2.0 Design Optimization Problem Formulation

2.1 General Formulation

Numerical optimization is a vast field which has been the subject of numerous text books [1,103,104]. To keep the scope of the discussion focused in this section, only gradient-based (sometimes referred to as first-order) methods will be treated. The solution procedure for gradient-based methods may be decomposed into four distinct steps: (a) evaluation of the objective function, F, to be minimized or maximized and any constraints, C_j, to be imposed, (b) evaluation of the gradients of the objective function, F, and constraints, C_j, with respect to the vector of design variables β_k, commonly referred to as sensitivity derivatives, (c) determination of the search direction, $\vec{s}$, upon which the design variables will be updated, and (d) determination of the optimum step length, α, along this search direction (termed line search). Techniques to accomplish these steps will be discussed in this, and subsequent, chapters.

Formally, the aforementioned procedure for constrained minimization may be stated as

$$\underset{\beta_k}{minimize} \;\; F(\beta_k) \tag{2.1}$$

subject to the vector of inequality constraints

$$C_j(\beta_k) \le \vec{0} \qquad j = 1, ncon \tag{2.2}$$

and possible side constraints on the vector of design variables

$$\beta_k^l \le \beta_k \le \beta_k^u \qquad k = 1, ndv \tag{2.3}$$

where β_k^l and β_k^u are the lower and upper bounds of the design variables, respectively. The design variables during the m^{th} design cycle may be updated as

$$\beta_k^m = \beta_k^{m-1} + \alpha\, \bar{s}^{m-1} \tag{2.4}$$

with the corresponding objective function

$$F\left(\beta_k^m\right) = F\left(\beta_k^{m-1} + \alpha\, \bar{s}^{m-1}\right) = F(\alpha) \tag{2.5}$$

Thus, while searching in the direction of $\bar{s}$, the design problem reduces from *ndv* variables to the determination of the scalar step length that minimized the objective function along the search direction. The iterative process of determining the step length, α, is referred to as the one-dimensional or line search algorithm, and will be discussed in section 2.3.2.

For gradient-based optimization methods, the search direction is determined using first derivatives of the objective function and constraints with respect to the vector of design variables. This is not to say that the search direction is soley based in first-derivative information. It is possible to *estimate* second-order derivatives using the *computed* first derivatives. When the evaluation of the objective function and constraints are also a function of a state vector $Q(\beta_k)$, whose value requires the satisfaction of a corresponding state equation $R(Q, \beta_k) = 0$, several alternatives exist for evaluating the needed sensitivity derivatives. The above optimization procedure will be denoted as *traditional* gradient-based numerical optimization when the state equation is exactly satisfied at each iteration (also known as a design feasible approach) during the design process.

2.2 Systems Governed by Partial Differential Equations

For aerodynamic optimization, the state equation is comprised of a system of partial differential equations (PDE). At this point differentiation of the system of PDE can be performed at one of two levels. The first method, termed the continuous or variational approach, differentiates the PDE prior to discretization. This method utilizes fundamental calculus of variations to define an *adjoint* set of equations to the continuous governing PDE. Subsequently, these adjoint equations are discretized and solved. The second method, termed the discrete approach, differentiates the PDE after discretization. In the present work, the discrete approach is adopted. For a more detailed discussion of the continuous approach to aerodynamic design optimization, the interested reader is directed to the literature [10,41,48,52,105].

For discrete aerodynamic shape sensitivity analysis the objective function and constraints may, in general, be expressed as $F(Q, X, \beta_k)$ and $C_j(Q, X, \beta_k)$, respectively. Here, Q is the disciplinary state vector on which the objective or constraint is defined, X is the computational mesh over which the PDE is discretized, and β_k is the vector of design variables which control the shape of the configuration. The sensitivity derivatives of the objective function, F, and the constraints, C_j, may be simply evaluated by finite differences; however, this approach is not only computationally expensive, it has been found at times to produce highly inaccurate gradient approximations. The preferable approach is to obtain the discrete sensitivity derivatives quasi-analytically via

$$F = \frac{\partial F}{\partial \beta_k} + \left(\frac{\partial F}{\partial Q}\right)^T \frac{\partial Q}{\partial \beta_k} + \left(\frac{\partial F}{\partial X}\right)^T \frac{\partial X}{\partial \beta_k} \tag{2.6a}$$

$$C_j = \frac{\partial C_j}{\partial \beta_k} + \left(\frac{\partial C_j}{\partial Q}\right)^T \frac{\partial Q}{\partial \beta_k} + \left(\frac{\partial C_j}{\partial X}\right)^T \frac{\partial X}{\partial \beta_k} \tag{2.6b}$$

To compute the sensitivity derivatives in Eqs.(2.6a,b), the sensitivity of the state vector $\partial Q/\partial \beta_k$ is needed. It should be noted that the sensitivity of the state vector is comprised of two parts; an interior cell contribution and a boundary contribution. The origins of these components will be discussed in greater detail in a later section. Nevertheless, this approach is referred to as the direct differentiation method and results in the difficulty of solving an extremely large system of linear equations. The number of systems needing to be solved is equal to the number of design variables, ndv. If, in the design problem under consideration, the sum of the objective function and constraints is less than the number of design variables (i.e., $ncon+1 < ndv$), a more efficient alternative approach may be formulated. This method is referred to as the discrete-adjoint variable approach, and may be written as

$$F = \frac{\partial F}{\partial \beta_k} + \left(\frac{\partial F}{\partial X}\right)^T \frac{\partial X}{\partial \beta_k} - \lambda_F^T \frac{\partial R}{\partial X}\frac{\partial X}{\partial \beta_k} \tag{2.7a}$$

$$C_j = \frac{\partial C_j}{\partial \beta_k} + \left(\frac{\partial C_j}{\partial X}\right)^T \frac{\partial X}{\partial \beta_k} - \lambda_{C_j}^T \frac{\partial R}{\partial X}\frac{\partial X}{\partial \beta_k} \tag{2.7b}$$

where λ_F and λ_{C_j} are adjoint vectors defined in such a way as to eliminate the dependence of the objective function and constraints on the design variables, and R is the disciplinary state equation. Similar to the direct differentiation approach, these adjoint vectors must be defined at both interior cells and boundary points. Furthermore, this method requires the solution of $ncon+1$ linear systems, and will be discussed in section 3.3.2.

The equations and methods used in the current work to obtain the state vectors for the aerodynamic system, and the sensitivity of this state vector, will be presented in chapters to follow.

2.3 Gradient-Based Numerical Optimization

As posed in Eq.(2.4), the first task in producing the design variable update during the m^{th} design cycle relies on the determination of a search direction. The choice of this search direction is not unique, but must be such that small changes in the design variables in this direction improve the objective function without violating the constraints. Once a search direction has been found, all points located along this direction may be expressed in terms of a scalar step length. The design problem then reduces to finding the step length that produces the best possible design along the given search direction. The methods used in the present work to determine the search direction and step length are discussed below.

2.3.1 Search Direction Determination

For unconstrained problems, the most well known method for determining the search direction is the steepest descent method. This steepest descent direction is simply

$$\vec{s} = -\frac{F}{\| \; F\|} \tag{2.8}$$

where $\|\;\|$ denotes the Euclidean norm. This search direction is insufficient for constrained minimization problems, however, since it does not account for constraint boundaries. The steepest descent method would produce a *useable* direction (i.e., any direction that will reduce the objective function), but not a *feasible* direction (i.e., any direction that for a small step in this direction, the design does not violate any constraint). Mathematically, a useable direction is one in which

$$\vec{s}^{\,T} \quad F < 0 \tag{2.9}$$

and a feasible direction satisfies

$$\vec{s}^{\,T} \quad C_j > 0 \qquad j \quad I_A \tag{2.10}$$

where I_A is the set of critical, or near active, constraints for the current design.

Thus there are two criteria for the determination of a search direction: (1) reduce the objective function as much as possible, and (2) keep away from the constraint boundaries as much as possible. The technique used in the present work to determine this search direction is the method of feasible directions [106,107]. This method solves a maximization problem for the search direction as

$$\underset{\vec{s}}{maximize} \quad \xi \tag{2.11a}$$

subject to

$$-\vec{s}^{\,T} \quad C_j + \theta_j \xi \leq 0 \quad j \quad I_A \tag{2.11b}$$

$$\vec{s}^{\,T} \quad F + \xi \leq 0 \tag{2.11c}$$

$$\vec{s}^{\,T} \quad \vec{s} \leq 1 \tag{2.11d}$$

where the θ_j are positive numbers called *push-off* factors which determine how far the design will move from the constraint boundaries. Note that in the above optimization problem, the components of the search direction are the *design variables* to be determined; the search is therefore called a direction-finding process [107]. For maximum reduction in the objective function the θ_j may be set to zero. The solution to the above maximization problem then becomes $\xi = -\vec{s}^{\,T} \quad F$. Furthermore, for finite values of θ_j, if the solution to Eq.(2.11) above results in $\xi > 0$ then a useable-feasible direction has been determined, and if $\xi = 0$ then the current design satisfies the necessary conditions for an isolated local minimum, e.g., the Kuhn-Tucker necessary conditions.

2.3.2 Line Search Algorithm

When the optimization process reaches the line search algorithm, the design problem has been converted from *ndv* variables to one that consists of finding the scalar step length that best improves the design along the predetermined search direction (see Eq.(2.5)). Several techniques are available to accomplish this one-dimensional search. Typical examples are bracketing methods [108], polynomial interpolation [109], and the Golden Section method [110,111]. Bracketing methods attempt to “bracket” the minimum between two points through recursive function evaluation. Polynomial interpolation evaluates the function at several sample points, and then constructs a polynomial fit of the data from which the minimum can easily be found. These methods, however, assume that the minimum exists in the space of the points tested, and usually rely on a bracketing method to first determine the upper and lower bounds. The Golden Section method first assumes that the one-dimensional design space is unimodal, i.e., only one minimum exists. Then, by optimum placement of the sample points, the minimum is systematically found. As shown in Ref. 1, a relationship exists to determine the total number of function evaluations required to obtain a specified interval of uncertainty about the function minimum. A more detailed discussion of this algorithm may be found in references 1,103,110, and 111. The shortcoming to each of these techniques is the large number of function evaluations required to determine the optimum step length along the line search for highly nonlinear problems. In the current work, the computational costs of evaluating the objective function is excessive and the design space tends to be highly nonlinear. Thus, a constant increment, $\Delta\alpha$, is chosen in advance to traverse the design space, and for the n^{th} iteration of the line search the step length is

$$\alpha^{n} = \alpha^{n-1} + \Delta\alpha \tag{2.12}$$

This method approaches the optimum step length in one direction, and has the advantage of only perturbing the design slightly between successive iterations. Hence, the solution to the state equations at one iteration may be used as a good initial guess at subsequent steps.