8-4-2001

Quasi 3D Multi-stage Turbomachinery Pre-optimizer

Chad Eric Burdyshaw

QUASI 3D MULTI-STAGE TURBOMACHINERY PRE-OPTIMIZER

By

Chad Eric Burdyshaw

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Computational Engineering
in the College of Engineering

Mississippi State, Mississippi

August 2001

QUASI 3D MULTI-STAGE TURBOMACHINERY PRE-OPTIMIZER

By

Chad Eric Burdyshaw

Approved:

James C. Newman III Assistant Professor of Aerospace Engineering (Director of Thesis)	J. Mark Janus Associate Professor of Aerospace Engineering (Committee Member)
David Whitfield Professor of Aerospace Engineering (Committee Member)	Daniel Hyams Assistant Research Professor (Committee Member)
Bradley D. Carter Professor of Computer Science and Graduate Coordinator	A. Wayne Bennett Dean of the College of Engineering

Name : Chad Eric Burdyshaw

Date of Degree : August 4, 2001

Institution : Mississippi State University

Major Field : Computational Engineering

Major Professor: Dr. James C. Newman III

Title of Study : QUASI 3D MULTI-STAGE TURBOMACHINERY PRE-OPTIMIZER

Pages in Study : 95

Candidate for Degree of Master of Science

A pre-optimizer has been developed which modifies existing turbomachinery blades to create new geometries with improved selected aerodynamic coefficients calculated using a linear panel method. These blade rows can then be further refined using a Navier-Stokes method for evaluation. This pre-optimizer was developed in hopes of reducing the overall CPU time required for optimization when using only Navier-Stokes evaluations. The primary method chosen to effect this optimization is a parallel evolutionary algorithm. Variations of this method have been analyzed and compared for convergence and degree of improvement. Test cases involved both single and multiple row turbomachinery. For each case, both single and multiple criteria fitness evaluations were used.

ACKNOWLEDGMENTS

I would like to thank my advisors, Dr. Newman, Dr. Janus, Dr. Hyams, and Dr. Whitfield for their continued guidance and assistance. I would also like to thank Dr. Eric McFarland for his correspondance and insight. Furthermore, I would like to thank Ed Luke for making available the L_aT_eX template which greatly simplified the formatting of this document. And I would like to acknowledge Dr. David Fogel for donating his valuable time to travel to a small university to present his work and encourage others to investigate Evolutionary Computing. Furthermore I would like to express my gratitude to ONR and NASA Glenn with Dr. Pat Purtell and Dr. Eric McFarland as technical monitors respectively, for their generous support of this work.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

NOMENCLATURE

Identifiers:

b	stream sheet thickness
U	transformed wheel speed
ρ	fluid density
ϕ	tangential coordinate
m	meridional or axial coordinate
x, y	transformed coordinates
α	absolute flow angle with respect to meridional direction
P	local static pressure
P_{inlet}	inlet static pressure
r	surface of revolution radius
V	absolute flow velocity
$V_{\rho b}$	estimated nonlinear term
v	absolute disturbance velocity
M_{init}	reference Mach number
a_{init}	reference critical velocity (speed of sound)
γ	ratio of specific heats
Bs, Bd, Bv	integral method influence coefficients

σ, δ, λ	integral method singularity strengths
C_p	coefficient of pressure
C_x	axial thrust coefficient
C_y	tangential thrust coefficient
C_m	moment coefficient
C_l	coefficient of lift
C_d	coefficient of drag
F_x	total force along the x axis
F_y	total force along the y axis
ζ	loss coefficient
$C(t)$	Bezier curve as a function of parameter t
$B(t)$	Bernstein polynomial as function of parameter t
b_i	Bezier control points

Subscripts:

$c,\ onset$	uniform flow value
est	estimated value
$isen$	isentropic value
$init$	reference value
$inlet$	value at inlet
out	value at outlet
ϕ	tangential coordinate direction
m	meridional or axial coordinate direction
x, y	transformed coordinate directions
p	point in flowfield

CHAPTER I

INTRODUCTION

1.1 Motivation

Since the first recorded conception of fixed wing flight by George Cayley in 1799 [1], and the success of the Wright brothers "Flyer" in 1903 [1], engineers have steadily improved the concept of the airfoil to a high degree of efficiency. The experience and intuition built up over the last century endows current designers with the ability to create efficient airfoil designs for many different applications. Improvements are often difficult to achieve without many iterations of design and experimental analysis. In the 1960's and 70's, advances in computational technology and numerical techniques allowed airfoil analysis to be conducted in the computer rather than in the wind tunnel [2, 3, 4]. Since then, both computing ability and Computational Fluid Dynamics techniques have evolved considerably. With current technology it is now possible to perform fully three dimensional viscous and turbulent analyses on complex geometries. These very accurate simulations however, lie at the current limit of computational capability. The CPU cost for running these analyses can be immense. It is not uncommon for a single analysis to take several days to converge.

Due to the fact that the aerodynamic properties of an airfoil are not determined solely by the effects of viscosity and turbulence, it is reasonable to assume that improvements can be made by utilizing an inviscid analysis in design iterations. Furthermore, by neglecting some three dimensional effects, the problem can be reduced to only two dimensions. With these simplifications, the enormous CPU cost of an accurate analysis is reduced to a very inexpensive approximation.

The cost reduction achieved in analysis provides the ability to test many different designs in the time it would take to perform only one very accurate analysis. While traditionally, engineers have relied on experience to lead them to an optimal design, the use of various numerical optimization techniques can achieve optimal designs without the need to rely on intuition. Obtaining optima from this increased objective design space is the motivation for this study.

1.2 Survey of Current Literature

The work done in this thesis involves both recent and historical developments from the fields listed in the following subsections. This brief survey of techniques is given to expand the reader's awareness of the many possible areas for continuing research and application.

1.2.1 Design Optimization

The practice of inventing or improving existing designs or techniques to meet some predetermined goal is a primary tenet of human endeavor. Numerical design optimization provides an objective tool which is applicable

to almost any field imaginable. With proper problem formulation involving identification of the objective and any known constraints, there are a wide variety of numerical optimization techniques available [5].

The most widely used variety of numerical optimization techniques utilizes the idea of sensitivity derivatives. These derivatives indicate how to change the design variables in order to improve the current design. The use of sensitivity derivatives while not at an optima, can achieve an improvement in the objective function. A disadvantage to the sensitivity derivative method is the need to calculate the derivatives with respect to all design variables. In an aerodynamic design case, this would require computation of flow over the geometry for each of the design variables in order to complete a single iteration. This disadvantage has been recently addressed by calculating sensitivity derivatives using a complex Taylor's series expansion (CTSE) method [6]. Unlike other second order derivative techniques such as the central finite difference method, CTSE provides each derivative in a single iteration and its accuracy does not suffer from subtractive cancellation error. The sensitivity derivative method has been considered as a candidate for use in this study and has been investigated further in Section 3.5.1.

Another technique which has recently received much attention is the use of a directed zeroth order method known as an Evolutionary Algorithm (EA) [7, 8, 9, 10]. The Evolutionary Algorithm is based on the natural mechanisms proposed in Darwin's *The Origin of Species* [11]. It has a number of advantages over traditional gradient based methods. One such advantage is that, since derivatives are not required, it is not necessary to

run analyses equal to the number of design variables. Another advantage to the EA is it's ability to resist stalling in the local optima of a complex design space. The EA's major disadvantage is that it is often neccessary to run many iterations to reach a converged solution. Also, the EA is not guaranteed to provide an improvement in the objective function for each iteration. For both EA and gradient based methods, a reduction in overall time can be gained through the use of parallel processing. The Evolutionary Algorithm was selected as the primary method for use in this study. A more detailed description of the EA method can be found in Chapter III.

More recent studies have involved a combination of methods known as "Hybrids". These hybrid methods typically involve an Evolutionary Algorithm on the global scale and usually some sort of gradient based algorithm on the local scale [12, 13, 14]. Other schemes involve utilizing an EA in obtaining search directions within a gradient based optimization scheme. The former hybrid method benefits the overall optimization by retaining the wide design space provided through the EA and the guarantee of finding an actual optimum in the local domain, through the gradient based method. Furthermore, the gradient based local optimization can be used periodically to refine the quality of population members for continued evaluation through the EA. These hybrid methods generally outperform standard Genetic Algorithms in terms of both time and degree of improvement [15]. A hybrid algorithm has been implemented in this study and an analysis of its performance is presented in Section 3.5.2.

1.2.2 Curve Approximation

Generally when given a set of data, it is necessary to organize the data into a form more suitable for the task at hand. In this study, the relevant data is a set of points which describe the surfaces of a turbomachine. For analysis it is necessary to reconstruct the surfaces from the points given, and the ability to alter these surfaces is needed for the optimization process. A surface or curve approximation method may be used to achieve these goals. Given a set of points, there are a number of ways to describe the curve or surface they represent.

The most obvious way to construct a surface is to simply connect the points with lines to create the surface. The disadvantage of this method becomes clear when attempting to manipulate the geometry. Perturbation of a single surface point would result in a very localized surface change which would manifest as a sharp protrusion or depression on the surface. Fluid flow is very sensitive to sharp changes in geometry and would therefore be drastically effected by such a surface perturbation. The approximation technique chosen therefore, must be able to perturb the geometry in such a way that the surface remains continuous and smooth. To accomplish this, a reduction in the amount of local control over the surface is needed.

To retain a smooth continuous curve, the most basic method would be to use a polynomial curve approximation. This would work for a small number of curve fit points but, as the number of points increase, the polynomial increases in degree and eventually becomes oscillatory. For the number of points needed to describe an airfoil, polynomial approximation is not appropriate. One solution to this would be to use a combination of low order

(cubic) polynomial curves attached at their endpoints in a spline [16]. The cubic spline however, cannot represent functions with more than one value at each point (i.e. parametric curves). Another disadvantage to the cubic spline is that to remain smooth, it requires knowledge of the derivatives at the endpoints of each piecewise cubic polynomial. Every subsequent perturbation of the geometry would result in the need to recalculate these derivatives. The flow solver used in this study employs a cubic spline for surface representation. This does not restrict the optimizer from using parametric curve representations, but it does require the airfoil geometry to be divisible into two non-parametric curves.

A simpler curve approximation can be obtained by using a Bezier curve. This parametric representation has the property of pseudo-local control and retains the continuity and smoothness needed for an airfoil geometry. Furthermore, the surface representation is reduced to the number of control points selected to fit the original curve. An increase in the degree of the Bezier curve will not result in oscillatory behavior and, because this is not a spline, the derivatives at endpoints are not needed. A disadvantage to Bezier curves is that they only interpolate at the endpoints. A least squares or other best fit algorithm must be used to find an acceptable approximation. The Bezier curve was selected for curve approximation in this study. Further information on this technique can be found in Chapter III.

A possible improvement to the chosen Bezier curve approximation method may be found in the use of Non-Uniform Rational Bezier Splines (NURBS) [17]. This method, though more complex than a simple Bezier curve, would provide more local control in the form of rational Bezier

curve splines which interpolate the original curve at the spline connections. Perturbation of surface points would result in changing only the piecewise curve in that local area, and the rational Bezier curve weighting factor could be manipulated to fine tune the geometry. The parametric spline formulation would also allow the representation of an entire airfoil using a single spline rather than with two Bezier curves.

1.2.3 Fluid Flow Solvers

The goal of this optimization study is to be able to produce turbomachinery designs that possess improved aerodynamic properties over the original design. The analysis is purely computational. Therefore, it is necessary to model the physical system as accurately as possible while keeping the CPU cost at a minimum. Some current methods, ranging from the most accurate to the least, have been briefly presented in this section.

The affects of body, pressure, and viscous forces on a fluid element can be completely described in the form of the Navier-Stokes equations [2, 18, 19]. These equations, developed by Navier in 1831 and in a more rigorous form by Stokes in 1845, combine the equation of state and the conservation laws of mass, momentum, and energy acting upon a discrete fluid element. It would seem that with the use of these equations, any type of fluid flow problem may be solved. This is true in theory however, the enormous number of computations required to resolve a fluid flow problem in this manner is prohibitive. Additionally, the discretization and numerical errors accumulated during the calculation can return a solution that is far from accurate.

Calculation of the turbulent, viscous, compressible, Navier-Stokes equations at a single point alone involves the solution of a set of nonlinear equations. To determine the turbulent flow about a simple two dimensional object requires a grid of points fine enough to resolve the turbulent effects at the surface of the body and in the surrounding area. The calculations must be made at each point and for a number of iterations to reach a converged solution. Approximate techniques such as Large Eddy Simulation (LES) and various Reynolds Averaged Navier-Stokes (RANS) turbulence models can greatly reduce the computations necessary by reducing the need for high grid resoluton.

If viscosity is neglected, the Navier-Stokes equations are simplified and become the Euler equations. By dropping the viscosity terms from the Navier-Stokes equations, the overall number of computations is reduced. Also, due to the fact that fine grid resolution is not necessary to resolve viscous effects, a coarser grid may be used thereby further reducing the number of computations necessary for each iteration.

Further simplification of the flow to include an irrotationality condition reduces the problem to a Laplace equation. One method of solving the Laplace equation is to use a panel method. Panel methods use information from the source, doublet or vortex strengths at the boundary to calculate the pressure distribution on the boundary. These problems require the solution of a large system of algebraic equations, but the domain is reduced to consider only the points at the surface of the body. Furthermore, the equations can be linearized through approximations to produce a set of linear equations which can then be solved using linear algebra techniques. Of the

methods presented here, the linear panel method is by far the least CPU intensive. A concise review of panel methods for aerodynamic applications may be found in Reference [20].

The objective of this study is primarily focused on multiple blade row turbomachinery design. This requires the use of an evaluation method that can handle multiple row, blade to blade flows. In order to limit the scope of this study, the use of existing code is desirable. A quasi 3-D, multistage, blade to blade flow solver utilizing the linear panel method has been chosen for evaluation in this study. The flow solver was developed by Eric R. McFarland [21] and is detailed in Chapter II.

1.3 Objectives

As previously stated, optimization of turbomachinery using Navier-Stokes methods for analysis is a computationally expensive task. Typically, the time it takes to perform a Navier-Stokes analysis of an airfoil is orders of magnitude greater than the same calculation performed by a linear panel method. Although the linear panel method does not fully resolve the complicated physics encountered in turbomachinery, its speed allows the analysis of many different geometries in the time it takes to run a single geometry using the Navier-Stokes equations. Therefore, a multistage turbomachinery pre-optimizer has been developed utilizing a linear panel method. This approach allows near optimal designs to be produced very quickly. Design performance verification and further refinement may be conducted with more accurate Navier-Stokes simulations. This approach can be used to significantly reduce the overall design time.

The amount of design time reduction that can be achieved is a function of the optimization technique used and how well that technique has been tailored to a specific set of problems. For this purpose an evolutionary optimization technique has been chosen which employs a form of Genetic Algorithm (GA). The GA method was selected for exploratory investigation. However, its relative indifference to the number of design variables is an advantage in this case. A disadvantage is that GA's and other probabilistic approaches require many objective function evaluations. This problem has been ameliorated by the use of parallel processing, for which GA's are inherently well suited. In Chapter III the formulation and analysis of various configurations of the evolutionary optimization method used in this study have been detailed.

It is the purpose of this study to investigate whether a limited model such as the linear panel method can provide sufficiently accurate information that will lead to an actual improved design. To achieve this aim, two separate cases have been investigated.

Test case 1 is a two row turbine involving one stator and one rotor. Comparisons of the original versus optimized geometries have been made. Test case 2 involves a single turbine guide vane cascade. This case has been optimized using the linear panel method and comparisons have been made to an Euler analysis [22]. Concerns have been raised regarding the Bezier approximation of the geometry, constraints on the design variables (feasibility of the design), proper formulation of the fitness function, and most importantly, the accuracy of the linear panel method.

CHAPTER II

LINEAR FLOW SOLVER

The flow solver code used for optimization analysis was PCStage. This code and solution method were developed by Eric R. McFarland [21, 23, 24]. The method used by PCStage calculates multistage, blade to blade flows. It solves linearized equations for steady, compressible, inviscid, irrotational flow with some loss approximations. The quasi 3D effects are described by the inclusion of stream sheet thickness variation, radius change, and blade row rotation.

The advantage of McFarland's method in solving turbomachinery problems is that it calculates flow based on an absolute reference frame. Historically, flows involving multiple blade rows were calculated separately in sections where the flow is relative to each blade row [25, 26]. This simplification introduces errors into the system by neglecting the influence between blade rows. A revision of the governing equations formulated by Parker [27] has been reformulated by McFarland to utilize an absolute reference frame and remove the neccessity for flowfield decomposition. McFarland has also extended the method to consider subsonic compressible flows and radial flow machinery.

This solution method, known as the panel method, is an approximation to the solution of the governing equations consisting of the continuity equation (equation 2.1) and the irrotationality condition (equation 2.2) listed below. These equations describe flow on a blade to blade surface revolution with respect to an absolute reference frame.

2.1 Governing Equations

$$\frac{\partial}{\partial \phi}(\rho b V_\phi) + \frac{\partial}{\partial m}(r \rho b V_m) = 0 \tag{2.1}$$

$$\frac{\partial}{\partial \phi}(V_m) - \frac{\partial}{\partial m}(r V_\phi) = 0 \tag{2.2}$$

A transformation is introduced using the following variable substitutions.

$$y = \phi \tag{2.3}$$

$$x = \int \frac{m \; dm}{r} \tag{2.4}$$

$$V_x = r \rho b V_m \tag{2.5}$$

$$V_y = r \rho b V_\phi \tag{2.6}$$

These transformations are substituted into the governing equations to obtain equation 2.7 and equation 2.8.

$$\frac{\partial}{\partial y}\left(\frac{V_y}{r}\right) + \frac{1}{r}\frac{\partial}{\partial x}(V_x) = 0 \tag{2.7}$$

$$\frac{\partial}{\partial y}\left(\frac{V_x}{r\rho b}\right) - \frac{1}{r}\frac{\partial}{\partial x}\left(\frac{V_y}{\phi b}\right) = 0 \tag{2.8}$$

These transformed governing equations can be simplified to form equations 2.9 and 2.10.

$$\frac{\partial}{\partial y}(V_y) + \frac{\partial}{\partial x}(V_x) = 0 \tag{2.9}$$

$$\frac{\partial}{\partial y}(V_x) - \frac{\partial}{\partial x}(V_y) = -\frac{V_y}{\rho b}\frac{\partial}{\partial x}(\rho b) \tag{2.10}$$

The governing equations are linearized by assuming that the fluid density only varies in the meridional direction. An estimation is made for the non-linear irrotationality equation 2.10 to give the linear form of equation 2.11.

$$\frac{\partial}{\partial y}(V_x) - \frac{\partial}{\partial x}(V_y) = -\frac{V_{y_{est}}}{(\rho b)_{est}}\frac{d}{dx}(\rho b)_{est} \tag{2.11}$$

2.2 Solution Methodology

The linear governing equations 2.9 and 2.11 can be solved by a superposition of solutions. The solution involves the fluid velocities V_x, and V_y which are composed of a uniform flow V_{x_c}, and V_{y_c} plus a disturbance flow v_x, and v_y as shown in equation 2.12.

$$\begin{aligned} V_x &= V_{x_c} + v_x \\ V_y &= V_{y_c} + v_y \end{aligned} \tag{2.12}$$

Substitution of these equations into the governing equations gives the following linear equations. Equation 2.16 is the estimated non-linear term which is strictly a function of x.

$$V_{x_c} = V_{onset}\cos(\alpha_c) \tag{2.13}$$

$$V_{y_c} = V_{onset}\sin(\alpha_c) + V_{\rho b} \tag{2.14}$$

$$\begin{aligned} \frac{\partial}{\partial x}v_x + \frac{\partial}{\partial y}v_y = 0 \\ \frac{\partial}{\partial x}v_y - \frac{\partial}{\partial y}v_x = 0 \end{aligned} \tag{2.15}$$

$$V_{\rho b} = \int^{(\rho b)_{est}} \left(\frac{V_y}{\rho b}\right)_{est} d(\rho b)_{est} \tag{2.16}$$

Note, V_{onset} is the uniform flow value of velocity, and α_c is uniform flow value of the absolute flow angle with respect to the meridional direction.

There are three boundary conditions used in determining the flow field. The flow relative to a body must remain tangent to a body's surface (i.e. relative velocity normal to each panel is zero). The flow is uniform upstream and downstream of a body. And, the circulation for each body is set by the Kutta condition. For further details on these boundary conditions, refer to the literature [21].

The flow estimates can be calculated using a throughflow calculation as the estimate or by making a one dimensional calculation of the blade to blade flow. PCStage uses a one dimensional calculation to give the estimated nonlinear term $V_{\rho b}$. To solve the linear equations, the source strength is assigned a value equal to the normal component of the uniform flow along

the body's surface. The normal velocity boundary condition (equation 2.17) is applied to each panel along the surface. The Kutta condition is applied to the trailing edge of the body, and the resulting system of equations is solved using an LU matrix decomposition. To satisfy the upstream and downstream boundary conditions simultaneously, an iterative process on the flow angle of the uniform flow (α_c) is performed. This angle is adjusted until the calculated upstream flow angle matches the specified upstream flow angle.

$$\begin{aligned} &-V_{onset}\sin(\theta_p - \alpha_c) + (V_{\rho b_p} - U_p)\cos(\theta_p) \\ &+ \sum_{i=1}^{npt-1} Bs_{pi}\sigma_i + \sum_{j=1}^{npt} Bv_{pj}\lambda_j + \sum_{k=1}^{m} Bd_{pk}\delta_k = 0 \end{aligned} \tag{2.17}$$

In equation 2.17, the terms without summations are the uniform parts of the solution. The summation terms are the disturbance flow parts of the solution. The parameters Bs, Bv, and Bd are the influence coefficients used in the integral equations. The variables σ, λ, and δ are the source, vortex, and jump singularity distributions calculated in the integral equation solution. The variable θ is the angle of the surface with respect to the meridional coordinate and U is the wheel speed. The subscript p denotes a point in the flowfield.

2.3 Loss Approximations and Limitations

Losses in the system are approximated by the introduction of specified input parameters. Adiabatic efficiency is used to control losses in mechanical

work. Total pressure loss and viscous blockage are used to describe viscous loss. The Kutta condition (i.e. velocity must be continuous at trailing edge) for the trailing edge flow angle is used to control flow turning and for modeling deviation and slip flows. This loss model allows some control over a solution containing small viscous effects however, the accuracy is dependent on prior insight into the flow being modeled.

This method is only accurate when considering subsonic flows with very small viscous effects and small flow separations. For a more detailed description of the PCStage solution method, the reader is directed to references [21, 23, 24].

CHAPTER III

OPTIMIZATION METHOD

Clearly the objectives of any optimization method are to provide improved feasible solutions. These solutions should also be obtained quickly and with the best possible result. The means by which these issues are addressed will be discussed below and in the following sections.

The flow solver PCStage was chosen for its ability to run flow solutions quickly. This allows relatively fast aquisition of objective function evaluations. A curve approximation technique, designed to reduce the number of design variables, has been implemented. Although the Genetic Algorithm is not significantly hindered by large numbers of design variables, a reduction in the number of these design variables is a benefit to any optimization scheme. The chosen method benefits more from the form of the curve approximation used in the reduction which is explained in Section 3.1. Genetic Algorithm functions described in Section 3.2 such as Parent Selection, Mating, and Mutation have been analyzed in order to find the best convergence rates and fitness improvements, and parallelization (Section 3.3) has been incorporated to drastically reduce evaluation time. Finally, in Section 3.5 a comparison has been made to a gradient based method, and an EA/Gradient based hybrid has been proposed.

3.1 Design Surface Parameterization

The optimization of airfoil geometries can involve many design variables. For shape design, the variables may be selected as the coordinates of the surface points. This choice may result in an excessive number of design variables. This number can be reduced with the use of curve approximation techniques. As discussed in Chapter I, the approximation technique chosen for this study is the Bezier curve. By representing the surfaces in this manner, the number of design variables can be reduced to the number of control points necessary to describe the geometry.

The Bezier curve representation has been chosen due to its many useful properties. The properties of most interest are end point interpolation and pseudo-local control. End point interpolation requires that the endpoints of the Bezier curve be identical to the end points of the original curve. This property allows the precise definition of these critical points. Pseudo-local control is useful for manipulating the geometry by small degrees without adversely affecting the rest of the surface. Pseudo-local control is handled in the following manner: Given a Bezier curve (see equation 3.1) with a number of control points $b_0, \ldots, b_m$, if some control point b_i is moved, then the curve is most affected about the points whose parameter is close to i/n, where n is the degree of the Bernstein polymomial. This gives the ability to alter a localized region of the surface by moving a single point. Some surface control is lost in this approximation, but the resulting curve is always continous and generally smooth.

The approximation is made by taking the surface points of the original geometry, and specifying the axial positions of the control points to be

calculated. The arclengths between the specified control points are obtained and used to calculate the tangential positions of the control points to construct a best fit Bezier curve. An example of an airfoil with its corresponding Bezier control point representation is shown in Figure 3.1.

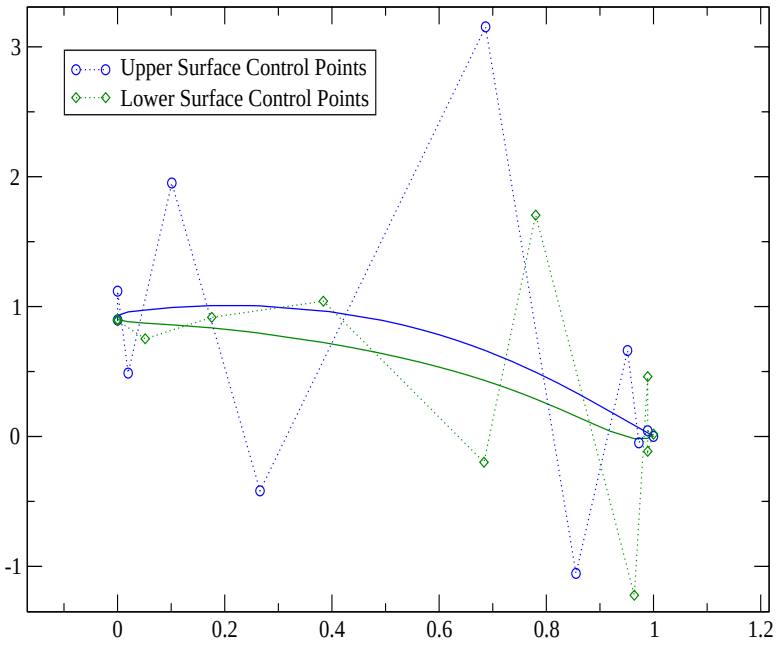

Figure 3.1: Approximated Geometry with Bezier Points Overlaid

A Bezier curve or polynomial of degree n is defined as:

$$C(t) = \sum_{i=0}^{n} B_i^n(t) b_i \tag{3.1}$$

where $b_i \epsilon \mathbf{R^2}$ are the control points and $B_i^n(t) = \left(\frac{n!}{i!(n-i)!}\right)(1-t)^{n-i}t^i$ $i = 0, 1, 2, \ldots, n$ are the Bernstein polynomial basis functions.

One disadvantage to using a Bezier curve approximation is that some information, particularly fine details, can be lost. This is due to the non-interpolative property of Bezier curves. As mentioned in chapter 1, these approximation errors can be diminished by increasing the degree of the Bezier curve or by utilizing piecewise Bezier curves (BSplines) on either side of the slope discontinuity. An example of feature loss is presented in Figure 3.2. For more information on BSplines, Bezier curves, and Bernstien polynomials see references [17, 28]

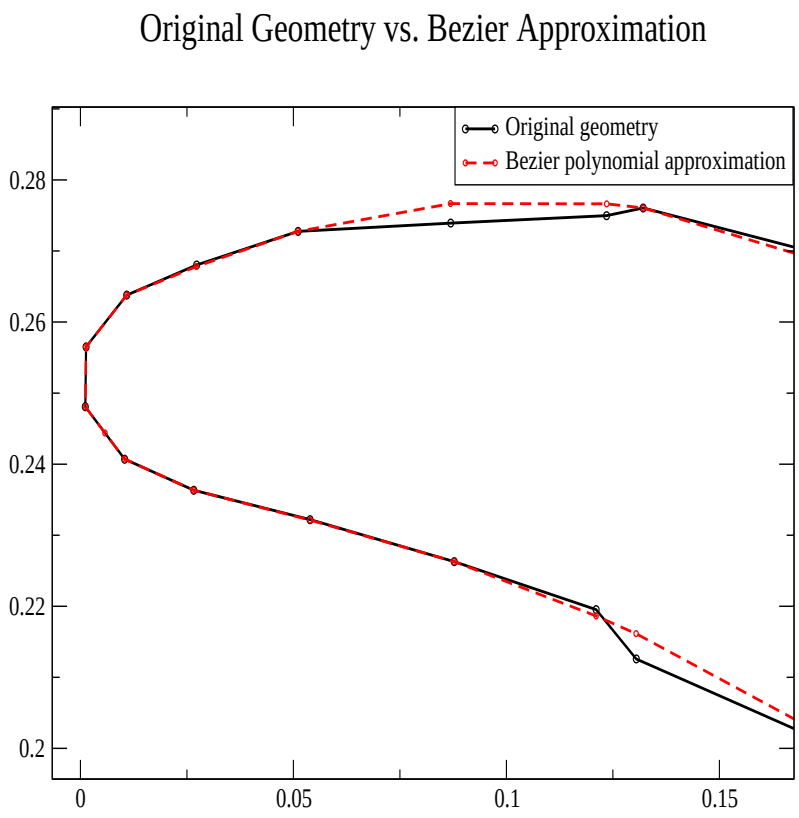

Figure 3.2: Feature Loss, Leading Edge Zoom

3.2 Evolutionary Optimization

Evolutionary optimization is a technique based on the processes of Natural Selection, otherwise known as "survival of the fittest" which was first clearly described in Darwin's *The Origin of Species* [11]. The selected optimization method employs a Real Valued Genetic Algorithm (RVGA) to encode and generate improved solutions based on a set of current solutions. These solutions "evolve" over a number of generations to improve their fitness values, thereby adapting to survive the fitness environment. For a more detailed discussion of GA's and their properties the reader is directed to the literature [29, 30, 31].

In this study each turbomachine geometry is represented as a member of a population. Each of these members is composed of the real values of the Bezier control point coordinates that describe its geometry. This representation defines the method as an RVGA, rather than a standard GA in which members are represented as a binary bit string. Every member of this population is evaluated by the flow solver and is assigned a fitness value based on the results. The fitness of a member influences how often it will be combined with other members to produce offspring and whether it will be replaced by a new offspring member. Random mutations are introduced to add new characteristics into the population. The application of the evaluation, selection, combination, and mutation operations result in the creation of a new generation. The process is repeated until a sufficient optimum has been achieved or a maximum number of generations has been reached. A flow chart of the evolutionary process is shown in Figure 3.3.

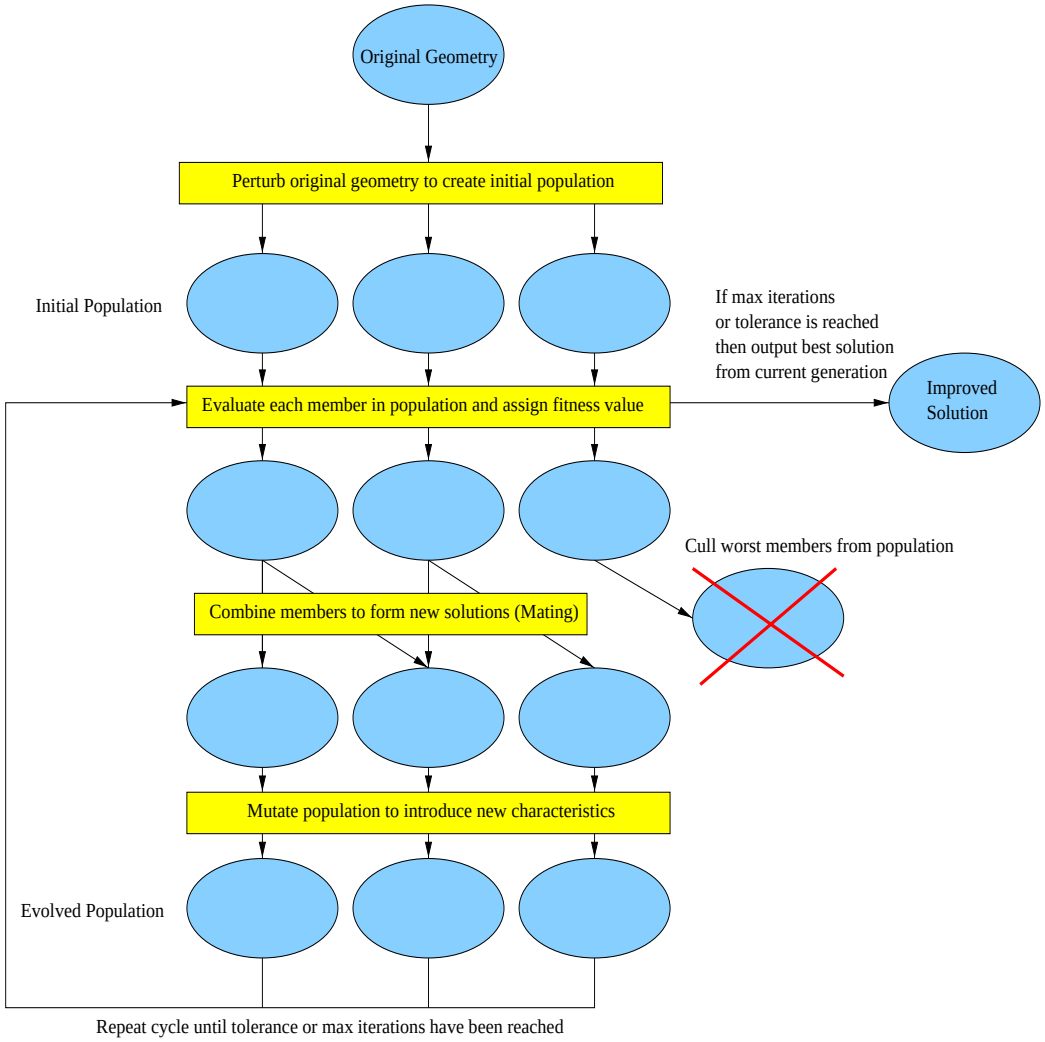

Figure 3.3: Evolutionary Algorithm Flow Chart

3.2.1 Fitness Criteria

The fitness criteria used in this study were based on the calculation of five coefficients by PCStage from the integration of the surface pressure around the body. These are the axial (C_x) and tangential (C_y) force coefficients, the lift coefficient (C_l), the drag coefficient (C_d), and the moment coefficient (C_m). A combination of one or all of these coefficients depending on the problem formulation, defines the fitness function. For example, the fitness function for an airfoil might be to maximize the ratio of $\frac{C_l}{C_d}$. Another fitness function useful in turbomachinery is to maximize axial and tangential force coefficients while reducing drag. An illustration of the coordinate system in which these coefficients are defined can be seen in Figure 3.5.

It should be noted that when the formulation of a fitness function is comprised of two or more properties, the property most sensitive to change will become the controlling factor in parent selection.

3.2.2 Generating New Solutions

Optimization using a Genetic Algorithm involves conflicting processes. To attain convergence, the solutions in the next generation must contain the best properties of the current generation. This requires selection and combination of only the best solutions. However, in order to avoid stalling in local optima, a wider variety of solutions must be included in subsequent generations, thereby hindering convergence. The balance of these conflicting processes is an active subject of study [32, 33].

The generation of new solutions is driven primarily by the fitnesses of the current solutions. This generation process is commonly referred to in

GA terms as Mating. Mating involves first selecting mating pairs, then combining these pairs to form new solutions, and finally introducing these new solutions into the population. There are a variety of ways to handle each of these steps.

Some of the issues involved in parent selection and combination are the following: How will the parents be selected? Will these parents mate only once or multiple times? How many offspring will each of the pairs produce per mating? These issues all have an effect on the balance of convergence and local stalling.

Some of the more common methods of parent selection are:

1. Choosing both parents strictly based on their fitness.
2. Choosing both parents randomly.
3. Choosing both parents randomly but weighted by their fitness (tournament selection).
4. Gather a group of members into a mating pool based on their fitness and select randomly from this pool.

Method 1 is the best way to facilitate convergence. It will also most likely stall very early into a local optimum. Method 2 will have a poor convergence rate, but it reduces the chance of local stalling. Methods 3 and 4 are attempts to strike a balance between the two goals.

The method chosen is a variation of method 3. Initially, the mating pool includes all members of the population. For each mating within the cycle, the best two members are chosen to mate. Once this mating has occurred,

there is a chance that one of these two parents will be removed from the mating pool. The chance of removal is statically determined by the user. For this study, the chance of removal was set at 40%. The next mating will again take the top two parents, disregarding those that have been removed from the pool. This method guarantees at least one elitist mating. It also reduces the possibility of converging to a local optimum by allowing a wider variety of parents to mate, thereby producing a wider variety of designs. When a mating occurs, the single offspring replaces the current worst member. A subsequent mating will replace the next worst member. This guarantees a constant population size and culling of the current worst designs. The offspring members, however, are still untested and may be inferior to the ones that they replace.

3.2.3 Mating Methods

For each mating, a new design must be generated by some combination of the parent geometries. Again, there are a myriad of ways to combine parents. The standard GA solution is to use some sort of crossover method [7]. Four different mating methods have been analyzed in order to determine their respective performances.

- Fixed single point crossover

  Each surface of the new solution is split at the middle control point. Half of the information comes from parent 1, the other half from parent 2. Which half each parent contributes is determined randomly. Each child is guaranteed to be composed of half of the information of each parent but not necessarily the best half.

- Variable single point crossover

  Like the fixed crosspoint method, each surface of the new solution is split and information from each of the parents is distributed on either side of the split or "crosspoint". However, in this case, the crosspoint is determined randomly and ranges from the first to the last control point. Therefore, it is possible for one parent to contribute more information to its child than the other parent.

- Surface Exchange

  In this scheme, for each body in the geometry, there is a 50% chance that the upper surface of that body will be chosen from either parent. The lower surface will be taken from the parent that was not chosen. Again, each child is guaranteed to be composed of half of the information of each parent.

- Point Average

  This is the simplest method of parent combination. The Bezier control points of each parent are simply averaged to create the child. While this method guarantees that all of the information from each parent is involved in child creation, it does not pass on the exact characteristics from each parent that make those parents the most fit.

Trials were run using all four mating methods and a plot of Fitness vs. Generation (Figure 3.4) was made to track convergence rates and degree of improvement. Though these plots will vary with each initial random seed, the average trend remains the same. The trials in this section were all run

with a mutation magnitude of 0.3 and mutation rate of 0.03 (see section 3.2.4), and all had the same random seed. Trials were also run at much higher rates and magnitudes, but due to the lack of convergence, the effect of mating methods was indeterminate. From the plots it is evident that all of the methods perform similarly, with a slightly higher fitness being achieved by the crossover methods. The optimizer code allows selection between mating methods, but the fixed single point method seems to consistently outperform the others though by a small degree only.

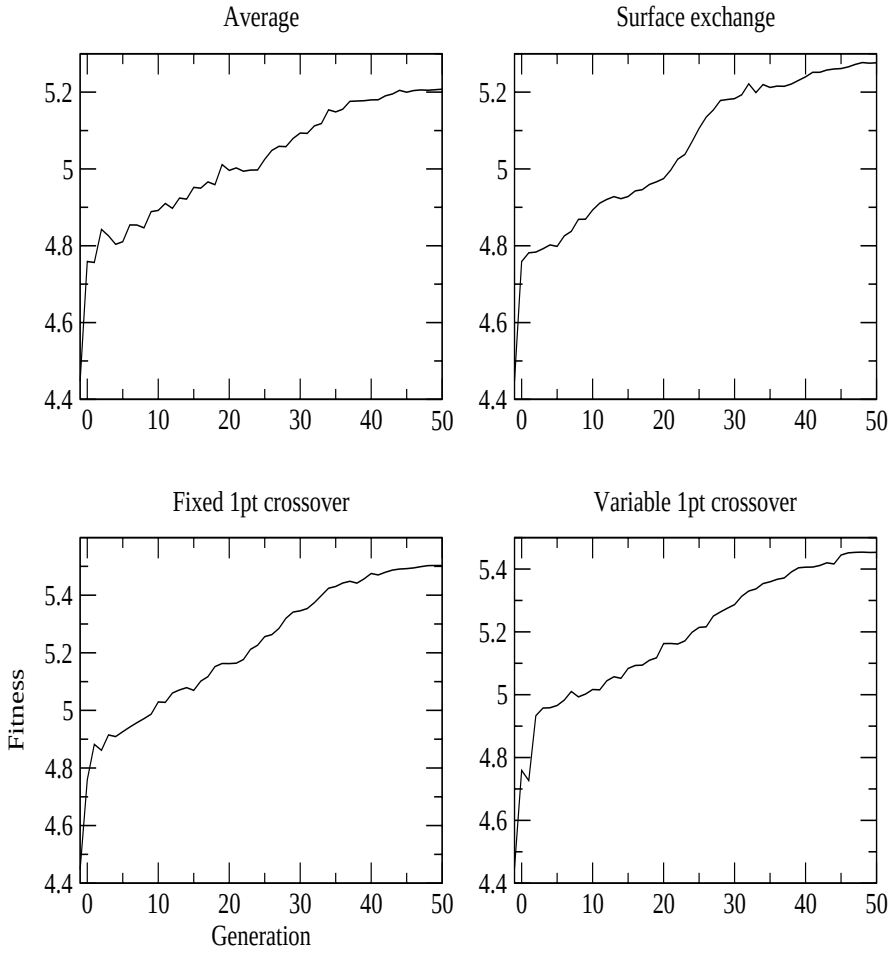

Figure 3.4: Mating Method Comparisons

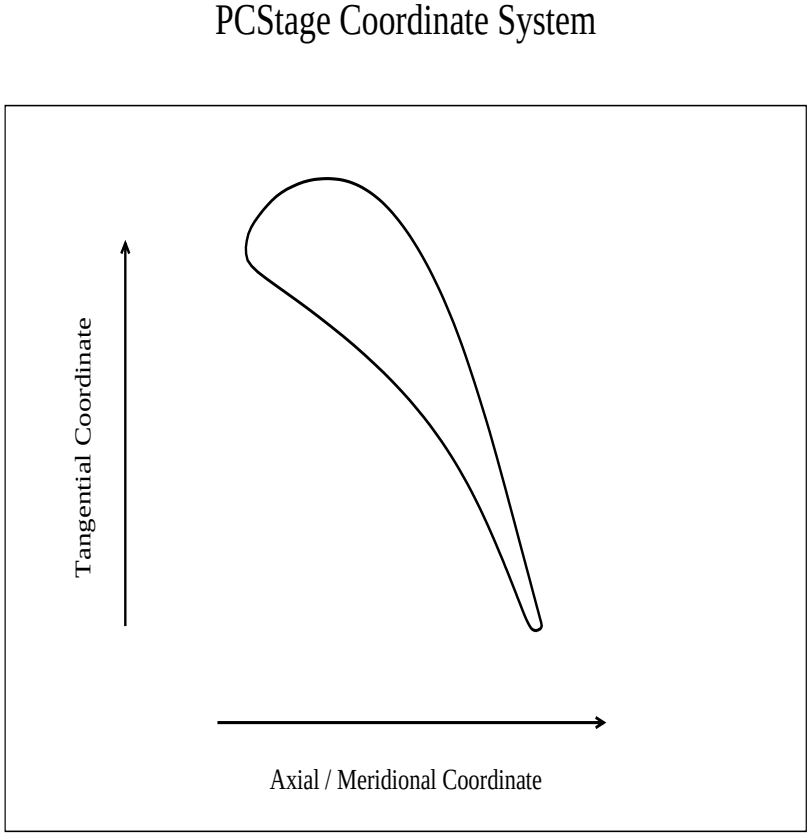

Figure 3.5: Coordinate system used in flow solver

3.2.4 Mutation

Mutation is the introduction of random perturbations to the geometry. In the strict sense of a GA, mutation is a flip of a random bit within the binary coding of the population member. For this RVGA, the process of mutation takes the form of a "perturbation" operator, but the term "mutation" will be used to signify a similar process. The mutation operator includes a mutation rate and magnitude. The rate determines the maximum percentage of control points which will undergo mutation. For the cases presented in this section, the mutation magnitude determines the maximum degree of perturbation along the tangential direction defined by a percentage of the tangential component of the control point location. In subsequent cases the magnitude is defined as the maximum degree of perturbation based on a percentage of the distance along the surface normal from the control point to the surface.

The purpose of mutation is to introduce new characteristics into a population and thereby reduce the possibility of converging to a local optimum. A drawback to mutation is that, depending on the degree, it can impede convergence. A solution to this problem is to make the mutation magnitude a function of the generation number [9]. In the beginning of the optimization process, a variety of solutions is desireable, so the mutation magnitude is kept relatively large. As the last generation is approached the solution should be converging to a global optimum, so the magnitude needs to be close to zero. Constant, linearly, and quadratically decreasing functions have been used to control the mutation magnitude and the results have been presented in Figures 3.6, 3.7, and 3.8. The mutation rate has also

been varied while holding the magnitude constant. The results of varied mutation rates are presented in Figure 3.9. For the purposes of creating a diverse initial population, the original geometry is subjected to a series of mutations in which a distinct initial mutation rate and magnitude are used. A subsequent (usually much smaller) rate and magnitude are used after the initial population has been created.

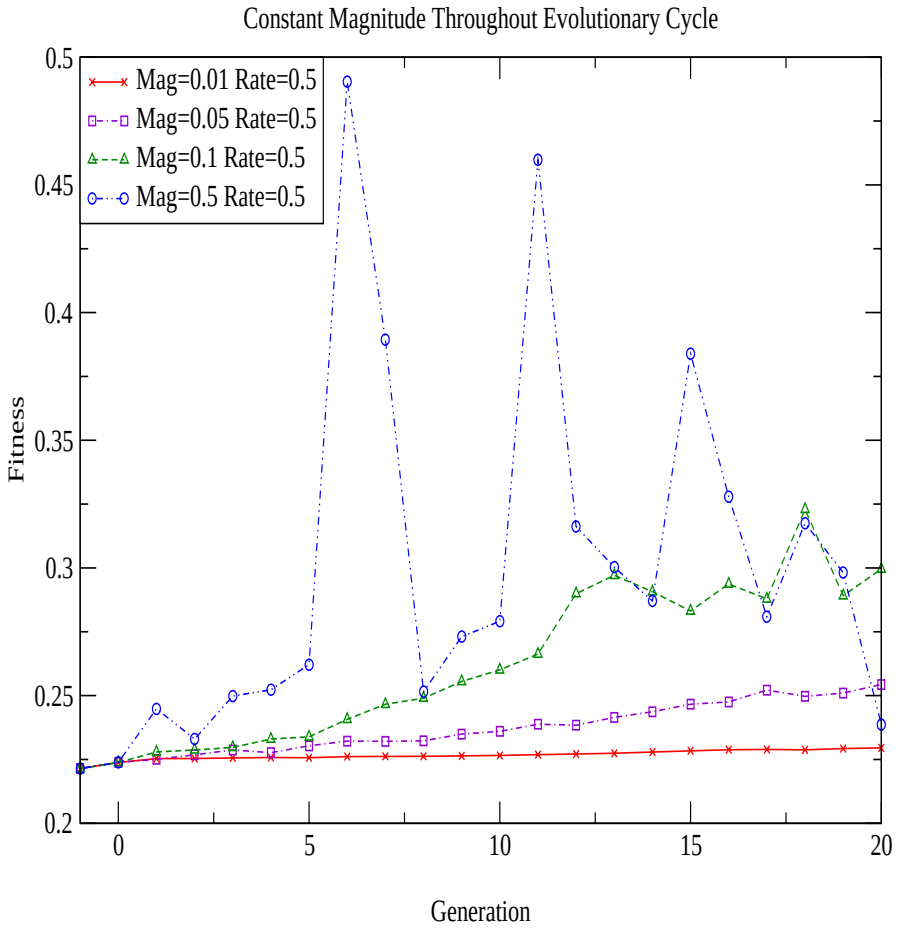

Figure 3.6: Comparison of Constant Mutation Magnitudes

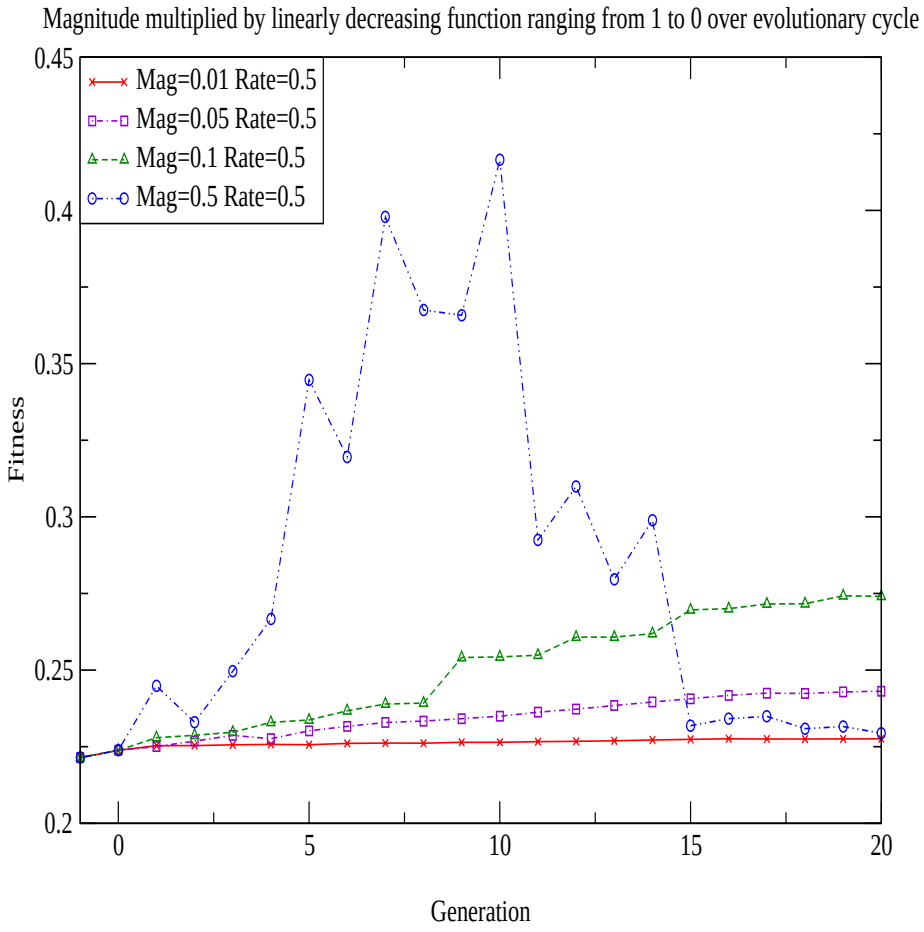

Figure 3.7: Comparison of Linearly Decreased Mutation Magnitudes

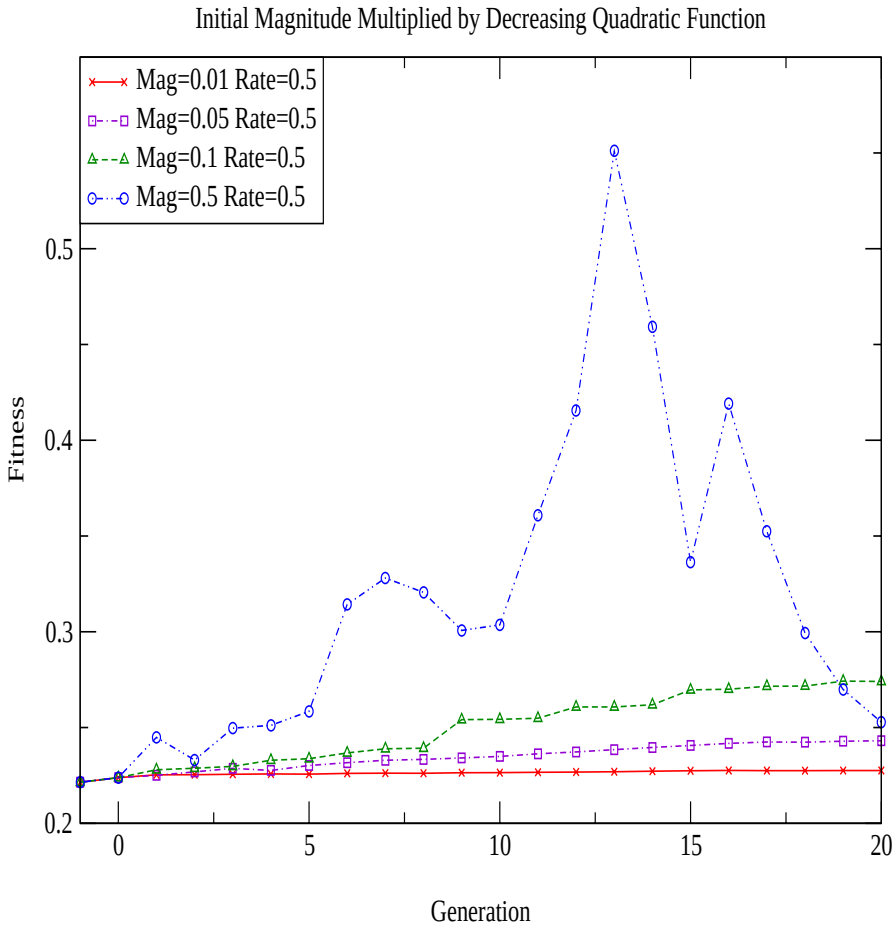

Figure 3.8: Comparison of Quadratically Decreased Mutation Magnitudes

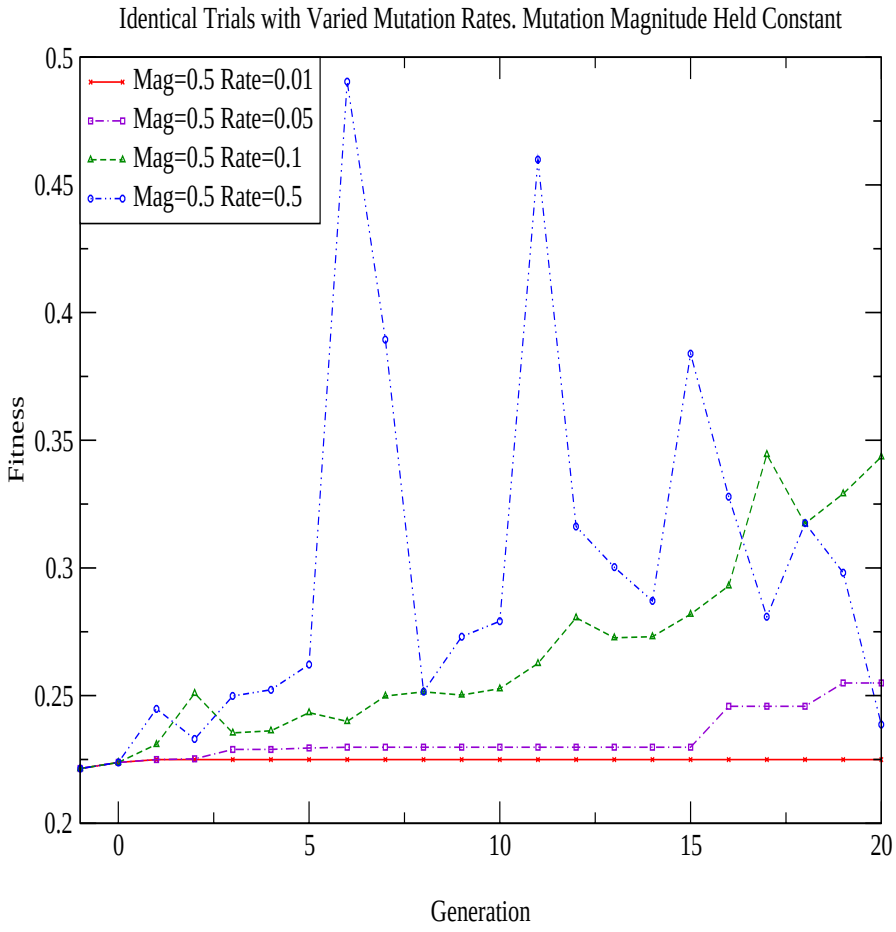

Figure 3.9: Comparison of Mutation Rates

In Figures 3.6, 3.7, and 3.8 it is demonstrated that an initial mutation magnitude of around 10% for all variations provides the best fitness gains while retaining convergence properties. The use of a decreasing function applied to the mutation magnitude shows a forced convergence towards the end of the evolution cycle. There do not appear to be any clear distinctions between the performance of linearly and quadratically decreasing mutation magnitudes. The effect of mutation rates, as shown in Figure 3.9 follow a similar trend as mutation magnitude. The rate that exhibits the best improvement while retaining convergence is also around 10%. As this study progressed, new features were added such as perturbation along a surface normal and addition of geometric constraints. These additions had the effect of altering the optimal values for mutation rate and magnitude. Furthermore, different starting geometries will respond differently to mutation rates and magnitudes such that it is difficult, if not impossible, to generally predict optimum values for these parameters. The study presented here serves to illustrate the existence of optimal values of mutation rate and magnitude for any given geometry.

3.3 Parallelization

Due to the fact that probabilistic methods often require many evaluations of an objective or fitness function, the cost of using a sequential algorithm is usually prohibitive. For example, in the case of the PCStage pre-optimizer using an evolutionary algorithm, an acceptable degree of convergence should be found after 50 generations have been evaluated. With a population size of seven members and a PCStage evaluation time of approximately

one minute per member, the total time necessary to produce the final solution is around 5 hours and 50 minutes. However, these methods are inherently parallelizable. That is, they are easily decomposed into simultaneous calculations. Largely due to the current accessibility of parallel and distributed computing, probabilistic methods are becoming more popular.

There are a variety of ways in which a GA may be parallelized [34]. In this study one of the simplest forms of Parallel Genetic Algorithm (PGA) has been implemented. The decomposition simply distributes the fitness evaluations for each member of the population to a separate processor. This would result ideally in a reduction of the time by the total size of the population, or at least by the number of processors available. Given our example of 50 generations and 7 members, the parallel version of the pre-optimzer would ideally produce the final solution in 50 minutes.

The actual degree of speedup is illustrated in Figure 3.10. For this speedup comparison each trial was run for five generations and population sizes were varied from one to seven. Plotting the completion time as a function of population size, the graph shows that both algorithms increase in a linear manner. The parallel algorithm trend has a slope of about one third that of the sequential algorithm. An ideal speedup would show a horizontal line at a completion time of 150 seconds. The actual speedup is obviously not ideal, but it definitely illustrates the advantage of parallelization. The trend indicates that a larger population size and hence a larger number of processors would provide a further increase in overall time reduction. An upper limit may exist in which communication costs outweigh the benefit

of more processors, but at the current level of investigation, this is not evidenced.

For the population size of one, the reader may notice that the parallel algorithm appears to achieve a speedup greater than that predicted by the speedup folklore theorem [35]. This is due to the fact that for any population size, the fitness of the original turbomachine is calculated at the beginning of the optimization. In the sequential code, this adds one additional evaluation to the total required. The parallel algorithm performs all evaluations on separate processors numbered from one to population size, and reserves processor zero for the optimization overhead, including the evaluation of the original turbomachine. This reduces the time needed for the parallel algorithm by one fitness evaluation, and it is the reason why it is necessary for the number of processors to be one greater than the population size.

Though most of the analysis was done using eight processors, the method would certainly benefit from a larger number of processors, and hence a larger population size. This would allow for a greater diversification of solutions and may offer greater performance through tailoring of the mating process.

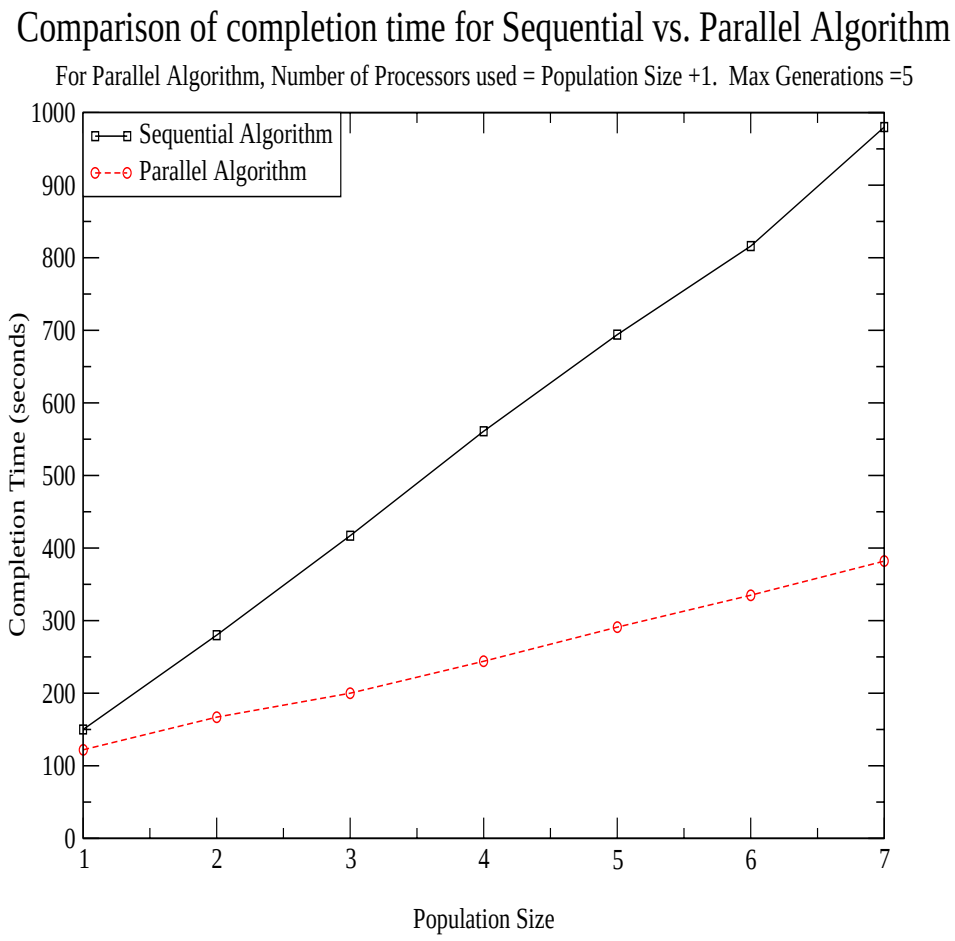

Figure 3.10: Comparison of Sequential vs. Parallel Algorithms

3.4 Error Handling/Geometric Constraints

The absence of an engineer's subjective, experiential, guidance from the optimization process presents a difficult problem. Designs which are clearly infeasible would likely be identified and discarded by an engineer. These designs must also be identified by the optimization program as infeasible. This results in the need to impose contraints, based on experience, while keeping these constraints to a minimum in order to limit restrictions on the design space. The following constraints were identified as critical and were therefore included in the optimization process. Geometries that return errors in the flow solver are assigned the minimum fitness value of zero. If they are not replaced by subsequent matings during the current round, they will be reset to the original geometry. Also, during the mutation stage, each blade of each member is checked for minimum thickness at every point. To retain the Kutta condition, the trailing edge control points (i.e., the last three for each surface) are held constant to fix the trailing edge flow angle. Further studies on constraint handling for Evolutionary Algorithms can be found in reference [36].

3.5 Investigation of Alternative Optimization Methods

3.5.1 Gradient Based Method

For the purposes of comparison, an unconstrained, gradient based (or sensitivity derivative) method was implemented using the Broyden, Fletcher, Goldfarb, Shanno (BFGS) variable metric method for direction finding [5]. These gradient based methods, briefly described in Section 1.2.1, use the

derivatives of the objective function with respect to each design variable to indicate the best way to alter the design variables in order to achieve an improvement in the objective function. Unlike the Evolutionary Algorithm, the performance of these methods in terms of CPU time is heavily dependent on the number of design variables involved. The gradient based method has the capability of finding the exact optimum, but has the tendency to become stalled in local optima. Like the EA method, gradient based methods are well suited to parallelization. This method has been parallelized such that each of the derivatives with respect to the objective function are calculated simultaneously. In addition, the line search performs a number of simultaneous function evaluations at distinct intervals along the search direction. The BFGS method was chosen due to the fact that it does not require exact line searches to achieve improvements each iteration. This allows us to be less precise in our line searches thereby reducing the number of flow solutions necessary per design iteration.

The test case used for this comparison was the VKI turbine guide vane cascade whose properties are described in Section 4.2. The objective for the following case is to increase the value of the axial thrust coefficient. The BFGS method used for direction finding is an unconstrained optimization method. This means that every value within the domain is treated without bias. Constraints have been applied to the geometry by simply limiting the domain for each design variable such that a minimum thickness is enforced and axial boundaries are maintained. These constraints are the same as those defined for the Evolutionary Algorithm in Section 3.4 and additional constraints are described in section 4.2.

Table 3.1 lists the five aerodynamic coefficients calculated by PCStage. These coefficients are: axial thrust (C_x), tangential thrust (C_y), moment (C_m), lift (C_l), and absolute drag ($|C_d|$). The parallelized sensitivity derivative algorithm was run using eight processors and took approximately one hour to run eight iterations. This optimization generated a 2.0% increase in the axial thrust coefficient. The geometry produced by this optimization is compared to the original in Figure 3.11.

Table 3.1: VKI PCStage Analysis, Optimized Using Sensitivity Derivatives (Increased C_x)

	C_x	C_y	C_m	C_l	$\|C_d\|$
Original	14.00077	7.307141	1.645532	15.79246	0.1187053
Optimized	14.27599	7.820427	-1.505815	16.27588	0.2424562

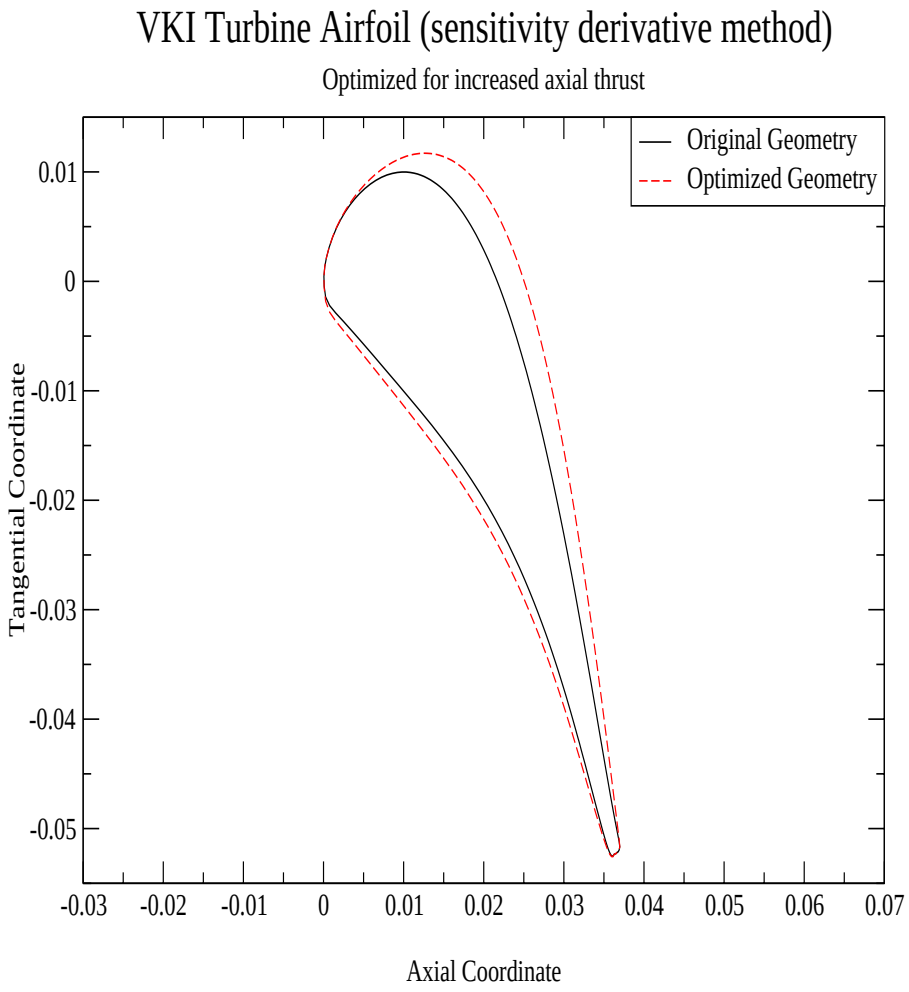

Figure 3.11: Geometry Comparison for VKI with Increased C_x, Obtained Using Sensitivity Derivatives.

3.5.2 Hybrid Evolutionary-Gradient Based Method

The ability of gradient based methods to provide exact optima is the ultimate goal of all optimization efforts. However, their tendency to stall in the local optima of a complex design space is a significant drawback. Evolutionary algorithms have the ability to provide improved geometries that are near optimal in the global domain, though their inability to produce exact optima renders them less than perfect. These methods both excel where the other fails, thereby lending themselves to a complementary combination. For further information on the variety of "hybrid" optimization schemes, the reader is directed to the literature [13, 14, 15].

A trial has been run in which the initial optimization of the VKI cascade was performed using the EA. This optimization ran for twenty generations which took approximately 18 minutes and produced a 22.1% increase in the axial thrust coefficient, compared to the 2.0% increase obtained by the gradient based method in the previous example. This indicates that the gradient based method had converged directly into a local optimum. The EA improved geometry was then refined using the BFGS gradient based method which converged in seven design iterations taking approximately 40 minutes, and resulted in a 104.5% overall improvement. These results are listed in Table 3.2.

Table 3.2: VKI PCStage Analysis, Optimization Utilizing an Evolutionary Algorithm with Refinement via Sensitivity Derivatives (Increased C_x)

	C_x	C_y	C_m	C_l	$\|C_d\|$
Original	14.00077	7.307141	1.645532	15.79246	0.1187053
EA Optimum	17.09682	8.129745	2.003717	18.93064	0.1584643
SD Refined	28.64504	11.05988	-2.140496	30.70276	0.4470350

The results of this test case clearly illustrate the advantage of utilizing zeroth order and gradient based methods in conjunction to overcome the disadvantages of both. The EA produced an optima that was clearly more globally fit than that produced by the gradient based method alone. The large improvement obtained from using gradient based refinement on the result of the EA optimization indicates that the near optima was far from exact. The resulting geometry therefore has the property of being much more globally fit and is at the peak of fitness in its local domain. The geometries produced by the successive design steps are shown in Figure 3.12.

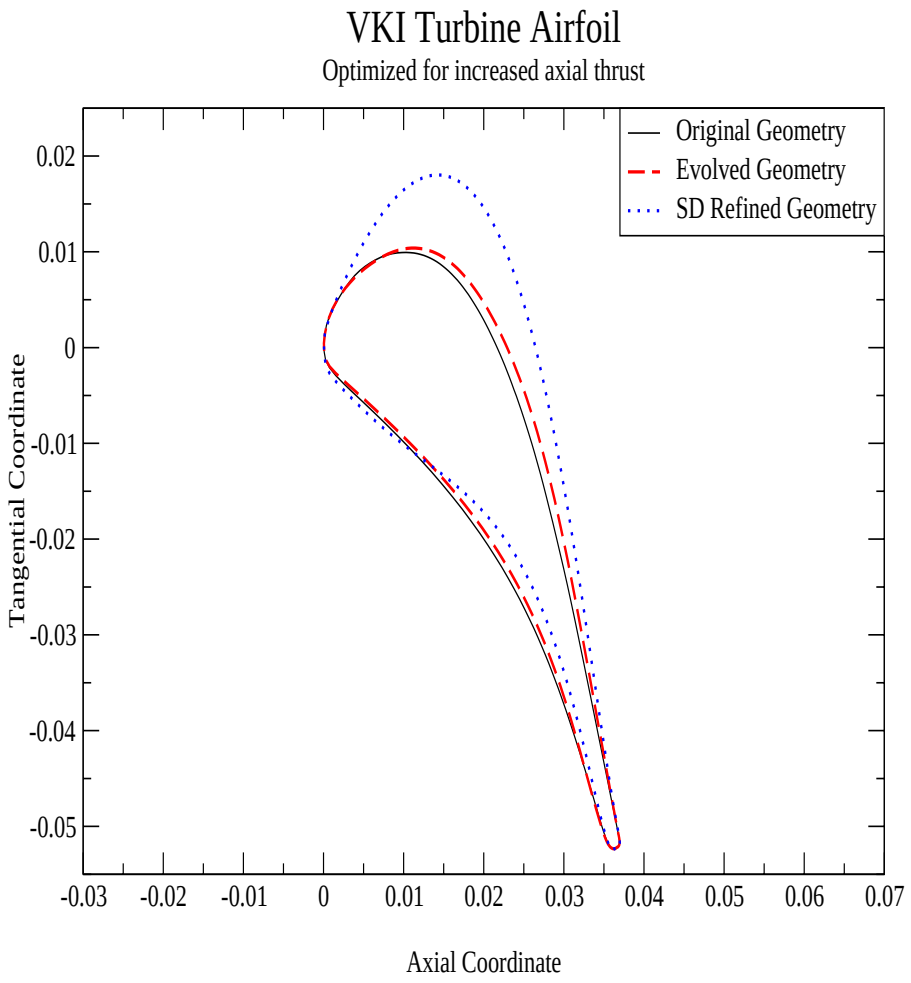

Figure 3.12: Geometry Comparison for VKI with Increased C_x, Obtained Using an Evolutionary Algorithm and Refined Using Sensitivity Derivatives.

CHAPTER IV

RESULTS AND DISCUSSION

Two turbine geometries have been examined in this study. The first case is a single stage (stator-rotor) turbomachine which was experimentally tested at the United Technology Research Center's (UTRC) Large Scale Rotating Rig Facility. The results for this particular turbine geometry were presented in a paper by Dring [37] and subsequently used for comparison in McFarland's PCStage verification [21]. The second case is a von Karman Institute (VKI) designed turbine guide vane cascade [38, 22]. This guide vane cascade geometry has been optimized and the PCStage results have been compared with an Euler analysis to determine any discrepancies between the panel method solutions and those of higher fidelity methods. The geometries were optimized for various properties and the analysis has been presented in the following sections. For all cases, a linearly decreasing mutation magnitude was used, and the crossover method chosen was fixed single point for each surface in the geometry.

4.1 Dring's Low Speed Axial Turbine

Dring's low speed axial turbine, which will be referred to as "DAT", consists of two blade rows. Row one, located upstream from row two, is

a stator containing 28 blades and has a trailing edge flow angle of -67^o. Row two contains 28 blades and rotates at a speed of 408.1 RPM in the positive tangential direction (as defined in Figures 4.1,4.3,4.5,4.7). Row two has a trailing edge flow angle of 61.5^o. The inlet flow velocity is 75 ft/sec with a flow angle of 0^o. The outlet flow angle is calculated at 40^o with a velocity of 100.2 ft/sec. The working fluid is air with an upstream total pressure and temperature of 2116 lb/ft^2 and 520 R, respectively. The initial and subsequent mutation magnitudes were 7% and 3%, respectively. The mutation rate was set at a constant 30%. The coefficient of pressure calculated by PCStage and used for comparison in this chapter is formulated in equation 4.1. The value of M_{inlet} is calculated directly from the input flow values.

$$C_p = \frac{(P - P_{inlet})}{\frac{1}{2}\gamma P_{inlet} M_{inlet}^2} \tag{4.1}$$

Optimization trials were run to improve the aerodynamic coefficient values of axial thrust (C_x), tangential thrust (C_y), and absolute drag ($|C_d|$) separately. A fourth optimization was then conducted utilizing a linear combination of these coefficients as the fitness function. The results are presented in Sections 4.1.1, 4.1.2, 4.1.3, and 4.1.4.

For these inviscid cases the value of the PCStage calculated drag coefficient (C_d) is composed of two parts: A measure of the numerical error which is second order accurate with respect to panel size, and the value of the integrated pressure forces along a mean flow angle. The integrated

pressure forces do not sum to zero for an airfoil in a cascade due to the effects of flow turning at upstream and downstream infinity.

4.1.1 Increased Average Axial Thrust

The objective of this first optimization is to increase the average value of the axial thrust coefficient (C_x). Table 4.1 lists the values of the aerodynamic coefficients for each blade row in the turbomachine and the average of these values. A comparison is made between the original geometry of Gen = 0 and the final "evolved" geometry of Gen = 25. By inspection of the data for the target coefficient C_x, it is evident that although the value has been increased for the stator, it has been reduced on the rotor. The combination still results in an overall improvement in average C_x. Inspection of the other coefficients show that average C_y and C_l have been reduced while average C_m and $|C_d|$ have been increased.

Table 4.1: DAT PCStage Analysis (Increased Average C_x)

Gen	Row	C_x	C_y	C_m	C_l	$\|C_d\|$
0	1	6.510614	4.264202	1.243707	1.537078	7.629476
	2	1.070044	-6.462905	-1.197102	-6.397294	1.410233
	Avg	3.790329	-1.099351	0.0233023	-2.430108	4.519854
25	1	7.200757	4.502012	1.662351	1.265173	8.397520
	2	0.7400496	-6.877953	-1.083056	-6.604517	2.057732
	Avg	3.970403	-1.187971	0.2896475	-2.669672	5.227626

The evolved geometry generated for an improved average C_x is shown in Figure 4.1. There is a very small change in the stator geometry, and a much larger change in the rotor geometry resulting in a rather blunt leading edge. These changes were not indicated by the relatively large increase of C_x on the stator and small decrease of C_x on the rotor. The percentage difference in C_x however, does indicate these changes. The optimization effects a 10.6% increase of C_x on the stator. A decrease of 30.8% is achieved on the rotor. Although the optimizer results show an increase in average C_x, the drastic change in geometry produces a rather large percentage decrease on the rotor alone. This indicates that the use of a fitness function based on average coefficient values is a poor indicator of performance for blade rows whose coefficient magnitudes vary significantly. A more judicial fitness function may utilize average percentage changes to indicate performance improvements.

The pressure coefficient plot in Figure 4.2 shows very little pressure change due to stator geometry but a rather large pressure change just after the leading edge of the rotor on the suction surface corresponding to the increase in surface area and blunt leading edge.

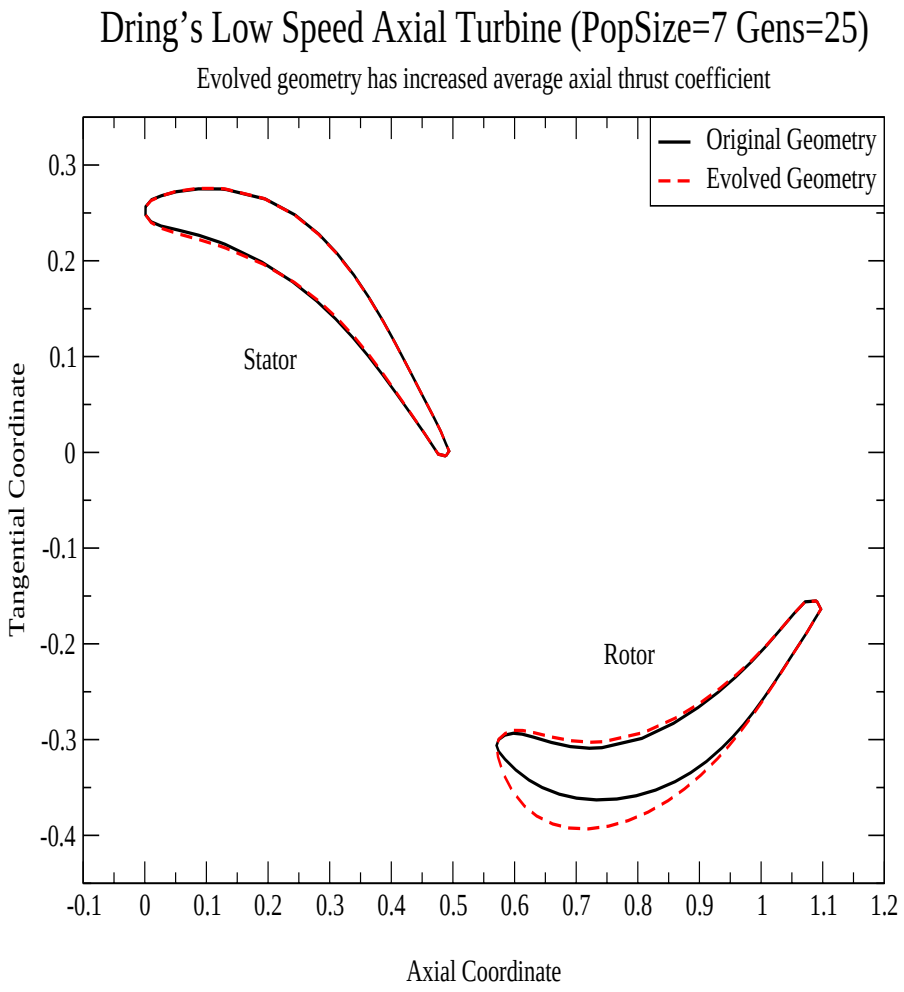

Figure 4.1: Geometry Comparison of DAT with Increased Average C_x .

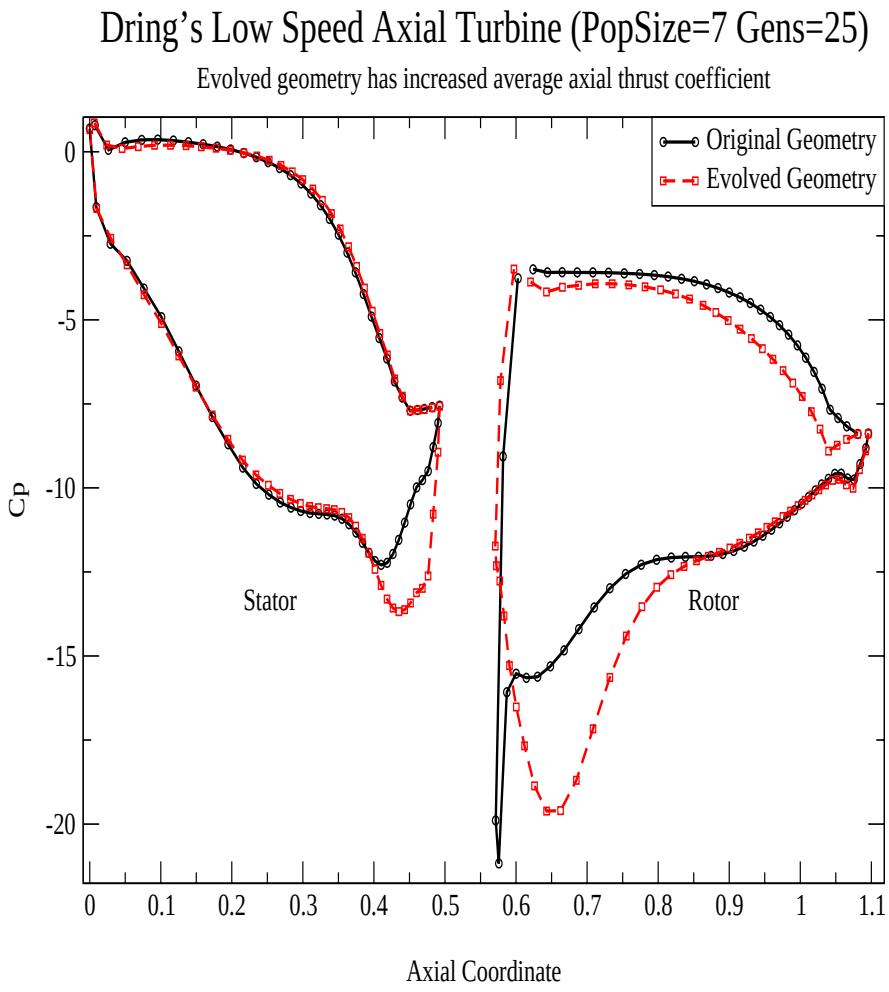

Figure 4.2: C_p Comparison of DAT with Increased Average C_x .

4.1.2 Increased Average Tangential Thrust

The objective is here to increase the average value of tangential thrust (C_y). From Table 4.2, a comparison is made between the coefficients of the original and evolved geometries. The optimization results in an increase of tangential thrust for both blade rows.

Table 4.2: DAT PCStage Analysis (Increased Average C_y)

Gen	Row	C_x	C_y	C_m	C_l	$\|C_d\|$
0	1	6.510614	4.264202	1.243707	1.537078	7.629476
	2	1.070044	-6.462905	-1.197102	-6.397294	1.410233
	Avg	3.790329	-1.099351	0.0233023	-2.430108	4.519854
25	1	6.469320	4.282306	1.384830	2.119511	7.463104
	2	0.5252466	-6.033959	-1.168760	-5.909227	1.328749
	Avg	3.497283	-0.8758234	0.1080346	-1.894858	4.395927

Due to the fact that the relative magnitudes of C_y are more comparable here than of C_x in the previous example, the fitness function is affected more evenly by changes in both geometries. The geometric result of this more evenly weighted evaluation can be seen in Figure 4.3. The stator undergoes a slight thinning, and the rotor now has an increasingly blunted leading edge and reduced thickness along the chord due to a flattening of the suction surface.

With the exception of the rotor suction surface, the coefficient of pressure plotted in Figure 4.4 shows a distribution very similar to the original.

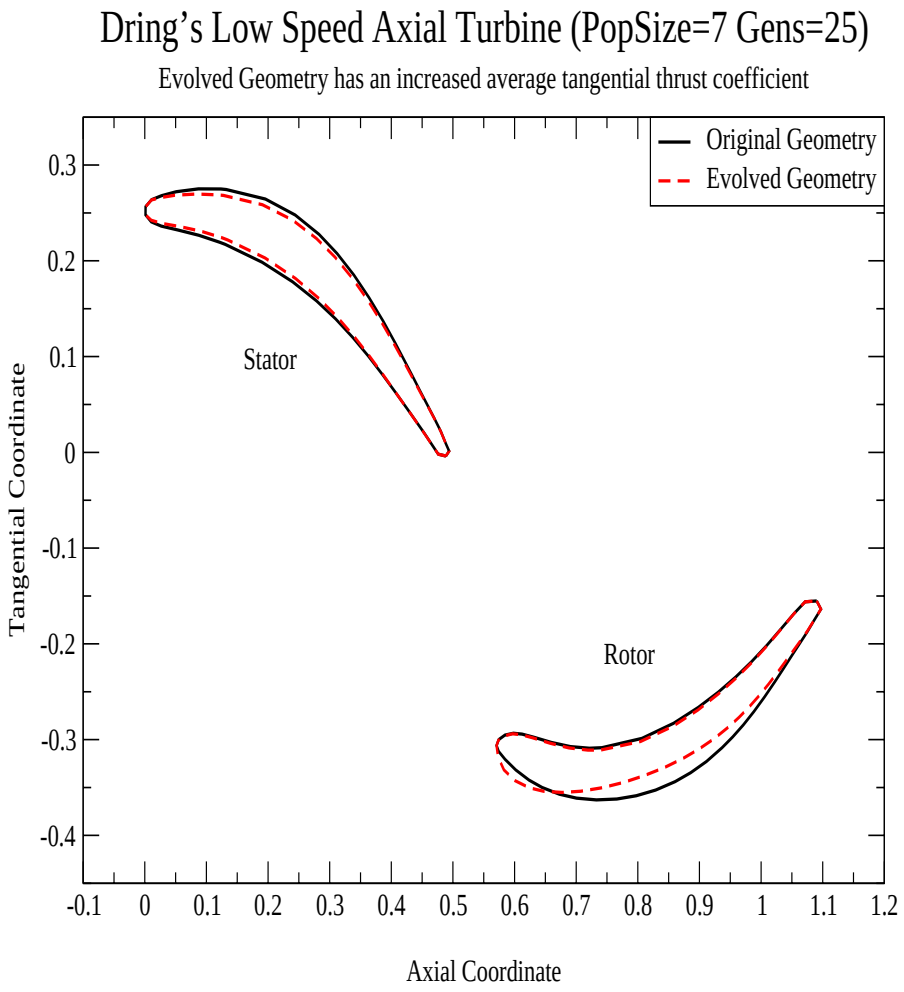

Figure 4.3: Geometry Comparison of DAT with Increased Average C_y.

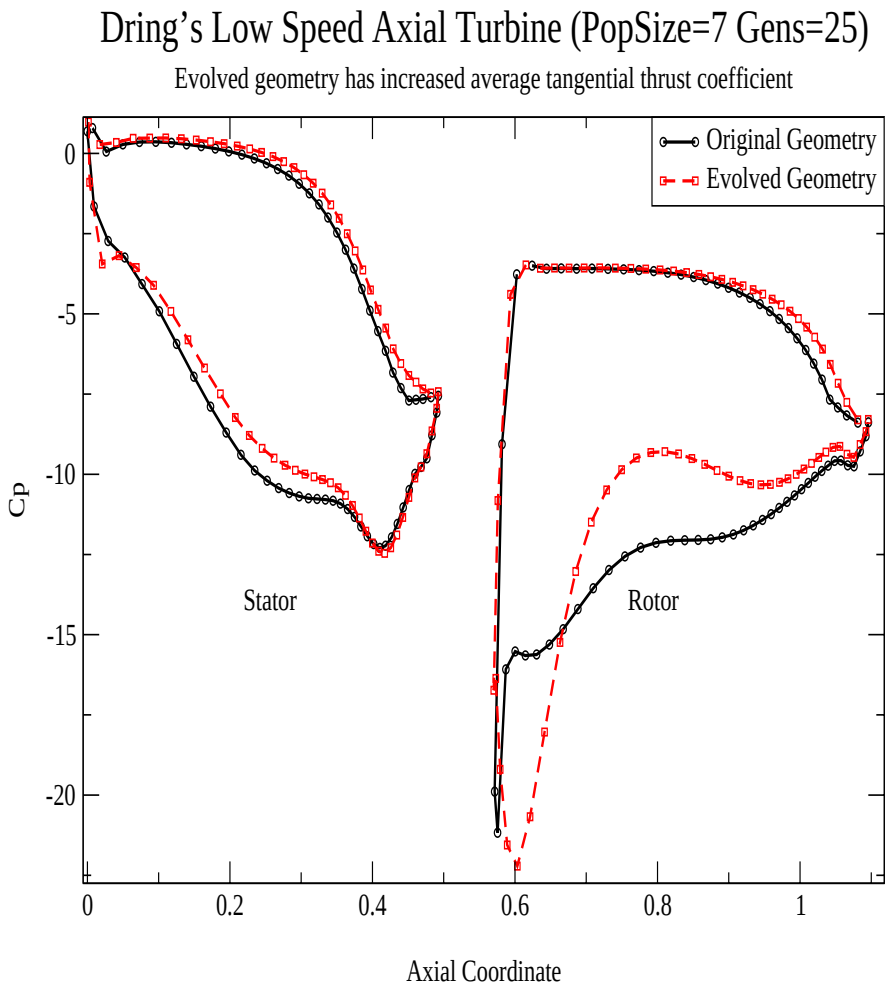

Figure 4.4: C_p Comparison of DAT with Increased Average C_y.

4.1.3 Decreased Average Drag

The objective for this optimization was a decrease in the average absolute drag coefficient ($|C_d|$). The results listed in Table 4.3 show a decrease in $|C_d|$ for both blade rows. Effects on the other coefficients are an increase in average C_y and C_l and a decrease in average C_x, and C_m.

Table 4.3: DAT PCStage Analysis (Decreased Average $|C_d|$)

Gen	Row	C_x	C_y	C_m	C_l	$\|C_d\|$
0	1	6.510614	4.264202	1.243707	1.537078	7.629476
	2	1.070044	-6.462905	-1.197102	-6.397294	1.410233
	Avg	3.790329	-1.099351	0.0233023	-2.430108	4.519854
50	1	3.642239	3.159335	0.2051005	2.449147	4.153190
	2	1.233360	-4.348439	-0.6978505	-4.499727	0.4272640
	Avg	2.437800	-0.594552	-0.2463750	-1.025290	2.290227

To achieve this drag coefficient reduction, the evolved geometry in Figure 4.5 exhibits a large change in both blade rows. As one might expect, the thickness of the blades has been reduced drastically and the surfaces are becoming aligned with the mean flow angle. The stator exhibits a slight S-curve near the trailing edge. The optimization clearly shows a decrease in the total amount of drag. However, this may be in part a result of the inability of the low fidelity simulation to accurately model the physics in the vicinity of the trailing edge. The stator in particular has a trailing edge angle that is clearly different from the original geometry. Due to the fact that the trailing edge angle for the Kutta condition is fixed to match the original velocity triangle, the flow on the optimized stator is subjected to a twist at the

trailing edge to match the Kutta angle. The actual trailing edge flow angle will most likely deviate from the expected angle, therefore the reliability of this result is in question. Problems such as this may be overcome by applying more stringent constraints to the trailing edge geometry.

The pressure coefficient plot for this result is shown in Figure 4.6. As expected, the pressure coefficients of the new design vary drastically from the original, particularly along the surface of the stator.

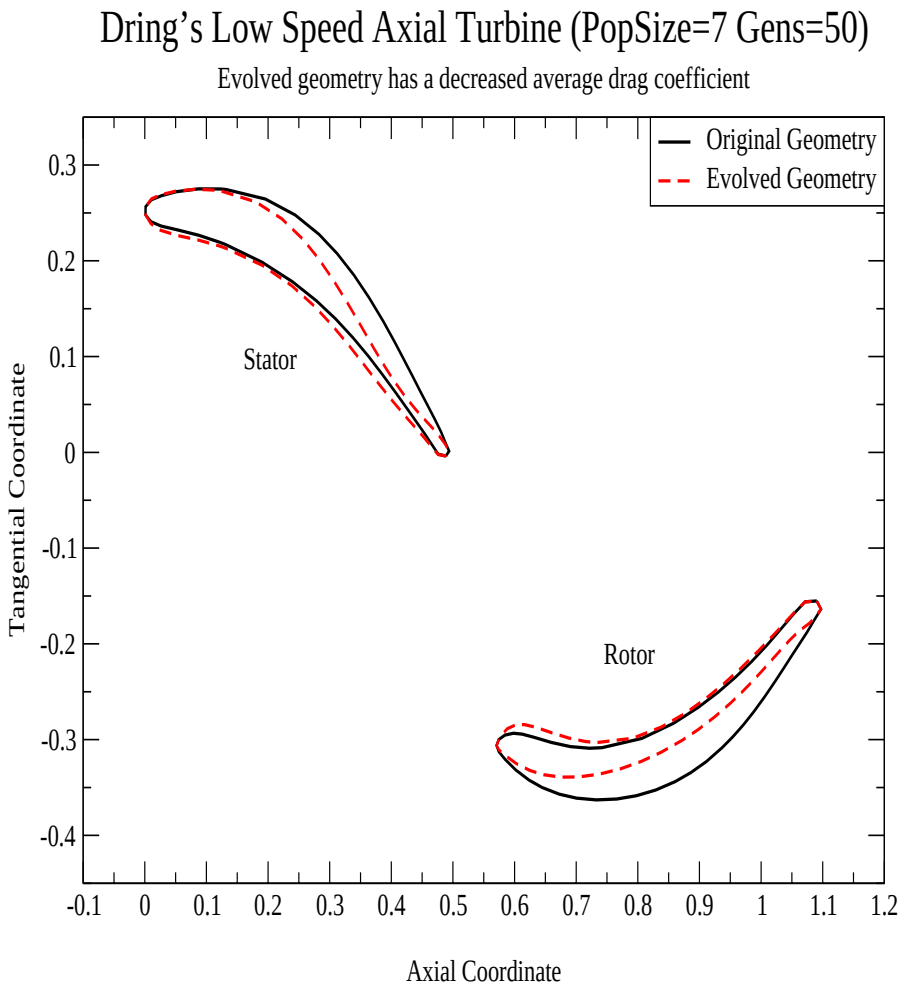

Figure 4.5: Geometry Comparison of DAT with Decreased Average $|C_d|$.

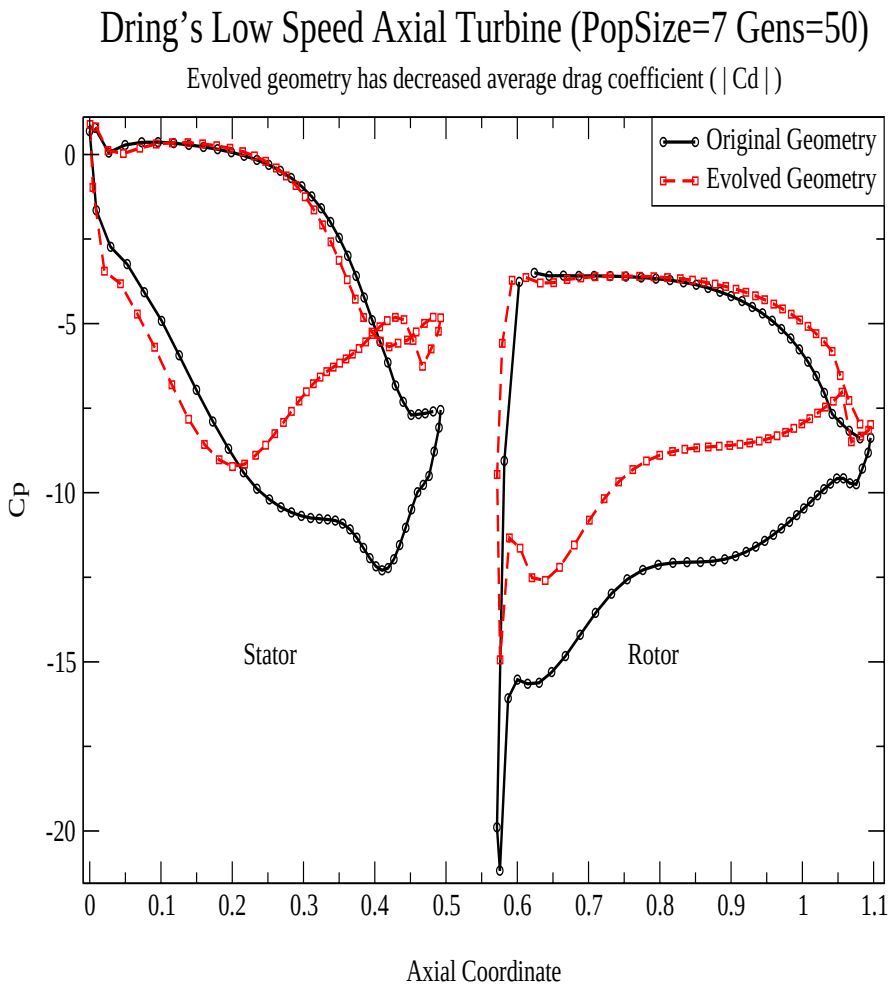

Figure 4.6: C_p Comparison of DAT with Decreased Average $|C_d|$.

4.1.4 Multiobjective Optimization

Table 4.4 lists the results of a multiobjective fitness function designed to increase the sum of the averages of C_x, and C_y, while decreasing the average value of $|C_d|$. A comparison of the average coefficients of the original geometry to those of the evolved geometry illustrates a decrease in average C_x, an increase in average C_y, and a decrease in average $|C_d|$. The sum results in a total increase for the combined coefficients.

Table 4.4: DAT PCStage Analysis (Increased Combination: $C_x + C_y - |C_d|$)

Gen	Row	C_x	C_y	C_m	C_l	$\|C_d\|$
0	1	6.510614	4.264202	1.243707	1.537078	7.629476
	2	1.070044	-6.462905	-1.197102	-6.397294	1.410233
	Avg	3.790329	-1.099351	0.0233023	-2.430108	4.519854
25	1	5.351418	3.858809	0.8009851	1.807322	6.345208
	2	2.081053	-5.646716	-1.175747	-6.017885	0.035184
	Avg	3.716235	-0.8939535	-0.1873810	-2.105282	3.190196
Combined Coefficients at Gen=0						-1.828876
Combined Coefficients at Gen=25						-0.3679145

The evolved geometry for this case, shown in Figure 4.7 is quite different from the previous examples. The stator remains relatively unchanged, but the rotor deviates drastically. The leading edge of the rotor is quite thin and creates a large increase in the blade curvature. A downward pointing cusp was created at the leading edge due to the fixed leading edge control point constraint.

Although the geometry is very similar for the stator, the coefficient of pressure plot, as seen in Figure 4.8 shows a pressure increase on the suction

surface near the trailing edge. As expected, the new rotor geometry results in a drastic change in pressure coefficients. The new design exhibits a much greater magnitude in the pressure spike at the leading edge of the rotor. The pressure then returns to a much higher value than the original and remains higher overall for the rest of the span.

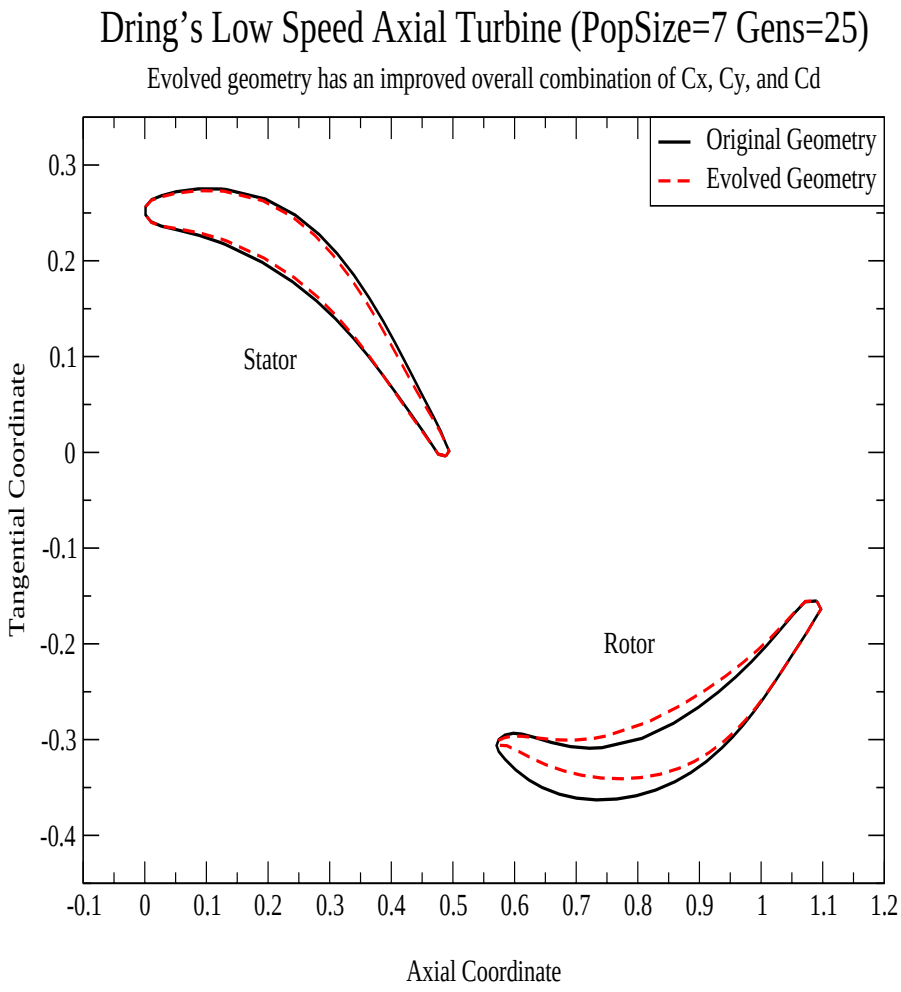

Figure 4.7: Geometry Comparison of DAT with Increased Combination of Average $C_x + C_y - |C_d|$.

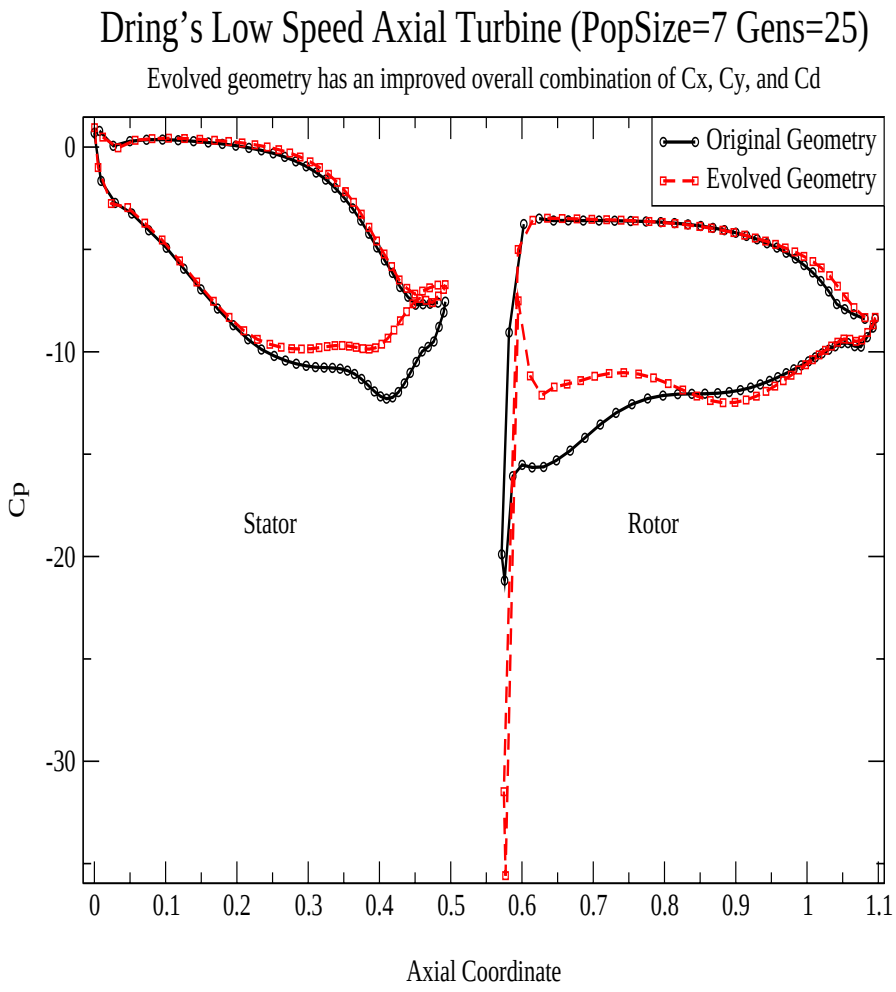

Figure 4.8: C_p Comparison of DAT with Increased Combination of Average $C_x + C_y - |C_d|$.

4.1.5 Summary of Results for Dring's Axial Turbine

According to PCStage, all four of these trials have resulted in an overall improvement of the selected aerodynamic coefficients. The reality of these results has yet to be seen. The results for decreased drag are suspect, due to the linear panel method's use of a Kutta condition at the trailing edge. Also, the design resulting from the multiobjective optimization has a cuspidal leading edge which may cause large flow separations that would not be predicted by PCStage. Although constraints have been included to reduce the amount of geometric infeasabilities, it is clear that further constraints for all cases need to be imposed. In addition, a reduced mutation magnitude would result in geometries more closely resembling the original and therefore less likely to contain questionable features. For multiple row cases such as this, a more evenhanded fitness function should be implemented in order to reduce the tendency of the optimizer to focus only on improving the blade with the largest coefficient value.

The next test case incorporates some of the lessons learned from the DAT optimization study. The mutation magnitudes have been reduced and greater constraints have been imposed. In addition, verificaton will be conducted by comparing the results with an Euler analysis.

4.2 VKI Guide Vane Cascade

The VKI guide vane shown in structured grid form in Figure 4.9, is a linear cascade composed of blades with a chord length of approximately 67 mm. For the PCStage input deck the flow was determined by the

following parameters: The inlet flow Mach number is 0.1 and inlet flow angle is 0^o. The working fluid is air, with a freestream temperature of 416 K and pressure of 0.32570×10^6 N/m^2. The mutation magnitudes were kept quite small for this case. The initial magnitude was 3% and the subsequent magnitude was 1%. The subsequent mutation rate was fixed at 30%.

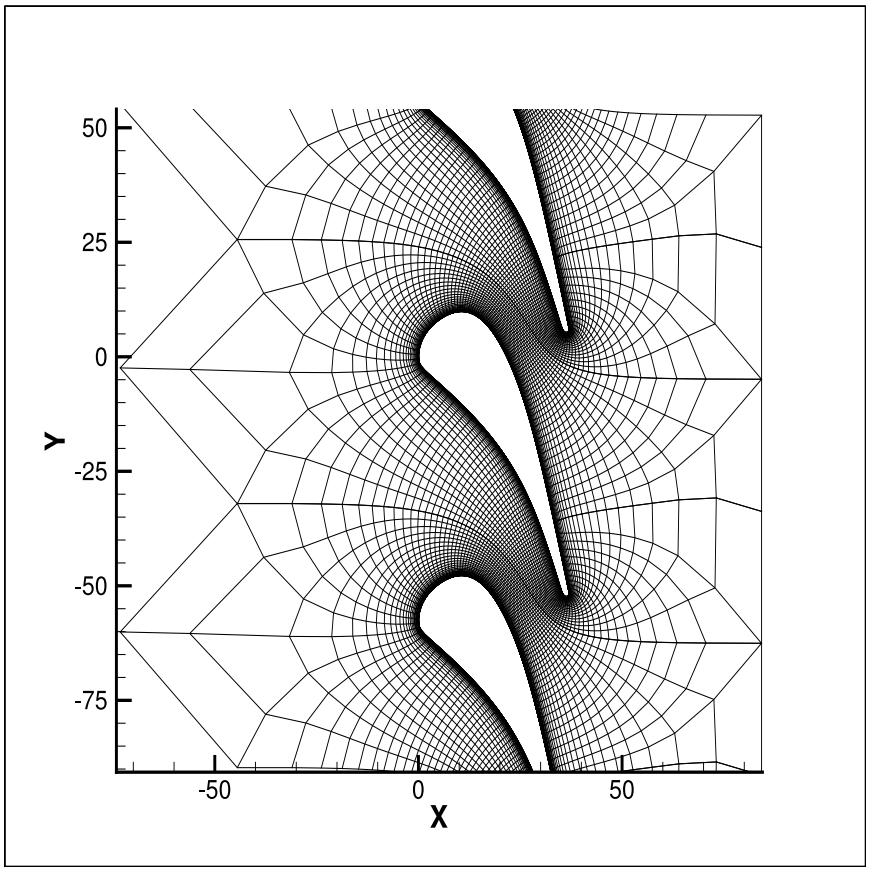

Figure 4.9: Original VKI Geometry and Structured Grid

The constraints applied to the VKI case are as follows: The control points interpolating the leading and trailing stagnation points are held constant. In addition, perturbation of the control points above and below the leading point are constrained to have an x position greater than or equal to the leading point. Furthermore, the adjacent control points above and below the leading point are constrained such that their y positions are greater and less than that of the leading point respectively. These constraints reduce the possibility of producing a cuspidal leading edge. Around the trailing edge, the group of points surrounding the stagnation point have been fixed in order to maintain the predetermined trailing edge flow angle. In addition to these constraints, a lower limit on the allowable thickness has been set to 1.5 mm which is equal to the thickness of the original trailing edge. Due to the fact that the VKI case involves only a single blade row, the fitness function was not modified to reduce unevenly weighted components. A weighted function may however, be beneficial in the multiobjective fitness function in order to evenly balance the importance of each of the objectives involved.

The VKI analysis follows the same procedure as DAT. The VKI guide vane was optimized for the three separate aerodynamic coefficients of axial thrust (C_x), tangential thrust (C_y) and absolute drag ($|C_d|$) respectively. A fourth optimization was performed in which the fitness function was composed of a linear combination of the same three coefficients of the previous examples.

The VKI optimized geometries are also compared to the results of an Eulers solution for the same geometry. The Euler solution was calculated using the inviscid mode of the Navier-Stokes solver NS2D developed by

Mark Janus [22]. The original VKI geometry grid used in the Euler solution is shown in Figure 4.9. The inlet Mach number calculated by the Euler algorithm is dependent on the outlet fluid pressure. In order to approximate the desired inlet Mach number (used as a starting condition in the PCStage calculation) the value of outlet pressure calculated by PCStage was used as an input parameter for the Euler code.

The Euler code returns the values of average axial (F_x) and tangential (F_y) forces in Newtons, and the nondimensional loss coefficient (ζ). The loss coefficient is defined in equation 4.2. Though not directly comparable, F_x and F_y are linear functions of C_x and C_y and should follow the same trend. The values of $|C_d|$ and loss coefficient ζ are less comparable due to the fact that for the inviscid case, the calculated outlet velocity should be the same as the isentropic outlet velocity, so ζ will only be a measure of numerical error. A percentage change is calculated to better illustrate the relative comparability of these methods.

In addition to comparing calculated force coefficient values, the C_p plots calculated by each method are compared for each VKI optimization case. The definition of C_p used in the Euler code is shown in equation 4.3. The values of C_p calculated by PCStage and NS2D should be equivalent provided the NS2D calculation is normalized with the inlet Mach number (M_{init}) as used in the PCStage calculation.

Although efforts were made to ensure a similar degree of convergence for all geometries, the convergence of the Euler solutions for the original and perturbed designs are not identical. Therefore the comparison of the Euler

calculated properties between the original and optimized geometries should be close but are not expected to be exact.

The formulation of C_p for the Euler solution is presented in equation 4.3.

$$\zeta = 1 - \frac{V_{out}^2}{V_{out,isen}^2} \tag{4.2}$$

$$C_p = \frac{(P - P_{inlet})}{\frac{1}{2}\rho_{init}\ a_{init}^2 M_{init}^2} \tag{4.3}$$

The results and analysis are presented in Sections 4.2.1, 4.2.2, 4.2.3, 4.2.4.

4.2.1 Increased Axial Thrust

The aerodynamic coefficients calculated by PCStage are presented in Table 4.5 for both the original (Gen = 0) and redesigned geometry (Gen = 20). The desired result in this case, is an increase in the value of the axial thrust coefficient. The results show an improvement in both C_x and C_y with a very small increase in $|C_d|$.

Table 4.5: VKI PCStage Analysis (Increased C_x)

Gen	C_x	C_y	C_m	C_l	$\|C_d\|$
0	14.00077	7.307141	1.645532	15.79246	0.1187053
20	16.48850	7.951503	2.125335	18.30527	0.1194973

Table 4.6: VKI Euler Analysis (Increased C_x)

Gen	F_x (N)	F_y (N)	ζ (%)
0	608.8968	330.3311	2.026984
20	726.8404	358.7554	2.137962

Table 4.6 lists the results of an Euler analysis of the original and redesigned geometries. Table 4.7 lists the percentage change calculated by each method. The changes in C_x and C_y are very similar to the changes in F_x and F_y. The change in $|C_d|$ however, is not mirrored very well by the change in ζ. This is expected due to the differences in definition.

Table 4.7: VKI Percentage Change in Selected Aerodynamic Coefficients (Increased C_x)

PCStage	$C_x = 17.7\%$	$C_y = 8.8\%$	$\lvert C_d \rvert = 0.67\%$
NS2D	$F_x = 19.4\%$	$F_y = 8.6\%$	$\zeta = 5.4\%$

At Gen = 20, the Euler computed values show an increase in F_x, F_y, and loss coefficient ζ. The optimizer has produced a geometry that according to the Euler analysis, has the desired improvement in C_x. The increase in both F_y and ζ were also predicted by PCStage. When examining the relative agreement between the PCStage panel method and Euler solution for this and the following cases it's important to keep in mind the relative CPU costs of each. For PCStage a typical flow solution takes approximately one minute to complete. The Euler solution typically takes about one hour

to produce a converged solution. Therefore, any discrepancies between the solutions are mitigated by a large CPU cost reduction.

The geometry produced by this optimization (Figure 4.10) exhibits a slight elevation along the length of the suction surface, leaving the rest of the surface relatively unchanged.

A comparison between the pressure coefficients calculated by PCStage and the Euler code is presented in Figure 4.11. The plot shows relatively good agreement, with some deviation along the suction surface.

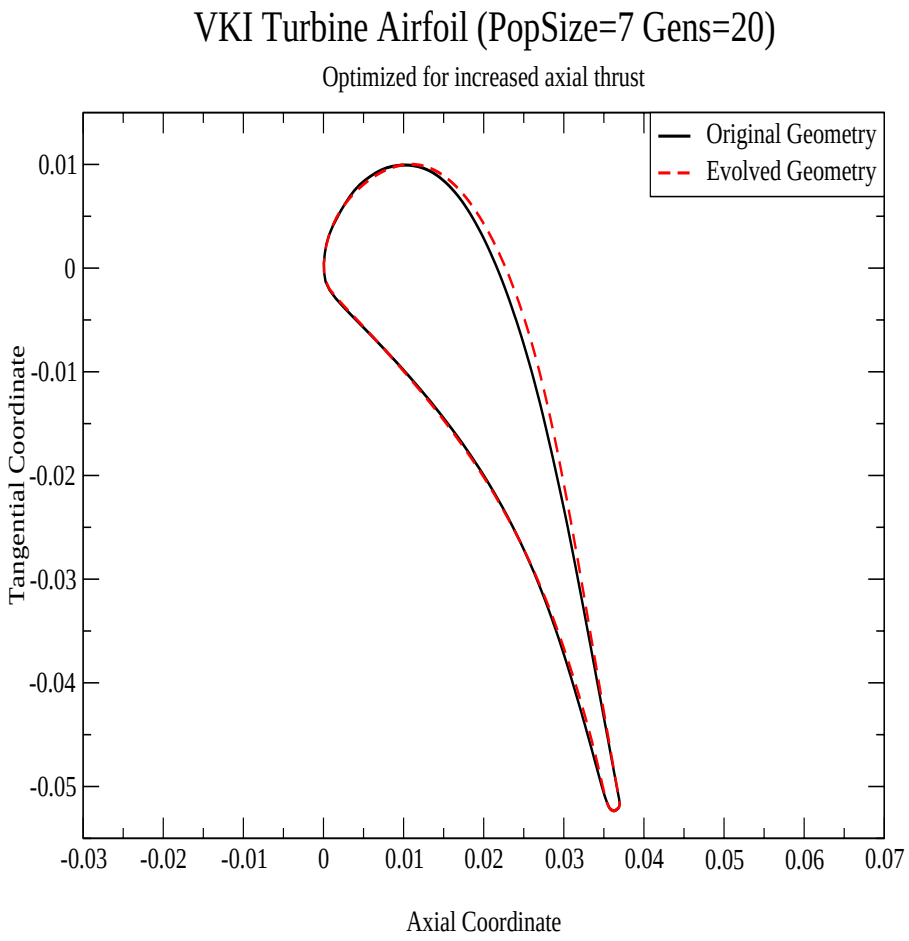

Figure 4.10: Geometry Comparison for VKI with Increased C_x.

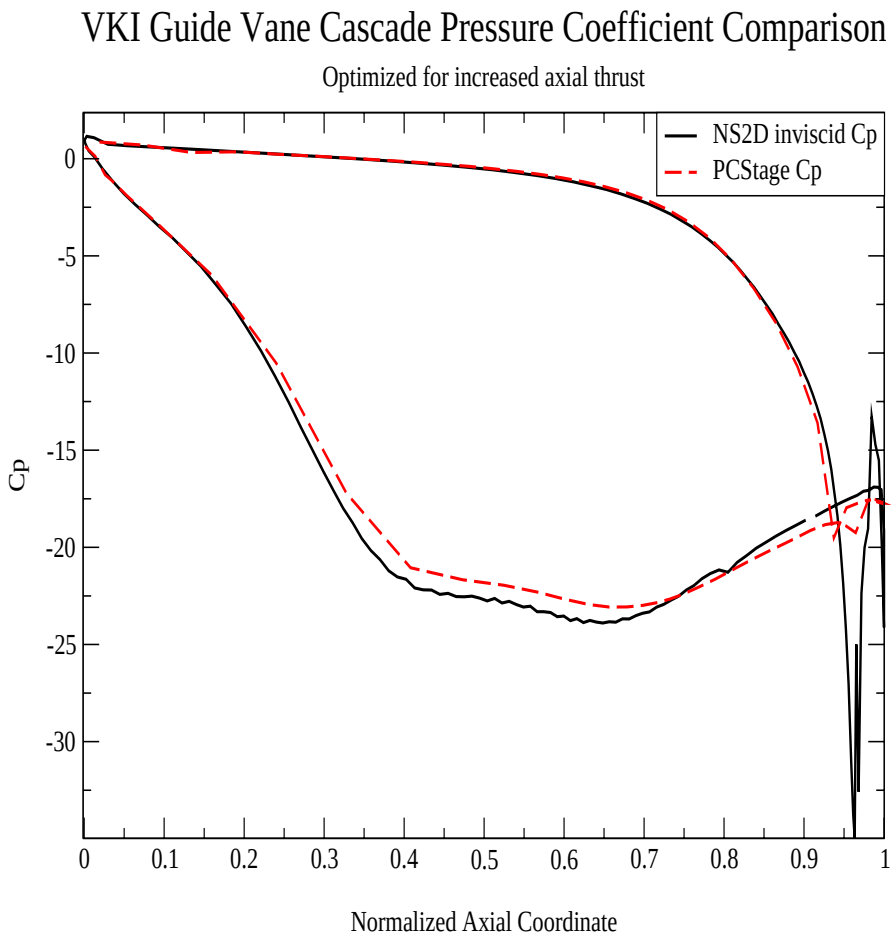

Figure 4.11: C_p Comparison for VKI with Increased C_x .

4.2.2 Increased Tangential Thrust

The results for the optimization designed to increase tangential thrust are presented in Table 4.8. PCStage calculates an increase in both C_x, C_y, and $|C_d|$.

Table 4.8: VKI PCStage Analysis (Increased C_y)

Gen	C_x	C_y	C_m	C_l	$\|C_d\|$
0	14.00077	7.307141	1.645532	15.79246	0.1187053
20	16.29585	7.933486	1.855481	18.12367	0.1656370

Table 4.9: VKI Euler Analysis (Increased C_y)

Gen	F_x (N)	F_y (N)	ζ (%)
0	608.8968	330.3311	2.026984
20	690.7169	348.1418	2.133284

The Euler verification of the optimized geometry (Table 4.9) shows an increase in F_x, F_y, and ζ. The target coefficient C_y has been improved and the increase in F_x and ζ are reflected in the PCStage analysis. The percentage change listed in Table 4.10 does not show very good agreement in the percentage change of tangential thrust.

Table 4.10: VKI Percentage Change in Selected Aerodynamic Coefficients (Increased C_y)

PCStage	$C_x = 16.4\%$	$C_y = 12.7\%$	$\lvert C_d \rvert = 39.5\%$
NS2D	$F_x = 13.4\%$	$F_y = 5.39\%$	$\zeta = 5.2\%$

The geometry generated for improved C_y (Figure 4.12) is very similar to the previous example involving improved C_x with the addition of a slightly raised pressure surface. The corresponding PCStage and Euler calculated pressure coefficients shown in Figure 4.13 also deviate very little from the previous example.

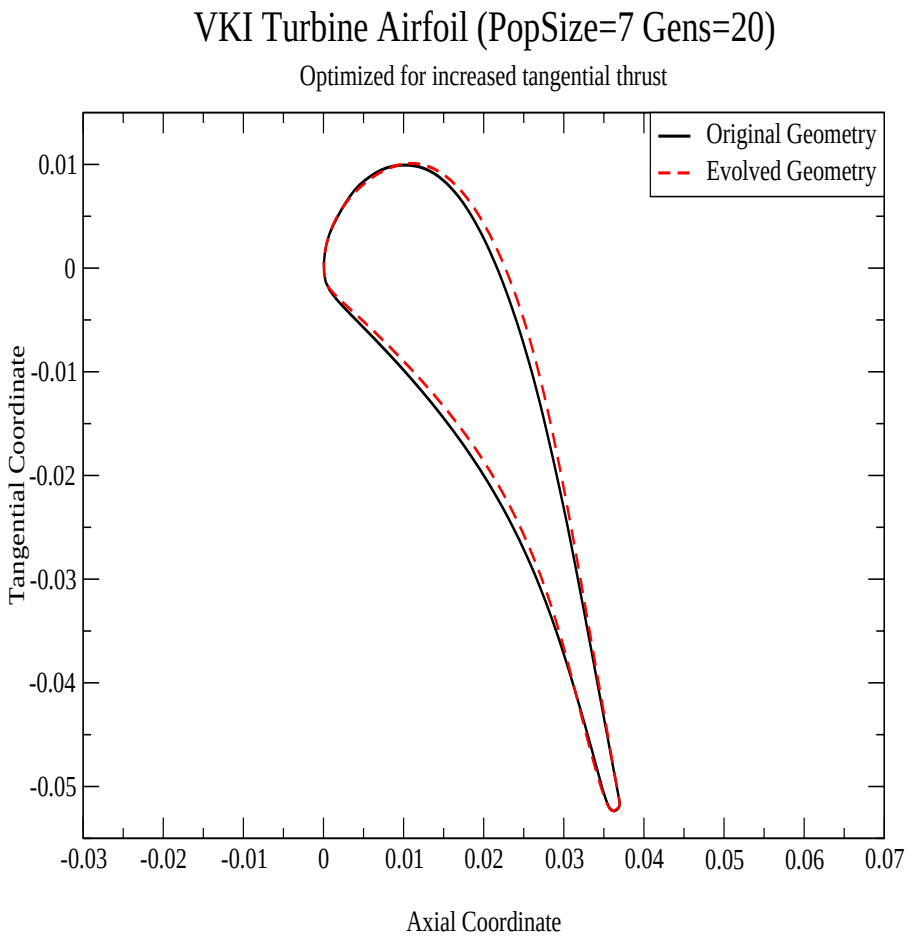

Figure 4.12: Geometry Comparison for VKI with Increased C_y.

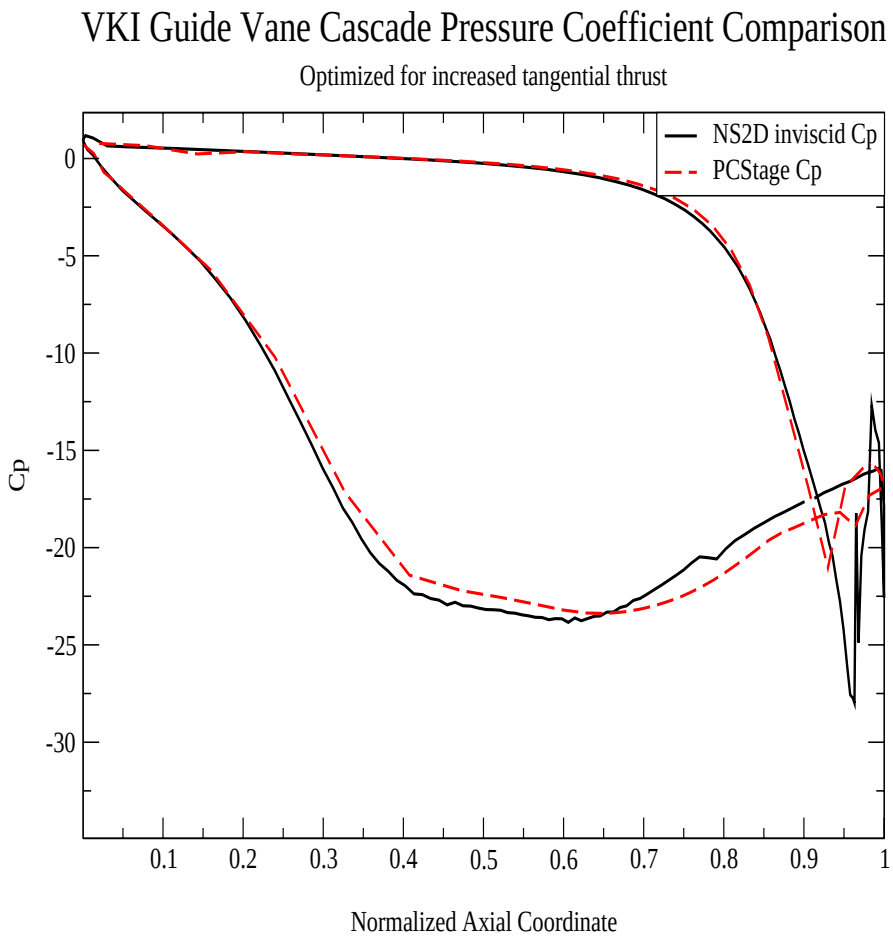

Figure 4.13: C_p Comparison for VKI with Increased C_y .

4.2.3 Decreased Drag

The PCStage results for a decreased absolute drag coefficient optimization are shown in Table 4.11. The values of C_x and C_y have been decreased slightly and $|C_d|$ has been greatly reduced.

Table 4.11: VKI PCStage Analysis (Decreased $|C_d|$)

Gen	C_x	C_y	C_m	C_l	$\|C_d\|$
0	14.00077	7.307141	1.645532	15.79246	0.1187053
20	12.96702	6.938470	1.366316	14.70666	0.01288605

Table 4.12: VKI Euler Analysis (Decreased $|C_d|$)

Gen	F_x (N)	F_y (N)	ζ (%)
0	608.8968	330.3311	2.026984
20	577.8099	329.7541	1.896473

The Euler results for this geometry are presented in Table 4.12 and show a reduction in F_x and ζ, and a very small reduction in F_y. The changes effected in these properties are not of the magnitude expected from the PCStage results. Further indication of the poor agreement in degree of change can be seen in Table 4.13.

Table 4.13: VKI Percentage Change in Selected Aerodynamic Coefficients (Decreased $|C_d|$)

PCStage	$C_x = -7.4\%$	$C_y = -5.04\%$	$\lvert C_d \rvert = -89.1\%$
NS2D	$F_x = -5.1\%$	$F_y = -0.17\%$	$\zeta = -6.4\%$

The optimized geometry plotted in Figure 4.14 shows a reduced curvature along both surfaces, with a rather drastic lowering of the pressure surface. The PCStage and Euler computed pressure coefficients for the redesigned geometry (Figure 4.15) show a maximum pressure drop (excepting the Euler calculated trailing edge spike) around the quarter chord distance along the suction surface. Unlike the previous cases the pressure quickly rises before leveling off at the trailing edge value.

The pressure coefficient comparison of Figure 4.15 indicates a discrepancy between the solution methods. The pressure minimum around the quarter length of the suction surface is predicted by both methods but the Euler solution shows a much larger decrease in pressure. The Euler method also exhibits a much larger pressure spike at the trailing edge. This is evidence of the linear panel methods error due to approximation. It is also evident from the plot that the surface resolution used by PCStage is much coarser than the grid used in the Euler calculation. An increased surface resolution may help to reduce the errors illustrated in this example.

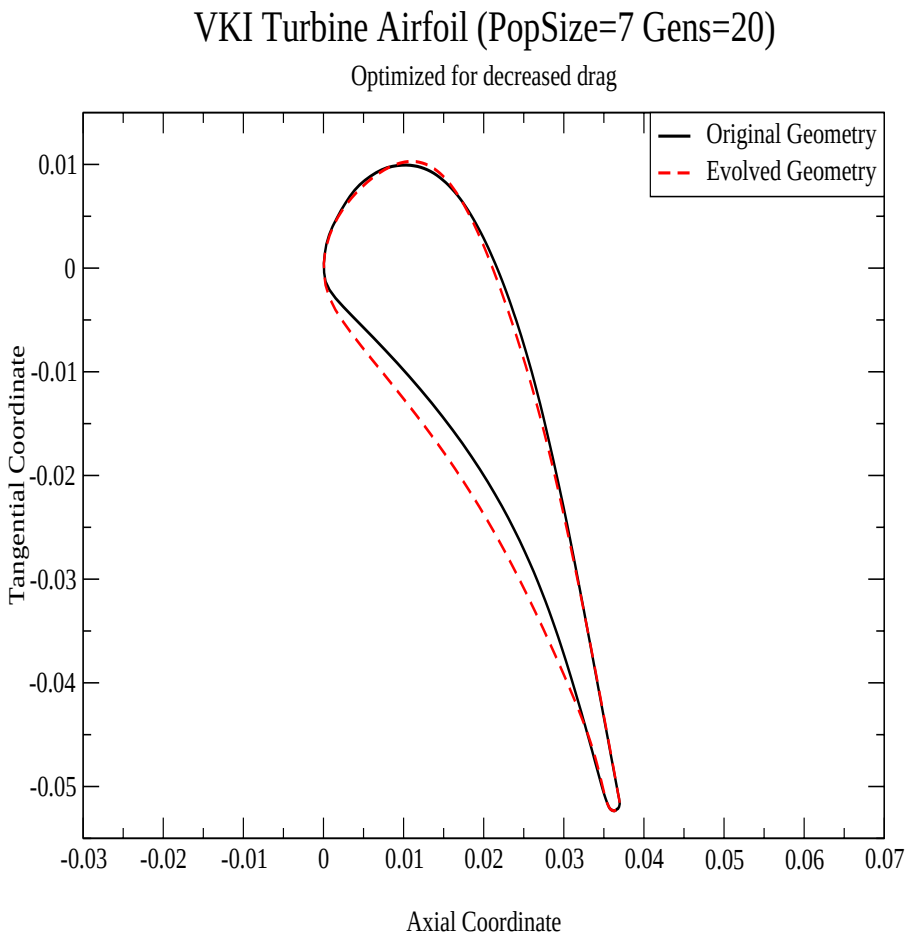

Figure 4.14: Geometry Comparison for VKI with Decreased $|C_d|$.

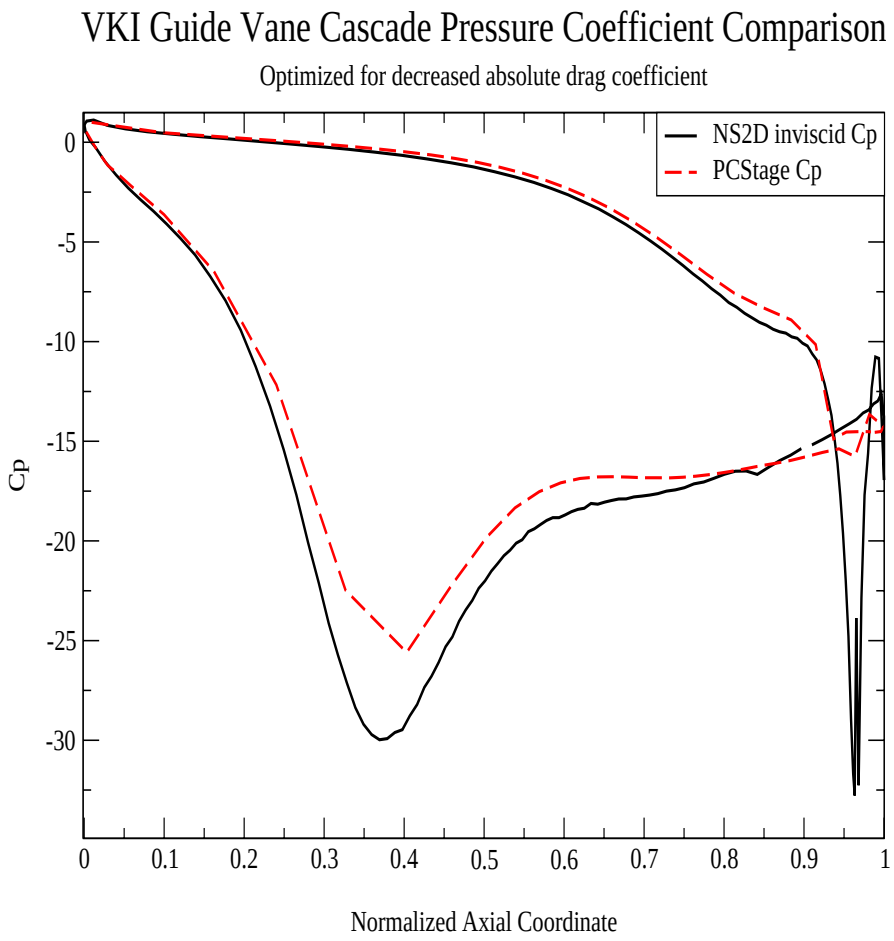

Figure 4.15: C_p Comparison for VKI with Decreased $|C_d|$.

4.2.4 Multiobjective Optimization

As in the DAT case, a multiobjective optimization was run for the VKI cascade that aimed to increase the combination of $C_x + C_y - |C_d|$. The results of the optimization listed in Table 4.14 show an increase in the values of C_x, C_y, and $|C_d|$. Despite the increase in $|C_d|$, the combination results in a total overall improvement.

Table 4.14: VKI PCStage Analysis (Increased Combination: $C_x + C_y - |C_d|$)

Gen	C_x	C_y	C_m	C_l	$\lvert C_d \rvert$
0	14.00077	7.307141	1.645532	15.79246	0.1187053
20	17.22413	8.151585	1.745002	19.05521	0.1335359
Combined Coefficients at Gen=0					21.89205
Combined Coefficients at Gen=20					25.24244

Table 4.15: VKI Euler Analysis (Increased Combination: $C_x + C_y - |C_d|$)

Gen	F_x (N)	F_y (N)	ζ (%)
0	608.8968	330.3311	2.026984
20	751.1648	366.4052	2.211776
Combined Coefficients at Gen=0			937.2009
Combined Coefficients at Gen=20			1115.358

The Euler computed coefficients of the evolved geometry for the multiobjective optimization are listed in Table 4.15. As in the previous cases, the changes predicted by PCStage are reflected in the Euler analysis.

The percentage change (Table 4.16) shows a very close agreement between the methods for axial and tangential thrust.

Table 4.16: VKI Percentage Change in Selected Aerodynamic Coefficients (Increased Combination: $C_x + C_y - |C_d|$)

PCStage	$C_x = 23\%$	$C_y = 11.55\%$	$\|C_d\| = 12.5\%$
NS2D	$F_x = 23.3\%$	$F_y = 10.9\%$	$\zeta = 9.11\%$
PCStage Combined Coefficient Change			15.3%
NS2D Combined Coefficient Change			19%

The evolved geometry presented in Figure 4.16 exhibits a rather large elevation along the first half of the suction surface. The corresponding pressure coefficient calculations of PCStage and the Euler equations are plotted in Figure 4.17. As expected, the elevated peak of the suction surface results in a significant pressure drop just behind the peak. The comparison shows a rather good agreement along the pressure surface with some deviation along portions of the suction surface.

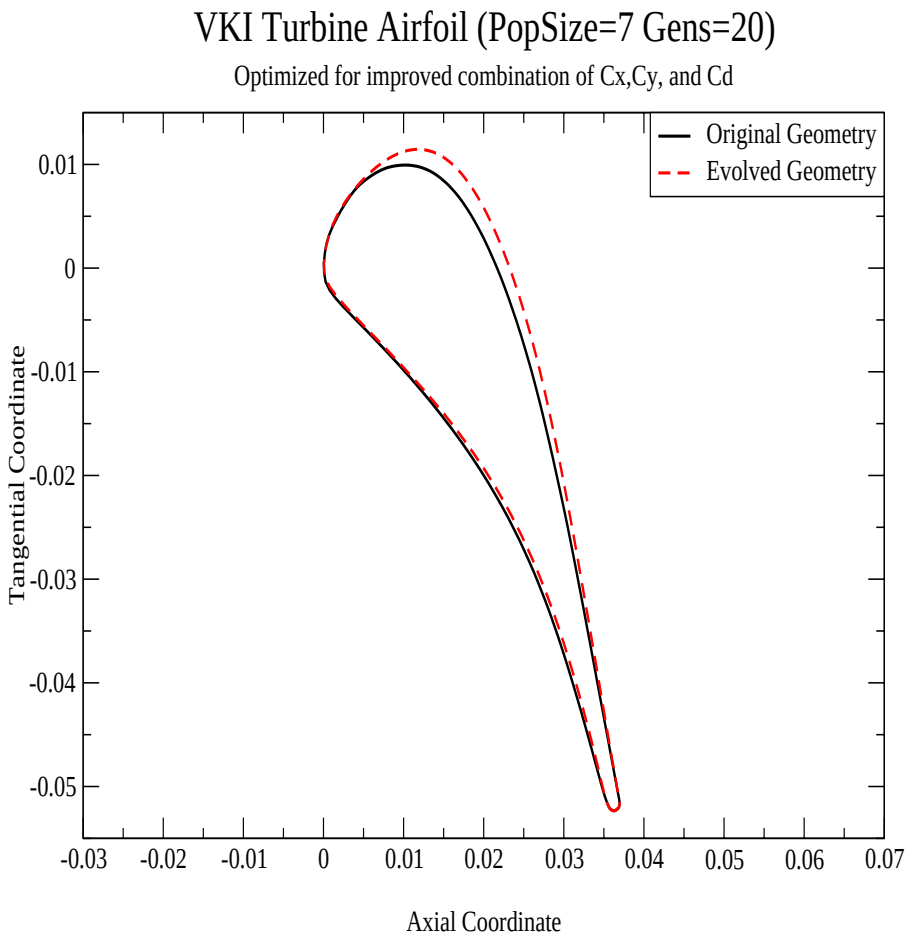

Figure 4.16: Geometry Comparison for VKI with Increased Combination of $C_x + C_y - |C_d|$.

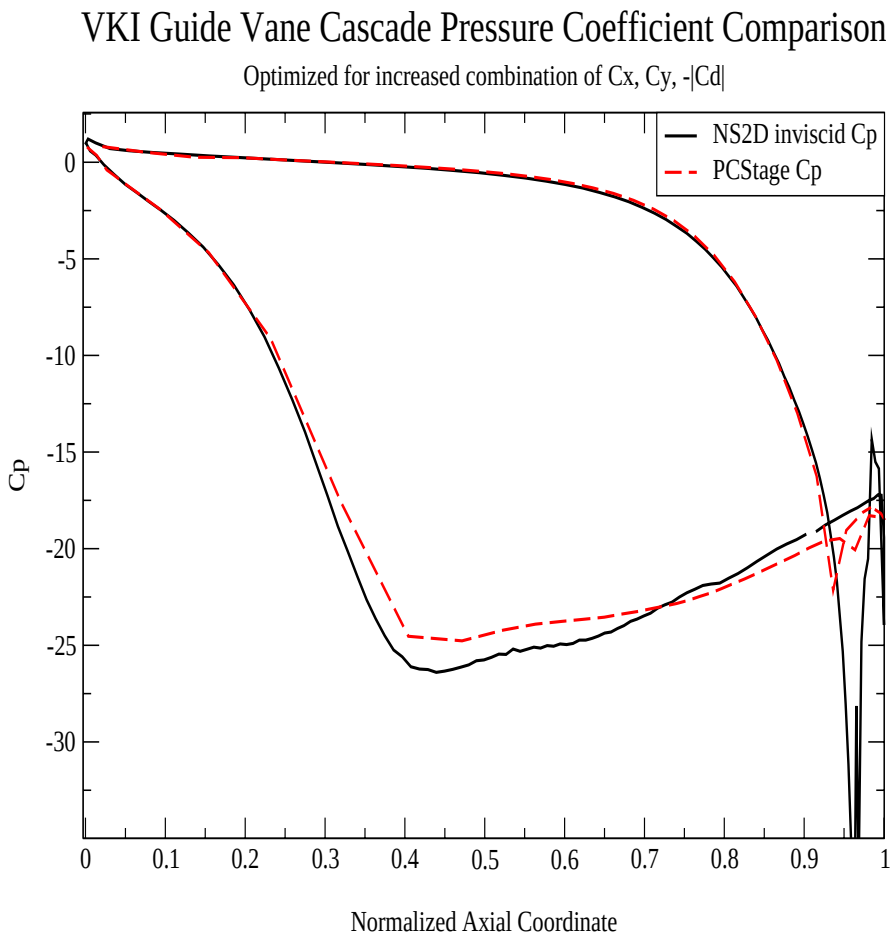

Figure 4.17: C_p Comparison for VKI with Increased Combination of $C_x + C_y - |C_d|$.

4.2.5 Summary of Results for VKI Guide Vane Cascade

The results from the VKI Guide Vane Cascade test case indicate three important pieces of information. First of all, the addition of greater constraints (see Section 4.2) on the optimization process appears to have been effective in producing more feasible designs. Secondly, the results of the multiobjective optimization show that the fitness function which was used, unevenly weighted the relative importance of each of the objectives. The third and most important piece of information gained from this case is that the simplified model of the linear panel method appears to provide information which is consistent with higher fidelity methods.

CHAPTER V

CONCLUSIONS

The results of this study clearly show that the PCStage linear panel method, used in conjunction with an evolutionary optimization method is effective in quickly producing turbomachinery designs with improved selected aerodynamic coefficients. The ability of the PCStage method to complete flow solutions approximately 60 times faster than an Euler method is purchased with an acceptable loss in fidelity. Also, the applicability is limited to designs with little or no viscous effects, limited flow separation and no shock conditions. Also, care must be taken in formulating adequate constraints to ensure feasability. The results produced by the pre-optimizer provide a significantly improved starting point for further analysis using higher fidelity methods.

5.1 Future Work and Suggested Improvements

There are many ways in which the accuracy and speed of this technique may be improved. Three major areas have been recognized as candidates for improvement. These areas are: the flow solver, surface representation methods, and the optimization technique. Some solutions which have been suggested in previous sections will be restated here.

Due to the inviscid linear panel method's inability to capture the viscous contribution to the problem, the flow solver may be coupled with a boundary layer code to improve the accuracy of the solution. This would require iteration between the flow solver and the boundary layer code to achieve a convergence in the boundary layer displacement thickness. Due to the fact that this would add considerable time to the process if it were done for every function evaluation, it could be performed periodically or only on the final solution.

Originally, the cubic spline surface parameterization performed within PCStage set the maximum number of panels to 90 per blade. The original data file for the VKI case uses 169 points to describe the guide vane. The grid used for Euler verification also uses 169 surface points to describe the guide vane. A closer agreement between methods should result if PCStage were altered to allow a larger number of surface points to be used.

Utilization of a single BSpline rather than two Bezier curves to represent a blade surface would be useful in eliminating possible discontinuous first derivatives (cuspidal shapes) and reduce the need for constraints at the leading edge. This would also allow the possibility of refining the geometries by altering the weights of the control points rather than their locations. A more flexible method should also be devised in order to alter the shape of the trailing edge without altering the flow angle. Currently this problem is handled by rigidly fixing the control points surrounding the trailing edge. An alternative would be to include the trailing edge angle in the fitness evaluation. Any geometries whose trailing edge flow angle deviates from

the desired flow angle would be penalized according to the magnitude of deviation and would therefore be replaced by more fit solutions.

In section 3.5.2, the advantage of utilizing a hybrid EA-gradient based method was illustrated. Due to the fact that the gradient based method may stall in local optima of a complex design space, an efficient optimization is suggested in which an EA optimization is run to produce a single global near-optimum. A gradient based optimization is subsequently performed on the near optimum to provide an exact globally fit optimum.

The method used in this study to obtain sensitivity derivatives for the gradient based optimization was a central finite difference method. This method requires two perturbed function evaluations to achieve a single derivative with a second order truncation error. Using the Complex Taylors Series Expansion method (CTSE), second order derivatives may be obtained in a single step, and without subtractive cancellation error. To utilize CTSE, the flow solver code must be converted to use complex numbers for selected internal calculations.

According to the data presented in Figure 3.10 an increase in the population size will result in an increase in the the overall CPU time required for an EA optimization. Despite this fact, an increase in population size would result in a more globally fit solution and may reduce completion time by increasing the degree of improvement for each iteration. These improvements can be achieved in the following manner:

1. First of all, a larger population size would allow more sophisticated methods of parent selection. The current method is to choose the best two members from the available pool and combine them. For each

mating, one of the parents has a possibility of being removed from the available pool for the rest of the current generations mating process. This method was used due to the limited number of parents available. It has the property of always selecting the best two members at least once, but can potentially produce an entire generation consisting of only these original parents and their offspring. This eventuality may cause the process to stall into a local optimum. With a larger population, greater diversity may be obtained by allowing a smaller segment to reproduce, leaving more of the less fit solutions to survive to the next generation. For example, the mating pool may consist of the best 20% of the population, which would then be combined randomly within this pool.

2. Secondly, a larger population size would produce a larger number of unfit members which could then be replaced by offspring. This would allow parent combination to produce two or more offspring per mating, thereby producing more varied combinations of desirable parental traits.

3. Thirdly, a large population size allows the use of many different parallelization schemes. One such scheme is the multiple population scheme [39]. In this method, the total population is divided into a number of distinct populations which evolve separately over a number of generations. These distinct populations are periodically interupted to perform a migration, which allows the transfer of the best members from each distinct population into the others. This

mechanism provides a much wider design space and thereby produces a solution which is much more globally fit.

Another parallelization scheme has been devised which more strictly determines parent selection and member culling [40, 41]. The population is envisioned as occupying a two-dimensional grid. Each member can only be combined with a member in an adjacent square. The offspring may replace the worst members adjacent to either parent. In this process, the best solutions propagate through the population like a ripple on a pond. The purpose of this method is to restrict the best solutions from quickly dominating the entire population. Similar to the multiple population scheme, this has the effect of producing a wider variety of solutions and therefore results in a more globally fit converged solution.

For multiple blade row geometries, a more precise fitness evaluation may be used rather than the average of the desired integrated property coefficient. In Section 4.1.1 it was suggested that a percentage change should be utilized as a more even handed evaluation method for multiple blade rows and multiple fitness criteria.

There are no doubt, many other possible solutions that have not been touched upon here, but suggestions have been made for all identified areas of possible improvement.

REFERENCES

[1] A. M. Josephy Jr. et al, *The American Heritage History of Flight*. New York, NY: American Heritage Publishing Company, Inc., 1962.

[2] Anderson, Tannehill, and Pletcher, *Computational Fluid Mechanics and Heat Transfer*. Washington DC: Taylor & Francis, second ed., 1997.

[3] F. Harlow and E. Welch, "Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluids with Free Surface," Tech. Rep. Vol. 8, p. 2182, Phys. Fluids, 1965.

[4] F. Harlow, D. Dickman, D. Harris, and R. Martin, "Two-Dimensional Hydrodynamic Calculations," Tech. Rep. Report LA-230, Los Alamos National Laboratory, 1959.

[5] G. N. Vanderplaats, *Numerical Optimization Techniques for Engineering Design*. Colorado Springs, CO: Vanderplaats Research and Development Inc., third ed., 1999.

[6] J. C. Newman III, W. K. Anderson, and D. L. Whitfield, "Multidisciplinary Sensitivity Derivatives Using Complex Variables," Tech. Rep. MSSU-EIRS-ERC-98-08, Mississippi State University, 1998.

[7] J. Holland, *Adaption in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.

[8] D. Fogel, "System Identification Through Simulated Evolution: A Machine Learning Approach to Modeling." Ginn Press, 1991.

[9] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolutionary Programs*. Springer Verlag, 1996.

[10] I. Wegener, "On the Expected Runtime and the Success Probability of Evolutionary Algorithms," in *Workshop on Graph-Theoretic Concepts in Computer Science*, pp. 1–10, 2000.

[11] C. Darwin, *The Origin of Species.* Oxford University Press, 1998. Reissue edition.

[12] R. Storn and K. Price, "Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces," tech. rep., International Computer Science Institute, 1995.

[13] H. B. Iridia, "Hybridize, Hybridize And Hybridize Again." URL: citeseer.nj.nec.com/361820.html.

[14] L. Shi and S. Olafsson, "A New Hybrid Genetic Algorithm," in *Late Breaking Papers at the Genetic Programming 1998 Conference* (J. R. Koza, ed.), (University of Wisconsin, Madison, Wisconsin, USA), Stanford University Bookstore, 22-25 1998.

[15] R. Salomon, "The Evolutionary-Gradient-Search Procedure," in *Genetic Programming 1998: Proceedings of the Third Annual Conference* (J. R. Koza, W. Banzhaf, K. Chellapilla, K. Deb, M. Dorigo, D. B. Fogel, M. H. Garzon, D. E. Goldberg, H. Iba, and R. Riolo, eds.), (University of Wisconsin, Madison, Wisconsin, USA), pp. 852–862, Morgan Kaufmann, 22-25 1998.

[16] R. L. Burden and J. D. Faires, *Numerical Analysis.* Pacific Grove, CA: Brooks/Cole, sixth ed., 1997.

[17] J. Gallier, *Curves and Surfaces in Geometric Modeling.* San Francisco, CA: Mogan Kaufmann Publishers, 2000.

[18] Z. Warsi, *Fluid Dynamics: Theoretical and Computational Approaches.* Boca Raton, FL.: CRC Press, second ed., 1998.

[19] F. M. White, *Fluid Dynamics.* McGraw Hill, fourth ed., 1999.

[20] H. Sytsma, B. Hewitt, and P. Rubbert, "A Comparison of Panel Methods for Subsonic Flow Computation," Tech. Rep. 241, NATO Advisory Group for Aerospace Research and Development, 7 Rue Ancelle 92200 Neuilly Sur Seine France, 1979.

[21] E. R. McFarland, "An Integral Equation Solution for Multistage Turbomachinery Design Calculations," Tech. Rep. 93-GT-41, ASME, 1993.

[22] J. Janus and J. C. Newman III, "Aerodynamic and Thermal Design Optimization for Turbine Airfoils," Tech. Rep. AIAA-2000-0840, AIAA, 2000.

[23] E. R. McFarland, "A Rapid Blade-to-Blade Solution for Use in Turbomachinery Design," *Journal of Engineering for Gas Turbines and Power*, vol. 106, pp. 376–382, 1984.

[24] E. R. McFarland, "Solution of Plane Cascade Flow Using Improved Surface Singularity Methods," *Journal of Engineering for Power*, vol. 104, pp. 668–674, 1982.

[25] J. Lewis, R. Delany, and E. Hall, "Numerical Prediction of Vane-Blade Aerodynamic Interaction," *Journal of Turbomachinery*, vol. 111, pp. 387–393, 1989.

[26] N. Kemp and W. Sears, "Aerodynamic Interference Between Moving Blade Rows," *Journal of Aeronautical Sciences*, vol. 20, no. 9, pp. 585–597, 1953.

[27] R. Parker, "Calculation of Flow Through Cascades of Blades Having Relative Motion and the Generation of Alternating Pressures and Forces Due to Interaction Effects," in *Proceedings of the Institute of Mechanical Engineering*, vol. 182, pp. 229–242, Institute of Mechanical Engineering, 1967. part 1.

[28] M. E. Mortenson, *Geometric Modeling*. New York, NY: John Wiley & Sons, 1985.

[29] D. Beasly, D. R. Bull, and R. R. Martin, "An Overview of Genetic Algorithms Part 1, Fundamentals," tech. rep., University of Wales College of Cardiff, University of Wales College of Cardiff, Cardiff, CF2 4YN, UK, 1993.

[30] D. Beasly, D. R. Bull, and R. R. Martin, "An Overview of Genetic Algorithms Part 2, Research Topics," tech. rep., University of Wales College of Cardiff, University of Wales College of Cardiff, Cardiff, CF2 4YN, UK, 1993.

[31] W. Spears, K. DeLong, T. Back, D. Fogel, and H. DeGaris, "An Overview of Evolutionary Computation," in *Proceedings of European Conference on Machine Learning*, pp. 442–459, 1993.

[32] H. Mühlenbein and D. Schlierkamp-Voosen, "Analysis of Selection, Mutation and Recombination in Genetic Algorithms." GMD Schloß Birlinghoven D-53754 Sankt Augustin, Germany, 1995.

[33] S. Luke and L. Spector, "A Revised Comparison of Crossover and Mutation in Genetic Programming," in *Proceedings of the Third Annual*

Genetic Programming Conference (GP98), pp. 208–213, San Fransisco: Morgan Kaufmann, 1998.

[34] E. Cantú-Paz, "A Survey of Parallel Genetic Algorithms," tech. rep., University of Illinois at Urbana-Champaign, 1998.

[35] S. G. Akl, *Parallel Computation, Models and Methods.* Upper Saddle River, NJ: Prentice Hall, 1997.

[36] C. A. C. Coello, "A Survey of Constraint Handling Techniques used with Evolutionary Algorithms." URL: citeseer.nj.nec.com/202945.html.

[37] R. Dring, H. Joslyn, and M. Blair, "The Effects of Inlet Turbulence and Rotor/Stator interactions on the Aerodynamics of Heat Transfer and of a Large Scale Rotating Turbine model,IV-Aerodynamic Data Tabulation," Tech. Rep. CR-179469, NASA, 1987.

[38] T. Arts, M. L. de Rouvroit, and A. Rutherford, "Aero-thermal Investigation of a Highly Loaded Transonic Linear Turbine Guide Vane Cascade," Tech. Rep. VKI-TN 174, von Karman Institute for Fluid Dynamics, Sept. 1990.

[39] W. J. Cohoon, S. Hedge and D. Richards, "Distributed Genetic Algorithms for the Floor Plan Design Problem," tech. rep., University of Virginia, 1988.

[40] S. Baluja, "A Massively Distributed Parallel Genetic Algorithm (mdpGA)," tech. rep., Carnegie Mellon University, 1992.

[41] W. P. E.D. Goodman and S. Lin, "Coarse-Grain Parallel Genetic Algorithms: Categorization and New Approach," in *Sixth IEEE Parallel and Distributed Processing*, pp. 28–37, 1994.