A Block Floating-Point Realization of the Gradient Adaptive Lattice Filter

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

***Abstract*—We present a novel scheme to implement the gradient adaptive lattice (GAL) algorithm using block floating point (BFP) arithmetic that permits processing of data over a wide dynamic range at a cost significantly less than that of a floating point (FP) processor. Appropriate formats for the input data, the prediction errors, and the reflection coefficients are adopted, taking care so that for the prediction errors and the reflection coefficients, they remain invariant to the respective order and time update processes. Care is also taken to prevent overflow during prediction error computation and reflection coefficient updating by using an appropriate exponent assignment algorithm and an upper bound on the step-size mantissa.**

***Index Terms*—Block floating-point arithmetic, exponent assignment, gradient adaptive lattice.**

I. Introduction

THE block floating-point (BFP) data format is a useful compromise between floating-point (FP) and fixed-point (FxP) schemes. In this format, the incoming data are partitioned into nonoverlapping blocks, and depending upon 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 FxP-like computation within every block, thereby enabling the user to handle data over a wide dynamic range with temporal and/or spatial complexities comparable to that of FxP-based systems. In recent years, the BFP format has been used successfully for efficient realization of several forms of digital filters [1], [2], [5], [6]. Some studies [2], [3] have also been made to investigate the associated numerical error behavior. Such efforts have, however, remained confined to the case of fixed coefficient digital filters only and were not extended to include adaptive filters that present more complex structures, including error feedback. A BFP treatment to adaptive filters faces certain difficulties not encountered in the fixed coefficient case, namely, a) 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, and b) 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.

Recently, the BFP concept has been used to obtain an efficient realization of the normalized least-mean-square-based transversal adaptive filter [7]. In this letter, we consider a similar BFP treatment for efficient realization of the gradient adaptive lattice algorithm [8]. However, as the lattice, unlike its transversal counterpart, is an order-recursive structure, new approaches are required to achieve this. This includes adoption of appropriate formats for representing the input data, the prediction errors, and the reflection coefficients, taking care so that for the prediction errors and the reflection coefficients, the chosen formats remain invariant to the respective order and time-update processes. Special arrangements are also made to prevent overflow during both prediction error computation and reflection coefficient updating—the former achieved via an appropriate exponent assignment algorithm and the latter by using a scaled representation for the step size with mantissa restricted to remain below a certain upper bound. The proposed scheme employs mostly FxP-type operations and can achieve considerable speed up over a FP-based realization.

II. Proposed Implementation

For an Lth-order linear prediction lattice filter, the pth stage $p \in Z_L = \{1, \ldots, L\}$ is characterized by the following order update equations:

$$f_p(n) = f_{p-1}(n) - k_p(n) b_{p-1}(n-1) \tag{1}$$

$$b_p(n) = b_{p-1}(n-1) - k_p(n) f_{p-1}(n) \tag{2}$$

where $f_p(n)$ and $b_p(n)$ are the pth-order forward prediction error (FPE) and backward prediction error (BPE), respectively, and $k_p(n)$ is the time-dependent reflection coefficient corresponding to the pth stage. The reflection coefficients are updated in time using a gradient-based approach, and there exist several forms of such update relationships [9]. For our treatment, we consider the simple and popular case of unnormalized lattice recursion, for which the time update relationship for $k_p(n)$ is given by

$$k_p(n+1) = k_p(n) + \mu_p (f_p(n) b_{p-1}(n-1) + b_p(n) f_{p-1}(n)) \tag{3}$$

where μ_p denotes the step size for the pth stage and should be chosen sufficiently small to guarantee convergence of the algorithm [8]. The order recursions (1) and (2) are initialized with the following zeroth-order FPE and BPE values: $f_0(n) = b_0(n) = x(n)$.

Manuscript received November 24, 2003; revised February 18, 2004. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Steven L. Grant.

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

A. Mitra was with the Department of Electronics and Electrical Communication Engineering, Indian Institute of Technology (IIT), Kharagpur, India. He is now with the Department of Electronics and Communication Engineering, Indian Institute of Technology (IIT), Guwahati, India (e-mail: a.mitra@iitg.ernet.in).

Digital Object Identifier 10.1109/LSP.2005.843781

The proposed scheme, which tries to develop appropriate BFP-based realizations of (1)–(3), employs three simultaneous BFP representations—one for the reflection coefficients $k_p(n)$, $p \in Z_L$, one for the given data $x(n)$, and another for the FPE and the BPE, i.e., $f_p(n)$ and $b_p(n)$. These are described as follows.

A. Format for the Reflection Coefficients

The proposed approach adopts a scaled representation for the reflection coefficients, as given by

$$k_p(n) = \overline{k}_p(n).2^{\phi_p(n)} \tag{4}$$

where $\overline{k}_p(n)$ and $\phi_p(n)$ are, respectively, the time-varying mantissa and exponent that are updated at each index n, with the latter chosen to ensure that $|\overline{k}_p(n)| < 1/2$, $n \in Z^+ = \{0, 1, 2, \ldots\}$, $p \in Z_L$. As shown later, $\phi_p(n)$, in our treatment, is a nondecreasing function of n. Further, the initial values of $\overline{k}_p(n)$ and $\phi_p(n)$ are taken as $\overline{k}_p(0) = 0$ and $\phi_p(0) = 1$, respectively.

B. BFP Representation of the Input Data

In this treatment, the input data $x(n)$ is partitioned into nonoverlapping blocks of N samples each ($N \geq L$) with the ith block ($i \in Z^+$) consisting of $x(n)$ for $n \in Z_i' = \{iN, iN+1, \ldots, iN+N-1\}$. Further, the data samples of each block are scaled jointly by a common factor to have a uniform BFP representation. This means that for each $n \in Z_i'$, $x(n)$ is expressed as

$$x(n) = \overline{x}(n).2^{\gamma_i} \tag{5}$$

where $\overline{x}(n)$ $(= x(n).2^{-\gamma_i})$ represents the mantissa of $x(n)$, and γ_i is the common block exponent for the ith block, chosen to satisfy $\gamma_i \geq ex_i$, where $ex_i = \lfloor \log_2 M_i \rfloor + 1$ and $M_i = \max\{|x(n)| \mid n \in Z_i'\}$, i.e., ex_i is the exponent of the data sample with the highest magnitude in the ith block. For such choice of γ_i, $|\overline{x}(n)| < 1$ for all $n \in Z_i'$. The block exponent γ_i is actually assigned as per an algorithm described later.

C. BFP Format of the Prediction Errors

The FPE $f_p(n)$ and the BPE $b_p(n)$, $p \in Z_L$ are expressed in the following format: $f_p(n) = \overline{f}_p(n).2^{\Gamma_{p,n}}$ and $b_p(n) = \overline{b}_p(n).2^{\Gamma_{p,n}}$, respectively, where the exponent $\Gamma_{p,n}$ is defined as

$$\Gamma_{p,n} = \gamma_i + \sum_{j=1}^{p} \phi_j(n) \tag{6}$$

and is updated recursively in time, as shown later (for $p = 0$, $\Gamma_{0,n} = \gamma_i$, and $\overline{f}_0(n) = \overline{b}_0(n) = \overline{x}(n)$). Next, from (1) and (2), it is seen that the computations at the pth stage of the lattice involve two exponents: $\Gamma_{p-1,n}$ and $\Gamma_{p-1,n-1}$, the former coming from $f_{p-1}(n)$ and the latter from $b_{p-1}(n-1)$. In order to have a common exponent, we next rescale the delayed BPE mantissas $\overline{b}_{p-1}(n-1)$, $p \in Z_L$ to $\tilde{b}_{p-1}(n-1)$, given by

$$\tilde{b}_{p-1}(n-1) = \overline{b}_{p-1}(n-1).2^{-\alpha_{p-1,n}} \tag{7}$$

where the update factor $\alpha_{p-1,n} (= \Gamma_{p-1,n} - \Gamma_{p-1,n-1})$ is defined as

$$\alpha_{p-1,n} = \begin{cases} \theta_n + \sum_{j=1}^{p-1} \Delta\phi_j(n), & p \geq 2 \\ \theta_n, & p = 1 \end{cases} \tag{8}$$

where $\Delta\phi_j(n) = (\phi_j(n) - \phi_j(n-1))$ and θ_n takes the value $\Delta\gamma_i = \gamma_i - \gamma_{i-1}$ at $n = iN$, i.e., at the starting index of the ith block, $i \in Z^+$ and *zero* otherwise. In practice, such rescaling is realized by passing each of the BPE mantissas $\overline{b}_{p-1}(n-1)$, $p \in Z_L$ through a rescaling unit that applies $\alpha_{p-1,n}$ number of right or left shifts on $\overline{b}_{p-1}(n-1)$, depending on whether $\alpha_{p-1,n}$ is positive or negative, respectively. The parameter $\alpha_{p-1,n}$ is updated order recursively as

$$\alpha_{p,n} = \alpha_{p-1,n} + \Delta\phi_p(n)$$

and using $\alpha_{p,n}$, $\Gamma_{p,n}$ is computed time recursively as

$$\Gamma_{p,n} = \Gamma_{p,n-1} + \alpha_{p,n}, p \in Z_L.$$

For the above description of $\Gamma_{p,n}$ and $\alpha_{p,n}$, the FPE and BPE mantissas can now be written as

$$\overline{f}_p(n) = \overline{f}_{p-1}(n).2^{-\phi_p(n)} - \overline{k}_p(n)\tilde{b}_{p-1}(n-1) \tag{9}$$

$$\overline{b}_p(n) = \tilde{b}_{p-1}(n-1).2^{-\phi_p(n)} - \overline{k}_p(n)\overline{f}_{p-1}(n). \tag{10}$$

Both the order update equations (9) and (10) above are based on FxP operations, and thus, it is required to ensure that no overflow arises during these computations. For this, we first consider $\tilde{b}_{p-1}(n-1)$, as given in (7). Noting that although in our treatment, $\Delta\phi_p(n) \geq 0$, since $\phi_p(n)$, $p \in Z_L$ is a nondecreasing function of n, as stated earlier, $\Delta\gamma_i$ can, however, be positive as well as negative and with negative $\Delta\gamma_i$, $\alpha_{p-1,n}$ in (8) can become negative at $n = iN$, thus giving rise to left shift operation on $\overline{b}_{p-1}(n-1)$ in (7) and to the possibility of overflow as a consequence. To ensure no overflow in $\tilde{b}_{p-1}(n-1)$, we need to maintain $|\tilde{b}_{p-1}(n-1)| < 1$. We meet this condition by first proposing an exponent assignment algorithm as follows.

Algorithm: *For any ith block ($i \in Z^+$),*
if $ex_i \geq ex_{i-1}$,
choose $\gamma_i = ex_i$
else (i.e., $ex_i < ex_{i-1}$)
choose $\gamma_i = ex_{i-1}$.

Note that when $ex_i \geq ex_{i-1}$, we can either have $\gamma_{i-1} > ex_i \geq ex_{i-1}$ (Case A), implying $\Delta\gamma_i < 0$, or $ex_i \geq \gamma_{i-1} \geq ex_{i-1}$ (Case B), meaning $\Delta\gamma_i \geq 0$. However, for $ex_i < ex_{i-1}$ (Case C), we always have $\Delta\gamma_i \leq 0$. It is, however, easy to see that for all $n \in Z_{i-1}'$, $|\overline{f}_0(n).2^{-\Delta\gamma_i}| = |\overline{b}_0(n).2^{-\Delta\gamma_i}| = |\overline{x}(n).2^{-\Delta\gamma_i}| < 1$, irrespective of whether $\Delta\gamma_i$ is positive or negative, as rescaling $\overline{f}_0(n)$ and $\overline{b}_0(n)$ by $2^{-\Delta\gamma_i}$ amounts to changing their exponent from γ_{i-1} to γ_i and from above $\gamma_i \geq ex_{i-1}$.

Proposition 1: Given $n = iN$, $i \in Z^+$ and block length $N \geq$ filter order L, $|2^{-\Delta\gamma_i}.\overline{b}_{p-1}(n-r)| < 1$ and

$|2^{-\Delta\gamma_i}.\overline{f}_{p-1}(n-r)| < 1$ for $r = 1, 2, \ldots L-p+1$ and for all $p \in Z_L$.

Proof: For $p = 1$, the inequalities are satisfied trivially from above. Suppose that the inequalities are satisfied for p up to some q, $1 \le q \le L-1$. Then, for $p = q+1$ and for $n = iN$, $r = 1, 2, \ldots, L-q$, we can write from (10) and (7)

$$\left|2^{-\Delta\gamma_i}.\overline{b}_q(n-r)\right| = 2^{-\Delta\gamma_i}.\left|(\overline{b}_{q-1}(n-r-1).2^{-\alpha_{q-1,n-r}} .2^{-\phi_q(n-r)} - \overline{k}_q(n-r)\overline{f}_{q-1}(n-r))\right|.$$

For the stated choice of n and r, $\theta_{n-r} = 0$ and, thus, from (8), $\alpha_{q-1,n-r} \ge 0$. Using this, applying the triangle inequality to the right-hand side (RHS) of the above equation, and making use of the assumptions on $\overline{b}_{q-1}(n-r-1)$ and $\overline{f}_{q-1}(n-r)$, we have

$$\left|2^{-\Delta\gamma_i}.\overline{b}_q(n-r)\right| < 2^{-\phi_q(n-r)} + \left|\overline{k}_q(n-r)\right| < 1$$

since the minimum value of $\phi_q(n-r)$ is one and $|\overline{k}_q(n-r)| < 1/2$. In a similar way, it can be shown that for $n = iN$ and $r = 1, 2, \ldots, L-q$, $|2^{-\Delta\gamma_i}.\overline{f}_q(n-r)| < 1$. Hence, the Proposition is proved. ∎

Proposition 2: For $p \in Z_L$, $|\overline{f}_p(n)| < 1$, $|\overline{b}_p(n)| < 1$, and $|\tilde{b}_{p-1}(n-1)| < 1$.

Proof: It is enough to prove the above for a block, i.e., for $n \in Z_i'$, $i \in Z^+$. We prove this by induction. Assume the following are given: i) $|\tilde{b}_{p-1}(n-1)| < 1$, $p \in Z_L$ for some $n \in Z_i'$ (Note from Proposition 1 that this is already satisfied for $n = iN$, since at $n = iN$, $|\tilde{b}_{p-1}(n-1)| \le |2^{-\Delta\gamma_i}.\overline{b}_{p-1}(n-1)| < 1$), and ii) $|\overline{f}_p(n)| < 1$ for p up to some r, $0 \le r \le L-1$. For $r = 0$, $|\overline{f}_0(n)| = |\overline{x}(n)| < 1$ is always satisfied. For $p = r+1$, we have, from (9), $|\overline{f}_{r+1}(n)| \le |\overline{f}_r(n)|.2^{-\phi_{r+1}(n)} + |\overline{k}_{r+1}(n)||\tilde{b}_r(n-1)| < 1$, since, as stated earlier, $|\overline{k}_{r+1}(n)| < 1/2$ and $\phi_{r+1}(n) \ge 1$, meaning that $2^{-\phi_{r+1}(n)} \le 1/2$. In a similar manner, it can be shown from (10) that $|\overline{b}_{r+1}(n)| < 1$. For the $(n+1)$th index, if $n = iN+N-1$, i.e., if $n+1$ is the starting index of the $(i+1)$th block, then, from Proposition 1, $|\tilde{b}_{p-1}(n)| < 1$, $p \in Z_L$. If, however, $n < iN+N-1$, then, for $p \ge 2$, from (7) and (8), $|\tilde{b}_{p-1}(n)| = |\overline{b}_{p-1}(n)|.2^{-\sum_{j=1}^{p-1}\Delta\phi_j(n+1)} < 1$. For $p = 1$, $|\tilde{b}_{p-1}(n)| = |\overline{b}_{p-1}(n)| = |\overline{x}(n)| < 1$. Hence, the Proposition is proved. ∎

From Proposition 2, it is, thus, clear that there will be no overflow in $\tilde{b}_{p-1}(n)$, as computed via (7), and in $\overline{f}_p(n)$ and $\overline{b}_p(n)$, as given by (9) and (10), respectively. For the above descriptions of $f_p(n)$, $b_p(n)$, $f_{p-1}(n)$, and $b_{p-1}(n-1)$, the reflection coefficient update (3) can now be written as $k_p(n+1) = \overline{v}_p(n).2^{\phi_p(n)}$, where

$$\overline{v}_p(n) = \overline{k}_p(n) + \overline{\mu}_{p,n}[\overline{f}_p(n)\tilde{b}_{p-1}(n-1) + \overline{f}_{p-1}(n)\overline{b}_p(n)] \tag{11}$$

where $\overline{\mu}_{p,n} = \mu_p.2^{2\Gamma_{p-1,n}}$. In other words, the proposed approach adopts a scaled representation for μ_p with $\overline{\mu}_{p,n}$ and $-2\Gamma_{p-1,n}$ denoting, respectively, the values of the time-dependent mantissa and exponent at the nth index.

To satisfy $|\overline{k}_p(n+1)| < 1/2$ for $p \in Z_L$, we first limit each $\overline{v}_p(n)$ to satisfy $|\overline{v}_p(n)| < 1$, $p \in Z_L$. Then, if each $\overline{v}_p(n)$ happens to be lying within $\pm 1/2$, we make the assignments: $\overline{k}_p(n+1) = \overline{v}_p(n)$, $\phi_p(n+1) = \phi_p(n)$. Otherwise, we scale down $\overline{v}_p(n)$ by 2, in which case $\overline{k}_p(n+1) = 1/2\overline{v}_p(n)$, $\phi_p(n+1) = \phi_p(n) + 1$. Since each $|\overline{k}_p(n)| < 1/2$ for $p \in Z_L$, from (11), it is sufficient to have $\overline{\mu}_{p,n}[|\overline{f}_p(n)||\tilde{b}_{p-1}(n-1)| + |\overline{f}_{p-1}(n)||\overline{b}_p(n)|] \le 1/2$ in order to satisfy the relation $|\overline{v}_p(n)| < 1$, $p \in Z_L$. Since $\overline{f}_p(n)$, $\overline{b}_p(n)$, $\overline{f}_{p-1}(n)$, and $\tilde{b}_{p-1}(n-1)$ have magnitudes less than one each, this then results in the following global upper bound: $\overline{\mu}_{p,n} \le 1/4$, $p \in Z_L$.

Next, we evaluate $\overline{\mu}_{p,n}$ time recursively as

$$\overline{\mu}_{p,n} = \overline{\mu}_{p,n-1}.2^{2\alpha_{p-1,n}} \tag{12}$$

which follows from noting that

$$\overline{\mu}_{p,n}.2^{-2\Gamma_{p-1,n}} = \mu_p = \overline{\mu}_{p,n-1}.2^{-2\Gamma_{p-1,n-1}}$$

and that $\Gamma_{p,n} = \Gamma_{p,n-1} + \alpha_{p,n}$. Finally, to ensure no overflow in $\overline{\mu}_{p,n}$ due to the left shift of $\overline{\mu}_{p,n-1}$ and also to prevent the stalling of the weight updating process that may occur when $\overline{\mu}_{p,n}$ becomes zero due to the right shift of $\overline{\mu}_{p,n-1}$ under finite

TABLE I
SUMMARY OF THE GAL ALGORITHM REALIZED IN BFP FORMAT

1. Initial Conditions (for $0 \le p \le L$):
   $\overline{b}_p(n) = \Gamma_{p,n} = \phi_p(n) = 0$ for $n < 0$; $\phi_p(0) = 1, \overline{k}_p(0) = 0$.
   Initial value of block index $i = 0$ and $\gamma_{-1} = 0$.
2. Preprocessing:
   Using the input data $x(n)$ for the i-th block
   (stored during the processing of the $(i-1)$-th block),
(a) Evaluate (i) γ_i as per the Algorithm of Section II,
   (ii) $\Delta\gamma_i = (\gamma_i - \gamma_{i-1})$,
(b) Express $x(n)$ as $x(n) = \overline{x}(n).2^{\gamma_i}$, $n \in Z_i'$.
3. Processing for the i-th block:
   For $n = iN$ *to* $iN+N-1$ (i.e., for $n \in Z_i'$)
(i) Processing for the zero-th stage:
   (a) $\overline{f}_0(n) = \overline{b}_0(n) = \overline{x}(n)$ and $\Gamma_{0,n} = \gamma_i$,
   (b) $\alpha_{0,n} = \theta_n = \Delta\gamma_i$ for $n = iN$,
   $\alpha_{0,n} = \theta_n = 0$ otherwise.
(ii) Processing for the p-th stage ($p > 0$):
   For $p = 1$ *to* L (i.e., for $p \in Z_L$)
   (a) Step size mantissa updating:
   $\overline{\mu}_{p,n} = \overline{\mu}_{p,n-1}.2^{2\alpha_{p-1,n}}$,
   (b) Rescaling:
   $\tilde{b}_{p-1}(n-1) = \overline{b}_{p-1}(n-1).2^{-\alpha_{p-1,n}}$,
   (c) Prediction error mantissa computation:
   $\overline{f}_p(n) = \overline{f}_{p-1}(n).2^{-\phi_p(n)} - \overline{k}_p(n)\tilde{b}_{p-1}(n-1)$,
   $\overline{b}_p(n) = \tilde{b}_{p-1}(n-1).2^{-\phi_p(n)} - \overline{k}_p(n)\overline{f}_{p-1}(n)$.
   (d) Exponent updating:
   • $\alpha_{p,n} = \alpha_{p-1,n} + \Delta\phi_p(n)$,
   (where, $\Delta\phi_p(n) = \phi_p(n) - \phi_p(n-1)$)
   • $\Gamma_{p,n} = \Gamma_{p,n-1} + \alpha_{p,n}$.
   (e) Reflection coefficient updating:
   $\overline{v}_p(n) = \overline{k}_p(n) + \overline{\mu}_{p,n}[\overline{f}_p(n)\tilde{b}_{p-1}(n-1) + \overline{f}_{p-1}(n)\overline{b}_p(n)]$.
   If $|\overline{v}_p(n)| < \frac{1}{2}$ for any $p \in Z_L$
   then
   $\overline{k}_p(n+1) = \overline{v}_p(n)$, $\phi_p(n+1) = \phi_p(n)$,
   else
   $\overline{k}_p(n+1) = \frac{1}{2}\overline{v}_p(n)$, $\phi_p(n+1) = \phi_p(n) + 1$.
   end.
   end.
   Put $i = i+1$ and repeat steps 2 to 3.

TABLE II
COMPARISON BETWEEN THE BFP AND THE FP-BASED REALIZATIONS OF THE GAL FILTER. NUMBER OF OPERATIONS REQUIRED PER ITERATION FOR (a) REFLECTION COEFFICIENT UPDATING AND (b) COMPUTATION OF FPE AND BPE ARE GIVEN. UNLESS SPECIFIED OTHERWISE, ALL THE GENERAL OPERATIONS INDICATE MANTISSA OPERATIONS. [$*$ THE EXTRA OPERATIONS THAT MAY BE NEEDED AT THE STARTING INDEX OF EACH BLOCK ARE INDICATED IN THE PARENTHESIS]

(a)	MAC	Shift	Magnitude Check	Exponent Comparison	Exponent Addition
BFP	$3L$	Nil($+L^*$)	L	Nil	Nil
FP	$3L$	$6L$	Nil	$3L$	$6L$

(b)	MAC	Shift	Exponent Comparison	Exponent Addition
BFP	$2L$	$2L(+L^*)$	Nil	Nil($+L^*$)
FP	$2L$	$4L$	$2L$	$4L$

precision, an appropriately long register may be used to represent $\overline{\mu}_{p,n}$, and the initial value of $\overline{\mu}_{p,n}$ may be assigned by placing a binary 1 in the central bit location, with other bits assigned binary 0.

It is also easily seen from above that $\phi_p(n)$ is a nondecreasing function of n, and it saturates at a steady state value Φ_p, given by the peak value K_p of the $|k_p(n)|$-versus-n trajectory as $\Phi_p = \lceil \log_2 K_p \rceil + 1, K_p = \max\{|k_p(n)| \; | n = 0, 1, 2, \ldots\}$. This implies that in the steady state, at all indices except for the starting index of each block, $\alpha_{p,n} = 0, 0 \leq p \leq L$ and, thus, the two operations, namely, i) updating of $\Gamma_{p,n-1}$, $\alpha_{p-1,n}$, and $\overline{\mu}_{p,n-1}$, and ii) rescaling of $\overline{b}_{p-1}(n-1)$ are no longer required.

III. DISCUSSIONS AND CONCLUSIONS

The BFP-based implementation of the gradient adaptive lattice (GAL) algorithm, as proposed above and summarized in Table I, relies mostly on FxP arithmetic and, thus, enjoys less processing time than its FP-based counterpart. For example, to compute the FPE and the BPE for all the L stages in the steady state (i.e., after $\phi_p(n)$, $p \in Z_L$ attain their saturation values), the proposed scheme requires a total of $2L$ "Multiply-Accumulate" (MAC) operations (FxP) and $2L$ shifts (assuming availability of single-cycle barrel shifters; add to this an additional L shifts and L exponent additions that may be necessary at the starting index of each block). In contrast, in a FP-based realization, this would require the following *additional* operations: a) $2L$ exponent comparisons, b) $2L$ shifts, and c) $4L$ exponent additions. Similar advantages exist in reflection coefficient updating also. In both cases, the number of additional operations required under FP-based realization increases linearly with the order L of the lattice. Table II provides a comparative account of the two approaches in terms of the number of operations required. It is easily seen from this table that given a fixed-point processor with single-cycle MAC and barrel shifter units, the proposed scheme is about *four times faster* than a FP-based implementation. The storage requirement in the FP scheme is also higher, as it needs to store several exponent values at each index.

The proposed algorithm has also been simulated in finite precision to study the effects of the constrained initial choice of $\overline{\mu}_{p,n}, p \in Z_L$ (which also defines μ_p) on the convergence speed. A second-order autoregressive signal $x(n)$ was generated as $x(n) = 1.55x(n-1) - 0.8x(n-2) + z(n)$, with $z(n)$ being a zero-mean, white input of variance 0.02. Block formatting of $x(n)$ was carried out for a block length of 10. The signed mantissas for each data sample within every block and also for each reflection coefficient, FPE and BPE were represented using 12 $(11+1)$ bits, while 4 $(3+1)$ bits were allocated to represent the respective exponents of each. The algorithm was simulated with $L = 2$ and $\mu_p = 2^{-6}$, $p = 1, 2$. The simulation results are displayed by plotting $k_1(n)$ and $k_2(n)$ against n and are shown in Fig. 1, which also displays similar simulation results for a FP-based realization using the same wordlength for the exponent and the mantissa but a larger value of 2^{-3} for μ_p. Clearly, the degradation in the convergence speed under BFP due to the above restricted choice of μ_p, though noticable, is very much within acceptable limits. For the BFP-based simulation, also note that both $k_1(n)$ and $k_2(n)$ have magnitudes less than one right from $n = 0$ onward, and thus, $\phi_p(n)$, $p = 1, 2$ saturates at $\Phi_p = 1$ at $n = 0$ itself.

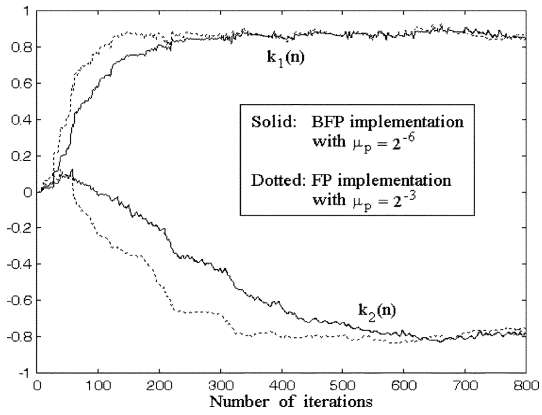

Fig. 1. Convergence results for $k_1(n)$ and $k_2(n)$ when implemented in BFP (with $\mu_p = 2^{-6}$), as shown by solid lines, and also in FP (with $\mu_p = 2^{-3}$), as shown by dotted lines, with the precision of 12 bits for mantissa and 4 bits for exponent for both the schemes.

REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] ——, "Asymptotic behavior of digital filters with block floating point arithmetic," in *Proc. ICASSP*, vol. III, Adelaide, Australia, 1994, pp. 609–612.

[5] 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.

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

[7] A. Mitra and M. Chakraborty, "The NLMS algorithm in block floating point format," *IEEE Signal Process. Lett.*, vol. 11, no. 3, pp. 301–304, Mar. 2004.

[8] S. Haykin, *Adaptive Filter Theory*, 4th ed. Englewood Cliffs, NJ: Prentice-Hall, 2001.

[9] Y. Iiguni, H. Sakai, and H. Tokumaru, "Convergence properties of simplified gradient adaptive lattice algorithms," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-33, no. 12, pp. 1427–1434, Dec. 1985.