Comparing Stochastic Optimization Methods for Variable Selection in Binary Outcome Prediction, With Application to Health Policy

Dimitris FOUSKAKIS and David DRAPER

Traditional variable-selection strategies in generalized linear models (GLMs) seek to optimize a measure of predictive accuracy without regard for the cost of data collection. When the purpose of such model building is the creation of predictive scales to be used in future studies with constrained budgets, the standard approach may not be optimal. We propose a Bayesian decision-theoretic framework for variable selection in binary-outcome GLMs where the budget for data collection is constrained and potential predictors may vary considerably in cost. The method is illustrated using data from a large study of quality of hospital care in the U.S. in the 1980s. Especially when the number of available predictors p is large, it is important to use an appropriate technique for optimization (e.g., in an application presented here where $p = 83$, the space over which we search has $2^{83} \doteq 10^{25}$ elements, which is too large to explore using brute force enumeration). Specifically, we investigate simulated annealing (SA), genetic algorithms (GAs), and the tabu search (TS) method used in operations research, and we develop a context-specific version of SA, improved simulated annealing (ISA), that performs better than its generic counterpart. When p was modest in our study, we found that GAs performed relatively poorly for all but the very best user-defined input configurations, generic SA did not perform well, and TS had excellent median performance and was much less sensitive to suboptimal choice of user-defined inputs. When p was large in our study, the best versions of GA and ISA outperformed TS and generic SA. Our results are presented in the context of health policy but can apply to other quality assessment settings with dichotomous outcomes as well.

KEY WORDS: Bayesian decision theory; Cross-validation; Genetic algorithm; Input–output analysis; Logistic regression; Maximization of expected utility; Monte Carlo methods; Prediction; Quality of health care; Sickness at hospital admission; Simulated annealing; Tabu search; Variable selection.

1. INTRODUCTION

1.1 Applied Context

Recent years have seen increased interest in measuring the quality with which institutions such as hospitals carry out their societal mandates. Quality assessment for hospitals generally is based on some or all of three main ingredients (e.g., Donabedian and Bashshur 2002): *process*, what goes on inside the institutions; *outputs*, outcome measures by which they may be evaluated; and *inputs*, such as relevant patient characteristics, because it is unfair to compare hospitals on their outputs without taking into account relevant differences in the patient cohorts that they admit. Typically, process is much more expensive to measure than inputs and outputs (e.g., Kahn et al. 1990a). From the mid-1980s to the present, this fact has encouraged a strategy of quality measurement for hospitals in the U.S. and U.K.—the *league table* or *input–output* (IO) approach (e.g., Draper 1995; Goldstein and Spiegelhalter 1996)—based solely on inputs and outputs. This approach also is known as *provider profiling* (see, e.g., Normand, Glickman, and Gatsonis 1997). With this strategy, each institution is treated as a "black box," with no attempt made to explicitly measure the processes going on inside the box; instead, the contents of the box are inferred indirectly, by evaluating the institution's outputs after controlling for its inputs.

The most popular way to carry out IO quality assessment in hospitals (e.g., Kahn et al. 1988; Draper 1995) is to compare the observed mortality rates at each of a number of hospitals with their expected rates, given the sickness of their patients on admission. The idea is that hospitals with large excess mortality in this comparison might be good candidates for a more detailed (and more costly) process evaluation of their quality of care. Investigation into the potential effectiveness of this method of screening for substandard hospitals dates back in the U.S. to the late 1980s (e.g., Dubois, Rogers, Moxley, Draper, and Brook 1987; Daley et al. 1988; Jencks et al. 1988), and the approach has been studied for some time now in the U.K. as well (e.g., Goldstein and Spiegelhalter 1996). Applications of a similar approach to quality measurement in education also have been described (see, e.g., Draper and Gittoes 2004), and indeed, the quality of many complex systems in a wide variety of fields in principle could be assessed by comparing outputs after adjustment for inputs (see Sec. 4 for examples in fields other than health and education).

If IO analysis were to be widely used as a screening technique for initial measurement of hospital quality of care, it would be far too expensive—given the governmental budget allocations available for, e.g., Medicare quality of care assessment (U.S. Department of Health and Human Services 2008)—to evaluate the sickness at admission for every Medicare patient in every U.S. hospital in any given year. An approach based on sampling hospitals and then sampling patients within the chosen hospitals would be necessary. Whatever the optimal numbers of hospitals and patients per hospital might turn out to be in such a sampling plan, clearly, in an environment of constrained budgets, the *cost-effective* measurement of patient sickness at admission would be crucial. Progress is being made in the U.S. (see, e.g., Centers for Medicare & Medicaid Services 2008 for details on Medicare's plans to compile a uniform clinical data set) and elsewhere on routine (automated) data collection of clinically richer sets of process and sickness variables

Dimitris Fouskakis is Lecturer, Department of Mathematics, School of Applied Mathematicial and Physical Sciences, National Technical University of Athens, Zografou Campus, Athens 15780, Greece (E-mail: *fouskakis@math.ntua.gr*). David Draper is Professor, Department of Applied Mathematics and Statistics, Baskin School of Engineering, University of California, Santa Cruz, CA 95064 (E-mail: *draper@ams.ucsc.edu*). Support for the research described here was provided in part by the European Commission and the University of Bath (U.K.). The authors are grateful to Katherine Kahn for providing data from the RAND DRG Quality of Care study, and to Merrilee Hurn, Chris Jennison, Sylvia Richardson, and the editor, associate editor, and especially a referee for comments and references that improved the manuscript.

Journal of the American Statistical Association
December 2008, Vol. 103, No. 484, Applications and Case Studies
DOI 10.1198/016214508000001048

for hospital patients than those previously available from administrative data bases, but for at least the next decade, the cost-effective collection of primary data will continue to be relevant to the design of quality of care studies in health policy. (See, e.g., NDNQI 2008 and California Nursing Outcomes Coalition 2008 for current examples in the field of nursing quality assessment, where extensive nonautomated primary data collection is both ongoing and planned.)

How should sickness at admission be measured? In a major study of quality of care in U.S. hospitals in the 1980s conducted by the RAND Corporation (Kahn et al. 1990b; Draper et al. 1990), a large number of items (on the order of 100 per disease) that expert physician judgment had indicated were relevant to admission sickness were collected (Keeler et al. 1990) from all sampled patient medical records. Some of the items (e.g., age and initial blood pressure) were common to all diseases, but most were disease-specific, such as a measure of shortness of breath for pneumonia and a variable measuring the severity of the cerebrovascular accident for stroke. Logistic regression models were then constructed (one for each disease), taking death at 30 days from admission as the outcome and the admission sickness indicators as predictors. This created a large variable selection problem, because there was a fair amount of redundancy in the set of predictor variables for each disease. Standard classical backward-selection methods from the model with all available predictors (e.g., Hosmer and Lemeshow 2000) were used in the RAND study to choose a parsimonious subset of sickness indicators that still had reasonably good predictive accuracy.

As an example of the results, Table 1 lists the 14 variables chosen in this manner for inclusion in the RAND sickness-at-admission scale for pneumonia from among an initial set of 83 variables. The total APACHE II score (Knaus, Draper, Wagner, and Zimmerman 1985) is a 36-point scale developed before the RAND study that was designed to measure the sickness of patients in intensive care units (see, e.g., Open Clinical 2008 for a recent update to this scale). Five of the variables in the scale given in Table 1 are dichotomies; four are quantitative measures of a single sign or symptom (e.g., initial temperature) or bodily function (e.g., blood urea nitrogen level, an indicator of kidney function), four are scales built up from other variables (e.g., the score quantifying the extent of the patient's congestive heart failure (if any), as indicated by the admission chest X-ray), and one (age) is demographic.

There are various ways to measure the predictive accuracy of this scale. The predicted death probabilities $\hat{p}$ from the logistic regression of death within 30 days of admission on the 14 variables in Table 1, which essentially form the RAND admission sickness scale, ranged from .004 to .974 in the RAND data for a disease with overall death rate of 15.8%. The *pseudo-*R^2 of the scale (e.g., Judge, Hill, Griffiths, Lütkepohl, and Lee 1988)—the difference between the mean $\hat{p}$ values for the patients who died and for those who lived—was .289. The sensitivity and specificity of the scale as a "screening test" for death, using a rule of the form {predict death if $\hat{p} > p^*$} (with p^* chosen to create equal numbers of false positives and negatives), were 54.1% and 91.4%, leading to an overall error rate of 14.5%. Other scales, using more or fewer of the available sickness indicators for pneumonia, are possible and these would have better or worse predictive performance than the RAND scale. However, any such summaries of the value of scales of this type fail to account for differences in the data collection costs of the variables on which the scales are based. The second column in Table 1 gives the estimated monetary costs associated with each variable in the RAND scale (based on data on record abstraction times collected during the RAND study and using an hourly abstraction fee that was realistic for the late 1980s), and it can be seen that they vary from smallest to largest by a factor of 20 to 1. Constructing a cost-effective admission sickness scale in the presence of such a wide variation in data collection costs requires weighing predictive accuracy against such costs. When this is done, a different form of variable selection in generalized linear models (GLMs) will arise than that on which the usual methods (either classical or Bayesian) are based, in which variables are retained in the scale only if they predict sufficiently well given how much they cost to collect. In this article

Table 1. The RAND admission sickness scale for pneumonia ($p = 14$ variables), with the marginal data collection costs per patient for each variable and the simple correlations with 30-day death

Variable	Cost (U.S.$)	Correlation with 30-day death	Cost-effective?
Total APACHE II score (36-point scale)	3.33	.39	
Age	.50	.17	*
Systolic blood pressure score (two-point scale)	.17	.29	**
Chest X-ray congestive heart failure score (three-point scale)	.83	.10	
Blood urea nitrogen	.50	.32	**
APACHE II coma score (three-point scale)	.83	.35	**
Serum albumin score (three-point scale)	.50	.20	*
Shortness of breath (yes, no)	.33	.13	**
Respiratory distress (yes, no)	.33	.18	*
Septic complications (yes, no)	1.00	.06	
Prior respiratory failure (yes, no)	.67	.08	
Recently hospitalized (yes, no)	.67	.14	
Ambulatory score (three-point scale)	.83	.22	
Initial temperature	.17	−.16	*

NOTE: Costs were measured in approximate minutes of record abstraction time and converted to money using an abstraction fee of U.S.$20/hour. The last column, explained in Section 1.3, identifies variables in models that yield near-optimal expected utility.

we detail one natural way to perform such cost-effective variable selection. Brown, Fearn, and Vannucci (1999) described an application of decision theory to variable selection in multivariate regression motivated by somewhat similar cost-benefit considerations in a different setting. Draper and Fouskakis (2000) offered some pilot study results that served as a prelude to the larger study on which we report here. Fouskakis, Ntzoufras, and Draper (2009a,b) presented two other approaches to solving the problem addressed in this article, based on a cost-adjusted version of the Bayes information criterion and a maximization of predictive accuracy subject to a bound on cost.

1.2 Problem Formulation

Following Fouskakis (2001), suppose that the 30-day mortality outcome y_i and data on p sickness indicators $(x_{i1}, \ldots, x_{ip})$ were collected on n individuals sampled randomly from a population $\mathcal{P}$ of patients with a given disease, and that the goal is to predict the death outcome for n^* new patients who in the future will be sampled randomly from $\mathcal{P}$ on the basis of some or all of the predictors $x_{\cdot j}$, when the marginal costs of data collection per patient $c_1, \ldots, c_p$ for the $x_{\cdot j}$ vary considerably. Our problem can be stated as: What is the best subset of the $x_{\cdot j}$ if a fixed amount of money is available for this task and the reward is based on the quality of the predictions? To solve this problem, we maximize expected utility (e.g., DeGroot 1970; Berger 1985), defined in a way that trades off predictive accuracy against data collection costs. The data on which we illustrate this method here consist of a representative sample of $n = 2{,}532$ elderly American patients hospitalized in the period 1980–1986 with pneumonia, taken from the RAND study mentioned previously. Because data on future patients are not available, we use a *cross-validation* approach (e.g., Gelfand, Dey, and Chang 1992; Hadorn, Draper, Rogers, Keeler, and Brook 1992) in which a random subset of n_M observations is drawn for creation of the mortality predictions (the *modeling* subsample) and the quality of those predictions is assessed on the remaining $n_V = (n - n_M)$ observations (the *validation* subsample, which serves as a proxy for future patients).

In the approach presented here, utility is quantified in monetary terms, so that the data collection utility is simply the negative of the total amount of money required to gather data on the specified predictor subset. Letting $I_j = 1$ if $x_{\cdot j}$ is included in a given model and 0 otherwise, the data collection utility associated with subset $I = (I_1, \ldots, I_p)$ for patients in the validation subsample is

$$U_D(I) = -n_V \sum_{j=1}^{p} c_j I_j; \tag{1}$$

the second column in Table 1 gives examples of the marginal costs c_j.

To measure the accuracy of a model's predictions, a metric is needed that quantifies the discrepancy between the actual and predicted values, and in this problem the metric must come out in monetary terms on a scale comparable to that used with the data collection utility. In the setting of this example, the actual values y_i are binary death indicators, and the predicted values $\hat{p}_i$, based on statistical modeling, take the form of estimated death probabilities. Our approach to the comparison of actual and predicted values involves dichotomizing the $\hat{p}_i$ with respect to a cutoff, to mimic the decision-making reality that actions taken on the basis of IO quality assessment will have an all-or-nothing character at the hospital level; for example, regulators must decide to either subject or not subject a given hospital to a more detailed, more expensive quality audit based on process criteria (see, e.g., Kahn et al. 1990a). Other, continuous approaches to the quantification of predictive utility are possible; for example, Bernardo and Smith (1994) used a log scoring method, and Lindley (1968) used squared-error loss to measure predictive accuracy in a less problem-specific framework than that presented here. Continuous utility functions in the setting of this article are the subject of ongoing investigation.

In the first step of our approach, given a particular predictor subset I, we fit a logistic regression model to the modeling subsample M and apply this model to the validation subsample V to create predicted death probabilities $\hat{p}_i^I$. In more detail, letting $y_i = 1$ if patient i dies within 30 days of admission and 0 otherwise, and taking $x_{i1}, \ldots, x_{ik}$ to be the k sickness predictors for this patient under model I, the usual sampling model underlying logistic regression in this case is

$$\begin{aligned} (y_i | p_i^I) &\overset{\text{indep}}{\sim} \text{Bernoulli}(p_i^I), \\ \log\left(\frac{p_i^I}{1 - p_i^I}\right) &= \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik}. \end{aligned} \tag{2}$$

We use maximum likelihood to fit this model (as a computationally efficient approximation to Bayesian fitting with relatively diffuse priors), obtaining a vector $\hat{\beta}$ of estimated logistic regression coefficients, from which the predicted death probabilities for the patients in subsample V are, as usual, given by

$$\hat{p}_i^I = \left[1 + \exp\left(-\sum_{j=0}^{k} \hat{\beta}_j x_{ij}\right)\right]^{-1}, \tag{3}$$

where $x_{i0} = 1$. (Here $\hat{p}_i^I$ may be considered the sickness score for patient i under model I.)

In the second step of our approach, we classify patient i in the validation subsample as predicted dead or alive according to whether $\hat{p}_i^I$ exceeds or falls short of a cutoff p^*, which is chosen by searching on a discrete grid from .01 to .99 by steps of .01 to maximize the predictive accuracy of model I. We then cross-tabulate actual versus predicted death status in a 2×2 contingency table, rewarding and penalizing model I according to the numbers of patients in the validation sample who fall into the cells of the right side of Table 2. The left side of the table records the rewards and penalties in U.S. dollars. The predictive utility of model I is then

$$U_P(I) = \sum_{l=1}^{2} \sum_{m=1}^{2} C_{lm}\, n_{lm}. \tag{4}$$

To elicit the utility values C_{lm}, we reason as follows:

1. Clearly, C_{11} (the reward for correctly predicting death at 30 days) and C_{22} (the reward for correctly predicting living at 30 days) should be positive, and C_{12} (the penalty for a false prediction of living) and C_{21} (the penalty for a false prediction of death) should be negative.

Table 2. Cross-tabulation of actual versus predicted death status: Monetary rewards and penalties for correct and incorrect predictions and frequencies in the 2 × 2 tabulation

		Rewards and penalties		Counts	
		Predicted		Predicted	
		Died	Lived	Died	Lived
Actual	Died	C_{11}	C_{12}	n_{11}	n_{12}
	Lived	C_{21}	C_{22}	n_{21}	n_{22}

2. Because it is easier to correctly predict that a person lives than dies with these data (the overall pneumonia 30-day death rate in our sample was 16%, so a prediction that every patient lives would be correct about 84% of the time), it is natural to specify that $C_{11} > C_{22}$.

3. Because it is arguably worse to label a "bad" hospital as "good" than the other way around, one should take $|C_{12}| > |C_{21}|$, and furthermore it is natural that the magnitudes of the penalties should exceed those of the rewards.

4. We completed the utility specification by eliciting information from health experts in the U.S. and U.K., first to anchor C_{21} to the cost of subjecting a "good" hospital to an unnecessary process audit and then to obtain ratios relating the other C_{lm} to C_{21}. Details on this final step are given in Appendix A. (This and Apps. B and C are available online at *www.amstat.org/publications/jasa/supplemental_materials/*.)

With the C_{lm} in hand, the overall expected utility function to be maximized over I is then simply

$$E[U(I)] = E[U_D(I) + U_P(I)], \tag{5}$$

where this expectation is over all possible cross-validation splits of the data. The number of possible cross-validation splits is far too large to evaluate the expectation in (5) directly; in practice, we use Monte Carlo methods to evaluate it, averaging over N random modeling and validation splits.

1.3 The Need for Stochastic Optimization

Even with the use of Monte Carlo sampling to evaluate (5), when the number p of available predictors is of realistic size (on the order of 50–100 or larger), the space of all 2^p possible subsets is too large to permit a brute force maximization of expected utility by an exhaustive examination of all possible subsets. This motivates the need for global optimization methods, for example, stochastic optimization techniques such as simulated annealing (e.g., Kirkpatrick, Gelatt, and Vecchi 1983), genetic algorithms (e.g., Holland 1975), and tabu search (e.g., Glover 1989), which are a main focus of the rest of the article.

Because the best way to compare optimization methods on the quality of the configurations (in this case, subsets of predictors) that they find is to know the truth in advance, we created a testbed for this comparison by evaluating (5) using $N = 500$ modeling/validation splits for all $2^{14} = 16{,}384$ possible subsets of the $p = 14$ variables in the RAND pneumonia sickness scale in Table 1. This choice of N was large enough to yield a Monte Carlo standard error for each expected utility estimate of only about U.S.$0.05, which is small enough to reliably identify the good models. Figure 1 presents parallel boxplots of all 16,384 estimated expected utility values as a function of the number of variables k, included in the scale. (In this problem, a configuration is a binary vector of length p.) Each utility evaluation took .4 seconds at 400 Unix MHz, meaning that full enumeration of all 16,384 subsets with $N = 500$ took

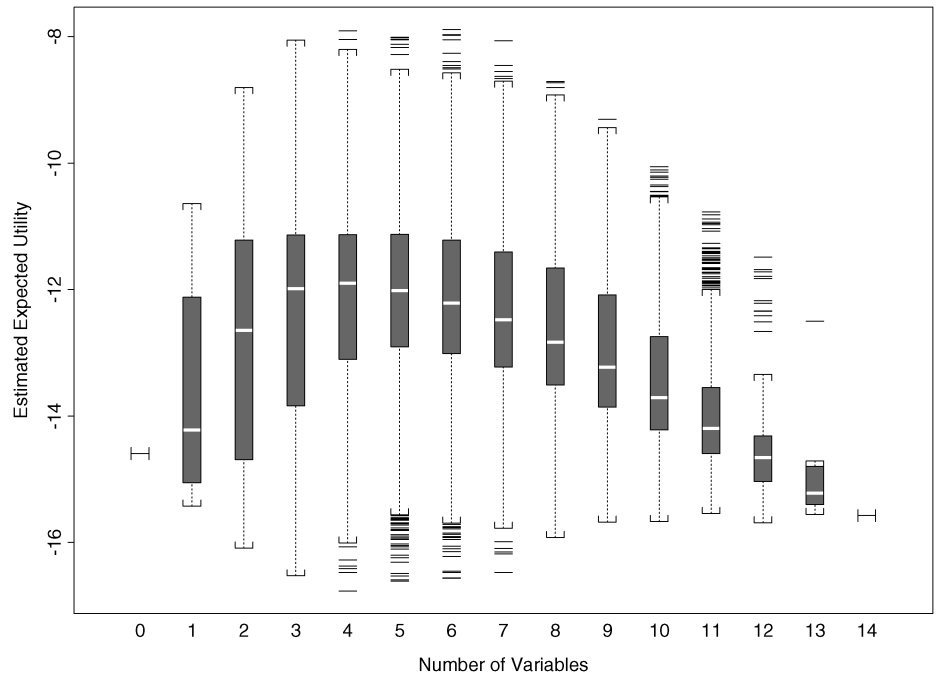

Figure 1. Parallel boxplots of estimated expected utility as a function of number of predictors retained, examining all $2^{14} = 16{,}384$ possible subsets of the $p = 14$ variables in the RAND pneumonia sickness scale with two-thirds of the data in the modeling subsample (from Fouskakis 2001).

38 days of CPU time; in other words, a full enumeration of all possible subsets in our problem (with a realistically large value of N) is already a lengthy undertaking even with only 14 variables among which to choose. The figure shows, as is intuitively reasonable, that as k increases from 0, the models get better on average up to a maximum or near maximum of $k = 3$–7 and then get steadily worse on average up to $k = 14$.

The 20 best models included the same three variables 18 or more times out of 20, and never included 6 other variables; the 5 best models were minor variations on each other, and included 4–6 variables. The four variables in the model that achieved the global maximum expected utility are identified with two asterisks in the last column of Table 1, and the four additional variables that appeared most often (along with the ∗∗ variables) among the best models are highlighted in that column with one asterisk. The best models save almost $8 per patient over the full 14-variable model; this would amount to significant savings were the league-table assessment method applied widely. Note that from Table 1, several of the variables with the strongest marginal predictive power for 30-day mortality (e.g., total APACHE II score and ambulatory score) do not appear in the best models, because their costs are too high in relation to their predictive usefulness (given the other available variables). Of course, we have the luxury of knowing the correct answer here only because the space of possible configurations was relatively small. Searching through this space with all $p = 83$ available sickness variables, where the total number of possible subsets is $2^{83} \doteq 9.7 \cdot 10^{24}$, is out of the question. But if stochastic optimization methods—all of which require the user to make a series of choices to tune the algorithms—are to be used to maximize the expected utility (5), it is not clear which method is best, and it is even less clear what algorithmic inputs should be used to get superior (or even adequate) performance. (The literature is remarkably silent on this point.)

The article is organized as follows. In Section 2 we briefly review each of the three stochastic optimization methods that we compare. In Section 3 we present our simulation results, focusing mainly on general purpose implementations of the optimization techniques that are not specifically tailored to the decision-theoretic problem of Section 1 (to see how well generic versions of the algorithms perform). First, we examine the moderate-dimension case of $p = 14$, where we also study the gain in performance obtained by equipping one of the methods, simulated annealing, with additional "knowledge" of the decision-theory problem, and then we present findings in the much larger space corresponding to $p = 83$. In Section 4 we conclude with a summary and discussion of the implications of our findings for variable selection, optimization, and quality assessment.

2. STOCHASTIC OPTIMIZATION METHODS

Here we present only brief sketches of the three optimization methods that we examine; Appendix B, available online (URL provided in Sec. 1.2), provides a more discursive summary and a comprehensive set of references (App. C, discussed below, also is available online as part of the same pdf document). See Fouskakis and Draper (2002) for a recent review paper contrasting the methods examined here and other stochastic optimization techniques.

A general issue arising in this problem, common to all approaches to maximizing expected utility, is as follows. When different optimization methods are compared to see which is the best at finding the global maximum of (5), to make the comparison fair they all must be given the same amount of CPU time with which to perform the search, and then the choice of N [the number of modeling/validation splits on which the Monte Carlo approximation of (5) is based] becomes one of the optimization variables. If N is small, then a given method can visit a large number of models but will obtain a noisy estimate of how good those models are. In contrast, if N is large, then the estimate of a model's quality will be more precise, but the number of models that can be visited given the time constraint will be much smaller. We specify our strategy for choosing N in Section 3.1.

2.1 Simulated Annealing

Simulated annealing (SA; e.g., Kirkpatrick et al. 1983) is a stochastic local search technique to approximate the maximum of an objective function $f : S \to \Re$ over a finite set S. It is an iterative method that randomly chooses elements y from a neighborhood, $N(x)$, of the present solution x. The candidate y is either accepted as the new solution or rejected. It may be accepted with a positive probability even if $f(y) < f(x)$. Thus the search process can "climb downhill" to escape from local maxima.

The long-run behavior of the search process depends critically on $a(x, T, y)$, the probability of accepting a candidate y given a present solution x; in turn, $a(x, T, y)$ is controlled by the temperature parameter, T (by analogy to a physical cooling process). To make the iterative search an inhomogeneous Markov chain, the temperature values typically are chosen independently of the process as a fixed sequence T_n, the temperature schedule. Usually, the Metropolis acceptance probability is used; that is, for $T > 0$,

$$a(x, T, y) := \begin{cases} 1 & \text{if } f(y) \geq f(x) \\ \exp\left[\dfrac{f(y) - f(x)}{T}\right] & \text{if } f(y) < f(x). \end{cases} \quad (6)$$

The rate at which T decreases as the number of iterations increases—the cooling schedule—and the choices of initial and final temperatures are often jointly crucial to the performance of SA. The geometric cooling schedule $T_i = T_0(\frac{T_f}{T_0})^{(i-1)/(M-1)}$ (where i is the iteration number, M is the target number of iterations, and T_0 and T_f are the initial and final temperatures) is a frequent default specification; other schedules examined in the literature include straight-line, reciprocal, and logarithmic choices (see online App. B for details).

The candidate moves are chosen according to a generating probability $G(x, \cdot)$, which often is the uniform distribution on the neighborhood $N(x)$. The simplest possible neighborhood structure in the problem that we study herein defines two models as neighbors if they differ by the inclusion or exclusion of a single variable; the moves that generate this neighborhood are known as *one-bit flips*. The final configuration at the end of the SA run can be reported as the approximate solution, or the K best solutions found so far (for some reasonable $K \geq 1$) can be maintained throughout the run and reported. (This requires some CPU and memory resources to implement but is often worthwhile.)

2.2 Genetic Algorithms

The *genetic algorithm* (GA), introduced by Holland (1975), has become a popular method for solving large optimization problems with multiple local optima. The phrase "genetic algorithm" is more aptly used in the plural, because of the wealth of variations on the basic idea that has grown up since the 1970s; here we use "GA" to represent any of these variations.

GA got its name from the process of drawing an analogy between components of the configuration vector x and the genetic structure of a chromosome. As before, the goal is to maximize a function, $f(x)$, of the vector $x = (x_1, x_2, \ldots, x_p)$, where here each x_i is binary. The basic GA starts by randomly generating an even number n of binary strings of length p to form an initial population. A positive fitness g then is calculated as a monotone increasing function of f for each string in the current generation, and n parents (configurations) for the next generation are selected, with replacement, with the probability p_j of choosing string j in the current population proportional to its fitness g_j. The new parents are considered in pairs, and for each pair a crossover operation is performed with a preselected probability p_c. If crossover occurs, then an integer k is generated uniformly at random between 1 and $(p - 1)$ (inclusive), and the last $(p - k)$ elements of each parent are exchanged to create two new configurations (offspring). If crossover does not occur, then the parents are copied unaltered into two new strings.

After the crossover operation, mutation is performed with a preselected probability p_m. If mutation occurs, then the value at each string position is switched from 0 to 1 or vice versa, independent at each element of each string. The algorithm is allowed to continue for a specified number of generations. On termination, the string in the final population with the highest value of f can be returned as the solution of the optimization problem. But because a good solution may be lost during the algorithm, a more efficient strategy is to note the best (or, even better, the K best) configurations seen at any stage (for some reasonable K) and return these as the solution. The population size n, parameters p_c and p_m, and fitness function g must be specified before GA is applied. It is often reasonable to take g equal to f, but in some problems a more careful choice may be required (see, e.g., Fouskakis 2001, for a survey of other ideas). Note that GA proposes moves away from the configurations currently under examination using a "neighborhood structure" that is completely different from the approach used in SA or tabu search (described later).

The foregoing is a description of the basic GA algorithm as it was initially proposed and used up through the mid-1980s. More recent variations include elitist strategies (see Sec. 3.1.3 for details) and more elaborate crossover rules, of which the two most important are as follows. In uniform crossover, at each position (with probability .5, say), each bit is chosen randomly from either of the two parent strings, repeating if two offspring are desired. A modified version of uniform crossover is the method used by the CHC Adaptive Search Algorithm (Eshelman 1991), which we call *highly uniform crossover*. This version crosses over half (or the nearest integer to half) of the nonmatching bits, where the elements to be exchanged are chosen at random without replacement. This operator always guarantees that the offspring are the maximum Hamming distance (a count of the number of discrepant bits) from their two parents.

2.3 Tabu Search

Tabu search (TS), substantially better known to investigators in operations research than in statistics, is a neighborhood-based "higher-level" heuristic procedure for solving optimization problems, designed (possibly in combination with other methods) to escape the trap of local optima. Originally proposed by Glover (1977) as an optimization tool applicable to nonlinear covering problems, its present form was proposed 9 years later (Glover 1986), and with even more details several years after that (Glover 1989).

TS's name comes from a milder version of the dictionary definition of "tabu," based on the idea of imposing restrictions to prevent a stochastic search from falling into infinite loops and other undesirable behavior. TS is divided into three parts: *preliminary search*, *intensification*, and *diversification*. Preliminary search, the most important part of the algorithm, works as follows. From a specified initial configuration, TS examines all neighbors and identifies the one with the highest value of the objective function. Moving to this configuration might not lead to a better solution, but TS moves there anyway; this enables the algorithm to continue the search without becoming blocked by the absence of improving moves and to escape from local optima. If there are no improving moves (indicating a kind of local optimum), then TS chooses one that least degrades the objective function. To avoid returning to the local optimum just visited, the reverse move must be forbidden. This is done by storing this move (or, more precisely, a characterization of this move) in a data structure—the *tabu list*—often managed like a circular list, empty at the beginning and with a first-in-first-out mechanism, so that the latest forbidden move replaces the oldest one. This list contains a number s of elements defining forbidden (tabu) moves; the parameter s is called the *tabu list size*. The tabu list as described may forbid certain relevant or interesting moves, as exemplified by those that lead to a better solution than the best one found so far. In view of this, TS introduces an aspiration criterion to allow tabu moves to be chosen anyway if they are judged to be sufficiently interesting. The next stage is intensification, which begins at the best solution found so far and clears the tabu list. The algorithm then proceeds as in the preliminary search phase. If a better solution is found, then intensification is restarted. The user can specify a maximum number of restarts after which the algorithm goes to the next step. If the current intensification phase does not find a better solution after a specified number of iterations, then the algorithm also goes to the next stage. Intensification provides a simple way to focus the search around the current best solution.

The final stage, diversification, again starts by clearing the tabu list and sets the s most frequent moves of the run so far to be tabu, where s is the tabu list size. Then a random state is chosen, and the algorithm proceeds to the preliminary search phase for a specified number of iterations. Diversification provides a simple way to explore regions that have been little visited to date. After the end of the third stage, the best solution (or K best solutions) found so far may be reported, or the entire algorithm may be repeated (always storing the K best solutions so far) either a specified number of times or until a preset CPU limit is reached.

3. SIMULATION RESULTS

3.1 The $p = 14$ Case

Our initial strategy for learning about the performance of the three stochastic optimization methods—exploring the sensitivity of their performance to user-chosen inputs and finding the best inputs—was to pit generic versions of them (i.e., implementations not specially tailored to the details of the Bayesian decision theory task) against each other in the situation summarized in Figure 1, in which the best subsets of the $p = 14$ variables in the RAND pneumonia scale of Table 1 were already known. To give all of the methods a realistically small amount of CPU time with $p = 14$ (to simulate the situation with larger p), we made a number of runs, limiting each method to only 20 minutes of CPU time at 400 (Unix) MHz. For each optimization method, we made 30 runs at each input setting with different random number seeds and averaged the results.

We needed a method for choosing N, the number of Monte Carlo replications on which the estimated expected utility is based. Rather than fixing N at, say, 15 for all runs, we implemented an idea involving adaptive choice of N. In our adaptive method, 20 models are chosen at random to initialize the search and evaluated with, for example, $N^* = 15$, creating a league table of the current 20 best models (our health policy experts advised that clinicians would find it far more useful to have a table of the $K = 20$ best models rather than the single best model found), and then a new model is chosen and evaluated with $N = 1$. If its apparent utility would place it somewhere in the current league table, then the utility is evaluated for $(N^* - 1) = 14$ more random splits and averaged. If it still belongs in the league table, then it is added at the appropriate place; if not, then it is ignored. We found that this adaptive-N^* approach was substantially better than the fixed-N approach for all optimization methods examined (see Fouskakis 2001 for quantitative comparisons), and we used the adaptive procedure to obtain all of the results presented herein.

3.1.1 Tabu Search. With TS in our problem, six user inputs were examined:

- The total number r of repetitions of the algorithm (varied from 1 to 6)
- The maximum number N^* of random modeling/validation splits on which the estimated expected utility is based (either 1 or N^* according to the adaptive method described earlier; we varied N^* from 2 to 20)
- The number l of preliminary searches per repetition (varied from 2 to 21)
- The number i of intensification searches per repetition (varied from 2 to 40)
- The number t of random restarts in each intensification search (varied from 0 to 8)
- The number d of diversification searches per repetition (varied from 1 to 15).

A full factorial simulation design across all six of these inputs was not possible because many of them led to much longer CPU times than the target of 1,200 seconds. By trial and error, we were able to find 49 combinations of input settings, each of which took approximately 1,200 seconds. The actual CPU time varied by input settings, from a mean (across the 30 runs) of 1,075 seconds to 1,575 seconds; we calculated both raw summaries and results adjusted (through regression) for differences in CPU time. In keeping with clinical judgment, as noted earlier, the main outcome variable that we studied was p_{20}, the percentage of the actual 20 best models (from the full-enumeration exercise summarized in Fig. 1) found with each set of inputs.

The performance of TS in our problem, which is summarized in Table 3, varied dramatically according to input settings, from an adjusted mean of 39% to 65%. (Tables 11 and 12 in online App. C contain the complete TS results.) Characterizing the effect of each input on performance is complicated, because of strong and high-order interactions among the input effects.

Table 3. An edited summary of simulation results in the $p = 14$ case for tabu search (TS), as a function of user-defined inputs

User-defined inputs						Mean CPU	p_{20} (% of 20 best models found)		
r	N^*	l	i	t	d	time, seconds	Raw mean	Raw SD	Adjusted mean
1	10	10	6	3	3	1,158	$.631_{(.020)}$	.109	.649
5	6	2	7	2	2	1,362	$.641_{(.016)}$	.089	.635
2	10	2	4	6	4	1,313	$.635_{(.016)}$	.090	.634
4	9	2	4	1	2	1,386	$.636_{(.017)}$	.091	.627
2	5	14	15	5	1	1,243	$.611_{(.016)}$	.089	.619
1	10	9	12	3	3	1,273	$.615_{(.015)}$	.084	.619
1	14	11	2	0	1	1,253	$.611_{(.019)}$	.106	.618
6	7	2	4	2	2	1,574	$.650_{(.017)}$	.095	.618
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6	6	4	2	2	3	1,319	$.523_{(.022)}$	.122	.522
1	20	3	5	0	2	1,240	$.475_{(.027)}$	.147	.483
2	10	4	2	1	6	1,300	$.441_{(.027)}$	.148	.442
2	10	4	8	1	2	1,541	$.446_{(.035)}$	.193	.418
3	2	15	35	8	2	1,448	$.430_{(.020)}$	.110	.413
1	15	5	3	1	6	1,191	$.398_{(.047)}$	.257	.412
1	15	6	12	1	5	1,523	$.431_{(.039)}$	.214	.406
5	8	4	1	1	3	1,327	$.395_{(.028)}$	.155	.392

NOTE: Values in parentheses are Monte Carlo standard errors; entries are sorted by adjusted means of p_{20}.

Only one clear pattern emerged: Good runs of TS tended to have input values in the middle of the ranges that we explored, and bad runs tended to have values at the extremes of those ranges.

3.1.2 Simulated Annealing. With SA, five user inputs were explored:

- The total number r of iterations (varied from 130 to 2,500)
- N^*, as in TS (varied from 2 to 20)
- The starting value T_f and final value T_0 of the temperature (varied across the settings $(T_f, T_0) = \{(10.0, 1.0), (10.0, .1), (2.5, .1), (1.0, .1), (.5, .05)\}$)
- The schedule, sc, used to decrease the temperature (1, straight line; 2, geometric; 3, reciprocal; 4, logarithmic).

By trial and error, we were able to find 108 combinations of input settings that spanned the full range of possibilities across these five factors, with the runs from each input setting taking approximately 1,200 seconds. The actual CPU time varied by input settings, from a mean (across the 30 runs) of 986 seconds to 1,471 seconds; thus, as before, we calculated both raw summaries and results adjusted (by regression) for differences in CPU time.

The performance of simulated annealing in this problem (see Table 4 for an edited summary, and Tables 13–15 in online App. C for the complete results) varied even more dramatically than in the TS case according to input settings, from an adjusted mean of 0% to 56%. Characterizing the effect of each input on performance again is complicated, because of strong and high-order interactions among the input effects. The following conclusions emerged:

1. The best SA runs were below the median TS results.
2. Small values of N^* (up to 10) were best.
3. $(T_f, T_0) = (1.0, .1)$ and $(.5, .05)$ gave the best performance.
4. The reciprocal and logarithmic cooling schedules were best in this problem.
5. $(T_f, T_0) = (10.0, 1.0)$, large N^*, and the straight-line schedule performed badly.

3.1.3 Genetic Algorithms. With GA, we explored six user inputs:

- The total number r of repetitions (varied from 2 to 237)
- N^*, as in TS and SA (varied for GA from 2 to 15)
- The population size n (three settings, 30, 50, and 80)
- The crossover strategy (c, p_c). We used $c = 1$ (simple), 2 (uniform), and 3 (highly uniform) crossover. With the first strategy, we used a crossover probability, p_c, of .88 when the population size was 30, .5 when the population size was 50, and .3 when the population size was 80.
- Elitist or nonelitist strategy (e, p_m). In the elitist strategy, the offspring are compared with the parents, and the best two among the four are chosen; with the nonelitist strategy, the offspring are always chosen. With the elitist strategy, mutation is not performed; with the nonelitist strategy, mutation occurs with probability $p_m = .01$
- At the end of each repetition, we either cleared the population and randomly generated a new one $(q = 0)$ or kept the population as it was and used it as the starting population for the new runs $(q = 100)$.

Here we were able to perform a complete full factorial experiment of 144 combinations, each of which was targeted to take approximately 1,200 seconds. As with TS and SA, the actual CPU time varied by input settings from a mean (across the 30 runs) of 988 to 1,994 seconds; thus, as before, we calculated both raw summaries and results adjusted (by regression) for differences in CPU time.

Table 4. An edited summary of simulation results in the $p = 14$ case for simulated annealing (SA), as a function of user-defined inputs

User-defined inputs					Mean CPU	p_{20} (% of 20 best models found)		
r	N^*	T_f	T_0	sc	time, seconds	Raw mean	Raw SD	Adjusted mean
1,050	4	1	.1	3	1,146	$.538_{(.021)}$	.113	.555
1,000	5	.5	.05	1	1,087	$.515_{(.016)}$	.088	.551
860	5	.5	.05	2	1,134	$.523_{(.023)}$	.125	.544
900	5	2.5	.1	3	1,140	$.520_{(.021)}$	.116	.539
1,120	3	1	.1	3	1,015	$.470_{(.023)}$	.127	.530
240	10	.5	.05	4	1,113	$.501_{(.018)}$	.097	.530
240	10	1	.1	4	1,085	$.490_{(.027)}$	.150	.527
900	5	1	.1	3	1,138	$.505_{(.018)}$	.098	.524
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
480	15	10	1	2	1,105	$.070_{(.014)}$	.077	.100
580	15	10	.1	1	1,210	$.098_{(.024)}$	.133	.094
360	15	10	1	4	1,147	$.068_{(.014)}$	.074	.085
1,000	10	10	1	1	1,202	$.076_{(.013)}$	.073	.075
360	20	10	1	2	1,128	$.050_{(.011)}$	.061	.073
340	20	10	1	3	1,133	$.036_{(.012)}$	.064	.058
360	20	10	1	1	1,163	$.033_{(.009)}$	.051	.044
800	15	10	1	1	1,426	$.055_{(.010)}$	.057	.000

NOTE: Values in parentheses are Monte Carlo standard errors; entries are sorted by adjusted means of p_{20}. Negative adjusted mean p_{20} values have been truncated at 0 but appear in rank order corresponding to their untruncated values.

Table 5. An edited summary of simulation results in the $p = 14$ case for the genetic algorithm (GA), as a function of user-defined inputs

									p_{20} (% of 20 best models found)		
r	N^*	n	p_c	p_m	c	e	q	Mean CPU time, seconds	Raw mean	Raw SD	Adjusted mean
22	2	80	0	0	2	1	100	1,231	$.646_{(.013)}$	.070	.665
98	5	50	0	0	3	1	100	1,301	$.655_{(.015)}$	.083	.664
46	2	50	0	0	2	1	100	1,282	$.638_{(.019)}$	.103	.650
20	5	50	0	0	2	1	100	1,307	$.630_{(.018)}$	.100	.638
16	10	30	0	0	2	1	100	1,236	$.590_{(.023)}$	.125	.608
74	10	30	0	0	3	1	100	1,282	$.591_{(.011)}$	.058	.603
166	2	50	0	0	3	1	100	1,267	$.585_{(.014)}$	.074	.599
165	5	30	0	0	3	1	100	1,335	$.588_{(.013)}$	.072	.593
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
4	10	50	0	.01	3	0	0	1,660	$.026_{(.007)}$	.040	0
2	15	80	0	.01	3	0	100	1,634	$.023_{(.007)}$	.038	0
4	10	50	0	0	2	1	0	1,670	$.020_{(.007)}$	.038	0
3	15	80	.3	.01	1	0	0	1,911	$.030_{(.007)}$	.038	0
2	15	80	0	.01	3	0	0	1,994	$.033_{(.009)}$	.047	0
2	15	80	0	0	2	1	0	1,979	$.028_{(.006)}$	.033	0
2	15	80	0	.01	2	0	0	1,984	$.025_{(.008)}$	.045	0
3	15	80	.3	0	1	1	0	1,889	$.011_{(.005)}$	.025	0

NOTE: Values in parentheses are Monte Carlo standard errors; entries are sorted by adjusted means of p_{20}. Negative adjusted mean p_{20} values have been truncated at 0 but appear in rank order corresponding to their untruncated values. For crossover schedules 2 and 3, the p_c column is not applicable. With the elitist strategy ($e = 1$), the p_m column is not applicable. When p_c and/or p_m are not applicable, the symbol 0 is used; this does not mean that the probability was 0.

Table 5 presents an edited version of the results for GA (Tables 16–19 in online App. C provide the complete findings). The performance of GA varied even more dramatically according to input settings compared with TS or SA, from an adjusted mean of 0% to 66%. This maximum value was better than that achieved by any settings of TS or SA. The conclusions in this case are clearer than those from the other two algorithms:

- It is far better to use elitist strategies and keep the population at the end of every repetition.
- The uniform and highly uniform crossover strategies are much better than the simple one-bit crossover. The more recent versions of GA in the literature, with elitist strategies and more complicated crossover operations, vastly outperform implementations based only on the basic GA ideas in use up through the mid-1980s.
- Again, small values of N^* (up to 10) were best.
- Smaller values of the population size n (30 and 50) gave better results than runs with population size 80.

3.1.4 Improved Simulated Annealing (ISA). The performance of generic SA in the $p = 14$ case was disappointing, so we explored two kinds of potential improvements to SA. The first of these was the TS idea of random restarts. A detailed examination of a number of SA output files revealed that even with temperature schedules that dropped the temperature slowly and with fairly high final temperatures, SA tended to get stuck in a local maximum of the criterion function fairly early in the run. The second was a method for teaching SA about the desirability of the predictor variables (i.e., making it more aware of the context of the Bayesian decision-theory problem), based on trading off data collection costs and predictive accuracy in the following ad hoc way:

- Scale the marginal costs c_j by calculating $\frac{c_j}{\min c_j}$; small values are cost-effective.
- Scale the correlations with the ratio $\frac{\max r_j}{|r_j|}$; again, the cost-effective variables are small on this.
- Take the product d_j, as in column 7 of Table 6.

As it happens, this particular ad hoc desirability measure correlates well with whether or not a given variable is cost-effective; the predictors with the eight smallest values of d_j in Table 6 agree with the eight cost-effective variables in Table 1. Of course, the only way we know this is to go through the exercise of maximizing expected utility; other ad hoc measures might look just as plausible, and how can we choose among them? Also, sorting on column 7 in Table 6 rank-orders the variables in desirability, but says nothing about how many should be used to achieve the optimal tradeoff. Nevertheless, one possible application of the d_j, when normalized to probabilities appropriately, would be to make a method like SA more "intelligent" in deciding which variable to bring next into the current model.

To achieve better simulated annealing results, we defined an improved SA (ISA) algorithm, as follows:

1. ISA begins by choosing $K = 20$ models completely at random and evaluating their estimated expected utility (EEU) values using N^* replications. This is done to initialize the league table of the 20 best models found so far.
2. Next, ISA starts the stochastic search at the null model (with no predictors), which becomes the current model, and computes its EEU value using the adaptive-N^* method.
3. ISA then begins proposing moves away from the current model using one-bit flips (at locations in the binary string governed by a pointer that scans from 1 to p and then resets to 1) and the variable desirability criterion d_j. From

Table 6. An ad hoc measure of the desirability of a variable in compromising between predictive accuracy and data collection costs, in the case where $p = 14$

Variable	Cost c_j (U.S.$)	Correlation r_j	Cost-effective?	(1) $\frac{c_j}{\min c_j}$	(2) $\frac{\max r_j}{\lvert r_j \rvert}$	$d_j =$ (1) · (2)	p_j^{in}
APACHE	3.33	.39		20	1.0	20.0	.220
Age	.17	.17	*	1	2.3	2.3	.802
SBP	.17	.29	**	1	1.3	1.3	.876
CHF	.83	.10		5	3.9	19.5	.226
BUN	.50	.32	**	3	1.2	3.7	.711
Coma	.83	.35	**	5	1.1	5.6	.605
Albumin	.50	.20	*	3	2.0	5.9	.590
Shortness of breath	.33	.13	**	2	3.0	6.0	.585
Respiratory distress	.33	.18	*	2	2.2	4.3	.675
Septic	1.00	.06		6	6.5	39.0	.118
Respiratory failure	.67	.08		4	4.9	19.5	.226
Recently hospitalized	.67	.14		4	2.8	11.1	.391
Ambulatory	.83	.22		5	1.8	8.9	.463
Temperature	.17	−.06	*	1	6.5	6.5	.562

NOTE: The rightmost column, a function of desirability calculated using (7), characterizes the probability of the given variable entering the model after not being included in the previous step. Cost-effective: Variables marked * and ** appeared frequently in the 20 best models; ** identifies the variables in the globally best model. Variables are in the same order as in Table 1, where their full names are given.

the d_j we created probabilities p_j^{in} and p_j^{out}—for flipping a 0 to a 1 and vice versa, using the (ad hoc) transformation from desirability to probability given by

$$p_j^{\text{in}} = p_{\min} + (p_{\max} - p_{\min})e^{-c(d_j - 1)}, \tag{7}$$

where $(p_{\min}, p_{\max}, c)$ are tuning constants to be specified by the user and $p_j^{\text{out}} = 1 - p_j^{\text{in}}$. Here $p_{\min}$ and $p_{\max}$ govern how dogmatic the inclusion and exclusion processes should be, and c controls the rate at which desirability translates into probability of inclusion. Experimentation led us to the choices $(p_{\min}, p_{\max}, c) = (.1, .9, .1)$, which yielded the inclusion probabilities in the last column of Table 6 in the 14-variable case.

A move away from the current model is then governed by two processes in sequence. First, a move is either proposed or not at random based on the p_j^{in} and p_j^{out} values, and then if a move is proposed, it either occurs or not according to the usual SA acceptance probabilities.

4. Step 3 is repeated until the algorithm gets stuck in the same place for k consecutive iterations, where, again after experimentation, we chose $k = 50$ as a good compromise for $p = 14$. If k successive steps without a move occur at any time during the run, then ISA implements a random restart. The temperature is again set to T_0, a random initial model is generated, and cooling from this temperature begins all over again, exactly as at the beginning of the entire algorithm.

Throughout the run, the league table of 20 best models is constantly updated. Steps 3 and 4 are then iterated until the desired amount of CPU time has been exhausted.

We performed a simulation experiment with ISA similar to the study with generic SA, the results of which were given in Section 3.1.2. The results, summarized in Table 7 and given completely in Tables 20–22 in online Appendix C were much better than those for generic SA, the best input settings achieved an adjusted mean value of p_{20} of 74.2% (about 34% better than the top result for generic SA). With the modifications of SA involving random restarts and the inclusion and exclusion of variables based on desirability, the new ISA outperformed both versions of TS and GA studied previously in the case where $p = 14$. (Peak ISA performance was about 14% and 12% better than the top results for TS and GA.)

It is possible that inclusion and exclusion probabilities based on variable desirability, as in ISA, could be used to improve TS as well. This is a subject of continuing investigation.

3.1.5 Summary of Results With $p = 14$. Figure 2 presents parallel notched boxplots comparing GA, SA, TS, and ISA in the 14-variable case as a function of their user inputs. Nonoverlap of the notch intervals for any pairwise comparison in the plot provides significant evidence of an underlying difference in median performance (Chambers, Cleveland, Kleiner, and Tukey 1983). The relative success of the four optimization methods can be evaluated in several ways: (1) which method has the best top performance (i.e., how the methods compare on their very best results); (2) which has the best median performance; and (3) which has the best bottom performance (this gives an idea of how well a method would perform if little attention were paid to fine-tuning the user inputs). The following conclusions with $p = 14$ (i.e., when optimizing over a space of modest size with a real-valued objective function and binary inputs) may be drawn about the role of user-specified inputs in stochastic optimization:

- In our optimization problem, ISA and SA were the overall winners and losers, respectively, in terms of top performance.
- TS was best and GA was worst (by a large margin) on median performance, and TS had by far the best bottom performance (corresponding to a user unfortunate enough to have specified sharply suboptimal algorithmic inputs).
- Tailoring a generic implementation of SA to the specifics of the optimization problem, as with ISA, can lead to noticeable improvements in the performance of the basic simulated annealing approach.

Table 7. An edited summary of simulation results in the $p = 14$ case for ISA, as a function of user-defined inputs

User-defined inputs				Mean CPU	p_{20} (% of 20 best models found)		
N^*	T_0	T_f	sc	time, seconds	Raw mean	Raw SD	Adjusted mean
15	.5	.05	4	1,349	$.748_{(.016)}$	.114	.742
20	.5	.05	4	1,494	$.753_{(.017)}$	.143	.734
15	10	.1	4	1,360	$.715_{(.017)}$	.089	.708
20	10	.1	4	1,349	$.706_{(.017)}$	.120	.700
20	2.5	.1	4	1,495	$.708_{(.016)}$	.110	.689
10	.5	.05	4	1,251	$.685_{(.018)}$	.109	.688
15	2.5	.1	4	1,407	$.698_{(.018)}$	.090	.687
10	10	.1	4	1,408	$.690_{(.018)}$	.102	.679
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
15	10	1	2	1,454	$.403_{(.016)}$	.101	.388
15	10	.1	2	1,073	$.361_{(.014)}$	.089	.380
2	10	1	1	1,377	$.381_{(.018)}$	.094	.373
2	10	1	2	1,350	$.378_{(.026)}$	.082	.372
20	10	1	1	1,096	$.320_{(.020)}$	.104	.337
20	10	1	2	1,174	$.305_{(.019)}$	.097	.315
20	10	.1	1	1,084	$.273_{(.014)}$	.096	.291
20	10	.1	2	1,073	$.233_{(.013)}$	.096	.252

NOTE: Values in parentheses are Monte Carlo standard errors; entries are sorted by adjusted means of p_{20}.

- GA had the greatest sensitivity, and TS was the least sensitive, to badly chosen user inputs.

It would seem that in problems of modest complexity, TS is relatively robust to less-than-optimal choices of user inputs. This suggests that it may deserve more attention in the statistics community than it has received so far.

3.2 The $p = 83$ Case

All of the results so far have been for the special case of $p = 14$ available predictor variables; in the full RAND pneumonia data, $p = 83$ predictors were available. Table 8 summarizes all of the "interesting" predictors, together with their data collection costs and their simple correlations with 30-day death. (Tables 23 and 24 in online App. C give details on the full set of all 83 variables.) A variable is "interesting" and thus included in this table if it (a) was in the original RAND scale, (b) was one of the eight best variables in the case where $p = 14$ (see Sec. 1.3 for details), or (c) was one of the 22 best variables in the case where $p = 83$ (see Sec. 3.2 for details). Table 8 shows that some of the variables in the $p = 83$ case, that were not in the original 14-variable RAND scale did not have a very high correlation with death, suggesting that the dimensionality of the space worth searching was less than that implied by including all 83 predictors. However, (a) many of the new variables did in fact have quite high marginal correlations with death (in fact,

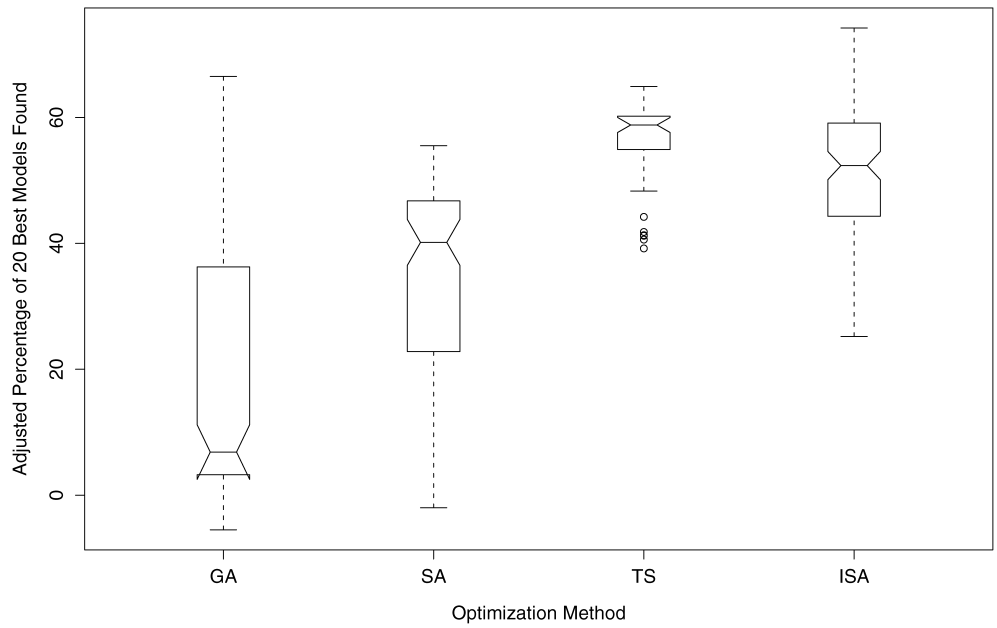

Figure 2. Parallel notched boxplots comparing GA, SA, TS, and ISA in the 14-variable case.

Table 8. The "interesting" subset of the full set of 83 variables, together with their approximate data collection costs per patient, simple correlations r with 30-day death, and presence or absence in the original RAND 14-variable scale

Variable	Cost c_j (U.S.$)	r	In RAND Scale?	Good? ($p = 14$)	Good? ($p = 83$)	%
Systolic blood pressure score	.17	.29	*	**	**	73
Age	.17	.17	*	*		
Blood urea nitrogen	.50	.32	*	**	**	100
APACHE II coma score	.83	.35	*	**	**	83
Shortness of breath day 1	.33	.13	*	**	*	31
Serum albumin score	.50	.20	*	*		
Respiratory distress score	.33	.18	*	*		
Septic complications	1.00	.06	*			
Prior respiratory failure	.67	.08	*			
Recently hospitalized	.67	.14	*			
Initial temperature	.17	−.06	*	*	**	71
Heart rate day 1	.17	.16			*	10
Chest pain day 1	.17	−.15			*	31
Ambulatory score	.83	.22	*			
Prior antibiotics	.17	−.02			*	3
Multiple myeloma	.17	−.02			*	2
APACHE respiratory rate score	.33	.24			*	8
Comorbid aspiration score	.17	.09			*	7
Admission systolic blood pressure	.17	−.20			**	64
Congestive heart failure chest X-ray score	.83	.10	*			
Total APACHE II	3.33	.39	*			
Respiratory rate day 1	.17	.22			**	92
Confusion day 1	.17	.30			*	25
Influenza score	.17	−.04			*	2
Arrest in emergency room score	.17	.17			*	48
Comorbid cirrhosis score	.17	.01			*	2
Comorbid congestive heart failure score	.17	.08			*	33
Comorbid alcoholism score	.17	−.03			*	15
Comorbid steroids score	.17	.01			*	1
Neurologic history score	.17	.28			*	1
Musculoskeletal score	.17	.17			*	5

NOTE: A variable is "interesting," and included in this table, if it was in the original RAND scale, it was one of the eight best variables in the case where $p = 14$ (see Sec. 1.2 for details), or it was one of the 22 best variables in the case where $p = 83$ (see Sec. 3.2 for details). The final three columns are explained in Section 3.2.

higher than some of the variables in the final 14-variable RAND scale), and (b) with $p = 83$, we wanted to learn about how well the three main optimization methods studied here (SA, GA, and TS) performed in a high-dimensional space and without a great deal of problem-specific tailoring of the algorithms. Results for ISA-style modifications of SA with $p = 83$ are summarized at the end of this section.

Because full enumeration was impossible with $p = 83$, our first task was to create a workable proxy for it. We created this proxy as follows:

- First, we chose one "good" input configuration each from TS, SA, and GA (from Tables 3–5), where "good" means a compromise between the best results from $p = 14$ and a desire for each method to visit a large number of models. In practice, we chose an input configuration for each method that was among the top 15 with $p = 14$.
- We then ran each algorithm with these "good" configurations for 1 week of CPU time at 400 MHz on the $p = 83$ case (using random starting models). Each method visited about 630,000 models in that time; the total across the three methods was 1,900,377, although this figure included many duplicate models.
- We eliminated all of the duplicates, finally arriving at 825,635 unique models visited by the three methods in 1 week each. We then sorted these models on their apparent utility (in each case N^* was 1, 4, or 5, based on the adaptive method), and extracted the 3,000 best models on this basis. Finally, we performed a full-enumeration exercise on these 3,000 models (with $N = 500$) to find their actual (as opposed to apparent) expected utility, and then sorted them one more time on their actual utility values.

In this section the 3,000 models and their utilities obtained in this way are cousidered "truth" for the purpose of comparing SA, GA, and TS with $p = 83$.

Figure 3 summarizes the estimated (actual) expected utility values from the 3,000 best models found in the 1-week runs, as a function of the number of variables in the model. This plot is a rough analog of Figure 1, with the roughness appearing because it is not a full enumeration of all models with 1–22 variables. Even so, the approximately concave shape traced out by the medians and maxima of most of the boxplots is clear and demonstrates that in the 83-variable case, the best models have 5–10 variables. This is only slightly larger than in the $p = 14$ case, where the optimal range was 4–7, even though the optimization

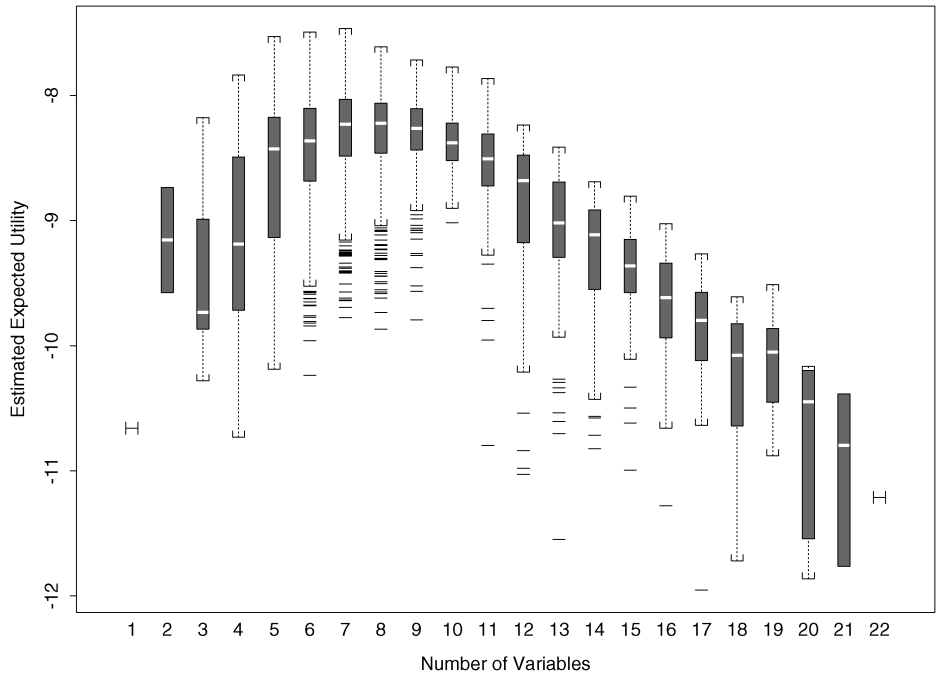

Figure 3. Estimated (actual) expected utility as a function of number of predictors retained, based on the 3,000 best models found from the 1-week runs with $p = 83$.

methods have 69 more variables to work with; this is because the RAND scale harvested so many of the variables with good univariate predictive performance. Table 9 summarizes the dimensions of the models found by the three methods. The table clearly shows that GA achieves its good results by finding its way faster (from a random starting point) to the smaller models where the best utilities are concentrated; 81% of the 3,000 best models were found by GA, with 19% discovered by TS, and only 1 out of the entire 3,000 (.03%) obtained by SA. When we restricted attention only to those models with $N > 1$, for which the utility determination was more accurate, the results were even more striking in favor of GA: 95% from GA, 5% from TS, and .1% from SA. As was the case with $p = 14$, the best versions of GA were those that incorporated the more recent innovations of highly uniform crossover strategy, elitist selection (and thus no mutation), and retention of 100% of the current members of the population at the beginning of each repetition of the algorithm.

Columns 6 and 7 of Table 8 identify the most promising variables from the 1-week runs. The second of these columns gives the percentage of time each variable appeared among the 100 best models if that frequency was at least 1%; 61 of the 83 variables failed this test. The six most common variables in the 100 best models, denoted by double asterisks (**) in column 6, were (in decreasing order of frequency) blood urea nitrogen, respiratory rate day 1, APACHE II coma score, systolic blood pressure score, initial temperature, and admission systolic blood pressure. Four of these variables were identified in the parallel exercise with $p = 14$, but two are new: there is extra cost-effective information in the actual values of the admission systolic blood pressure and respiratory rate on day 1 above and beyond what was present in the similar scales among the 14 RAND variables.

The overall best model found in the 1-week runs had seven variables: systolic blood pressure score, blood urea nitrogen, APACHE II coma score, initial temperature, admission systolic blood pressure, respiratory rate day 1, and arrest in emergency room score. The real utility achieved by this model, -7.47, was only U.S.$0.43 better than the corresponding figure with $p = 14$; this again is a consequence of the RAND 14-variable scale being so heavily packed with variables with good univariate predictive behavior.

GA's advantage over SA and TS with $p = 83$ arose because GA was able to examine many more models than the other methods in the same amount of CPU time. Table 10 compares the total number of utility evaluations achieved by the three optimization methods in a series of runs with 24 hours of CPU

Table 9. Distribution of model dimension in the 3,000 best models from the 1-week runs, by optimization method

	All n			$n > 1$		
Method	Mean	SD	%	Mean	SD	%
SA	18.0	0	.03	18.0	0	.1
GA	7.5	1.7	81.20	7.3	1.4	94.9
TS	14.9	2.0	18.77	12.5	1.2	5.0
Total	8.9	3.4	100.0	7.6	1.8	100.0

Table 10. Comparison of the total number of utility evaluations achieved by the three optimization methods in the 24-hour runs

		Number of models visited with		Number of utility
Method	N^*	$N = 1$	$N > 1$	evaluations
SA	4	62,912	1,004	66,928
GA	2	104,204	10,706	125,616
TS	5	37,235	5,653	65,500

time. With the same CPU budget, GA was able to find the time to evaluate almost twice as many utilities as the other two methods. Closer examination of the precise use of CPU time by the three algorithms revealed that this difference was attributable to the amount of extra "overhead" required by TS and SA that is not present in GA. SA spent a noticeable amount of time making calculations (involving expensive calls to the logarithm and exponential functions) to support its cooling schedule and acceptance probabilities, and TS used a substantial amount of CPU time managing the tabu list.

We also performed a number of runs in which GA, TS, SA and ISA were given 3 hours of CPU time for each of a number of choices of user inputs; details are given in Tables 25–28 in online Appendix C. As was the case with $p = 14$, modifying SA to permit random restarts and the inclusion and exclusion of variables based on desirability sharply improved the performance of the simulated annealing algorithm, with the result that ISA performed about as well as GA with $p = 83$.

4. CONCLUSIONS AND DISCUSSION

Our work offers a relatively new perspective on variable selection in GLMs: when monetary resources are constrained (as will almost always be true) and the purpose of a modeling exercise is the creation of a predictive rule for future cases, data collection costs need to be traded off against predictive accuracy in choosing an optimal subset of predictors. We have found that Bayesian decision theory, through maximization of expected utility, is a good way to quantify this trade-off.

We draw the following conclusions from this study regarding the performance of the stochastic optimization methods that we examined:

- When optimizing a real-valued criterion function $f(x_1, \ldots, x_p)$ for binary x_j, for moderate p (e.g., choosing among $2^{14} = 16{,}384$ possible input configurations), genetic algorithms performed relatively poorly in our problem for all but the very best user-defined input configurations, and generic simulated annealing also did not perform well, whereas tabu search had excellent median performance and was much less sensitive to suboptimal choice of user-defined inputs. When optimizing over spaces of moderate size, tabu search appears to deserve more attention in the statistics community than it has received so far.
- Providing off-the-shelf stochastic optimization methods with extra "intelligence" based on the statistical context of the problem (e.g., with something like the desirability criterion of Sec. 3.1.4) can lead to substantial performance improvements.
- For large p (e.g., when there are $2^{83} \doteq 10^{25}$ possibilities to search), the two best optimization methods studied here were the most recent versions of genetic algorithms (their good performance with large p, in contrast to the situation when the optimization problem is of moderate size, is due to lower algorithmic overheads compared with other leading methods) and the improved version of simulated annealing proposed in Section 3.1.4 (which did well because of the extra problem-specific information on which its variable selection was based).

The decision-theoretic methodology that we have examined here holds broad promise for cost-effective IO quality and performance assessment in any setting where the outcome is dichotomous and a number of predictors are potentially available that differ substantially in data collection costs. Examples other than the health policy setting of this article include education (with such outcomes as dropout rates and employment rates after graduation), personnel management (with a focus on retention rates in the workplace), and credit scoring in business (where defaulting on a loan is the outcome of interest).

[Received June 2006. Revised July 2007.]

REFERENCES

Berger, J. O. (1985), *Statistical Decision Theory and Bayesian Analysis*, New York: Springer-Verlag.

Bernardo, J. M., and Smith, A. F. M. (1994), *Bayesian Theory*, New York: Wiley.

Brown, P. J., Fearn, T., and Vannucci, M. (1999), "The Choice of Variables in Multivariate Regression: A Non-Conjugate Bayesian Decision Theory Approach," *Biometrika*, 86, 635–648.

California Nursing Outcomes Coalition (2008), "The California Nursing Outcomes Coalition Database Project," available at *www.calnoc.org*.

Centers for Medicare & Medicaid Services (2008), "Medicare Information Resource," available at *www.cms.hhs.gov/home/medicare.asp*.

Chambers, J. M., Cleveland, W. S., Kleiner, B., and Tukey, P. A. (1983), *Graphical Methods for Data Analysis*, Belmont, CA: Wadsworth.

Daley, J., Jencks, S., Draper, D., Lenhart, G., Thomas, N., and Walker, J. (1988), "Predicting Hospital-Associated Mortality for Medicare Patients With Stroke, Pneumonia, Acute Myocardial Infarction, and Congestive Heart Failure," *Journal of the American Medical Association*, 260, 3617–3624.

DeGroot, M. H. (1970), *Optimal Statistical Decisions*, New York: McGraw-Hill.

Donabedian, A., and Bashshur, R. (2002), *An Introduction to Quality Assurance in Health Care*, Oxford, U.K.: Oxford University Press.

Draper, D. (1995), "Inference and Hierarchical Modeling in the Social Sciences" (with discussion), *Journal of Educational and Behavioral Statistics*, 20, 115–147, 233–239.

Draper, D., and Fouskakis, D. (2000), "A Case Study of Stochastic Optimization in Health Policy: Problem Formulation and Preliminary Results," *Journal of Global Optimization*, 18, 399–416.

Draper, D., and Gittoes, M. (2004), "Statistical Analysis of Performance Indicators in U.K. Higher Education" (with discussion), *Journal of the Royal Statistical Society*, Ser. A, 167, 449–474.

Draper, D., Kahn, K., Reinisch, E., Sherwood, M., Carney, M., Kosecoff, J., Keeler, E., Rogers, W., Savitt, H., Allen, H., Wells, K., Reboussin, D., and Brook, R. (1990), "Studying the Effects of the DRG-Based Prospective Payment System on Quality of Care: Design, Sampling, and Fieldwork," *Journal of the American Medical Association*, 264, 1956–1961.

Dubois, R., Rogers, W., Moxley III, J., Draper, D., and Brook, R. (1987), "Hospital Inpatient Mortality: Is It a Predictor of Quality?" *New England Journal of Medicine*, 317, 1674–1680.

Eshelman, L. (1991), "The CHC Adaptive Search Algorithm: How to Have Safe Search When Engaging in Nontraditional Genetic Recombination," in *Foundations of Genetic Algorithms*, ed. G. Rawlins, San Mateo, CA: Morgan Kaufmann.

Fouskakis, D. (2001), "Stochastic Optimization Methods for Cost-Effective Quality Assessment in Health," unpublished doctoral dissertation, University of Bath, Dept. of Mathematical Sciences.

Fouskakis, D., and Draper, D. (2002), "Stochastic Optimization: A Review," *International Statistical Review*, 70, 315–349.

Fouskakis, D., Ntzoufras, I., and Draper, D. (2009a), "Bayesian Variable Selection Using Cost-Adjusted BIC, With Application to Cost-Effective Measurement of Quality of Health Care," *Annals of Applied Statistics*, forthcoming.

——— (2009b), "Population-Based Reversible-Jump MCMC for Bayesian Variable Selection and Evaluation Under Cost Constraints," *Journal of the Royal Statistical Society*, Ser. C, forthcoming.

Gelfand, A. E., Dey, D. K., and Chang, H. (1992), "Model Determination Using Predictive Distributions With Implementation via Sampling-Based Methods," in *Bayesian Statistics 4*, eds. J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, Oxford, U.K.: Oxford University Press, pp. 147–167.

Glover, F. (1977), "Heuristics for Integer Programming Using Surrogate Constraints," *Decision Sciences*, 8, 156–166.

——— (1986), "Future Paths for Integer Programming and Links to Artificial Intelligence," *Computers and Operations Research*, 13, 533–549.

——— (1989), "Tabu Search—Part I," *ORSA Journal on Computing*, 1, 190–206.

Goldstein, H., and Spiegelhalter, D. J. (1996), "League Tables and Their Limitations: Statistical Issues in Comparisons of Institutional Performance" (with discussion), *Journal of the Royal Statistical Society*, Ser. A, 159, 385–409.

Hadorn, D., Draper, D., Rogers, W., Keeler, E., and Brook, R. (1992), "Cross-Validation Performance of Patient Mortality Prediction Models," *Statistics in Medicine*, 11, 475–489.

Holland, J. H. (1975), *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: University of Michigan Press.

Hosmer, D., and Lemeshow, S. (2000), *Applied Logistic Regression* (2nd ed.) New York: Wiley-Interscience.

Jencks, S., Daley, J., Draper, D., Thomas, N., Lenhart, G., and Walker, J. (1988), "Interpreting Hospital Mortality Data: The Role of Clinical Risk Adjustment," *Journal of the American Medical Association*, 260, 3611–3616.

Judge, G. G., Hill, R. C., Griffiths, W. E., Lütkepohl, H., and Lee, T.-C. (1988), *The Theory and Practice of Econometrics* (2nd ed.), New York: Wiley.

Kahn, K., Brook, R., Draper, D., Keeler, E., Rubenstein, L., Rogers, W., and Kosecoff, J. (1988), "Interpreting Hospital Mortality Data: How Can We Proceed?" *Journal of the American Medical Association*, 260, 3625–3628.

Kahn, K., Rogers, W., Rubenstein, L., Sherwood, M., Reinisch, E., Keeler, E., Draper, D., Kosecoff, J., and Brook, R. (1990a), "Measuring Quality of Care With Explicit Process Criteria Before and After Implementation of the DRG-Based Prospective Payment System," *Journal of the American Medical Association*, 264, 1969–1973.

Kahn, K., Rubenstein, L., Draper, D., Kosecoff, J., Rogers, W., Keeler, E., and Brook, R. (1990b), "The Effects of the DRG–Based Prospective Payment System on Quality of Care for Hospitalized Medicare Patients: An Introduction to the Series," *Journal of the American Medical Association*, 264, 1953–1955.

Keeler, E., Kahn, K., Draper, D., Rogers, W., Sherwood, M., Rubenstein, L., Reinisch, E., Kosecoff, J., and Brook, R. (1990), "Changes in Sickness at Admission Following the Introduction of the Prospective Payment System," *Journal of the American Medical Association*, 264, 1962–1968.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983), "Optimization by Simulated Annealing," *Science*, 220, 671–680.

Knaus, W. A., Draper, E. A., Wagner, D. P., and Zimmerman, J. E. (1985), APACHE II: A Severity of Disease Classification System," *Critical Care Medicine*, 13, 818–829.

Lindley, D. V. (1968), "The Choice of Variables in Multiple Regression" (with discussion), *Journal of the Royal Statistical Society*, Ser. B, 30, 31–66.

NDNQI (2008), "National Database of Nursing Quality Indicators," available at *www.nursingquality.org*.

Normand, S., Glickman, M., and Gatsonis, C. (1997), "Statistical Methods for Profiling Providers of Medical Care: Issues and Applications," *Journal of the American Statistical Association*, 92, 803–814.

Open Clinical (2008), "AI Systems in Clinical Practice: APACHE III," available at *www.openclinical.org/aisp_apache.html*.

U.S. Department of Health and Human Services (2008), "FY 2009 President's Budget for HHS," available at *www.hhs.gov/budget/docbudget.htm*.