Optimization of Composite Stacking Sequence Using Mesh Adaptive Direct Search

Ashutosh Bhujbal

Department of Mechanical Engineering
McGill University
Montreal, Quebec

October 2017

A thesis submitted to
McGill University
in partial fulfillment of the requirements of the degree of
Master of Engineering

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere gratitude to my supervisor Prof. Michael Kokkolaras for his continuous support and advise. It is needless to say that, he has been externally patient in providing great guidance and motivation during my research and writing of this thesis. I could not have imagined having a better adviser than Prof. Kokkolaras.

I am also grateful to my fellow researcher Ibrahim Chamseddine for time to time fruitful discussions and motivation. A special thanks to Catherine Léger-Beaulieu for translating the abstract in French.

Lastly, I am thankful to my friends and family, especially my parents and sister, for their encouragement, moral support, and endless patience throughout my education.

ABSTRACT

Fiber-reinforced polymer (FRP) composites have several advantages over conventional isotropic materials used in land, marine and aerospace applications. Most importantly, they provide the flexibility to tailor their material properties, and changing the fiber orientations in the material is one of the ways to do so. However, this makes the design of laminated composite structures more complex and tedious due to the large number of design variables (fiber orientation angles). This offers an opportunity to use optimization techniques to design the best suited structure as per requirements of the specific application. Different optimization methods have been used in the past to optimize the stacking sequence of laminated composites, and genetic algorithms (GAs) are reported to be the most popular to solve such design problems. However, GAs generate solutions according to arbitrarily defined termination criteria, and the optimality of such solutions cannot be characterized. Moreover, they require a high number of function evaluations, making them computationally expensive compared to other evolutionary algorithms and direct-search methods. This motivates us to try and test other derivative-free optimization methods to address the drawbacks associated with GAs. In this work, we use the Mesh Adaptive Direct Search (MADS) algorithm, and compare its performance to GAs when solving composite stacking sequence optimization problems. Specifically, we compare the convergence rate, computational cost and the quality of the solution.

RÉSUMÉ

Les composites faits à partir de fibre renforcé de polymère ont plusieurs avantages par rapport aux matériaux isotropiques conventionnels utilisés pour de multiples applications terrestres, marines et aérospatiales. L'aspect le plus important des composites est leur capacité à modifier leurs propriétés; entres autres par le changement de l'orientation des fibres dans le matériel Cependant, le design des composites est plus complexe dû aux nombreuses variables présentes dans le matériel (angle d'orientation des fibres). Ceci offre une opportunité d'utiliser des techniques d'optimisation dans le but de concevoir des structures plus appropriées aux demandes spécifiques d'application. Diverses méthodes d'optimisation ont été utilisées dans le passé afin d'optimiser la séquence d'empilement des composites stratifiés et l'algorithme génétique (GAs) s'avère être le plus populaire afin de résoudre ce type de problème. Toutefois, les GAs génèrent des solutions où les critères sont arbitraires et dont l'optimum ne peut être caractérisé. De plus, le processus est couteux en termes de calcul comparativement aux autres algorithmes évolutifs et aux méthodes de recherche directes puisque cette technique nécessite un nombre élevé d'évaluation des fonctions. Les faiblesses de l'algorithme génétique nous motivent à tester d'autres méthodes d'optimisation sans dérivés. Dans cette recherche, nous utilisons l'algorithme Mesh Adaptative Direct Search (MADS) que nous comparons à l'algorithme génétique afin de résoudre des problèmes d'optimisation de séquence d'empilement des composites. Plus précisément, nous comparons le taux de convergence, le coût de calcul et la qualité de la solution.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Derivative-free optimization is useful in simulation-based engineering design, because in most of the engineering applications functions defining the optimization problem cannot be explicitly expressed in closed form mathematical expressions. This makes it either difficult or impossible to use gradient-based optimization methods, since gradients are either not available or cannot be approximated reliably. Moreover, these functions are often, if not always, highly non-linear, noisy, and discontinuous. Meta-heuristic techniques such as Genetic Algorithms, Simulated Annealing and other nature-inspired evolutionary algorithms have been successfully used to solve such engineering optimization problems in the past. Genetic Algorithms are robust and can find application in a broad range of optimization problems.

Since their early development, direct search methods (DSM) have proven to be robust and reliable in derivative-free optimization. Mesh Adaptive Direct Search (MADS) [1] introduced by Audet and Dennis in 2006 is a class of algorithms belonging to direct search methods. It has been successfully implemented in a software package called NOMAD (Non-linear Optimization by Mesh Adaptive Direct search). NOMAD is a black-box optimization software which can handle non-linear, non-smooth, non-convex objectives with mixed type of variables (continuous, discrete, categorical),

and can also handle non-linear constraints [1]. Furthermore, MADS is backed by a convergence analysis based on Clark's calculus [2].

In this work, we use the Mesh Adaptive Direct Search in stacking sequence optimization of laminated composites and compare its performance against GAs.

1.1 Motivation

Design of laminated composite structures is a complex and tedious task when compared to the design of structures using conventional materials (metals and alloys) due to the higher number of design variables involved. This is because, composites provide the ability to tailor material behavior and structural response, and changing the orientation of fibers in the material is on of the ways to achieve this. Considering fiber orientations as design variables essentially increases the number of design variables. Hence, to design more efficient composite structures, different optimization techniques have been used in the past, and Genetic Algorithms (GAs) have been reported to be the most popular among researchers. There is no doubt that genetic algorithms are a powerful tool in the field of optimization. However, genetic algorithms have some limitations such as, they require a high number of function evaluations before finding a suitable solution. Besides, they generate solutions according to randomly chosen termination criteria, and the optimality of such solutions cannot be characterized. A high number of function evaluations could be a major drawback if function evaluations are performed using costly black-box programs. MADS, on the other hand, has a proven convergence analysis, which means solutions generated by MADS possess convergence properties [2]. Moreover, since its

introduction, MADS has been proved to be effective and efficient in solving highly non-linear and discontinuous problems [3]. To our knowledge it has not been used in the composite stacking sequence optimization yet. This motivates us to use MADS to solve the stacking sequence optimization problem by considering the fiber orientations as design variables. We demonstrate its effectiveness in addressing some of the drawbacks associated with the genetic algorithms.

1.2 Objective

The primary objective of this thesis is to use the Mesh Adaptive Direct Search algorithm to solve composite stacking sequence optimization problems and compare its performance to the genetic algorithms. To assess the performance of both algorithms under consideration, we solve two optimization problems in this work. The first problem is a single objective unconstrained minimization with mixed variable formulation, and the second problem is a bi-objective maximization problem formulated with and without constraints. We analyze the results obtained to compare MADS with GAs on parameters such as convergence rate, computational cost and the quality of the solution.

1.3 Thesis Overview

This thesis is organized in 6 chapters. Chapter 1 briefly introduces the idea behind this work, the motivation and the objective defined for this work. Chapter 2 describes the definition, classification, advantages and limitations of composites. It also provides a detailed background about the mechanics of composite materials along with equations and notations used in this work. In Chapter 3, we discuss about

design optimization of laminated composites. A brief review on different optimization methods used in the past is presented in this chapter. Information on problem formulations, the nested genetic algorithm approach, and all-in-one MADS approach is presented in Chapter 4. A brief description of MADS is also presented in this chapter. Algorithm settings used and the numerical results obtained are reported in chapter 5. Finally, conclusion and suggestions for future work are presented in chapter 6.

CHAPTER 2
Introduction To Composites

2.1 Introduction

2.1.1 What are composites?

Composites are defined as materials formed by deliberately combining two or more constituents with significantly different physical and/or chemical properties, which after combining together produce a heterogeneous material with considerably different physical properties than its constituents. Unlike metal alloys, in composites the individual constituents remain distinct. This definition includes a wide classification of materials such as: fiber-reinforced polymers/plastics, reinforced concrete, particle filled plastics, rubber reinforced plastics, wood laminates, ceramic mixtures, metal-ceramic mixtures, etc. [4].

2.1.2 History of Composites

Early applications of composites can be traced back to history when it was found that, combining two materials could lead to a material with superior performance than that of its constituents. For instance, mud bricks reinforced with straw were used in ancient Israelites in Egypt, Samurai swords and Damascus gun barrels were made from combined layers of iron and steel for greater strength. Mongols made their bows from cattle tendons, wood and silk bonded together using the glue made from

animal hoofs and bones [4]. Since then composites have evolved significantly, and what we have today are advanced composite materials with much superior properties than the early composite materials used. In fact, some composites have properties which are either superior or at par with some of the advanced metals and alloys. The term advanced composites is generally associated with composites composed of unidirectional long fiber (with modulus greater than $200,000\ MPa$) reinforced layers, with grater than 50% fibers by volume [4].

Use of modern composites is believed to began in 1937 when it was found that fiberglass can be effectively used as a reinforcement. Soon, the advancements in composites took a fast pace during World War II when aircraft manufacturers began to adopt fiber reinforced polymer composites in structural and semi-structural parts of airplanes. The invention of high strength fibers, such as carbon fiber (1961), boron fiber (1965) and aramid fiber (1971) led to an increase in the use of fiber reinforced polymer (FRP) composites not only in aerospace industry but also in automobiles, armors, sports, medical applications, etc. [4].

2.1.3 Classification

Composites are mainly classified in three different categories depending on the type, geometry, and the orientation of the reinforcement phase as shown in figure 2–1. Among these types, continuous fiber composites are the most efficient in terms of strength and stiffness. This class of composites can further be divided into sub-categories based on the matrix material used such as, Polymer Matrix Composites,

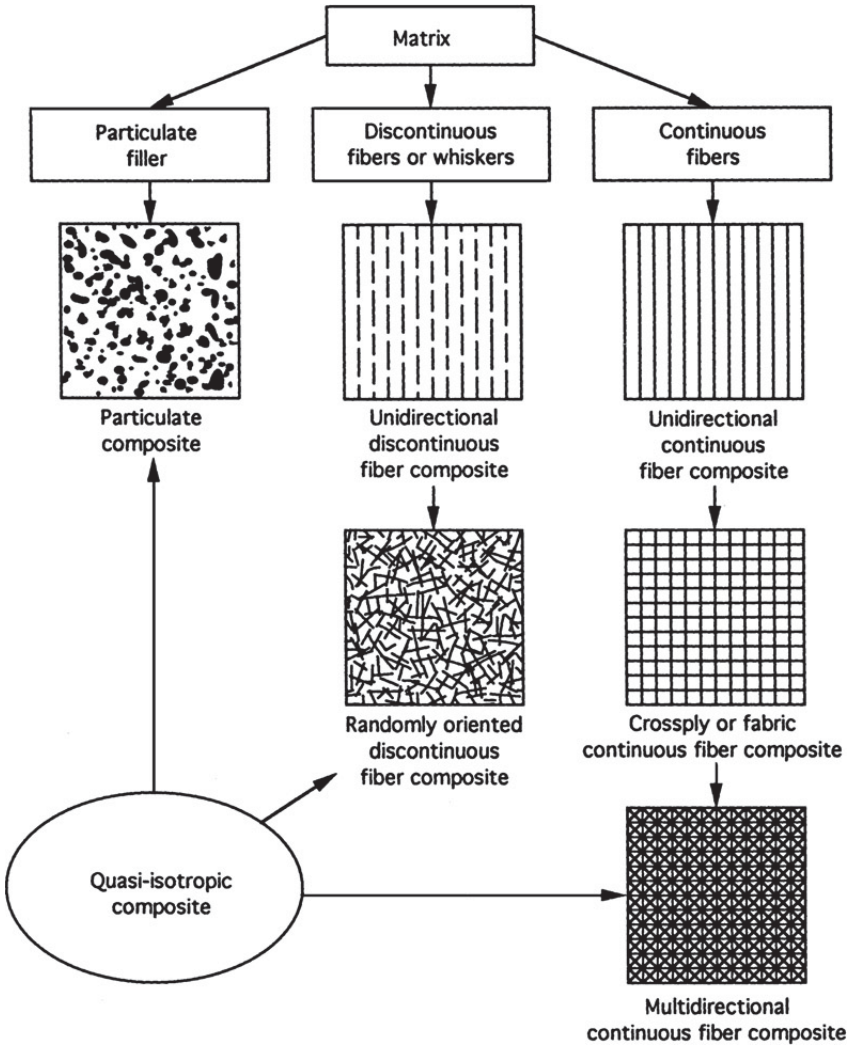


Figure 2–1: Classification of composite materials based on the type of reinfocement phase. (Source: [5])

Metal Matrix Composites, and Ceramic Matrix Composites. Polymer matrix composites are of more interest in terms of structural efficiency. Hence, they are usually the first choice for structural applications among other types [5, 6]. FRPs have several

advantages over the conventional isotropic materials used in structural applications. Some of these advantages are listed in next section along with limitations.

2.1.4 Advantages and Limitations

Higher specific strength (strength/ mass density) and specific modulus (modulus/ mass density) make fiber reinforced polymer composite structures relatively lighter than other comparable metals and alloys used in industries. Better corrosion resistance and fatigue life, ease to mold into complex shapes, stealth (low radar visibility) are some of the other advantages of composites over metals [6]. The fact that composite structures are lighter than metallic structures is evident from figure 2–2, where structural efficiency (strength/weight) of carbon fiber/epoxy composite is compared with titanium and aluminum alloys used in the aerospace industry. It is clear from

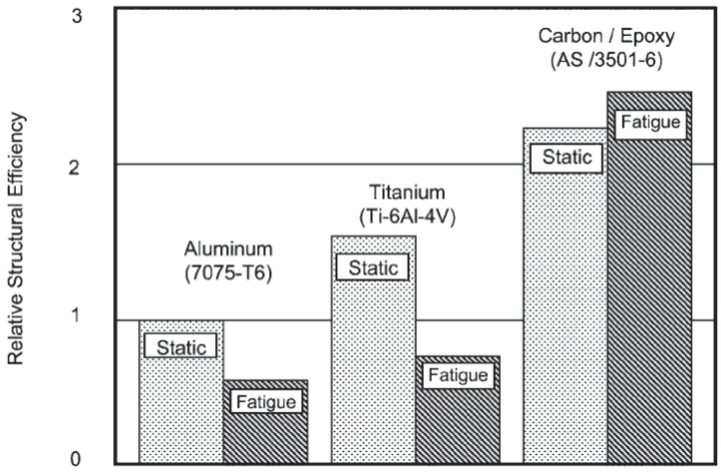


Figure 2–2: Relative structural efficiency of aerospace materials (Source:[6])

the figure that, carbon/epoxy composite is more efficient than aluminum and titanium in static as well as fatigue structural efficiency (fatigue strength/weight). This

essentially results in great weight savings, which ultimately translates into improved performance, higher payloads, longer range and fuel savings in commercial aerospace applications. Besides, the ability to tailor the material properties by selective use of fiber orientations in laminates interests structural designers the most as it gives designers more control to fine tune the structure as per the requirements of specific application.

All the above mentioned benefits come with some limitations too. For instance, the cost of advanced composite materials is higher than those of other conventional isotropic materials. Lack of generalized design guidelines, expensive and slow manufacturing techniques, shorter working temperature range, etc. are some of the other limitations of composites when compared with metals and their alloys [4, 6, 7, 8]. Moreover, unlike metals, composites are brittle, and the lack of ductility makes them more vulnerable to stress concentrations and impact loads [6]. However, with the advancement in technology and science, engineers and scientists are trying to overcome these limitations, and the use of composite structures is increasing in a variety of applications. For example, 50% of the air-frame of Boeing-787 is made of advanced composite material (mostly Carbon Fiber composites)(see fig. 2–3) which makes it about 20% lighter than aluminum designs [9].

2.2 Basic Concepts

Fiber reinforced polymer composites are made up of layers (also called plies) of a polymer reinforced with long, unidirectional (straight) fibers, as shown in figure 2–4.

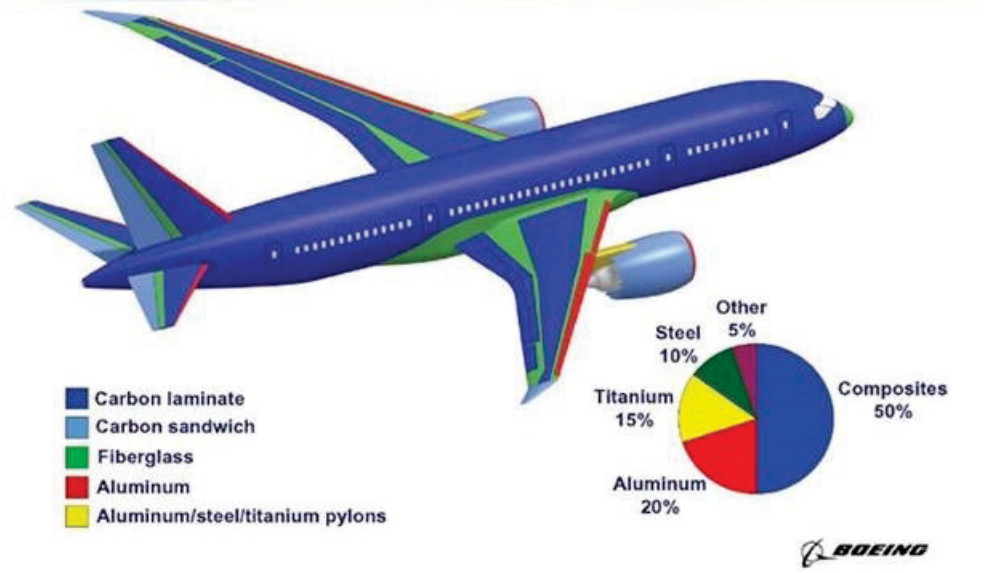


Figure 2–3: Boeing 787: Material composition (Source: The Boeing Company)

Fibers in each layer can be oriented in different directions to have different strength and stiffness.

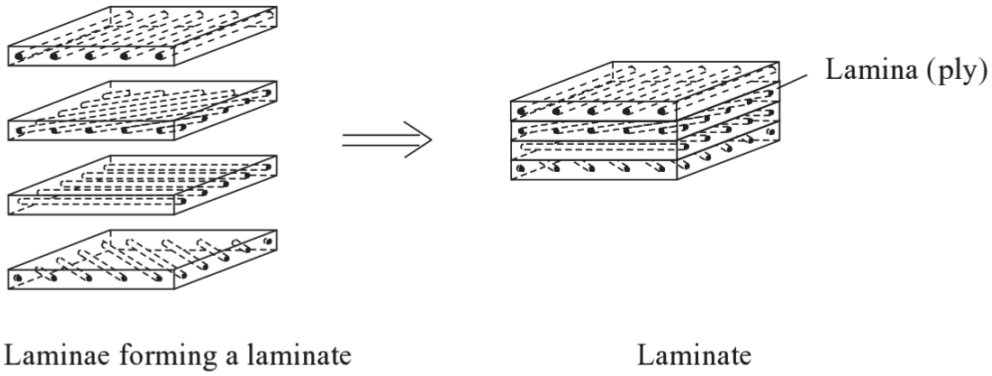


Figure 2–4: Typical laminated composite layup (Source: [7])

Before we start analyzing the structural behavior and the mechanics of composite materials, it is essential to establish some general conventions.

2.2.1 Laminate Code and Coordinate System

The stacking sequence of plies in a laminate is described by the orientation of fibers in continuous and unidirectional plies with respect to the x axis in an $x-y-z$ coordinate system as shown in figure 2–5. Here the z axis is perpendicular to the plane of the laminate. The $x-y-z$ coordinate system is also referred as "Global or Geometric or Loading" coordinate system. In the global coordinate system, a laminate can be represented by the sequence of orientation angles, θs, of fibers in individual plies. Such a representation is often called as "Laminate Code" and goes as follows:

$$[(\theta_1)_i/(\theta_2)_j/(\theta_3)_k/\cdots]$$

where, θ_i denotes the fiber orientation, θ_1 being the lowermost ply in the stack. Subscripts $i,\ j,\ k,\cdots$ denote the number of adjacent plies with same fiber orientation.

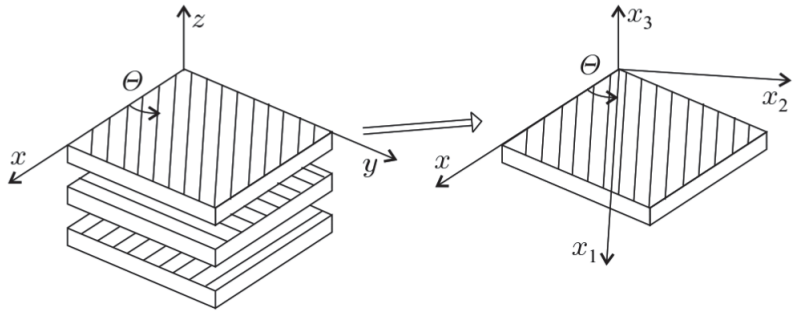


Figure 2–5: The laminate coordinate system $(x-y-z)$ and the ply coordinate system $(x_1-x_2-x_3)$ (Source: [7])

The ply coordinate system $x_1-x_2-x_3$ is generally referred as material coordinate system where axis x_1 is allied with the fibers and x_2 is perpendicular to the fibers

as shown in figure 2–5. For simplicity, the ply coordinate system $x_1 - x_2 - x_3$ will simply be referred as $1 - 2 - 3$ here after. Also please note that, θ is positive in the counterclockwise direction from x axis as shown in figure 2–5.

If there exists a symmetry about the mid plane of the laminate, such laminates are called as symmetric laminates and are represented by a subscript 's' in the laminate code as shown below:

$$[(\theta_1)_i/(\theta_2)_j/(\theta_3)_k/\cdots]_S$$

For example,

$$[0/30/90]_S \implies [0/30/90/90/30/0]$$

If the symmetry does not exist about the mid plane of the laminate, then such laminates are called unsymmetrical laminates. Figure 2–6 shows examples of both symmetric and unsymmetrical laminates with their laminate code.

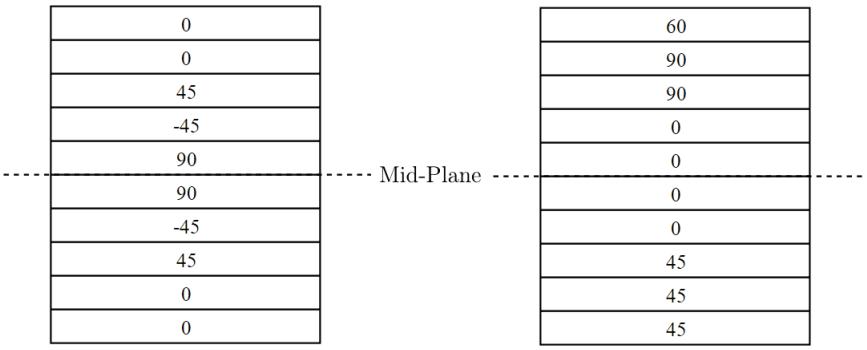


Figure 2–6: (a) Symmetric Laminate, (b) Unsymmetrical Laminate

2.2.2 Basic Lamina Properties

The basic material properties of an unidirectional ply are of vital importance. Based on the micro-mechanics, an unidirectional ply shown in figure 2–7 can be characterized by the following material properties [5]:

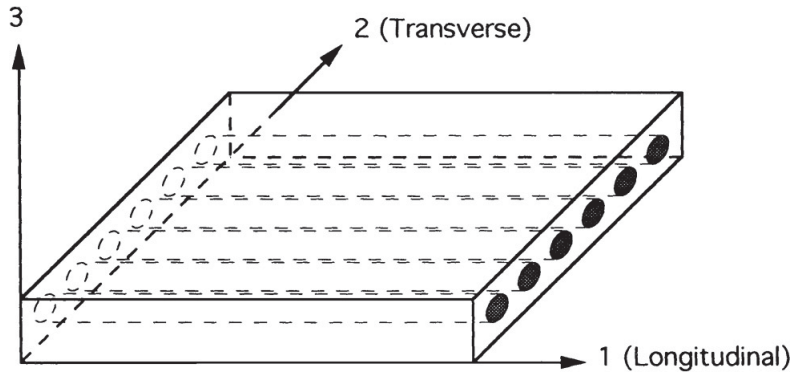


Figure 2–7: Unidirectional lamina and principal coordinate axes. (Source: [5])

E_1, E_2, E_3	Young's moduli along principle ply directions
G_{12}, G_{23}, G_{13}	Shear moduli in 1-2, 2-3, and 1-3 planes respectively (equal to G_{21}, G_{32}, G_{31} respectively)
$\nu_{12}, \nu_{23}, \nu_{13}$	Poisson's ratios (these Poisson's ratios are different from ν_{21}, ν_{32}, and ν_{31})
F_{1t}, F_{2t}, F_{3t}	Tensile strengths along principle ply directions
F_{1c}, F_{2c}, F_{3c}	Compressive strengths along principle ply directions
F_{12}, F_{23}, F_{13}	Shear strengths in 1-2, 2-3, and 1-3 planes respectively (these are equal to F_{21}, F_{32}, and F_{31} respectively)

2.3 Mechanics of Composite Materials

2.3.1 Elastic Behavior of Uni-directional Lamina

The stress and strain components at a point in a continuum for any anisotropic material as shown in figure 2–8 are related to each other by the generalized Hooke's law as shown in equations (2.3.1) and (2.3.2).

$$\begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{23} \\ \sigma_{31} \\ \sigma_{12} \\ \sigma_{32} \\ \sigma_{13} \\ \sigma_{21} \end{bmatrix} = \begin{bmatrix} C_{1111} & C_{1122} & C_{1133} & C_{1123} & C_{1131} & C_{1112} & C_{1132} & C_{1113} & C_{1121} \\ C_{2211} & C_{2222} & C_{2233} & C_{2223} & C_{2231} & C_{2212} & C_{2232} & C_{2213} & C_{2221} \\ C_{3311} & C_{3322} & C_{3333} & C_{3323} & C_{3331} & C_{3312} & C_{3332} & C_{3313} & C_{3321} \\ C_{2311} & C_{2322} & C_{2333} & C_{2323} & C_{2331} & C_{2312} & C_{2332} & C_{2313} & C_{2321} \\ C_{3111} & C_{3122} & C_{3133} & C_{3123} & C_{3131} & C_{3112} & C_{3132} & C_{3113} & C_{3121} \\ C_{1211} & C_{1222} & C_{1233} & C_{1223} & C_{1231} & C_{1212} & C_{1232} & C_{1213} & C_{1221} \\ C_{3211} & C_{3222} & C_{3233} & C_{3223} & C_{3231} & C_{3212} & C_{3232} & C_{3213} & C_{3221} \\ C_{1311} & C_{1322} & C_{1333} & C_{1323} & C_{1331} & C_{1312} & C_{1332} & C_{1313} & C_{1321} \\ C_{2111} & C_{2122} & C_{2133} & C_{2123} & C_{2131} & C_{2112} & C_{2132} & C_{2113} & C_{2121} \end{bmatrix} \begin{bmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \epsilon_{33} \\ \epsilon_{23} \\ \epsilon_{31} \\ \epsilon_{12} \\ \epsilon_{32} \\ \epsilon_{13} \\ \epsilon_{21} \end{bmatrix} \tag{2.3.1}$$

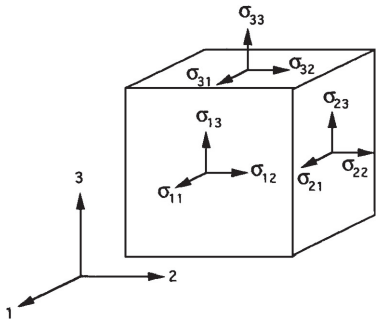


Figure 2–8: State of stress at a point in a continuum (Source: [5])

$$\begin{bmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \epsilon_{33} \\ \epsilon_{23} \\ \epsilon_{31} \\ \epsilon_{12} \\ \epsilon_{32} \\ \epsilon_{13} \\ \epsilon_{21} \end{bmatrix} = \begin{bmatrix} S_{1111} & S_{1122} & S_{1133} & S_{1123} & S_{1131} & S_{1112} & S_{1132} & S_{1113} & S_{1121} \\ S_{2211} & S_{2222} & S_{2233} & S_{2223} & S_{2231} & S_{2212} & S_{2232} & S_{2213} & S_{2221} \\ S_{3311} & S_{3322} & S_{3333} & S_{3323} & S_{3331} & S_{3312} & S_{3332} & S_{3313} & S_{3321} \\ S_{2311} & S_{2322} & S_{2333} & S_{2323} & S_{2331} & S_{2312} & S_{2332} & S_{2313} & S_{2321} \\ S_{3111} & S_{3122} & S_{3133} & S_{3123} & S_{3131} & S_{3112} & S_{3132} & S_{3113} & S_{3121} \\ S_{1211} & S_{1222} & S_{1233} & S_{1223} & S_{1231} & S_{1212} & S_{1232} & S_{1213} & S_{1221} \\ S_{3211} & S_{3222} & S_{3233} & S_{3223} & S_{3231} & S_{3212} & S_{3232} & S_{3213} & S_{3221} \\ S_{1311} & S_{1322} & S_{1333} & S_{1323} & S_{1331} & S_{1312} & S_{1332} & S_{1313} & S_{1321} \\ S_{2111} & S_{2122} & S_{2133} & S_{2123} & S_{2131} & S_{2112} & S_{2132} & S_{2113} & S_{2121} \end{bmatrix} \begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{23} \\ \sigma_{31} \\ \sigma_{12} \\ \sigma_{32} \\ \sigma_{13} \\ \sigma_{21} \end{bmatrix} \tag{2.3.2}$$

The equations (2.3.1) and (2.3.2) can be written in indicial notation as follows:

$$\begin{aligned} \sigma_{ij} &= C_{ijkl}\ \epsilon_{ij} \\ \epsilon_{ij} &= S_{ijkl}\ \sigma_{ij} \end{aligned} \tag{2.3.3}$$

where,

$$\begin{aligned} i, j, k, l &= 1, 2, 3, \\ C_{ijkl} &= \text{Stiffness Components}, \\ S_{ijkl} &= \text{Compliance Components} \end{aligned}$$

The symmetry of stress and strain tensors, $\sigma_{ij} = \sigma_{ji}$ and $\epsilon_{ij} = \epsilon_{ji}$, reduces the number of independent elastic constants from 81 to 36. And with the contracted notation[1] and using τ and γ to represent shear stress and shear strain respectively, equation

[1] The details on contracted notation are omitted for brevity. Readers may refer [5] for more details.

(2.3.1) can be written as,

$$\begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \tau_4 \\ \tau_5 \\ \tau_6 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} & C_{15} & C_{16} \\ C_{21} & C_{22} & C_{23} & C_{24} & C_{25} & C_{26} \\ C_{31} & C_{32} & C_{33} & C_{34} & C_{35} & C_{36} \\ C_{41} & C_{42} & C_{43} & C_{44} & C_{45} & C_{46} \\ C_{51} & C_{52} & C_{53} & C_{54} & C_{55} & C_{56} \\ C_{61} & C_{62} & C_{63} & C_{64} & C_{65} & C_{66} \end{bmatrix} \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \gamma_4 \\ \gamma_5 \\ \gamma_6 \end{bmatrix} \tag{2.3.4}$$

For orthotropic materials (which have three mutually perpendicular planes of material symmetry), the number of independent elastic constants, from equation (2.3.4), is reduced from 36 to 9 as shown in equation (2.3.5) [5].

$$\begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \tau_4 \\ \tau_5 \\ \tau_6 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & 0 & 0 & 0 \\ C_{21} & C_{22} & C_{23} & 0 & 0 & 0 \\ C_{31} & C_{32} & C_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & C_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & C_{66} \end{bmatrix} \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \gamma_4 \\ \gamma_5 \\ \gamma_6 \end{bmatrix} \tag{2.3.5}$$

Similarly,

$$\begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \gamma_4 \\ \gamma_5 \\ \gamma_6 \end{bmatrix} = \begin{bmatrix} S_{11} & S_{12} & S_{13} & 0 & 0 & 0 \\ S_{21} & S_{22} & S_{23} & 0 & 0 & 0 \\ S_{31} & S_{32} & S_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & S_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & S_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & S_{66} \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \tau_4 \\ \tau_5 \\ \tau_6 \end{bmatrix} \tag{2.3.6}$$

In structural applications, composites are generally used in the form of thin plates loaded only in the plane of the laminate. Therefore, it is assumed that, all the out-of-plane (direction 3) stress and strain components are zero, i.e.,

$$\sigma_3 = \tau_4 = \tau_5 = 0$$

Moreover, it is much easier to work with stress-strain relations when they are expressed in terms of the familiar engineering constants i.e. moduli and Poisson's ratios. The relationship between stress and strain expressed in terms of engineering constants can be derived by applying hypothetical normal and shear loads on the hypothetical unidirectional lamina as shown in figure 2–9.

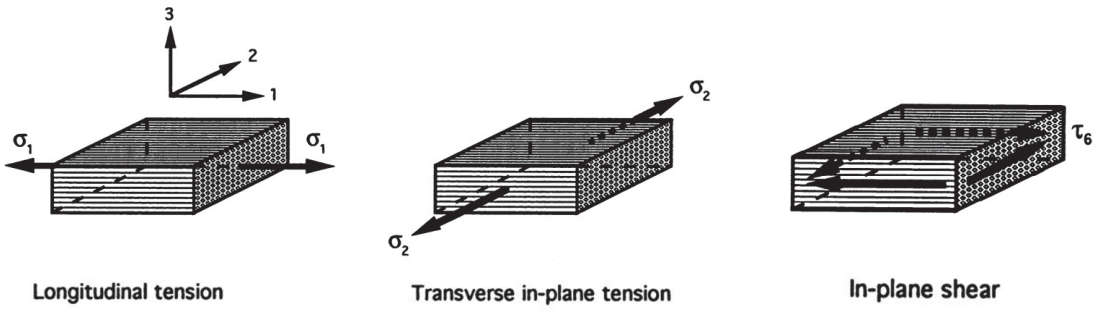


Figure 2–9: Normal and shear load applied to an uni-directional lamina (Source: [5])

1. Apply stress σ_1 in the fiber direction (Uni-axial longitudinal tensile test) [10]

$$\begin{aligned} \epsilon_1 &= \frac{1}{E_1}\,\sigma_1 \\ \epsilon_2 &= -\frac{\nu_{12}}{E_1}\,\sigma_1 \end{aligned} \tag{2.3.7}$$

2. Apply σ_2 in the matrix direction (Uni-axial transverse tensite test) [10]

$$\begin{aligned} \epsilon_2 &= \frac{1}{E_2}\,\sigma_2 \\ \epsilon_1 &= -\frac{\nu_{21}}{E_2}\,\sigma_2 \end{aligned} \tag{2.3.8}$$

3. Apply in-plane shear stress τ_6 [10]

$$\gamma_6 = \frac{1}{G_{12}}\ \tau_6 \tag{2.3.9}$$

Under the assumption that the material is linearly elastic, stress-strain relations for an uni-directional lamina can be obtained by superposing equations (2.3.7), (2.3.8), and (2.3.9).

$$\begin{aligned} \epsilon_1 &= \frac{1}{E_1}\ \sigma_1 - \frac{\nu_{21}}{E_2}\ \sigma_2 \\ \epsilon_2 &= -\frac{\nu_{12}}{E_1}\ \sigma_1 + \frac{1}{E_2}\ \sigma_2 \\ \gamma_6 &= \frac{1}{G_{12}}\ \tau_6 \end{aligned} \tag{2.3.10}$$

Equation (2.3.10) can be written in matrix form to get the strain-stress or compliance relations for an uni-directional lamina as,

$$\begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \gamma_6 \end{bmatrix} = \begin{bmatrix} \frac{1}{E_1} & -\frac{\nu_{21}}{E_2} & 0 \\ -\frac{\nu_{12}}{E_1} & \frac{1}{E_2} & 0 \\ 0 & 0 & \frac{1}{G_{12}} \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \tau_6 \end{bmatrix} \tag{2.3.11}$$

where,

$$\begin{bmatrix} \frac{1}{E_1} & -\frac{\nu_{21}}{E_2} & 0 \\ -\frac{\nu_{12}}{E_1} & \frac{1}{E_2} & 0 \\ 0 & 0 & \frac{1}{G_{12}} \end{bmatrix} = [S_{ij}] = \text{Compliance Matrix } \cdots\cdots\cdots\ i, j = 1, 2, 6$$

Similarly, a stiffness matrix, "Q", can be obtained by inverting equation (2.3.10) to get,

$$\begin{aligned} \sigma_1 &= mE_1\ [\epsilon_1 + \nu_{21}\epsilon_2] \\ \sigma_2 &= mE_2\ [\nu_{12}\epsilon_1 + \epsilon_2] \\ \tau_6 &= G_{12}\ \gamma_6 \end{aligned} \tag{2.3.12}$$

where,

$$m = [1 - \nu_{12}\nu_{21}]^{-1}$$

Equation (2.3.12) can be written in the matrix form with the stiffness components represented by Q_{ij} as,

$$\begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \tau_6 \end{bmatrix} = \begin{bmatrix} Q_{11} & Q_{12} & 0 \\ Q_{21} & Q_{22} & 0 \\ 0 & 0 & Q_{66} \end{bmatrix} \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \gamma_6 \end{bmatrix} \tag{2.3.13}$$

where

$$\begin{aligned} Q_{11} &= mE_1 \\ Q_{22} &= mE_2 \\ Q_{12} &= m\nu_{21}E_1 = Q_{21} = m\nu_{12}E_2 \\ Q_{66} &= G_{12} \end{aligned}$$

Because of the symmetry of both stiffness and compliance matrices we have,

$$\begin{aligned} S_{12} &= S_{21} \\ Q_{12} &= Q_{21} \\ \therefore\ \nu_{12}E_2 &= \nu_{21}E_1 \\ \implies \frac{\nu_{12}}{\nu_{21}} &= \frac{E_1}{E_2} \end{aligned}$$

Since ν_{12} and ν_{21} are related to each other through moduli E_1 and E_2, the material properties for an unidirectional ply can be defined using 4 interdependent engineering constants such as, E_1, E_2, G_{12}, and ν_{12} or ν_{21}.

2.3.2 Stress, Strain, and Material Property Transformation

Unlike isotropic materials, where material properties do not change with the reference coordinates, in composite materials properties do change with reference coordinates. Hence, transformation of stress, strain, and material properties for different orientations of fibers is really important. Moreover, often the lamina or local coordinates $(1-2)$ do not coincide with the loading or reference coordinates $(x-y)$ (see figure 2–10). In such case, the "on-axis" stress and strain (w.r.t. lamina coordinates $1-2$)

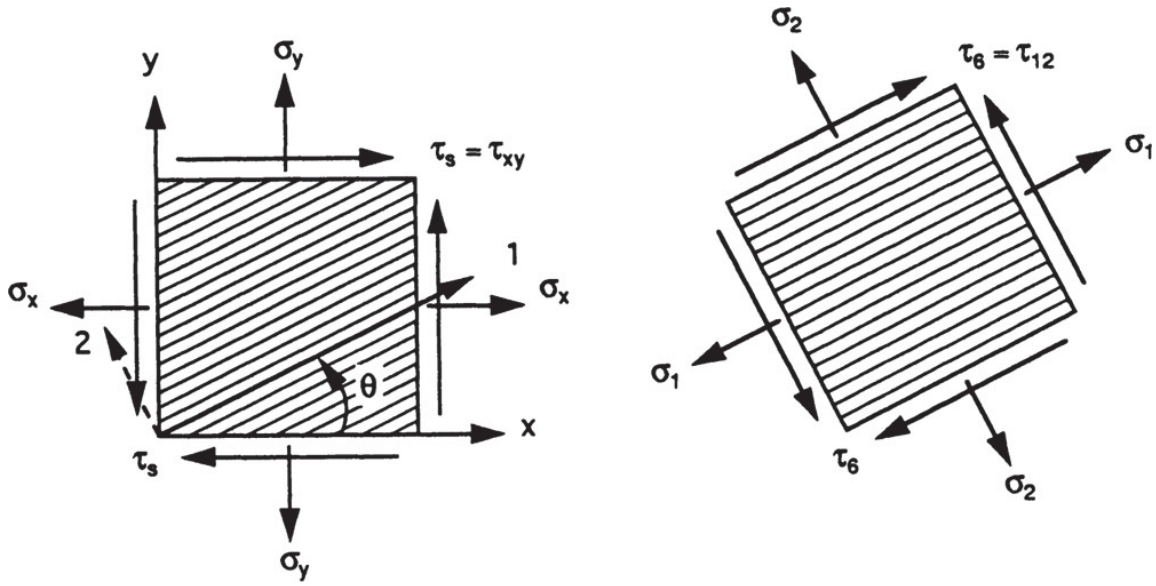


Figure 2–10: Stress components in an uni-directional lamina w.r.t. global and local coordinates (Source: [5])

can be expressed in terms of the off-axis quantities (w.r.t. reference coordinates $x-y$) as [10],

$$\begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \tau_6 \end{bmatrix} = \begin{bmatrix} T_\sigma \end{bmatrix} \begin{bmatrix} \sigma_x \\ \sigma_y \\ \tau_s \end{bmatrix} \qquad \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \gamma_6 \end{bmatrix} = \begin{bmatrix} T_\epsilon \end{bmatrix} \begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_s \end{bmatrix} \tag{2.3.14}$$

where

$$\begin{bmatrix} T_\sigma \end{bmatrix} = \begin{bmatrix} m^2 & n^2 & 2mn \\ n^2 & m^2 & -2mn \\ -mn & mn & m^2-n^2 \end{bmatrix} \; and \; \begin{bmatrix} T_\epsilon \end{bmatrix} = \begin{bmatrix} m^2 & n^2 & mn \\ n^2 & m^2 & -mn \\ -2mn & 2mn & m^2-n^2 \end{bmatrix}$$

$$m = \cos\theta \quad \text{and} \quad n = \sin\theta$$

So, by using equation (2.3.14) it is possible to transform stresses and strains from one coordinate system to another and equations (2.3.11) and (2.3.13) relate stress and strain in on-axis coordinate system. Using these relations, it is possible to relate stress to strain in the off-axis coordinate system as,

$$\begin{bmatrix} \sigma_x \\ \sigma_y \\ \tau_s \end{bmatrix} = \begin{bmatrix} Q_{xx} & Q_{xy} & Q_{xs} \\ Q_{yx} & Q_{yy} & Q_{ys} \\ Q_{sx} & Q_{sy} & Q_{ss} \end{bmatrix} \begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_s \end{bmatrix} \tag{2.3.15}$$

where $\begin{bmatrix} Q_{xx} & Q_{xy} & Q_{xs} \\ Q_{yx} & Q_{yy} & Q_{ys} \\ Q_{sx} & Q_{sy} & Q_{ss} \end{bmatrix} = \overline{[Q]}_{ij}$ is a transformed off-axis stiffness matrix and can be derived as follows,

$$\begin{aligned} \begin{bmatrix} \sigma_x \\ \sigma_y \\ \tau_s \end{bmatrix} &= \begin{bmatrix} T_\sigma^{-1} \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \tau_6 \end{bmatrix} = \begin{bmatrix} T_\sigma^{-1} \end{bmatrix} \begin{bmatrix} Q_{11} & Q_{12} & 0 \\ Q_{21} & Q_{22} & 0 \\ 0 & 0 & Q_{66} \end{bmatrix} \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \gamma_6 \end{bmatrix} \\ &= \begin{bmatrix} T_\sigma^{-1} \end{bmatrix} \begin{bmatrix} Q_{11} & Q_{12} & 0 \\ Q_{21} & Q_{22} & 0 \\ 0 & 0 & Q_{66} \end{bmatrix} \begin{bmatrix} T_\epsilon \end{bmatrix} \begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_s \end{bmatrix} \end{aligned} \tag{2.3.16}$$

Comparing equations (2.3.15) and (2.3.16),

$$\overline{[Q]}_{ij} = \begin{bmatrix} Q_{xx} & Q_{xy} & Q_{xs} \\ Q_{yx} & Q_{yy} & Q_{ys} \\ Q_{sx} & Q_{sy} & Q_{ss} \end{bmatrix} = \begin{bmatrix} T_\sigma^{-1} \end{bmatrix} \begin{bmatrix} Q_{11} & Q_{12} & 0 \\ Q_{21} & Q_{22} & 0 \\ 0 & 0 & Q_{66} \end{bmatrix} \begin{bmatrix} T_\epsilon \end{bmatrix} \tag{2.3.17}$$

Similarly, the transformed compliance matrix $\overline{[S]}_{ij}$ can be written as,

$$\overline{[S]}_{ij} = \begin{bmatrix} S_{xx} & S_{xy} & S_{xs} \\ S_{yx} & S_{yy} & S_{ys} \\ S_{sx} & S_{sy} & S_{ss} \end{bmatrix} = \begin{bmatrix} T_\epsilon^{-1} \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} & 0 \\ S_{21} & S_{22} & 0 \\ 0 & 0 & S_{66} \end{bmatrix} \begin{bmatrix} T_\sigma \end{bmatrix} \tag{2.3.18}$$

Among the above two transformed properties, the stiffness matrix $\overline{[Q]}_{ij}$ can be used to determine the behavior of a multidirectional laminate made up of of plies or ply groups with arbitrary fiber orientations.

2.3.3 Elastic Properties of General Laminates

The classical laminate theory (CLT) predicts the elastic behavior of the laminate under the following assumptions [5],

1. Each layer (lamina) of the laminate is quasi-homogeneous and orthotropic.
2. The laminate is thin with its lateral dimensions much larger than its thickness and is loaded in its plane only, i.e., the laminate and its layers (except for their edges) are in a state of plane stress, i.e. ($\sigma_z = \tau_{xz} = \tau_{yz} = 0$).
3. All displacements are small compared with the thickness of the laminate, i.e. ($|u|, |v|, |w| \ll h$)
4. Displacements are continuous throughout the laminate.
5. In-plane displacements vary linearly through the thickness of the laminate, i.e., u and v displacements in the x and y directions are linear functions of z.
6. Transverse shear strains γ_{xz} and γ_{yz} are negligible. This assumption and the preceding one imply that straight lines normal to the middle surface remain straight and normal to that surface after deformation.
7. Strain-displacement and stress-strain relations are linear.
8. Normal distances from the middle surface remain constant, i.e., the transverse normal strain ϵ_z is negligible (compared with the in-plane strains ϵ_x and ϵ_y).

Laminate notations with the individual lamina z-coordinates marked as h_i from the reference plane are shown in figure 2–11(a).

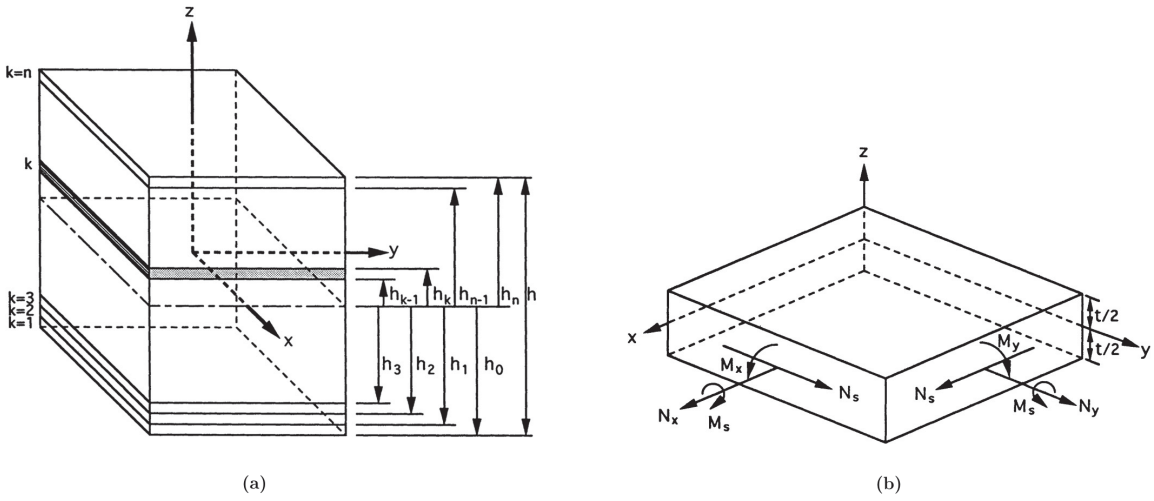


Figure 2–11: (a) Multidirectional laminate with individual ply coordinate notation, (b) Force and moment resultants acting on an element in a ply. (Source: [5])

The total strain at any point in a laminate can be written in terms of in-plane strains ϵ_i^o and laminate curvatures k_i as,

$$\begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_s \end{bmatrix}_z = \begin{bmatrix} \epsilon_x^o \\ \epsilon_y^o \\ \gamma_s^o \end{bmatrix} + z \begin{bmatrix} k_x \\ k_y \\ k_s \end{bmatrix} \tag{2.3.19}$$

where, the superscript "o" represents strain due to in-plane deformation and is independent of z. This is because of the thin laminate assumption. For small displacements, in plane strains and curvatures are given as [7, 5, 10],

$$\epsilon_x^o = \frac{\partial u_o}{\partial x}\ , \quad \epsilon_y^o = \frac{\partial v_o}{\partial y}\ , \quad \gamma_s^o = \frac{\partial u_o}{\partial y} + \frac{\partial v_o}{\partial x}$$

$$k_x = -\frac{\partial^2 w}{\partial x^2}\ , \quad k_y = -\frac{\partial^2 w}{\partial y^2}\ , \quad k_y = -\frac{2\partial^2 w}{\partial x \partial y}$$

The stress-strain relation for the k^{th} layer at distance z_k from the reference plane (see figure 2–11) w.r.t. laminate coordinate axes is given by equation (2.3.15). Substituting equation (2.3.19) in equation (2.3.15) we get,

$$\begin{bmatrix}\sigma_x\\ \sigma_y\\ \tau_s\end{bmatrix}_k = \begin{bmatrix}Q_{xx} & Q_{xy} & Q_{xs}\\ Q_{yx} & Q_{yy} & Q_{ys}\\ Q_{sx} & Q_{sy} & Q_{ss}\end{bmatrix}_k \begin{bmatrix}\epsilon_x^o\\ \epsilon_y^o\\ \gamma_s^o\end{bmatrix} + z_k \begin{bmatrix}Q_{xx} & Q_{xy} & Q_{xs}\\ Q_{yx} & Q_{yy} & Q_{ys}\\ Q_{sx} & Q_{sy} & Q_{ss}\end{bmatrix}_k \begin{bmatrix}k_x\\ k_y\\ k_s\end{bmatrix} \tag{2.3.20}$$

The resultant forces (N_i) and moments (M_i) acting on the an element in the k^{th} ply, as shown in figure 2–11(b), can be related to respective stress components as,

$$\begin{aligned}\begin{bmatrix}N_x\\ N_y\\ N_s\end{bmatrix}_k &= \int_{-t/2}^{t/2} \begin{bmatrix}\sigma_x\\ \sigma_y\\ \tau_s\end{bmatrix}_k dz\\ \begin{bmatrix}M_x\\ M_y\\ M_s\end{bmatrix}_k &= \int_{-t/2}^{t/2} \begin{bmatrix}\sigma_x\\ \sigma_y\\ \tau_s\end{bmatrix}_k z dz\end{aligned} \tag{2.3.21}$$

For a multi-layer laminate, the total force and moment resultants can be obtained by summing up resultants for all the layers and is given by,

$$\begin{aligned}\begin{bmatrix}N_x\\ N_y\\ N_s\end{bmatrix} &= \sum_{k=1}^{n} \int_{h_{k-1}}^{h_k} \begin{bmatrix}\sigma_x\\ \sigma_y\\ \tau_s\end{bmatrix}_k dz\\ \begin{bmatrix}M_x\\ M_y\\ M_s\end{bmatrix} &= \sum_{k=1}^{n} \int_{h_{k-1}}^{h_k} \begin{bmatrix}\sigma_x\\ \sigma_y\\ \tau_s\end{bmatrix}_k z dz\end{aligned} \tag{2.3.22}$$

where n is the total number of plies in the laminate and, h_k and h_{k-1} are z-coordinates of the upper and lower surfaces of the k^{th} ply, respectively.

Putting equation (2.3.20) in equation (2.3.22) and using indicial notation, we get,

$$\begin{aligned}\Big[N\Big]_{x,y} &= \left[\sum_{k=1}^{n} \overline{[Q]}^k_{x,y} \int_{h_{k-1}}^{h_k} dz\right] [\epsilon^o]_{x,y} \\ &+ \left[\sum_{k=1}^{n} \overline{[Q]}^k_{x,y} \int_{h_{k-1}}^{h_k} z dz\right] [k]_{x,y} \\ &= \left[\sum_{k=1}^{n} \overline{[Q]}^k_{x,y} (h_k - h_{k-1})\right] [\epsilon^o]_{x,y} \\ &+ \left[\tfrac{1}{2}\sum_{k=1}^{n} \overline{[Q]}^k_{x,y} (h_k^2 - h_{k-1}^2)\right] [k]_{x,y} \\ &= [\boldsymbol{A}]_{x,y}[\epsilon^o]_{x,y} + [\boldsymbol{B}]_{x,y}[k]_{x,y}\end{aligned} \tag{2.3.23}$$

and,

$$\begin{aligned}\Big[M\Big]_{x,y} &= \left[\tfrac{1}{2}\sum_{k=1}^{n} \overline{[Q]}^k_{x,y} (h_k^2 - h_{k-1}^2)\right] [\epsilon^o]_{x,y} \\ &+ \left[\tfrac{1}{3}\sum_{k=1}^{n} \overline{[Q]}^k_{x,y} (h_k^3 - h_{k-1}^3)\right] [k]_{x,y} \\ &= [\boldsymbol{B}]_{x,y}[\epsilon^o]_{x,y} + [\boldsymbol{D}]_{x,y}[k]_{x,y}\end{aligned} \tag{2.3.24}$$

where $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{D}$ are extensional stiffness or in-plane stiffness matrix, coupling stiffness matrix (relates in-plane loads to curvature and moments to in-plane strains), and bending/flexural stiffness matrix, respectively, and can be expressed as,

$$\begin{aligned}A_{ij} &= \sum_{k=1}^{n} Q_{ij}^k (h_k - h_{k-1}) \\ B_{ij} &= \frac{1}{2}\sum_{k=1}^{n} Q_{ij}^k (h_k^2 - h_{k-1}^2) \\ D_{ij} &= \frac{1}{3}\sum_{k=1}^{n} Q_{ij}^k (h_k^3 - h_{k-1}^3)\end{aligned} \tag{2.3.25}$$

with $i, j = x, y, s$.

Finally, equations (2.3.23) and (2.3.24) can be combined and expressed in a matrix form as,

$$\left[\begin{array}{c} N_x \\ N_y \\ N_s \\ \hline M_x \\ M_y \\ M_s \end{array}\right] = \left[\begin{array}{ccc|ccc} A_{xx} & A_{xy} & A_{xs} & B_{xx} & B_{xy} & B_{xs} \\ A_{yx} & A_{yy} & A_{ys} & B_{yx} & B_{yy} & B_{ys} \\ A_{sx} & A_{sy} & A_{ss} & B_{sx} & B_{sy} & B_{ss} \\ \hline B_{xx} & B_{xy} & B_{xs} & D_{xx} & D_{xy} & D_{xs} \\ B_{yx} & B_{yy} & B_{ys} & D_{yx} & D_{yy} & D_{ys} \\ B_{sx} & B_{sy} & B_{ss} & D_{sx} & D_{sy} & D_{ss} \end{array}\right] \left[\begin{array}{c} \epsilon_x^o \\ \epsilon_y^o \\ \gamma_s^o \\ \hline k_x \\ k_y \\ k_s \end{array}\right] \tag{2.3.26}$$

In this work, we solved two composite stacking sequence optimization problems. In CASE-I we solve the target stiffness achievement of a composite box-beam, and in CASE-II we optimize the stacking sequence of a rectangular sandwich panel with a goal to maximize the safety factor (More details about the problem formulations is given in chapter 4). In CASE-II, we can directly use load-strain relations given by equation 2.3.26 to calculate the safety factor for given loading conditions, N_i and M_i. For CASE-I, we use the mathematical model formulated by Simth and Chopra [11] to calculate the stiffness of a composite box-beam. A brief description about this model is presented in chapter 4.

CHAPTER 3
Introduction To Optimization of Composite Stacking Sequence

Design is a complex process, in which possibile alternatives are checked to choose the best suitable one, essentially making it a *decision-making* process [12]. The ultimate goal in the design process is either to minimize the efforts/investment or to maximize the desired benefit. These objectives in any practical situation can be expressed by a mathematical model, which is nothing but an abstract description of the system using functions of certain variables, relevant natural laws, geometry, etc. By having a criterion to compare these models for different choices, the best or *optimum* solution can be identified using various optimization techniques [12, 13]. Some of these techniques are addressed in this chapter in point of view of composite stacking sequence design optimization.

3.1 Introduction to Design Optimization

What is design optimization? While there exists numerous definitions for "design optimization" in the literature, Papalambros and Wilde in [12] have proposed that, design optimization is a process which involves:

1. The selection of a set of variables to describe the design alternatives.

2. The selection of an objective (criterion), expressed in terms of the design variables, which we seek to minimize or maximize.

3. The determination of a set of constraints, expressed in terms of the design variables, which must be satisfied by any acceptable design.

4. The determination of a set of values for the design variables, which minimize (or maximize) the objective, while satisfying all the constraints.

Let $\mathbf{x} = [x_1, x_2, x_3, \cdots, x_n]$ represents the set of design variables belonging to the subset $\mathcal{X}$ of n-dimensional real space $\mathbb{R}^n$, i.e., $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$. The objective and constraints can be quantifiably expressed as functions of design variables as, $f(\mathbf{x})$, $h_i(\mathbf{x}) = 0$, and $g_i(\mathbf{x}) \leq 0$, where, f, h_i, and g_i are objective, equality constraints, and inequality constraints, respectively. The optimization problem can then be written in a general form as,

$$\min \ f(\mathbf{x}) \qquad \text{subject to} \begin{cases} h_i(\mathbf{x}) = 0, \ \cdots\cdots \ i = 1, 2, \cdots, m_1 \\ g_i(\mathbf{x}) \leq 0, \ \cdots\cdots \ i = 1, 2, \cdots, m_2 \end{cases} \tag{3.1.1}$$

Various techniques present today to solve the problem expressed in the above expression, which can broadly be classified as,

1. Gradient-based Methods

2. Derivative-free Optimization

As the name says, Gradient-based Methods (GBM) use gradients of functions, to move towards the optimum/better solution from the initial point in a design space. Derivative-free methods, on the other hand, do not need gradients of the functions.

Instead, they only use function values to find the optimum/better solution to the problem. Both these techniques have their own advantages and limitations, perhaps, these advantages and limitations are often specific to the type and structure of the problem one wishes to solve.

The application of these techniques to solve composite design optimization problems is reviewed in the following sections.

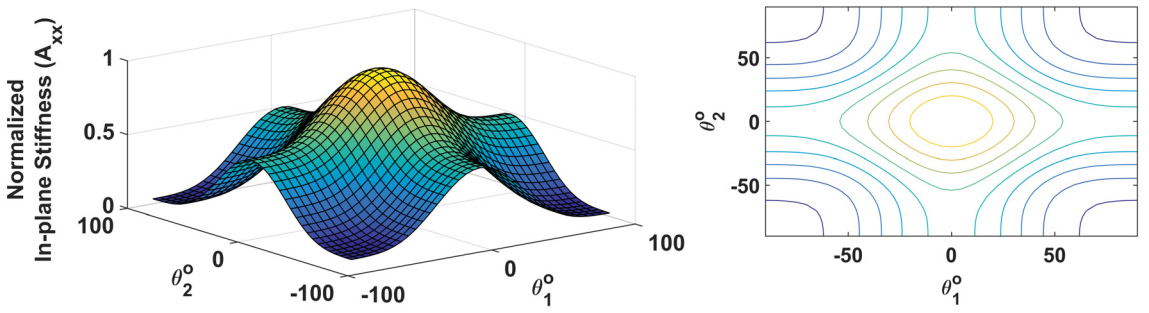
3.2 Gradient-based Methods

As discussed before, gradient-based methods use gradients of the function to move towards a better solution from the initial point in a design space. In numerical optimization it is believed that, gradients of the function posses the most useful information. For continuously differentiable functions, the first-order necessary condition and the second order sufficient condition can be used to find local optima. These necessary and sufficient conditions use the gradients of the function.

Gradient-based methods have been used in composite stacking sequence optimization in the past. Ghiasi et al. in [14, 15] have compared optimization techniques used to optimize the stacking sequence of laminated composites and highlighted potential areas which need more attention. In [14], the authors have mainly covered the constant stiffness design class, where the stacking sequence is uniform throughout the laminate. They compared some of the popular gradient-based methods, such as vanishing the function's first order gradient, the Steepest Decent method, the Conjugate Gradient method, the Quasi-Newton method, etc. All these methods are well known for their fast and reliable convergence and can find local optima

(or global optima in case of an unimodal function) in comparitively less number of iterations and function evaluations [8, 12, 14]. However, these methods have their disadvantages when applied to composite stacking sequence optimization. To illustrate this let us consider a simple unconstrained problem with the objective of maximizing the in-plane stiffness of a four-layer symmetric laminate with the stacking sequence $\Theta = [\theta_1/\theta_2]_s$. The symmetry of the laminate about the mid-plane reduces the number of variables from 4 to 2.

As seen in figure 3–1, the normalized in-plane stiffness, A_{xx}, for a four-layer symmetric Graphite/Epoxy laminate[1] is a multimodal function with multiple regions where the first order partial derivative of the function reduces to zero. We use the



(a) Surface Plot

(b) Contour Plot

Figure 3–1: In-plane stiffness A_{xx} as a function of ply angles θ_1 and θ_2 for a 4 layer symmetric Graphite/Epoxy laminate

sequential quadratic programming (SQP) algorithm in `MATLAB` optimization toolbox,

[1] For Graphite/Epoxy properties: $E_1 = 141.5GPa$, $E_2 = 9.8GPa$, $G_{12} = 5.9GPa$, and $\nu_{12} = 0.42$

to find optimum fiber orientations to maximize the in-plane stiffness A_{xx}. The results with different starting points are shown in the table 3–1. It is evident from figure 3–1a that, $\Theta^* = [0°/0°]_s$ is the global optimum for this problem. It can be

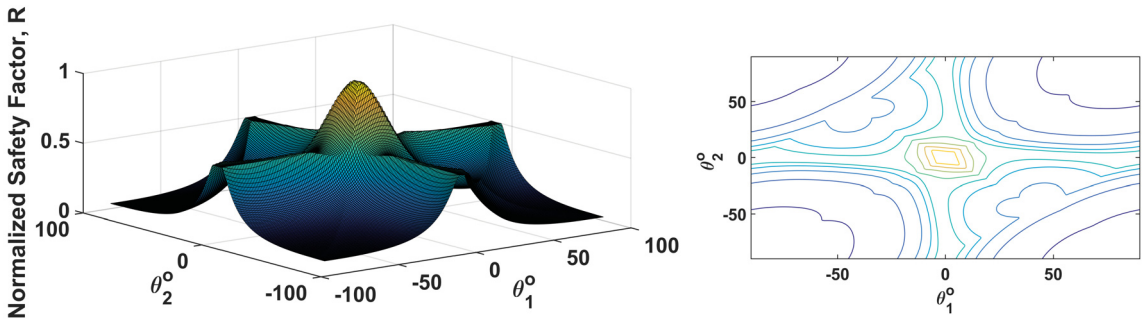
Table 3–1: Example: Maximizing the stiffness for a 4-layer symmetric laminate using GBM

Starting Point,Θ	**Function Evaluations**	**Solution, Θ^***	**Obj. Function, $A_{xx}(GPa.m)$**	**Optimization Status**
$[0_2]_s$	3	$[0_2]_s$	0.0728	Complete:Initial point is local optimum.
$[0/90]_s$	3	$[0/90]_s$	0.0389	Complete:Initial point is local optimum.
$[90_2]_s$	3	$[90_2]_s$	0.0050	Complete:Initial point is local optimum.
$[\pm 45]_s$	39	$[0_2]_s$	0.0728	Complete:Local optimum found.
$[89/-1]_s$	112	$[0_2]_s$	0.0728	Complete:Local optimum found.

seen from table 3–1, a fairly simple, unconstrained problem with a smooth objective function can be efficiently solved using GBMs. Here SQP, which essentially reduces to a Newton's method for unconstrained optimization problem, converges to a global optimum ($[0_2]_s$), even for the worst initial guess ($[89/-1]_s$), in merely 112 function evaluations.

Now, let us consider a similar problem, but with an objective to maximize the safety factor under the application of bending about the y-direction. Here, the minimum of

the safety factors calculated using three failure criteria such as, Maximum Stress criterion, Tsai-Wu Quadratic failure criterion, and the Hashin Failure criterion, is taken as the objective to maximize. The normalized safety factor as a function of design variables, $\Theta = [\theta_1/\theta_2]_s$, under the bending load of $1,000N$ (note that, this is bending moment per unit width in Nm/m) is shown in figure 3–2 for Graphite/Epoxy.



(a) Surface Plot

(b) Contour Plot

Figure 3–2: Bending safety factor as a function of ply angles θ_1 and θ_2 for a 4 layer symmetric Graphite/Epoxy laminate

The results for this optimization problem are presented in table 3–2. It can be seen from figure 3–2 that, the design space consists of multiple regions where the first order gradient of the function reduces to zero satisfying the first-order optimality condition. Hence, a gradient based method is most likely to get trapped in such local optima. Such a situation is evident from table 3–2, where starting points $[0/90]_s$ and $[\pm 45_s]$ lead to a local optimum, whereas a more educated guess such as, $[30/-30]_s$ leads to a near global optimum solution. Hence, perhaps, in such situations, a priori knowledge of the design space of the problem one wishes to solve is critical to get a better quality solution.

Table 3–2: Example: Maximizing the safety factor for a 4-layer symmetric laminate using GBM

Starting Point,Θ	**Function Evaluations**	**Solution, Θ^***	**Obj. Function, $A_{xx}(GPa.m)$**	**Optimization Status**
$[0_2]_s$	3	$[0_2]_s$	2.7565	Complete:Initial point is local minima.
$[0/90]_s$	3	$[0/90]_s$	1.4980	Complete:Initial point is local minima.
$[90_2]_s$	9	$[90_2]_s$	0.2002	Complete:Initial point is local minima.
$[\pm 45]_s$	410	$[72.83/-3.55]_s$	1.4830	Stopped:Local minima possible.
$[30/-30]_s$	428	$[2.26/-0.40]_s$	2.7522	Stopped:Local minima possible.

Besides, GBMs can only handle continuous variable, and ply orientations are generally limited to only a set of discrete angles such as, $0°, \pm 45°, 90°$. Considering these limitations of gradient-based methods, it is necessary to try derivative-free optimization methods, to be able to address more complex and realistic problems.

3.3 Derivative-free Methods

The inherent limitations present in gradient-based methods were, perhaps, one of the greatest motivations behind the development of derivative-free optimization methods. Moreover, in engineering design optimization, most of the time the objective and constraint functions are calculated using black-box simulations and cannot be explicitely expressed in closed form mathematical expressions. Hence, the derivatives of these functions are either unavailable or unreliable or very expensive to calculate.

In such case, one might still want to carry out optimization, and derivative-free optimization (DFO) methods provide the means to solve such problems. In fact, DFO is considered to be one of the most important, open, and challenging areas in computational science and engineering [16]. As mentioned earlier, derivative-free optimization methods only use function values to find an optimum/better solution. They provide a way to address problems with inherent system noise, highly non-linear functions, and with a high number of variables, or otherwise problems which are unseemly for classical gradient-based methods [17]. We shall now look into the DFO methods used in stacking sequence optimization of laminated composites.

Ghiasi et al. have reviewed some of the popular derivative-free optimization methods such as, Random Search methods, Simplex methods, Simulated Annealing (SA), and Genetic Algorithms (GAs) used in stacking sequence optimization of composites [14, 15]. Here the authors have concluded that, GAs have been the most popular in stacking sequence optimization of composite laminates, SA being the second most popular method. These methods have been reveled to be quite significant in composite lay-up design, as described in the following sections.

3.3.1 Genetic Algorithms

Genetic Algorithm belongs to the class of evolutionary algorithms. It mimics the Darwin's theory of survival of the fittest. In GA, a set of designs, usually refered as a population of individuals is randomely generated, and is evolved through iterations (usually referred as generations). In each generation, the population is transformed using genetic operators such as, selection, cross-over, mutation, and reproduction

[8]. Individuals with better function values are retained and passed on to the next generation. The individual with best function value is returned as the solution when the termination criterion are satisfied. A high-level representation of GA is shown in figure 3–3.

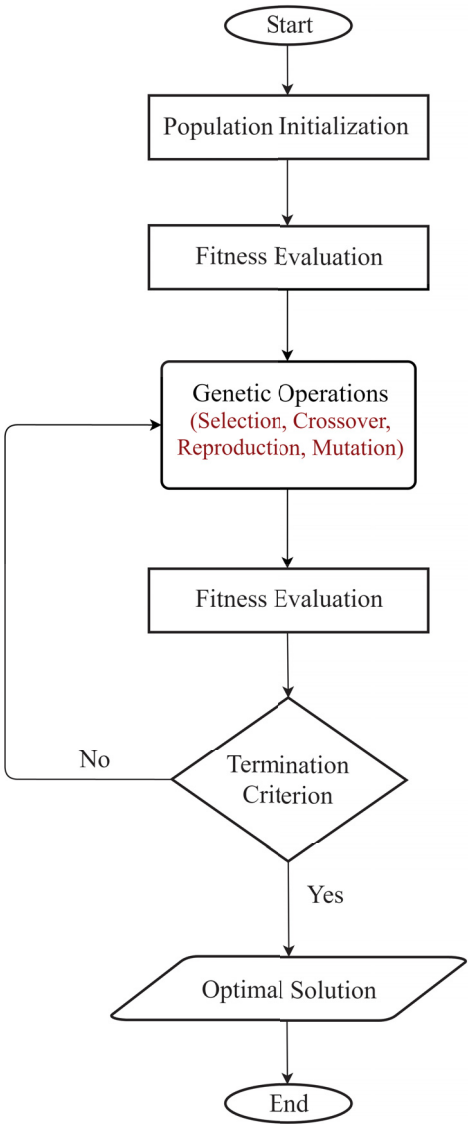


Figure 3–3: High-level representation of Genetic Algorithm

GAs have been used to find the optimal stacking sequence in variety of composite structures such as, simple rectangular plates, sandwich panels, I-section beam, box-beam, etc. Naik et al. in [18] used a simple binary coded GA to optimize the simple rectangular laminated composite plate to minimize the weight of the plate, with only ply orientations as variables. Here, the authors have considered strength constraints under the application of tensile, compressive, and shear loads, either alone or in combination. Almeida and Awruch in [19] used finite-element model, instead of an mathematical model. They formulated the problem with ply orientation and ply material as design variables while minimizing the weight and deflection of a rectangular plate and conclude that, performance and reliability of the GA can be improved by careful selection of parameters for genetic operators, and using memory technique to store previous results to avoid repeated fitness evaluations. However, this makes the algorithm complex to program and is not very effective if one wants to make use of GA's one of the advantages, that is its ease of programming. Similarly, different variants of GA have been used in the past for different objective functions [14, 15, 20, 21, 22, 23, 24]. Although, modifications in simple GA using different techniques, such as memory approach [19], parallel computing [25, 26], multi-level problem formulation [27], using problem specific genetic operators [21, 28], using approximation models [22, 23], etc. seem to improve the algorithm performance and robustness, it is almost impossible to comprehensively compare them, because GAs are probabilistic methods [14].

Although GAs has been reported to be the most popular technique for stacking sequence optimization of laminated composites, there are some limitations associated

with them. For instance, they require a high number of function evaluation before finding a better solution. The performance of GAs largely depends on population size and parameters which control genetic operators. Moreover, GAs generate a set of solutions according to a randomly defined termination criterion, and the solutions generated do not posses convergence properties. The improvement in quality of the solutions generated by GAs is usually at an expense of computational cost. Hence, this offers an opportunity to test other derivative-free optimization techniques to overcome shortcomings associated with GAs.

3.3.2 Simulated Annealing

Simulated annealing is an iterative procedure, which mimics the annealing process in metallurgy. The algorithm works as follows. At any instance, the algorithm randomly generates a design and tests it with the current design. The probability that the worst design gets selected for the following iteration is controlled by the controlling parameter, often referred as temperature in this case. At the beginning, the temperature is assigned with a high value, which implies a higher probability that the worst design will be passed down to the next iteration. This ensures extensive search of the design space which is often the objective in most of the global search algorithms. The temperature is gradually decreased as the algorithm moves forward with the iterations, similar to the slow and controlled cooling in annealing process [29]. In other words, at the beginning when the temperature is high, the search is almost random, and with the decrease in temperature, the search becomes almost greedy [30].

Although simulated annealing is extensively used in the past to solve combinatorial optimization problems, it is still a challenging task to define an ideal cooling schedule which can cover large range of problems [30]. Moreover, generating a sequence of points that converges to a non-optimal solution is another limitation of SA [14]. Therefore, several modifications have been proposed in the past to overcome this limitation and to improve the performance and robustness of the algorithm. Rao and Arvind in [30] used a tabu embedded simulated annealing (TSA) to solve a stacking sequence optimization of laminated composites formulated as a combinatorial problem. They used the tabu search in order to prevent recycling of recently visited points and to help algorithm to recover from the sub-optimal region. TSA was reported to be at par as GAs in obtaining multiple near optimal solutions, and the computational performance was reported to be superior for the problem type considered. Akbulut and Sonmez used a new variant of SA called as Direct Search Simulated Annealing (DSA) to optimize the rectangular laminated plate subjected to in-plane loads in [31] and combination of in-plan and flexural loads in [32]. The problem is formulated to minimize the thickness of the plate with fiber orientation angles and layer thickness as design variables. Unlike standard SA, where only one point moves in the design space, DSA uses a set of points in each move (iteration) and it always retains the best solutions making it more robust than the standard SA. The authors used 1° increment in fiber orientation to obtain a larger design space to check robustness of the proposed algorithm, and concluded that the DSA provided consistent and reliable results when compared to other variants of SA and GA [31, 32].

Although, SA and its variants have been extensively used in composite stacking sequence optimization in the past, there are certain limitations associated with it. For instance, there is no known cooling schedule that can be applied to a large set of problems and most of the time, user has to define the cooling schedule and the parameters associated with it using empirical evidences obtained from experimental runs of the algorithm [30]. Moreover, SA is not as easy as GAs to program, and unlike GAs, SA cannot be programmed to take advantages from the specific properties of a given problem [14].

3.3.3 Other Meta-heuristic Techniques

Apart from GAs and SA, other nature-inspired derivative-free optimization methods have also been used in the recent past. Kathiravan and Ganguli in [33] have used the particle swarm optimization (PSO), an evolutionary algorithm which mimics the behavior of a bird flock or a school of fish in search of the food, for maximizing the strength of a helicopter rotor box-beam. The performance of PSO is compared with sequential quadratic programming (SQP) and authors have reported that, PSO yielded results which are either superior or equivalent to that of SQP. S.N. Omkar et al. developed a version of an evolutionary algorithm from the artificial immune system (AIS) paradigm, called Objective Switching Clonal Selection Algorithm (OSCSA) for multi-objective optimization of composites [34]. Here, the authors compared the performance of the proposed method with their previous work (vector evaluated particle swarm optimization (VEPSO) [35]) to show the robustness of proposed method. Furthermore, S.N. Omkar et al. proposed a variant of Artificial Bee Colony (ABC) algorithm, another nature-inspired evolutionary algorithm based

on the behavior of social insects, called Vector Evaluated Artificial Bee Colony (VE-ABC) algorithm for multi-objective composite design optimization [36]. Here, the authors used separate evaluations and swarms for multiple objectives and concluded that the performance of this method is at par when compared to other evolutionary algorithms such as, GAs, AIS, and PSO.

All in all, although GAs and SA have been have been reported to be the most popular techniques in the past for optimization of composite stacking sequence, shortcomings associated with these methods encourage researchers to test other derivative-free optimization techniques.

3.3.4 Mesh Adaptive Direct Search

Direct search methods (DSM) use a systematic approach to find an optimum solution for a given problem. Enumeration search, Nelder and Mead (NM) Simplex method, random and greedy search, etc. have been used to optimize the stacking sequence of laminated composites [14]. Hakan Boyaci in his master's thesis used generalized pattern search (GPS) algorithm to optimize the stacking sequence of anti-buckled graphite/epoxy laminate for minimum weight [37]. Cardoso et al. have compared the genetic algorithm and the particle swarm algorithm with direct search algorithms [38]. Here, the authors have compared the performance of DSM with genetic algorithm when solving computationally expensive structural optimization problems and concluded that, DSM provided good quality solutions under low computational efforts. With the availability of sophisticated software packages implementing these methods, direct search methods can be promising in structural optimization..

Mesh adaptive direct search, which extends the GPS by allowing the local search in an asymptotically dense set of directions, has been proved to be effective and efficient when solving black-box optimization problems [1, 39, 40, 41]. MADS is developed to efficiently handle highly non-linear and discontinuous functions with numerical noise. Moreover, it generates solutions which posses convergence properties [2]. This encourages us to try and test MADS to solve composite stacking sequence optimization problems. We find it is worth the efforts to try and compare the performance of Mesh Adaptive Direct Search algorithm proposed by Audet and Dennis [1] to solve the composite lay-up design optimization problem. A detail description of MADS is presented in the next chapter.

CHAPTER 4

Optimization Using Mesh Adaptive Direct Search and GAs

As discussed in the previous chapter, various composite lay-up design problem formulations have been reported in the literature. Different objectives such as, strength maximization, weight minimization, buckling factor maximization, etc. have been used in previous studies. In this study, we have considered two composite stacking sequence optimization problems. In CASE-I we solve the target stiffness achievement of a composite box-beam, and in CASE-II we optimize the stacking sequence of a rectangular sandwich panel with a goal to maximize the safety factor. We solve the CASE-I using two variants of the GA, namely a nested GA approach and an all-in-one GA approach, and compare their performance with the all-in-one approach with mesh adaptive direct search (MADS) algorithm. Similarly, we solve the CASE-II using multi-objective GA (MOGA) and bi-objective MADS (bi-MADS) and compare the performance of these algorithms. A detail information about problem formulations and the algorithms is presented in following section.

4.1 Problem Formulation

4.1.1 CASE-I: Helicopter rotor box-beam target stiffness achievement

Murugan et.al in [27] solved the composite box-beam optimization problem to design the beam to satisfy the stiffness requirements. Here, the authors have considered stiffness values obtained from an aeroelastic optimization study by Murgan and Ganguli [42]. The elastic behavior of the box beam as shown in figure 4–1 is given by,

$$\begin{bmatrix} Q_x \\ M_x \\ -M_y \\ M_z \end{bmatrix}_k = \begin{bmatrix} K_{11} & K_{12} & K_{13} & K_{14} \\ K_{12} & K_{22} & K_{23} & K_{24} \\ K_{13} & K_{32} & K_{33} & K_{34} \\ K_{14} & K_{42} & K_{43} & K_{44} \end{bmatrix}_k \begin{bmatrix} u_e^{'} \\ \phi^{'} \\ \omega^{"} \\ \nu^{"} \end{bmatrix} \tag{4.1.1}$$

where $K = [K_{ij}]$ is a box-beam stiffness matrix. Please note that, subscripts $i, j = [1, 2, 3, 4]$ are different than the subscript notation used in chapter 2. Here, subscripts represent the position of the stiffness component K_{ij} in the 4×4 box-beam stiffness matrix K. The stiffness matrix is calculated using the mathematical model formulation for composite box-beams by Smith and Chopra [11]. A brief description about the model is reported in subsection 4.1.1.

The off-diagonal terms in equation (4.1.1) are called elastic couplings. It is possible eliminate the elastic couplings by making the walls of the box-beam made up of symmetric and balanced lay-up and all four walls with the same stacking sequence. Under this condition, Murugan et al. in [27] formulated the optimization problem with the objective to achieve the desired bending stiffness about y and z direction

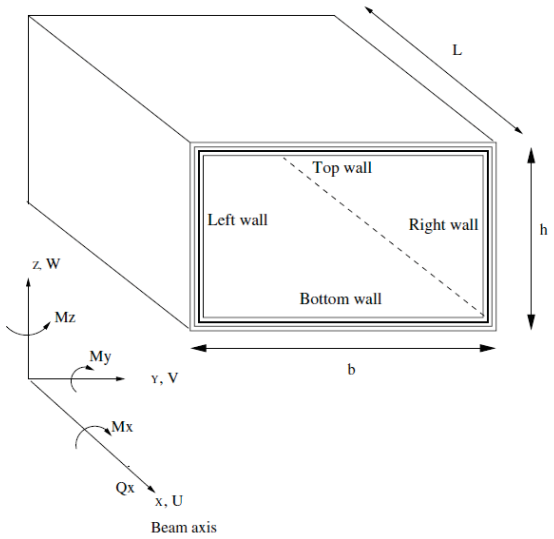


Figure 4–1: Composite box-beam configuration and the coordinate system. Source: [27]

and twisting stiffness about x direction. This objective translates into minimizing the percentage error between the target stiffness and the actual stiffness. The optimization problem can be written as,

$$\begin{aligned} \min \quad & \mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x})] \\ \textit{subject to} \quad & \mathbf{x}^{\mathbf{l}} \leq \mathbf{x} \leq \mathbf{x}^{\mathbf{u}} \\ & \mathbf{x} = [b, h, \Theta], \end{aligned} \tag{4.1.2}$$

where b and h represent the geometry of the box as shown in figure 4–1, $\Theta = [\theta_1, \theta_2, \theta_3, \cdots]$ represents the stacking sequence of the box-beam walls, and

$$f_1(\mathbf{x}) = \sqrt{\frac{(K_{22} - K_{22}^T)^2}{(K_{22}^T)^2}} \times 100$$

$$f_2(\mathbf{x}) = \sqrt{\frac{(K_{33} - K_{33}^T)^2}{(K_{33}^T)^2}} \times 100$$

$$f_3(\mathbf{x}) \;=\; \sqrt{\frac{(K_{44} - K_{44}^T)^2}{(K_{44}^T)^2}} \times 100$$

Here, K_{ii} represent the actual stiffness values of the box-beam given in equation (4.1.1) and the superscript 'T' in K_{ii}^T represents the corresponding target stiffness value. The target stiffness values obtained from the aeroelastic optimization [27, 42] are given in table 4–1.

Table 4–1: Target stiffness values

	Stiffness
K_{22} $(GJ,\ Nm^2)$	20,419.79
K_{33} $(EI_y,\ Nm^2)$	38,364.46
K_{44} $(EI_z,\ Nm^2)$	82,916.73

For simplicity, the three objectives in equation (4.1.2) are converted into a single objective function using *min-max* methods as,

$$\min \qquad \mathbf{J} = \max[f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x})] \tag{4.1.3}$$

The authors in [27] partitioned the design variables separating the continuous type geometry variables, b and h from the discrete fiber orientation variables. This separation of variables results in a nested loop optimization problem. The inner loop finds the optimal stacking sequence, $\Theta^* = [\theta_1, \theta_2, \theta_3, \cdots]$ for each box geometry $[b, h]$ in the outer loop. The nested loop problem formulation can be written as;

Outer loop

$$\begin{aligned} \min \quad & \mathbf{F}(\mathbf{b}, \mathbf{h}, \mathbf{\Theta}^*) \\ \textit{subject to} \quad & b^l \leq b \leq b^u \\ & h^l \leq h \leq h^u \end{aligned}$$

Inner loop: for a given $[b, h]$

$$\begin{aligned} \min \quad & \mathbf{F}(\mathbf{\Theta}) \\ \textit{subject to} \quad & \Theta \in \{\textit{set of discrete ply angles}\} \end{aligned}$$

Hence by using the nested approach, it is possible to find the best stacking sequence for different geometries and simply select the one with minimum error. This ensures thorough exploration of the design space.

In this study, to avoid an impractical solution, upper and lower bounds for geometry variables are considered as; $b : 3 \leq b \leq 5$, and $h : 2 \leq h \leq 3$. These bounds have been chosen to have reasonable cross-section properties of the box-beam based on the aeroelastic optimization of helicoptor rotor composite box-beam by Murugan et al. [42]. A 10° discretization is considered for ply orientations, i.e. $\theta_i \in \{10^\circ, 20^\circ, 30^\circ, 40^\circ, 50^\circ, 60^\circ, 70^\circ, 80^\circ, 90^\circ\}$.

The mathematical model for composite box-beams by Smith and Chopra [11] is presented in the next subsection.

Mathematical Model for Composite Box-Beams

The main advantage of the mathematical model for a composite box-beam by Simth and Chopra [11] is that, in this model the authors have considered individual walls of the box-beam as laminated composite plates, and the stiffness components of the

box-beam, K_{ij}, can be directly related to the the stiffness matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{D}$ of the individual walls. This makes the model easy to program and computationally less expensive. In our problem, we calculate three stiffness components for the composite box-beam, such as the torsional stiffness (K_{22}), the span-wise bending stiffness (K_{33}), and the chord-wise bending stiffness (K_{44}). The expressions to calculate these three stiffness components corresponding to the box-beam and the coordinate system shown in figure 4–1 and figure 4–2 are;

$$\begin{aligned}
K_{22} &= (1+\beta)^2 \iint_h Q_{ss}\ z^2\ dA\ +\ (1-\beta)^2 \iint_v Q_{ss}\ y^2\ dA \\
&\quad + d_0 \left[(1-\beta) \iint_v Q_{ys}\ y\ dA\ -\ (1+\beta) \iint_h Q_{ys}\ z\ dA \right] \\
&\quad + d_1(1-\beta) \iint_v Q_{ys}\ y^2\ dA\ -\ d_2(1+\beta) \iint_h Q_{ys}\ z^2\ dA \qquad (4.1.4) \\
K_{33} &= \iint_{h,v} Q_{xx}\ z^2\ dA\ -\ c_2 \iint_{h,v} Q_{xy}\ z^2\ dA \\
K_{44} &= \iint_{h,v} Q_{xx}\ y^2\ dA\ -\ b_1 \iint_{h,v} Q_{xy}\ y^2\ dA
\end{aligned}$$

where Q_{ij} is calculated from equation 2.3.17 for individual plies in the walls. The subscript h and v on the double integral sign represent horizontal and vertical walls,

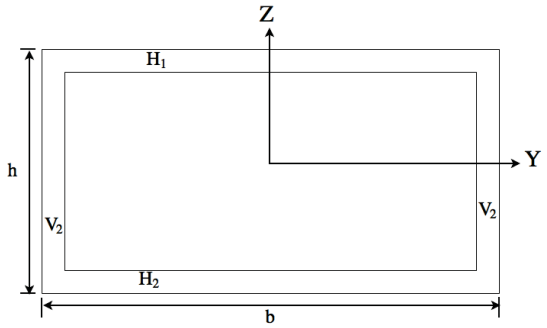


Figure 4–2: Composite box-beam cross-section and the coordinate axes

respectively. These stiffness expressions contain a number of constants such as β, b_1, c_2, d_0, d_1 and d_2. These constants are defined in [11] as below.

$$\begin{aligned}
\beta &= -\frac{(1-\alpha)}{(1+\alpha)} \\
\alpha &= \left(\frac{b}{h}\right) \\
b_1 &= \frac{\iint_{h,v} Q_{xy}\ y^2\ dA}{\iint_{h,v} Q_{yy}\ y^2\ dA} \\
c_2 &= \frac{\iint_{h,v} Q_{xy}\ z^2\ dA}{\iint_{h,v} Q_{yy}\ z^2\ dA} \\
d_0 &= \frac{\left[(1+\beta)\iint_h Q_{ys}\ z\ dA\ -\ (1-\beta)\iint_v Q_{ys}\ y\ dA\right]}{\iint_{h,v} Q_{yy}\ dA} \\
d_1 &= \frac{-(1-\beta)\iint_{h,v} Q_{ys}\ y^2\ dA}{\iint_{h,v} Q_{yy}\ y^2\ dA} \\
d_2 &= \frac{(1+\beta)\iint_{h,v} Q_{ys}\ z^2\ dA}{\iint_{h,v} Q_{yy}\ z^2\ dA}
\end{aligned}$$

These expressions can easily be expressed in terms of the individual wall stiffness components, A_{ij}, D_{ij}, and B_{ij} as shown below.

Let us consider the expression for K_{33} from equation 4.1.4,

$$K_{33} = \iint_{h,v} Q_{xx}\ z^2\ dA\ -\ c_2 \iint_{h,v} Q_{xy}\ z^2\ dA$$

The first term in the expression further expands to,

$$\iint_{h,v} Q_{xx}\ z^2\ dA\ =\ \iint_{H_1} Q_{xx}^{H_1}\ z^2\ dA\ +\ \iint_{H_2} Q_{xx}^{H_2}\ z^2\ dA + \iint_{V_1} Q_{xx}^{V_1}\ z^2\ dA\ +\ \iint_{V_2} Q_{xx}^{V_2}\ z^2\ dA$$

where H_1, H_2, V_1, and V_2 correspond to the horizontal and vertical walls as shown in figure 4–2. Let's simplify the first term in the above equation,

$$\begin{aligned}
\iint_{H_1} Q_{xx}^{H_1}\ z^2\ dA\ &=\ \iint_{H_1} Q_{xx}^{H_1}\ z^2\ dy\ dz \\
&= \int_{-b/2}^{b/2} \left[\int Q_{xx}^{H_1}\ z^2\ dz\right] dy \\
&= b\left[\sum_{i=1}^{m} \int_{\frac{h}{2}+z_i}^{\frac{h}{2}+z_{i+1}} (Q_{xx}^{H_1})^i\ z^2\ dz\right] \\
&= b\left[\sum_{i=1}^{m} (Q_{xx}^{H_1})^i \int_{\frac{h}{2}+z_i}^{\frac{h}{2}+z_{i+1}} z^2\ dz\right] \\
&= b\left[\sum_{i=1}^{m} (Q_{xx}^{H_1})^i\ \frac{z^3}{3}\Big|_{\frac{h}{2}+z_i}^{\frac{h}{2}+z_{i+1}}\right] \\
&= b\left[\sum_{i=1}^{m} (Q_{xx}^{H_1})^i\ \frac{1}{3}\left(\left[\frac{h}{2}+z_{i+1}\right]^3 -\ \left[\frac{h}{2}+z_i\right]^3\right)\right] \\
&= b\left[\sum_{i=1}^{m} (Q_{xx}^{H_1})^i \left(\left[\frac{z_{i+1}^3 - z_i^3}{3}\right] + h\left[\frac{z_{i+1}^2 - z_i^2}{2}\right] + \left(\frac{h}{2}\right)^2 (z_{i+1} - z_i)\right)\right]
\end{aligned}$$

from equation 2.3.25

$$\iint_{H_1} Q_{xx}^{H_1}\ z^2\ dA\ = b\left[D_{xx}^{H_1} + hB_{xx}^{H_1} + \left(\frac{h}{2}\right)^2 A_{xx}^{H_1}\right]$$

Similarly, it can be shown that,

$$\iint_{H_2} Q_{xx}^{H_2}\ z^2\ dA\ = b\left[D_{xx}^{H_2} - hB_{xx}^{H_2} + \left(\frac{h}{2}\right)^2 A_{xx}^{H_2}\right]$$

Similarly, other terms in expressions for stiffness parameters can be expressed using general notations as;

$$\begin{aligned}
\iint_{H_{1,2}} Q_{ij}^{H_{1,2}}\ z^2\ dA\ &= b\left[D_{ij}^{H_{1,2}} \pm hB_{ij}^{H_{1,2}} + \left(\frac{h}{2}\right)^2 A_{ij}^{H_{1,2}}\right] \\
\iint_{V_{1,2}} Q_{ij}^{V_{1,2}}\ y^2\ dA\ &= h\left[D_{ij}^{V_{1,2}} \pm bB_{ij}^{V_{1,2}} + \left(\frac{b}{2}\right)^2 A_{ij}^{V_{1,2}}\right] \\
\iint_{H_{1,2}} Q_{ij}^{H_{1,2}}\ y^2\ dA\ &= \frac{b^3}{12}\ A_{1,2}^{H_{1,2}} \\
\iint_{V_{1,2}} Q_{ij}^{V_{1,2}}\ z^2\ dA\ &= \frac{h^3}{12}\ A_{1,2}^{V_{1,2}} \\
\iint_{H_{1,2}} Q_{ij}^{H_{1,2}}\ z\ dA\ &= b\left[B_{ij}^{H_{1,2}} \pm \left(\frac{h}{2}\right) A_{ij}^{H_{1,2}}\right] \\
\iint_{V_{1,2}} Q_{ij}^{V_{1,2}}\ y\ dA\ &= h\left[B_{ij}^{V_{1,2}} \pm \left(\frac{b}{2}\right) A_{ij}^{V_{1,2}}\right] \\
\iint_{H_{1,2}} Q_{ij}^{H_{1,2}}\ dA\ &= b\ A_{ij}^{H_{1,2}} \\
\iint_{V_{1,2}} Q_{ij}^{V_{1,2}}\ dA\ &= h\ A_{ij}^{V_{1,2}}
\end{aligned}$$

Using the expressions given above, the mathematical model to estimate the stiffness components of a composite box-beam is programmed in `MATLAB`.

4.1.2 CASE-II: Design of a critical section of a bicycle handle bar for maximum safety factor

The bicycle handle bar is subjected to loads from the rider, as shown in figure 4–3a. The goal is to design a lay-up for a sandwich beam construction at a critical section of the handle. We assume that the critical section is made up of a rectangular sandwich

plate as shown in figure 4–3b. The cyclic twisting of the handle is resolved into three loads, an in-plane shear load N_6, a bending moment M_1, and a twisting moment M_6 as shown in figure 4–3b. We assume the width and the length of the plate are fixed

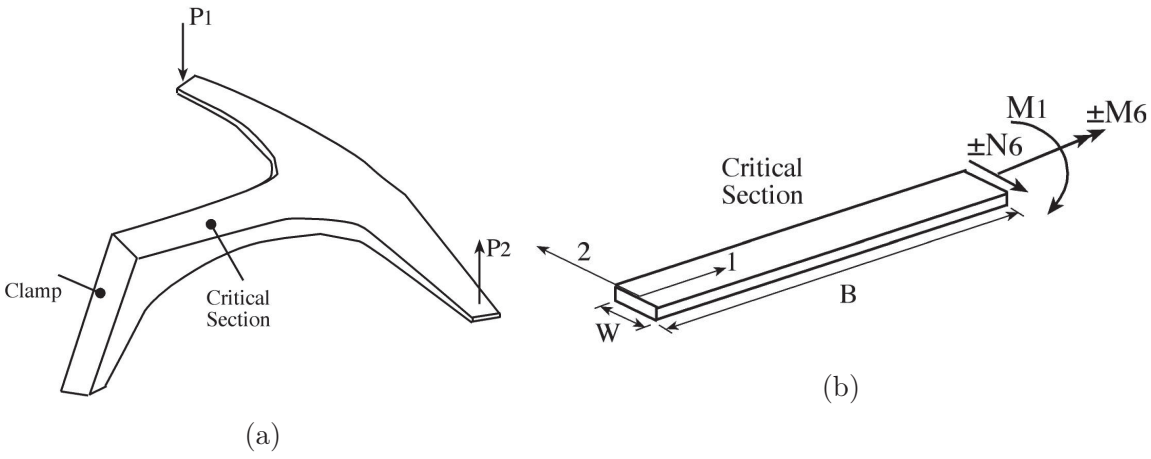


Figure 4–3: Bicycle handle bar

to $W = 3cm$ and $B = 10cm$ respectively. The loads $M_1 = 5.5kN$, $M_6 = \pm 3.25kN$, and $N_6 = \pm 89kN/m$ (please note that these are loads per unit width) result into two load cases;

1. **Load Case:1**

   $M_1 = 5.5kN,\ M_6 = 3.25kN,\ N_6 = 89kN/m$

2. **Load Case:2**

   $M_1 = 5.5kN,\ M_6 = -3.25kN,\ N_6 = -89kN/m$

The goal is to design the plate such that, it satisfies three failure criteria, maximum stress failure criterion, Tsai-Wu quadratic failure criterion, and Hashin failure criterion with the minimum safety factor for any layer to be at least 1.5. The

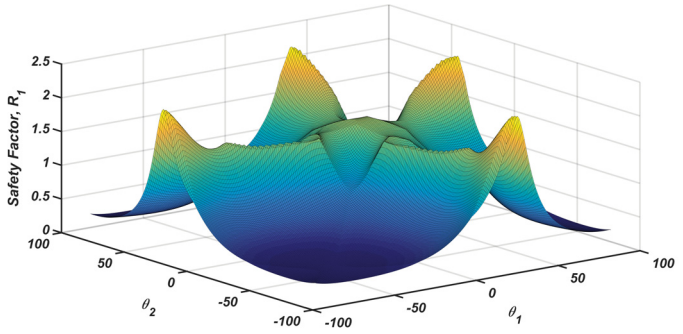
lightest possible design with a symmetric layup and with the $3cm$ thick honeycomb core at the middle of the laminate consists of 12 layers (plies) of the unidirectional Graphite/Thermoplastic (AS4/PEEK) composite. The material properties for AS4/PEEK are given in the table 4–2. The symmetry about the mid-plane re-

Table 4–2: Material properties for Graphite/Thermoplastic (AS4/PEEK)

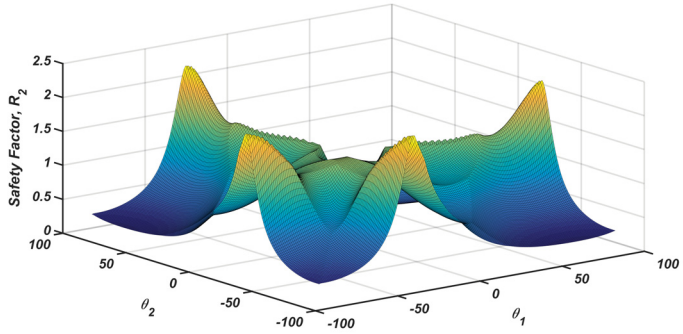
Elastic Properties			Strength Properties		
E_1 (GPa)	=	134	F_{1t} (MPa)	=	2130
E_2 (GPa)	=	8.9	F_{2t} (MPa)	=	80
G_{12} (GPa)	=	5.1	F_{1c} (MPa)	=	1100
ν_{12}	=	0.28	F_{2c} (MPa)	=	200
			F_{12} (MPa)	=	160

duces the number of design variables to 6 ply orientations. The laminate, in this case, can be represented by the laminate code $[\theta_1/\theta_2/\theta_3/\theta_4/\theta_5/\theta_6]_s$.

Safety factors, R_1 for the load case-1 and R_2 for the load case-2, are plotted as functions of ply orientation angles in the figure 4–4. The surface plots are generated for $\theta_1 = \theta_2 = \theta_3 = \theta_a$, and $\theta_4 = \theta_5 = \theta_6 = \theta_b$. It can be seen that, the safety factor is a highly non-linear function with multiple local minima and maxima. It can also been from figure 4–4 that, even though the shear load and the twisting moment only change the direction and the magnitude of these two loads remain the same, the safety factor as a function of the ply orientations behaves quite differently in the design space. Moreover, figure 4–5 shows the objective space, where R1 and R2 are plotted against each other. Here the red 'x' markers represent the safety factors for a 12 layer uni-directional symmetric laminate, $[(\theta)_6]_S$ (i.e. all six plies have the



(a) Safety factor R_1 as a function of ply orientation angles, [$(\theta_a)_3/(\theta_b)_3$].



(b) Safety factor R_2 as a function of ply orientation angles, [$(\theta_a)_3/(\theta_b)_3$].

Figure 4–4: Safety factor vs Fiber orientation angles. (a): For load case-1, (b): For load case-2

same fiber orientation), where θ is in the range of $-89°$ to $90°$. The blue 'o' markers represent the symmetric multi-directional laminate, $[(\theta_1)/(\theta_2)/(\theta_3)/(\theta_4)/(\theta_5)/(\theta_6)]_S$, where θ_i is in the range $-89°$ to $90°$ and $\theta_1 \neq \theta_2 \neq \theta_3 \neq \theta_4 \neq \theta_5 \neq \theta_6$.

The bi-objective maximization problem is first solved as an unconstrained problem. The results for this formulation are reported in subsection 5.2.1. The problem is then formulated as a constrained bi-objective maximization problem, and the results for this formulation are reported in subsection 5.2.2.

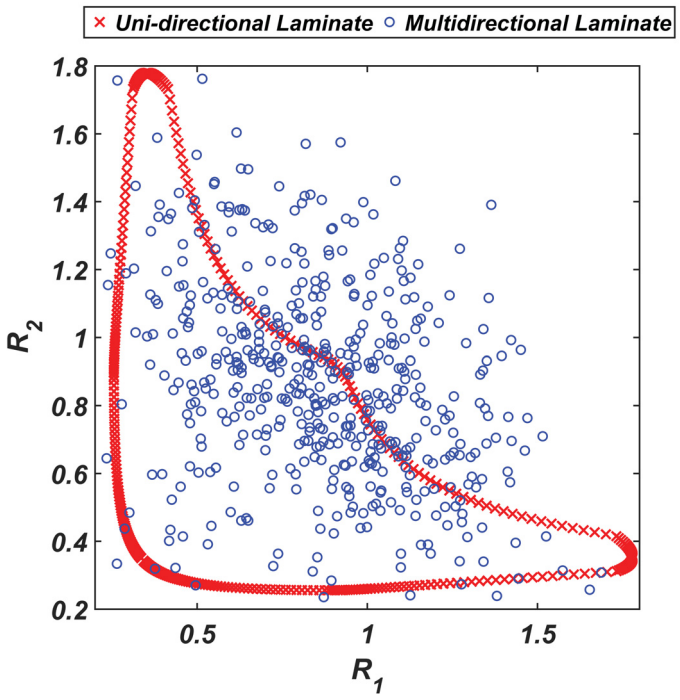


Figure 4–5: Safety factors, R_1 and R_2, as functions of fiber orientation angle

4.2 Algorithms

The primary objective in this thesis, as mentioned earlier, is to compare the performance of mesh adaptive direct search (MADS) with the GAs. The variant of the GA used in this study, the nested GA, and the MADS are briefly described in the following subsections.

4.2.1 Nested GA approach

A nested GA based on the one reported in [27] to solve the problem formulated in previous section is shown in figure 4–6. The algorithm uses two subroutines. The *Inner loop* finds the optimal stacking sequence for a given geometry, and the *Outer*

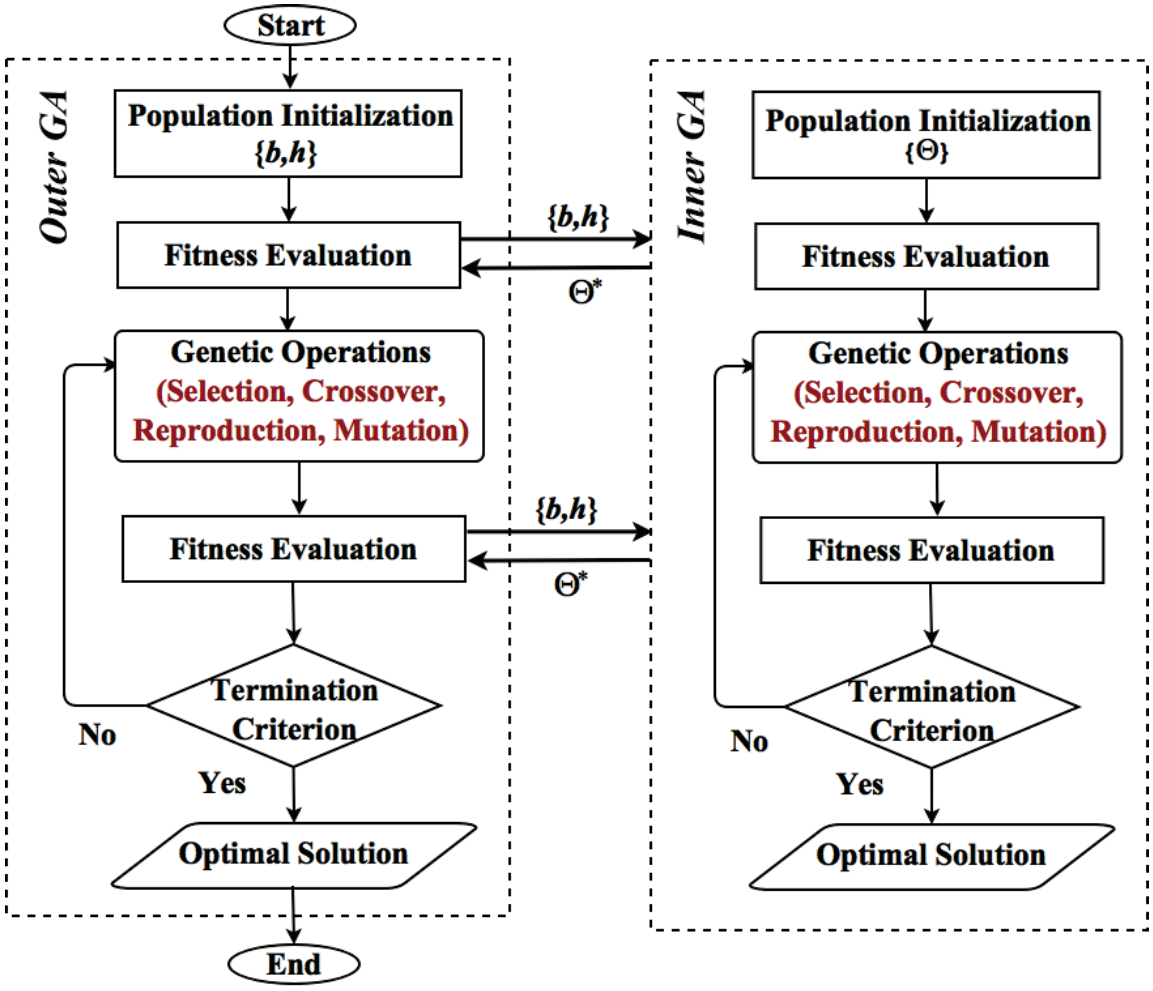


Figure 4–6: Representation of global and local GA

loop finds the optimal box geometry. In each iteration, the *Outer GA* randomly generates a population of $[b, h]$ (box geometry), and for each $[b, h]$ the *Inner GA* finds an optimal stacking sequence Θ^* and calculates the objective function value for $[b, h, \Theta^*]$. This optimal stacking sequence and the objective function value is then returned to the *Outer GA*.

The helicopter rotor box-beam target stiffness achievement problem presented in section 4.1.1 is solved using the nested GA approach and we compare the performance of this approach to the all-in-one approach using MADS algorithm. A brief description on MADS is presented in next section.

4.2.2 Mesh Adaptive Direct Search

Audet and Dennis introduced the Mesh Adaptive Direct Search (MADS) algorithm to solve non-linear constrained optimization problems [1]. MADS is an extension to the Generalized Pattern Search (GPS) algorithm. MADS allows the local search in an asymptotically dense set of directions in the neighborhood of a current iterate. This step of local exploration of the design space is usually referred as *Poll* step. The algorithm is backed by the convergence to the second-order stationary points based on Clark's calculus [43] for non-smooth functions [1, 2]. Besides the advantage of asymptotically dense polling directions over GPS, MADS can handle non-linear constraints with the barrier approach. This implies that, instead of applying the algorithm directly on the objective function f, MADS uses the barrier function, f_Ω, which is equal to f in the domain Ω and $+\infty$ outside Ω [1]. Here, Ω represents the feasible design space defined by constraints and is a subset of the entire design space.

MADS has been successfully implemented in a software package, called NOMAD (Nonlinear Optimization by Mesh Adaptive Direct Search) [44, 45, 46]. NOMAD is designed to handle black-box optimization problems, where optimizing functions are often costly programs or simulations with no gradient information available and

it is unworthy to calculate or approximate the gradients due to numerical noise and/or high computational cost. NOMAD is equiped with various features such as, bi-objective optimization, the ability to handle continuous, integer, binary, and categorical variables at the same time (i.e. Mixed Variable Programming), which according to our knowledge, no other optimization software provides all together. It also implements the Variable Neighborhood Search (VNS) as one of the in-built search strategies to make it a *Global Optimization Tool* [39]. As mentioned by authors, although the fundamental structure of both these algorithms are complementary, the convergence analysis of the combination follows directly to the convergence analysis of MADS. Apart from VNS, NOMAD also provides the user with generic search strategies such as, Model Search intoduced by Conn and Le Digabel [40], Speculative search, Latin-hypercube search. Besides academic researchers, NOMAD is being used in industries such as, Boeing, Airbus, ExxonMobil, GM, and Hydro-Québec, etc [45, 46]. Readers may refer [44, 45, 46] for detail information about NOMAD and its implementation.

Table 4–3 shows a high-level representation of MADS as described in [44]. At the beginning, the user must provide a initial point $x_0 \in \mathbb{R}^n$ in order to start the algorithm. Each iteration in MADS is denoted by the index k, as can be seen in table 4–3. At every iteration, MADS evaluates the function(s) at a finite number of trial points carefully selected from a conceptual mesh. This is done in two steps: an optional and more flexible *Search* step, and a mandatory and more rigorous *Poll* step. When we say more flexible search we mean that, *Search* allows the user to take advantage of his/her knowledge about the problem and define a strategy to choose the trial

Table 4–3: High-level Representation of MADS Algorithm

Initialization: Let $x_0 \in \mathbb{R}^n$ be an initial point and set the iteration counter $k \leftarrow 0$
Main loop:
repeat
    SEARCH on the mesh to find a better solution than x_k
    if *the* SEARCH *failed* **then**
        POLL on the mesh to find a better solution than x_k
    if *a better solution than* x_k *was found by either* SEARCH *or the* POLL **then**
        call it x_{k+1} and coarsen the mesh
    else
        set $x_{k+1} = x_k$ and refine the mesh
    Update parameters and set $k \leftarrow k + 1$
until *Stopping criterion is satisfied;*

points where the function(s) will be evaluated. Moreover, the uses can also use the available generic search strategies such as, speculative search, variable neighborhood search (VNS), Latin-hypercube sampling, model search, etc. The *Poll* step, on the other hand, has more rigorous restrictions on selecting the trial points to satisfy the convergence analysis of the algorithm. Readers may refer [1] and [47] for details on how the algorithms generates trial points during poll using a probabilistic strategy in [1] and a deterministic strategy in [47].

As mentioned earlier, both *Search* and *Poll* steps generate trial points belonging to a conceptual mesh, denoted as M_k. At any iteration k, the mesh structure is represented by the following expression [1],

$$M_k \;=\; \bigcup_{x \in V_k} \{x + \Delta_k^m D z : z \in \mathbb{N}^{nD}\} \tag{4.2.1}$$

where, $\Delta_k^m \in \mathbb{R}^+$ is a mesh size parameter, D is a $n \times n_D$ matrix representing a fixed finite set of n_D directions in $\mathbb{R}^n$ and is called the set of mesh directions. V_k is the set

of points where the objective and constraints have been evaluated by the start of k^{th} iteration where, V_0 contains the staring point(s).

Each MADS iteration starts with the *Search* step where a set of trial points is generated on a given mesh. Figure 4–7 illustrates this with a hypothetical example using a 2D mesh structure. Here the points t_1, t_2 and t_3 represent the set of trial point generated during the search step. Then, the algorithm launches the black box to evaluate

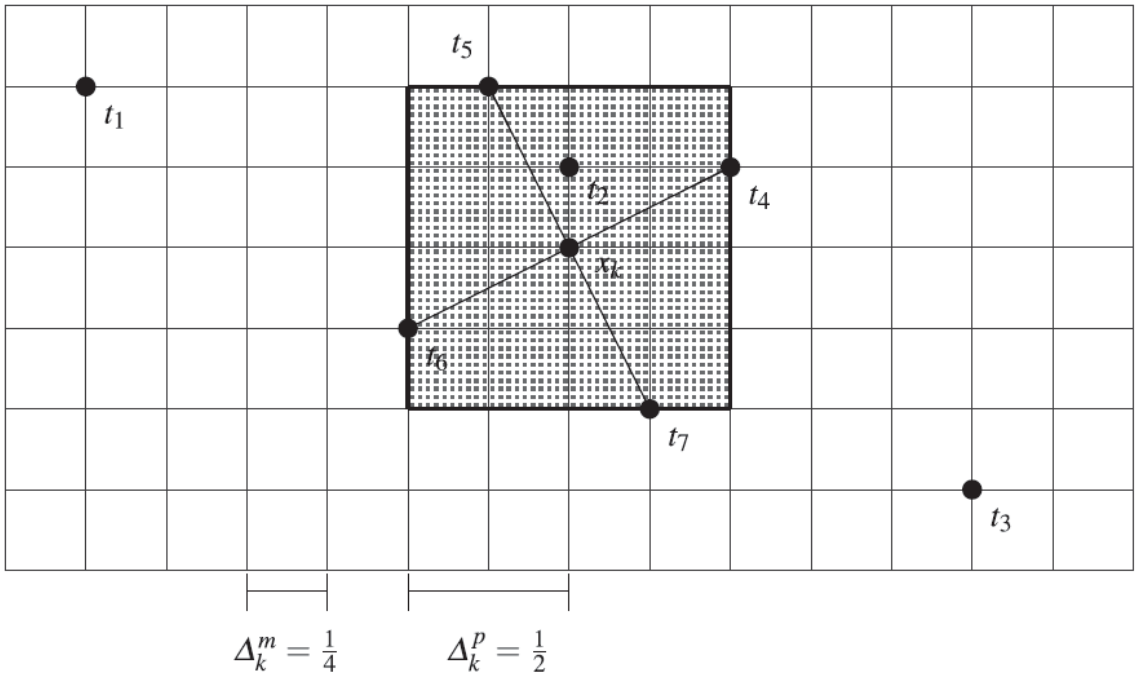


Figure 4–7: Example of MADS trial points during *Search* and *Poll* in a $\mathbb{R}^2$ design space. Showing *Search* trial points $\{t_1, t_2, t_3\}$ and *Poll* trial points $\{t_4, t_5, t_6, t_7\}$. Source: [41]

these trial points to test if the trial points are better than the current incumbent x_k. This step can be opportunistic, i.e. the iterations terminates if the algorithm

finds a point better than the current incumbent. And the new iteration starts with the recently found incumbent. If the search fails to find a better trial point on the mesh, the poll step generates a new set of trial points near the incumbent x_k. Again, the algorithm calls black-box program to evaluate the trial points to find a better point than the current incumbent. The poll step may be terminated as soon as a better point is found (opportunistic approach). The opportunistic approach during search and poll may results in the faster convergence. At the end of each iteration, the parameters are updated based on the outcome. There are two possibilities. If either the search or the poll generates a trial point t which is better than the current incumbent x_k, then the incumbent for the next iteration x_{k+1} is set to t, i.e. $x_{k+1} \leftarrow t$ and both the mesh size parameter and the poll size parameter are increased (mesh coarsening) or kept to the same value. Otherwise, the poll size parameter is decreased and the mesh size parameter is either decreased (mesh refining) or kept the same and the the algorithm starts the new iteration with the same incumbent as the last iteration, i.e. $x_{k+1} \leftarrow x_k$. Again, coarsening the mesh after a successful iterations helps in faster convergence.

After every failed iteration, mesh size Δ_k^m reduces much faster than poll size Δ_k^p [1, 2]. Which implies that, MADS can choose the polling direction from a larger set, i.e. the set of polling directions is asymptotically dense. To illustrate this, let us consider an example of a design space $\mathbb{R}^2$, where the set of directions, D, consist of eight directions $\{(d_1, d_2)^T \neq (0,0)^T : d_1, d_2 \in \{-1, 0, 1\}\}$. Figure 4–8 and 4–9 show frames generated by GPS and MADS, respectively, for different mesh size parameters. Note that, for MADS the pole size parameter is taken to be $n\sqrt{\Delta_k^m}$. It is evident from

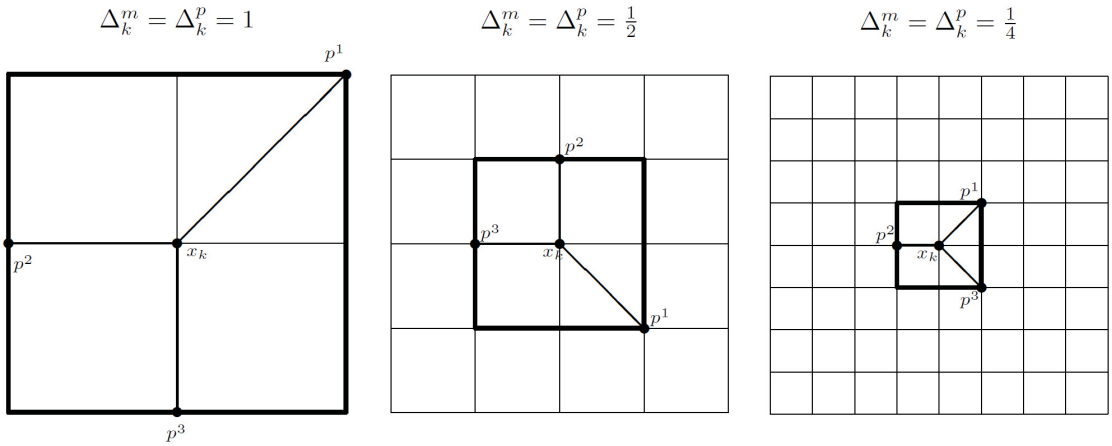


Figure 4–8: Example of GPS frames in $\mathbb{R}^2$ design space. Source: [1]

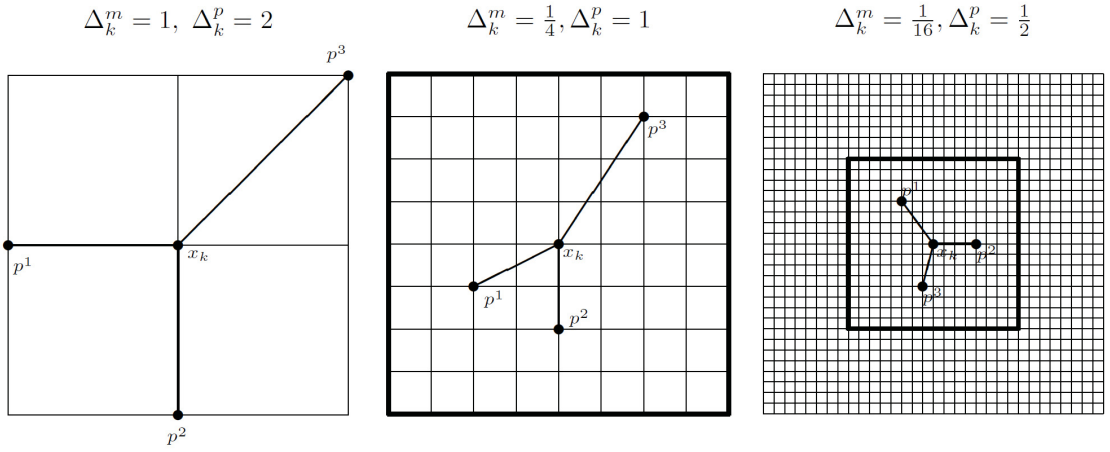


Figure 4–9: Example of MADS frames in $\mathbb{R}^2$ design space. Source: [1]

the figure 4–8 that, in GPS regardless of iteration, the mesh size, and the poll size, there could only be eight distinct positive bases with three direction to choose from. In MADS (figure 4–9), on the other hand, the set of distinct positive bases and the directions grow denser and denser as the mesh size and the poll size are reduced. A

numerical example to prove that MADS generates asymptotically dense set of poll directions given any directions in $\mathbb{R}^n$ is given in [1].

The two problem formulations and a brief description of the optimization algorithms used in this thesis are presented in this chapter. A brief description on how MADS works, its implementation in NOMAD and some insight on the features available in NOMAD are also presented in this chapter. Both the problems presented in this chapter are solved to find the optimum solution(s). The results obtained using the GAs and the MADS are analyzed to compare the performance of the two algorithms on parameters such as computational cost, convergence rate and the quality of the solution. We present the results and the discussion in the next chapter.

CHAPTER 5
Results and Discussion

Composite stacking sequence optimization problems described in section 4.1.1 and 4.1.2 were solved using MADS and GAs. Results and the assessment of the two algorithms based on these results are presented in this chapter. The primary goal of this investigation is to compare the MADS against GAs on parameters, such as convergence rate, computational cost, and quality of the solution.

5.1 CASE-I: Helicopter rotor box-beam target stiffness achievement

The helicopter rotor composite box-beam target stiffness achievement problem formulated by Murugan et. al in [27] and reported in 4.1.1 is solved using two variants of the genetic algorithm and MADS. The problem is first solved using a nested GA approach proposed by Murugan et.al in [27] and the results are reported in section 4.2.1. Then, the problem is solved using Mesh Adaptive Direct Search (MADS). Lastly, we use the mixed variable genetic algorithm from the `MATLAB` optimization toolbox to solve the problem using all-in-one GA approach. Results and discussion about these three approaches and the comparison between the algorithms are presented in the upcoming sections.

5.1.1 Nested GA approach

The target stiffness achievement problem is formulated with 7 variables, out of which the first two variables are continuous variables and define the box geometry in inches, and the rest five variables are of discrete type. They define fiber orientations in the box-beam walls. We assume a 10° discretization. In the nested approach, the problem is partitioned into two loops separating the continuous and discrete variables as described in section 4.2.1.

Genetic algorithms are probabilistic in nature, hence the algorithm is run 10 times with the same parameter setting. Results for these runs are averaged to assess the performance of the algorithm. Moreover, since the performance of GAs largely depend on the population size, we repeat the procedure for different population sizes for the inner loop. The average number of function evaluations, best, worst, and mean of the objective, and the standard deviation in objective for different population sizes are reported in Table 5–1.

Table 5–1: CASE-I: Nested GA Results

Population Size	**Avg. function evaluations**	**Objective**			**Standard Deviation**
		min	max	mean	
30	1,088,427	0.1931	1.2489	0.6371	0.3791
50	1,989,780	0.0224	0.9482	0.4359	0.2931
80	3,183,180	0.0500	0.9728	0.3699	0.3108
100	3,978,780	0.0758	0.4633	0.2495	0.1122

The average number of function evaluations and objective function values (minimum, maximum, and mean) are plotted against population size in figures 5–1 and 5–2, respectively. It is evident from these results that, the mean value of the objective

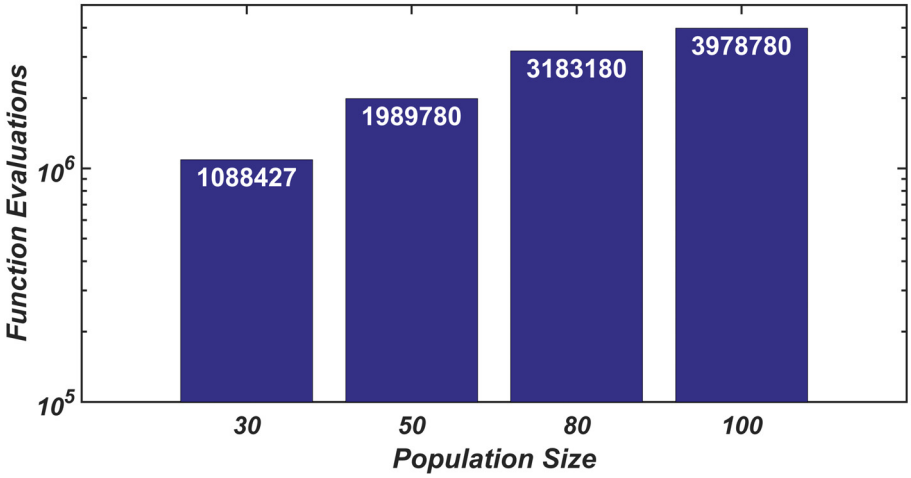


Figure 5–1: Nested GA: Average number of function evaluations vs Population size (CASE-I)

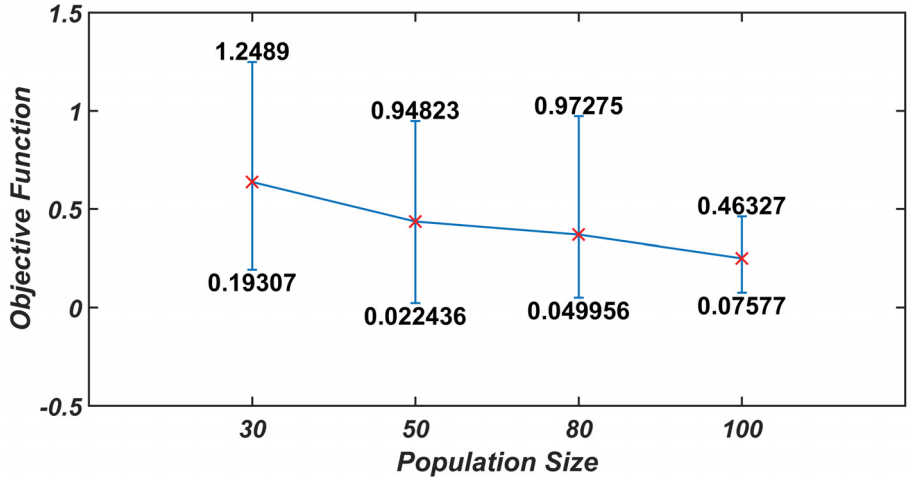


Figure 5–2: Nested GA: Objective vs Population size (CASE-I)

function decreases with an increase in the population size, whereas the number of

function evaluations increase with an increase in the population size as expected. Moreover, the standard deviation in the solutions generated during 10 runs of each population size does not follow this trend. It is evident from these results that, the quality of the solutions generated by GAs cannot be characterized, because among the 40 runs performed, the best solution was obtained with the population size of 50, not with the larger population size of 80 or 100. This means that, we still cannot quantify the quality of the solutions with respect to the population size. The only logical comment one can make with these results is that, the odds of converging to the worst solution can be decreased by increasing the population size.

5.1.2 MADS: All-in-one Approach

Results obtained with mixed variable programming in NOMAD are reported in this section. In the first part, results obtained with only the poll step are shown, whereas the second part covers results using MADS with the search. Throughout this study, we used Ortho-MADS [47] to choose the directions during the poll step.

I **MADS with only Poll**

MADS is run five times with randomly selected starting points (X_0) as shown in table 5–2. First four starting points are formed form upper and lower bounds on variables, whereas fifth starting point was formed with randomly selected values for variables. The objective function values for the initial design, $f(\mathbf{x_0})$, and for the final solution, $f(\mathbf{x}^*)$, the number of function evaluations, and the termination criterion are reported in table 5–3. The termination criteria "*Mesh index limit*" means that, the algorithm has reached a local optimum and the

Table 5–2: CASE-I: Starting points for MADS

Run#	$\mathbf{x_0}$
1	[3.00, 2.00, 0, 0, 0, 0, 0]
2	[5.00, 3.00, 0, 0, 0, 0, 0]
3	[3.00, 2.00, 90, 90, 90, 90, 90]
4	[5.00, 3.00, 90, 90, 90, 90, 90]
5	[3.60, 2.50, 40, 20, 70, 0, 60]

Table 5–3: CASE-I: MADS (Poll only) Results

Run#	**Objective**		**Function Evaluations**	**Termination Criterion**
	$f(\mathbf{x_0})$	$f(\mathbf{x}^*)$		
1	77.4619	3.9351	939	*Mesh index limit*
2	439.1329	0.1251	1,265	*Mesh index limit*
3	77.4619	1.1531	1,025	*Mesh index limit*
4	54.5267	0.8226	1,170	*Mesh index limit*
5	58.3098	0.1097	1,544	*Mesh index limit*

mesh size is reduced to the lowest possible value. It can be seen from table 5–3 that, despite starting from worse points, i.e. designs with a quite high percentage error between target and actual stiffness values, MADS could converge to local optima in less function evaluations than the nested GA approach presented in previous section. The average number of function evaluations for these 5 runs is 1188, which is less than the average number of function evaluations for the nested GA approach by almost 3 orders of magnitude. In other words, MADS certainly is computationally less expensive than the nested GA, which is a great advantage for an optimization algorithm, especially in the field of engineering

optimization, where function values are calculated using expensive computer simulations.

II **MADS with Search and Poll**

NOMAD has 4 in built search strategies, such as Speculative Search, Variable Neighborhood Search(VNS), Latin-Hypercube Search, and the Model Search, which can either be used alone or in combination. To show the effectiveness of search step in escaping local optima and further improving quality of the solutions, we use three search strategies, viz. Speculative Search, Model Search, and Variable Neighborhood Search (VNS). Results obtained with same starting points from table 5–2 are reported in table 5–4 in the similar format as reported in the previous part.

Table 5–4: CASE-I: MADS (Poll+Search) Results

Run#	**Objective**		**Function Evaluations**	**Termination Criterion**
	$f(\mathbf{x_0})$	$f(\mathbf{x}^*)$		
1	77.4619	0.0960 (97.56%)	4,251	Mesh index limit
2	439.1329	0.0730 (41.65%)	3,445	Mesh index limit
3	77.4619	0.4032 (65.03%)	2,453	Mesh index limit
4	54.5267	0.8532 (-3.72%)	3,386	Mesh index limit
5	58.3098	0.0152 (86.14%)	2,480	Mesh index limit

The percentage values in brackets are the percentage improvements in the solution obtained by employing the search step over the one obtained with only the poll step. It can be seen that, except the fourth run, MADS with the search

step could improve quality of the solution. MADS with the search strategy performed better and could escape local optima to find a better solution without much increase in the number of function evaluations. The average number of function evaluations in this case is 3203. Furthermore, MADS still preserves the convergence properties of solutions generated.

5.1.3 GA: All-in-one Approach

In all-in-one approach, we use the mixed variable genetic algorithm in `MATLAB` to solve the optimization problem. Similar to the nested GA approach, the algorithm is run 10 times each with different population size. Results are reported in table 5–5 corresponding to the population size. The maximum number of generations set to 600 for all runs. However, none of the 50 runs performed could reach allowable maximum number of generations. The algorithm terminated before it could reach the maximum number of generations, because the average change in the objective value is below its threshold of 10^{-6}. It is evident from table 5–5 and figures 5–3

Table 5–5: CASE-I: All-in-one GA Results

Population Size	**Avg. function evaluations**	**Objective Function**			**Standard Deviation**
		min	max	mean	
50	15,631	0.0380	1.4384	0.3575	0.4320
100	23,661	0.0667	0.8315	0.2599	0.2161
200	31,841	0.0105	1.0992	0.3068	0.3137
500	79,401	0.0236	0.4248	0.2081	0.1622
1,000	136,001	0.0184	0.4248	0.1679	0.1510

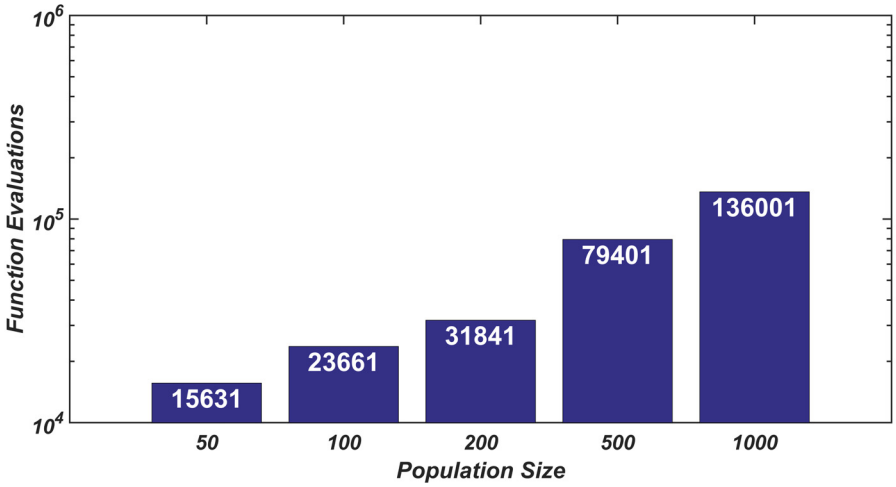


Figure 5–3: All-in-one GA: Function evaluations vs Population size (CASE-I)

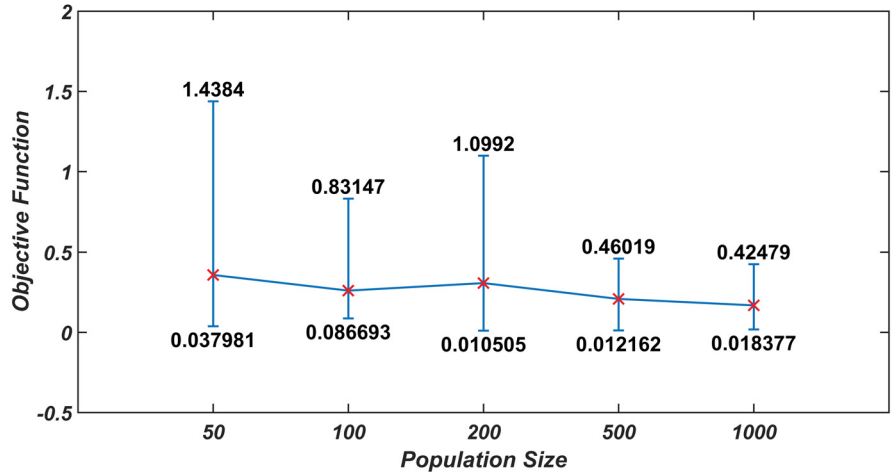


Figure 5–4: All-in-one GA: Objective vs Population size (CASE-I)

and 5–4 that, all-in-one formulation using mixed variable GA performed better than that of the nested approach. The mixed variable GA was able to converge to better solutions in less number of function evaluations than the nested GA approach. In fact the number of function evaluations were lower than that of nested GA by more than an order of magnitude. However, it was still more than the all-in-one approach

using MADS. Also, the quality of the solution (i.e. the lowest error in the target and actual stiffness values) was at par when compared with MADS.

Besides this, figure 5–5 shows improvements in the objective against function evaluations for MADS (Run-1 in table 5–4) and all-in-one GA for 5 different population sizes as shown in table 5–5. Numbers in the legend represent the population size for

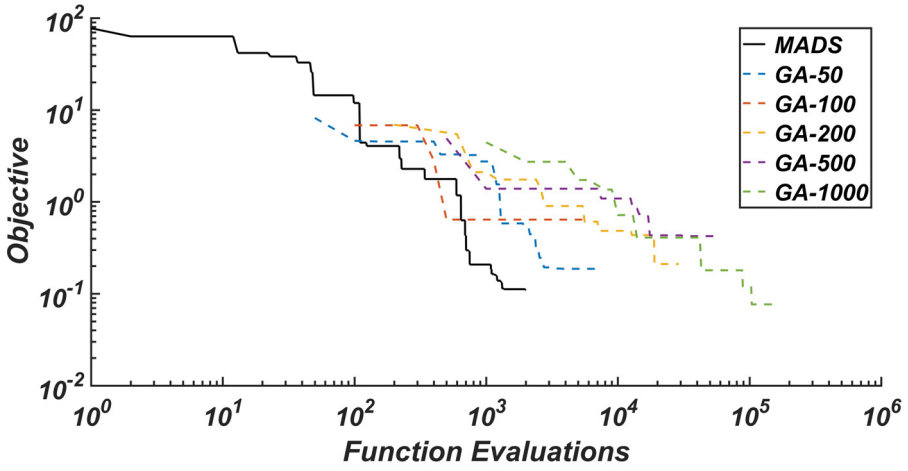


Figure 5–5: Convergence rate: MADS vs All-in-one GA (CASE-I)

the all-in-one GA. It can be seen from the figure 5–5 that, despite starting from the worse starting point, MADS converged faster than the all-in-one GA. As mentioned before, the convergence rate of an optimization algorithm is of vital importance in the field of engineering optimization, we believe that fast convergence is a great advantage MADS has over GAs.

5.2 CASE-II: Design of a critical section of a bicycle handle bar for maximum safety factor

We first solve the bi-objective maximization problem as an unconstrained problem. Results for this formulation are reported in subsection 5.2.1. The problem is then formulated as a constrained bi-objective maximization problem, and results for this formulation are reported in subsection 5.2.2.

5.2.1 Uncostrained bi-objective formulation

For unconstrained bi-objective formulation, we run the bi-objective MADS (bi-MADS) in NOMAD and the multi-objective GA (MOGA) in `MATLAB` to get the Pareto front approximations. The algorithm setting used for these algorithms are reported in table 5–6 and 5–7, and Pareto fronts obtained are shown in figure 5–6 and 5–7, respectively. respectively.

Table 5–6: MOGA setting

Population Size	`200`
# Generations	`100`
Creation Function	`Constraint dependent`
Selection Function	`Tournament`
Crossover Function	`Scattered`
Mutation Function	`Constraint dependent`

Please note that, since the MOGA is a probabilistic approach, it generates different Pareto front approximations every time we run the algorithm. Hence, the algorithm is run more than once. The Pareto front shown in the figure 5–6 is the best one of

Table 5–7: Bi-MADS setting

Starting Point	$[0_6^c]_S$
Speculative Search	`Disable`
Model Search	`Disable`
VNS	`Disable`
# Multi-MADS Runs	`100`
# Black box evaluations per Multi-MADS run	`200`
Polling direction type	`Ortho MADS 2N`

the Pareto fronts obtained during these runs. For MADS, on the other hand, we performed two runs; one without the search and one with the search. Algorithm parameters during these runs are set in such a way that, it performs approximately equal number of function evaluations as that of the MOGA. The Pareto front generated with bi-MADS is shown in figure 5–7.

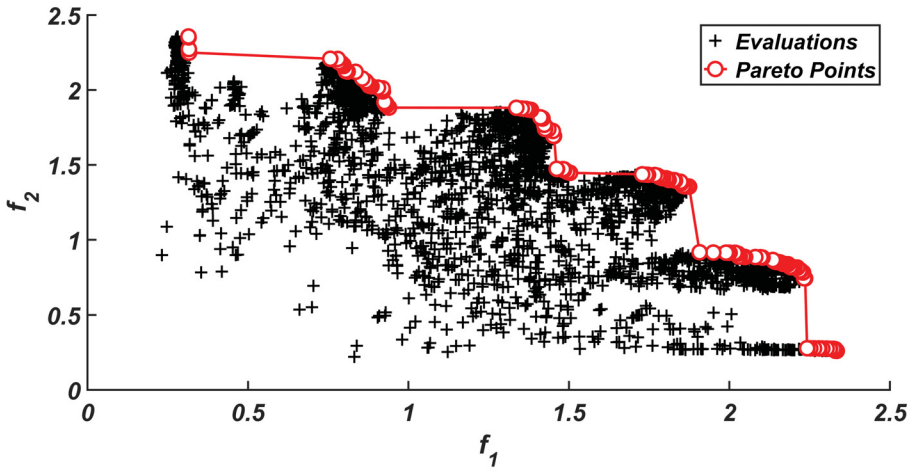


Figure 5–6: MOGA Pareto front

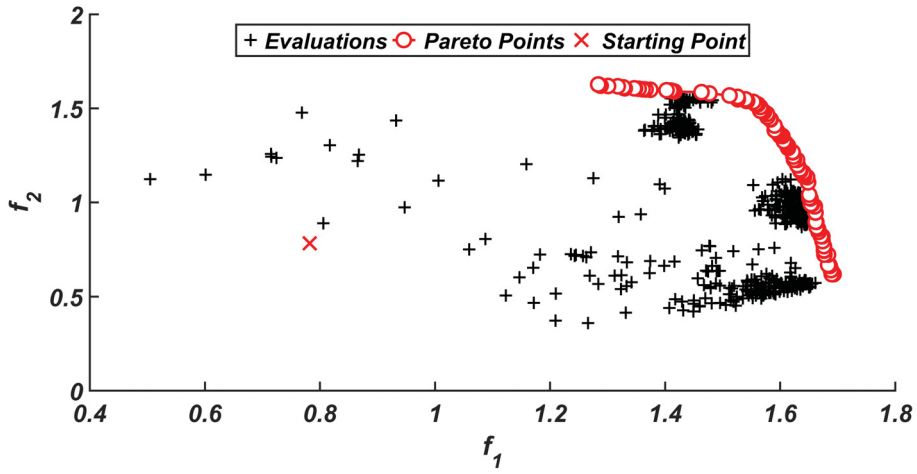


Figure 5–7: Bi-MADS Pareto front (Poll only)

Before we compare these Pareto fronts, it is interesting to notice that, a mere visual comparison shows that, the Pareto front with bi-MADS is smoother and continuous than that the one obtained with MOGA. Moreover, although bi-MADS generates a narrower Pareto front, it gives more number of better designs than the MOGA. To cover a wider objective space and have a wider Pareto front, we run the bi-MADS with Speculative Search, Model Search, and VNS enabled. The Pareto front obtained with this setting is shown in figure 5–8. It can be seen from the figure that, the Pareto front obtained form this run is wider, however, it is not continuous like the one shown in figure 5–7. The reason that the Pareto front with bi-MADS+search is wider is that, with the search, especially the variable neighborhood search, the algorithms leaves the local region and tries to find better design in multiple regions. This explains why the Pareto front with search is wider, since the algorithm covers a wider design space with the search.

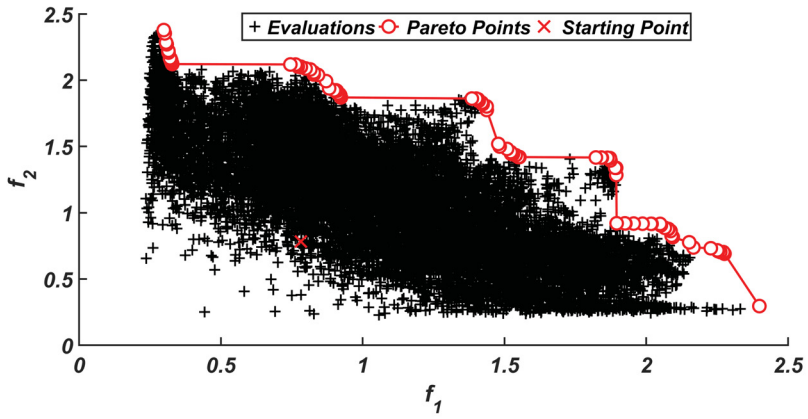


Figure 5–8: Bi-MADS Pareto front (Poll+Search)

There exist ways to compare these Pareto fronts, among which the hypervolume indicator is reported to be one of the most favorite choices [48, 49]. Moreover, it does not require the knowledge of the actual Pareto front. Hence, we compare the Pareto front approximations obtained from the MOGA and bi-MADS using the hypervolume indicator. The hypervolume indicator measures the size of the objective space between a reference point, r, and a Pareto front. The reference point, r is randomly chosen from the design space, i.e. $r \in \mathbb{R}^m$, and it should be dominated by the Pareto front under consideration. In case of bi-objective optimization, the hypervolume indicator essentially represents the area under the Pareto front bounded by the reference point as shown in figure 5–9. Here, the shaded area represents the hypervolume indicator for a minimization problem. The higher the hypervolume indicator, the better the Pareto front approximation.

Similar to the hypothetical case, figure 5–10 shows Pareto fronts obtained using MOGA and bi-MADS with sample objective points in the objective space. The hy-

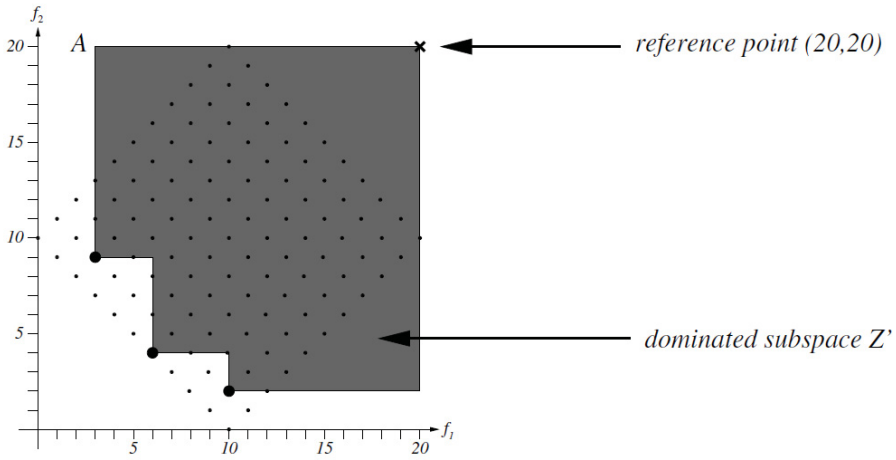


Figure 5–9: Illustration of the hypervolume indicator with hypothetical objective values for a bi-objective minimization. Source: [50]

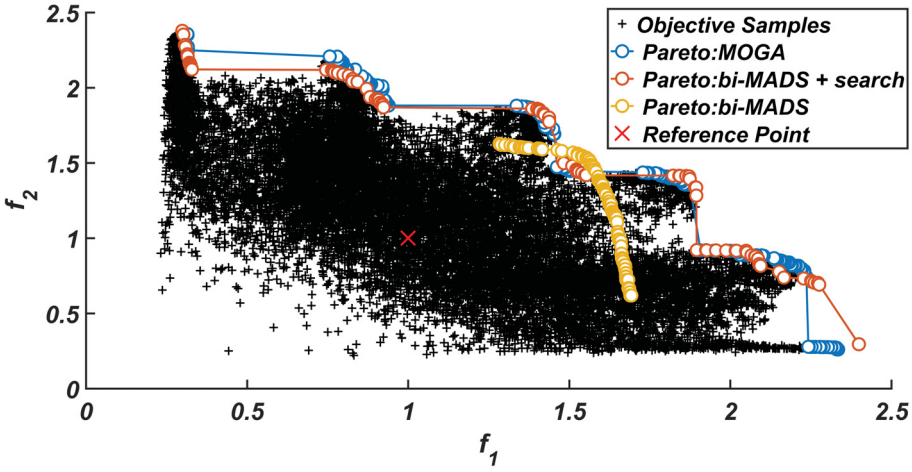


Figure 5–10: Pareto comparison: Bi-MADS vs MOGA

pervolume indicators for these three Pareto fronts are reported in Table 5–8. These

Table 5–8: Pareto comparison: Hypervolume indicators

MOGA	Bi-MADS	Bi-MADS+Search
1.7657	0.2731	1.8298

values are calculated using a MATLAB function developed by Johannes W. Kruisselbrink [51] with respect to the reference point, $[f_1(\mathbf{x}),\ f_2(\mathbf{x})] = [1,\ 1]$, as shown in figure 5–10.

It can be seen from visual inspection of the three fronts that, the Pareto front using MOGA (shown in blue) almost coincides with the Pareto front using bi-MADS+Search (shown in orange). The Pareto front using bi-MADS (shown in yellow) is dominated by other two fronts over most of the design space, which is also evident from hypervolume indicators. Hypervolume indicators show that, the Pareto front using bi-MADS+Search ($HV = 1.8298$) dominates the Pareto front using MOGA ($HV = 1.7657$). It is also interesting to notice that, Pareto points generated using bi-MADS without search dominate the other two Pareto fronts in the objective space where both the objectives are greater than 1.5. This is because, when we run the algorithm with the search, it performs iterations in multiple regions from the design space in search of Pareto optimal solution. Since bi-MADS uses series of single objective formulations of bi-objective problem and the series of formulations is constructed in such a way that it attempts an uniform coverage of the Pareto [52], the algorithm uses the allowed computational budget (the number of function evaluations) uniformly over the design space. Which is why bi-MADS with search could generate a wider Pareto front using given computational budget.

Now, please recall that, in this problem the objective space of interest is where both the objectives are greater than 1.5. Hence, before we make any concluding remarks about the quality of Pareto fronts generated by both these algorithms, we will restrict our objective space to be always greater than or equal to 1.5, i.e. $f_i(\mathbf{x}) \geq 1.5$.

Moreover, although bi-MADS with search produces a wider Pareto front than the bi-MADS without search, it spends the computational budget in the regions where it improves one of the objectives while compromising on the other. For instance, points where $f_1 > 2$ and $f_2 < 0.5$ and vise versa. To avoid this situation, we impose constraints on the objectives to always be at least 1.5. Which brings us to the constrained bi-objective optimization problem formulation. The numerical results for constrained bi-objective problem formulation are presented in the next section.

5.2.2 Constrained bi-objective formulation

In this problem, since we need to find a lay-up to satisfy the requirement that the safety factor for either of the load cases should be at lease 1.5, we imposed this condition as constraints on objectives. The problem formulation can be written as,

$$\begin{aligned} \max \quad & [\ f_1(\mathbf{x}), f_2(\mathbf{x})\ ] \\ \textit{subject to} \quad & C_i(\mathbf{x}) \geq 1.5 \cdots\cdots\cdots i = 1, 2 \\ & \mathbf{x} \in \mathbf{X}, \end{aligned}$$

where $C_1(\mathbf{x})$ and $C_2(\mathbf{x})$ are safety factors calculated for load case-I and load case-II, respectively.

We run MOGA and bi-MADS with the same algorithmic settings as shown in tables 5–6 and 5–7. MOGA was able to find Pareto points satisfying both constraints in 2 out of 10 runs. Rest of the runs were not able to find feasible designs satisfying both constraints. The first successful run found 2 Pareto points and the second successful

run found 11 Pareto points, i.e. total 13 points. The total of 13 Pareto points found in these 2 successful runs of are reported in the table 5–9 and the Pareto front approximation with the feasible designs dominated by Pareto front are plotted in figure 5–11.

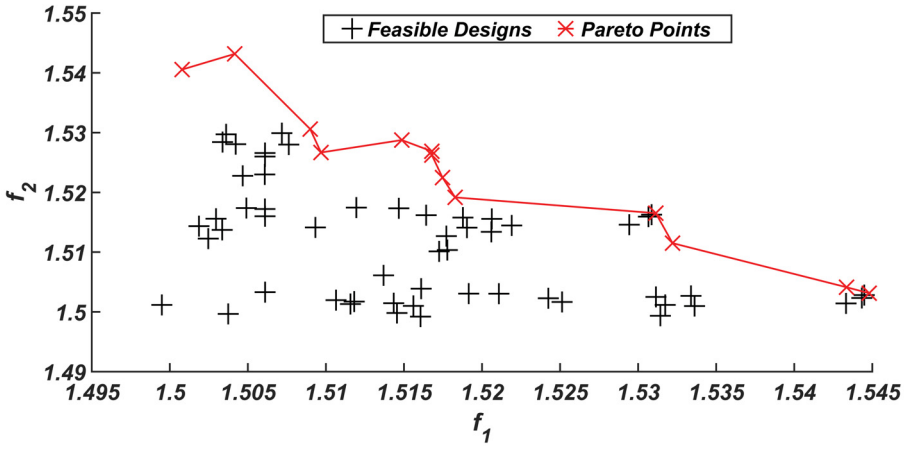


Figure 5–11: Pareto front: MOGA with constrained formulation

Similarly, 16 Pareto points generated by the bi-MADS are reported in the table 5–9 and figure 5–12 shows the Pareto with the feasible design points for this run.

To compare these two Pareto fronts, we use the hypervolume indicator. Figure 5–13 shows the Pareto fronts and sample points in the objective space. The lower bound imposed by constraints, i.e. $[f_1(\mathbf{x}),\ f_2(\mathbf{x})] = [1.5,\ 1.5]$, is taken as a reference point to calculated the hypervolume indicator.

The hypervolume indicators calculated for the Pareto fronts generated using MOGA and bi-MADS are $\mathbf{0.0018}$ and $\mathbf{0.0039}$, respectively. This shows that, the Pareto front approximation generated by bi-MADS is better than the one generated by MOGA.

Table 5–9: Pareto points for constrained bi-objective formulation

	MOGA		**Bi-MADS**	
	$f_1(\mathbf{x})$	$f_2(\mathbf{x})$	$f_1(\mathbf{x})$	$f_2(\mathbf{x})$
1	1.5448	1.5031	1.5648	1.5127
2	1.5434	1.5041	1.5642	1.5132
3	1.5322	1.5115	1.5637	1.5146
4	1.5311	1.5165	1.5583	1.5254
5	**1.5183**	**1.5191**	1.5568	1.5271
6	1.5175	1.5225	1.5510	1.5310
7	1.5168	1.5262	1.5501	1.5326
8	1.5168	1.5269	1.5491	1.5368
9	1.5149	1.5287	1.5475	1.5377
10	1.5097	1.5267	**1.5443**	**1.5447**
11	1.5090	1.5306	1.5413	1.5449
12	1.5042	1.5432	1.5410	1.5451
13	1.5008	1.5405	1.5362	1.5519
14	-	-	1.5282	1.5575
15	-	-	1.5125	1.5577
16	-	-	1.5052	1.5598

This conclusion, however, is based on the consideration that, the higher value of hypervolume indicator implies that the Pareto using bi-MADS dominates the one generated by MOGA. This is also clear from the figure 5–13. For the problem under consideration, the Pareto front generated by bi-MADS clearly dominates the Pareto front generated by MOGA. Moreover, among all the Pareto optimal solutions found, the best designs found with each of the two algorithms are shown in bold face in table 5–9. When we say the best design we mean the design with the largest minimum safety factor among f_1 and f_2. The best design with MOGA gives the safety factors

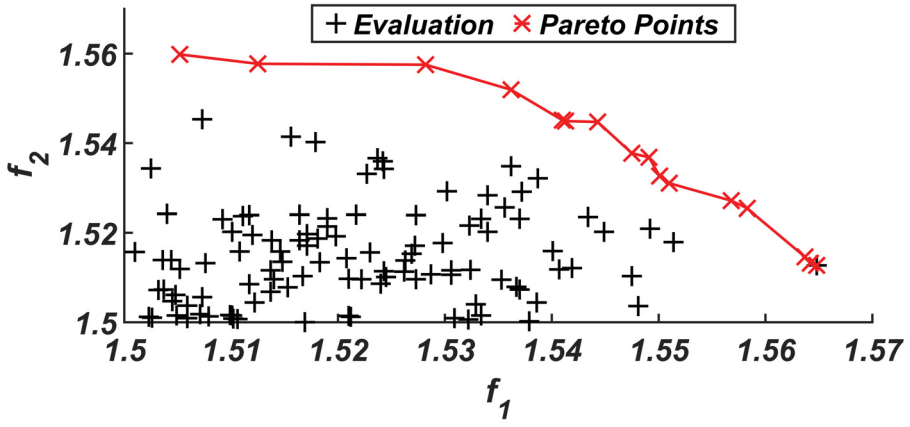


Figure 5–12: Pareto front: Bi-MADS with constrained formulation

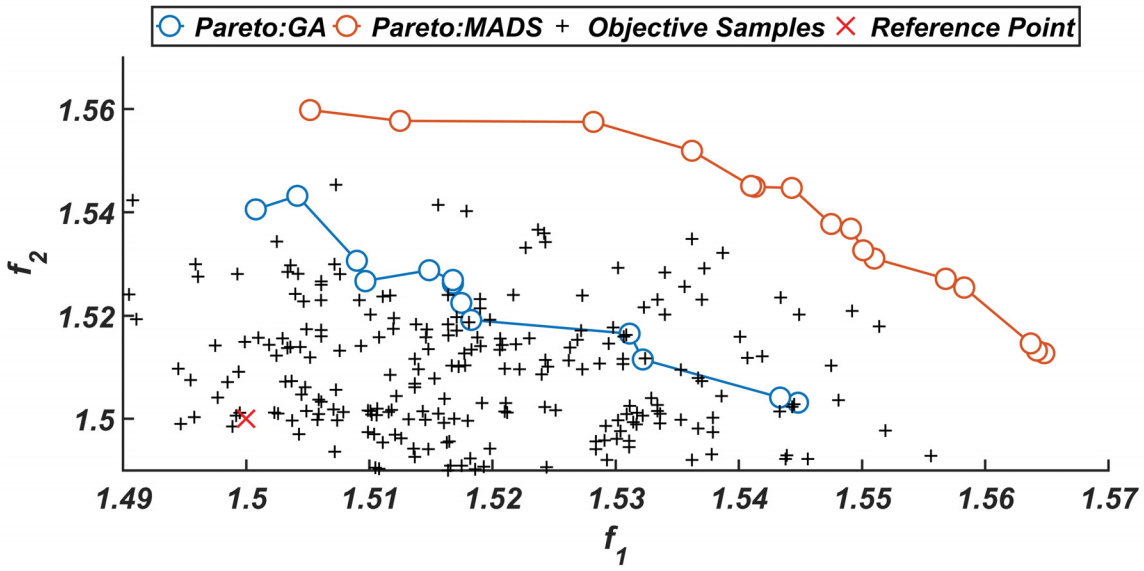


Figure 5–13: Feasible design space and Pareto fronts for constrained optimization

$f_1 = 1.5183$ and $f_2 = 1.5191$, whereas the best design with bi-MADS gives the safety factors $f_1 = 1.5443$ and $f_2 = 1.5447$.

5.3 Summary of Results

The numerical investigations have shown that, MADS outperformed GAs in stacking sequence optimization of laminated composites. For a single objective unconstrained problem with a non-linear objective function, we observed from the results that, the solutions generated with MADS with different starting points are either superior or at par than the solutions generated with the nested and all-in-one GA. Moreover, for the target stiffness achievement problem, the number of function evaluation for MADS are less than the nested GA approach proposed by Murugan et al. [27] by three orders of magnitude. All-in-one GA approach, on the other hand, performed better than the nested GA approach. However, still the average number of function evaluation with the all-in-one GA is more than the MADS by one order of magnitude. Besides computational cost, it is evident for figure 5–5 that, among the three approaches used to solve the target stiffness achievement problem, MADS converged faster than the GAs. Moving forward to the second problem, where we solve the bi-objective optimization problem, for a given allocated computational budget, the Pareto front generated by bi-MADS with and without search are better than the multi-objective GA. For both problems considered in this study, the quality of the solution with MADS is either at par or superior than the GAs. Moreover, MADS outperformed GAs in terms of the convergence rate and computational cost.

CHAPTER 6
Conclusion

6.1 Conclusion

Composite materials can be very promising considering their advantages over conventional materials. They provide a great flexibility in designing structural components in aerospace, land, and marine applications. However, the large number of variables involved make the design of laminated composite structures complex and tedious. This encourages designers to use optimization techniques to design more efficient structures using laminated composites while keeping in mind the physical and practical constraints. The presence of discrete variables, highly non-linear and multi-modal response functions are some of the challenges involved in composite design optimization. To address these challenges different optimization methods have been used in the past to solve composite stacking sequence optimization problems. A variety of problem formulations and customized versions of algorithms can be found in the literature addressing these challenges. Among all these algorithms, the genetic algorithms (GAs) are reported to be the most popular ones.

Genetic algorithm is definitely a very powerful tool in optimization with a broad scope. It has been reported to be very effective in solving a variety of optimization problems when other methods fall short. However, being a meta-heuristic approach

GAs generate a set of solutions according to arbitrarily chosen termination criteria and the optimality of these solutions cannot be characterized. Moreover, GAs require a high number of function evaluations before it can find a better solution and still finding the optimal solution is not always guarantied. In this thesis, we used the mesh adaptive direct search (MADS), a class of algorithms belonging to direct search methods, to try and address these limitations associated with GAs.

Direct search methods have proved to be very robust and reliable, both therotically and practically. Mesh adaptive direct search, one of the direct search algorithms which extends the pattern search algorithm by allowing the local exploration in asymptotically dense set of directions. It can handle noisy, non-smooth, highly non-linear function with mixed variable types (continuous, discrete, categorical). Besides, MADS is backed by the convergence analysis based on the Clark's calculus, which we feel is the greatest advantage MADS has over GAs. This encourages us to test MADS against GAs in solving stacking sequence optimization problems.

In our investigation of the results generated with variants of GA and MADS we observed that, MADS outperformed GAs in optimizing the fiber orientations in laminated composite structural elements. Note that, the results reported in the previous chapter are analyzed and quantified to compare both the techniques on computational cost, convergence rate and the quality of the solution.

The target stiffness achievement of a helicopter rotor composite box-beam problem is solved using nested GA, all-in-one GA and MADS. Among the three approaches, MADS is found to be the least computationally expensive. The number of function

evaluations with MADS is less than the nested GA and all-in-one GA by three orders of magnitude and one order of magnitude, respectively. MADS with only the poll step performed converged to local optima during the 5 runs with randomly chosen starting point with the average function evaluations of 1,188. For MADS with the three search strategies (variable neighborhood search, speculative search and model search), the average number of function evaluations is 3,203. It is evident that, MADS converges to the local optima in fairly less number of function evaluations, which in our opinion is vital in engineering optimization field. Especially when the functions are calculated using costly simulations, low computational cost is a great advantage of MADS over GAs. It is, however, important to see if the quality of the solutions generated with these algorithm before we conclude about the overall performance of these algorithm.

When we say the quality of the solution, how close an algorithm can find a solution to the desired solution. For the target stiffness achievement problem, the desired solution is to make the error between the target stiffness and the actual stiffness as small as possible, ideally zero. It can be seen that, MADS with only the poll step converges to the local optima, and the best design found with this setting is with an error of 0.12% error between stiffness values. The best designs found with the nested GA and all-in-one GA are with an error of 0.02% and 0.01%. The best design found with MADS with search strategies is with an error of 0.1%. It is interested to note here that, MADS with search strategies converged in merely 2,480 function evaluations. All-in-one GA on the other hand, took 31,841 function evaluations to reach the best design with the same error margins. Moreover, most of the solutions

generated with both the GA approaches were poor as compared to the solutions generated with MADS. One more point we would like to highlight here is that, MADS is more reliable than the GAs, since the solutions generated with MADS posses convergence properties.

For the second problem in consideration, i.e. design of a rectangular sandwich panel using bi-objective optimization, bi-MADS clearly generated better Pareto fronts than the multi-objective GA (MOGA), not only in terms of the Pareto spread but also Pareto front generated with bi-MADS consist of better designs. From all these observations, we can conclude that the solutions generated with MADS are either superior or at par with the solutions generated with GAs in both the problem formulations under consideration.

Moreover, it is evident from figire 5–5 that, despite starting from the much worst starting point, MADS converged faster than GAs. In GAs, we observed that the convergence rate is faster for the runs with lower populations size than the runs with a higher population size. However, it is still slower than the MADS. Moreover, the faster convergence is typically at a compromise in the quality of the solution, as GA with smaller population size fails to achieve better quality solutions. In other words, the convergence rate and the quality of the solutions in case of GAs are complementary to each other, and improvement in one may sometimes, if not always, deteriorate the other.

All in all, there is no doubt that GAs are a powerful tool in engineering optimization with a broad range of application. However, numerical investigation presented in

this thesis shows that MADS performs better than GAs in solving composite stacking sequence optimization problems formulated and presented in this thesis. MADS was found to converge faster than both variants of GA (nested and all-in-one) while solving the single objective mixed variable composite stacking sequence optimization problem. Moreover, the quality of the solutions with MADS was found to be either superiour or at par than those generated with GAs. The number of function evaluations required by MADS were found to be less than the number of function evaluations performed by GAs. Which, in our opinion, is a great advantage to have especially in engineering optimization. Also, quality of the solution generated with bi-objective MADS (bi-MADS) is better than that of multi-objective GA (MOGA). And with the availability of the sophisticated softwares like NOMAD, it has become easy to implement the algorithm. However, before we can generalize our conclusion about the performance of the MADS in composite design optimization field, we feel it is necessary to test MADS against more variants and custom approaches present in the literature to have a more clear picture on the performance of MADS in the field. This opens up many opportunities to further test MADS in the field of engineering optimization and it definitively worth more attention in the future.

6.2 Suggestions for Future Work

Although our initial analysis shows that, MADS has performance advantages in terms of the convergence rate and computational cost over GAs, it is important to note that the performance of optimization algorithms largely depends on the type and structure of the problems under consideration. Hence, to make our observations

more general, we feel it is necessary to test MADS with different problem formulations and with other popular evolutionary algorithms. It would be interesting to conduct the comparative studies similar to the one presented in this thesis to test the competence of MADS against other popular evolutionary algorithms such as, simulated annealing, artificial bee colony algorithm, particle swarm optimization, etc. As far as our knowledge, this is the first attempt to use MADS in composite stacking sequence optimization. We believe that, this study encourages research to try and test MADS in the composite optimization field and it gets more recognition in the future research in the field of engineering optimization.

BIBLIOGRAPHY

[1] C. Audet and J.E. Dennis, Jr. Mesh Adaptive Direct Search Algorithms for Constrained Optimization. *SIAM Journal on Optimization*, 17(1):188–217, 2006.

[2] Mark A Abramson and Charles Audet. Convergence of mesh adaptive direct search to second-order stationary points. *SIAM Journal on Optimization*, 17(2):606–619, 2006.

[3] Charles Audet. A survey on direct search methods for blackbox optimization and their applications. In *Mathematics Without Boundaries*, pages 31–56. Springer, 2014.

[4] A Brent Strong. *Fundamentals of Composites Manufacturing: Materials, Methods and Applications*. Society of Manufacturing Engineers, 2008.

[5] Isaac M Daniel, Ori Ishai, Issac M Daniel, and Ishai Daniel. *Engineering Mechanics of Momposite Materials*, volume 3. Oxford University Press New York, 1994.

[6] Flake C Campbell. *Structural Composite Materials*. ASM International, 2010.

[7] László P Kollár and George S Springer. *Mechanics of Composite Structures*. Cambridge University Press, 2003.

[8] Zafer Gürdal, Raphael T Haftka, and Prabhat Hajela. *Design and Optimization of Laminated Composite Materials*. John Wiley & Sons, 1999.

[9] Justin Hale. Boeing 787 from the Ground Up. Available at: `http://www.boeing.com/commercial/aeromagazine/articles/qtr_4_06/AERO_Q406_article4.pdf`. Accessed on: 2017-02-03.

[10] Larry Lessard. *Course Notes: MECH530-Mechanics of Composite Materials*. Department of Mechanical Engineering, McGill University, Montreal, 2010.

[11] Edward C Smith and Inderjit Chopra. Formulation and Evaluation of an Analytical Model for Composite Box-Beams. *Journal of the American Helicopter Society*, 36(3):23–35, 1991.

[12] Panos Y Papalambros and Douglass J Wilde. *Principles of Optimal Design: Modeling and Computation*. Cambridge University Press, 2000.

[13] Singiresu S Rao. *Engineering Optimization: Theory and Practice*. John Wiley & Sons, 2009.

[14] Hossein Ghiasi, Damiano Pasini, and Larry Lessard. Optimum stacking sequence design of composite materials Part I: Constant stiffness design. *Composite Structures*, 90(1):1–11, 2009.

[15] Hossein Ghiasi, Kazem Fayazbakhsh, Damiano Pasini, and Larry Lessard. Optimum stacking sequence design of composite materials Part II: Variable stiffness design. *Composite Structures*, 93(1):1–13, 2010.

[16] Andrew R Conn, Katya Scheinberg, and Luis N Vicente. *Introduction to Derivative-free Optimization*. SIAM, 2009.

[17] James C Spall. Stochastic Optimization. In *Handbook of Computational Statistics*, pages 173–201. Springer, 2012.

[18] G Narayana Naik, S Gopalakrishnan, and Ranjan Ganguli. Design optimization of composites using genetic algorithms and failure mechanism based failure criterion. *Composite Structures*, 83(4):354–367, 2008.

[19] FS Almeida and AM Awruch. Design optimization of composite laminated structures using genetic algorithms and finite element analysis. *Composite Structures*, 88(3):443–454, 2009.

[20] JH Park, JH Hwang, CS Lee, and W Hwang. Stacking sequence design of composite laminates for maximum strength using genetic algorithms. *Composite Structures*, 52(2):217–231, 2001.

[21] G Soremekun, Z Gürdal, RT Haftka, and LT Watson. Composite laminate design optimization by genetic algorithm with generalized elitist selection. *Computers & Structures*, 79(2):131–143, 2001.

[22] Akira Todoroki and Tetsuya Ishikawa. Design of experiments for stacking sequence optimizations with genetic algorithm using response surface approximation. *Composite Structures*, 64(3):349–357, 2004.

[23] M Abouhamze and M Shakeri. Multi-objective stacking sequence optimization of laminated cylindrical panels using a genetic algorithm and neural networks. *Composite Structures*, 81(2):253–263, 2007.

[24] I Rajendran and S Vijayarangan. Optimal design of a composite leaf spring using genetic algorithms. *Computers & Structures*, 79(11):1121–1129, 2001.

[25] Petri Kere and Juha Lento. Design optimization of laminated composite structures using distributed grid resources. *Composite Structures*, 71(3):435–438, 2005.

[26] Joseph Lynn Henderson. Laminated plate design using genetic algorithms and parallel processing. *Computing Systems in Engineering*, 5(4-6):441–453, 1994.

[27] MS Murugan, S Suresh, R Ganguli, and V Mani. Target vector optimization of composite box beam using real-coded genetic algorithm: a decomposition approach. *Structural and Multidisciplinary Optimization*, 33(2):131–146, 2007.

[28] Dhyan Jyoti Deka, G Sandeep, D Chakraborty, and A Dutta. Multiobjective optimization of laminated composites using finite element method and genetic algorithm. *Journal of Reinforced Plastics and Composites*, 24(3):273–285, 2005.

[29] Peter JM Van Laarhoven and Emile HL Aarts. Simulated annealing. In *Simulated Annealing: Theory and Applications*, pages 7–15. Springer, 1987.

[30] A Rama Mohan Rao and N Arvind. Optimal stacking sequence design of laminate composite structures using tabu embedded simulated annealing. *Structural Engineering and Mechanics*, 25(2):239–268, 2007.

[31] Mustafa Akbulut and Fazil O Sonmez. Optimum design of composite laminates for minimum thickness. *Computers & Structures*, 86(21):1974–1982, 2008.

[32] Mustafa Akbulut and Fazil O Sonmez. Design optimization of laminated composites using a new variant of simulated annealing. *Computers & Structures*, 89(17):1712–1724, 2011.

[33] R Kathiravan and Ranjan Ganguli. Strength design of composite beam using gradient and particle swarm optimization. *Composite Structures*, 81(4):471–479, 2007.

[34] SN Omkar, Rahul Khandelwal, Santhosh Yathindra, G Narayana Naik, and S Gopalakrishnan. Artificial immune system for multi-objective design optimization of composite structures. *Engineering Applications of Artificial Intelligence*, 21(8):1416–1429, 2008.

[35] SN Omkar, Dheevatsa Mudigere, G Narayana Naik, and S Gopalakrishnan. Vector evaluated particle swarm optimization (VEPSO) for multi-objective design optimization of composite structures. *Computers & Structures*, 86(1):1–14, 2008.

[36] SN Omkar, J Senthilnath, Rahul Khandelwal, G Narayana Naik, and S Gopalakrishnan. Artificial Bee Colony (ABC) for multi-objective design optimization of composite structures. *Applied Soft Computing*, 11(1):489–499, 2011.

[37] Hakan Boyaci. Master's Thesis: Stacking sequence optimization of the anti-buckled graphite/epoxy laminated composites for minimum weight using generalized pattern search algorithm. Available at: `http://openaccess.iyte.edu.tr/bitstream/handle/11147/4228/10013551.pdf?sequence=1&isAllowed=y`. Accessed on: 2016-09-03.

[38] JB Cardoso, PG Coelho, and AL Custódio. Parallel Direct Search in Structural Optimization. In *PARENG2011-The Second International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering*, 2011.

[39] Charles Audet, Vincent Béchard, and Sébastien Le Digabel. Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search. *Journal of Global Optimization*, 41(2):299–318, 2008.

[40] Andrew R Conn and Sébastien Le Digabel. Use of quadratic models with mesh-adaptive direct search for constrained black box optimization. *Optimization Methods and Software*, 28(1):139–158, 2013.

[41] Charles Audet, Kenneth Diest, Sébastien Le Digabel, Luke A Sweatlock, and Daniel E Marthaler. Metamaterial Design by Mesh Adaptive Direct Search. In *Numerical Methods for Metamaterial Design*, pages 71–96. Springer, 2013.

[42] Senthil Murugan and Ranjan Ganguli. Aeroelastic stability enhancement and vibration suppression in a composite helicopter rotor. *Journal of Aircraft*, 42(4):1013–1024, 2005.

[43] Frank H Clarke. *Optimization and nonsmooth analysis*. SIAM, 1990.

[44] C. Audet, S. Le Digabel, and C. Tribes. NOMAD user guide. Technical Report G-2009-37, Les cahiers du GERAD, 2009.

[45] S. Le Digabel. Algorithm 909: NOMAD: Nonlinear Optimization with the MADS algorithm. *ACM Transactions on Mathematical Software*, 37(4):1–15, 2011.

[46] M.A. Abramson, C. Audet, G. Couture, J.E. Dennis, Jr., S. Le Digabel, and C. Tribes. The NOMAD project. Software available at `https://www.gerad.ca/nomad/`.

[47] Mark A Abramson, Charles Audet, John E Dennis Jr, and Sébastien Le Digabel. OrthoMADS: A deterministic MADS instance with orthogonal directions. *SIAM Journal on Optimization*, 20(2):948–966, 2009.

[48] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation*, 7(2):117–132, 2003.

[49] AL Custódio, M Emmerich, and JFA Madeira. Recent developments in derivative-free multiobjective optimization. *Computational Technology Reviews*, 5:1–30, 2012.

[50] Eckart Zitzler, Joshua Knowles, and Lothar Thiele. Quality assessment of Pareto set approximations. *Multiobjective Optimization*, pages 373–404, 2008.

[51] Johannes W. Kruisselbrink. Hypervolume Computation. MathWorks file exchange.

[52] Charles Audet, Gilles Savard, and Walid Zghal. A mesh adaptive direct search algorithm for multiobjective optimization. *European Journal of Operational Research*, 204(3):545–556, 2010.