Hindawi Publishing Corporation
VLSI Design
Volume 2014, Article ID 217495, 14 pages
http://dx.doi.org/10.1155/2014/217495

Research Article

Design of Finite Word Length Linear-Phase FIR Filters in the Logarithmic Number System Domain

Syed Asad Alam and Oscar Gustafsson

Division of Electronics Systems, Department of Electrical Engineering, Linköping University, 581 83 Linköping, Sweden

Correspondence should be addressed to Oscar Gustafsson; oscarg@isy.liu.se

Received 4 November 2013; Accepted 20 February 2014; Published 9 April 2014

Academic Editor: Antonio G. M. Strollo

Logarithmic number system (LNS) is an attractive alternative to realize finite-length impulse response filters because of multiplication in the linear domain being only addition in the logarithmic domain. In the literature, linear coefficients are directly replaced by the logarithmic equivalent. In this paper, an approach to directly optimize the finite word length coefficients in the LNS domain is proposed. This branch and bound algorithm is implemented based on LNS integers and several different branching strategies are proposed and evaluated. Optimal coefficients in the minimax sense are obtained and compared with the traditional finite word length representation in the linear domain as well as using rounding. Results show that the proposed method naturally provides smaller approximation error compared to rounding. Furthermore, they provide insights into finite word length properties of FIR filters coefficients in the LNS domain and show that LNS FIR filters typically provide a better approximation error compared to a standard FIR filter.

1. Introduction

Finite-length impulse response (FIR) filters constitute a class of digital filters commonly used for their stability properties and the ability to obtain a linear phase response. The transfer function of an Nth-order FIR filter is

$$H(z) = \sum_{n=0}^{N} h_n z^{-n}, \tag{1}$$

where h_n are the impulse response coefficients.

The filter order and, therefore, the number of multiplications and additions for a straightforward realization grows approximately inversely proportional to the transition bandwidth of the magnitude response [1, 2]. As multiplications traditionally have a larger area complexity and power consumption compared to additions, much work has focused on reducing the number of multiplications in FIR filter realizations by using sparse filters or frequency response masking filters [3–5]. Work has also been done to reduce the complexity of each multiplication, for example, by introducing filter coefficients easily realizable using shifts, additions, and subtractions, sometimes referred to as multiplierless realizations [6–8].

Furthermore, the representation of data and coefficients affects both the switching activity and implementation complexity which, in turn, affects power consumption as well. Commonly, a fixed-point two's complement number representation is used to represent data in DSP systems, but other number representations have also been investigated as an efficient way of data representation for such systems [9–11]. Among them is the logarithmic number system (LNS) [12], which over the past few decades has been studied as an alternative to fixed-point number systems. The main motivation for doing so is the inherent simplification of basic arithmetic operations as multiplication, division, roots, and powers, due to its properties, which are reduced to addition, subtraction, multiplications, and divisions, respectively. They also have interesting numerical properties as higher dynamic range compared to fixed-point representations for a given number of bits [13] and better round-off noise performance than floating-point arithmetic for a given number of bits [14–16].

With LNS reducing multiplication to an addition, it finds application in low power signal processing systems, including digital filters [13, 17–23]. In [18, 19] LNS has also been proposed to reduce power dissipation in hearing aid devices [24], subband coding [25], and video processing [26]. It has been shown that LNS requires a reduced data word length to achieve the same SNR when compared to fixed-point representations [21, 23], with a reported power saving of nearly 60% in some cases. However, no discussion about the coefficient word length was included nor how to obtain finite word length coefficients.

Most efforts towards utilizing LNS for digital have focused on either implementing the nonlinear conversion to and from LNS, selecting the logarithm basis, or implementing the LNS addition and subtraction efficiently [12, 23, 27–30]. The finite word length filter design has to the best of the authors' knowledge not been considered but instead relied on rounding the obtained coefficients to the nearest LNS number.

In this work, an integer linear programming (ILP) approach to design optimal finite word length linear-phase FIR filters in the LNS domain is proposed. Here, instead of optimizing filters in the linear domain and converting them into LNS with rounding in the LNS, we optimize the filter directly in the LNS domain with finite word length constraints in the LNS domain. By optimizing the filters directly in the LNS domain it is also possible to compare the required word lengths to obtain further insights regarding the efficiency of LNS.

The rest of the paper is outlined as follows. In the next section LNS numbers are reviewed. In Section 3 ILP is reviewed and the issues involved by performing it in the LNS domain are discussed. This section also includes the formulation and the proposed variable selection and branching direction strategies. Finally, Sections 4 and 5 present the results and conclusions, respectively.

2. The Logarithmic Number System (LNS)

The LNS takes advantage of the fact that multiplications become additions, however, at the cost of increased complexity to implement addition. The LNS representation of a number X consists of a triplet $\mathcal{X}$ as follows [12]:

$$\mathcal{X} = \left(z_x, s_x, m_x\right), \tag{2}$$

where z_x is a one-bit flag to indicate if X is zero, s_x is the sign of X, and $m_x = \log_b|X|$ is the base-b logarithm of the absolute value of X [21]. Representation capabilities, computational complexity, and conversion to and from LNS numbers greatly depend on the choice of the base [10].

As stated before, the motivation behind using LNS is the simplicity of implementing the multiplication of $\mathcal{X}$ and $\mathcal{Y}$ which is reduced to the computation of the triplet $\mathcal{Z}$ [21]:

$$\mathcal{Z} = \left(z_z, s_z, m_z\right), \tag{3}$$

where $z_z = z_x$ or z_y; that is, the zero flag of the output, $s_z = s_x \text{xor } s_y$; that is, the sign of the output and the output itself, $m_z = m_x + m_y$.

The addition and subtraction are more complex and are given by (4)

$$\begin{aligned} \text{add} &= \max\left(m_x, m_y\right) + \log_b\left(1 + b^{-|m_x - m_y|}\right), \\ \text{sub} &= \max\left(m_x, m_y\right) + \log_b\left(1 - b^{-|m_x - m_y|}\right). \end{aligned} \tag{4}$$

In case of FIR filters, the filter coefficients are typically between −1 and 1. Representing the coefficients in LNS means that all the exponents in the LNS domain will be negative. This means that the larger the magnitude in LNS domain, the smaller it is in the linear domain. As the coefficients become smaller, the LNS number magnitude becomes larger and reaches the maximum number representable using i integer and f fractional bits, given by $\pm 2^{-(2^i - 2^{-f})}$. Important to note is that since the LNS numbers will always be negative for (most) FIR filter coefficients, there is no need to use a sign bit to represent the exponent in this case.

Any number smaller than this is not representable. This phenomenon is shown on a linear scale in Figure 1 where two scales are shown for one and two integer bits, respectively. As expected, the logarithmic numbers are unevenly spaced. However, there is a gap between the smallest nonzero number and zero which is much larger than the distance between the smallest and second smallest nonzero numbers. Any number that lies within this space is rounded to either zero or the smallest number. Hence, the number of integer bits will have an impact on the smallest number that can be represented.

2.1. Finite Word Length Effects. It is well established that coefficient quantization results in a static deviation of the transfer function, while data quantization results in round-off noise [1, 2]. In the linear domain the quantization happens after the multiplications, as the fractional word length is increased there. However, in the LNS domain, the fractional word length is not increased after the multiplications, as this corresponds to an addition. Instead, the quantization occurs in the addition, because addition, as shown in (4), requires look-up tables and cannot be represented exactly [15]. In the linear domain, the multiplier complexity grows linearly with the coefficient word length, while the adder complexity is independent of the coefficient word length. In the LNS domain, the multiplier complexity is dependent on the smallest of the data and coefficient word lengths, while the adder complexity is dependent on the largest of those.

3. Proposed Integer Linear Programming Design in the LNS Domain

3.1. Integer Linear Programming. Integer linear programming (ILP) is a class of linear programming where some or all the variables are restricted to have integer values [31]. Generally,

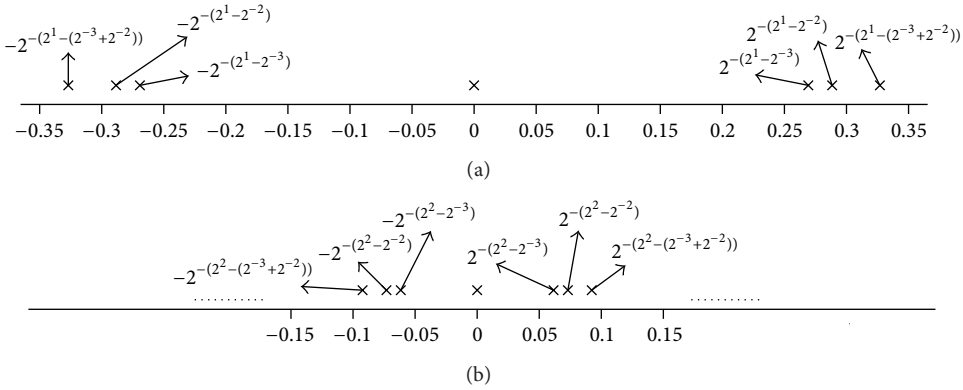

Figure 1: Distribution of smallest logarithmic values using base 2: (a) one integer bit and three fractional bits and (b) two integer bits and three fractional bits.

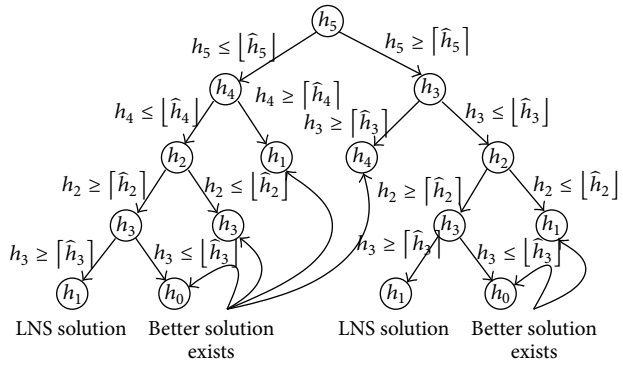

Figure 2: Example branch and bound tree for LNS FIR filter design.

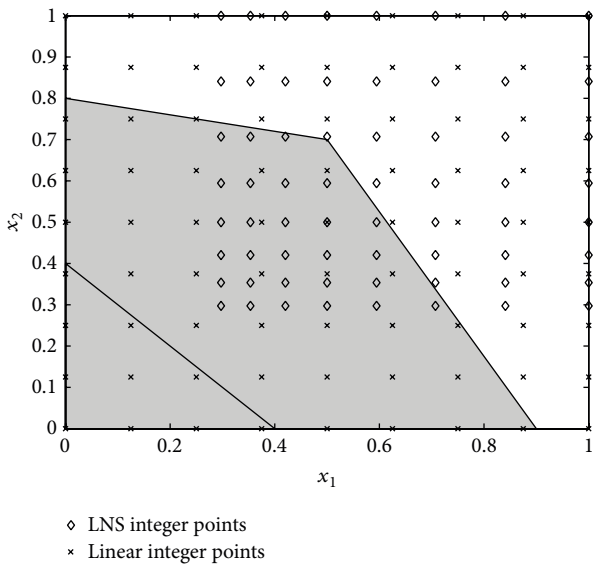

Figure 3: Example solution space for ILP with LNS (◇) and linear (×) integers with three fractional bits with valid region in gray and the line representing the pareto front.

the standard form to express a linear programming problem is given by

$$\begin{aligned} &\text{maximize} \quad c^T x \\ &\text{subject to} \quad Ax \le b \\ &\qquad\qquad x \ge 0 \\ &\qquad\qquad x, c \in \mathbb{R}^n,\ b \in \mathbb{R}^m,\ A \in \mathbb{R}^{n \times m}, \end{aligned} \tag{5}$$

where n is the number of variables and m is the number of constraints.

However, some real world problems require the result to be integers. This requirement is incorporated into the programming model by having an additional constraint $x \in \mathbb{Z}^n$. There are two important algorithms for solving ILPs: branch and bound (BB) and cutting plane. (Naturally, for FIR filters the "integers" are fixed-point values with a fractional part. However, to keep the standard nomenclature we select to denote any fixed-point value with integer.)

In this work we use the branch and bound (BB) algorithm which is a general algorithm for finding optimal solutions to integer or combinatorial problems. It consists of enumerating all candidate solutions and discarding candidates by using upper and lower bounds of the quantity being optimized.

ILP problems are nondeterministic polynomial (NP) hard. In order to solve them the integer constraint is relaxed and the variables are allowed to take noninteger values. This, and subsequent, LP-relaxations provide a *bound* on the best possible solution without the integer constraint. One noninteger variable is selected and *branched* on by forming two subproblems. One where the variable is restricted to be at least as large as the next higher integer value and one where it is restricted to be at most as large as the next lower integer value. This is illustrated in Figure 2, where the top node represents the LP-relaxation, variable h_5 is selected to branch on, and two subproblems are formed by adding the constraint on the edges. This procedure continues until any of the following happens.

(1) The subproblem is infeasible, as any possible further branching will not change that.

(2) The subproblem results in a solution where all coefficients are integers.

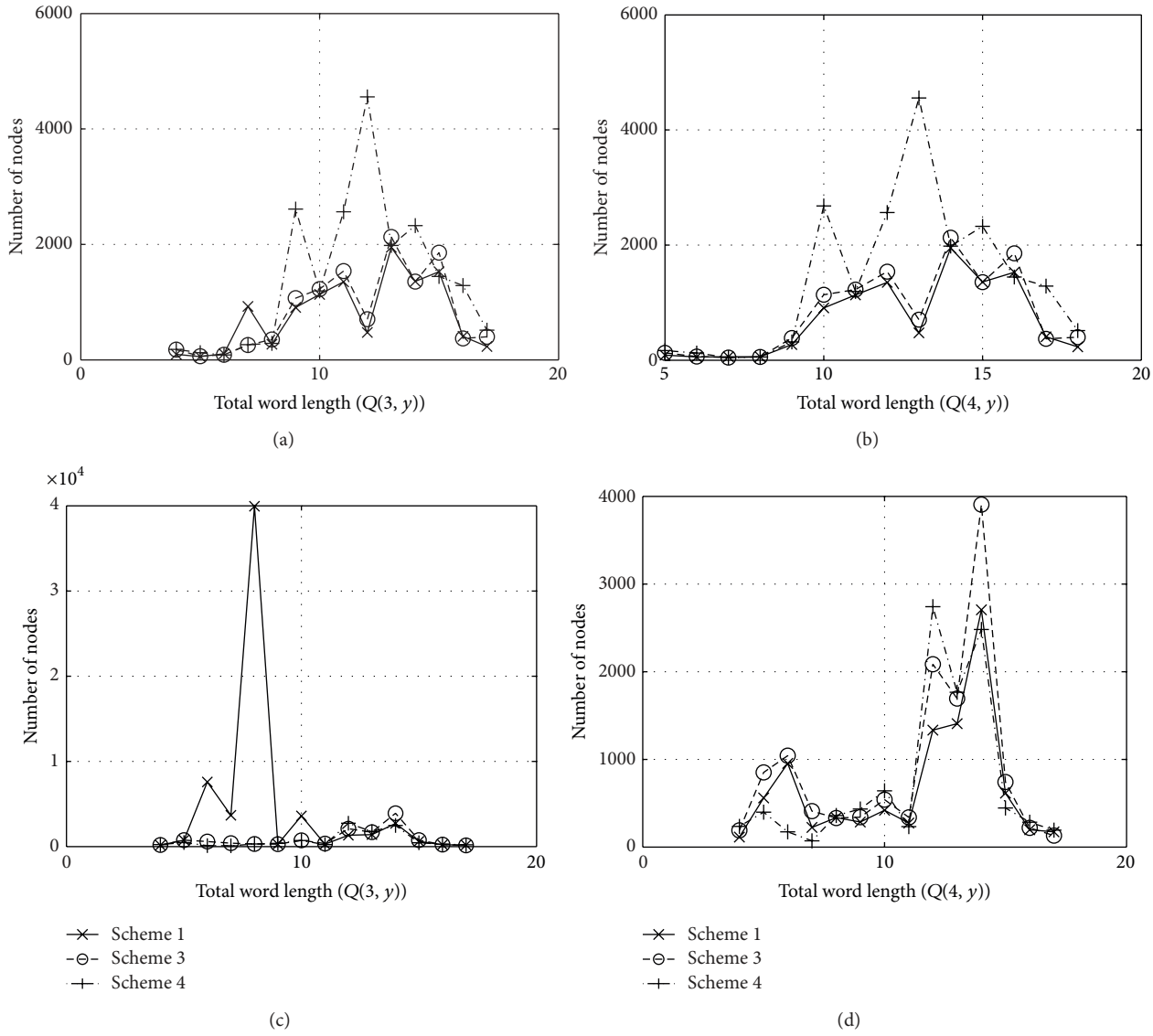

FIGURE 4: Number of visited nodes (solved subproblems) for specification 6: (a) $N = 20$, $Q(3, y)$, (b) $N = 20$, $Q(4, y)$, (c) $N = 34$, $Q(3, y)$, and (d) $N = 34$, $Q(4, y)$.

(3) The solution of the subproblem is worse than the best obtained integer solution, and, hence, no further subproblems generated from that node can possibly have better values.

In any of these cases the algorithm returns to the previous node and selects another subproblem to solve until all possibilities are exhausted and the optimal integer solution is obtained.

The nontrivial challenge in ILP is to know which variable to branch on and in which order. We will suggest and evaluate a number of branching schemes in later sections.

3.2. Linear Programming Design of FIR Filters. Linear-phase FIR filters can be designed to be linear and are a good candidate for linear programming optimization. The work by Rabiner [32] is one of the earliest papers using this technique to solve digital FIR filter problems. However, this technique has its limitations when finite word length restriction is imposed [33].

In ILP, the FIR optimization problem is the same as in the linear programming case. A general FIR optimization problem can be stated as the following minimization problem [1]:

$$\begin{aligned} &\text{minimize} \quad \delta \\ &\text{subject to} \quad -\delta \le E\left(\omega_i T\right) \le \delta, \quad i = 1, 2, \ldots, K, \end{aligned} \tag{6}$$

where δ is the approximation error, K is the number of frequency points, and

$$E\left(\omega T\right) = W\left(\omega T\right)\left[H_R\left(\omega T\right) - D\left(\omega T\right)\right], \quad \omega T \in \Omega, \tag{7}$$

where Ω is the union of the passband and stopband regions and is a dense set of frequency samples taken from

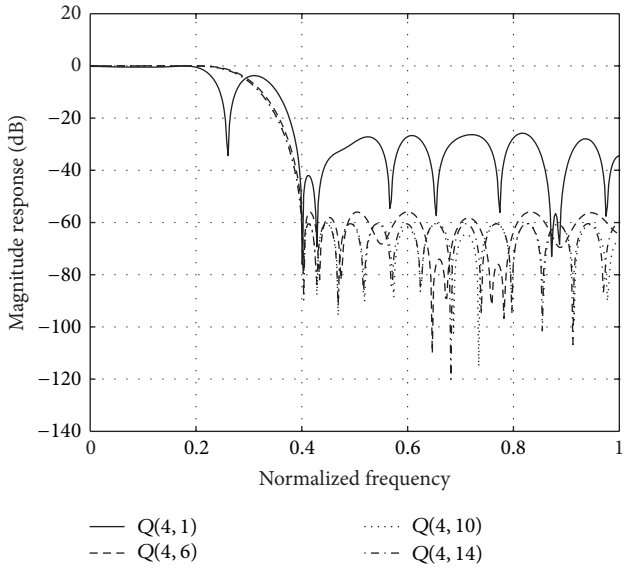

FIGURE 5: Magnitude responses for different fractional word length of spec. 3, $N = 34$.

the passbands and stopbands, including the band edges. $D(\omega T)$ is the desired function to be approximated by $H_R(\omega T)$. For a standard low-pass filter, $D(\omega T)$ is given by [1]

$$D(\omega T) = \begin{cases} 1, & \omega T \in [0, \omega_c T] \\ 0, & \omega T \in [\omega_s T, \pi]. \end{cases} \tag{8}$$

The weighting function $W(\omega T)$ specifies the cost of the deviation from the desired function. This basically means that by using this weighting function, one can obtain different deviations in ripples in different frequency bands. The larger the weighting function, the smaller the relative ripple. For a standard low-pass filter, the weighting function can be given as [1]

$$W(\omega T) = \begin{cases} 1, & \omega T \in [0, \omega_c T] \\ \dfrac{\delta_c}{\delta_s}, & \omega T \in [\omega_s T, \pi]. \end{cases} \tag{9}$$

$H_R(\omega T)$ is the real zero-phase frequency response, which is related to the frequency response as [1]

$$H\left(e^{j\omega T}\right) = e^{j\Theta(\omega T)} H_R(\omega T). \tag{10}$$

The magnitude response of $H(e^{j\omega T})$ is equal to the magnitude of $H_R(\omega T)$. However, $H_R(\omega T)$ can take on negative values where $|H(e^{j\omega T})|$ is a nonnegative function. The resulting filter will be optimized in the Chebyshev or minimax sense. Such filters are also called equiripple filters because all the ripples will be of equal size, subject to the weighting function [1].

The linear relaxation obtained without integer constraints will be the same minimax solution obtained using other FIR filter design techniques, such as the Remez exchange algorithm. Therefore, in later results the linear relaxation is used as a lower bound. Introducing integer constraints will constrain the solution space and increase the approximation error. Furthermore, the resulting filters may no longer be equiripple.

3.3. ILP Design of FIR Filters in the LNS Domain. In the LNS domain, our proposed design method focuses on finding the LNS equivalent of the coefficient values and implementing a branch and bound tree based on LNS integer values. In the branch and bound tree, the coefficients are, when picked as the branching variable, constrained to be LNS integers.

The solution space for an LNS ILP problem is different from that of an ILP problem using linear representation. An example of this is shown in Figure 3, clearly showing the difference between the two spaces.

In the LNS branch and bound tree, after solving a node, some of the coefficients will be LNS integers and some not. The linear relaxation of a noninteger coefficient, if selected, would be rounded up and down.

The first proposed branch variable selection scheme, named as Scheme 1, is to scan all the noninteger coefficients and pick the noninteger variable which has the largest absolute value. This may be efficient since this will introduce the largest quantization error in the linear domain, and, hence, possibly reduce the convergence time. Another scheme, denoted as Scheme 2, instead selects the coefficient, after scanning all noninteger coefficients, which is farthest from being an integer in the LNS domain. This is again based on the idea of introducing large quantization errors early.

For both schemes described above, both combinations of branching direction are evaluated. This means that for both schemes, the difference is calculated between the linear relaxation of the selected variable and its upper and lower bounds. In one implementation, the branching variable is constrained to the bound which gives the least difference first and then upon returning to that node constraining it to the other bound. In the other implementation, this selection is reversed.

A third scheme is proposed, denoted as Scheme 3, where the branching direction was limited to the bound with the least difference. In this scheme, all noninteger coefficients are scanned and the minimum distance to each of its bounds for each variable is calculated. The coefficient having the maximum of these minimum distances is selected as the branching variable.

All the above schemes either picked up the floor or ceiling value of the linear relaxation for the next branch without actually knowing whether that would produce a lower approximation error. The final proposed approach, Scheme 4, uses the same variable selection scheme as Scheme 3 but both subproblems are evaluated before the selection is done on which path to continue along. The decision is based on the most promising path, that is, the one with the smallest approximation error.

All these branch variable selection schemes are for locating the best variable to branch on. Apart from these schemes, the rest of the algorithm is the same for all cases. However, an error threshold is allowed while testing each coefficient for

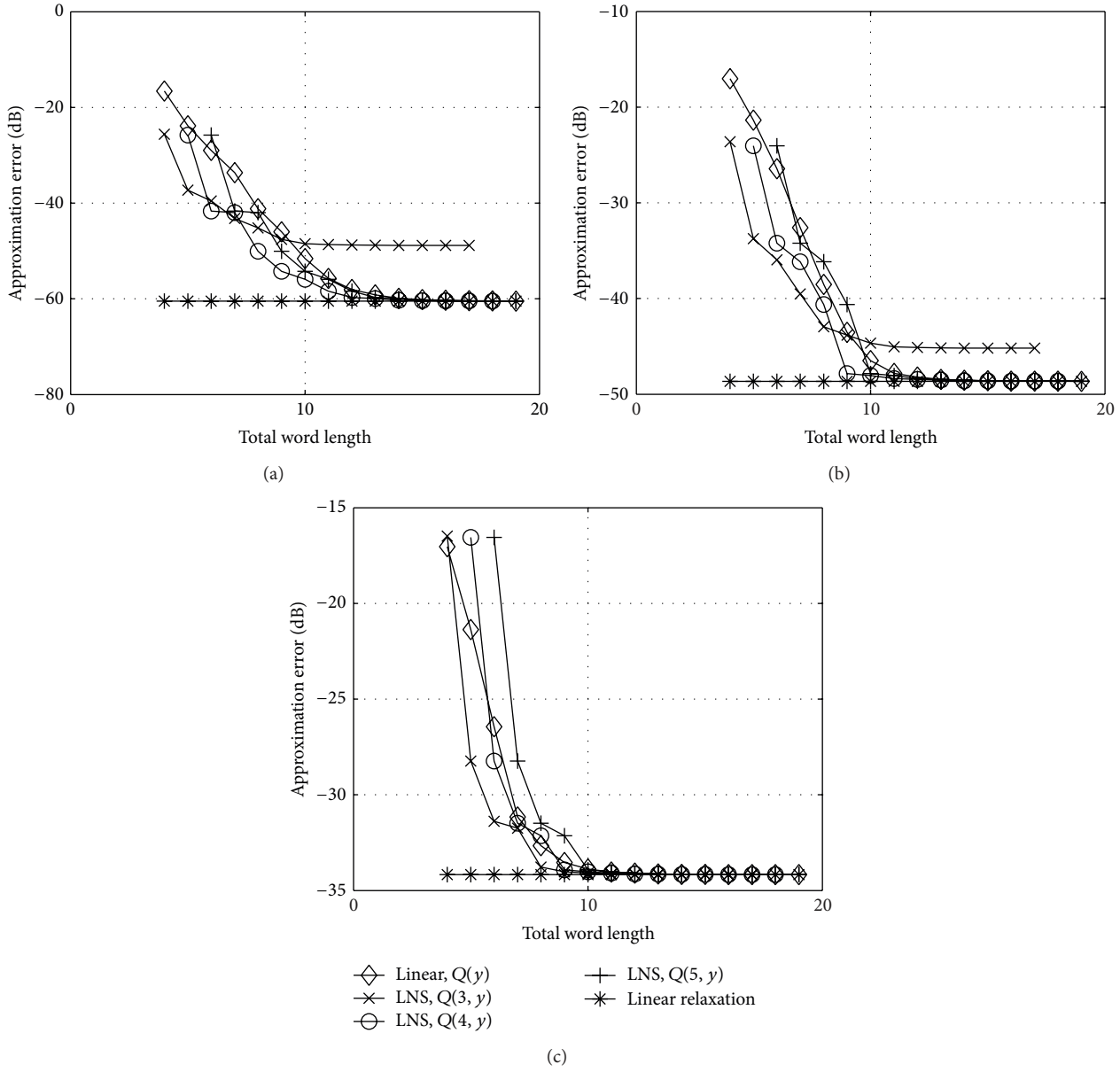

Figure 6: Effect of changing integer and fractional word length on approximation error: (a) spec. 3, $N = 34$, (b) spec. 6, $N = 34$, and (c) spec. 6, $N = 22$.

being an LNS integer, which means that the coefficient need not be exactly an LNS integer. Any number within a defined threshold of the corresponding LNS integer is taken as an LNS integer. This is a standard procedure in ILP solving as numerical errors in the computations may lead to noninteger values in the optimal solution, even though they in practice should be considered integers [31].

4. Results

For the results a number of low-pass filter specifications were devised arbitrarily to cover both narrow-band and wide-band filters. The passband and stopband edges are shown in Table 1 and the weighting function $W(\omega T)$ is 1 for both passband and stopband. Unless otherwise stated the logarithm base is selected as $b = 2$. Regarding word length, $Q(x, y)$ indicates the number of integer bits, x and fractional bits, y is used for the LNS integer, and $Q(y)$ denotes y fractional bits for the linear integers.

4.1. Comparison of Branching Schemes. A number of optimization runs were carried out with different alternatives. The purpose was to see the efficiency of each scheme and see the impact of word length on different filters.

Table 2 shows the number of nodes visited (subproblems solved) for the first two schemes with different branching directions. "L" denotes that the branching direction is determined on the closest integer, while "H" then denotes

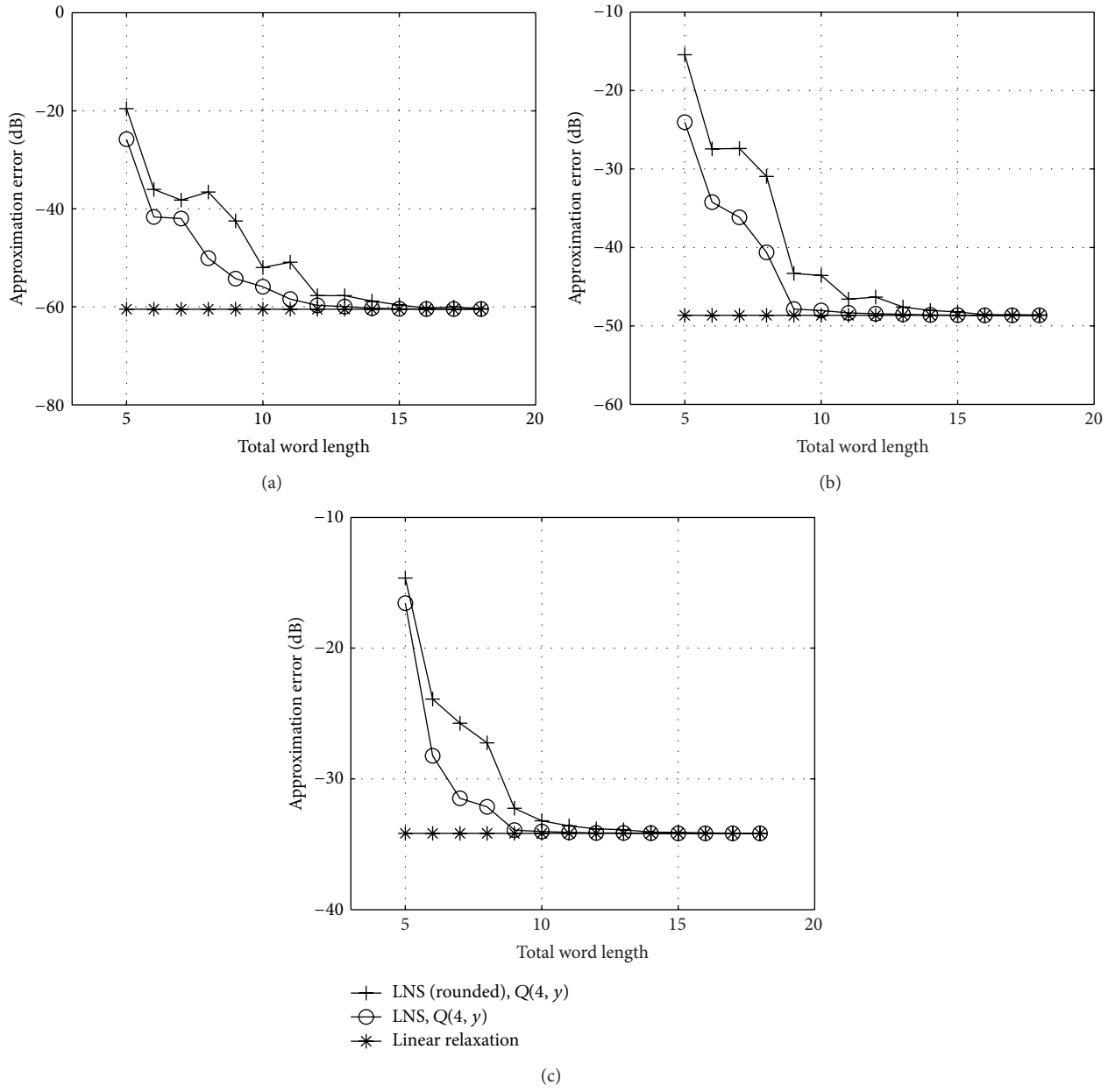

FIGURE 7: Approximation error as a function of coefficient word length for proposed method and direct rounding of the linear relaxation: (a) spec. 3, $N = 34$, (b) spec. 6, $N = 34$, and (c) spec. 6, $N = 22$.

TABLE 1: Filter specifications considered.

Filter spec.	Spec. 1	Spec. 2	Spec. 3	Spec. 4	Spec. 5	Spec. 6
$\omega_c T$, rad	0.2π	0.2π	0.2π	0.2π	0.1π	0.75π
$\omega_s T$, rad	0.3π	0.35π	0.4π	0.5π	0.25π	0.9π

the integer with the largest difference from the relaxed value. The number of integer and fractional bits is four and six, respectively, that is, $Q(4, 6)$. Table 2 establishes that branching on the direction based on the least difference gives the best result. Also, Scheme 1 proves superior to Scheme 2.

Scheme 1 is further compared with Schemes 3 and 4 and the results are shown in Table 3, again with $Q(4, 6)$. This table shows them to be comparable, apart from the case when the filter length is 64. This shows that Scheme 1 does not converge quickly when filter length is increased. This behavior is

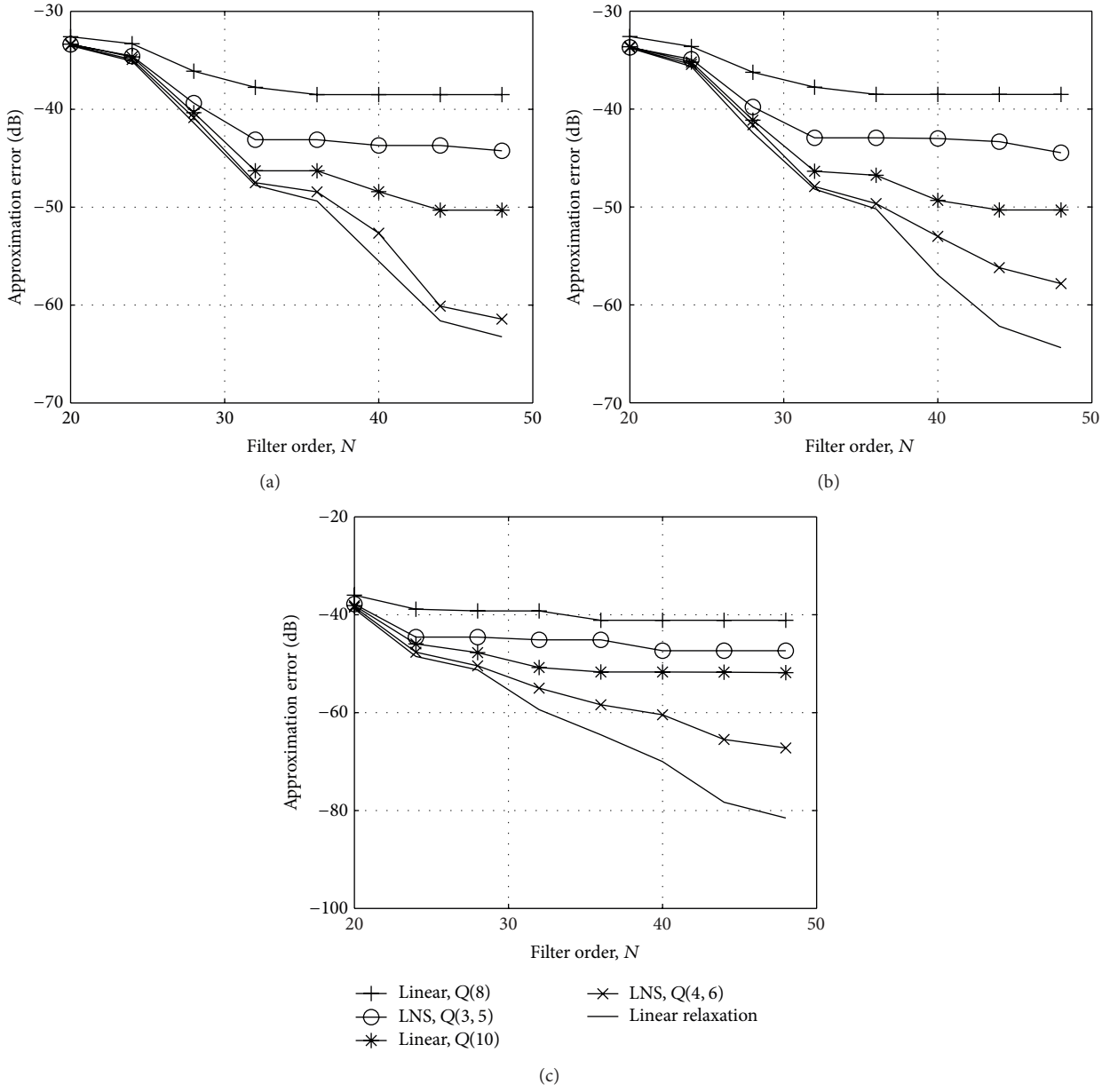

Figure 8: Approximation error as a function of filter length: (a) spec. 5, (b) spec. 6, and (c) spec. 3.

further observed in numerous optimization runs. For various integer and fractional bits and filter lengths, Scheme 3 gives, on average, the best convergence time. Scheme 1 appears to perform better when given a narrow transition band and small filter length.

The effect of word length is further studied and results are shown in Figure 4. In Figures 4(a) and 4(b), the number of nodes required to solve spec. 6, $N = 20$, for three and four integer bits, respectively, while the number of fractional bits varying from 5 to 14 is shown. Similarly, Figures 4(c) and 4(d) show the same for spec. 6, $N = 34$. When the filter length is 20, Scheme 1 converges in the least amount of time. Decreasing integer word length to three did not alter the efficiency of this scheme. However, as the filter length was increased while keeping the passband and stopband edges the same, one can see that decreasing the integer word length from four to three had a major impact on the convergence time of Scheme 1. Further optimization runs show the same effect. Table 4 shows, for a few cases, the number of nodes needed by each of these three schemes.

The figures and tables clearly show that if either (a) the transition bandwidth is increased or (b) the filter length is increased, Scheme 1 converges very slowly. This effect is magnified when the integer word length is decreased. This can be seen in the table in the case of spec. 4 with $N = 20$, where, in relation to spec. 6, $N = 20$, the filter length is kept the same but the transition bandwidth is increased. The same effect is seen for narrow band filters as well. Based on these

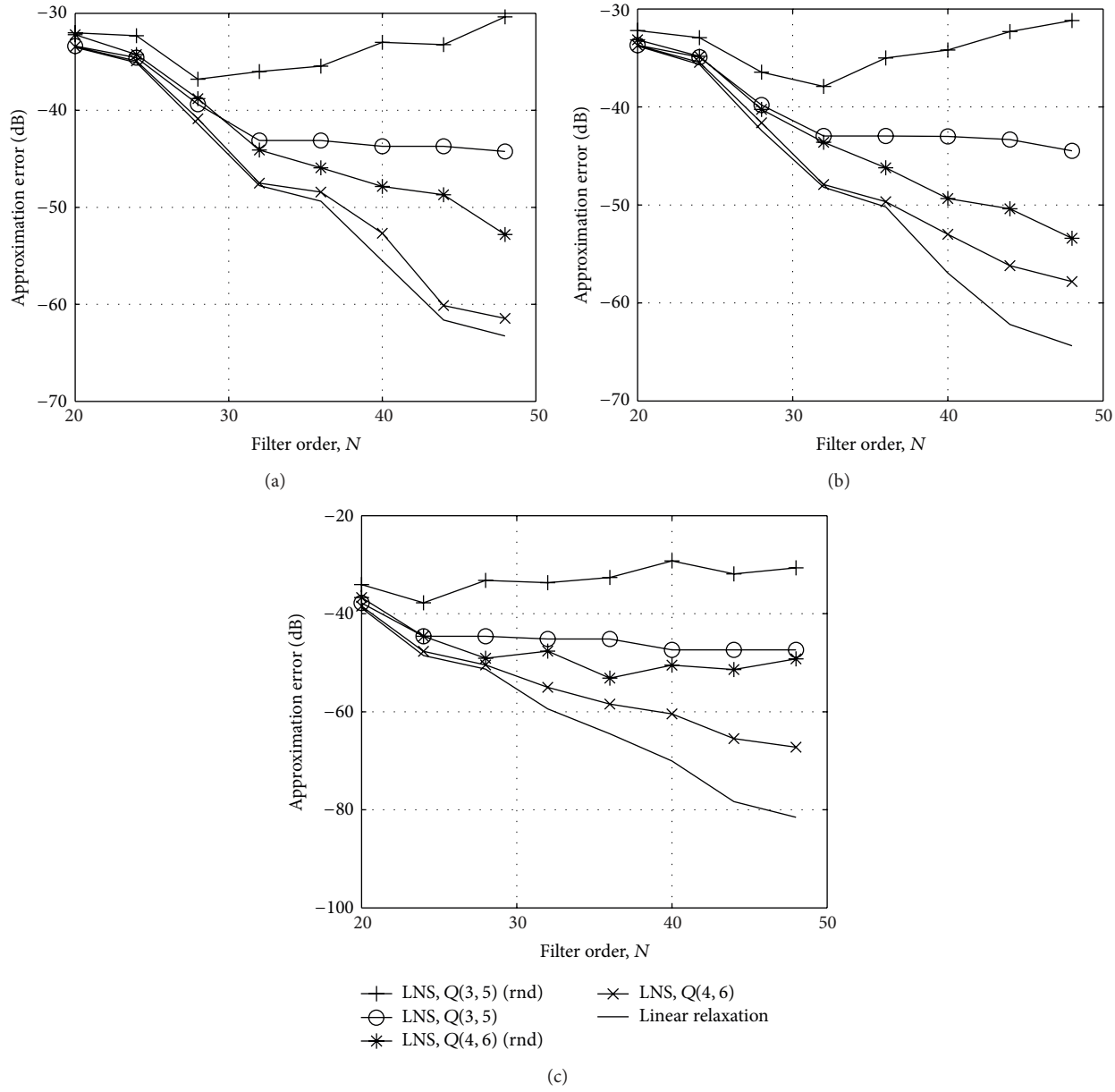

Figure 9: Approximation error as a function of filter length for proposed method and direct rounding of the linear relaxation: (a) spec. 5, (b) spec. 6, and (c) spec. 3.

observations, one can conclude that either Scheme 3 or 4 is on average the best alternative to solve the given optimization problem.

4.2. Effect of Word Length. Next, the effect of changing the integer and fractional word length on the magnitude response and approximation error was studied. Optimization runs were carried out for a number of specifications given in Table 1. For various optimization runs, either the integer word length was fixed and fractional word length varied or vice-versa. In Figure 5, the magnitude responses for filters with specification 3 are shown. The integer word length has been fixed at four while the fractional word length takes on values of one, six, ten, and fourteen.

To study the effect of changing integer and fractional word length on the approximation error in more detail for specification 3, the approximation error is plotted in Figure 6. In these figures, there are five curves, one showing the optimal approximation error for continuous coefficients (linear relaxation) and one shows the approximation error for linear integers, while the other three curves are for LNS integer word length three, four, and five, respectively, with total word length changing from four to 20 for each curve. All these figures give a detailed picture of the impact of changing integer and fractional word length (the largest coefficient value, and, hence, the number of most significant bits needed for the coefficients in the linear domain is also a function of filter specification. For a narrow-band low-pass filter, one

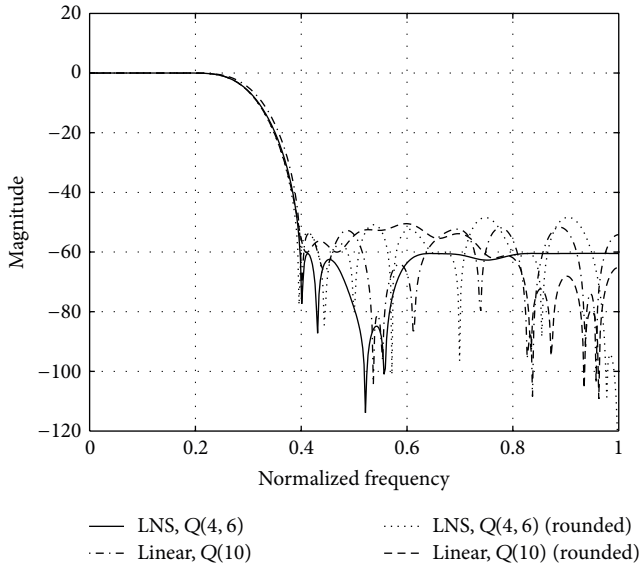

Figure 10: Magnitude response of finite word length LNS and linear coefficients for spec. 3, $N = 40$.

Table 2: Comparison of first two variable selection schemes.

(a)

Scheme	Branching direction	Number of nodes		
		Spec. 1 $N = 64$	Spec. 2 $N = 42$	Spec. 3 $N = 34$
1	L	30704	1410	948
	H	11113942	238320	20494
2	L	82336	5990	1088
	H	3887918	458974	12342

(b)

Scheme	Branching direction	Number of nodes		
		Spec. 4 $N = 20$	Spec. 5 $N = 34$	Spec. 6 $N = 26$
1	L	378	1334	210
	H	4256	38650	13352
2	L	365	1940	356
	H	3710	37248	17190

would require less number of bits than a wide-band filter. This can be realized by considering Mth band (or Nyquist) filters where a filter with band with π/M rad will have a maximum coefficient value of $1/M$. However, for simplicity, this aspect is ignored in the comparisons in this paper).

As shown in Figure 6, the LNS filters converge to optimum with both four and five integer bits. However, if the optimal approximation error is small, approximately less than 40 dB, filters with three integer bits also converge. Clearly, the number of fractional bits also has a large impact on the resulting approximation error. It is also shown that the linear domain finite word length filters are always beaten by an LNS filter.

Table 3: Comparison of variable selection schemes 1, 3, and 4 with $Q(4, 6)$.

(a)

Scheme	Number of nodes		
	Spec. 1 $N = 64$	Spec. 2 $N = 42$	Spec. 3 $N = 34$
1	30704	1410	948
3	8986	4814	1010
4	10240	1618	1092

(b)

Scheme	Number of nodes		
	Spec. 4 $N = 20$	Spec. 5 $N = 34$	Spec. 6 $N = 26$
1	378	1334	210
3	366	1800	290
4	164	3312	258

Table 4: Comparison of variable selection schemes 1, 3, and 4.

Spec.	Filter length	$Q(x, y)$	Scheme	Nodes
4	20	$Q(4, 7)$	1	250
			3	178
			4	96
4	20	$Q(3, 7)$	1	4056864
			3	1150
			4	84
6	20	$Q(4, 10)$	1	1948
			3	2128
			4	1982
6	20	$Q(3, 10)$	1	1948
			3	2128
			4	1982
3	34	$Q(4, 3)$	1	690
			3	962
			4	1442
3	34	$Q(3, 3)$	1	305490
			3	4880
			4	4694
1	64	$Q(4, 6)$	1	30704
			3	8986
			4	10240

Results on comparing against plainly rounding the coefficients are shown in Figure 7. As expected, results obtained by our proposed optimization method gives a smaller approximation error than obtained by rounding the real valued coefficients for the same specification. This means that it is possible to reduce the coefficient word length using the proposed technique and still meet the specification, hence, reducing the complexity of the arithmetic operators as discussed earlier. Also, by using the proposed method,

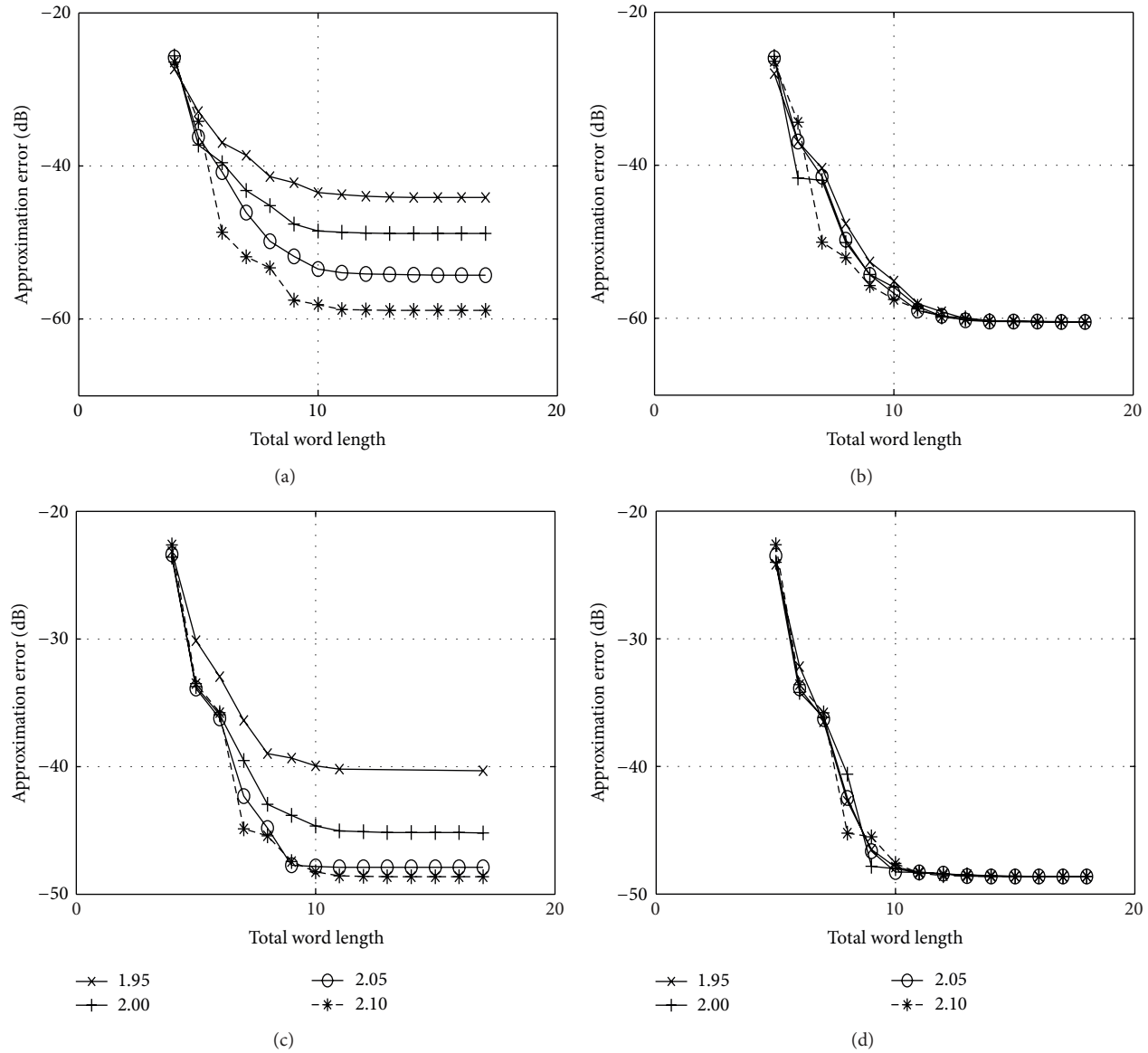

Figure 11: Approximation error as a function of logarithm base. ((a), (b)) Spec. 3, $Q(3, y)$, $Q(4, y)$. ((c), (d)) Spec. 6, $Q(3, y)$, $Q(4, y)$. $N = 34$.

monotonic results are obtained; that is, increasing the word length always improves the performance.

To further compare the performance of LNS against linear integers, approximation error as a function of filter length is plotted in Figure 8. The figure shows that for the same number of total bits, LNS gives a better approximation error. This improvement increases as the filter length is increased. To strengthen the argument made earlier that filters optimized with word length constraints in the LNS domain give better results than just rounding the linear relaxation, Figure 9 compares the approximation error of optimized filters with that of direct rounding, as a function of filter length. The plot further establishes the advantage of our proposed optimization over results obtained by rounding the real valued coefficients. In this case, the optimization leads to that a lower filter order can be reduced, and, hence, fewer arithmetic operators. Again, the proposed design method leads to monotonically decreasing approximation errors.

Finally, a comparison of the magnitude response of specification three for a filter length of 40 is shown in Figure 10. This further shows that LNS gives better results as compared with linear integers as well as illustrates the benefit of actually optimizing to obtain finite word length coefficients.

4.3. Changing the Base. Earlier works have indicated that the selection of logarithm base has an impact on the round-off noise performance. Hence, several filters are designed with varying bases. In addition, it was earlier shown that the number of integer bits is limiting the obtainable approximation error in some cases and changing the base can be seen to

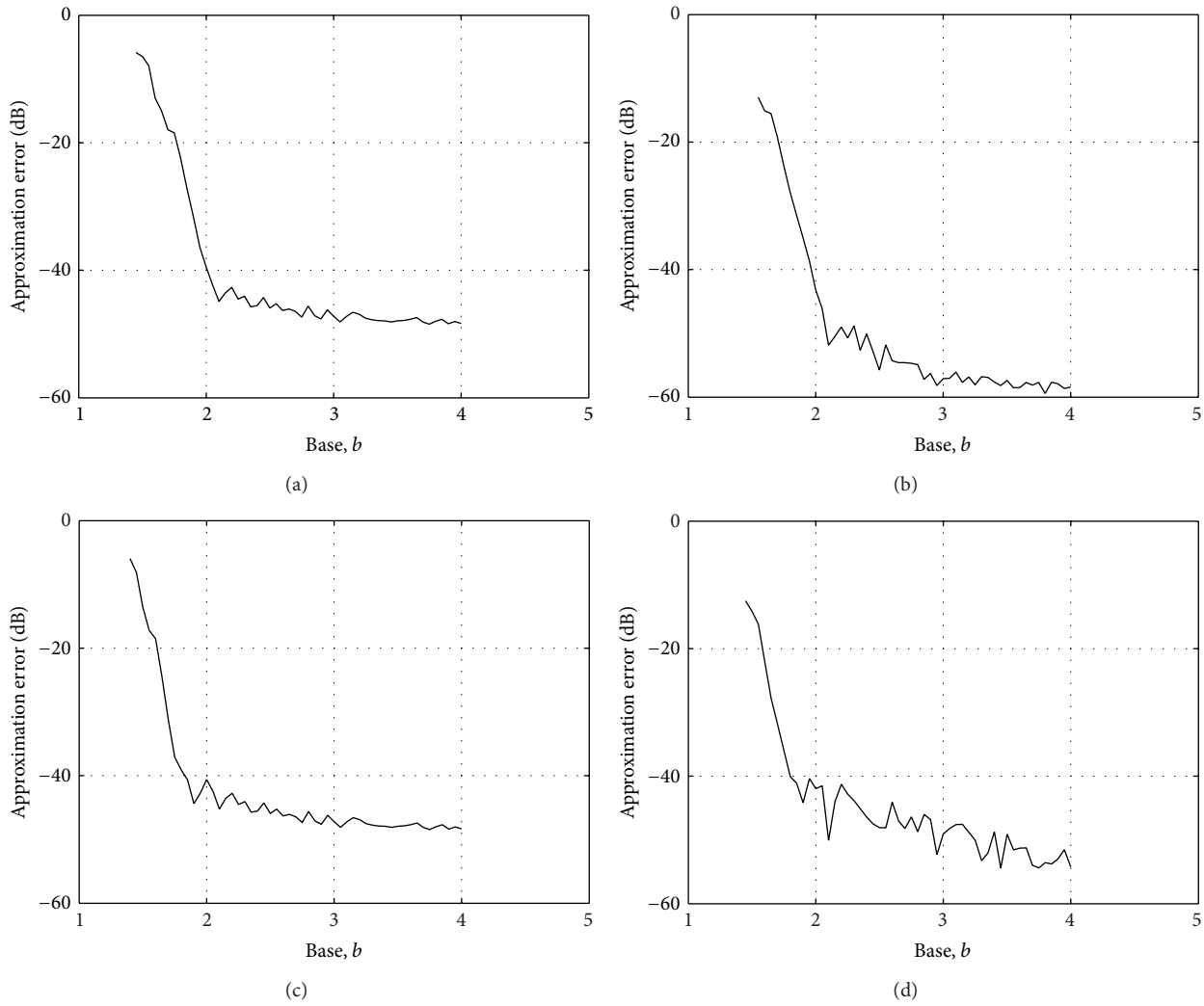

Figure 12: Approximation error as a function of logarithmic base with different word length. ((a), (b)) Spec. 3, $Q(3,4)$, $Q(4,4)$. ((c), (d)) Spec. 6 $Q(3,4)$, $Q(4,3)$. $N = 34$.

give a similar effect as having a noninteger number of integer bits. For example, using three integer bits and using a base larger than two will reduce the amplitude of the smallest representable number (compare to Figure 1).

In Figure 11 the approximation error for specifications 3 and 6 are shown using three and four integer bits. As seen, the above mentioned effect is clear; as for the three integer bit cases, the approximation error decreases when increasing the base (and increases when decreasing the base). For four integer bits, the effect of the base is not so clear. However, for short word lengths, it can be seen that there is some impact of the base.

To further illustrate the effect of selecting base the same filters were redesigned with a total word length of seven bits, that is, $Q(3,4)$ and $Q(4,3)$, but with a larger range of base values. The results are shown in Figure 12. Here, the same trend is clear and it is also evident that a base of at least two should be selected. However, it should be noted that the word length is rather small and that the same clear effect may be seen for a longer word length.

5. Conclusion

In this paper, a method for designing finite word length linear-phase FIR filters in the LNS domain was presented. Several branch variable selection and branching direction schemes were suggested and evaluated. The scheme where the branching variable was selected based on finding the largest minimum distance from the closest integer variable and branching in that direction was found to be the best on average among those suggested. The resulting filters are optimal in the minimax sense under finite word length conditions.

It was illustrated by examples that three or four integer bits were the best selection, with three being applicable for larger approximation errors. As opposed to finite word length

coefficients in the linear domain, the number of integer bits in the LNS domain determines the smallest nonzero coefficient that can be represented. Using a different and larger logarithm base than two can reduce the amplitude of the smallest representable number and slightly improve the results, especially when using three integer bits.

An interesting topic for further studies is the relation between coefficient word length (determining the magnitude response) and the data word length (determining the round-off noise). While these are completely disconnected in the linear scenario as well [1], the impact is different when LNS numbers are considered. Assuming that the data word length is shorter than the coefficient word length, the least significant bits of each multiplication will be independent of the data and vice-versa for the longer data word length. The impact of this on an architectural level may be interesting to investigate further.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] L. Wanhammar and H. Johansson, *Digital Filters*, Department of Electrical Engineering, Linköping University, 2007.

[2] S. K. Mitra, *Digital Signal Processing*, Tata McGrawHill, Santa Barbara, Calif, USA, 2006.

[3] Y. C. Lim, "Frequency-response masking approach for the synthesis of sharp linear phase digital filters," *IEEE Transactions on Circuits and Systems*, vol. 33, no. 4, pp. 357–364, 1986.

[4] T. Saramäki, "Finite impulse response filter design," in *Handbook for Digital Signal Processing*, S. K. Mitra and J. Kaiser, Eds., pp. 155–277, John Wiley & Sons, New York, NY, USA, 1988.

[5] T. Saramäki, T. Neuvo, and S. K. Mitra, "Design of computationally efficient interpolated FIR filters," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 1, pp. 70–88, 1988.

[6] H. Samueli, "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 7, pp. 1044–1047, 1989.

[7] W. J. Oh and Y.-H. Lee, "Implementation of programmable multiplierless FIR filters with powers-of-two coefficients," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 42, no. 8, pp. 553–556, 1995.

[8] O. Gustafsson, A. G. Dempster, K. Johansson, M. D. Macleod, and L. Wanhammar, "Simplified design of constant coefficient multipliers," *Circuits, Systems, and Signal Processing*, vol. 25, no. 2, pp. 225–251, 2006.

[9] P. E. Landman and J. M. Rabaey, "Architectural power analysis: the dual bit type method," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 3, no. 2, pp. 173–187, 1995.

[10] T. Stouraitis and V. Paliouras, "Considering the alternatives in low-power design," *IEEE Circuits and Devices Magazine*, vol. 17, no. 4, pp. 23–29, 2001.

[11] K.-H. Chen and T.-D. Chiueh, "A low-power digit-based reconfigurable FIR filter," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 53, no. 8, pp. 617–621, 2006.

[12] E. E. Swartzlander Jr. and A. G. Alexopoulos, "The sign/logarithm number system," *IEEE Transactions on Computers*, vol. C-24, no. 12, pp. 1238–1242, 1975.

[13] N. G. Kingsbury and P. J. W. Rayner, "Digital filtering using logarithmic arithmetic," *Electronics Letters*, vol. 7, no. 2, pp. 56–58, 1971.

[14] M. G. Arnold, T. A. Bailey, J. R. Cowles, and M. D. Winkel, "Applying features of IEEE 754 to sign/logarithm arithmetic," *IEEE Transactions on Computers*, vol. 41, no. 8, pp. 1040–1050, 1992.

[15] D. V. S. Chandra, "Error analysis of fir filters implemented using logarithmic arithmetic," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 45, no. 6, pp. 744–747, 1998.

[16] M. Haselman, M. Beauchamp, A. Wood, S. Hauck, K. Underwood, and K. S. Hemmert, "A comparison of floating point and logarithmic number systems for FPGAs," in *Proceedings of the 13th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM '05)*, pp. 181–190, Napa, Calif, USA, April 2005.

[17] P. Lee, "An FPGA prototype for a multiplierless FIR filter built using the logarithmic number system," in *Field-Programmable Logic and Applications*, vol. 975, pp. 303–310, Springer, Berlin, Germany, 1995.

[18] V. Paliouras and T. Stouraitis, "Logarithmic number system for low-power arithmetic," in *Integrated Circuit Design: Power and Timing Modeling, Optimization and Simulation*, vol. 1918, pp. 285–294, Springer, Berlin, Germany, 2000.

[19] V. Paliouras and T. Stouraitis, "Low-power properties of the logarithmic number system," in *Proceedings of the15th IEEE Symposium on Computer Arithmetic*, pp. 229–236, June 2001.

[20] A. Heřmánek, Z. Pohl, and J. Kadlec, "FPGA implementation of the adaptive lattice filter," in *Field Programmable Logic and Application*, vol. 2778 of *Lecture Notes in Computer Science*, pp. 1095–1098, Springer, Berlin, Germany, 2003.

[21] C. Basetas, I. Kouretas, and V. Paliouras, "Low-power digital filtering based on the logarithmic number system," in *Integrated Circuit and System Design. Power and Timing Modeling, Optimization and Simulation*, vol. 4644, pp. 546–555, Springer, Berlin, Germany, 2007.

[22] Y. Sun and M. S. Kim, "A high-performance 8-tap FIR filter using logarithmic number system," in *Proceedings of the IEEE International Conference on Communications (ICC '11)*, pp. 1–5, Kyoto, Japan, June 2011.

[23] I. Kouretas, C. Basetas, and V. Paliouras, "Low-power logarithmic number system addition/subtraction and their impact on digital filters," *IEEE Transactions on Computers*, vol. 62, no. 11, pp. 2196–2209, 2013.

[24] R. E. Morley Jr., G. L. Engel, T. J. Sullivan, and S. M. Natarajan, "VLSI based design of a battery-operated digital hearing aid," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '88)*, vol. 5, pp. 2512–2515, New York, NY, USA, April 1988.

[25] J. R. Sacha and M. J. Irwin, "Number representations for reducing switched capacitance in subband coding," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '98)*, vol. 5, pp. 3125–3128, Seattle, Wash, USA, May 1998.

[26] M. G. Arnold, "Reduced power consumption for MPEG decoding with LNS," in *Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures and Processors*, pp. 65–67, 2002.

[27] J. H. Lang, C. A. Zukowski, R. O. LaMaire, and C. An, "Integrated-circuit logarithmic arithmetic units," *IEEE Transactions on Computers*, vol. C-34, no. 5, pp. 475–483, 1985.

[28] O. Vainio and Y. Neuvo, "Logarithmic arithmetic in FIR filters," *IEEE Transactions on Circuits and Systems*, vol. 33, no. 8, pp. 826–828, 1986.

[29] O. Vainio, "Biased logarithmic arithmetic in FIR filters," *Electronics Letters*, vol. 41, no. 10, pp. 580–581, 2005.

[30] K. Johansson, O. Gustafsson, and L. Wanhammar, "Implementation of elementary functions for logarithmic number systems," *IET Computers and Digital Techniques*, vol. 2, no. 4, pp. 295–304, 2008.

[31] H. A. Taha, *Operations Research—An Introduction*, Pearson Prentice Hall, 8th edition, 2007.

[32] L. R. Rabiner, "Linear program design of finite impulse response FIR digital filters," *IEEE Transactions on Audio and Electroacoustics*, vol. 20, no. 4, pp. 280–288, 1972.

[33] D. M. Kodek, "Design of optimal finite wordlength FIR digital filters using integer programming techniques," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 3, pp. 304–308, 1980.