A method for the design of Farrow-structure based variable fractional-delay FIR filters

Amir Eghbali a,*, Håkan Johansson a, Tapio Saramäki b

a Division of Electronics Systems, Department of Electrical Engineering, Linköping University, Sweden
b Department of Signal Processing, Tampere University of Technology, Finland

A R T I C L E   I N F O

Article history:
Received 16 March 2012
Received in revised form 3 October 2012
Accepted 12 November 2012
Available online 20 November 2012

Keywords:
Farrow structure
Fractional-delay

A B S T R A C T

This paper proposes a method to design variable fractional-delay (FD) filters using the Farrow structure. In the transfer function of the Farrow structure, different subfilters are weighted by different powers of the FD value. As both the FD value and its powers are smaller than 0.5, our proposed method uses them as diminishing weighting functions. The approximation error, for each subfilter, is then increased in proportion to the power of the FD value. This gives a new distribution for the orders of the Farrow subfilters which has not been utilized before. This paper also includes these diminishing weighting functions in the filter design so as to obtain their optimal values, iteratively. We consider subfilters of both even and odd orders. Examples illustrate our proposed method and comparisons, to various earlier designs, show a reduction of the arithmetic complexity.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

In medical signal processing, digital communications, time/delay estimation, and nonuniform sampling, we may need either to perform sampling rate conversion (SRC) or to delay a digital signal by fractions of the sampling period [1–5]. In software defined radios or multistandard receivers, one may require both of these [5–8]. We can use dedicated blocks for SRC and signal delaying, for each standard. This however requires to either (i) design a large set of filters offline, or (ii) design the filters online.

An efficient way for dynamic SRC and signal delaying is to use the Farrow structure [9] which can approximate allpass functions with an adjustable fractional-delay (FD) of \( \mu(t) \). For SRC, one should delay each input sample by a time-varying \( \mu(t) \) whereas for signal delaying, all input samples are delayed by a time-invariant \( \mu(t) \) [5–8]. For simplicity, the sequel uses \( \mu \) instead of \( \mu(t) \).

The Farrow structure, shown in Fig. 1, is composed of fixed linear-phase finite-length impulse response (FIR) subfilters \( S_k(z) \), \( k = 0, 1, \ldots, L \), of order \( N_k \) as well as the variable multipliers \( \mu \). The transfer function is [9]

\[
H(z, \mu) = \sum_{k=0}^{L} S_k(z)\mu^k, \quad |\mu| \leq 0.5.
\]  

The design parameters are hence the number of subfilters, \( L + 1 \), the order of the subfilters, \( N_k \), and the coefficients of the subfilters, \( S_k(z) \). Then, the overall structure can approximate FD filters with an adjustable \( \mu \) over a frequency range of \( \omega \in [0, \omega_c] \).

1.1. Brief review of previous results

With realizable (nonideal) Farrow subfilters, the resulting FD filters will have an approximation error whose reduction has received considerable attention so far, e.g., [10–24], leading to minimax, least-squares (LS), or maximally flat characteristics. Some of the earlier design methods, e.g., [11,12], iteratively determine \( L \) and \( N_k \) and optimize \( S_k(z) \) so as to satisfy the specifications.
In some cases, e.g., [13], analytical formulas are derived for $S_0(z)$. We can generally allow $S_0(z)$ to have the same or different orders. Furthermore, these orders can be odd or even. If different subfilters have different orders, a main problem is then to find a suitable distribution of $N_k$ so as to satisfy the specifications with a low arithmetic complexity.

### 1.2. Contribution of the paper

This paper proposes a new method to determine $N_k$ using the fact that $|\mu| \leq 0.5$. According to (1), different powers of $\mu$, i.e., $\mu^k$, $k = 0, 1, \ldots, L$, are multiplied by the subfilters $S_k(z)$. With $|\mu| \leq 0.5$ and if $k$ increases, $|\mu|^k$ decreases. If each $S_k(z)$ is realized as the sum of an ideal transfer function and an error frequency response, we can allow larger approximation errors for larger $k$. This reduces $N_k$ and it also gives subfilters with different orders.

Designs with subfilters of different orders have been considered before, e.g., [12,25]. However, this paper introduces new weighting functions between the approximation errors of $S_k(z)$. These weighting functions are a direct consequence of (1) and $|\mu| \leq 0.5$. Our experimental results show that these new weighting functions reduce the arithmetic complexity as compared to the existing design methods. In this paper, these weighting functions are also included in the filter design so as to find their optimal values. We adjust these weighting functions in such a way that the overall arithmetic complexity is reduced. Therefore, our proposed method becomes superior to a wide range of existing design methods. This fact will be illustrated, in Section 6, through various design examples. With $\omega_c$ close to $\pi$, some design methods, e.g., [21,25], are more suitable but they are structurally different from Fig. 1. In other words, the methods in [21,25] reduce the arithmetic complexity by modifying the structure. The method, proposed in this paper, can however be applied to [21,25] as well.

In addition to the main contribution of the paper, discussed above, this paper deals with another issue, namely, the choice of $N_k$ to be odd or even. For example, for the same specification, filters of odd (even) $N_k$ may give a lower arithmetic complexity as compared to filters of even (odd) $N_k$. Therefore, besides the actual optimization problem, the designer should also consider this selection to further reduce the arithmetic complexity.

We will illustrate this fact by providing comparison results (based on our proposed method) for some existing filter specifications where both choices of $N_k$ are considered.

### 1.3. Paper outline

Section 2 gives some prerequisites. Then, Section 3 derives the bounds on the magnitude of the complex error for the Farrow subfilters. The filter design is treated in Section 4 which is followed by some additional discussion in Section 5. The design examples are provided in Section 6 and, finally, Section 7 concludes the paper.

### 2. Prerequisites: Farrow structure

With linear-phase FIR subfilters, the impulse response of $S_k(z)$ is symmetric (antisymmetric) if $k$ is even (odd). Hence, $S_k(z)$ could have any of the four types [26] of linear-phase FIR filters according to Table 1.

#### 2.1. Error frequency responses

For an FD filter, the desired complex and unwrapped phase responses are

$$H_{des}(e^{jo\omega},\mu) = e^{-j\omega + j\mu \omega}.$$  \hspace{1cm} (2)

$$\Phi_{des}(\omega,\mu) = -(\Lambda + \mu)\omega.$$ \hspace{1cm} (3)

Here

$$\Lambda = \frac{\max_k|N_k|}{2}$$ \hspace{1cm} (4)

and $\omega \in [0,\omega_c]$. As

$$H(e^{jo\omega},\mu) = |H(e^{jo\omega},\mu)|e^{j\Phi(\omega,\mu)},$$ \hspace{1cm} (5)

the magnitude and phase errors are

$$H_m^\Lambda(\omega,\mu) = |H(e^{jo\omega},\mu)|-1,$$ \hspace{1cm} (6)

$$H_p^\Lambda(\omega,\mu) = \Phi(\omega,\mu)+(\Lambda + \mu)\omega.$$ \hspace{1cm} (7)

The complex error is defined as

$$H_c^\Lambda(e^{jo\omega},\mu) = H(e^{jo\omega},\mu)-e^{-j\omega + j\mu \omega}.$$ \hspace{1cm} (8)

Further, the phase delay is

$$\tau_{pd}(\omega,\mu) = -\frac{\Phi(\omega,\mu)}{\omega}.$$ \hspace{1cm} (9)

Then, the phase delay error becomes

$$H_d^\Lambda(\omega,\mu) = -\frac{\Phi(\omega,\mu)}{\omega}+(\Lambda + \mu).$$ \hspace{1cm} (10)

#### 2.2. Arithmetic complexity

This section gives the number of fixed multiplications, in $S_k(z)$. Two Farrow structures, having the same $L$, require the same number of variable multiplications.

<table>
<thead>
<tr>
<th>$N_k$</th>
<th>$k$</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Even</td>
<td>Even</td>
<td>I</td>
</tr>
<tr>
<td>Even</td>
<td>Odd</td>
<td>III</td>
</tr>
<tr>
<td>Odd</td>
<td>Even</td>
<td>II</td>
</tr>
<tr>
<td>Odd</td>
<td>Odd</td>
<td>IV</td>
</tr>
</tbody>
</table>
due to $\mu$. As we shall see in Section 6, the FD filters designed by our proposed method, have the same (or lower) values of $L$ when compared to their corresponding references. Therefore, our designed filters require the same (or lower) number of variable multiplications than their corresponding references. Furthermore, the number of additions, in $S_h(z)$, scales proportionally to the number of fixed multiplications, in $S_h(z)$. For these reasons, our complexity comparisons will only consider the number of fixed multiplications, in $S_h(z)$.

With even $N_k$, the subfilters are Type I or III linear-phase FIR filters. Then, $S_0(z)$ is a pure delay and the number of fixed multiplications becomes

$$C_M = -\frac{[L]}{2} + \frac{1}{2} \sum_{k=1}^{L} \left( \frac{N_k}{2} + 1 \right).$$

If $N_k$ is odd, we have subfilters which are Type II or IV linear-phase FIR filters. Consequently,

$$C_M = \frac{1}{2} \sum_{k=0}^{L} N_k + 1.$$

3. Proposed method: error bounds

This section derives the bounds on the magnitude of the complex error for $S_h(z)$.

3.1. Magnitude of complex error: even $N_k$

With even $N_k$, the subfilter $S_0(z)$ is a pure delay and we then aim to design the causal filter

$$H(e^{j\omega_0} \mu) = e^{-j\omega_0 L} + \sum_{k=1}^{L} S_k(e^{j\omega_0}) \mu^k$$

using the nonideal subfilters

$$S_k(e^{j\omega_0}) = D_k(e^{j\omega_0}) + E_k(e^{j\omega_0}).$$

Here, $D_k(e^{j\omega_0})$ and $E_k(e^{j\omega_0})$ are the desired and the error frequency responses, respectively. In (14), the desired frequency responses are $k$th-order differentiators of the form [12]

$$D_k(e^{j\omega_0}) = \frac{(-j\omega_0)^k}{k!}.$$  

Then, (13) becomes

$$H(e^{j\omega_0} \mu) = e^{-j\omega_0 L} + \sum_{k=1}^{L} \frac{(-j\omega_0)^k}{k!} \mu^k + \sum_{k=1}^{L} E_k(e^{j\omega_0}) \mu^k.$$  

Assuming, for some $\epsilon_{\min} > 0$, that

$$|E_k(e^{j\omega_0})\mu^k| \leq \epsilon_{\min},$$

the maximum error in (16), due to the approximation of (15), is

$$\epsilon_{\max} = \sum_{k=0}^{L} \epsilon_{\min} = L\epsilon_{\min}.$$  

Then, (17) and (18) give

$$|E_k(e^{j\omega_0})\mu^k| \leq \frac{\epsilon_{\max}}{L}, \quad k = 1, 2, \ldots, L.$$  

With $|\mu| \leq 0.5$ and if $k$ increases, $|\mu^k|$ decreases and we can thus increase the bound on $|E_k(e^{j\omega_0})|$. Hence, (19) can be written as

$$|E_k(e^{j\omega_0})| \leq \frac{\epsilon_{\max}}{L|\mu|^k}, \quad k = 1, 2, \ldots, L,$$

where we replace $\mu$ by $\omega$ as motivated later in Section 5.1.

3.1.1. Choice of $L$

In the noncausal case, the Taylor series expansion of (2) gives

$$e^{-j\omega_0} = 1 + \sum_{k=1}^{L} \frac{(-j\omega_0)^k}{k!} \mu^k + \sum_{k=L+1}^{\infty} \frac{(-j\omega_0)^k}{k!} \mu^k.$$  

Each term $(-j\omega_0)^k/k!$ is a noncausal $k$th-order differentiator as in (15). Here, (21) requires an infinite number of ideal frequency responses. This is impractical and it is thus necessary to truncate (21). Preserving the first $L+1$ terms of (21), the total approximation error, for the removed terms, is

$$\left| \sum_{k=L+1}^{\infty} \frac{(-j\omega_0)^k}{k!} \mu^k \right| \leq \frac{\left|(-j\omega_0)^{L+1} \mu^{L+1}\right|}{(L+1)!}. $$  

With a desired approximation error $\epsilon_{\max}$, a proper $L$ can be estimated as

$$\min L \text{ subject to } \max_{\omega_{\min}} \left| \frac{(-j\omega_0)^{L+1} \mu^{L+1}}{(L+1)!} \right| < \epsilon_{\max}. $$

Considering

$$|\mu| \leq 0.5, \quad |\omega| \leq \omega_{\min},$$

we can rewrite (23) as

$$\min L \text{ subject to } \left( \frac{0.5\omega_{\min}}{L+1} \right)^{L+1} \frac{\omega_{\min}}{(L+1)!} < \epsilon_{\max}. $$

Note that (23) and (25) give an estimate of the required $L$. Throughout the paper, we will represent this estimated value as $\hat{L}$. With joint optimization of $S_h(z)$, the final value of $L$ may differ by one or two from $\hat{L}$. The proposed filter design algorithm will hence consider the values of $L \in [\hat{L}-2, \ldots, \hat{L}+2]$. Then, we will select the value which gives the lowest arithmetic complexity, i.e., smallest $C_M$.

3.2. Magnitude of complex error: odd $N_k$

With odd $N_k$, all $S_h(z)$ are general filters and $S_0(z)$ determines the lower bound on the approximation error. With nonideal filters, this lower bound is nonzero but independent of $L$ [11]. Using (14), we can write (1) as

$$H(e^{j\omega_0} \mu) = \sum_{k=0}^{L} D_k(e^{j\omega_0}) \mu^k + \sum_{k=0}^{L} E_k(e^{j\omega_0}) \mu^k.$$  

Like (17) and (18), the maximum approximation error of (26) is

$$\epsilon_{\max} = \sum_{k=0}^{L} \epsilon_{\min} = (L+1)\epsilon_{\min}.$$
Then, (17) and (27) give
\[ |E_k(e^{j\omega})| \leq \epsilon_{\text{max}} \frac{L+1}{L+1}, \quad k = 0, 1, \ldots, L. \] (28)

Similar to (20), we can rewrite (28) as
\[ |E_k(e^{j\omega})| \leq \epsilon_{\text{max}} \frac{L+1}{L+1}, \quad k = 0, 1, \ldots, L. \] (29)

For a desired \( \epsilon_{\text{max}} \), the values of \( L \) and \( \epsilon_{\text{min}} \) are related as in (27). The choice of \( \epsilon_{\text{min}} \) limits \( N_0 \) and \( A \). Then, \( L \) is determined and so are the other error bounds. Here, we also use (23) and (25) to determine \( L \). In the filter design, we consider the values of \( L \in \{L-2, \ldots, L+2\} \) and select the one with the smallest \( C_M \).

4. Proposed method: filter design

The proposed filter design problem is iteratively solved in two stages. For each iteration (with a specific \( \alpha \)), one first derives the bounds on \( |E_k(e^{j\omega})| \) according to (20) or (29). For each of these error bounds, the corresponding \( S_k(e^{j\omega}) \) are then designed, to meet the derived error bounds, thereby leading to a specific \( N_k \). This part is a convex design problem and it can be solved through, e.g., the \textit{linprog} algorithm in MATLAB.

After solving these convex design problems, we jointly optimize \( S_k(z) \). This joint optimization problem is generally (except when minimizing \( |H_k^o(e^{j\omega}, \mu)| \)) nonlinear and nonconvex which can be solved using, e.g., the \textit{fminimax} algorithm in MATLAB. Here, the subfilters, obtained from the convex designs, are used as the initial solutions to the joint optimization. If the specifications are met, we exit the filter design procedure. Otherwise, we increase \( \alpha \) and perform a new iteration, composed of a new set of convex and nonconvex design problems.

As discussed earlier, we run this iterative procedure for \( L \in \{L-2, \ldots, L+2\} \). Then, we select the values \( \{L, N_k, S_k(z)\} \) which lead to the smallest \( C_M \). Algorithm 1 summarizes our proposed method.

4.1. Joint optimization problems

For comparison with the existing design methods, this paper considers three joint optimization problems using the error frequency responses of Section 2. One of the considered optimization problems is

\( \min \delta \) subject to \( \|H_k^o(\omega, \mu)\| \leq \delta \). (30)

Here, \( \delta_{\alpha} \) is the desired bound on the phase error. If the phase delay error is of interest, we need another optimization problem as

\( \min \delta \) subject to \( \|H_k^o(\omega, \mu)\| \leq \delta \). (31)

where \( \delta_{\alpha} \) is the desired bound on the phase delay error. In some cases, one may choose to minimize the magnitude of the complex error. Then, the optimization problem becomes

\( \min \delta \) subject to

\[ H_k^o(\omega, \mu) \leq \delta. \]

5. Additional discussion on filter design

This section outlines some issues on the derivations and the filter design.

5.1. \( \mu \) versus \( \alpha \)

In (20) and (29), we have replaced \( \mu \) with \( \alpha \). In the proposed iterative filter design method and during each iteration, \( \alpha \) is set to only one value. The optimal value of \( \alpha \) and, hence, the weighting function can be determined as in Algorithm 1. However, at all iterations, \( \mu \) takes on all values according to (1).

5.2. A note on (20) and (29)

The formulas in (20) and (29) are similar to those of [12] except that the present paper uses a new weighting function, determined by \( \alpha \), between the approximation errors \( |E_k(e^{j\omega})| \). This new weighting function further reduces the arithmetic complexity. Also, the present paper includes this weighting function in the filter design so as to find its optimal value.

5.3. Other error bounds

We can define other error bounds on, e.g., the phase or phase delay. With known bounds on the magnitude of the complex error, we can determine the bounds on the corresponding magnitude and phase errors [12]. Although the exact values of these bounds are different, they will follow a similar trend. Again, this allows us to increase the approximation error, as \( k \) increases. However, the use of other error bounds does not affect the basic filter design procedure, outlined in Algorithm 1.

5.4. Iterative design issues in Algorithm 1

First, we use (23) or (25) to compute \( \hat{L} \) for both even and odd \( N_k \). With \( \hat{L} \) as in (23) or (25), our experimental results indicate that the value \( L < \hat{L} - 2 \) results in a set of \( S_k(z) \) for which the specification is not met. In addition, the value \( L > \hat{L} + 2 \) gives trivial subfilters of orders zero or one which do not affect the overall frequency response and they only increase \( C_M \).

Second, the initial value of \( \alpha \) depends on the specifications and one may hence not always need to initialize as \( \alpha = 0 \). This decreases the design time. A simple way to find a suitable initial \( \alpha \) is by trying some values in the range \( \alpha \in [0, 0.5] \) and checking the corresponding \( \delta \).

Third, for the joint optimization of \( S_k(z) \), the free design parameters are the coefficients of the subfilters \( S_k(z) \) but the values of \( \epsilon_{\text{max}}, \omega_c, L \) and \( \alpha \) are fixed. Fourth, when moving from \( \alpha \) to \( \alpha + \text{step} \), the values of \( N_k \) may sometimes not change. In such cases, we could skip the joint
6. Design examples

The aim of our design examples is to show that our proposed method is applicable to a wide range of specifications while reducing the arithmetic complexity. For comparison, we will design FD filters to meet some existing specifications in the literature, e.g., [12,16,17,19,20,27]. These specifications are summarized in Section 6.1 below.

6.1. Existing filter specifications for comparisons

This section details some specifications, available in the literature.

Spec. 1: \( \omega_c = 0.5\pi \) and \( \epsilon_{\text{max}} = \delta_p = 0.001 \) which require \( N_k = \{7,3,7,3,5\} \) [12].

Spec. 2: \( \omega_c = 0.6\pi \) and \( \epsilon_{\text{max}} = \delta_p = 0.005 \) which need \( N_k = \{7,7,7\} \) [17].

Spec. 3: \( \omega_c = 0.75\pi \), \( \epsilon_{\text{max}} = 0.025 \), and \( \delta_p = 0.005 \) requiring \( N_k = \{9,9,9,9\} \) [17].

Spec. 4: \( \omega_c = 0.9\pi \) and \( |H_p(\omega, \mu)| \leq 0.0042 \) which need \( N_k = \{30,30,12,24,8,12\} \) [12].

Spec. 5: \( \omega_c = 0.9\pi \), \( \epsilon_{\text{max}} = 0.01 \), and \( \delta_d = 0.001 \) which need \( N_k = \{27,9,27,11,19,5\} \) [12].

Spec. 6: \( \omega_c = 0.95\pi \) and \( \epsilon_{\text{max}} \approx 0.0213 \) with even \( N_k \) and 185 fixed multiplications [19].

Spec. 7: \( \omega_c = 0.95\pi \) and \( \epsilon_{\text{max}} \approx 0.0011 \) with even \( N_k \) and 141 fixed multiplications [20].

Spec. 8: \( \omega_c = 0.9\pi \) and \( \epsilon_{\text{max}} \approx 0.0172 \) with even \( N_k \) and 27 fixed multiplications [27].

Spec. 9: \( \omega_c = 0.9\pi \) and \( \epsilon_{\text{max}} \approx 10^{-5} \) with even \( N_k \) and 139 fixed multiplications [27].

Note that Spec. 5 also appears in [16]. Because [12] is superior to [16], we will only compare our proposed method with [12].

6.2. Comparison with previous design methods

Table 2 summarizes the design results to meet the specifications of Section 6.1. As can be seen, our proposed
Our designed filters also have even (odd) \( N_k \) interval. Selecting \( 6.3 \). Odd \( N_k \) versus even \( N_k \)

To approximate an FD filter, the designer can normally select \( N_k \) to be odd or even. In other words, as long as the overall frequency response approximates an FD filter and there are no other restrictions on \( N_k \), either of even or odd \( N_k \) can freely be used. Table 3 outlines some design results (using our proposed method) where we have compared the choices of odd or even \( N_k \) based on their corresponding \( C_M \). The main aim of the comparisons, in Table 3, is thus to illustrate the fact that one could obtain different arithmetic complexities by considering both of even and odd \( N_k \). As our simulations also indicate, for the specifications of Section 6.1, the choice of even \( N_k \) usually gives a smaller \( C_M \). This can be explained by the fact that with even \( N_k \), the subfilter \( S_0(z) \) is a pure delay and it hence does not contribute to \( C_M \). This is not the case with subfilters of odd \( N_k \). The designer should therefore consider both of these choices so as to select the one which gives the lowest arithmetic complexity.

6.4. Minimax versus LS

When comparing the design examples, in Table 2, we sometimes consider different filter design criteria, i.e., LS

**Table 2**

Design parameters and the complexity savings in the specifications of Section 6.1 with respect to the existing references.

<table>
<thead>
<tr>
<th>Spec.</th>
<th>( \omega_c )</th>
<th>( \epsilon_{\max} )</th>
<th>( \delta_p ) or ( \delta_d )</th>
<th>( L )</th>
<th>( x )</th>
<th>( C_M )</th>
<th>( N_k )</th>
<th>Eqns.</th>
<th>Saving (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.5\pi</td>
<td>0.001</td>
<td>( \delta_p = \epsilon_{\max} )</td>
<td>4</td>
<td>0.116</td>
<td>15</td>
<td>(9,5,7,3,1)</td>
<td>(29), (30)</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0.6\pi</td>
<td>0.005</td>
<td>( \delta_p = \epsilon_{\max} )</td>
<td>3</td>
<td>0.159</td>
<td>14</td>
<td>(9,5,7,3)</td>
<td>(29), (30)</td>
<td>13</td>
</tr>
<tr>
<td>3</td>
<td>0.75\pi</td>
<td>0.025</td>
<td>( \delta_p = 0.005 )</td>
<td>4</td>
<td>0.133</td>
<td>15</td>
<td>(11,3,7,3,1)</td>
<td>(29), (30)</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>0.9\pi</td>
<td>0.0042</td>
<td>( \delta_p = 0.001 )</td>
<td>4</td>
<td>0.229</td>
<td>39</td>
<td>(36,36,12,22,4)</td>
<td>(20), (32)</td>
<td>28</td>
</tr>
<tr>
<td>5</td>
<td>0.9\pi</td>
<td>0.01</td>
<td>( \delta_p = 0.001 )</td>
<td>4</td>
<td>0.24</td>
<td>46</td>
<td>(33,11,27,7,9)</td>
<td>(29), (31)</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>0.95\pi</td>
<td>0.0213</td>
<td>( \delta_p = \epsilon_{\max} )</td>
<td>4</td>
<td>0.185</td>
<td>40</td>
<td>(48,48,8,18,2)</td>
<td>(20), (30)</td>
<td>79</td>
</tr>
<tr>
<td>7</td>
<td>0.95\pi</td>
<td>0.0011</td>
<td>( \delta_p = \epsilon_{\max} )</td>
<td>5</td>
<td>0.22</td>
<td>105</td>
<td>(88,88,30,58,12,18)</td>
<td>(20), (30)</td>
<td>26</td>
</tr>
<tr>
<td>8</td>
<td>0.9\pi</td>
<td>0.0172</td>
<td>( \delta_p = 0.001 )</td>
<td>4</td>
<td>0.174</td>
<td>25</td>
<td>(26,26,8,10,2)</td>
<td>(20), (32)</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>0.9\pi</td>
<td>( 10^{-5} )</td>
<td>( \delta_p = \epsilon_{\max} )</td>
<td>7</td>
<td>0.224</td>
<td>138</td>
<td>(74,74,42,60,30,38,14,12)</td>
<td>(20), (32)</td>
<td>1</td>
</tr>
</tbody>
</table>

**Fig. 2.** The error frequency responses for an FD filter designed to meet Spec. 2 in Table 2.
versus minimax. Therefore, the comparisons may be delicate. The main aim, of those comparisons, is to show the applicability of our proposed method to various specifications. Further, we have slightly overdesigned the filters to obtain smaller ripples than their corresponding LS designs, see Fig. 3.

7. Conclusion

A method to design FD filters, using the Farrow structure, was introduced. The properties of $\mu$ and its powers are utilized and the approximation error for different subfilters is increased proportionally to the power of $\mu$. Subfilters with different orders are obtained and examples illustrate our proposed method. In comparison with some existing designs, the arithmetic complexity, in terms of the number of fixed multiplications, is reduced.

Table 3
Design parameters and the complexities in some of the specifications of Section 6.1 where we compare the choices of even and odd $N_k$ using our proposed method.

<table>
<thead>
<tr>
<th>Spec.</th>
<th>$C_m$ with odd $N_k$</th>
<th>$C_m$ with even $N_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>14</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>52</td>
<td>39</td>
</tr>
<tr>
<td>7</td>
<td>133</td>
<td>105</td>
</tr>
</tbody>
</table>

Fig. 3. The error frequency responses for an FD filter designed to meet Spec. 9 in Table 2.

References


