Chapter 5
Special Filters for Audio Applications

5.1 Introduction

In DASP, there is a huge amount of literature about filtering systems designed for specific applications [1]-[79].

In this chapter we describe and deepen the study of some circuit structures, here called "special", commonly used in acoustic signal processing. The study of these architectures is fundamental for the realization of audio effects, the modeling of complex acoustic systems, the conversion of sampling frequency into non-rational ratios, and so on [26].

In particular, in the first part of the chapter are introduced specific filters called *comb-filter*, their multi-channel extension denoted *feedback delay networks* (FDN), the *all-pass* (AP) filters, also called universal comb filters and the implementation structures of *circular buffer delay lines*. In the second part are introduced the orthonormal filter architecture like the Kautz-Broome and Laguerre filters, and the warped signal processing methods. The third part of the chapter concerns fractional delay lines and finally digital oscillators are briefly introduced.

5.2 Comb Filters

Comb filters are so called because their amplitude response, which somehow resembles the shape of a comb, is characterized by a number of resonances and anti-resonances. In general, comb filters can be implemented as non-recursive FIR type or as recursive filters or IIR type.

5.2.1 FIR Comb Filters

The non recursive comb filter or FIR-comb has the structure shown in Fig. 5.1.

The input-output relationship is expressed with the following difference equation:

$$y[n] = x[n] + g \cdot x[n-D] \tag{5.1}$$

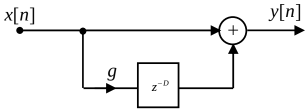


Fig. 5.1 Simple non-recursive comb filter.

while the transfer function (TF) turns out to be:

$$H(z) = 1 + gz^{-D} \tag{5.2}$$

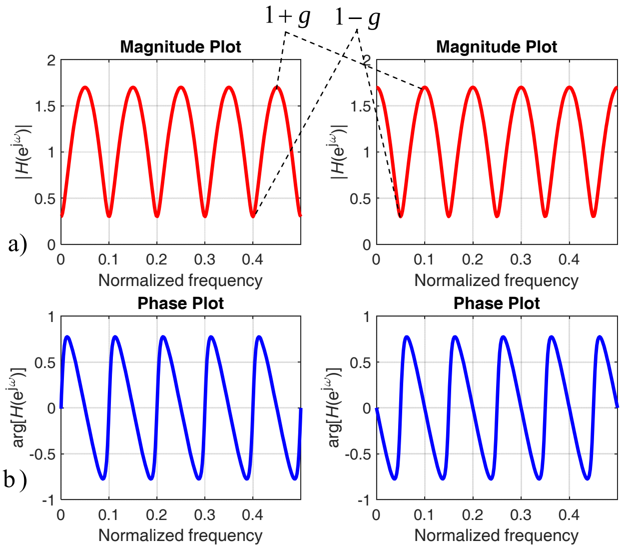


Fig. 5.2 Pole-zero, magnitude and phase response of non-recursive comb filter. a) $D = 10$ and $g = -0.7$. b) $D = 10$ and $g = 0.7$.

In case g =1, using De Moivre's formula, it is easy to verify that the frequency response of the filter is :

$$H(e^{j\omega}) = \sqrt{2\,(1 + \cos(\omega D))}$$

while the group delay is:

$$\tau_g(\omega) = \frac{D}{2}.$$

With reference to Fig.s 5.2 and 5.3, we can observe the particular structure of the frequency response characterized by D zeros (or $D/2$ pairs of complex-conjugate zeros) uniformly distributed on the unitary circle.

Therefore, the amplitude response results characterized by $D/2$ minimum points (zeros), equally spaced between the normalized frequency 0 and 0.5, which give to the frequency response a shape reminiscent of a comb.

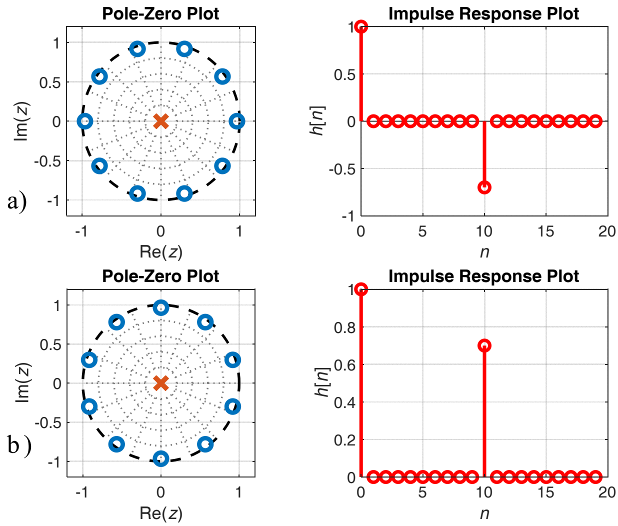


Fig. 5.3 Poles and zeros diagram of non-recursive comb filter. a) $D = 10$ and $g = -0.7$. b) $D = 10$ and $g = 0.7$.

5.2.2 IIR Comb Filters

The recursive comb filter or IIR-comb, has the structure shown in Fig. 5.4. Note that, in this case, the DL can be inserted in the feedforward or in the feedback branch. The input-output relations, for the topologies in Fig 5.4, are expressed with the following differences equation:

$$y[n] = x[n-D] - g \cdot y[n-D] \tag{5.3}$$
$$y[n] = x[n] - g \cdot y[n-D] \tag{5.4}$$

while the respective TFs turn out to be

$$H(z) = \frac{z^{-D}}{1 + gz^{-D}} \tag{5.5}$$
$$H(z) = \frac{1}{1 + gz^{-D}}. \tag{5.6}$$

Note that the network functions for the two topologies have the same denominator.

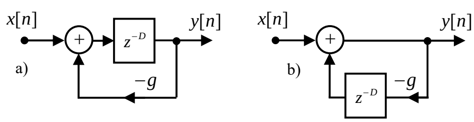


Fig. 5.4 IIR comb filters. a) DL in the feedforward branch. b) DL in the feedback branch.

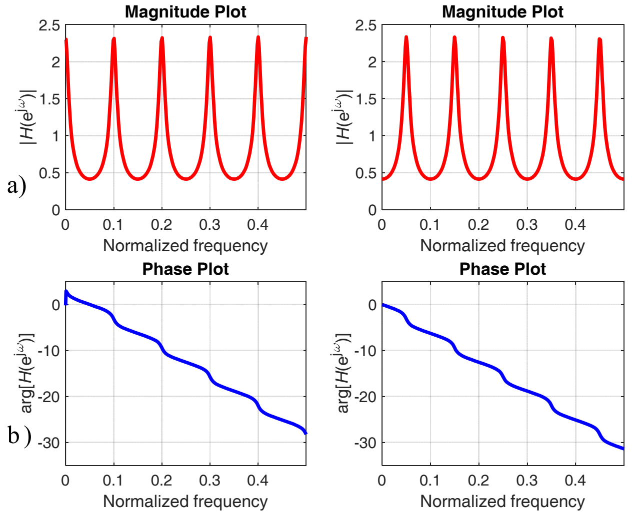


Fig. 5.5 Amplitude and phase response of the recursive comb filter (with DL in the feedforward branch) for: a) $D = 10$ and $g = -0.7$. b) $D = 10$ and $g = 0.7$.

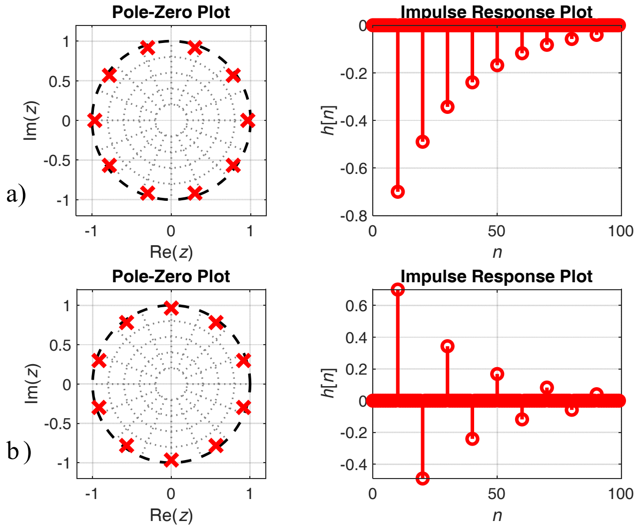


Fig. 5.6 Amplitude and phase response of the recursive comb filter (with DL in the feedforward branch) for: a) $D = 10$ and $g = -0.7$. b) $D = 10$ and $g = 0.7$.

Fig. 5.5 shows the frequency and phase response curves of TF in Eqn. (5.5). In Fig. 5.6 for the same TF, the poles-zeros diagram and the trend of the impulse response are shown.

5.2.3 Multidimensional Comb Filter: The Feedback Delay Networks

The single input-output comb structure, seen in the previous paragraphs, can be generalized to the vector case as multiple-input-multiple-outputs (MIMO) comb filter.

This model illustrated in Fig. 5.7 takes the name of *Feedback Delay Network* (FDN). FDNs have been proposed in order to model and implement artificial reverberators in an efficient and flexible way by Stautner and Puckette [70]-[72].

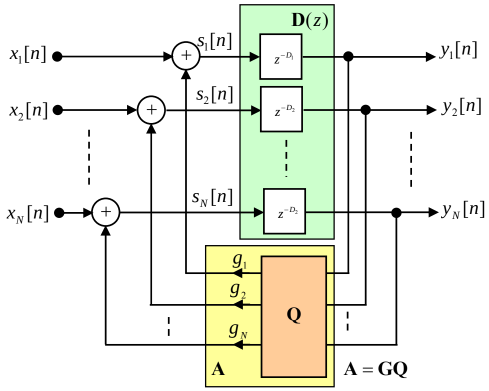


Fig. 5.7 General model of a multiple-input-multiple-outputs comb filter or *feedback delay networks* (FDN).

With reference to Fig. 5.7 in which $\mathbf{x}$ represents the input signals vector and $\mathbf{y}$ the output vector, the input-output relation results to be a multidimensional generalization of the comb filter equation (5.3). So we have

$$\mathbf{y}[n] = \mathbf{x}[n-D] + \mathbf{A} \cdot \mathbf{y}[n-D] \tag{5.7}$$

Where $\mathbf{A} = \mathbf{QG}$ is defined as a *transition* or *feedback matrix*, and $\mathbf{G} = \text{diag}(g_i)$, for $i=1, ..., N$, is diagonal matrix. In the frequency domain, the previous expressions become

$$\mathbf{Y}(z) = \mathbf{D}(z)\mathbf{X}(z) + \mathbf{AD}(z)\mathbf{Y}(z) \tag{5.8}$$

while the TF is

$$\mathbf{H}(z) = [1 - \mathbf{AD}(z)]^{-1}\mathbf{D}(z)$$

where

$$\mathbf{D}(z) \triangleq \begin{bmatrix} z^{-D_1} & 0 & \cdots & 0 \\ 0 & z^{-D_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & z^{-D_N} \end{bmatrix}.$$

Remark 5.1. Observe that, the FDN's output $y[i]$ are back-feeded to all inputs. In case the $\mathbf{Q}$ matrix (and consequently also the $\mathbf{A}$) were diagonal the FDN structure would behave like a bank of independent comb filters.

5.2.3.1 Space State FDN Representation and Stability

In case $D_1 = D_2 = \cdots = D_N = 1$, the (5.7) and (5.8) are equivalent to the standard state space model

$$\begin{aligned} \mathbf{s}[n+1] &= \mathbf{A}\mathbf{s}[n]+\mathbf{x}[n] \\ \mathbf{y}[n] &= \mathbf{s}[n] \end{aligned} \tag{5.9}$$

where $\mathbf{s}[n] \in \mathbb{R}^{N\times 1}$ is the state-variables vector.

In case the D_i delays are arbitrary the description can be made in terms of the *augmented state space model*, in which the position we have

$$\mathbf{s}[n+1] = \mathbf{A}\begin{bmatrix} s_1[n-D_1] \\ s_2[n-D_2] \\ \vdots \\ s_N[n-D_N] \end{bmatrix}.$$

Property 5.1. According to the Lyapunov's stability criterion, the stability of the FDN is ensured when the state vector norm is decreasing over time when the input signal is null [72]

$$\|\mathbf{s}[n+1]\| < \|\mathbf{s}[n]\|, \qquad \forall n \geq 0$$

From the the previous property, considering a L_2-norm, defined as $\|\mathbf{s}\|_{L_2} = \sqrt{s_1^2+s_2^2+...+s_N^2}$, and in terms of the transition matrix $\mathbf{A} = \mathbf{GQ}$, we have that

$$\|\mathbf{A}\mathbf{s}[n]\|_{L_2} < \|\mathbf{s}[n]\|_{L_2}.$$

If $\mathbf{Q}$ is an orthogonal matrix, stability is then guaranteed when $||<1$; that is

$$\mathbf{G} = \begin{bmatrix} g_1 & 0 & \cdots & 0 \\ 0 & g_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g_N \end{bmatrix}, \qquad |g_i| < 1$$

Property 5.2. FDN is said to be *lossless* if and only if the feedback matrix $\mathbf{A}$ has unitary module eigenvalues and linearly independent eigenvectors.

5.3 All-Pass Filters

An all-pass filter has a transfer function where the zeros are reciprocal of the poles. As a result, the amplitude response is flat (the zero module cancels the pole module) while, being the poles and zeros inside and outside the unitary circle respectively, the phases of the poles and zeros have the same sign. The phase, therefore, can assume even high values: the all-pass is not in fact at minimum phase filter[1]. This feature, as we will see later in the text, makes them particularly useful in many DASP application scenarios.

The TF of an all-pass filter of order N results to be

---

[1] A stable rational TF $H(z) = \frac{N(z)}{D(z)}$ is said *minimum-phase TF* if the zeros contribute positively to the phase: i.e. in the case of analog circuits the $N(z)$ roots are to the left of the imaginary axis; in the case of TD circuits the $N(z)$ roots are inside the unitary circle.

$$H(z) = \frac{z^{-N}A(z^{-1})}{A(z)} = \frac{a_N + a_{N-1}z^{-1} + ... + a_1 z^{-(N-1)} + z^{-N}}{1 + a_1 z^{-1} + ... + a_N z^{-N}} \tag{5.10}$$

it can be observed that the polynomial at the numerator is the *mirrored version* of the polynomial at the denominator.

In the time domain the input-output relationship is therefore

$$y[n] = a_N x[n] + ... + x[n-N] - a_1 y[n-1] - ... - a_N y[n-N].$$

In case the TF n takes the following expression

$$H(z) = \frac{g + z^{-D}}{1 + gz^{-D}} \tag{5.11}$$

the all-pass filter can be seen as a combination of a previously studied IIR and FIR comb filter and is also called the *universal comb filter* [4]. In this case, the input-output relationship simplified as

$$y[n] = gx[n] + x[n-D] - gy[n-D]$$

in this case the corresponding diagram (in direct form II) is the one shown in Fig. 5.8. The characteristic curves of the universal all-pass comb filter are shown in Fig.s 5.9 and 5.10.

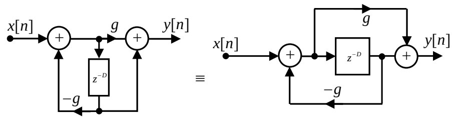


Fig. 5.8 All-pass or universal comb-filter, in direct form II.

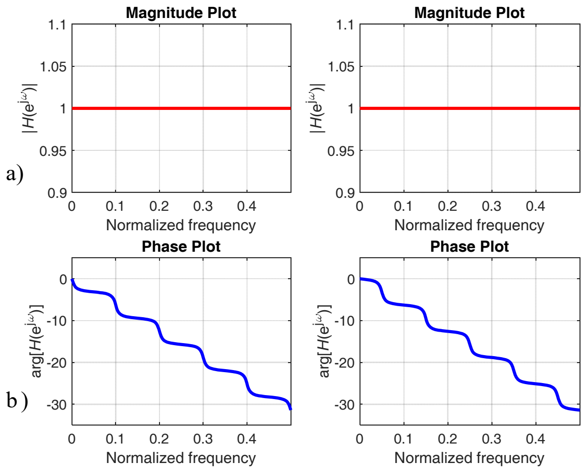


Fig. 5.9 Amplitude and phase response of the all-pass filter in Fig. 5.8 for: a) $D = 10$ and $g = -0.7$. b) $D = 10$ and $g = 0.7$.

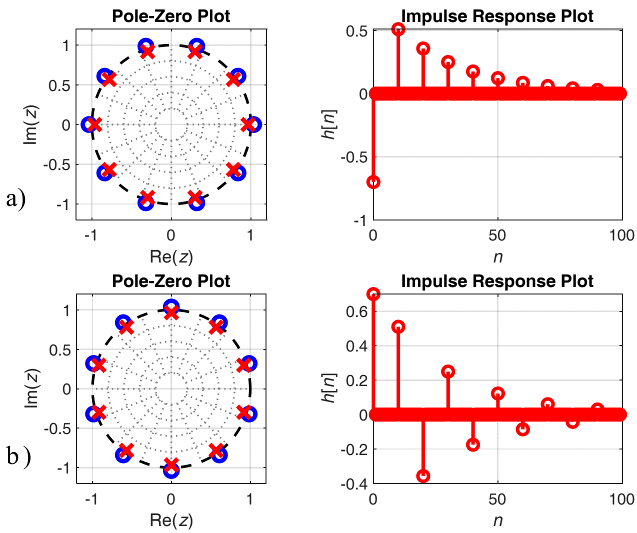


Fig. 5.10 Poles-zeros diagram and impulse response of the all-pass filter with FT (5.11) for a) $D = 10$ and $g = -0.7$. b) $D = 10$ and $g = 0.7$.

5.3.1 Nested All-Pass Filters

The nested AP filter architecture derives from the property the following property.

Property 5.3. If in an AP filter, e.g. of the type described by TF (5.10), every unit delay element z^{-1} is replaced with an FT $z^{-1}A(z)$, with $A(z)$ of AP type, the resulting FT is also an all-pass filter.

This property can be easily verified by considering for example an AP of the first order with a TF:

$$A_1(z) = \frac{z^{-1} + a_1}{1 + a_1 z^{-1}}$$

where each z^{-1} element is replaced by the following rule $z^{-1} \leftarrow z^{-1}A_2(z)$. The resulting TF can be easily calculated as

$$H(z) = \frac{z^{-1}A_2(z) + a_1}{1 + a_1 z^{-1} A_2(z)}.$$

Choosing $A_2(z)$ as a first order all-pass, as shown in Fig. 5.11

$$A_2(z) = \frac{z^{-1} + a_2}{1 + a_2 z^{-1}}$$

we obtain a total TF of the second order

$$H(z) = \frac{z^{-2} + (a_1 a_2 + a_2)z^{-1} + a_1}{1 + (a_1 a_2 + a_2)z^{-1} + a_1 z^{-2}}.$$

Note that, in the previous expression, as for Eqn. 5.10, the polynomial at the numerator is the mirrored version of the polynomial at the denominator. It results, therefore, that the $H(z)$ is of all-pass type.

Generalizing, starting from a TF AP $A_1(z)$, it is possible to create multiple nests where every z^{-1} element is replaced with the following rule

$$z^{-1} \leftarrow z^{-1}A_2(z) \leftarrow z^{-1}A_3(z) \leftarrow \cdots$$

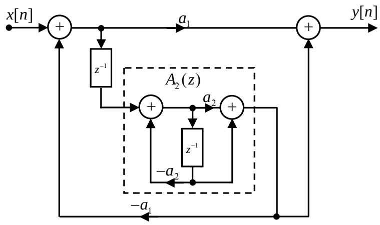


Fig. 5.11 Nested first order all-pass filters in Direct Form II.

Remark 5.2. Observe that, an all-pass filter can be implemented with "robust architectures" such as the *ladder* and *lattice* forms. For more information on such architectures, which are widely used in the audio field, see [62]-[66].

As an example, in Fig. 5.12-a) we report a way, based on a graphic-topological transformation, to derive the two-multiplier lattice structure.

In addition, in Fig. 5.12-b) is reported a N-order all-pass filter, implemented nesting N first-order sections . Note that, each first order section has only one multiplier [49].

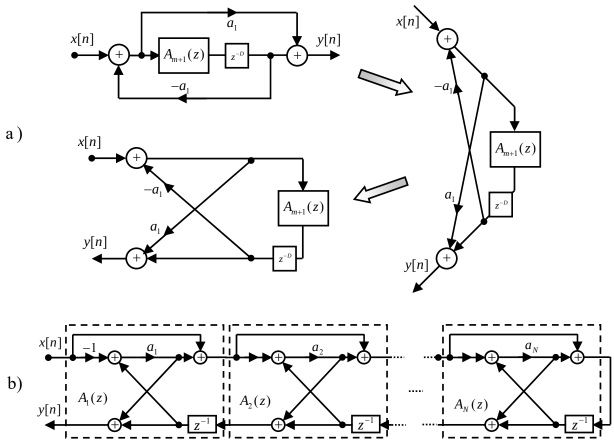


Fig. 5.12 Nested *lattice* AP filters. a) Derivation of the lattice form with simple graphic transformation from Direct Form II. b) N-order all-pass filter implemented with N first order all-pass nested filters (with a single multiplier).

5.4 Special Rational Orthonormal Filter Architecture

In many audio applications it is necessary to use filters to model acoustic phenomenon with long duration. However, a *room impulse response* (RIR) at the usual audio sampling frequencies can be very long and characterized by complex time-frequency structure. Although, this problem can be partially solved using IIR filters non recursive TFs have several contraindications such as difficult design, non-linear phase, stability problem when the poles are estimated as time-varying parameters, etc.

In general terms a RIR represents an “all-zeros” or FIR model, while considering conventional parametric models for room acoustics (see §3.2) the room transfer function (RTF) can be expressed in z-domain as in Eqn. (3.3) and factorized in term of poles and zeros (see §4.2.4, Eqn. (4.18)). Thus we can write

$$H_{RTF}(z) = \frac{\sum_{k=0}^{Q} b_k z^{-k}}{\sum_{k=0}^{P} a_k z^{-k}} = \left(\frac{b_0}{a_0}\right) \frac{\prod_{k=1}^{Q}(1-q_k z^{-k})}{\prod_{k=1}^{P}(1-p_k z^{-k})} \tag{5.12}$$

where q_k are the zeros that model the RIR’s anti-resonances and time delays, and p_k are the poles associated with room natural modes.

Rearranging the above we can write

$$H_{RTF}(z) = \sum_{k=1}^{N} \left[\frac{w_k^- - w_k^+ z^{-1}}{(1-p_k z^{-1})(1-p_k^\star z^{-1})}\right] = \sum_{k=1}^{N} P_k(z) \tag{5.13}$$

where $P_k(z)$ are second-order sections denoted *resonators*. The above expression corresponds to the parallel of N resonators as shown in Fig. 5.13 that represents a possible *physical parametric model* of the room. The relative RIR is a finite sum of exponentially decaying sinusoids, as in Eqn. (1.34), where the frequency and decay values are defined by the relative pole.

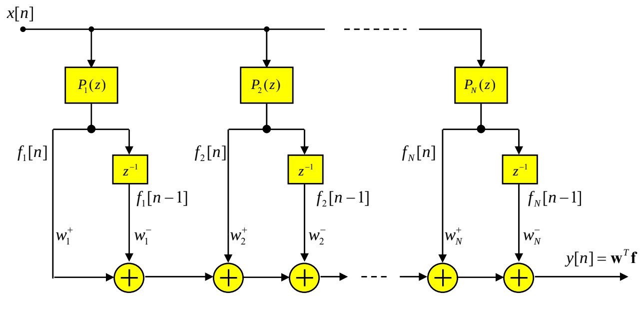


Fig. 5.13 Physical parametric RTF model as parallel of resonant circuits $P_k(z)$. The relative RIR is a finite sum of exponentially decaying sinusoids where the frequency and decay values are defined by the relative pole. (Modified form [17]).

In the past, one of the most critical aspects in the use of such parametric models was the problem of identifying their parameters. The recent progress in audio-specific adaptive algorithms, some of which will be presented and discussed in Chapter 6, and the increased computational power even in low-cost devices, have been the main architects of the renewed and recent interest in parametric structures for the computational analysis of the acoustic scene .

However, in addition to conventional parametric models of Eqn. (5.12), there has been renewed interest in the use of rational orthogonal structures, referred to as *orthogonal base filters* (OBF), which may be more appropriate for complex acoustic modeling. The main objective of this approach is to obtain a :

- compact representation of a IRs (in particular RIRs);
- parsimonious and also more controllable approximation of a IR useful for applications;
- efficient adaptive filtering algorithms.

This approach allows to map long impulse responses (IRs) in a small number of elementary TFs. These substructures are generally of reduced complexity, modular and allow a scalable approximation of the desired $H(z)$, for low cost hardware devices implementation, eliminating factors considered “less significant”, according to some criteria.

The following are briefly presented and discussed some methodologies for the factorization of a given TF $H(z)$, that originate from analog network synthesis techniques (later extended to discrete time), and some typical applications in the DASP scenario [6]-[13].

5.4.1 Kautz-Broome OBF Model

In 1954 Kautz in [6] developed a methodology to approximate the impulse response of a linear system, developing the $h(t)$ as a series of infinite terms, such that it could be approximated with truncation to the first N terms such as

$$h(t) = \sum_{i=1}^{N} w_i k_i(t) \tag{5.14}$$

where $k_i(t)$, hereafter denoted as *Kautz functions*, are defined as the inverse Laplace transforms of the TFs defiend as

$$K_i(s) = c_i \frac{\sqrt{p_i + p_i^\star}}{s + p_i} \prod_{n=0}^{i-1} \frac{s - p_n^\star}{s + p_n} \tag{5.15}$$

where $c_k \in \mathbb{C}$, $|c_k| = 1$, and such that the function $k_i(t)$ are orthonormal functions, i.e. we have that

$$\langle k_i(t), k_j(t) \rangle = \begin{cases} 1, & i = j \\ 0, & \text{elsewhere} \end{cases}$$

and defined as sums of damped exponentials and exponentially damped sinusoids. that form an orthonormal basis in Hilbert space [6]-[9].

Remark 5.3. Observe that, the Eqn. (5.15) can be written as $K_i(s) = V_i(s)A_i(s)$ where $V_i(s)$ is a low pass TF, and $A_i(s)$ an all-pass section, respectively defined as

$$V_i(s) = \frac{c_k \sqrt{p_k + p_k^\star}}{s + p_k}, \qquad A_i(s) = \frac{s - p_i^\star}{s + p_i}$$

such that the overall generative model of the impulse response $h(t)$ is equivalent of an analog ladder network as shown in Fig. 5.14.

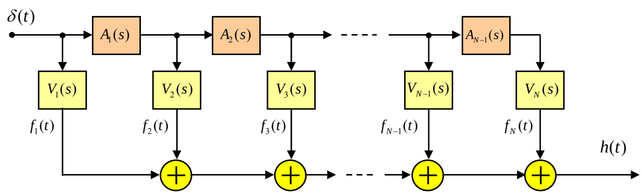


Fig. 5.14 Kautz ladder analog network generative model for the impulse response $h(t)$. The cells $V_k(s)$ are one pole low-pass filters, while the transversal cells $A_k(s)$ are all-pass TFs.

5.4.1.1 Discrete Time Kautz-Broome OBF Model

Later in 1965 Broome in [8], analyzed the problem over a discrete-time domain defining a set of orthonormal $k_i[n]$ functions that have exponential decay. However, the problem can be generalized by imposing the orthonormalization conditions of a given set of suitable functions. More recently in [11], has been proposed a z-domain TFs sequence defined as

$$F_{i,j}(z) = \frac{z^{-1}}{(1 - p_i z^{-1})^j}, \qquad i \in \mathbb{N}, \quad 1 \leq j \leq m_i \tag{5.16}$$

where p_i is a complex poles with $|p_i| < 1$, such that $p_i \neq p_k$, $i \neq k$, and where m_i is the multiplicity of pole p_i. Thus, applying Gram-Schmidt orthonormalization to the sequence of functions (5.16), the result is the so-called *discrete-time Kautz TFs*, also denoted as *Takenaka-Malmquist basis*[2], that are characterized by the following TFs

$$K_i(z) = \frac{\sqrt{1-p_i p_i^\star}}{1-z^{-1}p_i} \prod_{n=0}^{i-1} \frac{z^{-1}-p_n^\star}{1-p_n z^{-1}} \tag{5.17}$$

that have a identical structure to the analog TFs (5.15).

As for the analog case, with the above conditions, the rational TFs $K_i(z)$ represent a set of basis filters that are orthonormal in Hilbert space under mild conditions $\sum_i (1-|p_i|) = \infty$ [9], [18], thus we can write

$$H(z) = \sum_{i=1}^{N} w_i K_i(z) \tag{5.18}$$

wich will be called the discrete-time *Kautz-Broome OBF* (KB-OBF) model, where $K_i(z)$ are the so called discrete-time *Kautz-filters* or *-units*.

The the RTF can be modeled as a cascade of elementary rational orthogonal TFs such that the overall output is a linear combination of the Kautz units outputs $f_k[n]$

$$y[n] = \mathbf{w}^T \mathbf{f}$$

where $\mathbf{w}$ is the weights vector and $\mathbf{f}$ is the vector.

Remark 5.4. Observe that, the KB-OBF of Eqn. (5.17), can be written in the following recursive form [8], [12]

$$\begin{aligned} K_{2k-1}(z) &= C_1^{(k)}(1-a_1^{(k)})A^{(k)}(z) \\ K_{2k}(z) &= C_2^{(k)}(1-a_2^{(k)})A^{(k)}(z) \end{aligned}, \qquad k = 1, 2, ... \tag{5.19}$$

where the sub-sections $A^{(k)}(z)$ are second-order TFs defined as

$$A^{(k)}(z) = \frac{1}{(z-p_k)(z-p_k^\star)} \prod_{j=1}^{k-1} \frac{(1-p_j z)(1-p_j^\star z)}{(z-p_j)(z-p_j^\star)}. \tag{5.20}$$

In addition, to guaranty the orthonormality, for the parameters $a_1^{(k)}$, $a_2^{(k)}$, p_k, $C_1^{(k)}$ and $C_2^{(k)}$; the following conditions apply

$$(1+a_1^{(k)}a_2^{(k)})(1+p_k p_k^\star) - (a_1^{(k)}+a_2^{(k)})(p_k+p_k^\star) = 0 \tag{5.21}$$

$$C_j^{(k)} = \sqrt{\frac{(1-p_k^2)(1-p_k^{\star 2})(1-p_k p_k^\star)}{(1+a_j^{(k)2})(1+p_k p_k^\star) - 2a_j^{(k)}(p_k+p_k^\star)}}, \quad \text{for} \quad j = 1,2. \tag{5.22}$$

[2] Note that, this sequence of orthonormal functions was originally derived in the 1920s by Takenaka and Malmquist (see for example [11] and the references inside) and some-time will henceforth be referred to as the Takenaka–Malmquist functions.

The functions $K_k(z)$, $k = 1, 2, \ldots$, are usually called the *discrete Kautz functions.*

5.4.1.2 Orthonormal Filters Architectures

As for the analog ladder network in Fig. 5.14, to facilitate development, as suggested in [10] we can write $K_i(z) = V_i(z)A_i(z)$, where the TFs $V_i(z)$ are one pole low-pass filters defined as

$$V_i(z) = \frac{z^{-1}\sqrt{1-|p_i|^2}}{1-p_i z^{-1}}$$

and the TFs $A_i(z)$ are all-pass cells defined as

$$A_i(z) = \frac{z^{-i} - p_i^\star}{1-p_i z^{-i}}$$

where p_i, $i = 1, \ldots, N$ are the TF parameters, and where the term $\sqrt{1-p_i p_i^\star}$ is a normalization factor such that the filters $V_i(z)$ has unit gain at DC.

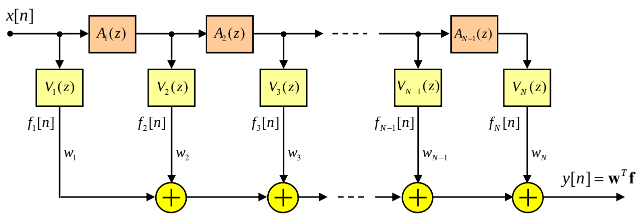


Fig. 5.15 Kautz discrete-time ladder filter for the representation of long impulse response $h[n]$. The cells $V_k(z)$ are one pole unit gain low-pass filters, while the transversal cells $A_k(z)$ are all-pass TFs.

Kautz's factorization corresponds to a discrete-time ladder network illustrated in Fig. 5.15. Thus, by definition the outputs available at each ladder section represent a set of orthogonal signals.

Remark 5.5. Note that, according to [17], the Kautz model can be seen as a generalization of the introduced above RTF physical parametric model. Thus the orthonormal generalization of Fig. 5.13, is shown in Fig. 5.16 where $P_i(z)$ are simply discrete-time resonators defined as

$$P_i(z) = \frac{1}{(1-p_i z^{-1})(1-p_i^\star z^{-1})}$$

the $A_i(z)$ are second-order all-pass sections

$$A_i(z) = \frac{(z^{-1}-p_i)(z^{-1}-p_i^\star)}{(1-p_i z^{-1})(1-p_i^\star z^{-1})}$$

and the TFs $N(z)$ defined as

$$N_i^\pm(z) = |1 \pm p_i| \sqrt{\frac{1-|p_i|^2}{2}}(z^{-1} \mp 1)$$

has been inserted in order to guarantee the orthonormal conditions.

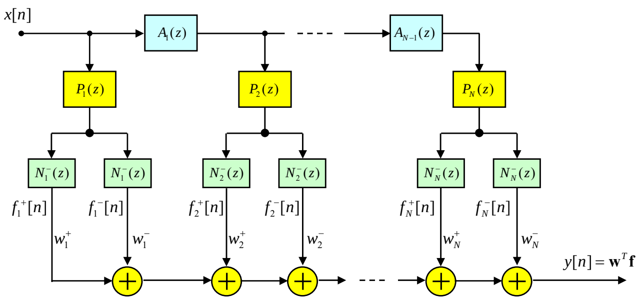


Fig. 5.16 Kautz orthogonal physical parametric RTF model. (Modified form [17]).

5.4.2 Parameters Estimation of KB-OBF Model

Although the adaptive algorithms for parameters estimation of acoustic models, will be introduced in Chapter 6, below are some methods applied for the orthogonal structures previously discussed.

5.4.2.1 Estimation of w_n parameters

The Kautz function are orthogonal, so we have that $\sum_n f_i[n] f_j^\star[n] = \delta_{i,j}$. It follows that the determination of the w_n coefficients that ensure the validity of the Eqn. (5.18), can be derived by solving the following normal equations

$$w_n = \sum_{k=1}^{M} h[k] f_n^\star[k], \qquad n = 1, ..., N.$$

Thus, Kautz OBF is a *linear in the parameters* (LIP) model w.r.t parameters w_n. In fact, let $\mathbf{F} \in \mathbb{C}^{N\times M}$ be the data matrix that contains the orthogonal signal $f_i[n]$, for $M = N$ the determination of parameters w_n is trivial. However, in parsimonious TF representation we have that $N < M$. In this case the parameter can be determined by LS criterion solving the normal equations; thus for under-determined linear system we have that the LS solution is

$$\mathbf{w} = \mathbf{F}^H(\mathbf{F}\mathbf{F}^H)\mathbf{h}.$$

Finally note that the impulse response energy, considering the Parseval theorem can be written as

$$E = \sum_n h[n]h^\star[n] = \frac{1}{2\pi}\int_0^{2\pi} H(e^{j\omega})H^\star(e^{j\omega})d\omega = \sum_{n=1}^{N} |w_n|^2 \tag{5.23}$$

5.4.2.2 Estimation of p_n Parameters and Model Reduction

The Kautz series is not unique and depend on the ordering of poles. This ordering can be chosen such that the first terms contribute most on the overall impulse response.

The availability of an ordered model allows the truncation of the Kautz series and thus a parsimonious and scalable representation of the RTF

In general, for the determination of poles starting from the RIR we proceed with iterative algorithms that minimize the given cost function (CF). For example, a CF commonly used in this problem is the *misalignments* defined as follows [15]-[17]

$$J(\mathbf{h}) = 10\log_{10}\frac{\sum_{n=0}^{L-1}|h[n]-\hat{h}[n]|^2}{\sum_{n=0}^{L-1}h^2[n]} \tag{5.24}$$

where, $h[n]$ is the actual impulse response and $\hat{h}[n]$ is the impulsive response obtained considering the desired RTF target model such as the KB-OBF model.

In practice, we proceed with an iterative method, gradually refining the target model, until we obtain an acceptable minimum of CF (5.24). Moreover, it should be noted that the so-called common poles (CAPs) (see §3.2.1.3), must also be taken into account when determining the KB-OBF models.

Remark 5.6. Note that, the parsimonious RTF representation is a central theme in modern DASP. Alternative methods to the simple LS are available in the literature that allow to insert particular *a priori* knowledge about the problem. If the system is of the so-called *sparse* type, for example when there are few *a priori* known dominant resonances or anti-resonance, it is possible to insert appropriate constraints to the optimization problem (5.24), that take into account these *a priori* knowledge.

Moreover, in addition to the classical estimation techniques, that allow an analysis according to various analytical criteria, it is also possible to consider specific perceptual audio metrics.

5.4.3 Laguerre Filters

Laguerre's filters are a special case of Kautz filters for $\beta_k = a$ where $a \in \mathbb{R}$ [18]-[20], [12]. So the conditions (5.21) and (5.22) simplify as

$$a_1^{(k)} = a,\ a_2^{(k)} = 1/a,\ C_1^{(k)} = \sqrt{1-a^2},\ C_2^{(k)} = aC_1^{(k)}.$$

The real discrete Laguerre z-transform TFs, is sequence of simple TFs defined as

$$L_i(z,a) = \frac{\sqrt{1-a^2}}{1-az^{-1}}\cdot\left(\frac{z^{-1}-a}{1-az^{-1}}\right)^{i-1}, \qquad i \geq 0,\ |a|<1$$

and the following properties apply:

- for $a=0$, $L_i(z,0) = z^{-i}$, i.g. the Laguerre polynomil is a standard delay-line;
- for $i \geq 0$, $L_{i+1}(z,0) = A(z,a)L_i(z,0)$, where $A(z,a)$ is a simple all-pass section.

i.e., these sequences can be generated in cascade, starting with a first order low-pass section $L_0(z,a)$, followed by first order simple all-pass sections $A(z,a)$.

Thus the Kautz's structure in Fig. 5.15 is simplified and resulting cascade form is shown in Fig. 5.18.

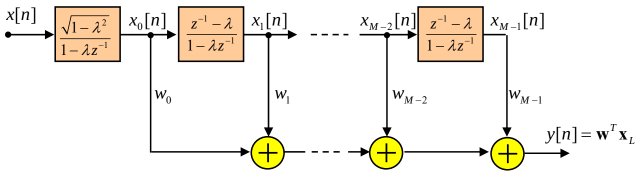


Fig. 5.17 Laguerre filter of order $M-1$. For $\lambda = 0$ it becomes a standard FIR filter.

The Laguerre filter, as shown in Fig.5.18, is obtained by a simple modification of the FIR filter replacing each unit delay element by a first order all-pass section, and applying a first order low-pass filter, with the same pole used in all-pass sections, to the filter input signal. By properly choosing the all-pass filter a parameter, Laguerre's filters are able to approximate long impulse responses with a reduced number of parameters than a FIR filter.

Denoting as $L^{(q)}\{\cdot\}$ the time-delay operator such that $x[n-1] = L^{(q)}\{x[n]\}$, and as $L_k^{(q,a)}\{\cdot\}$ operator that implement the all-pass section

$$x_k[n] = L_k^{(q,a)}\{x[n]\}$$

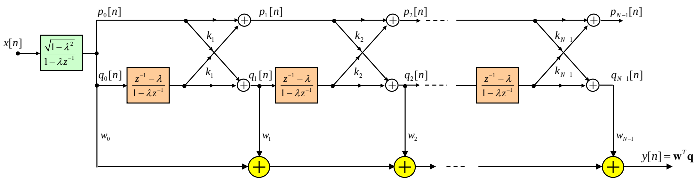


Fig. 5.18 Lattice Laguerre filter of order $N-1$. For $\lambda = 0$ it becomes a standard FIR filter.

5.4.4 Frequency Warped Signal Processing

The Discrete Fourier Transform (DFT) of a sequence, is defined as a scalar product between the sequence and a set of orthogonal basis functions which are uniformly spaced in frequency around the unitary circle of the z plane. By defining n and k, respectively, the time and frequency indexes, we can write

$$X(k) = \langle x[n], \varphi(e^{j\omega}) \rangle, \qquad \varphi(e^{j\omega}) = e^{-jk\frac{2\pi}{N}n}, \qquad k,n = 0,1,...,N-1.$$

Ths DFT is usually evaluated with algorithm such as the Fast Fourier Transform (FFT) [1]-[3].

In the audio domain, as for example in psychoacoustic analysis, non-uniform frequency scales such as the Bark scale are often used (see §2.3.4.4, Eqn. (2.4)). Hence, in the presence of non-uniform scales, as a DFT alternative, frequency analysis can be done through a, recursive or non-recursive, non-uniform filter bank such as, for example, Q-constant filter banks.

In 1971 Oppenheim, *et al.* [21], proposed a different processing paradigm based on the so called *Frequency Warped Filters*, which inherently allow efficient non-uniform frequency analysis. The procedure consists in defining a shift-variant transformation, implemented by a a henceforth called *warping operator* $\mathcal{W}(\cdot)$, based on suitable *dispersion functions*, based on a simple filter and inserted as preprocessing before the frequency analysis.

Remark 5.7. Note that, for the warping operator we can consider two distinct aspects. The first, is to apply the warping operator to a sequence $x[n]$ to produce a "warped" output sequence $x_w[n] = \mathcal{W}(x[n])$. The second aspect, is to apply warping operator to an impulsive response $h[n]$ to produce a "warped" impulse response $w[n] = \mathcal{W}(h[n])$ and then a linear *warped transfer function* (WTF).

While the production of a warped signal is of poor interest, the second aspect appears to be general as WTF could better meet some target TF specifications. In addition, the WTF has aspects of specific interest to the DASP. In fact, WTFs has proved particularly useful in DASP because the non-uniform frequency resolution can be tailored to better approximate the frequency representation of the hearing [24],[25].

5.4.4.1 Dispersion and Frequency Distortion Functions

The original sequence $x[n]$ is expanded into a set of basis functions $d_k[n]$, denoted as *dispersion basis functions*, with appropriate properties, according to the following linear transformation

$$x[n] = \langle g[n], d_k[n] \rangle = \sum_{k=-\infty}^{\infty} g[k] d_k[n]. \tag{5.25}$$

From the previous discussion, for the determination of consistent set of basis functions $d_k[n]$ and their z-transformations $D_k(z)$, we must enforce the following constraints.

1. The frequency transforms of the sequences $x[n]$ and $g[n]$, respectively defined as

$$X(e^{j\omega}) = \sum_{k-\infty}^{\infty} x[k] e^{-j\omega_k}, \quad \text{and} \quad G(e^{j\hat{\omega}}) = \sum_{k-\infty}^{\infty} g[k] e^{-j\hat{\omega}_k}$$

   are such that the frequencies ω and $\hat{\omega}$, are related by a simple change of variables $\omega = \theta(\hat{\omega})$, where $\theta(\cdot)$ is a *distortion function*.
2. A necessary condition for the distortion function $\theta(\cdot)$ is that

$$G(e^{j\theta(\hat{\omega})}) \equiv X(e^{j\hat{\omega}}), \quad \Leftrightarrow \quad \omega = \theta(\hat{\omega}) \quad \text{and} \quad \hat{\omega} = \theta^{-1}(\omega). \tag{5.26}$$

3. Remember that $z = r \cdot e^{j\omega}$, the mapping between z and $\hat{z}$ concerns the frequency, i.e. the only the angle $j\omega$ and not the radius r that remains unitary and inde-

pendent of the frequency. Therefore, the $d_k(n)$ basis functions must have all-pass characteristics.

4. The determination of the basis functions $d_k[n]$ is done by limiting the frequency mapping in the range $\omega, \hat{\omega} \in [0, 2\pi]$.

Therefore, for the above positions, requiring that $D_k(z) = Z\{d_k[n]\}$ are rational all-pass functions, we have that

$$D_k(z) = \left[\frac{z^{-1} - \lambda}{1 - \lambda z^{-1}}\right]^k \tag{5.27}$$

where the term $D_{k=1}(z)$ is denoted *dispersive element*. Consequently, the distortion function is a bilinear conformal mapping from the unit disk onto another unit disk. Moreover, by simple manipulation of the above, we have that the inverse transforma-

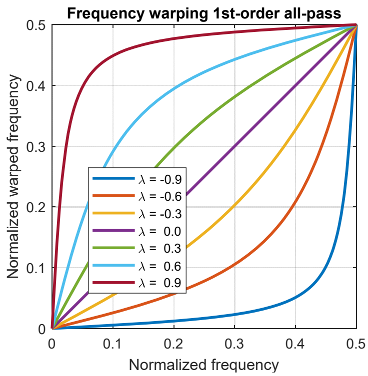


Fig. 5.19 Frequency warping of a 1-st order all-pass section $D_1(z)$ (see Eqn. (5.28)). λ is the *warping parameter*

tion is produced by replacing λ with $-\lambda$. In particular, for $k = 1$ we have

$$\begin{aligned} \omega = \theta(\hat{\omega}) &= \tan^{-1}\left[\frac{(1-\lambda^2)\sin(\hat{\omega})}{(1+\lambda^2)\cos(\hat{\omega}) - 2\lambda}\right] \\ \hat{\omega} = \theta^{-1}(\omega) &= \tan^{-1}\left[\frac{(1-\lambda^2)\sin(\omega)}{(1+\lambda^2)\cos(\omega) - 2\lambda}\right] \end{aligned} \tag{5.28}$$

where $-1 < \lambda < 1$ is denoted as *warping parameter* and whose trend is reported in Fig. 5.19.

In particular, as shown in Fig. 5.20, Eqn. (5.28) is a bilinear-transformation such that the unitary circle in the z-plane is mapped in the unitary circle in the $\hat{z}$-plane.

5.4.4.2 Frequency Warped Filters Architecture

The warped signal processing, consists in the generalization of the scheme in Fig. 5.20, where the warping operator is applied on the filter impulse response in order to synthesize a given target TF. In practice, the frequency warped filters are numerical

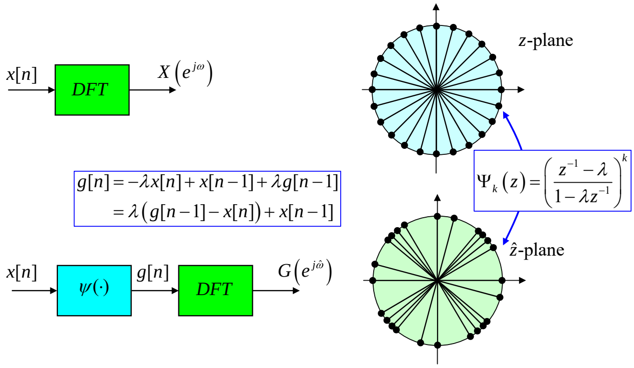


Fig. 5.20 Frequency warping of DFT transformation. The Eqn. (5.28) is a bilinear-transformation such that the unitary circle in the z-plane is mapped in the unitary circle in the $\hat{z}$-plane.

transversal or recursive filters, where the unit delays z^{-1} are replaced with dispersive elements $\hat{z}^{-1} = D_1(z)$. An example of M-length warped FIR (WFIR) filter is shown of Fig. 5.21 [24]-[26], and note that the architecture is similar to the Laguerre filter presented in §5.4.3.

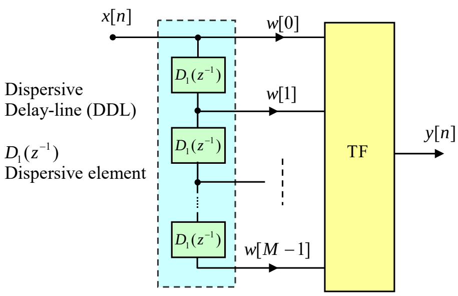


Fig. 5.21 Warped FIR (WFIR) filter. The the warped transfer function (WTF) is synthesized by replacing the delay elements z^{-1} with all-pass dispersive element $D_1(z)$. The terms $w[n]$ represent the warped impulse response.

Let $h[n] = Z^{-1}\{H(z)\}$ be the impulse response of a FIR filter defined with standard delay z^{-1}, a warped FIR (WFIR) filter is obtained by replacing $z^{-1} \rightarrow \hat{z}^{-1}$, i.e. $w[n] = \mathcal{W}(h[n]) = Z^{-1}\{H(\hat{z})\}$

Thus, we can define the following relationship VEDI (18) (19)

$$H(z) = \sum_{n=0}^{\infty} h[n]z^{-n} = \sum_{n=0}^{\infty} w[n] \left(\frac{z^{-1} - \lambda}{1 - \lambda z^{-n}}\right)^n$$

$$H(\hat{z}) = \sum_{n=0}^{\infty} w[n]\hat{z}^{-n} = \sum_{n=0}^{\infty} h[n] \left(\frac{\hat{z}^{-1} - \lambda}{1 - \lambda \hat{z}^{-n}}\right)^n$$

that define the direct and reverse mapping between TFs $H(z)$ and $H(\hat{z})$. Thus, the frequency response of the warping filter depends on the parameter λ. For $\lambda = 0$, the filter behaves like a normal FIR filter (i.e. $w[n] \equiv h[n]$), while for $\lambda \neq 0$, you have a bilinear mappings between z-domain and $\hat{z}$-domain. Therefore, the frequency-warping depends on the warping parameter λ as shown in Fig. 5.19.

Its TF can be written as

$$H_{firwrp}(z) = \sum_{m=0}^{M-1} w[m]\hat{z}^{-m} = \sum_{m=0}^{M-1} w[m]\left\{D_1(z)\right\}^{-m}$$

Note that, since each delay element is a first-order IIR all-pass filter tha overall impulse response of the warped FIR filter has infinite duration. In common practice the length is however truncated to only M values.

The TF function of an IIR(M,N) warped filter can be written as

$$H_{iirwrp}(z) = \frac{\sum_{m=0}^{M-1} b_m \left\{D_1(z)\right\}^{-m}}{1+\sum_{m=1}^{N-1} a_m \left\{D_1(z)\right\}^{-m}}$$

Since warping is a simple mapping from z-plane to warped $\hat{z}$-plane practically all conventional DASP methods can be revised in the warped-domain. The topic is very wide and for more details on implementation refer, for example, to the following literature [26]-[25].

5.5 Delay-Lines

As we know the *delay-line* (DL), sometime called *tapped-delay-line* (TDL), is the fundamental structure for the implementation of FIR and IIR number filters [1]-[5]. In audio signal processing DLs are extremely important as they are the basis for the realization of numerous audio effects such as *vibrato*, *flanger*, *chorus*, *slapback*, *echo*, and so on [4], [46], and, as mentioned above, the simulation of room acoustics [5], [47], [60]. For example, in the case of FIR digital filtering, as shown in Fig. 5.21-a), the delay line is the element on which the input signal samples are "shifted down" and then multiplied by the filter coefficients.

In audio signal processing, very often, the delay line is used as a pure delay: i.e. the signal enters at one end of the line and exits with a certain delay, due to the number of memory elements, at the opposite end. If the line consists of D elements z^{-1}, it can be represented with a single block as shown in Fig. 5.22. The DL's constitutive relation is therefore:

$$y[n] = x[n-D] \tag{5.29}$$

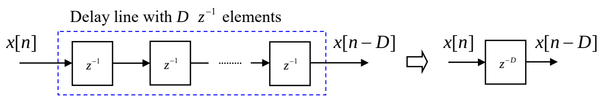
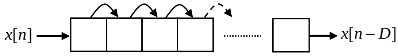


Fig. 5.22 A delay-line with D memory elements and compact representation.

5.5.1 Circular Buffer Delay-Lines

The DL's software implementation can be done with a vector in which the samples are shifted as shown in Fig. 5.23




```
double xdelay(int D, double *xdl, double x)
{
      register int i;
      for (i=D; i>=1; i--)    // reverse-order updating
            xdl[i] = xdl[i-1];
      xdl[0]=x;
      return xdl[D];
}
```

Fig. 5.23 A delay-line naive shift mechanism.

The signal sample feeds the first memory location: at each time-clock the sample flows (i.e. according to the Fig. 5.23, the sample shift-right) and frees the first vector position where the new incoming sample $x[n]$ is simultaneously entered. Therefore, to realize the shifting, the algorithm that implements the DL, performs D assignment operations $(w[i] = w[i-1])$.

Remark 5.8. Observe that, in common audio applications the delay D required to perform a certain type of processing (e.g. as in long echo effect), can be hundreds of [ms] and sometimes even some [s] and, therefore, at typical audio sampling rates the line length can reach tens of thousands of samples. In these cases, the computational cost of the shift-operations may not be negligible.

For an efficient implementation of long delay line, it is necessary to avoid shift-operations according to the technique called circular-buffer addressing [3],[5].

With reference to Fig. 5.24, it is possible to think of the buffer (that implements the DL), as if it were circularly arranged. Instead of scrolling the samples along the line, the value of the index (i.e. a *pointer* p) that point the position where the input sample is inserted, is increased.

The first signal sample is entered at position zero, the second at position one, and so on. When the buffer length is exceeded, the first position is overwritten, and so on (*wrap* operation).

The last D signal samples, are then always present in the buffer. Taking as output the value ahead the location where the input is loaded, the output is delayed by D samples. Always with reference to Fig. 5.24 when $n = 7$ the output is equal to the first element of the signal vector or $y[n] = x[n-D]$.

Fig. 5.25 shows two procedures to implement a DL of D samples.

A more general way to implement circular buffers, that realize a TF $H(z) = z^{-D}$, is the one called module addressing, described for example in [5], that uses two pointers: one to define the input p (write pointer) and the other for the output q (read pointer).

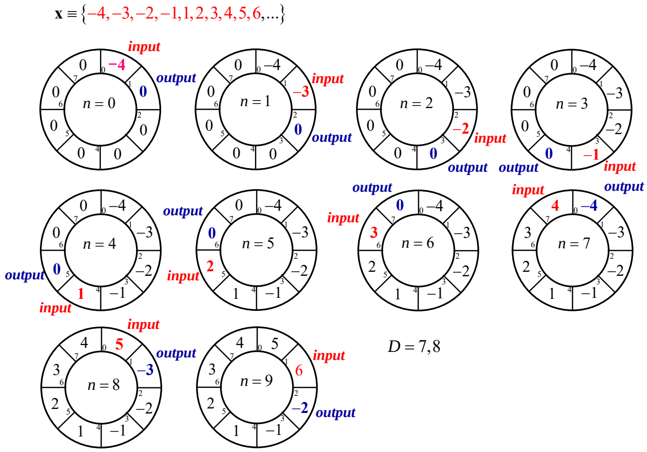


Fig. 5.24 Operating principle of a circular buffer. Diagram of the circular-buffer that realizes a DL with $D = 7$ (8 locations, from 0 to 7). For $n = 8$ the first position of the buffer is overwritten by the new incoming signal sample.

```
double DL1(double *w, int D, int p, double x)
{
  w[p] = x;   // Circular buffer index init p = 0;
  p = (p+1)%D // if (p =  = D) p = 0; else p++;
return w[p];
}

double DL2(double *w, int D, int p, double x)
{
  duoble y = w[p];    // read op.
  w[p++] = x;         // write op.
  if (p> = D) {p- = D;} // wrap pointer
return y;
}
```

Fig. 5.25 Possible implementations of a "C" functionthat implements the structure described in Fig. 5.24.

If you want to realize a DL of order D, said M the number of accessible contiguous memory locations, the input and output pointer is linked by the relationship

$$p = (q + D)\%M$$

where the $\%$ symbol indicates the modulus M operation.

At any given time, the input is written to the location addressed by p while the output is taken from location q. The two pointers are updated as

$$p = (p + 1)\%M$$
$$q = (q + 1)\%M.$$

The pointers are increased by respecting the buffer circularity.

Remark 5.9. Observe that, in certain dedicated architectures such as wavetable synthesizers, where the sampled waveforms are read sequentially from the buffer and sent to the D/A converter, the sample can be read with a variable increment pointer. In general, the sampled waveform that is available in the buffer has a certain duration. Sometimes, however, in the execution the entire waveform is not used but only a portion of it.

If, we indicate with 2^r the amount of global available memory locations, and with $M = 2^s$ (with $s < r$) the memories which are actually used, these locations are not contiguous and the update of the pointers will have to be done accordingly

$$p = (p + 2^{r-s})\%2^r$$
$$q = (q + 2^{r-s})\%2^r.$$

In practice, if the addressing is r-bit long, you don't need to explicitly calculate the module: you just need to sum it up avoiding overflow. The following also applies

$$p = (q + m2^{r-s})\%2^r.$$

5.5.2 Delay-Lines with Nested All-Pass Filters

Rather long delay lines, together with all-pass filters, are particularly used in artificial reverberation circuits [50]. In this case the nested AP structures are particularly interesting as it is possible to realize several nested filters on a single delay line as shown in Fig. 5.26.

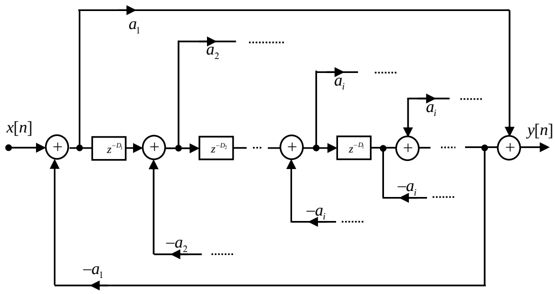


Fig. 5.26 All-pass filters nested in direct form II on a delay line.

As you can see, these circuits are simple extensions of the nested-AP filters seen in §5.3.1.

Let's consider a circuit with a generic AP TF defined as

$$A_i(z) = \frac{z^{-D_i} + a_i}{1 + a_i z^{-D_i}}.$$

This circuit can be implemented on a single DL according to the schematization in Fig. 5.27-a).

Replacing the k_i-th element of the DL with a TF $A_{i+1}(z)$ we get

$$z^{-k_1} \leftarrow z^{-k_2} A_2(z) \leftarrow z^{-k_3} A_3(z) \leftarrow \cdots$$

The resulting structure is composed of a number of all-pass nested one inside the other as shown in Fig. 5.27-b). This form is particularly efficient because, in practice,

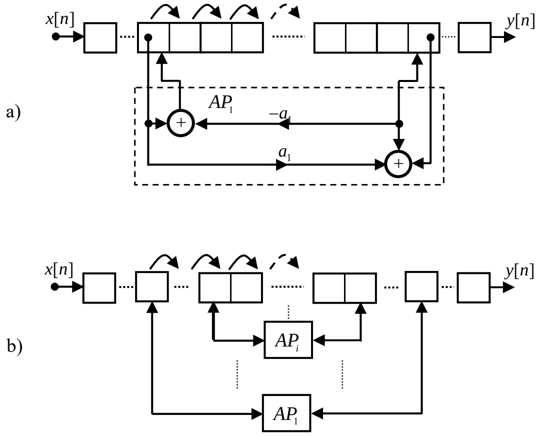


Fig. 5.27 All-pass filters. a) implemented on a single DL; b) all-pass filters nested on a single DL.

a single delay line is used whose length is equal to the sum of the delays of the individual APs $D = \sum_i D_i$

5.6 Fractional Delay-Lines

The digital delay line is characterized by a minimum delay that is defined by the sampling frequency f_s of the signal. The minimum time-delay is equal to the sampling period $T_s = 1/f_s$ which, considering the representation with the normalized sampling rate, is defined as unit delay.

In many applications it is necessary to have a delay that may not be exactly a multiple of the unit delay. In these situations, indicated as αT_s, for $\alpha \in [0, 1)$, it is necessary to define tools able to control a continuous delay or *fractional delay* (FD).

A FD may be necessary in applications such as: echo cancellation, phased-array antenna, or more generally array processing problems, pitch-synchronous speech synthesis, time-delay estimation and detection of arrivals, modem synchronization, physical modeling of musical instruments. In audio signal processing the fractional delay lines (FDLs) can be used for various types of applications such as, for example, in the conversion of the sampling rate with an irrational ratio, in various audio effects such as vibrato, microphones array processing, in the physical modeling of complex phenomena (FDL are essential in wave-field synthesis); just to name a few [28]-[41], [47].

From an implementation point of view, fractional delay lines can be seen simply as numerical filters. Therefore these can be FIR or IIR type, and designed with different philosophies and methodologies and optimization criterion like max-flat or min-max. As we will see, simple FDLs can be determined with simple intuitive considerations or by using optimization techniques usually adopted in the design of numerical filters. For example, the maximally flat FIR filter approximation is equivalent to the classical Lagrange interpolation method. However, it is not always convenient or possible to determine in closed form the impulse responses of decimation and interpolation filters:

in the case of real-time applications, the computational cost of the exact solution would be too expensive (see §4.7).

The topic of fractional delay lines is very specific and broad, and only a few aspects are explained here. For a more in-depth study please refer to the specific bibliography [27]-[43], [73]-[79].

5.6.1 Problem Formulation of Bandlimited Interpolation and Ideal Solution

The delay $D \in \mathbb{R}$ can be decomposed as a integer and a fractional part

$$D = D_i + \alpha$$

where D is the delay in term of samples, $D_i = \lfloor D \rfloor$ is the integer part of the delay and $\alpha \in [0\,1) = D - \lfloor D \rfloor$ the fractional part.

Let $x[n]$ be a sequence coming from band-limited analog signal by ideal conversion, the output of fractional delay line (FDL) is

$$y[n] = x[n - (D_i + \alpha)]$$

however, in order to avoid aliasing, it is necessary to verify the Nyquist band-limited condition, compared to the new sampling period.

There are numerous methods in the literature for the determination of FDL [28], which are generally based on the approximation a so called ideal delay-operator $L_D\{\cdot\}$ by an FIR/IIR filter or other interpolation techniques. The problem can be formulated by defining an L_D-operator such that

$$y[n] = L_D\{x[n]\} = x[n - D] \tag{5.30}$$

which in the frequency domain, in z-transform notation, is defined by the relationship

$$Y(z) = z^{-D} X(z)$$

so it turns out that the ideal transfer function (TF) is

$$H_{id}(e^{j\omega}) = e^{-j\omega D}, \qquad |\omega| \le \pi. \tag{5.31}$$

For the module and phase we have that

$$\left|H_{id}(e^{j\omega})\right| = 1, \qquad \arg\left\{H_{id}(e^{j\omega})\right\} = \Theta_{id}(\omega) = -D\omega.$$

where $\Theta_{id}(\omega)$ indicate the ideal phase response. The *group delay* is therefore

$$\tau_{g_{id}} = -\frac{\partial}{\partial \omega}\left[\Theta_{id}(\omega)\right] = D$$

while for the *phase delay* we have that

$$\tau_{p_{id}} = -\frac{\Theta_{id}(\omega)}{\omega} = D.$$

Having a group delay identical to the phase delay means, in fact, that the entire waveform, regardless of its frequency content, is delayed by a time equal to D.

The ideal solution to delay a signal by a $D \in \mathbb{R}$ quantity is a filter with a TF equal to (5.31). The impulse response is therefore

$$h_{id}[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{-j\omega D}e^{j\omega n}d\omega = \frac{\sin\left[\pi\left(n-D\right)\right]}{\pi\left(n-D\right)} = \mathrm{sinc}\left(n-D\right), \qquad \forall n. \tag{5.32}$$

When D is integer so $(D = n)$ the previous expression becomes unitary.

In general, the ideal solution is of limited usefulness for online applications because the h_{id}: 1) has an infinite length; 2) is non-causal.

5.6.2 Approximate FIR Solution

The ideal solution in Eqn. (5.32) can be approximated in many different ways. Of special interest in audio applications are digital filters that approximate the ideal interpolation in a maximally flat manner at low frequencies. In addition, in the case of the audio signal you should also be very careful with the following aspects: 1) online and real-time implementability; 2) group delay; 2) perceived quality.

5.6.2.1 Linear Interpolation: I Order FIR Filter

The simplest and most intuitive way to determine a fractional delay is to consider a linear interpolation between two successive samples of the signal as shown in Fig. 5.28.

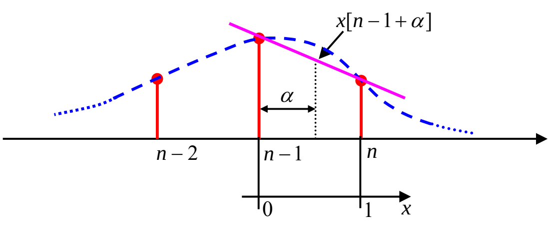


Fig. 5.28 Linear interpolation between two successive samples.

Let α be the value representing the order of interpolation between the $x[n-1]$ and $x[n]$ samples. The equation of the straight line in the ordinate passing between these points is worth

$$\frac{\alpha - 0}{1-0} = \frac{x[n-1+\alpha]-x[n-1]}{x[n]-x[n-1]}$$

the expression of the linear interpolator is therefore a FIR filter $\mathbf{h} = [1-\alpha \;\; \alpha]^T$. Therefore the interpolated sample can be compute as

$$x[n-1+\alpha] = \alpha x[n] + (1-\alpha)x[n-1] = x[n] - (x[n] + x[n-1])\alpha. \tag{5.33}$$

Based on the type of factorization of the previous expression, the linear interpolator filter scheme can be made with two multipliers or, using the equivalent polynomial Horner's scheme (or Farrow structure §5.6.7), with only one as shown in Fig. 5.29

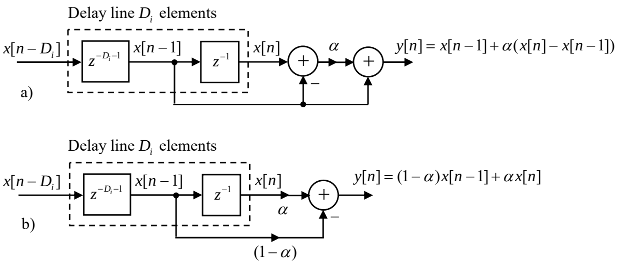


Fig. 5.29 Possible schemes for the realization of the linear interpolator. a) With one multiplier and two adders. b) With two multipliers and one adder.

To evaluate the frequency response of the linear interpolator filter, it is necessary to evaluate the DTFT of the Eqn. (5.33) for which we have

$$H(z)|_{z=e^{j\omega}} = \frac{Y(e^{j\omega})}{X(e^{j\omega})} = \alpha + (1-\alpha)e^{-j\omega} \tag{5.34}$$

Property 5.4. The linear interpolator can be derived from Taylor's expansion of the term $x[n+\alpha]$

$$x[n+\alpha] = x[n] + \alpha\dot{x}[n] + \alpha^2\ddot{x}[n] + \alpha^3\dddot{x}[n] + \cdots \tag{5.35}$$

considering the I order approximation and posing $\dot{x}[n] = (x[n+1] - x[n])/1$ we have that

$$x[n+\alpha] = x[n] + \alpha(x[n+1] - x[n])$$

which coincides with the non-causal version of Eqn.(5.33). It should also be noted that this approach can be used to define higher-order interpolation filters.

Fig. 5.30 shows the frequency and phase delay response of the expression (5.34) for some values of the α fractional delay. The amplitude response is almost flat for small α values. The linear interpolator "sounds good" when the signal is oversampled so

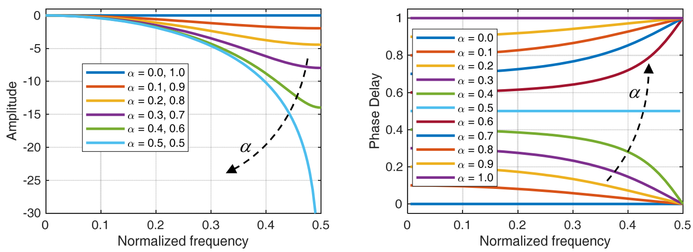


Fig. 5.30 Frequency response $|H(e^{j\omega})|$ and phase delay $-\arg\{H(e^{j\omega})\}/\omega$, of the first order linear interpolator $\mathbf{h} = [1-\alpha \;\; \alpha]$, for fractional delay values α from 0 to 1 step 0.1.

that the signal spectrum is concentrated at low frequency while for values higher than FD the interpolator behaves like a low-pass filter. In fact, although the linear interpolation technique has a low computational cost, it has some drawbacks. Below are some of them.

- *Linearity distortion* - The linear interpolator is a low-pass filter.
- *Amplitude and phase modulation* - The characteristic of the filter is time variance and introduces an overall variation in signal level and phase.
- *Aliasing* - Interpolation, in general, can be considered as a non optimal sampling rate conversion process.

5.6.2.2 Truncation and Causalization of the Ideal Impulse Response

A simple approximation of an ideal interpolator, is its causalized and truncated version. As shown in Fig. 5.31, the fractional output $x[n-D]$ is computed as a linear combination of its previous and subsequent samples. Considering an M-length FDL, the delayed sample is inside the M-length signal window starting from a given reference index M_0 appropriately chosen, i.e. $-M_0 < D < M - M_0 - 1$. For example, for $M_0 = M/2 - 1$, the output can be calculated as

$$x[n+\alpha] = \sum_{k=-M_0}^{M-M_0-1} h[k-M_0]x[n-k] = \sum_{k=0}^{M-1} h[k]x[n-M_0+k].$$

So, for a M-length filter we have

$$h[k] = \begin{cases} \mathrm{sinc}(k-D), & k \in [-M_0, M+M_0-1] \\ 0, & \text{otherwise} \end{cases} \tag{5.36}$$

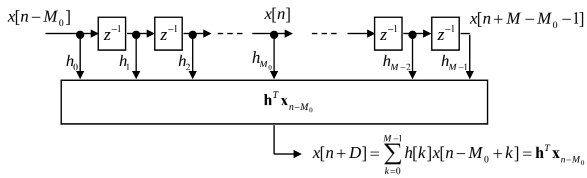


Fig. 5.31 Fractional delay FIR filter of length M. To have a causal filter, you will need to delay the output of $M-1-M_0$ samples.

Note that the smallest error for a given filter length is obtained when the overall delay D is placed around its group delay. Thus, for a linear phase FIR filter the reference index M_0 can be chosen around the group delay of the filter. For example, given a M-length FIR filter an possible choice of the reference index is

$$M_0 = \begin{cases} \dfrac{M}{2} - 1, & \text{for } M \text{ even,} \quad \text{and } \alpha \in [0, 1) \\ \dfrac{M-1}{2}, & \text{for } M \text{ odd,} \quad \text{and } \alpha \in [-0.5, 0.5). \end{cases} \tag{5.37}$$

As an example, Fig. 5.32 shows the impulse responses of a two FDL with $M = 16$ and $M = 17$. According to Eqn. (5.37), for the even-length filter we have $M_0 = 7$, with this choice, to have symmetry of the phase delay response (see Fig. 5.33), the fractional part is chosen in the interval $\alpha = [0, 1)$; for the odd-length filter we have $M_0 = 8$ and with this choice, for symmetric phase delay response, the fractional part is chosen in the interval $\alpha = [-0.5, 0.5)$.

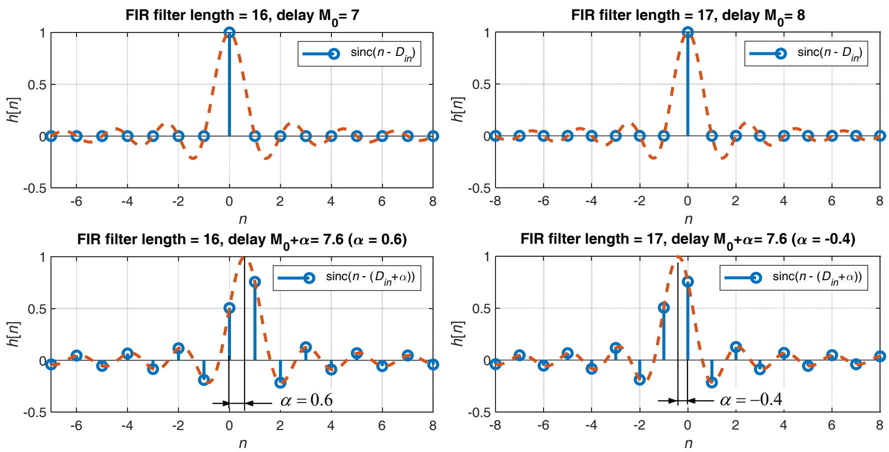


Fig. 5.32 Fractional delay FIR filters of length $M = 16, 17$. In the upper a integer delay equal to $M_0 = 7, 8$ is considered, so the filters impulse responses is a simple delayed unit. The related $\mathrm{sinc}(\cdot)$ function (dashed line) is null in correspondence of all the samples except for $n = 0$ where it is equal to $h[0] = 1$. In the lower part is reported the impulse responses of the fractional delay for $\alpha = 0.6, -0.4$ and their related $\mathrm{sinc}(\cdot)$ functions.

Remark 5.10. Observe that the frequency responses of even- and odd-length FIR FDL filters are different. As you can see from Fig.5.33, the even-length FIR FDL filter ($M = 16$) has a high ripple in the amplitude response (magnitude), while the phase delays are quite smooth and with minimum low frequency error. The odd-length FIR FDL filter ($M = 17$) has complementary characteristics, a low ripple in the magnitude, high ripple in the phase delay response and high error in the low frequencies.

For a better overview, in Fig. 5.34 are reported the 3D plot of the magnitude squared error $|H_{id}(e^{j\omega}) - H(e^{j\omega})|^2$ and of the phase-delay squared error $|\alpha - \tau_{H(e^{j\omega})}|^2$, evaluated for several even- and odd-length filters.

Finally, Fig. 5.35 shows the magnitude and delay of a FDL for $M = 2$. The reference index M_0 is $M_0 = M/2 - 1$ and, with this reference the impulse $\mathrm{sinc}(n - D)$ response is simply evaluated for $n = 0$ and $n = 1$ as

$$h[0] = \mathrm{sinc}(\alpha), \qquad \text{and} \qquad h[1] = \mathrm{sinc}(1 - \alpha).$$

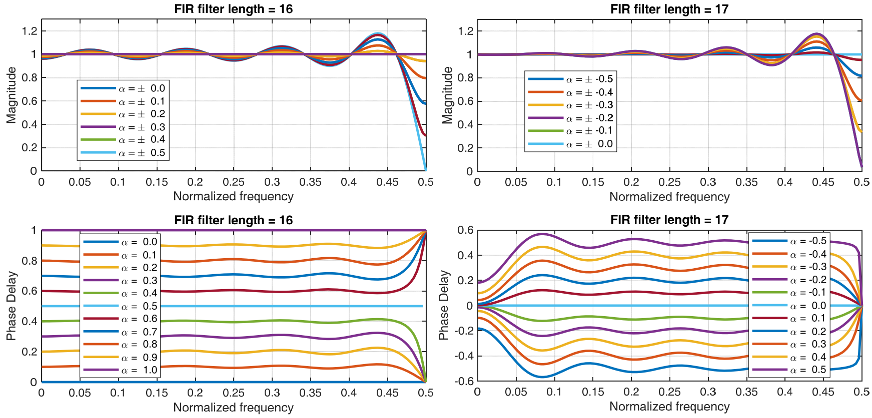


Fig. 5.33 Magnitude and phase-delay α response, such that $D = M_0 + \alpha$, of even-length (left) and odd-length (right) truncated sinc$(\cdot)$ FDL. Note that, for even length filter the phase delay is symmetric respect to the delay 0.5, while for odd length respect 0.

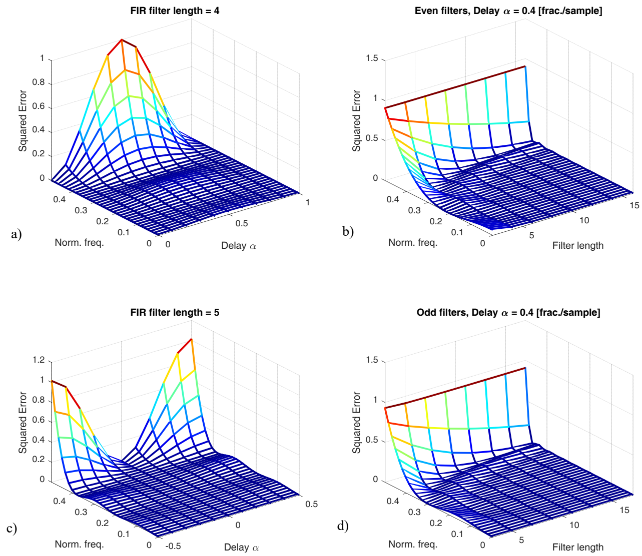


Fig. 5.34 Mean squared magnitude error and mean squared phase-delay error for even- and odd-length FIR filters. a) For fixed $M = 4$, as a function of frequency and of the delay $M_0 + [0,1)$. b) For even-length filters $M \in [2,16]$ with fixed target delay with $\alpha = 0.4$, as a function of $\omega/2\pi$. c) For fixed $M = 5$. c) For fixed $M = 5$, as a function of frequency and of the delay $M_0 + [-0.5, 0.5)$. d) For odd-length filters $M \in [3,17]$ with fixed target delay with $\alpha = 0.4$, as a function of $\omega/2\pi$.
.

Moreover, for $M = 2$ the interpolator filter can be implemented as shown in Fig. 5.36.

5.6.2.3 Fractional Delay FIR Filter by Least Squares Approximation

Let $E(e^{j\omega}) = H_{id}(e^{j\omega}) - H(e^{j\omega})$, be the difference between the desired response and the filter response, denoted as *error* possimo procedere come esposto nel §4.2.2.2, by

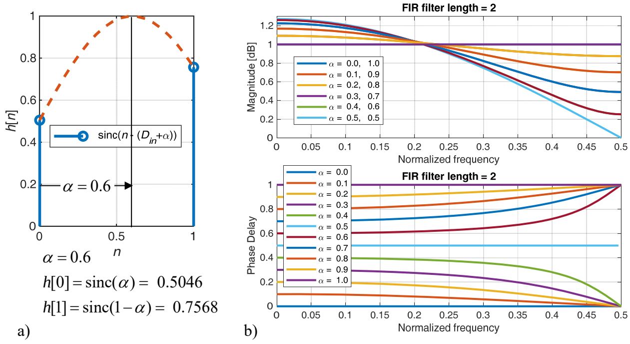


Fig. 5.35 Fractional delay FIR filter of length $M = 2$ with $\mathbf{h} = [\text{sinc}(\alpha)\;\; 1 - \text{sinc}(\alpha)]^T$. a) The impulse response filter and sinc function (dashed line) for $\alpha = 0.6$. b) Magnitude and phase delay for eleven values of delay α.

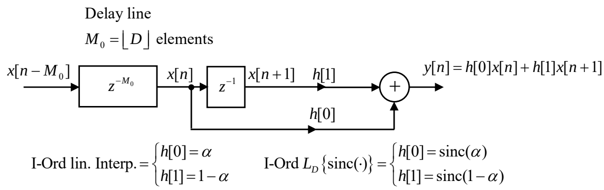


Fig. 5.36 Possible implementation schema of a fractional delay FIR filter of length $M = 2$.

placing $H_{id}(e^{j\omega}) = e^{-j\omega D}$ and solving the normal equations as Eqn. (4.8). Moreover, an improved WLS method for variable FD FIR filters design can be found in [37].

However, for the case of real implulse response it is not necessary the use of the LS approximation by virtue of the following property.

Property 5.5. The impulse response of an FD FIR filter, evaluated by L_2-norm error minimization, i.e. (4.8), coincides with the ideal sinc(·) truncated response. That is

$$\mathbf{h}_{LS} = (\mathbf{F}^H\mathbf{F})^{-1}\mathbf{F}^H\mathbf{d} = \text{sinc}(n - D)|_{n=0,\ldots,M-1}.$$

Proof. For the Parseval relation the L_2-norm error (4.5) can be expressed in time-domain as

$$J(\mathbf{h}) = \sum_{n=-\infty}^{\infty} |h_{id}[n] - h[n]|^2 = \sum_{n=-\infty}^{\infty} \left(h_{id}^2[n] + h^2[n] - 2h_{id}[n]h[n]\right) \tag{5.38}$$

where $h_{id}[n]$ =sinc$(n - D)$. Since, by definition $\sum_{n=-\infty}^{\infty} |h_{id}[n]|^2 = 1$, for a M-length FIR-filter Eqn. (5.38) can be written as

$$J(\mathbf{h}) = 1 + \sum_{n=0}^{M-1} \left(h^2[n] - 2h[n]\text{sinc}(n - D)\right).$$

The optimal solution is for $\frac{\partial J(\mathbf{h})}{\partial \mathbf{h}} \to 0$. Operating in scalar form, switching the derivation and sum operations, we can write

$$h_{opt}[n] \therefore \sum_{n=0}^{M-1} \frac{\partial \left(h^2[n] - 2h[n]\mathrm{sinc}(n-D)\right)}{\partial h[n]} = \sum_{n=0}^{M-1} \left(2h[n] - 2\mathrm{sinc}(n-D)\right) \to 0$$

that has a minimum point for

$$h[n] = \mathrm{sinc}(n-D), \qquad n = 0, 1, ..., M-1.$$

□

As in the standard FIR-filter design, to reduce the ripple (known as the Gibbs phenomenon) of the interpolator filter, the ideal impulse response can be multiplied with an appropriately shaped window $w[k-D]$.

$$h[k] = \begin{cases} w[k-D]\mathrm{sinc}(k-D), & k \in [-M_0, M+M_0-1] \\ 0, & \text{otherwise} \end{cases} \tag{5.39}$$

where $w[\cdot]$ is the window-function which can have various shapes, triangular (or Hanning window), raised cosine (Hamming window), and so on [1]-[3].

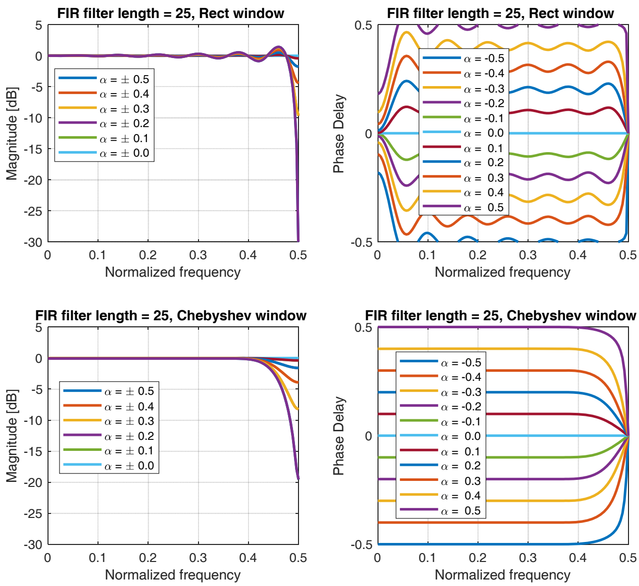


Fig. 5.37 Magnitude responses (left) and phase delay curves (right) of a 25-length FD FIR filter. In the upper the simple truncated ideal impulse response. In the lower the ideal impulse response multiplied by a Chebyshev window. Note that, the ripple decrease is associated with a widening of the filter transition band.

In Fig. 5.37 are reported the magnitude responses and the phase delays of a two FIR filter multiplied with different window-function, when the desired fractional phase delay varies from $\alpha = \pm 0.5$ with a step of 0.1 [frac/sample]. From the figure we can see that for a rectangular window (i.e. simple truncated sinc function), the amplitude response has a certain ripple. Using a standard Chebyshev window, the ripple is contained at very low values. However, the price paid for the ripple reduction is the widening of the filter transition band. Therefore, near the maximum (normalized) frequency 0.5, there is a degradation of the filter response [1]-[3].

Remark 5.11. Observe that, as shown in the next paragraphs, very often in non recursive filter interpolation, a FIR filter is used in the form of a linear on non-linear interpolator (polynomial or spline type) or Lagrange quadratic law [28], [41] and [48].

5.6.3 Approximate All-Pass Solution

Widely used for its low computational complexity, another technique to realize the FDL, is to use all-pass filters. Since the amplitude response is constant, we will not have any frequency-dependent attenuation (property not assured with FIR filters). The TF of an all-pass filter is of the type $H(z) = \frac{z^{-N}A(z^{-1})}{A(z)}$ (see Eqn. (5.11)), thus th TF a first-order all-pass filter can be written as

$$H(z) = \frac{a + z^{-1}}{1 + az^{-1}}$$

where $|a| < 1$ for stability.

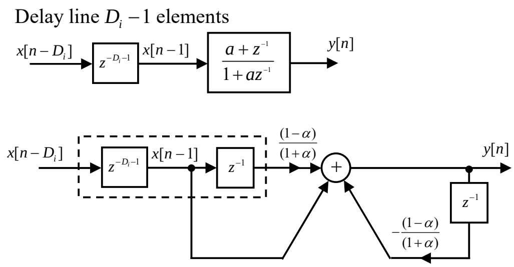


Fig. 5.38 Fractional delay line with all-pass approximation.

For filters of this type it is easy to verify that at low frequencies the following approximation applies

$$\arg\left[H(e^{j\omega})\right] \approx -\frac{\sin(\omega)}{a+\cos(\omega)} + \frac{a\sin(\omega)}{1+a\cos(\omega)} \approx -\omega\frac{1-a}{1+a}$$

for group $\tau_g(\omega)$ and phase $\tau_p(\omega)$ delays for $\omega \to 0$, results

$$\tau_g(\omega) \approx \tau_p(\omega) \approx \frac{1-a}{1+a}.$$

Given by definition $\alpha = \tau_g(0)$, the filter coefficient can be determined on the basis of the desired delay at low frequencies such as

$$a = \left.\frac{1-\tau_g(\omega)}{1+\tau_g(\omega)}\right|_{\omega\to 0} = \frac{1-\alpha}{1+\alpha}. \tag{5.40}$$

With this approximation the group delay is constant, with good approximation, up to $f_s/5$ [5].

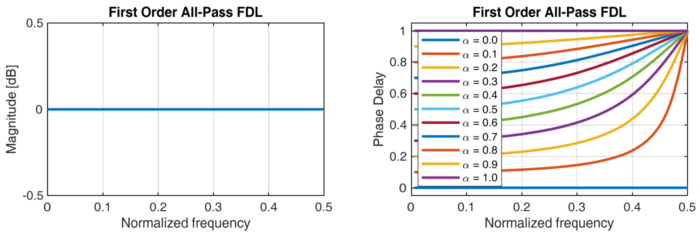


Fig. 5.39 First order all-pass interpolator. Magnitude response $|H(e^{j\omega})|$ and phase delay $-\arg\{H(e^{j\omega})\}/\omega$, of the first order all-pass interpolator $\mathbf{a} = [\alpha\ 1]$, $\mathbf{b} = [1\ \alpha]$ for fractional delay values α from 0 to 1 step 0.1.

In this case the fractional delay line scheme is the one shown in Fig. 5.38 where the whole part of the delay $(D_i - 1)$ is performed with a normal DL and the fractional part is relegated to the all-pass filter.

The input-output relation results to be

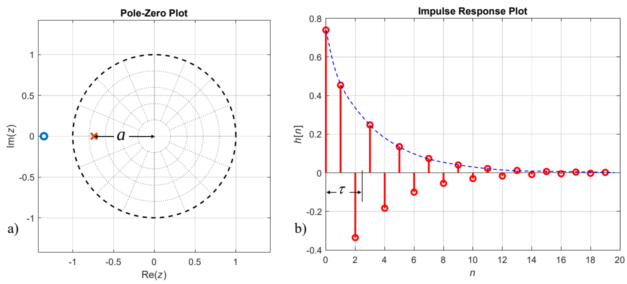
$$y[n] = ax[n] + x[n-1] - ay[n-1]$$

The frequency response of an all-pass filteis flat by definition. As with the linear interpolator, the all-pass has some acoustic drawbacks. Typically, However, for low α values, the all-pass structure sound better than the linear interpolator.

5.6.3.1 Reduced Transient Response

The use of recursive structures (such as the all-pass) must be done very carefully as they may give rise to annoying transients due to the unlimited filter impulse response. In fact, due to recurrent nature, unlike in the FIR case, in the allpass interpolation, interpolated values cannot be requested arbitrarily at any time in isolation or "random access mode" [60].

Fig. 5.40 Impulse response of an all-pass filter for $\alpha = 0.15$. The TF is $H(z) = \frac{a+z^{-1}}{1+az^{-1}}$ with $a = 0.7391$. a) Pole-zeros diagram. b) Impluse resposnse.



Moreover note that, as Fig. 5.40 shown, for $\alpha \to 0$ the impulse response of the all-pass filter is quite long. This results in nonlinear-phase distortion at frequencies close to half the sampling rate. This phenomenon produces a disturbance that is particularly audible when using low sample rates (e.g. f_s <44.1kHz) often used in low-cost devices. In audio applications, therefore, we would like to keep the impulse response duration short enough to sound instantaneous. Since the decay time constant of the impulse response of a pole of radius a is about $\tau \approx \frac{1}{1-a}$ [69], and since a 60-dB decay occurs in about 7τ, we can limit the pole of the allpass filter to achieve any prescribed specification on maximum impulse-response duration [60].

5.6.3.2 Thiran All-Pass Fractional Delay Filter

The Thiran all-pass filter has a TF (5.11) where the denominator coefficients a_k, are defined by the following formula [30].

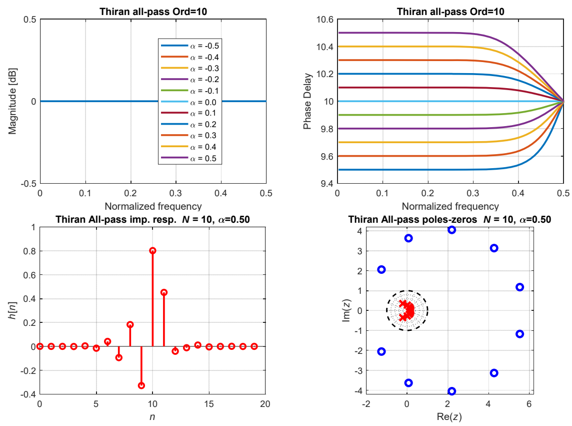


Fig. 5.41 Thiran all-pas filter order N=10. (Upper) Magnitude and phase-delay. (Lower) Impulse response and poles-zeros plot for $\alpha = 0.4$. Note that the FDL's delay performance are poor for $f > 0.3(f_s/2)$.

$$a_k = (-1)^k \frac{N!}{k!(N-k)!} \prod_{n=0}^{N} \frac{N-n-D}{N-k-n-D}, \qquad k = 0, 1, 2, \ldots, N. \tag{5.41}$$

Thiran allpass filters have unity gain at all frequencies and they produce a maximally flat group delay response at $\omega \to 0$. As illustrated in Fig. 5.41, performance is optimal at DC but poor close to the Nyquist frequency. Also note that the impulsive response is relatively short so it can be used in time-varying situations.

5.6.4 Polynomial Interpolation

The theoretical development of the methods described below is performed considering a continuous signal $x(t)$, defined in the interval $t \in [t_0, t_N]$, and known only in a finite set of $N+1$ samples $x(t_k)$, k = 0, 1, ..., N. The problem is the reconstruction of the whole waveform $x(t)$ in $t \in [t_0, t_N]$ starting only from the knowledge of $N+1$ samples. The estimation of the signal is done by means of an interpolating function $\Phi(t, \Delta t)$ which, in general, depends on the time t and the choice of the sampling interval t.

If the $N+1$ known samples of the signal are not uniformly spaced, the interpolating function should be evaluated for each t_k sample. Given $\hat{x}(t)$ the signal estimate $x(t)$, we have

$$\hat{x}(t) = \sum_{k=0}^{N} \Phi_k(t, \Delta t_k) x(t_k).$$

If a specific interpolating function (e.g. polynomial, B-spline, etc.) is considered, the previous expression can be rewritten in the convolution form as

$$\hat{x}(t) = \sum_{k=0}^{N} h(t, t_k) x(t_k) \tag{5.42}$$

which has a form that is equivalent to a time-varying FIR filter.

In case the samples are uniformly spaced, the function $\Phi_k(t, \Delta t)$ is the same for all the sampling intervals, and we can simply write

$$\hat{x}(t) = \sum_{k=0}^{N} \Phi_k(t) x(t_k).$$

In this case, regardless of the chosen interpolating function, it is possible to determine a time-invarinat FIR filter that implements the FDL, also called *fractional delay filter*, of the type

$$\hat{x}(t) = \sum_{k=0}^{N} h(t_k) x(t - t_k). \tag{5.43}$$

Now let's consider the case in which the signal is a sequence derived from the $x(t)$ signal by means of a sampling done in instants $t_n = nT$ (with a constant T sampling period and, for simplicity, $T = 1$) for which

$$x[n] = x(t)|_{t=nT}, \qquad n \in [0, N] \tag{5.44}$$

Said $x[n+\alpha]$, with $0 \le \alpha < 1$, the signal value at any point between two successive samples n and $(n+1)$, the Eqn. (5.43) can be written as

$$x[n+\alpha] = \sum_{k=0}^{N} h[k] x[n-k] \tag{5.45}$$

i.e. a simple FIR filter.

5.6.4.1 Interpolation with Polynomial FIR Filter

In the first method, proposed in [33], we analyze the interpolating function $\Phi(t)$ is realized with a polynomial of order N. The methodology derives considering a polynomial $p_N(x)$ of order N able to represent exactly an $x(t)$ function in a set of uniformly spaced $N+1$ samples.

Let us consider a polynomial,

$$x(t) = a_0 + a_1 t + a_2 t^2 + \cdots + a_N t^N = \sum_{k=0}^{N} a_k t^k \tag{5.46}$$

which passes in $(N+1)$ points (t_n, x_n) with n = 0, 1, ..., N.

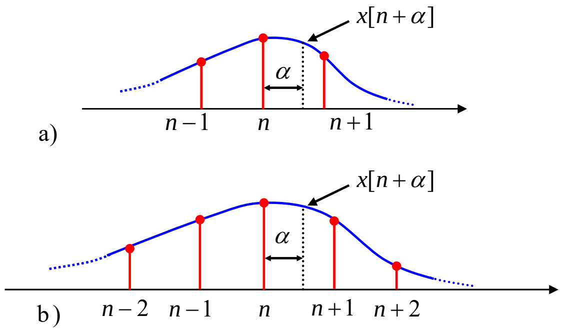


Fig. 5.42 Example of interpolating polynomial filter. a) Order $N = 2$. a) Order $N = 4$.

The above expression can be seen as a set of linear equations of the type $\mathbf{Ta} = \mathbf{x}$ where $\mathbf{T}$ is a Vandermonde matrix. Explicitly we have that

$$\begin{bmatrix} 1 & t_0 & t_0^2 & \cdots & t_0^2 \\ 1 & t_1 & t_1^2 & \cdots & \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & t_N & t_N^2 & \cdots & t_N^N \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_N \end{bmatrix} \tag{5.47}$$

where polynomial coefficients a_i are computed as $\mathbf{a} = \mathbf{T}^{-1}\mathbf{x}$.

If we consider a sequence $x[n]$, and II order interpolator $x(t) = a_0 + a_1 t + a_2 t^2$, for $t = n$, as shown in Fig. 5.42-a), we have that the points where the polynomial passes are $(n-1, x[n-1]), (n, x[n])$ and $(n+1, x[n+1])$. Therefore, the equations system to determine the a_k coefficients is

$$\begin{aligned} x[n-1] &= a_0 + a_1(n-1) + a_2(n-1)^2 \\ x[n] &= a_0 + a_1 n + a_2 n^2 \\ x[n+1] &= a_0 + a_1(n+1) + a_2(n+1)^2 \end{aligned} \tag{5.48}$$

that in matrix form can be written as

$$\begin{bmatrix} 1 & (n-1) & (n-1)^2 \\ 1 & n & n^2 \\ 1 & (n+1) & (n+1)^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} x[n-1] \\ x[n] \\ x[n+1] \end{bmatrix} \tag{5.49}$$

The system (5.49) can be symbolically solved, so we have

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \frac{n(n+1)}{2} & 1-n^2 & \frac{n(n-1)}{2} \\ -\frac{(2n+1)}{2} & 2n & -\frac{(2n-1)}{2} \\ \frac{1}{2} & -1 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} x[n-1] \\ x[n] \\ x[n+1] \end{bmatrix}.$$

Once the coefficients of the polynomial a_k have been determined, the sample interpolated at the instant $(n+\alpha)$, with $\alpha < 1$, is calculated by evaluating the Eqn. (5.46) for $t = (n+\alpha)$. Therefore it is equal to

$$x[n+\alpha] = \sum_{k=0}^{N} a_i t^k \bigg|_{t=(n+\alpha)} = a_0 + a_1(n+\alpha) + a_2(n+\alpha)^2. \tag{5.50}$$

Remark 5.12. Note that, it can be useful to express the sample with fractional delay as a function of the neighboring samples $x[n+\alpha] = F(x[n-1], x[n])$. This can be easily obtained by combining (5.50) with (5.48). It follows a linear relationship and that the sample $x[n+\alpha]$ can be seen as the output of a second-order FIR filter, i.e. with three coefficients, of the form in Eqn. (5.45)

$$x[n+\alpha] = c_{-1}x[n-1] + c_0 x[n] + c_1 x[n+1]$$

where

$$c_{-1} = \frac{1}{2}\alpha(\alpha-1), \qquad c_0 = -(\alpha+1)(\alpha-1), \qquad c_1 = \frac{1}{2}\alpha(\alpha+1) \tag{5.51}$$

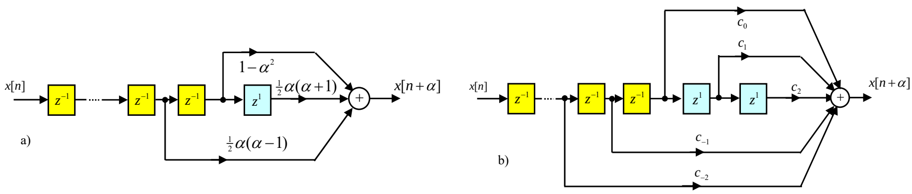


Fig. 5.43 Implementation scheme of interpolating polynomial filter. a) Order $N = 2$. a) Order $N = 4$.

Remark 5.13. Observe that, the FIR filter coefficients c_{-1}, c_0 and c_1 are time-invariant and depend only on the delay α. If the sampling interval is not constant this would no longer be true and the filter coefficients should be recalculated for each sample.

In general, the determination of the $x[n+\alpha]$ can be expressed as

$$x[n+\alpha] = \sum_{k=-N/2}^{N/2} c_k(\alpha) x[n-k]. \tag{5.52}$$

For example, for $N=4$ we have that

$$
\begin{aligned}
c_{-2} &= \frac{1}{24}(\alpha+1)\alpha(\alpha-1)(\alpha-2)\\
c_{-1} &= -\frac{1}{6}(\alpha+2)\alpha(\alpha-1)(\alpha-2)\\
c_{0} &= \frac{1}{4}(\alpha+2)(\alpha+1)(\alpha-1)(\alpha-2)\\
c_{1} &= -\frac{1}{6}(\alpha+2)\alpha(\alpha+1)(\alpha-2)\\
c_{2} &= \frac{1}{24}(\alpha+2)\alpha(\alpha+1)(\alpha-1)
\end{aligned}
\tag{5.53}
$$

The filter implementation structure is shown in Fig. 5.43. Similar expressions can be found for polynomial orders higher than $N=2M$. In general we can write

$$
c_k = \frac{1}{(-1)^{M-k}(M+k)!(M-k)!}\prod_{\substack{n=-M\\ n\neq k}}^{M}(\alpha-n), \qquad k=-M,\ \dots,\ M. \tag{5.54}
$$

5.6.4.2 Lagrange Polynomial Interpolation Filter

The problem of determining the sample value at a fractional delay $x[n+\alpha]$, with a polynomial approximation, where the polynomial coefficients are determined by Eqn. (5.54), can be reintroduced from different assumptions.

The Lagrange polynomial interpolation method, originates from considering a N-order polynomial that can represent exactly one function $x(t)$ in a set of $N+1$ samples t_i. In this case, however, it is *a priori* required that the polynomial assumes zero value except at a specific point. The polynomial characterized by this behavior is called *Lagrange's polynomial* (LP) .

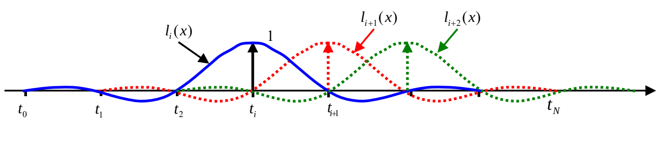


Fig. 5.44 Lagrangre polynomials (LPs). The assumption is that the polynomial takes zero values at the sampling point except in the sample of interest.

The Lagrange polynomial of order N, related to the i-th sample, indicated as $l_i^N(x)$, is defined as

$$
l_i^N(x) = \delta_{ik} = \begin{cases} 1, & i=k\\ 0, & \text{otherwise.}\end{cases}
$$

The LP, as shown in Fig. 5.44, is characterized by N zeros in positions t_0, t_1, ... while at the i-th value $x(t_i)=1$ applies. In numerical transmission jargon, this condition guarantees the absence of *intersymbolic distortion*. The basic idea in the use of the Lagrange polynomial is therefore to force to zero the impulsive response of the interpolator filter in correspondence of the neighbouring samples so as to reduce the effects of intersymbolic distortion.

From the LP's zeros it is easy to verify that this can be written as

$$l_i^N(x) = a_i(t-t_0)\cdots(t-t_{i-1})(t-t_{i+1})\cdots(t-t_N) \tag{5.55}$$

from the previous expression for $l_i^N(x) = 1$, the a_i coefficients are calculated as

$$a_i = \frac{1}{(t-t_0)\cdots(t-t_{i-1})(t-t_{i+1})\cdots(t-t_N)}.$$

Considering all the signal sampling points, the shape of the overall interpolator is the sum of the LPs related to all the $N+1$ points each of which is multiplied by the value of the function in that point

$$p_N(x) = \sum_{i=0}^{N} l_i^N(x)x(t_i) = l_0^N(x)x(t_0) + l_1^N(x)x(t_1) + \cdots + l_N^N(x)x(t_N). \tag{5.56}$$

By setting $a = \prod_{j=0}^{N}(t-t_j)$ the (5.55) can be rewritten as

$$l_i^N(x) = a_i\frac{a}{t-t_i} = \frac{1}{\prod_{j=0,j\neq i}^{N}(t_i-t_j)}\frac{\prod_{j=0}^{N}(t-t_j)}{t-t_i} = \prod_{j=0,j\neq i}^{N}\frac{t-t_i}{t_i-t_j} \tag{5.57}$$

additionally, in the case of uniformly spaced samples, by placing $t_i = t_0 + i\Delta t$ and $t_j = t_0 + j\Delta t$; $(i, j$ integer) and defining a new variable $\alpha < 1$ such that $t = t_0 + \alpha\Delta t$ we have that

$$\frac{t-t_i}{t_i-t_j} = \frac{t_0+\alpha\Delta t - t_0 - i\Delta t}{t_0 + i\Delta t - t_0 - j\Delta t} = \frac{\alpha - i}{i-j}$$

and the expression (5.57) can be rewritten as

$$l_i^N(x) = \prod_{j=0,j\neq i}^{N}\frac{\alpha-j}{i-j} \tag{5.58}$$

here denoted as Lagrange interpolation filter (LIF), where only sampling points appear. Note that the above expression is identical to (5.54) obtained by simple polynomial regression model.

Let's consider as usual numerical sequence where the available samples are uniformly spaced $n = t_n$, the interpolator output (5.56) can be calculated as a simple convolution

$$x[n+\alpha] = \sum_{k=0}^{N} h[n]x[n-k], \quad \text{where} \quad h[k] = l_k^N(\alpha). \tag{5.59}$$

It should be noted that this expression is similar to that seen above (5.52). In fact, in the case of uniform sampling the (5.59) would take the form of an FIR filter with constant coefficients that depend only on the value of the fractional delay.

Finally, as $D = D_i + \alpha$ where $D_i \in \mathbb{Z}^+$ is fixed, in Eqn. (5.58), in order to consider only the adjustable fractional delay $\alpha \in \mathbb{R}$, we have two null-phase version alternative for the interval $[0, N]$. A common choice for even and odd order is

$$\begin{array}{llll} N_{\text{sup}} = \dfrac{N}{2}, & N_{\text{inf}} = -N_{\text{sup}}, & \alpha \in \left[-\dfrac{1}{2}, \dfrac{1}{2}\right) & N\text{-even} \\ N_{\text{sup}} = \dfrac{N+1}{2}, & N_{\text{inf}} = -\dfrac{N-1}{2}, & \alpha \in [0, 1) & N\text{-odd} \end{array} \tag{5.60}$$

In fact, as previously explained in Fig. 5.32 and Fig. 5.33, for optimal performance, the fractional delay should be positioned approximately halfway along the filter length. So, in case of odd order we can also choose $N_{\text{sup}} = \frac{N-1}{2}$ and $N_{\text{inf}} = -\frac{N+1}{2}$.

```
function h = coeff_L(N, delay)
% -----------------------------------------------------------------
%  Returns (N + 1) taps of Lagrange interpolator FIR
%  filter which implements a fractional delay-line.
% -----------------------------------------------------------------
    if mod(N,2) == 0
        Nsup = N/2; Ninf = -Nsup;
    else
        Nsup = (N + 1)/2;  Ninf =  -(N - 1)/2;
    end
    n = 0;
    for i = Ninf : Nsup
        n = n + 1;
        for j = Ninf : Nsup
            if (i~=j) h(n) = h(n) * (delay - j)/(i - j); end
        end
    end
end
```

Fig. 5.45 Example of MATLAB code for Lagrange interpolation filter (LIF) coefficients determination in Eqn. (5.61).

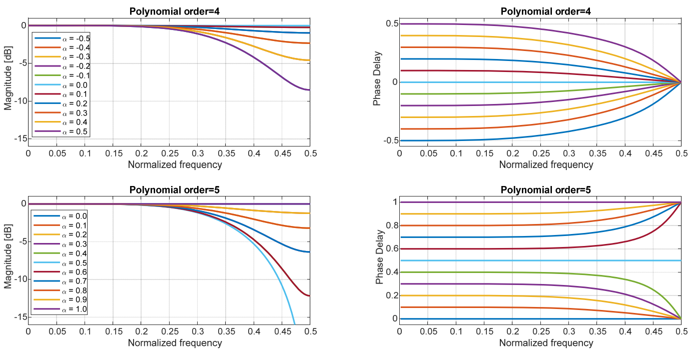


Fig. 5.46 Example of LIF magnitude and phase delay responses of even ($N = 4$), and odd orders ($N = 5$) Lagrange interpolation filters Eqn. (5.61).

Thus, Eqn. (5.58) can be rewritten as

$$h[n] = l_n^N(x) = \prod_{j=N_{\text{inf}}, j \neq n}^{N_{\text{sup}}} \frac{\alpha - j}{n - j} \tag{5.61}$$

So, considering the implementation using Eqn. (5.58) the argument is the ovarall delay $D_i + \alpha$, while with the null-phase expressions (5.61)), the argument is the only fractional part of the delay α.

In Fig. (5.45) an example of MATLAB code that implements the Eqn. (5.61); while Fig. 5.46 shown the magnitude and phase delay responses of even and odd orders Lagrange interpolation filters designed with Eqn. (5.61).

For example for N =2, using Eqn. (5.61) we will have

$$h[-1] = \prod_{j=-1, j\neq -1}^{1} \frac{\alpha - j}{-1-j} = \frac{1}{2}\alpha(\alpha-1)$$

$$h[0] = \prod_{j=-1, j\neq 0}^{1} \frac{\alpha - j}{0-j} = -(\alpha-1)(\alpha+1) = 1-\alpha^2$$

$$h[1] = \prod_{j=-1, j\neq 1}^{1} \frac{\alpha - j}{1-j} = \frac{1}{2}\alpha(\alpha+1).$$

It can be seen that this result is identical to that determined by the polynomial method described above (5.51).

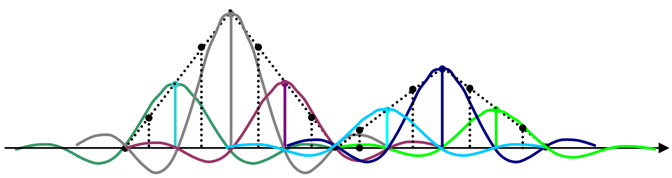


Fig. 5.47 Ideal interpolation with Nyquist filter: the dashed samples represent the interpolated signal. The filter impulse response is zero in correspondence with the samples adjacent to the one to which it refers.

Remark 5.14. The Lagrange polynomial behaviour is similar to that of the ideal interpolator filter realized by Nyquist's low-pass filter in Eqn. (5.31). This, in fact, has a null impulse response in correspondence of all the signal samples except for the i-th reference sample (see Fig.s 5.48 and 5.47).

The following Theorem is also valid.

Theorem 5.1. *For an infinite number of equally spaced samples* $t_{k+1} - t_k = \Delta$, *the Lagrange polynomial base converges to the sinc(·) function of type*

$$l_k(x) = \mathrm{sinc}\left(\frac{x-k\Delta}{\Delta}\right), \qquad k = ..., -2, -1, 0, 1, 2, ... \; .$$

Proof. Each analytical function is determined by its zeros and value at a point other than zero. Since the function $\sin(\pi x)$ is zero for x integer except zero, and since the function sinc(0) =1, it coincides with the Lagrange polynomial base for $k \to \infty$ and $k = 0$. □

However note that, as shown in Fig. 5.48, for finite length filter, the Lagrange solution in Eqn. (5.58) can also be obtained the windowing method where the window coefficients are computed using the binomial formula. For more detail on the connection between the sinc(·) function and the Lagrange interpolation refere to [67], [73]-[75].

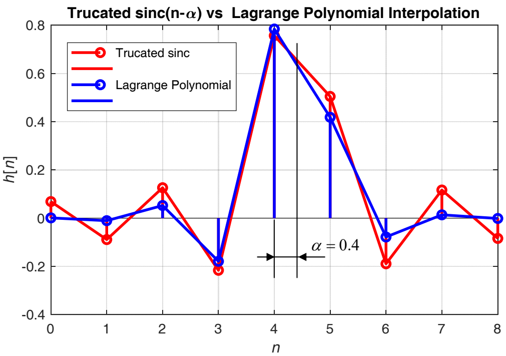


Fig. 5.48 Comparison of $L_D\{\text{sinc}(n-\alpha)\}$ (or LS solution) FIR filter impulse response and the LIF $l_n^N(x)$, evaluated with Eqn. (5.61), for $N = 8$, for fixed delay $\alpha = 0.4$. For $N \to \infty$ $\Rightarrow \{l_n^N(\alpha) \to \text{sinc}(n-\alpha)\}$

Fig. 5.49 shows the amplitude and phase delay responses for two polynomial filters. The Lagrange polynomial coefficients are evaluated with Eqn. (5.61). It can be observed that compared to the sinc(·) interpolator with the same order (see Fig. 5.33), the responses are much smoother.

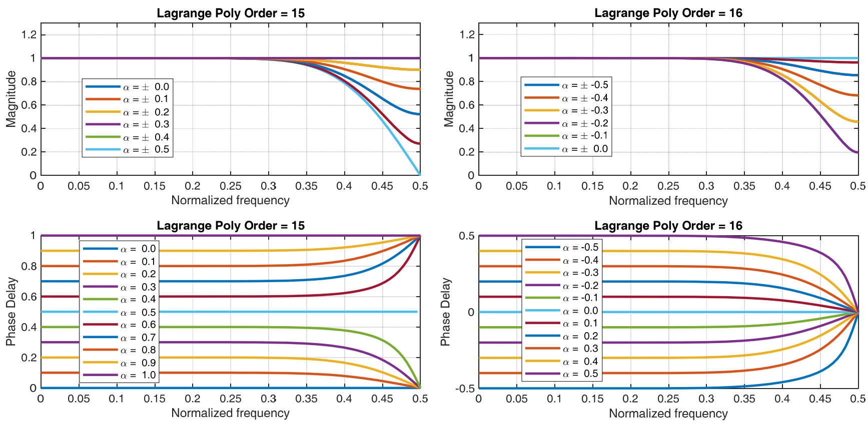


Fig. 5.49 Magnitude and phase-delay response of odd-order (left) and even-order (right) Lagrange polynomial interpolator. The polynomianl coefficients are evaluated with with Eqn. (5.61).

In Fig. 5.50 are reported the magnitude and phase-delay response of even- and odd-order Lagrange interpolating filter for fixed dealy $\alpha = 0.4$.

The Lagrange polynomial interpolation, is widely used in practice because:

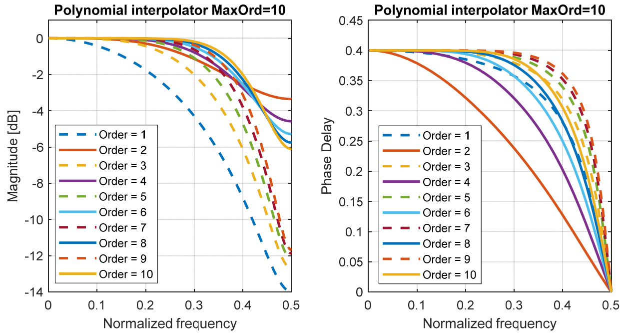


Fig. 5.50 Magnitude and phase-delay response of Lagrange interpolating filter for fixed dealy $\alpha = 0.4$. The continuous lines are related to polynomials with even orders, while the dashed lines to the odd ones.

1. no need for feedback, i.e. is implemented with a FIR filter;
2. the coefficients of the filter are easily calculable with an explicit formula that allows a simple implementability also in real time.
3. it has a maxflat frequency response the maximum of the magnitude response never exceeds unity.

Remark 5.15. Note that especially for low polynomial orders, LIF has optimal behavior up to a fraction of the Nyquist frequency. One possible approach to limit the error at high frequencies is to insert an upsampling block bringing the signal to a frequency L times higher than the input frequency. Usually a multiphase network is used that can be interpreted as as a fractional delay filter followed by an L-fold decimation actually evaluates only every L-th sample of the oversampled signal [31], [36].

5.6.4.3 Cubic B-Spline Interpolation

A methodology that guarantees more control possibilities, better performance and lower computational cost is based on the replacement of the Lagrange polynomial with a B-spline interpolator as proposed in [45]. The disadvantage of the B-spline method is a distortion of the spectrum of the interpolator filter which can, however, be easily pre-compensated with a simple equalizer to be put before the B-spline interpolator.

The theoretical development of the method is done as in the case of the polynomial interpolators seen above where the signal (at first supposed continuous) $x(t)$, $t \in [t_0, t_N]$ is evaluated by the knowledge of $N+1$ samples $x(t_k)$, $k = 0, 1, ..., N$; with the interpolating B-spline function $\Phi(t)$ such that $\hat{x}(t) = \sum_{k=0}^{N} \Phi_k(t) x(t_k)$.

Suppose we know the $N+1$ samples of a $x(t)$ signal in the range $[x(t_k), x(t_{k+1}), ..., x(t_{k+N})]$. The B-spline function of order N is defined as [4], [45].

$$\Phi_k^N(t) = \sum_{i=k}^{k+N+1} \frac{(t-t_i)_+^N}{\prod_{j=0, i\neq j}^{N+1} (t_i - t_j)} \tag{5.62}$$

where

$$(t-t_i)_+^N = \begin{cases} (t-t_i)^N & t \geq t_i \\ 0 & t < t_i. \end{cases}$$

The interpolator filter coefficients can be determined by evaluating the expression (5.62) for a generic delay by placing, in the case of uniformly spaced samples, $t_i = t_0 + i\Delta t$ e ; (i, j integer) and defining a new variable $\alpha < 1$ such that $t = t_0 + \alpha\Delta t$ we have that

$$\Phi_k^N(\alpha) = \sum_{i=k}^{k+N+1} \frac{(\alpha - i)_+^N}{\prod_{i=0, i\neq k}^{N+1}(i-j)}.$$

Since the $\Phi_k^N(t)$ function decreases as N increases, a normalized version of it is usually used defined as

$$N_k^N(t) = (t_{k+N+1} - t_k)\Phi_k^N(t)$$

then

$$N_k^N(\alpha) = (N+1)\sum_{i=k}^{k+N+1} \frac{(\alpha - i)_+^N}{\prod_{i=0, i\neq k}^{N+1}(i-j)}. \tag{5.63}$$

The calculation of the previous expression for N =2 produces the following coefficients of the interpolating FIR filter

$$\begin{aligned}
N_3^2(\alpha) &= h(0) = -\frac{1}{2}\alpha^2 \\
N_2^2(\alpha) &= h(1) = -\frac{1}{2}(1+\alpha)^2 + \frac{3}{2}\alpha^2 \\
N_1^2(\alpha) &= h(2) = -\frac{1}{2}(1-\alpha)^2.
\end{aligned}$$

For N =3 produces the following interpolating FIR filter coefficients

$$\begin{aligned}
N_3^3(\alpha) &= h(0) = \frac{1}{6}\alpha^3 \\
N_2^3(\alpha) &= h(1) = \frac{1}{6}(1+\alpha)^3 - \frac{2}{3}\alpha^3 \\
N_1^3(\alpha) &= h(2) = \frac{1}{6}(2-\alpha)^3 - \frac{2}{3}(1-\alpha)^3 \\
N_0^3(\alpha) &= h(3) = \frac{1}{6}(1-\alpha)^3.
\end{aligned}$$

Fig. 5.51 shows a comparison od a 3rd ($N = 3$) FDLs.

For N =6 you can prove that the $\Phi(t) = (\sin(t)/t)^7$. In fact, B-spline functions can be obtained by means of repeated convolutions of rectangular impulses.

Remark 5.16. The choice of the type of interpolation depends on the type of application. In general terms Rocchesso in [48] defined three properties that should be met:

- flat frequency response;
- linear phase response;
- the delay-times variation does not give rise to audible transient.

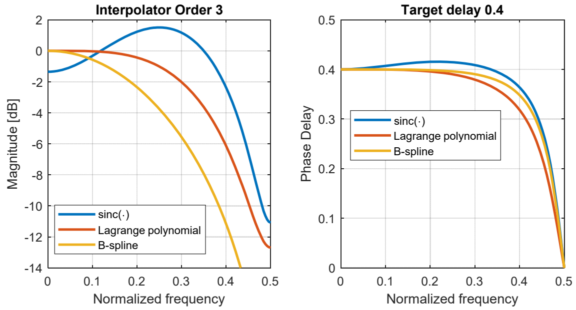


Fig. 5.51 Comparison of magnitude and phase-delay response of 3rd order FDLs.

It is clear that these properties are contradictory. For example an all-pass interpolator satisfies the 1) but not the 2) for an extended frequency range. The type of approximation choice for our application may not be easy. For more information on this, please refer to the literature in particular Laakso *et al* [28], Dattorro [46], Rocchesso [48], Smith [60] and Bhandari *et al* [68].

5.6.5 Time-Variant Delay Lines

In irrational sample rate conversion and in many digital audio effects, some of which will be described in later chapters, are based on the use of time-varying delay lines (TVDLs), i.e. delay D, or is fractional part α, is itself a time function. You therefore have

$$y[n] = x\left[n - D[n]\right]. \tag{5.64}$$

In general it may be convenient to express the variable delay over time as

$$D[n] = D_0 + D_1 f_D[n] = D_0\left(1 + m_D f_D[n]\right) \tag{5.65}$$

where D_0 represents the *nominal length* of the DL, the function $f_D[n]$, generally with null mean value, represents the variation law (or modulation type) and the constant $m_D \in [0, 1]$ represents the *modulation index*.

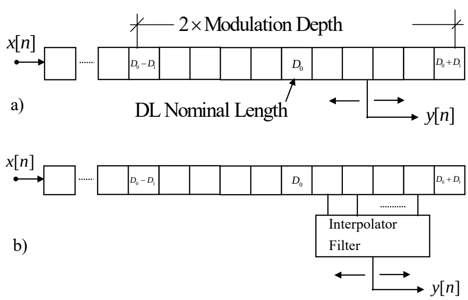


Fig. 5.52 Time-variant delay line (TVDL) a) General schema of DL with delay with variable length. b) TVDL structure with interpolator filter.

The obtained effect depends on the variation law $f_D[n]$ and its modulation depth $D_1 = m_D D_0$ (see Figure 7.30). In general the $D[n]$ value is not integer and must be interpolated with one of the techniques described in the previous paragraph.

Fig. 5.53 shows, as an example, the implementation in C++ of the Synthesis Tool Kit [61], an FD with linear interpolation.

```
static double A[N];
static double *rptr  =  A; // read ptr
static double *wptr  =  A; // write ptr

double setdelay(int M) {
    rptr  =  wptr - M;
    while (rptr < A) { rptr + =  N }
}

double delayline(double x)
{
  double y;
  A[wptr++]  =  x;
  long rpi  =  (long)floor(rptr);
  double a  =  rptr - (double)rpi;
  y  =  a * A[rpi] + (1-a) * A[rpi+1];
  rptr + =  1;
  if ((wptr-A) > =  N) { wptr - =  N }
  if ((rptr-A) > =  N) { rptr - =  N }
  return y;
}
```

Fig. 5.53 Possible implementation in C++ of a delay line with variable length and fractional delay with linear interpolator. (Courtesy of [61]).

5.6.6 Arbitrary Sampling Rate Conversion

The conversion between two arbitrary sampling frequencies, including the cases in which the ratio is an integer, rational or irrational, is of central importance in many DASP applications. In all the situations described the conversion problem consists in determining a new signal sample placed between two samples of the original signal by means of an interpolation or extrapolation process. Thus the problem can be solved by a time-varing delay line where the variation concern only the fractional part i.e. $\alpha \to \alpha(n)$ where, depending on the used technique $[-0.5 < \alpha(n) < 0.5]$ or $[0 < \alpha(n) < 1]$ (less than a fixed systematic integer delay).

In a more general form the problem of interpolation and extrapolation must be understood as a method to interface two signals with any sampling frequencies, even with irrational ratio.

An intuitive way to understand the convertion with irrational ratio, is to convert the signal back to analog and then resample it to the desired frequency. This procedure, even if theoretically consistent, is not practically feasible since it should be implemented with a dedicated and almost non-programmable hardware structure; furthermore, the conversion processes would still produce a series of artifacts, distortions, noise, etc. inherent in the A/D - D/A conversion.

In online audio applications, the ADC - DAC process is simulated using appropriate approximate interpolation scheme as polynomials or splines. In this case the rational interpolation process can be seen as a cyclic time-varying filtering process in which the filter coefficients are calculated at each sample of the input signal [33]-[42].

Remark 5.17. Observe that, in arbitrary sampling rate conversion the converted sample is a function of the known neighboring samples so, as illustrated in Fig. 5.54, for each new sample the fractional part and the relative interpolator filter must be recalculated. In real time applications, it is therefore necessary to have efficient algorithms both for the calculation of the filter coefficients and for the filtering operation.

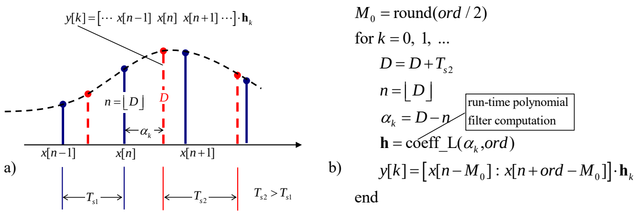


Fig. 5.54 Arbitrary sampling rate conversion. a) Example of dawn-sampling $T_{s1}\to T_{s2}$, with $T_{s2}>T_{s1}$. b) Possible simple implemenation algorithm.

The design of the interpolator filter is quite critical. In the previous paragraph we have seen that for the correct definition of the group delay on the whole band, there is a bandwidth restriction and *vice-versa*.

Fig. 5.55 shows an example of conversion, using different order Lagrange polynomial and all-pass techniques, from a sample frequency of 48kHz to 44.1kHz. In order to define a qualitative metric evaluation, the input signal consists of six sine waves centered at 20Hz, 200Hz, 1kHz, 10kHz, 15kHz and 20kHz, to cover a large portion of the spectrum at the maximum amplitude allowed (0dB). From the figure we can observe that all techniques produce artifacts especially at the higher frequencies (>10kHz in this case). As we could have expected from the discussion in the previous paragraphs,

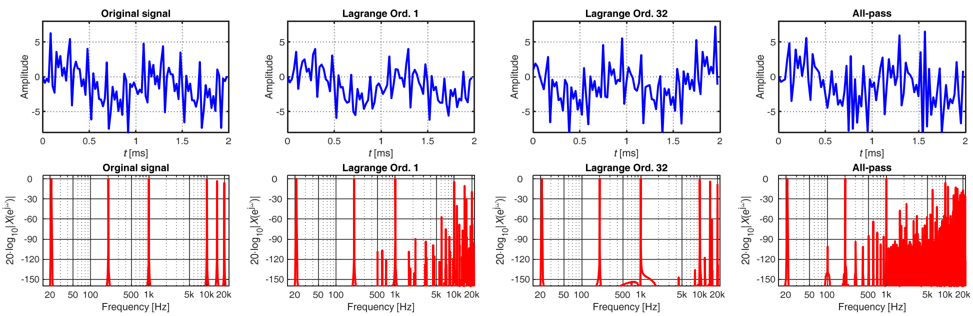


Fig. 5.55 Example of irrational signal sample-rate conversion from 48kHz to 44.1kHz, very common in audio applications. In the upper part is reported 2ms of the time-domain signal. In the lower part the magnitude spectrum of the original signal end the signal converted using different interpolator filters.

the worst result is that of the all-pass interpolator. The artifacts produced by the Lagrange filter of order 32, are all below -60 dB, therefore, with inaudible effects.

Remark 5.18. Note that, the computational cost of an N-order Lagrange intropolator is equal to the standard cost of the FIR filter: $N+1$-multiplications + N-additions $\times$ sample; to which the cost for the calculation of the filter parameters must be added. If we use the expression (5.58) to calculate the parameters we should add more N-multiplications + $2N$-additions $\times$ sample.

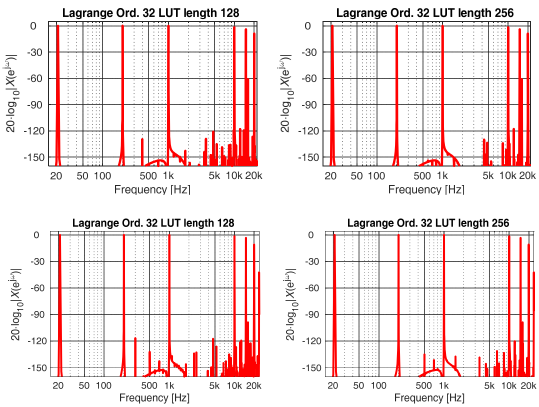


Fig. 5.56 Spectra of converted signals with irrational sampling rate using preloaded interpolation filters on different length LUTs. (Upper) down-sampled from 48kHz to 44.1kHz. (Lower) up-sampled from 44.1kHz to 48kHz.

Arbitrary conversion between sampling rates and its efficient implementation is a central theme in the DASP. For improved computational efficiency in [73], it has been proposed to store interpolator filter coefficients on an LUT, for a sufficient number of fractional delays α_k. This method is commonly used in closely related sample rate conversion problems.

By the way, in Fig. 5.56 are reported the results of the re-sampling process with Lagrange filters stored in LUTs of different lengths. The experiment was performed with the same signal in Fig. 5.55. From the figure we can observe that with a LUT with 256 pre-memorized coefficients produces a result very similar to that with run-time calculated filters.

5.6.7 Robust Fractional Delay FIR Filter

The implementation of time-variant FDL FIR, can have some critical issues when updating the filter status. In DASP, in particular, it can lead to audible and annoying artefacts. So sometimes, as seen also in the case of equalizers (see §4.5), it is convenient to use robust architectures in general at the expense of a computational overhead.

5.6.7.1 The Farrow's Structure

In order to make the system more robust, Farrow in [32] proposes to use a parallel filter bank *a priori* determined and kept fixed and, as indicated in Fig. 5.57, the

$D = D_i + \alpha$ parameter that regulates delay, can be placed outside the fixed filter bank [27]-[32].

The fixed filter bank, can therefore be realized with very robust and efficient circuit structures, especially in case you want to realize a dedicated hardware [33]-[40].

Let $h_D[n]$, the filter impulse response of the interpolation filter relate to the delay D, Farrow in [32], proposed to approximate each impulse response with a M-order polynomial of the type

$$h_D[n] = \sum_{m=0}^{M} c_{m,n} D^m, \qquad n = 0, 1, ..., N \tag{5.66}$$

where M is the order of the polynomials. The TF of the above expression can be written as

$$H_D(z) = \sum_{n=0}^{N} \sum_{m=0}^{M} c_{m,n} D^m z^n = \sum_{m=0}^{M} \left(\sum_{n=0}^{N} c_{m,n} z^n \right) D^m = \sum_{m=0}^{M} C_m(z) D^m \tag{5.67}$$

where $C_m(z)$, are the fixed TFs of a parallel FIR filters bank.

Property 5.6. For exact Lagrange interpolation of order N, the order of the subfilters $C_m(z)$ must equal to N (see for example [29]). The resulting Farrow's structure shown in Fig. 5.57, comes from the rewriting of the polynomial (5.67) with Horner's method. So it results

$$H_D(z) = C_0(z) + [C_1(z) + [C_2(z) + \cdots + [C_{N-1}(z) + C_N(z) \underbrace{D]D \cdots]D}_{N}. \tag{5.68}$$

As for Lagrange polynomial interpolation, a common choice for the delay is $D_i = N/2$ and $\alpha \in [-0.5, 0.5)$.

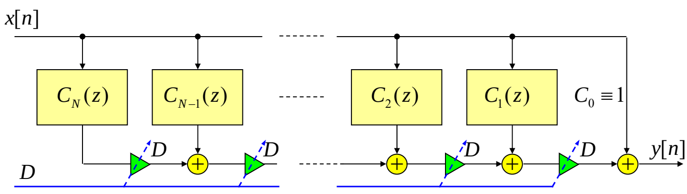


Fig. 5.57 Interpolation by the Farrow structure basis filter. The only variable parameter is external to the TF's filters-bank so the structure is robust.

The consistency of Farrow's method can be demonstrated by the computability of bank TFs $C_m(z)$.

5.6.7.2 LS Computation of the Farrow's Sub-Filters Coefficients

The Farrow structure, consists of bank of N-filters each with M parameters and, given the high order of degrees of freedom, the determination of the $C_m(z)$ sub-filters, which approximates a desired response, indicated as $\mathbf{D}$, can be done in several ways.

Here, the determination of the solution is done by minimizing a given cost function (CF) with the least squares (LS) criterion. For the formulation of the CF the following definitions and assumptions are considered.

First of all, we write the (5.66) as the following scalar product

$$\mathbf{h}_D = \mathbf{v}_D^T \mathbf{C}$$

where $\mathbf{v}_D \in \mathbb{R}^{(M+1)\times 1} = [1 \;\; D \;\; D^2 \;\; \cdots \;\; D^M]^T$ is the vector of delays, and $\mathbf{C} \in \mathbb{R}^{(M+1)\times(N+1)}$ is the matrix of overall polynomial coefficients of the filter bank that represents the FDL impulse response $\mathbf{h}_D$.

By choosing a set of values D_0 D_1 ..., D_L, usually uniformly spaced, where we want to approximate the desired response of the bank, indicated as $\mathbf{D} \in \mathbb{R}^{(L+1)\times(N+1)}$ and related to these values, we can write the CF as

$$J(\mathbf{H}) = \|\mathbf{D} - \mathbf{V}_D \mathbf{C}\|^2$$

where the matrix $\mathbf{V}_D \in \mathbb{R}^{(L+1)\times(M+1)} = [\mathbf{v}_{D_0}^T \;\; \mathbf{v}_{D_1}^T \;\; \cdots \;\; \mathbf{v}_{D_L}^T]^T$ contains the grid of delays. Being the quadratic cost function, the optimal solution is unique. By setting its gradient to zero you get the following normal equations

$$\mathbf{V}_D \mathbf{C} = \mathbf{D}$$

left multiply both members by $\mathbf{V}_D^T$ and solving w.r.t. $\mathbf{C}$, the optimal LS solution is

$$\mathbf{C}_{LS} = \left(\mathbf{V}_D^T \mathbf{V}_D\right)^{-1} \mathbf{V}_D^T \mathbf{D}.$$

The above solution is general for $M < N$ and $L > N$.

5.6.7.3 Computation of the Farrow's Sub-Filters Coefficients

Following a simpler and more direct approach, as proposed by Välimäki in [27] and reconsidered in [35], the determination of the coefficients $c_{m,n}$ of the TF $C_m(z)$, can be done by imposing equality of the overall Farrow TF and a target TF $T(z)$. If the output is a D-delayed version of the input the desired relation is $T(z) = \frac{Y(z)}{X(z)} = z^{-D}$. So, for Eqn. (5.67), and imposing this condition for various delays b_n, for n=0,1, ...; which for simplicity we consider integers we have that

$$H_D(z) \therefore \underset{c_{m,n}\in\mathbb{R}}{\arg\min} \left\| \sum_{m=0}^{N} C_m(z) b_n^m - z^{-b_n} \right\|, \qquad b_n = 0,\ 1,\ \ldots \tag{5.69}$$

where $\{b_n\}$ can be the set of the natural numbers. Thus, for simplicity, considering $N+1$ relations as $b_n = 0,\ 1,\ \ldots,\ N$, and writing the Eqn. (5.69) extensively we get

$$
\begin{aligned}
c_{0,0}0^0 + c_{0,1}0^1 + \cdots + c_{0,N}0^N &= z^{-0} \\
c_{1,0}1^0 + c_{1,1}1^1 + \cdots + c_{1,N}1^N &= z^{-1} \\
&\vdots \\
c_{N,0}N^0 + c_{N,1}N^1 + \cdots + c_{N,N}N^N &= z^{-N}
\end{aligned}
\tag{5.70}
$$

i.e. a set of $N+1$ equations that in matrix form can be written as

$$\mathbf{CV} = \mathbf{Z}$$

where $\mathbf{V}$ is a Vandermonde matrix defined as

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

and $\mathbf{Z} = \mathrm{diag}(1\ z^{-1}\ \cdots\ z^{-N})$ is the matrix of delay such that the coefficients $c_{m,n}$ of the TF $C_m(z)$ can be determined as

$$\mathbf{C} = \mathbf{ZQ}$$

where $\mathbf{Q} = \mathbf{V}^{-1}$ indicate the inverse of the Vandermonde matrix $\mathbf{V}$. In other words, as $\mathbf{z}$ represents the delay elements in z-domain, according with the above equation, the TFs $C_m(z)$ are obtained as the following scalar product

$$C_m(z) = \mathbf{q}(m)\mathbf{z}, \qquad m = 0,\ 1,\ ...,\ N \tag{5.71}$$

where with $\mathbf{q}(m)$ indicate the raw of the matrix $\mathbf{Q}$.

For example, for $N = 4$ we have that

$$\mathbf{V} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 & 16 \\ 1 & 3 & 9 & 27 & 81 \\ 1 & 4 & 16 & 64 & 256 \end{bmatrix}, \qquad \mathbf{Q} = \mathbf{V}^{-1} = \frac{1}{24}\begin{bmatrix} 24 & 0 & 0 & 0 & 0 \\ -50 & 96 & -72 & 32 & -6 \\ 35 & -104 & 114 & -56 & 11 \\ -10 & 36 & -48 & 28 & -6 \\ 1 & -4 & 6 & 4 & 1 \end{bmatrix}$$

so, the Farrow's TFs are

$$
\begin{aligned}
C_0(z) &= 1 \\
C_1(z) &= -\tfrac{25}{12} + 4z^{-1} - 3z^{-2} + \tfrac{4}{3}z^{-3} - \tfrac{1}{4}z^{-4} \\
C_2(z) &= \tfrac{35}{24} - \tfrac{13}{2}z^{-1} + \tfrac{19}{4}z^{-2} - \tfrac{7}{3}z^{-3} + \tfrac{11}{24}z^{-4} \\
C_3(z) &= -\tfrac{5}{12} + \tfrac{3}{2}z^{-1} - 2z^{-2} + \tfrac{7}{6}z^{-3} - \tfrac{1}{4}z^{-4} \\
C_4(z) &= \tfrac{1}{24} - \tfrac{1}{6}z^{-1} + \tfrac{1}{4}z^{-2} - \tfrac{1}{6}z^{-3} + \tfrac{1}{24}z^{-4}
\end{aligned}
$$

while the entire TF of the Farrow structure is

$$H_D(z) = C_0(z) + [C_1(z) + [C_2(z) + [C_3(z) + C_4(z)D]D]D]D.$$

To consider the fractional part only, we remove the integer part D_i. So, you need to properly transform the matrix $\mathbf{Q}$ as

$$\mathbf{C} = \mathbf{TQ}$$

where $\mathbf{T}$ is a transformation matrix defined as

$$\mathbf{T}^T_{j,k} = \begin{cases} M^{j-k} \cdot \dfrac{j!}{j!(j-k)!}, & \text{for } j \geq k \\ 0, & \text{for } j < k \end{cases}, \quad \text{where} \quad M = \begin{cases} \dfrac{N}{2}, & N\text{-even} \\ \dfrac{N-1}{2}, & N\text{-odd.} \end{cases}$$

This transformation is equivalent to substituting $D' = D - 1$.

```
function [C, V, Q, T] = Farrow_TF(N)
    if isodd(N), M =  (N-1)/2;
    else M = N/2,  end
    for m=0:N
        for n=0:N
            V(m+1,n+1) = m^n;
            if n>=m
                T(m+1,n+1) = M^(n-m)*nchoosek(n,m) ;
            end
        end
    end
    Q = inv(V);
    C = T*Q ;
end
```

```
function h = Farrow_imp_resp(C, alpha)
    M = length(C(1,:));
    N = M - 1;
    x = zeros(M,1);
    x(1) = 1;
    h = alpha*conv(C(N+1,:), x);
    for i= N : -1 : 1
        h = alpha*h + conv(C(i,:), x );
    end
end
```

Fig. 5.58 Example of MATLAB code for Farrow sub-filters specifications.

For example for $N = 3$ and $N = 4$ we have

$$\mathbf{T}^{(3)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \ \mathbf{C}^{(3)} = \frac{1}{6}\begin{bmatrix} 0 & 6 & 0 & 0 \\ -2 & -3 & 6 & -1 \\ 3 & -6 & 3 & 1 \\ -1 & 3 & -3 & 1 \end{bmatrix}; \ \mathbf{T}^{(4)} = \begin{bmatrix} 1 & 2 & 4 & 8 & 16 \\ 0 & 1 & 4 & 12 & 32 \\ 0 & 0 & 1 & 6 & 24 \\ 0 & 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \ \mathbf{C}^{(4)} = \frac{1}{24}\begin{bmatrix} 0 & 0 & 24 & 0 & 0 \\ 2 & -16 & 0 & 16 & -2 \\ -1 & 16 & -30 & 16 & -1 \\ -2 & 4 & 0 & -4 & 2 \\ 1 & -4 & 6 & -4 & 1 \end{bmatrix}.$$

Thus the shifted Farrows sub-filters TFs, respectively, are

$$\begin{aligned} C_0^{(3)}(z) &= z^{-1} \\ C_1^{(3)}(z) &= -\tfrac{1}{3} - \tfrac{1}{2}z^{-1} + z^{-2} - \tfrac{1}{6}z^{-3} \\ C_2^{(3)}(z) &= \tfrac{1}{2} - z^{-1} + \tfrac{1}{2}z^{-2} + \tfrac{1}{6}z^{-3} \\ C_3^{(3)}(z) &= -\tfrac{1}{6} + \tfrac{1}{2}z^{-1} - \tfrac{1}{2}z^{-2} + \tfrac{1}{6}z^{-3} \end{aligned}, \qquad \begin{aligned} C_0^{(4)}(z) &= z^{-2} \\ C_1^{(4)}(z) &= \tfrac{1}{12} - \tfrac{2}{3}z^{-1} + \tfrac{2}{3}z^{-3} - \tfrac{1}{12}z^{-4} \\ C_2^{(4)}(z) &= -\tfrac{1}{24} + \tfrac{2}{3}z^{-1} - \tfrac{15}{12}z^{-2} + \tfrac{2}{3}z^{-3} - \tfrac{1}{24}z^{-4} \\ C_3^{(4)}(z) &= -\tfrac{1}{12} + \tfrac{1}{6}z^{-1} - \tfrac{1}{6}z^{-3} + \tfrac{1}{12}z^{-4} \\ C_4^{(4)}(z) &= \tfrac{1}{24} - \tfrac{1}{6}z^{-1} + \tfrac{1}{4}z^{-2} - \tfrac{1}{6}z^{-3} + \tfrac{1}{24}z^{-4}. \end{aligned}$$

Property 5.7. From Fig. 5.59 we can observe that the TFs of the sub-filters, except for the $C_0(z)$, around $\omega = 0$, have high pass characteristics typical of the differentiators filters [1]-[3]. This depends on the global TF required by the bank. In fact, by developing the target function $H_D(e^{j\omega}) = e^{-j\omega D}$ in Taylor's series we get

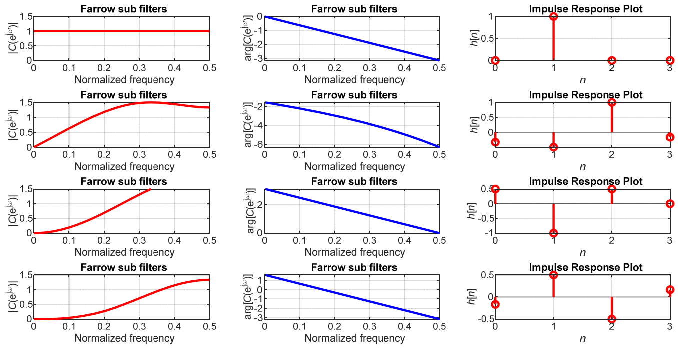


Fig. 5.59 Farrow sub-filters characteristic for $N = 3$. Note that $C_0(z)$ (top) is a simple delay while $C_1(z)$, $C_2(z)$ and $C_3(z)$, for $\omega \to 0$, have the typical high-pass characteristics of a differentiator filter.

$$
\begin{aligned}
H_D(e^{j\omega}) = e^{-j\omega D} &= \sum_{m=0}^{\infty} \frac{(-1)^m}{m!} \alpha^m (j\omega)^m e^{-j\omega D_i} \\
&= \sum_{m=0}^{\infty} \frac{(-1)^m}{m!} \alpha^m H_m(e^{j\omega})
\end{aligned}
\tag{5.72}
$$

where the $H_m(e^{j\omega})$ consists of the frequency response of the m order differentiator that for $m = 0$ is a simple delay.

The previous property suggests that the coefficients of the Farrow structure may be obtained using FIR approximations of ideal differentiators. This property, as we will also see in the following paragraphs, has been used for the alternative design of Farrow's structure [35]. However, given the high freedom degree in choosing sub-filters, the reverse is not necessarily true. Depending on the chosen philosophy design, the sub-filters are not required to approximate differentiators [38].

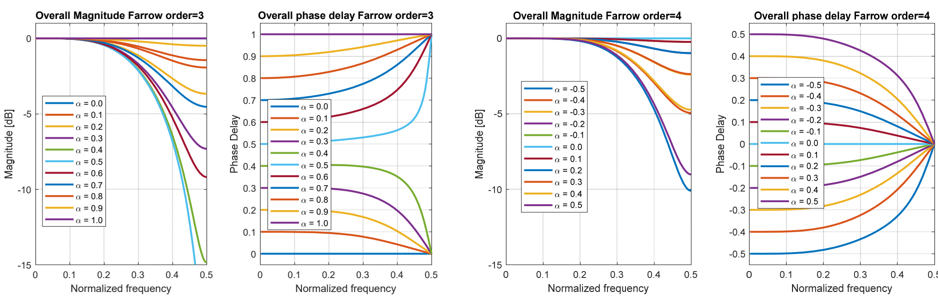


Fig. 5.60 Farrow FDL for $N = 3$ (left) and $N = 4$ (right).

Remark 5.19. Note that, the computational cost of Farrow's robust structure requires the calculation, even if in parallel, of N FIR filters for which you have $N(N+1)$-multiplications, N^2-additions. Considering also the N-multiplications for D, we have a cost computational total of N^2+2N-multiplications and N^2+2N-additions $\times$ sample. Moreover, since the delay line is shared throughout the bank, many intermediate calculations are common to multiple filters. In the above example for $N=4$, the term $\frac{2}{3}z^{-3}$ is common to $C_1(z)$ and $C_2(z)$. You can also see how many other terms are common to less than one sign. These terms are easily grouped together in order to decrease the number of multiplications, which generally requires more computation than sums.

However, more efficient architectures are available in the literature. For example in [43], a Farrow structure is proposed in which the filter bank is implemented with coefficients with values of power of two. So it is possible to reduce the multiplier complexity in case of hardware implementation.

5.6.8 Modular Structures by LIF Taylor Expansion

The LIF Taylor expansion, as seen above in Eqn. (5.72), consists of a set of basis-functions consisting of simple differentiator filters. Based on this observation, Candan in [76], proposes a simple structure based on LIF Taylor expansion carried out directly in the discrete-time domain.

Before proceeding, to simplify the development as suggested in [77], let's consider the following useful notations.

1. The term $\delta(\cdot)$ indicate the difference operator: $\delta(f[n]) = f[n]-f[n-1]$, i.e. the TF is $(1-z^{-1})$.
2. The term $f^{[N]} = f(f+1)(f+2)\cdot\cdots\cdot(f+N-2)(f+N-1)$, indicate the *factorial polynomials* (aka *rising factorials* or *Pochhammer symbol*);
3. let us indicate its first-difference (as $\frac{d}{df}f^N = Nf^{N-1}$), by the following recursive formula $\delta(f^{[N]}) = Nf^{[N-1]}$.

With the above notations the LIF truncated discrete-time Taylor series can be written as

$$y[t] = \sum_{n=0}^{N} \delta^n(x[k]) \cdot \frac{(t-k)^{[n]}}{n!} \tag{5.73}$$

the previous expression is also denoted as *Newton's backward difference formula* [78].

Now, indicating $\delta^n(x[k]) = \delta^{n-1}(x[k]) - \delta^{n-1}(x[k-1])$, with simple manipulations the (5.73) can be written recursively as

$$\delta^n(x[k]) \cdot \frac{(t-k)^{[n]}}{n!} = -\delta^{n-1}(x[k]) \cdot \frac{(t-k)^{[n-1]}}{(n-1)!} \cdot \frac{t-k+N-1}{N} \cdot \delta(x[k]). \tag{5.74}$$

Indicating with $D=k-t$ the desired fractional delay, the expression (5.74) yields a modular structure shown in Fig. 5.61, also denoted as a *Newton fractional-delay* (NFD) filter. In this case the sub-filters are simple differentiators implementable without multiplication, and the complexity for a N-order filter is $O(N)$, instead of $O(N^2)$ of Farrow's structure.

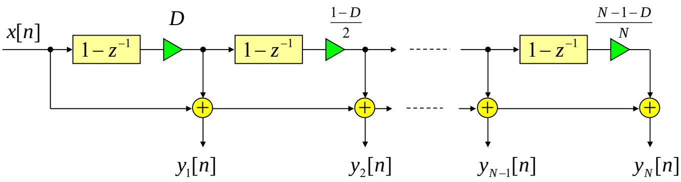


Fig. 5.61 Modular LIF structure also denoted as Newton fractional-delay (NFD) filter. At the summer outputs all LIF interpolators order are available (Modified from [76]).

In the LIF structure of Fig. 5.61 in the intermediate outputs, the delayed input signal is present at the same time, for different delay parameters. In addition, note that the structure in Fig. 5.61 consists of a series of sub-filters whose output is multiplied by a term that depends on the target delay D.

However, each output sample depends on the current and past values of the delay parameter. Therefore, the above LIF structure does not work correctly for the time-varying fractional delay (e.g., as in sample rate conversion).

To overcome the problem, in [79] a new architecture based on Newton's backwards difference formula (5.73) has been proposed, called the *Newton-transpose interpolation structure* (TNIS) illustrated in Fig. 5.62.

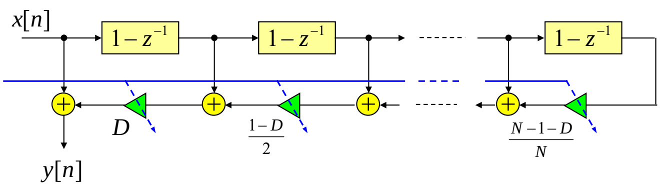


Fig. 5.62 Modular LIF structure also denoted as transposed Newton fractional-delay (TNFD) filter. (Modified from [79]).

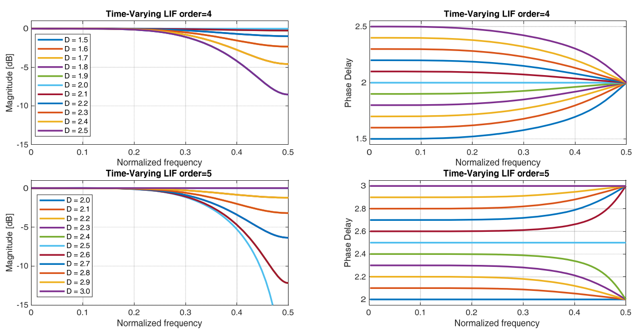


Fig. 5.63 Magnitude and absolutes delay responses of modular LIF implemented with the structure in Fig. 5.62 of orders 4, 5.

In this case the upper part of the structure is similar to a delay-line where the delay elements are replaced by differentiator-elemets $z^{-1} \to (1-z^{-1})$, with no intermediate operations. The parameter that defines the fractional delay is external to this chain so it can be time-varying because, as in Farrow's structure, it does not affect the internal state of the sub-filters. This is very interesting for audio applications as it is very robust, efficient, easily scalable and feasible in both software and dedicated hardware structures.

5.7 Digital Oscillator

As for analog signals very often also for numerical sequences, the generation of periodic waveforms as sines, quare waves, triangular waves etc. is necessary.

A simple approach for the generation of periodic signals consists in determining a filter with TF $H(z)$ such that its impulsive response $h[n]$ is equal to the desired waveform. By sending an unitary impulse $\delta[n]$ to this filter the desired waveform is generated. The computational cost of the generator is therefore that of the filtering process.

However, a more efficient way to generate any waveform is the so-called *wavetable technique*. The technique simply consists in storing a period of the waveform on a table consisting of a Random Access Memory (RAM) organized as a circular buffer and reading its contents periodically. The reading period is therefore equivalent to the fundamental frequency of the waveform.

5.7.1 Sinusoidal Digital Oscillator

The approach is based on the synthesis of a certain TF chr allows the simple generation of a sinusoidal signal at frequency f_0 with sampling frequency f_s [3]. Said $\omega_0 = 2\pi f_0/f_s$ the sinusoid pulsation the desired impulse response is the following

$$h[n] = R^n \sin(\omega_0 n) u[n]$$

which for $0 < R < 1$, corresponds to the generation of a sinusoid with exponential decay. In case $R = 1$, the generated signal is a pure sinusoid with frequency f_0. The corresponding TF results

$$H_s(z) = \frac{R \sin\omega_0 z^{-1}}{1 - 2R\cos\omega_0 z^{-1} + R^2 z^{-2}}.$$

Similarly, it is possible to generate a cosine wave oscillation. In this case the $h[n]$ results to be equal to

$$h[n] = R^n \cos(\omega_0 n) u[n]$$

and the resulting TF $H(z)$ is

$$H_c(z) = \frac{1 - R\cos\omega_0 z^{-1}}{1 - 2R\cos\omega_0 z^{-1} + R^2 z^{-2}}.$$

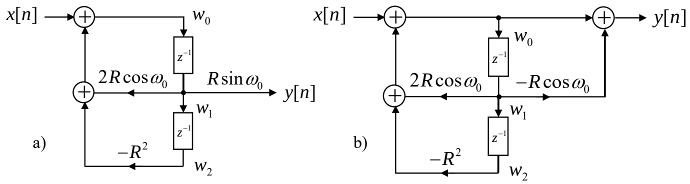


Fig. 5.64 Discrete-time circuit diagrams for the implementation of a pure oscillator. a) Sine oscillator. b) Cosine oscillator.

5.7.2 Wavetable Oscillator

In computer the wavetable oscillator is one of the earliest techniques use in sound synthesis [54]. The wavetable oscillator is realized with a RAM memory in which is stored a period of the waveform that is read periodically with a certain speed. The read values are sent to a digital to analog converter (DAC) that produces an analog output signal. The table on which the waveform is stored is called *look-up table* (LUT) and is generally read with the circular buffer technique described above [3], [55].

Let D be the length of the LUT where period of the waveform is stored, let f_s be the reading speed of the table samples, the frequency of the periodic sound is equal to $f = f_s/D$. However, if we want to realize a sound with the same waveform but with a different frequency, we can proceed in two ways:

- by varying the reading frequency f_s of the table;
- by virtually varying, with appropriate fractional interpolation techniques, the length of the table.

Since, for the pitch variation of the memorized sound, it is more complex to vary the LUT reading sampling frequency, the second technique is almost always considered.

The change in the length of the table is obtained by means of an interpolation process. That is, a table of a certain length is used (generally a rather long table is preferred) by taking the most appropriate value each time or by interpolation (see §5.6) between two (or more) adjacent points or by using the abscissa value closest to the desired one (zero order interpolation).

Said sampling increment n_{SI}, the distance between the two samples read successively, the fundamental frequency of the sound produced results as follows

$$f_0 = \frac{n_{SI} f_s}{D}. \tag{5.75}$$

Wavetable synthesizers read the sampled waveforms sequentially from the buffer and the sample can be read with a variable increment pointer.

Said p the instantaneous phase of the oscillator (pointer to the circular buffer) the reading algorithm can be implemented simply with a module D

$$\begin{aligned} p &= (p + n_{SI})\%D \\ s &= A \cdot \mathrm{LUT}[p] \end{aligned}$$

where A is the signal amplitude, LUT is the table where the waveform is stored, and s represents the output signal.

Remark 5.20. Note that, as we will explore in Chapter XXXX, the wavetable oscillator is the basic element for the realization of numerous sound synthesis techniques. This type of oscillator is used both in real time, with dedicated hardware, and in deferred time (off-line) with programs that store the file containing the song that will be available for listening in times after its generation [54]-[58].

5.7.2.1 Band-limited and Fractional Delay Wavetable Oscillator

Many waveforms useful in sound synthesis have discontinuities. Square and triangular and other waveforms, or waveforms derivative, have a spectrum that is not band limited, so their direct use in wavetable oscillators would give rise to aliasing phenomena. In the presence of aliasing, the harmonics above the Nyquist limit fold back in the low frequencies, causing annoyance and noise in the audio range [57]-[58].

Remark 5.21. A simple method to avoid aliasing is that the stored waveform must be appropriately limited in band, and the discontinuity should be rounded near the desired sampling time.

If in Eqn. (5.75) the parameter $n_{SI} > 1$, the sampling period is lowered. In this case, care must be taken to respect the Nyquist frequency. In fact, interpolated wavetable synthesis is not guaranteed to be bandlimited when the phase increment is larger than one sample

Thus, for a waveform with many harmonics, usually an upper bound is imposed on the phase increment by the highest harmonic of the signal in the wavetable. Let N_h the harmonic number, a common choice is

$$\max(n_{SI}) = \frac{Df_0}{f_s N_h} = \frac{D}{PN_h}$$

where $P = f_s/f_0$, not normally integer. Otherwise, to obtain a limited band signal, an anti-aliasing/anti-image filter must be applied before conversion. For example, to increase the pitch of a signal the sampling frequency should be augmented ($P < 1$). Consequently, to prevent aliasing must provide effective lowpass filtering. So, according to Eqn. (4.96), you need to consider an anti-aliasing filter $h_a(t) = \text{sinc}(f_a t)$ with $f_a = \min(f_s/2, f_0/2)$ (see §4.7.3).

In other words, up/down-sampling algorithm requires a low-pass filtering with a variable cutoff frequency that is controlled by the conversion ratio P. Thus, anti-aliasing wavetable methods utilize variable fractional delay filters as an essential part of the oscillator algorithm [64].

5.7.2.2 Signal-To-Noise Ratio of Wavetable Oscillator

The signal-to-noise ratio (SNR) of the wavetable oscillator can be analyzed with simple considerations in [55]-[56].

For a table of length $D = 2^b$ said $x_i[n]$ the reference signal (obtained with an ideal sampling) and $x[n]$ the signal obtained from the wavetable oscillator, the RMS error is defined as

$$e[n] = \sqrt{\frac{\sum\limits_{n=1}^{D} (x_i[n] - x[n])^2}{D}}.$$

Considering this error as additive disturbance, and zero-mean signals, we can easily calculate SNR as

$$SNR = \frac{\sigma_x^2}{\sigma_e^2} = \frac{\sum x^2[n]}{\sum e^2[n]}, \quad \text{or in decibel} \quad SNR_{dB} = 10 \log_{10}\left(\frac{\sigma_x^2}{\sigma_e^2}\right). \tag{5.76}$$

Therefore, the SNR depends on the input signal statistics: if the input signal level is low, the SNR decreases.

For a signal sample representation with a b-bit long word, the q quantization step is defined as $q = 2x_{\max}/2^b$ where $x_{\max} \therefore |x[n]| \leq x_{max}$, is the maximum level of the input signal. Considering uniform distributed quantization error, the noise variance turns out to be

$$\sigma_e^2 = \frac{q^2}{12} = \frac{x_{\max}^2}{(3)2^{2b}}.$$

So, the Eqn. (5.76) can be rewritten as

$$SNR = \frac{\sigma_x^2}{\sigma_e^2} = \frac{\sigma_x^2}{\frac{x_{\max}^2}{(3)2^{2b}}} = \frac{(3)2^{2b}}{\left(\frac{x_{\max}}{\sigma_x}\right)^2}$$

that in dB is

$$SNR_{dB} = 20 \log_{10} \frac{(3)2^b}{\left(\frac{x_{\max}}{\sigma_x}\right)} = 6.02b + 4.77 - 20 \log_{10}\left(\frac{x_{\max}}{\sigma_x}\right).$$

So, for a maximum amplitude sinusoidal signal we get

$$SNR_{dB} \approx 6.02b + 1.76$$

for a uniform distributed signal we have that

$$SNR_{dB} \approx 6.02b$$

while for a Gaussian signal the SNR is

$$SNR_{dB} \approx 6.02b - 8.5.$$

For an audio signal the Gaussian distribution is the one that is closest to reality and therefore the SNR is 8.5 [dB] lower than the best case. For example, on linear audio CDs, the signal is represented with 16 bits so the SNR is about 87.8 [dB].

References

1. A.V. Oppenheim, R.W. Schafer, J.R. Buck, "Discrete-Time Signal Processing", 3rd Edition, Pearson Education, 2010.
2. L.R. Rabiner and B. Gold, "Theory and Application of Digital Signal Processing", Prentice-Hall, Inc, Englewood Cliffs, N.J., 1975.
3. S. J. Orfanidis, "Introduction to Signal Processing", Prentice Hall, ISBN 0-13-209172-0, 2010.
4. P. Dutilleux, U. Zölzer, "Filters", in DAFX, Digital Audio Effects, J. Wiley, pp. 31-62, 2002.
5. D. Rocchesso, "Introduction to Sound Processing," ISBN-10: 8890112611, http://freecomputerbooks.com/Introduction-to-Sound-Processing.html, 2004.
6. W. H. Kautz, "Transient synthesis in the time domain," IRE Trans. Circuit Theory. 1(3):29-39, Sep. 1954.
7. T. Y. Young and W. H. Huggins, "Discrete orthonormal exponentials," in Proc. Nat'l Elec. Conj, pp. 10-18, Oct. 1962.
8. P. W. Broome, "Discrete orthonormal sequences," Journal As-SOC. Comput. Machinery. 12(2): 151-168, Dec 1965.
9. H. J. W. Belt, " Orthonormal bases for adaptive filtering," Eindhoven: Technische Universiteit Eindhoven, https://doi.org/10.6100/IR491853, 1997.
10. A.C. den Brinker, H.J.W. Belt, "Using Kautz Models in Model Reduction," in Signal Analysis and Prediction Ed. Di Ales Prochazka, N.G. Kingsbury, P.J.W. Payner, J. Uhlir, ISBN : 978-1-4612-7273-1, 1998.
11. P. S. C. Heuberger, T. J. de Hoog, P. M. J. van den Hof, and B. Wahlberg, "Orthonormal basis functions in time and frequency domains: Hambo transform theory," SIAM J. Control Optim., vol. 42, no. 4, pp. 1347-1373, 2003
12. B. Wahlberg, "System Identification Using Kautz Models, " IEEE Trans. on Automatic Control, Vol 39, No. 6 June 1994.
13. T. Paatero and M. Karjalainen, "Kautz Filters and generalized frequency resolution: Theory and audio applications," 110th AES Convention, Amsterdam, The Netherlands, 12-15, May 2001.
14. T J. Mourjopoulos and M.A. Paraskevas, Pole and zero Modeling of room transfer functions, J. Sound and Vib., vol. 146, 281-302, 1991.
15. Y. Haneda, S. Makino, and Y. Kaneda, "Common acoustical pole and zero modeling of room transfer functions, " IEEE Transactions on Speech and Audio Processing, vol. 2, no. 2, pp. 320 - 328, April 1994.
16. G. Bunkheila , R. Parisi and A. Uncini, "Model order selection for estimation of Common Acoustical Poles," IEEE International Symposium on Circuits and Systems, Seattle, WA, USA , pp.1180-1183, 8-21 May, 2008.
17. G. Vairetti, E. De Sena, M. Catrysse, S. H. Jensen, M. Moonen, T. van Waterschoot, "A Scalable Algorithm for Physically Motivated and Sparse Approximation of Room Impulse Responses With Orthonormal Basis Functions," IEEE/ACM Transaction On Audio, Speech, Ana Language Processing, Vol. 25, No. 7, July 2017.
18. T. Oliveira e Silva, "Rational orthonormal functions on the unit circle and on the imaginary axis, with applications in system identification," http://www.ieeta.pt/ tos/bib/8.2.ps.gz, 1995
19. Tomas Oliveira e Silva, "Optimality conditions for truncated Laguerre networks," IEEE Transactions on Signal Processing, vol. 42, no. 9, pp. 2528-2530, Sept. 1994.
20. Tomas Oliveira e Silva, "Laguerre Filters - An Introduction, " Revista do Detua, Vol. 1, No, 3, Jan. 1995.
21. A. V. Oppenheim, D. H. Johnson, and K. Steiglitz, "Computation of spectra with unequal resolution using the Fast Fourier Transform," Proc. of IEEE, vol. 59, pp. 299-301, 1971.
22. W. Schiissler, "Variable digital filters," Arch. Elek. obertragung, vol. 24, pp. 524-525, 1970.
23. A. G. Constantinides, "Spectral transformations for digital filters," Proc. IEEE, vol. 117, no. 8, pp. 1585-1590, August 1970.
24. A. Härmä, M. Karjalainen, L. Savioja, V. Välimäki, U. K. Laine, and J. Huopaniemi, "Frequency-Warped Signal Processing for Audio Applications," J. Audio Eng. Soc., vol. 48, no. 11, pp. 1011-1031, 2000.

25. A. Härmä, “Implementation of frequency-warped recursive Filters,” Signal Processing 80 543-548. 2000.
26. M. Karjalainen, T. Paatero, J. Pakarinen, and V. Välimäki, “Special Digital Filters for Audio Reproduction,” AES 32nd Intern. Conf., Hillerød, Denmark, September 21-23, 2007.
27. V. Välimäki, “Discrete-time modeling of acoustic tubes using fractional delay filters,” Ph.D. dissertation, Lab. Acoust. Audio Signal Process., TKK, Espoo, Finland, Dec. 1995.
28. T.I. Laakso, V. Välimäki, M. Karjalainen, and U.K. Laine, “Splitting the Unit Delay - Tools for Fractional Delay Filter Design,” IEEE Signal Processing Magazine, Vol. 13, No. 1, January 1996.
29. H. Meyr, M. Moeneclaey, S. A. Fechtel, “Digital Communication Receivers”, John Wiley & Sons, Inc., Print ISBN 0-471-50275-8, 1998.
30. J.P. Thiran, “Recursive Digital Filters with Maximally Flat Group Delay, ” IEEE Trans. on Circuits Theory Vol. CT-18, No. 6 Nov. 1971.
31. R. E. Crochiere and L. R. Rabiner, Multirute Digital Signal Processing. Englewood Cliffs, New Jersey: Prentice-Hall, 1983.
32. C. W. Farrow, “A continuously variable digital delay element,” in Proc. IEEE Int. Symp. Circuits Systems, Espoo, Finland, vol. 3, pp. 2641-2645, Jun. 1988.
33. G.S. Liu, C.H. Wei, “A new Variable Fractional Delay Filter with Nonlinear Interpolation,” IEEE Trans. Circuits an System II: Analog and Digital Signal Processing, Vol. 32, No 2, pp. 123–126. Febr. 1992.
34. H. Johansson and P. Löwenborg, “On the Design of Adjustable Fractional Delay Filters,” IEEE Trans. Circuits and Systems-II: Analog and Digital Signal Process., vol. 50, pp. 164-169, April 2003.
35. A. Franck, “Efficient Algorithms and Structures for Fractional Delay Filtering Based on Lagrange Interpolation,” J. Audio Eng. Soc., Vol. 56, No. 12, pp.1036-1056 December 2008.
36. A.Franck, K. Brandenburg, U. Richter, “Efficient Delay Interpolation for Wave Field Synthesis” 125th Audio Eng. Soc (AES) Convention, San Francisco, CA, USA.October 2,5, 2008
37. Wu-Sheng Lu, T. B. Deng, “An Improved Weighted Least-Squares Design for Variable Fractional Delay FIR Filters,” IEEE Trans. On Circuits ans System-II: Analog and DSP, Vol. 46, No. 8, Aug. 1999.
38. T. B. Deng, “Coefficient-Symmetries for Implementing Arbitrary-Order Lagrange-Type Variable Fractional-Delay Filters,” IEEE Trans. Signal Process., vol. 55, pp. 4078-4090 (2007 Aug.).
39. J. Vesma and T. Saramäki, “Optimization and efficient implementation of FIR filters with adjustable fractional delay,” in Proc. IEEE Int. Symp. Circuits Systems, Hong Kong, vol. 4, pp. 2256-2259, Jun. 1997.
40. K. Rajamani, Y.S. Lai, C.W. Farrow, “An Efficient Algorithm for Sample Rate Conversion from CD to DAT,” IEEE Signal Processing Letters, Vol. 7, No. 10, pp.288-290, Oct. 2000.
41. T.A. Ramstad, “Digital Methods for Convertion Between Arbitrary Sampling Frequency,” IEEE Trans. ASSP, Vol. ASSP-32, June 1984.
42. P. J. Kootsookos, and R. C Williamson,. “FIR approximation of fractional sample delay systems,” IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing, Vol. 43, No. 3, pp. 269-271 March 1996.
43. C.K.S. Pun, Y.C. Wu, S.C. Chan, K.L. Ho, “On the design and efficient implementation of the Farrow structure,” IEEE Signal Processing Letters 10(7):189 - 192, August 2003.
44. http://www.acoustics.hut.fi/software/fdtools/.
45. S. Cucchi, F. Desinan, G. Parlatori, G. Sicuranza, “DSP Implementation of Arbitrary Sampling Frequency Conversion for High Quality Sound Application,” Proc. IEEE ICASSP91, Toronto, pp. 3609-3612, May 1991.
46. J. Dattorro, “Effect design, part 2: Delay-line modulation and chorus,” J. Audio Eng. Soc. Vol. 45, No. 10, pp. 764-788, Oct. 1997.
47. M.R. Schroeder, “Digital simulation of sound transmission in reverberant spaces. Part 1,” Journal of Acoustic Soc. of America, Vol.47, No. 2, pp. 424-431, 1970.
48. D. Rocchesso, “Fractionally Addressed Delay Lines,” IEEE Trans. on Speech and Audio Proc., Vol. 8, No. 6, Nov. 2000.

49. A.H. Gray, and J.D. Markel, “Digital lattice and ladder filter synthesis,” IEEE Trans. Audio Electroacoustic., Vol. AU-21, pp.491-500, Dec. 1973.
50. William Grant Gardner, “The Virtual Acoustic Room,” Thesis S.B., Computer Science and Engine, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1982.
51. G. Martinelli, M. Salerno, “Fondamenti di Elettrotecnica,” Voll. 1 e 2, Edizioni Siderea, Roma, II ed. 1995.
52. A.H. Gray, and J.D. Markel, “A normalized digital filter structure,” IEEE Trans. Acoustic Speech and Signal Processing, Vol. ASSP-23, pp.268-277, June 1975.
53. P.A. Regalia, S.M. Mitra, P.P. Vaidaynathan, “The Digital All-Pass Filter: A Versatile Signal Processing Building Block,” Proceedings of IEEE, Vol. 76, No. 1, pp. 19-37, Jan. 1988.
54. M. V. Mathews, “The Technology of Computer Music,” Cambridge, MA: MIT Press, 1969.
55. W. Hartmann, “Digital Waveform Generation by Fractionally Addressing,” J. Acoust. Soc. Am., Vol. 82, pp.1883–1891, 1987.
56. F.R Moore, “Table Lookup Noise for Sinusoidal Digital Oscillator,” Comp. Music Journal, Vol. 1, No. 1, pp. 26-29, 1977.
57. H. G. Alles, “Music synthesis using real time digital techniques,” Proceedings of the IEEE, vol. 68, no. 4, pp. 436–449, April 1980.
58. M. Puckette, “The Theory and Technique of Electronic Music. Hackensack, “ NJ: World Scientific Publishing Co., 2007.
59. T. Stilson, J.O. Smith, “Alias-Free Digital Synthesis of Classic Analog Waveforms,” ICMM Proceeding 1996.
60. J.O. Smith, “Physical Audio Signal Processing,” web published at http://ccrma.stanford.edu/ jos/pasp/, Ed. 2010.
61. P. Cook and P. Scavone, “Synthesis Tool Kit in C++,” http://www-ccrma.stanford. edu/C-CRMA/software/STK/.
62. S. Mitra, K. Hirano, “Digital All-Pass Networks,” IEEE Transactions Circuits and Systems, Volume: 21 Issue: 5 , pp 688-700, Sept. 1974.
63. A. H. Gray, J. D. Markel, “Digital Lattice and Ladder Filter Synthesis,” IEEE Transactions on Audio and Electroacoustics ,Volume: 21, Issue: 6 , pp 491-500, Dec 1973.
64. J. Pekonen, V. Välimäki, . J. Namy, J. O. Smithy and J. S. Abel, “Variable Fractional Delay Filters in Bandlimited Oscillator Algorithms for Music Synthesis, ” International Conference on Green Circuits and Systems (ICGCS), 2010.
65. A. Uncini, “Proprietà strutturali dei circuiti tempo-discreto,” http://www.uncini.com/AudioDigitale, 2005.
66. P.P. Vaidyanathan, S Mitra, Y. Neuvo, “A New Approach to the Realization of Low-Sensitivity IIR Digital Filters,” IEEE Trans. on Acoustics Speech and Signal Processing, Vol. ASSP-34, No.2, pp. 350-361, Apr. 1986.
67. P. Kootsookos and R. C. Williamson, “FIR approximation of fractional sample delay systems,” IEEE Trans. Circuits Syst. -II: Analog and Digital Signal Processing, vol. 43, no. 2, Feb. 1996.
68. Ayush Bhandari and Pina Marziliano, ”Fractional Delay Filters Based on Generalized Cardinal Exponential Splines,” IEEE Signal Processing Letters, Vol. 17, No. 3, March 2010.
69. J.O. Smith, “Introduction to Digital Filters for Audio Applications,”, web published at https://ccrma.stanford.edu/ jos/filters/, online book, Ed. September 2007.
70. J. Stautner and M. Puckette, “Designing multi-channel reverberators,” Comput. Music J., vol. 6, no. 1, pp. 52-65, 1982.
71. V. Välimäki„ J. D. Parker, L. Savioja, J. O. Smith III, and J. S. Abel, “Fifty ¨years of artificial reverberation,” IEEE Trans. Audio, Speech, Language Process., vol. 20, no. 5, pp. 1421-1448, Jul. 2012.
72. S. J. Schlecht and E. A. P. Habets, S “On Lossless Feedback Delay Networks,” IEEE Trans. On Signal Process., vol. 65, no. 6, pp. 1554-1564, March 2017.
73. J. O. Smith and P. Gossett, “A flexible sampling-rate conversion method,” in Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, San Diego, vol. 2, (New York), pp. 19.4.1-19.4.2, IEEE Press, Mar. 1984, expanded tutorial and associated free software available at the Digital Audio Resampling Home Page: http://ccrma.stanford.edu/ jos/resample/.

74. E. Meijering, "A chronology of interpolation: From ancient astronomy to modern signal and image processing," Proceedings of the IEEE, vol. 90, pp. 319-342, Mar. 2002.
75. M. M. J. Yekta, "Equivalence of the Lagrange interpolator for uniformly sampled signals and the scaled binomially windowed shifted sinc function," Digital Signal Processing, vol. 19, pp. 838-842, Sept. 2009.
76. Ç. Candan, " An Efficient Filtering Structure for Lagrange Interpolation," Signal Processing Letters, IEEE, vol. 14, no. 1, pp. 17–19, Jan. 2007
77. J. O. Smith "Bandlimited Interpolation, Fractional Delay Filtering, and Optimal FIR Filter Design", from Lecture Overheads, https://ccrma.stanford.edu/j̃os/Interpolation/Interpolation.html, Center for Computer Research in Music and Acoustics (CCRMA), Stanford University, Jan. 2020.
78. E.W. Weisstein, "Newton's Backward Difference Formula." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/NewtonsBackwardDifferenceFormula.html
79. V. Lehtinen, M. Renfors, "Structures for Interpolation, Decimation, and Nonuniform Sampling Based on Newton's Interpolation Formula," HAL Id: hal-00451769 https://hal.archives-ouvertes.fr/hal-00451769, Submitted on 30 Jan 2010.