Experiments With Repeating Weighted Boosting Search for Optimization in Signal Processing Applications

Sheng Chen, *Senior Member, IEEE*, Xunxian Wang, and Chris J. Harris

***Abstract*—Many signal processing applications pose optimization problems with multimodal and nonsmooth cost functions. Gradient methods are ineffective in these situations, and optimization methods that require no gradient and can achieve a global optimal solution are highly desired to tackle these difficult problems. The paper proposes a guided global search optimization technique, referred to as the repeated weighted boosting search. The proposed optimization algorithm is extremely simple and easy to implement, involving a minimum programming effort. Heuristic explanation is given for the global search capability of this technique. Comparison is made with the two better known and widely used guided global search techniques, known as the genetic algorithm and adaptive simulated annealing, in terms of the requirements for algorithmic parameter tuning. The effectiveness of the proposed algorithm as a global optimizer are investigated through several application examples.**

***Index Terms*—Adaptive simulated annealing, boosting, evolutionary computation, genetic algorithm, global search, multistart, optimization, stochastic algorithm.**

I. Introduction

OPTIMIZATION problems with multimodal and/or nonsmooth cost functions are commonly encountered in a variety of signal processing applications. Gradient-based algorithms become ineffective in these applications due to the problem of local minima or the difficulty in calculating gradients. Optimization methods that require no gradient and can arrive at a global optimal solution offer considerable advantages in solving these difficult problems. Various research communities have always been interested in the topic of global optimization, due to its importance, and a variety of global optimization techniques have been developed; see, for example, [1]–[14]. Within the wide field of engineering, two well-known classes of such global optimization methods are the genetic algorithm (GA) [7]–[10] and adaptive simulated annealing (ASA) [11]–[14]. Both the GA and ASA have attracted considerable attention in signal processing applications; see, for example, [15]–[23]. The GA and ASA belong to a class of so-called guided random search methods. The underlying mechanisms for guiding optimization search process are, however, very different for the two methods. The GA is population based and evolves a solution population according to the principles of the evolution of species in nature. It is by far the most widely applied global optimization scheme in machine learning and engineering applications. The ASA, by contrast, evolves a single solution in the parameter space with certain guiding principles that imitate the random behavior of molecules during the annealing process. Unlike the conventional simulated annealing [11], [24], the ASA adopts an important mechanism called the reannealing scheme, which not only speeds up the search process but also makes the optimization process robust to different problems.

The motivation of this work comes out of our experience with the GA and ASA. In line with many other researchers' experience, we have found that the two algorithms generally perform well in very different problems and have similarly good convergence speeds. The search mechanisms of the GA are complicated, as it is difficult to understand exactly how the search space is being explored. A carefully designed GA requires major programming efforts. This difficulty may be circumvented by simply using an existing GA software packet. Tuning a GA, however, is by no means an easy task and requires considerable experience, as there are a number of algorithmic parameters that need to be chosen carefully in order to achieve fast global convergence. The ASA, to some extent, is easier to implement and has less parameters that require tuning. Even so, it is always advisable to design the ASA algorithm with cares. In particular, the reannealing scheme and annealing schedule require carefully design and tuning. What motivates this research is the desire to have a general global optimization algorithm that is very simple to program and easy to tune, yet has convergence speed comparable to those of the GA and ASA. To this end, we propose a guided random search algorithm as a global optimization tool, which we refer to as the repeated weighted boosting search (RWBS).

The proposed algorithm is remarkably simple, requiring a minimum software programming effort and algorithmic tuning. The basic process evolves a population of initially randomly chosen solutions by performing a convex combination of the potential solutions and replacing the worst member of the population with it until the process converges. The weightings used in the convex combination are adapted to reflect the "goodness" of corresponding potential solutions using the idea from boosting [25]–[28]. The process is repeated a number of times or "generations" to improve the probability of finding a global optimal solution. An elitist strategy is adopted by retaining the

Manuscript received June 7, 2004; revised October 11, 2004. This paper was recommended by Associate Editor Diane J. Cook.

S. Chen and C. J. Harris are with School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K.

X. Wang is with Neural Computing Research Group, Aston University, Birmingham B4 7ET, U.K.

Digital Object Identifier 10.1109/TSMCB.2005.845398

best solution found in the current generation in the initial population of the next generation. The inner iteration loop, known as the weighted boosting search process, is designed to efficiently find a minimum point within the convex hull defined by the initial population members. This capability as a local optimizer can be explained by the theory of weak learnability associated with boosting [25], [26]. Keeping the best solution found in the previous generation as a member of the initial population ensures that the information obtained regarding the previous search region is not lost. By repeating the weighted boosting search a number of generations, the algorithm resembles a random search algorithm called the multistart [1]. However, there are important differences between the multistart and the proposed RWBS algorithm. In the multistart, a single point is randomly generated, and starting from this point, a local optimizer is used to find a minimum. The process is then repeated. Note that drawing randomly a number of points adopted by the proposed algorithm is also the sampling strategy used in a class of global optimization methods referring to as clustering methods [1]. A number of experiments are performed, involving three different signal processing applications, to demonstrate the effectiveness of this proposed RWBS algorithm as a global optimization tool. Comparisons with the GA and ASA, in terms of software programming effort, algorithmic tuning, and convergence speed, are given.

II. Proposed Guided Random Search Method

Many signal processing applications pose the following generic optimization problem:

$$\min_{\mathbf{u}\in\mathcal{U}} J(\mathbf{u}) \tag{1}$$

where $\mathbf{u} = [u_1 u_2 \cdots u_n]^T$ is the n-dimensional parameter vector to be optimized, and $\mathcal{U}$ defines the feasible set of $\mathbf{u}$. The cost function $J(\mathbf{u})$ can be multimodal and nonsmooth. We propose a guided global search method to find a global minimum solution of this optimization problem. The basic component of the proposed guided random search method is the following weighted boosting search algorithm.

A. Weighted Boosting Search as a Local Optimizer

Consider a population of P_S points: $\mathbf{u}_i \in \mathcal{U}$ for $1 \le i \le P_S$. These points can be randomly chosen. Let $\mathbf{u}_{\text{best}} = \arg\min J(\mathbf{u})$ and $\mathbf{u}_{\text{worst}} = \arg\max J(\mathbf{u})$, where $\mathbf{u} \in \{\mathbf{u}_i, 1 \le i \le P_S\}$. Now, a (P_S+1)th point is generated by performing a convex combination of $\mathbf{u}_i$, $1 \le i \le P_S$, as

$$\mathbf{u}_{P_S+1} = \sum_{i=1}^{P_S} \delta_i \mathbf{u}_i \tag{2}$$

where the weightings satisfy $\delta_i \ge 0$ and

$$\sum_{i=1}^{P_S} \delta_i = 1. \tag{3}$$

Obviously, the point $\mathbf{u}_{P_S+1}$ is always within the convex hull defined by the P_S values $\mathbf{u}_i$, $1 \le i \le P_S$. A mirror image of $\mathbf{u}_{P_S+1}$ is then generated with respect to $\mathbf{u}_{\text{best}}$ and along the direction defined by $\mathbf{u}_{\text{best}} - \mathbf{u}_{P_S+1}$ as

$$\mathbf{u}_{P_S+2} = \mathbf{u}_{\text{best}} + (\mathbf{u}_{\text{best}} - \mathbf{u}_{P_S+1}). \tag{4}$$

If $\mathbf{u}_{P_S+1}$ or $\mathbf{u}_{P_S+2}$ are outside $\mathcal{U}$, they can always be projected back to $\mathcal{U}$. According to their cost function values, the best of $\mathbf{u}_{P_S+1}$ and $\mathbf{u}_{P_S+2}$ then replaces $\mathbf{u}_{\text{worst}}$. The process is repeated until the population converges. The convergence can be assumed, for example, if

$$\|\mathbf{u}_{P_S+1} - \mathbf{u}_{P_S+2}\| < \xi_B \tag{5}$$

where the small positive scalar ξ_B is the chosen accuracy.

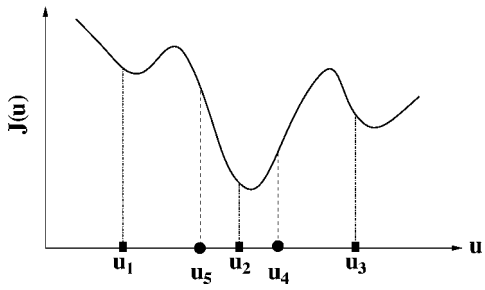


Fig. 1. Simple weighted search optimization process.

A simple illustration is depicted in Fig. 1 for a one-dimensional (1-D) case, where there are $P_S = 3$ points ($\mathbf{u}_1$, $\mathbf{u}_2$ and $\mathbf{u}_3$), $\mathbf{u}_{\text{best}} = \mathbf{u}_2$, and $\mathbf{u}_{\text{worst}} = \mathbf{u}_1$. The fourth point $\mathbf{u}_4$ is a weighted combination of $\mathbf{u}_1$, $\mathbf{u}_2$, and $\mathbf{u}_3$, and $\mathbf{u}_5$ is the mirror image of $\mathbf{u}_4$. As $\mathbf{u}_4$ is better than $\mathbf{u}_5$ (a smaller cost function value), it replaces $\mathbf{u}_1$ in the population. Clearly, how the convex weighted combination is performed is critical. The weightings δ_i for $\mathbf{u}_i$, $1 \le i \le P_S$, should reflect the "goodness" of $\mathbf{u}_i$, and the process should be capable of self-learning or adapting these weightings. This is exactly the basic idea of boosting [25]–[28]. Specifically, the AdaBoost algorithm of [26] is modified to adapt the weightings δ_i, $1 \le i \le P_S$ in this weighted boosting search process. The weighted boosting search can be seen as an optimizer that finds an optimal solution within the convex region defined by the initial population. Although a rigorous proof remains to be worked out, a heuristic explanation of this capability as a local optimizer can be given using the theory of weak learnability [25], [26]. The members of the population $\mathbf{u}_i$, $1 \le i \le P_S$ can be seen to be produced by a "weak learner," as they are generated "cheaply" and do not guarantee certain optimal property. Schapire [25] showed that any weak learning procedure can be efficiently transformed (boosted) into a strong learning procedure with certain optimal property. In our case, this optimal property is the ability of finding an optimal point within the defined search region. Boosting is a general method for improving the accuracy of any given learning algorithm, and the effectiveness of the boosting strategy for a wide rang of machine learning applications is well documented; see, for example, [25]–[28].

B. Repeated Weighted Boosting Search as a Global Optimizer

The aforementioned weighted boosting search is a local optimizer, as the solution obtained depends on the initial choice

of population. An effective strategy to "convert" a local optimizer to a global optimizer is to repeat it multiple times with some random sampling initialization. This, for example, is the strategy adopted in a stochastic algorithm for global optimization called the multistart [1]. We employ this proven strategy and repeat the weighted boosting search a number of times. Each run of the weighted boosting search process is referred to as a generation. An elitist initialization of the population for each generation is adopted, where each generation retains the solution found in the previous generation and fill the rest of the population randomly. This RWBS algorithm can now be summarized as follows.

Specify the following algorithmic parameters: P_S—population size; N_G—number of generations in the repeated search; N_B—number of iterations in the weighted boosting search; ξ_B—accuracy for terminating the weighted boosting search.

- •• **Outer loop: generations** For $g = 1 : N_G$
- *Generation initialization*: Initialize the population by setting $\mathbf{u}_1^{(g)} = \mathbf{u}_{\text{best}}^{(g-1)}$ and randomly generating rest of the population members $\mathbf{u}_i^{(g)}$, $2 \le i \le P_S$, where $\mathbf{u}_{\text{best}}^{(g-1)}$ denotes the solution found in the previous generation. If $g = 1$, $\mathbf{u}_1^{(g)}$ is also randomly chosen.
- *Weighted boosting search initialization*: Assign the initial distribution weightings $\delta_i(0) = (1/P_S)$, $1 \le i \le P_S$ for the population, and calculate the cost function value of each point

$$J_i = J\left(u_i^{(g)}\right), \quad 1 \le i \le P_S.$$

- **Inner loop: weighted boosting search** For $t = 1 : N_B$
- *Step 1: Boosting*

1) Find

$$i_{\text{best}} = \arg \min_{1 \le i \le P_S} J_i$$
$$i_{\text{worst}} = \arg \max_{1 \le i \le P_S} J_i.$$

Denote $\mathbf{u}_{\text{best}}^{(g)} = \mathbf{u}_{i_{\text{best}}}^{(g)}$ and $\mathbf{u}_{\text{worst}}^{(g)} = \mathbf{u}_{i_{\text{worst}}}^{(g)}$.

2) Normalize the cost function values

$$\bar{J}_i = \frac{J_i}{\sum_{j=1}^{P_S} J_j}, \quad 1 \le i \le P_S.$$

3) Compute a weighting factor β_t according to

$$\eta_t = \sum_{i=1}^{P_S} \delta_i(t-1)\bar{J}_i, \quad \beta_t = \frac{\eta_t}{1-\eta_t}.$$

4) Update the distribution weightings for $1 \le i \le P_S$

$$\delta_i(t) = \begin{cases} \delta_i(t-1)\beta_t^{\bar{J}_i}, & \text{for; } \beta_t \le 1 \\ \delta_i(t-1)\beta_t^{1-\bar{J}_i}, & \text{for; } \beta_t > 1 \end{cases}$$

and normalize them

$$\delta_i(t) = \frac{\delta_i(t)}{\sum_{j=1}^{P_S} \delta_j(t)}, \quad 1 \le i \le P_S.$$

- *Step 2: Parameter updating*

1) Construct the (P_S+1)th point using the formula

$$\mathbf{u}_{P_S+1} = \sum_{i=1}^{P_S} \delta_i(t)\mathbf{u}_i^{(g)}.$$

2) Construct the (P_S+2)th point using the formula

$$\mathbf{u}_{P_S+2} = \mathbf{u}_{\text{best}}^{(g)} + \left(\mathbf{u}_{\text{best}}^{(g)} - \mathbf{u}_{P_S+1}\right).$$

3) Compute the cost function values $J(\mathbf{u}_{P_S+1})$ and $J(\mathbf{u}_{P_S+2})$ for these two points, and find

$$i_* = \arg \min_{i=P_S+1, P_S+2} J(\mathbf{u}_i).$$

4) The pair $(\mathbf{u}_{i_*}, J(\mathbf{u}_{i_*}))$ then replaces $(\mathbf{u}_{\text{worst}}^{(g)}, J_{i_{\text{worst}}})$ in the population.[1]

- If $\|\mathbf{u}_{P_S+1} - \mathbf{u}_{P_S+2}\| < \xi_B$, exit **inner loop**
- **End of inner loop** The solution found in the gth generation is $\mathbf{u} = \mathbf{u}_{\text{best}}^{(g)}$.
- •• **End of outer loop** This yields the solution $\mathbf{u} = \mathbf{u}_{\text{best}}^{(N_G)}$.

To guarantee a global optimal solution as well as to achieve a fast convergence, the algorithmic parameters P_S, N_G, N_B, and ξ_B need to be set carefully. The appropriate values for these algorithmic parameters depends on the dimension of $\mathbf{u}$ and how hard the objective function to be optimized is. Generally, these algorithmic parameters have to be found empirically, just as in any global optimization algorithm. The elitist initialization is very useful, as it keeps the information obtained by the previous search generation, which otherwise would be lost due to the randomly sampling initialization. In the inner loop optimization, there is no need for every members of the population to converge to a (local) minimum, and it is sufficient to locate where the minimum lies. Thus, the number of weighted boosting iterations N_B can be set to a relatively small integer, and the accuracy for stopping the weighted boosting search ξ_B can be set to a relatively large value. This makes the search efficient, achieving convergence with a small number of the cost function evaluations. It should be obvious, although the formal proof is still required, that with sufficient number of repeats or generations, the algorithm will guarantee to find a global optimal solution. The question is as follows: Will this strategy of repeating weighted boosting search be efficient in terms of the required total number of the cost function evaluations? This will be investigated in the experiments of Section III. Here, we use a simple 1-D optimization problem to gain some experience with this RWBS algorithm.

The cost function to be optimized, which is depicted in Fig. 2(a), is given by

$$J(u) = \begin{cases} -\frac{\sin(6(u+5))}{2(u+5)} + 4, & -8 \le u \le -2 \\ -4\exp\left(-5(u+1)^2\right) + 4.16, & -2 < u \le 0 \\ -\frac{\sin(6(u-4))}{2(u-4)} + 4, & 0 < u \le 8. \end{cases}$$

Uniformly random sampling in $[-8, 8]$ was adopted for population initialization. With a population size $P_S = 4$, number of weighted boosting iterations $N_B = 6$, and stopping accuracy

[1]It will keep the weighting $\delta_{i_{\text{worst}}}(t)$. This weighting value will be updated anyway in the next round according to the new cost function value.

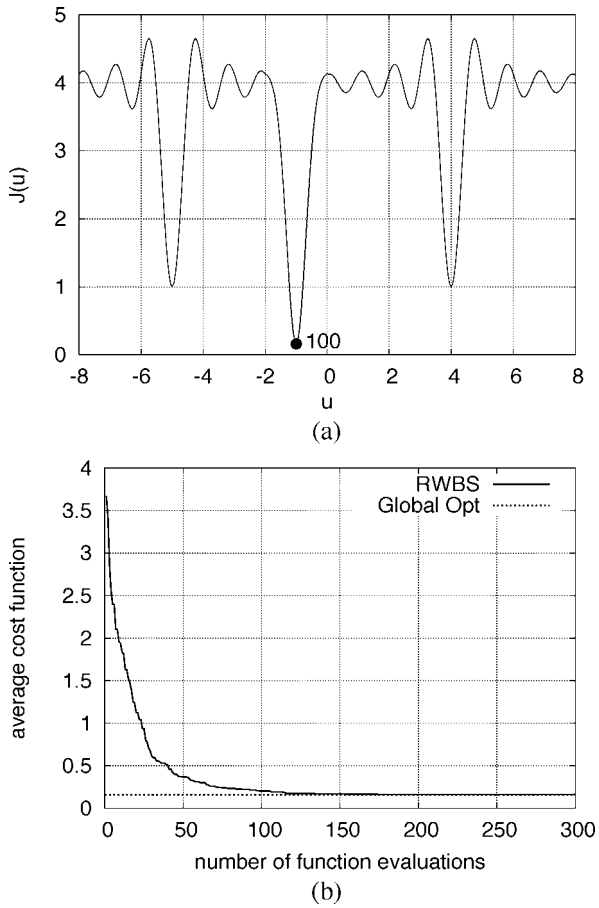


Fig. 2. Experimenting with a 1-D multimodal function minimization using the repeated weighted boosting search. (a) Cost function, where the number 100 beside the solution (point in the graph) indicates the convergence to the global minimum in all the 100 experiments. (b) Convergence performance averaged over 100 experiments.

for weighted boosting search $\xi_B = 0.02$ as well as by setting number of generations to $N_G > 6$, the RWBS algorithm consistently converged to the global minimum point at $u = -1$ in all the 100 experiments conducted, as can be seen from the convergence performance shown in Fig. 2(b). The averaged number of cost function evaluations required for the algorithm to converge to the global optimal solution is around 110.

C. Implementation Comparison With the GA and ASA

Because the GA and ASA are popular choices for global optimization in machine learning and engineering applications, we compare the implementation considerations of the proposed RWBS algorithm with the GA and ASA. The first implementation issue is the software programming effort required to code an algorithm. It is self-evident that the RWBS is extremely simple, requiring a minimum programming effort. The GA is anything but simple, in terms of programming efforts. The ASA, in the form presented in [21], is much easier to programme than the GA but still cannot compete with the simplicity of the RWBS. The difficulty with programming a GA can be circumvented by simply using some existing GA software packets written by experts, but the same cannot be said with tuning a GA. To tune a GA for a successful application requires considerable expertise, as there are a large number of the algorithmic parameters that need to be set/chosen carefully. The ASA has very few algorithmic parameters to tune, compared with the GA, but tuning a successful ASA is still much harder than the RWBS. This is because the choices of these algorithmic parameters have critical influence on the reannealing scheme and annealing schedule that ensure fast global convergence. The RWBS also has very few algorithmic parameters, and moreover, the choices of these parameters are relatively straightforward, in comparison with the GA and ASA. In Table I, we compare the tuning issue of these three global optimization algorithms by listing their algorithmic parameters. Since operations involved in the RWBS are straightforward and much simpler than those for the GA or ASA, the computational complexity of this algorithm will be much simpler than those for the GA and ASA, provided that the convergence speed of the algorithm, in terms of the number of total cost function calls, is comparable to those of the GA and ASA. Theoretical analysis of convergence speed of a generic global optimizer is very difficult if not impossible. We therefore turn to experiments for investigating this critical issue.

TABLE I
COMPARISON OF ALGORITHMIC TUNING REQUIREMENTS FOR THE GA, ASA, AND RWBS

GA	GA type
	Population size
	Number of generations
	Mutation type
	Probability of mutation
	Crossover type
	Probability of crossover
	Scaling scheme
	Genome type
	Initialization scheme
	Comparison scheme
	Encoding/decoding scheme
	Selection scheme
	Elitism
ASA	Number of acceptance points for reannealing
	Number of generated points for annealing
	Annealing rate control parameter
	Step size in calculating sensitives
RWBS	Population size
	Number of weighted boosting iterations
	Accuracy for stopping weighted boosting search
	Number of generations

III. OPTIMIZATION APPLICATIONS

The versatility of the proposed guided random search algorithm as potentially a global optimization tool is investigated using three very different signal processing application problems. The first two applications are typically global optimization problems, and the previous results involving the GA and ASA are available. These results are used as benchmarkers to compare with the convergence performance of the proposed RWBS algorithm.

A. Infinite-duration Impulse Response (IIR) Filter Design

The adaptive IIR filter is a classical research area, and many properties of IIR filters are well known [29], [30]. Because the cost function of IIR filters is generally multimodal with respect to the filter coefficients and the usual gradient-based algorithm can easily be stuck at local minima, global optimization methods

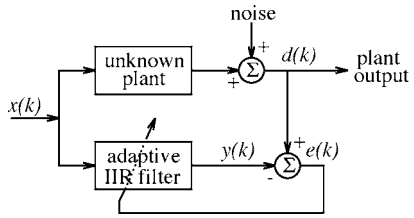


Fig. 3. Schematic of adaptive IIR filter for system identification, where $x(k)$ is the system input, $y(k)$ the filter output, and $d(k)$ the noisy plant observation.

have been applied to IIR filter design; see, for example, [15], [16], and [31]–[34]. Consider the use of IIR filter in system identification application, as depicted in Fig. 3, where the IIR filter with the model transfer function

$$H_M(z) = \frac{\sum_{i=0}^{L} a_i z^{-i}}{1 + \sum_{i=1}^{M} b_i z^{-i}} \tag{6}$$

is used to model the unknown plant with the system transfer function $H_S(z)$. The IIR filter design can be formulated as an optimization problem with the mean square error (MSE) as the cost function:

$$J(\mathbf{u}) = \mathrm{E}\left[e^2(k)\right] = \mathrm{E}\left[(d(k) - y(k))^2\right] \tag{7}$$

where $d(k)$ is the filter's desired response, $y(k)$ the filter's output, $e(k) = d(k) - y(k)$ is the filter's error signal, and $\mathbf{u} = [a_0\ a_1\ \cdots\ a_L\ b_1\ \cdots\ b_M]^T$ denotes the filter coefficient vector. The goal is to minimize the MSE (7) by adjusting $\mathbf{u}$. In practice, ensemble operation is difficult to realize, and the cost function (7) is usually substituted by the time-averaged cost function:

$$J_N(\mathbf{u}) = \frac{1}{N}\sum_{k=1}^{N} e^2(k). \tag{8}$$

When the filter order $M(\geq L)$ is smaller than the system order, local minima problems can be encountered [30]. To maintain the stability during optimization, we convert the direct-form IIR filter coefficients b_i, $1 \leq i \leq M$ to the lattice-form reflection coefficients κ_i, $0 \leq i \leq M-1$ and make sure that all the κ_i have magnitudes less than 1. Thus, the filter coefficient vector used in optimization is

$$\mathbf{u} = [a_0\ a_1 \cdots a_L\ \kappa_0 \cdots \kappa_{M-1}]^T. \tag{9}$$

Converting the reflection coefficients back to the direct-form coefficients is straightforward [35].

Example 1: This example is taken from [30]. The system and filter transfer functions are, respectively

$$H_S(z) = \frac{0.05 - 0.4z^{-1}}{1 - 1.1314z^{-1} + 0.25z^{-2}},\quad H_M(z) = \frac{a_0}{1 + b_1 z^{-1}}. \tag{10}$$

The analytical MSE (7) in this case is known when the input is a white sequence and the noise is absent. The cost function has a global minimum at $\mathbf{u}_{\mathrm{global}} = [-0.311\ -0.906]^T$ with the value of the normalized MSE 0.2772 and a local minimum at $\mathbf{u}_{\mathrm{local}} = [0.114\ 0.519]^T$ with the normalized MSE value 0.9762. In the population initialization, the parameters were uniformly randomly chosen as $(a_0, b_1) \in (-1.0, 1.0) \times (-0.999, 0.999)$ ($-1.0 < b_1 < 1.0$ for stability consideration). It was found empirically that the population size $P_S = 4$, the number of weighted boosting iterations $N_B = 5$, the stopping accuracy for weighted boosting search $\xi_B = 0.05$, and the number of generations $N_G > 15$ were appropriate for this example, and Fig. 4(a) depicts convergence performance of the RWBS algorithm averaged over 100 experiments. The previous study [21], [34] has applied the ASA to this example. The result of using the ASA is reproduced in Fig. 4(b) for comparison. It can be seen from Fig. 4 that both the RWBS and ASA have the same fast convergence speed, requiring an average of 200 cost function calls to reach the global minimum. The work [15] has applied a GA to the same example. The result given in [15] shows that the GA is slower to converge to the global minimum, requiring an average of 600 cost function evaluations to do so. The distribution of the solutions obtained in 100 experiments by the RWBS algorithm is shown in Fig. 5.

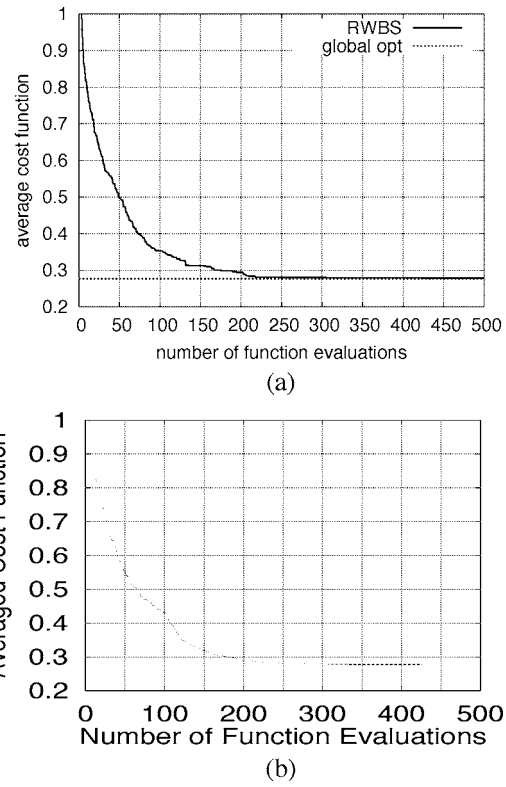


Fig. 4. Convergence performance averaged over 100 experiments for IIR filter design Example 1. (a) Using the repeated weighted boosting search. (b) Using the adaptive simulated annealing.

Example 2: This is a third-order system with the system transfer function given by

$$H_S(z) = \frac{-0.3 + 0.4z^{-1} - 0.5z^{-2}}{1 - 1.2z^{-1} + 0.5z^{-2} - 0.1z^{-3}}. \tag{11}$$

In the simulation, the system input $x(k)$ was a uniformly distributed white sequence, taking values from $(-1, 1)$, and the signal-to-noise ratio was SNR $= 30$ dB. The data length used in calculating the MSE (8) was $N = 2000$. When a reduced-order

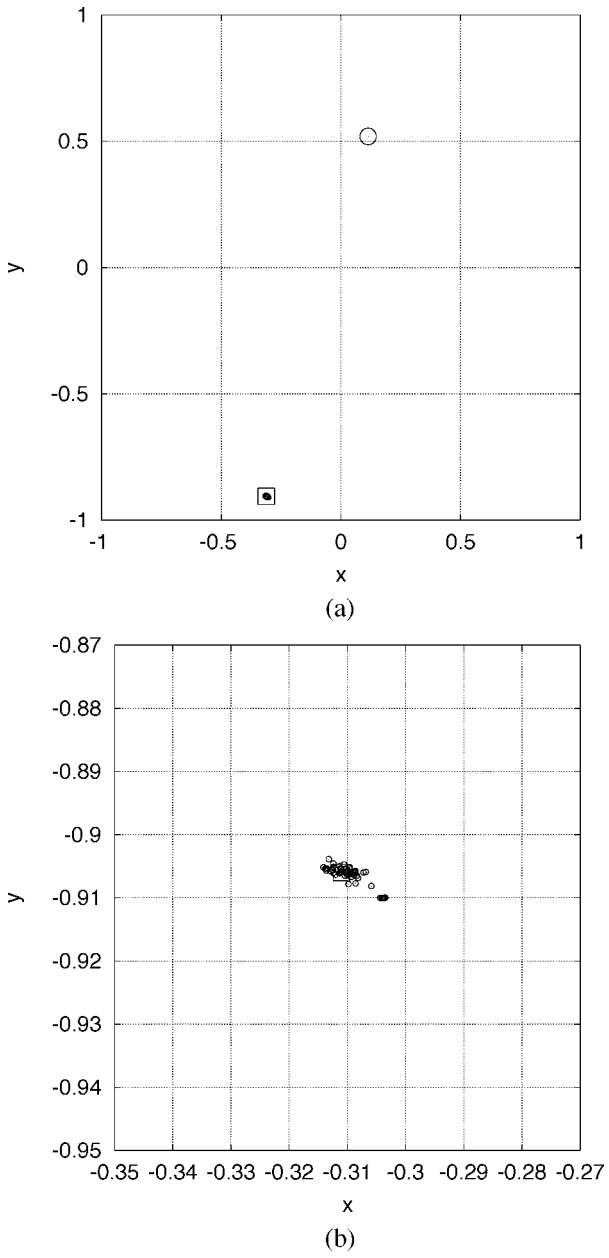


Fig. 5. Distribution of the solutions (a_0, b_1) (small circles) obtained in 100 experiments for IIR filter design Example 1 by the repeated weighted boosting search. (a) Entire search space, where the large square indicates the global minimum and the large circle the local minimum. (b) Zooming in the global minimum.

filter with $M = 2$ and $L = 1$ was used, the MSE was multimodal, and the gradient-based algorithm performed poorly as was demonstrated clearly in [34]. It was found that for the proposed RWBS algorithm, $P_S = 5$, $N_B = 10$, $\xi_B = 0.05$, and $N_G > 20$ were appropriate. Fig. 6(a) depicts convergence performance of the RWBS algorithm averaged over 500 experiments. In [21] and [34], convergence performance using the ASA was obtained by averaging 100 experiments,[2] and this result is also replotted in Fig. 6(b).Again, it is seen from Fig. 6 that both the RWBS and ASA have the same fast global convergence speed. The distribution of the solutions obtained in 500

[2]There were typing errors in [21] and [34]: The input sequence $x(k)$ was uniformly distributed in $(-1, 1)$ and not in $(-0.5, 0.5)$.

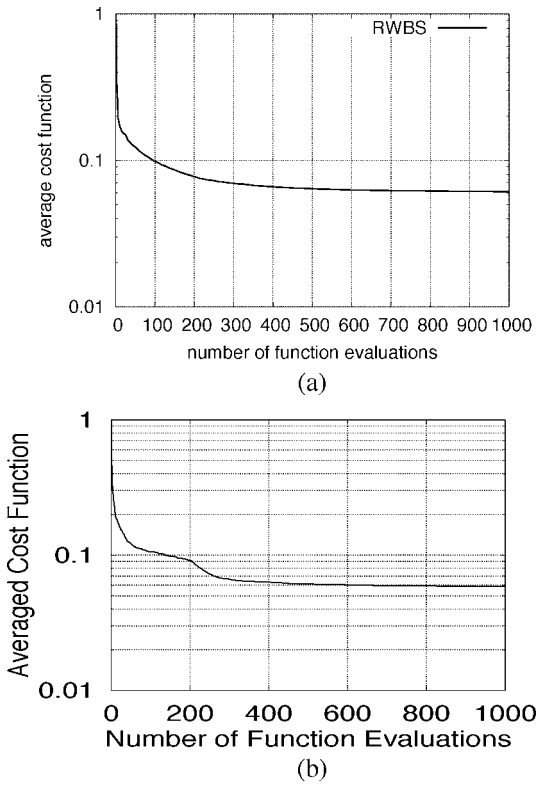


Fig. 6. Convergence performance for IIR filter design Example 2. (a) Averaged over 500 experiments using the repeated weighted boosting search. (b) Averaged over 100 experiments using the adaptive simulated annealing.

experiments by the RWBS is illustrated in Fig. 7. It is clear that for this example, there are infinitely many global minima, and the global minimum solutions for (b_1, b_2) form a 1-D space.

B. ML Joint Channel and Data Estimation

Consider the digital communication channel, whose received signal at sample k is modeled by

$$r(k) = \sum_{i=0}^{n_a-1} a_i s(k-i) + e(k) \tag{12}$$

where n_a is the channel length, a_i are the channel impulse response taps, the symbol sequence $\{s(k)\}$ is independently identically distributed with an M-pulse amplitude modulation (PAM) symbol constellation, and $e(k)$ is a channel Gaussian white noise. Let

$$\left.\begin{aligned} \mathbf{r} &= [r(1)r(2)\cdots r(N)]^T \\ \mathbf{s} &= [s(-n_a+2)\cdots s(0)s(1)\cdots s(N)]^T \\ \mathbf{a} &= [a_0\, a_1 \cdots a_{n_a-1}]^T \end{aligned}\right\} \tag{13}$$

be the vector of N received data samples, the corresponding transmitted data sequence, and the channel tap vector, respectively. The joint maximum likelihood (ML) estimate of $\mathbf{a}$ and $\mathbf{s}$ is obtained by maximizing the conditional probability density function of $\mathbf{r}$, given $\mathbf{a}$ and $\mathbf{s}$. Equivalently, the ML solution is the minimum of the cost function:

$$J_{\mathrm{ML}}(\hat{\mathbf{a}}, \hat{\mathbf{s}}) = \sum_{k=1}^{N} \left(r(k) - \sum_{i=0}^{n_a-1} \hat{a}_i \hat{s}(k-i) \right)^2 \tag{14}$$

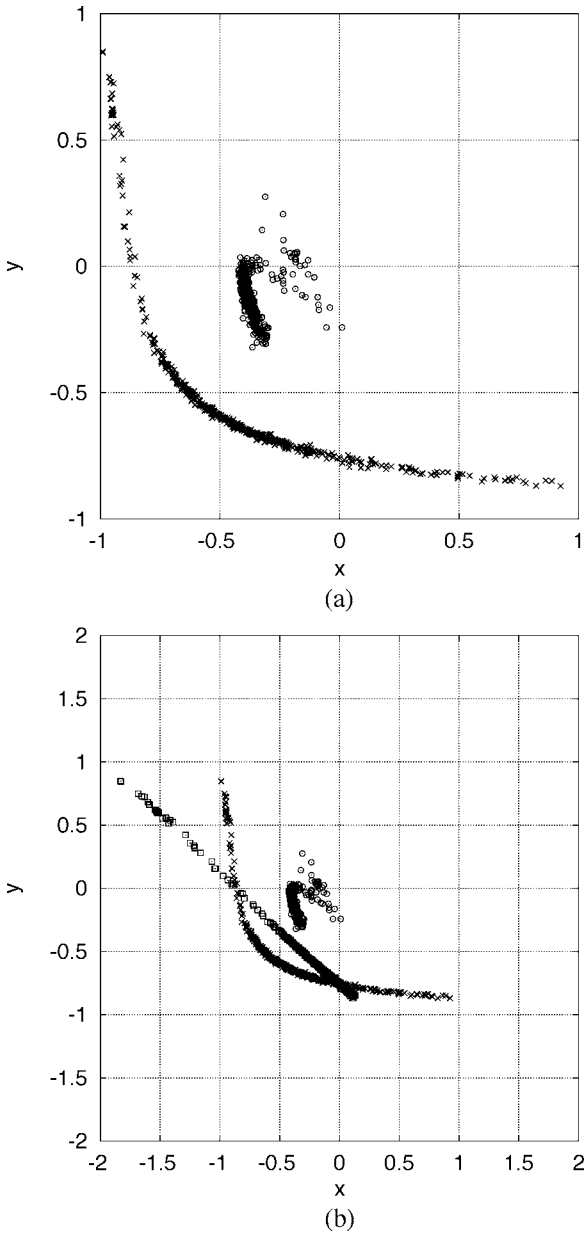


Fig. 7. Distribution of the solutions obtained with the repeated weighted boosting search algorithm in 500 experiments for IIR filter design Example 2. (a) (a_0, a_1) as circles and (κ_0, κ_1) as crosses. (b) (a_0, a_1) as circles, (b_1, b_2) as squares, and (κ_0, κ_1) as crosses.

that is

$$(\hat{\mathbf{a}}^*, \hat{\mathbf{s}}^*) = \arg\left[\min_{\hat{\mathbf{a}},\hat{\mathbf{s}}} J_{\mathrm{ML}}(\hat{\mathbf{a}}, \hat{\mathbf{s}})\right]. \tag{15}$$

This joint ML estimate, however, is too expensive to compute except for the simplest case. In practice, suboptimal solutions are sought for computational purposes. The algorithm based on a blind trellis search technique [36] is such an example.

The joint minimization process (15) can also be performed using an iterative loop first over the data sequences $\hat{\mathbf{s}}$ and then over all the possible channels $\hat{\mathbf{a}}$

$$(\hat{\mathbf{a}}^*, \hat{\mathbf{s}}^*) = \arg\left[\min_{\hat{\mathbf{a}}}\left(\min_{\hat{\mathbf{s}}} J_{\mathrm{ML}}(\hat{\mathbf{a}}, \hat{\mathbf{s}})\right)\right]. \tag{16}$$

The inner optimization can be carried out using the standard Viterbi algorithm (VA). The previous research has used the quantized channel algorithm [37], the GA [18], and the ASA [21] to perform the outer optimization. In this study, we apply the RWBS algorithm to perform the outer optimization. Specifically, given the channel estimate $\hat{\mathbf{a}}$, let the data sequence decoded by the VA be $\hat{\mathbf{s}}^*$. Then, the cost function used by the search algorithm is the MSE

$$J(\hat{\mathbf{a}}) = \frac{1}{N} J_{\mathrm{ML}}(\hat{\mathbf{a}}, \hat{\mathbf{s}}^*). \tag{17}$$

The search range for each channel tap is $-1 \leq a_i \leq 1$ since the channel can always be normalized. In practice, convergence of the algorithm is observed through the MSE (17). In simulation, the performance of the algorithm can also be assessed by the mean tap error (MTE), which is defined as

$$\mathrm{MTE} = \|\mathbf{a} - \mathrm{sgn}(a_0 \hat{a}_0) \cdot \hat{\mathbf{a}}\|^2 \tag{18}$$

Note that since both $(\hat{\mathbf{a}}^*, \hat{\mathbf{s}}^*)$ and $(-\hat{\mathbf{a}}^*, -\hat{\mathbf{s}}^*)$ are the solutions of the blind joint ML estimation problem, the channel estimate $\hat{\mathbf{a}}$ can converge to either the true channel $\mathbf{a}$ or $-\mathbf{a}$.

In the simulation study, the channel was given by

$$\mathbf{a} = [0.407\ 0.815\ 0.407]^T. \tag{19}$$

The algorithmic parameters for the RWBS were set to $P_S = 5$, $N_B = 6$, $\xi_B = 0.01$, and $N_G > 20$. Figs. 8 and 9 show the evolutions of the MSE and MTE with 2-PAM and 4-PAM symbols and different values of SNR, respectively. All the results were averaged over 100 experiments. The previous research [21] has

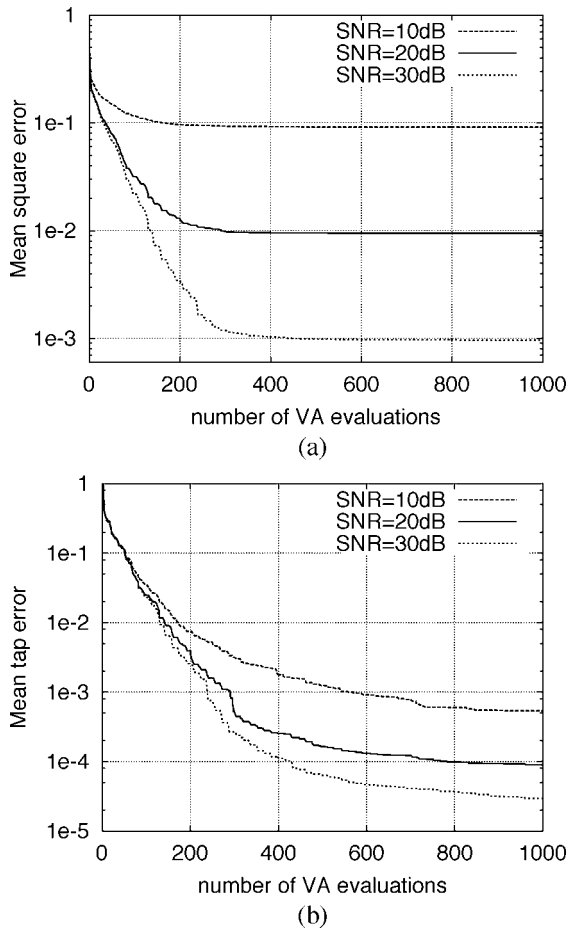


Fig. 8. (a) Mean square error and (b) mean tap error against number of VA evaluations averaged over 100 experiments, with 2-PAM symbols and data samples $N = 50$, using the repeated weighted boosting search algorithm.

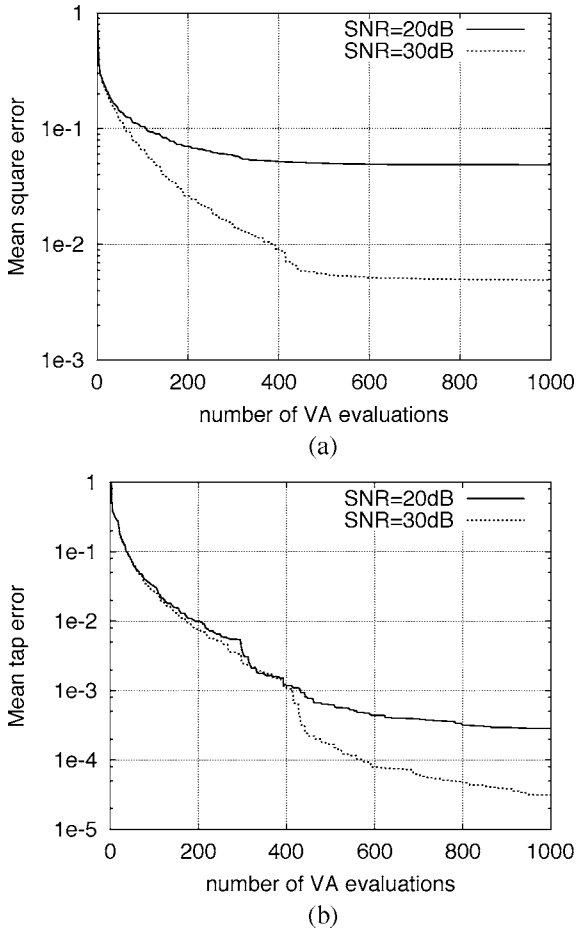


Fig. 9. (a) Mean square error and (b) mean tap error against number of VA evaluations averaged over 100 experiments, with 4-PAM symbols and data samples $N = 100$, using the repeated weighted boosting search algorithm.

applied the ASA to the same problem, and for a comparison, the results given in [21] are replotted in Figs. 10 and 11 for the 2-PAM and 4-PAM cases, respectively. Compared the results of using the RWBS and ASA in these four figures, it can be seen that the two algorithms have the same convergence speed in terms of the estimated MSE for this blind ML joint channel and data estimation. The results also shows that the RWBS is more accuracy than the ASA in terms of the MTE measure. The study in [18] applied a well-tuned micro GA to the same problem, and the results obtained in [18] showed that the micro GA has slightly faster convergence speed than the ASA in terms of the MSE but has poorer accuracy in terms of the MTE. The accuracy of the proposed RWBS algorithm is demonstrated by the distribution of solutions obtained in 100 experiments for the case of 4-PAM with SNR $= 30$ dB, which is depicted in Fig. 12.

C. Novel Kernel Classifier Design Approach

The state-of-the-art kernel modeling techniques, such as the support vector machine (SVM) and relevant vector machine (RVM) [38]–[41], have widely been adopted in classification applications. Typically, a kernel classification technique considers every training input pattern as a candidate kernel center and uses a single fixed kernel width for every kernel function.

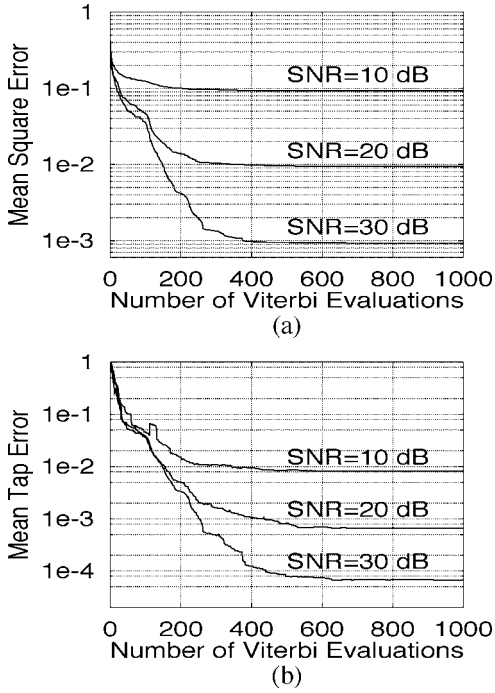


Fig. 10. (a) Mean square error and (b) mean tap error against number of VA evaluations averaged over 100 experiments, with 2-PAM symbols and data samples $N = 50$, using the adaptive simulated annealing algorithm.

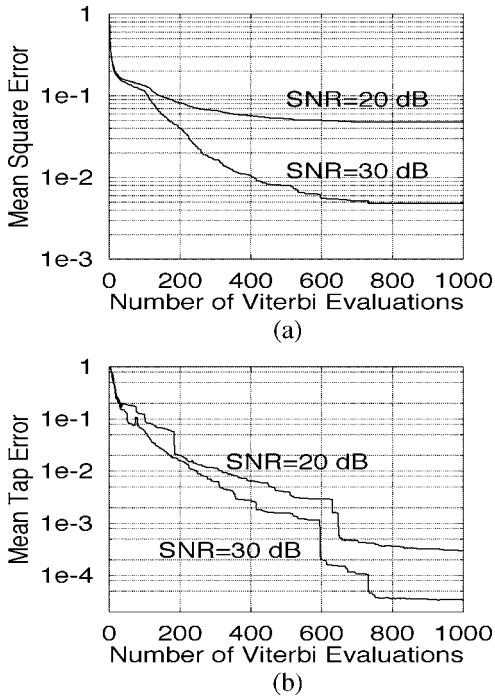


Fig. 11. (a) Mean square error and (b) mean tap error against number of VA evaluations averaged over 100 experiments, with 4-PAM symbols and data samples $N = 100$, using the adaptive simulated annealing algorithm.

A parsimonious or sparse representation is then sought. The value of the common kernel width has critical influence on the performance of the classifier and has to be learned via cross validation. This subsection reports an alternative kernel classifier design approach that incrementally constructs a sparse kernel classifier using the RWBS algorithm [42]. Unlike most

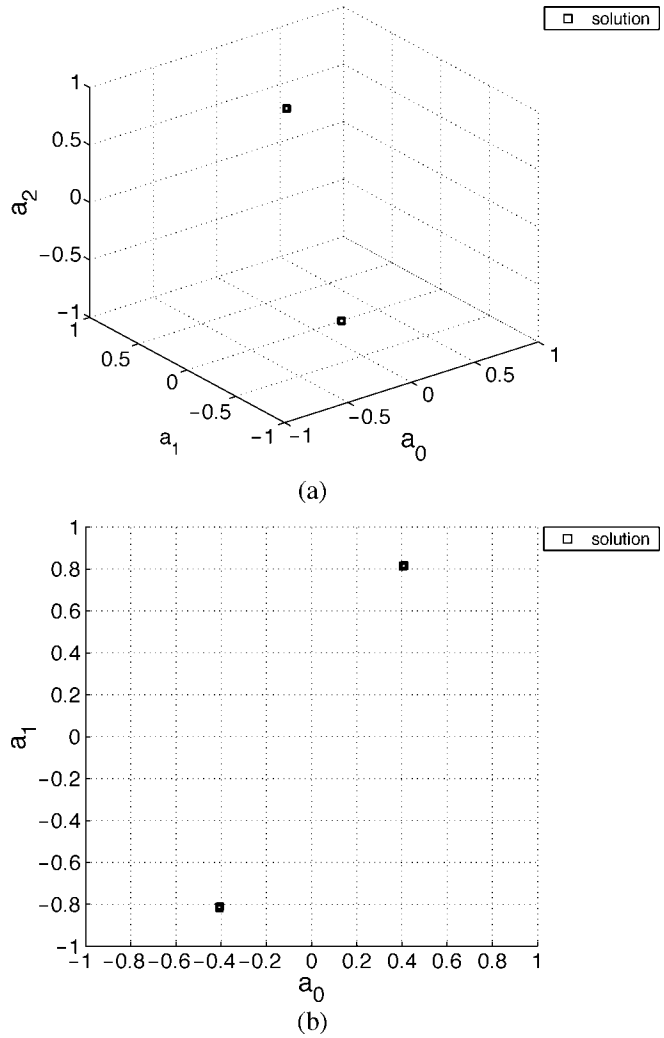


Fig. 12. Distribution of the solutions obtained in 100 experiments for the 4-PAM case with SNR = 30 dB and data samples $N = 100$. (a) ($\hat{a}_0$, $\hat{a}_1$, $\hat{a}_2$), and (b) ($\hat{a}_0$, $\hat{a}_1$), using the repeated weighted boosting search algorithm.

kernel classification methods, which restrict kernel means to the training input data and use a fixed common variance for all the kernel terms, the proposed technique can tune both the mean vector and diagonal covariance matrix of individual kernel by incrementally maximizing the Fisher ratio for class separability measure. The RWBS algorithm described in Section II is used to append kernels one by one in an orthogonal forward selection (OFS) procedure.

Consider the two-class kernel classifier of the form

$$\hat{c}(k) = \text{sgn}\left(y(k)\right) \quad \text{with} \quad y(k) = \sum_{i=1}^{M} w_i g_i\left(\mathbf{x}(k)\right) \tag{20}$$

where $\mathbf{x}(k)$ is an m-dimensional pattern vector with its associated class label $c(k) \in \{\pm 1\}$, $y(k)$ is the classifier output for input $\mathbf{x}(k)$, and $\hat{c}(k)$ is the estimated class label for $\mathbf{x}(k)$; w_i, $1 \le i \le M$ denote the classifier weights, M is the number of kernels, and $g_i(\bullet)$, $1 \le i \le M$ denote the classifier kernels. We allow the kernel function to be chosen as the general Gaussian function $g_i(\mathbf{x}) = G(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, with

$$G(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i)\right) \tag{21}$$

where the diagonal covariance matrix has the form of $\boldsymbol{\Sigma}_i = \text{diag}\{\sigma_{i,1}^2, \cdots, \sigma_{i,m}^2\}$. Given the N pairs of training data $\{\mathbf{x}(k), c(k)\}_{k=1}^N$, let us define the modeling residual as $e(k) = c(k) - y(k)$. Then, the classifier model (20) over the training data set can be expressed in matrix form as

$$\mathbf{c} = \mathbf{G}\mathbf{w} + \mathbf{e} \tag{22}$$

where $\mathbf{c} = [c(1)\ c(2)\ \cdots\ c(N)]^T$, $\mathbf{e} = [e(1)\ e(2)\ \cdots\ e(N)]^T$, the kernel matrix $\mathbf{G} = [\mathbf{g}_1\ \mathbf{g}_2\ \cdots\ \mathbf{g}_M]$ with $\mathbf{g}_l = [g_l(\mathbf{x}(1))\ g_l(\mathbf{x}(2))\ \cdots\ g_l(\mathbf{x}(N))]^T$, and the classifier weight vector $\mathbf{w} = [w_1\ w_2\ \cdots\ w_M]^T$. Let an orthogonal decomposition of $\mathbf{G}$ be $\mathbf{G} = \mathbf{P}\mathbf{A}$, where $\mathbf{A}$ is an $M \times M$ upper triangular matrix with unity diagonal elements, and

$$\mathbf{P} = [\mathbf{p}_1\ \mathbf{p}_2 \cdots \mathbf{p}_M] = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,M} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,M} \\ \vdots & \vdots & \vdots & \vdots \\ p_{N,1} & p_{N,2} & \cdots & p_{N,M} \end{bmatrix} \tag{23}$$

with orthogonal columns that satisfy $\mathbf{p}_i^T \mathbf{p}_j = 0$, if $i \ne j$. The model (22) can alternatively be expressed as

$$\mathbf{c} = \mathbf{P}\boldsymbol{\theta} + \mathbf{e} \tag{24}$$

where the "new" weight vector $\boldsymbol{\theta} = [\theta_1\ \theta_2\ \cdots\ \theta_M]^T$, satisfying the triangular system $\mathbf{A}\mathbf{w} = \boldsymbol{\theta}$.

A sparse l-term classifier model can be selected by incrementally maximizing a class separability measure in an OFS procedure [43], [44]. Define the two class sets $\mathcal{C}_\pm = \{\mathbf{x}(k) : c(k) = \pm 1\}$, and let the numbers of points in $\mathcal{C}_\pm$ be $N_\pm$, respectively, with $N_+ + N_- = N$. The means and variances of training samples belonging to classes $\mathcal{C}_\pm$ in the direction of basis $\mathbf{p}_l$ are given by

$$\begin{aligned} m_{+,l} &= \frac{1}{N_+} \sum_{i=1}^{N} \delta\left(c(i) - 1\right) p_{i,l} \\ \sigma_{+,l}^2 &= \frac{1}{N_+} \sum_{i=1}^{N} \delta\left(c(i) - 1\right) \left(p_{i,l} - m_{+,l}\right)^2 \end{aligned} \tag{25}$$

and

$$\begin{aligned} m_{-,l} &= \frac{1}{N_-} \sum_{i=1}^{N} \delta\left(c(i) + 1\right) p_{i,l} \\ \sigma_{-,l}^2 &= \frac{1}{N_-} \sum_{i=1}^{N} \delta\left(c(i) + 1\right) \left(p_{i,l} - m_{-,l}\right)^2 \end{aligned} \tag{26}$$

respectively, where $\delta(x) = 1$ for $x = 0$ and $\delta(x) = 0$ for $x \ne 0$. The Fisher ratio, which is defined as the ratio of the interclass difference and the intraclass spread, in the direction of $\mathbf{p}_l$ is given by [45]

$$F_l = \frac{(m_{+,l} - m_{-,l})^2}{\sigma_{+,l}^2 + \sigma_{-,l}^2}. \tag{27}$$

At the lth stage of incremental modeling, the lth kernel term is constructed by maximizing the Fisher ratio (27) with respect to the kernel mean vector $\boldsymbol{\mu}_l$ and the diagonal covariance matrix $\boldsymbol{\Sigma}_l$. The algorithm presented in Section II is used to perform this optimization task [42]. The forward selection procedure is terminated at the lth stage if

$$\frac{F_l}{\sum_{i=1}^{l} F_i} < \xi_c \tag{28}$$

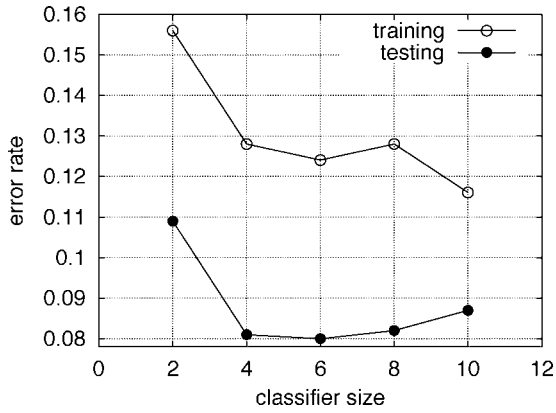


Fig. 13. Training and test error rates versus size of selected classifier for the synthetic data set using the OFS-RWBS algorithm.

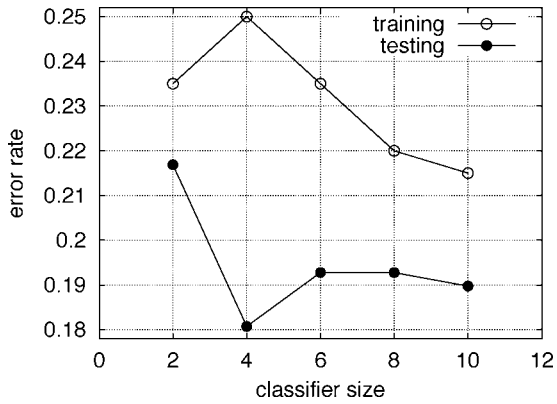


Fig. 14. Training and test error rates versus size of selected classifier for the Pima Diabetes data set using the OFS-RWBS algorithm.

TABLE II
COMPARISON OF CLASSIFICATION FOR THE SYNTHETIC DATA SET

	SVM	RVM	OFS-RWBS
classifier size	38	4	4
test error rate	10.6%	9.3%	8.1%

TABLE III
COMPARISON OF CLASSIFICATION FOR THE PIMA DIABETES DATA SET

	SVM	RVM	OFS-RWBS
classifier size	109	4	4
test error rate	20.1%	19.6%	18.1%

is satisfied, where the small positive scalar ξ_c is a chosen tolerance that determines the sparsity of the selected classifier model. The appropriate value for ξ_c is problem dependent and has to be found empirically. Alternatively, cross validation can be employed to terminate the OFS procedure. The least square solution for the corresponding sparse classifier weight vector $\mathbf{w}_l$ is readily available, given the least square solution of $\boldsymbol{\theta}_l$.

The synthetic two-class problem and Diabetes in Pima Indians, taken from [46], were used to investigate this proposed kernel classifier design approach, which is referred to as the OFS-RWBS, and to compare the results with those obtained using the existing state-of-the-art methods: the SVM and RVM [41]. The data sets were obtained from http://www.stats.ox.ac.uk/PRNN/.

Synthetic Data. The dimension of the feature space was $m = 2$. The training set contained 250 samples, and the test set had 1000 points. The optimal Bayes error rate for this example is around 8%. With the population size $P_S = 20$, number of weighted boosting iterations $N_B = 40$ (the inner loop simply runs N_B iterations), and number of generations $N_G = 20$, we applied the OFS-RWBS algorithm to the 250-sample training set, and Fig. 13 depicts the training and test error rates versus the size of the selected classifier. The result of Fig. 13 indicates that the four-term classifier is sufficient, and Table II compares this constructed four-term classifier with the results of using the SVM and RVM techniques given in [41]. It can be seen that the four-term classifier constructed by the OFS-RWBS algorithm achieves the optimal Bayes classification performance.

Pima Diabetes Data. The dimension of the input space was $m = 7$, the training data set contained 200 samples, and the test data set had 332 samples. With the population size $P_S = 50$, number of weighted boosting iterations $N_B = 80$, and number of generations $N_G = 20$ for the OFS-RWBS algorithm, Fig. 14 shows the training and test error rates versus the size of the constructed classifier, which clearly indicates that a four-term classifier is sufficient. Table III compares the performance of this constructed four-term classifier with those obtained by the SVM and RVM methods, quoted from [41]. The superior classification performance of the proposed design approach over the other two designs is self-evident.

IV. CONCLUSIONS

A guided random search optimization algorithm has been proposed. The local optimizer in this global search method evolves a population of the potential solutions by forming a convex combination of the solution population with boosting adaptation. A repeating loop involving a combined elitist and random sampling initialization strategy is adopted to guarantee a fast global convergence. The proposed guided random search method, referred to as the repeated weighted boosting search, is remarkably simple, involving minimum software programming effort, and can easily be adopted to a variety of practical applications. The versatility of the proposed method has been demonstrated in three different signal processing applications. In the two of these applications (IIR filter design and blind joint ML channel and data estimation), the proposed global search algorithm is seen to be as efficient as the GA and ASA in terms of global convergence speed, characterized by the total number of cost function evaluations required to attend a global optimal solution. In the third application, which is a novel incremental construction of sparse kernel classifiers, the proposed algorithm compares favorably with the existing state-of-the-art kernel classification techniques: the SVM and RVM. This study has demonstrated the potential of the proposed algorithm as a generic global optimizer, and further study is warranted to carried out an in-depth theoretical analysis as well as to compare it with other global optimization methods in a wider investigation.

REFERENCES

[1] F. Schoen, "Stochastic techniques for global optimization: A survey of recent advances," *J. Global Optimiz.*, vol. 1, pp. 207–228, 1991.

[2] B. C. Cetin, J. Barhen, and J. W. Burdick, "Terminal repeller unconstrained subenergy tunneling (TRUST) for fast global optimization," *J. Optimiz. Theory Applicat.*, vol. 77, no. 1, pp. 97–126, 1993.
[3] *Handbook of Global Optimization*, R. Horst and P. M. Pardalos, Eds., Kluwer, Dordrecht, The Netherlands, 1995.
[4] J. D. Pinter, *Global Optimization in Action*. Dordrecht, The Netherlands: Kluwer, 1996.
[5] C. A. Floudas, *Deterministic Global Optimization: Theory, Algorithms and Applications*. Dordrecht, The Netherlands: Kluwer, 1999.
[6] J. L. Maryak and D. C. Chin, "Efficient global optimization using SPSA," in *Proc. Amer. Control Conf.*, San Diego, CA, Jun. 1999, pp. 890–894.
[7] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.
[8] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.
[9] *Handbook of Genetic Algorithms*, L. Davis, Ed., Van Nostrand Reinhold, New York, 1991.
[10] K. F. Man, K. S. Tang, and S. Kwong, *Genetic Algorithms: Concepts and Design*. London, U.K.: Springer-Verlag, 1998.
[11] A. Corana, M. Marchesi, C. Martini, and S. Ridella, "Minimizing multimodal functions of continuous variables with the simulated annealing algorithm," *ACM Trans. Math. Softw.*, vol. 13, no. 3, pp. 262–280, 1987.
[12] L. Ingber and B. Rosen, "Genetic algorithms and very fast simulated reannealing: A comparison," *Math. Comput. Modeling*, vol. 16, no. 11, pp. 87–100, 1992.
[13] L. Ingber, "Simulated annealing: Practice versus theory," *Math. Comput. Modeling*, vol. 18, no. 11, pp. 29–57, 1993.
[14] ——, "Adaptive simulated annealing (ASA): lessons learned," *J. Control Cybern.*, vol. 25, no. 1, pp. 33–54, 1996.
[15] M. S. White and S. J. Flockton, "Genetic algorithms for digital signal processing," *Lecture Notes Comput. Sci.*, vol. 865, pp. 291–303, 1994.
[16] S. C. Ng, S. H. Leung, C. Y. Chung, A. Luk, and W. H. Lau, "The genetic search approach: A new learning algorithm for adaptive IIR filtering," *IEEE Signal Processing Magazine*, vol. 13, no. 6, pp. 38–46, 1996.
[17] S. Chen, Y. Wu, and S. McLaughlin, "Genetic algorithm optimization for blind channel identification with higher-order cumulant fitting," *IEEE Trans. Evol. Comput.*, vol. 1, no. 4, pp. 259–265, Nov. 1997.
[18] S. Chen and Y. Wu, "Maximum likelihood joint channel and data estimation using genetic algorithms," *IEEE Trans. Signal Processing*, vol. 46, no. 5, pp. 1469–1473, May 1998.
[19] J. Geigel and A. Loui, "Using genetic algorithms for album page layouts," *IEEE Multimedia*, vol. 10, no. 4, pp. 16–27, Oct./Dec. 2003.
[20] Y. H. Wang, W. L. Yan, and G. S. Zhang, "Adaptive simulated annealing for the optimal design of electromagnetic devices," *IEEE Trans. Magn.*, vol. 32, no. 3, pp. 1214–1217, May 1996.
[21] S. Chen and B. L. Luk, "Adaptive simulated annealing for optimization in signal processing applications," *Signal Process.*, vol. 79, no. 1, pp. 117–128, 1999.
[22] S. Chen, J. Wu, R. H. Istepanian, and J. Chu, "Optimizing stability bounds of finite-precision PID controller structures," *IEEE Trans. Autom. Control*, vol. 44, no. 11, pp. 2149–2153, Nov. 1999.
[23] P. Neumann and G. Muncill, "Using the adaptive simulated annealing algorithm to estimate ocean-bottom geoacoustic properties from measured and synthetic transmission loss data," *IEEE J. Ocean. Eng.*, vol. 29, no. 1, pp. 13–28, Jan. 2004.
[24] P. J. M. van Laarhoven and E. H. L. Aarts, *Simulated Annealing: Theory and Applications*. Dordrecht, The Netherlands: D. Reidel, 1987.
[25] R. E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, no. 2, pp. 197–227, 1990.
[26] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.
[27] L. Breiman, "Prediction games and arcing algorithms," *Neural Comput.*, vol. 11, no. 7, pp. 1493–1518, 1999.
[28] R. Meir and G. Rätsch, "An introduction to boosting and leveraging," in *Advanced Lectures in Machine Learning*, S. Mendelson and A. Smola, Eds. New York: Springer Verlag, 2003, pp. 119–184.
[29] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1985.
[30] J. J. Shynk, "Adaptive IIR filtering," *IEEE Acoust., Speech, Signal Process. Mag.*, vol. 6, no. 2, pp. 4–21, Apr. 1989.
[31] R. Nambiar, C. K. K. Tang, and P. Mars, "Genetic and learning automata algorithms for adaptive digital filters," in *Proc. ICASSP*, vol. IV, 1992, pp. 41–44.
[32] P. B. Wilson and M. D. Macleod, "Low implementation cost IIR digital filter design using genetic algorithms," in *Proc. Workshop Natural Algorithms in Signal Processing*, Chelmsford, U.K., 1993, pp. 4/1–4/8.
[33] S. Chen, "IIR model identification using batch-recursive adaptive simulated annealing algorithm," in *Proc. 6th Annu. Chinese Autom. Comput. Sci. Conf.*, Loughborough, U.K., Sep. 23, 2000, pp. 151–155.
[34] S. Chen, R. H. Istepanian, and B. L. Luk, "Digital IIR filter design using adaptive simulated annealing," *Digital Signal Process.*, vol. 11, no. 3, pp. 241–251, 2001.
[35] A. H. Gray Jr. and J. D. Markel, "Digital lattice and ladder filter synthesis," *IEEE Trans. Audio Electroacoust.*, vol. AU-21, no. 6, pp. 491–500, Dec. 1973.
[36] N. Seshadri, "Joint data and channel estimation using blind trellis search techniques," *IEEE Trans. Commun.*, vol. 42, no. 2/3/4, pp. 1000–1011, Feb./Mar./Apr. 1994.
[37] E. Zervas, J. Proakis, and V. Eyuboglu, "A quantized channel approach to blind equalization," in *Proc. Int. Contr. Conf.*, vol. 2, Chicago, IL, 1992, pp. 351.8.1–351.8.5.
[38] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.
[39] B. Schölkopf, K. K. Sung, C. J. C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, "Comparing support vector machines with Gaussian kernels to radial basis function classifiers," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2758–2765, Nov. 1997.
[40] B. Schölkopf and A. J. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2002.
[41] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Machine Learning Res.*, vol. 1, pp. 211–244, 2001.
[42] S. Chen, X. X. Wang, X. Hong, and C. J. Harris, "Kernel classifier construction using orthogonal forward selection and boosting with Fisher ratio class separability measure," *IEEE Trans. Neural Networks*, submitted for publication.
[43] K. Z. Mao, "RBF neural network center selection based on Fisher ratio class separability measure," *IEEE Trans. Neural Networks*, vol. 13, no. 5, pp. 1211–1217, Sep. 2002.
[44] S. Chen, L. Hanzo, and A. Wolfgang, "Kernel-based nonlinear beamforming construction using orthogonal forward selection with Fisher ratio class separability measure," *IEEE Signal Process. Lett.*, vol. 11, no. 5, pp. 478–481, May 2004.
[45] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.
[46] B. D. Ripley, *Pattern Recognition and Neural Networks*. Cambridge, U.K.: Cambridge Univ. Press, 1996.

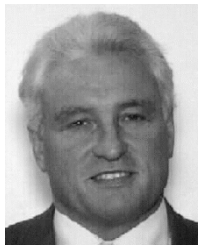
Sheng Chen (SM'97) received the B.Eng. degree in control engineering from the East China Petroleum Institute, Dongying, China, in 1982 and the Ph.D. degree in control engineering from the City University, London, U.K., in 1986.

He joined the School of Electronics and Computer Science, University of Southampton, in September 1999. He previously held research and academic appointments at the University of Sheffield, Sheffield, U.K.; the University of Edinburgh, Edinburgh, U.K.; and the University of Portsmouth, Portsmouth, U.K. His recent research works include adaptive nonlinear signal processing, wireless communications, modeling and identification of nonlinear systems, neural networks and machine learning, finite-precision digital controller design, evolutionary computation methods, and optimization. He has published over 240 research papers.

Dr. Chen is on the list of the highly cited researchers in the category that covers all branches of engineering subjects in the database of the world's most highly cited researchers in various disciplines, compiled by Institute for Scientific Information (ISI) of the USA.

Xunxian Wang received the Ph.D. degree in control theory and applications from Tsinghua University, Beijing, China, in July 1999.

From August 1999 to August 2001, he was a postdoctoral researcher with the State Key Laboratory of Intelligent Technology and Systems, Beijing. From September 2001 to December 2004, he was a research fellow with the University of Portsmouth, Portsmouth, U.K. He is currently a research fellow with Neural Computing Research Group, Aston University, Birmingham, U.K. His main interests are in machine learning and neural networks, control theory and systems, as well as robotics.



Chris J. Harris received the B.Sc. degree from the University of Leicester, Leicester, U.K., the M.A. degree from the University of Oxford, Oxford, U.K., and the Ph.D. degree from the University of Southampton, Southampton, U.K.

He previously held appointments at the University of Hull, Hull, U.K.; the University of Manchester Institute of Science and Technology, Manchester, U.K.; the University of Oxford, Oxford, U.K., and the University of Cranfield, Cranfield, U.K., as well as being employed by the U.K. Ministry of Defense. He returned to the University of Southampton as the Lucas Professor of Aerospace Systems Engineering in 1987 to establish the Advanced Systems Research Group and, more recently, the Image, Speech, and Intelligent Systems Research Group. His research interests lie in the general area of intelligent and adaptive systems theory and its application to intelligent autonomous systems such as autonomous vehicles, management infrastructures such as command and control, intelligent control, and estimation of dynamic processes, multisensor data fusion, and systems integration. He has authored and co-authored 12 research books and over 300 research papers, and he is the associate editor of numerous international journals.

Dr. Harris was elected to the Royal Academy of Engineering in 1996, was awarded the IEE Senior Achievement medal in 1998 for his work in autonomous systems, and the 2001 IEE Faraday medal, which is the highest international award of the IEE, for his work in intelligent control and neurofuzzy systems.