Genetic Algorithms for Composite Laminate Design and Optimization

Grant A. E. Soremekun

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
IN
ENGINEERING MECHANICS

Approved:
Zafer Gürdal, Chair
Layne T. Watson
Surot Thangjitham

February 5, 1997
Blacksburg, Virginia 24061–0219

Keywords: composite laminate, Genetic Algorithm, buckling,
stacking sequence, design, optimization

Genetic Algorithms for Composite Laminate Design and Optimization

Grant Soremekun

(Abstract)

Genetic algorithms are well known for being expensive optimization tools, especially if the cost for the analysis of each individual design is high. In the past few years, significant effort has been put forth in addressing the high computational cost GAs. The research conducted in the first part of this thesis continues this effort by implementing new multiple elitist and variable elitist selection schemes for the creation of successive populations in the genetic search process. The new selection schemes allow the GA to take advantage of a greater amount of important genetic information that may be contained in the parent designs, information that is not utilized when using a traditional elitist method selection scheme. By varying the amount of information that may be passed to successive generations from the parent population, the explorative and exploitative characteristics of the GA can be adjusted throughout the genetic search also. The new schemes provided slight reductions in the computational cost of the GA and produced many designs with good fitness' in the final population, while maintaining a high level of reliability.

Genetic algorithms can be easily adapted to many different optimization problems also. This capability is demonstrated by modifying the basic GA, which utilizes a single chromosome string, to include a second string so that composite laminates comprised of multiple materials can be studied with greater efficiently. By using two strings, only minor adjustments to the basic GA were required. The modified GA was used to simultaneously minimize the cost and weight of a simply supported composite plate under different combinations of axial loading. Two materials were used, with one significantly stronger, but more expensive than the other. The optimization formulation was implemented by using convex combinations of cost and weight objective functions into a single value for laminate fitness, and thus required no additional modifications to the GA. To obtain a Pareto-optimal set of designs, the influence of cost and weight on the overall fitness of a laminate configuration was adjusted from one extreme to the other by adjusting the scale factors accordingly. The modified GA provided a simple yet reliable means of designing high performance composite laminates at costs lower than laminates comprised of one material.

Acknowledgments

I will always be greatful for the unlimited guidance, understanding, and kindness that was given to me by my advisor, Dr. Zafer Gürdal. Without this constant positive reinforcement it would have been impossible for me to complete this work. I would also like to express gratitude to Dr. Layne Watson, who provided a seemingly endless amount of help and advice in the research and preparation of this work. I also thank Dr. Surot Thangjitham for kindly serving on my advisory committee. Finally, I would like to acknowledge the financial support offered by the Boeing Defense and Space Group under project number HU1147, and by the Air Force Office of Scientific Research under grant F49620–96–1–0104.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Genetic algorithms (GAs) have been gaining substantial attention in recent years. GAs are well known for their robustness and ability to search complex and noisy search spaces, phenomena which are frequently encountered in design and optimization problems. Genetic algorithms can be regarded as expensive optimization tools also, sometimes requiring tens of thousands of analyses to achieve convergence. However, there is a large amount of research work being done with GAs and it is continuing to grow, with many new ideas aimed at reducing computational cost.

The diversity of applications utilizing GAs in search and optimization is also quite impressive. Fields of study range anywhere from biology and the social sciences to computer science and engineering applications. A few examples include Alliot *et al.* [1] who used GAs for solving conflicts in air traffic control. Davis [2] and Giffler [3] applied genetic algorithms to problems involving job shop and production scheduling, respectively. These, along with countless other researchers have found GAs to be a valuable tool in problem solving and optimization environments.

Composite materials are becoming the material of choice in structural applications today and for the future. Although the high strength-to-weight properties of composite materials are attractive, their greatest advantage is that they provide the designer with the ability to tailor the directional strength and stiffnesses of a material to a given loading environment of the structure [4]. For laminated composite structures, which are studied in this work, each laminae has its greatest stiffness and strength properties along the direction that the fibers are oriented in, a more detailed description is given in Section 1.3. By orienting each layer at different angles, the structure can be designed specifically for its loading environment. Thus, performance capabilities are usually not constant throughout a composite structure, with areas exposed to high loading conditions possessing the greatest stiffness and strength properties.

On the negative side, composite structures are well known for being expensive to fabricate and, until recently, little was known about fatigue life or repair and maintenance procedures either. Furthermore, a composite structure may run into problems if placed under a load in an unexpected direction in which the strength is low, yielding the potential for structural failure. Such problems are easier to avoid by using isotropic materials such as aluminum or steel, which have uniform material properties in every direction. Designers have been reluctant to use composite materials in aerospace applications where there is a high demand for strong, yet light weight structures, because it is difficult to absolutely determine and design for the range of all possible loading conditions that a structure will encounter. An example of this is the latest and most technically advanced commercial transport aircraft, the Boeing 777. The structure of this revolutionary aircraft, which first flew in 1994, is only comprised of 9% by weight of composite materials [5].

However, methods for designing, manufacturing, and maintaining composite structures are vastly improving. An example of this is the Army's newest attack helicopter, the RAH–66 Comanche, built by Sikorsky Aircraft and the Boeing Co. Due to arrive in the year 2003, the Comanche will be the first helicopter to be constructed almost entirely of composite materials. By utilizing advanced optimization and manufacturing techniques, the total cost for constructing the Comanche is comparable to, if not better than an equivalent all metal structure. Some studies have shown that a 20%–30% savings in weight was gained by using an all composite structure instead of an equivalent metal structure also. Thus, the future for composite materials in aerospace applications looks very bright.

The advantages of using composite materials are somewhat offset by the more complex structural analysis that is required. Composite structures usually involve large, non-convex, integer programming problems that are discrete in nature. Genetic algorithms are ideal optimization algorithms for these type of problems and are simple to implement when compared to other optimization techniques, allowing for their application towards a wide array of problems in this area of study. However, if the cost to analyze each structure is high then a designer may be forced to use other optimization schemes which require fewer function evaluations then GAs.

The purpose of this work is to implement some new ideas aimed at improving the efficiency of GAs in composite structure design, and to explore other areas in their application toward composite structure design and optimization. The remainder of this chapter will provide some general information on genetic algorithms and some of the previous efforts undertaken to reduce computational cost. A comparison between GAs and other optimization techniques will be discussed. Next, an introduction into composite structure design will be given. Finally, a look into composite structure optimization using GAs and a description of what the remainder of this work entails will be presented.

1.1 Genetic Algorithms

The idea of a genetic algorithm was thought to have been conceived by John Holland at the University of Michigan in the 1970s. Holland [6] was interested in applying the laws of natural selection towards the development of artificial systems rather than systems that are based on some reasoning process [7]. These artificial systems could be constructed using computer software and applied to various disciplines which emphasize design, optimization and machine learning.

1.1.1 Basic Structure of Genetic Algorithms

GAs are probabilistic algorithms that utilize the processes of natural selection by mimicking the concept of survival of the fittest. The main element of a GA is the organism which usually consists of a fixed number of chromosomes. In turn, each chromosome may consist of one or many genes. Typically, each gene of a chromosome is coded using a binary alphabet, showing whether a gene is active (represented by a 1) or inactive (represented by a 0). Other representation have used gene alphabets with many more elements or multiple gene alphabets for different types of genes. The complexity of an organism can be controlled by the length and number of chromosome and gene strings, and the size and number of gene alphabets.

A genetic algorithm is usually made up of a group of organisms commonly referred to as a sub-population or population of organisms. If more than one group of organisms exist, then each group is called a sub-population. A group of sub-populations is called a population. Such terminology is often used when discussing parallel genetic algorithms. A parallel GA invokes several sub-populations at once (usually implemented on a parallel computer) and allows each one to migrate towards an optimal solution. In some methods, sub-populations are allowed to interact with one another to improve the performance GA.

If only one group of organisms exists, then the terms sub-population and population are synonymous. The research conducted in this work deals with a single group of organisms only and from here on will be referred to as a population. A schematic of a typical GA structure is given in Figure 1.1.

1.1.2 GA Implementation

The first population of organisms is initialized using some type of randomized process and is termed the first generation of the search. Each organism is then placed into a common environment where it competes and breeds with other members of the population. The characteristics of an organism are provided in the gene strings of each chromosome, the most important of which is fitness. An organism's fitness shows how well it has adapted to its environment. In many GA applications, the environment is more commonly referred to as the design space or the set of all possible choices that exist for a given problem. The GA's task is to locate the area(s) in the design space which will give the best solution to the problem.

The fittest organisms in the population are given the best opportunity to become parents of a child and may survive into the next generation. Breeding is accomplished through a recombination operator, commonly referred to as crossover, and is given a certain probability of being implemented. The main objective of crossover is to take good characteristics from organisms in the parent population and create child organisms which will hopefully be better suited to their environment than their predecessors. Two of the most popular implementations of crossover are one-point and two-point crossover where the chromosome string of each parent organism is randomly split at one or two points, respectively. Pieces from each parent organism are then recombined to create

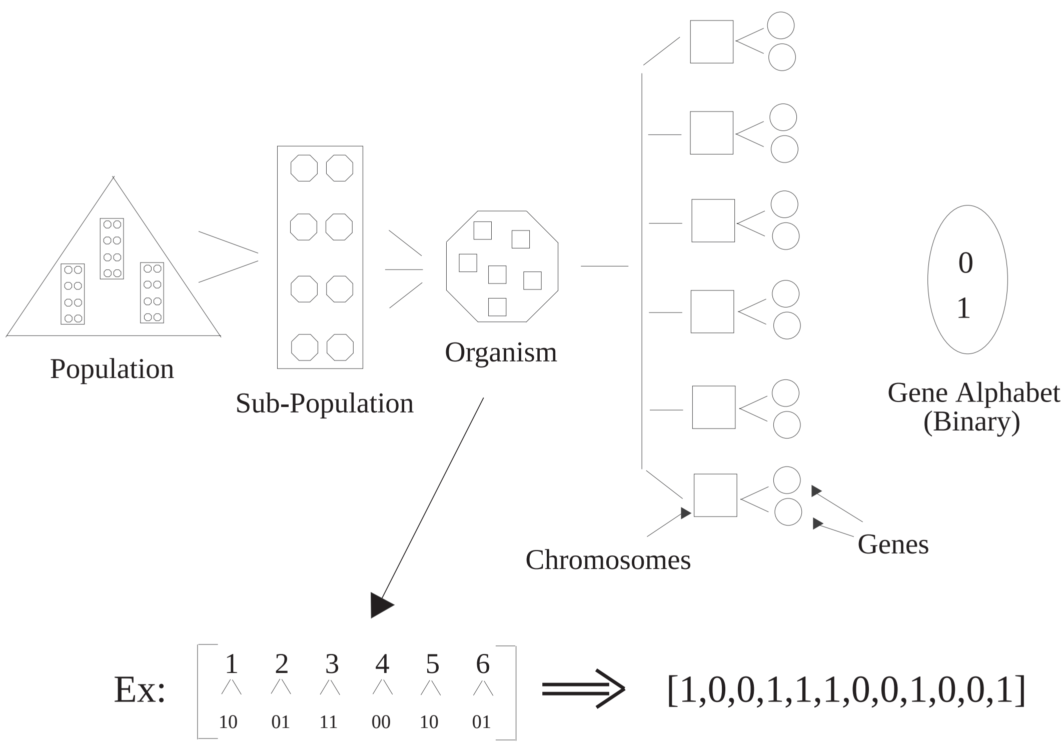


Figure 1.1: A typical GA structure.

a child. Many different types of crossover have been implemented as seen in Le Riche [8] who experimented with no fewer than seven derivatives of this operator.

One-point and two-point crossover are effective recombination operators when the gene alphabet consists of integer numbers but may not be very productive when using real values. In this case, the crossover operator will be restricted to the set of real numbers given in the initial population of organisms. A mutation operator, which will be described momentarily, is the only way to introduce new real numbers. This may be unsatisfactory if the range of real numbers that can be used is large. Another way of handling real numbers is with an averaging crossover operator. Similar to the other crossover methods, a child is partially comprised of chromosomes from both parent strings. The remaining chromosome(s) in the child string consist of genes that contain averaged information from genes in the corresponding parent chromosomes. This type of operator also works well with gene strings that contain both real and integer values [9].

Once children have been created, they may be exposed to a mutation operator that allows for the introduction of new, random information that may aid the algorithm in creating stronger organisms. New information is taken from the gene pool which consists of an allowable range of real numbers and/or set of integer values. Mutation can be implemented with different types of probability. First, there can be a probability of mutating one of the genes in the string to another number located in the alphabet. However, if the length of the gene string is large it may be more desirable to give each gene a small probability of mutating on its own. There also may be different probabilities of mutation for each type of gene contained in a chromosome.

Although crossover and mutation are the most common genetic operators, many others have been developed to try to increase a genetic algorithm's search power. These include gene-swap, permutation and inversion, and repair. Gene-swap is the simplest operator to implement and works by randomly choosing two genes in a string and switching their positions. Permutation works by inverting the genes between two randomly chosen points without changing the meaning of each position on the gene string. The inversion operator works in the same manner as permutation but keeps track of the position of each gene at all times. Inversion is typically used to prevent genes in the string that are physically far apart from one another to be unaffected by crossover [10]. When dealing with constrained optimization problems, repair operators are sometimes used to guide the GA from unfeasible to feasible areas of the design space. Repair operators have been found to be most effective when implemented with a small probability [11] to prevent the GA from getting stuck in one area of the design space.

The genetic algorithm used in this work is armed with 1-point crossover, mutation, and gene-swap operators. An in-depth discussion of applying these operators towards stacking sequence design of composite laminates is given in the next chapter. Schematics of the permutation/inversion and the repair operators are given in Figure 1.2 and Figure 1.3, respectively.

1.1.3 Improving the Efficiency and Reliability of Genetic Algorithms

Many recent studies have concentrated on improving the genetic algorithm's reliability and efficiency. Le Riche and Haftka [12] studied the problem of composite panel weight minimization subject to buckling and strength constraints. By using a combination of penalty parameters, convergence to feasible designs was guaranteed while simultaneously increasing the efficiency of the genetic search. With these modifications and the utilization of a less disruptive permutation operator, a 56% reduction in the price of the search was obtained.

Kogiso et al. implemented two different methods for trying to reduce the cost of GA optimization. In the first method a binary tree was set up to store pertinent information about laminate

Coded gene string	[1,0,0,**1,1,1,0,0,1**,0,0,1]
Position in gene string	1 2 3 **4 5 6 7 8 9** 10 11 12

Before Permutation

Coded gene string	[1,0,0,**1,0,0,1,1,1**,0,0,1]
Position in gene string	1 2 3 **4 5 6 7 8 9** 10 11 12

After Permutation

Coded gene string	[1,0,0,**1,1,1,0,0,1**,0,0,1]
Position in gene string	1 2 3 **4 5 6 7 8 9** 10 11 12

Before Inversion

Coded gene string	[1,0,0,**1,0,0,1,1,1**,0,0,1]
Position in gene string	1 2 3 **9 8 7 6 5 4** 10 11 12

After Inversion

Figure 1.2: Permutation and Inversion operators.

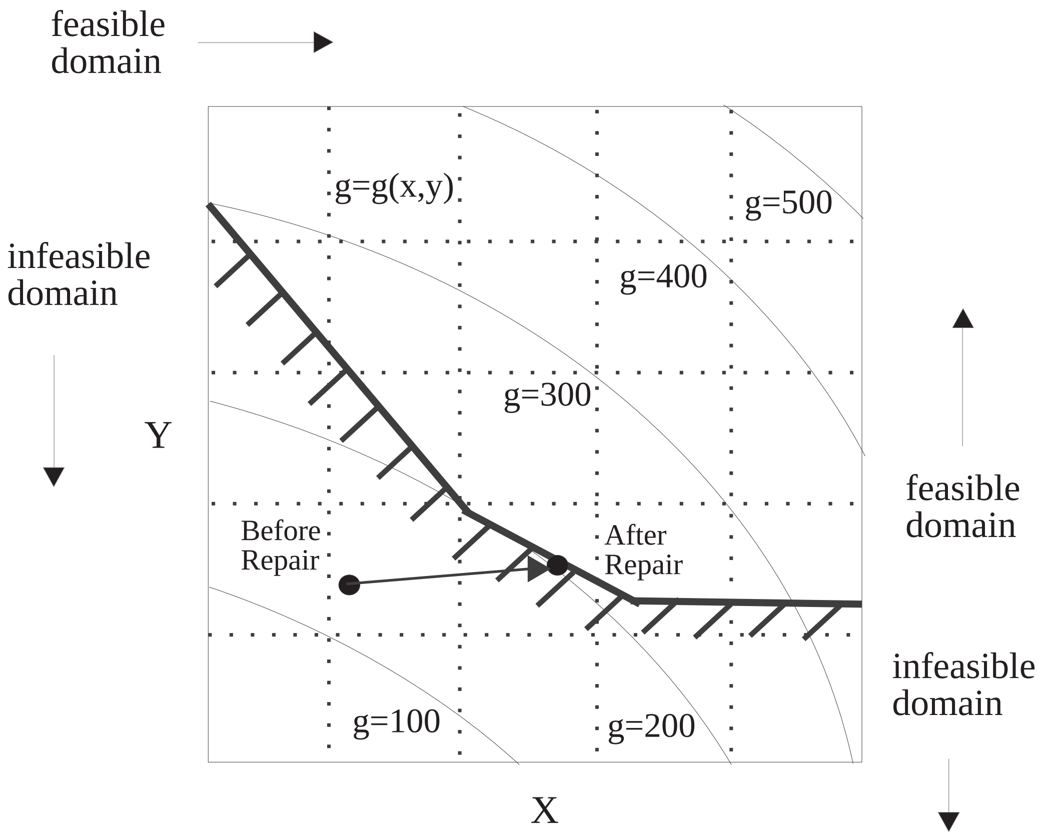


Figure 1.3: A simple repair operator.

designs that have already be analyzed [13]. After the creation of a new population of designs, the tree is searched for either laminates designs with identical stacking sequence or similar performance, such as laminates with identical in-plane strains. Depending on the kind of information that can be retrieved from the tree, the analysis for a given laminate may not be required or significantly reduced. While such a method can reduce computational cost, a binary tree requires a large amount of computer memory and, depending on the problem being investigated, may not be cost effective.

A second method, local improvement, involves searching in small discrete spaces (neighborhoods) around nominal designs by using approximations [14]. This technique was applied to the problem of maximizing the buckling load of a rectangular laminated composite plate. The neighborhood area is determined by randomly interchanging any two plies in a nominal design. The performance of neighborhood designs is effectively and efficiently determined by use of a binary tree (described above) and numerical approximations based on lamination parameters. Local improvement was found to substantially reduce the cost of the algorithm although problems were encountered with singular optima.

In other approaches, parallel processing is used where multiple sub-populations, with identical GA representations, are used to generate solutions. In injection island parallelism, information is both given and received by all sub-populations to obtain optimal designs [15]. Malott and Averill modified this idea by using a GA which simultaneously uses different search strategies. Strategies with increasing levels of resolution simultaneously search small abstract spaces for areas of possible interest by using child sub-populations. The level of resolution depends on how complex the problem is modeled in the gene string of each sub-population [15]. Information obtained from the lower resolution searches is continuously passed to parent sub-populations where the most refined search is implemented. Convergence of the low resolution searches occurs quickly and is then discontinued, saving valuable CPU time. Improvements in both the quality of designs found by the GA and the cost of the search were realized.

Large computation costs have also been avoided by reducing the complexity of the analysis required when using the GA. Time consuming analysis techniques such as the finite element method (FEM) can be eliminated when using GAs by utilizing a combination of optimization methods. Yamazaki used a two-level optimization technique in maximizing the critical buckling load of composite plates and stiffened panels [16]. In the first level of optimization, structural and sensitivity analyses are carried out, taking a significant portion of the total CPU time required. Lamination parameters and the dimensions of the structure are used as design variables and optimized using a linear programming algorithm. Once the optimal lamination parameters have been determined, a second level of optimization implements a GA to find the best stacking sequence. The second level of optimization does not require the implementation of the structural analysis, allowing the GA to be more efficient in its search for the optimum stacking sequence [16].

1.2 GAs Among Other Evolutionary Algorithms and Optimization Techniques

GAs are one of many that have been grouped in a class of search and optimization schemes called evolutionary algorithms (EAs). Trademarks of EAs are their reliance on organism fitness, random information, and probabilistic or deterministic breeding rules to help guide their way toward an optimal point of a design space. However, genetic algorithms have important differences when compared to other EAs such as evolution strategies (ESs) and evolutionary programming (EP). An in-depth comparison of these three types of EAs is given by Bäck [17] who notes how each strategy depends on particular characteristics of EAs to achieve a successful evolutionary search process.

In ESs and EP, mutation is regarded as the main search operator and utilizes self-adaptation rules to control mutation rates. ESs utilize several different types of recombination operators which are also used for certain self-adaptation procedures rather than searching the design space. ESs and EP also use extinctive selection schemes where some individuals in the population are excluded from being selected for reproduction [17]. EP algorithms do not utilize any type of recombination operator and use deterministic selection methods. In comparison, genetic algorithms use crossover as their main search operator and do not make use of any type of self-adaptation procedure. Extinctive selection methods are not utilized in GAs either.

Both Goldberg [7] and Buckles [18] have noted several distinct differences between GAs and classical optimization algorithms as well. First, GAs work with objection function or fitness information only and do not rely on derivative information. Thus, GAs are applicable to design problems in which discontinuous functions must be evaluated. Derivative-based optimization schemes are ideally suited to smooth functions with a single optimum point, see Figure 1.4, but would have difficulty with the non-smooth function given in Figure 1.5. However, deterministic pattern search algorithms can effectively deal with non-differentiable problems.

Second, GAs work with a group of design points instead of a single point. With this property, a genetic algorithm can search many areas of the design space at once. While one design area which contains good information is being exploited, other areas of the design space can be explored. Working with a population of designs also make GAs less susceptible to difficulties encountered in problems with noisy design spaces. On the other hand, calculus-based methods work with a single point at a time and may have problems with functions with local optimum points. If a calculus-based method were to try and find the largest value of the function given in Figure 1.6 by starting

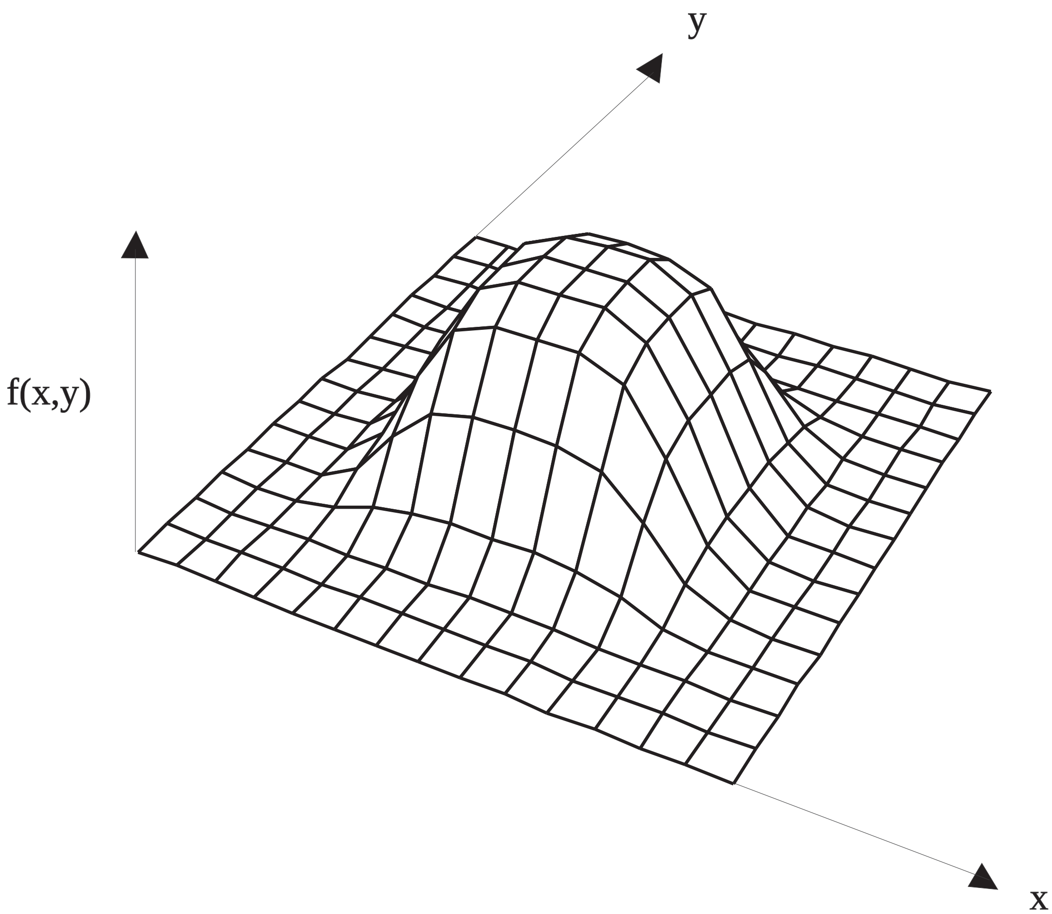


Figure 1.4: A function with a single optimum point.

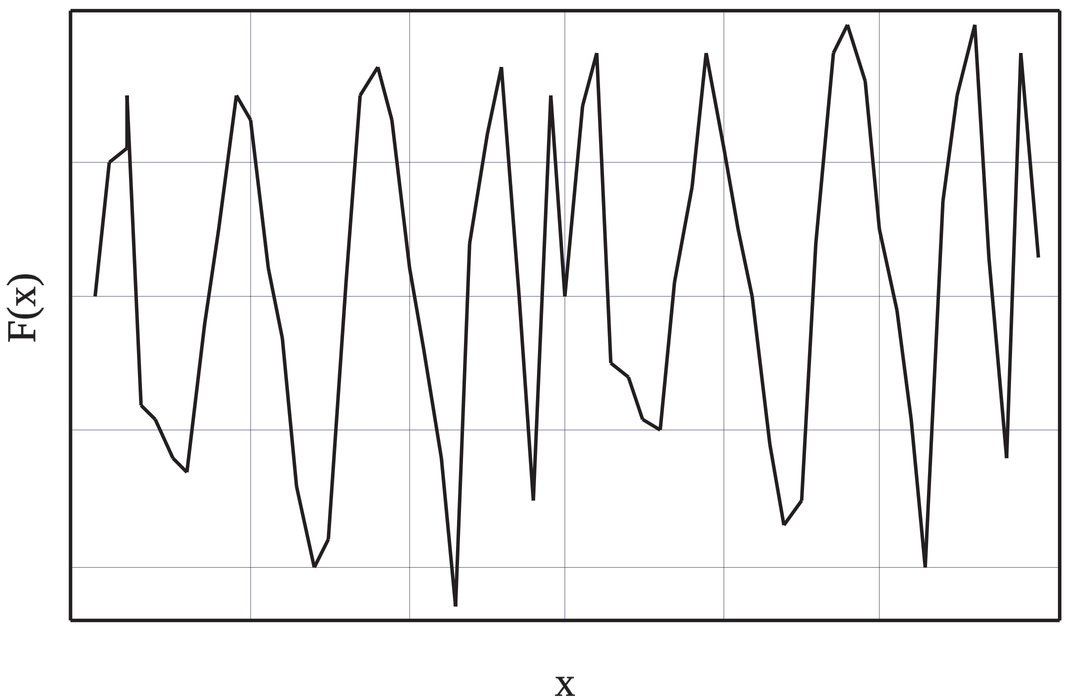


Figure 1.5: A function ill-suited to derivative-based optimization methods.

at point A, it may incorrectly find point B as the solution instead of point C.

Thirdly, GAs are implicitly parallel algorithms because they can test many areas of the design space without having to evaluate every point in each area [18]. From conducting only a few function evaluations, a GA can determined whether a particular area in the design space is worth exploiting, or explore other areas in the design space. However, there is a trade off between exploitation and exploration. A genetic algorithm which is allowed to explore too much may miss finding an optimum design point because it failed to exploit attractive areas of the design space. Likewise, a genetic algorithm which does not explore enough will exploit attractive areas of the design space satisfactorily and reach a local optimum, thereby missing other favorable areas which may contain the global optimum design.

Finally, genetic algorithms rely on a set of probabilistic rules instead of deterministic ones, allowing for some of the decisions during the search to be made by drawing random numbers. By making random moves in the design space, a GA can avoid local optima points easier [8]. However, it is important to note that GAs are not random searches, but use random information to help guide an organized search scheme toward an optimum point of coded parameter space [7]. Unlike gradient-based methods, a GA can be initialized with the same population and converge to different solutions also. Many or all of these properties are shared by classical pattern search and probabilistic optimization methods. Thus, while the implementation mechanisms of GAs are distinctive, the fundamental concepts and features are not unique to GAs.

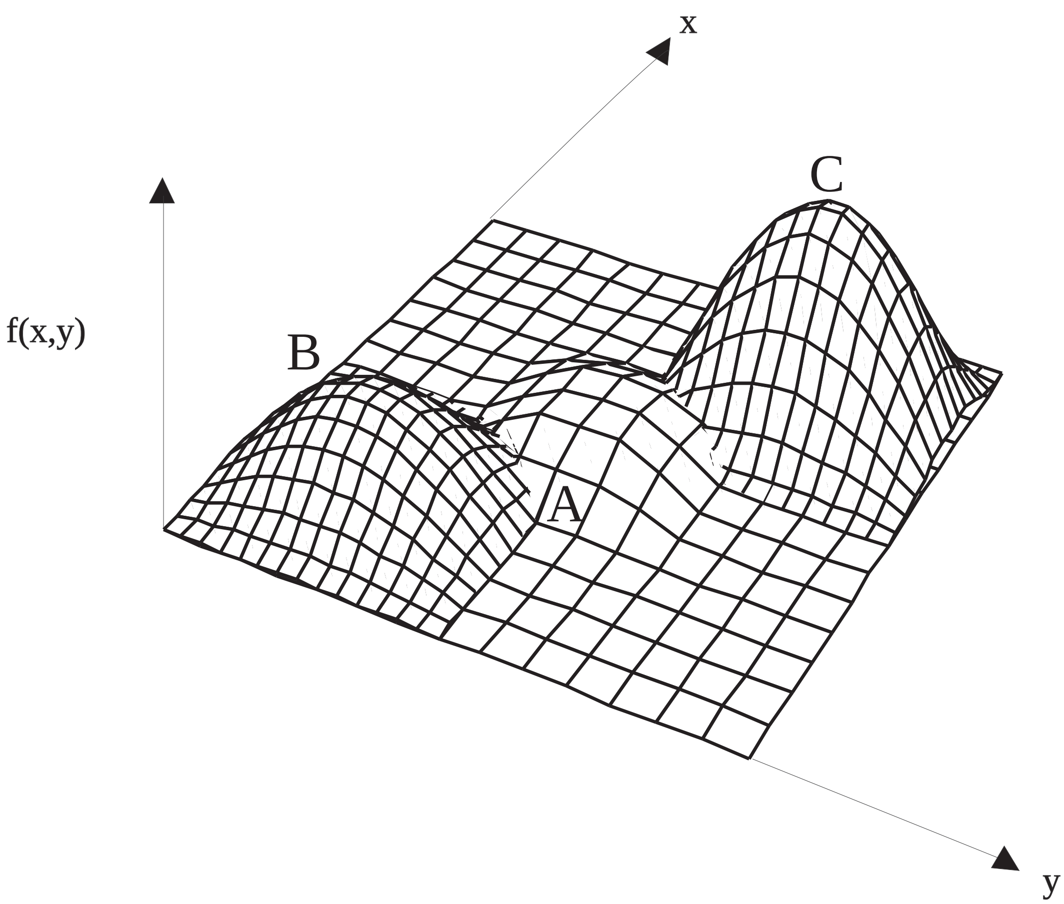


Figure 1.6: A function with local optimum points.

1.3 Composite Structure Design and Optimization

Composite structures are usually comprised of laminates where thin layers of material (plies) are stacked on top of one another and held together with a matrix material such as epoxy. Each layer of material consists of very small diameter fibers that may be oriented at different angles, see Figure 1.7. The matrix also supports and protects the fibers and provides a means for distributing and transferring loads between the fibers [4]. The strength and stiffness of the fibers are highest in the direction that the fiber is oriented in, and lowest perpendicular to the fiber direction. The goal of the designer is to find the correct number of layers and ply orientation angles that will provide a structure with the best performance given certain loading conditions. Such design problems are often placed under certain geometry, manufacturing, cost, and failure constraints as well. The growing use of composite materials is due to their higher stiffness-to-weight and strength-to-weight properties, and their ability to be finely tuned to the loading environment of the structure. However, composite laminate design and optimization often requires complicated analysis routines and has thus received a substantial amount of attention in recent years.

In early works, Foye [19] used a random search method to find the optimal stacking sequence with the smallest number of plies, while satisfying strength and stiffness requirements. Waddoups [20] employed a brute force method in which all possible designs were evaluated. In both studies [19] and [20] multiple in-plane loading conditions were considered and ply orientation angles and the number of plies were treated as design variables. Verette [21] and Kenoshi [22] conducted laminate optimization studies that used stability constraints based on simplified buckling analyses to avoid complications involved with solving eigenvalue problems.

Composite laminate design has also been formulated as a continuous optimization problem,

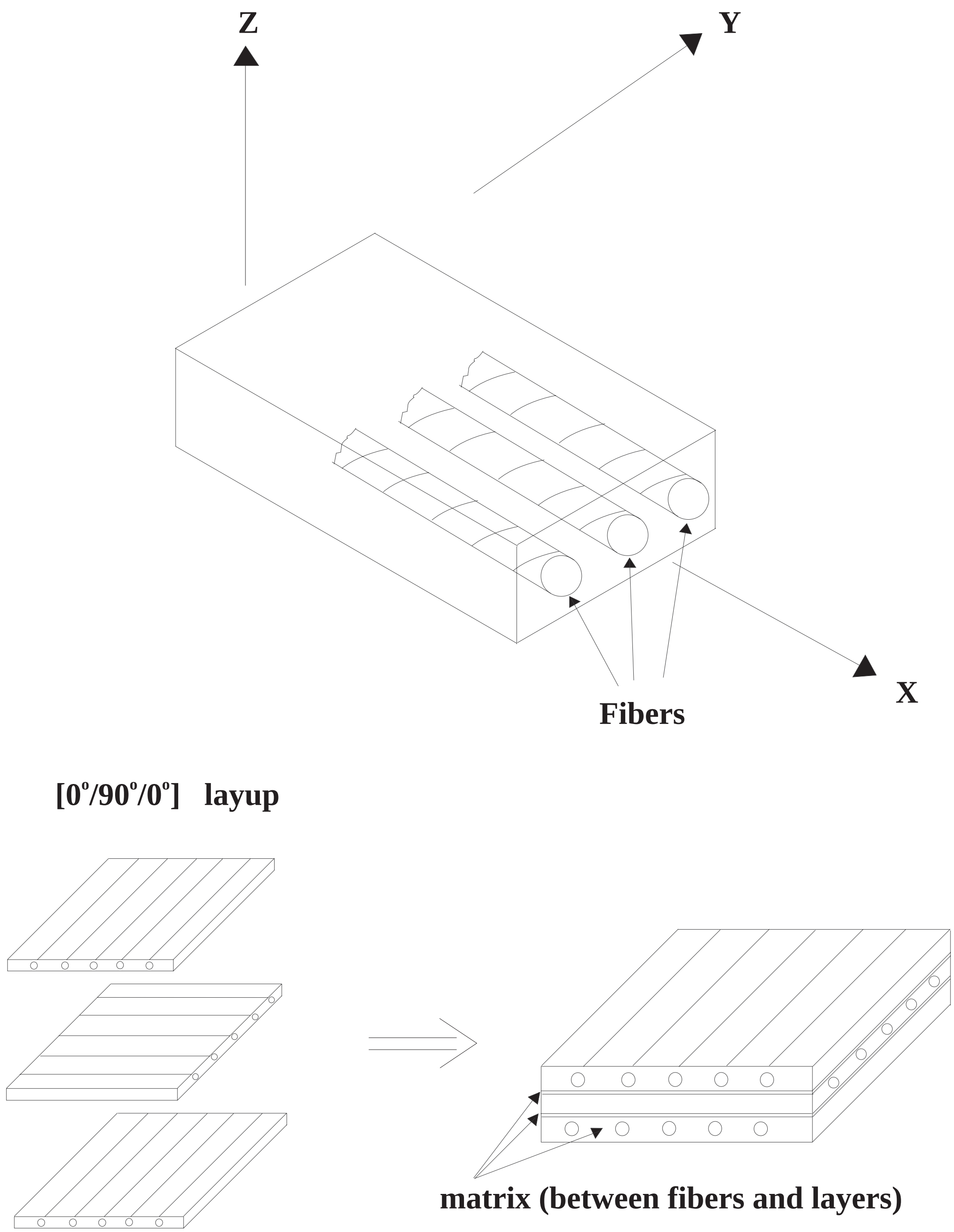


Figure 1.7: Composite laminate make-up.

once again using ply thicknesses and ply orientation angles as design variables. Schmit and Farshi [23] studied stacking sequence design by using a more complicated approach. Ply thicknesses were treated as continuous design variables and a small set of pre-assigned orientation angles were used to avoid problems associated with discrete programming, and to avoid encountering numerous local minima in the design space. Ply thicknesses were then rounded off to an appropriate value after a solution was obtained. The objective function and stiffness constraint were found to be linear functions of the design variables and the strength constraint, which was non-linear was transferred to set of sequential linear problems that could be solved easily [23]. The efficiency of this method was improved by removing constraints that were not potentially critical during certain stages of the search. This approach was also successfully applied to the problems involving buckling by sequentially linearizing the buckling constraint with respect to the ply thicknesses [24]. Gürdal [25] used continuous optimization in conjunction with a penalty function to force the ply orientation angles to discrete values. However, restricting the ply angles to a small set of acceptable values resulted in a substantial reduction in the buckling. The penalty function approach also showed difficulty in locating the global optimum for the problem.

Other studies have applied the branch and bound algorithm in the design and optimization of laminated composite plates. Haftka and Walsh [26] solved the stacking sequence problem for buckling load maximization. The non-linear problem, resulting from using ply thicknesses as design variables, is linearized by using ply-orientation-identity variables, and then solved using a branch and bound algorithm. Nagendra [27] solved a similar problem with the addition of strain constraints, once again introducing non-linearities to the problem. This problem was solved by using a sequence of linear integer programming techniques.

1.4 Genetic Optimization of Composite Structures

It has been established that stacking sequence design of composite laminates requires discrete programming since ply thicknesses and orientation angles are restricted to a discrete set of values. This restriction is due to manufacturing limitations because plies are fabricated at certain thickness values. Furthermore, a majority of composite structures are still manually constructed and it is often to difficult to accurately hand-lay plies at odd orientation angles.

Design procedures discussed in the previous section with continuous design variables have several disadvantages. First, composite laminate design and optimization problems possess non-linear functions of the number of plies, ply thicknesses, and orientation angles and substantial effort is required for transformation to a linear problem. Second, composite laminate design often involves many local optimum designs because their performance is characterized by a number of parameters which is typically smaller than the number of design variables [8]. Sequential linearization is a standard approach to solving non-linear problems but can often get trapped in local optimum design areas [27]. Thirdly, rounding off design variables when using continuous optimization methods haven shown to produce sub-optimal or even unfeasible designs [28]. Continuous optimization and branch-and-bound algorithms may not be well suited to complex composite structures either because they exhibit an exponential dependence on the number of design variables [8].

In recent years, genetic algorithms have been successfully applied to large, non-convex, integer programming problems, see for example Hajela [29] and Rao et al. [30]. Thus, it was obvious that GAs would be well suited for the design and optimization of laminated composite plates. Early works include Callahan [31] who used GAs for stacking sequence optimization of composite plates, and Nagendra [32, 33, 34] who did extensive research work with GAs and stiffened composite panels.

As discussed in Section 1.2, GAs are excellent all-purpose discrete optimization algorithms because they can handle linear and non-linear problems or noisy search spaces by using payoff (objection function) information only. Genetic optimizers are also global in scope and are less likely to be trapped in local optimum design areas, although they have been known to converge to local optimum designs if they are used blindly [8]. By working with a population of designs, GA are also useful in finding the optimal and perhaps many near-optimal designs as well, which may be advantageous to the designer. If, for example, it is not possible to efficiently manufacture an optimal laminate design, it may be replaced with a design of similar performance that can be manufactured.

1.4.1 Research Objectives

Similar to integer programming algorithms, GAs are often too expensive to implement if the cost of each laminate design is expensive, even though substantial effort has been made to improve their efficiency, as discussed in Section 1.1.3. Unfortunately, this is often the case with composite structures which usually involve complex finite element packages for accurate structural analysis. Furthermore, it is usually not a trivial process to adapt or expand certain analysis and optimization routines to composite laminate design when additional constraints or modifications are introduced to the problem. This is not the case with genetic algorithms which are quite flexible and accommodating to a wide range of other problems also.

Thus, the objective of this work is two-fold. The first part of this research is directed towards furthering the effort in improving the efficiency of GAs in stacking sequence design of composite laminates. The second half of this work will emphasize the flexibility of genetic algorithms in the design and optimization of composite structures. This will be accomplished by incorporating information concerning the cost of the structure into the problem. Along with structural performance and weight, cost is an area of great interest when considering optimization studies in structural design. Typically, the cost of a composite laminate is minimized by reducing the weight (i.e., the amount of material required) of the structure or by minimizing cost functions which account for manufacturing issues. However, another method for reducing cost that has received little if any attention is to allow for more than one material in the stacking sequence. In having a multi-material stacking sequence, a designer can use layers of low cost material, which have inferior performance qualities, at locations in the structure where performance is not critical. Hence, the flexibility of the GA will be displayed by modifying the algorithm to handle multiple materials in the stacking sequence.

In Chapter 2, the methodology of applying a genetic algorithm to minimum weight design of symmetrically laminated composite plate is presented. Chapter 3 introduces the concepts of multiple elitist and variable elitist selection in which a prescribed number of designs among the top performers of the parent generation are passed on to the child generation to produce new populations in the GA optimization process. Three different implementations of both a multiple elitist scheme and variable elitist scheme are explored. The primary objective of the new selection module is to increase the GA's reliability and final population richness (to be defined) while decreasing the computational cost of the algorithm. In Chapter 4, a comparison between the performance of the new selection schemes and a typical elitist scheme is given using a genetic algorithm for maximization of the bend twist displacement of a cantilevered composite plate.

A genetic algorithm for simultaneous cost and weight minimization of simply supported composite plates comprised of two materials is explored in Chapter 5. Materials are chosen such that one has superior strength and stiffness capabilities but is expensive, while the second material is less expensive but has inferior performance. Convex combinations of cost and weight objective functions are used to study tradeoffs between a laminate's weight and cost by adjusting the influence that each has on the overall performance of the laminate. In the final chapter, conclusions to this research and some ideas on future work are presented.

Chapter 2

A GA for Stacking Sequence Design of Laminated Composite Plates

The objective of this Chapter is to devise a genetic algorithm for stacking sequence design of symmetrically laminated composite plates. Stacking sequence design implies the determination of the number of plies in the laminate as well as their orientation. With this feature, the GA may be used to control laminate weight by adjusting the number of plies in the laminate stacking sequence. The genetic algorithm will not be allowed to adjust the dimensions of the plate throughout the optimization process. Two different versions of a genetic algorithm are explored.

In section 2.1, a genetic algorithm is devised for designing composite laminates comprised of one material only. A description of the genetic code is given first, followed by a detailed description of the genetic algorithm procedure which is referred to as GA–I. The GA–I algorithm will be used for maximization of the bend-twist displacement of a cantilevered composite plate with strength and ply contiguity constraints. Details of this problem are discussed later in Chapter 4.

In section 2.2, a second version of the genetic algorithm, referred to as GA–II, for designing composite laminates comprised of multiple materials is described. Modifications to the basic genetic code, GA–I, and genetic procedure for the GA–II algorithm are also discussed. This version of the GA will be used for simultaneously optimizing the cost, weight, and buckling load of a simply supported composite plate, which will be discussed in Chapter 5.

A discussion of other implementation issues concerning both versions of the GA is given in the last part of this chapter. Topics include stopping criterion, objective function formulation for minimum weight design, and an introduction to ideas concerning the selection of successive generations of laminates throughout the GA process.

2.1 GA–I: Composite Laminates Comprised of One Material

For the GA–I algorithm, one string of genes is used to represent one half of a symmetrically laminated composite plate. The length of the gene string is kept fixed throughout the optimization process. Each gene in the string is represented by an integer value between 0 and 10 and determines whether the ply stack location is empty or occupied with a 3-ply stack which may be oriented at any angle between 0° and 90°, in increments of 10°, see Figure 2.1. Although the gene string length is fixed, having empty plies makes it possible to change the laminate thickness during the optimization process. Coding the ply orientation angles as consecutive integers is not really necessary, since all the genetic operators could apply directly to the angles. However, the implementation of random choices with given probabilities is both easier to describe and program for integer intervals than for arbitrary sets of objects.

All plies in the stacking sequence have the same prescribed thickness value and are stacked in groups of three (i.e., 0°_3, 10°_3, 20°_3, etc.) to keep the number of design variables used and the size of the design space to a minimum. An example of a decoded stacking sequence is given in Figure 2.2, where E represents an empty 3-ply stack. Note that empty stacks are pushed to the outer edge (left end) of the laminate stacking sequence to avoid having voids in the laminate.

2.1.1 GA Procedure

An initial population of genetic strings with randomly chosen genes is created first. The size of the population used in the present work remains constant throughout the genetic optimization. Various genetic operators are applied at given probabilities to generate new laminates. In order to form successive generations, parents are chosen from the current population based on their fitnesses, as described in the following subsection. The fitness calculation usually involves function values that are determined from separate analysis subroutines or packages. Next, the crossover, mutation, and ply swap operators are applied to create child designs, who are hopefully better suited to their environment than their parents. The child population is then analyzed and ranked. To complete the generation cycle, a selection scheme is implemented which determines which laminates from the child and parent population will be placed in the next generation. One generation after another is created until some stopping criterion is met. A schematic of the genetic algorithm procedure is given in Figure 2.3.

2.1.2 Parent Selection

Parent selection is accomplished using a roulette wheel concept. This method of selection differs from other evolutionary algorithms because it gives every member of the population a chance to become a parent (i.e., a non-extinctive breeding procedure).

Orientation gene

0 = 1 three-ply empty stack
1 = 1 three-ply stack oriented at 0^o
2 = 1 three-ply stack oriented at 10^o
3 = 1 three-ply stack oriented at 20^o
4 = 1 three-ply stack oriented at 30^o
5 = 1 three-ply stack oriented at 40^o
6 = 1 three-ply stack oriented at 50^o
7 = 1 three-ply stack oriented at 60^o
8 = 1 three-ply stack oriented at 70^o
9 = 1 three-ply stack oriented at 80^o
10 = 1 three-ply stack oriented at 90^o

Figure 2.1: GA–I code key for laminate stacking sequence.

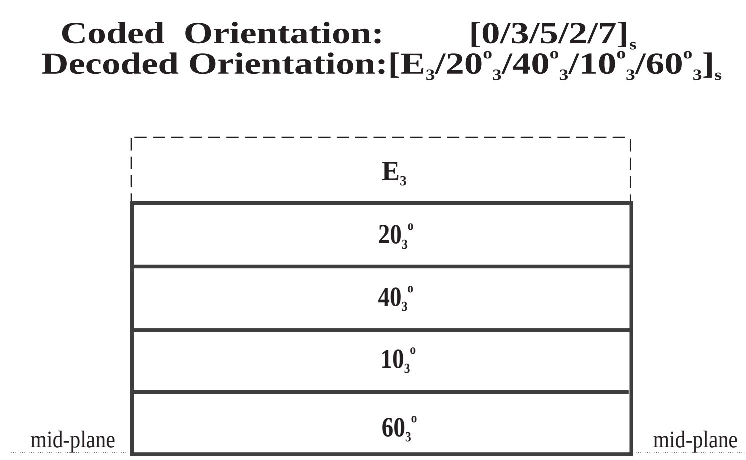


Figure 2.2: Sample stacking sequence arrangement for GA–I.

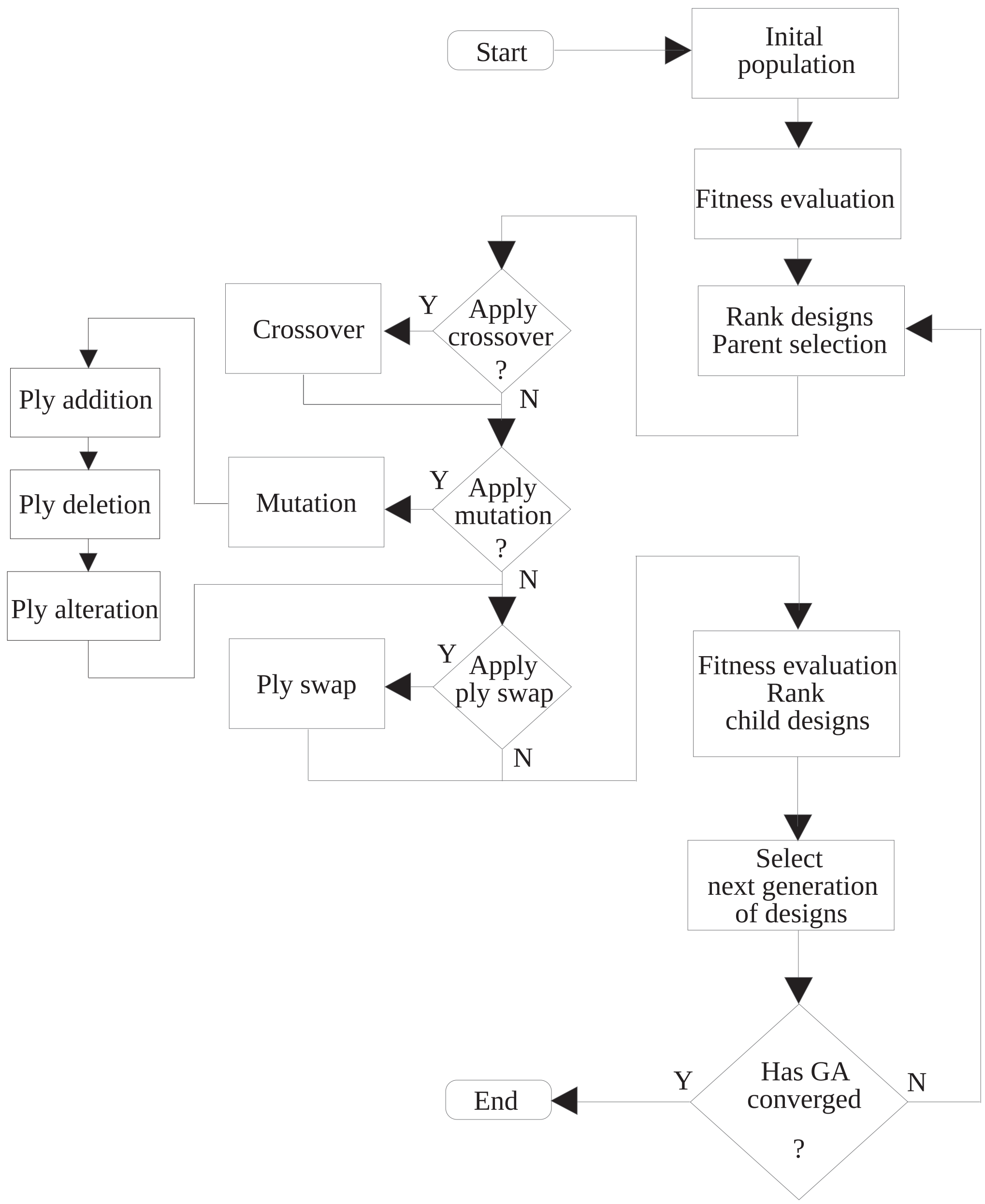


Figure 2.3: Genetic algorithm procedure.

Before parent selection can begin, all laminates must be ranked from best to worst according to the value of each laminate's objective function. A roulette wheel is implemented where the i^{th} ranked laminate in the population is given an interval $[\phi_{i-1}, \phi_i)$, whose size depends on the population size, P, and its rank, i, in the population:

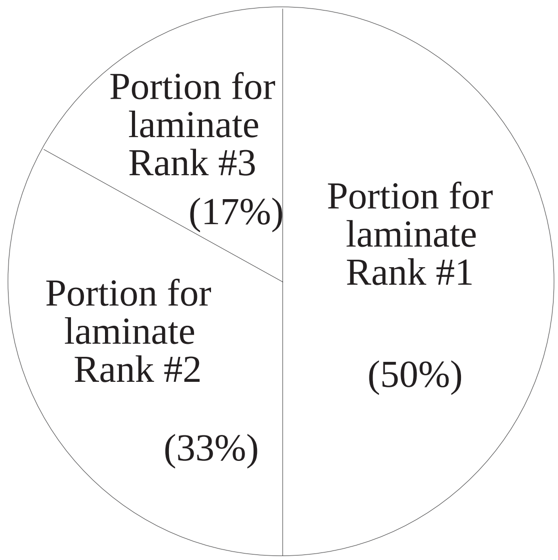
$$\phi_i = \phi_{i-1} + \frac{2(P-i+1)}{P(P+1)}, \tag{2.1}$$

where $\phi_o = 0$, and $i = 1, ..., P$. For example, if there are three laminates in a population, the roulette wheel is divided into three pieces with the best laminate taking 50% of the wheel, the second best taking 33%, and the poorest taking 17%, see Figure 2.4–a.

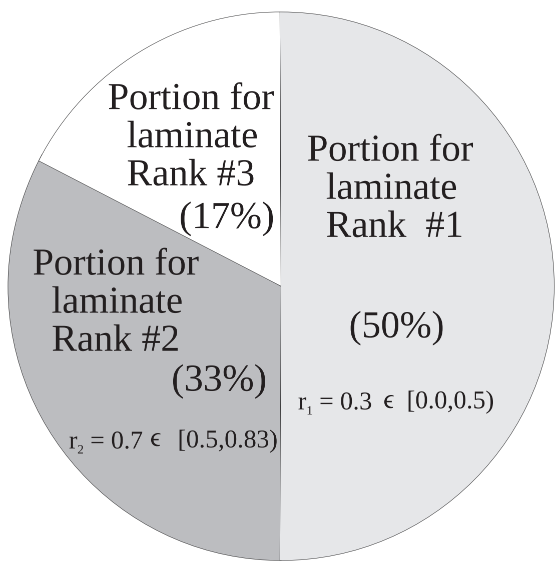
A uniformly distributed random number is generated between 0 and 1; laminate i is selected as a parent if the number lies in the interval $[\phi_{i-1}, \phi_i)$. Continuing with the above example, if random numbers $r_1 = 0.3 \in [0, .5)$ and $r_2 = 0.7 \in [.5, .83)$ are drawn, then laminate 1 and laminate 2 will become parents of the first child, see the shaded regions in Figure 2.4–b. Parents of a child are required to be distinct laminates from the population.

2.1.3 Crossover

Children are created by combining a portion of each parent's genetic string in an operation called one-point crossover. To determine the crossover point, a uniformly distributed random number is chosen and then multiplied by one less than the maximum number of non-empty genes in the two parents. The integer ceiling value of this product determines the crossover point, see Figure 2.5. The gene string is then split at the same point in both parents. The left piece from parent 1 and the right piece from parent 2 are combined to form a child laminate. To ensure that empty plies are not swapped, all empty plies are pushed to the left side of the coded string (this corresponds to



a) Roulette wheel distribution for 3 laminates.



b) Parent selection using random numbers, r_1 and r_2.

Figure 2.4: Parent selection using a roulette wheel.

the outer edge of the laminate). The random crossover point is restricted to fall in the non-empty region of both parent laminates to ensure that the child laminate is unique, see Figure 2.5. If, during the creation of the child population, crossover is not applied then one of the parent laminates is cloned into the child string. Child laminates are also forced to be distinct from each other and from laminates in the parent population. If a distinct child cannot be found after a prescribed number of iterations, then one of the parents is cloned into the child population also. The crossover process is repeated as many times as necessary to create a new population of laminates.

2.1.4 Mutation

After a child is created, the operations of adding, deleting, or mutating genes occur with small probabilities. These operators make up genetic mutation, and are illustrated in Figure 2.6. When adding a ply stack, a uniform random number is chosen to determine the orientation. For the design problems considered in this work, outer plies in the laminate will get set up faster because they have a greater influence on the objective function. Thus, added ply stacks are always introduced at the mid-plane of the laminate, see Figure 2.6–a.

To delete a ply stack, a random number is chosen and the corresponding stack is removed from the stacking sequence by replacing it with a 0 gene. The laminate is then re-stacked so that all empty plies are pushed to the outer edge of the laminate, see Figure 2.6–b.

Gene alteration is shown in Figure 2.6–c. Each gene in the string switches with a small probability to any other permissible integer value (as defined in Figure 2.1) except 0 and the value of the gene before ply alteration occurs. Ply alteration does not operate on empty genes either.

GA Code

crossover point

Parent 1: $[\mathbf{0/3}\;|\;/5/2/7]_s$
Parent 2: $[0/0\;|\;\mathbf{/6/4/2}]_s$ $\longrightarrow$ $[\mathbf{0/3/6/4/2}]_s$

Stacking Sequence

crossover point

Parent 1: $[\mathbf{E_3/20^o{}_3}\;|\;/40^o{}_3/10^o{}_3/60^o{}_3]_s$
Parent 2: $[E_3/E_3\;|\;\mathbf{/50^o{}_3/30^o{}_3/10^o{}_3}]_s$ $\longrightarrow$ $[\mathbf{E}_3/\mathbf{20^o{}_3/50^o{}_3/30^o{}_3/10^o{}_3}]_s$

Figure 2.5: One-point crossover.

2.1.5 Ply Swap

In previous works, a permutation operator (see Figure 1.2) was often used to aid the genetic search but was found to shuffle the digits in the gene string too much [12]. Thus, a less disruptive operator, ply swap, was designed and will be used in favor of permutation for the design problems considered in this work. The ply swap operator is implemented by randomly selecting two genes in the string and switching their positions, see Figure 2.7. Ply swap can be effective for problems where certain parts of the laminate stacking sequence get set up faster than others. For example, if the optimal stacking sequence for the outer section of the laminate has been determined first (as is the case for laminate design problems which involve bending), the ply swap operator may help the GA determine the optimal orientations for the inner part of the laminate by swapping plies from each section.

2.2 GA–II: Composite Laminates with Multiple Materials

In this section, modifications to the GA–I algorithm to allow for stacking sequences with multiple materials will be discussed. The second version of the genetic algorithm will be called GA–II. In the previous section, the entire laminate was comprised of one material. Thus, one chromosome consisting of one gene was sufficient to represent the laminate stacking sequence. However, to accommodate two or more materials, each chromosome is expanded to include two gene strings, one for ply orientation and another one for material definition. The representation of genes by integers in each string is maintained. Genes in the first string will once again determine whether the ply location is empty or filled with a ply of prescribed orientation. Corresponding genes in the second string determine the ply material if the ply is present. By employing two gene strings, the

GA Code

Before Ply Addition: $[\mathbf{0}/3/6/4/2]_s$
After Ply Addition: $[3/6/4/2/\mathbf{10}]_s$

Stacking Sequence

Before Ply Addition: $[\mathbf{E_3}/20^{\circ}_3/50^{\circ}_3/30^{\circ}_3/10^{\circ}_3]_s$
After Ply Addition: $[20^{\circ}_3/50^{\circ}_3/30^{\circ}_3/10^{\circ}_3/\mathbf{90^{\circ}_3}]_s$

a) Ply Addition (at least 1 empty stack)

GA Code

Before Ply Deletion: $[10/3/\mathbf{6}/4/2]_s$
After Ply Deletion: $[10/3/\mathbf{0}/4/2]_s$
Restack: $[\mathbf{0}/10/3/4/2]_s$

Stacking Sequence

Before Ply Deletion: $[90^{\circ}_3/20^{\circ}_3/\mathbf{50^{\circ}_3}/30^{\circ}_3/10^{\circ}_3]_s$
After Ply Deletion: $[90^{\circ}_3/20^{\circ}_3/\mathbf{E_3}/30^{\circ}_3/10^{\circ}_3]_s$
Restack: $[\mathbf{E_3}/90^{\circ}_3/20^{\circ}_3/30^{\circ}_3/10^{\circ}_3]_s$

b) Ply Deletion (at least 2 full stacks)

GA Code

Before Ply Alteration: $[0/10/3/\mathbf{4}/2]_s$
After Ply Alteration: $[0/10/3/\mathbf{6}/2]_s$

Stacking Sequence

Before Ply Alteration: $[E_3/90^{\circ}_3/20^{\circ}_3/\mathbf{30^{\circ}_3}/10^{\circ}_3]_s$
After Ply Alteration: $[E_3/90^{\circ}_3/20^{\circ}_3/\mathbf{50^{\circ}_3}/10^{\circ}_3]_s$

c) Single Ply-Stack Alteration (filled plies only)

Figure 2.6: Mutation.

GA Code

Before Ply Swap: $[0/10/\mathbf{3}/6/\mathbf{2}]_s$
After Ply Swap: $[0/10/\mathbf{2}/6/\mathbf{3}]_s$

Stacking Sequence

Before Ply Swap: $[E_3/90^o_3/\mathbf{20^o_3}/50^o_3/\mathbf{10^o_3}]_s$
After Ply Swap: $[E_3/90^o_3/\mathbf{10^o_3}/50^o_3/\mathbf{20^o_3}]_s$

Figure 2.7: Ply swap.

number of materials that may be used in the stacking sequence may be changed easily by adjusting the size of the material gene alphabet.

In the application of the two material design problem, single ply stacks are used instead of stacks of 3 plies, with ply orientation choices of $0°$, through $90°$ with $\pm 15°$ increments, see Figure 2.8. Ply thickness may take one of two prescribed values depending on the material that a ply is comprised of, as shown in the sample stacking sequence of Figure 2.9.

2.2.1 Decoding the Gene Strings

To incorporate the two material concept into the GA, a more complex decoding procedure was required. Orientation genes that are coded as integer values between 2 and 6 represent either the positive or the negative value of the corresponding ply orientation angle defined in Figure 2.8. For example, a 4 represents either a $+45°$ or a $-45°$ ply in the orientation gene. For the problem considered in Chapter 5 laminates will be constrained to have a balanced stacking sequence to simplify analysis procedures. To maintain a balanced laminate or obtain a laminate as close to balanced as possible, the $\pm\theta$ plies are decoded alternately. For example, the first 4 (starting from the outer edge of the laminate) encountered for a particular material is decoded as a $+45°$ ply and the next 4 for the same material is decoded as a $-45°$ and so on. This decoding methodology applies for all plies accept those oriented at $0°$ and $90°$, which are decoded in the normal fashion. Thus, a laminate stacking sequence of $\pm\theta$ plies is balanced if each $+\theta$ ply is matched with a $-\theta$ ply of the same material. If the $\pm\theta$ plies are balanced for one material but not the other, the laminate is unbalanced.

Orientation gene

0 = 1 empty ply
1 = 1 ply oriented at 0^o
2 = 1 ply oriented at +/- 15^o
3 = 1 ply oriented at +/- 30^o
4 = 1 ply oriented at +/- 45^o
5 = 1 ply oriented at +/- 60^o
6 = 1 ply oriented at +/- 75^o
7 = 1 ply oriented at 90^o

Material gene

0 = 1 empty ply
1 = material #1
2 = material #2

Figure 2.8: GA–II code key for laminate stacking sequence.

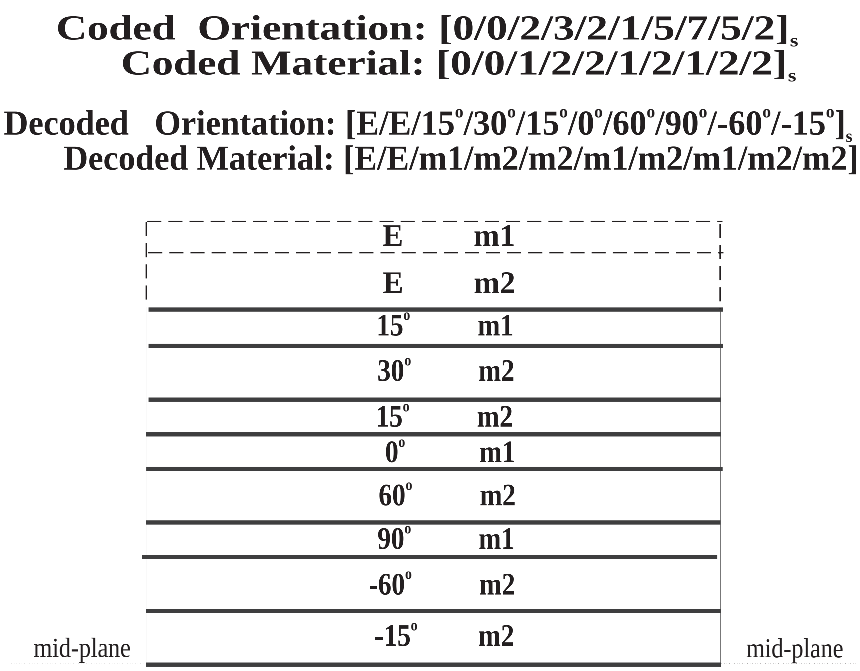


Figure 2.9: Sample stacking sequence arrangement for GA–II.

2.2.2 Modifications to the genetic operators

The procedure for the GA-II algorithm remains mostly unchanged from the one used in the GA-I version, except for small modifications made to the genetic operators. When a parent is selected for reproduction, both the ply orientation gene string and material gene string are used when creating a child. In the crossover procedure, the orientation and material gene strings are split at the same point in both parents. The left pieces of both the orientation and material gene strings from parent one and the corresponding right pieces from parent 2 are then combined to form a child laminate, see Figure 2.10. Crossover may also have the effect of changing some $+\theta$ plies to $-\theta$ also. This phenomenon is also seen in Figure 2.10 if one looks at the plies that are coded as 3 in both parents. In the second parent there is only one 3 for material 2 which gets decoded as $+30°$ ply. When the child is created, the ply coded as a 3 that is passed from parent 2 now gets decoded as a $-30°$ ply. This is because parent 1 also passed a ply to the child laminate that was coded as a 3 made of the same material. Since the $30°$ ply from parent 1 will be closer to the outer edge of the laminate, it will be decoded as $+30°$ where as the second 3, which came form parent 2 gets decoded as $-30°$.

The procedure for the mutation operator is modified slightly also. Ply addition and deletion are done simultaneously on both the orientation and material gene strings. Added plies are once again introduced at the mid-plane of the laminate. When a ply is added, the corresponding material gene is also added, see Figure 2.11. When a ply is deleted, it is picked at random with the corresponding material gene also being deleted, see Figure 2.12. Gene alteration is implemented separately on each gene string, with the same or different probabilities. If a gene is altered in the orientation gene, the corresponding material gene may not necessarily be altered, see Figure 2.13. Furthermore, when genes are switched in the ply swap operator, both the orientation and material genes are swapped

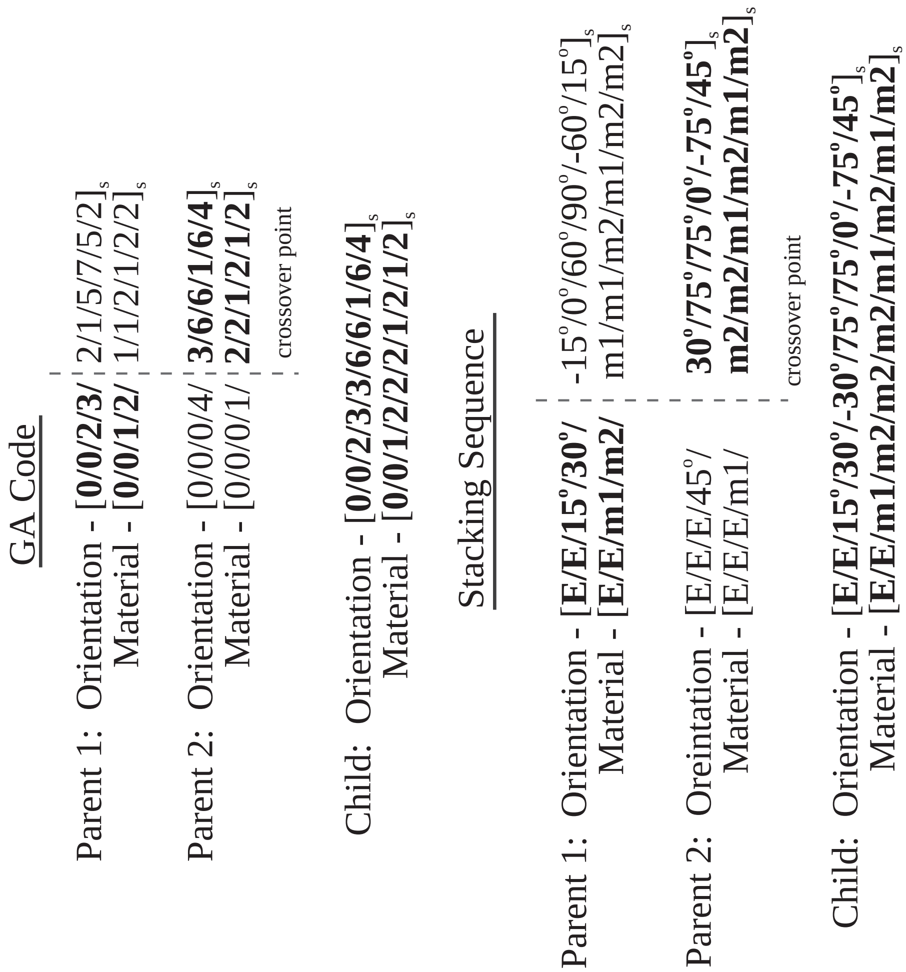


Figure 2.10: Modified crossover operator.

GA Code

Before ply addition: Orientation - $[0/\mathbf{0}/2/3/3/6/6/1/6/4]_s$
Material - $[0/\mathbf{0}/1/2/2/2/1/2/1/2]_s$

After ply addition: Orientation - $[0/2/3/3/6/6/1/6/4/\mathbf{6}]_s$
Material - $[0/1/2/2/2/1/2/1/2/\mathbf{1}]_s$

Stacking Sequence

Before ply addition: Orientation - $[E/\mathbf{E}/15^\circ/30^\circ/-30^\circ/75^\circ/75^\circ/0^\circ/-75^\circ/45^\circ]_s$
Material - $[E/\mathbf{E}/m1/m2/m2/m2/m1/m2/m1/m2]_s$

After ply addition: Orientation - $[E/15^\circ/30^\circ/-30^\circ/75^\circ/75^\circ/0^\circ/-75^\circ/45^\circ/\mathbf{75^\circ}]_s$
Material - $[E/m1/m2/m2/m2/m1/m2/m1/m2/\mathbf{m1}]_s$

Figure 2.11: Modified mutation operator - ply addition.

GA Code

Before ply deletion: Orientation - [0/6/2/3/3/**6**/6/1/6/4]$_s$
Material - [0/1/1/2/2/**2**/1/2/1/2]$_s$

After ply deletion: Orientation - [0/6/2/3/3/**0**/6/1/6/4]$_s$
Material - [0/1/1/2/2/**0**/1/2/1/2]$_s$

Restack: Orientation - [0/**0**/6/2/3/3/6/1/6/4]$_s$
Material - [0/**0**/1/1/2/2/1/2/1/2]$_s$

Stacking Sequence

Before ply deletion: Orientation - [E/75°/15°/30°/-30°/**75°**/-75°/0°/75°/45°]$_s$
Material - [E/m1/m1/m2/m2/**m2**/m1/m2/m1/m2]$_s$

After ply deletion: Orientation - [E/75°/15°/30°/-30°/**E**/-75°/0°/75°/45°]$_s$
Material - [E/m1/m1/m2/m2/**E**/m1/m2/m1/m2]$_s$

Restack: Orientation - [E/**E**/75°/15°/30°/-30°/-75°/0°/75°/45°]$_s$
Material - [E/**E**/m1/m1/m2/m2/m1/m2/m1/m2]$_s$

Figure 2.12: Modified mutation operator - ply deletion.

GA Code

Before ply alteration: Orientation - [0/0/6/2/**3**/3/6/1/6/4]$_s$
Material - [0/0/1/1/2/2/**1**/2/1/**2**]$_s$

After ply alteration: Orientation - [0/0/6/2/**5**/3/6/1/6/4]$_s$
Material - [0/0/1/1/2/2/**2**/2/1/**2**]$_s$

Stacking Sequence

Before ply alteration: Orientation - [E/E/75°/15°/**30°**/-30°/-75°/0°/75°/45°]$_s$
Material - [E/E/m1/m1/m2/m2/**m1**/m2/m1/**m2**]$_s$

After ply alteration and decoding: Orientation - [E/E/75°/15°/**60°**/30°/75°/0°/-75°/45°]$_s$
Material - [E/E/m1/m1/m2/m2/**m2**/m2/m1/**m1**]$_s$

Figure 2.13: Modified mutation operator - ply alteration.

simultaneously, see Figure 2.14. As in crossover, the other genetic operators may switch the sign on the orientation angle of a ply when they are applied, see for example the plies coded as 3 in the ply alteration procedure depicted in Figure 2.14.

2.3 Other Implementation Concerns

Aside from the modifications addressed for the inclusion of multiple material in the laminate stacking sequence, the genetic procedures for both the GA–I and GA–II algorithms are largely the same. However, there are some general implementation concerns pertaining to both versions of the algorithm that need to be discussed and are the focus of this section.

2.3.1 Selecting a Stopping Criterion for the GA

The first issue is the stopping criterion for the genetic algorithm. The genetic search may be stopped after a prescribed number of iterations with no improvement of the top design in the population. This stopping criterion is well suited for measuring the search if an effort to improve the efficiency of the GA is being made. A simpler stopping criterion is to use an upper bound on the total number of function evaluations conducted by the GA. The second stopping criterion may be preferred when conducting a number of independent searches, and simplifies the calculation of the statistics of the GA at the end of the search since each optimization run will have the same number of generations. Both stopping criteria will be utilized in Chapter 4 for analyzing the search characteristics of different implementations of the GA, but will not be compared. The first stopping criterion will be used in Chapter 5.

GA Code

Before ply swap: Orientation - $[0/0/\mathbf{6}/2/5/3/6/1/6/\mathbf{4}]_s$
Material - $[0/0/\mathbf{2}/1/2/2/2/2/1/\mathbf{2}]_s$

After ply swap: Orientation - $[0/0/\mathbf{4}/2/5/3/6/1/6/\mathbf{6}]_s$
Material - $[0/0/\mathbf{2}/1/2/2/2/2/1/\mathbf{2}]_s$

Stacking Sequence

Before ply swap: Orientation - $[E/E/\mathbf{75^\circ}/15^\circ/60^\circ/30^\circ/-75^\circ/0^\circ/75^\circ/\mathbf{45^\circ}]_s$
Material - $[E/E/\mathbf{m2}/m1/m2/m2/m2/m2/m1/\mathbf{m2}]_s$

After ply swap and decoding: Orientation - $[E/E/\mathbf{45^\circ}/15^\circ/60^\circ/30^\circ/75^\circ/0^\circ/75^\circ/\mathbf{-75^\circ}]_s$
Material - $[E/E/\mathbf{m2}/m1/m2/m2/m2/m2/m1/\mathbf{m2}]_s$

Figure 2.14: Modified ply swap operator.

2.3.2 Fitness Calculation

The weight of the laminate can be implicitly or explicitly defined in the objective function for a laminate. In Chapter 4, the weight of the laminate is controlled by a material failure constraint. Although the thinnest laminates will yield the best performance, they are heavily penalized if the material fails under the given loading condition. Thus, the laminates that yield the best performance without failing the material strength constraint will automatically be the lightest (i.e., have the fewest number of plies).

For the multi-objective optimization problem presented in Chapter 5, two objective functions will be utilized. The first objective function will explicitly contain the weight of the laminate by counting the total number of plies in the stacking sequence. A second function will contain information about the manufacturing and material cost of the laminate. The physical weight and cost the of the laminate are then adjusted using information pertaining to the buckling and strength constraint satisfaction of the laminate. The objective functions are then scaled by the corresponding objective functions of a nominal design to ensure that the cost and weight of the laminate are represented accordingly. The overall fitness of the laminate is obtained as a convex combination of the two objective functions. The convex combination can then be adjusted to allow cost and weight to contribute to the fitness calculation in any desired manner. An in-depth discussion is given in Chapter 5.

2.3.3 Methods for Selecting the Next Generation

The remainder of this section will serve as a preface to the next chapter which discusses various methodologies in the selection of the successive generations of laminates in the GA process. In order to ensure that the genetic algorithm continues to search in the direction of an optimal laminate, some information from the past generation is usually carried through to the current generation. Chapter 3 discusses ideas of what and how much information should be passed on. The purpose of exploring these ideas is to determine if the reliability and efficiency of the algorithm can be improved. The quality of the information contained in the population after the GA has converged will also be monitored and will be referred to as the richness of the final population. These alternate selection schemes will be tested in full using the design problem discussed in Chapter 4. Selection schemes yielding the best results in Chapter 4 will be applied to the design problem given in Chapter 5.

Chapter 3

Exploring Alternate Selection Schemes with Genetic Algorithms

An important component of any evolutionary algorithm, including GAs, is the selection scheme used for determining how the next generation of designs will be chosen from the set of designs contained in the parent and child populations. In general, there are two main types of selection schemes: deterministic and probabilistic. Evolutionary strategies (ESs) often utilized deterministic selection rules, evolutionary programming (EP) uses probabilistic selection, and GAs have been found to take advantage of both.

In a deterministic selection scheme, the top P laminates from both the child or combined parent and child populations are placed into each consecutive generation, where P is the size of the parent population. Thus, the least fit members of each population never have an opportunity to advance to the next generation. Probabilistic selection gives each member of the parent and/or child populations a chance at being included in the new population. A simple example of each method will be given here although many different approaches have been explored. For a more in-depth discussion, see Holland [6], Bäck [17], and Schwefel [35].

In a typical deterministic scheme used in ESs, the size of the parent population (P_p) and child population (P_c) are allowed to be different. The next generation is then selected in one of two ways. In the first method, the top P_p designs from the child population are placed into the next generation (for this method, $P_c \geq P_p$). This scheme has been shown to be well suited to changing environment [17], with Schwefel [35] demonstrating that best results are achieved using a ratio of $P_p/P_c = 1/7$. Alternatively, members of both the child and parent populations are ranked together, with the top P_p designs chosen from the combined population. Although the second method prevents the best design from being lost, studies have shown that it may jeopardize the adaptive mechanisms inherent to ESs.

One type of probabilistic scheme that has been used in both EP and GAs is tournament selection. One variation of tournament selection is implemented by combining both the parent and child populations of size P into one population. The i^{th} member in the combined population, X^i_{2P}, is then compared to a randomly chosen subset of designs of size N, where $1 \leq i, N \leq 2P$. Each member is then graded according to how many designs in the corresponding subset have lower fitness values. The top P individuals having the highest scores are passed on to the next generation. As the value of N increases, the best design in the combined population will have an increasing probability of being included in the next generation, resulting in an elitist type of selection scheme.

Although the typical GA often utilizes probabilistic selection procedures [17], many GAs applied to composite laminate design have implemented simple deterministic selection schemes. An elitist method is typically used where the top design from the parent population is always passed to the next generation. As mentioned in the previous chapter, the purpose of the various genetic operators is to produce successively improved populations of designs. However, an elitist method is required because there is no guarantee that there will be a design in the child population that is better than the best design of the parent population.

The focus of this chapter is to try an improve the search capabilities of the GA for the design and optimization of composite laminates by experimenting with alternate selection schemes. The motivation for this work comes from the fact that designs which contain important information in the parent population (other than the top one) are lost after each successive generation when using an elitist selection scheme. If other members from the parent population, which may contain information that is vital for achieving the optimal design, are passed to the next generation, the GA may be able to converge faster and the final population may provide additional good designs other than the optimal.

In Section 3.1, the elitist scheme is described in detail. In Section 3.2, three versions of multiple elitist (ME) selection are discussed where more than just the top design is given a chance to survive into the next generation. In ME selection, the number of top performers passed to each successive generation remains constant throughout the genetic search. In Section 3.3, three versions of a variable elitist (VE) selection scheme are explored where the number of top performers passed to each generation is allowed to vary throughout the search. In the final section of this chapter, criteria for how these different selection schemes will be compared are discussed.

3.1 Elitist selection

An elitist method (designated EL) ranks the child population and parent population of laminates separately. The best laminate from the parent population and the worst laminate from the child population are identified. To create the new population, the best laminate from the parent population replaces the worst laminate from the child population. The EL method is illustrated in Figure 3.1. The potential problem with EL selection is that important genetic information that may exist in other desirable laminates of the parent population is lost, such as laminate B in Figure 3.1. If such information can be identified and preserved, it may help the GA converge in less time, find better laminates, and generate additional laminates in the final population with very good fitnesses.

3.2 Multiple elitist selection

The multiple elitist schemes are designed to preserve more information about good laminate stacking sequences from the parent population than the elitist method. The amount information preserved from the parent population will affect the search qualities of the GA. Preserving small amounts of information from the parent population maintains the GAs ability to search for attractive areas in the design space by allowing large amounts of new information provided by the child population to be introduced to the search process. As the amount of preserved information is increased, the GA becomes more of an exploitative algorithm, rigorously searching the area of the design space occupied by the parent laminates. The explorative properties of the GA are disabled since only a small amount of new information from the child population is being utilized. Three different methods of multiple elitist selection are explored.

3.2.1 ME_1 Selection

In the first multiple elitist selection method, ME_1, the parent and child populations of size P are combined into one population, forming $2P$ laminates. Members of the combined population are

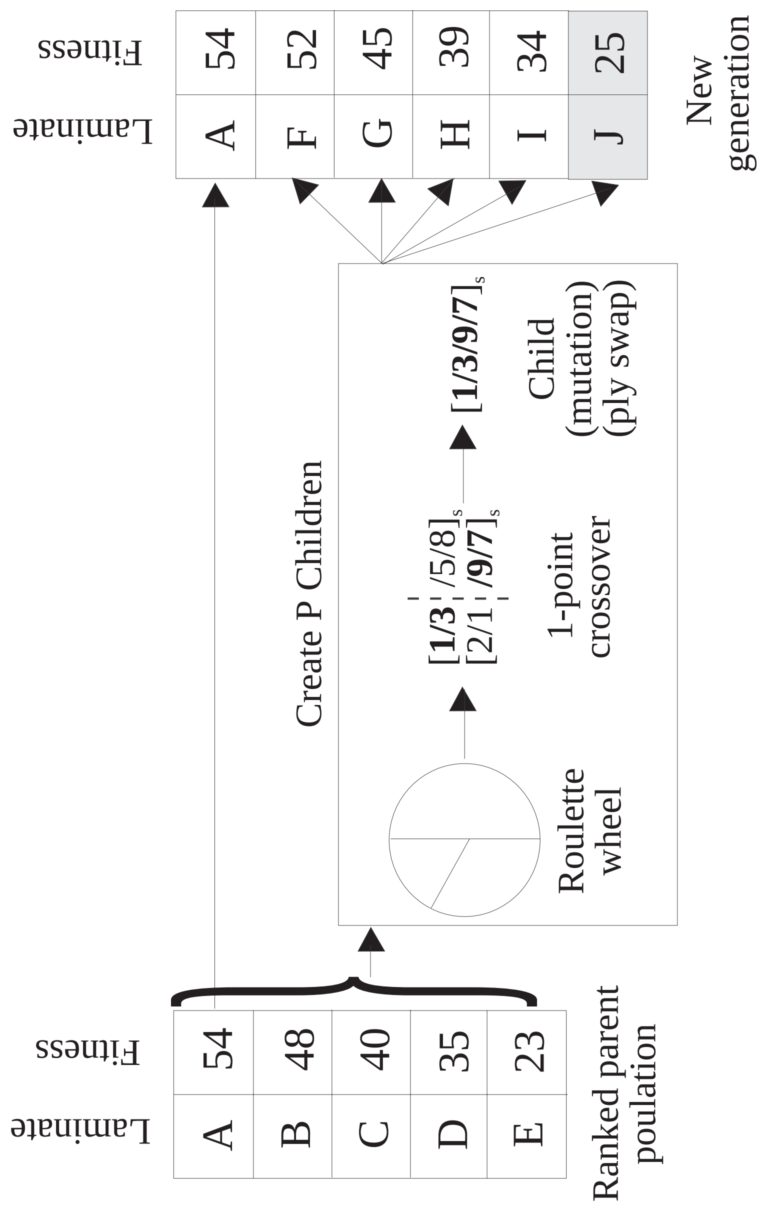


Figure 3.1: Elitist selection ($P = 5$).

then arranged from best to worst according to their fitness values. The number of top laminates from the combined population which will be carried over to the next generation is designated N_k. If N_k is equal to P (the maximum possible value of N_k), then the next generation is filled with the first half of the combined population and the procedure is complete. If N_k is less than P, then the top N_k laminates from the combined population comprise the first part of the new generation.

To fill the remainder of the new generation the ranked child population is searched, starting from the best laminate, for laminates that have not already been passed on to the new generation. The first child found which is not located in the new generation fills the first empty location in the new generation, and successive children fill out the rest of the new generation. The ME_1 selection method is demonstrated in Figure 3.2 for a population size of 5 and $N_k = 3$.

Although the ME_1 method may appear identical to the EL method when $N_k = 1$, this is not necessarily the case. When the top laminate comes from the child population the next generation will consist entirely of child laminates. However, if the top laminate comes from the parent population, then the ME_1 and EL methods are identical.

3.2.2 ME_2 Selection

The second multiple elitist scheme, ME_2, is very similar to the ME_1 scheme. The difference between the two methods is in the way child laminates are added to the new population after N_k laminates from the combined population have been selected. In ME_2 selection, laminates from the portion of the child population not already in the new population will be randomly chosen to fill the remainder of the new population. This method will allow for the possibility of some of the poorer laminates, which may contain important genetic information, from the child population to migrate to the new population. Both ME_1 and ME_2 selection are identical if N_k is set equal to

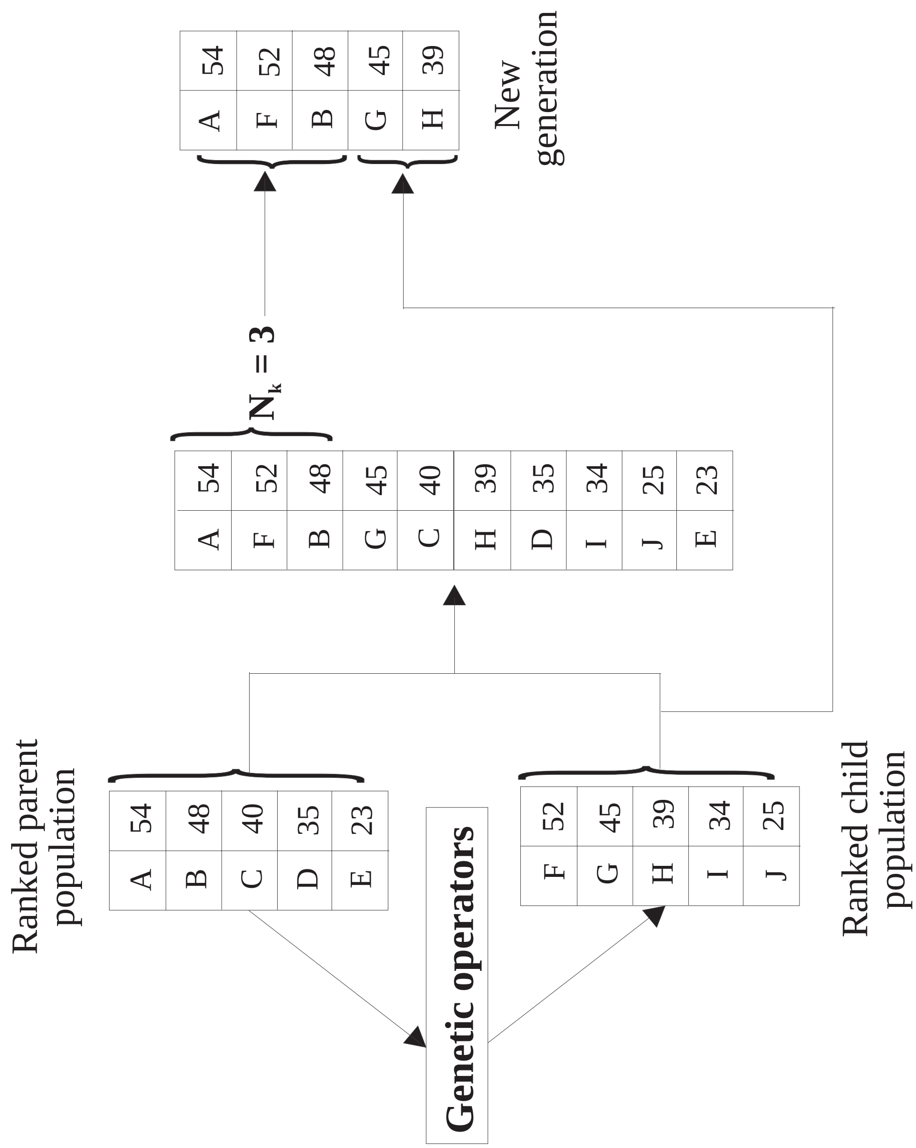


Figure 3.2: ME_1 selection ($P = 5,\ N_k = 3$).

the population size. The ME_2 scheme is shown in Figure 3.3.

3.2.3 ME_3 Selection

The final multiple elitist selection scheme, ME_3, is an extension of the EL method. The top N_k laminates from the population are selected and placed into the new population. The number of child laminates required to fill the remainder of the new population are created from the parent population and placed into the new population. For ME_3 selection, the value of N_k cannot be set equal to the population size since this would code all the parent laminates into the new generation and prevent the GA from exploring the design space. Note that setting $N_k = 1$ in ME_3 selection is a variation of the EL selection method. The ME_3 selection scheme is depicted in Figure 3.4.

3.3 Variable elitist selection

As discussed in the previous section, when small amounts of information are preserved from the parent population in ME selection (i.e., small values are used for N_k), the exploitative and explorative capabilities of the GA are low and high, respectively. These capabilities are reversed as the amount of information utilized from the parent population (N_k) is increased. By varying the value of N_k, the effects of having both a highly exploitative and explorative GA at different stages of the search process can be monitored. Thus, the purpose of the variable elitist selection schemes is to determine what the effects of using multiple values of N_k during an optimization run will have on GA performance.

The VE_1 scheme is a modified version of ME_1 selection. To implement this method, a value of N_k is selected and the optimization process is started using the ME_1 selection scheme. After a certain number of generations without improvement in the fitness of the best laminate, the value of

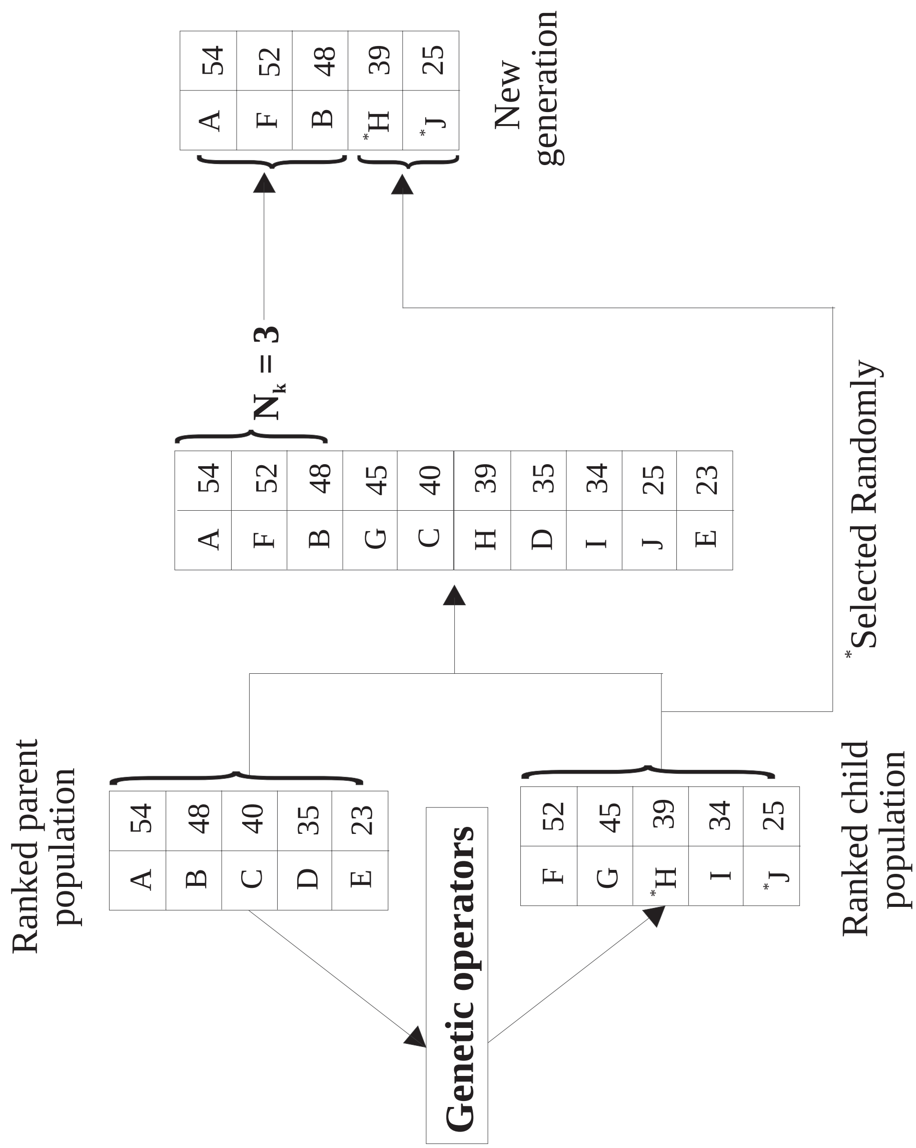


Figure 3.3: ME_2 selection ($P = 5$, $N_k = 3$).

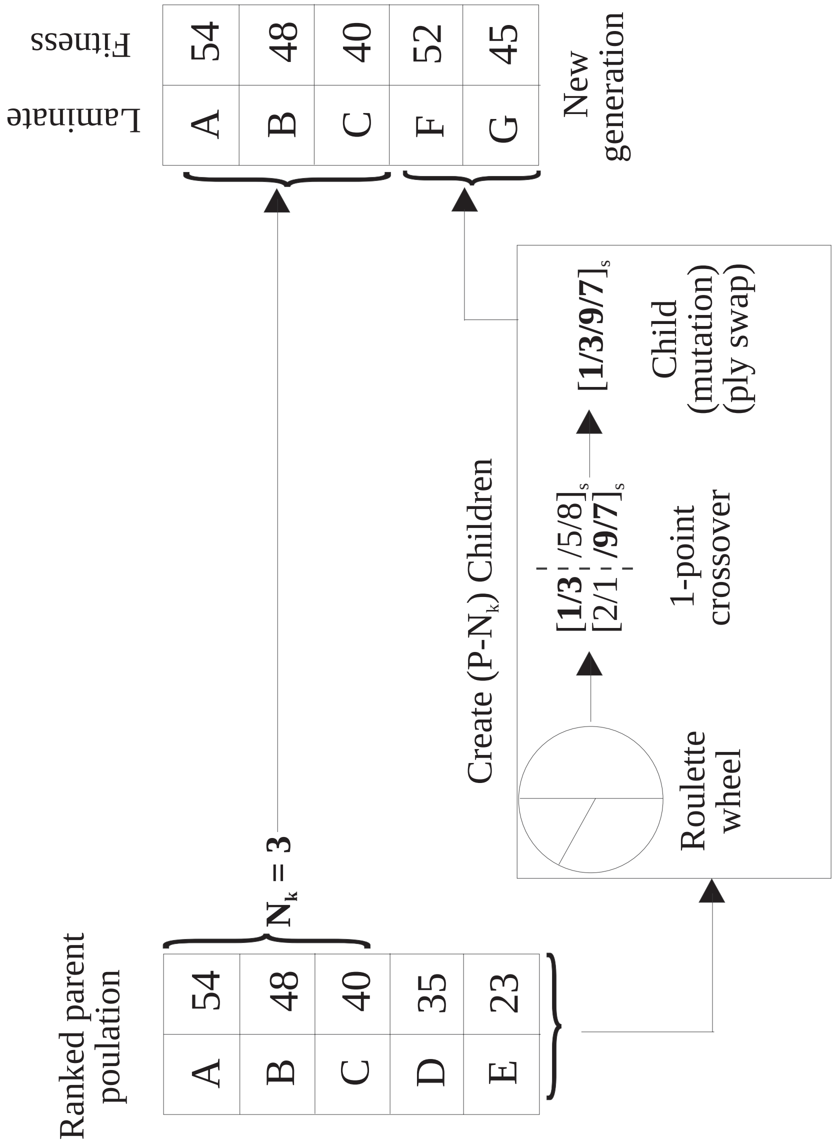


Figure 3.4: ME_3 selection ($P = 5$, $N_k = 3$).

N_k is increased to the value of the population size (the largest possible value) and is held constant until the GA converges. The VE_2 and VE_3 schemes are identical to VE_1 except that the processes are started with the ME_2 and ME_3 selection schemes, respectively. When the value of N_k is changed in the VE_3 scheme, it is increased to a value of $P-1$ instead of P due to the restriction $1 \leq N_k \leq (P-1)$ in ME_3 selection.

3.4 Criteria for Comparing selection methods

To compare the different selection methods, three criteria will be used. The first criterion is apparent reliability which is determined by taking the number of runs that the GA found the best known design and dividing it by the total number of runs conducted. Given n optimization runs with apparent reliability, R, the standard deviation of R is given as

$$\sigma_r = \sqrt{\frac{R(1-R)}{n}}. \tag{3.1}$$

The richness percentage of the final population will be the second criterion. Population richness is calculated by taking the total number of laminates in the final population of each run with objective values within a small percentage of the best known design and dividing it by the product of the total number of runs conducted and the population size.

A final criterion will be the cost of the genetic search. The cost of the algorithm is measured by the average number of analyses per run required for convergence. In this work, an optimization run has converged when a certain number of generations have been recorded without improvement of the fitness of the top laminate in the population. For all selection schemes except the ME_3 and VE_3 selection schemes, the average number of analyses conducted over n optimization runs is determined by

$$A_n = \frac{\sum_{i=1}^{n} X_g^i P}{n}, \tag{3.2}$$

where X_g^i is the total number of generations analyzed in the ith run, and P is the size of the population. The cost calculation for the ME_3 selection scheme is determined by

$$A_n = P + \frac{\sum_{i=1}^{n} (X_g^i - 1) N_c}{n}, \tag{3.3}$$

where N_c is the number of child laminates created in each generation after the initial generation. For the VE_3 selection scheme, the cost is calculated using

$$A_n = P + \frac{\sum_{i=1}^{n} [(X_{gb}^i - 1) N_c + X_{ga}^i]}{n}, \tag{3.4}$$

where X_{gb}^i and X_{ga}^i are the number of generations in the i^{th} optimization run conducted before and after N_k is maximized, respectively. All selection schemes will be compared using the GA–I algorithm for maximization of the twisting displacement of a cantilevered plate, presented in the next Chapter.

Chapter 4

GA Application–I: Maximization of the Twisting Displacement of a Cantilevered Composite Plate

The effectiveness of each of the selection schemes discussed in the previous chapter will be determined in this portion of the research work. The GA–I algorithm will be individually equipped with each selection scheme to maximize the twisting displacement of a cantilevered composite plate subjected to strength and ply contiguity constraints. A description of the problem and the optimization formulation are presented in section 4.1 and section 4.2, respectively. Next, a discussion of the determination of effectiveness of each selection scheme will be provided. Results are presented in section 4.4, starting with some general results. The remainder of the results section will give a more in-depth comparison of the different selection schemes, and will monitor the effects that the population size and the size of the design space have on the optimization capabilities of the GA.

In the final section, some concluding remarks are given along with a discussion of what positive aspects found in this chapter will be used in the optimization problem given in Chapter 5.

4.1 Problem Description

The design problem under consideration consists of a cantilevered plate 14.0 in long by 5.5 in wide and loaded under a pure bending moment ($M_x^a = 1200.0$ lbs) applied along the right end of the panel. The symmetrically laminated composite plate is made of Glass-Epoxy; see Table 4.1 for material properties. Each ply in the stacking sequence may be oriented at any angle between 0 and 90 degrees in increments of 10 degrees as indicated in Figure 4.1.

The analysis used in the GA to determine the bend-twist response of the composite plate is based on classical lamination theory (CLT). Such composite plate problems are often used as simplified models of wing or hydrofoil structures. Aerodynamic loading caused by a wind gust or certain aircraft maneuver are simulated by the applied moment at the tip of the plate, causing the wing to bend and/or twist. As seen in Figure 2.1, ply orientation angles are restricted to positive values only. The result is an unbalanced laminate stacking sequence which will allow the plate to twist as it bends under the influence of the moment applied at the plate tip. By designing the plate for maximum opposite twist at the plate tip, the effect of the aerodynamic loading or maneuver can be reduced or eliminated, as was done in Mallot and Averill [15]

First, to simplify the analysis, the out-of-plane deflection of the plate, $w(x, y)$, is assumed to vary linearly along the y-axis. This assumption eliminates the κ_y curvature and uncouples κ_x from κ_{xy} , which are defined by

$$\kappa_x = -\frac{\partial^2 w}{\partial x^2},$$

Table 4.1: Material properties for Glass-Epoxy

Property	Value	Property	Value
Young's modulus, E_1	5.6×10^6 psi	X_t, tensile strength	3.0×10^4 psi
Young's modulus, E_2	2.0×10^6 psi	Y_t, tensile strength	4.0×10^3 psi
Shear modulus, G_{12}	0.7×10^6 psi	X_c, compressive strength	3.5×10^4 psi
Poisson's ratio, ν_{12}	0.2	Y_c, compressive strength	9.0×10^3 psi
Ply thickness, t	0.011 in	S, shear strength	1.5×10^4 psi

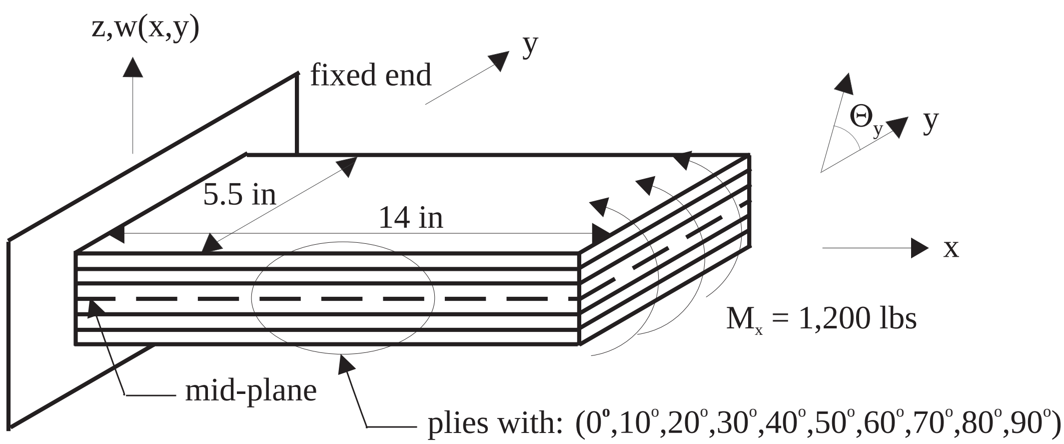


Figure 4.1: Configuration and loading conditions for cantilevered plate.

$$\begin{aligned}\kappa_y &= -\frac{\partial^2 w}{\partial y^2} = 0, \\ \kappa_{xy} &= -2\frac{\partial^2 w}{\partial x \partial y}.\end{aligned} \tag{4.1}$$

By integrating the expressions for the curvatures twice and imposing the boundary conditions

$$w = \frac{\partial w}{\partial x} = \frac{\partial w}{\partial y} = 0 \qquad \text{when} \qquad x = 0 \quad \text{and} \quad 0 \leq y \leq b, \tag{4.2}$$

where b is the width of the plate along the y–axis, an expression for the out-of-plane displacement can be derived as

$$w(x, y) = -\frac{x}{2}\left[x\ \kappa_x + y\ \kappa_{xy}\right]. \tag{4.3}$$

A relationship between the applied moment and the strains and curvatures of the laminate is given using classical lamination theory:

$$\begin{Bmatrix} M_x \\ M_y \\ M_{xy} \end{Bmatrix} = \begin{bmatrix} B_{11} & B_{12} & B_{16} \\ B_{12} & B_{22} & B_{26} \\ B_{16} & B_{26} & B_{66} \end{bmatrix} \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{Bmatrix} + \begin{bmatrix} D_{11} & D_{12} & D_{16} \\ D_{12} & D_{22} & D_{26} \\ D_{16} & D_{26} & D_{66} \end{bmatrix} \begin{Bmatrix} \kappa_x \\ \kappa_y \\ \kappa_{xy} \end{Bmatrix}, \tag{4.4}$$

where $[B_{ij}]$ and $[D_{ij}]$ are the coupling and bending stiffnesses for the laminate, respectively. The symmetry of the stacking sequence eliminates coupling between bending and extension (i.e., $[B_{ij}] = 0$), yielding the expression

$$\begin{Bmatrix} M_x \\ M_y \\ M_{xy} \end{Bmatrix} = \begin{bmatrix} D_{11} & D_{12} & D_{16} \\ D_{12} & D_{22} & D_{26} \\ D_{16} & D_{26} & D_{66} \end{bmatrix} \begin{Bmatrix} \kappa_x \\ \kappa_y \\ \kappa_{xy} \end{Bmatrix}. \tag{4.5}$$

By taking the partial derivative of Eq. (4.3) with respect to y and using Eq. (4.5) for the case of pure bending along the x–axis (i.e., $M_y = M_{xy} = 0$), the twist angle Θ_y (see Figure 4.1) of the plate can be determined by

$$\Theta_y = \frac{\partial w}{\partial y} = -\frac{x}{2}\ M_x d_{16}, \tag{4.6}$$

where $[d] = [D]^{-1}$. The twist angle is a function of x only and has a maximum value at the tip of the plate.

During desing optimization, the Tsai-Hill failure criterion is used to ensure that the material does not fail. Using classical lamination theory once again, the stresses in the kth layer of a laminate under pure bending can be calculated using the relationship

$$\begin{Bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{Bmatrix}^{(k)} = z^{(k)} \begin{bmatrix} \overline{Q}_{11} & \overline{Q}_{12} & \overline{Q}_{16} \\ \overline{Q}_{12} & \overline{Q}_{22} & \overline{Q}_{26} \\ \overline{Q}_{16} & \overline{Q}_{26} & \overline{Q}_{66} \end{bmatrix}^{(k)} \begin{Bmatrix} \kappa_x \\ \kappa_y \\ \kappa_{xy} \end{Bmatrix}. \tag{4.7}$$

To determine the stresses along the principal material directions $(\sigma_1, \sigma_2, \tau_{12})$, the off-axis stresses from Eq. (4.7) are transformed through the orientation angle of the kth ply, $\theta^{(k)}$, using the transformation equations provided by Jones [4]

$$\begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \tau_{12} \end{Bmatrix}^{(k)} = \begin{bmatrix} \cos^2\theta^{(k)} & \sin^2\theta^{(k)} & 2\sin\theta^{(k)}\cos\theta^{(k)} \\ \sin^2\theta^{(k)} & \cos^2\theta^{(k)} & -2\sin\theta^{(k)}\cos\theta^{(k)} \\ -\sin\theta^{(k)}\cos\theta^{(k)} & \sin\theta^{(k)}\cos\theta^{(k)} & \cos^2\theta^{(k)} - \sin^2\theta^{(k)} \end{bmatrix} \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{Bmatrix}^{(k)}. \tag{4.8}$$

The Tsai-Hill criterion for the kth ply in the lamiante, P^k_{th}, is determined by

$$P^k_{th} = \frac{(\sigma^k_1)^2}{X^2} - \frac{\sigma^k_1\,\sigma^k_2}{X^2} + \frac{(\sigma^k_2)^2}{Y^2} + \frac{(\tau^k_{12})^2}{S^2}, \tag{4.9}$$

where X, Y, and S are the longitudinal, transverse, and shear strengths for Glass-Epoxy, respectively. Values for the material strength constants used in Eq. (4.9) depend on whether the ply is in tension or compression, see Table 4.1. Material failure is assumed to occur if Eq. (4.9) yields values greater than one. Note that for a more detailed analysis, the Tsai-Hill criterion would be enforced at the plate root where stresses would be highest.

4.2 Optimization Procedure

The goal of the optimization is to find the laminate with the largest twist angle without laminate failure due to excessive stress. The stacking sequence is also constrained to have no more than three contiguous plies with the same orientation to avoid problems with matrix cracking. To limit the number of design variables, plies will be stacked in groups of three (i.e., $0_3, 10_3, 20_3$, etc.). Thus, a laminate will be penalized if any two contiguous ply stacks have the same orientation. The contiguity constraint will be relaxed at the mid-plane of the plate where the symmetry of the stacking sequence forces the two ply stacks adjacent to the laminate mid-plane to have identical orientations. The GA optimization code works with a string that corresponds to one half of the laminate stacking sequence only, thereby automatically satisfying the symmetry constraint.

The optimization problem can be formulated as:

$$\text{maximize } \Theta_y \quad \text{such that} \quad \max_k \; P_{th}^k \leq 1, \tag{4.10}$$

where P_{th}^k is the value of the Tsai-Hill strength constraint for the kth ply in a laminate. Due to the nature of the problem (maximization of the twisting displacement) the weight of the laminate does not have to be incorporated into the optimization formulation. Large twisting displacements may be achieved by making the laminate very thin. However, as the laminate becomes thin, bending stresses will increase. The minimum weight of a laminate will be governed by the Tsai-Hill strength constraint. If a laminate is too thin, then the material will fail, violating the strength constraint, and make the laminate design undesirable. The thinnest laminates that do not violate any constraint will yield the largest twist angles at the end of the plate and the best performance.

To apply a genetic algorithm the degree of constraint violation must be transformed into penalty parameters which augment the unconstrained objective function [36]. For the problem studied here, the numerical values of the penalty parameters are used to the scale the value of the twist angle at the plate tip.

The augmented function is defined as

$$\Phi = \begin{cases} \dfrac{\Theta_y}{P_c}, & \max\limits_k \; P_{th}^k \leq 1, \\ \\ \dfrac{\Theta_y}{P_c\,(P_{th}^k)^\lambda}, & \max\limits_k \; P_{th}^k > 1, \end{cases} \tag{4.11}$$

where P_c is the ply contiguity constraint and is defined as

$$P_c = 1 + n_c. \tag{4.12}$$

The variable n_c used in Eq. (4.12) is the number of contiguous ply stacks with the same orientation in half the laminate, excluding the two stacks adjacent to the laminate mid-plane. If the strength constraint is not violated, then the first expression in Eq. (4.11) is used to determine the value of the objective function, otherwise, the second expression is used. A laminate is not penalized if the only contiguous ply stacks with the same orientation are the two found at the mid-plane of the laminate (i.e., $n_c = 0$). Laminates that do not violate any constraints will have an objective function value equal to the magnitude of the twist angle at the end of the plate.

For laminates that fail under the given loading condition (i.e., $P_{th}^k > 1$) the twist angle of the laminate is divided by P_{th}^k (k indicating the ply in the laminate which most violates the Tsai-Hill strength constraint). Thus, the greater the degree of constraint violation the greater the laminate is penalized. During testing of the algorithm it was found that very thin laminates appeared desirable even when the strength constraint was being violated. This problem was handled by adding the exponent λ to the scale factor. The value of λ may vary depending on the thickness of a ply stack. For the value of t given in Table 4.1, λ is set to a value of 2.

4.3 Evaluating Selection Scheme Effectiveness

As mentioned in the previous chapter, reliability, richness, and computational cost of the algorithm will be used to compare each selection scheme. To accurately determine these criteria, fifty optimization runs will be conducted for the EL method and for each value of N_k in the multiple elitist and variable elitist methods. Mean values for the computational cost from each set of 50 runs are also reported. A reliability of 90% or greater will be considered acceptable, corresponding to $\sigma_r = 4.2\%$ (see Eq. 3.1). Both the richness percentage and the computation cost of the algorithm will only be considered if the reliability of the algorithm is 90% or greater.

4.4 Results

Results will be presented and compared in four sub-sections to determine the positive and negative aspects of each selection method. The first three sub-sections will discuss results obtained using the design problem discussed in section 4.1. Section 4.4.1 will discuss some general characteristics of the GA's performance using all selection methods. A detailed comparison between the EL, ME, and VE schemes is given in sections 4.4.2 and 4.4.3. Further comparisons between selection schemes will be presented in section 4.4.4, where minor modifications to the design problem are made for purposes of increasing the size of the design space. This will give further insight into the effectiveness of each selection schemes since the GA must work harder when searching for the optimal laminate in the modified problem.

4.4.1 General Results

The maximum number of 3–ply stacks allowed in half the stacking sequence is 15 (30 for the entire laminate). Thus, with 11 choices for each design variable, there are $11^{15} \cong 4 \times 10^{15}$ possible laminate designs. Two optimum design points were found for all runs conducted when using 3–ply stacks in the stacking sequence, see Table 4.2. These designs exhibit significant differences in stacking sequence with only a difference of approximately 1.7% in fitness. Results also showed that as N_k approached the size of the population in the multiple elitist and variable elitist selection methods, the GA's ability to find the optimal design was significantly reduced (i.e., there was premature convergence to the local optimum design). This can be explained by looking at the method of reproduction in the multiple elitist selection. If N_k is set to a large value, then once the GA converges to a particular stacking sequence pattern the majority of the remaining laminates in the population will be saturated with stacking sequences with minor perturbations in ply orientation angle only. This is an undesirable feature of multiple and variable elitist selection since the ability of the genetic operators to explore other regions of the design space is disabled.

For example, if the ME_1 selection scheme is used and N_k is set equal to the population size of 10, a final population of laminates similar to the one shown in Table 4.3 will result if the GA converges to the local optimum design. Consider what happens if the top two laminates, which are most likely to be chosen, are selected as parents. After crossover is applied, the resulting child laminate will have the exact same stacking sequence as the second parent, regardless of the crossover point that is selected. In fact, this phenomenon will occur if any pair of the top six, eighth or ninth laminates are selected as parents. Furthermore, the ply alteration and permutation operators can not alter the stacking sequence enough to guide the GA away from the local optimum. Thus, once the outermost plies of the stacking sequence have been fixed, they can never be significantly altered again. The severity of this problem is exacerbated because the outermost plies have the most influence on the fitness of the laminate, due to the nature of the problem.

The GA is less likely to get trapped at the local optimum design point when EL selection is used, or when N_k is kept small when using multiple elitist or variable elitist selection. The reason for this is that while the top few laminates in the population are preserved, the remaining laminates are free to search the design space for the optimal design (i.e., the opportunity exists for the top laminates to get crossed with other dissimilar designs in the population). The advantage of the VE selection schemes is that a low value of N_k at the beginning of the search allows the GA to converge to a near optimal stacking sequence. When the GA is close to converging, the value of N_k is maximized to rigorously search the design space around the optimum.

It also became apparent that the richness criterion is not of particular importance for the problem being studied here. Laminates that qualified for the richness criterion were simple permutations of the optimal stacking sequence with the best fitness, where the orientation of the inner most ply (which has the least affect on the bend-twist performance of the plate) is switched to other permissible angles. An example of this is shown in Table 4.3. Thus, the final population of an optimization run which found the optimal stacking sequence and had a high richness percentage did not possess other laminates with significantly different design characteristics. Other problems, where many designs with good performance have different design properties, may benefit from high richness percentages. Thus, richness percentages will be given for all runs conducted to give the reader an idea of how alternate selection schemes can increase the richness of the final population.

Initial results presented in this section were generated using a population size of 10. To achieve acceptable reliability and prevent the GA from getting stuck in the local optimum design area, the

Table 4.2: Properties of optimum and local optimum laminate designs using 3–ply stacks.

Optimum Design	$\mathbf{P_{th}^{k}}$	$\mathbf{\Theta_y}$
$[E_8/20_3/10_3/20_3/10_3/20_3/30_3/20_3]_s$	0.9995	2.02°
Local Optimum Design	$\mathbf{P_{th}^{k}}$	$\mathbf{\Theta_y}$
$[E_8/10_3/20_3/30_3/20_3/30_3/90_3/20_3]_s$	0.9968	1.98°

Table 4.3: Convergence to local optimum design, ME_1 selection, $P = N_k = 10$.

Rank	**Laminate**	**Rank**	**Laminate**
1	$[E_8/10_3/20_3/30_3/20_3/30_3/90_3/20_3]_s$	6	$[E_8/10_3/20_3/30_3/20_3/30_3/90_3/70_3]_s$
2	$[E_8/10_3/20_3/30_3/20_3/30_3/90_3/30_3]_s$	7	$[E_8/10_3/20_3/30_3/20_3/30_3/20_3/90_3]_s$
3	$[E_8/10_3/20_3/30_3/20_3/30_3/90_3/80_3]_s$	8	$[E_8/10_3/20_3/30_3/20_3/30_3/90_3/50_3]_s$
4	$[E_8/10_3/20_3/30_3/20_3/30_3/90_3/10_3]_s$	9	$[E_8/10_3/20_3/30_3/20_3/30_3/90_3/60_3]_s$
5	$[E_8/10_3/20_3/30_3/20_3/30_3/90_3/40_3]_s$	10	$[E_8/10_3/20_3/30_3/20_3/30_3/20_3/30_3]_s$

comparison studies given in the next two sub-sections will utilize population sizes of 20 and 40, respectively. This will allow the effect that different population sizes have on the various selection scheme to be studied also. Furthermore, since the performance of the GA using multiple elitist and variable elitist selection was degraded for large values of N_k, its value was varied from one to half the population size only for all results reported in the following sub-sections. The GA operator probabilities used to generate these results are listed in Table 4.4.

4.4.2 Population size: 20

For this section, the stopping criterion for the GA was 300 generations without improvement of the fitness of the top laminate in the population. A comparison of the reliability and richness percentages are shown in Figure 4.2 for the EL and ME selection schemes. Data at the top and bottom of the plot represent reliability and richness respectively. EL selection produced almost perfect reliability, finding the optimal design in 49 of 50 runs. Acceptable levels of reliability could only be achieved for values of $N_k \leq 5$ when using multiple elitist selection methods. For these values of N_k richness percentages were increased by as much as 15 percentage points over elitist selection.

Convergence characteristics of the GA using EL and ME selection for typical optimization runs where the optimal design was found are shown in Figure 4.3. For simplicity, the value of N_k was set to a value of 5 for all ME schemes to try and illustrate any general trends in the convergence characteristics of the GA. A more informative comparison could be presented by conducting several optimization runs over a wide range of values for N_k.

Using a fixed stopping criterion of 300 generations, Figure 4.3 shows the fitness value of the top laminate from the population after each generation. The GA converges rapidly to the optimal

Table 4.4: GA operator implementation probabilities.

Operator	**Probability**
Crossover	1.00
Single ply alteration	0.02
Ply addition	0.05
Ply deletion	0.10
Permutation	0.80

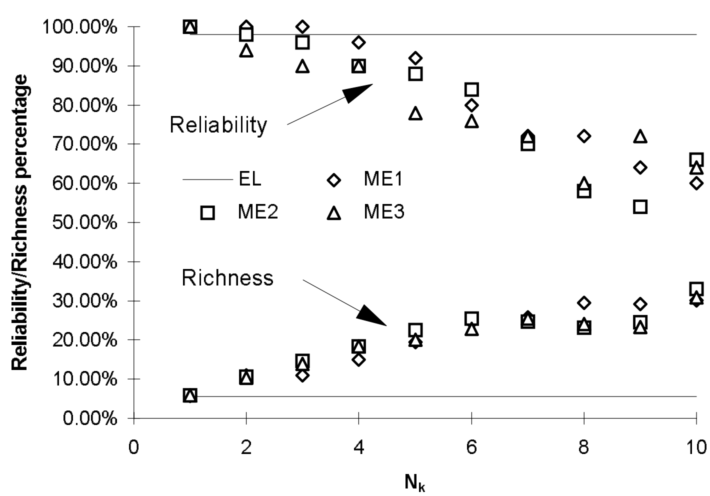


Figure 4.2: Reliability/Richness % comparison – EL vs. ME_1, ME_2, and ME_3 ($P = 20$).

design regardless of the selection scheme used, with the EL and ME_3 selection schemes moving the quickest. By the 30th iteration the GA is within 10% of optimal design for all selection schemes, see Figure 4.3–a). In all cases, the GA converged well before the 300 iterations for this set of runs. The ME_3 selection scheme converged first (Generation #86, Figure 4.3–b), followed by ME_1 (Generation #95, Figure 4.3–b), EL (Generation #120, Figure 4.3–c), and the ME_2 scheme (Generation #127, Figure 4.3–c). An interesting aspect of Figure 4.3 is the behavior of the GA when close to convergence. All selection schemes have a tendency to get stuck in local optimum areas for short periods of time before moving to the next level and eventually finding the optimum design. These consecutive local optimum areas represent small jumps in laminate fitness, indicating that the GA has more difficulty in determining the optimal stacking sequence of the inner plies which have the least influence on the objective function. An example of this can be seen with ME_2 selection in Figure 4.3–b,c.

A cost comparison for the elitist method and each of the multiple elitist selection schemes is depicted in Figures 4.4, 4.5, and 4.6. The dashed lines and errors bars in each figure represent $\pm$ one standard deviation for the 50 runs conducted using the elitist method and the corresponding multiple elitist scheme, respectively. Results using multiple elitist selection show that for acceptable levels of reliability ($N_k \leq 5$ for ME_1, $N_k \leq 4$ for ME_2, $N_k \leq 4$ for ME_3), slightly lower computational costs were achieved for all ME selection methods when looking at the average number of analyses per run.

The ME_3 selection scheme gave the best cost results which makes sense since fewer analyses are required when creating each new generation of laminates as N_k increases. For a value of $N_k = 3$ or $N_k = 4$, ME_3 selection will take less analyses to converge for any given run, than on average for EL selection, and this difference is statistically significant. For all other runs conducted, there

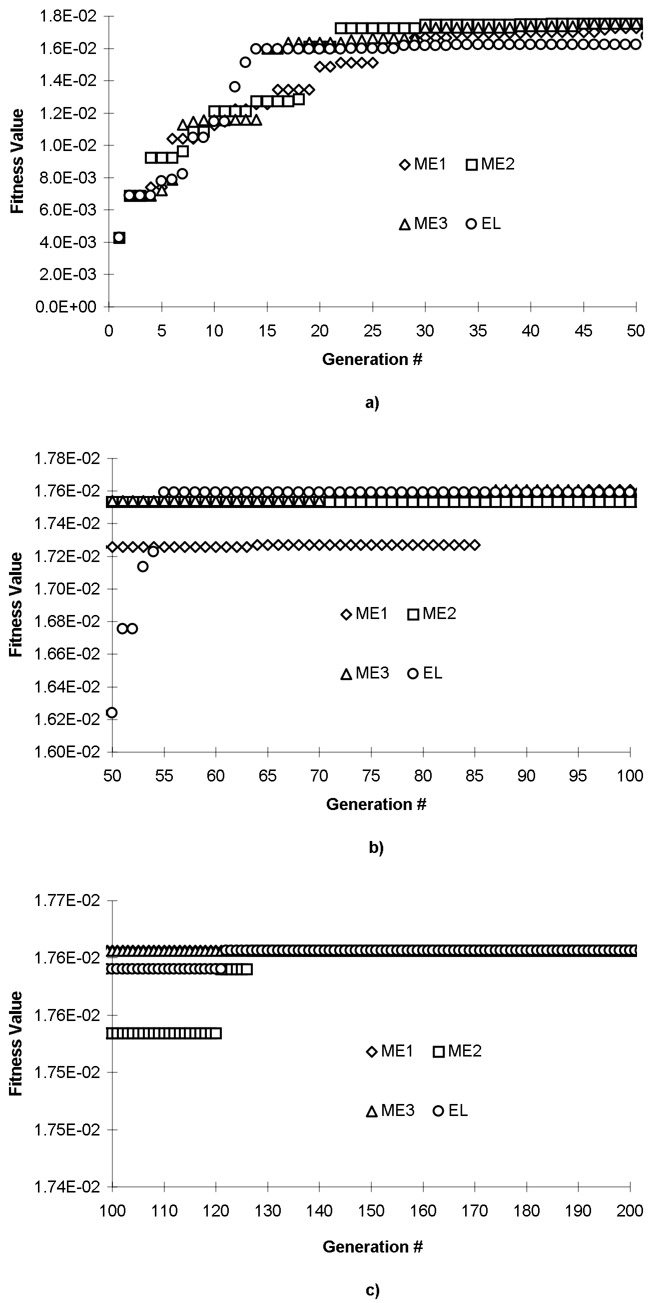


Figure 4.3: GA convergence comparison – EL vs. ME_1, ME_2, and ME_3 ($P = 20$).

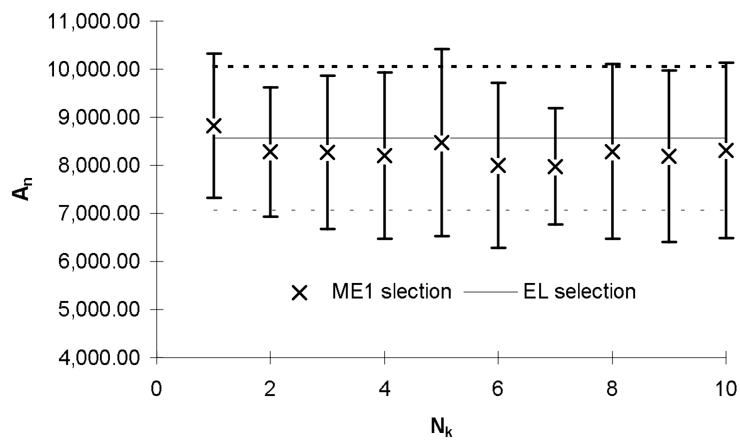


Figure 4.4: Cost comparison - EL vs. ME_1 ($P = 20$).

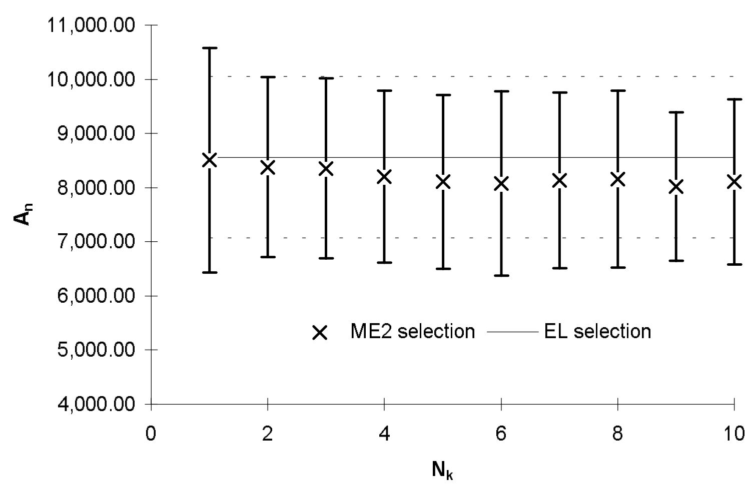


Figure 4.5: Cost comparison - EL vs. ME_2 ($P = 20$).

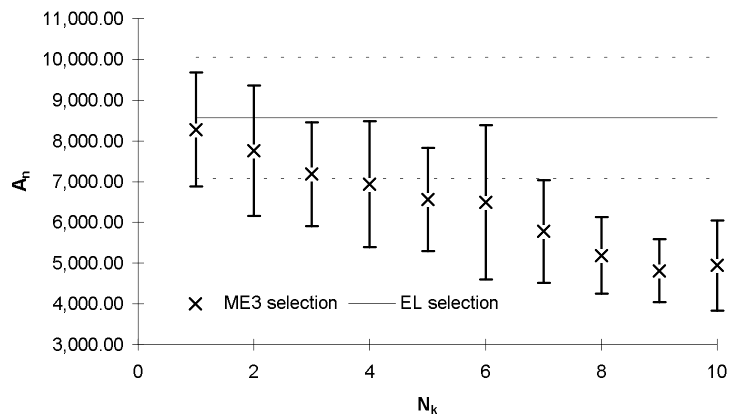


Figure 4.6: Cost comparison - EL vs. ME_3 ($P = 20$).

was essentially no difference between selection methods when looking at the computational cost of the algorithm from a statistical standpoint. To show a clear improvement in the cost of the search, the average number of analyses plus one standard deviation for any alternate selection method should be less than the average number of analyses minus one standard deviation for the elitist method. It is also important to note that by relaxing the convergence criterion to 120 iterations without improvement of the best laminate for the EL selection method, the minimum level of acceptable reliability, 90%, can be achieved on average in approximately 4,700 analyses with a standard deviation of $\pm 1,200$ analyses.

For the VE selection schemes, N_k is increased to the size of the population (the largest possible value) after the 200th iteration is recorded without improvement of the best laminate to ensure that the effectiveness of the GA's search process is not diminished by using large values of N_k early in the search, as was the case in Section 4.4.1. A comparison of the reliability and richness percentages between the EL and VE selection schemes is shown in Figure 4.7. Acceptable levels of reliability are achieved for $N_k \leq 4$ for both the VE_1 and VE_3 selection schemes, and for $N_k \leq 2$ for VE_2 selection. Richness percentages were improved to almost 50% when using either VE_1 or VE_2 while maintaining acceptable levels of reliability. In contrast, richness percentages for VE_3 selection improved slowly as the initial value for N_k increased (similar to ME_3 results), and appear to be unaffected when the value of N_k is set to the maximum value towards the end of the search. This is because the amount of new information being provided to the GA is drastically reduced as N_k increases, with only one child design being created in each successive generation when N_k reaches its maximum value. In VE_1 and VE_2 selection, each new generation is provided with the best laminates from both the parent and child population when N_k is set to the maximum value, yielding high population richness at the end of the search, regardless of what value of N_k is used initially.

Setting N_k to the maximum value after the 200th iteration in the VE schemes had little influence on the genetic search (except for increasing final population richness) because the GA had already found the best design . Thus, convergence characteristics for the VE schemes were identical to those given for ME selection (shown in Figure 4.6), and are not presented here.

The computational costs of the algorithm using EL and VE selection schemes are compared in Figure 4.8, Figure 4.9, and Figure 4.10. Slight improvement in the average computational cost of an optimization run is achieved when using VE_1 selection (Figure 4.8), while no improvement was gained using the VE_2 scheme (Figure 4.9). Best results are seen with the VE_3 scheme, see Figure 4.10. When N_k is set to a value of 2, the average number of analyses plus one standard deviation for VE_3 is slightly larger than the average number of analyses minus one standard deviation for EL selection (standard deviation for the EL scheme is ± 1487 analyses). Low cost figures for the VE_3 scheme are achieved because only one child is being created in each successive generation (i.e., one laminate analysis is required per generation), after the value of N_k is set to the maximum value. However, creating only one laminate per generation substantially reduces the capabilities of the genetic search, which in turn increases the probability of having the GA converge prematurely. This phenomenon was not seen in this problem because the optimal design was found well before N_k was set to its maximum value.

4.4.3 Population size: 40

In this section, a comparison between the EL, ME, and VE selection schemes is given using a population size of 40. The stopping criterion was not modified from the previous section and N_k was once again set to its maximum value after 200 iterations without improvement of the best

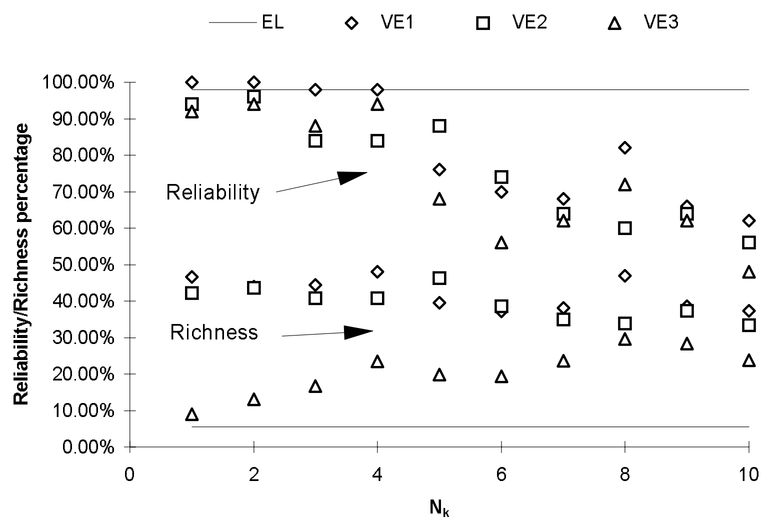


Figure 4.7: Reliability/Richness % comparison – EL vs. VE_1, VE_2, and VE_3 ($P = 20$).

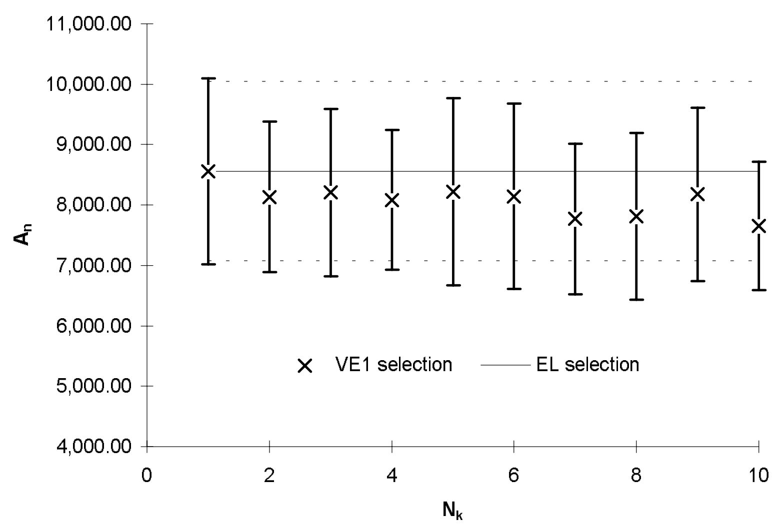


Figure 4.8: Cost comparison - EL vs. VE_1 ($P = 20$).

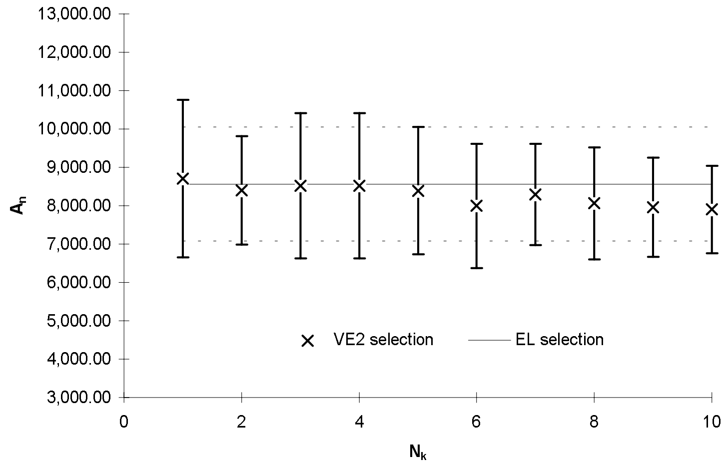


Figure 4.9: Cost comparison - EL vs. VE_2 ($P = 20$).

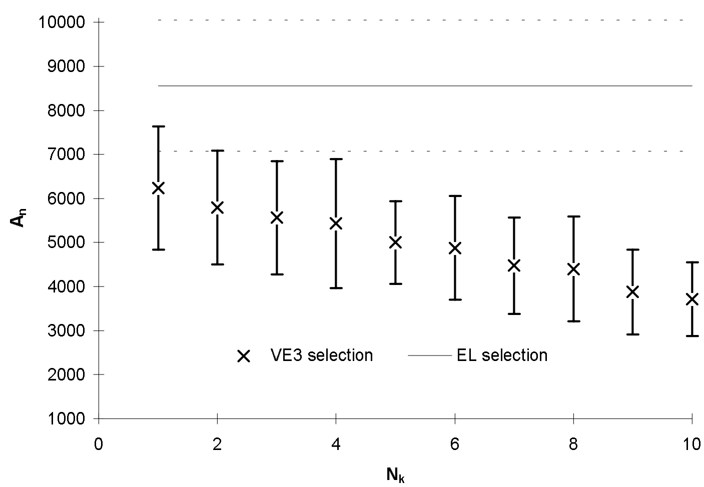


Figure 4.10: Cost comparison - EL vs. VE_3 ($P = 20$).

laminate for VE selection.

Further improvements in reliability and richness percentages are realized for the ME during this set of optimization runs, see Figure 4.11. This makes sense since the larger population size gives the GA access to more information when searching for the optimal laminate. The EL selection found the optimal laminate in every optimization run, while acceptable levels of reliability are achieved for values as high as 14 for ME_1 selection, 12 for ME_2 selection, and 10 for ME_3 selection. Richness percentages obtained using ME selection are improved to 30% (ME_1, $N_k = 14$) compared to 5.5% for the elitist selection scheme.

Convergence characteristics for optimization runs which found the optimal design are shown in Figure 4.12 for the EL and ME selection schemes. For a population size of 40, N_k was set to a value of 10 for the ME schemes. Convergence to within 10% of the optimum design is rapid once again, with the ME_2 scheme taking only 11 generations. Both the ME_1 and ME_2 selections schemes converge first, requiring 42 iteration each, see Figure 4.12–a. The EL selection schemes converged after the 65th generation (Figure 4.12–b), and the ME_3 scheme required 188 generations to converge in its optimization run (Figure 4.12–c). One again, the GA has difficulty in determining the optimal stacking sequence of the inner plies by getting stuck in local optimum areas of the design space, especially with the ME_3 selection scheme, see Figure 4.12–c).

Improvements are also achieved using ME selection when comparing the computational cost of the algorithm for the larger population size, see Figure 4.13, Figure 4.14, and Figure 4.15. While minor improvements were achieved with the ME_1 and ME_2 selection, the ME_3 scheme showed the best results once again, with approximately 4,000 fewer analyses required (on average) for convergence compared to the EL scheme when N_k is equal to 8. However, for a minimum acceptable level of reliability of 90%, the EL scheme only requires approximately 5600 analyses on

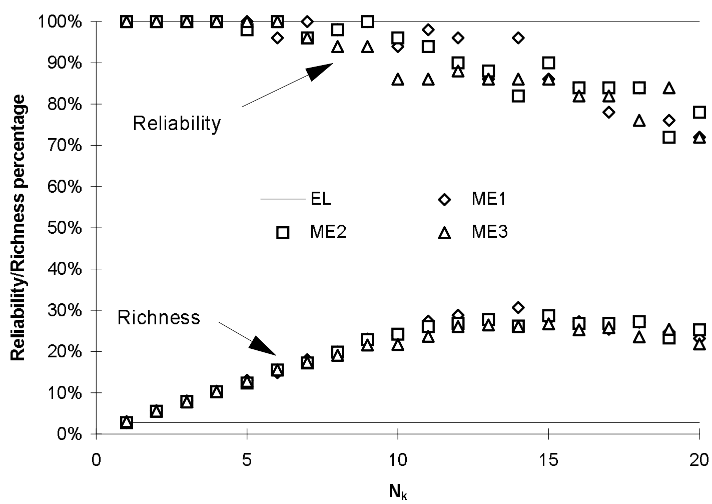


Figure 4.11: Reliability/Richness % comparison – EL vs. ME_1, ME_2, and ME_3 ($P = 40$).

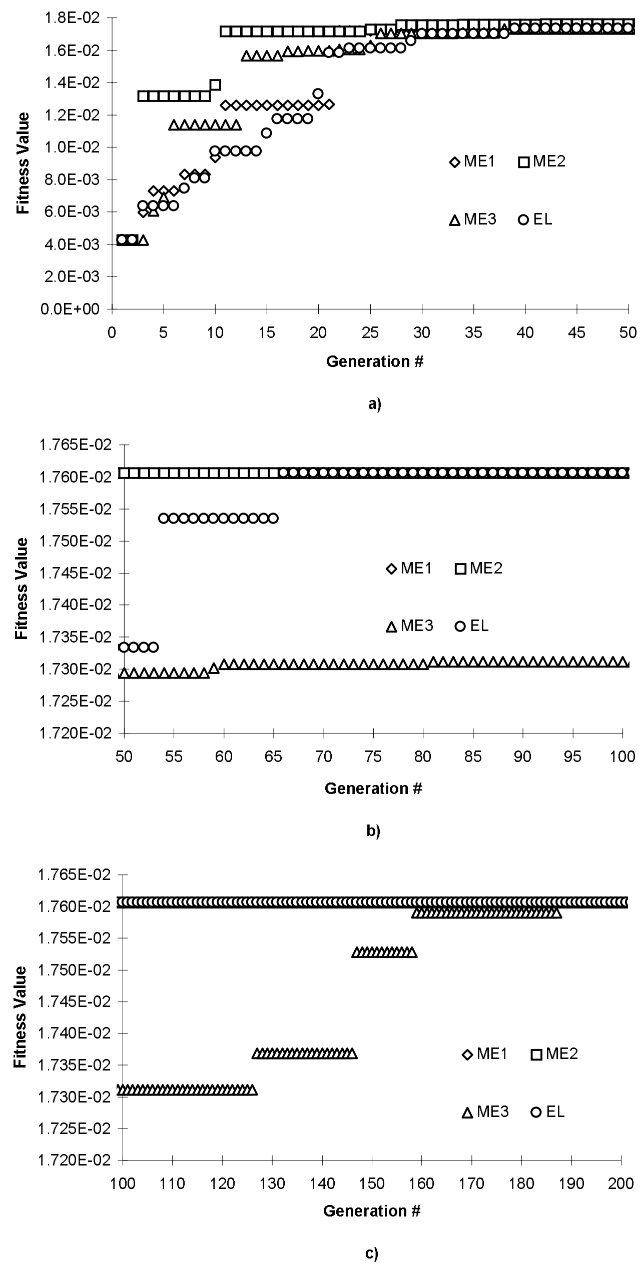


Figure 4.12: GA convergence comparison – EL vs. ME_1, ME_2, and ME_3 ($P = 40$).

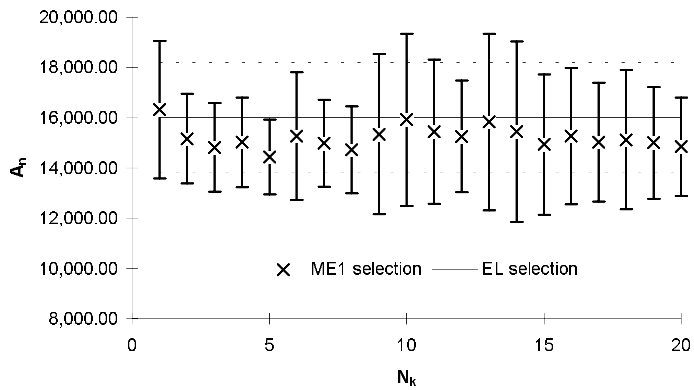


Figure 4.13: Cost comparison - EL vs. ME_1 ($P = 40$).

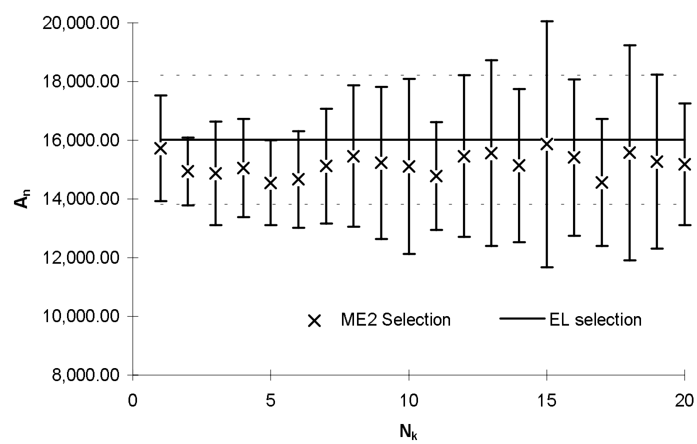


Figure 4.14: Cost comparison - EL vs. ME_2 ($P = 40$).

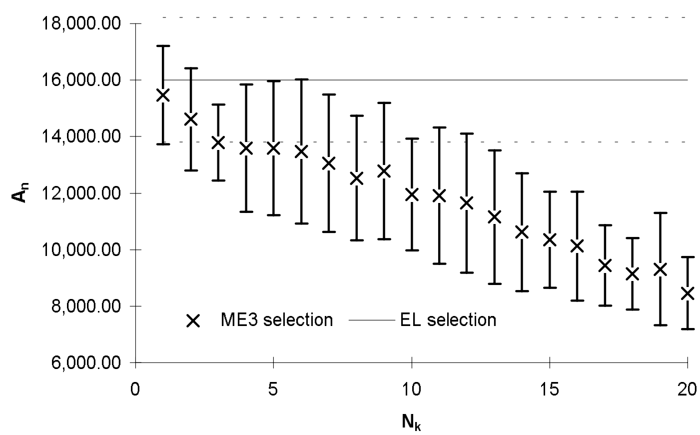


Figure 4.15: Cost comparison - EL vs. ME_3 ($P = 40$).

average to converge with a standard deviation of $\pm 1,197$ analyses.

Reliability and richness percentages for the EL and VE schemes are shown in Figure 4.16 using a population size of 40. For values of $1 \leq N_k \leq 10$, richness percentages as high as 30% were achieved for the VE schemes while maintaining acceptable levels of reliability. Similar to the results found for a population size of 20, richness percentages for the VE_1 and VE_2 schemes remain relatively constant with N_k and steadily increases with the value of N_k for the VE_3 scheme. Note also that richness percentages obtained for a population size of 40 are lower when compared to results for a population size of 20 when using VE selection. This is because the total number of laminates used to calculate the richness percentage increases with an increase in population size, while the number of laminates with 0.5% of the fitness of the optimal laminate remains constant.

Cost comparisons for VE selection and a population size of 40 are presented in Figure 4.17, Figure 4.18, and Figure 4.19. VE_3 selection shows the best results once again, requiring as few as 8500 analyses for convergence when N_k is 10, see Figure 4.19. By adding one standard deviation to the average number of analyses (A_n) for VE_3 ($N_k = 10$) and subtracting one standard deviation from A_n for EL (the standard deviation for EL in this case is $\pm 2,200$ analyses), the VE_3 scheme still requires 3,500 fewer analyses to converge. However, as explained in the previous sub-section, the only way the VE_3 scheme can produce acceptable levels of reliability at low costs is by finding the optimal design before N_k is set to the maximum value (this also explains the similarities with the ME_3 results). If this is not the case, then the reliability of the algorithm and the chances of the GA converging are reduced. Thus, since the VE_3 scheme would most likely produce unacceptable results for problems where the GA cannot find the optimal design before N_k is maximized, the improvements in cost are insignificant.

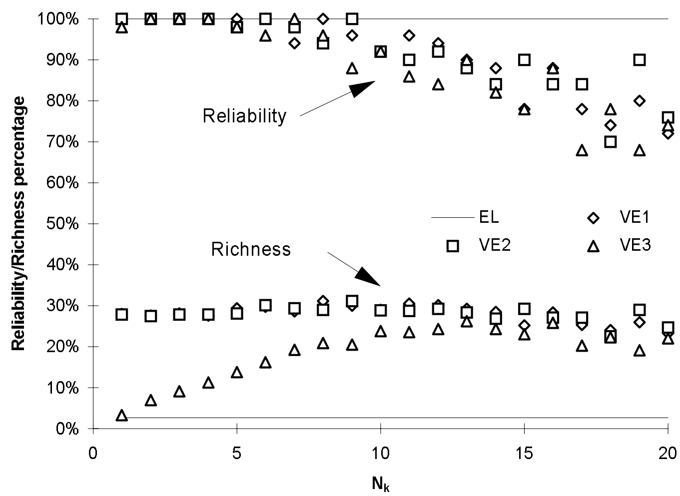


Figure 4.16: Reliability/Richness % comparison – EL vs. VE_1, VE_2, and VE_3 ($P = 40$).

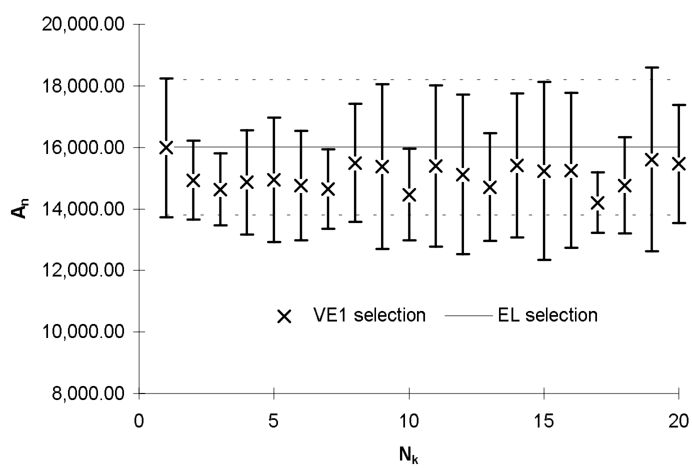


Figure 4.17: Cost comparison - EL vs. VE_1 ($P = 40$).

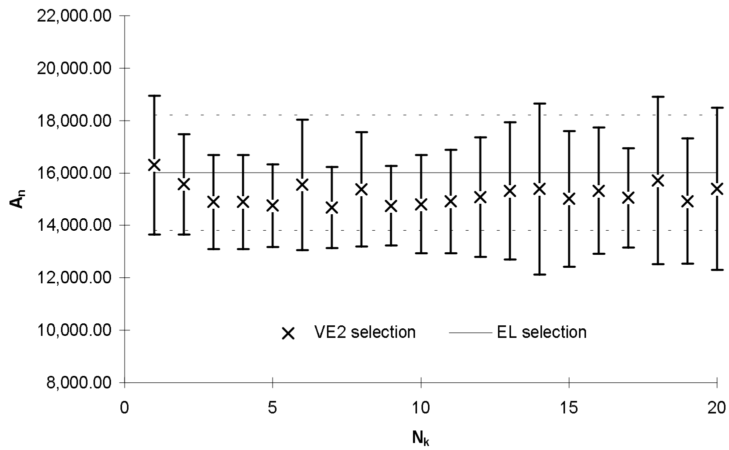


Figure 4.18: Cost comparison - EL vs. VE_2 $(P = 40)$.

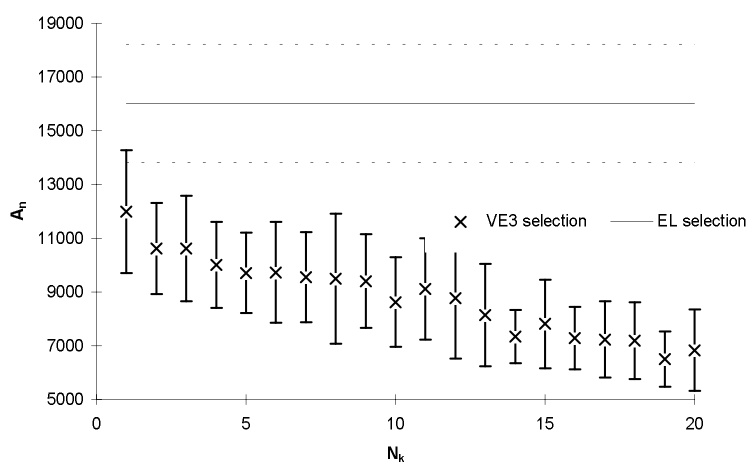


Figure 4.19: Cost comparison - EL vs. VE_3 $(P = 40)$.

4.4.4 Increased Design Space

For the design problem described earlier, the size of the design space was artificially reduced by creating stacks of 3 plies. In this section we remove that limitation and allow each ply to assume one of the ply orientations listed in Figure 2.1. With this modification, the maximum number of single plies in half the laminate stacking sequence that any laminate can have is increased to 30 (60 for the entire laminate) giving $11^{30} \cong 1.74 \times 10^{31}$ possible laminate designs. The ply contiguity constraint, on the other hand, was kept the same requiring no more than 3 plies of the same orientation to appear adjacent to one another. Therefore, the GA implementation of this constraint was modified.

The optimal stacking sequence, strength constraint, and twist angle for the best known design in the modified problem are given in Table 4.5. By using single plies in the stacking sequence, the thickness of the optimal laminate for the modified problem is two plies thinner than the optimal design found when using 3-ply stacks. This is because laminate thickness can decrease in increments of 0.011 inches (by dropping a single ply) instead of 0.033 inches (by dropping a 3-ply stack). The thinner laminate increased the twist angle of the plate by 6% compared to results generated using 3–ply stacks.

Due to the large number of possible laminate designs, the size of the population was increased to 200 in an effort to obtain acceptable levels of reliability. Fifty optimization runs were conducted once again to compare both selection schemes. Figure 4.20 shows a comparison of reliability and richness percentages for the EL and ME selection methods, with N_k taking on odd values from 1 to 100 for ME selection (i.e., 1,3,5,....etc.). The EL scheme found the optimal stacking sequence only once in 50 runs. Reliability percentages for ME selection are lowest for the smallest values of N_k which makes sense since the characteristics of the EL and ME schemes are most similar in

Table 4.5: Properties of optimum laminate design using single-ply stacks.

Optimum Design	$\mathbf{P_{th}^{k}}$	$\mathbf{\Theta_y}$
$[E_{10}/(10/20)_2/10_3/20/10_2/20_2/10_3/20/10_2/20/10]_s$	1.0000	2.15°

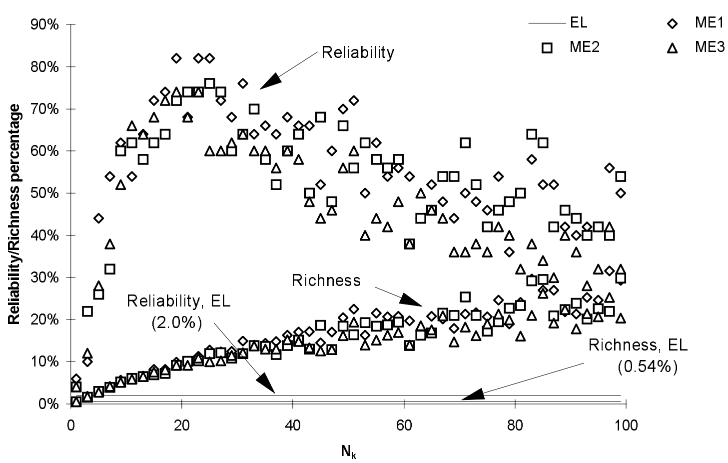


Figure 4.20: Reliability/Richness % comparison – EL vs. ME_1, ME_2, and ME_3 ($P = 200$).

these cases. However, as N_k increases to values between 15 and 30, reliability percentages increase substantially with ME_1 ($N_k = 19$) peaking at 82%. Richness percentages were substantially higher for the ME schemes also.

These results are illustrated further in Figure 4.21 and Figure 4.22 which show the number of laminates from each final population of the 50 runs that have fitnesses within a certain percentage of the optimal design. The EL scheme can only find approximately 100 laminates that are within 0.05% of the optimal design, see Figure 4.21. In contrast, Figure 4.22 shows that the ME_1 selection scheme ($N_k = 19$) finds 150 laminates that match the fitness of the optimal laminate and approximately 750 laminates with fitnesses that are within 0.05% of the optimal. Thus, the EL scheme has much more difficulty with problems where the optimal design is surrounded by many near optimal designs (as is the case here), and explains the large discrepancy in reliability percentage with the ME selection schemes. Furthermore, in runs where the GA finds the optimal laminate, other designs are also found with fitnesses that are good enough to be considered optimal according to the accuracy of the fitness calculation. This is why reliability is 82% and not 100% for ME_1 selection when N_k is 19, even though 150 optimal laminate designs were found.

Convergence characteristics for the EL and ME schemes are shown in Figure 4.23. The stopping criterion for this set of runs was fixed at 700 iterations. The value of N_k was set to 25 for all ME selection schemes. The EL scheme takes the largest number of generations to come within 10% of the optimal design (Figure 4.23–a) and cannot search through the local optimum areas to find the best design in the 700 generations allotted (Figure 4.23–c). For the ME schemes, the GA requires slightly less time to reach the optimal area of the design space (Figure 4.23–a) and then steps its way through the local optimum areas before reaching the best design (Figure 4.23–b and Figure 4.23–c). ME_1 converged on the 68th generation, ME_2 on the 89th generation, and ME_3

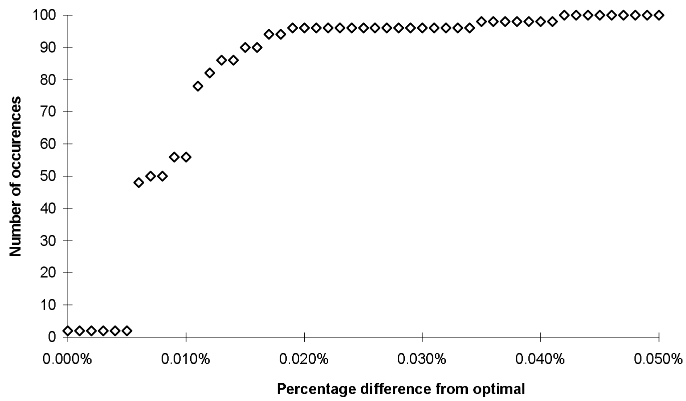


Figure 4.21: Number of laminates from final population within X% of optimal – EL selection (50 runs, $P = 200$).

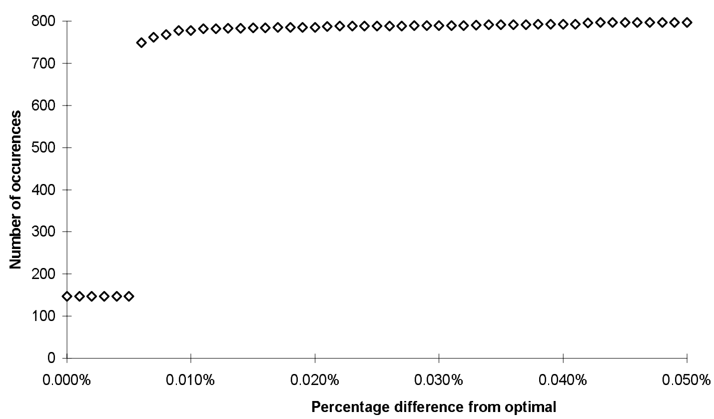


Figure 4.22: Number of laminates within X% of optimal – ME_1 selection (50 runs, $P = 200$, $N_k = 19$).

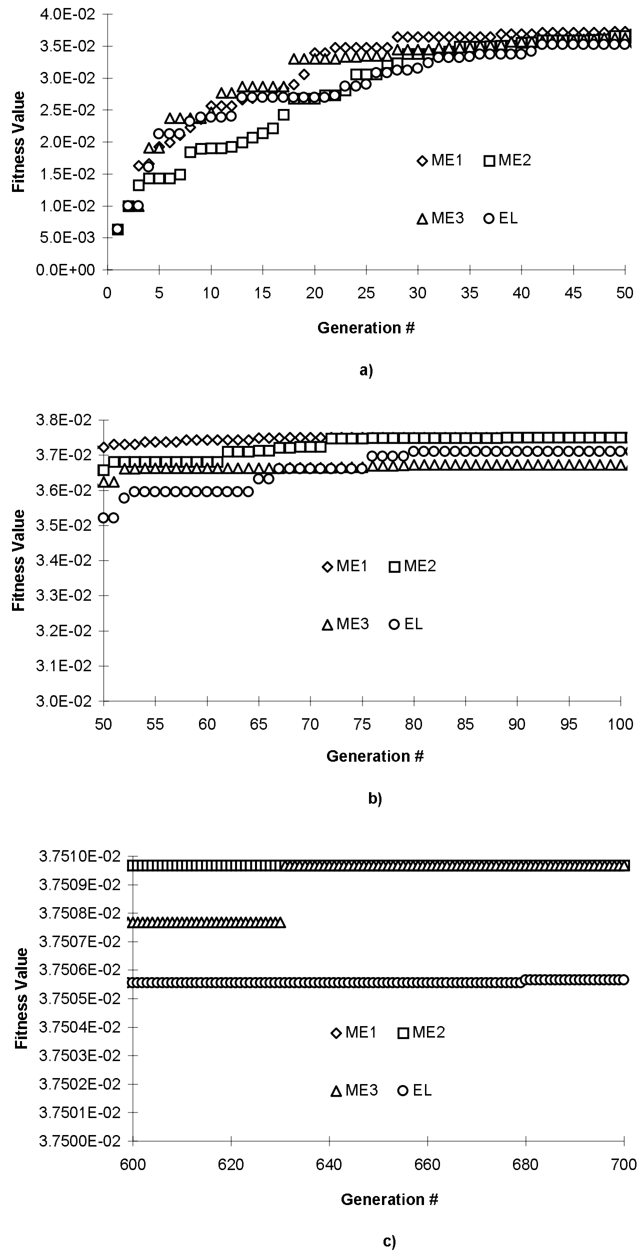


Figure 4.23: GA convergence comparison – EL vs. ME_1, ME_2, and ME_3 ($P = 200$).

on the 630th generation. Thus, the exploitative characteristics inherent to the ME schemes for $15 \leq N_k \leq 30$ not only increase richness percentages, but help the GA find the optimum design more often, thereby increasing reliability.

Since reliability was so low for EL selection, a comparison of the average computational cost was established between the ME selection schemes only. As discussed earlier, the characteristics of the genetic search change from explorative to more of an exploitative nature as the value of N_k increases. The exploitative GA tends to converge prematurely more often, resulting in the reliability drop seen in Figure 4.20 and corresponding drop in computational cost, see Figure 4.24. Thus, highest reliability percentages obtained using ME_1 selection are achieved at the expense of slightly larger computational costs. As expected, the ME_3 scheme had the lowest costs, but was not as effective as the ME_1 and ME_2 schemes in finding the optimal laminate.

To implement the VE schemes for this problem, the value of N_k was increased to the maximum value after 600 iterations without improvement of the best design. The same stopping criterion used for the EL and ME comparison was maintained here. Reliability percentages reached as high as 98% when using VE_1 selection ($N_k = 1$), see Figure 4.25, an improvement of over 90 percentage points when compared to ME_1 selection for $N_k = 1$. Thus, for problems where many local optimum designs are located around the optimal, utilizing both a highly explorative search at the beginning ($N_k = 1$), and a more focused search towards the end of the optimization run ($N_k = P = 200$) is highly beneficial to the GA when trying find the best design. Similar results were achieved using the VE_2 scheme also.

Once again, the VE_3 scheme achieved reliability and richness percentages very close to those found using ME_3 selection (this was the case in the previous two sub-sections as well). When the initial value for N_k is small, VE_3 selection behaves in a similar fashion to the EL scheme, with

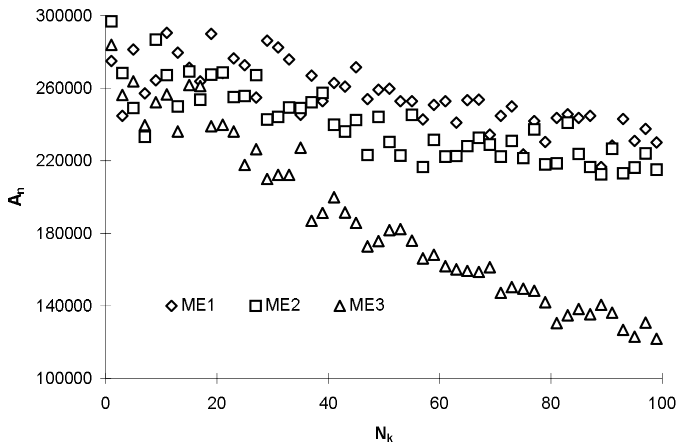


Figure 4.24: Cost comparison – ME_1, ME_2, and ME_3 ($P = 200$).

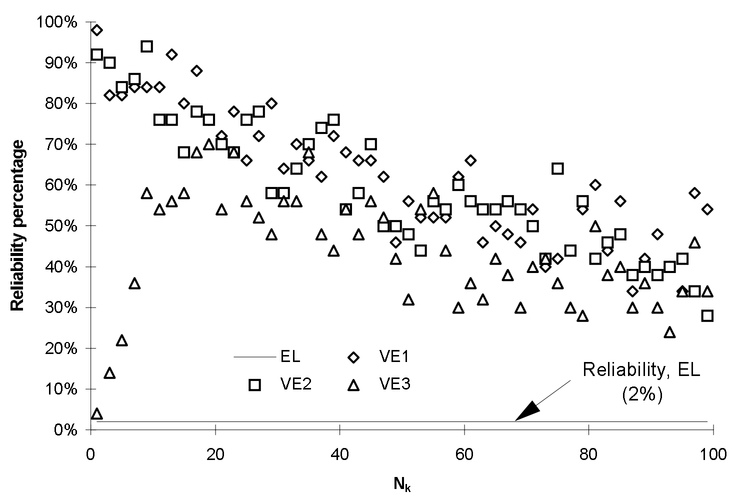


Figure 4.25: Reliability % comparison – EL vs. VE_1, VE_2, and VE_3 ($P = 200$).

only the top few designs in the population possessing good qualities. After N_k is maximized, the genetic search is disabled because the GA is left with a population comprised mostly of poor designs that can depend on only one new child per generation to advance the search towards the optimal. Furthermore, the chances of creating a child with good performance are reduced because a large population that possesses only a few good laminates increases the possibility for poorer designs to be selected as a parents (i.e., the portion of the roulette wheel given to each potential parent laminate is based on the population size and laminate rank only, not a laminate's fitness value). Thus, for small values of N_k, reliability percentages attained using VE_1 and VE_2 selection could not be achieved with the VE_3 scheme. The higher reliability percentages achieved with the VE_3 scheme shown in Figure 4.25 are based solely on the initial value of N_k, which determines the amount of information kept from past generations during the search.

Richness percentages for the VE schemes are shown in Figure 4.26. The robust nature of the VE_1 and VE_2 schemes is illustrated here, achieving 100% richness for N_k = (1,2,3, or 5), showing that there are numerous local optimum designs for this problem. As the value of N_k increases, the richness percentages decreases along with reliability since the optimal design is being found on fewer occasions. As expected, VE_3 selection was substantially less effective then the other VE schemes, producing similar richness percentages to those obtained with the ME_3 selection scheme.

Plots showing the number of optimal and near-optimal designs that are found by the GA using the VE_1 and VE_3 schemes are shown in Figure 4.27 and Figure 4.28, respectively. The VE_1 scheme ($N_k = 1$) found the optimal design(s) over 250 times, and found laminates with fitness no more than 0.015% from optimal almost 10,000 times during a 50 run period. In contrast, the VE_3 scheme ($N_k = 20$) found the optimal design(s) approximately 100 times, and found 800 laminates that were within 0.015% of optimal. The VE_2 scheme produced similar results to those found using VE_1

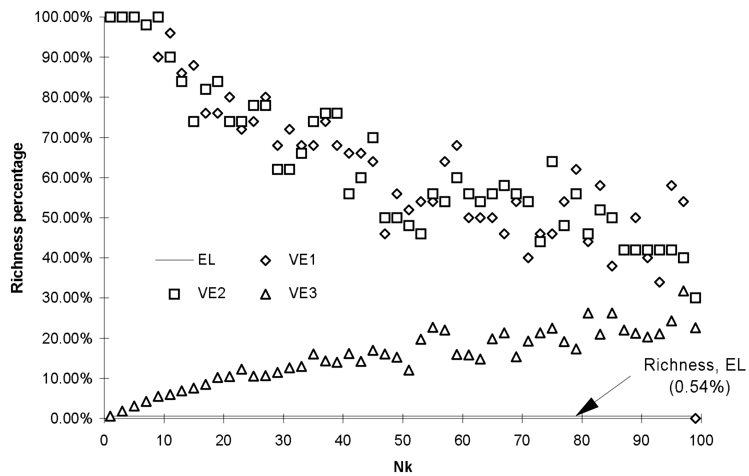


Figure 4.26: Richness % comparison – EL vs. VE_1, VE_2, and VE_3 ($P = 200$).

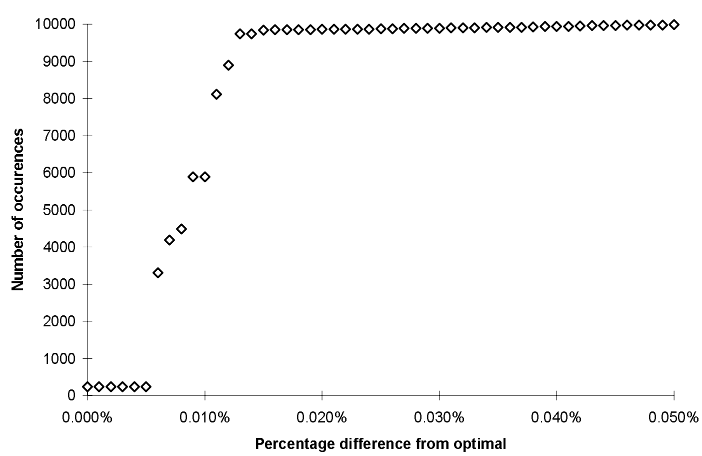


Figure 4.27: Number of laminates within X% of optimal – VE_1 selection (50 runs, $P = 200$, $N_k = 1$).

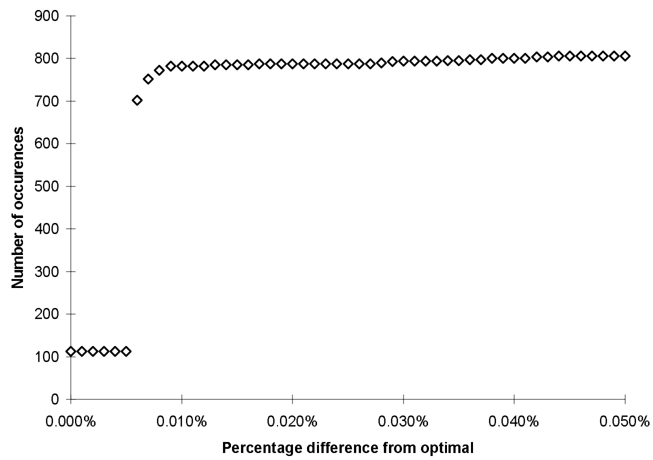


Figure 4.28: Number of laminates within X% of optimal – VE_3 selection (50 runs, $P = 200$, $N_k = 20$).

selection and thus, are not shown here.

Convergence characteristics for the modified problem are shown in Figure 4.29 using the EL and VE selection schemes. The GA has little difficulty advancing towards the optimal design, and is within 10% of the best design after the 80th generation regardless of the selection scheme used (Figure 4.29–a, b). However, after the 100th generation, the GA seems to get trapped in local optimum areas, with the EL and VE_3 schemes unable to find the best design in the 700 generation allotted. The VE_1 and VE_2 schemes also show difficulty in converging until the 600th generation is reached, and the value of N_k is maximized. From this point, the increased robustness of the search helps the GA find the optimal after an additional 26 generations using the VE_2 scheme and 58 generation using VE_1 selection, see Figure 4.29–c.

A cost comparison between VE schemes is given in Figure 4.30. The high reliability and richness percentages obtained for small values of N_k using the VE_1 and VE_2 schemes are paid for by a large increase in the computational cost of the algorithm. The cost is higher for VE_1 and VE_2 selection because the GA often found the optimal design after the 600th generation (towards the end of the search, when N_k was maximized). This extends the length of the genetic search considerably since the stopping criterion is re-initialized each time the GA finds an improved design. For example, if the optimal design is found on the 650th generation, the stopping criterion is reset to zero, requiring another 700 generations before the GA is shut down. Thus, it was of interest to see if the cost could be reduced by increasing N_k to the maximum value earlier in the search without sacrificing reliability. Table 4.6 shows results for VE_1 selection by setting N_k to a value of 1 initially, and then maximizing it after 200 iteration without improvement. These modifications reduced the cost of the search by over 26% while maintaining an acceptable level of reliability and 100% final population richness.

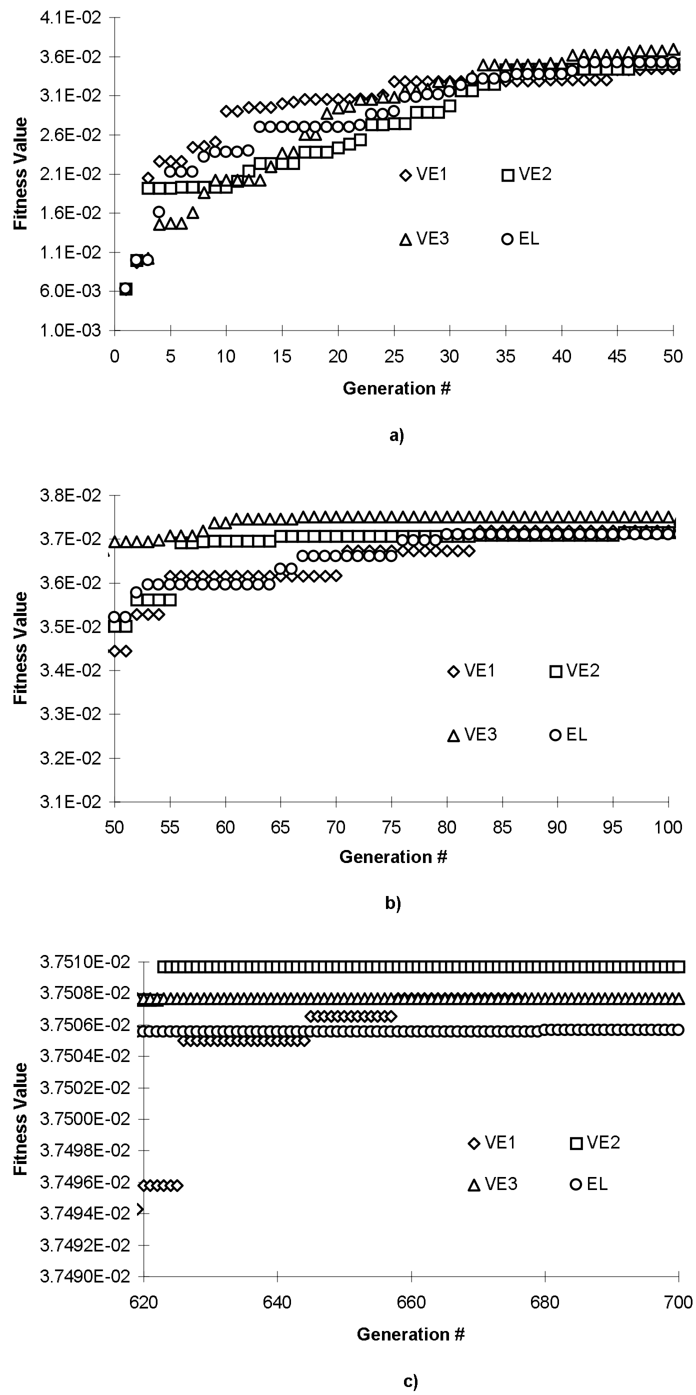


Figure 4.29: GA convergence comparison – EL vs. VE_1, VE_2, and VE_3 ($P = 200$).

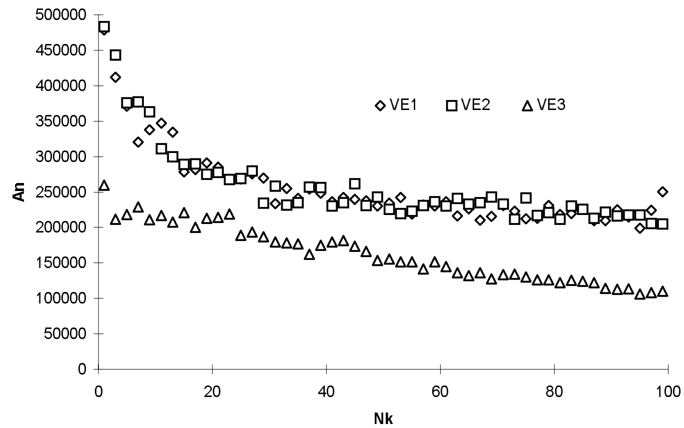


Figure 4.30: Cost comparison – VE_1, VE_2, and VE_3 ($P = 200$).

Table 4.6: Run characteristics for different implementations of VE_1 selection ($N_k = 1$).

Property	$\mathbf{N_k}$ **maximized after 600 iterations**	$\mathbf{N_k}$ **maximized after 200 iterations**	**Percent difference**
A_n	478,964.0	352,204.0	-26.5
Standard deviation	112,965.0	75,051.4	-33.5
Reliability	98.0%	94.0%	-4.1
Richness	100%	100%	0.0

4.5 Concluding Remarks

Acceptable levels of reliability could not be achieved for large values of N_k when using multiple or variable elitist selection. However, with $1 \leq N_k \leq P/4$, improvements in reliability, richness, and computational cost were realized for the problems studied in this chapter. For results obtained with the original design problem (i.e., using 3–ply stacks), the reliability of the EL selection was so high that little or no improvement could be obtained with alternate selection schemes. Thus, an effort was made to try and improve final population richness and decrease computational cost of the algorithm while maintaining a high level of reliability.

Population richness (although not significant for this problem) was increased substantially using ME and VE selection while maintaining high levels of reliability. Small reductions in the computational cost were also achieved using all alternate selection methods, with best results obtained using the ME_3. The VE_3 scheme produced the lowest computational costs by reducing the effectiveness of the genetic search (increasing the potential for premature convergence of the GA) after N_k was maximized. This situation arises because only one child is created in each successive generation, and explains why similar performance was seen with the ME_3 scheme. Results obtained with the VE_3 may have been substantially worse had the value of N_k been increased much earlier in the genetic search. Thus, improved cost performances could not be justified with the VE_3 scheme.

When the size of the design space was increased by switching to single plies in the stacking sequence, the alternate selection schemes were clearly more effective than the standard EL scheme. The VE_1 and VE_2 schemes were the only selection methods to attain acceptable levels of reliability and 100% population richness at competitive cost levels for the modified problem.

In summary, the VE_1 and VE_2 schemes gave the best performance of all selection schemes when the initial value of N_k was set to an appropriate value. VE_1 selection provided significant improvement over the EL scheme when the population size was increased in the original design problem and when the size of the design space was increased for the modified problem. In the following chapter, the number of design variables increases along with the difficulty of the optimization problem. Thus, from the conclusion drawn here, the VE_1 scheme will be utilized to supply the GA with the ability to behave as both an explorative algorithm, useful at the beginning of the search, and an exploitative algorithm which may be beneficial towards the end of the search.

Chapter 5

GA Application–II: Simultaneous Cost and Weight Minimization of a Simply Supported Composite Plate

In the previous chapter, the optimization problem utilized a single objective function that was used to maximize the twist angle of a cantilevered composite plate made from one material. In this chapter, the complexity of the design problem is increased by allowing for two materials in the stacking sequence of a simply supported composite plate. The optimization problem will be constructed to simultaneously minimize the cost and weight of the plate, while subjected to numerous constraints. Furthermore, methods for representing the weight directly in the objective function will be utilized, instead of controlling weight through strength constraints which penalize thin designs because of material failure (as in Chapter 4).

The purpose of this chapter is to demonstrate the GA's ability to handle more complex composite optimization problems through simple modifications to the basic GA used in chapter 4. These modifications were covered in Chapter 2, which discussed how to accommodate multiple materials in the stacking sequence. Further modifications, which are provided here, focus on the optimization formulation for such problems, and demonstrate one way of easily incorporating multiple objective functions into a genetic algorithm.

In the first section, a brief introduction into multi-objective optimization is presented, followed by a detailed discussion of the design problem in section 5.2. The optimization formulation is given in Section 5.3 and is divided into three parts — derivation of laminate weight and cost and their corresponding objection functions is given in section 5.3.1 and section 5.3.2, respectively, and the methodology for the multi-objective formulation is provided in section 5.3.3. Section 5.4 will present results obtained using the GA–2 algorithm in two different sub-sections, considering cases where the plate is loaded uniaxially, section 5.4.1, and biaxially, section 5.4.2. Concluding remarks are given in section 5.5.

5.1 Multi-Objective Optimization

The use of genetic algorithms for multi-objective optimization has been growing considerably in the past few years. GAs were originally used for maximization or minimization of an unconstrained function. However, there has been increasing interest in optimizing two or more criteria simultaneously, especially if it is difficult to represent one criteria in terms of another. These problems are often referred to as multi-objective (or vector-valued) optimizations problems. One such field of study utilizing this concept is the aerospace industry, where an effort has been made to incorporate cost directly into the design process. This methodology can lead to high performance designs that can be built with available materials and manufacturing techniques. Furthermore, such studies can be used to formulate trade off studies between cost and weight which may aid in the selection of a design that minimizes cost and/or weight [40], two of the most important considerations in aerospace applications.

The goal of single objective optimization problems is straightforward: find the maximum or minimum value of a function for a given set of parameters. The optimization concept is less clear for multi-objective problems, since the best value for one objective usually does not imply that the other objective(s) is simultaneously optimized. Thus, the concept of Pareto-optimality is often used in multi-objective problems to help determine the best way to simultaneously satisfy all objectives to the greatest extent possible.

Pareto-optimality can be explained by looking at a simple example where two generic objectives, P_1 and P_2 need to be minimized, and furthermore, it is difficult to estimate P_1 in terms of P_2, or vice versa. $(\hat{P}_1, \hat{P}_2)$ is a Pareto optimum set if there is no other point $(P_1, P_2) < (\hat{P}_1, \hat{P}_2)$. In Figure 5.1, the relation between P_1 and P_2 is presented, showing five different points. Scenarios A, B, and C provide the best solutions to this problem, although neither one is best at minimizing both quantities at the same time (i.e., there is tradeoff in this problem since one quantity tends to increase as the other decreases). These three points (A,B,C) are referred to as Pareto-optimal (or non-dominated) [7] since no other point can generate lower values for both P_1 and P_2 simultaneously. The other options, D and E, are not Pareto-optimal (or are dominated) since C ¡ E and B ¡ D. Ultimately, the decision maker is left to decide which solution from the Pareto-optimal set is best.

Many optimization studies have been aimed at optimizing two or more quantities simultaneously. The main difference between these studies is in the methodology for obtaining the Pareto-optimal curve. Kassapoglou [40] used multi-objective optimization to simultaneously minimize the cost and

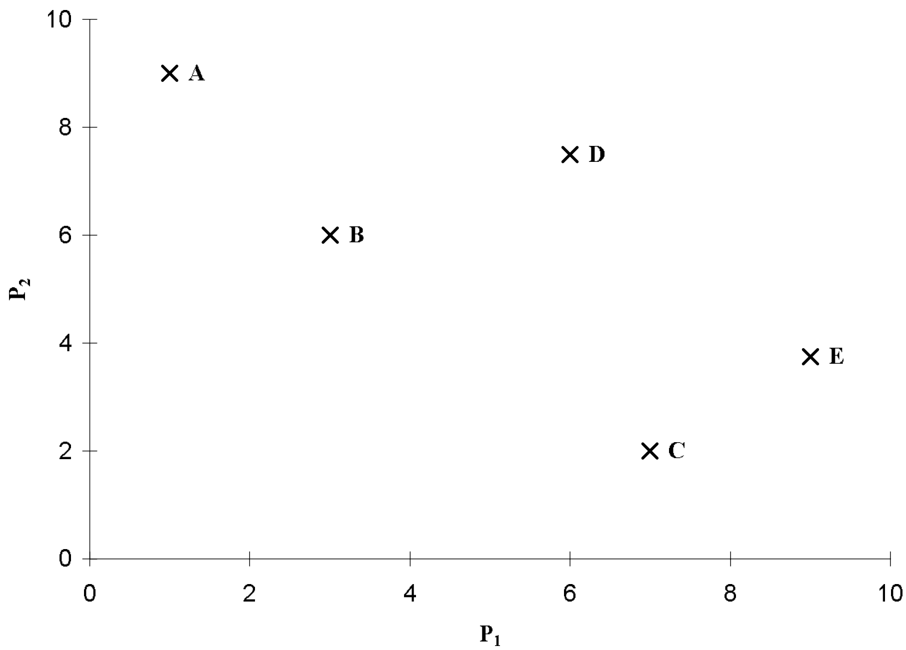


Figure 5.1: Multi-objective optimization. The set $\{(P_1(x), P_2(x))\}$ for 5 different design points.

weight of composite stiffened panels subjected to compression and shear loads. The first step in the optimization procedure involved minimizing each parameter separately. The lowest weight and cost configurations were then identified and placed in the Pareto-optimal set. Designs from the group optimized for cost that were lighter than the minimum cost configuration, and designs from the group optimized for weight that were cheaper to fabricate than the minimum weight configuration comprised the remainder of the candidate Pareto-optimal set. The optimum configuration from this set was chosen to be the one that minimized a certain penalty function. Although the individual minimum weight and cost designs did not coincide, results showed that a set of near-optimal designs could be found. Panels configured with "J" stiffeners provided the lowest weight, while "T" stiffeners produced the lowest cost designs and the best tradeoff between cost and weight.

Not surprisingly, GAs have also been applied to multi-objective problems. Schaffer [41] used genetic algorithms for multi-objective problems by creating equally sized sub-populations. Each sub-population worked on optimizing a single objective. Although selection was carried out in each sub-population individually, crossover was performed between members of both populations. Results showed that this implementation scheme was susceptible to bias against individuals that satisfied both objectives well but did not provide the optimum solution for either criteria, making it difficult to find the entire set of Pareto-optimal designs.

Belegundu et al. [42] implemented a GA in a slightly different manner for multi-objective optimization of a wide range of problems. The selection procedure in the GA was modified by replacing traditional roulette wheel selection with a scheme based on dominated and non-dominated designs. Designs that were non-dominated were given a rank of 1 while designs that were dominated or violated a constraint were given a rank of 2 and thrown away. Successive populations were made up of the parent designs (Rank 1) and a prescribed number of their offspring. If necessary, additional designs (referred to as immigrants) were randomly created to fill the remainder of the population, which also added diversity to the genetic search. This process continued until the entire population was filled entirely with non-dominated designs, the Pareto-optimal set. Preliminary testing of the GA showed that points on the Pareto curve were bunched into small groups instead of being spread out evenly, a phenomenon known as speciation [7]. This problem was handled by assigning twin or near twin designs a rank of two, thereby eliminating them from the Pareto set. Further testing of the algorithm showed that the GA was effective in generating Pareto solutions for optimizing aeromechanical responses for turbomachinery airfoils, and minimizing the cost and residual stresses in the fabrication of ceramic composite plates.

In this work, the multi-objective formulation will be carried out by applying a scale factor to cost and weight objective functions. To obtain a Pareto set of designs, the influence of cost and weight on the overall fitness of a plate configuration is adjusted from one extreme to the other by varying the scale factor accordingly. This allowed the general configuration of the GA to be maintained since the fitness of each laminate design is still based on a single value that is comprised of both cost and weight information. An indepth discussion of this methodology will be presented shortly.

5.2 Problem Formulation

The composite panel under consideration is 36 in long, 30 in wide, and simply supported on all four sides, see Figure 5.2. The panel can be loaded under any combination of axial and shear loads (i.e., N_x, N_y, and N_{xy}). Each ply in the panel may be made of either graphite-epoxy or Kevlar-epoxy (see Table 5.1 for properties) and can have any ply orientation angle between -75° and 90°, in increments of 15°, as shown in Figure 5.2. The load handling capabilities of laminated composite plates comprised of two materials is the analysis used in this section of the paper. To determine these capabilities, two quantities must be found: the margin of safety for the critical buckling load, and the margin of safety for the principal ply strains.

5.2.1 Critical buckling load

To find the critical buckling loads for a symmetrically laminated anisotropic composite plate ($N_{x_{cr}}$, $N_{y_{cr}}$, $N_{xy_{cr}}$), the Galerkin energy method outlined in Whitney [38], was utilized. Realizing that there is no coupling between bending and extension for symmetric laminates (i.e., $[B_{ij}] = 0$), the strain energy, U, for transverse bending of a laminated plate of length ($x = a$) and width ($y = b$) is

$$
\begin{aligned}
U &= \frac{1}{2}\int_0^b\int_0^a \left[D_{11}\left(\frac{\partial^2 w}{\partial x^2}\right)^2 + 2D_{12}\frac{\partial^2 w}{\partial x^2}\frac{\partial^2 w}{\partial y^2} + D_{22}\left(\frac{\partial^2 w}{\partial y^2}\right)^2 + 4\left(D_{16}\frac{\partial^2 w}{\partial x^2}\right.\right. \\
&\quad \left.\left. + D_{26}\frac{\partial^2 w}{\partial y^2}\right)\frac{\partial 2w}{\partial x \partial y} + 4D_{66}\left(\frac{\partial^2 w}{\partial x \partial y}\right)^2\right] dxdy., \qquad (5.1)
\end{aligned}
$$

where the bending stiffness' of the plate ($[D_{ij}]$) are determined using classical lamination theory, see Jones [4]. Next, the potential energy, V, of the biaxial and shear loads (N_x^a, N_y^a, and N_{xy}^a) that are applied to the plate is considered

$$
V = \frac{1}{2}\lambda\int_0^b\int_0^a \left[N_x^a\left(\frac{\partial w}{\partial x}\right)^2 + N_y^a\left(\frac{\partial w}{\partial y}\right)^2 + 2N_{xy}^a\left(\frac{\partial^2 w}{\partial x \partial y}\right)\right] dxdy. \qquad (5.2)
$$

To determine the governing equation for the composite plate, Hamilton's principle [39] is used

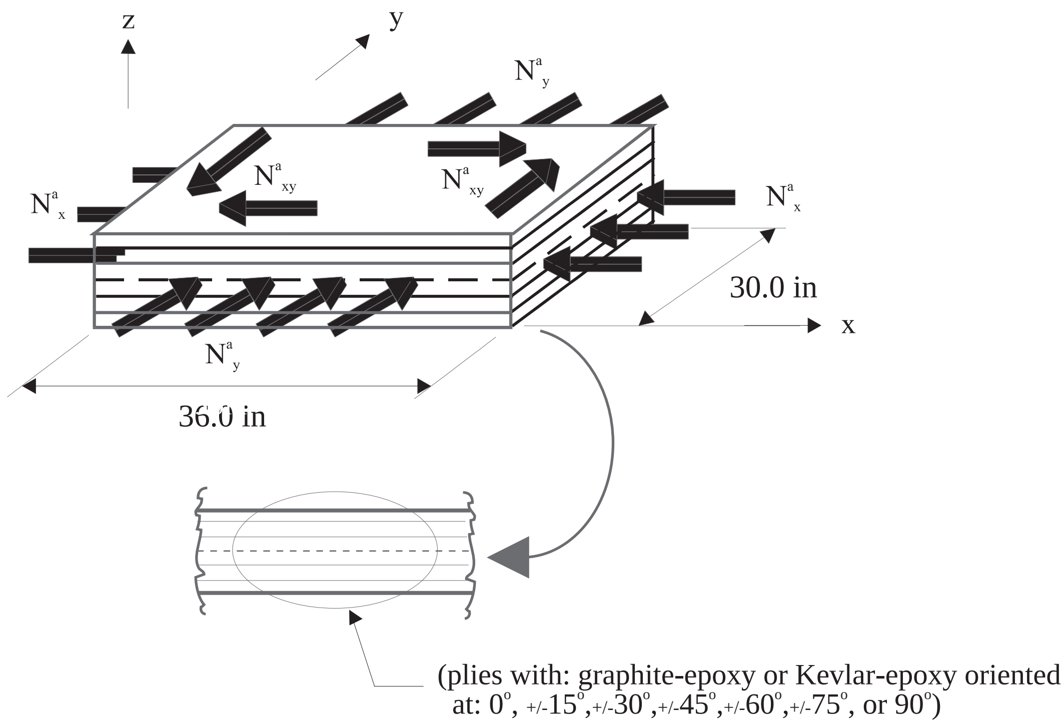


Figure 5.2: Configuration and loading conditions for simply supported plate.

Table 5.1: Material properties for Kevlar-Epoxy and Graphite-Epoxy.

Property	**Gaphite-Epoxy**	**Kevlar-Epoxy**
Young's modulus (longitudinal)	$E_{11} = 22.0 \times 10^6$ psi	$E_{11} = 11.9 \times 10^6$ psi
Young's modulus (transverse)	$E_{22} = 1.50 \times 10^6$ psi	$E_{22} = 0.6 \times 10^6$ psi
Shear modulus	$G_{12} = 0.52 \times 10^6$ psi	$G_{12} = 0.4 \times 10^6$ psi
Poisson's ratio	$\nu_{12} = 0.25$	$\nu_{12} = 0.25$
Ply thickness	$t = 0.00525$ in.	$t = 0.00716$ in.
Material density	$\rho = 0.0055 \, \frac{\text{lb}}{\text{in}^3}$	$\rho = 0.0048 \, \frac{\text{lb}}{\text{in}^3}$
Cost factor (per pound of material)	$C_f = 3.0$ /lb	$C_f = 1.0$ /lb
Allowable strain (longitudinal)	$\epsilon_{11}^{all} = 9.89 \times 10^{-3}$	$\epsilon_{11}^{all} = 2.87 \times 10^{-3}$
Allowable strain (transverse)	$\epsilon_{22}^{all} = 3.87 \times 10^{-3}$	$\epsilon_{22}^{all} = 3.00 \times 10^{-3}$
Allowable shear strain	$\gamma_{12}^{all} = 1.90 \times 10^{-3}$	$\gamma_{12}^{all} = 1.21 \times 10^{-3}$

$$- \int (-\delta U - \delta V) dt = 0, \tag{5.3}$$

where δU and δV are the first variations in strain energy and potential energy due to the in-plane loads, respectively. The governing equation for the composite plate takes the form of

$$\begin{aligned}
&\int_0^b \int_0^a \left[D_{11} \frac{\partial^4 w}{\partial x^4} \; + \; 2\left(D_{12} + 2D_{66}\right) \frac{\partial^4 w}{\partial x^2 \partial y^2} \; + \; 4 \left(D_{16} \frac{\partial^4}{\partial x^3 \partial y} \; + \; D_{26} \frac{\partial^4 w}{\partial x \partial y^3} \right) \right. \\
&\qquad \left. + \; D_{22} \frac{\partial^4 w}{\partial y^4} \; - \; \lambda \left(N_x^a \frac{\partial^2 w}{\partial x^2} \; - \; 2N_{xy}^a \frac{\partial^2 w}{\partial x \partial y} \; - \; N_y^a \frac{\partial^2 w}{\partial y^2} \right) \right] \delta w \; dx \; dy \\
&\qquad + \; 2 \oint_0^b \left[D_{26} \frac{\partial^2 w}{\partial x \partial y} \right] \; \frac{\partial(\delta w)}{\partial y} \; dx \; + \; 2 \oint_0^a \left[D_{16} \frac{\partial^2 w}{\partial x \partial y} \right] \; \frac{\partial(\delta w)}{\partial x} \; dy \; = \; 0.
\end{aligned} \tag{5.4}$$

Since the effects of the D_{16} and D_{26} terms are not neglected, the surface integrals in Eq. (5.4) are included in the governing equation. This allows the transverse deflection and first variation of the transverse deflection to be formulated in a double sine series

$$\begin{aligned}
w \;&=\; \sum_{i=1}^{I} \sum_{j=1}^{J} A_{ij} \sin\left(\frac{i\pi x}{a}\right) \sin\left(\frac{j\pi y}{b}\right) \\
\delta w \;&=\; \sum_{m=1}^{M} \sum_{n=1}^{N} \delta A_{mn} \sin\left(\frac{m\pi x}{a}\right) \sin\left(\frac{n\pi y}{b}\right).
\end{aligned} \tag{5.5}$$

Substituting Eq. (5.5) into Eq. (5.4), performing the necessary integrations, and noting that

$$w = \frac{\partial^2 w}{\partial x^2} = \frac{\partial^2 w}{\partial y^2} = 0 \qquad \text{(on the plate boundaries)}, \tag{5.6}$$

the following set of algebraic equations are obtained:

$$
\begin{aligned}
\sum_{m=1}^{M}\sum_{n=1}^{N}\Bigg\{ \pi^4 \Bigg[& m^4 D_{11} \;+\; 2\,(D_{12} \;+\; 2D_{66})\,(mn)^2 \;+\; (nR)^4 D_{22} \\
& + \left(\frac{am}{\pi}\right)^2 \lambda N_x^a \;+\; \left(\frac{an}{\pi}\right)^2 \lambda N_y^a \Bigg] A_{mn} \\
- 32mnR\pi^2 \Bigg[& \sum_{i=1}^{M}\sum_{j=1}^{N} M_{ij}(i^2 \;+\; m^2)D_{16} \;+\; R^2(n^2+j^2)D_{26} \\
& + \left(\frac{a}{\pi}\right)^2 \lambda N_{xy}^a \Bigg] A_{ij} \Bigg\} \quad = \quad 0, \qquad \text{where}
\end{aligned}
\tag{5.7}
$$

$$
M_{ij} = \begin{cases} \dfrac{ij}{(m^2-i^2)(n^2-j^2)} & (m \pm i) \;\; \text{odd}, \quad (n \pm j) \;\; \text{odd} \\[2ex] 0 & \text{otherwise}, \end{cases}
$$

and R is defined as the plate aspect ratio (a/b). Equation (5.7) yields MN homogeneous equations that can be broken into the form of $[A]\{x\} - \lambda[B]\{x\} = 0$. The coefficient matrix, contains terms involving N_x^a, N_y^a, and N_{xy}^a only. The smallest value of λ, λ_{cr}, for which the determinant of the coefficient matrix vanishes will give the values for the critical buckling loads

$$
\begin{bmatrix} N_{x_{cr}} \\ N_{y_{cr}} \\ N_{xy_{cr}} \end{bmatrix} = \lambda_{cr} \begin{bmatrix} N_x^a \\ N_y^a \\ N_{xy}^a \end{bmatrix}. \tag{5.8}
$$

Finally, λ_{cr} is used to calculate the margin of safety:

$$
\lambda_b = 1 - \lambda_{cr}. \tag{5.9}
$$

5.2.2 Principal ply strains

To complete the analysis, the margins of safety for the principle ply strains must be determined. The laminate strains for the composite plate are determined from the stress-strain relationship

$$\begin{bmatrix} \epsilon_x \\ \\ \epsilon_y \\ \\ \gamma_{xy} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{16} \\ \\ A_{12} & A_{22} & A_{26} \\ \\ A_{16} & A_{26} & A_{66} \end{bmatrix} \begin{bmatrix} N_x^a \\ \\ N_y^a \\ \\ N_{xy}^a \end{bmatrix}, \tag{5.10}$$

where the extensional stiffnesses, $[A_{ij}]$ are determined using classical lamination theory once again. To calculate the principal ply strains, the laminate strains are transformed through the ply angle θ using the methods described in Jones [4]

$$\begin{bmatrix} \epsilon_{11} \\ \\ \epsilon_{22} \\ \\ \gamma_{12} \end{bmatrix} = \begin{bmatrix} \cos^2\theta & \sin^2\theta & \sin\theta\cos\theta \\ \\ \sin^2\theta & \cos^2\theta & -\sin\theta\cos\theta \\ \\ -2\sin\theta\cos\theta & 2\sin\theta\cos\theta & \cos^2\theta - \sin^2\theta \end{bmatrix} \begin{bmatrix} \epsilon_x \\ \\ \epsilon_y \\ \\ \gamma_{xy} \end{bmatrix}. \tag{5.11}$$

The largest ratio of principal ply strain (ϵ_{ij}) to the corresponding allowable strain (ϵ_{ij}^{all}) is then used to calculate the margin of safety,

$$\lambda_s = 1 \;-\; \max\left\{\frac{\epsilon_{11}}{\epsilon_{11}^{all}}, \frac{\epsilon_{22}}{\epsilon_{22}^{all}}, \frac{\gamma_{12}}{\gamma_{12}^{all}}\right\}. \tag{5.12}$$

The allowable strains for each direction (ϵ_{11}^{all}, ϵ_{22}^{all}, and γ_{12}^{all}) were determined by comparing maximum compressive and tensile values and choosing the smaller of the two (a conservative approach). The resulting values for each material are listed in Table 5.1

5.3 Optimization Procedure

The goal of the optimization is to find the stacking sequence of the plate which provides the lowest weight and cost but does not buckle or fail due to excessive strain. For simplicity, it is also assumed that the laminate stacking sequence is symmetric about the mid-plane and balanced. Since the GA works with a string that represents one half of the laminate stacking sequence, the symmetry constraint is automatically satisfied. The balance constraint, which ensures that each ply oriented at $+\theta^\circ$ is complemented with another ply oriented at $-\theta^\circ$ throughout the stacking sequence, will be enforced using penalty parameters. Discussion of the optimization procedure is split into three sections: laminate weight and cost calculation, objective function formulation, and the formulation for multi-objective optimization.

5.3.1 Calculating Laminate Weight and Cost

The panel weight is calculated as

$$W = ab[\rho_{ke}t_{ke}N_{ke} + \rho_{ge}t_{ge}N_{ge}] \tag{5.13}$$

where a and b are the dimensions of the plate, ρ_{ke} and ρ_{ge} are the material densities, t_{ke} and t_{ge} are the corresponding ply thicknesses for each material, and N_{ke} and N_{ge} are the number of plies of Kevlar-epoxy and graphite-epoxy in the laminate stacking sequence, respectively.

The cost of a laminate is based on two quantities: material cost and lay-up cost. The material cost for a laminate, C_m, is determined by multiplying the weight of each material in a laminate by its corresponding cost factor (C_f) given in Table 5.1:

$$C_m = ab[C_{f_{ke}}\rho_{ke}t_{ke}N_{ke} + C_{f_{ge}}\rho_{ge}t_{ge}N_{ge}]. \tag{5.14}$$

Lay-up cost, C_l, is based on the amount of time required by the lay-up machine to construct each laminate. Data was obtained from a standardized manufacturing process relating ply orientation angle and plate dimensions. However, since the dimensions of the plate are the same for each laminate, lay-up cost becomes a function of ply orientation angle only.

The analysis procedure used to compute the layup cost can not be revealed in this document because they are company proprietary information. Instead, we have given a table which shows a multiplier for the layup cost as a function of the ply orientation angle (the table is coded into the algorithm to provide information on layup cost when needed). This information, depicted in Figure 5.3, shows that the most expensive plies to construct are those oriented at $\pm 45°$. Plies oriented at $0°$ degrees are more expensive than $90°$ plies because the plate is 6 inches longer in the x direction. The total cost for a laminate can now be determined by adding the corresponding material and lay-up costs:

$$C_t = C_m + C_l. \tag{5.15}$$

5.3.2 Objective Function Formulation

The optimization problem can be formulated as:

$$\begin{gathered} \text{minimize } W \text{ and } C_t \text{ such that} \\ \lambda_b, \lambda_s \geq 0, \end{gathered} \tag{5.16}$$

where λ_b and λ_s are the margins of safety for the critical buckling load and principal ply strains, respectively. Two fitness functions will be utilized for this problem, one for laminate weight (Φ_w) and another for laminate cost (Φ_c). To accommodate the genetic algorithm, the degree of constraint

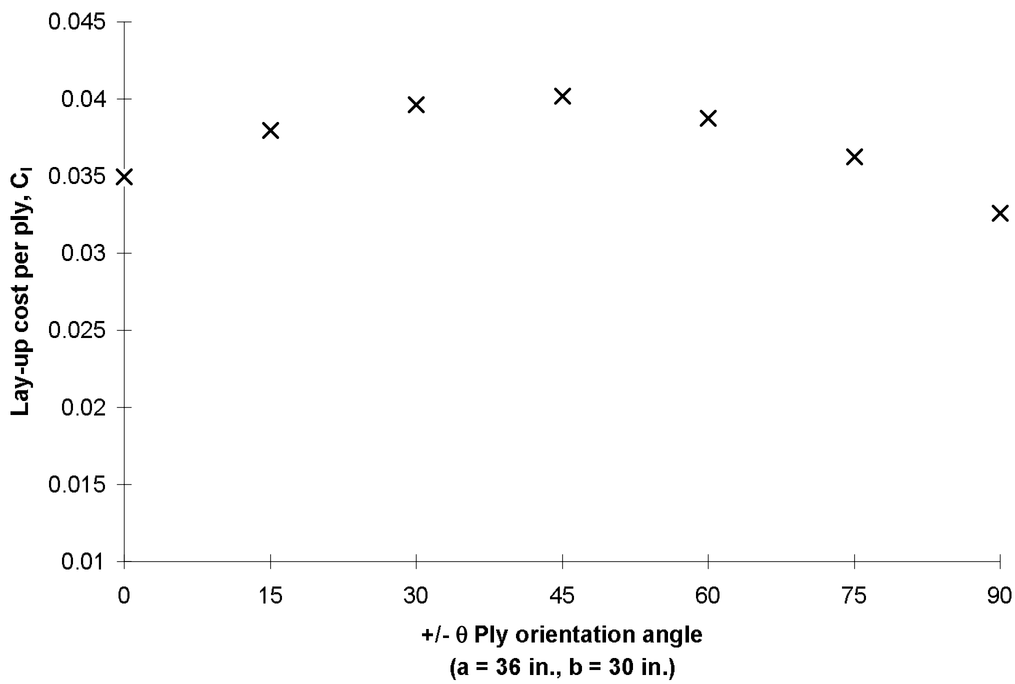


Figure 5.3: Lay-up costs for different ply orientation angles.

violation or satisfaction must be transformed into added penalties or bonuses [25] that augment each objective function.

The fitness function for laminate weight is defined by

$$\Phi_w = \begin{cases} W + P_{ke}^w + P_{ge}^w - \epsilon_r^w, & \text{feasible laminates,} \\ W\epsilon_p + P_{ke}^w + P_{ge}^w, & \text{infeasible laminates.} \end{cases} \tag{5.17}$$

Since the first objective is weight minimization, the numerical values of the bonus and penalty parameters given in Eq. 5.17, which will be discussed in the following paragraphs, are proportional to the ply weights of each material, providing a means of penalizing or rewarding a laminate by adjusting laminate weight. For example, if a laminate violates a constraint, a penalty parameter is added to the weight, W, making the laminate less desirable because its weight has been artificially increased. To enforce the balance constraint for both materials, two penalty parameters, P_{ke}^w and P_{ge}^w, are added to the objective function

$$P_x^w = \begin{cases} 0, & \text{laminate balanced,} \\ N_{ub_x} w_x, & \text{laminates unbalanced,} \end{cases} \tag{5.18}$$

where w is the weight of a single ply, N_{ub} is the number of unbalanced plies in the laminate, and $x = ke, ge$ for Kevlar-epoxy and graphite-epoxy, respectively. Thus, if either material has unbalanced plies in a laminate, the laminate is penalized by an amount equal to the weight of the number of unbalanced plies, making the laminate artificially heavier and thereby less desirable.

The formula for feasible laminates in Eq 5.17 is used if all constraints are satisfied. Feasible laminates are rewarded with a bonus, ϵ_r^w, whose value depends on the average material weight of a ply, and the amount of constraint satisfaction so that designs satisfying the constraints by a larger margin become more desirable. The constraint that is closest to being violated is used to calculate the bonus parameter:

$$\epsilon_r^w = w_a \min\{\lambda_b, \lambda_s\}, \tag{5.19}$$

where w_a represents the average weight of a ply.

If a constraint is violated the formula for infeasible laminates shown in Eq. 5.17 is used. A laminate is artificially made heavier using the penalty parameter, ϵ_p:

$$\epsilon_p = (1 - \min\{\lambda_b, \lambda_s\})^P. \tag{5.20}$$

The penalty parameter is rationalized by first using a scale factor determined by the value of the most violated constraint. Laminates that are very thin may appear desirable even when a constraint is violated. This problem is handled by adding the exponent P to the scale factor. A value for P will be determined in the multi-objective formulation given in the next section. Parameters ϵ_r^w and ϵ_p are used only if all constraints are satisfied or at least one constraint is violated, respectively.

The objective function for laminate cost is formulated in a manner similar to the one for weight, except that bonus and penalty parameters are now proportional to the material and lay-up costs for a ply:

$$\Phi_c = \begin{cases} C_t + P_{ke}^c + P_{ge}^c - \epsilon_r^c, & \text{feasible laminates,} \\ C_t \epsilon_p + P_{ke}^c + P_{ge}^c, & \text{infeasible laminates.} \end{cases} \tag{5.21}$$

The balance constraints were modified first

$$P_x^c = \begin{cases} 0, & \text{(laminate balanced)}, \\ N_{ub_x}(c_a^l + c_x^m), & \text{(laminates unbalanced)}, \end{cases} \tag{5.22}$$

where c_a^l is the average lay-up cost for a ply and c_x^m is the cost for a single ply of material $x = ke, ge$ for Kevlar-epoxy and graphite-epoxy, respectively. The parameter c_a^l is determined by dividing the sum of the lay-up costs for each permissible ply orientation angle, c_i^l, by the total number of permissible angles, N_{pa}:

$$c_a^l = \frac{\sum_{i=0°,15°,30°\ldots}^{90°} c_i^l}{N_{pa}}. \tag{5.23}$$

Next, the bonus parameter, ϵ_r^c was developed to make feasible laminates artificially appear less expensive and thus more desirable:

$$\epsilon_r^c = (c_a^l + c_a^m)\min\{\lambda_b, \lambda_s\}, \tag{5.24}$$

where c_a^m is the average material cost of a single ply. The parameter used to penalize infeasible laminates, ϵ_p is identical to the one used for the weight objective function, see Eq. 5.20.

5.3.3 Multi-Objective Formulation

The multi-objective formulation is carried out by using a convex combination of the weight and cost objective functions determined in the previous section. This allows the fitness of each laminate to be represented by a single quantity:

$$\Phi_{lam} = \alpha\Phi_c + (1-\alpha)\Phi_w, \qquad 0 \le \alpha \le 1. \tag{5.25}$$

By combining Eq. 5.17 and Eq. 5.21, a more detailed expression can be obtained

$$\Phi_{lam} = \begin{cases} \left[W + P_{ke}^w + P_{ge}^w - w_a M\right](1-\alpha) \ + & \text{feasible laminates,} \\ \left[C_t + P_{ke}^c + P_{ge}^c - (c_a^l + c_a^m)M\right]\alpha, & \\ \left[W(1-M)^P + P_{ke}^w + P_{ge}^w\right](1-\alpha) \ + & \text{infeasible laminates,} \\ \left[C_t(1-M)^P + P_{ke}^c + P_{ge}^c\right]\alpha, & \end{cases} \tag{5.26}$$

where $M = \min\{\lambda_b, \lambda_s\}$. A set of Pareto optimal designs can be determined by varying α in small increments from zero to one. To complete the formulation, the GA was tested to determine a value for P. Initial results using a value of $P = 1$ showed that thin laminates which violated a constraint were more desirable than feasible laminates. Further testing showed that increasing P to a value of 2.7 eliminated this problem without penalizing feasible laminates too much.

5.4 Results

Results will be given in two subsections. The first subsection will consider results obtained for uniaxial loading conditions, and the second subsection will consider results for biaxial loading. For these load cases, the value of α was varied from 0 to 1 in increments of 0.01, yielding 101 different combinations of cost and weight in the objective function. Fifty optimization runs, using a population size of fifty for each run, were conducted for each value of α. The best design from each set of runs is placed in the Pareto-optimal set. Using this approach, there is a possibility for different values of α to yield the same optimal design due to the discrete nature of the problem.

But since the finite number of Pareto-optimal designs is unknown, many values for α were used to improve the chances of finding the entire set. For all runs conducted, the GA was equipped with the VE_1 selection scheme using a value of 5 for N_k. GA operator probabilities are listed in Table 5.2.

5.4.1 Uniaxial Loading

In this section, the laminated plate was placed under a compressive load of 100 lbs/in along the x–axis of the plate, see Figure 5.4. For this configuration, 6 designs were found in the Pareto-optimal set. The GA was successful in finding only five of these designs, see Figure 5.5. Point A depicts the minimum weight design while point F is the lowest cost design. Design D, represented by an "$\times$" in Figure 5.5, could not be found by the GA for any of the values of α used. The reasons for this will be explained shortly.

The properties of each design are given in Table 5.3 and Table 5.4 (the strain constraint was not critical and is not listed). The stacking sequence representation gives the ply orientation angles and material makeup for one half of the symmetrically laminated plate, with the left end corresponding to the outer edge. In the discussion of the various laminate designs that follow, references will be made to the left half of the laminated stacking sequence only. Plies that are made from graphite are superscripted with a (1), while those made from Kevlar are superscripted with a (2).

As seen in Table 5.3, the possibility for different values of α yielding the same design was realized for this problem. Since numeric values for cost are considerably larger than those for weight, many of the different designs were found for small values of α, where the objective function for laminate weight had sufficient influence on the overall fitness of a laminate. The small number of designs found by the GA is due to the discrete nature of the problem. Unlike continuous optimization

Table 5.2: GA operator implementation probabilities.

Operator	**Probability**
Crossover	1.00
Single ply alteration (orientation)	0.05
Single ply alteration (material)	0.05
Ply addition	0.05
Ply deletion	0.10
Permutation	0.75

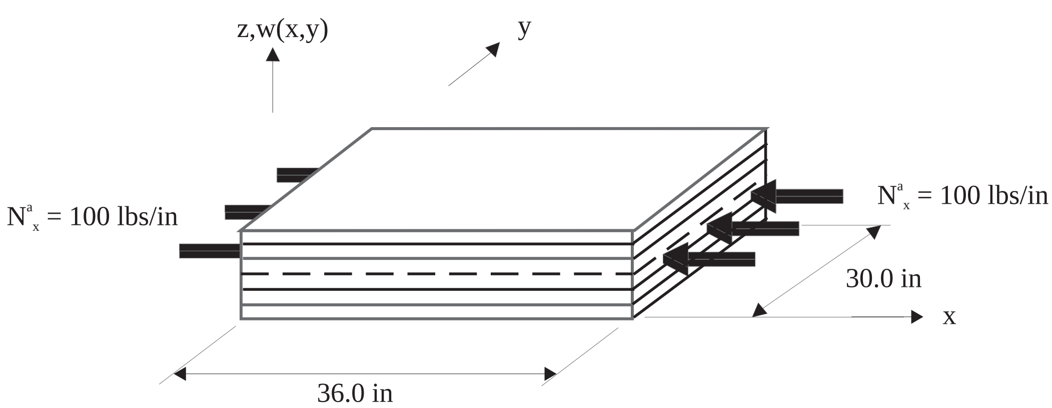


Figure 5.4: Plate configuration: uniaxial loading.

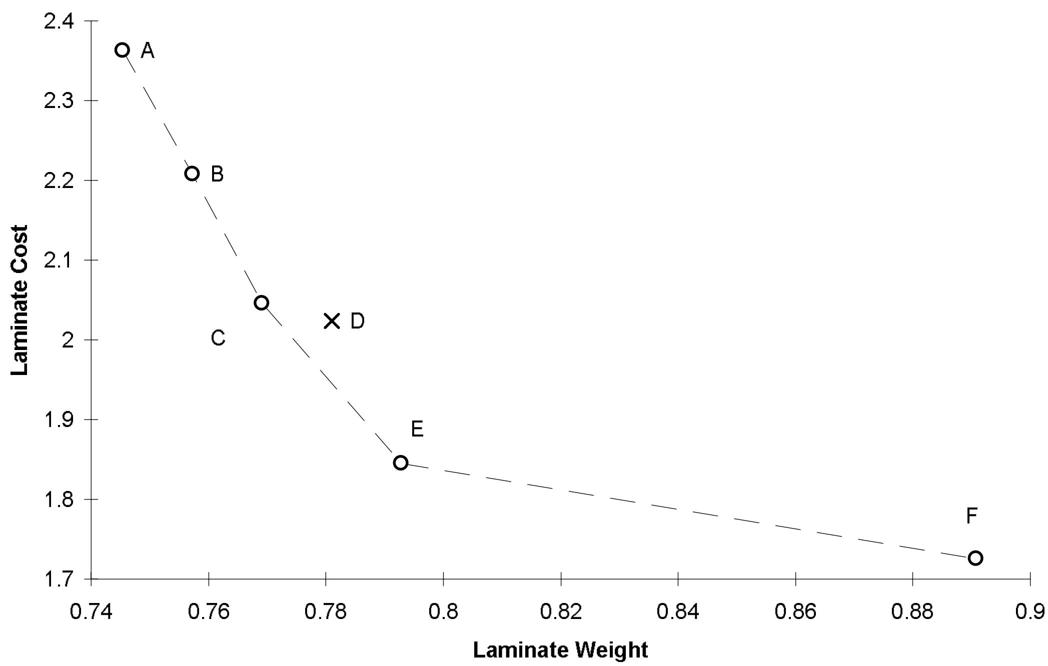


Figure 5.5: Set of Pareto-optimal designs for uniaxial loading.

problems, an infinite number of possibilities for laminate weight and cost do not exist between point A and point E. The set of optimal laminate designs for this problem are comprised of either 11 or 12 plies. Any design consisting of 10 plies violates the critical buckling load constraint, regardless of stacking sequence or material type arrangement. All laminate comprised of 13 or more plies are dominated, in a Pareto sense, by other designs.

Once the minimum number of plies for a laminate is determined, the only way for the GA to obtain the Pareto-set of designs is to adjust a ply's material type or its orientation angle, both of which offer a discrete set of choices. Although there are numerous designs with different values for cost, there are very few choices when trying to find laminates with different weight. This is because weight can only be adjusted by altering a ply's material type or shifting between 11 or 12 plies. On the other hand, the cost of a laminate can be adjusted in many ways by using different ply orientation angles (in addition to switching material types and the number of plies). Thus, the number of designs in the Pareto-set is governed by the few available choices for laminate weight.

Design A represents the lightest laminate. Since the critical buckling load is the active constraint, the GA chose a combination of plies from both materials that yields the lightest laminate with the highest bending stiffness. For this problem, high bending stiffness was achieved by using a combination of high strength material, orienting plies at 45°, and increasing the laminate thickness as much as possible. Since plies that are furthest from the laminate mid-plane have the most influence on a laminate's bending stiffness, the GA found a design with six $\pm 45°$ plies of high strength graphite-epoxy at the outer edges of the laminate. The inner portion of the laminate is comprised of Kevlar-epoxy plies, four oriented at $\pm 45°$ and one oriented at 90° to satisfy the balanced constraint. Although lower in strength, each ply of Kevlar is over 35% thicker than a ply of graphite. By using thicker plies, the graphite material is pushed further away from the mid-plane than would

Table 5.3: Laminate properties for Pareto-optimal designs, uniaxial loading.

Design	**Stacking sequence**	**Thickness**	**No. of plies**	α
A	$[\pm 45_3^{(1)}/\pm 45_2^{(2)}/90^{(2)}]_s$	0.1346in	11	0.00, 0.01, 0.02
B	$[\pm 45_2^{(1)}/90^{(1)}/\pm 60_2^{(2)}/90_2^{(2)}]_s$	0.1384in	11	0.03, 0.04,...,0.06
C	$[\pm 45_2^{(1)}/90_7^{(2)}]_s$	0.1422in	22	0.07, 0.08,...,0.10
D	$[\pm \mathbf{45^{(1)}/90^{(1)}/\pm 45_4^{(2)}}]_\mathbf{s}$	**0.1461in**	**11**	**not found**
E	$[\pm 45^{(1)}/\pm 45^{(2)}/\pm 60^{(2)}/90_5^{(2)}]_s$	0.1499in	11	0.11, 0.12,...,0.40
F	$[30^{(2)}/60^{(2)}/-30^{(2)}/-60^{(2)}/90_8^{(2)}]_s$	0.1718in	12	0.41, 0.42,...,1.00

Table 5.4: Laminate performance for Pareto-optimal designs, uniaxial loading.

Design	**Weight**	**Layup cost**	**Material Cost**	**Total Cost**	**Buckling Load**
A	0.7454	0.8686	1.4938	2.3624	99.32 lbs/in
B	0.7573	0.8270	1.3810	2.2080	99.94 lbs/in
C	0.7691	0.7776	1.2681	2.0457	100.06 lbs/in
D	**0.7810**	**0.8686**	**1.1552**	**2.0238**	**100.63 lbs/in**
E	0.7929	0.8023	1.0423	1.8446	101.404 lbs/in
F	0.8908	0.8350	0.8908	1.7258	100.72bs/in

otherwise be achieved by using graphite-epoxy throughout the stacking sequence, and provides the maximum bending stiffness.

If the GA were to use all graphite-epoxy plies, the lowest weight design that does not buckle would be $[\pm 45_6^{(1)}/90^{(1)}]_s$, requiring two additional plies to satisfy the buckling constraint, an increase of over 4% in weight when compared to design A. Since graphite plies are thinner, more of them are required to provide the necessary laminate thickness to prevent buckling. Likewise, if Kevlar-epoxy were used throughout the stacking sequence, the lightest laminate would consist of 12 plies. This design is 19.5% heavier than A, requiring more material to satisfy the buckling constraint since Kevlar is not as strong as graphite. Thus, to find the lightest design, the GA found a design comprised of plies which provided the correct balance of high strength material and laminate thickness in the stacking sequence.

Point B represents the second lightest design in the Pareto set. Although cheaper to fabricate, this design is also slightly heavier than design A. Material cost is reduced by trading two plies of graphite for Kevlar. Since the GA works with only half of the laminate stacking sequence, this results in an odd number of graphite plies, one oriented at $90°$ to satisfy the balanced constraint. Layup cost is decreased by switching 2 stacks of $\pm 45°$ plies to a stack of $\pm 60°$ and two plies oriented at $90°$. Although there is a smaller amount of graphite in the stacking sequence, and fewer plies that are oriented at $\pm 45°$, there is sufficient increase in laminate thickness to satisfy the buckling constraint. The end result is an increase in weight of approximately 1.6% and a savings of over 6.5% in laminate cost, with almost no change in the critical buckling load.

As laminate cost becomes more influential in the optimization process, the GA searches for ways to satisfy the buckling constraint while using plies of less expensive Kevlar. Furthermore, the GA must find designs with fewer $\pm 45°$ plies in the stacking sequence since they are more expensive to layup. The GA responds by once again changing a single ply of graphite to Kevlar, which increases laminate thickness and reduces material cost. Of all graphite plies, the one oriented at 90° is changed because it has the smallest effect on laminate bending stiffness due to it is orientation angle and its distance from the mid-plane. Switching the 90° ply does not disturb the balanced constraint either. The remaining plies of Kevlar are also oriented at 90°, minimizing laminate cost but still providing enough thickness to prevent buckling. This modification reduces the overall cost of the laminate by over 7.5% when compared to design B.

The stacking sequence for design E is influenced mostly by the cost of the laminate. Thus, the GA eliminates all but two plies of graphite-epoxy. With less graphite in the laminate, the buckling constraint is satisfied because thickness is increased and plies are oriented at angles which give the most bending stiffness (i.e., $\pm 45°$ and $\pm 60°$). Once again, plies furthest from the mid-plane are oriented at $\pm 45°$ to satisfy the buckling constraint.

In finding designs A, B, and C in the Pareto set, the GA gradually increased laminate weight by trading one ply of graphite for Kevlar in each successive step. However, the weight increase from design C to E resulted from changing two plies of graphite to Kevlar. Thus, it was of interest to see if there was a design, comprised of three plies of graphite and eight plies of Kevlar, that would provide weight and cost characteristics between those of designs C and E. To see if the GA could find this design, α was varied between 0.10 (the value of α which produced design C) and 0.11 (which produced design E), in increments of 0.001. Although fifty optimization runs were conducted for each refined value of α, the GA could still not find design D. Thus, design D is represented with an "$\times$" in Figure 5.5

However, by manually adjusting the ply orientation angles using three plies of graphite (placed at the outer edge) and eight plies of Kevlar, design D was quickly located. The reason that the

GA could not find this design can be explained by looking at the fitness values for designs C, D, and E for each value of $0.1 < \alpha < 0.11$. This data, listed in Figure 5.6, shows that design C is the most attractive design initially. As α increases, the fitness value for design E becomes smaller than design C's, while design D's fitness value is substantially higher than these two, regardless of the value for α. Although design D's fitness may eventually become more attractive than design C, it will never simultaneous be smaller than both D and E, making it impossible for the GA to find it. A possible explanation of this phenomenon is given by Das and Dennis [43] who argue that using convex combinations of objective functions will not produce the entire set of Pareto-optimal points if the Pareto-optimal curve is not convex. It can be seen in Figure 5.5 that design D clearly makes the Pareto curve non-convex.

For design F, cost is the only consideration in the optimization process. Thus, to achieve the lowest possible material cost the GA uses only Kevlar in the stacking sequence. Since Kevlar is not strong as graphite, 12 plies are required in the stacking sequence to satisfy the buckling constraint (as opposed to 11 for all other designs). To reduce layup cost as much as possible the stacking sequence consists of eight plies oriented at $90°$. The remainder of the laminate is made up plies oriented at $\pm 30°$ and $\pm 60°$ and are placed at the outer edges of the laminate. This configuration reduces layup cost as much as possible put produces enough bending stiffness to satisfy the buckling constraint. Although a large jump in laminate weight exists between points E and F, designs found between these two points were either dominated, or violated the buckling constraint.

The general buckling mode shape for all designs in the Pareto set is shown in Figure 5.7. Under uniaxial loading conditions, the plate deforms in the shape of one half sine wave in both the x and

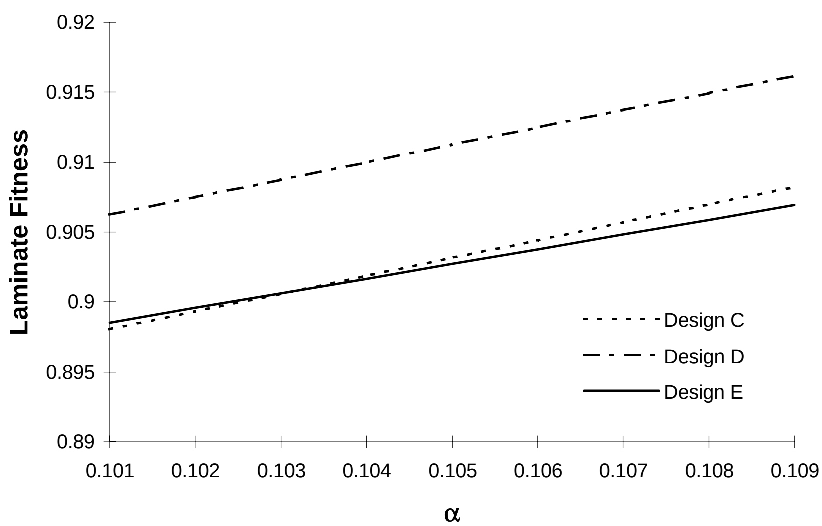
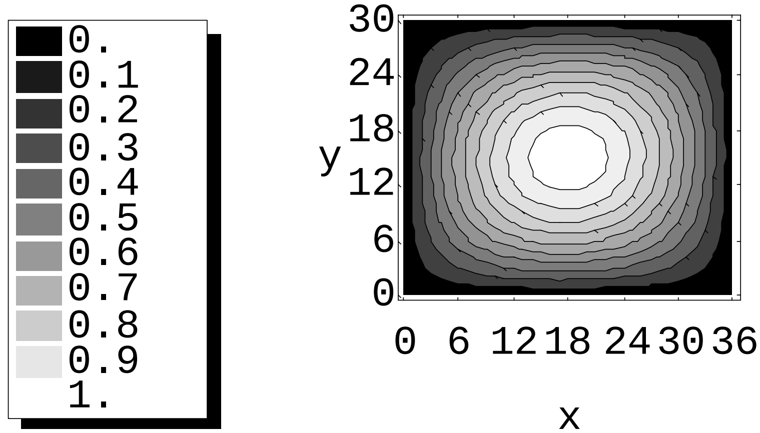


Figure 5.6: Fitness comparison between designs C, D, and E for specific values of α.



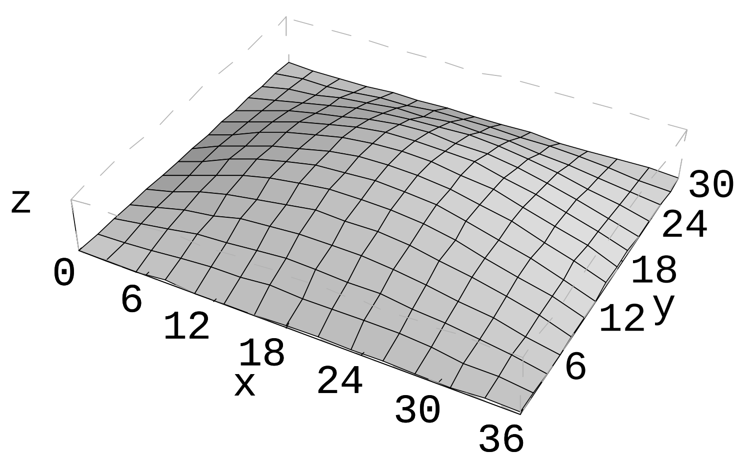


Figure 5.7: Buckling mode shapes for uniaxial loading.

y directions.

5.4.2 Biaxial Loading

In this section, results are reviewed for biaxial loading conditions, see Figure 5.8. A discrete set of Pareto-optimal points, consisting of 9 designs was found for this case, see Figure 5.9. Laminate properties for all designs are listed in Table 5.5 and Table 5.6. The GA used similar techniques discussed in the previous section to find this set of designs. To begin with, the GA finds the combination of graphite and Kevlar that yield the lowest possible number of plies. For design G, the lightest design, the minimum number of plies required to satisfy the buckling constraint (the strain constrain was inactive once again) is 15, 8 plies of graphite and 7 plies of Kevlar. As cost is gradually added to the objective function, the GA replaces graphite with Kevlar and finds stacking sequences that are less expensive to layup but still provide enough bending stiffness to prevent the laminate from buckling. The cheapest design, O, is made entirely of Kevlar and is comprised of 16 plies.

Points J, L, and N represent designs that were not found by the GA for any value of α during the initial set of runs conducted, and are listed in bold in Table 5.5 and Table 5.6. Once again, these designs contained a combination of materials that yielded a value for laminate weight between designs previously found by the GA. For example, since design I contains six plies of graphite and design K contains four plies of graphite, it seemed likely that there may be a design between these two which may fit into the Pareto set (i.e., design J, which contains five plies of graphite). Similar scenarios existed for designs L and N also.

To see if the GA could locate these designs, steps utilized in the previous subsection (for locating design D) were used again for this problem. For design J, α was varied between 0.09 (which pro-

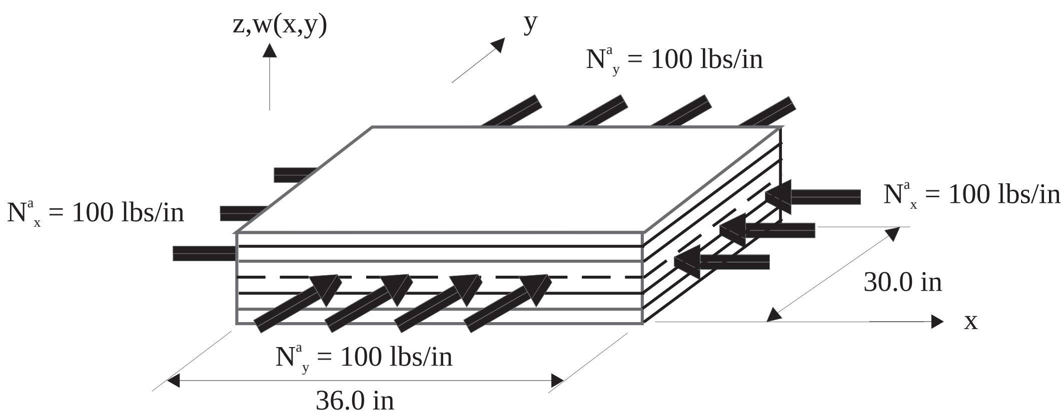


Figure 5.8: Plate configuration: biaxial loading.

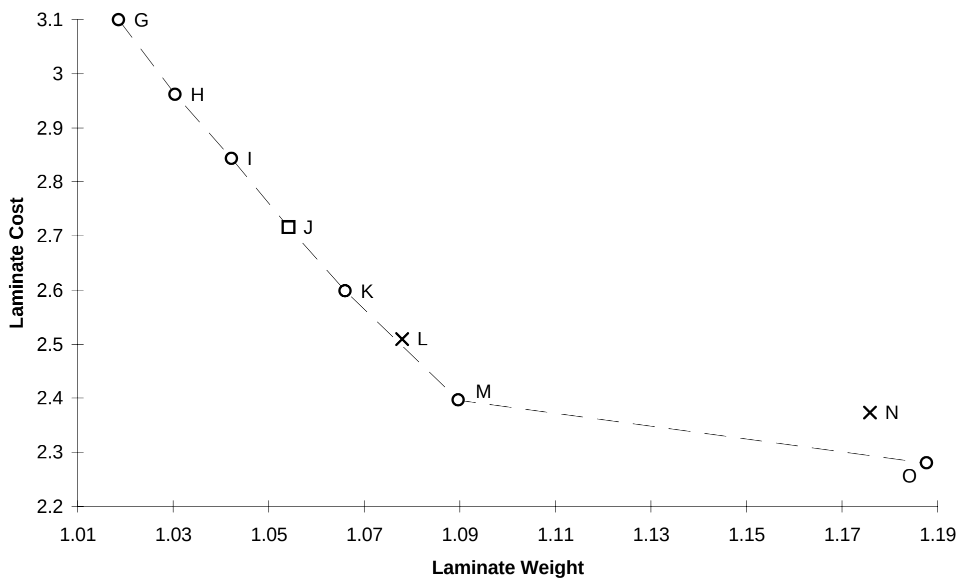


Figure 5.9: Set of Pareto-optimal designs for biaxial loading.

Table 5.5: Laminate properties for Pareto-optimal designs,biaxial loading.

Design	**Stacking sequence**	**Thickness**	**No. of plies**	α
G	$[\pm 45^{(1)}/\pm 60_3^{(1)}/90_7^{(2)}]_s$	0.1842in	15	0.05, 0.06, 0.07
H	$[\pm 45^{(1)}/\pm 60_2^{(1)}/90^{(1)}/90_8^{(2)}]_s$	0.1881in	15	0.08
I	$[\pm 60_3^{(1)}/90_9^{(2)}]_s$	0.1919in	15	0.09
J	$[\pm \mathbf{45_2^{(1)}/90^{(1)}/90_{10}^{(2)}}]_\mathbf{s}$	**0.1957in**	**15**	**0.091, 0.092**
K	$[\pm 45^{(1)}/\pm 60^{(1)}/90_{11}^{(2)}]_s$	0.1995in	15	0.10
L	$[\pm \mathbf{45^{(1)}/90^{(1)}/\pm 60_2^{(2)}/90_8^{(2)}}]_\mathbf{s}$	**0.2033in**	**15**	**not found**
M	$[\pm 45^{(1)}/\pm 60_2^{(2)}/90_9^{(2)}]_s$	0.2072in	15	0.11, 0.12,...,0.44
N	$[\mathbf{90^{(1)}/\pm 45^{(2)}/90_{13}^{(2)}}]_\mathbf{s}$	**0.2253in**	**16**	**not found**
O	$[\pm 60_2^{(2)}/90_{12}^{(2)}]_s$	0.2291in	16	0.45, 0.46,...1.00

Table 5.6: Laminate performance for Pareto-optimal designs, biaxial loading.

Design	**Weight**	**Layup cost**	**Material Cost**	**Total Cost**	**Buckling Load**
G	1.0186	1.0821	2.0165	3.0986	100.29 lbs/in
H	1.0305	1.0574	1.9037	2.9611	100.34 lbs/in
I	1.0423	1.0518	1.7908	2.8426	102.09 lbs/in
J	**1.0542**	**1.0383**	**1.6779**	**2.7162**	**100.30 lbs/in**
K	1.0661	1.0327	1.5650	2.5977	100.39 lbs/in
L	**1.0779**	**1.0574**	**1.4521**	**2.5096**	**99.95 lbs/in**
M	1.0898	1.0574	1.3393	2.3967	100.03 lbs/in
N	**1.1759**	**1.0731**	**1.3006**	**2.3738**	**100.64 lbs/in**
O	1.1878	1.0923	1.1878	2.2800	102.42 lbs/in

duced design I) and 0.1 (which produced design K) in increments of 0.001 For designs L and N, α was varied between 0.10 and 0.11, and 0.44 and 0.45, respectively. Although the GA was able to locate design J (repesented by a "$\square$" in Figure 5.9) for $\alpha = 0.092$, 0.093, it was unable to find designs L or N. Figure 5.10 shows that varying α between 0.10 to 0.11, the fitness value for design M becomes more attractive than design K, while design L's fitness is substantially higher than both of these during the transition. Similarly, the fitness of design N is never simultaneously lower than values attained for designs M or O, see Figure 5.11. In addition, by looking at Figure 5.9, designs L and N make the Pareto-optimal curve non-convex, supporting the claim made by Das and Dennis [43] that such design points cannot be found with a convex combination of objective functions. Thus, it was impossible for the GA to find these designs which are represented with an "$\times$" in Figure 5.9.

The general buckling mode shape for the entire set of Pareto-optimal designs found for biaxial loading are shown in Figure 5.12. Once again, the plate deforms into a half sine wave in both the x and y directions.

5.5 Concluding Remarks

Through simple modifications, the basic structure of the genetic algorithm used in the previous chapter was configured to optimize simply supported composite plates comprised of two materials. By utilizing two chromosome strings, ply orientation angles and material types could be represented individually, allowing for a straight forward transition from single material to multiple material stacking sequences. Using materials that possessed significant differences in material cost and performance, a multi-objective optimization problem was formulated to design high performance

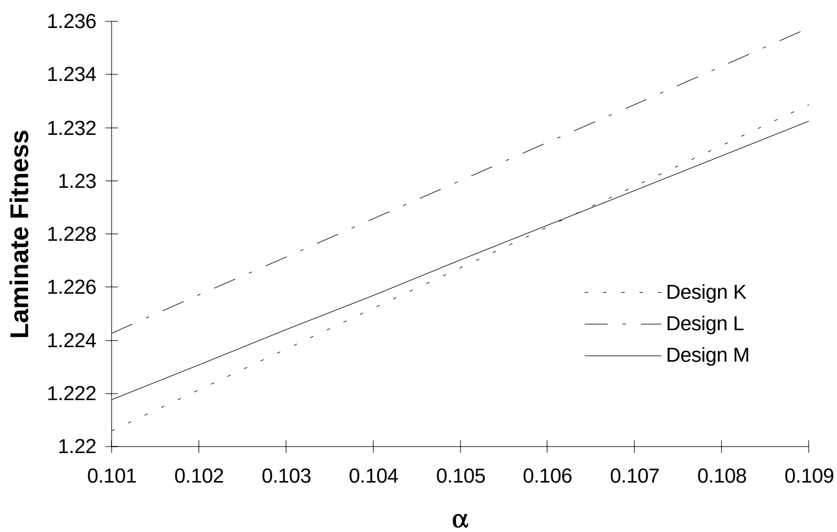


Figure 5.10: Fitness comparison between designs K, L, and M for specific values of α.

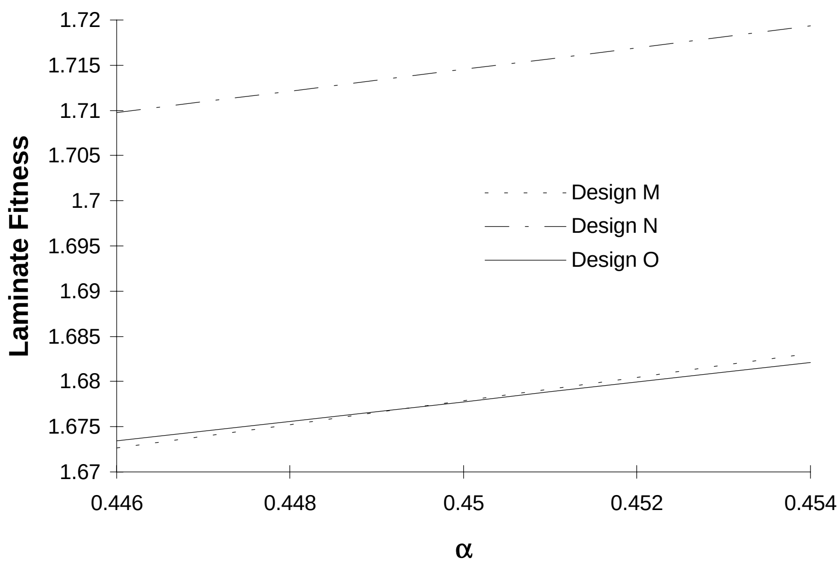


Figure 5.11: Fitness comparison between designs M, N, and O for specific values of α.

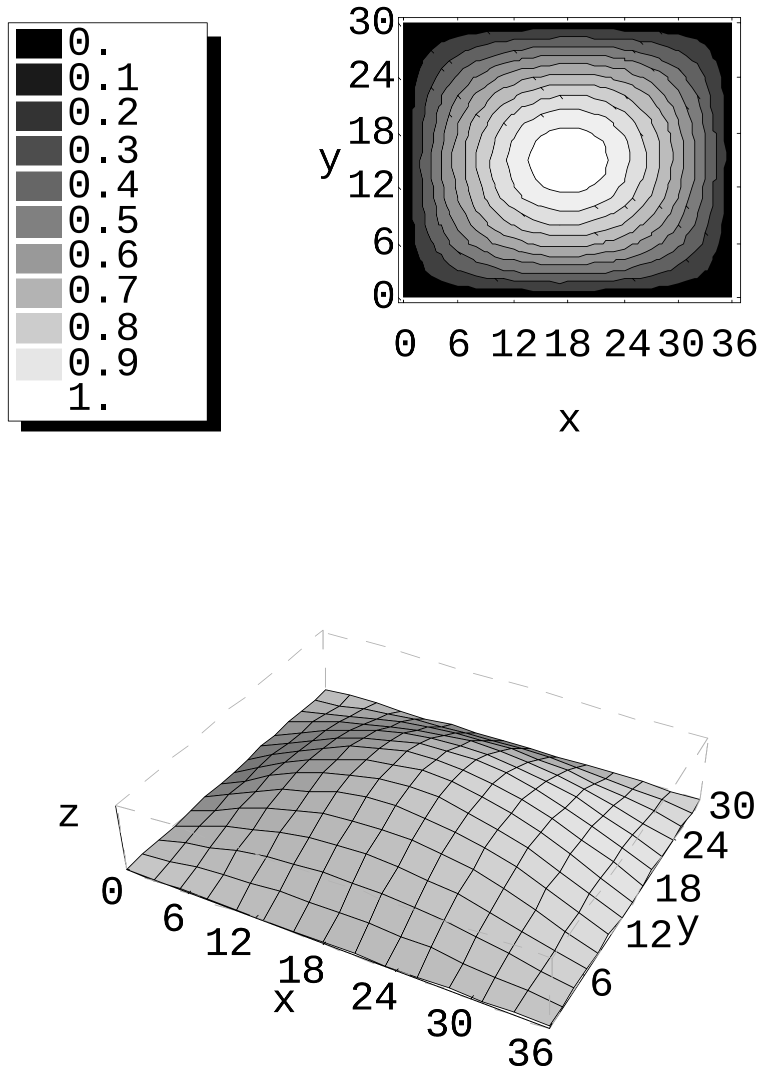


Figure 5.12: Buckling mode shapes for biaxial loading.

laminates at the lowest possible cost.

The multi-objective optimization was carried out by first formulating separate weight and cost objective functions, and then forming a convex combination of the objectives. Compared to other multi-objective schemes, this method was simple to implement, requiring no additional modifications to the GA, since laminate fitness was still represented by a single value (i.e., by the combined cost and weight objective functions). However, results showed that this was not the most viable means of multi-objective optimization since it prevented the GA from finding the entire set of Pareto-optimal designs. This fault lies not with the GA, but with the fact that there is no convex combination of objective function values that will yield a Pareto-optimal point if the point does not lie on the Pareto-optimal curve, a phenomenon which occurred in this study. Nonetheless, further research is required to either improve on the ideas discussed in the chapter, or find better methods (such as those described in Section 5.1) when using GAs for multi-objective optimization problems. The multi-objective scheme used in this work also required the scale factor to be adjusted in fine increments in order to obtain the set of Pareto-optimal designs for each loading condition, making this method somewhat computationally expensive.

Chapter 6

Conclusions

Composite laminate design and optimization requires discrete programming in order to find the correct number of plies with thicknesses, orientation angles, and material types which are usually restricted to a discrete set of values. Genetic algorithms are one of the few optimization tools available that are well suited to such discrete problem solving environments. The main goal of this work was to try and improve the computational efficiency of the GA (its most serious drawback), and demonstrate the GA's ability to be easily adapted to different types of composite laminate design optimization problems. Two different versions of a genetic algorithm, GA–I and GA–II, were developed specifically to accomplish these tasks.

The GA–I algorithm was created to work with a single chromosome string, making it well suited for designing composite laminates comprised of plies with a single prescribed thickness and material type. However, since GAs are well known for being expensive optimization tools, the main purpose of the GA–I algorithms was to provide a solid test bed for the implementation of various selection to try and improve the overall performance and reliability of the GA. Typically, an elitist (EL) selection scheme is implemented where the worst design from the child population is replaced with the best design from the parent population to ensure that good designs found previously are not lost. A potential problem with the EL scheme is that important genetic information that may exist in other desirable laminates from the parent population is lost

Thus, three different versions of a multiple elitist (ME) strategy were implemented to preserve more information about good laminate designs from the parent population. The first two implementations combined both the parent and child populations, and arranged them from best to worst according to their fitness values. A prescribed number of laminates (designated N_k) were then selected from the combined population to fill the first part of the new generation. Laminates from the child population filled the remainder of the new population, with the restriction that each child be represented only once. Child laminates were selected by searching from the top of the ranked child population (the ME_1 scheme), or randomly (the ME_2 scheme). In a third scheme (ME_3 selection), successive populations were constructed by creating $P - N_k$ children (where P is the population size), and combining them with the top N_k laminates from the parent population. Initial findings showed that small values of N_k provided a highly explorative search of the design space, while larger values produced more of an exploitative search. This provided the basis for developing three additional selection schemes. These new variable elitist (VE) schemes used different values of N_k throughout an optimization run to determine the effects of using an explorative algorithm at the beginning of the search, and more of an exploitative algorithm towards the end of the search.

The various selection schemes were rigorously tested on a composite optimization problem which involved maximizing the twisting displacement of a cantilevered plate subjected to a bending moment along the x-axis. For the initial plate configuration, results showed that the new selections schemes produce small improvements in the GAs computational cost when compared to using the standard EL scheme, while maintaining a high level of reliability. Final population richness, the number of designs in the final population with very good fitness, was increased substantially also. When modifications were made to increase the difficulty of design problem, the ME and VE schemes were much more successful in finding the optimal design than the EL scheme was, with the VE_1 and VE_2 schemes producing reliability percentages as much as 96 percentage points higher than those for the EL scheme for certain values of N_k. Overall, the VE_1 scheme provided the best improvements in algorithm reliability and efficiency.

To demonstrate the flexibility of the GA structure, the GA–II algorithm was devised to handle more complex composite laminate configurations constructed from multiple materials. The modified GA utilized two chromosome strings to represent the composite laminate. The first string defined the orientation angle of each ply, and the second string defined a ply's material type. By using two different chromosome strings, only small modifications to the various genetic operators were required. The two chromosome string concept was tested on a multi-objective optimization problem which involved minimizing laminate cost and weight of a simply supported composite plate subjected to various in-plane loading conditions, and numerous constraints. The cost of the laminate was based on the required layup time and by its material make-up. To achieve high performance laminates at the lowest cost, two materials were allowed in the stacking sequence, one with high strength and cost, the other with lower strength and cost. The optimization formulation was carried out by determining separate cost and weight objective functions. A convex combination of these two objectives was used for laminate fitness, and thus required no additional modifications to the GA. To obtain a Pareto-optimal set of designs, the influence of cost and weight on the overall fitness of a laminate configuration is adjusted from one extreme to the other by adjusting the convex combination accordingly.

Results showed that the multi-objective scheme employed in this work was not the best choice for the GA since some of the Pareto-optimal points did not lie on the convex part of the Pareto-optimal curve, making them impossible to find. Furthermore, the designs that were found required small steps in the combination parameter which increased the computational cost of the problem significantly. Overall, the two chromosome concept proved useful in designing composite laminates with multiple materials and demonstrated how GAs can be easily adapted to more complex problems. When this concept was coupled with a simple multi-objective scheme, a reasonable means of designing high performance, low cost composite laminates was realized.

Future Work

In order to make GAs more competitive with traditional optimization tools, further research must be conducted to improve their efficiency. The effort put forth in this work to help achieve this goal can be improved upon in several different areas, such as tailoring the implementation probabilities of the various genetic operators or making further refinements in the different selection schemes. The Variable Elitist selection schemes provided the GA with both explorative and exploitative search capabilities, but could be improved upon by determining when and by how much these capabilities should be utilized throughout the search. Another shortcoming of GAs is that their performance is highly problem specific. Thus, future research work needs to address what general improvements can be made to GAs that will apply to a wide range of optimizations problems. Multi-objective optimization using GAs has been successfully implemented in many different ways. But currently, there is no information on which method is most effective, or the type of problems that each method is best suited for. Although the multi-objective scheme implemented in this work displayed some weaknesses, further research may provide significant improvements in its efficiency and reliability that would compliment the ease with which it can be accommodated into the genetic algorithm.

Bibliography

[1] Alliot, J., Gruber, H., Joly, G., and Schoenauer, M., "Genetic Algorithms for Solving Air Traffic Control Conflicts." *Proceeding of the* 9^{th} *Conference on AI Applications*, 1993.

[2] Davis, L., "Job Shop Scheduling with Genetic Algorithms." *Proceedings of the First International Conference on Genetic Algorithms*, 1985, pp. 136–140.

[3] Giffler, B. and Thompson, G. L., "Algorithms for Solving Production Scheduling Problems." *Operations Research*, Vol. 8, 1969, pp. 90–99.

[4] Jones, R.M., "Mechanics of Composite Materials." *Hemisphere*, New York, 1975.

[5] "Boeing 777 roll out" *Aviation Week and Space Technology*, April 11, 1994, pp. 36–58.

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

[7] Goldberg, D. E., "Genetic Algorithms in Search, Optimization, and Machine Learning." *Addison-Wesley Publishing Company, Inc.*, Reading, MA, 1989.

[8] Le Riche, R., "Optimization of Composite Structures by Genetic Algorithm." Ph.D. Dissertation, Virginia Polytechnic Institute and State University, 1994.

[9] Gürdal, Z., Haftka, R. T., and Hajela, P., "Design and Optimization of Laminated Composite Materials." To be published.

[10] Le Riche, R., Haftka R. T., "Optimization of Stacking Sequence Design for Buckling Load Maximization by Genetic Algorithm." *Proceedings of the* 33^{rd} *AIAA/ASME/ASCE/AHS/ASC SDM Conference*, Dallas, TX, 1992.

[11] Orvosh D. and Davis L., "Shall We Repair? Genetic Algorithms, Combinatorial Optimization, and Feasibility Constraints." *Proceedings of the* 5^{th} *International Conference on Genetic Algorithms*, University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers, 1993, p. 650.

[12] Le Riche, R., and Haftka, R. T., "Improved Genetic Algorithm for Minimum Thickness Composite Laminate Design." *Proceeding of the International Conference on Composite Engineering*, Aug 28–31, New Orleans, LA, 1994.

[13] Kogiso, N., Watson, L. T., Gürdal Z., Haftka, R. T., and Nagendra, S., "Design of Composite Laminates by a Genetic Algorithm with Memory." *Mechanics of Composite Materials and Structures*,Vol. 1, 1994, pp. 95–117.

[14] Kogiso, N., Watson, L. T., Gürdal Z., and Haftka, R. T., "Genetic Algorithms with Local Improvement for Composite Laminate Design." *Structural Optimization*, Vol. 7, No. 4, 1994, pp. 207–218.

[15] Malott, B., and Averill, R. C., "Use of Genetic Algorithms for Optimal Design of Laminated Sandwich Panels with Bend Twisting Coupling." *AIAA PAper 96-1538-CP*, 1996.

[16] Yamazaki, K., "Two-Level Optimization Technique of Composite Laminate Panels by Genetic Algorithms." *AIAA Paper 96-1539-CP*, 1996.

[17] Bäck, T., Schwefel, H., "An Overview of Evolutionary Algorithms for Parameter Optimization." *Evolutionary Computation*, Vol. 1, 1993, pp. 1–23.

[18] Buckles, B. P., and Petry, F. E., "Genetic Algorithms." *IEEE Computer Society Press*, Los Alamitos, CA. 1992.

[19] Foye, R., "Advanced Design for Advanced Composite Airframes." *Airforce Materials Laboratory*, Wright–Patterson Air Force Base, Ohio, AFML TR–69–251, 1969.

[20] Waddoups, M. E., "Structural Airframe Application of Advanced Composite Materials–Analytical Methods." *Air Force Materials Laboratory*, Wright–Patterson Air Force Base, Ohio, AFML–TR–69–101, 1969.

[21] Verette, R. M., "Stiffness, Strength, and Stability Optimization of Laminated Composites." *Northrop Aircraft Corporation*, Report NOR–70–138, Hawthorne, CA., 1970.

[22] Konishi, D. Y., and Lee, K. J., "Strength and Stability of an Unstiffened Panel Under Multiple Inplane Loads" *Advanced Composite Design Guides,* 3^{rd} *edn.*, Airforce Materials Laboratory, Wright–Paterson Air Force Base, Ohio, 1973.

[23] Schmit, L. A., and Farshi, B., "Optimum Laminate Design for Strength and Stiffness." *International Journal for Numerical Methods in Engineering*, Vol. 7, No. 4., pp. 519–536, 1973.

[24] Schmit, L. A., and Farshi, B., "Optimum Design of Laminated Fiber Composite Plates." *International Journal for Numerical Methods in Engineering*, Vol. 11, No. 4, 1977, pp. 623–640.

[25] Gürdal, Z., and Haftka, R. T., "Optimization of Composite Laminates." Presented at the *NATO Advanced Study Institute on Optimization of Large Structural Systems*, Berchtesgaden, Germany, Sept. 23–Oct. 4, 1991.

[26] Haftka, R. T., and Walsh, J. L., "Stacking Sequence Optimization for Buckling of Laminated Plates by Integer Programming." *AIAA Journal*, Vol. 30, No. 3, 1992, pp. 814–819.

[27] Nagendra, S., Haftka, R. T., and Gürdal, Z., "Stacking Sequence Optimization of Simply Supported Laminates with Stability and Strain Constraints. "*AIAA Journal*, Vol. 30, No. 8, 1992, pp. 2132–2137.

[28] Mesquita, L., and Karmat, M. P., "Optimization of Stiffened Laminated Composite Plates with Frequency Constraints." *Engineering Optimization*, Vol. 11, 1987, pp. 77–86.

[29] Hajela, P., "Genetic Search – An Approach to the Non-convex Optimization Problem." *AIAA Journal*, Vol 28, No. 7, 1990, pp. 1205–1210.

[30] Rao, S. S., Pan, T. S., and Venkayya, V. B., "Optimal Placement of Actuators in Actively Controlled Structures Using Genetic Algorithms." *AIAA Journal*, Vol. 29, No. 6, 1990, pp. 942–943.

[31] Callahan, K. J., and Weeks, G. E., "Optimum Design of Composite Laminates Using Genetic Algorithm." *Composite Engineering*, Vol. 2, No. 3, 1992, pp. 149–160.

[32] Nagendra, S., Haftka, R. T., and Gürdal, Z., "Design of Blade Stiffened Composite Panels by a Genetic Algorithm." *Proceedings of the* 34^{th} *Structures, Structural Dynamics, and Materials Conference*, La Jolla, CA, April 19–21, 1993, pp. 2418–2436.

[33] Nagendra, S., "Optimal Stacking Sequence Design of Stiffened Composite Panels with Cutouts." Ph.D. Dissertation, Virginia Polytechnic Institute and State University, June 1993.

[34] Nagendra, S., Jestin, D., Gürdal, Z., and Watson, L. T., "Improved Genetic Algorithm for the Design of Stiffened Composite Panels." *Computers & Structures*, Vol. 58, No. 3, 1996, pp. 543-555.

[35] Schwefel, H., –P., "Collective Phenomena in Evolutionary Systems." *The* 31^{st} *Annual Meeting of the International Society for General System Research*, Budapest, 1987, pp. 1025-1033.

[36] Gürdal, Z., Haftka, R. T., and Nagendra, S., "Genetic Algoritm for the Design of Laminated Composite Panels." *SAMPE Journal*, Vol. 30, No. 3, May/June 1994, pp. 29-35.

[37] Kuhn, P., and Peterson, J. P., "Strength Analysis of Stiffened Beam Webs." *NACA TN 1364*, 1947.

[38] Whitney, J. M., "Structural Analysis of Laminated Anisotropic Plates". *Technomic Publishing Company, Inc.*, Lancaster,PA., 1989.

[39] Langhaar, H. L., "Energy Methods in Applied Mechanics". *John Wiley and Sons, Inc.*, New York, N.Y., 1962.

[40] Kassapoglou, C., "Simultaneous Cost and Weight Minimization of Composite Stiffened Panels Under Compression and Shear". *Composites Part A*, Vol. 28A, 1997, pp. 419-435.

[41] Shaffer, J. D., "Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms". Unpublished Doctoral Dissertation, Vanderbilt University, Nashville, 1984.

[42] Belegundu, A. D., Murthy, P. L. N., "A New Genetic Algorithm for Multi-Objective Optimization". *6th AIAA/NASA/ISMO Symposium on Multidisciplinary Analysis and Optimization*, Bellevue, WA, Sept. 4–6, 1996, pp. 1727–1736.

[43] Das, I., Dennis, J. E., "A Closer Look at Drawbacks of Minimizing Weighted Sums of Objectives for Pareto Set Generation in MultiCriteria Optimization Problems." Dept. Of Computational and Applied Mathematics Tech Report 96–36. Rice University, Houston, TX, 1996.

Appendix A

Convergence Study For Buckling Load Calculation

Two studies are provided here to show the convergence characteristics of the critical buckling load used in the design problem presented in Chapter 5. The first study shows convergence characteristics for uniaxial plate loading, and the second for the biaxial loading case.

Designs A and F, two of the Pareto–optimal designs found by the GA for the uniaxial load case were used in the first study, see Table A.1. The values for m and n used in Fourier sine series expansion functions was varied from 1 to 10 in increments of 1. Figure A.1 shows that there is little change in the magnitude of the critical buckling load as m and n increase.

Designs G and O from the set of Pareto–optimal designs found for the biaxial loading case were used in the second study, see Table A.1. Once again, there is little change in the buckling load as m and n are incremented from 1 to 10, see Figure A.2. These results show that using a value of $m, n = 2$ for the buckling load calculation in Chapter 5 was sufficient to generate accurate results, for both uniaxial and biaxial loading conditions.

Table A.1: Pareto-optimal designs used in convergence study.

Design	**Stacking sequence**	**Loading condition**
A	$[\pm 45_3^{(1)}/\pm 45_2^{(2)}/90^{(2)}]_s$	unixaxial ($N_x^a = 100$ lbs/in)
F	$[30^{(2)}/60^{(2)}/-30^{(2)}/-60^{(2)}/90_8^{(2)}]_s$	unixaxial ($N_x^a = 100$ lbs/in)
G	$[\pm 45^{(1)}/\pm 60_3^{(1)}/90_7^{(2)}]_s$	biaxial ($N_x^a = N_y^a = 100$ lbs/in)
O	$[\pm 60_2^{(2)}/90_{12}^{(2)}]_s$	biaxial ($N_x^a = N_y^a = 100$ lbs/in)

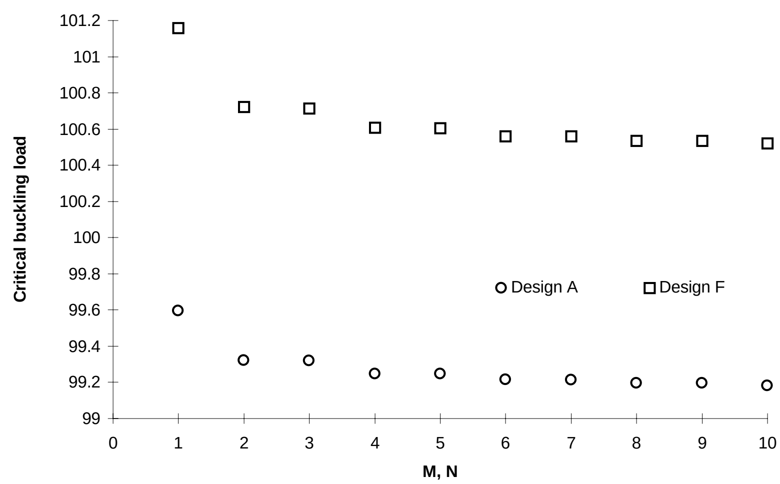


Figure A.1: Convergence of critical buckling load (uniaxial loading).

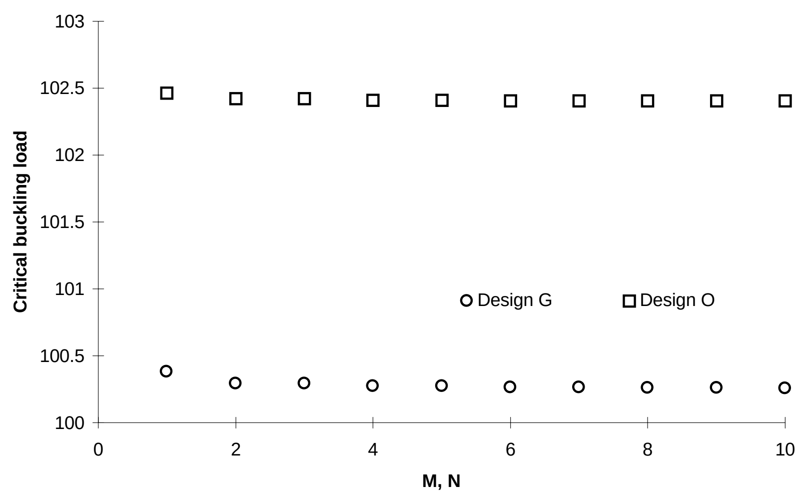


Figure A.2: Convergence of critical buckling load (biaxial loading).

Vita

Grant Ayodele Ebun Soremekun was born in 1971 in the city of Hamilton, Ontario, Canada. He spent the first part of his life in Thornhill, located a few miles outside of Toronto. Much of his childhood was spent playing competitive tennis throughout the province of Ontario. After completing highschool in the Spring of 1990, he began looking at schools in the United States, choosing Virginia Tech to provide the necessary competition that would further improve his tennis skills. Although his tennis career did not materialize he continued on with his studies, graduating with a Bachelor degree in Aerospace Engineering in the Spring of 1994. The following summer, he offered his services to Dr. Gürdal for conducting research on genetic algorithms, which he kindly accepted. As a graduate student in Virginia Tech's department of Engineering Science and Mechanics, research work continued for two and a half years, resulting in this document – the final step towards the attainment of his Masters degree in Engineering Mechanics.

Grant Soremekun

February 6 , 1997.