Numerical Optimization Using Computer Experiments*

Michael W. Trosset
Adjunct Associate Professor
Department of Computational & Applied Mathematics
Rice University
Houston, TX

Virginia Torczon
Assistant Professor
Department of Computer Science
College of William & Mary
Williamsburg, VA

Abstract

Engineering design optimization often gives rise to problems in which expensive objective functions are minimized by derivative-free methods. We propose a method for solving such problems that synthesizes ideas from the numerical optimization and computer experiment literatures. Our approach relies on kriging known function values to construct a sequence of surrogate models of the objective function that are used to guide a grid search for a minimizer. Results from numerical experiments on a standard test problem are presented.

*This research was supported in part by the Air Force Office of Scientific Research (AFOSR) under Grant No. F49620-95-1-0210 and also by NASA Contract Number NAS1-19480 while the authors were in residence at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681-0001.

1 Introduction

We consider the problem of minimizing an objective function $f : \Re^p \to \Re$ subject to bound constraints, i.e.

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in [a, b], \end{array} \tag{1}$$

where $a, x, b \in \Re^p$ and we write $x \in [a, b]$ to denote $a_i \leq x_i \leq b_i$ for $i = 1, \ldots, p$. We are concerned with special cases of Problem (1) for which evaluation of the objective function involves performing one (or more) complicated, deterministic computer simulation(s). Many such problems arise as engineering design problems and are often distinguished by two troubling characteristics that preclude solution by traditional algorithms for bound-constrained optimization.

First, the output of a complicated computer simulation is usually affected by a great many approximation, rounding and truncation errors. These errors are not stochastic—repeating the simulation will reproduce them—but their accumulation introduces high-frequency, low-amplitude distortions of the idealized objective that we would have liked to optimize. In consequence, optimization algorithms that compute or approximate (by finite differencing) derivatives of f often fail to exploit general trends in the objective function and become trapped in local minimizers created by high-frequency oscillations. In order to develop effective algorithms for such applications, we restrict attention to derivative-free methods for numerical optimization.

Second, complicated computer simulations are often expensive to perform. Frank (1995) suggested that one must address problems in which a typical function evaluation costs several hours of supercomputer time. For example, Booker (1996) and Booker et al. (1996) studied an aeroelastic and dynamic response simulation of a helicopter rotor blade for which a single function evaluation requires approximately six hours of cpu time on a Cray Y-MP. We formalize the notion that the objective function is expensive to evaluate by imposing an upper bound V on the number of evaluations of f that we are allowed to perform. The severity of this restriction will depend (in part) on the relation between V and p.

When attempting to minimize an objective function f that is too expensive for standard numerical optimization algorithms to succeed, it has long been a standard engineering practice, described by Barthelemy and Haftka (1993), to replace f with an inexpensive surrogate $\hat{f}$ and minimize $\hat{f}$ instead. (For example, one might evaluate f at $V - 1$ carefully selected sites, construct $\hat{f}$ from the resulting information, use a standard numerical optimization algorithm to minimize $\hat{f}$, and evaluate f at the candidate minimizer thus obtained.) This practice may also have the salutory effect of smoothing high-frequency oscillations in f. The rapidly growing literature on computer experiments offers new and potentially better ways of implementing this traditional practice. The prescription that seems to be gaining some currency in the engineering community was proposed by Welch and Sacks (1991); following current convention, we refer to it as DACE (Design and Analysis of Computer Experiments). Frank (1995) offered an optimizer's perspective on this methodology, suggested that the "minimalist approach" of minimizing a single $\hat{f}$ is not likely to yield satisfactory results, and proposed several sequential modeling strategies as alternatives. Booker (1996) studied several industrial applications of DACE and two alternative approaches.

It is not our purpose in this report to provide a thorough critique of DACE as a method for minimizing expensive objective functions. We regard DACE and traditional iterative methods for numerical optimization as occupying opposing ends of a spectrum. When V is large relative to p, say $p = 2$ and $V = 500$, then the expense of function evaluation is not an issue and we are content to rely on traditional iterative methods. When V is not large relative to p, say $p = 2$ and $V = 5$, then the expense of function evaluation is completely crippling and we are content to rely on DACE. (If $V < p$, then the methodologies that we consider are not appropriate.) In this report we are concerned with intermediate situations and we borrow ideas from both the numerical optimization and the computer experiment literatures. We describe a sequential modeling strategy in which computer experiment models are used to guide a grid search for a minimizer. Our methods elaborate and extend an important special case of the general model management strategy proposed by Dennis and Torczon (1996) and developed by Serafini (1997). These efforts are part of a larger collaboration described by Booker et al. (1995) and Booker et al. (1996).

2 Pattern Search Methods

We require a method of solving Problem (1) that does not require derivative information. For unconstrained optimization, a popular derivative-free method is the simplex method proposed by Nelder and Mead (1965). This method is sometimes adapted for constrained optimization by means of a simple *ad hoc* device, viz. setting $f(x) = \infty$ when $x \notin [a, b]$. Unfortunately, the Nelder-Mead simplex method is suspect even for unconstrained optimization. For example, McKinnon (1996) has constructed a family of strictly convex, differentiable objective functions on $\Re^2$ for which there exist starting points from which Nelder-Mead will fail to converge to a stationary point. Instead, we rely on a class of methods for which a convergence theory exists, the *pattern search methods* explicated by Torczon (1997) for the case of unconstrained optimization and extended by Lewis and Torczon (1996a) to the case of bound-constrained optimization.

Pattern search methods are iterative algorithms for numerical optimization. Such algorithms produce a sequence of points, $\{x_k\}$, from an initial point, x_0, provided by the user. To specify an algorithm, one must specify how it progresses from the current iterate, x_c, to the subsequent iterate, x_+. One of the distinguishing features of pattern search methods is that they restrict their search for x_+ to a grid (more formally, a lattice) that contains x_c. The grid is modified as optimization progresses, according to rules that ensure convergence to a stationary point.

The essential logic of a pattern search is summarized in Figure 1. The reader is advised that this description of pattern search methods differs from the presentation in Torczon (1997) and Lewis and Torczon (1996a, 1996b). For example, the choice of a starting point is usually not restricted and the initial grid is constructed so that it contains the starting point. More significantly, pattern search methods are usually specified by rules that prescribe where the algorithm is to search for the subsequent iterate and the notion of an underlying grid is implicit in these rules. In this report, the grid is explicit and the search for a subsequent iterate is not restricted to a specific pattern of points. What should be appreciated is that the present description preserves all of the elements of pattern search methods required by their convergence theory. A detailed explication of the connection between these perspectives will be provided by Serafini (1997).

It is evident that the crucial elements of a pattern search algorithm are contained in the specification of 2(b) in Figure 1. The fundamental idea is to try to find a point on the current grid that strictly decreases the current value of the objective function. Any such point can be taken to be the subsequent iterate. If one fails to find such a point, then one replaces the current grid with a finer one and tries again.

Torczon (1997) described the search in 2(b)(i) as an *exploratory moves algorithm*. Here we distinguish two components of an exploratory moves algorithm: an *oracle* that produces a set of trial points on the current grid and a *core pattern* of trial points on the grid at which the objective function must be evaluated before the algorithm is permitted to refine the grid. The convergence theory requires that the core pattern satisfy certain hypotheses; no hypotheses are placed on the oracle.

Because the methods proposed in this report critically depend on the arbitrary nature of the or-

1. Specify the current grid. Select x_0 from the current grid. Let $x_c = x_0$.
2. Do until convergence:
   (a) Let $T(+) = \emptyset$.
   (b) Do while $T(+) = \emptyset$:
      i. Search the current grid for a set of $x_t \in [a, b]$ at which f is then evaluated. Let $T(+)$ denote the set of grid points $x_t \in [a, b]$ thus obtained for which $f(x_t) < f(x_c)$.
      ii. Update the grid.
   (c) Choose $x_+ \in T(+)$.
   (d) Let $x_c = x_+$.

Figure 1: Pattern search methods for numerical optimization.

acle, we emphasize that *any method whatsoever* can be employed to produce points that potentially decrease the current value of the objective. We might perform an exhaustive search of the current grid or we might specify a complicated pattern of points at which to search. We might appeal to our prior knowledge of, or our intuition about, the objective function. It does not matter—the convergence theory for pattern search methods encompasses all such possibilities.

For the sake of clarity, we describe more fully a specific pattern search algorithm. First, we construct the grids on which the searches will be conducted. For $n = 0, 1, 2, \ldots$, we define vector lattices $\Gamma(n)$ restricted to the feasible set $[a, b]$ as follows: $x \in \Gamma(n)$ if and only if for each $i = 1, \ldots, p$ there exists $j_i \in \{0, 1, \ldots, 2^n\}$ such that

$$x_i = a_i + \frac{j_i}{2^n} (b_i - a_i) .$$

Thus, $\Gamma(0)$ comprises the vertices of $[a, b]$ and $\Gamma(n+1)$ is obtained from $\Gamma(n)$ by halving the distance between adjacent grid points (see below). When we update the current grid, say $\Gamma(n)$, in 2(b)(ii), we either retain $\Gamma(n)$ or replace $\Gamma(n)$ with $\Gamma(n+1)$.

Next we specify a core pattern. Given $x_c \in [a, b]$, we say that $x_t \in [a, b]$ is adjacent to x_c if and only if there exists $k \in \{1, \ldots, p\}$ such that

$$x_{tk} = x_{ck} \pm \frac{1}{2^n} (b_k - a_k)$$

and $x_{ti} = x_{ci}$ for $i \neq k$. We take as the core pattern the set of grid points adjacent to the current iterate. (For example, if the current grid is the integer lattice on $\Re^2$ restricted to $[a = (0, 0)', b = (8, 8)']$, then the core pattern of $(2, 0)'$ comprises $(3, 0)'$, $(2, 1)'$, and $(1, 0)'$.) We refine the grid, i.e. we replace $\Gamma(n)$ with $\Gamma(n + 1)$, if and only if we have evaluated f at each grid point x_t adjacent to x_c and failed to find $f(x_t) < f(x_c)$.

If f is continuously differentiable, then the theory developed by Lewis and Torczon (1996a) guarantees that the specified algorithm will produce a sequence $\{x_k\}$ that converges to a Karush-Kuhn-Tucker point of Problem (1). In practice, of course, the algorithm must terminate in a finite number of steps. Termination criteria for pattern search methods—indeed, for direct search methods in general—have not been studied extensively, but that does not concern us here. By definition, the assumption that Problem (1) is expensive means that we cannot afford enough evaluations of the objective function to terminate by the traditional criteria of numerical optimization. For the problems that we consider, the relevant termination criterion is whether or not we have exhausted the permitted number (V) of function evaluations.

Derivative-free methods for numerical optimization can be quite profligate with respect to the number of function evaluations that they require. Because the number of function evaluations available to us is severely limited, we want to use these evaluations as efficiently as possible. On the bright side, the convergence theory for pattern search methods allows us to replace x_c with *any* x_t for which $f(x_t) < f(x_c)$. Hence, no matter how comprehensive a search for trial points we may have envisioned, we can abort it as soon as we find a single trial point that satisfies $f(x_t) < f(x_c)$. On the dark side, the oracle may require a great many function evaluations to produce even one x_t for which $f(x_t) < f(x_c)$. Furthermore, if the oracle is unsuccessful, then we can not refine the current grid until after f has been evaluated at each grid point adjacent to x_c—a process that may require as many as $2p$ additional function evaluations if x_c is an interior grid point and f has not yet been evaluated at any of the grid points adjacent to x_c.

The fundamental purpose of this report is to introduce methods that reduce the expense of the oracle. We do not attempt to reduce the number of points in the core pattern. For unconstrained optimization, Lewis and Torczon (1996b) have introduced core patterns that contain only $p + 1$ points.

Pattern search algorithms may intentionally use large numbers of function evaluations. For example, the oracle employed by Dennis's and Torczon's (1991) *parallel direct search* intentionally casts a wide net, relying on parallel computation to defray the expense of evaluating the objective function at a great many grid points. We are concerned with problems for which huge numbers of function evaluations are not possible. Here, we want an oracle that proposes promising trial points using only a small number of function values. Our strategy for creating such an oracle will be to use previous function values to construct a current global model, $\hat{f}_c$, of f, then use $\hat{f}_c$ to predict trial points x_t at which $f(x_t) < f(x_c)$. Thus, we will employ the strategy described in Section 1, not once to replace Problem (1), but repeatedly to guide us to a solution of it. We now turn to a description of the models that our oracles will exploit.

3 Computer Experiment Models

Suppose that we have observed $y_i = f(x_i)$ for $i = 1, \ldots, n$. On the basis of this information, we want to construct a global model $\hat{f}$ of f. Such inexpensive surrogates for f will be used by the oracle in the pattern search algorithm to identify promising trial points at which to compute additional function values.

We assume that there is no uncertainty in the $y_i = f(x_i)$, i.e. that no stochastic mechanism is involved in evaluating the objective function. It is then reasonable to require the surrogate function to interpolate these values, i.e. to require that $\hat{f}(x_i) = f(x_i)$. Furthermore, we desire families of surrogate functions that are rich enough to model a variety of complicated objective functions. We are led to consider certain families of models that have been studied in the spatial statistics and computer experiment literatures.

The families of models that we consider are usually motivated by supposing that f is a realization of some (nice) stochastic process. For certain geostatistical applications, this supposition may be quite plausible. In the context of using computer simulations to facilitate the engineering of better product designs, its plausibility is less clear. The high-frequency, low-amplitude oscillations that we have described do resemble the realization of a stochastic process, but the general trends that are our primary concern do not. In any case, we regard the supposition of an underlying stochastic process as nothing more than a convenient fiction. The value of this fiction lies in its power to suggest plausible ways of constructing useful models and we will try to avoid invoking it excessively. When we do invoke it, it should be appreciated that optimality criteria such as BLUP and MLE (see below) are defined with respect to the fictional stochastic process and should not be invested with more importance than the practical utility of the models to which they lead.

Our requirement that $\hat{f}(x_i) = f(x_i)$ will immediately suggest spline interpolation to the approximation theorist and kriging to the geostatistician. In fact, as explicated by Watson (1984) and others, these two well-known methodologies are formally equivalent. Their motivations, however, are somewhat different: whereas the goal of the former is to interpolate the $f(x_i)$ as smoothly as possible, the goal of the latter is to approximate f as accurately as possible. It is evident that the kriging perspective is more germane to our present concerns. The remainder of this section briefly summarizes some relevant facts about kriging in the context of computer experiments. See Sacks, Welch, Mitchell and Wynn (1989), Booker (1994), and Koehler and Owen (1996) for comprehensive surveys of computer experiment methodology.

We begin by assuming that f is a realization of a stochastic process F that is indexed by the continuous parameter set $\Re^p$. We assume that this process has known mean $\mu(x) = 0$ and known covariance function $c(\cdot,\cdot)$, and that each symmetric $p \times p$ matrix $c(s,t)$ is strictly positive definite. Let

$$y = \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix}$$

and for each $x \in \Re^p$ define $b(x) \in \Re^n$ to minimize $E[y'b - F(x)]^2$. Then $\hat{f}(x) = y'b(x)$ is the *best linear unbiased predictor* (BLUP) of $f(x)$ and it is well-known that

$$\hat{f}(x) = y'C^{-1}\vec{c}(x), \tag{2}$$

where C is the symmetric positive definite $n \times n$ matrix $[c(x_i, x_j)]$ and

$$\vec{c}(x) = \begin{bmatrix} c(x_1, x) \\ \vdots \\ c(x_n, x) \end{bmatrix}.$$

This is a simple example of kriging. Notice that kriging necessarily interpolates: since $C^{-1}C = I$ and $\vec{c}(x_j)$ is column j of C,

$$\hat{f}(x_j) = y'C^{-1}\vec{c}(x_j) = y'e_j = y_j = f(x_j).$$

Thus far we have assumed that the stochastic process is known. We now suppose that F is a Gaussian process with mean $\mu(x) = a(x)'\beta$ and covariance function $c(s,t) = \sigma^2 r_\theta(s,t)$. We assume that $a : \Re^p \to \Re^q$ is a known function, that $\beta \in \Re^q$ is an unknown vector, that $\sigma^2 > 0$ is an unknown scalar, and that $r_\theta(\cdot,\cdot)$ is an unknown element of a known family of correlation functions.

Next let A denote the $n \times q$ matrix $[a_j(x_i)]$, let $R(\theta)$ denote the symmetric $n \times n$ matrix $[r_\theta(x_i, x_j)]$, and let

$$r(x;\theta) = \begin{bmatrix} r_\theta(x_1, x) \\ \vdots \\ r_\theta(x_n, x) \end{bmatrix}.$$

Then, for β and θ fixed, the BLUP of $f(x)$ is (cf. equation (2))

$$\hat{f}(x) = a(x)'\beta + (y - A\beta)'R(\theta)^{-1}r(x;\theta). \tag{3}$$

Thus, by varying β and θ, we define a family of interpolating functions from which we can select a specific $\hat{f}$ to model f.

Given a family of interpolating functions defined by (3), we require a sensible way of specifying $(\beta, \sigma^2, \theta)$ and thereby $\hat{f}$. For θ fixed, the *maximum likelihood estimates* (MLEs) of β and σ^2 have explicit formulas:

$$\hat{\beta}(\theta) = \left[A' R(\theta)^{-1} A\right]^{-1} A' R(\theta)^{-1} y$$

and

$$\hat{\sigma}^2(\theta) = \frac{1}{n} \left[y - A\hat{\beta}(\theta)\right]' R(\theta)^{-1} \left[y - A\hat{\beta}(\theta)\right].$$

To compute $\hat{\theta}$, the MLE of θ, it turns out that one must minimize

$$n \log \hat{\sigma}^2(\theta) + \log \det R(\theta) \tag{4}$$

as a function of θ.

It is now evident that, in specifying a family of correlation functions, there is a potential tradeoff between the richness of the family defined by (3) and the ease of maximizing (4). The richer the family of models, the more difficult it may be to select a plausible member of it. Most of the previous research on computer experiments has been concerned with deriving a single model $\hat{f}$ that will be used as a permanent surrogate for f. Understandably, the authors have used rich families with rather complicated correlation functions for which θ is a vector of dimension p or greater. This makes maximizing (4) difficult, but $\hat{\theta}$ need only be computed once. In contrast, we are concerned with deriving a sequence of models that will be used for the sole purpose of guiding our optimization of f. Hence, we are content to sacrifice some flexibility in (3) in order to simplify minimizing (4). In the numerical experiments that we report in Section 5, we used the 1-parameter isotropic correlation function defined by

$$r_\theta(s, t) = \exp\left(-\theta \|s - t\|^2\right), \tag{5}$$

where $\|\cdot\|$ denotes the Euclidean norm on $\Re^p$.

4 Model-Assisted Grid Search

The optimization strategies developed in this report are all predicated on a simple idea, viz. that providing the oracle in Section 2 with an inexpensive surrogate model of the objective function will allow it to more efficiently identify promising trial points and thereby reduce the cost of optimization. The surrogate models will be constructed from known function values by kriging, as described in Section 3. In this section, we describe the interplay between the pattern search algorithm and the kriging models in greater detail.

We begin with an instructive analogy. For unconstrained minimization of smooth objective functions, the numerical algorithms of choice are the *quasi-Newton methods*. An elementary exposition of these methods was provided by Dennis and Schnabel (1983), who emphasized the following interpretation: a quasi-Newton algorithm constructs a quadratic model $\hat{f}_c$ of the objective function f at the current iterate x_c and uses $\hat{f}_c$ to identify a trial point x_t at which it is predicted that $f(x_t) < f(x_c)$. For example, *trust region methods* obtain x_t by (approximately) minimizing $\hat{f}_c$ subject to the constraint that x_t be within a specified neighborhood of x_c. The rationale for the constraint is that the model $\hat{f}_c$, which is usually constructed by computing or approximating the second-order Taylor polynomial of f at x_c, can only be trusted to approximate f locally.

1. Specify an initial grid, Γ_0, on $[a, b]$.
2. Perform an initial computer experiment:
   (a) Select initial design sites $x_1, \ldots, x_N \in \Gamma_0$.
   (b) Compute $f(x_1), \ldots, f(x_N)$.
   (c) Construct $\hat{f}_0$ by kriging.
3. Let $\Gamma_c = \Gamma_0$. Let $x_c = \text{argmin}(f(x_1), \ldots, f(x_N))$. Let $\hat{f}_c = \hat{f}_0$ and $\text{Eval}_c = \{x_1, \ldots, x_N\}$.
4. Do until convergence:
   (a) Do until $\text{Core}(x_c) \subset \text{Eval}_c$:
      i. Apply an optimization algorithm to $\hat{f}_c$ to obtain $x_t \in \Gamma_c \setminus \text{Eval}_c$.
      ii. Compute $f(x_t)$. Let $\text{Eval}_c = \text{Eval}_c \cup \{x_t\}$. Update $\hat{f}_c$.
      iii. If $f(x_t) < f(x_c)$, then let $x_c = x_t$.
   (b) Refine Γ_c.

Figure 2: Model-Assisted Grid Search (MAGS) for minimizing an expensive objective function.

Trust region methods make effective use of simple local models of the objective function. Because we are concerned with situations in which evaluation of the objective function is prohibitively expensive, we are prepared to invest more resources in constructing and optimizing more complicated global models of the objective function.

Similarly, classical response surface methodology, from Box and Wilson (1951) to Myers and Montgomery (1995), constructs local linear or quadratic regression models of a stochastic quadratic objective function. Again, the purpose of these models is to guide the search for a minimizer or maximizer. Glad and Goldstein (1977) also exploited quadratic regression models for optimization, as did Elster and Neumaier (1995) to guide a grid search. Recently, *nonparametric response surface methods* have been proposed in which global models of more complicated objective functions are constructed. This work, e.g. Haaland (1996), is closely related to ours. Frank (1995) surveyed numerous issues that arise when using global models of expensive objective functions to facilitate optimization.

The remainder of this section develops a specific methodology for using global models to minimize expensive objective functions. The essential logic of the methods that we propose is summarized in Figure 2. For simplicity, we assume that the expense of all operations performed on the model(s) is negligible in comparison to the expense of function evaluation(s). Dennis and Torczon (1996) have proposed a general model management strategy that can be employed when it is necessary to balance these expenses. This model management strategy is currently being developed and extended by Serafini (1997) and is capable of accommodating our methods as a special case.

Techniques for designing the initial computer experiment in step 2 do not concern us here, except for two details. First, it is convenient to employ a technique that permits the initial design sites to be selected from the initial grid. Several such techniques were described by Koehler and Owen (1996). Second, we must specify N, the number of initial design sites. A great deal of numerical experimentation will be required before it becomes possible to suggest useful guidelines for the choice of N. Because we are concerned with problems for which sequential optimization is practicable, considerably less than the entire budget of functions evaluations (V) will be invested in initial design, i.e. we will choose $N \ll V$. At present, we prefer to choose $N \geq p+1$, thereby permitting at least a simplex design.

Because evaluation of the objective function is expensive, it seems sensible to cache the function values that have been computed. We denote the current set of points at which f has been evaluated by Eval_c, which in step 3 we initialize to comprise the initial design sites. For any point $x \in \Gamma_c$, we denote by $\mathrm{Core}(x)$ the core pattern of points adjacent to x. Then, in loop 4(a), $\mathrm{Core}(x_c) \subset \mathrm{Eval}_c$ is precisely the condition required by the convergence theory for pattern search methods that must be satisfied before the current grid can be refined. Notice that this condition must eventually obtain, i.e. loop 4(a) cannot repeat indefinitely, because Γ_c is finite and we only consider trial points $x_t \in \Gamma_c$ at which f has not yet been evaluated. Thus, the logic of loop 4 is to (a) search for points of improvement on the current grid until (b) we replace the current grid with a finer grid. Loop 4 repeats until we have exhausted our budget of objective function evaluations.

It is within loop 4(a) that we exploit the models obtained by kriging. Step 4(a)(i) produces a trial point x_t at which f is evaluated in the hope that $f(x_t) < f(x_c)$. As soon as $f(x_t)$ has been computed, we add x_t to the set Eval_c and update the current model $\hat{f}_c$. It is not entirely clear how best to update $\hat{f}_c$. One could completely refit the family of models to the new set of function values, but one might decline to re-estimate certain parameters if one believed that the value of updating them did not justify the expense.

The generality of the convergence theory for pattern search methods permits us to be rather vague in specifying step 4(a)(i). This ambiguity is desirable because it encompasses a great many possibilities deserving consideration. Thus, we can search for a local minimizer of $\hat{f}_c$ using whatever algorithm we prefer, e.g. quasi-Newton, steepest descent, pattern search, etc. We can start our search from whatever point we prefer or we can use multiple starting points and pursue multiple searches before determining x_t. We can even search for a global minimizer of $\hat{f}_c$ if we are so inclined. Furthermore, we can terminate our search whenever we please. (We envision searching until we near a local minimizer of $\hat{f}_c$, but we can terminate sooner if the search becomes expensive. Trade-offs between the cost of evaluating the objective function and the cost of constructing and optimizing the models can be mediated by the model management strategy proposed by Dennis and Torczon (1996) and developed by Serafini (1997).) The only requirement is that we eventually return a trial point x_t that belongs to the current grid and at which f has not yet been evaluated.

Most search strategies for a desirable x_t will not restrict attention to the current grid. Because we require $x_t \in \Gamma_c$, we suggest terminating the search when step lengths become appreciably smaller than the minimal distance between grid points and choosing x_t to be a grid point that is near the terminal iterate of the search. There are various plausible ways of implementing this suggestion. We might prefer the nearest grid point or we might prefer a nearby grid point at which the model predicts greater decrease. Because we require $x_t \notin \mathrm{Eval}_c$, we might not actually select the nominally preferred grid point. Similarly, if we are impatient to refine the grid, we might select $x_t \in \mathrm{Core}(x_c)$ even when a nonadjacent point is nominally preferred.

5 Numerical Experiments

We now describe some numerical experiments intended to illustrate the viability of the methods proposed in Section 4. These experiments were performed on a Toshiba Satellite 100CS notebook computer with a 75MHz Pentium processor, using S-PLUS for Windows software. In what follows, univariate optimization (for parameter estimation) was performed using the S-PLUS function *opti-*

Minimizer	$f(x)$	Frequency
(0,-10)	3	38
(-6,-4)	30	44
(18,2)	84	8
(12,8)	840	10

Table 1: Minima of the rescaled Goldstein-Price objective function on $[-20, 20] \times [-20, 20]$ found by 100 quasi-Newton searches.

mize, an implementation of Brent's (1973) safeguarded polynomial interpolation procedure. When finite-difference quasi-Newton methods for multivariate optimization were employed (typically for minimizing models), they were performed using the S-PLUS function *nlminb*, an implementation of a trust-region method for bound-constrained optimization developed by Gay (1983, 1984).

The objective function used in our experiments was a rescaled version of an eighth-order polynomial of $p = 2$ variables constructed by Goldstein and Price (1971). One of the test problems employed by Dixon and Szegö (1978) in their study of global optimization was to minimize the Goldstein-Price polynomial in the feasible region $[-2, 2] \times [-2, 2]$. We rescaled this problem so that the feasible region was $[-20, 20] \times [-20, 20]$.

The Goldstein-Price polynomial is a complicated function that has four minimizers and is not easily modelled. Because it is a polynomial, the minimizers are easily located by a finite-difference quasi-Newton method. To estimate the relative sizes of their basins of attraction, we drew 100 points from a uniform distribution on the feasible region and started *nlminb* from each point. The results are summarized in Table 1.

The models that we employed are a special case of the family specified by (3). We set $q = 1$, so that β is a scalar parameter, and $a(x) \equiv 1$. We used the correlation function specified by (5) and estimated the scalar parameter θ by applying *optimize* to (4).

We implemented two strategies for derivative-free optimization with a small value of V, the permitted number of objective function evaluations. First, we implemented a DACE strategy with $V = 11, 16, 21, 26$. For these procedures, we constructed a model, $\hat{f}$, from $N = V - 1$ design sites obtained by Latin hypercube sampling. We then minimized $\hat{f}$ by a finite-difference quasi-Newton method, starting from the initial design site with the smallest objective function value. The objective function was then evaluated at the minimizer of $\hat{f}$ so obtained and the smallest of the V function values was recorded.

Second, we implemented the MAGS strategy described in Section 4 with $V = 11, 16$. For these procedures, it is first necessary to specify an initial grid. This involves a trade-off: coarse grids tend to safeguard against unlucky initial designs, whereas fine grids permit more function evaluations before it becomes necessary to evaluate f at a core pattern of points in order to refine the grid. Our choice of a family of grids was further complicated by the fact that the minimizers of our objective function occur on the integer lattice in $\Re^2$. To avoid using grids that contain the minimizers, we selected an initial grid of the form

$$x_i = -20 + j_i \pi / 2,$$

for $j_i = 0, 1, 2, \ldots$. Subsequent grids, which were rarely required, were obtained by halving the current distance between adjacent grid points. To obtain an initial design on the initial grid, we first obtained $N = 5$ points by Latin hypercube sampling, then moved each point to the nearest grid point. The initial model was constructed from this design.

Percentile	DACE11	DACE16	DACE21	DACE26	MAGS11	MAGS16
Minimum	3.41	5.29	5.02	3.26	5.77	3.43
5%	9.60	11.99	10.64	9.89	5.77	5.77
10%	15.63	13.74	17.16	14.42	6.78	5.77
25%	37.26	30.83	26.04	24.75	28.98	6.78
50%	116.75	71.61	51.69	41.70	43.89	30.48
75%	367.95	122.35	99.96	86.33	87.78	64.55
90%	601.86	240.92	190.71	154.20	343.21	101.63
95%	951.55	423.13	230.89	281.84	529.02	135.22
Maximum	1027.52	1243.90	774.91	349.43	1617.61	885.32

Table 2: Percentiles of smallest values of the Goldstein-Price objective function found by six optimization strategies, each replicated 100 times.

To obtain a trial point, x_t, from the current iterate, x_c, a finite-difference quasi-Newton method was started from x_c to obtain a minimizer, $\hat{x}$, of the current model, $\hat{f}_c$. Let $\bar{x}$ denote the grid point nearest $\hat{x}$. If the objective function had not previously been evaluated at $\bar{x}$, then we set $x_t = \bar{x}$; otherwise, we set x_t equal to the point in Core($\bar{x}$) nearest $\hat{x}$. Each time that a new function value was obtained, we re-estimated all three of the model parameters, $(\beta, \sigma^2, \theta)$. This process was repeated until V function evaluations had been performed, at which time the smallest function value was recorded.

Each of the six procedures described above was replicated 100 times. The results are summarized in Table 2, from which several interesting conclusions can be drawn. First, we note that each strategy did indeed do better when permitted more function evaluations. Thus, DACE with $V = 16$ function evaluations usually outperformed DACE with $V = 11$ function evaluations and MAGS with $V = 16$ function evaluations usually outperformed MAGS with $V = 11$ function evaluations. This result is not surprising.

Second, we note that MAGS typically found smaller function values with fewer function evaluations than did DACE. The crucial comparisons are between DACE and MAGS with $V = 11$ function evaluations and between DACE and MAGS with $V = 16$ function evaluations. In each case, typical performance decisively favors MAGS. For example, the median best function value found by MAGS11 is less than 40 percent of the median best value found by DACE11. Similarly, the median best function value found by MAGS16 is less than 45 percent of the median best value found by DACE16.

The DACE experiments with $V = 21, 26$ were included to benchmark the performance of MAGS. Except for the extreme upper percentiles, the distribution of best function values found by MAGS with $V = 11$ is strikingly similar to the distribution of best function values found by DACE with $V = 26$ function evaluations. Thus, another way to state the case for MAGS is to observe that DACE typically required more than twice as many function evaluations to match the performance of MAGS11. Such observations confirm the central thesis of this report, that optimization is most efficiently accomplished by iterative methods. This is a familiar issue in statistics that is often raised when comparing Taguchi and response surface methods, e.g. in the panel discussion edited by Nair (1992) and by Trosset (1996). Our results strengthen the cases made by Frank (1995), Booker et al. (1995), Booker et al. (1996) and Booker (1996) for using sequential modeling strategies to optimize expensive functions.

We note that despite its generally superior performance, MAGS occasionally disappoints. For example, the lower 75 percent of the distributions of best values found by DACE with $V = 26$ and by MAGS with $V = 11$ are remarkably similar, but the former's upper 10 percent is markedly better than the latter's. MAGS with $V = 16$ found a smallest function value less than or equal to 142.70 on 97 occasions, but found smallest values of 688.75, 855.64, and 885.32 on the other three. This anomalous phenomenon seems to result from unfortunate initial designs. With only $N = 5$ design sites, Latin hypercube sampling on a grid occasionally results in transparently bad designs, e.g. five collinear sites, from which MAGS has difficulty recovering. To safeguard against this possibility, we recommend using more sophisticated procedures for determining the initial design.

6 Conclusions and Future Work

The results presented in Section 5 suggest that the potential value of the methods that we have proposed for optimizing expensive objective functions is considerable. Of course, comprehensive numerical experiments on a variety of objective functions in a variety of dimensions will be required to fully appreciate the advantages and disadvantages of these methods. We plan to undertake such investigations in future work. We conclude this report by describing some modifications of MAGS that we envision for these investigations.

We begin by noting that the design of the current model, $\hat{f}_c$, is sequential, comprising the initial design sites $x_1, \ldots, x_N$ and the trial sites x_t subsequently identified in 4(a)(i) in Figure 2. At present, the initial design sites are selected according to the principles of experimental design without regard to optimization, whereas the subsequent trial sites are determined by an optimization algorithm without regard to the design of experiments. Because the sequence of trial points generated by an optimization algorithm tends to cluster in promising regions, this sequence is rarely space-filling and is not likely to be a good experimental design. Moreover, especially during the early stages of optimization, greater gains are likely to come from improving the fit of $\hat{f}_c$ to f than from accurately identifying a minimizer of $\hat{f}_c$. Our present implementation of MAGS is a greedy algorithm in the sense that it tries to minimize $\hat{f}_c$ without concern for the fit of the subsequent model, $\hat{f}_+$.

The desirability of balancing the concerns of numerical optimization and the concerns of experimental design was recognized by Frank (1995), who proposed a dichotomous search strategy. Given an optimization criterion, e.g. the objective function, and a design criterion, one obtains some fraction of new design sites using one criterion and the balance using the other. An implementation of this Balanced Local-Global Search (BLGS) strategy was described by Booker et al. (1995) and employed by Booker (1996) and Booker et al. (1996).

In contrast to the dichotomous BLGS strategy, we envision modifying 4(a)(i) as follows: apply an optimization algorithm to a merit function, Φ_c, to obtain $x_t \in \Gamma_c \setminus \mathrm{Eval}_c$. The merit function should be specified so as to balance the potentially competing goals of finding sites at which $\hat{f}_c$ is small and choosing good design sites. Notice that this modification in no way affects the convergence theory for MAGS—it is a purely empirical matter whether or not it improves the performance of the algorithm.

To illustrate what we have in mind, we again invoke the fiction that the objective function, f, is a realization of a stochastic process. Under the assumptions delineated in Section 3, the mean squared error of (3) for predicting $f(x)$ is

$$\mathrm{MSE}\left[\hat{f}(x)\right] = \sigma^2 \left(1 - [a(x)', r(x;\theta)'] \begin{bmatrix} 0 & A' \\ A & R(\theta) \end{bmatrix}^{-1} \begin{bmatrix} a(x) \\ r(x;\theta) \end{bmatrix} \right).$$

As detailed by Sacks, Welch, Mitchell and Wynn (1989), this function plays a fundamental role in several approaches to optimal design. It is also the design criterion employed by Booker et al.

(1995) and Booker (1996) in BLGS. Our idea is to construct a merit function that encourages the selection of trial points at which the mean squared error is large, i.e. at which we believe that less is known about the objective function. This strategy has a great deal in common with the Sequential Design for Optimization (SDO) algorithm for global optimization proposed by Cox and John (1996).

The mean squared error function can be estimated by replacing the parameters $(\beta, \sigma^2, \theta)$ with their MLEs, resulting in an expression that we denote by

$$\widehat{\mathrm{MSE}}\left[\hat{f}(x)\right].$$

Then a natural family of merit functions comprises those of the form

$$\Phi_c(x) = \hat{f}_c(x) - \rho_c \widehat{\mathrm{MSE}}\left[\hat{f}_c(x)\right],$$

where $\rho_c \geq 0$. We anticipate that a great deal of numerical experimentation will be required to determine useful choices of the ρ_c.

We also note that our family of kriging models was derived under the assumption of a stationary stochastic process. In fact, it seems rather implausible that the same correlations will obtain throughout the feasible set, so that we likely will prefer different values of θ for different regions of $[a, b]$. This poses a problem, because the $\hat{f}_c$ are global models. As we descend into the basin of a local minimizer, we would like to use a value of θ that is suited to that basin, not a value of θ that was determined by a global compromise.

An obvious way of addressing this difficulty is to implement the "zoom-in" process proposed by Frank (1995). To do so, we occasionally modify the feasible set, restricting it to a much smaller set in which a minimizer is deemed to lie. Then the current model, $\hat{f}_c$, is determined by the design sites in, and restricted in domain to, the current feasible set, $[a_c, b_c]$. From a theoretical perspective, this is a delicate strategy that must be carefully implemented in order to preserve the guaranteed convergence that has thus far been inherited from the standard theory for pattern search methods. However, properly implemented, updating the feasible set may permit a useful localization of the kriging models.

Assuming that we are prepared to replace the current feasible set with a subset of it, when should we do so? When step 4(a)(i) begins to produce trial points $x_t \in \mathrm{Core}(x_c)$, one might guess that x_c is near a local minimizer and that the algorithm is preparing to refine the current grid. As previously noted, this can be an expensive undertaking, requiring as many as $2p$ evaluations of f. At this stage, an interactive user might very well instruct the algorithm to refine the grid *without* actually evaluating f on $\mathrm{Core}(x_c)$, reasoning that an asymptotic convergence theory may be of little relevance when one can afford only a small number of iterations.

Alternatively, one might interpret the same scenario as a signal to recalibrate the problem. First, one would replace the current feasible set with a smaller one. Second, instead of performing $2p$ function evaluations on $\mathrm{Core}(x_c)$—a set of points that is *not* a good design—one would replace the current grid with a finer grid, design an experiment for the new grid, and proceed. At present, there is no theory to justify these tactics, but it seems quite plausible that they will result in practical improvements of an already-promising procedure.

Acknowledgments

This research evolved from an ongoing collaboration with John Dennis and David Serafini (Rice University); Andrew Booker, Paul Frank and Greg Shubin (Boeing Company); and Andrew Conn (IBM Corporation). Although we have been profoundly influenced by the ideas of our collaborators, some of the opinions expressed in this report may not be shared by all members of the collaboration.

References

Barthelemy, J.-F. M. and Haftka, R. T. (1993). Approximation concepts for optimum structural design—a review. *Structural Optimization*, 5:129–144.

Booker, A. J. (1994). DOE for computer output. Technical Report BCSTECH-94-052, Boeing Computer Services, Seattle, WA.

Booker, A. J. (1996). Case studies in design and analysis of computer experiments. In *Proceedings of the Section on Physical and Engineering Sciences*. American Statistical Association.

Booker, A. J., Conn, A. R., Dennis, J. E., Frank, P. D., Serafini, D., Torczon, V., and Trosset, M. (1996). Multi-level design optimization. Technical Report ISSTECH-95-031, Boeing Information & Support Services, Seattle, WA. Boeing/IBM/Rice Collaborative Project 1996 Final Report.

Booker, A. J., Conn, A. R., Dennis, J. E., Frank, P. D., Trosset, M., and Torczon, V. (1995). Global modeling for optimization. Technical Report ISSTECH-95-032, Boeing Information & Support Services, Seattle, WA. Boeing/IBM/Rice Collaborative Project 1995 Final Report.

Box, G. E. P. and Wilson, K. B. (1951). On the experimental attainment of optimum conditions. *Journal of the Royal Statistical Society, Series B*, 13:1–45. Includes discussion.

Brent, R. (1973). *Algorithms for Minimization Without Derivatives*. Prentice-Hall, Englewood Cliffs, NJ.

Cox, D. D. and John, S. (1996). SDO: A statistical method for global optimization. In Alexandrov, N. and Hussaini, M. Y., editors, *Multidisciplinary Design Optimization: State of the Art*. SIAM, Philadelphia.

Dennis, J. E. and Schnabel, R. B. (1983). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, New Jersey.

Dennis, J. E. and Torczon, V. (1991). Direct search methods on parallel machines. *SIAM Journal on Optimization*, 1:448–474.

Dennis, J. E. and Torczon, V. (1996). Managing approximation models in optimization. In Alexandrov, N. and Hussaini, M. Y., editors, *Multidisciplinary Design Optimization: State of the Art*. SIAM, Philadelphia.

Dixon, L. C. W. and Szegö, G. P. (1978). The global optimisation problem: An introduction. In Dixon, L. C. W. and Szegö, G. P., editors, *Towards Global Optimisation 2*, pages 1–15. Elsevier North-Holland, New York.

Elster, C. and Neumaier, A. (1995). A grid algorithm for bound constrained optimization of noisy functions. *IMA Journal of Numerical Analysis*, 15:585–608.

Frank, P. D. (1995). Global modeling for optimization. *SIAG/OPT Views-and-News*, (7):9–12.

Gay, D. M. (1983). Algorithm 611. Subroutines for unconstrained minimization using a model/trust-region approach. *ACM Transactions on Mathematical Software*, 9:503–524.

Gay, D. M. (1984). A trust region approach to linearly constrained optimization. In Lootsma, F. A., editor, *Numerical Analysis. Proceedings, Dundee 1983*, pages 171–189, Berlin. Springer.

Glad, T. and Goldstein, A. (1977). Optimization of functions whose values are subject to small errors. *BIT*, 17:160–169.

Goldstein, A. A. and Price, J. F. (1971). On descent from local minima. *Mathematics of Computation*, 25:569–574.

Haaland, P. D. (1996). Nonparametric response surface methods. In *Abstracts: Summaries of Papers Presented at the 1996 Joint Statistical Meetings in Chicago, Illinois*, page 324.

Koehler, J. R. and Owen, A. B. (1996). Computer experiments. In Ghosh, S. and Rao, C. R., editors, *Handbook of Statistics, Volume 13*, pages 261–308. Elsevier Science, New York.

Lewis, R. M. and Torczon, V. (1996a). Pattern search algorithms for bound constrained minimization. Technical Report 96-20, ICASE, NASA Langley Research Center, Hampton, VA. To appear in *SIAM Journal on Optimization*.

Lewis, R. M. and Torczon, V. (1996b). Rank ordering and positive bases in pattern search algorithms. Technical Report 96-71, ICASE, NASA Langley Research Center, Hampton, VA.

McKinnon, K. I. M. (1996). Convergence of the Nelder-Mead simplex method to a non-stationary point. Technical Report MS 96-006, Department of Mathematics & Statistics, University of Edinburgh, Edinburgh, Scotland.

Myers, R. H. and Montgomery, D. C. (1995). *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. John Wiley & Sons, New York.

Nair, V. N. (1992). Taguchi's parameter design: A panel discussion. *Technometrics*, 34:127–161.

Nelder, J. A. and Mead, R. (1965). A simplex method for function minimization. *Computer Journal*, 7:308–313.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical Science*, 4:409–435. Includes discussion.

Serafini, D. (1997). *A Model Management Framework for Nonlinear Optimization of Computationally Expensive Functions*. PhD thesis, Rice University, Houston, TX. In progress.

Torczon, V. (1997). On the convergence of pattern search methods. *SIAM Journal on Optimization*, 7:1–26.

Trosset, M. W. (1996). Taguchi and robust design. Technical Report 96-31, Department of Computational & Applied Mathematics, Rice University, Houston, TX.

Watson, G. S. (1984). Smoothing and interpolation by kriging and with splines. *Mathematical Geology*, 16:601–615.

Welch, W. J. and Sacks, J. (1991). A system for quality improvement via computer experiments. *Communications in Statistics—Theory and Methods*, 20:477–495.