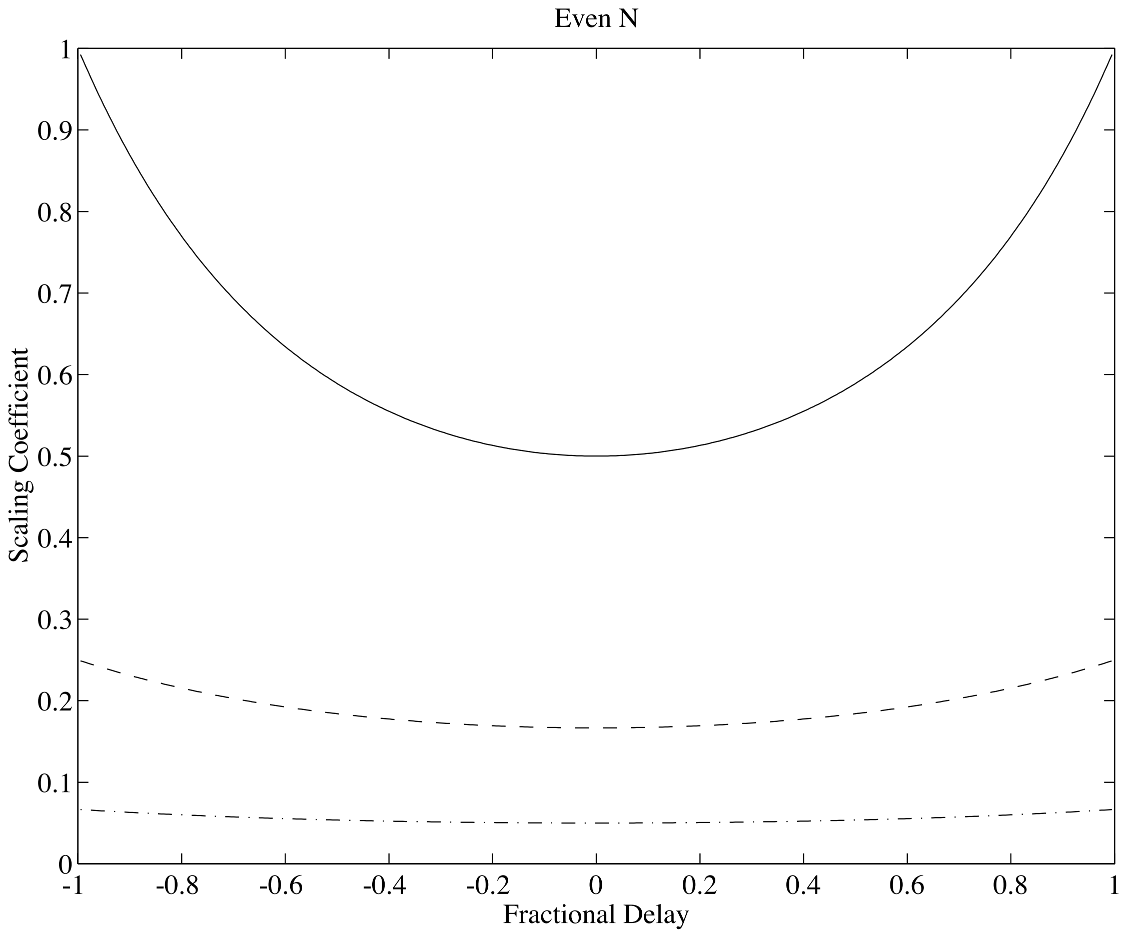

Fig. 3.6 Examples of the scaling coefficient of the binomial window for computation of *even-order* Lagrange interpolation coefficients (solid line: $N = 2$; dashed line: $N = 4$; dash-dot line: $N = 6$).

where $w_{\text{bin}}(n)$ is the *binomial window function* defined by

$$w_{\text{bin}}(n) = \binom{N}{n} \quad \text{for } n = 0, 1, 2, \ldots, N \tag{3.78}$$

and $C_{\text{bin,e}}(D)$ is the scaling coefficient (in the case of even N)

$$C_{\text{bin,e}}(D) = \frac{\pi(N+1)}{\sin(\pi D)} \binom{D}{N+1} \tag{3.79}$$

Note that the binomial window approaches the Gaussian function as N approaches infinity. The scaling coefficient $C_{\text{bin,e}}(D)$ is illustrated in Fig. 3.6 for some low-order Lagrange interpolators when $D - N/2$ varies between –1 and 1.

Odd N

When N is *odd*, the sinc function can be rewritten as

$$h_{\mathrm{id}}(n) = \mathrm{sinc}(n - D) = \frac{\sin[\pi(n - D)]}{\pi(n - D)}$$
$$= \frac{(-1)^{N-n}\sin(\pi D)}{\pi(n - D)} \quad \text{for } n = 0,1,2,\ldots,N \ (N \text{ odd}) \tag{3.80}$$

Now the formula of the Lagrange interpolation coefficients can be manipulated in the following way:

$$h(n,D) = (-1)^{N-n}\binom{D}{n}\binom{D-n-1}{N-n}$$
$$= \frac{(-1)^{N-n+1}\sin(\pi D)}{\pi(n-D)}\binom{D}{N+1}\binom{N}{n}\frac{\pi(N+1)}{\sin(\pi D)} \tag{3.81}$$
$$= (-1)\frac{\pi(N+1)}{\sin(\pi D)}\binom{D}{N+1}\binom{N}{n}h_{\mathrm{id}}(n) = C_{\mathrm{bin,o}}(D,N)w_{\mathrm{bin}}(n)h_{\mathrm{id}}(n)$$

where the scaling coefficient is

$$C_{\mathrm{bin,o}}(D,N) = -\frac{\pi(N+1)}{\sin(\pi D)}\binom{D}{N+1} \tag{3.82}$$

Figure 3.7 shows the scaling coefficient for some low-order Lagrange interpolators (with odd N) for fractional delay values $0 \le d < 1$.

General Result

The two results can be combined by defining a scaling coefficient $C_{\mathrm{bin}}(D,N)$ as

$$C_{\mathrm{bin}}(D,N) = (-1)^N \frac{\pi(N+1)}{\sin(\pi D)}\binom{D}{N+1} \tag{3.83}$$

and the general form of the window-based design of Nth-order for Lagrange interpolator can be expressed as

$$h(n) = C_{\mathrm{bin}}(D,N)w_{\mathrm{bin}}(n)\mathrm{sinc}(n-D), \quad \text{for } 0,1,2,\ldots,N \tag{3.84}$$

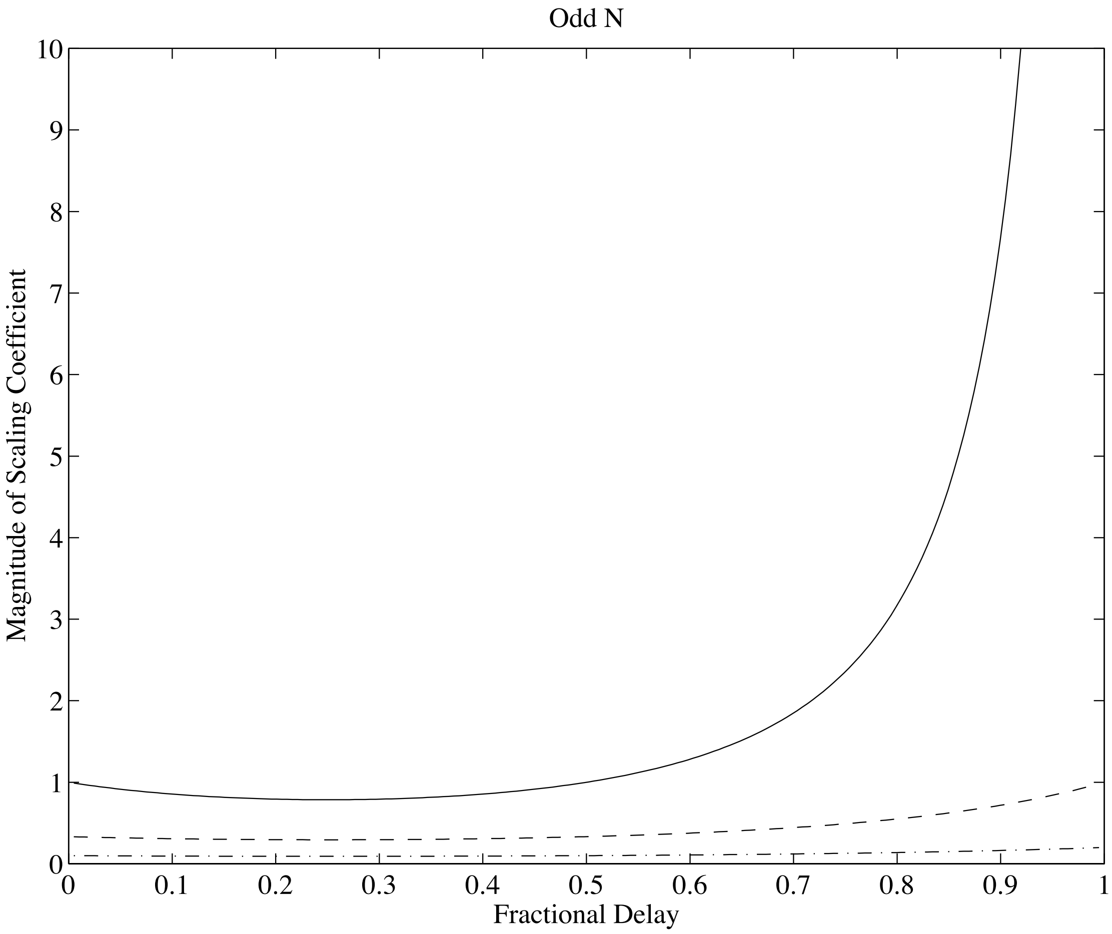

Fig. 3.7 Examples of the scaling coefficient of the binomial window for computation of *odd-order* Lagrange interpolation coefficients (solid line: $N = 1$; dashed line: $N = 3$; dash-dot line: $N = 5$).

3.3.6 Approximation Errors of Lagrange Interpolation

The approximation error of Lagrange interpolation depends drastically on the fractional part d of the interpolation interval D. The error vanishes when $d = 0$. Namely, if D is an integer, Eq. (3.63) can be written as

$$h(n) = \delta(n - D) \quad \text{for } n = 0,1,2,\ldots N \tag{3.85}$$

where $\delta(n - D)$ is the Kronecker delta function defined by

$$\delta(n - D) = \begin{cases} 1 & \text{when } n = D \\ 0 & \text{when } n \neq D \end{cases} \tag{3.86}$$

In other words, the impulse response of the Lagrange interpolator reduces to a delayed unit impulse when D is an integer. This implies that the Lagrange interpolation polynomial goes through the known signal samples.

The worst case is obtained with $d = 0.5$. Then, in the odd-order case, the impulse response of the interpolator is even-symmetric with respect to its center of gravity, or

$$h(n) = h(N - n) \quad \text{for } n = 0,1,2,\ldots,N \tag{3.87}$$

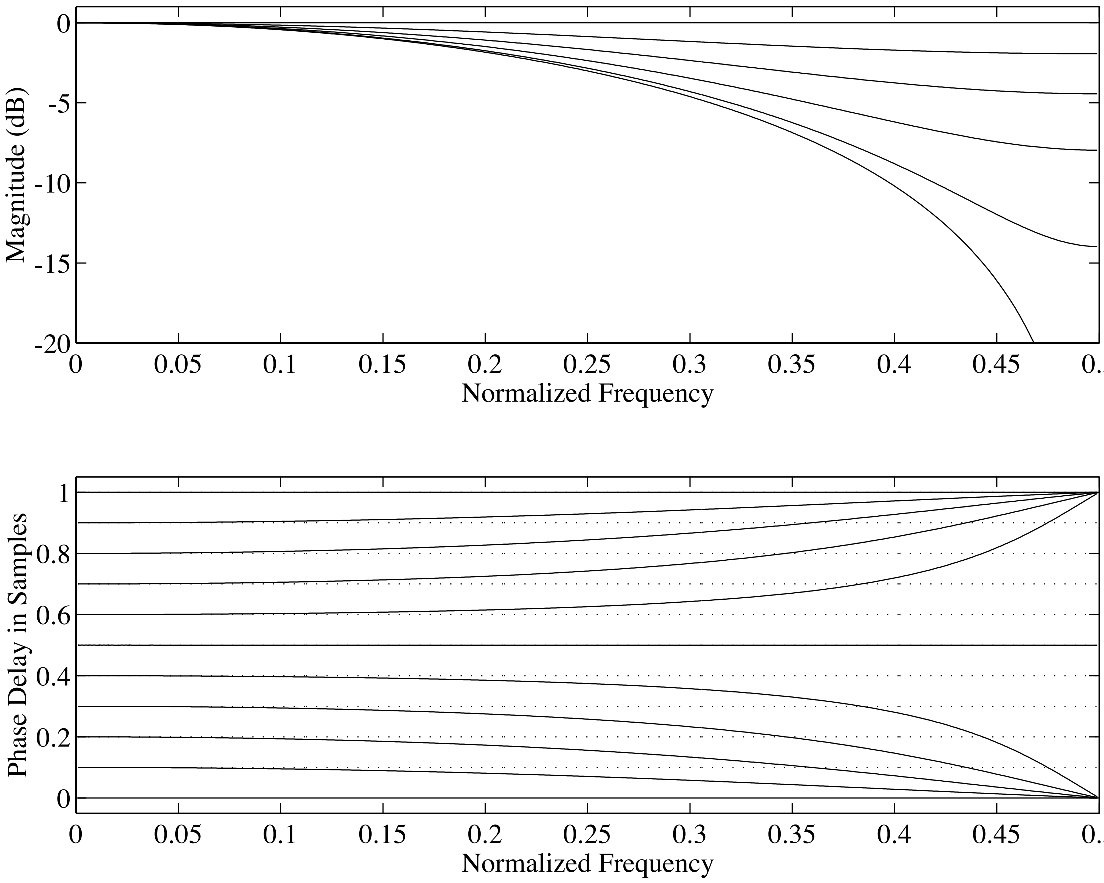

Fig. 3.8 The magnitude (upper) and phase delay (lower) response of a *linear interpolator* for eleven different fractional delay values (D = 0, 0.1, 0.2, ..., 1.0). Note that there are only six different curves in the upper figure (not eleven), because the magnitude responses for fractional delays d and $1–d$ are the same. In the lower figure, the ideal phase delay in each case is illustrated by a dotted line.

The interpolator is thus a linear-phase FIR filter and its phase response is exactly $-D\omega$ as it should be. Unfortunately, for an odd-order FIR filter this also means that it has a real zero at the Nyquist frequency (see, e.g., Parks and Burrus, 1987, pp. 23–26) and consequently the total error is larger than for any other value of d. For an even-order Lagrange interpolator there is no particular symmetry when $d = 0.5$, but the approximation error in both magnitude and phase is the largest.

The magnitude and phase responses of the first, second, and third-order Lagrange interpolators are illustrated in Figs. 3.8–3.10. Note that in these figures the normalized frequency 0.5 corresponds to the Nyquist frequency. It is seen that both the magnitude and the phase delay curves coincide with the ideal response at low frequencies as expected.

Cain *et al.* (1994) have compared the approximation error of Lagrange interpolation with that of other FD FIR filters. They concluded that it is preferable for applications where a low-order FD filter is needed and a fullband approximation is not necessary.

From the viewpoint of waveguide models, Lagrange interpolation has two favorable features:

1) it is accurate at low frequencies and

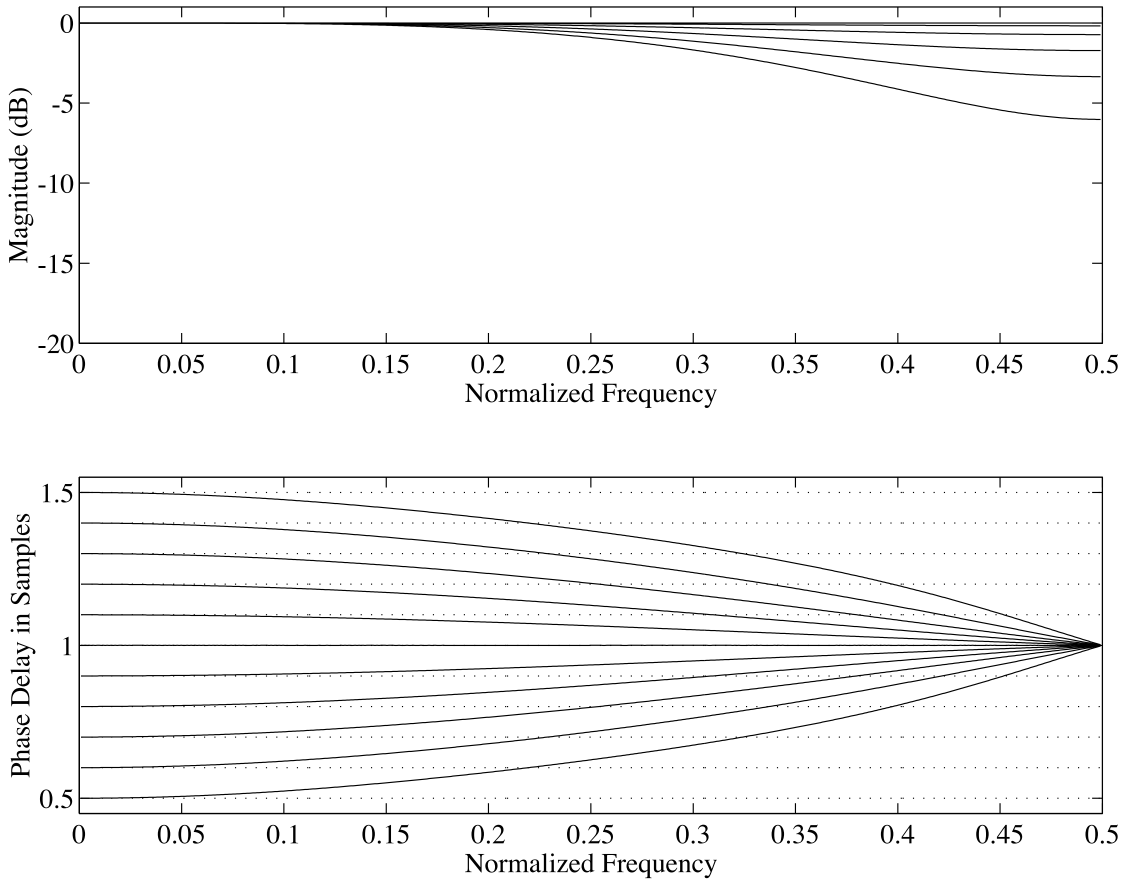

Fig. 3.9 The magnitude (upper) and phase delay (lower) response of a *second-order Lagrange interpolator* ($N = 2$) for eleven different fractional delay values (D = 0.5, 0.6, ..., 1.0, ..., 1.4, 1.5). Note that the magnitude responses for fractional delays D and $N - D$ are the same. In the lower figure, the ideal phase delay in each case is illustrated by a dotted line.

2) it never overestimates the amplitude of the signal when the delay has been chosen so that $(N - 1)/2 \leq D \leq (N + 1)/2$ when N is odd and $(N/2) - 1 \leq D \leq (N/2) + 1$ when N is even.

The first advantage is justified by the fact that audio signals are usually lowpass signals and thus it is wise to use an approximation technique that has the smallest error at low frequencies.

The second property is called *passivity* and it implies that the magnitude response of the Lagrange interpolator is less than or equal to one for the mentioned values of D. This property is advantageous because in digital waveguide models the interpolator is normally used inside a feedback loop and then it is extremely important to preserve the loop gain less than unity. Otherwise the system may become unstable. Since a Lagrange interpolator is a passive filter, the interpolation error only decreases the loop gain but never increases it. It is interesting that in the case of even-order Lagrange interpolators, the filter is passive on a range of *two unit delays*, while odd-order filters only have a range of one unit delay.

It has been experimentally noticed that the magnitude response of the Lagrange interpolator exceeds unity when the delay parameter is out of the optimal range. An example of this phenomenon is given in Fig. 3.11 for the third-order Lagrange interpo-

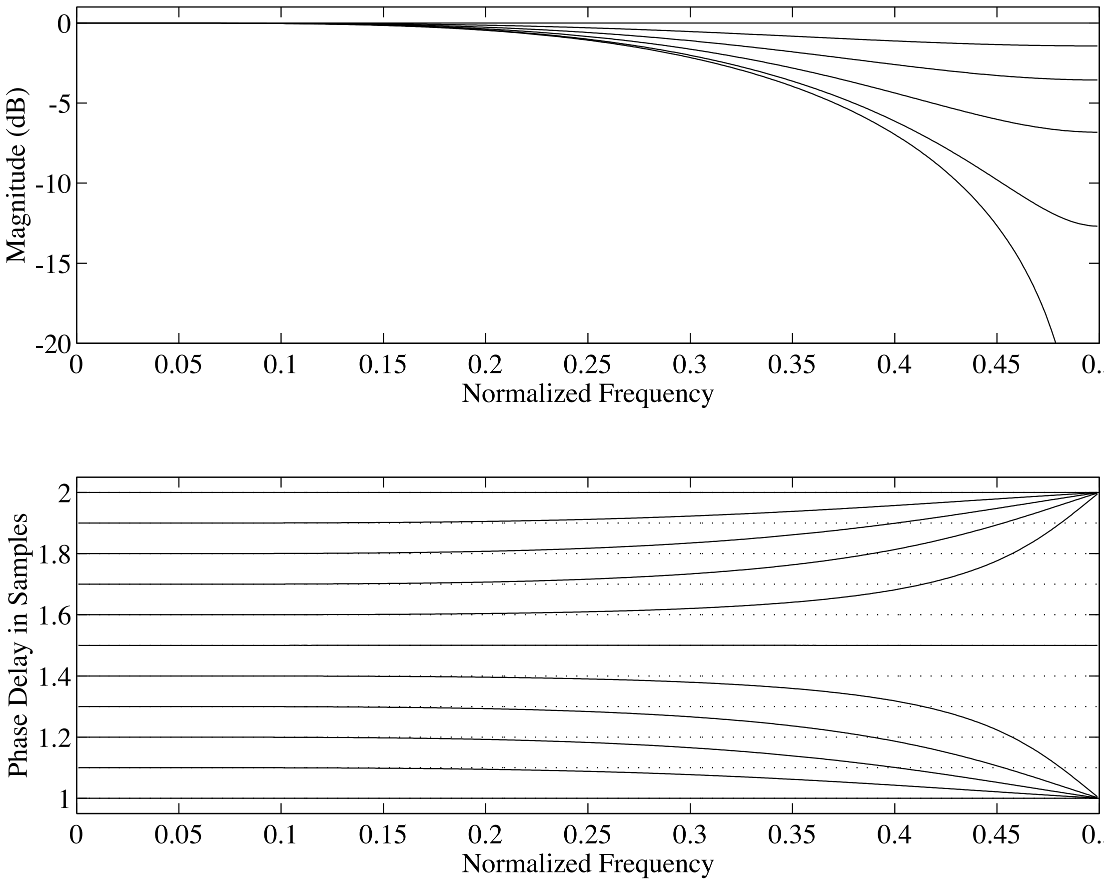

Fig. 3.10 The magnitude (upper) and phase delay (lower) response of a *third-order Lagrange interpolator* ($N = 3$) for 11 equally spaced fractional delay values ($D = 1.0, 1.1, 1.2, ..., 2.0$).

lator when the parameter D varies from 0 to 1. (In this case the delay parameter should be in the range $1 \leq D \leq 2$). When $0.5 \leq D < 1$, the magnitude response is greater than one at all other frequencies except $\omega = 0$. When $0 < D < 0.5$, the magnitude response exceeds unity at middle frequencies but shows lowpass behavior at the high end. (Note that the scale of the magnitude response in Fig. 3.11 is different than in Figs. 3.8–3.10.) Also the phase delay approximation appears to be substantially worse now than when D is on the optimal range (compare with the phase response in Fig. 3.10).

The squared integral error has been computed for Lagrange interpolators of different order using Eq. (3.31). Figure 3.12 shows the error as a function of the delay parameter for odd-order filters $N = 1$, 3, and 5 and Fig. 3.13 for even-order filters $N = 2$, 4, and 6.

The odd-order Lagrange interpolators have an error curve that is symmetrical with respect to the point $D = N/2$. The lobe between the middle taps has approximately the form of the sine-squared function (Laakso *et al.*, 1995a).

The even-order Lagrange interpolators also have error curves that are symmetrical with respect to $D = N/2$ (see Fig. 3.13). Note, however, that individual lobes of the error curves are asymmetrical. In practice the shape of these curves suggests that the lowest error is obtained when the even-order interpolator is used for the values $(N - 1)/2 \leq D \leq (N + 1)/2$. This is the same requirement as for the odd-order filter, although the even-order filter is passive when $(N/2) - 1 \leq D \leq (N/2) + 1$.

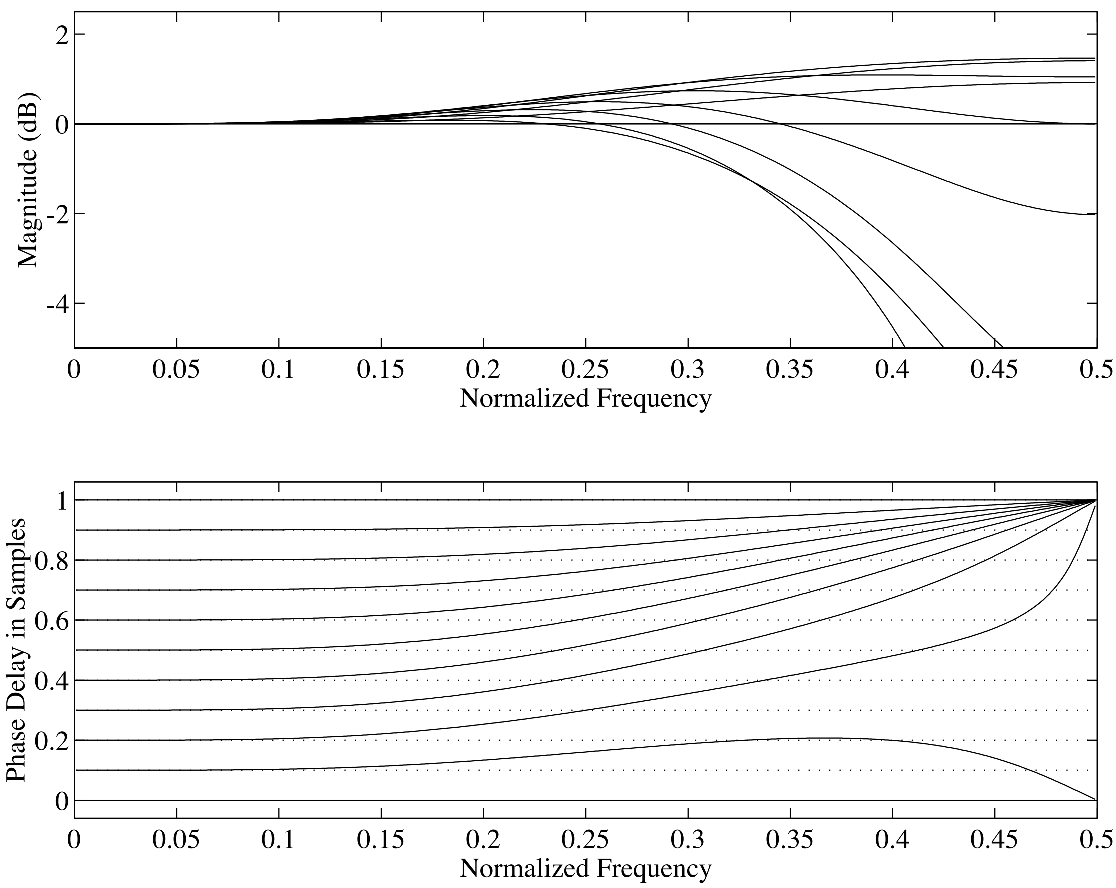

Fig. 3.11 The magnitude (upper) and phase delay (lower) response of a *third-order Lagrange interpolator* ($N = 3$) the for 11 equally spaced fractional delay values ($D = 0.0, 0.1, 0.2, ..., 1.0$). Note that now D is varied on a non-optimal range since $D \leq (N-1)/2$.

As seen above, both even and odd-order Lagrange interpolators can be used for FD approximation. This is in contrast to some statements in the literature that recommend the use of odd-order interpolators only (see, e.g., Erup *et al.*, 1993, p. 999). However, in dynamic FD applications even-length interpolators introduce a practical problem: when d passes the value 0.5, the locations of the filter taps have to be changed to maintain d within half a sample from the center point of the filter. This ensures that the overall approximation error is always as small as possible. However, this change of filter taps causes a discontinuity to the output signal. The odd-order Lagrange interpolators do not have this problem since the filter taps need to be moved only when D passes an integer value.

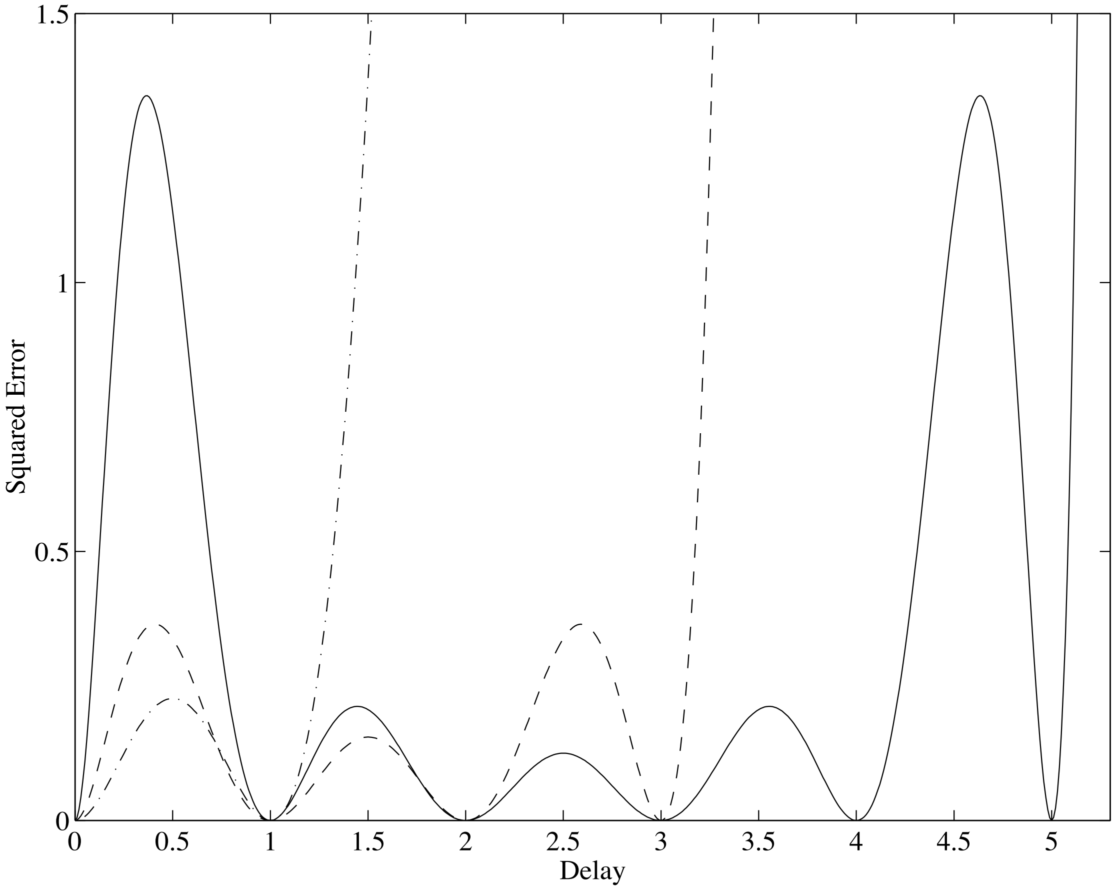

Fig. 3.12 Squared error of some odd-order Lagrange interpolators as a function of delay D: $N = 1$ (dash-dot line), $N = 3$ (dashed line), and $N = 5$ (solid line).

3.3.7 Farrow Structure of Lagrange Interpolation

In this section we present a new implementation structure for Lagrange interpolation. This derivation has been first published by Välimäki (1994a, 1994b, 1995a)†. Lagrange interpolation is usually implemented using a direct-form FIR filter structure. An alternative structure is obtained approximating the continuous-time function $x_c(t)$ by a polynomial in D, which is the interpolation interval or fractional delay. The interpolants, i.e., the new samples, are now represented by

$$y(n) = x_c(n - D) = \sum_{k=0}^{N} c(k) D^k \tag{3.88}$$

that takes on the value $x(n)$ when $D = n$. The coefficients $c(k)$ are solved from a set of $N + 1$ linear equations. Farrow (1988) suggested that every filter coefficient of an FIR interpolating filter could be expressed as an Nth-order polynomial in the delay parameter D. He stated that this results in $N + 1$ FIR filters with constant coefficients. The above approach to Lagrange interpolation is seen to be related to Farrow's idea.

† After publication of the theory of this new structure, it was found out that the same idea had already been mentioned by Erup *et al.* (1993, Appendix, pp. 1007–1008) but without derivation. This is however an independent work. To the knowledge of the author, the derivation given here has not been published by anybody else.

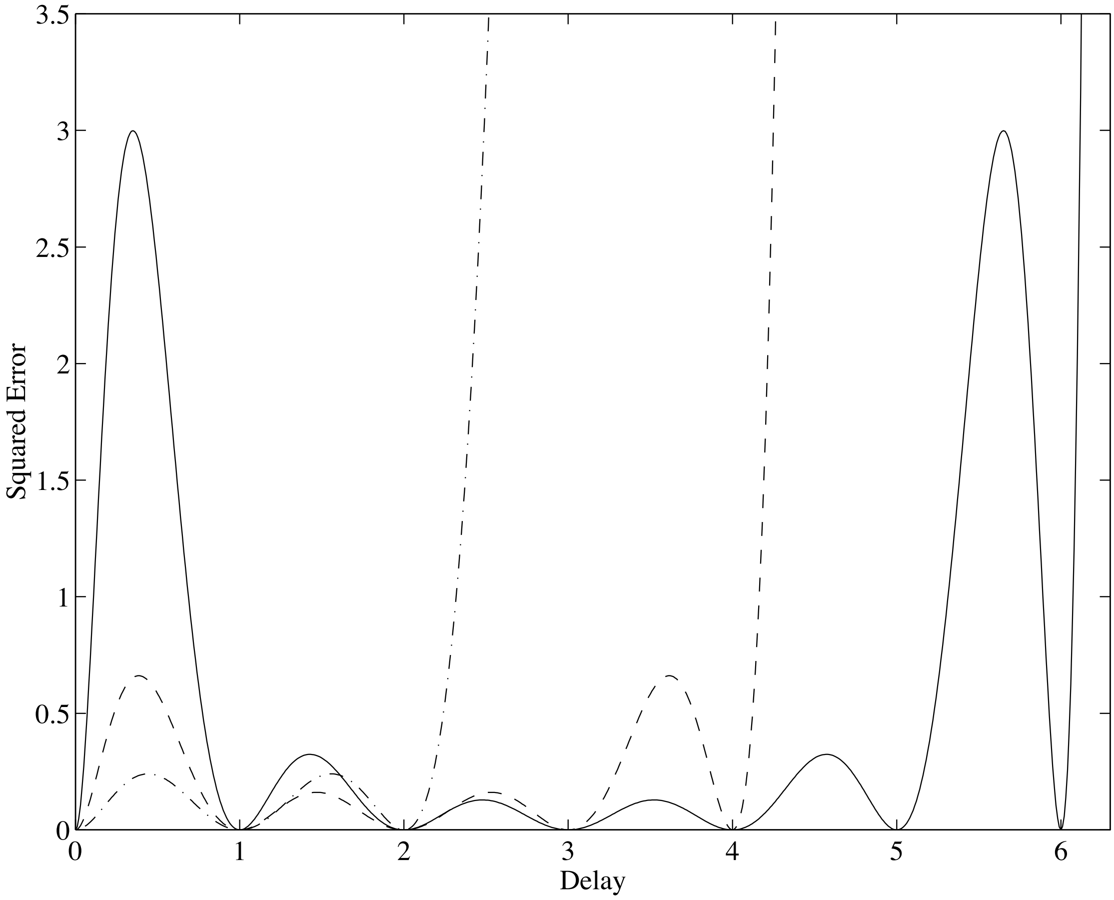

Fig. 3.13 Squared error of some even-order Lagrange interpolators as a function of delay D: $N = 2$ (dash-dot line), $N = 4$ (dashed line), and $N = 6$ (solid line).

The alternative implementation for Lagrange interpolation is obtained formulating the polynomial interpolation problem in the z-domain as

$$Y(z) = H(z)X(z) \tag{3.89}$$

where $X(z)$ and $Y(z)$ are the z-transforms of the input and output signal, $x(n)$ and $y(n)$, respectively, and the transfer function $H(z)$ is now expressed as a polynomial in D (instead of z^{-1}).

$$H(z) = \sum_{k=0}^{N} C_k(z)D^k \tag{3.90}$$

The familiar requirement that the output sample should be one of the input samples for integer D may be written in the z-domain as

$$Y(z) = z^{-D}X(z) \quad \text{for } D = 0,1,2,\ldots,N \tag{3.91}$$

Together with Eqs. (3.89) and (3.90) this leads to the following $N + 1$ conditions

$$\sum_{k=0}^{N} C_k(z)D^k = z^{-D} \quad \text{for } D = 0,1,2,\ldots,N \tag{3.92}$$

This may be expressed in matrix form as

$$\mathbf{Uc} = \mathbf{z} \tag{3.93}$$

where the $L \times L$ matrix $\mathbf{U}$ is given by

$$\mathbf{U} = \begin{bmatrix} 0^0 & 0^1 & 0^2 & \cdots & 0^N \\ 1^0 & 1^1 & 1^2 & & 1^N \\ 2^0 & 2^1 & 2^2 & & 2^N \\ \vdots & & & \ddots & \vdots \\ N^0 & N^1 & N^2 & \cdots & N^N \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 1 & & 1 \\ 1 & 2 & 4 & & 2^N \\ \vdots & & & \ddots & \vdots \\ 1 & N & N^2 & \cdots & N^N \end{bmatrix} \tag{3.94}$$

vector $\mathbf{c}$ is

$$\mathbf{c} = \begin{bmatrix} C_0(z) & C_1(z) & C_2(z) & \ldots & C_N(z) \end{bmatrix}^T \tag{3.95}$$

and the delay vector

$$\mathbf{z} = \begin{bmatrix} 1 & z^{-1} & z^{-2} & \ldots & z^{-N} \end{bmatrix}^T \tag{3.96}$$

Note that $\mathbf{U}$ is the transpose of the matrix $\mathbf{V}$ given in Eq. (3.60b).

The matrix $\mathbf{U}$ has the Vandermonde structure and thus it has an inverse matrix $\mathbf{U}^{-1}$. The solution of Eq. (3.93) can thus be written as

$$\mathbf{c} = \mathbf{U}^{-1}\mathbf{z} \tag{3.97}$$

The inverse matrix $\mathbf{U}^{-1}$, that we shall denote by $\mathbf{Q}$, may be solved using Cramer's rule.

The rows of the inverse Vandermonde matrix $\mathbf{Q}$ contain the filter coefficients used in the new structure, and thus it is convenient to write

$$\mathbf{Q} = [\mathbf{q}_0 \quad \mathbf{q}_1 \quad \mathbf{q}_2 \quad \cdots \quad \mathbf{q}_N]^T \tag{3.98}$$

The transfer functions $C_n(z)$ are thus obtained by inner product as

$$C_n(z) = \mathbf{q}_n\mathbf{z} = \sum_{k=0}^{N} q_n(k)z^{-k} \quad \text{for } n = 1,2,\ldots,N \tag{3.99}$$

The coefficients $q_n(k)$ for the FIR filters $C_n(z)$ are computed inverting the Vandermonde matrix $\mathbf{U}$.

By setting $D = 0$ in Eq. (3.92), it is seen that

$$\sum_{k=0}^{N} C_k(z)0^k = 1 \quad \Rightarrow \quad C_0(z) \equiv 1 \tag{3.100}$$

This implies that the transfer function $C_0(z) = 1$ regardless of the order of the interpolator. The other transfer functions $C_n(z)$ given by Eq. (3.99) are Nth-order polynomials in z^{-1}, that is, they are Nth-order FIR filters.

We shall call this the implementation technique describe above the *Farrow structure of Lagrange interpolation*. A remarkable feature of this form is that the transfer func-

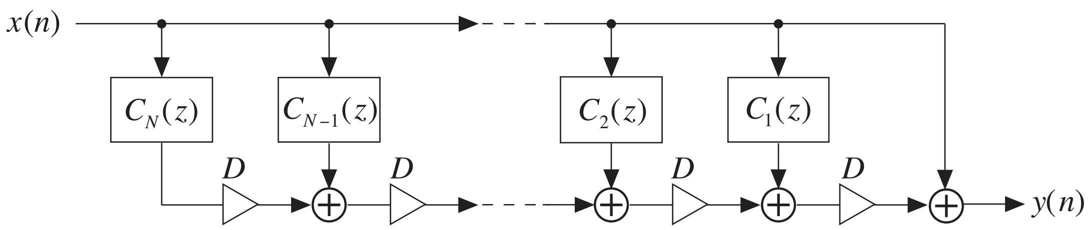

Fig. 3.14 The Farrow structure of Lagrange interpolation implemented using Horner's method. The overall transfer function has been formulated as a polynomial in the delay parameter *D*. The transfer functions $C_n(z)$ are *N*th-order FIR filters with constant coefficients for a given *N*.

tions $C_n(z)$ are *fixed* for a given order *N*. The interpolator is directly controlled by the fractional delay *D*, i.e., no computationally intensive coefficient update is needed when *D* is changed.

The Farrow structure is most efficiently implemented using *Horner's method* (see, e.g., Hildebrand, 1974, p. 28), that is

$$\sum_{k=0}^{N} C_k(z)D^k = C_0(z) + [C_1(z) + [C_2(z) + \ldots + [C_{N-1}(z) + C_N(z)\overbrace{D]D\cdots]D}^{N} \tag{3.101}$$

With this method *N* multiplications by *D* are needed. A general *N*th-order Lagrange interpolator that employs the suggested approach is shown in Fig. 3.14. Since there is no need for the updating of coefficients, this structure is particularly well suited to applications where the fractional delay *D* is changed often, even after every sample interval.

If the delay is constant or updated very seldom, it is recommended to implement Lagrange interpolation using the standard FIR filter structure because it is computationally less expensive. Namely, with the FIR filter structure *N* + 1 multiplications and *N* additions are needed. In Farrow's structure, there are *N* pieces of *N*th-order FIR filters which results in $N(N+1)$ multiplications and N^2 additions. There are also *N* multiplications by *D* and *N* additions. Altogether this means $N^2 + 2N$ multiplications and $N^2 + N$ additions per output sample.

As examples, let us solve for the transfer functions $C_n(z)$ for linear interpolation and second-order Lagrange interpolation. For *N* = 1, Eq. (3.93) yields

$$\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} C_0(z) \\ C_1(z) \end{bmatrix} = \begin{bmatrix} 1 \\ z^{-1} \end{bmatrix} \tag{3.102}$$

The solution is given by

$$\begin{bmatrix} C_0(z) \\ C_1(z) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ z^{-1} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ z^{-1} \end{bmatrix} = \begin{bmatrix} 1 \\ -1 + z^{-1} \end{bmatrix} \tag{3.103}$$

It is seen that $C_1(z) = z^{-1} - 1$ when $N = 1$. The overall transfer function $H(z)$ of the Farrow structure of linear interpolation is written as

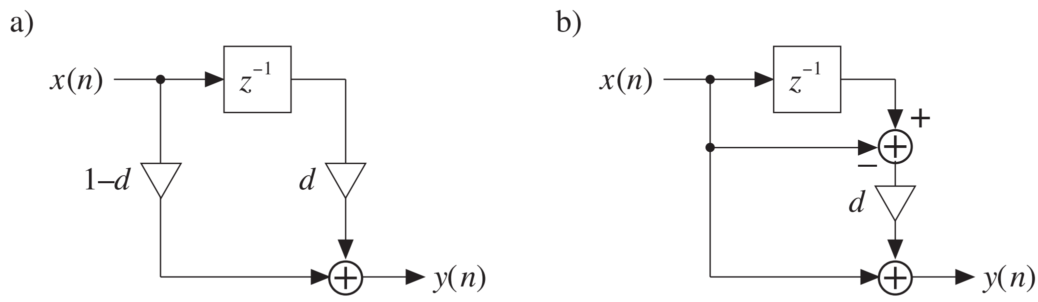

Fig. 3.15 a) The direct-form FIR filter structure for linear interpolation and b) the equivalent Farrow structure.

$$H(z) = 1 + (z^{-1} - 1)D \tag{3.104}$$

Note that in this case $D = d$. The linear interpolator may thus be implemented by the structure illustrated in Fig. 3.15b. It is seen that the Farrow structure is as efficient as the direct-form nonrecursive structure (Fig. 3.15a) when $N = 1$, since the number of operations is the same in both. If multiplication is more expensive than addition, like it is in VLSI implementations, then the Farrow structure (Fig. 3.15b) is preferable.

For $N = 2$, Eq. (3.93) is written as

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \end{bmatrix} \begin{bmatrix} C_0(z) \\ C_1(z) \\ C_2(z) \end{bmatrix} = \begin{bmatrix} 1 \\ z^{-1} \\ z^{-2} \end{bmatrix} \tag{3.105}$$

Now the inverse Vandermonde matrix **Q** is given by

$$\mathbf{Q} = \mathbf{U}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -3/2 & 2 & -1/2 \\ 1/2 & -1 & 1/2 \end{bmatrix} \tag{3.106}$$

The transfer functions thus obtained are

$$C_0(z) = 1, \quad C_1(z) = -\frac{3}{2} + 2z^{-1} - \frac{1}{2}z^{-2}, \quad C_2(z) = \frac{1}{2} - z^{-1} + \frac{1}{2}z^{-2} \tag{3.107}$$

and the overall transfer function $H(z)$ can be written as

$$H(z) = C_0(z) + C_1(z)D + C_2(z)D^2 \tag{3.108}$$

The Farrow form of parabolic or second-order Lagrange interpolation is illustrated in Fig. 3.16. In this example it is seen that the transfer functions $C_n(z)$ can *share unit delays*, because they all use the same delayed signal values $x(n - k)$ with $k = 0, 1, 2, ..., N$. Furthermore, the number of multiplications can be reduced in a practical implementation. It may be taken into account that some of the coefficients have the value 1 or –1 thus eliminating the need for a multiplication [e.g., $q_2(2)$, the second coefficient of $C_2(z)$] and that the corresponding coefficients of two transfer functions can be equal

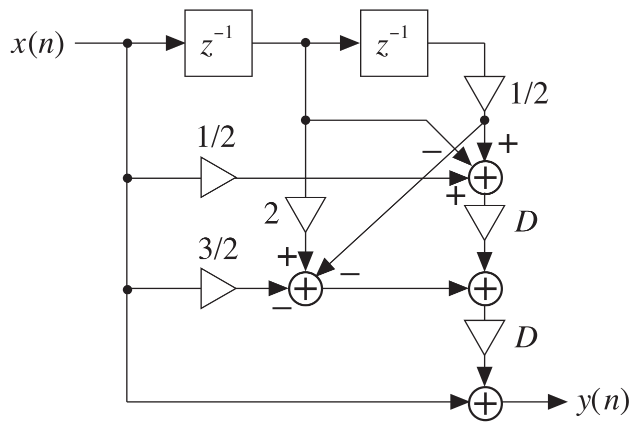

Fig. 3.16 The Farrow structure for a second-order ($N = 2$) Lagrange interpolator. This structure involves 6 additions and 6 multiplications. The best fractional delay approximation is obtained when $0.5 \le D \le 1.5$.

[e.g., the third coefficient of $C_1(z)$ and $C_2(z)$]. These special cases are often met with the inverse Vandermonde matrix.

Modified Farrow Structure

The Farrow structure can be made more efficient changing the range of the parameter D so that the integer part is removed. The new parameter range is $0 \le d \le 1$ (for odd N) or $-0.5 \le d \le 0.5$ (for even N). This change can be obtained introducing a transformation matrix $\mathbf{T}$ defined by

$$T_{n,m} = \begin{cases} \text{round}\left(\dfrac{N}{2}\right)^{n-m} \dbinom{n}{m} & \text{for } n \ge m \\ 0 & \text{for } n < m \end{cases} \tag{3.109}$$

where $n, m = 0, 1, 2, ..., N$. For $N = 2$ this matrix is expressed as

$$\mathbf{T} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.110}$$

Multiplying the coefficient matrix $\mathbf{Q}$ by matrix (3.110) a modified coefficient matrix $\tilde{\mathbf{Q}}$ is obtained as

$$\tilde{\mathbf{Q}} = \mathbf{TQ} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ -3/2 & 2 & -1/2 \\ 1/2 & -1 & 1/2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1/2 & 0 & 1/2 \\ 1/2 & -1 & 1/2 \end{bmatrix} \tag{3.111}$$

This transformation is equivalent to substituting $D' = D + 1$. The FIR filters of the modified structure are written as

$$\tilde{C}_0(z) = z^{-1}, \quad \tilde{C}_1(z) = -\frac{1}{2} + \frac{1}{2} z^{-2}, \quad \tilde{C}_2(z) = \frac{1}{2} - z^{-1} + \frac{1}{2} z^{-2} \tag{3.112}$$

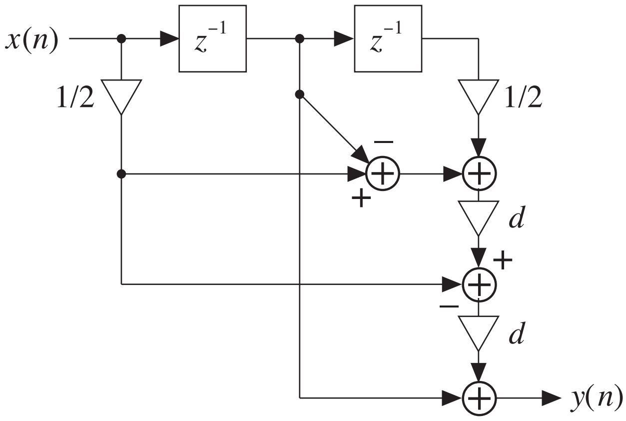

Fig. 3.17 The modified Farrow structure for a second-order ($N = 2$) Lagrange interpolator. This structure involves 4 additions and 4 multiplications. In this case the best fractional delay approximation is obtained when $-0.5 \le d \le 0.5$. Note, however, that the structure produces an extra delay of one sample so that the actual delay then lies within the range [0.5, 1.5].

Figure 3.17 shows the implementation of this modified second-order Farrow structure of Lagrange interpolation. It is seen that now only 4 multiplications and additions are needed in contrast with 6 in the basic implementation shown in Fig. 3.16.

The final example of the modified Farrow's structure of Lagrange interpolation presents the third-order system. The inverse Vandermonde matrix $\mathbf{Q}$ associated with the third-order case is given by

$$\mathbf{Q} = \mathbf{U}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -11/6 & 3 & -3/2 & 1/3 \\ 1 & -5/2 & 2 & -1/2 \\ -1/6 & 1/2 & -1/2 & 1/6 \end{bmatrix} \tag{3.113}$$

This coefficient matrix does not yield a very efficient implementation since there are only two pairs of coefficients that can be combined. The coefficient matrix $\hat{\mathbf{Q}}$ for the modified structure is obtained multiplying $\mathbf{Q}$ by $\mathbf{T}$, which yields

$$\hat{\mathbf{Q}} = \mathbf{TQ} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{Q} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1/3 & -1/2 & 1 & -1/6 \\ 1/2 & -1 & 1/2 & 0 \\ -1/6 & 1/2 & -1/2 & 1/6 \end{bmatrix} \tag{3.114}$$

The block diagram of this system is shown in Fig. 3.18. It is seen that 11 additions and 9 multiplications are needed in the implementation of this system.

For comparison we consider third-order Lagrange interpolation implemented using the direct-form FIR filter structure with the coefficients updated according to the technique suggested in Section 3.3.4. The computational load for the coefficient update would be 3 additions and 10 multiplications, and for the computation of the output 3 additions and 4 multiplications. Altogether this yields 6 additions and 14 multiplica-

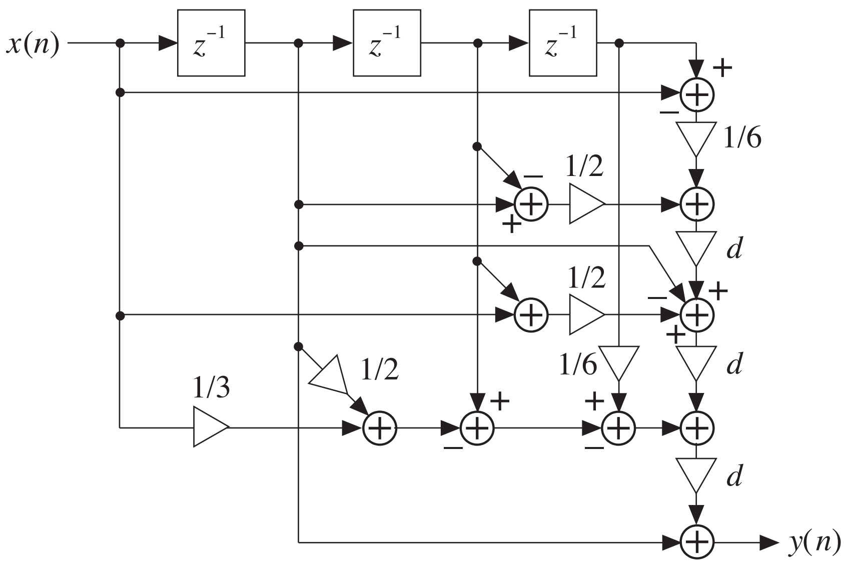

Fig. 3.18 The modified Farrow structure for a third-order (N = 3) Lagrange interpolator. This structure involves 11 additions and 9 multiplications. In this case the best fractional delay approximation is obtained when $0 < d \le 1$. Note, however, that the structure produces an extra delay of one sample so that the actual delay lies within the range [1, 2].

tions, which makes 20 operations—the same number as with the modified Farrow structure. The number of multiplications, however, is smaller in the case of the modified Farrow structure.

3.3.8 Related Polynomial Interpolation Techniques

Lagrange interpolation is one representative of a class of polynomial interpolation techniques. Other interpolation methods based on polynomials have not been widely studied. Here we discuss some examples from recent DSP literature.

Matsui *et al*. (1991) proposed a *tangent-line interpolation* (TLI) technique for fractional delay approximation. They used it to control the fundamental frequency of a speech synthesizer. The basic idea of the TLI method is to approximate the value of a sampled signal in the neighborhood of a known sample $x(n)$ by a straight line that takes on the value $x(n)$ at point n. This technique is not equivalent to linear interpolation since the slope of the line is computed from the previous and next sample values. This results in a three-tap FIR filter with coefficients

$$h(0) = \frac{1-D}{2}, \quad h(1) = 1, \quad \text{and} \quad h(2) = \frac{D-1}{2} \tag{3.115}$$

The frequency-domain properties of the TLI method were studied in Välimäki (1994b). It was found that there is an overshoot in the magnitude response of the filter with a maximum at about $f_s / 4$. In the worst case (d = 0.5) this overshoot is about 1 dB. The phase delay of the TLI filter is exact $\omega = 0$ as in the case of Lagrange interpolators. Unfortunately the phase delay approaches the value D = 1 almost linearly as a function of frequency. Thus this method is unsuitable for tasks where the accuracy of fractional delay approximation is important.

Another polynomial interpolator with a closed-form expression for the coefficients has been introduced by Erup *et al.* (1993). This technique is called *piecewise parabolic interpolation* (PPI), and it is implemented with a four-tap FIR filter with coefficients

$$\begin{aligned} h(0) &= \alpha d^2 - \alpha d \\ h(1) &= -\alpha d^2 + (\alpha - 1)d + 1 \\ h(2) &= -\alpha d^2 + (\alpha + 1)d \\ h(3) &= \alpha d^2 - \alpha d \end{aligned} \tag{3.116}$$

where α is a real-valued parameter and d is the fractional delay ($0 \le d \le 1$). This interpolation technique produces a constant delay of one sample plus a delay of approximately d. Note that the coefficients $h(0)$ and $h(3)$ are identical. With $\alpha = 0$, the PPI filter reduces to linear interpolation.

Erup *et al.* (1993) recommend the PPI method with parameter value $\alpha = 0.5$ for FD approximation. This filter was analyzed in Välimäki (1994b). Both the magnitude response and the phase delay are almost flat at low frequencies. At the high end the magnitude decreases and the phase delay approaches 1 or 2 depending on d. The main drawback of this technique is that the magnitude response exceeds unity at middle frequencies. In the worst case ($d = 0.5$), the overshoot at the culmination point $0.2 f_s$ is 0.74 dB or slightly less than 9%.

The properties of the PPI filter are quite comparable to those of low-order Lagrange interpolators. Furthermore, the PPI filter can be implemented very efficiently with the Farrow structure (Erup *et al.*, 1993). This method is preferred over the third-order Lagrange interpolator if the small overshoot in its magnitude response is not a serious drawback. In waveguide models, the PPI filter could be used for tuning the delay-line lengths if the loop gain were much less than unity at middle frequencies, that is, if a great deal of losses were included. In this case the overshoot would not risk the stability of the waveguide system.

Splines are a class of polynomial interpolation techniques that have several applications in numerical computation. So far they have been tried in many DSP applications, such as image processing (Hou and Andrews, 1978; Unser *et al.*, 1993b), sampling-rate conversion (e.g., Cucchi *et al.*, 1991; Zölzer and Boltze 1994), and signal reconstruction (Unser *et al.*, 1992, 1993a). Aldroubi *et al.* (1992) have proved that a cardinal spline interpolator converges toward the ideal interpolator as the order of the filter approaches infinity. An interesting field of future research is to compare this technique with other known methods, e.g., Lagrange interpolation.

3.3.9 Conclusion and Discussion

In this section the maximally flat design of an FD FIR filter—which is equivalent to classical Lagrange interpolation—was discussed. It appears to be well suited to waveguide modeling because the approximation error is small at low frequencies and it is passive (i.e., its magnitude response never exceeds unity) when the delay parameter is within the optimal range. It was shown that both the even and odd-order Lagrange interpolation filters can also be designed by windowing the shifted and sampled sinc function with a scaled binomial window.

A novel implementation technique for the Lagrange interpolation was derived. This formulation—called the Farrow structure—leads to a version of Lagrange interpolation that is well suited to time-varying FD filtering. The computational cost of this structure is in general the same as that for the direct-form FIR filter when the Lagrange interpolation coefficients are updated every cycle. The number of multiplications, however, is always smaller in the new structure. Suitable applications for the Farrow structure of Lagrange interpolation are, for example, irrational sampling-rate conversion (Laakso *et al.*, 1995a) or time-varying resampling of a discrete-time signal.

Before leaving Lagrange interpolation, it is worthwhile emphasizing a potential point of misunderstanding. In a classical paper Rabiner and Schafer (1973) explain that when Lagrange interpolation is used for increasing the sample rate of a discrete-time signal with an integer factor Q, the operation will not change the phase of that signal since the interpolator has a *linear phase function*. Above we have conversely seen that in general Lagrange interpolators do not have a linear phase response. This disagreement is caused by the different way the interpolator is used: in upsampling with factor Q, the interpolator will compute $Q-1$ new signal values between every two input samples. Each of the new samples is obtained with a different nonlinear-phase Lagrange interpolator but the resulting signal (consisting of the original signal and its $Q-1$ fractionally shifted versions) has a linear phase. On the other hand, in fractional delay filtering (usually) a single interpolator is used to produce the contiguous output samples—one sample per sampling interval, and for this reason the output signal suffers from phase distortion. If upsampling by Q is followed by decimation by an integer factor P, the resulting signal is not generally linear phase.

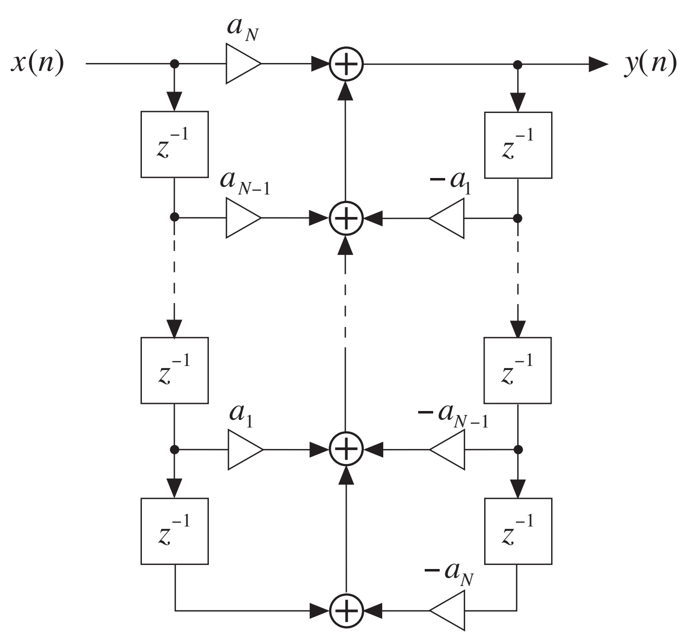

Fig. 3.19 Direct form I implementation of an Nth-order discrete-time allpass filter.