Acceleration of Differential Evolution for Aerodynamic Design

by

Tim Rogalsky

B.R.S. (Mennonite Brethren Bible College) 1991
B.Sc. (University of Manitoba) 1996
M.Sc. (University of Manitoba) 1998

A dissertation submitted to
the Faculty of Graduate Studies
in partial satisfaction of the requirements for the degree of

Doctor of Philosophy
in
Mathematics

at the
UNIVERSITY OF MANITOBA

Acceleration of Differential Evolution for Aerodynamic Design

by

Tim Rogalsky

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION

Acceleration of Differential Evolution for Aerodynamic Design

BY

Tim Rogalsky

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirement of the degree

Of

DOCTOR OF PHILOSOPHY

Abstract

Acceleration of Differential Evolution for Aerodynamic Design

by

Tim Rogalsky

Ph.D. in Mathematics, University of Manitoba

It has been demonstrated that Differential Evolution (DE) is a robust optimizer for aerodynamic design of fan blade profiles, but it can require 50,000 flow calculations to converge to a solution. This is feasible with only the simplest aerodynamic model. Accelerated convergence is required for the design algorithm to be more useful. This thesis presents, as benchmarks, convergence rates for three design cases using Bezier parameterization of airfoils and optimizing with DE. These benchmark rates are accelerated in two ways. First, an improved solution space is provided by Bezier-PARSEC airfoil parameterization. To compare their representation abilities, Bezier and Bezier-PARSEC parameterizations are used to reproduce 63 airfoils. Second, DE is modified in three different ways to provide improved convergence characteristics with the new parameterization: 1) A new selection operator is introduced, variable birthrate, which can bias the search toward the most promising regions of the solution space, 2) DE is hybridized with Downhill Simplex, a local search method, and 3) DE is accelerated by an algorithm modeling the biological immune system. The most successful strategy is Hybridized Immune Accelerated DE (HIADE). Using the BP 3333 parameterization, it converges within 10,000 flow calculations, four to ten times faster than the benchmarks.

Acknowledgements

This work has been supported financially by the National Science and Engineering Research Council of Canada (NSERC), Manitoba Hydro, University of Manitoba, and Canadian Mennonite University.

Thank-you to my advisor, Robert Derksen, for the guidance, suggestions, and many enjoyable chats over Tim Horton's coffee; To the committee members, Tom Berry and Abba Gumel; To Jouni Lampinen for agreeing to examine the thesis; To Trudy Matechuck, who performed early experiments with the BP 3424 and 3434 parameterizations.

Finally, a big thank-you to my family for bearing with me these last months: To Deb for being a single parent for too long, and not complaining too much; To Nathan, for the baseball and basketball breaks; To Lois for the constant reminders to "Finish the thesis so we can have a holiday"; To Mark for dangling the Big Foots carrot; To Philip for the delightful "Hi daddy, hi daddy" each time I emerged from the study.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Aerodynamic Design

Aerodynamic design can be thought of as a search for an optimal shape. The meaning of "optimal" depends on the application. Some require high lift – for example an aircraft that must carry an extremely heavy load. Others might require low drag – for maximum fuel efficiency. The ultimate goal of the research described here is a more efficient fan. An efficient fan converts most of the energy required to run it into actual airflow. In many current fans, much of the energy is used to overcome the drag on the blades, or is converted into heat. A better design would improve energy conservation (and decrease operational cost) without sacrificing capability.

Aerodynamic design has its roots in wind-tunnel experiments. Physical models would be built, tested, refined, and re-tested repeatedly until satisfactory results were obtained. In the 1930's and 1940's, systematic investigations of this type were performed by NACA (see for example Chapter 7 in Abbot and von Doenhoff, 1959). Of course, this process is arduous and expensive. It is desirable to streamline as much of it as possible.

One helpful insight is that the performance of an airfoil depends greatly on the distribution of pressure (or velocity) along its surface. Stratford (1959a, 1959b), for example, developed a pressure distribution that, theoretically, would achieve any specified pressure rise in the shortest possible distance and with the least possible dissipation of energy. Thus it became possible to pose what is known as an inverse design problem: Given a pre-specified pressure distribution, obtain the airfoil for which that distribution is realized.

In the 1950's, Richard Eppler (1957) began to solve inverse design problems using conformal transformations, ultimately developing a computer program to do so (Eppler, 1990). Building on Stratford's work, Liebeck and Ormsbee (1970) designed a family of pressure distributions to provide maximum possible lift in an incompressible flow. Liebeck (1973, 1978, 1990) later extended this work, designing airfoils for aircraft, racing cars, and even a model pterodactyl!

Another approach to inverse design is to treat it as an optimization problem. An objective function calculates the pressure distribution around a given blade, measures the difference from the target distribution, and defines that to be the cost of the blade. An optimizer is asked to find the shape that minimizes this "cost." For example, Venkataraman (1996b) used a generalized reduced gradient method to solve several inverse problems.

In that direction lies our earlier work (Rogalsky, 1998). The object of design is the cross-sectional shape of the fan blade. Software was designed, implementing a new design algorithm. Differential Evolution (DE) (Storn & Price, 1995) was demonstrated to be an effective optimizer for the problem. Several test cases were performed, in which DE was able to search out near-optimal solutions, even when other commonly used optimizers failed (Rogalsky et. al., 2000).

The focus of the current thesis is to accelerate the optimization component of the aerodynamic design method. Although DE was robust, it commonly required the evaluation of 50,000 shapes, a computational nightmare in fluid dynamics. If the number of function evaluations can be reduced significantly, it should be possible to incorporate a more sophisticated flow solver, making the end product more useful.

The DOS-based software developed earlier (Rogalsky, 1998) was made more user-friendly by adding a windows interface. Dialog boxes are used for input of DE and acceleration parameters. Run-time convergence information is displayed on-screen, including the shapes of the current blade and pressure distribution. This enabled insight to be gained into the nature of convergence, and sparked some of the acceleration ideas. The result is FanOpt v. 3.5, and is included, with documentation, as Appendix A on the accompanying CD.

The remainder of this introductory chapter will briefly review the FanOpt aerodynamic design algorithm, and then introduce ideas for acceleration. There are three components to the design method: 1) optimization with DE (section 1.2), 2) representation of the geometric shape as a real-valued vector (section 1.3), and 3) simulation of the flowfield around the shape (section 1.4). The cost function to be minimized by DE is described in section 1.5. Finally, in section 1.6, the specific focus of this thesis will be introduced: acceleration of DE for aerodynamic design of fan blades.

1.2 Optimization by Differential Evolution

Many current design methods use local optimizers in the search. Designers start with a shape already being used – but in some way inadequate – for the required application. Minor modifications are made by the optimizer, and usually these are accepted only if they improve performance. After a small number of iterations, the improved design is accepted. The result, of course, is a design with only minor improvements.

It is preferable to perform a more general search – with potential to find a radically new design, unbiased by preconceived conceptions of what works best. Such a search could be used not only to improve a design already in use, but also to design "from scratch" a shape useful for a new application. What is required, then, is a fast algorithm for global optimization. And that is where Differential Evolution (DE) comes in real handy!

DE is a member of a broader class of algorithms called Evolutionary Algorithms (EAs). The most common of these is the Genetic Algorithm (GA). GAs operate on bit-strings, suitable for discrete or integer optimization problems. They interpret the objective function value at a point as a measure of that point's fitness as an optimum. Then, guided by the principle of survival of the fittest, an initial population is transformed into a solution bit-string through repeated cycles of mutation, recombination, and selection. Sporadic attempts to incorporate these principles in optimization have been made since the 1960's (see a review in Chapter 4 of Goldberg, 1989). It was the work of Holland (1975), though, that established GAs on a sound theoretical basis.

GAs have proven to be very effective at finding the global optimum in complicated, multidimensional landscapes. Among other researchers, aerodynamic designers are turning to GAs in steadily increasing numbers (Chan, 1998; Obayashi & Tsukahara, 1996; Takahashi et. al., 1999; Vicini & Quagliarella, 1999, Perez et. al., 2000). Shapes such as two-dimensional airfoils or three-dimensional wings are encoded as long bit strings using various computer graphics techniques. The GAs are then used to find the shape that minimizes some aerodynamic fitness function, such as the drag-to-lift ratio, or the deviation from an aerodynamic objective (a desired pressure distribution, for example.)

Another type of EA, the Evolution Strategy (ES), uses the same principles, but operates on real-valued vectors. ESs are thus better suited for continuous parameter optimization problems. They were introduced in the 1960's by three students at the Technical University of Berlin – Rechenberg, Bienert, and Schwefel. The three were looking for a decision-making tool for an experimental aerodynamic design problem - minimizing drag for a flexible, slender, three-dimensional body in a wind tunnel. After some commonly used optimizers failed, Rechenberg (1965) proposed the idea of random decision-making. Using a population of one, random mutations of the design variables determined the next shape to be tested. If the new design had lower drag, it survived to the next generation. Bienert (1967) actually constructed a robot that could perform the decisions and actions automatically. Rechenberg (1973) later increased the population size, and the theory was further developed by Schwefel (1975a, 1975b).

DE is an ES that grew out of Ken Price's attempts to solve the Chebychev polynomial fitting problem that had been posed to him by Rainer Storn (Storn & Price, 1995). DE is emerging as one of the most impressive EAs. Storn and Price (1997b) have demonstrated, using an extensive testbed of objective functions, that DE converges faster and with more certainty than many other acclaimed global optimization methods, including several ESs.

The list of applications for which DE has been effective is long. It includes optimization of bioprocesses such as the growth of penicillin (Balsa-Canto, 1998), layout design of mass transit signaling systems (Chang and Du, 1999), allocation of processors in a parallel architecture (Rae and Parameswaran, 1998), solution of scheduling problems (Rüttgers, 1997), redundancy optimization for MPEG (Storn, 1995), optimal control of Differential-Algebraic Systems (Wang and Chiou, 1997), and characterizing a structure when one can only observe the X-ray scattering pattern (Wormington et. al., 1999). References to many others can be found in the DE bibliography maintained by Jouni Lampinen (2001).

The crucial difference between DE and other ESs lies in mutation. Traditional ESs (and GAs) use predetermined probability distribution functions to perturb vectors. This leaves them unable to adapt the perturbation magnitude to the topology of the objective function. DE, on the other hand, uses the difference of two randomly chosen vectors to perturb another vector. The magnitude is thus automatically appropriate to the given landscape, and the search is less random, being dictated by the shape of the given objective function. This property of DE is known as self-organization. Ultimately, it results in better convergence properties as the algorithm nears the global minimum.

Since DE lies at the heart of this thesis, the essential elements will be described here. For a more detailed summary, see Price (1999). The overall structure of the DE algorithm resembles that of most other population-based searches. Two arrays are maintained, each of which holds a population of N, D-dimensional, real-valued vectors. The primary array holds the current population while the secondary array accumulates vectors that are selected for the next generation. The population evolves toward a solution by a process of natural selection, or survival of the fittest. (See Figure 1.1.)

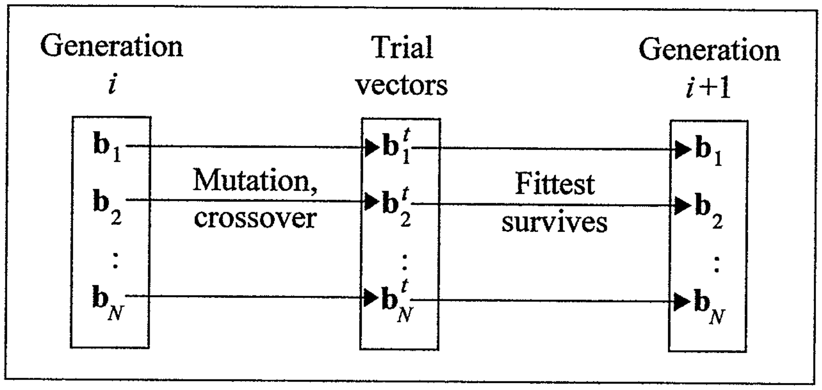


Figure 1.1 *Overall structure of Differential Evolution.*

Each vector $\mathbf{b}_a$ in the primary array is a "parent", which generates a trial vector $\mathbf{b}_a^t$ (its "child") through mutation and recombination (described below). The fitness of any given vector $\mathbf{b}_a$ is determined by its cost, $C(\mathbf{b}_a)$, where $C(\mathbf{x})$ is the objective function to be minimized. The cost of each parent is compared to that of its child, and the fittest vector (the one with the smaller objective function value) survives to the next generation, so that the vector $\mathbf{b}_a^{<i+1>}$ in generation $i + 1$ is either $\mathbf{b}_a$ from generation i, or its child, $\mathbf{b}_a^t$:

$$\mathbf{b}_a^{<i+1>} = \begin{cases} \mathbf{b}_a & \text{if } C(\mathbf{b}_a) \le C(\mathbf{b}_a^t) \\ \mathbf{b}_a^t & \text{if } C(\mathbf{b}_a^t) < C(\mathbf{b}_a) \end{cases}. \tag{1.1}$$

Mutation is an operation that makes small random alterations to one or more parameters of an existing population vector. Mutation is crucial for maintaining diversity in a population, and is typically performed by perturbation. DE uses the population itself as a convenient source of appropriately scaled perturbations. Each pair of vectors ($\mathbf{b}_c$, $\mathbf{b}_d$)

in the primary array (generation *i*) defines a vector differential, $\mathbf{b}_c - \mathbf{b}_d$. When these two vectors are chosen randomly, their weighted difference can be used to perturb another vector in the primary array, $\mathbf{b}_a$:

$$\mathbf{b}'_a = \mathbf{b}_a + F(\mathbf{b}_c - \mathbf{b}_d), \tag{1.2}$$

where $\mathbf{b}'_a$ is a mutation of $\mathbf{b}_a$. The weight, *F*, is a user-supplied constant. The optimal value of *F* for most functions lies in the range $0.4 \le F \le 1.0$.

An effective variation of this scheme involves keeping track of the best vector so far, $\mathbf{b}^*$. This can be combined with $\mathbf{b}_a$ and then perturbed, yielding

$$\mathbf{b}'_a = \mathbf{b}_a + F(\mathbf{b}^* - \mathbf{b}_a) + F(\mathbf{b}_c - \mathbf{b}_d). \tag{1.3}$$

In this scheme – known in the DE community as DE/rand-to-best/1 and used throughout this thesis – the most successful member of a population influences all trial vectors.

Recombination, or crossover, provides an alternative and complementary means of creating viable vectors. Designed to resemble the natural process by which a child inherits DNA from its parents, new parameter combinations are built from the components of existing vectors. This efficiently shuffles information about successful parameter combinations, enabling the search for an optimum to focus on the most promising areas of the solution space.

Each primary array vector $\mathbf{b}_a$ is targeted for recombination with $\mathbf{b}'_a$ (the mutated vector in (1.2) or (1.3)) to produce a trial vector, $\mathbf{b}^t_a$. Thus the trial vector is the child of two parents - $\mathbf{b}_a$, the primary array vector against which it must compete, and $\mathbf{b}'_a$, which is itself a random mutation of $\mathbf{b}_a$. DE can use two types of crossover - binary and exponential. When combined with the mutation operation in (1.3), the strategy is labeled DE/rand-to-best/1/bin or DE/rand-to-best/1/exp, respectively. The amount of information (DNA) shared in recombination is determined by the crossover constant *CR*, where $0 \le CR \le 1$.

In exponential crossover, a starting parameter, *p*, is selected at random. *CR* is compared to a uniformly distributed random number from within the interval [0,1). Subsequent trial vector parameters are chosen from $\mathbf{b}'_a$ until the random number generator produces a value larger than *CR* (or until all *D* parameters have been determined). The remaining parameters then come from the primary array vector. Thus, if *r* random numbers were generated before one of them was greater than *CR*, the trial vector parameters, $\left(\mathbf{b}^t_a\right)_j$, $1 \le j \le D$, are determined by

$$\left(\mathbf{b}^t_a\right)_j = \begin{cases} \left(\mathbf{b}'_a\right)_j, & p \le j < p+r \\ \left(\mathbf{b}_a\right)_j, & \text{otherwise} \end{cases} \tag{1.4}$$

for values of *r* smaller than $D-p$, and

$$\left(\mathbf{b}^t_a\right)_j = \begin{cases} \left(\mathbf{b}'_a\right)_j, & p \le j \le D \text{ and } 1 \le j < p+r-D \\ \left(\mathbf{b}_a\right)_j, & \text{otherwise} \end{cases} \tag{1.5}$$

for values of *r* larger than $D-p$.

In binary crossover, the random experiment is performed for each parameter. If the random number is smaller than *CR*, the trial vector parameter is chosen from $\mathbf{b}'_a$, otherwise it comes from $\mathbf{b}_a$. Thus, for each parameter, *j*, in the trial vector, a random number r_j is chosen, where $0 \le r_j < 1$. The trial vector parameters, $\left(\mathbf{b}_a^t\right)_j$, $1 \le j \le D$, are then determined by

$$\left(\mathbf{b}_a^t\right)_j = \begin{cases} \left(\mathbf{b}_a'\right)_j, & \text{if } r_j < CR \\ \left(\mathbf{b}_a\right)_j, & \text{otherwise} \end{cases} \tag{1.6}$$

In both types of crossover, exponential and binary , when $CR = 1$ every trial vector parameter comes from $\mathbf{b}_a'$, making the trial vector $\mathbf{b}_a^t$ an exact replica of the random mutation of $\mathbf{b}_a$.

Once new trial solutions have been generated, selection determines which among them will survive into the next generation. Each child $\mathbf{b}_a^t$ is pitted against its parent $\mathbf{b}_a$ in the primary array. Only the fitter of the two is then allowed to advance into the next generation.

In all, just three parameters control evolution: the population size N, the weight F applied to the differential in mutation, and the constant CR that mediates the crossover operation. DE has not been patented in the hopes that scientists around the world will develop it further. It has been coded in a variety of languages, including C, C++, Matlab and Java, and can be downloaded at no expense from Rainer Storn's DE webpage http://www.icsi.berkeley.edu/~storn/code.html.

1.3 Geometric Representation

DE operates on real-valued vectors, not on shapes. So one task of the design algorithm is to encode the geometry of the airfoil. In Rogalsky (1998), a new method of airfoil representation was developed using Bezier curve parameters. A brief review of the parameterization literature and a summary of the Bezier method follow.

Many methods have been used for geometric representation of airfoils. They include the inverse Theodorsen transformation (Theodorsen and Garrick, 1933), linear combination of basis shapes (Vanderplaats et. al., 1975), basis functions (Hicks and Henne, 1977), orthogonal shape functions (Chang et. al., 1995), Legendre polynomials (Coiro & Nioclosi, 1995), the extended Joukowski transformation (Jones, 1990), and PARSEC parameterization (Sobieczky, 1998, 1999). Many of these have proven unsuitable for aerodynamic optimization, being susceptible to wild oscillations, and requiring many parameters (Burgreen et al., 1992; Venkataraman, 1995b). Others cause slow convergence when used with EAs (Oyama, 1999). Bezier polynomials on the other hand, are proving to be quite useful (Venkataraman, 1995a).

P. Bezier, of the French firm Regie Renault, pioneered the use of computer modeling of surfaces in automobile design. His UNISURF system, initiated in 1962 and used by designers since 1972, has been applied to define the outer panels of several cars marketed by Renault (Bezier, 1972, 1974). The foundations of Bezier curves, however, go back much further. In 1926, S. Bernstein presented a constructive proof of the Weierstrass approximation theorem (Davis, 1963), using functions that have become known as Bernstein polynomials. Bezier curves have a very similar form, and are sometimes referred to as Bezier-Bernstein polynomials. An *n*th order Bezier curve is defined parametrically using $n+1$ two-dimensional control points.

In one of the first examples of their use in aerodynamic design, Birckelbaw (1989) used two 44th order Bezier curves to define an airfoil. The 180 control point variables were used as the optimization parameters. Venkataraman (1996a) used four Bezier curves to define an airfoil – two each for the top and bottom surfaces – reducing the number of design variables to 19. Aerodynamic shapes other than airfoils have also been parameterized. Burgreen et al. (1992), for example, represented the surface of an internal-external nozzle with Bezier curves in place of grid points. This reduced the number of design variables from 47 to six, and the CPU time by a factor of almost four.

The airfoil parameterization developed in Rogalsky (1998) is an enhancement of Venkataraman's method. Two Bezier curves are joined end-to-end to form the camber profile (the mean-line running down the center of the airfoil). Two are joined to form the half thickness profile (measured perpendicularly above and below the camber). See Figure 1.2. The curves are scaled to represent an airfoil with unit chordlength (the distance from the nose to the trailing edge). Camber-thickness definition of airfoils dates back to the 1930s when it was discovered that several effective wing sections had nearly the same thickness distribution when the mean line was straightened (Abbot and von Doenhoff, 1959). That is, the aerodynamic properties of an airfoil are more directly dependent on camber and thickness than on upper and lower surface shapes.

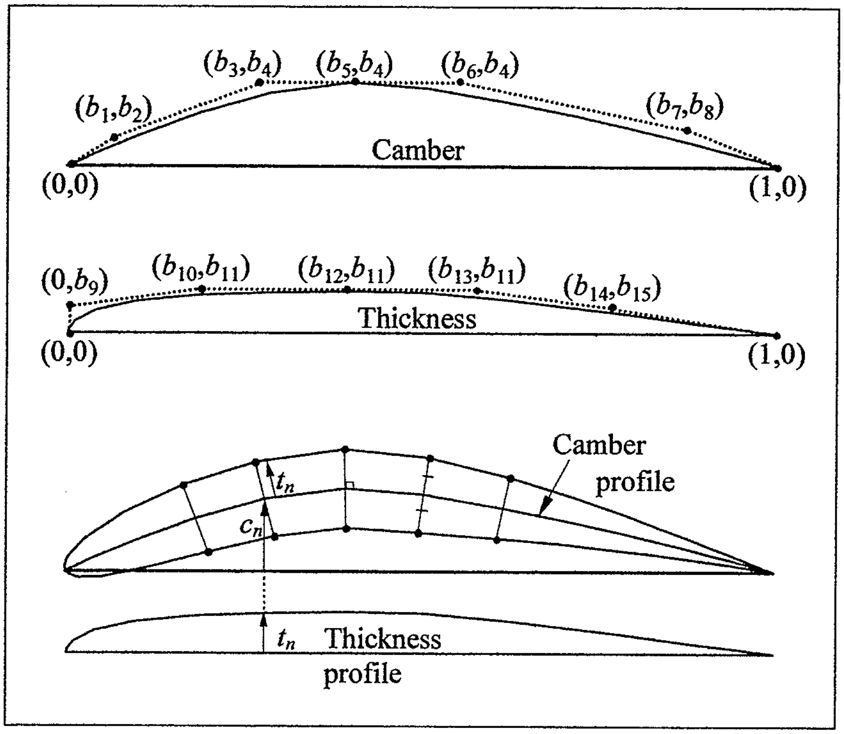


Figure 1.2 *Bezier parameterization of an airfoil. Fifteen Bezier parameters define the camber and half-thickness profiles. The half-thickness is measured perpendicular to the camber, forming the airfoil shape.*

1.4 Aerodynamic Calculations

The aerodynamic calculations represent by far the most computationally intensive component of the design algorithm. In fact, it is only in the last few years that computer power has become sufficient to solve inverse problems in reasonable time. Even now, only multi-million dollar super parallel computers are able to solve this problem reliably. A full flow field simulation for a single airfoil can take over an hour on a high end PC.

The two-dimensional flow field around a given object is most accurately modeled by the Navier-Stokes equations,

$$\begin{cases} u\dfrac{\partial u}{\partial x}+v\dfrac{\partial u}{\partial y}=-\dfrac{1}{\rho}\dfrac{\partial p}{\partial x}+\dfrac{\mu}{\rho}\nabla^2 u \\ u\dfrac{\partial v}{\partial x}+v\dfrac{\partial v}{\partial y}=-\dfrac{1}{\rho}\dfrac{\partial p}{\partial y}+\dfrac{\mu}{\rho}\nabla^2 v \end{cases}, \tag{1.7}$$

where the vector field $\langle u(x,y),v(x,y)\rangle$ represents the velocity of the flow at position (x,y), μ is the viscosity of the fluid, ρ is the mass density of the fluid, and p is the local static pressure. The nonlinearity in this system of partial differential equations makes it extremely difficult to solve. Existence and Uniqueness Theorems remain an open problem, more than 100 years after the development of the model. Even numerical simulation is very expensive computationally. Lombardi et al (2000) solve numerically the Navier-Stokes equations for flow around a NACA 0012 airfoil. The computational effort require 70 – 150 minutes on a desktop PC, depending on turbulence model. At this rate, a design problem requiring 50,000 flow simulations (such as the one discussed by Rogalsky et. al., 2000) would take at least six years to complete.

Fortunately, there are simplified equations that can be solved in significantly less time. Although the results are less detailed – and in some cases less reliable – they are adequate for most common situations. More significantly, though, these simplifications allow research to be performed using a common desktop PC. Design problems can be solved in about twenty minutes for which the optimizer has searched over 50,000 different shapes.

The simplification made here is to assume fully potential flow, i.e. zero viscosity. Viscosity can be thought of as the friction force that acts between fluid particles. In a turbulent flow field, when fluid particles are interacting with each other in many directions, viscous effects can be quite high. However, for flow over an aerodynamic body such as a fan blade, viscous effects are almost negligible, as long as the flow remains "attached" to the blade. (In attached flow, the air particles move along a path – called a streamline – that is nearly parallel to the solid body. Particles do not interact with each other so much as with the body itself. Contrast this with the turbulence in a region of separation – think about being passed in a snowstorm by a large truck.)

Flow solvers that assume potential flow are known as panel methods. The shape is approximated by a set of line segments called panels, and a numerical scheme is used to compute the flow solution. Despite the simplified model, panel methods are useful, and are still being applied in many situations (Pfeiffer, 1990).

The origins of the panel method can be found in classical mathematics. Kellog (1929) wrote a comprehensive book about potential theory. Since potential flow is incompressible and inviscid, the Navier-Stokes equations (1.7) can be simplified to Laplace's equation for the velocity potential,

$$\nabla^2 \psi = 0, \text{ where } u = \frac{\partial \psi}{\partial y},\ v = \frac{\partial \psi}{\partial x}. \tag{1.8}$$

Integrating an elementary solution over the body surface, Kellog developed an integral equation that represents the flow past a body immersed in a uniform stream. Panel methods approximate these integrals by discretizing the curves into panels, and then integrating numerically.

The specific inviscid potential flow model used here is Martensen's (1959) surface vorticity panel method (described in Lewis, 1991). Assuming fully attached flow, the boundary layer around an airfoil is approximated as an infinitesimally thin vorticity sheet. The inviscid surface velocity can then be found in terms of the vorticity strength on the boundary (equation (1.10) below).

Consider a small vorticity element $\gamma(s)ds$, where $\gamma(s)$ is defined as the vorticity strength per unit length at point *s*. Since the thickness of the element (normal to the surface) is infinitesimal, the circulation around it is just $(v_s - v_i)ds$, where v_s and v_i are the fluid velocities just outside and inside the sheet. This can be equated to the total amount of vorticity enclosed by the contour. That is,

$$(v_s - v_i)ds = \gamma(s)ds\,. \tag{1.9}$$

Since the no-slip condition on the body surface requires that $v_i = 0$, we have

$$v_s = \gamma(s)\,. \tag{1.10}$$

The body surface is represented discretely by a finite number of short, straight panels. Martensen’s boundary integral equation for two-dimensional flow (Lewis, 1991) relates the vortex strength at any given point to the vortex strengths at all other points on the surface. This integral is then approximated numerically. The resulting linear system of *n* equations is solved for the vorticities $\gamma(s)$ on each of the *n* panels. The vorticity on any given panel is then exactly equal to the inviscid velocity along that panel.

Jacob & Riegels (1963) first successfully implemented Martensen's method on a digital computer. An analysis of an airfoil with 36 elements took fifteen minutes to execute, a remarkable feat for that time. Wilkinson (1967) identified and resolved many modeling and computational obstacles and extended his work to mixed-flow turbomachinery cascades (Wilkinson, 1969). An excellent summary of vorticity methods is given by Sarpkaya (1989).

In a turbomachinery cascade – such as the axial flow fan – adjacent surfaces both influence the flow through any two blades. The cascade is thus modeled as an infinite rectilinear array of airfoils, set at equal pitch interval t parallel to the y-axis, and with equal stagger angle λ. (See Figure 1.3.) The flow enters the cascade with inlet velocity W_1 at inlet angle β_1; and exits with outlet velocity W_2 at outlet angle β_2. The blade spacing, t and λ, play a key role in the fan's overall performance (Rogalsky et al, 1999). These parameters are thus included in the vector **b** encoding the fan's geometry.

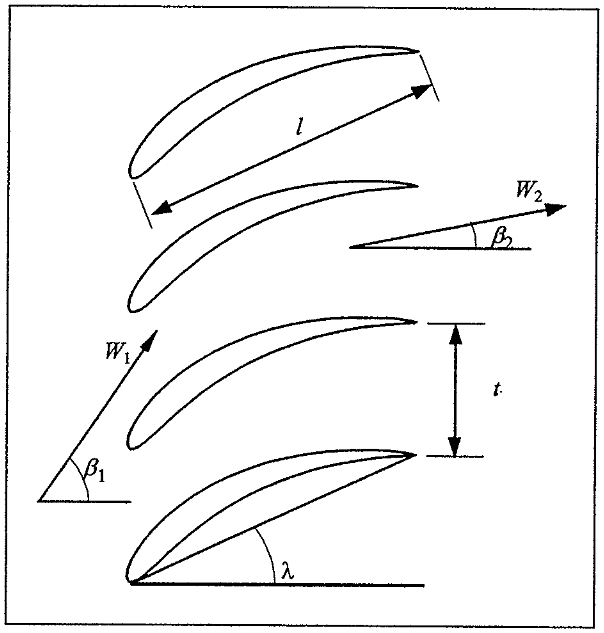


Figure 1.3 *Turbomachinery cascade geometry and flow velocities.*

Of course, the ultimate goal of the current research is to incorporate a more realistic flow solver into the design method. For example, the Keller Box module (Keller, 1975) assumes nonzero viscosity only in a boundary layer along the surface. Then the Navier-Stokes equations (1.7) can be simplified to the boundary layer equation,

$$u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}-\frac{\mu}{\rho}\frac{\partial^2 u}{\partial y^2}=-\frac{1}{\rho}\frac{dp}{dx}, \tag{1.11}$$

where μ is the viscosity of the fluid, ρ is the mass density of the fluid, and p is the local static pressure. A panel method is used to calculate the inviscid velocity distribution (just outside the boundary layer). Finite difference equations are used to solve (1.11) numerically throughout the boundary layer. Newton's method is used to deal with the nonlinearity of (1.11), normally requiring ten iterations.

A boundary layer method such as this would enable approximation of drag and location of any regions of separation, improving the confidence in any resultant designs. However, the additional complexity could easily add several orders of magnitude to the computational requirements. The inviscid model solves an $n \times n$ linear system of equations, where n (the number of panels) is typically 60-100. Boundary layer methods typically use a grid with n points (normal to the surface) in the boundary layer on each panel. The finite differencing then results in an $n \times n$ nonlinear system at each panel, and Newton's method requires the solution of ten of these each time. Thus, one $n \times n$ system is solved in the panel method, but $10n^2$ $n \times n$ systems are solved in the boundary layer method. If n is 100, the result is an increase in computational requirements by a factor of 10^3. If 50,000 panel method calculations take twenty minutes, then we would expect 50,000 boundary layer calculations to take fourteen days. Before incorporating such a module, we need to ensure feasibility. Thus, the immediate focus of this thesis is to reduce the number of flowfield simulations required to arrive at a solution.

1.5 Cost Function

As a fan rotates its blades through the air, the velocity of the air varies around the surface of each blade. The variation of velocity produces a variation of air pressure near the surface of the blade. This is usually measured in terms of the distribution of the pressure coefficient, c_p, along the blade. The performance of any fan is directly related to this so-called pressure distribution. In inverse design, the cost (or objective) function evaluates the proximity of a fan's performance to some target pressure distribution.

Liebeck (1973) designed the pressure distribution shown in Figure 1.4 for maximum possible lift in an incompressible flow. A few words of explanation are in order. A zero pressure coefficient represents the normal pressure of the air. A negative pressure coefficient represents a point of low pressure. Low-pressure regions induce lift on an aircraft wing and suction on a fan blade. By convention, this suction surface is defined to be the upper surface of the blade. For that reason the direction of the y-axis is reversed. In the Liebeck pressure distribution shown, the pressure along the upper surface rapidly drops along the leading 25% of the blade. It then uses the Stratford (1959a) distribution along the pressure recovery region (the trailing portion of the upper surface), which is designed to avoid separation (thereby avoiding high drag) by a small margin. In principle, this recovers the maximum possible pressure over that distance.

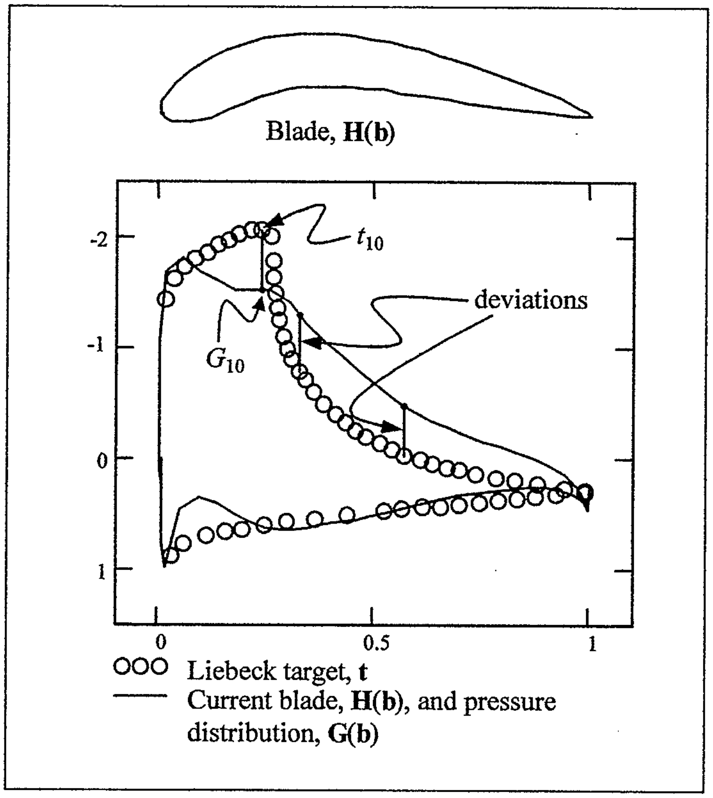


Figure 1.4 *Sample target – a Liebeck pressure distribution. Any member of the population,* **b**, *represents a fan blade shape and spacing. The cost of* **b** *is the deviation of its corresponding pressure distribution from the Liebeck target.*

The aerodynamic goal can thus be quantified as a discrete target pressure distribution **t**, together with an outlet angle, β_2. For any given encoded vector **b**, the decoded blade **H** and blade spacing are passed to the flow solver. At each x-coordinate of the target, the pressure coefficient is found and stored as **G**. The deviations between pressure coefficients in **t** and **G** (see Figure 1.4) form the first components of the error vector. The last component is the deviation, in radians, of the actual from the target outlet angle. The cost is defined to be the length of that error vector, i.e. the ℓ_2-norm of those differences,

$$C(\mathbf{b}) = \sqrt{\sum (G_i - t_i)^2 + \left(\beta_2{}^{\text{target}} - \beta_2{}^{\text{actual}}\right)^2}\ . \tag{1.12}$$

1.6 Acceleration of DE for Aerodynamic Design

The immediate goal of this dissertation is to determine whether the convergence rate of Rogalsky's (1998) aerodynamic design algorithm can be accelerated. If the number of flow solver calls can be reduced significantly, we are confident that the incorporation of a boundary layer module is feasible.

This is not the first attempt at acceleration of DE. Masters and Land (1997), for example hybridize DE with a direct gradient descent method. Every generation, a few vectors are selected at random, their derivatives are computed, and a single line minimization is performed along the gradient. This "appears to tremendously speed [sic] convergence while almost certainly having little impact on its ability to find a good global optimum."

Chiou and Wang (1998) embed two additional operations into DE. An acceleration operator performs a gradient descent on the population any time the cost does not increase from one generation to the next. A migration operator restarts the algorithm in another region any time population diversity decreases too much. Applying their algorithm to bioprocess control, they achieve a tenfold reduction in cost function calls, while actually finding a better solution than conventional DE.

Other new operators have been introduced for acceleration, such as a trigonometric mutation operator (Fan and Lampinen, 2002) and a population refreshment mechanism (Šmuc, 2002). Hendtlass (2001) combines DE with a particle swarm algorithm. Liu and Lampinen (2002) use fuzzy logic to define adaptively DE's parameters, making DE easier to use and more efficient. Others ideas for acceleration can be found in Lampinen's (2001) DE bibliography.

The key to significant acceleration is to exploit the uniqueness of the specific problem. Ken Price (2001) has suggested that a speed-up by more than a factor of two is likely not sustainable over a wide class of functions. In fact, this is made very clear by the No Free Lunch (NFL) Theorem for optimization (Wolpert and Macready, 1997), which states that no *a priori* claim can be made about any optimization algorithm:

Theorem (No Free Lunch): *For any pair of optimization algorithms a_1 and a_2,*

$$\sum_f P\left(d_m^y \middle| f, m, a_1\right) = \sum_f P\left(d_m^y \middle| f, m, a_2\right),$$

where m is the sample size – the number of distinct points that are examined by the algorithm, d_m^y is the set of m sample costs y, f is any objective function, and $P\left(d_m^y \middle| f, m, a\right)$ is the conditional probability of obtaining the sample cost set d_m^y on the objective function f by applying the algorithm for m iterations.

That is, averaged over all possible objective functions, all optimization algorithms perform equally. To quote Wolpert (2002): "In short, according to these theorems there is no free lunch; without tailoring one's algorithm to the domain at hand, one has no assurances that that algorithm will perform well on that domain."

The somewhat surprising consequence of the NFL Theorem is that it is impossible to analyze convergence in general. In fact, EA convergence analysis has been limited to very simple solution spaces – n-dimensional hemispheres, for example (Rudolph, 1997abc, Beyer, 1997, 2001, 2002). Rigorous convergence analysis of EAs for more complicated spaces remains an open problem. Of course, interesting, practical problems are not simple. In aerodynamic design, in particular, the solution space is extremely nonlinear, even chaotic.

While we cannot claim that any acceleration achieved will be applicable to every optimization problem, we would like to be reasonably confident of its general applicability for aerodynamic design of turbine blading. To ensure this, we use a methodology common in aerodynamic design. Three very different design targets are chosen. The design process is simulated for these targets, and the convergence rates are compared with simulations under a modified design algorithm. Any improvement that is applicable generally would have to be observed for all three design cases. The three targets are described in Chapter 2.

The first attempt to incorporate problem-specific information is to modify the hard constraints. (These force DE to consider only realistic airfoils. The flow solver does not evaluate any constraint violations.) As seen in Chapter 2, the modification of the solution space significantly reduces the cost of the final solution, but has only a minor impact on convergence rate. The resulting convergence rates for the three targets are used as benchmarks throughout the remainder of the thesis.

In Chapter 3, we present a new parameterization method for airfoils. The Bezier parameterization used previously is inadequate for several reasons, including the loss of second-order continuity at the juncture between leading and trailing curves. Furthermore, many of the Bezier parameters are not directly related to the aerodynamic properties of the shape. The incorporation of aerodynamic shape parameters, such as the leading edge radius or the trailing wedge angle, enables DE to traverse the domain more quickly.

Several optimizer-specific modifications were made as well. In Chapter 4 a variable birthrate is used, essentially neutering the costliest individuals in the population. In

Chapter 5 a hybrid strategy is developed. DE is used in concert with Downhill Simplex, a local optimizer that does not require gradient information. In Chapter 6, DE is combined with another biological model – the immune system. A particular combination of hybridization and immune acceleration is shown to reduce dramatically the number of flow solutions required to converge.

A brief digression is made in Chapter 7. Often certain soft engineering constraints must be incorporated into the design. A penalty function method for imposing such constraints is demonstrated. Finally, Chapter 8 summarizes the main results, and makes recommendations for future work.

Chapter 2 Benchmarks for Acceleration

This chapter develops a new set of constraints for the design algorithm (section 2.2), and describes the three test cases that will be used throughout the thesis (section 2.3). The convergence rates using the new constraints will serve as benchmarks for acceleration. Before discussing constraints, a few more details about Rogalsky's (1998) Bezier parameterization are needed.

2.1 Bezier Parameterization

A parametric Bezier curve $\mathbf{P}(u)$ of degree n is uniquely determined by the $n+1$ vertices of a polygon, called the control points $\mathbf{P}_i$, as follows:

$$\mathbf{P}(u)=\sum_{i=0}^{n}\mathbf{P}_i\frac{n!}{i!(n-i)!}u^i(1-u)^{n-i},\ \ 0\le u\le 1. \tag{2.1}$$

Four Bezier curves form the airfoil, as shown in Figure 1.2.

Bezier curves have many properties that are attractive for aerodynamic design. The end points of each curve are automatically fixed at the two end vertices. At an endpoint, the curve is tangent to the vector between that endpoint and the closest control point, making it simple to join curves with first order continuity. The curve always lies within the convex figure defined by the extreme points of the polygon. The curve is nth order continuous throughout and never oscillates wildly away from its defining control points.

In the encoding of the airfoil, some of the Bezier control parameters must be fixed. The endpoints of both profiles are fixed at (0,0) and (0,1), creating a blade with unit chord length. At the juncture between leading and trailing curves, 1) the last control point on the leading curve is made equal to the first on the trailing curve, and 2) the three central control points are made to be horizontally colinear. This enforces first order continuity throughout. Finally, on the leading thickness curve, the second control point has a zero *x*-coordinate, which enforces a rounded leading edge. Together, these conditions allow the airfoil shape to be defined with the fifteen variables b_i in Figure 1.2.

However, this encoding is not sufficient to ensure a realistic airfoil shape. It does not prevent negative thickness, loops, or any bumps that could cause the flow to separate (which would make the panel method extremely unreliable). The solution space must be constrained to avoid these problems.

2.2 Hard Constraints

Hard constraints ensure that any shapes considered are in fact airfoils. In Chapter 7, we discuss a method of imposing soft constraints – such as a user-defined minimum thickness. A penalty function is used to impose the hard constraints. It assigns a large random value to the objective function whenever a constraint is violated. The penalty is enforced without performing any aerodynamic computations. Normally, they would be meaningless anyway (for a shape with negative thickness, for example), and this conserves computation.

Technically, a constraint violation is evaluated by the objective function, but the computational cost is insignificant compared to that of a flow solution. Thus the number of function evaluations (NFEs) is defined to be the number of flow calculations performed by the code. This best reflects the actual computational expense of the algorithm. When comparing convergence rates, this measure is always the one used.

In the previous work (Rogalsky, 1998), constraints were imposed directly on the Bezier control points, which is not entirely satisfactory. For example, to ensure positive thickness, the *y*-values of thickness control points were constrained to be positive. This will, of course, ensure a positive-valued thickness profile, but it also eliminates from contention many interesting – and possibly useful – thickness profiles, such as the one in Figure 2.1. The negative *y*-value of the sixth distinct control point produces an inflection point without causing the curve to dip below the axis.

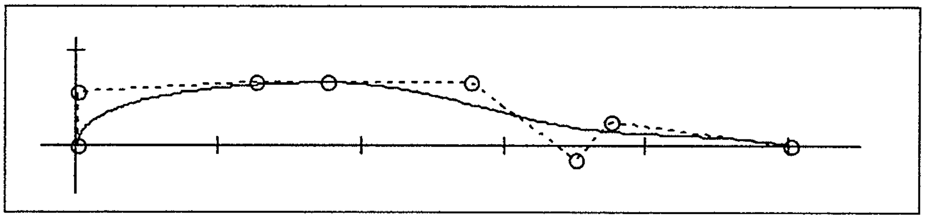


Figure 2.1. *Negative-valued control points do not necessarily create negative-valued profiles.*

Similarly, to ensure that the profiles will not loop back upon themselves, all control points were ordered downstream (i.e. left-to-right). While this is sufficient to produce single-valued profiles, again it is not necessary. For example, in the camber profile shown in Figure 2.2, the sixth distinct control point lies upstream of the fifth. Yet the profile is not only aerodynamically valid, but in fact has a potentially valuable inflection point caused by that ordering.

Furthermore, the constraints as proposed were not general enough to remove all irregularities. Two examples are shown in Figure 2.3. These were removed by

1) requiring a minimum horizontal separation between all control points, and 2) imposing a maximum value by which any control point could exceed vertically the juncture. But again, a consequence is that some valid shapes will not be included in the solution space.

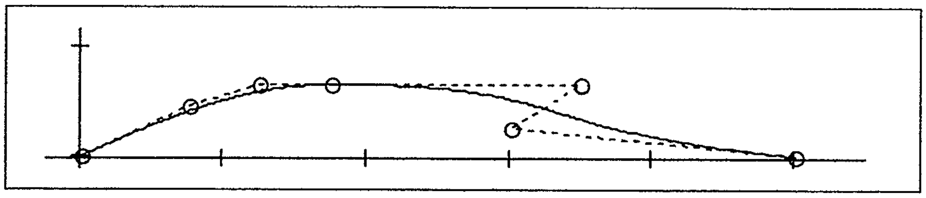


Figure 2.2. *Reverse-ordered control points do not necessarily create multi-valued profiles.*

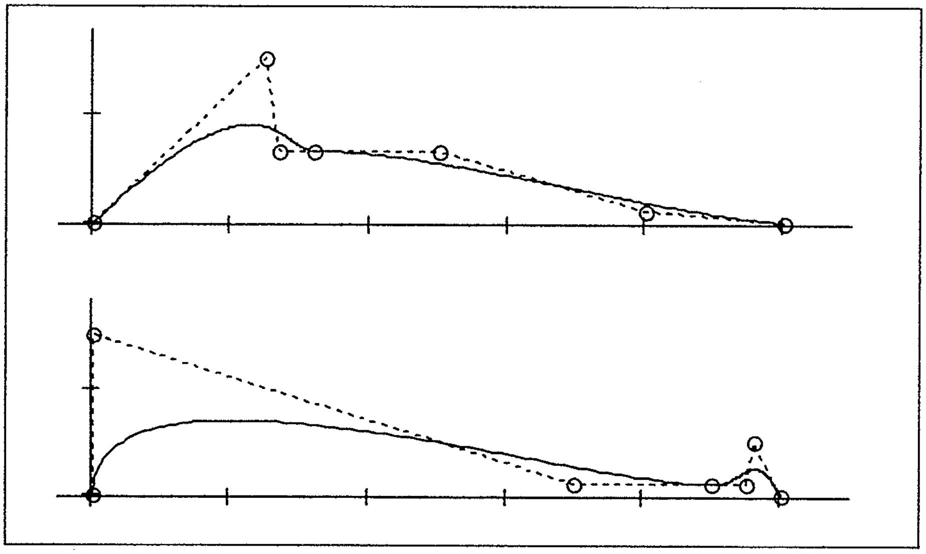


Figure 2.3. *Some irregularities that can occur if the leading or trailing curves are not one-to-one.*

An improved set of constraints is required. What we really want is to control the curves – not the points. There is, in fact, a very simple solution: Each of the four Bezier curves must be one-to-one functions, with the leading curves increasing monotonically, and the trailing curves decreasing monotonically. This enforces the condition of an airfoil without bumps, loops, or negative thickness; while allowing negative-valued, reverse-ordered, or very close control points – along with any beneficial features that they may produce.

We will see in Chapter 3 that this one-to-one constraint is entirely sufficient for the new airfoil representation methods developed there. For the Bezier parameterization, however, that is not quite true. One control point constraint has to be maintained. Since curves are joined with only first-order continuity, the juncture was not always reasonably smooth. If the colinear points at the juncture between curves are too close to each other, the juncture may not even appear first-order continuous (Figure 2.4). To regulate this, we enforce a minimum horizontal separation of the juncture points. The minimum separation is 0.05 (5% of the chord), a somewhat arbitrary value that seems nonetheless to work.

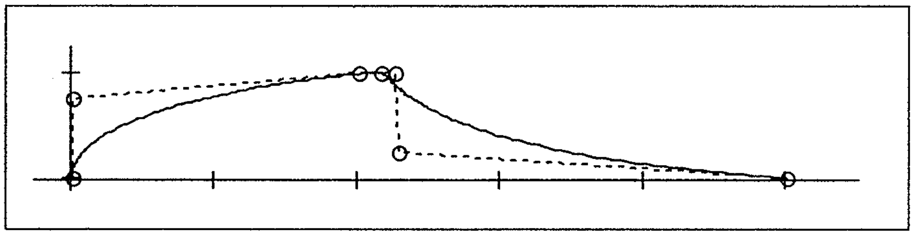


Figure 2.4 *In addition to the one-to-one constraints, a minimum separation of juncture points is required to approximate second-order continuity.*

As will be seen in the next section, these new one-to-one constraints allow DE to find a better solution, without detriment to the rate of convergence.

2.3 Benchmarks

Three design cases will be examined throughout this dissertation. These are diverse enough that any consistent acceleration pattern should be applicable across a wide spectrum of designs. In this section, the convergence rates for each design using the old design algorithm will be given. These will serve as benchmarks with which to compare acceleration strategies.

For two design cases, the solutions are known. Gostelow (1964, 1984), using conformal transformations, provided several standard cascade profiles with exact surface pressure distributions. These have been used to verify numerical results. Two of these – a C4/70/C50 airfoil and a highly cambered (112°) impulse cascade profile (Figure 2.5) – were used by Lewis (1991) to demonstrate the validity of the Martensen vorticity panel method. The pressure distributions of these two standard blades were used as design targets in Rogalsky et. al. (2000). They will be used throughout here – although in slightly modified form – as discussed below. The third design target is the Liebeck pressure distribution, shown in Figure 1.4.

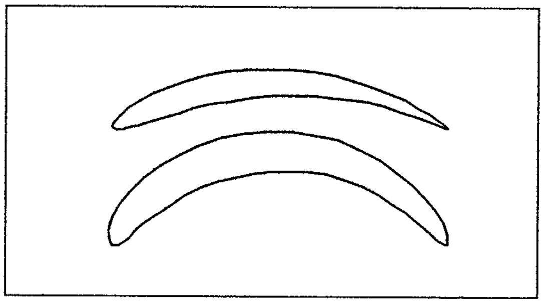


Figure 2.5. *Known solutions for two design targets: C4/70/C50 (top), 112°-cambered blade (bottom).*

Design parameters for the benchmarks are as follows. For each case, the DE/rand-to-best/1/bin variation is used, with *NP*=10*D* (*D*=17 for Bezier designs with variable blade spacing), *F*=0.85, and *CR*=1.0 (no crossover). This variant of DE is advertised by its authors as effective for a broad range of problems, and was found to be robust for aerodynamic acceleration in Rogalsky et al (2000). One-to-one and juncture constraints are used for all three benchmarks. An additional leading edge constraint was necessary for the Liebeck case.

The sections that follow describe the design targets and any modifications made, examine the effect of the modified targets and of the different constraints on convergence, and provide the benchmarks for acceleration.

2.3.1 C4/70/C50 Target

The C4/70/C50 blade was used for compressors in the U.K. in the 1960's. It has a C4 base profile distributed upon a 70° circular arc camber line. C-series airfoils are still very representative of turbine blading today. By experimentation, Gostelow was able to reproduce this blade using a conformal transformation. Exact flow solutions were obtained for two inlet angles, $\beta_1=\pm 35^\circ$, using pitch/chord ratio t/l=0.900364 and stagger angle λ=0 for each. As shown by Lewis, these are in "excellent agreement" with the surface pressure distributions obtained by Martenson's panel method.

In our previous work (Rogalsky et al, 2000), the two pressure distributions computed by the panel method were used as design targets, with pitch/chord ratio and stagger angle fixed. For β_1=+35°, the solution was closely approximated by not only DE, but also two other optimizers – Downhill Simplex and Simulated Annealing. However, both of these failed badly for β_1=-35°, while DE found a solution in 57,000 FEs. Since we would like to challenge the design scheme, only the more difficult target (β_1=-35°) is used here. The corresponding outlet angle is β_2=-25.0384°.

The pressure distribution has *Cp* coefficients with extremely high magnitude near the leading edge. (See Figure 2.6.) Relative to the other target points, the deviations at the leading edge points will tend to be higher. Yet the proximity of the solution there is less important than the proximity for the rest of the distribution. That is, we are unduly forcing the algorithm to match closely the target in a region that is both extremely difficult to match and relatively unimportant to the blade's performance. A better design target would allow the pressure distribution to be somewhat free for magnitudes higher than a certain value. We have thus eliminated from the target pressure distribution all pressure coefficients with absolute value greater than 6. Convergence to the new target is used as the new benchmark for comparison.

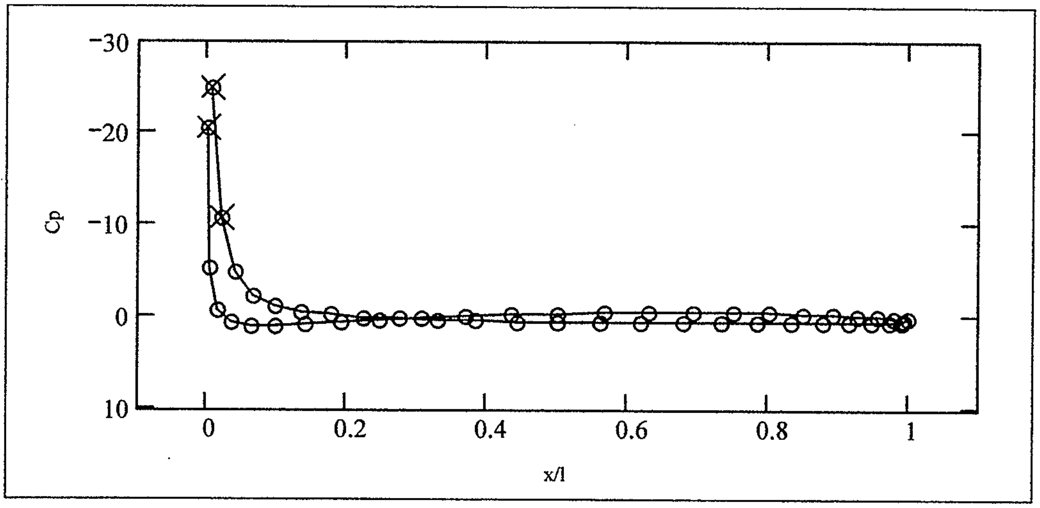


Figure 2.6 *Original C4/70/C50 target distribution, calculated with β_1=-35°, t/l=0.900364, λ=0. Pressure coefficients with magnitude greater than 6 are deleted for the remainder of the thesis.*

The different convergence rates are shown in Figure 2.7. For comparison the results of Rogalsky et al (2000), using fixed blade spacing, are included. This in fact results in the highest converged cost. All other results use variable spacing, since it is not reasonable in general to assume that the spacing is known in advance. Interestingly, the addition of variable spacing alone enables DE to find a solution with lower cost. The modified target results in a much lower cost, for obvious reasons, although the actual rate of convergence doesn't appear to change much. As expected, the new one-to-one constraints allowed greater geometric freedom, resulting in the best solution.

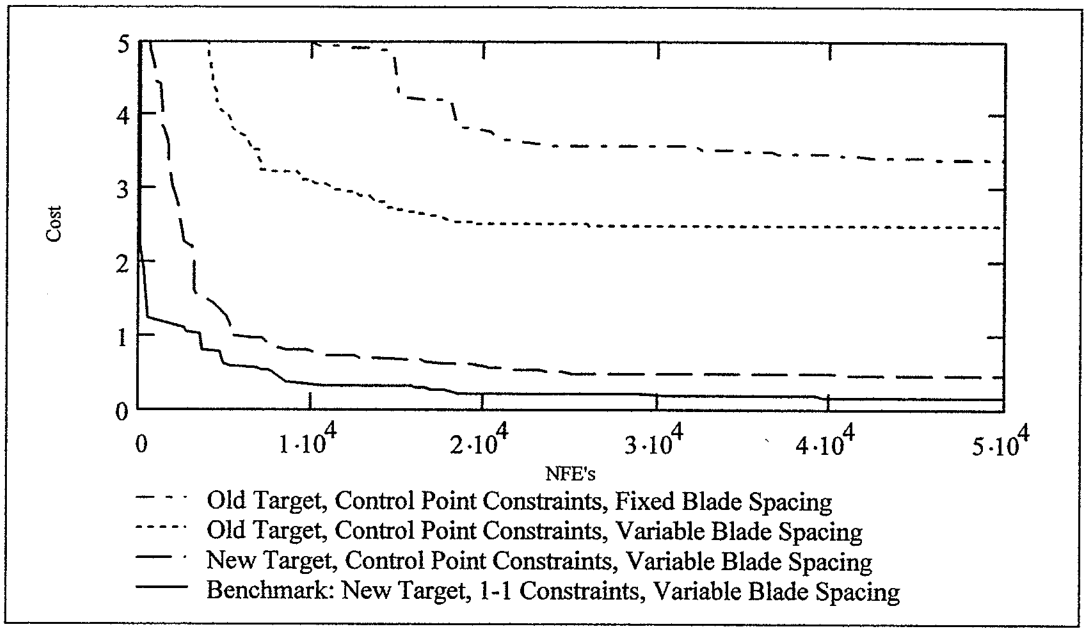
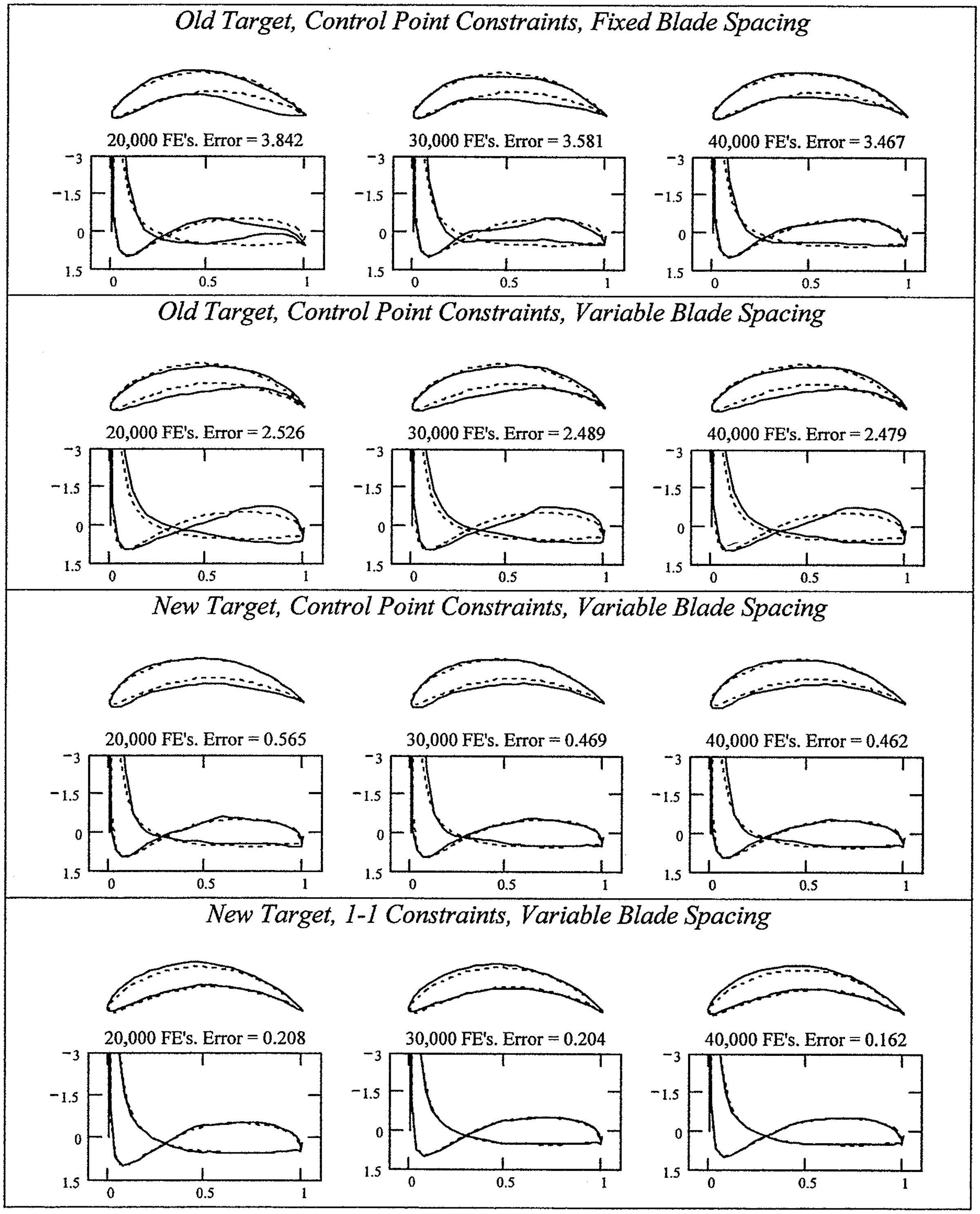


Figure 2.7 *Effect of target modification and constraint system on convergence to the C4/70/C50 target.*

It is also informative to compare "snapshots" of the design process. Figure 2.8 shows the blade shape and pressure distribution as it is developed by DE – after 20,000, 30,000, and 40,000 FEs. In the original case (fixed spacing), significant change occurs after 30,000 FEs, but for the remaining three, the shape has nearly converged by then, and subsequent modifications are only minor.



Old Target, Control Point Constraints, Fixed Blade Spacing

20,000 FE's. Error = 3.842

30,000 FE's. Error = 3.581

40,000 FE's. Error = 3.467

Old Target, Control Point Constraints, Variable Blade Spacing

20,000 FE's. Error = 2.526

30,000 FE's. Error = 2.489

40,000 FE's. Error = 2.479

New Target, Control Point Constraints, Variable Blade Spacing

20,000 FE's. Error = 0.565

30,000 FE's. Error = 0.469

40,000 FE's. Error = 0.462

New Target, 1-1 Constraints, Variable Blade Spacing

20,000 FE's. Error = 0.208

30,000 FE's. Error = 0.204

40,000 FE's. Error = 0.162

Figure 2.8 *Convergence snapshots showing the effects of target modification and constraint system on the C4/70/C50 design. The target is shown with the dotted line.*

2.3.2 112°-cambered Target

As a more extreme test, Gostelow developed a highly cambered impulse cascade profile, based on a 112° circular arc camber line. The exact solution was given for β_1=50°, t/l=0.5899644, and λ=0. Again, the pressure distribution compares well with the panel method solution, and is used as the target in Rogalsky et al (2000) with fixed blade spacing. The target outlet angle is β_2=-53.2832°.

Recall that the Bezier parameterization in the design algorithm operates under the assumption that each blade has leading edge at (0,0) and trailing edge at (1,0). This is not the case for the data points reported in Lewis. The blade was thus shifted to the origin, and then magnified to have chordlength one. Recalculating the pressure distribution provides a target for which the algorithm is designed.

A brief digression is necessary regarding panel length. Martenson's method is most effective if any given panel has length equal to the blade thickness at the panel's midpoint. The design algorithm automatically accomplishes this in the representation of the blade. However, a maximum panel length must be imposed for most airfoils. In general, a maximum length of 0.04 is sufficient for fast, accurate results. In the case of the 112°-cambered blade, however, the thickness increases so rapidly at the leading edge that too much data is lost, making it difficult to approximate the pressure coefficients there. Changing the maximum thickness to 0.03 resolved the problem. Design for the highly cambered case will use 0.03 throughout. The other two design problems will be solved with maximum panel length 0.04.

The developing cost is compared in Figure 2.9, and design snapshots are shown in Figure 2.10. In all cases, there are no significant shape or pressure distribution changes after 30,000 FEs. Modifying the target significantly decreased the deviations, especially along the suction surface. Again, the one-to-one constraints allowed the design method to find the lowest cost.

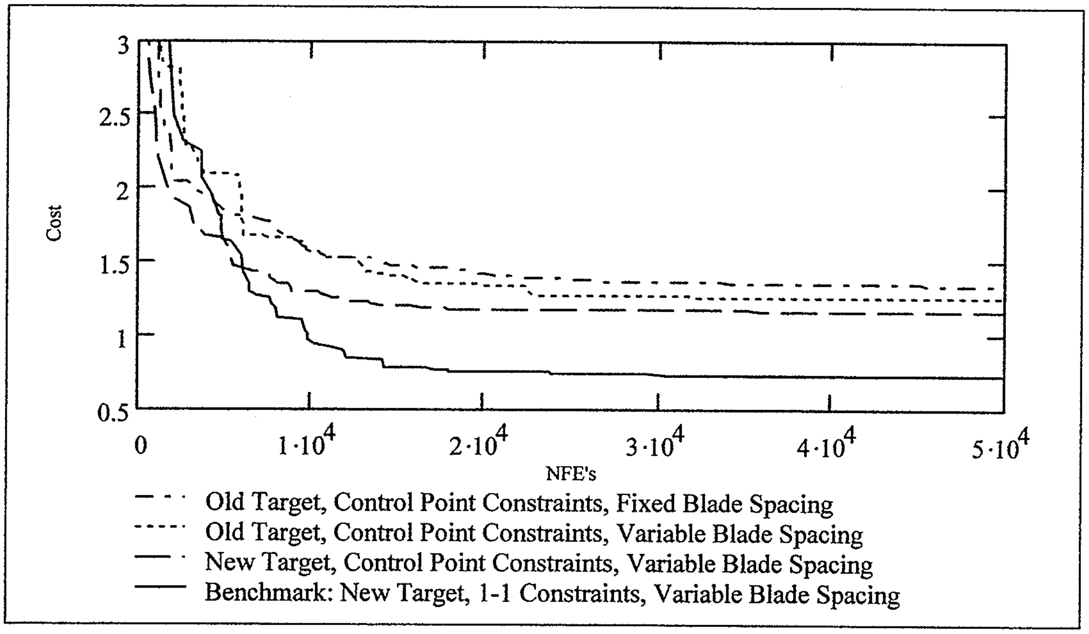


Figure 2.9 *Effect of target modification and constraint system on convergence to the 112°-cambered target.*

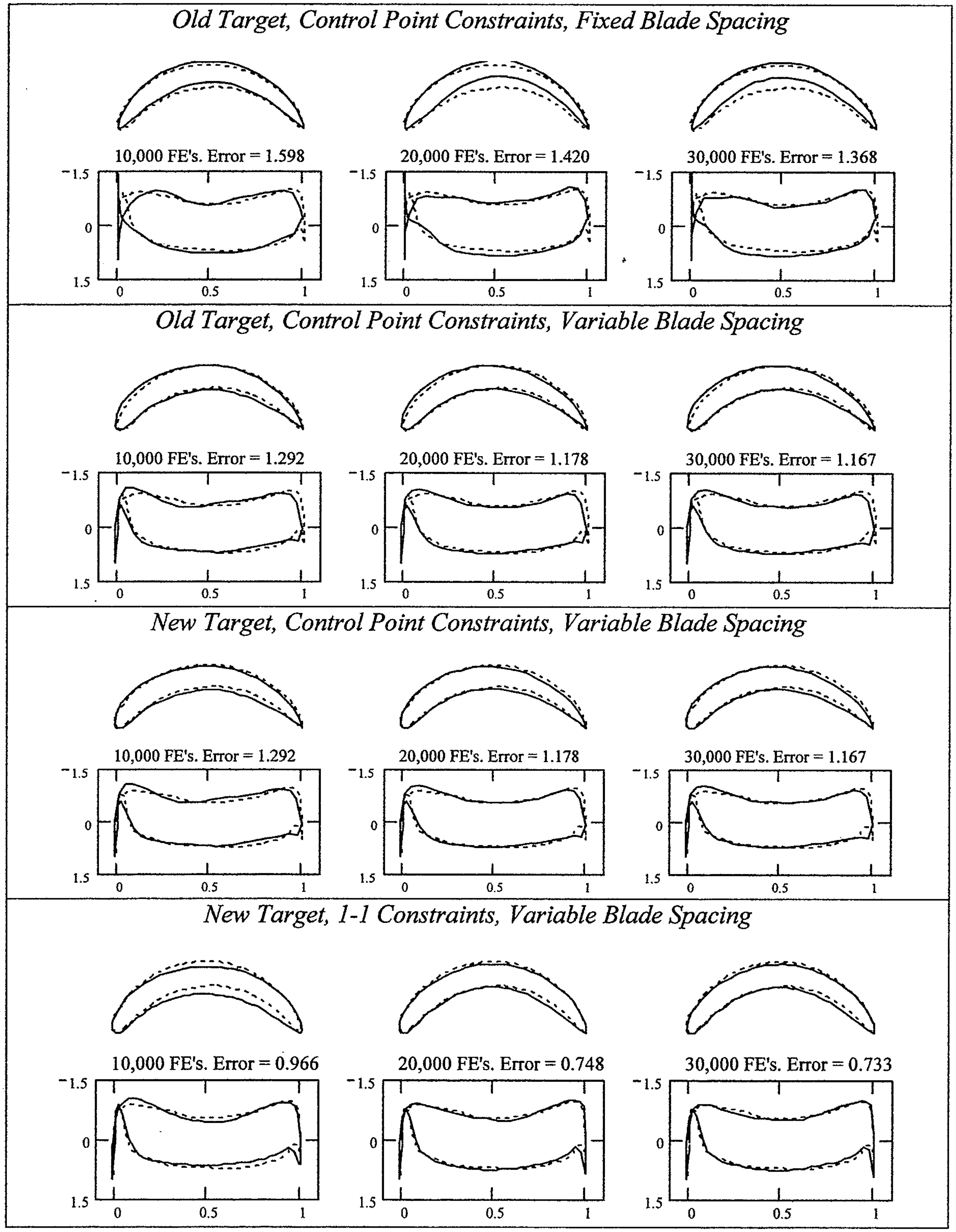


Figure 2.10 *Convergence snapshots showing the effects of target modification and constraint system on the 112°-cambered design.*

2.3.3 Liebeck Target

The final design target is one for which the solution is not known. Liebeck pressure distributions have been discussed in Chapter 1. They generate excellent lift coefficients while avoiding separation. The target chosen is shown in Figure 1.4, with inlet flow angle β_1=30°, and target outlet flow angle β_2=0°.

The new constraint system, consisting of one-to-one curves and a minimum separation of juncture control points, was insufficient for this case. The resulting design was unrealistically thin and had a sharp leading edge, as seen in the design snapshots (Figure 2.12). To ensure a rounded leading edge for the Liebeck design, we require a minimum vertical separation of 0.05 for the first two thickness control points. The resulting convergence is set as the benchmark.

Convergence is compared in Figure 2.11, and snapshots in Figure 2.12. Results are dramatic. Under the new constraint system, the design has not only converged by 10,000

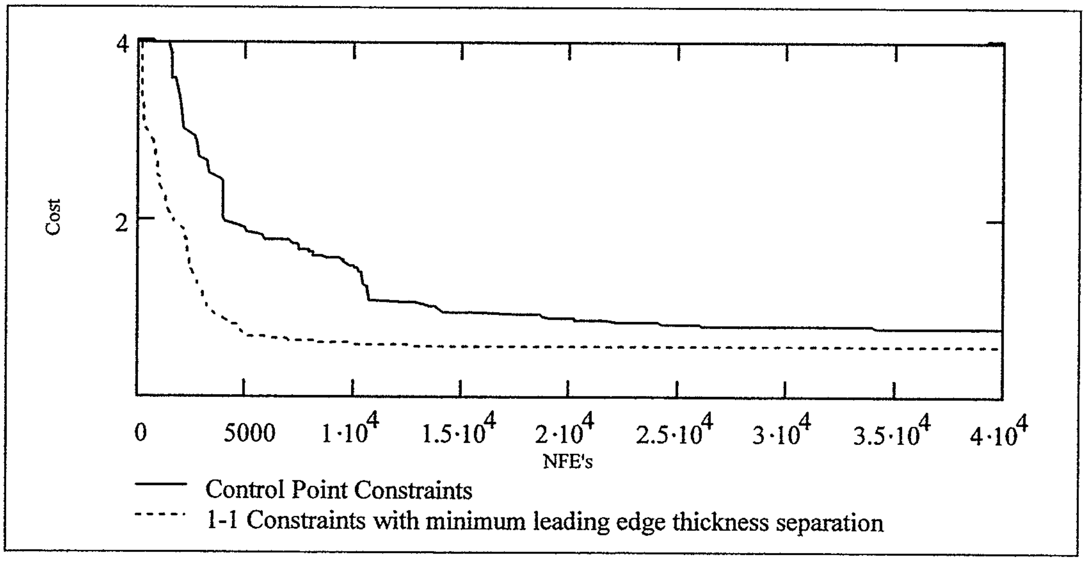


Figure 2.11 *Effect of constraint system on convergence to the Liebeck target.*

FEs (vs. about 30,000 FEs under the control point constraints), but has also achieved a significantly lower error (0.566 vs. 0.785 after 40,000 FEs).

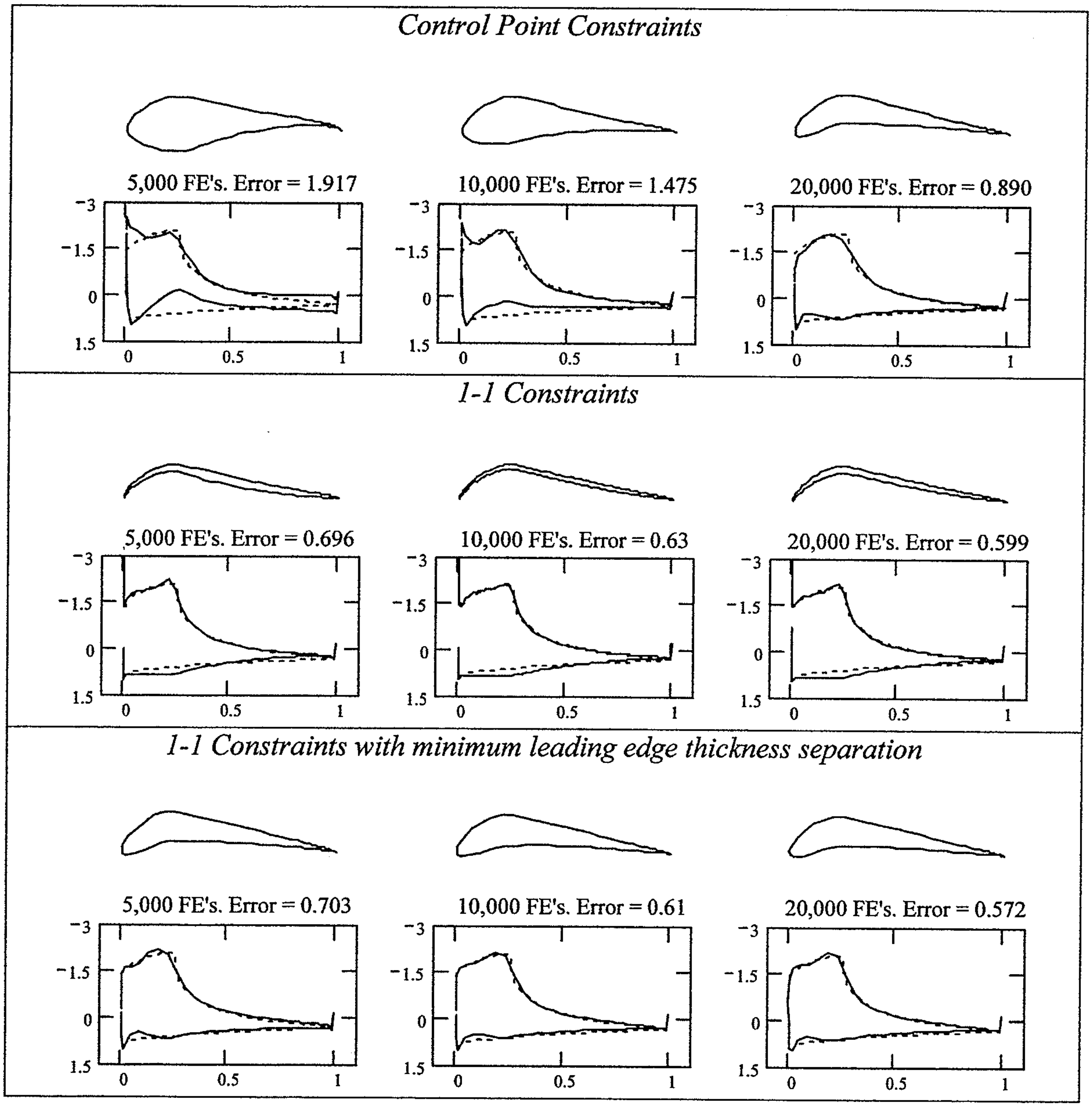


Figure 2.12 *Convergence snapshots showing effect of constraint system on the Liebeck design.*

Chapter 3 Geometric Representation of Airfoils

Airfoil data is typically tabulated with 50 to 80 data points. In aerodynamic design, however, the number of function evaluations required for convergence tends to increase linearly with the number of parameters used to represent the geometric shape. Thus it is preferable that airfoils be represented by a much smaller number of parameters.

There are several goals to keep in mind when developing a method to represent airfoils geometrically. First, the number of parameters used to represent the shape should be kept to a minimum. Second, the method should be able to represent a wide variety of airfoils. Third, any constraints on the design should be simple to formulate and to impose. Fourth, the parameterization should lend itself to effective and efficient optimization.

This chapter describes a new method of representing the geometry of airfoils. It is an extension of the Bezier parameterization discussed in sections 1.3 and 2.1, using new aerodynamic parameters. These are similar to the PARSEC parameters developed by Sobieczky (1999). Bezier-PARSEC (BP) parameterization is summarized in section 3.1, and two instances of it are developed sections 3.2 and 3.3. To compare the ability of the Bezier and BP methods to represent typical airfoils, section 3.4 tests 63 different airfoils. (Details of the parameterization of each of the 63 appear in Appendix B on the accompanying CD.) Finally, the BP parameterization is implemented into the aerodynamic design code. Section 3.5 discusses the effects on robustness and convergence speed.

3.1 Bezier-PARSEC Parameterization

The parameters, b_i, in the Bezier parameterization are actual Bezier control points. They define the Bezier curves, which in turn define the airfoil, but they influence the flow only indirectly. It is possible to parameterize an airfoil using quantities that more directly control its aerodynamics - such as the leading edge radius and the trailing wedge angle. PARSEC parameterization of airfoils (Sobieczky, 1998, 1999) uses flow phenomena oriented parameters, and was developed specifically for aerodynamic optimization.

PARSEC airfoils are parameterized by their upper and lower curves, each of which is a linear combination of shape functions as follows

$$z_i = \sum_{n=1}^{6} a_n^i x^{n-1/2}, \quad i = 1,2\,, \tag{3.1}$$

where z_1 is the height of the upper curve, z_2 is the height of the lower curve, x is the distance along the chord, and a_n^i are undetermined coefficients. Eleven parameters (illustrated in Figure 3.1) define an airfoil with unit chord length: r_{le}, the radius of curvature of the leading edge; x_{up}, z_{up}, the position of the upper crest; z_{xxup}, the second derivative of (3.1) at the upper crest; x_{lo}, z_{lo}, the position of the lower crest; z_{xxlo}, the second derivative at the lower crest; α_{te}, the direction of the trailing edge; β_{te}, the wedge angle; z_{te}, the height of the trailing edge; and Δz_{te}, the trailing edge thickness. These quantities, together with the x-coordinate of the trailing edge, are substituted into (3.1). The resulting system of twelve equations is solved for the twelve coefficients a_n^i.

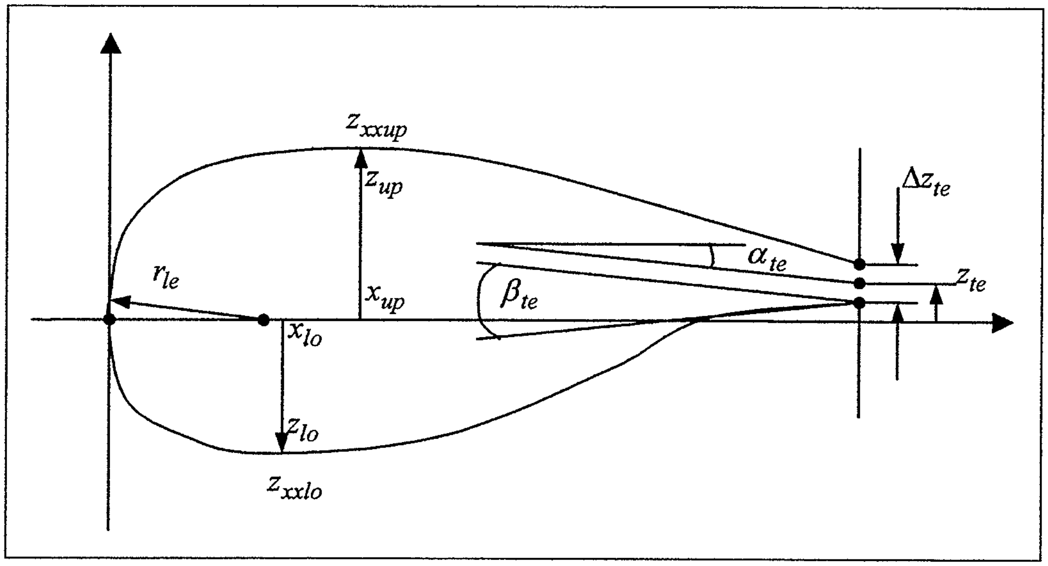


Figure 3.1 *PARSEC airfoil geometry defined by 11 basic aerodynamic parameters.*

It has been shown by Oyama et al (1999) that this type of parameterization improves both robustness and convergence speed for aerodynamic optimization, and that it is particularly well-suited to optimization by GAs. He compares five parameterization methods, both for their ability to reproduce known airfoils, and for their robustness and convergence speed in aerodynamic design. These five are: the extended Joukowski transformation (Jones, 1990), the inverse Theodorsen transformation (Theodorsen and Garrick, 1933), B-spline curves (similar to Bezier curves), orthogonal shape functions (Chang et. al., 1995), and PARSEC parameterization.

Oyama examined first the ability of each parameterization to represent two airfoils, one a NASA supercritical airfoil, and the other a four-digit NACA airfoil. The Extended Joukowski and orthogonal shape function methods were unable to reproduce these airfoils.

Four parameterizations were integrated into an aerodynamic design algorithm, using a GA for optimization. The objective function was the maximization of the lift-to-drag ratio. The results for PARSEC (L/D 39.40) and B-Spline (L/D 39.02) were significantly better than those for Extended Joukowski (L/D 34.73) and Theodorsen (L/D 31.87). However, the convergence rate for the PARSEC-parameterized design was significantly better than that for the B-Spline design, requiring roughly half the number of function evaluations to converge.

The superior performance of the PARSEC parameterization is likely due to its ability to minimize an optimization phenomena known as epistasis (the nonlinear manner in which the objective function is dependent on the design parameters). Small changes in several variables can result in large changes in the objective function. Epistatic functions are difficult to minimize because they provide so few clues as to the location of the global minimum. In general, a reduction of this nonlinear interaction will enable the optimizer to converge more quickly. Because the PARSEC parameters are motivated by aerodynamics, the nonlinearities should be of lesser magnitude. For example, a small variation in r_{le} should result in a small change in the flow at the leading edge region. After an analysis of the interactions between PARSEC parameters, Oyama found that the relationships between z_{up}, z_{lo}, and z_{te} were still very complicated. He was able to achieve better results by converting the upper and lower heights into camber and thickness heights.

Motivated by the work of Sobieczky and Oyama, a new method of airfoil parameterization is proposed. Bezier-PARSEC (BP) airfoils are designed to combine the benefits of the Bezier parameterization - notably the camber-thickness formulation and the advantages of Bezier curves, including usage by industry - with the improved convergence characteristics provided by PARSEC variables. It is hoped that the new method will accelerate the convergence of aerodynamic design using DE.

The BP parameterization uses a new set of optimization parameters, which are aerodynamically oriented. These parameters are then used to determine the control points of four Bezier curves - leading and trailing camber, leading and trailing thickness. The airfoil is composed as in Figure 1.2, with thickness measured normal to the camber curve. One deficiency of the Bezier parameterization is corrected for all BP shapes: Leading and trailing curves are joined with second-order continuity.

The BP variables are as follows. Several are PARSEC variables: r_{le}, α_{te}, β_{te}, and z_{te}. Others are similar to PARSEC parameters: γ_{le}, the leading edge direction; x_c, y_c, the position of the camber crest; κ_c, the curvature at the camber crest; x_t, y_t, the position of the thickness crest; κ_t, the curvature at the thickness crest; and dz_{te}, the half thickness at the trailing edge. Several Bezier variables are also used, and are labeled according to the convention used in Figure 1.2: b_0, b_2, b_8, b_{15}, b_{17}. The last two variables are used for degree four trailing edge curves. They represent the x-value of the fourth control point on the thickness and camber curves, respectively.

The Bezier variables, b_i, have a slightly different interpretation in BP parameterization. Since one goal is to decrease epistasis, it is preferable that the Bezier variables used be related to the aerodynamics of the shape. For example, b_{15} should most directly influence the existence and/or nature of an inflection point on the trailing thickness curve. However, if x_t is modified, while the control point b_{15} remains constant, the position and shape of the inflection point can change dramatically. This effect is diminished if b_{15} represents the ratio between x_t and x=1 (the trailing edge position). Thus we let all Bezier variables in the BP parameterizations represent the ratio between the endpoints of the curve it defines. For example, $b_{15} = (x - x_t)/(1 - x_t)$, where x is the actual x-value of the fourth control point on the trailing thickness curve.

It will be necessary to distinguish between the four Bezier curves and their control points. Each parametric curve is composed of two functions, $x(u)$ and $y(u)$. A two-letter superscript will be used to distinguish a curve. The first letter is either l or t, indicating leading or trailing, respectively. The second is either t or c, representing thickness or camber, respectively. Control points are superscripted according to their corresponding curve, and subscripted by number. For example, $\left(x^{lc}(u), y^{lc}(u)\right)$ represents the leading camber curve of degree n, with Bezier control points $\left(x_i^{lc}, y_i^{lc}\right)$, $i = 0,\ldots,n$.

Two instances of the Bezier-PARSEC parameterization are examined in this chapter. Each begins with the initials BP, followed by four digits. The digits represent (in order) the degree of each of the following Bezier curves used by the parameterization: Leading thickness curve, trailing thickness curve, leading camber curve, trailing camber curve. Thus the BP 3333 parameterization uses exclusively degree three Bezier curves, and BP 3434 has a degree 3 leading thickness curve, a degree 4 trailing thickness curve, a degree 3 leading camber curve, and a degree 4 trailing camber curve. A BP 3424 parameterization was also integrated into FanOpt, but was less robust and showed no acceleration. Details of BP 3424 are not given.

The Bezier-PARSEC parameterization is envisioned to have advantages over both the Bezier and the PARSEC methods. It avoids the second-order discontinuity problem inherent with the Bezier parameterization, and it uses aerodynamic parameters, which should result in accelerated convergence for design optimization. In contrast to the PARSEC parameterization, the camber-thickness formulation is more natural for airfoils than PARSEC's upper-lower curve formulation. The leading edge radius can now be measured in the direction of the camber, rather than that of the axis, and the leading edge direction – not used by PARSEC – is now made explicit. Finally, BP parameterizations incorporate Bezier curves, which have widespread use in industry.

3.2 BP 3333 Parameterization

The BP 3333 parameterization is defined by twelve exclusively aerodynamic parameters. These are illustrated in Figure 3.2. Compared with the Bezier parameterization, there are two significant improvements that are expected to accelerate the convergence of DE in the design process. As has been mentioned in section 3.1, the aerodynamic character of the parameters should reduce epistasis in the objective function, making the global minimum easier to find. Additionally, however, there is a significant reduction in the number of parameters. In fact, if the trailing edge is fixed at (1,0) with zero thickness - as it is for the Bezier parameterization - the number of variables used to represent an airfoil is reduced from 15 to 10. Equations for the associated Bezier control points are derived in sections 3.2.1 – 3.2.4.

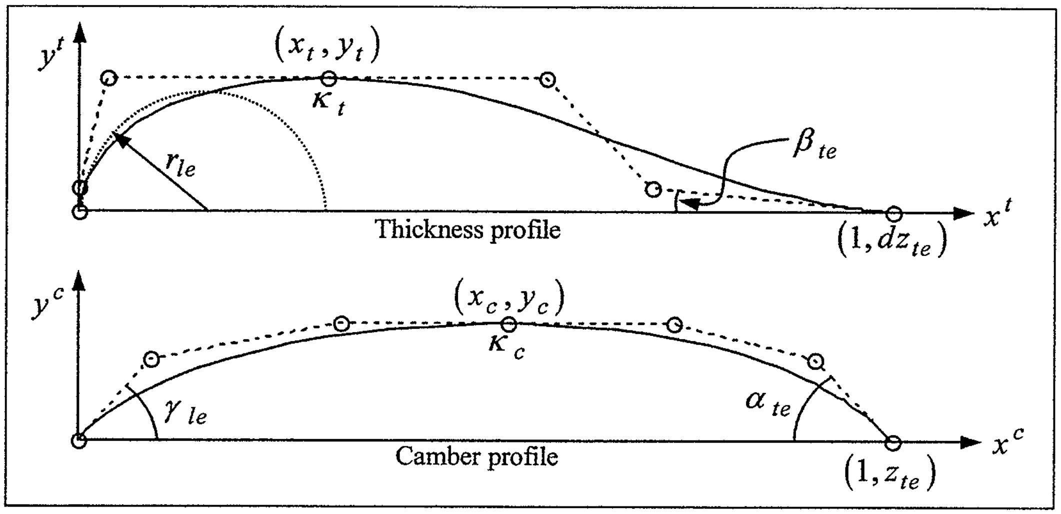


Figure 3.2 *BP 3333 airfoil geometry and Bezier control points defined by twelve basic aerodynamic parameters.*

3.2.1 Leading Thickness Curve (Degree 3): (x_i^{lt}, y_i^{lt})

Degree three Bezier curves are given by

$$\begin{cases} x(u) = x_0(1-u)^3 + 3x_1u(1-u)^2 + 3x_2u^2(1-u) + x_3u^3 \\ y(u) = y_0(1-u)^3 + 3y_1u(1-u)^2 + 3y_2u^2(1-u) + y_3u^3 \end{cases}, \tag{3.2}$$

with first derivatives

$$\begin{cases} x'(u) = -3x_0(1-u)^2 + 3x_1(1-4u+3u^2) + 3x_2(2u-3u^2) + 3x_3u^2 \\ y'(u) = -3y_0(1-u)^2 + 3y_1(1-4u+3u^2) + 3y_2(2u-3u^2) + 3y_3u^2 \end{cases}, \tag{3.3}$$

and second derivatives

$$\begin{cases} x''(u) = 6x_0(1-u) + 6x_1(-2+3u) + 6x_2(1-3u) + 6x_3 u \\ y''(u) = 6y_0(1-u) + 6y_1(-2+3u) + 6y_2(1-3u) + 6y_3 u \end{cases}. \tag{3.4}$$

Since the leading edge is at (0,0), $x^{lt}(0) = y^{lt}(0) = 0$, which together with (3.2) implies

$$x_0^{lt} = y_0^{lt} = 0. \tag{3.5}$$

For the blade to have a rounded leading edge, the thickness curve must be vertical at (0,0). That is, $\left(x^{lt}\right)'(0) = 0$. From (3.3) and (3.5), then

$$x_1^{lt} = 0. \tag{3.6}$$

In general, the radius of curvature of a parametric curve at $u = 0$ is

$$r = \frac{\left[\left(x'(0)\right)^2 + \left(y'(0)\right)^2\right]^{\frac{3}{2}}}{x'(0)y''(0) - y'(0)x''(0)}.$$

Since $\left(x^{lt}\right)'(0) = 0$ from above, the leading edge radius is given by

$$r_{le} = \frac{\left[\left(y^{lt}\right)'(0)\right]^2}{-\left(x^{lt}\right)''(0)} = \frac{\left[-3y_0^{lt} + 3y_1^{lt}\right]^2}{-6x_0^{lt} + 12x_1^{lt} - 6x_2^{lt}},$$

from (3.3), (3.4). Using (3.5) and (3.6), this results in the following equation relating x_2^{lt} and y_1^{lt},

$$-2r_{le}x_2^{lt} = 3\left(y_1^{lt}\right)^2. \tag{3.7}$$

The juncture between the leading and trailing curves has coordinates $x^{lt}(1) = x_t$, $y^{lt}(1) = y_t$ and is horizontal (i.e. $\left(y^{lt}\right)'(1) = 0$). Thus, from (3.2) and (3.3),

$$\begin{cases} x_3^{lt} = x_t \\ y_2^{lt} = y_3^{lt} = y_t \end{cases}. \tag{3.8}$$

In summary, the control points for the leading thickness curve are given by

$$\begin{cases} x_0^{lt} = 0 \\ x_1^{lt} = 0 \\ x_2^{lt} = ? \\ x_3^{lt} = x_t \end{cases} \qquad \begin{cases} y_0^{lt} = 0 \\ y_1^{lt} = ? \\ y_2^{lt} = y_t \\ y_3^{lt} = y_t \end{cases},$$

with variables x_2^{lt} and y_1^{lt} satisfying (3.7). These will be determined in the subsequent section.

3.2.2 Trailing Thickness Curve (Degree 3): (x_i^{tt}, y_i^{tt})

The juncture between the leading and trailing curves has coordinates $x^{tt}(0) = x_t$, $y^{tt}(0) = y_t$ and is horizontal (i.e. $\left(y^{tt}\right)'(0) = 0$). Thus, from (3.2) and (3.3),

$$\begin{cases} x_0^{tt} = x_t \\ y_0^{tt} = y_1^{tt} = y_t \end{cases}. \tag{3.9}$$

In general, the curvature of a parametric curve at $u = 0$ is

$$\kappa = \frac{x'(0)y''(0) - y'(0)x''(0)}{\left[\left(x'(0)\right)^2 + \left(y'(0)\right)^2\right]^{\frac{3}{2}}}$$

Since $\left(y^{tt}\right)'(0)=0$ from above, the curvature at the juncture between curves is given by

$$\kappa_t=\frac{\left(y^{tt}\right)''(0)}{\left[\left(x^{tt}\right)'(0)\right]^2}=\frac{6y_0^{tt}-12y_1^{tt}+6y_2^{tt}}{\left[-3x_0^{tt}+3x_1^{tt}\right]^2},$$

from (3.3), (3.4). Using (3.9), this results in the following equation relating x_1^{tt} and y_2^{tt},

$$2\left(y_2^{tt}-y_t\right)=3\kappa_t\left(x_1^{tt}-x_t\right)^2 \tag{3.10}$$

The curvature at the juncture should be the same for both leading and trailing curves. That is,

$$\frac{\left(y^{lt}\right)''(1)}{\left[\left(x^{lt}\right)'(1)\right]^2}=\frac{\left(y^{tt}\right)''(0)}{\left[\left(x^{tt}\right)'(0)\right]^2}.$$

To satisfy this requirement, it is sufficient that $\left(y^{lt}\right)''(1)=\left(y^{tt}\right)''(0)$ and $\left(x^{lt}\right)'(1)=\left(x^{tt}\right)'(0)$. Consider first y''. Using (3.4) with control points (3.8), (3.9),

$$\begin{aligned}\left(y^{lt}\right)''(1)&=6y_1^{lt}-12y_2^{lt}+6y_3^{lt}\\&=6y_1^{lt}-6y_t\end{aligned},$$

and

$$\begin{aligned}\left(y^{tt}\right)''(0)&=6y_0^{tt}-12y_1^{tt}+6y_2^{tt}\\&=6y_2^{tt}-6y_t\end{aligned}.$$

Equating these second derivatives implies that control points y_1^{lt} and y_2^{tt} must be equal:

$$y_1^{lt} = y_2^{tt} . \tag{3.11}$$

Now consider x'. Using (3.3) with control points (3.8), (3.9),

$$\begin{aligned}\left(x^{lt}\right)'(1) &= -3x_2^{lt} + 3x_3^{lt} \\ &= -3x_2^{lt} + 3x_t\end{aligned} ,$$

and

$$\begin{aligned}\left(x^{tt}\right)'(0) &= -3x_0^{tt} + 3x_1^{tt} \\ &= -3x_t + 3x_1^{tt}\end{aligned} .$$

Equating these first derivatives results in the following relationship between x_2^{lt} and x_1^{tt}:

$$x_2^{lt} = 2x_t - x_1^{tt} . \tag{3.12}$$

Since the trailing edge of the thickness curve is at $(1, dz_{te})$, $x^{tt}(1) = 1$ and $y^{tt}(1) = dz_{te}$, which, from (3.2), implies

$$\begin{cases} x_3^{tt} = 1 \\ y_3^{tt} = dz_{te} \end{cases} . \tag{3.13}$$

The trailing edge angle of the thickness curve is β_{te}, which will be taken to be positive, so the slope at the trailing edge must equal $-\tan(\beta_{te})$. That is,

$$\left.\frac{dy}{dx}\right|_{x=1} = \frac{\left(y^{tt}\right)'(1)}{\left(x^{tt}\right)'(1)} = -\tan(\beta_{te}) ,$$

where

$$\begin{aligned}\left(y^{tt}\right)'(1) &= -3y_2^{tt} + 3y_3^{tt} \\ &= -3y_2^{tt} + 3dz_{te}\end{aligned}$$

from (3.3), (3.13), and

$$\begin{aligned}\left(x^{tt}\right)'(1) &= -3x_2^{tt} + 3x_3^{tt} \\ &= -3x_2^{tt} + 3\end{aligned}$$

from (3.3), (3.13). Solving for x_2^{tt} results in

$$x_2^{tt} = 1 + \left(dz_{te} - y_2^{tt}\right)\cot(\beta_{te}). \tag{3.14}$$

There are five thickness variables yet to be determined (x_2^{lt}, y_1^{lt}, x_1^{tt}, x_2^{tt}, y_2^{tt}), and five equations governing them. These equations are repeated here for the reader's convenience.

$$-2r_{le}x_2^{lt} = 3\left(y_1^{lt}\right)^2 \tag{3.7}$$

$$2\left(y_2^{tt} - y_t\right) = 3\kappa_t\left(x_1^{tt} - x_t\right)^2 \tag{3.10}$$

$$y_1^{lt} = y_2^{tt} \tag{3.11}$$

$$x_2^{lt} = 2x_t - x_1^{tt} \tag{3.12}$$

$$x_2^{tt} = 1 + \left(dz_{te} - y_2^{tt}\right)\cot(\beta_{te}) \tag{3.14}$$

The first four equations do not depend on x_2^{tt}, so equation (3.14) will be used only after the other four variables are found. Equation (3.11) is easily eliminated, resulting in the following nonlinear system of equations in three unknowns, x_2^{lt}, y_1^{lt}, x_1^{tt},

$$\begin{cases} 2r_{le}x_2^{lt} + 3\left(y_1^{lt}\right)^2 = 0 \\ 2\left(y_1^{lt} - y_t\right) - 3\kappa_t\left(x_1^{tt} - x_t\right)^2 = 0. \\ x_2^{lt} + x_1^{tt} - 2x_t = 0 \end{cases} \tag{3.15}$$

This can be reduced to the following fourth order polynomial equation in x_2^{lt},

$$\begin{aligned} &\frac{27}{4}\kappa_t{}^2\left(x_2^{lt}\right)^4 - 27\kappa_t{}^2x_t\left(x_2^{lt}\right)^3 + \left(9\kappa_t y_t + \frac{81}{2}\kappa_t{}^2x_t{}^2\right)\left(x_2^{lt}\right)^2 \\ &+\left(2r_{le} - 18\kappa_t x_t y_t - 27\kappa_t{}^2x_t{}^3\right)\left(x_2^{lt}\right) + \left(3y_t{}^2 + 9\kappa_t x_t{}^2 y_t + \frac{27}{4}\kappa_t{}^2x_t{}^4\right) = 0 \end{aligned}, \tag{3.16}$$

which is solved numerically as discussed below.

Letting $x_2^{lt} = b_9$ represent the root of (3.16), the remaining control point variables are known from (3.15),

$$x_1^{tt} = 2x_t - b_9, \tag{3.17}$$

$$y_1^{lt} = \frac{3}{2}\kappa_t\left(x_t - b_9\right)^2 + y_t. \tag{3.18}$$

It remains to solve for x_2^{lt}. The polynomial in (3.16) will normally have multiple real roots, so we must consider any bounds on x_2^{lt}. For an aerodynamic shape, the leading edge of the thickness profile must be concave down and curve from (0,0) to (x_t, y_t), the position of maximum thickness. Thus the leading edge radius must be negative, and, from (3.7),

$$x_2^{lt} > 0.$$

At the juncture between the curves, the thickness profile should be travelling left to right to avoid a loop. That is, $\left(x^{lt}\right)'(1) > 0$, where $\left(x^{lt}\right)'(1) = 3\left(x_t - x_2^{lt}\right)$ from (3.3), (3.8). Thus

$$x_2^{lt} < x_t .$$

Since a blade's thickness must be positive, the leading edge must travel upwards. That is, $\left(y^{lt}\right)'(0) > 0$, where $\left(y^{lt}\right)'(0) = 3y_1^{lt}$ from (3.3) and (3.5). Thus $y_1^{lt} > 0$. From (3.18), this implies that $\frac{3}{2}\kappa_t\left(x_t - x_2^{lt}\right)^2 + y_t > 0$. Since $\kappa_t < 0$, this is equivalent to $\left(x_t - x_2^{lt}\right)^2 < \frac{-2y_t}{3\kappa_t}$.

Since it is already required that $x_2^{lt} < x_t$, it follows that

$$x_2^{lt} > x_t - \sqrt{\frac{-2y_t}{3\kappa_t}} .$$

Thus the root $x_2^{lt} = b_9$ of (3.16) must satisfy

$$\max\left\{0, x_t - \sqrt{\frac{-2y_t}{3\kappa_t}}\right\} < b_9 < x_t . \qquad (3.19)$$

Within these bounds, the solution of (3.16) proceeds as follows. Equation (3.16) can be written as $P_4(x) = 0$, where $x = x_2^{lt} = b_9$, and $P_4(x)$ is a fourth-order polynomial. The first derivative and critical values of $P_4(x)$ are computed symbolically using Maple. The critical points and intervals of increase and decrease of $P_4(x)$ are used to determine whether $P_4(x)$ has any real roots. If there is no real root, the code is informed that the aerodynamic parameters resulted in a non-aerodynamic shape, and the flow is not calculated. If there are real roots, the critical points are used to partition the real line into intervals, within each of which exactly one root of $P_4(x)$ exists. The roots are then found by numerical means. If more than one valid root exists, the smallest one is chosen as it will result in the smoothest juncture between the leading and trailing curves, and thus the most aerodynamic shape.

In summary, the control points for the leading thickness curve are given by

$$\begin{cases} x_0^{lt} = 0 \\ x_1^{lt} = 0 \\ x_2^{lt} = b_9 \\ x_3^{lt} = x_t \end{cases} \qquad \begin{cases} y_0^{lt} = 0 \\ y_1^{lt} = \dfrac{3}{2}\kappa_t(x_t - b_9)^2 + y_t \\ y_2^{lt} = y_t \\ y_3^{lt} = y_t \end{cases}. \tag{3.20}$$

The control points for the trailing thickness curve are given by

$$\begin{cases} x_0^{tt} = x_t \\ x_1^{tt} = 2x_t - b_9 \\ x_2^{tt} = 1 + \left[dz_{te} - \left(\dfrac{3}{2}\kappa_t(x_t - b_9)^2 + y_t\right)\right]\cot(\beta_{te}) \\ x_3^{tt} = 1 \end{cases} \qquad \begin{cases} y_0^{tt} = y_t \\ y_1^{tt} = y_t \\ y_2^{tt} = \dfrac{3}{2}\kappa_t(x_t - b_9)^2 + y_t \\ y_3^{tt} = dz_{te} \end{cases} \tag{3.21}$$

where b_9 is the left-most root of (3.16) that lies within the bounds (3.19).

3.2.3 Leading Camber Curve (Degree 3): (x_i^{lc}, y_i^{lc})

Since the leading edge is at (0,0), $x^{lc}(0) = y^{lc}(0) = 0$, which together with (3.2) implies

$$x_0^{lc} = y_0^{lc} = 0. \tag{3.22}$$

The angle of the camber curve at the leading edge is γ_{le}, so the slope at the leading edge must equal $\tan(\gamma_{le})$. That is,

$$\left.\frac{dy}{dx}\right|_{x=0} = \frac{\left(y^{lc}\right)'(0)}{\left(x^{lc}\right)'(0)} = \tan(\gamma_{le}),$$

where, from (3.3) and (3.22), $\left(y^{lc}\right)'(0) = -3y_0^{lc} + 3y_1^{lc} = 3y_1^{lc}$ and $\left(x^{lc}\right)'(0) = -3x_0^{lc} + 3x_1^{lc} = 3x_1^{lc}$. Letting $y_1^{lc} = b_1$ results in

$$\begin{cases} x_1^{lc} = b_1 \cot(\gamma_{le}) \\ y_1^{lc} = b_1 \end{cases}, \tag{3.23}$$

where a solution for b_1 will be given in the next section.

The juncture between the leading and trailing curves has coordinates $x^{lc}(1) = x_c$, $y^{lc}(1) = y_c$ and is horizontal (i.e. $\left(y^{lc}\right)'(1) = 0$). Thus, from (3.2) and (3.3),

$$\begin{cases} x_3^{lc} = x_c \\ y_3^{lc} = y_3^{lc} = y_c \end{cases}. \tag{3.24}$$

The curvature at the juncture between curves is given by

$$\kappa_c = \frac{\left(y^{lc}\right)''(1)}{\left[\left(x^{lc}\right)'(1)\right]^2} = \frac{6y_1^{lc} - 12y_2^{lc} + 6y_3^{lc}}{\left[-3x_2^{lc} + 3x_3^{lc}\right]^2},$$

from (3.3), (3.4). Substituting the control points already known from (3.23) and (3.24), the result is a quadratic equation in the remaining unknown, x_2^{lc},

$$\left(x_2^{lc}\right)^2 - 2x_c\left(x_2^{lc}\right) + x_c^{\,2} - \frac{2(b_1 - y_c)}{3\kappa_c} = 0.$$

This has two solutions,

$$x_2^{lc} = x_c \pm \sqrt{\frac{2(b_1 - y_c)}{3\kappa_c}},$$

but only one which will result in an aerodynamic shape. Bezier curves are tangent at their right endpoint to the line segment joining the last two control points. Consequently, if the x-coordinate of the second last control point is greater than that of the last, the curve will be multi-valued. That is, the camber curve would have a loop in it at the joint. Thus we require that $x_2^{lc} < x_3^{lc} = x_c$. With this restriction, we have

$$x_2^{lc} = x_c - \sqrt{\frac{2(b_1 - y_c)}{3\kappa_c}}. \tag{3.25}$$

In summary, the control points for the leading camber curve are given by

$$\begin{cases} x_0^{lc} = 0 \\ x_1^{lc} = b_1 \cot(\gamma_{le}) \\ x_2^{lc} = x_c - \sqrt{\dfrac{2(b_1 - y_c)}{3\kappa_c}} \\ x_3^{lc} = x_c \end{cases} \qquad \begin{cases} y_0^{lc} = 0 \\ y_1^{lc} = b_1 \\ y_2^{lc} = y_c \\ y_3^{lc} = y_c \end{cases}, \tag{3.26}$$

where b_1 is given by (3.33) with bounds (3.34), calculated in the subsequent section.

3.2.4 Trailing Camber Curve (Degree 3): (x_i^{tc}, y_i^{tc})

The juncture between the leading and trailing curves has coordinates $x^{tc}(0) = x_c$, $y^{tc}(0) = y_c$ and is horizontal (i.e. $\left(y^{tc}\right)'(0) = 0$). Thus, from (3.2) and (3.3),

$$\begin{cases} x_0^{tc} = x_c \\ y_0^{tc} = y_1^{tc} = y_c \end{cases}. \tag{3.27}$$

The curvature at the juncture should be the same for both leading and trailing curves. That is,

$$\frac{\left(y^{lc}\right)''(1)}{\left[\left(x^{lc}\right)'(1)\right]^2} = \frac{\left(y^{tc}\right)''(0)}{\left[\left(x^{tc}\right)'(0)\right]^2}. \tag{3.28}$$

To satisfy this requirement, it is sufficient that $\left(y^{lc}\right)''(1) = \left(y^{tc}\right)''(0)$ and $\left(x^{lc}\right)'(1) = \left(x^{tc}\right)'(0)$. Consider first y''. Using (3.4) with control points (3.26), (3.27),

$$\begin{aligned} \left(y^{lc}\right)''(1) &= 6y_1^{lc} - 12y_2^{lc} + 6y_3^{lc} \\ &= 6b_1 - 6y_c \end{aligned},$$

and

$$\begin{aligned} \left(y^{tc}\right)''(0) &= 6y_0^{tc} - 12y_1^{tc} + 6y_2^{tc} \\ &= 6y_2^{tc} - 6y_c \end{aligned}.$$

Equating these second derivatives and solving for y_2^{tc} results in

$$y_2^{tc} = b_1. \tag{3.29}$$

Now consider x'. Using (3.3) with control points (3.26), (3.27),

$$\begin{aligned}\left(x^{lc}\right)'(1) &= -3x_2^{lc} + 3x_3^{lc} \\ &= -3\left(x_c - \sqrt{\frac{2(b_1 - y_c)}{3\kappa_c}}\right) + 3x_c\end{aligned},$$

and

$$\begin{aligned}\left(x^{tc}\right)'(0) &= -3x_0^{tc} + 3x_1^{tc} \\ &= -3x_c + 3x_1^{tc}\end{aligned}.$$

Equating these first derivatives and solving for x_1^{tc} results in

$$x_1^{tc} = x_c + \sqrt{\frac{2(b_1 - y_c)}{3\kappa_c}}. \tag{3.30}$$

Since the trailing edge is at $(1, z_{te})$, $x^{tc}(1) = 1$ and $y^{tc}(1) = z_{te}$, which from (3.2) implies

$$\begin{cases} x_3^{tc} = 1 \\ y_3^{tc} = z_{te} \end{cases}. \tag{3.31}$$

The trailing edge angle of the camber curve is α_{te}, where α_{te} is taken to be positive, so the slope at the trailing edge must equal $-\tan(\alpha_{le})$. That is,

$$\left.\frac{dy}{dx}\right|_{x=1} = \frac{\left(y^{tc}\right)'(1)}{\left(x^{tc}\right)'(1)} = -\tan(\alpha_{te}),$$

where

$$\begin{aligned}\left(y^{tc}\right)'(1) &= -3y_2^{tc} + 3y_3^{tc} \\ &= -3b_1 + 3z_{te}\end{aligned}$$

from (3.3), (3.29), (3.31), and

$$\begin{aligned}\left(x^{tc}\right)'(1) &= -3x_2^{tc} + 3x_3^{tc} \\ &= -3x_2^{tc} + 3\end{aligned}$$

from (3.3), (3.31). Solving for x_2^{tc} results in

$$x_2^{tc} = 1 + \left(z_{te} - b_1\right)\cot\left(\alpha_{te}\right). \tag{3.32}$$

Finally, it is possible to solve for b_1. When matching the curvature (3.28), it was sufficient to match y'' and x' at the juncture. Matching additionally the second derivative of x will result in an equation for b_1, while maintaining or improving the smoothness of the juncture. Using (3.4) with control points (3.26),

$$\begin{aligned}\left(x^{lc}\right)''(1) &= 6x_1^{lc} - 12x_2^{lc} + 6x_3^{lc} \\ &= 6b_1\cot\gamma_{le} + 4\sqrt{6\frac{b_1 - y_c}{\kappa_c}} - 6x_c\end{aligned}$$

and with control points (3.27), (3.30), (3.32),

$$\begin{aligned}\left(x^{tc}\right)''(0) &= 6x_0^{tc} - 12x_1^{tc} + 6x_2^{tc} \\ &= -6b_1\cot\alpha_{te} - 4\sqrt{6\frac{b_1 - y_c}{\kappa_c}} - 6x_c + 6z_{te}\cot\alpha_{te} + 6\end{aligned}$$

Equating these second derivatives of x results in the following expression,

$$b_1\left(\cot\gamma_{le} + \cot\alpha_{te}\right) + 8\sqrt{\frac{b_1 - y_c}{6\kappa_c}} - z_{te}\cot\alpha_{te} - 1 = 0,$$

for which b_1 is the only unknown. This has two solutions,

$$b_1 = \frac{1}{3\kappa_c(\cot\gamma_{le} + \cot\alpha_{te})^2}\Big[16 + 3\kappa_c(\cot\gamma_{le} + \cot\alpha_{te})(1 + z_{te}\cot\alpha_{te}) \pm 4\sqrt{16 + 6\kappa_c(\cot\gamma_{le} + \cot\alpha_{te})(1 - y_c(\cot\gamma_{le} + \cot\alpha_{te}) + z_{te}\cot\alpha_{te})}\Big] . \tag{3.33}$$

Upper and lower bounds can be placed on solutions (3.33). For aerodynamic reasons, the curvature at the juncture must be negative. Otherwise, the camber will have a valley at the juncture, creating a region of separation in the middle of the blade. Thus, since $\kappa_c < 0$, x_2^{lc} (3.25) and x_1^{tc} (3.30) exist only if $b_1 < y_c$. Furthermore, consideration of equations (3.23) implies that b_1 must be positive. If it is zero, the leading edge angle is ignored. If it is negative, the camber curve begins in the wrong direction. Thus the following restriction must be placed on solutions (3.33),

$$0 < b_1 < y_c . \tag{3.34}$$

In practice, both solutions of (3.33) are calculated to determine which one is valid. If the solutions are not real, or if neither falls within the bounds (3.34), the code is informed that the aerodynamic parameters resulted in a non-aerodynamic shape, and the flow is not calculated. If both solutions are within the interval, the value chosen is that closest to y_c, as this will result in the smoothest juncture between the curves.

In summary, the control points for the trailing edge camber curve are given by

$$\begin{cases} x_0^{tc} = x_c \\ x_1^{tc} = x_c + \sqrt{\dfrac{2(b_1 - y_c)}{3\kappa_c}} \\ x_2^{tc} = 1 + (z_{te} - b_1)\cot(\alpha_{te}) \\ x_3^{tc} = 1 \end{cases} \qquad \begin{cases} y_0^{tc} = y_c \\ y_1^{tc} = y_c \\ y_2^{tc} = b_1 \\ y_3^{tc} = z_{te} \end{cases}, \tag{3.35}$$

where b_1 is given by (3.33) with bounds (3.34).

A possible limitation of the BP 3333 parameterization is the size of the design space - that is the number of airfoils that can be reproduced. Frequently airfoils are designed with inflection points on their trailing thickness and/or trailing camber curves. In the BP 3333 parameterization, the inflection point positions are entirely dependent on the other parameters. Increasing the degree of the trailing edge curves would allow more freedom for these positions. Also, in order to match the curvature, it is necessary to choose one of (possibly) multiple roots of equations relating the curvature to other parameters. Airfoil shapes formed by other roots are never considered, yet they might be better designs. The BP 3434 parameterization is an attempt to address these possible deficiencies.

3.3 BP 3434 Parameterization

In the BP 3434 parameterization, the design space is enlarged by increasing the trailing curve degrees from three to four. This requires two additional variables, b_{15} and b_{17} (each the x-value of the fourth control point on their respective curve). The thickness curvature is replaced by the Bezier control parameter b_8 (the height of the second control point). The camber curvature is replaced by two Bezier parameters, b_0, b_2, for a total of 15 BP parameters. These are illustrated in Figure 3.3. Equations for the associated Bezier control points are derived in sections 3.3.1 - 3.3.4.

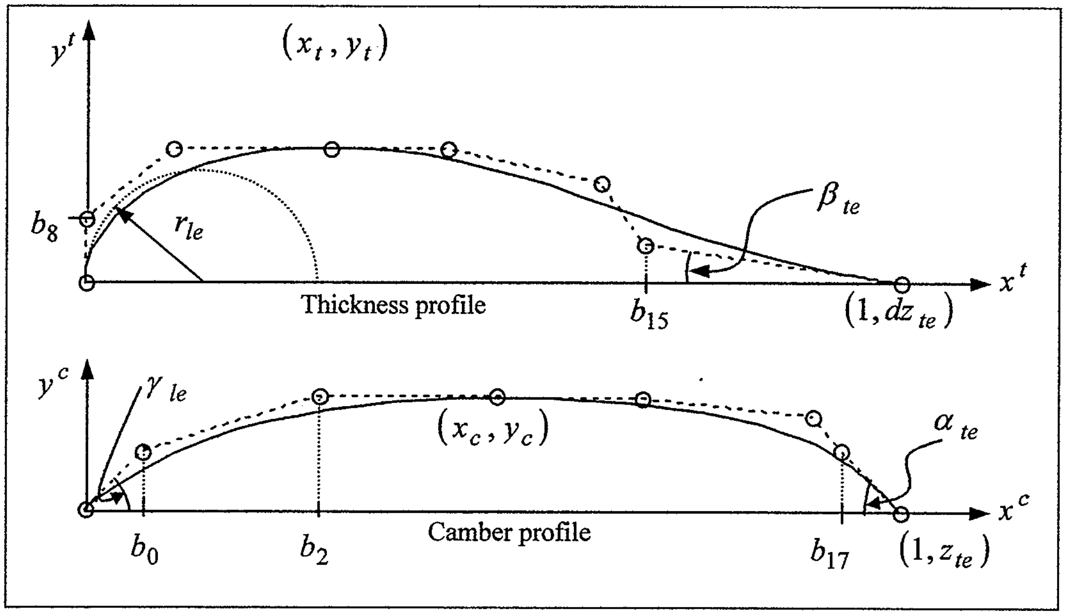


Figure 3.3 *BP 3434 airfoil geometry and Bezier control points defined by ten aerodynamic and five Bezier parameters.*

3.3.1 Leading Thickness Curve (Degree 3): $\left(x_i^{lt}, y_i^{lt}\right)$

The degree three Bezier curves and derivatives are given by (3.2) – (3.4). The leading edge begins at (0,0) and is vertical. Thus

$$x_0^{lt} = y_0^{lt} = x_1^{lt} = 0. \tag{3.36}$$

The curve ends at (x_t, y_t) and is horizontal there, so that

$$\begin{cases} x_3^{lt} = x_t \\ y_2^{lt} = y_3^{lt} = y_t \end{cases}. \tag{3.37}$$

The leading edge radius is given by

$$r_{le} = \frac{\left[\left(y^{lt}\right)'(0)\right]^2}{-\left(x^{lt}\right)''(0)} = \frac{\left[-3y_0^{lt} + 3y_1^{lt}\right]^2}{-6x_0^{lt} + 12x_1^{lt} - 6x_2^{lt}},$$

which from (3.36) reduces to

$$-2r_{le}x_2^{lt} = 3\left(y_1^{lt}\right)^2. \qquad (3.38)$$

Thus we set y_1^{lt} equal to the Bezier variable b_8, which replaces the curvature variable κ_t of the BP 3333 parameterization. Some restrictions must be placed on b_8 to ensure an aerodynamic shape. For the blade to have positive thickness, we must have $b_8 > 0$. For the crest curvature to be negative (concave down), it is necessary that $b_8 < y_t$. Finally, to avoid a loop at the crest, we must have $x_2^{lt} < x_t$, which from (3.38) implies that $b_8 < \sqrt{-2r_{le}x_t/3}$. Thus the following restrictions are placed on b_8:

$$0 < b_8 < \min\left(y_t, \sqrt{-2r_{le}x_t/3}\right) \qquad (3.39)$$

In summary, the following equations determine the Bezier control points of the leading thickness curve,

$$\begin{cases} x_0^{lt} = 0 \\ x_1^{lt} = 0 \\ x_2^{lt} = \dfrac{-3b_8{}^2}{2r_{le}} \\ x_3^{lt} = x_t \end{cases} \qquad \begin{cases} y_0^{lt} = 0 \\ y_1^{lt} = b_8 \\ y_2^{lt} = y_t \\ y_3^{lt} = y_t \end{cases}, \qquad (3.40)$$

where b_8 must satisfy (3.39).

3.3.2 Trailing Thickness Curve (Degree 4): $\left(x_i^{tt}, y_i^{tt}\right)$

From (2.1), the degree four Bezier curve is

$$x(u) = x_0(1-u)^4 + 4x_1u(1-u)^3 + 6x_2u^2(1-u)^2 + 4x_3u^3(1-u) + x_4u^4, \qquad (3.41)$$

with first derivative

$$\begin{aligned} x'(u) = &\left(4u^3 - 12u^2 + 12u - 4\right)x_0 + \left(-16u^3 + 36u^2 - 24u + 4\right)x_1 \\ &+ \left(24u^3 - 36u^2 + 12u\right)x_2 + \left(-16u^3 + 12u^2\right)x_3 + 4t^3x_4 \end{aligned}, \qquad (3.42)$$

and second derivative

$$\begin{aligned} x''(u) = &\left(12u^2 - 24u + 12\right)x_0 + \left(-48u^2 + 72u - 24\right)x_1 \\ &+ \left(72u^2 - 72u + 12\right)x_2 + \left(-48u^2 + 24u\right)x_3 + 12u^2x_4 \end{aligned}. \qquad (3.43)$$

The trailing curve begins at (x_t, y_t) and is horizontal there, so that

$$\begin{cases} x_0^{tt} = x_t \\ y_0^{tt} = y_1^{tt} = y_t \end{cases}. \qquad (3.44)$$

The curvature at the juncture should be the same for both leading and trailing curves. That is,

$$\frac{\left(y^{lt}\right)''(1)}{\left[\left(x^{lt}\right)'(1)\right]^2} = \frac{\left(y^{tt}\right)''(0)}{\left[\left(x^{tt}\right)'(0)\right]^2}.$$

To satisfy this requirement, it is sufficient that $\left(y^{lt}\right)''(1) = \left(y^{tt}\right)''(0)$ and $\left(x^{lt}\right)'(1) = \left(x^{tt}\right)'(0)$. From (3.3), (3.4), and (3.40),

$$\left(y^{lt}\right)''(1) = 6b_8 - 6y_t \text{ and } \left(x^{lt}\right)'(1) = -3x_2^{lt} + 3x_t\,,$$

while from (3.42) - (3.44),

$$\left(y^{tt}\right)''(0) = -12y_t + 12y_2^{tt} \text{ and } \left(x^{tt}\right)'(0) = -4x_t + 4x_1^{tt}\,.$$

Thus, for second order continuity at the juncture, we require

$$\begin{cases} y_2^{tt} = \dfrac{y_t + b_8}{2} \\ x_1^{tt} = \dfrac{7x_t - 3x_2^{lt}}{4} \end{cases}, \tag{3.45}$$

where x_2^{lt} is given in (3.40). Matching additionally the second derivative of x will reduce by one the number of parameters, while maintaining or improving the smoothness of the juncture. From (3.4) and (3.40),

$$\left(x^{lt}\right)''(1) = -12x_2^{lt} + 6x_t\,,$$

while from (3.43) and (3.44),

$$\left(x^{tt}\right)''(0) = 12x_t - 24x_1^{tt} + 12x_2^{tt}$$

which together with (3.45) imply that

$$x_2^{tt} = 3x_t - \frac{5}{2}x_2^{lt}\,. \tag{3.46}$$

The trailing edge coordinates are

$$\begin{cases} x_4^{tt} = 1 \\ y_4^{tt} = dz_{te} \end{cases}. \tag{3.47}$$

The wedge angle satisfies

$$\frac{\left(y^{tt}\right)'(1)}{\left(x^{tt}\right)'(1)} = -\tan(\beta_{le}),$$

where from (3.42),

$$\left(y^{tt}\right)'(1) = -4y_3^{tt} + 4dz_{te} \quad \text{and} \quad \left(x^{tt}\right)'(1) = -4x_3^{tt} + 4.$$

Thus

$$y_3^{tt} = dz_{te} + \left(1 - x_3^{tt}\right)\tan(\beta_{te}). \tag{3.48}$$

Finally, let

$$x_3^{tt} = b_{15} \tag{3.49}$$

be a free parameter to allow for a flexible inflection point on the trailing thickness curve.

In summary, the trailing thickness Bezier control points are given by

$$\begin{cases} x_0^{tt} = x_t \\ x_1^{tt} = \dfrac{7x_t - 3x_2^{lt}}{4} \\ x_2^{tt} = 3x_t - \dfrac{5}{2}x_2^{lt} \\ x_3^{tt} = b_{15} \\ x_4^{tt} = 1 \end{cases} \qquad \begin{cases} y_0^{tt} = y_t \\ y_1^{tt} = y_t \\ y_2^{tt} = \dfrac{y_t + b_8}{2} \\ y_3^{tt} = dz_{te} + (1 - b_{15})\tan(\beta_{te}) \\ y_4^{tt} = dz_{te} \end{cases} \tag{3.50}$$

3.3.3 Leading Camber Curve (Degree 3): $\left(x_i^{lc}, y_i^{lc}\right)$

Two Bezier variables determine the x-coordinates of the second and third Bezier control points:

$$\begin{cases} x_1^{lc} = b_0 \\ x_2^{lc} = b_2 \end{cases}. \tag{3.51}$$

The remaining leading camber control points are uniquely determined by the leading edge position (0,0), the crest position, (x_c, y_c), and the leading edge angle γ_{le}, resulting in the following set of equations,

$$\begin{cases} x_0^{lc} = 0 \\ x_1^{lc} = b_0 \\ x_2^{lc} = b_2 \\ x_3^{lc} = x_c \end{cases} \qquad \begin{cases} y_0^{lc} = 0 \\ y_1^{lc} = b_0 \tan(\gamma_{le}) \\ y_2^{lc} = y_c \\ y_3^{lc} = y_c \end{cases}. \tag{3.52}$$

3.3.4 Trailing Camber Curve (Degree 4): (x_i^{tc}, y_i^{tc})

The position of the crest dictates the initial control points,

$$\begin{cases} x_0^{tc} = x_c \\ y_0^{tc} = y_1^{tc} = y_c \end{cases}. \tag{3.53}$$

To enforce a second order continuous juncture, we set $\left(x^{lc}\right)'(1) = \left(x^{tc}\right)'(0)$, $\left(x^{lc}\right)''(1) = \left(x^{tc}\right)''(0)$, and $\left(y^{lc}\right)''(1) = \left(y^{tc}\right)''(0)$. From (3.3), (3.4), and (3.52), the leading edge derivatives at the camber crest are:

$$\begin{cases} \left(x^{lc}\right)'(1) = -3b_2 + 3x_c \\ \left(x^{lc}\right)''(1) = 6b_0 - 12b_2 + 6x_c \\ \left(y^{lc}\right)''(1) = 6b_0 \tan(\gamma_{le}) - 6y_c \end{cases}.$$

These must match the trailing edge values at the crest, which from (3.42), (3.43), and (3.53) are:

$$\begin{cases} \left(x^{tc}\right)'(0) = -4x_c + 4x_1^{tc} \\ \left(x^{tc}\right)''(0) = 12x_c - 24x_1^{tc} + 12x_2^{tc} \\ \left(y^{tc}\right)''(0) = -12y_c + 12y_2^{tc} \end{cases}.$$

This results in the following equations

$$\begin{cases} x_1^{tc} = \dfrac{7x_c - 3b_2}{4} \\ x_2^{tc} = \dfrac{b_0 - 5b_2 + 6x_c}{2} \\ y_2^{tc} = \dfrac{y_c + b_1}{2} \end{cases}. \tag{3.54}$$

The trailing edge coordinates are

$$\begin{cases} x_4^{tc} = 1 \\ y_4^{tc} = z_{te} \end{cases}. \tag{3.55}$$

The trailing direction angle requires that

$$y_3^{tc} = z_{te} + \left(1 - x_3^{tc}\right)\tan\left(\alpha_{te}\right). \tag{3.56}$$

Finally, let

$$x_3^{tc} = b_{17} \tag{3.57}$$

be a free parameter to allow for a flexible inflection point on the trailing camber curve.

In summary, the trailing camber Bezier control points are given by

$$\begin{cases} x_0^{tc} = x_c \\ x_1^{tc} = \dfrac{3x_c - y_c \cot(\gamma_{le})}{2} \\ x_2^{tc} = \dfrac{-8y_c \cot(\gamma_{le}) + 13x_c}{6} \\ x_3^{tc} = b_{17} \\ x_4^{tc} = 1 \end{cases} \qquad \begin{cases} y_0^{tc} = y_c \\ y_1^{tc} = y_c \\ y_2^{tc} = \dfrac{5y_c}{6} \\ y_3^{tc} = z_{te} + (1 - b_{17}) \tan(\alpha_{te}) \\ y_4^{tc} = z_{te} \end{cases} . \qquad (3.58)$$

3.4 Airfoil Representation

A useful parameterization method must have the ability to represent a wide range of airfoils. This has an obvious influence on the robustness of a design algorithm. It cannot be truly robust if the design space is so small that the optimal shape cannot be represented.

In this section, the representation abilities of the Bezier and BP methods are evaluated and compared. Each parameterization is used to reproduce 63 known airfoils: 20 NACA symmetric airfoils, 20 NACA asymmetric airfoils (Abbott and von Doenhoff, 1959), 15 Eppler airfoils (Eppler, 1990), and 8 low-speed airfoils (Selig et. al., 1995). Plots for each parameterization are given in Appendix B on the CD.

In section 3.4.1, the optimization method used to reproduce the airfoils is presented. This includes a discussion of the error threshold and of any constraints placed on the curves. In sections 3.4.2 – 3.4.5, the results for each category of airfoil are summarized. Success rates are given, and any failures are examined in more detail.

3.4.1 Method of Airfoil Reproduction

The design algorithm was modified to include the option of matching a known airfoil. The cost function is thus the deviation between the target airfoil and the current parameterization. At each data point along the target, the vertical difference is measured, using linear interpolation to estimate the height of the current shape. The total deviation is measured using the $\mathcal{L}_2$-error norm. DE is then used to minimize that cost.

Table 3.1 *Bounds on the initial population used by DE to represent the airfoils.*

Bezier parameterization			BP parameterizations		
Parameter	Lower	Upper	Parameter	Lower	Upper
b_0	0.13	0.22	γ_{le}	0.05	0.1
b_1	0.08	0.17	b_0	0.01	0.1
b_2	0.28	0.37	b_2	0.1	0.3
b_3	0.13	0.22	x_c	0.2	0.5
b_4	0.43	0.52	y_c	0	0.2
b_5	0.58	0.67	κ_c	-0.2	0
b_6	0.76	0.85	b_{17}	0	0.9
b_7	0.03	0.12	z_{te}	0	0.01
b_8	0.01	0.1	α_{te}	0.05	0.1
b_9	0.18	0.27	r_{le}	-0.04	-0.001
b_{10}	0.06	0.15	b_8	0	0.7
b_{11}	0.38	0.47	x_t	0.15	0.4
b_{12}	0.58	0.67	y_t	0.05	0.15
b_{13}	0.78	0.87	κ_t	-0.5	0
b_{14}	0	0.09	b_{15}	0	0.9
			dz_{te}	0	0.001
			β_{te}	0.001	0.3

The optimizer specifications were kept fairly consistent for all tests. The DE variation rand-to-best/1/exp was used for all tests, with F=0.85, CR=1. In general, the population size was NP=150. A maximum of 500 generations was allowed. The initial population was chosen at random within the bounds shown in Table 3.1. If convergence did not occur within 500 generations, additional runs were performed with different population sizes. In a few cases, when a different population size did not improve convergence, the initial population was modified to more closely resemble the target.

To ensure feasibility of the study, a cost limit of 0.01 was used in most cases. That is, optimization was halted if the deviation between the parameterization and the target was smaller than 0.01. To ensure complete convergence, it is preferable to allow the objective function value to become nearly constant over 10,000 to 20,000 function evaluations. However, this is not practical in a study of this size (63 different targets, each parameterized three different ways.) In a few cases, optimization was allowed to progress beyond the error limit, to test the ability of the parameterizations to converge to a smaller deviation.

The 0.01 limit is reasonable in light of the actual error that exists between a manufactured blade and its theoretical design. For example, in a study of 34 low-speed airfoils (Selig et. al., 1995), models with 12-inch chord length were built for wind tunnel tests. The deviation of the model from the theoretical airfoil was measured at each data point and averaged over the entire blade. The smallest such average deviation was 0.0017 in, the largest 0.0384 in, and the average over all 34 models was 0.0092 in. For an airfoil of unit chord length, these measurements correspond to the following (non-dimensional) average deviations: minimum 1.4×10^{-4}, maximum 3.2×10^{-3}, and average 7.7×10^{-4}. Compare with the BP 3333 parameterization (converged to an $\mathscr{l}_2$-error of 0.01) of the following airfoils: NACA 66_3-218, average deviation 1.1×10^{-3}; Low-speed FX 63137, average deviation 7.9×10^{-4}. These are reasonably close to the average — and well within the upper bound — of the deviations in the low speed study.

The constraints imposed include the one-to-one constraints discussed in section 2.2, although this had to be relaxed for some Eppler airfoils, in which the trailing camber is not one-to-one. For the Bezier parameterization, the minimum separation of juncture control points was also imposed. For the BP parameterizations, parameters were constrained within the bounds discussed above ((3.19), (3.34) for BP 3333, and (3.39) for BP 3434). If any constraint is violated, the shape is not compared to the target, and is not included in the NFEs reported.

3.4.2 NACA Symmetric Airfoils

In their classic reference book, *Theory of Wing Sections*, Abbot and Von Doenhoff (1959) summarize the NACA class of airfoils. These continue to be prevalent, not only for airplane wings, but also for high performance fans. The 20 airfoils shown in Figure 3.4 were selected to represent a broad range of symmetric NACA shapes.

Notice that the cost limit of 0.01 does not depend on the number of data points in the target. That is, if a target has fewer data points, the converged shape will have a larger average deviation. Most NACA airfoils have 52 data points, and low-speed sections can have as many as 98. However, six of the symmetric NACA airfoils contain only 34 data points. For these six — namely 0008-34, 0010-34, 0010-64, 0012-34, 16-006, and 16-018 — the limit was reduced to 0.005.

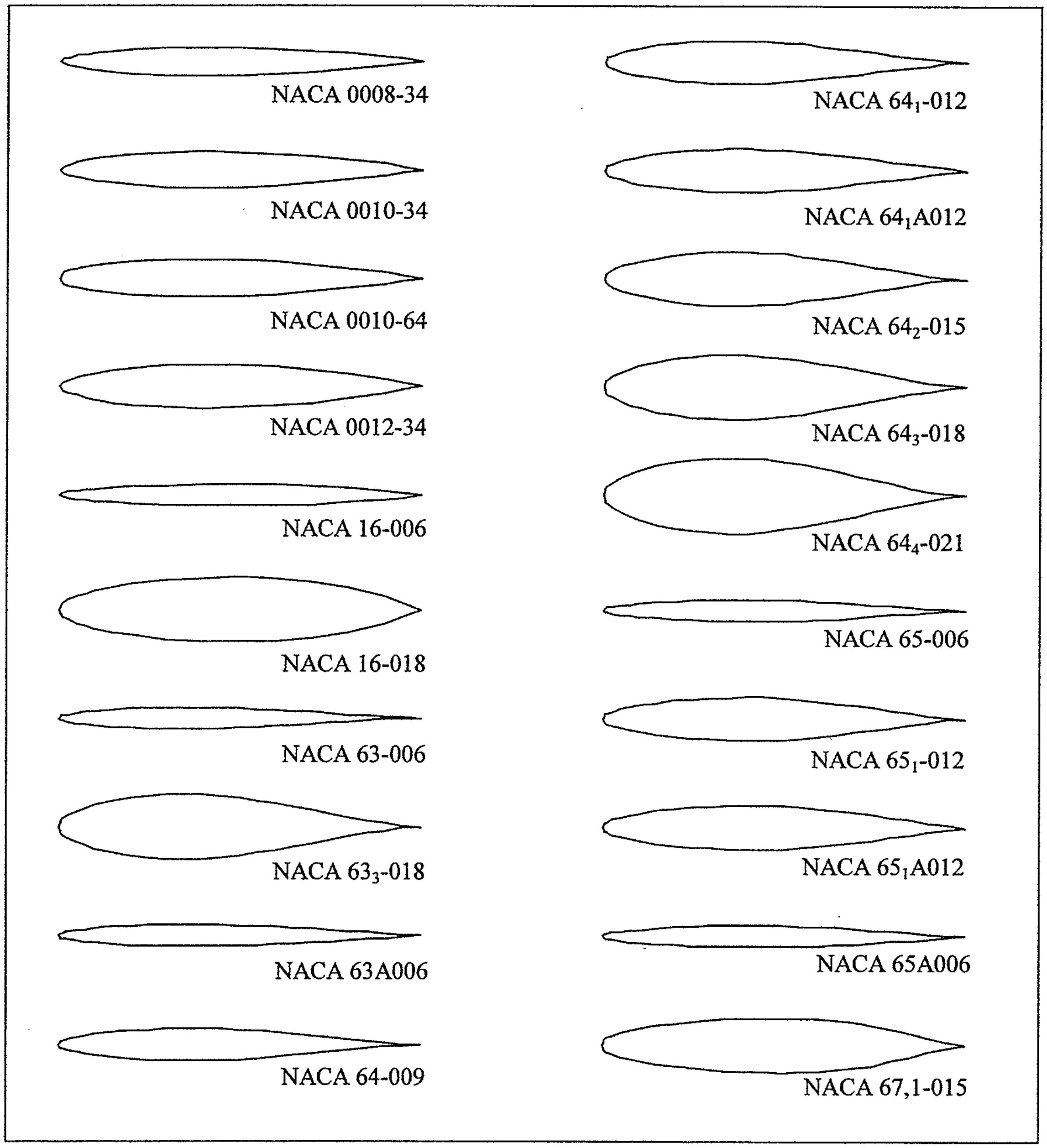


Figure 3.4 *Twenty symmetric NACA airfoils selected for representation.*

For each optimization using a BP parameterization, the trailing edge height, z_{te}, and thickness, dz_{te}, were both fixed at zero. Since symmetric curves have constant camber, all camber variables could have been fixed for each parameterization. However, to test the ability of DE to find symmetric shapes, they were left variable.

Each parameterization method was able to reproduce all 20 airfoils to within the specified limit. The final deviation and the number of function evaluations (NFEs) required are listed in Table 3.2.

Table 3.2 *NFEs required for convergence to symmetric NACA airfoils.*

Airfoil	Cost	Bezier	BP 3333	BP 3434
0008-34	0.005	3268	1832	2475
0010-34	0.005	2196	1939	2924
0010-64	0.005	2415	2471	4778
0012-34	0.005	3252	1721	3248
16-006	0.005	2662	5520	2563
16-018	0.005	3219	2767	3136
63-006	0.01	2078	4250	3007
63_3-018	0.01	2343	2404	3076
63A006	0.01	1598	2074	1901
64-009	0.01	2766	1836	3274
64_1-012	0.01	2089	2620	3061
64_1A012	0.01	2496	1459	1872
64_2-015	0.01	3256	2393	3334
64_3-018	0.01	2635	2536	2707
64_4-021	0.01	3338	3073	3196
65-006	0.01	1605	2705	2520
65_1-012	0.01	2020	1541	2665
65_1A012	0.01	2252	1228	2565
65A006	0.01	1715	2074	1777
67,1-015	0.01	3027	9510	3262
Converged:		100%	100%	100%
Avg NFEs:		2512	2798	2867

The NFEs required to find a reasonably close representation of the symmetric airfoils is remarkably small. Each parameterization averaged fewer than 3000, with only one requiring more than 6000 (the BP 3333 parameterization of NACA 67,1-015).

Compare this with the 50,000 function evaluations required for convergence to an aerodynamic target – using the Bezier parameterization – in the previous study. The task of matching a geometric shape is easier than that of inverse design.

Although all parameterization methods were successful, it is informative to examine one airfoil in detail. Consider the first one in Table 3.2, NACA 0008-34. To test the parameterizations further, the cost limit is lowered to 0.001. The cost, NFEs, and average deviation are listed in Table 3.3. For each parameterization, the average deviation is actually smaller than the minimum average deviation (1.4×10^{-4}) for the models built in Selig's low-speed study. (See section 3.4.1.) The actual deviations of the different representations are compared in Figure 3.5.

Table 3.3 *Convergence summary for NACA 0008-34, with error limit 0.001.*

	Bezier	BP 3333	BP 3434
Cost	7.26×10^{-4}	9.07×10^{-4}	9.90×10^{-4}
NFEs	6612	4243	6180
Avg. Deviation	1.06×10^{-4}	1.24×10^{-4}	1.26×10^{-4}

The thickness curves for NACA 4- and 5- digit airfoils have known analytic equations (see Abbot and von Doenhoff, 1959, p 117). Thus we can compare the aerodynamic quantities of the BP parameterizations with the actual quantities of the target. The NACA 0008-34 has the following thickness curve, with coefficients accurate to 10^{-5},

$$y^t(x) = \begin{cases} 0.05938\sqrt{x} + 0.07729x - 0.22326x^2 + 0.11328x^3, & 0 \le x \le 0.4 \\ 0.00080 + 0.12600(1-x) - 0.09333(1-x)^2 - 0.01296(1-x)^3, & 0.4 \le x \le 1 \end{cases},$$

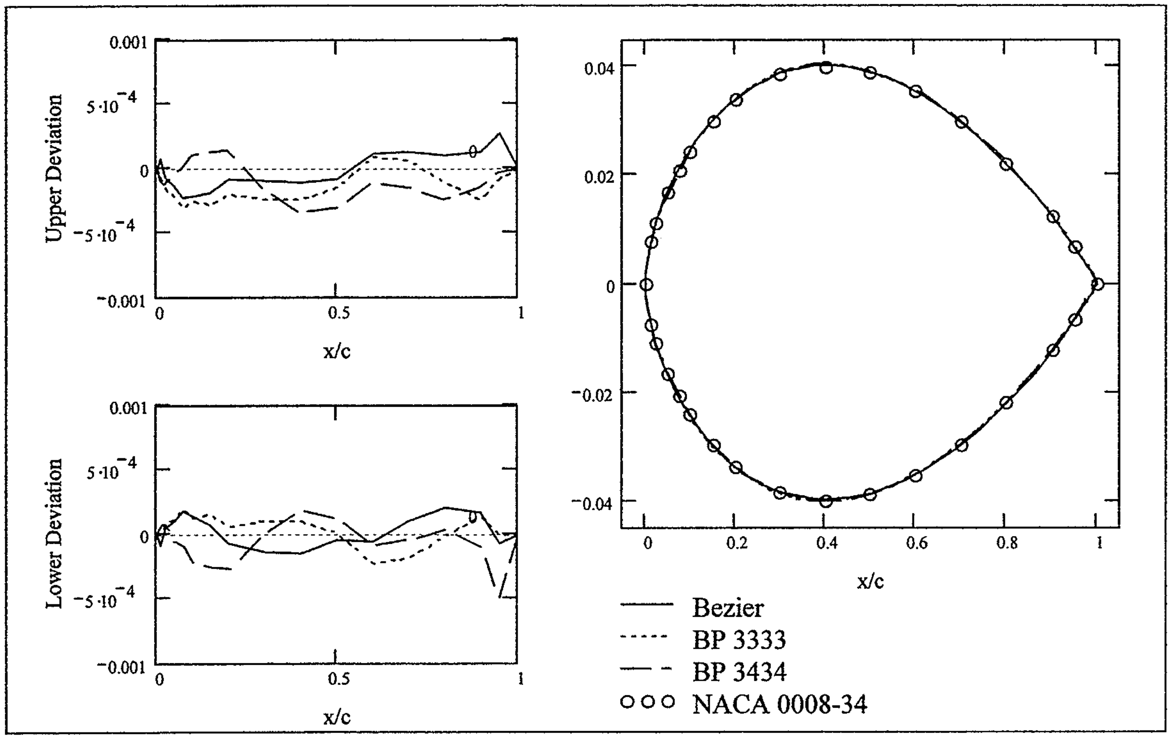


Figure 3.5 *Representation of NACA 0008-34 by the three parameterizations, including upper and lower curve deviations.*

from which the true aerodynamic quantities are calculated. These appear in Table 3.4, along with the parameters for each BP representation. For comparison, the crest curvature was calculated for the BP 3434 parameterization. The agreement for the crest position is within two significant digits, but the other quantities only agree to one significant digit.

Table 3.4 *BP parameterizations of NACA 0008-34 compared with known aerodynamic shape quantities.*

	0008-34	BP 3333	BP 3434
r_{le}	-0.00174	-0.00157	-0.00178
b_8			0.614
x_t	0.4	0.398	0.402
y_t	0.04	0.0402	0.0403
κ_t	-0.233	-0.264	-0.286
b_{15}			0.977
β_{te}	0.125	0.160	0.103

3.4.3 NACA Asymmetric Airfoils

All airfoils discussed above were symmetric and did not test the abilities of the camber parameterizations. In this section, 20 asymmetric NACA airfoils are chosen for representation (Figure 3.6). Again, these were selected to represent a broad range of

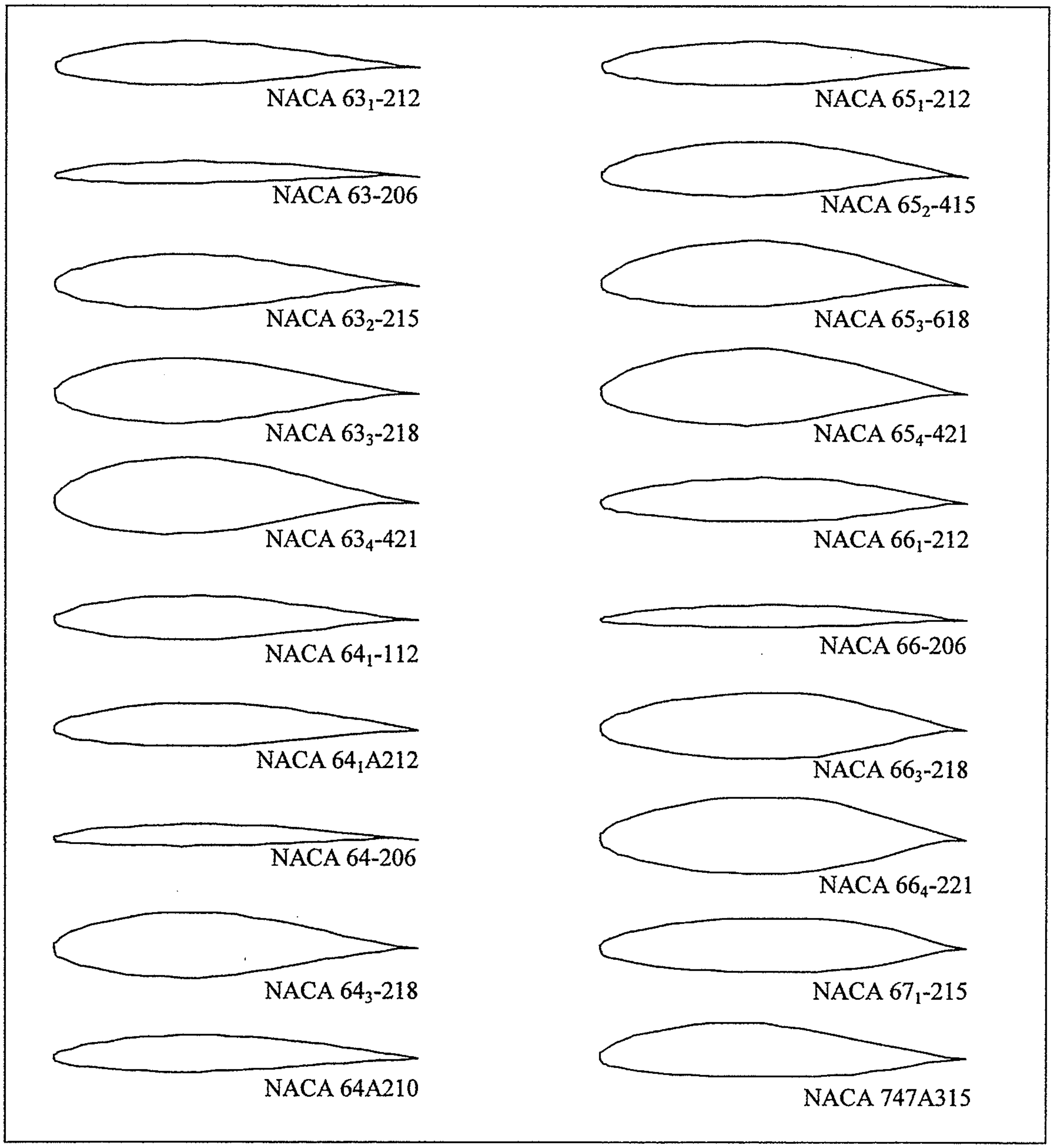


Figure 3.6 *Twenty asymmetric NACA airfoils selected for representation.*

shapes. The trailing edge height, z_{te}, and thickness, dz_{te}, were both fixed at zero. The cost limit was 0.01 for all airfoils.

A summary of convergence is given in Table 3.5. The Bezier and BP 3434 parameterizations were able to reproduce all 20 airfoils to within the cost limit. BP 3333 failed only with NACA 747A315 (cost 0.141). This failure will be examined in further detail below.

Table 3.5 *NFEs required for convergence (cost 0.01) to asymmetric NACA airfoils.*

Airfoil	Bezier	BP 3333	BP 3434
63_1-212	4664	1772	5394
63-206	1523	943	4161
63_2-215	4355	2116	3636
63_3-218	3278	2185	5009
63_4-421	3099	4989	7441
64_1-112	2262	4144	5123
64_1A212	4033	1585	5995
64-206	1765	3475	3772
64_3-218	8255	2880	5662
64A210	3347	1539	3559
65_1-212	4156	2236	3032
65_2-415	3498	3409	6162
65_3-618	2643	5519	3826
65_4-421	3321	4342	3908
66_1-212	2275	1290	2157
66-206	1431	1259	1980
66_3-218	3011	2567	4239
66_4-221	3309	2972	5173
67_1-215	2845	2411	2555
747A315	2968	*	5705
Converged:	100%	95%	100%
Avg NFEs:	3302	3071	4424

* Cost 0.014 after 500 generations

The NFEs required is larger than that for the symmetric airfoils, but is still quite small. Of the 59 converged representations, only two required significantly more than 6000 function evaluations (Bezier 64_3-218 and BP 3434 63_4-421).

For the first in the list (63_1-212), DE was allowed to converge completely (500 generations). Results are shown in Table 3.6. BP 3333 converges at a significantly higher deviation. Recall that the minimum average deviation in Selig's low-speed study was 1.4×10^{-4}, and the average was 7.7×10^{-4}. (See section 3.4.1.) The BP 3333 representation is still better than Selig's average, and the other two are close to Selig's minimum.

Table 3.6 *Convergence summary for NACA 63_1-212, after 500 generations of population size 150.*

	Bezier	BP 3333	BP 3434
Cost	1.55×10^{-3}	4.93×10^{-3}	1.99×10^{-3}
Avg. Deviation	1.27×10^{-4}	5.57×10^{-4}	1.98×10^{-4}

The point-wise deviations of the different representations are compared in Figure 3.7. The BP 3333 representation closely resembles the other representations except near the trailing edge. The trailing edge thickness curve of the target wing section is slightly cusped, with an inflection point close to the endpoint. The BP 3333 parameterization, which has an inflection point fixed by the other parameters, cannot match the correct position as closely as the other two parameterizations.

As noted in Table 3.5, the BP 3333 parameterization was unable to reproduce the NACA 747A315 wing section to the required minimum error. The average deviation is 1.49×10^{-3}, which is still within the range of the deviations of the Eppler models in Selig's study. However, an examination of the deviations and representation reveals one of the weaknesses of the BP 3333 parameterization method.

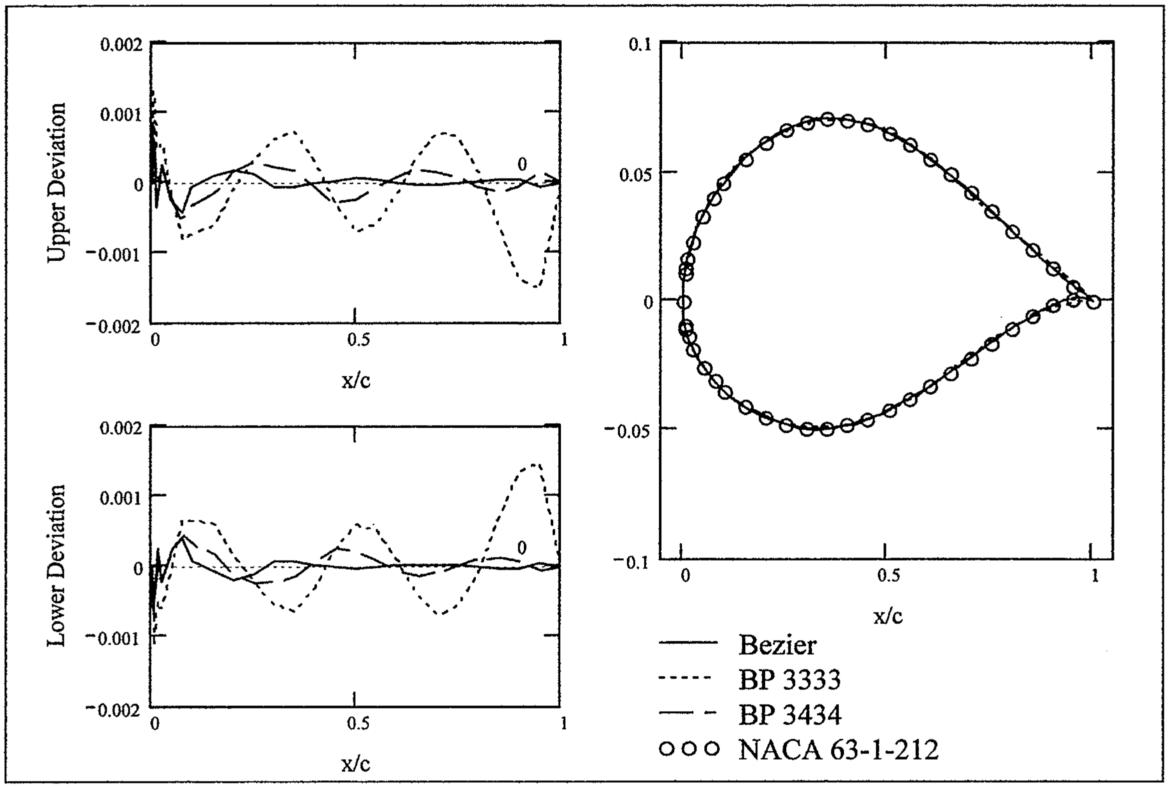


Figure 3.7 *Representation of NACA 63$_1$-212 by the three parameterizations, including upper and lower curve deviations.*

For comparison, the BP 3434 parameterization is allowed to converge to a cost of 0.005, which occurs after 176 generations and 17,000 FEs. The camber and thickness profiles, resultant airfoils, and point-wise deviations are compared in Figure 3.8. The BP 3333 parameterization cannot match the camber of the NACA 747A315. The BP 3434 trailing camber uses its additional trailing camber control point to position the inflection point correctly, while simultaneously matching the camber crest and the trailing edge direction.

While the failure of the BP 3333 parameterization is due mostly to the lack of sufficient freedom on the trailing camber, there is also a more minor problem in the trailing thickness profile, where it cannot match the sharp cusp in the NACA 747A315.

The BP 3434 parameterization is able to represent the trailing edge thickness shape correctly, once again due to the additional control point.

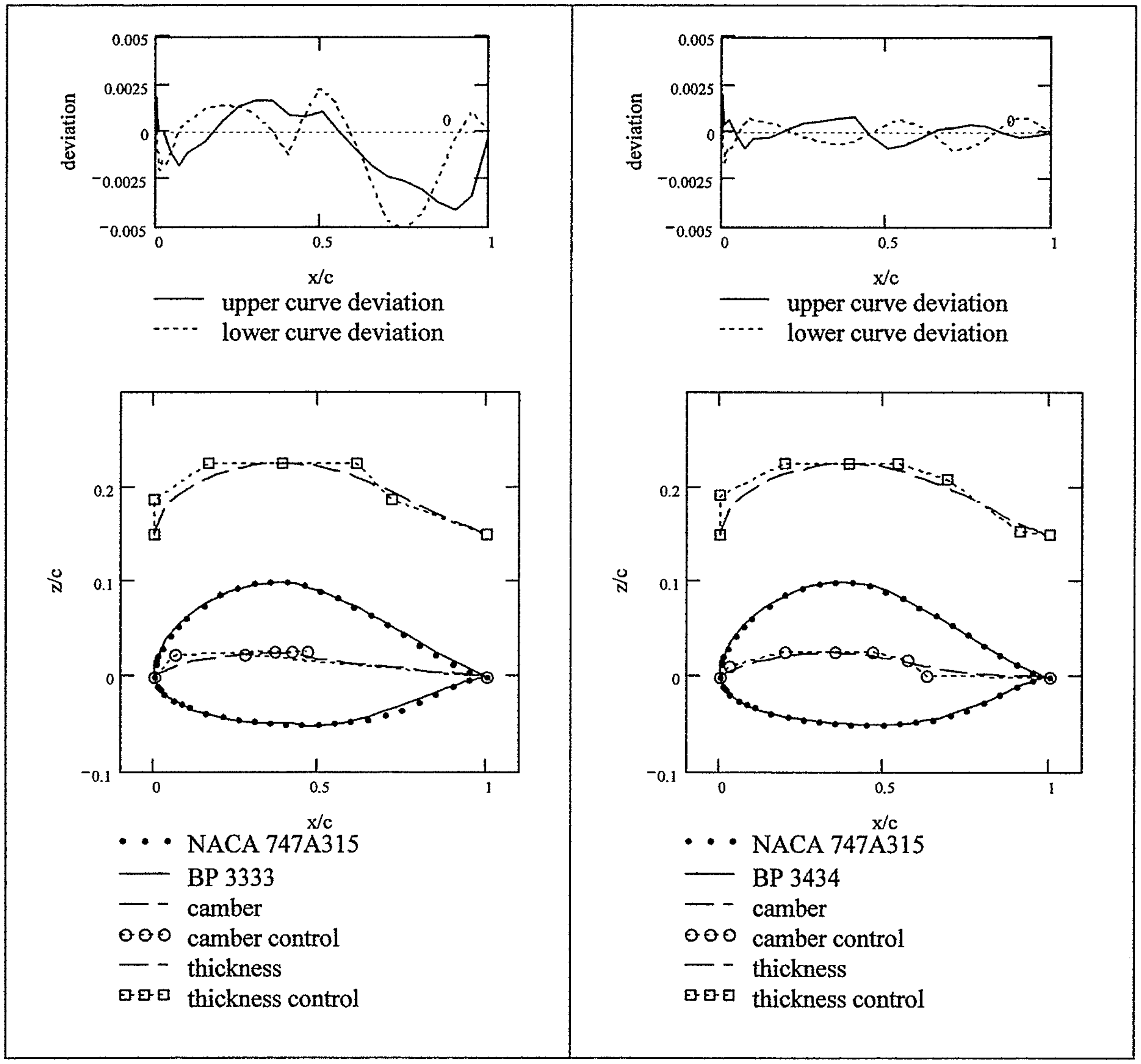


Figure 3.8 *Representations of NACA 747A315 by BP 3333 (left) and BP 3434 (right) after full convergence.*

3.4.4 Eppler Airfoils

In the 1950's, NACA shifted its focus to high-speed aerodynamics. The laminar-flow airfoil design scene shifted to Germany where Richard Eppler (1957) pursued the development of more accurate theoretical methods – especially conformal transformations. Over the years, Eppler developed an inverse method capable of designing airfoil shapes with prescribed boundary layer characteristics. This work culminated in a Fortran-based computer code (Eppler and Somers, 1980), and the

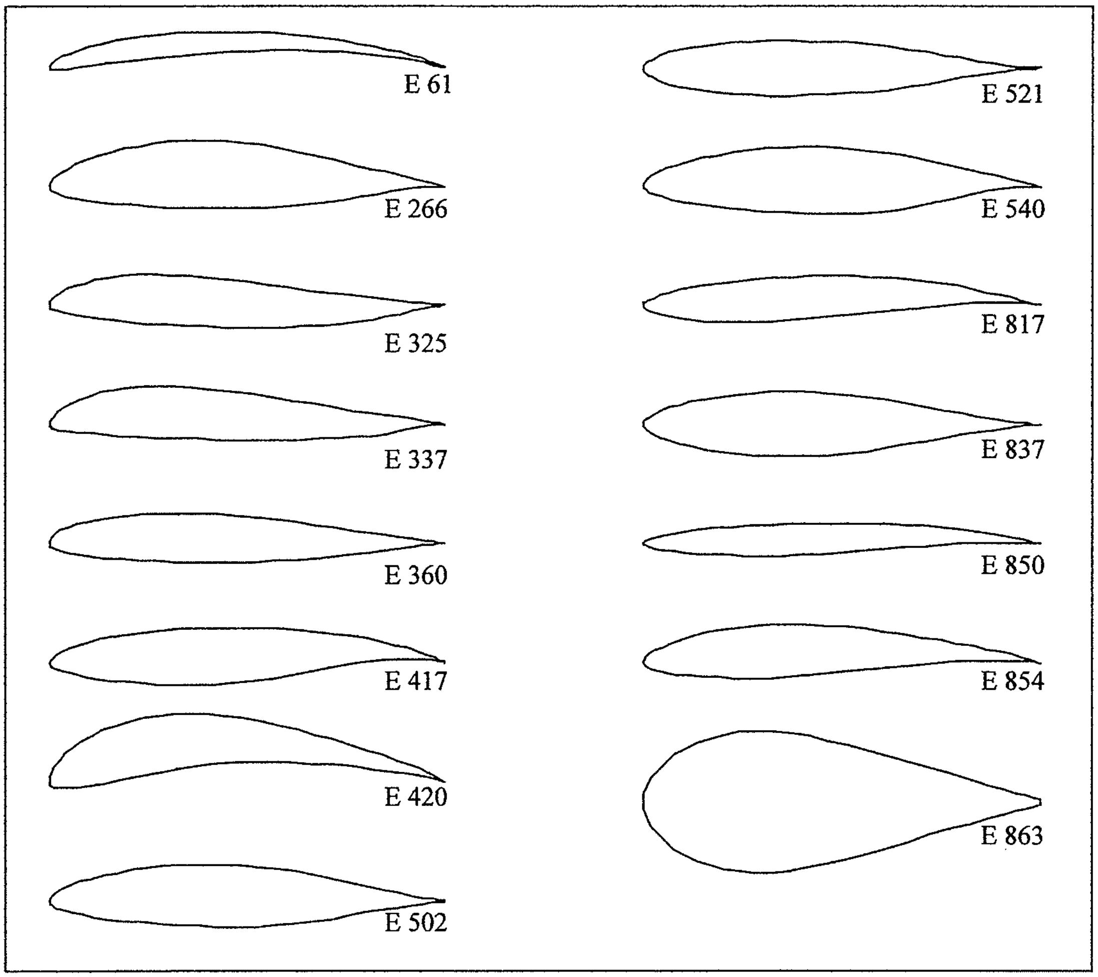


Figure 3.9 *Fifteen Eppler airfoils selected for representation.*

publication of an airfoil catalog summarizing the results (Eppler, 1990). 15 airfoils from this catalog are chosen here for reproduction (Figure 3.9). The shapes chosen are less conventional than the NACA wing sections, and are intended to challenge the parameterization methods beyond the ability of the NACA shapes to do so.

Optimization was stopped when the cost dipped below 0.01. However, the Eppler shapes are catalogued with a higher number of data points (70-80 data points, compared to 34 or 52 for the NACA shapes). Thus the average deviation is actually smaller than

Table 3.7 *NFEs required for convergence to Eppler airfoils. Unless otherwise noted, cost at convergence is 0.01.*

Airfoil	Bezier	BP 3333	BP 3434
E 61	3659	4490	3349
E 266	$^+$33902	***	15632
E 325	**	***	32262
E 337	**	***	5627
E 360	3488	$^+$26951	5417
E 417	*	$^+$39128	*
E 420	9139	27370	10878
E 502	3646	3423	4907
E 521	3704	2343	3152
E 540	4670	3898	5997
E 817	2762	5677	5539
E 837	3138	2184	3219
E 850	2390	3472	3489
E 854	2561	3269	6540
E 863	***	3376	7639
Acceptable:	73 %	80 %	93 %
Avg NFEs:	6641	10,465	8117
Avg NFEs excluding E266-E420	3316	3570	4870

$^+$acceptable, cost $\in$ (0.01, 0.0125)
*unacceptable bump
**unacceptable second-order discontinuity
***cost > 0.0125

that for a NACA representation with the same cost. Furthermore, several reproductions are quite acceptable, but converged to a cost slightly higher than 0.01. These all had cost within (0.01, 0.0125). The Eppler airfoils were indeed more challenging, and some representations were unacceptable, even though they achieved a cost lower than 0.01. The success of each parameterization is summarized in Table 3.7. Problematic shapes will be discussed below, in the order in which they appear in the table.

Consider first the E 266 airfoil. The Bezier representation converged at a cost of 0.01055. The average deviation is 9.29×10^{-4} – well within the range of average deviations in Selig's study. Figure 3.10 shows that the deviations are very similar to those

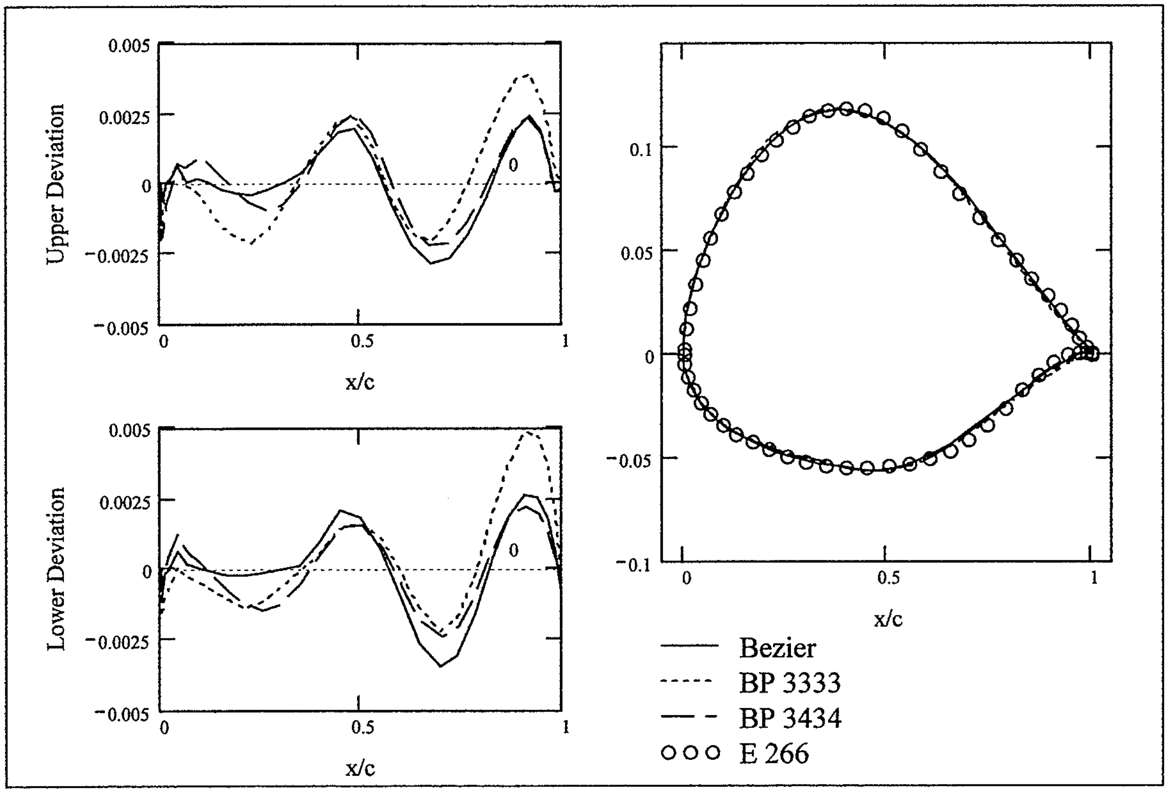


Figure 3.10 *Representations of E 266, including upper and lower curve deviations.*

of the BP 3434, which did converge to 0.01, and that the two representations are nearly indiscernible.

The BP 3333 representation of E 266 converged with a cost of 0.01331. The deviations (Figure 3.10) are nearly the same on the upper and lower surface, indicating it has reproduced the thickness curve quite accurately. In fact, the BP 3333 deviations are very similar in magnitude to the others, except in the last 15% of the chord. The trailing portion of the BP 3333 and 3434 representations are compared in Figure 3.11. The BP 3434 trailing camber, with its additional control point, is able to match the crest curvature with its first three control points, and uses the fourth to keep the camber high enough throughout before dipping down to the trailing edge position. The BP 3333 does not have

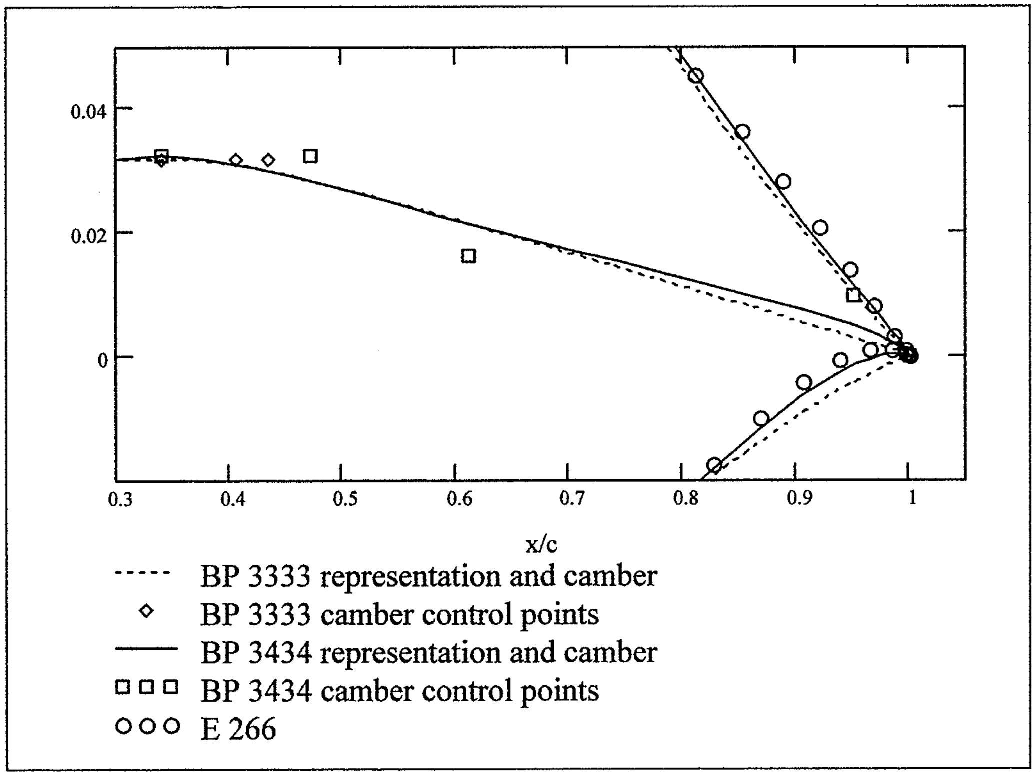


Figure 3.11 *Trailing portion of BP 3434 and BP 3333 representations of E 266.*

the luxury of the extra control point, and thus cannot simultaneously match the crest and the trailing edge shape of the camber.

Consider next the E 325 and E 337. These airfoils have a positive leading edge direction, and negative trailing edge direction. (Figure 3.12 shows the E 325 only, but the E 337 is similar.) The trailing camber curve dips below the axis prior to its trailing edge position of (1,0). An airfoil with such a camber cannot be represented by the BP 3333 parameterization. To match the camber curvature at the crest, the *y*-values of the second

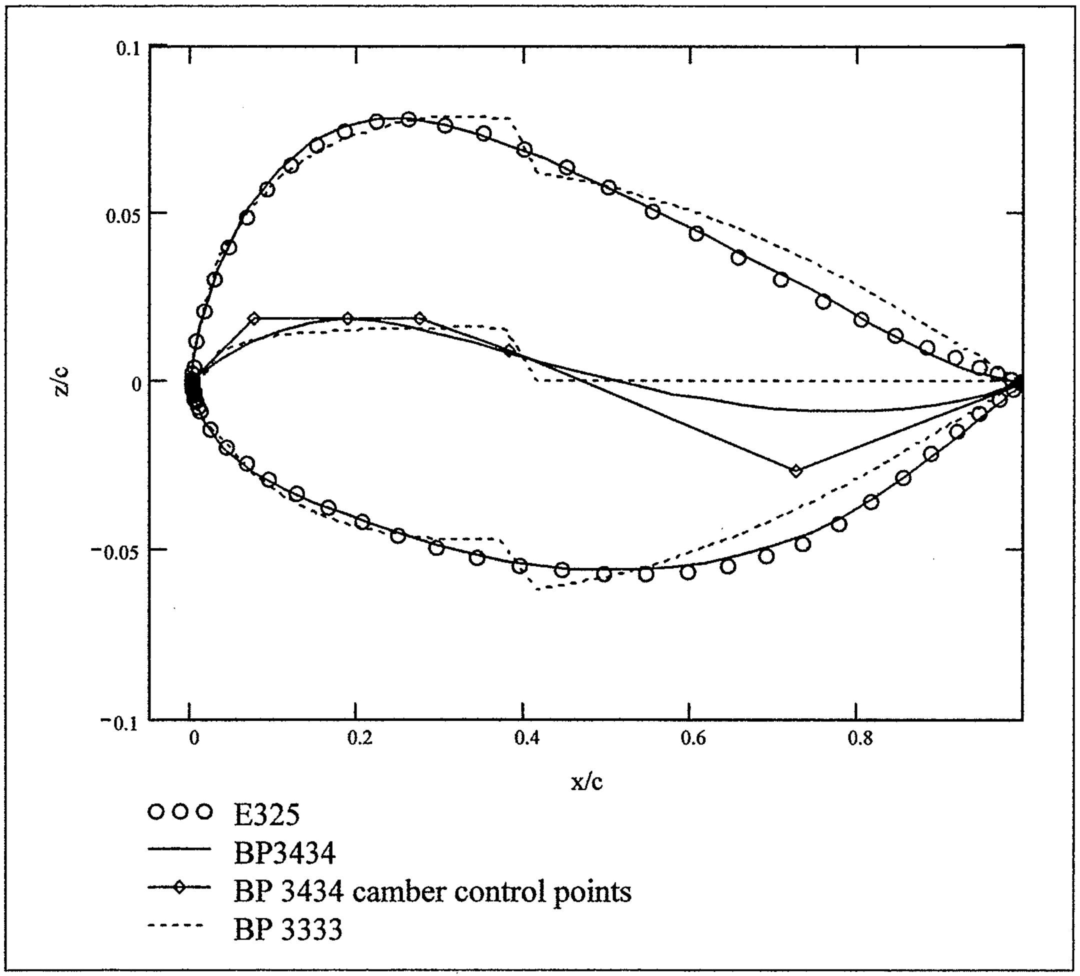


Figure 3.12 *BP 3434 and BP 3333 representations of E 325.*

leading edge control point and the second-last trailing control point must be equal. (See equations (3.26) and (3.35)). Thus, if the leading edge direction is positive ($\gamma_{le} > 0$), then $y_1^{lc} > 0$, which implies that $y_2^{tc} > 0$. The y-values of the remaining BP 3333 trailing camber control points are fixed by the crest and trailing edge positions: $y_0^{tc} = y_1^{tc} = y_c$ and $y_3^{tc} = 0$. Recall that Bezier curves always lie within the convex hull formed by their control points. In this case, one implication is that the BP 3333 trailing camber curve will always be non-negative if $\gamma_{le} > 0$. The parameterization attempts to mitigate this restriction by setting α_{te} very nearly zero. Not only is that insufficient to represent the trailing edge, but it also results in an unacceptable jump near the camber crest.

Although the Bezier parameterization does match these airfoils within the required tolerance, it does so at the expense of the crest curvature. For the Bezier representation of E 325, the camber crest curvature is –0.0016 from the left and –0.932 from the right. For E 337, it is –0.064 from the left, and –1.72 from the right. In both cases, the discrepancy is unacceptable. Thus, the BP 3434 parameterization is the only one of the three that can represent this kind of airfoil correctly. The additional trailing camber control point allows it to match simultaneously the crest curvature, trailing inflection point, and negative trailing edge angle.

Consider next the E 360. The BP 3333 parameterization converges to 0.01057, higher than 0.01, but is in fact quite acceptable. The average deviation is 9.57×10^{-4}. The point-wise deviations are small and comparable to the other parameterizations.

The E 417, on the other hand, was problematic for all parameterization methods. BP 3333 converges to a cost of 0.01216, but actually finds the best shape. (BP 3333 average deviation is 1.16×10^{-3}.) The E417 camber crest is far aft (at greater than 70% of the chord). The long leading camber is difficult to match with degree three curves. The BP 3333 method under-represents both upper and lower curves in the leading 20% of the chord. (See Figure 3.13.) The tradeoff is that its deviations decrease significantly in magnitude toward the trailing edge.

Notice that the BP 3333 deviation curve is the smoothest overall. The Bezier and BP 3434 deviations have sharp spikes between 50% and 80% of the chord. The symmetry of the deviations indicates an unacceptable bump in the camber curve (at about 56% for Bezier and 74% for BP 3434). The camber crest is so far aft (Figure 3.14) that the search must stray far from the initial population. By the time the Bezier and BP 3434 parameterizations have found the right position, the populations have converged around a camber with a sharp edge. The BP 3333 camber, on the other hand, is smooth.

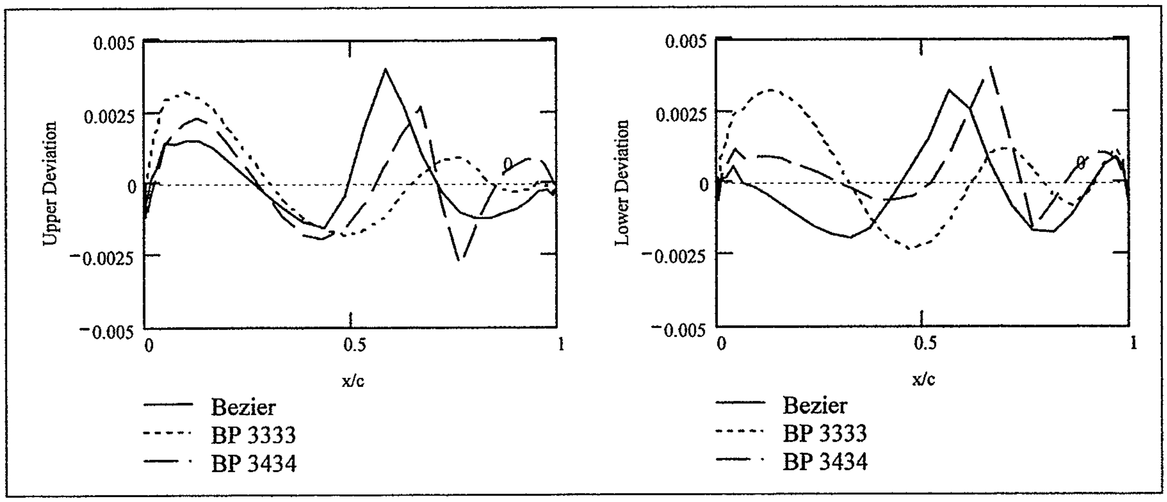


Figure 3.13 *Point-wise deviations for representations of E 417.*

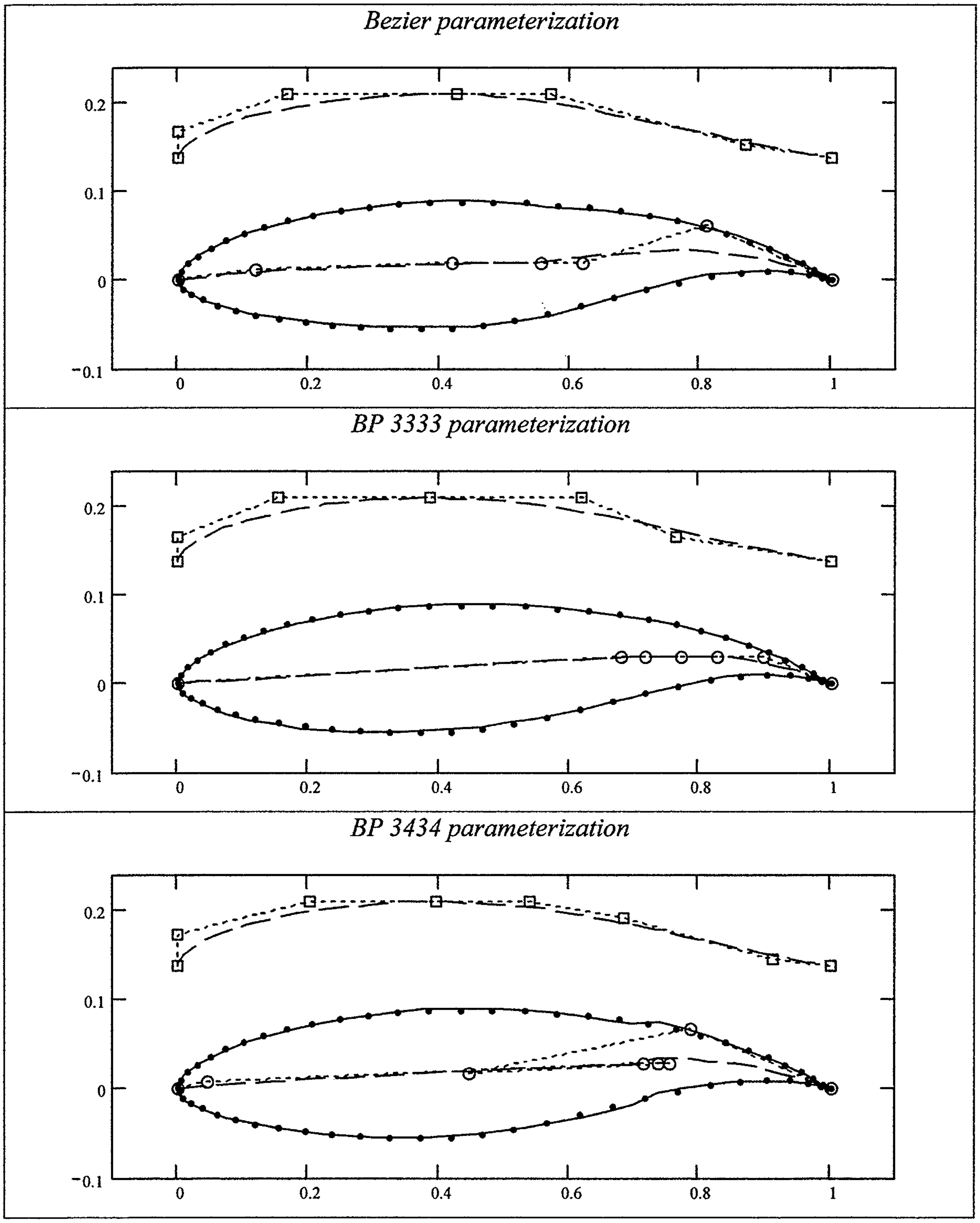


Figure 3.14 *Representations of E 417, including thickness and camber profiles and control points.*

Notice also the erratic nature of the BP 3434 camber control points in Figure 3.14. On the trailing camber curve, the third control point is far to the left of the second. This is in fact the cause of the sharp edge in the camber. Because the BP 3333 parameterization has the least freedom, its control points must be less erratic, and it is more likely to stay away from shapes with sharp edges.

Finally, consider the E 863. This airfoil has a non-zero trailing edge thickness, which is not admitted in Bezier parameterizations. The representations and deviations are depicted in Figure 3.15. Notice that the Bezier parameterization reproduces the airfoil quite well except near the trailing edge. Compared to the other parameterizations, the only significant deviation is in the last 5% of the chord.

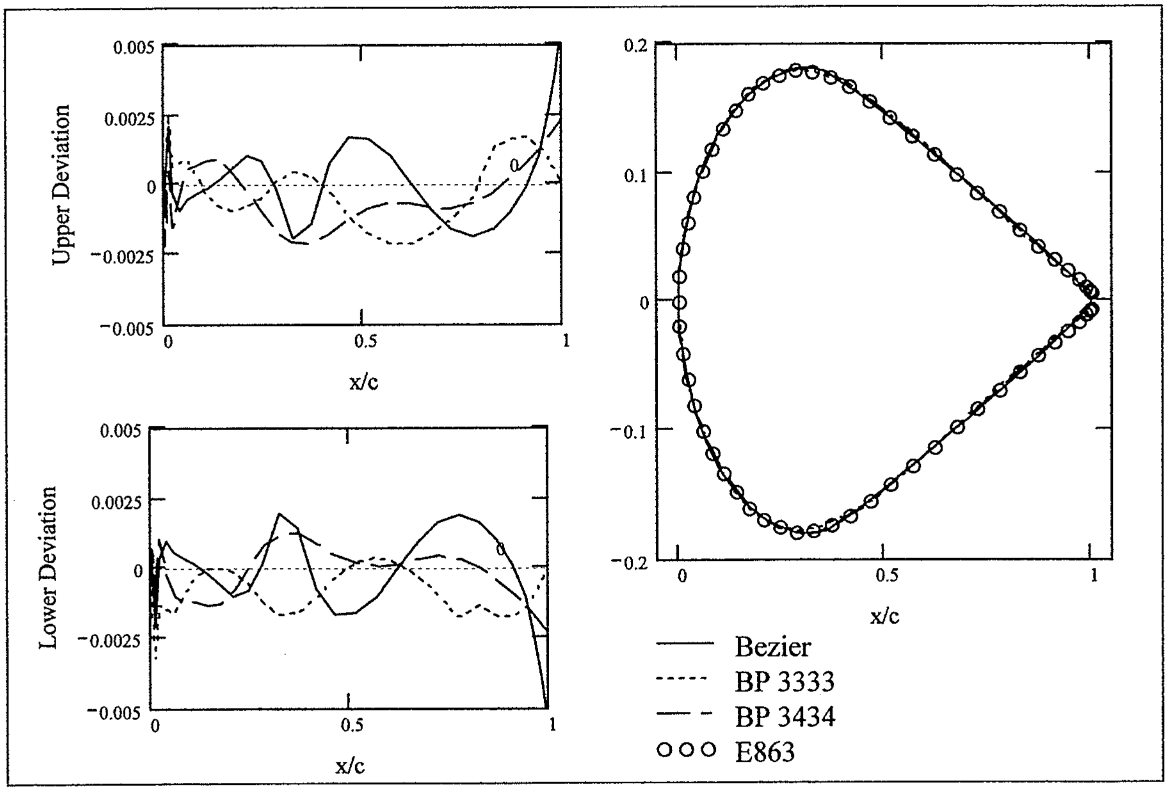


Figure 3.15 *Representations of E 863, including upper and lower curve deviations.*

3.4.5 Low-speed Airfoils

Selig's low-speed airfoil study has been discussed above (Section 3.4.1). Eight of the wing sections tested in the study are chosen for reproduction here. Again, the purpose is to challenge the parameterizations beyond their ability to reproduce NACA airfoils. The eight airfoils chosen are shown in Figure 3.16.

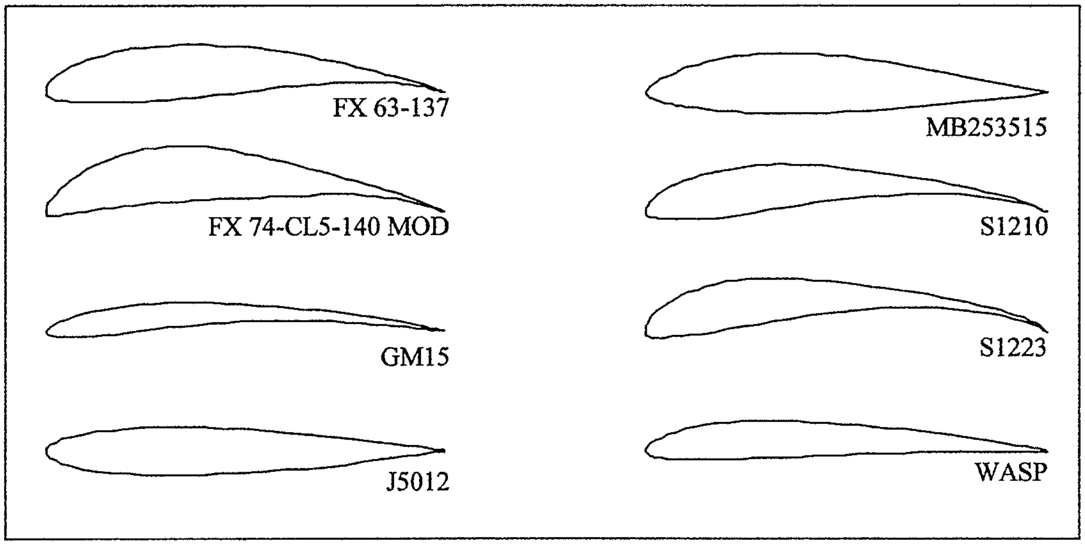


Figure 3.16 *Eight low-speed airfoils selected for representation.*

The cost limit for all representations was set to 0.01. The success of each parameterization is summarized in Table 3.8. The Bezier and BP 3434 methods converged to within the limit for all eight airfoils. In one case, BP 3333 converges to a cost slightly higher than 0.01, but represents the airfoil well. In another, the target had to be shifted for BP 3333 to match it. These two cases are discussed in more detail below.

Although the BP 3333 representation of FX 74-CL5-140 MOD cannot attain the 0.01 cost, Figure 3.17 shows that the representation is acceptable. There are no large deviations from the target. The average deviation is 9.7×10^{-4}, compared to 7.7×10^{-4} for the BP 3434 representation.

Table 3.8 *NFEs required for low-speed airfoils. Cost at convergence is 0.01.*

Airfoil	Bezier	BP 3333	BP 3434
FX 63-137	6,556	10,632	22,995
FX 74-CL5-140 MOD	8,983	[+]22,903	9,620
GM15	4,146	5,457	6,892
J5012	2,134	1,208	3,304
MB253515	3,304	4,807	5,085
S1210	6,010	9,891	10,476
S1223	7,643	[++]11,999	16,259
WASP	5,086	3,807	5,318
Acceptable:	100%	100%	100%
Avg NFEs:	5,483	8,838	9,994

[+] acceptable, cost 0.01161

[++] converged after leading edge was shifted to origin

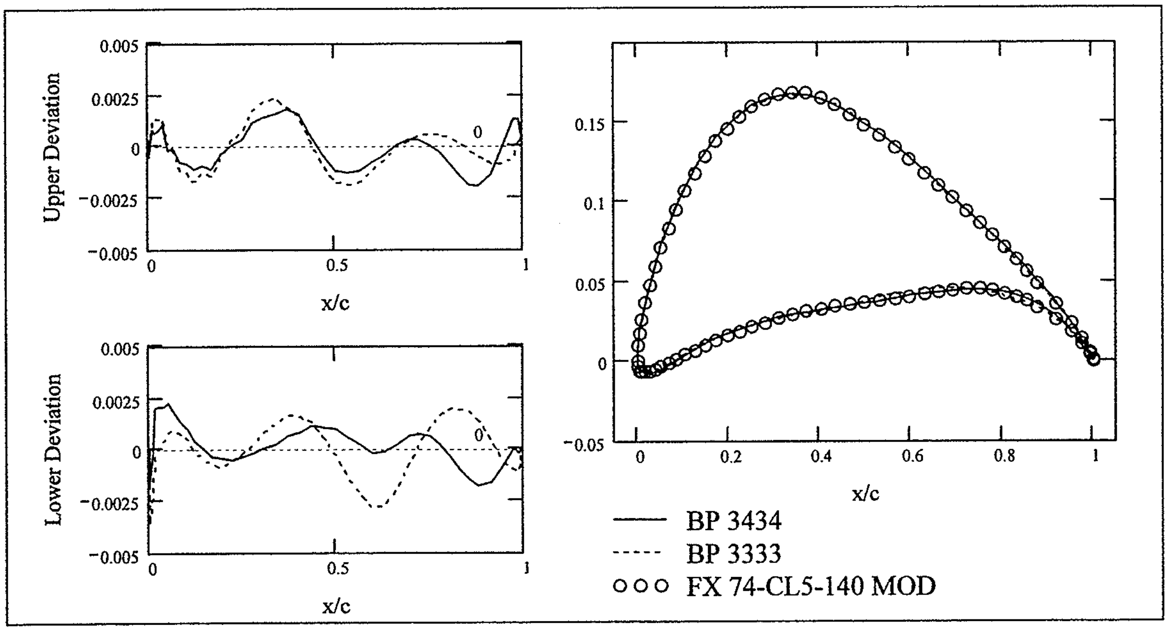


Figure 3.17 *Representation of FX 74-CL5-140 MOD by BP 3434 and BP 3333, including upper and lower curve deviations.*

The BP 3333 representations failed to reproduce the S1223 airfoil using the data points published by Selig. The biggest reason is that the leading edge of the S1223 does not start at the origin. The target was modified by shifting it to the origin, and expanding it to have trailing edge at x=1. This resulted in a nonzero z_{te}, which was therefore left variable for optimization. BP 3333 was able to match the modified target to within the specified limit. The Bezier and BP 3434 parameterizations are able to represent the original data by using leading edge directions that are nearly horizontal (Figure 3.18), but BP 3333 does not have the freedom to do so.

Notice once again the erratic nature of the trailing camber control points for the BP 3434 parameterization (Figure 3.18). In fact, the trailing edge angle is in the wrong quadrant. Magnifying the trailing edge camber, one would find it to be multivalued. The scale is so small, however, that the parameterization is not affected. This phenomenon

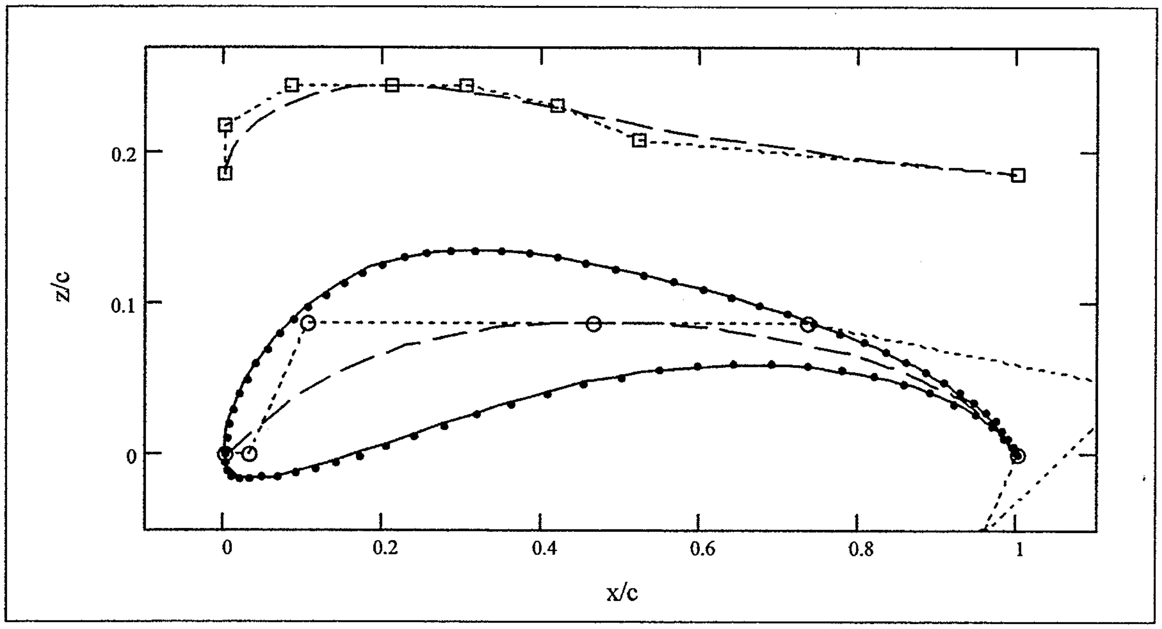


Figure 3.18 *BP 3434 representation of S1223, including thickness and camber curves and control points.*

was not uncommon for BP 3434 designs. The additional control point freedom, while necessary in some cases, was too much for it to handle in others. It occasionally resulted in premature convergence, which had to be corrected by narrowing the initial bounds.

3.4.6 Summary

All three parameterizations are able to represent a high percentage of the sample of airfoils chosen. Of 63 airfoils, the Bezier parameterization reproduced 58, BP 3333 reproduced 59, and BP 3434 reproduced 62. For the vast majority, convergence occurred in fewer than 10,000 FEs. For shapes near the edge of design space, convergence sometimes required 20,000 – 30,000. Overall the Bezier parameterization required the fewest NFEs, and BP 3434 the highest, which is not surprising. The Bezier parameterization is designed geometrically, so it should work well for this kind of geometry-matching problem. BP 3434 has the most parameters, and thus the largest design space.

One of the biggest limitations of the Bezier parameterization is the discontinuous second derivative at the camber and thickness crests. The Bezier representations of E 325 and E 337, for example, showed unacceptable curvature differences fore and aft of the crest. For the E 417, the camber juncture has a visible sharp edge, which shows up as spike in the plot of deviations from the target airfoil. A second deficiency is that the trailing edge thickness of the Bezier parameterization is fixed at zero, which kept it from reproducing E 863.

The BP 3333 parameterization has the least control point freedom of the three methods. This can be a good thing, as in the case of E 417. The lack of freedom means it cannot search regions with sharp edges, and it thus ends up with the best design. There are at least two classes of shapes that BP 3333 cannot represent. The first is any airfoil with a radical change in the trailing camber curvature (NACA 747A315, E 266). The second is any airfoil with a camber that dips below the *x*-axis (E 325, E 337). BP 3333 also does not represent sharp cusps as well as the other methods, although this did not result in any convergence failures.

The BP 3434 was the most robust. The robustness is due to the additional control points on the trailing edge curves. In many cases, however, these are not necessary for the design, and BP 3434 doesn't quite know what to do with them. The E 417, for example, has its camber crest far aft. Before BP 3434 can find the right shape, it converges to a camber with a sharp edge. In other cases, the control point polygon is somewhat erratic, which can result in premature convergence to shapes with sharp edges, or with wrong trailing edge directions.

3.5 Effect of Parameterization on Design Speed

In the previous section it was demonstrated that all three parameterizations can represent a broad range of airfoil shapes. We are now interested in whether the particular parameterization method chosen influences the robustness and rate of convergence for inverse design. The focus is narrowed to compare only BP 3333 and Bezier parameterization. The BP 3333 solution space is smaller than that of BP 3434, but with fewer parameters overall, and no Bezier parameters, it should have more potential for acceleration. This was verified in a very preliminary way with a few inverse design experiments. BP 3434 tended to converge more slowly, and sometimes to a higher cost.

The three design cases discussed in Chapter 2 are considered in this section. The convergence benchmarks developed for the Bezier parameterization serve as the standard for design speed. Section 3.5.1 discusses the optimization parameters used for design. Sections 3.5.2 – 3.5.4 compare Bezier and BP 3333 convergence to the three design targets. A summary of the results is given in section 3.5.5.

3.5.1 Optimization parameters

The same design parameters are used for each of the three design cases, with one exception: As in Chapter 2, the maximum panel length was 0.03 for the 112°-cambered target, and 0.04 for the other two.

The bounds on the initial BP 3333 population are given in Table 3.9. The ranges for a few variables are slightly different that those used for airfoil representation (Table 3.1). These new ranges were found to be more effective for the broad range of design targets

Table 3.9 *Bounds on the initial BP 3333 population used by DE for the inverse design problems.*

Camber Parameter	Lower	Upper	Thickness Parameter	Lower	Upper
γ_{le}	0.05	0.5	r_{le}	-0.04	-0.001
x_c	0.3	0.6	x_t	0.15	0.4
y_c	0.01	0.3	y_t	0.05	0.2
κ_c	-1.0	-0.1	κ_t	-1.0	-0.1
z_{te}	0	0	dz_{te}	0	0.001
α_{te}	0.05	0.5	β_{te}	0.001	0.3

used here. The initial range for the stagger angle is [-15°,15°], and that for the pitch/chord ratio is [0,1].

Geometric constraints for BP 3333 are the same as those used in the previous section. Each Bezier curve must be one-to-one, and parameter bounds (3.19) and (3.34) are imposed. The trailing edge position and thickness are fixed at zero, resulting in D=13 variables.

The effectiveness for the Bezier parameterization of the DE/rand-to-best/1/CR=1 variant does not necessarily imply the same for the BP 3333 parameterization. In fact, it resulted in misconvergence for the 112°-cambered target (Figure 3.19). Mutation alone (CR=1) is the recommended strategy for epistatic optimization problems (Price, 1999). This is related to the fact that mutation in DE is rotationally invariant, while crossover is not. However, the BP 3333 parameterization should result in more separability, which suggests the use of crossover.

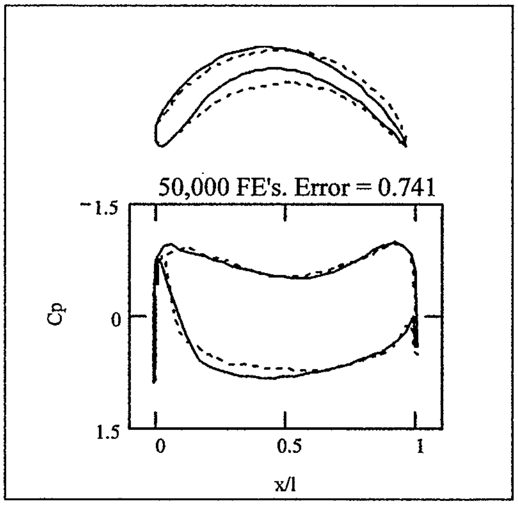


Figure 3.19 *Misconvergence of BP 3333 for the 112°-cambered design, using DE with NP=120 and CR=1.*

A variety of crossover strategies were attempted. Overall, binary crossover with CR=0.95 was the most effective. Exponential crossover tended to result in slower convergence. Later experiments confirmed that binary crossover is more amenable to acceleration. Smaller values of CR (i.e. more crossover) also converged more slowly. Thus the DE variant used throughout for BP 3333 design is DE/rand-to-best/1/bin with NP=120, F=0.85, and CR=0.95.

3.5.2 Convergence to C4/70/C50

The convergence of the cost for the C4/70/C50 pressure distribution is shown in Figure 3.20. Design snapshots are given in Figure 3.21. Results are essentially comparable. Initially, BP 3333 converges slightly more slowly than Bezier, but by 30,000 FEs, the costs are the same.

3.5.3 Convergence to 112°-cambered blade

Convergence to the 112°-cambered pressure distribution is shown in Figure 3.22. Design snapshots are given in Figure 3.23. The misconvergence that plagued the BP 3333 design for CR=1 (Figure 3.19) is no longer a problem when crossover is used. The BP 3333 design converges sooner, to an airfoil more closely approximating the target, and to a lower cost than the Bezier benchmark.

3.5.4 Liebeck pressure distribution

Convergence to the Liebeck pressure distribution is shown in Figure 3.24, with snapshots in Figure 3.25. The BP 3333 design converges sooner and to a lower cost than the benchmark. The shape it finds is thinner, and has higher curvature on the lower surface. Note that the minimum leading edge control point constraint, used to enforce a rounded leading edge for the Bezier design, is not necessary for the BP 3333 design. The new r_{le} parameter is sufficient to steer the design away from a sharp leading edge.

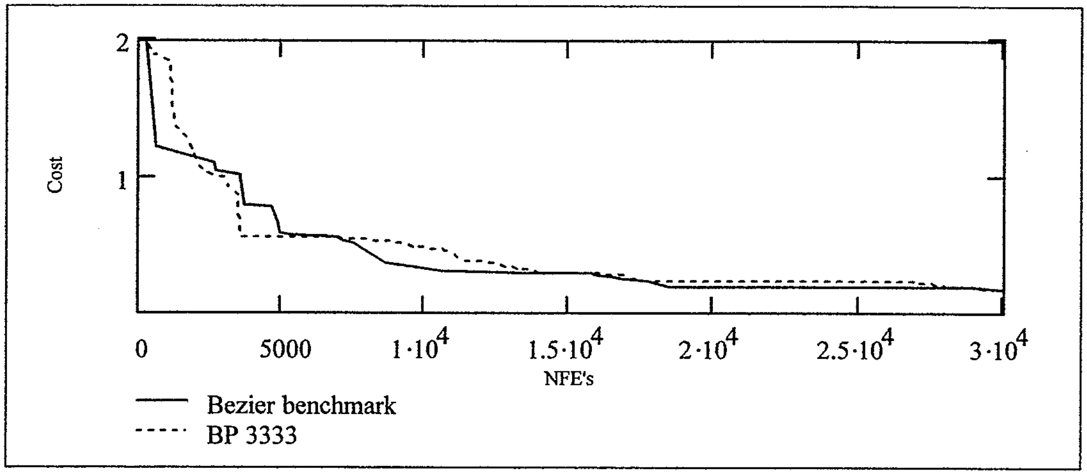


Figure 3.20 *Effect of airfoil parameterization on convergence to the C4/70/C50 target.*

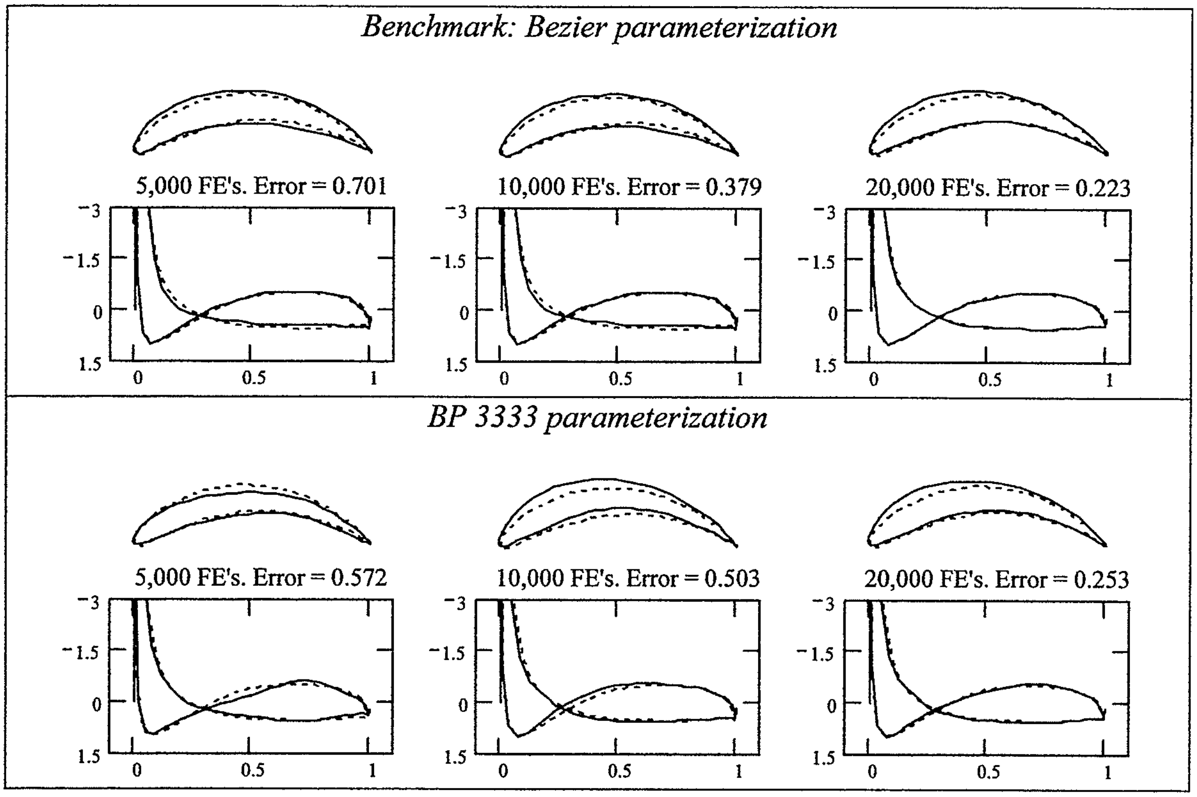


Figure 3.21 *Convergence snapshots showing the effect of airfoil parameterization on the development of the C4/70/C50 design.*

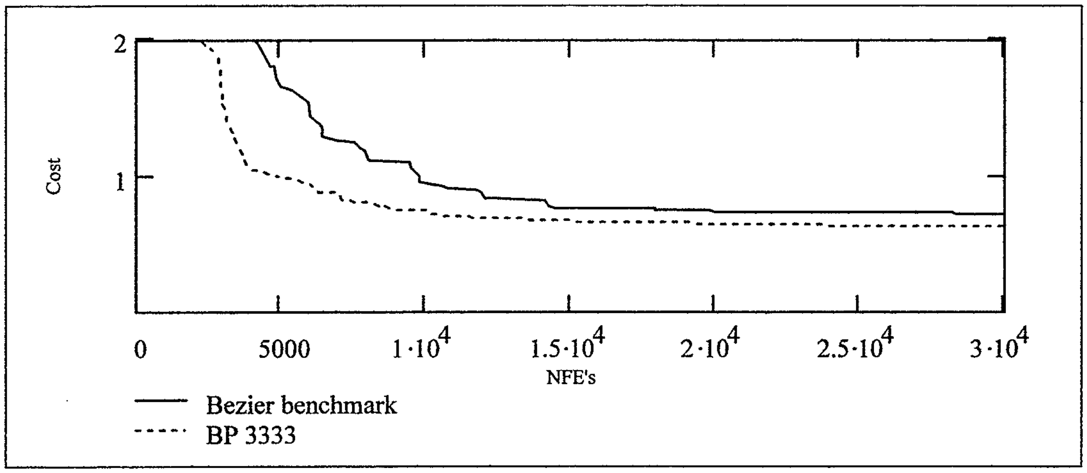
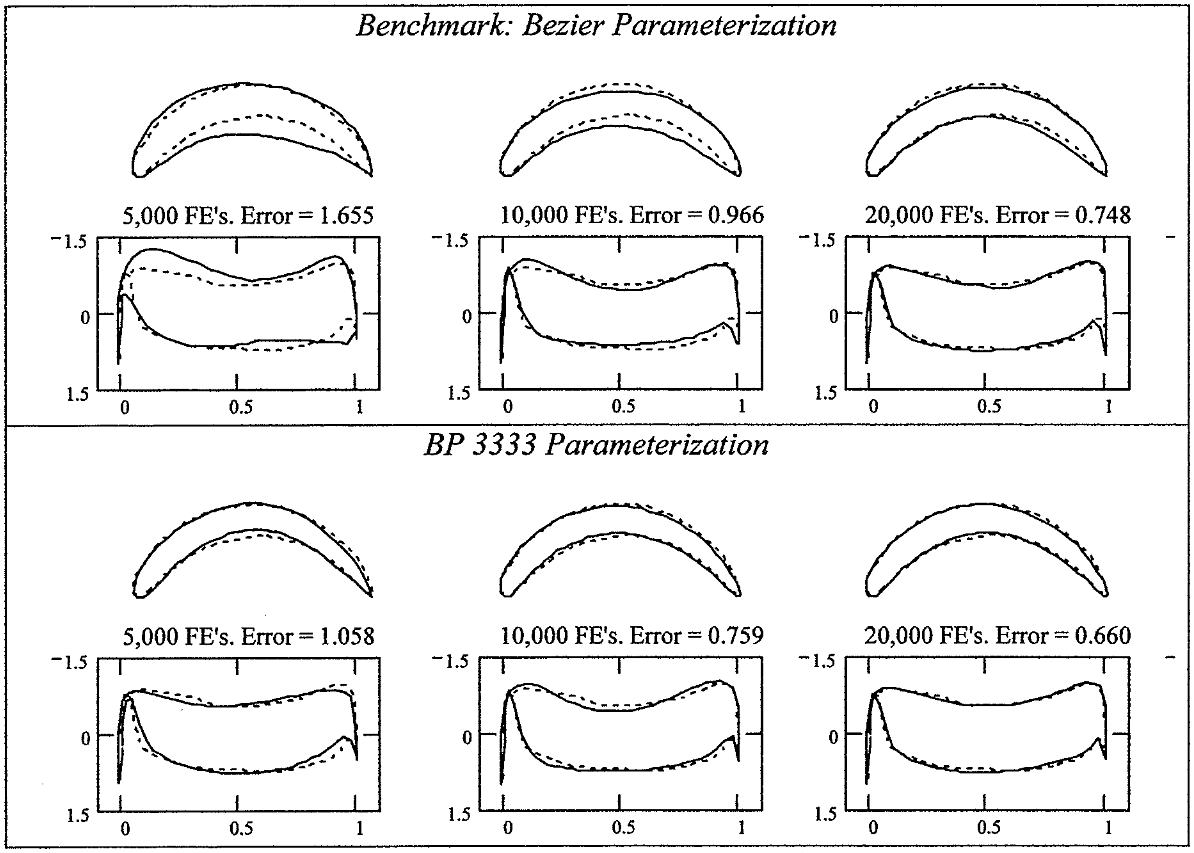


Figure 3.22 *Effect of airfoil parameterization on convergence to the 112°-cambered target.*


Benchmark: Bezier Parameterization

5,000 FE's. Error = 1.655

10,000 FE's. Error = 0.966

20,000 FE's. Error = 0.748

BP 3333 Parameterization

5,000 FE's. Error = 1.058

10,000 FE's. Error = 0.759

20,000 FE's. Error = 0.660

Figure 3.23 *Convergence shapshots showing the effect of airfoil parameterization on the development of the 112°-cambered design.*

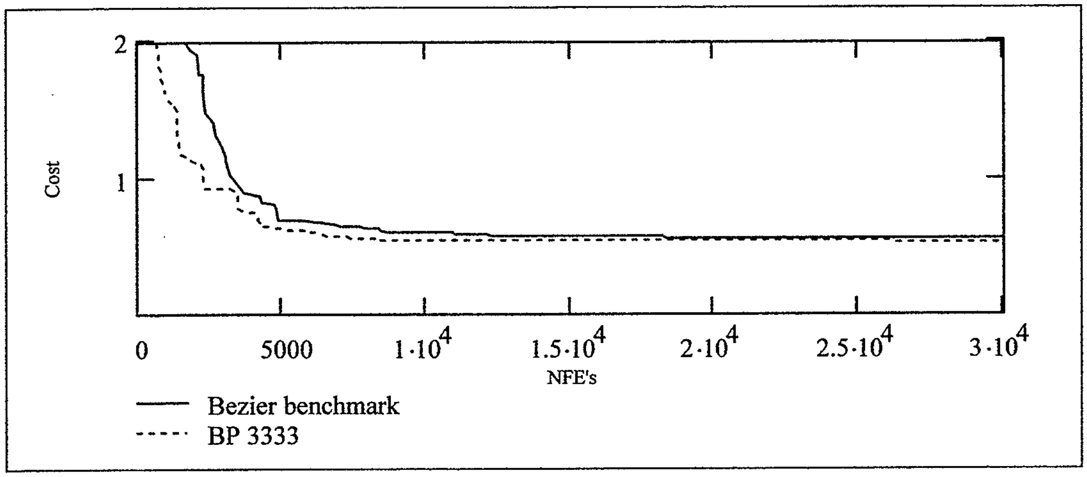
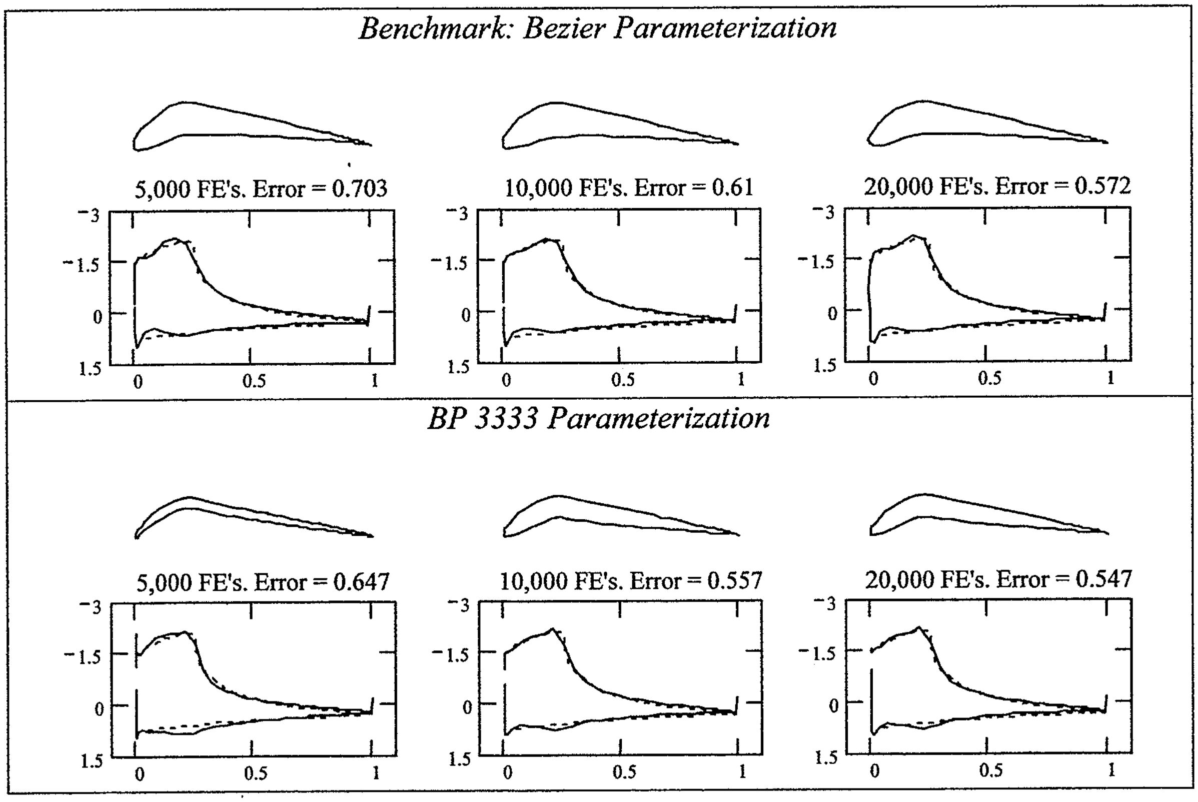


Figure 3.24 *Effect of airfoil parameterization on convergence to the Liebeck target.*


Benchmark: Bezier Parameterization

5,000 FE's. Error = 0.703

10,000 FE's. Error = 0.61

20,000 FE's. Error = 0.572

BP 3333 Parameterization

5,000 FE's. Error = 0.647

10,000 FE's. Error = 0.557

20,000 FE's. Error = 0.547

Figure 3.25 *Convergence shapshots showing the effect of airfoil parameterization on the development of the Liebeck design.*

3.5.5 Conclusions

The performance of the two parameterizations for the C4/70/C50 target is very similar. They find roughly the same shape using roughly the same NFE's. For the other two design cases, however, BP 3333 performs significantly better than Bezier – with accelerated convergence to a better design.

Overall, then, BP 3333 is a definite improvement over the Bezier parameterization – at least for this particular aerodynamic design problem. It is more separable, is more closely linked to the aerodynamics of the shape, and has fewer parameters, all of which should give it more potential for acceleration. Without acceleration, convergence is at least comparable, and in two cases significantly better. It has better continuity characteristics, and more aesthetic appeal. It has the ability to steer the design away from undesirable features. Constraints become much easier to envision and impose due to the aerodynamic nature of the variables. For example, a minimum leading edge radius or minimum crest curvature can be imposed to avoid sharp corners. Structural constraints such as a minimum wedge angle are similarly easy to achieve.

While BP 3434 is more robust than BP 3333, it also uses more parameters, and some of these will contribute nonlinearity to the objective function. BP 3333 thus has more potential for acceleration. Furthermore, by avoiding sharp edges, BP 3333 has the ability to focus the search within regions of acceptable aerodynamic shapes. It is unable to represent a small percentage of airfoils, but this shortcoming is overcome by its other advantages.

Throughout the remainder of this thesis, therefore, the BP 3333 parameterization will be used in place of the Bezier parameterization. In the subsequent chapters, it will be demonstrated that the potential for acceleration is indeed great.

Chapter 4 Acceleration by Variable Birthrate

In this chapter, the first of three DE-specific acceleration strategies is discussed. In Variable Birthrate DE (VBDE), not all population vectors are allowed to generate trial vectors for the next generation. Section 4.1 introduces the concept, and examines the quality of the population in generations 0-100. Section 4.2 examines the effect of VBDE on convergence for the three design case studies. Conclusions are made in section 4.3.

4.1 Selective Reproduction

Although ethically deplorable, socially unacceptable, and practically ineffectual for a human population, one wonders how selective reproduction might influence a simulation such as DE. An initial DE population can consist of a wide variety of potential solutions. Some might have very low cost. Others may lie within the constraints but have high cost. Still others may violate the constraints altogether. In DE, all have the same opportunity to generate new offspring. It seems possible that the search could be accelerated if some high-cost vectors are neutered.

The idea is not new to evolutionary computation. In the first GA's – developed by Holland (1962) and his students – fitter individuals were given more opportunities to reproduce. Later, in his PhD thesis, Hollstien (1971) adapted a variety of animal breeding strategies for use in a GA.

A common probability of reproducing was the ratio of the individual's cost to the total population cost (measured, for example, as the sum of costs of all individuals). For many objective functions, however, this ratio is similar throughout the entire population –

producing very weak selection pressure. De Jong (1975) modified the idea by calculating individual cost relative to the highest cost in the population. This often resulted in the opposite problem – overly strong selection pressure leading to premature convergence.

A strategy that can sometimes mediate between weak and strong selection is that of ranked selection. Individuals are ranked by cost within the population, and probability is a linear function of rank. Another common technique is tournament selection (Goldberg and Deb, 1991), in which many small sets of individuals are randomly selected, and the best few in each set are selected to reproduce.

VBDE is an attempt to incorporate some of these ideas for DE. Using a Gaussian probability distribution, each individual is assigned a reproductive probability, or birthrate. Those with lowest cost will have birthrates near one. Those with highest cost have birthrates near zero. The birthrate function $b(\mathbf{x})$ is:

$$b(\mathbf{x}) = \exp\left(-(c(\mathbf{x}) - c_{\min})^2 / 2s_c^2\right), \tag{4.1}$$

where $c(\mathbf{x})$ is the cost of vector $\mathbf{x}$, $c_{\min}$ is the lowest cost in the population, and s_c is the standard deviation of costs in the population. The distribution of birthrates shown in Figure 4.1 is for a population of twenty with equidistant cost values.

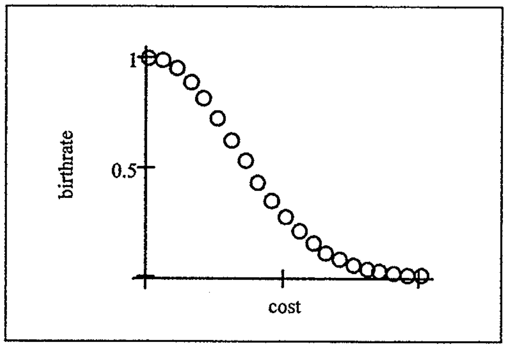


Figure 4.1 *Distribution of birthrates for a population with 20 equidistant cost values.*

In general, of course, costs need not be even approximately equidistant. For example, when there are many constraint violations, most of the population could have cost greater than 5000, while a few could have very low cost. In a case such as this, all constraint violators will have birthrates near zero, and the few good solutions will have birthrates near one.

Reproduction is regulated as follows. The user selects a birthrate cutoff, *BR*, where $BR \in [0,1]$. Every vector **x** with birthrate $b(\mathbf{x})$ > BR is allowed to procreate as normal. For each **x** with $b(\mathbf{x}) \leq BR$, choose r, a random value between 0 and *BR*. If $b(\mathbf{x})>r$, **x** procreates as normal. Otherwise, an individual with birthrate greater than *BR* is chosen at random to generate a child vector, **c**. The fitter of **x** and **c** survives to the next generation. Note that if the user selects *BR*=0, every vector procreates as usual, and the search is performed by DE without modification. If *BR*=1, the best vector in the population is likely to produce many children for the next generation. Any others can procreate at most once, with the probability that they do so at all decreasing as their cost-distance from the best increases.

To examine the birthrate spread for the aerodynamic design problem, birthrates were recorded for each individual in populations 0-100. DE/rand-to-best/1 was used with no crossover, the BP 3333 airfoil parameterization, and NP=120. Figures 4.2 - 4.4 show the distribution of birthrates for several generations of the C4/70/C50 design process. These go through three phases. In phase one (generations 0-40) the population contains mostly high cost vectors, and many birthrates near zero. In phase two (generations 40-60) the population is in transition from high cost to low cost. There are only a few high-cost vectors, and many birthrates near one. In phase three (generations 70+), the entire population is low-cost, and the whole range of birthrates is represented.

In phase one (Figure 4.2), there are many constraint violations. Recall that these have cost between 5000 and 10,000, assigned randomly by the penalty function. In the initial population, the best cost is 4645, the worst is 46,609 (not shown), and the rest lie between 5000 and 10,000. Since all vectors have high cost, there is a range of birthrates. At generation 1, one vector has been found with significantly lower cost (135). It receives a birthrate of 1, and the rest of the population has birthrates smaller than 10^{-3}. As the search progresses, the number of constraint violations decreases. By generation 40, there are only five.

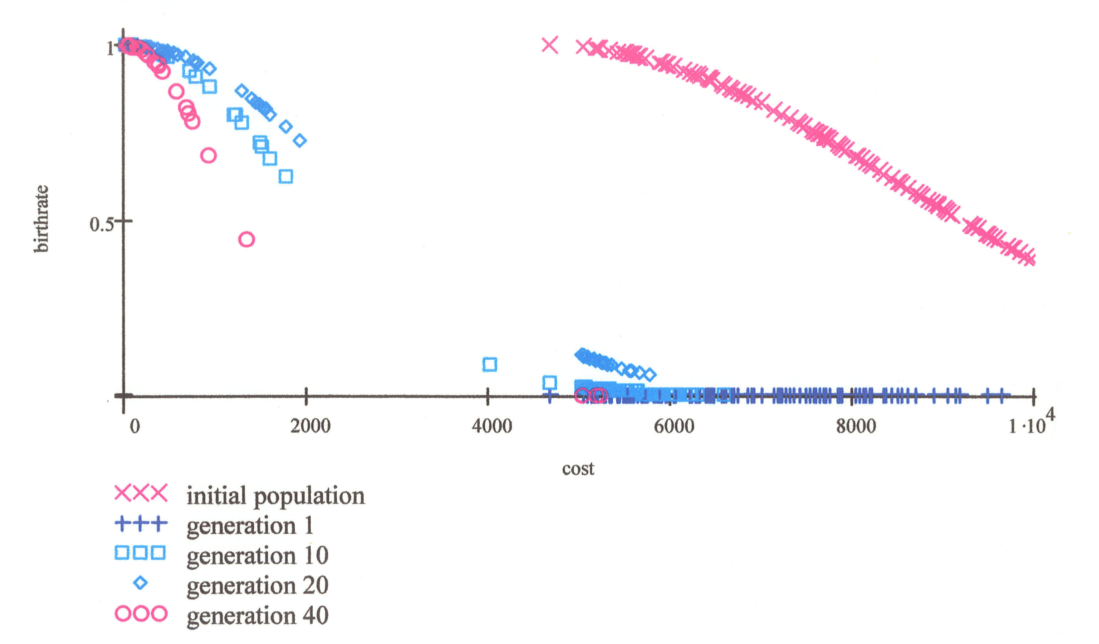


Figure 4.2 *Phase 1: High cost. Distribution of birthrates in generations 0-40, C4/70/C50 target.*

The tendency in phase one is to neuter all constraint violations if the population contains at least one reasonably good shape. The optimization parameter *BR* would have to be very low for many constraint violations to produce children. In early generations (1-10), it is likely that only constraint violators will be neutered, but as the generations progress, it becomes more probable that some high-cost valid shapes will neutered also.

By generation 50 (Figure 4.3, which includes generation 40 for comparison) the population has begun the transition to low cost. There are no constraint violations, but still a few high vectors with cost greater than 200. These have birthrate smaller than 0.02. The one medium range cost (65.5) has birthrate 0.58. The rest have cost less than 25, birthrates greater than 0.95. By generation 60, there is only one vector with cost (249.5) greater than 10 and birthrate smaller than 10^{-3}. All remaining vectors have birthrate greater than 0.9. In phase two, there is a strong likelihood of neutering only the few high-cost vectors.

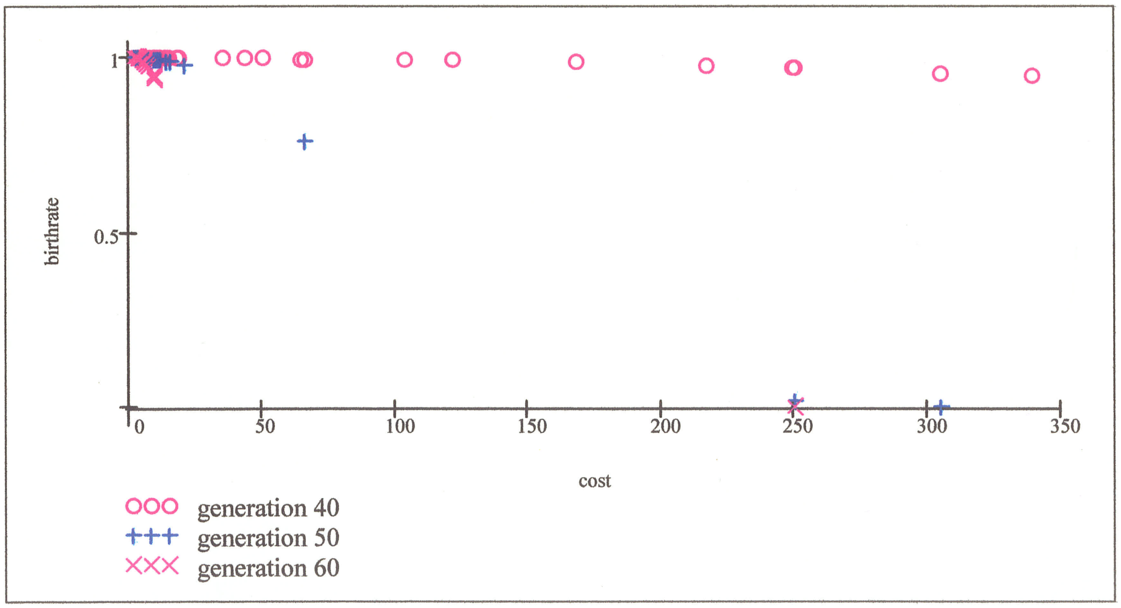


Figure 4.3 *Phase 2: Transition. Distribution of birthrates in generations 40-60, C4/70/C50 target.*

At generation 70 (Figure 4.4 , which includes generation 60 for comparison) the population is beginning to converge. The highest cost is down to 6.2. At generation 100, the highest is 1.9. The spread of birthrates is much more uniform than in the previous two phases. Thus it becomes increasingly likely that many good shapes will not reproduce.

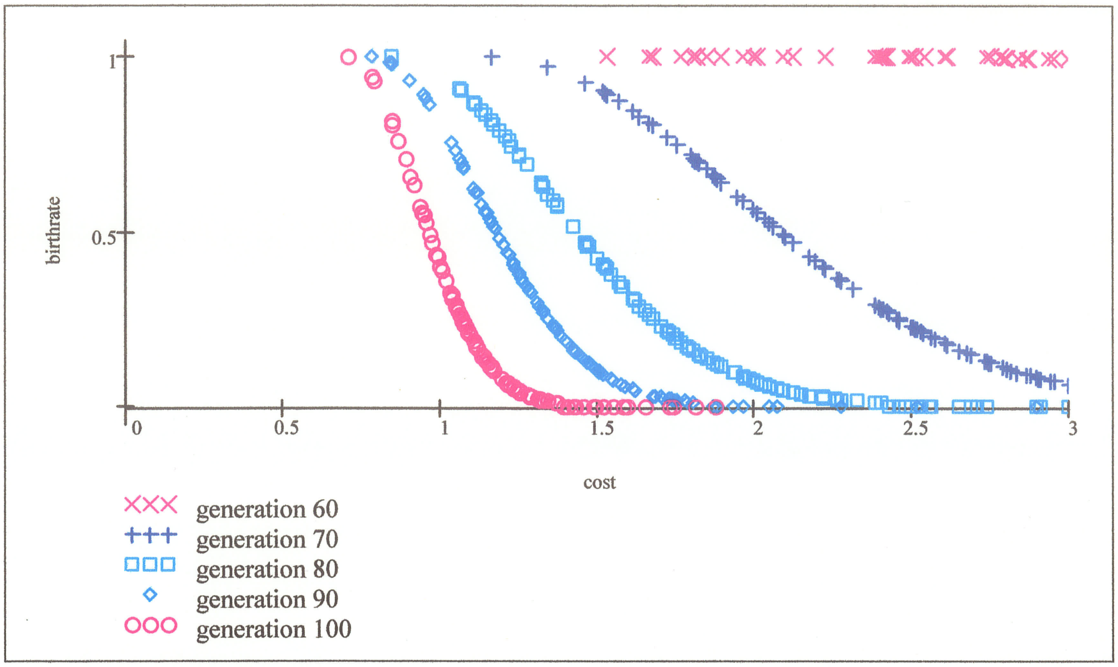


Figure 4.4 *Phase 3: Low cost. Distribution of birthrates in generations 60-100, C4/70/C50 target.*

It is important to note just how few valid airfoil shapes are generated in the first generations. Figure 4.5 plots the cost and function evaluations after every 10 generations. Only 25 valid shapes have been found after 10 generations, with 1320 vectors attempted.

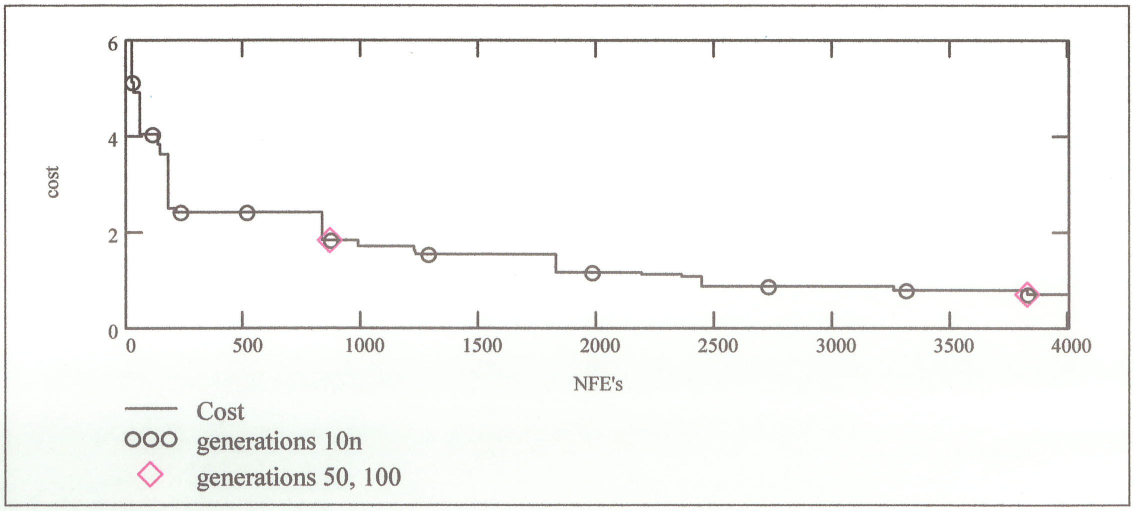


Figure 4.5 *Tracking NFEs (valid airfoil shapes) by generation for the C4/70/C50 design.*

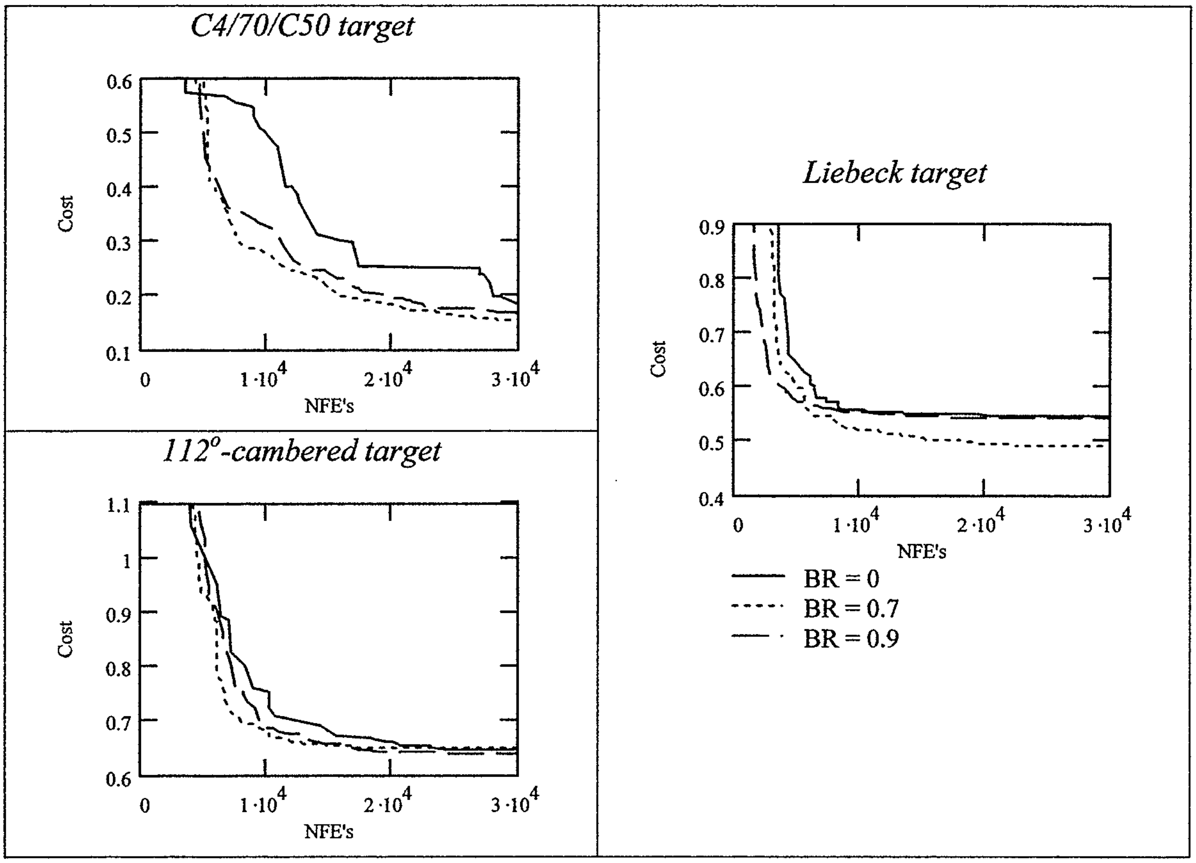
Only 224 valid airfoils are tested after 30 generations. By 50 generations, enough airfoils (860 valid shapes) have been tested that some of the best shapes are in the general region of the solution. Prior to generation 30, however, it is probably of limited value to distinguish between good and bad shapes. They're all bad.

4.2 VBDE Results

Convergence to the three test cases was compared for birthrates *BR* = 0.1, 0.2,... 1.0. Each test uses the BP 3333 parameterization with DE/rand-to-best/1/bin, *CR*=.95, *F*=.85, *NP*=120. Note that *BR*=0 corresponds to the BP 3333 convergence in section 3.5.

Two birthrates – 0.7, 0.9 – resulted in accelerated convergence for all three cases. These are shown in Figure 4.6, with comparison to *BR*=0. In each case, using *BR*=0.7 was the most effective. For the C4/70/C50 and Liebeck targets, all birthrates except *BR*=1.0 improved the performance significantly. Best results in both cases are for 0.7, 0.8, 0.9, with roughly twofold acceleration. Note that a significant cost improvement is observed for the Liebeck target with *BR*=0.7. For the 112°-cambered target, twofold acceleration occurred with *BR*=0.7,0.9, but birthrates smaller than 0.5 resulted in significant degradation, and premature convergence occurred with *BR*=1.0.

Given the small number of valid airfoil shapes generated in early generations, it would seem that any use of VBDE – especially with high values of *BR* – would reduce the population's diversity so much as to result in premature convergence. Yet that was not the case. DE's mutation operator is able to maintain the diversity by sampling the differential from the whole population. In fact, even over the first 3000 FEs, VBDE with high birthrates can show some improvement over DE (Figure 4.7).



C4/70/C50 target

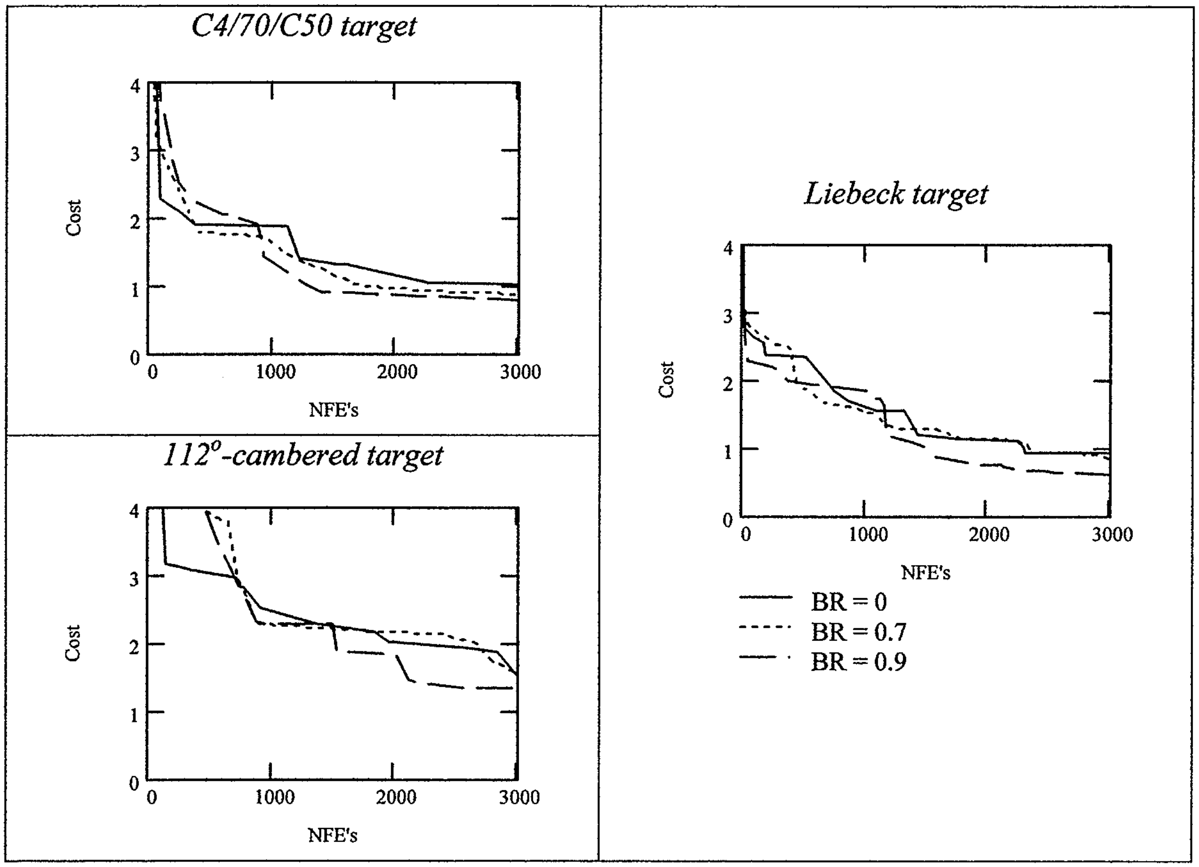
112°-cambered target

Liebeck target

Figure 4.6 *Effect of variable birthrate on convergence.*

4.3 Conclusions

The variable birthrate method for reproductive selection has some potential for acceleration of DE. In general, *BR* values between 0.7 and 0.9 are best, with *BR*=0.7 roughly doubling the convergence rate. That is, it seems to be more effective to neuter a high percentage of vectors. Although most procreating vectors in a generation are then very similar in nature, DE's mutation operation is able to maintain sufficient diversity to energize the search in new directions.



C4/70/C50 target

112°-cambered target

Liebeck target

Figure 4.7 *Effect of variable birthrate over the first 3000 FEs.*

Performance degradation is possible, and was observed both for very low birthrates and for the upper limit of 1.0. This was expected at the upper limit, when the best vector dominates the sub-population of fertile parents, causing "genetic drift". It was not expected, though, that the same would occur with low birthrates. Of course, as long as *BR* is nonzero, constraint violations in phase one have a very small likelihood reproducing. Thus with low birthrates, VBDE quickly eliminates constraint violations, but the best vectors have not been given enough opportunity to procreate. So the population quickly fills up with mediocre but valid shapes, which can increase the number of flow solutions required to find good solutions.

In fact, it was observed that higher birthrates (especially $BR > 0.4$) actually increased the number of constraint violations. That is, later generations contained more invalid shapes with VBDE than without. This could explain the pattern of improved performance for higher birthrates. The search arrives at later generations sooner (i.e. with fewer FEs). This provides greater opportunity for the most promising shapes to reproduce, with less computational expense.

Chapter 5 Acceleration by Hybridization

In this chapter, significant acceleration is achieved by hybridizing DE with a local search method. The Hybridized DE (HDE) algorithm is discussed in section 5.1. The effect on convergence is presented in section 5.2. In section 5.3, VBDE and HDE are combined to determine whether the individual improvements might be cumulative. Section 5.4 presents conclusions and a summary of the acceleration achieved thus far.

5.1 Hybridization of DE with Downhill Simplex

Combining different optimization techniques into a hybrid algorithm has the potential of exploiting the advantages of each technique while masking their deficiencies. In particular, utilizing the speed of a local search to improve EA performance has been a popular suggestion – almost from the beginning (Brady, 1985; Goldberg, 1989; Davis, 1991; Michalewicz, 1992).

Often EAs are combined with a gradient-based local search, in which the derivative of the objective function is calculated to determine the best direction of descent. Vicini and Quagliarella (1999), for example, combined a gradient method with a GA to design airfoils and wings, achieving a 30% - 70% reduction in computation time. Gradient methods have been combined with DE in other applications (Chiou and Wang, 1998; Masters and Land, 1997). Unfortunately the objective function discussed here is non-differentiable due to the penalty method of rejecting infeasible shapes.

Downhill Simplex (DS) (Nelder and Mead, 1965) is a downhill search optimizer that quickly finds local minima, without requiring any knowledge of gradients. In this chapter, DE is combined with DS to create a new algorithm - Hybridized DE (HDE). In a *D*-dimensional optimization problem, the simplex is a *D+1*-sided polygon. After its initial definition, it is modified so as to make its way downhill to a local minimum. Three basic steps are used: reflection, expansion, and contraction of the simplex.

The main mechanism of DS is that of reflection. The highest vertex is reflected through the opposite face of the simplex to a lower (fitter) vertex. This reflection is given by

$$\mathbf{x}_{new} = \frac{2}{D}\left(\sum_{i=1}^{D+1}\mathbf{x}_i\right) - \left(\frac{2}{D}+1\right)\mathbf{x}_{high} \qquad (5.1)$$

where $\mathbf{x}_i$ are the vertices in the simplex. Only downhill steps are accepted. That is, if $C(\mathbf{x}_{new}) > C(\mathbf{x}_{high})$, then $\mathbf{x}_{new}$ is rejected. This has been called the rudimentary steep descent method. Depending on the landscape of the objective function, *C*, the simplex is either expanded to move downhill more quickly or contracted to move more slowly.

The search operations of DE are mutation and recombination, whereas that of DS is reflection. HDE will use all three operations, but this hybridization requires a balance. Reflection sends an individual quickly down local hills, while mutation and recombination use the diversity of the population to ignore the local landscape. HDE should improve convergence without converging prematurely to a local minimum. This can be achieved by using DS only sparsely.

Any generation created by DE can be considered as an intermediate generation. From this intermediate generation, *D*+1 vectors are chosen to form a simplex. Through reflection, the simplex is modified until one (or several) individuals are improved.

Improved vectors are then chosen to move on to the next generation, as shown in Figure 5.1. Diversity can be maintained in two ways. First, DS is used only after every *k* generations. Second, only a few (*Nit*) DS iterations are used at each step.

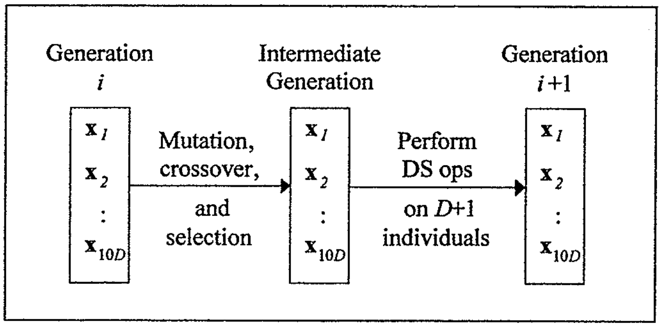


Figure 5.1 *Method of hybridizing DE with DS.*

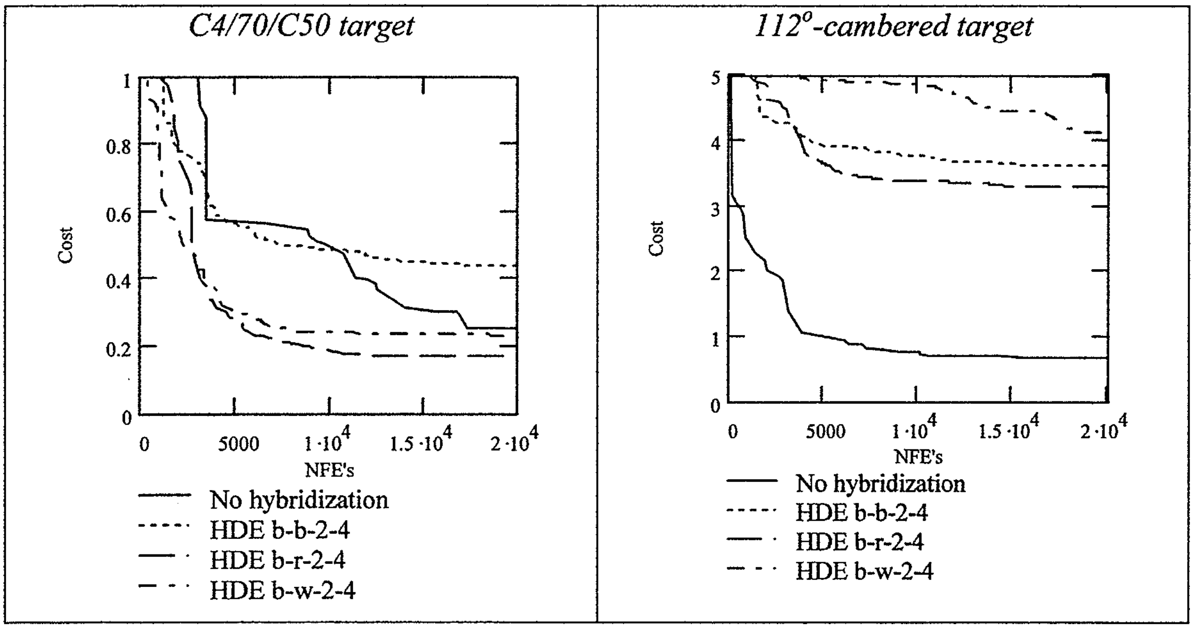
Several strategies can be used for selecting individuals for the initial simplex. Coded into FanOpt are the following possibilities (with short-hand label in brackets): (b) the best D+1, (br) the single best plus D chosen at random, (r) D+1 chosen completely at random. Similarly, there are several strategies for choosing which vectors those improved by DS replace. The D+1 vectors in the new simplex can replace either: (b) the best D+1, (w) the worst D+1, (r) a random selection. Alternatively, the single best simplex vector can replace (1b) the single best vector in the population, (1r) a randomly chosen vector.

To compare the effectiveness of strategies, we will use the following label system. "HDE *s-r-k-Nit* " refers to hybridization with selection strategy *s*, replacement strategy *r*, performing DS every *k* generations for *Nit* iterations. For example, HDE b-b-50-100 selects the best D+1 vectors every 50 generations, performs 100 DS iterations, and replaces the original D+1 vectors by those in the new simplex.

5.2 HDE Results

In a first attempt at hybridizing DE (Rogalsky and Derksen, 2000), we found HDE b-b/r/w-2-4 to be effective strategies for Bezier parameterization with no crossover. In particular, b-r-2-4 significantly improved the convergence rate for all three targets.

Using the BP 3333 parameterization with binary crossover, no clear acceleration pattern could be observed for the same strategies. For example, HDE b-r-2-4 produced tremendous acceleration for the C4/70/C50 case, but bad misconvergence for the 112°-cambered case (Figure 5.2). In fact, all three replacement strategies resulted in premature convergence for the 112°-cambered target.



C4/70/C50 target

112°-cambered target

Figure 5.2 *Effect of the HDE b-b/r/w-2-4 strategies on convergence.*

In Chapter 4, it was shown that there are very few good BP 3333 shapes per population in the early generations. It is understandable, then, that the early use of DS might not be effective. It is better to let DE maneuver itself into a good region, and then give DS the opportunity to run downhill. Thus *k* was increased. When DS is applied less frequently, it must run longer to have any effect, so *Nit* is increased as well.

The following strategies were applied with a variety of selection and replacement simplexes: 5-10, 10-20, 20-50, 50-100, 100-500, 120-500, 140-500. Applying DS every 5, 10, or 20 generations was not consistently effective – occasionally accelerating, but often slowing convergence, and sometimes misconverging. Choice of selection and replacement strategies made very little difference. A long DS run (500 – 1000 iterations) after 100, 120, or 140 generations showed some potential, but had no effect if the population diversity was insufficient, and misconverged if there was too much diversity.

The most consistent strategy was HDE b-b-50-100. In Table 5.1, convergence is compared numerically. Two costs are given for each target. By the first, the design is already excellent, and by the second it has converged completely. (In some cases a further cost reduction is achieved, but the improvement is the interpolation error, not the actual design.) A 2- to 4-fold reduction of NFEs is achieved compared to the BP 3333,

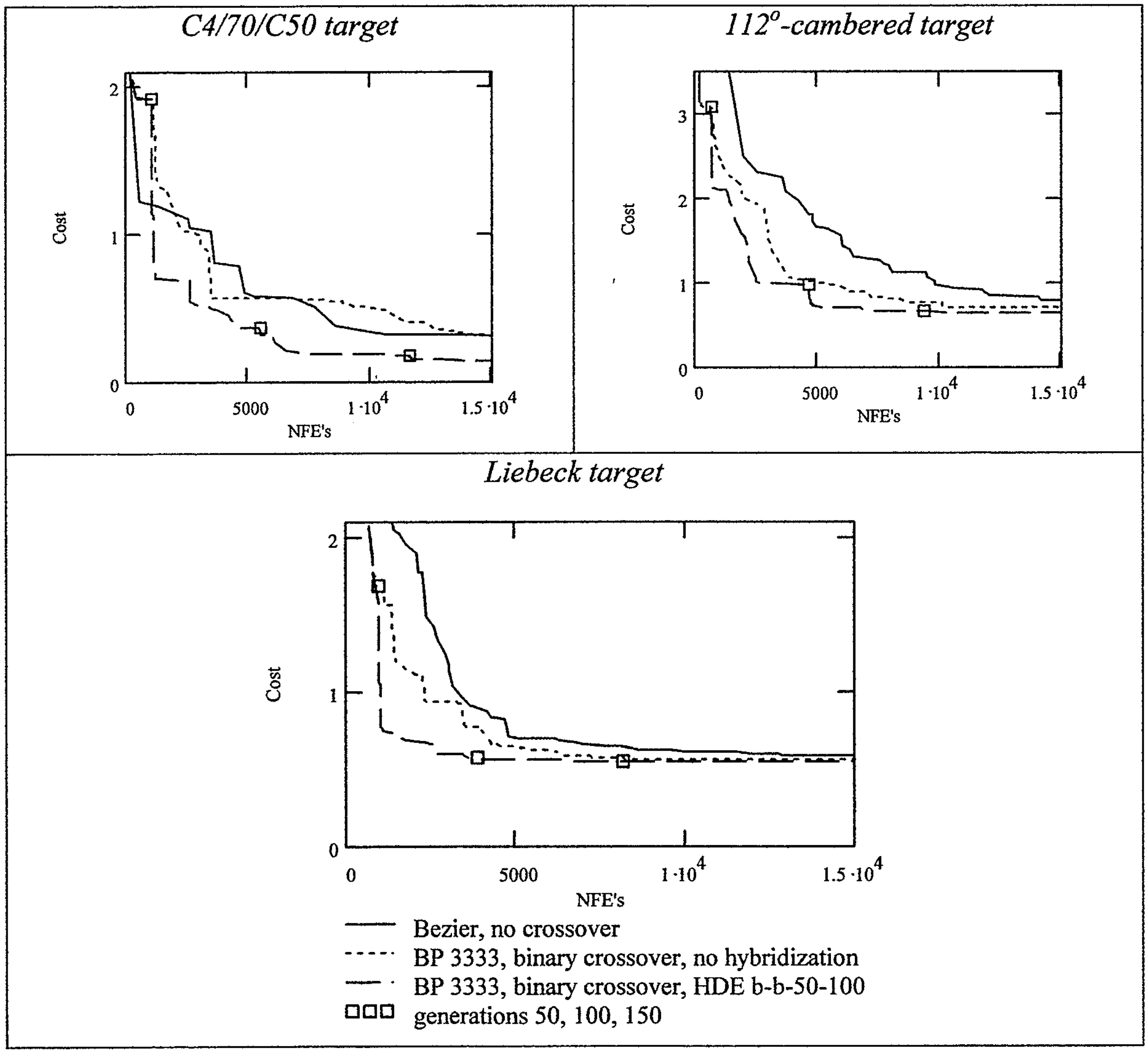
Table 5.1 *Acceleration factors due to hybridization.*

	Cost	BP 3333, binary cr, HDE b-b-50-100	BP 3333, binary cr, no hybridization		Bezier benchmark	
		NFEs	NFEs	Accel factor	NFEs	Accel factor
C4/70/C50	0.200	6,626	28,015	4.2	30,441	4.6
	0.150	12,849	51,383	4.0	45,105	3.5
112°-cambered	0.720	4,827	10,581	2.2	44,263	9.2
	0.650	9,398	24,458	2.6	N/A*	
Liebeck	0.600	2,753	6,254	2.3	11,887	4.3
	0.550	6,422	15,333	2.4	N/A**	

*cost 0.71 at 100,000 FEs
**cost 0.566 at 50,000 FEs

binary crossover convergence, and a 3- to 5-fold reduction with respect to the Bezier benchmarks, with a 9-fold reduction for the 112°-cambered target. Note that, for the 112°-cambered and the Liebeck design, by 10,000 FEs the BP 3333, binary crossover, HDE b-b-50-100 has achieved a lower cost than is ever reached by the Bezier design.

Convergence plots are compared in Figure 5.3. Notice the step-wise convergence pattern for the b-b-50-100 hybridization. The first DS run (after 50 generations) results in



C4/70/C50 target

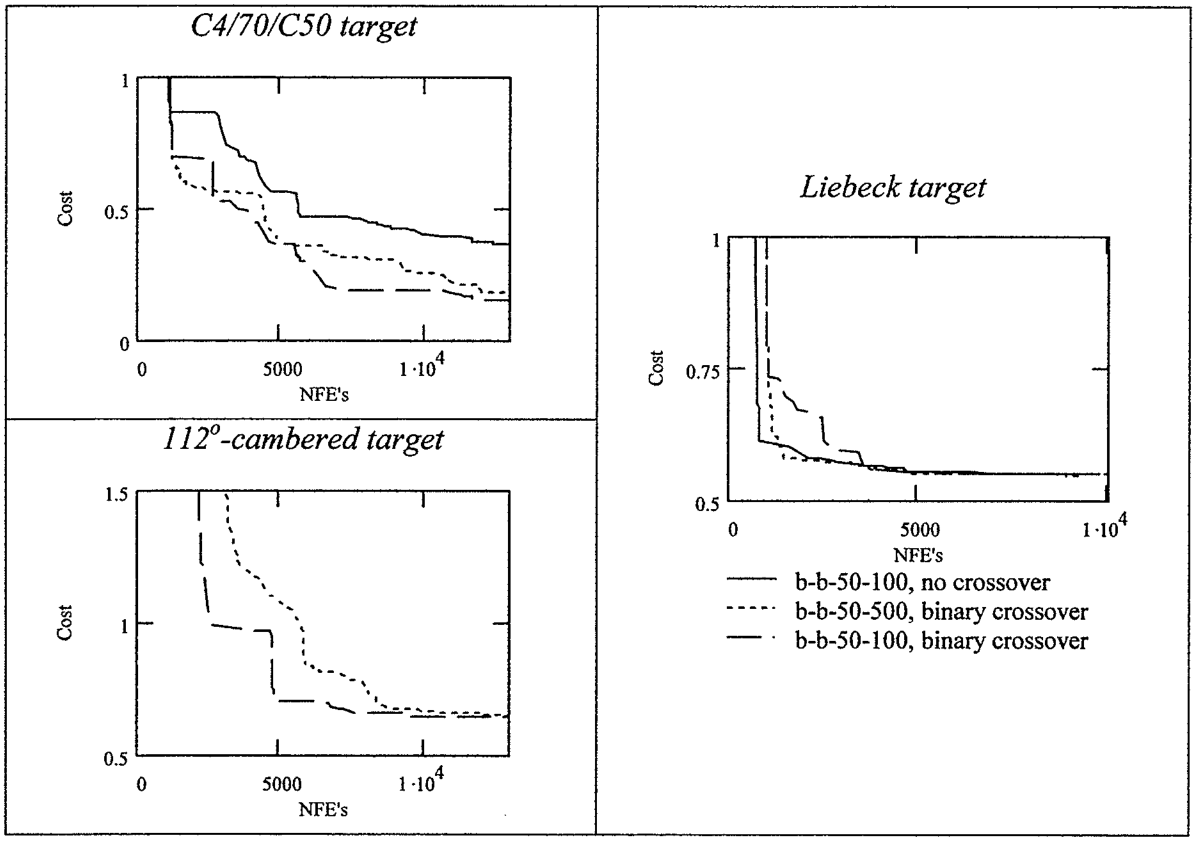
112°-cambered target

Liebeck target

Figure 5.3 *Effect of HDE b-b-50-100 on convergence.*

a dramatic cost improvement. This is followed by a period of stagnation – up to several thousand FEs in which no improvement is seen. By 100 generations, DE has built up sufficient diversity for DS to realize another significant cost jump. At 150 generations, improvements are minor, and the process has essentially converged.

Since 100 DS iterations is so effective, one might expect more to be better. For the Liebeck target, HDE b-b-50-500 (Figure 5.4) does indeed show improvement over the 50-100 strategy. Using 100 iterations, cost is reduced from 1.68 to 0.73. The 400 additional DS iterations reduce it further to 0.58, whereas the 50-100 strategy needs an


C4/70/C50 target

112°-cambered target

Liebeck target

Figure 5.4 *Effect of increasing the number of DS iterations and of removing the crossover operation.*

additional 1,500 FEs to get to the same cost. In the Liebeck case, the design has very nearly converged after the first 500 DS iterations, and there is very little work left for DE.

Although it did improve the Liebeck convergence, 50-500 was actually detrimental for the other two cases. After 100 DS iterations, one vertex of the simplex is already at the bottom of the local valley. Iterating any more only results in the simplex converging around that one point. Thus the 50-500 strategy has less diversity in the population at generation 51, and it takes longer for DE to generate sufficient energy to drive the cost lower. Even in the Liebeck case, it doesn't take long for the 50-100 to catch up to the 50-500. In fact, by 4000 FEs, 50-100 has the lower cost. Thus, it is clear that overall b-b-50-100 is the most effective.

The b-b-50-100 hybridization strategy does not have the same effect when used without crossover (Figure 5.4). It only accelerates the Liebeck convergence, which is already much quicker than the other two. Premature convergence occurs for each of the other targets. (For the 112°-cambered target, the cost converges at 2.2 and is not shown.) The crossover operation helps to spread information throughout the population after a DS run, expanding the focus away from the single valley found by DS.

5.3 Hybridized Variable Birthrate DE (HVBDE)

In Chapter 4, it was shown that VBDE with high *BR* values tended to accelerate DE. Having found that HDE b-b-50-100 also improves DE's performance; we now turn our attention to combinations of the two strategies – HVBDE.

The most obvious combination is to use variable birthrate throughout, together with b-b-50-100. The most successful birthrate cutoffs in Chapter 4 were *BR* = 0.7, 0.9. These and 0.8 were used here in combination with HDE. None produce consistent results. For the C4/70/C50 target, HVBDE with *BR*=0.9 converges prematurely. For the 112°-cambered target, there is misconvergence with both *BR*=0.7 and *BR*=0.8. All three slow down convergence to the Liebeck target. Smaller birthrate cutoffs were tested also, but they consistently produced worse results than HDE alone.

Having no success with the simple combination of strategies, the following refinement was made. We wondered whether VBDE might be more effective if used on a population closer to convergence. (Recall that there are many constraint violations in generations 0-50.) FanOpt was recoded to start using variable birthrate only after the first DS run. Again, this resulted in worse performance for the whole range of *BR* values.

One more refinement was attempted. Recall that VBDE caused a higher rate of constraint violations, resulting in convergence at later generations (but fewer FEs). This could mean that the population is not close enough to a good solution at 50 generations for the HDE b-b-50-100 to be effective. So we tried VBDE with a b-b-75-100 strategy. Using *BR*=0.9, the DS run at generation 75 occurs at under 2000 FEs, and does arrive at a better solution at that point. But the run at generation 150 does not happen until nearly 10,000 FEs, and by then the cost is significantly higher. In an attempt at lowering the cost before generation 150, DS was run at generations 75, 100, 125, 150, with *BR*=0.9 throughout. This didn't seem to provide DE with the diversity necessary to drive the cost down, resulting in convergence slower than HDE b-b-50-100 in some cases, and premature convergence in others.

5.4 Conclusions

The acceleration produced by HDE b-b-50-100 in conjunction with the BP 3333 parameterization and binary crossover is tremendous. Compared to the Bezier benchmarks, the same cost is achieved 3-5 times faster. Convergence occurs by 13,000 FEs, and in two cases the cost at convergence is significantly lower than the benchmark cost at 50,000.

HVBDE was not effective. Variable birthrate strategies tended to degrade the performance of HDE b-b-50-100. The hybridization of DS and DE seems to strike just the right balance of hill-sliding and population diversity, a balance that is disrupted by any introduction of VBDE.

A summary of the convergence results obtained thus far is given in Figure 5.5. The steady progression toward faster convergence is clear. Although in one case (Liebeck) VBDE finds a shape with lower cost, in all cases VBDE converges more slowly than HDE b-b-50-100.

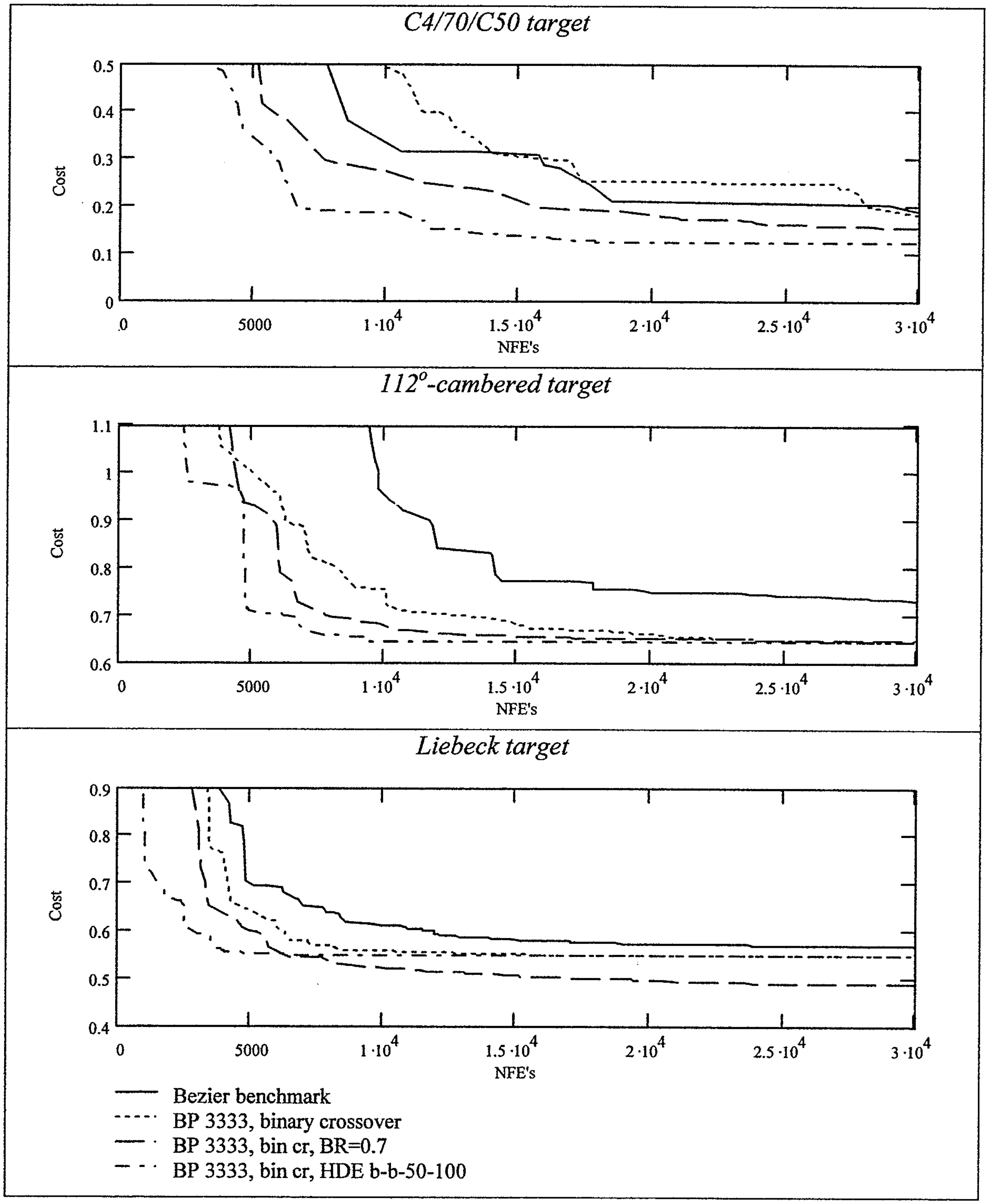


Figure 5.5 *Summary of acceleration due to parameterization, variable birth rate, and hybridization.*

Chapter 6 Immune System Acceleration

This chapter discusses one last acceleration idea, modeled after the immune system. Section 6.1 describes the model. The effect of immune acceleration alone is given in section 6.2. In section 6.3 it is shown to accelerate HDE b-b-50-100, the best results of Chapter 5. Section 6.4 provides a summary of the acceleration obtained in chapters 3 – 6. One additional test case is introduced as verification that the acceleration can be observed for a wide variety of designs.

6.1 Immune System Modeling

The human immune system is designed to protect the body against antigens – foreign microorganisms such as viruses, bacteria, and parasites. One of the key components of the immune system is the antibody. Each antibody has two amino acid polymers hooked together – a Heavy chain and a Light chain. They fold around each other to form an HL-pair. Two HL-pairs connected side-by-side form the antibody molecule. The folding creates a site capable of binding to antigen (Figure 6.1).

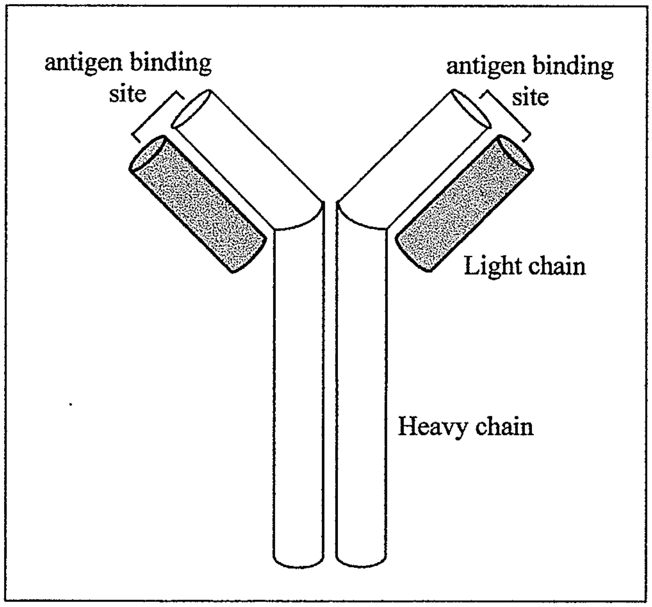


Figure 6.1 *Schematic of an antibody produced by a biological immune system.*

Binding of antibody to antigen ultimately results in the disabling of the antigen's chemical action. Some antibodies combine with plasma to kill the bacteria directly. Others prevent the movement of antigen through cell walls. Still others make the antigen easier to ingest by white cells.

Each antibody can bind only to a specific molecular shape. The immune system maintains a "library" of millions of different antibodies, each of which is produced by a unique B-cell. The B-cell is also able to bind only to that particular shape.

Although binding is particular, it is also approximate. Each antibody is capable of binding with a range of antigens through partial recognition. This allows them to tag antigen that has never before been encountered by the body. It also decreases the number of antibodies required to protect against all possible invaders. It is estimated that the human body contains 10^8 different antibodies, which are able to recognize about 10^{16} different antigens.

A consequence of partial binding is that detection and elimination of a specific antigen can be slow, allowing it to replicate inside the body. So the immune system adapts itself whenever a disease is encountered. When a B-cell is activated through partial binding, it does two things. First, it begins producing antibodies, which themselves bind to the antigen – although only partially. Second, through mutation, recombination, and selection, it begins evolving a population of new B-cells designed specifically to detect and bind to that particular antigen. The more affinity a new B-cell has to the antigen, the more likely it is to replicate itself and produce new antibodies. This process – known as affinity maturation – can be modeled by a GA to accelerate global search.

Just as the mechanisms of biological evolution inspired the development of GA's, so the immune system has been modeled as a problem-solving technique. (For a survey, see Dasgupta, 1999.) In fact the two ideas have been combined (computationally) to evolve antibodies using GA's (Forrest and Perelson, 1991, Hightower et al, 1995) and to accelerate local hill climbing in a GA (Bersini and Varela, 1991). Other applications of artificial immune systems include the solution of scheduling problems (Hart et al, 1998), and protection against computer viruses (Marmelstein et al, 1997).

These ideas have also been implemented for design optimization. Hajela and others developed a GA-based model of the immune system, and used it to solve problems such as optimal design of a truss system (Hajela and Lee, 1996; Hajela et al, 1997; Hajela and Yoo, 1999). The model described in this chapter is patterned after theirs.

In Hajela's model, a GA evolves antibodies with affinity to certain antigens. Antibodies and antigens are encoded as binary strings. An objective function measures the degree to which an antibody matches antigens present in the system. Typically, this would count the number of matching bits between a pair of strings. In the simplest case, when a population of antibodies is exposed to a single antigen, maximization of the objective function results in a population of specialist antibodies that match the antigen. If several different antigens exist, the population can be evolved into generalist antibodies. Each specialist antibody would closely match one of the antigens, whereas a generalist antibody would in some sense cover all antigens.

In FanOpt, the encoding is with real-valued vectors, and DE minimizes the objective function, so some modifications to Hajela's model are required. The following algorithm calculates Immune Cost, $IC(\mathbf{x})$ for the population of n antibodies, given a set of antigens with which to "bind". Vectors with least cost match the antigen pool the best. DE is used to minimize the Immune Cost - resulting in a population of antibodies designed to match the antigens.

Algorithm to calculate Immune Cost

1. Initialize the cost of each antibody to zero: $IC(\mathbf{x}) \equiv 0$.

2. Initialize the number of times selected to zero: $ns(\mathbf{x}) \equiv 0$.

3. Repeat steps a) through d) $3n$ times, where n is the number of antibodies.

   a) Randomly select an antigen, **g**.

   b) Randomly select a sample of N different antibodies, $\mathbf{x}_i$.

   c) The match score of each antibody selected is the ℓ_2-distance from the antigen.

   Add the match score to the immune cost:

$$IC(\mathbf{x}_i) = IC(\mathbf{x}_i) + d(\mathbf{g},\mathbf{x}_i)$$

   d) Increment the number of times $\mathbf{x}_i$ has been selected:

$$ns(\mathbf{x}_i) = ns(\mathbf{x}_i) + 1$$

4. The Immune Cost of each antibody is its total match score normalized by the number of times it was selected:

$$IC(\mathbf{x}) = IC(\mathbf{x}) \,/\, ns(\mathbf{x})$$

This simulation can be used to accelerate the convergence of DE for aerodynamic optimization by biasing the search towards vectors with lowest cost in a given generation.

These are the antigens. The overall structure of Immune Accelerated DE (IADE) – shown in Figure 6.2 – is similar to that of HDE. For a given generation of vectors representing airfoils, the standard DE operations are performed to generate an intermediate generation of (mostly) improved vectors. The best of these are selected as antigens. The worst are selected as antibodies. They are conditioned to match with the antigens selected. Those vectors whose aerodynamic cost is improved by the immune conditioning survive to the next generation.

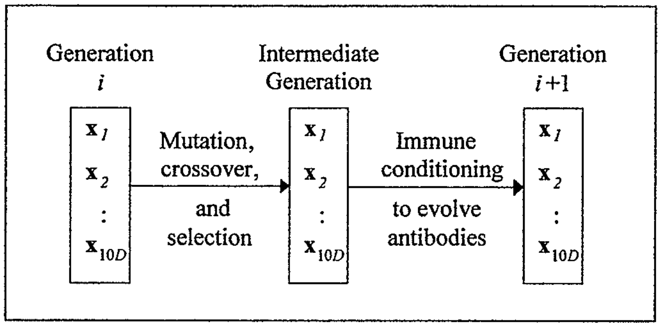


Figure 6.2 *Immune system conditioning to accelerate DE.*

The following parameters control the immune conditioning. The user selects percentages which determine the number of antigens (PGA in the dialog box – Percent antiGen for Acceleration), antibodies (PBA – Percent antiBody for Acceleration), and the sample size (PEA – Percent Exposure for Acceleration). The first two are percentages of the entire population (PGA*120, PBA*120). The last is a percentage of the antibodies chosen (sample size N = PEA*PBA*120). Actual sizes are calculated using the floor function – rounding down to an integer. After every aerodynamic generation, antibodies are evolved for 50 immune conditioning generations, using the DE strategy chosen for aerodynamic optimization. Note that no aerodynamic calculations are made until the end of those 50 generations, so the immune conditioning adds very little relative computational cost.

The following shorthand notation is developed to describe the strategy used for acceleration. IADE *g-b-e* represents Immune Accelerated DE, with *g*% of the population chosen as antigens, *b*% of the population chosen as antibodies, and *e*% of the antibodies chosen for exposure to a specific antigen at a time.

6.2 IADE Results

Hajela et al (1999) suggest using the top 3% of the population as antigens, the entire population as antibodies, and a sampling size of 2% of the population. As long as the sampling size is smaller than the number of antigens, the antibodies should evolve as generalists. This IADE 3-100-2 strategy was tried first. Note that when PBA=100%, there are 2*NP* Cost function calls every generation, so the rate of aerodynamic calculation per generation is doubled. This strategy did not perform well. When it did converge to the right shape, the rate was much slower than that without immune conditioning. Immune conditioning the entire population after each generation seems to reduce the population diversity too much. DE's search becomes less global.

Reducing the number of antibodies should provide a better balance between diversity and acceleration. Several strategies were tested: IADE 3-50-4, 3-25-8, 3-10-20. That is, the number of antigens was held constant at 3 ($\lfloor .03 * 120 \rfloor$), and the sampling size was held constant at 2, but fewer population vectors were immune conditioned after each generation. Overall this resulted in less tendency toward slower or premature convergence, but the rates were at best comparable to those using DE alone.

Immune conditioning after each generation results in the search being biased too strongly toward early-generation vectors. We have already seen that DE does not find reasonably good shapes until about generation 50. But IADE cannot provide the same acceleration power as HDE if run every 50 generations. HDE sends a few individuals sliding rapidly down into the local valley that they surround. IADE can only gather the population closer to the hills surrounding that valley.

6.3 Hybridized Immune Accelerated DE (HIADE)

Although immune conditioning could not by itself accelerate DE for this design problem, the valley analogy suggests a possible use of IADE in conjunction with HDE. After a DS run, there are 13 (D+1) vectors with significantly lower cost than the rest of the population. This leaves sufficient diversity for DE to build up energy to decrease the cost further, but it may not necessarily require so much diversity. Immune conditioning could be effective at gathering the rest of the population closer to the valley found by DS. This suggests performing immune conditioning immediately following each DS run.

Even when immune conditioning every 50 generations, large values for PBA resulted in slower convergence. Furthermore, using only 3% of the population as antigens was not at all effective. Positive results were finally obtained when, noting that DS operates on approximately 10% of the population, PGA was increased to 10%. The best strategy found thus far has been to use 10-10-10 immune conditioning after each DS run.

The result, Hybridized Immune Accelerated DE (HIADE) b-b-50-100 10-10-10, is compared with HDE b-b-50-100 in Figure 6.3. The effect on convergence for the Liebeck design is negligible, but there is significant improvement for the other to cases. The HIADE convergence pattern is more step-wise. The immune conditioning energizes DE to take bigger steps, but also results in some stagnation following those steps.

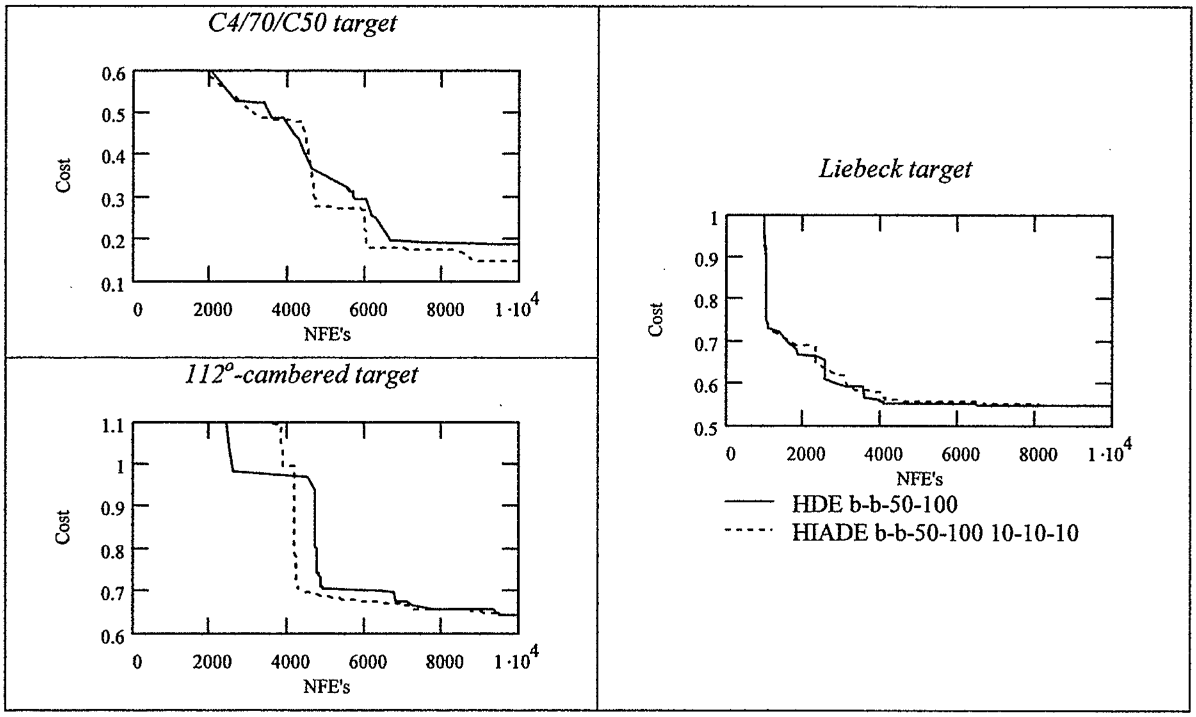


Figure 6.3 *Effect on convergence of HIADE b-b-50-100 10-10-10.*

6.4 Acceleration of DE: A Synopsis

This section summarizes the acceleration obtained. The NFEs required for HIADE to converge is low enough that the addition of a more sophisticated flow solver should be feasible. This has been accomplished without loss of robustness. To confirm that the results are not unique to the three design cases studied, a fourth is introduced, and is shown to follow the same pattern of acceleration.

In chapter 2, a new set of constraints was imposed on the Bezier parameterization. The resulting convergence benchmarks had lower cost than, but similar convergence rate to, the designs obtained in previous work. In chapter 3 we saw that the BP 3333 parameterization with binary crossover had potential for doubling the benchmark convergence rate, although this improvement was not observed for the C4/70/C50 target. In chapter 4, VBDE was shown to have capacity for some further acceleration, but the most significant improvement came in chapter 5 with HDE b-b-50-100. Although the combination of VBDE and HDE was not at all effective, we have seen in the current chapter that an immune conditioning strategy was able to accelerate HDE further yet. The resulting HIADE b-b-50-100 10-10-10 strategy is the one recommended for aerodynamic optimization with BP 3333 parameterization of airfoils.

The progression of cost convergence plots from the Bezier benchmark, to BP 3333, to HDE, to HIADE is shown in Figure 6.4 A five- to ten-fold acceleration is apparent from benchmark to HIADE. The pattern of the HIADE convergence is also significant. In all three cases, cost decreases very rapidly until about 6,000 FEs, at which point the design is already excellent. By 10,000 FEs, any further refinements that might be possible have been completed. The designs at 6,000 and 10,000 FEs are compared in Figure 6.5.

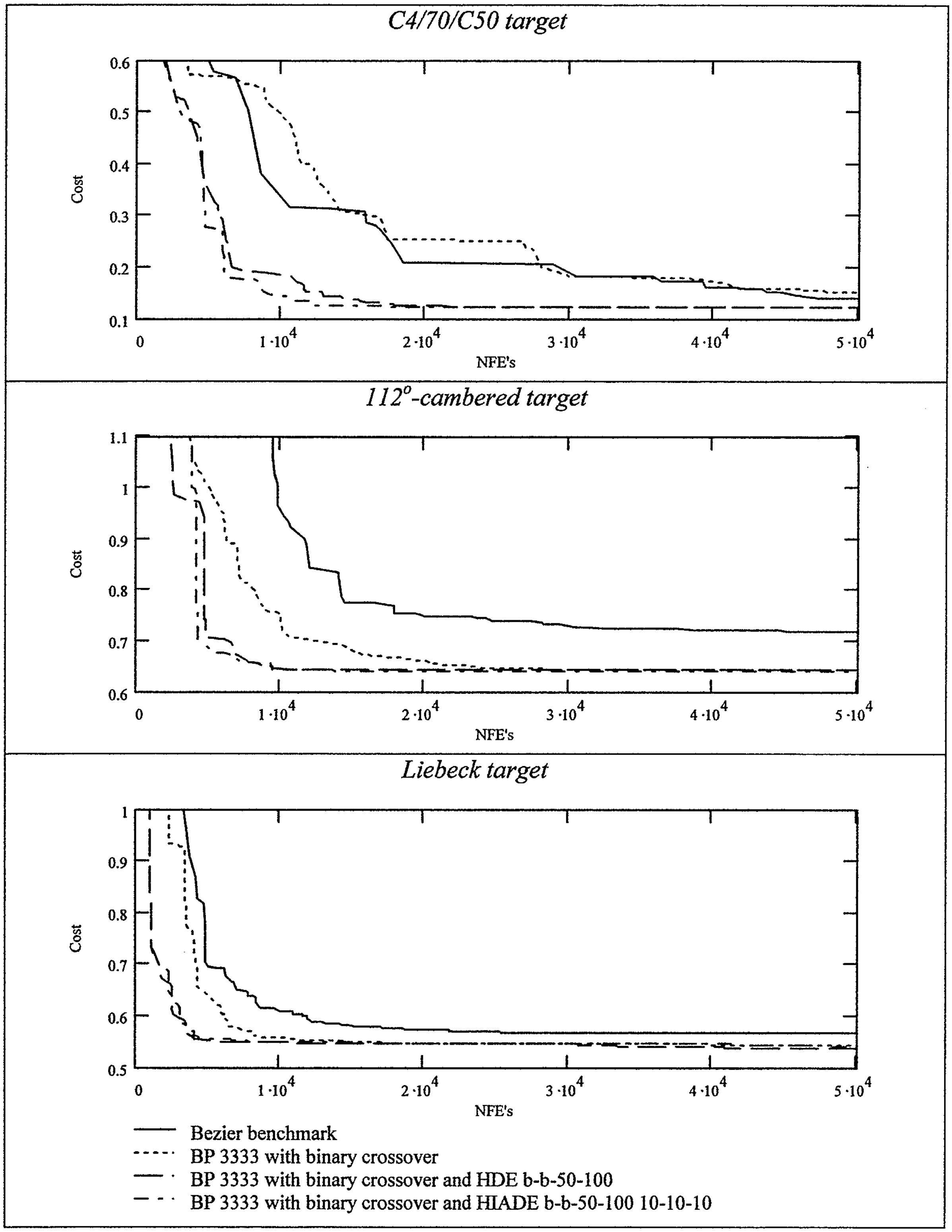
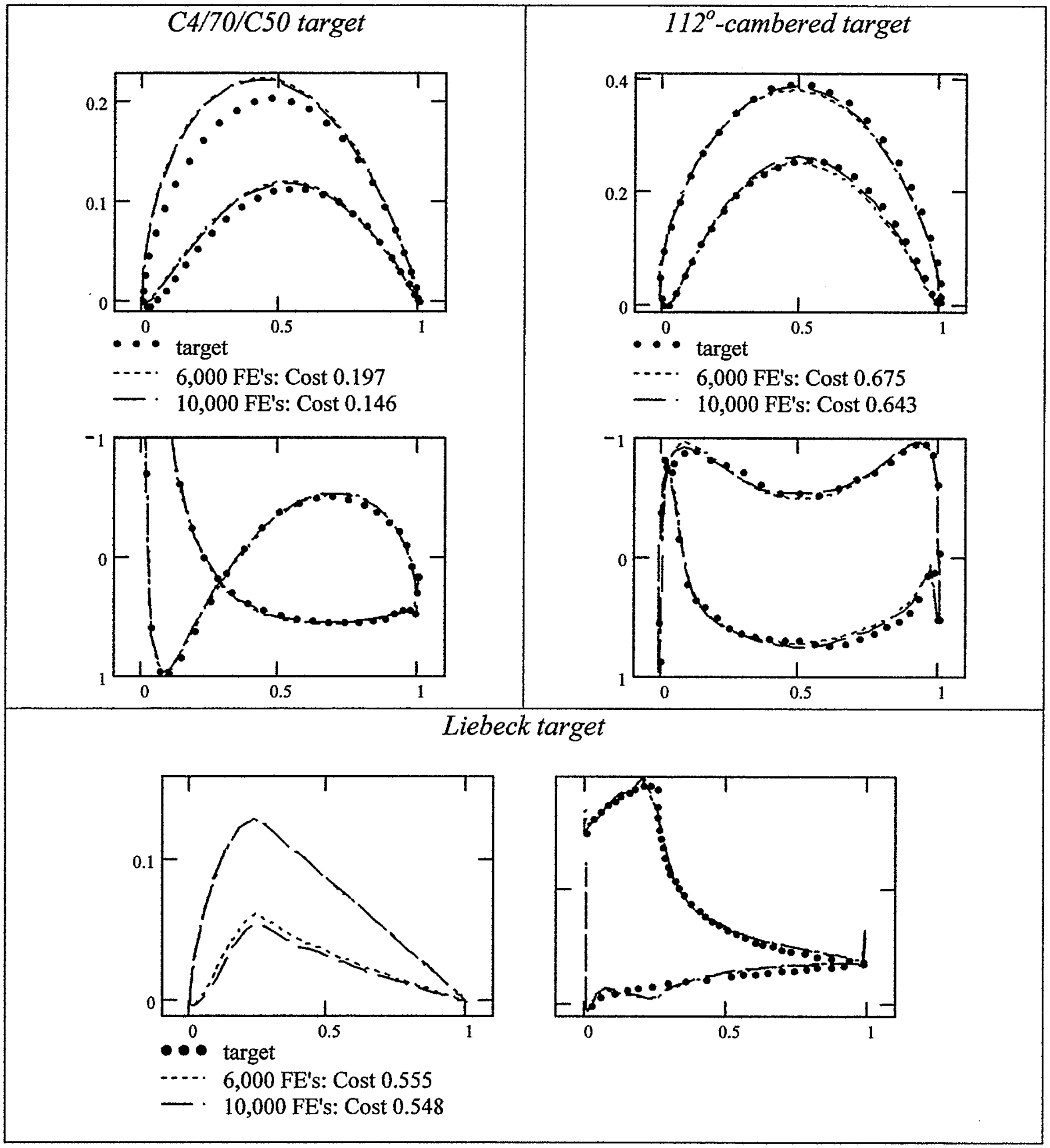


Figure 6.4 *Synopsis of acceleration: From Bezier benchmark to BP 3333 with IAHDE.*


C4/70/C50 target

112°-cambered target

Liebeck target

Figure 6.5 *HIADE designs at 6,000 and 10,000 FEs.*

As evidence that the acceleration is replicable across a broad range of aerodynamic design problems, a fourth design case is considered. The Eppler E850 airfoil is designed for the tip of a propeller. A pressure distribution is calculated and used as the design target. For the surface vorticity method to generate accurate results, one condition is that the upper and lower surface data points lie in pairs, mirrored about the camber line. Unfortunately, this is not true for the Eppler data. It is, however, true for any blade generated by our design algorithm. Since the E850 has been represented by the BP 3333 parameterization in Chapter 3, we use that representation as the design target. The flow was computed for λ=0, t/l=1, and β_1=5°. Outlet angle was computed to be β_2= -3.36876°. The same acceleration pattern can be observed in Figure 6.6. By 6000 FEs, the HIADE cost is 0.05, and it converges to 0.005 shortly after 10,000. Designs are compared in Figure 6.7.

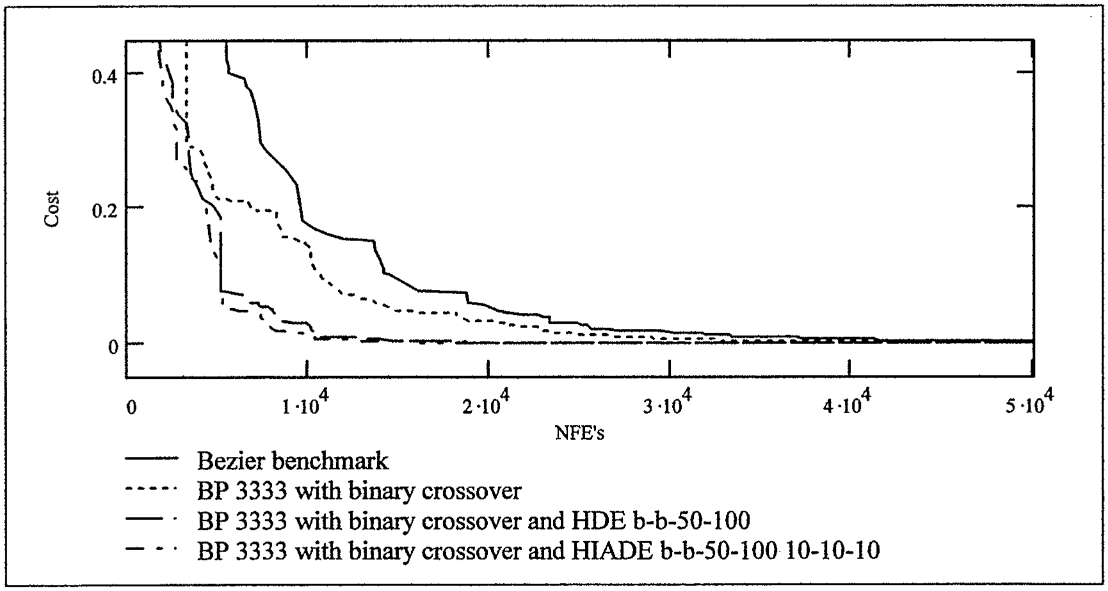


Figure 6.6 *Acceleration pattern replicated for a new design target: the Eppler E850 propeller.*

Convergence factors for all four cases are shown in Table 6.1. Two cost values are chosen for each case. By the first, the design is excellent, and by the second it has very nearly converged. Compared to the Bezier benchmark, the HIADE design convergence is

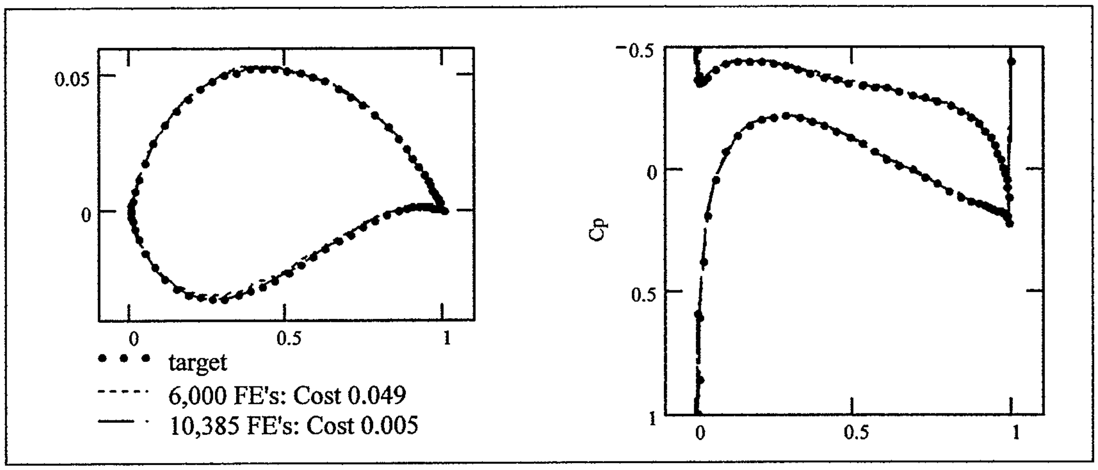


Figure 6.7 *HIADE design of theE850 airfoil at 6,000 and 10,000 FEs.*

faster by a factor of 4 to 10, with an average of 5.8. This takes into account both the new parameterization and the actual acceleration of DE. Comparing convergence for the same parameterization, we find that HIADE is two to six times faster than DE, with an average of 3.2. This second rate should be observable across a broader range of problems than merely aerodynamic optimization.

Table 6.1 *Acceleration factors due to HIADE for four case studies.*

	Cost	BP 3333, binary crossover, HIADE b-b-50-100 10-10-10	BP 3333, binary crossover, DE		Bezier benchmark	
		NFEs	NFEs	Accel factor	NFEs	Accel factor
C4/70/C50	0.200	5,994	28,015	4.7	30,441	5.1
	0.150	8,791	51,383	5.8	45,105	5.1
112°-cambered	0.720	4,224	10,581	2.5	44,263	10.5
	0.650	8,792	24,458	2.8	N/A*	N/A
Liebeck	0.600	3,153	6,254	2.0	11,887	3.8
	0.566	4,042	8,206	2.0	34,269	8.5
E850	0.050	5,271	14,796	2.8	20,449	3.9
	0.005	10,385	28,972	2.8	38,803	3.7
			Average acceleration factor:	3.2		5.8

*cost 0.71 at 100,000 FEs

Chapter 7 Handling Soft Constraints

There are two types of constraints that can be imposed in aerodynamic design. The first is a hard constraint, usually imposed by a barrier method in which no infeasible solution is considered. An example is the set of constraints imposed in chapter 2, which ensured that any shape considered is a valid airfoil. The second type is a soft constraint. An aerodynamic designer will often have engineering requirements limiting the final design. For example, a minimum thickness may be required for structural reasons, or a minimum leading edge radius for off-design performance. There is usually a slight tolerance to such requirements. A design that is slightly outside a predetermined solution space may be acceptable if it has other desirable characteristics. This chapter will demonstrate two techniques for soft-constrained DE search. Constrained search is an important feature of aerodynamic design, but is not the main focus of this thesis, so this chapter will be fairly brief.

7.1 Penalty function

The standard manner in which constraints are imposed in GA search is to use a penalty function. In chapter 2, the penalty function simply applied a large random value to any infeasible designs. This is a form of a barrier method. We don't bother calculating the flow around a shape that isn't even an airfoil. The penalty applied is large in an attempt to remove all infeasible solutions from the population relatively early in the search.

The barrier method is acceptable for hard constraints. (The flow solution around an airfoil with negative thickness is simply meaningless, for example, so it doesn't make much sense to assign a lower penalty to airfoils with less negative thickness.) However, it is not effective for imposing soft constraints. Optimal solutions frequently lie on or near the boundary of the solution space. Rejecting outright any vector that lies outside the constraint boundary steers the search too far away from that boundary (Smith and Tate, 1993; Michalewicz, 1995a; Coit et al, 1996).

The most common penalty function approach to handling soft constraints is due to Richardson et al (1989). First calculate the cost of the vector as if it did lie within the solution space. Then, add to the vector's cost a weighted distance from the constraint boundary. Thus the search is allowed to wander outside the solution space, but the further from the boundary a vector lies, the less fit it becomes. This idea is similar to the Lagrangian relaxation method for combinatorial optimization problems (Avriel, 1976; Fisher, 1981), which temporarily relaxes the most strident constraints in early stages of the search.

This distance-based approach is very easy to implement with the BP 3333 parameterization. Upper and lower bounds can be imposed on any BP 3333 parameter, x_i. The user provides a weight, w_i, which determines how far beyond the boundary DE will search. The penalty function is

$$P(\mathbf{x}) = \sqrt{\sum \left[w_i (b_i - x_i) \right]^2} \, , \tag{7.1}$$

where parameter x_i has exceeded the bound b_i, and the sum is taken over all constraint violations. The total cost of vector $\mathbf{x}$ is then

$$\text{Cost} = C(\mathbf{x}) + P(\mathbf{x}), \tag{7.2}$$

where $C(\mathbf{x})$ is the aerodynamic cost defined in equation (1.3), p 16.

The difficulty of this approach is in determining appropriate weights. Too strong, and DE will stay away from any solutions near the boundary. Too weak, and the search could converge well beyond the boundary. Factors to consider include the relative scaling of the parameters being constrained, the difficulty of satisfying a constraint, and the seriousness of a constraint violation. Often some experimentation is required. Alternative approaches include the dynamic penalty function, in which the severity of the penalty increases as the search progresses (eg. Joines and Houck, 1994), and non-penalty-based constraint handling (Michalewicz and Janikow, 1991; Michalewicz, 1995b), but these are not explored here. Hajela and Yoo (1999) propose the use of an immune network (similar to that described in chapter 6) to handle constraints. This method was coded into FanOpt, but has not yet been tested thoroughly.

7.2 Constrained Search: An Example

To demonstrate the use of the penalty function for soft-constrained search, the Liebeck design will be refined. Notice the large curvature on the lower surface of the BP 3333 Liebeck blades. This corresponds to a bump in pressure coefficients. (See, for example, Figure 6.5, p 130) We will attempt to smooth the pressure distribution imposing a maximum (absolute) curvature on both thickness and camber profiles. Constraining $|\kappa_c|$ in [0, 0.5] and $|\kappa_t|$ in [0,1], both with weight 10, resulted in a design with a very sharp leading edge. While this may in fact work best under ideal circumstances, a slight change in wind direction, or the presence of foreign material on the blade could result in severe degradation. To improve off-design performance, a minimum leading edge radius constraint was imposed as well.

Designs for the following two constraint sets are compared in Figure 7.1.

$$\begin{cases} |\kappa_c| \leq 0.5, \ w_1 = 10 \\ |\kappa_c| \leq 1.0, \ w_2 = 10 \\ |r_{le}| \geq 0.01, \ w_3 = 10 \end{cases} \tag{7.3}$$

$$\begin{cases} |\kappa_c| \leq 0.5, \ w_1 = 1 \\ |\kappa_c| \leq 1.0, \ w_2 = 1 \\ |r_{le}| \geq 0.01, \ w_3 = 100 \end{cases} \tag{7.4}$$

Notice that the constraints themselves are the same, but the weighting is different. Both sets result in a design with a smoother pressure distribution. The first (7.3) does not weight the leading edge radius deviation high enough for the final design to lie within the boundary. The result has $|r_{le}| = 0.002$, with a cost of 0.488. The second set (7.4) weights the leading edge constraint two orders of magnitude higher than the curvature constraints. The result is a design lying within the constraints, with $|r_{le}| = 0.0106$, and a cost of 0.459. In both cases, the cost is significantly lower than that with no constraints (0.55).

Designs shown in Figure 7.1 were obtained after 10,000 FE's of the HIADE b-b-50-100 10-10-10 acceleration strategy. Convergence rates for the constrained designs are compared with the HIADE convergence rate for the unconstrained design in Figure 7.2. Constrained design is slower over the first 3,000 FE's, but the 6,000 – 10,000 pattern is still observed. That is, the design is excellent by 6,000 FE's, and has converged by 10,000.

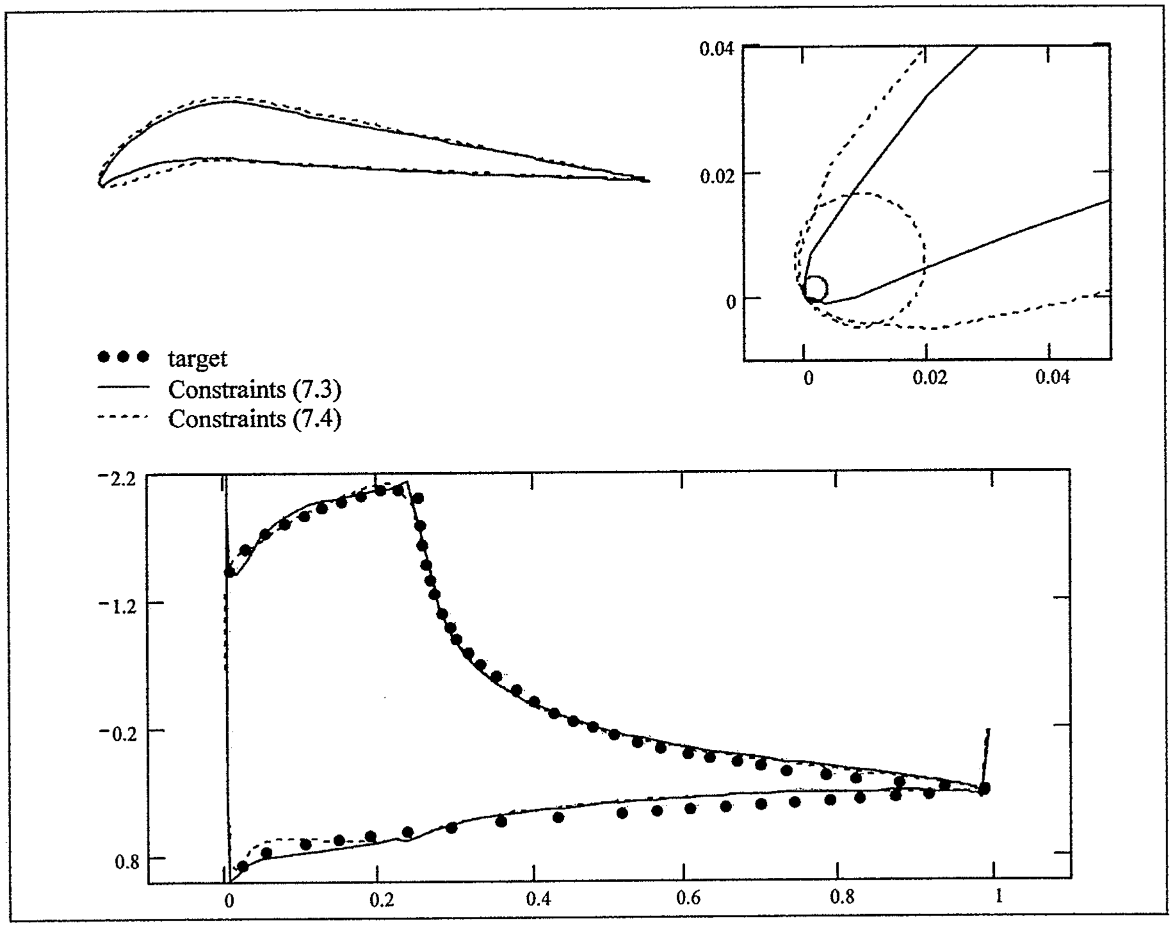


Figure 7.1 *Constrained Liebeck designs.*

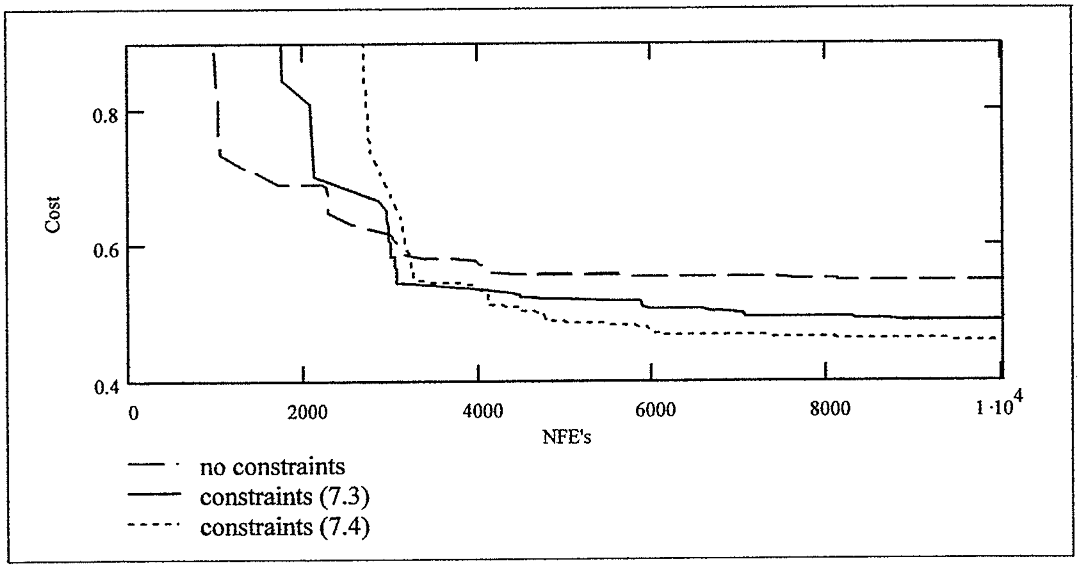


Figure 7.2 *Unconstrained vs. constrained HIADE convergence.*

Chapter 8 Conclusions and Recommendations

Aerodynamic design optimization for fans has as its ultimate goal the design of a more efficient fan. This thesis has contributed toward that goal by enhancing FanOpt – the software package developed in Rogalsky (1998) – to accelerate the design process. A new parameterization for airfoils has been developed, and the number of function evaluations required by Differential Evolution has been reduced significantly. This concluding chapter will summarize the results obtained, and recommend future work, in three areas: 1) parameterization of airfoils, 2) acceleration of DE, and 3) aerodynamic design optimization.

8.1 Airfoil Parameterization

It has been recognized for some time that GA's require problem-specific information to converge quickly. In the current optimization problem, airfoil parameterization can impact both robustness and convergence rate. The Bezier-PARSEC parameterization was designed to enhance both.

Present in the previously-used Bezier method (Rogalsky, 1998) were discontinuous second derivatives, constraints that limited the size of the solution space, and parameters with nonlinear interactions. All three issues are addressed by the BP 3333 and 3434 parameterizations. Curves are joined with second order continuity; control point constraints are no longer required; and new aerodynamic parameters are incorporated. Bezier curves are still used to create the airfoil, but the Bezier control points are now defined by aerodynamic parameters. DE operates on the new parameters, which reduces epistasis in the design objective function. BP 3333 uses exclusively aerodynamic parameters, which define four degree three Bezier curves. BP 3434 uses degree four curves on the trailing profiles, giving it freedom to reproduce more complicated shapes, but requiring a mixture of aerodynamic and Bezier parameters.

To compare the robustness of the Bezier and BP parameterizations, 63 airfoils were represented, using DE to minimize the deviation between the parameterization and the target airfoil. All three methods successfully represented a high percentage, with BP 3434 having the most success. The second-order discontinuity in the Bezier parameterization occasionally showed up as a sharp edge in the airfoil. BP 3434 failed once when the solution was far from the initial population, converging to a shape with a sharp edge before it could find the correct airfoil. BP 3434 also showed a penchant for wildly erratic control points – producing airfoils with tiny loops, and trailing edges that, when magnified sufficiently, begin in the wrong direction.

BP 3333 correctly reproduces 94 % of the airfoils chosen, but there are airfoils that it cannot represent well. Because it uses degree three trailing Bezier curves, the inflection point is not variable. Any shape with strong variation in trailing curvature presents problems. This includes airfoils with cusps (indicating a strong inflection point near the end of the thickness profile), and those with camber profiles that dip below the *x*-axis. The relatively small sacrifice in representation ability seems reasonable to accept in light of the advantages over BP 3434. It has fewer nonzero parameters and fewer total parameters, which will contribute toward acceleration of design optimization; and it avoids sharp edges, steering the search toward true aerodynamic shapes.

The BP 3333 parameterization was used with DE in three inverse design cases. These three represent a broad range of pressure distributions and blade shapes. In all three, a good solution was found, indicating the robustness of BP 3333 for design optimization. Using DE with binary crossover, in two cases BP 3333 convergence was about twice as fast as that for Bezier with no crossover. In the third case, convergence rates were about the same. Binary crossover generally is effective only for separable (or at least non-epistatic) objective functions. It improves the convergence characteristics of BP 3333 optimization due to the increased linearity of this parameterization.

For future work, several recommendations can be made. First, the representation ability of the BP 3333 parameterization should be examined more thoroughly. Some of its limitations are known, but there may be others.

Second, the initial population for optimization could be completely redefined. Currently, it randomly selects parameters from within user-defined bounds. Instead, a population of known airfoils could be used – for example the reproductions found in section 3.4 – or a combination of known airfoils and random sets of parameters. This would certainly reduce the number of constraint violations in early generations, but it is unknown how convergence speed might be affected.

Third, design optimization convergence should be examined for the BP 3434 parameterization. This thesis concentrates on accelerating DE for use with BP 3333. BP 3434 enlarges the design space, but initial indications are that it slows convergence. It may be possible to accelerate BP 3434 convergence enough for it to be useful as well.

8.2 DE Acceleration

In combination with the new parameterization of airfoils, DE itself is modified to obtain accelerated convergence. Three strategies are used. First, the rate at which individuals "give birth" is made variable in VBDE. Only the fittest members of the population are guaranteed to generate trial vectors through mutation and crossover. Second, DE is hybridized with DS, a local search algorithm, in HDE. Together, the two can quickly search local valleys without becoming stuck in a local minimum. The third modification is inspired by the immune system. In each generation of IADE, the worst vectors are conditioned to take on characteristics of the best.

VBDE shows some potential for acceleration. This is especially true for high values of *BR* (when most population members have a low probability of reproducing). For lower values, the population tended to converge prematurely. It was observed that DE only begins to find reasonably good solutions at about generation 50, which likely explains the lack of effect of variable birthrate. It depends on good solutions to improve the quality of the whole population, but in the early stages, there aren't any good solutions. Use of a variable birthrate increased the number of hard constraint violations, so that convergence occurred at even later generations. The acceleration observed was not as significant as that obtained with hybridization. When the two were combined, the resulting VBHDE tended to converge prematurely. VBDE is not the recommended strategy for aerodynamic optimization, but it may be useful for other problems.

Hybridization is by far the most effective accelerator, but only when DS is used sparsely. Any strategies that performed local search in early generations were not effective. Again, this is understandable, given the multitude of costly solutions early in the search. The increased linearity of BP 3333 lends itself well to local optimization, but only when the focus has narrowed on to some reasonably good solutions. The most effective strategy was HDE b-b-50-100, which leaves DE alone for 50 generations, then performs 100 iterations on the best D+1 vectors. By 50 generations, DE has found some very promising shapes. The 100 iterations of DS is just sufficient for some simplex vertices to slide deep into the nearest valley, but also few enough for the simplex to maintain a diversity of improved solutions. DE uses this diversity to find even lower valleys until the next DS slide 50 generations later. By the third slide (at generation 150), the process has very nearly converged – requiring from 6,500 to 13,000 flow solutions.

IADE does not show any consistent acceleration pattern when used on its own. In fact it makes premature convergence more likely. Immune conditioning biases the search toward the most promising solutions in early generations, but these are not good enough to lead the search in the right direction. However, when used sparsely together with hybridization, immune conditioning is able to accelerate even the convergence rate of HDE. The combined strategy – HIADE b-b-50-100 10-10-10 – results in acceleration by a factor of four to ten when compared with the benchmark convergence rates. By 6,000 flow solutions, HIADE has found an excellent solution, and by 10,000 it has converged.

The acceleration observed is actually due to a combination of four factors: 1) BP 3333 parameterization, 2) binary crossover. 3) local search every 50 generations, and 4) immune conditioning after every local search. The effect of the last two alone is to accelerate convergence by a factor of two to six, a speed up that should be observed for other objective functions as well.

For future work, several refinements of these acceleration strategies are suggested, some of which may improve performance yet more. The first is adaptive hybridization. After the first hill-slide at 50 generations, the population will have very many good solutions. Using DS more frequently after that may increase the convergence rate of this second stage of the search. A strategy such as br-1b-2-10 is suggested. The random selection in the initial simplex should send DS down into new valleys, and replacing only one vector will not decrease population diversity.

Second is a modification to VBHDE. Even though VBDE showed better convergence characteristics than DE alone over the first 2,500 FEs, combining it with HDE was either slower than HDE, or resulted in premature convergence. One possible reason is that VBDE converges at later generations. A 75-100 strategy was attempted. This showed promise initially, but the second DS slide (after 150 generations) didn't occur until after 10,000 FEs. Between generations 75 and 150, there are significantly fewer constraint violations, and thus higher NFEs. A better VBHDE strategy may be to run DS after a fixed number of FEs (for example after 2000, 5000, 10,000), rather than a fixed number of generations.

Finally, the following IADE recommendations are made. First, Immune conditioning was not effective when used after every generation. This biases the search too strongly. Using it more sparsely – after every *k* generations – might be more effective. Second, the algorithm currently introduces all conditioned antibodies into the next generation. An alternative is to use the entire population for antibodies ($PBA =$ 100%), but only introduce the best few into the next generation. Third, instead of selecting a fixed percentage of antigens and antibodies, birthrates could be imposed. For example, vectors with birthrates higher than 0.9 could be selected as antigens.

8.3 Aerodynamic Optimization

The advantage of using DE for aerodynamic optimization is its ability to find global solutions. This means that fans can be designed without prior bias toward existing blading. Ultimately, this could result in an unanticipated shape with superior performance. By improving airfoil representation and accelerating convergence, significant steps have been made toward realizing that potential.

The new airfoil parameterization improves considerably the quality of the solution space for optimization. BP 3333 removes many of the unnatural constraints used by Bezier parameterization, allowing many additional potentially valuable designs to be investigated. It also places more natural constraints on the shape, such as second-order continuity. Thus fewer non-aerodynamic shapes are considered. The result is a smaller solution space with a higher proportion of valuable shapes.

This new solution space is not only easier to search, it is also easier to constrain, which is key to a useful design algorithm. A simple penalty method for imposing soft constraints has been demonstrated. For future work, this soft constrainer should be examined more thoroughly. The single experiment conducted suggests that convergence rate is not increased dramatically, but this may not be true in general. Other constrainers, such as the immune system method, may perform better. There may be additional categories of constraints that should be included.

It has been shown earlier (Rogalsky et al, 2000) that DE is too slow to be used with anything but the simplest aerodynamic model – that of potential flow. By ignoring viscosity in the search for the best design, the strength of the design algorithm is reduced in two respects. First, it is possible that the true viscous effects for the final design are not minimal, in which case the design may not in fact be optimal. Second, low drag will be a key feature of any efficient fan, but without viscosity, drag cannot be calculated, and so low drag cannot be used as a design objective.

Hence the need for acceleration. We are confident that the acceleration achieved is applicable broadly for aerodynamic optimization. Convergence rate for the C4/70/C50 design, which is very representative of turbine blading, was increased five-fold. The acceleration pattern was observed for a broad range of design targets. This includes a fourth case – a propeller design – chosen after the acceleration strategies had been developed.

In all four cases, accelerated convergence occurred in under four minutes on a desktop PC that is no longer state of the art (Pentium 4, 1.6GHz, 256MB RAM). Very rough estimates suggest that a boundary layer solution could multiply computation time by three orders of magnitude. If so, by the end of the second day of computation (6,000 FEs), the user will know whether convergence will occur. By the end of the third (10,000

FEs), convergence will be complete. Using today's state of the art, complete convergence would take less than a day.

Thus our final recommendation is to supplement the flow solver with a boundary layer solution. The Keller box method, for example, could be attached seamlessly to the vorticity panel method. The result will be a more useful aerodynamic optimization method, capable of considering both flow separation and drag.

Bibliography

Abbot, I.H., and von Doenhoff, A.E. (1959). *Theory of Wing Sections*. Dover Publications, New York.

Avriel, M. (1976). *Nonlinear Programming: Analysis and Methods*. Prentice-Hall, Englewood Cliffs, NJ.

Balsa-Canto, E., Alonso, A.A. and Banga, J.R. (1998). "Dynamic optimization of bioprocesses: deterministic and stochastic strategies." *Proceedings of ACoFoP IV (Automatic Control of Food and Biological Processes),* 21-23 Sept., Göteborg, Sweden. Available via `ftp` from ftp://nautilus.iim.csic.es/pub/jrbanga/do_bio.ps.

Bersini, H. and Varela, F.J. (1991). "The immune recruitment mechanism: A selective evolutionary strategy." in R.K. Belew, L.B. Booker (eds.), *Proceedings of the Fourth International Converence on Genetic Algorithms*. Morgan Kaufmann, pp 520-526.

Beyer, H.G. (1997). "Local performance measures: Evolution Strategies and Evolutionary Programming." In: Back, T., Fogel, D.B., and Michalewicz, Z. (eds.), *Handbook of Evolutionary Computation*. IOP Publishing and Oxford University Press, Bristol and NewYork, pp. B2.4.1-19.

Beyer, H.G. (2001). *The Theory of Evolution Strategies*. Natural Computing Series. Springer, Heidelberg.

Beyer, H.G., Schwefel, H.P., and Wegener, I. (2002). "How to analyze Evolutionary Algorithms." *Theoretical Computer Science*, Vol. 287, pp. 101-130.

Bezier, P. (1972). *Numerical Control – Mathematics and Applications*. Translated by Forrest, A.R. and Pankhurst, A.F. John Wiley & Sons, London.

Bezier, P. (1974). "Mathematical and Practical Possibilities of UNISURF." In: Barnhill, R.E. and Riesenfeld, R.F. (eds.). *Computer-Aided Geometric Design*. Academic Press, New York, pp 127-152.

Bienert, P. (1967). "Aufbau einer Optimierungsautomatik für drei Parameter." Dipl.-Ing. Thesis, Technical University of Berlin, Institute of Measurement and Control Technology.

Birckelbaw, L. (1989). "Inverse Airfoil Design using the Navier-Stokes Equations." *AIAA Paper* 89-2202.

Brady, R.M. (1985). "Optimization strategies gleaned from biological evolution." *Nature*, Vol. 317, pp 804-806.

Burgreen, G.W., Baysal, O., and Eleshaky, M.E. (1992). "Improving the Efficiency of Aerodynamic Shape Optimization Procedures." *4th AIAA/USAF/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization.* Cleveland. *AIAA Paper* 92-4697.

Chan, Y.Y. (1998). "Applications of Genetic Algorithms to Aerodynamic Design." *Canadian Aeronautics and Space Journal*, Vol. 44, No. 3, September, pp 182-187.

Chang, C.S. and Du, D. (1999). "Further improvement of optimisation method for mass transit signalling block-layout design using differential evolution." *IEE*

Proceedings on Electric Power Applications, Vol. 146, No. 5, September, pp 559–569.

Chang, I., Torres, G.J., and Tung, C. (1995). "Geometric Analysis of Wing Sections." NASA TM 110346.

Chiou, J.P. and Wang, F.S. (1998). "A hybrid method of differential evolution with application to optimal control problems of a bioprocess system." *IEEE International Conference on Evolutionary Computation Proceedings*. IEEE World Congress on Computational Intelligence, 1998, IEEE, New York, pp 627–632.

Coiro, D.P. and Nicolosi, F. (1995). "Design and Optimization of Glider Components." *Technical Soaring*, Vol. 19, No 2.

Coit, D.W., Smith, A.E., and Tate, D.M. (1996). "Adaptive penalty methods for genetic optimization of constrained combinatorial problems." *INFORMS J. Comput.* Vol. 8, pp 173-82.

Dasgupta, D. (ed.) (1999). *Artificial Immune Systems and their Applications*. Springer-Verlag.

Davis, L., ed. (1991). *Handbook of Genetic Algorithms*. Van Nostrand Reinbold, New York.

Davis, P.J. (1963). *Interpolation and Approximation.* Blaisdell Publishing Company, New York.

De Jong, K. (1975). "An Analysis of the Behavior of a Class of Genetic Adaptive Systems." Ph.D. Thesis, University of Michigan.

Eppler, R. (1957). "Direct calculation of airfoils from pressure distribution." NASA TT F-15,417, 1974. (Translated from Ingenieur-Archiv, Bd. 25, Heft 1, 1957, pp 32-57.)

Eppler, R. (1990). *Airfoil Design and Data*. Springer-Verlag, Berlin.

Eppler, R. and Somers, D.M. (1980). "A Computer Program for the Design and Analysis of Low-Speed Airfoils." NASA TM 80210.

Fan H.-Y. and Lampinen J. (2002). "A trigonometric mutation approach to Differential Evolution." In: Giannakoglou K.C., Tsahalis D.T., Periaux J., Papailiou K.D., and Fogarty T. (eds.) (2002). *Evolutionary Methods for Design, Optimization and Control* (*Proceedings of the EUROGEN2001 Conference*, Athens, Greece, September 19-21, 2001). CIMNE, Barcelona (Spain), pp 65–70.

Fisher, M.L. (1981). "The Lagrangian relaxation method for solving integer programming problems." *Management Sci.* Vol 27, pp 1-18.

Forrest, S. and Perelson, A.S. (1991). "Genetic algorithms and the immune system." *Proceedings of the 1st International Conference on Parallel Problem Solving from Nature*. Springer-Verlag, Berlin (Lecture Notes in Computer Science).

Goldberg, D.E (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley.

Goldberg, D.E. and Deb, K. (1991). "A comparative analysis of selection schemes used in genetic algorithms." *Foundations of Genetic Algorithms*, ed. G.J.E. Rawlins, Morgan Kaufmann Press, San Mateo, pp 69-93.

Gostelow, J.P. (1964). "Potential flow through cascades: A comparison between exact and approximate solutions." A.R.C. CP.807.

Gostelow, J.P. (1984). *Cascade Aerodynamics.* Pergamon Press, Oxford.

Hajela, P. and Lee. J. (1996). "Constrained genetic search via schema adaptation: An immune network solution." *Structural Optimization*, Vol. 12, No. 1, pp 11-15.

Hajela, P. and Yoo, J. (1999). "Immune network modelling in design optimization." In: Corne, D., Dorigo, M., and Glover, F. (eds.), *New Ideas in Optimization*, McGraw-Hill, New York, pp 203-215.

Hajela, P., Yoo, J., and Lee, J. (1997). "GA-based simulation of immune networks – Applications in structural optimization." *Journal of Engineering Optimization.* Vol. 29, pp 131-149.

Hart, E., Ross, P., and Nelson, J. (1998). "Producing robust schedules via an artificial immune system." *Proceedings of the 1998 IEEE International Converence on Evolutionary Computation*, Alaska, IEEE Press.

Hendtlass, Tim (2001). "A combined Swarm Differential Evolution algorithm for optimization problems." *Lecture Notes in Computer Science*, no. 2070. Springer-Verlag, pp 11-18.

Hicks, R.M. and Henne, P.A. (1997). "Wing design by numerical optimization." *AIAA Paper* 77-1247.

Hightower, R., Forrest, S., and Perelson, A.S. (1995). "The evolution of emergent organization in immune system gene libraries." In: Eshelman, L.J. (ed.)

Proceedings of the 6^{th} International Conference on Genetic Algorithms. Morgan Kaufmann, pp 344-350.

Holland, J.H. (1962). "Outline for a logical theory of adaptive systems." *J. ACM* Vol. 3, pp 297-314.

Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor.

Hollstien, R.B. (1971). "Artificial Genetic Adaptation in Computer Control Systems". Ph.D. Thesis, University of Michigan.

Jacob, K. and Riegels, F.W. (1963). "The calculation of the pressure distributions over aerofoil sections of finite thickness with and without flaps and slats." *Z. Flugwiss*, Vol. 11, No. 9, pp 357-367.

Joines, J.A. and Houck, C.R. (1994). "On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GA's." *Proceedings of the 1^{st} IEEE Conference on Evolutionary Computation*. IEEE, Piscataway, NJ. pp 579-584.

Jones, R.T. (1990). *Wing Theory*. Princeton University Press.

Keller, H.B. (1975). "Some computational problems in boundary layer flows." *Proceedings of the 4^{th} International Conference on Numerical Methods in Fluid Dynamics, Lecture Notes in Physics #35*, Springer, pp 445-451.

Kellog, O.D. (1929). *Foundations of Potential Theory*. Frederick Ungar Publishing Company, New York.

Ladson, C.L., Brooks, C.W.Jr., Hill, A.S., and Sproles, D.W. (1996). "Computer program to obtain ordinates for NACA airfoils." NASA Technical Memorandum 4741. http://techreports.larc.nasa.gov/ltrs/ltrs.html.

Lampinen, J. (1999). "Differential Evolution – New naturally parallel approach for engineering design optimization." In: Topping, B.H.V. (ed.), *Developments in Computational Mechanics with High Performance Computing.* Civil-Comp Press, Edinburgh (Scotland), pp 217–228.

Lampinen, J. (2001). "A bibliography of Differential Evolution algorithm." Technical Report. Lappeenranta University of Technology, Department of Information Technology, Laboratory of Information Processing. Available via Internet http://www.lut.fi/~jlampine/debiblio.htm. Cited 02/07/2003.

Lewis, R.I. (1991). *Vortex Element Methods for Fluid Dynamic Analysis of Engineering Systems*. Cambridge University Press, Cambridge.

Liebeck, R.H. (1973). "A class of airfoils designed for high lift in incompressible flow." *Journal of Aircraft*, Vol. 10, No. 10, October, pp 610-617.

Liebeck, R.H. (1978). "Design of subsonic airfoils for high lift." *Journal of Aircraft*, Vol. 15, No. 9, September, pp 547-561.

Liebeck, R.H. (1990). "Computational Aerodynamics Applied to General Aviation/ Business Aircraft." Chapter 5 in Henne, P.A. (ed.), *Applied Computational Aerodynamics, Progress in Astronautics and Aeronautics*, Vol 125, American Institute of Aeronautics and Astronautics, Inc., Washington DC.

Liebeck, R.H. and Ormsbee, A.I. (1970). "Optimization of airfoils for maximum lift." *Journal of Aircraft*, Vol 7, No. 5, September, pp 409-415.

Liu, J. and Lampinen, J. (2002). "Adaptive parameter control of Differential Evolution." In: Matoušek, R. and Ošmera, P. (eds.) (2002). *Proceedings of MENDEL 2002, 8th International Mendel Conference on Soft Computing*, June 5–7 2002, Brno, Czech Republic. Brno University of Technology, Faculty of Mechanical Engineering, Brno (Czech Republic), pp 19–26.

Lombardi, G., Salvetti, M., and Pinelli, D. (2000). "Numerical Evaluation of Airfoil Friction Drag." *Journal of Aircraft: Engineering Notes*, Vol. 37, No. 2, pp 354-356.

Marmelstein, R.E., Van Veldhuizen, D.A., and Lamont, G.B. (1997). "A distributed architecture for an adaptive computer virus immune system." *Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE Press, pp 3838-3843.

Martensen, E. (1959). "Berechnung der druckverteilung an gitterprofilen in ebener potentialstromung mit einer fredholmschen integralgleichung." *Archive for Rational Mechanics and Analysis*, Vol. 3, pp 235-270.

Masters, T. and Land, W. (1997). "A new training algorithm for the general regression neural network." *IEEE International Conference on Systems, Man, and Cybernetics, Computational Cybernetics and Simulation.*, Volume 3, pp 1990–1994.

Michalewicz, Z. (1992). *Genetic Algorithms + Data Structures = Evolution Programs.* Springer, Berlin.

Michalewicz, Z. (1995a). "Genetic Algorithms, numerical optimization, and constraints." In: Eshelman, L.J. (ed.), *Proceedings of the 6th International Conference on Genetic Algorithms*. Morgan Kaufmann, pp 151-158.

Michalewicz, Z. (1995b). "A survey of constraint handling techniques in Evolutionary Computation methods." In: McDonnell, J.R., Reynolds, R.G., and Fogel, D.B. (eds.) *Proceedings of the 4th Annual Conference on Evolutionary Programming.* MIT Press, pp 135-155.

Michalewicz, Z. and Janikow, C. (1991). "Handling constraints in Genetic Algorithms." In: Belew, R.K. and Booker, L.B. (eds.), *Proceedings of the 4th International Conference on Genetic Algorithms*. Morgan Kaufmann, pp 151-157.

Nelder, J.A. and Mead, R. (1965). “A simplex method for function minimization.” *Computer Journal*, Vol. 7, pp 308-313.

Obayashi, S. and Tsukahara, T. (1996). “Comparison of optimization algorithms for aerodynamic shape design.” *14th Applied Aerodynamics Conference*, AIAA Paper 96-2394-CP, New Orleans.

Oyama, A., Obayashi, S., and Nakahashi, K. (1999). "Fractional factorial design of genetic coding for aerodynamic optimization." AIAA Paper 99-3298.

Perez, R.E., Behdinan, K., and Chung, J. (2000). "Airfoil shape optimization using Genetic Algorithms." *Proceedings of the Canadian Aeronautics and Space Institute Conference 2000, May, Ottawa.*

Pfeiffer, N.J. (1990). "Computational aerodynamics applied to general aviation/business aircraft." Chapter 5 in Henne, P.A. (ed.), *Applied Computational Aerodynamics,* *Progress in Astronautics and Aeronautics*, Vol 125, American Institute of Aeronautics and Astronautics, Inc., Washington DC.

Press, W.H., Teukolsky, S.A., Vetterling, W.T., and Flannery, B. P. (1992). *Numerical Recipes in C*. Cambridge University Press, Cambridge.

Price, K.V. (1999). "An introduction to Differential Evolution." In: Corne, D., Dorigo, M., and Glover, F. (eds.), *New Ideas in Optimization*, Mc-Graw-Hill, New York, pp 79-108.

Price, K.V. (2000). Private correspondence re: HDE.

Rae, A. and Parameswaran, S. (1998). "Application-specific heterogeneous multiprocessor synthesis using differential-evolution." *11th International Symposium on System Synthesis*, December 2-4, pp. 83–88.

Rechenberg I. (1973). *Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*. Frommann-Holzboog, Stuttgart.

Rechenberg, I. (1965). *Cybernetic Solution Path of an Experimental Problem*. Royal Aircraft Establishment Library Translation 1122.

Richardson, J.T., Palmer, M.R., Liepins, G., and Hilliar, M. (1989). "Some guidelines for Genetic Algorithms with penalty functions." In: Schaffer, J.D. (ed.), *Proceedings of the 3rd International Conference on Genetic Algorithms*. Morgan Kaufmann, pp 191-197.

Rogalsky, T. (1998). "Aerodynamic Shape Optimization of Fan Blades". M.Sc. Thesis. University of Manitoba. Department of Applied Mathematics.

Rogalsky, T. and Derksen, R.W. (2000). "Hybridization of Differential Evolution for aerodynamic design". *Proceedings of the 8th Annual Conference of the Computational Fluid Dynamics Society of Canada.* June 11-13, pp 729-736.

Rogalsky, T., Derksen, R.W., and Kocabiyik, S. (1999). "An aerodynamic design technique for optimizing fan blade spacing." *Proceedings of the 7th Annual Conference of the Computational Fluid Dynamics Society of Canada*, pp 2.29 – 2.34.

Rogalsky, T., Derksen, R.W., and Kocabiyik, S. (2000). "Differential Evolution in aerodynamic optimization." *Canadian Aeronautics and Space Journal*, Vol. 46, No. 4, pp 183-190.

Rudolph, G. (1997a). *Convergence Properties of Evolutionary Algorithms.* Kovac Press, Hamburg.

Rudoph, G. (1997b). "Modes of stochastic convergence." In: Back, T., Fogel, D.B., and Michalewicz, Z. (eds.), *Handbook of Evolutionary Computation.* IOP Publishing and Oxford University Press, Bristol and NewYork, pp. B2.3.1-3.

Rudoph, G. (1997c). "Local performance measures: Genetic Algorithms." In: Back, T., Fogel, D.B., and Michalewicz, Z. (eds.), *Handbook of Evolutionary Computation.* IOP Publishing and Oxford University Press, Bristol and NewYork, pp. B2.4:20-27.

Rüttgers, M. (1997). "Differential Evolution: A method for optimization of real scheduling problems." International Computer Science Institute, TR-97-013.

Sarpkaya, T. (1989). "Computational methods with vortices – The 1988 Freeman Scholar Lecture." *Journal of Fluids Engineering.* Vol. 111, March, pp 5-52.

Schwefel, H.P. (1975a). "Evolutionsstrategie und Numerische Optimierung." Dissertation, Technical University of Berlin.

Schwefel, H.P. (1975b). "Binäre optimierung durch somatische mutation." Technical Report, Technical University of Berlin and Mecial University of Hanover.

Selig, M.S., Guglielmo, J.J, Broeren, A.P., and Giguère, P. (1995*). Summary of Low-Speed Airfoil Data*, Vol. 1. SoarTech Publications, Virginia Beach.

Smith, A.E. and Tate, D.M. (1993). "Genetic optimization using a penalty function." In: Forrest, S. (ed.) *Proceedings of the 5th International Conference on Genetic Algorithms*. Morgan Kaufmann. pp 499-505.

Šmuc, T. (2002). "Improving convergence properties of the Differential Evolution algorithm." In: Matoušek, R. and Ošmera, P. (eds.) (2002). *Proceedings of MENDEL 2002, 8th International Mendel Conference on Soft Computing*, June 5–7 2002, Brno, Czech Republic. Brno University of Technology, Faculty of Mechanical Engineering, Brno (Czech Republic), pp 80-86.

Sobieczky, H. (1998). "Manual Aerodynamic Optimization of an Oblique Flying Wing." 36th Aerospace Sciences Meeting & Exhibit, Reno, NV. *AIAA Paper* 98-0598.

Sobieczky, H. (1999). "Parametric Airfoils and Wings". In: Fujii, K. and Dulikravich, G.S. (eds.), *Recent Development of Aerodynamic Design Methodologies*. Friedr. Vieweg & Sohn Verlagsgesellschaft mbH, Braunschweig/Wiesbaden. pp 71-87.

Storn, R. (1995). "Modeling and optimization of PET-redundancy assignment for MPEG-sequences." Technical Report TR-95-018, ICSI.

Storn, R. and Price, K. (1995). "Differential Evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces." Technical Report TR-95-012, ICSI, ftp.icsi.berkeley.edu.

Storn, R. and Price, K. (1997a). "Differential Evolution." *Dr. Dobb's Journal*, Vol. 22, No. 4, April, pp 18-24.

Storn, R. and Price, K. (1997b). "Differential Evolution – a simple and efficient heuristic for global optimization over continuous spaces." *Journal of Global Optimization*, Kluwer Academic Publishers, Vol. 11, No. 4, December, pp 341–359.

Stratford, B.S. (1959a). "The prediction of separation of the turbulent boundary layer." *Journal of Fluid Mechanics*, Vol. 5, pp 1-16.

Stratford, B.S. (1959b). "An experimental flow with zero skin friction throughout its region of pressure rise." *Journal of Fluid Mechanics*, Vol. 5, pp 17-35.

Takahashi, S., Obayashi, S., and Nakahashi, K. (1999). "Inverse design optimization of transonic wings based on multi-objective Genetic Algorithms." *AIAA Journal*, Vol. 37, No. 12, December, pp 1656-1662.

Theodorsen, T. and Garriock, I.E. (1933). "General potential theory of arbitrary wing sections." NACA TR 452.

Vanderbilt, D. and Louie, S.G. (1984). "A Monte Carlo simulated annealing approach to optimization over continuous variables." *Journal of Computational Physics*, Vol. 56, pp 259-271.

Vanderplaats, G.N., Hicks, R.N., and Murman, E.M. (1975). "Applications of numerical optimization technique to airfoil design." NASA Ames Research Center, NASA SP-347, Part II, March, pp 749-768.

Venkataraman, P. (1995a). "A new procedure for airfoil definition." *13th Applied Aerodynamics Conference, AIAA Paper* 95-1875-CP, San Diego.

Venkataraman, P. (1995b). "Optimum airfoil design in viscous flows." *American Institute of Aeronautics and Astronautics 13th Applied Aerodynamics Conference, AIAA Paper* 95-1876-CP, San Diego.

Venkataraman, P. (1996a). "Optimal airfoil design." *American Institute of Aeronautics and Astronautics 14th Applied Aerodynamics Conference, AIAA Paper* 96-2371-CP, New Orleans.

Venkataraman, P. (1996b). "Inverse airfoil design using design optimization." *American Institute of Aeronautics and Astronautics 14th Applied Aerodynamics Conference, AIAA Paper* 96-2503-CP, New Orleans.

Vicini, A. and Quagliarella, D. (1999). "Airfoil and wing design through hybrid optimization strategies." *AIAA Journal*, Vol. 37, No. 5, May, pp 634-641.

Wilkinson, D.H. (1967). "A numerical solution of the analysis and design problems for the flow past one or more aerofoils or cascades." *A.R.C.R&M,* No. 3545.

Wilkinson, D.H. (1969). "The analysis and design of blade shape for radial, mixed and axial turbomachines with incompressible flow." *M.E.L. Report No. W/M(3F), English Electric Co.* Whetstone, Leicester.

Wolpert, D. (2002). "William Dembski's treatment of the No Free Lunch theorems is written in jello." www.talkreason.org/articles/jello.cfm. Cited 02/07/03.

Wolpert, D. and Macready, W. (1997). "No Free Lunch Theorems for optimization." *IEEE Transactions on Evolutionary Computation.* Vol. 1, No.1, pp 67-82.

Wormington, M., Panaccione, C., Matney, K.M., Bowen, D.K. (1999). "Characterization of structures from X-ray scattering data using genetic algorithms." *The Royal Society, Philosophical Transactions: Mathematical, Physical and Engineering Sciences* 357(1761), October, London, UK, pp 2827–2848.

Appendix A FanOpt

FanOpt is the interactive window-based software package developed to perform aerodynamic optimization. All results reported in this thesis were obtained using FanOpt. Two versions of the software are included on the accompanying CD. In FanOpt v 3.3, immune acceleration occurs after every generation. In FanOpt v 3.5, immune acceleration occurs with the same frequency as that of hybridization. The HIADE b-b-50-100 10-10-10 results reported in Chapter 6 were obtained using v 3.5.

To use the software, most systems will require several .dll files. These are included on the CD in the folder "Appendix A FanOpt\dll_files". They should be copied to the "Windows\System" folder. Note that Windows Explorer sometimes does not show these "hidden files". You may have to change the view settings to "view all files"

To install the software, copy all files from the folder "Appendix A FanOpt\run_files" into any folder on your machine.

Documentation for FanOpt is included on the CD in the document "Appendix A FanOpt.htm".

Appendix B Airfoil Representation

In Chapter 3, two new airfoil parameterization methods were developed and compared to Bezier parameterization. The BP 3333 and BP 3434 parameterizations use a new set of parameters to define their component Bezier curves. 63 airfoils were chosen for representation by the three parameterizations. Links to the representations are provided on the accompanying CD in the document "Appendix B Airfoil Representation.htm".