Chapter 4

Evaluation of Clock Synchronization Techniques

This chapter describes the effect of clock offset, including interference and phase rotation, and how it degrades the system performance. Various timing correction schemes are also described. The second section introduces the interpolation methods for the clock offset correction.

4.1 The Effect of Sample Clock Frequency Offset

The frequencies of the oscillators that respectively drive DAC and ADC at the transmitter and receiver will never be the same. As shown in Figure 4.1, there exists a constant delay t_d and clock offset t_Δ between transmitted samples and received samples. The clock frequency offset is defined by [11]

$$t_\Delta = \frac{T' - T}{T} \tag{4.1}$$

where T and T' are sampling period of transmitter and receiver, respectively. The relationship between transmitted signal $\boldsymbol{r(.)}$ and received signal $\boldsymbol{\hat{r}(.)}$ is

$$\hat{r}(mT') = r(t_d T + m(1 + t_\Delta)T), \quad m = 0,1,2,... \tag{4.2}$$

where $\boldsymbol{m}$ is the sample index. The clock frequency offset degrades the system performance due to a slow shift relative to each other. The problem has been analyzed by several authors [12,13,14,15].

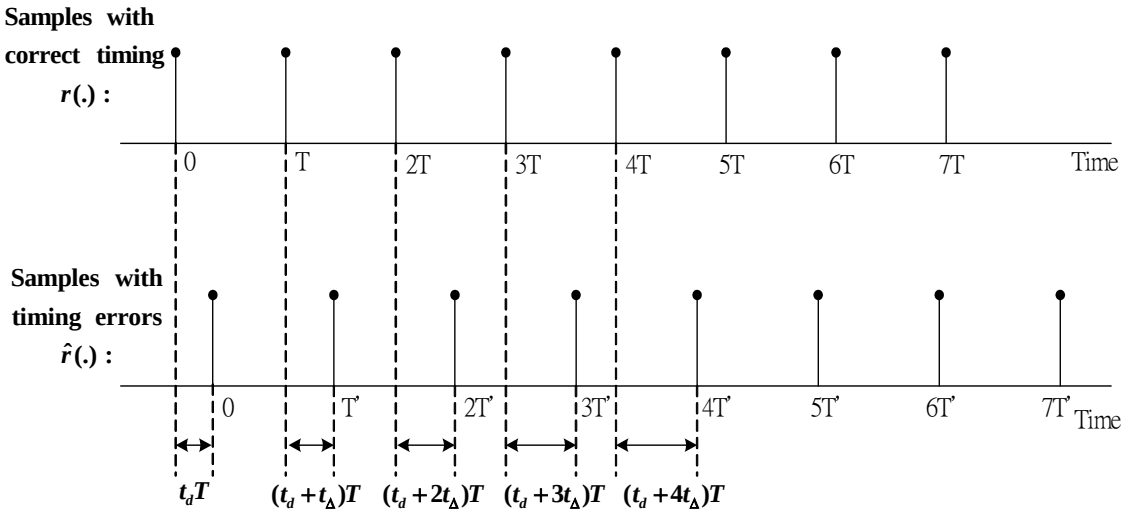


Figure 4.1 Sampling frequency mismatch between transmitter side and receiver side

Assume that t_d equals to zero for simplicity, then the result $R_{n,k}$ of the k^{th} tone after FFT is [11,16]

$$R_{n,k} = e^{j2\pi knt_\Delta \frac{N+N_g}{N}} H_{n,k} X_{n,k} \operatorname{sinc}(\pi k t_\Delta) + N_{n,k} + N_{t_\Delta}(n,k) \tag{4.3}$$

where n is the DMT symbol index, k is the tone index, $H_{n,k}$ is the channel response, $X_{n,k}$ is the transmitted data, Δf is the tone spacing, N is the block size of FFT , N_g is the length of GI, $N_{n,k}$ is additive white noise and $N_{t_\Delta}(n,k)$ is the inter-carrier interference. The first term of Eq. (4.3) shows that the effect of a sample phase shift results in a rotation of the FFT outputs. This is so-called the "delay-rotor property". The interference $N_{t_\Delta}(l,k)$ due to clock offset can be approximated by [16]

$$P_{t_\Delta} \approx \frac{\pi}{3}(kt_\Delta)^2 \tag{4.4}$$

Therefore, the degradation grows at a rate proportional to the square of the product of the offset t_Δ and tone index *k*. It means the outermost sub-carrier is most affected.

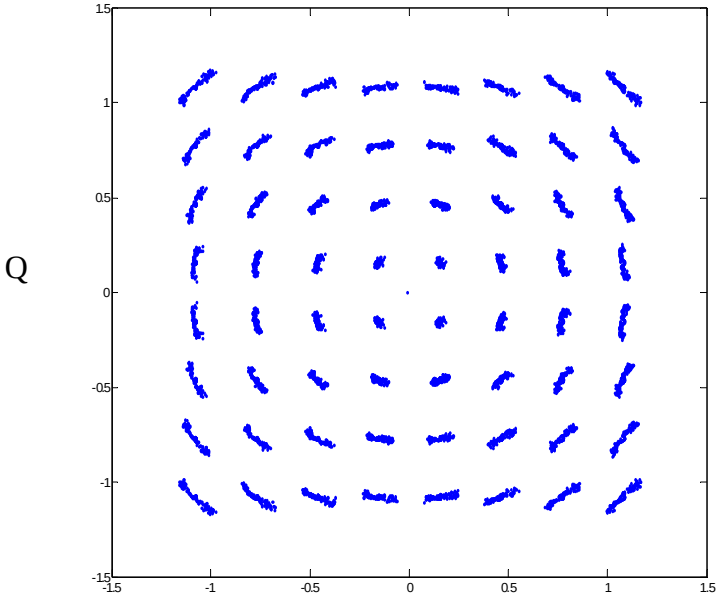
The degradation can be also expressed as SNR loss in dB [13] and shown in the following approximation.

$$D_n \approx 10\log_{10}(1+\frac{\pi^2 E_s}{3N_o}(kt_\Delta)^2) \tag{4.5}$$

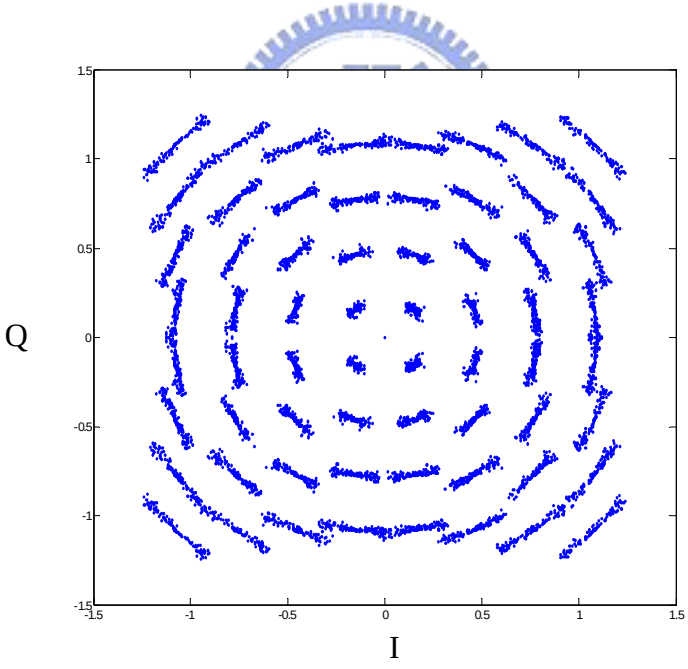
Once the clock is synchronized, the residual frequency offset would always be small. Thus, the interference caused by residual clock offset can be ignored. However, there still exists a significant phase rotation problem, namely the first term of Eq. (4.3):

$$e^{j2\pi kt_\Delta n\frac{N+N_g}{N}} \tag{4.6}$$

Though t_Δ is always small, the phase rotation increases with symbol number. The system performance will be severely damaged if there is no tracking for clock offset. Figure 4.2 (a) (b) show the constellation of two received symbols after FFT under clock offset = 2 and 4 ppm, respectively, while random 64-QAM signals are transmitted on each tone. As expected, the phase rotation is proportional to clock offset, and the outer constellation points are more seriously affected than the inner constellation points at the same phase rotation. Figure 4.3 and Figure 4.4 show that the phase rotation is proportional to sub-carrier index and symbol index under a non-zero clock offset.



(a)



(b)

Figure 4.2 The constellation of the received 64-QAM data after FFT over 2 VDSL symbols with 4096 tones under clock offset of 2 ppm (a) and 4 ppm (b), respectively.

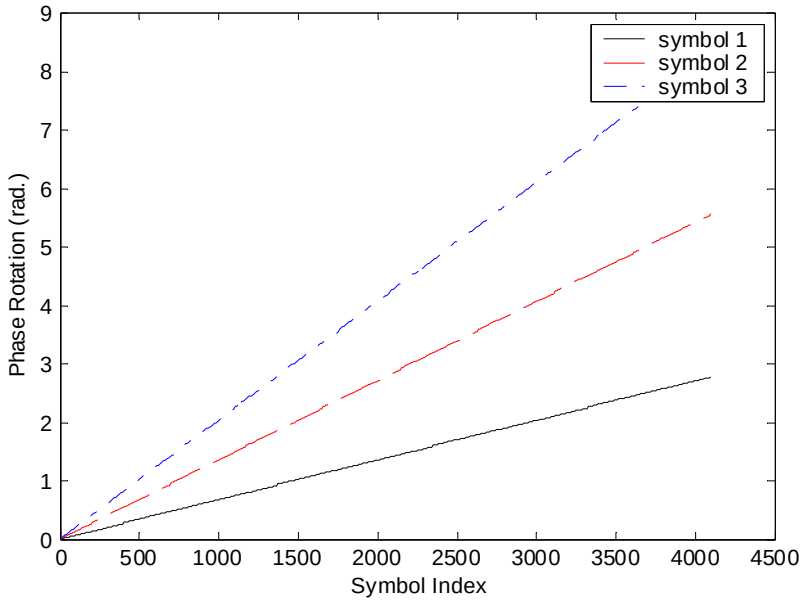


Figure 4.3 Phase rotations of symbol 1, 2, 3 versus tone index under 100 ppm clock offset.

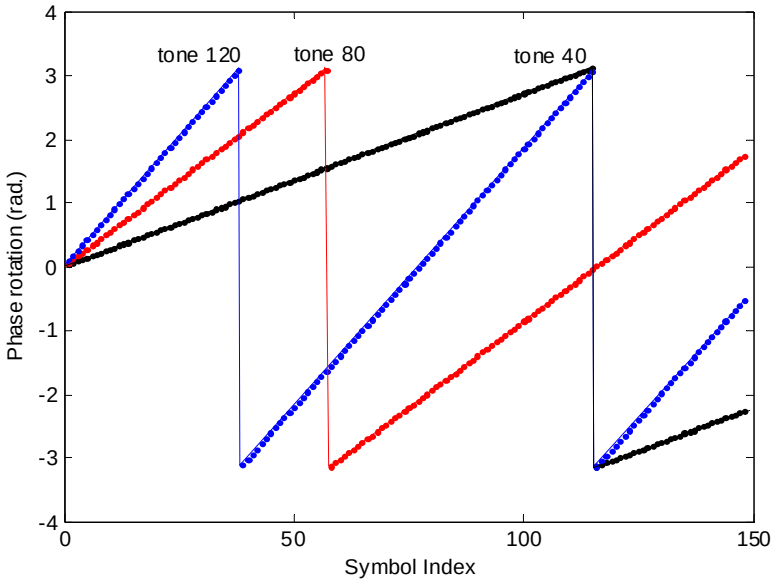


Figure 4.4 Phase rotations of tone 40, 80 and 120 versus symbol index under 100 ppm clock offset.

In [12,13,14,17,18,19,20], various sample timing correction schemes have been proposed. An overview of these sample timing correction methods is discussed from section 4.1.1 to section 4.1.4.

4.1.1 Continuous-time Sample Timing Correction [12,13,14]

If a synchronized sampling frequency is desired at receiver side, a possible solution is to adjust the sampling frequency of ADC in the continuous-time domain. As soon as the timing error information is obtained from the output of phase error detector, it will be low-pass filtered by a loop filter and then be fed back to a VCXO (Voltage-controlled crystal oscillator). The VCXO is used to control the sampling frequency of ADC. Figure 4.5 shows how this scheme works. However, a VCXO usually has higher cost and higher noise jitter than a XO (Crystal oscillator).

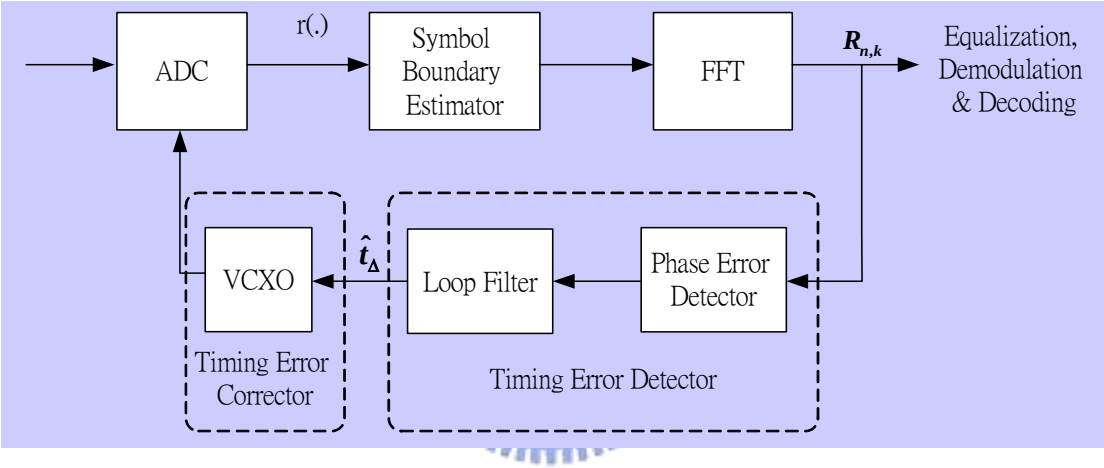


Figure 4.5 Continuous-time sample timing correction

4.1.2 Discrete-time Sample Timing Correction [13,14,17,18,20]

With a free-running oscillator, timing correction can be done by some discrete-time signal processing techniques on sample sequence. The timing correction is usually performed by an interpolator, as shown in Figure 4.6. A simple interpolator may be a finite impulse response filter (FIR) that produces a fractional delay. The interpolator coefficients are time-varying, because they depend on the timing error to correct. Due to the time-varying filter, the FFT output data suffers a time-varying phase rotation and attenuation. Such distortion introduces additional noises, especially on high frequency tones that are close to the Nyquist frequency. The noise can be

reduced by over-sampling the received signal or by using a high-order interpolator. However, due to the high sampling rate and high computation complexity, this scheme might not be suitable for very high-rate systems.

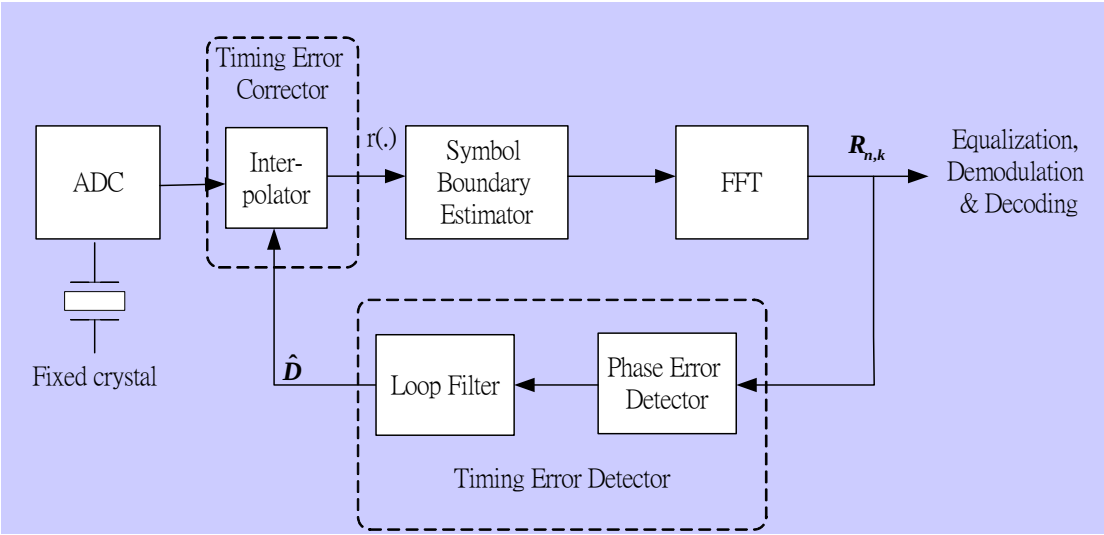


Figure 4.6 Discrete-time sample timing correction

4.1.3 Frequency-Domain Correction [14]

As mentioned in section 4.1, Eq. (4.3) shows the delay-rotor property, which means that this timing correction can be performed in the frequency domain by rotating the FFT outputs as shown in Figure 4.7. The rob/stuff block is responsible for correcting the integer part of estimated timing error $\hat{D}$, usually ± 1 sample. The fractional part of estimated timing error $\hat{D}$ is corrected by a rotor after FFT. Therefore, no over-sampling is needed and the timing error correction can be done by a single complex multiplication. However, this method works well only when a small frequency offset is guaranteed. Accurate and expensive XOs are required at both transmitter and receiver sides.

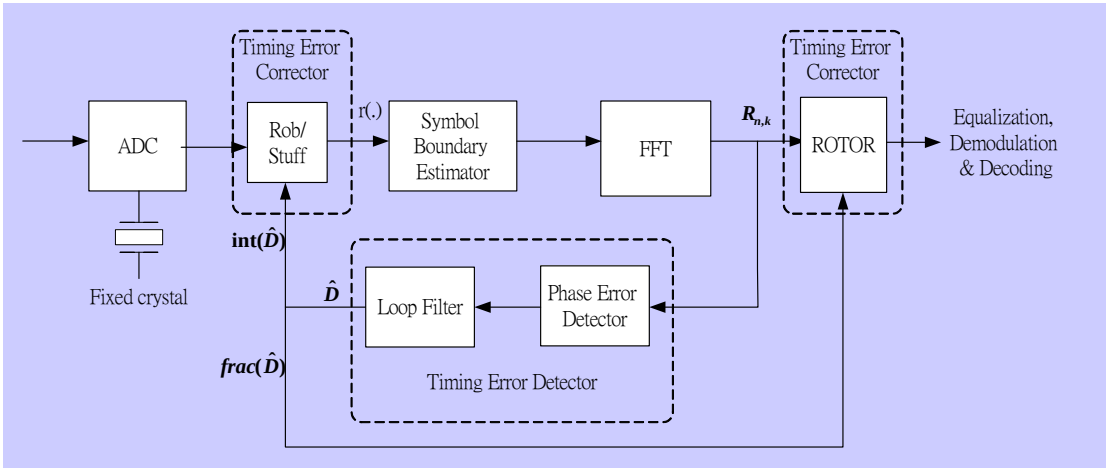


Figure 4.7 Frequency-domain sample timing correction

4.1.4 Hybrid Time/Frequency-Domain Correction [14,19]

Figure 4.8 depicts the time error correction based on a interpolation filter in time domain and a rotor in frequency domain [1,2,3]. An interpolator filter is used to correct the clock offset t_{Δ}. The rob/stuff block corrects the integer part of estimated timing error $\hat{D}$. Finally, the rotor provides phase rotation to obtain the desired accuracy of the timing error.

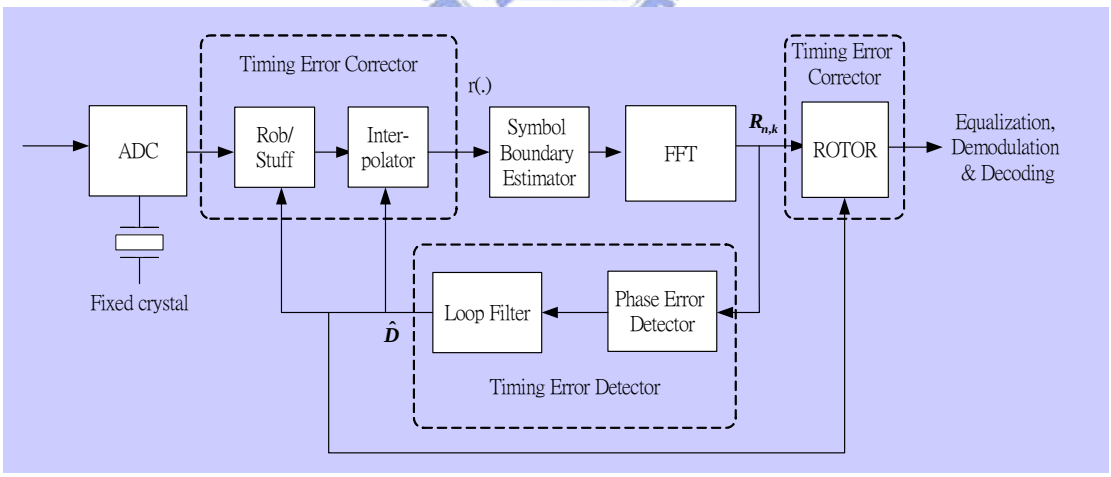


Figure 4.8 Hybrid time/frequency-domain correction

4.1.5 Comparisons of Sample Timing Correction Methods

Continuous-time sample timing correction is a straightforward method to synchronize the sampling timing. In this scheme, a VCXO with relative high cost and

high noise jitter is unavoidable. To reduce the cost, one might try a low-precision VCXO and correct the residual timing error in the discrete domain. However, a low-precision VCXO introduces additional noise. Continuous-time sample timing correction seems not suitable for a VDSL system, which is very sensitive to synchronization errors. All-digital timing correction schemes are more feasible than directly changing the sampling frequency with a VCXO. All-digital timing correction, namely discrete-time domain, frequency-domain and hybrid time/frequency-domain corrections, are performed by discrete signal processing techniques on non-synchronized input samples, as described in the previous sections. However, the cost of updating the rotor coefficients is quite high. If a look-up table were used for coefficient updating, it would require a large memory. Besides, the delay-rotor property is only valid while clock offset is very small.

In this work, we apply the discrete-time domain correction based on signal interpolation. The key drawback of this scheme is the high sampling rate and high computation complexity. The design goal is then set to find a suitable interpolator, given a fixed over-sampling ratio and the length of interpolator.

4.2 Interpolator Designs

Delaying a signal in time domain by an amount D can be considered as a linear time-invariant filtering, as shown in Eq. (4.7) and Figure 4.9.

$$H_{ideal}(e^{j\omega}) = e^{-j\omega D}, \text{ where } \omega = 2\pi f \tag{4.7}$$

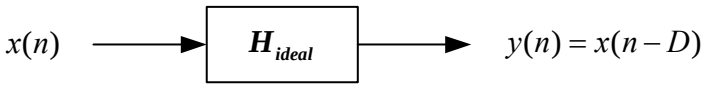


Figure 4.9 An ideal delay system

The desired frequency response is specified by both the magnitude and the phase response as:

$$|H_{ideal}(e^{j\omega})|=1 \text{ for all } \omega \tag{4.8}$$

$$\arg\{H_{ideal}(e^{j\omega})\} = -D\omega \tag{4.9}$$

An approximation to $H_{ideal}(e^{j\omega})$ can be obtained by minimizing the *p-th* order cost function E_p of the frequency-domain error function $E(e^{j\omega})$ as

$$E(e^{j\omega}) = H(e^{j\omega}) - H_{ideal}(e^{j\omega}) \tag{4.10}$$

$$E_p = \frac{1}{\pi}\int_0^{\pi} |E(e^{j\omega})|^p d\omega, \text{ where } p = 1,2,3,...,\infty \tag{4.11}$$

A straightforward solution is to minimize the cost function in the least-square error sense, which means *p* equals to 2. The cost function becomes

$$\begin{aligned} E_{LS} &= \frac{1}{\pi}\int_0^{\pi} |E(e^{j\omega})|^2 d\omega \\ &= \frac{1}{\pi}\int_0^{\pi} |H(e^{j\omega}) - H_{ideal}(e^{j\omega})|^2 d\omega \end{aligned} \tag{4.12}$$

In some cases, the desired frequency response does not need to be defined explicitly in the whole Nyquist band. By defining the response only in the desired frequency band $[0, \omega_p]$, the general least-square cost function can be rewritten as [21,22]

$$E_{Gen.LS} = \frac{1}{\pi}\int_0^{\omega_p} |H(e^{j\omega}) - H_{ideal}(e^{j\omega})|^2 d\omega \tag{4.13}$$

If the order of cost function is set to $p \to \infty$, then the resulting cost function will lead to a minimax solution [21,22]. The minimax solution is unique and equal to the equirriple solution. Equirriple approximation has the advantage of keeping the peak error value in a given limit, as shown in Eq. (4.14). The equirriple solution can be found by some iterative algorithms [23].

$$E_{\max} = \min_{p\to\infty}\{E_p\} = \min\left\{\max_{\omega\in[0,\omega_p]}\left\{\left|E(e^{j\omega})\right|\right\}\right\} \tag{4.14}$$

4.2.1 Resampling on Reconstructed Continuous-time Signals

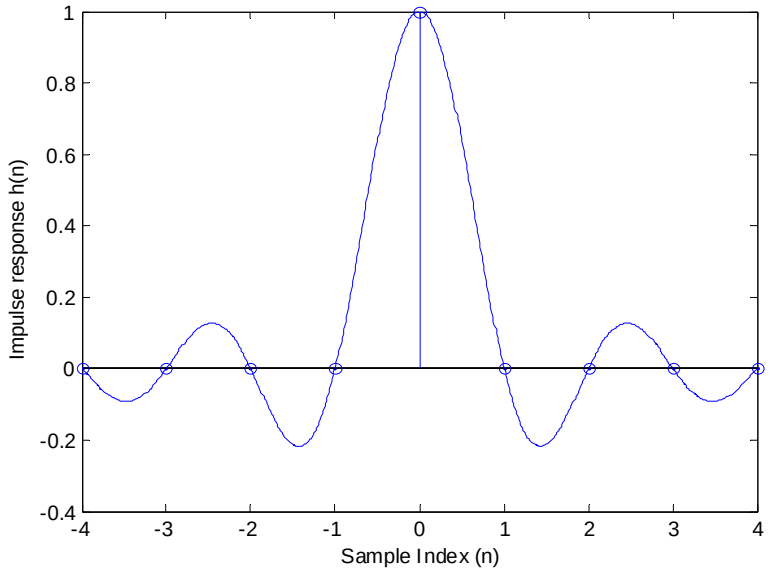
Signal interpolation is conceptually equal to a resampling operation on reconstructed signals in continuous-time domain. To obtain the reconstructed continuous-time signals, sinc interpolation and polynomial-based interpolation are frequently used.

4.2.1.1 Windowed Sinc Delay Interpolator [21,22]

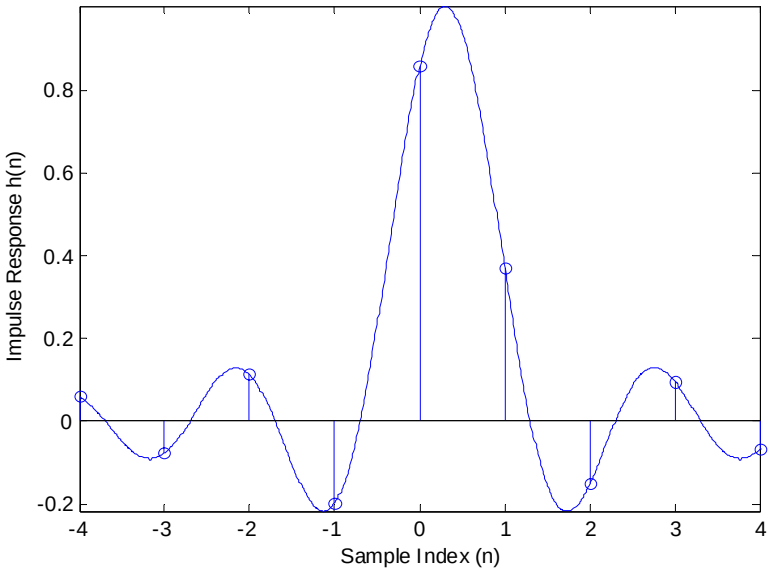
By minimizing the least-square cost function, the corresponding impulse response of $H(e^{j\omega})$ can be found by performing inverse Fourier transform.

$$h(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{-j\omega D}\cdot e^{j\omega n}\,d\omega = \mathrm{sinc}(n-D), \quad -\infty \le n \le \infty \tag{4.15}$$

Unfortunately, this impulse response is infinitely long and non-causal. Figure 4.10 shows the impulse response *h*(*n*) (Eq. 4.15) with delay (a) D=0 sample and (b) D=0.3 samples. In practice, a causal and finite length approximation is desired.



(a)



(b)

Figure 4.10 Impulse response of a sinc interpolator with the delay (a) D=0.0 (b) D=0.3

4.2.1.1.1 Rectangular-Windowed Sinc Delay Interpolator

A finite length approximation can be found by truncating Eq. (4.15) with an L-point rectangular window [23], as in Eq. (4.16).

$$w(n) = \begin{cases} 1, & 0 \le n < L-1 \\ 0, & \text{otherwise} \end{cases} \tag{4.16}$$

The resulting finite impulse response is expressed as Eq. (4.17)

$$h(n) = w(n-D)\,\mathrm{sinc}(n-D) = \begin{cases} \mathrm{sinc}(n-D), & 0 \le n < L-1 \\ 0, & otherwise \end{cases} \quad (4.17)$$

where L is the FIR filter length. Figure 4.11 shows a causal interpolation FIR with L=7 and D=3.3.

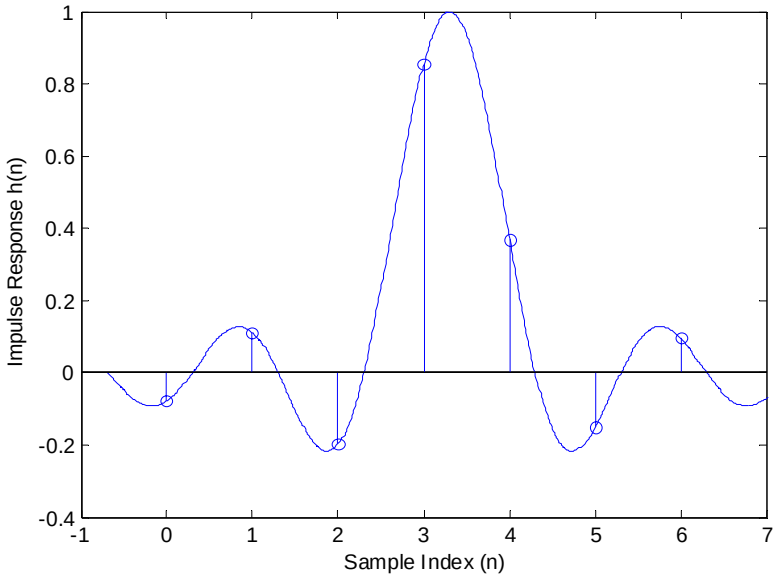


Figure 4.11 A causal interpolation FIR with L=7 and D=3.3

Rectangular window has the narrowest main-lope among all type of windows, which gives a sharpest transition band. However, it has a well-known bad feature, i.e., Gibbs Phenomenon. The peak error value of frequency response is nearly held constant regardless of filter lengths. To reduce the Gibbs Phenomenon, several time-domain window functions are described below.

4.2.1.1.2 Hamming-Windowed Sinc Delay Interpolator [24]

By tapering the window smoothly to zero, the side-lopes are greatly reduced at the price of wider transition band. A commonly used window is Hamming window, which is represented as

$$w(n)=\begin{cases}0.54-0.46\cos(2\pi\dfrac{n}{L-1}), & 0\le \mathrm{n} < L-1\\ 0, & \text{otherwise}\end{cases} \tag{4.18}$$

4.2.1.1.3 Kaiser-Windowed Sinc Delay Interpolator and Chebyshev-Window Sinc Delay Interpolator [23,24,25]

Given the filter length, Kaiser window and Chebyshev window provide more flexibility than other windows, because a trade-off between main-lope width and side-lope amplitude can be achieved. The Kaiser window [24] is characterized by a shape parameter β, which controls the maximal side-lope level as in Eq. (4.19) and Eq. (4.20)

$$w(n)=\begin{cases}\dfrac{I_0\left(2\beta\sqrt{\dfrac{n}{L-1}-\left(\dfrac{n}{L-1}\right)^2}\right)}{I_0(\beta)}, & 0\le \mathrm{n} < L-1\\ 0, & \text{otherwise}\end{cases} \tag{4.19}$$

where $I_0(\beta)$ is the 0[th]-order Bessel function. If desired side-lope attenuation is α dB, then

$$\beta=\begin{cases}0.1102(\alpha-8.7), & \alpha>50\\ 0.5842(\alpha-21)^{0.4}+0.07886(\alpha-21), & 21\le\alpha\le 50\\ 0, & \alpha<21\end{cases} \tag{4.20}$$

Chebyshev window [23,24] minimizes the mainlobe width, while a particular sidelobe height is given. Note that its side-lobes all have the same height. The design procedure of a Chebyshev-windowed Sinc interpolator involves iterative optimization procedures. The Chebyshev window is easy to use because it is included in certain commercial mathematical program packages, such as MATLAB. For both window functions, a lower side-lope amplitude results in a wider main-lope width. Figure 4.12

shows the trade-offs between main-lope width and side-lope amplitude of a Kaiser window and a Chebyshev window. Figure 4.13 shows the described four window functions and their resulting frequency responses for a sinc function.

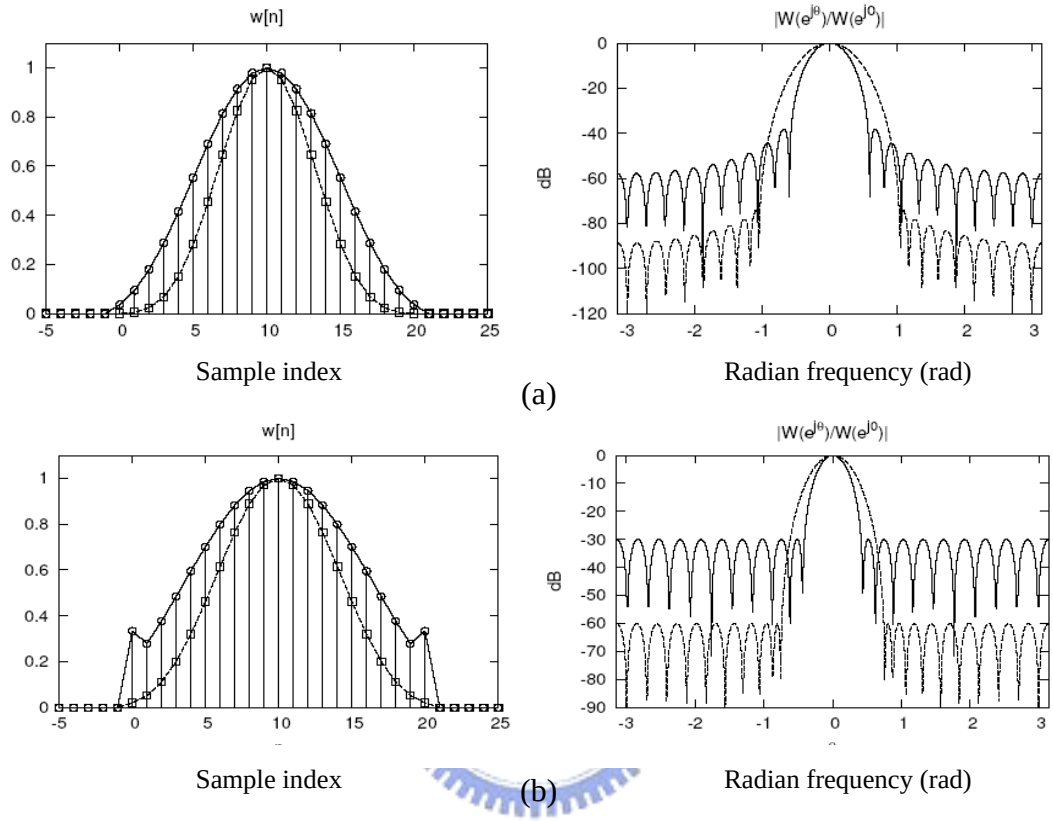


Figure 4.12 Trade-offs between main-lope width and side-lope amplitude of a Kaiser window (a) and a Chebyshev window (b).

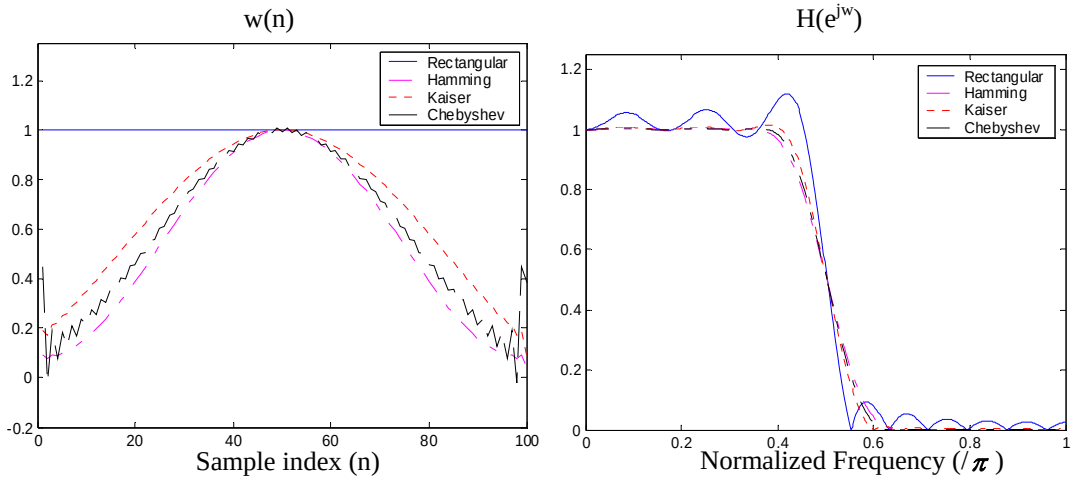


Figure 4.13 Various window functions and their frequency responses

4.2.1.2 Polynomial-based Interpolations [26,27]

In addition to formulating the design objective with the concept of linear filtering subject to minimum frequency errors, polynomial-based interpolations are also popular. Lagrange interpolation [26] and splines [27] have been suggested for fractional delay approximations. These techniques are not optimal in that their frequency responses deviate from the ideal delay response noticeably. However, they usually have good responses at low frequencies.

4.2.1.2.1 Lagrange Interpolation [26]

As shown in Figure 4.14, Lagrange interpolating polynomial is a polynomial of degree (*L-1*), which passes through *L* known sample points. The equation and the fractional delay filter corresponding to Lagrange interpolation are shown in Eq. (4.21) and Eq. (4.22) respectively.

$$\tilde{r}(DT) = \sum_{n=0}^{L-1} r_n \prod_{\substack{k=0 \\ k \neq n}}^{L-1} \frac{DT - kT}{nT - kT} \tag{4.21}$$

$$h(n) = \prod_{\substack{k=0 \\ k \neq n}}^{L-1} \frac{D-k}{n-k}, \ \mathrm{n} = 0, 1, 2, ..., \mathrm{L}-1 \tag{4.22}$$

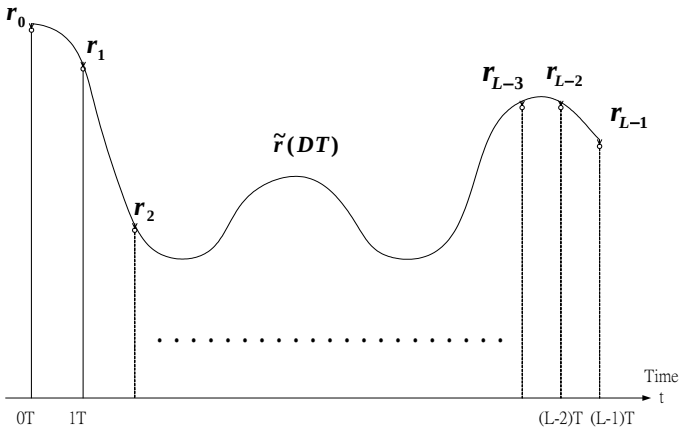


Figure 4.14 Lagrange Interpolation

4.2.1.2.2 Spline Interpolation [27]

Spline interpolation is developed under the consideration that general signals are continuous in their waveforms and also their derivatives in physical world. A popular spline interpolation method is “Cubic Spline”. It is a kind of third-order polynomial interpolation technique that assures 1st and 2nd derivatives of the common boundary point between two adjacent interpolation intervals are respectively the same. This matches the intrinsic property of general natural continuous signals whose derivatives are also continuous. The interpolation process starts with L-point received data $r_0 \sim r_{L-1}$. As shown in Figure 4.15, 3rd polynomials $\tilde{r}_i(t) = C_{3,i}t_i^3 + C_{2,i}t_i^2 + C_{1,i}t_i + C_{0,i}$ are used to fit the curve in the *i*-th area between two consecutives known samples. To solve the total 4*(*L*-1) unknown coefficients, 4*(*L*-1) equations are required. Assume that the first derivative (slope) and second derivative (curvature) of the common boundary point are respectively the same, totally we have 2*(*L*-1)+(*L*-2)+(*L*-2) known equations. The remaining two equations may be obtained by assuming that the very first and last points have zero slopes (nature type) or else. Due to the large amount of matrix operations for solving the linear equations, this method is impractical to be applied to fractional delay interpolation. Luckily, there is a special basis for the space of all splines, called “B-spline”. Its details are described in section 4.2.1.2.3.

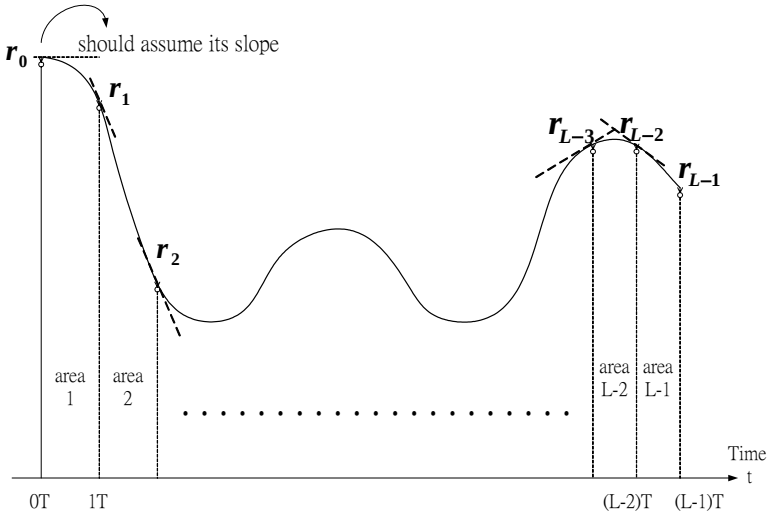


Figure 4.15 Cubic Spline Interpolation

4.2.1.2.3 B-spline Interpolation

B-splines provide a flexible way to make splines with arbitrary order of continuity. The simplest way to construct a B-spline is to convolve a square function $h_1(t)$ with itself as many times as desired, as shown in Figure 4.16.

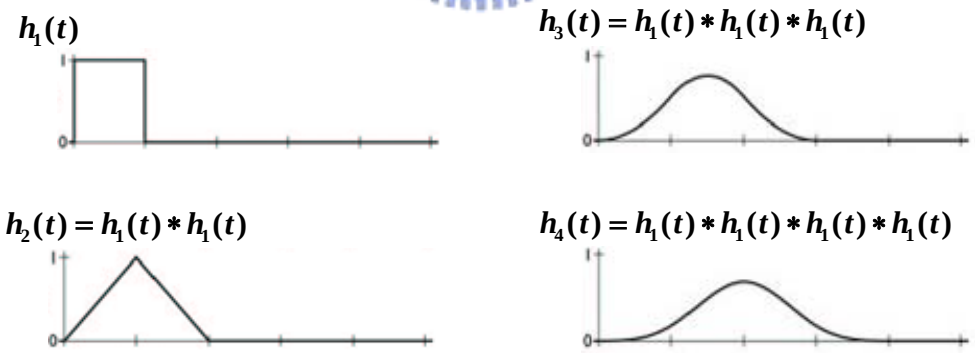


Figure 4.16 Construction of B-splines

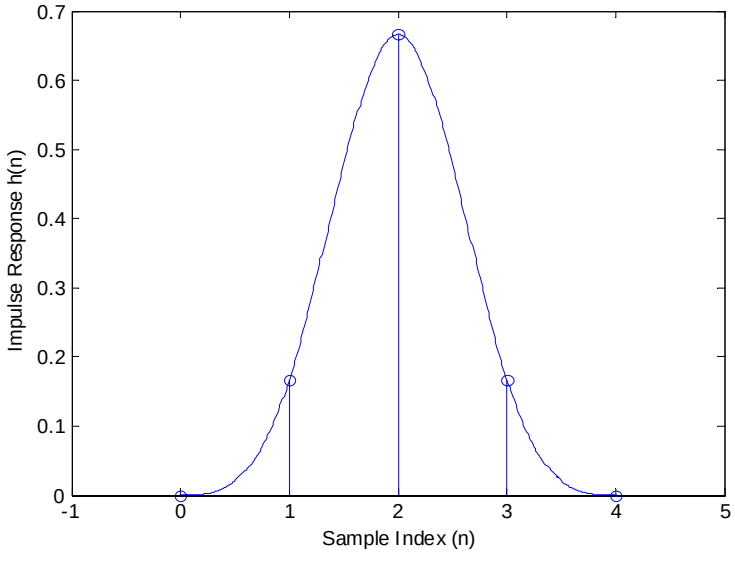
In the figure, $h_2(t)$, $h_3(t)$ and $h_4(t)$ are so-called “Linear B-spline”, “Qudratic B-spline” and “Cubic B-spline,” respectively, as described by Eq. (4.23), Eq. (4.24) and Eq. (4.25). Coefficients of the desired causal fractional delay FIR can be easily obtained by time-shifting and substituting t with corresponding values. Figure 4.17

shows the response $h(n)$ of a cubic B-spline interpolator with (a) D=2 and (b) D=2.3. Note that for time-domain signals, a B-spline interpolator is poor because the interpolated points will not be the same as input points even when the delay D equals to zero. However, its excellent frequency response, which has coherent magnitude responses and least phase error, makes it suitable for a DMT-based system.

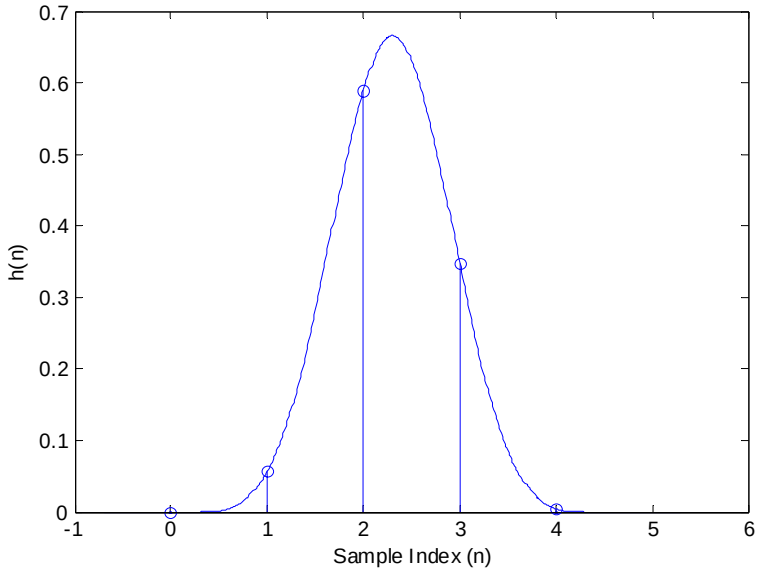
$$\text{Linear B-spline}: h_2(t) = \begin{cases} t+1, & -1 \le t \le 0 \\ 1-t, & 0 \le t \le 1 \end{cases} \tag{4.23}$$

$$\text{Quadratic B-spline}: h_3(t) = \begin{cases} \frac{1}{2}(t+\frac{3}{2})^2, & -\frac{3}{2} \le t \le -\frac{1}{2} \\ \frac{3}{4}-t^2, & -\frac{1}{2} \le t \le \frac{1}{2} \\ \frac{1}{2}(t-\frac{3}{2})^2, & \frac{1}{2} \le t \le \frac{3}{2} \end{cases} \tag{4.24}$$

$$\text{Cubic B-spline}: h_4(t) = \begin{cases} \frac{2}{3}+\frac{1}{2}|t|^3-t^2, & 0 \le |t| \le 1 \\ \frac{1}{6}(2-|t|)^3, & 1 \le |t| \le 2 \end{cases} \tag{4.25}$$



(a)



(b)

Figure 4.17 A cubic B-spline interpolator with (a) D=2 and (b) D=2.3

4.2.2 Discrete-time Over-sampling Approach [21,28,29]

The previous windowed-sinc and polynomial-based FD interpolators are realized by resampling of the reconstructed analog functions from the original discrete samples. Conceptually, they are continuous-time interpolation operations. Obviously, pure digital signal interpolation is a fundamental and popular approach for FD interpolation. That is, to achieve high-resolution FD, one can perform digital "over-sampling" operations [28,29] to solve the desired fraction-delayed samples. With *N*-times oversampling, we can split the unit delay into N divisions, as shown in Figure 4.20. In the figure, $H(z)$ is an *N*th-band low-pass filter. Note that the proto-type filter $H(z)$ can be obtained by various optimization techniques, such as least-square and equirriple approximations.

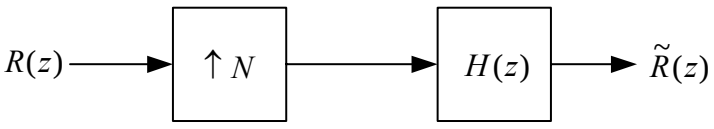
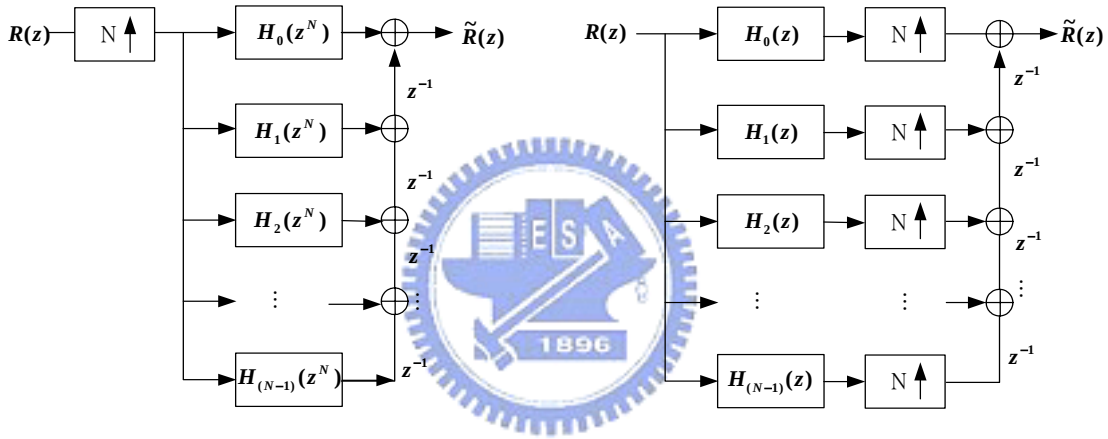


Figure 4.18 Digital LTI *N*-times over-sampling

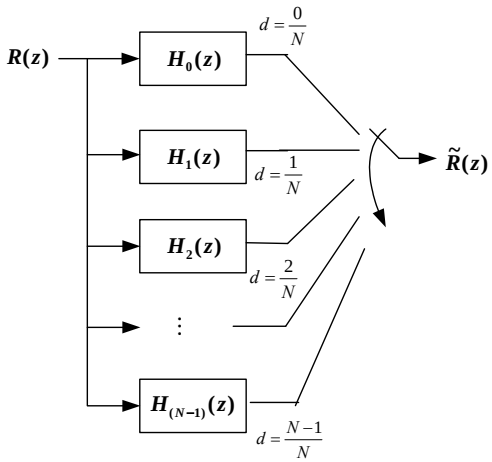
Applying polyphase decomposition techniques, an N-branch polyphase interpolator can be formed. In what follows, z-transforms are used to derive a polyphase interpolator. Note that each branch is an *L*-tap filter, which is shown in Eq. (4.26) and Figure 4.21.

$$\begin{aligned} H(z) &= h(0) + h(1)z^{-1} + ... + h(LN-1)z^{-(LN-1)} \\ &= H_0(z) + z^{-1}H_1(z^N) + z^{-2}H_2(z^N) + ... + z^{-(N-1)}H_{N-1}(z^N) \end{aligned} \tag{4.26}$$

where $H_i(z^N) = h(i) + h(i+N)z^{-N} + h(i+2N)z^{-2N} + ... + h(i+(L-1)N)z^{-(L-1)N}$ and $i \in \{0,1,...,(N-1)\}$.



(a) (b)



(c)

Figure 4.19 Original form of a polyphase interpolator (a), a more efficient version (b) and an equivalent polyphase interpolator with an output commutator (c)

In the upsampling polyphsae structure, the k[th] branch is responsible for generating the samples with fractional delay of k/N, where $k \in \{0,1,...,(N-1)\}$, coefficients of the k^{th} branch polyphase filter H_k is selected. Although the upsampling operation is computationally intensive especially for high-resolution fractional delays, the complexity can be much reduced by only considering those required FD sampled. The upsampling-based digital FD interpolators are disadvantageous in that it can not realize arbitrary delay. If the desired delay d falls into an open interval $(k/N, k+1/N)$, the result is undefined. To solve this problem, a well-known approach [28] is to approximate each branch's polyphase filter coefficient as a p^{th}-order polynomial of fractional delay d, as shown in Eq. (4.27). Usually, the approximating polynomial of the filter coefficients is obtained by fitting them in the least-squares sense.

$$h(n) = \sum_{m=0}^{p} c_{m,n} d^m, \quad n \in \{0, 1, 2, ..., (L-1)\} \tag{4.27}$$

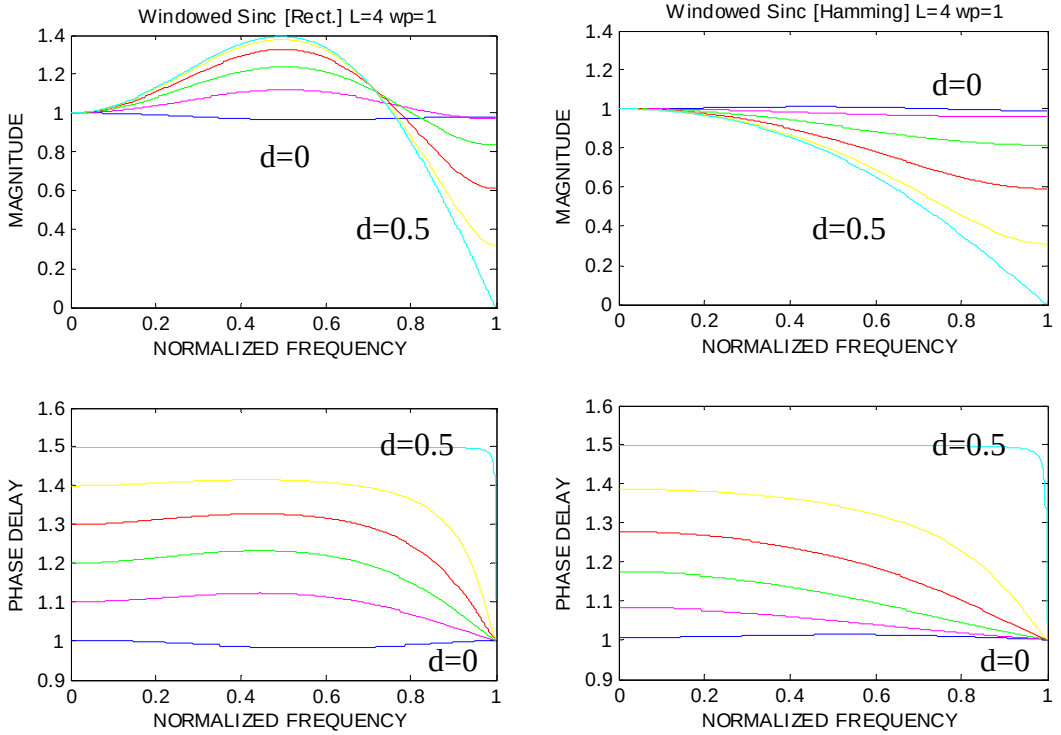
where $\boldsymbol{c_{m,n}}$ are real-valued coefficients of approximating polynomials. Then the resulting interpolator will be able to produce continuous delay. In addition, the well-known Farrow structure can be applied to the approximating polynomials for low-complexity realization, which will be shown in Section 5.1.2.

4.2.3 Frequency Responses of Various Interpolator Designs

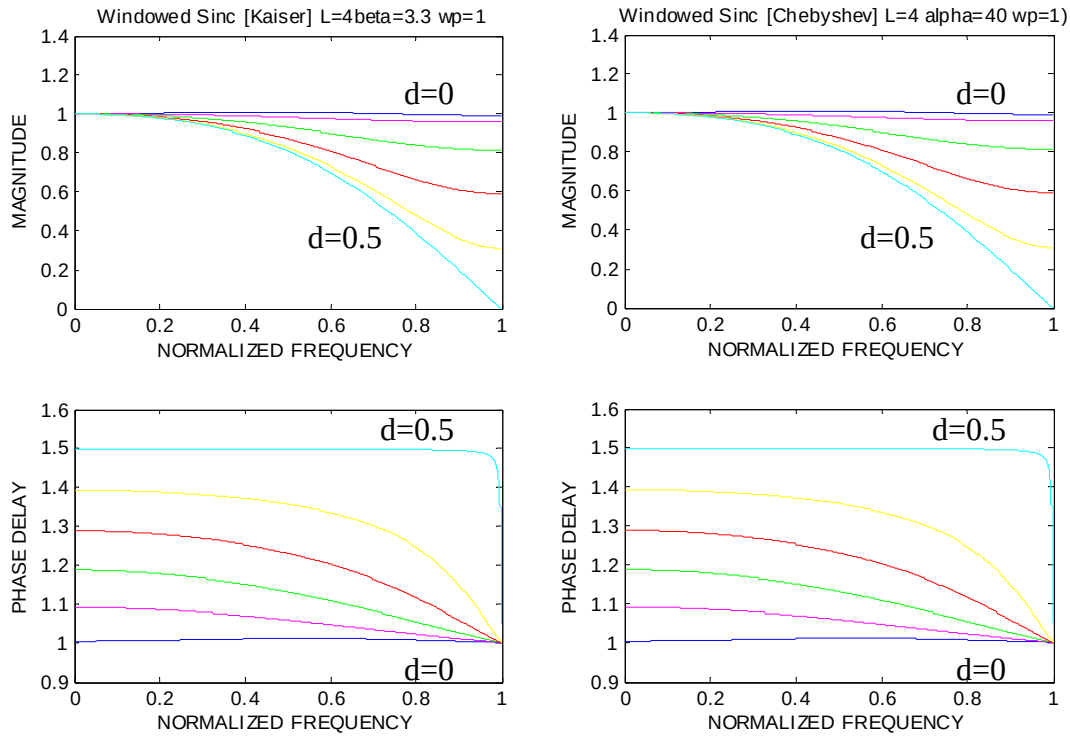
Designing an interpolator can be viewed as designing a finite impulse response that produces a fractional delay. Figure 4.18 (a) (b) (c) (d) shows the responses of 4-tap fractional-delay FIRs, which are designed by various windowed sinc functions. Figure 4.19 (a) (b) (c) (d) shows the responses of various 4-tap fractional-delay FIRs, which are designed by general least-squares approximation, equirriple approximation,

Lagrange interpolation and Cubic B-spline interpolation. Since the received data of a DMT system will be demodulated and decoded in frequency domain, the frequency response of the interpolator is important. Due to the time-varying coefficients, a fractional delay FIR has different frequency responses with respect to the fractional part d of input delay D in sample.

The magnitude dispersion and phase delay error will introduce additional noise. Therefore, an interpolator with small phase error and coherent magnitude response is desired. As shown in Figure 4.18, the rectangular-windowed sinc interpolator has the sharpest transition band but the most severe Gibbs Phenomenon. Other windowed sinc interpolators have lower peak errors but wider transition bands. Note that all the 4-tap windowed sinc interpolators have poor phase responses. In Figure 4.19, the general least-squares interpolator gains a better phase response than windowed sinc interpolators, and so is the equirriple interpolator. Lagrange interpolator is characterized by its good phase response and a flat magnitude response around radian frequency $\omega = 0$. Due to the non-zero tap weight, the output of a cubic B-spline interpolator will not be the same as input sample when fractional delay $d = 0$. As shown in Figure 4.19 (d), the frequency response is not all-pass when $d = 0$. It is not a perfect interpolator in time domain but results in coherent magnitude responses among various fractional delays in frequency domain. In addition, a cubic B-spline interpolator also has a good phase response. It is hard to determine which FD FIR is better by simply inspecting their frequency responses. Guidelines of interpolator design will be discussed later in section 5.3.

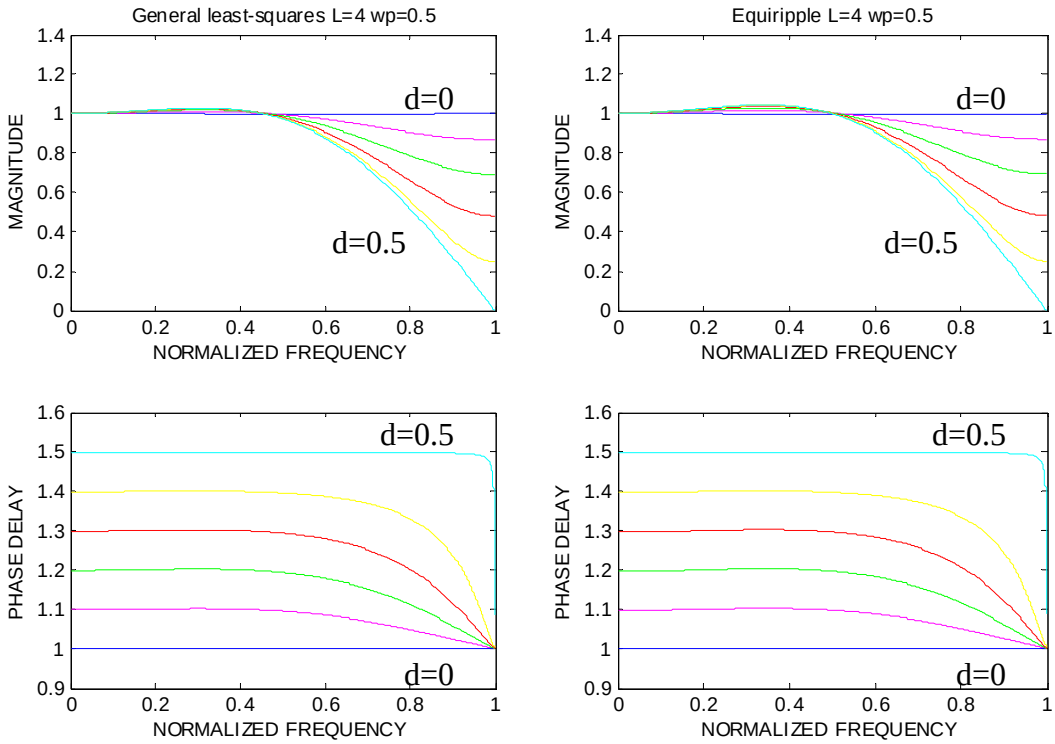


(a) (b)

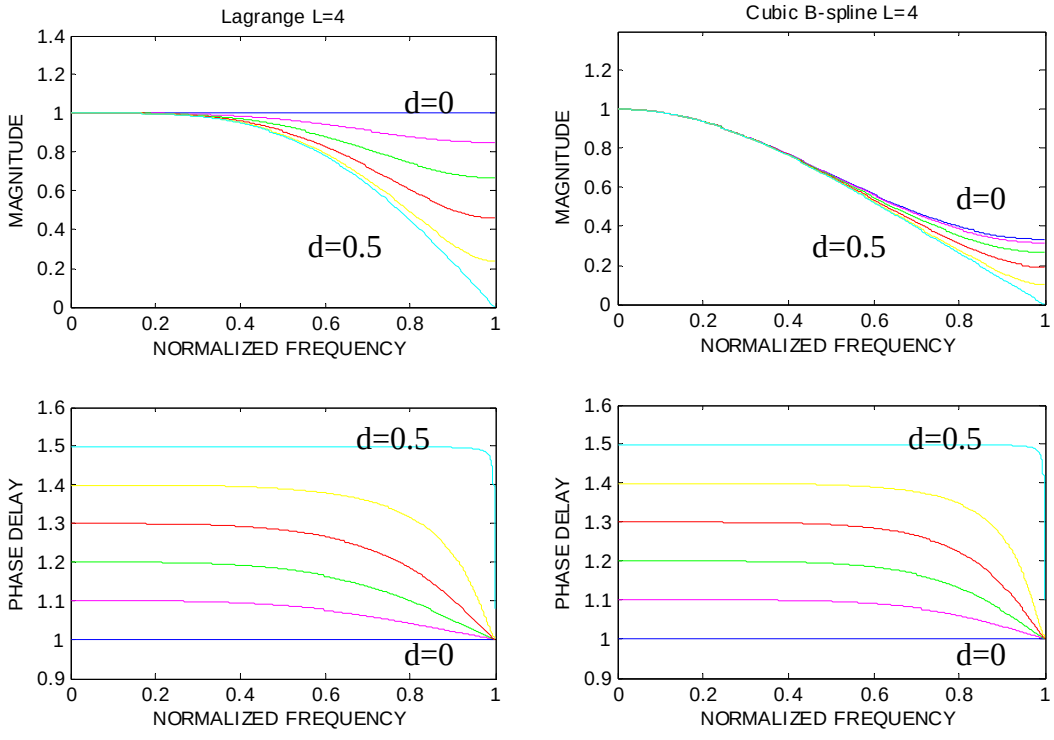


(c) (d)

Figure 4.20 Windowed Sinc FD FIR filters: (a) Rectangular window; (b) Hamming window; (c) Kaiser window; (d) Chebyshev window.



(a) (b)



(c) (d)

Figure 4.21 Various FD FIR filters: (a) General least-squares approximation; (b) Equirriple approximation; (c) Lagrange interpolation; (d) Cubic B-spline interpolation