IIR MODEL IDENTIFICATION USING BATCH-RECURSIVE ADAPTIVE SIMULATED ANNEALING ALGORITHM

Sheng Chen

Department of Electronics and Computer Science
University of Southampton, Highfield, Southampton SO17 1BJ, U.K.
Tel./Fax: 023 8059 6660/4508 Email: sqc@ecs.soton.ac.uk

ABSTRACT

System identification using infinite-impulse-response (IIR) model is considered. Because the error surface of IIR filters is generally multi-modal, global optimisation techniques are preferred in order to avoid local minima. An efficient global optimisation method, called the adaptive simulated annealing (ASA), is adopted, and a new batch-recursive ASA algorithm is developed for on-line identification. Simulation study shows that the proposed approach is accurate and has a fast convergence rate, and the results obtained demonstrate that the ASA offers a viable tool to IIR model identification.

Keywords – System identification, IIR filter, global optimisation, adaptive simulated annealing, genetic algorithms.

1. INTRODUCTION

Adaptive IIR filtering has been an active area of research for many years [1, 2]. A major concern in IIR filtering applications is that the cost function is generally multi-modal with respect to the filter coefficients, and the gradient-based algorithm can easily be stuck at local minima. In order to achieve a global minimum solution, global optimisation techniques are needed. Global optimisation methods require extensive computations and are batch-type algorithms, as the cost function is usually evaluated on a block of data. In contrast, gradient learning can be implemented recursively to update the filter coefficients as each new data sample is acquired. Despite of these drawbacks, applying global optimisation methods to IIR filter design is attractive, since in many applications a global optimal solution can be much better than local optimal ones.

When considering global optimisation methods for IIR filter design, the genetic algorithm (GA) [3, 4, 5] seems to have attracted the main attention [6, 7, 8]. Simulated annealing (SA) [9, 10, 11] by contrast has not received similar interests. The ASA [12, 13, 14, 15], an improved version of SA, is known to provide significant improvement in convergence speed over standard versions of SA. This study investigates the use of the ASA to IIR system identification. A new batch-recursive ASA is proposed for adaptive applications. Simulation results confirm that the efficiency of the ASA appears to be in the same orders as GAs. This suggests that the ASA offers a viable alternative to IIR model identification.

2. IIR SYSTEM MODEL IDENTIFICATION

System model identification using an IIR filter is depicted in Fig. 1. The IIR filter is governed by the difference equation:

$$y(k) + \sum_{i=1}^{M} b_i y(k-i) = \sum_{i=0}^{L} a_i x(k-i), \tag{1}$$

where $x(k)$ and $y(k)$ are the filter's input and output, respectively, and M $(\geq L)$ is the filter order. The transfer function of this IIR filter is:

$$H_M(z) = \frac{A(z)}{B(z)} = \frac{\sum_{i=0}^{L} a_i z^{-i}}{1 + \sum_{i=1}^{M} b_i z^{-i}}. \tag{2}$$

The unknown plant has a transfer function $H_S(z)$ to be identified using $H_M(z)$. The task is formulated as an optimisation problem with the mean square error (MSE) as

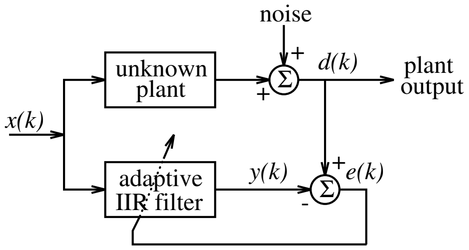

Figure 1: Adaptive IIR filter for system identification.

the cost function:

$$J(\mathbf{w}_H) \triangleq \mathrm{E}[e^2(k)] \approx \frac{1}{N}\sum_{k=1}^{N} e^2(k). \qquad (3)$$

where $d(k)$ is the desired response, $e(k) = d(k) - y(k)$ is the error signal,

$$\mathbf{w}_H = [\mathbf{a}^T \ \mathbf{b}^T]^T = [a_0 \ a_1 \cdots a_L \ b_1 \cdots b_M]^T \qquad (4)$$

denotes the filter coefficient vector, and N is the number of samples used to approximate ensemble operation. When the filter order M is smaller than the system order, local minima problems can be encountered [2], giving rise to a multi-modal cost function J.

An important consideration during the adaptive process is to maintain the stability. An efficient way is to convert the direct form of IIR filter (1) to the lattice form [16] and to make sure that all the reflection coefficients κ_i, $0 \le i \le M-1$, have magnitudes less than 1. Thus the actual filter coefficient vector used in optimisation is:

$$\mathbf{w} = [a_0 \ a_1 \cdots a_L \ \kappa_0 \cdots \kappa_{M-1}]^T = [w_1 \cdots w_D]^T, \qquad (5)$$

where $D = M + L + 1$ is the dimension of the filter coefficient vector. Converting the reflection coefficients back to b_i, $1 \le i \le M$, is straightforward [16].

3. THE ASA ALGORITHM

Theoretic fundamentals and convergence analysis of the ASA algorithm can be found in [12, 13, 14]. The appendix summarizes the search mechanisms of the ASA method. An implementation of the ASA, shown in Fig. 2, is detailed:

(i) Initialisation An initial $\mathbf{w}$ is randomly generated, the initial temperature of the acceptance probability function, $T_c(0)$, is set to the initial value of the cost function $J(\mathbf{w})$, and the initial temperatures of the parameter generating probability functions, $T_i(0)$, $1 \le i \le D$, are set to 1.0. A control parameter c in annealing process is given, and the annealing times, k_i for $1 \le i \le D$ and k_c, are all set to 0.

(ii) Generating The algorithm generates a new point in the parameter space with:

$$w_i^{\text{new}} = w_i^{\text{old}} + g_i\,(U_i - V_i) \ \text{and} \quad w_i^{\text{new}} \in [U_i,\ V_i],\ 1 \le i \le D. \qquad (6)$$

Here U_i and V_i are the lower and upper bounds for w_i, g_i is calculated as

$$g_i = \mathrm{sgn}\left(u_i - \frac{1}{2}\right) T_i(k_i) \left(\left(1 + \frac{1}{T_i(k_i)}\right)^{|2u_i - 1|} - 1 \right), \qquad (7)$$

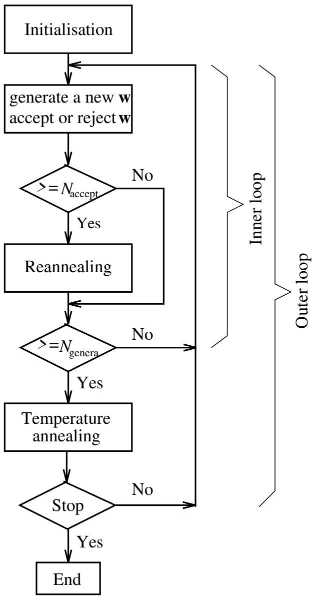

Figure 2: Flow chart of the adaptive simulated annealing.

and u_i a uniformly distributed random variable in $[0,\ 1]$. The value of the cost function $J(\mathbf{w}^{\text{new}})$ is then evaluated and the acceptance probability function of $\mathbf{w}^{\text{new}}$ is given by

$$P_{\text{accept}} = \frac{1}{1 + \exp\left(\left(J(\mathbf{w}^{\text{new}}) - J(\mathbf{w}^{\text{old}})\right)/T_c(k_c)\right)}. \qquad (8)$$

A uniform random variable P_{unif} is generated in $[0,\ 1]$. If $P_{\text{unif}} \le P_{\text{accept}}$, $\mathbf{w}^{\text{new}}$ is accepted; otherwise it is rejected.

(iii) Reannealing After every N_{accept} acceptance points, calculating the sensitivities:

$$s_i = \left| \frac{J(\mathbf{w}^{\text{best}} + \mathbf{1}_i\,\delta) - J(\mathbf{w}^{\text{best}})}{\delta} \right|, \ 1 \le i \le D, \qquad (9)$$

where $\mathbf{w}^{\text{best}}$ is the best point found so far, δ is a small step size, the D-dimensional vector $\mathbf{1}_i$ has unit ith element and the rest of elements of $\mathbf{1}_i$ are all zeros. Let $s_{\max} = \max\{s_i,\ 1 \le i \le D\}$. Each T_i is scaled by a factor $s_{\max}/s_i$ and the annealing time k_i is reset:

$$T_i(k_i) = \frac{s_{\max}}{s_i} T_i(k_i),\ k_i = \left(-\frac{1}{c}\log\left(\frac{T_i(k_i)}{T_i(0)}\right)\right)^D. \qquad (10)$$

Similarly, $T_c(0)$ is reset to the value of the last accepted cost function, $T_c(k_c)$ is reset to $J(\mathbf{w}^{\text{best}})$ and the annealing time k_c is rescaled accordingly:

$$k_c = \left(-\frac{1}{c}\log\left(\frac{T_c(k_c)}{T_c(0)}\right)\right)^D. \qquad (11)$$

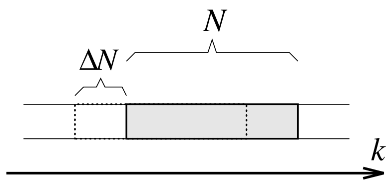

Figure 3: Batch-recursive moving window. The data block contains N samples and, after every ΔN cost-function evaluations, the data block is shifted by ΔN samples.

(iv) Annealing After every N_{genera} generated points, annealing takes place with

$$\left.\begin{array}{l} k_i = k_i + 1 \\ T_i(k_i) = T_i(0)\exp\left(-ck_i^{\frac{1}{D}}\right) \end{array}\right\} \quad 1 \le i \le D \tag{12}$$

and

$$\left.\begin{array}{l} k_c = k_c + 1 \\ T_c(k_c) = T_c(0)\exp\left(-ck_c^{\frac{1}{D}}\right) \end{array}\right\}. \tag{13}$$

Otherwise, goto step (ii).

(v) Termination The algorithm is terminated if the parameters has remained unchanged for a few successive reannealings or a preset maximum number of cost function evaluations has been reached; Otherwise, goto step (ii).

The inner loop in the ASA ensures that the parameter space is searched sufficiently at a given temperature, which is necessary for the algorithm to find a global optimum. The user only needs to assign a control parameter c and set two values N_{accept} and N_{genera}. The above ASA is a batch algorithm. For adaptive applications, it is desired to have some tracking capability. This can be achieved by employing a moving window scheme, as illustrated in Fig. 3. The cost-function is evaluated using a block of N samples and the data block is shifted by ΔN samples after every ΔN cost-function evaluations. This version of the algorithm will be referred to as the batch-recursive algorithm.

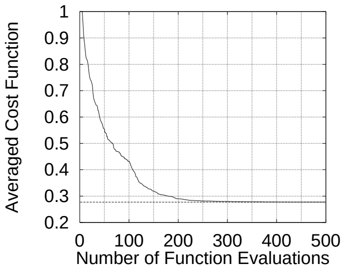

Figure 4: Normalized cost function versus number of cost function evaluations averaged over 100 runs of the batch ASA for Example 1. The dashed line indicates the global minimum.

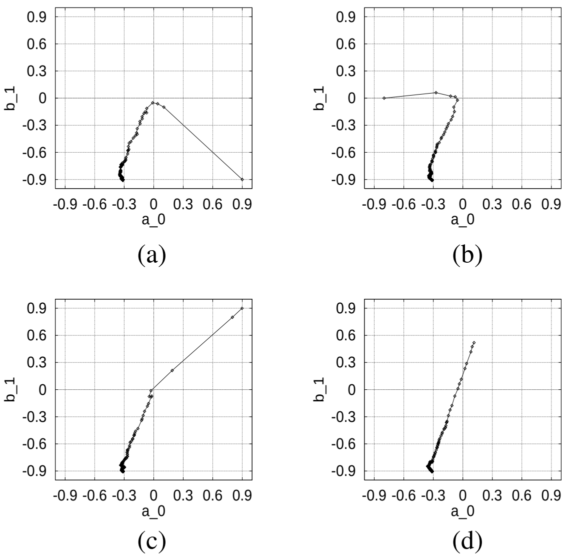

Figure 5: Trajectories of the filter parameter vector averaged over 100 different runs of the batch ASA, started from the fixed initial positions: (a) $[0.9\ -0.9]^T$, (b) $[-0.8\ 0.0]^T$, (c) $[0.9\ 0.9]^T$ and (d) $[0.114\ 0.519]^T$, for Example 1.

4. SIMULATION EXAMPLES

Example 1. This example is taken from [2]. The system and filter transfer functions are respectively

$$H_S(z) = \frac{0.05 - 0.4z^{-1}}{1 - 1.1314z^{-1} + 0.25z^{-2}}, \tag{14}$$

$$H_M(z) = \frac{a_0}{1 + b_1 z^{-1}}. \tag{15}$$

The analytical cost function J is known when the input is a white sequence and the system noise variance $\sigma_n^2 = 0$. The cost function has a global minimum at $\mathbf{w}^{\mathrm{global}} = [-0.311\ -0.906]^T$ with the normalized cost function value 0.2772 and a local minimum at $\mathbf{w}^{\mathrm{local}} = [0.114\ 0.519]^T$. Fig. 4 depicts the evolution of the normalized cost function averaged over 100 different random runs of the batch ASA. Each run had a randomly chosen initial $\mathbf{w}$ and a random algorithm setting. Fig. 5 shows the trajectories of the filter parameter vector averaged over 100 different runs of the batch ASA, started from four fixed initial positions. It can be seen that the batch ASA consistently found the global optimal solution and the algorithm converged after 300 function calls.

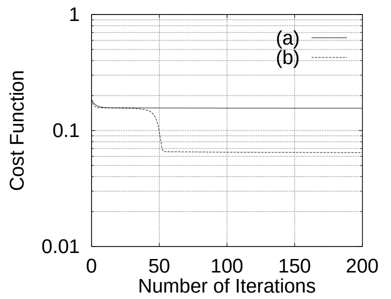

Figure 6: Convergence behaviours of the batch gradient algorithm, started from the two initial conditions: (a) $[0.0\ 0.0\ 0.3\ 0.1]^T$ and (b) $[0.0\ 0.0\ 0.3\ 0.0]^T$, for Example 2.

Example 2 This is a $3rd$ order system with the transfer function given by

$$H_S(z) = \frac{-0.3 + 0.4z^{-1} - 0.5z^{-2}}{1 - 1.2z^{-1} + 0.5z^{-2} - 0.1z^{-3}}. \qquad (16)$$

In the simulation, the system input $x(k)$ was a uniformly distributed white sequence, taking values from $(-0.5,\ 0.5)$, and the signal to noise ratio was SNR=30 dB. When an IIR filter model with $M = 2$ and $L = 1$ was used, the MSE was multi-modal and this was demonstrated by the batch gradient algorithm which converged to the two final states, depending on the initial conditions, as illustrated in Fig. 6. The batch ASA consistently reached the global optimal solution, as shown in Fig. 7. The batch-recursive ASA algorithm was also tested for this example. A moving window of $N = 100$ and $\Delta N = 1$ was used. Fig. 8 plots the evolution of the cost function averaged over 100 different random runs of the batch-recursive ASA. Again each run had a randomly chosen initial $\mathbf{w}$ and a random algorithm setting. It can be seen that the batch-recursive ASA algorithm consistently converges to the global minimum.

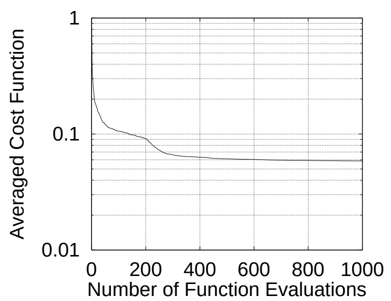

Figure 7: Cost function versus number of cost function evaluations averaged over 100 random runs of the batch ASA for Example 2. Each run had a randomly chosen initial point.

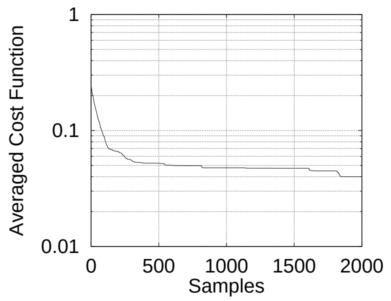

Figure 8: Cost function versus number of cost-function evaluations averaged over 100 random runs of the batch-recursive ASA for Example 2. A moving window of $N = 100$ and $\Delta N = 1$ was used. Each run had a randomly chosen initial point.

5. CONCLUSIONS

Although the IIR system model identification is a well researched area, major difficulties still exist in practice. An efficient global optimisation method known as the ASA has been applied to overcome the problems associated with local minima. A batch-recursive ASA algorithm is proposed for on-line applications. Simulation study has demonstrated that the ASA is robust and has a fast convergence speed. Compared with the results of using GAs for adaptive IIR filtering available in the literature, the efficiency of the ASA appears at least to be in the same order as GAs. This study has confirmed that the ASA offers an alternative design approach for IIR filtering.

Appendix: Search guiding mechanisms

The ASA is a global optimization scheme for solving for the following general optimization problem:

$$\min_{\mathbf{w} \in \mathcal{W}} J(\mathbf{w}) \, . \qquad (17)$$

It evolves a single point $\mathbf{w}$ in the parameter or state space $\mathcal{W}$. The seemingly random search is guided by certain underlying probability distributions.

1. Generating probability density function

$$G(w_i^{\mathrm{old}}, w_i^{\mathrm{new}}, T_i; 1 \le i \le D) \, . \qquad (18)$$

This determines how a new state $\mathbf{w}^{\mathrm{new}}$ is created, and from what neighbourhood and probability distributions it is generated, given the current state $\mathbf{w}^{\mathrm{old}}$. The generating "temperatures" T_i describe the widths or scales of the generating distribution along each dimension w_i of the state space.

Often a cost function has different sensitivities along different dimensions of the state space. Ideally, the generating distribution used to search a steeper and more sensitive dimension should have a narrower width than that of the distribution used in searching a dimension less sensitive to change. The ASA adopts a so-called reannealing scheme to periodically re-scale T_i, so that they optimally adapt to the current status of the cost function. This is an important mechanism, which not only speeds up the search process but also makes the optimization process robust to different problems.

2. Acceptance function

$$P_{\text{accept}}(J(\mathbf{w}^{\text{old}}), J(\mathbf{w}^{\text{new}}), T_c)\,. \tag{19}$$

This gives the probability of $\mathbf{w}^{\text{new}}$ being accepted. The acceptance temperature T_c determines the frequency of accepting new states of poorer quality.

Probability of acceptance is very high at very high temperature T_c, and it becomes smaller as T_c is reduced. At every acceptance temperature, there is a finite probability of accepting the new state. This produces occasionally uphill move, enables the algorithm to escape from local minima, and allows a more effective search of the state space to find a global minimum. The ASA also periodically adapts T_c to best suit the status of the cost function. This helps to improve convergence speed and robustness.

3. Reduce temperatures or annealing schedule

$$\left.\begin{array}{l} T_c(k_c) \longrightarrow T_c(k_c+1) \\ T_i(k_i) \longrightarrow T_i(k_i+1),\ 1 \le i \le D \end{array}\right\}, \tag{20}$$

where k_c and k_i are some annealing time indexes. The reduction of temperatures should be sufficiently gradual in order to ensure that the algorithm finds a global minimum.

This mechanism is based on the observations of the physical annealing process. When the metal is cooled from a high temperature, if the cooling is sufficiently slow, the atoms line themselves up and form a crystal, which is the state of minimum energy in the system. The slow convergence of many SA algorithms is rooted at this slow annealing process. The ASA, however, can employ a very fast annealing schedule, as it has self adaptation ability to re-scale temperatures.

6. REFERENCES

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

[2] Shynk, J.J. Adaptive IIR filtering. *IEEE ASSP Magazine*. April (1989), pp.4–21.

[3] Holland, J.H. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.

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

[5] Davis, L. (Ed.) *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, 1991.

[6] Nambiar, R., Tang, C.K.K., and Mars, P. Genetic and learning automata algorithms for adaptive digital filters. In *Proc. ICASSP*, 1992, Vol.IV, pp.41–44.

[7] Wilson, P.B., and Macleod, M.D. Low implementation cost IIR digital filter design using genetic algorithms. In *Workshop on Natural Algorithms in Signal Processing*, Chelmsford, Essex, 1993, pp.4/1–4/8.

[8] Ng, S.C., Leung, S.H., Chung, C.Y., Luk, A., and Lau, W.H. The genetic search approach: a new learning algorithm for adaptive IIR filtering. *IEEE Signal Processing Magazine*. November (1996), pp.38–46.

[9] Kirkpatrick, S., Gelatt, Jr., C.D., and Vecchi, M.P. Optimization by simulated annealing. *Science*. **220** (1983), pp.671–680.

[10] Corana, A., Marchesi, M., Martini, C., and Ridella, S. Minimizing multimodal functions of continuous variables with the simulated annealing algorithm. *ACM Trans. Mathematical Software*. **13** (1987), pp.262–280.

[11] van Laarhoven, P.J.M., and Aarts, E.H.L. *Simulated Annealing: Theory and Applications*. D. Reidel, Dordrecht, Netherlands, 1987.

[12] Ingber, L., and Rosen, B. Genetic algorithms and very fast simulated reannealing: a comparison. *Mathematical and Computer Modelling*. **16** (1992), pp.87–100.

[13] Ingber, L. Simulated annealing: practice versus theory. *Mathematical and Computer Modelling*. **18** (1993), pp.29–57.

[14] Ingber, L. Adaptive simulated annealing (ASA): lessons learned. *J. Control and Cybernetics*. **25** (1996), pp.33-54.

[15] Chen, S., Luk, B.L., Liu, Y. Application of adaptive simulated annealing to blind channel identification with HOC fitting. *Electronics Letters*. **34** (1998), pp.234–235.

[16] Gray, Jr., A.H., and Markel, J.D., Digital lattice and ladder filter synthesis. *IEEE Trans. Audio and Electroacoustics*. **AU-21** (1973), pp.491–500.