Power-of-Two Adaptive Filters Using Tabu Search

Stefano Traferro and Aurelio Uncini

***Abstract*—Digital filters with power-of-two or a sum of power-of-two coefficients can be built using simple and fast shift registers instead of slower floating-point multipliers, such a strategy can reduce both the VLSI silicon area and the computational time. Due to the quantization and the nonuniform distribution of the coefficients through their domain, in the case of adaptive filters, classical steepest descent based approaches cannot be successfully applied. Methods for adaptation processes, as in the *least mean squares* (LMS) error and other related adaptation algorithms, can actually lose their convergence properties. In this brief, we present a customized Tabu Search (TS) adaptive algorithm that works directly on the power-of-two filter coefficients domain, avoiding any rounding process. In particular, we propose TS for a time varying environment, suitable for real time adaptive signal processing. Several experimental results demonstrate the effectiveness of the proposed method.**

***Index Terms*—Adaptive digital filter, finite precision design of adaptive digital filter, finite wordlength, global optimization, signed digit code, signed power-of-two, tabu search.**

I. Introduction

The use of adaptive systems, i.e., capable of altering their transfer functions as the environment changes, appears in many applications, such as the equalization of a communication channel or the automatic control of a system. Many adaptation procedures are based on the use of algorithms derived from the well-known *least mean squares* (LMS) [16].

The growing need for high-speed processing in communication hardware has attracted the attention of designers toward digital FIR filters with signed power-of-two (SPT), or combinations of SPT, coefficients. These architectures make it possible to compute multiplications quickly and efficiently, as well as reducing the area on a silicon chip, thus achieving a higher degree of integration and a high throughput signal rate.

In the case of adaptive-filter fast hardware implementation, the tuning optimization algorithms can require some modifications. The solutions yielded by the LMS have to be rounded in some way. In [17], the authors proposed a hardware solution to perform the stochastic gradient algorithm. A power-of-two quantizer is used for the equalizer output error, and a hard limiter for the receiving data to reduce the multiplication to a simple shift. Such an approach, here called the log-log LMS algorithm, is presented in [12], where the authors proposed only a fast adaptation algorithm, while the filtering section is unchanged and the filter coefficients are represented as floating-point. On the contrary, to assure a high filter-throughput rate, we also need a fast filtering section to avoid a cumbersome hardware multiplier.

In this work, we propose a new adaptive method based on the Tabu Search (TS) optimization algorithm [6]–[8], that works directly on the quantized filter coefficients domain made of a sum of SPT's. The proposed method avoids any coefficient rounding and is suitable for a simple hardware implementation [1], [2], and [18].

The TS algorithm has been widely employed in typical operational research problems and for static digital filter design [4], [5], [15]; here, we have exploited its capability in finding optimal points in a dynamical search space.

In the case of SPT static filter design, some researchers have tried to use nonconventional optimization procedures, like the Genetic Algorithm (GA) [9], [11] or Simulated Annealing (SA) [10]. However, due to their slow convergence properties, algorithms like GA or SA cannot be straightforwardly used for fast adaptive DSP applications.

In Section II, we simply introduce the TS algorithm, derive some functional conditions, and work out simple modifications so that the employed method is able to track the solution in a time-varying cost-function environment. In Section III, we indicate a brief formulation of the equalization problem and report some experimental results.

II. TS Optimization Algorithm

TS is a heuristic technique for solving combinatorial optimization problems on a wide range of applications. It was introduced by Glover [6]–[8] in the late 1970's and is being adapted to certain engineering problems.

Unlike other combinatorial optimization algorithms, as in GA or SA, where each step is performed independently from the previous moves, TS keeps trace of the visited region in the search space in order to avoid entrapment in local minima and to prevent cycling. The search mechanism can be viewed as a dynamic system which makes it possible to reduce the amount of inspected points and, consequently, the computational burden.

A. The Basic TS Algorithm

Let us define our notation to describe a basic version of the TS algorithm:

X	the discrete domain of the optimization problem; it is also named feasible region;
$\mathbf{x} \in X$	the variables array;
$f(\cdot)$	the objective function. Its value, evaluated on a point $\mathbf{x}$, is indicated as $z = f(\mathbf{x})$ throughout the text;
$\mathbf{x}^* \in X$	the point of the domain where $f(\cdot)$ reaches its minimum, $z^* = f(\mathbf{x}^*)$;
$\mu(\cdot)$	a function, usually called *move*, which generates a new point during the search phase. It is defined as $\mu: X \rightarrow X$, i.e., $\mu(\mathbf{x}) \in X$. Each move is associated with a value indicated as *move value*, $mv = f(\mu(\mathbf{x})) - f(\mathbf{x})$, and a *tabu status* which states whether it is forbidden;
$CL(\mathbf{x})$	the Candidate List representing a set of different moves applicable to the point $\mathbf{x}$, $CL(\mathbf{x}) = \{\mu \vert \mu(\mathbf{x}) \in X, \forall \mu\}$;
T	is the Tabu Set that collects the forbidden moves. A move is considered tabu, i.e., prohibited, if it has applied less than τ past iterations. The coefficients τ is defined as *tabu tenure*.

TS is a searching algorithm which visits the feasible region X to look for the points where the objective function reaches its minima. It exploits a prohibiting mechanism to avoid the most recent moves to be repeated in order to escape from local minima and prevent cycling. The

Manuscript received June 1999; revised February 2000. This work was supported in part by Ministero dell'Universitá e delle Ricerca Scientifica e Tecnologica (MURST) of Italy. This paper was recommended by Associate Editor J. Astola.

S. Traferro is with the Electronics and Automatics Department, University of Ancona, Italy via Brecce Bianche, I-60131, Ancona, Italy.

A. Uncini is with the INFOCOM Department, University of Rome "La Sapienza," I-00184, Rome, Italy.

Publisher Item Identifier S 1057-7130(00)04983-1.

following pseudo-code illustrates a basic version of the TS, where $\leftarrow$ denotes an assignment operation.

---

1) Initialization phase:
   $\mathbf{x} \leftarrow \mathbf{x}_0 \in X, \mathbf{x}^* \leftarrow \mathbf{x}, z^* \leftarrow f(\mathbf{x}^*), T \leftarrow \emptyset$ (empty set),
   $k \leftarrow 0$ (iteration counter)
2) Compute the Candidate List, $CL(\mathbf{x}) = \{\mu|\mu(\mathbf{x} \in X, \forall\mu\}$
3) If $CL(\mathbf{x}) - T = \emptyset$ then go to step 7)
4) Choose the best move in $CL(\mathbf{x}) - T, \mu_k$
5) Update the trial solution $\mathbf{x} \leftarrow \mu_k(\mathbf{x})$
6) If $f(\mathbf{x}) < z^*$ then $\mathbf{x}^* \leftarrow \mathbf{x}$, $z^* \leftarrow f(\mathbf{x}^*)$
7) Update T
8) Increment the iteration counter, k
9) If $k < k_{\max}$ then go to step 2)

End: the solution is $\mathbf{x}^*$ and $z^* = f(\mathbf{x}^*)$.

Each step through the domain is performed using the minimum-valued nontabu move chosen from the Candidate List, CL (steps 2–5). The solution $\mathbf{x}^*$ and the Tabu Set T are updated when every move has been executed (steps 6 and 7). In practice, T is implemented as τ length first-in first-out queue: when a move enters into T, during the kth iteration, the one executed at iteration $(k - \tau)$ escapes. In other words, T encapsulates the short term memory on which the algorithm is based, in order to carry out its search.

More sophisticated selection mechanisms can be introduced during step 4. They are called *aspiration*, *intensification*, *diversification*, and *strategic oscillation*, and are well described in [6]–[8].

The efficiency of the algorithm depends heavily on the type of encoding used for the domain variables and the constraints. In the basic implementations of TS, the only quantity to tune is the tabu tenure τ.

B. TS for Time-Varying Problems

TS has only been applied in static optimization problems. On the contrary, in adaptive applications, the functional characteristics change as time elapses. Referring to a fixed-coefficients domain, we have to solve the following formal problem:

$$\min f(\mathbf{x}, t), \quad \mathbf{x} \in X \subseteq \Re^N \tag{1}$$

where $f(\cdot)$ is the objective function, X denotes the feasible region for the coefficients and t explicitly indicates the temporal dependence. It is easy to perceive that the solution, $\mathbf{x}^*$ is, in general, connected to the temporal parameter t, $\mathbf{x}^*(t)$, so that the training heuristic must be able to track the variation of the functional as time elapses while remaining in the search space.

In order to tailor TS to a temporal problem, we define a property called *temporal coherence*. When we bring TS from a steady to a temporal varying environment, we want the actions it performs to maintain their meaning to produce the same effects. We call such behavior *temporal coherence*. It is not trivially guaranteed in every condition and sometimes requires high computational power.

We shall study this property by analyzing three aspects of the algorithm stated more extensively here:

2) the selection of the best move from $CL(\mathbf{x}) - T$ (step 4);
3) the admissibility mechanism;
4) the solution updating (step 6).

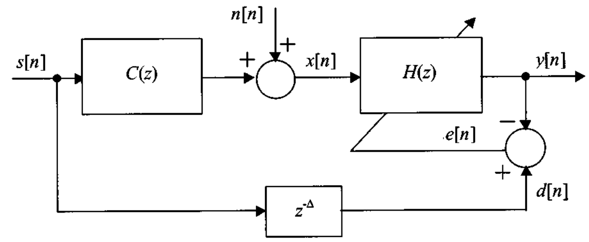

Fig. 1. Channel equalization scheme.

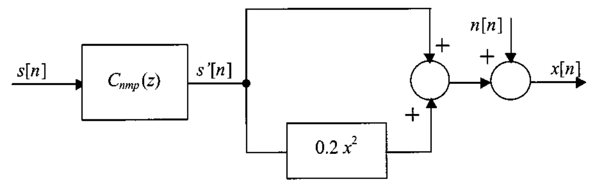

Fig. 2. Nonlinear channel model used in the experiment.

As specified in the previous section, every move $\mu(\mathbf{x})$ is associated to a value mv, defined as the variation of the objective function value due to the move operation. In a time varying environment, the parameter t induces a modification inside the $f(\cdot)$, thus for each move μ, we have

$$mv(\mu) = f(\mu(\mathbf{x}), t_1) - f(\mathbf{x}, t_0) \tag{2}$$

where the subscript 0 indicates the instant when the solution has been updated the last time. This quantity can be divided into two parts

$$mv(\mu) = \{f(\mu(\mathbf{x}), t_1) - f(\mathbf{x}, t_1)\} + \{f(\mathbf{x}, t_1) - f(\mathbf{x}, t_0)\} \tag{3}$$

where the former states the variation of the objective function value due to the translation μ, while the latter is due to the time elapsed. The change induced by a move can be kept under control by the heuristic, whereas the last term can be seen as an external agent. If we generate CL in a sequential manner, we will evaluate each move at different times. Consequently, the comparison process in step 4 becomes meaningless, as the selection is performed between different objective functions. This effect is more evident if the objective function temporal variations are greater than the descending steps.

In order to avoid temporal incoherence, the algorithm remains stationary during the execution of steps 2–4.

The second issue to analyze is the role of the short-term memory, accomplished through the Tabu Set T, in a nonstationary environment. In the static case, it is useful to avoid trapping in local minima, though when we deal with a dynamical objective function, this task can automatically be done because of the time variation of the cost function surface. The setting of the tabu tenure is less critical than in the steady-state case, even though we must always consider its constraining effect during the exploration of the search space.

The last aspect of the TS to study is the storage of the best objective function value ever found during the execution of the heuristic, $z^* = f(\mathbf{x}^*)$ (step 6). In the time-varying case, this reference totally loses its meaning. The cost function used to compute z* no longer exists (when time elapses, the temporal coefficients change their value, making the function different), thus the solution is not updated when TS finds a better minimum, but only if the objective function value is lower than that in the point $\mathbf{x}^*$. The use of the quantity z^* puts an uncontrollable element into the heuristic when the temporal variations are prominent. Therefore, the adoption of this quantity, as in the stationary case, makes the algorithm temporally incoherent.

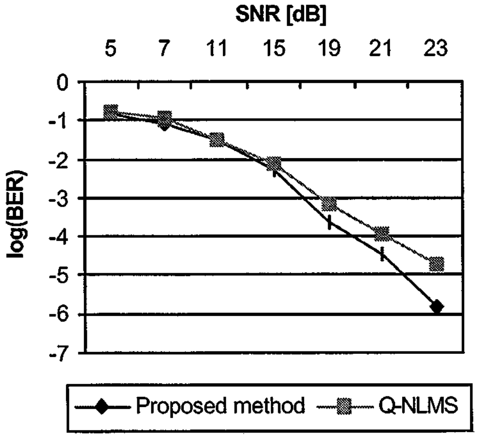

Fig. 3. Performance comparison. Channel $C_{mp}(z)$ and filter-order 10.

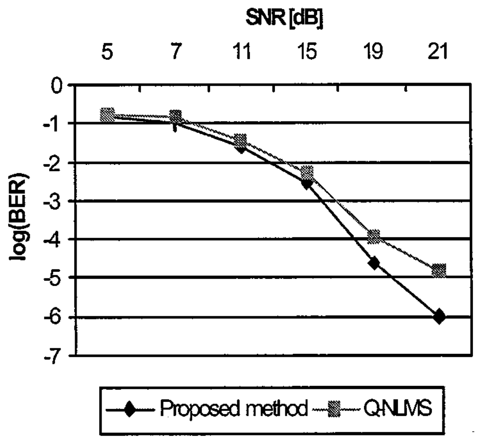

Fig. 4. Performance comparison. Channel $C_{mp}(z)$ and filter-order 7.

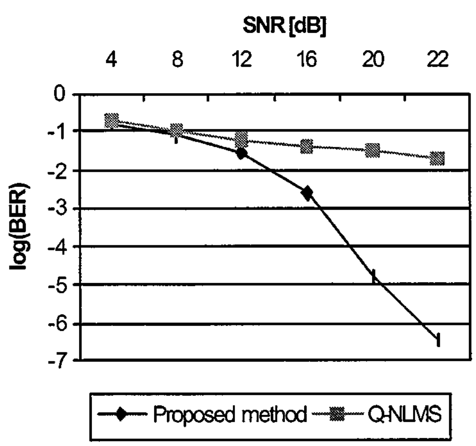

Fig. 5. Performance comparison. Channel $C_{nmp}(z)$ and filter-order 7.

A simple way to avoid the latter drawbacks consists in executing more than a move at each time step, i.e. the sequence of steps 2–7 is repeated more than once while the time is considered steady.

In order to render the adaptation procedure more flexible, the total amount of repetitions of the sequence 2–7, indicated as τ_{it}, is computed according to a measure of environmental variations; defining δ

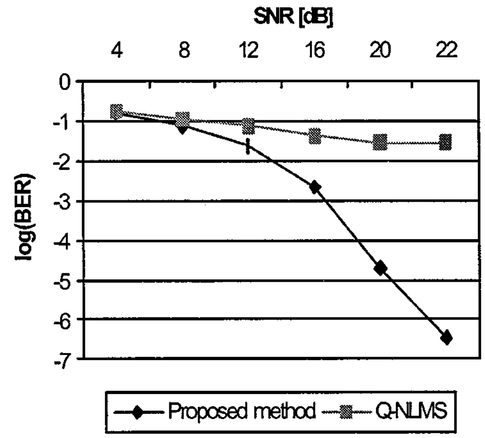

Fig. 6. Performance comparison. Channel $C_{nmp}(z)$ and filter-order 10.

as the relative difference between $f(\mathbf{x}, t)$ and the minimum of the cost function within a given temporal window, we adopt the following rule:

$$\tau_{it}(\delta) = \begin{cases} 1, & \delta \leq 0 \\ [(m_{it} - 1)\delta] + 1, & 0 < \delta < 1 \\ m_{it}, & \delta \geq 1 \end{cases} \quad (4)$$

where $[\cdot]$ denotes the integral part and m_{it} is set to ten. We also use a simple aspiration criterion within the TS mini-cycle, and the tabu tenure τ is set as half the total number of repetitions in order to reduce loop length in the average.

III. Experimental Results

In the case of adaptive SPT filters, classical algorithms, like LMS or its quantized version, denoted as Q-LMS, lose their convergence properties, as they deal with discrete domains [17]; thus, in order to verify the effectiveness of the proposed method, we decided to apply the TS heuristic to the problem of a linear equalization of a communication channel using the RLS criterion [13], [16].

We considered the channel equalization scheme described in Fig. 1, where $s[n]$ is the transmitting binary sequence, $C(z)$ states the equivalent transfer function of the communication channel [13], $n[n]$ denotes the noisy sequence, $y[n]$ indicates the output of the linear adaptive equalizer whose transfer function is $H(z)$, $d[n]$ is the target signal obtained delaying the sequence $s[n]$, and $e[n]$ states the error which drives the TS to tune the coefficients of the linear SPT FIR filter.

We chose the classical RLS objective function to minimize

$$\varepsilon[n] = \sum_{k=n-R+1}^{n} \gamma^{n-k} e[k]^2, \quad e[k] = d[k] - \mathbf{x}_k^T \cdot \mathbf{h} \quad (5)$$

where the inner product $\mathbf{x}_k^T \cdot \mathbf{h}$ represents the output of the equalizer with taps $\mathbf{h}$ at instant k; the vectors $\mathbf{x}_k$ and $\mathbf{h}$ are defined as $\mathbf{x}_k = (x[k], x[k-1], \cdots, x[k-N+1])^T$ and $\mathbf{h} = (h_1[n], h_2[n], \cdots, h_N[n])^T$, respectively, and N denotes the order of the filter. The weighting coefficient γ (called *forgetting factor*) has been set to 1, R indicates the temporal window length and has been set to 75 samples.

Simulations employed two linear channels are

$$C_{mp}(z) = 1 - 0.8z^{-1} + 0.5z^{-2} \quad (6a)$$

$$C_{nmp}(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}. \quad (6b)$$

$C_{mp}(z)$ and $C_{nmp}(z)$ indicate a *minimum phase* and *non minimum phase* linear channel, respectively. A nonlinear channel, reported in Fig. 2, C_{nl}, was built cascading the channel $C_{nmp}(z)$ and the nonlinear element $x + 0.2x^2$. White Gaussian noise $n[n]$ with zero mean is also added.

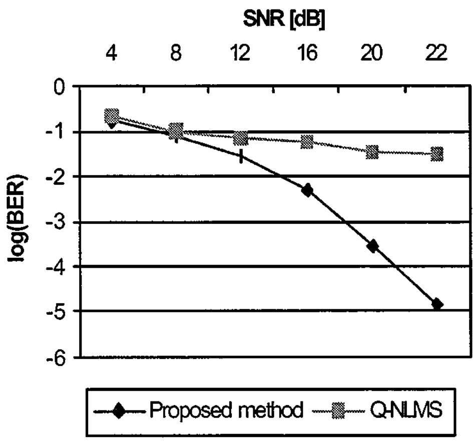

Fig. 7. Performance comparison. Nonlinear channel C_{nl} and filter-order 7.

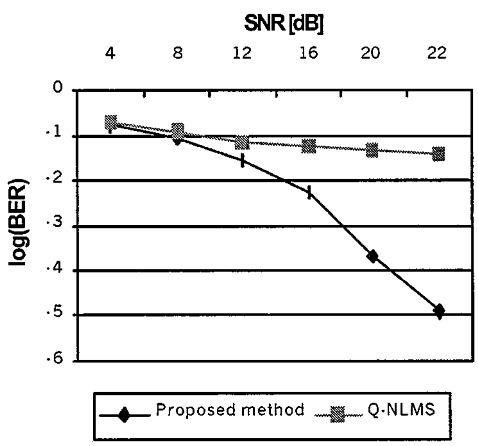

Fig. 8. performance comparison. Nonlinear channel C_{nl} and filter-order 10.

We employed equalizers of order $N = 7$ and $N = 10$ with coefficients made of a sum of two SPT's, thus belonging to the following domain:

$$D = \{\alpha \in \Re | \alpha = c_1 2^{-g_1} + c_2 2^{-g_2}\}, \quad c_{1,2} \in \{-1, 0, 1\}, g_{1,2} \in \{0, 1, \cdots, B\} \quad (7)$$

where B has been set to 12 bits.

Each test was composed of two phases: an adaptation phase, to tune the filter coefficients by TS, and a measurement step, to evaluate the performances using the BER index over a sequence of 5 million signal samples.

The results were compared with those obtained using a quantized version of the normalized LMS algorithm, Q-NLMS, where the filter taps are rounded to the nearest element of D after each iteration. They are plotted in the graphs of Figs. 3–8.

We obtain a better performance as the SNR increased. In fact, the Q-NLMS algorithm was not able to control the quantization error, on the contrary, TS could effectively contrast it by displaying only slightly different performances over the channel range.

A physical limit is also evident when we analyze the results as a function of the equalizer order [3]: it is impossible to obtain lower BER levels using linear systems, due to the inherent nonlinear nature of this kind of problem.

Concerning the computational burden, TS can suffer a drop in performance with respect to the standard LMS-based algorithms. However, in order to guarantee the needed speed, a hardware real-time TS implementation can be simply tailored to this problem [1], [18].

IV. Conclusion

In this brief, we presented some modifications to the TS optimization algorithm to solve time varying problems. We tested the performances of the proposed method through the adaptive equalization of a communication channel using an FIR digital filter with combination of SPT coefficients. Results show a better quality of the adaptation process driven by the TS than that achieved by the Q-NLMS. The better capability in irregular domains of the TS is paid off by a greater computational effort.

Our intention was to introduce nonconventional optimization techniques for adaptive filtering problems to provide for the lack of convergence of the LMS-based algorithms. Further developments can certainly be made in the modification of the computational procedure and in the design of suitable hardware architectures that could even equip embedded systems using a general optimization engine.

References

[1] G. Anzellotti, R. Battiti, I. Lazzizzera, G. Soncini, A. Zorat, A. Sartori, G. Tecchiolli, and P. Lee, "Totem: A highly parallel chip for triggering applications with inductive learning based on the reactive tabu search," *Int. J. Modern Phys. C*, vol. 6, no. 4, pp. 555–560, 1995.

[2] R. Battiti and G. Tecchioli, "Training neural nets with the reactive tabu search," *IEEE Trans. Neural Networks*, vol. 6, pp. 1185–1200, Sept. 1995.

[3] S. Chen, G. J. Gibson, C. F. N. Cowan, and P. M. Grant, "Reconstruction of binary signals using an adaptive radial-basis," *Signal Processing*, vol. 22, pp. 77–93, 1991.

[4] A. Fanni, G. Giacinto, M. Marchesi, and A. Serri, "Tabu search coupled with deterministic strategies for optimal design of MRI devices," *Int. J. Appl. Electromag. Mechan.*, vol. 10, pp. 21–31, 1999.

[5] A. Fanni, M. Marchesi, F. Pilo, and A. Serri, "Tabu search metaheuristic for designing digital filters," *COMPEL, Int. J. Comput. Math. in Elect. and Electron. Eng.*, vol. 17, no. 5/6, pp. 789–796, 1998.

[6] F. Glover, "Tabu search—Part I," *ORSA J. Comput.*, vol. 1, no. 3, pp. 190–206, Summer 1989.

[7] ——, "Tabu search—Part II," *ORSA J. Comput.*, vol. 2, no. 1, pp. 4–32, Winter 1990.

[8] F. Glover and M. Laguna, *Tabu Search—Modern Heuristic Techniques for Combinatorial Problems*, C. R. Reeves, Ed. Oxford, U.K.: Blackwell Scientific, 1993.

[9] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[10] S. Kirkpatrick, C. D. Gelatt, Jr., and P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, May 1983.

[11] Q. Ma and C. F. N. Cowan, "Genetic algorithms applied to the adaptation of IIR filters," *Signal Processing*, vol. 48, pp. 155–163, 1996.

[12] S. S. Mahant-Shetti, S. Hosur, and A. Gather, "The log-log LMS algorithm," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, Munich, Germany, Apr. 1997.

[13] J. G. Proakis, *Digital Communications*, 2nd ed. New York: McGraw-Hill, 1989.

[14] S. Traferro and A. Uncini, "Power-of-two adaptive filters using tabu search," presented at the IEEE ISCAS'99 Int. Symp. Circuits and Systems, Orlando, FL, May 30–June 2, 1999.

[15] S. Traferro, F. Capparelli, F. Piazza, and A. Uncini, "Efficient allocation of power-of-two terms in FIR digital filter design using tabu search," presented at the IEEE ISCAS'99 Int. Symp. Circuits and Systems, Orlando, FL, May 1999.

[16] B. Widrow and S. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1985.

[17] P. Xue and B. Liu, "Adaptive equalizer using finite-bit power-of-two," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 1603–1611, Dec. 1986.

[18] S. Traferro and A. Uncini, "A simple hardware implementation of the tabu search heuristic for DSP application," in *Proc. DSP for Multimedia Communication and Services (ECMCS'99)*, Krakow, Poland, June 1999.

[19] N. Benvenuto, M. Marchesi, and A. Uncini, "Application of simulated annealing for the design of special digital filters," *IEEE Trans. Signal Processing*, vol. 40, Feb. 1992.