Genetic Algorithms for Digital Signal Processing

Conference Paper · August 2000

DOI: 10.1007/3-540-58483-8_22 · Source: CiteSeer

2 authors, including:

Stuart Flockton
Royal Holloway, University of London

Genetic Algorithms for Digital Signal Processing

Michael S. White and Stuart J. Flockton

Physics Department, Royal Holloway (University of London), Egham, Surrey
TW20 OEX, UK

Abstract. Recursive digital filters are potentially less computationally expensive than their non-recursive counterparts. However, algorithms for adjusting the coefficients of recursive filters may produce biased or sub-optimal estimates of the optimal coefficent values. In addition, recursive filters may become unstable if the adaptive algorithm updates a feedback coefficient so that one of the poles remains outside the unit circle for any length of time. This paper details an adaptive algorithm for optimizing the coefficients of recursive digital filters based on the genetic algorithm. Stability considerations are addressed by implementing the population of adaptive filters as lattice structures which allows the entire feasible, stable coefficient space to be searched whilst ensuring that crossover and mutation do not produce invalid (unstable) filters. Results are presented showing the application of this technique to the tasks of system identification and adaptive data equalization.

1 Introduction

Adaptive filtering techniques have been applied to many important areas of digital signal processing. Recursive digital filters offer potential computational savings over non-recursive filters but conventional adaptive algorithms for recursive filters suffer a number of drawbacks. Algorithms utilizing the equation-error formulation may converge on biased estimates of the optimal filter coefficients whilst output-error approaches can produce multi-modal error surfaces, leading to the possible entrapment of gradient-based searches in local minima. This paper reports on the use of genetic algorithms for adapting recursive digital filters for a variety of different tasks.

A brief introduction to digital signal processing and adaptive filtering is given in Section 2. This is followed by a review of the application of genetic algorithms to the problem of modelling unknown systems and Section 4 details work carried out by the authors in this same area. Section 5 introduces the task of data equalization and simulation results of a genetic algorithm approach to this problem are presented in Section 6. The paper concludes with a summary and details areas for further study.

2 Digital Signal Processing

Digital Signal Processing (DSP) is used to transform and analyze data and signals that are either inherently discrete or have been sampled from analogue sources. With the availability of cheap but powerful general-purpose computers and custom-designed DSP chips, digital signal processing has come to have a great impact on many different disciplines from electronic and mechanical engineering to economics and meteorology. In the field of biomedical engineering, for example, digital filters are used to remove unwanted 'noise' from electrocardiograms (EKG) while in the area of consumer electronics DSP techniques have revolutionised the recording and playback of audio material with the introduction of compact disk and digital audio tape technology.

Any DSP algorithm or processor can be reasonably described as a filter. Digital filters may be divided into recursive and non-recursive categories depending on their use of feedback. The response of non-recursive, or FIR filters is dependent only upon present and previous values of the input signal. Recursive, or IIR filters, however, depend not only upon the input data but also upon one or more previous output values. As a consequence of this feedback, recursive filters with just a few coefficients are often able to obtain similar output characteristics to non-recursive filters requiring (say) 100 or more coefficients. This potentially greater computational efficiency of recursive filters over their non-recursive counterparts is tempered by several possible shortcomings. Firstly, recursive filters may become unstable, that is, their output may grow without limit, if the feedback coefficients are chosen incorrectly. In addition, recursive filters are, in general, unable to produce the linear-phase responses achievable by non-recursive filter implementations and the presence of feedback may have an adverse effect on the accuracy to which the filter coefficients need to be specified.

The design of a conventional digital signal processor requires a priori knowledge about the statistics of the data to be processed. When this information is inadequate or when the statistical characteristics of the input data are known to change with time, adaptive filters are used. Adaptive filters have the property of self-optimization. They consist, primarily, of a time-varying filter, characterised by a set of adjustable coefficients and a recursive algorithm which updates these coefficients as more information concerning the statistics of the relevant signals is learned. Most current applications of adaptive signal processing (the modelling of unknown systems, echo cancellation and the digital representation of speech etc) utilise non-recursive digital filters. Since non-recursive filters do not have feedback, the output is a linear function of the coefficients and this greatly simplifies the derivation of gradient-based adaptive algorithms.

The greater computational efficiency that recursive filters offer has led researchers to try and develop reliable algorithms for their adaptation. Fundamentally, two approaches to the problem of adaptive IIR filtering have been investigated (Shynk, 1989). In both cases, after each iteration of the adaptive algorithm the performance of the digital filter is assessed on the basis of some optimization criterion, commonly some function of the total or mean squared error. The two approaches differ in the formulation of this prediction error. The equation-error formulation effectively transforms the recursive filter into two coupled non-recursive filters, allowing well-understood FIR adaptive algorithms to be used. Unfortunately, in the presence of noise, adaptive algorithms based on this formulation can converge to biased estimates of the filter coefficients. The second approach, known as the output-error formulation adjusts the coefficients of the time-varying digital filter directly in recursive form. The output of a recursive filter is a non-linear function of the coefficients. Consequently, the prediction error is not a quadratic function and may have multiple local minima. Adaptive algorithms based on gradient-search methods, such as the widely used LMS, may then converge to sub-optimal estimates of the filter coefficients. This paper seeks to introduce a class of adaptive algorithms based on the natural processes of evolution and population genetics which can overcome some of these problems.

3 Genetic Algorithm Approaches to System Identification

Many problems in the areas of adaptive control and signal processing can be reduced to that of system identification. In this task, a system with adjustable coefficients is used to model the dynamics of an unknown system also known as the plant. The model is frequently a non-recursive filter in order that conventional adaptive algorithms based on least-squares or gradient techniques may be used to produce the best estimate of the unknown system. Approaches to system identification which employ an IIR filter system model seek to benefit from the computational economy that they offer. In (Etter et al. 1982) a genetic adaptive algorithm was used to adjust the coefficients of a population of 11 recursive adaptive filters. Each system model was represented as a binary string by the genetic algorithm. Every generation, 10 pairs of these filters were probabilistically chosen to undergo single-point crossover on the basis of their ability to model the dynamics of the unknown system. A single filter was similarly chosen to undergo mutation. This scheme was shown to be able to correctly identify simple systems (first or second order) whose response surfaces were either uni- or bimodal.

Nambiar et al (1992) demonstrated similar performance for their GA-based adaptive algorithm. In this case, a parallel filter realization was implemented in order that the stability of the system models could be monitored. Several 'unknown' systems, from fourth, to tenth order were used and the effect of varying some of the genetic algorithm parameters (population size, crossover and mutation rates) was investigated. Kristinsson and Dumont (1992) applied a genetic algorithm to the identification of both discrete and continuous time systems. Second-order systems with minimum and non-minimum phase characteristics were modelled by IIR adaptive filters realized as cascade structures. Their results indicated that in some cases the genetic adaptive algorithm converged on solutions which were biased from those of the system and they attributed this to an insensitivity to changes in the biased coefficients. A population size of 100 was implemented and the single-point crossover and mutation operators were applied at rates of 0.8 and 0.01 respectively. During a run, if the percent involvement (the proportion of the current population producing offspring) declined significantly, as would be the case when a few 'super' individuals were receiving most of the opportunities to reproduce, a fitness ranking scheme was implemented. This limited the number of offspring that any individual model could produce and helped to maintain population diversity.

4 A Genetic Adaptive Lattice Algorithm

Earlier work by the authors (1993) addressed the problem of maintaining the stability of the genetically-derived adaptive filters by realizing them as IIR lattice structures. Monitoring the stability of conventional, direct-form filter implementations is computationally expensive and generally not robust. Alternative filter realizations such as the parallel and cascade forms used in the work detailed in Section 3 simplify stability monitoring by reducing the problem to one of ensuring that the coefficients of the cascaded second-order filter sections lie within the 'stability triangle'. The lattice realization enables stability to be controlled even more simply as it can be shown that a necessary and sufficient condition for the filter to remain stable, is for the feedback coefficients to have magnitude less than unity. The structure of a lattice filter is illustrated below in Figure 1.

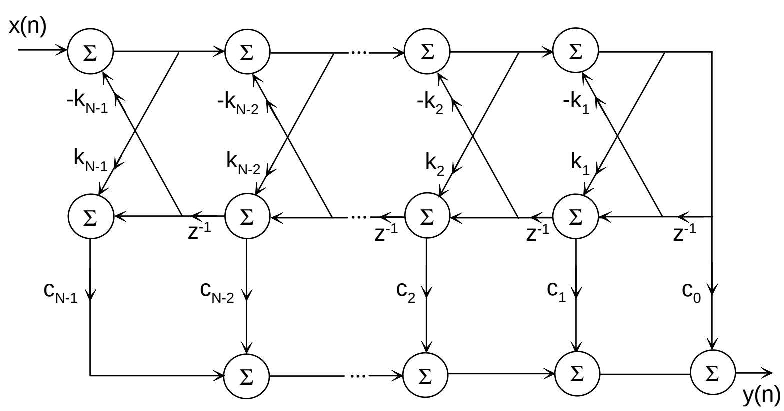

Fig. 1. Structure of a lattice filter

This genetic adaptive algorithm differs from conventional adaptive algorithms in its population-based approach. Rather than adjusting the filter coefficients of a single filter, the genetic algorithm operates on a number of points in the search space simultaneously (at least in principle — simulation on a serial machine means that all evaluations have to take place sequentially). Each lattice filter is represented as a bit-string. These strings are constructed by quantizing the feedback and feedforward coefficients (the k's and c's in Figure 1) of a single filter and concatenating them to form a binary code. The initial population of filters is randomly generated. Each filter is then evaluated according to its ability to accurately model the plant. On the basis of this fitness value, selection probabilities are generated, with filters corresponding to smaller error values being assigned a proportionally higher selection probability. Subsequent generations are generated by selecting members of the current population according to their assigned probability and applying genetic operators to a proportion of these filter structures in order to introduce variation.

The lattice structure is ideally suited to adaptation by a genetic algorithm. The maintenance of filter stability is, in essence, achieved 'for free' as the GA necessarily specifies a range onto which the quantized coefficients are mapped (decoded) when evaluation of the filter structures takes place. This is in marked contrast to other, non-lattice structures which require factorization of polynomials and schemes to regain stability or unduly restrict the range of values which the feedback coefficients can take. Because the stability criterion is built into the coefficient decoding mechanism the mutation and crossover operators are unable to generate invalid (unstable) filters, consequently, there are no constraint violations to be dealt with.

The two-point crossover mechanism used in these simulations exchanges portions of the 'genetic material' (binary strings) of two randomly selected parents in order to create a pair of new filter structures. This is accomplished by randomly selecting two cut-points on the parent bit strings and crossing-over the bits in between these points. In the real filter coefficient space this has one of two possible effects. Should the cut-points fall between the binary codes for two coefficients, the child structures each receive some of the coefficients of the two parent structures. If, however, one or both of the cut-points falls within the code for a coefficient then a child receives the most significant part of the binary-encoded coefficient of one parent and the least significant part from the other. This can be viewed as a combining of coefficient values from the two parents along with a perturbation of the coefficient within which a cut-point falls. The mutation operator is used to find new points in the search space to evaluate and acts by flipping randomly chosen bits in the binary strings. In the real coefficient space this has the effect of perturbing the coefficient within which the mutation occurs, the size of perturbation relating to the bit or bits mutated. This evaluation-selection-variation cycle is repeated either for a fixed number of iterations (generations) or until the unknown system has been modelled to the desired accuracy.

Initial experiments sought to investigate the suitability of lattice filters for genetic adaptation. The example given in (Johnson and Larimore 1977) uses an adaptive filter model of lower order than the plant, to produce an error surface which has both a local and a global minimum. This example was originally conceived to demonstrate the inability of the recursive LMS algorithm to locate the optimal coefficients of the plant and has since been used to highlight the inadequacies of true gradient algorithms when started from within the basin of attraction of a local minimum. Figure 2 shows the results of a simulation run using a genetic adaptive algorithm to identify the system described above. In this graph, the normalised squared error in decibels (the mean of ten runs) is plotted against the generation number. The dB reference level was taken as the the mean squared error generated by an adaptive filter with all coefficients set to zero. From the randomly initialized population of 40 lattice filters, the genetic algorithm requires three generations to concentrate its efforts about the two minima. Two generations later the local minimum has been abandoned and the entire population is clustered around the global minimum.

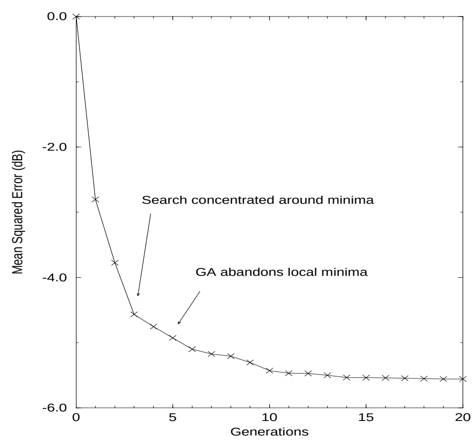

Fig. 2. Identification of example from Johnson and Larimore (1977)

Two methods have been used to characterize the dynamics of the unknown system. In the first, a unit impulse, which is finite at time $n = 0$ but zero elsewhere, is applied to the input of the unknown system. The output of the unknown system is then compared to that of each time-varying filter in turn in order to determine its fitness. As the 'excitation' is confined to the instant $n = 0$, any output signal observed after this time is characteristic of the system itself. This output is known as the impulse response and is of finite length for non-recursive digital filters (hence non-recursive filters are also known as finite impulse response or FIR filters). Conversely, the impulse response of a recursive or infinite impulse response (IIR) filter never decays exactly to zero. Although the impulse response uniquely characterizes a digital filter, certain practical problems arise in the production of a unit impulse for input to a real continuous-time system. As a consequence, simulations have also been undertaken using pseudo-random gaussian noise as the input to the plant and time-varying filters. These runs take significantly more generations for the genetic adaptive algorithm to converge on the optimal adaptive filter coefficients.

In an attempt to provide a comparison with other non-traditional adaptive algorithms some attempt was made at minimizing the number of impulse response samples that were used to characterize the plant and adaptive filters in this first simulation. Results seem to indicate that for the unknown system detailed above, impulse responses of less than approximately 40 samples cause the genetic adaptive algorithm to converge to sub-optimal values of the filter coefficients. This is because the unknown system is a recursive filter and thus has an infinitely long impulse response and cannot be uniquely characterized by a very short burst of this output signal. However, with 40 samples of impulse response being processed by each of the 30 time-varying lattice filters in the population, 1200 time samples are evaluated every generation. From Figure 2 above, 13 generations are required for convergence, on average, resulting in a total of 15,600 time samples being processed in each run. In comparison the Stochastic Learning Automata (SLA) approach of Nambiar et al. (1992) requires about 11,000 time samples to be processed in order to identify the same system. Whilst this means that on this problem the genetic adaptive algorithm performs no better than the SLA, genetic algorithms are thought to be able to tackle problems with high-dimensional search spaces which are difficult for the stochastic learning automata approach to solve. Additionally, the genetic algorithm requires only 40 different time samples as the same output signal is used in the evaluation of every generation. This could give a GA-based approach an advantage if the adaptation could be accomplished off-line.

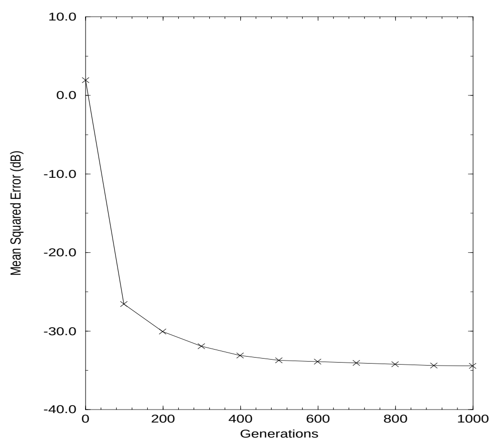

Fig. 3. Identification of sixth order plant

Subsequent simulations sought to test the ability of the genetic adaptive algorithm to optimize higher order adaptive filters. The plant in this case was a sixth order, low-pass Butterworth filter, providing a thirteen dimensional space for the genetic algorithm to search. Mutation and crossover rates were as in the previous example but a much larger population size (720) was found to be necessary. Figure 3 shows the mean squared error in dB (the dB reference level was derived as in the previous experiment) of ten independent runs plotted against the generation number. Convergence to a mean squared error (MSE) of $-30\,\mathrm{dB}$

was accomplished in just 200 generations but after this point the improvement slowed down dramatically, resulting in a final MSE value of $-40\,$dB after 10000 generations had been evaluated. The minimum mean squared error achievable with the 10 bit precision used to encode each filter coefficient is $-51\,$dB and this was achieved by one run out of the ten.

5 Intersymbol Interference and Data Equalization

In an analogue communications system information from analogue sources is transmitted directly over the communications channel using one of the conventional modulation techniques. In a digital communications system, analogue signals are converted into digital form prior to transmission. The most commonly employed technique for transmitting this digital information is known as pulse code modulation or PCM (Stremler 1990). This pulse modulation technique represents the amplitude of the analogue signal at regular sampling intervals as digital words in a serial bit stream. The transmission characteristics of most real communications channels are usually far from perfect. Twisted pairs of wires, coaxial cable or radio channels all have nonideal frequency response characteristics and may introduce noise or interference which will corrupt the signal transmitted through the channel. A result of the amplitude and delay distortion caused by the nonideal channel frequency response characteristic is intersymbol interference (ISI). Digital pulses subject to intersymbol interference are elongated so that a pulse corresponding to any one bit will smear into adjacent bit slots. The effect of intersymbol interference on a stream of randomly generated polar encoded (± 1) digital pulses is illustrated in Figure 4. The discrete-time channel model is represented as a second order transversal filter taken from an example in (Proakis 1983). On this graph, the channel input and output are superimposed in order to show the extent of the distortion caused by ISI.

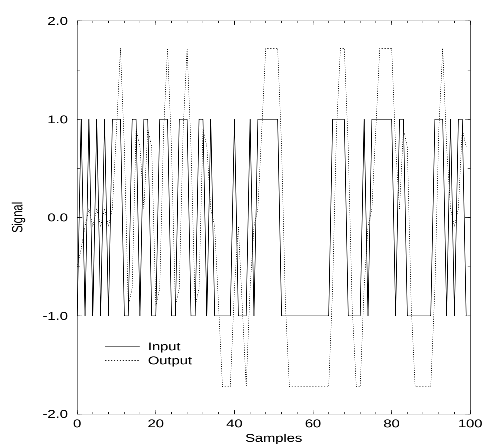

Fig. 4. Intersymbol interference

The effects of intersymbol interference can be reduced by equalization. An equalizing filter is a structure designed to compensate for the imperfect transmission characteristics of the channel. In practical communications systems the frequency response of the channel is usually not known with sufficient accuracy to enable a time-invariant equalizer to be constructed. Similarly, in communication systems operating over switched telephone lines the variation in channel characteristics from one line to another may be so great that the equalizer has to adjust its response to each individual channel. Consequently, an adaptive equalizer is often used, with a response which can be adjusted to meet specific measured channel characteristics. A widely implemented form of adaptive equalizer is the linear transversal equalizing filter. This non-recursive filter is realized as a tapped delay-line with tap weights adjusted by some recursive adaptation algorithm. The criterion used to optimize the equalizing filter coefficients is usually some function of the mean squared error or peak distortion (worst case intersymbol interference). The next Section details preliminary work in the genetic adaptation of recursive equalizing filters undertaken by the authors.

6 A Genetic Adaptive Algorithm for Data Equalization

The results of simulation experiments are presented here in order to illustrate the capabilities of a genetic adaptive algorithm used for data equalization. Two different channels are modelled, producing intersymbol interference at low and high degrees of severity. In each case a traditional bit-string genetic algorithm was used to optimize the coefficients of a population of time-varying recursive filters. The defining parameters of the GA were, for these initial experiments, those identified by Grefenstette (1986) as producing optimal performance with respect to the online performance measure, the on-line performance being defined as the average performance of all tested structures over the course of the search. Thus, the population size was kept small at 30 time-varying filters and the genetic operators of mutation and two-point crossover were applied at rather high frequencies, 0.01 and 0.95 respectively. A sequence of 100 pseudo-random, polar (± 1) encoded, values was used as the input to the channel throughout each run and the genetic algorithm was set the task of minimizing the mean squared error, generated by subtracting the output of the channel from the desired output value at each sample time. Use of this minimization criterion assumes that the adaptive filter has prior knowledge of the transmitted information sequence in order for it to form the error signal . Such information (known variously as a learning sequence or preamble) is typically made available during a short training period before the data is transmitted. The coefficients of time-varying filters may be continuously updated if a decision-directed mode of operation is implemented, in which decisions on the output of the equalizer are assumed to be correct and used in place of the desired output sequence. As long as the receiver is operating at low error rates adaptive algorithms are able to converge on the optimal equalizing filter coefficients.

The results of the first equalization problem are shown in Figure 5. The

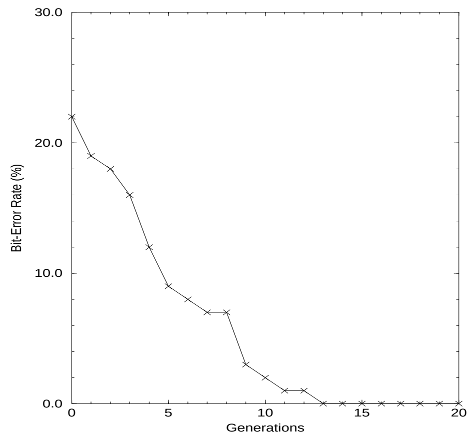

Fig. 5. Equalization of data channel

percentage of bit-errors (the mean of 10 runs) is plotted against the generation number. In this simulation, a channel model with a low-pass frequency and linear phase characteristic is implemented. The transfer function of this channel is:

$$H(z) = 0.407 + 0.815z^{-1} + 0.407z^{-2} \tag{1}$$

and the distorting effect that it has on an input stream of polar encoded binary digits was illustrated previously in Figure 4. The effect of the intersymbol interference is such that 22% of the bits arriving at the receiver are incorrect. After having evaluated only 13 generations the genetic adaptive algorithm is able to find a set of equalizing filter coefficients which reduce the bit-error rate to zero.

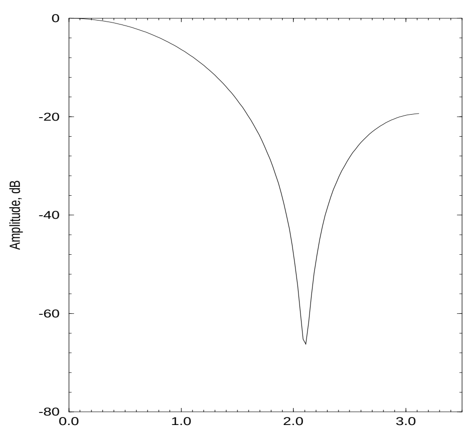

Fig. 6. Frequency response of channel with severe intersymbol interference

The next simulation aims to demonstrate the ability of the genetic adaptive algorithm to compensate for a severely distorting channel. The frequency response of the fourth order discrete-time channel model is shown in Figure 6, illustrating its poor spectral characteristics. The intersymbol interference generated by the channel as a result of its imperfect frequency and phase characteristics distort the input signal such that 40% of the data bits arriving at the receiver are erroneous.

Using a recursive equalizer to compensate for the severe intersymbol interference of the channel, the genetic adaptive algorithm requires, on average, 160 generations to reduce the bit-error rate to zero. This performance was compared to that of a non-recursive equalizer using the same number of filter coefficients (nine). The results of this comparison are given in Figure 7. On this graph, the bit-error rate (mean of ten runs) is plotted against the number of generations evaluated. Within the length of the simulation run the bit-error rate never drops below 15% when using the non-recursive equalizer. This simulation then, clearly illustrates the improved performance that can be obtained from a recursive equalizing filter over a non-recursive equalizer of similar complexity.

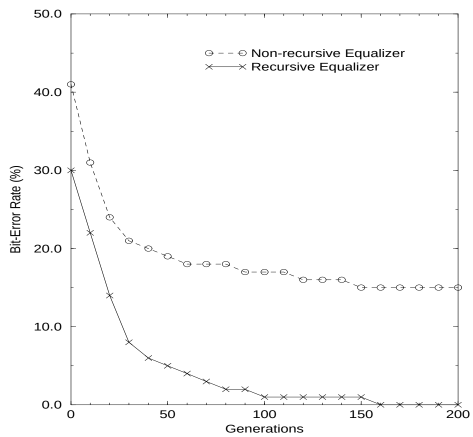

Fig. 7. Equalization of channel with severe intersymbol interference

7 Summary

The results of the simulation experiments presented in this paper demonstrate that a genetic adaptive algorithm can be used to optimize the coefficients of recursive time-varying digital filters. By choosing to implement the adaptive filters as lattice structures the entire feasible coefficient space can be searched without there being any risk of the coefficient set becoming unstable. Additionally, lattice filters are known to be less sensitive to the effects of coefficient round-off. Since one of the major problems of recursive output-error adaptive filters is their potentially multimodal error surfaces, the ability of the genetic algorithm to search spaces of this type is a significant advantage. Other optimization algorithms have also demonstrated this capability. However, when higher-order filters are adapted using the SLA approach of Nambiar et al. the number of actions of the automata becomes increasingly large thus slowing the speed of convergence. In contrast, genetic algorithms have been shown to be capable of tackling very high dimensional problem spaces (Mühlenbein and Schlierkamp-Voosen 1993).

In the system identification configuration, the genetic adaptive algorithm has demonstrated its ability to converge to the optimal filter coefficient values. Using a standard bit-string genetic algorithm, the precision to which these values can be determined is dependent on the number of bits used to encode each coefficient value. For very high orders of adaptive filter, this would require a correspondingly large string representation for each filter structure. Consequently, research is currently being undertaken by the authors into applying real-coded genetic algorithms to the task of filter adaptation. This type of genetic algorithm manipulates vectors of floating-point numbers rather than strings of binary digits and and has been shown to be faster and more consistent from run to run in certain problem domains (Janikow and Michalewicz 1991). The mutation and recombination operators of a real-coded genetic adaptive algorithm may be tailored for the particular task in hand and act at the level of the filter coefficients themselves rather than on a binary representation of them.

The comparison between recursive and non-recursive equalizers (Figure 7) highlights the need for algorithms which can reliably adapt recursive digital filters. Results from the data equalization simulations demonstrate that a GA-driven recursive equalizing filter is able to compensate for the imperfect frequency and phase characteristics of a digital communications channel. Future work in this area will investigate the use of genetic adaptive methods in equalizing time-varying channels. Previous simulations indicate that a conventional genetic algorithm using a strong selection policy and small mutation rate quickly eliminates population diversity as it seeks out the global optimum. Whilst this may be desirable in problems where the global optimum remains static, the performance of the standard genetic adaptive algorithm is adversely affected when the channel characteristics alter with time. In essence, this type of algorithm is often unable to track a moving optimum. Recent studies (Grefenstette 1992; Cobb and Grefenstette 1993), have explored the effectiveness of various mutation-based schemes in enhancing the performance of genetic algorithms operating in changing environments. In addition to these mechanisms, the authors are currently investigating the integration of local search (individual learning) heuristics into the genetic adaptive algorithm framework.

8 Acknowledgements

This work was supported by the UK Science and Engineering Research Council and the Defence Research Agency under the CASE scheme.

References

Cobb, H. G., Grefenstette, J. J.: Genetic Algorithms for Tracking Changing Environments. Proceedings of the Fifth International Conference on Genetic Algorithms.

(1993) 523–530
Etter, D. M., Hicks, M. J., Cho, K. H.: Recursive Adaptive Filter Design Using an Adaptive Genetic Algorithm. Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 82). **2** (1982) 635–638
Flockton, S. J., White, M. S.: Pole-Zero System Identification Using Genetic Algorithms. Proceedings of the Fifth International Conference on Genetic Algorithms. (1993) 531–535
Grefenstette, J. J.: Optimization of Control Parameters for Genetic Algorithms. IEEE Trans. Systems, Man, and Cybernetics. **16** (1986) 122–128
Grefenstette, J. J.: Genetic Algorithms for Changing Environments. Parallel Problem Solving from Nature 2. (1992) 137–144
Janikow, C. Z., Michalewicz, Z.: An Experimental Comparison of Binary and Floating Point Representations in Genetic Algorithms. Proceedings of the Fourth International Conference on Genetic Algorithms. (1991) 31–36
Johnson, C. R., Larimore, M. G.: Comments on and Additions to 'An Adaptive Recursive LMS Filter'. Proceedings IEEE. **65** (1977) 1399–1401
Kristinsson, K., Dumont, G. A.: System Identification and Control Using Genetic Algorithms. IEEE Trans. Systems, Man, and Cybernetics. **22** (1992) 1033–1046
Mühlenbein, H., Schlierkamp-Voosen, D.: Predictive Models for the Breeder Genetic Algorithm: I. Continuous Parameter Optimization. Evolutionary Computing. **1** (1993) 25–49
Nambiar, R., Tang, C. K. K., Mars, P.: Genetic and Learning Automata Algorithms for Adaptive IIR Filtering. Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 92). **5** (1992)
Oppenheim, A. V., Schafer, R. W.: Discrete-Time Signal Processing Prentice-Hall (1989)
Proakis, J. G.: Digital Communications. McGraw-Hill (1983)
Shynk, J. J.: Adaptive IIR Filtering. IEEE ASSP Magazine. April (1989) 4–20
Stremler, F. G.: Introduction to Communication Systems, Third edition. Addison-Wesley (1990)
White, M. S., Flockton, S. J.: A Genetic Adaptive Algorithm for Data Equalization. Proceedings of the IEEE World Congress on Computational Intelligence. (to appear)

This article was processed using the LaTeX macro package with LLNCS style