ELSEVIER Computational Statistics & Data Analysis 22 (1996) 633–651

COMPUTATIONAL STATISTICS & DATA ANALYSIS

Genetic algorithms and their statistical applications: an introduction

Sangit Chatterjee[a,*], Matthew Laudato[a], Lucy A. Lynch[b]

College of Business Administration, Northeastern University, 214 Hayden Hall Boston, MA, 02115, USA
Lynch Analysis & Design, Brookline, MA, 02115, USA

Received May 1995; revised December 1995

Abstract

Genetic algorithms (GA) are stochastic optimization tools that work on "Darwinian" models of population biology and are capable of solving for near-optimal solution for multivariable functions without the usual mathematical requirements of strict continuity, differentiability, convexity and other properties. The algorithm begins by choosing a large number of candidate solutions which propagate themselves through a "selection criteria" and are changed by the application of well-developed genetic operators. GAs are applied to problems in statistical estimation and the results are compared to the output of standard software. It is argued that many statistical and mathematical restrictions that usually restrict modeling and analysis can be dispensed with by employing the GA as an optimization technique. The use of GAs for solving discrete optimization problems with applications in statistics for the variable selection problem in regression and other multivariate statistical methods are also discussed.

Key words: Binary digit; Evolutionary operators; Natural selections; Statistical modeling; Stochastic optimization

1. Introduction

Much of the work of statisticians involves model building, estimation of model parameters and validation of such models. Most applied statistical work is clearly undertaken with mathematical and computational considerations and restrictions in mind. Think of two simple models: first, $y = x\beta + \varepsilon$ and second, $|y| = x\beta + \varepsilon$, where ε is i.i.d. $N(0, \sigma^2)$. As the reader knows, the first model is easily estimated by

PII S0167-9473(96)00011-4

using standard calculus and readily available software, while this is not the case for the second. In this paper, we will describe a method of estimation in which there is no difference in the difficulty of estimating the parameters β in these two models. It may be argued that the second model is not realistic. But we could, for example, think of y as risk and the model as being of absolute risk in empirical finance. The mathematical requirements of smoothness, continuity and differentiability for the functional form of a model are artificial requirements imposed by the readily available methods of estimation. That this is so is easily seen when we consider nonstandard distribution laws for the errors ε and the associated difficulties of finding maximum likelihood estimates by relying on methods of standard calculus. A genetic algorithm (GA), the subject of this paper, removes the restrictions on allowable models and error laws. The genetic algorithm is so indifferent to model that the minimum is required: Given a set of parameters, the response can be calculated and the value of the error function determined. This provides a definition of "better" fit and it is all that is required. GAs require "softer" mathematical requirements and in turn provide "softer" (although perhaps very good) assurances of optimization.

A GA is a simple heuristic optimization tool (for both continuous and discrete variables) that solves for near-global optimal value even for poorly behaved functions. This is done by iteratively applying principles of 'Darwinian natural selection" to a population of computer representations of the solution domain. The algorithm attempts to mimic the natural evolution of a population by allowing solutions to reproduce, creating new solutions, and to compete for survival in the next iteration. Every generation will have members that may not be an improvement over the previous generation and far from the global optimum. Average fitness, though, typically improves over generations and the best (most fit) solution after many generations is usually near the global optimum. Diversity among candidate solutions is very helpful for convergence towards the optimal solution. A formal structure for genetic algorithm has three components; (1) the environment and the elements in the environment (the candidate solutions); (2) an adaptive plan (application of evolutionary operators) and (3) selection based on a measure of performance (the fitness of the solutions).

Holland (1975) pioneered the development of the GA. Now it has become an area of its own. Holland et al. (1986), Goldberg (1989) and Koza (1992), Holland (1993) are some very good reference points to the literature. For other applications of the GA in statistics see Chatterjee and Laudato (1994). In a broad context, a GA can be thought of as a dynamic self-organizing system, the union of simple systems driven towards a complex system guided by the fitness criteria. A philosophical discussion along these lines can be found in Gell-Mann (1994).

The GA can be employed not only to estimate parameters of standard and nonstandard models but with the aid of a bootstrap like tool, can also be used to estimate the standard errors of the parameters. The point here is that the union of a GA and a resampling tool for estimating the variability of estimates can free data analysis from several kinds of computational limitations. The computer becomes the central focus for data analysis. This leaves the analyst with complete freedom

for model choice and choice of probability distributions without worrying about computational difficulty.

The rest of the paper is organized as follows. Section 2 discusses the genetic algorithm in more detail but in the abstract while in Section 3 we formulate the statistical estimation problems in terms of a GA. In Section 4, consistent with the introductory nature of the paper, we produce a hand simulation of a simple optimization problem. Section 5 presents the results for different problems to illustrate the nature of the GA. In Section 6 we briefly discuss the GA in solving discrete problems. Here we try to illustrate the potential usefulness of a GA for solving many difficult problems in statistics especially those which can normally be attacked only through special purpose algorithms. Some technical issues of GAs are addressed in Section 7. The last section contains conclusions of the paper and an overall discussion. We consciously try to preserve the introductory nature of the paper throughout.

2. What is a genetic algorithm?

A genetic algorithm comprises three parts: Solution representation; operators which produce altered solutions; and fitness selection. For problems that require real number solutions, a simply binary representation is used where unique binary integers are mapped onto some range of the real number line. Since the value of a binary 0 or 1 in an integer is position dependent, we can define the basic genetic operation, crossover (C), that splits a pair of binary integers at a random position and combines the head of one with the tail of the other and vice versa. The resulting pair of integers is in general different from the first, and the new numbers may be nearer to the optimal solution being sought. Additional operations, such as inverting a section of the binary representation (inversion, I) or randomly changing the state (0 or 1) of individual bits (mutation, M), also transform the population. Before each such cycle (generation), population members are selected on the basis of their fitness (the value of the objective function for that solution) to be the "parents" of the new generation.

Some authors present the GA with only selection and crossover. For continuous variables, additional operations such as inversion and mutation are useful and speed up convergence towards the global optima. In the literature, the theoretical basis for convergence of the GA towards the global optimum values is usually discussed through the formation and preservation of *schema* (local optimal patterns) at rates acceptable for solving practical problems. With a binary representation, schema are bit patterns based on a ternary alphabet: 0, 1, and * (do not care). The crossover operation allows two individual solutions, both of which may contain optimal schema, to share information and generate new potentially superior solutions. Holland's analysis of the convergence of genetic algorithms involves mapping them (through schema formation aided by evolutionary operations) to the decision making in one and multi-armed bandit problems. He shows

the equivalence of optimal decisions for bandit problems in decision theory and the optimal growth of desirable schema through the so-called fundamental theorem of genetic algorithms (FTGA). These topics are further discussed in Section 6.

3. Problem encoding in GA

The domain for all problems studied here is $\mathbb{R}$, the real numbers, so we choose to represent possible solutions (the parameter values being estimated) as binary integers, which are then mapped onto the real numbers. If, for example, we wish to encode solutions on the real interval $[-d, d]$, the binary number $000\ldots000$ would represent $-d$, and the number $111\ldots111$ encodes d; as we count up from $000\ldots000$, adding binary 1 to an existing number increases its value by $d/2^{D-1}$, where D is the length (number of digits) of the binary representation. This coding scheme provides simple scaling with d, and further permits the use of fast, bitwise operations on binary integers during processing.

A large initial population of random candidate solutions is generated. These are then continually transformed through: (1) Selection and (2) Crossover and other operations modeled on genetic reproduction. These steps are repeated for a prescribed number of steps. The selection step encourages good solutions to propagate while weeding out poor solutions – implementing the FTGA. Extreme selection pressure can be too much of a good thing though – all solutions except the current best are weeded out of the population and no scope for improvement exists. Such homogeneity in the solution pool is called convergence by the computer science community but it is the worst condition for convergence in the parameter estimates; the GA is fated to stick at a local optimum value. To overcome such premature convergence through homogeneity, sexual reproduction and other operations such as inversion and mutation are employed. These operators introduce diversity among the population members (candidate solutions) and prevent the algorithm from getting stuck at a local optimum value.

Suppose we have data consisting of N observations (y, x), where y is univariate and x is k-variate. We assume that y and x are related by a function f (with unknown parameters q) through

$$y = f(x) + \varepsilon, \tag{1}$$

where ε is assumed to be stochastic and distributed as an arbitrary probability function. The task is then to find the minimum value of the error E over N observations:

$$E = \sum_{i=1}^{N} h(f(x_i), y_i, q), \tag{2}$$

where $h(f(x_i), y_i, q)$ is the error function for the model and norm. If the distribution of ε is known, the estimation of $f(x)$ is typically carried out by the method of maximum likelihood. When the distribution of ε is unknown, other norms are used

including least squares and sum of absolute deviations. Given these, we can estimate the associated parameters by choosing a large number of random or pseudo-random parameter sets, and find the optimal set by using a genetic algorithm with (2) as its fitness function.

An example is useful to illustrate this procedure. We will choose the model $f(x) = \alpha x + \beta y + \varepsilon$ under the least-squares norm. Then q is a member of the set of 2-vectors (q_1, q_2) on $\Re$, representing possible values of α and β. Our candidate solutions are then pairs of binary integers that are mapped onto the domain of q as described above. A solution is evaluated by performing the sum in (2) as

$$E = \sum_{i=1}^{N} [y_i - (q_1 x_i + q_2)]^2. \tag{3}$$

The candidate's fitness is taken as the reciprocal of this value; this has the useful property of increasing fitness as E is minimized (Remark 1).

Suppose, for example, that we begin with 1000 initial pairs of random integers. Each of these would be mapped to a real number and the sum in (3) performed to evaluate the pair's fitness. To create a new population of solutions a tournament is held. Two candidate pairs are chosen at random and compared for fitness. The more fit has two copies of itself placed in the next generation with high probability, typically 0.75. The less fit survives this tournament selection with low probability, usually 0.25. This continues until 1000 solutions have been copied into the new generation (Remark 2). We then perform the three genetic operations (crossover, inversion, and mutation) to transform the population.

Fig. 1 illustrates the effects of the operators on binary numbers. In Fig. 1(A), two candidate solutions B_1 and B_2, represented by their dyadic expansion of n digits, are undergoing a crossover (C) operation at the randomly chosen bit location j to produce two new solutions B_1^* and B_2^*. The first j bits of B_1^* are the same as that of B_1 while the last $(n - j)$ bits of B_1^* are those of B_2. The digits of B_2^* are similarly formed (Remark 3). Fig. 1(B) shows a candidate solution B_1 undergoing an inversion (1) operation at randomly chosen points j and m. The inverted candidate solution B_1^* has the first $(j - 1)$ and last $(n - m - 1)$ digits as that of B but the $(j + 1)$th digit through the mth digit of B_1 has been inverted in B_1^* in the corresponding positions. In Fig. 1(C), a candidate solution B_1 is undergoing mutation (M) at the two bits b_2 and b_k (chosen randomly with a given rate, typically 1 in 1000) to not-b_2 and not-b_k.

GAs have natural connections to both biological and to the traditional approaches to function optimization. The biological connection is obvious – the binary digits correspond to a chromosome, crossovers correspond to mating with random shuffling, mutation corresponds to physical mutation, etc. For an analogy with the traditional approach, we can say that the binary digits correspond to an orthogonal direction system, crossovers represent moving randomly in multiple directions simultaneously from one point of the surface to another point while mutation refers to searching along a single randomly chosen direction.

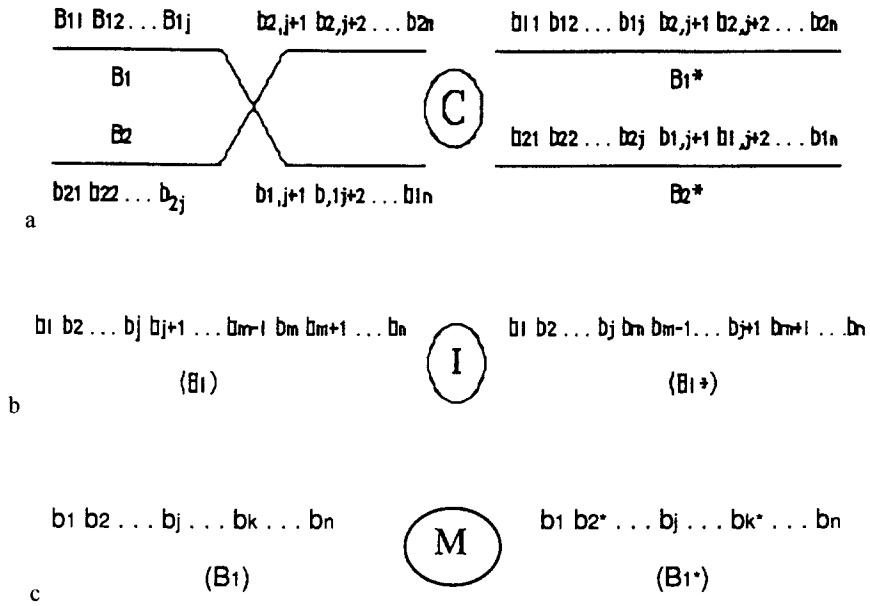

Fig. 1. (a) Two candidate solutions B_1 and B_2 undergoing a crossover operation at the randomly chosen bit location j to produce two new solutions B_1^* and B_2^*. (b) A candidate solution B_1 is shown undergoing an inversion (I) operation at randomly chosen points j and m. (c) A candidate solution B_1 is undergoing mutation (M) at two bits b_2 and b_k.

To give the reader a better feel for why the GA may work better where traditional approaches fail, consider the example of a bivariate (or multivariate) irregular surface with many local maxima. We then have the following: (1) Each GA element (candidate solution) is a fixed point on the surface. (2) It is obvious that a traditional derivative-based approach may get stuck at a local maximum. With genetic algorithms, crossover (mating) provides a means for possible improvement over either parent especially when the parents straddle a local maximum and for sufficiently regular problems the offspring is usually at least as fit as the minimum fit parent. (3) Mutations provide a means of branching out to unexplored portions of the parameter space. (4) The different tuning constants (reproduction probabilities, mutation rates) must be carefully chosen so that one does not get stuck at a local maximum, and also does not continually drift away from promising regions of the surface.

A GA however is not a panacea. GAs are very computer intensive and can converge to a local maximum for extremely ill-behaved functions. Such local convergence can occur if the optimizing surface is very wiggly or the existence of a big spike in the midst of a trough. The chief reason for such behavior is the emergence of lower-order schemas dominating the population and rendering the problem to the status of GA deception (discussed further later in the paper). Finally, we should remember that even in the best of circumstances GA approaches are only

as good as the function to be optimized. A poor choice of fitness function is entirely possible in statistical applications.

Remarks (1) Though we have used the reciprocal of E in (1) as our criterion for fitness, there are other possibilities.

(2) Choosing the winner and the loser with a fixed probability (such as 0.75 and 0.25, respectively), often called the tournament rule (stronger player typically wins but upsets are possible), is only one of many possibilities. Other possibilities include assigning probabilities according to the relative fitness of the candidate solutions (called the roulette wheel method) or according to the ranks of the solutions. Most methods will give reasonable results.

(3) In multiparameter problems, we must choose whether to perturb one parameter at a time or to vary them all simultaneously. We call the former strategy "micro evolution" and the latter "macro evolution". The programs we have developed have an "evolution" switch which toggles between these two states: a change in state can be used to dislodge a population "stuck" as a local optimum.

4. Simulating a toy problem

In order to clarify the method, we present a complete hand solution to a "toy" problem. We wish to find the minimum value of the function $f(x) := x^2 - 42x + 152$, $0 \leq x \leq 63$ and x integer (This requirement is imposed only to simplify the presentation.) Since $2^6 = 64$, we will use 6-bit binary numbers (representing integers from 0 to 63) to represent our candidate solutions. Five such randomly created solutions, their function values, and the average and the minimum of the function values are represented in Table 1.

Given this starting random population we observe, the minimum value achieved in -225 while the average fit is -9.2. We then proceed through the two basic step: Selection and reproduction through the application of the evolutionary operators of crossover, mutation and inversion (C, I M), respectively.

Table 1

Solutions	Bit patterns						f
1	1	0	0	1	1	0	0
2	0	0	1	1	0	1	−225
3	1	0	0	0	1	1	−93
4	1	0	1	1	1	0	336
5	0	0	0	1	1	0	−64

f_{avg}: -9.2, $f_{\text{minimum}} = -225$.

Table 2

Solutions	Bit patterns						f
1	1	0	0	1	1	0	0
2	0	0	1	1	0	1	−225
3	1	0	0	0	1	1	−93
4	1	0	0	1	1	0	0
5	0	0	0	1	1	0	−64

$f_{\text{avg}} = -76.4$ and f_{minimum}: -225.

Table 3

Solutions	Bit patterns						f
1	1	0	0	1	1	0	0
2	0	0	1	1	0	1	−225
3	1	0	0	0	1	1	−93
4	0	0	1	1	0	1	−225
5	0	0	0	1	1	0	−93

$f_{\text{avg}} = -127.2$ and $f_{\text{minimum}} = -225$.

Table 4

Solutions	Bit patterns						f
1	1	0	1	1	0	1	287
2	0	0	1	1	0	1	−225
3	1	0	0	0	1	1	−93
4	0	0	0	1	1	0	−64
5	0	0	0	1	1	0	−93

$f_{\text{avg}} = -37.6$ and $f_{\text{minimum}} = -225$.

Selection: Suppose now, pairs consisting of solutions 1 and 4 are chosen randomly for selection. Since $f(1) < f(4)$, the member (4) will be replaced by the member (1) and the resulting population will be as shown in Table 2.

After five such random selections, the population looks as in Table 3.

Note after the selection process, no new solutions have been created but the frequencies of surviving members have altered significantly.

Crossover: Suppose members 1 and 4 are chosen for crossover and the location of crossover is decided to be the fourth bit (both chosen randomly). The population of solutions will then be as shown in Table 4.

Table 5

Solutions	Bit patterns						f
1	0	1	0	1	1	1	285
2	0	1	0	0	0	1	−273
3	0	1	0	0	1	1	−288
4	1	0	1	1	1	0	−285
5	0	0	0	1	1	0	−33

$f_{\text{avg}} = -232.8$ and $f_{\text{minimum}} = -288$.

We undergo four pairs of such crossing over the solution pool is given in Table 5.

Now, to display mutation, suppose the third member of the solution pool is chosen at random for mutation and the sixth bit is chosen (again at random) to undergo mutation. Thus, the mutated solutions is 0 1 0 1 0 1 giving a further improvement of f to -289. This step completes the cycle and the population undergoes the loop (selection and the three operations crossover, mutation and inversion) a fixed number of times (Remarks 4 and 5).

Remarks (4) The number of pairs undergoing selection can vary from problem to problem. In all our examples, we choose 100% selection (an arbitrary choice), i.e., if the number of candidate solutions is 100 we choose $\binom{100}{2} = 4950$ random pairs for selection. Thus any particular solution may be chosen zero or one or more times. Similar comments apply to percentages chosen for crossover, inversion and mutation operations.

(5) The fact that the average solution has decreased most of the time is only a coincidence and that the optimal value is found is more an accident than a property of a GA. A GA need not find an exact optimal solution (a needle in the hay stack) but most often finds solutions in the neighborhood of the global optimum.

5. Applications

In this section we illustrate parameter estimation through the GA for four models each with a special characteristic. These examples illustrate the power of genetic algorithms and the independence of the GA code from models and the space of norms or the underlying probability distributions. The code required for solving each of these problems is same except for evaluating the fit (selection) function. In each case, for comparison purposes, we provide the solution obtained from commercial software, SYSTAT (1992). We also provide standard errors of the parameter estimates computed by applying the GA on 250 bootstrap samples of the data. We employ the simple bootstrap of resampling the entire population and estimating the parameters from the resampled data. The default for the switch

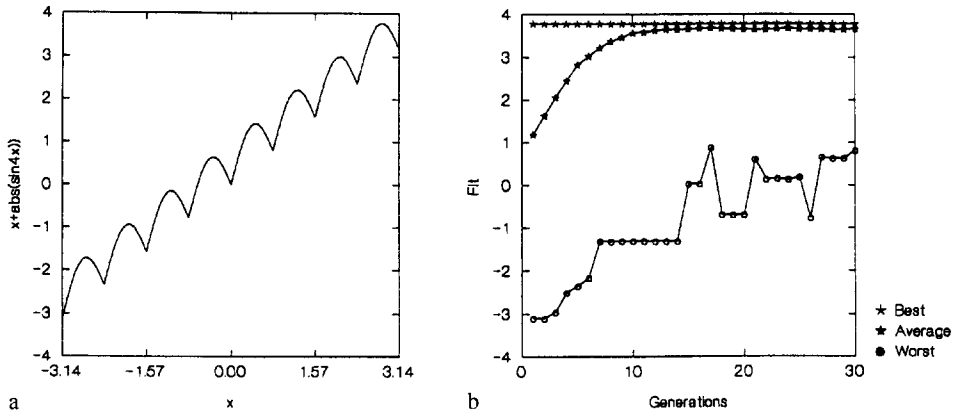

Fig. 2. The function $f(x) = x + \text{Abs}\,[\sin 4x]$, $-\pi \leq x \leq \pi$ is plotted in (a), (b) plots the best, average and worst fit solutions (for the variable x) with generations (G) for a particular run of the GA. Though the worst fits improve somewhat with generation, every generation keeps poor fit solutions that provide genetic diversity. The best-fit solutions converges with 5 generation and dominates the average fit (which converges somewhat slower) for all generations.

Evolution is macro but it is set to micro for Problem 3 and 4. For all problems, unless otherwise mentioned, the population size is 1000, the number of generations the program is run 30 and the selection, crossover, inversion and mutation (per bit) percentages are set at 70, 65, 75 and 0.1, respectively.

Problem 1 (*A deterministic problem*). We are interested in finding the global maximum of the function given by $f(x) = x + \text{Abs}\,[\text{Sin}\,4x]$, $-\pi < x < \pi$. The function is plotted in Fig. 2(A). The computer algebra system *MATHEMATICA* (Wolfram, 1993) failed to find to global maximum and returned the message: "*The problem involves transcendental functions in an essentially nonalgbraic way*". The GA solution is $x = 2.81206$ which is the correct solution to 3 decimal places obtained by grid search. The best fit, the average fit and the least fit is plotted against the number of generations G in Fig. 2(B). The best solution is achieved within 3 generations while the average solution stabilize around generation 20 but still remains lower than the "best-fit" solutions. The worst-fit solutions also improve but very slowly. The point of this (somewhat trivial) exercise is to demonstrate the robustness of a GA where a conventional calculus-based method can fail easily for a problem of this type.

Problem 2 (*An exercise in multiple regression*). A multiple regression model given by $y = \alpha + \beta x_1 + \gamma x_2 + \varepsilon$ is estimated for data consisting of Angell's index of "moral integration" for 29 US cities with two explanatory variables, heterogeneity and mobility indices (Fox,, 1984). Least squares is used for the fitness function. The

Table 6
Solution characteristics of the GA

Problem	Model	Norm	Standard solution[a]	GA solution
Deterministic	Max $\{x + \text{Abs Sin}(4x)\}$, $-p < x < p$	NA	[b]	2.81206
Angell	$y = b_0 + b_1x_1 + b_2x_2$	Least squares	21.8, −0.167, −0.214 (2.19, 0.0552, 0.0514)	21.797, −.1667 −.2137 (2.142, 0.049, 0.059)
CHD data	$y = \text{Logit}(b_0 + b_1x)$	Max. likelihood	−5.30, 0.111 (1.113, 0.024)	−5.436, 0.113 (1.258, 0.027)

[a] Obtained from SYSTAT (1992).
NA: Not applicable.
[b] There is no closed-form solution but the GA solution matches up to 3 decimals with grid search.
Note: The various problems, the corresponding models and the norms used for the estimation of the parameters are given in the first three columns. The last two columns provide the standard or parametric solutions as provided by SYSTAT and the GA solutions. The numbers in the parentheses are the standard errors of the estimates and the corresponding numbers in the GA column is obtained from 250 bootstrap solutions.

GA is run 50 times and we keep the parameter estimates associated with the best fit. Since the GA is stochastic, it is important to run it multiple times before accepting a solution. The GA and the SYSTAT solutions are given in Table 6. Figs. 3(A)–(C) give the distribution of 50 GA estimates for the three parameters α, β and γ (called a, b, c), respectively. Fig. 3(G) shows the distribution associated with the best fits. The plots of the best, average and the least (worst)-fits against generations are in Fig. 3(H). Note that both the best and the average fits rise steadily and reach steady-state values within the first 30 generations.

We also compute the standard errors of the parameter estimates using a bootstrap. We generate bootstrap samples of the data and run the GA on 250 bootstrap samples. This is used to illustrate the combined power that the GA and a resampling based tool for variability estimation can offer to data analysis. The parametric estimates of the standard errors and those obtained from the bootstrap through the GA are given in Table 6. Fig. 3(D)–(F) give the sample histogram of the distribution of 250 bootstrap samples of the estimates a, b and c.

We reflect here the possible use of GA in data analysis and statistical inference. Figs. 3(A)–(G) give the estimates of the three parameters of interest and the distribution of these estimates and these are essentially theory-free. Thus the GA, with the aid of a bootstrap like tool, may enable us to make statistical inference of any quantity of interest without severe restriction on the models employed or the norm used in their estimation.

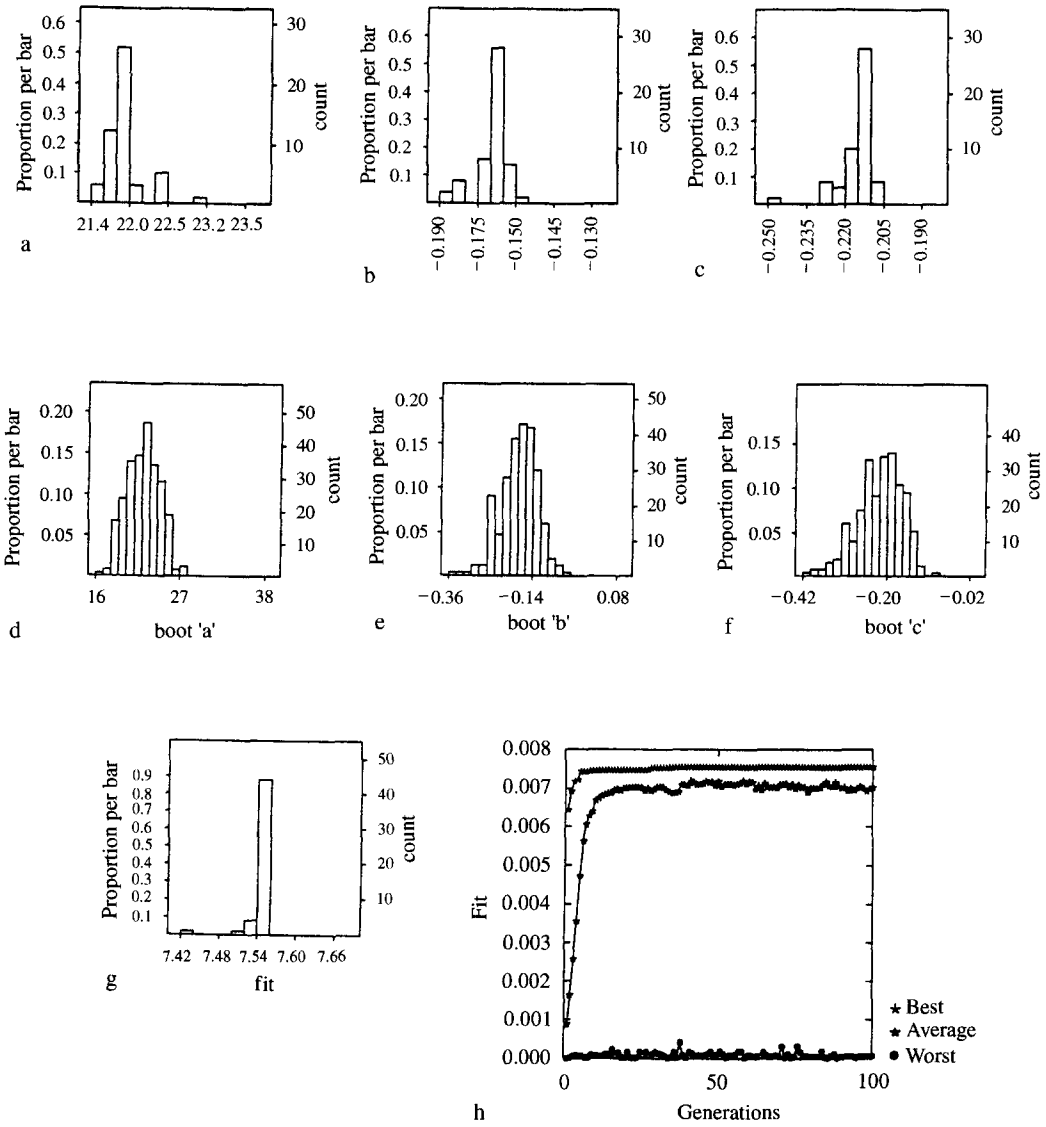

Fig. 3. (a)–(c) The sampling distribution of the three parameters a, b, c for the Angell data for 50 converged solutions (the best achieved within the run) of the GA (each with a population of size 1000 for an evolution of 50 generations). (d)–(f) The bootstrap distributions of the three parameters a, b and c. (g) The distribution of the best-fit solutions for the 50 runs of the GA (i.e. it is the distribution of the best). The plot of the best, average and worst solutions against generation for a given run of the GA is given in (h).

Problem 3 (*Logistic regression, CHD data*). We next estimate the parameters of a logistic linear regression model given by

$$\pi(x) = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}} \tag{2}$$

and the log-likelihood to be maximized is given by
$L(\beta_0, \beta_1) = \sum_{i=1}^{n} \{y_i \ln[\pi(x_i) + (1 - y_i)\ln[1 - \pi(x_i)]\}$, where y_i is 1 or 0 with probability $\pi(x_i)$ and $1 - \pi(x_i)$ given by (2). The data is from Hosmer and Lemeshow (1989) where the explanatory variable is the age (in years) of a random sample of 100 people and the dependent variable is 1 or 0 whether they have coronary heart disease (CHD) or not. The results are in Table 1. The agreement with the SYSTAT solution is good.

Problem 4 (*A robust estimator: Andrews Sine function*). For our final example we estimate the parameters of a linear model by employing a robust criteria (see Hogg, 1979 for an introduction) popular in statistics for the last two decades. This example is illustrative of the universality of the GA as an estimation technique since commonly available commercial software does not produce robust estimates for arbitrary ρ and ψ functions (defined later).

We consider a linear model given by

$$y_i = \sum_{i=0}^{j=3} \beta_j X_{ij} + \varepsilon_i, \quad i = 1, 2, \ldots, n, \tag{3}$$

where $\beta_0 = 1$ and $X_{i0} = 1$ for all i.

The parameters of the model are obtained by minimizing

$$\sum_{i=1}^{i=n} \rho\left(\frac{y_i - \sum_{j=0}^{j=3} \beta_j X_{ij}}{s}\right), \tag{4}$$

where s is a scale measure (see Remark 6). For the function ρ, we use Andrews' (1974) Sine function given by

$$\begin{aligned} \rho(x) &= a^2\left(1 - \operatorname{Cos}\frac{x}{a}\right), && |x| \le \pi a \\ &= 2a^2 && |x| > \pi a. \end{aligned}$$

We assume $a = 1.5$ as suggested by Andrews. The conventional procedure is to solve the normal equations by differentiating (4) with respect to the parameters and setting the derivatives equal to zero and solving the resulting the nonlinear normal equations given by

$$\sum_{i=1}^{n} \psi\left(\frac{y_i - \sum_{j=0}^{j=3} \beta_j X_{ij}}{s}\right) X_{ij}.$$

Here $\psi = \rho'$ where prime denotes first derivative with respect to the parameters β_i. These equations can be solved by one of many routines available for solving nonlinear least-squares problems including the weighted least-squares algorithm. However, it is well known these routines can be very sensitive to starting values (conditions), rounding errors and other convergence problems.

We obtain the GA estimates for a set of data originally analyzed by Daniel and Wood (1971) and re-analyzed by Andrews (1974) using his Sine function and an iterative weighted least-squares technique for optimization. The main feature of this data set is that the least-squares estimates are not stable and greatly influenced by

Table 7
Comparison of GA estimates for a robust model

Methods	b_0	b_1	b_2	b_3	Total error	Fit
LS(all data)[a]	−39.9	0.716	1.29	−0.15	42.01	0.0238
	(11.9)	(0.13)	(0.37)	(0.16)		
LS (reduced data)[b]	−37.7	0.80	0.58	−0.07	20.40	0.049
	(4.73)	(0.07)	(0.17)	(0.06)		
Robust	−37.2	0.82	0.52	−0.07	27.42	0.0365
	[c]	(0.05)	(0.12)	(0.04)		
GA	−37.2266	0.816	0.524	−0.071	26.97	0.0371
	(5.86)[d]	(0.12)	(0.13)	(0.05)		

[a] Least squares.
[b] Data points 1, 3, 4, and 21 deleted.
[c] Note provided by Andrews.
[d] From 250 bootstrap replications.
Note: The linear least squares for the full and reduced data set for Andrews data are given along with the weighted least squares and the GA estimates for the parameters. The estimated standard errors are given in the parentheses.

the presence of four data points (1, 3, 4, and 21). Robust estimates are similar to the least-squares estimates with these influential points deleted. The results are given in Table 7.

The GA estimates are obtained by running the algorithm 50 times for different random starts and collecting the most fit (least total error) solution. The distribution of these fits (not given to save space) are similar to the one given in Fig. 3(G) for Problem 2. The GA standard errors are obtained through 250 bootstrap samples while the standard errors for Andrews results were provided by him. The total error column contains the square errors obtained from model (3) while the last two rows are obtained by minimizing (4). The total error column is converted into a fit criterion by defining fit = 1/total error, since in a GA estimates and the weighted least-squares estimates obtained by Andrews. In conclusions, we see the power of the GA in obtaining numerical estimates that can usually be obtained through special purpose algorithms and the joint use of bootstrap and the GA for fruitful data analysis.

Remark 6. Andrews uses $\underset{i}{\text{med}}[|\varepsilon_i|]$ where as others proposed $\underset{i}{\text{med}}[|\varepsilon_i|/0.6745]$. For our GA estimates we set the scale equal to 1 for the following reason. The scaling is meaningful only when the errors have some constraints on them (such as they sum to zero). Since in the GA, no such constraints are imposed the scale tend to become very large and the parameter estimates become meaningless. However, simply minimizing (3) with $s = 1$ removes all problems and the parameter estimates converge to their true values.

6. GA, combinatiorial optimization, and statistics

In our previous sections, all our optimization problems involved solving for variables whose values are contained in the real line, i.e. $x \in \mathbb{R}$. However, there are large number of problems in optimization where the decision variables are discrete (or mixed). Since problems that take on integer values can be reduced to problems with 0 or 1 binary variables, our discussion is only in terms of problems with 0 or 1 decision variables. A large number of such 0–1 decision problems occur in operations research including the knapsack problem, set covering, set partitioning, set packing and the most well-known traveling salesman problem (TSP) among others. Other applications include routes for packets switched over networks over a large number of nodes, routing of airlines, scheduling, design of VLSI chips from generic chips by burning interconnections, image compression. We will restrict ourselves to a brief discussion of the TSP.

In the TSP we are given inter-city distances between n cities and the problem is to find the optimal sequence that minimizes the total distance for visiting all cites (beginning and ending with the same city). The problem is to minimize a function f for a permutation $\pi(n)$ on n integers (cities). Many versions of TSP exist such as Euclidean and non-Euclidean (that can be accommondated by changing f). The TSP is a NP-complete problem and both heuristics and exact solutions are available, though it is considered to be an unsolved problem in its generality.

From a GA point of view, note how the solution representation has to be different from our previous examples for continuous variables. Now, a candidate solution must represent a legal route and at the same time must produce legal offspring (routes) after undergoing the evolutionary operations. Clever mapping of candidate solutions to bit maps representing tours (Forrest, 1993) are available but it is known that convergence for such algorithms is very slow because they do not preserve schema formation. The difficulties associated with problem representation for TSPs are summarized in Chatterjee et al. (1994) where novel ways of representing the GA for TSP are also presented. Most of the difficulties arise in finding representations that preserve schema formation and at the same time produce legal offspring for defined evolutionary operations. It is also worth pointing out here that the three evolutionary operators that we have used for continuous variables are somewhat arbitrary. Most GA applications, in theory at least, could be developed with only two operations: Selection for survival and crossover for change (We have even questioned the latter in our work on TSP). To overcome difficulties associated with representations, we have introduced the novel idea of an asexual reproduction and have solved very large TSP problems within 2.5% of optimal values. Also, when the routes are represented as candidate solutions, the operations of crossover, mutation, inversion are merely three different ways of changing the order of cities visited. Seen this way, there is no reason not to generalize the evolutionary operations themselves. This is how we have approached the TSP where we use k-cut evolutionary operations representing different legal ways of altering a route.

We have identified a number of areas in which we expect asexual reproduction to be a useful approach.

Problems of discrete choice arise in statistics in many instances. Consider the variable selection problem which can be stated as follows: Given a large number of candidate explanatory variables P, a model form and a criteria to fit the model, the problem is how to choose p ($\ll P$) variables that yield the best fit. The variable selection problem has applications in linear and nonlinear (including logistic) regression, discriminant and principal components analysis, nonparametric regression and other related models such as the popular computer intensive classification and regression trees (CART) of Breiman et al. (1984). Here we are not interested in a discussion on the criteria of fit or other statistical issues associated with the delicate problem of variable selection. We are simply addressing the problem of choosing q variables, when other statistical issues are resolved. Exhaustive enumeration is never a practical alternative and the various stepwise and stage wise procedures necessarily produce local optima. In the currently available methods the final solutions are local optima. The regression trees as developed in the CART methodologies, necessarily depend on the order of the entry of the variables. A GA has much potential in addressing the sort of problems that are computationally explosive and intractable.

The variable selection problem is not of theoretical interest only. For example, consider a potentially useful application in epidemiology. A researcher is faced with a large number of candidate or carrier variables (clinical, psychological, dietary, soci-economic, ethnic, etc.) and a known condition ($y = 1$ or 0) where the data may be soft and incomplete. How are the q best explanatory variables to be identified? The problem of locating homologies in the human genome is also a very important discrete choice problem.

Development of symbolic regression systems is also a desirable goal, freeing the researcher from predetermining the functional from of a regression. In symbolic regression the algorithm not only finds the parameter estimates but also functional specifications chosen from a finite number of functions allowed; typically, Log, Exp. Sin, Cos, power transform of the Box–Cox type or other operators as the application might demand are allowed. The principle remains the same, i.e. of selecting functions and parameters to maximize a meaningful fitness criteria. This is one area in which data analysis can be supplemented with techniques from artificial intelligence (see Koza, 1992; Chatterjee and Laudato, 1994).

7. Some technical issues of GA

There are three aspects of the mathematics of GA that have been discussed in the literature so far. First, the so-called fundamental theorem of GA (FTGA) which states that for a population of solutions to converge towards the global optima, small favorable schemas possessing above average fitness must be present in the

population and must grow at an exponential rate (the obvious converse is that unfavorable schemas must die off in an exponential manner). Notice that FTGA is not only a theorem for a GA but also defines it. For our toy problem we observe the following:

The optimal solution for our toy problem is 0 1 0 1 0 1. We study a particular 3-alphabet schema given by * * 0 1 0 * for 20 generations with a population of size 20. We see that solutions that do not contain the schema die off or the solution does not converge. On the other hand, solutions that converge comes to be dominated with population members possessing the favorable schema. This is the natural parallelism implicit in a GA. Since processing of a candidate solution from a population of candidate solutions can proceed independently, GAs are ideal for explicit parallel processing. In general, we observe about 75% of the members of a population will contain any given short favorable schema when the population as a whole has converged. This is our observation of algorithms we have implemented, both sexual and asexual.

Second, schema-based searches in a GA are equivalent to random searches along hyperplanes (hence they are sometimes called hyperplane-based sampling). The analytical method of Walsh transforms (Forrest and Mitchell, 1991) can be used to analyze the performance of a GA. Walsh polynomials (defined as sums of Walsh functions) are suitable for defining any real valued function on bit strings akin to Fourier analysis for approximating arbitrary functions to any degree of precision (through sines and cosine of progressively higher frequencies). The Walsh-schema transform allows useful analysis of expected performance of a GA and can also be used to predict the suitability of a problem for a GA implementation. However, if the coding scheme is not binary, other methods of analysis have to be devised.

Finally, how does the solution time and solution accuracy depend on the problem size and model (norm) complexity? Though we have not studied the subject very broadly, a small amount of experimentation with the GA for problems (coded on a binary bit string) of various size and complexity reveals several things. All things equal, computation time is strictly proportional to the size of the population and the number of generations a problem is run. For a given accuracy most problems will yield a (roughly) hyperbolic relationship between number of generations and the size of the populations. In other words, one can trade a certain number of generations and the size of the populations. In other words, one can trade a certain number of generations for an increase in population size and vice versa, but there are limits.

As the number of parameters estimated increase, the solution accuracy will decrease. Similarly, for a fixed population size, number of generations and number of parameters, the solution accuracy decreases for increasing complexity (nonlinearity) of the model. Our experience has been that we pay with more generations for model complexity, while for model size (number of parameters) we pay with a larger population size. More research needs to be done for a formal understanding of how problem complexity, problem size and number of observations (size of the data sets) enter in to the computation time, accuracy and population size of a GA.

8. Discussion and conclusions

We undertook this survey to bring to the statistical community some of the developments in GAs that may be helpful in statistical modeling. GAs are general purpose stochastic search algorithms, related to such algorithms as simulated annealing. Such algorithms are collectively known as homotopy methods. The GAs are highly influenced by biological and evolutionary models and consequently are population-based methods.

We have examined the genetic algorithm as a method for statistical parameter estimation for several models and norms and find that the GA solutions are quite comparable to the ones obtained through the conventional methods. Combined with a bootstrap-like tool for estimating standard errors, we have a data analysis strategy for which there need not be any mathematical restrictions on the model or on a particular norm. We have also used the GA for estimating the weights of strengths of connections among synapses in an artificial neural network (Cheng and Titterington, 1994) which is a highly nonlinear problem. Our preliminary analysis indicates that the GA estimates of these weights is comparable (and sometimes superior) to the estimates obtained by the conventional method of back propagation (Chatterjee and Laudato, 1994). The GAs are general purpose optimization tools designed to search irregular, poorly characterized function spaces and that are easily implemented on parallel computers.

In this paper we have examined only unconstrained optimization problems in continuous variables. Constrained optimzation problems can be handled (at least in theory) by incorporating the constraints in the fit function with penalty for violated constraints. As discussed, discrete choice problems can also be harnessed thorough clever mapping of candidate solutions into computer representable forms that allow legal descendants through meaningful evolutionary operations. Many statistical procedures of a discrete nature (such as variable selection, order of entry in CART, clustering and others) can be attacked, we believe, more efficiently through the methods of genetic algorithms.

Acknowledgements

We are thankful for the helpful criticism offered by two referees and an associate editor. The editor's help is also acknowledged.

References

Andrews, D.E., A robust method for multiple linear regression, *Technometrics*, **16** (1974) 523–531.

Breiman, L., J.H. Friedman, R.C. Olshen and C.J. Stone, *Classification and regression trees*, (Wadsworth International Group, Belmont, CA, 1984).

Chatterjee, S. and M. Laudato, Genetic algorithms in statistics: procedures and applications, submitted for publications in: *Commun. Statist.* (1994).

Chatterjee, S. and M. Laudato, Statistical modeling using neural network, preprint (1994).
Chatterjee, S., C. Carrera and L. Lynch, Genetic algorithms and the traveling salesman problems, forthcoming in: *Eur. J. Oper. Res.* (1994).
Cheng B. and D.M. Titterington, Neural networks: a review from a statistical perspective (with discussion), *Statist. Sci.*, **9** (1994) 2–54.
Daniel, C. and F.S. Wood, *Fitting equations of data* (Wiley, New York, 1971).
Forrest, S. and M. Mitchell, *The performance of genetic algorithms on Walsh polynomials: some anomalous results and their explanation*, in: *Proc. 4th Internat. Conf. on Genetic Algorithms*, R.K. Belew and L.B. Booker (Eds.), Morgan Kaufmann Publishers, San Mateo, CA.
Forest, S. *Genetic algorithms*: principles of natural selection applied to computation, *Science*, **261** (1993) 872–878.
Fox, J., *Linear statistical models and related methods* (Wiley, New York, 1984).
Gell-Mann, M., *The quark and the Jaguar: adventures in the simplex and the complex* (W.F. Freeman and Company, New York, 1994).
Goldberg, D.E., *Genetic algorithms in search, optimization and machine learning* (Addison-Wesley, Reading, MA, 1989).
Hogg, R.V., An introduction to robust estimation, in: R.B. Launer and G.N. Wilkinson (Eds.), I *Robustness in Statistics*, (p. 1–17). Academic Press, New York, (1979).
Holland, J.M., *Adaptation in natural and artificial systems* (The University of Michigan Press, Ann Arbor, 1975).
Holland, J.M., *Adaptation in natural and artificial systems* (MIT Press, Cambridge, MA, 1992a).
Holland, J.M., Genetic algorithms, *Scientific American*, July, 66–72 (1992b).
Holland, J.M., I.J. Holyoak, R.E. Nisbet and P.R. Thagard, *Induction* (The MIT Press, Cambridge, MA, 1986).
Hosmer, D.W. and S. Lemeshow, *Applied logistic regression* (Wesley, New York, 1989).
Koza, J.R., *Genetic programming* (The MIT Press, Cambridge, MA, 1992).
SYSTAT. Inc. Version 5.2, Evanston, IL, (1992).
Wolfram, S., *Mathematica*, 2nd edn. (Addison-Wesley Reading, MA, 1993).