AIAA 96-0555

Three-Dimensional Aerodynamic Shape Optimization Using Genetic Evolution and Gradient Search Algorithms

Norman F. Foster and George S. Dulikravich
The Pennsylvania State University
University Park, PA

Jeff Bowles
NASA Ames Research Center
Moffett Field, CA

34th Aerospace Sciences Meeting & Exhibit
January 15-18, 1996 / Reno, NV

Three-Dimensional Aerodynamic Shape Optimization Using Genetic Evolution and Gradient Search Algorithms

Norman F. Foster* **and George S. Dulikravich†**
Department of Aerospace Engineering
The Pennsylvania State University
University Park, PA 16802

Jeff Bowles
Systems Analysis Branch
Mail Stop 237-11
NASA Ames Research Center
Moffett Field, CA 94035-1000

Abstact

This study introduces various gradient search methods as well as hybrid genetic techniques that achieve impressive convergence rates on constrained problems. These methods are applied to three-dimensional shape optimization of ogive-shaped, star-shaped, and spiked projectiles and lifting bodies in a hypersonic flow. Flow field analyses are performed using Newtonian flow theory and, in certain cases verified using a parabolized Navier-Stokes (PNS) flow analysis algorithm. Three-dimensional geometrical rendering is achieved using a variety of techniques including beta-splines from the computer graphics industry.

Introduction

An integral part of a system's development very often includes the determination and design of the shape of some surface.

Shape design generally affords a special degree of difficulty. Determination of a surface's geometry so that some dependent parameter(s) is satisfactory involves a certain degree of intuition. The designer often needs cosiderable personal experience in order to predict whether the aesthetics of some shape will be functional. In an attempt to reduce the need for a designer's intuition, this study presents a tool to assist in the aerodynamic shape design process

Previous Work

Optimization of hypersonic bodies with axisymmetric cross sections has been performed in the past[1-6]. These accomplishments did not address optimization of arbitrary and complex geometries. Optimization of arbitrary three-dimensional hypersonic vehicles was made feasible[7-9] by the use of an inexpensive method (a method which will also be used in this work) of determining aerodynamic forces. A problem faced was a limited 'rank two update' optimization process which could not operate on highly constrained problems. Other works have concentrated on optimization of small regions within a larger configuration, such as nose shape[10], forebody[11], airfoils[12], wing size[13], and scramjet engine integration[14].

Optimization Problem Statement

In the most general sense, optimization is the process of achieving the best outcome of a given operation while satisfying a set of given restrictions. The *cost* (or *objective*) *function* is the term applied to this outcome that needs to be improved (or *optimized*). In a computational sense, this cost function is expressed as a scalar value and it is mathematically dependent upon a set of variables. These variables are referred to as the *design variables*. The design variables are the unknowns sought such that the cost function is improved. As a convention, the cost function is typically said to be optimized if it is reduced. The 'best' solution of an optimization problem would be the set of design variables such that the cost function reaches its global minimum value.

Most often in engineering problems the design variables are restricted in some sense. Such a restriction is referred to as a *constraint*. The existence of constraints placed upon design variables can dramatically alter the nature, complexity, and solution method of an optimization problem. A set of design variables that does not violate any constraint is said to be *feasible*, while one that does violate a constraint is said to be *infeasible*. If a constraint is violated, or on the verge of violation, it is known as an *active* constraint.

---

* Presently Engineer, Computational Fluid Dynamics, Knolls Atomic Power Laboratory, Schenectady, NY

† Associate Professor, Associate Fellow AIAA

design can still be selected for reproduction. Elitism ensures that the value for the best fitness does not worsen from one generation (design cycle) to the next. However elitism might lead to narrowing of the choices and convergence to a local minimum.

Genetic Algorithms and Constraints

The classical GA can handle bounds (boundary constraints) on the design variables, but it is not inherently capable of handling equality or inequality constraint functions[21]. Previous implementations of the GA have involved problems posed in such a way as to eliminate constraint functions, or to penalize the cost function when a constraint is violated. These treatments of constraints reduce the chance of arriving at the global minimum.

One successful method of satisfying constraints during a genetic optimization process is to actively ensure that the initial design population, as well as any later generated population members (created by crossover and mutation of the previous generation) are feasible. This can be accomplished by performing a *feasible search* on any design that is determined to be infeasible[15]. A feasible search is a sub-optimization problem in itself. One type of feasible search sub-problem would be to implement steepest descent method so as to minimize $\sum_i \left| g_i(\mathbf{x}) \right| \; i \in I_a$.

Another method for bringing an infeasible design into the feasible domain is to utilize the restoration move used in Rosen's projection method , Equation (10), until all violated constraints are simply active.

While implementing feasible searches is not a method of transforming the classical GA into a true constrained algorithm, it can effectively lead the GA to constrained solutions. This is because, before a new generation of designs is produced, all of its parent designs have been proven to be feasible. By doing so, the GA can be altered so it can handle non-linear constraint functions, which is not at all common for the classical GA.

If an entire set of newly generated designs has been analyzed without any improvement to the best current design, then a search direction is determined as is done according to the Nelder-Mead method[22]. A line search is then performed in this search direction.

This new type of genetic algorithm that incorporates Rosen's restoration move to handle constraints, and computation of a search direction using method[22] followed by a line search if there is no reduction in the cost function after a standard GA design cycle, will be referred to as the *hybrid genetic* algorithm. The hybrid gradient and the hybrid genetic algorithm are the two optimization techniques that were utilized in this study.

Geometry Treatment

The external geometry of the hypersonic vehicle configuration will constitute the design variables. Therefore, the entire geometry must be represented by a single vector of scalar quantities.

In this study, the method for describing the three-dimensional vehicle geometry was done in several ways depending upon the type of problem considered. However, in all cases the surface nodes did not move axially along the length of the body. All points were grouped onto cross sectional planes, and the total length of the vehicle was always preserved. Each surface point location was specified as to how it can vary on its cross section. Examples of different kinds of point motions include cartesian and polar motion, and motion according to a chosen spline technique[16].

Beta-Spline Control

There is a parametric piecewise curve representation method that uses what is called *beta-splines*. These beta-splines[23] have been used for geometric computer modeling, and are closely related to the more common v-splines and B-splines or Bernstein polynomials. A beta-spline curve is specified by a set of points called *control vertices*. The positions of these vertices completely define the shape of the dependent curve (beta-spline), although they generally do not lie on it. The vertices are an ordered sequence of points that form what is referred to as the *control polygon*.

The beta-spline utilizes a piecewise representation, in order to achieve local control, by defining each segment as a function of only a few adjacent vertices. Specifically each curve segment can be regarded as a weighted average of four local vertices. Let

$$\mathbf{G}_i(u) = \begin{Bmatrix} x_i(u) \\ y_i(u) \end{Bmatrix} \tag{20}$$

denote the i^{th} parametric two dimensional beta-curve segment, where u is a non-dimensional curve-following coordinate that varies from 0 at the beginning of the segment, to unity at the end of the segment. This segment is dependent on four local vertices according to

$$\mathbf{G}_i(u) = \sum_{r=-2}^{1} b_r(\beta_1, \beta_2; u)\mathbf{V}_{i+r} \quad (0 \le u < 1) \tag{21}$$

where $\mathbf{V}_{i+r}$ is the $(i+r)^{\text{th}}$ control vertex position vector. The scalar weighting factors in Equation (21), $b_r(\beta_1, \beta_2; u)$, are called *basis functions*. These basis functions depend on the domain parameter, u, and two

cost function calculated at that design point. Thus a fitness vector is established as

$$\mathbf{fit} = \begin{Bmatrix} -f(\mathbf{x}_1) \\ -f(\mathbf{x}_2) \\ \vdots \\ -f(\mathbf{x}_{n_{pop}}) \end{Bmatrix} \tag{18}$$

When the cost function has been computed for each design (population member), the population members (designs) are ranked according to their fitness. Using this information, **M** and **fit** can be reordered according to descending fitness such that the first row of **M** contains the design with the best fitness, $\mathbf{fit}_1$, and so forth.

Next, population members are selected for 'reproduction' based upon their fitness. That is to say, design sets corresponding to lower cost functions possess a better chance for being selected for the reproduction process. Typically, n_{pop} number of member (design) pairings are determined through some random selection process that is weighted such that the members (designs) with a greater fitness have a greater chance of being selected. Any member (design) can appear any number of times in the pairings, and duplicate pairings are allowed.

Once these pairings have been established, the 'crossover' procedure takes place. In this step, for each reproduction pairing, the two selected population members (or *parents*) are merged to create two new design sets (or *children*). In order to merge the two parent design sets, each design variable of both parents must be 'coded' into what is referred to as *bit strings*. These bit strings are binary ($base_2$) representations of the design variables' values as a percentages of their allowed values defined by their upper and lower bound constraints. For instance, the first design variable of population member (design) #1 is denoted as $x_{1,1}$. The bit string representation for this variable would be

$$x_{1,1}^{\text{bit}} = \text{base}_2\left\{\text{int}\left[\left(\frac{x_{1,1} - x_{L_1}}{x_{U_1} - x_{L_1}}\right)\left(2^{L_{str}} - 1\right)\right]\right\} \tag{19}$$

where x_{L1} and x_{U1} are the upper and lower bound constraints specified for the first design variable, and L_{str} is the number of binary digits (*bits*) that have been previously allotted to describe the design variable. In effect, L_{str} is a measure of precision of the $base_2$ representation of the design variable, and is chosen before the optimization process. From Equation (19), it should be noted that upper and lower boundary constraints are *required* for the coding (and decoding) process. As an illustration of this coding process, let the variables in Equation (19) take on example values. Suppose $x_{1,1}$= 4.0 and its boundary constraints are x_{U_1}=11.0 and x_{L_1}=1.0 . Then, using a bit string precision of $L_{str} = 5$,

$$x_{1,1}^{\text{bit}} = \text{base}_2\left\{\text{int}\left[\left(\frac{4.0-1.0}{11.0-1.0}\right)\left(2^5-1\right)\right]\right\}$$

$$= \text{base}_2(9).$$

Since

$$\left(2^{L_{str}} - 1\right) = \left(2^5 - 1\right) = 31 = 2^4 + 2^3 + 2^2 + 2^1 + 2^0$$

can be represented as a binary number (11111), then

$$x_{1,1}^{\text{bit}} = \text{base}_2(9) = 01001 = 0 + 2^3 + 0 + 0 + 2^0.$$

Once the design variables of the two parents of a reproduction pairing have been coded into bit strings, then the crossover procedure can begin. Stepping by one design variable at a time, the bit strings of the design variables of each parent are cut at a random location and swapped. For the purpose of illustration let design parent #1 equal $\{4.0 \quad 6.0\}^T$, let design parent #2 equal $\{7.0 \quad 5.0\}^T$, and let the boundary constraints be $\mathbf{x}_U = \{11 \quad 11\}^T$ and $\mathbf{x}_L = \{1 \quad 1\}^T$. Table 1 shows decimal and binary representations for each parent, and for the new resulting children after the crossover has taken place. This crossover process is conducted for every set of reproductive parings.

When the crossover procedure is complete, a *mutation* procedure is conducted. In the mutation process, every bit of every design variable of every child design is subject to a chance for change from a '0' to a '1', or vise-versa. This chance for change is referred to as a *mutation probability* and is typically a small percentile. That is, there is a small chance for each bit in a child design to be toggled. This procedure introduces another element of randomness to the overall optimization process.

Completion of the mutation process marks the end of a design cycle. At this point, the fitness of each design child is evaluated (cost function analysis). Then the new population is ranked and reordered as discussed earlier. This is followed by reproduction, crossover, mutation, and so forth. This cycle is continued until the best fitness design reaches some acceptable value, or until it does not change after many iterations.

A common practice known as *elitism* is often used to ensure that the design corresponding to the best fitness is not lost. Essentially, it involves transferring the best design directly forward into the next generation without crossover or mutation. This best

The vector **j** contains positive numbers called 'push-off' factors because their magnitudes determine how far **x** will move from the constraint boundaries. For example, if j_1 is a large number in comparison to the other entries in the **j** vector then the resultant search direction vector **s** will tend to point in the direction more normal to the first active constraint. The vector **b** which minimizes Equation (13) yields the direction vector **s** that compromises between two goals: minimizing the cost function, and attempting to stay as far away from any active constraint boundaries as possible. This search direction is referred to as the *usable-feasible* direction: *usable* because it reduces the cost function, and *feasible* because it points in the direction that does not immediately violate any constraints.

Once a search direction has been determined, then a line search is performed to find α^1 so as to minimize $f(\alpha^1) = f(\mathbf{x}^0 + \alpha^1 \mathbf{s}^0)$ along $\mathbf{s}^0$. Once again, the line search should ensure that the resulting value for α^1 reduces the cost function and at the same time yields a feasible design when the design variables are lastly updated by $\mathbf{x}^1 = \mathbf{x}^0 + \alpha^1 \mathbf{s}^0$.

Hybrid of Projection and Feasible Direction Methods

While it is clear that the previously discussed methods all have different uses, advantages, and disadvantages, it would be very desirable to incorporate most of their individual strengths into one optimization technique.

In order to effectively manage both equality and inequality constraints, a natural approach would be to join Rosen's projection method with the MFD. Simply put, the search direction will be determined by first computing a usable-feasible search direction $\mathbf{s}_{uf}$ by minimizing Equation (13) with any *active inequality* constraints, then projecting it into a subspace tangent to any *equality* constraints that exist by using

$$\mathbf{s} = [\mathbf{P}]\{\mathbf{s}_{uf}\} \tag{14}$$

where

$$\mathbf{P} = \mathbf{I} - \mathbf{N}(\mathbf{N}^{\mathrm{T}}\mathbf{N})^{-1}\mathbf{N}^{\mathrm{T}} \tag{15}$$

and

$$\mathbf{N} = \left[\nabla \mathbf{h}_1 \middle| \nabla \mathbf{h}_2 \middle| \ldots \middle| \nabla \mathbf{h}_i\right] \qquad i = 1,2,\ldots,n_e \tag{16}$$

When this search direction has been computed, a line search should be performed to find α^{k+1} with the objective to reduce the cost function so that the resulting update in the design $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha^{k+1}\mathbf{s}^k$ does not violate a constraint.

This method, which will be referred to as the *hybrid gradient* method, was one of two optimization techniques that were utilized in this study[16].

Genetic Evolution Algorithms

Genetic algorithms (GA) are non-gradient methods[17-18,15] that offer a promising answer to complex optimization problems.

In general, a genetic algorithm is broken into three major steps: reproduction, crossover, and mutation. An initial population of *complete* design variable sets is analyzed according to some cost function. Then this population is merged using a crossover methodology to create a new population. This process continues until a global minimum is found. Generally, the design variable set that corresponds to the current minimum point will be representative of the most 'successful' features of previous 'generations' of designs in the optimization process. The GA can be exceptional at avoiding local minima because it tests possible designs over a large domain in the design variable space.

In order to begin the genetic evolutionary process, an initial population (of designs) must be defined. Each population member (design) is in itself an entire complete set of design variables. The population may be represented in matrix-form as

$$\mathbf{M} = \begin{bmatrix} \text{Pop. member}_1 \\ \text{Pop. member}_2 \\ \vdots \\ \text{Pop. member}_{n_{pop}} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^{\mathrm{T}} \\ \mathbf{x}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{x}_{n_{pop}}^{\mathrm{T}} \end{bmatrix} \tag{17}$$

$$= \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n_{pop},1} & x_{n_{pop},2} & \cdots & x_{n_{pop},n} \end{bmatrix}$$

where

n = number of design variables

n_{pop} = number of population members (designs)

Each member in this initial population can be specified meaningfully by performing inverse design of aerodynamic shapes[19-20]. However, commonly only the first member is specified and the rest are generated randomly.

Once the initial population has been constructed, each member is evaluated for what is referred to as its *fitness*. Essentially the fitness of a population member is a measure of its ability to solve the problem at hand. In common optimization terms, this means that the fitness of a population member is the negative of the

Optimization problems faced by engineers are commonly constrained. The general constrained optimization problem can be mathematically stated as

minimize: $f(\mathbf{x}) \quad \mathbf{x} = \{x_1, x_2, \ldots, x_n\} \quad \mathbf{x}: \mathbf{x} \in \Re^n$ (1)

such that: $\mathbf{x}_L \le \mathbf{x} \le \mathbf{x}_U$ (2)

$g_j(\mathbf{x}) \le 0 \qquad j = 1,2,\ldots,n_g$ (3)

$h_k(\mathbf{x}) = 0 \qquad k = 1,2,\ldots,n_e$ (4)

Here, f is the cost function, $\mathbf{x}$ is the vector of n design variables, $\mathbf{x}_L$ is the vector of lower boundary constraints, $\mathbf{x}_U$ is the vector of upper boundary constraints, g_j is one of n_g inequality constraints, and h_k is one of n_e equality constraints.

Rosen's Projection Method

This method[15] is specifically designed to handle the existence of constraints. It is based on the idea of projecting the search direction into the subspace tangent to any active constraints.

Let $\mathbf{N}$ be defined as a matrix composed of the active constraint gradients augmented columnwise

$$\mathbf{N} = \left[\nabla\mathbf{g}_1 | \nabla\mathbf{g}_2 | \ldots | \nabla\mathbf{g}_i\right] \qquad i \in I_a \tag{5}$$

where I_a is the set of n_{ac} active constraints. From some initial design, $\mathbf{x}^0$, a search direction is projected onto a surface tangent to the active constraints. That is,

$$\mathbf{s} = -\mathbf{P}\nabla f \tag{6}$$

where $\mathbf{P}$ is the projection matrix defined as

$$\mathbf{P} = \mathbf{I} - \mathbf{N}\left(\mathbf{N}^{\mathrm{T}}\mathbf{N}\right)^{-1}\mathbf{N}^{\mathrm{T}} \tag{7}$$

Note that for real engineering problems, there are often many more design variables than active constraints. Therefore, $\mathbf{N}$ is typically not a square matrix and so Equation (7) does not yield zero.

After the search direction has been determined, then the design is updated by

$$\mathbf{x}^1 = \mathbf{x}^0 + \alpha^1\mathbf{s}^0 \tag{8}$$

The value for α^1 can be determined from a standard line search (while ensuring that its final value will not lead to a constraint violation), or it can be determined by specifying a desired percentage decrease in f. This can be computed by

$$\alpha^1 = -\frac{pf}{\mathbf{s}^{\mathrm{T}}\nabla f\left(\mathbf{x}^0\right)} \tag{9}$$

where p is the percentage decrease in the cost function that is desired.

After it has been updated, the resulting design $\mathbf{x}^1$ obtained by Equation (8) may become infeasible. This can easily happen when Equation (9) is used to determine α. If the design is found to be infeasible, then a *restoration* move is required.

In order to restore the design to a location $\mathbf{x}^b$ lying along the constraint boundary, a restoration move is required from $\mathbf{x}^1$ so as to reduce $g_1(\mathbf{x})$ to zero. This restoration move is defined as

$$\left(\mathbf{x}^b - \mathbf{x}^1\right) = -\mathbf{N}\left(\mathbf{N}^{\mathrm{T}}\mathbf{N}\right)^{-1}\mathbf{g}_a\left(\mathbf{x}^1\right) \tag{10}$$

where $\mathbf{g}_a(\mathbf{x}^1)$ is the vector containing the values of the active constraints in I_a. Note that $\mathbf{N}$ should be evaluated at $\mathbf{x}^1$ for high accuracy; however constraint gradient values at $\mathbf{x}^0$ can often be used as an approximation to those at $\mathbf{x}^1$, thus making the approach more economical.

Rosen's projection method is therefore usually a two step process. First, the search direction is determined by projecting $-\nabla f(\mathbf{x}^0)$ along a direction tangent to $g_i(\mathbf{x}^0)$ $i \in I_a$ using Equations (5) through (7). After the design is updated by Equation (8), then if it is determined that $\mathbf{x}^1$ is infeasible or too far away from the constraint boundary, then a revised design is obtained by

$$\mathbf{x}^{1\,\text{revised}} = \mathbf{x}^1 + \left(\mathbf{x}^b - \mathbf{x}^1\right) = \mathbf{x}^b \tag{11}$$

with the use of Equation (10).

This projection method is particularly well suited for problems with equality constraints (replace $\mathbf{g}_i$ with $\mathbf{h}_i$ in Equation (5) because the progress of the design follows constraint boundaries. Therefore at any design cycle, including the ending point, at least one constraint is active.

Method of Feasible Directions

This is another method[15] designed to handle constrained optimization problems. Unlike Rosen's projection method which follows constraint boundaries at all times, the *method of feasible directions* (MFD) attempts to search in directions that avoid any constraint boundaries.

Suppose at the current design, $\mathbf{x}^0$, two inequality constraint functions are active. In order to determine the search direction, $\mathbf{s}$, in which to proceed, gradients of the active constraints g_1 and g_2, as well as the cost function, f, must be computed. With this gradient information the search direction can be computed by minimizing a subproblem. This subproblem can be stated as

find: $\mathbf{b} = b_i \qquad i = 1,2,\ldots,\left(n_{ag}+1\right)$ (12)

that minimizes: $f_{sub} = -\mathbf{b}^{\mathrm{T}}\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)\mathbf{j} + \mathbf{b}^{\mathrm{T}}\mathbf{b}$ (13)

where,

$\mathbf{A} = \left[\nabla\mathbf{g}_1 \middle| \nabla\mathbf{g}_2 \middle| \ldots \middle| \nabla\mathbf{g}_j \middle| \nabla f\right] \qquad j \in I_{ag}$

$I_{ag} \equiv$ set of n_{ag} active inequality constraint functions.

When this subproblem has been solved, then the search direction is given by

$$\mathbf{s} = -\mathbf{A}\mathbf{b}$$

shape parameters, β_1 and β_2. Each basis function is itself defined as a cubic polynomial,

$$b_r(\beta_1,\beta_2;u) = \sum_{j=0}^{3} c_{j,r}(\beta_1,\beta_2)u^j \tag{22}$$

In Equation (22), there are sixteen unknown constants, $c_{j,r}(\beta_1,\beta_2)$, where $0 \le u < 1$, j = 0, 1, 2, 3 and r = -2, -1, 0, and 1. These constants are fixed quantities (provided β_1 and β_2 are fixed) that can be found by imposing the following three connectivity boundary conditions on any two segments.

$$\mathbf{G}_{i+1}(0) = \mathbf{G}_i(1) \tag{23}$$

$$\frac{d\mathbf{G}_{i+1}(0)}{du} = \beta_1 \frac{d\mathbf{G}_i(1)}{du} \tag{24}$$

$$\frac{d^2\mathbf{G}_{i+1}(0)}{du^2} = \beta_1^{\,2} \frac{d^2\mathbf{G}_i(1)}{du^2} + \beta_2 \frac{d\mathbf{G}_i(1)}{du} \tag{25}$$

The first boundary condition, Equation (23), enforces simple connectivity. The second boundary condition, Equation (24), is derived from enforcing continuity of the unit tangent vector at the joint of two segments. The last boundary condition, Equation (25), comes from imposing continuity of the curvature vector (and thus curvature) at the connecting point of two adjacent segments.

The shape parameter β_1 is referred to as the *bias* parameter. For purposes of the present study, the bias parameter was set to unity. The other shape parameter, β_2, is called the *tension* parameter and should always be positive. For high values of β_2 the beta-curve will be strongly attracted to the control vertices, and in the limit, will be identical to the control polygon. In this study, β_2 was also fixed at unity.

Implementing beta-splines for the control of the geometry in this study was accomplished by specifying the number of control vertices on each of the x,y parallel planes. The locations of the control vertices are defined to be the design variables to be optimized.

Results

A single computer program was produced that performed three-dimensional hypersonic vehicle shape optimization. Two optimization techniques were included: the hybrid gradient and the hybrid genetic techniques, both using Rosen's projection methodology for improved equality constraint treatment. Also, two analysis methods were included: MNIT (modified Newtonian impact theory) and a PNS solver. Either optimizer could use either flow solver at any point during execution.

Verification Test 1: Half Sphere-Cone Analysis

A series of hypersonic wind tunnel experiments[24] were reported in 1962. Some of these experiments were performed on three-dimensional models called 'half sphere-cones'. Aerodynamic measurements were taken for a wide range of angles of attack, for several different values of Mach number. This test case attempts to duplicate these experimental results using the MNIT flow solver.

One particular set of experimental data was chosen for comparison. The geometry of the chosen half sphere-cone is shown in Figure 1. The radius of curvature of the hemispherical nose measured 0.365 inches. The diameter of the semi-circular rear plane measured 2.43 inches. The total length of the model measured 4.0 inches, and the planform area was 6.025 square inches. This model was subjected to a flow at a Mach number of 12.6; the Reynolds number (based on model length) was 490,000. Measurements were taken for various angles of attack.

These test conditions were duplicated and analyzed using the MNIT flow solver. The MNIT analyses were performed for angles of attack ranging from -60 to 60 degrees, stepping every 10 degrees. Figure 2 shows experimental data plotted with MNIT computed values for lift-to-drag ratio. Figure 3 displays computed and experimental values for lift and drag coefficients. These coefficients use planform area and a dynamic pressure of 2,320.78 pounds per square foot for normalization.

The computed results as shown in Figures 2 and 3 show a reasonable agreement with the experimental data. Therefore, at these flight conditions the MNIT maintained a high level of accuracy.

Verification Test 2: Wave Drag Minimization

This test case is intended to validate the operation of the optimization techniques. In this exercise, the geometry of a three-dimensional axisymmetric body was optimized to reduce wave drag at zero angle of attack

Optimal bodies of revolution that minimize drag have previously been analytically determined. Two such solutions are known as the Von-Karman and Sears-Haack bodies[25]. These two bodies yield the minimum drag under two different sets of assumptions.

The Von-Karman body assumes that the body terminates with a flat plane, that the base area in this plane is known, and that the total length of the body is specified. The expression for the radius as a function

of axial distance from the nose for this body is

$$r_{VK}(x) = \sqrt{\frac{A_{base}}{\pi^2}\left\{\pi - \theta_{VK}(x) + \frac{1}{2}\sin\left[2\theta_{VK}(x)\right]\right\}} \quad (26)$$

where

$$\theta_{VK}(x) = \cos^{-1}\left(\frac{2x-L}{L}\right),$$

A_{base} is the specified base area, and L is the specified total body length.

The Sears-Haack body assumes that the body is pointed at both ends, that the total volume is known, and that the total length of the body is given. In this test case only the front half of this body is of interest. The expression for the radius as a function of axial distance from the nose to the middle of this body is given by

$$r_{SH}(x) = \sqrt{\frac{4V}{\pi^2 L}\left\{\sin\left[\theta_{SH}(x)\right] - \frac{1}{3}\sin\left[3\theta_{SH}(x)\right]\right\}} \quad (27)$$

where

$$\theta_{SH}(x) = \cos^{-1}\left(\frac{x-L}{L}\right),$$

V is the specified volume, and L is the specified length for the front half of the body.

The hybrid gradient optimizer was used to determine computationally the body of revolution that minimizes wave drag at Mach = 10 and an altitude of 18 km. The MNIT was used for the flow field analyses. Initially, the body was specified to be a 10-meter-long, 15-degree angle right-circular cone. The design variables for this exercise were specified to be the radii of the body at 10 cross sections. Each design variable (the cross sectional radii) was allowed to vary from 0 to 10 meters. During the optimization process, the total volume of the body was constrained (with an equality constraint) not to change by more than 1.0 cubic meter from its initial value. The optimization process converged to the three-dimensional configuration shown in Figure 4. This 'bulged' conical body is called an *ogive*. Figure 5 displays plots of the profiles of the initial and final (optimized) bodies. The base area of the optimized body, and the total volume (fixed) were used to compute Von-Karman and Sears-Haack bodies. The profiles of these analytically optimal bodies of revolution are also plotted in Figure 5. Inspection of Figure 5 shows the computed optimized body to be in excellent agreement with the analytic bodies.

5-1-3 Verification Test 3: Specified Center of Pressure

In this exercise, an initial body was optimized in order to minimize drag at zero angle of attack using the hybrid gradient optimizer and the MNIT flow solver. The initial body is a right-circular cone with a spherical nose. The radius of curvature of the spherical nose measured 0.02 meters. The radius of the circular rear base area measured 0.1 meters. The length from the center of curvature of the nose axially to the rear plane measured 1.0 meter. The initial profile of the body was linear and tangent to the spherical nose.

The optimization problem was to find the optimal body that minimizes drag such that:

- the total length of the body was fixed,
- the base area and radius were fixed,
- the radius of curvature of the nose was fixed,
- and the aerodynamic center of pressure, $x_{c.p.}$, was specified (with an equality constraint) to be a distance of 0.572 meters from the center of curvature of the nose along the axis of symmetry within a tolerance of 0.01 meters.

The design variables were the radii of the 9 cross sections between the spherical nose and the base. The spherical nose was described by 10 fixed cross sections.

The optimization process converged to the body shown in Figure 6. The analytic solution to this problem has been previously determined[1] using Newton impact theory assumptions. The expression for the radius as a function of the non-dimensional axial distance (measured from the center of the spherical nose) for the analytically optimal body is given by

$$r_{cp=.572}\left(\frac{x}{L}\right) = \left\{\left(\frac{r_0}{L}\right)^{4/3} + \left[\left(\frac{r_{base}}{L}\right)^{4/3} - \left(\frac{r_0}{L}\right)^{4/3}\right]\frac{x}{L}\right\}^{3/4} \quad (28)$$

where r_0 is the specified radius of the spherical nose, r_{base} is the specified radius of the circular rear plane, and L is the specified total body length from the center of the spherical nose to the rear plane. Note that this expression is the optimal drag body for the case when $r_0/L = 0.02$, $r_{base}/L = 0.1$, and $x_{c.p.}/L = 0.572$ *only*.

The profiles of the initial, the optimized, and the analytically optimal body are plotted in Figure 7. This figure shows good agreement between the computed and the analytically optimum bodies.

Exploratory Case 1

In case 1, an initial body was optimized to minimize drag at zero angle of attack using the hybrid gradient optimizer and the MNIT flow solver. The initial shape was a right-circular cone. The shape of the body was optimized holding its total length and volume fixed. The geometry was described by six cross sectional planes with forty nodes on each plane. The total length of the vehicle was 10 meters. The body

was assumed to be traveling at Mach = 10 and at an altitude of 18 km.

All of the surface nodes on the first cross section move together radially and are controlled by one design variable. On the other five planes, all of the surface nodes have two degrees of freedom except for the 'seam' points whose x-coordinate is zero (the points on the vertical plane of symmetry). These 'seam' points were specified to move only vertically (in the y-direction) in their plane.

The optimization process was executed until convergence was reached. Figure 8 shows the final optimized body. The final configuration reduced drag by 77 percent over 75 design cycles, and called the flow solver 60,001 times. The execution took 4,282 seconds on a Cray C-90 supercomputer. Figure 9 displays the hybrid gradient optimizer convergence history for this case.

The initial geometry and the final optimized geometry were analyzed using the parabolized Navier-Stokes (PNS) flow solver[26]. The size of the computational grid was 240 circumferential cells, by 30 radial cells, by 200 longitudinal cells. The freestream temperature, density, and Mach number were specified to be 218 K, 0.1206 kg/m^3, and 10 respectively. A nonequilibrium gas model and Wilkes mixing rule were used in the PNS flow analysis. The total drag of the optimized body, as computed by the PNS flow field solver, was found to be 53% lower than the drag of the initial conical geometry.

The predominant characteristic of the optimized geometry is the deep channels that formed along the length of the body. It is interesting to note that star-shaped hypersonic projectile shapes have been studied experimentally in the past[27-28]. Russian researchers[29] determined that star-shaped bodies to be optimal (lowest drag) at high Mach numbers and at altitudes below 90 km.

Another characteristic of the optimal body shown in Figure 8 is the spiked nose. The conical shape of the nose was enforced because geometry of the first cross section was specified to be dependent upon one design variable: radial distance from the pole. In order to reduce drag, the optimizer could only reduce the inclination angle of these panels by reducing the radius of the first cross section, and add the resulting lost volume by increasing the size of the spikes on the aft body. Aerodynamic bodies that have spiked or needled noses have also been studied in the past[30-31]. The fact that the nose was driven by the optimizer to a slender spike rather than to a wide cone is interesting in light of these previous studies.

Exploratory Case 2

An initial body was optimized to maximize the lift-to-drag ratio at zero angle of attack using the MNIT flow field solver. For this case, the initial shape was specified to be a 15-degree right-circular cone, identical to the one in Exploratory Case 1. The shape of the body was optimized holding its total length and volume fixed. The geometry was described by six cross sectional planes on which there were forty nodes each. The total length of the vehicle was 10 meters. Every surface node was specified to vary radially on its cross sectional plane. There were 40 points on each of 6 cross sectional planes.

The hybrid gradient optimizer was executed until convergence was reached. Execution terminated after 40 design iterations that spent 1,458 CPU seconds on a Cray C-90 and required 19,961 objective function analyses. Figure 10 shows the final optimized body. The converged optimal lift-to-drag ratio corresponding to this geometry was L/D=1.29. This geometry is cambered and has ridges that have formed on its upper surface. The optimizer cambered the body so that a greater surface area on the underside faced the freestream so as to increase lift, and formed ridges on top of the body so that downward pressure was minimized. Yet the body still has an ogive appearance, which helps to reduce overall drag.

At this point (after the hybrid gradient optimizer had converged), the optimization method was changed to the hybrid genetic optimizer. The hybrid genetic optimizer continued to further reduce the cost function. The execution of the hybrid genetic optimizer was stopped after it had performed an additional 52 design cycles that required only an additional 748 objective function analyses. The final optimized geometry is shown in Figure 11. This 'optimized' geometry is obviously acrodynamically nonsensical, but an important conclusion can be drawn from this exercise. Figures 12 and 13 show the total convergence history for this case. Figure 12 shows reduction of the objective function plotted against design cycles. The important fact learned from this case is that the hybrid genetic optimization technique was able to significantly reduce the cost function after the more standard gradient method had converged and could go no further. Not only did the hybrid genetic optimizer reduce the cost function, but it did so while performing fewer objective function analyses. This is clearly seen in Figure 13, which shows the objective function reduction plotted against the total number of objective function analyses.

The reason why the hybrid genetic optimizer produced such a nonsensical-looking geometry is a

consequence of the flow field analysis method that was used. The MNIT flow solver is extremely simple and can yield inaccurate results for extremely complex and undulating geometries such as the one in Figure 11. The MNIT has no way to account for the viscous forces or the complex shock wave interactions that would occur in the flow field around such a geometry. Therefore, the geometry depicted in Figure 11 is certainly not optimal in light of a correctly modeled flow. But it demonstrates the ability of the hybrid genetic optimizer to significantly and rapidly improve on a 'converged' solution (according to conventional optimization techniques) that holds important implications.

Exploratory Case 3

In this case, an initial body was optimized to maximize the lift-to-drag ratio at zero angle of attack using the hybrid gradient optimizer and the MNIT flow solver. The initial shape is shown in Figure 14. The shape of the body was optimized holding its total length and volume fixed. The geometry was described by six cross sectional planes on which there were forty nodes each. The total length of the vehicle was 10 meters.

The geometry for this case was entirely described using beta-splines. Therefore, the design variables for this case were specified to be the locations of the beta-spline control vertices, and the actual geometry surface was generated by computing the beta-splines which depend on the control vertices. Only one half of the geometry was optimized, and the other half was 'mirrored' across the vertical plane of symmetry. The slope of the geometry was specified to be perpendicular to the plane of symmetry at the points 'on the seam' (points on the plane of symmetry).

Initially, the geometry was modeled using only one beta-spline on each half cross section. This modeling required 24 design variables for the entire geometry. The hybrid gradient optimizer was used during the program execution until convergence was reached. At this point (when convergence was reached) the converged geometry was redefined using two beta-splines on each half cross section, and the optimization process was restarted. This redefined geometrical model required 48 design variables for the entire geometry. Once again, when convergence was reached more degrees of freedom were added to the problem. This entire cycle was repeated twice more: first with 144 design variables, and finally with 240 design variables for the entire geometry.

Figure 15 shows the convergence history for the entire process. Figure 16a shows the initial geometry, and Figures 15b through 15e show the converged geometry for each stage of the procedure. The final lift-to-drag ratio was L/D=1.77, and the total optimization process required a total of 5,288 objective function analyses and a total of 141 CPU seconds on a Cray C-90.

The results from this case demonstrate that the use of beta-splines in geometry definition can lead to an improved design costing fewer objective function analyses. Because fewer design variables were used at first, the optimizer was able to quickly determine the general features of the optimal design. After progressively adding degrees of freedom to the geometry the final solution had a comparatively high L/D and was smooth and realistic-looking.

Exploratory Case 4

This case is exactly the same as Exploratory Case 3 except the hybrid genetic optimizer was used for the entire execution and the geometry was only remodeled once. The initial shape modeled by one beta-spline on each half cross section (24 design variables) is shown in Figure 14. This case was executed for 600 design cycles using the initial modeling (24 design variables). The converged geometry to this point required 15,287 objective function analyses and 2,660 CPU seconds on a Cray C-90 and is shown in Figure 17. After convergence was reached, the geometry was remodeled using two beta-splines on each half cross section (48 design variables) and the optimization process was continued. The execution was then finally stopped after an additional 129 design cycles requiring an additional 3,122 objective function analyses and an additional 2,355 CPU seconds. The final optimized geometry is shown in Figure 18. The convergence history for this case is displayed in Figure 19.

The final value for the lift-to-drag ratio was L/D = 2.4527. This value for L/D is higher than that of any other case in this study[16]. Interestingly, this case is modeled with the fewest number of geometric design variables (48 for the entire body, using beta-splines). This case shows again the ability of the hybrid genetic optimizer to effectively find optimal designs. It should be noted that the point at which more degrees of freedom were added to the problem and the point at which the entire execution was stopped were chosen by inspection of the convergence history. Unlike gradient methods, the hybrid (or any) genetic optimization technique does not generally exhibit smooth convergence characteristics. Rather they may show slow (or no) improvement on the design over many design cycles; then suddenly drastic improvement may occur. The implementation of a Nelder-Mead search direction calculation, as was done in the current hybrid genetic method, can significantly improve convergence

regularity. Even so there is no way to know if convergence has actually been reached, or if a large design improvement is only a few design cycles away, when using most non-gradient methods.

Figure 20 displays the decrease of the cost functions for this case and Exploratory Case 3 plotted together against the number of objective function analyses. This figure shows that the hybrid gradient optimizer initially reduces the cost function with fewer cost function analyses; however, the hybrid genetic optimizer quickly surpasses the gradient method and obtains designs that have a higher L/D than any design that the gradient method *ever* achieves.

Conclusions

Results from this study have indicated that hybrid non-gradient methods can be effectively applied to difficult engineering problems. A hybrid genetic algorithm that has been shown to achieve convergence rates comparable with more standard gradient techniques was developed. Furthermore, the hybrid genetic algorithm was able to explore design configurations that gradient methods would not. The hybrid genetic algorithm presented in this study was also able to simultaneously adhere to inequality and equality constraints.

When the hybrid genetic algorithm is applied to shape optimization in which the objective function depends on aerodynamic properties computed from a flow analysis, it is advantageous to describe the variable geometry using local control spline techniques. The use of beta-splines was shown to effectively lead to optimized geometry configurations when either technique was used.

Both the MNIT and the PNS flow field analysis methods can be used with either of the optimization techniques. It was found, however, that the PNS flow solver could be used in the optimization cycle only in restricted cases. During the course of the optimization procedure, intermediate design geometries that may cause areas of recirculation in the flow field need to be analyzed. The PNS flow solver will become unstable and fail during the flow analysis for such geometries. Therefore, additional geometrical constraints might be required to keep intermediate designs adequately smooth so that flow analyses will maintain stability. Alternatively, the PNS flow field solver can be used to simply verify the final results that are obtained using the MNIT flow analysis method, as was done in this study.

The MNIT flow solver was used extensively in this study. However, because of its simplicity the MNIT does not account for some of the important physical phenomena that actually occur in the flow field. As a result the optimizer may arrive at a design that in fact would be nonsensical if all of the physical phenomena were taken into consideration. The optimizer is not dependent on the flow solver that is utilized, but will search for an optimal design regardless of the accuracy of the procedures used to compute the objective function. For this reason it would be desirable to use a flow solver that accounts for the true physical phenomena in the flow field, such as a full Navier-Stokes flow field solver, with an optimizer that requires the fewest number of objective function analyses.

From the results in this study, it can be seen that the internal design update logic of the hybrid genetic optimization method is much more expensive in CPU time than that in the hybrid gradient method. However, the hybrid genetic algorithm evaluates the objective function many fewer times. Therefore if a very expensive flow solver, such as a full Navier-Stokes flow solver, were utilized then the hybrid genetic method might be superior as far as CPU expense is concerned. It has been seen that the hybrid genetic method can also arrive at designs that the gradient method can not, because the genetic algorithm is very effective at avoiding local minima. For these reasons hybrid genetic optimization techniques, such as the one presented in this study, may become widely used to solve difficult modern engineering problems.

Acknowledgements

This work was supported in part by the NASA Ames-University Consortium Agreement NCA2-769. The authors would like to express their appreciation for the advice and help provided by Dr. Isaiah Blankson of NASA HQ and Drs. Gordon Blom and Gregory Molvik of Boeing Co., and for the computing time on NASA Ames NAS Cray-C90 computer.

References

1. Strand, T., "Design of Missle Bodies for Minimum Drag at Very High Speeds - Thickness Ratio, Lift, and Center of Pressure Given," *Journal of the Aero/Space Sciences*, September 1959, pp. 568-570.
2. Shipilin, A.V., Optimal Shapes of Bodies with Attached Shock Waves," Mekhanika Zhidkosti i Gaza, Vol. 1, No. 5, 1966, pp 9-13.
3. Brown, L B., "Axisymmetric Bodies of Minimum Drag in Hypersonic Flow," University of Texas at Austin Report U-Tex-EMRL-TR-1016, July 1967.
4. Huang, H.Y., "Variational Approach to Conical Bodies Having Maximum Lift-to-Drag Ratio at Hypersonic Speeds," *Journal of Optimization*

Theory and Applications, Vol.2, No. 5, September, 1968, pp. 348-362.

5. Dulikravich, G.S., Buss, R.N., Strang, E.J. and Lee, S., "Aerodynamic Shape Optimization of Hypersonic Missiles," AIAA Paper 90-3073, *Proceedings of the AIAA 8th Applied Aerodynamics Conference*, Portland, OR, August 20-22, 1990.
6. Lewis, M.J. and McRonald, A.D., "Design of Hypersonic Waveriders for Aeroassisted Interplanetary Trajectories," *Journal of Spacecraft and Rockets*, Vol. 29, No. 5, September-October 1992, pp. 653-659.
7. Dulikravich, G.S. and Sheffer, S.G., "Aerodynamic Shape Optimization of Arbitrary Hypersonic Vehicles", Proc. of 3rd International Conference on Inverse Design Concepts and Optimization in Engineering Sciences (ICIDES-III), Editor: G.S. Dulikravich, Washington, D.C., Oct. 23-25, 1991, pp. 347-358.
8. Dulikravich, G.S. and Sheffer, S.G., "Aerodynamic Shape Optimization of Hypersonic Configurations Including Viscous Effects", AIAA 92-2635, AIAA 10th Applied Aerodynamics Conference, Palo Alto, CA, June 22-24, 1992.
9. Sheffer, S.G. and Dulikravich, G.S., "Constrained Optimization of Three-Dimensional Hypersonic Vehicle Configurations", AIAA Paper 93-0039, Thirty-first Aerospace Sciences Meeting and Exhibit, Reno, NV, January 11-14, 1993.
10. Large, E., "Nose Shape for Minimum Drag in Hypersonic Flow," *Journal of Aerospace Sciences*, January 1962, p. 98.
11. Maestrello, L. and Ting, L., "Optimum Shape of a Blunt Forebody in Hypersonic Flow," ICASE Report No. 89-51, NASA CR-181995, December 1989.
12. Thompson, R.A. and Hull, D.G., "Hypersonic Airfoils of Maximum Lift-to-Drag Ratio," University of Texas at Austin Report U-Tex-AMRL-TR-1009, September 1969.
13. Wilhite, A.W., "Optimum Wing Sizing of a Single-Stage-to-Orbit Vehicle," *Journal of Spacecraft*, Vol. 20, No. 2, March-April 1983, pp. 115-121.
14. O'Neill, M.K. and Lewis, M.J., "Optimized Scramjet Integration on a Waverider," *Journal of Aircraft*, Vol. 29, No. 6, November-December 1992, pp. 1114-1121.
15. Haftka, R.T. and Gürdal, Z., Elements of Structural Optimization, 3rd ed., Kluwer Academic Publishers, Boston, MA, 1992.
16. Foster, N.F., "Shape Optimization Using Constrained Genetic Evolution and Gradient Search Algorithms," M.Sc. Thesis, Dept. of Aerospace Engineering, The Pennsylvania State University, University Park, PA, August 1995.
17. Misegades, K.P., "Optimization of Multi-Element Airfoils," Von Karmon Institute for Fluid Dynamics, Belgium, Project Report 1980-5, June 1980.
18. Goldberg, D.E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989.
19. Dulikravich, G.S., "Aerodynamic Shape Design and Optimization: Status and Trends", *AIAA Journal of Aircraft*, Vol. 29, No. 5, Nov./Dec. 1992, pp. 1020-1026.
20. Dulikravich, G.S., "Shape Inverse Design and Optimization for Three-Dimensional Aerodynamics", AIAA invited paper 95-0695, AIAA Aerospace Sciences Meeting, Reno, NV, January 9-12, 1995.
21. Crispin, Y., "Aircraft Conceptual Optimization Using Simulated Evolution", AIAA paper 94-0092, Reno, NV, January 10-13, 1994.
22. Nelder, J.A. and Mead, R., "A Simplex Method for Function Minimization," *Computer Journal*, Vol. 7, 1965, pp. 308-313.
23. Barsky, B.A., Computer Graphics and Geometric Modeling Using Beta-Splines, Springer-Verlag, Berlin, Germany, 1988.
24. Geiger, R.E., "Experimental Lift and Drag of a Series of Glide Configurations at Mach Numbers 12.6 and 17.5," *Journal of Aerospace and Sciences*, April 1962, pp.410-419.
25. Ashley, H. and Landahl, M., Aerodynamics of Wings and Bodies, Addison-Wesley Publishing Company Inc., MA, 1965.
26. Molvik, G.A., "A Computational Model for the Prediction of Hypersonic, Reacting Flows", A Doctorial Thesis in Mechanical Engineering, The Pennsylvania State University, 1989.
27. Vedernikov, Yu.A., Gonor, A.L., Zubin, M.A., and Ostapenko, N.A., "Aerodynamic Characteristics of Star-Shaped Bodies at M= 3-5," Izv. Akad. Nauk SSSR, Mekh. Zhidk. Gaza, No. 4, 1981, pg. 88.
28. Gusarov, A.A., Dvoretskii, V.M., Ivanov, M.Ya., Levin, V.A., and Chernyl, G.G., "Theoretical and Experimental Investigation of the Aerodynamic Characteristics of Three-Dimensional Bodies," Izv. Akad. Nauk SSSR, Mekh. Zhidk. Gaza, No. 3, 1979, pg. 97.
29. Bunimovich, A.I., and Kuz'menko, V.I, "Aerodynamic and Thermodynamic Characteristics of Three-Dimensional Bodies in a Rarefied Gas," Izv. Akad. Nauk SSSR, Mekh. Zhidk. Gaza, No. 4, July-August, 1982, pp. 181-183.