AERODYNAMIC OPTIMIZATION OF A TRANSONIC AERO-ENGINE FAN MODULE

A Thesis Submitted to
The Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in Mechanical Engineering

by
Orçun KOR

July 2016
İZMİR

We approve the thesis of **Orçun KOR**

Examining committee members:

_____
Assist. Prof. Dr. Ünver ÖZKOL
Department of Mechanical Engineering, İzmir Institute of Technology

_____
Prof. Dr. Gökmen TAYFUR
Department of Civil Engineering, İzmir Institute of Technology

_____
Assoc. Prof. Dr. Erdal ÇETKİN
Department of Mechanical Engineering, İzmir Institute of Technology

_____
Assist. Prof. Dr. Ziya Haktan KARADENİZ
Department of Mechanical Engineering, İzmir Katip Çelebi University

_____
Assist. Prof. Dr. Levent AYDIN
Department of Mechanical Engineering, İzmir Katip Çelebi University

22 July 2016

_____
Assist. Prof. Dr. Ünver ÖZKOL
Supervisor, Department of Mechanical Engineering, İzmir Institute of Technology

_____
Prof. Dr. Metin TANOĞLU
Head of the Department of Mechanical Engineering

_____
Prof. Dr. Bilge KARAÇALI
Dean of the Graduate School of Engineering and Science

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor Assist. Prof. Dr. Ünver Özkol for his valuable advises and guidance throughout my graduate studies. I am deeply grateful to my thesis progress and defense jury members Prof. Dr. Gökmen Tayfur, Assoc. Prof. Dr. Erdal Çetkin, Assist. Prof. Dr. Levent Aydın, Assist. Prof. Dr. Ziya Haktan Karadeniz for their guidance, suggestions, and remarkable contributions to my thesis.

I have to thank my colleague and also dear friend Dr. Sercan Acarer, for allowing me using the solver he developed. His guidance and tutoring was indispensable in understanding not only the structure, but also the physics of his code.

I also acknowledge TUSAŞ Engine Industries (TEI) for administratively supporting the studies that are in scope of this thesis.

My dear family members, Nurcan Kayar, Celalettin Kayar, Kazım Kor and Atacan Kor, with their immense patience, were always supporting me during my university studentship, longing from undergraduate until the end of my doctorate studies. They all deserve the heartiest gratitude in the acknowledgments of this thesis.

And my lovely Tuğçe… She was always, without any hesitation, there, when I needed her shoulder next to mine. I thank her for being in my life…

ABSTRACT

AERODYNAMIC OPTIMIZATION OF A TRANSONIC AERO ENGINE FAN MODULE

Aerodynamic design of an aero-engine fan blade is a multi-step process with multi-variables. The general purpose in aerodynamic design is to obtain proper blade angles and flowpath geometry providing the necessary pressure ratio with maximum efficiency, while respecting the structural and aerodynamic constraints. The throughflow design in aerodynamic design procedure is a key step where one can obtain a basic aero-design which generally fixes 80% to 90% of the final fan geometry, by adjusting parameters like blade exit angle distribution, solidity, hub and shroud contour, meridional chord length, etc. Throughout this procedure, the aim of the designer is to obtain an optimum (i.e. light, reliable and robust) system with highest efficiency.

Among optimization methods, zero order methods are reported to fit best for turbomachinery problems, due to their good performance in discrete and non-differentiable problems and their ability to find the global optimum. Genetic algorithm is the most widely used optimization method in turbomachinery optimization. Methods inspired by swarm intelligence are reported as promising global optimizers, whereas, to the author's knowledge, there are no reported studies that employs such algorithms in turbomachinery throughflow optimization. These methods can find the neighborhood that provides the globally optimum design, rather than exactly finding the global design. This drawback is overcome by hybridizing genetic/swarm inspired algorithms by first order (gradient based) methods.

Within this aspect, the present study focuses on developing genetic and swarm inspired algorithms hybridized with gradient based algorithms to find the optimum throughflow design of a transonic aero-engine fan module.

ÖZET

TRANSONİK REJİMDE ÇALIŞAN UÇAK MOTORU FAN MODÜLÜNÜN AERODİNAMİK OPTİMİZASYONU

Uçak motorlarında fan modülünün aerodinamik tasarımı çok aşamalı ve birden fazla değişkeni kapsayan bir süreçten oluşmaktadır. Fan aerodinamik tasarımında amaç, istenen basınç oranını sağlayacak uygun kanatçık açılarının ve akış yolu geometrisinin, maksimum verim ile yapısal ve aerodinamik kısıtlar göz önünde bulundurarak elde edilmesidir. Aerodinamik tasarımın anahtar aşaması olan aksi-simetrik tasarımdaysa kanatçık çıkış açısı, veter-kanatçık açıklığı oranı, akış yolu geometrisi, meridyonel veter uzunluğu vb. parametreler değiştirilerek en iyi tasarım elde edilir ve fan pali ve modülü geometrisi %80~90 oranında sabitlenir. Bu en iyileştirme süreci esnasında amaç en yüksek verimde çalışan en optimum (en ucuz, en güvenilir ve en dayanıklı) sistemin elde edilmesidir.

Optimizasyon metotları arasında sıfırıncı dereceden metotlar, türevlenemez ve süreksiz problemlere uygunlukları ve global optimum noktasını tespit edebilirlikleri dolayısıyla türbomakina optimizasyonuna en uygun olanlarıdır. Bu kapsama giren genetik algoritmalar türbomakina optimizasyonunda yaygın olarak kullanılmaktadır. Sürü davranışlarından esinlenen optimizasyon metotlarının da global optimum noktasını tespit etme konusunda başarılı olduğu bildirilse de, yazarın bildiği kadarıyla bu metotların türbomakina optimizasyonu alanında kullanıldıkları bildirilmemiştir. Öte yandan, genetik algoritmaların ve sürü davranışlarından esinlenen metotların global optimum noktasını tam olarak olmasa da, optimuma en yakın noktaları tespit ettikleri bilinmektedir. Global optimum noktasının kesin olarak tespiti, genetik algoritma ve sürü davranışlarından esinlenen metotların, birinci dereceden (gradyan temelli) metotlarla melezleştirilmesiyle elde edilen yöntemlerin kullanılmasıyla mümkün olmaktadır.

Bu çalışma, genetik algoritma ve sürü davranışlarından esinlenen metotların gradyan temelli metotlarla melezleştirilmesiyle elde edilecek yeni bir metot yardımıyla, transonik rejimde çalışan uçak motoru fan palinin ve modülünün aerodinamik aksi-simetrik optimum tasarımının elde edilmesini amaçlamaktadır.

"One's ideas must be as broad as Nature if they are to interpret Nature…"

Sherlock Holmes

TABLE OF CONTENTS

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

Table **Page**

LIST OF SYMBOLS

a(j)	output vector
b	Bias factor
B	Bernstein polynomial
c	chord
c_p	Cognitive attraction constant (in PSO)
c_g	Social attraction constant (in PSO)
C	Child (in GA)
cr	Reduction rate (in SA)
E	Energy state (in SA)
f(x)	Objective function
g(x)	Inequality constraint
h	Blade height
h	Entalphy (kJ/kg)
h(x)	Equality constraint
H(x)	Hessian matrix
k	Boltzmann prob. const.
l	Binary string length (GA)
N	Number of individuals in a population
P	Parent (in GA)
P(E)	Boltzmann prob. dist.
p(j)	Input vector
R_c	Radius of curvature (m)
r	Penalty multiplier
r_p	Cognitive attraction random weight factor (in PSO)
r_g	Social attraction random weight factor (in PSO)
S	Search direction
s	Entropy (kJ/kg K)
s	Number of selected individuals (GA)
T	Temperature (in SA)
V	Absolute velocity (m/s)
w	Weight factor

W	Relative velocity (m/s)
x	Design variable

Greek letters

α	Step length
δ	Kronecker delta
Δ	Rise/drop of given quantity
φ	Pseudo-objective function
η	Efficiency
μ	Spanwise average
∇	Gradient operator
τ	Pheromone
θ	Tangential component

Abbreviations

ANN	Artificial Neural Network
ACO	Ant Colony Optimization
BPR	By-Pass Ratio
DE	Differential Evolution
DF	Diffusion Factor
EA	Evolutionary Algorithm
GA	Genetic Algorithm
LB	Lower Bound
NS	Navier Stokes equations
SA	Simulated Annealing
FT	Transfer Function
MO	Multi-Objective
MOO	Multi-Objective Optimization
OF	Objective Function
PR	Total – to – total pressure ratio
PSO	Particle Swarm Optimization
SOO	Single-Objective Optimization
std	Standard deviation

TF Twist Factor

UB Upper Bound

Supercripts

low lower limit

up upper limit

CHAPTER 1

INTRODUCTION

The aerodynamic design of aero-engine compressor and fan blades is a multi-step process with multi-variables. The general purpose in aerodynamic design is to obtain a blade and flowpath geometry providing the necessary pressure ratio with minimum losses, therefore maximum efficiency, while respecting the structural constraints. In order to obtain the desired geometry satisfying the necessary boundary conditions, one has to adjust the geometric parameters properly to obtain a throughflow design which generally fixes 80% to 90% of the final fan design (Joly, et al., 2012), where these parameters can be listed as solidity, hub and shroud contour, aspect ratio (***h/c***), twist factor (***TF***) etc. Once throughflow design is fixed, the designer may further improve the geometry by adjusting blade stacking axis, refining 3D blade geometries (such as lean, sweep and bow) and 3D endwall profiles. In addition to the aerodynamic considerations, the designer will also be responsible for satisfying maximum stress, fatigue loads, weight and aero-acoustic level demands etc. Obviously, throughout this procedure, the aim of the designer is to obtain a light, reliable and robust system with high efficiency while reducing the costs and the time spent for the design process. In other words, one targets to obtain an ***optimum*** machine that provides the best results under given circumstances as this effort can be stated as the typical definition of optimization (Rao, 1996).

This thesis first introduces the basic definitions used in optimization literature as they are frequently used within the text. Afterwards, optimization methods that are used in the literature is given. Next, a brief summary about fan/compressor throughflow design procedure is discussed and the parameters that are optimized are stated. Eventually the tools that are utilized in transonic aero-engine fan optimization are summarized, together with the optimum geometries that are obtained using these tools.

CHAPTER 2

LITERATURE REVIEW

2.1. Literature Review on Optimization

Aim of optimization can be suggested as finding the conditions that give the maximum or minimum value of a function, depending on the constraints imposed by the designer. In Figure 2.1, a typical function, $f(x)$, can be seen, where optimization procedure can also be defined as finding minimum of $f(x)$and/or maximum of $-f(x)$. It should be noted that in real life problems, ***f*** is not only a function of ***x*** but more than one parameter, which makes the optimization a complex procedure. It is obvious that manual optimization has several disadvantages, since meeting multiple requirements simultaneously is often time consuming and vulnerable to human-sourced errors, as this situation necessitates the use of advanced automatic optimization techniques, which gives rise to the use of computational methods, complex algorithms and powerful computers.

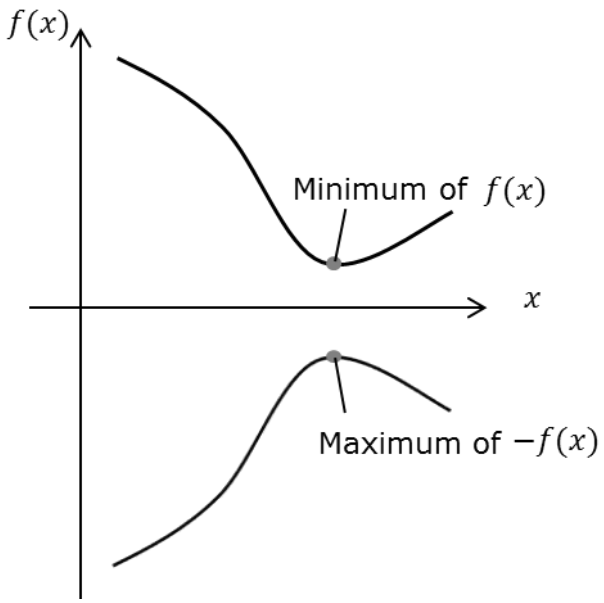


Figure 2.1. Typical definition of optimization

2.1.1. Basic Definitions

Prior to proceeding to the details of optimization, some important definitions that are widely used in optimization literature have to be introduced. Following definitions are extracted from the literature (Verstraete, 2012). (Equations (2.1) ~ (2.5) should be taken into account while reading this chapter)

Any optimization problem can be defined with the formulation given in Equations (2.1) ~ (2.5) (Deb, 2010).

Minimize	$f_i(\vec{x}) = 0$		(2.1)
Subject to	$g_j(\vec{x}) \leq 0$	$j = 1 \dots m$	(2.2)
	$h_k(\vec{x}) = 0$	$k = 1 \dots n$	(2.3)
	$x_p^{low}(\vec{x}) \leq x_p \leq x_p^{up}(\vec{x})$	$p = 1 \dots q$	(2.4)
Where	$\vec{x} = \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_q \end{Bmatrix}$		(2.5)

Objective function is the function that is tried to be minimized. For turbomachinery blading problems, this function can be assigned as efficiency, pressure ratio, mass flow rate etc. Objective function can be formulized as given in Equation (2.1). In case the quantity is aimed to be maximized, one may find the minimum value of the function and take the inverse of it as the optimum value (Verstraete, 2008). In some algorithms (such as adjoint methods) the term ***cost function*** is also used for the same purpose.

Design vector contains the design parameters that can be modified throughout the optimization process. For turbomachinery blading problems, this vector can be assigned as geometric position of the flowpath, radius, fillet, thickness distribution, stacking line etc. Design vector can be formulized as given in Equation (2.5).

Optimization problems can be constrained. This means that while minimizing/maximizing objective function, some other functions may be asked to be kept in certain limits. Otherwise, the optimization problem is said to be unconstrained**.**

Inequality constraints represent the quantities that should be satisfied in function evaluation. Minimum strength, minimum stall margin or maximum weight are some examples of this category in turbomachinery problems, see Equation (2.2).

Equality constraints represent functional equalities that should not be violated throughout the optimization process, see Equation (2.3). Equality constraints are usually more difficult to handle and their usage should be avoided as much as possible (Deb, 2010). However, such constraints are usually explicit and good choice of design variables and/or inequality constraints eliminates them. For example, one can employ two inequality constraints in order not to deal with an equality constraint, like using $4.5 \leq h_k(\vec{x}) \leq 5.5$, instead of using $h_k(\vec{x}) = 5$ (Deb, 2010).

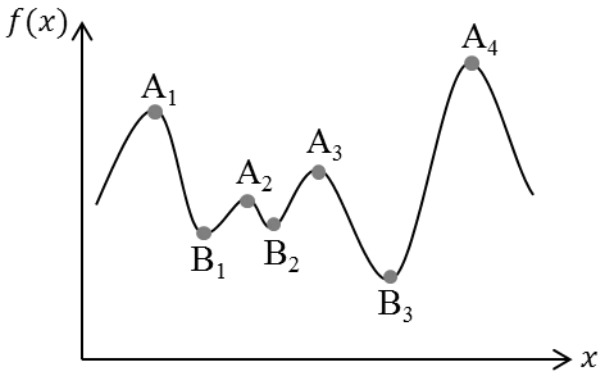
Most optimization techniques are dedicated to find the new design by employing the knowledge of previous data, as formulated in Equation (2.6):

$$\vec{x}_{i+1} = \vec{x}_i + \alpha_i \vec{S}_i \tag{2.6}$$

The parameter $\boldsymbol{\alpha}$ is a scalar quantity, specifying a relative measure of distance from the point $\vec{x}_i$ to $\vec{x}_{i+1}$, which may also be interpreted as step length (Deb, 2010), and the vector $\vec{S}_i$ deals for the search direction. It is worth noting that main difference between numerous optimization algorithms stands from the way that how $\boldsymbol{\vec{S}_i}$ and $\boldsymbol{\alpha_i}$ are determined (Verstraete, 2008).

Local minimum and maximum: $f_i(\vec{x})$ is said to have a local minimum at $\vec{x} = \vec{x}^*$ if $f_i(\vec{x}^*) \leq f_i(\vec{x} + \alpha)$ and a local maximum if $f_i(\vec{x}^*) \geq f_i(\vec{x} + \alpha)$ for all values of $g_j(\vec{x}) \leq 0$ (Equation (2.2)) and $h_k(\vec{x})$ sufficiently close to zero, Figure 2.2 (Equation (2.3)) (Rao, 1996) .

Global minimum and maximum: $f_i(\vec{x})$ is said to have a global minimum at $\vec{x} = \vec{x}^*$ if $f_i(\vec{x}^*) \leq f_i(\vec{x})$ and a global maximum if $f_i(\vec{x}^*) \geq f_i(\vec{x})$ for all $\vec{x}$ values in the domain over which $f_i(\vec{x})$ is defined (Rao, 1996) (See Figure 2.2).



A₁, A₂, A₃ → Local maximum ; ***B₁, B₂*** → Local minimum ***A₄*** → Global maximum ; ***B₃*** → Global minimum

Figure 2.2. Local and global extreme points

2.1.2. Single and Multi Objective Optimization

Single objective optimization can be formulized as given in Equation (2.7), providing that $i = 1$.

Multi objective optimization can be formulized as given in Equation (2.7), providing that $i > 1$. The aim in MOO is to obtain the optimum for the cases where number of objective functions is more than one.

$$f_i(\vec{x}) = 0 \quad ; \quad i = \begin{cases} 1, & SOO \\ > 1, & MOO \end{cases} \tag{2.7}$$

There exists several algorithms which directly solve the MOO problems, however such methods are usually too expensive in terms of computation time; therefore, the classical approach is like either:

1. Choosing the most important objective as the objective while including other objectives as constraints (Deb, 2010).

2. Converting the problem to SOO by giving weights (w_j) to each objective. In this case, the function to be optimized is now called as pseudo-objective function, $F(\vec{x})$, see Equation (2.8). Posing a certain weight value to the objective functions building pseudo-objective function defines their relative importance with respect to other objective functions.

$$F(\vec{x}) = \sum_{j=1}^{m} w_j \ f_j(\vec{x}) \quad , \quad \sum_{j=1}^{m} w_j = 1 \tag{2.8}$$

In MOO problems, the designer may be in a situation, where he/she has to handle objectives those of which are conflicting with each other and has to make compromise in order to obtain an optimum solution (i.e. a solution cannot be improved with respect to any other objective without worsening at least one other objective). In the absence of any further information, one of these optimum solutions cannot be said to be better than the other (Deb, et al., 2002). This situation is illustrated in the car optimization problem given in Figure 2.3.It can be seen that one can increase the comfort of a car, in expense of increasing cost. One will also see from Figure 2.3 that there exists more than one solution lying on the line called Pareto Front. Points B, C, D are optimum solutions among infinite number of other alternative optimum and non-dominated solutions lying on Pareto Front; however, point A states a dominated (therefore, non-preferable) solution with respect to the others, which means that point A does not represent an optimum solution.

Pareto optimal sets can be of varied dimensions (Konak, et al., 2006). It can be deduced from Figure 2.3 that the designer can obtain infinite number of designs by giving different weight values, i.e. posing different relative importance levels, for each objective function. The presence of constraints is the main limiting factor for this particular case as well as other optimization tasks. The designs violating constraints lie in the gray region of the figure and therefore named as unfeasible. As mentioned before, designs lying on the Pareto Front (B, C, D) are said to be the non-dominated optimum designs and the designer can select each of them as his/her final design based on his/her decision of which objective is more important. For example, it will be more reasonable to select design D as the optimum, if the designer is asked to design a car with a highest comfort

level while the designs with high costs are acceptable. Note that the case is the opposite for design B and, finally, for design C, it can be suggested that comfort and cost has equal importance. Other designs which are in the feasible region but are not lying on the Pareto Front are said to be non-optimum or dominated optimum designs. Deeper knowledge about pareto front techniques can be found in the literature (Konak, et al., 2006) , (Deb, et al., 2002) , (Deb, 2008), (Verstraete, 2012).

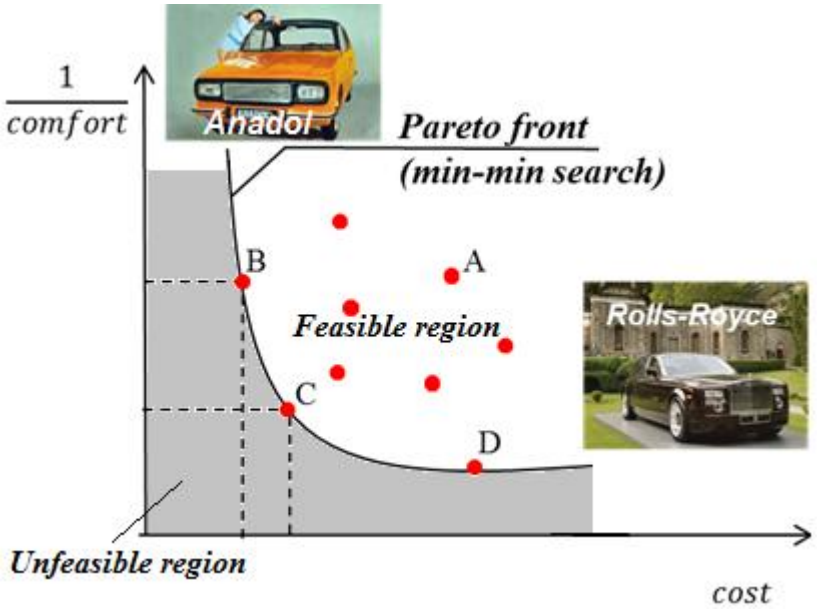


Figure 2.3. A typical MOO problem with conflicting objectives

Classical optimization methods work with a single weight factor combination approach, where the weight factors, w_1,w_2, …, w_n are changed at successive evaluations, such that one can obtain more than one Pareto-optimal solution throughout the optimization process (Hajela & Lin, 1992), (Deb, 1999), (Konak, et al., 2006), the sub-indice, *n*, being the number of objective functions. This method is noted to be as the most straightforward in MOO evaluation; however too many runs are needed to estimate the Pareto Front (Haupt & Haupt, 2004). In case the designer has the information about the weight factors, he/she can directly evaluate Equation (2.8) and obtain the optimum solution with only one objective function evaluation. However, weight information provided to the designer should be accurate and reliable since small perturbations on weight values can lead to quite different solutions (Konak, et al., 2006).

2.1.3. Optimization Methods

Many optimization methods exist in order to solve the optimization problem stated in Equations (2.1) ~ (2.5). These methods can be classified like

- Zero order methods
- First order (gradient based) methods

In zero order (derivative-free or non-gradient or direct) methods, the objective function itself is used in the search for minimum. However in gradient based methods, first and second derivatives of the objective function are used, respectively. An overview of these methods will be given in the next sections.

For any type of optimization method, the computation time exponentially increases with increasing number of design parameters, which is so called curse of dimensionality (Bellman, 1961). The reason behind this is clear: since the number of design variables that is needed to be perturbed increases, the algorithms need more objective function evaluations in order to finalize the optimization process. Based on the experience reported in the literature, the limit for the maximum design parameter number is reported as 40, where higher number of design variables will result in very high computational cost (Verstraete, 2012).

2.1.3.1. Zero Order Methods

Zero-order methods require only function evaluations by making random or systematic sweep of the design space, using evolutionary algorithms or by means of methods inspired by swarm intelligence (Van den Braembussche, 2008). The objective function does not need to be continuous and differentiable; therefore, zero order methods are suitable for turbomachinery applications where the objective functions are totally discrete and discontinuous.

2.1.3.1.1. Random Search Methods

These methods are quite simple and no complex algorithms are used. The design space is scanned randomly and no algorithm for better candidate vector selection is used. Therefore, in order to obtain the optimum, one has to perform an enormous number of objective function evaluations, whereas the determination of the global optimum is still not guaranteed.

Random Search: This method is based on the use of random numbers in finding the minimum point (Rao, 1996). A large number of candidate vectors, $\vec{x}$, are selected and the objective functions, $f_i(\vec{x})$, of each are evaluated. The candidate vector providing the minimum or maximum objective function will be the optimum. No information from previous iterations is used in this method.

Random Walk: The basics of this method are same with Random Search method, but information from previous iterations are used by means of Equation (2.6), where present design is perturbed by a random perturbation, $\boldsymbol{\alpha_i}$.

2.1.3.1.2. Simulated Annealing (SA)

Simulated annealing method mimics the physics of thermal annealing of critically heated solids. In the annealing of metals, the temperature is first raised, and then decreased gradually to a very low value ($\boldsymbol{T_{min}}$), while ensuring that one spends sufficient time at each temperature level. This is because at high temperature levels, metal atoms can move freely with respect to each other, whereas this freedom is attenuated as the temperature is decreased, eventually leading to an ordered crystallic state with minimum possible energy. Once the temperature decrease is so steep (i.e. cooling time is so fast), the crystalline state won't be achieved, instead, a poly-crystalline material (a form that is not wanted) will be obtained (Deb, 2010), (Bandyopadhyay, et al., 2008).

The cooling process described above is simulated to achieve the minimum function value in a minimization problem (Rao, 1996). A temperature-like parameter controlled by *Boltzmann's probability distribution* is used in this method, which states that the energy of a system E in thermal equilibrium at temperature T is distributed probabilistically according to Equation (2.9).

$$P(E) = e^{-E/kT} \tag{2.9}$$

Where k is the Boltzmann constant. Equation (2.9) states that at high temperature, the probability of achieving any energy state (E1, E2, E3) is high whereas the probability decreases with decreasing temperature, as seen in the representative case plotted in

Figure 2.4. Note that annealing process is first simulated for optimization problems by Metropolis et al. (Metropolis, et al., 1953). In this approach, one may model his/her search process with reference to Boltzmann's probability distribution and manipulate the convergence of the algorithm by means of temperature, T. The probability of accepting the next design point depends on the energy state difference (in other words, objective function difference – Equation (2.11)) at two different design points. In case $\Delta E \leq 0$, $P[E_{k+1}]$ will be automatically 1, therefore $\vec{x}_{k+1}$ will always be accepted. In general, the new design point will be accepted in case of $P[E_{k+1}] > P[E_k]$

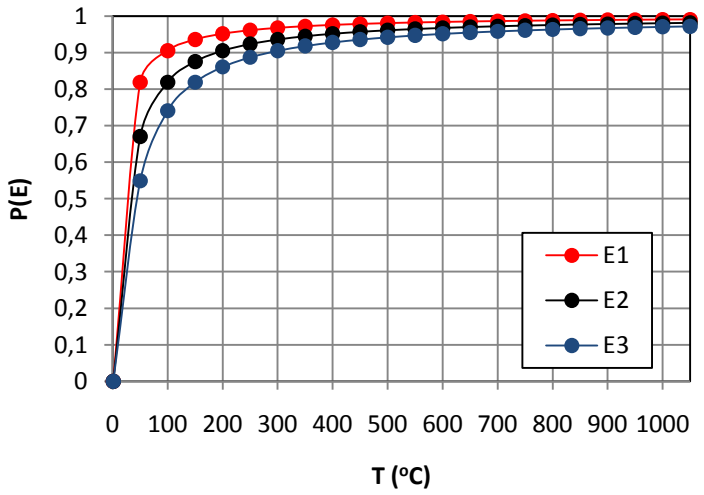


Figure 2.4. Boltzmann probability distribution – a representative case

$$E_k = f_k \tag{2.10}$$

$$\Delta E = \Delta f = f_{k+1} - f_k = f(\vec{x}_{k+1}) - f(\vec{x}_k) \tag{2.11}$$

$$P[E_{k+1}] = min\{1, e^{-\Delta E/kT}\} \tag{2.12}$$

Within this aspect, the algorithm summarizing ***Simulated Annealing*** is given in Appendix A, which can be summarized like as follows:

Step 1: Start with an initial design vector, $\vec{x}_1$ (for example, radius values of the flow path in successive axial locations of the bypass region of an axial fan).

Evaluate objective function, f_1 (for example, entropy generation between two axial stations of the axial fan), corresponding to randomly picked design vectors; take the mean of these objective function values as temperature $\boldsymbol{T}$. Take the reduction factor, $\boldsymbol{c}$, as 0.5; define maximum number of iterations, $\boldsymbol{n}$.

Step 2: Evaluate objective function, f_1, corresponding to $\vec{x}_1$, set iteration number k=1.

Step 3: Generate new design point $\vec{x}_{k+1}$, compute f_{k+1} and $\Delta f = f_{k+1} - f_k$.

Step 4: Accept or reject $\vec{x}_{k+1}$ using Equations (2.10), (2.11), (2.12).

Step 5: Update iteration number as $k = k + 1$. If $k \geq n$ proceed to **Step 8.** Otherwise, proceed to **Step 6**.

Step 6: Reduce the temperature level to $(cr \times T)$.

Step 7: Check for convergence. Stop if convergence is satisfied, Else, go to **Step 3**.

Further information and detailed examples about simulated annealing can be found in the literature (Rao, 1996), (Deb, 2008).

2.1.3.1.3. Evolutionary Algorithms

Evolutionary algorithms (EAs) are developed to simulate processes observed in natural evolution (Giannakoglou, 1999). These algorithms mimic Darwinian evolution where populations of individuals evolve over a search space and adapt to the environment.

2.1.3.1.3.1. Genetic Algorithm (GA)

Genetic algorithms have been extensively used as search and optimization tools over the last decades (Deb, 2008). Selection, crossover, mutation, and elitism are the operators that are used in genetic algorithms, with the goal of finding the fittest solution

to a problem (i.e. objective function with lowest/highest value depending on the case) (Mengitsu & Ghaly, 2008), (Joly, et al., 2010), (Tayfur, 2012).The terminology used in GA literature is listed in Table 2.1 (Pierret, 1999), as will be used in this text frequently.

Table 2.1. GA terminology

Gene	One design variable
Chromosome	Set of design variables
Individual	Design vector
Population	Set of design vectors
Fitness $\left(=\frac{1}{f(x)}\right)$	Performance measure of an individual

A brief explanation of the working mechanism of genetic algorithms is as follows:

Step 1: An initial population is generated, where each individual is represented by certain design variables. Each variable is converted from decimal to binary forms (Figure 2.5) of length ***l***, where ***l*** is the number of digits of a design variable. An individual with ***n*** design variables will be represented by an ($n \times l$) long chromosome. Note that number of bits in the binary string can be increased in order to provide more accurate representation of the design variable and design vector. The individuals should respect the lower and upper variable bounds, see Equation (2.4).

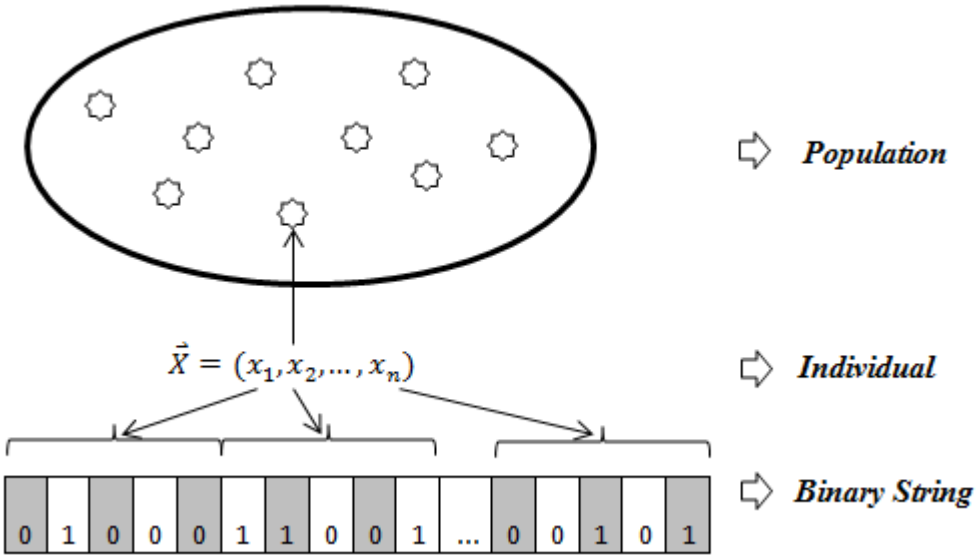


Figure 2.5. Genetic algorithm – basic definitions

Step 2: Using the numerical values of design variables provided in population initialization, the objective function and constraint values are evaluated. Fitness values of the solutions are obtained with reference to the evaluation of the objective functions.

Step 3: Selection is performed on the population. Generally speaking, two different selection methods are used in the literature:

Tournament method*:** In this method, ***s many individuals are selected from the population, where the best two of selected individuals are picked as parents. The variable ***s*** can be chosen between $1 < s \leq N$, where ***N*** is the total individual number. As the value of ***s*** is increased, the selection becomes more elitist and the method will behave like ***roulette wheel method***, which is described next. Large values of ***s*** will also cause the method to stuck into a local optimum. It is reported that ***s***=2 gives the best result in tournament selection method (Van den Braembussche, 2008).

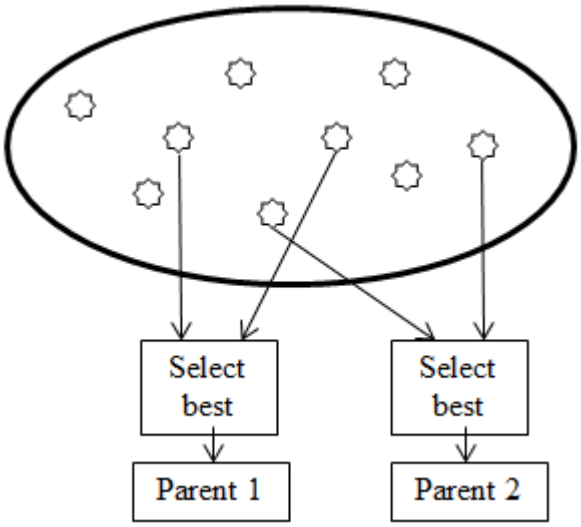


Figure 2.6. Tournament selection method

Roulette wheel method*:** Selection operator randomly shifts a defined number of individuals to the next generation, keeping them unchanged. The probability for an individual to be selected is proportional to its fitness, as any individual with higher fitness value has more chance to be selected for the new population. In other words, this scheme favors the best individuals as parents. It is ***elitist and has larger chance to get stuck in a local optimum. In summary, one will obtain fitter individuals by using this method; however the diversity will be poorer with respect to tournament method (Van den Braembussche, 2008), (Verstraete, 2012), (Manzan, et al., 2008).

Step 4: The individuals obtained in the selection process are paired with each other with respect to their fitness and new children are generated by a reproduction mechanism which is so called ***crossover*** (Figure 2.7, Figure 2.8). Parents represented by binary strings are cross-overed in a random manner. The location of the crossing site shown by the dashed red line in Figure 2.7 is chosen randomly, meaning that its position may change from one optimization run to another.

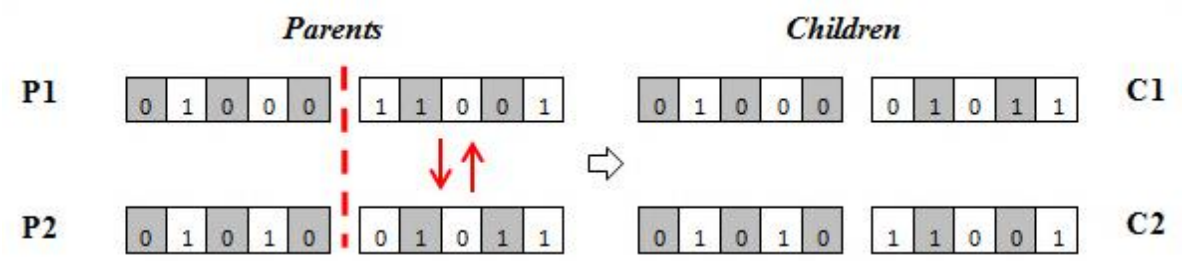


Figure 2.7. An example for crossover operation

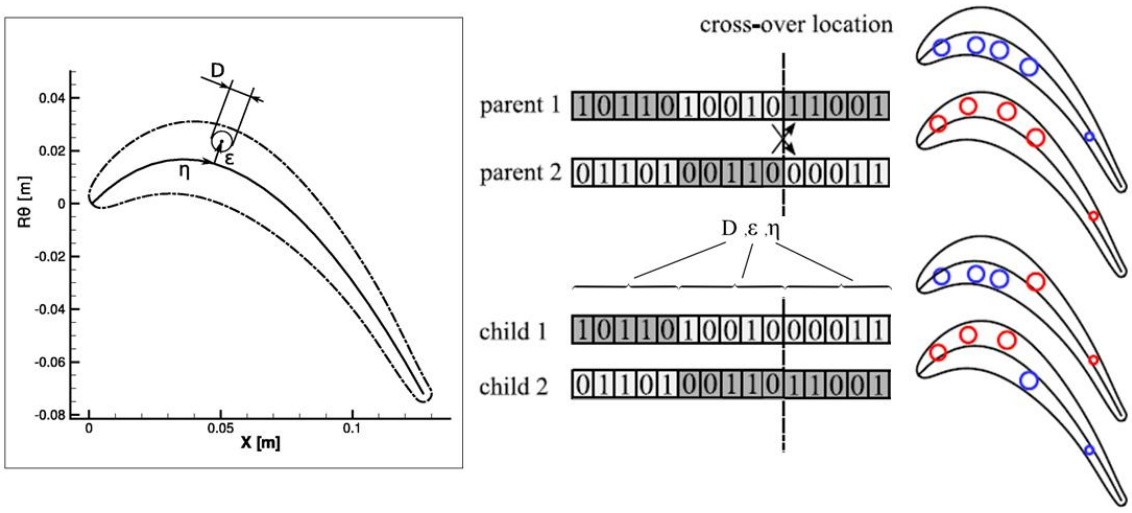


Figure 2.8. Crossover example for turbine cooling holes
(Source: Verstraete, 2012)

Step 5: ***Mutation*** is applied to the new individuals to make the new children fitter and also to increase diversity by changing any offspring string "1" to "0" or vice versa (Figure 2.9). The operator mimics the behavior of mutation in the evolution process. Likewise in the evolution of the species, mutation introduces diverse members to the search space and also it prevents the algorithm to get stuck into a local minimum. Optimum mutation rate is reported to be as 1% in the literature (Deb, 2008); in other words, 1 gene over 100 is switched from "1" to "0" or vice versa. Higher values of probability leads to a convergence problem, since the probability of passing good genes of the parents to the children will be diminished by this mean.

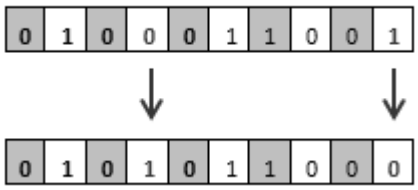


Figure 2.9. An example for mutation operation

Step 6: New individuals obtained in previous steps are subjected to function evaluation. Objective function (in other words, fitness) and constraint function values are provided in this phase. Most computational time consuming phase of the algorithm is this step, since the number of computations conducted is equal to the number of family members. However, genetic algorithm is a discrete method; therefore one can get over the computational time problem by using parallel processors.

Step 7 Now, the optimizer has ***2N*** many individuals, where ***N*** of them belongs to parents (old generation) and ***N*** of them to children (new generation). In GA, a population consists of a fixed number of individuals (Verstraete, 2012), hence, in order to guarantee a constant population size, each pair of parents can give birth to two children. With this philosophy, after each crossover operation, the children either replace their parents, if they are fitter, or eliminated if they have poor fitness with reference to their parents, provided that they are respecting the constraints. The fittest ***N*** many members from this ***2N*** membered family are picked for the next steps, whereas the unfit members are killed, as so in evolution in the nature.

Step 8 For the first few generations, the individuals will be diverse, however as the iterative process continues, new generations will be alike each other. In other words, the solution will start to converge to an optimum solution, therefore the objective and constraint function values get closer and closer. In case the termination time is satisfied, the optimizer is stopped. Otherwise, generation counter is increased by one and the algorithm is directed to process through **Step 3**. Whole process is generated to create ***N*** individuals of the next generation. The procedure is repeated for ***t*** generations and it is assumed that the best individual of the last generation is the optimum (Van den Braembussche, 2008). It is stated in the literature that population size needs to be even (Verstraete, 2012). The author also notes that a probability of 70-90% is given to

crossover operation which causes almost all the parents to experience the reproduction process.

The flow chart including genetic algorithm steps is given in Figure 2.10.

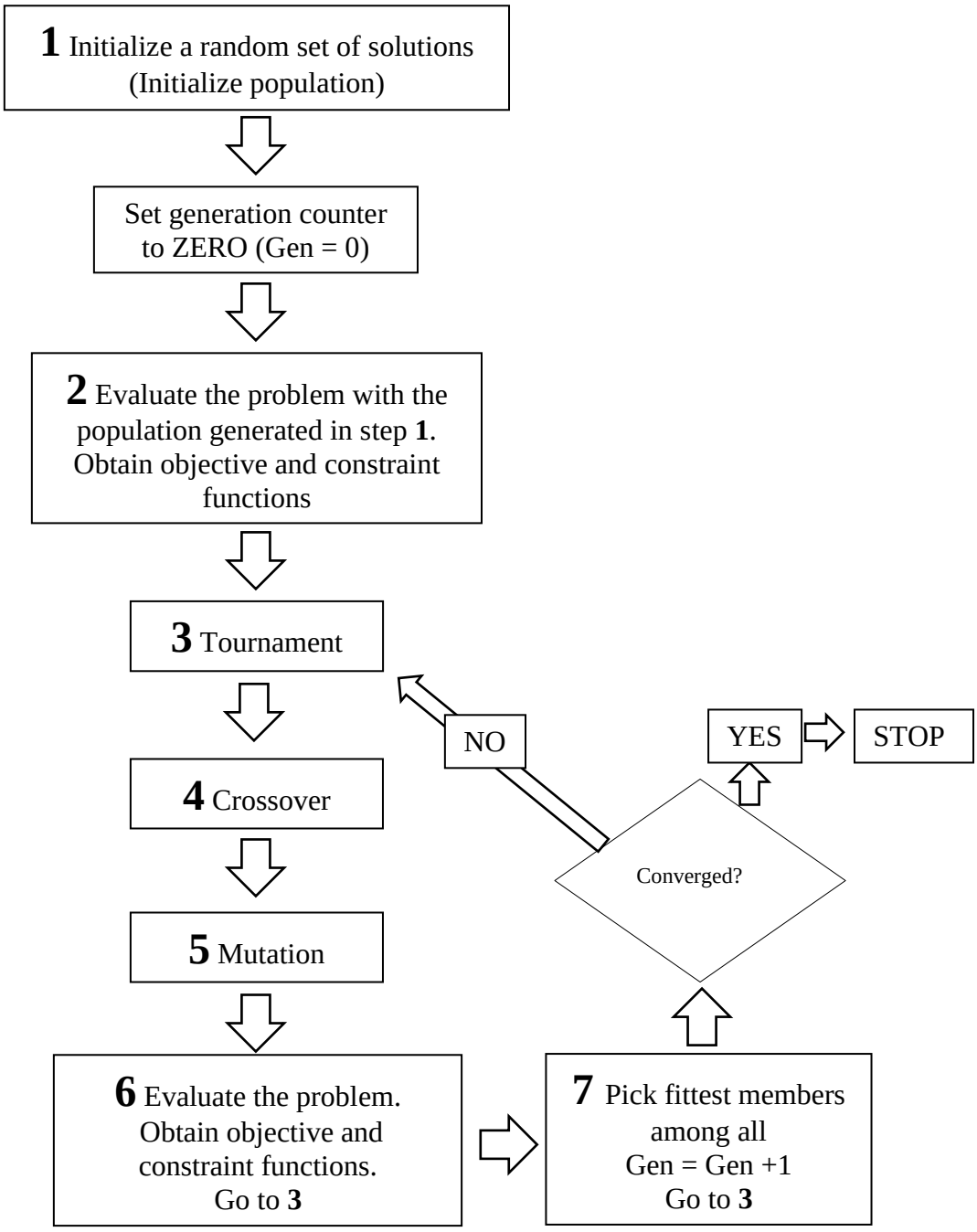


Figure 2.10. Flow chart for genetic algorithm

2.1.3.1.3.2. Differential Evolution

Differential evolution is a global optimization method with an algorithm similar to genetic algorithms. New individuals are generated thanks to the information exchange between the present individuals, but in a different manner. Contrary to the GA, DE works with continuous variables, where new members of the population are generated by adding a weighted difference vector between two population members to a third member as mathematically expressed in Equation (2.13) (Storn & Price, 1997). The decision of keeping the old or the new individual is made by comparing their objective function values: the fittest survives. Schematic representation of the method is presented in Figure 2.11.

$$\overrightarrow{X_{new}} = \overrightarrow{X_1} + R.\left(\overrightarrow{X_2} - \overrightarrow{X_3}\right) \tag{2.13}$$

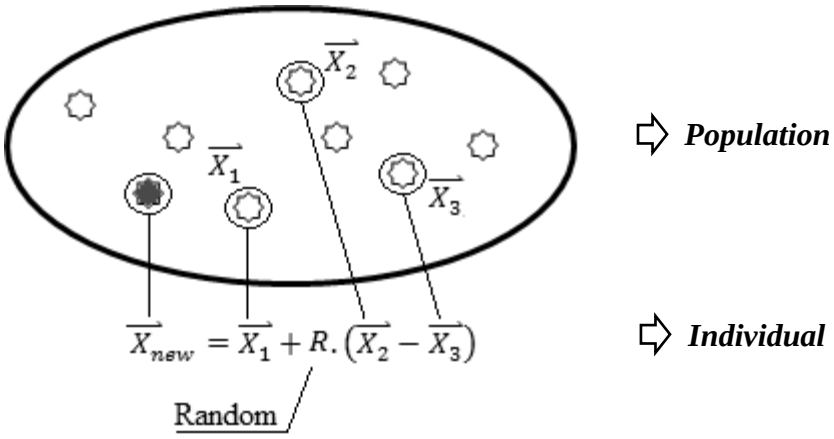


Figure 2.11. Differential evolution

Flow chart presented for genetic algorithm Figure 2.10 is still valid for this method but crossover operator in Step 4 is substituted by differentiation. Mutation operation in Step 5 is not used in DE anymore, however it's reported several studies are being performed for the method to provide more diverse solution sets.

2.1.3.1.4. Methods Inspired by Swarm Intelligence

Swarm intelligence is a relatively new approach, based upon the imitation of the social behaviors of animals and insects (Dorigo, et al., 2006). These methods are inspired by the massive movement of the bird flocks, fish schools, insect swarms or ant colonies that are searching for the optimum path to reach the food.

2.1.3.1.4.1. Particle Swarm Optimization (PSO)

PSO is similar to GA in the sense that both methods are population based. In PSO, the choreographic movements of swarms of birds are imitated, as in such populations the swarm organizes itself and adapts to the environment in its effort to find the optimum path for food, or minimizing the possibility of meeting predators, by means of information exchange between the individuals - in PSO notation, *particles* - and social cognitive intelligence (Hassan, et al., 2005), (Eberhart & Shi, 2001). With this philosophy, any particle within the swarm modifies its behavior thanks to the experience and knowledge of itself, as well as its neighbor particles (Engelbrecht, 2007).

The PSO algorithm is initiated with a random initial swarm population. The initial swarm contains particles that move in the space and search for the best global position over the number of iterations (Safari, et al., 2013). Each particle keeps track of its coordinate history and among all visited positions, the one giving the best objective function value is marked as particle best. Similarly, the location providing the whole population's best objective function value is marked as global best. At each time step, each particle are manipulated towards it particle best and global best locations, in addition to the effect of the particle inertia (Eberhart & Shi, 2001). Explanations given above mathematically can be expressed as given in Equations (2.14) ~ (2.17) (Verstraete, 2012).

$$\overrightarrow{x_k^i} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \\ \vdots \\ x_n \end{Bmatrix}_t \quad , \qquad x_p = rand(x_p^u, x_p^l) \tag{2.14}$$

$$\overrightarrow{v_k^i} = \begin{Bmatrix} v_1 \\ v_2 \\ \vdots \\ v_p \\ \vdots \\ v_n \end{Bmatrix}_t \quad , \quad v_p = rand\left(-\left(x_p^u - x_p^l\right), +\left(x_p^u - x_p^l\right)\right) \tag{2.15}$$

The velocity of the particle ***i*** at (***k+1)***-th iteration is calculated as,

$$\overrightarrow{v_{k+1}^i} = w.\overrightarrow{v_k^i} + c_p.r_p.\frac{\left(\overrightarrow{p_k^i} - \overrightarrow{v_k^i}\right)}{\Delta t} + c_g.r_g.\frac{\left(\overrightarrow{g_k^i} - \overrightarrow{v_k^i}\right)}{\Delta t} \tag{2.16}$$

The position of the particle ***i*** at (***k+1)***-th iteration is then,

$$\overrightarrow{x_{k+1}^i} = \overrightarrow{x_k^i} + \overrightarrow{v_{k+1}^i}.\Delta t \tag{2.17}$$

The term on the left hand side of Equation (2.16) is the particle's velocity at (k+1)-*th* step. This term, after being multiplied by step time, $\Delta \boldsymbol{t}$, is directly added to the particles position vector (in other words, design variable). It is a function of particle's current velocity, $\overrightarrow{v_t^k}$ (k-*th* step), the particle's position at k-*th* step with respect to the particle's local best position, and the particle's position at k-*th* step with respect to all particles' global best position. The literature lacks specific provision about step time, $\Delta \boldsymbol{t}$; however it is taken as unity in the common practice in PSO literature (Rao, 1996), (Eberhart & Shi, 2001), (Engelbrecht, 2007), (Perez & Behdinan, 2007), (Versraete, 2012), (Engelbrecht, 2007). Hence, Equation (2.16) and (2.17) take the form,

$$\overrightarrow{v_{k+1}^{i}} = w.\overrightarrow{v_k^i} + c_p.r_p.\left(\overrightarrow{p_k^i} - \overrightarrow{v_k^i}\right) + c_g.r_g.\left(\overrightarrow{g_k^i} - \overrightarrow{v_k^i}\right) \quad (2.18)$$

$$\overrightarrow{x_{k+1}^{i}} = \overrightarrow{x_i^k} + \overrightarrow{v_{k+1}^{i}} \quad (2.19)$$

Shi & Eberhart underlines that the presence of the velocity ($\overrightarrow{v_k^i}$) term in Equation (2.18) is similar to the mutation operation in evolutionary algorithms, such as genetic algorithm. It is initially picked as a random number as defined in Equation (2.15) and iteratively changed thanks to Equation (2.16). It can be stated that maximum velocity value can serve as an indirect tool that constrains the global exploration ability of the optimizer; on the other hand, setting velocity to a low value will favor the algorithm towards local search, no matter which value is assigned for inertia weight, ***w***, which is explained in detail in the following paragraph (Shi & Eberhart, 1998).

The first term, ***w***, in Equation (2.16) is named as inertia weight and represents the inertia of the particle, which is utilized as a tool for forcing the PSO optimizer to *directly* search for local or global optimum. Large values of ***w*** promote a global search and in such cases, it is used to enhance the velocity of the particle to prevent a premature convergence of the algorithm. The inertia weight can also be used as a dampener in case the designer wants to perform the search in a narrower design space. This is provided by assigning small values for ***w***, which forces the algorithm to search for a local optimum. Shi & Eberhart report that if ***w*** is picked as zero (***w=0***), the final solution will be strongly dependent on initial population.

Now, we have two values that allows as to favor local or global search in PSO. Among these, a better global exploration control is possible through solely varying inertia weight, ***w***, since its presence is only due to this purpose, whereas velocity, $\overrightarrow{v_k^i}$, also contains information of the particle's history.

For any optimization algorithm, it is usually desired to search for the optimum in a wider design space at the beginning, with the aim of finding a promising design point that is close to global optimum. Having the almost global optimum at hand, the search is progressively narrowed for localizing the search in the almost global optimum neighborhood. The algorithm now will be smoother that the optimizer will work as a fine tuner and find the more accurate global best in the global best neighborhood. Within this aspect, it would be wise to address that to achieve an optimum optimization strategy,

global and local search throughout the process should be balanced. This is possible by using an inertia weight decreasing linearly with increasing number of iterations as given in Equation (2.21), instead of a fixed quantity (Shi & Eberhart, 1998), (Eberhart & Shi, 2001). It should be noted that number of iterations are given as an input to the PSO optimizer just before the computation.

$$w_{k-th\ iteration} = w_{final} - \left(\frac{w_{final} - w_{initial}}{k_{final}}\right) k \tag{2.20}$$

where $w_{initial}$ and w_{final} are values of the inertia weight at the initial and final iterations, *k* representing the iteration number. Typical values for w_{final} and $w_{initial}$ are 0.9 and 0.4, respectively (Shi & Eberhart, 1998), (Eberhart & Shi, 2001). The authors state that, by this way, one can reach to the optimum solution in less number of iterations. Change of inertia coefficient, ***w***, with iteration number, ***k,*** is plotted in Figure 2.12 to provide a clear understanding.

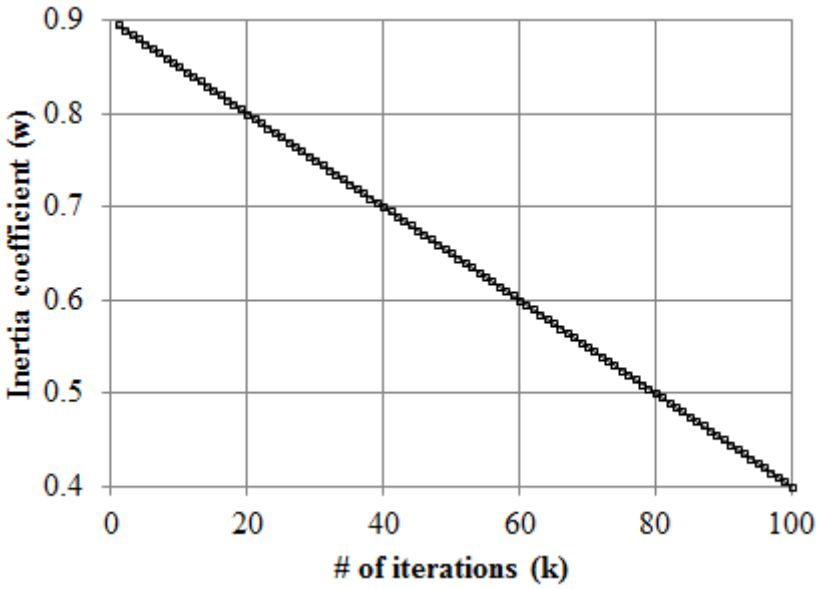


Figure 2.12. Change of inertia coefficient with iteration number

The constants $c_p.r_p$ and $c_g.r_g$, refers to cognitive attraction and social attraction, respectively. r_p and r_g are random weight factors that range between 0 and 1, where these variables are used to provide a stochastic approach to determine how much each factor makes effect on the particle's new position (Cagnina, et al., 2011). Randomness of r_p and r_g are reported to be useful in providing effective design space exploration, thus, increasing the diversity. Presence of r_p and r_g also prevents the early convergence of the algorithm to a non-optimal solution (Trelea, 2003).

Earlier studies recommend to assign c_g and c_p each equal to 2, based on the knowledge collected by trial and error. This practice is recently left, since it is demonstrated by a pure algebraic analysis that PSO stability can be obtained given that following criteria are satisfied (Perez & Behdinan, 2007):

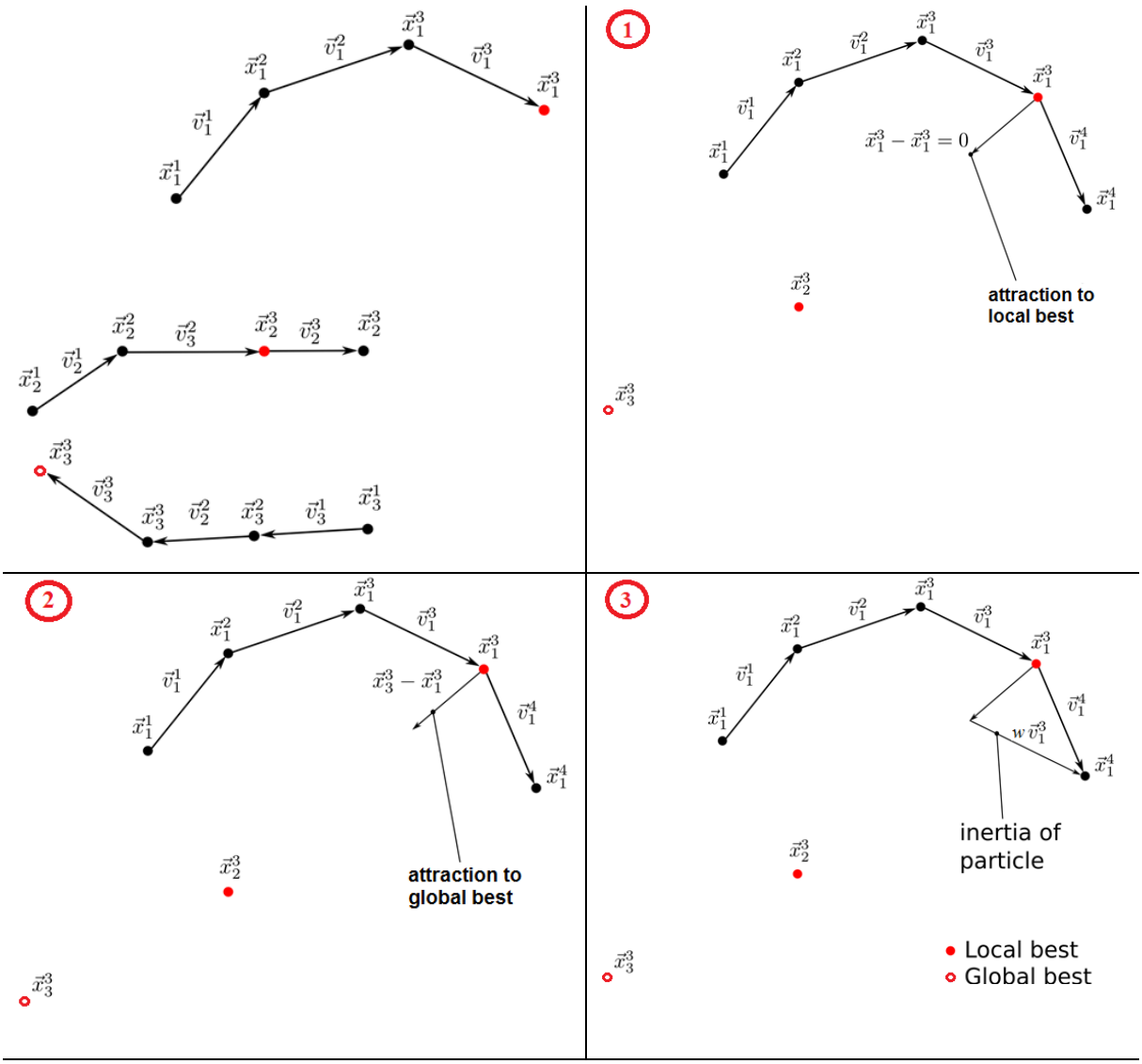
$$0 < c_g r_g + c_p r_p < 4 \tag{2.21}$$

$$\frac{c_g r_g + c_p r_p}{2} - 1 < w < 1 \tag{2.22}$$

The authors' note that Equations (2.21) and (2.22)do not necessarily provide a local or global minimizer; instead, a stable equilibrium point is obtained. In throughflow optimization, $c_g r_g$ and $c_p r_p$ values are taken as 1.25 and 0.5.

It is suggested that particle swarm optimization is computationally more efficient with respect to genetic algorithm for unconstrained nonlinear problems with continuous design variables (Hassan, et al., 2005).

A schematic representation of PSO is provided in Figure 2.13. The flow chart of particle swarm optimization is given in Figure 2.14.



$$\overrightarrow{v_t^{k+1}} = \underbrace{w.\overrightarrow{v_t^k}}_{\mathbf{3}} + \underbrace{c_p.r_p.\left(\overrightarrow{p_t} - \overrightarrow{x_t^k}\right)}_{\mathbf{1}} + \underbrace{c_g.r_g.\left(\vec{g} - \overrightarrow{x_t^k}\right)\overrightarrow{x_t^{k+1}}}_{\mathbf{2}} = \overrightarrow{x_t^k} + \overrightarrow{v_t^{k+1}}.\Delta t$$

Figure 2.13. Particle Swarm Optimization working principle
(Source: Verstraete, 2012)

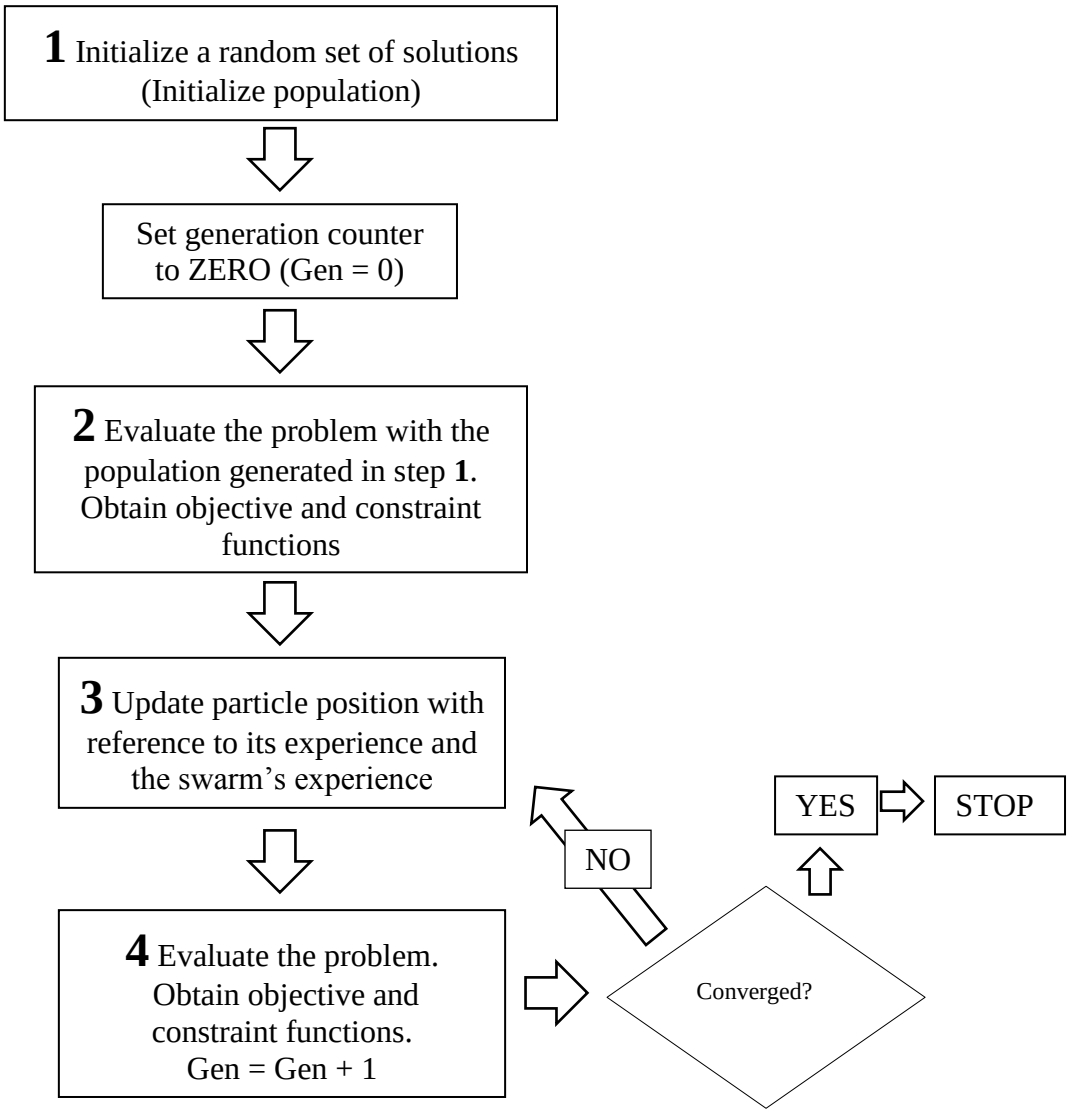


Figure 2.14. Flow chart for particle swarm optimization

2.1.3.1.4.2. Ant Colony Optimization

Ant colony optimization is a technique inspired from a number of models of different behaviors observed in ant and termite colonies (Engelbrecht, 2007). The biological process that is imitated by ACO is as follows: Argentine ants deposit a chemical on the ground, namely ***pheromone*** that can be easily sensed and traced by subsequent ants, which are likely to follow the trail rather than moving randomly (Zervogiannis, et al., 2001). However, it is shown by biologists that ***pheromone*** is an evaporative substance and after a certain time it cannot be sensed by the ants. On the

other hand, with the increasing number of ants passing through the same route, the pheromone effect gets stronger and stronger, causing the ants at the beginning of their journey to select the correct direction, Figure 2.15. The ants, which are still in the nest and thinking where to go, choose their direction by following the route in which ***pheromone*** concentration is strongest. The probability of the ants to select the non-optimum path is small since the pheromone amount in that route is weak. (Dorigo, et al., 2006).

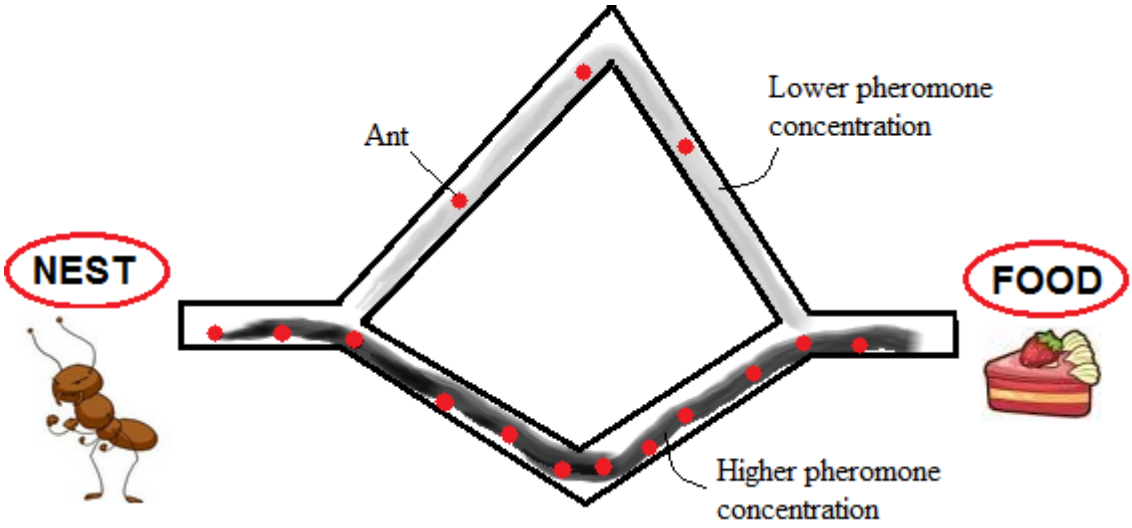


Figure 2.15. Ant Colony Optimization Strategy

Ant Colony Optimization algorithm is initialized with a starting population. After initialization, the program iterates following the given steps:

i. At each iteration, the problem is evaluated with present design parameters,

ii. Optionally, solutions found in Step I are subjected to local search,

iii. Pheromone is updated such that subsequent ants (i.e. design parameters) are directed to the values that provided best results in the latest iteration.

Mathematically speaking, the procedure summarized can be summarized as follows:

An ant at location ***i***, uses ***pheromone*** trail τ_{ij} to calculate the probability of choosing ***j*** as the next node:

$$p_{ij}^{(k)} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}}{\sum_{j\epsilon N_i^{(k)}} \tau_{ij}^{\alpha}}, & if\ j \in N_i^{(k)} \\ \\ 0, & if\ j \notin N_i^{(k)} \end{cases} \tag{2.23}$$

where α denotes degree of importance of the pheromones and $N_i^{(k)}$ deals for the set of neighborhood nodes of ant k when located in node i (Rao, 1996). The ***pheromone***, $\boldsymbol{\tau_{ij}}$, associated with the edge joining nodes ***i*** and ***j*** is updated as,

$$\tau_{ij} \leftarrow (1-\rho).\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{(k)} \tag{2.24}$$

with **ρ** dealing for evaporation rate and m standing for number of ants. $\boldsymbol{\Delta\tau_{ij}^{(k)}}$ is the quantity of pheromone laid on edge (***i*** , ***j***) by ant ***k*** and formulized as given in Equation (2.25), where ***Q*** is a constant and $\boldsymbol{L_k}$ is the length of the tour constructed by ant ***k***.

$$\Delta\tau_{ij}^{(k)} = \frac{Q}{L_k} \tag{2.25}$$

Methods inspired by swarm intelligence are used in many technological applications. However, PSO is reported to be “in the making” and many versions are likely to appear (de Weck, 2010).

2.1.3.2. First Order (Gradient Based) Methods

Zero order methods described in the previous sections requires many function evaluations to converge to the optimum. On the contrary, gradient based methods exploit the derivative information of the function and are usually faster search methods (Deb, 2010).

Some algorithms require the use of only first derivatives (Equation (2.26)) of the objective function, whereas some others also need second derivatives (in case ***n=2*** in Equation (2.27)). The first and second order derivatives are obtained by means of forward or central difference techniques (Rao, 1996), (Deb, 2010).

$$\nabla f_{n\times 1} = \begin{Bmatrix} \partial f/\partial x_1 \\ \partial f/\partial x_2 \\ \vdots \\ \partial f/\partial x_n \end{Bmatrix} \tag{2.26}$$

$$H(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 x_n} \\ \frac{\partial^2 f}{\partial x_2 x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n x_1} & \frac{\partial^2 f}{\partial x_n x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix} \tag{2.27}$$

The speed and accuracy of gradient based methods is highly dependent on the reliability of the initial guess, as providing a good starting point, the algorithm would find the local optimum with ease (Rao, 1996). The search for optimum is solely performed by mathematical operations; therefore the optimizer's stability will highly be dependent on objective/constraint functions' differentiability: In case the problem is non-differentiable, it will not be possible to perform gradient optimization, or the solution will eventually suffer stability problems. Furthermore, in some problems, the design variable may only be allowed to take integer values, likewise the number of rotor-stator stations in an aero-engine, where differentiation will again arise as a problem. Such situations are reported as

the main limitations and also drawbacks of gradient based optimizers (Giles & Pierce, 2001).

A novel method, namely adjoint optimization, recently used not only in turbomachinery, but also in other fluid machinery optimization benefits the service of adjoint operator and control theory that is based on derivative information (Shahpar, 2012), (Papadimitrou, et al., 2005). The sensitivity of the objective function with respect to preselected design variables is the backbone of the algorithm. The method is powerful, accurate and cheap, in terms of computational time, also the unstability problem is now eliminated. On the other hand, this approach is helpful in the context of gradient-based optimization and therefore has its own limitations. As observed in other gradient based optimizers, the method is applicable once it is a priori known that objective function does not have many local minimas, otherwise, it generally converges to the nearest local minimum. In addition, the complexity and programming difficulties of adjoint method (in particular, discrete adjoint method) makes it less popular, despite of its many aforementioned advantages (Giles & Pierce, 2001), (Wang & He, 2010).

In addition to those summarized above, there are many gradient based optimization algorithms. The explanation of complicated mathematical background of such methods is out of this thesis. On the other hand, a very basic gradient based algorithm, steepest descent method, is explained below, that provides the essentials of gradient based optimization techniques.

2.1.3.2.1. Steepest Descent Method

Steepest descent method is one of the most widely known gradient based methods. The method is capable of finding the maximum or minimum of unconstrained problems; whereas special treatment is required once there exist constraints that are needed not to be violated.

This method is a first order gradient based method. By definition, the first derivative of the objective function, $\nabla f(x^{(t)})$, at any point represents the direction of the maximum increase of the function value. Therefore, one who searching for the minimum point should search along the opposite direction, $-\nabla f(x^{(t)})$. Thus, objective functions

evaluated by the new $x^{(t)}$ value in the direction inside the hatched region in Figure would have a smaller value than the ones obtained by the old $x^{(t)}$.

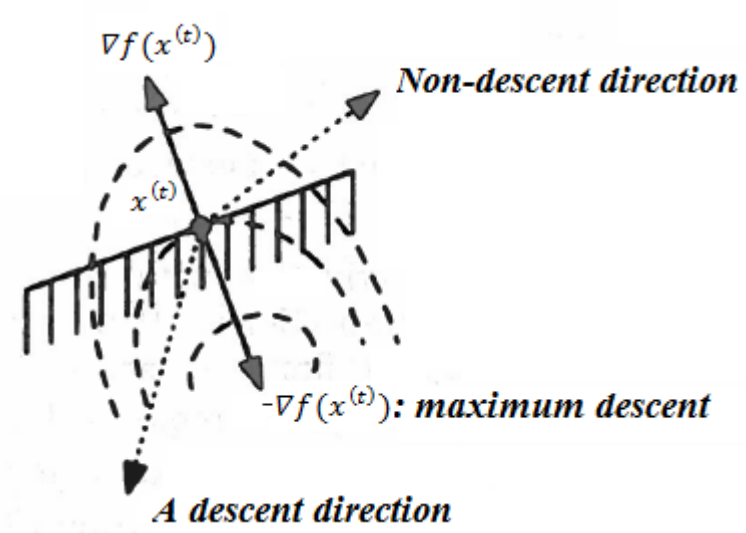


Figure 2.16. Descent and non-descent directions
(Source: Deb, 2010)

This method of obtaining the search direction is called as ***Steepest Descent Method*** (so known Cauchy's Method) and formulized as given in Equation (2.28). Now, one can obtain his/her new design point using Equation (2.29).

$$\vec{S}_i = -\nabla f(\vec{X}_i) \tag{2.28}$$

$$\vec{X}_{i+1} = \vec{X}_i + \alpha_i^* \vec{S}_i \tag{2.29}$$

Where $\vec{S}_i$ is the search direction and α_i^* is the step length between two design points.

Steepest descent method allows for the approximate calculation of the steepest path to the investigated function's optimum value (Carroll, 1961). However, as mentioned before, the method is capable of finding the local minimum or maximum of any optimization problem without any constraints. For the proper handling of the constraints, several approximations are developed for finding the maximum or minimum

of multivariable constrained optimization problem, in which steepest descent method is utilized properly.

2.1.3.3. Constraint Handling with Penalty Function Method

The presence of inequality constraints given in Equation (2.2) makes it difficult and time consuming to obtain an optimum solution in a direct way. In such a situation, one may prefer to convert the constrained problem into unconstrained by employing pseudo-objective function.

Assume that we are trying to optimize the problem given in Equation (2.30), $f_i(\vec{x}) = 0$, subjected to the inequality constraints given in Equation (2.2), $g_j(\vec{x}) \leq 0$. The problem is converted into an unconstrained problem as given in Equation (2.30).

$$\varphi_i = \varphi(\vec{x}, r_k) = f_i(\vec{x}) + r_k \sum_{j=1}^{m} G_j[g_j(\vec{x})] \tag{2.30}$$

Where $\boldsymbol{\varphi_i}$ is the pseudo-objective function, $\boldsymbol{G_j}$ is a function of constraint $\boldsymbol{g_j}$ and $\boldsymbol{r_k}$ is the penalty multiplier, which is a constant that should be a positive number defining the weights given to satisfying the constraints (Verstraete, 2012). The method works with successive iterations where the penalty multiplier, $\boldsymbol{r_k}$, is modified at each iteration with reference to previous sequences (Rao, 1996) , (Deb, 2010). The constrained problem is, now, transformed into a sequence of unconstrained problems by defining penalty terms for each constraint violation (Deb, 2010). Methods used for transforming a constrained optimization problem into an unconstrained optimization problem by means of transformation techniques given in Equation (2.30) are called as *Penalty Function Methods*.

It is noted in the literature that, the pseudo objective function is “ill conditioned” and some constraints may remain unsatisfied at the end of optimization process (Vanderplaats, 1984). The final decision on whether to choose objective or pseudo

objective function will be given by the optimizer that a compromise will have to be made between computation time and accuracy.

Most commonly used penalty function methods are *interior* and *exterior* penalty methods, where it is the form of function $\boldsymbol{G_j}$ that determines the type of the method.

Once the penalty function method is taken into service, the optimization problem is said to be transformed to an alternative formulations such that the new form of the problem is unconstrained. Thereafter, any optimization method can be used for finding the optimum point.

Interior Penalty Function Method: The interior penalty function method is first developed by Carroll (Carroll, 1961). The method is also called as barrier function method because a large barrier is constructed around the feasible region (Arora, 2004) . A penalty function method is said to be interior in case $\boldsymbol{G_j}$ is in the forms given in Equation (2.31) and (2.32) where most popular interior penalty functions are listed below.

$$G_j = -\frac{1}{g_j(\vec{x})} \tag{2.31}$$

$$G_j = \log[-g_j(\vec{x})] \tag{2.32}$$

The method is initialized with a ***feasible point*** that satisfies all constraints with a strict inequality sign (pseudo objective function tends to infinity as $\boldsymbol{g_j(\vec{X}) = 0}$). Following the initialization, steepest descent method is utilized for minimizing the pseudo objective function given in Equation (2.30), where at each sequence, the penalty function is added to the objective function. Same procedure is repeated successively until the optimum point respecting the constraints is obtained. Note that the penalty multiplier, $\boldsymbol{r_k}$, is assigned a large value for the first iteration and degraded in the following sequences as this will prevent the algorithm from reaching the barrier of the feasible region (Jensen & Bard, 2002). A representative problem is graphed in Figure 2.17a in order to explain the general characteristic of the interior penalty function method. Within this aspect, the algorithm summarizing interior penalty function method is given in Appendix B. Step by step explanation of the method can be listed as follows:

Step 1: Set i=1. Start with a feasible initial design vector, $\vec{X}$, satisfying all constraints with a strict inequality sign: $g_j(\vec{X}) < 0$. Set $k = 1$.

Step 2: Set an initial value for penalty multiplier, r_k, respecting that $r_k > 0$

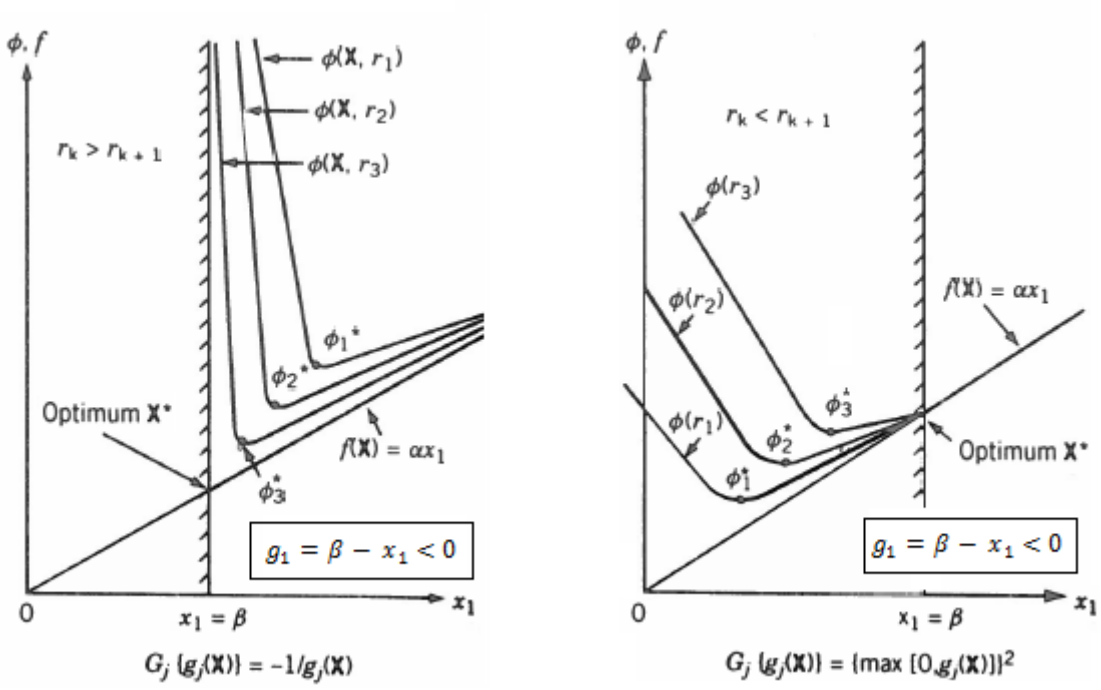
Step 3: Minimize pseudo objective function, $\varphi(\vec{X}, r_k)$ using ***Steepest Descent*** method and obtain $\vec{X}^*$.

Step 4: In case $\vec{X}^*$ is optimum, terminate the process. Otherwise, proceed to **Step 5**.

Step 5: Set $r_{i+1} = c\,.r_i$, where $c < 1$. $g_1 = \beta - x_1 < 0$

Step 6: Set $k = k + 1$ and $\vec{x} = \vec{x}^*$. Go to Step 2.

Since genetic algorithm provides a feasible starting point, interior penalty function method is selected as the gradient based optimizer in this study.



(a) **(b)**

Figure 2.17. (a) Interior and (b) Exterior penalty function methods
(Source: Rao, 1996)

Exterior Penalty Function Method: In this method, the penalty function term in the pseudo-objective function is commonly defined as given in Equation (2.33).

$$G_j = \{\max[0, g_j(\vec{X})]\}^q \quad (2.33)$$

It is worth noting that G_j takes the value of ZERO once the constraints are satisfied, i.e. $\boldsymbol{g_j(\vec{x}) \leq 0}$; otherwise, q-th power of constraint function is added as penalty function to the pseudo objective function. Contrary to the interior penalty function method, an infeasible point is selected as the initial solution. In addition, penalty multiplier, $\boldsymbol{r_i}$, , is increased in successive iterations in contrast to interior penalty function method (Rao, 1996), (Jensen & Bard, 2002). A representative problem is graphed in Figure 2.17b in order to explain the general characteristic of the interior penalty function method. Within this aspect, the algorithm summarizing exterior penalty function method is given in Appendix A. Step by step explanation of the method can be listed as follows:

Step 1: Set i=1. Start with any initial design vector, $\vec{X}$. Set $k = 1$.

Step 2: Set an initial value for penalty multiplier, r_k, respecting that $r_k > 0$

Step 3: Minimize pseudo objective function, $\varphi(\vec{X}, r_k)$ using Steepest Descent method and obtain $\vec{X}^*$.

Step 4: In case $\vec{X}^*$ is optimum, terminate the process. Otherwise, proceed to **Step 5**.

Step 5: Set $r_{k+1} > r_k$, where $c > 1$.

Step 6: Set $k = k + 1$ and $\vec{X} = \vec{X}^*$. Go to Step 2.

For genetic algorithm and particle swarm optimization, exterior penalty function method is used for constraint handling.

2.1.3.4. Comparison of Zero Order and Gradient Based Methods

The literature (Shahpar, 2000) summarizes the main advantages of zero order methods as follows:

i. The search is population based, and it is not bounded by a single parameter set.

ii. They are powerful in finding global optima.

iii. There occurs no necessity for calculating gradients.

iv. They are efficient in optimization problems where objective functions are discrete and non-differentiable.

The disadvantages are stated like (Andersson, 2001),

i. It cannot be proven that the optimum found by the zero order method is the actual optimum. However, by conducting several optimization runs with different initial conditions, it could be made probable that the global optimum is truly found.

ii. EA methods require more function evaluations than gradient based methods, and are thus computationally more expensive.

It is noted that gradient based methods are more efficient due to the fact that they use gradient information (Versraete, 2012). However, in these methods the search is from a point to another point, therefore the probability of finding a local peak instead of the global is increased significantly as compared with EA and swarm intelligence methods (Öksüz & Akmandor, 2005).

As mentioned above, gradient based methods exploit the derivative information of the function and are usually faster search methods. However, they are not applicable on aerodynamic design problems since such problems are contains non-differentiable functions containing many singularities. A recent study (Mengitsu & Ghaly, 2008) states that since aerodynamic design optimization problems are multi-modal and discontinuous in nature, gradient-based numerical optimization algorithms risk of getting trapped in local minima or run into an infeasible design for which the flow simulation does not converge. It is underlined in the literature that the recent trend in industry is to go away from gradient based methods to evolutionary optimization strategies using surrogate methods like artificial neural networks (Shahpar, 2012).

The drawbacks experienced in the use of zero order and gradient based methods are minimized by hybridizing these methods in order to obtain the exact optima. The idea behind hybrid techniques is to switch to gradient based method once the zero order method reaches its convergence criteria. From this point on, zero order method optimization solution is taken as the starting point for the gradient based search. This approach takes the advantages of both zero order methods and gradient-based methods and can improve the search efficiency and ensure the global search ability simultaneously (Lian & Liou, 2005).

2.2. Literature Review on Turbomachinery Optimization

2.2.1. Basics of Aero-Engine Fan/compressor Aerodynamics

A turbo-machine can be defined as a device which transfers mechanical energy in the form of a shaft work, to or from a continuously flowing fluid by the dynamic action of the blade rows (Baskharone, 2006). The turbomachinery family involves many members such as pumps, propellers, hydraulic turbines, axial/centrifugal compressors, axial fans, aircraft engines, helicopter engines etc. The main point of interest of this study is the design optimization of the fan/compressor module of a turbofan engine, see Figure 2.18.

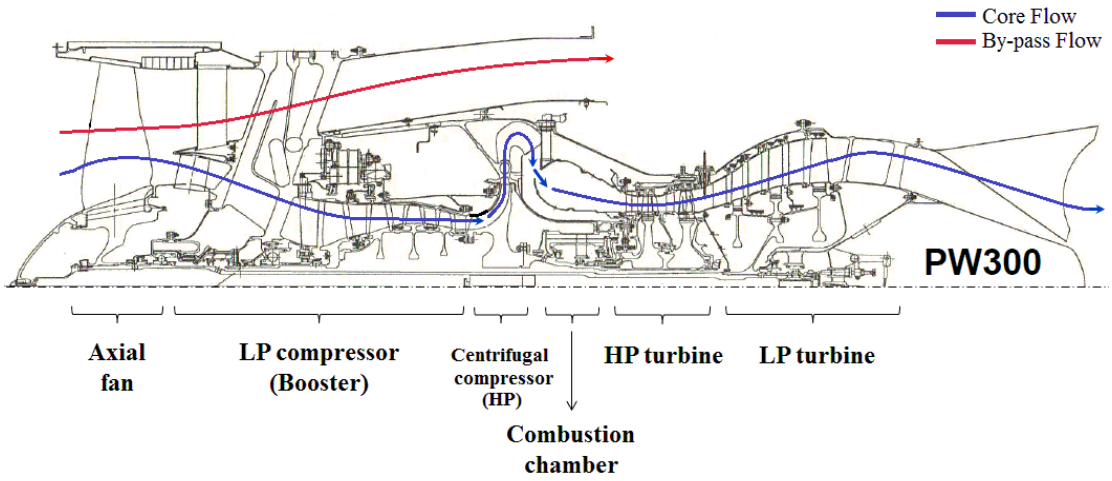


Figure 2.18. A typical turbofan engine (Pratt and Whitney - PW300)

In a turbofan engine, axial fan provides pre-compression of the air passing through the core components (i.e. compressor, combustion chamber and turbine) and also pressurizes bypass nozzle air, which makes a significant contribution to the thrust of the engine. In addition to the pressure rise provided in the fan, axial/radial compressor conditions air such that the pressure demand for an optimum combustion process is satisfied. Components in charge of compression (i.e. compressor and fan) have the same

working mechanism but different functions, where they use the shaft work generated by low pressure (LP) and high pressure (HP) turbines.

2.2.2. Basic Fan/Compressor Design

The turbomachinery design starts with the definition of the machines mission: electricity production, marine propulsion or aircraft propulsion (Dénos, 2005). As mentioned before, the focus of this study will be on aircraft propulsion machines, in particular turbofans. The design procedure continues as stated in the following:

1. **Cycle definition:** In order to proceed to the design of the fan, one has to decide the complete thermodynamic cycle of the engine. Thermodynamic cycle of the engine is calculated with reference to the required thrust that the final engine is desired to provide. Mass flow rate of the engine is calculated in this phase.
2. **1D meanline design:** The design is preceded with 1D meanline design: Number of stages (rotor-stator couple) and the decision of work distribution between stages are made. Velocity triangles at the meanline (i.e. the line passing through hub and shroud curves of the fan/compressor) are drawn, thermodynamic quantities before and after each rotor/stator station is decided in this phase. The general geometry of the flowpath in streamwise direction is provided as it will be modified iteratively in the following steps.
3. **Throughflow and blade-to-blade design:** Starting with an initial guess, the three dimensional unsteady flow in turbo-machines is approximated iteratively with low fidelity flow solver, namely throughflow method (Chauvin, 1977). Inlet and outlet angles of the airfoils are decided at a number of spanwise locations. Blades are generated either by using standardized blade profiles (i.e. NACA, DCA, MCA) or by designing custom profiles.
4. **3D design:** 2D airfoils obtained in blade-to-blade design are stacked and 3D blades are built. Final design is subjected to high fidelity Navier-Stokes solutions. The design should be checked and adjusted with reference to the results of the Navier-Stokes calculations.

The algorithm summarized above is given as a flow chart in Figure 2.19.

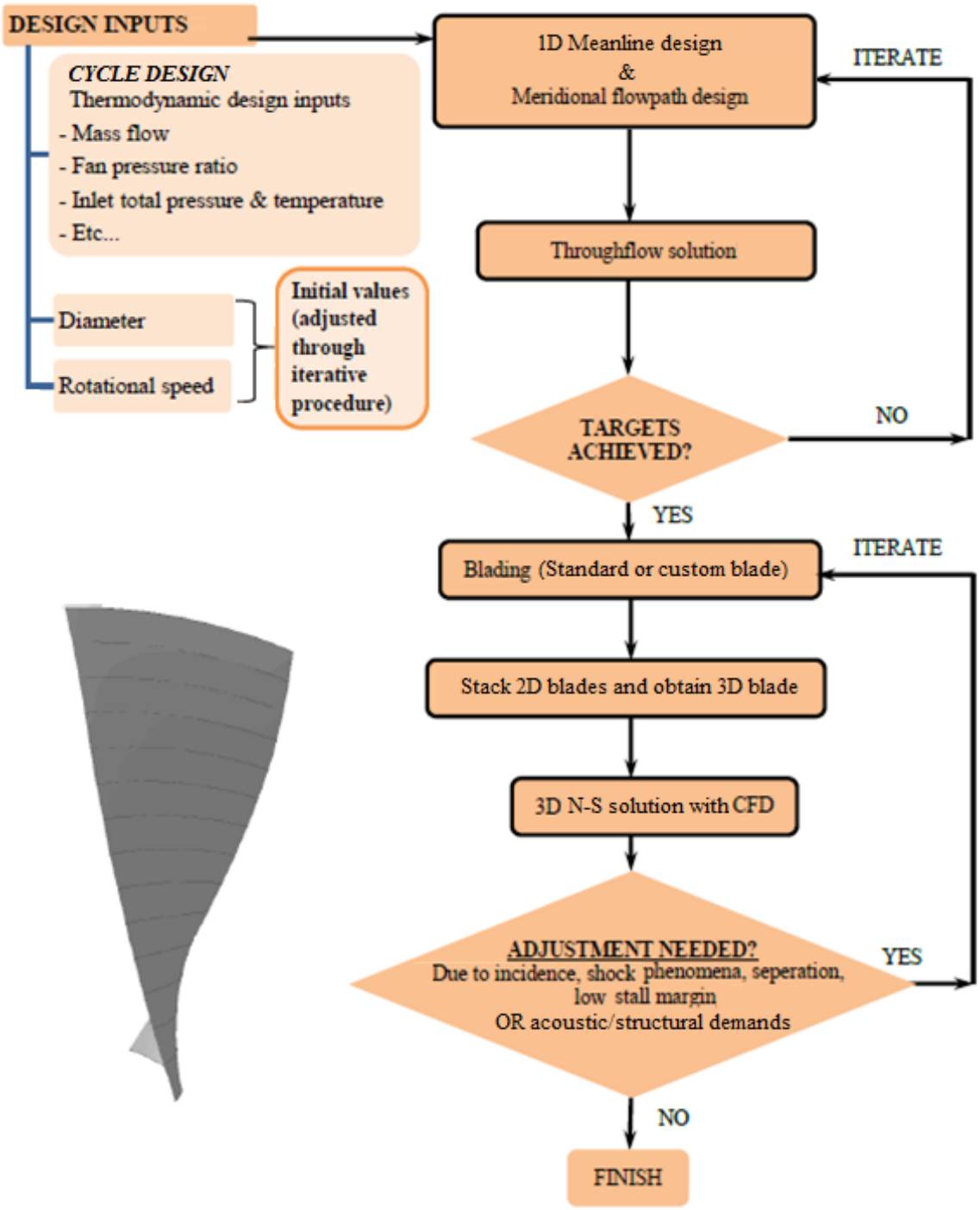


Figure 2.19. Fan/compressor design flow chart
(Source: Acarer, 2015)

2.2.3. Turbomachinery Optimization

Evolutionary methods have recently been used in turbomachinery optimization, in particular, in throughflow, blade-to-blade and 3D optimization of compressor and turbines aspects.

Early studies reported by (Massardo, et al., 1990) and (Massardo & Satta, 1990) use a gradient method in optimizing their axial compressor's meanline and throughflow design while parameterizing their variables with polynomials. Their study revealed that a ~0.05 increase in compressor efficiency is possible while computational cost is quite low. However, as noted previously, it is expected that the optimization scheme used in the study lacks of finding the global optimum.

In a recent study, the throughflow design of a highly loaded fan's first stage is optimized using differential evolution (DE) with the aim of maximizing efficiency, minimizing tip radius and minimizing the relative flow turning at sections with high Mach number, respecting to the constraints that has to be obeyed due to mechanical reasons. The flowpath just after the by-pass splitter is not modeled in the scope of this study. Their optimization strategy is enabled a 65% attenuation of the stress compared to an aero-only optimization at the cost of 1% in aerodynamic efficiency. The author's final geometry provides a pressure ratio of 2.1 with a flow efficiency of 88% and a structural safety margin of 32% (Joly, et al., 2012).

An effort of hybridizing genetic algorithm with both, simplex method, simulated annealing and gradient search method is performed by (Petrovic, et al., 2001), for the purpose of optimizing the throughflow model of a multistage turbine, achieving a 1.17% gain in efficiency with respect to their initial design. The author's selected the hub and shroud geometries as the flow variables, as well as spanwise distributions of tangential velocity following each blade row.

An application of throughflow optimization of a mixed flow turbine was reported by (Cox, et al., 2010) , as differential evolution is selected as the optimizer. The optimum geometry provided a 3% increase in efficiency. Another case investigated in the same study showed that a design with 20-30% less inertia can be obtained while keeping efficiency in the same level.

A 3-stage booster is optimized by (Park, et al., 2011) in order to obtain maximum efficiency, while providing minimum mass flow rate, booster length, and rotor/stator

blade numbers. The study utilized genetic algorithms to approach the global optimum, while exact optimum is obtained by are hybridizing the optimizer with gradient based methods. 0.5~1.5% efficiency increase is observed at the end of the optimization procedure, whereas by-pass flow path and booster inlet velocity distribution (i.e. fan exit velocity distribution close to hub) are not optimized.

Genetic algorithms and ANN are used for the optimization of a turbine blade seeking for maximum efficiency in a recent work (Öksüz & Akmandor, 2005). It is reported that an efficiency value of %85.9 is obtained, corresponding a 2% increase with respect to the baseline design. Resulting geometry can be seen in Figure 2.20. The blade geometry is parameterized using Bezier curves.

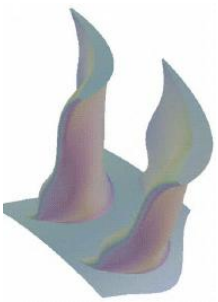


Figure 2.20. Optimum blade geometry
(Source: Öksüz & Akmandor, 2005)

A study reported in 2002 employs genetic algorithms for the purpose of obtaining an aerodynamically optimum turbine blade by minimizing the secondary flows, which is possible by adjusting the turning angle and the stacking line. The blades are leaned at the endwall region for the same purpose. After 2 iterations, they obtained an increase in efficiency in the order of 0.4% , Figure 2.21. The blade geometry is parameterized using the parametric blade generator NUMECA AutoBlade © (Pierret & Hirsch, 2002).

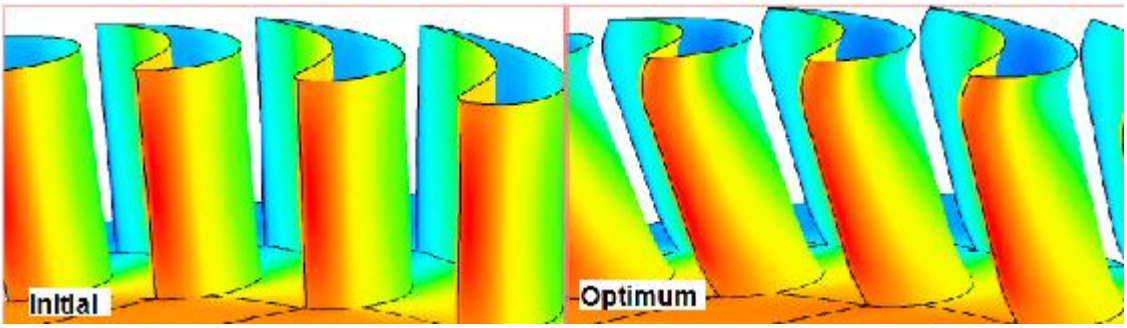


Figure 2.21. Initial and optimum blade geometries
(Source: Pierret & Hirsch, 2002)

A NACA65 compressor airfoil is optimized by (Mengitsu & Ghaly, 2008) in a two dimensional space using genetic algorithms and ANN, where the blade geometry is parameterized using non-uniform rational B-spline (NURBS). The improvements obtained by optimization process is given in Figure 2.22.

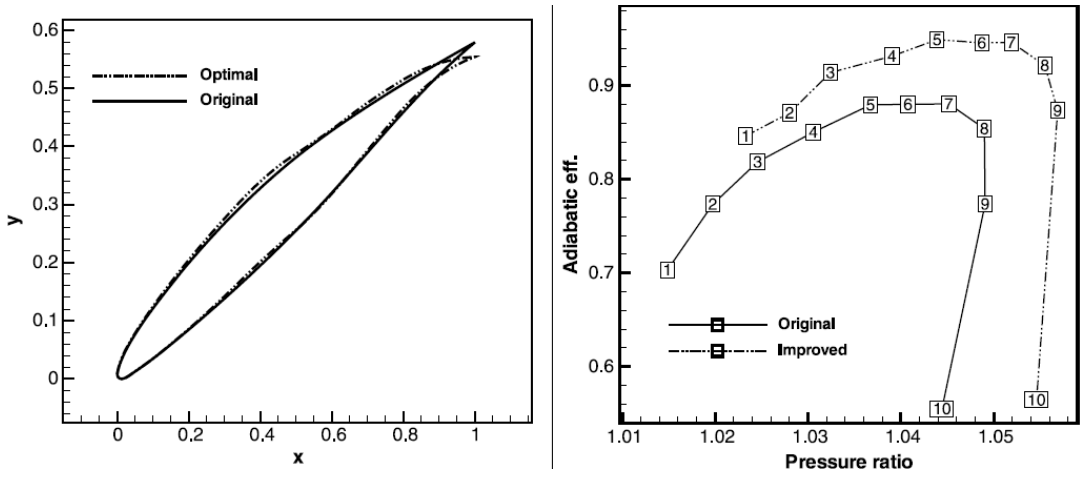


Figure 2.22. Optimum blade geometry and performance
(Source: Mengitsu & Ghaly, 2008)

The transonic axial flow compressor geometry is optimized by (Okui, et al., 2011) using genetic algorithms and ANN. Final geometry has a strong backward swept stacking line, with a barreling chord length which weakens the shock wave in the mid span region. Backward sweep with an optimized S shaped camber line improved the efficiency by 0.6%. The camber line control eliminated the negative effects of the backward sweep on the stability.

Samad & Kim report the use of genetic algorithm for the sake of optimizing an axial compressor rotor blade in a multi-objective sense. The authors' target is to maximize adiabatic efficiency at the rotor exit and to maximize total pressure ratio across the compressor blade. The design variables for this study are blade lean (see Figure 2.23) and blade thickness, where the blade profiles are designed using 3rd order spline and Bézier curve. As seen from the objectives, the problem is not about the design of the geometry from scratch, but it is assumed that the throughflow design of the module is accomplished and the study is built upon the work performed in quasi-3D phase. Reference and optimized blade geometries reported in the study is provided in Figure 2.24. The authors' note that adiabatic efficiency and total pressure ratio are increased by 0.74% and 0.41 % for optimum (1) geometry. For optimum (2) geometry, adiabatic efficiency is increased by 1.76% percent, whereas 2.10% decrease is observed in pressure ratio (Samad & Kim, 2008).

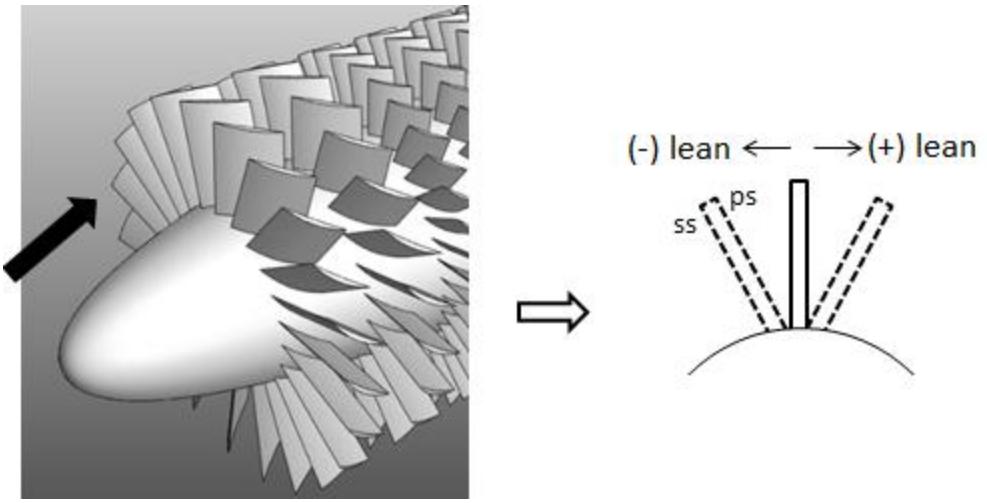


Figure 2.23. Lean definition / ss: suction side , ps: pressure side (front view)

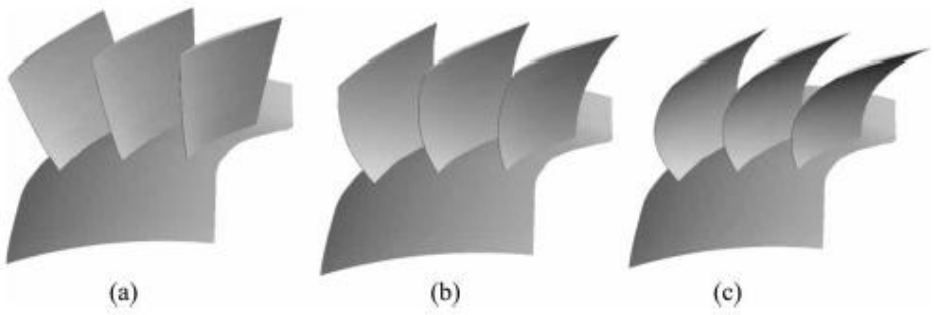


Figure 2.24. (a) Reference, (b) Optimum 1, (c) Optimum 2
(Source: Samad & Kim, 2008)

In addition to the cited literature given above, there exist many other examples of use of evolutionary algorithms, both in throughflow optimization and 3D geometry optimization (Joly, et al., 2012), (Verstraete, 2008), (Verstraete, 2012), (Van den Braembussche, 2008), (Giannakoglou, 1999), (Mengitsu & Ghaly, 2008), (Joly, et al., 2010), (Pierret, 1999) , (Verstraete, 2012), (Öksüz & Akmandor, 2005), (Öksüz, 2011), (Sivashanmugam, 2011). To the author's knowledge, there exist no applications of genetic algorithms on the optimization of throughflow design of fans and the by-pass flow path, where these parameters plays key roles both in thermodynamic and propulsive efficiency of a turbofan. Few exceptions are those reported by (Joly, et al., 2012) and (Joly, et al., 2010) that the main point of interest of these studies is to obtain an optimized throughflow solution of a compressor using DE and the study reported by (Petrovic, et al., 2001), where the module optimized is a multi-stage turbine, not a fan.

To the author's knowledge, PSO and ACO use in turbomachinery optimization is not reported yet such that no published study is available in the open literature. One of the few studies is reported by Guglieri. The author aims to obtain an optimum aerodynamic performance of a helicopter rotor blade by changing the blade geometry. The measure for better aerodynamic performance is picked as minimizing power needed to be provided to the rotor for the same flight regime. Results show that PSO converges to almost same geometries as GA, see Table 2.2. The author eventually states that, PSO is slower but it provides accurate results with that of the GA (Guglieri, 2012). It should be noted that in this turbomachinery optimization problem, external aerodynamics is investigated.

Table 2.2. Output of the optimization study of Guglieri

	Nominal (kwatt)	GA (kwatt)	PSO (kwatt)
$\mu = 0.00$	153	139	138
$\mu = 0.1$	94	74	74
$\mu = 0.2$	91	70	70

2.3. Motivation

It is seen from literature survey that the literature is lack of open/accessible reports of throughflow optimization applications on high by-pass turbofans. Therefore, one of the aims of this study is decided as shedding light on this issue. A hybrid program coupling genetic algorithm and gradient optimizer is decided to be utilized for this purpose, since the method's robustness for turbomachinery problems is frequently stated in the literature. The literature survey also showed that there are no open/accessible studies on throughflow optimization using particle swarm optimization. This untouched area on the field is also another motivation for the author; therefore a chapter of this thesis is dedicated on the application of particle swarm optimization on throughflow optimization. Some details about the throughflow method, namely streamline curvature (SLC), that is to be optimized in this study and the studies reported in the literature about throughflow optimization will be introduced in the following sections. The data coming from cycle design and 1D meanline design (1st and 2nd steps of the design procedure) is taken as the inputs of our work; therefore, the details about these phases are not going to be discussed.

CHAPTER 3

THROUGHFLOW DESIGN

Throughflow design is one of the basic tools used in turbomachinery component design. Spanwise (hub to shroud) variations of fluid thermodynamic and aerodynamic properties (work, mass flow, velocity vectors, temperature, pressure, density, etc.) throughout the machine are determined in this stage (Wisler, et al., 1989).

The computations are performed at (generally) the bladeless domain shown in Figure 3.1 and flow properties are calculated at the locations where stations and streamlines intersect. Stations can either be simple radial lines, leaned lines or arbitrary curves. Moreover they can be aligned with blade leading and trailing edges, otherwise interpolation should be done to obtain information at leading and trailing edges. Basically, one imposes a swirl velocity distribution at the stations following the trailing edges of each blade (e.g. St3 and St5 in Figure 3.1). In what follows, an initial guess for meridional (axial+radial) velocity field is made, practically based on *meanline* design. The streamlines are then re-constructed in order to respect mass continuity, momentum and energy balance meaning that initial velocity guess will be recomputed iteratively. This throughflow technique is also called as Streamline Curvature (SLC) Method.

Several assumptions have to be made in order to perform the procedure explained. These are,

- ✓ The flow through the blade row is axis-symmetric.
- ✓ The flow is assumed to be adiabatic.
- ✓ Viscous effects and body forces are negligibly small.
- ✓ The flow is in steady state condition.

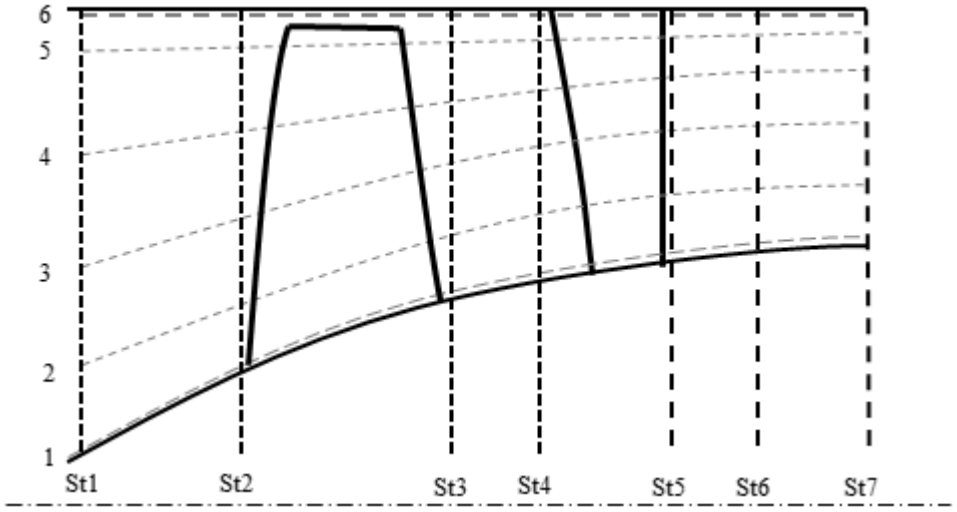


Figure 3.1. Radial stations (St) and streamlines in throughflow calculations

Obviously, such assumptions provide a solution that does not fully correspond to the actual case. In this sense, some modifications are performed to make the throughflow solution more real-like. First, the flow is neither 2D and axis-symmetric, nor inviscid: The presence of blades and the end-walls results in boundary layer growth. The negative effect of boundary layer on entropy generation will be catastrophically amplified once it interacts with the strong shock structures occurring at the shroud region. Moreover, secondary flows originated at the hub region such as corner vortices also play role in entropy generation. Such effects are taken into account by means of cascade calculations or correlations and throughflow calculations are iteratively adjusted.

3.1. Radial Equilibrium Equation

Mathematically, throughflow methods are based on equilibrium of the radial (or along station direction for more general case) forces acting on a fluid particle (see Figure 3.2), where these forces can be summarized as pressure and centrifugal forces (due to the presence of V_θ and streamline curvature).

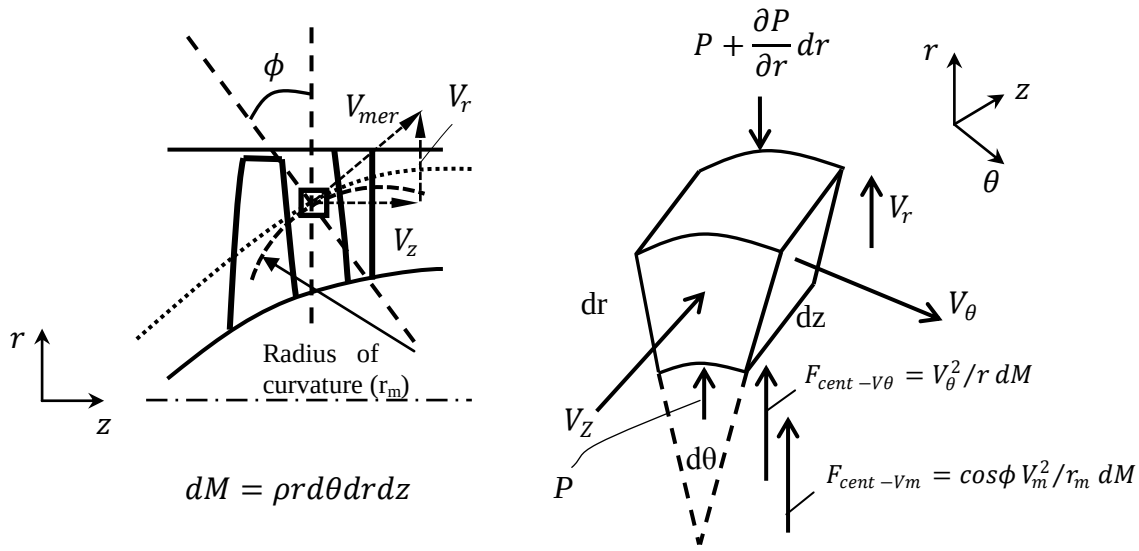


Figure 3.2. Radial equilibrium – geometric representation

The purpose of this study is to optimize the throughflow solution of the fan module of a turbofan, therefore it is not preferred to focus on the derivation of the equations; and instead the physics behind them are given. Complete derivation of the equation is described in (Chauvin, 1977). The final form of the formulation arranged for streamline curvature method is given in Equation (3.1) ~ (3.4) (Chauvin, 1977).

$$\frac{\partial V_m^2}{\partial R} + L(R) V_m^2 = N(R) \tag{3.1}$$

$$L(R) = 2\left[-\frac{\sin(\phi)}{V_m}\frac{\partial V_m}{\partial m} + \frac{\cos(\phi)}{r_m} + \frac{1}{2}\left(\frac{1}{Q}\frac{\partial Q}{\partial R}\right)\right] \tag{3.2}$$

$$N(R) = -2\left[\frac{1}{Q}\frac{\partial (QH)}{\partial R} - \frac{V_\theta}{R}\frac{\partial R V_\theta}{\partial R} - \frac{V_\theta^2}{R}\left(\frac{1}{Q}\frac{\partial Q}{\partial R}\right)\right] \tag{3.3}$$

$$Q = e^{s/c_p} = \frac{\left(\frac{P_{ref}}{P_i}\right)^{\frac{\gamma-1}{\gamma}}}{\frac{T_{ref}}{T_i}} \tag{3.4}$$

where m is the meridional direction (Figure 3.2), V_m is the meridional velocity, V_t is the tangential velocity, r_m is streamline radius of curvature, ϕ is the angle between meridional and axial direction, H is total specific enthalpy. Q is the loss term (at rotor or stator cascades), where P_{ref} and T_{ref} are reference (rotor or stator cascade inlet in this case) total pressure and temperature, respectively; P_0 and T_0 are local (rotor and stator cascade outlet in this case) total pressure and temperature, respectively. The loss in the cascade makes Q smaller than one while it is equal to one for isentropic flow.

3.1.1. Radial Equilibrium Equation: Inputs And Outputs

Inputs for radial equilibrium equation can be listed as:

- ✓ Total pressure and total temperature at the inlet.
- ✓ Total or static pressure at the outlet.
- ✓ Rotational speed.
- ✓ Meridional flowpath (can be given in the form of polynomial or Bezier.curve)
- ✓ Initial guess for blade row inlet velocities and streamlines.
- ✓ **Number of streamlines:** Lower number of streamlines will provide an inaccurate solution, whereas higher number of streamlines will increase the computational cost excessively.
- ✓ **Design choice:** Flow angle distribution at the stations following each row (in radial direction, can be given in the form of polynomial or Bezier.curve). Free vortex, forced vortex, twist factor are some examples for parameterized flow angle distributions reported in the literature (Chauvin, 1977).

Outputs of the radial equilibrium solutions are,

- ✓ Blade inlet angle distributions.
- ✓ Final form of streamlines (note that streamlines represents circumferentially averaged properties)
- ✓ Thermodynamic properties at each station, efficiency.

- ✓ Incidence (difference between flow angle and metal angle at inlet, i) and deviation (difference between flow angle and metal angle at outlet, δ) values, see Figure 3.3 .

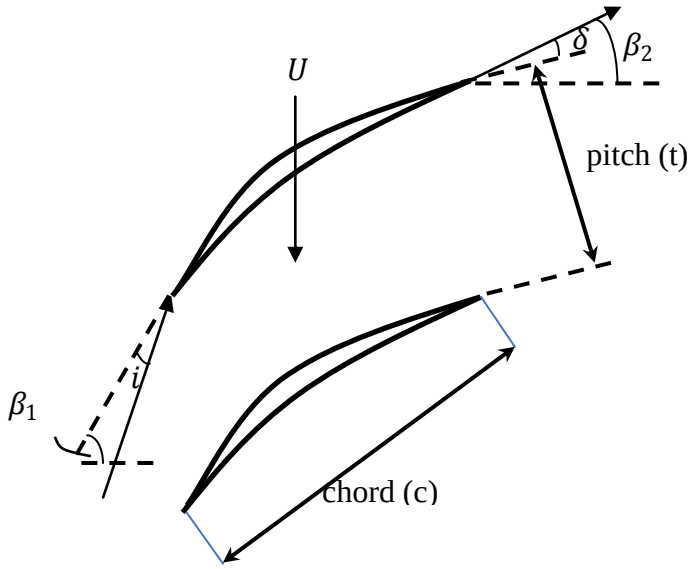


Figure 3.3. Incidence (i), deviation (δ), flow angles (β) and basic geometric parameters

3.1.2. Parameterization of the Input / Output Data

Input variables such as flow angle distribution at the stations (design choice) or meridional flowpath are needed to be parameterized as geometric curves that are composed of several control points.

3.1.2.1. Blade exit angle distribution

The work consumed by the fan/compressor is governed by Euler's turbo-machine equation for axial engines (Equation (3.5)) which states that either radius (so circumferential speed) or blade loading (turning) must be large for a given enthalpy rise.

$$\Delta H = U \times \Delta V_\theta \tag{3.5}$$

The parameterization of the blade exit angle distribution in this study is chosen to be in the form called as **twist factor (TF)**, see Equation (3.6). r is the section (2D blade) radius at the trailing edge, V_θ is the section outlet tangential velocity and $V_{\theta-ML}$ is the exit tangential velocity at meanline (50% span). This directly defines the flow swirl, thus work distribution where ***TF*** is one of the variables that is adjusted to obtain optimum solutions.

$$V_\theta = V_{\theta-ML}\left(\frac{r_{ML}}{r}\right)^{TF} \tag{3.6}$$

As seen from Equation (3.6), higher twist factor, ***n***, means higher tip loading (i.e. turning, -ΔV_θ) and vice-versa (This factor does not change the overall loading of the blade, but redistributes loading over the blade). Following conclusions can be stated from Equations (3.5) and (3.6), also see Figure 3.4:

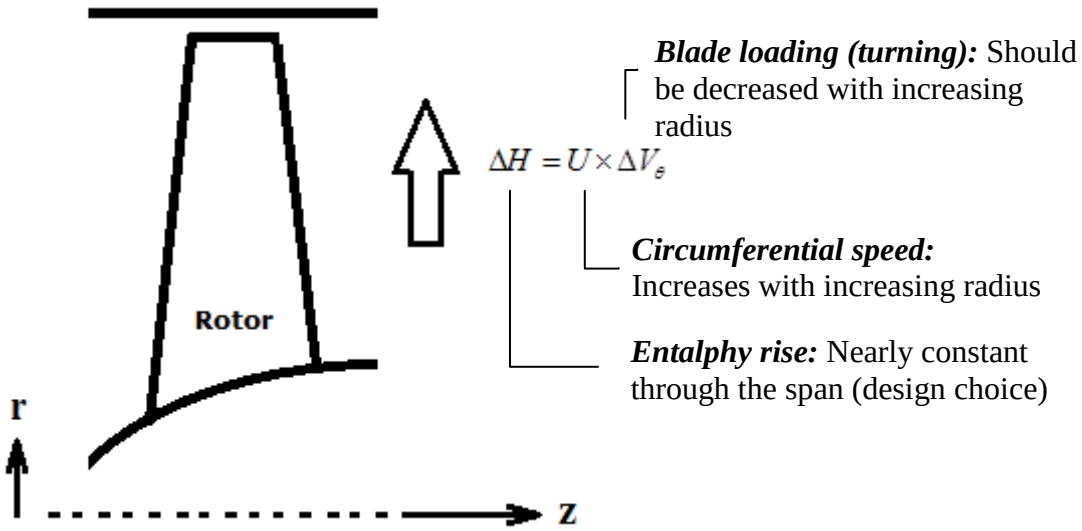


Figure 3.4. Blade loading distribution procedure

i. Blade sections with lower radius should have higher loading (i.e. high ΔV_θ) since circumferential speeds at this regions are smaller.

ii. Enthalpy rise at sections with higher blade radius is provided by high circumferential speed (i.e. high *U*), hence loading (i.e. turning) should be kept at low levels.

iii. Bigger twist factor further increases hub work in favor of tip up to a limit where losses become significant (boundary layer separation) due to excessively high hub loading.

These conditions can be satisfied by using low twist factor values, especially in the first stages operating at high Mach number values. Obviously, performing such an application manually will lead the designer towards the desired design, however, he/she will not be able to cover all possible design alternatives in the design space. Therefore, using automatic optimization techniques is a must for the sake of obtaining an optimum design.

3.1.2.2. Meridional Flowpath Parameterization

The meridional flowpath is the path which globally bounds the domain that the air flows through, see Figure 3.5. It is obtained by fitting a curve passing through several control points those of which represent the design variables that are to be modified in the search for optimum design.

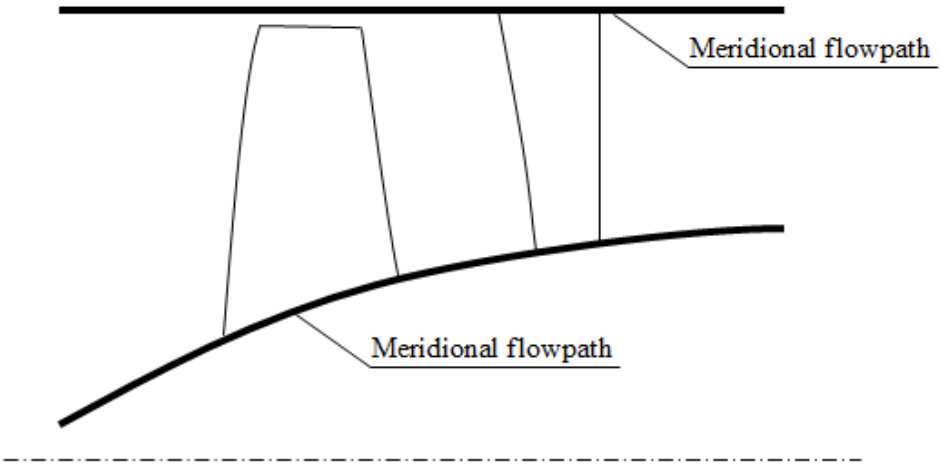


Figure 3.5. Representation of meridional flowpath

An important aspect in flowpath design is that, the resulting geometry should be continuous and its smoothness has to be ensured, since boundary layer transition and shock wave existence are strongly dependent on any discontinuity at the wall boundaries (Pierret, 1999), (Van den Braembussche, et al., 2004). This can be provided in case the first and second order derivatives of the curves are zero.

Using fundamental curve fitting techniques such as analytical polynomials or cubic interpolation, the curves become so noisy (Pierret, 1999), (Van den Braembussche, et al., 2004), (Verstraete, 2012). As the order of the polynomial increases slightly, the probability of obtaining a wiggling geometry increases, therefore, it becomes a problematic to generate new flowpath geometries by perturbing the ones at hand. Finally, problems should also be taken into account like obtaining curves with first order and second order derivative values of non-zero values.

On the other hand, Bézier curves are robust and secure in terms of the problems experienced in other curve fitting methods. The definition of Bézier curves is relatively simple. The resulting curve may show a wiggling characteristic as well; however the probability of such a situation occurs rarely (Al Salihi, 2010). Use of Bézier curves are common in turbomachinery design, where the method is used as a state of the art approximation in many studies (Giannakoglou, 1999), (Pierret, 1999), (Mengitsu & Ghaly, 2008), (Al Salihi, 2010), (Okui, et al., 2011), (Öksüz, 2011).

A final remark should be addressed that computational time required for the optimization is strongly dependent on the number of design variables; therefore, the parameterization approach has to be selected so as to provide minimum number of design

variables, whereas, too few variables may result in loss of important information: a trade-off has to be made (Pierret, 1999). In addition to aforementioned benefits of utilizing Bézier curve approach, it also provides a curve fitting procedure with less but necessarily enough control points.

3.1.3. Construction of Bézier Curves

Given $n+1$ points, $P_0, P_1, P_2, \ldots, P_n$, a Bézier curve can be formulated like (Al Salihi, 2010),

$$C(u) = \sum_{i=0}^{n} B_{n,i}(u) P_i \tag{3.7}$$

$B_{n,i}(u)$ is a Bernstein polynomial with $u \in [0,1]$,

$$B_{n,i}(u) = \frac{n!}{i!\,(n-i)!} u^i (1-u)^{n-i} \tag{3.8}$$

where n is the degree of the Bézier curve. The constructed curve always passes from the first and $(n+1)$-th point and always tangent to P_0 , P_1 and P_{n+1} , P_n at the end points. An example curve constructed by means of the method is provided in Figure 3.6.

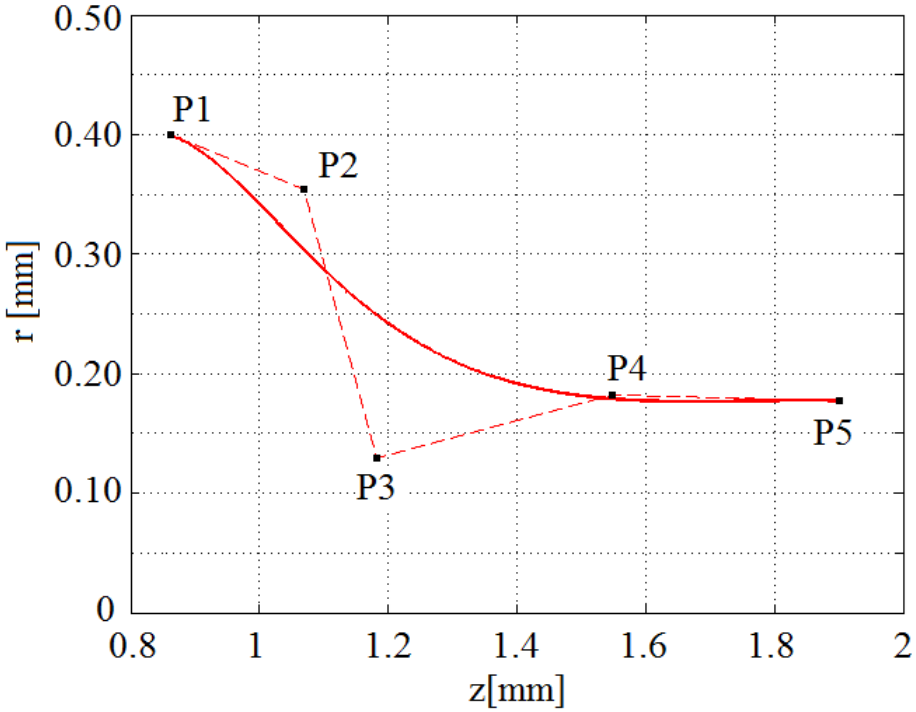


Figure 3.6. A curve constructed by Bézier Curve Method

CHAPTER 4

IN-HOUSE OPTIMIZATION PROGRAM DEVELOPMENT

This thesis aims to provide an optimization program to find an optimum turbomachinery geometry respecting the given constraints. For this purpose, an in-house genetic algorithm program is coded and it is hybridized with the gradient optimization tool of MATLAB© which utilizes penalty function methods. Algorithm and code structure can be found in Figure 4.1 and Figure 4.2.

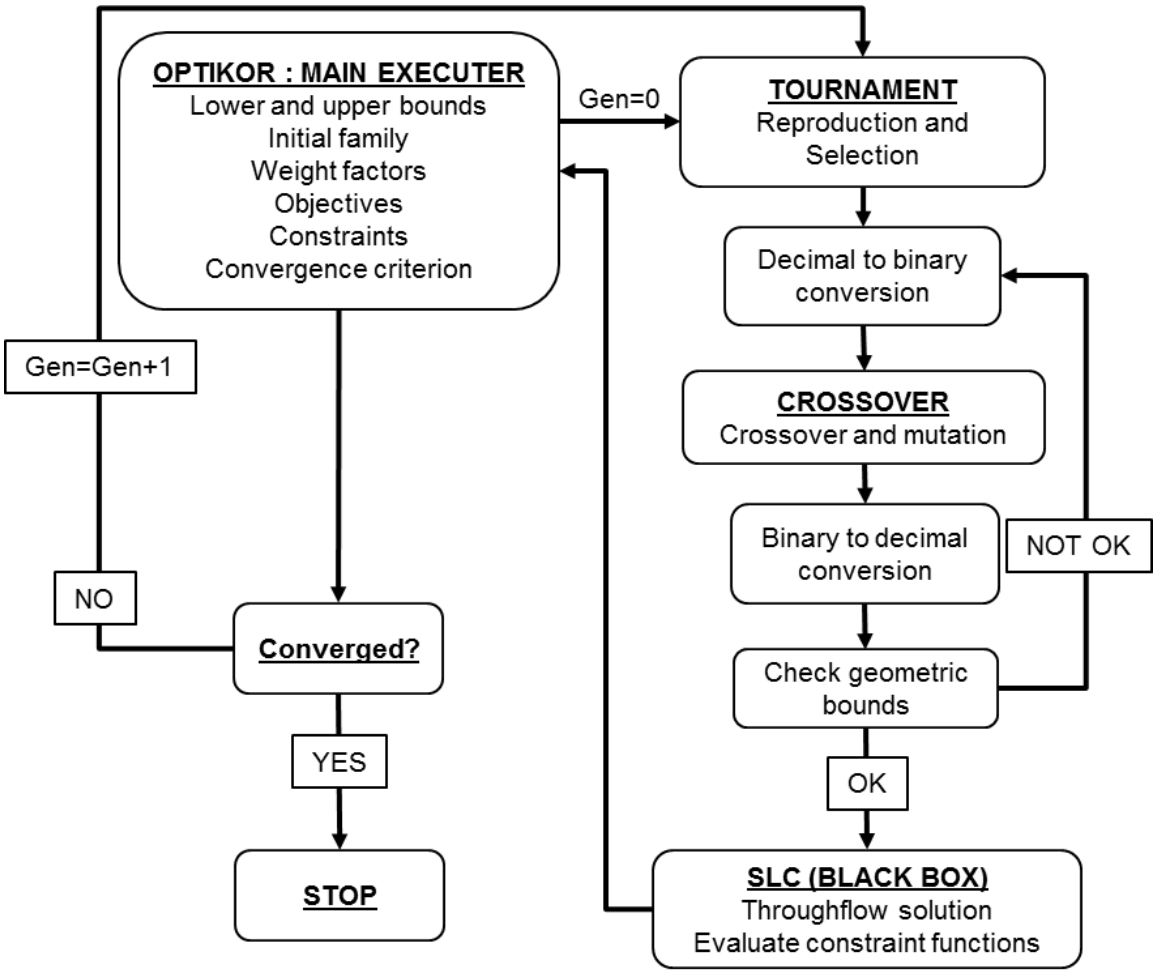


Figure 4.1. Algorithm and code structure for genetic algorithm

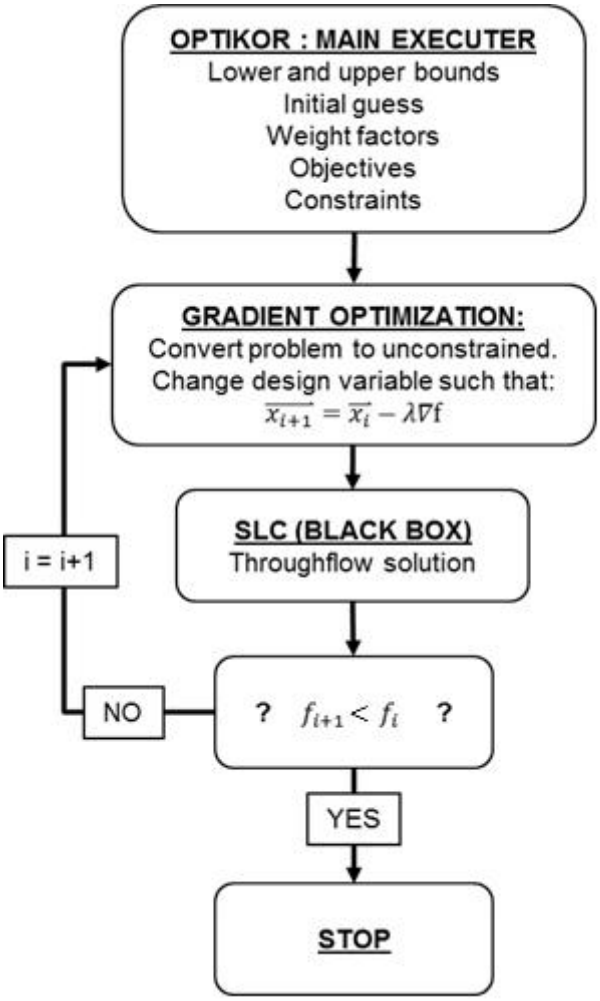


Figure 4.2. Algorithm and code structure for gradient optimizer

As seen in Figure 4.1 and Figure 4.2, the in-house code employs a streamline curvature (SLC) solver for function evaluation. This solver is developed in the scope of doctorate thesis of Sercan Acarer. It is validated not only by CFD calculations, but also by experimental data available in the literature for fans and compressors (Acarer, 2015), (Acarer & Özkol, 2015), (Acarer & Özkol, 2015). The program works respecting the physical principles introduced in Chapter 3.

4.1. Validation of the In-House Program

Prior to its use for throughflow optimization, the in-house program is first tested on an analytical test function (Equation (4.1)) available in the literature (Deb, 1999).

$$f(r) = 1 - e^{4r} sin^4(5\pi r) \tag{4.1}$$

In Figure 4.3, several discrete runs of genetic algorithm (GA-1, GA-2, …) , hybridized code in which genetic algorithm and gradient optimizer work together, and exact analytical solution are plotted. It is seen that GA is capable of finding the global minimum neighborhood, but provides a weak performance on finding the exact minimum. On the other hand, hybridized solution can find almost-exact minimum with a proximity of $\boldsymbol{r_{exact} - r_{hybrid}}$ = **0.000047**. In conclusion, the author suggests that the in-house program is capable of finding the design variable that minimizes the objective function, hence providing the optimum.

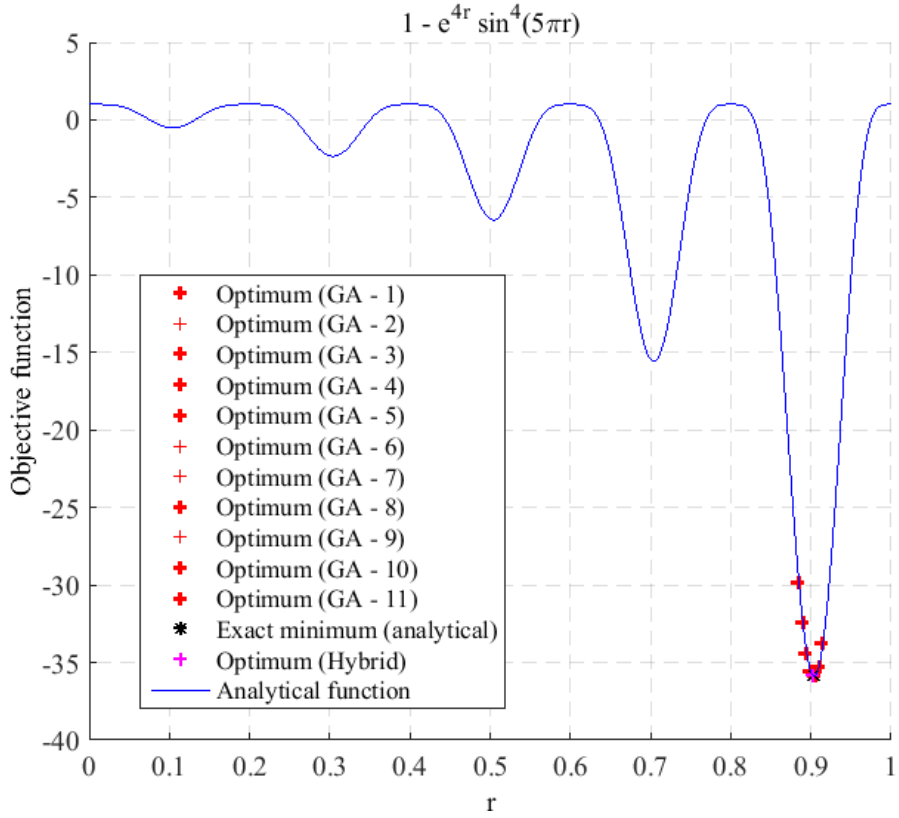


Figure 4.3. Validation of the code

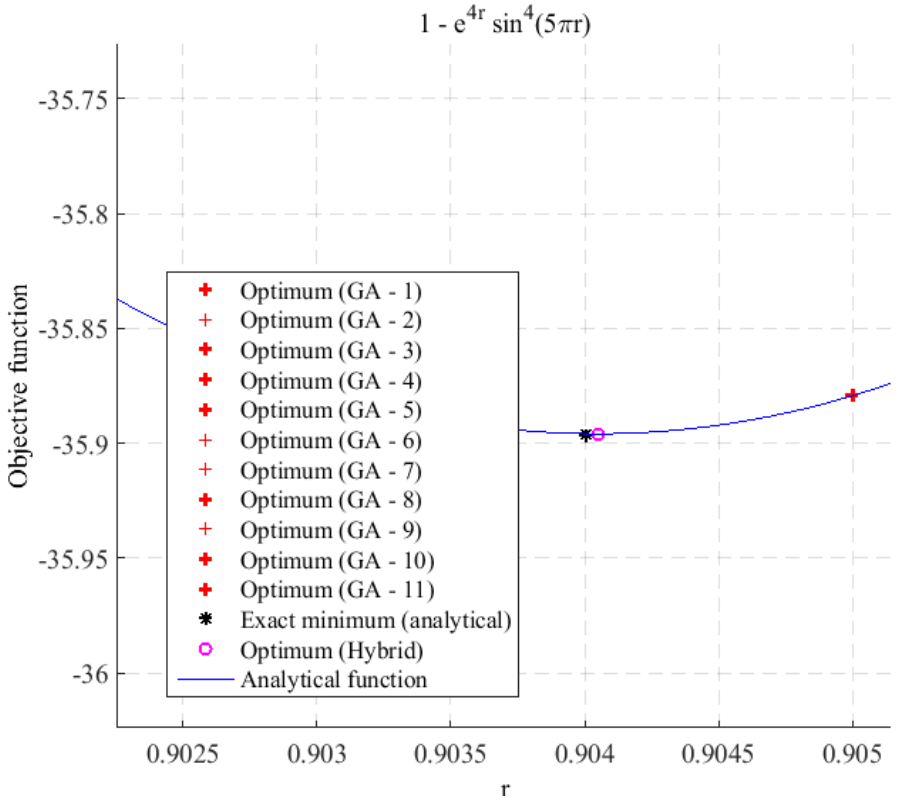


Figure 4.4 Validation of the code (Zoomed view)

4.2. Implementation of the In-house Program on a Real Life Problem

Prior to its use on throughflow optimization, the in-house optimization program is applied on a comparatively simpler real-life problem. The problem chosen for this purpose is from turbomachinery world; where it is aimed to obtain an optimum ***vaneless return channel*** geometry that connects two successive radial compressor impellers in a turboshaft engine, see Figure 4.5.

Return channel geometry is assumed as axis-symmetric due to the circumferential uniformity of the profile sketched in Figure 4.5. Optimization problem is defined with 29 design variables, all of which are geometric positions of the control points, see Figure 4.6. The geometry is constructed in ANSYS Design Modeler. Lines forming the U-bend and L-bend regions are formed by the program's spline generator, whereas the vaneless channel connecting these bends is constructed with straight lines. Since the splines and the line forming the vaneless channel are tangent to each other at connection points, the continuity of the profiles is guaranteed.

At the inlet boundary, tangential component of the velocity vector is not taken into account in order to have a simplified problem. Any flow quantity imposed at the boundaries is averaged spatially. The problem is assumed steady since the flow is coming from a non-rotating part (i.e. wedge diffuser). 3D effects like secondary flows, tangential components, etc. are not taken into account, whereas an optimization task that utilizes 2D Navier Stokes calculations will be reported in the scope of this thesis. It is obvious that, the simplifications assumed in this stage do not represent the real physics of the problem, however an optimum design obtained by 2D optimization would be a good starting point for a 3D optimization case that aims fine tuning the geometry based on best 2D design, as it is a frequently used method in optimization literature (Pierret, 1999), (Verstraete, 2012).

The objective function to be minimized is total pressure drop throughout the channel (i.e. total pressure difference between *inlet* and *outlet* stations). For numerical stability of the CFD computations, inlet and outlet stations are extended; hence the computations are performed using the model with dummy inlet and outlet regions.

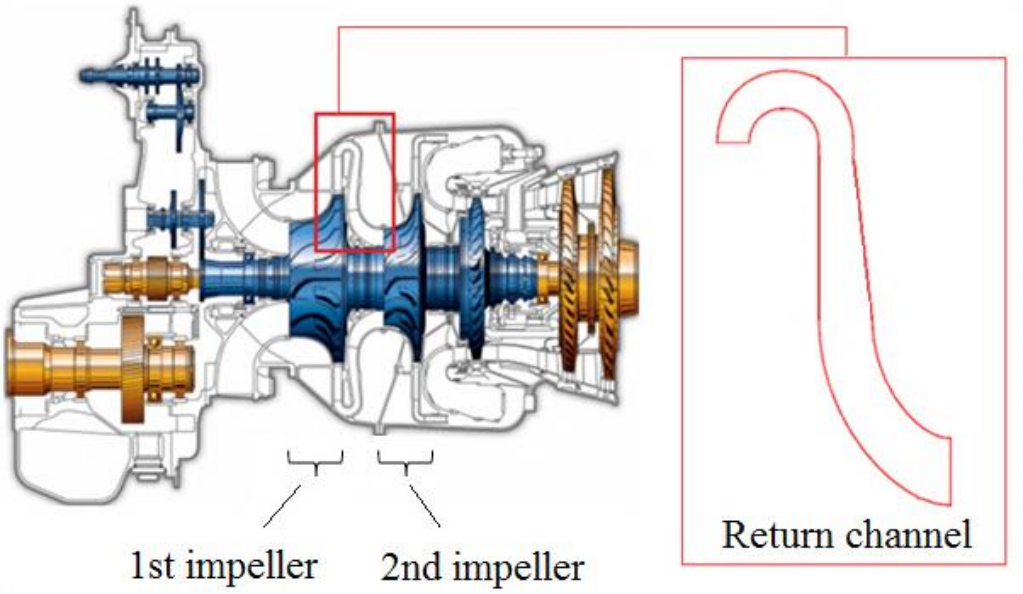


Figure 4.5. Representative view of return channel (MTU Turbomeca RR - MTR 390)

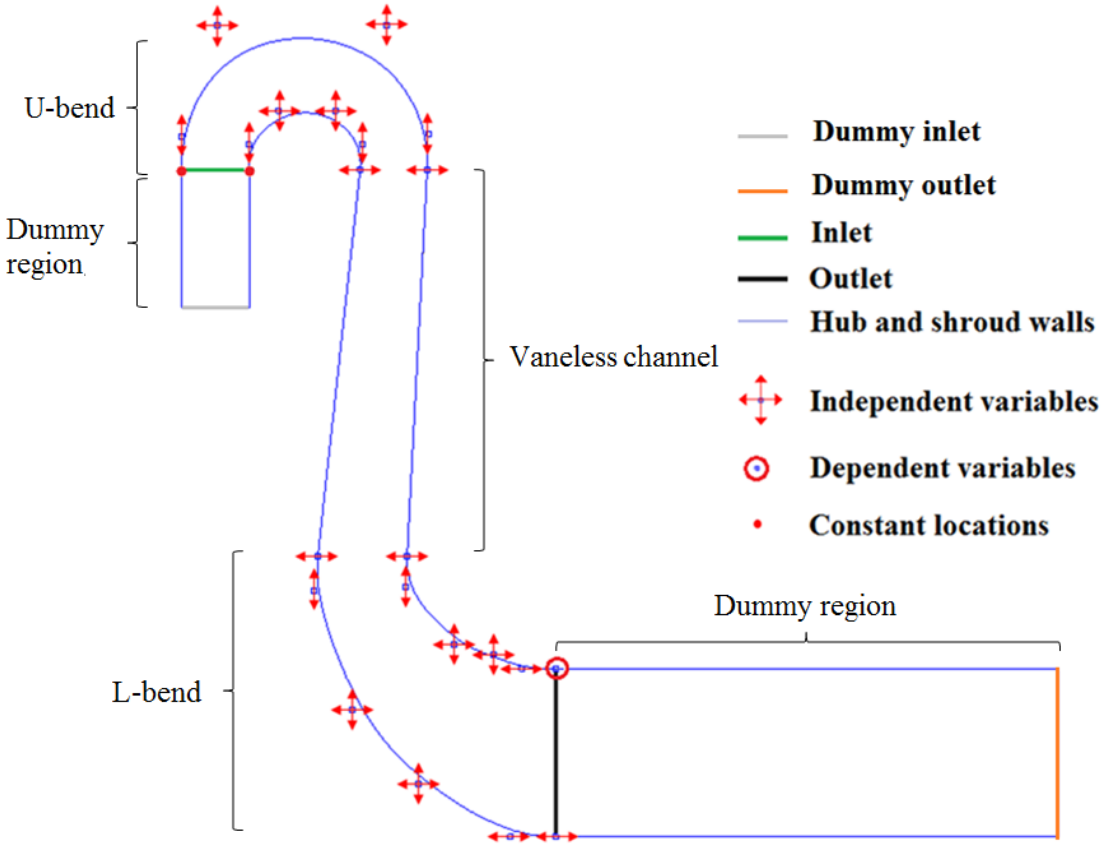


Figure 4.6. Baseline vaneless return channel geometry

4.2.1. Definition and Solution of the Problem in ANSYS Workbench

The definition and solution of the problem necessitates the use of a commercial program that is capable of building the problematic geometry, meshing and performing CFD calculations. For this purpose, ANSYS Design Modeler, Mesh and Fluent programs are utilized, where the parametric solver property of ANSYS Workbench program makes it possible to parameterize the problem. In order to proceed the process hands-off, the batch mode of the program is benefited. Once the program configurations are set up, the genetic algorithm that is coded in the scope of this work is coupled with ANSYS and optimization process is performed.

4.2.1.1. Mesh and Solver Configurations

Fully structured mesh of the return channel geometry is built by means of ANSYS Mesh 15.0, with an element number of 9230, where boundary layer refinement is provided in the near wall regions, see Figure 4.7. Maximum ***y+*** value is 5 for the reference geometry, Figure 4.7.

Solver configuration is set to work in axis-symmetric mode. Ideal gas (air) is selected as the working fluid.

Compressibility and temperature effects on flow properties are taken into account:

- "Sutherland correlation with three coefficients" option is selected for viscosity calculations.
- Specific heat capacity is estimated using piecewise polynomial.
- Kinetic theory is utilized for calculating thermal conductivity correctly.

Spalart-Allmaras method is chosen as the turbulence model where viscous heating is taken into account. Pressure-Velocity coupling scheme is selected to be as "Coupled", and any other discretization schemes are selected to be as second order.

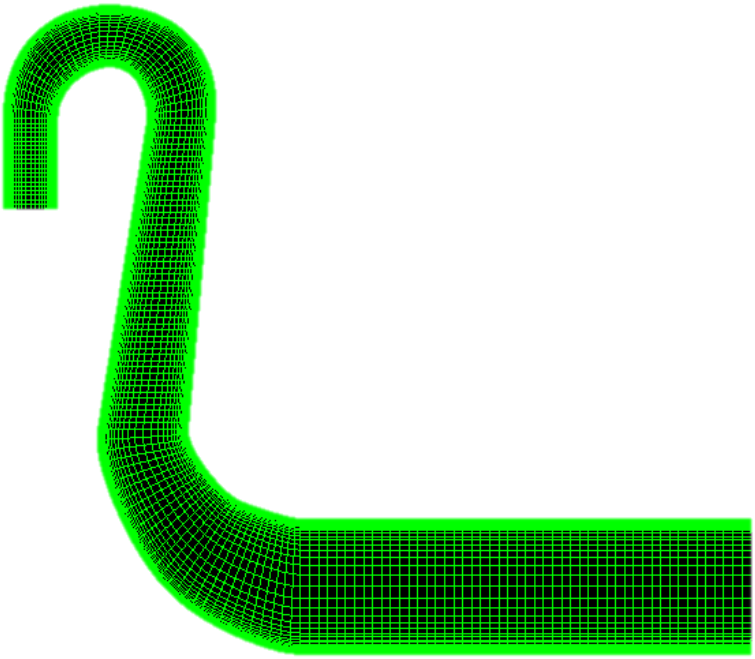


Figure 4.7. Return channel mesh

4.2.1.2. Results of Vaneless Return Channel Optimization

Geometric parameters are constrained such that any variable should be in the range of ±3 mm of the baseline design geometric locations, where any individual violating this rule is rejected. Based on experience, an initial population with 58 members (parameter number x 2) is randomly generated, where all members of it respects the geometric constraints. The mutation rate is decided to be 1%. Tournament method is picked as the reproduction scheme.

At the end of 11 iterations (i.e. genetic algorithm is run 11 times,), optimization process is stopped and the individual providing minimum total pressure drop is obtained, leading to an improvement in objective function in the order of ~15%. Initial and final geometries are provided in Figure 4.8 where total pressure contours are included in the illustrations.

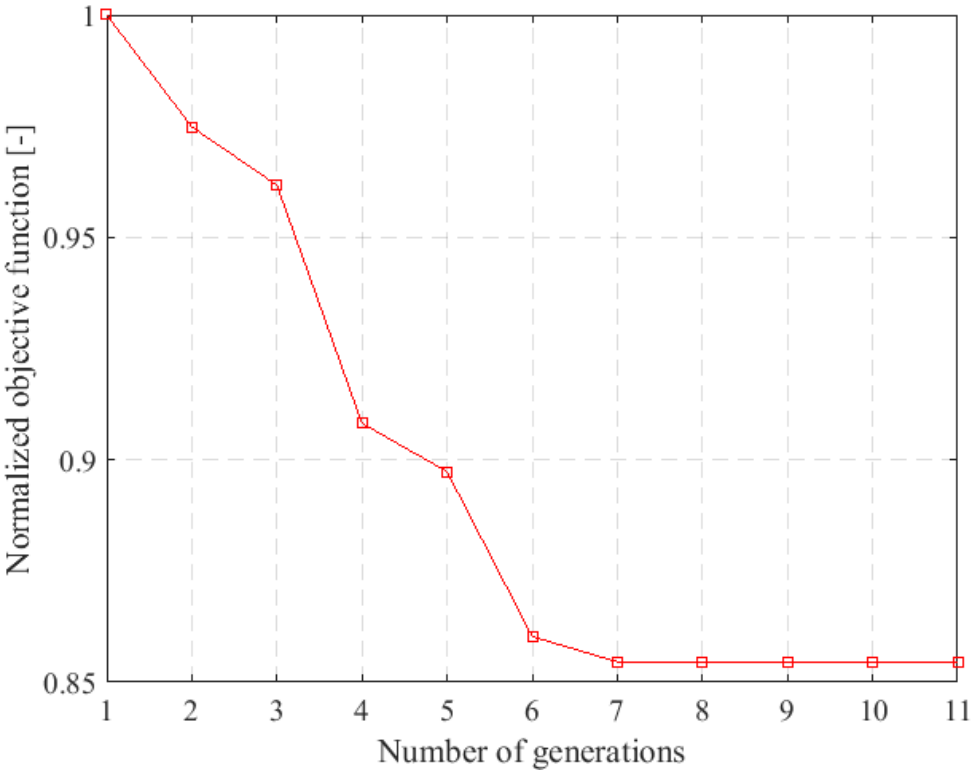


Figure 4.8. Normalized objective function for successive generations

The percentage improvement of normalized objective function seen in Figure 4.8 can also be observed from Figure 4.9 that in the region following the L-bend, the total pressure loss observed in the initial best geometry is eliminated in the optimum solution. This is thought to be due to the widening of the passage between the u-turn and the L-bend, which results in a decrease in Mach number, therefore the curvature effect of the geometry of the flow becomes less problematic.

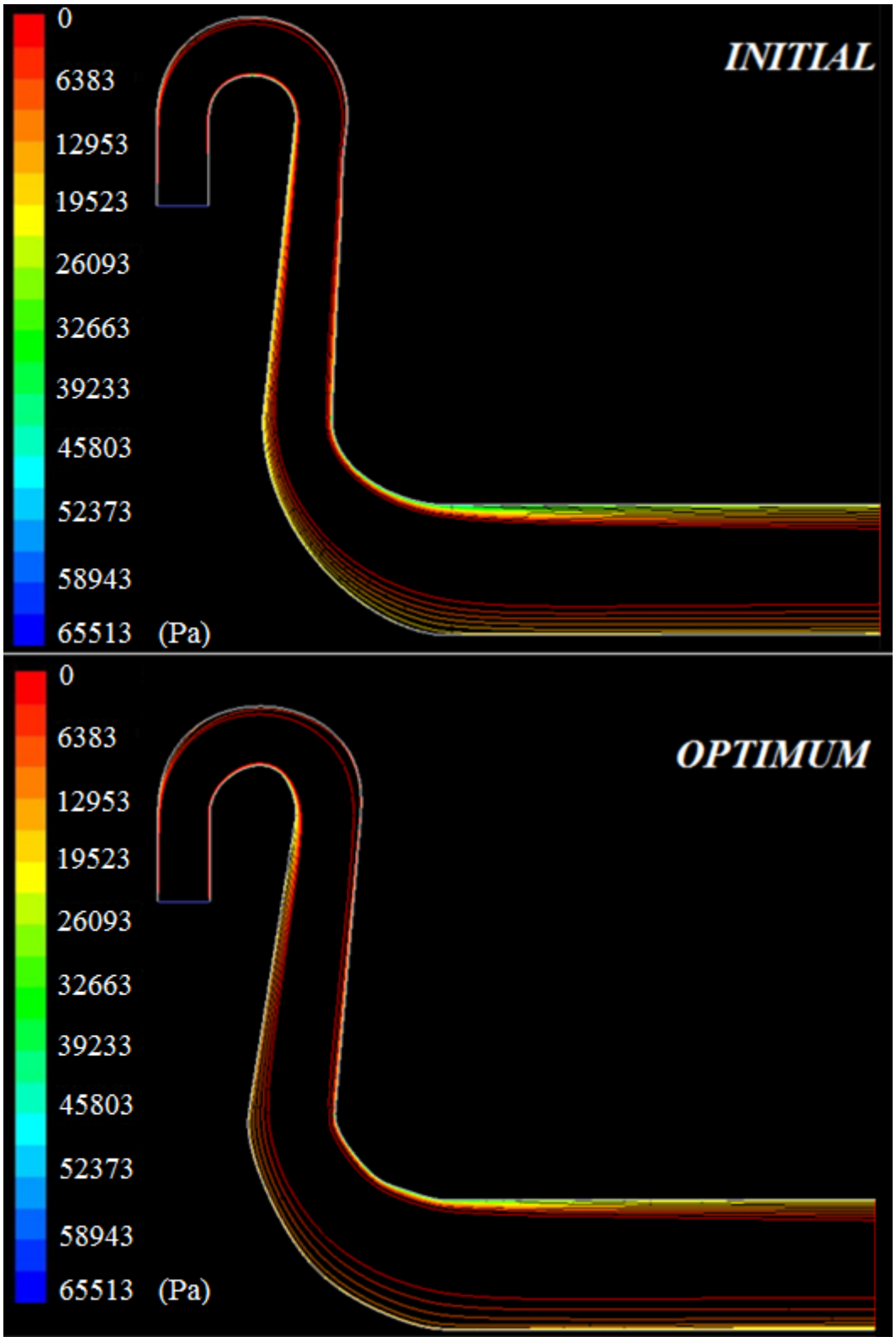


Figure 4.9. Contours of total pressure drop: initial and optimum geometries

CHAPTER 5

THROUGHFLOW OPTIMIZATION USING GA

5.1. Generic Single Objective Throughflow Optimization Using GA

Among infinite number of possible variables, objective functions and constraints, followings which are considered to have upmost importance are selected and investigated in the scope of this study.

5.1.1. Variables

Blade exit angle distribution: The work distribution should be adjusted to be as uniform as possible in spanwise direction, considering the suggestions listed in Chapter 3.1.2.1. Therefore, uniform radial total pressure distribution behind last stator stage should be imposed as a constraint, providing an equally distributed work in the radial direction.

Meridional Chord: This quantity directly controls the solidity (σ = chord / pitch), of the blades, see Figure 3.3. It should be noted that low values for solidity will cause lower efficiency since the flow will not be well guided and the deviation will be too much. On the other hand, once the solidity is chosen so high, losses due to friction will be too much, resulting in attenuation in efficiency. Therefore, the designer should properly adjust his/her meridional chord value such that optimum efficiency is obtained.

Flow path: The meridional flow path is an input for throughflow solution, which is initially defined in 1D meanline design. However, the initial guess is based on 1D considerations and it must be optimized in a multi-dimensional sense for the sake of obtaining optimum acceleration/deceleration characteristics.

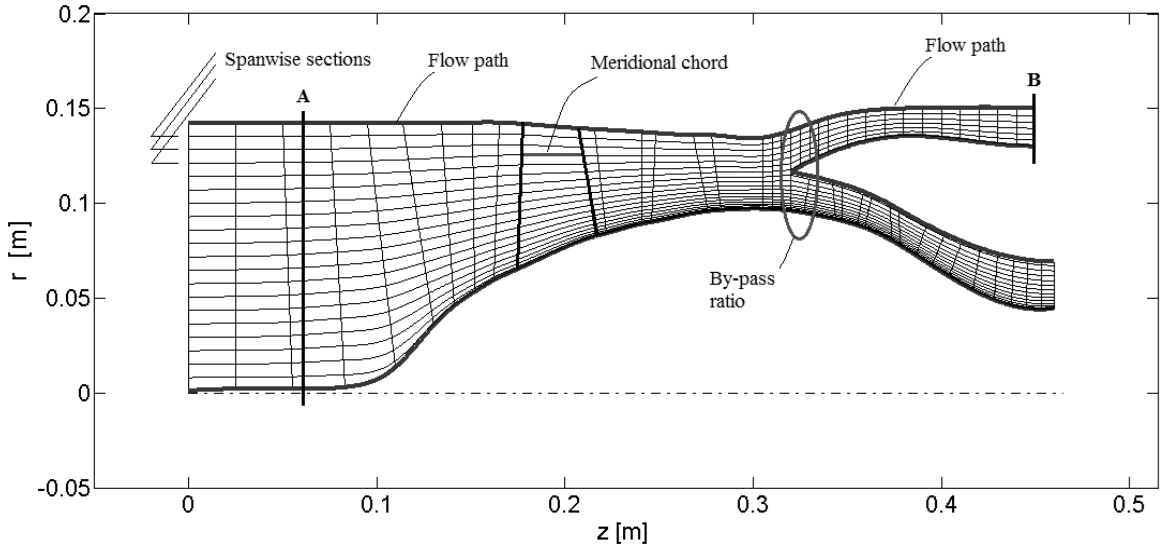


Figure 5.1. Meridional view of a low by-pass military turbofan engine
(Source: Acarer, 2015)

5.1.2. Objective Function and Constraints

As mentioned before, objective functions are the values that are to be optimized. For this study, it is proposed to set these values as,

- Deviation from total pressure average value at the latest section. The formulation is provided in Equations (5.1) and (5.2) where N is total number of spanwise sections, i is the station number (Figure 5.1), B is the fan by-pass outlet station, μ is the spanwise average of total pressure

$$Std(P^0)_B = \frac{1}{N-1}\sum_{i=1}^{N}\left|P^0_{B,i} - \mu\right|^2 \tag{5.1}$$

$$\mu = \frac{1}{N}\sum_{i=1}^{N} P^0_{B,i} \tag{5.2}$$

- Minimizing entropy generation, in other words, maximizing efficiency, Equations (5.3) and (5.4).

$$\Delta s_{AB} = s_B - s_A \tag{5.3}$$

$$\eta_{AB} = \frac{\Delta h_{ideal}|_{AB}}{\Delta h_{real}|_{AB}} \tag{5.4}$$

In this study, objective function value obtained at the end of each optimization task is normalized with initial family's fittest member's objective function value. All graphs presenting the improvement of objective function is plotted by this mean, since such information will directly provide objective function's relative improvement with reference to initial family's fittest (or best) member. Being f the objective function, normalized objective function, $\bar{f}$, can be formulized as given in Equation (5.5).

$$\bar{f} = \frac{f_{best,gen}}{f_{best,init}} \tag{5.5}$$

Another parameter that has high importance in fan/compressor design is diffusion factor (*DF*). Generally speaking, in 2D cascades, this quantity provides a measure for the amount of deceleration or diffusion of the surface velocity, which eventually determines the wake momentum thickness. In the literature, this quantity is reconstructed with the name "equivalent diffusion factor" (DF^*) which is formulized in Equation (5.6), where W_2 is station outlet relative velocity and W_{max} is the maximum velocity through the station.

$$DF^* = \frac{W_{max}}{W_2} \tag{5.6}$$

Note that the bigger the diffusion factor, the more fan/compressor pressurizes the fluid. In other words, the pressure rises in the direction that the fluid is trying to flow; therefore beyond $DF^* > 2$, abrupt increase in losses are observed (Aungier, 2003) since

the fluid will no more follow the blade but separate. Based on these considerations, diffusion factor has to be limited in automatic optimization studies and any design violating the constraint $DF^* > 2$ should be directly eliminated.

Smooth acceleration and/or deceleration at the endwalls play key role on turbomachinery design. A design with severe acceleration/deceleration will be vulnerable to separation at the endwalls. Therefore this phenomena is posed as a constraint to be obeyed in this study.

Total-to-total pressure ratio that the compressor/fan is demanded to produce is posed as an equality constraint. However, as mentioned before, the computational expense of respecting equality constraints is much higher than that of the inequality constraints. Therefore, this equality constraint is converted into an inequality constraint, such as $PR_{min} < PR < PR_{max}$.

Finally, the ratio of the mass flow rate by-passing the core and the mass flow rate passing through the core, namely bypass ratio, should meet the design constraints.

Besides those given above the optimization program should respect the geometric constraints such as maximum engine radius and engine length, maximum curvature at the endwalls.

5.1.3. GA Configurations

Number of initial family members is decided to be 2 times of the number of design variables, with reference to common practice in turbomachinery literature. The crossover operation is performed with four points among each chromosome. Mutation rate is taken as 1%. In addition, an analysis is also performed so as to understand the effect of mutation rate on genetic algorithm results. Tournament method is utilized in reproduction and selection process.

5.1.4. Optimization of the Model Using Cubic Interpolation Method

The in-house genetic algorithm program developed by the author is coupled with the in-house throughflow solver program developed by Acarer (Acarer, 2015), (Acarer & Özkol, 2015), (Acarer & Özkol, 2015) and a generic run is performed in order to

understand if two programs can work together accurately or not. Objective function, constraints and design variables for the generic throughflow optimization problem are provided in Equation (5.7) ~ (5.14), where the graphical illustration of the problem can be found in Figure 5.3.

In the first phase of the study, cubic interpolation method is used for curve-fitting the points of which the bypass region hub geometry is composed of. The reason is that the throughflow program was developed using cubic interpolation method, hence it required a certain time to convert the program such that it is capable of generating geometry using Bézier curves.

Minimize $$\Delta s_{AB} \tag{5.7}$$

Subject to $$DF^* < 2 \tag{5.8}$$

$$V_{hub} > 60m/s\,,\, V_{shroud} > 60m/s \tag{5.9}$$

$$R_{C,endwalls} < 1.2[\max(R_{C,endwalls})]_{initial} \tag{5.10}$$

$$1.5 < PR < 1.6 \tag{5.11}$$

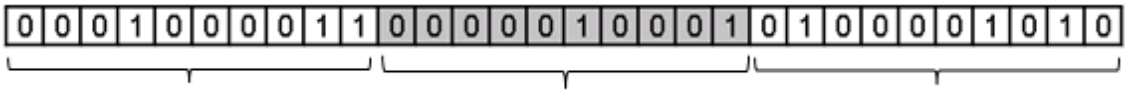
$$5.9 < BPR < 6.4 \tag{5.12}$$

$$min[(\vec{R}_1)_{initial}] < R_1 < max[(\vec{R}_1)_{initial}] \tag{5.13}$$
$$min[(\vec{R}_2)_{initial}] < R_2 < max[(\vec{R}_2)_{initial}]$$

Where $$\vec{x}^T = \{R_1, R_2, TF\} \tag{5.14}$$

The axis-symmetric geometry, obtained by using cubic spline interpolation method, is seen in Figure 5.3, where splitter hub curve is parameterized by 4 points highlighted by red-spots. Note that, only the points named as R1 and R2 are perturbed in optimization process, whereas others are kept unchanged. Twist factor, TF, in V_θ distribution formulation (Equation (3.6)) is selected as another design variable. To summarize, the optimization study is carried on by perturbing 3 design variables in total. The binary string representing the design variables are provided in Figure 5.2. Each variable is multiplied by a certain value, in order to make it possible to represent floating numbers without loss of information that may occur due to any possible rounding. After

the crossover operation, new design variables are obtained by dividing the resulting values with the multiplication value, i.e. 1000 for the case exemplified in Figure 5.2, mentioned in the previous sentence. Mutation rate is taken as 1%, which is reported to be best in the literature (Konak, et al., 2006), meaning that 1 genes over 100 are mutated for the case under investigation.



$[R1 - R1_{LB}] \times 1000 = 67$ **$[R2 - R2_{LB}] \times 1000 = 18$** **$[TF - TF_{LB}] \times 1000 = 266$**

Figure 5.2. Representation of the design variables with binary numbers

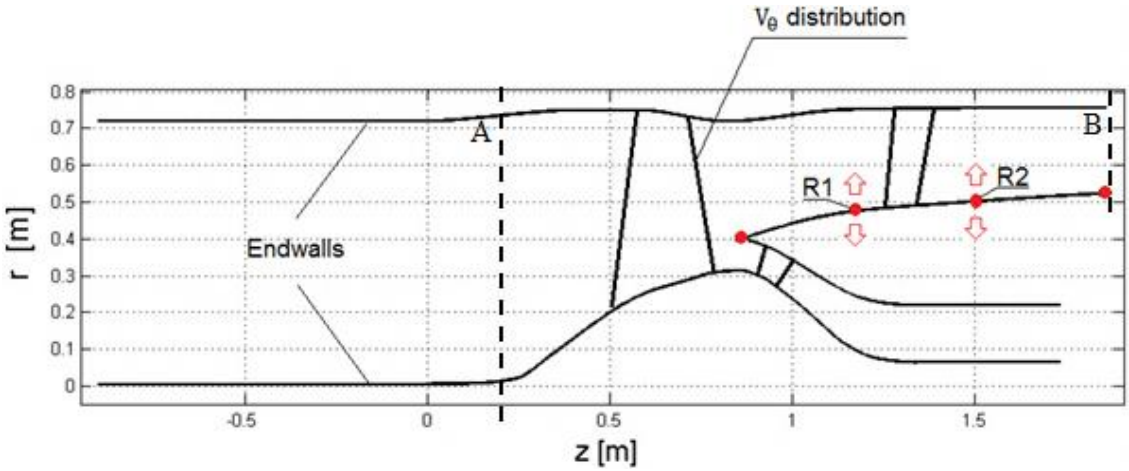


Figure 5.3. Graphical illustration of the optimization problem and design variables

As noted earlier, genetic algorithm needs an initial population that consists of designs which respect the constraints given in Equation (5.8). For the generic optimization problem, an initial population with 10 different designs (individuals) is provided, Figure 5.4.

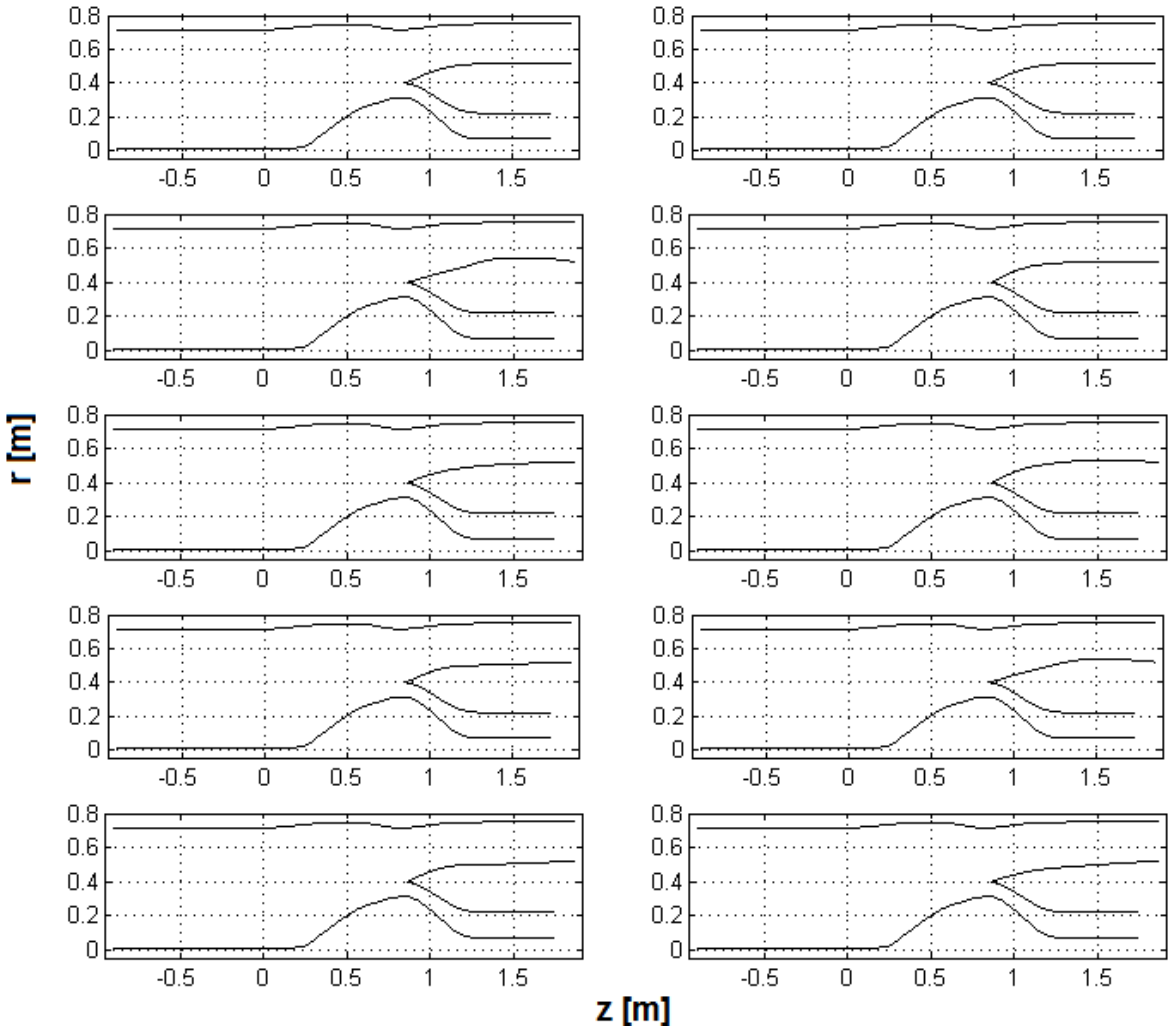


Figure 5.4. Initial family for the optimization problem

5.1.4.1. Results of Throughflow Optimization

The genetic algorithm converged to the optimum solution after 10 generations, where computational time is 15.6 hours. Different designs are plotted together in Figure 5.5, where the bold red geometry represents the optimum design point providing minimum entropy generation. The objective function values obtained in successive iterations are plotted in Figure 5.6.

For this unique case, the genetic algorithm resulted in design variable vectors (radius values) and objective function values (entropy generation) as tabulated in Table 5.1.

As mentioned earlier, genetic algorithm finds not the exact global optimum, but it provides the almost-global optimum point. In case one is seeking for the exact optimum, it is necessary to employ gradient based methods. Therefore, gradient based optimization is performed, where the almost global optimum point obtained via genetic algorithm is posed as the initial design. It is seen that a slight decrease in objective function is observed, see Table 5.2. The computational time is 2382 seconds. It is seen that 0.4 % improvement in terms of objective function is observed with reference to the initial best values.

Table 5.1. Design variables providing optimum geometry

R1	0.4670 m
R2	0.4670 m
TF	0.9660
LOSS	18.908 kj / kg.K

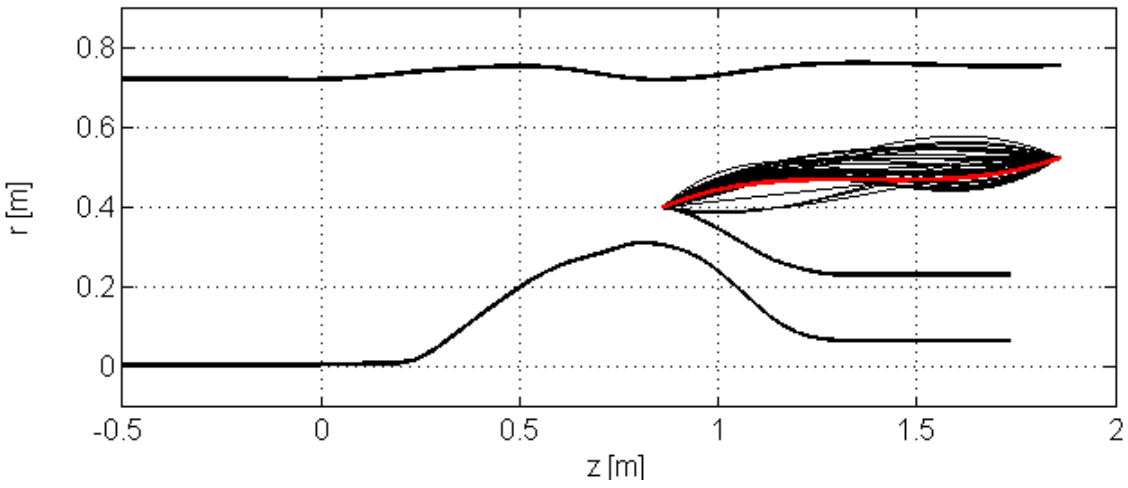


Figure 5.5. Geometries obtained in optimization: optimum geometry → (red curve)

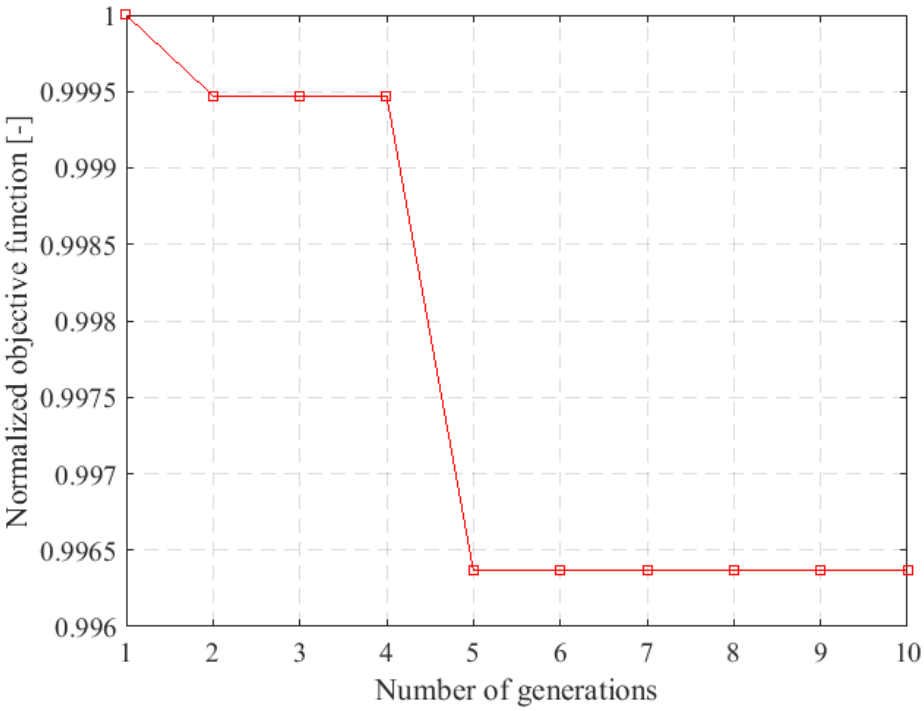


Figure 5.6. Normalized objective function for successive generations

The constraint values are tabulated in Table 5.3 for almost-global and exact global optimum design cases and it is seen that all constraints are respected in the final geometry.

Table 5.2. Throughflow optimization: Design variables

	Almost-global optimum (GA alone)	**Global optimum (GA and Grad. Coupled)**
R1	0.4670 m	0.4553 m
R2	0.4670 m	0.4602 m
TF	0.9660	0.9138
LOSS	18.908 kj / kg.K	18.89 kj / kg.K
% improvement	% 0.11 (with ref. to initial best: 18.911)	

Table 5.3. Throughflow optimization: Constraint values

	GA alone	Hybrid	Constraint limits
DF	1.5731	1.7325	< 2
$\mathbf{V_{hub}}$ $\mathbf{V_{shroud}}$	134.1 m/s	124.41 m/s	> 60 m/s
$\mathbf{R_{c,\ endwalls}}$	0.737 m	1.058 m	< 1.755 m
PR	1.572	1.579	$1.5 < PR < 1.6$
BPR	5.62	5.59	$5.6 < BPR < 6.4$

5.1.5. Optimization of the Model Using Bézier 5 Control Points

As noted earlier, Bézier curves are reported to give good performance in turbomachinery optimization. However, preliminary studies are performed using cubic interpolation method, since the throughflow program is coded in advance by this mean. Presently, the code is modified such that it is now possible to use Bézier curves for geometry construction, which makes it convenient to increase the number of design variables, since the resulting geometry becomes rigid against wiggling problem observed in other curve fitting techniques. The axis-symmetric geometry, obtained by using Bézier curves is seen in Figure 5.7, where splitter hub and tip curves are parameterized by 5 points highlighted by red solid spots. Blade angle distribution is parameterized by using ***twist factor***, symbolized by TF.

Objective function, constraints and design variables for the generic throughflow optimization problem are provided in Equation (5.15) ~ (5.23). The problem is now multi-objective, as the target is to minimize entropy generation (Δs) and minimize normalized total pressure distribution non-uniformity (or standard deviation of normalized total pressure distribution - i.e. $\overline{std(P^0)_B} = (std(P^0)_B)/2000$) at station B, simultaneously. The second objective is normalized with 2000 that both objectives will be in the same order of magnitude. For the present problem, each objective is weighted with the factor 0.5, meaning that each objective has the same importance level.

Minimize $$0.5 \times \Delta s_{AB} + 0.5 \times \overline{std(P^0)_B} \quad (5.15)$$

Subject to $$DF^* < 2 \quad (5.16)$$

$$V_{hub} > 60m/s \, , V_{shroud} > 60m/s \quad (5.17)$$

$$R_{C,endwalls} < 1.2[\max(R_{C,endwalls})]_{initial} \quad (5.18)$$

$$1.5 < PR < 1.62 \quad (5.19)$$

$$5.5 < BPR < 6.4 \quad (5.20)$$

$$\left.\frac{\partial V_m}{\partial m}\right]_{bypass} < 1.5\left[max\left(\left.\frac{\partial V_m}{\partial m}\right]_{bypass}\right)\right]_{initial} \quad (5.21)$$

$$0.40 < R_1 < 0.65 \; ; \; 0.45 < R_2 < 0.65$$
$$0.50 < R_3 < 0.65 \; ; \; 0.65 < R_4 < 0.85 \quad (5.22)$$
$$0.65 < R_5 < 0.85 \; ; \; 0.7 < TF < 1.00$$

Where $$\vec{x}^T = \{R_1, R_2, R_3, R_4, R_5, TF\} \quad (5.23)$$

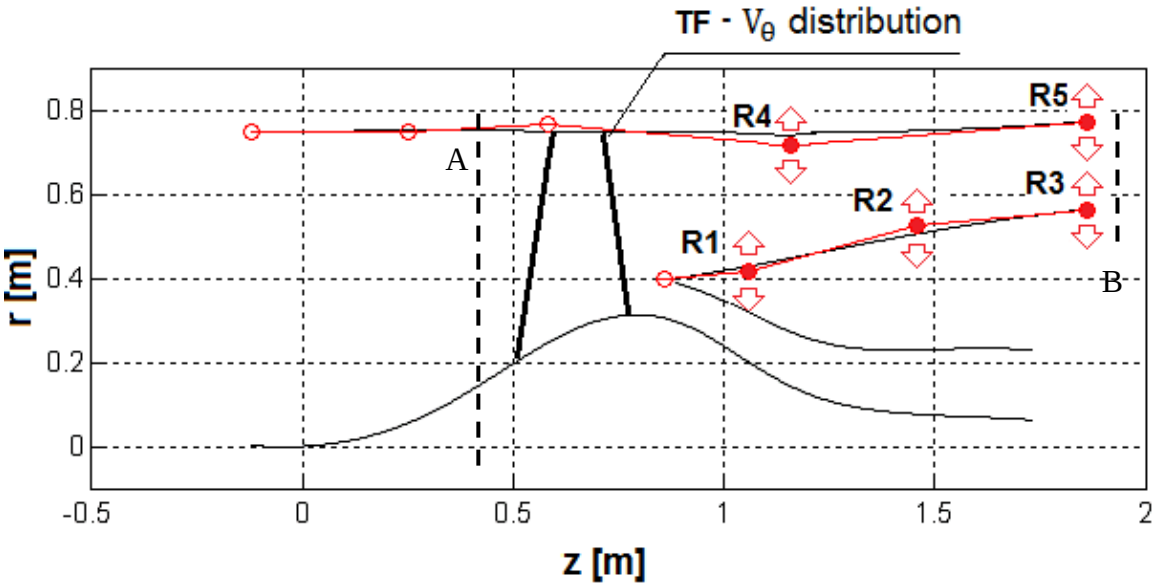


Figure 5.7. Graphical illustration of the optimization problem and the design variables

As noted earlier, genetic algorithm needs an initial population that consists of designs which respect the constraints given in Equation (5.16). For the optimization problem, an initial population with 10 different designs (individuals) is provided, Figure 5.8.

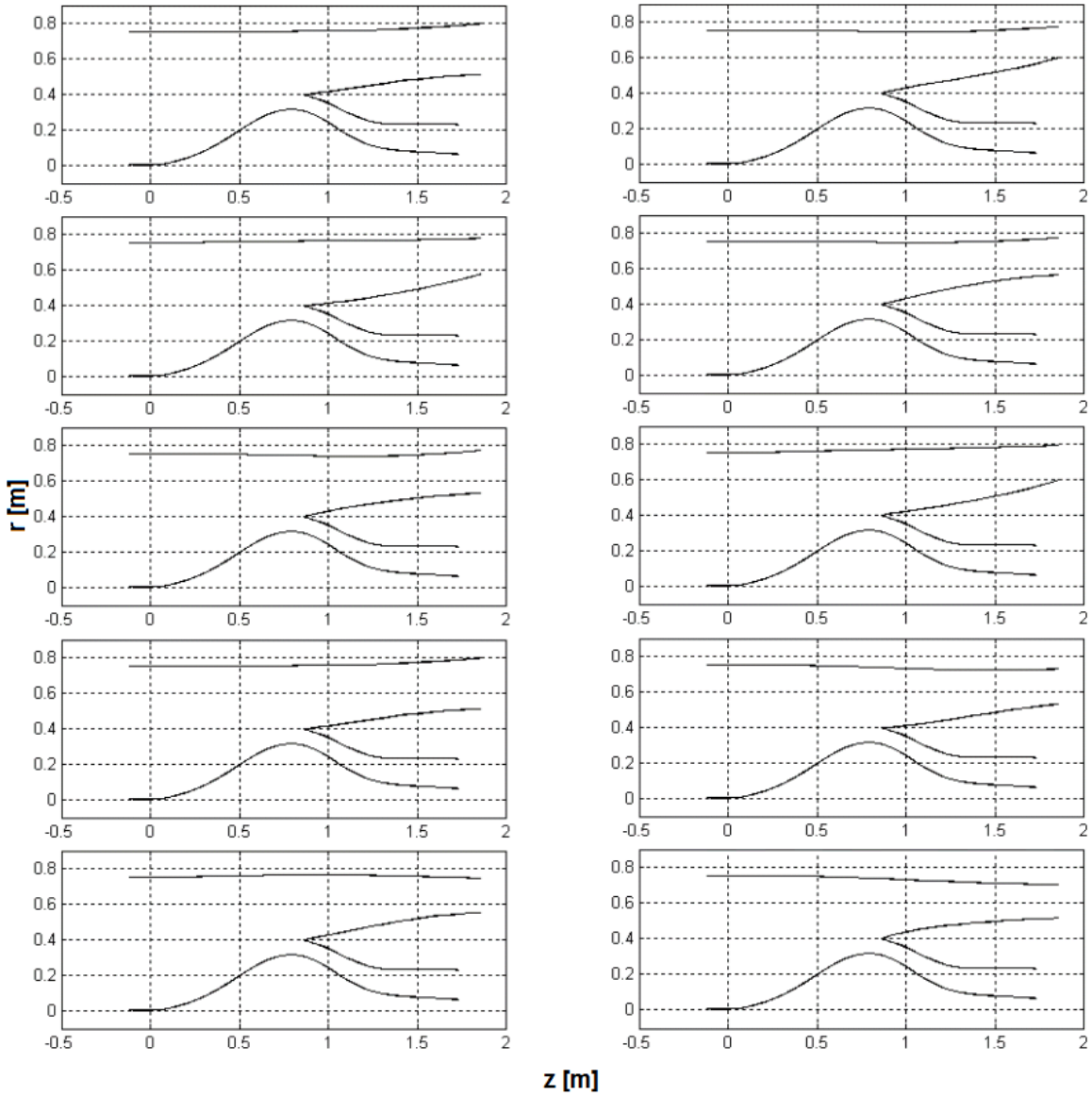


Figure 5.8. Initial population for multi objective optimization problem

5.1.5.1. Results

The computational time is needed for the throughflow optimization to converge is 15.55 hours. The reason for the higher computational time with respect to the former case is the higher number of design variables. Contrary to the previous case, different values of design variables for successive iterations are not provided since the figure becomes so incomprehensible. Instead, the band that the variables are distributed are plotted, see

Figure 5.9. The objective function values obtained in successive iterations are plotted in Figure 5.10. The optimum objective function value is obtained in 9th iteration.

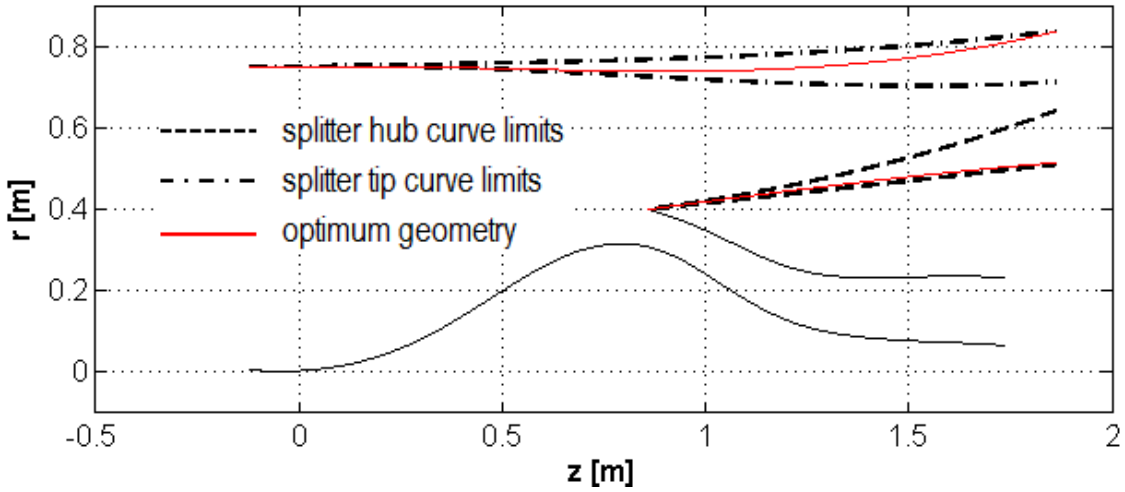


Figure 5.9. Design geometries obtained in optimization

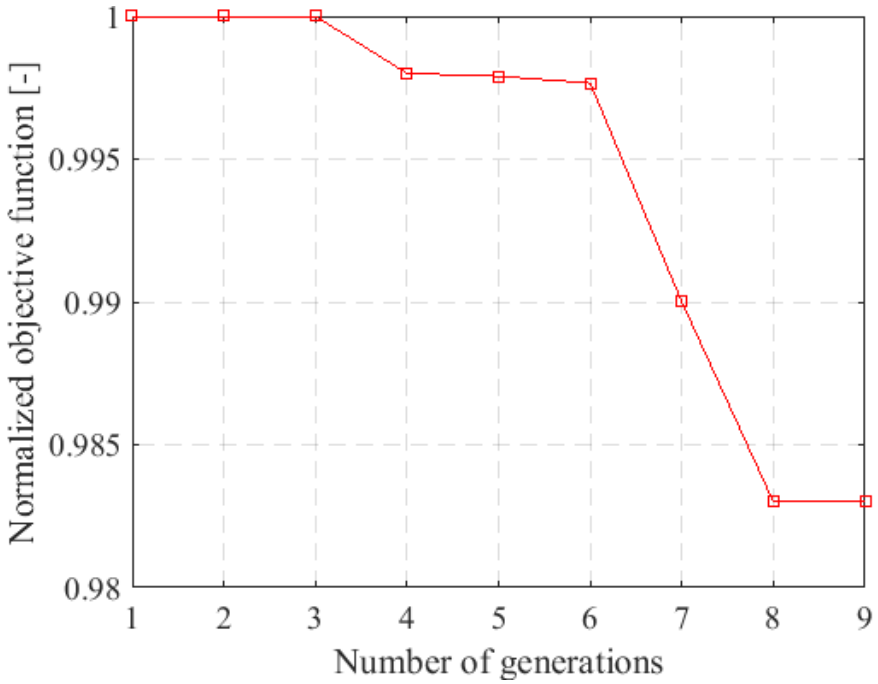


Figure 5.10. Normalized objective function for successive generations

For this unique case, the genetic algorithm resulted in design variable vector and objective function value (entropy generation) as tabulated in Table 5.4. It is seen that 1.73 % improvement in terms of objective function is observed with reference to the initial best values. The significant rise in improvement with reference to that of the preliminary studies is obviously due to higher number of design variables used in the latter case, which makes it possible to perform a search in a wider and diverse space.

Table 5.4. Throughflow optimization: Objective functions

LOSS	18.803 kj / kg.K
Std(P^o)$_B$	1319 kPa
$\boldsymbol{f(\vec{x})}$	0.7978
Improvement	1.73 %

The constraint values are tabulated in Table 5.5 and it is seen that all constraints are respected in the final geometry.

Table 5.5. Throughflow optimization: Constraint values

	GA	**Constraint limits**
DF	1.778	< 2
V_{hub} , V_{shroud}	130.55 m/s	> 60 m/s
$R_{c,\ endwalls}$	0.960 m	< 1.755 m
$\boldsymbol{\left.\frac{\partial V_m}{\partial m}\right]_{bypass}}$	457.25 1/s	$< 1.5 \left[max\left(\left.\frac{\partial V_m}{\partial m}\right]_{bypass}\right)\right]_{initial}$
PR	1.612	$1.5 < PR < 1.62$
BPR	6.13	$5.5 < BPR < 6.4$

5.1.6. Optimization of the Model Using Bézier 19 Control Points

In previous section, number of Bézier control points are kept in a lower level, since the priority in the previous section was to provide a well-working methodology that solves the problem. As a further step, number of control points is increased so as to increase the dimensions of the search space, and have more control on the design. Now the turbofan geometry is controlled with 19 geometric control points, in addition to the twist factor, see Figure 5.11.

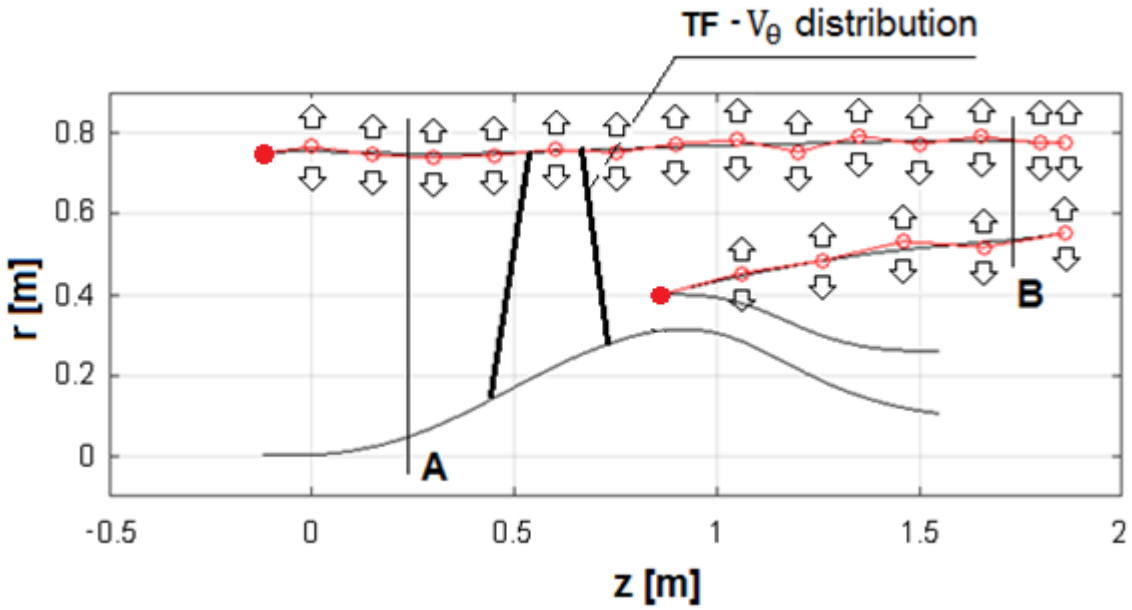


Figure 5.11. Graphical illustration of the problem with 19 geometric design variables

Due to the higher number of design variables, initial geometry population for the problem has to be regenerated. From the experience, it is known that individual number of the initial population has to be at least 2 times the number of design variables. Therefore a randomly generated initial population with 44 members is provided.

Objective function, constraints and design variables for the throughflow optimization with higher number of design variables are provided in Equations (5.24) ~ (5.31). The problem is, again, multi-objective. At this phase of the study, instead of minimum entropy generation, it's decided to use an objective function which has a wider usage in turbomachinery field, such as maximizing efficiency (η), Equation (5.4). In

addition, normalized standard deviation of total pressure distribution at station B is targeted to be minimized. In this case, normalization is performed with 1000 Pa that both objectives will be in the same order of magnitude. Design variables are constrained such that they are allowed to have values in the range of ±0.0252 m for geometric parameters and twist factor is allowed to be in the range of $0.700 < TF < 0.850$.

For the present problem, each objective is weighted with the factor 0.5, meaning that each objective has the same importance level.

Minimize $$0.5 \times \left(\frac{1}{\eta_{AB}}\right) + 0.5 \times \overline{std(P^0)_B} \quad (5.24)$$

Subject to $$DF^* < 2 \quad (5.25)$$

$$V_{hub} > 60m/s\ ,\ V_{shroud} > 60m/s \quad (5.26)$$

$$R_{C,endwalls} < 1.2[\max(R_{C,endwalls})]_{initial} \quad (5.27)$$

$$1.60 < PR < 1.63 \quad (5.28)$$

$$6.23 < BPR < 7.54 \quad (5.29)$$

$$\left.\frac{\partial V_m}{\partial m}\right]_{bypass} < 1.5\left[max\left(\left.\frac{\partial V_m}{\partial m}\right]_{bypass}\right)\right]_{initial} \quad (5.30)$$

Where $$\vec{x}^T = \{R_1, R_2, \dots, R_{19}, TF\} \quad (5.31)$$

5.1.6.1. Results

The computation time for each generation is 4.5~5 hours. As noted before, the initial population members of this case contains 44 individuals, which are randomly generated and totally different than the previous case. Due to the excessive number of initial family population, these individuals are not plotted in this section. Different values of design variables for successive iterations are also not provided since the figure becomes so incomprehensible. Instead, the band that the variables are distributed are plotted, where the optimum geometry is highlighted in red, see Figure 5.12. The objective function values obtained in successive iterations are plotted in Figure 5.13. The optimization run is stopped in 13th iteration since no improvement is observed in last 2

generations. Note that effect of generation number is investigated in the following chapters.

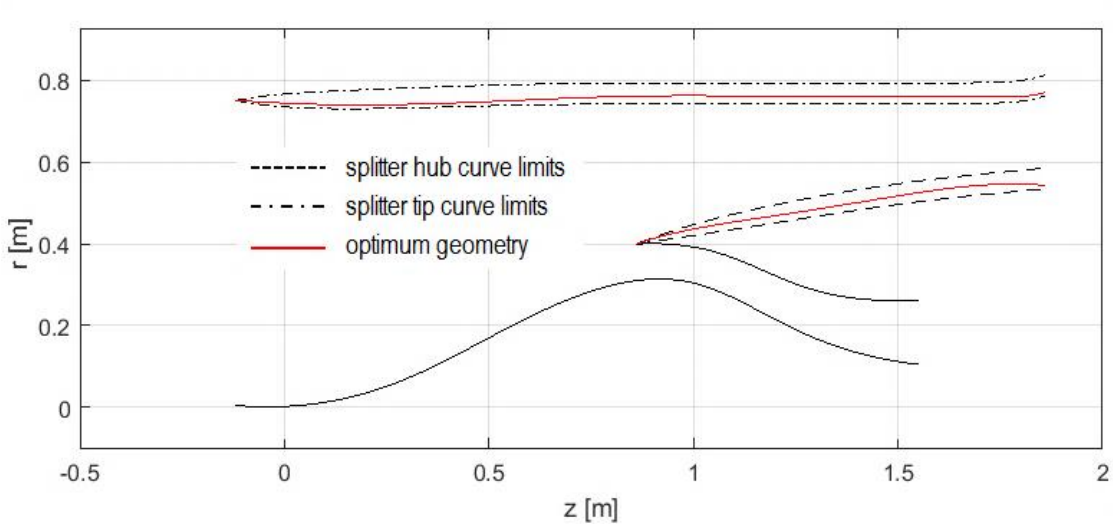


Figure 5.12. Designs obtained in optimization with 19 geometric design variables

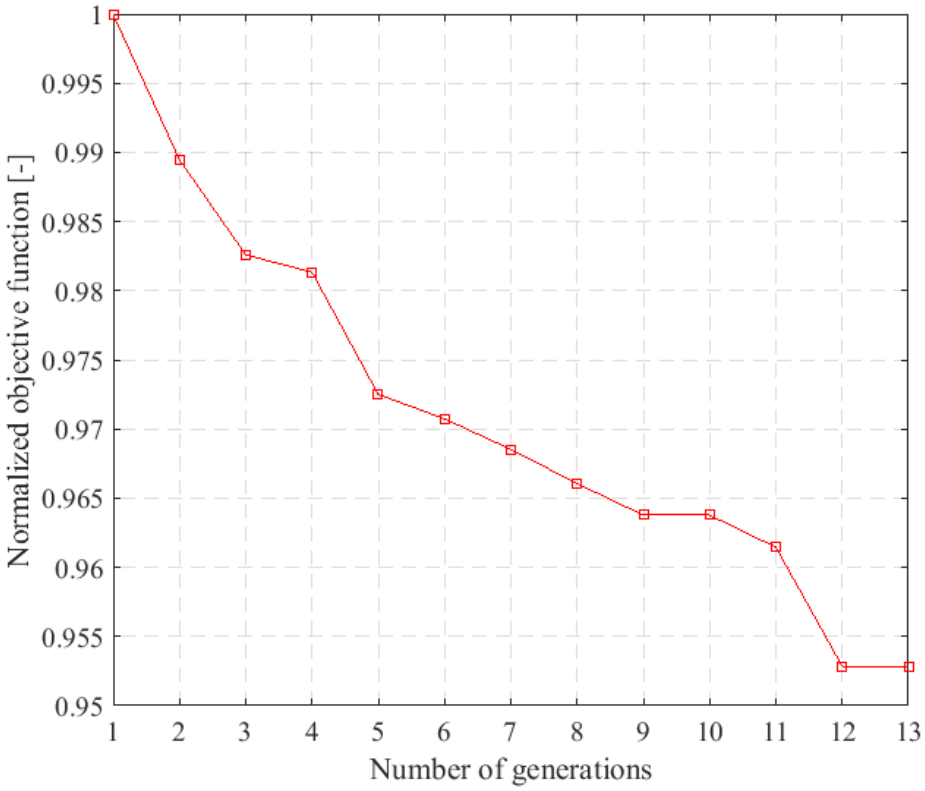


Figure 5.13. Normalized objective function for successive generations

For this unique case, the genetic algorithm resulted in design variable vector and objective function value (maximum efficiency and minimum total pressure deviation at the fan outlet) as tabulated in Table 5.6. It is seen that 4.71 % improvement in terms of objective function is observed with reference to the initial best values. By increasing the number of design variables, it is seen that the design space is explored much efficiently though the lower and upper bound limits are narrower with respect to the previous case. Note that the computations are stopped at 13th iteration, although the convergence criterion is still not met. Thus, one can conclude that with a payback in terms of computational time, genetic algorithm can provide better results with higher number of design variables. The constraint values are tabulated in Table 5.7 and it is seen that all constraints are respected in the final geometry.

Table 5.6. Objective function values for optimum geometry

	Initial best	**Optimum (GA alone)**	**Improvement (GA alone)**	**Optimum (Hybrid)**	**Improvement (Overall)**
η	87.9%	88.1%	0.16%	88.1 %	88.1%
$\mathbf{Std(P^o)_B}$	1527 Pa	1403.6 Pa	8.1%	1399.6 Pa	8.34%
$f(\vec{x})$	1.3323	1.2695	4.71%	1.2675	4.87%

Table 5.7. Throughflow optimization: Constraint values

	GA Alone	**Hybrid**	**Constraint limits**
DF	1.69	1.69	< 2
$\mathbf{V_{hub}}$, $\mathbf{V_{shroud}}$	112.11 m/s	111.98 m/s	> 60 m/s
$\mathbf{R_{c,\ endwalls}}$	1.09 m	1.08 m	< 1.70 m
$\left.\frac{\partial V_m}{\partial m}\right]_{bypass}$	903 1/s	896 1/s	$< 1.5 \left[max\left(\left.\frac{\partial V_m}{\partial m}\right]_{bypass}\right)\right]_{initial}$
PR	1.615	1.615	$1.60 < PR < 1.63$
BPR	7	7	$6.23 < BPR < 7.54$

Mach number and entropy distributions of the optimum geometry, as well as the initial population members with worst and best objective function values are provided in Figure 5.16 and Figure 5.17. As expected, one cannot observe significant differences between these contours since the efficiency levels of each individual are nearly same, see Table 5.6 and Figure 5.14. The author concludes that the perturbation in the fan geometry provided by genetic algorithm do not change the effects of secondary flows or any other loss mechanisms, neither in positive, nor in negative way. Another possibility is that geometric changes are dominated by the effects of twist factor, which plays a major role on blade geometry, therefore on total pressure distribution (and its standard deviation).

On the other hand, the effect of optimization is clearly observed in total pressure standard deviation levels. As previously seen in Table 5.6, 8.1% improvement is observed in this quantity. Although a small decrease in pressure ratio with reference to the initial best individual is observed, the standard deviation of this quantity is now much matter, which will result in a significant improvement in the thrust of the engine where the fan is mounted. These conclusions can also be deduced from Figure 5.15. Note that the discontinuity observed for the span value of 0.2~0.3 stands for the reason that the flow regime passes to compressible regime in this region, where the correlations used for estimating losses change.

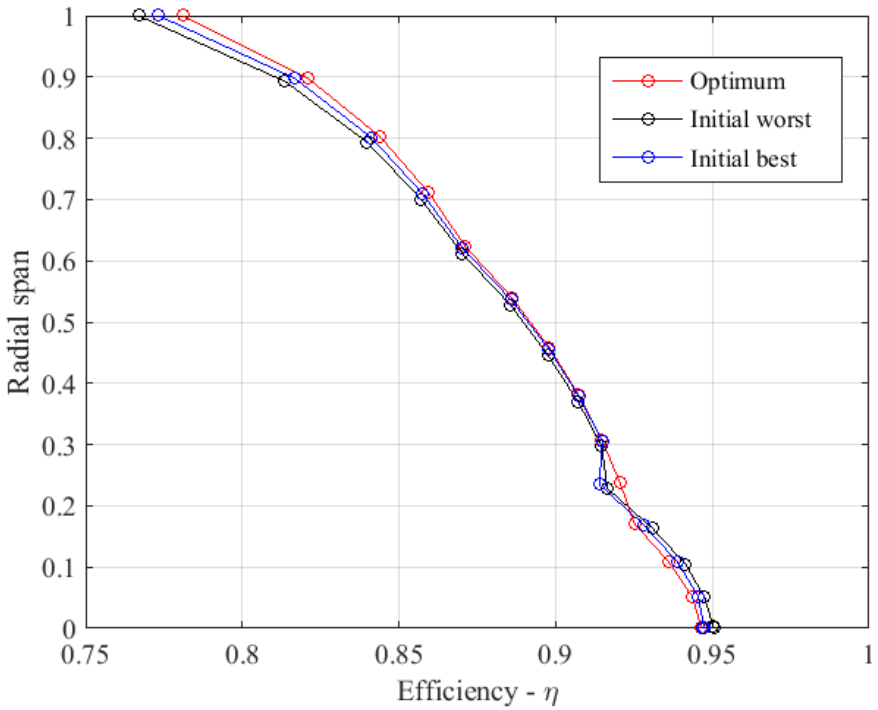


Figure 5.14. Spanwise efficiency distributions

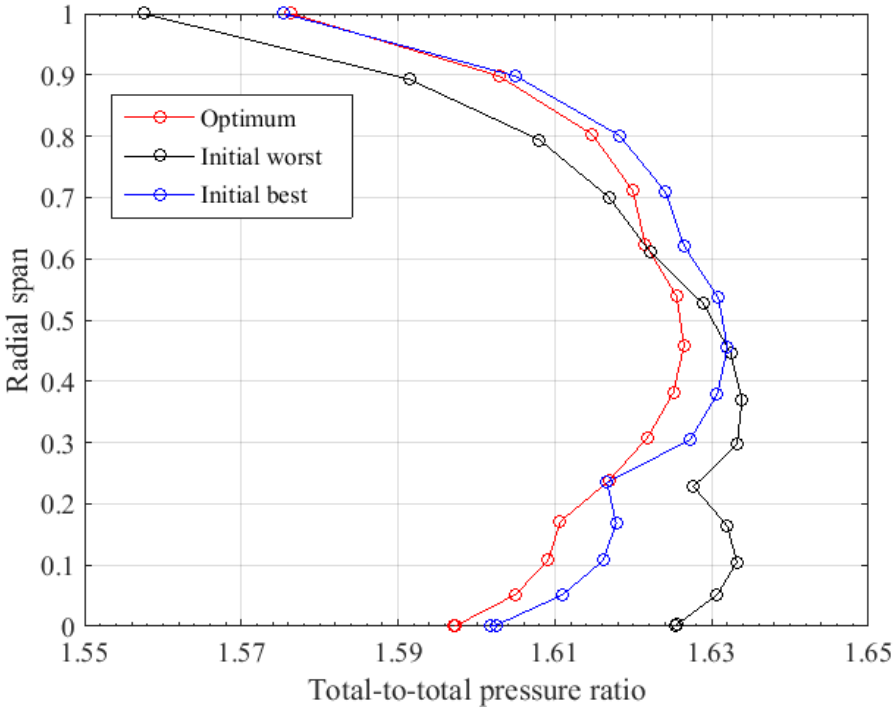


Figure 5.15. Spanwise total pressure distributions

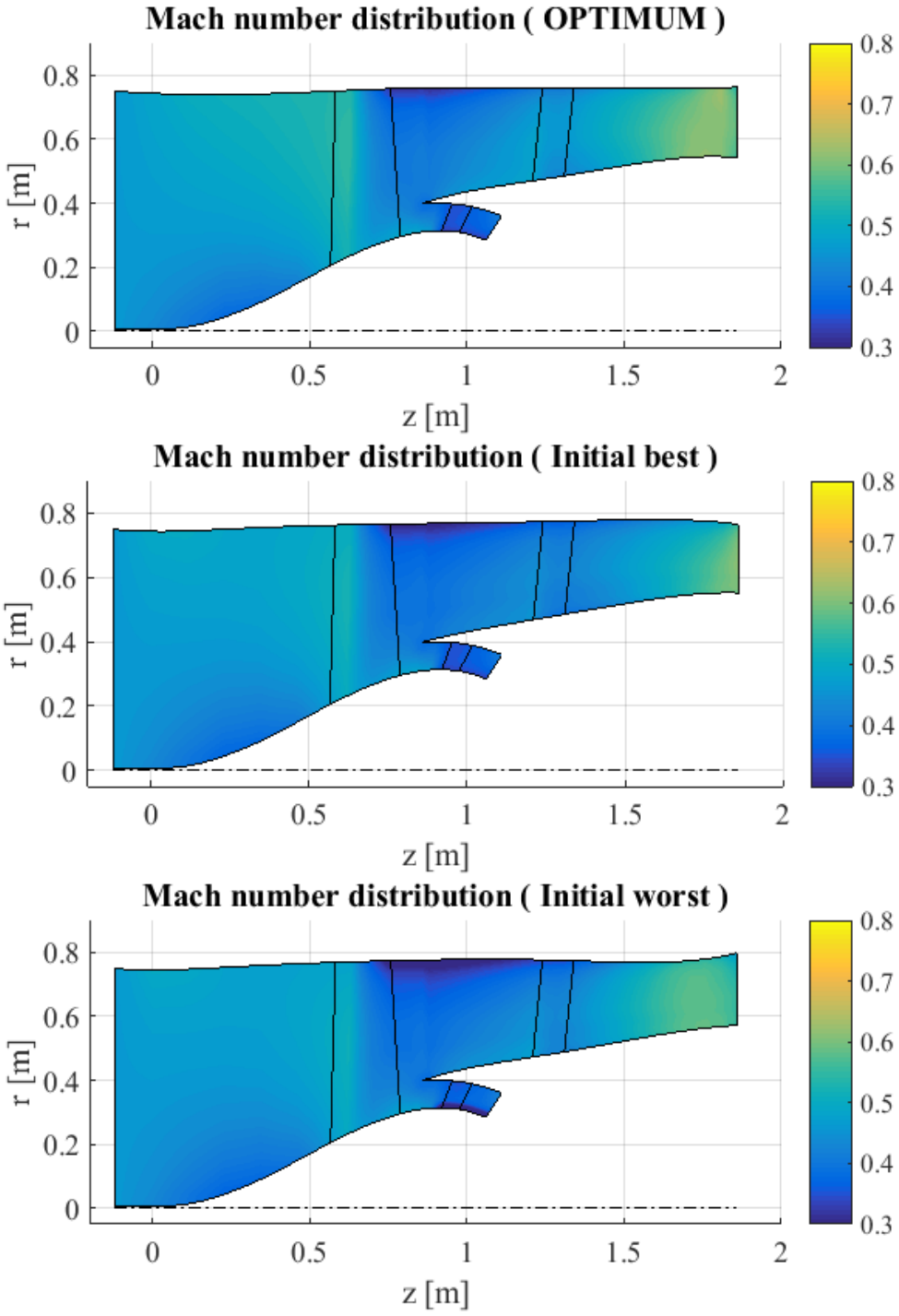


Figure 5.16. Mach number contours for initial and optimum geometries

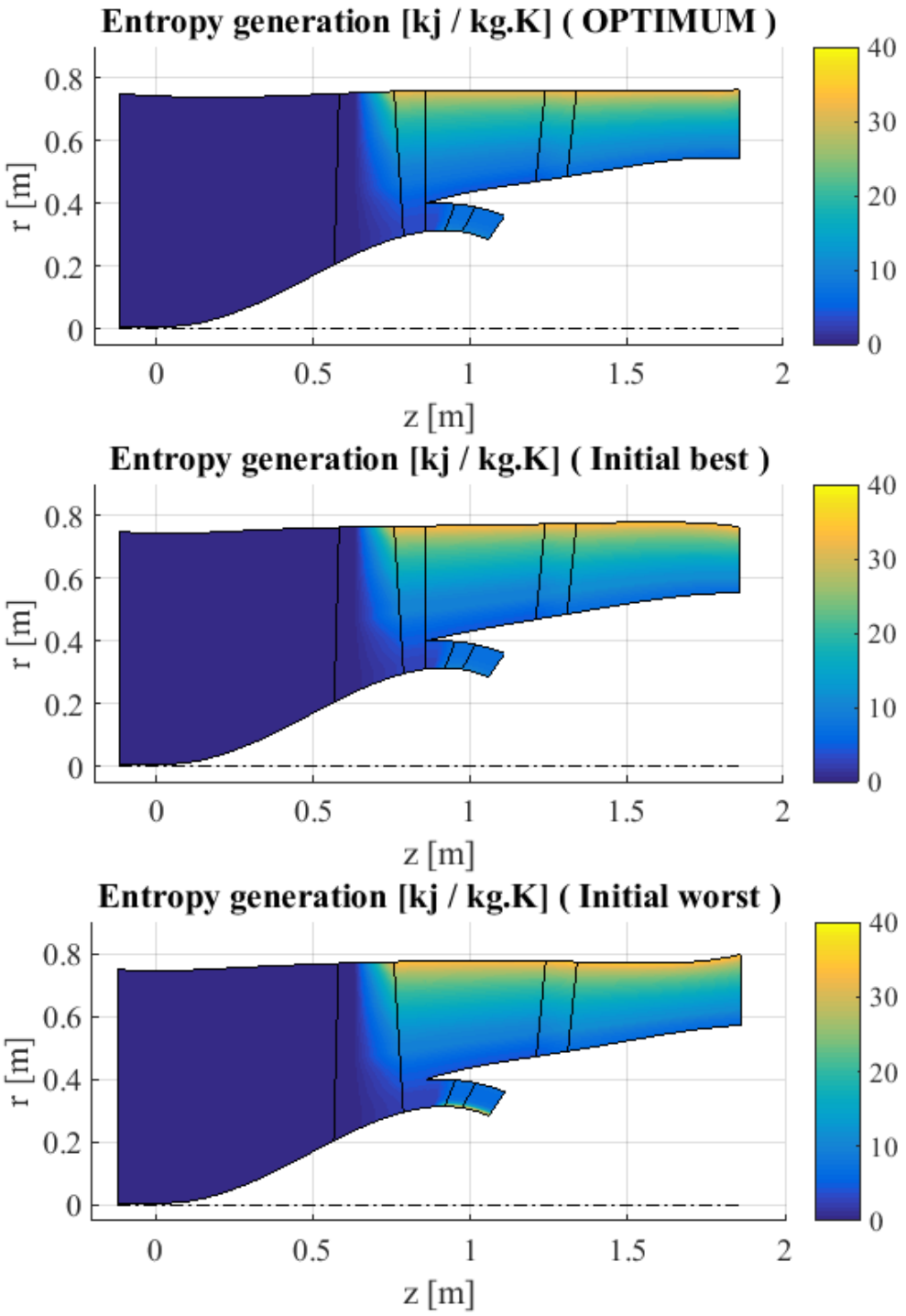


Figure 5.17. Entropy generation contours for initial and optimum geometries

Following the optimization with genetic algorithm, gradient optimization (i.e. hybrid method) is implemented on the almost optimum geometry. Gradient optimizer is stopped as the convergence criteria is satisfied (i.e. 0.1% relative difference between optimum results obtained in successive iterations). This procedure provided an overall

improvement in the order of 4.87% on the objective function value, Table 5.6, where all constraints are satisfied, Table 5.7. At the end of the procedure, it is seen that the geometry is kept almost unchanged (Figure 5.18); however a smooth adjustment is made on twist factor (**TF**) in the order of 0.25%. Resulting total-to-total pressure distribution is plotted in Figure 5.19.

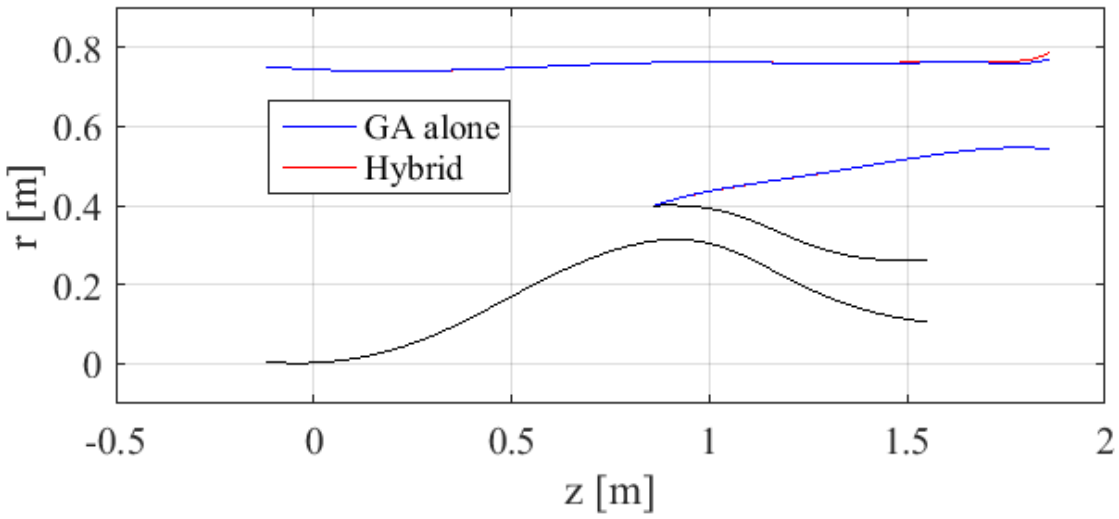


Figure 5.18. Comparison of flow path geometries for GA and hybrid optimization

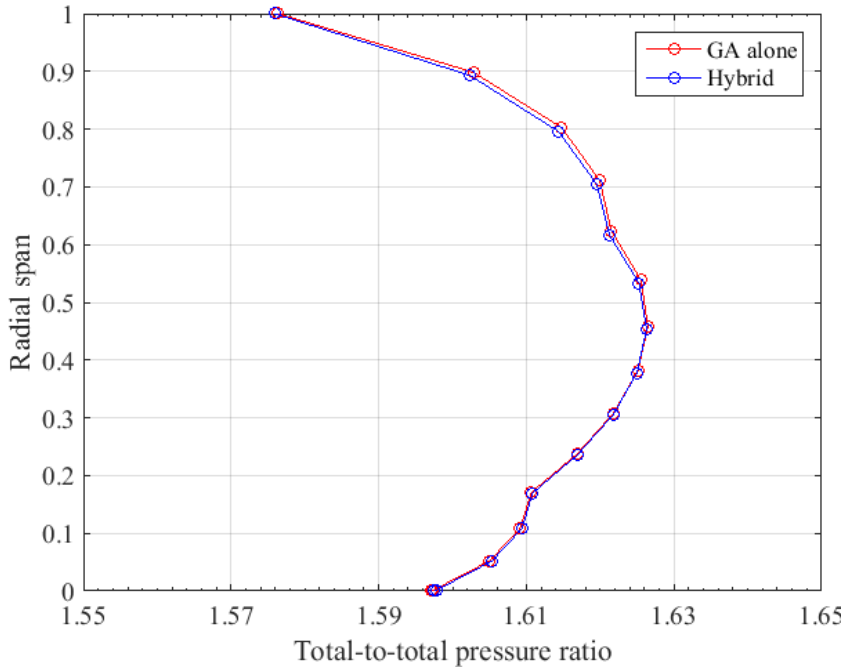


Figure 5.19. Spanwise total pressure distributions of GA and hybrid optimization

5.1.6.2. Effect of Various Parameters on GA Optimization

5.1.6.2.1. Effect of mutation rate

The effect of mutation rate on the GA performance is investigated. For this purpose, different values are posed as mutation rate value, while keeping design variable numbers (20 variables), objective and constraint functions same, as well as the initial family. The study is illustrated in Figure 5.20. The abscissa and ordinate represents mutation rate and convergence time, respectively. The numeric values highlighted with red in the figure shows the improvement in the objective function value. The highest mutation rate presented in the figure is in the order of 4%, since no convergence can be obtained for higher values. One can suggest from Figure 5.20 that after a certain percentage (4%), higher mutation rate results in a significantly higher computation time, as an increase is also observed in objective function value improvement. Note that these deductions are valid for the case studied on this thesis; therefore a universal rule cannot be stated.

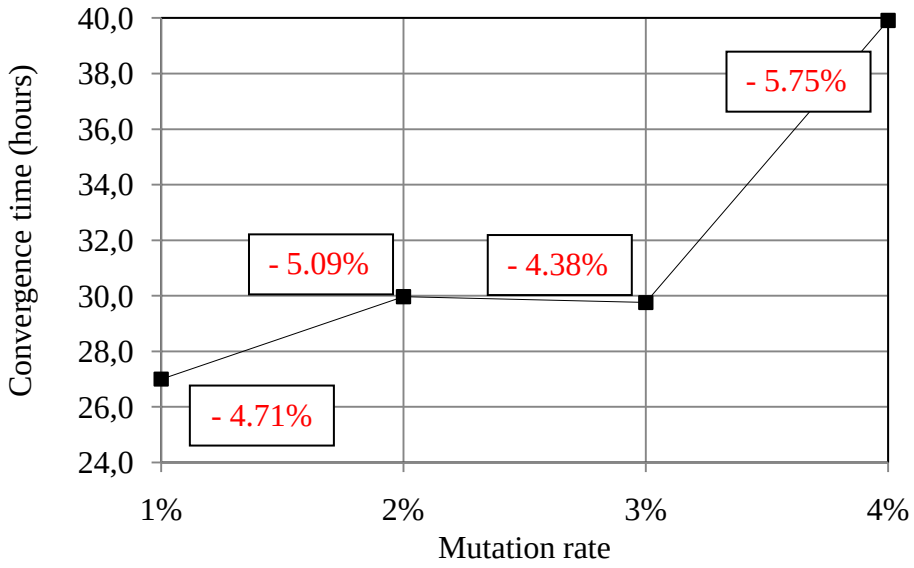


Figure 5.20. Effect of mutation rate on convergence time and improvement

5.1.6.2.2. Effect of Number of Generations

In previous chapters, optimization task is finished in case no improvement is observed in objective function at 2 successive generations. It is decided, then, to increase proceed the optimization task, even though the case stated in previous sentence has taken place. Effect of increasing number of generations is plotted in Figure 5.21 for different range of design variable bound values. It is seen from the figure that, further improvement in objective function is possible, even in the cases where no improvement is observed in previous (more than one) iterations. For example, improvement in objective function is observed at 22nd generation, following 10 generations (i.e. between 10th and 20th generations) without any improvement. The situation is due to the presence of mutation operator, which is responsible for exploring the design space and introducing diversity to the algorithm. Thus, the author suggests that the designer would better pick computational time as the convergence criteria, rather than the relative difference between the population's best members' fitness values at successive iterations.

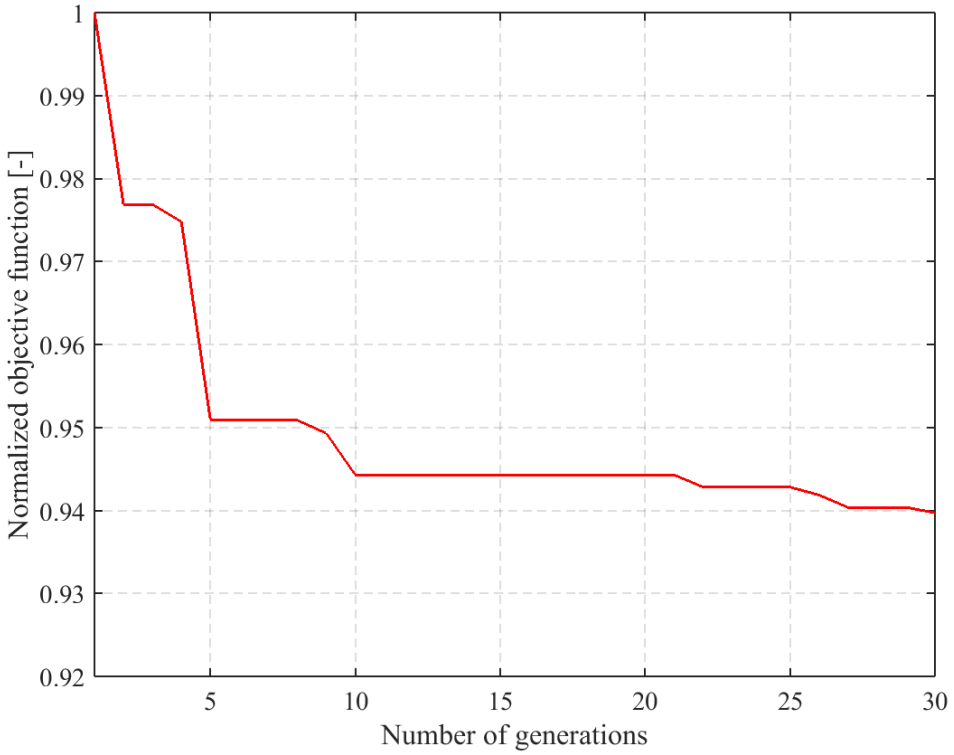


Figure 5.21. Effect of number of generations on objective function improvement

5.1.6.2.3. Effect of Geometric Constraints

In the results summarized in Section 5.1.6.1, the bound range for each geometric design variable (i.e. the numerical difference of each geometric variable: [Lower bound value]$_i$ – [Upper bound value]$_i$) was picked as 50 mm. The bound range for the twist factor was 0.150 for the same reference case. In order to understand the effect of lower and upper bound limits, different bound range values are used as given in Figure 5.22. For each computation point in the figure, mutation rate is taken as 1%, which is the same value as the reference case. Note that constraint limits are different for different bound range values from that of the reference case. It is deduced from Figure 5.22 that objective function improvement increases remarkably for the range $25 < \Delta < 75\, mm$ and corresponding twist factor range depicted in the figure. The reason for this is that there are local optimum solutions for $\Delta < 75\, mm$, and once a wider search is provided, the global optimum is hit. It is understood that the optimum solution lies in the range $75 < \Delta < 275\, mm$, increasing search space bounds does not help the optimizer converge to a remarkably better solution. The computation time for each generation presented in Figure 5.22 is 4.5~5 hours in average. The improvement of normalized objective function for different range of design variable bounds at each generation is given in Figure 5.23. Radial distributions of objective function values for the problem are given in Figure 5.24 and Figure 5.25.

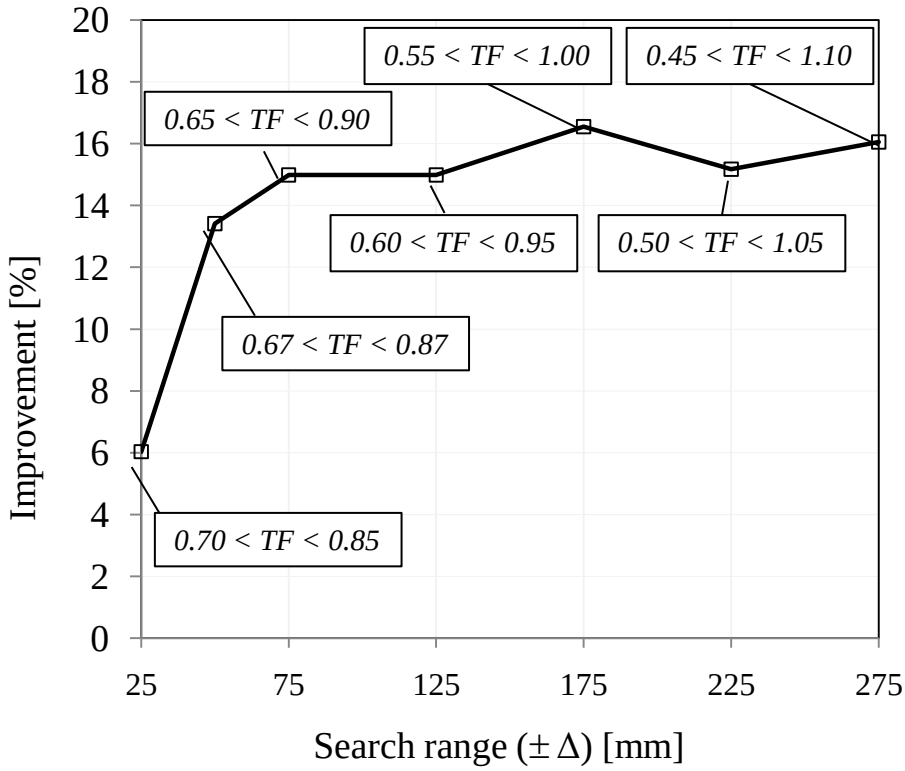


Figure 5.22. Effect of bound range on objective function improvement

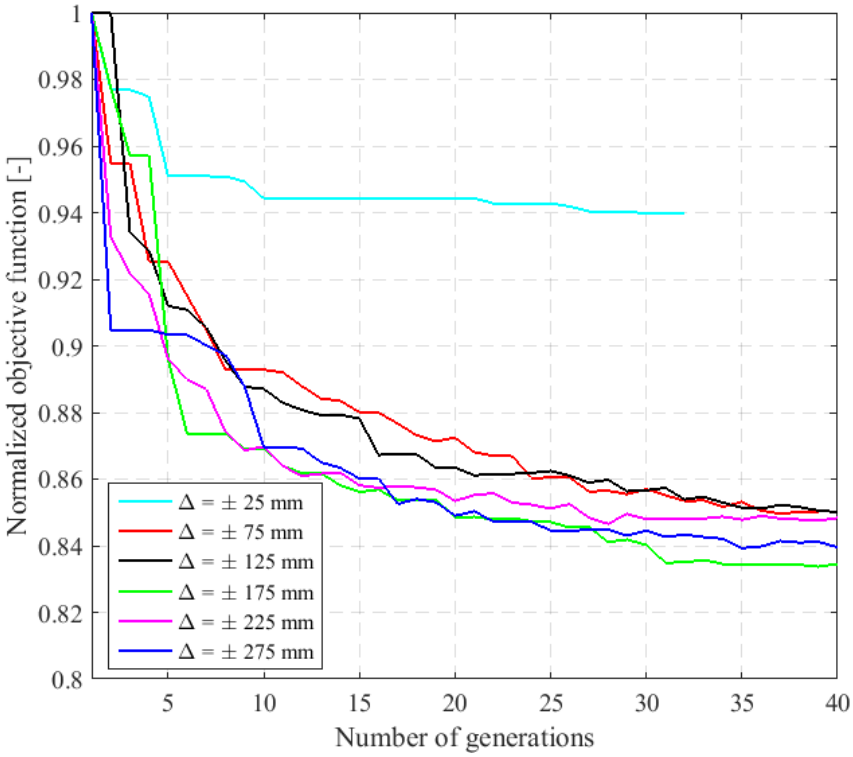


Figure 5.23. Effect of bound range on objective function improvement

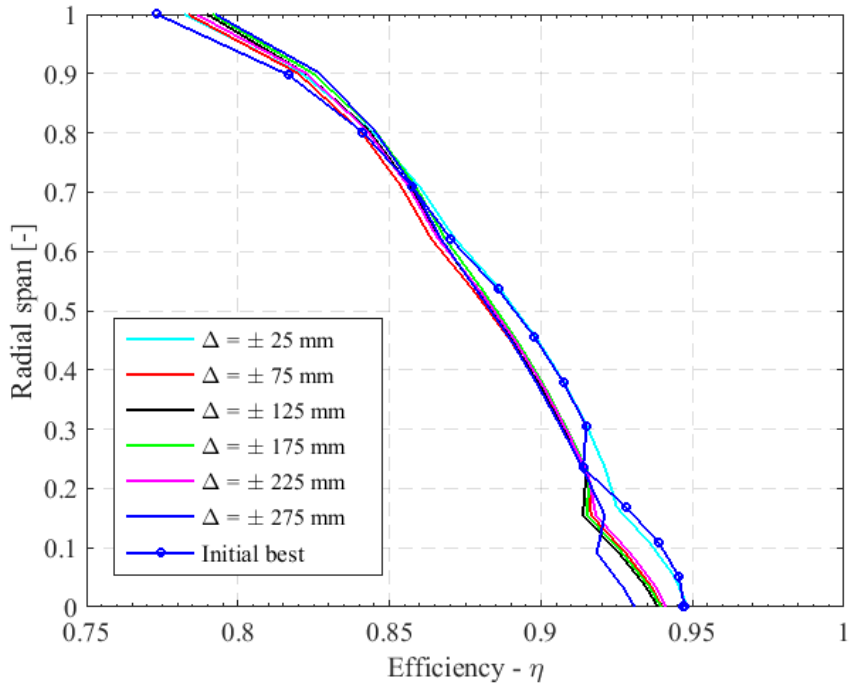


Figure 5.24. Spanwise efficiency distributions

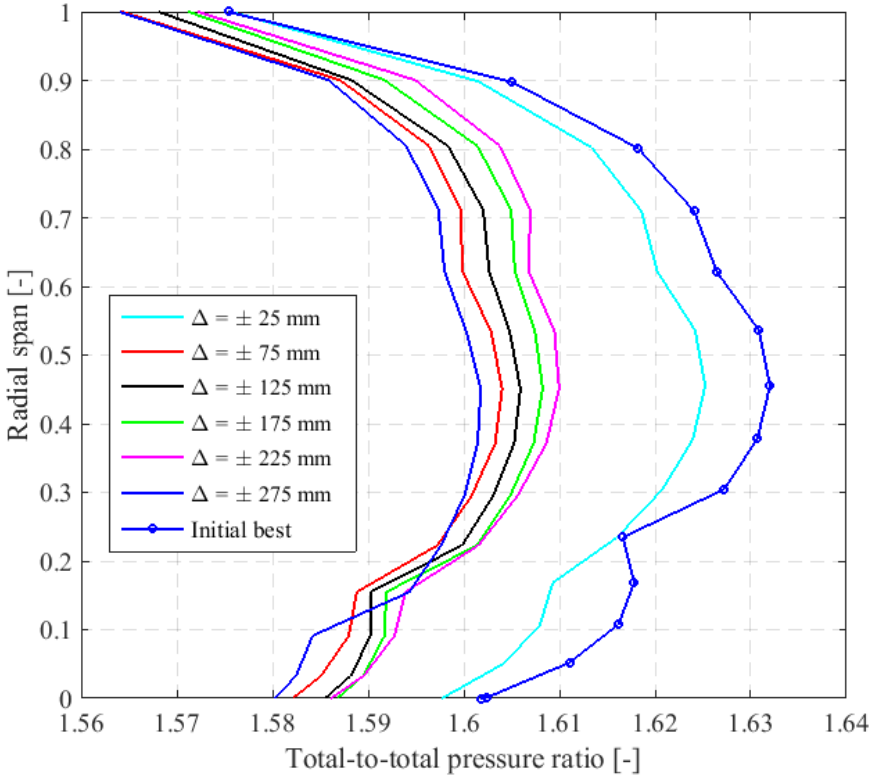


Figure 5.25. Spanwise total-to-total pressure ratio distributions

CHAPTER 6

THROUGHFLOW OPTIMIZATION USING PSO

As mentioned in the literature survey, aerodynamic optimization of the throughflow problem with particle swarm optimization method is an untouched area. Therefore, an effort has been put to understand the applicability of the method on the problem studied in this thesis. For this purpose, the open source code *Constrained Particle Swarm Optimization* is used. The tool is coded in MATLAB environment and is available in MATLAB User Community portal (Chen, 2016).

The PSO program utilized in this study is implemented in a workstation working with an Intel Zeon CPU ® (2.66 GHz – 4 processors). The memory of the workstation is 64 GB. The results represented in this section includes optimization tasks performed with one processor, however the program is available for parallel computing applications due to lack of MATLAB Parallel Computing Toolbox.

6.1. Validation of the Open-Source PSO Program

The program's global optimization capability is validated by means of various test cases. As an example, validation test of the open-source PSO program with the analytical test function (Equation (4.1)) available in the literature (Deb, 1999) is presented, see Figure 6.1. The figure shows that the program is capable of finding the global minimum with good accuracy.

$$f(r) = 1 - e^{4r} sin^4(5\pi r) \tag{6.1}$$

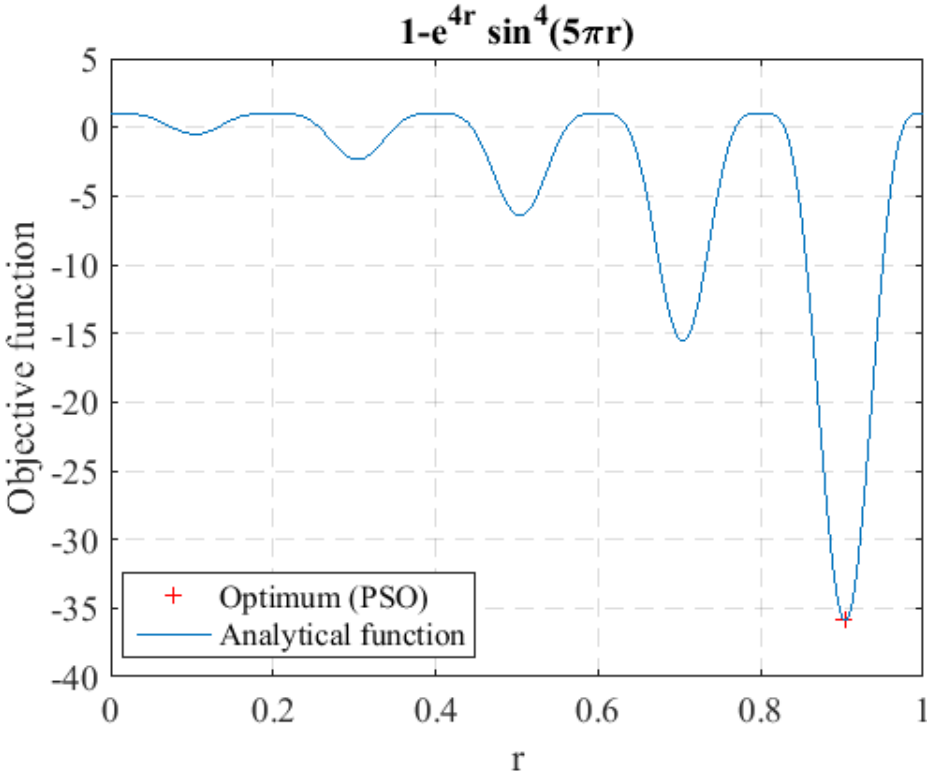


Figure 6.1. Validation of the open-source PSO program

6.2. PSO Configurations

The PSO algorithm, as discussed in Section 2.1.3.1.4.1, uses Equation (2.16) for finding the global optimum. As mentioned before, unit time step is used in Equation (2.17) as it is the common practice in PSO literature thus, Equation (2.16) takes the form given in Equation (4.4):

$$f\overrightarrow{v_t^{k+1}} = w.\overrightarrow{v_t^k} + c_p.r_p.\left(\overrightarrow{p_t} - \overrightarrow{v_t^k}\right) + c_g.r_g.\left(\vec{g} - \overrightarrow{v_t^k}\right) \tag{6.2}$$

In throughflow optimization, weight factor, $\boldsymbol{w}$, is assigned as 0.9 for the first iteration, where it is linearly decreased to the limit of 0.4 by increasing iteration number. Social and cognitive constants, $c_g r_g$ and $c_p r_p$, are fixed as 1.25 and 0.5, where the limits for these values given in Equation (2.21) and (2.22) are respected.

6.3. Optimization of Single Variable Throughflow Problem

The applicability of PSO on throughflow optimization is investigated in this chapter. For this purpose, the throughflow geometry utilized in Section 5.1.6 is taken into service. As a starting point, geometric parameters are now frozen and only twist factor is picked as a design variable. Physically speaking, this means that during the optimization process, aerodynamic flow path is kept constant, whereas blade exit tangential velocity distribution, which is controlled by twist factor (TF) is perturbed. The bound range for the twist factor is, $0.5 < TF < 1$.

The aim of PSO optimization is decided as obtaining minimum standard deviation of total pressure distribution and maximum efficiency at the fan by-pass exit region, as it was done so in GA optimization practice introduced in previous chapters. The problem is now multi-objective, as the target is to maximize efficiency (minimize $1/\eta_{AB}$) and minimize normalized total pressure distribution non-uniformity (or normalized standard deviation of total pressure distribution - i.e. $\overline{std(P^0)_B} = (std(P^0)_B)/1000$) at station B, simultaneously (See Figure 5.11). For the present problem, each objective is weighted with the factor 0.5, meaning that each objective has the same importance level. In order to obtain a fast solution, an unconstrained optimization problem is taken into consideration. Note that, the same unconstrained problem is also optimized with genetic algorithm with the same design variable and objective function. The initial families for the genetic algorithm and particle swarm optimizer problems are same, which include randomly generated 20 individuals. The iteration number for genetic algorithm is picked as 6. Similarly, number of generations for particle swarm optimizer is decided as 6. In order to understand whether the optimizer can find the optimum starting with non-optimum initial family members, twist factor values are picked to be different than the a priori known optimum value.

$$\text{Minimize} \qquad 0.5 \times \left(\frac{1}{\eta_{AB}}\right) + 0.5 \times \overline{std(P^0)_B} \tag{6.3}$$

$$\text{Where} \qquad \vec{x}^T = \{TF\} \tag{6.4}$$

Table 6.1. Unconstrained throughflow optimization with GA & PSO

	Initial best	Optimum (GA alone)	Optimum (PSO alone)
$\vec{x}$	0.808	0.752	0.752
$f(\vec{x})$	0.995	0.931	0.931
Improvement		~ 6%	~ 6%
Comp. time		11 h	39 h

Table 6.1 points to the situation that GA and PSO find the same optimum, given that both optimizers start with the same initial family. Number of iterations for both algorithms is selected to be equal; therefore a more reasonable comparison became possible. It should be noted that computational time for PSO calculations takes more than three times with respect to that of the GA optimizer. The reason behind this may be high the value of the weight factor used in PSO, which deals for global exploration of the design space. Lower values for weight factor would decrease the computation time; however it cannot be guaranteed that the optimizer will find the same global optimum. A sensitivity analysis is therefore recommended as a future study.

6.4. Optimization of MO Constrained Throughflow Problem

Once the applicability of PSO is provided, the algorithm is now utilized to optimize the multi objective throughflow problem with 20 design variables, with same initial family, same objectives, same constraints and same design variable bounds with that of used in 5.1.6, case $\Delta = 75\,mm$. To provide a better understanding, the optimization problem is summarized below,

Minimize $$0.5 \times \left(\frac{1}{\eta_{AB}}\right) + 0.5 \times std(\overline{P^0})_B \quad (6.5)$$

Subject to $$DF^* < 2 \quad (6.6)$$

$$V_{hub} > 60m/s\,,\, V_{shroud} > 60m/s \quad (6.7)$$

$$R_{C,endwalls} < 1.2[\max(R_{C,endwall\ s})]_{initial} \quad (6.8)$$

$$1.60 < PR < 1.63 \tag{6.9}$$

$$6.23 < BPR < 7.54 \tag{6.10}$$

$$\left.\frac{\partial V_m}{\partial m}\right]_{bypa\ ss} < 1.5\left[max\left(\left.\frac{\partial V_m}{\partial m}\right]_{bypass}\right)\right]_{initial} \tag{6.11}$$

Where $$\vec{x}^T = \{R_1, R_2, \dots, R_{19}, TF\} \tag{6.12}$$

The optimization results of GA and PSO tasks for the problem summarized above is seen in Table 6.2. The constraint values of PSO computations are tabulated in Table 6.3 and it is seen that all constraints are respected in the final geometry. After 40 PSO iterations, 8% improvement in objective function is observed, where all constraints are respected. The computation time for this optimization run is 25.8 hours. The improvement observed for GA is slightly same at the end of same computational time, i.e. 8.5%. It is observed that once genetic algorithm is let run for higher number of generations, a significant increase is observed in objective function improvement with an order of 13%. On the other hand, no further improvement in objective function can be seen in PSO by increasing number of iterations. The author's opinion is that this may be due to configuration of PSO factors given in Chapter 6.2. Improving PSO's performance is not in the scope of this work, but it's the author recommendation to the future researchers to keen on this subject.

Table 6.2. Objective function values for optimum geometry

		GA (40 gen.)		**GA (5 gen.)**		**PSO**	
	Init. best	**Opt.**	**Imp.**	**Opt.**	**Imp.**	**Opt.**	**Imp.**
η	87.9%	87.7%	-0.23%	87.6%	-0.4%	88.3%	0.45%
$Std(P^o)_B$	1527 Pa	1125 Pa	26.3%	1295 Pa	15.2%	1320 Pa	13.55%
$f(\vec{x})$	1.332	1.1414	13.04%	1.219	8.5%	1.226	8%
Com. time		232 h		29 h		26 h	

Table 6.3. Throughflow optimization with PSO: Constraint values

	Constraint values	**Constraint limits**
DF	1.67	< 2
$\mathbf{V_{hub}}$, $\mathbf{V_{shroud}}$	118.2 m/s	> 60 m/s
$\mathbf{R_{c,\ endwalls}}$	1.41 m	< 1.70 m
$\left.\frac{\partial V_m}{\partial m}\right]_{bypass}$	1217 1/s	$< 1.5 \left[max\left(\left.\frac{\partial V_m}{\partial m}\right]_{bypass}\right)\right]_{initial}$
PR	1.61	$1.60 < PR < 1.63$
BPR	7.03	$6.23 < BPR < 7.54$

The spanwise distribution graphs of objectives are presented in Figure 6.2 and Figure 6.3, in order to provide a better physical understanding of the final situation of the problem. It can be seen from the figures that total-to-total pressure ratio standard deviation is remarkably reduced at the end of optimization procedure. Figure 6.2 shows that efficiency is increased at tip region, whereas a slight reduction is observed in the hub in case genetic algorithm is used. This drawback observed in GA is not seen in case PSO is utilized as optimizer. To the author's consideration, making a rigid statement on this issue is not possible since the efficiency levels are still very close to each other at the end of computations using both techniques.

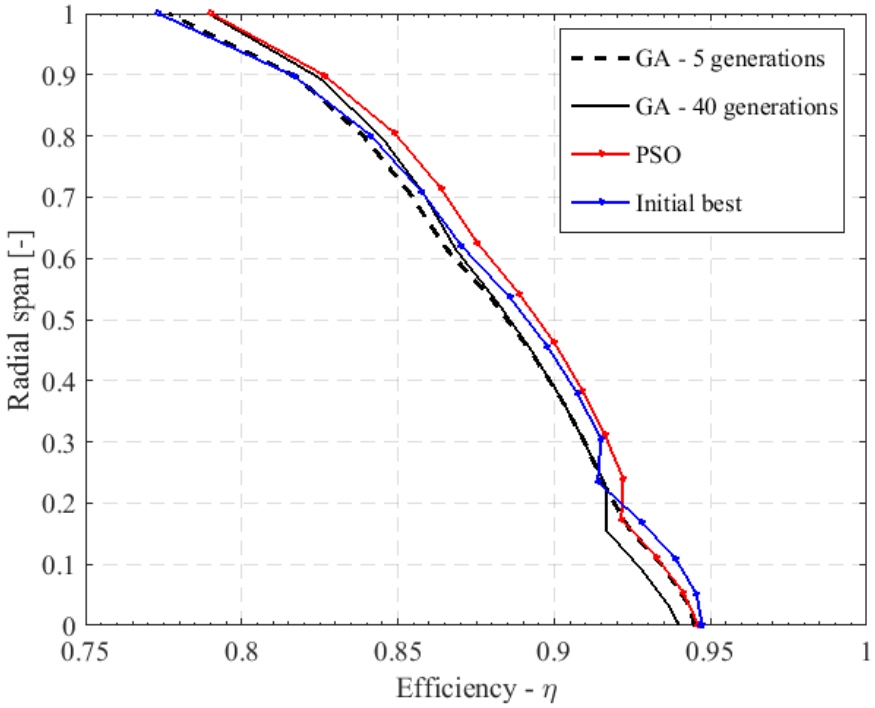


Figure 6.2. Spanwise efficiency distribution

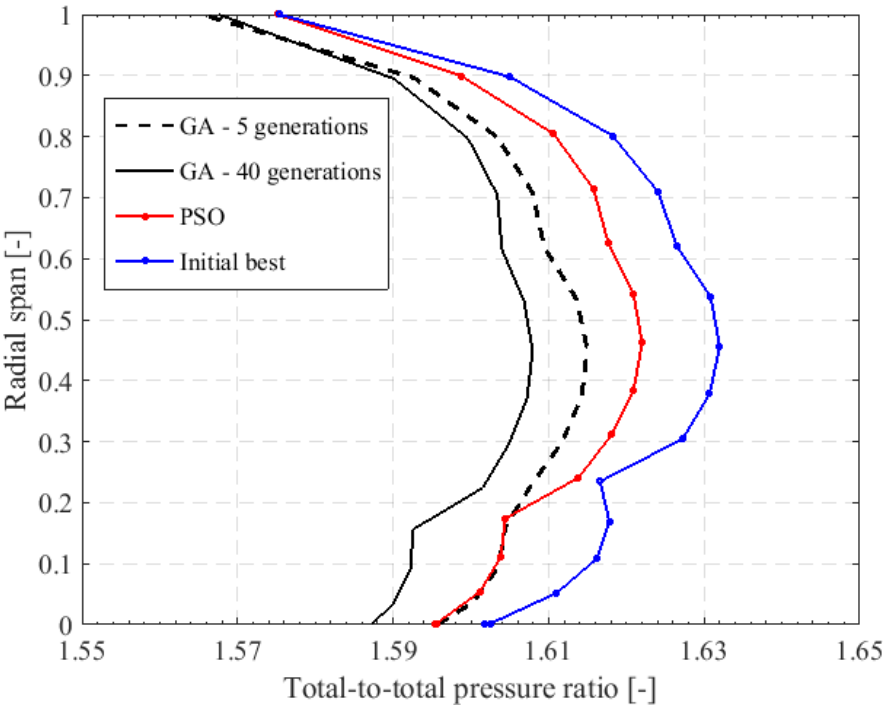


Figure 6.3. Spanwise total-to-total pressure ratio distribution

The Mach number and entropy generation distribution contours of optimum results are graphically illustrated in Figure 6.4. The similarity of the entropy distribution contours is expected since efficiency values are very slightly different as presented in Figure 6.2.

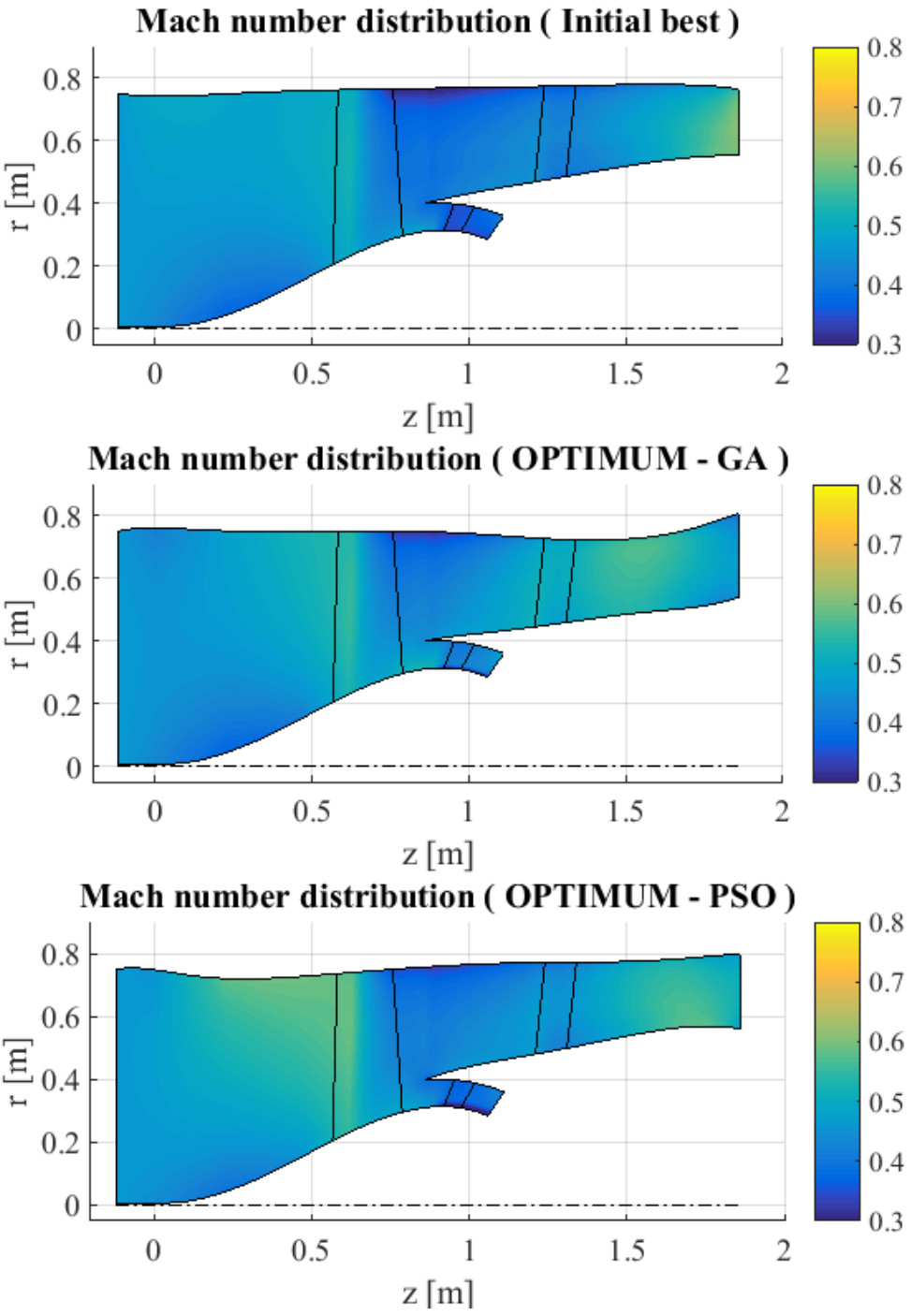


Figure 6.4. Mach number contours for initial best and optimum geometries

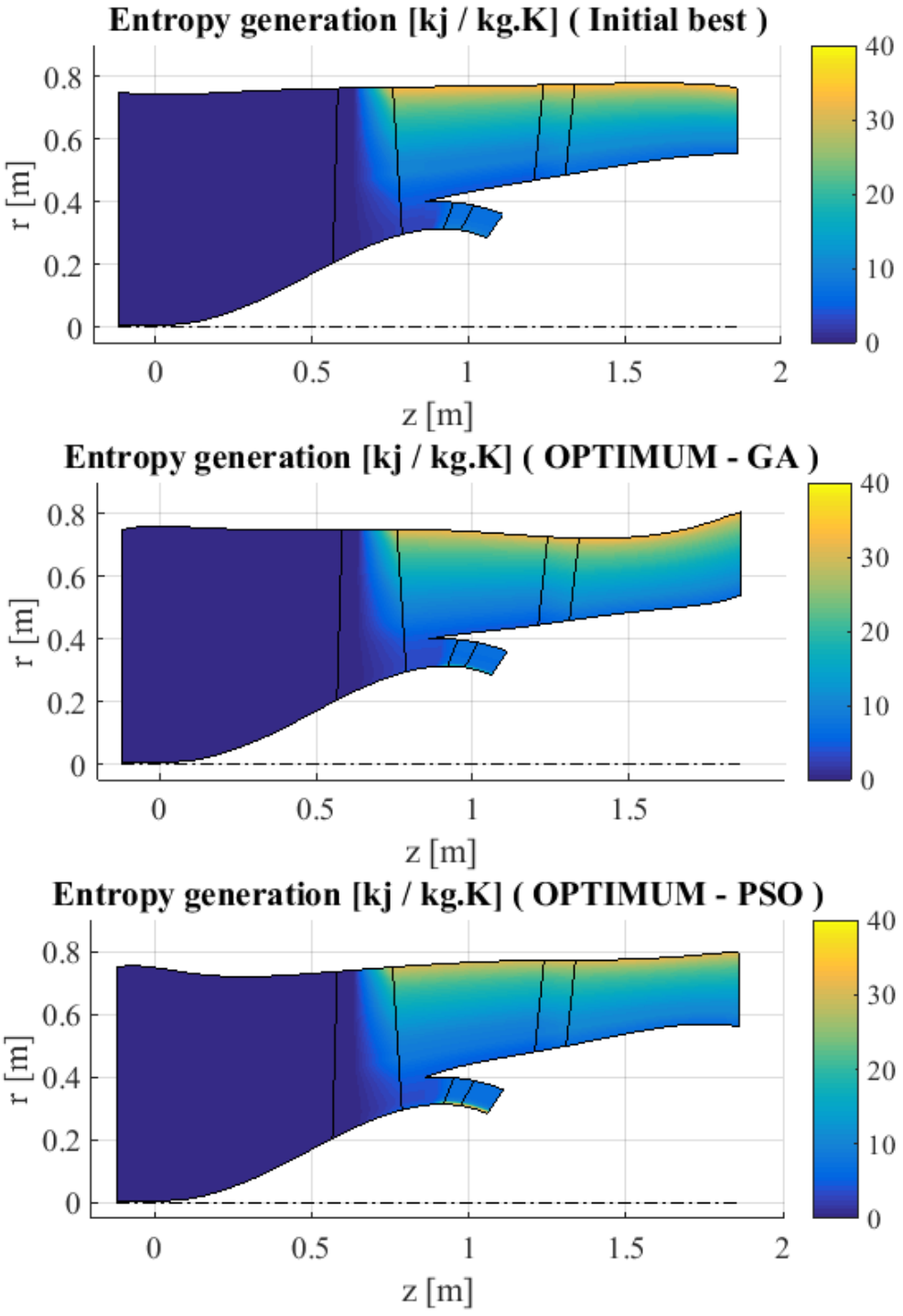


Figure 6.5. Entropy generation contours for initial best and optimum geometries

CHAPTER 7

CONCLUSIONS

Aerodynamic optimization of a transonic aero-engine fan module is studied in this thesis. The aerodynamic design of the module is obtained by a throughflow solver developed by Acarer (Acarer, 2015). This phase of the design is known to be the backbone of the turbomachinery aerodynamic design procedures. In order to get most efficient and economic designs, automatic optimization techniques using evolutionary and swarm inspired techniques are taken into service.

Among zero order methods, evolutionary algorithms are reported to fit best for turbomachinery problems, due to their good performance in discrete and non-differentiable problems and their ability to find the global optimum with ease. It is seen that genetic algorithms are the most widely used evolutionary algorithms in turbomachinery component design. However, aerodynamic optimization of the high by-pass ratio fan modules is rather untouched in the open/accessible literature. The author, therefore, keened on this subject where significant improvements in the fan efficiency and uniformity of the total pressure distribution at fan exit are obtained. Within this aspect, an in-house optimization program capable of finding the global optimum of an optimization problem using genetic algorithm and gradient based method is coded. The program is tested on analytical test functions and it is seen that it is capable of finding the global optimum of a sinusoidal curve with an almost-perfect tolerance. Moreover, the validity of the code is tested on a return channel geometry optimization and a remarkable improvement in objective function value – **15%** - is observed for this case. In addition, a generic study with limited number of design variables is performed, where the geometry is modeled using curves obtained by cubic interpolation method. In this case, genetic algorithm and gradient method are hybridized and the resulting single objective optimization code is coupled with the throughflow solver. It's seen that the program is capable of finding the design variables providing slightly better results – 0.4 % - while respecting the constraints. Furthermore, the geometry is modeled utilizing Bézier curves (with 5 control points in total) and the throughflow problem is subjected to *multi*

objective optimization process, where genetic algorithm is used alone, providing 1.73 % improvement in objective function. Final modifications on the geometric model is made such that it is now represented with 19 Bézier control points and a *twist* factor representing the twist of the rotor blade, where it is deduced that 4.87% improvement is possible with increased number of design variables. Gradient optimization using genetic algorithm results provided an additional improvement on objective function, in the order of 0.5 %. Final numerical experiments are performed by increasing design variable bounds, mutation ratio and number of generations for genetic algorithm. Among them all, the effect of number of generations is observed to be noticeable, which provided an additional improvement in the order of 2%.

Methods inspired by swarm intelligence are reported as promising global optimizers, whereas there was seen no studies that employs such algorithms for the sake of optimizing turbomachinery components. Seeing this gap in the literature, an open source particle swarm optimizer is modified and coupled with the throughflow solver, where similar improvements with that of the genetic algorithm are obtained, but with similar computational cost. Prior to its use in turbomachinery optimization, the open source code is tested on the sinusoidal test function, which also has been used in genetic algorithm validation. The results are found accurate that the code finds the optimum with ease. Throughflow optimization using particle swarm approach resulted in 8% improvement in objective function, similar to 8.5% improvement observed in genetic algorithm application for the same problem with same initial family, objective function and constraint limits.

BIBLIOGRAPHY

Acarer, S., 2015. *Development of a streamline curvature throughflow design method for fan module of turbofan engines (Ph.D. thesis).* İzmir Institute of Technology.

Acarer, S. & Özkol, Ü., 2015. An extension of the streamline curvature through-flow design method for bypass fans of turbofan engines. *Proc IMechE Part G: Journal of Aerospace Engineering,* pp. 1-14.

Acarer, S. & Özkol, Ü., 2015. *Development of a new universal inverse through-flow program and method for fullycoupled split-flow turbomachinery systems.* Montréal, Canada

Al Salihi, Z., 2010. *Bézier curves.* Rhode St. Genése: von Karman Institute for Fluid Dynamics.

Andersson, J., 2001. *Multiobjective optimization in engineering design - Applications to fluid power systems (Ph.D. thesis).* Linköpings Universitet - Institute of Technology

.

Arora, J., 2004. *Introduction to optimum design.* California, USA: Elsevier Academic Press.

Aungier, R., 2003. *Axial-flow Compressors.* New York: ASME Press.

Bandyopadhyay, S., Saha, S., Maullik, U. & Deb, K., 2008. A simulated annealing based multi-objective optimization algorithm: AMOSA. *IEEE Transactions on Evolutionary Computation,* 12(3), pp. 269-283.

Baskharone, E., 2006. *Principles of Turbomachinery in Air-Breathing Engines.* New York: Cambridge University Press.

Bellman, R., 1961. *Adaptive control processes: A guided tour.* New York: Princeton University Press.

Cagnina, L., Esquivel, S. & Coello Coello, C., 2011. Solving constrained optimization oroblems with a hybrid particle swarm optimization algorithm. *Engineering Optimization,* 43(8), pp. 843-866.

Carroll, C., 1961. The created response surface technique for optimizing nonlinear, restrained systems. *Operations Research,* 9(2), pp. 169-185.

Chauvin, J., 1977. *Meridional flow in axial turbomachines / von Karman Institute for Fluid Dynamics Course Notes 99.* Rhode St. Genése, Belgium.

Chen, S., 2016. *MathWorks.* [Online] Available at: http://www.mathworks.com/matlabcentral/fileexchange/25986-constrained-particle-swarm-optimization [Accessed 30 5 2016].

Cox, G., Fischer, C. & Casey, M., 2010. *The application of throughflow optimization to the design of radial and mixed flow turbines,* s.l.: Institution of Mechanical Engineers.

de Weck, O., 2010. *Multidisciplinary system design optimization: Genetic Algorithms (cont.), particle swarm optimization, tabu search, optimization algorithm selection.* Boston, New York, USA: MIT OpenCourseWare.

Deb, K., 1999. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation,* 7(3), pp. 205-230.

Deb, K., 2008. *Multi-objective optimization using genetic algorithms.* 2 ed. s.l.:John Wiley and Sons Press.

Deb, K., 2010. *Optimization for engineering design - Algorithms and examples.* New Delhi: PHI Learning Private Ltd.

Deb, K., Pratap, A., Meyarivan, T. & Agarwal, S., 2002. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA II. *IEEE Transactions of Evolutionary Computation,* 6(2), pp. 182-197.

Dénos, R., 2005. *Turbomachinery cycles and components / von Karman Institute for Fluid Dynamics Course Note 163.* Rhode St. Genése, Belgium.

Dorigo, M., Birattari, M. & Stützle, T., 2006. Ant colony optimization - Artificial ants as a computational intelligence technique. Volume 1, pp. 28-39.

Eberhart, C. & Shi, Y., 2001. *Particle swarm optimization: Developments, applications, resources.* Seoul, Korea, pp. 81-86.

Engelbrecht, A., 2007. *Computational intelligence: An introduction.* 2 ed. West Sussex, England: John Wiley & Sons Ltd..

Giannakoglou, K., 1999. *Designing turbomachinery blades using evolutionary methods.* Indianapolis, USA.

Giles, M. & Pierce, N., 2001. An introduction to the adjoint approach to design. *Flow, Turbulence and Combustion,* Volume 65, pp. 393-415.

Guglieri, G., 2012. Using of particle swarm for performance pptimization of helicopter rotor blades. *Applied Mathematics,* Volume 3, pp. 1403-1408.

Hajela, P. & Lin, C., 1992. Genetic search strategies in multicriterion optimal design. *Structural and Multidisciplinary Optimization,* 4(2), pp. 99-107.

Hassan, R., Cohanim, B. & de Weck, O., 2005. *A comparison of particle swarm optimization and the genetic algorithm.* Texas, USA, pp. 1-13.

Hassan, R., Cohanim, B., de Weck, O. & Venter, G., 2005. *A comparison of particle swarm optimization and the genetic algorithm.* Austin, Texas, USA.

Haupt, R. & Haupt, S., 2004. *Practical genetic algorithms.* 2 ed. Hoboken, New Jersey: John Wiley & Sons, Inc.

Haykin, S., 2005. *Neural networks - A comprehensive foundation.* 9th Indian ed. Delhi, India: Pearson Education.

Jensen, P. & Bard, J., 2002. *Operations research models and methods.* s.l.:Wiley.

Joly, M., Verstraete, T. & Paniagua, G., 2010. *Attenuation of vane distortion in a transonic turbine using optimization strategies - Part I - Methodology.* Glasgow, UK, pp. 1-10.

Joly, M., Verstraete, T. & Paniagua, G., 2012. *Full design of a highly loaded fan by multi-objective optimization of through-flow and high-fidelity aero-mechanical performance.* Copenhagen, Denmark.

Konak, A., David, W. & Smith, A., 2006. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering and System Safety,* Volume 91, pp. 992-1007.

Lian, Y. & Liou, M., 2005. Multi-objective optimization of transonic compressor blade using evolutionary algorithm. *Journal of Propulsion and Power,* 21(6), pp. 979-987.

Manzan, M., Nobile, E., Pieri, S. & Pinto, F., 2008. Multi-objective optimization in convective heat transfer. In: *Optimization and computational fluid dynamics.* Berlin: Springer, pp. 217-266.

Massardo, A. & Satta, A., 1990. Axial flow compressor design optimization: Part I—Pitchline analysis and multivariable objective function influence. *Journal of Turbomachinery,* Volume 112, pp. 399-404.

Massardo, A., Satta, A. & Marini, M., 1990. Axial flow compressor optimization: Part II - Throughflow analysis. *Journal of Turbomachinery,* Volume 112, pp. 405-410.

Mengitsu, T. & Ghaly, W., 2008. Aerodynamic optimization of turbomachinery blades using evolutionary methods and ANN-based surrogate models. *Optimization and Engineering,* 9(3), pp. 239-255.

Metropolis, N., Rosenbluth, A., Rosenbluth, M. & Teller, A., 1953. Equation of state calculations by fast computing machines. *The journal of chemical physics,* 21(6), pp. 1086-1092.

Okui, H., Verstraete, T., Van den Braembussche, R. & Alsalihi, Z., 2011. *Three dimensional design and optimization of a transonic rotor in axial flow compressors.* Vancouver, British Columbia, Canada.

Öksüz, Ö., 2011. *Multiploid genetic algorithms for multi-objective turbine blade aerodynamic optimization (Ph.D. thesis).* Middle East Technical University.

Öksüz, Ö. & Akmandor, İ., 2005. *Turbine blade shape aerodynamic design using artificial intelligence.* Reno-Tahoe, Nevada, USA.

Papadimitrou, D., Zymaris, A. & Giannakoglou, K., 2005. *Discrete and continuous adjoint formulations for turbomachinery applications.* Munich, Germany
.

Park, K., Turner, M., Siddappaji, K. & Soumitr, D., 2011. *Optimization of a 3-stage booster Part I - The axissymmetric multi-disciplinary optimization approach to compressor design.* Vancouver, British Columbia, Canada.

Perez, R. & Behdinan, K., 2007. Particle swarm approach for structural design optimization. *Computers and Structures,* Volume 85, pp. 1579-1588.

Petrovic, M., Dulikravich, G. & Martin, T., 2001. Optimization of multistage turbines using a throughflow code. *Proceedings of the Institution of Mechanical Engineers, Journal of Power and Energy,* 215(A), pp. 559-569.

Pierret, S., 1999. *Designing turbomachinery blades by means of the function approximation concept based on artificial neural network, genetic algorithm, and the Navier-Stokes equations (Ph.D. thesis).* s.l.:von Karman Institute for Fluid Dynamics.

Pierret, S. & Hirsch, C., 2002. *An integrated optimization system forturbomachinery blade shape design.* Paris, France.

Pierret, S. & Van den Braembussche, R., 1999. Turbomachinery blade design using a Navier-Stokes solver and artificial neural network. *Journal of Turbomachinery,* Volume 121, pp. 326-332.

Rao, S., 1996. *Engineering optimization, theory and practice.* 3 ed. New York: John Wiley and Sons Inc.

Safari, A., Lemu, H., Jafari, S. & Assadi, M., 2013. A comparative analysis of nature inspired optimizaion approaches to 2D geometric modelling for turbomachinery applications. *Mathematical Problems in Engineering,* Volume 2013, pp. 1-15.

Samad, A. & Kim, K.-Y., 2008. Shape optimization of an axial compressor blade by multi-objective genetic algorithm. *Proc. IMechE Part A: J. Power and Energy,* Volume 222, pp. 599-611.

Shahpar, S., 2000. *A comparative study of optimisation methods or aerodynamic design of turbomachinery blades.* Munich, Germany.

Shahpar, S., 2012. *Optimisation strategies used in turbomachinery design from an industrial perspective / von Karman Institute for Fluid Dynamics Lecture Series.* Rhode St. Génese, Belgium.

Shi, Y. & Eberhart, R., 1998. *A modified particle swarm optimizer.* Anchorage, Alaska, USA, pp. 69-73.

Shi, Y. & Eberhart, R., 1998. *Parameter selection in particle swarm optimization.* Berlin, Germany, pp. 591-600.

Sivashanmugam, V., 2011. *Three dimensional aero-structural shape optimization of turbomachinery blades (M.Sc. thesis)*:Concordia University.

Storn, R. & Price, K., 1997. Differential evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces. *Journal of Global Optimization,* 11(4), pp. 341-359.

Tayfur, G., 2012. *Soft computing in water resources engineering.* 1 ed. s.l.:WIT Press.

Trelea, I., 2003. The particle swarm optimization algorithm, convergence analysis and parameter selection. *Information Processing Letters,* Volume 85, pp. 317-325.

Van den Braembussche, R., 2008. Numerical optimization for advanced turbomachinery. In: *Optimization and computational fluid dynamics.* Berlin: Springer.

Van den Braembussche, R., Alsalihi, Z. & Verstraete, T., 2004. *Fast multidisciplinary optimization of turbomachinery components / von Karman Institute for Fluid Dynamics Lecture Series.* Rhode St Génese, Belgium.

Vanderplaats, G., 1984. *Numerical Optimization Techniques for Engineering Design.* New York: McGraw-Hill.

Versraete, T., 2012. *Introduction to optimization and multidisciplinary design / von Karman Institute for Fluid Dynamics Course Note 217.*

Verstraete, T., 2008. *Multidisciplinary turbomachinery component optimization considering performance, stress and internal heat transfer (Ph.D. thesis).*von Karman Institute for Fluid Dynamics.

Verstraete, T., 2012. *Introduction to optimization and multidisciplinary design - Lecture notes.* Rhode St. Génese, Belgium: von Karman Institute for Fluid Dynamics.

Verstraete, T., 2012. *Introduction to optimization and multidisciplinary design in aeronatics and turbomachinery / von Karman Institute for Fluid Dynamics Lecture Series.* Rhode St. Genése, Belgium.

Verstraete, T., 2012. *Multidisciplinary optimization of turbomachinery components using differential evolution / von Karman Institute for Fluid Dynamics Course Notes 218.* Rhode St Génese, Belgium.

Wang, D. & He, L., 2010. Adjoint aerodynamic design optimization for blades in multistage turbomachines - Part I: Methodology and verification. *Journal of Turbomachinery*, Volume 132, pp. 1-14.

Wisler, D. et al., 1989. Chapter III: Fan and Compressor Systems. In: *Jet Engines and Propulsion systems For Engineers.* GE Aircraft Engines, pp. 3.1-3.74.

Zervogiannis, T. et al., 2001. *Inverse design of aerodynamic shapes using ant colony optimization.* Thessaloniki, Greece.

Zhaoyun, S., Liu, B., Xiaochen, M. & Xiaofeng, L., 2016. *Optimization of tandem blade based on improved particle swarm algorithm.* Seoul, Korea, pp. 1-12.

APPENDIX A

SIMULATED ANNEALING FLOW CHART

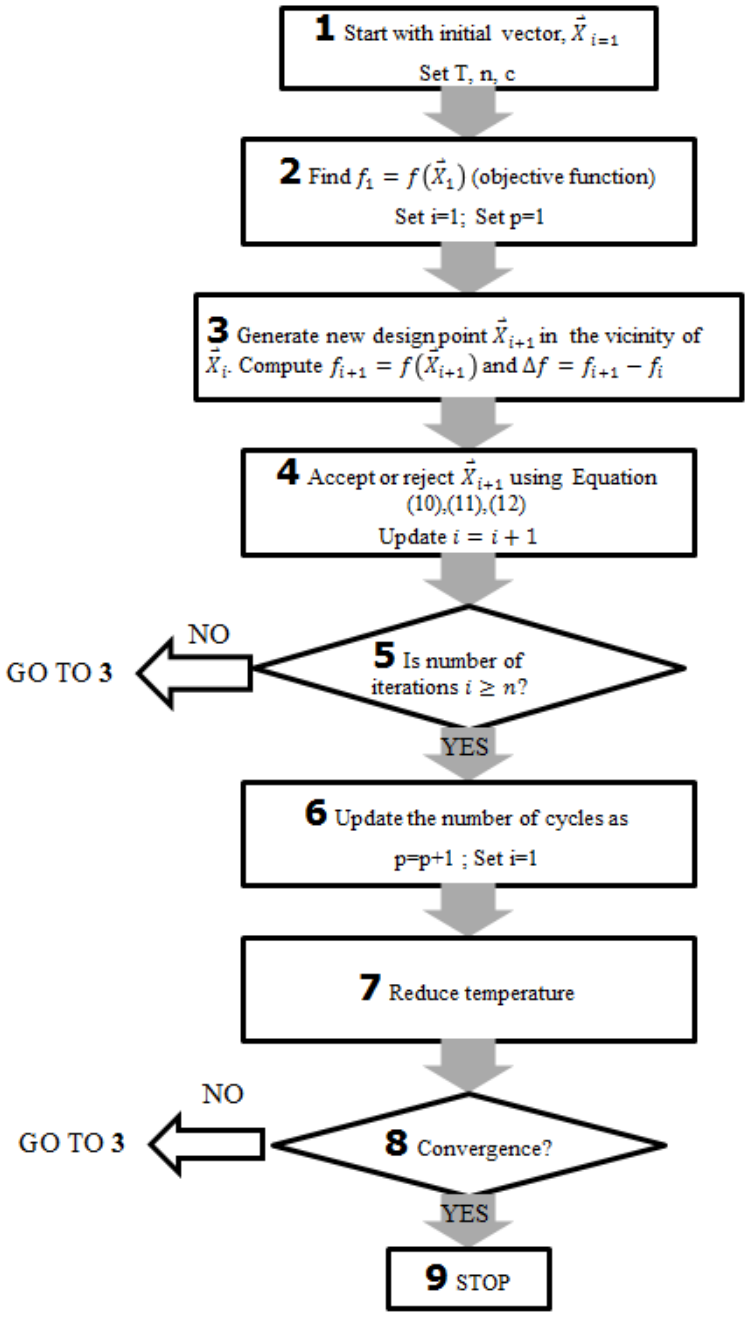


Figure A.1. Simulated Annealing flow chart

APPENDIX B

PENALTY FUNCTION METHOD CHART

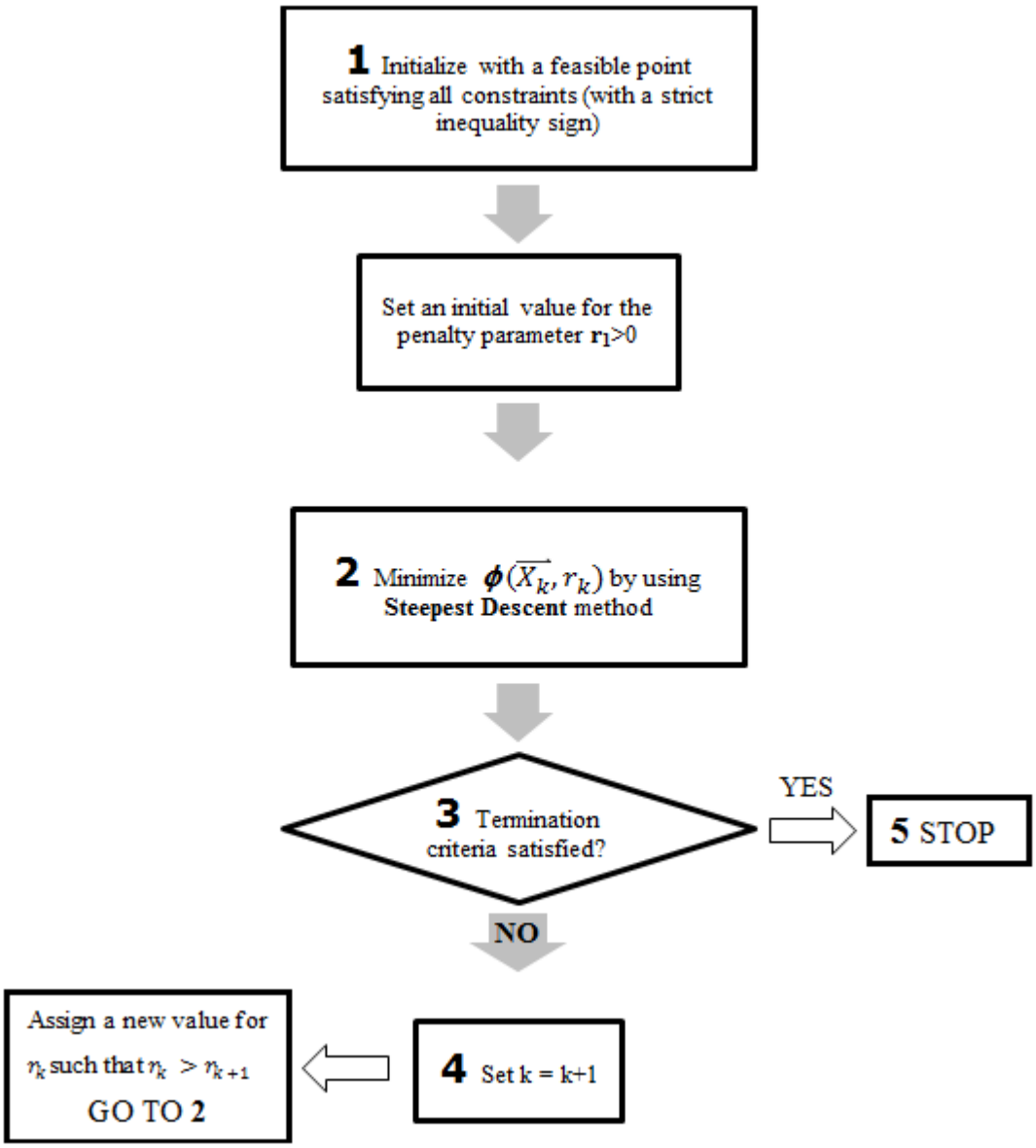


Figure B.1. Penalty function method flow chart

VITA

Orçun Kor was born in Izmir/Turkey in July 28, 1984. He graduated from Dokuz Eylül University, Mechanical Engineering Dept. in 2008. After working 1 year in the industry, he started his Ms.C. at İzmir Institute of Technology, Mechanical Engineering Dept. and attained his degree in 2011. Then, he commenced his Ph.D. as a research assistant at the same department. In the meantime, he attended the diploma course of the von Karman Institute for Fluid Dynamics in 2012, where he studied on compressor aerodynamics. There, he involved in a project dealing with the turbulence measurements using fast response aerodynamic pressure probes and hot wire anemometry. Back in Turkey, he left his research assistantship by September'12 and started working in TUSAŞ Engine Industries (TEI). Since then, he has been working in the company as a lead aerodynamics engineer.

Publications

- **Kor, O.**, Acarer. S. , Özkol, Ü (2016), "Aerodynamic Optimization of Through-flow Design Model of A High By-Pass Transonic Aero-Engine Fan Using Genetic Algorithm," (ongoing for a journal publication)
- **Kor, O.**, Acarer, S., Özkol, Ü. (2016), "Aerodynamic Optimization of a Transonic Fan Blade", National Aerospace Conference (UHUK), Kocaeli, Turkey (originally written in Turkish).
- **Kor, O.** (2015), "Optimization of a Turboshaft Engine Compressor's Return Channel Geometry", ANOVA User Community Conference, Ankara, Turkey (originally written in Turkish).
- **Kor, O.**, Malak, S., Özkol, Ü. (2012), "Effects of Acoustic Actuation Frequency and Nozzle Geometry on Heat Transfer and Flow Characteristics of an Impinging Confined Water Jet", 9th International Conference on Heat Transfer, Fluid Mechanics and Thermodynamics.