JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 15, No. 3, May-June 1992

Control System Optimization Using Genetic Algorithms

K. Krishnakumar*
University of Alabama, Tuscaloosa, Alabama 35487
and
David E. Goldberg†
University of Illinois at Urbana-Champaign, Urbana, Illinois 61801

The use of genetic algorithms as a technique for solving aerospace-related control system optimization problems is explored in this paper. Genetic algorithms are parameter search procedures based on the mechanics of natural genetics. They combine a Darwinian survival-of-the-fittest strategy with a random yet structured information exchange among a population of artificial chromosomes. The genetic algorithm technique is used to design a lateral autopilot and a windshear controller. The results show that a variety of aerospace control system optimization problems can be addressed using genetic algorithms with no special problem-dependent modifications. Suggestions for other uses related to aerospace control system optimization are presented.

Introduction

MANY techniques are used today for optimizing the design of control systems. Most of these techniques can be broadly classified under two main classes: 1) calculus-based techniques and 2) enumerative schemes. The calculus-based techniques, although extensively used, have the following drawbacks: they are local in scope, i.e., the extrema they seek are the ones closer to the current point, and they depend on the existence of either derivatives or some function evaluation scheme (see, for example, Refs. 1 and 2). Thus, calculus-based methods lack robustness over the broad spectrum of optimization functions. Many enumerative schemes have been proposed to overcome the shortcomings of calculus-based methods (see, for example, Ref. 3). These schemes lack efficiency because many practical search spaces are too large to search one at a time. Another type of algorithm that has gained popularity is the random search technique. This algorithm lacks efficiency and in the long run can be expected to do no better than enumerative schemes.[4]

One technique that is both global and robust over a broad spectrum of problems is the genetic algorithm (GA). Genetic algorithms are search procedures based on the mechanics of natural genetics. All natural species survive by adapting themselves to the environment. This natural adaptation is the underlying theme of GA. Genetic algorithm search combines a Darwinian survival-of-the-fittest strategy to eliminate unfit characteristics and uses random information exchange, with exploitation of knowledge contained in old solutions, to effect a search mechanism with surprising power and speed.

Bramlette and Cusin[5] have investigated the application of GAs in the design and manufacture of aeronautical systems. They used GAs in their work as a part of a system that contained several optimization techniques. Freeman et al.[6] have investigated using GAs in fine-tuning fuzzy logic controllers in aerospace applications.

Genetic algorithms were originally developed by Holland[7] and have been analyzed and extended further by De Jong,[8] Goldberg,[4] and others.[9-11] The main emphasis of this paper is the aerospace control system applications of GAs; however, to appreciate the power of GA, a brief introduction into the workings of a simple genetic algorithm is presented next. In the remainder of this paper, GA optimization techniques are applied to two feedback flight control system design problems.

Genetic Algorithms

Genetic algorithms are different from normal search methods encountered in engineering optimization in the following ways:

1) GAs work with a coding of the parameter set, not the parameters themselves.

2) GAs search from a population of points, not a single point.

3) GAs use probabilistic transition rules, not deterministic transition rules.

Genetic algorithms require the natural parameter set of the optimization problem to be coded as a finite-length string. As an example, a feedback control system with two feedback gains is shown in Fig. 1. For this optimization problem, the two-gain controller is discretized by mapping from a smallest possible gain set K_{min} to a largest possible gain set K_{max}. This mapping uses a 10-bit binary unsigned integer for both K_1 and K_2. In this coding a string code 0000000000 maps to K_{min}, and a 1111111111 maps to K_{max} with a linear mapping in between. Next, the two 10-bit gain sets are chained together to form a 20-bit string representing a particular controller design. A single 20-bit string represents one of the $2^{20} = 1{,}048{,}576$ alternative solutions. Genetic algorithms work iteration by iteration, generating and testing a population of strings. This population-by-population approach is similar to a natural population of biological organisms where each generation successively evolves into the next generation by being born and

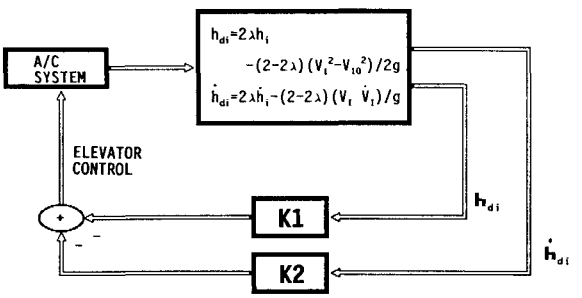

Fig. 1 Windshear feedback controller.

Presented as Paper 90-3488 at the AIAA Guidance, Navigation, and Control Conference, Portland, OR, Aug. 20-22, 1990; received March 11, 1991; revision received Aug. 27, 1991; accepted for publication Aug. 29, 1991.

*Assistant Professor, Aerospace Engineering, Box 870280. Member AIAA.

†Associate Professor, General Engineering, 117 Transportation Building, 104 S. Mathews Avenue.

raised until it is ready to reproduce. This approach is very different from classical search methods, where movement is from one point in the search space to another point based on some transition rule. Another important difference between GAs and the classical approaches is in the selection of the transition rules. In classical methods of optimization, the transition rule is deterministic. In contrast, GAs use probabilistic operators to guide their search.

A simple genetic algorithm is composed of three operators: 1) reproduction, 2) crossover, and 3) mutation. Reproduction is a process where an old string is carried through into a new population depending on the performance index (i.e., fitness) values. Due to this move, strings with better fitness values get larger numbers of copies in the next generation. Selecting good strings for the reproduction operation can be implemented in many different ways. In the method used in this study (there are many other equivalent techniques; Refs. 4 and 12 present a comparison study of selection schemes currently in use), strings with higher fitness values F get a proportionally higher probability of reproduction selection (roulette wheel selection[4]) based on

$$P_{(\text{select})} = F_i / \Sigma F_i$$

where i = string index. This strategy, in which good strings get more copies in the next generation, emphasizes the survival-of-the-fittest concept of genetic algorithms.

A simple crossover follows reproduction in three steps. First, the newly reproduced strings are paired together at random. Second, an integer position n along every pair of strings is selected uniformly at random. Finally, based on a probability of crossover, the paired strings undergo crossing over at the integer position n along the string. This results in new pairs of strings that are created by swapping all of the characters between characters 1 and n inclusively. As an example, consider two strings X and Y of length 5 mated at random from the mating pool of the new generation (numbers in brackets show a binary coded representation of a possible combination):

$$\begin{array}{lll} X = X1\ X2\ X3 & X4\ X5 & [0\ 0\ 0\ \ 1\ 1] \\ Y = Y1\ Y2\ Y3 & Y4\ Y5 & [1\ 1\ 1\ \ 0\ 0] \end{array}$$

If the random draw chooses position 3, the resulting crossover yields two new strings, X^* and Y^*, after the crossover:

$$X^* = Y1\ Y2\ Y3\ X4\ X5 \qquad [1\ 1\ 1\ 1\ 1]$$

$$Y^* = X1\ X2\ X3\ Y4\ Y5 \qquad [0\ 0\ 0\ 0\ 0]$$

Although the crossover operation is a randomized event, when combined with reproduction it becomes an effective means of exchanging information and combining portions of good quality solutions. Reproduction and crossover give GAs most of their search power.

The third operator, mutation, is simply an occasional random alteration of a string position (based on probability of mutation). In a binary code, this involves changing a 1 to a 0 and vice versa. The mutation operator helps in avoiding the possibility of mistaking a local minimum for a global minimum. When mutation is used sparingly (about one mutation per thousand bit transfers) with reproduction and crossover, it improves the global nature of the genetic algorithm search. Schaffer et al.[10] have shown with statistical support that the behavior of GAs is function independent. They also present empirical analysis of the three GA operators and their influence on GA performance.

Table 1 Genetic algorithm parameters

	Population size	Crossover rate	Mutation rate
Dejong's[8] parameters	50–100	0.6	0.001
Grefenstette's[9] parameters	30	0.95	0.01

A GA's performance is usually measured in terms of on-line performance and off-line performance. On-line performance is the average performance measure (average fitness value) of all of the tested strings over the course of the search. Off-line performance is the average of all of the generation-based best performance strings. Studies conducted by De Jong[8] and Grefenstette[9] (see Table 1) can be used as guidelines for choosing the GA parameters.

At first glance, it seems strange, or at least interesting, that such simple mechanisms should motivate anything useful, let alone the robust search technique claimed earlier. To understand the performance of GAs, it is useful to think of a GA as generating computational innovation. What is it that we are doing when we are being innovative or creative? Most often we are combining notions that worked well in some context with some notions that worked well in another context to form a new idea of how to undertake the task at hand. In the same way, genetic algorithms combine partial strings to form new solutions that are possibly better than their predecessor. This kind of methodology is strictly inductive when compared with other search methods, which are ploddingly deductive. But induction for its own sake is not a compelling argument to use any method, unless it can be shown how and when the method is likely to converge. Holland's schema theorem[4,7] places the theory of genetic algorithms on rigorous footing by calculating a bound on the growth of useful similarities or building blocks. The fundamental principle of GAs is to make good use of these similarity templates. There are, however, instances when the underlying coding mechanism may inhibit utilization of these similarity templates. Problems of this type are known as GA-deceptive problems.[13] In a recent study by Goldberg et al.,[13] a type of GA known as messy GA was introduced, and it was shown that GA-deceptive problems can be solved using messy GAs. Also, it was conjectured that global convergence using messy GAs is probabilistically polynomial. If shown correct, this conjecture could revolutionize the global solution of almost any problem.

It is relevant at this point to compare GAs with other so-called artificial intelligence (AI) techniques that are currently in use for the design and optimization of control systems. These AI techniques include, but not limited to, artificial neural networks (ANN), knowledge-based systems (KBS), fuzzy logic systems (FLS), and simulated annealing (SA). KBS and FLS applications in control rely on a priori knowledge of the problem to be solved and therefore cannot be classified under techniques that learn from past performance. On the other hand, ANNs learn through repeated exposure to desired input-output relationships.[14] The fundamental difference between ANNs and GAs in control system design is that GAs provide a technique for optimizing a given control system structure, whereas the ANN can only be an integral part of that structure. In other words, GAs can be used to design ANN but not vice versa.

The last technique just listed, namely, simulated annealing,[15] is a combinatorial optimization technique that has attracted attention recently for optimizing problems of large scale. Simulated anealing draws its strength from exploiting the statistical mechanics analogy to combinatorial optimization. In a recent review,[16] the performance of GA, SA, and ANN were compared for the traveling salesman problem and it was shown that the GA delivered some of the best performance in the context of this review. The connections between GA and SA have also been explored by Goldberg.[17] The distinct difference between GA and SA lies in the fact that the GA focuses on the importance of recombination and other operators

found in living systems. This gives GA the potential to expand once the operators found in nature are better understood. The overview of GA presented earlier is of a simple three-operator GA. Goldberg[4] presents discussions on specialized higher-order operators and selection procedures that are currently being investigated.

Genetic Algorithm Applications

The next few sections present the application of GAs for the optimization of the following problems: 1) a linear quadratic regulator (LQR) design of a lateral autopilot and 2) a wind-shear controller.

The aforementioned problems are chosen to illustrate the versatility of GAs in solving linear and nonlinear optimization problems related to aerospace applications.

Lateral Autopilot Design

One of the most common products of the modern control theory is the theory of the linear quadratic regulator. In this section an example problem presented by Bryson and Ho,[18] a lateral autopilot design, is optimized using the GA (this example was chosen to show the inner workings of a GA and not to suggest GA as an alternative to classical LQR design). Specifically, the GA designs lateral feedback gains based on a quadratic performance index to maintain heading and roll attitude. The perturbation equations of lateral motion are given by a fifth-order system:

$$\dot{X} = [A]\,X + [B]\,U \tag{1}$$

where

$X = [\beta\ r\ p\ \Phi\ \Psi]^T$
$U = [\delta_r\ \delta_a]^T$
β = sideslip angle
r = yaw rate
p = roll rate
δ_r = rudder deflection
δ_a = aileron deflection
Ψ = yaw angle
Φ = roll angle

The all-state feedback system and the quadratic performance index are defined respectively as

$$\begin{bmatrix}\delta_r\\ \delta_a\end{bmatrix} = -\begin{bmatrix}K11 & K12 & K13 & K14 & K15\\ K21 & K22 & K23 & K24 & K25\end{bmatrix}\begin{bmatrix}\beta\\ r\\ p\\ \Phi\\ \Psi\end{bmatrix} \tag{2}$$

$$\min_u J[u] = \lim_{tf-t0\to\infty} \tfrac{1}{2}\int_{t_0}^{tf} [\delta_a^2 + \delta_r^2 + (\beta+\Psi)^2 + \Phi^2]\,dt \tag{3}$$

To solve this problem, a simple GA was chosen with the following operators: 1) roulette wheel selection, 2) simple crossover, and 3) mutation. Grefenstette's parameters, presented in Table 1, were used for the GA implementation. Each gain in the gain set (K_{ij}) was represented by a 7-bit string. The strings were concatenated to produce a 70-bit string, each string representing one feedback design. As an example, a probable 70-bit string with spaces added for emphasis is shown below:

0100101 0110100 1010110 0101011 1011101 1100101
K11 K12 K13 K14 K15 K21

1011100 1100110 1101110 1011000
K22 K23 K24 K25

This string represents one possible solution out of 2^{70} solutions. In the previous representation, 0000000 represents a predetermined minimum value for the gains and 1111111 represents a predetermined maximum value for the gains. Note here that the mapping of the gain set on to the 7-bit string can be different for different gains depending on the desired range and resolution of the gain values.

The fitness function for the GA implementation is

$$\min_{K_{ij}} F(K_{ij}) = \sum_{m=1}^{4} \tfrac{1}{2}\int_0^{120\,s} [\delta_a^2 + \delta_r^2 + (\beta+\Psi)^2 + \Phi^2]\,dt \tag{4}$$

where m = 1–4 represents four random initial condition responses of the lateral system. Four initial conditions are chosen to insure excitation of all lateral nodes. For this problem, K_{min} and K_{max} were set at -1 and 1, respectively, and for the 7-bit representation the resolution was calculated to be $2/(2^7 - 1) = 0.0157$. When there are no guidelines available for setting the maximum and minimum values, floating point coding can be used. In this type of coding, the bits can be used to represent the mantissa and the exponent of a floating point number.[4]

The procedural flow of the optimization conducted using the GA is outlined next.

1) The initial generation of gain sets (population of 30) is chosen randomly.

2) The lateral airplane system simulation is conducted, and the fitness value (performance index) is evaluated for each gain set.

3) Each set of gains is reproduced by applying selection, crossover, and mutation operations.

4) Steps 1 and 2 are repeated 20 times (20 generations).

5) After 20 generations, the computation is terminated when, for a particular generation, the average fitness is within 1% of the best fitness value in the population. This indicates convergence in the population.

6) The gain set based on the best fitness value from the current generation population of gain sets is chosen as the optimal gain set.

Figure 2 presents the evolution of the average and best (minimum) fitness values over 65 generations for the lateral autopilot design problem.

To make a comparison between GA and an established parameter optimization algorithm, the same problem with the identical initial conditions was optimized using Powell's conjugate direction set method.[19] The results obtained are compared in Table 2. Table 2 also presents the results from the infinite-horizon formulation [Eq. (3)].[18] The results presented show that the GA solution matches with the LQR results better than does Powell's method. In addition, the GA method took fewer function evaluations to reach the optimal solution. It should be noted here that the results obtained by the LQR problem via the steady-state Riccati equation solution is the exact solution to the infinite regulator problem. The

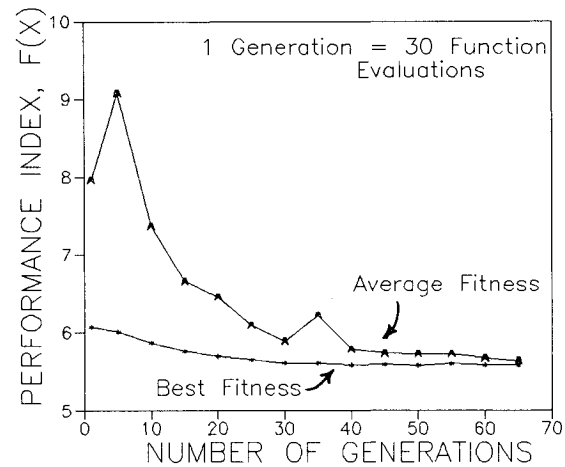

Fig. 2 Average and best fitness evolution for the LQR problem.

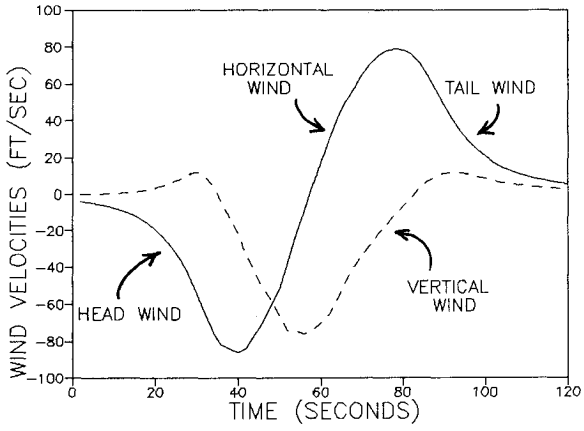

Fig. 3 Wind velocities of a severe microburst.

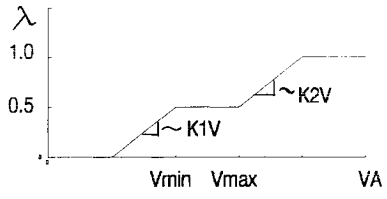

Fig. 4 Airspeed-based λ design.

parameter optimization solution using the GA and Powell's method are approximate due to the finite nature of the performance index [Eq. (4)]. The big differences noted in the gains reflect the poor sensitivity of the performance measure to some of these gains. This is illustrated in columns 5 and 6 of Table 2. It is noted from Table 2 that the fitness function [Eq. (4)] exhibits the least sensitivity to the variations in the gains that are substantially different from the infinite-horizon LQR gain set.

Wind Shear Controller Optimization

In this example, the principle of inertial energy distribution error control for optimal airplane microburst windshear penetration, presented by Krishnakumar and Bailey,[20] is used to design a feedback controller for a takeoff flight in a microburst. A genetic algorithm is utilized for the optimization of the design. A microburst is technically defined as a downburst of cool air that interacts with the Earth to cause, at low altitudes, a complex wind field that may have high windshear and dilation.

In the literature, three performance measures related to optimal flight through microbursts have been shown to be equivalent. These measures include minimax height loss,[21,22] minimax gamma loss,[23] and minimization of a quadratic performance measure based on inertial energy distribution error.[20] In this section, the minimax height loss performance measure is chosen for the design of the feedback system. This measure is chosen to show the versatility of a GA in optimization of complex search spaces. A longitudinal aircraft energy performance measure based on inertial energy distribution was found by Krishnakumar and Bailey[20] to yield a good control strategy for windshear encounters. Simply stated, this strategy suggests that the error in total airplane energy (loss of energy due to the windshear encounter) should be equally distributed between the airplane potential energy and the airplane kinetic energy to minimize deviations from the desired trajectory. This equidistribution of energy is realized by a feedback controller, shown in Fig. 1, to yield minimum height loss trajectories.

A body-fixed axis system is used to define the longitudinal nonlinear aircraft dynamic equations. The aircraft used for the simulation study is a Boeing B-727 aircraft powered by three JT8D-17 turbofan engines. A Zhu-Etkin three-dimensional doublet sheet microburst model,[24] which models both horizontal windshear and downdraft, is chosen to simulate the windshear disturbances. A severe windshear profile (Fig. 3) is chosen for the optimization problem. Based on the assumption that the power setting is at maximum level during takeoff, elevator becomes the only longitudinal control. For the optimization problem, nonlinear limits on angle of attack and the elevator deflection are imposed. The aircraft model used for this study is documented in Ref. 25.

The feedback elevator perturbation control is computed based on the inertial energy distribution error between inertial height and inertial speed as

$$\delta e = K_1 \cdot h_{di} + K_2 \cdot \dot{h}_{di} \tag{5}$$

where

$h_{di} = 2\lambda h_i - (2 - 2\lambda)(V_1^2 - V_{10}^2)/2g$
$\dot{h}_{di} = 2\lambda \dot{h}_i - (2 - 2\lambda)(V_1 \dot{V}_1)/g$
h_i = inertial height error
$V_1 - V_{10}$ = inertial velocity error
h_{di} = specific inertial energy distribution error

The choice of λ is based on the maximum allowable fluctuations in airspeed (V_{min} and V_{max}). Figure 4 presents the λ variation used in this study; K_{1V} and K_{2V} are optimized by augmenting the original minimax height loss performance measure with penalty functions. Penalty methods are sometimes criticized because of the steep ridges they impose on otherwise smooth problems. These ridges can cause difficulty for search techniques that depend on a particular shape of local search surface. This objection is not pertinent to GA performance because GAs do not depend on continuity or derivative existence for their operation. The penalty method is simply used both to extract information from the infeasible solutions and to eliminate infeasible regions in the search space.

For the GA implementation, each of the four gains (K_1, K_2, K_{1V}, and K_{2V}) is discretized by mapping from a smallest possible gain K_{min} to a largest possible gain K_{max}, using a 7-bit binary unsigned integer. In this coding, a string code 0000000 maps to K_{min} (0.0 in this study) and a 1111111 maps to K_{max} (0.01 in this study) with a linear mapping in between. Next, the four 7-bit strings are chained together to form a 28-bit string, representing a particular controller design. A single 28-bit string represents one of the 2^{28} alternative solutions. The optimization procedure is conducted in a manner similar to the previous example.

To appreciate the complexity of the search space and the usefulness of GAs, the cost surface for K_1 and K_2 is presented

Table 2 Optimal gain set comparison

Gains and fitness	Infinite-horizon LQR gain set[18]	GA gain set	Powell's gain set	Change in gains, %	Change in fitness, %
$K11$	0.32	0.669	−1.13	170	5.2
$K12$	1.01	0.96	21.5	950	5.0
$K13$	0.069	0.488	−3.27	1200	5.11
$K14$	0.076	0.318	−0.514	510	5.04
$K15$	0.55	0.661	1.355	115	5.0
$K21$	0.177	0.181	−2.0	570	5.0
$K22$	0.388	0.480	11.3	970	5.0
$K23$	0.74	0.795	−1.361	180	5.2
$K24$	1.03	0.897	0.17	50	5.3
$K25$	0.83	0.708	1.298	55	5.4
Fitness	5.562	5.581	5.696		
No. of function evaluations	Not applicable	1950	14,500		

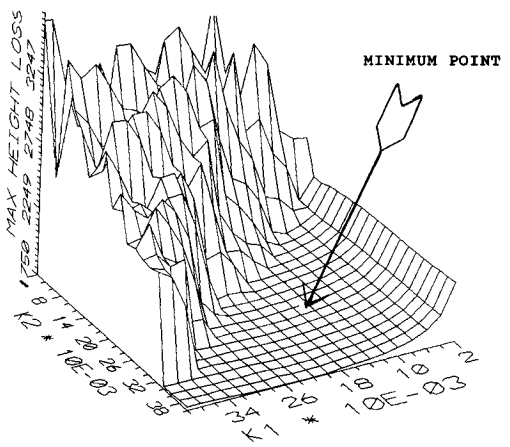

Fig. 5 Cost surface of the minimax height loss problem.

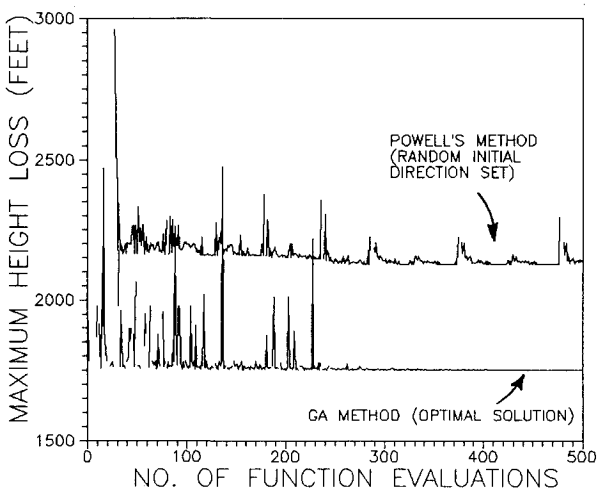

Fig. 6 Search evolution for the genetic algorithm and Powell's methods for the minimax height loss problem.

in Fig. 5 (with no penalty functions and with K_{1V} and K_{2V} fixed at 0.0025 and 0.0075, respectively). It is apparent from the figure that the cost surface has many local minima, and to solve this problem with any calculus-based search technique, the starting point will be critical. Since the GA searches from a population of points, for the GA this cost surface is as easy as any other well-behaved cost surface. This fact is demonstrated by using a Powell's conjugate directions method to optimize the two-gain problem. Figure 6 documents the search evolution of the GA and Powell's methods. As seen in this figure, the initial direction set chosen for Powell's method is critical for finding the global optimum. The noisy evolution of the GA search indicates the randomness that is part of the GA search. The noisiness in Powell's search reflects the nature of the search space (Fig. 5).

To utilize a gradient-based method, the minimax optimization problem (Chebyshev problem) has to be reformulated as a Bolza problem. Such a transformation is not necessary while using a GA due to the fact that the GA implementation does not depend on the nature of the performance measure.

Conclusions

In this paper, genetic algorithm optimization techniques solved two aerospace control system optimization problems. A standard linear quadratic regulator problem was optimized using a genetic algorithm and was compared with the traditional approach. A nonlinear windshear feedback controller optimization problem was also addressed using a simple genetic algorithm. The results show the wide applicability of genetic algorithm optimization techniques.

Currently, many types of aerospace-related optimization problems can be solved using the simple genetic algorithm. Many complex and nonlinear control systems are fine-tuned using heuristic approaches. These practices can be augmented with genetic algorithms, which are relatively insensitive to the complexities of the optimization problem and provide a robust search method. Other suggested areas where genetic algorithms can be successfully used include 1) adaptive gain tuning problems, 2) fuzzy logic controller optimization, and 3) multiple-objective optimization. Although more work is necessary, genetic algorithms should provide a useful weapon in the arsenal of control system optimization.

Acknowledgment

The second author's contribution was based on work supported by the National Science Foundation under Grant CTS-8451610.

References

[1]Wilde, D. J., and Beightler, C. S., *Foundations of Optimization,* Prentice-Hall, Englewood Cliffs, NJ, 1969.

[2]Jacoby, S. L. S., Kowelik, J. S., and Pizzo, J. T., *Iterative Methods for Non-Linear Optimization Problems,* Prentice-Hall, Englewood Cliffs, NJ, 1972.

[3]Hill, J. D., "A Search Technique for Multimodal Surfaces," *IEEE Trans. Sys. Sci. Cybern.*, Vol. SSC-3. No. 1, 1969, pp. 2-8.

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

[5]Bramlette, M. F., and Cusin, R., "A Comparative Evaluation of Search Methods Applied to Parametric Design," *Genetic Algorithms and Their Applications: Proceedings of the Third International Conference on Genetic Algorithms,* Morgan Kaymann, San Mateo, 1989, pp. 213-218.

[6]Freeman, L. M., Krishnakumar, K., Karr, C. L., and Meredith, D. L., "Tuning Fuzzy Logic Controllers Using Genetic Algorithms—Aerospace Applications," Aerospace Applications of Artificial Intelligence Conference, Dayton, OH, Oct. 1990.

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

[8]De Jong, K. A., "An Analysis of the Behavior of Genetic Adaptive Systems," *Dissertation Abstracts International,* University Microfilms, Michigan, 41(9), 3503B, 1975.

[9]Grefenstette, J. J., "Optimization of Control Parameters for Genetic Algorithms," *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. 16, No. 1, 1986, pp. 122-128.

[10]Schaffer, J. D., Caruana, R. A., Eshelman, L. J., and Das, R., "A Study of Control Parameters Affecting On-Line Performance of Genetic Algorithms for Function Optimization," *Genetic Algorithms and Their Applications: Proceedings of the Third International Conference on Genetic Algorithms,* Morgan Kaymann, San Mateo, 1989, pp. 51-60.

[11]Schaffer, J. D., "Multiple Objective Optimization with Vector Evaluated Genetic Algorithm," *Genetic Algorithms and Their Applications: Proceedings of the First International Conference on Genetic Algorithms,* Morgan Kaymann, San Mateo, 1985, pp. 93-100.

[12]Goldberg, D. E., and Deb, K., "A Comparative Analysis of Selection Schemes Used in Genetic Algorithms," The Clearinghouse for Genetic Algorithms, TCGA Rept. 90007, Univ. of Alabama, Tuscaloosa, AL, July 1990.

[13]Goldberg, D. E., Deb K., and Korb, B., "Messy Genetic Algorithms: Motivation, Analysis, and First Results," *Complex Systems,* Vol. 3, 1989, pp. 493-530.

[14]Miller, W. T., Sutton, R. S., and Werbos, P. J., *Neural Networks for Control,* MIT Press, Cambridge, MA, 1990, pp. 1-58.

[15]Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by Simulated Annealing," *Science,* 220(4598), 1983, pp. 671-680.

[16]Petersen, C., "Parallel Distributed Approaches to Combinatorial Optimization," *Neural Computation,* Vol 2, No. 3, 1990, pp. 261-269.

[17]Goldberg, D. E., "A Note on Boltzmann Tournament Selection for Genetic Algorithms and Population-Oriented Simulated Annealing," The Clearinghouse for Genetic Algorithms, TCGA Rept. 90003, Univ. of Alabama, Tuscaloosa, AL, May 1990.

[18]Bryson, A. E., and Ho, Y., *Applied Optimal Control,* Hemisphere, New York, 1975, pp. 172-175.

[19]Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling,

W. T., *Numerical Recipes,* Cambridge Univ. Press, New York, 1989, pp. 294–301.

[20]Krishnakumar, K., and Bailey, J. E. "Inertial Energy Distribution Control for Optimal Wind Shear Penetration," *Journal of Guidance, Control, and Dynamics,* Vol. 13, No. 6, 1990, pp. 944–951.

[21]Miele, A., Wang, T., and Melvin, W. W., "Optimal Flight Trajectories in the Presence of Wind Shear; Part 1, Take-off," AIAA Paper 85-1843, 1985.

[22]Miele, A., Wang, T., and Melvin, W. W., "Optimization and Acceleration Guidance of Flight Trajectories in a Wind Shear," *Journal of Guidance and Control,* Vol. 10, No. 4, 1986, pp. 368–377.

[23]Miele, A., Wang, T., Melvin, W. W., and Bowles, R. L., "Gamma Guidance Schemes for Flight in a Wind Shear," *Journal of Guidance and Control,* Vol. 11, No. 4, 1987, pp. 320–327.

[24]Zhu, S., and Etkin, B., "Fluid-Dynamic Model of a Downburst," Univ. of Toronto, UTIAS Rept. 271, CNISSM 0082-5255, Toronto Ontario, Canada, 1987, pp. 368–377.

[25]Krishnakumar, K., "Energy Concepts Applied to Control of Airplane Flight in Wind Shear," Ph.D. Thesis, Dept. of Aerospace Engineering, Univ. of Alabama, Tuscaloosa, AL, June 1988.