Model-Assisted Pattern Search Methods for Optimizing Expensive Computer Simulations[1]

Christopher M. Siefert, Department of Computer Science, University of Illinois, Urbana, IL 61801. E-mail: siefert@uiuc.edu

Virginia Torczon, Department of Computer Science, College of William & Mary, P.O. Box 8795, Williamsburg, VA 23187-8795. E-mail: va@cs.wm.edu

Michael W. Trosset, Department of Mathematics, College of William & Mary, P.O. Box 8795, Williamsburg, VA 23187-8795. E-mail: trosset@math.wm.edu

Abstract

The design and analysis of computer experiments (DACE) usually envisions performing a single experiment, then replacing the expensive simulation with an approximation. When the simulation is a nonlinear function to be optimized, DACE may be inefficient, and sequential strategies that synthesize ideas from DACE and numerical optimization may be warranted. We consider several such strategies within a unified framework in which sequential approximations constructed by kriging are used to accelerate a conventional direct search method. Computational experiments reveal that hybrid strategies outperform both DACE and traditional pattern search.

1 Introduction

Computer experiments are planned evaluations of deterministic functions. In a typical application, the function f incorporates an expensive computer simulation of a physical phenomenon, e.g. the aeroelastic and dynamic response simulation of a helicopter rotor blade studied by Booker (1996). The notion that f is expensive to evaluate can be formalized by imposing an upper bound, V, on the number of evaluations of f that are practicable. When V is restrictive, then a computer experiment is performed for the purpose of efficiently obtaining information about f. Comprehensive surveys of computer experiment methodology have been provided by Sacks, Welch, Mitchell, and Wynn (1989) and by Koehler and Owen (1996).

In design optimization, f is a performance criterion and the arguments of f are design parameters. The goal is to discover better products, e.g. rotor blades, by varying product design. We formalize this goal by considering the problem of minimizing $f : \Re^p \to \Re$ subject to bound constraints, i.e.

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

where $\ell, x, u \in \Re^p$ and we write $x \in [\ell, u]$ to denote $\ell_i \leq x_i \leq u_i$ for $i = 1, \ldots, p$. We are concerned with special cases of Problem (1) for which V is too small for the use of standard numerical optimization algorithms to be practicable. Notice that the present study restricts attention to simple bound constraints, although actual engineering design problems often involve more complicated constraints.

When trying to minimize an objective function f that is too expensive for standard algorithms to succeed, a common practice is 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 algorithm to minimize $\hat{f}$, and evaluate f at the site thus obtained.) Although many authors have remarked that one reason for performing a computer experiment may be to facilitate optimizing f, their remarks are usually followed by the same suggestion, that one might minimize $\hat{f}$ instead of f. Frank (1995) offered an optimizer's perspective on this practice, suggested that the "minimalist approach" of minimizing a single $\hat{f}$ is not likely to yield satisfactory results, and proposed several sequential strategies as alternatives. Booker (1996) studied several industrial applications of this practice and two alternative approaches. Trosset (1998) argued that, because optimizing computer simulations is usually less expensive than optimizing industrial processes, the former should entail at least as much sequential experimentation as response surface methodology. The

[1]This research was supported by a Batten Scholarship from the College of William & Mary, by the William & Mary Endowment Association, by the National Science Foundation under Grants CCr-9734044 and EIA-9712718, by the National Aeronautics and Space Administration under NASA Contract No. NAS1-97046, and by a gift from the Upstream Strategic Research Center of the Mobil Technology Company.

merits of sequential experimentation versus one massive experiment are often discussed when comparing response surface and Taguchi methods for robust design, as in the panel discussion edited by Nair (1992) and by Trosset (1996).

The first optimization strategy to appear in the literature on computer experiments was proposed by Welch and Sacks (1991) for the (more complicated) purpose of quality improvement. Adapting their generic seven-step strategy to Problem (1) results in the prescription displayed in Figure 1, which has gained considerable currency in the engineering design community. Notice that Step 6 admits the possibility of sequential experimentation, but on the basis of the adequacy of the global approximation $\hat{f}$ rather than the behavior of f at a minimizer of $\hat{f}$.

1. Postulate a model for f.
2. Plan and perform an initial computer experiment.
3. Construct $\hat{f}$. "*Before proceeding further, the adequacy of the predictors should be assessed and, if necessary, improved.*"
4. Perform diagnostics.
5. Perform a tentative optimization of $\hat{f}$ to obtain a putative minimizer x^*.
6. "*If Step 3 indicated that the predictor is not accurate enough yet,*" then restrict the feasible region to a subregion deduced from Steps 4 and 5, and return to Step 2.
7. Evaluate $f(x^*)$.

Figure 1: The generic optimization strategy of Welch and Sacks (1991).

In contrast, a naive sequential optimization strategy is displayed in Figure 2. This strategy constructs an initial approximation of f, minimizes the approximation to obtain a new design site x_t, evaluates $f(x_t)$, uses $f(x_t)$ to update the approximation, and continues. It is our impression that this simple strategy has already been assimilated into the computer experiment folklore, but it has not been adequately discussed in the computer experiment literature.

Closely related to the naive strategy in Figure 2. are several heuristic search strategies for global optimization. Both the Sequential Design Optimization algorithm proposed by Cox and John (1997) and the expected improvement algorithm developed in Schonlau and Welch (1996), Schonlau, Welch, and Jones (1997, 1999), and Jones, Schonlau, and Welch (1999) adapt methods for designing and analyzing computer

1. Postulate a model for f.
2. Perform an initial computer experiment:
   (a) Select initial design sites $x_1, \ldots, x_N \in [a, b]$.
   (b) Compute $f(x_1), \ldots, f(x_N)$.
   (c) Construct $\hat{f}_0$ by kriging.
3. Let $x_c = \text{argmin}(f(x_1), \ldots, f(x_N))$ and $\hat{f}_c = \hat{f}_0$.
4. Do until convergence:
   (a) Apply an optimization algorithm to $\hat{f}_c$ to obtain x_t.
   (b) Compute $f(x_t)$ and update $\hat{f}_c$.
   (c) If $f(x_t) < f(x_c)$, then let $x_c = x_t$.

Figure 2: A naive sequential optimization strategy.

experiments to the task of global optimization. The latter authors have emphasized that "the key to using response surfaces for global optimization lies in balancing the need to exploit the response surface (by sampling where the surface is minimized) with the need to improve the surface (by sampling where prediction error may be high)." As noted by Trosset (1998), the same needs must be balanced when developing efficient local search strategies. Torczon and Trosset (1998) provide a simple example that illustrates this theme.

This paper reports some of our computational experiments with different optimization strategies for solving Problem (1) when V is small. Fundamental questions include: Is sequential experimentation better than performing one massive experiment? Do strategies that use computer experiments improve on conventional strategies for numerical optimization? How should a sequential strategy select new design sites? To answer these questions we will first embed several different strategies in a common framework, described in Section 2. In Section 3 we describe our numerical experiments, present some results, and offer some conclusions.

2 The MAPS Framework

The output of a complicated computer simulation is often 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.

None of the optimization strategies described in Section 1 are guaranteed to converge to even a local minimizer of Problem (1). Just as we would prefer to use consistent estimation procedures even with small samples, so would we prefer to use provably convergent optimization algorithms even when we expect to terminate because we have expended our small budget of function evaluations. Hence, we rely on a class of derivative-free methods for which a convergence theory exists, the *pattern search methods* explicated by Torczon (1997) for the case of optimization without constraints and extended by Lewis and Torczon (1999) to the respective cases of optimization with bound constraints. Torczon and Trosset (1997) provide an elementary introduction to pattern search methods and an account of their origins in the literature on response surface methodology; Lewis, Torczon, and Trosset (1998) provide an accessible exposition of the convergence theory.

Unfortunately, pattern search methods typically require large numbers of function evaluations. Because our budget, V, is severely limited, we want to use these evaluations as efficiently as possible. Dennis and Torczon (1997) suggested the possibility of using low-fidelity simulations of f to guide a pattern search for a minimizer of f. This idea was extended to a comprehensive framework for managinging surrogate information by Booker et al. (1999), who used previous function values to construct a current global approximation, $\hat{f}_c$, of f, then used $\hat{f}_c$ to predict trial points x_t at which $f(x_t) < f(x_c)$. This optimization strategy is similar to the naive strategy displayed in Figure 2, but with the guarantees provided by pattern search convergence theory.

Trosset and Torczon (1997) described a particular implementation of Booker et al.'s more general surrogate management framework. This implementation was subsequently extended and refined by Siefert (2000), who called it Model-Assisted Pattern Search (MAPS). The logic of MAPS, detailed in Figure 3, can accommodate natural variants of the optimization strategies described in Section 1, thereby affording meaningful comparisons of six different strategies:

1. Postulate a model for f, i.e. specify the family of functions from which $\{\hat{f}_k\}$ will be selected.
   Specify an initial grid G_0 that conforms to the feasible region $\ell \le x \le u$.
   Choose $N < V$ and set $k = 0$.
2. Perform an initial computer experiment:
   (a) Select initial design sites $x^1, \ldots, x^N \subset G_0$.
   (b) Compute $f(x^1), \ldots, f(x^N)$.
   (c) Construct $\hat{f}_0$, the initial approximation, by kriging.
   (d) Specify S_0, the initial search criterion.
3. Set $x_0 = \text{argmin } \{f(x^1), \ldots, f(x^N)\}$.
   Set $\text{Eval}_0 = \{x^1, \ldots, x^N\}$.
4. Terminate if $N + k \ge V$. Otherwise:
   (a) Apply an optimization algorithm to S_k to obtain $x^t \in G_k \setminus \text{Eval}_k$.
   (b) Compute $f(x^t)$. Set $\text{Eval}_k = \text{Eval}_k \cup \{x^t\}$.
   (c) If $f(x^t) < f(x_k)$, then $x_k = x^t$.
   Else if $\text{Core}(x_k) \subset \text{Eval}_k$, then refine G_k.
   (d) Update $\hat{f}_k$ and then S_k.
   (e) Set $k = k + 1$.

Figure 3: Model-Assisted Pattern Search.

1—DACE Our implementation of the strategy described in Figure 1. We postulate that each f was a realization of a second-order stationary Gaussian stochastic process with unknown mean $\beta \in \Re$ and covariance function of the form $c(s,t) = \sigma^2 r_\theta(s,t)$, where $\sigma^2 > 0$ was unknown and r_θ was an unknown element of the family of isotropic correlation functions

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

We select $V - 1$ design sites by Latin hypercube (LHC) sampling, then construct $\hat{f}$ by kriging. We estimate the unknown parameters $(\beta, \sigma^2, \theta)$ by the method of maximum likelihood, set

$$\begin{aligned} R(\theta) &= [r_\theta(x_i, x_j)], \\ r(x;\theta) &= [r_\theta(x_i, x)], \end{aligned}$$

and compute

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

Multiple compass searches (see below) are used to locate a putative global minimizer of $\hat{f}$.

2—Compass A simple pattern search, described by Lewis, Torczon, and Trosset (1998), started from a randomly sampled $x_0 \in G_0$. The set $\mathrm{Core}(x_k)$ comprises the $\leq 2^p$ feasible grid points that are adjacent (in the coordinate directions) to x_k. If there is no $x^t \in \mathrm{Core}(x_k)$ for which $f(x^t) < f(x_k)$, then the mesh of the current grid is halved.

3—LHC-Compass A variant of compass search. For $N < V$, we select $x^1, \ldots, x^N \in G_0$ by LHC sampling, then start a compass search from the x^i at which $f(x^i)$ was minimal.

4—MAGS A variant of the naive sequential strategy described in Figure 2. The family of models is identical to DACE, the initial design sites are selected by LHC sampling, the current search criterion $S_k = \hat{f}_k$, and the underlying pattern search is Compass. This strategy was implemented by Trosset and Torczon (1997), except that we now use a compass search to minimize $\hat{f}_k$.

5—MAPS-SDO A variant of the SDO algorithm proposed by Cox and John (1997) for global optimization. MAPS-SDO is identical to MAGS, except that the current search criterion is

$$S_k(x) = \hat{f}_k(x) - w_c \sqrt{\widehat{\mathrm{MSE}}\left[\hat{f}_k(x)\right]},$$

where $w_c = 2$ and

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

As noted in Section 1, this strategy balances the need to exploit $\hat{f}_k$ and the need to construct a better $\hat{f}_{k+1}$.

6—MAPS-Dynamic A variant of MAPS-SDO. Instead of setting $w_c = 2$, w_c is dynamically adjusted according to a scheme proposed by Siefert (2000).

3 Numerical Experiments

We studied two sets of objective functions, one comprising five classic objective functions from the global optimization literature, one comprising forty randomly generated objective functions. The Shekel and Hartman families of functions are described by Dixon and Szegö (1978); we used three Shekel functions ($n = 4$, $m = 5, 7, 10$), denoted g_m, and two Hartman functions ($n = 3, 6$, $m = 4$), denoted h_n. Here $n = p$ is dimension and m is a parameter that controls the number of local minimizers. The randomly generated functions were produced by drawing pseudorandom samples of spatially correlated function values, interpolating the observed values by kriging, then adding a trend. See Trosset (1999), Padula (2000), and Trosset and Padula (2000) for details and software. For each of $p = 2, 3, 4, 5$, we generated five samples of function values using isotropic correlation functions of the form (2). From each sample, we obtained two functions, by kriging and then adding either a constant or a quadratic trend. We identified the local minima of each objective function by starting a large number of pattern searches from randomly chosen points and running each until it converged.

By a *run* we mean a randomly initialized optimization strategy applied to an objective function f with a budget $V = 50$ function evaluations. Each run returns a putative minimum, the best value of f observed during the run. By observing 1000 runs, we obtained a random sample of 1000 putative minima. Our preferred way of displaying these samples, described by Trosset and Padula (2000), is to construct nonparametric estimates of the corresponding probability density functions, e.g. by kernel methods. These *density plots* clearly reveal many salient features of algorithmic performance: the probability of finding a small function value, the tendency to converge to a local (or global) minimizer, etc. Density plots also permit striking visual comparisons of different algorithms; unfortunately, space precludes presenting them in this forum.

Here we focus on two specific attributes of each run: absolute success in finding a global minimum and relative success in finding a small function value. Absolute success means that the run returned a function value less than $(y_1 + y_2)/2$, where y_1 is the global minimum of f and y_2 is the smallest nonglobal minimum. Relative success means that the run returned a function value less than the 25th percentile of the 1000 putative minimizers returned by DACE on the same objective function. For the Shekel and Hartman objective functions, the numbers of absolute and relative successes are displayed in Tables 1 and 2. We summarize similar tables, included in Appendices C and B of Siefert (2000), for the 40 randomly generated objective functions.

For $p = 2$, DACE often produced absolute successes (259–583), always more often than Compass, usually more often than MAGS (9 of 10 functions), and sometimes more than LHC-Compass (5 of 10). However, MAPS-SDO typically (7 of 10) and MAPS-Dynamic always produced more absolute successes than DACE. Compass (2 of 10) and MAGS (3 of

	g_5	g_7	g_{10}	h_3	h_6
DACE	0	0	0	629	0
Compass	28	25	19	525	0
LHC-Compass	13	17	12	812	0
MAGS	261	213	181	897	566
MAPS-SDO	260	293	280	954	664
MAPS-Dynamic	276	374	369	983	621

Table 1: Absolute success: number of runs in 1000 trials that each optimization strategy bettered the average of the two smallest minima for the Shekel (g) and Hartman (h) objective functions.

	g_5	g_7	g_{10}	h_3	h_6
DACE	250	250	250	249	250
Compass	989	997	995	385	172
LHC-Compass	987	991	985	467	169
MAGS	1000	1000	1000	897	960
MAPS-SDO	1000	1000	1000	954	1000
MAPS-Dynamic	1000	1000	999	981	999

Table 2: Relative success: number of runs in 1000 trials that each optimization strategy bettered the 25th percentile of 1000 putative minimizers returned by DACE for the Shekel (g) and Hartman (h) objective functions.

10) rarely produced more than 250 relative successes; LHC-Compass (6 of 10) did somewhat better. MAPS-SDO (245–855) and MAPS-Dynamic (389–927) typically produced considerably more than 250 relative successes.

For $p = 3$, DACE produced fewer than 10 absolute successes for 8 of the 10 objective functions, 39–40 for the other two. Both pattern searches always produced more absolute successes than DACE and usually bettered q_{25}, the 25th percentile of DACE. For one objective function, Compass produced 151 absolute successes to DACE's 39, but only bettered q_{25} on 201 runs. MAGS usually produced more absolute successes than Compass (7 of 10 functions) and always produced more relative successes. MAGS usually produced more absolute successes than LHC-Compass (6 of 10 functions, 1 tie), but LHC-Compass often produced more relative successes (6 of 10). MAPS-SDO and MAPS-Dynamic typically produced more absolute successes than MAGS (9 of 10) and always produced more relative successes. MAPS-Dynamic typically produced the most absolute successes of any algorithm (8 of 10 functions, 1 tie) and always produced the most relative successes.

For $p = 4$, DACE never produced more than 1 absolute success. Both Compass (0–61) and LHC-Compass (0–82) always produced more absolute successes than DACE. Furthermore, both Compass (485–654) and LHC-Compass (603-725) consistently bettered q_{25}. MAGS always outperformed the pattern searches, both absolutely and relatively. MAPS-SDO and MAPS-Dynamic produced more absolute successes than MAGS for 8 of the 10 objective functions and always produced more relative successes.

For $p = 5$, DACE never succeeded absolutely and the pattern searches rarely did. However, both Compass (626–687) and LHC-Compass (683–776) consistently bettered q_{25}. MAGS always outperformed the pattern searches, both absolutely and relatively. MAPS-SDO and MAPS-Dynamic always outperformed MAGS, except in one case that MAGS bettered q_{25} on 891 runs versus 870 runs for MAPS-Dynamic.

We offer the following conclusions:

1. Sequential experimentation is desirable for optimization. Except for $p = 2$, DACE was dramatically outperformed by the sequential optimization strategies. Auxiliary experiments suggest that constructing and minimizing $\hat{f}$ rarely improves on the initial $V-1$ design sites. Thus, DACE succeeds by randomly sampling $[\ell, u]$, a strategy that is more susceptible than most to the curse of dimensionality. Because the sequential strategies only draw $N \ll V-1$ initial design sites ($N = 1$ for Compass), they are less likely than DACE to place an initial design site near a global minimizer. Usually, however, their ability to improve on the initial design sites is crucial. DACE occasionally performs reasonably well when p is quite small and the basin that contains the global minimizer is hard to find, but it generally performs abysmally.

2. Using interpolating approximations to guide a pattern search does improve performance. This was the premise that motivated Booker et al. (1999), but Siefert (2000) appears to have been the first researcher to document this phenomenon with extensive numerical experimentation.

3. The best sequential strategies intelligently balance the need to exploit $\hat{f}_k$ and the need to construct a better $\hat{f}_{k+1}$. Again, this point has already been made by Frank (1995), by Trosset (1998), and by Jones, Schonlau, and Welch (1999), but Siefert (2000) provides the most comprehensive documentation to date.

References

Booker, A. J. (1996). Case studies in design and analysis of computer experiments. In *Proceedings of the*

Section on Physical and Engineering Sciences, pages 244–248. American Statistical Association.

Booker, A. J., Dennis, J. E., Frank, P. D., Serafini, D. B., Torczon, V., and Trosset, M. W. (1999). A rigorous framework for optimization of expensive functions by surrogates. *Structural Optimization*, 17(1):1–13.

Cox, D. D. and John, S. (1997). SDO: A statistical method for global optimization. In Alexandrov, N. and Hussaini, M. Y., editors, *Multidisciplinary Design Optimization: State of the Art*, pages 315–329. SIAM, Philadelphia.

Dennis, J. E. and Torczon, V. (1997). Managing approximation models in optimization. In Alexandrov, N. and Hussaini, M. Y., editors, *Multidisciplinary Design Optimization: State of the Art*, pages 330–347. 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.

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

Jones, D., Schonlau, M., and Welch, W. (1999). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, ??:1–39.

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. (1999). Pattern search algorithms for bound constrained minimization. *SIAM Journal on Optimization*, 9:1082–1099.

Lewis, R. M., Torczon, V., and Trosset, M. W. (1998). Why pattern search works. *Optima*, (59):1–7.

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

Padula, A. D. (2000). Interpolation and pseudorandom function generation. Senior honors thesis, Department of Mathematics, College of William & Mary, Williamsburg, VA. Available with software at `http://www.math.wm.edu/~trosset/`.

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.

Schonlau, M. and Welch, W. J. (1996). Global optimization with nonparametric function fitting. In *Proceedings of the Section on Physical and Engineering Sciences*, pages 183–186. American Statistical Association.

Schonlau, M., Welch, W. J., and Jones, D. R. (1997). A data-analytic approach to Bayesian global optimization. In *Proceedings of the Section on Physical and Engineering Sciences*, pages 186–191. American Statistical Association.

Schonlau, M., Welch, W. J., and Jones, D. R. (1999). Global versus local search in constrained optimization of computer models. In Flournoy, N., Rosenberger, W. F., and Wong, W. K., editors, *New Developments and Applications in Experimental Design*. Institute of Mathematical Statistics, Hayward, CA. To appear.

Siefert, C. M. (2000). Model-assisted pattern search. Senior honors thesis, Department of Computer Science, College of William & Mary, Williamsburg, VA. Available with software at `http://www.cs.wm.edu/~va/`.

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

Torczon, V. and Trosset, M. W. (1997). From evolutionary operation to parallel direct search: Pattern search algorithms for numerical optimization. *Computing Science and Statistics*, 29(1):396–401.

Torczon, V. and Trosset, M. W. (1998). Using approximations to accelerate engineering design optimization. In *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization: A Collection of Technical Papers, Part 2*, pages 738–748. American Institute of Aeronautics and Astronautics.

Trosset, M. W. (1996). Taguchi and robust design. Technical Report 96-31, Department of Computational & Applied Mathematics, Rice University, 6100 Main Street, Houston, TX 77005-1892.

Trosset, M. W. (1998). Optimization on a limited budget. In *Proceedings of the Section on Physical and Engineering Sciences*, pages 210–215. American Statistical Association.

Trosset, M. W. (1999). The krigifier: A procedure for generating pseudorandom nonlinear objective functions for computational experimentation. Interim Report 35, Institute for Computer Applications in Science & Engineering, NASA Langley Research Center, Hampton, VA 23681-0001.

Trosset, M. W. and Padula, A. D. (2000). Designing and analyzing computational experiments for global optimization. Technical Report TR00-25, Department of Computational & Applied Mathematics—MS 134, Rice University, 6100 Main Street, Houston, TX 77005-1892.

Trosset, M. W. and Torczon, V. (1997). Numerical optimization using computer experiments. Technical Report 97-38, Institute for Computer Applications in Science & Engineering, NASA Langley Research Center, Hampton, VA 23681-0001.

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