MULTIPLOID GENETIC ALGORITHMS FOR MULTI-OBJECTIVE TURBINE BLADE AERODYNAMIC OPTIMIZATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÖZHAN ÖKSÜZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
AEROSPACE ENGINEERING

DECEMBER 2007

Approval of the thesis:

MULTIPLOID GENETIC ALGORITHMS FOR MULTI-OBJECTIVE TURBINE BLADE AERODYNAMIC OPTIMIZATION

submitted by **ÖZHAN ÖKSÜZ** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Aerospace Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _______________
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. İsmail H. Tuncer _______________
Head of Department, **Aerospace Engineering**

Prof. Dr. İ. Sinan Akmandor
Supervisor, **Aerospace Engineering Dept., METU** _______________

Examining Committee Members:

Assoc. Prof. Dr. Sinan Eyi _______________
Aerospace Engineering Dept., METU

Prof. Dr. İ. Sinan Akmandor _______________
Aerospace Engineering Dept., METU

Prof. Dr. Ünver Kaynak _______________
Electric Electronic Engineering Dept.,
TOBB University of Economics and Technology

Assoc. Prof. Dr. Serkan Özgen _______________
Aerospace Engineering Dept., METU

Assoc. Prof. Dr. Oğuz Uzol _______________
Aerospace Engineering Dept., METU

Date: December 24, 2007

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Özhan Öksüz

Signature :

ABSTRACT

MULTIPLOID GENETIC ALGORITHMS FOR MULTI-OBJECTIVE TURBINE BLADE AERODYNAMIC OPTIMIZATION

Öksüz, Özhan

Ph.D., Department of Aerospace Engineering

Supervisor: Prof. Dr. İ. Sinan Akmandor

December 2007, 217 Pages

To decrease the computational cost of genetic algorithm optimizations, surrogate models are used during optimization. Online update of surrogate models and repeated exchange of surrogate models with exact model during genetic optimization converts static optimization problems to dynamic ones. However, genetic algorithms fail to converge to the global optimum in dynamic optimization problems. To address these problems, a multiploid genetic algorithm optimization method is proposed. Multi-fidelity surrogate models are assigned to corresponding levels of fitness values to sustain the static optimization problem. Low fidelity fitness values are used to decrease the computational cost. The exact/highest-fidelity model fitness value is used for converging to the global optimum. The algorithm is applied to single and multi-objective turbine blade aerodynamic optimization problems. The design objectives are selected as maximizing the adiabatic efficiency and torque so as to reduce the weight, size and the cost of the gas turbine engine. A 3-D steady Reynolds-Averaged Navier-Stokes solver is coupled with an automated unstructured grid generation tool. The solver is validated by using two well known test cases. Blade geometry is modelled by 37 design variables. Fine and coarse grid solutions are respected as high and low fidelity surrogate models, respectively. One of the test cases is selected as the baseline and is modified in the design process. The effects of input parameters on the performance of the multiploid genetic algorithm are studied. It is demonstrated that the proposed algorithm accelerates the optimization cycle while providing convergence to the global optimum for single and multi-objective problems.

Keywords: Genetic Algorithms, Diploid, Multiploid, Surrogate models, Low-fidelity, Turbine Blade Design, Aerodynamic Optimization, Multi-objective.

ÖZ

TÜRBİN KANATÇIKLARININ ÇOK AMAÇLI OPTİMİZASYONU İÇİN ÇOK KROMOZOMLU GENETİK ALGORİTMALAR

Öksüz, Özhan
Doktora, Havacılık ve Uzay Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. İ. Sinan Akmandor

Aralık 2007, 217 Sayfa

Genetik algoritmaların hesaplama maliyetlerini düşürmek için, indirgenmiş modeller kullanılmaktadır. Bu modellerin optimizasyon sırasında çevrim içi güncellenmesi ve kesin modeller ile tekrarlanan değiş tokuşlar yapılması, statik problemi dinamik hale getirmektedir. Ancak genetik algoritmalar dinamik optimizasyon problemlerinde global optimum'a yakınsayamazlar. Bu sorunları çözmek için, yeni çok kromozomlu bir genetik algoritma önerilmektedir. Optimizasyon problemini statik halde korumak için, değişik doğruluktaki indirgenmiş modeller, farklı seviyelerde uygunluk değerlerine eşlenmektedir. İndirgenmiş modellerin uygunluk değerleri hesaplama maliyetini düşürmek için kullanılırken, kesin model ise global optimum'a yakınsamak için kullanılır. Algoritma tek ve çok amaçlı türbin kanatçık aerodinamik optimizasyonu problemlerine uygulanmıştır. Tasarım amaçları olarak adiyabatik verim ve torkun arttırılması olarak seçilmiştir. 3 boyutlu Reynolds Averaged Navier Stokes çözücüsü otomatik grid çözücüsü ile akuple edilmiştir. Çözücünün doruluğu iyi bilinen iki test problemi ile sabitleştirilmiştir. Kanatçık geometrisi 36 tasarım parametresi ile modellenmiş, ayrıca bir kademedeki pale sayısı da buna eklenmiştir. Kaba ve ince grid çözümleri, sırasıyla, kesin ve indirgenmiş model olarak kullanılmıştır. Test problemlerinden biri baz alınmış ve tasarım prosesi ile modifiye edilmiştir. Giriş parametrelerinin çok kromozomlu genetik algoritmanın performansı üzerine etkisi çalışılmıştır. Önerilen çok kromozomlu genetik algoritmanın tek ve çok amaçlı optimizasyon çevrimini hızlandırdığı, aynı zamanda global optimum'a yakınsadığı gösterilmiştir.

Anahtar Kelimeler: Genetic Algoritmalar, Diploid, Çok kromozomlu, Basitleştirilmiş Model, Türbin Kanatçık Tasarımı, Aerodinamik Optimizasyon, Çok-amaçlı.

ACKNOWLEDGMENTS

I would like to express my optimum sincere appreciation to Prof. Dr. İ. Sinan Akmandor for his invaluable guidance and insight throughout this research. No doubt, he is the best of my teachers.

I offer sincere thanks to my darling Burçin for her love, for her continuous support and endless patience.

My father and mother devoted their valuable life to my education. I will never underestimate their help, and I cannot compensate this in any way.

Many thanks to the entire staff of the Aerospace Engineering Department for their invisible inevitable presence.

The financial support provided by Tubitak- BAYG is appreciated.

Finally, I express my sincere to TuAF ALC for their support and encourage of finishing this thesis.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURE

LIST OF SYMBOLS

SYMBOLS

A	Airfoil area
$\vec{A}$	Face Area Vector at Node
A_c	Cell Volume per Unit Depth
$\vec{A}_n$	Normal Area Vector
$\vec{B}$	Streamline Cell Face Vector
c	Chord
c	log-layer constant
CL	Confidence Level
C_{ax}	Axial Chord
C_D	Dissipation Coefficient
C_D	Drag Coefficient
C_f	Skin Friction Coefficient
C_L	Lift Coefficient
CP_s	Local static pressure coefficient
CP_0	Local total pressure coefficient
$d_k(\mathbf{x})$	Relative distance of an individual from elite member k
D	Drag
D_M	Radius normalized distance along meridional curve
D_s	Diffusion parameter
e	Elite member vector
$\vec{e}$	Node Vector

f(**x**) Objective function

f_μ Proportionality constant

$\vec{f}_b$ Body Force Vector

f_i, g_i Geometric Functions

F Tangential force

F(x) Objective function vector

g(**x**) Constraints function

h Static Enthalpy

h_t Total Enthalpy

H Schema representation

H, H^*, H^{**} Shape Parameters

H_k, H_k^* Kinematic Shape Parameters

I_{max} Airfoil maximum moment of inertia

I_{min} Airfoil minimum moment of inertia

I Rothalpy

l_t Turbulence length scale

L Lift

L Layer Number

m Mass

$\dot{m}$ Mass Flow Rate

M Mach number

N Number of blades in a stage

n Number of bits

$\vec{n}$ Unit Face Normal Vector

Δn Gap between Streamline and Wall

$\vec{N}$ Average Cell Normal Vector

p Static pressure

p_{cross} Crossover probability

p_k Latent potential value of an elite member

p_t Absolute Total Pressure

P Power

P Individual selected as a Parent

P_c Pressure Correction Term

Q Flow Coefficient

q number of eliminated solutions from EMS

$\vec{q}$ Velocity Vector

$\vec{r}$ Position Vector

R Universal Gas Constant

$R_{e\theta}$ Reynols Number Based on the Momentum Thickness

R_s Source Term due to Rotation in Streamwise Direction

R_n Source Term due to Rotation in Normal Direction

s Pitch

$\hat{s}$ Local Unit Flow Vector

S Design space

$\vec{S}$ Average Streamwise Vector

ΔS Design variable space length

T_t Absolute Total Temperature

T Temperature

Tu Turbulence Intensity

t Time

t Distance between camber line point and surface point

t Generation number in GA / MOGA

t_r Thickness ratio

t_{LE} Leading Edge Thickness

t_{TE} Trailing Edge Thickness

u Velocity in ξ-Direction

u^+ near wall velocity

u_τ friction velocity

U	Tangential Blade Velocity
U_t	Turbulent velocity scale
U_T	velocity tangent to the wall at a distance of Δy from the wall
v	Velocity in η-Direction
V_D	Fluid domain volume
$\vec{V}$	Absolute Velocity Vector
$\mathbf{x}$	Design variable vectors
X	Axial distance
x_n	Design variables
x, y	Cartesian Coordinates
Δy	distance from the wall
y^+	dimensionless distance from the wall
y_R	equivalent sand grain roughness height
W_{LE}	Leading Edge Wedge Angle
W_{TE}	Trailing Edge Wedge Angle
α	Angle of Attack
β	Flow Angle
β_{LE}	Blade Layer Leading Edge Inlet Metal Angle
β_{TE}	Blade Layer Trailing Edge Exit Metal Angle
δn	Grid Node Displacement Change
$\delta \rho$	Density Change
σ	Solidity
δ^*	Displacement Thickness
ϕ	Streamline Curvature Angle
γ	Specific Heat Ratio
η	Blade Efficiency
φ	Tangential Direction
μ	Dynamic Viscosity

μ_t Turbulent Viscosity

Π Pressure on the Streamline

Θ Rotation around Z axis (circumferential rotation)

θ Momentum Thickness

ρ Density

T Torque

τ Total shearing plus Reynolds stress

τ_ω Wall Shear Stress

Ω Angular velocity

$\vec{\Omega}$ Rotational Velocity Vector

ξ Stagger Angle

ξ, η Local Coordinates Along and Normal to Boundary Layer

ψ_z Zweifel loading coefficient

κ von Karman constant

SUBSCRIPTS

1 Quantity at Left Cell Face or Inlet

2 Quantity at Right Cell Face or Exit

a Average Value

b at the 1/3 of streamwise distance; Leading Edge Cut of the blade

c at the 2/3 of streamwise distance; Trailing Edge Cut of the blade

e Quantity at Edge of the Boundary Layer

e Exit

i Streamwise Station Index

j Streamtube Index

le Leading edge

m Mean

m	Meridional Component
n	Normal
max	Maximum value
min	Minimum value
opt	Optimum value
PS	Pressure side
SS	Suction Side
r	Radius
r	Radial Component
s	Suction side
te	Trailing Edge
t	Stagnation Quantity
θ	Tangential Component
x	x (axial) component
y	y component
φ	Tangential Component

SUPERSCRIPTS

+	Upper Streamline
-	Lower Streamline
^{T}	Transpose
'	calculated at the node

ABBREVIATIONS

2D	Two-dimensional
3D	Three-dimensional

ANN	Artificial Neural Networks
B2B	Blade to Blade Plane
CFD	Computational Fluid Dynamics
CPU	Central Processing Unit
EMS	Elite Members Set
GA	Genetic Algorithm
GAXL	Multiploid Genetic Algorithm
GAs	Genetic Algorithms
HPT	High Pressure Turbine
MOGA	Multi-objective Genetic Algorithm
MOGAXL	Multiploid Multi-objective Genetic Algorithm
RANS	Reynolds Averaged Navier-Stokes equations

CHAPTER 1

INTRODUCTION

The design of today's modern high performance turbo engine components requires fast but sophisticated design tools. The flow in a turbomachine is extremely complex because of the presence of blade boundary layers, interaction of rotating and non-rotating blades with the upstage wakes, development of secondary flow vortices and overall periodic but still inherently unsteady flow. Since the conventional experimental design methods are very expensive and time consuming, investigators started to make use of low-cost numerical methods for the design of internal aerodynamic blade shapes.

The numerical simulation methods of turbine blade aerothermodynamics have improved extensively over the last 10 years. In spite of the limitation due to modeling approximations such as turbulence and transition modeling, and heat transfer predictions, these methods are now capable of analyzing the performance of turbine blades with a flow accuracy which is acceptable for most engineering purposes. Computational fluid dynamics has also matured to the point where it is widely used as a key tool for aerodynamic design. Moreover, the CPU speeds of latest personal computers are extremely fast when compared to those of 10 years ago. Despite these advantageous progresses in science and technology, most designers still adopt a 'trial and error' approach for the final design process; analyzing the current blade shape, modifying it in function of the computational results or the experimental data, and abiding by empirical rules or to their own experience.

In this study, it is shown that intelligent optimization techniques such as genetic algorithms (GA) coupled with accurate objective function solvers (CFD) running on parallel fast computing computers, a fast, automatic, and robust turbine blade aerodynamic optimization method has been developed. This algorithm can prevent human intervention from becoming a bottleneck, and can preserve design engineers' experience.

1.1 Overview of Design Methods

Design methods can be categorized as inverse design methods and optimization methods. In the first approach, the inverse algorithm calculates the corresponding blade geometry once the surface pressure or velocity distribution is specified. On the other hand, the second category couples a conventional flow solver with an optimization routine to modify the parameterized blade geometry iteratively with the aim of minimizing some cost function such as difference between calculated and required blade loading.

1.1.1 Inverse Methods

Inverse methods based on the analytical solutions are called potential inverse methods. The analytical solution of the given surface pressure distribution directly defines the blade geometry. Some analytical examples of such approach is briefly summarized below: A conformal mapping method was developed by Lighthill [1], a stream function method was developed by Stanitz [2], and using method of characteristics Dulikravitch [3] developed a transonic cascade inverse design method. All of these methods are limited to 2D potential and irrotational flow and are no longer used in real design problems.

Iterative inverse methods are based on the modification of blade geometry iteratively until the desired pressure distribution is reached. Hawthorne [4] developed a method based on circulation imposed on blade walls instead on velocity distribution for 2D axial slender geometries with incompressible flows. Borges [5] extended the method to radial machines and Zangeneh [6] included viscous and compressible effects and blockage effect due to blade thickness. Dang [7] used a 3D Euler flow solver, and applied the method for transonic flow regimes. Due to inviscid flow, blade forces are assumed to be proportional to mean swirl distribution.

Most of the recent iterative inverse methods are based on Euler solvers. This subgroup can be divided into two sections, methods using different set of equations for flow field solution and geometry modification, and methods using the same set of equations for both flow solution and geometry modification. Leonard [8] and Bartelheimer [9] used the former method, and Meauze [10] and Demeulenaere [11] used the latter method. In the second group, blade geometry is modified by using either impermeable or permeable blade walls and planning mesh movements accordingly.

These methods are currently still being developed and applied to 2D and 3D problems and can be very efficient in terms of computational time. Moreover, there is no need to define a complex objective function. However, objective functions of these methods are mainly restricted to an imposed pressure distribution. The real performance measures such as efficiency, torque, outlet flow angles and loss coefficient cannot be imposed. Therefore, the designer needs to convert the performance requirements to pressure distribution. Most of the time, pre-defined pressure distributions do not meet the given performance criteria. Therefore, designer has to redefine the pressure distribution until solution converges to the design criteria. Moreover, there is an existence problem, which means one may not always be able to converge to the target pressure distribution. These main problems increase the design cycle time dramatically.

1.1.2 Optimization Methods

Numerical optimization methods aim to shorten and simplify the iterative process of inverse methods, while significantly improving the design output. All optimization problems contain three components:

- Objectives: describing what one hopes to achieve through the optimization process. In this thesis, the objective functions are set to be the blade torque and efficiency, since these are the key design parameters of a stage.
- Optimization parameters: describe how the system is to be adjusted in order to best meet the objectives. Parameters that determine the shape of the boundary are an example. In this study, the optimization

parameters are the blade shape design parameters and the number of blades in a stage which determines the pitch distance.

- Constraints: guide the optimization through states that must be satisfied. In our case, these are the geometric limitations, aerodynamic constraints and mechanical constraints.

Optimization techniques can be classified in three categories: local, global, and other methods. Local methods are gradient-based algorithms, which only search one part of the design space and stop after finding a local optimum. Adjoint, single or multi-grid preconditioners, alternating direction implicit methods are all local methods [12]. Global methods are stochastic methods that take into consideration the entire design space and only require objective function values, not the derivatives. Genetic algorithms, simulated annealing, random search methods are all considered as global methods. They also have the advantage of operating on discontinuous design spaces. Other optimization algorithms that do not fall entirely within either of these two categories are one-shot or inverse methods [13]. In this thesis, only global methods are discussed.

Random search is the simplest approach to minimize a function. In this context, a large number of candidates are selected randomly throughout the design space and objective function of these candidates is evaluated and the minima or the maxima of this set is called the optimum [14]. A slightly modified version of this method is called random walk and includes a search direction [15]. Regarding the number of function evaluations, random search or walk method is the most inefficient global optimization method, and therefore, cannot be applied on real industrial problems involving CFD analyses.

Simulated annealing ideas are derived from the annealing of solid bodies [16]. At a given temperature, the state of the system varies randomly. If a state results in a lower energy level, it is immediately accepted (for a minimization problem). If however a higher energy state results from the variation, it is only accepted with an *acceptance probability* defined as the Boltzmann Distribution of the temperature. Aly [17] applied simulated annealing to airfoil aerodynamic shape design and Tekinalp [18] proposed a new multi-objective simulated annealing algorithm recently. In the simulated annealing optimization method, the cost function replaces the energy of the system, and the optimization variables represent the atoms. This idea was first used by Kirkpatrick [19] to solve discrete combinatorial optimization problems.

Genetic Algorithms are search algorithms that mimic the behavior of natural selection to find the global optimum point in a given design space [20]. They operate on a population of potential solutions applying the principle of survival of the fittest to produce better and better approximations to a solution. At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the design space and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation. GAs are becoming more and more widely used in mechanical and aerodynamic problems, including preliminary design of turbines [21], aerodynamic optimization using CFD [22], [23], [24], [25], [26], optimization of target pressure distribution for inverse design methods [27], [28], multi-objective aerodynamic shape optimization [29], and multidisciplinary optimization of wings [30].

1.2 Research Goals

Although genetic algorithms are able to converge to the global optimum, they consume considerable time to converge for a 3D aerodynamic design problem which requires expensive high-fidelity RANS solver. This drawback of genetic algorithms was avoided by using faster but less accurate low-fidelity 2D solvers such as an Euler [31] or Euler/boundary layer coupled flow solvers [26] in the past. On the other hand, researchers used to implement parallel processing techniques to decrease the convergence time. However, even implementing parallel processing, one needs to decrease the computational time of the genetic optimization technique further.

The computational power of computers converged to a degree which investments for further improvements are really infeasible. While a CPU speed could be doubled in each year a decade ago, for the last few years, CPU speeds did not increase remarkably. On the contrary, new developed sophisticated commercial CFD analysis tools demand more and more computational power during the solution of complex problems.

Furthermore, designers tend to increase the complexity of the problem in order to achieve satisfying results and tackle powerful CFD software. For example, full 3D calculations, even multi-stage turbo component analysis need very fine meshes. Even simulation of the full characteristics of the engine and multiple operating point performance analysis are common nowadays. They even tend to try unsteady computations for this kind of analyses. This high-fidelity solver demand already consumes the computational power supplied by the newest technologies.

Therefore, a robust, fast, and accurate global optimization technique needs to be employed while preserving the accuracy of the optimization. Consequently, in this thesis, a new multiploid genetic optimization method based on multi-fidelity solutions is presented and applied for single and multi objective turbine blade optimization problem.

1.3 A New Optimization Method

Today, advanced CFD software is routinely used in the design process. These solvers are capable of analyzing 3D, viscous, transonic and turbulent flows. Analyzing accuracy of these codes is so powerful that, design engineers reduced the number of experimental studies and increased the CFD calculations. Consequently, in this method, a powerful commercial CFD code is used for analyzing the performance of the blade.

The increase in the number of CFD calculations and the complexity of the geometries brings the fact that, the design engineer needs to handle hundreds of design parameters, and ten thousands of CFD solutions for different geometries at different boundary conditions. Therefore, it is very difficult to handle this massive information and decide which way to go. One way to solve this problem is to define an objective function which translates the design engineer's decision criteria into a mathematical formulation that computer can handle. Consequently, knowing that design engineer may have more than one goal, a set of objective functions are used for this new multi objective optimization method.

As stated in the research goals of this thesis, the optimization algorithm needs to be robust, fast, and accurate. A robust algorithm prevents the human interaction become a bottleneck during the optimization cycle, and allows the solution to converge to the global optimum. A fast optimization method is able to use the available high-fidelity resources such as a 3D RANS solver. An accurate optimization algorithm is one which is not fooled by some low-fidelity solver solutions such as 2D results or surrogate models such as ANN, and which is able to preserve human expertise. Consequently, the new method uses a modified multi-chromosome genetic algorithm, which is called "multiploid", which is able to interpret separately the high-fidelity, high quality, very expensive information such as a fine mesh solution of the blade, and the low-fidelity, low quality inexpensive information such as a coarse mesh blade solution. The algorithm does not degrade the quality of the optimization result by mixing the information of multi-fidelity solutions.

1.4 Thesis Outline

In the present study, a multiploid genetic algorithm method is proposed for computationally intensive multiple objective problems and it is applied for the multi objective aerodynamic optimization of the turbine blades.

Chapter 2 gives a brief discussion on high pressure turbine rotor blade aerodynamic design. Turbine blade aerodynamic design concepts are presented followed by the description of the three-dimensional blade profile design method used in this study. For the flow field solution, automatic mesh generator and three-dimensional RANS solver is discussed with application to test cases. The flow field parallel solution properties are described. Finally, the optimization problem of this thesis is defined.

Chapter 3 is devoted to single objective optimization problems, starting with the definition of maximizing torque and the efficiency of a rotor stage. Simple Haploid Genetic Algorithm (GA) is described with evolutionary operators. The chapter is concluded with the introduction of the theory of Multiploid Genetic Algorithms (GAXL) and a discussion of understanding how they work.

Chapter 4 deals with multi-objective optimization problem. The multi-objective optimization problem is formulated at the beginning. With the definition of dominance and Pareto-optimality, elite preserving haploid multi-objective genetic algorithm (MOGA) is described which implements a distance based fitness assignment. Test case solutions of MOGA are provided with two different multi-objective algorithms existing in the literature. Finally, multiploid multi-objective genetic algorithm (MOGAXL) is defined with multi-fidelity handling fitness assignment, selection and crossover operators.

Chapter 5 includes the results of the single and multi objective optimization problems. In the first part, the two single objective problems are solved with GA and GAXL and the results are compared in terms of global optimum values found and the computational cost of both algorithms to converge the global optimum. In the second part, multi-objective optimization problem is solved by MOGA and MOGAXL. The global Pareto-optimum frontiers of both methods are compared with the computational costs incurred. Finally, the flow field properties of the optimized blade shapes are discussed briefly.

This study is concluded with discussions and recommendations for future work in Chapter 6.

CHAPTER 2

AXIAL TURBINE ROTOR BLADE AERODYNAMIC DESIGN

Traditional aerodynamic design of turbine blades usually assumes two distinct design stages. One dimensional flow analysis and blade-to-blade detailed design. However, this approach assumes the flow is two dimensional and cannot take the three dimensional secondary flows into account. For that reason, today, experienced design engineers make use of three dimensional RANS solvers, and by trial and error approach, try to converge to an acceptable design.

Since the efficiency of the gas turbines is getting more and more attention, design engineers begin implementing the popular RANS solvers more often, from 20 to 50 RANS computations [32]. By increasing efficiency, torque, and pressure ratio, it is possible to reduce the number of blades per stage needs but this definitely needs more and more computational results to be exploited. Although this trend increased the efficiency of a stage by a few points, a definite need for shortening the design cycle and increasing the efficiencies promptly requires an optimization tool to be applied. GE Aircraft Engines, one of the largest gas turbine manufacturers, has already applied optimization tools for preliminary design of axial compressors and turbines [33], [34], and for designing airfoil layers [35], [36], and for optimizing three dimensional blade profiles by maximizing efficiency while meeting mechanical requirements [37].

2.1 Turbine Blade Aerothermodynamics

The cross section of a high pressure turbine rotor blade with disk is shown in Figure 2.1. The blade is mounted to disk at its hub. The disk is located on the gas turbine shaft. High temperature and high pressure flow originally coming from the combustor rotates the turbine blades.

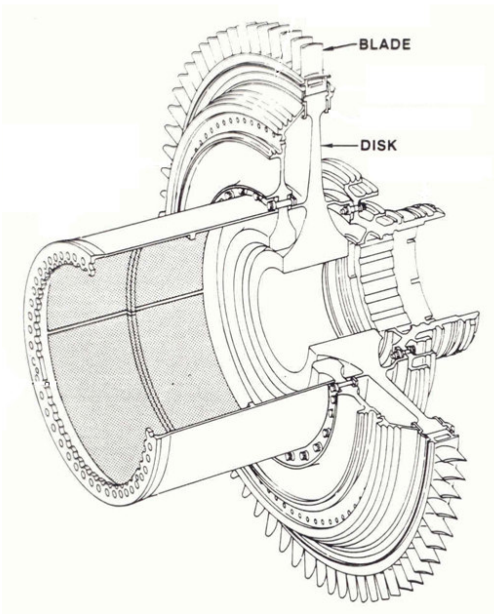


Figure 2.1 High Pressure Turbine Rotor cross-section schematic on radial-axial plane

2.1.1 Coordinate System

The coordinate system of a turbine stage consists of three directions as shown in Figure 2.2:

- *Axial direction*: parallel to the axis of rotation
- *Radial direction*: radial through the axis of rotation
- *Tangential direction*: tangent to the rotating stage.

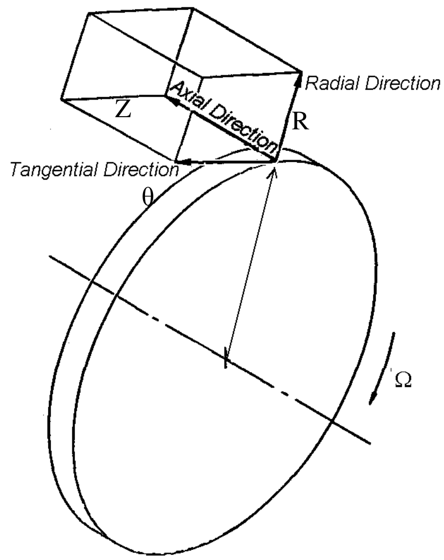


Figure 2.2 Coordinate System

Analysis of flow in the meridional (*radial-axial)* plane depicts the circumferentially averaged (blade-to-blade average) radial and axial variation of the desired flow parameters. For many types of calculations, blade-to-blade variation of parameter values can be ignored and only average values are used. Such a calculation is called *axisymmetric or through flow analysis* [38].

Calculations made in the blade-to-blade (*axial-tangential)* or *radial-tangential* planes are usually at some constant value (rather than for average conditions) of the third coordinate. Velocity diagrams, as well as blade-to-blade velocity variation calculations are usually made in these planes. When flow is predominantly radial or when the flow is axially symmetric, such as at the inlet to a radial-flow turbine, the radial-tangential plane is used. When flow is predominantly axial, such as in an axial-flow turbine, the axial-tangential plane is used. In this thesis, the flow is predominately axial therefore, the latter is used.

2.1.2 Meridional and Blade-to-Blade Planes

The complex 3-dimensional geometry of a blade is broken into two 2-dimensional planes, meridional and blade-to-blade planes. The *Meridional plane* is used to define the blade in radial vs. axial space. From this definition, the streamlines are generated which are required for the *Blade-to-Blade (B2B) plane*. The cross section of the succeeding blade rows of rotor and stator at a given span percentage creates a turbine blade *layer* in the B2B plane which defines the blade layer in tangential vs. axial space. The basic turbine layer terminology in B2B plane is shown in Figure 2.3.

A three dimensional turbine blade is represented by layers at various spanwise locations in meridional plane. The data from these planes is used to create the 3-dimensional blade geometry, using one blade's layer at a time.

In the blade to blade plane, flow enters the layer through the leading edge of the blade and leaves it through the trailing edge. The spacing between the blades is called as *pitch*. Blade inlet and exit angles as well as flow inlet and exit angles are also shown in Figure 2.3.

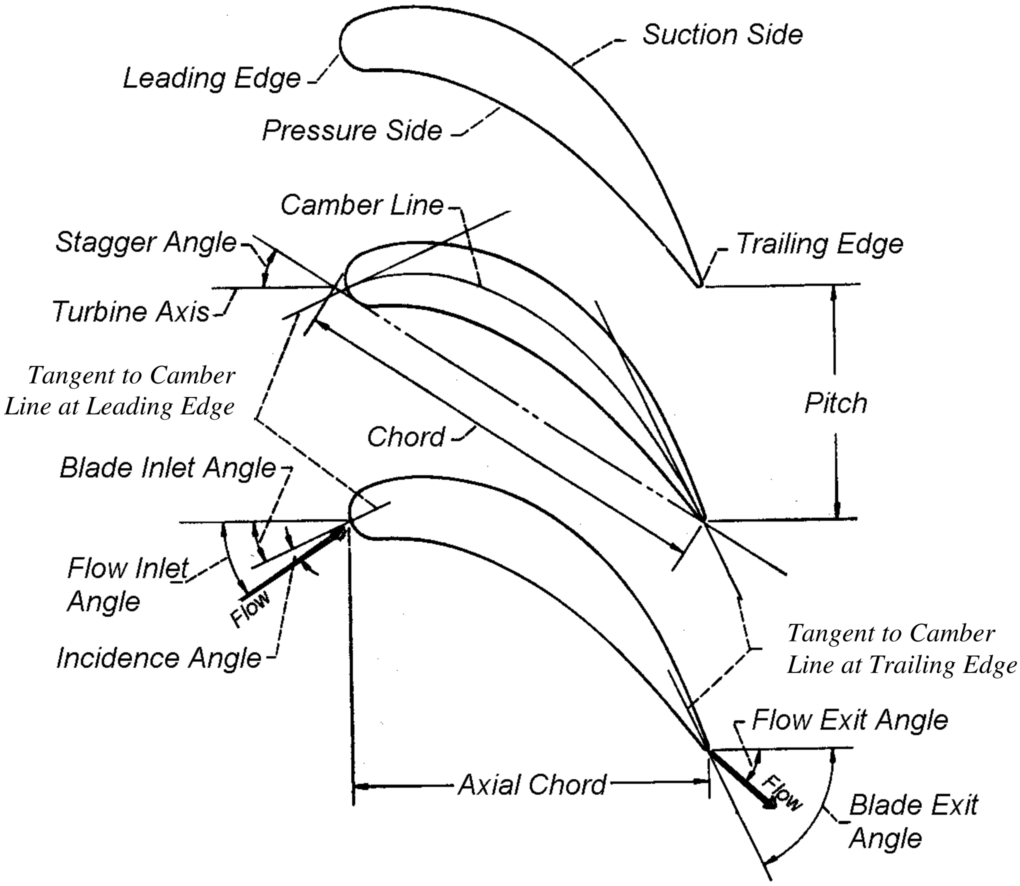


Figure 2.3 Turbine blade layer terminologies in B2B plane

2.1.3 Velocity Vectors and Diagrams

Flow kinematics and related energy transfer between the blade and the fluid are the most important topics of turbine aerodynamics. For that analysis, velocity vector diagrams are used. The inlet and exit flow characteristics are given as inputs to the system. For convenience, absolute velocity vectors for inlet ($\vec{V}_1$) and exit ($\vec{V}_2$) flow are shown for a rotating disk in Figure 2.4.

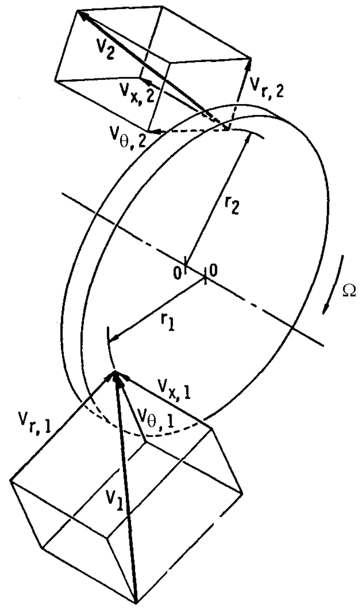


Figure 2.4 Velocity components for a generalized rotor [39]

According to Figure 2.4, blade inlet velocity is the vector sum of axial velocity $V_{X,1}$ and tangential velocity $V_{\theta,1}$. Similarly, blade exit velocity is found by vector summation of the axial velocity $V_{X,2}$ and tangential velocity $V_{\theta,2}$.

2.1.4 Blade Loading, Torque, and Efficiency

It is the change in tangential momentum of the fluid that results in the transfer of energy from the fluid to the rotor. As the fluid flows through the cascade between each pair of blades, through a curved passage, a centrifugal force acts on it in the direction of the pressure (concave) surface. To counterbalance, a pressure force is established to turn the fluid through its curved path. The pressure force is directed normal to the flow and toward the suction (convex) surface. Thus, the pressure in the passage is highest at the pressure surface and lowest at the suction surface. The resulting difference of the static pressure on the blade surfaces is called as the blade loading. The blade force acting in the tangential direction, *tangential blade force*.

The tangential blade force F on a blade section can be expressed as:

$$F = L\cos\alpha_m + D\sin\alpha_m \tag{2.1}$$

$$F = L\cos\alpha_m\left(1 + \frac{C_D}{C_L}\tan\alpha_m\right) \tag{2.2}$$

where L is lift, D is drag, and C_D and C_L are drag and lift coefficients respectively. The mean flow angle α_m is defined by:

$$\tan\alpha_m = \tfrac{1}{2}(\tan\alpha_1 + \tan\alpha_2) \tag{2.3}$$

Turbine rotor stage torque is used as one of the two objective functions in this thesis and calculated as:

$$\mathrm{T} = \sum_{BladeSurface} Np\vec{A}\cdot\left(\vec{n}_y\cdot\vec{e}_x - \vec{n}_x\cdot\vec{e}_y\right) \tag{2.4}$$

where N is the number of blades in a stage, p is the static pressure, and $\vec{A}, \vec{n},$ and $\vec{e}$ are face area at node, face normal, and node vectors, respectively. The remaining objective function of this thesis is the *blade efficiency*, defined as follows:

$$Q = \vec{A}\cdot\vec{V}_n \tag{2.5}$$

where Q is the mass flux. The mass flow calculated at a node is given as:

$$\dot{m}' = \rho Q \tag{2.6}$$

Then, the mass flow averaged total pressure and temperature values at the inlet and exit boundaries of the blade are found by:

$$p_{t,b} = \frac{\sum \dot{m}' p'_{t,b}}{\sum \dot{m}'} \tag{2.7}$$

$$T_{t,b} = \frac{\sum \dot{m}' T'_{t,b}}{\sum \dot{m}'} \tag{2.8}$$

$$p_{t,c} = \frac{\sum \dot{m}' p'_{t,c}}{\sum \dot{m}'} \tag{2.9}$$

$$T_{t,c} = \frac{\sum \dot{m}' T'_{t,c}}{\sum \dot{m}'} \tag{2.10}$$

where the subscripts b and c indicates the flow conditions at the inlet and exit of the blade, respectively. Finally the blade efficiency is:

$$\eta = \frac{1 - \dfrac{T_{t,c}}{T_{t,b}}}{1 - \left(\dfrac{p_{t,c}}{p_{t,b}} \right)^{\frac{\gamma - 1}{\gamma}}} \tag{2.11}$$

2.2 Three-Dimensional Blade Profile Design Method

Following sections cover the aspects of blade shape design method implemented in this thesis. Firstly, the importance of the blade spacing selection, which can be expressed non-dimensionally as *solidity* (ratio of chord to pitch in B2B plane), will be discussed in the next section and shown that the number of blades in a stage is a very practical design parameter for representing the solidity. Secondly, blade profile design, including B2B design parameters, will be defined. Finally, blade geometry reshaping algorithm is discussed in the last section with examples of 3-dimensional blade shapes generated which shows the capability of the developed method.

2.2.1 Solidity

One of the important aspects of blade design is the selection of the blade solidity, which is the ratio of chord or axial chord to blade spacing at a given span as shown in Figure 2.5. A minimum value is usually desired from the standpoint of reducing weight, cooling flow, and cost. However, increasing the blade spacing eventually results in decreased blade efficiency due to separated flow, on the other hand lowered frictional losses. Therefore, an optimum solidity should be a fully attached flow with maximum blade spacing, and minimum frictional and secondary losses.

A very widely used traditional approach is the use of tangential loading coefficient, which is introduced by Zweifel [40]. Zweifel loading coefficient ψ_z is used to relate the actual and ideal blade loading in terms of the flow absolute inlet and exit angles α_1 and α_2, respectively. In equation form the axial solidity is given as,

$$\sigma_x = \frac{2}{\Psi_z}\frac{\cos\alpha_2}{\cos\alpha_1}\sin(\alpha_1 - \alpha_2) \qquad (2.12)$$

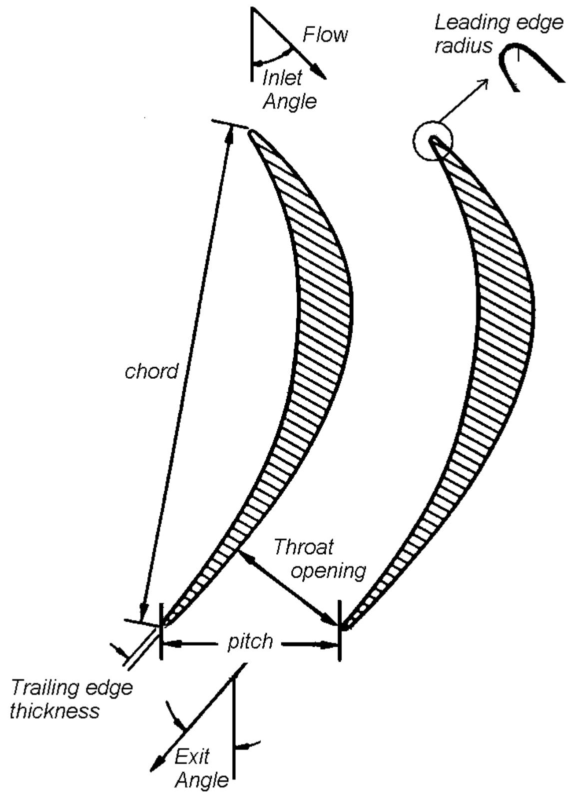


Figure 2.5 Definition of solidity and thicknesses

Therefore, according to Zweifel, solidity depends on exit and inlet angles, and for a given Zweifel loading coefficient, axial solidity should be decreased for decreasing exit flow angle to keep optimum solidity. This means that as long as the absolute value of exit angle increases, the blade spacing should be increased to decrease blade loss.

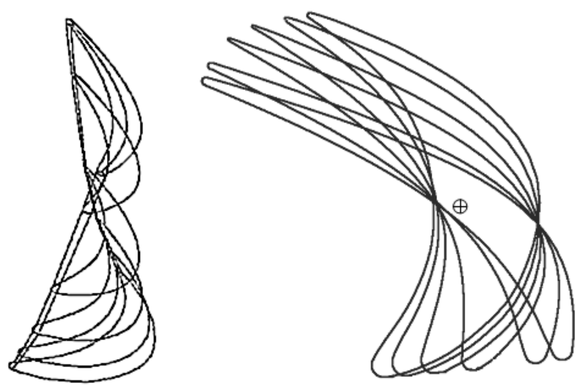
However, there are other loss factors affecting the solidity design parameter. The primary cause of losses is the boundary layer that develops on the blade and end-wall surfaces. Other losses occur because of shocks, tip-clearance flows, windage (disk friction), and flow incidence. A minimum loss occurs at some optimum solidity [41]. As solidity increases, the amount of frictional surface area per unit flow is increasing. On the contrary, as solidity reduced, the loss per unit surface area is increasing because of the increased surface diffusion required. Therefore, a minimum loss is expected to occur at a given value for solidity, and this requires an accurate prediction of blade losses.

For a 3-dimensional design problem, solidity varies along the blade, and is a function of blade height, it is not constant. Therefore, it is not a suitable to implement solidity as a design parameter. Rather, a constant parameter for a stage should be defined. The easiest and convenient way to represent solidity in a stage is to define and fix the number of blades in a row. In this thesis, an integer value which represents the number of blades in a row is used as the only design parameter in the place of solidity.

2.2.2 Blade Profile Design

The geometric model of the 3-dimensional blade profile must be defined as few as possible parameters in order to simplify the optimization problem. On the other hand, a robust geometric model should be able to cover distinct blade profiles in every corner of the design space, almost entirely, while generating realistic blade profiles. Consequently, the parameterization of the blade profile should minimize the possibility of generating geometrically unrealistic profiles.

In that sense, the 3-dimensional blade is represented by several 2-dimensional cylindrical surface layers which are stacked along the centroids (center of gravities) that are based on the hub layer centroid location in the tangential direction, as shown in Figure 2.6. The stacking on centroids has the main advantage of satisfying mechanical constraints to reduce bending stresses under rotational loads. Additionally, since a linear stacking line passing through the centroids of the layers is used, no any additional design variables that may be required for defining a stacking curve is necessary. This also helps avoiding increasing the number of design variables due to additional stacking curve parameters.



(a) (b)

Figure 2.6 Stacking layers (a) on centroid (b)

The number of layers, the spanwise locations of the layers, and the layer angles between the layer plane and the axial direction might be included in the design parameters. However, pre-setting these variables avoid a dramatic increase in design parameters. Moreover, for the design point of view, a focus on aerodynamic parameters such as blade angles rather than aforementioned geometric parameters is more suitable for the design engineer.

For these reasons, the number of layers is fixed and selected as 6. The spanwise locations of the layers are taken to be constant. The mid-layers' spanwise locations are set equal to that of the baseline blade, and hub and shroud layers are set as the hub and tip sections of the blade, respectively. Shroud layer plane is set in the shroud axis whereas the remaining layers' planes are set in the axial direction for the in this study. Consequently, the selected layers in the meridional view and their names are shown in Figure 2.7.

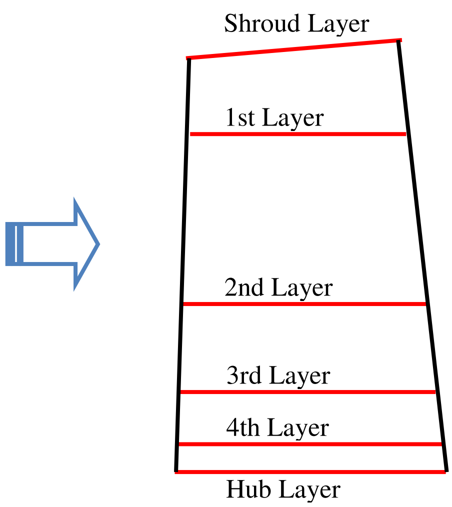


Figure 2.7 Six layers in the meridional view with definitions

3-dimensional end wall shaping is a recent practice for decreasing the adverse effects of secondary flows. Both in axial and circumferential directions, end wall contours are used to control corner vortex generation. One can include design parameters which sets the meridional and axial-tangential curves. However, this kind of investigation in blade design is beyond the objectives of this thesis and left for further investigations. Consequently, 2-dimensional end walls of the baseline blade are used and not included as a geometric design parameter.

2.2.3 Layer Geometric Parameterization

The geometric parameterization of the two-dimensional cylindrical surface layers is the critical item in the successful implementation of a turbine blade shape optimization method. The geometric model should include as few as possible parameters and be able to cover distinct and realistic layer shapes in every corner of the design space, almost entirely.

The point by point presentation of suction and pressure sides of a layer does not ensure smoothness of the blade passage and includes discontinuities which cause unavoidable shock waves. Additionally, an unnecessary large number of parameters needed to be used. Therefore, there are several attempts found in the literature.

The most basic and popular airfoil definition method is the NACA family [42], in which the blade is defined by a mean camber line and a thickness distribution. These airfoils cannot be applied to an optimization problem due to limited and restricted number of airfoils. A recent study shows an airfoil modification algorithm for layers shaped by adding thickness distribution to camberline [43]. However, this method uses only 4 design parameters and relies on the baseline camber and thickness distribution.

Pritchard implemented an 11 variable airfoil with circular arc and 3rd order polynomial curves tangent to each other [44]. However, polynomial curves suffer from discontinuities due to inflection points as well. Trigg improved this method by replacing the polynomial curves with Bezier curves [45], and increased the number of variables to 17. Anders represented the airfoil with two 5th order Bezier curves which requires 20 parameters [46]. Yamamoto and Inoue used cubic B-splines of camber line and thickness distribution for wing sections [47].

Pierret draw a comprehensive literature review on two-dimensional geometric turbomachinery airfoil parameterization [48]. He observed that any curvature discontinuity should be avoided in the critical parts of the airfoil, such as leading edge and throat regions. He proposed three-curve airfoil geometry and a second order curvature derivative for the leading edge and first order for the trailing edge. This geometry includes a circle at the trailing edge and two Bezier curves for suction and pressure sides. In order to decrease the number of variables and increase the realistic airfoil generation probability, he defined a camber line and selected the Bezier control points according to the camber line. However, this method decreases the coverage of the shapes in the design space, since the airfoil shape is built on 3-control point Bezier curve.

In this study, the design engineers' traditional aerodynamic design parameters such as leading and trailing edge radiuses, and inlet and outlet metal angles are respected while generation the layer shapes, since these parameters are often imposed by manufacturing, mechanical or aerodynamic constraints. The layer is parameterized using five angles as shown in Figure 2.8. These angles are the wedge angles of leading and trailing edges (W_{TE}, and W_{LE}), blade inlet and exit metal angles (β_{LE}, and β_{TE}), and the stagger angle (ξ). Additionally, circumferential rotation angle Θ is selected as the sixth design parameter since this angle sets the lean of the blade in three dimensional geometry.

Consequently, there are six design parameters selected for a layer geometry shaping. Since there are six pre-determined layers and one more design parameter which is the number of blades, a total number of 37 design parameters are used to define one particular 3-dimensional blade profile.

2.2.4 Generation of Layer Curves

In order to generate the layer profiles, four different curves are used. Two Bezier curves are selected to represent pressure and suction sides of the layer, and remaining two ellipse cuts for leading and trailing edges. Continuity up to second order derivative is ensured at the junction points of these curves.

A complete description of curves and surfaces can be found in the literature [49] and [50]. The main reason of using Bezier curves instead of polynomials is that although inflection points can occur with Bezier curves, they are far less than polynomials. Moreover, curve definition is simpler, and the degree of the curve can be very easily increased. More importantly, the parametric form of the equations allows obtaining the coordinates of the blade points at any points [48].

A Bezier curve is specified by the coordinates of control points in which only the first and the last lie on the curve they define. The curve is the weighted average of these control points defining the curve. The Bezier curves of 3rd degree (which is also used in this study) require 4 control points, and are defined as follows:

$$\vec{R} = (1-t)^3 \vec{P}_0 + 3t(1-t)^2 \vec{P}_1 + 3t^2(1-t)\vec{P}_2 + t^3 \vec{P}_3 \qquad (2.13)$$

where $\vec{P}_k$ are the vectors of the 4 control points, t is an auxiliary parameter varying between 0 and 1 and $\vec{R}$ is the coordinates of the corresponding point coordinates vector on the Bezier curve.

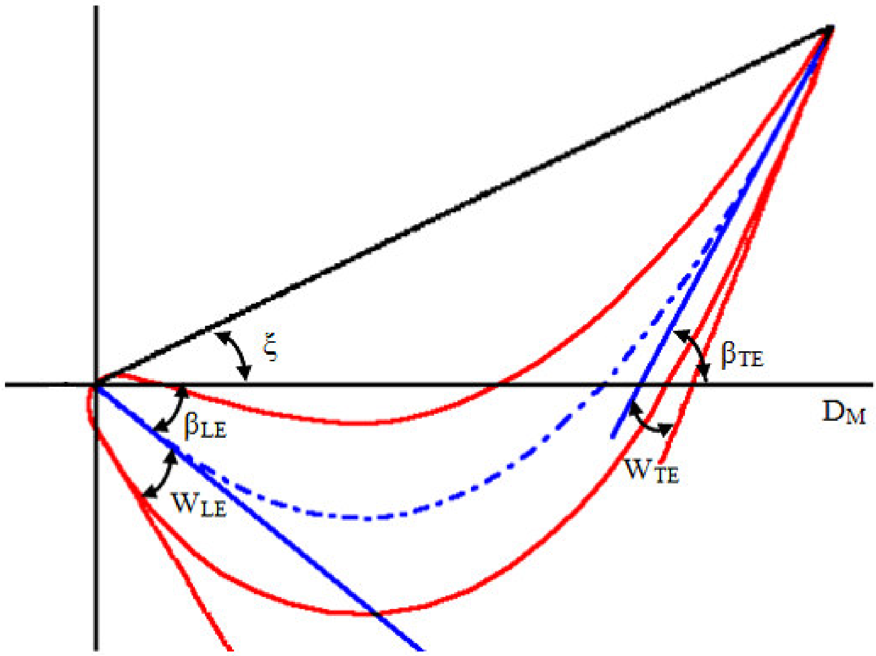


Figure 2.8 Layer geometric parameterization

Note that the 3rd order Bezier curve is constructed from 4 control points. The tangents at both end control points are defined by the line connecting the control points. The second derivative at end control points depends only the first three points.

The number of Bezier control points and their parameterization is another decision making issue for the designer. One can set more than 4 control point and the dimension of the Bezier curve can be increased freely. However, this increases the number of design parameters unnecessarily since a sufficient number of control points can cover the design space almost entirely. For demonstration purposes, only 4 control points are set to define the Bezier curves in this study, and it is suggested to use this value as a minimum requirement for the curve parameterization for future investigations.

In a two-dimensional definition, each control point of the Bezier curve requires two variables to set the coordinate values. Therefore, a two-dimensional 3rd order Bezier curve which has four control points is represented using 8 variables. However, optimization models utilizing global search methods such as genetic algorithms involve random generation of blade shapes. Therefore, it is critical to generate as much realistic blade shapes as possible at the blade profile generation phase.

Consequently, to limit the variation of these control points and generate as realistic blade profiles as possible, some profile generation procedures are applied. First, as shown in Figure 2.9 and 2.10, the end control points $P_{0,SS}$, $P_{0,PS}$, $P_{3,SS}$, and $P_{3,PS}$ are fixed by leading and trailing edge thicknesses, and blade inlet and exit metal angles (β_{LE}, and β_{TE}), respectively. The construction of the leading edge of the blade layer profile is shown in Figure 2.9. The trailing edge construction is similar. The leading and trailing thicknesses are set equal to that of the baseline blade, since these parameters are pre-determined according to blade cooling flow scheme. The blade leading and trailing edge shapes are constructed from an elliptical edge, which is formed through the intersection of the camberline and the meridional limit.

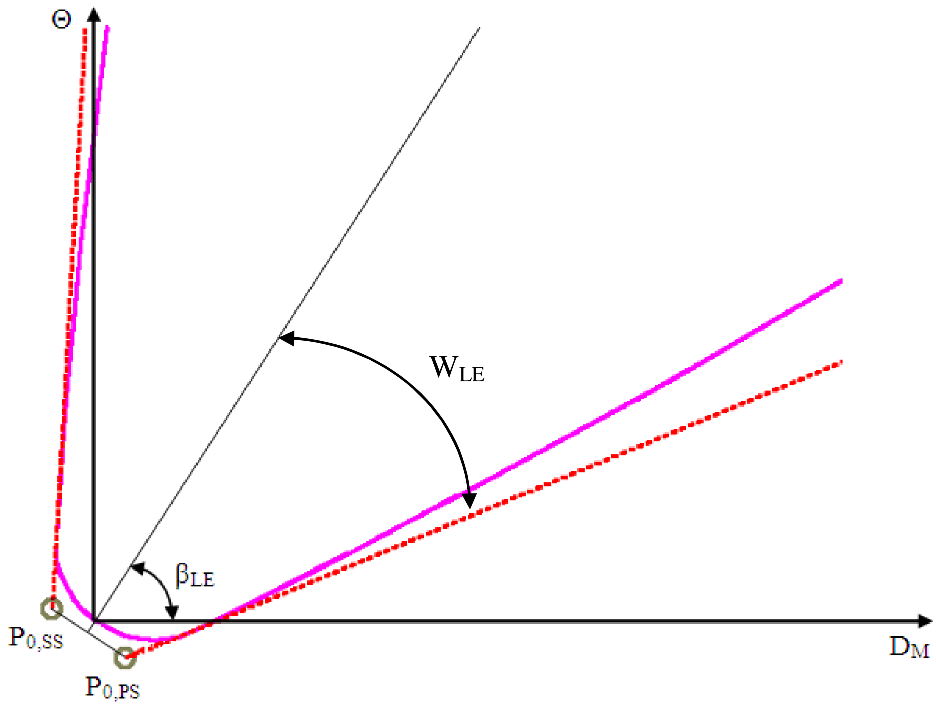


Figure 2.9 Leading edge construction of a blade layer profile

The remaining mid control points are limited by setting their D_M axis values fixed according to the baseline blade. Firstly, baseline blade point-by-point pressure and suction side curves are converted to Bezier curves of 3rd order, with end control points same as the layer profile Bezier curve end control points. Secondly, the D_M axis values of the baseline Bezier curves' mid control points are selected as the same as that of the generated blade layer profile mid control points. Hence, the D_M values of the mid control points are fixed. Finally, the design parameters that are the wedge angles of leading and trailing edges (W_{TE}, and W_{LE}) are used to set the Θ axis values of the mid control points as shown in Figure 2.10.

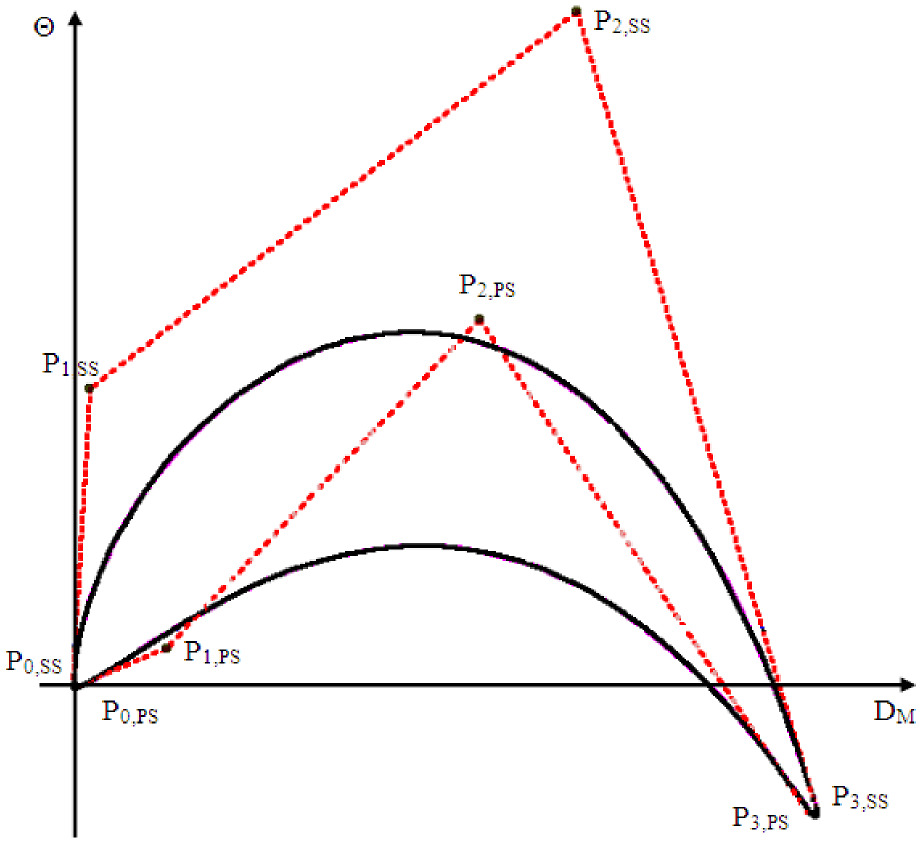


Figure 2.10 Blade layer profile control points

2.2.5 Blade Profile Reshaping Algorithm

The blade profile reshaping algorithm is given in Figure 2.11. According to the algorithm, there are two main steps. In the first step, the baseline blade geometry is parameterized. In the second step, the baseline blade geometry parameters are modified according to the design parameters.

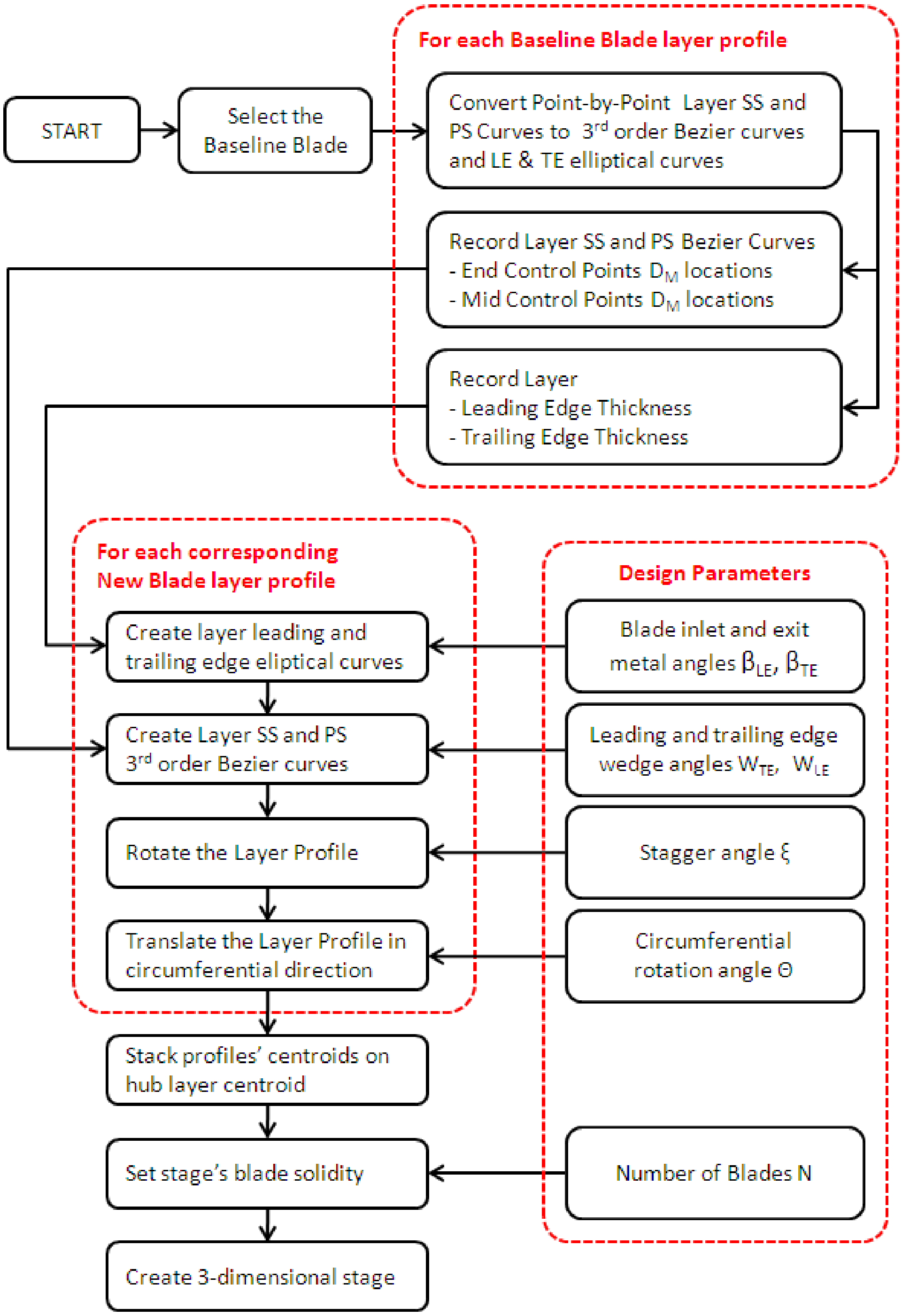


Figure 2.11 Blade profile reshaping algorithm

2.2.6 Blade Profile Reshaping Examples

Figure 2.12 shows four different blade profile examples generated from the baseline blade using the reshaping model. These profiles demonstrate the capabilities of the method to represent a diverse range of blade shapes in the design space. Except swept blades, the reshaping algorithm is capable of producing conventional, positive/negative leaned, tapered, twisted blade shapes.

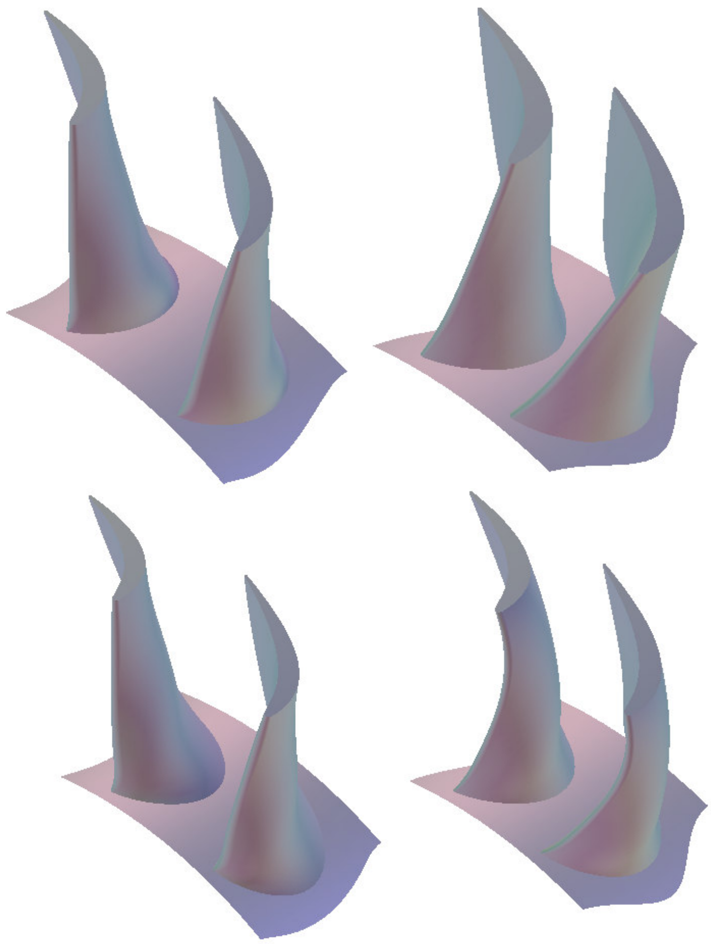


Figure 2.12 Three-dimensional blade profile examples of conventional rotor, positive lean, twisted, and negative lean profiles

2.3 Automatic Mesh Generation

For 3-dimensional meshing, a mesh generator, ANSYS® TurboGrid™ software [51], is used for unstructured automatic volume meshing. Mainly, the program automatically generates a triangular surface mesh and the interior tetrahedral mesh using the Octree approach [52]. Automatic meshing of the volume is done by successive refinement until all grid density requirements are met. The resulting tetrahedral mesh is an adaptive mesh of non-uniform density. The automatically generated UH turbine 1st stage baseline stage mesh is shown in Figure 2.13 [53].

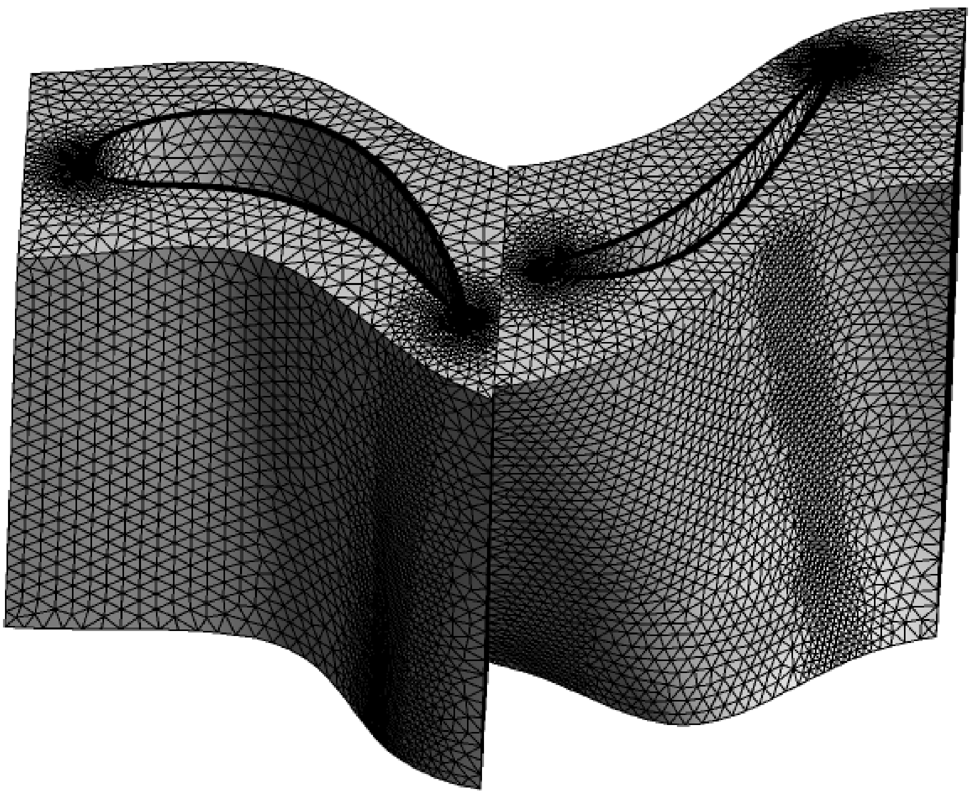


Figure 2.13 UH turbine 1st stage rotor automatic mesh generation

In leading and trailing edge regions of high surface curvature, the mesh is automatically refined in order to maintain the necessary geometric resolution as shown in Figure 2.14. This improves the robustness and eliminates the additional user input.

The flow adjacent to suction and pressure side walls are characterized by high flow variable gradients in the normal direction. Moreover, the boundary layer development in the turbine blade is the main factor affecting the efficiency of the blade. While tetrahedral cells can be used in such boundary layers, greater accuracy can be achieved using prism elements.

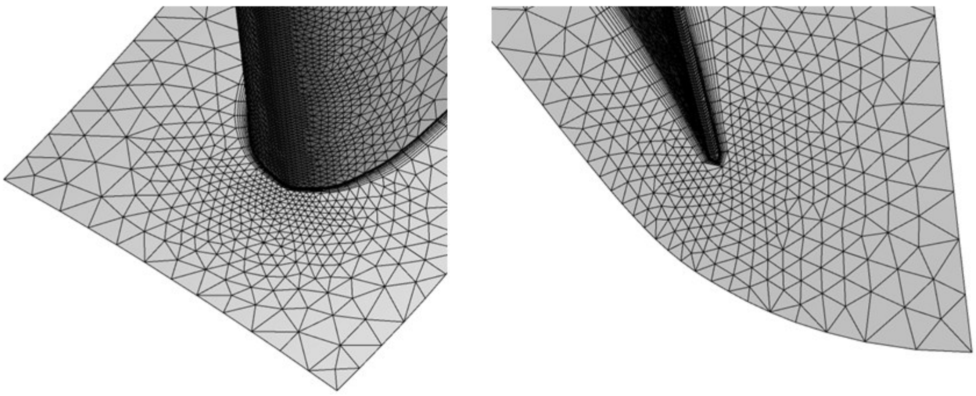


Figure 2.14 Zoomed view of leading and trailing edge regions at hub

The meshing software automatically arranges layers of prism elements near the boundary surfaces in order to appropriately model close to wall physics. Prisms are generated by extruding the triangular surface mesh. As a result, a hybrid tetrahedral grid consisting of prism elements near the boundary surfaces and tetrahedral elements in the interior of the space is meshed. The prismatic mesh is inflated by a prescribed number of layers, which is set as ten layers in this study. The prismatic mesh of the UH stator blade surface boundary is shown in Figure 2.15.

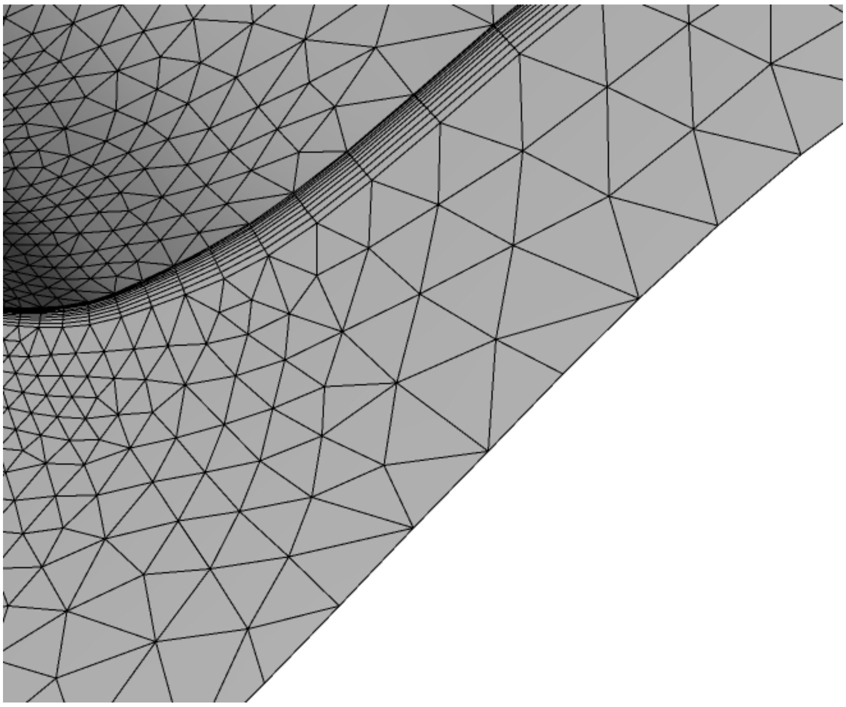


Figure 2.15 Zoomed view of leading edge pressure side region showing prismatic mesh elements on the boundary

Since the automatic meshing of the volume is done by successive refinement, a pre-defined grid refinement level is used to meet all grid density requirements. The mesh quality represents a tradeoff between the accuracy of the solution and computational cost. In this thesis, two different grid refinement levels is used for a low quality and a high quality mesh generation, which are called as coarse grid, and fine grid, respectively.

2.4 Three-dimensional RANS Flow Solver

A commercial RANS flow solver, ANSYS® CFX® software [54], is used in this study for evaluating the objective functions. The code is capable of solving a three-dimensional steady state compressible viscous Reynolds averaged Navier Stokes equations in conservation form.

The five equations of mass, momentum and energy conservation in a stationary or rotating frame of reference are solved for seven unknowns ($\vec{V}$, p, T, ρ, h), which the set is closed by adding two algebraic thermodynamic equations: the Equation of State, which relates density to pressure and temperature; and the Constitutive Equation, which relates enthalpy to temperature and pressure.

The original unsteady Navier-Stokes equations are modified by the introduction of averaged and fluctuating quantities to produce the Reynolds Averaged Navier-Stokes (RANS) equations. These equations represent the mean flow quantities only, while modeling turbulence effects without a need for the resolution of the turbulent fluctuations. All scales of the turbulence field are being modeled. The averaging procedure introduces additional unknown terms containing products of the fluctuating quantities, which act like additional stresses in the fluid. These Reynolds' stresses terms are difficult to determine directly and so become further unknowns. The Reynolds (turbulent) stresses need to be modeled by additional equations of known quantities in order to achieve "closure".

The equations used to close the system define the type of turbulence model. Turbulence models close the Reynolds-averaged equations by providing models for the computation of the Reynolds stresses and Reynolds fluxes. Among many popular turbulence models available such as 2 equation turbulence models, a zero-equation turbulence model is used in this study in order to keep robustness and decrease sophistication. The Zero Equation model implemented in this study is simple to implement and use, can produce approximate results very quickly, and provides a good guess. In this context, a constant turbulent eddy viscosity is calculated for the entire flow domain.

Very simple eddy viscosity models compute a global value for turbulent viscosity, μ_t, from the mean velocity and a geometric length scale using an empirical formula. Because no additional transport equations are solved, these models are termed 'zero equation'. The zero equation model in the solver uses an algebraic equation to calculate the viscous contribution from turbulent eddies. A constant turbulent eddy viscosity is calculated for the entire flow domain.

The turbulence viscosity is modeled as the product of a turbulent velocity scale, U_t, and a turbulence length scale, l_t, as proposed by Prandtl and Kolmogorov [55];

$$\mu_t = \rho f_\mu U_t l_t \tag{2.14}$$

where f_μ is proportionality constant. The velocity scale is taken to be the maximum velocity in the fluid domain. The length scale is derived using the formula:

$$l_t = (V_D^{1/3})/7 \tag{2.15}$$

where V_D is the fluid domain volume.

Engineering transition predictions are based mainly on two modeling concepts. The first is the use of low-Reynolds number turbulence models, where the wall damping functions of the underlying turbulence model trigger the transition onset. The second approach is the use of experimental correlations. The correlations usually relate the turbulence intensity, Tu, in the free-stream to the momentum-thickness Reynolds number, $R_{e\theta}$, at transition onset.

For the zero equation transition model a prescribed intermittency, which is used to trigger transition, is set by the solver. The specified intermittency is based on the x, y and z co-ordinates. This way, if else statements are used to defined geometric bounds where the intermittency can be specified as zero (laminar flow) or one (turbulent flow). This method is used to prescribe laminar zones at the leading edges of the blades.

One of the ways of modeling the flow in the near-wall region is the wall function method. The wall-function method of Launder and Spalding [56] is implemented in the solver. In this approach, the viscosity affected sub -layer region is resolved by employing empirical formulas to provide near-wall boundary conditions for the mean flow and turbulence transport equations. Thus, computational resources are significantly saved using these empirical formulas. These formulas connect the wall conditions (e.g., the wall-shear-stress) to the dependent variables at the near-wall mesh node which is presumed to lie in the fully-turbulent region of the boundary layer.

The logarithmic relation for the near wall velocity is given by:

$$u^+ = \frac{U_T}{u_\tau} = \frac{1}{\kappa}\ln(y^+) + c \qquad (2.16)$$

where

$$y^+ = \frac{\rho \Delta y u_\tau}{\mu} \qquad (2.17)$$

$$u_\tau = \left(\frac{\tau_\omega}{\rho}\right)^{1/2} \qquad (2.18)$$

u^+ is the near wall velocity, u_τ is the friction velocity, U_T is the known velocity tangent to the wall at a distance of Δy from the wall, y^+ is the dimensionless distance from the wall, τ_ω is the wall shear stress, κ is the von Karman constant and c is a log-layer constant depending on wall roughness.

The above wall function equations are appropriate when the walls can be considered as hydraulically smooth. For rough walls, the logarithmic profile still exists, but moves closer to the wall. Roughness effects are accounted for by modifying the expression for u^+ as follows:

$$u^+ = \frac{1}{\kappa}\ln\left(\frac{y^+}{1+0.3k^+}\right) + c \qquad (2.19)$$

where

$$k^{+} = \frac{\rho y_R u_\tau}{\mu} \tag{2.20}$$

and y_R is the equivalent sand grain roughness height [57], which is not exactly equal to the real roughness height of the surface under consideration. Wall friction depends not only on roughness height but also on the type of roughness (shape, distribution, etc.); therefore, an appropriate equivalent sand-grain roughness of 0.01 mm is used throughout of this study.

2.5 Boundary Conditions

Inlet, outlet, periodic and wall boundary layers are used to numerically close the RANS equations. Total pressure, total temperature and swirl angle profiles are used in absolute frame of reference for the inlet boundary conditions. For a stator, flow inlet conditions are used. If the blade is a rotor, stator exit conditions are imposed. Either static pressure or mass flow rate is used as the outlet boundary condition.

Instead of simulating one stage, the problem is divided first into two rows, and then one blade only. First, the stator row is solved by freezing the rotor row. In the stator row, only one blade is solved by implementing periodicity boundary conditions between the blades. Finally, the rotor blade is solved using the exit conditions of the stator in the rotating frame of reference.

For the hub, shroud and blade surfaces, wall boundary condition is used. Note that a wall roughness of 0.01 mm is set for these surfaces. Blade, hub and shroud surfaces are modeled with no-slip adiabatic wall conditions. Both stator and rotor is solved using rotating frame of reference with setting either zero angular velocity or the rpm of the machine, respectively. Rotor shroud is set stationary in the absolute frame of reference.

2.6 Test Case Studies With Different Grid Qualities

Although the commercial flow solver has been tested for various problems, the ability of the current solution procedure is demonstrated for two popular test cases. This will further demonstrate the selected methods such as the turbulence model, transition prediction and boundary layer developments. Moreover, since, two different grid refinement levels is used for a low quality and a high quality mesh generation, which are called as coarse grid, and fine grid, respectively, in this thesis, the effect of grid quality is further investigated by test cases.

The two selected experimental test cases are relevant to three-dimensional flow calculations for viscous flows in axial turbine calculations: VKI and UH. The VKI low speed annular turbine blade row test case is used to validate radial-tangential downstream plane flow variables [58]. The VKI test case is solved for coarse and fine grids, and the grid quality effect on the results are shown. The UH 4-stage low speed turbine test case results are used to validate stage performance parameters for only the first stage [53]. This test case is used to demonstrate the off-design performance predictions of the solver.

2.6.1 VKI Low Speed Annular Turbine Blade Row

The VKI test case is a low speed, low aspect ratio, high speed annular nozzle guide vane [58]. The blades have a constant profile over the blade height and are untwisted. To account for differences in the upstream flow conditions for an inlet guide vane and an intermediate stage vane, the annular cascade was tested with skewed inlet end wall boundary layers. The inlet skew was generated by rotating the upstream hub end wall.

One of the two aims of the test case validation is to verify 3D flow field through the blade row with particular attention to the end wall and wake regions. The second aim is to investigate the grid quality effect on the calculated flow parameters, and to compare the required computational time for coarse and fine grids.

The cascade geometry is first constructed according to the test case parameters. Then, the geometry is parameterized according to design parameters as shown in Figure 2.11. Finally, the nozzle is solved using test case boundary conditions of upstream inlet total temperature, total pressure and swirl angle, and downstream static pressure using coarse and fine girds.

The coarse and fine gird generation parameters are listed in Table 2.1. For both of the cases, the automatic grid is generated with 10 layer prismatic boundary cells with an expansion factor of 1.2. The aspect ratio of all elements is below 5. Corresponding generated grids are shown for coarse and fine meshes in Figure 2.16 and 2.17, respectively.

For the solution of the test case, standard air is used as the working fluid. The same settings and methods used in the optimization problems are set for the solver such as the turbulence model, wall roughness and so on. The calculated average values of Reynolds number and turbulence is $1.5\ 10^5$ and 7%, respectively.

Table 2.1 Comparison of Coarse and Fine Grid Parameters

	Coarse Grid	**Fine Grid**
Number of Prismatic Layers	10	10
Prismatic Layers Expansion Factor	1.2	1.2
Number of Nodes in Surface Mesh	7,946	31,597
Number of Faces in Surface Mesh	17,292	63,194
Number of Nodes in Volume Mesh	81,885	454,785
Number of Tetrahedral Elements in Volume Mesh	263,107	1,955,673
Number of Prismatic Elements in Volume Mesh	64,420	206,570
Number of Elements in Combined Mesh	327,527	2,162,243

Table 2.2 Comparison of Coarse and Fine Grid Solution CPU Times

	Coarse Grid	**Fine Grid**
Geometry Generation [s]	10	10
Grid Generation [s]	157	705
Solver Run [s]	827	3,564
Total CPU Time [min]	16.6	71.3

The CPU times for the coarse and fine mesh solutions are tabulated in Table 2.2. It is shown that a coarse mesh solution is more than 4 times faster than the fine mesh solution. Although the accuracy level of the coarse mesh solution is lower than the fine mesh solution, the solution time is considerable faster for the former, therefore, is a very suitable low fidelity model of the latter solution for the use in optimization method developed in this study.

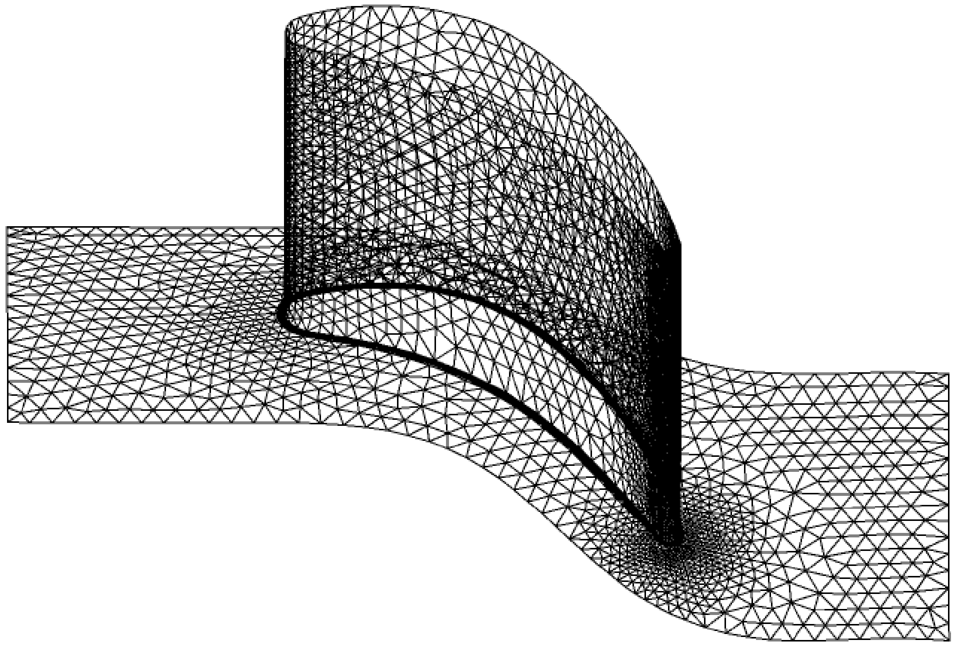


Figure 2.16 VKI Test Case - Coarse Mesh

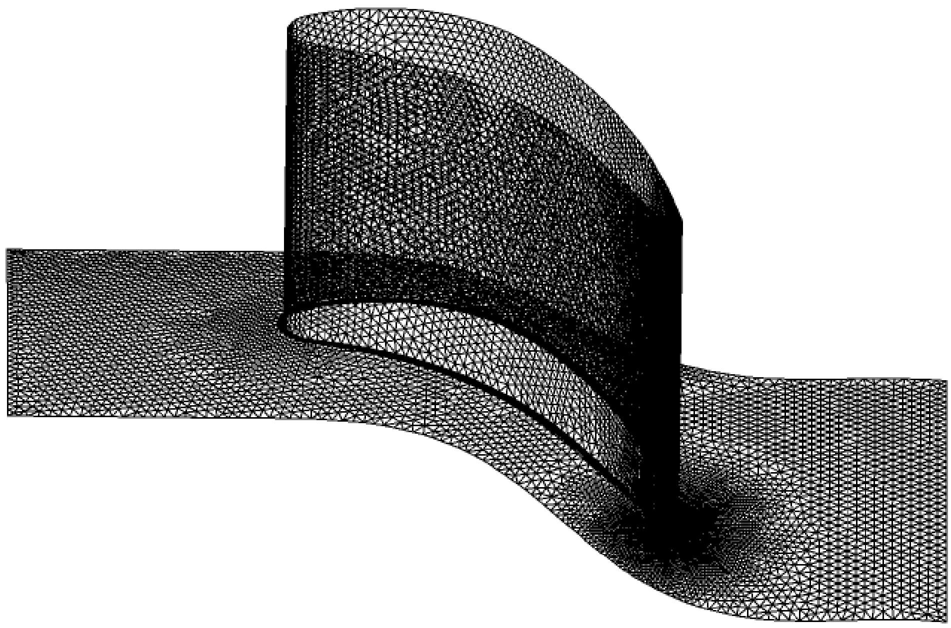


Figure 2.17 VKI Test Case - Fine Mesh

The test case upstream flow conditions are measured at a distance of X/C_{ax} = - 0.70, whereas the rotating hub extends from X/C_{ax} = - 0.15 to X/C_{ax} = -4.16. The numerical model takes into account only the measured upstream flow conditions as the inlet conditions of the problem, and the rotating hub is not modeled at all. Therefore, the remaining part of the rotating hub after the measurement plane could not be included in the model in any way.

The VKI test case measurement planes of X/C_{ax} = 0.86 and X/C_{ax} = 1.11 are shown in Figure 2.18. The former is located between the blades near trailing edge, and the latter is located at the downstream of the trailing edge. Contour plots of the static and total pressure coefficients CP_s and CP_0 at the measurement planes of X/C_{ax} = 0.86 and X/C_{ax} = 1.11 are shown in Figure 2.19 and 2.20, respectively. Contour plots of blade to blade exit flow angle β at the measurement planes of X/C_{ax} = 0.86 and X/C_{ax} = 1.11 are shown in Figure 2.21. These figures represent the exceptionally strong three dimensional features of the flow field.

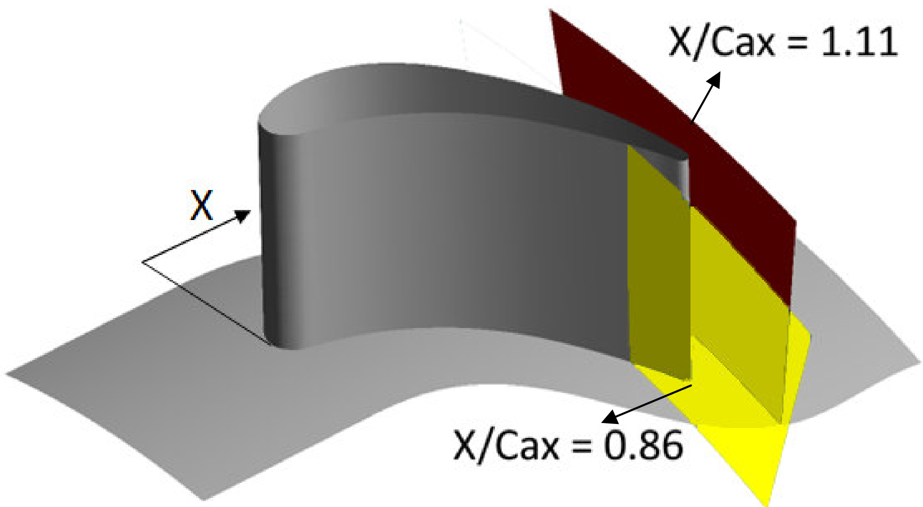


Figure 2.18 VKI test case measurement planes of X/C_{ax} = 0.86 and 1.11

According to Figures 2.19 and 2.20, both CP_s and CP_0 values are in very good agreement with coarse and fine mesh solutions. Hub and shroud wall surface region measurements have steeper gradients with respect to calculated results. This is attributed to simplicity of the selected wall roughness and turbulence model of the solver. When compared with the fine mesh solution, the coarse mesh solution resolution is somewhat lower, however at an acceptable degree. The results have the same order of magnitude and topology. It can also be said that the measurement plane location has no significant effect on the flow parameter prediction.

According to Figures 2.21 exit flow angle β values are in a very good agreement with coarse and fine mesh solutions. When compared with the fine mesh solution, the coarse mesh solution resolution is lower at an acceptable degree. The results have the same order of magnitude and topology. It can also be said that the measurement plane location has no significant effect on the exit flow angle parameter prediction.

Consequently, the VKI test case demonstrates the ability to predict stage losses sufficiently, which in turn is used to calculate the efficiencies. Flow turning is directly related to torque calculations and therefore accurate predictions of the flow exit angles are essential. The solution method is validated against the VKI test case which represents exceptionally strong flow turnings. Additionally, the test case is used to compare the difference of flow filed calculations for the coarse and fine mesh. The comparison showed that although the resolution of the calculated contours is lower for the coarse mesh, the flow filed has the same topology with the same order of magnitude as of the fine mesh.

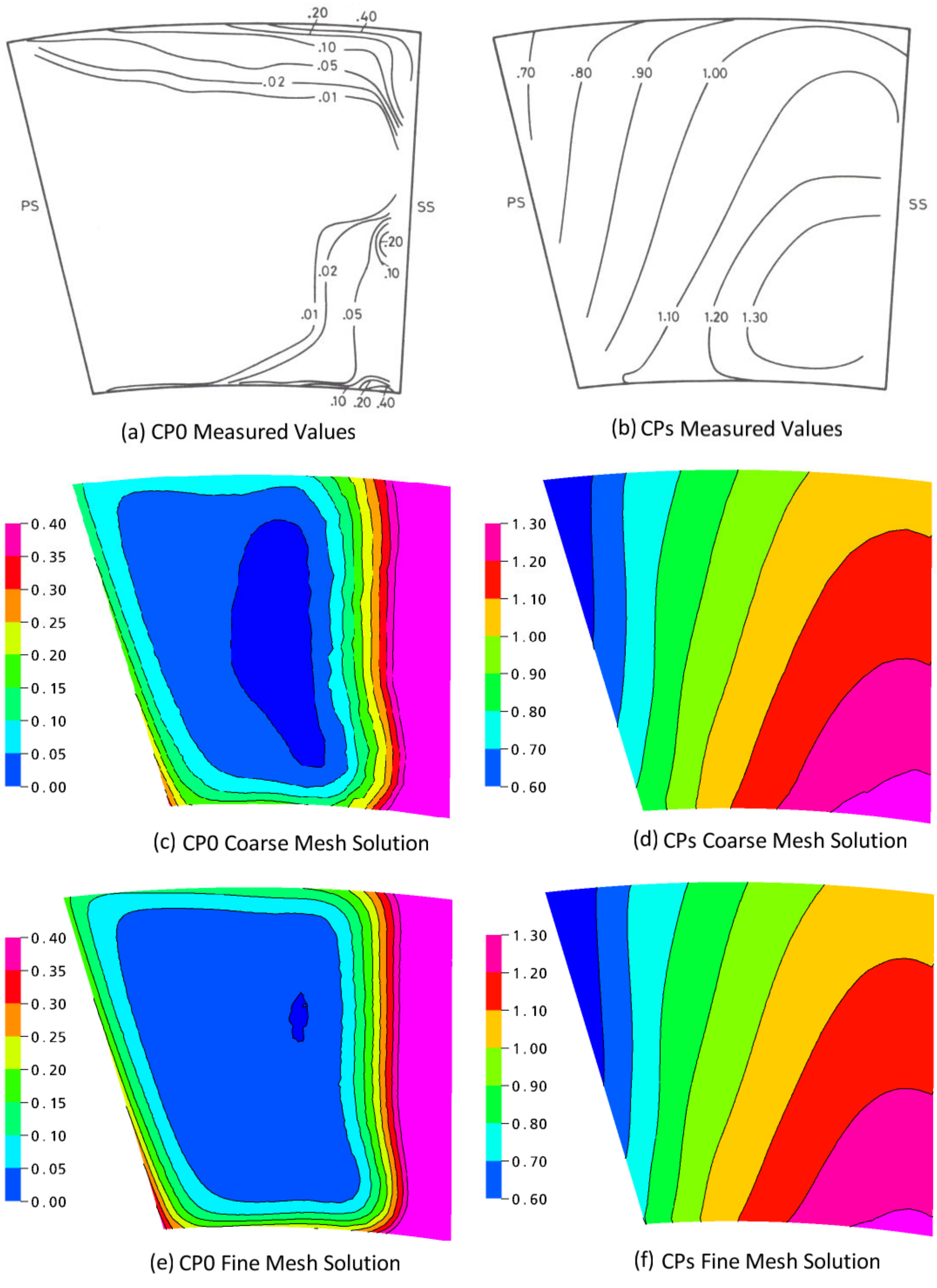


Figure 2.19 Coarse and fine mesh RANS solution comparison of CP_0 and CP_s values with measurements [58] at plane $X/C_{ax} = 0.86$

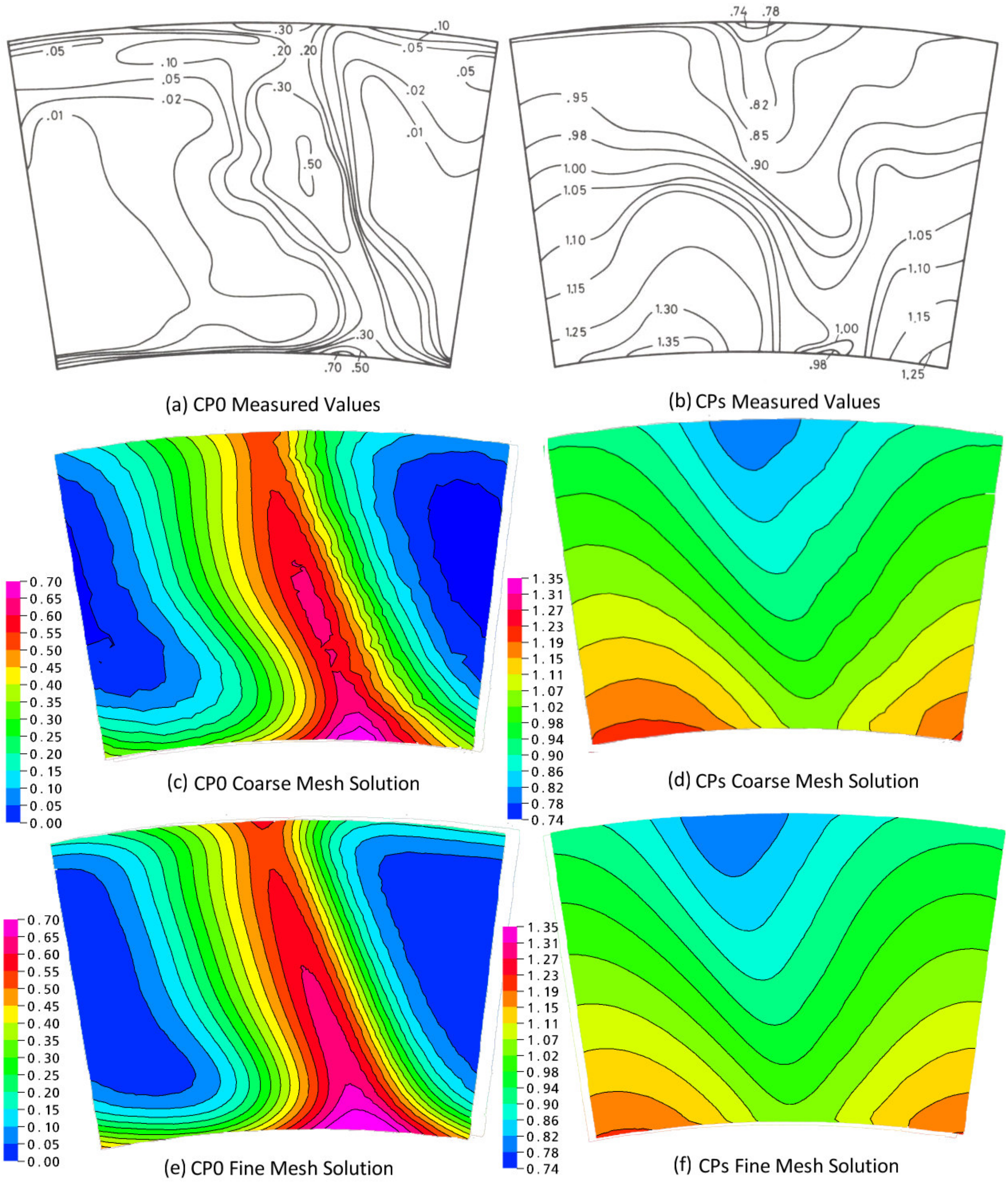


Figure 2.20 Coarse and fine mesh RANS solution comparison of CP_0 and CP_s values with measurements [58] at plane $X/C_{ax} = 1.11$

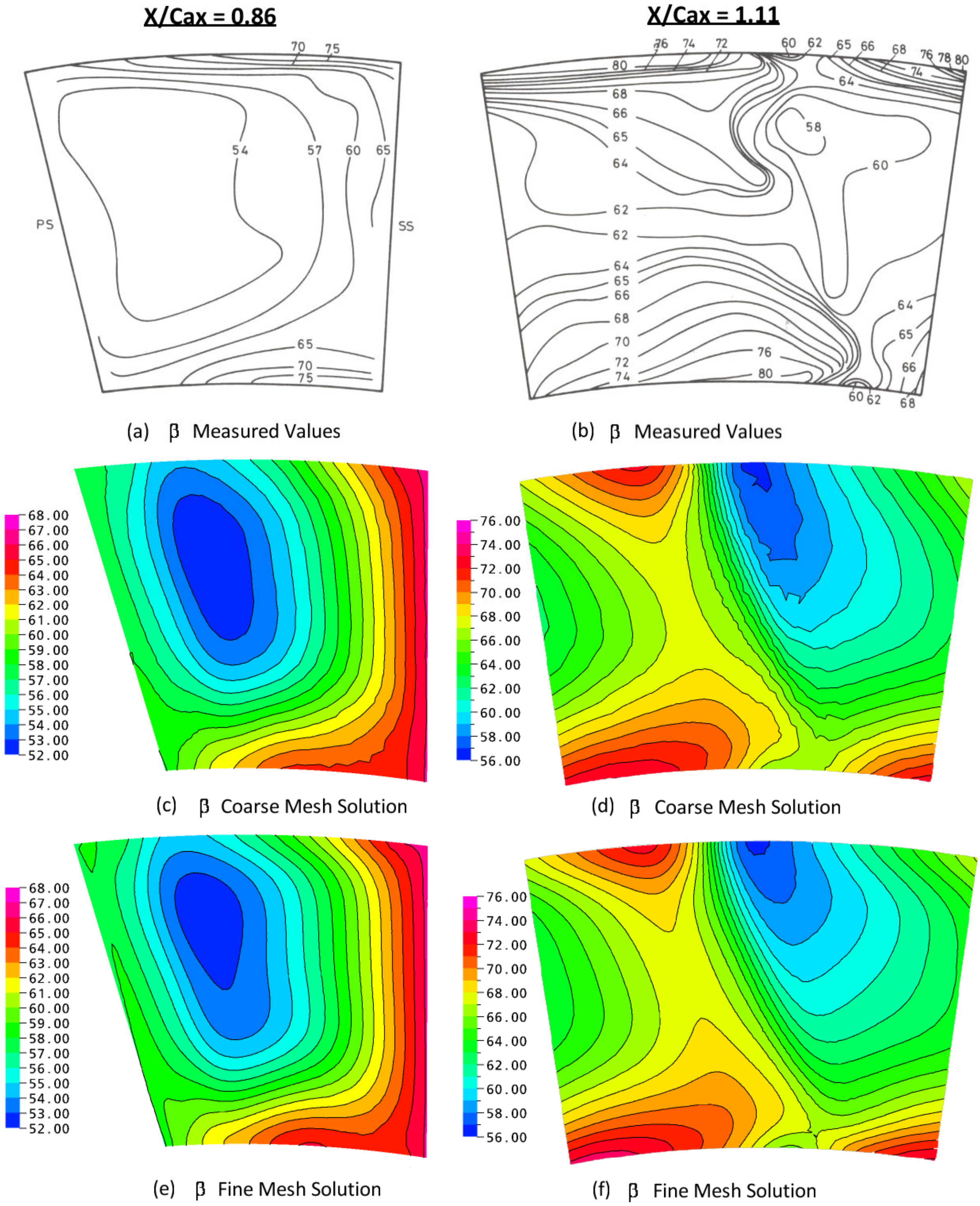


Figure 2.21 Coarse and fine mesh RANS solution comparison of β values with measurements [58] at planes $X/C_{ax} = 0.86$ and $X/C_{ax} = 1.1$

2.6.2 UH 4-Stage Low Speed Turbine

The UH 4-stage low speed annular turbine blade row test case results are used to validate spanwise stage performance parameters for only the first stage [53]. This test case is used to demonstrate the on-design and off-design performance predictions of the solver.

The UH turbine is designed for a rotational speed of 7500 rpm, and a mass flow rate of 7.8 kg/s which is set with the aid of by-pass. The blading is of the free-vortex type with a 50-percent degree of reaction at the middle section of the last stage. A tip clearance of 0.4 mm is used for the rotor. Only through flow radial traverse measurements at the upstream of the stator and the downstream of the rotor were made. Total pressure p_t, static pressure p, total temperature T_t, and flow angle β are measured for design and off-design conditions.

The stage geometry is first constructed according to the test case parameters. Then, the geometry is parameterized according to design parameters as shown in Figure 2.11. Finally, the stage is solved using test case boundary conditions measured at the upstream inlet total temperature, total pressure and swirl angle, and mass flow rate. Only coarse mesh solution is used for validation.

The coarse gird generation parameters are listed in Table 2.3. The automatic grid is generated with 10 layer prismatic boundary cells with an expansion factor of 1.2. The aspect ratio of all elements is below 5. Corresponding generated mesh of stator and rotor is shown in Figure 2.22.

Table 2.3 Coarse grid parameters of stator and rotor

	Stator	**Rotor**
Number of Nodes in Surface Mesh	16,556	16,909
Number of Faces in Surface Mesh	33,112	33,818
Number of Nodes in Volume Mesh	206,650	214,977
Number of Tetrahedral Elements in Volume Mesh	637,502	669,731
Number of Prismatic Elements in Volume Mesh	177,265	182,856
Number of Elements in Combined Mesh	814,767	852,587

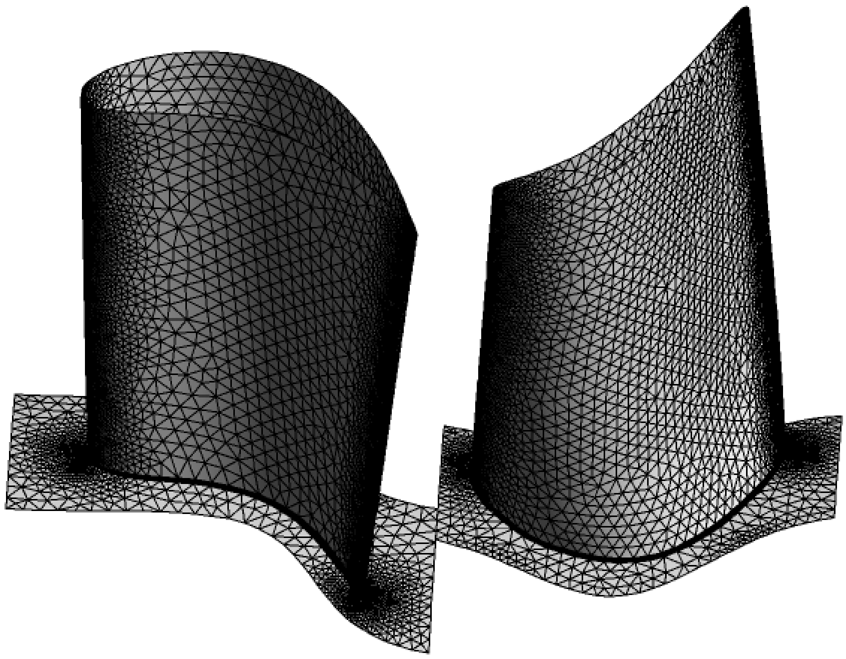


Figure 2.22 UH turbine 1st stage coarse mesh of stator and rotor

The problem is solved using frozen rotor approach. The stator is solved with problem boundary conditions, first. Then, the rotor is solved using outlet flow conditions of the stator as the upstream boundary conditions, and setting the same mass flow rate. Finally, the test case is solved for design and six off-design conditions. For the solution of the test case, standard air is used as the working fluid. The same settings and methods used in the optimization problems are set for the solver such as the turbulence model, wall roughness and so on. The calculated average values of Reynolds number and turbulence is 1.5 10^5 and 7%, respectively.

Table 2.4 Coarse mesh average CPU Times for stator and rotor

	Stator	**Rotor**
Geometry Generation [s]	10	10
Grid Generation [s]	287	298
Solver Run [s]	1,945	2,783
Total CPU Time [min]	37,4	51,5

The CPU times for the coarse mesh solutions for design and of design conditions are averaged per run and tabulated in Table 2.4. It is shown that a stator solution is faster than the rotor solution, and the test case can be solved in one and a half hour using coarse mesh.

Table 2.5 The UH test case experimental conditions

	RPM	**Mass Flow [kg/s]**
On-design Condition	7500	7,8
Off-design Condition 1	7500	6,5
Off-design Condition 2	7500	4,6
Off-design Condition 3	7500	4,1
Off-design Condition 4	5625	5,5
Off-design Condition 5	5625	3,9
Off-design Condition 6	5625	3,2

The on-design and off-design conditions are listed in Table 2.5. The UH test case is solved for these 7 cases and the calculated results are compared. Figure 2.23 shows the measured and computed total pressure distribution along the span at the downstream of the rotor blade. The measurements are made in nine radial locations. It is seen that the predictions are 2 to 3 percent lower than the measured values. This is acceptable in the uncertainty region of the measured values, which is indicated as 2 %.

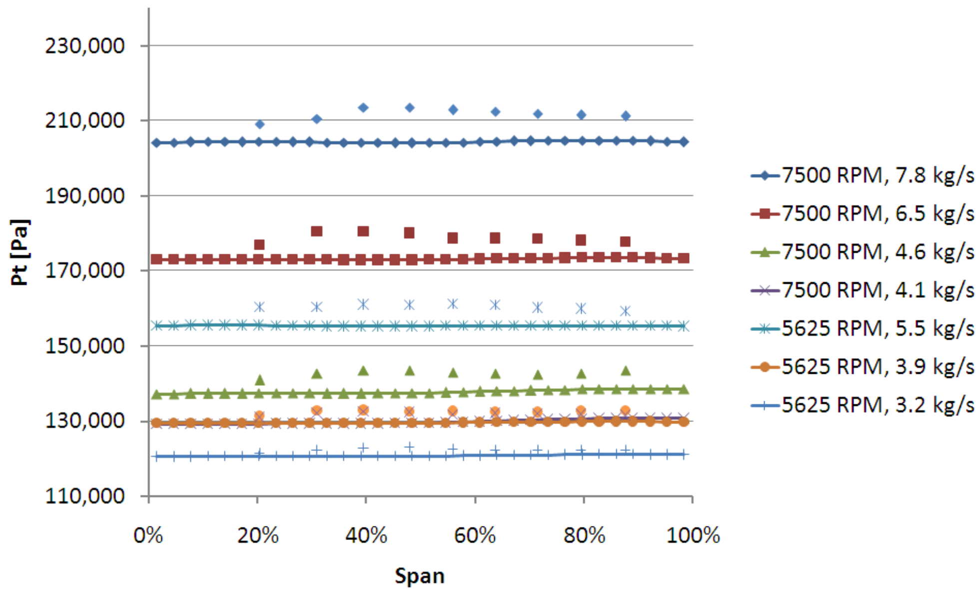


Figure 2.23 Comparison of spanwise total pressure distribution at the downstream plane of the rotor

Figure 2.24 compares the measured static pressure values with respect to the calculated values at on and off-design conditions. In any case, the calculated values are slightly lower than the measured values, and the difference is well within the uncertainty level of 2 %. Figure 2.25 compares the measured total temperature values with respect to the calculated values at on and off-design conditions. The predicted values matched exactly from mid-span to shroud. However, there is a slight difference between measured and calculated values from hub to mid-span, more on the near hub region. This difference is attributed to the boundary layer development at the hub end-wall of the rotor.

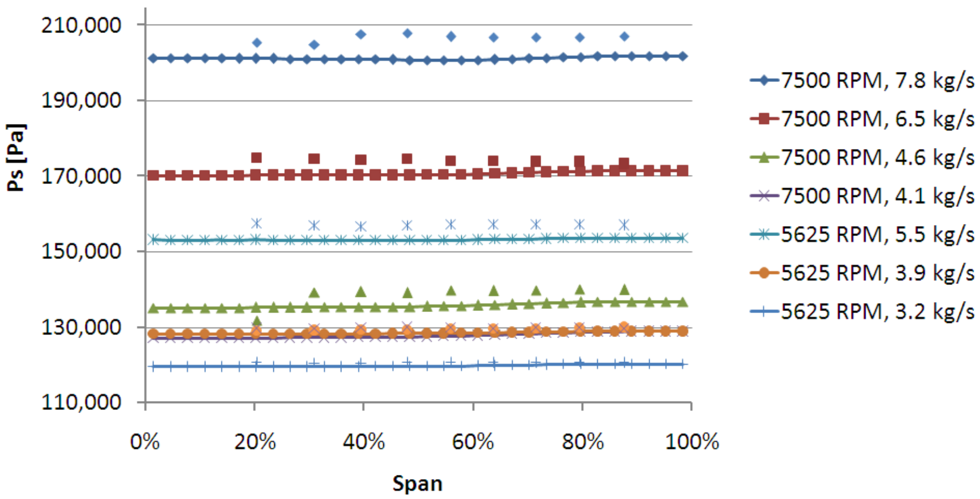


Figure 2.24 Comparison of spanwise static pressure distribution at the downstream plane of the rotor

The most difficult predicted parameter is the exit flow angle. This is because the simulation is performed without any tip clearance at the rotor, whereas the experimental setup has a tip clearance of 0.4 mm. Additionally, since there was no measured data between the stator and rotor, the rotor inlet conditions could only be supplied from the calculated stator downstream flow field. Moreover, rotor-stator interaction is not taken into consideration since the stage is not solved simultaneously, but the stator and rotor are solved independently. According to these observations, Figure 2.26 shows that the exit flow angle matches best near the mid-span location. Farther from the mid-span, the deviation grows, especially at the hub and shroud regions. The order of magnitudes is the same for measured and calculated exit flow angles for all conditions. The variation of exit flow angle with respect to mass flow and RPM parameters are simulated very well.

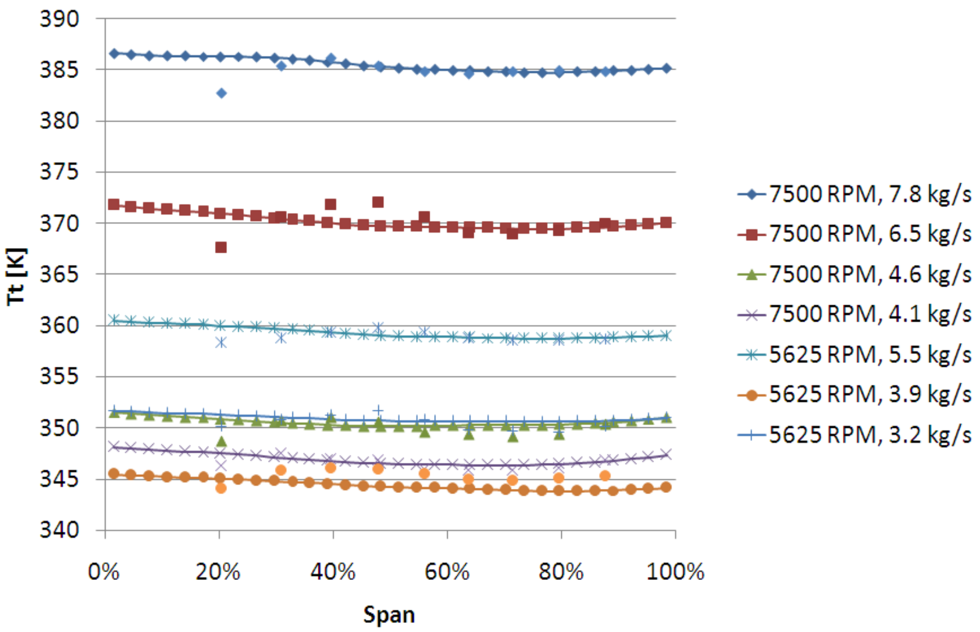


Figure 2.25 Comparison of spanwise total temperature distribution at the downstream plane of the rotor

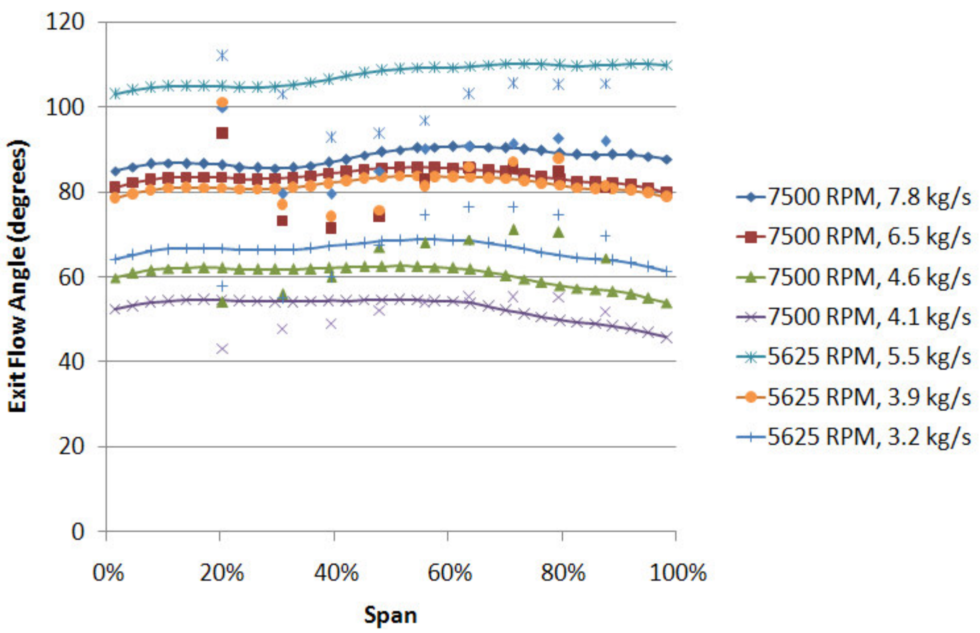


Figure 2.26 Comparison of spanwise exit flow angle distribution at the downstream plane of the rotor

2.7 Parallel Processing

Today's modern computational power is constructed using parallel processing super computers. The computations can be performed using distributed CPU nodes, which can be more than 1000 in some cases. Therefore, in an optimization environment, the power of parallel processing should be implemented to decrease the optimization time.

In this thesis, 50 CPU clusters are used for parallel processing at most. Although the solver is able to run in parallel, since the genetic algorithm is parallel in nature, the solver is not parallelized. Instead, the optimization algorithm is run in parallel such that in one generation, each of the individual blades in a population is evaluated at one PC which is called as slave PCs. The so called master PC is the one which the genetic algorithm itself is running.

The slave PCs are controlled in Microsoft Windows® environment. A visual basic application is developed to let the master PC command the slave PCs to run the solver. The application is defined as a service program, which starts at the startup of the slave PC, letting the master PC control at any time. Moreover, the data flow is controlled by TCP/IP network connection. Design parameters and evaluated objective function values are transferred by this simple network sharing.

2.8 Optimization Problem

The UH test case 1st stage rotor is selected as the baseline blade. The on-design conditions are selected as the operating conditions. The stator is set as fixed and is not optimized. The rotor is parameterized as shown in Figure 2.11. The objective functions are selected as the rotor torque T and efficiency η.

In this study, two different optimization problems are constructed: single and multi-objective aerodynamic optimization of turbine rotor. There are two separate single objective optimization problems solved, which are defined as:

$$\begin{aligned} &\text{minimize } f(x) = T \\ &\qquad \text{subject to } g_i(x) \leq 0 \end{aligned} \tag{2.21}$$

and:

$$\begin{aligned} &\text{maximize } f(x) = \eta \\ &\qquad \text{subject to } g_i(x) \leq 0 \end{aligned} \tag{2.22}$$

where the $g_i(x)$ are the design constraints.

The multi-objective objective function is defined as follows:

$$\begin{aligned} &\text{minimize } f(x) = T \\ &\text{maximize } f(x) = \eta \\ &\qquad \text{subject to } g_i(x) \leq 0 \end{aligned} \tag{2.23}$$

The only constraints are the mechanical constraints imposed on the optimization problems. For mechanical strength and cooling holes, trailing edge thickness which is determined by the wedge angle design parameter is constrained according to the baseline blade. In order to have enough metal to withstand the stresses, airfoil area is also constrained. The stagger angle is limited to restrict the overhang of the blade and to prevent interference with the dovetails of the adjacent blades in a row. Consequently, the constrained values with parameters are shown in Table 2.6. These values are given in ratios calculated by dividing with baseline values. As long as the constrained values are exceeded, the blade is assigned with the lowest objective function value possible.

Table 2.6 Mechanical Constraints

	min	**max**
TE Thickness	1	2
LE Thickness	0.80	1.60
Airfoil	0.75	1.25

CHAPTER 3

SINGLE OBJECTIVE OPTIMIZATION

In this chapter, the descriptions of the haploid and multiploid genetic algorithms are presented together with application of single objective aerodynamic optimization of turbine rotors. First, formulation of the design optimization problems is discussed. Then, a brief description of the haploid genetic algorithm (GA) technique is presented in the next section, followed by a more detailed description of the technique applied to the UH turbine rotor blade in section 3. Next, multiploid genetic algorithm is introduced. Finally, the application of the multiploid genetic algorithm to design problem is presented.

3.1 Formulation of the Design Optimization Problem

In general, design optimization is the process of achieving the best solution of a given objective or objectives while satisfying certain restrictions. If a *single-objective* function is to be minimized or maximized, then the problem is of single objective optimization nature. If there are several conflicting objectives, then the problem is formulated as a *multi-objective* problem, in which the goal is to minimize and/or maximize several objective functions simultaneously. This chapter deal with only one objective and the discussion will be based on the formulation of single-objective problems.

3.1.1 Optimization Variables

In a design optimization task, the numerical quantities for which values are to be chosen will be called the *design variables* or decision variables. In mathematical formulation, these quantities are denoted by x_n where n=1,2,…,N. These variables creates a *decision vector* **x** given as

$$\mathbf{x} = [x_1, x_2, ..., x_N]^T \tag{3.1}$$

where ^{T} denotes the transpose of the vector. Any specific vector **x** composed of numbers is called a *solution*. All solutions create the *design space* which is N-dimensional by nature.

Accordingly, there are six design parameters selected for a layer geometry shaping: the wedge angles of leading and trailing edges (W_{TE}, and W_{LE}), blade inlet and exit metal angles (β_{LE}, and β_{TE}), the stagger angle (ξ), and the circumferential rotation angle (Θ). Since there are six pre-determined layers and one more design parameter which is the number of blades, a total number of 37 design parameters are used to define one particular 3-dimensional blade profile. Therefore, the design space is 37-dimensional. Note that, any numerical combination of these variables constructs a different blade shapes two of which are illustrated in Figure 3.1. Corresponding design variable values of the two sample blade shapes are given in Table 3.1. There are infinite numbers of these solutions and the aim is to approximate the best solution in these many of solutions. In GA nomenclature, each one of the blade shape is called an *individual* of a generation.

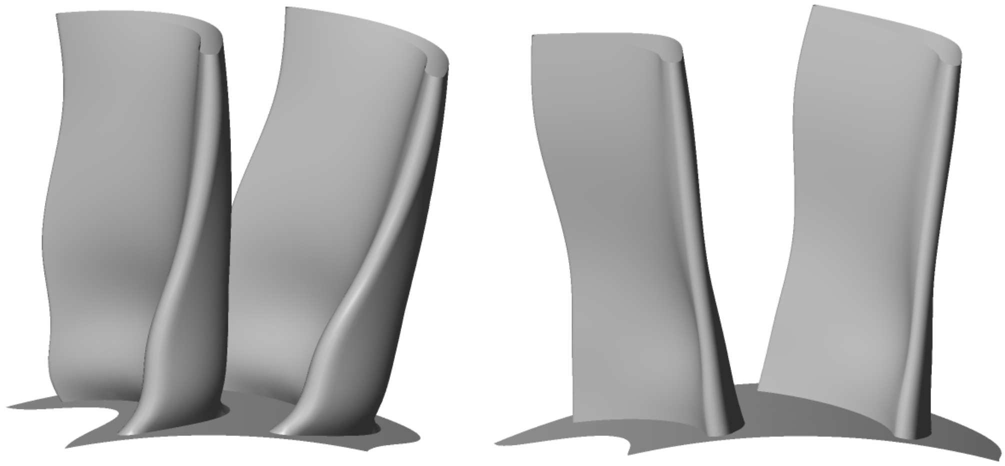


Figure 3.1 Two different blades (paired) in the design space

Table 3.1 Design parameters of sample blade shapes

Layer	Sample Blade 1 N = 32						Sample Blade 2 N = 26					
	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}
1 hub	12	1	-50	72	23	7	30	0	-10	50	20	5
2	20	-1	-50	60	24	4	30	0	-5	50	18	5
3	23	-2	-43	68	23	4	30	0	-10	50	15	5
4	30	-4	-32	60	23	3	30	0	5	50	20	5
5	36	-6	1	62	24	7	36	0	10	50	20	5
6 tip	40	-6	17	75	26	6	36	0	15	50	20	5

3.1.2 Constraints

In each engineering task, there are some restrictions dictated by environment and process and/or resources, which must be satisfied in order to produce an acceptable solution. These restrictions are collectively called *constraints*, and describe the dependencies among design variables. The constraints are written in the mathematical form of inequalities and sometimes equalities. The general form of writing inequality constraint is

$$g_k(\mathbf{x}) \geq 0 \text{ for k=1,2,…,K} \tag{3.2}$$

In certain models, we can have equality constraints as follows

$$h_m(\mathbf{x}) = 0 \text{ for m=1,2,…,M} \tag{3.3}$$

Note that the number of equality constraints, M, must be less than the number of design variables, N. Otherwise, the problem is over-constrained, and there is no degrees of freedom left for optimization.

There is no single way of implementing the constraints in a GA problem. The best way is to simplify the constraints so that they are only constraining the design variables. Keeping this in mind, the range of each of design variables of the turbine blade are geometrically constrained by the design space as shown in Table 3.2

Table 3.2 Design space constraints

Layer	Minimum Values						Maximum Values					
	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}
1 hub	10	0.1	-60	60	17	3	20	3	-50	75	35	9
2	15	-2	-55	60	17	3	25	-0.1	-45	75	35	9
3	20	-3.5	-45	60	17	3	30	-1.5	-35	75	35	9
4	30	-5	-35	60	17	3	40	-3.5	-20	75	35	9
5	35	-7	0	60	17	3	45	-5	10	75	35	9
6 tip	50	-8	10	60	17	3	60	-6	20	75	35	9
N	26						33					

Additionally, mechanical constraints are imposed on the optimization problems. The constrained values with parameters are shown in Table 3.3. These values are set in consistent with the baseline blade values. As long as the constrained values are exceeded, the blade is assigned with the lowest objective function value possible.

Table 3.3 Mechanical Constraints

	Minimum values			Maximum values		
L	t_{LE} (mm)	t_{TE} (mm)	A (mm^2)	t_{LE} (mm)	t_{TE} (mm)	A (mm^2)
1 hub	3	0.5	300	6	1	500
2	3	0.5	300	6	1	500
3	2.5	0.5	200	5	1	400
4	2	0.5	200	4	1	400
5	1.5	0.5	150	3	1	300
6 tip	1	0.5	150	2	1	300

3.1.3 Objective Function

In the process of selecting a good solution from all solutions, which satisfy the constraints, there must be a criterion, which allows these solutions to be compared. The mathematical formulation of this criterion is called *objective function*, f(**x**) and given as a function of decision variables. It can be either linear or nonlinear.

Aerodynamic optimization of turbine blades offers many objective functions such as total pressure loss coefficient, lift coefficient, drag coefficient, lift to drag ratio, blade loading, and blade energy loss, Mach number distribution. Since the blade efficiency which directly takes into account the losses, and blade torque are the final result of the design aim of a rotor, they are selected as the objective functions in this problem.

It is interesting that as an objective function, neither efficiency, nor the torque is simply an explicit function of the design variables. This is one of the two main reasons of utilizing a flow solver in order to calculate the objective functions. The solver calculates flow field of a given blade and then the objective functions value is found using equations presented in Chapter 2. The second reason is that the designer may not only wish to see the numerical value of objective functions, but also the pressure and velocity distribution of the flow field so that some physical meaning can be assigned such as shock location, boundary layer development, and separation.

3.1.4 Optimization Problems

Once the decision variables, constraints, and the objective function are determined, the optimization problem is constructed as follows:

$$\begin{array}{ll} \max. & f(\mathbf{x}) = f(x_1, x_2, ..., x_N) \\ \text{such that} & g_k(\mathbf{x}) \leq 0 \\ & h_m(\mathbf{x}) = 0 \\ & \mathbf{x} \geq 0 \end{array} \qquad \text{for k=1,2,…,K m=1,2,…,M} \quad (3.4)$$

Accordingly, the optimization problems of this thesis in a compact form is that

Torque Optimization:

minimize f(x) = T (3.5)
subject to Table 3.2. and Table 3.3

Efficiency Optimization:

maximize f(x) = η (3.6)
subject to Table 3.2. and Table 3.3

The calculated objectives are subsequently analyzed by the optimization algorithm, which evolutionarily approaches the optimum turbine blade shape.

3.2 Genetic Algorithm Optimization (GA)

The question now becomes which method should we use in order to efficiently and globally search the aerodynamically optimum blade in a multi-dimensional design space. To answer this question, the reasoning and arguments of Goldberg [20] is given here:

"The current literature identifies three main types of search methods: calculus-based, enumerative, and random…First, *calculus-based* methods are local in scope; the optima they seek are the best in a neighborhood of the current point…Second, calculus-based methods depend upon the existence of derivatives (well-defined slope values). Even if we allow numerical approximation of derivatives, this is a severe shortcoming…The real world of search is fraught with discontinuities and vast multimodal, noisy search spaces…It comes as no surprise that methods depending upon the restrictive requirements of continuity and derivative existence are unsuitable for all but a very limited problem domain. For this reason and because of their inherently local scope of search, we must reject calculus-based methods. They are insufficiently robust in unintended domains.

Enumerative schemes look at objective function values at every point in the search space; one at a time…such schemes must ultimately be discounted in the robustness race for one simple reason: lack of efficiency…Even the highly touted enumerative scheme *dynamic programming* breaks down on problems of moderate size and complexity. Random search algorithms have achieved increasing popularity as researchers have recognized the shortcomings of calculus-based and enumerative schemes...The *genetic algorithm* is an example of a search procedure that uses random choice as a tool to guide a highly exploitative search through a coding of a design space. Using random choice as a tool in a directed search process seems strange at first, but nature contains many examples. Another currently popular search technique, *simulated annealing*, uses random processes to help guide its form of search for minimal energy states...The important thing to recognize at this juncture is that randomized search does not necessarily imply directionless search."

Since the aerodynamic optimization of turbomachinery blades is highly non-linear and the large design space is highly multimodal, calculus-based and enumerative techniques were discounted as being either not robust enough or not efficient enough to find a global optimum. Thus, the two most efficient algorithms that are robust enough to search for a global optimum given this highly non-linear problem in this highly multimodal phase space (many local minimum) are the genetic algorithm [20] and the simulated annealing technique [59]. Advantages and disadvantages of these techniques are discussed by Davis [59]. Sufficient comparisons have not yet been made between the two techniques and there was no clear reason to use one method over the other; so, the genetic algorithm was chosen for its biological and evolutionary appeal.

This section describes the GA methodology to solve the problems (3.5 & 3.6). Genetic algorithms in general are discussed in the first part, with a short overview on the structure and basic algorithms. Following parts deal with genetic operators: initialization, discretization and binary coding, selection, recombination, mutation, and regeneration. For an excellent in-depth treatment of the subject of GA's, Goldberg's book [33] is highly recommended.

3.2.1 Brief Description of Genetic Algorithms

Genetic Algorithms are search algorithms that mimic the behavior of natural selection to find the global optimum point in a given design space. They operate on a population of potential solutions applying the principle of survival of the fittest to produce better and better approximations to a solution. At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the design space and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation. Evolutionary algorithms model natural processes, such as selection, crossover, and mutation. Figure 3.2 shows the structure of a simple genetic algorithm.

Evolutionary algorithms work on populations of individuals instead of single solutions. In this way, the search is performed in a parallel manner. Therefore, GA may climb many peaks in parallel thus; the probability of finding a local peak instead of the global is reduced significantly with GAs as compared with the conventional methods that search from a point to another point like the gradient-based methods.

A genetic algorithm operates on the Darwinian principle of "survival of the fittest". At the beginning, an initial *population* (a number of individuals) is created from random design parameters. Each parameter set represents the *individual*'s chromosomes. Each of the individuals is assigned a *fitness* based on how well each individual's chromosomes allow it to perform in its environment, computationally equal to their objective function value.

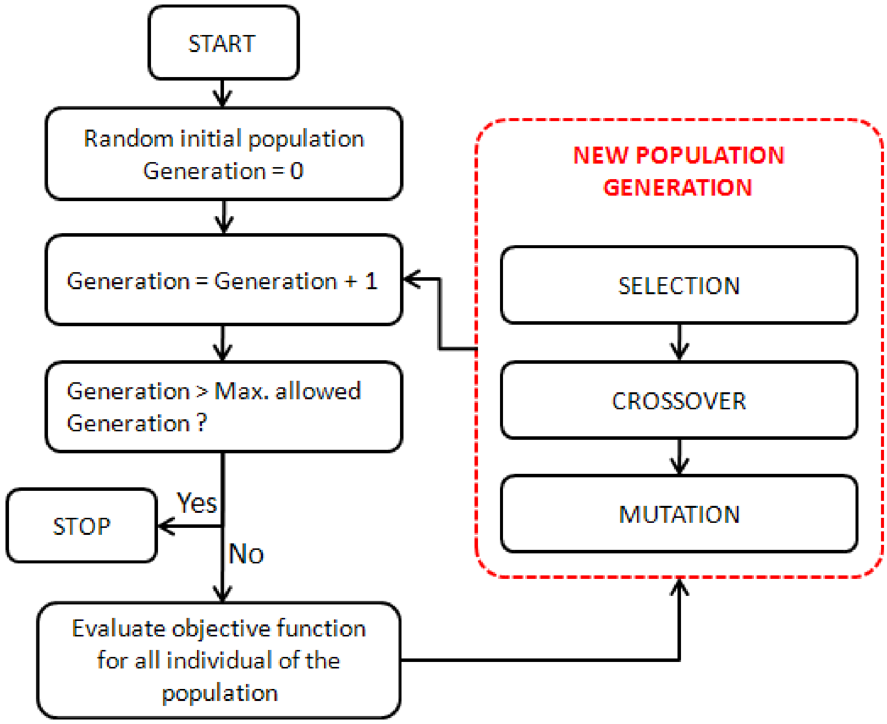


Figure 3.2 Structure of a single population genetic algorithm

If the optimization criteria are not met, the creation of a new generation starts. Individuals are selected according to their fitness for the production of children. Parents are recombined to produce children. All children will be mutated with a certain probability. The fitness of the children is then computed. They are inserted into the population replacing the parents, producing a new generation. This cycle is performed until the optimization criteria are reached.

From the above discussion, it can be seen that genetic algorithms differ substantially from more traditional search and optimization methods:

- GAs search a population of points in parallel, not a single point.
- GAs do not require derivative information or other auxiliary knowledge; only the objective function and corresponding fitness levels influence the directions of search.
- GAs use probabilistic transition rules, not deterministic ones.
- GAs are generally more straightforward to apply.
- GAs can provide a number of potential solutions to a given problem. The final choice is left to the user. Thus, in cases where the particular problem does not have one individual solution, as in the case of multi-objective optimization, then the genetic algorithm is potentially useful for identifying these alternative solutions simultaneously.

Similar to the above structure of genetic algorithms, a GA is developed based on the Fortran source code of Carroll [61]. The optimization algorithm modifies the UH rotor blade by changing the optimization parameters. Then, the RANS flow solver calculates the objective function value –either the efficiency or the torque- and sends it back to the GA. Accordingly, GA uses this value to assign a fitness value for the blade geometry. The corresponding automated optimization flow chart is given in Figure 3.3.

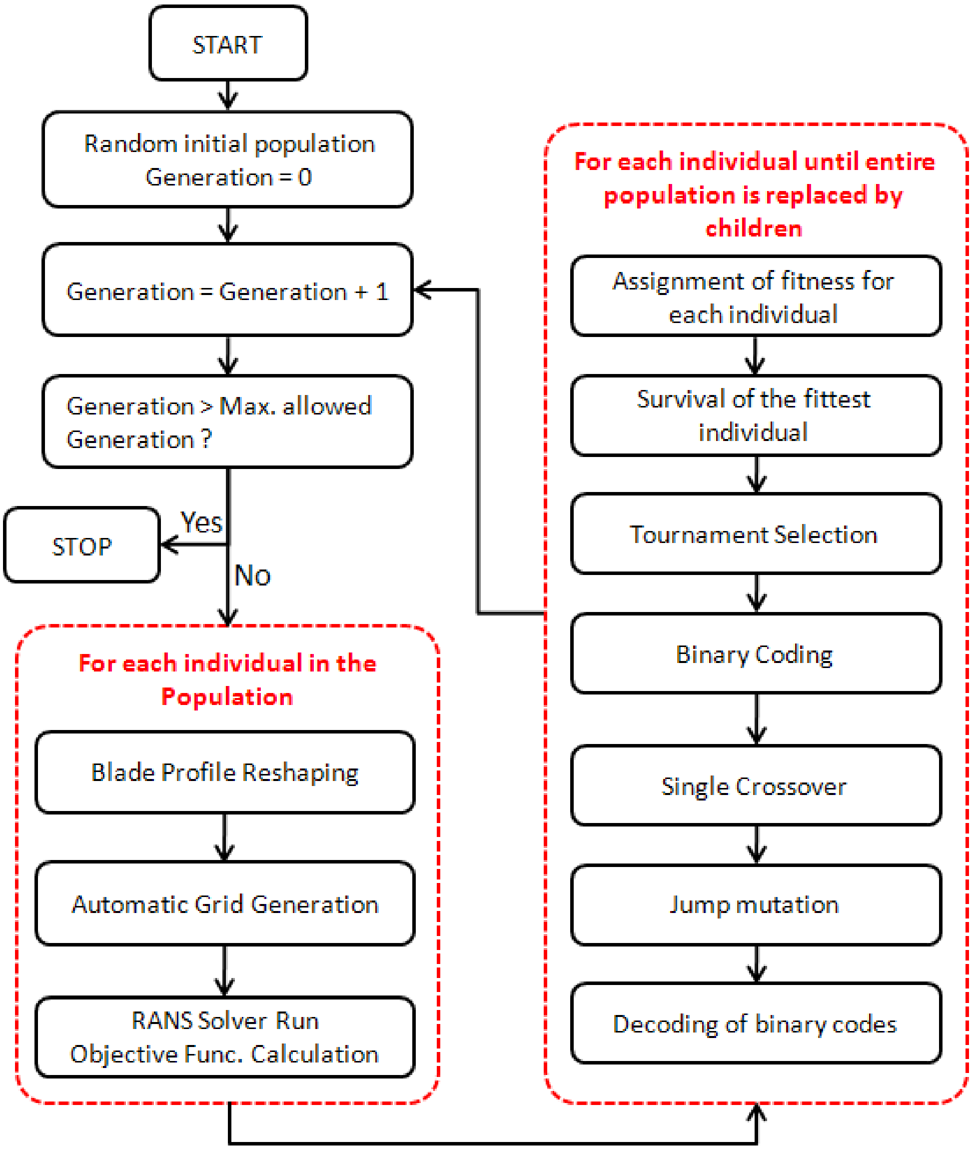


Figure 3.3 Automated GA optimization flow chart

3.2.2 Initial Population

Unlike other optimization tools, GAs require an initial population of solutions (individuals) in the design space, before the application of GA operators. Initiation of this population begins with selection of two GA parameters. They are mainly the population size and the chromosome length.

A population size of 5 individuals will be very insufficient for most of the engineering problems since the diversity in the design space will not be obtained. However, increasing the population size will increase the computational cost incurred on the CPU. As the population size increases, number of objective function evaluation increases. Therefore, there is a big trade of between the population size and computational time. Frequently, population sizes of 50 to 200 individuals are used in the literature [62].

Chromosome length selection is another tradeoff between the computational cost and diversity. A very short chromosome means a lack of diversity of the solutions and possible a worse optimization result. However, longer lengths cause a decrease in the convergence speed of the population towards the optimal solution. Mostly 16 or 32 bits chromosomes are used.

Specifying the population size and chromosome length, the GA code becomes ready to create first population which is called a *generation*. For illustration purposes, a population size of 10 individuals will be created randomly and a chromosome length of 20 bits will be assigned for each individual for an optimization problem with 4 design variables; inlet and exit angles, leading and trailing edges wedge angle values. This means that 5 bits will correspond to a design variable. These settings will be used throughout in this chapter to clearly illustrate how the GA code process. While using these operators, the GA generates random numbers, which is the essence of the method. In this application, random numbers were generated using Knuth's subtractive method [63]. Knuth's algorithm is regarded as one of the best random number generators.

In this thesis, as mentioned before, each variable set defines a blade configuration with a corresponding chromosome. The real value of each design variable is expressed as a string of binary digits, which is called as *binary coding*, e.g.: 101101. The associated chromosome for a specific blade is formed by placing the binary digits corresponding to each variable back to back in one string. For example, if blade inlet and exit angles, leading and trailing edges wedge angle values were binary coded as 001100, 010101, 001011, and 110001 respectively, and then the chromosome string would be 001100010101001011110001. The binary coding of ten blades (individuals) is shown in Table 3.4.

Table 3.4 Initial generation

GENERATION 1

Ind	Exit Angle	LE Wedge Angle	TE Wedge Angle	Inlet Angle	Binary Coding	Fitness Value
1	-38.39	2.500	0.895	31.29	10001111110101000100	0.065
2	-36.13	1.281	0.444	31.29	10000011010001100100	0.000
3	-38.39	1.755	2.056	39.03	10001101001110001100	0.000
4	-56.45	1.077	1.863	38.71	11001010101100101011	0.000
5	-22.58	2.161	0.895	34.52	01010110100101001110	0.030
6	-11.29	2.229	2.185	35.48	00101110111111010001	0.000
7	-65.48	2.500	0.250	35.81	11101111110000010010	0.000
8	-33.87	1.823	0.895	39.03	01111101010101011100	0.051
9	-40.65	2.297	1.927	36.45	10010111001101000100	0.000
10	-47.42	1.958	1.347	35.81	10101101111000100010	0.098
					Average function value	**0.025**
					Maximum function value	**0.098**

3.2.3 Discretization

At this point, one may question that how can a number such as –38.39 binary coded? The answer lies in the explanation of *discretization*. The design space is virtually continuous and there are infinite numbers of possible blade configurations. However, in practice, this design space needs to be discretized, so that there are a finite number of possible blades in the design space, but yet too many to choose the optimum one. For the illustrative setting of 20 bits for each chromosome consisting of four design variables, a design variable is specified by five bits (0 or 1). Therefore, for this setting, a design variable can have $2^5=32$ possible values. For example, the exit angle in Table 3.4 is constrained to be

$$0° > \alpha_2 > -70° \tag{3.7}$$

Using this expression, exit angle can have infinite numbers between 0 and –70. Discretizing the variable space using five bits yields

$$0 > \alpha_2 = I\frac{\Delta S}{2^n - 1} = I\frac{70-0}{2^5-1} \approx 2.258I > -70$$
$$I=-1, -2, \ldots, -32 \tag{3.8}$$

where ΔS is the variable space length and n is the number of bits of the design variable. Accordingly, exit angle has random values corresponds to I = -17, -16, -17, -25, -10, -5, -29, -15, -18, -21 in the first generation given in Table 3.4. Note that the minus sign of the integer I is specific to this design variable and do not exist in other design variables.

It is clear that increasing the number of bits in a chromosome increases the number of possibilities of a design variable value; therefore more accurate search on the design space can be established. However, as mentioned, this increases the search time. For the illustrative settings, each design variable has 32 values in the design space, which means that there are a possible of $32^4 \cong 1$ million blade shapes to be searched. The optimum blade is one of these 1 million blades in this simple discretized problem.

3.2.4 Binary Coding

Binary coding is followed after discretization. While discretization, integer numbers such as *I* in Equation 3.8 are obtained for each blade configurations. Thereafter, base 10 is converted to base two. Considering, for instance individual #1 in Table 3.4, if I = 17 for exit angle and, 32, 10, 4 for LE and TE wedge and inlet angle respectively, then

$$(17)_{10} = (10001)_2 \tag{3.9.a}$$

$$(32)_{10} = (11111)_2 \tag{3.9.b}$$

$$(10)_{10} = (01010)_2 \tag{3.9.c}$$

$$(04)_{10} = (00100)_2 \tag{3.9.d}$$

Putting simply these digits back to back, the chromosome of the first blade is obtained as 10001111110101000100. While numerical values of design variables have no meaning for GA, chromosome of the blade has all meaning for GA. Now, GA can operate on these chromosomes.

3.2.5 Selection

In selection, the parents producing children are chosen. Selection determines which individuals are chosen for mating (recombination) and how many children each selected individual produces. Methods of fitness assignment are:

- proportional fitness assignment [60],
- rank-based fitness assignment [64].

In proportional fitness assignment, the fitness is proportional to objective values and normalized to the unity. In the rank based assignment, on the other hand, the population is sorted according to the objective values. The fitness assigned to each individual depends only on its position in the individuals rank and not on the actual objective value. These fitness values are calculated and given in Table 3.5. The GA code in this thesis assigns fitness of the blades directly equal to the objective function value, as shown in Table 3.5. There is no clear rule of thumb in fitness assignment in literature and since the fitness is the performance of a blade in a given environment (flow field in a turbine stage), the direct assignment is considered to be the most suitable way.

Selection, which is also called *reproduction*, is a process in which blades are selected according to their fitness values. This implies that a blade with a higher fitness value (objective function value) has a higher probability of contributing in the next generation. Generally, each individual in the selection pool receives a reproduction probability depending on the fitness value of its own and of all other individuals in the selection. This probability is used for the actual selection step afterwards. There are several selection procedures in the literature:

- roulette-wheel selection [65],
- stochastic universal sampling [65],
- local selection [66], [67],
- truncation selection [68],
- tournament selection [69].

Table 3.5 Various fitness assignments

GENERATION 1

Ind	Objective Function	Proportional Fitness	Rank-based Fitness	Direct Fitness
1	0.065	0.266	9	0.065
2	0.000	0	0	0.000
3	0.000	0	0	0.000
4	0.000	0	0	0.000
5	0.030	0.123	7	0.030
6	0.000	0	0	0.000
7	0.000	0	0	0.000
8	0.051	0.209	8	0.051
9	0.000	0	0	0.000
10	0.098	0.402	10	0.098
Average fitness		0.1	3.4	**0.0245**
Maximum fitness		0.402	10	**0.098**

It is worth discussing only roulette-wheel selection and tournament selection methods and comparing them to clarify selection procedure and its different applications.

The simplest selection scheme is *roulette-wheel selection*, also called stochastic sampling with replacement. In this stochastic algorithm, the individuals are mapped to contiguous segments of a line, such that each individual's segment is equal in size to its fitness. A random number is generated and the individual whose segment spans the random number is selected and copied to *mating pool*. The process is repeated until the desired number of individuals is obtained in the mating pool. This technique is analogous to a roulette wheel with each slice proportional in size to the fitness, and the wheel is rotated so that an individual is selected. Figure 3.4 shows probabilities corresponding to direct fitness assignment of the 1st generation.

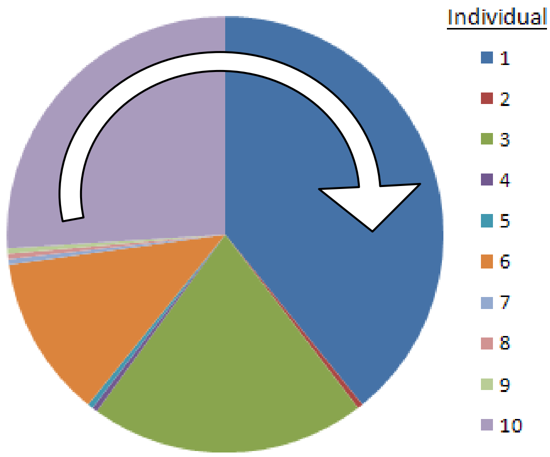


Figure 3.4 Illustration of Roulette Wheel selection

Instead of roulette-wheel selection, present GA code utilizes *tournament selection*. Because, Goldberg and Deb [70] showed that tournament selection has better or equivalent convergence and computational time complexity properties when compared to any other selection operator in the literature. In tournament selection, after assigning the fitness to each blade configuration, selection for mating is performed [20]. A *pair* of blades is selected randomly from the population and the "better" (fitter) blade is assigned to be first parent. Then, again randomly two *other* blades chosen from the population and compared, the better is assigned as the second parent. Once two parents are determined, a new individual, called as *child*, who has some mix of the two parents' chromosomes, is created using crossover and mutation operators. This cycle is repeated until a new population is generated form children as illustrated in Figure 3.5.

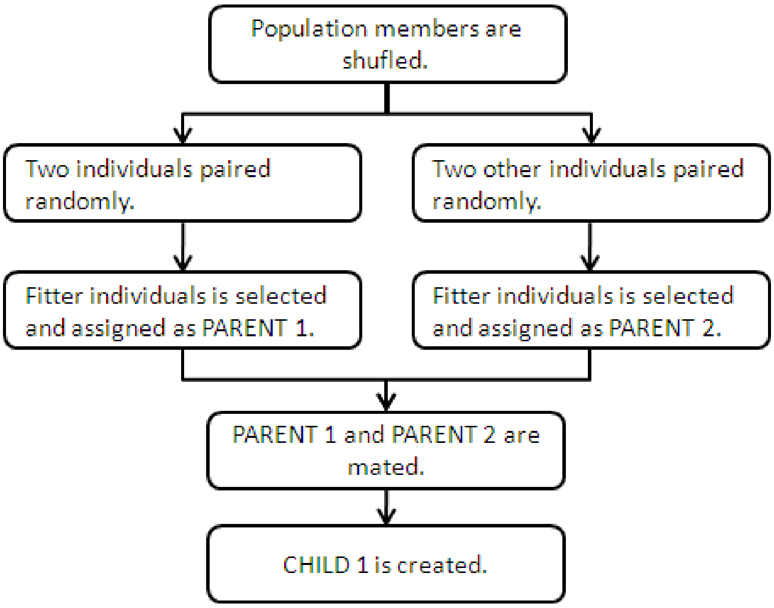


Figure 3.5 Tournament selection methodology, illustrating the first child creation

In this thesis, twice of the population size tournaments are performed, because two parents will be used to create one child. Note that each individual, therefore, will participate to four tournaments.

Table 3.6 is the selection performed on the 1st generation. Randomly generated pairs are compared and individuals with higher fitness are mated to create ten children for the new generation. Detailed mating of parents is discussed in the crossover and mutation section. It is interesting to note that how better blades with high fitness values have made themselves to exist in many mating slots in Table 3.6. This is precisely the purpose of a selection operator. An interesting aspect of tournament selection operator is that just by changing the comparison operator ("better" to "worse"), the minimization and maximization problems can be handled easily. One other beauty of the selection is that even the worst fit individual such as Ind # 6 may exist in the mating so that the diversity of the population can be obtained.

Table 3.6 Selection scheme

Ind	Fitness	Random Tournaments	Winner (Fitter)	Mating Parents	Child to be created
1	0.065	Ind # 4 & Ind # 6	Ind # 6	Ind # 6 & Ind # 10	Child #1
2	0.000	Ind # 8 & Ind # 10	Ind # 10		
3	0.000	Ind # 9 & Ind # 3	Ind # 3	Ind # 3 & Ind # 1	Child #2
4	0.000	Ind # 1 & Ind # 7	Ind # 1		
5	0.030	Ind # 2 & Ind # 4	Ind # 4	Ind # 4 & Ind # 10	Child #3
6	0.000	Ind # 10 & Ind # 5	Ind # 10		
7	0.000	Ind # 1 & Ind # 3	Ind # 1	Ind # 1 & Ind # 9	Child #4
8	0.051	Ind # 7 & Ind # 9	Ind # 9		
9	0.000	Ind # 6 & Ind # 2	Ind # 2	Ind # 2 & Ind # 8	Child #5
10	0.098	Ind # 8 & Ind # 4	Ind # 8		
-	-	Ind # 4 & Ind # 7	Ind # 7	Ind # 7 & Ind # 10	Child #6
-	-	Ind # 10 & Ind # 1	Ind # 10		
-	-	Ind # 3 & Ind # 8	Ind # 8	Ind # 8 & Ind # 5	Child #7
-	-	Ind # 2 & Ind # 5	Ind # 5		
-	-	Ind # 4 & Ind # 9	Ind # 4	Ind # 4 & Ind # 10	Child #8
-	-	Ind # 10 & Ind # 8	Ind # 10		
-	-	Ind # 7 & Ind # 3	Ind # 3	Ind # 3 & Ind # 5	Child #9
-	-	Ind # 5 & Ind # 2	Ind # 5		
-	-	Ind # 1 & Ind # 6	Ind # 1	Ind # 1 & Ind # 4	Child #10
-	-	Ind # 9 & Ind # 4	Ind # 4		

3.2.6 Crossover

Once the parents to be mated are determined, crossover is applied to the parents according to a specified probability previously. Noting that selection operator does not produce any new solutions (individuals), but only makes more copies of solutions to be used for new generation at the expense of not-so-good solutions. The creation of the child is performed only by crossover and mutation operators.

Like selection, there exist many crossover operator types in the GA literature [71]. Mostly, some portion of the chromosomes of parents is exchanged to create a (or two) new child. Two important crossover operators widely used are single point and uniform crossover. Only single point crossover is applied in this thesis.

In *single point crossover*, a crossing site is randomly chosen. All bits on the right hand side of the parents are exchanged. This can be easily illustrated in an example, recalling the first generation given in Table 3.4, and the selection performed in Table 3.6. The crossover probability is set to 50%, therefore, if a randomly generated number is less than 0.5, the crossover operates; otherwise, the first parent is copied exactly as a child. Running the GA code, the crossover did not operate on the first two rows of mating parents in Table 3.6. In the third row of mating parents, we have two parents namely Ind #4 and Ind #10. The code performed single point crossover on the parents as illustrated in Figure 3.6. The crossing site is again determined randomly as on the 9th bit.

Note that the above crossover operator created two solutions. In this thesis, only one child is created from two parents. However, the second child is shown in Figure 3.6 for illustration purposes. The crossover operator created C#1, a blade with objective function value is equal to 0.168. This fitness value is better than both of the parents are. One may wonder if a different crossover site were chosen or two other parents were chosen, whether we would have found a better of the parents every time? It is true that every crossover between two parents is not likely to find child better than both parent solution, but it will be clear in a while that the chance of creating better solutions is far better than random. This is true because the parent chromosomes being crossed are not any two arbitrary random chromosomes. These parents have *survived* tournaments played with other solutions during the selection operator. Thus, they have some good bit combinations in their chromosome representations.

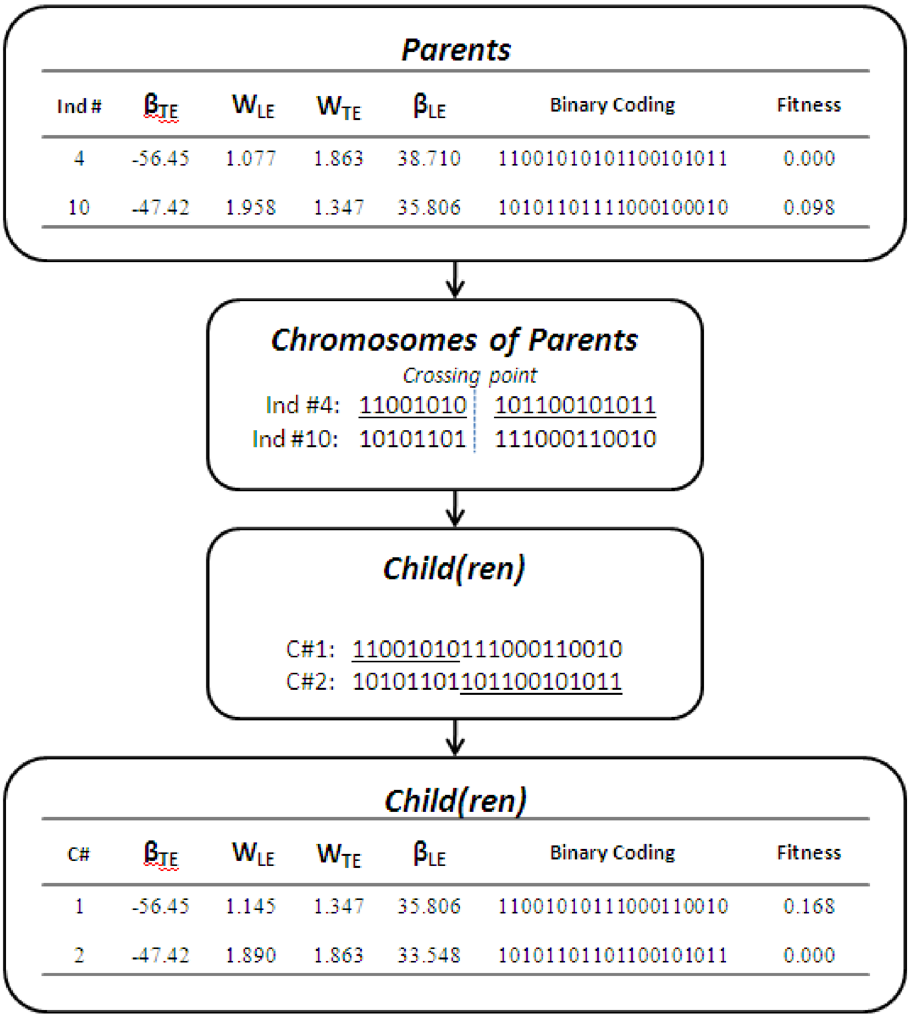


Figure 3.6 Illustration of single point crossover operator

Moreover, not every crossover may produce better solutions, but if so, they will be eliminated in the next selection, hence, have a shorter life. Nevertheless, they are useful since they preserve the diversity on the design space. Additionally, the crossover probability preserves some good parents. For the illustrated example, a 50% probability means that, half of the parents will be reproduced again in the next generation, by simply copying. The single point crossover results for the first generation are tabulated in Table 3.7 with specific crossing point.

Table 3.7 Single point crossover results

P#	Fitness	Chromosomes	Cr. site	C #	New individual (child)	Fitness
6	0.000	00101110111111010001	-	1	00101110111111010001	0.000
10	0.098	10101101111000110010	-			
3	0.000	10001101001110011100	-	2	10001101001110011100	0.000
1	0.065	10001111110101000100	-			
4	0.000	11001010101100111011	9	3	11001010111000110010	0.168
10	0.098	10101101111000110010	9			
1	0.065	10001111110101000100	3	4	10010111001101010100	0.000
9	0.000	10010111001101010100	3			
2	0.000	10000011010001100100	5	5	1000***0***101010101011***0***00	0.057
8	0.051	01111101010101011100	5			
7	0.000	11101111110000010010	-	6	11101111110000010010	0.000
10	0.098	10101101111000110010	-			
8	0.051	01111101010101011100	8	7	01111100100101001110	0.050
5	0.030	01010110100101001110	8			
4	0.000	11001010101100111011	-	8	11001010101100111011	0.000
10	0.098	10101101111000110010	-			
3	0.000	10001101001110011100	10	9	10001101000101001110	0.062
5	0.030	01010110100101001110	10			
1	0.065	10001111110101000100	-	10	10001111110101***1***00100	0.047
4	0.000	11001010101100111011	-			

In the case of *uniform crossover*, one child is constructed by choosing every bit with a probability from either parent, as shown in Figure 3.7. For this example, again a 50 % probability is used. In this example, 2nd, 3rd, 5th, 7th, 9th, 10th, 12th, 14th, 15th, and 18th bits are exchanged between the parents and two children are created.

In terms of the extent of search power of a crossover operator, a single point crossover preserves the structure of the parent strings to the maximum extent in the children. The extent of chromosome preservation reduces with increasing crossing sites, and is minimized in the case of a uniform crossover operator. Spears [71] has analyzed a number of these operators and has shown this fact.

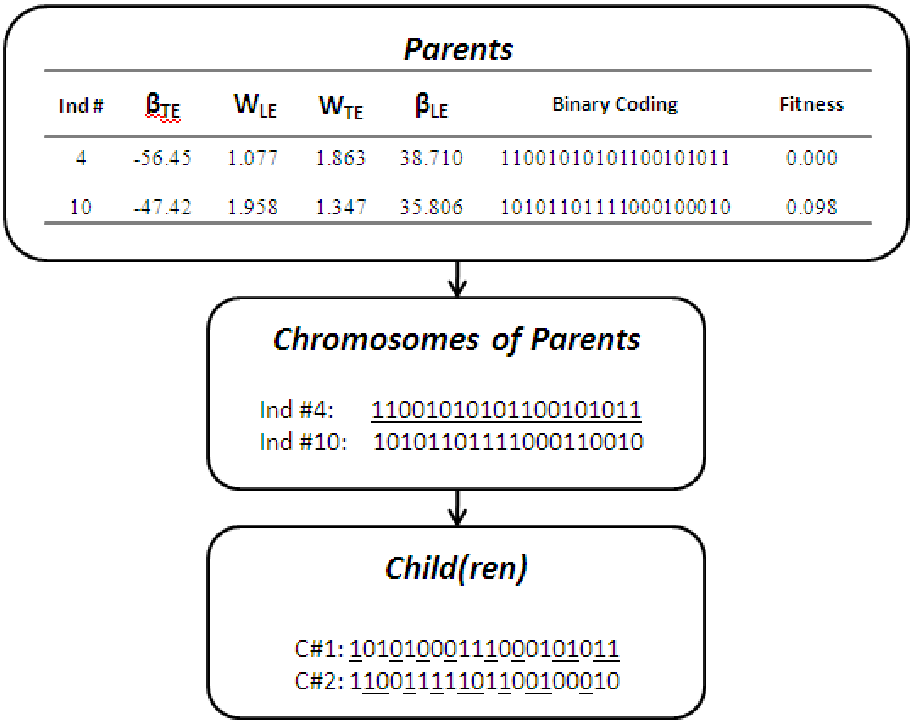


Figure 3.7 Illustration of uniform crossover

3.2.7 Mutation

Mutation is a bit change of a chromosome that occurs during the crossover process. Mutation implies a random walk through the string space and plays a secondary role in the GA: keep diversity in the population. The operator changes a bit from 1 to 0 or 0 to 1 according to a given probability. If a random number is less than the given probability, mutation occurs. For example, a child may be faced with a mutation like $C'_1 : 1\boldsymbol{\mathit{0}}100111 \Rightarrow C'_1 : 1\boldsymbol{\mathit{1}}100111$ on the second bit of its chromosome. The mutations of the first generation are shown bold and italic in Table 3.7. Here, the probability was set to 2 %. Accordingly, ten children with 20 bit each means 200 bits. With that probability, only three bits were mutated.

3.2.8 Regeneration

After the creation of children using crossover and mutation operators, they are used to replace old population consisting of their parents. All of the old population members are deleted and all of the children become a new population called as a new *generation*.

One may wonder about the best individual of a population. Elitism (i.e. best individual replicated into next generation) may be invoked in the GA code. This ensures the survival of the best individual from each generation. Therefore, after the population is generated, the GA checks to see if the best parent has been replicated; if not, then a random individual is chosen and the chromosome set of the best parent is mapped into that individual. Although this operator is not necessary, it was found to help prevent the random loss of good chromosome strings. Therefore, at each generation, the best individual is either the same of or better than the best individual of the previous generation.

In the first generation of the illustrated problem, crossover and mutation operators discarded the best individual, Ind #10. For that reason, invoking the elitism, the GA selected the 10th child and replaced it by the old super-individual. The final population of the new generation is given in Table 3.8. Note that the blades now have better fitness in average, and GA succeeded to find a better solution, Ind #3.

There are various stopping criteria for GAs. In this thesis, the stopping criterion is the maximum generation number. Sufficient generations are selected and stated, and the code checks whether this number is achieved or not. Typically, 100 generations are enough for an optimization run.

Table 3.8 An evolved population of the 2nd generation

GENERATION 2

Ind	Exit Angle	LE Wedge Angle	TE. Wedge Angle	Inlet Angle	Binary Coding	Fitness
1	-11.29	2.229	2.185	35.48	00101110111111010001	0.000
2	-38.39	1.755	2.056	39.03	10001101001110011100	0.000
3	-56.45	1.145	1.347	35.81	11001010111000110010	0.168
4	-40.65	2.297	1.927	36.45	10010111001101010100	0.000
5	-36.13	1.823	0.895	37.74	10000101010101011000	0.057
6	-65.48	2.5	0.25	35.81	11101111110000010010	0.000
7	-33.87	1.619	0.895	34.52	01111100100101001110	0.050
8	-56.45	1.077	1.863	38.71	11001010101100111011	0.000
9	-38.39	1.755	0.895	34.52	10001101000101001110	0.062
10	-47.42	1.958	1.347	35.81	10101101111000110010	0.098
					Average function value	**0.043**
					Maximum function value	**0.168**

3.2.9 Understanding GAs

Comparing Table 3.4 and 3.8, an improvement from the initial population can easily be recognized. Selection, crossover and mutation operators completed one generation of the GA simulation and average objective function values of population members increased more than 75%. Moreover, the objective function value of the best individual of the second generation is around 72% higher than that of the first generation.

Even though all operators used random numbers, GA produces a directed search, which usually results in an increase in the average quality of solutions from one generation to the next. It is true for the fitness of best individual in a population, too.

If we investigate carefully among the bit groups of the two populations, some similarities in chromosome positions can be observed. By the application of three GA operators, the number of bit groups with similarities at certain chromosome positions has been increased from the initial population to the new population. These similarities, called *schema*, represents a set of bits with certain similarities at certain chromosome positions. Schemas represented by 1, 0, and *; where * represents (1 or 0). The order and length of schema may vary in different analyses. For example the schema

H_1=(* * * * * * * * * * * * * * * * 0 0 1 0)

is represented two times, Ind #7 and Ind # 10 in the initial population, and there are three chromosomes represented by this schema in the second generation; Ind #3, Ind #6, Ind #10. On the other hand, even though there was one chromosome corresponding to schema

H_2=(* * * * * 0 1 1 0 1 0 * * * * * * * * *) in the initial population, there is not one in the new population. There are some other schemas that may be investigated and concluded whether the number of chromosomes they represent is increased from the initial population to the new population or not.

A schema represents a number of similar chromosomes. Thus, a schema can be thought of as representing a certain region in the design space. For example, schema H_1 represents blades with inlet angle of either 30.645^o or 35.806^o with any combination of exit, LE and TE wedge angles. Similarly, H_2 represents blades with 1.281 LE wedge angle, TE wedge angle values varying from 0.24 to 1.218, and any other inlet and exit angles. Since our aim is to maximize the objective function, GA increased the number of H_1 schema and decreased the number of H_2 schema without having to count all schemas in whole design space. That way, GA succeeded to find a better blade.

While GA operators are applied on a population of chromosomes, a number of such schema in various parts along the chromosome are emphasized. Once adequate number of such schemas is present in a GA population, they are combined together due to the action of the GA operators to form better schema. This process finally leads a GA to find the optimal solution. This hypothesis is known as the *Building Block Hypothesis* [20].

3.3 Multiploid Genetic Algorithm Optimization (GAXL)

Although, GAs are able to converge to the global optimum, they consume considerable time to converge to the global optimum for a 3D aerodynamic design problem which requires expensive high-fidelity RANS solver. This drawback of genetic algorithms is avoided by using faster but less accurate low-fidelity models. In this study, the high fidelity model corresponds to fine grid RANS solution of the flow field, and coarse grid RANS solutions corresponds to the low fidelity model.

Since aerodynamic optimization problems can be solved for different quality levels of mesh, the expensive aerodynamic optimization problems are very suitable for multi-fidelity optimization. Consequently, the novel method developed in this thesis involves a modified genetic algorithm, which is titled as "multiploid", and is able to interpret the high-fidelity, high quality, very expensive information such as a fine mesh solution of the blade, and the low-fidelity, low quality inexpensive information such as a coarse mesh blade solution, and does not degrade the quality of the optimization result by mixing the information of multi-fidelity solutions.

3.3.1 Brief Description of Multiploid Genetic Algorithm

A *haploid* genetic structure is one in which the genotype is composed of a single chromosome. The GA discussed previously is a haploid genotype. However, *multiploid* –mostly *diploid*- genotype also exists in nature which contains two (diploid) or more sets of single chromosomes. Although the reason of existence of multiploid genotype in nature might be different than the reason of this study, it will be shown that multiploid genotype can efficiently be used for much faster converging optimization algorithms.

In the literature, there exist many studies regarding application of surrogate assisted evolutionary optimization for solving computationally expensive design problems on a limited computational budget [72]. Most of these studies consist of building a low cost less accurate model of the expensive fitness from a small number of data points that represents the design space. The model is then exploited by the evolutionary algorithm as an auxiliary fitness function in order to get the maximum amount of information out of these initial data points. Subsequently, the model is updated online based on new data points obtained during the surrogate-assisted evolutionary search [73], [74], [75]. For a review of existing surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems, the reader may refer to [76].

Earlier research efforts related to evolutionary optimization have focused on the use of problem specific knowledge to increase the computational efficiency. Oksuz et al. [26] used Euler/Boundary layer coupled flow solver instead of RANS solver to increase the computational efficiency to satisfy the limited computational budget. Even though such problem specific heuristics can be effectively used to achieve performance improvements, there are finite limits to the design improvements achievable by such techniques, since the lower cost solvers are not accurate enough.

Several efforts have been made over recent years, particularly using GAs, applying surrogate models of accurate solvers. The most popular ones are response surface methodology, the Kriging model, design of experiments, neural networks, and the support vector machines. Ratle [77] examined a strategy for integrating GAs with Kriging models. This work uses a heuristic convergence criterion to determine when an approximate model must be updated. The same problem was revisited by El-Beltagy et al. [78], where the issue of balancing the design of experiments is addressed. Jin et al. [79] presented a framework for coupling GAs and neural network-based surrogate models. This approach uses both the expensive and approximate models throughout the search, with an empirical criterion to decide the frequency at which each model should be used. In Song [80], a real-coded GA was coupled with Kriging in structural optimization. A recent study of Pierret [48] presents the design of turbomachinery blades by means of function approximation. The concept is based on the use of online trained Artificial Neural Network (ANN) as a surrogate model of RANS solver with GA optimization algorithm.

Evolutionary algorithms, especially GAs, face a big convergence problem when solving dynamic optimization problems (DOPs) [81]. Convergence deprives GAs' adaptability to the changing environment: Once converged, they are unable to adapt to the new environment when change occurs. In order to enhance the performance of GAs for DOPs, several approaches have been developed [82], such as random immigrants [83], [84], hyper-mutation [85], memory [86], [87], [88], [89], [90], and multi-population schemes [91]. Further literature review about the subject can be found in [81]. All of these studies prove the fact that GAs are not suitable as it is condition for dynamically changing environments without any modification to GA itself. The main reason behind this is, once the GA converges, it cannot adapt to the changing environment. Therefore, optimization methods proposed such as in [48] has the problem of convergence. In this specific case, the ANN is the dynamic environment, which is trained online with each new generation. Meanwhile, a simple GA is used to find the optimum of this DOP. As stated before, the GA will not follow the changing ANN environment, won't be able to adapt and will stuck an optimum of an older environment.

Although the multiploid GAs have been integrated successfully into traditional GAs [92], [93], [94], the main aim was to enhance their performance for DOPs. However, the aerodynamic optimization in this problem differs from the above optimization problems in the fact that:

- the surrogate model already exists, and
- the problem environment is not dynamic.

Consequently, in order to use limited computational budget efficiently, without degrading the optimization quality, existing surrogate model is implemented with a modified version of a GA. First, the best surrogate model available for the aerodynamic performance prediction of turbine blade is selected as the coarse mesh solution. Then, GA is modified to yield multiploid genetic representation in order not to fool GA with changing fitness environment. Finally, the optimization is performed on multi-fidelity fitness values.

Figure 3.8 shows the flowchart of a multiploid multi-fidelity genetic algorithm, GAXL in short notation. The XL abbreviation stands for "multi-level" fitness assignment. A higher level fitness means that the objective function value is calculated using a higher-fidelity model. On the contrary, a lower level fitness corresponds to a lower-fidelity model calculated value of the objective function.

The operators are the same for both GA and GAXL, although there are main modifications to selection and crossover operators of GA to handle multi-fidelity objective function values.

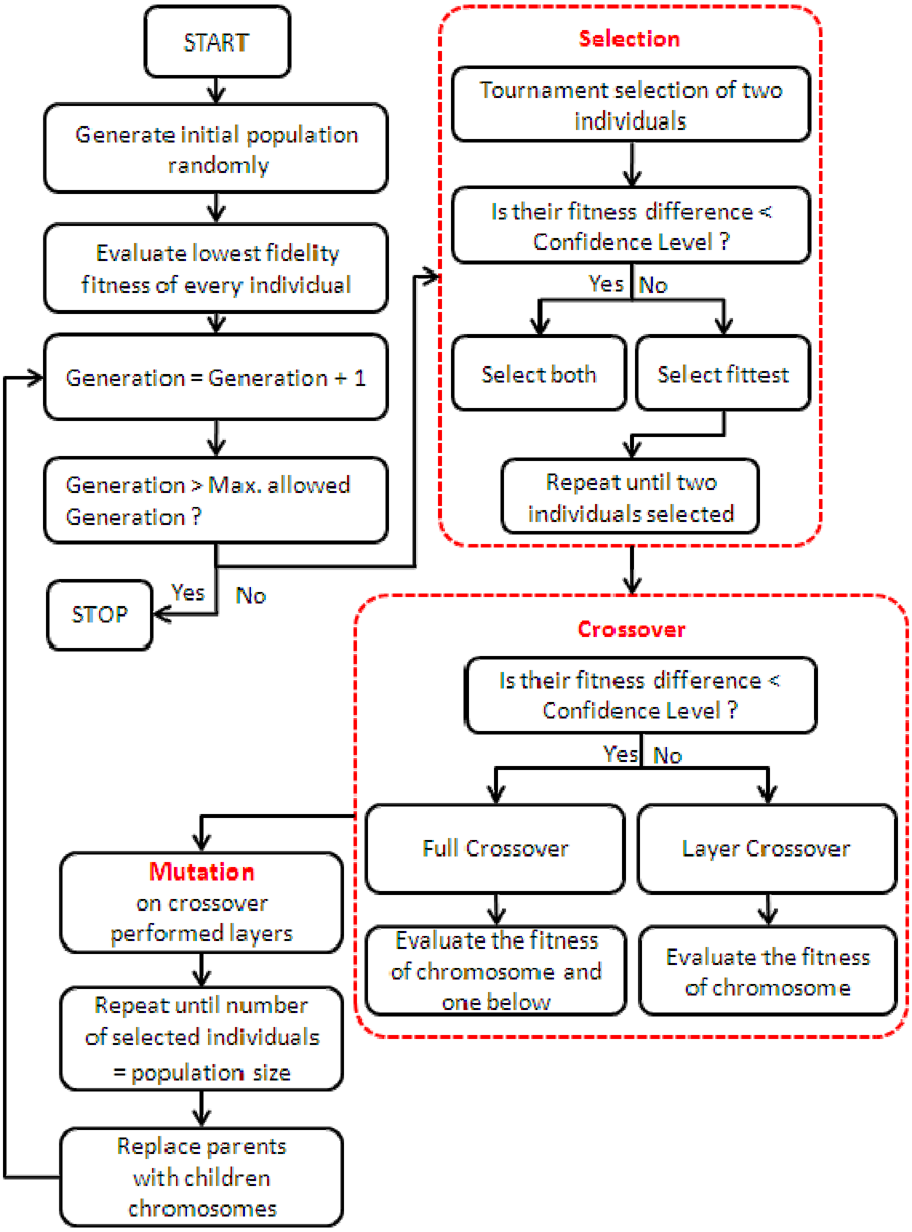


Figure 3.8 Flowchart of GAXL

From the above discussion, it can be seen that GAXL differ substantially from more traditional GA:

- GAXL has multiploid genetic representation, i.e. more than one chromosome for each individual, whereas GA has only one.
- GAXL assigns multi-fidelity fitness values of an individual whereas GA assigns only one –highest fidelity- fitness.
- Although tournament selection is common, GAXL has a pre-defined confidence level parameter which sets the decision base for the selection.
- Although single point crossover is common, GAXL implements two different kinds of crossover, namely full crossover, and level crossover.
- Although jump mutation method is common, GAXL mutates only crossover performed chromosomes.

Similar to the above structure of multiploid genetic algorithms, a GAXL is developed based on the GA in section 3.2. Again, for comparison with GA, the optimization algorithm modifies the UH rotor blade by changing the optimization parameters. Then, GAXL requests either the low fidelity or the high fidelity fitness value from the RANS flow solver. Coarse mesh or fine mesh, respectively, are generated and solved to calculate the objective function value –the efficiency or the torque- and assigned to fitness of the correct level of chromosomes. Accordingly, GAXL interprets these values to create a new generation. The corresponding automated optimization flow chart is given in Figure 3.9.

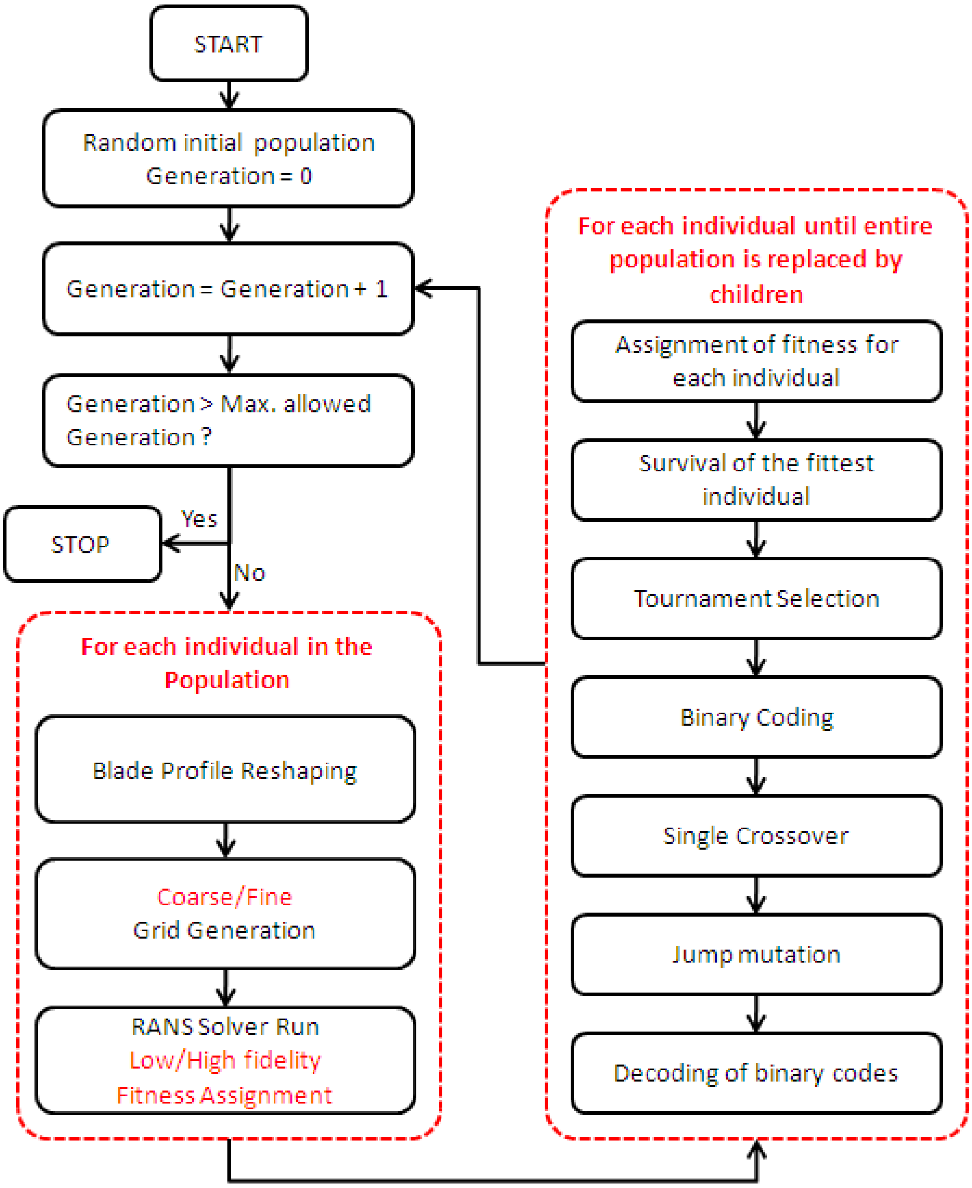


Figure 3.9 Automated GAXL optimization flow chart

3.3.2 Initial Population

Similar to GA, GAXL require an initial population of solutions (individuals) in the design space, before the application of GAXL operators. Initiation of this population begins with selection of two GA and one GAXL parameters. They are mainly the population size and the chromosome length, and number of chromosomes i.e. number of fitness levels, respectively.

In this study, only diploid genetic representation which consists of two chromosomes is used. Every method presented can be applied to multiploid presentations. A population size of 10 individuals is used for illustration purposes, which is the same size with the GA. Chromosome length of the example is selected as 20 bits, again consistent with GA.

First generation is created from 10 individuals randomly and twin chromosomes with a length of 20 bits are assigned for each individual for the same illustrative optimization problem given in section 3.2.2. These settings will be used remaining of this chapter to clearly illustrate how the GAXL code process. GAXL generates random numbers using the same method of GA, which is the Knuth's subtractive method [36].

In this thesis, as mentioned before, each variable set defines a blade configuration with two corresponding chromosomes. The real value of each design variable is expressed as a string of binary digits, which is called as *binary coding*, e.g.: 101101. The associated chromosomes for each blade are formed by placing the binary digits corresponding to each variable back to back in one string. For example, if blade inlet and exit angles, leading and trailing edges wedge angle values were binary coded as 001100, 010101, 001011, and 110001 respectively, and then one chromosome string would be 001100010101001011110001.

At the initial generation, the two chromosome strings are set equal to each other. The top chromosome string corresponds to a lower fidelity fitness value, and bottom chromosome string corresponds to higher fidelity. Consequently, for a diploid GAXL, the top chromosome can be interpreted as the geometry with coarse mesh, and the bottom chromosome can be interpreted as the same blade geometry with fine mesh. Note that only lowest fidelity fitness values are evaluated at the initial generation. The binary coding of ten blades (individuals) is shown in Table 3.9.

Table 3.9 GAXL Initial generation

GENERATION 1

Ind	Exit Angle	LE.Wedge Angle	TE.Wedge Angle	Inlet Angle	Binary Coding	Fitness Values
1	-38.39	2.500	0.895	31.29	10001111110101000100	0.061
					10001111110101000100	-
2	-36.13	1.281	0.444	31.29	10000011010001100100	0.000
					10000011010001100100	-
3	-38.39	1.755	2.056	39.03	10001101001110001100	0.000
					10001101001110001100	-
4	-56.45	1.077	1.863	38.71	11001010101100101011	0.000
					11001010101100101011	-
5	-22.58	2.161	0.895	34.52	01010110100101001110	0.022
					01010110100101001110	-
6	-11.29	2.229	2.185	35.49	00101110111111010001	0.000
					00101110111111010001	-
7	-65.48	2.500	0.250	35.81	11101111110000010010	0.000
					11101111110000010010	-
8	-33.87	1.823	0.895	39.03	01111101010101011100	0.055
					01111101010101011100	-
9	-40.65	2.297	1.927	36.45	10010111001101000100	0.000
					10010111001101000100	-
10	-47.42	1.958	1.347	35.81	10101101111000100010	0.090
					10101101111000100010	-

3.3.3 Discretization and Binary Coding

The design space of the illustrative optimization problem is discretized similar to GA example, i.e. each design variable can have 32 distinct values, which means that there is a possible of $32^4 \cong 1$ million blade shapes to be searched. The optimum blade is one of these 1 million blades in this simple discretized problem.

Binary coding is followed after discretization. At the first generation, in which all of the blades are randomly chosen, once a chromosome of a blade is obtained similar to GA, then it is copied as much as the number of levels - surrogate models - of GAXL. In the case of a diploid GAXL, low fidelity level top chromosome is binary coded from randomly chosen blade parameters. Then, the high fidelity level bottom chromosome is found by copying exactly the top chromosome. Finally, all of the blades in the first generation are randomly generated and then binary coded.

As the population evolves with each new generation, different level chromosomes depart from each other because of "level crossover" and mutation operators which will be discussed in the forthcoming sections.

3.3.4 Multi-fidelity Selection

One of the biggest disadvantages of GA is that although one can use a low-fidelity fitness model such as ANN or a coarse grid, the GA is NOT able to converge to the global optimum in changing environments. That is because assignment of low-fidelity fitness values to the chromosome of an individual at the initial generations, and then high-fidelity fitness values at later generations, the schemas mentioned in section 3.2.9 does not work for convergence to global optimum. The genes (binary digits) coming from ancestors fool the children's chromosomes with low-fidelity fitness values. For GA, the only operator to get rid of these low-fidelity well performing but high fidelity bad performing genes is mutation which has a very low probability to happen to an individual. Therefore, GA will eventually converge to a local optimum rather than the global.

On the contrary to optimizing the problem in a dynamic environment using GA, fitness values are separately assigned to each chromosome of an individual for each surrogate model calculation in GAXL. Each fitness value corresponds to either the exact or approximate solution of the objective function.

Therefore, there must be a method to select the fitter individual by comparing multi-fidelity fitness values. One will either combine the chromosomes and fitness values and create only one chromosome with one average fitness value, or will leave the chromosomes and fitness values as it is and define a method to select which fitness value to use for selection. The former fools the chromosomes as indicated in the previous paragraph, whereas the latter, which is used in GAXL, guides the chromosomes for faster convergence but never manipulates on fitness values.

The selection method which is developed for multiploid GAXL includes an additional input parameter, called as *confidence level*, CL. Expressed in percentage; the confidence level indicates the expected accuracy of the surrogate model used. A confidence level of 100% is assigned for the highest-fidelity model solution of an objective function. Therefore, the bottom level fitness values have 100% confidence level automatically.

On the other hand, lower percentages of confidence levels are assigned for a lower fidelity fitness levels. There might not exist a unique error percentage between high and low fidelity accuracy of the objective function values. Nevertheless, confidence level is set by finding the mean deviation of the calculated objective function values between surrogate and exact solutions, and can be expressed as:

$$CL \approx 1 - \frac{\left| f_{exact} - f_{surrogate} \right|}{f_{exact}} \tag{3.10}$$

This equation is calculated for few points in the design space and the average value can be set as the confidence level. Note that, each objective function surrogate model has a unique confidence level throughout the optimization. In the case of this study, since diploid genetic structure is used, and only one surrogate model is used, one confidence level is set for the low fidelity level fitness value.

Tournament selection is used in GAXL algorithm, same as in the case of GA. The population is sorted randomly, and a *pair* of blades is selected as $\mathbf{x}_1$ and $\mathbf{x}_2$ from the sorted list. Starting from highest level (most accurate) fitness to lower levels, existence of fitness is checked. If any of two individuals do not have a fitness assigned on a given level, a lower fidelity level existing fitness values are checked. For the specific case of diploid GAXL; if the bottom (high-fidelity) fitness values exist (calculated at the previous generation) for both $\mathbf{x}_1$ and $\mathbf{x}_2$, then selection is performed based on high-fidelity fitness values, $F^H(\mathbf{x}_1)$, and $F^H(\mathbf{x}_2)$, respectively. Else, the selection is performed based on low-fidelity fitness values $F^L(\mathbf{x}_1)$, and $F^L(\mathbf{x}_2)$.

If the decision is based on high-fidelity fitness values, $F^H(\mathbf{x}_1)$, and $F^H(\mathbf{x}_2)$, then the fitter is selected as the Parent 1, or P_1. A very rare case, both of them are selected given their fitness values are the same. Otherwise, the decision is based on low-fidelity fitness values, $F^L(\mathbf{x}_1)$, and $F^L(\mathbf{x}_2)$, and there are three possible outcome of this tournament: either $\mathbf{x}_1$ or $\mathbf{x}_2$, or both are selected. In this case, "better" (fitter) individual is selected using:

$$
\begin{aligned}
&\text{If } \frac{\left|F^L(\mathbf{x}_1)-F^L(\mathbf{x}_2)\right|}{\min\left(F^L(\mathbf{x}_1),F^L(\mathbf{x}_2)\right)} > CL \quad \text{then} \\
&\qquad \text{If } \; F^L(\mathbf{x}_1) > F^L(\mathbf{x}_2) \quad \text{then} \quad \textit{Select } \mathbf{x}_1 \textit{ as Parent 1} \\
&\qquad \text{Else if } \; F^L(\mathbf{x}_1) < F^L(\mathbf{x}_2) \quad \text{then} \quad \textit{Select } \mathbf{x}_2 \textit{ as Parent 1} \\
&\qquad \text{Endif} \\
&\text{Else } \; \textit{Select } \mathbf{x}_2 \textit{ and } \mathbf{x}_2 \textit{ as Parent 1 and Parent 2} \\
&\text{Endif}
\end{aligned}
\tag{3.11}
$$

Note that, if the fitness values of both individuals are zero, they are both selected for the next generation. The fitness level according to which selection of the parent performed is recorded for the second step.

In the second step, in case only one Parent P_1 is selected in above process, another pair of blades is selected as $\mathbf{x}_3$ and $\mathbf{x}_4$ from the sorted list. However, in that case, only the individuals which have a fitness assigned on the same recorded level of the P_1 participate to selection. Remaining individuals are omitted from the sorted list if any.

If the recorded level is the highest-fidelity fitness level, then the fitter of $\mathbf{x}_3$ and $\mathbf{x}_4$ is selected as the Parent 2, P_2. In case both have the same fitness value, one of them is chosen randomly, $\mathbf{x}_3$.

If the first tournament selection is performed on the low-fidelity fitness level, then the decision is based on low-fidelity fitness values, $F^L(\mathbf{x}_3)$, and $F^L(\mathbf{x}_4)$, and there are three possible outcome of this tournament: either $\mathbf{x}_3$ or $\mathbf{x}_4$ is selected as the P_2, or both are selected according to equation 3.11. Since only one remaining parent is to be selected, if both $\mathbf{x}_3$ and $\mathbf{x}_4$ is selected in the second tournament, then randomly choose $\mathbf{x}_3$ as P_2. The flowchart of multi-fidelity tournament selection process is given in Figure 3.10.

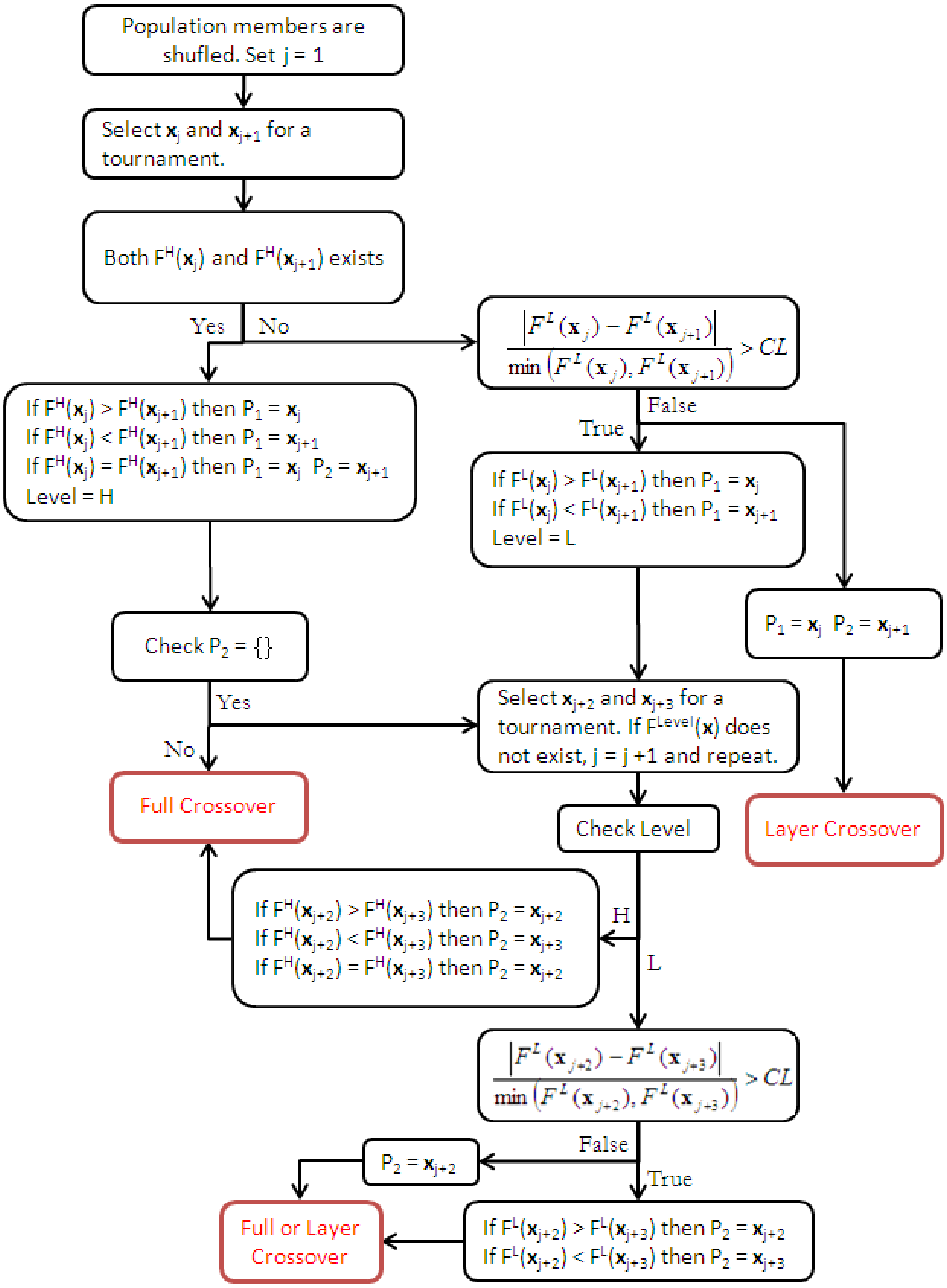


Figure 3.10 Multi-fidelity tournament selection algorithm of a diploid GAXL

3.3.5 Full and Layer Crossover

Once the parents to be mated are determined, crossover is applied to the parents according to a pre-selected probability. The crossover operator determines whether the information coming from the genes of the selected parents will be transferred to the next generation. Therefore, for a proper flow of the information from past to future generations, and to prevent mixing of low and high fidelity information, crossover operator should carefully be modified. Consequently, in GAXL, two different kinds of crossover operator used simultaneously, in which both of them implements the single point crossover technique.

There are mainly two quality levels of information flowing from ancestors to the next generation in GAXL, when compared to GA in which only fixed quality (accuracy) information is transferred. These are the better performing genes according to exact fitness values (highest-fidelity fitness level), and the better performing genes according to surrogate fitness values (lower-fidelity fitness levels). If the selected individuals' lower level fitness values are far away when compared to confidence level, the information coming from the lower level genes can be used to guide higher level chromosomes. Therefore, "full crossover" is used in this case. On the other hand, if the selected individuals' lower level fitness values are within the confidence level, the information coming from the lower level genes cannot be used to guide higher level chromosomes, since one cannot decide if the better performing individual according to low fidelity fitness is really performing better when exact fitness is compared. Therefore, "level crossover" is used in that case. For the specific case of diploid GA, the decision of full versus level crossover is shown in Figure 3.10 and given in equation form as:

$$\text{If} \quad \frac{\left|F^L(P_1) - F^L(P_2)\right|}{\min\left(F^L(P_1), F^L(P_2)\right)} > CL \quad \text{then} \quad \mathit{FullCrossover}$$
$$\text{Else} \quad \mathit{LayerCrossover} \tag{3.12}$$
$$\text{Endif}$$

Note that if the selection is performed according to highest level fitness values, then full crossover is performed. If the selection is performed on the low-fidelity level, and two parents are selected in the first tournament, then level crossover performed. In all other cases, equation 3.12 is used to determine which crossover type is used on the parents.

Full crossover operator requires chromosomes of all levels of selected individuals are crossed over at the same point simultaneously. This ensures the better genes information is passed on all levels. The illustration of full crossover is given in Figure 3.11 for the initial population.

Level crossover operator works only on one layer. This is the layer which selection of the individuals is performed according to this layer's fitness. Level crossover prevents any false information to be passed to higher level chromosomes. The crossover is performed on only the chromosome of the selection performed level. The illustration of level crossover is given in Figure 3.12 for the initial population.

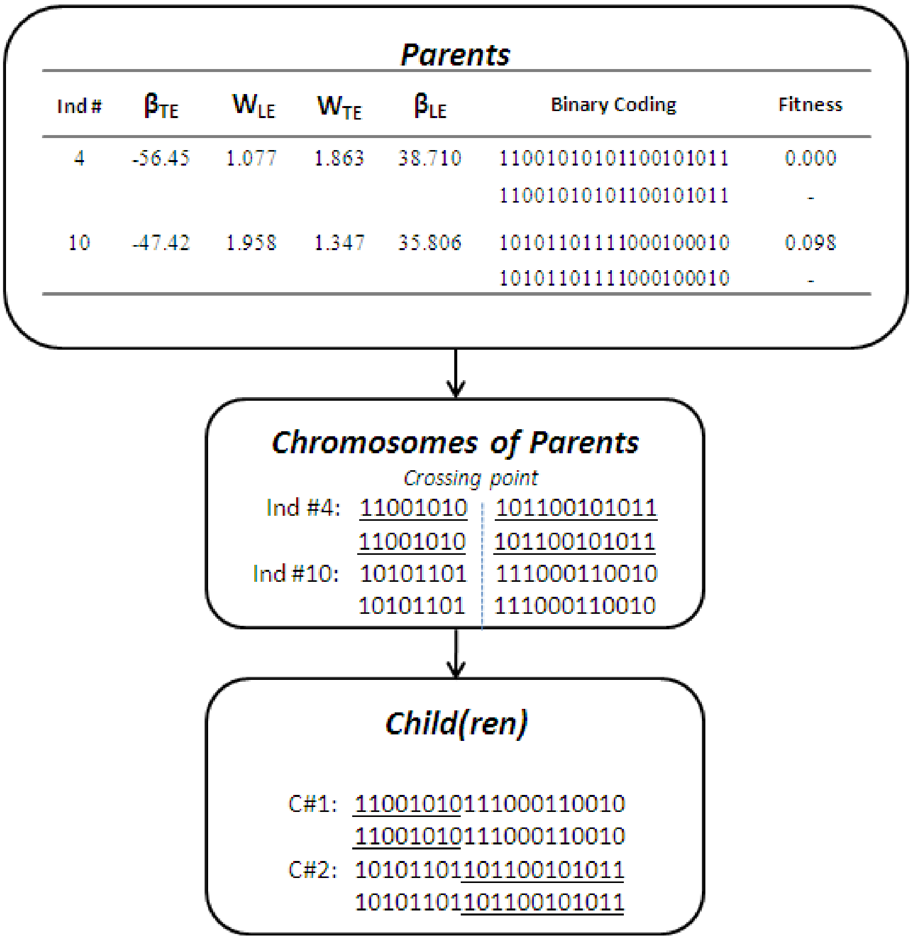


Figure 3.11 Diploid GAXL Full Crossover Illustration

3.3.6 Mutation

Mutation is a bit change of a chromosome that occurs during the crossover process. In GAXL, mutation operator is only applied to the chromosomes of selection performed levels. Mutation probability is set as the probability of performing mutation on a bit of the chromosome which selection is performed on.

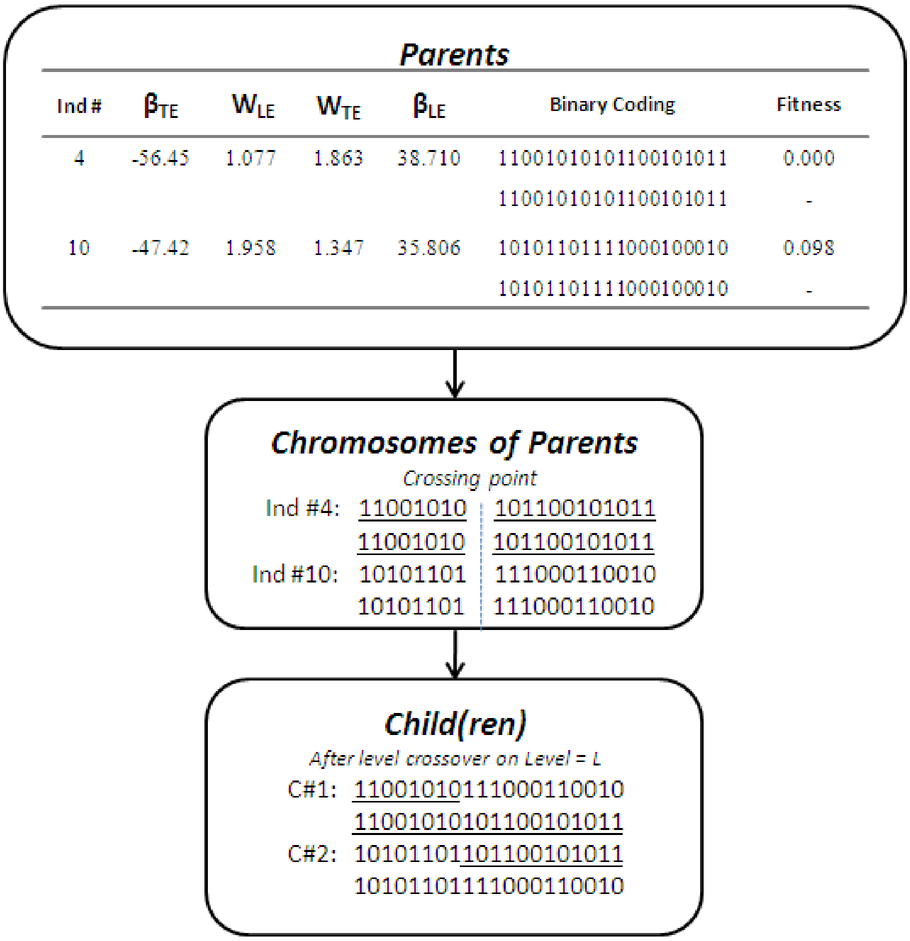


Figure 3.12 Diploid GAXL Level Crossover Illustration

3.3.7 Regeneration and High-Fidelity Fitness Value Assignment

After the creation of children using crossover and mutation operators, they are used to replace old population consisting of their parents. All of the old population members are deleted and all of the children become a new population called as a new *generation*.

The costly objective function evaluation is the main concern for regeneration in GAXL. In traditional GA, fitness value of exact objective function evaluation is calculated at each generation. However, since the main aim and also advantage of GAXL is reducing the number of exact objective function evaluations, a decision must be made for which level of fitness will be calculated and when at each new generation.

As indicated in Section 3.3.2, at the initial generation, only the lowest level fitness values are evaluated for each individual. Then, selection is performed by comparing the lowest fidelity level fitness values. For a given probability of occurrence, crossover and mutation operators works, according to this selection performed level. In the next generation, evaluation of a higher level fitness value for an individual is decided according to previous selection. If in the previous selection, the selected individuals' fitness values are close enough to eachother according to confidence level, then in the next generation, the same level and one level higher fitness values for both individuals are evaluated. If in the previous selection, the selected individuals' fitness values are far away from eachother according to confidence level, then only the same level fitness value of the individual is evaluated in the next generation. For the specific case of diploid GA, the decision of high-fidelity function evaluation is given in equation form as:

$$\begin{aligned}
&\text{If} \quad \frac{\left|F^L(P_1) - F^L(P_2)\right|}{\min\left(F^L(P_1), F^L(P_2)\right)} > CL \quad \text{then} \quad \textit{evaluate}\ F^L(C_1) \\
&\text{Else} \quad \textit{evaluate}\ F^L(C_1)\ \textit{and}\ F^H(C_1) \qquad\qquad (3.13) \\
&\text{Endif}
\end{aligned}$$

Note that if the individuals have already assigned the highest level fitness values, then another evaluation is unnecessary.

3.3.8 Understanding GAXLs

In Section 3.2.9, schemas are defined and discussed to show how GAs converge to a global optimum. These schemas are built from the initial generations and carry valuable information to the next generation. The information carried away is that the schema is a good performer or it is a bad performer in the environment. When it is talked about an individual's performance in a given environment, in traditional GA, environment corresponds to the objective function and performance corresponds to the objective function value evaluated result of the individual's design parameters.

In GAXL, instead of changing the environment, two or more environments are defined and the performance of an individual in these environments is evaluated. The clue is that, one of the environments is exact objective function and the remaining ones are the surrogate models. The schemas work in all of the environments. The selection operator of GAXL ensures the performance of any pair of individuals is compared in only the same environment, and the better performing is selected for sure. If it is not sure, then, both of them are selected and the performance of the pair is evaluated in a higher fidelity level environment. The crossover operator, on the other hand, ensures the information flows from one environment to the other only in the correct way. If the fitness comparison is confident enough that one individual is better than the other in the lower level environment, then this information is passed to higher fidelity level chromosomes of the child by full crossover. However, if the comparison is not confident enough, then, only level crossover is implemented, and the transfer of any wrong information to a higher level is prevented. The full crossover operator works on the schemas, and accelerates the convergence of GAXL without any loss of or misleading information.

CHAPTER 4

MULTI-OBJECTIVE OPTIMIZATION

In this chapter, a haploid and a multiploid multi-objective genetic algorithm is presented together with an application of a turbine rotor aerodynamic optimization problem. First, formulation of the design optimization problems is discussed. Then, a brief description of the haploid multi-objective genetic algorithm (MOGA) technique is presented in the next section, followed by a more detailed description of the operators applied to the UH turbine rotor blade in section three. Next, multiploid multi-objective genetic algorithm (MOGAXL) is introduced. Finally, the application of the multiploid genetic algorithm to design problem is presented.

4.1 Formulation of the Design Optimization Problem

Most real world problems involve multiple objectives. As long as an optimization problem involves more than one objective function, the problem is solved using a multi-objective optimization technique. There are many algorithms and case studies using multi-objective techniques, however, much of these studies transform multi-objective design problem into a single objective function by using some user-defined parameters. On the other hand, in problems with more than one conflicting objective, there is no single optimum solution. There exist a number of solutions which are all optimum. This is the fundamental difference between a single objective and a multi-objective optimization task.

Although there exists many optimal solutions for problems with more than one conflicting objective, a decision maker practically needs only one decision to act. Knowing the optimal solutions, decision maker can select this one optimum solution according to higher level information which is usually qualitative, even experience driven.

4.1.1 Optimization Variables

The *decision vector* **x** is the same as the single objective optimization problem case eq. (3.1) and is given as

$$\mathbf{x} = [x_1, x_2, ..., x_N]^T \tag{4.1}$$

where ^{T} denotes the transpose of the vector. Any specific vector **x** composed of numbers is called a *solution*. All solutions create the *design space* which is N-dimensional by nature.

As in the case of single objective optimization, there are six design parameters selected for a layer geometry shaping: the wedge angles of leading and trailing edges (W_{TE}, and W_{LE}), blade inlet and exit metal angles (β_{LE}, and β_{TE}), the stagger angle (ξ), and the circumferential rotation angle (Θ). Since there are six pre-determined layers and one more design parameter which is the number of blades, a total number of 37 design parameters are used to define one particular 3-dimensional blade profile. Therefore, the design space is 37-dimensional.

4.1.2 Constraints

As in the case of single objective optimization problem, the multi-objective optimization problem usually has a number of constraints which any feasible solution must satisfy. The general form of writing inequality constraint is

$$g_k(\mathbf{x}) \geq 0 \text{ for k=1,2,...,K} \tag{4.2}$$

In certain models, we can have equality constraints as follows

$$h_m(\mathbf{x}) = 0 \text{ for m=1,2,...,M} \tag{4.3}$$

Note that the number of equality constraints, M, must be less than the number of design variables, N. Otherwise, the problem is over-constrained, and there is no degrees of freedom left for optimization.

The constraints in this chapter are only constraining the design variables. Keeping this in mind, the range of each of design variables of the turbine blade are geometrically constrained by the design space, which is taken as the same lower and upper bounds of single objective problem, as shown in Table 4.1.

Table 4.1 Design space constraints

Layer	Minimum Values						Maximum Values					
	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}
1 hub	10	0.1	-60	60	17	3	20	3	-50	75	35	9
2	15	-2	-55	60	17	3	25	-0.1	-45	75	35	9
3	20	-3.5	-45	60	17	3	30	-1.5	-35	75	35	9
4	30	-5	-35	60	17	3	40	-3.5	-20	75	35	9
5	35	-7	0	60	17	3	45	-5	10	75	35	9
6 tip	50	-8	10	60	17	3	60	-6	20	75	35	9
N	26						33					

Additionally, mechanical constraints are imposed on the multi-objective optimization problems. The constrained values with parameters are shown in Table 4.2. These values are set in consistent with the baseline blade values. As long as the constrained values are exceeded, the blade is assigned with the lowest objective function value possible.

Table 4.2 Mechanical Constraints

	Minimum values			Maximum values		
L	t_{LE} (mm)	t_{TE} (mm)	A (mm^2)	t_{LE} (mm)	t_{TE} (mm)	A (mm^2)
1 hub	3	0.5	300	6	1	500
2	3	0.5	300	6	1	500
3	2.5	0.5	200	5	1	400
4	2	0.5	200	4	1	400
5	1.5	0.5	150	3	1	300
6 tip	1	0.5	150	2	1	300

4.1.3 Objective Function

There are I *objective functions*, $\mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots f_I(\mathbf{x}))^T$ which are either linear or nonlinear considered in the multi-objective optimization problem. Each objective function, which can be either minimized or maximized, is converted to a minimization problem by multiplying the objective function by -1 using duality principle [62].

In addition to the decision variable (design) space, the objective functions constitute a multi-dimensional space, which is called as *objective space*. For each solution x in the design space, there exists a point in the objective space, denoted by $f(\mathbf{x}) = \mathbf{z}$. The mapping takes place between N-dimensional decision vector and I-dimensional objective vector. Figure 4.1 illustrates these two spaces and a mapping between them.

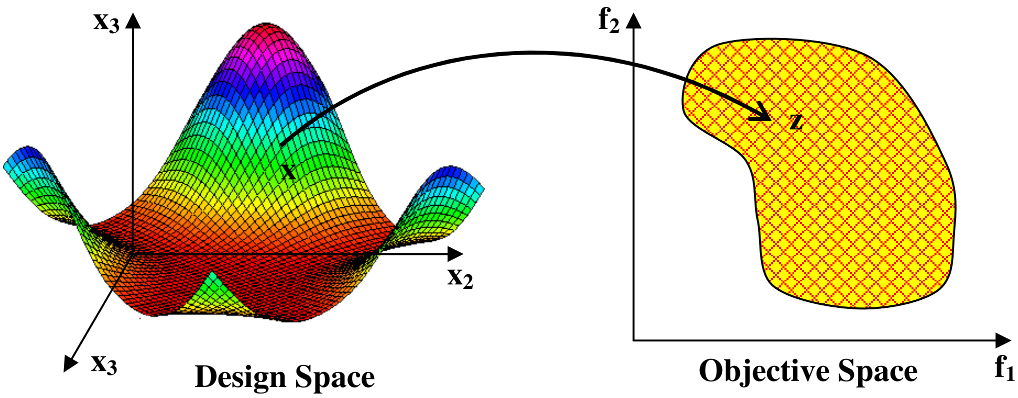


Figure 4.1 Representation of the design space and the objective space

Aerodynamic optimization of turbine blades offers many objective functions such as total pressure loss coefficient, lift coefficient, drag coefficient, lift to drag ratio, blade loading, and blade energy loss, Mach number distribution. Since the blade efficiency which directly takes into account the losses, and blade torque are the final result of the design aim of a rotor, they are selected as the objective functions in this problem (I=2).

4.1.4 Optimization Problem

Once the decision variables, constraints, and the objective function are determined, the optimization problem is constructed as follows:

$$\begin{array}{lll} \text{min} & f_i(\mathbf{x}) = f_i(x_1, x_2, ..., x_N) & \\ \text{such that} & g_k(\mathbf{x}) \geq 0 & \\ & h_m(\mathbf{x}) = 0 & \\ & \mathbf{x} \geq 0 & \end{array} \quad \text{for} \begin{cases} \text{i} = 1,2,\ldots,\text{I} \\ \text{k} = 1,2,\ldots,\text{K} \\ \text{m} = 1,2,\ldots,\text{M} \\ \text{n} = 1,2,\ldots,\text{N} \end{cases} \qquad (4.4)$$

Accordingly, the optimization problems of this thesis in a compact form is that

$$\begin{array}{ll} \text{minimize} & f_1(x) = T \\ & f_2(x) = -1.\eta \\ & \text{subject to Table 4.1 and Table 4.2} \end{array} \qquad (4.5)$$

4.2 Multi-Objective Genetic Algorithm Optimization (MOGA)

This section describes the MOGA methodology to solve the problem given in eq. 4.5. In order to interpret the outcomes of multi-objective optimization, some basic principles and definitions of multi-objective genetic algorithm optimization are discussed in the first part, with a short overview on the structure and basic algorithms. Following parts deal with MOGA structure and operators: initialization, discretization and binary coding, selection, recombination, mutation, and regeneration. For an in-depth treatment of the subject of MOGA's, Deb's book [62] is highly recommended.

4.2.1 Dominance and Pareto-Optimality

Multi-Objective Genetic Algorithms usually use the concept of domination. For an optimization problem of which all functions are to be minimized, a solution $\mathbf{x}_1$ is said to *dominate* the other solution $\mathbf{x}_2$, if and only if:

$$\begin{cases} & f_i(\mathbf{x}_2) \le f_i(\mathbf{x}_1) \text{ for all i = 1,2...I} \\ and & \\ & f_i(\mathbf{x}_2) < f_i(\mathbf{x}_1) \text{ for at least one i} \in \text{[1,2...I]} \end{cases} \tag{4.6}$$

If any of the above condition is violated, the solution $\mathbf{x}_1$ does not dominate the other solution $\mathbf{x}_2$, and $\mathbf{x}_2$ is said to be a non-dominated solution with respect to the other.

For the turbine blade aerodynamic optimization problem, a population of randomly selected six blades is shown in the objective space in Figure 4.2. The objective functions of both efficiency and torque are to be maximized. The design variables of the six blade shapes are shown in Table 4.3. The RANS solution of each blade shape have different corresponding efficiency and torque results, as tabulated in Table 4.3.

Table 4.3 Design parameters of six different blade shapes

Layer	**Blade #1** η = 0.8467 T = 169.9 Nm						**Blade #2** η = 0.8321 T = 149.2 Nm					
	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}
1 hub	19	0.1	-58	70	33	8	11	0	-53	70	33	7
2	21	-0.4	-52	60	23	5	24	-2	-52	74	31	4
3	25	-3.1	-42	64	17	4	22	-2	-39	65	33	8
4	40	-4.2	-34	70	31	4	33	-5	-32	67	31	4
5	44	-5.1	10	73	19	5	40	-6	10	61	34	6
6 tip	51	-6.8	19	71	19	6	50	-8	17	73	25	4
N	32						29					

Layer	**Blade #3** η = 0.8070 T = 159.5 Nm						**Blade #4** η = 0.8400 T = 119.1 Nm					
	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}
1 hub	11	3	-56	73	27	4	14	2	-52	74	25	5
2	20	-1	-49	71	21	8	25	0	-45	70	18	4
3	27	-3	-44	73	17	3	21	-3	-40	67	28	9
4	39	-4	-21	72	18	7	32	-4	-34	60	34	5
5	44	-7	9	60	32	4	35	-6	9	69	26	7
6 tip	54	-7	12	66	23	3	59	-7	17	68	18	7
N	30						26					

Layer	**Blade #5** η = 0.8192 T = 205.8 Nm						**Blade #6** η = 0.8293 T = 186.6 Nm					
	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}
1 hub	17	1	-57	68	24	5	17	0	-54	69	31	3
2	19	-1	-51	64	19	8	21	-1	-45	67	22	7
3	25	-3	-36	64	20	6	29	-3	-41	62	35	3
4	33	-4	-26	74	18	3	37	-4	-31	71	30	4
5	45	-6	1	71	22	4	44	-5	3	65	19	4
6 tip	59	-6	19	73	22	5	50	-8	11	62	34	5
N	33						32					

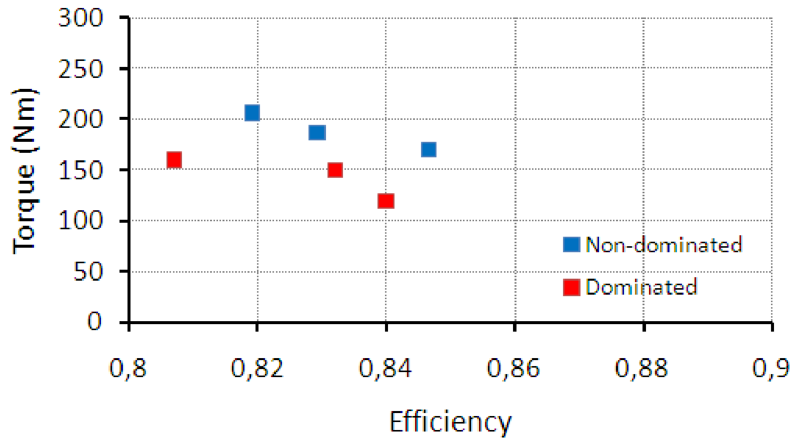


Figure 4.2 Six sample blade shape solutions in the objective space

The concept of domination allows us to select the better among any two given blade shapes in terms of both objectives. For these six blade shapes, all possible pair-wise comparisons are performed and dominated and non-dominated pairs are tabulated in Table 4.4. Dominated blades are marked red in the table. The set of blades, any two of which do not dominate each other, dominates all other blades which do not belong to this set. That is, the blades of this set are fitter than the remaining blades. Consequently, the set of solutions which dominates all other solutions is called the *non-dominated set* for the given set of solutions. For in the case of the sample blade population; blades #2, 3, and #4 are dominated, and #1, #5, and #6 constitutes the non-dominated set of the given population.

When all possible (feasible) individuals in the design space are generated and corresponding objective functions are evaluated, the entire objective space is mapped. The resulting non-dominated set of the entire objective space is called the global *Pareto-optimal* set, or *Pareto-frontier*. Since individuals of the *Pareto-frontier* are non-dominated by any feasible individual, they are the optimal solutions of the MOGA. The Pareto-frontiers of the objective space for four combinations of two types of objectives; min-min, min-max, max-min, max-max; always consist of solutions from a particular edge of the feasible search region.

Table 4.4 Pair-wise comparisons of six different blade shapes

First Blade			Second Blade			Dominance
Blade #	η	T (Nm)	Blade #	η	T (Nm)	
1	0.8467	169.9	2	0.8321	149.2	1 dominates 2
1	0.8467	169.9	3	0.8070	159.5	1 dominates 3
1	0.8467	169.9	4	0.8400	119.1	1 dominates 4
1	0.8467	169.9	5	0.8192	205.8	Non-dominated
1	0.8467	169.9	6	0.8293	186.6	Non-dominated
2	0.8321	149.2	3	0.8070	159.5	Non-dominated
2	0.8321	149.2	4	0.8400	119.1	Non-dominated
2	0.8321	149.2	5	0.8192	205.8	Non-dominated
2	0.8321	149.2	6	0.8293	186.6	Non-dominated
3	0.8070	159.5	4	0.8400	119.1	Non-dominated
3	0.8070	159.5	5	0.8192	205.8	5 dominates 3
3	0.8070	159.5	6	0.8293	186.6	6 dominates 3
4	0.8400	119.1	5	0.8192	205.8	Non-dominated
4	0.8400	119.1	6	0.8293	186.6	Non-dominated
5	0.8192	205.8	6	0.8293	186.6	Non-dominated

4.2.2 Elite Preserving in Multi-Objective Genetic Optimization

In order to use the previously found best individual of the latest generated population in the subsequent generation, an elite-preserving operator is often utilized. Although there are a number of multi-objective genetic algorithms which do not use any elite preserving operator, recent comparative studies of multi-objective evolutionary algorithms shows that, elitism play a major role in the performance of the algorithm. Zitzler et. al. [95, 96] compared eight multi-objective genetic algorithms -only one of which has an elitist-preserving operator-on the six different test problems, and found out that the elitist algorithm clearly outperformed any other algorithm. Accordingly, results indicate that elite-preservation is an important matter for converging to the Pareto-frontier. Therefore, elitist MOGA is used in this study.

The *elite members* **e** is the same as the decision vector **x** given in eq. (4.1), but this case it represents a non-dominated solution obtained and is given as

$$\mathbf{e} = [e_1, e_2, \ldots, e_N]^T \tag{4.7}$$

where ^{T} denotes the transpose of the vector. Any specific vector **e** listed in EMS is called an *elite member* or a *Pareto solution*. At the end of an optimization simulation, all Pareto solutions create the *Pareto Optimal Frontier* of the min-min problem.

4.2.3 Structure of the Multi-Objective Genetic Algorithm

The MOGA is developed based on the "Distance Based Pareto Genetic Algorithm" of Osyczka and Kundu [97]. The method of selecting a set of Pareto optimal solutions and modified distance method are also discussed in Osyczka's book [98].

In general, the optimization algorithm modifies the baseline blade by changing the optimization parameters. Then, the RANS flow solver calculates the objective function values of both the efficiency and the torque, and sends them back to the MOGA. Accordingly, MOGA uses these values to assign a fitness value to every blade. The detailed automated optimization flow chart is given in Figure 3.3.

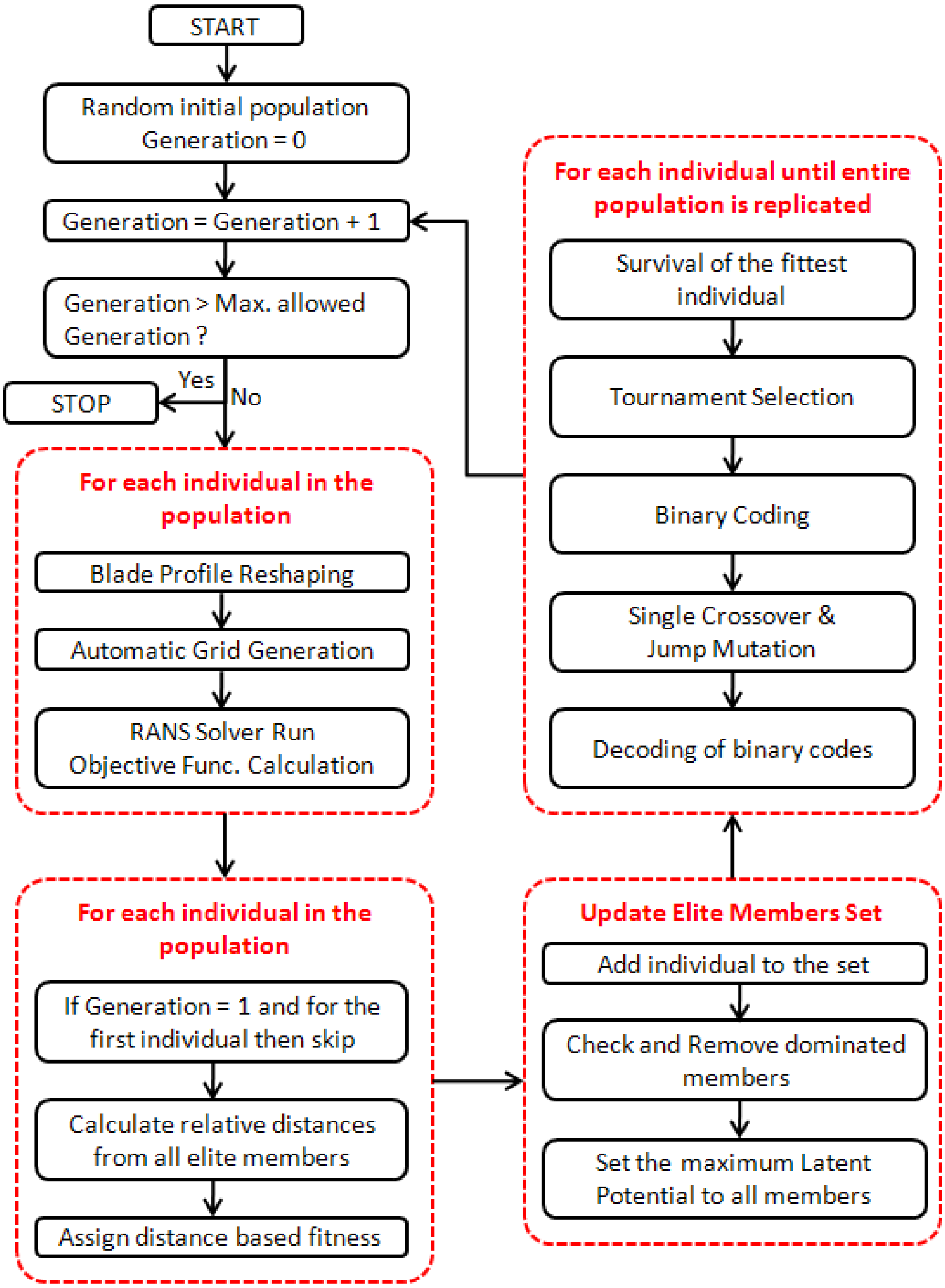


Figure 4.3 Optimization flow chart of MOGA

The initial population is randomly constructed by generating a prescribed number of blades. For each individual, blade shape is constructed. An automatic fine grid is applied and RANS flow solver is run to get efficiency and torque values. The MOGA has an *elite members set* (EMS) which is updated at each generation and initially empty. The first individual of the first generation is directly copied to the EMS, and fitness is assigned. A *distance based fitness* value is calculated for the remaining individuals while the EMS is updated simultaneously according to dominance principle. Only non-dominated individuals are allowed in the EMS. Finally, based on distance based fitness values, GA operators of selection, crossover and mutation are applied to generate the next population. The population generation cycle is repeated until a pre-defined maximum number of function evaluations or generations. At the end of the optimization, the EMS set consists of the Pareto optimal frontier individuals which are to be presented to the decision maker.

From the above discussion, it can be seen that multi objective genetic algorithms differ substantially from more traditional search and optimization methods in the sense that:

- Besides having multiple objectives, MOGAs have two goals instead of one: progressing towards Pareto frontier and diversity of Pareto solutions.
- MOGAs deal with two search spaces: design space and objective space.
- MOGAs are true multi-objective algorithms since they do not require any artificial objective functions such as weighted sum of objectives.

- Elite preserving MOGAs have an additional members set, EMS, which contains the Pareto optimal solutions up to that generation.
- MOGAs provide a number of non-dominated Pareto optimal solutions to a given problem. The final choice is left to the user. Thus, the genetic algorithm is potentially useful for identifying these alternative solutions simultaneously.
- In case of the objective function values can be calculated for the same cost of only one function calculation, such as both efficiency and torque calculation require same RANS solution, MOGAs represent higher level of information about the objective space landscape. MOGAs not only give information of possible optimum solutions for the problem, but also provide the variation of Pareto optimum frontier of an objective function with respect to the other one.

4.3 Distance Based MOGA Operators

Basically, there is only a minor modification of the simple genetic algorithm to construct the MOGA. In this section, the modified GA operators and new MOGA operators will be discussed using a sample population of 10 individuals.

Similar to GAs, MOGA requires an initial population of solutions (individuals) in the design space, before the application of MOGA operators. Initiation of this population begins with the selection of two

MOGA parameters. They are mainly the population size and the chromosome length.

A population size of 10 individuals will be very insufficient for most of the engineering problems since the diversity in the design space will not be obtained. However, increasing the population size will increase the computational cost incurred on the CPU. As the population size increases, number of objective function evaluation increases. Therefore, there is a big trade of between the population size and computational time. Frequently, population sizes of 50 to 200 individuals are used in the literature [62].

Chromosome length selection is another tradeoff between the computational cost and diversity. A very short chromosome means a lack of diversity of the solutions and possible a worse optimization result. However, longer lengths cause a decrease in the convergence speed of the population towards the optimal solution. Practically, chromosome length is determined from the number of design parameters and the desired resolution of the design space for each one of the design parameters.

For illustrating the MOGA's working principles, an example blade optimization problem with 4 design variables; number of blades N, leading flow inlet angle β_{LE}, and leading edge wedge angle W_{LE}, and leading edge circumferential angle Θ values is constructed. For the initial generation, a population size of 10 individuals is created randomly and a chromosome length of 18 bits will is assigned for each individual. The binary coding of ten blades (individuals) is shown in Table 4.5. Note that individuals with unrealistic geometries and inexistent RANS solution are assigned objective function values of zero and not shown in the objective space. This population will be evolved in the remainder of this chapter to clearly illustrate how the MOGA process.

Table 4.5 Initial population

#	N	β_{LE}	W_{LE}	Θ	Binary Coding	Torque (Nm)	Eff. %
1	26	-38.55	21.65	-0.56	000101000100011010	123.4	83.612
2	26	-40.16	34.42	-0.11	000011111111011110	0	0
3	26	-35.32	18.16	-3.27	000111100001000010	120.9	85.693
4	26	-39.19	25.13	-2.15	000100100111001100	130.9	82.671
5	29	-39.19	18.16	-2.71	011100100001000111	0	0
6	32	-43.39	18.74	-0.56	110001010001111010	0	0
7	32	-41.45	32.10	-1.81	110010111101001111	0	0
8	26	-37.58	26.29	-1.35	000101111000010011	0	0
9	26	-40.81	30.94	-0.90	000011011100010111	0	0
10	27	-35.97	18.74	-1.92	001111000001101110	0	0

4.3.1 Fitness Assignment

Although GA uses direct assignment of the objective function value as the fitness of an individual, since MOGA handles more than one objective function, direct fitness assignment is not possible. Therefore, a distance based fitness assignment method is implemented in order to be able to make a comparison for two individuals and decide which one will be selected.

Distance based fitness assignment was proposed and modified later by Osyczka and Kundu [97, 99]. The MOGA has two separate populations: one standard population P_t where GA operators are performed, and another elite members set (EMS) E_t containing all non-dominated solutions found so far, where t indicates the generation number. According to the distance based method, fitness is assigned to each solution in P_t based on its farthest distance from the E_t members. Two new parameters play a key role in determination of the fitness of an individual. These are the *relative distance* of a P_t individual from an elite member k in E_t, and *latent potential value* of an elite member k in an E_t set, $d_k(\mathbf{x})$, and p_k, respectively.

At the first generation, the first member of the initial population P_0 is directly copied to the empty E_0 set and a random latent potential value p_k is assigned. Now, there is an elite member with a latent potential value in the EMS, and this will help in setting the fitness values of the remaining members in the initial population P_0 and the new members of the next generations.

The relative distances are calculated according to its distance from the elite set, $E_t = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_K\}$, where K is the number of elite members in the EMS. For each individual in the P_t the relative distances from all elite members in E_t are calculated by:

$$d_k(\mathbf{x}) = \sqrt{\sum_{i=1}^{I}\left(\frac{f_i(\mathbf{e}_k) - f_i(\mathbf{x})}{f_i(\mathbf{e}_k)}\right)^2} \qquad \text{for } k = 1, 2, \ldots K \qquad (4.8)$$

For the calculation of fitness value of the individual $\mathbf{x}$, the minimum distant elite member is considered where the minimum distance is found as:

$$d_{k^*} = \min(d_k(\mathbf{x})) \qquad \text{for all } k = 1, 2, \ldots K \qquad (4.9)$$

where the index k^* indicates which of the existing elite members in E_t is nearest to the individual **x**.

After the determination of the minimum distance and the minimum distant elite member, the individual is checked whether it is dominated by the elite members or not. If the individual is a new Pareto solution, the E_t is updated by adding the new non-dominated solution **x** and removing members $\mathbf{e}_k$ which are dominated by **x**. The fitness and the latent potential values of the individual **x** (which is now an elite member in the EMS, too) are calculated using:

$$F(\mathbf{x}) = p_{k^*} + d_{k^*} \tag{4.10}$$

$$p_k = F(\mathbf{x}) \tag{4.11}$$

where index k in Eqn. 4.11 belongs to the new elite member in the EMS.

On the other hand, if the individual **x** is dominated by any elite member in E_t, then it is not accepted in the EMS and its fitness is calculated as follows:

$$F(\mathbf{x}) = \begin{cases} p_{k^*} - d_{k^*} & \text{for } F(\mathbf{x}) \geq 0 \\ 0 & \text{for } F(\mathbf{x}) < 0 \end{cases} \tag{4.12}$$

Note that in this equation, a minimum fitness value of zero is assigned for avoiding non-negative fitness values.

The fitness values of all individuals in the population are evaluated while the EMS is constantly updated. At the end of a generation, GA operators like selection, crossover, and mutation is applied to create a new population. Before starting the fitness assignment calculations of the new generation, latent potential values of all elite members in the EMS are replaced by that of maximum among the elite members. That is,

$$p_k = \max\left(p_k(\mathbf{e})\right) \qquad \text{for all k = 1, 2, … K} \tag{4.13}$$

Consequently, all elite members in the EMS are assigned an equal latent potential value regardless of their previous values or position in the objective space. This ensures that, no distinction is made between elite members, since they are all non-dominated and there is no other higher level information to decide which elite member is a better performing than the other.

Distance based fitness assignment helps MOGA in three ways. First, a dominated individual will have a worse fitness value compared to a non-dominated individual. Second, if the new individual dominates some elite members, then the fitness assignment procedure supports solutions closer to the Pareto-optimal frontier by assigning bigger fitness values to distant solutions from the existing EMS. Third, if the non-dominated solution lies in the same Pareto front along with the EMS, the distance based fitness assignment helps maintaining the diversity among EMS by assigning bigger fitness values to an isolated solution on the same front.

For the max-max sample problem with an initial population given in Table 4.5, the calculated parameters determining the fitness of individuals in the population of the first generation together with the updated EMS are listed in Table 4.6. The first member of the population is directly copied to the empty EMS and a latent potential value of 2 is assigned, which is also equal to the fitness of the individual. Since there is only one member in the EMS, the distance between the elite member and the second individual is calculated as 1.4142 and set as the minimum distance with k^* is equal to one, the index of minimum distant elite member. Since the elite member dominates the second individual, the fitness of it is calculated using 4.12 and EMS remains unchanged. For the third individual, the EMS is updated by adding this individual since it is a new non-dominated solution. Similar computations are performed for the remaining individuals of the population, and three Pareto solutions exist after the first generation. Note that, all three non-zero fitness individuals of the first population are added to EMS. This is only due to the fact that these random individuals are not dominated by each other.

Table 4.6 Generation #1 fitness calculation and EMS update: P_0

#	N	β_{LE}	W_{LE}	Θ	Torque (Nm)	Eff. %	k^*	d_{k^*}	$F(\mathbf{x})$	EMS Update	K
1	26	-38.55	21.65	-0.56	123.4	83.61	-	-	2	Yes	1
2	26	-40.16	34.42	-0.11	0	0	1	1.4142	0.5858	No	1
3	26	-35.32	18.16	-3.27	120.9	85.69	1	0.0325	2.0325	Yes	2
4	26	-39.19	25.13	-2.15	130.9	82.67	1	0.0623	2.0623	Yes	3
5	29	-39.19	18.16	-2.71	0	0	1	1.4142	0.5858	No	3
6	32	-43.39	18.74	-0.56	0	0	1	1.4142	0.5858	No	3
7	32	-41.45	32.10	-1.81	0	0	1	1.4142	0.5858	No	3
8	26	-37.58	26.29	-1.35	0	0	1	1.4142	0.5858	No	3
9	26	-40.81	30.94	-0.90	0	0	1	1.4142	0.5858	No	3
10	27	-35.97	18.74	-1.92	0	0	1	1.4142	0.5858	No	3

Contrary to zero objective function value, these individuals also have a positive small fitness value assigned. Therefore, although these individuals have an unrealistic blade shape due to one or more design parameters, non-negative fitness assignment helps the individual to be selected by a low probability so that remaining parameters of the individual can contribute to the next generations. This is so useful for MOGA converge to the global optimum rather than a local optimum.

Noting the assigned fitness values of the individuals, they are all greater than the dominated individuals. This helps GA converge to the Pareto frontier at each consecutive generation. Note also that the non-dominated individuals have different fitness values. The distant individual to the EMS receives a higher fitness value when compared to an individual closer to the elite members in the objective space. This helps MOGA to keep diversity of the Pareto frontier, which is the main goal of any multi-objective optimization technique.

The second generation calculations are shown in Table 4.7. Note that, the calculations in this generation start with a three-member EMS, which consists of the Pareto optimal individuals of the first generation. In this generation, 2nd and 8th non-zero objective function individuals are dominated by an EMS member, whereas 4th and 5th individuals are non-dominated by the EMS. Although 4th individual is added to the EMS to have a 4-member population, addition of the 5th individual in the EMS did not alter the set size, since 5th individual dominated one ex-elite member in the EMS and caused its removal.

Finally, the 3rd generation calculation results are given in Table 4.8. In this case, 5th individual is non-dominated, and added to the EMS. This individual dominated two ex-elite members in the EMS and caused their removal from the elite set. There is no guarantee that the number of elite members in the EMS will increase in each successive generation as shown in this case. But, considering the entire evolution of MOGA, one may expect two goals to be accomplished: convergence of Pareto frontier to the global optimum frontier, and the increase of the number of elite members in the Pareto frontier with a diverse distribution.

Table 4.7 Generation #2 fitness calculation and EMS update: P_1

#	N	β_{LE}	W_{LE}	Θ	Torque (Nm)	Eff. %	k^*	d_{k^*}	$F(\mathbf{x})$	EMS Update	K
1	33	-44.03	32.10	-1.81	0	0	1	1.4142	0.6481	No	3
2	26	-38.55	17.00	-0.56	119.6	83.77	2	0.0244	2.0379	No	3
3	32	-43.71	25.13	-2.15	0	0	1	1.4142	0.6481	No	3
4	26	-37.58	26.87	-1.47	123.4	85.02	1	0.0168	2.0791	Yes	3
5	26	-37.90	30.94	-0.90	128.7	84.12	3	0.0245	2.1037	Yes	4
6	32	-43.71	22.23	-0.56	0	0	1	1.4142	0.6481	No	4
7	26	-39.84	25.71	-1.92	0	0	1	1.4142	0.6481	No	4
8	27	-41.45	32.10	-1.81	126.1	83.37	4	0.0218	2.0819	No	4
9	26	-38.55	21.65	-0.11	0	0	1	1.4142	0.6481	No	4
10	26	-39.19	21.06	-1.92	0	0	1	1.4142	0.6481	No	4

4.3.2 Dominance Check

In the process of distance based fitness assignment, EMS needs to be updated. This is accomplished by comparing the new individual with the elite members in the EMS using the dominance check procedure described in this section.

Table 4.8 Generation #3 fitness calculation and EMS update: P_2

#	N	β_{LE}	W_{LE}	Θ	Torque (Nm)	Eff. %	k^*	d_{k^*}	$F(\mathbf{x})$	EMS Update	K
1	26	-39.19	25.71	-1.92	0	0	1	1.4142	0.6894	No	4
2	26	-41.45	32.10	-1.81	118.8	82.56	2	0.04	2.0637	No	4
3	26	-37.58	26.87	-0.56	127.0	84.03	4	0.0131	2.0906	No	4
4	26	-38.55	17.00	-0.56	119.6	83.77	2	0.0244	2.0792	No	4
5	27	-42.10	32.10	-1.47	131.6	84.80	4	0.0239	2.1276	Yes	3
6	26	-37.58	26.87	-1.35	0	0	1	1.4142	0.6894	No	3
7	32	-43.39	17.00	-0.56	0	0	1	1.4142	0.6894	No	3
8	32	-43.71	21.65	-0.90	0	0	1	1.4142	0.6894	No	3
9	27	-41.77	32.10	-0.11	0	0	1	1.4142	0.6894	No	3
10	26	-37.58	26.87	-1.69	117.2	83.03	2	0.0431	2.0606	No	3

The concept of domination is described in section 4.2.1. The eqn. 4.6 is used to check the dominance of the individual during the fitness assignment procedure. The flowchart of EMS update procedure according to the dominance check is given in Figure 4.4. Accordingly, the individual under consideration can fall in any of three cases:

Case 1

The individual is dominated by one or more elite members in the EMS; therefore there is no change in the EMS.

Case 2

Although the individual is a new Pareto solution, it does not dominate any of the existing elite members in the EMS, therefore the individual simply added to the EMS.

Case 3

The individual is a new Pareto solution which dominates at least one elite member (called as ex-elite) in the EMS. In this case, ex-elite members are removed from the EMS and the individual is added as the new elite member to the EMS.

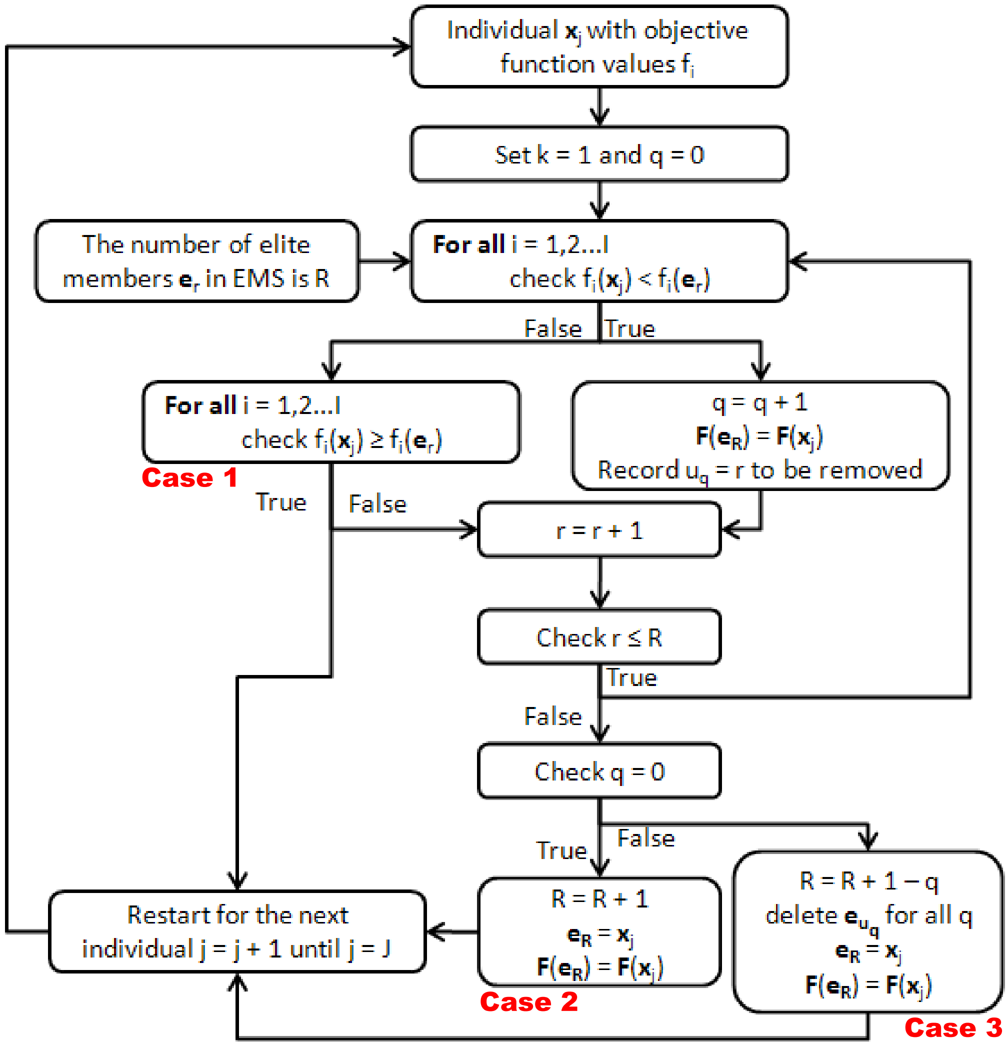


Figure 4.4 EMS Update procedure flowchart

For the sample optimization problem, the final elite members in the EMS after the 3rd generation are listed in Table 4.9. In the Pareto optimal frontier, there are three non-dominated members, which serve a basis for an implementation of higher level information by the decision maker. For example, if the number of blades should be 27 due to the clocking effect, then the decision maker may elect the 3rd Pareto optimal solution at the end.

Table 4.9 EMS at the end of 3rd Generation: E_3

#	N	β_{LE}	W_{LE}	Θ	Torque (Nm)	Eff. %
1	26	-37.58	26.87	-1.47	123.4	85.02
2	26	-35.32	18.16	-3.27	120.8	85.69
3	27	-42.10	32.10	-1.47	131.6	84.80

4.3.3 Remaining MOGA Operators

As soon as the distance based fitness assignment procedure is completed for the population, simple GA operators are applied on the population considering the assigned fitness values. There is no difference between the GA operators and MOGA operators theoretically and for the sake of convenience, the same numerical parameters of selection, crossover and mutation are used for both GA and MOGA.

For the example optimization problem, which is a max-max type, the evolution of the Pareto optimum frontier is illustrated in Figure 4.4 for the first 3 generations. In this figure, the concave initial Pareto front transforms to a convex type in the next generations. Moreover, non-dominated initial solutions are dominated better performing solutions in the sense of both objective functions. The MOGA optimization result presents a set of non-dominated Pareto solutions as shown in this figure.

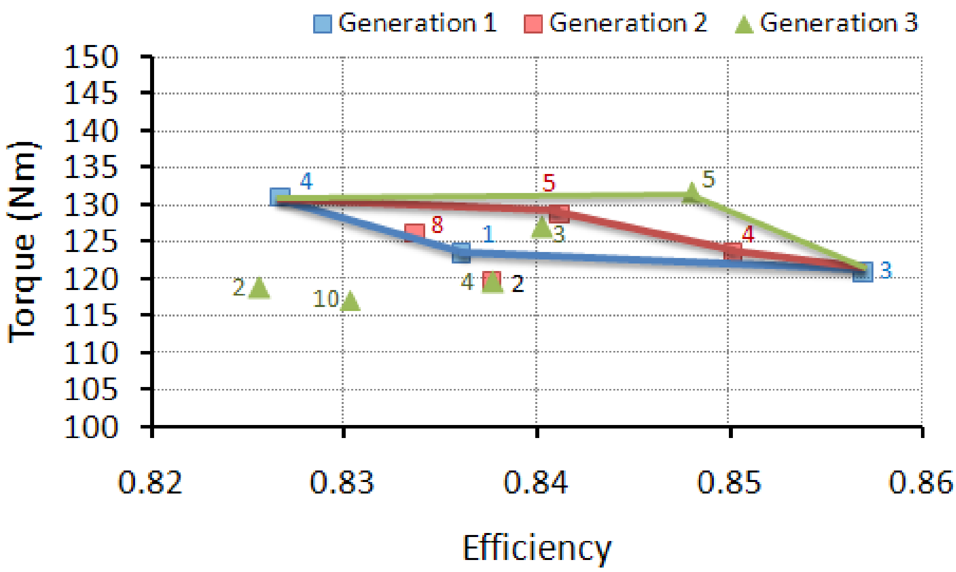


Figure 4.5 Pareto frontier evolutions in three successive generations

4.4. Application of Test Problems

In order to validate the distance based MOGA and compare the performance of MOGA with respect to other multi-objective optimization algorithms, two test problems are selected. These test problems are optimized using MOGA, and the calculated Pareto-optimal frontier is compared with the results of a simulated annealing algorithm MC-MOSA and a genetic algorithm NSGA-II given in [18, 100].

4.4.1 Test Problem 1

The first test problem is a structural optimization problem of the design of a two-bar truss structure [18]. The two objectives are to minimize the three dimensional design space; distance, y, and the maximum stress on the two truss members A_1 and A_2. There are also constraints on the geometric dimensions and maximum stress allowed, as the optimization problem is given below:

$$\begin{array}{ll} \min & f_1 = A_1\sqrt{16+y^2} + A_2\sqrt{16+y^2} \\ \min & f_2 = \max(\sigma_{AC}, \sigma_{BC}) \\ \text{such that} & \begin{cases} \max(\sigma_{AC}, \sigma_{BC}) \le 10^5 \text{ kPa} \\ 1 \le y \le 3 \text{ m} \\ 0 \le A_i \le 0.01 \text{ m}^2 \quad \text{for i =1,2} \end{cases} \end{array} \tag{4.14}$$

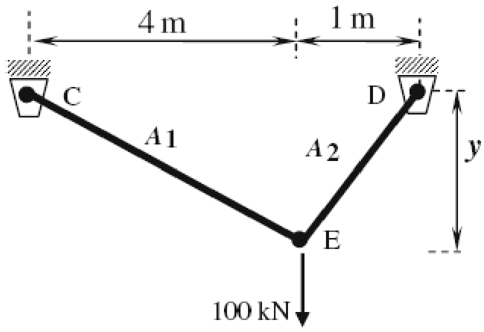


Figure 4.6 Two bar truss for Test Problem 1 [18]

The optimization problem is solved using a population size of 100 individuals, and the MOGA is run up to 200 generations. Single point cross-over probability is set to .90 meanwhile jump mutation probability is 0.04. The chromosome length is selected to be 45, equally shared between the three design variables. The problem is solved with 30 different randomly chosen initial population and the average of the number of elite members found in the EMS is given in Table 4.10.

The number of elite members found after the given number of objective functions evaluations is compared with the NSGA-II and MC-MOSA in the Table 4.10. Although NSGA-II is a genetic algorithm and MC-MOSA is a simulated annealing algorithm, their performance is comparable to each other but worse than MOGA in terms of elite members found in the Pareto-optimum frontier. The distribution of the members in the Pareto frontier found by MOGA after 20,000 function evaluations are shown in Figure 4.7 with that of MC-MOSA. Both algorithms maintain a good diversity along the frontier.

Table 4.10 Comparison of elite members in the global Pareto-optimum frontier of MOGA with NSGA-II and MC-MOSA for Test Problem 1

Number of Function Evaluations	NSGA-II	MC-MOSA	MOGA
1,000		30	129
5,000		73	292
10,000	96	86	391
20,000		99	630

The number of non-dominated elite members in the EMS is plotted in Figure 4.8 with respect to the number of generations. It is clear that the size of the EMS increases with generation, except for occasional drops due to new non-dominated solutions which eliminates some of the ex-elite members in EMS by dominating them. When all elite members in the EMS converge to the global Pareto-optimum frontier, then the EMS size will monotonically increase or remain constant with successive generations.

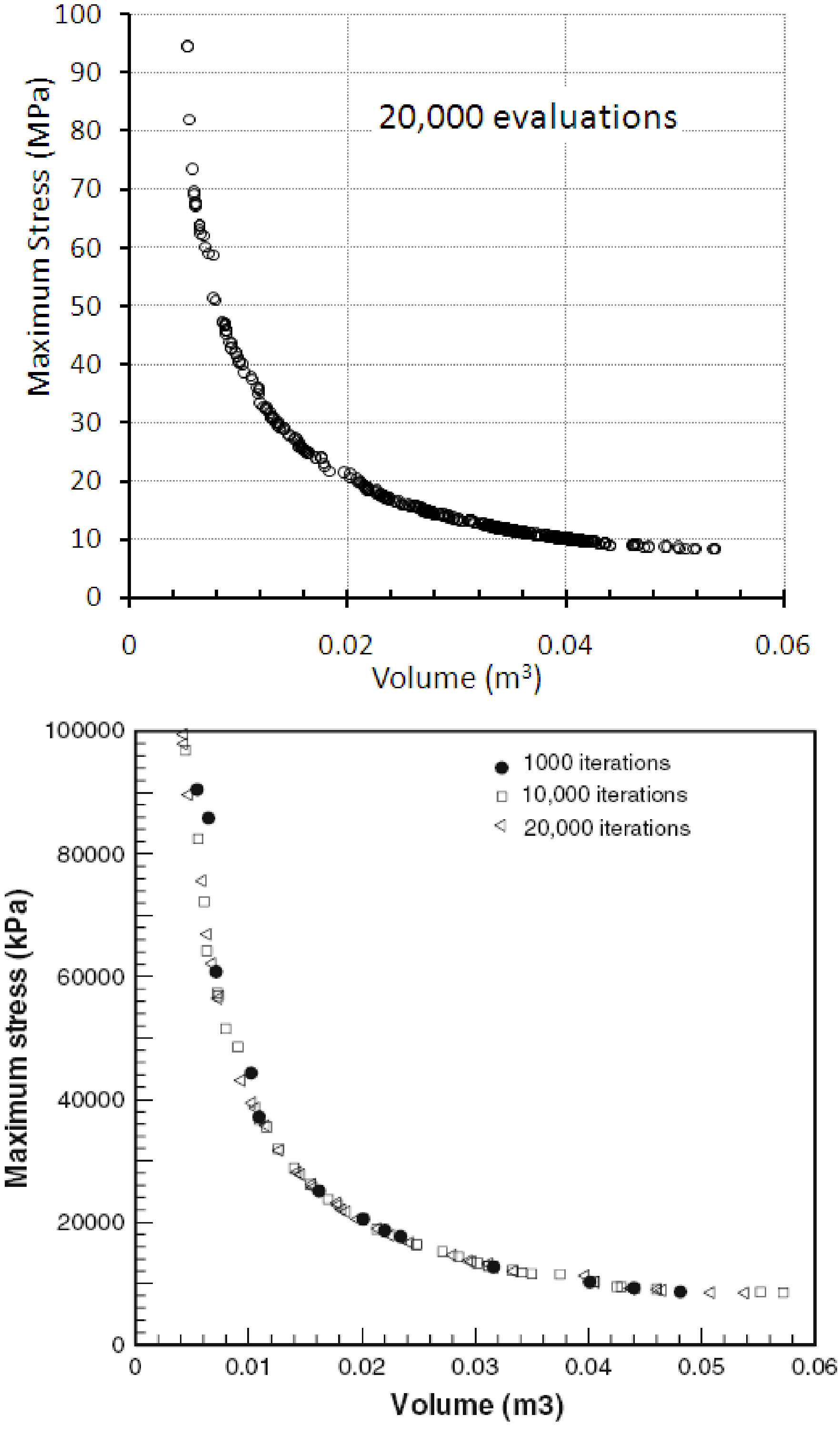


Figure 4.7 Comparison of Pareto-optimum fronts of MOGA (upper) and MC-MOSA (lower) after 20,000 function evaluations

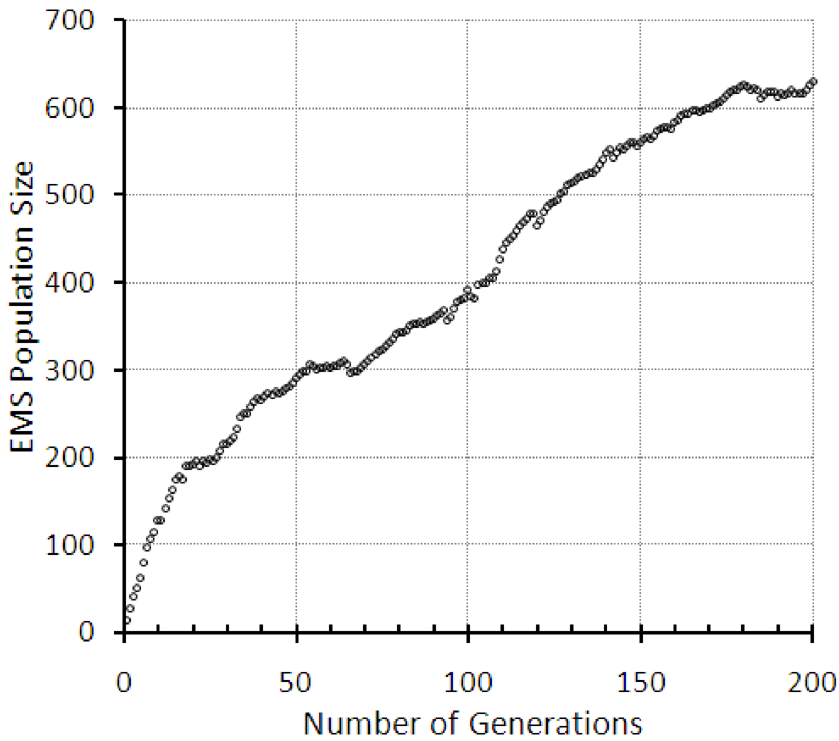


Figure 4.8 Growth of EMS size with respect to generation number

4.4.2 Test Problem 2

This test problem is given in [35] and formulated as follows:

$$
\begin{aligned}
&\min && f_1 = 2 + (x_1 - 2)^2 + (x_2 - 1)^2 \\
&\min && f_2 = 9x_1 - (x_2 - 1)^2 \qquad\qquad (4.15)\\
&\text{such that} && \begin{cases} x_1{}^2 + x_2{}^2 \le 225 \\ x_1 - 3x_2 + 10 \le 0 \\ -20 \le x_1 \le 20 \\ -20 \le x_2 \le 20 \end{cases}
\end{aligned}
$$

The optimization problem is solved using a population size of 100 individuals, and the MOGA is run up to 200 generations. Single point cross-over probability is set to 0.90 meanwhile jump mutation probability is 0.04. The chromosome length is selected to be 30, equally shared between the three design variables. The problem is solved with 30 different randomly chosen initial populations and the average of the number of elite members found in the EMS is given in Table 4.11.

Table 4.11 Comparison of elite members in the global Pareto-optimum frontier of MOGA with NSGA-II and MC-MOSA for Test Problem 2

Number of Function Evaluations	NSGA-II	MC-MOSA	MOGA
1,000		58	181
5,000		134	1571
10,000	96	170	3583
20,000		212	7454

The number of elite members found after the given number of objective functions evaluations is compared with the NSGA-II and MC-MOSA in the Table 4.11. The simulated annealing algorithm MC-MOSA performed better than NSGA-II which is a genetic algorithm, but both of them are worse than MOGA in terms of elite members found in the Pareto-optimum frontier. The distribution of the members in the Pareto frontier found by MOGA after 20.000 function evaluations are shown in Figure 4.9. MOGA maintain a good diversity along the frontier. The number of non-dominated elite members in the EMS is plotted in Figure 4.10 with respect to the number of generations.

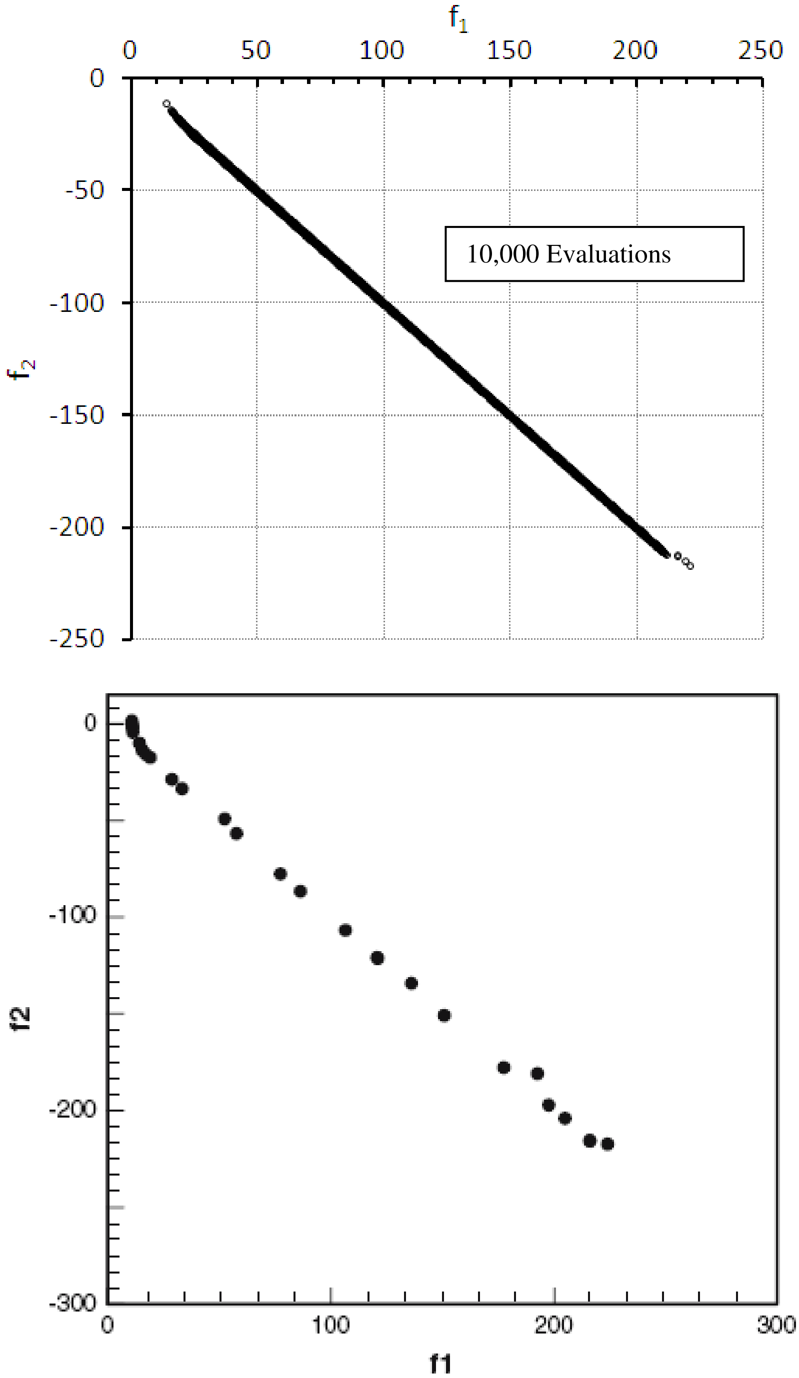


Figure 4.9 Comparison of Pareto-optimum fronts of MOGA (upper) and MC-MOSA (lower) after 10,000 function evaluations

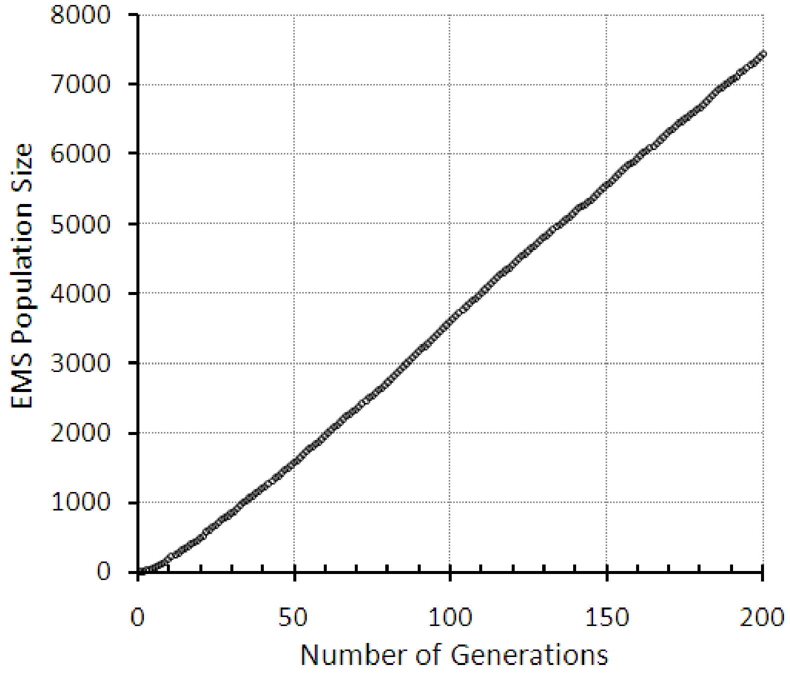


Figure 4.10 Growth of EMS size with respect to generation number

4.5 Multiploid Genetic Algorithm Optimization (MOGAXL)

Although, MOGA is able to converge to the global Pareto-optimum frontier, it consumes considerable computational power to converge to the global frontier for a 3D aerodynamic design problem which requires expensive high-fidelity RANS solutions. This is the similar drawback of genetic algorithms as discussed in Chapter 3 and it is avoided by using faster but less accurate low-fidelity models that can be implemented in MOGA.

The novel method developed in Chapter 3 is applied to multi-objective genetic algorithm. The method again is able to interpret the high-fidelity, high quality, very expensive information such as a fine mesh solution of the blade, and the low-fidelity, low quality inexpensive information such as a coarse mesh blade solution, and does not degrade the quality of the optimization result by mixing the information between different fidelity solutions. Called as the multiploid multi-objective genetic algorithm (MOGAXL), the method uses more than one chromosome to represent an individual in the design space.

The automated optimization flow chart is given in Figure 4.11. The initial population is randomly constructed by generating a prescribed number of blades. For each individual, blade shape is constructed. An automatic coarse or fine grid is applied, and RANS flow solver is run to get efficiency and torque values. The MOGAXL has two *elite members sets* (EMS-L and EMS-H) which are updated at each generation and initially empty. The first individual of the first generation is evaluated using low fidelity coarse grid solution and directly copied to the low fidelity elite members set EMS-L, and fitness is assigned. A *distance based fitness* value is calculated for the remaining individuals while the EMSs are updated simultaneously according to dominance principle. Only non-dominated individuals are allowed in the EMSs. Finally, based on distance based low and high-fidelity fitness values, GAXL operators of selection, crossover and mutation are applied to generate the next population. The population generation cycle is repeated until a pre-defined maximum number of function evaluations or generations. At the end of the optimization, the EMS-H set consists of the Pareto optimal frontier individuals which are to be presented to the decision maker.

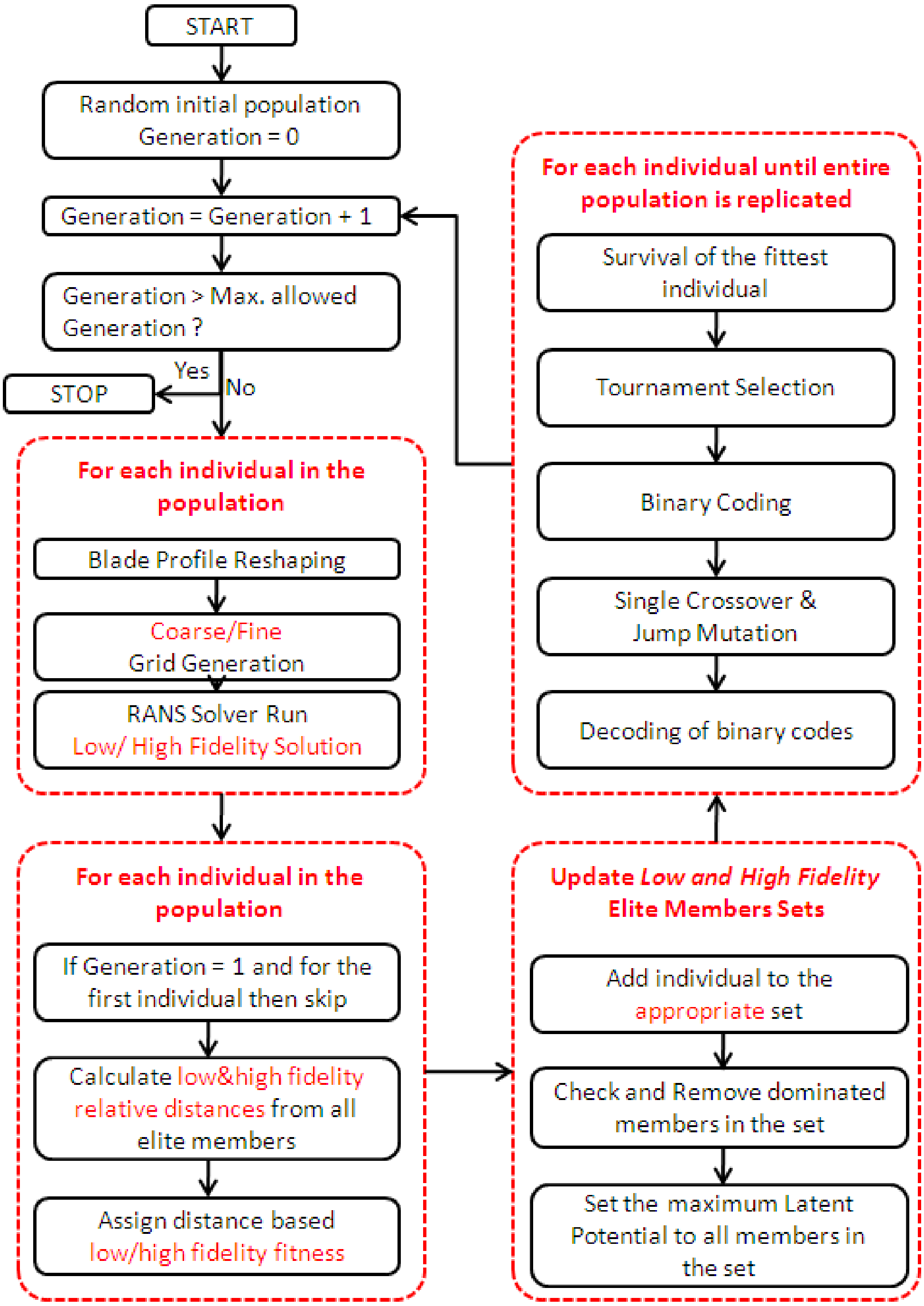


Figure 4.11 Flowchart of MOGAXL

From the above discussion, it can be seen that MOGAXL differ substantially from MOGA:

- MOGAXL has multiploid genetic representation, i.e. more than one chromosomes for each individual, whereas MOGA has only one chromosome per individual.
- MOGAXL maintains two elite members sets; a low fidelity set called as EMS-L, and a high fidelity set called as EMS-H. MOGA has only one EMS.
- Distance based fitness assignment procedure of MOGAXL involves the assignment of high and low-fidelity fitness values for an individual whereas MOGA assigns only one fitness, which is the high fidelity fitness.
- Although tournament selection is common, MOGAXL has a pre-defined confidence level parameter which sets the base for selection decision.
- Although single point crossover is common, MOGAXL implements two different kinds of crossover, namely full crossover, and level crossover.
- Although jump mutation method is common, MOGAXL mutates only crossover performed chromosomes.

Besides these differences, similar to the above structure of multi-level genetic algorithms, a MOGAXL is developed based on the MOGA in section 4.2. Again, for comparison with MOGA, the optimization algorithm modifies the UH rotor blade by changing the optimization parameters. Then, MOGAXL requests either the low fidelity or the high fidelity fitness value from the RANS flow solver. Coarse mesh or fine mesh, respectively, are generated and solved to calculate the objective function value –the efficiency or the torque- and assigned to fitness of the correct level of chromosomes. Accordingly, MOGAXL interprets these values to create a new generation.

4.5.1 Initial Population

Similar to MOGA, MOGAXL require an initial population of solutions (individuals) in the design space, before the application of MOGAXL operators. Initiation of this population begins with selection of two MOGA and one MOGAXL parameters. They are mainly the population size and the chromosome length, and number of chromosomes i.e. genetic structure, respectively.

In this study, only diploid genetic representation which consists of two chromosomes is used. Every method presented can be applied to multiploid presentations. Although initial population is constructed randomly in every new problem, in order to be consistent with the MOGA discussion, the first generation population is copied from 10 individuals given in Table 4.5, and twin chromosomes with a length of 18 bits each are assigned for each individual for the same illustrative optimization problem given in section 4.2. These settings will be used remaining of this chapter to clearly illustrate how the MOGAXL code process. MOGAXL generates random numbers using the same method of GA, which is the Knuth's subtractive method [36].

In the concept of diploid MOGAXL, as mentioned before, each variable set defines a blade configuration with two corresponding chromosomes. At the initial generation, the two chromosome strings are set the same of each other. The upper chromosomes correspond to lower fidelity fitness, and lower chromosomes correspond to higher fidelity. Consequently, for a diploid MOGAXL, the upper chromosome can be interpreted as the geometry with coarse mesh and the lower chromosome can be interpreted as the same blade geometry with fine mesh. Note that only low fidelity fitness values are evaluated at the initial generation. The binary coding of ten blades (individuals) is shown in Table 4.12.

Table 4.12 MOGAXL Initial population, objective function values are calculated for only low fidelity coarse mesh solution

#	N	β_{LE}	W_{LE}	Θ	Binary Coding	Torque (Nm)	Eff. %
1	26	-38.55	21.65	-0.56	000101000100011010	120.3	81.11
					000101000100011010	-	-
2	26	-40.16	34.42	-0.11	000011111111011110	0	0
					000011111111011110	-	-
3	26	-35.32	18.16	-3.27	000111100001000010	113.7	86.84
					000111100001000010	-	-
4	26	-39.19	25.13	-2.15	000100100111001100	142.7	83.21
					000100100111001100	-	-
5	29	-39.19	18.16	-2.71	011100100001000111	0	0
					011100100001000111	-	-
6	32	-43.39	18.74	-0.56	110001010001111010	0	0
					110001010001111010	-	-
7	32	-41.45	32.10	-1.81	110010111101001111	0	0
					110010111101001111	-	-
8	26	-37.58	26.29	-1.35	000101111000010011	0	0
					000101111000010011	-	-
9	26	-40.81	30.94	-0.90	000011011100010111	0	0
					000011011100010111	-	-
10	27	-35.97	18.74	-1.92	001111000001101110	0	0
					001111000001101110	-	-

4.5.2 Discretization and Binary Coding

The illustrative optimization problem design space is discretized similar to MOGA example, i.e. each design variable has 32 values except the number of blades with 8 possibilities in the design space, which means that there are a possible of 262,144 blade shapes to be searched. The optimum blade is one of these blades in this simple discretized problem.

Binary coding is followed after discretization. At the first generation, in which all of the blades are randomly chosen, once a chromosome of a blade is obtained similar to GA, then it is copied as much as the number of levels of MOGAXL. In the case of a diploid MOGAXL, low fidelity level chromosome is binary coded from randomly chosen blade parameters. Then, the high fidelity level chromosome is copied exactly from the low fidelity level chromosome. Finally, all the blades in the first generation are randomly generated and then binary coded.

As the population evolves with each new generation, different level chromosomes differ from each other because of level/full crossover and mutation operators which will be discussed in the forthcoming sections.

4.5.3 Distance Based Multi-fidelity Fitness Assignment

Although MOGA uses distance based fitness assignment of the objective function value as the fitness of an individual, since MOGAXL handles multi-fidelity models to calculate objective function values, simple distance based fitness assignment is not possible. Therefore, a modified version of distance based fitness assignment method is implemented in order to be able to make use of surrogate models in MOGAXL computations.

The MOGAXL has more than one elite members set. The number of EMSs depends on the number of surrogate models. Since only one low fidelity model is used in this thesis, two EMSs will be implemented, namely the EMS-L and EMS-H, corresponding to lower-fidelity solution and the higher-fidelity solution, respectively. As in the case of MOGA, only one standard population P_t exists where GA operators are performed. Among the elite members sets (EMSs) E_t^L and E_t^H, E_t^L contains all non-dominated lower-fidelity solutions found so far, where t indicates the generation number. Similarly, E_t-H contains all non-dominated higher-fidelity solutions found so far. According to the *multi-fidelity distance based fitness assignment method*, two (or more depending on the number of chromosomes) fitness values are assigned to each solution in P_t based on its farthest distance from the E_t^L and E_t^H members. For each EMS, corresponding *relative distance* $d_k(\mathbf{x})$ of a P_t individual from an elite member k in EMS, and *latent potential value* p_k of an elite member k in EMS are calculated separately and used to determine the low and high fidelity fitness values.

The relative distances are calculated according to its distance from the elite set, $E_t^l = \{\mathbf{e}_1^l, \mathbf{e}_2^l, \ldots, \mathbf{e}_K^l\}$, where K is the number of elite members in the l^{th}-level EMS. Diploid MOGAXL has two l values corresponding to low fidelity fitness and high fidelity fitness; L and H, respectively.

For each individual in the P_t the relative distances from all elite members in E_t^l are calculated by:

$$d_k(\mathrm{x}) = \sqrt{\sum_{i=1}^{I} \left(\frac{f_i^{\,l}(\mathrm{e}_k^{\,l}) - f_i^{\,l}(\mathrm{x})}{f_i^{\,l}(\mathrm{e}_k^{\,l})} \right)^2} \qquad \text{for k = 1, 2, … K} \qquad (4.16)$$

where f^l terms represent the l-fidelity calculated values of the objective functions.

For the calculation of *l*-fidelity fitness value of the individual **x**, the minimum distant elite member in the E_t^l is considered, where the minimum distance is found as:

$$d_{k^*} = \min\left(d_k(\mathbf{x})\right) \qquad \text{for all k = 1, 2, … K} \tag{4.17}$$

where the index k* indicates which of the existing elite members in E_t^l is nearest to the individual **x**.

After the determination of the minimum distance and the minimum distant elite member, the individual is checked whether it is dominated by the elite members of E_t^l or not. If the individual is a new Pareto solution, the E_t^l is updated by adding the new non-dominated solution **x** and removing members $\mathbf{e}_k^l$ which are dominated by **x**. The *l*-fidelity fitness and the corresponding latent potential values of the individual **x** (which is now an elite member in the E_t^l, too) are calculated using:

$$F^l(\mathbf{x}) = p_{k^*} + d_{k^*} \tag{4.18}$$

$$p_k = F^l(\mathbf{x}) \tag{4.19}$$

where $F^l(\mathbf{x})$ term represents the *l*-fidelity fitness values, and index k in Eqn. 4.19 belongs to the new elite member in the E_t^l.

On the other hand, if the individual **x** is dominated by any elite member in E_t^l, then it is not accepted in the E_t^l and its *l*-fidelity fitness is calculated as follows:

$$F^l(\mathbf{x}) = \begin{cases} p_{k^*} - d_{k^*} & \text{for } F^l(\mathbf{x}) \geq 0 \\ 0 & \text{for } F^l(\mathbf{x}) < 0 \end{cases} \tag{4.20}$$

Note that in this equation, a minimum fitness value of zero is assigned for avoiding non-negative fitness values.

4.5.4 Regeneration and High-Fidelity Function Evaluation

At the initial generation, only low-fidelity objective function values are calculated for every individual and, the first member of the initial population P_0 is directly copied to the empty E_0^L set while a random latent potential value p_k is assigned to it. Now, there is an elite member with a latent potential value in the EMS-L, and this will help in setting the low-fidelity fitness values of the remaining members in the initial population P_0 and the new members of the next generations.

Once the low-fidelity fitness values of all individuals in the population are evaluated while the EMS-L is constantly updated at the initial generation, multi-fidelity tournament selection is carried out. None of the individuals in the first generation has high-fidelity fitness values assigned, and the E_t^H set is left empty at the first generation.

At the end of the fitness assignment, the multi-fidelity selection method which is developed for multiploid GAXL is also used for MOGAXL. The method is described in Section 3.3.4 in the previous chapter. The full and level crossover operators are described in section 3.3.5 in the same chapter. The mutation operator is discussed in section 3.3.6.

Before starting the fitness assignment calculations of the new generation, corresponding latent potential values of all elite members in the EMS-L and EMS-H are replaced by that of maximum among the elite members. That is,

$$p_k = \max\left(p_k(\mathbf{e})\right) \qquad \text{for all k = 1, 2, … K} \tag{4.21}$$

Consequently, all elite members in the each EMS are assigned an equal latent potential value regardless of their previous values or position in the objective space. This ensures that, no distinction is made between elite members in an EMS, since they are all non-dominated. Note that the latent potential values of EMS-L and EMS-H are not the same.

The costly objective function evaluation is the main concern for regeneration in MOGAXL. In MOGA, high-fidelity objective function evaluation is carried out to assign fitness at each generation. However, since the main aim and also advantage of MOGAXL is reducing the number of costly high-fidelity objective function evaluations, a decision must be made for which level of fitness will be assigned and when at each new generation.

As indicated, at the initial generation, only the lowest level fitness values are assigned for each individual. Then, selection is performed by comparing the lowest fidelity level fitness values. For a given probability of occurrence, crossover and mutation operators works, according to this selection performed level. In the next generation, assignment of a higher level fitness value for an individual is decided according to previous selection. If in the previous selection, the selected individuals' low-fidelity fitness values are close enough to eachother according to confidence level, then in the next generation, the same level and one level higher fitness values for both individuals are assigned. If in the previous selection, the selected individuals' fitness values are far away from eachother according to confidence level, then only that level fitness value of the individual is assigned in the next generation. For the specific case of diploid GA, the decision of high-fidelity fitness assignment is given in equation form as:

$$
\begin{aligned}
&\text{If} \quad \frac{\left|F^L(P_1) - F^L(P_2)\right|}{\min\left(F^L(P_1), F^L(P_2)\right)} > CL \quad \text{then} \quad \textit{evaluate } F^L(C_1) \\
&\text{Else} \quad \textit{evaluate } F^L(C_1) \textit{ and } F^H(C_1) \qquad (4.22) \\
&\text{Endif}
\end{aligned}
$$

Note that if the individuals have already assigned the highest level fitness values, then repeated evaluation is unnecessary. The EMS update works only for fitness assigned individuals. If a high-level fitness is not yet assigned, this individual is not included in the dominance check of EMS-H. On the other hand, if the individual is going to be assigned a high-level fitness, then dominance check procedure is applied for the EMS-H update. Note that, if the EMS-H is empty, the first individual to be assigned a high-level fitness will be directly added to the EMS-H.

CHAPTER 5

OPTIMIZATION RESULTS

This chapter presents the single and multi-objective aerodynamic turbine blade shape optimization results. The baseline UH 1st stage turbine rotor blade shape is optimized for the given test conditions. Objective functions are selected as maximizing torque and maximizing efficiency. In the first part of this chapter, these objectives are solved using a haploid GA and a diploid GAXL. In the second part, the objective functions are optimized by the multi-objective optimization algorithms; haploid MOGA and diploid MOGAXL. The geometric and flow-field properties of the optimum blade shapes are discussed at each related optimization section separately. The performance variations of GAXL and MOGAXL according to the algorithm parameters are discussed at the remaining part.

5.1 Single-objective Optimization Results

The baseline UH rotor blade is first optimized for maximum torque. Second, the objective function is set as maximizing the blade adiabatic efficiency. Both optimization problems are solved using a haploid GA and a diploid GAXL. Table 5.1 shows the GA and GAXL parameter values used in the optimization processes. The parameters are kept same for both of the optimization problems.

Table 5.1 Optimization parameter values of GA and GAXL

Parameter	GA	GAXL
Number of Chromosomes	1	2
Design Parameters	37	37
Chromosome Length	219	219
Number of child	1	1
Maximum Generation	100	100
Population Size	100	100
Elitism	Yes	Yes
Cross-over Probability	50%	50%
Mutation Probability	2%	2%
Low-fidelity Confidence Level	-	5%

Table 5.2 UH Baseline blade shape parameterization

Layer	Baseline Blade					
	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}
1 hub	10	1.5	-55	65	32	9
2	19	-0.5	-50	65	25	7
3	27	-2.3	-39	65	25	6
4	36	-3.9	-27	66	25	5
5	38	-6.2	0	70	20	4
6 tip	53	-6.9	12	71	17	3
N	30					
Torque (Nm)	143.3					
Efficiency	86.27 %					

Before the optimization, the baseline blade shape is parameterized using blade profile reshaping algorithm and the parameter values are listed in Table 5.2.

5.1.1 Maximizing Torque

The optimum rotor blade shape is found by using GA and GAXL algorithms. The two algorithms have converged to different torque values in 100 generations. The optimum design parameters of GA-optimized and GAXL-optimized blade shapes are compared in Table 5.3. The optimized torque values and corresponding efficiencies are given in the table as well.

Table 5.3 Optimum blade shape parameters of GA and GAXL

Layer	Optimum Blade found by GA						Optimum Blade found by GAXL					
	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}
1 hub	16	0.3	-55	61	32	3	11	2.9	-56	73	25	6
2	16	-0.5	-51	73	26	6	23	-0.9	-52	71	20	5
3	28	-2.9	-42	70	21	6	30	-2.5	-42	61	22	5
4	35	-4.4	-23	61	32	6	39	-4.7	-23	67	35	9
5	44	-6.2	5	71	26	7	45	-5.6	4	60	28	9
6 tip	53	-7.4	18	62	19	7	60	-6.3	16	60	24	7
N	33						33					
Torque (Nm)	253.8						254.9					
Efficiency	78.24%						72.96%					

The number of blades increased from 30 to 33 for maximizing the baseline blade torque. GAXL optimized blade is twisted more than the GA optimized blade according to the stagger angles. Circumferential angles are lower than the baseline blade values for the GAXL optimized blade, indicating the lean of the optimum blade is less. Leading edge flow angles are consistent with the baseline blade and the optimized blades, except higher inlet angles at the mid-to-tip sections of the optimized blades. The GAXL optimized blade exit angle at the hub is higher than the baseline blade, contrary to GA optimized blade. Optimized blades have lower tip exit angle when compared to baseline blade. The data in the table indicates thicker leading edge radius from mid to tip sections of the optimized blades. While baseline blade trailing edge thickness decreases from hub to tip, optimized blades have higher trailing edge thickness at the tip regions.

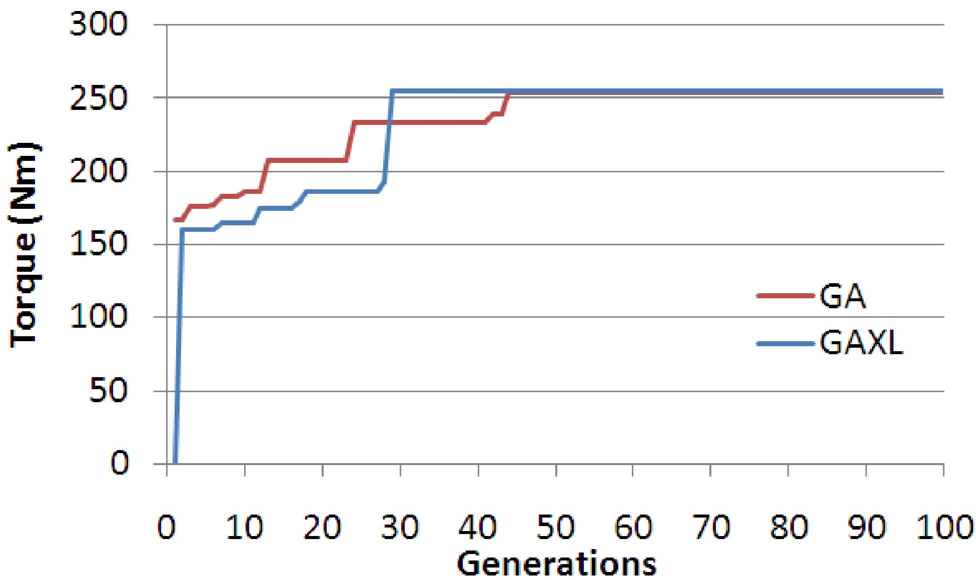


Figure 5.1 Maximization of torque using GA and GAXL

The evolution of the 100-member population towards 100 generations is shown in Figure 5.1 for GA and GAXL. Both optimization algorithms converged to a torque value very close to each other. Although in this figure, GA looks like outperformed GAXL at the initial 30 generations according to faster convergence, GAXL converged to the global optimum after 29 generations, whereas GA converged after 44 generations. Convergence of GAXL to the global optimum for this problem demonstrated the GAXL's ability of global convergence using surrogate models.

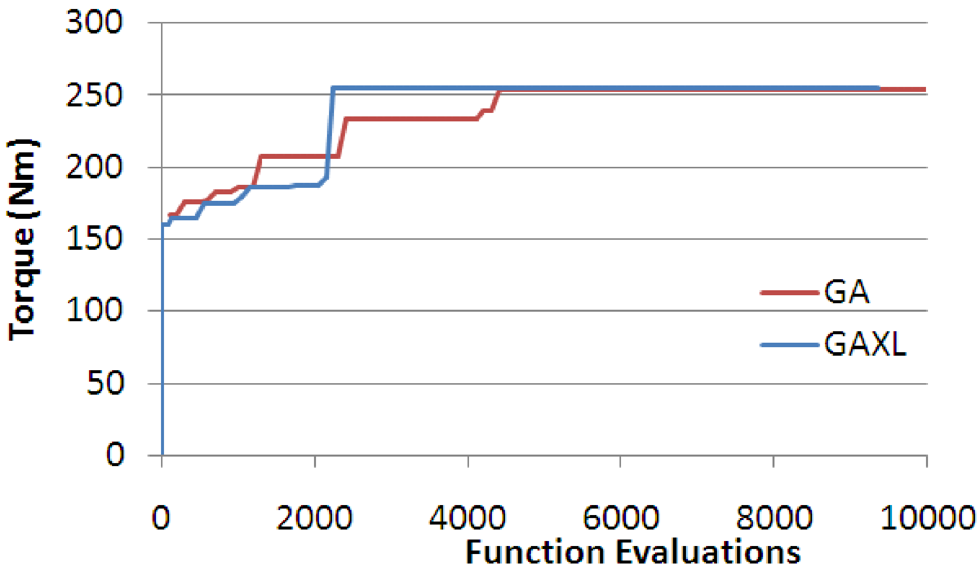


Figure 5.2 Computational Cost of GA and GAXL optimization cycle

The computational cost of the optimization cycle is equivalent to the number of high-fidelity function evaluations needed for convergence to the global optimum. For this purpose, the time evolution of the optimizations per high-fidelity (fine grid solution) function evaluation is given in Figure 5.2. At the beginning, GA increased the maximum blade torque faster than GAXL. However, GAXL converged to global optimum faster than GA; only in 2244 high-fidelity function evaluations, when compared to 4400 function evaluations of GA. GAXL converged to the global optimum almost two times faster than GA.

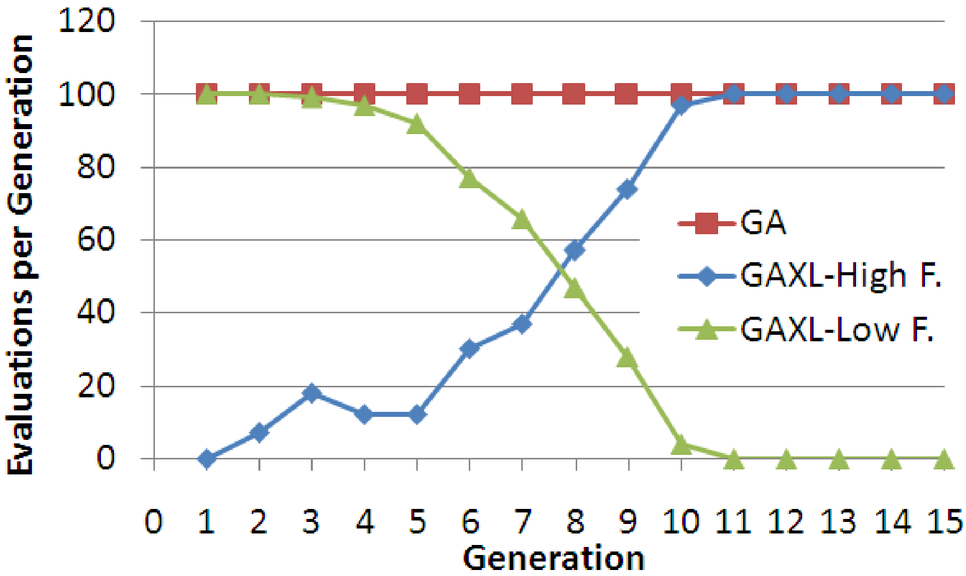


Figure 5.3 High and low fidelity function evaluations of GA and GAXL

The accelerated convergence of GAXL is investigated by comparing the high and low function evaluations during successive generations. For this reason, the number of high/low fidelity objective function evaluations at each generation is shown in Figure 5.3. The figure is limited to only first 15 generations, since GA and GAXL have only evaluated high fidelity values beyond the 15^{th} generation.

Since GA only uses high fidelity evaluations, the number of evaluations is fixed at 100 at each generation, which is equal to the population size. Besides 100 low-fidelity evaluations, GAXL did not evaluate any high-fidelity torque values at the first generation. After the second generation, low fidelity function evaluations of GAXL decreased monotonically to zero, until the 11^{th} generation. Beginning from the second generation, high fidelity evaluations are increased to 100 evaluations at the 11^{th} generation. In the remaining part of the optimization, according to the GAXL algorithm theory, GAXL basically acted as a simple GA automatically. This is because all of the individuals in the 11^{th} population have their high-fidelity fitness values calculated, and the selection process is performed considering only the high-fidelity fitness values. Therefore, low-fidelity calculations are not needed after the 11^{th} generation.

Until the 11^{th} generation, GAXL performed 344 high-fidelity computations, compared to 1000 computations of GA. The number of computations of GA and GAXL are equal after the 11^{th} generation. The accelerated global optimization of GAXL is not only due to the difference of 656 computations of the first 10 generations, but also mainly because of the faster convergence of GAXL in the later steps of the optimization at the same number of function evaluations compared to GA. The main reason of this acceleration is the schemes that are built by the low fidelity chromosome strings in the initial generations and carried over the next generations.

5.1.2 Maximizing Efficiency

The design variables of maximum efficiency blade shape and baseline rotor are compared in Table 5.4. The number of blades decreased from 30 to 26, which is also the design space lower constraint. According to the stagger angles, the blade is twisted more at the hub, and same in the remaining sections. Circumferential angles of optimized blades are similar to the baseline blade values. Leading edge flow angles are lower for the GA optimized blade, and higher for the GAXL optimized blade at the hub sections compared to baseline blade values. The exit angles of the GAXL optimized blade are remarkably higher than the baseline blade values at the hub section. The data in the table indicates inversely proportional leading edge wedge angles with the baseline blade at the mid sections. The trailing edge wedge angles of the GAXL optimized blade are consistent with the baseline blade values.

Table 5.4 Optimum blade shape parameters of GA and GAXL

Layer	Optimum Blade found by GA						Optimum Blade found by GAXL					
	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}	ξ	Θ	β_{LE}	β_{TE}	W_{LE}	W_{TE}
1 hub	16	0.8	-60	65	32	8	14	1.5	-50	70	28	9
2	19	-0.3	-45	69	25	3	18	-0.3	-46	70	28	8
3	25	-2.4	-42	61	19	4	25	-1.7	-44	63	35	5
4	35	-3.7	-27	67	29	6	39	-4.4	-23	69	21	4
5	37	-6.3	6	60	18	4	37	-6.5	6	64	19	4
6 tip	50	-7.1	19	74	18	7	50	-6.7	13	65	18	4
N	26						26					
Torque (Nm)	122.9						131.3					
Efficiency	87.29%						87.44%					

The maximization of the efficiency problem is solved using a simple haploid GA and the diploid GAXL. The evolution of the 100-member population towards 100 generations is shown in Figure 5.4 for GA and GAXL. Contrary to torque optimization, GAXL clearly outperformed than GA at the end of the optimization cycle according to faster convergence, and converged to a better efficiency value.

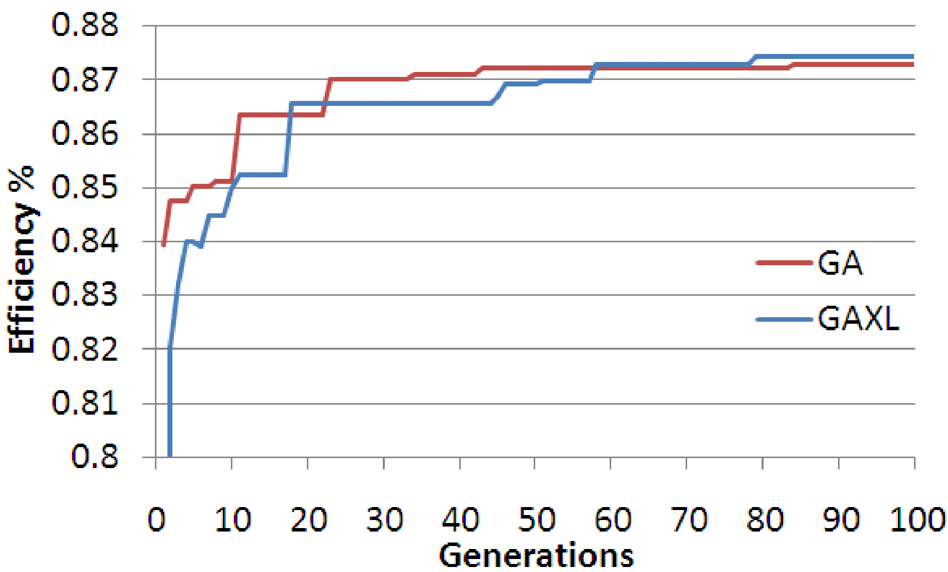


Figure 5.4 Maximum efficiency values per generation

For this specific optimization case, GAXL did not evaluate the high-fidelity efficiency values for the first generation. Again, GAXL accelerated the optimization and converged faster than GA in this optimization problem. After the 60th generation, GAXL converged to a better individual at each successive generation than the simple GA.

The computational cost comparison of GA and GAXL per generation is given in Figure 5.5. Until 2000 high-fidelity objective function evaluations, GA and GAXL performed equally well. Afterwards, GA converged to better efficiency values than GAXL until the 6000th generation. In the last part however, GAXL converged faster and a better efficiency value than GA, as in the case of torque optimization problem.

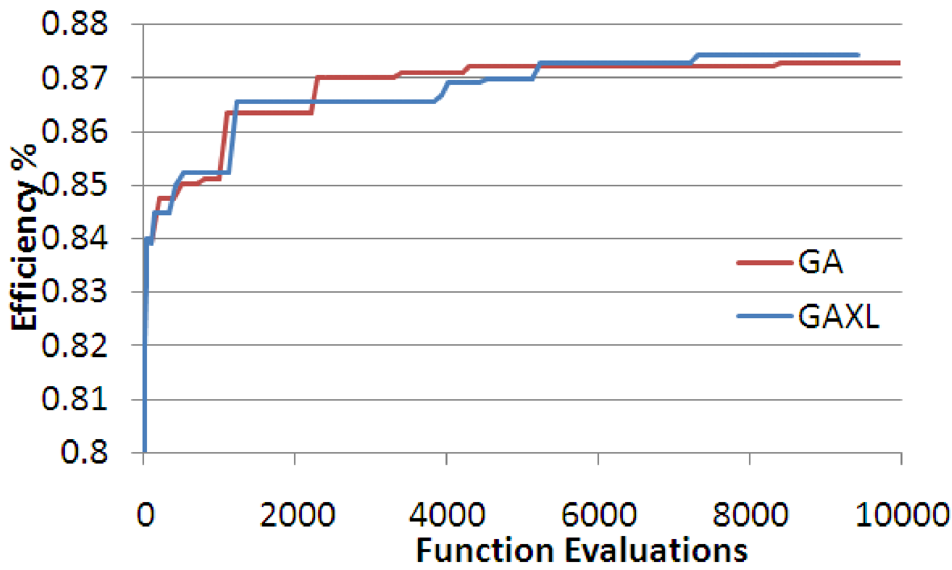


Figure 5.5 Computational cost of maximizing efficiency

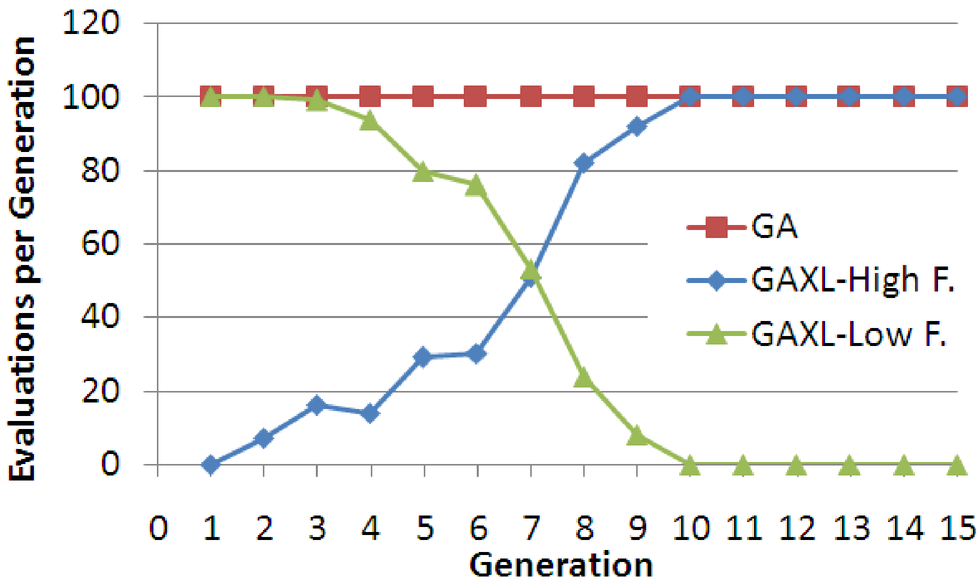


Figure 5.6 High and low fidelity function evaluations of GA and GAXL

Figure 5.6 represents the high and low fidelity function evaluations of GA and GAXL for the first fifteen generations. At the first generation, GAXL only requested the low fidelity function evaluations. The high fidelity function evaluations started from the second generation until the 10th generation, where fitness values all of the 100 individuals are assigned only high-fidelity. From that point on, no low fidelity fitness assignment is performed on the individuals, and GAXL acted similar to the simple GA. On the other hand, GA only uses high fidelity function evaluations from the first generation, which is equal to the number of individuals in the population. The optimum rotor geometries for maximum efficiency and torque are shown in Figure 5.7.

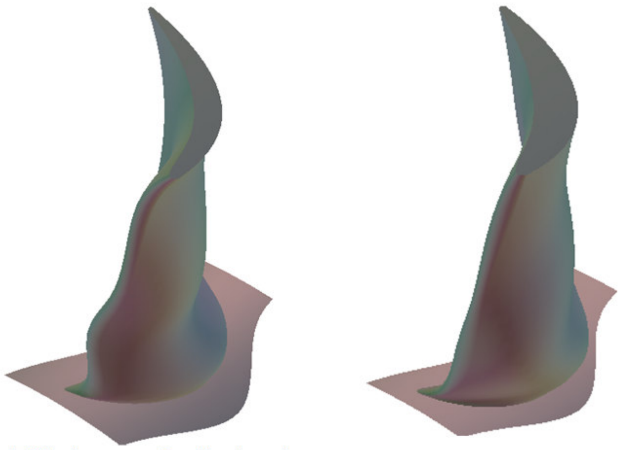


Figure 5.7 Optimum rotor geometries of maximum efficiency and torque

5.1.3 Effect of population size on the GA/GAXL performances

To investigate the effects of population size on the performances of GA and GAXL, the optimization problems are solved using a 25-individual population size. The evolution of the 25-member population towards 40 generations is shown in Figure 5.8 for GA and GAXL. Although in this figure, GA looks like clearly outperformed GAXL, as will be discussed later, GAXL meets the two objectives of this study, since only the high-fidelity torque values are plotted in Figure 5.8. For this specific optimization case, GAXL did not evaluate the high-fidelity torque values for the first generation. From the 2nd till the 10th generation, some of the individuals are assigned high-fidelity torque values. In the remaining part of the optimization, GAXL algorithm switched to simple GA automatically since all of the individuals in the 10th population have their high-fidelity fitness values calculated. Therefore, low-fidelity calculations are not performed after the 10th generation.

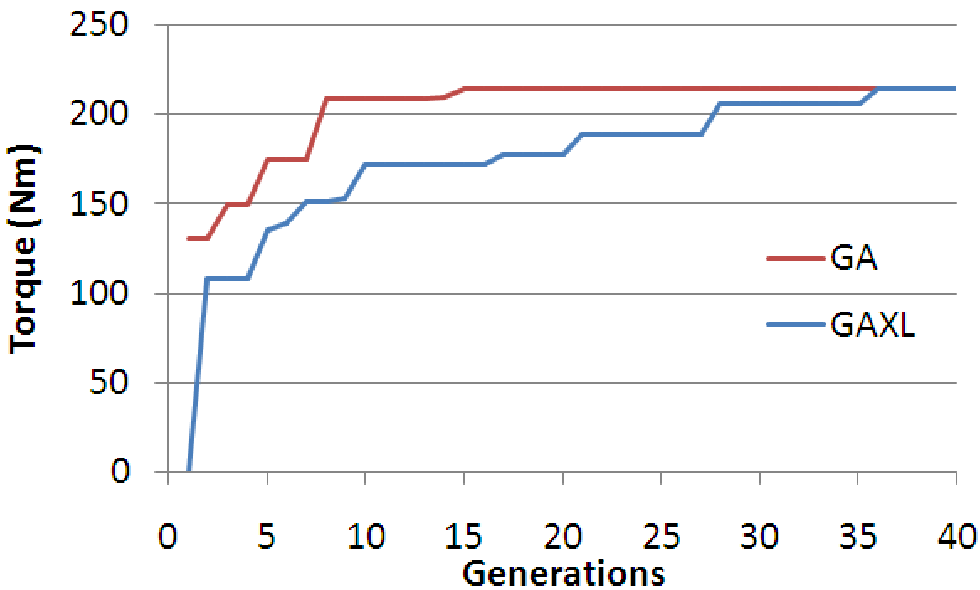


Figure 5.8 Maximum torque values per generation for 25-member population size

With the maximized torque value of 214.15 Nm, neither GA, nor GAXL is able to converge to the global optimum found by 100-member population, which is 254.9 Nm. However, the GAXL is able to converge to an optimum value slightly better than the one of GA in 40 generations, which satisfies the first objective of converging global optimum.

Regarding the second objective of the study, acceleration in the optimization cycle is aimed by GAXL, so that the expensive computational cost of the GAs would be decreased. Figure 5.9 represents the high-fidelity function evaluations needed to maximize torque. The number of high-fidelity function evaluations of simple GA is equal to the multiplication of the population size with the generation numbers. From the 1st till the 10th generation, GAXL requested high and low fidelity function evaluations simultaneously, and there is no direct proportionality between the number of generations versus high-fidelity function evaluations. After the 10th generation, the number of high-fidelity function evaluations is equal to the multiplication of the population size with the generation numbers, as in the case of simple GA.

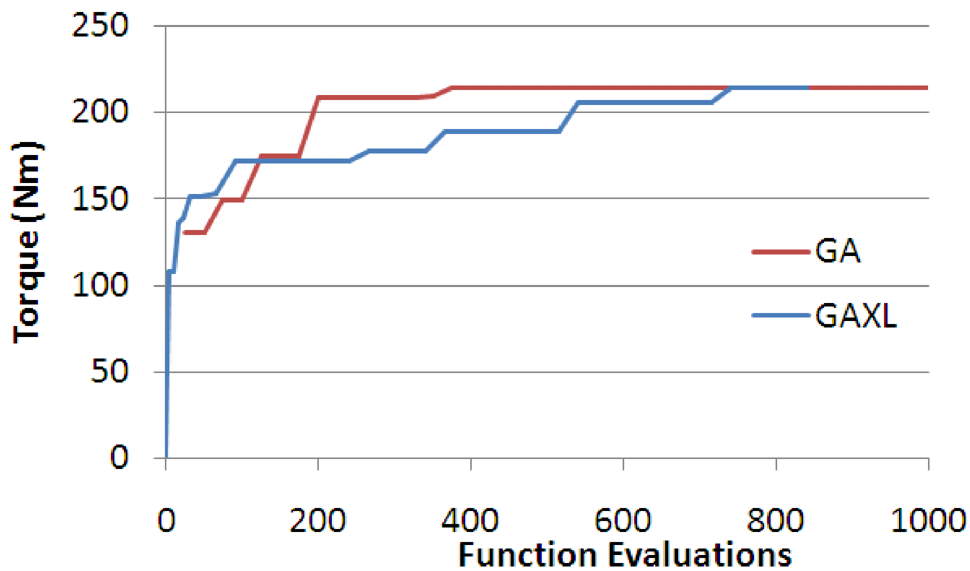


Figure 5.9 Computational Cost of Maximizing Torque

At the initial generations of the optimization cycle, GAXL converged rapidly than GA. The high-fidelity objective function value of GAXL was 171.56 Nm after only 91 function evaluations when compared to 149.05 Nm after 125 function evaluations of GA. Therefore, until the 5th generation of GA, which corresponds to 10th generation of GAXL in terms of number of high-fidelity objective function evaluations, GAXL converged faster and to a better torque value than the simple GA. This optimization part, from the 1st generation of GAXL to the 10th generation, is the only part of new GAXL operators performing on low fidelity objective function values.

As shown in Figure 5.10, after the 10th generation of GAXL, the outcomes of GAXL operators (selection, crossover and mutation) have no difference from the simple GA operators, since all individuals in the population have their high-fidelity fitness values assigned. Therefore, even the simple GA converged faster than GAXL after the 10th iteration; this is solely because of the random initial distribution of the two populations. Because, GAXL will have the same individuals in the successive generations with GA, as long as all of the individuals have their high-fidelity fitness values assigned.

The maximization of the efficiency problem is solved using a simple haploid GA and the diploid GAXL. The evolution of the 25-member population towards 40 generations is shown in Figure 5.11 for GA and GAXL. Contrary to torque optimization, GAXL clearly outperformed than GA in the entire optimization generations.

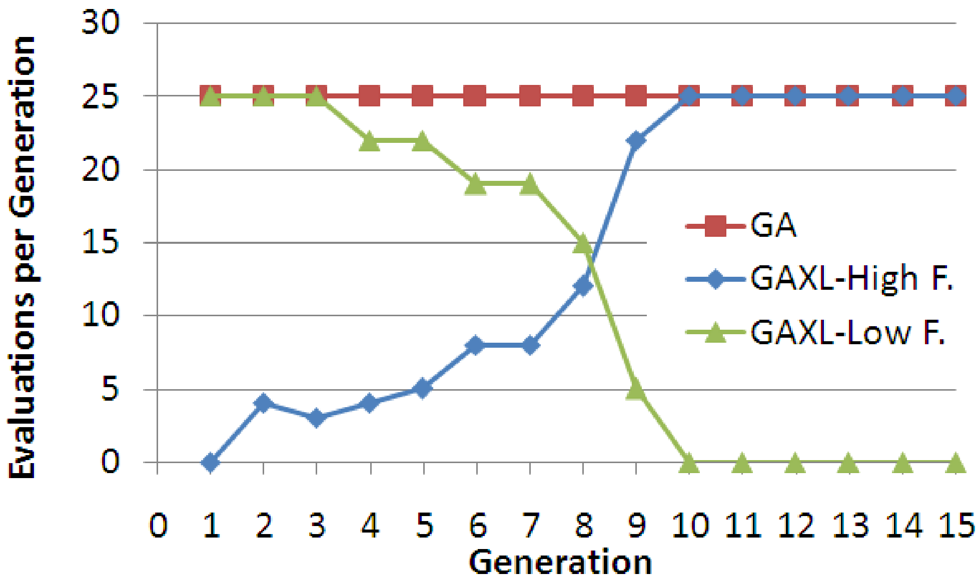


Figure 5.10 High and low fidelity function evaluations of GA and GAXL

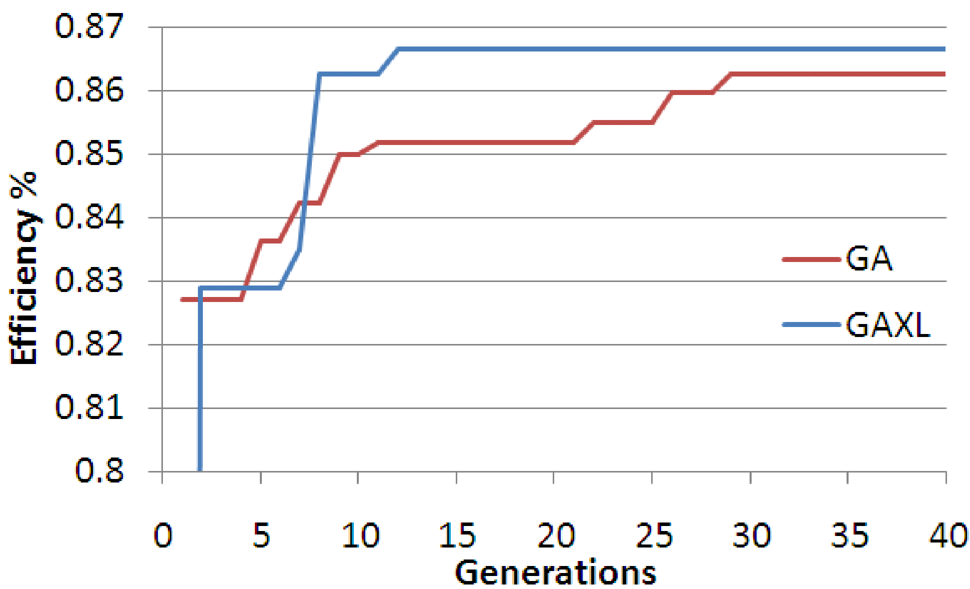


Figure 5.11 Maximum efficiency values per generation for 25-member population size

For this specific optimization case, GAXL did not evaluate the high-fidelity efficiency values for the first generation. From the 2nd till the 7th generation, some of the individuals are assigned high-fidelity efficiency values. In the remaining part of the optimization, GAXL algorithm switched to simple GA automatically since all of the individuals in the 7th population have their high-fidelity fitness values calculated. Therefore, low-fidelity calculations are not performed after the 7th generation as shown in Figure 5.12.

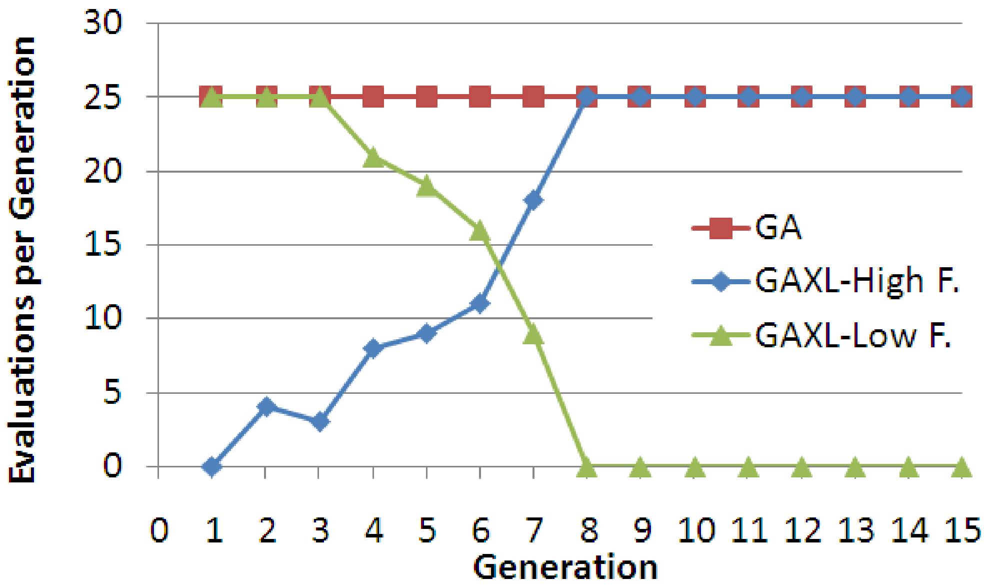


Figure 5.12 High and low fidelity function evaluations of GA and GAXL

With the maximized efficiency of 86.65 %, GAXL is able to converge to an optimum, which GA could not yet converge after 40 generations. The maximum efficiency found by GA optimization cycle after 40 generations is 85.19 %. Therefore, again in this optimization problem, the GAXL is able to converge better than simple GA in a 40-generation design loop, which satisfies the first objective.

Regarding the second objective of the study, the acceleration effect of GAXL is studied in Figure 5.13. GAXL is able to converge to the optimum efficiency blade in only 178 high-fidelity function evaluations. This corresponds to the 5th generation of a 25-population simple GA. In this case, the simple GA could not even converge to this efficiency value after 40 generations.

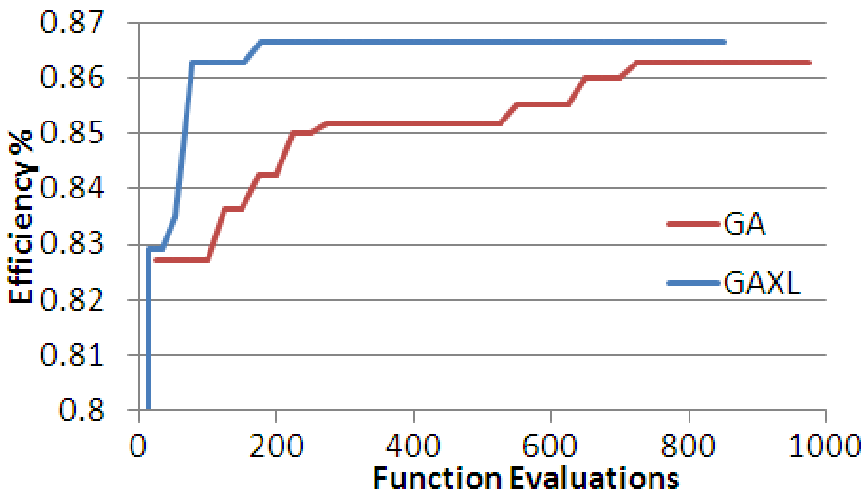


Figure 5.13 Computational cost of maximizing torque

In Figure 5.14 and 5.15, the importance of population size on the global convergence of GAXL is presented for torque and efficiency maximization problems, respectively. In the former problem, 25-member population GAXL could not converge to the global optimum in 100 generations, whereas, in the latter problem, population size did not affect the convergence of GAXL to the global optimum.

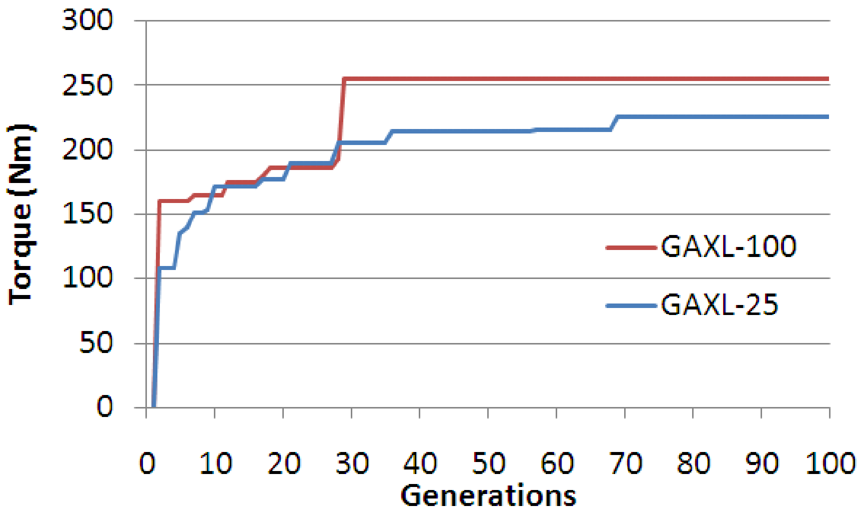


Figure 5.14 Effect of population size on the performance of GAXL – maximizing torque

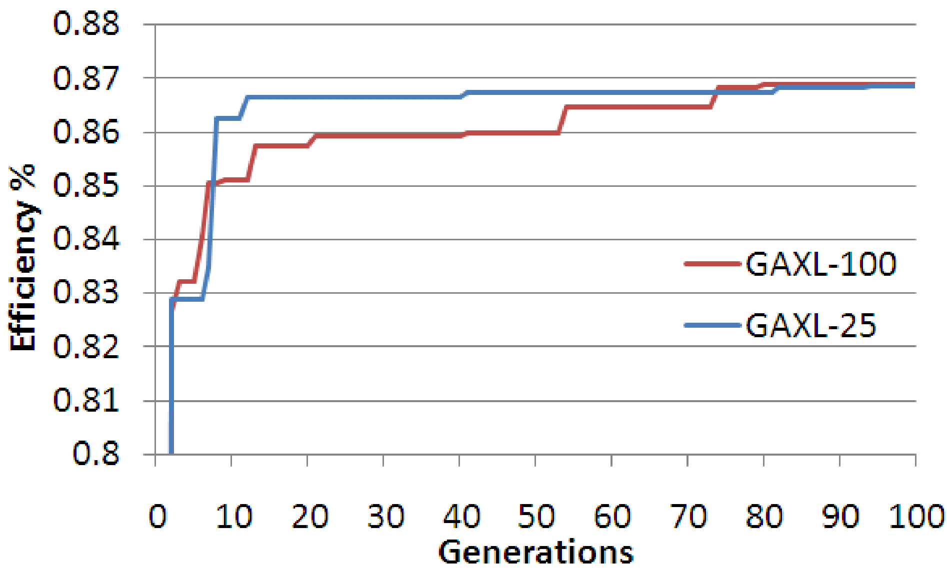


Figure 5.15 Effect of population size on the performance of GAXL – maximizing efficiency

However, in terms of computational cost, the number of generations does not give the correct interpretation. Figure 5.16 shows that a crowded population converges slowly at the beginning of the optimization cycle. For the maximization of torque problem, 25-individual population converged to a better torque value faster than 100-individual population.

The same case for the efficiency maximization is presented in Figure 5.17. In this case, again, the 25-member population converged to better efficiency values faster than the 100-member population. This indicates that, GAXL has inertia directly proportional to its population size, which has two effects. A large inertia GAXL will converge slowly but to a better optimum than a small inertia GAXL, which will converge faster at the initial generations, but will fail converging to global optimum in limited number of generations.

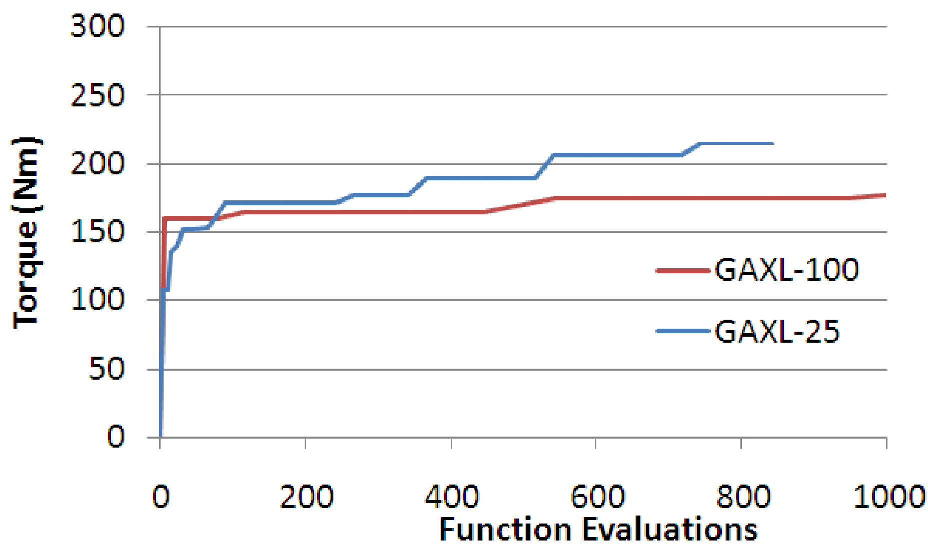


Figure 5.16 Computational cost comparison for different population size– torque maximization

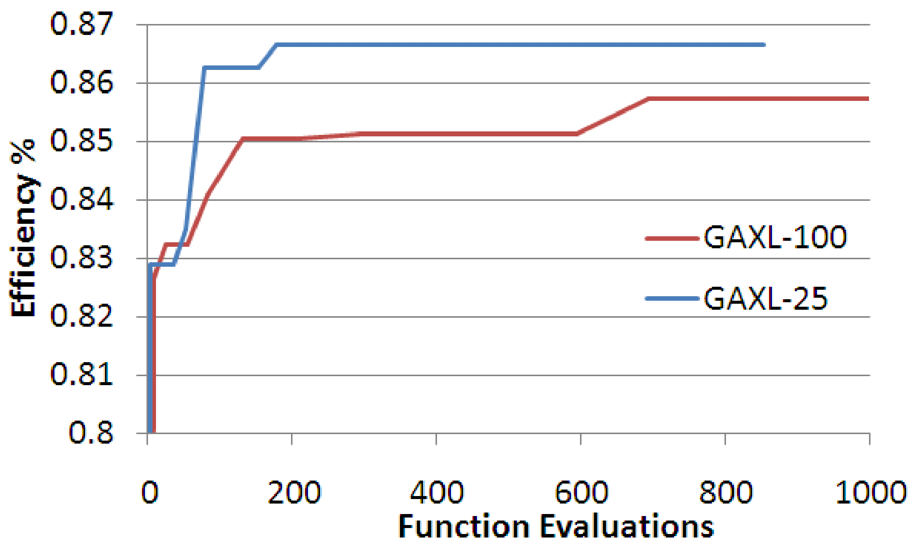


Figure 5.17 Computational cost comparison for different population size – efficiency maximization

The reason behind the faster convergence of low population size, but better convergence of high-population size is based on the initial generations of the optimization cycle. This is shown in Figure 5.18 and Figure 5.19 for the torque and efficiency maximization problems, respectively.

For both of the optimization problems, at the initial generations, 100-member population is saturated with high fidelity fitness values later than the 25-member population. In the torque optimization problem, 100-member population is saturated at the 11th generation, whereas 25-member population is saturated at the 10th generation. This early saturation of 25-member population allowed more high-fidelity computation calculations in the earlier generations per individual, which led to faster convergence than 100-member population.

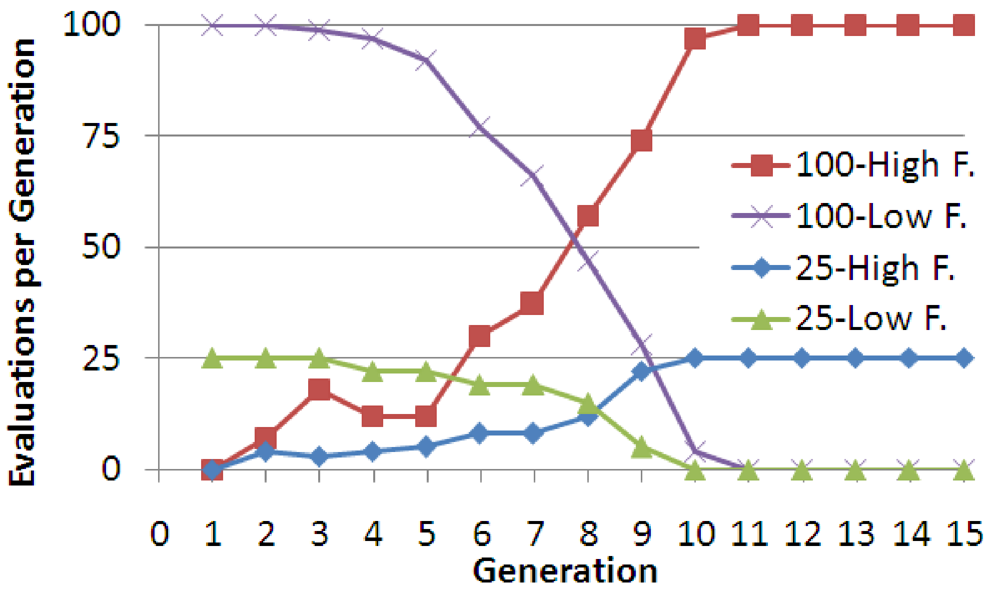


Figure 5.18 Effect of population size on high/low fidelity computations – torque maximization

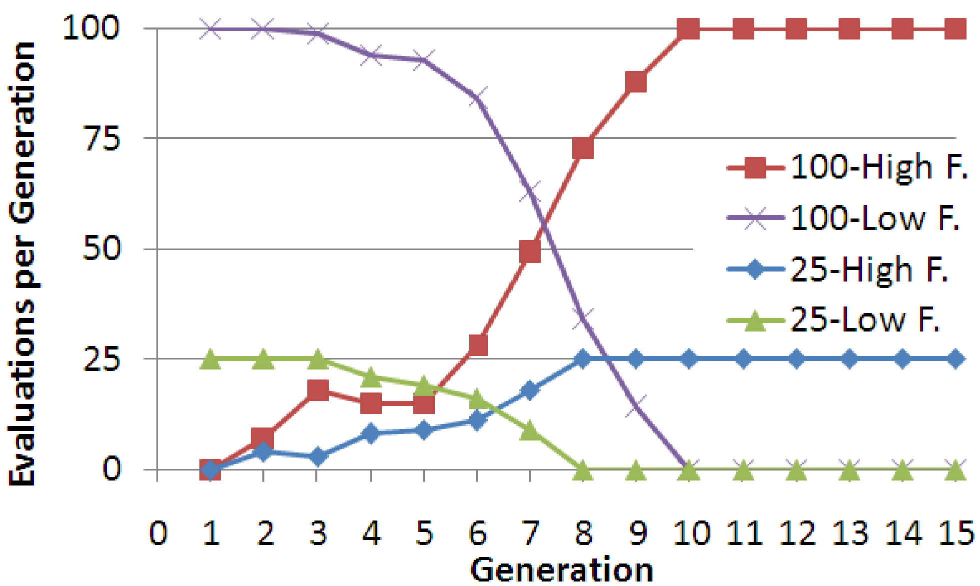


Figure 5.19 Effect of population size on high/low fidelity computations – torque maximization

Similarly, in the efficiency optimization problem, 100-member population is saturated at the 10th generation, whereas 25-member population is saturated at the 8th generation. The older saturation and much more low-fidelity function evaluations led to the fact that the individuals are guided by ancestors' schemas in the chromosomes, and accelerated the global convergence at the remaining generations, which in turn provided global convergence faster in terms of number of generations. In conclusion, the inertial effect of population size proved the ability of accelerated GAXL convergence to the global optimum.

5.1.4 The effect of confidence level parameter on the GAXL performance

The confidence level parameter, which is between 0 and 100 %, is introduced as an additional parameter in the GAXL. For the maximizing efficiency problem, the effect of different confidence level parameters is studied in this section. For this purpose, the confidence levels are selected as 5% and 50 % respectively. A 5% confidence level indicates a lower accuracy of the low-fidelity model, whereas a 50% confidence level indicates a better accuracy of the low-fidelity model.

As shown in Figure 5.20, 5% confidence level optimization run performed much better than 50% confidence level. High confidence level could not converge to the global optimum after 100 generations yet. Moreover, during the optimization cycle, low confidence level outperformed the higher one at nearly all generations.

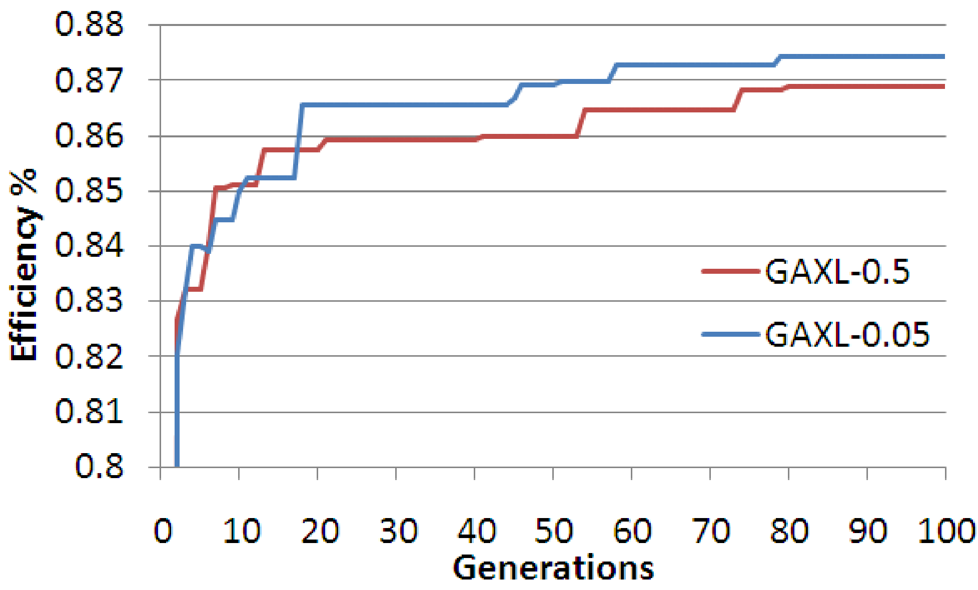


Figure 5.20 Effect of confidence level on the GAXL performance

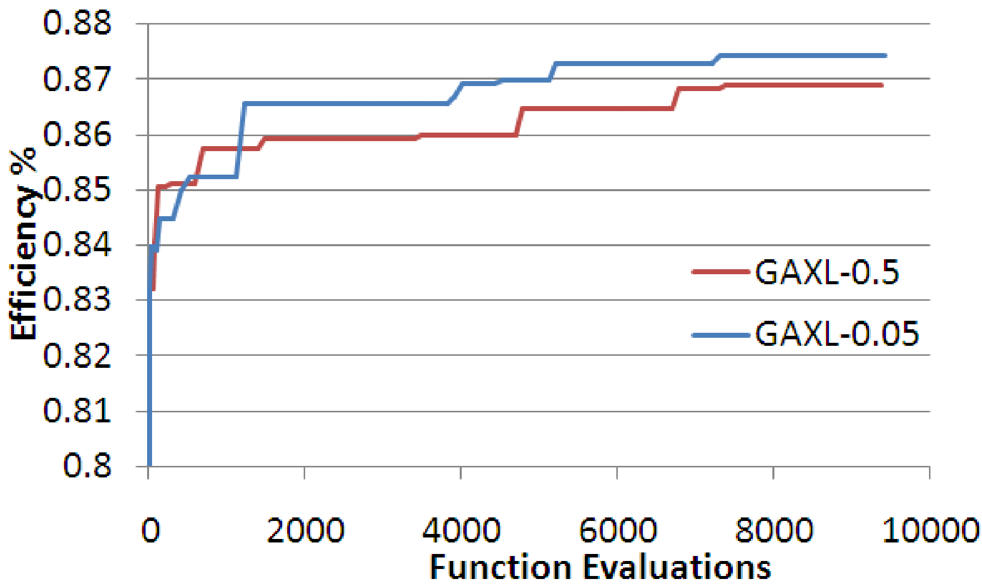


Figure 5.21 Effect of confidence level on the computational cost of GAXL

The computational costs of optimization for 5 and 50% confidence levels are compared in Figure 5.21. As in the case of number of generations, the figure indicates that the 5% confidence level converged much faster than 50%, and costs less than the latter.

The reason behind the better and faster convergence of 5% confidence level is investigated in Figure 5.23. In this figure, for both 5 and 50% confidence level value optimization runs, all of the individuals are saturated by high fidelity fitness values at the 10^{th} generation. Therefore, confidence level has no effect on the saturation generation number. However, the number of low fidelity function evaluations of 50% is higher than 5%, whereas, the number of high fidelity function evaluations is almost equal for both cases. This indicates that, usage of confidence level beyond the accuracy of the low-fidelity surrogate model decreases the acceleration effect of GAXL, and delays the global convergence.

5.2 Multi-objective Optimization

The baseline UH rotor blade is optimized for maximum torque and maximum adiabatic efficiency simultaneously. The optimization problem is solved using a haploid MOGA and a diploid MOGAXL. Table 5.5 shows the MOGA and MOGAXL parameter values used in the optimization processes.

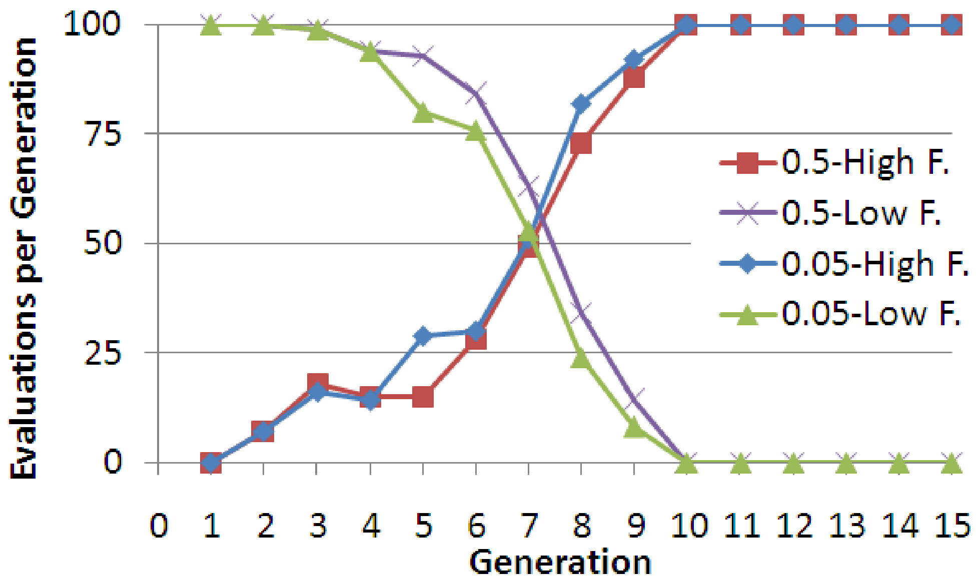


Figure 5.22 Effect of confidence level on the high/low function evaluations

5.2.1 Optimization Results

The high-fidelity Pareto-optimal frontiers after 100 generations of MOGA and MOGAXL are shown in Figure 5.23. The figure represents the trade-off between efficiency and torque and the decision maker may chose one blade shape out of all individuals in the Pareto-optimal frontier according to higher level information, such as stress calculations or rotor/stator interactions. In this figure, baseline rotor blade is also pointed for comparison. Baseline blade is non-dominated by all MOGA optimum solutions in the Pareto-front, and the baseline blade does not dominate any MOGA optimum solutions either. This indicates that, after 100 generations, MOGA did not converged sufficiently to the global optimum frontier yet.

Table 5.5 Optimization parameter values of MOGA and MOGAXL

Parameter	MOGA	MOGAXL
Number of Chromosomes	1	2
Design Parameters	37	37
Chromosome Length	219	219
Number of child	1	1
Maximum Generation	100	100
Population Size	100	100
Elitism	Yes	Yes
Cross-over Probability	90%	90%
Mutation Probability	2%	2%
Initial Latent Potential p_k	2	2
Low-fidelity Confidence Level	-	5%

On the other hand, MOGAXL succeeded finding as many as eleven non-dominated optimum blades that also dominates the baseline blade. Although the Pareto-optimal frontier of MOGAXL may not be the global frontier of the problem, MOGAXL will eventually converge to it by increasing the number of generations of the optimization cycle.

The diversity of the optimal solutions is a main concern in multi-objective optimization problems. In the present case, the non-dominated individuals are broadly spread from a maximum torque value of 227 Nm to a maximum efficiency value of 86.8 %. Although it is desired to find many non-dominated solutions in the objective space, because of the complexity of the problem, only eleven optimum solutions could be achieved. In case of increasing the number of generations of the optimization loop, MOGAXL may converge to a better Pareto-frontier with more non-dominated blade shapes.

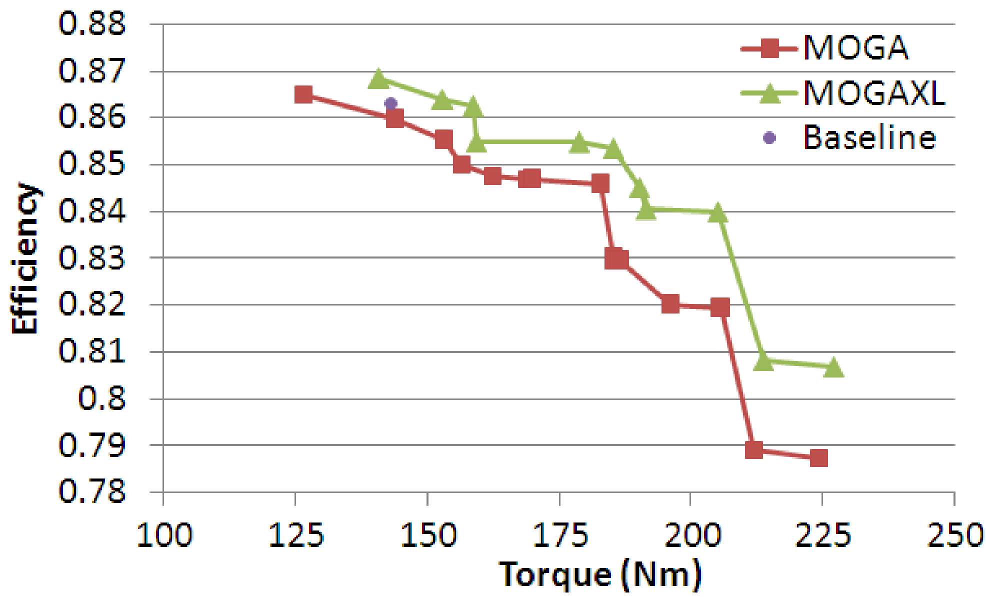


Figure 5.23 Pareto optimal frontiers of MOGA and MOGAXL after 100 generations

From the beginning of the first generation until the end of the optimization, the number of non-dominated individuals in the elite members set is given in Figure 5.24. Only high-fidelity fitness assigned elite members of MOGAXL are counted per generation. At the initial generations, MOGAXL found as twice as much as elite members compared to MOGA. Besides MOGA has fifteen elite members at the end of the generation, all of these members are dominated by MOGAXL elite members.

The second objective of this paper is the acceleration of the optimization cycle in order to decrease the total computational cost. This cost is evaluated by counting the number of high-fidelity objective function evaluations in the optimization process. Figure 5.25 shows the number of high and low fidelity evaluations with respect to the generation number of the first fifteen generations.

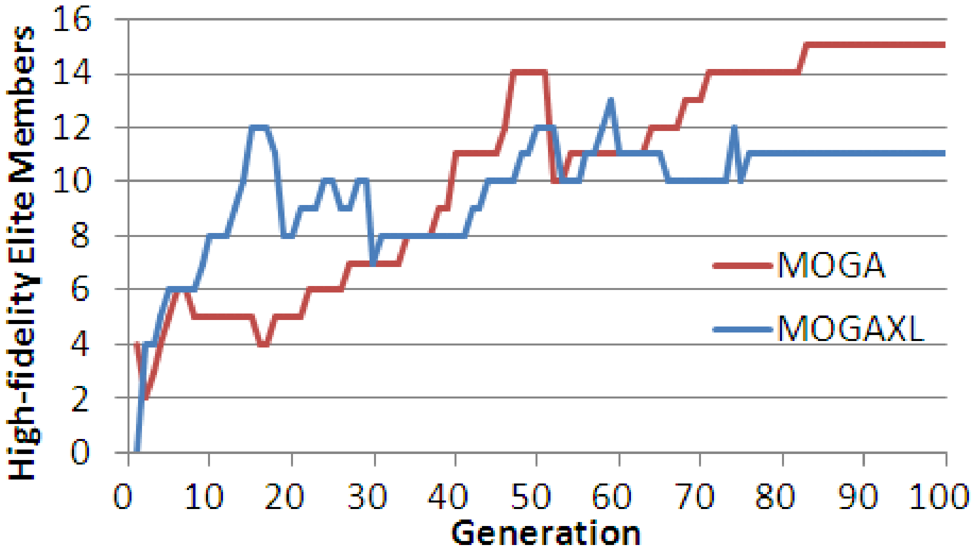


Figure 5.24 Number of elite members in the EMS-H

MOGAXL algorithm operates exactly same as the MOGA algorithm after all of the individuals in the population are assigned their high-fidelity fitness values, which is called as the saturation of individuals with high-fidelity fitness values. In this specific case, MOGAXL assigned the high-fidelity fitness values to every individual of the 10th generation population. After 10th generation, both MOGA and MOGAXL will perform a hundred evaluations at each generation which corresponds to the population size.

At the initial nine generations, MOGAXL used very little computational resources when compared to MOGA. In these 9 generations 293 and 900 high fidelity RANS solutions are requested by MOGAXL and MOGA, respectively. Consequently, the 9th generation of MOGAXL costs less than the 3rd generation of MOGA in terms of computational expenses. Therefore, in order to compare the Pareto-optimal frontiers of the both algorithms fairly, the 9th generation EMS-H of MOGAXL and the 3rd generation EMS of MOGA are compared in Figure 5.26.

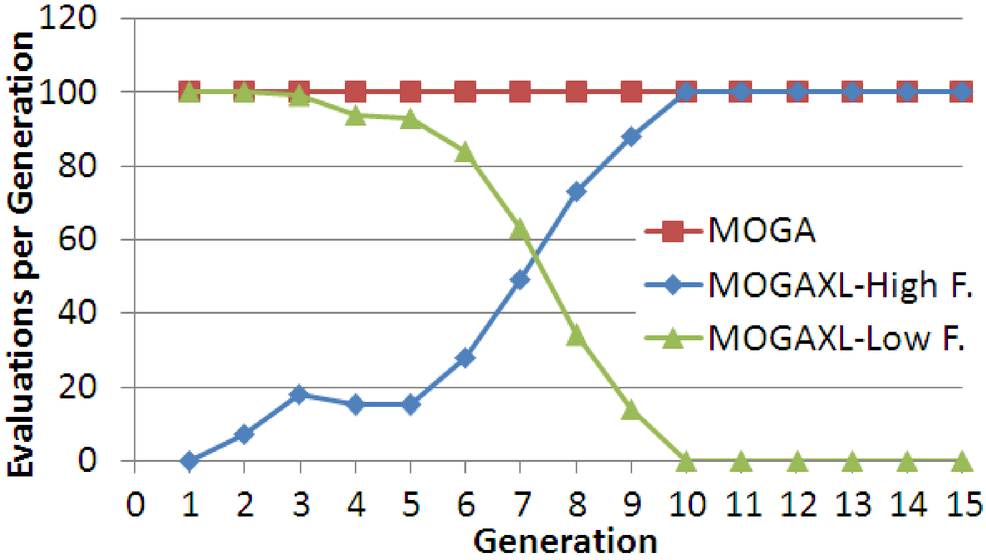


Figure 5.25 Computational cost comparison

According to Figure 5.26, MOGA could only find three elite members in the Pareto-optimal frontier, and only one of them is non-dominated by MOGAXL Pareto-optimal frontier. Whereas, MOGAXL has already eight non-dominated members, and only two of them are dominated by MOGA. Additionally, MOGAXL has an excellent distribution of elite members when compared to the diversity of the simple MOGA.

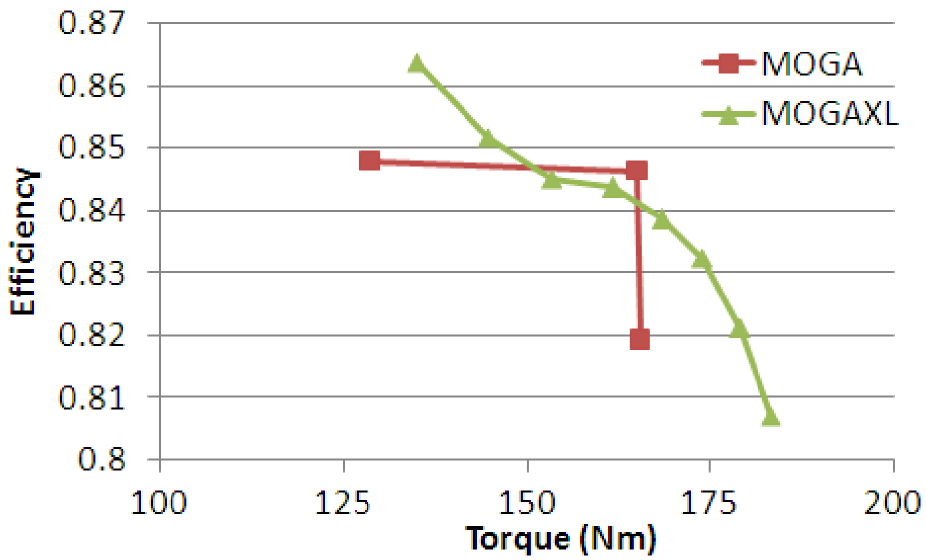


Figure 5.26 Pareto optimal frontiers after 3 generations of MOGA and 9 generations of MOGAXL

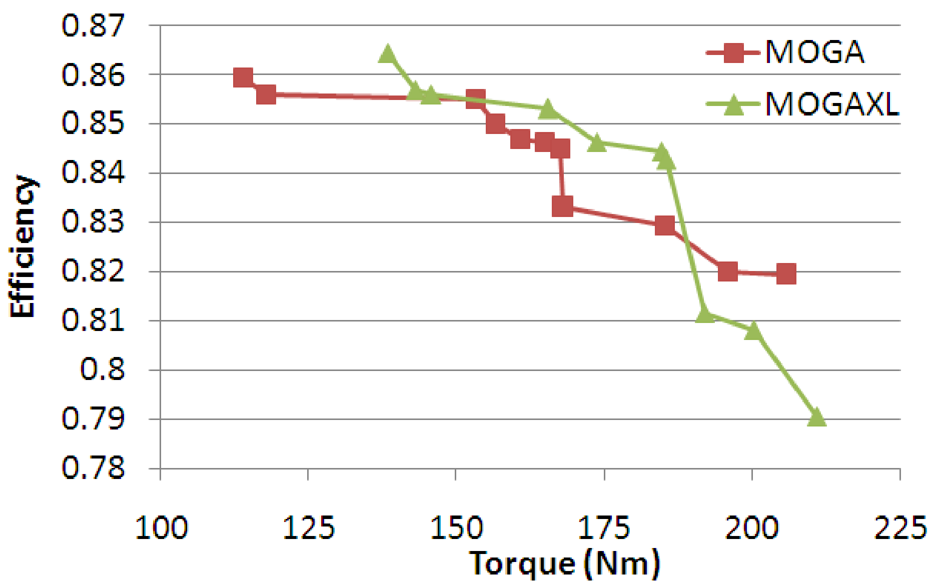


Figure 5.27 Pareto optimal frontiers after 40 generations of MOGA and 46 generations of MOGAXL

During the successive new generations of optimization cycle, the same analysis is performed for the 40th generation of MOGA. In terms of number of high fidelity computations, this corresponds to the 46th generation of MOGAXL. The Pareto-optimal frontiers are shown in Figure 5.27. There are eleven elite members in the EMS of MOGA and ten elite members in the EMS-H of MOGAXL. Only two of the MOGAXL elite members are dominated by MOGA, and the remaining eight members are the non-dominated solutions of the problem for this computational cost. This clearly indicates the accelerated convergence of MOGAXL to the global Pareto-optimal frontier. Moreover, the diversity of the MOGAXL elite members is better the MOGA elite members.

5.2.2 Effect of Population Size on MOGA Performance

The effect of population size is studied on the MOGA by solving the same multi-objective optimization problem. The MOGA solutions with 50 and 100 individual populations are presented with Pareto-optimal frontiers after 100 generations in Figure 5.28, MOGA-50 and MOGA-100, respectively. There are nine elite members in the EMS of 50-individual population when compared to fifteen elite members in the EMS of 100-individual population solutions. Five members of the each optimization are non-dominated by any other elite members. Therefore, in terms of number of non-dominated elite members, both population sizes have the same number of non-dominated members.

In terms of Pareto-optimal frontier diversity, 100-individual population solution keeps a much better diversity of elite members in the objective space. Lowering the population size has an adverse effect on the diversity of the optimization solution. 50-individual Pareto-frontier could only find one elite member below 83% efficiency and this member is already dominated by 100-individual members.

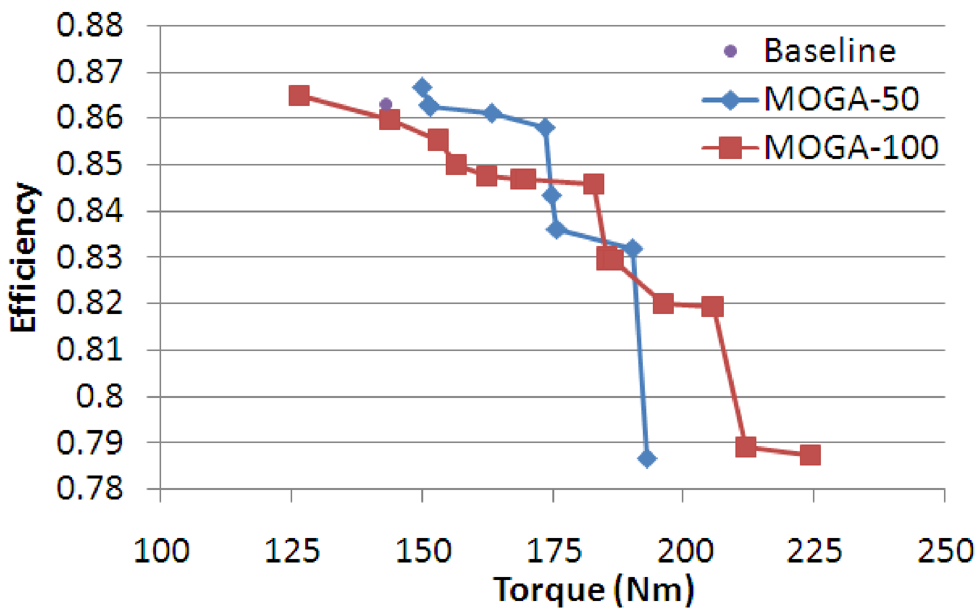


Figure 5.28 Effect of population size on MOGA performance

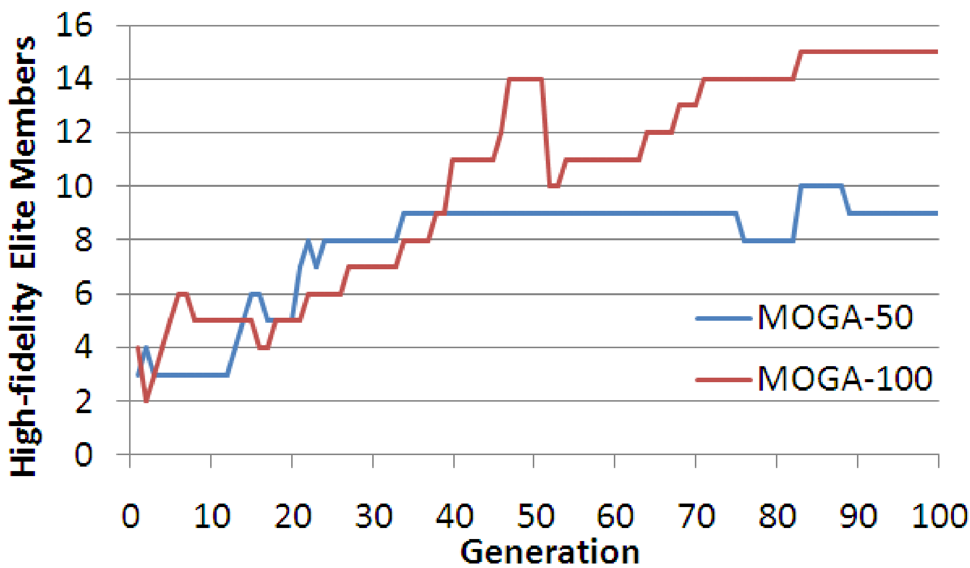


Figure 5.29 Number of elite members in the EMS

From the beginning of the first generation until the end of the optimization, the number of non-dominated individuals in the elite members set is compared in Figure 5.29. Until the 40th generation, MOGA-50 and MOGA-100 found as much as the same number of elite members, which monotonically increases with consecutive generations. However, this increase stops for MOGA-50 afterwards, and the number of elite members in the EMS remains the same. On the other hand, MOGA-100 continues to increase the elite members monotonically until the end of the optimization. This suggests that, lower population size MOGA suffers from the convergence and divergence problems during the optimization cycle.

5.2.3 Effect of Population Size on MOGAXL Performance

The effect of population size is studied on the MOGAXL by solving the same multi-objective optimization problem. The MOGAXL solutions with 25, 50 and 100 individual populations are presented with Pareto-optimal frontiers after 100 generations in Figure 5.30, MOGAXL-25, MOGAXL-50 and MOGAXL-100, respectively. There are eight elite members in the EMS-H of 25-individual population when compared to fourteen elite members in the EMS-H of 50-individual population solutions, and eleven elite members in the EMS-H of 100-individual population solutions. None of the elite members of these solutions are dominated by baseline blade. Therefore, the Pareto-optimal frontiers of these different population size solutions provide a better solution to the problem than the baseline blade. The closest global optimum frontier is found by MOGAXL-100, however the distance between MOGAXL-100 frontier and the MOGAXL-25 and MOGAXL-50 frontiers are comparable small compared to the convergence effect of population size on MOGA. Only one elite member of MOGAXL-100 is dominated by the other two solutions' elite members.

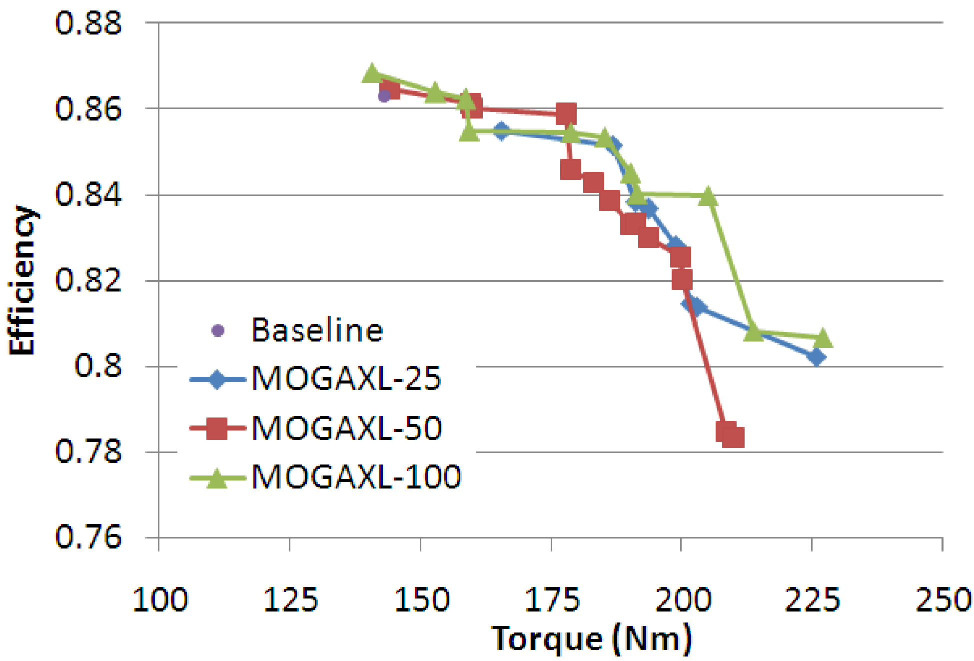


Figure 5.30 Effect of population size on MOGAXL performance

Concerning the diversity of the Pareto-optimal frontiers, it is found that the elite members of all solutions have a very good distribution in the objective space. Contrary to diversity suffering of MOGA by decreasing the population size, MOGAXL does not poses any performance degradation in terms of elite members' diversity. Below the 83% efficiency, MOGAXL-50 could generate five elite members, MOGAXL-25 could generate four elite members, compared to only one elite member of MOGA-50 after 100 generations.

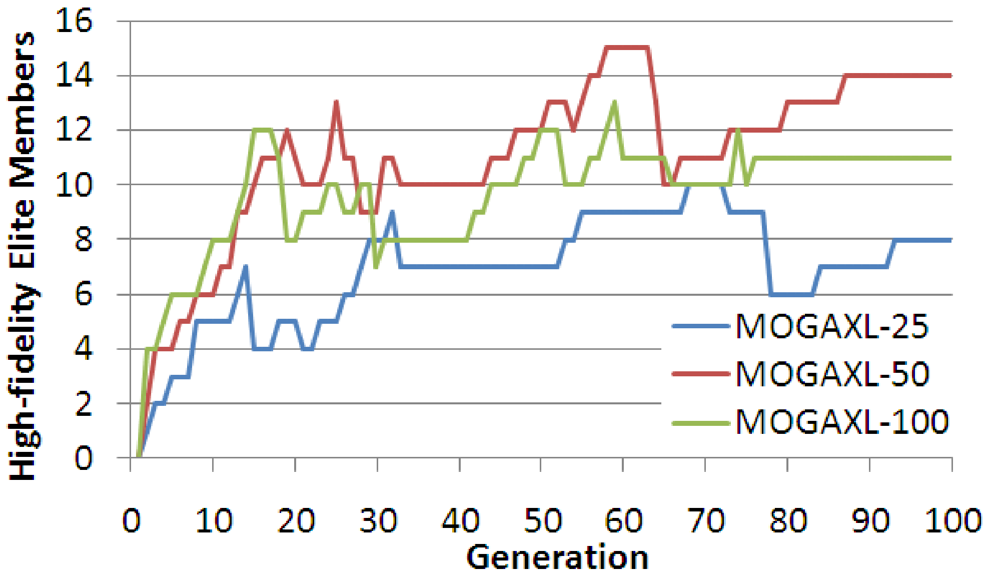


Figure 5.31 Number of elite members in the EMS-H

From the beginning of the first generation until the end of the optimization, the number of non-dominated individuals in the elite members set is compared in Figure 5.31. Until the 30th generation, MOGAXL-25, MOGAXL-50 and MOGAXL-100 found as much as the same number of elite members, which monotonically increases with consecutive generations. Afterwards, the MOGAXL-100 looks like having more number of elite members in the EMS-H, however, again, all three optimizations found the same number of elite members until the 75th generation. Beyond this point, MOGAXL-100 elite member size monotonically increases, MOGAXL-50 elite member size remains the same, and MOGAXL-25 elite member size decreases. This confirms that MOGAXL-25 and MOGAXL-50 Pareto-frontiers did not yet converged to the global optimum close enough and therefore, some new elite members in the new generations dominates many of the elite members in the EMS-H, preventing the size of EMS-H to increase. The convergence of MOGAXL-100 is simply because of twice and quadruple number of function evaluations after 100 generations compared to MOGAXL-50 and MOGAXL-25, respectively.

Finally, the effect of population size is compared between MOGA and MOGAXL by studying a 50-individual population optimization case. As shown in Figure 5.32, even in the low population size, MOGAXL clearly outperformed, converged to a better frontier, and a more diverse elite member distribution with much higher number of elite members. MOGA could only find 9 elite members compared to 14 elite members of MOGAXL after 100 generations. Only one elite member of MOGAXL is dominated by MOGA, and only one elite member of MOGA is non-dominated by MOGAXL.

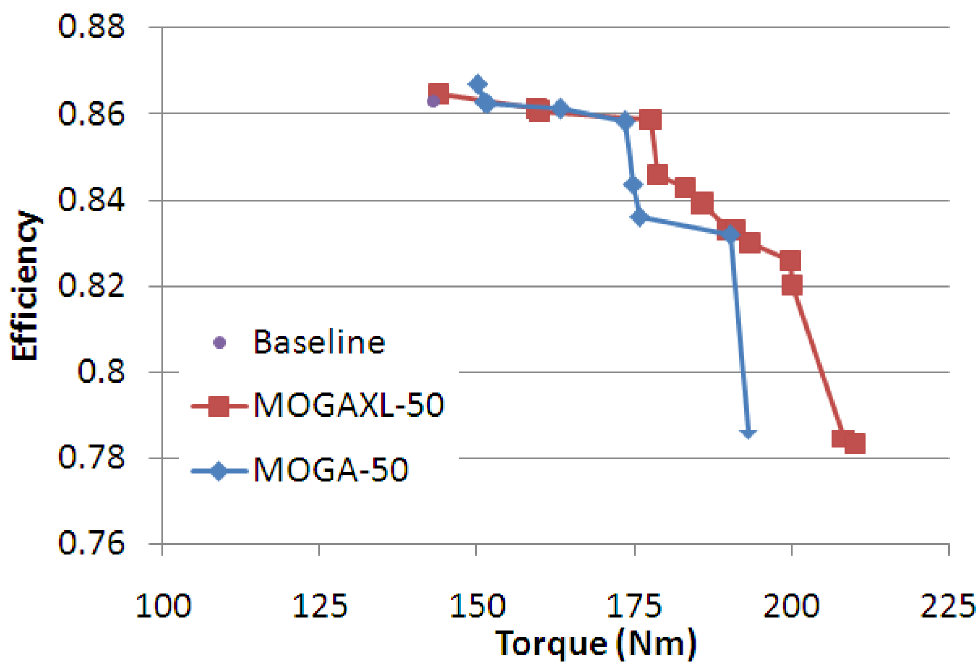


Figure 5.32 Effect of population size on MOGA/MOGAXL performance

In terms of elite members in the EMS of MOGA and EMS-H of MOGAXL, both algorithms increase the size until the 40th generation. Afterwards, MOGAXL clearly continue to improve the elite member size by converging to the global optimum while increasing the diversity of the EMS-H members. On the other hand, MOGA is out of breath after the 40th generation, and could not manage to increase the population size timely as shown in Figure 5.33.

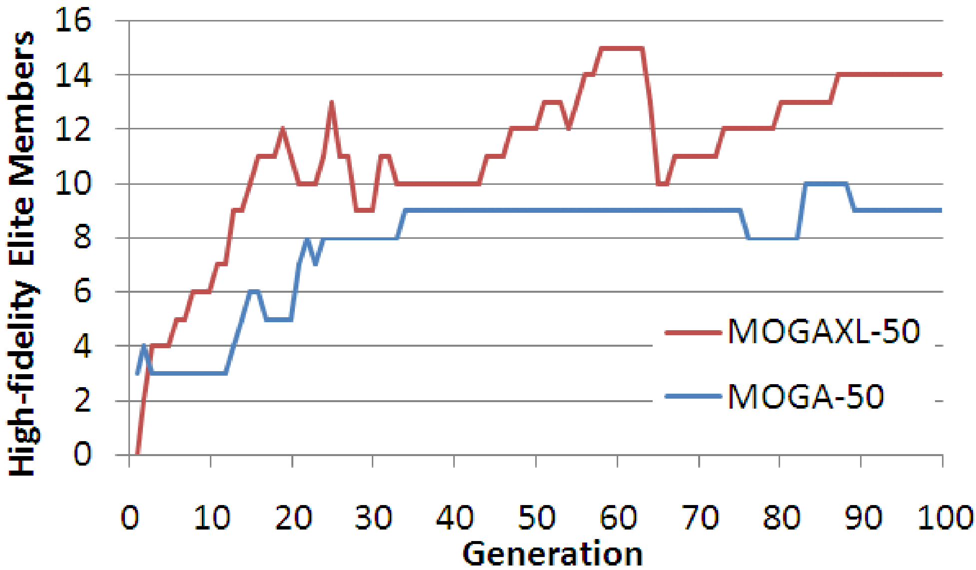


Figure 5.33 Number of elite members in the EMS/EMS-H

CHAPTER 6

CONCLUSION

A GA-based multi-objective aerodynamic shape design optimization tool capable of handling surrogate models for a turbine blade has been developed and demonstrated on a baseline test case.

A blade reshaping algorithm is constructed to parameterize three-dimensional blade shapes. 37 design parameters are used to create one blade shape at a time. Leaned, twisted, bended, and tapered blades can be obtained using the reshaping algorithm.

A commercial RANS flow solver is coupled with a commercial automated grid generator for flow field solution. Two well-known test cases are solved for the validation of the coupled flow solver. Highly three dimensional VKI nozzle test case solution demonstrated the secondary flow and loss prediction capabilities of the solver and the pre-selected CFD models. First stage of the UH high pressure turbine test case is solved to validate the off-design prediction capability of the coupled solver. Successful demonstration of the predictions is shown for mass flow and RPM variations.

For the purpose of obtaining high and low fidelity model solutions of the objective functions, the parameters of the automatic grid generator are modified. Two mesh structures with different quality grids are obtained by changing the spacing between the nodes and increasing the number of volumetric elements in the flow field. The effect of coarse and fine grid on the flow field solution is investigated using the VKI test case. It was observed that, the coarse grid solution can also be used to predict the flow field scarifying a level of accuracy. Therefore, according to the demand of the optimization algorithm, fine or coarse grid RANS solutions of a blade shape are obtained for high or low fidelity objective function values, respectively.

To eliminate the genetic algorithm performance degradations when used with surrogate models, a multiploid genetic algorithm GAXL is developed and applied for single objective optimization problems. Assuring the convergence to the global optimum, and lowering the computational cost of optimization cycle are selected as the two key performance degradations.

Maximization of blade torque and maximization of blade adiabatic efficiency problems are selected as two objectives, and solved separately using a conventional GA and the new GAXL. For both optimization problems, GAXL converged to a better performing blade shape than that of GA. For the torque optimization problem, GAXL converged almost twice faster than GA, and therefore computational cost is halved. In the efficiency optimization problem, again, GAXL accelerated the convergence by 40%, and lower corresponding computational cost.

Both optimization problems revealed the fact that, GAXL operators operate only at the initial ten to twelve generations, until when all the individuals are saturated by high-fidelity fitness values. From that point on, GAXL operators act just the same as the GA operators. However, the accelerated behavior of GAXL does not only belong to these initial generations, but also to the remaining generations. The schemes of the low fidelity level chromosomes that are formed in the initial generations work during the entire optimization cycle, and guide the population towards the global optimum by decreasing the necessary high-fidelity objective function evaluations.

The effect of population size on GAXL and GA performance is studied for the same problems. The population size is decreased from 100 to 25 individuals. First, GAXL and GA algorithms are compared. Both algorithms converged to the same torque value after 40 generations. However, when checking the first 10 generations, GAXL has already converged to a better blade shape than GA in less number of high-fidelity function evaluations. In case of efficiency optimization, GAXL clearly outperformed GA in terms of optimum blade shapes and computational cost.

The effect of population size on GAXL performance is studied for the same problems. In the torque problem, 25-member population GAXL could not converge to the global optimum in 100 generations, whereas, in the efficiency problem, population size did not affect the convergence of GAXL to the global optimum. However, in terms of computational cost, the number of generations does not give the correct interpretation. For the maximization of torque problem, 25-individual population converged to a better torque value faster than 100-individual population.

These results indicate that, GAXL has inertia directly proportional to its population size, which has two effects. A large inertia GAXL will converge slowly but to a better optimum than a small inertia GAXL, which will converge faster at the initial generations, but will fail to converge to the global optimum in limited number of generations. In conclusion, the inertial effect of population size proved the ability of accelerated GAXL convergence to the global optimum.

The effect of confidence level parameter is demonstrated using the maximization of efficiency problem. The computational costs of optimization for 5 and 50% confidence levels are compared and the 5% confidence level converged much faster than 50%, and costs less than the latter. Investigation of the reason behind the better and faster convergence of 5% confidence level revealed that usage of confidence level beyond the accuracy of the low-fidelity surrogate model decreases the acceleration effect of GAXL, and delays the global convergence.

The multiploid genetic structure is applied for multi-objective problems as well. For this purpose, the baseline UH rotor blade is optimized for maximum torque and maximum adiabatic efficiency simultaneously. The optimization problem is solved using a haploid MOGA and a diploid MOGAXL. The high-fidelity Pareto-optimal frontiers after 100 generations of MOGA and MOGAXL are compared, and it is found that baseline blade is non-dominated by all MOGA optimum solutions in the Pareto-front, but the baseline blade does not dominate any MOGA optimum solutions either. This indicates that, after 100 generations, MOGA did not converge sufficiently to the global optimum frontier yet. In other words, the number of high-fidelity function evaluations is not sufficient for MOGA to converge to the global optimum.

On the other hand, MOGAXL succeeded finding as many as eleven non-dominated optimum blades that also dominate the baseline blade. Although the Pareto-optimal frontier of MOGAXL may not be the global frontier of the problem, MOGAXL will eventually converge to it by increasing the number of generations of the optimization cycle. This confirms the accelerated global convergence behavior of MOGAXL.

Although it is desired to find many non-dominated solutions in the objective space, because of the complexity of the problem, only eleven optimum solutions could be achieved. In case of increasing the number of generations of the optimization loop, MOGAXL may converge to a better Pareto-frontier with more non-dominated blade shapes. Diversity of the MOGAXL elite members found to be much better distributed when compared to MOGA elite members.

The effect of population size is studied on the MOGA by solving the same multi-objective optimization problem for 50 and 100 individual populations. In terms of number of non-dominated elite members, both population sizes have the same number of non-dominated members. In terms of Pareto-optimal frontier diversity, 100-individual population solution keeps a much better diversity of elite members in the objective space. Lowering the population size has an adverse effect on the diversity of the optimization solution.

The effect of population size is studied on the MOGAXL by solving the same multi-objective optimization problem. The MOGAXL solutions with 25, 50 and 100 individual populations are compared with Pareto-optimal frontiers after 100 generations, MOGAXL-25, MOGAXL-50 and MOGAXL-100, respectively. The Pareto-optimal frontiers of these different population size solutions provide a better solution to the problem than the baseline blade. The closest global optimum frontier is found by MOGAXL-100, however the distance between MOGAXL-100 frontier and the MOGAXL-25 and MOGAXL-50 frontiers are comparably small compared to the convergence effect of population size on MOGA. Only one elite member of MOGAXL-100 is dominated by the other two solutions' elite members. Concerning the diversity of the Pareto-optimal frontiers, it is found that the elite members of all solutions have a very good distribution in the objective space. Contrary to diversity suffering of MOGA by decreasing the population size, MOGAXL does not pose any performance degradation in terms of elite members' diversity.

Finally, the effect of population size is compared between MOGA and MOGAXL by studying a 50-individual population optimization case. Even in the low population size, MOGAXL clearly outperformed, converged to a better frontier, and a more diverse elite member distribution with much higher number of elite members.

REFERENCES

[1] Lightlill, J. M., "A New Method of Two-Dimensional Aerodynamic Design", ARC R&M 2112, 1945.

[2] Stanitz, J. D., "Design of Two-Dimensional Channels With Prescribed Velocity Distributions Along the Channel Walls", NACA Report 1115, 1953.

[3] Dulikravitch, D. S., "A Computational Design Method For Transonic Turbomacinery Cascades", ASME Paper 82-GT-117, 1982.

[4] Hawthorne, W.R., Tan C.S., Wang C., and McCune J.E., "Theory of Blade Design For Large Deflections: Part I: Two Dimensional Cascades", Journal of Eng. For Gas Turbines and Power, pp. 346-353, 1984.

[5] Borges, J., "A Proposed Through-Flow Inverse Method For the Design Of Mixed Flow Pumps", Int. Journal for Numerical Methods in Fluids, pp. 1097-1114, 1993.

[6] Zangeneh, M., "Inviscid- Viscous Interaction Method for 3D Inverse Design of Centrifugal Impellers", ASME Paper 94-GT-103, 1993.

[7] Zangeneh, M., "Design Modification Of Rotor 67 by 3D Inverse Method – Inviscid Flow Limit", ASME Paper 97-GT-484, 1997.

[8] Leonard, O., "Subsonic and Transonic Cascade Design", AGARD R-780, Special Course on Inverse Methods for Airfoil Design for Aeronautical and Turbomachinery Applications, pp. 7.1-7.18, 1990.

[9] Bartelheimer, W., " An Inverse Method For The Design of Transonic Airfoils and Wings", Journal of Inverse Problems for Engineering, pp. 21-51, 1996.

[10] Meauze, G., "An Inverse Time Marching Method for the Definition of Cascade Geometry", Journal of Engineering for Power, pp. 650-656, 1982.

[11] Demeulenaere, A., "An Inverse Method Concept for the Turbomachinery Blade Generation", PhD. Thesis, VKI, 1997.

[12] Ta'asan S., "Trends in Aerodynamics Design Optimization: A Mathematical Viewpoint", AIAA Paper 95-1731, June 1995.

[13] Blaize M., Knight D., and Rasheed K., "Automated Optimal-Design of 2-Dimensional Supersonic Missile Inlets", Journal of Propulsion and Power, Vol. 14, No. 6, 890-898, 1998.

[14] Vanderplaats G.N., "Numerical Optimization Techniques for Engineering Design", McGraw-Hill, 1984.

[15] Misegades K.P., "Optimization of Multi-element Airfoils", VKI PR 1980-5, 1980.

[16] Van Laarhoven P.J.M., Aarts E.H.L., "Simulated Annealing: Theory and Applications" Kluwer Academic Publishers, 1987.

[17] Aly S., Marconi F., Ogot M., Pelz R., Siclari M., "Stochastic Optimization Applied To CFD Shape Design", AIAA-95-1647-CP, pp. 11-20, 1995.

[18] Tekinalp O., Karslı G., "A new Multiobjective Simulated Annealing Algorithm", Journal of Global Optimization, 2006.

[19] Kirkpatrick, S., Gelatt, C.D., Vechi, M.P., "Optimization by simulated annealing", Science, vol. 220(4598), pp. 671–680, 1983.

[20] Goldberg, D. E., Genetic Algorithms in Search, Optimization, and Machine Learning, Reading, Addison-Wesley, 2-6, 1989.

[21] Holt G, Bassler S., "Preliminary Design of Axial Compressors Using Artificial Intelligence and Numerical Optimization Techniques", ASME Paper 91-GT-334, 1991.

[22] De Falco I., "An Introduction to Evolutionary Algorithms and Their Application to Aerofoil Design Problem", VKI LS 1997-05, 1997.

[23] Quagliarella D., "Genetic Algorithm Applications in Computational Fluid Dynamics", Genetic Algorithms in Engineering And Computational Science, pp. 417-442, John Wiley & Sons, 1995.

[24] Periaux J., Sefrioui, M, Stoufflet B., "Robust Genetic Algorithms for Optimization Problems in Aerodynamic Design", Genetic Algorithms in Engineering And Computational Science, pp. 371-396, John Wiley & Sons, 1995.

[25] Trigg M.A, Tubby G.R., Sheard A. G., "Automatic Genetic Optimization Approach to 2D Blade Profile Design For Steam Turbines", Journal of Turbomachinery, pp. 11-17, 1999.

[26] O. Oksuz, I.S. Akmandor, M. S. Kavsaoglu, "Aerodynamic Optimization of Turbomachinery Cascades Using Euler/Boundary Layer Coupled Genetic Algorithms", Journal of Propulsion and Power, Vol.18, No.3, May-June 2002, pp.652-657.

[27] Obayashi S., Takanashi S., "Genetic Optimization of Target Pressure Distributions for Inverse Design Methods", AIAA 95-1649-CP, pp. 33-42, 1995.

[28] Valentin V., "Three Dimensional Optimization Of a Compressor Blade at Design and Off-design Operations", VKI- PR-1998-26, 1998.

[29] Poloni C., "Hybrid GA for multi-objective aerodynamic shape optimization", Genetic Algorithms in Engineering And Computational Science, pp. 397-415, John Wiley & Sons, 1995.

[30] K. Chiba, S. Obayashi, K. Nakahashi, "Design Exploration of Aerodynamic Wing Shape for Reusable Launch Vehicle Flyback Booster", Journal of Aircraft, Vol.43, No. 3, pp. 832-836, 2006.

[31] O. Oksuz, I.S. Akmandor, "Turbine Cascade Optimization Using an Euler Coupled Genetic Algorithm", ISABE-2001-1234, 2001.

[32] Denton, J. D., "Designing in three Dimensions", AGARD-LS-195, 1994.

[33] Holt G., Bassler S., "Preliminary Design of Axial Compressors Using Artificial Intelligence and Numerical Optimization Techniques", ASME 91-GT-334, 1991.

[34] Tong S. S., Gregory B. A., "Turbine Preliminary Design Using Artificial Intelligence and Numerical Optimization Techniques", Journal of Turbomachinery, pp. 1-7, January 1992.

[35] Shelton M.L., Gregory B.A., Lamson S.H., Moses H.L., Doughty R.L., Kiss T., "Optimization of a Transonic Turbine Arifoil Using Artificial Intelligence, CFD, and Cascade Testing", ASME 93-GT-161, 1993.

[36] Goel S., Cofer J.I., Singh H., "Turbine Airfoil Design Optimization", ASME 96-GT-158, 1996.

[37] Cofer J.I., "Advances in Steam Path Technology", Journal of Engineering For Gas Turbines And Power, pp. 337-352, April 1996.

[38] Wu, C.H., "A General Theory of Three-dimensional Flow in Subsonic and Supersonic Turbomachines of Axial-, Radial-, and Mixed-Flow Types", NACA TN 2604, 1952.

[39] Steward W. L., and Glassman A. J., "Blade Design", Turbine Design and Application, NASA SP-290, 22,1973.

[40] Zweifel O., "The spacing of Turbo-Machine Blading, Especially with Large Angular Deflection", Brown Boveri Rev., Vol.32, No. 12, pp. 436-444, 1945.

[41] Steward W. L., and Glassman A. J., "Blade Design", Turbine Design and Application, NASA SP-290, p.112, 1973.

[42] Abbott I. H., Von Doenhoff A.E., Stivers L.S., "Summary of Airfoil Data", NACA Report No.824, 1945.

[43] O. Oksuz, I.S. Akmandor, "Aerodynamic Optimization of Turbomachinery Cascades Using Euler/Boundary Layer Coupled Genetic Algorithms", AIAA-2001-2577, 2001.

[44] Pritchard L. J., "An Eleven Parameter Axial Turbine Airfoil Geometry Model", ASME 85-GT-219, 1985.

[45] Trigg M. A., Tubby G.R., Sheard A.G, "Automatic Genetic Optimization Approach to Two-dimensional Blade Profile Design For Steam Turbines", Journal of Turbomachinery, pp. 11-18, 1999.

[46] Anders J. M., Haarmeyer J., "A Parametric Blade Design System", VKI LS 1999-02: Turbomachinery Blade Design Systems, 1999.

[47] Yamamoto K., Inoue O., "Applications of Genetic Algorithm to Aerodynamic Shape Optimization", AIAA-95-1650, 1995.

[48] Pierret S., "Designing Turbomachinery Blades By Means Of The Function Approximation Concept Based on Artificial Neural Network, Genetic Algorithm, and the Navier-Stokes Equations", VKI PhD. Thesis, 1999.

[49] Farin G, "Curves and Surfaces for Computer Aided Geometric Design", Academic Press Inc., 1993.

[50] Faux I.D., Pratt M.J., "Computational Geometry for Design and Manufacture", Ellis Horwood Ltd, 1979.

[51] ANSYS® TurboGrid™, http://www.ansys.com/, ANSYS Inc., 2007.

[52] Lefebvre S., Hornus S., Nyret F., "Octree Textures on the GPU", Programming Techniques for High-Performance Graphics and General-Purpose Computation, pp. 595–613, 2005.

[53] Evers B., Kötzing P., "Test Case E/TU-4 4-Stage Low Speed Turbine", Test Cases for Computation of Internal Flows in Aero Engine Components, AGARD-AR-275, pp.365-375, 1990.

[54] ANSYS® CFX®, http://www.ansys.com/, Ansys Inc., 2007.

[55] Rodi, W., "A note on the empirical constant in the Kolmogorov-Prandtl eddy-viscosity expression", ASME Journal of Fluids Engineering, vol. 97, pp. 386-389, 1975.

[56] Launder, B.E. and Spalding, D.B., "The numerical computation of turbulent flows". Comp Meth Appl Mech Eng, vol.3, pp. 269-289, 1974.

[57] M. White, F.M.. "Viscous Fluid Flow". McGraw-Hill, 1979.

[58] Boletis E., Sieverding C.H., van Hove W., "Test Case E/TU-1 Low Speed Annular Turbine Blade Row", Test Cases for Computation of Internal Flows in Aero Engine Components, AGARD-AR-275, pp.322-330, 1990.

[59] Davis, L., "Genetic Algorithms and Simulated Annealing", Pitman, London, 1987.

[60] Goldberg, D. E., "Genetic Algorithms in Search, Optimization, and Machine Learning", Reading, Addison-Wesley, 1989.

[61] Carroll D.L., "Chemical Laser Modeling with Genetic Algorithms", AIAA Journal, Vol.34, No.2, pp.338-346, 1996.

[62] Deb K., "Multi-Objective Optimization using Evolutionary Algorithms", John Wiley & Sons, Chichester, 2001.

[63] Press W. H., Flannery B. P., Teukolsky S. A., and Vetterling W. T., "Numerical Recipes: The Art of Scientific Computing (FORTRAN Version)", Cambridge Univ. Press, Cambridge, 1989.

[64] Baker J. E., "Adaptive Selection Methods for Genetic Algorithms", Proceedings of an International Conference on Genetic Algorithms and their Application, New Jersey, USA: Lawrence Erlbaum Associates, pp.101-111, 1985.

[65] Baker J. E., "Reducing Bias and Inefficiency in the Selection Algorithm", Proceedings of the Second International Conference on Genetic Algorithms and their Application, Hillsdale, New Jersey, USA: Lawrence Erlbaum Associates, pp.14-21, 1987.

[66] Gorges S. M., "Explicit Parallelism of Genetic Algorithms through Population Structures", Parallel Problem Solving from Nature, Vol. 496, Springer-Verlag, pp.150-159, 1991.

[67] Voigt H. M., Santibanez K. I., and Born J., "Hierarchically Structured Distributed Genetic Algorithm", Parallel Problem Solving from Nature, Amsterdam: Elsevier Science Publishers, pp.145-154, 1992.

[68] Mühlenbein H., and Schlierkamp V. D., "Analysis of Selection, Mutation and Recombination in Genetic Algorithms", Technical

Report 93-24, GMD, 1993.

[69] Blickle T., and Thiele L., "A Comparison of Selection Schemes used in Genetic Algorithms", TIK Report No. 11, 1995.

[70] Goldberg D. E., and Deb K., "A comparison of selecting schemes used in genetic algorithms", Foundation of genetic algorithms 1 (FOGA-1), pp. 69-93,1991.

[71] Spears W. M., "The Role of Mutation and Recombination in Evolutionary Algorithms", Ph. D. Thesis, Fairfax, VA: George Mason University, 1998.

[72] Ong Y.S., Nair P.B., Keane A.Z., "Evolutionary Optimization of Computationally Expensive Problems via Surrogate Modeling," AIAA Journal, Vol. 41, No. 4, pp. 687-696, 2003.

[73] Ratle A.,"Accelerating Evolutionary Algorithms Using Fitness Function Models," Parallel Problem Solving from Nature, Vol. 5, pp. 87-96, 1998.

[74] El-Beltagy M.A, Nair P.B., Keane A.J., "Metamodelling Techniques For Evolutionary Optimization of Computationally Expensive Problems:Promise and Limitations," Proceeding of the Genetic and Evolutionary Computation Conference, pp. 196-203, 1999.

[75] Jin Y., Olhofer M., Sendho B., "A Framework for Evolutionary Optimization with Approximate Fitness Functions," IEEE

Transactions on Evolutionary Computation, Vol 6, No. 5, pp. 481-494, 2002.

[76] Ong Y.S., Nair P.B., Keane A.J, Wong K.W, "Surrogate-Assisted Evolutionary Optimization Frameworks for High-Fidelity Engineering Design Problems", Knowledge Incorporation in Evolutionary Computation, Studies in Fuzziness and Soft Computing Series, Springer Verlag, 2004.

[77] Ratle A, "Kriging as a Surrogate Fitness Landscape in "Evolutionary Optimization", Artificial Intelligence for Engineering Design Analysis and Manufacturing, Vol. 15, No. 1, pp. 37-49, 2004.

[78] El-Beltagy M.A, Nair P.B., Keane A.J., "Metamodelling Techniques For Evolutionary Optimization of Computationally Expensive Problems:Promise and Limitations," Proceeding of the Genetic and Evolutionary Computation Conference, pp. 196-203, 1999.

[79] Jin Y., Olhofer M., Sendho B., "A Framework for Evolutionary Optimization with Approximate Fitness Functions," IEEE Transactions on Evolutionary Computation, Vol 6, No. 5, pp. 481-494, 2002.

[80] Song W. B., "Shape Optimisation Of Turbine Blades", University of Southampton, PhD Thesis, 1998.

[81] Yang S., "On the Design of Diploid Genetic Algorithms for Problem Optimization in Dynamic Environments", IEEE Congress on Evolutionary Computation, Vancouver, 2006.

[82] Branke J., "Evolutionary Optimization in Dynamic Environments", Kluwer Academic Publishers, 2002.

[83 Grefenstette JJ., "Genetic algorithms for changing environments", Parallel Problem Solving from Nature II, pp. 137-144, 1992.

[84] Yang S., "Memory-based immigrants for genetic algorithms in dynamic environments", Proc. of the Genetic and Evolutionary Computation Conference, vol. 2, pp. 1115-1122, 2005.

[85] Cobb H. G., Grefenstette J. J., "Genetic algorithms for tracking changing environments", Proc. of the 5th Int. Conference on Genetic Algorithms, pp. 523–530, 1993.

[86] Branke J. "Memory enhanced evolutionary algorithms for changing optimization problems", Proc. of the Congress on Evolutionary Computation, vol. 3, pp. 1875-1882, 1999.

[87] Louis S. J., Xu Z., "Genetic algorithms for open shop scheduling and re-scheduling", Proc. of the 11th ISCA Int. Conference on Computers and their Applications, pp. 99-102, 1996.

[88] Mori N., Kita H., Nishikawa Y., "Adaptation to changing environments by means of the memory based thermodynamical

genetic algorithm", Proc. of the 7th Int. Conference on Genetic Algorithms, pp. 299-306, 1997.

[89] Simoes A, Costa E., "An immune system-based genetic algorithm to deal with dynamic environments: diversity and memory", Proc. of the 6th Int. Conference on Neural Networks and Genetic Algorithms, pp. 168-174, 2003.

[90] Trojanowski K., Michalewicz Z., "Searching for optima in nonstationary environments", Proc. of the Congress on Evolutionary Computation, pp. 1843-1850, 1999.

[91] Branke J., Kaussler T., Schmidth C., Schmeck H., "A multipopulation approach to dynamic optimization problems", Proc. of the Adaptive Computing in Design and Manufacturing, pp. 299–308, 2000.

[92] Goldberg D.E., Smith R.E., "Nonstationary function optimization using genetic algorithms with dominance and diploidy". Proc. of the 2nd Int. Conference on Genetic Algorithms, pp. 59-68, 1987.

[93] Ng K.P., Wong K.C., "A new diploid scheme and dominance change mechanism for non-stationary function optimization", Proc. of the 6th Int. Conference on Genetic Algorithms, 1995.

[94] Ryan C., "The degree of Loneness", Proc. of the 1994 ECAI Workshop on Genetic Algorithms, 1994.

[95] Zitzler, E., "Evolutionary Algorithms for Multi-objective Optimization: Methods and Applications", PhD. Thesis, EHT No. 13398, Zurich, 1999.

[96] Zitzler, E., Deb, K., Thiele L., "Comparison of Multi-objective Evolutionary Algorithms: Emprical Results", Evolutionary Computation Journal, vol. 8-2, pp. 125-148, 2000.

[97] Osyczka A., Kundu S., "A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm", Structural Optimization, vol. 10(2), pp. 94-99, 1995.

[98] Osyczka A., "Evolutionary Algorithms for Single and Multicriteria Design Optimization", Physica-Verlag, New-York, pp.93-113, 2002.

[99] Osyczka A., Kundu S., "A modified distance method for multicriteria optimization using genetic algorithms", Computers in Industrial Engineering, Special Issue on Genetic Algorithms, vol.30-4 pp.871-882, 1996.

[100] Deb K., Pratap A., Agarwal S., Meyarivan T., "A fast and elitist multi-objective genetic algorithm: NSGA-II", IEEE Transactions on Evolutionary Computation, Vol.6, No.2, pp.182-197, 2002.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name:	Öksüz, Özhan
Nationality:	Turkish (TC)
Date and Place of Birth:	5 August 1977, İstanbul
Marital Status:	Married with a child
Phone:	+90 536 252 22 49
email:	ozhan@ae.metu.edu.tr

EDUCATION

1992 - 1994 **High School** İstanbul, Turkey
Kabataş Erkek Lisesi

1994 - 1999 **Aeronautical Engineering, BS.** Ankara, Turkey
Dept. of Aeronautical Engineering, METU.
Graduated with an Honors Degree: 3.04 / 4.00

1999 - 2001 **Engineering Management, MS.** Ankara, Turkey
Dept. of Industrial Engineering, METU.
Graduated with High-Honors Degree: 3.75 / 4.00
Thesis: Aerospace Industry Analysis and Development of Turkey's Aerospace Technology Strategy.

1999 - 2002 **Aerospace Engineering, MS.** Ankara, Turkey
Dept. of Aerospace Engineering, METU.
Graduated with an Honors Degree. 3.43 / 4.00
Thesis: Aerodynamic Optimization of Turbine Cascades Using an Euler/Boundary Layer Solver Coupled Genetic Algorithm.

2003 - 2004 **Turbomachinery Diploma, DC.** Brussels, Belgium
Dept. of Turbomachinery, VKI.
Graduated with High-Honors Degree
Thesis: Aerothermal Investigation of Axial Turbine Tip Gap Flows.

WORK EXPERIENCE

11/2006 – Present **Technical Consultant** Ankara, Turkey
Tusaş Motor Sanayi A.Ş. (TEI)
Technical Services Dept.

- Technical Consultant position at Turkish Air Force Air Logistics Center. (GE F110-100&129 and J79&J85).

11/2005 – 07/2006 **Lieutenant** Ankara, Turkey
Jet Engine Division
Combatant Aircraft Office, TuAF Logistics.

- Working as a Systems Manager of GE F110-100&129 jet engines.

10/1999 – 07/2005 **Research Assistant** Ankara, Turkey
Middle East Technical University (METU)
Dept. of Aerospace Engineering

06/1999 - 09/1999 **Aerospace Engineer** Brussels, Belgium
von Karman Institute (VKI)
Turbomachinery Dept.

- Experimentally investigated the usability and reliability of the fast response heat transfer gauges.

ACHIEVEMENTS

- *September 1994* Ranked 1st at the Dept. of Aeronautical Engineering, METU. — ÖYS Point
- *June 1999* Ranked 4th at the Dept. of Aeronautical Engineering, METU. — CGPA
- *June 2001* Ranked 1st at the Dept. of Industrial Engineering, METU. — CGPA
- *2002 – 2006* PhD. scholarship awarded by TUBİTAK-BAYG. — Academic Success
- *June 2004* Ranked 1st at the Dept. of Turbomachinery, von Karman Institute. — CGPA
- *June 2004* Best Technical Presentation Award, von Karman Institute. — Thesis Presentation
- *October 2005* Ranked 1st at the Turkish Army Lieutenant Training Program. — Overall Success
- *May 2006* Honored by Turkish Air Force due to F110 Engine Systems Management success. — Systems Availability
- *April 2007* Awarded by Turkish Air Force Logistics Commander — Technical Innovation

PROJECTS

6/2003 – 6/2005 Helicopter Engine T701C Turbine Blade Redesign and Optimization — Turkish Land Forces 5.ABM

9/2003 – 6/2004 SNECMA Gas Turbine Engine Turbine Blade Tip Damage Aerothermal Experimental & Numerical Investigation — Von Karman Institute

5/2003 - 11/2003 Realtime Data Acquisition Setup of an Experimental Turbocharger Based Jet Engine — METU BAP 2003-03-13-01

6/2002 - 9/2002 Setup of an Experimental Small Turbojet Engine — Tubitak MİSAG 102M063

6/2001 - 6/2002 Genetic Optimization of Gas Turbine Blades — METU AFP 2001-03-13-01

8/2001 - 10/2001 On-site Performance Assessment of a LM2500+ Power Plant — Zorlu Energy O&M

FOREIGN LANGUAGES

Advanced English, Beginner German

RESEARCH INTERESTS

- *Turbomachinery Systems and Components:* Turbines, Compressors
- *Optimization techniques and applications*: GA, MOGA, GAXL.
- *Heat Transfer*: Novel fast response measurement techniques, aerothermal transient experimental investigations.
- *Artificial intelligence*: Petri-nets, reinforcement learning, neural-nets, fuzzy-logic, artificial neural networks.
- *Computational fluid dynamics*: Theory, coding, application, testing.
- *Component performance design*: Compressors, pumps, vanes and turbines.
- *Advanced engineering cycles simulation*: Analysis, construction, simulation and optimization.

- *Energy*: Cogeneration, wind energy, combined cycles.
- *Engineering management*: Technology management, MIS, R&D, organizational behavior, project management, total quality management, operations research, engineering economy.
- *Power plants*: performance diagnosis & health monitoring, efficiency, life cycle cost assessment, distributed generation.

SELECTED PUBLICATIONS

Arts, T., Ginibre, P., Oksuz, O., Iliopoulou, V., Key, N., "Comparison Of Turbine Tip Leakage Aero Thermal Flows For Flat Tip And Squealer Tip Geometries At High-Speed Conditions - Experimental And Numerical Investigation", 6th European Conference On Turbomachinery: Fluid Dynamics And Thermodynamics, Lille, France, March 7-11, 2005.

Öksüz Ö., "Aerothermal Investigation of Turbine Tip Gap Flows", VKI-DC-2004-06, von Karman Institute, June 2004, Belgium.

Öksüz Ö., Akmandor İ. S., Kavsaoğlu M. S., "Aerodynamic Optimization of Turbomachinery Cascades Using Euler/Boundary Layer Coupled Genetic Algorithms", Journal of Propulsion and Power, Vol.18, No.3, May 2002, pp.652-657.

Akmandor İ. S., Öksüz Ö., Gökaltun M. S., Bilgin M., "Genetic Optimization of Steam Injected Gas Turbine Power Plants", GT-2002-30416,ASME TURBO EXPO 2002, 3-6 June 2002.

Akmandor İ. S., Öksüz Ö., "Optimizing the Boost Performance of an Aircraft Gas Turbine with Water Injection", 4th Aerospace Symposium, Kayseri-Turkey, 13-15 May, 2002.

Öksüz Ö., Akmandor İ. S., "Turbine Cascade Optimization Using an Euler Coupled Genetic Algorithm", ISABE-2001-1234, ISABE International Symposium on Air Breathing Engines, 2-7 September 2001.

Öksüz Ö., Akmandor İ. S., "Aerodynamic Optimization of Turbomachinery Cascades Using Euler/Boundary Layer Coupled Genetic Algorithms", AIAA-2001-2577, AIAA Computational Fluid Dynamics Conference, 11-14 June 2001.

Öksüz Ö., “Technology Strategies of Developing Countries and Innovation Networks”, TMMOB UUHUMK National Aerospace Engineering Congress, 12-13 May 2001.

Öksüz Ö., Arts T., "A Fast Response Total Temperature Probe for Transient Flow Measurements," von KARMAN Report 1999-30, September 1999.

Çelebioğlu O., Akmandor I. S., Öksüz Ö., "Experimental Setup of a Low Speed Axial Compressor Stage," 2nd Aerospace Symposium, Kayseri-Turkey, 11-15 May, 1998., pp. 227-231.