A Block Floating-Point Treatment to the LMS Algorithm: Efficient Realization and a Roundoff Error Analysis

Abhijit Mitra, *Member, IEEE*, Mrityunjoy Chakraborty, *Senior Member, IEEE*, and Hideaki Sakai, *Senior Member, IEEE*

***Abstract*—An efficient scheme is presented for implementing the LMS-based transversal adaptive filter in block floating-point (BFP) format, which permits processing of data over a wide dynamic range, at temporal and hardware complexities significantly less than that of a floating-point processor. Appropriate BFP formats for both the data and the filter coefficients are adopted, taking care so that they remain invariant to interblock transition and weight updating operation, respectively. Care is also taken to prevent overflow during filtering, as well as weight updating processes jointly, by using a dynamic scaling of the data and a slightly reduced range for the step size, with the latter having only marginal effect on convergence speed. Extensions of the proposed scheme to the sign–sign LMS and the signed regressor LMS algorithms are taken up next, in order to reduce the processing time further. Finally, a roundoff error analysis of the proposed scheme under finite precision is carried out. It is shown that in the steady state, the quantization noise component in the output mean-square error depends on the step size both linearly and inversely. An optimum step size that minimizes this error is also found out.**

***Index Terms*—Block floating-point arithmetic, least mean square methods, overflow, roundoff errors.**

I. Introduction

A HIGH signal-to-quantization-noise ratio (SNR) over a reasonably large dynamic range is an important desirable feature for digital signal processing systems. The block floating-point (BFP) data format [1] is a viable alternative to the floating-point (FP) concept for achieving this, requiring much less processing time than the latter at a moderately low processor complexity and cost. Under this scheme, the incoming data is partitioned into nonoverlapping blocks, and based on the data sample with the highest magnitude in each block, a common exponent is assigned to the block. This permits an overall FP-like representation of the data, with fixed point (FxP)-like computation within every block.

The advantages provided by the BFP format, namely a wide dynamic range like the FP scheme with temporal and processor complexity, power requirement, and cost that are similar to that of FxP-based processing, have prompted its usage in efficient implementation of many signal processing algorithms, like state–space digital filters [2], [3], direct form digital filters [4]–[7], distributed digital filters [8], fast Hartley transform [9], fast Fourier transform (FFT) [10], etc. Some studies [4], [11] have also been made to investigate the associated numerical error behavior. Amongst other advantages of the BFP scheme, the following have been widely recognized:

1) less storage requirement as there is no need to store several exponent values;
2) suitability for block-based implementation of digital filters [2];
3) limit-cycle-free realization of stable filters irrespective of structure chosen [11].

The BFP format has, in fact, been used in several digital audio data transmission standards like NICAM (a stereophonic sound system for PAL TV standard), the audio part of MUSE (the Japanese HDTV standard), and DSR (the German Digital Satellite Radio System).

However, to the best of our knowledge, no effort has so far been made to extend the BFP treatment to adaptive filters, which present more complex structures, including error feedback. A BFP treatment to adaptive filters faces certain difficulties, not encountered in the fixed coefficient case, namely, the following.

- Unlike a fixed coefficient filter, the filter coefficients in an adaptive filter *cannot* be represented in the simpler fixed-point form, as the coefficients in effect evolve from the data by a time update relation.
- The two principal operations in an adaptive filter—filtering and weight updating—are mutually coupled, thus requiring an appropriate arrangement for joint prevention of overflow.

In this paper, we present a novel scheme for BFP realization of the LMS-based transversal adaptive filter [12]. The proposed approach adopts appropriate BFP formats for the data and the filter coefficients separately, taking care so that 1) the chosen format for the filter coefficients remains invariant to the weight adjustment process and 2) a uniform BFP representation of the filter input vector is available during block-to-block transition phase when the data comes from two adjacent blocks with two different block exponents. Care is also taken to prevent overflow

Manuscript received August 11, 2003; revised January 20, 2005. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ta-Hsin Li.

A. Mitra is with the Department of Electronics and Communication Engineering, Indian Institute of Technology, Guwahati, 781039 Assam, India (e-mail: a.mitra@iitg.ernet.in).

M. Chakraborty is with the Department of Electronics and Electrical Communication Engineering, Indian Institute of Technology, Kharagpur 721302, W.B., India (e-mail: mrityun@ece.iitkgp.ernet.in).

H. Sakai is with the Department of System Science, Graduate School of Informatics, Kyoto University, Kyoto 6068501, Japan (e-mail: hsakai@i.kyoto-u.ac.jp).

Digital Object Identifier 10.1109/TSP.2005.859342

1053-587X/$20.00 © 2005 IEEE

during filtering and weight updating processes jointly by using a dynamic scaling of the data and a new upper bound on the algorithm step size μ. The latter is slightly less than the well-known upper limit of μ for algorithm convergence, i.e., $2/\mathrm{tr}\mathbf{R}$ ($\mathbf{R}$: input correlation matrix) and has only a marginal effect on convergence speed. We also consider extensions of the proposed approach to two sign LMS algorithms, namely, the sign–sign algorithm and the signed regressor algorithm. Since these algorithms do not need multipliers in the weight updating process, they are easier to implement in practice, and a BFP treatment to them enjoys certain additional advantages not present in the general LMS case. Finally, we adopt the BFP roundoff noise model presented in [4] and evaluate the steady-state value of the output mean-square error for finite precision realization of the proposed scheme. In particular, we show that the contribution from the various quantization noise in the output mean-square error is proportional to μ, both linearly and inversely. An optimum value of μ that minimizes this error is then found out.

The organization of the paper is as follows. In Section II, the BFP arithmetic is discussed in brief. Section III presents the proposed BFP realization of the LMS algorithm, including its extensions to the sign–sign and signed-regressor algorithms. Section IV takes up the steady-state roundoff error analysis. Finally, finite-precision-based simulation results are presented in Section V.

II. The BFP Arithmetic

The BFP representation can be considered as a special case of the FP format, where every nonoverlapping block of N incoming data has a joint scaling factor corresponding to the data sample with the highest magnitude in the block. In other words, given a block $[x_1, \ldots, x_N]$, we represent it as $[x_1, \ldots, x_N] = [\overline{x}_1, \ldots, \overline{x}_N] \cdot 2^{\gamma}$, where $\overline{x}_l (= x_l \cdot 2^{-\gamma})$ represents the mantissa for $l = 1, 2, \ldots, N$, and the block exponent γ is defined as $\gamma = \lfloor \log_2 \mathrm{Max} \rfloor + 1 + S$, where $\mathrm{Max} = \max(|x_1|, \ldots, |x_N|)$, "$\lfloor . \rfloor$" is the so-called floor function, meaning rounding down to the closest integer and the integer S is a scaling factor that is needed to prevent overflow during filtering operation. Due to the presence of S, the range of each mantissa is given as $0 \leq |\overline{x}_l| < 2^{-S}$. The scaling factor S can be calculated from the inner product computation representing filtering operation. An inner product is calculated in BFP arithmetic as

$$\begin{aligned} y(n) &= \mathbf{w}^t \mathbf{x}(n) \\ &= [w_0 \overline{x}(n) + \cdots + w_{L-1} \overline{x}(n-L+1)] \cdot 2^{\gamma} \\ &= \overline{y}(n) \cdot 2^{\gamma} \end{aligned} \tag{1}$$

where $\mathbf{w}$ is a length L, fixed-point filter coefficient vector, and $\mathbf{x}(n)$ is the data vector at the nth index, represented in the aforesaid BFP format. For no overflow in $y(n)$, we need $|\overline{y}(n)| < 1$ at every time index, which can be satisfied [4] by selecting $S \geq S_{\min} = \lceil \log_2 (\sum_{k=0}^{L-1} |w_k|) \rceil$, where "$\lceil . \rceil$" is the so-called ceiling function, meaning rounding up to the closest integer.

Note that, if ($B_d +$ one sign) bits are used to represent each mantissa within the block and if ($B_\gamma +$ one sign) bits are used to account for the block exponent, then effectively, under BFP system, each sample can be equivalently represented with $(B_d + 1) + (B_\gamma + 1)/N$ bits because the block exponent is assigned only once for the whole block. This particular strength makes this format more attractive than FxP or FP systems.

III. The Proposed Implementation

A. BFP Realization of the LMS Algorithm

Consider a length L LMS-based adaptive filter that takes an input sequence $x(n)$ and updates the weights as

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \mathbf{x}(n) e(n) \tag{2}$$

where $\mathbf{w}(n) = [w_0(n) w_1(n) \ldots w_{L-1}(n)]^t$ is the tap weight vector at the nth index, $\mathbf{x}(n) = [x(n) x(n-1) \ldots x(n-L+1)]^t$, $e(n) = d(n) - \mathbf{w}^t(n)\mathbf{x}(n)$, with $d(n)$ being the so-called desired response available during the initial training period and μ denoting the so-called step-size parameter.

The proposed scheme starts with two simultaneous BFP representations—one for the filter coefficient vector $\mathbf{w}(n)$ and the other for the given data, namely, $x(n)$ and $d(n)$, which are then used to obtain BFP-based expressions for the filter output, the output error, and the weight update relations.

1) BFP Format of the Filter Coefficient Vector: In this, at each index of time, we have a scaled representation of the filter coefficient vector as

$$\mathbf{w}(n) = \overline{\mathbf{w}}(n) \cdot 2^{\psi_n} \tag{3}$$

where ψ_n is a time-varying block exponent that needs to be updated at each index n and is chosen to ensure that each $|\overline{w}_k(n)| < 1/2$ for $k \in Z_L = \{0, 1, \ldots, L-1\}$. If a data vector $\mathbf{x}(n)$ is given in the aforesaid BFP format as $\mathbf{x}(n) = \overline{\mathbf{x}}(n) \cdot 2^{\gamma}$, where $\gamma = ex + S$, $ex = \lfloor \log_2 M \rfloor + 1$, $M = \max(|x(n-k)| \, |k \in Z_L)$, and S is an appropriate scaling factor, then, the filter output $y(n)$ can be expressed as $y(n) = \overline{y}(n) \cdot 2^{\gamma + \psi_n}$ with $\overline{y}(n) = \overline{\mathbf{w}}^t(n) \overline{\mathbf{x}}(n)$ denoting the output mantissa. To prevent overflow in $\overline{y}(n)$, it is required that $|\overline{y}(n)| < 1$. However, in the proposed scheme, we restrict $\overline{y}(n)$ to lie between $+1/2$ and $-1/2$, i.e., $|\overline{y}(n)| < 1/2$. Since $|\overline{y}(n)| \leq \sum_{k=0}^{L-1} |\overline{w}_k(n)| |\overline{x}(n-k)|$, $0 \leq |\overline{x}(n-k)| < 2^{-S}$, and $|\overline{w}_k(n)| < 1/2$, this implies a lower limit of S as $S_{\min} = \lceil \log_2 L \rceil$. The two conditions $|\overline{w}_k(n)| < 1/2, k \in Z_L$, and $|\overline{y}(n)| < 1/2$ ensure no overflow during updating of $\overline{\mathbf{w}}(n)$ and computation of output error mantissa, respectively, as shown later.

2) BFP Representation of the Given Data: The input data $x(n)$ and the desired response sequence $d(n)$ are partitioned jointly in nonoverlapping blocks of N samples each ($N \geq L - 1$), with the ith block ($i \in Z$) consisting of $x(n)$, $d(n)$ for $n \in Z_i' = \{iN, iN+1, \ldots, iN+N-1\}$. Further, both $x(n)$ and $d(n)$ are jointly scaled so as to have a common BFP representation within each block. This means that, for $n \in Z_i'$, $x(n)$ and $d(n)$ are expressed as

$$x(n) = \overline{x}(n) \cdot 2^{\gamma_i}, \;\; d(n) = \overline{d}(n) \cdot 2^{\gamma_i} \tag{4}$$

where γ_i is the common block exponent for the ith block and is given as $\gamma_i = ex_i + S_i$, where $ex_i = \lfloor \log_2 M_i \rfloor + 1$ and $M_i = \max\{|x(n)|, |d(n)| \, |n \in Z_i'\}$. The scaling factor S_i is assigned as per the following algorithm.

```
Algorithm:
Assign $S_{\min} = \lceil \log_2 L \rceil$ as the scaling factor
   to the first block and for any $(i - 1)$th
   block, assume $S_{i-1} \geq S_{\min}$.
Then, if $ex_i \geq ex_{i-1}$,
 choose $S_i = S_{\min}$ (i.e., $\gamma_i = ex_i + S_{\min}$)
 else (i.e., $ex_i < ex_{i-1}$)
 choose $S_i = (ex_{i-1} - ex_i + S_{\min})$, s.t. $\gamma_i = ex_{i-1} +$
 $S_{\min}$.
```

Note that when $ex_i \geq ex_{i-1}$, we can either have $ex_i + S_{\min} \geq \gamma_{i-1}$ (case A) implying $\gamma_i \geq \gamma_{i-1}$, or $ex_i + S_{\min} < \gamma_{i-1}$ (case B) meaning $\gamma_i < \gamma_{i-1}$. However, for $ex_i < ex_{i-1}$ (case C), we always have $\gamma_i \leq \gamma_{i-1}$.

In addition, we rescale the elements $\overline{x}(iN - L + 1), \ldots, \overline{x}(iN - 1)$ by dividing by $2^{\Delta\gamma_i}$, where $\Delta\gamma_i = \gamma_i - \gamma_{i-1}$. Equivalently, for the elements $x(iN - L + 1), \ldots, x(iN - 1)$, we change S_{i-1} to an effective scaling factor of $S'_{i-1} = S_{i-1} + \Delta\gamma_i$. This permits a BFP representation of the data vector $\mathbf{x}(n)$ with common exponent γ_i during block-to-block transition phase as well, i.e., when part of $\mathbf{x}(n)$ comes from the $(i-1)$th block and part from the ith block.

In practice, such rescaling is effected by passing each of the delayed terms $\overline{x}(n - j)$, $j = 1, \ldots, L - 1$, through a rescaling unit that applies $\Delta\gamma_i$ number of right or left shifts on $\overline{x}(n - j)$ depending on whether $\Delta\gamma_i$ is positive or negative, respectively. This is, however, done only at the beginning of each block, i.e., at indexes $n = iN$, $i \in Z$. Also, note that though for the case A, $\Delta\gamma_i \geq 0$, for B and C, however, $\Delta\gamma_i \leq 0$, meaning that in these cases, the aforesaid mantissas from the $(i-1)$th block are actually scaled up by $2^{-\Delta\gamma_i}$. It is, however, not difficult to see that the effective scaling factor S'_{i-1} for the elements $x(iN - L + 1), \ldots, x(iN - 1)$ still remains lower bounded by $S_{\min}$, thus ensuring no overflow during filtering operation.

3) Formulation of the LMS Algorithm in BFP Format: First, using the above representations, the filter output $y(n)$, $n \in Z'_i$ is obtained as $y(n) = \overline{y}(n) \cdot 2^{\gamma_i + \psi_n}$. The output error $e(n)$ is then evaluated as $e(n) = \overline{e}(n) \cdot 2^{\gamma_i + \psi_n}$, where the mantissa $\overline{e}(n)$ is given by

$$\overline{e}(n) = \overline{d}(n) \cdot 2^{-\psi_n} - \overline{y}(n). \tag{5}$$

Clearly, computation of $\overline{e}(n)$ involves an additional step of right-shift operation on $\overline{d}(n)$—an operation that comes up frequently in FP arithmetic. However, since in an adaptive filter, filter coefficients are derived from data and thus cannot be represented in the FxP format when data is given in a scaled form, such a step seems to be unavoidable. It is easy to see that $|\overline{e}(n)| < 1$, since

$$\begin{aligned} |\overline{e}(n)| &\leq |\overline{d}(n)| \cdot 2^{-\psi_n} + |\overline{y}(n)| \\ &< 2^{-(S_i + \psi_n)} + \frac{1}{2} \leq \frac{2^{-\psi_n}}{L} + \frac{1}{2} \end{aligned} \tag{6}$$

as $2^{-S_i} \leq 1/L$. Except for $\psi_n = 0$, $L = 1$, the right-hand side (RHS) is always less than or equal to 1.

For the above description of $e(n)$, $\mathbf{x}(n)$, $d(n)$, and $\mathbf{w}(n)$, the weight update equation (2) can now be written as $\mathbf{w}(n+1) = \overline{\mathbf{v}}(n) \cdot 2^{\psi_n}$, where

$$\overline{\mathbf{v}}(n) = \overline{\mathbf{w}}(n) + \mu\overline{\mathbf{x}}(n)\overline{e}(n) \cdot 2^{2\gamma_i}. \tag{7}$$

As stated earlier, $\overline{\mathbf{w}}(n+1)$ is required to satisfy $|\overline{w}_k(n+1)| < 1/2$ for $k \in Z_L$, which can be realized in several ways. Our preferred option is to limit $\overline{\mathbf{v}}(n)$ so that $|\overline{v}_k(n)| < 1$, $k \in Z_L$. Then, if each $\overline{v}_k(n)$ happens to be lying within $\pm 1/2$, we make the assignments:

$$\overline{\mathbf{w}}(n+1) = \overline{\mathbf{v}}(n), \;\; \psi_{n+1} = \psi_n. \tag{8}$$

Otherwise, we scale down $\overline{\mathbf{v}}(n)$ by 2, in which case

$$\overline{\mathbf{w}}(n+1) = \frac{1}{2}\overline{\mathbf{v}}(n), \;\; \psi_{n+1} = \psi_n + 1. \tag{9}$$

In order to have $|\overline{v}_k(n)| < 1$, $k \in Z_L$ satisfied, we observe that $|\overline{v}_k(n)| \leq |\overline{w}_k(n)| + \mu|\overline{x}(n-k)||\overline{e}(n)| \cdot 2^{2\gamma_i}$. Since $|\overline{w}_k(n)| < 1/2$, $k \in Z_L$, it is sufficient to have $\mu|\overline{x}(n-k)||\overline{e}(n)| \cdot 2^{2\gamma_i} < 1/2$. Taking the upper bound of $|\overline{e}(n)|$ as $[2^{-(S_i + \psi_n)} + L/2 \cdot 2^{-S_i}]$ and recalling that $|\overline{x}(n-k)| < 2^{-S_i}$, this implies

$$\mu \leq \frac{2^{-2ex_i}}{2^{-\psi_n + 1} + L}. \tag{10}$$

It is easy to verify that the above bound for μ is valid not only when each element of $\overline{\mathbf{x}}(n)$ in (7) comes purely from the ith block, but also during transition from the $(i-1)$th to the ith block with $ex_i \geq ex_{i-1}$, for which, after necessary rescaling, we have $S'_{i-1} \geq S_i = S_{\min}$ implying $|\overline{x}(n-k)| < 2^{-S_i}$ and thus $\overline{y}(n) < L/2 \cdot 2^{-S_i}$. For $ex_i < ex_{i-1}$, however, the upper bound expression given by (10) gets modified with ex_i replaced by ex_{i-1}, as in that case, we have $\gamma_i = ex_{i-1} + S'_{i-1}$ with $S'_{i-1} = S_{\min} < S_i$ meaning $|\overline{x}(n-k)| < 2^{-S'_{i-1}}$ and thus $\overline{y}(n) < L/2 \cdot 2^{-S'_{i-1}}$, leading to $|\overline{e}(n)| < [2^{-(S'_{i-1} + \psi_n)} + L/2 \cdot 2^{-S'_{i-1}}]$.

From above, we obtain a general upper bound for μ by equating ψ_n to its lowest value of zero and replacing ex_i by $ex_{\max} = \max\{ex_i | i \in Z\}$ in (10). The general upper bound is given by

$$\mu \leq \frac{2^{-2ex_{\max}}}{L + 2}. \tag{11}$$

The above bound is actually less than $2/\text{tr}\mathbf{R}$, which is the upper bound of μ for convergence of the LMS algorithm. To see this, we note that $|x(n)| < 2^{ex_{\max}}$ and thus $E[x^2(n)] < 2^{2ex_{\max}}$. This implies $\text{tr}\mathbf{R} < L \cdot 2^{2ex_{\max}}$ and thus $2/\text{tr}\mathbf{R} > 2^{-2ex_{\max}}/(L+2)$.

Finally, for practical implementation of $\overline{\mathbf{v}}(n)$ as given by (7), we need to evaluate the product $\mu\overline{x}(n-k)\overline{e}(n)2^{2\gamma_i}$ in such a way that no overflow occurs in any of the intermediate products. This is realized by expressing $2^{2\gamma_i}$ as $2^{2\gamma_i} = 2^{2ex_i} \cdot 2^{S_i} \cdot 2^{S_i}$ and distributing the factors as per the following steps :

Step 1 $\rightarrow \mu \cdot 2^{2ex_i} = \mu_1$ (say),
Step 2 $\rightarrow [\mu_1 \overline{e}(n)] \cdot 2^{S_i} = \overline{e}_1(n)$ (say) and
Step 3 $\rightarrow \overline{e}_1(n)[\overline{x}(n-k) \cdot 2^{S_i}] = \overline{x}_1(n-k)$ (say).

Once again, for the block-to-block transitional case with $ex_i < ex_{i-1}$, ex_i in Step 1 and S_i in Steps 2 and 3 are to be replaced by ex_{i-1} and $S'_{i-1}(= S_{\min})$, respectively.

It is easy to check that each of the intermediate products computed in Steps 1–3 above has magnitude less than one, and thus there is no intermediate overflow. First, from (10), it follows that $\mu_1 \leq 1/(2^{-\psi_n+1} + L)$. Next, from this and recalling that the upper bound of $\overline{e}(n)$ is given by $[2^{-(S_i+\psi_n)} + L/2 \cdot 2^{-S_i}]$ [replace S_i by S'_{i-1} for the block-to-block transitional case with $ex_i < ex_{i-1}$], it is easily seen that $|\overline{e}_1(n)| < 1/2$. An important point to note here is that in Step 2 above, we evaluate $\overline{e}_1(n)$ as $[\mu_1 \overline{e}(n)] \cdot 2^{S_i}$, rather than as $[\mu_1 \cdot 2^{S_i}]\overline{e}(n)$, or as $[\overline{e}(n) \cdot 2^{S_i}]\mu_1$, as neither of the quantities $[\mu_1 \cdot 2^{S_i}]$ and $[\overline{e}(n) \cdot 2^{S_i}]$ is guaranteed to have magnitude less than one. Finally, in Step 3, $|\overline{x}_1(n-k)| < 1/2$, as $|\overline{x}(n-k) \cdot 2^{S_i}| < 1$ and $|\overline{e}_1(n)| < 1/2$. However, here too we compute $\overline{x}_1(n-k)$ as $\overline{e}_1(n)[\overline{x}(n-k) \cdot 2^{S_i}]$, rather than as $[\overline{e}_1(n)\overline{x}(n-k)] \cdot 2^{S_i}$, or as $[\overline{e}_1(n) \cdot 2^{S_i}]\overline{x}(n-k)$, since, in the case of the former, the product $[\overline{e}_1(n)\overline{x}(n-k)]$ can be very small, resulting in large rounding error/underflow in finite precision, while, in the case of the latter, $[\overline{e}_1(n) \cdot 2^{S_i}]$ is not bounded by one, thus having the possibility of overflow.

It is also interesting to examine the variation of ψ_n with n. From (8) and (9), it follows that ψ_n is a nondecreasing function of n. Further, we note that while $\overline{\mathbf{w}}(n+1)$ is either $\overline{\mathbf{v}}(n)$ or $(1/2)\overline{\mathbf{v}}(n)$, $\mathbf{w}(n+1)$ is, however, always given as $\mathbf{w}(n+1) = \overline{\mathbf{v}}(n) \cdot 2^{\psi_n}$. Thus, for any $j \in Z_L$, the condition $|w_j(n+1)| < (>)\ |w_j(n)|$ implies $|\overline{v}_j(n)| < (>)\ |\overline{w}_j(n)|$. As a result, at any index n, if it is given that $|w_j(n+1)| \leq |w_j(n)|$ for *all* $j \in Z_L$, then, since $|\overline{w}_j(n)| < 1/2$, it follows that $|\overline{v}_j(n)| < 1/2$ for each $j \in Z_L$, and thus $\psi_{n+1} = \psi_n$. However, if for any $j \in Z_L$, $|w_j(n+1)| > |w_j(n)|$, then $|\overline{v}_j(n)|$ is not guaranteed to be less than 1/2, and we have $\psi_{n+1} = \psi_n + 1$ if $|\overline{v}_j(n)|$ equals or exceeds 1/2. Clearly, if we consider the L "$w_j(n)$-versus-n" trajectories, then whenever a power of 2 in the order $1/2, 1, 2, \ldots$ is crossed by these trajectories for the first time, ψ_n is incremented and its value held until the next power of 2 is crossed. It is then easy to check that the steady-state value of ψ_n will be given by $(\lfloor \log_2 W \rfloor + 2)$, where $W = \max\{|\overline{w}_j(n)|\,|j \in Z_L,\ n = 0, 1, 2, \ldots\}$.

The proposed BFP-based LMS algorithm is summarized in Table I. Details of its advantages over its FP-based counterpart in terms of less processing time and hardware complexity are discussed in Section III-C. It is, however, possible to reduce complexities further by getting rid of the $(L+2)$ shift operations present in Steps 1–3 above. This is achieved by considering extensions of the proposed approach to special cases of the LMS algorithm like the sign LMS algorithm [13], as discussed next.

B. Extension to the Sign LMS Algorithm

In the literature, there exist three versions of the sign LMS algorithm, namely, the sign–sign algorithm, the signed regressor algorithm, and the sign algorithm [13], all three requiring only half as many multiplications as in the LMS algorithm, thus making them attractive from practical implementation point of view. We present here a BFP treatment to the first two. The same for the third can be developed along analogous lines and is not discussed here. In addition, for both the cases considered, we retain the same format for data and filter weights as employed for the LMS implementation above.

TABLE I
SUMMARY OF THE LMS ALGORITHM REALIZED IN BFP FORMAT (INITIAL VALUE OF $\psi_n = 0$)

```
1. Preprocessing:
   Using the data for the $i$-th block , $x(n)$ and $d(n)$, $n \in {Z'}_i$
   (stored during the processing of the $(i-1)$-th block),
(a) Evaluate block exponent $\gamma_i$ as per the Algorithm
    of Section 3 and express $x(n)$, $d(n)$, $n \in Z'_i$ as
    $x(n) = \overline{x}(n).2^{\gamma_i}$, $d(n) = \overline{d}(n).2^{\gamma_i}$,
(b) Rescale the following elements of the $(i-1)$-th block:
    $\{\overline{x}(n) | n = iN - L + 1, ..., iN - 1\}$ as
    $\overline{x}(n) \rightarrow \overline{x}(n).2^{-\Delta\gamma_i}$, $\Delta\gamma_i = \gamma_i - \gamma_{i-1}$.
2. Processing for the $i$-th block:
   For $n \in Z'_i = \{iN, iN+1, ..., iN+N-1\}$
(a) Filter ouput:
    $\overline{y}(n) = \overline{\mathbf{w}}^t\ (n)\overline{\mathbf{x}}(n)$,
    $ex\_out(n) = \gamma_i + \psi_n$.
    ($ex\_out(n)$ is the filter output exponent)
(b) Output error (mantissa) computation:
    $\overline{e}(n) = \overline{d}(n).2^{-\psi_n} - \overline{y}(n)$.
(c) Filter weight updating:
    Compute $\overline{x}_1(n-k) = \mu\overline{x}(n-k)\overline{e}(n)2^{2\gamma_i}$ for all $k \in Z_L$
    following $Step1 - Step3$ of Section 3.
    $\overline{\mathbf{v}}(n) = \overline{\mathbf{w}}(n) + \overline{\mathbf{x}}_1(n)$.
    (where $\overline{\mathbf{x}}_1(n) = [\overline{x}_1(n), \ldots, \overline{x}_1(n-L+1)]^t$)
    If $|\overline{v}_k(n)| < \frac{1}{2}$ for all $k \in Z_L = \{0, 1, ..., L-1\}$
    then
        $\overline{\mathbf{w}}(n+1) = \overline{\mathbf{v}}(n)$,
        $\psi_{n+1} = \psi_n$,
    else
        $\overline{\mathbf{w}}(n+1) = \frac{1}{2}\overline{\mathbf{v}}(n)$,
        $\psi_{n+1} = \psi_n + 1$.
    end.
    $i = i + 1$.
    Repeat steps 1 to 2.
```

1) The Sign–Sign Algorithm: The sign–sign algorithm is obtained from the LMS algorithm by considering only the sign of the gradient rather than its total value. The corresponding weight update recursion is given by,

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \operatorname{sgn}\{\mathbf{x}(n)\} \operatorname{sgn}\{e(n)\} \tag{12}$$

where $\operatorname{sgn}\{.\}$ is the well-known signum function. Although μ is a constant, for the BFP treatment, we express it as $\mu = \overline{\mu}_n \cdot 2^{\psi_n}$, with $\overline{\mu}_n$ updated from a knowledge of ψ_n and ψ_{n+1} as $\overline{\mu}_{n+1} = \overline{\mu}_n \cdot 2^{(\psi_n - \psi_{n+1})}$. Since $\operatorname{sgn}\{\mathbf{x}(n)\} = \operatorname{sgn}\{\overline{\mathbf{x}}(n)\}$ and $\operatorname{sgn}\{e(n)\} = \operatorname{sgn}\{\overline{e}(n)\}$, we can express (12) as $\mathbf{w}(n+1) = \overline{\mathbf{v}}(n) \cdot 2^{\psi_n}$, where $\overline{\mathbf{v}}(n)$ is now given as

$$\overline{\mathbf{v}}(n) = \overline{\mathbf{w}}(n) + \overline{\mu}_n \operatorname{sgn}\{\overline{\mathbf{x}}(n)\} \operatorname{sgn}\{\overline{e}(n)\}.$$

In other words, $\overline{v}_j(n) = \overline{w}_j(n) \pm \overline{\mu}_n$, $j \in Z_L$. As before, we limit $\overline{\mathbf{v}}(n)$ so that $|\overline{v}_j(n)| < 1$, $j \in Z_L$. It is easy to see that this is satisfied for $\overline{\mu}_n \leq 1/2$. To have this condition satisfied, we first observe that ψ_0 (i.e., initial value of ψ_n) is zero and thus, $\mu = \mu_0$. Second, ψ_n is a nondecreasing function of n, meaning that $\overline{\mu}_n$ can only decrease with n. Thus, it is enough to choose

μ by considering a binary word that keeps at least one zero after the binary point.

2) The Signed-Regressor Algorithm: For the signed-regressor algorithm, the weight update recursion is given as

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu\,\mathrm{sgn}\{\mathbf{x}(n)\}\,e(n). \tag{13}$$

Proceeding as above, we have, in this case, $\overline{\mathbf{v}}(n) = \overline{\mathbf{w}}(n) + \mu\,\mathrm{sgn}\{\overline{\mathbf{x}}(n)\}\,\overline{e}(n)\cdot 2^{\gamma_i}$. Recalling that $\overline{e}(n) < [2^{-(S_i+\psi_n)} + L/2 \cdot 2^{-S_i}]$, it is easy to check that the condition $|\overline{v}_k(n)| < 1$, $k \in Z_L$, is satisfied for

$$\mu \leq \frac{2^{-ex_{\max}}}{L+2}.$$

As before, the product $\mu\ \overline{e}(n)\cdot 2^{\gamma_i}$ is evaluated subject to no overflow in the intermediate products. This is done as per the following steps:

```
Step A → $\mu \cdot 2^{ex_i} = \mu_1$ (say),
Step B → $[\mu_1 \overline{e}(n)] \cdot 2^{S_i}$.
  [Replace $ex_i$ and $S_i$ by $ex_{i-1}$ and $S'_{i-1} (= S_{\min})$
  respectively for the block-to-block
  transitional case with $ex_i < ex_{i-1}$, as
  explained earlier.]
```

Finally, for both (i) and (ii) above, $\overline{\mathbf{w}}(n+1)$ and ψ_{n+1} are obtained from $\overline{\mathbf{v}}(n)$ and ψ_n in the same way as before, i.e., via (8) and (9).

C. Complexity Issues

The proposed schemes rely mostly on FxP arithmetic and are largely free from the usual FP-operations-like shift, exponent comparison, exponent addition, etc., resulting in computational complexities significantly less than that of their FP-based counterparts. For example, to compute the filter output, the BFP method requires L "multiply and accumulate" (MAC) operations (FxP) and, at the most, one exponent addition for each n, in all the schemes proposed above. In FP, this, however, requires the following additional operations per sample: 1) $2L$ shifts (assuming availability of single cycle barrel shifters), 2) L exponent comparisons, and, 3) $2L-1$ exponent additions. Similar advantages exist in weight updating as well. In both cases, the number of additional operations required under FP-based realization increases linearly with filter length L. In the context of the LMS algorithm, Table II provides a comparative account of the two approaches in terms of number of operations required. It is easily seen from this table that given a low-cost, simple FxP processor with single-cycle MAC and barrel shifter units, the proposed scheme is about *two and a half times faster* than an FP-based implementation. The speed-up gets enhanced further to about *three* in the case of the sign–sign LMS and the signed-regressor LMS algorithms, for which the operation counts for weight updating are provided in Table III (for filtering, there is no change in the number of operations from those quoted in Table II).

TABLE II

COMPARISON BETWEEN THE BFP VIS-À-VIS THE FP-BASED REALIZATIONS OF THE LMS ALGORITHM. NUMBER OF OPERATIONS REQUIRED PER ITERATION FOR (a) WEIGHT UPDATING AND (b) FILTERING ARE GIVEN. UNLESS SPECIFIED OTHERWISE, ALL THE GENERAL OPERATIONS INDICATE MANTISSA OPERATIONS

(a)	MAC	Shift	Magnitude Check	Exponent Comparison	Exponent Addition
BFP	$L+1$	$2L+2$	L	Nil	1
FP	$L+1$	$2L+1$	Nil	L	$2L+2$

(b)	MAC	Shift	Exponent Comparison	Exponent Addition
BFP	L	Nil	Nil	1
FP	L	$2L$	L	$2L$

TABLE III

COMPARISON BETWEEN THE BFP VIS-À-VIS THE FP-BASED REALIZATIONS OF (a) THE SIGN–SIGN LMS AND (b) THE SIGNED-REGRESSOR LMS ALGORITHMS. OPERATIONAL COUNTS PER ITERATION FOR WEIGHT UPDATING ONLY ARE GIVEN (FOR FILTERING, FIGURES REMAIN SAME AS GIVEN IN TABLE II). UNLESS SPECIFIED OTHERWISE, ALL THE GENERAL OPERATIONS INDICATE MANTISSA OPERATIONS

(a)	Add with Sign Check	Shift	Magnitude Check	Exponent Comparison	Exponent Addition
BFP	L	$L+1$	L	Nil	1
FP	L	$2L$	Nil	L	L

(b)	MAC	Add with Sign Check	Shift	Magnitude Check	Exponent Comparison	Exponent Addition
BFP	1	L	$L+2$	L	Nil	1
FP	1	L	$2L+1$	Nil	L	$L+2$

IV. A ROUNDOFF ERROR ANALYSIS

A. BFP Roundoff Error Model

A numerical error analysis of the proposed method when implemented in finite precision requires an appropriate roundoff error model for the BFP scheme. However, conventional models like the additive roundoff error model for the FxP and relative roundoff error model for the FP are not directly applicable to the BFP representation, as, unlike the FxP scheme, the BFP is a scaled number representation system with distinct block exponents for different blocks. At the same time, unlike the FP system, the BFP is not a normalized data format. For our treatment, we adopt a *scaled additive roundoff error* model, proposed recently in [4]. In this model, the roundoff error $\alpha(n)$ associated with the quantization of $x(n)$ is given by

$$\begin{aligned}\alpha(n) &= Q[x(n)] - x(n)\\ &= (Q[\overline{x}(n)] - \overline{x}(n))\cdot 2^{\gamma_i} = \overline{\alpha}(n)\cdot 2^{\gamma_i}\end{aligned} \tag{14}$$

where $Q[.]$ denotes the quantized value of a quantity and $\overline{\alpha}(n)$ is the mantissa quantization error modeled as an uncorrelated random sequence. The block exponents $\gamma_i, i \in Z$ are also assumed to be uncorrelated. Then, under the usual rounding-to-nearest assumption, the roundoff error $\alpha(n)$ will have zero mean and variance given by

$$\sigma_\alpha^2 = \sigma_{\overline{\alpha}}^2 \cdot E[2^{2\gamma_i}] = \frac{2^{-2B}}{12}\cdot\sum_{l=1}^{N_\gamma} p_\gamma(\gamma_l) 2^{2\gamma_l} \tag{15}$$

where B is the number of bits used to represent each mantissa, $p_\gamma(\gamma_l)$, $l = 1, \ldots, N_\gamma$ is the *probability mass function* of the block exponent, and N_γ is the number of available distinct block exponent levels.

In practice, exact evaluation of $p_\gamma(\gamma_l)$ is very difficult as this requires knowledge of the joint probability density of the block variables. Instead, it is approximated using some marginal distributions, which provide acceptable results [4]. Assuming that the signal $x(n)$ is Gaussian and i.i.d. with variance σ_x^2, $p_\gamma(\gamma_l)$ is then approximated by

$$p_\gamma(\gamma_l) = \left[\operatorname{erf}\left(\frac{2^{\gamma_l - S_{\min}}}{\sqrt{2}\sigma_x}\right)\right]^N - \left[\operatorname{erf}\left(\frac{\frac{1}{2}2^{\gamma_l - S_{\min}}}{\sqrt{2}\sigma_x}\right)\right]^N \tag{16}$$

where $\operatorname{erf}(x)$ is the *error function*, i.e., $\operatorname{erf}(x) = (2/\sqrt{\pi})\int_0^x e^{-t^2}dt$.

B. Roundoff Error Analysis of the Proposed Scheme

In this section, we carry out a steady-state roundoff error analysis of the proposed realization by adopting the approach of [14] to the given context. We assume, as before, that the quantization is restricted to mantissas only and each datum mantissa as well as each filter coefficient mantissa are represented, respectively, by (B_d + one sign) bits and (B_c + one sign) bits in the usual two's complement form. Assuming that rounding is used in the quantization process, the corresponding quantization errors will then have the variances $\sigma_d^2 = 2^{-2B_d}/12$ and $\sigma_c^2 = 2^{-2B_c}/12$, respectively. We also follow the convention of denoting finite and infinite precision quantities by primed and unprimed variables, respectively.

Using the BFP roundoff error model, we then have the quantized input $x'(n)$ and desired response $d'(n)$, $n \in Z_i'$, given by $: x'(n) = \overline{x}'(n)\cdot 2^{\gamma_i} = (\overline{x}(n) + \overline{\alpha}(n))\cdot 2^{\gamma_i} = x(n) + \alpha(n)$ and $d'(n) = \overline{d}'(n)\cdot 2^{\gamma_i} = (\overline{d}(n) + \overline{\beta}_1(n))\cdot 2^{\gamma_i} = d(n) + \beta_1(n)$, where $\overline{\alpha}(n)$ and $\overline{\beta}_1(n)$ denote the respective mantissa quantization errors and $\alpha(n) = \overline{\alpha}(n)\cdot 2^{\gamma_i}$, $\beta_1(n) = \overline{\beta}_1(n)\cdot 2^{\gamma_i}$. Since the weight updating process is recursive in time, the computed weight vector $\mathbf{w}'(n)$ at the nth index would include the cumulative effect of the errors $\alpha(k)$, $\beta_1(k)$ and also the errors resulting from quantization involved in filtering and weight updating for all k up to $(n-1)$. As a result, under finite precision, the filter weight mantissa vector $\overline{\mathbf{w}}'(n)$ and exponent ψ_n' will, in general, be different from their infinite precision counterparts, thus giving rise to a weight error vector $\boldsymbol{\rho}(n)$, where $\mathbf{w}'(n) = \overline{\mathbf{w}}'(n)\cdot 2^{\psi_n'} = \mathbf{w}(n) + \boldsymbol{\rho}(n)$. Using this, the quantized filter output $y'(n)$ can be expressed as $y'(n) = \overline{y}'(n)\cdot 2^{\gamma_i + \psi_n'}$, where $\overline{y}'(n) = Q[\overline{\mathbf{w}}'^t(n)\overline{\mathbf{x}}'(n)] = \overline{\mathbf{w}}'^t(n)\overline{\mathbf{x}}'(n) + \overline{\eta}(n)$, with $\overline{\mathbf{x}}'(n) = [\overline{x}'(n), \overline{x}'(n-1), \ldots, \overline{x}'(n-L+1)]^t$. The error term $\overline{\eta}(n)$ takes into account the quantization errors arising from each of the L multiplications present in $\overline{\mathbf{w}}'^t(n)\overline{\mathbf{x}}'(n)$ and thus has variance $L{\sigma_d}^2$. Under finite precision, the output error is then given by $e'(n) = \overline{e}'(n)\cdot 2^{\gamma_i + \psi_n'}$, where $\overline{e}'(n) = Q[\overline{d}'(n)\cdot 2^{-\psi_n'}] - \overline{y}'(n) = \overline{d}'(n)\cdot 2^{-\psi_n'} + \overline{\beta}_2(n) - \overline{y}'(n)$ and $\overline{\beta}_2(n)$ is the so-called block denormalization error of variance σ_d^2, caused by right shifting $\overline{d}'(n)$ and then rounding. The four error terms $\overline{\alpha}(n)$, $\overline{\beta}_1(n)$, $\overline{\beta}_2(n)$, and $\overline{\eta}(n)$ are modeled as zero-mean, white sequences that are independent of each other and also of the signals, with the first three terms having variance of σ_d^2.

Our purpose here is to evaluate the output mean-square error (OMSE) $E[(e'(n))^2]$ and thus determine how the various quantization errors contribute to the OMSE in the steady state. From the above definitions of $\overline{e}'(n)$ and $\overline{y}'(n)$, we express $e'(n)$ as

$$\begin{aligned} e'(n) &= \overline{e}'(n)\cdot 2^{\gamma_i + \psi_n'} \\ &= d'(n) + \tau(n) - \mathbf{w}'^t(n)\mathbf{x}'(n), \end{aligned} \tag{17}$$

where $\tau(n) = \beta_2(n) - \eta(n)$, $\beta_2(n) = \overline{\beta}_2(n)\cdot 2^{\gamma_i + \psi_n'}$ and $\eta(n) = \overline{\eta}(n)\cdot 2^{\gamma_i + \psi_n'}$. Replacing $d'(n)$, $\mathbf{w}'(n)$, and $\mathbf{x}'(n)$ by $(d(n) + \beta_1(n))$, $(\mathbf{w}(n) + \boldsymbol{\rho}(n))$, and $(\mathbf{x}(n) + \boldsymbol{\alpha}(n))$, respectively, where $\boldsymbol{\alpha}(n) = [\alpha(n), \alpha(n-1), \ldots, \alpha(n-L+1)]^t$ and neglecting products between error terms, we can then write

$$e'(n) = e(n) + \beta_1(n) + \tau(n) - \mathbf{w}^t(n)\boldsymbol{\alpha}(n) - \boldsymbol{\rho}^t(n)\mathbf{x}(n). \tag{18}$$

Following the approach of [14] including the "independence assumption," the OMSE is then obtained as

$$\begin{aligned} E[(e'(n))^2] = {} & E[e^2(n)] + \sigma_d^2\epsilon_\gamma + \sigma_d^2(L+1)\epsilon_\gamma\upsilon_\psi \\ & + E[\mathbf{w}^t(n)\mathbf{w}(n)]\sigma_d^2\epsilon_\gamma + \operatorname{tr}[E(\boldsymbol{\rho}(n)\boldsymbol{\rho}^t(n))\mathbf{R}] \end{aligned} \tag{19}$$

where $\epsilon_\gamma = E(2^{2\gamma_i})$, $\upsilon_\psi = E(2^{2\psi_n'})$ and tr[.] denotes the trace of the matrix in the argument. In the steady state, as $n \to \infty$, $E[e^2(n)] \to \xi_{\min}(1 + (1/2)\mu\operatorname{tr}\mathbf{R})$ [12], where $\xi_{\min}$ is the minimum mean-square error corresponding to the optimal Wiener filter $\mathbf{w_o} = \mathbf{R}^{-1}\mathbf{p}$, $\mathbf{p} = E[\mathbf{x}(n)d(n)]$. In addition, from [14], we have $\lim_{n\to\infty} E[\mathbf{w}^t(n)\mathbf{w}(n)]\sigma_d^2\epsilon_\gamma = \sigma_d^2(|\mathbf{w_o}|^2 + (1/2)\mu L\xi_{\min})\epsilon_\gamma$. In order to obtain the steady-state value of $\operatorname{tr}[E(\boldsymbol{\rho}(n)\boldsymbol{\rho}^t(n))\mathbf{R}]$, we first consider the finite precision counterpart of (7), given by

$$\begin{aligned} \overline{\mathbf{v}}'(n) &= \overline{\mathbf{w}}'(n) + Q[\mu\overline{\mathbf{x}}'(n)\overline{e}'(n)\cdot 2^{2\gamma_i}] \\ &= \overline{\mathbf{w}}'(n) + \mu\overline{\mathbf{x}}'(n)\overline{e}'(n)\cdot 2^{2\gamma_i} + \overline{\boldsymbol{\sigma}}(n) \end{aligned} \tag{20}$$

where the noise vector $\overline{\boldsymbol{\sigma}}(n)$ depends primarily on the way the term $\mu\overline{\mathbf{x}}'(n)\overline{e}'(n)\cdot 2^{2\gamma_i}$ is computed. Since the usual trend is to employ longer registers to store the intermediate product values, we neglect the quantization errors arising in the intermediate multiplication operations. In that case, $\overline{\boldsymbol{\sigma}}(n)$ can be modeled as a zero-mean vector sequence with covariance matrix $\sigma_c^2\mathbf{I}$. Next, using (20) and recalling that $\mathbf{w}'(n+1)$ is always given by $\overline{\mathbf{v}}'(n)\cdot 2^{\psi_n'}$ irrespective of whether $\overline{\mathbf{w}}'(n+1)$ is $\overline{\mathbf{v}}'(n)$, or $(1/2)\overline{\mathbf{v}}'(n)$, we can write

$$\mathbf{w}'(n+1) = \mathbf{w}'(n) + \mu\mathbf{x}'(n)e'(n) + \boldsymbol{\sigma}(n)$$

with $\boldsymbol{\sigma}(n) = \overline{\boldsymbol{\sigma}}(n)\cdot 2^{\psi_n'}$. (Note that although division of $\overline{\mathbf{v}}'(n)$ by 2 may produce additional roundoff errors, we neglect them here, as in the steady state, ψ'_n remains constant most of the time, or, equivalently, the need to divide $\overline{\mathbf{v}}'(n)$ by 2 occurs very rarely.)

Replacing $\mathbf{w}'(n+1)$, $\mathbf{w}'(n)$, $\mathbf{x}'(n)$, and $\overline{e}'(n)$ by $\mathbf{w}(n+1)+\boldsymbol{\rho}(n+1)$, $\mathbf{w}(n)+\boldsymbol{\rho}(n)$, $\mathbf{x}(n)+\boldsymbol{\alpha}(n)$, and the expression given in (18), respectively, and ignoring products between error terms as before, we obtain

$$\boldsymbol{\rho}(n+1) = \mathbf{G}(n)\boldsymbol{\rho}(n) + \mathbf{h}(n) \tag{21}$$

where

$$\mathbf{G}(n) = \mathbf{I} - \mu\mathbf{x}(n)\mathbf{x}^t(n)$$

and

$$\begin{aligned}\mathbf{h}(n) = \mu[&\mathbf{x}(n)\beta_1(n) + \mathbf{x}(n)\tau(n)\\ &- \mathbf{x}(n)\mathbf{w}^t(n)\boldsymbol{\alpha}(n) + \boldsymbol{\alpha}(n)e(n)] + \boldsymbol{\sigma}(n).\end{aligned}$$

It is then rather straightforward [14] to obtain from (21) the steady-state value of $\mathrm{tr}[E(\boldsymbol{\rho}(n)\boldsymbol{\rho}^t(n))\mathbf{R}]$ as

$$\lim_{n\to\infty} \mathrm{tr}[E(\boldsymbol{\rho}(n)\boldsymbol{\rho}^t(n))\mathbf{R}] = \frac{\mathrm{tr}(\mathbf{Q})}{2\mu - \mu^2\mathrm{tr}\mathbf{R}} \tag{22}$$

where $\mathbf{Q}$ is the steady-state value of $E[\mathbf{h}(n)\mathbf{h}^t(n)]$ and is given by

$$\begin{aligned}\mathbf{Q} = \mu^2\Bigg[&\sigma_d^2(L+1)\epsilon_\gamma\upsilon_\psi\mathbf{R} + \sigma_d^2\epsilon_\gamma\xi_{\min}\left(1+\frac{1}{2}\mu\mathrm{tr}\mathbf{R}\right)\mathbf{I}\\ &+\sigma_d^2\epsilon_\gamma\left(|\mathbf{w_o}|^2+\frac{1}{2}\mu L\xi_{\min}\right)\mathbf{R}+\sigma_d^2\epsilon_\gamma\mathbf{R}\Bigg]+\sigma_c^2\upsilon_\psi\mathbf{I}.\end{aligned} \tag{23}$$

Substituting (23) in (22) and then (22) in (19), we obtain the steady-state value ξ of the OMSE as

$$\begin{aligned}\xi =& \xi_{\min}\left(1+\frac{1}{2}\mu\mathrm{tr}\mathbf{R}\right) + \sigma_d^2\epsilon_\gamma + \sigma_d^2(L+1)\epsilon_\gamma\upsilon_\psi\\ &+ \sigma_d^2\epsilon_\gamma\left(|\mathbf{w_o}|^2+\frac{1}{2}\mu L\xi_{\min}\right)\\ &+ \frac{\sigma_d^2.\epsilon_\gamma[(1+(L+1)\upsilon_\psi)\mathrm{tr}\mathbf{R} + L\xi_{\min}\left(1+\frac{1}{2}\mu\mathrm{tr}\mathbf{R}\right)]}{\frac{2}{\mu}-\mathrm{tr}\mathbf{R}}\\ &+ \frac{\sigma_d^2\cdot\epsilon_\gamma[(|\mathbf{w_o}|^2+\frac{1}{2}\mu L\xi_{\min})\mathrm{tr}\mathbf{R}]}{\frac{2}{\mu}-\mathrm{tr}\mathbf{R}} + \frac{L\sigma_c^2\upsilon_\psi}{2\mu-\mu^2\mathrm{tr}\mathbf{R}}.\end{aligned} \tag{24}$$

For small μ, the above can be approximated as

$$\begin{aligned}\xi =& \xi_{\min} + \sigma_d^2\epsilon_\gamma[1+L\upsilon_\psi+|\mathbf{w_o}|^2]\\ &+ \frac{1}{2}\mu\sigma_d^2\epsilon_\gamma\left[L\upsilon_\psi\mathrm{tr}\mathbf{R} + L\xi_{\min} + |\mathbf{w_o}|^2\mathrm{tr}\mathbf{R}\right]\\ &+ \frac{1}{2}\mu\xi_{\min}(L\epsilon_\gamma\sigma_d^2+\mathrm{tr}\mathbf{R}) + \frac{L\sigma_c^2\upsilon_\psi}{2\mu}.\end{aligned} \tag{25}$$

Optimal μ: An examination of (24) and also (25) reveals that in the steady state, the OMSE is proportional to μ both linearly and inversely—the former coming through the excess mean-square error and also through terms dictated by ${\sigma_d}^2$, i.e., data quantization error variance, while the latter is given by the last term in (24) and (25), which is governed by coefficient quantization noise. As a result, the OMSE increases with both decreasing as well as increasing μ and the optimal $\mu = \mu_{\mathrm{opt}}$ is obtained by minimizing ξ with regard to μ. This gives us

$$\begin{aligned}&\mu_{\mathrm{opt}} =\\ &\sqrt{\frac{L\upsilon_\psi\sigma_c^2}{(\xi_{\min}(L\epsilon_\gamma\sigma_d^2+\mathrm{tr}\mathbf{R})+\sigma_d^2\epsilon_\gamma[L\upsilon_\psi\mathrm{tr}\mathbf{R}+|\mathbf{w_o}|^2\mathrm{tr}\mathbf{R}+L\xi_{\min}])}}\end{aligned} \tag{26}$$

which can be approximated as

$$\mu_{\mathrm{opt}} \approx \frac{\sigma_c}{\sigma_d\sqrt{\epsilon_\gamma\mathrm{tr}\mathbf{R}}}\left[\frac{L\upsilon_\psi}{\frac{\xi_{\min}}{\sigma_d^2\epsilon_\gamma}+L\upsilon_\psi+|\mathbf{w_o}|^2}\right]^{0.5}. \tag{27}$$

It is desirable that μ_{opt} be less than the upper bound of μ given by (11). To satisfy this, we first note that $\mu_{\mathrm{opt}} < (\sigma_c/\sigma_d)(\mathrm{tr}\mathbf{R}\epsilon_\gamma)^{-0.5}$. Subjecting $(\sigma_c/\sigma_d)(\mathrm{tr}\mathbf{R}\epsilon_\gamma)^{-0.5}$ to be less than $2^{-2ex_{\max}}/(L+2)$, we obtain the following condition:

$$B_c > B_d + \left(2ex_{\max} + \frac{1}{2}\log_2 L - \log_2\sigma_x - \frac{1}{2}\log_2\epsilon_\gamma\right) \tag{28}$$

where we have replaced $\mathrm{tr}\mathbf{R}$ by $L\sigma_x^2$ and assumed $L \gg 2$. To give a numerical example from our simulation studies, consider a situation where $ex_{\max} = 1$, $\epsilon_\gamma \approx 3.25$ and $L = N = 4$. Then, (28) implies that B_c and B_d should be chosen to satisfy $B_c > B_d + 2.1498$.

V. Simulation Studies

The proposed scheme requires a stronger upper bound on μ than the well-known bound $2/\mathrm{tr}\mathbf{R}$ for algorithm convergence. In order to study the resulting slow-down effects on convergence speed, the proposed scheme was first simulated in finite precision taking μ equal to the upper bound (11) and then compared with a high level (infinite precision) LMS simulation that takes μ close to $2/\mathrm{tr}\mathbf{R}$. Two system identification problems were considered for this. For system A, the system output $y(n)$ was given by $y(n) = 0.7x(n)+0.65x(n-1)+0.25x(n-2)+\nu(n)$, with $x(n)$ and $\nu(n)$ representing the system input and the zero-mean observation noise (white), respectively, with the following variances : $\sigma_x^2 = 1$ and $\sigma_\nu^2 = 0.01$. The variance σ_y^2 of $y(n)$ was found to be 0.935. To calculate the upper bound of μ as given by (11), we first observe that $M_{\max} = \max\{|x(n)|, |y(n)||n \in Z\}$ can be safely taken as 1.99 $\max\{\sigma_x, \sigma_y\}$ so as to contain almost 95% of the samples of $x(n)$ and $y(n)$. This gives rise to $ex_{\max} = 1$ and with $L = 3$, an upper bound of 0.05 for μ.

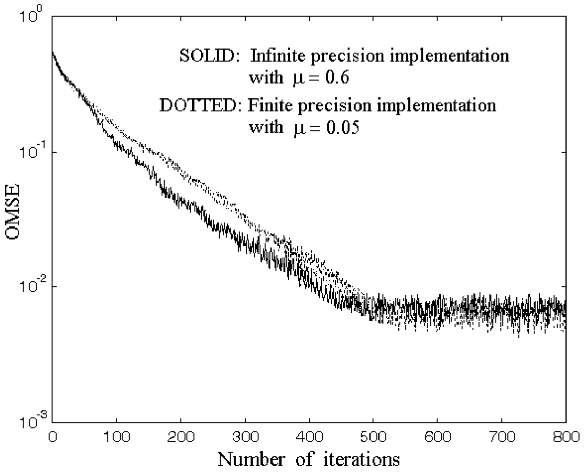

Fig. 1. Learning curves for the LMS algorithm for identifying system A, realized using the proposed BFP-based scheme under finite precision (with $\mu = 0.05$), and in infinite precision (with $\mu = 0.6$), shown by the dotted and the solid lines, respectively.

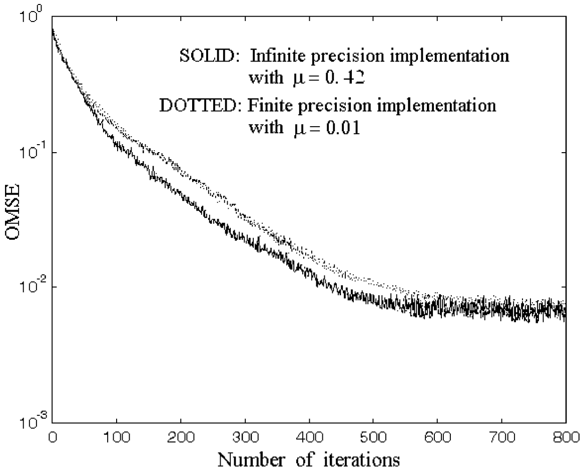

Fig. 2. Learning curves for the LMS algorithm for identifying system B, realized using the proposed BFP-based scheme under finite precision (with $\mu = 0.01$) and in infinite precision (with $\mu = 0.42$), shown by the dotted and the solid lines, respectively.

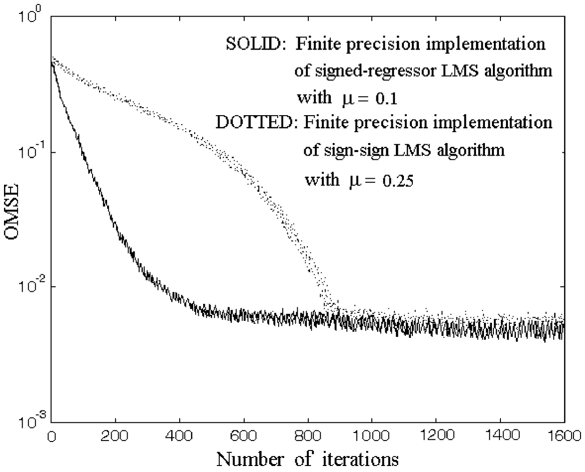

Fig. 3. Learning curves for the BFP-based signed-regressor LMS algorithm (with $\mu = 0.1$) and sign–sign LMS algorithm (with $\mu = 0.25$) for identifying system A, shown by the solid and the dotted lines, respectively.

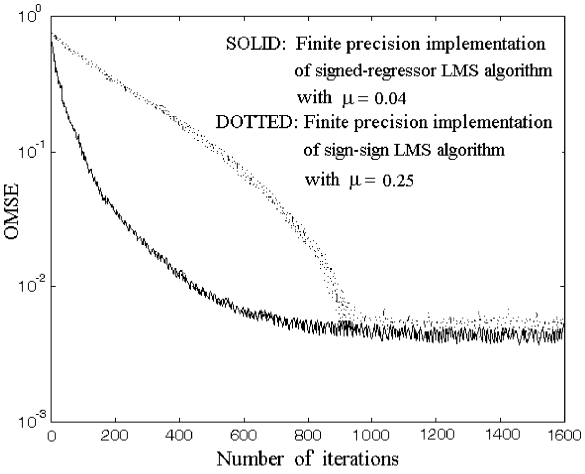

Fig. 4. Learning curves for the BFP-based signed-regressor LMS algorithm (with $\mu = 0.04$) and sign–sign LMS algorithm (with $\mu = 0.25$) for identifying system B, shown by the solid and the dotted lines, respectively.

Taking $\mu = 0.05$, the algorithm was first simulated in finite precision, choosing block length N as 4 and allocating 9 (i.e., $1 + 8$) bits for the data mantissa, 13 (i.e., $1 + 12$) bits for the filter coefficient mantissa, and 4 (i.e., $1+3$) bits for the exponent of both. The corresponding learning curve is obtained by plotting the OMSE versus number of iterations and is shown by the dotted line in Fig. 1. Fig. 1 also shows, by the solid line, the corresponding learning curve for an infinite-precision-based LMS simulation with $\mu = 0.6(\approx 2/\mathrm{tr}\mathbf{R})$. For system B, the system output $y(n)$ was given by $y(n) = x(n) + 0.7x(n-1) + 0.4x(n-2) + 0.5x(n-3) + \nu(n)$, with $x(n)$ and $\nu(n)$ remaining same as before. For this case, $L = 4$ and with σ_y^2 found as 1.92, we have $ex_{\max} = 2$ and the upper bound (11) as 0.01. Choosing the same block length and also the same precision for the coefficient as well as data mantissa and exponents as before, the proposed scheme was simulated under finite precision for $\mu = 0.01$. The learning curves for this and also for an infinite-precision-based realization of the LMS algorithm with $\mu = 0.42(\approx 2/\mathrm{tr}\mathbf{R})$ are shown, respectively, by the dotted and the solid lines in Fig. 2. It is clear from both Figs. 1 and 2 that the speed of convergence of the proposed scheme is less than the corresponding infinite-precision-based LMS, which is expected, as the step size chosen for the former is considerably less than that chosen for the latter. However, for all practical purposes, the degradation in convergence speed is clearly seen to be within acceptable limits. Simulation results for the proposed BFP-based signed-regressor LMS and sign–sign LMS algorithms are also shown, respectively, by the solid and the dotted lines in Fig. 3 (for system A) and Fig. 4 (for system B). The corresponding learning curves and speeds of convergence conform to their usual characteristics [13].

VI. Conclusion

This paper presents an efficient realization of the LMS algorithm using the block floating-point (BFP) data representation system. The proposed scheme can process data over a wide dynamic range at a processing cost comparable to fixed-point (FxP)-based processing. For this, appropriate BFP formats for the data and the filter coefficients are adopted, and the LMS algorithm is recast in terms of the chosen formats. The proposed

BFP-LMS algorithm relies on FxP arithmetic only and enjoys a speed-up of about two and a half over a fixed-point (FP)-based realization. Further speed-up can be achieved by considering extensions of the proposed approach to the sign-LMS family. Finally, a steady-state roundoff error analysis is carried out, and an optimum step size that minimizes the quantization noise component in the output mean-square error is derived.

REFERENCES

[1] J. H. Wilkinson, *Rounding Errors in Algebraic Processes*. Englewood Cliffs, NJ: Prentice-Hall, 1963.

[2] K. R. Ralev and P. H. Bauer, “Realization of block floating point digital filters and application to block implementations,” *IEEE Trans. Signal Process.*, vol. 47, no. 4, pp. 1076–1086, Apr. 1999.

[3] S. Sridharan, “Implementation of state space digital filters using block floating point arithmetic,” in *Proc. 1987 IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, Dallas, TX, 1987, pp. 908–911.

[4] K. Kalliojärvi and J. Astola, “Roundoff errors in block-floating-point systems,” *IEEE Trans. Signal Process.*, vol. 44, no. 4, pp. 783–790, Apr. 1996.

[5] S. Sridharan and G. Dickman, “Block floating point implementation of digital filters using the DSP56000,” *Microprocess. Microsyst.*, vol. 12, no. 6, pp. 299–308, Jul.–Aug. 1988.

[6] S. Sridharan and D. Williamson, “Implementation of high order direct form digital filter structures,” *IEEE Trans. Circuits Syst.*, vol. CAS-33, no. 8, pp. 818–822, Aug. 1986.

[7] A. V. Oppenheim, “Realization of digital filters using block floating point arithmetic,” *IEEE Trans. Audio Electroacoust.*, vol. AE-18, no. 2, pp. 130–136, Jun. 1970.

[8] F. J. Taylor, “Block floating point distributed filters,” *IEEE Trans. Circuits Syst.*, vol. CAS-31, no. 3, pp. 300–304, Mar. 1984.

[9] A. Erickson and B. Fagin, “Calculating FHT in hardware,” *IEEE Trans. Signal Process.*, vol. 40, no. 6, pp. 1341–1353, Jun. 1992.

[10] A. V. Oppenheim and C. Weinstein, “Effects of finite register length in digital filtering and the fast Fourier transform,” *Proc. IEEE*, vol. 60, pp. 957–976, Aug. 1972.

[11] P. H. Bauer, “Absolute error bounds for block floating point direct form digital filters,” *IEEE Trans. Signal Process.*, vol. 43, no. 8, pp. 1994–1996, Aug. 1995.

[12] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1986.

[13] B. Farhang-Boroujeny, *Adaptive Filters—Theory and Application*. Chichester, U.K.: Wiley, 1998.

[14] C. Caraiscos and B. Liu, “A roundoff error analysis of the LMS adaptive algorithm,” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-32, no. 1, pp. 34–41, Feb. 1984.

Abhijit Mitra (S’03–M’04) was born in Serampore, India, in 1975. He received the B.E. (Hons.) degree from R. E. College, Durgapur, India, in 1997, the M.E.Tel.E. degree from Jadavpur University, India, in 1999, and the Ph.D. degree from the Indian Institute of Technology, Kharagpur, India, in 2004, all in electronics and communication engineering.

Since 2004, he has been an Assistant Professor with the Indian Institute of Technology, Guwahati, India. His research interests include finite-wordlength digital signal processing, statistical signal processing, adaptive signal processing, and wireless communications.

Mrityunjoy Chakraborty (S’91–M’95–SM’99) received the B.Eng. degree in electronics and telecom engineering from Jadavpur University, Calcutta, India, in 1983, the M.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, in 1985, and the Ph.D. degree in electrical engineering from the Indian Institute of Technology, Delhi, in 1994.

Since 1994, he has been a Faculty Member in the Department of Electronics and Electrical Communication Engineering at the Indian Institute of Technology, Kharagpur, where he is now a Professor. He has visited many universities overseas as Visiting Professor/Scholar, including Kyoto University, Japan; the National University of Singapore; and Chonbuk National University, South Korea. His teaching and research interests are in digital and adaptive signal processing, including very large scale integration (VLSI) for signal processing and application of signal processing to communications.

Dr. Chakraborty is presently an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS I.

Hideaki Sakai (M’78–SM’02) received the B.E. and Dr.Eng. degrees in applied mathematics and physics from Kyoto University, Kyoto, Japan, in 1972 and 1981, respectively.

From 1975 to 1978, he was with Tokushima University, Tokushima, Japan. He spent six months from 1987 to 1988 at Stanford University, CA, as a Visiting Scholar. He is currently a Professor in the Department of Systems Science, Graduate School of Informatics, Kyoto University. His research interests are in the area of statistical and adaptive signal processing.

Dr. Sakai was an Associate Editor of *IEICE Transactions Fundamentals of Electronics, Communications, and Computer Sciences* from 1996 to 2000 and the IEEE TRANSACTIONS ON SIGNAL PROCESSING from 1999 to 2001. He is currently on the Editorial Board of the EURASIP *Journal of Applied Signal Processing*.