AIAA JOURNAL
Vol. 42, No. 3, March 2004

Global Approximation and Optimization Using Adjoint Computational Fluid Dynamics Codes

Stephen J. Leary,* Atul Bhaskar,† and Andrew J. Keane‡
University of Southampton, Southampton, England SO17 1BJ, United Kingdom

Approximation methods have found increasing use in the optimization of complex engineering systems. The approximation method provides a surrogate model that, once constructed, can be used in lieu of the original expensive model for the purposes of optimization. These approximations may be defined locally, for example, a low-order polynomial response surface approximation that employs trust region methodology during optimization, or globally, by the use of techniques such as kriging. Adjoint methods for computational fluid dynamics have made it possible to obtain sensitivity information on the model's response without recourse to finite differencing. This approach then allows for an efficient local optimization strategy where these sensitivities are utilized in gradient-based optimization. The combined use of an adjoint computational fluid dynamics code with approximation methods (incorporating gradients) for global optimization is shown. Several approximation methods are considered. It is shown that an adjoint-based approximation model can provide increased accuracy over traditional nongradient-based approximations at comparable cost, at least for modest numbers of design variables. As a result, these models are found to be more reliable for surrogate assisted optimization.

I. Introduction

COMPUTATIONALLY expensive simulation codes based on mathematical models of some system of interest are commonly used throughout the engineering industry. One example is the field of computational fluid dynamics (CFD), where a single evaluation of the model may take many hours of computer run time. If the goal is to find the global optimum of this model, we are often overcome by the model's expense. Direct optimization, which requires many calls to the model of interest, is, more often than not, unrealistic.

Over the past 15 years, the use of adjoint codes for CFD design optimization has gained momentum. These have developed from potential flow, to the Euler equations, and finally to the Navier–Stokes equations.[1–6] The complexity of the models has also increased; two-dimensional airfoils, three-dimensional wings, and, finally, full aircraft configurations have been considered.[6–9]

The adjoint approach is efficient for optimization because all sensitivities are available at a cost of one CFD evaluation. When the number of dimensions is large, this is much more efficient than the application of finite differencing to obtain the derivatives. As a result, gradient-based optimization can be performed very efficiently. For example, in Ref. 10, a full aircraft configuration consisting of over 4000 design variables was considered.

However, gradient-based optimizers are only local search algorithms and tend to become trapped in local minima. One possibility, therefore, is to make multiple restarts (for example, Ref. 11). Another is to use a direct search strategy, again, as in Ref. 11. Finally, one could use global stochastic optimization techniques such as simulated annealing (as in Ref. 12) or genetic algorithms (as in Ref. 13). However, these do not employ gradient information when searching over the design space, and so another strategy is called for. Approximation methods provide one solution: These are used to construct cheap surrogates of the expensive model. They are based on a limited number of calls to the expensive model (using some design of experiment). Once this surrogate model is constructed, it replaces the original model for the purposes of optimization.

When we consider optimization, our approximations can be defined locally or globally. Local approximations (as in Ref. 14) are typically based on low-order polynomials and are defined only over a specific region of interest, usually around the current best design. These models are only valid in some neighborhood of this design, and so the optimization is performed by the use of some trust region strategy: We only optimize over a portion of the domain in which the approximation is valid.[15]

Global approximations, on the other hand, try to capture the model's behavior over the entire domain of interest. Many different examples of global approximations exist; we focus on just three, Shepard weighting functions,[16] radial basis functions,[17] and kriging.[18]

It seems a natural idea to combine the disciplines of adjoint CFD analysis and approximation methods together, and this was attempted by Chung and Alonso[19] with a modified polynomial response surface approximation. The adjoint approach allowed approximations to be constructed in $\mathcal{O}(k)$ CFD evaluations, where k is the number of design variables [as opposed to $\mathcal{O}(k^2)$ for a traditional response surface]. As a result, the surrogates were constructed very efficiently. Earlier works by Rodriguez et al.[20] and by Alexandrov et al.[21] also considered the construction of local response surfaces by the use of gradient information. These algorithms employ trust region methodologies and, as a result, convergence (albeit to a local minimum) is guaranteed.

To our knowledge, there is little work in the literature that attempts to combine the output of an adjoint CFD code together with global approximation techniques. The method of kriging has been developed to incorporate gradients in Ref. 22, and so this is a natural contender. A preliminary study applied to analytic functions with an adjoint method in mind was performed in Ref. 23. This paper also contained a discussion of applications to finite element modeling of structures incorporating sensitivities. Independently, a preliminary study with an adjoint code was performed in Ref. 24. However, kriging is a relatively sophisticated approximation technique. Simpler models are, therefore, discussed first: We formulate a gradient-based version of the Shepard weighting function, as well as a gradient-based version of radial basis functions. We then compare all three approaches for global approximation of the output of an adjoint CFD code. Throughout, we consider models of directly comparable cost. Because the adjoint solver is as expensive as a

Received 17 July 2002; revision received 10 October 2003; accepted for publication 15 October 2003.

*Research Fellow, Computational Engineering and Design Center, School of Engineering Sciences; S.J.Leary@soton.ac.uk.

†Senior Lecturer, Computational Engineering and Design Center, School of Engineering Sciences; A.Bhaskar@soton.ac.uk.

‡Professor, Computational Engineering and Design Center, School of Engineering Sciences; Andy.Keane@soton.ac.uk.

CFD evaluation, we must take this expense into account. We then show how such approximations can be utilized for consideration of global optimization.

II. Global Approximation Methods

In this section, we consider three global approximation methods with and without gradient information included. The first is the simplest: Shepard weighting functions. The second, radial basis functions, is more sophisticated, but the model can be constructed at modest cost. Finally, we consider kriging. This is the most expensive model to construct because we are required to train the model's hyperparameters. This is the most sophisticated model considered in this paper.

A. Shepard Weighting

1. Derivative Information Excluded

Consider an underlying function f with value $f(\boldsymbol{x}^{(i)})$ at an input vector $\boldsymbol{x}^{(i)}$ for $i = 1, 2, \ldots, N$. We consider input vectors of arbitrary dimension k, that is, $\boldsymbol{x}^{(i)} = \{x_1^{(i)}, x_2^{(i)}, \ldots, x_k^{(i)}\}$. In its simplest form, the method of Shepard[16] can be written as

$$F(\boldsymbol{x}) = \sum_{j=1}^{N} W_j(\boldsymbol{x}) f\left(\boldsymbol{x}^{(j)}\right) \Bigg/ \sum_{i=1}^{N} W_i(\boldsymbol{x}) \tag{1}$$

where $F(\boldsymbol{x})$ is our interpolating approximation.

The relative weights are defined by the inverse distance functions

$$W_j(\boldsymbol{x}) = \left[\frac{(R_w - d_j)_+}{R_w d_j}\right]^2 \tag{2}$$

where

$$(R_w - d_j)_+ = \begin{cases} R_w - d_j & \text{if} \quad d_j < R_w \\ 0 & \text{if} \quad d_j \geq R_w \end{cases} \tag{3}$$

Here, d_j denotes the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{x}^{(j)}$ and R_w is a radius of influence around $\boldsymbol{x}^{(j)}$. The data at $\boldsymbol{x}^{(j)}$ only influence interpolated values within this radius.

The method has undergone modification over the years, for instance, Renka[25] replaced $f(\boldsymbol{x}^{(j)})$ in Eq. (1) by a quadratic function $Q_j(\boldsymbol{x})$, satisfying

$$Q_j\left(\boldsymbol{x}^{(j)}\right) = f\left(\boldsymbol{x}^{(j)}\right) \tag{4}$$

the coefficients of this quadratic being chosen via a weighted least-squares minimization procedure.

2. Derivative Information Included

We now assume the derivatives $\partial f(\boldsymbol{x}^{(i)})/\partial x_l$, $l = 1, 2, \ldots, k$, are also available to us. It can be shown[25] that the weights

$$\bar{W}_j = W_j \Bigg/ \sum_{i=1}^{N} W_i$$

satisfy

$$\bar{W}_j\left(\boldsymbol{x}^{(i)}\right) = \delta_{ji} = \begin{cases} 0 & \text{if} \quad j \neq i \\ 1 & \text{if} \quad j = i \end{cases} \tag{5}$$

$$\sum_{j=1}^{N} \bar{W}_j(\boldsymbol{x}) = 1 \tag{6}$$

$$\frac{\partial \bar{W}_j\left(\boldsymbol{x}^{(i)}\right)}{\partial x_1} = \frac{\partial \bar{W}_j\left(\boldsymbol{x}^{(i)}\right)}{\partial x_2} = \cdots = \frac{\partial \bar{W}_j\left(\boldsymbol{x}^{(i)}\right)}{\partial x_k} = 0 \tag{7}$$

Suppose we replace $f(\boldsymbol{x}^{(j)})$ in Eq. (1) by an arbitrary function L_j, from Eqs. (5) and (7) we then obtain

$$\frac{\partial F\left(\boldsymbol{x}^{(i)}\right)}{\partial x_l} = \sum_{j=1}^{N} \left[\frac{\partial \bar{W}_j\left(\boldsymbol{x}^{(i)}\right)}{\partial x_l} L_j\left(\boldsymbol{x}^{(i)}\right) + \bar{W}_j\left(\boldsymbol{x}^{(i)}\right) \frac{\partial L_j\left(\boldsymbol{x}^{(i)}\right)}{\partial x_l}\right]$$

$$= \frac{\partial L_i\left(\boldsymbol{x}^{(i)}\right)}{\partial x_l} \tag{8}$$

for $l = 1, 2, \ldots, k$.

Hence, if we choose

$$L_j(\boldsymbol{x}) = \frac{\partial f\left(\boldsymbol{x}^{(j)}\right)}{\partial x_1}\left(x_1 - x_1^{(j)}\right) + \frac{\partial f\left(\boldsymbol{x}^{(j)}\right)}{\partial x_2}\left(x_2 - x_2^{(j)}\right)$$

$$+ \cdots + \frac{\partial f\left(\boldsymbol{x}^{(j)}\right)}{\partial x_k}\left(x_k - x_k^{(j)}\right) + f\left(\boldsymbol{x}^{(j)}\right) \tag{9}$$

then we obtain an approximation that gives

$$F\left(\boldsymbol{x}^{(j)}\right) = f\left(\boldsymbol{x}^{(j)}\right) \tag{10}$$

$$\frac{\partial F\left(\boldsymbol{x}^{(j)}\right)}{\partial x_l} = \frac{\partial f\left(\boldsymbol{x}^{(j)}\right)}{\partial x_l}, \qquad l = 1, 2, \ldots, k \tag{11}$$

that is, the prediction interpolates the data with correct derivatives at the data points.

Figure 1 shows approximations to a given function f on the domain [0, 10]. Five data points are given, and a Shepard approximation with and without gradients is calculated with $R_w = 5$. As can be seen, the incorporation of gradient information leads to a more accurate prediction.

B. Radial Basis Functions

1. Derivative Information Excluded

Once more, given an underlying function f with value $f(\boldsymbol{x}^{(i)})$ at a $k \times 1$ input vector $\boldsymbol{x}^{(i)}$ for $i = 1, 2, \ldots, N$, we require some interpolating approximation $F(\boldsymbol{x})$. We restrict our discussion to a model where the basis function centers are the input vectors. As a result, the radial basis function approach introduces a set of N basis functions, one for each data point. These take the form $\phi(\|\boldsymbol{x} - \boldsymbol{x}^{(i)}\|)$, where $\phi(\cdot)$ is some (generally) nonlinear function. The ith such function depends on the distance $\|\boldsymbol{x} - \boldsymbol{x}^{(i)}\|$ between $\boldsymbol{x}$ and $\boldsymbol{x}^{(i)}$, usually taken as Euclidean distance. The output is taken as a linear combination of basis functions, that is,

$$F(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \phi\left(\left\|\boldsymbol{x} - \boldsymbol{x}^{(i)}\right\|\right) \tag{12}$$

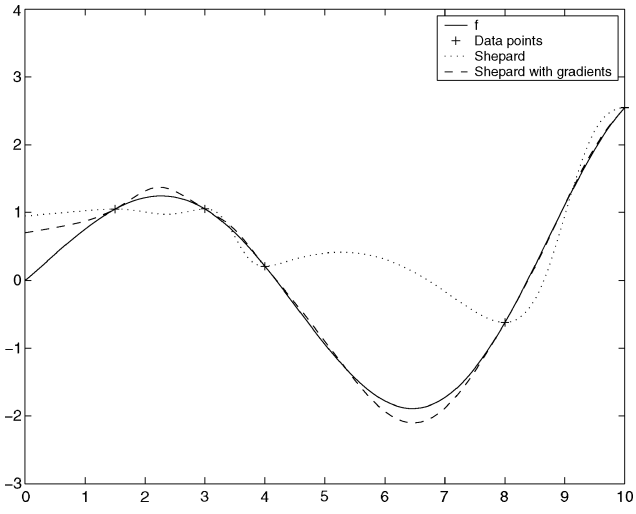


Fig. 1 Shepard model.

The idea is to find weights w_i, such that the function in Eq. (12) interpolates the data. To do this, we are required to satisfy

$$F\left(\boldsymbol{x}^{(j)}\right) = \sum_{i=1}^{N} w_i \phi\left(\left\|\boldsymbol{x}^{(j)} - \boldsymbol{x}^{(i)}\right\|\right) = f\left(\boldsymbol{x}^{(j)}\right) \tag{13}$$

By definition of the vectors $\boldsymbol{f} = [f(\boldsymbol{x}^{(1)}), f(\boldsymbol{x}^{(2)}), \ldots, f(\boldsymbol{x}^{(N)})]'$, $\boldsymbol{w} = [w_1, w_2, \ldots, w_N]'$, and the matrix Φ with ijth entry $\Phi_{ij} = \phi(\|\boldsymbol{x}^{(j)} - \boldsymbol{x}^{(i)}\|)$, this simplifies to $\Phi\boldsymbol{w} = \boldsymbol{f}$. Then, provided the inverse of Φ exists, the weights are determined by

$$\boldsymbol{w} = \Phi^{-1}\boldsymbol{f} \tag{14}$$

Many different functions $\phi(r)$ can be considered. We list some of the more common ones here:

1) Linear $\phi(r) = r$.
2) Cubic $\phi(r) = r^3$.
3) Gaussian $\phi(r) = \exp[-r^2/2\sigma^2]$, $\sigma > 0$.
4) Multiquadratics $\phi(r) = (r^2 + \sigma^2)^{1/2}$, $\sigma > 0$.
5) Inverse multiquadratics $\phi(r) = (r^2 + \sigma^2)^{-1/2}$, $\sigma > 0$.
6) Thin-plate splines $\phi(r) = r^2 \log r$.

2. Derivative Information Included

It is possible to construct a radial basis function model that not only interpolates the data, but also correctly predicts the gradient values at sampled data points $\boldsymbol{x}^{(j)}$. Suppose we define

$$\phi_{0,j}(\boldsymbol{x}) = \phi\left(\left\|\boldsymbol{x} - \boldsymbol{x}^{(j)}\right\|\right) \tag{15}$$

$$\phi_{i,j}(\boldsymbol{x}) = \frac{\partial \phi}{\partial x_i}\left(\left\|\boldsymbol{x} - \boldsymbol{x}^{(j)}\right\|\right) \tag{16}$$

where $i = 1, 2, \ldots, k$ and $j = 1, 2, \ldots, N$. Then $\{\phi_{i,j} | 0 \le i \le k, 1 \le j \le N\}$ define the basis, and the interpolation conditions for

$$\sum_{i,j} w_{i,j}\phi_{i,j}(\boldsymbol{x})$$

are

$$\sum_{i,j} w_{i,j}\phi_{i,j}\left(\boldsymbol{x}^{(s)}\right) = f\left(\boldsymbol{x}^{(s)}\right), \qquad s = 1, 2, \ldots, N \tag{17}$$

$$\frac{\partial}{\partial x_r}\left(\sum_{i,j} w_{i,j}\phi_{i,j}(\boldsymbol{x})\right)\bigg|_{\boldsymbol{x}=\boldsymbol{x}^{(s)}} = \frac{\partial f}{\partial x_r}\left(\boldsymbol{x}^{(s)}\right)$$

$$r = 1, \ldots, k, \qquad s = 1, \ldots, N \tag{18}$$

Figure 2 shows approximations to a given function f on the domain [0, 10]. Five data points are given, and a Gaussian radial basis function (RBF) approximation with and without gradients is calculated by the use of $\sigma^2 = 1$. As can be seen, the incorporation of gradient information again leads to a more accurate prediction.

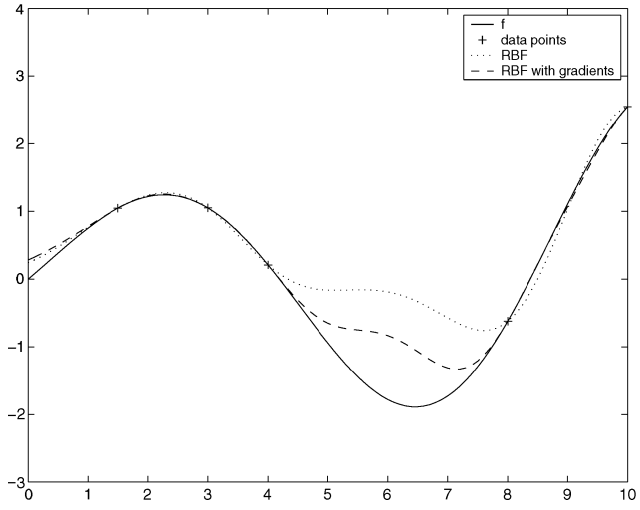


Fig. 2 RBF model.

C. Kriging

1. Derivative Information Excluded

Given a set of input data $[\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(N)}]$, together with the response $f(\boldsymbol{x})$ at these data points, the kriging model can again be used to make a prediction $F(\boldsymbol{x})$ at untested points $\boldsymbol{x}$ in the design space. The uncertainty in the approximation to f is here represented by a Gaussian stochastic process.

A correlation matrix of the data with the (i, j)th entry

$$\boldsymbol{R}\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right) = \exp\left[-d\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right)\right] \tag{19}$$

is first sought (for example, Ref. 18), where d is some distance measure. For example,

$$d\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right) = \sum_{h=1}^{k} \theta_h \left|x_h^{(i)} - x_h^{(j)}\right|^2, \qquad \theta_h \ge 0 \tag{20}$$

where θ_h are some as yet undetermined parameters.

When we wish to sample at a new point $\boldsymbol{x}$, we form a vector of correlations between the new points and the training data:

$$\boldsymbol{r}(\boldsymbol{x}) = \left[\boldsymbol{R}\left(\boldsymbol{x}, \boldsymbol{x}^{(1)}\right), \ldots, \boldsymbol{R}\left(\boldsymbol{x}, \boldsymbol{x}^{(N)}\right)\right]^T \tag{21}$$

The prediction is then given by

$$F(\boldsymbol{x}) = \mu + \boldsymbol{r}^T\boldsymbol{R}^{-1}(\boldsymbol{f} - \mathbf{1}\mu) \tag{22}$$

where $\mathbf{1}$ is an $N \times 1$ vector of ones.[26] The mean and variance of the prediction are

$$\mu = \mathbf{1}^T\boldsymbol{R}^{-1}\boldsymbol{f}/\mathbf{1}^T\boldsymbol{R}^{-1}\mathbf{1} \tag{23}$$

$$\sigma^2 = (\boldsymbol{f} - \mathbf{1}\mu)^T\boldsymbol{R}^{-1}(\boldsymbol{f} - \mathbf{1}\mu)/N \tag{24}$$

respectively.

The parameters θ_h are determined by maximization of the likelihood

$$\frac{1}{(2\pi)^{N/2}(\sigma^2)^{N/2}|\boldsymbol{R}|^{\frac{1}{2}}} \exp\left[\frac{-(\boldsymbol{f} - \mathbf{1}\mu)^T\boldsymbol{R}^{-1}(\boldsymbol{f} - \mathbf{1}\mu)}{2\sigma^2}\right] \tag{25}$$

of the sample, where $|\boldsymbol{R}|$ is the determinant of $\boldsymbol{R}$. Care must be taken when Eq. (25) is optimized because this function is often strongly multimodal. We note that this model strictly interpolates the training data.

2. Derivative Information Included

Following Morris et al.,[22] we observe the response f and its derivatives $\partial f/\partial x_1, \partial f/\partial x_2, \ldots, \partial f/\partial x_k$ at the N design points $D = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(N)}\}$ and store these in the $N(k+1)$ vector $\boldsymbol{f}$. We wish to use this information to obtain predictions to $f(\boldsymbol{x}^*)$, where $\boldsymbol{x}^*$ is some as yet unsampled input.

We again represent the uncertainty in the approximation to f (and, hence, $\partial f/\partial x_1, \partial f/\partial x_2, \ldots, \partial f/\partial x_k$) by a Gaussian stochastic process. We do not assume that errors are uncorrelated as in regression, but that the errors are correlated, the correlation between errors being related to some distance measure.

We introduce the correlation function

$$R_l\left(x_l^{(i)}, x_l^{(j)}\right) = \exp\left[-\theta_l\left(x_l^{(i)} - x_l^{(j)}\right)^2\right]$$

$$\theta_l > 0, \qquad l = 1, \ldots, k \tag{26}$$

where θ_l, $l = 1, \ldots, k$, are parameters as yet to be determined. The overall correlation is defined by the product correlation

$$R\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right) = \prod_{l=1}^{k} R_l\left(x_l^{(i)}, x_l^{(j)}\right) \tag{27}$$

Desirable properties of this correlation function are given in Ref. 18. For the derivative-based approximations, we also note that another requirement of this function is that it is at least twice differentiable. Clearly the correlation function in Eq. (26) satisfies this property.

An $N(k+1) \times N(k+1)$ correlation matrix C of the sampled data D is then defined, as in Ref. 22. This matrix is defined in terms of the correlation functions $R_l, l = 1, \ldots, k$, as well as their first and second derivatives $R_l', R_l'', l = 1, \ldots, k$. Similarly, an $N(k+1)$ vector of correlations $\boldsymbol{r}$, between a new point $\boldsymbol{x}^*$ and the already sampled data D can be defined. For full details of this procedure, we refer the reader to Ref. 22.

The hyperparameters $\theta_1, \ldots, \theta_k$ are again chosen to maximize the likelihood of the sample. The log-likelihood is expressed as

$$L(\theta) = -N(k+1)\,\ell_n\sigma^2 - \ell_n|C| - (1/\sigma^2)(\boldsymbol{f} - \boldsymbol{v}\mu)^T C^{-1}(\boldsymbol{f} - \boldsymbol{v}\mu) \tag{28}$$

where μ and σ^2 represent the mean and variance of the data and $\boldsymbol{v}$ is an $N(k+1)$ binary vector with 1 in position $(i-1)(k+1)+1, i = 1, \ldots, N$, and 0 everywhere else.

For fixed θ, maximization of L over μ and σ^2 is obtained by

$$\hat{\mu} = \boldsymbol{v}^T C^{-1}\boldsymbol{f} / \boldsymbol{v}^T C^{-1}\boldsymbol{v} \tag{29}$$

$$\hat{\sigma}^2 = [1/N(k+1)](\boldsymbol{f} - \boldsymbol{v}\mu)^T C^{-1}(\boldsymbol{f} - \boldsymbol{v}\mu) \tag{30}$$

Substituting Eqs. (29) and (30) into Eq. (28), we obtain a function of $\theta_l, l = 1, \ldots, k$, only this time we maximize to obtain $\hat{\theta}$ and, hence, an estimate of the overall correlation matrix C.

Formulas (29) and (30) then provide us with an estimate of $\hat{\mu}$ and $\hat{\sigma}^2$.

The predictor at an unsampled point is then given by Ref. 22:

$$F(\boldsymbol{x}^*) = \hat{\mu} + \boldsymbol{r}^T C^{-1}(\boldsymbol{f} - \boldsymbol{v}\hat{\mu}) \tag{31}$$

If $\boldsymbol{x}^*$ corresponds to a sampled point $\boldsymbol{x}^{(i)}$, then $\boldsymbol{r}$ corresponds to the ith row of C; as a result, the model interpolates the data. The advantage of this model over ordinary kriging models is that the model also gives the correct gradient at a sampled point, and, as a result, predictions can be far more accurate. Its disadvantage is that the correlation matrix of the sampled data is larger and, hence, the maximization of Eq. (28) is more expensive, particularly for large k.

Figure 3 shows approximations to a given function f on the domain [0, 10]. Five data points are given, and a kriging approximation with and without gradients is calculated by the use of hyperparameters obtained by maximization of the likelihood. Once again, the incorporation of gradient information leads to a more accurate prediction. (The original function is almost indistinguishable from the kriging approximation that uses derivative information.)

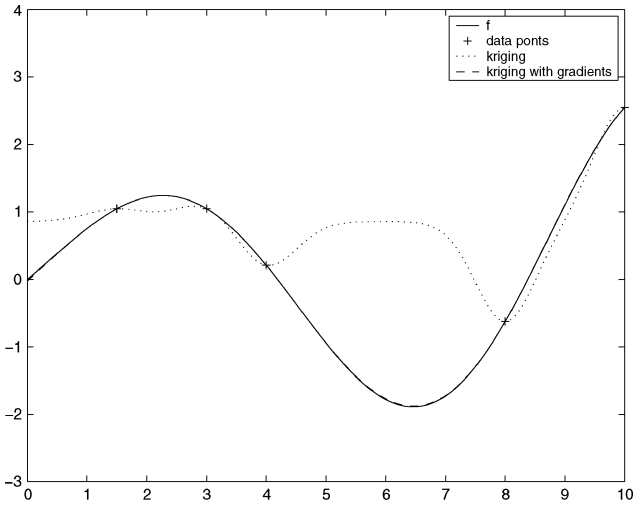


Fig. 3 Kriging model.

III. Approximating the Output of a Potential Flow Code

A. Problem Definition

In this section, ψ relates to a scalar potential whose gradient gives the velocity field in a fluid flow problem. We consider steady two-dimensional compressible, inviscid flow around a gas-turbine inlet nacelle as a demonstrative example. The vector velocity field is $\boldsymbol{u}$. Further more, under the assumption that the flow is irrotational, a scalar potential ψ can be written by

$$\boldsymbol{u} = \nabla\psi \tag{32}$$

and the flow is described as potential flow.

We solve the steady continuity equation

$$\nabla \cdot (\rho\nabla\psi) = 0 \tag{33}$$

by a finite element method. Suitable boundary conditions reflecting far-field behavior, zero flow through solid boundaries, and input mass flow into the engine input are enforced. The flow is subsonic throughout the entire domain of interest. We note that with this type of solver, there is virtually no noise in the model's response, and so interpolation is a sensible option. If we were to use, for example, a Reynolds-averaged Navier–Stokes code, noise may well be present, and some form of regression should be considered. We do not address the issue of noise in this paper; this will be an area of future research.

The inlet geometry (nacelle) is defined when two Hicks-Henne bump functions[27] are added to a fixed baseline geometry, one to the upper and one to the lower half of the nacelle. Each Hicks–Henne bump function is defined by three design variables, namely, the height of the bump, the position of its peak, and the width of the bump. Thus, there are six design variables in all. These are varied between the upper and lower bounds shown in Table 1.

Note that for the purposes of constructing approximations, these design variables are scaled so that the inputs $x_i \in [0, 1]$. An example of an inlet design and the computational mesh used in these studies are shown in Figs. 4 and 5, respectively.

Table 1 Design variables

Description	Parameter	Lower bound	Upper bound
Height (lower)	x_1	−0.15	0.05
Position (lower)	x_2	0.4	0.8
Width (lower)	x_3	2	10
Height (upper)	x_4	0	0.15
Position (upper)	x_5	0.5	0.85
Width (upper)	x_6	2	5

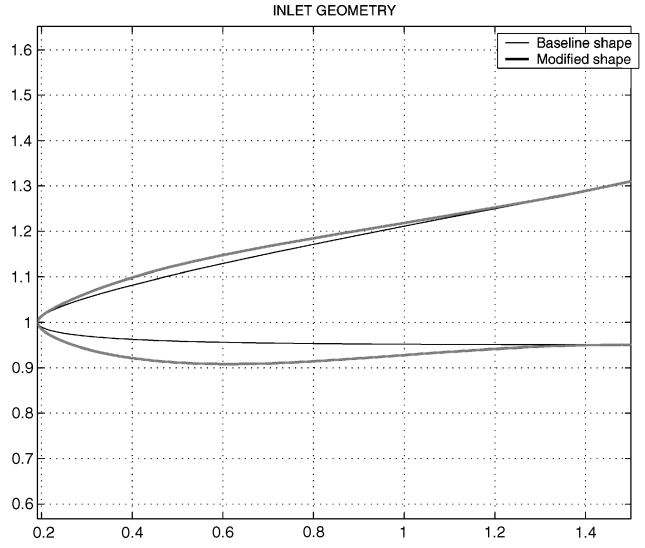


Fig. 4 Example of nacelle geometry.

Two objective functions were defined. The first relates to a target velocity defined around the nacelle, where we minimize the difference between our actual velocity and this target velocity. Second, we attempt to minimize the peak velocity along the nacelle. Two problems were considered, a two-design-variable test case where variables x_2 and x_5 are altered and the rest are set to a predefined value, and a six-variable problem where all of the design variables can be altered.

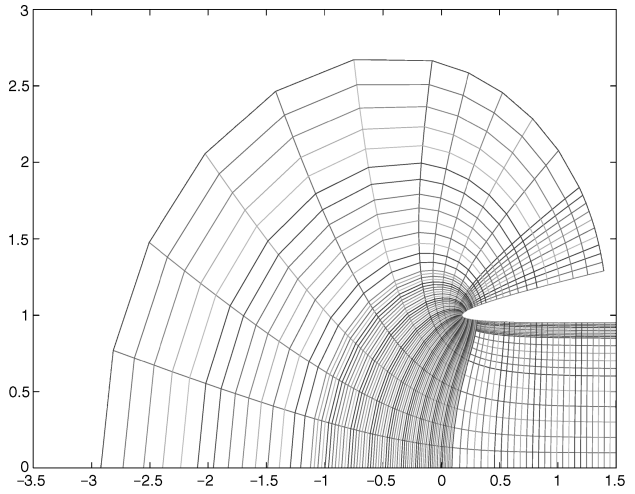


Fig. 5 Computational mesh used in this study.

For the purposes of assessing the accuracy of our approximations, we first construct a design of experiments consisting of 200 LP_τ runs[28] for both the two- and six-design-variable study. (See Ref. 29 for a detailed description of, and a detailed algorithm for the implementation of the construction of, LP_τ sequences.) The adjoint CFD code is then run at these locations to give objective function values and sensitivities at these locations in our design space.

Because LP_τ sequences are built up sequentially, we can then use a subset of these 200 runs for the construction of an approximation to our objective. We consider a nongradient approximation at N points and use the remaining $200 - N$ points for the purposes of assessing our model's accuracy. Because an adjoint solver requires the same computational effort as one CFD evaluation, we compare the adjoint solver with a gradient-based approximation at $N/2$ points. The same $200 - N$ testing points assess the accuracy of the model. When considering the two-design-variable models, we take $N = 10, 20, \ldots, 50$, whereas for the six-design-variable models we take $N = 10, 20, \ldots, 100$.

The results of application of the three approximation methods discussed to the problems described are shown in Figs. 6–9. Here, we consider two radial basis functions, a Gaussian and an invervse multiquadratic. Figures 6–9 show the average and maximum errors in our approximation model (when compared to the aforementioned testing data), as well as correlations between our approximation model and our expensive analysis code (again using the testing data). The correlation is defined by

$$r^2 = S_{xy} \Big/ \sqrt{S_{xx} S_{yy}} \tag{34}$$

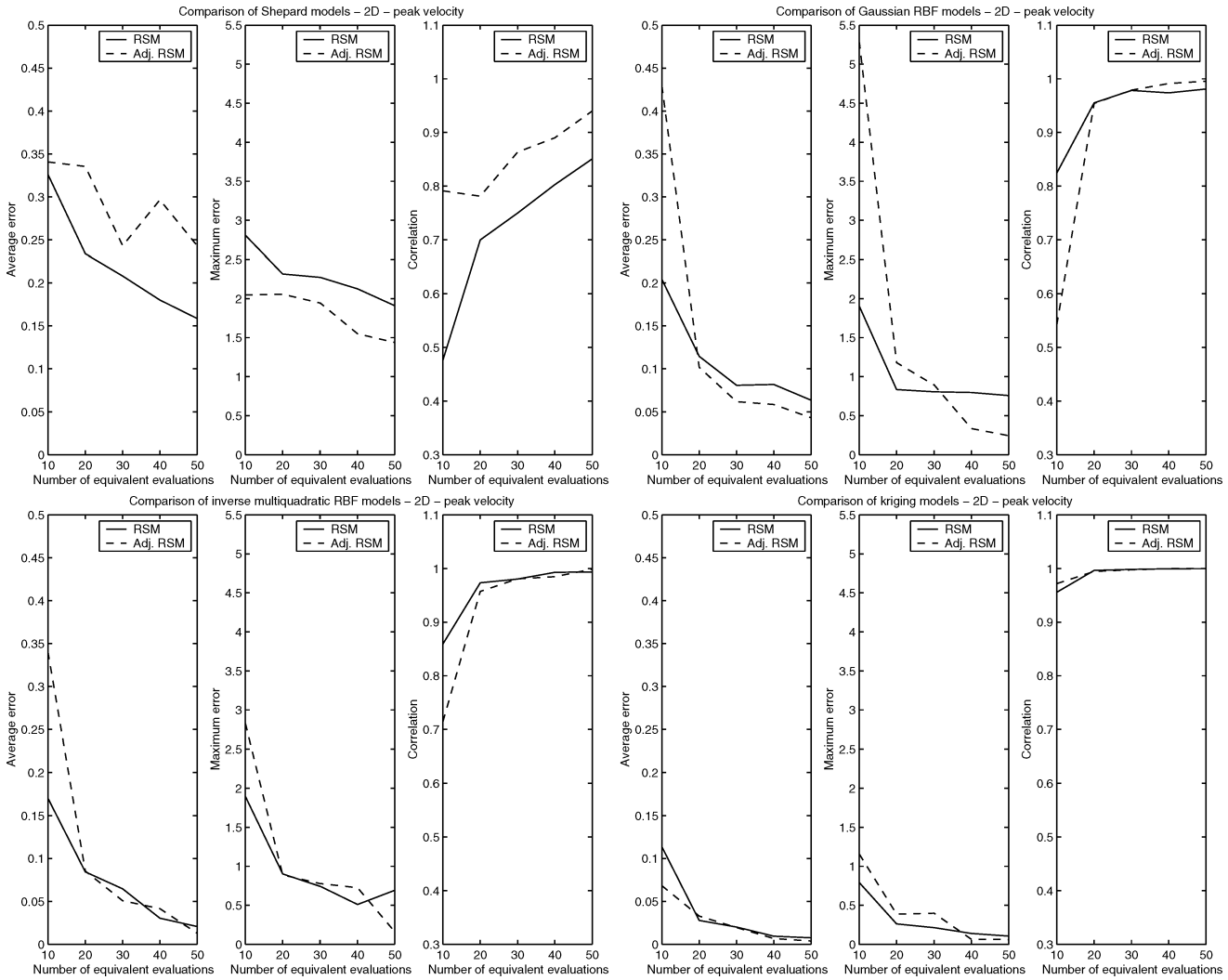


Fig. 6 Results for two-dimensional modeling; peak velocity.

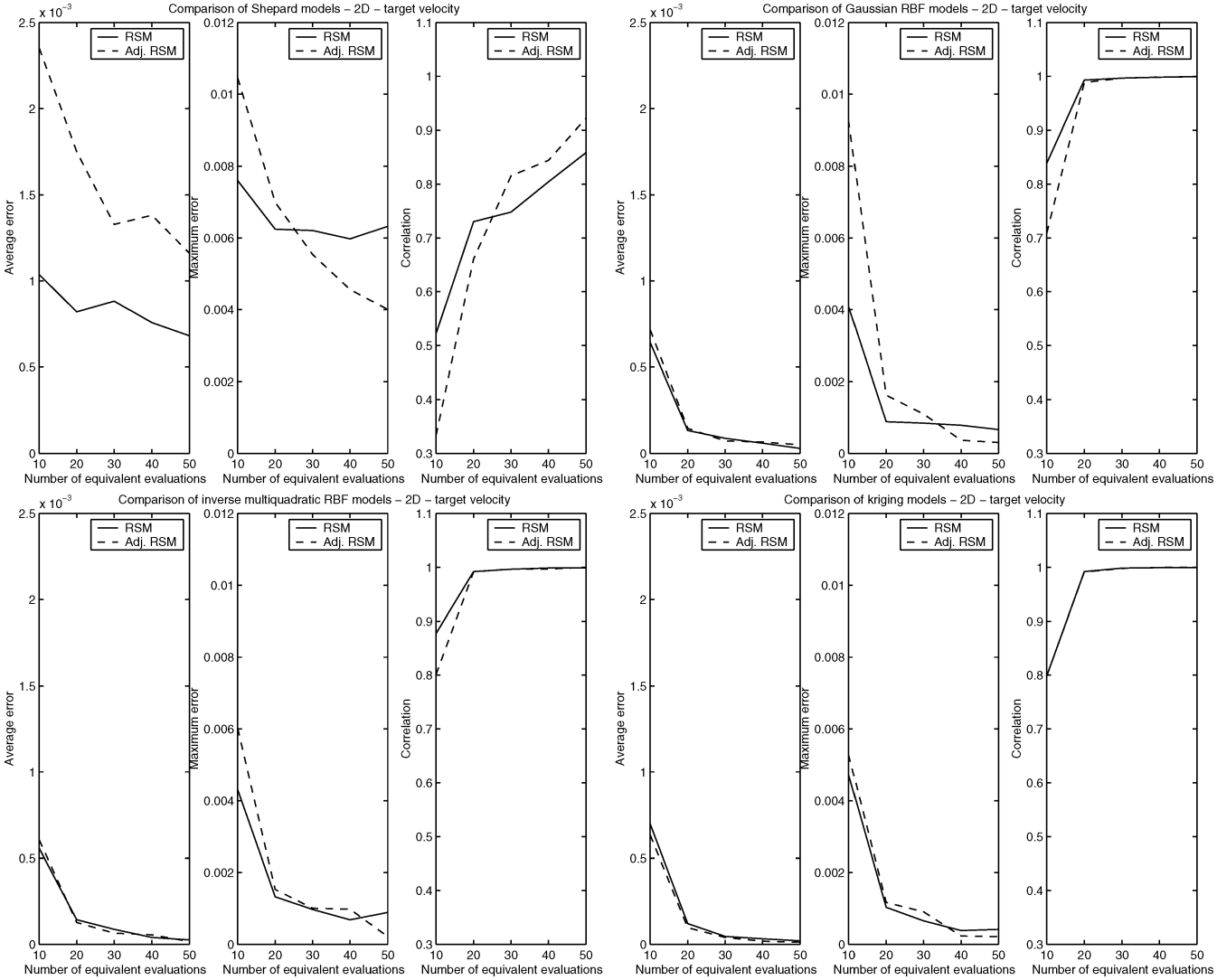


Fig. 7 Results for two-dimensional modeling; target velocity.

where

$$S_{xx} = \sum_{i=1}^{N} y\left(\boldsymbol{x}^{(i)}\right)^2 - \left[\sum_{i=1}^{N} y\left(\boldsymbol{x}^{(i)}\right)\right]^2 \Big/ N \tag{35}$$

$$S_{yy} = \sum_{i=1}^{N} \hat{y}\left(\boldsymbol{x}^{(i)}\right)^2 - \left[\sum_{i=1}^{N} \hat{y}\left(\boldsymbol{x}^{(i)}\right)\right]^2 \Big/ N \tag{36}$$

$$S_{xy} = \sum_{i=1}^{N} y\left(\boldsymbol{x}^{(i)}\right)\hat{y}\left(\boldsymbol{x}^{(i)}\right) - \left[\sum_{i=1}^{N} y\left(\boldsymbol{x}^{(i)}\right)\right]\left[\sum_{i=1}^{N} \hat{y}\left(\boldsymbol{x}^{(i)}\right)\right] \Big/ N \tag{37}$$

and a correlation near zero indicates a bad fit, whereas a correlation near one indicates a good fit.

B. Comparison of Approximate Models

1. *Gradient vs Nongradient Approximations*

The approaches are considered with adjoint solvers in mind; therefore, comparisons are made by the use of the function's response only at N points and the function's response only at $N/2$ points, together with $N/2$ adjoint solutions. The latter is of a similar computational cost as an adjoint solution is of a similar cost to solving the original equations. The former we refer to as a nongradient-based approach, whereas the latter we refer to as a gradient-based approach. The nongradient-based approach gives us N pieces of information spread evenly throughout the design space. With k design variables, an adjoint solve gives us k pieces of information (one for each design variable); thus, we have $N(k+1)/2$ pieces of information in all. However, these are clustered into $N/2$ space filling groups of $k+1$. An example of these in two design variables with $N = 10$ is shown in Fig. 10.

When k is relatively large, the gradient-based model provides us with much more information than the nongradient approach. As a result, we might expect our results to be more accurate: This is the case in the six-design-variable problem where the gradient-based approximation usually significantly outperforms the nongradient-based approach. (See Figs. 8 and 9, and note that the gradient-based approximations typically have lower average and maximum errors and higher correlations when compared against a set of testing data.) The differences when the two-design-variables are used is not so large (Figs. 6 and 7).

2. *Comparison of the Approximation Methods*

The first technique discussed was the Shepard weighting function. The main advantage of this method is that it is very cheap to compute, both with and without gradient information, because no matrix inversions are required. One open question is the best choice of R_w. It has been held constant here but could be determined by the use of an approach such as leave-one-out cross validation. The disadvantage of the method is its relative inaccuracy: Predictions can be quite

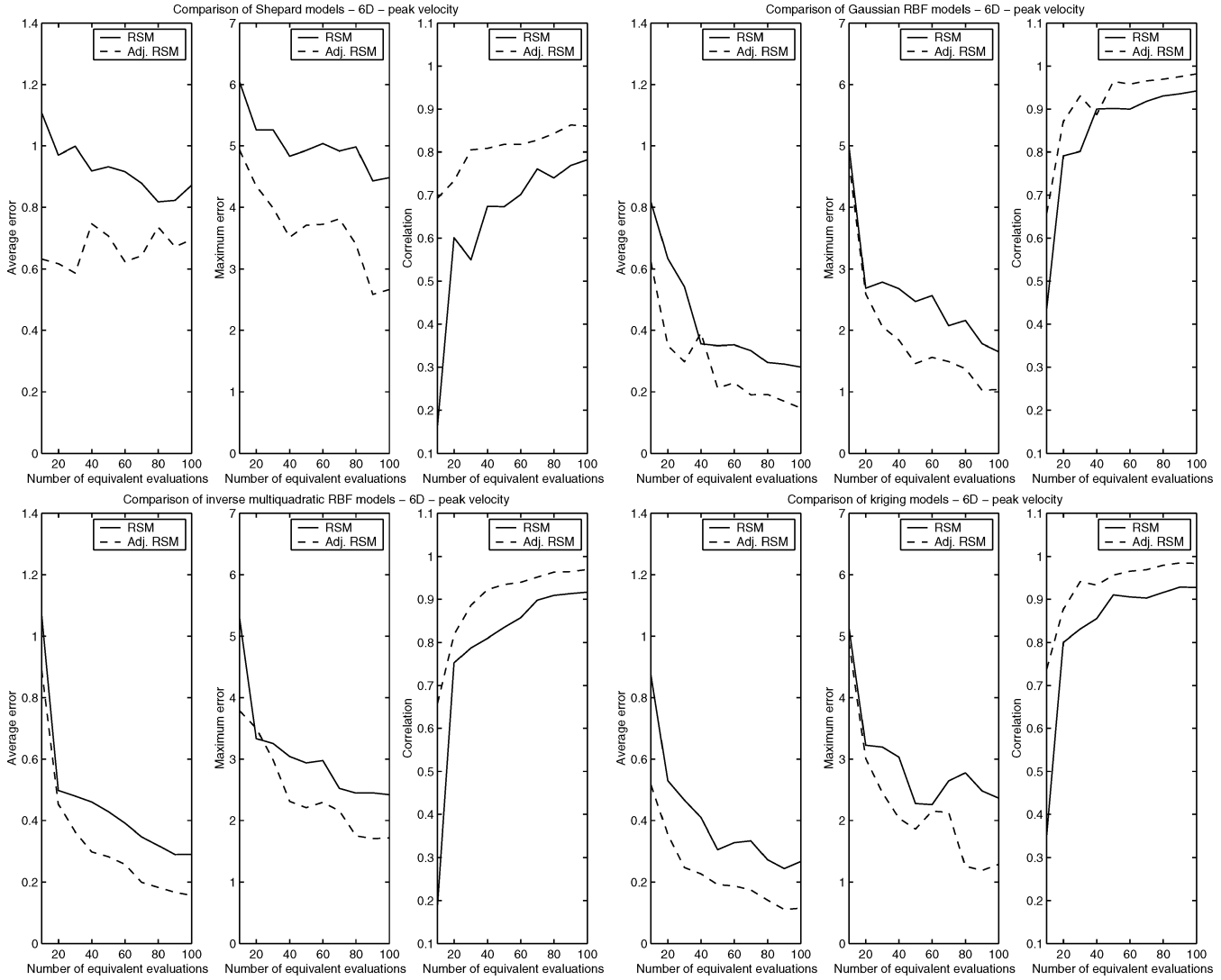


Fig. 8 Results for six-dimensional modeling; peak velocity.

poor. In general, they have larger average and maximum errors and smaller correlations when compared to a set of testing data derived by other approximation methods (Figs. 6–9). The incorporation of gradient information can help somewhat. (This always reduced the maximum error and increased the correlation between the prediction and a set of testing data, although sometimes the average error was increased.)

Radial basis functions with constant σ^2 are more computationally expensive. By the use of the nongradient approach, this requires that an $N \times N$ matrix inversion be performed to construct the approximation. The gradient-based approach is seen to improve the accuracy of the prediction as the number of design variables increases. [Lower average and maximum errors and larger correlations against a set of testing data was, more often than not, observed (Figs. 6–9).] However, the cost of constructing the approximation is increased because we now require a $[N(k+1)/2] \times [N(k+1)/2]$ matrix inversion.

There are other factors to bear in mind when a radial basis function approximation is used, namely, 1) the choice of basis function and 2) the choice of σ^2 (if required).

The former is not considered here. However, the second partial derivative of $\phi(\|\boldsymbol{x} - \boldsymbol{x}^{(i)}\|)$ with respect to x_k must exist. The basis function and its first and second derivatives must also be bounded above and below over the domain of interest. Our two examples, a Gaussian radial basis function and an inverse multiquadratic do indeed satisfy this property; however, others do not. For example, a thin-plate spline basis $\phi(d) = d^2 \ln d$ has second derivative $\phi''(d) = 2 \ln d + 3$. This is unbounded at $d = 0$. We also note that these conditions themselves are not sufficient because a linear basis function satisfies the property, however, in this case, gradient-enhanced approximations cannot be constructed.

The choice of σ^2 is perhaps more important because this can dramatically affect the quality of prediction. Here, we opt for one of three choices 0.1, 1, or 10, chosen by a simple search. On the scaled inputs, these were seen to work relatively well. However σ^2 could again be chosen with an optimization procedure that uses, for example, leave-one-out cross validation, to define the objective. The search for an ideal value of σ^2 would result in a much more expensive approximation, and, because the purpose here is to try to make an efficient approximation, we did not consider this option further.

Finally, a kriging model was considered. This technique was the most expensive in terms of model construction, requiring inversion of an $N \times N$ matrix throughout maximum likelihood optimization in the case of the nongradient-based approach. It is even more costly for the gradient-based approach, requiring inversion of an $[N(k+1)/2] \times [N(k+1)/2]$ matrix throughout the maximum likelihood optimization. Whether or not this is practical depends on the expense of the computer model we are approximating, but, as a rough guideline, we find training can be very expensive if $[N(k+1)/2] > 1000$, or if the number of variables and, hence, hyperparameters is large. This was not much of an issue in the example in this paper, but it is certainly something the reader should be aware of. Some suggestions as to what to do if $[N(k+1)/2] > 1000$ can be found in Sec. V.

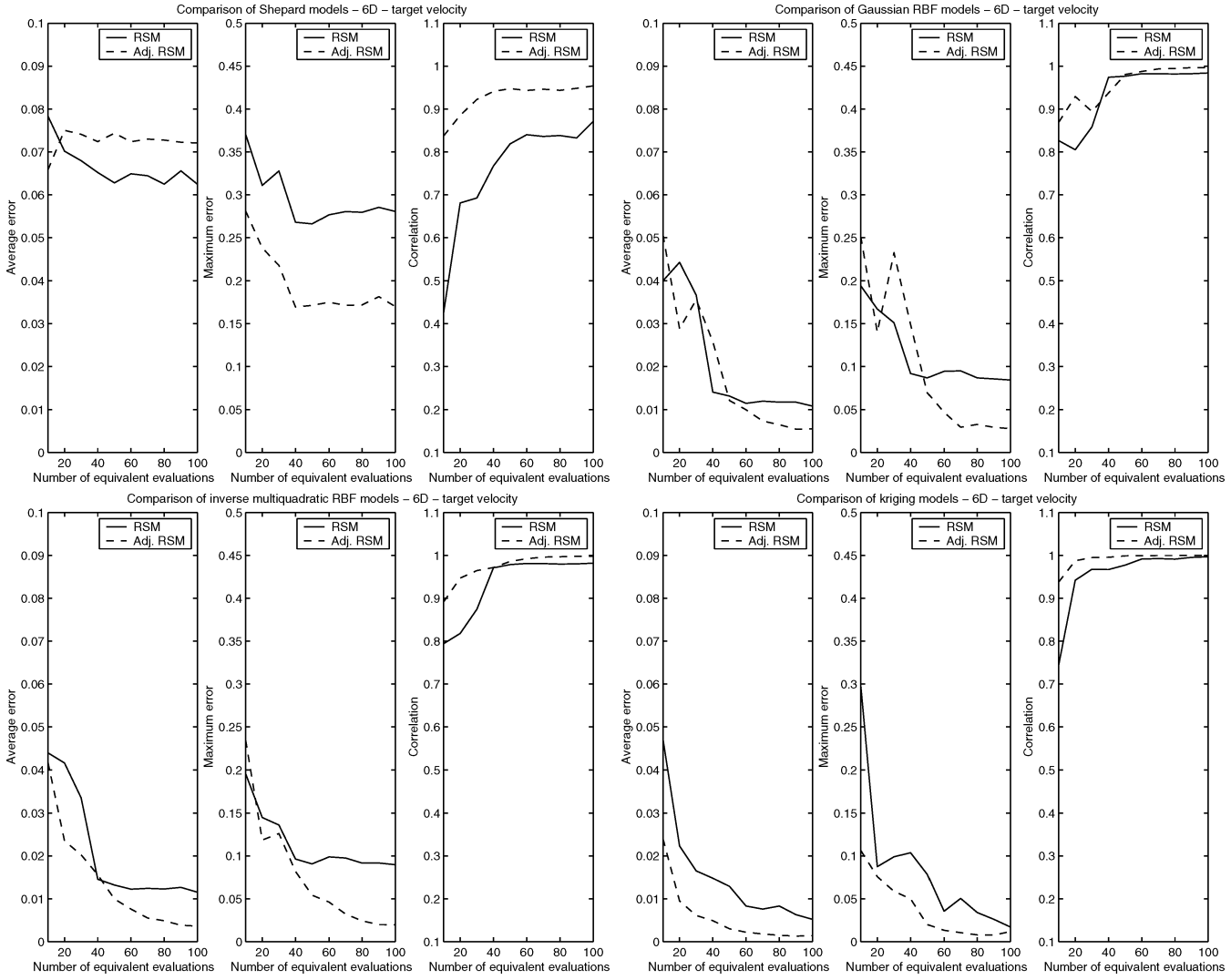
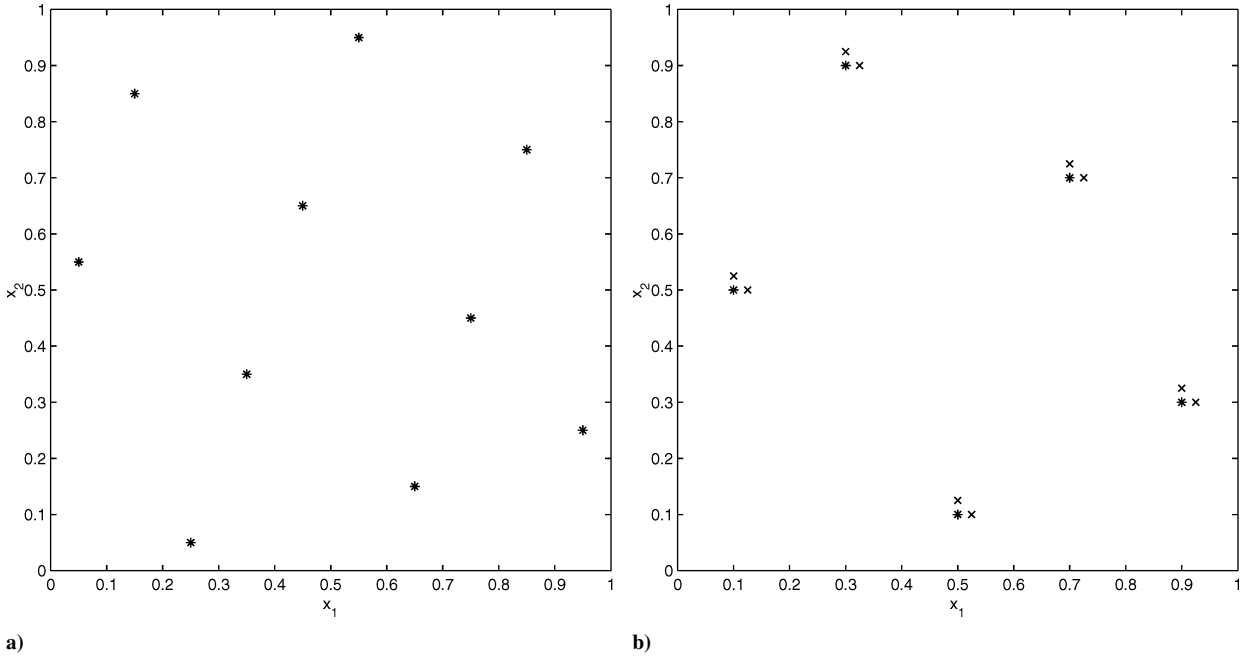


Fig. 9 Results for six-dimensional modeling; target velocity.



a) **b)**

Fig. 10 Design of experiments for a) nonadjoint model with $N = 10$ and b) adjoint model with $N = 5$.

The advantage of the kriging approach is that because the hyperparameters are tuned to the data, the approximations can be very accurate. The traditional kriging model was the most accurate nongradient approximation. The gradient-based kriging model produced the most accurate gradient-based approximation. As the number of design variables increased, this was seen to be the most accurate approximation of all (Figs. 6–9).

Once more, there is a choice to be made regarding which correlation function is to be used. Several functions are described in Ref. 30. Again, a requirement for the gradient-based approximation is that the correlation function be twice differentiable. We have opted here for the most popular correlation function, a Gaussian.

IV. Optimization

The assumption in this paper is that the model under investigation is computationally expensive, so that direct optimization is unrealistic. However, we have chosen to study a simple potential flow code with a run time of the order of 1 min because we can then compare our results with direct optimization strategies. Of course, such a study would generally not be possible. We do it here simply to assess the applicability of our approach to adjoint CFD codes. The direct optimization in this section was performed with a BFGS optimization routine.[31] The gradients required for this optimizer are provided when the adjoint equations are solved.

We compare the results of direct optimization with the use of nonadjoint-based and adjoint-based surrogate models. First, an N-point design of experiments is defined by the use of LP_τ sequences. The potential code is then run at these locations to gain some information on the model. An approximation is then constructed with the approximation method. This approximation can then replace the original analysis code for the purposes of optimization.

Because the surrogate surface is inexpensive, we perform a detailed stochastic search on this surface. Once an optimum is found, we evaluate the full potential code at this point, reconstruct a new approximation, and repeat. This process can be repeated until we have in some sense converged, generally when the distance between two successive design points is within a certain tolerance, or until the maximum number of allowable calls to the expensive model is reached. This procedure is shown schematically in Fig. 11.

As a direct comparison, we also consider a gradient-based approximation method, initially by the use of $N/2$ computer experiments.

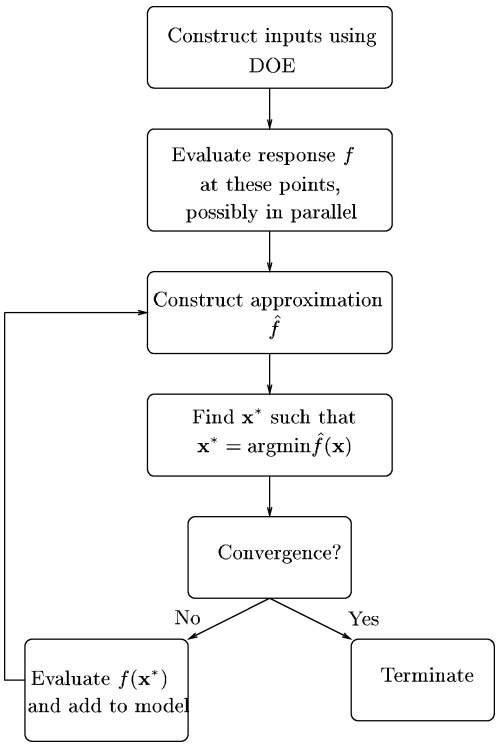


Fig. 11 Nonderivative-based surrogate optimization strategy.

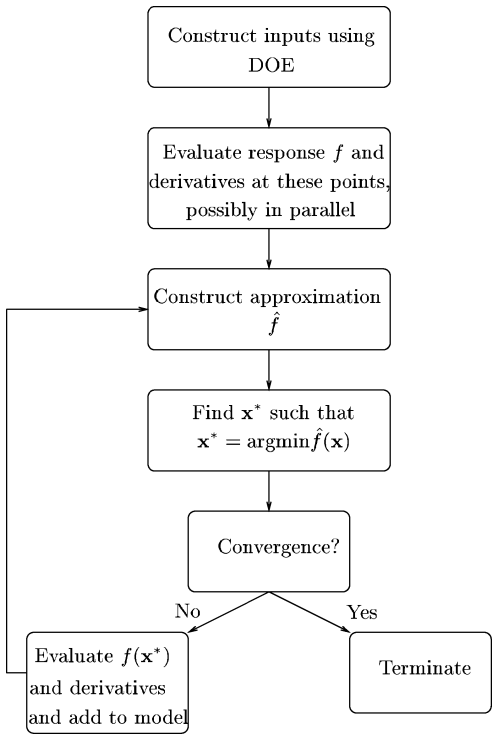


Fig. 12 Derivative-based surrogate optimization strategy.

The potential flow code and its adjoint are then evaluated at these points to gain some information on the model under consideration. An approximation is constructed and again optimized by stochastic search. Once the optimum is found, the objective and its gradients are computed (with the potential code and its adjoint), and the process is repeated. This is shown schematically in Fig. 12. We refer to these approaches as direct surrogate optimization.

We note that with these approaches, we run the risk of becoming trapped in a local minimum of the approximating surface. With kriging, for a nongradient-based surrogate model, an elegant global optimization algorithm that balances optimization of the surrogate model, together with the uncertainty in this model, is described in Ref. 18. This is generalized to gradient-based surrogate models in Ref. 23. We can view this schematically, as in Figs. 11 and 12, only replacing the step "optimize the approximation" with the step "optimize the expected improvement."

Results of optimization are shown in Figs. 13 and 14. Here, we consider the full six-design-variable problem using direct optimization (BFGS) and direct surrogate optimization. In this study, we used kriging and derivative-based kriging for the purposes of constructing the surrogate models. N was arbitrarily taken as 40.

For the surrogate-based optimization, it is possible to encounter problems with ill conditioning of the correlation matrices (19) and (27). This can particularly occur close to convergence, when sampled points become close together. As a result, we adopt the following convergence criterion here: We scale the inputs so that they lie in [0,1], and once any two sampled points are within a distance (taken here as 0.02), we terminate the search. With this strategy, satisfactory convergence was observed, and no problems with ill conditioning were encountered.

When both peak velocity and a target velocity are considered as objectives, the convergence histories of the BFGS are averaged over 10 runs by the use of different starting points. With this optimizer, only new function evaluations are logged in Figs. 13 and 14. If, during the course of the search, an old point is revisited, then its function value is extracted from the database of existing runs. The results of all optimization strategies follow.

A. Peak Velocity

For this objective, the BFGS optimizer performed reasonably well. From an average starting objective function value of 15.8021,

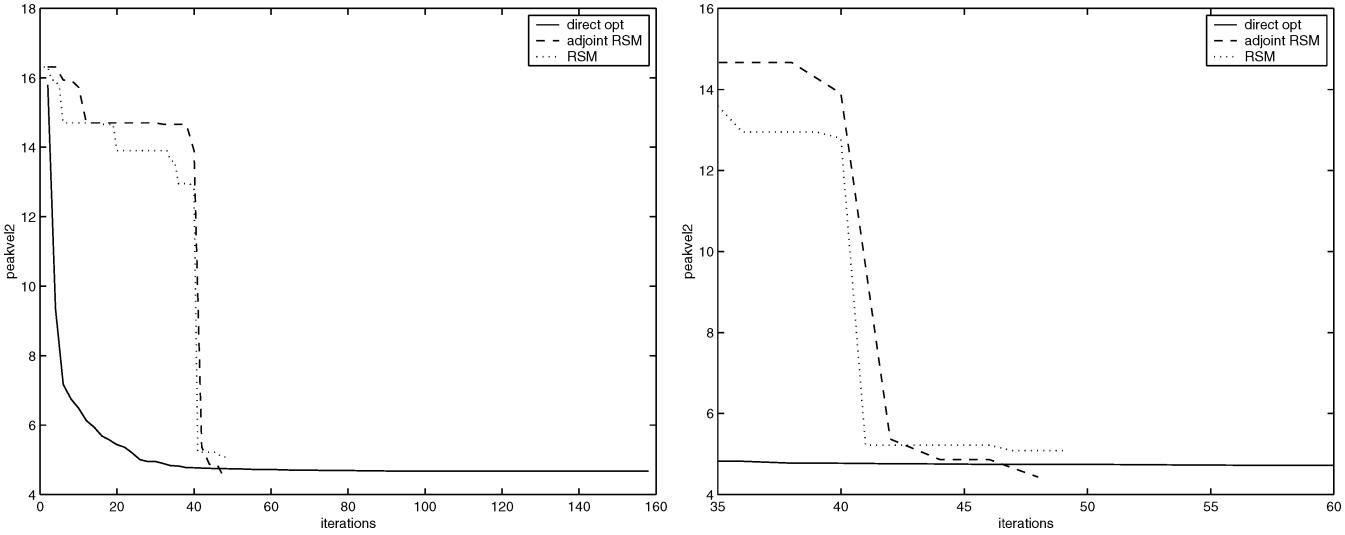


Fig. 13 Results for six-dimensional optimization; peak velocity: a) full search and b) zoomed plot.

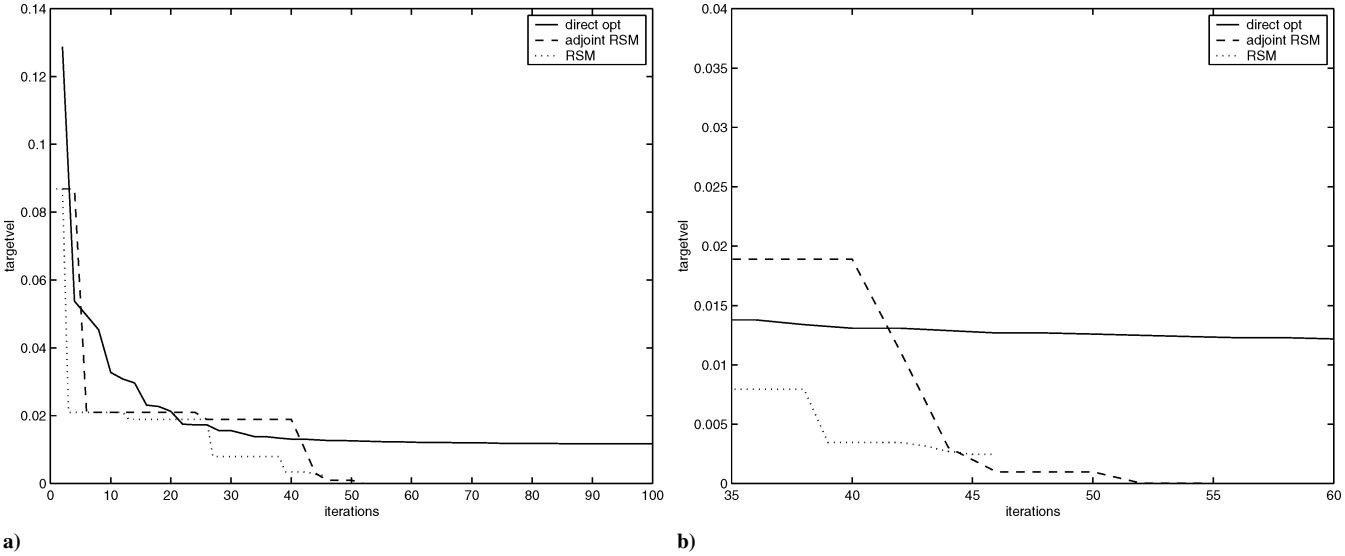


Fig. 14 Results for six-dimensional optimization; target velocity: a) full search and b) zoomed plot.

it reduced this value to an average of 4.6771 in 158 evaluations (79 function evaluations and 79 adjoint evaluations for the gradients), and good solutions were identified in the early stages of the search.

For both of the surrogate optimizations, the convergence histories shown in Fig. 13 include the design of experiments (DOE) phase (first 40 iterations). In this exploratory phase, the kriging model performs better than the derivative-based kriging model in terms of best objective function value found because it considers N evaluations of this objective spread throughout the design space. The adjoint based kriging model only considers $N/2$ evaluations (the first $N/2$ of the N nongradient model) plus $N/2$ adjoint evaluations for the gradients.

Once the optimization phase starts, dramatic improvements are quickly made; the derivative-based kriging model converges to a value of 4.4312 in eight further iterations (four function and four adjoint evaluations), requiring a total of 48 iterations (24 function and 24 adjoint evaluations). The kriging model without gradients converges to an objective function value of 5.0827 after a total of 49 iterations (49 objective function values). We see that once the optimization phase starts, kriging with and without gradient information is very efficient for optimization; however, as the model built without gradients clearly shows, we may not converge to the true minimum of the original response. The most effective search in this instance was the gradient-enhanced kriging model, which found the best solution value at a cost of only 48 iterations.

B. Target Velocity

For this objective, the BFGS optimizer performed poorly. From an average starting objective function value of 0.1288, it reduced this to an average of 0.0117, converging after 100 iterations (50 objective function plus 50 adjoint evaluations).

For both surrogate-based optimization approaches, the DOE stage is again included in the convergence histories, and in this exploratory phase the model without gradient information included once more performs better for the reasons stated earlier.

Once the optimization phase starts, dramatic improvements are again observed. The derivative-enhanced kriging model has not quite converged after 20 more iterations (10 more objective functions and 10 more adjoint evaluations), but an objective function value of 9.508e-7 was achieved. The kriging model converged to a value of 2.472e-3 after six further iterations (requiring a total of 46 objective function evaluations). In this example, both surrogate optimization strategies outperform the direct search. Once again, the gradient enhanced kriging model was seen to perform best of all.

V. Conclusions

Approximation methods that utilize gradient information have been presented. The primary motivation is that adjoint CFD codes give this gradient information cheaply, and so it is natural to include it inside an approximation.

Three approximation methods were considered, and gradient- vs nongradient-based approximations of equal cost have been compared. It is seen that as the dimensionality of the problem increases, the use of gradient information becomes more and more advantageous, leading to predictions that have low error and that are highly correlated when compared with a set of testing data.

With regard to the approximation methods, the kriging model was seen to be the most accurate. The radial basis function (with constant σ^2) also led to reasonable predictions at cheaper cost. The Shepard weighting functions, although even cheaper to construct, appear to be relatively inaccurate, particularly in higher dimensions.

Although not an issue in the example described in this paper, the main drawback of the kriging model is the cost of determining the hyperparameters. This is especially true when we consider gradient-based approximations. If the size of the correlation matrix is large, maximum likelihood estimation can become computationally very expensive. As a general rule, we train the model if this matrix has size <1000, that is, $N(k+1)/2 < 1000$ for gradient-based approximations. Note also that if we have many variables, and, hence, hyperparameters to train, this further increases the difficulty in training a kriging model. It may be possible to consider a larger matrix than this because, ultimately, how long we can realistically spend training our kriging model will depend on the expense of our original analysis code. Note, however, that when data are collected to build approximations, it is trivial to carry these out in parallel. Optimization of the kriging hyperparameters is much less simple to handle in this way.

One possible way to overcome the aforementioned limitation would be to consider a statistical screening study to highlight the most important design variables. It may be the case that only a subset $k^* < k$ of these design variables are important, and so we could reduce the dimensionality of the problem. As a result, we would also need fewer training data N^* and fewer hyperparameters. We would then consider kriging, as long as $N^*(k^*+1)/2 < 1000$. Here, we assume nothing about the behavior of the response, and so a model-independent screening strategy should be considered. If it were not possible to reduce k or N, and $N(k+1)/2 > 1000$, we would recommend a radial basis function approximation with constant σ^2.

With regard to screening studies, they also traditionally utilize function values only and make no use of gradients. With the advent of adjoint methods, an interesting research area in applied statistics would be the development of global screening strategies that incorporate local sensitivity information.

The use of adjoint codes is well established in the field of local optimization, where it is computationally much more efficient than a finite difference approach. We have tried to show how adjoint codes can be utilized in a global optimization framework. Here, the computationally expensive model is replaced by an inexpensive surrogate. Surrogate modeling can vastly improve the efficiency of the optimization process. Moreover, the surrogate models incorporating gradient information were seen to outperform the accuracy of the nongradient-based surrogate models because of their added information when this approach is employed.

Acknowledgments

We thank Rolls–Royce plc. for the financial support received as a part of the University Technology Partnership in design. We also thank the Oxford University University Technology Centre for providing the example used in Sec. III.

References

[1]Jameson, A., "Aerodynamic Design via Control Theory," *Journal of Scientific Computing*, Vol. 3, 1988, pp. 233–260.

[2]Jameson, A., "Optimum Aerodynamic Design Using Computational Fluid Dynamics and Control Theory," AIAA Paper 95-1729, July 1995.

[3]Jameson, A., "Optimum Aerodynamic Design Using Control Theory," *Computational Fluid Dynamics Review*, Wiley, New York, 1995, pp. 495–528.

[4]Reuther, J., and Jameson, A., "Control Based Airfoil Design Using the Euler Equations," AIAA Paper 94-4272, 1994.

[5]Reuther, J., Jameson, A., Farmer, J., Martinelli, L., and Saunders, D., "Aerodynamic Shape Optimization of Complex Aircraft Configurations via an Adjoint Formulation," AIAA Paper 96-0094, Jan. 1996.

[6]Jameson, A., Pierce, N., and Martinelli, L., "Optimum Aerodynamic Design Using the Navier–Stokes Equations," *Theoretical and Computational Fluid Dynamics*, Vol. 10, 1998, pp. 213–237.

[7]Jameson, A., "Reengineering the Design Process Through Computation," *Journal of Aircraft*, Vol. 36, 1999, pp. 36–50.

[8]Reuther, J., Jameson, A., Alonso, J. J., Rimlinger, M. J., and Saunders, D., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers, Part 1," *Journal of Aircraft*, Vol. 36, 1999, pp. 51–60.

[9]Reuther, J., Jameson, A., Alonso, J. J., Rimlinger, M. J., and Saunders, D., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers, Part 2," *Journal of Aircraft*, Vol. 36, 1999, pp. 61–74.

[10]Jameson, A., and Vassberg, J., "Computational Fluid Dynamics (CFD) for Aerodynamic Design: Its Current and Future Impact," AIAA Paper 2001-0538, Jan. 2001.

[11]Cox, S. J., Haftka, R. T., Baker, C. A., Grossman, B., Mason, W. H., and Watson, L. T., "A Comparison of Global Optimization Methods for the Design of a High-Speed Civil Transport," *Journal of Global Optimization*, Vol. 21, 2001, pp. 415–433.

[12]Goffe, W. L., Ferrier, G. D., and Rogers, J., "Global Optimization of Statistical Functions with Simulated Annealing," *Journal of Econometrics*, Vol. 60, 1994, pp. 65–99.

[13]Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley Longman, Reading, MA, 1988, pp. 1–54.

[14]Myers, R. H., and Montgomery, D. C., *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*, Wiley, New York, 1995, pp. 1–67.

[15]Conn, A. R., Gould, N. I. M., and Toint, P. L., *Trust-Region Methods*, Society for Industrial and Applied Mathematics, Philadelphia, 2000.

[16]Shepard, D., "A Two Dimensional Interpolation Function for Irregularly Spaced Data," *Proceedings of the 23rd National Conference*, Association for Computing Machinery, New York, 1968, pp. 517–523, Chaps. 1 and 2.

[17]Powell, M. J. D., "Radial Basis Functions for Multivariable Interpolation: A Review," *Algorithms for Approximation*, Clarendon, Oxford, 1987, pp. 143–167.

[18]Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization*, Vol. 13, 1998, pp. 455–492.

[19]Chung, H. S., and Alonso, J. J., "Using Gradients to Construct Response Surface Models for High-Dimensional Design Optimization Problems," AIAA Paper 2001-0922, Jan. 2001.

[20]Rodriguez, J. F., Renaud, J. E., and Watson, L. T., "Trust Region Augmented Lagrangian Methods for Sequential Response Surface Approximation and Optimization," *Journal of Mechanical Design*, Vol. 120, 1998, pp. 58–66.

[21]Alexandrov, N. M., Dennis, J. E., Jr., Lewis, R. M., and Torczon, V., "A Trust-Region Framework for Managing the Use of Approximations in Optimization," *Structural Optimization*, Vol. 15, 1998, pp. 16–23.

[22]Morris, M. D., Mitchell, T. J., and Ylvisaker, D., "Bayesian Design and Analysis of Computer Experiments: Use of Derivatives in Surface Prediction," *Technometrics*, Vol. 35, 1993, pp. 243–255.

[23]Leary, S. J., Bhaskar, A., and Keane, A. J., "A Derivative Based Surrogate Model for Approximating and Optimizing the Output of an Expensive Computer Simulation," *Journal of Global Optimization* (to be published).

[24]Chung, H. S., and Alonso, J. J., "Using Gradients to Construct Cokriging Approximation Models for High-Dimensional Design Optimization Problems," AIAA Paper 2002-0317, Jan. 2002.

[25]Renka, R. J., "Multivariate Interpolation of Large Sets of Scattered Data," *ACM Transactions on Mathematical Software*, Vol. 14, No. 2, 1988, pp. 139–148.

[26]Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and Analysis of Computer Experiments," *Statistical Science*, Vol. 4, No. 4, 1989, pp. 409–435.

[27]Hicks, R. M., and Henne, P. A., "Wing Design by Numerical Optimization," *Journal of Aircraft*, Vol. 15, 1978, pp. 407–412.

[28]Sobol, I. M., "On the Systematic Search in a Hypercube," *SIAM Journal of Numerical Analysis*, Vol. 16, 1979, pp. 790–793.

[29]Statnikov, R. B., and Matusov, J. B., *Multicriteria Optimization and Engineering*, Chapman and Hall, New York, 1995, pp. 192–223.

[30]Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D., "Bayesian Prediction of Deterministic Functions, with Applications to the Design and Analysis of Computer Experiments," *Journal of the American Statistical Association*, Vol. 86, 1991, pp. 953–963.

[31]Kelley, C. T., *Iterative Methods for Optimization*, Vol. 18, Frontiers in Applied Mathematics, Society of Industrial and Applied Mathematics, Philadelphia, 1999.

E. Livne
Associate Editor