1-1-2002

Optimization of engineering systems using genetic algorithm enhanced computational techniques

Nicholas Farouk Ali
Ryerson University

Optimization of Engineering Systems using Genetic Algorithm enhanced Computational Techniques

by

Nicholas Farouk Ali
B. Eng., Ryerson University, 2001

A thesis

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

MASTERS OF APPLIED SCIENCE

in the Program of

Mechanical Engineering

Toronto, Ontario, Canada, 2002

Your file Votre référence
ISBN: 0-612-87147-9
Our file Notre référence
ISBN: 0-612-87147-9

Declaration

Title of Thesis: Optimization of Engineering Systems using Genetic Algorithm enhanced Computational Techniques

Author: ___________

Nicholas Ali

Sept. 25, 2002

Borrower's Page

Optimization of Engineering Systems using Genetic Algorithm enhanced Computational Techniques

Masters of Applied Science
Nicholas Ali
Department of Mechanical, and Industrial Engineering
Ryerson University

Abstract

The field of optimization has been and continues to be an area of significant importance in the industry. From financial, industrial, social and any other sector conceivable, people are interested in improving the scheme of existing methodologies and products and/or in creating new ideas. Due to the growing need for humans to improve their lives and add efficiency to a system, optimization has been and still is an area of active research. Typically optimization methods seek to improve rather than create new ideas. However, the ability of optimization methods to mold new ideas should not be ruled out, since optimized solutions usually lead to new designs, which are in most cases unique. Combinatorial optimization is the term used to define the method of finding the best sequence or combination of variables or elements in a large complex system in order to attain a particular objective.

This thesis promises to provide a panoramic view of optimization in general before zooming into a specific artificial intelligence technique in optimization. Detailed information on optimization techniques commonly used in mechanical engineering is first provided to ensure a clear understanding of the thesis. Moreover, the thesis highlights the differences and similarities, advantages and disadvantages of these techniques. After a brief study of the techniques entailed in optimization, an artificial intelligence algorithm, namely genetic algorithm, was selected, developed, improved and later applied to a wide variety of mechanical engineering problems. Ample examples from various fields of engineering are provided to illustrate the versatility of genetic algorithms.

The major focus of this thesis is therefore the application of genetic algorithms to solve a broad range of engineering problems. The viability of the genetic algorithm (GA) as an optimization tool for mechanical engineering applications is assessed and discussed. Comparison between GA generated results and results found in the literature are presented when possible to underscore the power of GA to solve problems. Moreover, the disadvantages and advantages of the genetic algorithms are discussed based on the results obtained. The mechanical engineering applications studied include conceptual aircraft design, design of truss structures under various constraints and loading conditions, and armour design using established penetration analytical models.

Results show that the genetic algorithm developed is capable of handling a wide range of problems, is an efficient cost effective tool, and often provides superior results when compared to other optimization methods found in the literature.

Acknowledgements

The author would like to extend sincere thanks to his supervisor and co-supervisor for their guidance and support throughout this research. I would also like to extend my gratitude to my peers who instilled good habits and constantly encouraged me to push on with my research ventures. Many thanks to my friends who provided a balance to my life and fostered the passion that they claim is the ingredient to my success.

Lastly, I wish to extend my dearest thanks to my family for their understanding and patience. To them I am indebted.

To Margaret
without her ambition, love and sacrifice
this
and much else
would not be possible.

To Lisa and Catherine
for their unwavering support and Love.

To Passion
out of whose comfort
neither this thesis
nor I
would have been created.

Table of Contents

List of Figures

List of Tables

Nomenclature

This list defines the symbols used throughout this thesis. Symbols not defined here are defined as they appear in the text.

pc	probability of crossover
pm	probability of crossover
gen#	generation/iteration number
pop_size	population size
h1/h2	ratio of front plate to back plate thickness for armour
tot_thk	total thickness of the armour
h1	thickness of front plate
h2	thickness of back plate
mat1	material index for front plate
mat2	material index for back plate
W_0	gross weight (lbs.)
C_r	wing root chord (ft)
C_t	wing tip chord (ft)
b	wing span (ft)
S	gross wing area (ft^2)
S_{ht}	horizontal tail area (ft^2)
l_{fus}	length of the fuselage (ft)
S_t	gross tail area (ft^2)
V_h	horizontal tail volume coefficient
V_v	vertical tail volume coefficient
F	stick force (lb.)

$C_{m\alpha}$	change of pitching moment coefficient with angle of attack
C_{mo}	coefficient of moment at zero angle of attack
$(X_{cg}/\bar{C})_{for}$	forward centre of gravity limit
$(X_{cg}/\bar{C})_{aft}$	aft centre of gravity limit
$n_{ult.}$	ultimate load factor
a	$a_p + 2h_1$
a_p	radius of the projectile's armour-piercing (AP) core
d_1	density of the ceramic tile
d_2	density of the backing plate
h_1	thickness of the ceramic tile
h_2	thickness of the backing plate
m_p	mass of the projectile
V_p	the predicted value of the ballistic limit
V_i	Assumed impact velocity
V_r	Residual velocity
T	total thickness of armour
h1/h2	thickness ratio of front plate to back plate of armour
ε_2	breaking strain of the backing plate
σ_2	the ultimate tensile strength of the backing plate
ρ_1	density of the ceramic tile
ρ_2	density of the backing plate

Acronyms

OR	Operations research
AI	Artificial intelligent
GAs	Genetic algorithms

SA	Simulated annealing
TS	Tabu search
ANN	Artificial Neural Networks
EOQ	Economic order quantity
FEA	Finite Element Analysis
APDL	ANSYS parametric design language
GFRP	Glass Fibre Reinforced Plastic
FSP	Fragment simulating projectile
AP	Armour piercing
UTS	Ultimate tensile strength
1D	one dimensional
2D	two dimensional
3D	three dimensional

Chapter 1

Introduction

1.1 Categories of Optimization

Optimization can generally be branched into two broad groups, namely, operation research methods and artificial intelligent or heuristic methods. Since this thesis is highly application based, this chapter presents an overview of optimization. Against this background, we set a stage for machine learning strategy of genetic algorithms as a technique for solving engineering problems. Terminology on the topic is kept simple so as to facilitate easy reading and understanding of the material. In later chapters, when the optimizer is applied to an engineering problem, a separate introduction is given to introduce previous but similar applications to the problem.

One major branch of optimization is operations research (OR). OR is concerned with the quantitative specifications of problems and the use of mathematical techniques to solve these problems. The latter can be achieved with the aid of a computer. On the other hand, the other branch of optimization, that is, artificial intelligence (AI), is concerned with making computers capable of emulating intelligent behavior [1]. Since the ability to solve problems is one of the hallmarks of human intelligence, a major AI focus is on computer-based techniques for solving problems [2]. Moreover, OR has found usefulness in decision-making, while AI pursuit is in automatic decision-making.

It must be noted that both OR and AI have approached the issue of problem solving from decidedly different angles. Whereas OR has emphasized the quantitative, AI has tended to stress the qualitative aspects of problems. Moreover, OR approaches have tended to be calculational and algorithmic, while AI solution methods tend to rely on logic and inference. Both OR and AI involve the use of descriptive knowledge (i.e. data and information). Beyond this however, OR focuses on the management of procedural knowledge whereas AI tends to focus on the management of reasoning knowledge.

Although it is not the focus of this thesis, in general there exists a common ground between OR and AI. Historically, the OR and AI research communities worked in relative isolation from one another. On the one hand, this separation is difficult to understand, since both disciplines share many similarities deeply concerned with questions of human problem solving and decision-making. In addition, both are highly computer dependent, and finally both share a common conceptual framework. However, despite the similarities these two disciplines are highly distinct from each other as explained above. Despite their differences, OR and AI are united in one major effort to solve real world problems. One important aspect in which this union can be useful -and has been fruitful- is in the decision support system (DSS) arena.

Nonetheless, the complexity amidst real world problems has exposed the limitations of both disciplines and has forced researchers to look more deeply into improving the mechanics of the search and optimization algorithm of both methods, or create hybrid systems that integrate OR and AI tools. Perhaps, the most significant call for integrative approaches came from Simon [3], who stressed the common problem solving foundations of both fields. Simon also stressed that the pitfall in either methods can be found in the strength of the other methods. Thus integration was argued then as the only way to solve the problems encountered with either OR or AI approaches. Consequently, some development in OR/AI interface has emerged. Both OR and AI require methods that effectively cope with uncertainty and imprecision. For details on handling these concepts in both approaches, please consult references 4 and 5.

1.1.1 History and Objective

Due to the vast nature of optimization, this thesis primarily focuses on one technique on the branch of optimization that umbrellas AI approach. Nonetheless, general descriptive information on both OR and AI techniques are presented to underscore the scope of optimization and further facilitate ease of reading and understanding of the thesis. Thus a panoramic view of optimization is initially presented, and then the study zooms in on one particular AI method that is the backbone of this thesis. The AI method investigated is genetic algorithms (GAs). Genetic algorithms are a type of search algorithm for finding optimal solutions to computationally difficult problems, and are based on analogies to biological reproductive processes [6]. The basic ideas of genetic algorithms were developed by Holland [7] while investigating how to build intelligent machines that have the capability of learning. After the publication of Holland's work in 1975 the term 'genetic algorithm" gained popularity mainly because of its success in applications such as job-shop scheduling, pipeline systems and vehicle routing, just to name a few. Today, it can be argued that genetic algorithms are some of the most robust evolutionary algorithms in use. Furthermore, genetic algorithms have found extensive application in the field of research and development. The objective of this thesis is the investigation of genetic algorithms application to mechanical engineering problems.

The application of genetic algorithms to an engineering problem studied in this thesis is preceded by its own introduction. The goal of this thesis is to highlight some successful and important applications of genetic algorithms in the engineering arena. Some of these applications that is pertinent to this thesis is underscored here and is repeated in later chapters for completeness. Quagliarella and Cioppa [8]; Perez et al. [9]; Anderson and Gebert [10]; Marx et al. [11]; Ewing and Downs [12]; Tse and Chan [13]; Butler et al. [14]; and Gage et al.[15], were all successful in applying GA to study different design aspects of the aircraft wing. Tong et al. [16]; Mosetti and Poloni [17]; and Hajela [18] were also instrumental in applying the GA to study complex design problems in the aerospace industry. Furthermore, Crossley and Laananen [19] were successful in applying the GA to study the conceptual design of helicopters. In addition, Sobieski et al.[20]; Gallman and

Mason [21]; Malone and Mason [22]; Crispin [23]; and Bos [24] were all able to utilize the GA to study aircraft design.

Goldberg [25], – a pioneer in the work of GA- and Samtani appeared to be the first that employed the GA for structural optimization. A 10 bar truss problem – which is today a benchmark problem in structural optimization - was utilized to study the GA's role in structural optimization. Jenkins [26] utilized a simple GA created by Goldberg [27] to study a truss-beam roof structure. Hajela and Shih [28] looked into the applicability of GA as a function optimizer for structural optimization. Rajeev and Krishnamoorty [29] proposed a simple discrete GA for optimizing truss structures. Rajan [30] extended the magnitude of the problem by combining size, shape and topology optimization of trusses using a modified GA. Deb and Gulati [31] also proposed a GA for size, shape and topology optimization using a real-coded GA. Camp et al. [32], looked into a simple genetic algorithm for discrete optimization of 2D structures using a finite element analysis program developed by Zienchawicz.

To the author's knowledge, Florence [33] appeared to be the first to present a simple analytical model to analyze the impact onto a ceramic/composite armour. Florence model – based on the ideas of Wilkins [34-37]- was used to estimate the ballistic limit. Studies conducted by Prior [38], Gage [39] and Rajagopalan [40] have found satisfactory agreement between experimental results and the theoretical predictions by Florence models. Hetherington [41] later revised Florence model to account for the energy absorbing capabilities of the armour. Woodward [42] later proposed a 1D model for the perforation of a ceramic armour using a lumped mass approach, which takes into account the erosion of both projectile and target. Many of the ideas in Woodward's model were adapted from previous work of Wilkins, Florence and Hetherington. Den Reijer's model [43] is the most recent analytical model found in the literature today. The model is a little more sophisticated than Woodward's model in that it takes into account projectile mushrooming effects. A different approach to armour design was investigated by Ben Dor et al. [44], whose objective function was geared towards armour weight minimization.

Hence, this thesis focuses on the use of a non-traditional optimization method (genetic algorithms) to solve some engineering problems. Most of the applications studied in this thesis entailed the use of imperial units mainly because many publications studied utilized such units which, facilitated comparison and validation of the results. The viability of this method to the particular mechanical engineering application is discussed. Comparison between GA generated results and results found in the literature are presented when possible to underscore the power of GAs to solve mechanical engineering problems. Moreover, the disadvantages and advantages of the genetic algorithms are discussed based on the results obtained.

This study illustrates that for small search spaces, classical exhaustive OR methods usually suffice; however, for large search spaces - which are an inherent property of large complex problems - artificial intelligence techniques must be employed. The randomness of the GA, coupled with its capabilities of learning, offers a somewhat new and promising approach to traditional design techniques. Results indicate that genetic algorithms are capable of solving a wide variety of mechanical engineering problems. From the results we can also glean that GA generated results are reasonable and practical for the engineering applications studied. More importantly, GA can be used as a search and optimization tool that can save significant costs and time in the industry. The thesis concludes with some discussion on the applicability of GAs to solve mechanical engineering problems and some thoughts on unresolved issues.

1.2 Thesis Organization

Chapter 2 briefly highlights some of the methods common to problem solving in mechanical engineering. Both operations research methods and artificial intelligent methods of optimization are presented in order to paint a global picture of optimization.
Chapter 3 focuses on an AI method of optimization, namely, genetic algorithms. Genetic algorithms from Chapter 3 onwards, become the main theme of this thesis. In chapter 3, detailed development of genetic algorithms are presented, along with methods and schemes to improve the basic genetic algorithm technique. Also the method of implementing problems into the genetic algorithm is also underscored.

Chapter 4 marks the beginning of the genetic algorithm application to problem solving. The applications in this chapter are kept simple so as to test the correctness of functionality of the genetic algorithm. Moreover, chapter 4 serves as a springboard for application and integration of real world problems to the genetic algorithm created.

The contents of chapter 5 are centered on the conceptual design of aircrafts using genetic algorithms. Genetic algorithms are utilized to reduce the lengthy time and effort spent creating and integrating aerodynamics codes, sizing routines and performance modules, entailed in a typical aircraft conceptual design process.
Chapter 6 is a study of the use of genetic algorithm to study structures. In this chapter, the genetic algorithm is integrated into a finite element software so as to exchange information for design. This integration paved the way for optimization of any structure created within the finite element software. Moreover, the integration saved significant amount of time by utilizing already existing finite element tools.

In Chapter 7, genetic algorithm is employed to study, analyze and optimize armours against ballistic impact. The analysis is strictly the fusion of ballistic analytical models into the genetic algorithms for optimization. Initially, the ballistic analytical model is used by itself in order to ensure correctness in programming and concepts. Having achieved the latter, integration of the model into the genetic algorithm is conducted. Various case studies are conducted to test the validity of the problem formulation and definition within the optimizer.

Chapter 8 provides general conclusions about the use of genetic algorithm for optimizing and designing in mechanical engineering. In addition, some suggestions for future work are presented to extend this current research venture.

Chapter 2

Panoramic View of most Techniques Entailed in Engineering Optimization

Most optimization techniques in engineering can be classified as either an operation research (OR) approach or an artificial intelligent (AI) approach. The prime objective of this chapter is to illustrate the importance of OR approaches and AI approaches in engineering optimization. The chapter will later allude to some of the techniques covered in the two different approaches and then compare these techniques under performance and robustness criteria. The chapter also highlights the area in which there is a common ground for OR and AI approaches. More importantly we learn why an AI method was chosen as an optimizer for all the applications studied in later chapters.

2.1 An OR Research Perspective

From an OR perspective there is specific problem and there is a specific procedure for solving those problems (see Table 2.1 adapted from Ref. 1). A procedure is a step-by-step specification of how to accomplish some task. It is the wording or problem statement that helps us to decide which OR method to use. Hence a given procedure requires a certain type of problem statement as a precondition to execution. This to some extent has made OR procedures unattractive for solving complex problems.

Table 2.1. OR problems and corresponding solution method

Examples of OR problems	**Corresponding Procedure**
Minimize some quantity subject to certain constraints being satisfied	Mathematical programming algorithms
Determine the quantity of an item (having certain demand, holding cost, and order cost)	EOQ formulas
What is the earliest or latest starting time for some activity?	Network analysis algorithms
What is the expected return for a particular investment?	Expected value calculations
Determine a forecasting function based on historic data	Regression algorithms
Forecast a quantity	Forecasting functions
How many servers should be in place?	Queuing theory methods
What are the impacts of altering a system in particular ways	Simulation algorithms

2.2 An AI Research Perspective

From an AI viewpoint, "computer systems can be imbued with behaviors that would be regarded as intelligent if witnessed in a human"[1]. AI research is centered on making progress in emulating some intelligent human behavior. Some practical progress aimed at emulating human behavior has been made (see Ref. 45). In addition, AI focuses on discovering new ideas, techniques and languages for developing computer systems that exhibit increasing degree of intelligence. Some prominent achievements in AI research include language understanding, pattern recognition, automatic programming, knowledge representation schemes, reasoning systems, learning and robotics.

2.3 Some OR and AI methods employed in mechanical engineering applications

In recent years the field of optimization has become an area of active research -especially in engineering – so as to improve, and to a lesser extent create new ideas in the field of human endeavors. Several other reasons for this interest includes the need to handle a wider class of problems, to include realistic definitions of design variables, to find techniques to improve the efficiency of the numerical procedure [30]. Optimization methods found in the literature vary widely and thus in order to draw a clear comparison between OR and AI approaches, some of these methods are briefly covered for completeness. Consequently, this chapter addresses the techniques involved in the OR and AI approaches, while chapter 3 focuses on a specific technique employed in this study.

2.3.1 The Common Process for all Optimization Techniques

All optimization methods share a common process for finding an optimum solution: search and improvement [46]. This improvement is towards a certain objective. Hence, all methods have an objective and an objective function that defines how the objective is measured. The objective function formulates the objective in terms of the parameters and variables involved in the optimization problem. The search mechanism, the objective, and the objective function may vary from method to method to suit the optimization problem.
Some of the search methods ranging from exhaustive search to more efficient (and more complicated) searches are highlighted in a latter section of this chapter.

2.3.2 Optimization defined and general model presented

Any problem in which design parameters have to be determined, (assuming such parameters exist), given certain constraints, can be treated as an optimization problem [6]. The first step in optimization is to identify a set of parameters called the design parameters, a cost or objective function to be minimized or maximized and the problem constraints. In the case of minimization a lower cost is indicative of a better solution. The entire goal of any optimization routine is to continually find a method of improving the value of the objective function. Restrictions on the objective function are called the constraints. Constraints show the value that the design variables cannot have and they often help guide the search towards feasible region in the search space. Without constraints, most real world problems become meaningless. Constraints can be represented by inequalities while others, by equalities. A typical design optimization flow chart adapted from Ref. 6 is shown in Figure 2.3.1.

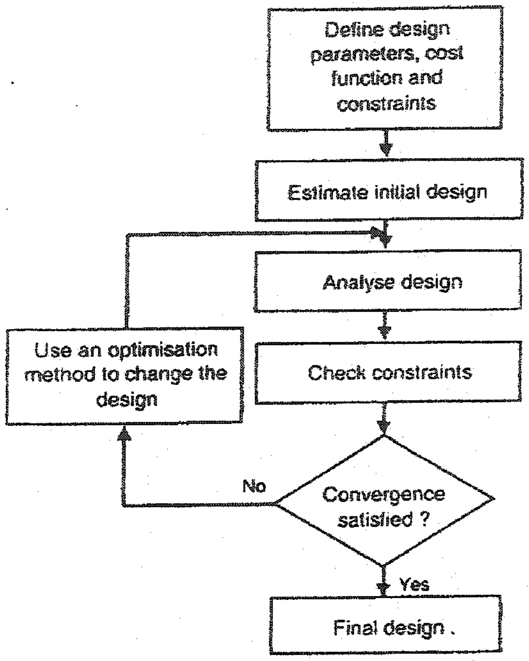

Define design parameters, cost function and constraints
Estimate initial design
Analyse design
Use an optimisation method to change the design
Check constraints
No
Convergence satisfied ?
Yes
Final design .

Figure 2.3.1: Design Optimization Process

A general mathematical form for typical optimization model (as adapted from Ref. 2) is as follow:

Find the vector $x = (x_1, x_2, ... x_n)$ with n components, where x_i represents the value of the parameter.

$$f(x) = f(x_1, x_2 ... x_n)$$

subjected to p equality constraints:

$$h_j(x) = h_j(x_1, x_2, ... x_n) = 0 \quad 1 \le j \le p \text{ and}$$

m inequality constraints

$$g_i(x) = g_i(x_1, x_2, ... x_n) \le 0 \quad 0 \le i \le m$$

A feasible solution is one that satisfies all constraints. For some optimization algorithms finding a feasible solution does not mean finding the global optimum. Thus researchers have spent much time and effort looking at ways to prove that a global optimum has been found. These methods are usually mathematical based and lean more towards operations research (OR) procedures as opposed to artificial intelligence (AI) methods. Artificial intelligence methods have no proof of optimality since these methods are stochastic.

2.4 Operations Research Optimization Techniques

Operations research (OR) is concerned with the quantitative specifications of problems and the use of the mathematical techniques to solve these problems. The minimization or maximization of a function $f(x)$ without any constraints on x is called unconstrained optimization. Introduction of constraints into an optimization problem is called constrained optimization. If an optimization problem has a linear cost or objective function it is called linear programming problem (LPP). An integer-programming problem (IPP) is a linear programming problem in which the design variables are restricted to be nonnegative integers only. Otherwise the problem is called non-integer programming problem. A problem that entails a quadratic objective function subjected to linear constraints is called a quadratic programming problem (QPP) [47].

2.4.1 Unconstrained Optimization Methods in Operations Research

The optimization of certain objective functions without any constraints is called an unconstrained optimization method. Some of these methods are presented below:

2.4.1.1 Direct Search Methods

Optimization methods that do not use derivative information are called direct search methods. Direct search methods require only the computation of the objective function value. Since finding the derivative can be difficult and in some cases impossible, this presents one immediate disadvantage of this method. There are three main groups of direct search methods, namely: tabular, sequential and linear methods. For complete details on these methods please consult Ref. 48.

2.4.1.2 Gradient–Based Methods

Gradient-based methods utilize the first order or higher order derivative of the objective function to determine a search direction. There are three main strategies that umbrella gradient-based methods: the method of steepest descent, the Newton Ralphson method and the Conjugate-Newton technique. For complete details on these three methods consult Adby and Dempster [49].

Both direct search methods and gradient-based procedures can be broken down into two phases: (1) the search direction and, (2) the step size determination. As mentioned before gradient information enables these optimization methods to check when an optimum is obtained and find the search direction. The step size can be determined by employing either of three algorithms: internal search, golden section search or polynomial interpolation. The central idea behind these methods is to successively reduce the interval of uncertainty to a small acceptable value. For details about these methods consult Ref. [49].

2.4.2 Constrained Optimization Methods in Operations Research

The use of constraints in an optimization problem is called constrained optimization. These methods are more popular than unconstrained optimization methods, mainly because most practical problems involve some restraints on the problem.

2.4.2.1 Simplex Method and Goal Programming

Like unconstrained optimization there are many techniques available for solving constrained optimization problems. The most well known method is the simplex method used to solve linear programming problems. This method requires that the objective function and its constraints be linear functions of the decision variables. It is an extension of the standard Gauss Jordan elimination process for solving a set of linear equations. In some situations, there may be multiple objectives and no feasible solution satisfying all objectives. In such a case goal programming techniques must be used. For complete details and implementation of the simplex method and goal programming please consult [50,51].

2.4.2.2 Integer Programming

Integer programming is similar to linear programming, with the one additional restriction that the variables must be integer values [50]. Variations of integer programming include mixed integer programming and binary integer programming, which incorporate greater flexibility for problem solving. However, most integer programming problems are solved by using bound techniques. Branch and bound methods find the optimal solution to an integer-programming problem by efficiently enumerating the points in a feasible region.

2.4.2.3 Other Constrained Optimization Methods

In most constrained optimization methods Taylor series expansion is used to linearize non-linear problems. This linearized sub-problem can usually be transformed into a quadratic programming problem, which can then be solved by an extension of the simplex method. Two well-known methods for solving constrained problems are constrained steepest descent and quasi-Newton method [52].

Exhaustive enumeration and random search methods are among the simplest and most robust non-gradient methods that can be used in searching for an optimal design. These methods can be applied to a wide range of problems and are not limited by discontinuity in the search space, as is the case for mathematical programming techniques.

Zero order methods in mathematical programming such as Nelder mead simplex method, complex method of box, the Hookes-Jeeves pattern search, Rosenbrock's method and Powell's conjugate direction method are also not limited by discontinuity in the search space and can be applied to a wide range of problems.

Recently, computational paradigms that are philosophically based on ideas such as natural evolution and the laws of thermodynamics use random sampling to guide the search process. These paradigms are of significant importance and thus some of these methods are presented next.

2.5 Artificial Intelligent Optimization Techniques

Researchers faced with combinatorial optimization problems have preferred to employ AI techniques. Combinatorial optimization involves determining how best to arrange (group, sequence, or assign) the controllable elements in large complex systems to achieve a specified objective or goal [2]. For example, the popular knapsack problem is one in which a hiker considers which of n objects to pack into a knapsack. Each object has a certain weight and value. The objective is to select a subset of objects that have the greatest combined value and whose total weight does not exceed the capacity of the knapsack. Both OR and AI methods has attempted this problem as a way of testing the robustness of the algorithms. Combinatorial optimization has been utilized to optimize problems as diverse as vehicle routing, airfoil shape optimization, manufacturing plant layout and even portfolio selection.

An algorithm is a sequence of operations that can be carried out in a finite amount of time [2]. An algorithm prescribes a process that may be repetitive in some sense (perhaps iterative or recursive) but that will eventually terminate against some test for optimality. Some examples of algorithms include cooking recipes, instructions for setting up a stereo system and a computer program that does not contain infinite loops.

With this in mind this section underscores some of the fundamental concepts of artificial intelligent algorithms. However, before this is done the reasons for engineers and scientist acceptance of near-optimal or sub-optimal solutions is presented. In addition, an overview of the types of problems these algorithms is best suited for and the reasons why even these problems cannot be solved accurately by these algorithms is also addressed.

2.5.1 Why accept near optimal or sub-optimal solutions

There are some strong and justifiable reasons for accepting computational results that are imperfect or sub-optimal. Firstly, models created by researchers are not perfect representations of a system being analyzed. Hence, even if we could obtain exact solutions to the problem, it would not reflect exact solutions to be applied to the real system. Hence costly computational effort to achieve perfect solutions may not be warranted.

Secondly, when solving problems the exact representation of real numbers require the use of many binary digits. However, the finite number of bits typically used for storing numerical values in computer memory implies that real numeric data cannot always be represented exactly in computers [53]. The backdrop of this is round-off errors and after repeated iterative calculations in an algorithm involving many inexact values the result is accumulated round-off errors.

Finally, although the algorithm underlying a solution process might be proven to yield optimal results, ill-conditioned problem data and numerical instability can practically preclude obtaining solutions of any reasonable quality [2].

The innate difficulty of some problems might suggest that accepting sub-optimal solutions is the only practical approach. In concluding, in the real world of imperfect models, precarious data, unavoidable numerical inaccuracies and time constraints, insisting upon so-called optimal solutions may be entirely inappropriate.

2.5.1.1 Optimality vs. Practicality

For many practical problems in engineering, the only way to be sure of finding an optimal solution is to search completely through the whole set of possible solutions. If there exist a very large but finite number of possible solutions, the idea of complete search is tempting and often quite easy to implement into a computer program. The problem however is that the time required to carry out such an exhaustive search is, although finite, far greater than most institutions can afford. The pretext then is to try and find short-cuts that will allow us to organize the search process so that it is no longer a complete search over all possible solutions, but rather it becomes an affordable search that is likely to find optimal or near optimal solutions [54]. These methods are called heuristic methods or artificial intelligent techniques. Heuristic methods are often applied to computationally intractable NP-hard problems, simply because otherwise the best (most efficient) methods we know of for solving these problems exactly (or optimally) can take an exponential amount of computational time [54]. Four optimization techniques that are considered intelligent and that are constantly being researched, utilized and share a wide range of applications are described below. These methods are Tabu search (TS), simulated annealing (SA), genetic algorithms (GA) and neural networks (NN). This chapter provides only a concise definition of these methods since it is not the main focus of this thesis. In later chapters, genetic algorithms, its detail development and implementation, and finally its applications to mechanical engineering problems are undertaken as the primary objective.

2.5.2 Tabu Search

The tabu search (TS) algorithm was developed independently by Glover [55] and Hansen [56] for solving combinatorial optimization problems. Some researchers see tabu search as advancement over rule-based expert systems. It is an iterative process that functions by the use of flexible memory. Tabu search was partly motivated by the observation that human behavior appears to operate with a random element that leads to inconsistent behavior given similar circumstances. As Glover pointed out, the resulting tendency to deviate from a charted course might be regretted as a source of errors but can prove to be a source of gain. The tabu search operates in this way, with the exception that new courses are not chosen randomly.

Tabu search is capable of escaping from local minima and to search in areas beyond a local minimum. Figure 2.5.1 shows the flow of control for a typical tabu search algorithm adapted from Ref. 6. Hence, it has the ability to find the global minimum of a multimodal search space. A tabu list is employed to store the characteristics of accepted moves so that these characteristics can be used to classify certain moves as tabu (i.e. to be avoided) in later iterations [6].

A tabu list is a set of constraints that impose restrictions on the search. So for example, a search that yields a solution that is already present in the tabu list is avoided. Note that if a move in the tabu list yields a better solution than the one encountered before, it is accepted. This is based on the best solution criterion. This means that the tabu list is not completely binding. Instead, moves in the list are allowed if the aspiration criteria allow it. Aspiration criteria can be based on best solutions. However, other aspiration criteria based on objectives, search direction, influence etc. can be found in some previous work [55, 57].

A simple tabu search algorithm consists of three main strategies: forbidding strategy, freeing strategy and short term strategy [55, 56].

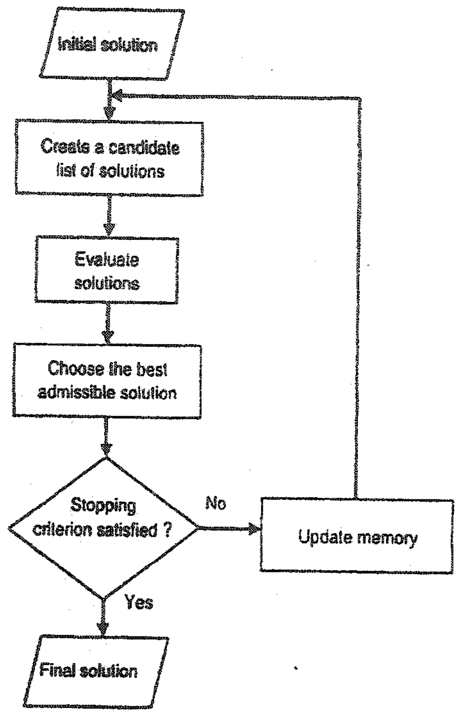

Initial solution
Create a candidate list of solutions
Evaluate solutions
Choose the best admissible solution
Stopping criterion satisfied ?
No
Update memory
Yes
Final solution

Figure 2.5.1: Flowchart of a standard Tabu search algorithm

The forbidding strategy controls what enters the tabu list. The freeing strategy controls what exits the tabu list and when. Finally the short-term strategy manages the interplay between the forbidding and freeing strategies to select the trial solutions. Apart from these three strategies, a learning strategy, which collects information during a tabu search run is usually employed and this information is later used to direct the search in subsequent runs specified by the user. The number of generations specified by the user is usually used as the stopping criteria to terminate a tabu search procedure. Tabu search is relatively new to the engineering optimization field. Hence it is difficult to find a single efficient and acceptable implementation of the algorithm. Nonetheless, TS, like GA and SA attempts to overcome the problem of relative optima. For greater depth into this heuristic search method consult references 2 and 6.

Clearly the key to this algorithm is tabu list management. To recap, tabu list management concerns updating the tabu list, i.e. deciding on how many and which moves have to be set tabu within any iteration of the search.

In concluding the main aim of a tabu search is to avoid entrapment in cycles by forbidding or penalizing moves that take the solution in the next iteration to points in the solution space previously visited.

2.5.3 Simulated Annealing

Before simulated annealing, the discipline of operations research (OR) attempted to understand the formal properties of algorithms and the inherent complexity of problems [58]. However, in the 1970s and 1980 tremendous advancements was made in the arena of optimization. At that time two classes of problems was defined. One class called the P class involved problems that can be solved by an algorithm within an amount of computational time proportional to some polynomial function of the problem size.

The other class (called class NP) contains problems that may require computational time proportional to exponential (or larger) function of problem size. For detailed description of these problem classes' consult references 59-61.

Within the class of NP there is another special class of problems called NP complete or NP hard problems. These problems are known to be the most difficult problems in NP. It is here that simulated annealing capabilities have been utilized and given much attention by researchers and engineers to solve many practical problems that fall under the umbrella of NP hard.

Local heuristic techniques for optimization usually fall into the trap of converging to a local optimum. This phenomenon is a natural consequence of computational process that moves monotonically in an improving direction, from an arbitrary starting point [2]. Simulated annealing is a local improvement probabilistic mechanism, in which non-improving moves are occasionally allowed and therefore offers a chance for the algorithm to escape from local optima. Simulated annealing is based on the concepts of thermodynamics, which involves the behavior of a liquid substance as it is slowly cooled into a solid to produce a stronger more stable substance.

Moreover, simulated annealing algorithm was derived from statistical mechanics and is motivated by the analogy to the behavior of the physical annealing process. In 1983

Kirpatrick et al. [62] restated this simulated annealing procedure developed by Metropolis et al. [63] in order to make it applicable to optimization.

Annealing is the process of heating up a solid and then cooling it down slowly until it crystallizes. At high temperatures the atoms in a solid have high energies and thus move about freely. As the temperature is lowered the atomic energies decrease. A crystal with regular structure is obtained at the state where the system has minimum energy. At a given temperature, the probability distribution of the system is determined by Boltzman probability and is adapted from Ref. 6 as follow:

$$P(r_i) = \exp\{-E(r_i)/KT\} \qquad (1)$$

where each configuration r_i belongs to the set of all possible atomic configurations.

$P(r_i)$ = probability of a configuration r_i

$E(r_i)$ = energy (in joules) of the system in configuration r_i

K = Boltzmann constant (in joules per degree Kelvin)

T = temperature in degrees Kelvin

A plot of this curve is shown Figure 2.5.2 that was adapted from Ref. 6. Note as the temperature approaches a very low value, the probability of the occurrence of a new configuration approaches zero since the system is in a nearly stable state. Hence, at low temperatures the exponent becomes very large and negative, and thus $P(r_i)$ approaches zero. On the other hand, at higher temperatures there is more atomic movement within the substance, hence more different configurations occur, and therefore the probability of occurrence of any given r_i becomes greater as temperature increases.

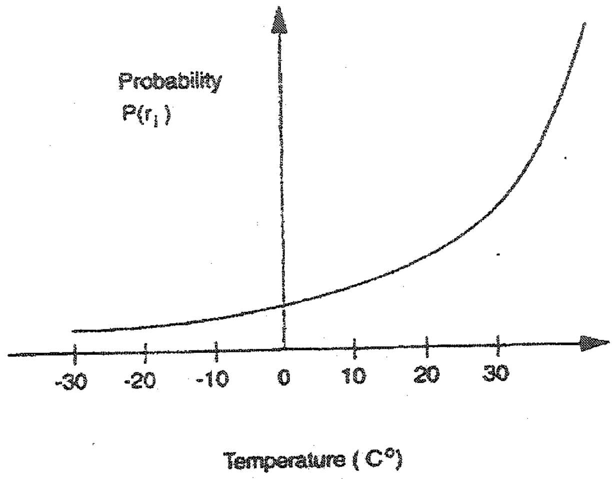


Figure 2.5.2: Boltzmann Distribution Probability

In the context mentioned above a "configuration" means assignment of some value to the decision variables. The temperature is indicated by a simpler parameter called θ. A sequence of classes of configurations is then generated. Within each class a parameter θ determines the magnitude of the objective function value fluctuation that occurs within that class. Each class is asymptotically distributed as a Boltzmann distribution. The process of determining the distribution for any given value of θ is called equilibration. The optimization process is comprised of a series of equilibration; each equilibration is associated with a temperature parameter θ, and the equilibration is done at successively lower temperatures [6].

Each equilibration begins with the system in some initial configuration and then iteratively carries out the following process until a stable state is reached:

Generate a new configuration arbitrarily
Calculate ΔF = new objective function value – current objective function value
If $\Delta F \leq 0$, then accept new configuration unconditionally
If $\Delta F \geq 0$, then accept new configuration only with probability exp $(-\Delta F / \theta)$

Equilibrations are carried out until it is observed that practically no configurations are being generated (and accepted) that have a better (lower) objective function value than the current configuration (i.e. until the "acceptance ratio" is essentially zero). At this stage the optimization process is complete and best configuration observed so far is taken as the result.

The entire process adapted from Ref. 6 is show in Figure 2.5.3 below.

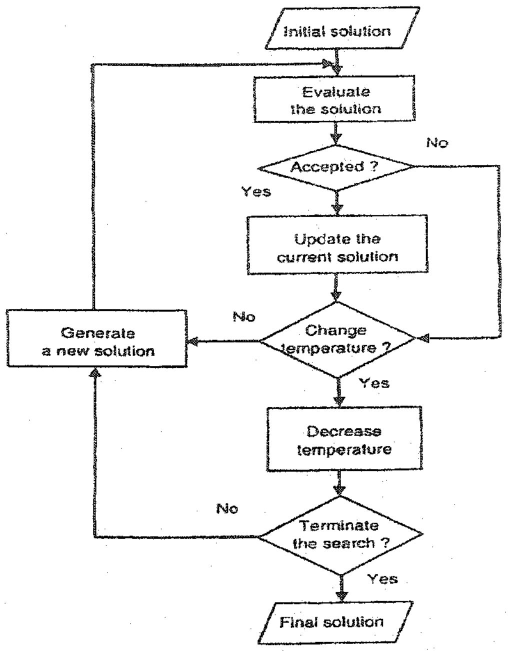


Figure 2.5.3: Flowchart of a standard Simulated Annealing algorithm

The critical issue in any simulated annealing algorithm is the annealing schedule which consists of (1) the initial temperature (2) a cooling function (3) the number of iteration to be performed at each temperature and (4) a stopping criterion to terminate the algorithm. A system that is cooled too fast may freeze at an undesirable high energy level. The freezing of a system at an undesirable energy state corresponds to the problem of an undesirable local optimization.

2.5.4 Genetic Algorithms

Genetic algorithms (GAs) are search and optimization tools that have the capabilities of handling a population of designs using many mixed design variable types under numerous design constraints. A simple GA paradigm - adapted from Ref. 64 - is shown in Figure 2.5.4. Moreover, the GA is able to explore a complex search space and find combinations of design variables that by far exceed that analyzed by the traditional design methods. The GA is often selected as an optimization tool since it differs and possesses the following advantages – as cited from Goldberg [27]- over the traditional optimization techniques:

(1) The search strategy is blind
(2) The search parameter is stochastic (meaning it is defined by a random distribution of probabilities.
(3) The search is conducted from a population of solution as opposed to one point.
(4) Manipulation of the coding instead of the parameters.

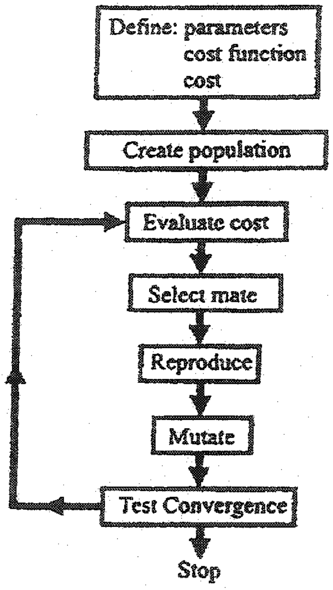


Figure 2.5.4: A simple genetic algorithm

It can be argued that the genetic algorithm is one of the most robust evolutionary algorithms in use today. Furthermore, GAs have found extensive application in the field of research and development, especially in areas where optimization is the key objective. For small spaces, classical exhaustive methods usually suffice; however, for larger spaces special artificial intelligence techniques must be employed.

Genetic algorithms are among such techniques; they are stochastic algorithms whose search methods model some natural phenomena: genetic inheritance and Darwinian strife for survival [65]. The range of application and robustness of the GA compared to other methods is presented in Figure 2.5.5 and was adapted from Ref. 66. As Figure 2.5.5 indicates, even though highly problem specific questions can outperform the GA, their range of application is very small. Although highly problem specific methods can outperform a genetic algorithm, their domain of applicability is small [6]. GAs are quite robust, and through random operators can be made efficient.

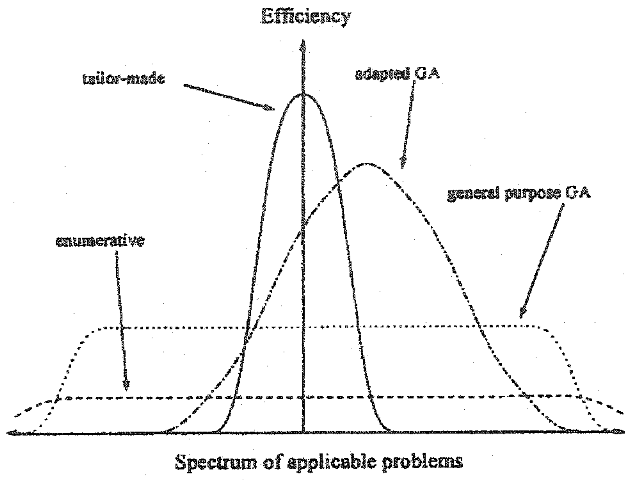


Figure 2.5.5: Comparison of the robustness of GA with more traditional methods

Despite its efficiency some have argued that there are some pitfalls and challenges in assuming that the GA is a true optimization tool. Moreover, since no gradient information is used in the GA search approach there is no proof of convergence. Hence the GA lacks a reliable means of asserting optimality. Genetic algorithms usually get close to an optimum point but can occupy long computing time to locate it exactly. Consequently, researchers realizing this problem have created hybrid GAs that employs the traditional optimization approach (mostly OR methods) at the end of the GA search to quickly find the optimum. Nonetheless, often it is more effective to run the GA for several cases to near optimality rather than run a single case to exact optimality [64]. Even though the GA may be less effective as a true optimization method it can be quite efficient as a technique to narrow the vast design space to the most interesting areas, which is a major goal at the start of most research projects [32].

Genetic algorithms are a class of general-purpose (domain independent) search methods, which strike a remarkable balance between exploration and exploitation of the search space [65]. The GA has proven to outperform the more traditional gradient methods of optimization. These methods are limited in application since they require finding the derivative, they are constrained to small domains whose first derivative must be continuous and they can handle only a small number of variables. As Figure 2.5.5 suggest, given enough information about the search space it will always be possible to construct a search method that will outperform the GA. However, obtaining such information is for many problems almost as difficult as solving the problem itself [66]. Thus, the robustness of GAs makes them an ideal candidate for an optimization tool.

Since GAs are the main focus of this thesis more detailed information about the technique, its creation and usefulness in solving a wide range of problems is provided in later chapters.

2.5.5 Artificial Neural Network

Neural Networks are modeled on the mechanism and structure of the brain [67-68]. Theoretically artificial neural networks (ANN) have a parallel-distributed information processing structure. Two of the major features of ANN are their ability to learn from examples and their tolerance to noise and damage to their components.

Thus the motivation of ANN research is to solve a variety of problems that are proving very difficult to solve even with conventional computers. ANN learns how to process information usually by being given either supervised training or graded training. Both of these methods run through a series of trial.

2.5.5.1 Supervised Training

For supervised training, the neural network is supervised with both input data and desired output data (correct answers as examples). After each trial, the network compares its own output with the right answers (derived output), corrects any differences, and tries again, iterating until the output error reaches an acceptable level.

2.5.5.2 Graded Training

Graded training the network is given input data but no desired output data; instead, after each trial or series of trials it is given a grade or performance score that tells it how well it is doing.
In both cases, after training, the network is ready to process genuine inputs.

2.5.5.3 Physics of the ANN

Neurons are analogy devices. This means they take their synaptic inputs, perform a computation on these inputs, and generate an output, which is usually a continuous valued firing frequency [6]. Neurons talk to one another. The connection between neurons is called synapses. A simple model of a neuron is shown in Figure 2.5.6.

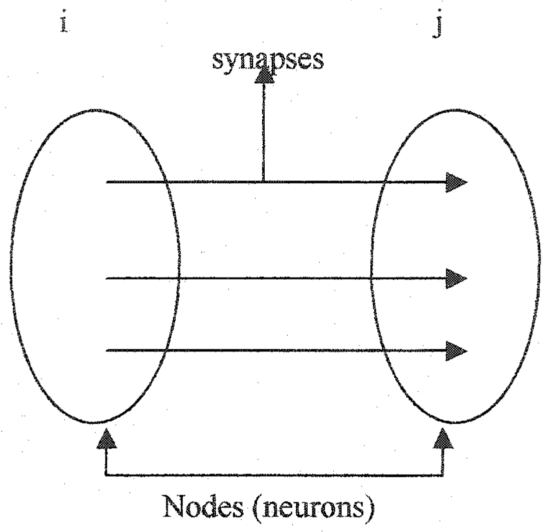


Figure 2.5.6: Simple model of a neuron

Neural networks models are algorithms for cognitive tasks such as learning and optimization, which are in a loose sense based on concepts, derived from research into the nature of the brain [68].

The transfer function usually is taken to have the form:

$$f(\sum_{ik} w_{ik} n_k - v_k) \qquad (2)$$

where w_{ik} is the real value weight

n_k is the number of iterations

v_k is the real value bias

Neural networks differ from a traditional computer program in that the steps in the program are not executed sequentially, but in parallel within each elementary unit.
Neural Networks replaces the deterministic evolution law by a stochastic law.

2.5.5.3.1 Why neural Networks?

(1) The desire to understand the principles in which the human brain works.

(2) The wish to build machines that are capable of performing complex tasks for which the sequentially operating, programmable computers are not well suited.

2.5.5.3.2 Advantage and Disadvantage of ANN

(1) An important advantage of the neural network is their high degree of error resistivity.
(2) A major disadvantage of neural networks is the broad lack of understanding of how they actually solve a given cognitive task.

2.6 Reason for the Necessity of Heuristic/Soft Computing Techniques

Many real-world problems today entail the fusion of many disciplines. It is often very difficult, time consuming and expensive to undertake a project by simply looking at just one section of the project. Hence multidisciplinary optimization (MDO) that employs soft computing or heuristic techniques has become very popular, particular in engineering.

2.6.1 Definition of Soft Computing Techniques

Soft computing represents one of the newest fields in the area of computational intelligence. It employs state of the art computational tools to deal effectively with systems characterized by complex structuring and ill-defined dynamics. These tools involve recently developed computational techniques based on fuzzy set theory, connectionist modeling and evolutionary computation [69]. Depending on the type of application, these tools could be used in either integrated or stand alone schemes. Chapter 3 represents a through overview of the fundamental components of designing one of these soft computing techniques.
According to Zadeh [70], the founder of Fuzzy Set and Fuzzy Logic: "Soft Computing is an emerging approach to computing which parallels the remarkable ability of the human mind to reason and learn in an environment of uncertainty and imprecision."

Soft computing techniques was also defined by Zadeh [70] as:

> Soft computing differs from conventional (hard) computing in that, unlike hard computing, it is tolerant of imprecision, uncertainty and partial truth. In effect, the role model for soft computing is the human mind. The guiding principle of soft computing is: Exploit the

tolerance of imprecision, uncertainty and partial truth to achieve tractability, robustness and low solution cost.

Hence, soft computing aims at mimicking human reasoning - particularly in a computer environment - and thus in practice, the soft computing systems are adaptive and intelligent in nature.

2.6.2 Multidisciplinary Design Optimization Defined

Multidisciplinary design optimization (MDO) has recently emerged as a field of research and practice that brings together many previously disjointed disciplines and tools of engineering. MDO can be described as a technology, environment, or methodology for the design of complex, coupled engineering systems, such as aircraft, automobiles, and other mechanisms, the behavior of which is determined by interacting subsystems [71]. The MDO methodology is applicable not only to design in general, but also to parameter identification. It attempts to make the design process less expensive and more reliable.

MDO eases the process of design and improves system performance by ensuring that the latest advances in each of the contributing disciplines are used to the fullest, taking advantage of the interactions between the subsystems [71]. Although the potential of MDO for improving the design process and reducing the manufacturing cost of complex systems is widely recognized by the engineering community, the extent of practical application of the methodology is not as great as it should be, due to the shortage of easily applicable MDO tools. Thus the exchange of expertise is crucial for the implementation of this exiting and promising methodology.

Typically, MDO problems are characterized by discrete as well as continuous design variables. Some operations research methods (traditional methods) can only handle continuous design variables. In addition, MDO problems often entail a large number of design variables and constraints, a task that is fill with problems if traditional methods are used. Thus the use of gradient-based methods is limited by the presence of discrete variables, the propensity of these methods to converge to a relative local optimum, and the multi-modality in the search space. It is for these reasons that AI methods have become attractive and ever so popular for solving highly complex engineering problems. However, this does not negate the importance of traditional methods as would become clearer in later chapters where various applications are investigated.

2.6.3 Soft computing technique as Industries path to optimization in the near future

The field of optimization has gained a renewed interest and has thus witnessed a healthy growth in recent years. This is due to the recognition that the design and development of complex systems, which includes many disciplines, can no longer be done in subsections with each discipline isolated from each other. An increase level of complexity within sub-disciplines, and the need to extract the advantages of a synergistic design process, dictate the need for a more comprehensive strategy [58]. Generic solution strategies for multidisciplinary system have evolved more recently, and are applicable to the problem of multidisciplinary design. Typically MDO involves many design variables, many constraints and analysis from various contributing disciplines that are not completely independent, and thus coupling between disciplines may exist.

The difficulties incurred when MDO problems are treated, as a single large-scale optimization problem using traditional gradient methods cannot be removed by simply using non-gradient methods of optimization. In fact, higher computational cost is one major problem encountered in using soft computing techniques. However, these methods have received tremendous attention in research project because the do not require gradient information, they are more robust in handling both continuous and discrete design variables, and above all they share an enhanced ability to locate the globally optimal solution.

2.6.4 Current and ongoing research software developed in which artificial intelligent techniques are utilized

Many of the AI methods are not easy to program but are becoming readily available via optimization toolboxes in other programming software such as C [72] and MATLAB [73]. The difficulty encountered with using these tools is the lack of understanding of how the software works and how to implement a problem that need to be optimized into the software.

Apart from this route, other commercial software that are available are now including an optimization section. For example finite element software such as ANSYS [74], Nastran [75], PAM-OPT [76] has an optimization section that is available to users. The problem encountered with these software is that the optimization techniques used in these software are typically not AI based.

Commercial software base on genetic algorithms is available and some are listed below. A good survey of these software can be found in Ref. 77. **Omega** and **Xpert Rule Gen A Sys** are menu-based rule systems that have spreadsheet interfaces. **Evoler**, a product of Palisade Corporation, solves constrained optimization problems using a GA. Other general-purpose software targeted towards easy integration is **GAGA** (**G**enetic **A**lgorithms for **G**eneral **A**pplication), **Genesis**, **Genitor** (**Gen**etic **I**mplemen**tor**), and **OOGA** (object oriented genetic algorithm).

Software based on the other AI techniques is also available, and often most of these techniques are lumped into one package in order to provide great flexibility to user.

It is believe that tremendous power is rendered when one understands, creates, and then develop his/her own optimization codes for use. In this way the user is fully aware of the concepts that surrounds the algorithms and above all is very capable of implementing any problem in the optimization routine.

2.7 Why Genetic Algorithms

Apart from the major advantages of the GA over traditional methods of optimization already mentioned, GA was chosen as the AI method of study in this thesis because of the ease of understanding the concepts that underlies the algorithm. In addition, the software used to program the GA, namely MATLAB, offered capabilities that are well suited for programming a GA. In addition, assistance of peers and other researchers active in the field of optimization could not be ignored. Thus with these factors glued together a robust general purpose genetic algorithm was developed and applied to a series of engineering problems. The quality of the results obtained after implementation of the various engineering applications would support the claim of robustness and generality of the GA created for this study.

Limited availability of literature material fully describing the technique, complexity of the technique, and time constraints, were all deterrents in selecting the other AI techniques. In addition, the means of applying the method to solve problems was also a major difficulty faced in understanding the concepts of other AI techniques.

Chapter 3

Genetic algorithms the Soft Computing Technique of Choice

Ever since the work of Golderg [27], and his students, much work has been done on genetic algorithms (GAs), most readable but somewhat incapable of shedding light in the arena of application. This chapter serves as a guideline for understanding the components and backbone concepts of a genetic algorithm. The creation of a genetic algorithm can be a beneficial tool because of its applicability to a vast area of problem solving and optimization. The language that was employed to create the GA is MATLAB [73], however some pseudo-codes are also provided for those interested in exploring other programming languages.

3.1 What is a Genetic Algorithm (GA)?

Genetic algorithms have proven to be very powerful search and optimization tools especially when only little of the structure in the data is known [64]. The main power of the GA lies in their capability to create new combinations of variables that could lead to a better design. The idea is that, given a certain problem representation, the GA is able through repeated used of genetic operators such as selection, crossover and mutation to combine those parts of a solution that are necessary to form a globally optimal solution.

Thus, the underlying principle of a genetic algorithm is based on the mechanism of evolution [64]. A genetic algorithm searches or optimizes by means of the use of multiple searching points or solution candidates (population based search). A GA starts with a population of strings and thereafter generates a successive population of strings. In genetic algorithms each individual represents a certain solution to a given problem. The quality of this solution is expressed by a so-called fitness value. Offspring's with a higher fitness value have a better chance to survive and to reproduce (quite indicative of nature).

3.2 Difference and Advantages of GA from Conventional Optimization Techniques

(1) The search strategy is blind

(2) The search parameter is stochastic (meaning it is defined by a random distribution of probabilities.

(3) The search is conducted from a population of solution as opposed to one point.

(4) Manipulation of the coding instead of the parameters.

3.2.1 Disadvantages of GA

(1) Convergence speed near the global optimum is usually slow.

(2) No proof of optimality

(3) Reproducibility of results is sometimes difficult

3.3 Genetic Operators

A simple GA adapted from Ref. 64 - and repeated here for clarity - is comprised of three essential genetic operators (see Figure 3.3.1), all of which depend on random probabilities and are listed as:

(1) Selection/ Reproduction } mimics Darwinian survival of fittest

(2) Crossover } mimics the heredity
(3) Mutation } of genes

These operators are passed iteratively through a loop after the fitness/cost is calculated for each chromosome and is employed at each generation of the GA search procedure. The idea is that, given a certain problem representation, the GA is able through repeated used of genetic operators such as selection, crossover and mutation to combine those parts of a solution that are necessary to form a globally optimal solution [64] (see Figure 3.3.1).

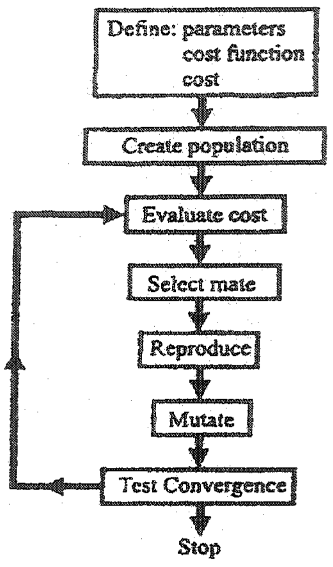
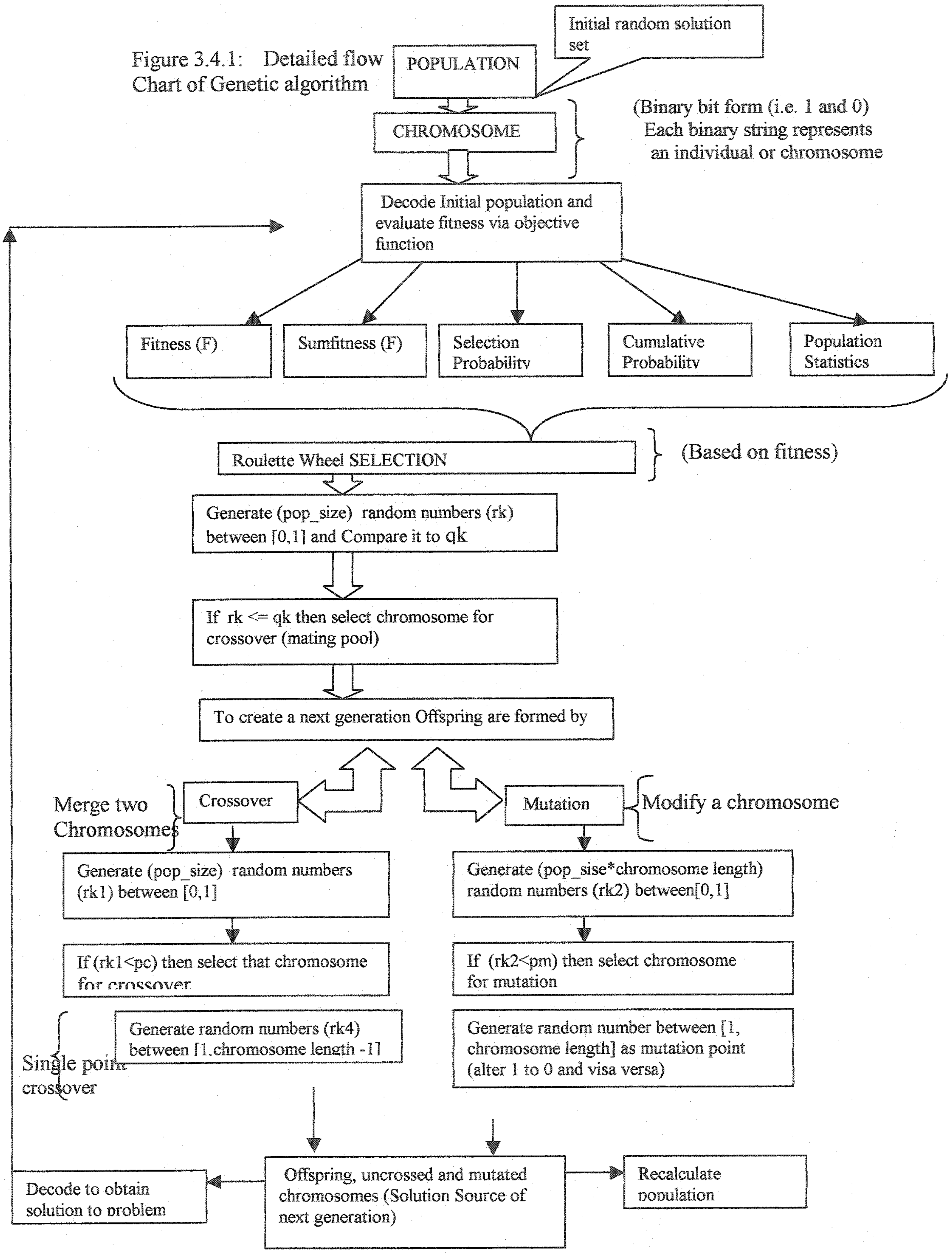


Figure 3.3.1: Simple flow chart of a Genetic Algorithm

Figure 3.4.1: Detailed flow Chart of Genetic algorithm



POPULATION
Initial random solution set
CHROMOSOME
(Binary bit form (i.e. 1 and 0) Each binary string represents an individual or chromosome
Decode Initial population and evaluate fitness via objective function
Fitness (F)
Sumfitness (F)
Selection Probability
Cumulative Probability
Population Statistics
Roulette Wheel SELECTION
(Based on fitness)
Generate (pop_size) random numbers (rk) between [0,1] and Compare it to qk
If rk <= qk then select chromosome for crossover (mating pool)
To create a next generation Offspring are formed by
Merge two Chromosomes
Crossover
Mutation
Modify a chromosome
Generate (pop_size) random numbers (rk1) between [0,1]
If (rk1<pc) then select that chromosome for crossover
Generate random numbers (rk4) between [1,chromosome length -1]
Single point crossover
Generate (pop_sise*chromosome length) random numbers (rk2) between[0,1]
If (rk2<pm) then select chromosome for mutation
Generate random number between [1, chromosome length] as mutation point (alter 1 to 0 and visa versa)
Offspring, uncrossed and mutated chromosomes (Solution Source of next generation)
Decode to obtain solution to problem
Recalculate population

3.4 Basic Elements of a Genetic Algorithm

A detailed flow chart that highlights all the operations involved, as well as the steps that should be taken to create a successful genetic algorithm is shown above in Figure 3.4.1. Every aspect of the flow chart is elucidated in detail below. However, it is important that one reviews and understands the flow of events in the above flow chart before reading the material below. This task can significantly aid is grasping the ideas and concepts of a genetic algorithm. Thus without a proper working decoding scheme one cannot begin working on selection; without a proper working selection scheme one cannot start working on crossover and so on. Hence there exist tremendous interdependencies in creating a GA. As such, care and caution must be taken when deciding if the particular technique works and when to move to the next stage of GA development. The objective of any GA is to coordinate selection, crossover and mutation in a single generation or iteration.

The evolutionary process of a genetic algorithm entails the fundamental process of selection or reproduction, crossover and mutation. These genetic operators serve a vital role in guiding the GA search process.

3.4.1 Selection

Reproduction provides a way of allowing fitter chromosomes to progress into the next generation. A binary string represents each chromosome. This string is made up of genes, which carries the parameters of the design or variables need to optimize. So for example, the function $f(x) = x^2$, x is the first and only gene in chromosome, which is encoded into a binary string. A chromosome represents a certain fitness value. The chance of being selected is dependent on the string fitness value. Therefore only the selected chromosomes
the next generation. Once this string has been selected, the process of crossover creates offspring. Further indebt details on this operator are provided later in this chapter.

3.4.2 Crossover

Without crossover, the average fitness of the population will climb until it equals the fitness of the fittest member [65]. After this point the GA can only improve via mutation. Crossover provided a method whereby information from differing solutions can be melded to allow the exploration of new parts of the search space [66]. In crossover we merge two chromosomes. This operator produces two new strings, that is, offspring from two selected parents. Further details on the mechanism of crossover are presented later in this chapter.

3.4.3 Mutation

Mutation is sort of an assurance against premature loss of important information. It also serves the crucial role of forcing the population out of a static condition by adding new genetic information to it [78]. This implies that the GA is now able to search in regions of the search space never explored and as a result, induce that information into the population. This is often referred to as diversifying the population. This is achieved by the occasional random alteration of a bit value.

Hence a good genetic algorithms performance requires the choice of high crossover probability, low mutation probability and moderate population size [27].

3.5 The Coding of the algorithm

In order to elucidate the coding of the flow chart given in Figure 3.4.1, an example is used. The example highlights the steps and warns of possible pitfalls in the development of a genetic algorithm. The following basic mathematical objective function, which is to be maximized, is used:

$$\texttt{maximize}(\ f(x,y)\ ) = x^2 + y^2$$
$$0 \le x \le 10$$
$$0 \le y \le 10$$

3.5.1 Inputs required by program

Initially a few constants and decision variables must be specified before running any GA program. The variables are as follows:

(a) Population size

(b) Number of parameter

(c) Probability of crossover and mutation

(d) Constraints (problem dependent and often known)

(e) Degree of accuracy required

(f) The length of a chromosome

(g) The number of generations

Deciding upon a value to use for the variables above is given below. Note that in some areas such as population size, research is still active in determining the effects of this value on the convergence quality of the GA.

3.5.1.1 Population size

There is no hard and fast rule in deciding the size of the population. However, if the population is too small then the loss of genetic diversity may compromise the search and may lead to sub-optimal rather than optimal solution. In addition, too large a population can really hinder the speed in finding a solution since there is a much larger area to search. A population size of 10 was used for the problem defined above. It can be argued that a relatively small population with a large generation number is best for speedy results. Note however that this is not always the case especially when dealing with complex objective functions.

Population size and the number of generations are inversely proportional to each other i.e. we can increase the number of generations and at the same time decrease the population size needed to converge, and visa versa.

3.5.1.2 Number of Parameter

The number of parameters depends on the number of dependent parameters in the objective function or the number of design variables directly linked to the objective function. For the problem defined above the number of parameters is two, i.e. x and y. Both x and y need to be encoded into one individual (i.e. a chromosome) as our initial inputs. Therefore, a chromosome is divided into genes (x and y). A gene is the GA's representation of a single factor value for control.

Chromosome = $[x \ , \ y]$

Objective function = f (Decoded Chromosomes)

For this problem we have 10 chromosomes, since the population size was set as 10.

3.5.1.3 Probability of crossover and mutation

For this problem a probability of crossover of 50% (pc=0.5) and a probability of mutation of 1% (pm=0.01) were used. pc=0.5 means that we expect 50 % of the chromosomes to undergo crossover i.e. 0.5*10=5 chromosomes selected for crossover. In addition having pm=0.01 implies that we expect 1% of the chromosome length to undergo mutation. In most cases pc and pm can assume constant values that do not change over the generations. Typically values for the probability of crossover vary between 50% and 90%, while values for the probability of mutation values vary between 1% and 5%.

3.5.1.4 Constraints

Constraints are usually given in the problem statement or have to be decided upon in the problem definition in order to add some practicality to the problem. For the problem above, the range of values for x and y can be seen as soft constraints since they restrain the problem to those regions of the search space.

3.5.1.5 The degree of accuracy required

The use of common sense is required when determining the degree of accuracy needed as output for the objective function values since this is one determining factor for the chromosome length. Every degree of accuracy requires an additional 3.3 bits needed to be encoded in each chromosome. This can mean a longer computational time. So eight decimal point degree of accuracy requires 8*3.3=26.4 or 27 bits in the gene, which for the above problem posed, is too high. Perhaps four decimals place is sufficient.

3.5.1.6 Length of chromosome

Having decided on a reasonable degree of accuracy one can then determine the length of each gene (i.e. x and y for the problem above) that constitutes the individual; these individuals are represented by a chromosome. The following formula adapted from ref. 79 is employed:

$$2^{mj-1} < (b_j - a_j)10^n \leq 2^{mj} - 1 \qquad (1)$$

where b_j and a_j are the range and domain respectively for the design variables. n is number of degree of accuracy required, and mj is a value found trough trial and error to satisfy the inequality. The value of mj is the length of the gene in the chromosome. In our case we have two inputs and since the domain range is the same for these inputs then m1=m2=17. As a result our total chromosome length =m1+m2=34.

3.5.1.7 The number of generations

Each run of a GA usually consists of either satisfying a problem specific success predicate or completing a specific number of generations. The later is one easy way of terminating the program and is used for all applications in this thesis.

3.5.2 Generating the population and decoding

Since the genetic algorithm studied is binary, a matrix of binary numbers must be generated to create the population. The initial population is totally random and generated by the computer using the random command found in many computer languages. This population of random binary digits must then be decoded. Decoding is important because the objective function cannot be evaluated using a binary string. The objective function value plus a constraint violation value (penalty value) is called the fitness value. As a result, the binary digits (encoded value(s)) must be converted to real number(s) so that the fitness function value can be calculated. Being able to decode a chromosome into real number(s) would then allow for decoding of the entire population into real value(s). After decoding and obtaining the real number values in the entire population, these values are then simply substituted into the fitness function for evaluation. This would result in a set of fitness values equal to the population size.

For the problem defined above: Fitness (real value x, real value y) = real value x^2 + real value y^2. This would result in 10 fitness values, one for each individual in the population.

Knowing these fitness values the maximum fitness, total fitness, selection probability, and cumulative probability can be evaluated as follows:

Maximum fitness = **max** (fitness); total fitness= **sum** (fitness); selection probability (i) = fitness (i) / total fitness; and finally cumulative probability=**sum**(selection probability (i) + selection probability(i +1)). Here i represent an individual in the population and the words in bold represent a simple mathematical function command present in most programming languages.

3.5.3 Genetic Operators

There are three basic genetic operators that constitute most genetic algorithms; they are, selection, crossover and mutation. These operators are the main components of the GA that work together to explore and exploit various areas of the search space in order to locate a global optimum.

3.5.3.1 Selection Operators

Apart from the population size and control parameters (i.e. probability of crossover and probability of mutation), the selection type – as will be evident – strongly influences the quality of convergence. There are many selection schemes that have been created. Many of them attempt to make the selection process more random and, to a larger extent, replicate a survival phenomenon in nature. Currently there are three selection operators available for use in the genetic algorithm, they are, roulette wheel selection, tournament selection, and normal geometric selection.

3.5.3.1.1 Roulette Wheel Selection

Roulette wheel selection or fitness proportional selection assigns reproduction opportunities to an individual based on their relative fitness [27]. Thus it is a stochastic process of selection, which will choose individuals on the basis of performance with respect to other individuals in the population. Individual strings are copied according to their fitness values [64]. The fitness value is a measure of the goodness or profit. Strings with a higher fitness have a better chance of contributing one or more offspring's in the next generation.

Creating a biased roulette wheel is used to perform selection. Each time a chromosome is required a spin of the roulette wheel yields a reproduction candidate. Once a string has been selected for reproduction an exact replica of the string is made. This string is then place into a mating pool as part of a new population. This new population would then be allowed to mate and reproduce using crossover and mutation.

3.5.3.1.2 Tournament Selection

Tournament selection is a random process that closely mimics mating competition. From the population, a fixed number of competitors (tournament size) are randomly selected. The individual with the highest fitness wins the tournament and is then placed in a mating pool. Note that as the tournament size gets larger the selection intensity increases [80]. This implies that we can adjust the probability of selection by simply changing the number of competitors.

Tournament selection mimics mating competition in nature more closely than roulette selection. Hence in some applications it was used on a comparison bases with roulette wheel selection. This scheme is implemented by three simple steps outline below:

(a) A tournament size of N chromosomes is randomly selected from the population.

(b) The chromosome from (a) that has the highest fitness is the winner of the tournament and so is selected as a parent.

(c) This process is repeated for a number of tournaments equivalent to the population size.

3.5.3.1.3 Normal Geometric Selection

Normal geometric selection scheme was added as part of the selection techniques available within the GA. Normal geometric selection is a ranking geometric operator based on the normalized geometric distribution. The input to the program is the old population and the selection probabilities. The output is the new population selected from the old population. This method behaves better than roulette selection scheme but is still not as powerful as tournament selection. It was also observed that this method does not allow a speedy convergence due to domination by a few super chromosomes.

3.5.3.2 Crossover Operators

Without crossover, the average fitness of the population will climb until it equals the fitness of the fittest member [65]. After this point it can only improve via mutation. Crossover provides a method whereby information for differing solutions can be melded to allow the exploration of new parts of the search space [66].

3.5.3.2.1 One point Crossover

Using the chromosomes selected via a selection scheme crossover can then be performed. In crossover we merge two chromosomes. This operator produces two new strings (offspring) from two selected parents combining their information. For example consider the two parents:

101011001 $\longleftarrow$ parent 1
001011100 $\longleftarrow$ parent 2

A crossover point or cut-point, which is randomly selected between the first and the second to last bit, is first generated. Assume for illustrative purposes that the cut-point is 3; this implies every bit to the right of 3 is swapped. Figure 3.5.1 demonstrates this procedure:

101011001 $\longleftarrow$ parent 1
001011100 $\longleftarrow$ parent 2

101011100 $\longleftarrow$ Offspring 1
001011001 $\longleftarrow$ Offspring 2

Figure 3.5.1: One point crossover applied to binary string at random point

Generally the process of crossover is performed as follows:

(a) First we generate a population size of random numbers.

(b) If the random numbers are less than the probability of crossover, then select the chromosome as a parent for crossover.

(c) We must then generate a random number between 1 and the second to last bit (i.e. 33 for the problem defined above)

3.5.3.2.2 Two point Crossover

Two-point crossover is performed in the same way as one point crossover but instead the process is repeated twice. The idea behind two point and multi-point- and indeed many of the variations on the crossover operator - is that parts of the chromosome representation that contribute most to the performance of a particular individual may not necessarily be contained in adjacent sub-strings. Two-point crossover appears to encourage a greater exploration of the search space when compared to one-point crossover. Thus rather than favoring the convergence to highly fit individuals early in the search, two-point crossover spends more time exploring the search space, thus making the search more robust [65].

3.5.3.2.3 Multi-point Crossover

The idea behind multi-point crossover is to meld parts of the chromosome representation that contribute most to the fitness of a particular individual. This information may not necessarily be contained in adjacent substrings. Furthermore, the disruptive nature of multi-point crossover appears to encourage a greater exploration of the search space, when compared to one point and two point crossover.

3.5.3.2.4 Uniform Crossover

Uniform crossover generalizes single and multi-point crossover schemes to make every locus a potential cross point. A crossover mask, the same length as the individual structure, is created at random. The parity of the bits in the mask indicates which parent will supply the offspring with which bits [32]. Uniform crossover, like multi-point crossover, has been claimed to reduce the bias associated with the length of the binary representation used and the particular coding for a given parameter set [27,78].

3.5.3.2 Mutation Operators

Mutation helps to diversify the population, and is achieved by the occasional random alteration of the value of a string. The role this serves is that it induces new information into the population and consequently allows the GA to search in new regions of the search space.

3.5.3.2.1 One-bit mutation

One bit mutation is achieved by the occasional random alteration of the value of a string bit value as shown in Figure 3.5.2. This simply means changing a 1 to a 0, and visa versa, at the bit position that the random number is less than the probability of mutation. To elaborate mutation is achieved as follows:

(a) Generate population size * chromosome length random numbers between 0 and 1. For the above problem this implies, 10*34=340 random numbers.

(a) If any of these random numbers (termed rk2) is less than pm (probability of mutation) then we alter the bit from one to zero or visa versa. For example consider previous example:

101011001 $\longleftarrow$ sample 1

If the rk2<pm at bit number 5 then binary string becomes:

101001001 $\longleftarrow$ mutated sample 1

Figure 3.5.2: One bit mutation of a binary bit string

Figure 3.5.2 illustrates the fundamental principal behind one bit mutation and the many other forms of mutation.

3.5.3.2.2 Two-bit mutation

Two bit swapping mutation was employed to introduce diversity and encourage the GA to explore the search space. Two-bit mutation is very similar to single bit mutation; but instead, a two-bit swapping mutation operator exchange the values of randomly selected bits in a specific population member based on the probability of mutation.

3.5.4 Stopping Criterion

The stopping criterion is simply a way to terminate the program. Throughout this thesis the number of generation specified is used as the stopping criteria.

Tabulating the results and if possible plotting maximum fitness and average fitness versus generation number can help visualize the GA search history, and more importantly, help determine whether the GA is functioning. For example, it is expected that both maximum fitness and average fitness should increase as the run progresses when maximizing.

3.6 Handling Constraints

Constraint optimization is often referred to as non-linear programming. This sort of programming involving constraints is where 'real world' problems take over (since many practical problems are nonlinear). For example we want to design a structure that can withstand a certain prescribed load but will not exceed a certain deflection or stress. The central problem in applying a GA is how to handle the deflection or stress constraint.

One common method that is discussed here is the penalizing strategy, though they are other strategies presented in the literature such as the rejecting method and the repairing method. The main role that the penalty strategy plays in the GA is that it unconstraint our problem by adding or multiply a certain penalty to the objective function for any violation of the constraint. This addition or multiplication will have the effect of adding 'weight' to our fitness value thereby decreasing its goodness if we are minimizing. Hence, weight plays the role of penalties if a potential solution does not satisfy them [81]. One major advantage of the penalty strategy compared to other methods is that it does not disregard infeasible solutions; instead, it uses these solutions in such a way to aid the search to a better solution [79]. Here the problem of striking a balance between exploitation and exploration arise."… the penalty technique is used to keep a certain amount of infeasible solutions in each generation so as to enforce genetic search towards an optimal solution from both sides of feasible and infeasible regions"[79]. The Figure 3.6.1 illustrates this point clearly.

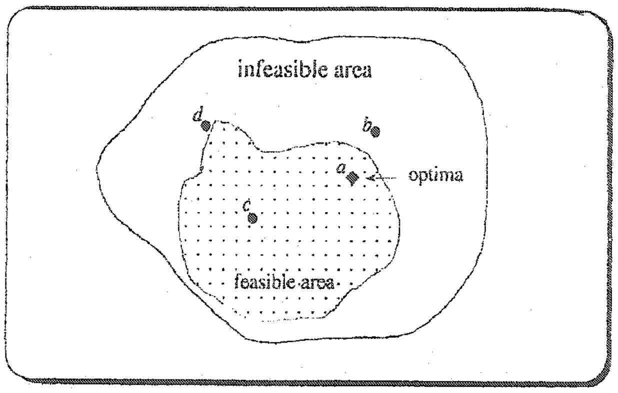


Figure 3.6.1: Solution space: feasible area and infeasible area

The solution space contains two regions i.e. feasible and infeasible.

In Figure 3.6.1- adapted from Ref. 79- point "b", which is infeasible, is closer to the optimum than feasible "c" and infeasible "d". We can also believe that "b" contains more information about optima "a" even though it is infeasible. The main issue of the penalty strategy is how to design the penalty function p(x) which can effectively guide genetic search toward the promising area of solution space" [79]. Thus, we wish not to reject our infeasible solutions because some of them may provide much more useful information about the optimum than the feasible solutions.

The outline below should be followed as an initial encounter with handling constraints using the GA. The traditional method outline below should be followed.

3.6.1 Traditional Approach

Fitness(x) = f(x) + p(x)
Where f(x) is our objective function that we want to optimize
p(x) is our penalty function

For maximization problems:	p(x) = 0 if x feasible
	p(x) < 0 otherwise
For minimization problems:	p(x) = 0 if feasible
	p(x) > 0 otherwise

f(x) is problem dependent and so is p(x), but here are some guidelines that can prove useful:

Let p(x)=0 if fitness is feasible

otherwise let $p(x)=\sum_{i=1}^{i=m} r_i g_i^2$ (2)

where m is the number of constraint
where r_i is a variable penalty coefficient for the i^{th} constraint;

g_i is the amount of violation of the i^{th} constraint.
g_i = max{ 0, constraint(i) – calculated constraint value}

3.6.2 Adaptive Penalty

The advantages of this method over the traditional approach is that it is non-parameteric and, most importantly, it is problem independent [79, 81].

For minimization:
Fitness = objective value * penalty value
Fitness = f(x)*p(x)
Where f(x) is the objective value and p(x) is the penalty value.
p(x)=1 if feasible
p(x)>1 otherwise
The penalty value is computed using the method outlined below:

$$p(x) = 1 + \frac{1}{m}\sum_{i=1}^{m}\left(\frac{\Delta b_i(x)}{(\Delta b_i)_{\max}}\right)^{\alpha} \qquad (3)$$

where: $\Delta b_i(x) = \max(0, g_i(x))$

$(\Delta b_i)_{\max} = \max(\varepsilon, \Delta b_i(x) | x \in p(x))$

where:
g_i= the degree of violation of the constraints, dynamically scaled according to the best solution found in the population set; ε= small positive value that is used to avoid division by zero when calculating p(x); m= number of constraints. $\Delta b_i(x)$= degree of violation of the constraint i for the chromosome x; $(\Delta b_i)_{\max}$= maximum violation in constraint i among the current population; α= value used to adjust the severity of the penalty function. Traditionally, α varies between 1 and 3. The α used throughout this thesis is 2.

3.7 Techniques to Improve the Basic Genetic Algorithm Performance

The field of optimization is constantly under research in order to find quicker and more efficient ways of finding solutions. One phase you may stumble upon when involved with GA is 'finding a balance between exploration and exploitation' [79]. How to achieve this balance is the question that have driven researcher to look for alternatives to the traditional methods used in order to advance the capabilities of the GA. A few techniques employed to improve the simple GA used in this thesis are as follows:

(1) Elitism

(2) Fitness scaling or ranking

(3) Inverse scheme

(4) Dynamically changing probability of crossover and mutation

(5) Dynamically changing population size

(6) Better visualization of results

(7) Optimizing computing time by:

  (a) Calculating fitness only once

  (b) Writing a separate subroutine for the objective function

3.7.1 Elitism

In view of the probabilistic random nature of selection, the best individual in the population is not guaranteed to be selected and the selection process does not necessarily overlook the worst individual in the population [79]. Elitism is a way of preventing the loss of a potential good chromosome in the population by ensuring that its presence is maintained in the population at every generation until a better solution is found. For many large-scale applications, the search speed can be greatly improved by not losing the best, or elite, member between generations [73].

3.7.2 Fitness scaling or ranking

In our proportional selection procedure (i.e. roulette wheel selection) the probability of selection is dependent on the fitness. This scheme has some undesirable features in that there is a tendency for a few super chromosomes to dominate the selection process; in later generations, when the population is largely converged competition between chromosomes is less strong and a random search behavior will emerge [65].

Scaling and ranking is proposed to prevent this problem. Domination by a few super chromosomes usually causes the GA to reach sub-optimal rather than optimal value(s). Thus scaling and ranking is used to prevent premature convergence by maintaining a reasonable differential between relative fitness rating of the chromosomes.

Scaling usually requires finding constants such as “a” and “b” which has to be determined by the user. Because the user usually must have some knowledge about GA or must ‘play’ around with “a” and “b” this method is usually not preferred. Note also that these constants are problem dependent and so they lack generality.

Consequently, fitness ranking which has similar effects as scaling but avoids the need for extra scaling parameters (such as “a” and “b”) is chosen as the preferred method [79]. The ranking method ignores the actual objective function values; instead, it uses a ranking of chromosomes to determine survival probability [65].

The idea is simple:

(1) Sort the population from the best fitness to worst fitness

(2) Assign selection probability to each chromosome according to the ranking and not its fitness. (rank=1 means best individual, rank= population size means the worst one).

(3) Choose a ranking method:

  (a) Linear ranking

  (b) Exponential ranking

For linear ranking prob(rank)=q-(k-1)*r

where q is probability of best chromosome

k is the chromosome number

r=(q-qo)/(pop_size-1)

where qo is the probability of worst chromosome

For Exponential ranking we have:

prob(rank)=q(1-q)^(rank-1)

where q is the probability of best chromosome.

3.7.3 Inverse scheme

Since genetic algorithms optimize via maximization, the problem of changing the GA process to minimization arises. For minimization the fitter chromosome has a lower fitness or cost. Thus in the selection process we must be able to transform the large number into a small number. To achieve this the inverse scheme was employed.

The basic concept of the inverse scheme comes form a simple logical deduction of – the result of inverting a large number is a small number [32]. The fitness values are scaled as follows:

$$F_i' = \frac{F_{\max} - (F_{\min} / a)}{F_i - (F_{\min} / a)} \tag{4}$$

where F_i' is the scaled fitness values; $F_{\max}$ and $F_{\min}$ are the maximum and minimum fitness values respectively; a is a value slightly greater than one and F_i is the fitness values.

3.7.4 Dynamically changing probability of crossover and mutation

One important feature that can be added to the GA is its dynamically changing crossover and mutation rate. The process of implementing the above features is quite simple and is highly recommended for larger problems. It is known that the probability of crossover and the probability of mutation have significant effects on the convergence quality of GA. Typically these two probabilities are assigned constant values within the GA. In addition, it is generally accepted that the crossover probability should be much higher than the probability of mutation. Moreover, it is also known that initially the crossover probability (pc) should be lower in order to facilitate a wider exploration of the search space, and higher later on, to focus the search on an area in which the optimum lies. With respect to mutation we would also like a low probability of mutation initially and then to very slowly increase its probability, since a high mutation rate can lead to a random walk process. In effect what we desire is a fine balance between exploration early on and exploitation later on in the search process. The pc adaptation approach used is adapted from Ref. 82 and is as follows:

$$Pco = 0.1$$
$$Rc[i] = {gen\_no.}/{\max\_gen} \tag{5}$$

$$Pc(i) = Pco + (1 - Pco) * Rc[i]$$

where
Pco is the initial probability of crossover specified within the GA program; $Rc[i]$ is a scaling factor at generation number i; and $Pc(i)$ is the probability of crossover at generation i. gen_no is the current generation number and $\max\_gen$ is the maximum generation for the GA search and optimization process.

The probability of mutation is also dynamically altered during the search as follow:

$$Pmo = 0.01$$

$$Rm = (Pmo/0.05)^{1/\max\_gen} \tag{6}$$

Within the GA main loop:

$$Pm(i+1) = Pmo/Rm$$

where Pmo is the initial probability of mutation; Rm is a scaling factor; and $Pm(i+1)$ is the probability of mutation at generation i.

3.7.5 Dynamically Changing Population Size

The population size can dynamically change after some amount of generation runs. The aim here is to reduce the computational effort entailed in processing a population of designs and restrict the search to regions in which an optimum is located. It is believed that if the population can be accordingly reduced with the number of generations, the computation effort can be significantly reduced. The logic behind this idea comes from the concept of exploration and exploitation of the search space. Exploration is highly demanded initially in the GA search; however, later on when the search have converged to a near optimum there is less of a need for a large population. The central disadvantage of the GA i.e. slow convergence near global optimum stems from this very point. In order to eliminate, or at best curb this problem, a dynamically changing population size is recommended. Dynamic changing the population size encourages the optimization process to investigate better solutions in more restricted favorable regions of the search space [83]. Therefore, each reduction on the population size can be viewed as a step closer to the global optimum. The process of achieving this is quite simple and can be implemented in the following 6 steps:

(1) Start GA run with a relatively large population size

(2) After level "n" convergence quality is achieved randomly generate a number to determine the factor (r) by which to reduce population size

where

$$1 < r < b * pop\_size$$

"b" is a reduction factor set =1/4

Note that 'b' varies between 0 and 1.

(3) Generate r random numbers between 1 and population size, ensuring that no random number is repeated more than once.

(4) Extract the chromosomes from (3) and at same time ensure that the best design found thus far is preserved in the reduced population. This can be achieved by simply copying the new chromosome to the reduced population.

(5) Recall GA to process the new reduced population.

(6) Repeat process until a quarter or "b" of the initial population size remains.

Note that b=1/4 is an arbitrary number and can be increased or decreased accordingly, depending on the initial size of the population. In addition, level "n" convergence means the stage "n" of the population reduction level presently at. Hence initially "n" is unknown since we do not know at which generation we would obtain 25% improvement in fitness. Moreover, level "n" convergence means that we have achieved a b% (i.e. 25%) improvement in fitness values from generation 1 to generation n. At generation 'n' we would reduce the initial population size. Since GA is a stochastic process n varies from one test to the next.

3.7.6 Better Visualization of Results

Plotting the maximum fitness, minimum fitness, average fitness and total fitness versus the generation number is one way of visualizing the output of the GA. However other schemes that show how the search is taking place can also be developed to give some idea of how the GA is exploring the search space. This is illustrated in a later chapter on applications where benchmark mathematical functions are used and an animated search process is use to visualize how the GA search progresses.

3.8 Optimizing Computing Time

For any application of the GA, saving on computing time required to converge to an optimal solution is always demanded. A few simple techniques that can be employ to facilitate the creation of a more robust GA are given below.

3.8.1 Calculating fitness values only once

By manipulating function calls- apart from vectorising in a language like MATLAB - mainly reduces the number of times the fitness is calculated. This is achieved by first writing a separate function in which the fitness values are calculated. Then this function can be called only once. For example, calculating fitness after mutation and then again after elitism is not necessary. The fitness need not be computed again in elitism. Recalled in elitism we compare the fitness before selection and after mutation and randomly replace the best chromosome in the new population if necessary.
Apart from reducing time in elitism, time can also be reduced by sending the new fitness values, penalties and objective values after mutation or elitism up to selection. Recall that the new population after mutation or elitism -if there is the need- is sent up to the beginning of the program so that selection can now use this population to select from.

Note that after these changes have been made and the program is executed, there should be a noticeable reduction in computing time. Taking the time to optimize the code itself can be beneficial especially for large complex problems.

3.8.2 Stand-alone Genetic Algorithm

One addition to the program that can avoid difficulties, but does not save so much on time, is to have a separate function to calculate the fitness. In this way the GA can stand-alone. There is no need to ever make any changes to the main GA program. Consequently, all that is need is to create a fitness function and a penalty function. Note most penalty functions are problem dependent and this is really a bonus because it provides control over the way the constraints are coded. Approaches to handling constraints have been presented.

3.9 GUI for the GA

A graphical user interface (GUI) was also created in order to make the GA more user friendly. However, currently this interface is linked only to application in which mathematical functions are optimized. Due to the priority given to the projects this GUI was not linked to all applications, since it was not necessary for the running of the program.

3.10 Functionality of GA as an Optimization Tool

Despite its efficiency some have argued that there are some pitfalls and challenges in assuming that the GA is a true optimization tool. Moreover, since no gradient information is used in the GA search approach there is no proof of convergence. Hence the GA lacks a reliable means of asserting optimality. Genetic algorithms usually get close to an optimum point but can occupy long computing time to locate it exactly. Consequently, researchers realizing this problem have created hybrid GAs that employs the traditional gradient optimization approach at the end of the GA search to quickly find the optimum. Nonetheless, often it is more effective to run the GA for several cases to near optimality rather than run a single case to exact optimality [84]. Even though the GA may be less effective as a true optimization method it can be quite efficient as a technique to narrow the vast design space to the most interesting areas, which is a major goal of conceptual design.

Chapter 4

Basic Applications to Validate GA Code

The applications covered in this chapter are comprised of simple objective functions. The intention of this chapter is to reinforce that the GA created in Chapter 3 is working and functioning. More importantly, this chapter also aims to foster confidence in the power of the GA to solve a wide range of problems and avoid entrapment in local optima. Thus in some respect this chapter is a stepping-stone to larger and more complex mechanical engineering applications to be addressed in later chapters. Henceforth, this thesis progresses with increased level of difficulty as the number of applications grows. Enough information is presented to define the problem and in most cases a formulation of the problem and its implementation into the GA is also presented. These problems are intended to illustrate the versatility of genetic algorithms and provide some insight into the robustness of this algorithm.

4.1 Complex Mathematical functions

Traditionally in order to test the robustness of any optimization tool, highly multimodal mathematical functions are attempted. The results of these functions usually provide some idea about the ability of the optimizer to avoid entrapment into local optimums. Consequently, some of these mathematical functions have become benchmark functions for testing the robustness of new optimization methods. Two of these mathematical functions are listed below. It is recommended that these functions first be attempted before proceeding to solve any practical problem. In order to find the exact global optimum of these two functions a 3D plot of the function was carried out and then the location of highest peak was "eye balled" in order to approximate the location of the maximum or minimum value depending on whether we are maximizing or minimizing. Note that these test functions do not require a penalty function. In addition, it is important to remember that the GA is inherently a maximization tool. Since a GA is inherently a maximization tool, to minimize a function, set the function f(x) to f(x) = -(function); so for example, maximizing f(x)= $x^2 - 1$ in the interval $-1 \leq x \leq 1$ is the same as minimizing the function f(x)=1-x^2 over the same interval. Also since the GA expects the fitness function to always evaluate positive values, a constant may be added, for example, f(x)=20-($x^2 - 1$).

Using four degree of accuracy bit length for both x and y in these two example results in a string length of 17. A population size of 20, probability of crossover of 85%, probability of mutation of 1% and a generation number of 100 was used. An attempt to solve this problem with or without the improvements to the basic GA technique should yield acceptable results. An animation technique was created in order to illustrate how the GA search progresses as the number of generation increase. The global optimum found by the GA as well as the convergence history and 3D plot indicating by black dots points found by the GA are shown below.

The two test functions that are used to validate the working of the GA are given below.

(1) f(x,y)=$x\sin(4x)+1.1y\sin(2y)$ where $0 \leq x \leq 10$, $0 \leq y \leq 10$

```
(2) Peaks function
     f(x,y)  =   3*(1-x).^2.*exp(-(x.^2) - (y+1).^2) ...
    - 10*(x/5 - x.^3 - y.^5).*exp(-x.^2-y.^2) ...
  - 1/3*exp(-(x+1).^2 - y.^2)     where   -3≤ x≤3
                                          -3≤ y≤3
```

4.1.1 Results to the two functions outlined above

Test Function 1

$$f(x,y) = x\sin(4x) + 1.1y\sin(2y)$$

Function evaluated with the following schemes: Elitism, two-point crossover, one bit mutation, and tournament selection. Results are given in Figure 4.1.1 and Figure 4.1.2.

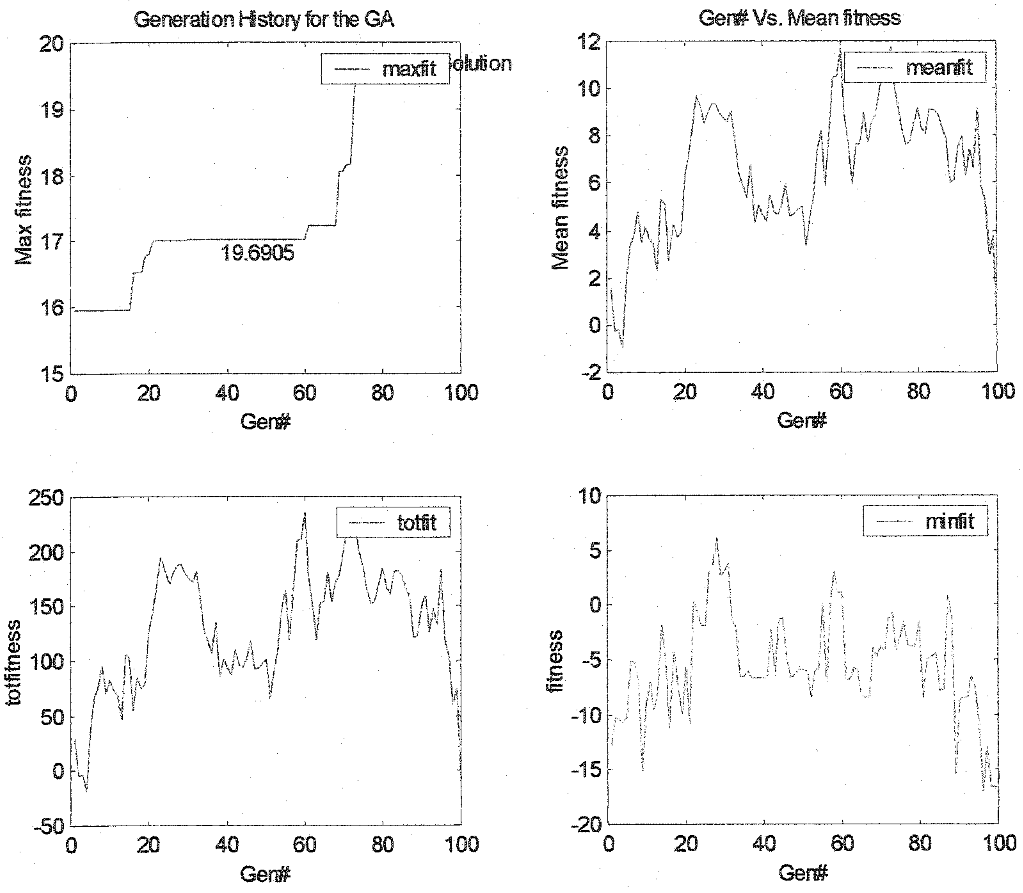


Figure 4.1.1: Convergence history of the GA for the above mathematical function

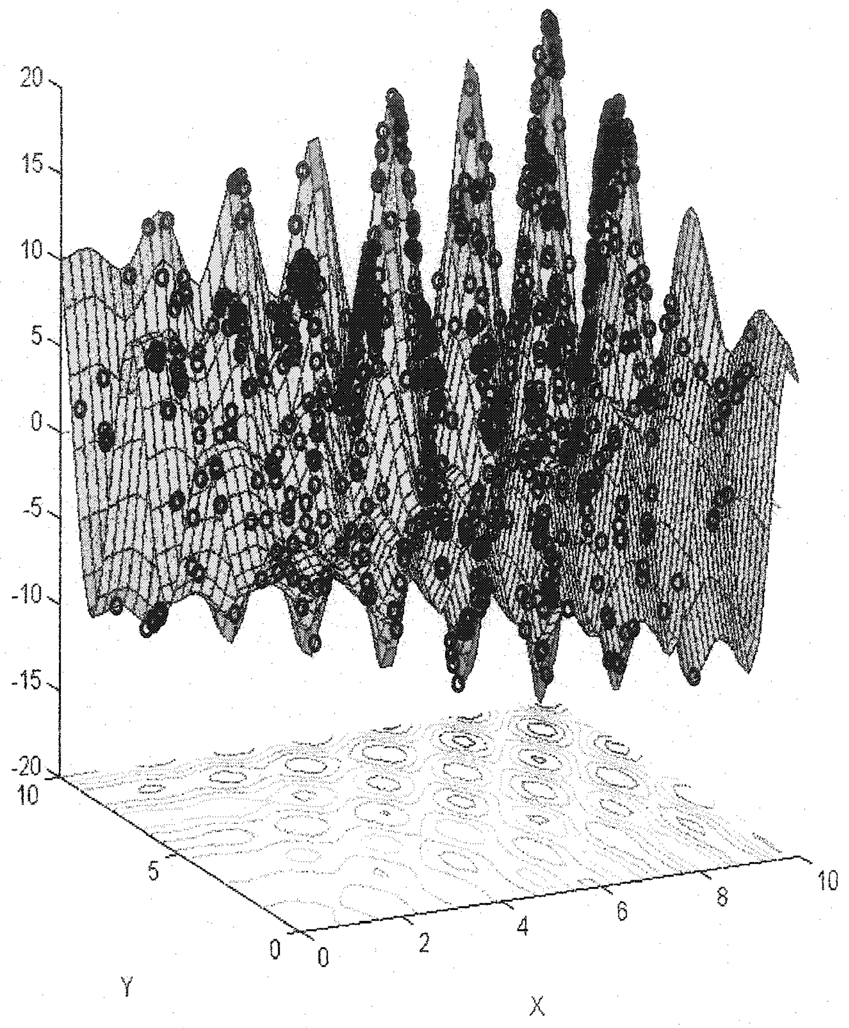


Figure 4.1.2: Tracing procedure that indicates the GA search history over the surface of the above function

Test Function 2

```
f(x,y) =  3*(1-x).^2.*exp(-(x.^2) - (y+1).^2) ...
        - 10*(x/5 - x.^3 - y.^5).*exp(-x.^2-y.^2) ...
   - 1/3*exp(-(x+1).^2 - y.^2)      where      -3≤ x≤3
                                               -3≤ y≤3
```

Function was evaluated with the following schemes: elitism, two-point crossover, one bit mutation, and tournament selection. Results are given in Figure 4.1.3 and Figure 4.1.4.

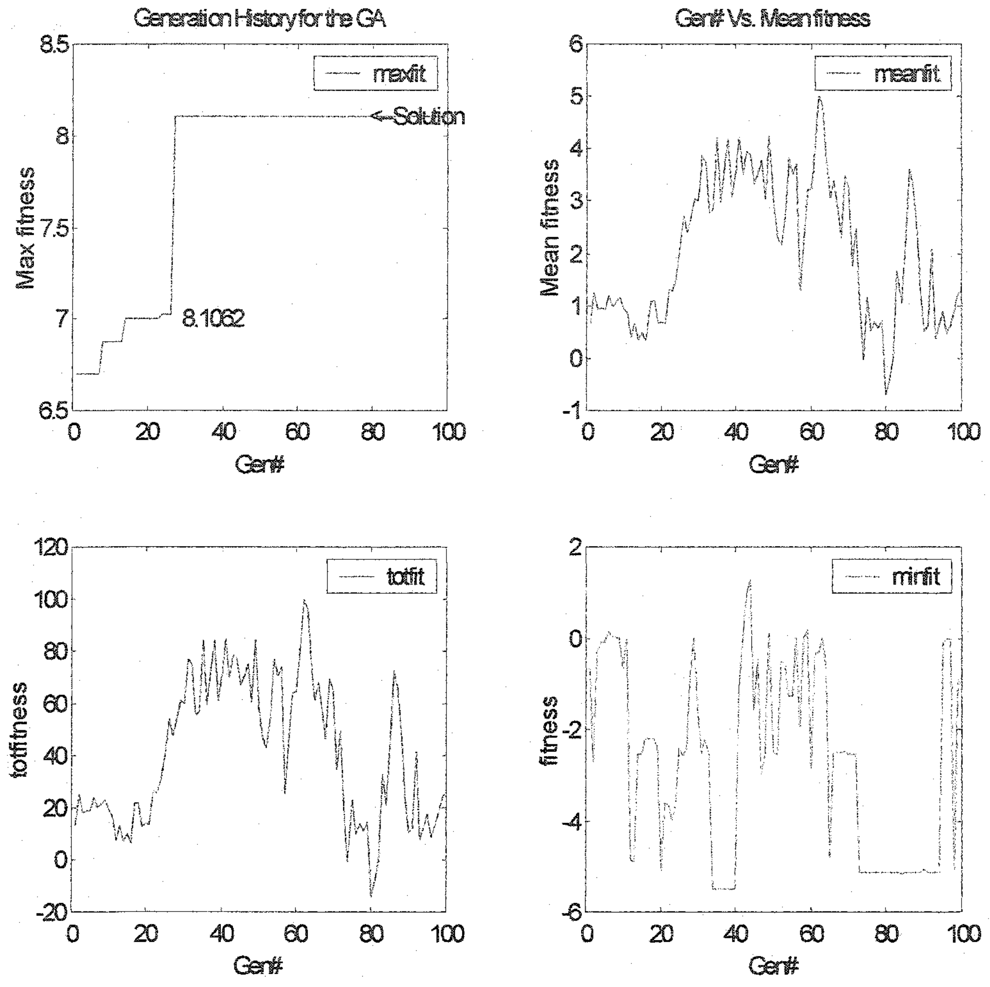


Figure 4.1.3: Convergence history of the GA for the above mathematical function

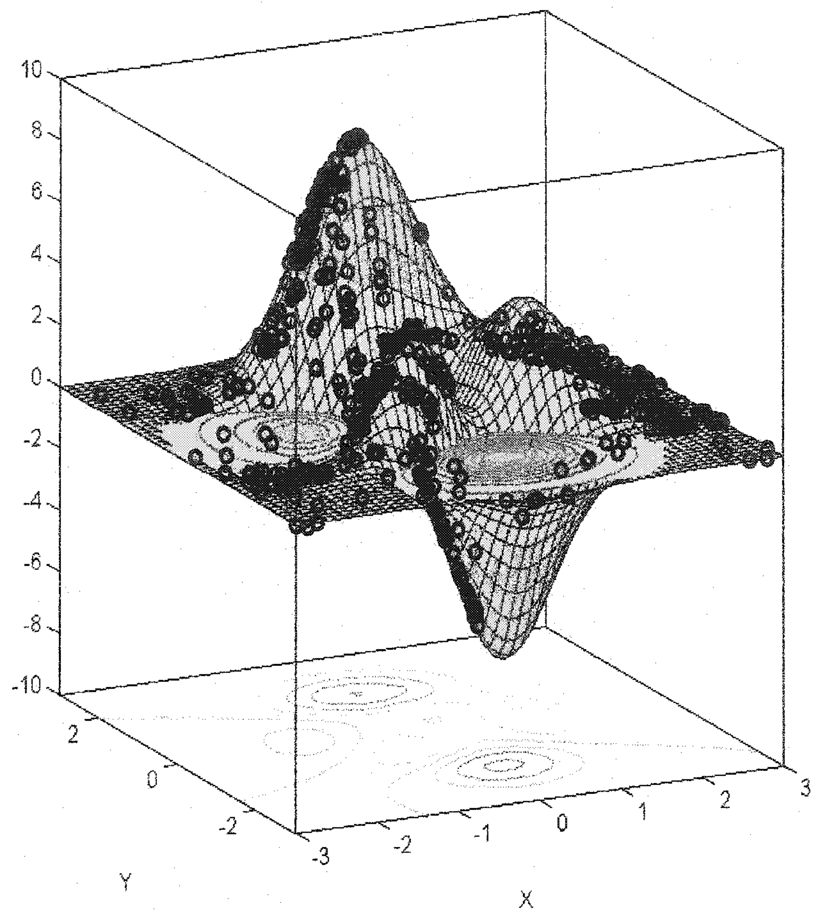


Figure 4.1.4: Tracing procedure that indicates the GA search history over the surface of the above function

It would be misleading not to mention that GAs are inherently a maximization tool, and do not handle negative numbers effectively. Thus, in order to minimize a function or weight/cost of a structure, some sort of scaling scheme must be added to change the maximizing problem into a minimizing problem. Procedures to handle this problem were addressed in Chapter 3. Note that in the above plots the red area is where the global maximum occurs, while the dark blue area is where the global minimum occurs. Also note that the GA is quite effective in locating the global optimum of both multimodal functions (notice the black dots that cover the red area). Since the GA was capable of finding the solution to the above functions, it is time to proceed to something more practical.

4.2 Optimizing the Volume of a Rectangular Cantilever Beam

Consider a simple rectangular cantilever beam with a prescribed load of P=1000N at the free end. We wish to minimize the volume of the beam with the maximum allowable stress of 3000 N/m^2 and maximum allowable deflection of 1mm. The length of the beam was assumed to be a constant, that is, L=50m. We are to constrain the base b and height h in range of 1 to 10 m and 1 to 5 m respectively. The objective is to find the minimum volume of the beam that will sustain the above load and satisfy the above constraints. This problem is used to illustrate how the above problem definition - and in fact, any problem definition - is integrated into the GA for optimization.

4.2.1 Problem Implementation into GA

In this problem we need a way to handle the constraints. The traditional approach to handling constraints already present in Chapter 3 is employed for this problem. The procedure for implementing this problem into the GA is highlighted below:

(1) Create the objective function: minimize (Volume) = bhL

(2) Decide upon which variables to encode (encode design/decision variable) e.g. b and h

(3) Write constraint equation e.g. stress= -My/I in terms of encoded variables listed in (2).

We get stress= 6PL/bh^2

(4) Finally the fitness function can be written as f (x) = bhL + pen

The mathematical formulation of the above problem procedures is given below along with sketch of cantilever beam (see Figure 4.2.1)

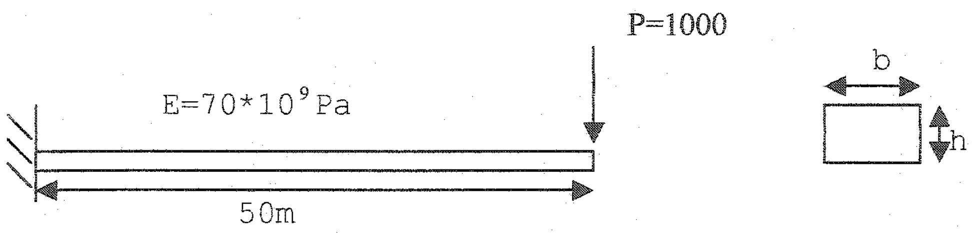


Figure 4.2.1: Sketch of the Cantilever Beam for Problem Define

Objective function: minimize (Volume) where Volume= bhL

Encoded/design variables: b and h

Constraints: Stress: $\sigma = \frac{My}{I} = \frac{6PL}{bh^2}$ $\qquad \sigma \leq 3000Pa$

Deflection: $\delta = \frac{PL^3}{3EI} = \frac{4PL^3}{bh^3E}$ $\qquad \delta \leq 0.001m$

Fitness= Volume + Penalty

where M is the bending moment, y is the distance from centroid to top of the beam, I is the moment of inertia, E is young's modulus of elasticity, b is the base of the beam, h is the height of the beam and L is the length of the beam.

4.2.2 Handling Constraints

The MATLAB program implementation to calculate the penalty value (that is 'pen' is step 4) above may be written as follows:

Let r_val1 be the first encoded GA design variable, that is, the base of the beam

Let r_val2 be the second encoded GA design variable, that is, the height of the beam

```
P=1000; % applied load
L=50; % length of the beam
r=[50, 10];  % multiplier constants → assumed
constraint1=3000; % max allowable stress in KPa to be < 3KPa
constraint2=0.001; %we want max deflection to be < 1mm
stress=(6*P*L)./( r_val1.*((r_val2).^2) );
deflection=(4*P*L^3)/( r_val1* r_val2^3*E);
g1= stress- constraint1;
g2= deflection- constraint2;
h1=max(0, g1);
h2=max(0, g2);
pen = r (1) *h1.^2 + r(2)*h2^2;  %penalty value needed for step 4
```

4.2.3 Results to the above GA problem implementation

The GA generated solution to the above problem after 100 generations is shown in Figure 4.2.2. The minimum volume found by the GA to satisfy constraints imposed on the problem with given loading condition is 2000.6 m^3 where the base of the beam is 8.005m and height of the beam is 4.9984m. Note that the design variables found exactly satisfy the constraints imposed, thus demonstrating that the GA handles the constraints quite effectively. In addition, the GA was capable of exploring the search space and arriving at a solution in less than ten seconds.

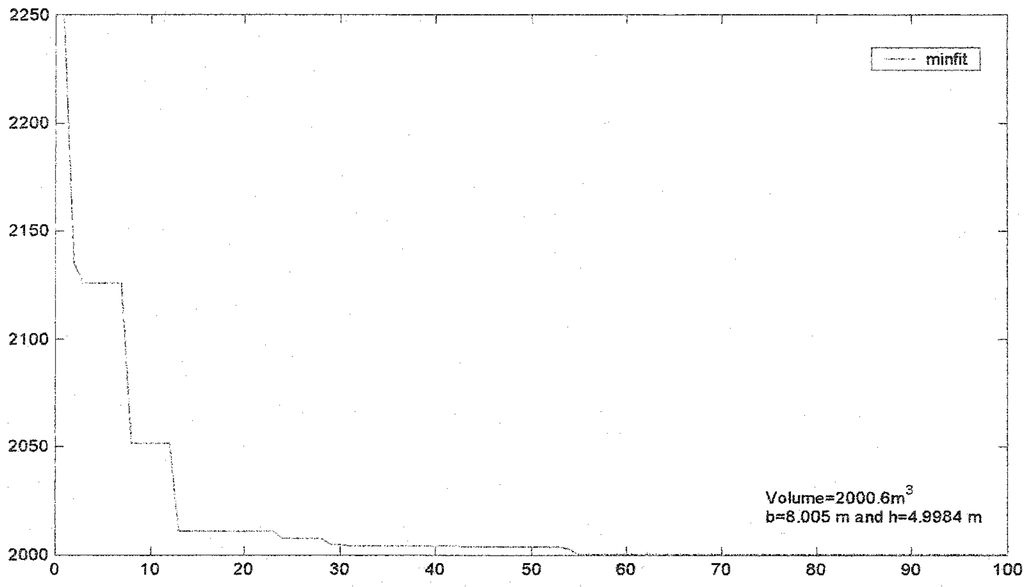


Figure 4.2.2: Convergence history of GA for optimizing the volume of a cantilever beam

4.3 Minimizing the Weight of a Scuba Diving Tank (Cylindrical Pressure Vessel)

In this problem a tank consisting of a cylinder and two hemispheres, one connected at either end of the tank, is considered. The thickness of the tank is allowed to vary between 0.1 to 1 inch and the radius of the tank varies from 1 to 12 inches. Length of the tank is a constant and was assumed to be 50 inches. The tank is made of aluminium 7075-T6 (sheet metal) density of 0.101 lb./in^3. Pressure in the tank is 500 lb./in^2. Some constraints imposed are that the tank must have factor of safety of 2.1 with respect to ultimate stress and 1.5 with respect to yield stress, where ultimate stress is $\sigma_u = 80Ksi(T)$ and yield stress is $\sigma_y = 70ksi$. Other constraints imposed on the above problem definition are given in the mathematical formulation presented below.

4.3.1 Problem Implementation into GA

Objective function: minimize (weight) weight $=\gamma V$

where the weight is approximated by: $weight = ((2\pi rt)L + 4/3\pi r^3)\rho$

Encoded/ design variables: r and t

Constraints: Stress allowable: $\sigma_{allow} = \dfrac{\sigma_u}{2.1} = 38.097Ksi$

Stress allowable: $\sigma_{allow} = \dfrac{\sigma_y}{1.5} = 46.67Ksi$

Stress allowable =38.097 is the lower stress value therefore it governs the design.

$$\sigma \leq \sigma_{all} = 38.097ksi$$

Since hoop stress, $\sigma_H < \sigma_a$ the axial stress, then the tank will fail in the hoop region first

Therefore first constraint is: (i) $\sigma_H = \dfrac{\Pr}{2t} < \sigma_{all}$

(ii) Second constraint is on the aspect ratio that validates the above formulas.

Aspect ratio $= \dfrac{r}{t} \geq 10$ where r is the radius of the tank and t the thickness of the tank.

(iii) Since axial force, f_a, is less than hoop force, f_s, it will govern the design:

$$f_s = \sigma_H t = \frac{\Pr}{2t}t < 800lb$$

Fitness= weight + penalty

where r is the radius of the tank, t is the thickness of the tank wall, L is the length of the tank, ρ is the density of the tank material, V is the volume of the tank, γ is the specific gravity and P is the pressure inside the tank.

4.3.2 Results to the above GA problem implementation

The solution to the above problem after 50 generations is given in Figure 4.3.1. The minimum weight found by the GA is 40.0549 lbs. with thickness of 0.3 inch and radius of 2.9966 inches. Note this problem was designed simply as a means to test the capability of the GA to solve problems. For practical purposes the geometry of scuba tank found may not be useful, primarily because the constraints and the length of tank used may not be practical, however; the solution does prove that the GA is an effective tool for optimizing a wide range of problems.

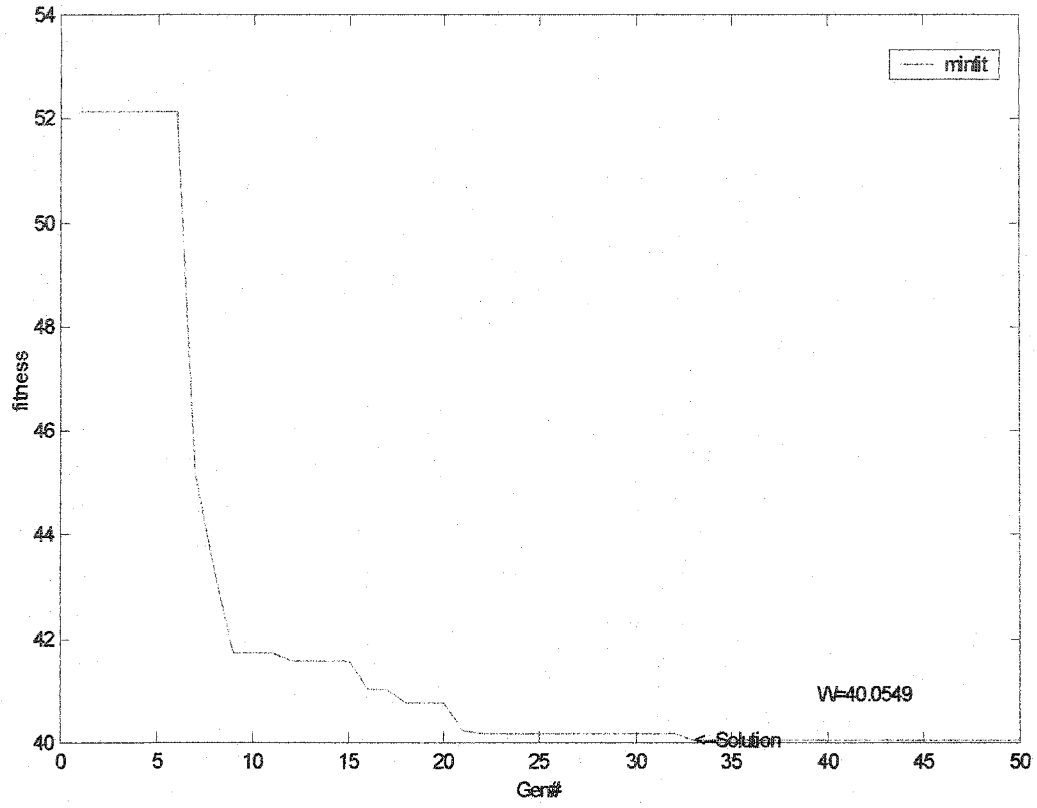


Figure 4.3.1: Convergence history of the GA for optimizing the weight of a cylindrical tank

4.4 Discussions and Conclusion

The result from the three problems solve above acted as reassurance that the GA is functioning as expected. Moreover, the GA generated results indicated that the algorithm was quite capable of solving a range of problems, and more importantly, exploring and exploiting a vast search space so as to avoid falling into the trap of local optima. Hence, armed with this knowledge it is time to fully test the potential of the GA by proceeding to more complex mechanical engineering problems to fully ascertain robustness and versatility of the genetic algorithms.

Chapter 5

Conceptual Aircraft Design – A Genetic Search and Optimization Approach

5.1 Summary

With recent advancements of computers and the advent of a search and optimization tool such as the genetic algorithm (GA), the manipulation of numerous aircraft design parameters in reasonable amount of time becomes feasible. It is from this standpoint that when one examines the aircraft design process, that is, the lengthy time and effort spent creating and integrating aerodynamics codes, sizing routines and performance modules, that the GA becomes beneficial. Consequently, a GA was created and employed as a tool to explore possible aircraft geometries that are more efficient and less costly than an existing design. The adaptive penalty method is employed in the GA to handle all constraints imposed on the design. In addition, the effects of adjusting varying degree of selection and crossover intensities and types on the aircraft evolutionary process are studied. A design study is also conducted to compare the GA optimized aircraft shape and configuration with that of the existing aircraft. Results indicated that the GA is a powerful multi-disciplinary optimization and search tool, that is capable of managing and reforming numerous aircraft design parameters, to arrive at aircraft conceptual designs that are both efficient and cost effective.

5.2 Introduction

A word that often comes to mind in aircraft design is compromise. In fact, it can be argued that the entire aircraft design process entails finding the correct balance or compromise between numerous design variables and design constraints. Moreover, the traditional design technique usually requires the expertise of designers and engineers in order to arrive at an efficient design. The pitfall of such an approach is that it relies entirely on the designer's and engineer's knowledge and creativity. Hence the problem of human error cannot be ruled out. Unfortunately, each design discipline is limited to a subset of configuration parameters, goals, and constraints due to the highly interdisciplinary process, which involves a high number of variable couplings [64]. Hence, due to the processes entailed in the traditional design approach, the designer may not fully grasp how the design objectives conflicts with another. Finally, the traditional design approach entails lengthy meeting time and large sums of money to create and integrate the various disciplines of aircraft design.

On the other hand, the GA was developed to eliminate or at best mitigate the difficulties, time and costs encountered in the traditional design approach. It can then be argued that the genetic algorithm (GA) is one of the most commonly used evolutionary algorithms.

A few of the studies that are relevant to aircraft design and that have been successful in applying genetic algorithms to optimization are listed below. Quagliarella and Cioppa [8]; Perez et al.[84]; Anderson and Gebert [10]; Marx et al. [11]; Ewing and Downs [12]; Tse and Chan [13]; Butler et al. [14]; and Gage et al.[15], were all successful in applying GA to study different design aspects of the aircraft wing. Tong et al. [16]; Mosetti and Poloni [17]; and Hajela [18] were also instrumental in applying the GA to study complex design problems in the aerospace industry. Furthermore, Crossley and Laananen [19] were successful in applying the GA to study the conceptual design of helicopters. In addition, Sobieski et al. [20]; Gallman and Mason [21]; Malone and Mason [22]; Crispin [23]; and Bos [24] were all able to utilize the GA to study aircraft design.
In this chapter, special attention is directed to conceptual aircraft geometrical optimization – a design phase that is critical to the preliminary stage of the design process. Consequently, a

GA was created and employed as a tool to explore possible aircraft geometries that are more efficient and less costly than an existing design.

Tremendous detailed information about genetic algorithms, its creation and implementation to solve problems, and its choice of use for this thesis was already presented in Chapter 3. For detailed information about genetic operators and schemes used for this study, please consult Chapter 3. The MATLAB [73] coded binary genetic algorithm created was utilized to explore its capabilities as an aircraft conceptual design tool.

5.3 Problem Statement

This chapter brings a focus on the application of the GA to the conceptual design of an aircraft with constraints in sizing, performance, and stability and control. In addition, a study was conducted to analyze the evolution of the aircraft as the GA search progresses. The design variables chosen for optimization are given in Table 5.3.1. The range of these variables is determined from historical and statistical data. The length of the bit string is determined from the range of these variables and the degree of accuracy required. The twenty-one design variables along with the design process and constraints utilized by the GA to find the correct balance or combinations of design variables is presented in a later section. Weight was employed as the main parameter of study in the aircraft fitness function to optimize. A reduction in weight and the computing time required to arrive at efficient conceptual aircraft designs are the main goals of this study.

5.3.1 Design Goals

The objective of this chapter is to find the right combination of design variables that will provide the initial design layout to speed up the aircraft conceptual design process. This objective must be achieved with minimal constraint violations and a combination of variables that will lead to a practical and feasible conceptual aircraft designs.

The main optimization goal is to minimize the aircraft weight (W). The aircraft weight is directly related to the total cost of the aircraft. Furthermore, it can be argued that an aerospace structure is a structure whose usefulness diminishes significantly with increasing weight [85].

The model formulation is as follow:

Objective function:

$$\text{Minimize } f(x) = W * p(x) \qquad (1)$$

subjected to constraints in performance, aerodynamics, sizing, and stability and control.

The total weight, W, includes the fuel weight, aircraft empty weight, and payload. Fuel weight is calculated based on the fuel fraction used in each segment of the mission profile. In addition, the weight attributed to 5% fuel reserve and 1% trapped fuel is taken into account. Aircraft empty weight is the sum of all structural components, and is calculated using a component weight estimation method developed in [85, 86]. The payload consists of a crew of 6 and 110 passengers, all weighing 150 lbs. In addition, all passengers are allowed baggage weight of 80 lbs. p(x) is the penalty value added for any constraint violation and is calculated as shown in Chapter 3.

Table 5.3.1. Critical Aircraft Design Variables

Design Variable	Variable Type	Variable Description	Variable Domain	String Length (# of bits)
W/S	(c)	Wing loading (lb/ft^2)	70-140	8
T/W	(c)	Thrust Weight Ratio	0.3-0.5	5
$C_{l\max}$	(c)	Maximum Airfoil Lift	1-1.5	3
AR_w	(c)	Wing Aspect Ratio	6-12	7
λ_w	(c)	Wing Taper Angle	0.1-0.9	6
Λ_w	(c)	Wing sweep angle (deg)	0-50	9
t/c	(c)	Wing thickness	0.1-0.15	3
AR_{ht}	(c)	Horizontal tail Aspect Ratio	3-5	5
λ_{ht}	(c)	Horizontal tail Taper Ratio	0.2-0.6	4
Λ_{ht}	(c)	Horizontal tail Sweep angle (deg)	5-45	7
T_{Type}	(d)	Tail Type T Tail–0 Conventional-1	0-1	2
AR_{vt}	(c)	Vertical tail Aspect Ratio	0.7-1.2	1
λ_{vt}	(c)	Vertical tail Taper angle	0.3-0.9	5
Λ_{vt}	(c)	Vertical tail Sweep angle (deg)	35-55	4
N_{eng}	(i)	Number of Engines	2-4	7
Material	(i)	Material Type 0-conventional 1-Composite	0-1	1
$Nseats$	(d)	Seating arrangement (1-11)	2-11	4
l_t	(c)	Horizontal tail arm (ft)	40-100	8
S_t	(c)	Horizontal tail area (ft^2)	200-700	10
$C_{h_{\delta e}}$	(c)	Hinge moment due to Elevator deflection	-0.1 - -1	4
l_v	(c)	Vertical tail arm (ft)	40-100	8

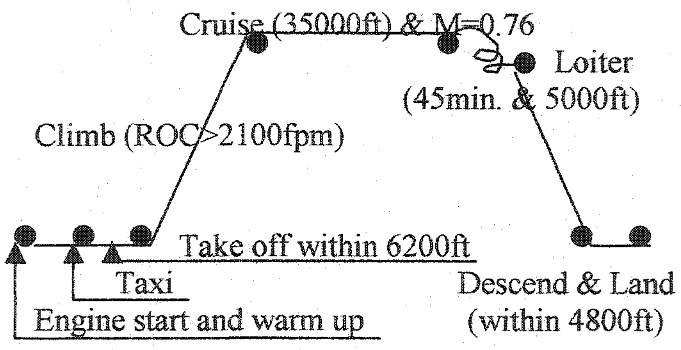
Note: c-continuous, i-integer, d-discrete

An analytical and graphical comparative study has also been conducted to check the GA design attributes with that of the existing aircraft. For this study, a fitness module, which includes a sizing and constraint routine, has been given.

Note that the objective function is not the cost since this value alters considerably with the economy over time. Moreover, the cost of the aircraft is also influenced by factors such as material, machining, maintenance and operations, which also fluctuate with the economy over time. Nonetheless, there are crude estimations that can be made to formulate cost as the objective function. The crudeness of these estimations coupled with the time constraints has discouraged the author from implementing cost as the objective function in this study. Readers interested in incorporating cost into aircraft design should consult Ref. 87.

5.3.2 Mission Profile

The mission profile is based on the type of aircraft used for study, in this case the Boeing 717. The Boeing 717 is a medium size transport aircraft [88] and a typical mission profile is illustrated in Figure 5.3.1. For this study the number of passengers and crew is fixed at 110 and 6 respectively. In addition the average weight of crew and passenger is implemented into the program.



Payload 26950lbs. & $n_{ult.} = 3.5$ g's

Figure 5.3.1: Transport aircraft mission profile

Since a medium size transport aircraft is used in this study, FAR 25 regulations have been adapted to govern the design process. The mission profile also yields some constraints that must be imposed on the aircraft design, such as takeoff with a runway length of 7000ft at standard sea level, climb to a certain altitude, and land within a runway length of 5000 ft.

5.3.3 Fitness evaluation

The fitness function is geared towards weight minimization with adaptive penalty strategy employed to handle the constraints. In effect a multidisciplinary design optimization (MDO) approach is taken whereby geometry, sizing, aerodynamics, performance and stability and control are analyzed and integrated.

Briefly the sizing routine is based on the 'rubber engine sizing methodology'. The aerodynamic characteristics are calculated using detail component buildup method adapted from Ref. 85. Performance characteristics for the aircraft brings a focus on satisfying federal aviation requirements during takeoff, climb, descent and landing as adapted from references 85-91. Stability and control analysis is conducted assuming a minimum value for the static margin. This design discipline strongly correlates the aircraft geometry and layout. In all of the above aircraft design disciplines a total of fifteen design constraints are implemented. The total weight of the aircraft can be subdivided in three main areas i.e. fuel weight, payload and empty weight of the aircraft.

The fuel weight is determined from the mission objectives and the intended use of the aircraft. Fuel fractions are then estimated using the mission profile and fuel consumption based on performance analysis. The weight of the payload is constant and is directly related to the number of passengers, crew and cargo. The empty weight is estimated using the component buildup method, with estimation techniques adapted from Ref. 85, 86 and 87.

5.3.4 Design Constraints

The mission profile that has been employed is typical of a medium size commercial aircraft i.e. warm up, taxi, takeoff, climb, cruise, loiter, descent and land. Governing transportation rules produce certain constraints on some of the aforementioned flight segments. For example, for this particular type of aircraft, it is required to have a minimum field takeoff run distance where the maximum field run length is specified by governing bodies such as Federal Aviation Rules, FAR 25 [85 and 87]. The total takeoff distance can be converted in a penalty term as follows:

$$g_i = \left(0, \frac{(W/S)_{TO}}{TOP\,\sigma C_{L_{TO}}} \right) \qquad (2)$$

where g_i = the degree of violation and is zero if the total aircraft field length is less than the maximum field run length specified by FAR, or greater than zero if otherwise. $\left(W/S\right)_{TO}$ = wing loading at takeoff; σ =density ratio between takeoff altitude and sea level, $C_{L_{TO}}$ =

coefficient of lift at takeoff and TOP is the takeoff parameter used to estimate balanced field runs depending on aircraft type, propulsion and regulations. A similar constraint implementation is performed for climb, cruise, landing, sizing, aerodynamics, performance and stability and control. All constraints have been treated equally, and the severity of the penalty value imposed on constraints violated can be controlled by α.

5.4 Results and Comparison of GA Optimized Designs with Boeing 717

Table 5.4.1 and Table 5.4.2 are based on a population size of forty, with two hundred generations employed in each run. Table 5.4.1 and 5.4.2 list the best designs found from a sample of five tests. The other four tests are neglected in this work since they do not aid in understanding of the results. A fixed probability of crossover and probability of mutation of 80% and 1% respectively, has been implemented. In addition, elitism strategy has been used and inverse scheme adopted to scale the fitness values.

A plot of the generation history for the best designs found from the five runs is shown in Figure 5.4.1. From Figure 5.4.1 it can be seen that the search converges upon an optimum area of the design space after approximately 80 generations. From this point onward the GA attempts to "fine-tune" or focus in on this converged area to find the global optimum. Running the GA further, it may be possible to find this optimum, but this task can be computationally expensive. However, in most cases finding the global optimum is of little practical use. Consequently, one major disadvantage of the GA is that convergence speed near the global optimum is slow.

Table 5.4.1. GA Optimized Conceptual Aircraft Designs

	Run 1	Run 2	Run 3	Run 4	Run 5	B-717
Crossover Type	1 point	2 point	Uniform	Uniform	2 point	
Selection Type	Tourn.	Tourn.	Tourn.	Roul.	Roul.	
Design	Variables					
W/S	96.63	108.71	102.67	103.76	87.84	115
$C_{l\,\max}$	1.5	1.5	1.43	1.43	1.5	1.4
AR_w	6	6.05	6.38	6.66	6.14	8.7
λ_w	0.32	0.62	0.47	0.30	0.31	0.38
Λ_w	24.66	24.17	26.81	22.80	30.82	26.9
t/c	0.1	0.1	0.11	0.1	0.13	0.11
AR_{ht}	4.29	3.19	3.19	3.97	4.68	5.21
λ_{ht}	0.49	0.31	0.45	0.41	0.47	0.4
Λ_{ht}	12.87	12.56	36.02	7.21	32.72	36.87
T_{Type}	1	0	0	1	0	0
AR_{vt}	2	2	2	2	2	1.05
λ_{vt}	0.55	0.78	0.34	0.73	0.40	0.8
Λ_{vt}	40.33	35	45.67	40.33	37.68	41
N_{seats}	5	6	5	5	5	5
l_t	74.73	73.22	70.88	70.74	73.90	N/A
S_t	201.47	207.47	216.35	225.06	203.42	N/A
l_v	73.77	69.51	61.412	65.80	74.31	N/A
Wo	103330	103341	104832	104725	105918	115000
Constraint Violation %	11.33%	10.24%	9.28%	9.92%	9.24%	

Note: Tourn. – Tournament Selection
Roul. – Roulette Selection

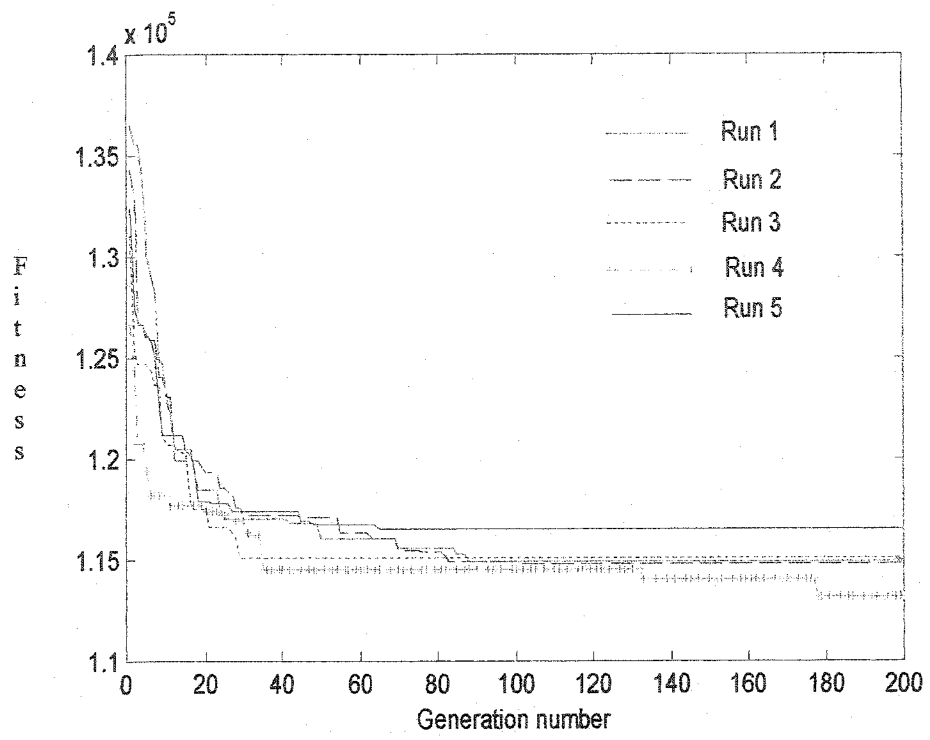


Figure 5.4.1: Convergence history for genetic algorithm conceptual aircraft designs for run 1 to 5

Table 5.4.2. Other Affiliated GA Optimized Design Variables

	Run 1	Run 2	Run 3	Run 4	Run 5
Crossover Type	1 point	2 point	Uniform	Uniform	2 point
Selection Type	Tourn.	Tourn.	Tourn.	Roul.	Roul.
Design	Variables				
C_r	20.24	15.44	17.24	18.89	21.39
C_t	6.46	9.6	8.07	5.73	6.62
b	80.1	75.82	80.71	81.99	86.05
S	1069.3	950.59	1021	1009.3	1205.7
l_{fus}	109.74	105.68	109.74	109.74	109.74
V_h	0.971	1.220	1.059	1.17	0.814
V_v	0.088	0.082	0.089	0.089	0.082
F	38.97	65.48	44.81	59.41	37.67
$(X_{cg}/\bar{c})_{for}$	0.51	0.53	0.49	0.56	0.47
$(X_{cg}/\bar{c})_{aft}$	0.58	0.63	0.58	0.66	0.53

Chapter 2 addressed the reasons why researchers are usually content with sub-optimal solutions and so these comments are not repeated here. Nonetheless, acceptance of these designs is justified by observing that in the real world problems of imperfect models, unavoidable numerical inaccuracies, and time constraints, insisting upon optimal solution is entirely inappropriate [53].

Results indicate that uniform crossover coupled with tournament selection prove to be the best genetic operators. Uniform crossover encouraged the greatest information exchange between designs, while tournament selection closely imitated mating competition in nature.

One-point and two-point crossover schemes even though useful were not effective in quickly exploiting and exploring the search space. This is partly because the bit string that represents each design was large and there is not enough information exchange facilitated by these genetic schemes for the short generation time under study.

Roulette selection scheme fused with any crossover scheme was also ineffective. This is mainly because roulette selection has some undesirable features. With roulette selection, there is a tendency for a few super chromosomes to dominate the selection process; in later generations, when the population is largely converged, competition between chromosomes is less strong and a random search behaviour emerges [79].

The aircraft conceptual geometry design variables found and plotted agree within the scope of aircraft design. This implies that apart from finding a design that yields a reasonable weight, it is important for the GA to arrive at a design that "looks good" or looks like an aircraft. This can only be achieved if the constraints imposed on the design are reasonably satisfied. As Table 5.4.1 reveals, the adaptive penalty method is effective in enforcing the design constraints. Table 5.4.1 also suggests that all designs yield some degree of constraint violation and therefore lack some degree of feasibility. Increasing α can mitigate the degree of violation of the constraints.

5.4.1 Evolution of the aircraft design

Figure 5.4.2 to Figure 5.4.5 below shows how the top view of aircraft configuration evolves as the GA search progresses. Since the GA is a population-based search the best design based on the fitness value was selected and plotted. The design at generation number 5, 20, 60 and 120 are presented for run 3 to illustrate the GA evolutionary design process. Note that after generation number 120 all the runs reached a steady state condition. The aircraft designs exhibited little change, since the GA had converged onto a near optimum design and was proceeding slowly to the global optimum.

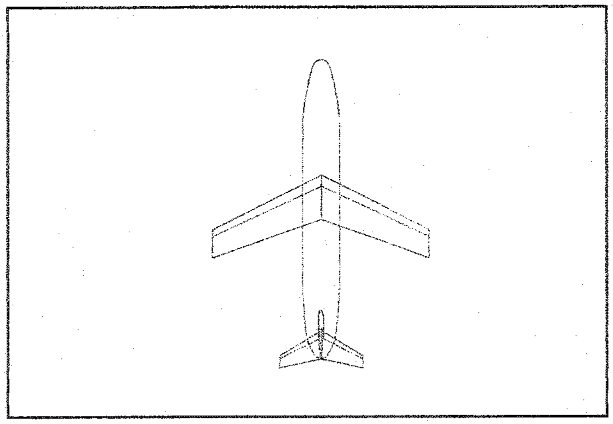


Figure 5.4.2: Best design at generation 5

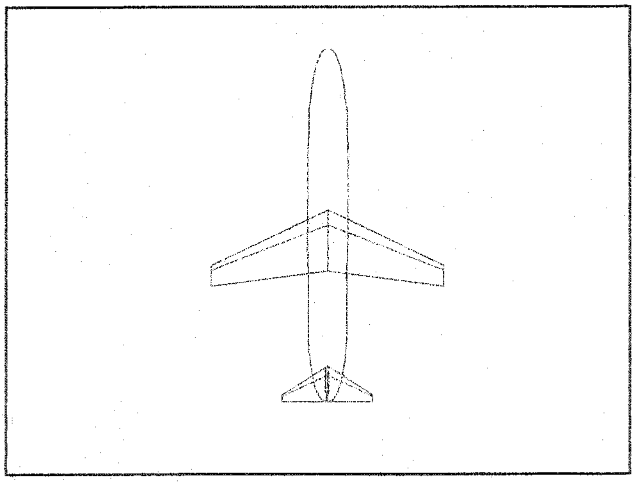


Figure 5.4.3: Best design at generation 20

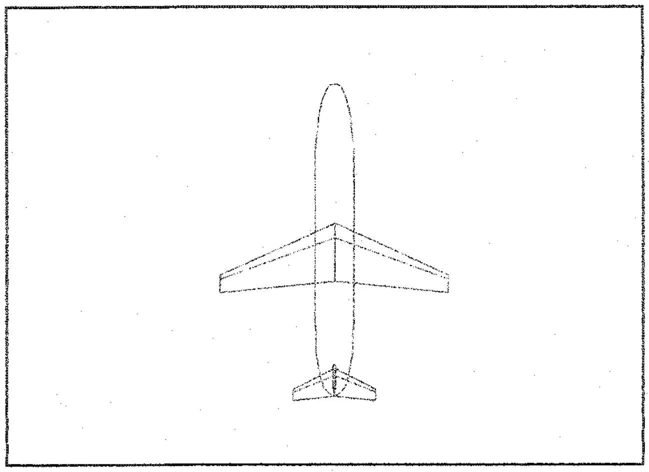


Figure 5.4.4: Best design at generation 60

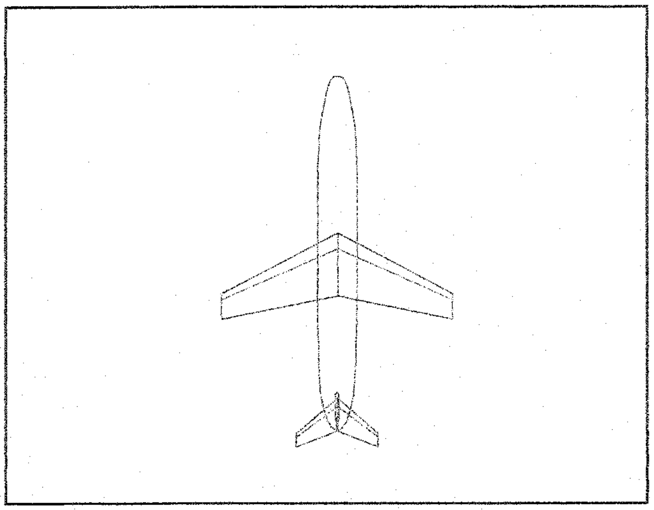


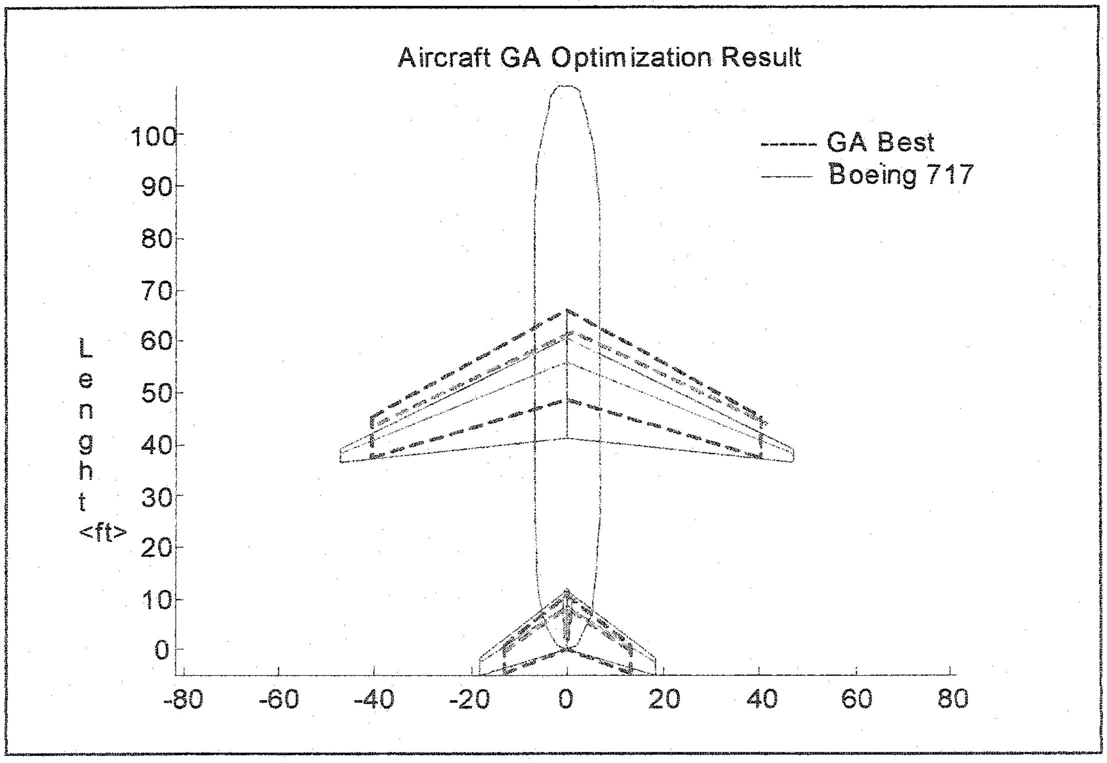
Figure 5.4.5: Best design at generation 120

Figure 5.4.2 to Figure 5.4.5 above suggests that the GA advance from an impractical design to a conceptual aircraft design that is reasonable. In other words, the GA capability of learning is dependent on time. This fact transcends to our every day life in which our learning abilities are influence by a time factor. In addition, the evolved design coupled with reductions in constraints violation somewhat reinforces the fact that convergence towards optimal or slight optimal solutions is taking place and perhaps can be reached.

Note that even though the genetic operators used for run number 4 are not the most efficient, the GA is capable of finding a reasonable conceptual aircraft design in a small amount of time. The design in run 4 is not too efficient partly due to the selection scheme employed.

It can be argued that the design found in run number 3 is the best of the five runs since this run offers the best combination of design variables with a low weight and small constraint violation. Uniform crossover and tournament selection therefore were the most effective schemes in exploring the search space and locating the most interesting areas of this complex design space, which is the major goal of conceptual design. The reason being uniform crossover encourages the greatest amount of information exchange when compared to 1-point or 2-point crossover, while tournament selection serves a vital role of imitating mating competition in nature more closely than roulette selection.

The best aircraft design layout at the end of run 4 (at generation 200) is also plotted and a graphical comparison with the existing design is indicated in Figure 5.4.6. Take note of the variations in dimensions and layout of the GA evolved aircraft with the existing design.



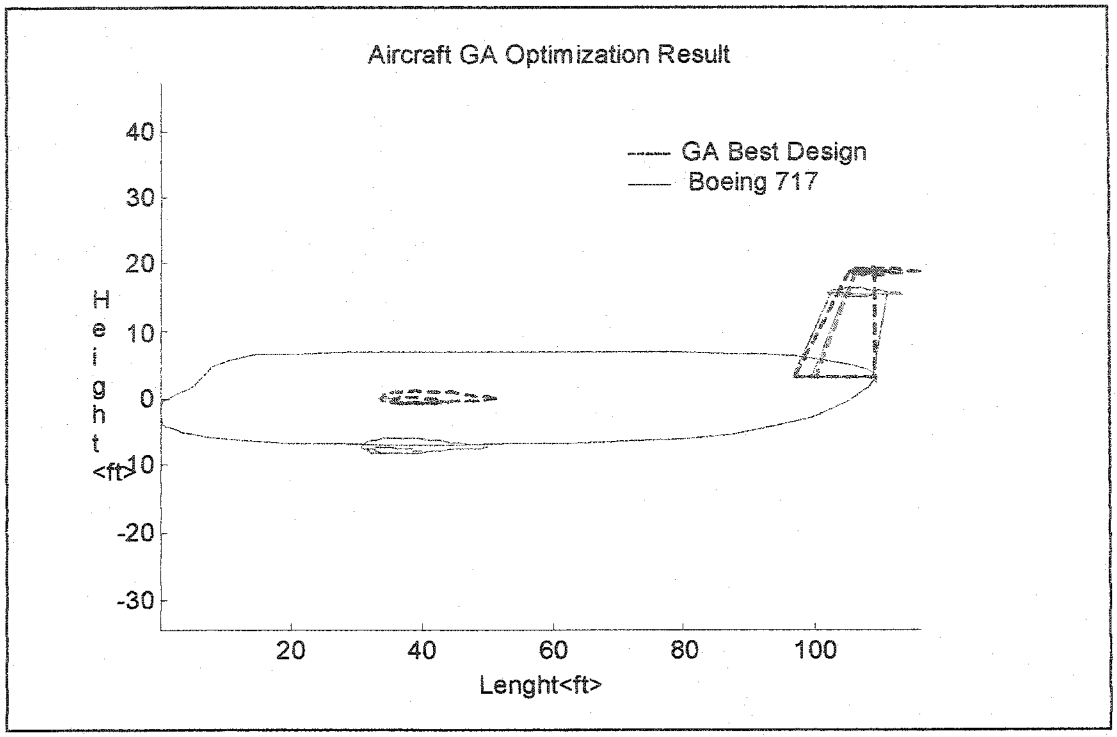


Figure 5.4.6: Top view and side view comparison of optimized aircraft found by the GA with existing design.

The fuselage dimensions are similar to the existing aircraft since the number of seats found by the GA is identical. Significant changes in horizontal and vertical tail arm, reduction in tail area, and wing location should be noted.

A reduction in wingspan leads to a reduction in aspect ratio of the aircraft; see Table 5.4.1 and Table 5.4.2. Low aspect ratios result in more pronounced induced flow effects, which reduce the peak velocities on the wing surface [87, 90]. Hence, critical Mach numbers will increase with decreasing aspect ratio. A low aspect ratio also results in a higher induced drag and a reduced lift curve slope. To alleviate such a problem, one can implement end plates at the wing tips [89, 90]. In addition, a second segment climb constraint that has been neglected in this study can be included to increase the aspect ratio. The consequences of high induced drag and low lift curve slope are that higher angles of attack will be required for takeoff and landing, which can present such problems as tail scraping and poor pilot visibility [90, 91]. Moreover, increased induced drag can also translate into increase fuel consumption. The difficulties encountered above are the result of reduced aspect ratio that has not been accounted for in the design methodology.

5.5 Discussion and Conclusions

Clearly, the GA can provide a reasonable aircraft design in a short amount of time compared with the traditional design techniques. Moreover, the GA have the ability of simultaneously learning the task from all design disciplines as well as integrating the design parameters from these disciplines to arrive at efficient aircraft designs. In addition the GA was capable of handling numerous design variables, design formulations and design constraints. Hence in some ways the GA can replicate the task assign to a design team and in some cases underscore some crucial design areas that may be overlooked by a traditional design approach. On the other hand, the convergence history of the GA is slow. As a result, solutions obtained may be indicative of a simplified version of the true optimum. This is as a result of problems arising out of questions such as how large the population should be or what crossover or mutation rate to use? Or perhaps what can be done to better the GA selection process? These questions are vital since they determine the convergence quality of the GA and can play a major role in the GA decision-making.
Moreover, there is no proof of optimality with the GA generated designs since there is no gradient information available. Hence, one general limitation to the GA is the lack of simple and reliable convergence criteria. However, this lack of mathematical proof inherent in the GA is, to some extent, traded off for the ability to search a large complex search space employing numerous constraints and design variables.
As indicated in the GA convergence history, the GA seem to get close to the optimum but takes a long time to locate this point in the search space. Thus it may be more fruitful to run several near optimum design and later combine different aspects of these various designs to achieve an optimal conceptual design.

Finally this study indicated that genetic algorithms are a flexible and efficient means of generating conceptual aircraft designs that can increase the scope and decrease the time and cost entailed in the traditional design approach. As a result, the proposed methodology can provide initial layouts and geometries of reasonable conceptual aircraft design maps – given a particular mission profile – to initiate a speedy design process.

This study also suggests that the GA is capable of handling the complex problem of simultaneously managing and integrating many design disciplines. The aircraft conceptual designs found in this study are different from the existing design layout of the Boeing 717. This difference in most cases provides a favourable balance between aircraft parameters that lead to a lighter aircraft geometry and configuration than that of the existing design. From this standpoint, serious consideration should be paid to designs found in this and other research utilizing GAs, since such research acts as a starting point of generating a design map to meet the required design objective in a short period of time – the main objective of conceptual design.

Chapter 6

Applicability and Viability of a GA based Finite Element Analysis Architecture for Structural Design Optimization

In this chapter, a genetic algorithm (GA) based finite element analysis (FEA) methodology was developed for size and shape optimization of planar and space trusses. The proposed procedure interfaces a binary GA within a FEA software package in order to initially test the applicability and viability of such integration. In addition, special features of the GA were included to dynamically alter the population size, and the crossover and mutation rate in order to facilitate faster convergence and hence reduce the computational effort required. In other words, the GA adapted itself as the search and optimization process progressed. This chapter also brings a focus on the applicability of integrating a GA as an optimization tool within a FEA software. It was shown by way of many examples –solved by numerous mathematical, as well as other heuristic approaches in the literature- that the proposed methodology is quite efficient and capable of finding lighter and reasonable structural designs than that reported in the literature. Moreover, it was shown that the proposed method removes the immense effort required in coding ones own finite element codes by utilizing already existing finite element software. Nonetheless, it was found that even with a GA, optimization for very large problems was computationally extensive.

6.1 Introduction

Structural optimization has been and continues to be a large area of active research. Vanderplaats [92] appeared to be among few whom was successful in applying numerical approaches such as gradient method, sequential quadratic programming, and approximation methods to the optimization of structures. Many of these techniques were based on Taylor series expansion of the objective and constraint function with respect to the design variables.

Typically, Taylor series expansion is used to linearize a non-linear system in order to apply some operations research technique. It is important to note that it is these same numerical methods that are present in some FEA software such as ANSYS [74]. Apart from Vanderplaats [92], Salajegheh et al. [93] and Hansen et al. [94] also followed a similar approach using an approximation method based on first order Taylor series expansion of the member forces. In addition, Smith and Miura [95] also used an approximate technique based on Taylor series expansion for the sizing problem. Imai [96] used numerical methods based on Langrange multipliers often referred to as the primal-dual method. This method is capable of handling both equality and inequality constraints.

Application of linear programming techniques to structural optimization was also heavily studied by Belegundu [97], Qian et al. [98], Hall et al. [99], Erbutur and Al-Hussainy [100], just to name a few. To date, no single mathematical method has been found to be entirely efficient and robust for the entire range of engineering optimization problems.

Apart from the above mathematical programming based approaches, optimality criterion methods that utilize the Kuhn Tucker condition with Langrangian multiplier was also used for structural optimization by Gellaty and Berke [101], Venkayya et al [102], Allwood and Chung [103], Kirsch [104], Chang [105], Rozvany and Zhou [106], just to name a few. In structural optimization it is well known that optimality criteria (OC) techniques and mathematical programming (MP) based methods have received great interest and application during the last several decades [83].

The complexity of implementation of structural optimization numerical methods, the need for derivatives, and the deterministic and exhaustive nature, has made MP and OC methods less attractive. Consequently, research was oriented in the direction of improving the above techniques and/or perhaps finding better approaches of dealing with the aforementioned problems in structural optimization. Soft computing techniques such as the GA were found to behave efficiently over a wide range of applications, even when employing numerous design variables and constraints.

Goldberg [25], – a pioneer in the work of GA- and Samtani appeared to be the first that employed the GA for structural optimization. A 10 bar truss problem – which is today a benchmark problem in structural optimization - was utilized to study the GA's role in structural optimization. Jenkins [26] utilized a simple GA created by Goldberg [27] to study a truss- beam roof structure. Hajela and Shih [28] looked into the applicability of GA as a function optimizer for structural optimization. Rajeev and Krishnamoorty [29] proposed a simple discrete GA for optimizing truss structures. Rajan [30] extended the magnitude of the problem by combining size, shape and topology optimization of trusses using a modified GA. Deb and Gulati [31] also proposed a GA for size, shape and topology optimization using a real-coded GA. Camp et al. [32], looked into a simple genetic algorithm for discrete optimization of 2D structures using a finite element analysis program developed by Zienchawicz. Thus research was directed towards expanding the scope of GA in the structural optimization arena. Moreover, it also appeared as if research efforts were also aimed at ways of improving efficiency and at the same time reducing the computational effort required for structural optimization. Neural networks were also used in structural design optimization to tackle the problem of lengthy computational time. To name just a few, Hajela and Berke [107] utilized back-propagation neural computing strategies for structural analysis. Adeli and Park [108-109] used hybrid counter propagation neural network technique for discrete optimization of large structures to improve the learning rate of the algorithm.

This chapter proposes an adaptive genetic algorithm based on dynamically changing probability of crossover, probability of mutation and population size, interfaced with an existing FEA package to increase applicability, and robustness. Discrete as well as continuous design variables can be handle by the proposed approach. Genuinely continuous data can easily be discretized whereas genuinely discrete data is sometimes impossible to make continuous [82].

Hence the current research effort extends prior research in an attempt to facilitate easy applications for structural optimization using the GA as optimizer. In other words, having a model of the structure created in a particular FEA package is all that is required to initiate the optimization process. Moreover, no programming effort or user intervention is required to conduct the structural optimization process. Weight (as opposed to cost) is specified as the objective function, and constraints are handled by the adaptive penalty method. Many examples found in the literature are solved by the proposed method and compared. Discussion and conclusions are centered on ease of applicability, and viability of the approach to optimize structural problems faced by engineers and scientist. This chapter concludes by addressing some of the possible problems of the proposed method and presents some insights into solving these problems in the near future.

6.2 Proposed FEA and Optimization Technique

The state of the art in this chapter is the fusion of a soft computing technique, namely, the genetic algorithm into an already existing finite element software package. This integration provides the ease in structural design and optimization, flexibility to conduct multidisplinary analysis and significant savings in terms of time and money. As will be argued in a latter section of this chapter, the presented hybrid approach though requiring long computing time for large problems, can provide a tradeoff for the lengthy time sent creating and integrating finite element codes with optimization techniques.

6.2.1 Propose Structural Technique

For structural design optimization it was important that the model be created in a software by means of a parameter list. The parameter list usually contained all the design variables that were required for building the model. This step is essential since it enabled smooth movement between the FEA software and the GA. For example, for size optimization of truss structures the cross sectional areas of the members were design variables and so appeared in the parameter list. In this the chapter cross sectional area can be either discrete or continuous. Pre-defined discrete cross sectional areas, their properties, constraints and design checks were taken from the American Institute of Steel Construction Allowable Stress Design (AISC-ASD) specifications manual [110]. On the other hand, the cross sectional properties such as moment of inertia and element length of continuous cross sectional areas of a member was calculated or if possible extracted from the FEA software package. Note the effects of discrete and continuous design variables are discussed in a latter section. Configuration optimization entailed setting the nodal coordinates as the design variables and subsequently including these variables in the parameter list. Note that this work does not include topology optimization but this can readily be incorporated and is the topic of future work.

Finally, if we were to design a simple cantilever beam in ANSYS, the length, height and width of the beam would be defined as scalar parameters for later use of model building. Note that it is not necessary to define all parameters in a list, but only the design variables that would be encoded in the GA binary string. Once the model was built using these design parameters, the objective function and constraints can be coded (specified) either within the FEA package or within the GA. The latter option is recommended since implementing the objective function and constraints within a FEA software depended on the user skills with that particular software. On the other hand, implementation of the constraints within the GA offered greater ease and flexibility to the user. The output of FEA process was the volume, element length, stresses, nodal deflection, and other problem specific constraint conditions. This entire process was carried out in batch mode and processing of the largest structure in this chapter using ANSYS took approximately 6 seconds. The output file from the FEA package was then used for post-processing and extraction of the necessary data required for calculating the weight and satisfying the required constraints.

6.2.2 Proposed Optimization Technique

Genetic algorithms are a class of soft computing technique that can be used to solve a wide range of NP hard combinatorial problems. The underlying principles of genetic algorithm were covered in Chapter 3. Today, many marvel at the GA concept and its pioneers who formulated the analogy between evolution and a tool effective for optimization. The GA proposed technique requires little user intervention. More importantly, a method of dealing with the problem faced by some, i.e. appropriate selection of population size, mutation rate and crossover rate was dealt with. Further details on creating and employing the GA for problem solving can be found in references 64, 66, 78-81, and 111 and references 65, 112-113.

6.3 Optimization Model

The objective function of the structural optimization model was the weight. Note that even though cost is a function of weight, cost as the objective function is more complicated since it entailed certain aspects such as maintenance, machinability, number of connections, material etc., that there is no exact relationship for and/or whose cost varied considerably from one time horizon to the next. Consequently, the model was formulated using weight as the objective function and is given as follow:
Generally we would like to find:

Find $[X = x_1, x_2 ... x_n]$ so as to minimize

Fitness function = objective function value + penalty value (1)

or

Fitness function = $F(X)$ + degree of violation of constraints (2)

Subject to: m structural constraint

where $X = x_1, x_2 ... x_n$ is a set of decision or design variables encoded in a binary string; n is the number of design variables; and m is the total number of constraints. In order to ensure only feasible designs, a penalty value was assigned to any design that violated any of the m constraints. Moreover, a penalty value was added to the objective function value thereby penalizing it and moving it away from the optimum by a distance proportional to the degree of violation of the constraint.
A more detailed optimization model is as follow:

$$Minimize\ W = \sum_{i=1}^{N} \rho_i L_i A_i \qquad (3)$$

for continuous design variables, or,

$$Minimize\ W = \sum_{i=1}^{N} \rho_i L_i A_i(id) \qquad (4)$$

for discrete design variables.

where i is the individual element . ρ_i, L_i and A_i are the material density, element length and element area respectively. id is a random discrete identification number within the binary design string. $A_i(id)$ is the area extracted from a database of AISC sections at a particular identification number. N is the total number of elements. The above functions were restraint by the following conditions, some of which are adapted from Ref. 94:

(1) Side constraints that limits areas

$$A_i(id)^L \le A_i(id) \le A_i(id)^U \qquad (5)$$

or

$$A_i^L \le A_i \le A_i^U \qquad (6)$$

where A_i^U is the maximum area limit and A_i^L is the lower area limit. Similar definition applies for discrete area members where an integer identification number identifies the area.

(2) Area linking

$$A_d = A_i \qquad (7)$$

where A_d and A_i are the dependent and independent areas respectively.

(3) Side constraints that limit nodal coordinates

$$(X_j, Y_j, Z_j)^L \le X_j, Y_j, Z_j \le (X_j, Y_j, Z_j)^U \qquad (8)$$

where $(X_j, Y_j, Z_j)^U$ is the upper limit and $(X_j, Y_j, Z_j)^L$ is the lower limit on the x, y and z nodal coordinates.

(4) Nodal coordinates linking

$$(X_K, Y_k, Z_k) = a_k + b_k(X_i, Y_i, Z_i) \tag{9}$$

where a_k and b_k are constants; X_k, Y_k, Z_K are dependent coordinate variables; and X_i, Y_i, Z_i are the independent coordinate variables.

(5) Fabrication restriction

Availability of a cross sectional area in the database predefined by AISC.

(6) Element stresses

$$\sigma_i^- \leq \sigma_{ij} \leq \sigma_i^+ \tag{10}$$

where σ_i^+ is the maximum tensile stress and σ_i^- is the maximum compressive stress in an element i when subjected to loading condition j.

(7) Nodal Deflection

$$\delta_k^- \leq \delta_{kj} \leq \delta_k^+ \tag{11}$$

where δ_k^+ and δ_k^- is the maximum upward and maximum downward deflection respectively, at node k when subjected to loading condition j.

(8) Euler Buckling

$$\sigma_{ij} < \sigma_{ci} \tag{12}$$

where σ_{ci} is the critical stress for element i.

The critical stress is given as:

$$\sigma_{ci} = -\frac{\pi^2 E_i}{\left(K_i L_i / r_i\right)^2} \tag{13}$$

where E_i is Young's modulus of elasticity; K_i is boundary condition dependent; r_i is the radius of gyration, and L_i is the element length for each member i in the structure. Square cross sections were assumed to simplify the buckling analysis. σ_{ij} is the stress of member i when subjected to loading condition j.

(9) Natural Frequency limit

$$NF_j < NF \qquad (14)$$

where NF is the natural frequency limit and NF_j is the natural frequency of the structure when subject to loading condition j.

Note that serviceability and fatigue constraints are more complicated and was ignored in this study, but its inclusion is also the topic of future work.

6.4 Techniques Interfaced

Finite element analysis (FEA) is a numerical method that is used to solve complex engineering problems. FEA was - and can be - applied to fields beyond that of structural analysis. In addition, FEA has become a standard method of structural analysis. Today much software is developed and is available to facilitate an easy finite element analysis for problems faced in many fields. The software used to interface with the binary GA created in MATLAB [73] is ANSYS/ professional FEA program from Swanson Analysis Systems Inc. [74]. ANSYS architecture provided fast and easy integrated pre and post processing. In addition, ANSYS also proved to be quite suited for iterative analysis with the GA. More importantly, ANSYS allowed for easy integration, facilitating quick batch processing, which saved significant amount of time in the analysis of the solution. In order to reduce the computational effort the GA based FEA required smooth integration between the GA and the FEA software. Batch processing was facilitated by ANSYS log file (based on ASCII text input) and written in ANSYS parametric design language (APDL); this is shown in Figure 6.4.1 as the structural analysis input template. The structural analysis input template contains instructions in APDL code to preprocess, solve, and post process the structure under study. These instructions are coded in MATLAB, and sent at each iteration into ANSYS to process the population of structural design generated by the GA. Using the results analysis file - which contains ANSYS output parameters such as stresses - the constraints and convergence criteria are check. Dynamic data exchange (DDE) was used to transfer output data from ANSYS to MATLAB in order to perform these checks. If the convergence criteria are satisfied, the program stops and the best design is output to the screen.

The architecture of the overall GA based FEA approach for structural optimization is shown in Figure 6.4.1.

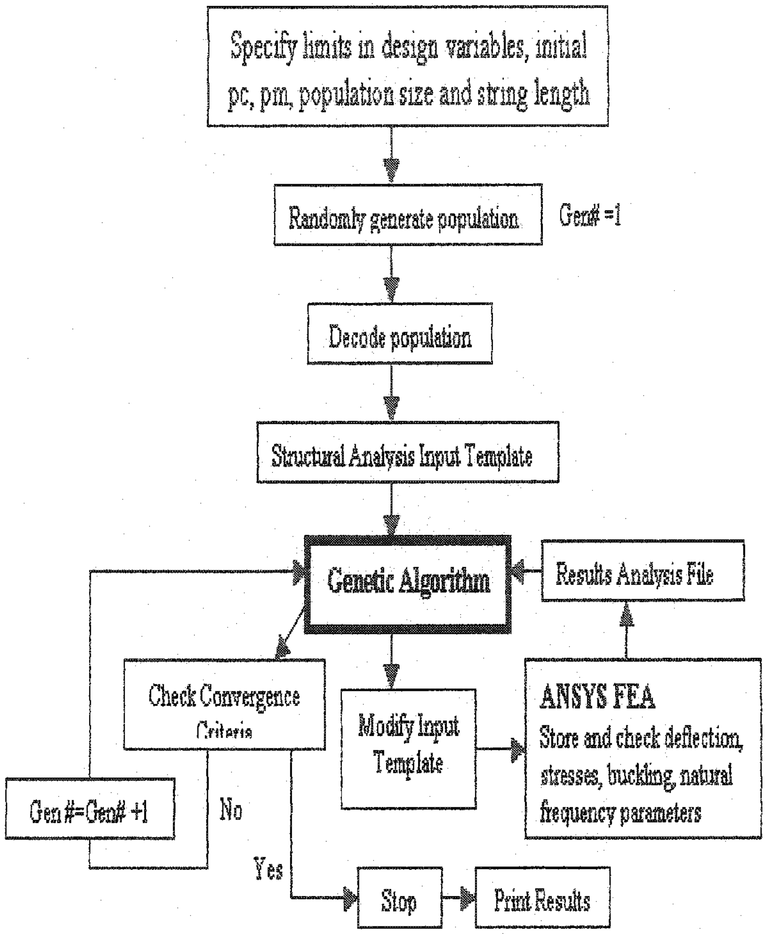

Specify limits in design variables, initial pc, pm, population size and string length
Randomly generate population
Gen# =1
Decode population
Structural Analysis Input Template
Genetic Algorithm
Results Analysis File
Check Convergence Criteria
Modify Input Template
ANSYS FEA
Store and check deflection, stresses, buckling, natural frequency parameters
Gen#=Gen# +1
No
Yes
Stop
Print Results

Figure 6.4.1: Design methodology and technique interface.

6.5 Illustrative Examples

The first example presented is a 3 bar truss, and it was used to draw comparisons between the optimization methods native to ANSYS i.e. the subproblem approximation method and the first order method, and the proposed method using GA as an optimizer. The subproblem approximation method is an advanced zero order method that does not require derivative information and uses a penalty method to reject infeasible solutions. The first order method on the other hand, required gradient information but also unconstraint the problem by adding a penalty function value to the objective function. The gradient calculation used with the first order method, typically employed steepest descent or conjugate direction method in order to find an appropriate search direction.

Other examples were solved, presented and compared with results found in the literature. Moreover, the power and limitation of the proposed method is also exposed via the examples presented. The GA operators and properties used are identical for all problems. An initial population size of 200 is used, with an initial probability of crossover of 0.6; initial probability of mutation taken as 0.01 and "b" as one fourth. The generation number is used as the ultimate stopping criteria and was set at 100. The structure initial geometry, characteristic properties, loading conditions and constraint values are given with each example.

The examples studied were variations of the 3 bar truss, the 10 bar truss, the 18 bar planar truss, 25 bar space truss subjected to two load conditions, a 72 bar transmission power, and a 47 bar planar tower subjected to three load conditions. Comparison of the results with the literature is drawn and discussion about the methodology is highlighted.

6.5.1 3-bar Truss

The initial geometry of the 3 bar truss is shown in Figure 6.5.1, and it was subjected to a single loading condition. This problem was adapted from Ref. 74 where the cross sectional area, A of each member, and base dimension, B, are the design variables. The constraint is axial stress, which should be less than 400psi. In addition, there are two side constraints for area and nodal coordinates. The applied loads, material constants and constraint value are given in Table 6.5.1. A comparison of the results from the proposed technique with the methods native to ANSYS is presented in Table 6.5.2. The initial and optimized geometry of the 3 bar truss structure is shown in Figure 6.5.1 and Figure 6.5.2 respectively.

Table 6.5.1. Loading condition for the 3 bar truss problem shown in Figure 6.5.1.

Node	Fx(lbs)	Fy(lbs.)	Fz(lbs)
4	200000	-200000	0

Constants

Young's Modulus	=2.1E6 psi
Density	=2.85E-4 lb/in3
Allowable stress	=400psi

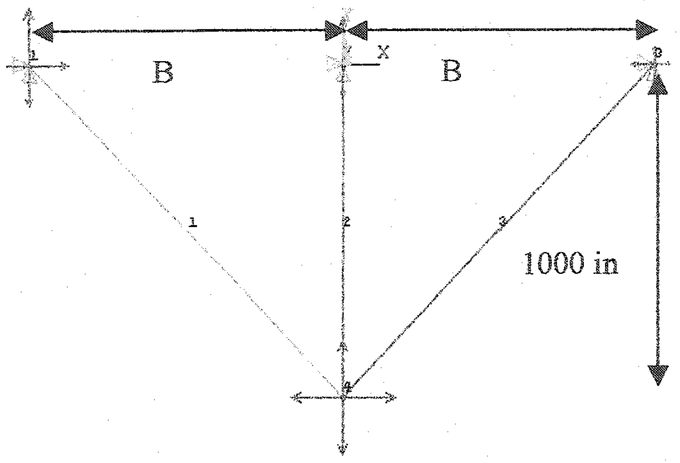


Figure 6.5.1: 3-bar truss, initial geometry

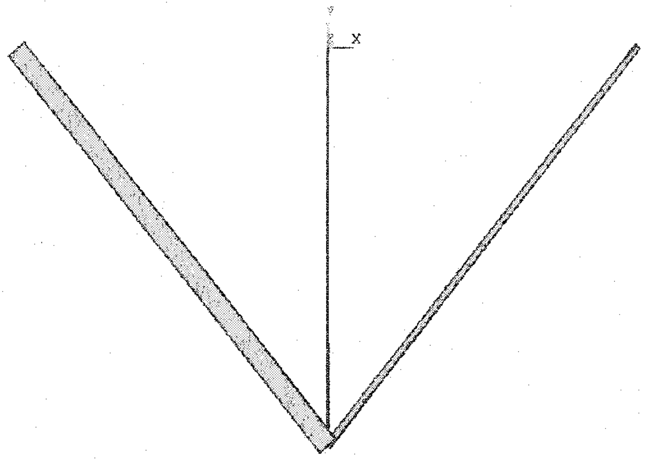


Figure 6.5.2: 3-bar truss, Optimized geometry

Table 6.5.2. Comparison of results with optimization techniques native to ANSYS

Method / Member	First order Method	Subproblem Approximation Method	GA-FEA
1	685.58	794.84	707.33
2	6.29	3.05	1.00
3	1.00	237.04	2.17
Dimension (B)	1001.20	590.019	993.89
Weight (lbs)	301.23	343.96	285.38

From the results it can be gleaned that the mathematical optimization tools native to ANSYS were not too effective in finding the optimum size and geometry of this structure. This ineffectiveness became more apparent as the problem got larger or the number of design variables increased. These mathematical tools were unable to provide better results than the GA since they were unable to handle a large search domain and since their search procedures were deterministic in nature. This example was undertaken simply to compare results of the optimization method native to ANSYS and the proposed method, and consequently illustrate the need for new and more effective optimization techniques in finite element software.

6.5.2 10-bar Cantilever Truss

The 10 bar cantilever truss problem, due to its simple configuration, have been used as a benchmark to verify the efficiency of diverse optimization methods. The initial geometry of the structure is shown in Figure 6.5.3, and the loading conditions, material properties and constraints are given in Table 6.5.3. This problem was adapted from Ref. 32. The results found are presented in Table 6.5.4 alongside the results in the literature.

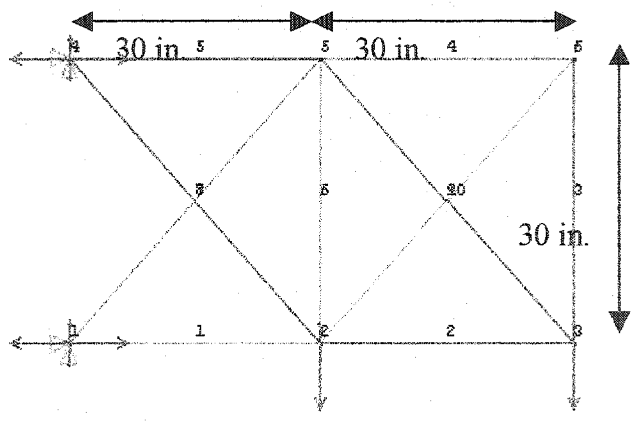


Figure 6.5.3: 10-bar truss, initial geometry

Table 6.5.3. Loading and design condition for 10-bar truss problem shown in Figure 6.5.3.

Node	Fx(lbs)	Fy(lbs.)	Fz(lbs)
2	0	-100000	0
3	0	-100000	0

Constants

Young's Modulus	=1E7 psi
Density	=0.1 lb/in3
Areas range	=0.1-35 in2
Allowable stress	=25 Ksi
Allowable nodal deflection in x and y direction	=2 inches.

As mentioned before, the 10 bar truss problem is a benchmark problem for structural optimization. This problem was solved by numerous mathematical as well as heuristic methods. Some of these methods and their results are illustrated in Table 6.5.4. Note that apart from this research effort, the other proposed heuristic approach also excelled when compared with mathematical and optimality criteria techniques.

The 10 bar truss was solved by numerous mathematical as well as heuristic methods. Some of these methods and their results are illustrated in Table 6.5.4. With respect to the other heuristic methods presented, the adaptive nature of the GA in this study provided comparable results. Note however that neural networks applying back-propagating strategies [107] provided better results than the present study.

Table 6.5.4. Comparison of design variables and objective function values for 10-bar truss show above

Researcher	Gellatly [101]	Venkayya [102]	Schmit [95]	Qian [98]	Belegundu [97]	Camp [32]	Hajela & Berke [107]	Current Study
Method / Member	Optimality Criteria	Two Subspace Approach	MP	LP	MP	Genetic Algorithm with Zienkiewicz Program	Neural Networks	GA-FEA
1	22.21	23.416	23.76	23.545	25.70	24.07	30.774	22.48
2	15.6	14.904	14.56	14.96	0.1	13.96	0.112	14.94
3	0.24	0.53	0.53	0.297	25.11	0.56	17.40	0.42
4	0.1	0.128	0.1	0.1	19.39	0.1	11.425	0.14
5	31.35	30.416	30.67	30.902	0.10	28.92	0.108	30.75
6	0.1	0.101	0.1	0.1	0.10	0.1	0.487	0.12
7	22.06	21.084	21.07	21.275	15.40	21.95	5.593	21.10
8	8.35	8.578	8.578	7.611	20.32	7.69	22.953	7.26
9	0.14	0.101	0.1	0.1	20.74	0.1	20.886	0.13
10	20.03	21.077	20.96	21.156	1.14	22.09	0.100	20.74
Weight (lbs.)	5112	5084.9	5076.85	5069.4	5472.00	5076.35	4692.49	**5054.98**

-MP- Mathematical Programming
-LP- Linear Programming

6.5.3 18-bar Cantilever Truss

The 18 bar truss problem was adapted from Ref. 94 and was also tested in Ref. 96 and 114. The initial geometry of the structure is shown in Figure 6.5.4. The truss was subjected to a single load condition. Two different constraint scenarios were studied. First, the constraints were in stress, area and geometry. For this case the element areas are linked as follow: A1=A4=A8=A12=A16; A2=A6=A10=A14=A18; A3=A7=A11=A15 and A5=A9=A13=A17. In addition, the geometry of the truss was allowed to vary in both x and y directions at nodes 3, 5, 7 and 9. Consequently, there were 4 independent area design variables and 8 geometry design variables. The loading condition, material properties and constraints are given in Table 6.5.5. Figure 6.5.5 shows the optimized geometry, while Table 6.5.6 compares the results obtained.

Secondly, the 18 bar truss was resolved with exactly the same condition as above, but with additional constraints in buckling and fabrication. The pre-defined structural members and their properties were taken from AISC manual [110].

The results are compared with the literature and are given in Table 6.5.7. The convergence history for this run is shown in Figure 6.5.6. Figure 6.5.7 illustrates the optimized design.

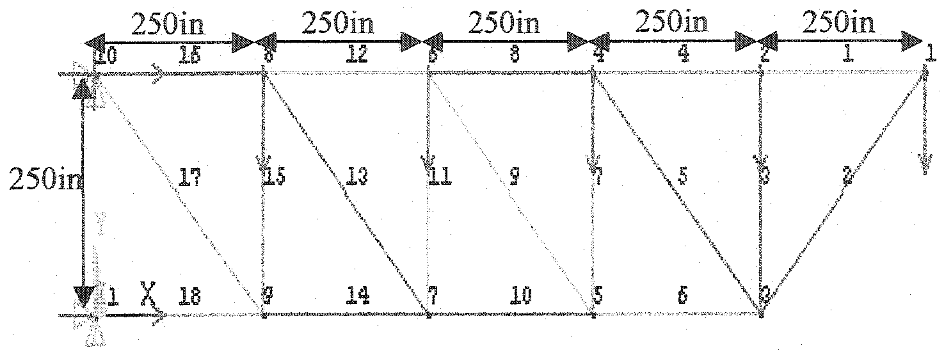


Figure 6.5.4: 18-bar truss, initial geometry

Table 6.5.5. Loading condition for the 18 bar truss problem shown in Figure 6.5.4.

Node	Fx(lbs)	Fy(lbs.)	Fz(lbs)
1	0	-20000	0
2	0	-20000	0
4	0	-20000	0
6	0	-20000	0
8	0	-20000	0

Constants

Young's Modulus	=1E7 psi
Density	=0.1 lb/in3
Buckling Coefficient	=4.0
Allowable stress	=20 Kpsi

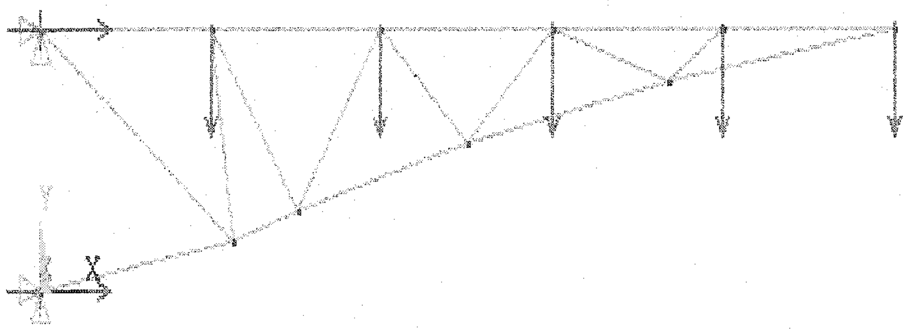


Figure 6.5.5: 18- bar truss, optimized structure when subject to **stress constraint**

Table 6.5.6. Comparison of design variables for 18-bar truss subject **to stress constraint**

Researcher	Felix [114]	Hansen [94]	Current Study
Method / Design Variables	MP	MP	GA-FEA
A1	11.05	10.71	11.45
A2	15.07	15.19	15.2
A3	4.54	1.94	1.91
A5	5.33	5.19	4.384
X3	991.2	881.4	920.8
Y3	19.7	178.8	200.2
X5	745.9	628.9	625.2
Y5	15.1	124.9	140.6
X7	494.6	390.5	378.76
Y7	34.5	66.8	76.64
X9	249.5	313.2	282.2
Y9	23.6	45.0	45.7
Weight (lbs)	4524.7	3906.8	3899.1

Table 6.5.7. Comparison of design variables for the 18-bar planar truss subject to **stress, fabrication and buckling constraint**

Researcher	Hansen [94]	Imai [96]	Felix [114]	Current Study
Method / Design Variables	MP	ALM	MP	GA-FEA
A1	12.76	11.24	11.34	13.5
A2	17.77	15.68	19.28	15.4
A3	5.55	7.93	10.97	4.68
A5	3.26	6.49	5.30	2.65
X3	881.4	891.1	994.6	848.613
Y3	178.8	143.6	162.3	191.48
X5	628.9	608.2	747.4	690.512
Y5	124.9	105.4	102.9	161.703
X7	390.5	381.7	482.9	418.875
Y7	66.8	57.1	33.0	110.827
X9	313.2	181.0	221.7	235.571
Y9	45.0	-3.2	17.1	53.584
Weight (lbs)	4505.0	4667.9	4524.7	4171.95

-ALM- Augmented Lagrange multiplier

This problem clearly demonstrated the power of the GA as a search and optimization tool. Note that the inclusion of buckling into the design led to a heavier structure. Moreover buckling response of the 18 bar truss introduced a greater sense of practicality into the design.

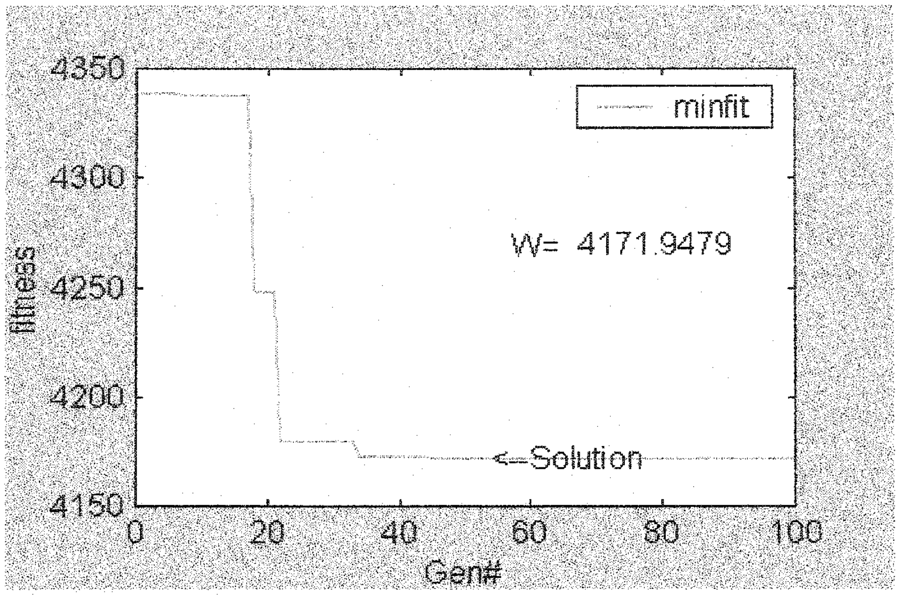


Figure 6.5.6: 18-bar truss, convergence history when subjected to **stress, fabrication and buckling constraint**

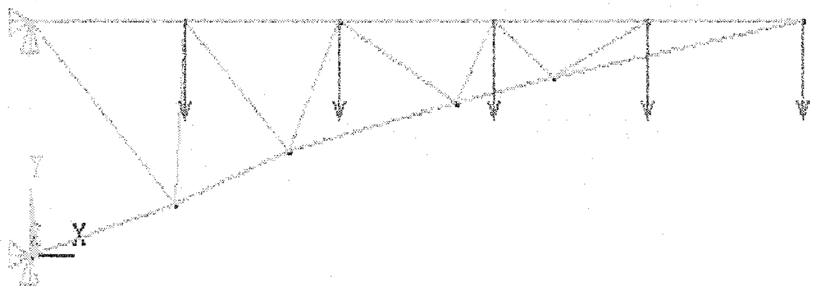


Figure 6.5.7: 18 bar truss, optimized structure when subjected to **stress, fabrication and buckling constraint**

Note that even though the inclusion of fabrication constraint typically leads to a heavier structure, the proposed technique was capable of finding a weight much lower than previous work. Furthermore, the results indicated that the GA was clever in selecting AISC sections and a geometry that yielded a low weight at the same time satisfied all imposed constraints.

In a later section of this chapter, the inclusion of vibration into the design of this structure is conducted.

6.5.4 25-bar Space Truss

Three different cases of the 25 bar space truss were studied. The structure is adapted from Ref. 94. The initial geometry of the structure is shown in Figure 6.5.8.

6.5.4.1 Case 1

The structure was analyzed using a single load condition -as given in Table 6.5.9 - subjected to area, stress and displacements constraints. The members' connectivity table is given in Table 6.5.8. The material properties and constraint values are also given in Table 6.5.9. Comparison of the results with the literature is given in Table 6.5.10.

6.5.4.2 Case 2

The structure was designed to support two independent load conditions that are given in Table 6.5.11. The structure was subjected to area, geometry, stress and buckling constraints. Member linking is given in Table 6.5.8. The coordinate variables that were linked to preserve symmetry are X4, Y4, Z4, X8 and Y8. The optimized geometry is given in Figure 6.5.9 while Table 6.5.12 compares the results obtained with those found in the literature.

6.5.4.3 Case 3

Natural frequency constraints i.e. vibration is studied separately in order to demonstrate the ease and flexibility available in employing a FEA software aided by the GA for optimization. The vibration analysis utilized the Block Lanczos method of mode extraction, which in turn used the sparse matrix solver to conduct the analysis. These methods are native to ANSYS and allowed for quick and easy vibration analysis. The Block Lanczos method also automatically uses lumped mass approximation. The lumped mass approximation method in ANSYS is recommended since it was found to yield better results and incurred lower run time when compared to the other methods available in the software.
Pre-stress effects, which considered the stress state of the structure and influenced the stiffness matrix was also incorporated. The procedures entailed in the vibration analysis

were model building, load application, solution processing, mode expansion and extraction and finally post processing. For further details on this method please consult ANSYS help files using vibration as keyword.

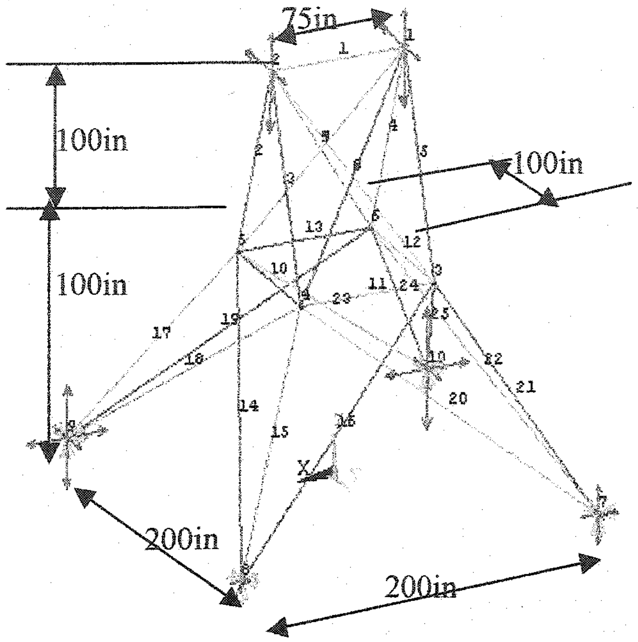


Figure 6.5.8: 25-bar space truss, initial geometry

Table 6.5.8. Member grouping for the 25-bar space truss shown in Figure 6.5.8

Group Number	Members
1	1-2
2	1-4,2-3,1-5,2-6
3	2-5,2-4,1-3,2-6
4	3-6,4-5
5	3-4,5-6
6	3-10,6-7,4-9,5-8
7	3-8,4-7,6-9,5-10
8	3-7,4-8,5-9,6-10

Table 6.5.9. Case 1 and Case 3 loading condition for the 25-bar space truss problem shown in Figure 6.5.8.

Node	Fx(lbs)	Fy(lbs.)	Fz(lbs)
1	1000	-10000	-10000
2	0	-10000	-10000
3	500	0	0
6	600	0	0
			Constants
Young's Modulus		=1E7 psi	
Density		=0.1 lb/in3	
Allowable stress		=± 40 Kpsi	
Allowable nodal deflection in x and Y direction at nodes 1 and 2		=± 0.35in.	

Table 6.5.10. Comparison of design variables for the 25-bar space truss subject to **area, stress and deflection constraints**

Researcher	**Rajeev [29]**	**Coello [45]**	**Adeli & Park [107-108]**	**Current Study**
Method / Area [in2] Coordinates x-y[in]	**Simple Discrete GA**	**Discrete GA**	**Neural Networks**	**GA - FEA**
A1	**0.100**	**N/A**	**0.6**	**0.1006**
A2	**1.800**	**N/A**	**1.4**	**0.1029**
A6	**2.300**	**N/A**	**2.8**	**3.4103**
A10	**0.200**	**N/A**	**0.5**	**0.1086**
A12	**0.010**	**N/A**	**0.6**	**1.8284**
A14	**0.800**	**N/A**	**0.5**	**0.74813**
A18	**1.800**	**N/A**	**1.5**	**0.1**
A22	**3.000**	**N/A**	**3.0**	**3.7166**
Weight (lbs.)	**546.01**	**493.94**	**543.95**	**459.82**

Table 6.5.11. Case 2 loading condition for the 25-bar space truss problem shown in Figure 6.5.8.

Node	Fx(lbs)	Fy(lbs.)	Fz(lbs)
		Load Condition 1	
1	0	-20000	-5000
2	0	-20000	-5000
3	0	0	0
6	0	0	0
		Load Condition 2	
1	1000	-20000	5000
2	0	-20000	-5000
3	500 0	0	
6	500 0	0	
		Constants	
Young's Modulus			=1E7 psi
Density			=0.1 lb/in3
Allowable stress			=± 40 Kpsi

Table 6.5.12. Comparison of design variables for the 25 bar space truss subject to **area, geometry, stress and buckling constraints**

Researcher	**Vanderplaats [115]**	**Hansen [94]**	**Felix [114]**	**Current Study**
Method / **Area [in^2] Cordinates x-y [in]**	**Two-Subspace approach**	**MP**	**MP**	**GA –FEA**
A1	**0.032**	**0.01**	**0.013**	**0.138**
A2	**0.565**	**0.487**	**0.414**	**0.161**
A6	**0.811**	**0.836**	**0.842**	**0.345**
A10	**0.028**	**0.021**	**0.033**	**0.074**
A12	**0.047**	**0.123**	**0.101**	**0.161**
A14	**0.097**	**0.084**	**0.121**	**0.01**
A18	**0.749**	**0.698**	**0.739**	**0.345**
A22	**0.551**	**0.548**	**0.554**	**0.161**
X4	**12.9**	**23.7**	**21.5**	**15.116**
Y4	**48.2**	**49.3**	**48.3**	**87.524**
Z4	**97.4**	**97.7**	**100.3**	**100.9**
X8	**37.1**	**27.5**	**22.1**	**51.516**
Y8	**94.1**	**96.4**	**96.4**	**164.72**
Weight (lbs.)	**133.5**	**128.3**	**128.5**	**61.75**

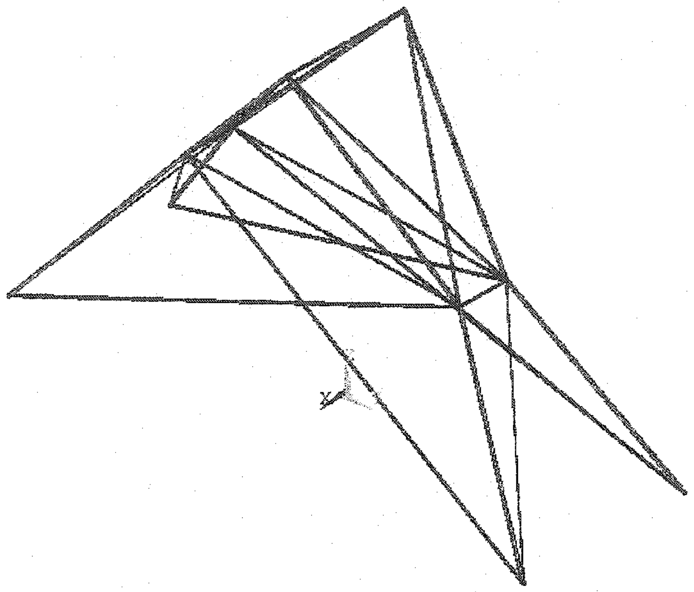


Figure 6.5.9: 25-bar space truss, optimized structure subject to **area, geometry, stress and buckling constraints**

The results of the 25 bar truss demonstrated the robustness of the technique to find feasible optimum designs. The structural design that incorporated only area design variables led to a much heavier structure than that which included area and geometry design variables, as well as added constraints. In other words, the more design variables and constraints added to the design to fully capture the problem, the more practical the output design became. On the other hand, increasing the number of design variables and constraint also led to increase computational effort.

When subjected to stress and deflection constraint the proposed approach was capable of finding better designs than other GA methods and even neural networks. Perhaps, the neural networks by [108-109] yielded poor results because a discrete optimization technique was used. Nonetheless, the results obtained in this study were primarily due to the adaptive nature of the GA as mentioned before. The results also indicated that the adaptive GA and its operators provided a quick and broader search when compared to other techniques. For the case where the truss was subjected to stress and buckling constraints, the subspace approach used by Vanderplaats, and the approximate structural technique based on Taylor expansions of various structural responses by Felix, as well as the approximate structural method based on Taylor series expansion of the member forces by Hansen, all proved to be incapable of the finding a lighter truss than the proposed method. These mathematical programming techniques were unable to provide any reasonable results mainly because of the high non-convexity of the search space and the high number of design variables involved [45].

The results with the inclusion of natural frequency constraint (case 3) are given in Table 6.5.13. The optimize geometry found by the proposed method is given in Figure 6.5.10.

Table 6.5.13. Design Variables for the 25-bar space truss subjected **to area, geometry, stress, and buckling and natural frequency constraints**

Area [in2] Coordinates x-y [in]	**GA Results**
A1	**0.074**
A2	**0.345**
A6	**0.345**
A10	**0.074**
A12	**0.100**
A14	**0.074**
A18	**0.222**
A22	**0.100**
X4	**35.473**
Y4	**60.861**
Z4	**131.86**
X8	**68.165**
Y8	**55.622**
Weight (lbs.)	**58.23**

The inclusion of vibration analysis into the study provided some interesting results. From the results one can conclude that the inclusion of vibration led to a lighter but more practical truss. Note also that the base of the structure is much wider at all fixed ends compared to the previous two cases studied, implying a greater distribution of the center of gravity of the structure.

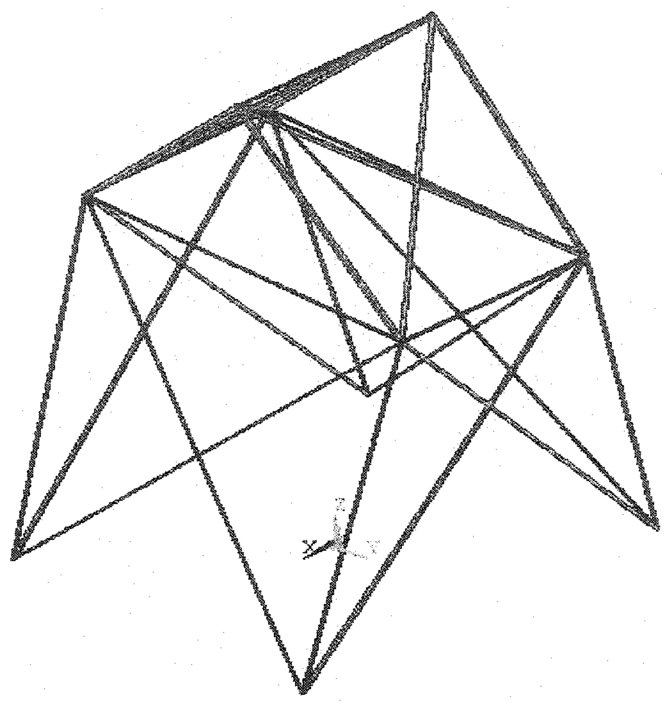


Figure 6.5.10: 25-bar space truss, optimized geometry subject to **area, geometry, stress, buckling and natural frequency constraints**

6.5.5 72-bar Transmission Tower

A 72 bar transmission tower was design to support two independent load conditions that are given in Table 6.5.14. The loading condition, material properties and constraints are also given in Table 6.5.14. The structure was subjected to area, stress and deflection constraints. The initial geometry of the design is shown in Figure 6.5.11. This structure was adapted from Ref. 83. The areas are linked as follows: A1-A4; A5-A12; A13-A16; A17-A18; A19-A22; A23-A30; A31-A34; A35-A36; A37-A40; A41-A48; A49-A52; A55-A58; A59-A66; A67-A70 and A71-A72.

Hence in total there were 16 design variables for this problem.

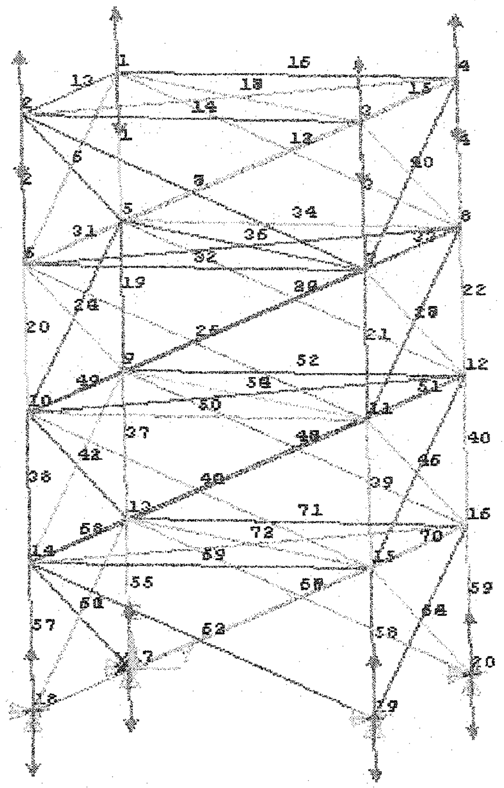


Figure 6.5.11: 72-bar transmission tower, initial geometry

The results found are presented in Table 6.5.15 alongside the results in the literature.

Since this problem only considers size optimization, since the design variables are highly linked, and since there are few constraints, this problem should be much simpler to solve than the 18 bar truss and the 25 bar truss previously studied.

However, from Table 6.5.15 it can be gleaned that the results obtained are not superior to all the methods found in the literature. This maybe partly due to the large population used to solve such a small problem. This means that the GA spends more time exploring the search space than exploiting an area in the search space where a near optimum or optimum may be located. Perhaps the results found is near the global optimum and finding this point takes a long time. This problem is one of the inherent disadvantages of genetic algorithms.

Table 6.5.14. Loading Conditions for the 72-bar Transmission Tower shown in Figure 6.5.11

Node	Fx(lbs)	Fy(lbs.)	Fz(lbs)
		Load Condition 1	
1	5000	5000	-5000
		Load Condition 2	
1	0	0	-5000
2	0	0	-5000
3	0	0	-5000
4	0	0	-5000
		Constants	
Young's Modulus			=1E7 psi
Density			=0.1 lb/in3
Allowable tensile stress			=25 Kpsi
Allowable nodal deflection in x and y direction			=± 0.25in.

Table 6.5.15. Comparison of the design variables and objective function value for 72-bar transmission tower

Researcher	**Gellatly [101]**	**Venkayya [102]**	**Schmit [116]**	**Renwei [117]**	**Xicheng [118]**	**Erbatur [83]**	**Current Study**
Method / **Member [in^2]**	**Optimality Criteria**	**Two Subspace Approach**	**MP**	**MP**	**Parallel Algorithm**	**GA Based Optimum Structural Design**	**GA-FEA**
A1	0.1492	0.161	0.1585	0.1641	0.157	0.161	0.100
A5	0.7733	0.557	0.5936	0.5552	0.537	0.544	0.590
A13	0.4534	0.377	0.3414	0.4187	0.411	0.379	0.345
A17	0.3417	0.506	0.6076	0.5758	0.571	0.521	0.283
A19	0.5521	0.611	0.2642	0.5327	0.509	0.535	0.406
A23	0.6084	0.532	0.5480	0.5256	0.522	0.535	0.467
A31	0.1000	0.100	0.1000	0.1000	0.100	0.103	0.100
A35	0.1000	0.100	0.1509	0.1000	0.100	0.111	0.100
A37	1.0235	1.246	1.1067	1.2893	1.286	1.301	1.571
A41	0.5421	0.524	0.5793	0.5201	0.516	0.498	0.651
A49	0.1000	0.100	0.1000	0.1000	0.100	0.110	0.100
A53	0.1000	0.100	0.1000	0.1000	0.100	0.103	0.100
A55	1.464	1.818	2.0784	1.9173	1.905	1.910	1.693
A59	0.5207	0.524	0.5034	0.5207	0.518	0.525	0.467
A67	0.1000	0.100	0.1000	0.1000	0.100	0.122	0.100
A71	0.1000	0.100	0.1000	0.1000	0.100	0.103	0.100
Weight (lbs.)	395.97	381.2	388.63	379.66	380.84	383.12	**384.645**

6.5.6 47-bar Planar Tower

The 47 bar planar tower shown in Figure 6.5.12 was designed to support three different loading conditions given in Table 6.5.16. The structure was adapted from Ref. 94. The material constants and constraint values are also given in Table 6.5.16. The structure is subjected to constraints in geometry, area, stress and buckling. The maximum tensile stress is restricted to 20000psi,while the maximum compressive stress is restricted to 15000psi. The areas are linked as followed A1=A3, A2=A4, A6=A5, A9=A8, A11=A12, A13=A14, A16=A15, A17=A18, A19=A20, A21=A22, A23=A24, A25=A26, A29=A30 A32=A31, A34=A35, A37=A36, A39=A40, A42=A41, A44=A45 and A47=A46. The geometry is also linked with X2, X4, Y4, X6, Y6, X8, Y8, X10, Y10, X12, Y12, X14, Y14, X20, Y20, X21 and Y21 as independent variables. This resulted in a total of 44 design variables subjected to stress and buckling constraints, where the intent is to minimize the weight. The optimized geometry is given in Figure 6.5.13 and Table 6.5.17 and Table 6.5.18 compares the results of the design with the results found in the literature.

This problem was undertaken to test the proposed technique viability to solve large problems. This problem entailed fewer linked variables, thus leading to a higher number of design variables. Fewer members linking were used to render the problem harder to optimize. Results indicate that the proposed approach found better designs than the approach by Vanderplaats et al., and Felix. Vanderplaats et al. used a stress design method to solve the sizing problem and a constrained steepest descent method to solve the configuration problem. On the other hand, Felix used an approximate structural technique based on Taylor series expansion of various structural responses.

Table 6.5.16. Loading conditions for the 47-bar planar truss tower shown in Figure 6.5.12.

Node	Fx(lbs)	Fy(lbs.)	Fz(lbs)
		Load Condition 1	
17	6000	-14000	0
22	6000	-14000	0
		Load Condition 2	
17	6000	-14000	0
22	0	0	0
		Load Condition 3	
17	0	0	0
22	6000	-14000	0
		Constants	
Young's Modulus			=3E7 psi
Density			=0.3 lb/in3
Allowable tensile stress			=20 Ksi
Allowable Compressive stress			=15 Ksi

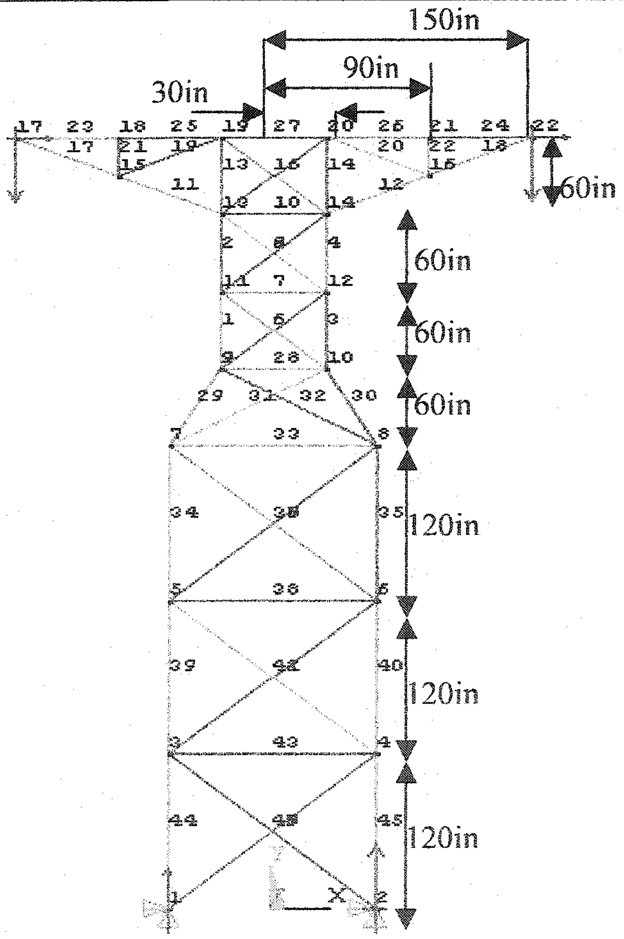


Figure 6.5.12: 47-bar planar tower, initial geometry

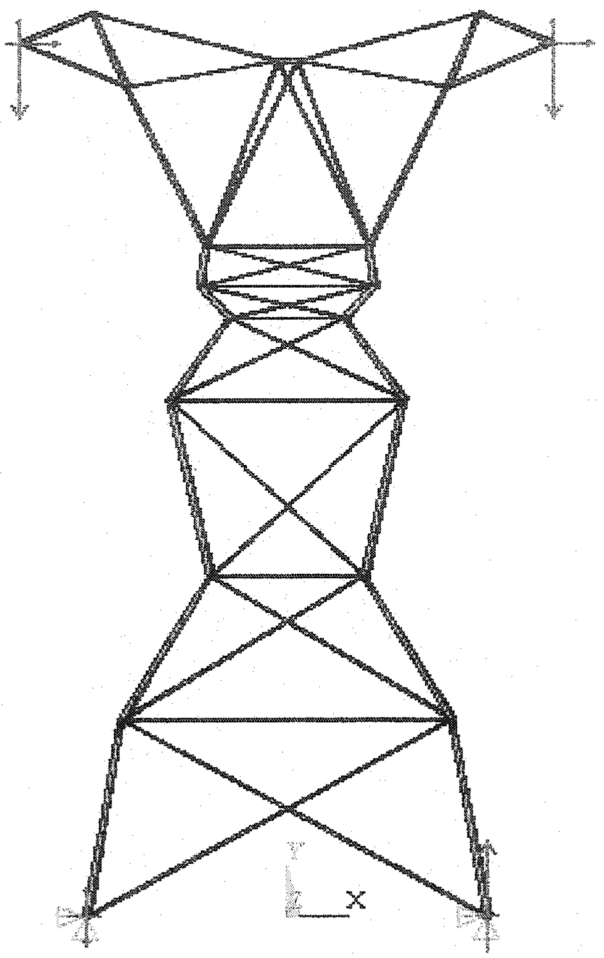


Figure 6.5.13: 47-bar planar tower, optimized geometry

Table 6.5.17. Comparison of area design variables for the 47-bar planar tower subject to **area, geometry, stress and buckling constraints**

Researcher	Vanderplaats [119]	Hansen [94]	Felix [114]	Current Study
Method / Area [in2] Coordinates x-y [in]	MP	MP	MP	GA- FEA
A3	2.80	2.42	2.73	2.86
A4	2.56	2.35	2.47	2.43
A5	0.77	0.82	0.73	0.23
A7	10-6	0.10	0.21	0.21
A8	0.67	0.86	0.94	0.10
A10	1.80	1.15	1.08	0.71
A12	1.94	1.77	1.69	0.93
A14	0.65	0.67	0.69	0.10
A15	1.06	0.86	1.06	0.10
A18	1.695	1.24	1.41	0.71
A20	0.16	0.33	0.26	0.1
A22	0.61	1.22	0.81	1.14
A24	1.31	0.93	1.06	0.90
A26	1.32	0.86	1.05	0.26
A27	1.05	0.69	0.82	0.87
A28	0.52	0.15	0.30	0.21
A30	2.94	2.46	2.77	1.50
A31	0.73	0.90	0.66	0.30
A33	10-6	0.10	0.21	0.39
A35	3.16	2.74	2.90	2.36
A36	0.97	0.92	0.27	0.10
A38	0.18	0.10	1.41	0.10
A40	3.47	2.94	3.43	1.50
A41	1.02	1.13	0.99	0.50
A43	10-6	0.10	0.17	0.10
A45	3.71	3.12	3.65	2.00
A46	0.93	1.10	1.01	0.50

Table 6.5.18. Comparison of geometry design variables for the 47-bar planar tower subject to **area, geometry, stress and buckling constraints**

Researcher	Vanderplaats [119]	Hansen [94]	Felix [114]	GA Results
X2	85.0	107.1	90.0	111.97
X4	72.4	91.2	90.0	92.83
Y4	121.3	122.8	123.4	135.43
X6	59.1	74.2	83.4	42.84
Y6	241.4	241.4	244.5	233.86
X8	49.5	65.5	70.5	65.51
Y8	356.8	324.6	355.1	355.12
X10	42.0	57.1	60.0	32.20
Y10	422.8	400.4	425.0	411.42
X12	45.6	49.3	58.2	48.74
Y12	480.5	472.3	478.0	433.31
X14	40.2	47.4	59.6	45.43
Y14	530.8	507.5	519.5	460.83
X20	25.5	3.9	15.0	5.49
Y20	596.5	586.5	607.6	587.8
X21	90.2	83.3	96.9	108.98
Y21	609.0	636.0	633.7	620.47
Weight lbs.	1879.0	1850.4	1904.0	**1009.07**

6.6 Discussion of Results and Conclusions

The structural analysis in this study entailed finding the nodal deflections, axial stresses, critical stresses, and natural frequencies all of which were calculated with the aid of a static analysis in ANSYS.

The proposed genetic algorithm based finite element analysis technique was capable of finding lighter structural designs when compared with mathematical, optimality criteria and other heuristic approaches. This is partly attributed to the adaptive feature of the GA as well as the disruptive random genetic operators employed. Via numerous examples two and three-dimensional structures were studied and designed under stress, deflection, buckling, fabrication and natural frequency constraints, as well as multiple loading conditions.

Comparison of the tables revealed that the GA performed satisfactory and subsequently provided better results in most cases than that found in the literature. All the results indicated that the above analysis were effective in retrieving reasonable designs with the GA as optimizer. Moreover, from the results obtained, all the analysis provided reasonable response to the structures studied. Hence the GA-based FEA techniques proposed is arguably an efficient method for optimizing any structure. More importantly, the proposed method is flexible and allowed easy integration of many different structural aspects. Hence once a structure had been modeled, optimizing it with the integrated open architecture method proposed became an easy task. Nonetheless, a shortcoming of the approach was the large computational effort required. The largest structural model presented in this chapter i.e. the 47 bar truss planar tower, took as much as 14 hours - on a Pentium III processor- to optimize, where convergence was achieved within 100 generations. This was primarily due to the large number of design variables involved for this particular structure. Optimization of a structural system comprising numerous design variables is a challenging problem due to its huge search space. In fact, it is not easy for any optimization algorithm to carry out an effective exploration without locating a local optimum. The dimension of the search space grows exponentially with the addition of design variables [83]. This concept is often referred to as "combinatorial explosion." Typically, research dealing with this problem has centered on multi-objective optimization as a way of reducing the large search space incurred when studying large problems.

Nonetheless, the results of the examples studied illustrated that the GA was quite successful in locating the region of the search space in which the global optimum is present. Further improvement in the results beyond this point however was slow. Thus one of the major inherent disadvantages of the GA i.e. slow convergence speed near global optimum was not alleviated in its entirety by the adaptive procedures employed in the GA. Perhaps, computational time can be reduced by utilizing multiple CPUs that can simultaneously process different segments of the population before reproduction takes place.

The use of the proposed technique to facilitate quick and easy structural optimization under various responses represents a step forward in structural optimization. Perhaps this work can be extended to write the GA in a language native to the FEA software. So for example, coding the GA in APDL language would remove the need for conversating and data exchange between the MATLAB coded GA and ANSYS, thus leading to a reduction in computing time. For now it appears that the advantage of the GA to process numerous structural designs was in itself a disadvantage when attempting to integrate it with a finite element software, since this process requires added computational effort. The proposed technique attempted to reduce the effort required by using a dynamically changing or adaptive population size. However, even with this in place processing time for the largest structure was still extensive. Using a faster computer can further reduce computing time.

Moreover, the use of a local search technique to aid the GA search procedure later on in the generation can help curb the computational effort entailed in this study. In other words, a hybrid GA can be employed to assist the search process in finding the absolute global optimum after it had converged on the area in which the absolute global optimum is located. Future research direction is also oriented towards using GA integrated artificial neural networks (ANN) to reduce the large computational effort required for structural optimization.

In addition, since GAs are not mathematical based they lack proof of optimality. In some ways this lack of proof is traded off for the ability to search a large search space. Moreover, it is the random nature of the GA that aids it in searching this vast search space.

In concluding, the approach proved to be beneficial, practical and feasible for structural design yielding efficient and weight effective structures, despite long computational time incurred for large structures. Nonetheless, the issue of rapidly optimizing a structure is a shortcoming of this study due to the inherent slow convergence speed of the GA near the global optimum. However, this should not be seen as a limitation in light of the advancements made at improving the processing speed of computers.

Chapter 7

Optimization of Two-component Armour Systems against Ballistic Impact using Genetic Algorithms

Shielding devices for personal protection to prevent impact damage by projectiles of various shapes and mass, traveling over a wide range of velocities is a major concern in both military and civilian application. Currently the trend is to reduce the weight of the armour. However, the demand for weight reduction of the armour must typically be accompanied by reduced cost and increased performance. This is commonly achieved today by searching for better materials or changing the armour geometry. Moreover, this scenario suggests the use of advanced materials and perhaps multiple layered armours. Today there is much research in the literature on impact analysis of single layer and multiple layer targets. Publications emanating from such activities typically utilize analytical modeling, numerical simulation, experimentation or any combination of the three methods, to investigate the interaction between projectile and target.

This study addresses the optimization of two-component armours using existing analytical models to predict the penetration mechanism of the armour. Various analytical models are studied, coded, and then integrated into the GA optimization technique. The aim of this study is to reduce the number of target configurations (armour designs), so as to mitigate the experimental and numerical simulations required. Attaining the latter will result in reducing the cost required for experimentation and the lengthy computing time required by numerical simulations. Comparison of the optimized results with results found in the literature is conducted when such data is available. Both the ballistic limit and the areal density are selected as the objective function for optimization depending on the approach taken in formulating and developing the analytical model employed. The ballistic limit velocity is a measure of performance of the armour; it is the velocity at which the projectile has a 50% chance of perforating the target when striking it at normal incidence. The areal density, on the other hand, is mass of the armour per meter squared. It is the product of material density and thickness of the plate. The areal density is indicative of the weight of the armour.

Results prove that genetic algorithms are quite useful for armour design and optimization. In addition, results also suggest that the fusion of existing analytical models with the genetic algorithm produces reasonable results requiring significantly less computing time and cost than that typically associated with experimental research. The results also indicate that the genetic algorithms have the capability to outperform other traditional methods used for armour design and optimization. Satisfactory qualitative and quantitative comparisons are obtained between published and optimized results.

7.1 Introduction

The subject of defeating projectiles or strikers impacting onto a target has received tremendous attention particularly in the military over the past three decades. Some publications can be found on optimization of two component armours using analytical models in which numerical or experimental studies are also conducted to draw comparisons and foster discussion. Recently the need for optimization of the armour has arisen due to the need to reduce the weight but at the same time decrease cost and increase performance. Wilkins et al. [34-37] demonstrated via extensive experimental work that ceramic performed well as an armour. They also discovered that the ceramic first blunts the projectile due to its high hardness and spreads the impact load onto the backing by forming a conoid. Ceramic armours backed by composite layers are becoming an area of intense research because of their outstanding performance against small and medium caliber projectiles when weight reduction is the main objective. The major role of the ceramic is to erode and rupture the projectile, while the composite functions to absorb the kinetic energy and prevent ceramic cracking.

To the author's knowledge Florence [33] appears to be the first to present a simple analytical model to analyze the impact onto ceramic/composite armour. Florence model – based on the ideas of Wilkins [34-37]- was used to estimate the ballistic limit. Studies conducted by Prior [38], Gage [39] and Rajagopalan [40] have found satisfactory agreement between experimental results and the theoretical predictions by Florence model. Hetherington [41] later revised Florence model to account for the energy absorbing capabilities of the armour. Woodward [42] later proposed a one-dimensional model for the perforation of a ceramic armour using a lumped mass approach, which takes into account the erosion of both projectile and target. Many of the ideas in Woodward's model were adapted from previous work of Wilkins, Florence and Hetherington. Den Reijer's model [43] is the most recent analytical model found in the literature. The model is a little more sophisticated than Woodward's model in that it takes into account projectile mushrooming effects. A different approach to armour design was investigated by Ben Dor et al. [44], whose objective function was geared towards armour weight minimization.

The analytical models used for this study were first presented to describe the approach. This has already been treated extensively in the literature, but is repeated here for completeness. The intention of this study is not to improve the existing analytical models but instead to use them together with an optimization tool to produce efficient armour designs. Thus based on separate studies by Florence, Hetherington and Ben Dor et al., this study investigates the optimization of a two-component armour where genetic algorithms (GAs) are used as optimizer. Hence, the intention of this study is to narrow down the number of target configurations (armour designs), so as to mitigate the experimental cost and time required for numerical simulations. The study compares ballistic limit and areal density found by the GA program with results in the literature as well as theoretically generated results.
Comparison of the GA generated results with those available in the literature is presented only when applicable. This is mainly due the fact that the limited results available cannot be easily correlated due to differences in target and projectile materials, nose shapes, impact geometries and projectile speeds.

Various projectile types and geometries are studied at various impact velocities.
Typical values obtained from analytical or even numerical studies will not necessarily agree with experimental results due to the inability of the former approaches to fully capture all deformation mechanisms entailed in the penetration process. The three analytical models used for this study do not take into account the following mechanisms: crack initiation and propagation, petalling, brittle fracture, ductile hole growth, plugging and fragmentation. As such, the simplicity of the analytical model used justifies its extensive use in the literature and its use as a starting point to build new analytical models.

Hence one major objective of this study is to provide an idea of the type of materials required for the front and back plate, and the thicknesses of these materials. A database of numerous materials and their properties was used. The optimized results are presented, which includes the ballistic limit, the target configuration, and the materials and its properties. Some studies are also carried out to optimize the areal density of the armour given a particular ballistic limit.

7.2 Areas of Research of Composite Armour

The design of an armour is a very complex task primarily because the mechanisms involved in the armour penetration process are very complicated. Also, the penetration process is very rapid requiring sophisticated equipment to assess it. Today there are three main areas of research delving into the design of composite armour. They are analytical modeling, numerical simulation, and experimentation.

7.2.1 Analytical Modeling

Analytical Modeling entails finding a simple model to represent the penetration mechanism. There are five popular analytical methods found in the literature. Among these, the models proposed by Florence [33], Hetherington [41] and Ben-Dor et al. [44] are studied. Analytical models can provide a fast and simple way of studying the penetration mechanisms without losing much accuracy with respect to numerical codes.

7.2.2 Numerical Simulation

An alternative approach is to use numerical simulation. Numerical simulation involves the use of numerical finite element and finite difference codes to study the impact process. One such finite element code that is popular for impact analysis is LS-DYNA [120]. These codes usually require some material properties, which are very often not known. Another major disadvantage of numerical simulation is that they require large computing time even on powerful computers.

7.2.3 Experimental Studies

Experimentation is a widely used method in the literature. Empirical methods are popular because they offer reliability, but they are extremely expensive and the results do not give enough information such as: the history of the projectile or the trends when changing the configurations [121]. Often experimental results are supplemented by numerical simulations based on 2D continuum mechanics codes similar to those created by Laible [122], Wilkins [123] and Wang et al. [124].

7.3 Theoretical Modeling

There are two types of problem formulation common to the design of two-composite armours. In the first scenario we let the objective function be the ballistic limit and assume a certain impact velocity. Here we attempt to maximize the ballistic limit subjected to some constraints. In the second scenario we attempt to minimize the areal density (weight of the armour) assuming a particular ballistic limit, subjected to some constraints. In this thesis, both cases were studied for various threats. The formulation for the first scenario is presented below.

Note that the criteria presented provide a quick and simple tool for the armour design engineers interested in exploring the many configurations of the armour. Thus, since analytical models are the main focus of this thesis, detailed formulation of their use is presented below.

7.3.1 Analytical Models

Florence [33] appeared to be among the first to develop a model for estimating the ballistic limit. The model was based on the assumption that the ceramic front plate of the armour only distributes the impact load over a large area onto the back plate, which later absorbs all the energy of impact (see Figure 7.3.1 adapted from Ref. 124).

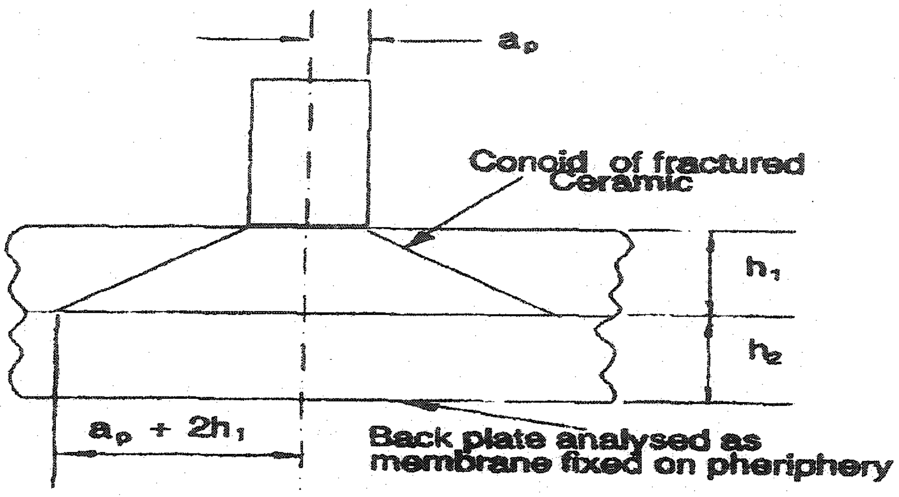

a_p
Conoid of fractured Ceramic
h_1
h_2
$a_p + 2h_1$
Back plate analysed as membrane fixed on pheriphery

Figure 7.3.1: Florence's model

Hetherington [41] later presented a reworked model based on Florence ideas to obtain the optimum thickness ratio between the front and backing plates under a given areal density, while keeping the total thickness of the armour constant. Woodward [42] also proposed a 1D model for the perforation of ceramic armour using a lumped mass concept, which takes into account the erosion of both projectile and ceramic and allows for the consideration of both thin and thick backing.
Recently, Den Reijer [43] presented a more sophisticated model allowing for ceramic and projectile erosion, projectile mushrooming and different deformation models for the backing plate, with reference to the constitutive behavior of comminuted ceramics.

This thesis is based solely on the work of Florence, Hetherington and Ben-Dor et al. Although not mentioned before, Woodward model and Den Reijer' s model adapted their ideas from Florence, Hetherington and other researchers before them. The thesis will first present the models formulation and then the results obtained from this formulation for a known armour material and a known projectile. If the subsequent results are feasible, that is, they compare well with the literature or seem practical; the GA is then employed as an optimization tool.

7.3.1.1 Florence Model

Florence model because of its simplicity and its lack of dependency on experimental data has received a lot of attention by researchers as a tool to guide armour design. The model assumes that a short cylindrical rod striking normally into the ceramic plate would force it to break progressively into a cone of fractured material, which distributes all the impact energy to the backing plate over a larger area than the projectile diameter [125] (see Figure 7.3.1 above). The backing plate is also assumed to deform as a uniform membrane under constant tension. Note that since the energy dissipated in ceramic fracture and projectile erosion is ignored, the mechanism for backing plate failure is simplified. The expression obtained by Florence for the ballistic limit of a ceramic/aluminium composite armour is as follow:

$$V_p = \sqrt{\frac{\varepsilon_2 \sigma_2 h_2}{0.91 m_p f(a)}} \qquad (1)$$

where

$$a = a_p + 2h_1 \qquad (2)$$

and

$$f(a) = \frac{m_p}{[m_p + (h_1 d_1 + h_2 d_2)\pi a^2]\pi a^2} \qquad (3)$$

where Vp is the ballistic limit; ap is the radius of the projectile; m_p is the mass of the projectile, ε is the breaking strain; σ is the UTS, h is the thickness of the plates, and d is the density of the plates. Subscript 1 and 2 represent the front plate and back plate up respectively.

7.3.1.2 Hetherington Model

Based on Florence model, Hetherington developed an equation to obtain the optimum thickness ratio. In the following analysis an expression of the thickness ratio of the front to back plate will be derived which gives the best protection at a constant value of total thickness. Note that the areal density is not constant and if the total thickness of the armour is taken as T then:

$$h_2 = T - h_1 = T - \frac{a - a_p}{2} \tag{4}$$

Substituting Eqn. (4) into Eqn. (1) yields:

$$\frac{0.91 V_p^{\,2} m_p}{\varepsilon_2 \sigma_2} = \frac{T - \frac{a - a_p}{2}}{f(a)} \tag{5}$$

where a is the only variable.

Differentiating the equation with respect to a, setting V_p to zero and simplifying gives:

$$\frac{0.91 \times 2 V_p \dot{V}_p m_p}{\varepsilon_2 \sigma_2} = \frac{\frac{f(a)}{2} + \dot{f}(a)(T - \frac{a - a_p}{2})}{f^2(a)} = 0$$

or:

$$\frac{f(a)}{\dot{f}(a)} = -2T + a - a_p \tag{6}$$

From Eqn. (6):

$$\dot{f}(a) = -\frac{f(a)[2m_p + 4Td_2 \pi a^2 + \frac{d_1 - d_2}{2}\pi(5a - 4a_p)a^2]}{m_p a + Td_2 \pi a^3 + \frac{d_1 - d_2}{2}\pi(a - a_p)a^3} \tag{7}$$

Combining Eqn. (6) and (7) with much manipulation gives:

$$\begin{aligned} &3(d_1 - d_2)\pi a^4 + 5\pi[(d_1 - d_2)a_p + (2d_2 - d_1)T]a^3 \\ &- 2\pi[4T^2 d_2 + 4(d_1 - 2d_2)Ta_p + (d_1 - d_2)a_p^{\,2}]a^2 \\ &+ 3m_p a - 2m_p(2T + a_p) = 0 \end{aligned} \tag{8}$$

Once the projectile and armour specs are known, Eqn. (8) is used to find a. With a known the optimum thickness ratio can be found via:

$$\frac{h_1}{h_2} = \frac{\frac{a - a_p}{2}}{T - \frac{a - a_p}{2}} = \frac{a - a_p}{2T - (a - a_p)} \tag{9}$$

7.3.1.3 Ben-Dor et al. Model

Thus far we sought to maximize the ballistic limit, with the armour thickness and material as design variables. Now a different aspect of armour design is investigated, where the objective function is geared towards armour weight minimization. In order to facilitate this task the work and ideas of Ben Dor et al. [44] are solicited. Ben Dor et al. goals were to extend Florence and Hetherington's model to any material so as to seek to fully minimize the weight of the armour. To the author's knowledge, the other two models presented above investigated only ceramic/aluminium and ceramic/GFRP combinations. The dimensionless formulas presented by Ben Dor et al. allow us to investigate the problem of armour design completely and for a general case, namely arbitrary combinations of materials for the front and back plates.

Using Florence model to determine the ballistic limit, Ben Dor introduced a multiplication factor α (see Eqn. 10), which can only be determined from available experimental data in order to increase the accuracy of the predictions. When $\alpha = 1$ the model becomes Florence model:

$$V_p^{\,2} = \frac{\alpha\varepsilon_2\sigma_2 h_2 z[(\rho_1 h_1 + \rho_2 h_1)z + m_p]}{0.91 m_p^{\,2}} \qquad (10)$$

where $z = \pi(R + 2h_1)$

ρ_1 and ρ_2 are the density of the front and back plate respectively, and R is the radius of the projectile.

The objective of this section of the thesis is to use the ideas of Ben Dor et al. to find the thickness of the plates h_1 and h_2 which provide the minimum areal density of the armour, A, for a given ballistic limit V_p.

The following formulation of dimensionless variables adapted from the work by Ben Dor et al. [44] is necessary to begin:

$$\bar{h}_i = \frac{h_i}{R}, \qquad \bar{\rho}_i = \frac{\pi R^3 \rho_i}{m_p}, \qquad i = 1,2 \,, \qquad (11)$$

$$\overline{w}^2 = V_p\sqrt{\frac{0.91\rho_2}{\alpha\varepsilon_2\sigma_2}}, \qquad \overline{A} = \frac{\pi R^2 A}{m_p}$$

The definition of the areal density A is repeated here for clarity:

$$A = \rho_1 h_1 + \rho_1 h_1 \tag{12}$$

Eqn. 10 and Eqn. 12 can be rewritten respectively in dimensionless form as:

$$\overline{w}^2 = \overline{\rho}_2\overline{h}_2\overline{z}[(\overline{\rho}_1\overline{h}_1 + \overline{\rho}_2\overline{h}_2)\overline{z} + 1] \tag{13}$$

$$\overline{A} = \overline{\rho}_1\overline{h}_1 + \overline{\rho}_2\overline{h}_2 \tag{14}$$

where $\overline{z}$ is define as:

$$\overline{z} = \frac{z}{\pi R^2} = (1 + 2\overline{h}_1)^2$$

Eqn. 13 is a quadratic equation in terms of $\overline{\rho}_2\overline{h}_2$, solving for this term and substituting it into Eqn. 14 yields:

$$\overline{A}(\overline{h}_1, \overline{\rho}_1, \overline{w}) = \frac{\overline{\rho}_1\overline{h}_1\overline{z} - 1 + \sqrt{(\overline{\rho}_1\overline{h}_1\overline{z} + 1)^2 + 4\overline{w}^2}}{2\overline{z}} \tag{15}$$

Eqn. 15 is now a function of only one variable $\overline{h}_1$; $\overline{h}_2$ can then be found using Eqn. 14.

It is important to realize that the dimensionless areal density $\overline{A}$ is a function of one variable and depends on only two parameters, that is, $\overline{\rho}_1$ and $\overline{w}$.

Let $\overline{h}_1^{opt} = \varphi_1(\overline{\rho}_1, \overline{w})$ be the function that provides the minimum $\overline{A}$, then the dimensionless minimum areal density $\overline{A}^{opt}$ and the optimal ratio of the areal density of the second plate to the areal density of the first plate are also functions of $\overline{\rho}_1$ and $\overline{w}$.

$$\overline{A}^{opt} = \overline{A}[\varphi_1(\overline{\rho}_1, \overline{w}), \overline{\rho}_1, \overline{w}] = \varphi_A(\overline{\rho}_1, \overline{w}) \tag{16}$$

$$\begin{aligned} \frac{\overline{\rho}_2\overline{h}_2^{\,opt}}{\overline{\rho}_1\overline{h}_1^{\,opt}} &= \frac{\rho_2 h_2^{\,opt}}{\rho_1 h_1^{\,opt}} = \frac{\overline{A}^{opt}}{\overline{\rho}_1\overline{h}_1^{\,opt}} - 1 \\ &= \frac{\varphi_A(\overline{\rho}_1, \overline{w})}{\overline{\rho}_1\varphi_1(\overline{\rho}_1, \overline{w})} - 1 \end{aligned} \tag{17}$$

and

$$\begin{aligned}\overline{\rho}_2\overline{h}_2^{opt} &= \overline{A}^{opt} - \overline{\rho}_1\overline{h}_1^{opt} \\ &= \varphi_A(\overline{\rho}_1,\overline{w}) - \overline{\rho}_1\varphi_1(\overline{\rho}_1,\overline{w}) \end{aligned} \tag{18}$$

Once the dimensionless parameters are obtained the results for a given material can be determined:

$$\begin{aligned} h_1^{opt} &= R\varphi_1(\overline{\rho}_1,\overline{w}), \\ h_2^{opt} &= \frac{m_p}{\pi R^2 \rho_2}\varphi_2(\overline{\rho}_1,\overline{w}), \\ A^{opt} &= \frac{m_p}{\pi R^2}\varphi_A(\overline{\rho}_1,\overline{w}) \end{aligned} \tag{19}$$

where the superscript *opt* indicates the optimal parameters, and $\overline{\rho}$ and $\overline{w}$ are determined from Eqn. 11.

As will become evident later Ben-Dor et al. formulation reduces the number of design variables required for optimization thus reducing the time required for calculations.

7.4 Theoretical results for Florence and Hetherington model

The armour materials used to study Florence and Heterington's model formulation is given in Table 7.4.1. Three different threats namely, the NATO 7.62mm armour piercing (AP) rounds, mp=10g; 0.5 in. ball, mp=42g; and 20mm fragment simulating projectile (FSP), mp=50g are all studied. The mass of the projectile was assumed in some cases when not given.

Table 7.4.1. Physical properties employed for armour

Material properties	front plate	backing plate
	Aluminium 6061-T6	Ceramic 94% Al_2O_3
Density ρ (kg/m^3)	2712	3620
UTS σ (Gpa)	0.31	0.18
Breaking strain ε_2	0.17	—

Figure 7.41 to Figure 7.4.4 show the ballistic limit versus the thickness ratio for the three threats at various values of total thickness.

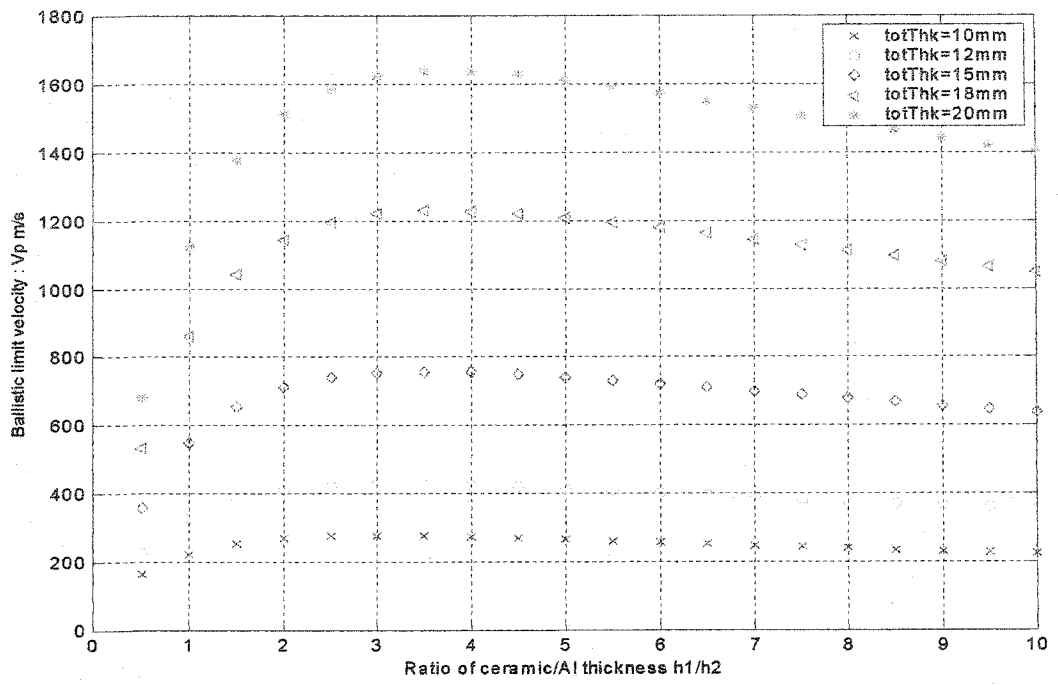


Figure 7.4.1: Theoretical plot of h1/h2 vs. ballistic limit Vp: projectile threat: 7.62mm armour piercing (AP)

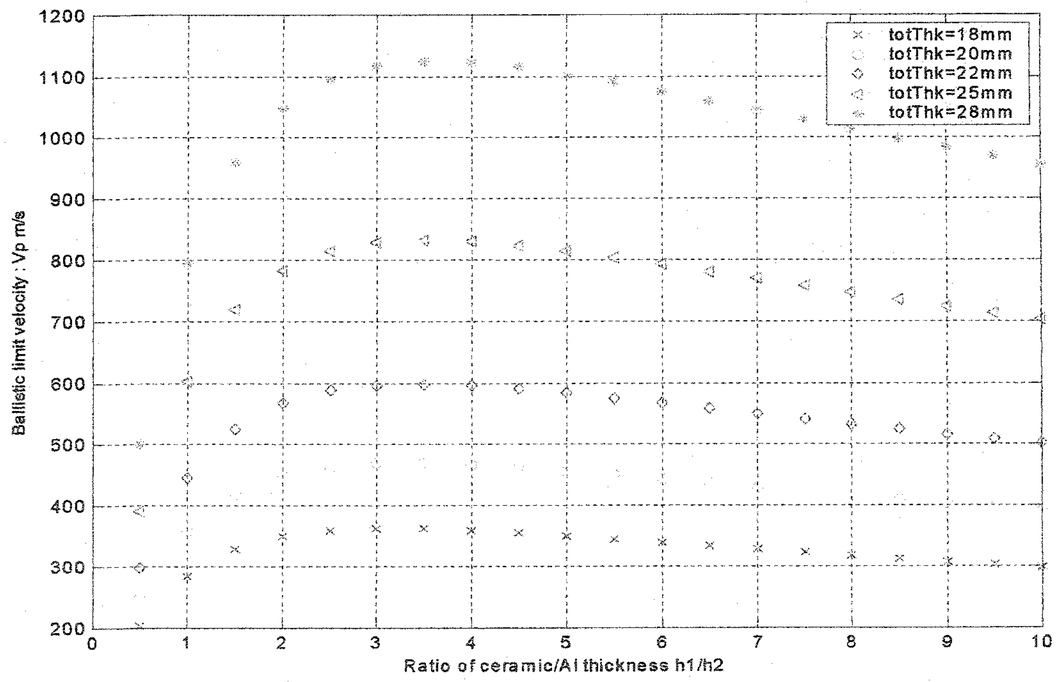


Figure 7.4.2: Theoretical plot of h1/h2 vs. ballistic limit Vp: projectile threat: 0.5 in ball

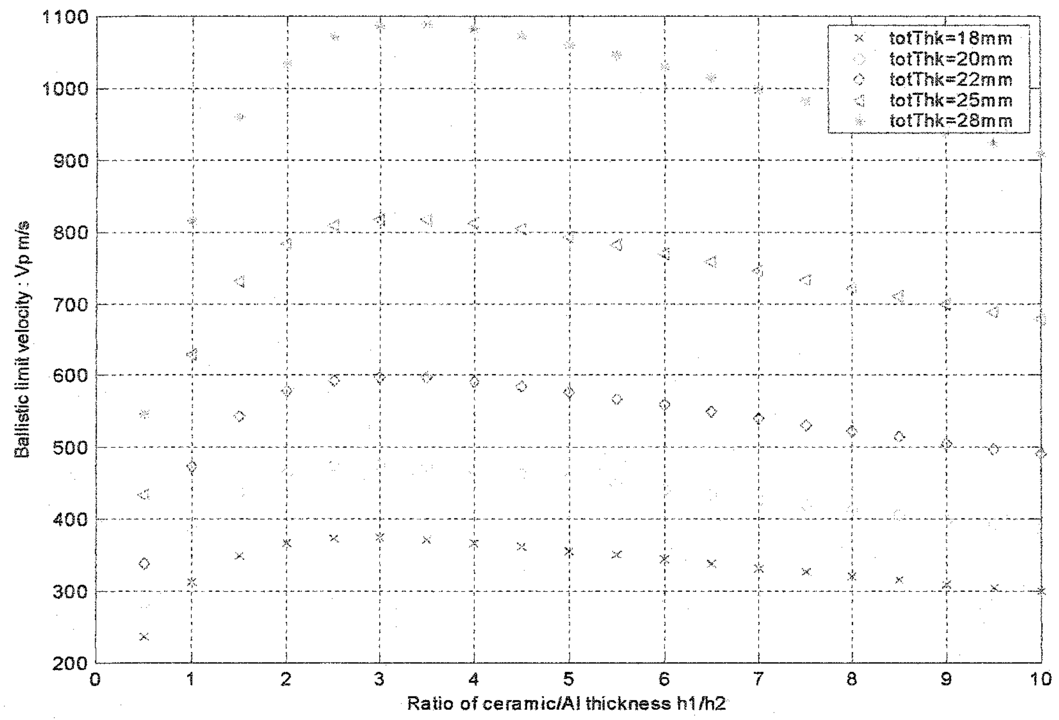


Figure 7.4.3: Theoretical plot of h1/h2 vs. ballistic limit Vp projectile threat: 20mm fragment simulating projectile

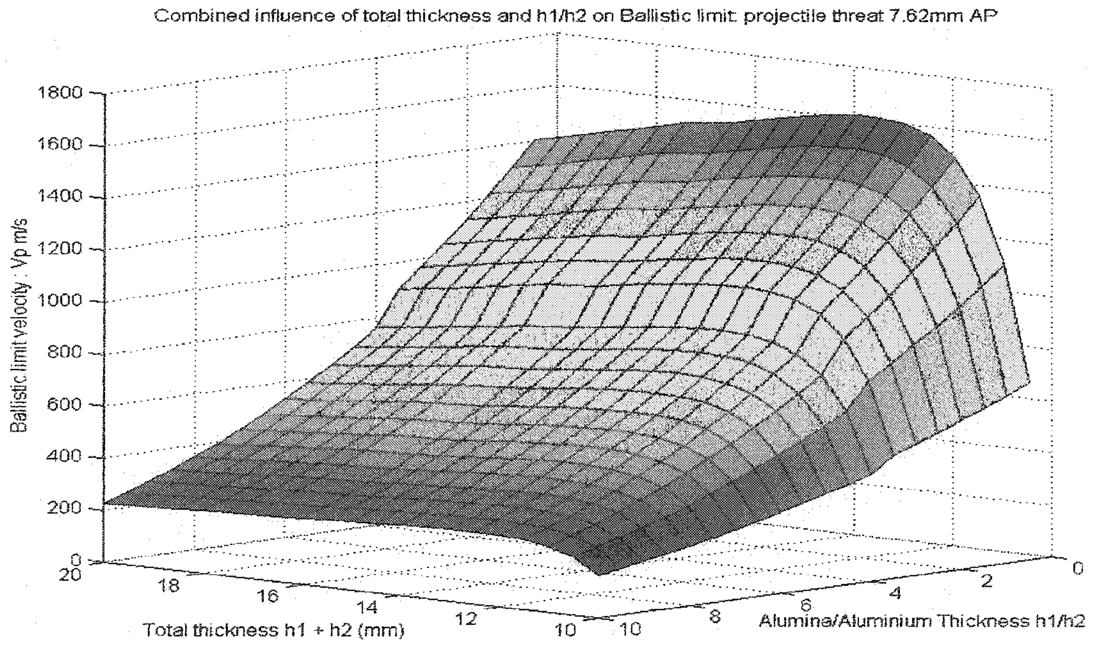


Figure 7.4.4: Combined influence of the total thickness and the thickness ratio on the ballistic limit: 3D plot of h1/h2, h1 + h2 and Vp.

Calculations were carried out using Eqn. 8 and Eqn. 9. From the figures above we glean that at constant total thickness the ballistic limit increases rapidly, then slowly decreases as the thickness ratio increases. More importantly we notice that the armour performs best at thickness ratio between 3 and 4. As the thickness ratio increases beyond this range the armour performance reduces.

From the figures above we can also glean that as total thickness increases the protective level increases. Also, careful examinations of Figure 7.4.1 to 7.4.3 would also illustrate that the optimum thickness ratio increases slightly as total thickness increases. Due to the difference in mass of the projectiles, the armour shows the best level of protection against the 7.62mm AP and the least level of protection against FSP. Figure 7.4.4 shows a 3D plot of the combined influence of ballistic limit, total thickness and thickness ratio for the NATO 7.62 mm AP round.

Since there are no experimental data available to draw any comparison, the results do compare well with published results [125], which report an optimum thickness ratio around 3 for the specific armour materials employed. Hence the next step was to implement the analytical model of Florence and Hetherington to test its viability in armour design.

The GA problem definition was developed using Florence and Hetherington's models already presented. Initially, Florence model was applicable to only ceramic faced and aluminium backed armours. Hence a database of 31 different types of ceramics materials and 56 different types of aluminium was employed. The database contains all necessary material properties required by the model formulation, such as density, ultimate tensile strength (UTS), breaking strain etc. The armour design is implemented into the GA for optimization using the following formulation:

First we define the threat: 7.62 mm NATO AP where $m_p = 10g$

Objective function: Maximize (Vp)

Design Variables: h1/h2, total thickness, material index for front plate, material index for back plate

Constraints: (1) h1/h2 >1

(2) total thickness = h1+h2 > 10mm

h1/h2 is allowed to vary between 0.5 and 15 while total thickness of the armour varies between 10mm and 15mm. Two separate databases are used; one for the front plate whose index range from 1 to 31, and the other for the backup plate whose index range from 1 to 56. A population size of 20 was used while the probability of crossover was taken as 0.85 and the probability of mutation was taken as 0.01. 200 generations was used as the stopping criteria. The results found from two tests are shown below. The convergence history for run number two is presented below when the 7.62mm threat is studied, simply to illustrate the GA search process.

7.4.1 Results after analytical model implementation into GA

Threat: *7.62 mm NATO AP where* $m_p = 10g$

Two different runs were conducted for the above threat and the results are given below:

Run # 1:

```
The fittest individual so far is 1252.26 at generation number 200

The degree of violation of our constraint(s) is 0 percent

Design Variables =

    3.8175     0.015     26     39

elapsed time =

    13.689
```

Run # 2:

```
The fittest individual so far is 1242.4 at gen# 200

The degree of violation of our constraint(s) is 0.89 percent

Design_Variables =

    3.2143     0.015     25     39

elapsed_time =
    13.719
```

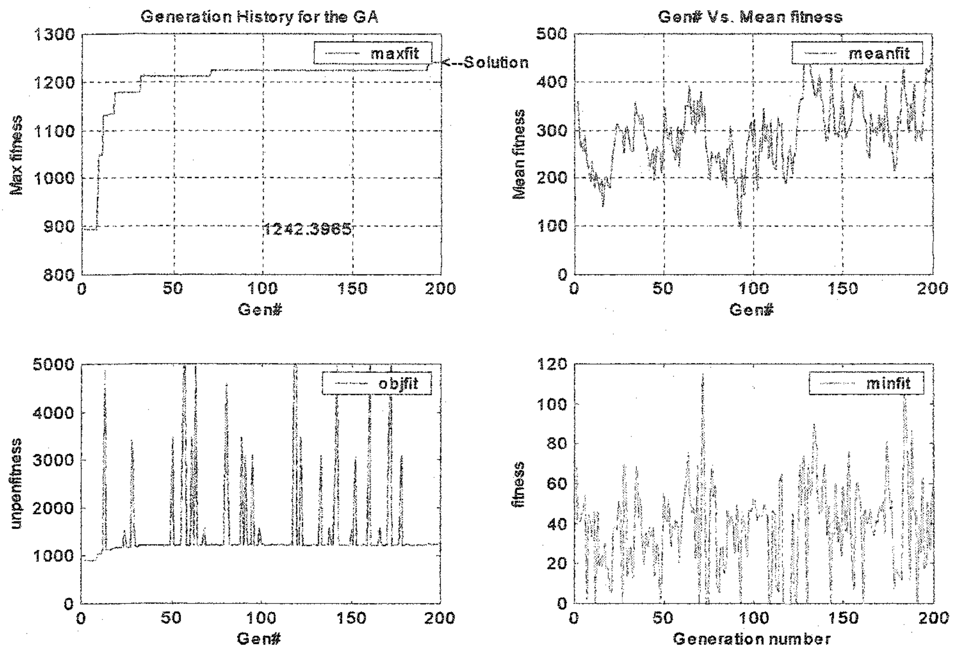


Figure 7.4.5: Convergence history of GA.

For the materials that correspond to material index of 25, 26 and 39, please consult Appendix A.

Threat: *12.7 mm NATO AP where* $m_p = 42g$

Two different runs were conducted for the above threat and the results are given below:

Run #1:

The fittest individual so far is 517.221 at gen# 200

The degree of violation of our constraint(s) is 0 percent

```
Design_Variables =

    3.5159     0.015     26     39

elapsed_time =

    12.087
```

Run #2:

The fittest individual so far is 489.768 at gen# 200

The degree of violation of our constraint(s) is 0 percent

```
Design_Variables =

    2.9127    0.014683      26      39

elapsed_time =  12.057
```

Threat: 20 mm FSP where $m_p = 50g$

Two different runs were conducted for the above threat and the results are given below:

Run # 1:

```
The fittest individual so far is 457.111 at gen# 200

The degree of violation of our constraint(s) is 0 percent

Design_Variables =

    2.4603      0.015      25      35

elapsed_time =      12.658
```

Run # 2:

```
The fittest individual so far is 459.446 at gen# 200

The degree of violation of our constraint(s) is 0 percent

Design_Variables =

    2.7619      0.015      25      35

elapsed_time = 12.438
```

7.4.2 Discussion of results obtained by the GA

All the results obtained by the GA seem reasonable when compared to Figure 7.4.1 to Figure 7.4.4 – the theoretical results. For instance, -although not shown here – if the range of total thickness increases e.g. 10-20mm then the ballistic limit increases but more importantly the areal density (weight) of the armour increases. Note also that the GA in most runs selected the highest total thickness. We can also glean that an increase in thickness ratio leads to an increase in ballistic limit, which agrees with the theoretical results obtained via the aforementioned models.

At this stage of the research the results obtained are sufficient to warrant that the genetic algorithms can be a useful tool for armour design and optimization. Up to this point however, no mention was made of weight or energy absorbed by armour. These are very important concepts that must be addressed.

7.5 Areal Density Consideration

In armour design, the demand is to have the best protection at a minimum weight. Armour cost could also become important. The purpose of this section is to seek the ratio of front plate thickness to back plate thickness, which will provide a specified level of protection at a minimum weight. Thus for a fixed areal density, A, the aim is to determine h1/h2 for a composite armour which will give the best performance. This requires a slight adjustment to the above two formulations as presented below after adaptation from Ref. 41. Some formulas are repeated here for clarity.

Areal density= $A = h_1 d_1 + h_2 d_2$ (20)

From Eqn. 20 solve for h_2

$$h_2 = \frac{1}{d_2}(A - h_1 d_1) \tag{21}$$

Using Eqn. 2 substitute h_1 into Eqn. 21:

$$h_2 = \frac{1}{d_2}(A - \frac{d_1}{2}(a - a_p)) \tag{22}$$

Substituting h_2 from Eqn 22 into Eqn. 1 give an expression in which the only variable is a:

$$V_p = (\frac{\varepsilon_2 \sigma_2 (A - 0.5 d_1 (a - a_p))}{0.91 m_p d_2 f(a)})^{1/2} \tag{23}$$

where $f(a)$ can be obtained via Eqn. 3.

Combining Eqn. 2 and Eqn. 20 with a lengthy manipulation yields:

$$\frac{h_1}{h_2} = \frac{d_2(a - a_p)}{2A - d_1(a - a_p)} \tag{24}$$

Recall Florence model and Hetherington's model assumes a characteristic geometry for the fracture conoid in the ceramic front plate.

7.5.1 Results after Areal density consideration

Using the material properties of alumina for front plate and material properties of alumunium for back plate along with 7.62 mm AP round (see specific data below), Figure 7.5.1 was obtained via above formulation.

Data input to the program:
Mass of projectile (mp) = 10g
Diameter of projectile = 7.62mm.
Density of ceramic (94% Al2O3)= 3620 kg/m^3
Density of backing (Al 6061-T6) = 2712 kg/ m^3
Breaking strain of backing = 0.17
UTS of backing = 0.31e9 N/m^2

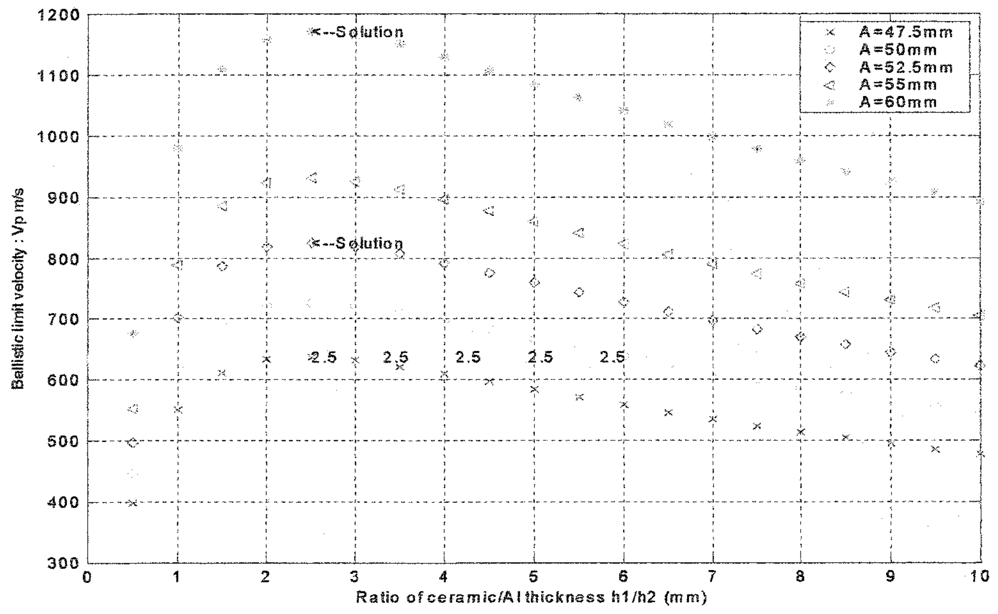


Figure 7.5.1: Theoretical dependence of ballistic limit on h1/h2 for contours of constant areal density.

From Figure 7.5.1 one can glean that the peak of the curve is quite flat and thus many optimum values of thickness ratio exist for a given areal density. The five values shown on the curve indicates the first thickness ratio at which the maximum ballistic limit was found. It is critical to note from Figure 7.5.1, that the position of the optimum thickness ratio is insensitive to the areal density. This means that once an optimum thickness ratio for a particular material is known we can change the areal density without affecting the position of the optimum thickness ratio. Hence, the "rule of thumb" implies that, for hard faced two-component armours, the position of the peak (optimum) is insensitive to the scale of the armour, it being determined principally by the density of the materials employed [41]. Thus in order to investigate the effect of areal density with the GA, a twist to the problem definition previously presented is given below.

7.5.2 Results after Areal density consideration is implemented into GA

Objective function: Maximize (Vp)
Design Variables: h1, h2, material index for front plate, material index for back plate
Constraints: (1) Areal density < 45 kg/m^2
(2) h1/h2 > 1

The above problem formulation employed a material database of 31 different ceramics and 56 different aluminium impacted by a 7.62 mm AP threat ($m_p = 10g$).The twist to the formulation of Florence and Hetherington model presented above was implemented into MATLAB codes. These codes were then integrated into the GA. A 12.7mm caliber ($m_p = 30g$) was also tested and the results compared with the literature. The results from four different runs are presented in Table 7.5.1 for the 7.62 mm AP.

Table 7.5.1. GA generated results

Run#	h1(m)	h2 (m)	mat1	mat2	Vp (m/s)	CPU time (s)
1	0.009206	0.00744	14	3	810.877	3.285
2	0.007587	0.006524	30	42	1013.2	4.186
3	0.010286	0.005143	30	35	947.477	5.328
4	0.015143	0.005143	14	39	1771.57	4.213

mat1=material index for front plate
mat2=material index for back plate

Using the previous problem definition, a sensitivity study was conducted to check the effects of an added constraint, changing the projectile type and changing the mass of the projectile.

7.5.2.1 Sensitivity Study

Various scenarios on constraints changes, projectile type changes and projectile mass changes are explored and investigated to check the effects on ballistic limit. The effects and results of the aforementioned changes are presented below.

7.5.2.1.1 Effect of added constraint

Using the above formulation, a third constraint was added while everything else remained the same.

Constraint (3): Total Thickness $= h_1 + h_2 < 16mm$

Two test were conducted to check the effect of adding the above constraint and the results are shown below:

Run #1:

```
The fittest individual so far is 947.477 at gen# 400

The degree of violation of our constraint(s) is 0 percent

Design_Variables =

   0.010286   0.0051429      30      39

elapsed_time =      7.911
```

Run #2:

```
The fittest individual so far is 947.477 at gen# 200

The degree of violation of our constraint(s) is 0 percent

Design_Variables =

   0.010286   0.0051429      30      39

elapsed_time =

    10.826
```

Note that the addition of a constraint on total thickness had no effect on the armour design. In fact without this constraint in place the exact design was found as illustrated in Table 7.5.1. Thus it can be argued that a constraint on thickness ratio is sufficient to capture the geometry aspect of the armour design.

7.5.2.1.2 Effect of changing projectile type

Having investigated the effect of adding a constraint on total thickness, the next step was to study the effect of changing the projectile. The projectile was changed to a 12.7mm caliber in which the mass of the projectile is 30g. The results obtained using the above formulation with the three constraints are given in Table 7.5.2.

Since the GA is searching for the best sequence of design variables to satisfy the imposed constraints then an increase in projectile mass and radius is accompanied by a decrease in ballistic limit.

Table 7.5.2. Results after constraint 3 was added

Run#	h1(m)	h2 (m)	mat1	Mat2	Vp (m/s)	CPU time (s)
1	0.010286	0.005143	30	39	414.973	5.208
2	0.010286	0.005143	30	39	414.973	15.031

Note that the diameter and mass of the projectile has significant effects on the ballistic limit of the armour. It can be said that as the mass and diameter of the projectile increases, the constraints on areal density and thickness ratio has to be relaxed. In addition, more advanced materials can be added to the material databases in order to find an armour design of substantial ballistic limit.

7.5.2.1.3 Effect of increasing the mass of the projectile

Finally a 12.7mm caliber was used but the mass of the projectile was taken as 50g. Again the formulation above is adopted with three constraints. The following results were obtained:

```
The fittest individual so far is 267.712 at gen# 200

The degree of violation of our constraint(s) is 0 percent

Design_Variables =

   0.010286   0.0051429     30     39

elapsed_time =      14.901
```

Note that the mass of the projectile has tremendous influence on the ballistic limit of the armour. Note also that the results found by the GA, which includes the thicknesses of the armour plate as well as the material used, are identical to results previously obtained. To explain these findings we need only consult Florence model. From Florence model we know that an increase in projectile mass leads to a decrease in the ballistic limit. The thicknesses of the plates and materials are identical because the GA finds these variables to provide a favorable combination to satisfy the imposed constraints.

7.5.3 Discussion on the concept of Areal Density

A few conclusions can be drawn from the results obtained after delving into the study of the effects of areal density on the armour design. First and foremost, for a given areal density better performance can be obtained with thickness ratios greater than 1. In addition, the type of threat and mass and diameter of the threat significantly influences the performance of the armour. For the materials employed experimental results obtained by Hetherington [41] (see Figure 7.5.2) confirmed that the theoretical optimum ratio of 2.5 would give an armour with a near optimum performance.

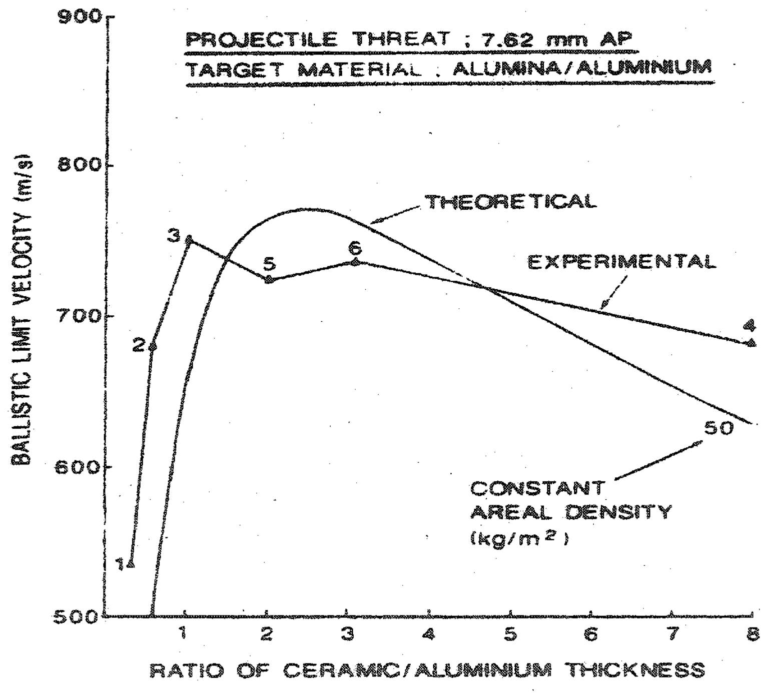


Figure 7.5.2: Dependence of ballistic limit (Vp) on h1/h2 for constant areal density – comparison of theoretical and experimental results

Further research is needed to assess the validity of the model to other types of material combinations. Finally for a given areal density, good performance can be obtained with thickness ratios of one or more than one; ratios less than one can lead to greatly reduced armour performance.

Thus far we have addressed the concept of areal density. As this chapter progresses one hopes to have energy and all previous topics studied glued together to better capture the problem of the armour design. Hence, the next stage of this section of the thesis addresses the concept of energy absorption by the armour.

7.6 Energy absorption Considerations

In order to study the concept of energy absorption, a paper titled " *An investigation into the energy absorbed during ballistic perforation of composite armour*," by Hetherington and Rajagoaplan [126] was studied. This paper uses ceramic for the front plate and GRFP for the backing. Here we attempt to add a flavor of energy absorbing capabilities of the armour to design and optimization. Armour designers are interested in an armour which gives maximum protection with minimum weight i.e. which combines maximum energy absorbing capability with areal density. Note that throughout this study a short cylindrical rod represented the bullet.

Adopting the ideas from Ref. 18 the residual velocity is taken as

$$V_r = V_i - V_p \tag{25}$$

where the ballistic limit (V_p) is calculated using Florence model and the impact velocity (V_i) is assumed. Once the residual velocity is known the kinetic energy after impact (i.e. the residual kinetic energy) can be obtained.

The energy absorbed by the armour can also be found via:

$$\text{Energy absorbed} = \tfrac{1}{2} m_p (V_i^2 - V_r^2) \tag{26}$$

Note above the mass of the projectile m_p is a constant and was assumed to remain constant (uneroded) after impact. The residual velocity is the velocity after the bullet perforates the armour. Ideally we would like to have $V_i \leq V_p$ so as to minimize the residual energy. The work of Hetherington and Rajagoaplan [126] also attempted to validate the use of glass fiber reinforced plastic (GFRP) for Florence model.

7.6.1 Results after energy absorption consideration

The input data below along with Eqn. 25 and Eqn. 26 were adopted for this study.

Data input to the program:
Mass of projectile (mp) = 46.8g
Diameter of projectile = 12.7 mm. (US Ball M33 0.5in caliber)
Density of ceramic = 3499 kg/m^3
Density of backing (GFRP) = 2229 kg/ m^3
Breaking strain of backing = 0.04
UTS of backing = 1e9 N/m^2
Assumed impact velocity= 880 m/s

Using the above data, Florence model, and Eqn. 25 and Eqn. 26, the ballistic limit, residual velocity and thus energy absorbed by the armour can be obtained. The experimental results from Ref. 126 were adopted to compare with theoretical predictions. From the experimental studies in Ref. [126], ceramic tiles and GFRP of various thicknesses were used with the aim of generating data for 12 different target configurations. The different configurations and their dimensions are given in Table 7.6.1.

Table 7.6.1. Target plate dimensions

Configuration number	Ceramic Tile size (mm)	GFRP plate size (mm)
1	$50.4\times50.4\times4$	$150\times150\times5$
2	$50.4\times50.4\times4$	$150\times150\times8$
3	$50.4\times50.4\times4$	$240\times240\times10$
4	$50.4\times50.4\times6$	$150\times150\times5$
5	$50.4\times50.4\times6$	$150\times150\times8$
6	$50.4\times50.4\times6$	$240\times240\times10$
7	$50.4\times50.4\times9$	$150\times150\times5$
8	$50.4\times50.4\times9$	$150\times150\times8$
9	$50.4\times50.4\times9$	$240\times240\times10$
10	$50.4\times50.4\times18$	$150\times150\times5$
11	$50.4\times50.4\times18$	$150\times150\times8$
12	$50.4\times50.4\times18$	$240\times240\times10$

The experimental data found after firing on the above armour configurations is given in Table 7.6.2, as adapted from Ref. 126 for comparison purposes.

Table 7.6.2. Impact and residual velocities

Configuration number	Projectile impact velocity V_i (m/s)	Projectile residual velocity V_r (m/s)
1	893.2	832.6
2	882.3	826.3
3	881.4	802.4
4	880.7	800.5
5	893.9	802.5
6	878.1	760.6
7	898.2	693.9
8	880.1	658.3
9	882.9	621.5
10	895.6	425.5
11	888.5	329.5
12	876.2	299.3

Using the experimental data provided in Table 7.6.2 along with Eqn. 25 and Eqn. 26 and input data given above, Figure 7.6.1 to Figure 7.6.5 were obtained:

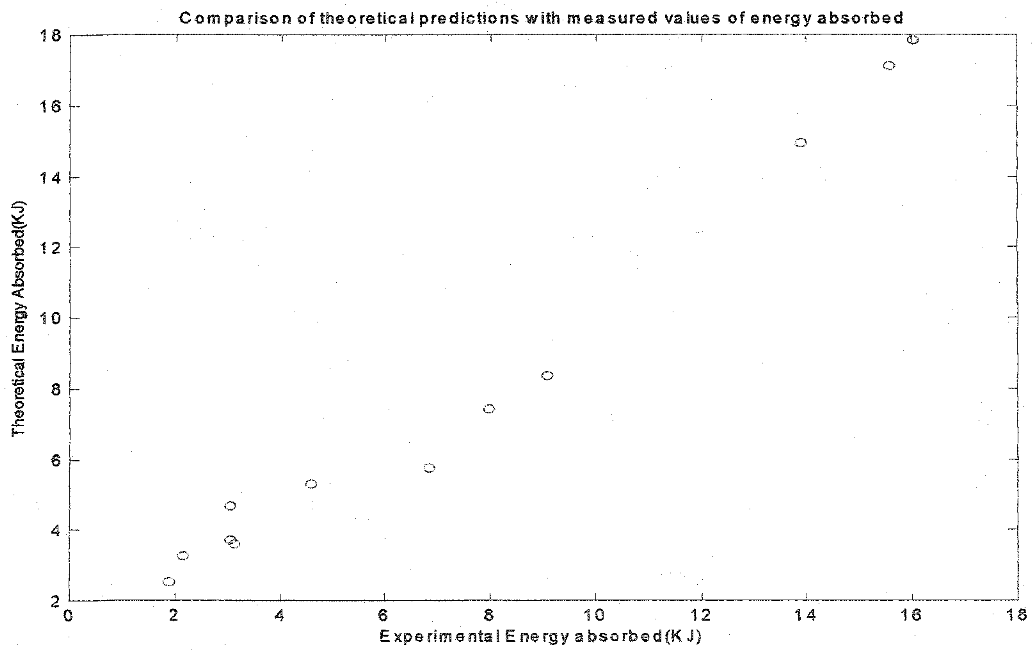


Figure 7.6.1: Comparison of theoretical predictions with measured values of energy absorbed.

Figure 7.6.2 to Figure 7.6.5 were plotted to compare with published results, and to ensure the correctness of the formulation by [126] so that when the GA is fused with the analytical model the correctness and accuracy of the optimized results are not in doubt.

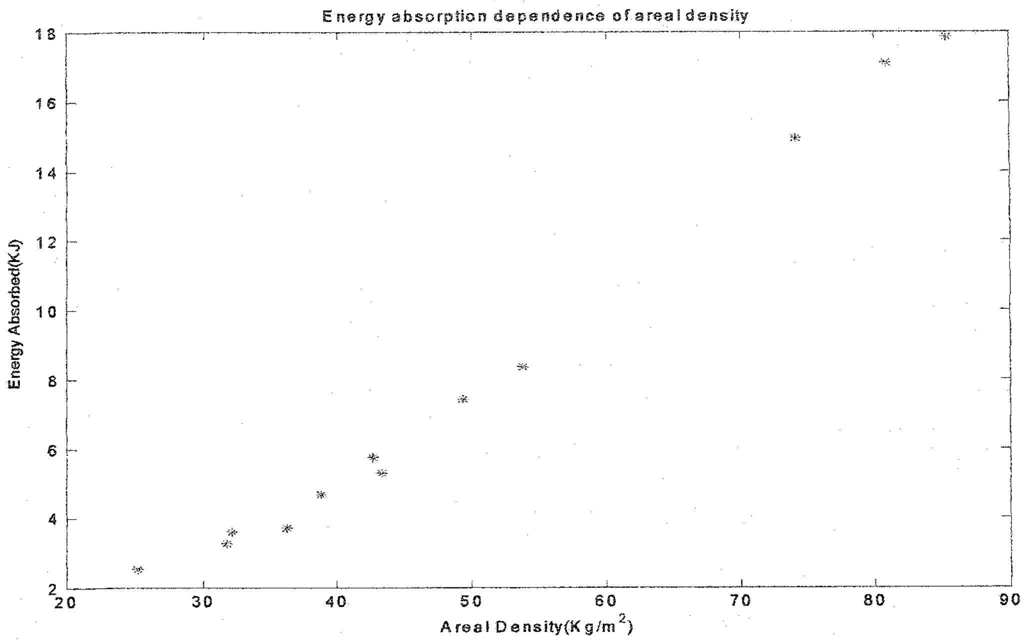


Figure 7.6.2: Energy absorption dependence of areal density

Figure 7.6.2 illustrates the way in which the energy absorbing capacity of the armour increases linearly with increased areal density approximately above 45 kg.m^2.

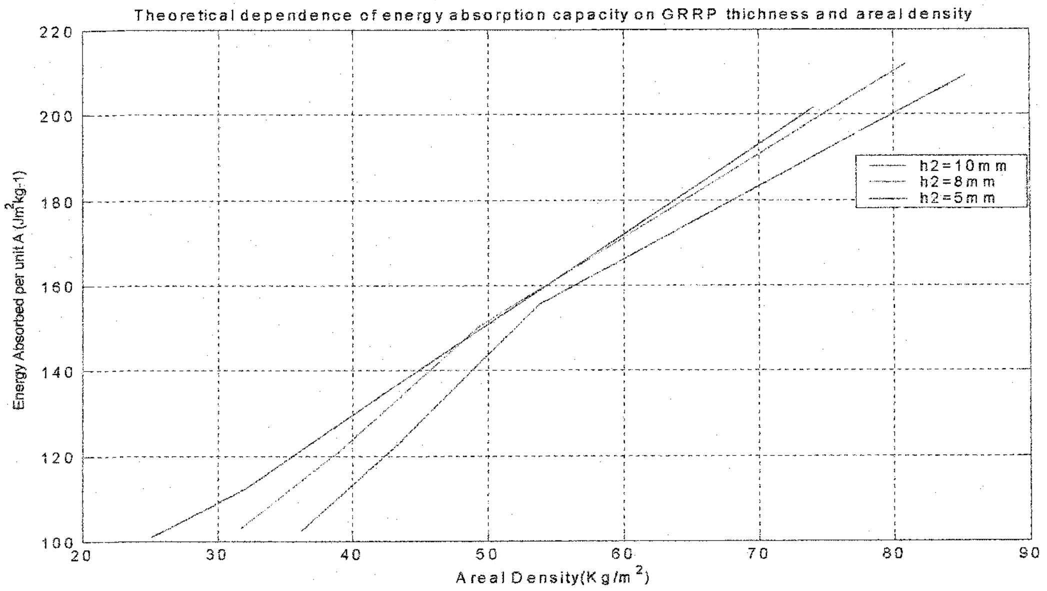


Figure 7.6.3: Theoretical dependence of energy absorption capacity on GFRP thickness and areal density.

Figure 7.6.3 indicates that for a particular value of areal density, more energy is absorbed per unit areal density with lower values of GFRP thickness.

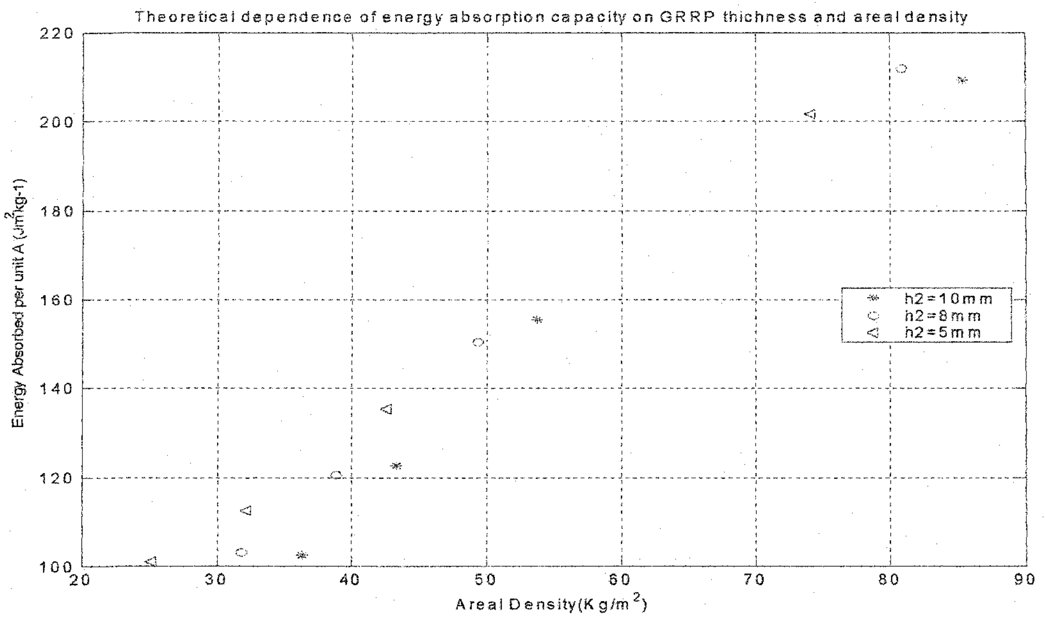


Figure 7.6.4: Dependence of energy absorption capacity on GFRP thickness and areal-density.

Figure 7.6.4 indicates that the thinner the backing plate the greater the energy absorbed per unit areal density.

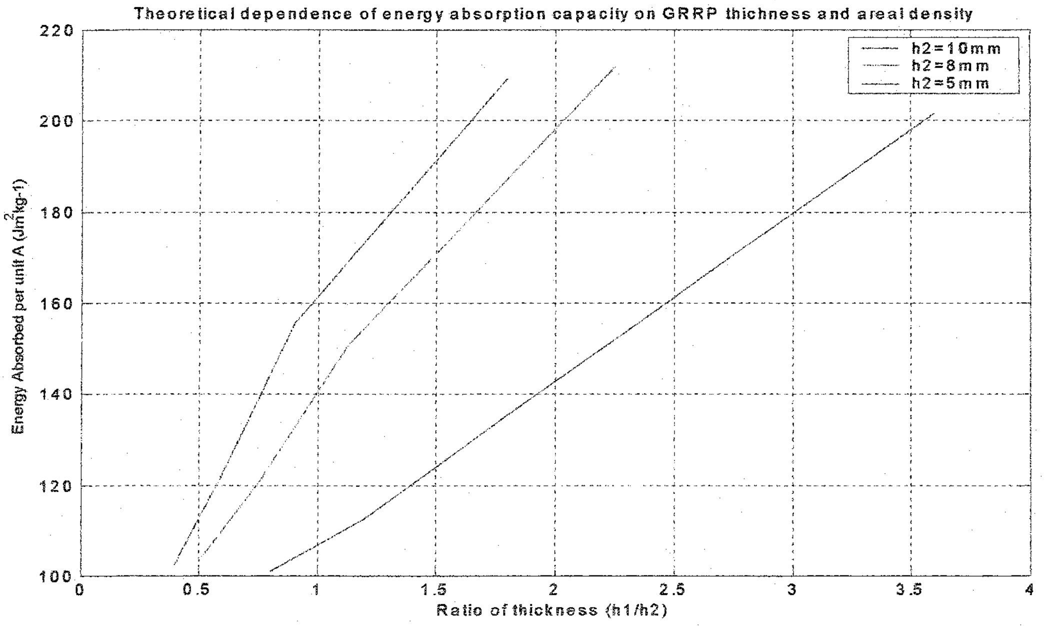


Figure 7.6.5: Dependence of energy absorption capacity on ceramic/GFRP ratio – theoretical predictions

Figure 7.6.1 illustrates a good correlation between experimental and theoretical predictions based on Eqn. 25 and Eqn. 26, thus validating the use of GFRP for Florence model. The other approach for calculating residual energy as argued by Shu [127] in his discussion about the legitimacy of Hetherington's approach was demerit in this study. Figure 7.6.2 shows the variation of energy absorbed with areal density. From Figure 7.6.2 we learn that as areal density increases the energy absorbed increases for areal density approximately above 45 kg/m^2. Some interesting information can also be obtained from Figure 7.6.3. The latter indicates that for a given areal density more energy is absorbed per unit areal density (Jm^2/kg) for lower thickness of GFRP. The experimental results adopted from Ref. [126] are shown in Table 7.6.2, which both illustrate and confirm the theoretical trends. The variation of energy absorbed per unit areal density with thickness ratio is shown in Figure 7.6.5. The energy absorbed per unit areal density increases almost linearly for thickness ratios greater than 0.5.

These findings indicate that GFRP can be added to the materials backing database, since its validity to Florence model was ascertained. In addition, we can add energy absorbed as a constraint to previous GA problem definition. However, in order to do this we need to assume an impact velocity.

From the above results it is clear that the MATLAB codes created to handle the concept of energy absorption works well and thus integration of these codes into the GA is the next stage. The following new problem definition is adopted with the aid of Florence and Hetherington's model.

7.6.2 Results after energy absorption consideration is implemented into GA

Objective function: Maximize (Vp)
Design Variables: h1, h2, index for material front plate, index for material back plate
Constraints: (1) Areal density < 45 kg/m^2
(2) h1/h2 > 1
(3) Total thickness = h1 + h2 < 18mm
(4) Energy absorbed > 700J (assumed)

- A material database containing 31 different ceramics and 56 different aluminiums are employed. (Note at this point GFRP was not added to the database)
- US Ball M33 0.5 in Caliber (mp=46.8g) was tested and results compared with published work.

- Impact velocity assumed is 400m/s

After the above problem definition was integrated into the GA the following results were obtained:

```
The fittest individual so far is 356.14 at gen# 100

The degree of violation of our constraint(s) is 0 percent

Design_Variables =

   0.012714    0.0051429      15       39

elapsed_time =       5.348
```

The results found by the GA indicate that since there is zero degree of constraint violation we can have an impact velocity that exceeds the ballistic limit but still have a reasonable level of energy absorption by the armour. Strictly speaking if the impact velocity greatly exceeds the ballistic limit, penetration will occur. For the above problem definition since a value of 700 J was assumed this constraint restricts the GA to the ballistic limit found.

7.6.3 Discussion on the concept of energy absorption

Note in calculating the energy absorbed, it was assumed that the mass of the projectile (m_p) remains unchanged during the penetration process The residual velocity is the difference between the impact velocity and the ballistic limit velocity [126]. Moreover, Florence model is valid for both AP and ball projectiles. It can be argued that energy absorbed may not be the best constraint when considering the energy absorbing capability of the armour. This argument is supported by the work of Hetherington [128] who claimed that energy absorbed is not a good measure of performance for the armour. This is mainly because only when the impact velocity is close to the ballistic limit will the penetration mechanism be similar to Florence model.

Moreover, the application of GFRP to Florence model was verified. Hence, to the backing database, 6 different types of GFRP were added.

In addition, from the results (not shown) obtained after studying the concept of energy absorption by the armour, it was found with the aid of the GA that minimizing the residual velocity or residual energy has the same effect as maximizing the amount of energy absorbed. This fact can simply be deciphered from Eqn. 26 given above.

The impact velocity should be less than ballistic limit velocity to prevent penetration. Thus the next stage of this study was to add the residual energy as a constraint. Hence, the residual energy was assumed to be a small value, with the intention that such energy would have no fatal effect on the armour carrier. This assumption still has to be verified if not analytically, experimentally or numerically, since if residual velocity were greater than zero this would seem to imply we would have penetration.

Moreover experimental studies would need to be carried out to determine whether the trends depicted in Figure 7.61 to Figure 7.6.5 are specific to the particular threat used or whether they can find wider applications. With the validity of the Florence equation established to a reasonable degree of accuracy, for this specific threat level and selected armour material, the model can be employed to assist the armour designer.

Even though the energy absorbed by the armour has been implemented into the problem as a constraint, further studies are needed to justify this idea. A later study in this section delves further into the concept of energy.

7.7 Looking into Energy absorbed as a proper constraint for GA problem definition

The only way to maximize the amount of energy absorbed is to maximize the ballistic limit. Thus in some sense the constraint on energy absorbed seems repetitive, but ensures armour configurations that satisfy this constraint. In addition, energy absorbed during perforation reduces with increased impact velocity, as is evident from the equation below:

$$\text{Energy absorbed} = \tfrac{1}{2} m_p (2V_i V_p - V_p^{\,2})$$

Note also that the energy absorbed may not be a good constraint since the value at which to constraint it is unknown. Moreover, the conventional way of assessing the ability of an armour to withstand a specified threat is to determine the ballistic limit velocity, however; some research effort has also looked into the effects of maximizing the energy absorbed as a method for optimizing the performance of the armour. In order to check this claim, a few case studies are conducted to check the need of energy absorbed by the armour as a constraint. Residual energy is added as a constraint and is calculated via Eqn. 27.

$$\text{Residual Energy} = \tfrac{1}{2} m_p V_r^2 \qquad (27)$$

Florence and Hetherington's models are employed. A program previously created is used for this case study with the following problem definition:

Objective function: Maximize (Vp)
Design Variables: h1, h2, index for material front plate, index for material back plate
Constraints: (1) Areal density < 45 kg/m^2
(2) h1/h2 > 1
(3) total_thk= h1 + h2 < 18mm
(4) Energy absorbed > 700J
(5) Residual energy < 50J

- A material database containing 31 different ceramics and 67 backing materials comprising of different aluminium and GFRP are employed.
- A 7.62 AP (mp=10g) was tested and results compared with published work.
- GFRP was added to database.
- Impact velocity used is 400m/s

After above problem definition was integrated into the GA the following results were obtained:

```
The fittest individual so far is 499.99 at gen# 100

The degree of violation of our constraint(s) is 0 percent

Design_Variables =

  0.0078571   0.0048571     21      14
elapsed_time =      70.381
```

Having the above results various case studies were conducted to compare the effect of having energy absorbed as a constraint in the problem definition. Hence, using the above problem formulation the following results were obtained after various runs:

Case 1

Table 7.7.1. Remove constraint 5 with everything else the same.

Run#	h1(m)	h2 (m)	mat1	mat2	Vp (m/s)
1	0.010286	0.00357	31	51	541.05
2	0.007857	0.007429	14	52	541.25
3	0.007857	0.004857	2	31	540.87

Case 2

Table 7.7.2. Remove constraint 4 and constraint 5, with everything else the same.

Run#	h1(m)	h2 (m)	mat1	mat2	Vp (m/s)
1	0.012714	0.004857	15	39	1296.81
2	0.012714	0.004857	15	39	1296.81

Case 3

Table 7.7.3. Remove constraint 4 and keep constraint 5, with everything else the same.

Run#	h1(m)	h2 (m)	mat1	mat2	Vp (m/s)
1	0.010286	0.002286	2	48	499.64
2	0.007857	0.004857	21	14	499.99

From various runs conducted with the program it was found that a constraint on residual energy automatically satisfied a constraint on energy absorbed and momentum transfer, or perhaps a constraint on energy absorbed is not needed at all. Moreover, for Case 3 the results are identical to results obtained above when all constraints are present. Hence the use of constraint 4 is not warranted.

As evident from the equations presented, the amount of energy from a bullet is at a maximum when it strikes the target at the ballistic limit velocity. As the residual velocity increases the amount of energy absorbed by the target and the amount of momentum transferred from the bullet to the target reduces.

The 6 different types GFRP added to the material backing database had no effect on the GA choice of material selected (consult the tables above).

7.7.1 Effect of increased impact velocity on armour design

Previous studies have shown the effect of adding and removing the constraints of energy. The results produced by the GA if the impact velocity is increased are now investigated. The following formulation was adopted to carry out this study.

Objective function: Maximize (Vp)
Design Variables: h1, h2, index for material front plate, index for material back plate
Constraints: (1) Areal density < 45 kg/m^2
(2) h1/h2 > 1
(3) total_thk= h1 + h2 < 18mm
(4) Residual energy < 50J

- A material database containing 31 different ceramics and 67 backing materials comprising of different aluminium and GFRP are employed.
- A 7.62 AP (mp=10g) was tested and results compared with published work.
- Impact velocity assumed is 500m/s

From the experimental findings by Hetherington [128] we expect that energy absorbed during perforation to reduce with increased impact velocity. Secondly, an increase in impact velocity results in a dramatic reduction in the fraction of the round's energy. Finally, the greatest reduction in mass and velocity occurs close to the ballistic limit and these reduce with increased impact velocity.

The following GA generated results from the above problem definition is shown in Table 7.7.4.

Table 7.7.4. GA generated results

Run#	h1(m)	h2 (m)	mat1	mat2	Vp (m/s)	CPU time (s)
1	0.010286	0.007429	14	23	599.639	69.81
2	0.010286	0.002286	20	32	599.723	142.43
3	0.010286	0.003571	15	27	599.632	49.792

Note that the results like those previously presented follow a particular trend, that is, the ballistic limit exceeds the impact velocity by approximately 100 m/s. This makes sense since the model assumes no mass erosion and thus the only parameter affecting changes in residual energy is the residual velocity. The residual velocity is simply the difference between the impact and ballistic limit velocity.

For example, the residual kinetic energy = $\frac{1}{2} m_p V_r^2$ = ½ (10e-3)$(V_i - V_p)^2$

$$=1/2*(10e\text{-}3)*(500-599.639)^2 = 49.64J$$

Note also that the thickness of the front plate (h1) is the same for all runs. The reason for this observation is not clear, and one can only find an explanation for such trend by perhaps looking into the constraints imposed.

7.7.1.1 Sensitivity Study

The above observation forced one to test the effect of the constraint regarding residual energy when the impact velocity is fixed. The residual energy constraint was reduced to 10 J and the impact velocity was set at 500 m/s while everything else in the previous problem definition remained the same. The results are shown in Table 7.7.5.

Table 7.7.5. GA generated results after constraint on residual energy is implemented

Run#	h1(m)	h2 (m)	mat1	mat2	Vp (m/s)	CPU time (s)
1	0.007857	0.004857	21	31	543.821	24.926
2	0.010286	0.003571	31	2	544.484	42.531
3	0.007857	0.004857	21	31	543.821	49.842

From Eqn. 27, the results in Table 7.7.5 are not surprising, since in order to achieve a residual energy of 10 J when impact velocity is 500 m/s, requires a reduction in ballistic limit.

The results above surely requires the reiteration of a statement already made in Ref. 128, that is, only when the impact velocity is near to the ballistic limit velocity will the mechanism be similar to that described by Florence. For this reason the amount of energy absorbed during perforation is not a very effective indicator of the potential performance of the armour.

7.7.2 Discussion after investigation into the effects of increase impact velocity on armour design

As we attempt to reduce the residual velocity we simultaneously increase the energy absorbing capabilities of the armour. Studies done by Navarro et al. [129] suggested that a safe value for the residual kinetic energy was 670 J, a typical value used in armour design. 670 J is much higher than what was considered in this work. Nonetheless, a low residual energy value only ensures better performance of the armour and relaxing the constraint to a value of 670 J only enables the GA to provide better results.

7.8 Ben Dor et al. model results

The formulation by Ben Dor et al. is initially used to generate some theoretical results before integration into the GA. The material data previously used, along with the threat and materials from Ref. 128 - which is repeated here for clarity - was used to generate Figure 7.81 to Figure 7.8.4.

Data input to the program:
Mass of projectile (mp) = 46.8g
Radius of projectile (ap) = 12.7 mm. (US Ball M33 0.5in caliber)
Density of ceramic = 3499 kg/m^3
Density of backing (GFRP) = 2229 kg/ m^3
Breaking strain of backing = 0.04
UTS of backing = 1e9 N/m^2

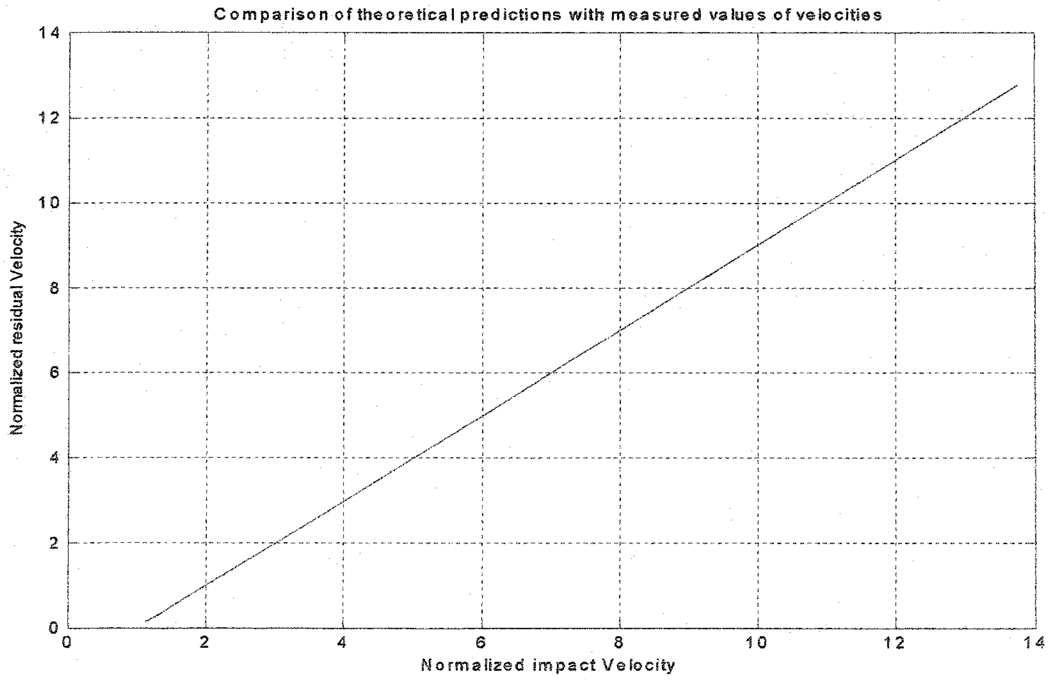


Figure 7.8.1: Comparison of theoretical predictions with experimental data adapted from studies by Hetherington [128]

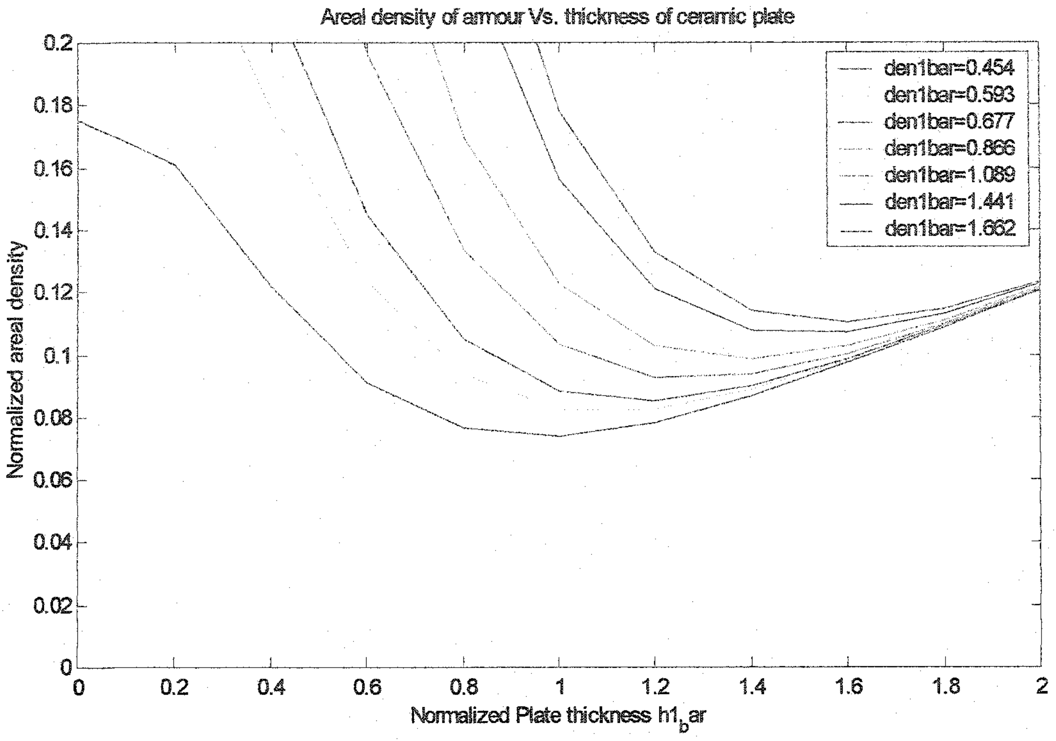


Figure 7.8.2: Areal density of armour vs. thickness of ceramic plate.

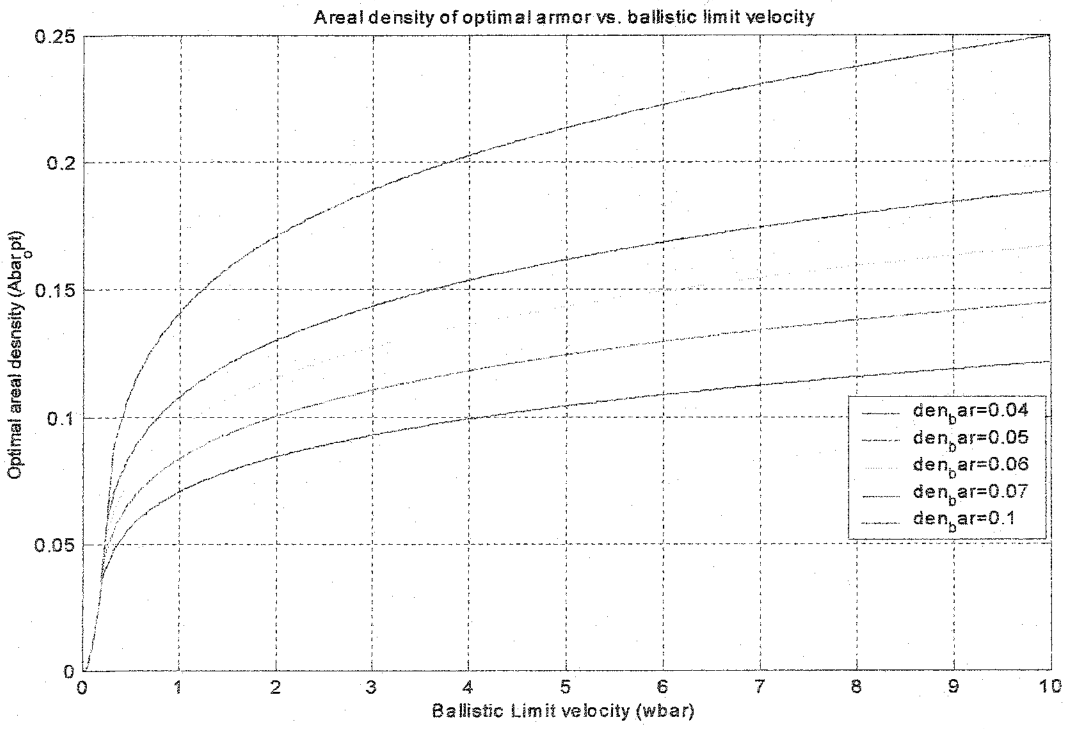


Figure 7.8.3: Areal density of optimal armour vs. ballistic limit velocity

Figure 7.8.3 is slightly off from the published results found in Ref. 44. The reason for the discrepancies is not clear. Nevertheless, the trend of the data is identical, so the results are presented and formulation used.

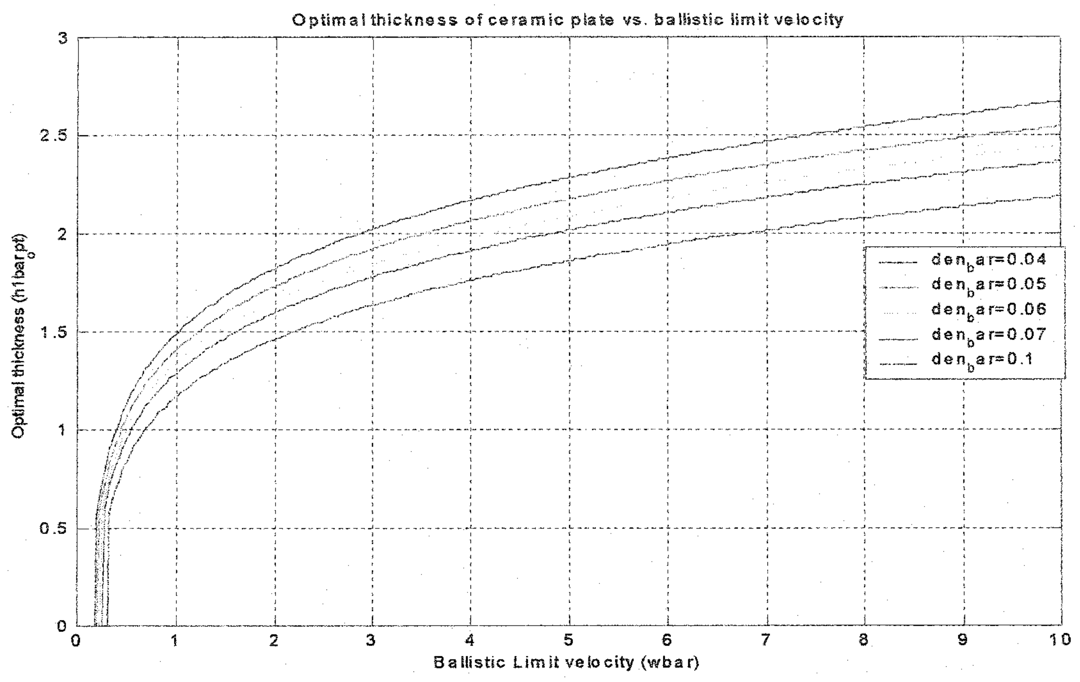


Figure 7.8.4: Optimal thickness of ceramic plate vs. ballistic limit velocity.

Note since the results of Figure 7.8.3 is slightly off, the results in Figure 7.8.4 were also off because of the dependency of Aopt on hopt.

7.9 Ben Dor et al. model results after implementation into GA

Since the trend in the data is similar to published results, the MATLAB formulation of Ben Dor et al. program was integrated into the GA for optimization. Initially, a simple problem definition was used in order to allow for comparison with published results found in Ref. 44. This problem definition entailed the use of only one design variable and one constraint given below:

Objective function: Minimize $(\overline{A})$
Design Variables: $\overline{h}_1$
Constraints: (1) total thickness = h1 + h2 < 20mm

- The front plate is set as ceramic and the backing as GFRP.
- A ballistic limit velocity of 400 m/s is used initially
- A 12.7 mm AP (mp=46.8g) was tested and results compared with published work.
- Alpha was taken as 0.9 (determined from experimental data presented in Ref. 128)

The results from the above formulation are given in Table 7.9.1.

Table 7.9.1. GA generated results

Run#	V_p (m/s)	$\bar{h}_1$	$\bar{A}$	h_1 (m)	h_2 (m)	ρ_1 (kg/m^3)	ρ_1 (kg/m^3)	A (kg/m^2)
1	400	2.4194	0.17232	0.01536	4.54E-03	3499	2229	63.9
2	100	0.87097	0.0629	5.53E-03	1.78E-03	3499	2229	23.3

At $V_p = 400$ m/s, $\bar{w} = 3$ and the optimum thickness $\bar{h}_1^{opt}$ obtained when published results [13] are consulted is approximately 2.4, a value close to that found by the GA. Also when $V_p = 100$ m/s, $\bar{w} = 0.75$ and the optimum thickness $\bar{h}_1^{opt}$ obtained when with published results are consulted is approximately 0.85. Hence it can be said that the formulation and its fusion with the GA works fairly well. Since Ben Dor et al. formulation works well, two new design variables were added to the problem definition, namely, material index for the front and back plate respectively. Important to remember that since the formulation is dimensionless any material combination can be tested or moreover any material can be added to the material databases. In this study, materials are only added to the backing plate database.

Initially the database previously created i.e. front plate database of 31 different types of ceramics and backing plate database of 67 different materials was used along with the following problem definition:

Objective function: Min (Areal density)
Design Variables: h1_bar, mat1, mat2
Constraints: (1) total thickness = h1 + h2 < 20mm
(2) h2_bar > 0

- A ballistic limit velocity of 400 m/s is used initially
- A 12.7 mm AP (mp=46.8g) was tested and results compared with published work.
- Alpha was taken as 1 (Florence model)

Two tests were conducted and the results are given in Table 7.9.2.

Table 7.9.2. GA generated results

Run #	V_p (m/s)	$\bar{h}_1$	mat 1	mat 2	Areal density (kg/m^2)
1	400	2.5806	14	39	35.7796
2	400	2.4516	14	39	35.8036

Using the above formulation a sensitivity study was conducted in order to check the effects of certain constraints.

7.9.1 Sensitivity Study

Using the above problem formulation we replace the constraint on front plate thickness with a constraint on thickness ratio as done by Hetherington; that is h1/h2 > 1. Constraint on total thickness is changed to total thickness < 18mm. The results after two run of the GA is shown in Table 7.9.3.

Table 7.9.3. GA generated results after h1/h2 > 1 is added as a constraint

Run #	Vp (m/s)	h1_bar	mat 1	mat 2	Areal density (kg/m^2)
1	400	2.3226	15	29	42.8483
2	400	2.3226	15	35	41.6713

Results appear similar to results found when using the problem definition used by Hetherington. This may be so because the material database is the same for both programs. Finally, thirteen new materials were added to the database to study their effect on the GA decision-making. The thirteen new materials were among various types of Kevlar, spectra, nomex etc. Although not presented, the latter problem formulation incorporating the thirteen new materials had no effect on the GA choice of material. However, this may simply be due to the way in which the problem was defined. For instance, if the total thickness constraint is changed to say total thickness less than 25 mm, in most cases one of the new materials added was selected by the GA. It is not quite clear why the GA typically selects aluminium over the other materials when Ben Dor et al. formulation is used. Further studies are required to understand the effects of the model formulation on the GA choice of design variables.

7.9.2 Discussion on Ben-Dor et al. model

As mentioned earlier the thirteen new materials added to the backing database had little effect on the GA choice of material selection. The GA selects the new materials added to the database only when the constraint on the total thickness is relaxed. Nonetheless, incorporating the energy absorbed into Ben Dor et al. formulation seem like a futile task since it would require the user to input the value of the impact velocity. Recall that the ballistic limit had to be inputted to the program by the user for Ben Dor et al. formulation, hence immediately the residual velocity is known and is constant if an impact velocity is assumed. Consequently the energy absorbed by the armour and the residual energy is also known and constant. Therefore, when this approach of armour design is taken, it would seem wise to ignore the aspect of energy absorption by the armour.

This seems like the way to go, since Hetherington [41] who attempted to optimize the ballistic limit obtained similar results to Ben Dor et al. Thus it is reasonable to claim that regardless of the approach taken for armour design, one is apt to obtain similar results for similar materials. However, one wonders whether the approach by Hetherington [41] captures the armour problem more completely than Ben Dor approach. This is an entirely different area of research and requires much more indebt study than that provided in this thesis.

7.10 General Discussion and Conclusions

The use of genetic algorithms for armour design is without question after studying the results obtained. Analytical models developed by Florence, Hetherington et al. and Ben Dor et al. were studied, formulated and later integrated into the GA. Viable and practical designs were obtained after the analytical models were fused with the genetic algorithm. The analytical models studied and fused with the genetic algorithm were capable of finding feasible armour designs so as to reduce the number of armour configurations required for numerical and experimental studies. Hence in many ways the objective of this research effort was achieved since it help to reduce the cost of experimentation and the lengthy time required for numerical analysis.

A summary of the best designs found by the GA for a particular threat, and for a given model formulation is summarized in the table below. Florence model does not appear in the Table 7.10.1 since is not studied independently but instead it is fused with other models so as to obtain the ballistic limit.

Table 7.10.1. Summary of Best GA generated designs

Model	Threat Type	mp (g)	dp (mm)	h1(m)	h2(m)	h1/h2	Tot. thickness (m)	mat1	mat2	Vp (m/s)
Hetherington	AP	10	7.62			3.8175	0.015	26	39	1252.26
	AP	42	12.7			3.5159	0.015	26	39	517.221
	FSP	50	20			2.7619	0.015	25	35	459.446
Hetherington with Areal Density considerations	AP	10	7.62	0.02	0.005143			14	39	1771.57
	AP	30	12.7	0.01	0.005143			30	39	414.973
Hetherinton with Residual Energy considerations (Vi=400m/s)	AP	10	7.62	0.01	0.004857			21	14	499.99
Ben-Dor Assumed Vp=400m/s	AP	46.8	12.7	0.01	0.002788			15	35	400
Vp=1000m/s	AP	46.8	12.7	0.02	0.003776			31	58	1000

If Hetherington's approach is used, knowledge of the impact velocity is required and if Ben Dor et al. approach is used knowledge of the ballistic limit velocity is desired. Due to the difference in approach, it is believed that the two methods cannot be combined, but independently, each method should yield similar results given the same material.

From this study one wonders why GA was not used before to aid in design of armour. To the author's knowledge no publications are available that alludes to the use of GA in armour design. However, with the integration of simple analytical models into the GA one can quickly study numerous configurations and materials for the armour with little computing time and at little cost.

7.11 Appendix A

Material database for the armour front plate

Material Index	Density (Kg/m^3)
1	3420
2	3600
3	3700
4	3720
5	3800
6	3900
7	3920
8	3960
9	3970
10	3650
11	3750
12	4400
13	4200
14	1800
15	2300
16	2500
17	4050
18	3820
19	4100
20	3760
21	3670
22	3870
23	3930
24	5650
25	6070
26	6100
27	6000
28	5850
29	2700
30	2800
31	2530

Material database for the armour back plate

Material Index	Density (Kg/m^3)	UTS (Pa)	Breaking strain
1	2700	3.10E+08	0.12
2	2700	2.40E+08	0.22
3	2700	4.60E+08	0.1
4	2710	3.80E+08	0.1
5	2710	3.10E+08	0.08
6	2710	2.62E+08	0.1
7	2710	3.52E+08	0.05
8	2700	1.50E+08	0.2
9	2700	2.40E+08	0.12
10	2700	3.30E+08	0.08
11	2710	3.79E+08	0.05
12	2710	4.13E+08	0.05
13	2710	4.00E+08	0.08
14	2720	3.60E+08	0.18
15	2720	3.95E+08	0.12
16	2710	3.40E+08	0.12
17	2710	2.90E+08	0.24
18	2700	2.20E+08	0.15
19	2710	3.10E+08	0.12
20	2720	4.00E+08	0.1
21	2700	3.10E+08	0.11
22	2710	2.50E+08	0.2
23	2690	2.40E+08	0.12
24	2700	1.70E+08	0.16
25	2700	2.60E+08	0.08
26	2830	3.80E+08	0.15
27	2830	3.95E+08	0.17
28	2830	4.05E+08	0.12
29	2800	4.25E+08	0.2
30	2800	4.70E+08	0.13
31	2810	4.00E+08	0.19
32	2750	3.40E+08	0.24
33	2570	5.25E+08	0.05
34	2590	5.50E+08	0.06
35	2780	4.70E+08	0.2
36	2750	3.00E+08	0.27
37	2780	5.15E+08	0.06
38	2750	4.85E+08	0.18
39	2780	4.70E+08	0.2

Material database for the armour back plate continued

Material Index	Density (Kg/m^3)	UTS (Pa)	Breaking strain
40	2770	4.27E+08	0.05
41	2780	4.85E+08	0.06
42	2770	4.55E+08	0.05
43	2840	4.15E+08	0.1
44	2840	4.55E+08	0.1
45	2840	4.75E+08	0.1
46	2810	4.35E+08	0.1
47	2840	5.80E+08	0.12
48	2780	3.90E+08	0.15
49	2780	3.50E+08	0.13
50	2810	5.70E+08	0.11
51	2730	4.00E+08	0.13
52	2730	4.50E+08	0.13
53	2800	5.50E+08	0.12
54	2800	5.20E+08	0.12
55	2800	5.20E+08	0.12
56	2880	6.59E+08	0.04
57	2600	3.45E+09	0.048
58	2460	4.89E+09	0.057
59	2490	4.59E+09	0.054
60	2700	3.24E+09	0.044
61	2720	4.50E+09	0.048
62	2540	4.14E+09	0.048
63	1440	3.62E+09	0.036
64	1440	3.00E+09	0.024
65	1470	3.45E+09	0.03
66	1780	4.15E+09	0.018
67	1790	4.07E+09	0.018
68	970	2.59E+09	0.035
69	970	3.00E+09	0.027
70	1110	5.50E+07	0.04
71	1170	9.70E+07	0.04
72	1394	1.38E+08	0.04
73	4820	1.19E+09	0.08
74	4820	1.29E+09	0.09
75	1170	4.50E+09	0.016
76	1780	4.15E+09	0.018
77	1790	4.01E+09	0.018

Material database for the armour back plate continued

78	1790	6000e6	0.022
79	1380	607e6	0.305
80	1390	3000e6	0.044
81	1270	2.95e9	0.033
82	1240	3.38e9	0.033
83	887	3.1e9	0.035

Chapter 8

General Discussion, Conclusions and suggestions for future work

Optimization, multidisciplinary design optimization, and soft computing techniques were defined. Initially a global picture of optimization was presented. Some optimization techniques common to mechanical engineering that constitute this global picture were highlighted. A more concise view was then adopted to focus on a particular soft computing technique, namely, genetic algorithms. Detailed information about the genetic algorithms creation and process of implementation to solve real world problems was presented. Finally the genetic algorithm was applied to solve various engineering problems. At first the applications were simple, but were studied so as to ensure correct functionality of the genetic algorithm. Having proved the effectiveness of the genetic algorithm, more complex problems of conceptual aircraft design, structural design and analysis, and finally armour design and analysis, were performed.

The genetic algorithm used in this study was capable of searching for and finding optimal solutions. For instance, genetic algorithms were used to show that GAs are a flexible and efficient means of generating conceptual aircraft designs, that can increase the scope and decrease the time and cost entailed in the traditional design approach. In addition, the genetic algorithm based finite element analysis technique developed was capable of finding lighter structural designs when compared with mathematical, optimality criteria and other heuristic approaches. Genetic algorithms were even applied to armour design. Analytical models developed by Florence, Hetherington et al. and Ben Dor et al. when studied, formulated, and later integrated into the GA, resulted in viable and practical designs. Hence the analytical models fused with the genetic algorithm were capable of finding feasible armour designs. This enabled a reduction in the number of armour configurations required for numerical and experimental studies.

Hence in many ways the objective of this research effort was achieved. The GA also aided in providing initial layouts and geometries of reasonable conceptual aircraft design maps. This initiated a speedy design process providing a quick and easy structural optimization methodology capable of reducing cost and computational effort.

This study suggests that the GA is capable of handling the complex problem of simultaneously managing and integrating many design disciplines. From this standpoint, serious consideration should be paid to designs found in this and other research utilizing GAs. Furthermore, such research acts as a starting point for generating designs that meet the required design objective in a short period of time; a condition that ultimately translates into saving time and money.

In concluding, the genetic algorithm integrative approach proved to be beneficial, as well as practical and feasible for design and optimization. However, for some problems studied the long computational time incurred acted as a deterrent. Hence, the issue of rapidly optimizing a problem was a shortcoming of some applications. This is due to the inherent slow convergence speed of the GA near the global optimum. However, this should not be seen as a limitation in light of the advancements made in improving the processing speed of computers.

It is hoped that the present research effort will prompt further research into the use of artificial intelligence methods for research, but more importantly, to meet industry demands. The following are some suggestions for future work as summarized from the application chapters of this thesis.

- Investigation into other AI methods such as SA, TS and ANN for mechanical engineering problems can greatly add a greater level of flexibility and sophistication in choice of algorithms for the particular problem encountered.
- All mechanical engineering applications studied in this thesis utilized a Pentium III processor. The use of a faster and more efficient computer can significantly improve the performance of the GA especially when large complex problems are studied.
- The use of a local search mathematical based technique to aid the GA search procedure later on in the generation can help curb the computational effort entailed in this study. In other words, a hybrid GA can be employed to assist the search process in finding the absolute global optimum after it had converged on the area in which the absolute global optimum is located. Future research direction can also be oriented towards using GA integrated artificial neural networks (ANN) to reduce the large computational effort required for structural optimization.
- Finally this work opens the door for extending this research effort to other mechanical engineering problems.

References

[1] Holsapple, C.,W., Jacob, V.,S., and Whinston, A.,B., "Operations research and artificial intelligence," 1986.

[2] Carter, M., W., "Operations Research: A Practical Introduction," CRC Press, 2001.

[3] Simon, H.,A., "Two heads are better than one: the coloration between AI and OR," Interfaces, 17,8-15, 1987.

[4] Kanal, L.,N., and Lemmer, J.,F., " Uncertainty in Artificial intelligence," North Holland, Amsterdam,1986.

[5] Pearl, J., "Probabilistic Reasoning in Intelligent Systems," Morgan Haufman, San Mateo, Ca., 1988.

[6] Pham, D.,T., and karaboga,D., " Intelligent Optimization Techniques," Springer, 2000.

[7] Holland, J.H., " Adaptation in Natural and Artificial Systems," University of Michigan Press, Ann Arbor, MI, 1975.

[8] Quagliarella, D., and Cioppa, D.,A., " Genetic algorithms applied to aerodynamic design of transonic airfoils," Journal of aircraft, vol. 32. No. 4, pp. 889-891, 1995.

[9] Perez, E.,R., Behdinan, K., Chung, J., " Airfoil shape optimization using genetic algorithms," Proceedings of 47th. Annual Conference of the Canadian Aeronautics and Space Institute, Aircraft Design and Development Symposium, Ottawa, ON, April 30 – May 3, 2000.

[10] Anderson, B., M., and Gebert, D.," Using Pareto genetic algorithms for preliminary subsonic wing design," Paper 96-4023, AIAA, Sept. 1996.

[11] Marx, J. Schrafe, D.,P., and Mavris, D., N., "Integrated product development for the wing structural design of high speed civil transport," School of Aerospace Engineering, Georgia Institute of Technology, USRAHSCT Work shop, December, 1993.

[12] Ewing M., and Downs, K., " Optimization of rectangular cross-section wing-box using genetic search algorithm, Paper 96-1536, AIAA, 1996.

[13] Tse, D., and Chan, L., " Transonic Airfoil Design optimization using Soft-Computing Methods, Proceeding 47th. Annual Conference of the Canadian Aeronautics and Space Institute, Aircraft design and Development Symposium, Ottawa, ON, April 30 – May 3m 2000.

[14] Butler, R., Hansson, E., Lillico, M., and Dalen, F., "Comparison Multidisciplinary design and optimization codes for Conceptual and Preliminary wing design, "Journal of Aircraft, Vol. 36, No. 6, AIAA.

[15] Gage, P.,J., Kroo, I.,M., and Sobieski, I., P., " Variable-Complexity Genetic algorithm for topological design," AIAA Journal, Vol. 33, No. 11, November 1995.

[16] Tong, S., S., Powell, D., and Skolnick, M., "EnGENEous: Domain Independent, Machine learning for Design Optimization," Proceedings of the 3rd International Conference on Genetic algorithms, M. Kaufmann Publishers, San Mateo, CA, 1989, pp. 151-159.

[17] Mosetti, G., and Polini, C., "Aerodynamic Shape Optimization by means of genetic algorithms," Proceedings of the 5th International Symposium on Computational Fluid Dynamics, Vol. 2, Japan Society of Computational Fluid Dynamics, Tokyo, Japan, 1993, pp. 279-284.

[18] Hajela, P., "Genetic Search – An approach to the Nonconvex Optimization Problem," AIAA Journal, Vol. 28, No. 7, 1990, pp. 1205-1210.

[19] Crossley, A.W., Laananen, H.,D., " Conceptual Design of Helicopters via Genetic Algorithm," Journal of Aircraft, Vol. 3, No. 6, Nov.-Dec. 1996.

[20] Sobieski, J., Barthelemey, J.,F., M., and Giles, G.,L., "Aerospace Engineering Design by Systematic decomposition and multilevel optimization," Proceeding of the 14th Congress of International Council of Aeronautical science, Paper 84-4.7.3., AIAA, Washington DC, 1984, pp. 828-840.

[21] Gallman, W., R., Kaul, R., W., Chandrasekharan, R.,M., and Hinson, M.,C., "Optimization of Advance Business Jet," Proceedings of the 5th AIAA/USAF/NASA/ISSMO symposium on multidisciplinary analysis and optimization, AIAA 94-4303, Panama City Beach, FL., Sept. 1994.

[22] Malone, B. and Mason, W., H., " Aircraft Conceptual Optimization using the Global sensitivity approach and parametric multi-objective figures of merit," Proceeding, AIAA Aircraft design systems meeting, Hilton Head, SC, AIAA 92-4221, August 24-2, 1992.

[23] Crispin, Y., "Aircraft Conceptual optimization using simulated evolution," AIAA, Paper 94-0092, January 1994.

[24] Bos, D., "Multidisciplinary design Optimization of a supersonic transport aircraft using Hybrid Genetic/Gradient Based algorithm," Proceeding of the 6th AIAA/NASA/ISSMO symposium on Multidisciplinary Analysis and Optimization, AIAA 96-4055-CP, Bellevue, WA, Sept. 4-6, pp. 692-706, 1996.

[25] Goldberg, D., E., and Samtani, M., P., " Engineering Optimization via the genetic algorithm," 9th Conference on Electronic Computation, New York, ASCE, pp. 471-482, 1986.

[26] Jenkins, W., M., "Towards structural Optimization via the genetic algorithm," Computers and Structures, Vol. 40. No. 5, pp. 1321-1327, 1991.

[27] Goldberg, D.,E., " Genetic Algorithms in search, optimization and machine learning, Reading, MA, Addison-Wesley, 1989.

[28] Hajela, P., and Shih, C.J., "Multiobjective optimum design in mixed integer and discrete design variable problems," AIAA Journal, Vol. 118, No. 4., pp. 670-675, 1990.

[29] Rajeev, S., and Krishnamoorty, C.S., "Discrete optimization of structures using genetic algorithms," Journal of Structural Engineering, Vol. 118, No. 5, pp. 1233-1250, 1992.

[30] Rajan, S., D., "Sizing, Shape, and Topology design optimization of trusses using genetic algorithms," Journal of Structural Engineering, Vol. 121, No. 10, pp. 1480-1487, 1995.

[31] Deb, K., and Gulati, S., "design of truss-structures for minimum weight using genetic algorithms," Finite Elements in Analysis and Design, Vol. 37, pp. 447-465, 2001.

[32] Camp, C., Shahram, P., and Guozhong, C., "Optimized design of Two-Dimensional Structures using genetic algorithm," Journal of Structural Engineering, Vol. 124, No. 5., pp. 551-559, 1998.

[33] Florence, A.L., "Interaction of projectiles and composite armour, Internal Report, August, 1969.

[34] Wilkins, M. L., Honodel, C., and Sawle, D., " An approach to the study of light armour," Lawrence Radiation Laboratory, University of California, Livermore, CA, U.S.A., UCRL-50284, 1967.

[35] Wilkins, M. L., " Second Progress report on Light armour Program," Lawrence Radiation Laboratory, University of California, Livermore, CA, U.S.A., UCRL-50349, 1967.

[36] Wilkins, M. L., "Third Progress on light armour Program, Lawrence Radiation Laboratory, University of California, Livermore, CA, U.S.A., UCRL-50460, 1968.

[37] Wilkins, M. L., " Mechanics of penetration and perforation," Int.. J. Engng. Sci. Volo. 16, pp. 793-807, 1978.

[38] Prior, A.M., "The ballistic impact of small calibre ammunition on ceramic composite armour," Ph.D. thesis, RMCS, Shrivenham, July 1988.

[39]Gagne, M.P., "The penetration mechanics of small arms projectiles in ceramic-faced vehicle armours," No. 16, MVT Course, RMCS, Shrivenham, U.K., 1989.

[40] Rajagopalan, B.P., "The experimental validation of an analytical model for use in composite armour design," No. 17 MVT Course, RMCS, Shrivenham, U.K., 1989.

[41] Hetherinton, J.G., "The optimization of two-component composite armors," Int. J. Impact Engng., Vol. 12, pp. 409-414, 1992

[42] Woodward, R. L., "A basis for modelling ceramic composite armour defeat," material research laboratory, DSTO, Ascot Vale, Victoria, Australia, MRL-RR-3-89, 1989.

[43] Den Reijer, P.C., "Impact on ceramic faced armour, Ph.D. thesis, Delft Univ. Tech., Delft, The Netherlands, 1991.

[44] Ben-Dor, G., Dubinsky, A., Elperin, T., and Frage, N., " Optimisation of two-component ceramic armor for a given impact velocity," Theorectical and Applied Fracture mechanics, Vol. 33, pp. 185-190, 2000.

[45] Coello C.A. and Christiansen A.D., "Multiobjective optimization of trusses using genetic algorithms," Journal of computer and structures, Vol. 75, p. 647-660, 2000.

[46] Manoharan, S., and Shanmuganatahn, S., " A comparison of search mechanisms for structural optimization," Computers and Structures, Vol. 73, pp. 363-372, 1999.

[47] Walsh., G.,R., " Methods of Optimization," John Wiley & Sons, New York,1975.

[48] Arora., J.S., "Introduction to Optimum Design," McGraw Hill, New York, 1989.

[49] Adby, P.R., Dempster, M.A.H., "Introduction to Optimization Methods," John Wiley and Sons, New York, 1974.

[50] Hillier F., S., and Lieberman G., J., " Introduction to Operations Research," 6th. Edition, McGraw-Hill Inc., New York, 1995.

[51] Winston, L. W., " Operations research: applications and algorithms," Duxbury Press,3rd. ed., 1993.

[52] Vlach J. and Singhal K., "Computer methods for circuit analysis and design," 2nd. Ed., Kluwer Academic Publishers, 1993.

[53] Cheney, E.W. and Kincaid, D., "Numerical Mathematics and Computing," Brooks/Cole, Monterey, CA, 1980.

[54] Vidal, R.,V.,V., " Applied Simulated Annealing," New York: Springer-Verlag, 1993.

[55] Glover, F., "Tabu Search," Part I, ORSA Journal on Computing,1, 190-206,1989.

[56] Hansen, P., " The steepest ascent mildest descent heuristic for combinatorial programming, Conf. On Numerical Methods in Combinatorial Optimization," Capri, Italy, 1986.

[57] Glover, F., "Tabu search-Part II," ORSA Journal on Computing, Vol. 2, No. 1, p. 4-32, 1990

[58] Hajela P., "Nongradient Methods in Multidisciplinary Design Optimization-Status and Potential," AIAA Journal, NO1, Feb.-Jan. 1999.

[59] Baase, S. and Gelder, A. van, " Computer Algorithms: Introduction to Design and Analysis, 3rd ed., Addison Wesley, Reading, MA, 2000.

[60] Manber, U., " Introduction to Algorithms: A Creative Approach," Addison Wesley, Reading, MA, 1989.

[61] Hein,J.L., " Discrete Structures, Logic, and Computability," Jones and Barlett, Boston, 1995.

[62] Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P., "Optimization by Simulated annealing," Science, 220, 671-680, 12 May 1983.

[63] Metropolis,N., Rosenbluth, A., Rosenbluth M., Teller, A., and Teller, E., Journal of Chemistry and Physics., 21,1087,1953.

[64] Haupt L. R. and Haupt E. S. – Practical genetic algorithms, New York, Wiley, 1998.

[65] Michealwicz Z. Genetic algorithms + data structures = Evolution, Second Edition, Springer-Verlag, Berlin, 1994.

[66] Coley A.D., An introduction to genetic algorithms for Scientist and Engineers, River Edge, New Jersey, World Scientific, 1999.

[67] Kohonen, T., "Self Organization and Associative Memory,"3rd Edition, Springer-Verlag, Berlin, 1989

[68] Hecht-Nielsen, R., " Neurocomputing," Addison-Wesley, Reading, MA, 1990.

[69] Karray, F., "Soft computing techniques for intelligent machines," Pattern analysis and Machine Intelligence Laboratory, CRC Press, 2000, 147-194.

[70] Zadeh, L., "Fuzzy Sets," Information and Control, Vol. 8, pp. 338-356, 1965.

[71] Alexandrov, N., and Hussaini, M.Y., "Multidisciplinary design optimization: State of the art," Virginia, 1995.

[72] Weiss, M.A., "Efficient C Programming: A Practical Approach," Prentice Hall, 1995

[73] MATLAB's User Guide, The Mathworks Inc., 3 Apple Hill Drive, Natick, MA, 1999.

[74] ANSYS V5.6 User Manual, ANSYS Inc., Southpointe, 275 Technology Drive, Canonsburg, PA 15317, USA.

[75] MSC. Nastran, MSC.Software Corporation, NYSE:MNS

[76] PAM-OPT, PAM System International, PSI, the software Company of ESI Group

[77] Filha, J.L/R., Treleaven, P.C., and Alippi, C., " Genetic algorithm programming environments," Computer, pp. 28-43, June 1994.

[78] Holland J. H., Adaptation in Natural and Artificial Systems, MIT Press, Cambridge, MA, 1992

[79] Gen M. and Cheng R., Genetic Algorithms & Engineering Design, Whiley & Sons, New York, 1997.

[80] Harik G. and et al. – The Gambler's Ruin Problem, Genetic Algorithms and the Sizing of Populations, IEEE International Conference, April 1997.

[81] Coit D.W. and Smith A.E., Penalty Guided Genetic search for Reliability Design Optimization, Computers In Engineering Journal, Vol. 30, No. 4, 1996, pp. 895-904.

[82] Jenkins, W., M., "On the application of natural algorithms to structural design optimization," Engineering Structures, Vol. 19, No. 4, pp. 302-308, 1997.

[83] Erbatur, F., Hasancebi, O., Tutuncu, I., and Kilic, H., "Optimal design of planar and space structures with genetic algorithms," Computers and Structures, Vol. 75, pp. 209-224, 2000.

[84] Perez, E.,R., Behdinan, K., Chung, J., " Airfoil shape optimization using genetic algorithms," Proceedings of 47th. Annual Conference of the Canadian Aeronautics and Space Institute, Aircraft Design and Development Symposium, Ottawa, ON, April 30 – May 3, 2000.

[85] Raymer D. P., "Aircraft Design: A conceptual Approach," American Institute of Aeronautics and Astronautics, Washington DC, third ed., 1999.

[86] Torenbeek E., "Synthesis of Subsonic Airplane Design," Delft University Press and Kluwer Academic Publishers, 1990.

[87] Roshkam J., "Airplane Design," Volumes 1-8, DARC Corporation, Ottawa, Kansas, 1998.

[88] Taylor, J., "Jane's All the World Aircraft," London, UK, Jane's Yearbook, 1997 and 2000.

[89] McCormick B. W., "Aerodynamics Aeronautics and Flight Mechanics," John Wiley and Sons Inc., Second Edition, 1995

[90] Pamadi B. N., "Performance, Stability, Dynamics, and Control of Airplanes," American Institute of Aeronautics and Astronautics Inc., Reston, Virginia, 1998

[91] Stinton D., "The Design of the Aeroplane," Blackwell Science Ltd., Great Britain, 1991

[92] Vanderplaats, G., N., " Numerical Optimization Techniques for Engineering Design," McGraw Hill Inc., pp. 250-282, 1984.

[93] Salajegheh, E., and Vanderplaats, G. N., "A New Approximation Method for Stress Constraints in Structural Synthesis," AIAA Journal, Vol. 27, No. 3, 1989, pp. 352-358.

[94] Hansen, S., R., and Vanderplaats, G., N., " Approximation Method for Configuration Optimization of Trusses," AIAA Journal, Vol. 28, No. 1, 1990, pp. 161-168.

[95] Schmit, L., A., and Miura, H., "Approximation Concepts for Efficient Structural Synthesis," NASA CR-2552, 1975.

[96] Imai, K., "Configuration Optimization of Trusses by the Multiplier Method," Mechanics and Structures Department, School of Engineering and Applied Science, University of California, Los Angeles, Rpt. No. UCLA-Eng-7842, 1978.

[97]Belegundu, A., D., "Study of mathematical programming methods for structural optimization," PhD thesis, University of Iowa, Iowa City, Iowa.

[98]Qian , L., X., Zhang, W., X., Siu, Y., k., and Zhang, J., T., "Efficient Optimum design of structures – program DDDU," Computer Methods Applied Mechanics and Engineering, Vol. 30, pp. 209-224, 1982

[99]Hall, S., K., Cameron, G., E., and Grierson, D., E., "Least Weight design of steel frameworks accounting for PA effects," Journal of structural Engineering, ASCE, Vol. 115, No. 6, pp. 1463-1475, 1989.

[100]Erbatur, F., and Al-Hussainy, M., M., "Optimum design of frames," Computer and structures, 45(5/6), 887-891, 1992.

[101]Gellatly, R., A., and Berke, L., "Optimal structural design," Rep. No. AFFDL-TR-70-165, Air Force Flight Dynamics Laboratory, Wright-Patterson Air force Base, Ohio, 1971.

[102]Venkayya, V., B., "Design of Optimum structures," International Journal of Computers and Structures, 1, pp. 265-309, 1971.

[103]Allwood, R., J., and Chung, Y., S., "Minimum-weight design of trusses by an optimality criteria method," International Journal of Numerical Methods in Engineering, Vol. 20, pp. 697-713, 1984.

[104]Kirsch, U., "Feasibility and Optimality in Structural design," Computer and Structures, Vol. 41, No. 6, pp. 1349-1356, 1991.

[105]Chang, K., J., "Optimality criteria methods using K-S functions," Structure Optimization, Vol. 4, pp. 213-217, 1992.

[106]Rozvany, G., I., N., and Zhou, M., " Continuum based optimal criteria (COC) methods: an introduction in optimization of large structural systems," Proc., NATO/DFG ASI, Berchtesgaden, Kluwer, Dordrecht, The Netherlands, pp. 1-26, 1993.

[107] Hajela, P., and Berke, L., "Neurobiological computational models in structural analysis and design," Computer and Structures, Vol. 41, No. 4. pp. 657-667, 1991.

[108] Adeli, H., and Park, H., S., " Hybrid CPN-Neural Dynamics models for discrete optimization of steel structures," Microcomputers in Civil Eng., Vol. 11, pp. 355-366, 1996.

[109] Adeli, H., and Park, H.,S., " Neurocomputing for Design Automation," CRC Press, Boca Roca, Florida, 1998.

[110]"Manual of steel construction – allowable stress design," American Institute of Steel Construction, Chicago, III, 1989.

[111]Ali, N., and Behdinan, k., "Genetic Algorithm Optimized Conceptual design of Aircraft with Constraints in Stability and Control," 48th Annual Conference of the Canadian Aeronautics and Space Institute, Toronto, Canada, pp. 215-225, 2001.

[112]Ali, N., and Behdinan, K., "Conceptual Geometry Optimization of Aircraft using Artificial Intelligence Techniques," International Conference on Multidisciplinary Design in Engineering, Concordia University, Montreal, Canada, 2001.

[113]Ali, N. and Behdinan K., " Stability and Control in Aircraft Conceptual Design using Genetic Algorithms," 1st AIAA Aircraft Technology, Integration and Operations Forum, California, Los Angeles, 2001.

[114]Felix, J., E., "Shape Optimization of Trusses Subject to Strength, Displacement, and Frequency Constraints," Masters Thesis, Naval Postgraduate School, 1981.

[115]Vanderplaats, G.,N., "Design of Structures for Optimum Geometry," Proceedings of the Third Brazilian Congress of Mechanical Engineering, Rio do Janeiro, Brazil, 1975.

[116]Schmit, L., A., and Farshi, B., "Some Approximation concepts for structural synthesis," AIAA Journal, Vol. 12, pp. 231-233, 1974.

[117] Renwei, X., and Peng, L., "Structural optimization based on second order approximations of functions and dual theory," Computer Methods in Applied Mechanics and Engineering, Vol. 65, pp. 101-104, 1987

[118] Xicheng, W., and Guixu, M., "A parallel iterative algorithm for structural optimization," Computer Methods in Applied Mechanics and Engineering," Vol. 96, pp. 25-32, 1992.

[119]Vanderplaats, G., N., and Moses, F., "Automated Design of Trusses for Optimum Geometry," Journal of Structural Division, No. ST3, pp. 671-690, 1972.

[120] Hallquist, J. O., LS-DYNA Theoretical Manual, Livermore: Livermore Software Technology Corporation, 1998

[121]Chocron Benloulo, I.S., and Sanchez-Galvez, " A new analytical model to simulate impact onto ceramic/composite armors," Int. J. Impact Engng., Vol. 21, No. 6, pp. 461-471, 1998.

[122]Laible, R.C., Ceramic composite armour, in R.C Laible (ed.), "Ballistic materials and penetration mechanics, Elsevier, Amsterdam, pp. 135-142, 1980.

[123] Wilkins, M.L., Computer simulation of penetration phenomena, in R.C. Laible (ed.), "Ballistic materials and penetration mechanics," Elsevier, Amsterdam, pp. 225-252, 1980.

[124] Wang, B., Lu, G., and Lim, M.K., " Experimental and numerical analysis of response of aluminium oxide tiles under impact loading," J. mater. Process. Technol., Vol. 51, pp. 321-345, 1995.

[125] Wang, B., Lu, G., " On the optimization of two-component plates against ballistic impact," J. mater. Process. Technol. Vol. 57, pp. 141-145, 1996.

[126] Hetherington, J.G. and Rajagopalan, B.P., " An investigation into the energy absorbed during ballistic perforation of composite armours," Int. J. Impact Engng., Vol. 11, No. 1, pp. 33-40, 1991.

[127]Hetherington, J.G., "Correspondence in "An investigation into energy absorbed during ballistic perforation of composite armours," Int. J. Impact Engng. Vol. 12, pp. 325-327, 1992.

[128] Hetherington, J.G., "Energy and momentum changes during ballistic perforation," Int. J. Impact Engng., Vol. 18., No. 3., pp. 319-337, 1996.

[129]Navarro, C., Martinez, M.A.,, Cortes, R., and Sanchez-Galvez, V., "Some observations on the normal impact on ceramic faced armours backed by composite plates," Int. J. Impact Engng. , Vol. 13, No. 1, pp. 145-156, 1993.