OPTIMIZATION USING SURROGATE OBJECTIVES ON A HELICOPTER TEST EXAMPLE [*]

ANDREW J. BOOKER [§], J. E. DENNIS, JR. [‡], PAUL D. FRANK [§], DAVID B. SERAFINI [‡], AND VIRGINIA TORCZON [¶]

Abstract. This paper presents results for a 31 variable helicopter rotor design example. Results are given for several numerical methods. This is a brief description of a portion of the Boeing/IBM/Rice University collaboration whose purpose is to develop effective numerical methods for managing the use of approximation concepts or response surface methodology in design optimization.

Key Words: Approximation concepts, surrogate optimization, response surface methodology, direct search methods, derivative-free optimization.

1. Introduction. The purpose of this paper is to present preliminary numerical results for a new generalized version of the model management framework introduced by Dennis and Torczon in [11]. This note forms a brief summary of a portion of the doctoral dissertation of David Serafini. More detail and more results can be found in [21] and [3].

The premise of our work is summed up in the following list:

- Many optimal design problems are too computationally expensive for direct application of optimization codes.
- Gradient methods may be inconvenient to apply and likely to converge to design refinements rather than breakthrough designs.

Current practice (See the contribution to this volume by B. Grossman and [5], [1], [6] [14]) is to sample widely in design variable space, build a response surface model of the expensive high fidelity objective, and possibly of the constraints, and then to use the models as surrogates for the high fidelity objective function and constraints in doing optimization on the problem posed using the high fidelity functions. A difficulty is what to do next if the surrogate optimum is not sufficiently good to satisfy the designer. Current strategies are sensible, but expensive and not rigorously justified.

We have developed, analyzed, and implemented a rigorous framework to manage this process. Preliminary results given here show potential for establishing a new procedure that significantly improves current practice.

The example problem we consider here is one of interest to Boeing. It is to choose values for 31 design variables to minimize a measure of vibration of a helicopter rotor blade. A simplified simulation code is used in which the flow field in the wake of the rotor blade is held constant. This simplified simulation code only requires a few minutes per objective function value. In a real design situation, this code might itself be viewed as a surrogate for the full wake simulation, which requires hours to execute. The constant wake simulation captures most of the qualitative behavior of the full physics simulation, and we are assured by the engineers that results for this test example will be relevant in deciding how to tackle the real problem.

[*] Research was supported by DOE FG03-93ER25178, CRPC CCR-9120008, AFOSR-F49620-95-1-0210, The Boeing Company, and the REDI Foundation.

[‡] Department of Computational and Applied Mathematics & Center for Research on Parallel Computation, Rice University, P. O. Box 1892, Houston TX 77005.

[§] Mathematics & Engineering Analysis, Boeing Information Support Services, Research and Technology, Box 3707, M/S 7L-68, Seattle, WA 98124

[¶] Computer Science Department, College of William & Mary, P. O. Box 8795, Williamsburg, VA 23187

The variables to be chosen correspond to physical rotor characteristics at 10 segments along the span of the blade. They comprise 10 masses, 10 centers of gravity, and 11 single direction stiffnesses. There are simple constraints in the form of upper and lower bounds on the variables and an upper bound on the total mass. The objective function is the weighted sum of the first and second harmonic frequencies of the vibration of the blade for two flight conditions.

Thus, we can write the "real" problem as:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \mathcal{B} \\ & c^T x \leq 0 \ , \end{aligned} \tag{1}$$

where $\mathcal{B} = \{x \in \Re^n \ : \ a_i \leq x_i \leq b_i \text{ for } i = 1, \ldots, n\}$.

However, we will compute values of $f(x)$ reluctantly and only when we have used values of a current surrogate $\hat{f}(x)$ to identify designs x for which the expense seems justified.

Another complication in this problem, and others for which our work is intended, is that $f(x)$ can not be computed for a significant percentage of points that satisfy all the explicit constraints. This is not so surprising when one considers that the simulations underlying many of the problems of this type involve sometimes fragile iterative methods for solving the discretized problem. Our methods cope easily with difficulties of this type if we assume that the designer would not accept as a best design a design for which the simulation code does not converge. This is reasonable, but it is not so obvious as it might seem since we have encountered simulation failures at points very close to local minimizers. In contrast to our methods, a finite difference gradient based method may have difficulty dealing with nonconvergent points if they arise during the gradient computation.

Throughout the remainder of this paper, we will have to distinguish between whether or not we have counted the number of total calls or just the successful calls, when a value of $f(x)$ actually was returned. For this particular problem, about 60% of the simulations fail to converge for representative feasible points, for example those in the $\mathcal{OA}$s of the next section. At first, this seems to indicate that the simulation code is not very trustworthy. However, it must be remembered that codes of this type were not developed for use with optimization algorithms; they are used primarily by engineers experienced in dealing with the code's frailties. In order to make the case for using optimization tools in design, we must deal with the codes as they are. This is the most direct route to convince the developers that it is worth the trouble to make simulation codes more robust so that optimization tools can be used more effectively.

In the next section we will discuss the method used to obtain the sequence of surrogates needed for our approach. We must specify the way data sites were chosen as well as the form of the interpolating functions. Although we chose to use interpolating functions in this problem, the model management framework we have developed could also be used with the simplified simulation code as the surrogate for the full simulation, or a hierarchy of surrogates of varying accuracy and cost.

2. DACE Response Surface Models. The specific design vectors for which we compute f values to use in building the surrogate models are chosen from the same class of experimental designs used in quasi-Monte Carlo integration: orthogonal arrays ($\mathcal{OA}$s) [19]. The designs we used were *strength* 2 designs, which means that if there are l distinct values for each variable then every subset of 2 variables is a grid in the l values. This gives one confidence that these designs are "infiltrating" the design space well.

An advantage of using $\mathcal{OA}$s is the availability of code for generating them (STATLIB `http://lib.stat.cmu.edu`). $\mathcal{OA}$s are for box-constrained regions. To create a design meeting the mass constraint, we generated a large design in the box $\mathcal{B}$ defined by the variable bounds, and then we kept all the points in that design inside a slightly expanded constraint boundary. [4]

The surrogates are from a class of flexible models that can be adjusted locally to some extent as new points are added. They are motivated by the supposition that the output being modeled is a realization of a Gaussian spatial process. Called kriging models, they are recommended in the Design and Analysis of Computer Experiments (DACE) literature, surveyed in [20]. These models are made to interpolate the observations, and they depend on a set of correlation parameters [20] that we estimated via maximum likelihood estimation (MLE), as in [9]. In [13, 9], it is shown that MLE can be thought of as a form of cross-validation.

This Gaussian process supposition is a convenient fiction ([23]) that provides useful error estimates, termed *mean squared errors* (*mse*), derived from the putative underlying Gaussian process. The *mse* at a point x is the variance of the spatial process at x given the sampled values (observations). Note that the *mse* approaches 0 as x approaches an observation, and it is (relatively) large when x is far from any observations. There is some evidence that this may not be an unreasonable framework from which to predict future prediction errors [20, 15], particularly if one can diagnose the fit [15] and determine the reasonableness of the Gaussian process assumption or determine a transformation of the response that makes it reasonable. In the BLGS algorithm mentioned below, *mse* is used to guide the selection of new points for improving the surrogate's accuracy.

The surrogate $\hat{f}$ provides a global model of f, and a promising use of the surrogate is in estimating the important variables in f. This is done by applying to $\hat{f}$ the functional analysis of variance (ANOVA) introduced and described in [12, 19, 20]. The functional ANOVA provides estimates of contributions of individual variables to $\hat{f}$ (*main effects*) and of combinations of variables to $\hat{f}$ (*interaction effects*). We use these quantities obtained from $\hat{f}$ as estimates of the corresponding quantities for f.

Currently, we are testing an 11 variable problem derived from the 31 variable test example used here through an ANOVA decomposition. Obtaining this reduction in dimension did involve a 201 point model (requiring 554 total simulation calls), but such information is extremely valuable to the designer. After all, one rarely solves a design problem by solving the first few formulations of it. In a real sense, the purpose of optimization tools in design is to aid the designer in arriving at an understanding of the design he or she eventually chooses. ANOVA applied to the surrogate appears all the more valuable in that light.

3. Model Management Framework. The test results we give are for the most rudimentary version of our framework. There are many variants that seem likely to lead to more efficiency, and we plan to explore them. Still, the results for this first version suffice to make our point that this is an exciting research direction.

We begin the solution process assuming that we have a "global model" constructed as an initial surrogate. In this case, a DACE model of the 31 variable surface over the box defined by the bound constraints on the individual variables. The models used here were based on 59, 201, and 381 converged samples of the "truth" simulation we are trying to optimize. The number of function evaluations reported for MMF to reach a given value of the objective function includes this initialization cost. As the solution process proceeds, new values of the true objective are obtained and the model is modified to interpolate at these new points. At this stage, we also report all unsuccessful calls to the simulation code in the number of function values required to reach a certain objective value.

The MMF variants tested here all are based on evaluating the surrogate at a adaptively scaled and oriented pattern, or template, of 29,800 points in design space. This template of points is one that could be used to take one step of a parallel direct search (PDS) method on the surrogate [10], [22]. When the surrogate is evaluated at these designs, we assume that the lower the value of the surrogate at a given design vector, the better the chance that the true objective would have an actual lower value there.

Suppose the template designs that produce decrease in the surrogate objective are ordered by how much improvement they produce in the surrogate objective. The difference in the MMF methods tested so far is simply how many of the best of these points are tested by evaluating the true objective to see if they do actually produce a better design. Of course, this is much larger than any PDS template one would use on a real function. For example, the numerical results we present below for PDS applied directly to the true objective function is for a template size of 96 points. However, it is well to remember that probably for only about 45 of these points do we actually obtain true function values. The PDS results given are the total number of calls.

In all the MMF variants, if a better design is found, then the best of the new designs is the new iterate. The new truth values are added to the surrogate, and then the recalibrated surrogate is used to generate the next iterate. If no better design is found, then the convergence theory we use [17], [18] requires a "poll" step on the current scale before we can decrease the scale of the search template. Without going into greater detail, the scale of the template sets the distances between points on the template, not the number of points in the template.

A poll step consists of evaluating the actual objective on a positive basis set of steps from the current iterate. A positive basis can be thought of as a set of vectors with the property that if the gradient at the current iterate is not zero, then at least one of the steps in the set makes a positive inner product with the negative gradient, and so it is a descent direction.

The algorithm uses the surrogate to choose the order of evaluation of steps from a positive basis until it finds a better design or exhausts all the steps in the positive basis without finding a better design. When this happens, then the algorithm is allowed to shrink the scale of the template, and it is in this way that the search settles down to converge based on the size of the step in design space. Without derivatives, the size of the step, the amount of improvement, and the total number of function evaluations done so far are about the only stopping criteria available.

A poll step can be expensive. If it is unsuccessful, then it may require between $n+1$ and $2n$ truth evaluations depending on the choice of positive basis [18]. In fact, our plan is that we will switch to the derivative-free optimization algorithm (DFO) presented in this volume by Andrew Conn after we encounter the first unsuccessful complete poll step. This situation is likely to indicate that we are in a basin of the objective, and so an efficient local method should be used to home in more quickly than direct search methods can be expected to. Starting DFO in this way would completely eliminate the variability in the DFO results caused by taking a random first step. [8], [7].

4. Sample Test Results. In this section, we will present a graphical summary of the test results so far for the 31 variable constant wake helicopter rotor design example. The variables in this problem are scaled over 10 orders of magnitude, and so all of the algorithms, except as noted for some DFO results, rescaled the variables. We will present some results from several algorithms:

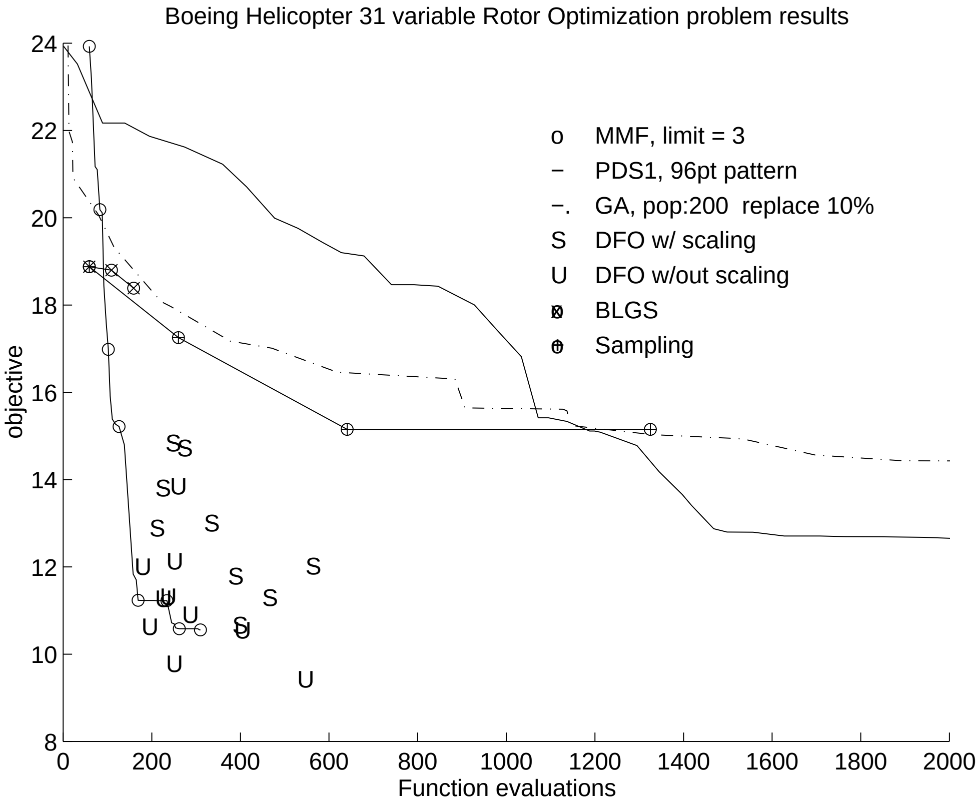
- **MMF**: This is the new method discussed in the last section. In this variant, optimization on the surrogate is used to identify 3 points at which an improved design is most strongly indicated. The candidates are evaluated one at a time using the full accuracy simulation until the first candidate is encountered that truly gives a better design. This is an obvious knob for the algorithm. The number of function evaluations given are the converged points for the initial model and the total of all the simulation calls requested by the MMF.
- **DFO**: This is the interesting method from Conn et al. mentioned in the last section. Here, we present results from all the runs we were provided since the results were so variable. DFO has a strong random component, which may account for the anomaly that the best runs were for the unscaled problem. An amazing fact is that DFO requested function values at points for which only 3% failed to return a value. The numbers reported for DFO are the total number of values requested.
- **PDS**: This is Torczon's implementation [22] of the parallel direct search method of [10]. The numbers reported for PDS are the total number of values requested. We used a smaller template than the default template on the distributed code. As advertised, it steadily descends to a quite good objective value, slowing down after about 1500 evaluations, but continuing to descend out to 5500 function evaluations.
- **GA**: This is an algorithm from PGAPack [16]. We used the parameter settings suggested to us for this problem by its author David Levine of the Boeing Company. The numbers reported for GA are the total number of values requested. It performs as advertised - nice initial decrease and then leveling out. It makes little progress after 1500 evaluations and terminates at 3300 evaluations.
- **BLGS**: This is an intuitively appealing method of Booker and Frank [2] in which several truth values are computed at each iteration. The number of expensive evaluations allocated to each iteration is fixed, in this case at 50. Some sites are chosen because the current model predicts them to be a better design, and some are chosen because they are relatively far from any current design sites. The numbers reported for BLGS are the total number of values requested.

We feel that these results are very good – so good in fact that they may indicate that this is an easier problem than we anticipated. DFO and MMF both get very good answers in a number of function evaluations that would be considered very good for quasi-Newton methods. We have seen unpublished results for a finite-difference quasi-Newton method, and they are not nearly as good as any of these methods. A higher final objective value is reached after more evaluations. This is either due to the difficulty in choosing finite-difference step sizes for such a rough function, or to the often cited tendency of derivative-based methods to converge to a local optimum.

These results strengthen our confidence in the research plan we have been pursuing in the Boeing/IBM/Rice collaboration. Specifically, we will use DACE models of sufficient accuracy to provide a useful ANOVA decomposition to provide insight into the problem. Then, we will use some variant of MMF (possibly one with a BLGS flavor) to get into a good basin for the true function. When MMF begins to slow down after it enters the basin, we will provide a warm start for DFO to efficiently find the solution.

5. Acknowledgments. The computational results presented in this paper were enriched significantly by the cooperation and collaboration of Katya Scheinberg in providing DFO test results. We also thank David Levine for suggesting genetic algorithm parameter values for use in PGAPack.

Much of this work was accomplished during the year John Dennis and David Serafini spent in the Mathematics and Engineering Analysis Group at Boeing Information Support Services Research and Technology Division in Seattle. They wish to thank DOE and AFOSR respectively for support, and the Boeing Company for the hospitality they were shown.



REFERENCES

[1] Jean-Francois M. Barthelemy and Raphael T. Haftka. Approximation concepts for optimum structural design – a review. *Structural Optimization*, 5:129–144, 1993.

[2] A. J. Booker, A. R. Conn, J. E. Dennis, Jr, Paul D. Frank, Michael Trosset, and Virginia Torczon. Global modeling for optimization: Boeing/IBM/Rice collaborative project 1995 final report. Technical Report ISSTECH–95–032, Boeing Information Support Services, Research and Technology, Box 3707, M/S 7L–68, Seattle, Washington 98124, December 1995.

[3] A. J. Booker, J. E. Dennis, Jr., Paul D. Frank, David B. Serafini, Michael Trosset, and Virginia Torczon. A rigorous framework for optimization of expensive functions by surrogates. In preparation, 1997.

[4] A. J. Booker, Paul D. Frank, A. R. Conn, J. E. Dennis, Jr., David B. Serafini, Virginia Torczon, and Michael Trosset. Multi-level design optimization: Boeing/IBM/Rice collaborative project 1996 final report. Technical Report ISSTECH–96–031, Boeing Computer Services, Research and Technology, Box 3707, M/S 7L–68, Seattle, Washington 98124, December 1996.

[5] G. E. P. Box. The exploration and exploitation of response surfaces: Some general considerations and examples. *Biometrics*, 10:16–60, March 1954.

[6] S. L. Burgee, A. A. Giunta, V. Balabanov, B. Grossman, W. H. Mason, R. Narducci, R. T. Haftka, and L. T. Watson. A coarse grained parallel variable–complexity multidisciplinary optimization program. Technical Report 95–20, Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, October 1995.

[7] A. R. Conn, K. Scheinberg, and Ph. L. Toint. On the convergence of derivative-free methods for unconstrained optimization. Technical Report 96/10, Department of Mathematics, Faculté Universitaires, B-5000 Namur, Belgium, December 1996.

[8] A. R. Conn and Ph. L. Toint. An algorithm using quadratic interpolation for unconstrained derivative free optimization. In Gianni Di Pillo and Franco Giannessi, editors, *Nonlinear Optimization and Applications*. Plenum Publishing, 1995.

[9] C. Currin, T. J. Mitchell, M. Morris, and D. Ylvisaker. A Bayesian approach to the design and analysis of computer experiments. Technical Report ORNL–6498, Oak Ridge National Laboratory, 1988.

[10] J. E. Dennis, Jr. and Virginia Torczon. Direct search methods on parallel machines. *SIAM J. Optimization*, 1(4):448–474, November 1991.

[11] J. E. Dennis, Jr. and Virginia Torczon. Managing approximation models in optimization. In Natalia Alexandrov and M.Y. Hussaini, editors, *Multidisciplinary Design Optimization: State-of-the-Art*, pages 330–347. SIAM, Philadelphia, 1997. Also available as Rice U. CAAM Tech. Report 95-19.

[12] B. Efron and C. Stein. The jacknife estimate of variance. *The Annals of Statistics*, 9(3):586–596, 1981.

[13] S. Geisser and W. F. Eddy. A predictive approach to model selection. *Journal of the American Statistical Association*, 74:153–160, 1979.

[14] A. A. Giunta, V. Balabanov, D. Haim, B. Grossman, W. H. Mason, L. T. Watson, and R. T. Haftka. Wing design for a high-speed civil transport using a design of experiments methodology. In *Proceedings of the Sixth AIAA/NASA/USAF Symposium on Multidisciplinary Analysis and Optimization, Bellevue, WA, 1996.* American Institute of Aeronautics and Astronautics, 1996. AIAA Paper 96-4001.

[15] D. R. Jones, M. Schonlau, and W. J. Welch. A data analytic approach to bayesian global optimization. In *Proceedings of the ASA*, 1997.

[16] David Levine. Users guide to the PGAPack parallel genetic algorithm library. Technical Report ANL-95/18, Argonne National Laboratory, January 1996. Available from URL ftp://info.mcs.anl.gov/pub/pgapack/pgapack.tar.Z.

[17] Robert Michael Lewis and Virginia Torczon. Pattern search algorithms for bound constrained minimization. Technical Report 96–20, ICASE, NASA Langley Research Center, Hampton, VA 23681–0001, March 1996. To appear in *SIAM Journal on Optimization*.

[18] Robert Michael Lewis and Virginia Torczon. Rank ordering and positive bases in pattern search algorithms. Technical Report 96–71, ICASE, NASA Langley Research Center, Hampton, VA 23681–0001, December 1996. Submitted to *SIAM Journal on Optimization*.

[19] A. B. Owen. Orthogonal arrays for computer experiments, integration and visualization. *Statistica Sinica*, 2:439–452, 1992.

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

[21] David B. Serafini. *A Framework for Managing Models in Optimization for Computationally Expensive Functions*. PhD thesis, Rice University, Houston, TX, 1997.

[22] Virginia Torczon. PDS: Direct search methods for unconstrained optimization on either sequential or parallel machines. Technical Report 92–9, Department of Mathematical Sciences, Rice University, Houston, TX 77251–1892, 1992. Submitted to *ACM Transactions on Mathematical Software*.

[23] Michael W. Trosset and Virginia Torczon. Numerical optimization using computer experiments. *Technometrics*, 1997. Submitted for publication.