Optimization and Engineering, 5, 235–262, 2004

Optimal Aeroacoustic Shape Design Using the Surrogate Management Framework

ALISON L. MARSDEN
Mechanical Engineering Department, Stanford University

MENG WANG
Center for Turbulence Research, Stanford University

JOHN E. DENNIS, JR.
Department of Computational and Applied Mathematics, Rice University

PARVIZ MOIN
Mechanical Engineering Department, Stanford University

Received November 15, 2003; Revised February 10, 2004

Abstract. Shape optimization is applied to time-dependent trailing-edge flow in order to minimize aerodynamic noise. Optimization is performed using the surrogate management framework (SMF), a non-gradient based pattern search method chosen for its efficiency and rigorous convergence properties. Using SMF, design space exploration is performed not with the expensive actual function but with an inexpensive surrogate function. The use of a polling step in the SMF guarantees that the algorithm generates a convergent subsequence of mesh points in the parameter space. Each term of this subsequence is a weak local minimizer of the cost function on the mesh in a sense to be made precise later. We will discuss necessary optimality conditions for the design problem that are satisfied by the limit of this subsequence. Results are presented for an unsteady laminar flow past an acoustically compact airfoil. Constraints on lift and drag are handled within SMF by applying the filter pattern search method of Audet and Dennis, within which a penalty function is used to form and optimize a surrogate function. Optimal shapes that minimize noise have been identified for the trailing-edge problem in constrained and unconstrained cases. Results show a significant reduction (as much as 80%) in acoustic power with reasonable computational cost using several shape parameters. Physical mechanisms for noise reduction are discussed.

Keywords: surrogate optimization, aeroacoustics, pattern search methods, optimal shape design

1. Introduction

Reduction of noise generated by turbulent flow past an airfoil trailing-edge is a challenge in many engineering applications including airframe noise, submarine detection, wind turbine design, and rotorcraft applications. Aeroacoustics problems related to such applications necessitate the use of modern computational techniques such as large-eddy simulation (LES) in order to capture a wide range of turbulence scales which are the source of broadband noise. Much previous work has focused on development of accurate computational methods for the prediction of trailing edge noise. For instance, aeroacoustic calculations of the flow over a model airfoil trailing edge using LES and aeroacoustic theory have been presented in Wang and Moin (2000) and were shown to agree favorably with experiments. To make the simulations more cost-effective, Wang and Moin (2002) successfully employed wall models in the trailing-edge flow LES, resulting in a factor of ten reduction in computational cost with minimal degradation of the flow solutions. With the recent progress in simulation capabilities, the focus can now move from noise prediction to noise control. The goal of the present work is to apply shape optimization to the trailing edge flow previously studied, in order to control aerodynamic noise. In the work presented here, the surrogate management framework, developed in Booker et al. (1999) and Serafini (1998), is used to optimize a trailing-edge shape for noise reduction. An unsteady laminar flow with vortex shedding is used for validation of the optimization method, using several shape design parameters and constraints on lift and drag. The laminar problem considered here is meant to act as a model problem for validation of the optimization method, with the intention of extending the work to turbulent flow in the future.

1.1. Choice of optimization method

One general distinction among optimization techniques is between gradient-based methods and non-gradient-based methods. The choice of method for a particular problem depends on factors such as the cost of evaluating the function, the availability of gradient information, the level of noise in the function, and the complexity of implementation. Gradient information is generally obtained using adjoint solutions or finite difference methods. Non-gradient based methods include pattern search methods, approximation models, response surfaces and evolutionary algorithms. There are several considerations in choosing a tractable optimization method for the trailing-edge problem. The primary concern is the computational expense of the function evaluations, especially when each function evaluation requires a large-eddy simulation of turbulent flow. Other considerations are availability of gradient information and robustness of the optimization method.

Calculation of cost function gradients with respect to the control parameters is the main challenge for application of gradient-based optimization methods to the trailing-edge problem. Adjoint solvers have been applied widely in aeronautics problems to obtain gradient information for use with standard gradient descent methods. Unlike finite-differences, the cost of obtaining gradients using an adjoint solver does not grow with the number of optimization parameters. The efficiency of this technique has been successfully demonstrated by Jameson et al. (1998) and Pironneau (1984). To address the difficulties in implementation, there has been considerable work in the development of automatic differentiation tools (see Bischof et al., 1992, for example). However, adjoint methods still present difficulties for time-accurate calculations, particularly with regard to flow field data storage since the adjoint equations must be integrated backwards in time. Additionally, adjoint solvers are not portable from one flow solver to another, and the addition of constraints can present significant added expense. These issues have previously been a roadblock for shape optimization in time-dependent flow problems. In this work, we address these difficulties through the use of derivative-free methods.

Approximation modeling is a general term for methods which use surrogate functions to approximate the cost function. In general, one constructs a surrogate function to interpolate a set of known data points. In the case that the surrogate interpolant is a polynomial, the surrogate is called a "response surface" (Myers, 1971). Other commonly used surrogate functions are splines and Kriging functions. Several surrogate-based optimization methods have been developed for engineering problems which require the use of expensive numerical codes. Gradient information for these problems is often difficult or impossible to obtain. In these methods, optimization may be performed not on the expensive actual function, but on the surrogate, which is cheap to evaluate. For example, surrogate functions have been incorporated into a trust region method by Alexandrov et al. (1998) and Chung et al. (2002), and have also been used by Ong et al. (2003) in evolutionary algorithms to reduce the cost of optimization. An overview of the use of surrogate methods in engineering is given in Guinta (2002).

Another attractive class of methods used in derivative-free optimization are pattern search methods. This mesh-based class of methods offers a flexible framework for optimization in problems with little or no available gradient information, and also provides robust convergence properties. The convergence of pattern search methods has been studied extensively by Audet and Dennis (2000, 2003), Audet (2002), and Torczon (1997). Convergence results for problems with bound, linear and nonlinear constraints have been derived by Lewis and Torczon (1999, 2000, 2002). The surrogate management framework (SMF) (Booker et al., 1999; Serafini, 1998) was developed to increase the efficiency of pattern search methods for expensive problems by incorporating the use of surrogate functions. This method falls into both the categories of approximation modeling methods and pattern search methods. Use of the SMF method has been demonstrated, among others, in Booker et al. (1999) and Serafini (1998), where the method was successfully applied to a helicopter rotor blade design problem with 31 design variables. Mixed variable problems with bound and linear constraints have been studied by Audet and Dennis (2000) and Abramson (2002), respectively.

The SMF method provides a robust and efficient alternative to traditional gradient methods. In this work, the SMF method is applied for trailing-edge optimization in a time-dependent flow problem. Several interesting optimal shapes have been identified, all of which result in significant reduction of vortex-shedding noise. In particular, the development of a trailing-edge bump in the constrained case is an unexpected result which illustrates the trade-off between noise reduction and loss of lift. The remainder of the paper is outlined as follows. Formulation and cost function definition are given in Section 2. An introduction to surrogate methods is given in Section 3 and the concept is illustrated using a one-parameter example. In Section 4, the SMF algorithm is outlined in full, including a discussion of Kriging surrogate functions. Results are presented using the SMF method for the trailing-edge problem with two parameters in Section 5. Constraints are then applied using the filter method of Audet and Dennis (2000) in conjunction with a penalty function. Implementation of this method is discussed in Section 6. In Section 7, constrained optimization results are presented along with unconstrained ones using five shape paraemters, and physical mechanisms for noise reduction are discussed. We conclude with discussion and outlook in Section 8.

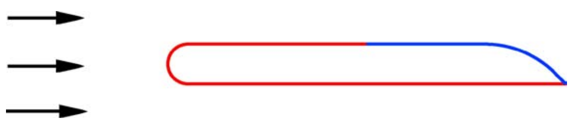

Figure 1. Blake airfoil used in unsteady laminar flow problem. The right half section of the upper surface is allowed to deform.

2. Problem formulation

The general optimization problem may be formulated with bound constraints as follows,

$$\begin{aligned} &\text{minimize} \quad J(x) \\ &\text{subject to} \quad x \in \Omega. \end{aligned} \tag{1}$$

In the above problem statement, $J : \mathbb{R}^n \to \mathbb{R}$ is the cost function, and x is the vector of design parameters. The parameter space is defined by $\Omega = \{x \in \mathbb{R}^n \mid l \leq x \leq u\}$, where $l \in \mathbb{R}^n$ is a vector of lower bounds on x and $u \in \mathbb{R}^n$ is a vector of upper bounds on x. In the present problem, the function $J(x)$ depends on the solution of the Navier-Stokes equations, which allows us to compute the acoustic source.

In this work, the surrogate management framework is implemented and validated for optimization of a time-dependent flow problem. The airfoil geometry is shown in Figure 1 and is a shortened version of the airfoil used in experiments of Blake (1975). The airfoil chord is 10 times its thickness, and the right half of the upper surface is allowed to deform. The flow is from left to right and results presented in this work are at a chord Reynolds number of $Re = 10,000$.

2.1. *Cost function definition*

Before discussion of the optimization method, we define the cost function corresponding to noise generation. For unsteady laminar flow past an airfoil at low Mach number, the acoustic wavelength associated with the vortex shedding is typically long relative to the airfoil chord. Noise generation from an acoustically compact surface can be expressed using Curle's extension to the Lighthill theory (Curle, 1955). Based on this, a cost function directly proportional to the total radiated acoustic power can be derived (Marsden et al., 2002). It is given by the following expression

$$J = \overline{\left(\frac{\partial}{\partial t} \int_S n_j p_{1j}(\mathbf{y}, t) d^2\mathbf{y} \right)^2} + \overline{\left(\frac{\partial}{\partial t} \int_S n_j p_{2j}(\mathbf{y}, t) d^2\mathbf{y} \right)^2}, \tag{2}$$

where $p_{ij} = p\delta_{ij} - \tau_{ij}$ is the compressive stress tensor, composed of pressure and viscous stress, n_j is the direction cosine of the outward normal to the airfoil surface S, and $\mathbf{y}$ is the source field position vector. The overbar denotes time-averaging, and repeated indices follow the usual summation convention. All variables have been made dimensionless, with airfoil chord C as the length scale, free stream velocity U_∞ as velocity scale, and C/U_∞ as the time scale. The pressure is normalized by ρU_∞^2, where ρ is the free-stream density. The radiation is of dipole type, caused by the the fluctuating lift and drag forces. More details on the derivation of the cost function are documented in Marsden et al. (2002). Further discussion of the computation of airfoil self-noise due to vortex shedding may be found in Wang et al. (1996).

The cost function J depends on control parameters corresponding to the airfoil surface deformation. Each parameter corresponds to a deformation point on the airfoil surface, and its value must be within prescribed allowable bounds. The value of each parameter is defined as the displacement of the control point relative to the original airfoil shape, in the direction normal to the surface. A positive parameter value corresponds to displacement in the outward normal direction, and a negative value corresponds to the inward normal direction. A spline connects all the deformation points to the trailing edge point and the left (un-deformed) region (see Figure 1) to give a continuous airfoil surface. Both ends of the spline are fixed. While the surface must be continuous and smooth on the left side, the trailing edge angle is free to change.

2.2. *Cost function evaluation*

For a given set of parameter values, there is a unique corresponding airfoil shape. To calculate the cost function value for a given shape, a mesh is generated, and the Navier-Stokes equations are integrated for sufficiently long time to wash out the initial transients arising from the shape change. A finite difference code discussed in Wang and Moin (2000) is used to solve the time-dependent incompressible two-dimensional Navier-Stokes equations in generalized curvilinear coordinates. The mesh is a C-type mesh consisting of approximately 131,000 cells. Second-order central difference is used for spatial discretization on the staggered mesh. Time advancement is done with a fractional-step type method, with the Crank-Nicolson method for viscous terms, and third order Runge-Kutta for convective terms. A pressure Poisson equation is solved using a multigrid method to enforce continuity. The no-slip velocity boundary condition is enforced on the airfoil surface, and a fixed uniform velocity is enforced on the C-shaped inlet boundary. At the downstream boundary, a convective outflow condition is applied to allow the vortical disturbances to smoothly leave the domain.

Because the flow has unsteady vortex shedding, the cost function is oscillatory. In the optimization procedure, the mean cost function J (cf. (2)) is used, which is obtained by averaging in time until convergence. An example of the oscillatory cost function, and time averaged value is shown in Figure 2. The case shown corresponds to the original airfoil shape. With each shape modification, the flow field is allowed to evolve for sufficiently long time to establish a new quasi-steady state before the time averaging is taken.

Although the flow under consideration is laminar, there are significant computational costs associated with evaluating the cost function. In order to sufficiently converge the cost function value for a given airfoil shape, roughly 24 hours of computer time are required using 4 processors of a parallel SGI O3K machine. These costs will increase substantially when optimizing the trailing-edge shape in turbulent flow in future work.

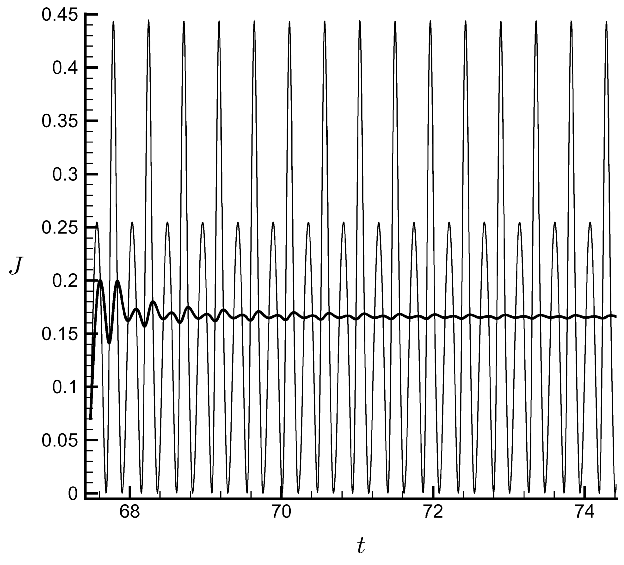

Figure 2. Instantaneous (thin line) and time-averaged (thick line) cost function vs. time. Oscillatory cost function is time averaged until the mean converges. The case shown is for the original airfoil shape.

3. Introduction to surrogate optimization

Before outlining the SMF algorithm in detail, we will first look at a simpler method, which we term the 'strawman' method, to gives a flavor of optimization using surrogates. Use of approximation modeling methods is also illustrated with example functions in Torczon and Trosset (1998). The strawman method is simply implemented as follows.

Let us assume we wish to find a minimum of the one-dimensional function $y = f(x)$ within an allowable domain $x_{\min} \leq x \leq x_{\max}$. First, we begin with a set of initial data points $x = [x_1, x_2, \ldots, x_n]$ where the function values are known. A surrogate function is constructed to fit through the known data points and approximate the actual function. We express the surrogate function as $\hat{y} = \hat{f}(x)$. Because the surrogate function is inexpensive to evaluate, we can easily search for the minimum of the surrogate (within the allowable range of x) using standard optimization methods. In each iteration of the stawman method, the surrogate is used to predict the minimizer of the actual function. Then the actual function value is computed at the predicted minimizer and the surrogate is updated to incorporate the new data. This process continues iteratively until convergence. Possible criteria for convergence are when sufficient cost function reduction has been acheived, or when the incumbent (the best point found so far) remains unchanged from one iteration to the next. In summary, the steps in the strawman algorithm (for one or more parameters) are as follows.

1. Fit a surrogate function through the set of known data points
2. Estimate the function minimizer using the surrogate function
3. Evaluate the true function value at the estimated minimum

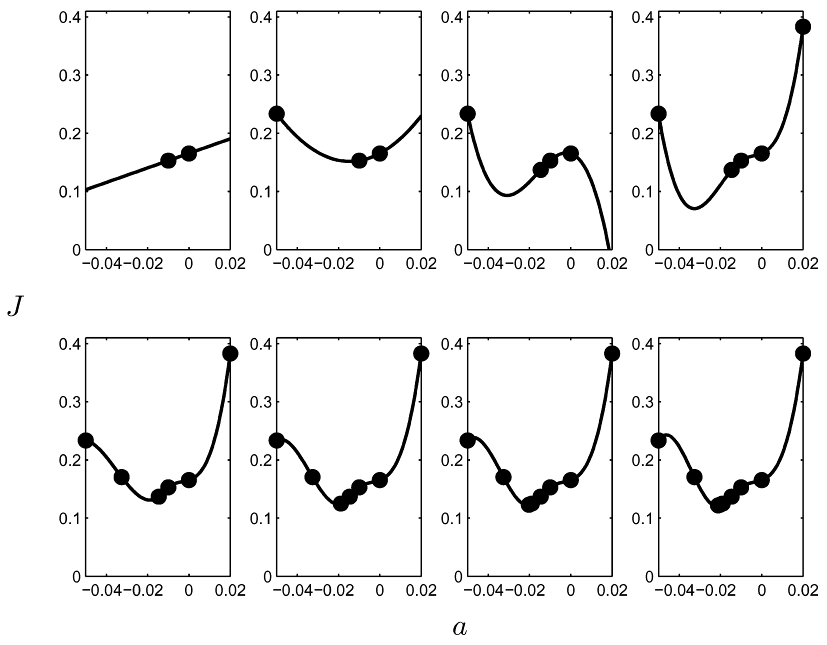

Figure 3. One parameter case with cubic spline as surrogate function. Each plot shows mean cost function J vs. shape parameter a where $a = 0$ corresponds to the original shape. Solid line is surrogate function fit, dots are known function values.

4. Check for convergence
5. Update surrogate using new data points
6. Iterate until convergence

To demonstrate the use of the strawman algorithm with surrogates, we present results for optimization of the trailing-edge shown in Figure 1 using a single parameter a. The value of a determines the normal displacement of a control point on the surface, with a negative value corresponding to inward displacement and a positive value corresponding to outward displacement, as described in Section 2.1. The control point is centered between the trailing-edge and the left side of the deformation region and the bounds on its displacement are $-0.05 \leq a \leq 0.02$. For reference, the initial maximum thickness of the original airfoil is 0.1, and the chord length is unity. Figure 3 shows the evolution of the surrogate spline function as data points are added to the surrogate. The first plot, on the upper left shows a linear fit to the two initial data points. As new data points are added, the surrogate function evolves until it converges in the final plot on the lower right. The maximum cost function reduction obtained is 27% using 12 function evaluations. The corresponding optimal airfoil shape is shown in Figure 4 together with the initial shape. We observe that the shape deformation is in the inward normal direction, and we point out that any additional inward deformation results in flow separation. Results using third and fourth order polynomial surrogate functions are given in Table 1, and these cases both produced a smaller reduction in

Table 1. Summary of results for several surrogate function choices with a single shape parameter. Cost function reduction and number of function evaluations needed for convergence are compared for all cases.

Surrogate	Opt. method	Parameters	% J reduction	Evaluations
3rd order polynomial	Strawman	1	19	4
4th order polynomial	Strawman	1	26	8
Cubic spline	Strawman	1	27	12

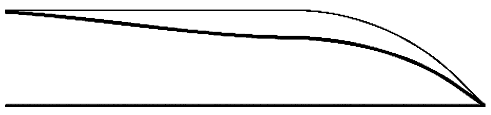

Figure 4. Initial (thin line) and final (thick line) airfoil shapes using one parameter with spline as surrogate function.

cost function and qualitatively similar airfoil shapes. Because the surrogate spline fits exactly through the data points, it captures more detail in the function than either polynomial case.

Using only one parameter, we have demonstrated that a simple surrogate based optimization method produces a significant cost function reduction with a modest number of function evaluations. While this method is extremely easy to implement, and may produce a significant cost function reduction, it has several disadvantages. First, the 'strawman' approach is not strictly guaranteed to converge to a local minimum of the function. This can be easily illustrated by considering the following counter example. Suppose that the initial data set consists of three points, where two points are symmetric around the center point. Using a parabolic surrogate function, the initial fit will be a symmetric parabola with its minimum point at the center data point. The 'strawman' approach will immediately converge to the location of the center point, which is a minimizer of the surrogate, but not necessarily a minimizer of the true function. Another disadvantage of the strawman method is that the algorithm may take very small steps towards the minimum. We will see in the next section that this tendency may be avoided by restricting the algorithm to generate points on a mesh.

4. Outline of the surrogate management framework

In this section, we outline the steps used for trailing-edge shape optimization with the surrogate management framework (SMF). The surrogate management framework, introduced in Booker et al. (1999), is a pattern search method which incorporates surrogate functions to make the optimization cost effective. The main idea behind the SMF method is to use a surrogate function as a predictive tool, while retaining the robust convergence properties of pattern search methods. Like pattern search methods, SMF is a mesh based algorithm, so that all points evaluated are restricted to lie on a mesh. The method is applied to trailing-edge shape optimization with two and five parameters in Sections 5 and 7, respectively.

The first step in the optimization is to choose a set of initial data. Latin hypercube sampling (LHS), introduced by McKay et al. (1979), is commonly used to find a well distributed set of initial data in the parameter space, thus ensuring that each input variable has all portions of its range represented in the chosen data set. To choose a sample set of m vectors in the parameter space, each dimension is divided into m subintervals, and a point within each subinterval is selected (this is often done by randomly sampling from a uniform distribution over the subinterval). The sample set is then obtained by randomly grouping these points to form vectors. Consequently, for each dimension, each interval appears exactly once in the set.

Once the initial data set, $\{x_1, \ldots, x_m\}$, has been chosen, the cost function $J(x)$ is evaluated at these points, and an initial surrogate function is constructed. The surrogate interpolates the data using Kriging (to be discussed in Section 4.1), and then it can be used to predict the value of the function at a particular location in the parameter space. As the optimization progresses, the surrogate should be updated to include new data. After constructing an initial surrogate, all points subsequently evaluated by the algorithm are restricted to lie on a mesh in the parameter space. The mesh definition is flexible so long as it is defined by a set of vectors that positively span $\mathbb{R}^n$ (Lewis and Torczon, 1996). A positive spanning set of a matrix is simply the set of positive linear combinations of its column vectors. If none of the vectors in a given set can be formed from a non-negative combination of the others, the set is considered positively independent (Davis, 1954). A positive basis is then defined as a positively independent set whose positive span is $\mathbb{R}^n$. If we let D be a matrix whose columns form a positive spanning set in $\mathbb{R}^n$, then the set of mesh points surrounding a point x are given by

$$M(x, \Delta) = \{x + \Delta Dz : z \in \mathbb{N}^{n_D}\}, \tag{3}$$

where Δ is the mesh size parameter, and n_D is the number of columns in D. The mesh may be refined or coarsened by changing $\Delta > 0$, and the mesh may be rotated from one iteration to the next as long as it satisfies this definition. Additional technical restrictions are discussed in Torczon (1997).

The SMF algorithm consists of two steps, `SEARCH` and `POLL`. The exploratory `SEARCH` step uses the surrogate to aid in the selection of points which are likely to improve the cost function. The `SEARCH` step provides means for local and global exploration of the parameter space, but is not strictly required for convergence. Because the `SEARCH` step is not integral to convergence, it affords the user a great deal of flexibility and may be adapted by a knowledgeable user to a particular engineering problem.

Convergence of the SMF algorithm is guaranteed by the `POLL` step, in which points neighboring the current best point on the mesh are evaluated in a positive spanning set of directions to check whether the current best point is a mesh local optimizer. A set of $n + 1$ `POLL` points are required to generate an positive basis. An example of such a basis is constructed in $\mathbb{R}^n$ as follows. We let V be the matrix whose columns are the basis elements. Then construct $D = [V, -V \cdot e]$, where e is the vector of ones and $-V \cdot e$ is the negative sum of the columns of V. The columns of D form a positive basis for $\mathbb{R}^n$ using $n + 1$ vectors. For example, in three dimensions such a basis could be given by $(1, 0, 0), (0, 1, 0), (0, 0, 1), (-1, -1, -1)$.

Following evaluation of the initial data, the first step in the optimization is a `SEARCH` step. In the `SEARCH` step, optimization is performed on the surrogate in order to predict the location of one or more minimizing points, and the function is evaluated at these points. If an improved cost function value is found, the search is considered successful, the surrogate is updated, and another search step is performed. If the SEARCH fails to find an improved point, then it is considered unsuccessful and a POLL step is performed, in which the set of POLL points are evaluated. It should be noted that as soon as an improved point is found, it is no longer necessary to evaluate the remainder of the set of POLL points, thereby reducing the number of required function evaluations. If the POLL produces an improved point, then a SEARCH step is performed on the current mesh. Otherwise, if no improved points are found, then the current best point is defined to be a *mesh local optimizer* as in Audet and Dennis (2003). This terminology is to acknowledge that it may not be a local minimizer of the objective function on the mesh since the POLL set is typically a subset of all the points in the mesh adjacent to the POLL center. For greater accuracy, the mesh may be refined, at which point the algorithm continues with a SEARCH. Convergence is reached when a local minimizer on the mesh is found, and the mesh has been refined to the desired accuracy. Each time new data points are found in a SEARCH or POLL step, the data is added to the surrogate and it is updated. The steps in the algorithm are summarized below, where the set of points in the initial mesh is M_0, the mesh at iteration k is M_k, and the current best point is x_k.

1. SEARCH
   - (a) Identify a finite set T_k of trial points on the mesh M_k.
   - (b) Evaluate $J(z)$ for all trial points $z \in T_k \subset M_k$.
   - (c) If for any trial point in T_k, $J(z) < J(x_k)$, a lower cost function value has been found, and the *SEARCH* is *successful.* Increment k and go back to (a).
   - (d) Else, if no trial point in T_k improves the cost function, *SEARCH* is *unsuccessful.* Increment k and go to POLL.
2. POLL
   - (a) Choose a set of positive spanning directions, and form the poll set X_k as the set of mesh points adjacent to x_k in these directions.
   - (b) If $J(x_{\text{poll}}) < J(x_k)$ for any point $x_{\text{poll}} \in X_k$, then a lower cost function has been found and the *POLL* is *successful.* Increment k and go to SEARCH.
   - (c) Else, if no point in X_k improves the cost function, *POLL* is *unsuccessful.*
     - i. If convergence criteria are satisfied, a converged solution has been found. STOP.
     - ii. Else if convergence criteria are not met, refine mesh. Increment k and go to SEARCH.

Because the method has distinct SEARCH and POLL steps, convergence theory for the SMF method reduces to convergence of pattern search methods. Convergence of the SMF method is discussed at length by Booker et al. (1999) and by Serafini (1998). Pattern search convergence theory is presented by Audet and Dennis (2003), Torczon (1997) and Lewis and Torczon (1999, 2000, 2002).

4.1. Construction of surrogates using Kriging

One of the important features of SMF is the use of a surrogate to predict the minimum of the cost function. In this section, we derive an expression for a general Kriging approximation, following Lophaven et al. (2002) and Sacks et al. (1989). Kriging, also called DACE (design and analysis of computer experiments), is a statistical method originating from the field of geostatistics, which is based on the use of spatial correlation functions. It is easily extended to multiple dimensions, making it attractive for optimization problems with several parameters. In this work, the MATLAB DACE package described in Lophaven et al. (2002) was incorporated for surrogate function building.

We wish to approximate the function value at an unknown location $x \in \mathbb{R}^n$ based on a set of known data points. We start with m known data points $\{s_i\} \subset \mathbb{R}^n$ and define $y_s \in \mathbb{R}^m$ to be the column vector whose elements are the corresponding function values; *i.e.* $[y_s]_i = \{y(s_i)\}, i = 1, 2, \ldots, m$. We wish to predict the value of the function based on the values of known points, so we consider the linear predictor

$$\hat{y}(x) = c(x)^T y_s, \tag{4}$$

where $c(x) \in \mathbb{R}^m$ (hereafter referred to as c for simplicity) is a vector of weights applied to the known functions values y_s. By determining the vector c, we will find an approximation of the function at any location x, given a set of known data points. We may assume that the deterministic function $\hat{y}(x)$ can be modeled as the realization of a stochastic process $Y(x)$, which is the sum of a regression model having basis functions $f_j : \mathbb{R} \to \mathbb{R}$ and coefficients $\beta_j, j = 1, 2, \ldots, k$, and a random function $Z : \mathbb{R}^n \to \mathbb{R}$, giving $Y(x) = \sum_{j=1}^{k} \beta_j f_j(x) + Z(x)$, or

$$Y(x) = \beta^T f(x) + Z(x), \tag{5}$$

where $\beta = (\beta_1, \beta_2, \ldots, \beta_k)^T$ and $f(x) = (f_1(x), f_2(x), \ldots, f_k(x))^T$. For any point $x \in \mathbb{R}^n$, the random process $Z(x)$ is assumed to have zero mean, variance σ^2, and correlation $R(w, x)$ between x and any other point w.

The best choice of Kriging weights will be determined by minimizing the mean squared error (MSE) of the predictor, which is the error between the predicted value and the actual value at location x; namely,

$$MSE[\hat{y}(x)] = E[c^T Y_s - Y(x)]^2, \tag{6}$$

where $Y_s \in \mathbb{R}^m$ is the vector defined by $[Y_s]_i = Y(s_i), i = 1, 2, \ldots, m$. Letting $F = [f(s_1), \ldots, f(s_m)] \in \mathbb{R}^{k \times m}$ and $Z = [z_1, \ldots, z_m]$ we have

$$c^T Y_s - Y(x) = c^T Z - z + (F^T c - f(x))^T \beta, \tag{7}$$

where z is a realization of $Z(x)$, and $z_1, \ldots, z_m$ are realizations of $Z(s_i), i = 1, 2, \ldots, m$. We impose an unbiasedness constraint (ensuring that the components of c sum to one)

$$F^T c - f(x) = 0, \tag{8}$$

so that (7) becomes

$$c^T Y_s - Y(x) = c^T Z - z. \tag{9}$$

Now, the MSE is

$$MSE[\hat{y}(x)] = E[z^2 + c^T ZZ^T c - 2c^T Zz]. \tag{10}$$

From the covariance of Z, we have $E[z^2] = \sigma^2$, $E[Zz] = \sigma^2 r$ and $E[ZZ^T] = \sigma^2 R$, where $r \in \mathbb{R}^m$ is a vector of correlations between the known points and an untried point x, and $R \in \mathbb{R}^{m \times m}$ is the matrix of correlations between the known points. With this, the MSE becomes

$$MSE[\hat{y}(x)] = \sigma^2(1 + c^T Rc - 2c^T r). \tag{11}$$

We wish to find the weights, c, which minimize the MSE subject to the constraint (8). To do this, we use the method of Lagrange multipliers with the Lagrangian function

$$L(c, \lambda) = \sigma^2(1 + c^T Rc - 2c^T r) - \lambda^T(F^T c - f), \tag{12}$$

where λ is the Lagrange multiplier. Taking the gradient of the Lagrangian and setting it to zero, we obtain

$$\begin{aligned} Rc + F\tilde{\lambda} &= r \\ F^T c &= f, \end{aligned} \tag{13}$$

where $\tilde{\lambda} = -\lambda/2\sigma^2$. Solving this system and substituting into the predictor, we have

$$\hat{y}(x) = f(x)^T \beta^* + r(x)^T \gamma^*, \tag{14}$$

where we have defined

$$\begin{aligned} \beta^* &= (F^T R^{-1} F)^{-1} F^T R^{-1} Y_s \\ \gamma^* &= R^{-1}(Y - F\beta^*). \end{aligned}$$

For a given set of data and choice of regression and correlation functions, β^* and γ^* are fixed and need not be recomputed for each new point x.

To complete our description of Kriging surrogates we must choose a regression model and a correlation function. The most common choice of regression model is simply $f(x) = 1$ so that the Kriging predictor becomes

$$\begin{aligned} \hat{y}(x) &= \beta^* + r(x)^T R^{-1}(Y_s - 1 \cdot \beta^*) \\ \beta^* &= \frac{\sum_j Y(s_j) \sum_i R^{-1}_{i,j}}{\sum_{i,j} R^{-1}_{i,j}} \end{aligned} \tag{15}$$

The correlation function is chosen as the product of stationary one dimensional correlations, making the surrogate easily extendable to multiple dimensions. A common choice of correlation function is to express the correlation between two points x and w in terms of a Gaussian process

$$\mathcal{R}(\theta, w, x) = \prod_{j=1}^{n} \exp(-\theta_j (w_j - x_j)^2). \tag{16}$$

The Kriging surrogate in (15) is completed with the matrix of correlations between the values of z at any two known design sites, which is defined by

$$R_{ij} = \mathcal{R}(\theta, s_i, s_j), \quad i, j = 1, 2, \ldots, m,$$

and the vector of correlations between the value of z at a known design site and any point x

$$r(x) = [\mathcal{R}(\theta, s_1, x), \ldots, \mathcal{R}(\theta, s_m, x)].$$

The optimal value θ^* of θ is found using maximum likelihood estimation in each dimension, so that θ^* solves

$$\min_{\theta} \left\{ \psi(\theta) \equiv |R|^{\frac{1}{m}} \hat{\sigma}^2 \right\}, \tag{17}$$

where $|R|$ is the determinant of R, and

$$\hat{\sigma}^2 = \frac{1}{m}(y_s - F\beta^*)^T R^{-1}(y_s - F\beta^*).$$

5. Unconstrained two parameter results

In Section 3, significant cost function reduction was demonstrated using the ‘strawman’ approach with one shape parameter. In this section, the full SMF method is validated using two shape parameters in several cases, all of which give significant improvements over the one parameter case. The placement of the airfoil surface control points for the two parameters, a and b, is shown in Figure 5. For each set of parameters, the airfoil surface is interpolated using a Hermite spline. The bounds on the parameters are chosen to correspond to a straight line connecting the left edge of the deformation region and the trailing edge.

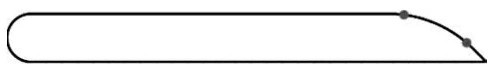

Figure 5. Placement of control points for optimization with two parameters.

Table 2. Two parameter 'strawman' method cases.

Surrogate	Opt. method	Init. data	Parameters	% J reduction	Evaluations	Iterations
Spline	Strawman	6 rand	2	29	17	11
Spline	Strawman	5 star	2	52	24	19
Kriging	Strawman	7 LHS	2	54	18	11

To give a basis for comparison, results for three cases using the 'strawman' method (no POLL step) with two shape parameters are presented in Table 2. The first case uses a biharmonic spline surrogate function with a random set of initial data, the second case uses a biharmonic spline with a star shaped initial data pattern, and the third case uses a Kriging surrogate with initial data from Latin hypercube sampling. While it is impossible to conclude anything from a single statistical sample, comparison of these results suggests that choosing a well-distributed data set, such as LHS, can make a difference in the results. The results also suggest that the influence of the surrogate function can be substantial. The third case in Table 2, using Kriging and LHS, produced the most significant cost function reduction of 54% after 18 function evaluations. The optimal shape obtained in this case is shown on the left of Figure 6. The corresponding normalized cost function reduction is given on the right of Figure 6, and the initial and final Kriging surrogates are shown in Figure 7. To make the 'strawman' case consistent with cases using the SMF method, all points evaluated were restricted to lie on a mesh of the same size. Convergence for the 'strawman' was reached when surrogate minimum point was the same as the previous iteration.

For purpose of comparison, all cases using the full SMF method use the same set of LHS initial data as the 'strawman' case shown on the left side of Figure 7. All tables in this section list the total number of function evaluations as well as the number of iterations, where one iteration is a complete SEARCH or POLL step. In all cases, the number of function evaluations includes the number of initial data points. The number of iterations includes all

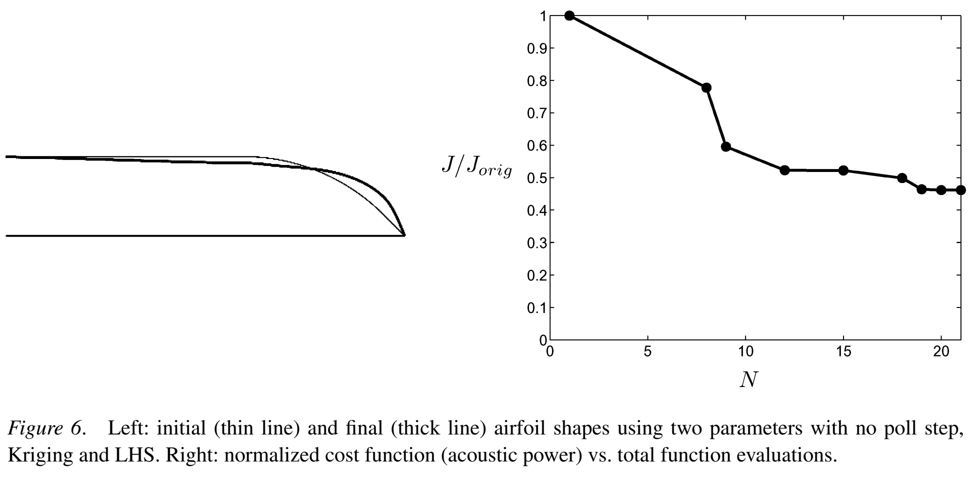

Figure 6. Left: initial (thin line) and final (thick line) airfoil shapes using two parameters with no poll step, Kriging and LHS. Right: normalized cost function (acoustic power) vs. total function evaluations.

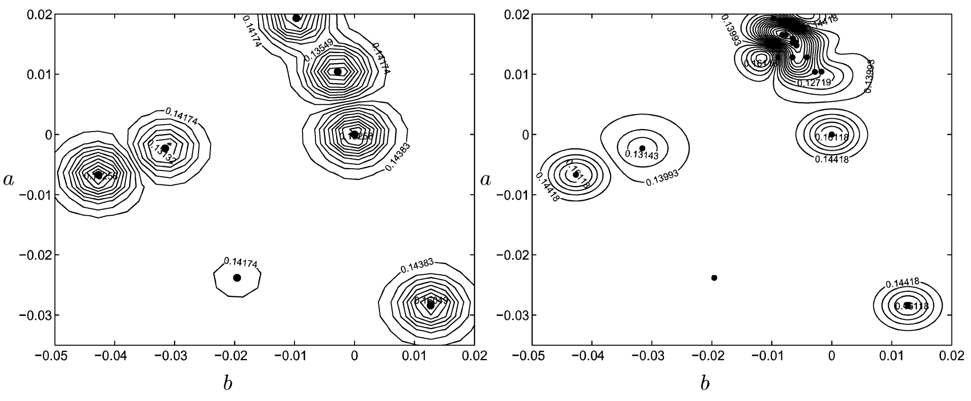

Figure 7. Initial and final Kriging surrogate functions for “strawman” case. Left plot shows initial data obtained with Latin hypercube sampling; right side shows final surrogate fit.

SEARCH and POLL steps, but not evaluation of the initial data set. Each table gives converged results on three subsequently finer meshes.

The full SMF method is implemented with the POLL step for two parameters, and results are given in Table 3. In this case, one point, the surrogate minimizer rounded to the mesh, is evaluated in each SEARCH step. Even on the original mesh, a 69% cost function reduction has been obtained compared with 54% in the ‘strawman’ method. One mesh refinement gives a slight improvement (72% total reduction), while the second refinement has no effect on the cost function. Comparing with the ‘strawman’ case, the number of required iterations has increased from 11 to 15 for convergence with two mesh refinements using equivalent meshes. Although the POLL step adds some computational expense, we have demonstrated that it can also result in a significantly lower cost function value. As discussed in Section 4, the POLL step ensures a convergent subsequence of *mesh local optimizers* in the parameter space.

Tables 4 and 5 explore the effect of using multiple points in the SEARCH step of the algorithm. In each case, one SEARCH point is found by a direct search for the minimum of the surrogate function on the mesh. It is well known that the ‘pile-up’ of points in Kriging functions can lead to degradation of the surrogate accuracy (Audet et al., 2000). Because of this, additional search points may be chosen by searching the mesh for areas in which there is not much data. These ‘space-filling’ points are added in an effort to keep the data set well distributed and prevent degradation of surrogate accuracy. The addition of ‘space-filling’

Table 3. Two parameter SMF—one search point from surrogate.

Surrogate	Refinement	Init. data	Parameters	% J reduction	Evaluations	Iterations
Kriging	0	7 LHS	2	69	21	9
Kriging	1	7 LHS	2	72	29	13
Kriging	2	7 LHS	2	72	33	15

Table 4. Two parameter SMF—two search points (one from surrogate, one space filling).

Surrogate	Refinement	Init. data	Parameters	% J reduction	Evaluations	Iterations
Kriging	0	7 LHS	2	69	27	9
Kriging	1	7 LHS	2	72	37	13
Kriging	2	7 LHS	2	72	42	15

Table 5. Two parameter SMF—three search points (one from surrogate, two space filling).

Surrogate	Refinement	Init. data	Parameters	% J reduction	Evaluations	Iterations
Kriging	0	7 LHS	2	71	25	6
Kriging	1	7 LHS	2	77	33	9
Kriging	2	7 LHS	2	77	46	13

points does not necessarily increase the overall wall-clock time since the SEARCH points may be evaluated in parallel. Results using two points in each SEARCH step are given in Table 4. In this case the SEARCH points are the surrogate minimizer and one additional 'space-filling' point. The cost function reduction and the optimal shape are identical to the case with only one search point. Using one and two search points, 13 iterations were required to acheive the maximum cost function reduction, which is a slight increase over the 11 required in the 'strawman' case.

Using three SEARCH points (one surrogate minimizer and two 'space-filling' points) a larger cost function reduction of 77% was achieved. Results for this case are given in Table 5 for the original mesh and two mesh refinements. Figure 8 shows the optimized airfoil shape (left) and the cost function reduction (right) for this case. Comparing with the results in Tables 3 and 4, the case with three search points used fewer iterations. This savings is

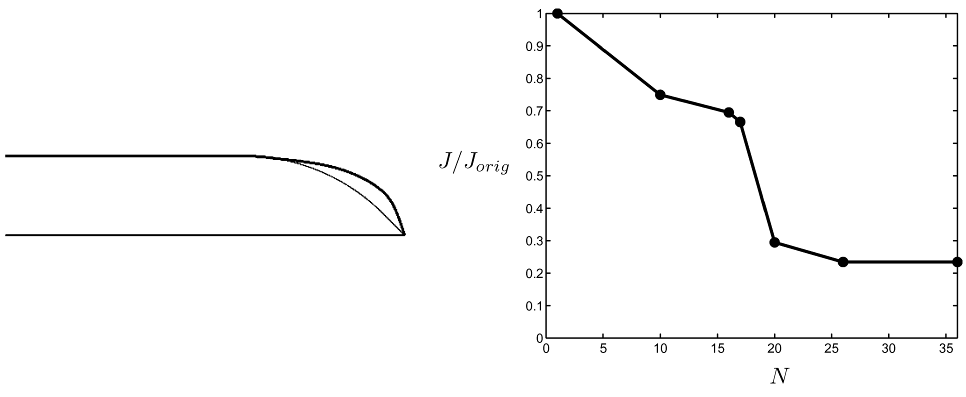

Figure 8. Left: initial (thin line) and final (thick line) airfoil shapes using two parameters with SMF method. Right: normalized cost function (acoustic power) vs. total function evaluations. Each SEARCH step used three function evaluations.

explained by a higher surrogate quality in the final iterations, resulting in fewer polling steps. A comparison of surrogate quality is made by evaluating the mean squared error predicted by the surrogate.

The two-parameter results show a dramatic improvement in cost function reduction compared with one-parameter results. We have also shown that efforts to reduce surrogate degredation can pay off, resulting in a lower cost function solution. In addition, the two parameter cases, shown in Figures 6 and 8, resulted in airfoil shapes with a blunt trailing-edge, which was at first sight counter-intuitive. The magnitude of acoustic power has decreased significantly for the optimized shapes compared to the original. However, with the 77% acoustic power reduction in the best two parameter case, the optimized airfoil has 20% lower lift than the original, which is often not acceptable in engineering practice. This emphasizes the need for addition of constraints on lift and drag, which is addressed in Section 6.

To further quantify the contribution of the surrogate in the best two-parameter case, we note that 88% of the total cost function reduction for this case was the result of surrogate based `SEARCH` points. The `POLL` steps accounted for the remaining 12% of the cost function reduction. There were a total of 8 `SEARCH` steps, three of which were successful, and a total of 5 `POLL` steps, two of which were successful. Based on this data, we can make a rough extrapolation to a case without surrogates, using `POLL` steps alone. Given that the 8 `SEARCH` steps averaged 11% of the total reduction per iteration, and the 5 `POLL` steps averaged 2.4% reduction per iteration, it is estimated that an algorithm based on `POLL` steps alone would require at least 42 iterations to achieve an equivalent cost function reduction. Although this is a simplistic estimate, it suggests that a direct search algorithm without surrogates would result in a significant computational cost increase over the 13 iterations used by the surrogate-based SMF method. We also note that all of the successful searches were due to surrogate minimum points, and not due to space filling points, suggesting that the surrogate-based searches are much more effective than purely random searches. It is likely that the surrogate does not always accurately capture the global behavior of the true function. However, the success of the surrogate-based searches suggests that the it is crucial to efficiency of the method for this problem. It is important to emphasize that the surrogate accuracy is in no way related to the convergence theory of the algorithm, but that the surrogate simply acts as a guide for the selection of search points.

6. Constrained optimization using filters

To make the trailing-edge design problem more realistic, constraints are added to keep lift and drag at desirable levels. The loss of lift observed in the two parameter cases in Section 5 underscores this need. Constraints are enforced by using a filter, as added to pattern search methods by Audet and Dennis (2000). The use of filters for constrained optimization was originally introduced by Fletcher and Leyffer (2002). We consider the general constrained optimization problem

$$\begin{aligned} &\text{minimize} && J(x), \\ &\text{subject to} && x \in \Omega, C(x) \leq 0. \end{aligned} \tag{18}$$

In the above problem statement, $J : \mathbb{R}^n \to \mathbb{R}$ is the cost function, and x is the vector of parameters. The constraints are given by m functions $c_i : \mathbb{R}^n \to \mathbb{R}, i = 1, 2, \ldots, m$ such that $C(x) = (c_1(x), \ldots, c_m(x))^T$. The bounds on the parameter space are defined by a polyhedron in $\mathbb{R}^n$ denoted by Ω.

We begin by defining a non-negative constraint violation function $H : \mathbb{R}^n \to \mathbb{R}^+$, which is a weighted norm of the constraint violations. The value of H indicates how closely the problem constraints are being met. With multiple constraints, H may be the sum of several constraint functions, with weights chosen according to relative importance. The goal of the optimization problem is to find solutions which have a small cost function value, together with a small (or zero) value of H.

The feasible region is defined as the set of points that exactly satisfy $H(x) = 0$. Thus, a point x is infeasible if $H(x) > 0$. An infeasible point x' is considered *filtered*, or *dominated*, if there is an infeasible point x belonging to the filter for which $H(x) \leq H(x')$ and $J(x) \leq J(x')$. A filter, $\mathcal{F}$, is defined here to be the finite set of non-dominated infeasible points found so far. An example of a typical filter is shown in Figure 9, where the cost function value (J) is plotted versus the constraint violation (H). The points in the filter are connected with vertical and horizontal lines to form a dividing line between filtered and unfiltered regions. The best feasible point, marked with a square, is the point with the lowest cost function value, which satisfies the constraints (i.e., where $H = 0$). The least infeasible point, marked with a triangle, is the filter point with the lowest non-zero constraint violation function value. Other points in the filter are marked with circles.

The steps in the filter optimization algorithm fit within the framework of the SMF method, and the basic structure is the same as presented in Section 4. The differences in implementation between unconstrained SMF and the filter method lie in the criteria which make SEARCH and POLL steps successful or unsuccessful. In the filter method, a SEARCH or POLL step is formally considered successful if it improves the filter, which

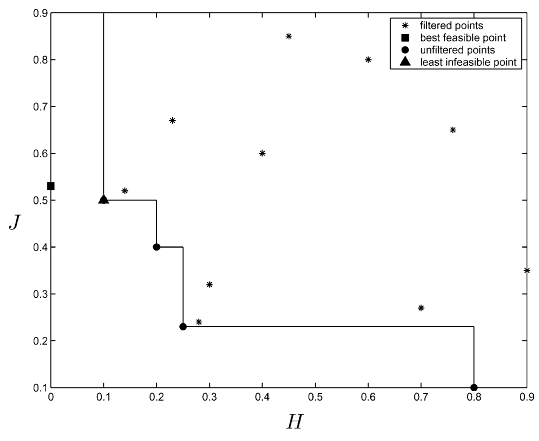

Figure 9. Example of filter for constrained optimization problem shown on plot of J vs. H. The best feasible point is the square, the least infeasible point is the triangle, the filter points are the circles, and dominated points are stars.

means that a new non-dominated point was identified. For example, referring to Figure 9, a point with values $J = 0.2$, $H = 0.5$ would improve the filter. Like the SMF method, the algorithm consists of `SEARCH` and `POLL` steps as follows. Convergence theory of this method is also based on pattern search theory, and is discussed at length in Audet and Dennis (2000). The set of points in the initial mesh is M_0, the mesh at iteration k is M_k, the current least infeasible point is LF_k and the current best feasible point is BF_k.

1. `SEARCH`
   (a) Identify finite set T_k of trial points on the mesh M_k.
   (b) Evaluate $J(z)$ for all trial points $z \in T_k \subset M_k$.
   (c) If any point in T_k, is an unfiltered point, the *`SEARCH`* is *successful.* Increment k and go back to (a).
   (d) Else, if no trial point in T_k is an unfiltered point, the *`SEARCH`* is *unsuccessful.* Increment k and go to `POLL`.
2. `POLL`
   (a) Choose a set of positive spanning directions, and form the poll set X_k as the set of mesh points adjacent to either LF_k or BF_k in these directions.
   (b) If any point in X_k is an unfiltered point, the *`POLL`* is *successful*. Increment k and go to `SEARCH`.
   (c) Else, if no point in X_k is an unfiltered point, the *`POLL`* is *unsuccessful.*
      i. If convergence criteria are satisfied, an approximate solution has been found. `STOP`.
      ii. Else, if convergence criteria are not met, refine mesh. Increment k and go to `SEARCH`.

Formally, it is allowable to enforce a stricter criterion for the success of the `POLL` step. In the constrained trailing-edge optimization in Section 7, we require improvement of either the best feasible point or the least infeasible point for success of the `POLL` step. The algorithm is then restricted to explore areas of known interest, reducing the number of required `POLL` steps. This approach is used in the constrained trailing-edge optimization because the problem requires particularly expensive function evaluations. For problems in which cost is less critical, it may be beneficial to keep the less restrictive definition to afford more exploration of the design space. As stated in step 2(a), it is allowable to poll around *either* the best feasible point or the least infeasible point or both. In the current problem, `POLL` steps alternated between the best feasible point and the least infeasible point. This can be decided according to the users discretion as long as one performs a thorough exploration of the design space.

In the trailing-edge problem, we have chosen to include the best feasible point in the filter so that any infeasible point with a higher cost function value than the best feasible point will be filtered. This is an allowable choice made solely due to the significant expense of the function evaluations. In general, it may be beneficial to keep the less restrictive filter definition because a relaxation of the constraint can lead to more thorough exploration of the design space. Technical reasons for not including the feasible points in the filter (i.e., tracking them separately) have been presented by Fletcher and Leyffer (2002).

6.1. Incorporation of a penalty function

Many optimization methods rely on the use of a penalty function to enforce constraints. Penalty functions are used to artificially increase the cost function value when constraints have been violated, thus steering the optimizer to areas which satisfy the constraint. They are attractive due to ease of implementation into existing optimization frameworks. Challenges usually involve the choice of arbitrary weighting of the constraint function relative to the cost function. Penalty functions can be easily incorporated into the SMF filter method, and can be extremely useful in aiding the selection of SEARCH points. Here, we present a systematic approach for choosing the penalty constant by making use of the filter framework.

For the trailing-edge problem, the penalty function

$$\hat{J} = J_{\text{orig}} + \alpha H \tag{19}$$

is formed, and a surrogate is constructed to approximate the function $\hat{J}$, so that it is used to predict areas of the function which satisfy the constraint. The minimum of the modified surrogate function can then be evaluated in the SEARCH step.

The parameter α is chosen based on the current set of filter points (including the best feasible point). We wish to choose α so as to bias the surrogate towards points with low values of H, which will improve the filter. Let us first consider a filter with two points, a and b, for which $H(a) < H(b)$ and $J(a) > J(b)$. We wish to choose α so that point a is favored by the surrogate because it has a smaller constraint violation. We therefore require

$$J(a) + \alpha H(a) < J(b) + \alpha H(b)$$

and we see that for this pair of points α must be at least as large as the negative of the slope of the line connecting them. When considering the set of points making up a filter, α should therefore be at least as large as the negative slope of the steepest line connecting any two points in the filter. This choice guarantees domination of the point with the smallest value of H in the filter. Since the filtered points are not of interest in the optimization, they need not be considered in the choice of α. If there are less than two points in the filter set, $\alpha = 0$. As points are evaluated in the optimization, the filter evolves and the value of α is updated in each iteration. Values of J and H for all data points should be saved so that previous data points may be updated in the surrogate as the value of α changes.

7. Results of constrained optimization

In this section, the SMF method is applied for shape optimization using five design parameters. An increase in the number of parameters gives greater flexibility in the airfoil

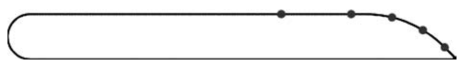

Figure 10. Placement of control points for optimization with five parameters.

geometry and will also demonstrate feasibility and cost of the SMF method for more realistic applications. Increasing the number of parameters also presents challenges for searching on the surrogate, and these are addressed in this section. Constraints are applied to keep lift and drag at desirable levels using the filter methods discussed in Section 6, and results are compared to the equivalent unconstrained case. Figure 10 shows the placement of the five control points on the upper surface of the airfoil.

7.1. Shape parameterization with thickness constraint

In order to enforce a thickness constraint in the case of multiple parameters, strict bounds must be defined for the allowable shape deformation. As in the two parameter case, the airfoil surface is defined by interpolation between the control points using a Hermite spline, and displacement of the airfoil surface must be normal to the surface of the original airfoil. However, in this case, we take care to enforce strict bounds on all points on the airfoil surface, not just at the control points.

The bounds on the surface deformation region are based on pre-defined minimum and maximum thickness functions. Displacement of all points relative to the original airfoil surface is normalized by the maximum allowable distance in the inward and outward normal directions. All parameters a_i have values such that $-1 \leq a_i \leq 1$, where $a_i = 0$ corresponds to the original airfoil shape, $a_i = -1$ corresponds to the maximum inward displacement, and $a_i = 1$ corresponds to the maximum outward displacement. Use of a Hermite cubic spline as the interpolating function guarantees that values of all interpolated points are bounded by the minimum and maximum values of the known data points. This definition guarantees that no point on the surface will be displaced more than the maximum allowable displacement distance.

To define the minimum bound, a straight line is drawn from the left side of the upper surface deformation region to the trailing-edge point. This line is used to define the minimum allowable airfoil thickness at all points on the surface. Because displacement of all points in normalized by the maximum, this line defines a maximum inward normal displacement for all surface points. The maximum airfoil thickness is defined by an equal displacement from the surface in the outward normal direction. This method can be easily generalized for any prescribed function which defines the thickness constraint.

7.2. Selection of SEARCH *points*

Using a surrogate in multiple dimensions requires use of an optimization method to search for the surrogate minimizer. With two shape parameters in Section 5, the surrogate minimizer was found using a direct search on the mesh. To search the surrogate in the five parameter cases, a standard covariance matrix adaptation evolutionary strategy (CMA-ES) is employed. Evolutionary algorithms (EA's) usually provide a fast initial function descent, and they allow for thorough global search in the parameter space. The undesirable aspects of EA's are usually the computational expense, which can be thousands of function evaluations, and the general lack of formal convergence theory. In this case, a surrogate minimum is not required since the purpose of the surrogate is to find promising areas of the function for the `SEARCH` step. Since each surrogate function evaluation is essentially cost-free compared to that of the true function, computational expense of the surrogate is not an issue in this application. The covariance matrix adaptation enables the evolution strategy to efficiently minimize even badly scaled and non-separable problems through adaptation of strategy parameters. Further details of the CMA-ES may be found in Hansen et al. (2003) and Hansen and Ostermeier (1996). Accuracy is increased by running the CMA-ES optimization several times and taking the minimum value.

Based on experience with two parameters, three points are chosen in each `SEARCH` step for the five-parameter cases. They are chosen to meet the following goals:

1. global search
2. local search around current best point
3. surrogate improvement.

The first point is chosen using the surrogate function as a predictor. In the constrained case, the surrogate is modified with a penalty function as in Eq. (19). The minimum of the surrogate is found using the CMA-ES. The nearest mesh point to this minimizing point is evaluated in the `SEARCH`.

The second `SEARCH` point takes advantage of the surrogate to do a local search around the current best point. In order to pre-empt the need for a `POLL` step, the surrogate is used to predict the values of the `POLL` points neighboring the current best point. In the constrained case, the current best point may be either the best feasible point or the least infeasible point. The `POLL` point with the smallest surrogate value is then evaluated. In the event that the `SEARCH` step fails, one of the `POLL` points has already been evaluated. This step guarantees that the cost of any `POLL` step will be N evaluations, rather than $N + 1$.

The third point is for both surrogate improvement and global search. Since a direct search for space-filling points on a five-dimensional mesh is impractical, the mean squared error is used to identify areas for surrogate improvement. The CMA-ES is used to search the Kriging surrogate for the point of maximum mean squared error (MSE), and the nearest mesh point is used in the `SEARCH`. This is done in an attempt to find areas with the fewest data points, as well as mitigate surrogate degradation.

7.3. *Comparison of constrained and unconstrained results*

Unconstrained results using five shape parameters are shown in Table 6. The first line in the table gives results for convergence on the original mesh, and the second line gives the converged solution after one mesh refinement. Based on results from the two parameter cases,

Table 6. Five parameter cases with SMF method, unconstrained.

Params	Refinement	% J reduction	% change lift	% change drag	Evaluations	Iterations
5	0	65	−20	−12	58	13
5	1	77	−17	−12	88	22

additional mesh refinements were not performed. The number of evaluations in the table includes the initial data set of 15 points, found with Latin hypercube sampling. The number of iterations includes all search and poll steps following evaluations of initial data set. The converged airfoil shape for the unconstrained case is shown on the left side of Figure 11, and the cost function reduction for this case is shown on the right. The blunt trailing-edge shape is qualitatively similar to the shapes obtained with the two parameter optimization, confirming robustness of the SMF method. The maximum cost function reduction for this case is 77% using 22 iterations after one mesh refinement, which agrees with the two parameter results. As in the two parameter case, the blunt trailing-edge results in a significant (nearly 20%) loss in lift. The loss of lift is not surprising, since the optimized shape is more symmetric than the original shape. We observe that the blunt trailing-edge reduces the surface area affected by the separation region at the trailing-edge. It is hypothesized that pressure fluctuations are reduced in this region due to the smaller separation area. This in turn results in less vorticity generation, and a smaller vorticity magnitude in the wake.

We can now examine the effect of the surrogate and the three types of `SEARCH` points used in the five parameter unconstrained case. There were 12 successful and 5 unsuccessful `SEARCH` steps, for a total of 17, and three successful and two unsuccessful `POLL` steps, totaling 5. Of the total cost function reduction in this case, 70% was the result of `SEARCH` steps and 30% was the result of `POLL` steps. Although the relative contribution of the surrogate has decreased slightly compared with the two parameter case, it continues to play a crucial role in the optimization. It is also useful to look at the effectiveness of the three

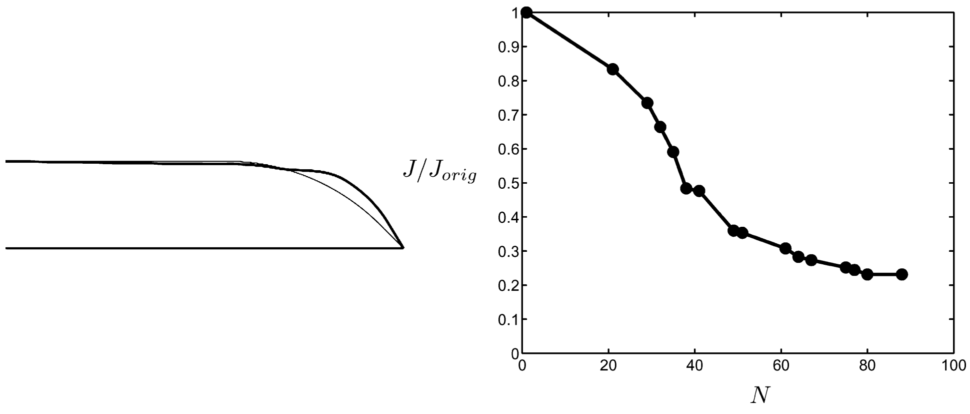

Figure 11. Left: initial (thin line) and final (thick line) airfoil shape with 5 shape parameters, unconstrained case. Right: corresponding normalized cost function reduction vs. number of function evaluations.

Table 7. Five parameter cases with SMF method, penalty constraints on lift and drag.

Params	Refinement	% J reduction	% change lift	% change drag	Evaluations	Iterations
5	0	42	+0.1	−9	74	17
5	1	43	+0.2	−9	92	22

types of SEARCH points used in finding improved points. The surrogate minimum points (type 1) contributed 45% of the cost function reduction resulting from SEARCH steps, and the local surrogate-based SEARCH points (type 2) contributed the remaining 55%. Based on these results, we claim that the surrogate is an effective guide for selecting both global and local SEARCH points. Additionally, the data suggests that use of a surrogate results in a substantial cost savings in the SMF algorithm compared with a method based purely on POLL steps.

Results for the constrained case are presented in Table 7. The surrogate for this case is constructed as the sum of the constraint violations, in order to keep lift and drag at desirable levels. Since this choice of constraint violation function is based on the L_1-norm, it is consistent with the exactness property, which states that there exists an α sufficiently large such that the solution to $\min(J)$ is also a solution to $\min(\hat{J})$ (Nocedal and Wright, 1999). The following constraint violation function is used in the trailing-edge optimization,

$$H = \max\left(0, \frac{L^* - L}{L^*}\right) + \max\left(0, \frac{D - D^*}{D^*}\right), \tag{20}$$

where L^* and D^* are the original airfoil lift and drag. The surrogate is constructed using Eq. (19), so that a penalty is added if the surrogate predicts that *either* the lift decreases or the drag increases. In this way, we allow the lift to increase and/or the drag to decrease.

The optimized airfoil shape for the constrained case is shown on the left side of Figure 12. The constraints are effective in keeping the lift at the target value, and the drag for this case has fortuitously decreased by 9%. The cost function reduction for this case is 43%, requiring 22 iterations for convergence on a once-refined mesh. The bump near the trailing-edge reduces the magnitude of the unsteady vortex shedding by reducing the size of the separation region. However, when compared to the unconstrained case, the bump size has been compromised to maintain lift, and the trailing-edge shape is closer to a cusp. Comparison of the shapes for the constrained and unconstrained cases also illustrates the sensitivity of the flow to very small changes in the shape of the airfoil.

The final filter is shown in Figure 13. The left side shows the entire filter domain, and the right side shows a magnified view of the filter region. The filter shows the tradeoff between cost function reduction and constraint violation. The cluster of points around the filter, and on the $H = 0$ axis verifies that the algorithm is expending most of its effort in the most promising areas of the parameter space. For comparison, the rightmost filter point corresponds to a shape with a 64% cost function reduction and a 13% loss in lift. The other filter points show the range of possible airfoil designs evaluated between this point and the optimal point.

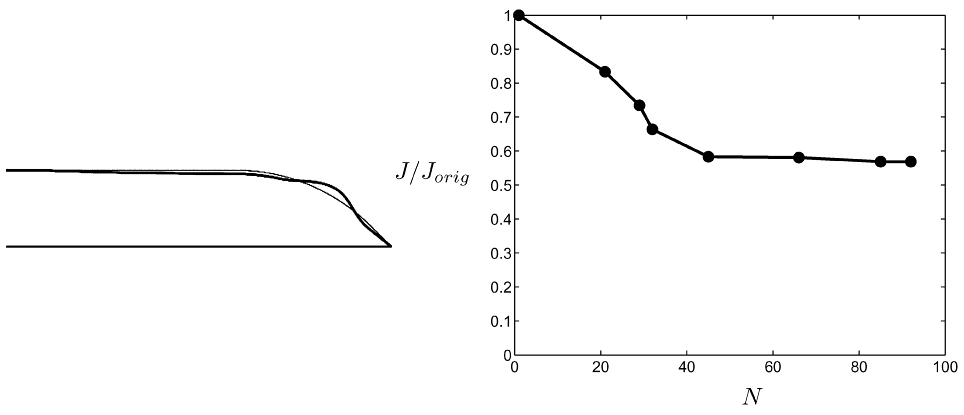

Figure 12. Left: initial (thin line) and final (thick line) airfoil shape with 5 shape parameters, and constraints on lift and drag. Right: corresponding normalized cost function reduction vs. number of function evaluations.

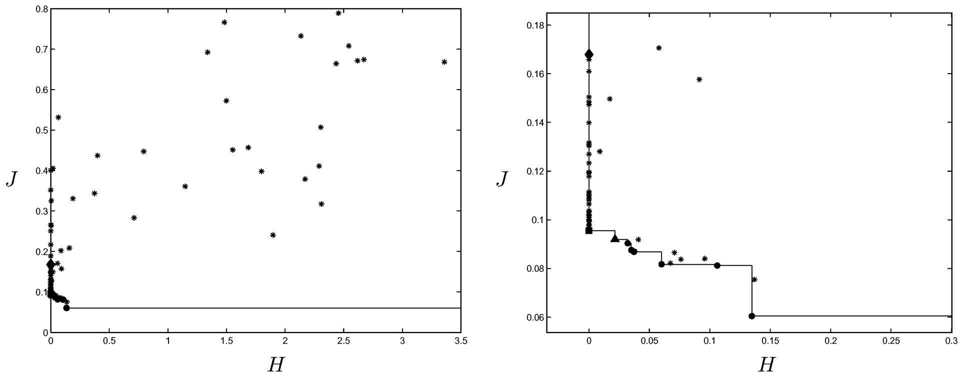

Figure 13. Final filter for constrained 5 parameter optimization problem. Cost function J vs. constraint violation function H. The best feasible point is the square, the least infeasible point is the triangle, the filter points are the circles, and filtered points are stars. The original airfoil cost function is marked with a diamond. Right figure is close-up of filter region in left figure.

In the five parameter constrained case, the surrogate continues to play a key role in the optimization. In this case, there were a total of 8 successful, and 7 unsuccessful SEARCH steps, and a total of 5 successful and 2 unsuccessful POLL steps. The SEARCH steps resulted in 58% of the total reduction of the cost function value of the best feasible point, and POLL steps resulted in 42% of the total reduction. The effect of the surrogate has been somewhat reduced compared with the 5 parameter unconstrained case. However, some reduction in surrogate effectiveness is expected with the addition of constraints due to the modification of the surrogate with the penalty function. Despite this, the surrogate is responsible for the majority of the cost function reduction, and once again is likely to be responsible for a large savings in computational cost.

8. Discussion

Application of the SMF method to trailing-edge optimization has resulted in significant reduction in acoustic power in all cases, as well as several interesting and previously unexpected airfoil shapes. The SMF method is robust and cost effective for several design parameters with and without constraints. The SMF filter method has been applied to enforce constraints on airfoil lift and drag. Surrogate building incorporated the use of a penalty function, with a systematic method for choosing the penalty parameter. We reiterate that the penalty function was only used to aid in the selection of the SEARCH points, and is in no way required for enforcement of constraints with the filter method. Comparison between the constrained and unconstrained cases using five parameters clearly showed a trade-off between noise reduction and loss of lift.

Theoretical analysis of trailing-edge noise for the Blake airfoil geometry in turbulent flow is presented by Howe (1988). Results from this work show that the lift dipole is much more significant in contributing to the noise spectra than the thickness (drag) dipole, which was confirmed by our simulations. Although Howe's work is an analysis of turbulent trailing-edge flow, it is worth noting that his analysis predicted a decrease in trailing-edge noise with an increase in trailing-edge angle. This result agrees qualitatively with the blunt shapes found by the optimization method in the two and five parameter cases, both of which resulted in a dramatic reduction in trailing-edge noise.

The similarity between optimal shapes obtained in the unconstrained two and five parameter cases demonstrates the robustness of the SMF method. Comparison of the full SMF method with the 'strawman' approach showed that the POLL step improves robustness and can lead to a greater cost function reduction with minimal additional cost. In general, the number of iterations required by the SMF method was modest. However, it may be possible to further reduce the cost of the method through surrogate quality improvement. The use of a second Gaussian process in Kriging was introduced in Audet et al. (2000) to prevent surrogate degradation and has been shown to reduce the number of POLL steps in several test cases. This is an area for future study.

In this work, we have demonstrated successful use of the SMF method for an expensive function involving a time-dependent cost function. The methodology described in this paper is not restricted to the unsteady laminar flow problem considered here. It can be applied to a wide range of fluids problems with complex geometries, unsteadiness and turbulence. Because of the portability of the method, it can be coupled to turbulent flow solvers based on LES or unsteady RANS for high Reynolds number flows. Use of the SMF method for time-dependent fluid dynamics problems avoids significant difficulties with the addition of constraints, implementation and data storage that arise with adjoint solvers. In addition, the SMF method shows promise for multi-objective problems involving the integration of several simulation codes. In these problems (for example, fluid structure interaction), gradient information is not easily available, and it is often desirable to treat the function and constraints as a 'black box.' Even in problems in which gradients are available, the SMF method has many desirable properties. Using only the sign of the gradient, polling directions can be 'pruned' to reduce cost (Abramson et al., 2003). Gradient information can be incorporated into the surrogate function to improve accuracy using co-Kriging as demonstrated in Chung and Alonso (2002). The SMF method has proven to reduce the risk of quickly converging to a shallow local minimum, as is often the case in standard gradient methods.

Acknowledgments

This work was supported by the Office of Naval Research under grant N00014-01-1-0423 with Ronald Joslin and Patrick Purtell as program managers. Computer time was provided by the DoD's HPCMP through NRL-DC and ARL/MSRC. John Dennis was supported by the AFOSR, CSRI, the Institute for Mathematics and its Applications, the National Science Foundation and the Ordway Endowment at the University of Minnesota. The authors wish to thank the IMA for providing a forum for collaboration, as well as Charles Audet and Petros Koumoutsakos for valuable discussions. The authors are also indebted to two anonymous referees for their detailed and helpful comments.

References

M. A. Abramson, "Pattern search algorithms for mixed variable general constrained optimization problems," PhD thesis, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 2002.

M. A. Abramson, C. Audet, and J. E. Dennis Jr., "Generalized pattern searches with derivative information," Technical Report TR02-10, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 2003. To appear in *Mathematical Programming Series B*.

N. Alexandrov, J. E. Dennis Jr., R. M. Lewis, and V. Torczon, "A trust-region framework for managing the use of approximation models in optimization," *Structural Optimization* vol. 15, no. 1, pp. 16–23, 1998.

C. Audet, "Convergence results for pattern search algorithms are tight. Les Cahiers du GERAD G-2002-56," Ecole Polytechnique de Montréal, 2002. To appear in *Optimization and Engineering*.

C. Audet and J. E. Dennis Jr, "A pattern search filter method for nonlinear programming without derivatives," Technical Report TR00-09, Department of Computational and Applied Mathematics, Rice University, Houston TX, 2000. To appear in *SIAM Journal on Optimization*.

C. Audet and J. E. Dennis Jr, "Analysis of generalized pattern searches," *SIAM Journal on Optimization* vol. 13, no. 3, pp. 889–903, 2003.

C. Audet, J. E. Dennis Jr., and D. W. Moore, "A surrogate-model-based method for constrained optimization," in *Eighth AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*. AIAA Paper, 00-4891, 2000.

C. Audet and J. E. Dennis Jr, "Pattern search algorithms for mixed variable programming," *SIAM J. Optim.* vol. 11, pp. 573–594, 2000.

C. Bischof, A. Carle, G. Corliss, A. Griewank, and P. Hovland, ADIFOR—generating derivative codes from FORTRAN programs," *Scientific Programming* vol. 1, pp. 1–29, 1992.

W. K. Blake, "A statistical description of pressure and velocity fields at the trailing edge of a flat strut," DTNSRDC Report 4241, David Taylor Naval Ship R & D Center, Bethesda, Maryland, 1975.

A. J. Booker, J. E. Dennis Jr., P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset, "A rigorous framework for optimization of expensive functions by surrogates," *Structural Optimization* vol. 17, no. 1, pp. 1–13, 1999.

H. Chung and J. J. Alonso, "Design of a low-boom supersonic business jet using cokriging approximation models," AIAA Paper, 2002–5598, 2002.

N. Curle, "The influence of solid boundary upon aerodynmaic sound," *Proc. Royal Soc. Lond. A* vol. 231, pp. 505–514, 1955.

C. Davis, "Theory of positive linear dependence," *American Journal of Mathematics*, pp. 448–474, 1954.

R. Fletcher and S. Leyffer, "Nonlinear programming without a penalty function," *Mathematical Programming* vol. 91, pp. 239–269, 2002.

A. A. Guinta, “Use of data sampling, surrogate models, and numerical optimization in engineering design,” AIAA Paper, 2002-0538, 2002.

N. Hansen, D. Müller, and P. Koumoutsakos, “Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES),” *Evolutionary Computation* vol. 11, no. 1, 2003.

N. Hansen and A. Ostermeier, “Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation,” in *Proc. of the 1996 IEEE Int’l. Conf. on Evolutionary Computation*, 1996, pp. 312–317.

M. S. Howe, “The influence of surface rounding on trailing edge noise,” *J. Sound and Vib.* vol. 126, no. 3, pp. 503–523, 1988.

A. Jameson, L. Martinelli, and N. A. Pierce, “Optimum aerodynamic design using the Navier-Stokes equations,” *Theoret. Comp. Fluid Dynamics* vol. 10, pp. 213–237, 1998.

R. M. Lewis and V. Torczon, “Rank ordering and positive bases in pattern search algorithms,” Technical Report 96–71, Institute for Computer Applications in Science and Engineering, Mail Stop 132C, NASA Langley Research Center, Hampton, Virginia 23681–2199, 1996.

R. M. Lewis and V. Torczon, “Pattern search algorithms for bound constrained minimization,” *SIAM J. Optim.* vol. 9, pp. 1082–1099, 1999.

R. M. Lewis and V. Torczon, “Pattern search methods for linearly constrained minimization,” *SIAM J. Optim.* vol. 10, pp. 917–941, 2000.

R. M. Lewis and V. Torczon, “A globally convergent augmented lagrangian pattern search algorithm for optimization with general constraints and simple bounds,” *SIAM J. Optim.* vol. 12, pp. 1075–1089, 2002.

S. N. Lophaven, H. B. Nielsen, and J. Sondergaard, “DACE: A MATLAB Kriging toolbox version 2.0,” Technical Report IMM-TR-2002-12, Technical University of Denmark, Copenhagen, 2002.

A. L. Marsden, M. Wang, and P. Koumoutsakos, “Optimal aeroacoustic shape design using approximation modeling,” Annual Research Briefs, Center for Turbulence Research, Stanford University, 2002.

M. D. McKay, W. J. Conover, and R. J. Beckman, “A comparison of three methods for selecting values of input variables in the analysis of output from a computer code,” *Technometrics* vol. 21, pp. 239–245, 1979.

R. Myers, *Response Surface Methodology*, Allyn and Bacon: Boston, 1971.

J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Verlag: New York, 1999.

Y. S. Ong, P. B. Nair, and A. J. Keane, “Evolutionary optimization of computationally expensive problems via surrogate modeling,” *AIAA Journal* vol. 41, no. 4, pp. 687–696, 2003.

O. Pironneau, *Optimal Shape Design for Elliptic Systems*, Springer-Verlag: New York, 1984.

J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, “Design and analysis of computer experiments,” *Statistical Science* vol. 4, no. 4, pp. 409–423, 1989.

D. B. Serafini, “A framework for managing models in nonlinear optimization of computationally expensive functions,” PhD thesis, Rice University, Houston, TX, 1998.

V. Torczon, “On the convergence of pattern search methods,” *SIAM J. Optimization* vol. 7, pp. 1–25, 1997.

V. Torczon and M. W. Trosset, “Using approximations to accelerate engineering design optimization,” Technical report, NASA Langley Research Center, 1998.

M. Wang, S. K. Lele, and P. Moin, “Computation of quadropole noise using acoustic analogy,” *AIAA J.* vol. 34, no. 11, pp. 2247–2254, 1996.

M. Wang and P. Moin, “Computation of trailing-edge flow and noise using large-eddy simulation,” *AIAA J.* vol. 38, no. 12, pp. 2201–2209, 2000.

M. Wang and P. Moin, “Dynamic wall modeling for large-eddy simulation of complex turbulent flows,” *Physics of Fluids* vol. 14, no. 7, pp. 2043–2051, 2002.