Pergamon

Engng Applic. Artif. Intell. Vol. 8, No. 2, pp. 177–189, 1995
Elsevier Science Ltd. Printed in Great Britain

0952-1976(94)00064-6

Contributed Paper

Least Median Squares Curve Fitting Using a Genetic Algorithm

C. L. KARR
Tuscaloosa Research Center, U.S.A.

B. WECK
Tuscaloosa Research Center, U.S.A.

D. L. MASSART
Vrije Universiteit Brussel, Belgium

P. VANKEERBERGHEN
Vrije Universiteit Brussel, Belgium

(Received April 1994; in revised form November 1994)

***Least median squares** (LMS) curve fitting is a method of robust statistics that guards the process of data analysis from perturbations due to the presence of outliers[1]. This procedure has several advantages over classic least squares (LS) curve fitting, especially in the noisy problem environments addressed by today's process-control engineers. Although LMS curve fitting is a powerful technique, there are some limitations to the LMS approach. However, these limitations can be overcome by combining the search capabilities of a genetic algorithm with the curve-fitting capabilities of the LMS method. **Genetic algorithms** are search techniques that model the search that occurs in nature via genetics. This paper presents a procedure for utilizing genetic algorithms in an LMS approach to curve fitting. Several examples are provided from a number of application areas, thereby demonstrating the versatility of the genetic-algorithm-based LMS approach.*

Keywords: Genetic algorithm, curve fitting, least median squares, least squares.

1. INTRODUCTION

A major objective of statistical data analysis is to aid in the extraction and explanation of information contained in a particular data set. Although statistical techniques are used for a wide array of objectives, many of these objectives are often directly related to curve fitting. Curve fitting receives considerable attention because it plays a key role in a number of engineering endeavors. As engineers strive to take full advantage of the advances in computers for computationally cumbersome, real-time systems such as numerical modelling,[2] equipment design,[3] and quality control,[4] regression techniques are pushed to their limits. Recent advances in artificial-intelligence-based process-control strategies[5] have been particularly demanding on traditional statistical techniques because these process-control strategies are extremely sensitive to the accuracy of the information extracted by the choice of a curve-fitting technique.

Classical statistical procedures such as least squares (LS) curve fitting have been used for both the extraction and the explanation of data across a broad spectrum of application domains. However, classical statistical techniques can be quite sensitive to *outliers* (points that are not consistent with the rest of the data set). A number of robust statistical techniques have been developed that attempt to overcome sensitivity to outliers.[6] The basic virtue of these robust statistical methods is that small deviations from the model assumptions caused by the presence of outliers hinders their performance only slightly, whereas larger deviations do not cause the methods to fall apart altogether. Generally, these robust methods rely on iterative techniques, wherein outliers are identified and delegated to a dimi-

Correspondence should be sent to: Dr C. L. Karr, U.S. Bureau of Mines, Tuscaloosa Research Center, P.O. Box L, University of Alabama Campus, Tuscaloosa, AL 35486-9777, U.S.A.

nished role. Unfortunately, the problem of outlier detection can be computationally demanding.

Recently, a number of researchers have attempted to produce robust regression techniques that overcome the problem of outlier detection. Several effective techniques have been developed by altering the basic LS approach to curve fitting. Of course, LS curve fitting consists of minimizing the sum of the squared residuals; a residual is the difference between the actual data value and the value predicted by the model resulting from the curve-fitting technique. The LMS approach[1] replaces the sum of the squared residuals with the median of the squared residuals, thereby yielding an approach that is particularly effective at managing outliers. For instance, in LS curve fitting one outlier can dramatically affect the accuracy of the result. However, in LMS curve fitting up to 50% of the data points can be outliers without degrading the accuracy of the curve fit because only the median residual value is considered. Since the LMS method can withstand the presence of up to 50% of its points being outliers, the method is said to have a *breakdown point* of 50%. Despite the effectiveness of the LMS method, it can face three situations in particular that give it trouble:

(1) when there are a large number of data points or when there is a large number of coefficients the method tends to be slow;
(2) when the issue of speed is placed at a premium, the accuracy of the method suffers due to some approximations that are made; and
(3) nonlinear curve-fitting problems are difficult for the LMS method.[7]

The problems associated with large data sets and nonlinear curve fitting are not unique to the LMS method. In fact, these issues are apparent, to a lesser extent, in LS curve fitting. Researchers have acknowledged and addressed these issues in a number of different ways. A promising approach to tackling the problems associated with performing regression on large data sets and with using nonlinear models is to combine the search capabilities of a genetic algorithm with the curve-fitting capabilities of LS. In fact, this approach has been shown to be effective in a number of problem domains.[8] Genetic algorithms are search techniques based on the mechanics of natural genetics. They combine a Darwinian survival-of-the-fittest approach with an effective information exchange system.[9] They are able to locate near-optimum solutions to large and difficult search problems quite rapidly.

Because of their rapid convergence characteristics, genetic algorithms can be used to search for the constants needed in a least median squares (LMS) curve-fitting problem. The genetic algorithm exhibits properties that are desirable in the complex search associated with selecting LMS parameters. The results of the LMS curve fit can often be achieved more rapidly using a genetic algorithm, and the solution is more accurate in certain problems.

In this paper, a genetic algorithm is used to improve LMS curve fitting. First, the basics of both LS and LMS curve fitting are outlined, with special emphasis placed on the effect outliers have on the two methods. Second, the basic mechanics of a simple genetic algorithm are introduced. Third, an example is provided in which the details of using a genetic algorithm for LMS curve fitting are provided. Fourth, a number of examples are provided in which genetic algorithm LMS curve fitting is used to solve complex, highly nonlinear system characterization problems. Results are compared to traditional LMS curve fitting when possible.

2. THE LS AND LMS METHODS

The purpose of *regression analysis* is to fit equations of an assumed form to observed variables. The classical linear model assumes an equation of the form

$$y_i = x_{i1}\Theta_i + \cdots + x_{ip}\Theta_p + e_i \tag{1}$$

for $i = 1, \ldots, N$ where N is the number of data points. The variables $x_{i1}, \ldots, x_{ip}$ are the independent variables, and y_i is the dependent variable. In classical statistics, the error term, e_i, is assumed to be normally distributed with mean zero and unknown standard deviation, σ. One then tries to estimate the vector of unknown parameters Θ from the N data points (x_i, y_i). Applying a regression estimator (which can be computed analytically for a number of model forms) to such a data set yields a set of estimated regression coefficients Θ'. Although the actual regression coefficients, Θ, are unknown, the independent variables can be multiplied by the estimated regression coefficients to obtain

$$y_i' = x_{i1}\Theta_1' + \cdots + x_{ip}\Theta_p' \tag{2}$$

where y_i' is called the "predicted" or "estimated" value of y_i. The residual r_i of the ith case is the difference between what is actually observed and what is estimated:

$$r_i = y_i - y_i'. \tag{3}$$

LS curve fitting is the most popular regression approach and dates back to Gauss and Legendre (see Ref. 10 for some historical discussion). In the LS method, the sum of the squares of the residuals is minimized. Mathematically, the objective is to minimize

$$\sum_{i=1}^{N} r_i^2. \tag{4}$$

The basic idea is to optimize the fit by making the sum of the residuals as small as possible. The extent of the method's popularity is easily understood when one realizes that at the time of the method's conception

there were no computers, and the fact that the estimated regression coefficients could be calculated explicitly via some matrix algebra, made LS the most practical approach to curve fitting. Even today, most statistical packages utilize the LS method because of tradition and speed of computation. Actually, there are a number of situations for which LS curve fitting is more than adequate. However, there are situations for which the method is not suitable.

One of the more glaring weaknesses of the LS method becomes apparent when the data set that is being fitted contains outliers. These data points, which for one reason or another do not fit the assumed form of the model, can cause the LS method to give results that are not satisfactory. A single outlier can dramatically affect the results of the LS method, especially when the outlier is severe. Generally, the number of outliers it takes to contaminate a method is defined by a breakdown point. The *breakdown point* is the smallest fraction of contamination that can cause the estimator to take on values arbitrarily far from the assumed model. For the LS method, the breakdown point tends to zero, which means that only one outlier is needed to cause the method to give inappropriate results. Under tightly controlled conditions (or when some prescreening is performed), the number of outliers that are present in a data set can be minimized if the data points have at most three dimensions.[7] However, in a number of the real-time engineering applications that are possible with today's high-level computers, data sets cannot be adequately purged of outliers in the necessary time. Fortunately, there are a number of regression techniques that are capable of handling the occurrence of outliers more efficiently than the LS method.

Perhaps the first step toward a regression method that tends toward a high breakdown point is the *least absolute values regression estimator* in which the objective is to minimize the sum of the absolute values of the residuals.[11] The idea is that the effect of the residuals from any outliers will not be as damaging because their contribution is not squared. Next, Huber[12] developed the *M-estimators approach*, wherein the squared residuals of the LS method are replaced by a function of the residuals. This approach yields a system of equations that is not always easy to solve. Various other methods have been proposed[13-16] that all exhibit various breakdown points. Siegel[17] was perhaps the first to present a method with a breakdown point of 50% (which is the highest breakdown point that can reasonably be achieved). The method that is considered in this paper, namely, the LMS method,[1] was developed independently of Siegel's method.

The *LMS method of regression* consists of minimizing the median squared residual. An estimator is used to compute each of the N residuals, and the median residual is found. This is the term that is minimized. The LMS method has a breakdown point of 50% since the points are sorted and the poorest half of the points do not influence the estimator. As it turns out, this estimator is very robust with respect to outliers. However, the method is not without its problems. For instance, when the data set is large, the LMS method can take a long time to converge. This occurs because the algorithm is based on examining the possible combinations of pairs of the data points, and for each combination an estimator is computed which includes performing a computer sort. There are situations where in it is infeasible to examine all of the possible combinations. In such instances the number of possibilities considered is reduced in some meaningful way. However, the savings gained in computational time can be offset by a failure to locate the best possible solution. Additionally, this approach to LMS regression is limited by the fact that the model must pass through two of the data points exactly in high-order equations. There are, however, situations in which the best fit does not pass directly through any of the data points. It is these issues of speed of convergence versus accuracy of the method that the genetic algorithm is used to address.

The remainder of this paper focuses on using a genetic algorithm to locate the estimator for a LMS regression; here, "estimator" refers to the constants in the chosen model equation. In the next section, the basic mechanics of a genetic algorithm are introduced. Then, the details of just how a genetic algorithm can be used to solve an LMS curve-fitting problem are presented. Finally, several examples of genetic algorithm LMS curve fitting are provided. These examples go beyond the use of LMS for simple curve fitting, into the domain of system characterization. These examples demonstrate the effectiveness of using genetic algorithm LMS for a number of engineering problems.

3. THE MECHANICS OF A SIMPLE GENETIC ALGORITHM

Genetic algorithms are broadly applicable, efficient search algorithms based on the mechanics of natural genetics. They imitate nature with their Darwinian survival-of-the-fittest approach. This approach allows genetic algorithms to speculate on new points in the search space with expected improved performance by exploiting historical information. In nature, performance is gaged by an organism's ability to survive; in a genetic algorithm, performance is measured by a user-defined "fitness function" which is problem specific. Genetic algorithms imitate nature, and they exhibit some fundamental differences from more-conventional search techniques. Genetic algorithms differ from more-conventional search techniques in four ways:

(1) Genetic algorithms do not work directly with the parameter set; rather, they manipulate strings of characters representing the parameters themselves;

(2) Genetic algorithms consider many points in the search space simultaneously;
(3) Genetic algorithms use random choice to guide their search;
(4) They require no derivative information.

Genetic algorithms require the natural parameter set of the problem to be coded as a finite string of characters. Although many character sets have been used in real-world examples,[18] the parameter sets in this study are coded as strings of zeros and ones. For example, the two constants needed to define a line of the form $y = \Theta_1 x + \Theta_2$, where Θ_1 is the slope and Θ_2 is the y-intercept, are quite easily represented as a binary string. Eleven bits are allotted for defining each constant (although fewer or more bits can be used). The first bit position is devoted to the sign of Θ_1, i.e. when the value is zero, Θ_1 is positive, and when the value is one, Θ_1 is negative. The next 10 bits, positions 2 through 11, are interpreted as a binary number (1001010111 is the binary number 599). This value is mapped linearly between some user-determined minimum ($\Theta_{\min}$) and maximum ($\Theta_{\max}$) values according to the following:

$$\Theta_1 = \Theta_{\min} + \frac{b}{2^M - 1}(\Theta_{\max} - \Theta_{\min}) \tag{5}$$

where b is the integer value represented by an M-bit string. The values of $\Theta_{\min}$ and $\Theta_{\max}$ in a given problem are selected by the user, on the basis of personal knowledge of the problem. If necessary, a rapidly converging, course optimization method may be used for selecting the limiting values. This same form is used to represent Θ_2, and the two 11-bit strings are concatenated to form a single 22-bit string representing the entire parameter set (Θ_1 and Θ_2).

In other problems, the creation of effective finite string codings may require more-complicated mappings, such as those found in Ref. 18. But numerous codings have been developed and demonstrated to be effective. Since genetic algorithms work directly with a coding of the parameter set and not the parameters themselves, they are difficult to fool with local optima; and they do not depend upon continuity of the parameter space.

Genetic algorithms consider many points in the search space simultaneously. and therefore have a reduced chance of converging to local minimums or maximums. In most conventional search techniques a decision rule governs the movement from one point to the next. These methods can be dangerous in multimodal (many-peaked) search spaces because they can converge to local minimums or maximums. However, genetic algorithms generate entire populations of points (coded strings), test each point individually, and combine qualities from existing points to form a new population containing improved points. Aside from producing a more global search, the genetic algorithm's simultaneous consideration of many points makes it highly adaptable to parallel machines, since the evaluation of each point is an independent process.

A genetic algorithm requires only information concerning the quality of the solution produced by each parameter set (objective or fitness function values defined by the user for specific problems). This is contrary to many optimization methods which require derivative information or, worse yet, complete knowledge of the problem structure and parameters. Since genetic algorithms do not require such problem-specific information, they are more flexible than most search methods.

Last, genetic algorithms differ from more-typical search techniques in that they use random choice to guide their search. Although chance is used to define their decision rules, genetic algorithms are not random walks through the search space. They use chance efficiently in their exploitation of prior knowledge to rapidly locate near-optimum solutions.

Although some genetic algorithms have become quite complex, good results can be achieved with relatively simple genetic algorithms. The simple genetic algorithm used in this study consists solely of reproduction, crossover and mutation—operators basic to all genetic algorithms. As will be seen, these operators can be easily implemented by anyone with basic computer skills.

Before examining the individual operators, consider the overall processing of a genetic algorithm. An initial population of k strings, each of length, M, are generated at random. (Keep in mind that each string represents one possible solution to the problem at hand; one possible combination of the input parameters.) Each string is decoded, yielding the actual parameters. The parameter set represented by each individual string is sent to a numerical model of the problem, a solution based on the input parameters is returned, and the string is assigned a fitness value which is simply a nonnegative measure of the quality of the string's solution. The assignment of fitness values is problem-dependent; they are the user's subjective interpretation of the quality of a string as expressed by a fitness function. This fitness is then used to direct the application of the three operators which produce a new population of strings (a new generation). On average, this new generation will contain better solutions to the problem. The new population of strings is again individually decoded, evaluated, and transformed into a subsequent generation using the basic operators. This relatively simple process continues until convergence within a population is achieved.

Reproduction is simply a process by which strings with large fitness values, good solutions to the problem at hand, receive correspondingly large numbers of copies in the new population. In this study use is made of expected number control selection. This form of

reproduction makes a certain number of copies, num_i, of a string in accordance with the equation:

$$num_i = \frac{f_i}{f_{avg}} \tag{6}$$

where num_i is the number of copies of the string in the next generation, f_i is the fitness of the individual string in the current generation, and f_{avg} is the average fitness of the current generation. (A mechanism for handling round-off and assuring a constant population size is required.) Thus, reproduction is the survival-of-the-fittest aspect of the genetic algorithm. The best strings receive more copies in subsequent generations so that their desirable traits may be passed on to their offspring.

Crossover affords a means for strings to mix and match their desirable qualities through a random process. After reproduction, simple crossover proceeds in three steps. First, two newly reproduced strings are selected from the strings created by previous selection. These strings are then "crossed" with a finite probability, $P_{crossover}$. Second, a position along the string is selected uniformly at random. This is illustrated below where two binary coded strings A and B of length 11 are shown nested for crossover:

$$\begin{array}{ll} A = 111 & 11111111 \\ B = 000 & 00000000. \end{array}$$

Notice how crossing site 3 has been selected in this particular example through random choice, although any of the other nine positions could have been selected just as easily. The third step is to exchange all characters following the crossing site. The two new strings following this example cross are shown below as A′ and B′:

$$A' = 11100000000$$
$$B' = 00011111111.$$

String A′ is made up of the first part of string A and the tail of string B. Likewise, string B′ is made up of the first part of string B and the tail of string A. Although crossover has a random element, it should not be thought of as a random walk through the search space. It is an effective means of exchanging information and combining portions of high-quality solutions.

The mechanics of the reproduction and crossover operators are quite simple; they involve nothing more than making copies of strings and exchanging portions of strings. However, reproduction and crossover together give genetic algorithms much of their searching power.

Although reproduction and crossover give genetic algorithms most of their searching power, the third operator, mutation, enhances the ability of the genetic algorithm to find near-optimum solutions. Mutation is the occasional alteration of a value at a particular string position. Its purpose is to serve as an insurance policy; it insures against the loss of a particular piece of information. A generation may be created that is void of a particular character at a given string position. For example, a generation may exist that does not have a one in the third string position, when a one in the third position may be critical to obtaining a quality solution. Neither reproduction nor crossover will ever produce a one in this third position in subsequent generations. Mutation, however, allows for the possibility of a zero in the third position to be changed to a one. Thus, the critical piece of information can be reinstated into the population. Although mutation is a vital part of any genetic algorithm, it should be noted that it occurs with a small probability (on the order of one mutation per thousand string positions).

This has been a brief overview of a simple genetic algorithm. For details of the processing power and the convergence properties of genetic algorithms reference should be made to Ref. 19.

4. APPLICATION OF GENETIC ALGORITHM TO LS AND LMS CURVE FITTING

In this section, a simple genetic algorithm is used to solve two specific LS and LMS curve-fitting problems. Although the problems investigated may well have been solved by other methods, they serve as a demonstration of the genetic algorithm's versatility and power in the area of curve-fitting data.

Two specific examples are presented, in which results produced by the genetic algorithm in LS and LMS curve fitting are compared to the optimum results of the conventional calculus-based LS technique. The first example involves the equation $y = \Theta_1 x + \Theta_2$ which is simply the equation of a line. This is a simple example, presented to illustrate the basic premise of using a genetic algorithm for curve fitting. The second example is a slightly more complex problem. In this second example, data is fitted to the equation of a parabola, $y = \Theta_1 x^2 + \Theta_2 x + \Theta_3$. This is a more difficult curve-fitting problem for a genetic algorithm because the parameter set (Θ_1, Θ_2 and Θ_3) has increased by 50%. These two problems are meant simply to demonstrate the details of using a genetic algorithm for curve fitting. The next section will provide real-world examples, in which the approach of using a genetic algorithm for curve fitting represents a very effective solution method.

4.1. Preliminary considerations

Before considering the two specific problems, two points concerning the application of a genetic algorithm must be addressed. There are basically two decisions to be made when applying a genetic algorithm to any given problem:

(1) how to code the parameters of the problem as a finite string; and

(2) how to evaluate the merit of each string (each parameter set).

An effective method for coding the strings has been discussed previously, so consider the question of how to evaluate each string. In LS curve-fitting problems, the objective is to minimize the sum of the squares of the distances between a curve of a given form and the data points. Thus, if y is the ordinate of one of the data points, and y' is the ordinate of the corresponding point on the theoretical curve, the objective of LS curve fitting is to minimize the quantity $(y-y')^2$. This square of the error which is to be minimized affords a good measure of the quality of the solution. However, the genetic algorithm seeks to maximize the fitness when expected number control reproduction is used. Thus, the minimization problem must be transformed into a maximization problem. To accomplish this transformation, the error is simply subtracted from a large positive constant so that

$$f = C - \sum_{i=1}^{N} (y_i - y_i')^2 \tag{7}$$

yields a maximization problem. It should be reiterated here that there are faster ways of solving this particular curve-fitting problem. This example is presented strictly to illustrate the approach of using a genetic algorithm.

In conventional curve-fitting techniques, solving an LS and an LMS problem involve entirely different mathematics. This is definitely not the case when a genetic algorithm is used for curve fitting. To solve an LMS curve-fitting problem using a genetic algorithm, only the fitness function must be altered. In LMS curve fitting, no longer is the desire to minimize the sum of the squares of the residuals, but rather to minimize the median residual term. Thus, the appropriate fitness function is

$$f = C - med(y-y')^2. \tag{8}$$

The ease with which a user can switch from solving an LS curve-fitting problem to solving an LMS curve-fitting problem makes the genetic algorithm an especially inviting tool in the curve-fitting domain. The only change required appears in the definition of the fitness function; the basic algorithm remains unchanged.

The selection of an appropriate coding and the determination of a fitness function representation are the only aspects of a genetic algorithm that are in general problem-specific. Once these two decisions have been made, every problem is the same to a genetic algorithm. The genetic algorithm generates populations of strings, evaluates them, and uses these evaluations to produce subsequent generations of more highly fit strings. Thus, the genetic algorithm is a very flexible search method.

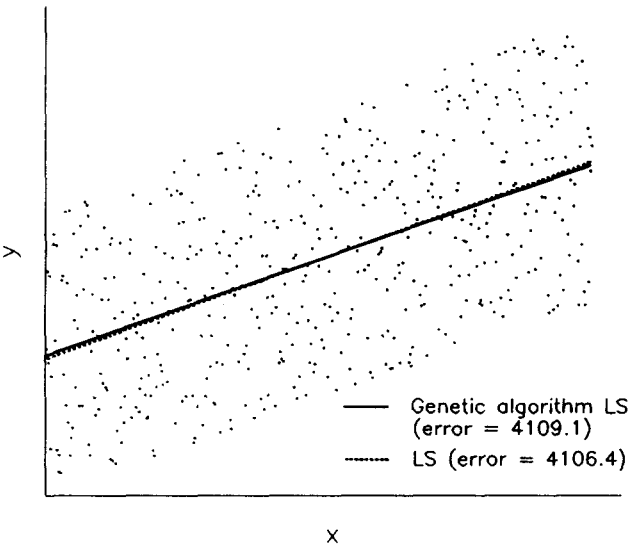

Fig. 1. Comparison of genetic algorithm LS and traditional LS.

The simple genetic algorithm described earlier has been implemented in the C programming language by modifying the code found in Ref. 9. The genetic algorithm is run in the studies presented in this paper with the following parameters:

$$\text{popsize} = 100$$

$$P_{\text{crossover}} = 0.65$$

$$P_{\text{mutation}} = 0.1//\text{popsize}.$$

These values are consistent with De Jong's suggestions[20] for moderate population size, high crossover probability, and low mutation probability.

4.2. Example 1: line

This problem involves the fitting of data to a line having the form $y = \Theta_1 x + \Theta_2$. Both Θ_1 and Θ_2 must be selected so as to minimize the square of the distance between the points and the curve. A data set containing 900 points was selected to be fitted. A genetic algorithm was used to perform both an LS and an LMS curve fit. The strings manipulated by the genetic algorithm were 22 bits long, thereby representing a search space with $2^{22} = 4.19 \times 10^6$ possible solutions.

The results of the genetic algorithm LS curve fit are shown and compared to a classic LS curve fit in Fig. 1. The genetic algorithm found a near-optimum solution in approx 130 generations, or 13,000 function evaluations. The results of the genetic algorithm LMS curve fit are shown and compared to results found using PROGRESS software which can be used for LMS curve fitting[7] in Table 1. It is worth mentioning that the PROGRESS took approx. 45 min (all computational times given in this paper were determined using a 486-based personal computer) to determine a solution that had a median squared error term of 0.37, while the genetic algorithm LMS method located a solution with a median squared error term of 0.1915 after only 4 min.

Thus, the genetic algorithm found a better solution than PROGRESS to the curve-fit problem, in less time. This is probably due to the fact that PROGRESS does not consider each possible combination of values for the search parameters when the number of points is very large. The genetic algorithm LMS curve-fit routine is not restricted in this way.

4.3. Example 2: parabola

The second example in this section involves fitting data to the equation of a parabola. This curve-fit problem is more difficult for a genetic algorithm, simply because the bit strings it processes are 50% longer (they increase from 22 bits to 33 bits). This is, of course, because three parameters must be determined instead of the two needed in the previous example. There are, however, no other changes to be made to the genetic algorithm code. From this point on in the paper, consideration will be made only of the LMS curve-fitting approach. Many of the examples presented in the following section, along with the current parabola example, could be solved using an LS approach. Using an LS approach would not present an especially difficult challenge for a genetic algorithm. However, for the purpose of brevity, only the LMS approach will be considered.

Results from the genetic algorithm LMS solution are shown in Fig. 2. Here, a genetic algorithm LMS solution is found when PROGRESS requires the use of transformed regressors to determine a solution. As in the previous example of the line, the genetic algorithm evaluated approx. 12,000 fitness functions and took approx. 4 min to run. Note that once again outliers have been intentionally introduced into the data set.

4.4. Summary

This section has provided the details necessary for applying a genetic algorithm to either an LS or an LMS curve-fitting problem. Once the issue of parameter coding has been resolved, the fitness function can easily be altered to toggle back and forth between the two curve-fitting approaches. This is an important point, because the basic curve-fitting algorithm remains unchanged. Additionally, the genetic algorithm approach does not require that an LMS curve fit pass exactly through any of the data points, which allows the genetic algorithm to locate improved solutions in some instances. In a later section it will be shown that in the domain of LS curve fitting, there are instances in which the model equation is difficult, if not impossible, to solve for the theoretical LS constants. In such instances, a genetic algorithm is not hindered in any way, and therefore represents an effective means for fitting the data.

Table 1. Comparison of genetic algorithm LMS to PROGRESS for a line

	Genetic algorithm LMS	PROGRESS
Θ_1	0.9988	1.0078
Θ_2	0.3619	0.0685
LMS error	0.1915	0.3700
Run time	4 min	45 min

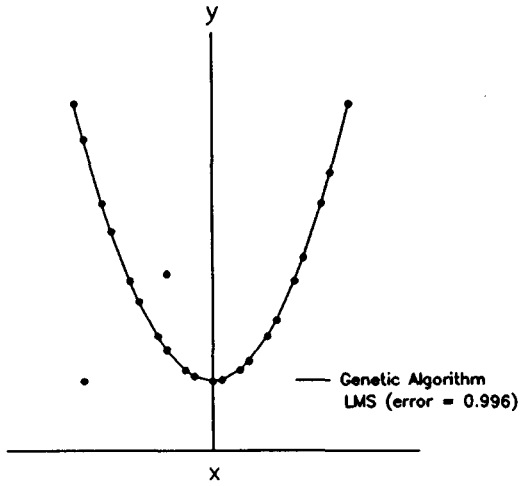

Fig. 2. Genetic algorithm LMS curve fit to a parabola.

5. EXAMPLES OF GENETIC ALGORITHM CURVE FITTING

This section provides several examples of genetic algorithm curve fitting. Each of these examples pertains to engineering problems that are, for one reason or another, of immediate interest to researchers at the U.S. Bureau of Mines. Sometimes, the curve fitting is the main concern of the problem. But, more commonly the curve fitting must be performed as a precursor to a more detailed problem, such as equipment design or process control.

The examples provided represent but a few of the areas in which researchers at the U.S. Bureau of Mines have applied genetic algorithms to curve fitting. These examples demonstrate some of the problems associated with conventional LS and LMS curve-fitting algorithms, and bring to light some of the strengths of genetic algorithm curve fitting. In total, five examples are presented:

(1) a solution is found to the non-linear Ree–Eyring equation used to describe the viscosity of non-Newtonian fluids;
(2) a system characterization problem is solved to determine the magnitude of a resistor and an inductor in an RLC electric circuit;
(3) the parameters in a cart–pole system (sometimes called the broom balancer system) are determined by solving a system characterization problem;
(4) a temperature-distribution problem is solved, yielding the heat conducting properties of composite material; and

(5) a supersonic flow problem is solved.

All of these problems are solved using the genetic algorithm LMS curve-fitting technique introduced above.

5.1. Ree–Eyring Equation

One of the major concerns of the mineral-processing industry is the dewatering of mineral slurries. Researchers at the U.S. Bureau of Mines have developed innovative techniques for dewatering numerous mineral slurries, and a number of these techniques are based on knowledge of the non-Newtonian components of viscosity of the slurry. The non-Newtonian components of viscosity can be found by curve-fitting data.

The Ree–Eyring equation allows for the estimation of the non-Newtonian components of viscosity, and is of particular interest to the minerals industry in the area of dewatering phosphatic clay wastes. The Ree–Eyring equation may be written as

$$y = C_1 + \frac{C_2 \sinh^{-1}(C_3 x)}{C_3 x}, \tag{9}$$

where y is the viscosity (the dependent variable), x is the shear rate (the independent variable), and c_1, c_2 and c_3 are the constants to be determined. In most instances, experimental viscosity and shear rate data is available through laboratory tests. However, curve fitting the data is not always easy because of the form of the equation, which is highly nonlinear.

Attempts to curve-fit data to the Ree–Eyring equation using traditional LS techniques were unsuccessful. The difficulty arises in solving for the constant c_3. A symbolic math package was used in this attempt, but the package failed to produce a solution. However, the Marquardt method[4] can be used to solve for the associated parameters. Unfortunately, the Marquardt method requires good initial guesses for the coefficients. The genetic algorithm, on the other hand, does not have such a requirement.

Fortunately, a genetic algorithm was up to the task of solving the Ree–Eyring curve-fit problem. The process of fitting the data using a genetic algorithm is the same as in the previous examples. The parameter set (c_1, c_2 and c_3) is again represented as a 33-bit string of 0's and 1's and evaluated using the previously discussed fitness function which allows for the minimization of the distance between the data points and the resulting curve.

Ref. 21 used a transformation method to solve the Ree–Eyring equation for a sodium ion-exchanged clay. The data used in that report were also used here, and the results of the curve produced using the genetic algorithm are compared to those previously obtained. Once again the genetic algorithm quickly converged to a quality solution; the smallest error produced by the genetic algorithm calculated curve (Error = 1.57) is less than that of the curve calculated by the transformation method used by Stanley, Webb and Scheiner (Error = 12.55). Figure 3 shows how well the curve calculated by the genetic algorithm ($c_1 = 168.8$, $c_2 = 34.20$ and $c_3 = 0.097$) fits the data. c_1, c_2 and c_3 were allowed to range between 0.0 and 200.0 for the coding necessary in the genetic algorithm application.

In the previous examples involving a line and a parabola, using a genetic algorithm to do LMS curve fitting made sense because the genetic algorithm approach took less time or was more accurate than the PROGRESS algorithm. However, in the example of the Ree–Eyring equation presented in this section, a genetic algorithm was used because the traditional techniques were not as efficient. It is encouraging to note that extending the genetic algorithm LMS curve-fitting approach used to fit a line and a parabola to the Ree–Eyring equation required very little effort. In fact, there is simply one line in the computer code that had to be changed; that being the model equation (which was changed from that of a line or a parabola to that of the Ree–Eyring equation). Both the fitness function and the coding scheme remained unchanged. Furthermore, to complete a genetic algorithm curve fit based on the LS criteria instead of on the LMS criteria, only the fitness function would be altered.

The successful application of a genetic algorithm to the problem of curve fitting data to the Ree–Eyring equation has enabled researchers at the U.S. Bureau of Mines to improve dewatering techniques. Results obtained using the genetic algorithm LMS curve-fitting technique represent potential cost savings for the minerals industry. Furthermore, these results are helping Bureau researchers to develop dewatering techniques that help reduce waste imparted to the environment.

5.2. Electric circuit

This example is the first of several in which *system characterization* is considered. In a system-characterization problem, a mathematical model of a system is provided. However, the model is a "black-

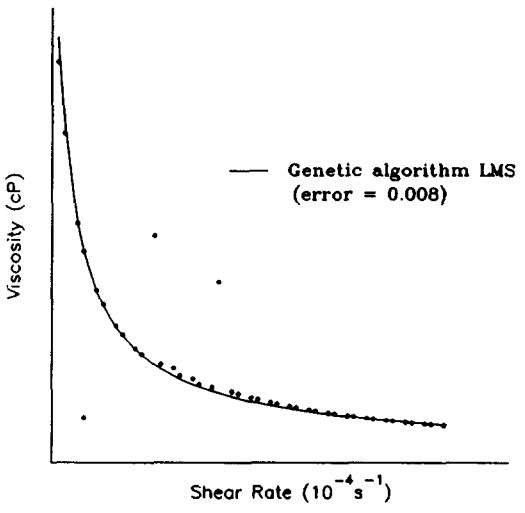

Fig. 3. Genetic algorithm LMS curve fit of Ree–Eyring equation.

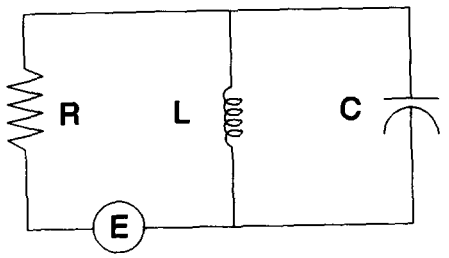

Fig. 4. A schematic of an RLC electric circuit.

box" model; the equations that describe the response of the system are unknown to the user. Thus, there exists some model $f(x_1, x_2, \ldots, x_m, \ldots, x_p, t)$ that prescribes the response of the system, f, as a function of time, t, and several parameters, x_1 through x_p. Also provided are data points collected by measuring the response of the system to some forcing function. These data points are of the form (f, t). Finally, both the forcing function and several of the system parameters, x_1 through x_m, are known. The problem is to use the above information to determine the unknown system parameters, x_{m+1} through x_p. Fortunately, this problem can be solved using a genetic algorithm curve-fitting approach. And in fact, it is a great advantage that the genetic algorithm can operate on such a black-box model.

A classic electric circuit can be used to demonstrate a system characterization problem. Figure 4 shows a schematic of an RLC circuit that contains a resistor, an inductor, and a capacitor. This system has a transient voltage response that can be measured. The voltage can be described by the following equation

$$\frac{d^2v}{dt^2} + \frac{1}{RC}\frac{dv}{dt} + \frac{1}{LC}v = 0 \qquad (10)$$

where v is the voltage, R is the resistance, L is the inductance, and C is the capacitance. This differential equation has been solved and put in the form of a computer model that receives R, L, C, $v(t)$, and t to predict $v(t+dt)$. For the purposes of this example, consider the form of the model to be unknown. Information is provided about the magnitude of the capacitance, C, as well as data points concerning voltage and time (v, t). The task is to determine the values of resistance and inductance in the circuit.

The system characterization problem described above brings to light another strength of the genetic algorithm curve fitting: no manipulation of the model equations is needed. In traditional LS curve fitting, derivatives of the model equations are computed with respect to unknown constants. This manipulation is not necessary in genetic algorithm curve fitting; only access to the model of the system is needed so as to compute a fitness function as described earlier in this paper. Additionally, the basic approach outlined previously is still very much valid. No changes to either the coding scheme or the fitness function evaluations are necessary.

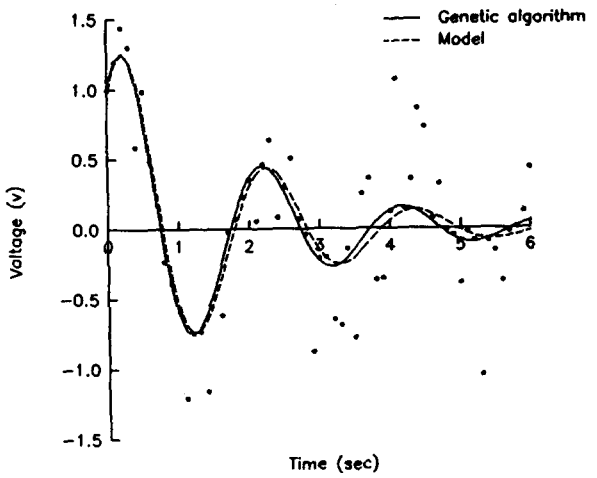

Fig. 5. Genetic algorithm LMS curve fit in RLC circuit problem.

Genetic algorithm LMS curve fitting has been used to solve the RLC electric circuit system characterization problem described above. The particular example considered 60 data points that were produced by adding Gaussian noise to model predicted values. Additionally, selected outliers were added. The model used to produce the data considered a resistance of 1 Ω and an inductance of 10 micro H. Figure 5 shows the results of the curve fit. Two curves are plotted in the figure:

(1) the model with the original parameters used to produce the data (this figure has had the noise removed); and
(2) the model predicted response with the parameters as determined by the genetic algorithm LMS curve fit.

Table 2 further demonstrates the accuracy of the approach.

This RLC electric circuit system characterization problem has provided a situation in which the genetic algorithm curve-fitting approach has proven to be effective and necessary. Situations in which the model equations that describe the system are unknown create insurmountable barriers for traditional LS and LMS curve-fit algorithms. Neither the form nor the nature of the model equations is of any consequence to a genetic algorithm's ability to solve the problem; only that the model is available so a fitness function can be calculated. Certainly, the nature of the fitness landscape does affect the convergence of a genetic algorithm.

In several RLC circuit systems, the effective resis-

Table 2. Performance of genetic algorithm LMS in RLC circuit problem

	Actual parameters	Genetic algorithm LMS
Resistance	1.0	1.034
Inductance	10.0	10.293
LMS error		0.071
Run time		101 s

tances and inductances vary with time. Quite frequently, these values cannot be measured directly, yet are vital to establishing suitable control of the system. In these instances, genetic algorithm curve fitting has demonstrated tremendous potential. In fact, there are a number of systems in the mineral-processing field that provide system characterization problems; so many so, that an approach to adaptive process control has been developed that includes a genetic algorithm curve-fitting approach to solving system characterization problems.[5] The remaining examples in this paper describe system characterization problems from various fields of engineering.

5.3. Cart–pole system

Consider a cart–pole system (sometimes referred to as a broom balancer system) as depicted in Fig. 6. A cart is free to translate along a one-dimensional track while a pole is free to rotate only in the vertical plane of the cart and track. This problem environment is a classic control problem, and has been somewhat unofficially adopted as a testing ground for artificial-intelligence-based process-control systems. Here, the system is used to demonstrate a system characterization problem. In the traditional control problem, a multi-valued force, F, is applied at discrete time intervals in either direction to the center of mass of the cart. The objective is to apply forces to the cart until it is motionless at the center of the track and the pole is balanced in a vertical position.

This task of centering a cart on a track while balancing a pole is representative of a number of important control problems. For instance, it is often used as an example of the inherently unstable, multiple-output, dynamic systems present in many balancing situations, e.g. two-legged walking and the aiming of a rocket thruster. Researchers at the U.S. Bureau of Mines are interested in the cart–pole system because they have used it to develop and test an adaptive process-control system based on neural networks, fuzzy logic and genetic algorithms.[5] One of the key aspects of the adaptive control system is the ability to solve system characterization problems by performing genetic algorithm curve fitting.

The system-characterization problem considered here involves computing system parameters on the basis of knowledge of the forcing function and the response of the system (data points that describe cart position and pole angle, both as a function of time). To solve the problem using the approach outlined in the previous example, a model of the cart–pole system is required. The state of the cart–pole system at any time is described by four real-valued state variables:

x = position of the cart;
$\dot{x}$ = linear velocity of the cart;
θ = angle of the pole with respect to the vertical;
and $\dot{\theta}$ = angular velocity of the pole.

The system is modeled by the following nonlinear ordinary differential equations:

$$\ddot{\theta} = \frac{g \sin\theta + \cos\theta \left[\dfrac{-F - m_p l \dot{\theta}^2 \sin\theta + \mu_c \operatorname{sign}(\dot{x})}{(m_c + m_p)} \right] - \dfrac{\mu_p \dot{\theta}}{m_p l}}{l \left[\dfrac{4}{3} + \dfrac{m_p \cos^2\theta}{(m_c + m_p)} \right]} \tag{11}$$

$$\ddot{x} = \frac{F + m_p l [\dot{\theta}^2 \sin\theta - \theta \cos\theta] - \mu_c \operatorname{sign}(\dot{x})}{(m_c + m_p)} \tag{12}$$

where:

g = -9.81 m/s^2, acceleration due to gravity;
m_c = 1.0 kg, mass of cart (as will be discussed later, this value changes with time);
m_p = 0.1 kg, mass of pole;
l = 0.5 m, length of pole;
μ_c = 0.0005, coefficient of friction of cart on track;
μ_p = 0.000002, coefficient of friction of pole on cart;
and

$-10.0N \leq F \leq 10.0\,N$ force applied to cart's center of mass.

The solution of these equations was approximated using Euler's method, thereby yielding the following difference equations:

$$\dot{\theta}^{t+1} = \dot{\theta}^t + \ddot{\theta}^t \Delta t \tag{13}$$

$$\theta^{t+1} = \theta^t + \dot{\theta}^t \Delta t \tag{14}$$

$$\dot{x}^{t+1} = \dot{x}^t + \ddot{x}^t \Delta t \tag{15}$$

$$x^{t+1} = x^t + \dot{x}^t \Delta t \tag{16}$$

where the superscripts indicate values at a particular time, Δt is the time step, and the values and $\ddot{x}^t$ and $\ddot{\theta}^t$ are evaluated using equations (11) and (12). A time step of 0.02 s was used because this time step struck a balance between the accuracy of the solution and the computational time required to find the solution. It is important to note that although the details of a model have been specified, in solving the system characterization problem only a black-box version of the model is available; the inputs to the black box are the system parameters and the current values of the state variables, while the output is the values of the state variables at the next time step.

The unknown system parameters are cart mass, pole mass and pole length. The solution of the system

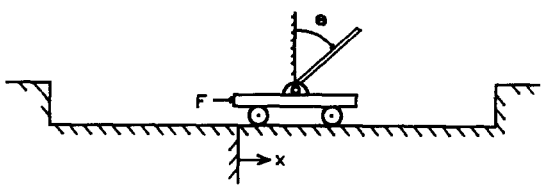

Fig. 6. A schematic of a cart–pole system.

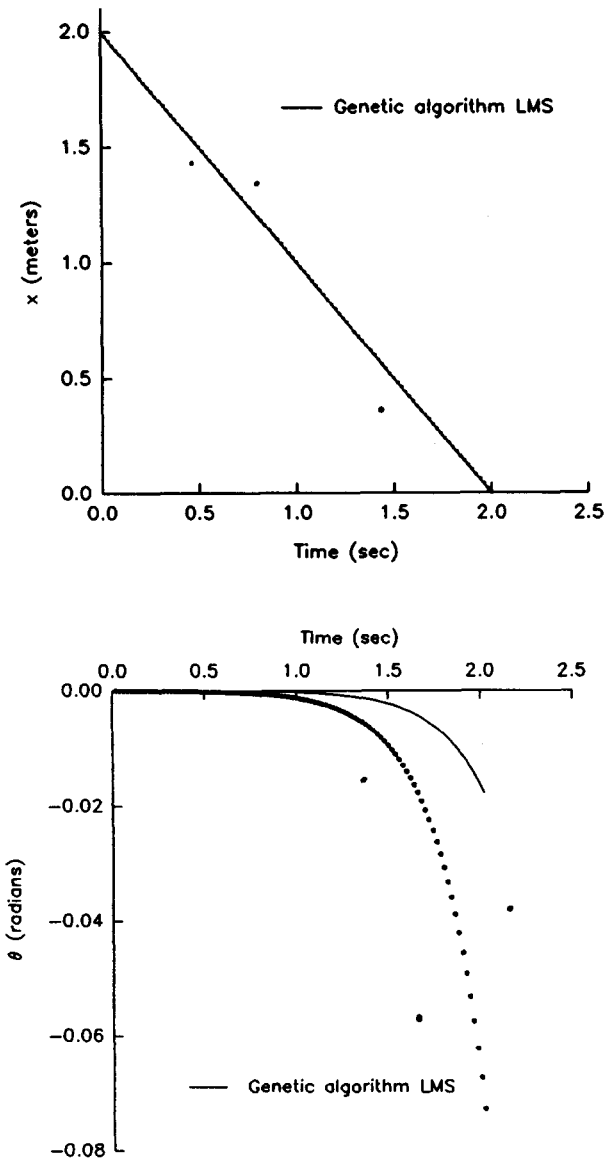

Fig. 7. Genetic algorithm LMS curve fit for cart position and pole angle.

characterization problem involves selecting prospective values for the unknown system parameters with a genetic algorithm, testing those prospective values by curve fitting the data using an LMS criterion, and allowing the genetic algorithm to locate the values of the system parameters that elicit the measured response. Upon completion, the solution of the system characterization problem yields the value of the system parameters that could not, for whatever reason, be directly measured in the cart–pole system. Thus, in a sense, the ability to solve the curve-fitting problem can compensate for the lack of sensory information from the environment that must be controlled.

Figure 7 shows results that demonstrate the ability of a genetic algorithm to solve the system-characterization problem presented above. The system parameters determined by the genetic algorithm produce a response that matches the actual response of the cart–pole system (as depicted by the data points) rather nicely. In fact, the parameters were each accurate within 3% of the actual values. It is important to note that the response of the system was measured for a short period of time because this curve-fit algorithm is actually part of a real-time adaptive control system in which only 2 s can be devoted to system characterization.

5.4. Temperature distribution

The example considered here involves solving a system-characterization problem involving a partial differential equation, namely, the one-dimensional heat equation. The U.S. Bureau of Mines has a large materials program that is concerned with the development and analysis of efficient ceramic materials, and with the production of "smart material". A small group of researchers in the materials program has identified a situation in which solving a system characterization problem can yield the thermal diffusivity of a composite material.

A thin bar of uniform cross-section is heated from both ends. As time passes, the temperature in the bar goes to a steady-state value. The thermal diffusivity of the material determines just what the temperature distribution in the bar looks like, as a function of time and space. The temperature distribution is governed by the following partial differential equation:

$$\frac{\partial T}{\partial t} = a^2 \frac{\partial^2 T}{\partial x^2} \tag{17}$$

where T is the temperature, a^2 is the thermal diffusivity of the bar, t is time, and x is the distance along the axis of the bar. For various boundary conditions, an analytical solution to the heat equation can be written, but the result is a highly nonlinear equation. Thus, consider a black-box model like the one discussed in the previous examples.

In this example, data is available in the form of (x, t, T). Despite the increased complexity of considering changes in both time and distance, this problem can still be reduced to a curve-fitting problem: select the value of a^2 that when used in conjunction with the black-box model produces an appropriate temperature distribution. Actually, several curves must be fitted simultaneously; the data is broken down into ordered pairs of the form (x, T) for several specified times.

The above problem has been solved using the genetic algorithm curve-fitting techniques outlined previously. Figure 8 shows the results of the effort. The temperature of a ceramic rod has been plotted as a function of distance along the rod at five distinct times. The genetic algorithm LMS curve-fitting approach predicted a value of $a^2 = 0.09785$ in a run time of approx. 22 s; subsequent analysis determined the value of $a^2 = 0.100$. The LMS error for this run was 0.8396. Thus, in a very short time, the genetic algorithm LMS approach yielded a solution that was 97% accurate.

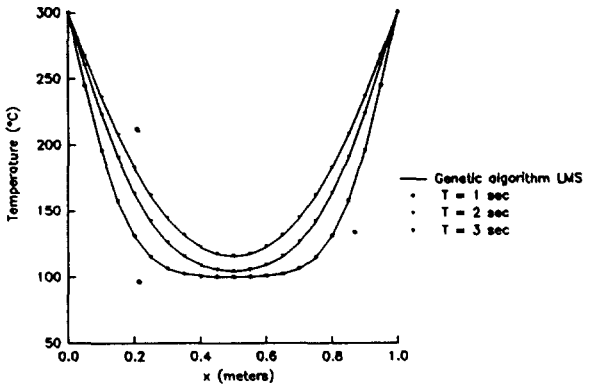

Fig. 8. Genetic algorithm LMS curve fit to temperature distribution.

The above outlined genetic algorithm LMS approach provides a method for rapidly determining the thermal diffusivity of a material. By measuring nothing more than the temperature of a rod at various locations and times, the heat-conducting properties of the material can be determined. This approach may provide economic savings in the materials industry.

5.5. Mach number about a circular cylinder

The previous examples have come from the areas of mineral engineering, electrical engineering, process control and materials development. This next example comes from the field of aerospace engineering. It involves determining the Mach number of a circular cylinder in an inviscid flow field. Figure 9 shows a schematic of the physical system. Pressure data is provided in the form of ordered pairs (θ, P) where θ describes a location on the cylinder and P is the pressure at the surface of the cylinder.

The system can be described and modelled using a source panel method of numerical computation.[22] The model happens to be complex and relatively computationally cumbersome. Here, however, the only thing that is important is that a model of the system exists. Figure 10 shows the results of solving the system-characterization problem (the Mach number of the flow is to be determined) using genetic algorithm LMS curve fitting. In this particular example, the actual Mach

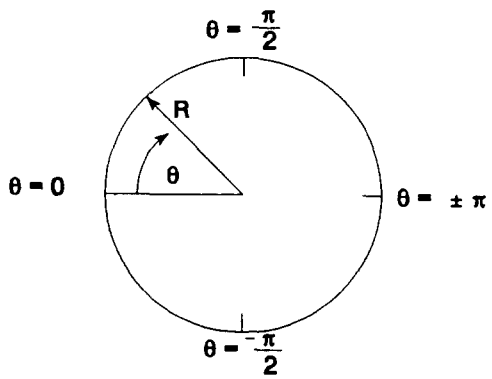

Fig. 9. A schematic of the circular cylinder.

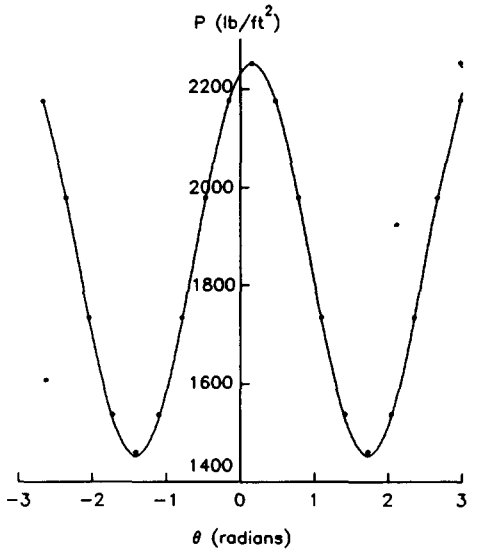

Fig. 10. Genetic algorithm LMS curve fit for the Mach number problem.

number was 10.00; the genetic algorithm approach computed the Mach number to be 10.069. Again, the genetic algorithm was quite effective.

6. SUMMARY

Curve-fitting problems appear in a number of engineering disciplines. Two of the more popular approaches to curve fitting are the classical LS and LMS algorithms. However, both of these approaches have their shortcomings. LS methods can be faced with situations for which it is extremely difficult, if not impossible, to solve for the curve-fitting constants. LMS methods are difficult to apply in nonlinear environments. Furthermore, both of these methods can be rendered ineffective in system-characterization problems. This paper has proposed a method of curve fitting that involves the use of a genetic algorithm.

Genetic algorithms are search algorithms based on the mechanics of natural genetics. They are able to locate near-optimal solutions after having sampled only small portions of the search space, and they are flexible enough to be effective in a wide range of problems. Their novel approach to solving search problems is fundamentally different from more-conventional search techniques. They have been used in this paper to solve curve-fitting problems using both the LS and the LMS criteria.

Examples have been provided, demonstrating the effectiveness of using genetic algorithms for solving curve-fitting problems. These examples come from various engineering application areas such as aerospace, materials, minerals and process control. The examples presented represent real examples that are of current interest. Of particular interest should be the fact that, despite the very different nature of the examples discussed, the genetic algorithm approach to curve fitting described requires very little alteration even

when changing from an LS to an LMS criteria. At first glance, it appears that only the computer model needs to be replaced. However, this would be an oversimplification. The user must still be experienced with genetic algorithms to ensure maximum performance.

REFERENCES

1. Rousseeuw P. J. Least median of squares regression. *J. Am. statist. Ass.* **79**, 871–880 (1984).
2. Karr C. L. Air-injected hydrocyclone optimization via genetic algorithm. In *The Genetic Algorithms Handbook* (Edited by Davis L. D.), pp. 222–236. Van Nostrand Reinhold Company, New York (1991).
3. Karr C. L. Analysis and optimization of an air-injected hydrocyclone. Doctoral dissertation: The University of Alabama, Tuscaloosa, AL (1989).
4. Hines W. W. and Montgomery D. C. *Probability and Statistics in Engineering and Management Science.* John Wiley & Sons, New York (1972).
5. Karr C. L. and Gentry E. J. Control of a chaotic system using fuzzy logic. In *Fuzzy Control Systems* (Edited by Kandel A. and Langholz G.), pp. 475–497. CRC Press, West Palm Beach, FL (1993).
6. Huber P. J. *Robust Statistics.* John Wiley & Sons, New York (1981).
7. Rousseeuw P. J. and Leroy A. M. *Robust Regression and Outlier Detection.* John Wiley & Sons, New York (1987).
8. Karr C. L., Stanley D. A. and Scheiner B. J. *A Genetic Algorithm Applied to Least Squares Curve Fitting* (Report of Investigations No. 9339). U.S. Department of the Interior, Bureau of Mines, Washington, DC (1991).
9. Goldberg D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison Wesley, Reading, MA (1989).
10. Plackett R. L. Studies in the history of probability and statistics XXIX: The discovery of the method of least squares. *Biometrika* **59**, 239–251 (1972).
11. Edgeworth F. Y. On observations relating to several quantities. *Hermathena* **6**, 279–285 (1887).
12. Huber P. J. Robust regression: asymptotics, conjectures and Monte Carlo. *Ann. Stat.* **1**, 799–821 (1973).
13. Wald A. The fitting of straight lines if both variables are subject to error. *Ann. math. Statist.* **11**, 284–300 (1940).
14. Bartlett M. S. Fitting a straight line when both variables are subject to error. *Biometrics* **5**, 207–212 (1949).
15. Bickel P. J. On some analogues to linear combination of order statistics in the linear model. *Ann. Statist.* **1**, 597–616 (1973).
16. Andrews D. F. A robust method for multiple linear regression, *Technometrics* **16**, 523–531 (1974).
17. Siegel A. F. Robust regression using repeated medians. *Biometrika* **69**, 242–244 (1982).
18. Davis L. D. *The Genetic Algorithms Handbook.* Van Nostrand Reinhold, New York (1991).
19. Holland J. H. *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor, MI (1975).
20. De Jong K. A. Analysios of the behavior of a class of genetic adaptive systems. *Diss. Abstr. Int.* **36**, 5140B (1975).
21. Stanley D. A., Webb S. W. and Scheiner B. J. *Rheology of Ion-exchanged Montomorillonite Clays* (Report of Investigations No. 8895). U.S. Department of the Interior, Bureau of Mines, Washington, DC (1986).
22. Press W. H., Flannery B. P., Teukolsky S. A. and Betterling W. T. *Numerical Recipes in C.* The Cambridge University Press, Cambridge (1988).

AUTHOR'S BIOGRAPHY

Chuck Karr is now an Assistant Professor in the Engineering Science and Mechanics Department at the University of Alabama. He also works closely with the U.S. Bureau of Mines. He received his Ph.D. in Engineering Mechanics from the University of Alabama in 1989. He works to develop engineering applications of artificial intelligence in the mineral processing industry. His interests include genetic algorithms, fuzzy logic, fluid dynamics and process control.