Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Kolja Westphal

Entwicklung einer Methode zur Generierung und Bewertung von geblendeten FVK-Lagenaufbauten

Fakultät Technik und Informatik
Department Fahrzeugtechnik und Flugzeugbau

Faculty of Engineering and Computer Science
Department of Automotive and Aeronautical Engineering

Kolja Westphal

Entwicklung einer Methode zur Generierung und Bewertung von geblendeten FVK-Lagenaufbauten

Masterarbeit eingereicht im Rahmen der Masterprüfung

im Studiengang M.Sc. Flugzeugbau
am Department Fahrzeugtechnik und Flugzeugbau
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

in Zusammenarbeit mit:
Deutsches Zentrum für Luft- und Raumfahrt e.V.
Institut für Systemleichtbau
Abteilung Funktionsleichtbau
[redacted]

Erstprüfer/in: Prof. Dr. Eckart Nast
Zweitprüfer/in : Dr. David Zerbst

Abgabedatum: 26.03.2024

Zusammenfassung

Name des Studierenden
Kolja Westphal

Thema der Masterthesis
Entwicklung einer Methode zur Generierung und Bewertung von geblendeten FVK-Lagenaufbauten

Stichworte
Laminate, Graphentheorie, Tiefensuche, Ameisenalgorithmus

Kurzzusammenfassung
In dieser Arbeit wird eine innovative Methode vorgestellt, mit der unabhängige Laminatabschnitte effektiv auf mögliche kontinuierliche Lagen untersucht werden können. Dieses Erzeugen von kontinuierlichen Lagen innerhalb einer Laminatstruktur wird im Allgemeinen als Blending bezeichnet. Im Gegensatz zu konventionellen Ansätzen, bei denen bereits bei der Rücktransformation von Laminatparametern in diskrete Lagenaufbauten Anforderungen bezüglich kontinuierlicher Lagen definiert werden, führt diese Methode das Blending als Nachbearbeitungsschritt durch. Das Schlüsselkonzept besteht darin, eine Blendingmatrix zu erstellen, die Positionsinformationen über potenzielle kontinuierliche Lagen zwischen den einzelnen Abschnitten erfasst, welche dann in ein Graphobjekt umgewandelt wird, um verschiedene Blendingmöglichkeiten mit einem Suchalgorithmus zu finden. Anhand mehrerer Beispiele zeigt der Autor, dass die Methode in der Lage ist, alle möglichen Blendinglösungen zwischen zwei symmetrischen Laminatabschnitten mit fünfzig individuellen Lagen in weniger als einer Minute zu finden. Zusammen mit einem neuen Blendingkriterium wird die Methode in einen Optimierungsprozess integriert, um die Fähigkeit zu demonstrieren, geblendete Laminatabschnitte aus strukturell optimierten Laminatparametern zu finden.

Name of Student
Kolja Westphal

Title of the paper
A new laminate blending recovery method for the design of composite structures

Keywords
Laminates, Blending, Depth First Search, Ant Colony Optimization

Abstract
This thesis introduces an innovative laminate blending recovery method for effectively blending independent laminate sections. Unlike conventional approaches, which involve defining blending constraints during the retransformation of lamination parameters into discrete stacking sequences, this method performs blending as a post-processing step. The key concept involves creating a blending matrix to capture positional information about potential continuous plies between stackings, which is then transformed into a graph object to explore diverse blending possibilities with an established search algorithm. Through multiple examples, the author shows the method’s capability of finding all possible blending solutions between two symmetric laminate sections with fifty individual plies in under one minute. Together with a new blending criterion, the method is integrated into an optimization process to showcase the ability to find blended laminate sections from structurally optimized lamination parameters.

Contents

List of Figures

List of Tables

Acronyms

ACO Ant Colony Optimization.

BFS Breadth-First-Search.

BPR Blended Plies Ratio.

CFRP Carbon Fiber-Reinforced Polymers.

CLT Classical Laminate Theory.

DFS Depth-First-Search.

FSDT First-Order-Shear-Deformation-Theory.

LP Lamination Parameter.

MAE Mean Absolute Error.

MOO Multi-Objective Optimization.

RMSE Root Mean Square Error.

SAF Sustainable Aviation Fuel.

1 Introduction

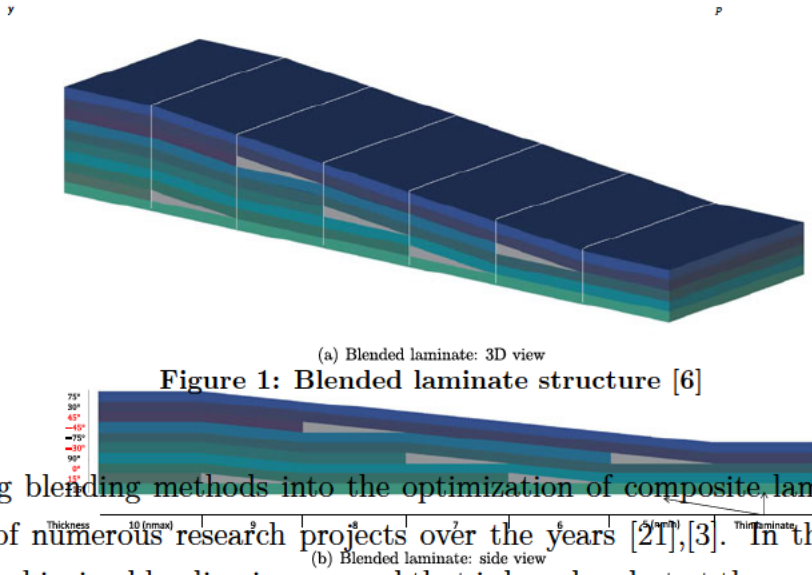
Climate change has become one of the most pressing challenges of our time. The aviation industry, a key player in global transportation, has been identified as a major contributor to greenhouse gas emissions [20]. The dire consequences of climate change necessitate a paradigm shift in the way we conceptualize and engineer aircraft, pushing researchers and engineers to explore sustainable alternatives and environmentally friendly solutions.

Recognizing the significant environmental impact of airplanes, researchers are actively exploring new configurations and propulsion systems to minimize ecological consequences. This quest for more sustainable air travel extends to novel aircraft designs, including electric and hybrid-electric propulsion systems, aimed at reducing emissions and enhancing operational sustainability. Additionally, the aviation industry is embracing sustainable aviation fuels (SAFs), derived from renewable sources. Integrating SAFs into aircraft operations aligns with the industry's commitment to mitigating its carbon footprint, fostering a greener future for air travel.

Simultaneously, a crucial aspect of this transformative journey involves structural optimization to achieve lighter aircraft components. Recognizing that every kilogram matters in the context of fuel efficiency, engineers are focusing on innovative ways to design and fabricate structurally sound yet lightweight components. The objective is to strike an optimal balance between structural integrity and weight reduction, ensuring that the aircraft remains robust while minimizing its overall mass.

The evolution of aircraft design has seen a significant shift towards the use of advanced materials, particularly carbon fibre-reinforced polymers (CFRP). CFRPs are a material structure consisting of layers of fibres, embedded in a matrix. They offer a high strength-to-weight ratio, making them lightweight yet strong. They provide excellent corrosion resistance and fatigue resistance, which makes them ideal for applications in aerospace, automotive, and other industries where durability and performance are crucial [33]. As recently as the 1990s, the proportion of carbon fibre-reinforced plastics in passenger aircraft was only ten per cent. Today's aircraft types such as the A350 XWB and Boeing Dreamliner 787 already consist of more than 50 per cent carbon fibre-reinforced plastics [4]. The composites are mainly used in the aircraft's fuselage and wings but are now also tested to be incorporated in new aircraft turbines such as the Rolls-Royce UltraFan [30]. The extensive incorporation of CFRPs marks a significant transformation in the aviation industry, underscoring the crucial role of lightweight materials in attaining improved fuel efficiency and overall sustainability.

In optimizing these composite laminates, it is vital to design laminates capable of effectively carrying loads in different directions through the fibres. This is achieved by enforcing stacking sequence continuity across the multiple panels comprising the structure. This continuity of plies is often referred to as blending, a term first introduced in a publication of Kristinsdottir et al. [8]. Ensuring a seamless transition of fibre across the panels, stress concentrations and potential failure points can be reduced. An example of a blended laminate structure is given in figure 1.



(a) Blended laminate: 3D view

Figure 1: Blended laminate structure [6]

Incorporating blending methods into the optimization of composite laminates has been the subject of numerous research projects over the years [21],[3]. In this thesis, a new approach to achieving blending is proposed that is based on but at the same time contrasts with some aspects of previous methods presented in the literature review. The approach is capable of producing multiple blending solutions for one structure. A criterion for evaluating these solutions is additionally proposed.

The thesis begins with reviewing relevant literature to present the current state of technology and from there to derive the motivation for the novel approach. A detailed explanation of the blending method and its limits together with the blending criterion is given afterwards. It follows a discussion of the results of self-created blending and benchmark optimization problems with a conclusion of the work and an outlook on future activities.

2 Literature Survey

The presented literature survey includes publications with the state of the art of current implementations on how to achieve blending between several laminate sections. Reviewing these publications also serves to identify specific criteria made by the authors to define blending. A collection of these criteria can be used to assess the proposed method of evaluating blended laminates.

In 2001, Kristinsdottir et al. [8] introduced the term *Blending* in the context of designing manufacturable laminate structures where loads vary across the structure. In the paper, a method was developed using a *ply-add-and-drop* technique [21].
In the same year, Liu and Haftka [10] published a paper on establishing continuity between two adjacent laminates by applying continuity constraints in the design process. They proposed two continuity measures namely the *composition continuity measure* and the *stacking sequence continuity measure*. The first measure examines two adjacent laminates only by the appearance of common layers in relation to the total thickness of one laminate. The second one examines the laminates further by counting possible continuous plies and setting the overall thickness of these possible plies in relation to the thickness of one laminate.
After that, the meaning of blending, designing manufacturable laminate structures, remained the same but scientists incorporated different criteria on how to accomplish these design tasks and pushed the boundaries for perfecting the laminates.
Liu and Haftka and later Soremekun et al. [25] implemented a genetic algorithm in their optimization process, called *DARWIN*, to retrieve blended laminates. Genetic algorithms are part of the evolutionary algorithms and are based on selection and genetics. In theory, an artificial population undergoes an evolutionary process in which the individuals are evaluated based on their fitness for a certain criterion. Through mutation and crossover, descendants are produced and integrated back into the main population [15],[23]. This algorithm performs reasonably well for this kind of solution finding as the approach focuses mainly on the discrete combinatorial nature of blending [21].
One drawback of designing fully blended laminate structures with these genetic algorithms was the high computational cost [3]. Adams et al. [1] and Seresta et al. [24] adapted this method and implemented additionally a guiding stack from which all laminates in a structure are obtained by deleting a contiguous series of outermost or innermost plies [3]. The strategy was then known as *Inner and Outward Blending* (figure 2). Using a guide for the stackings further simplifies the blending problem and reduces the computational cost as it greatly reduces the design space [3].

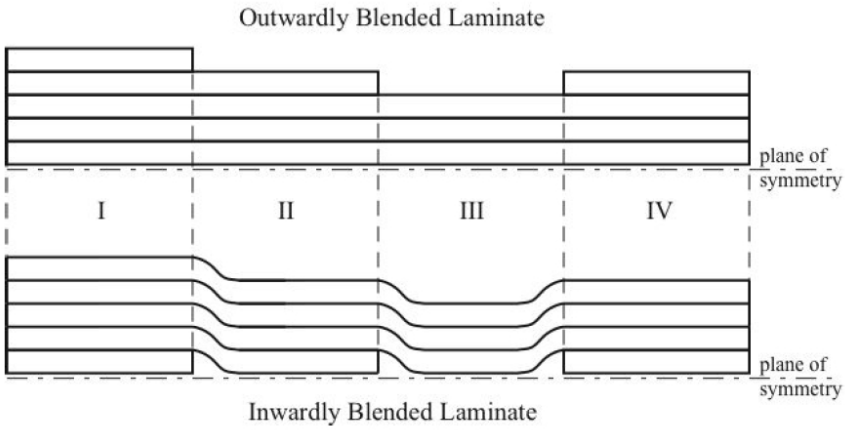


Figure 2: Inwardly and Outwardly Blended Laminate [3]

Optimizing to a fully blended structure comes with one disadvantage, which is a possible increase in the overall weight due to the inflexibility of the layup for each section [3]. As a guide always represents a possible layup for every section in the laminate structure, it is obvious that each section won't reach its optimum with regards to the number of layers and stiffness value, thus increasing the number of layers to achieve the minimum stiffness is needed.

By selecting multiple guides for the structure, this inflexibility can be reduced. Zeng et al. [35] proposed a *multiple-template-blending method*, where a *template* refers to the previous explained guide. The difference between a single-template-blending and a multiple-template-blending is displayed in figure 3.

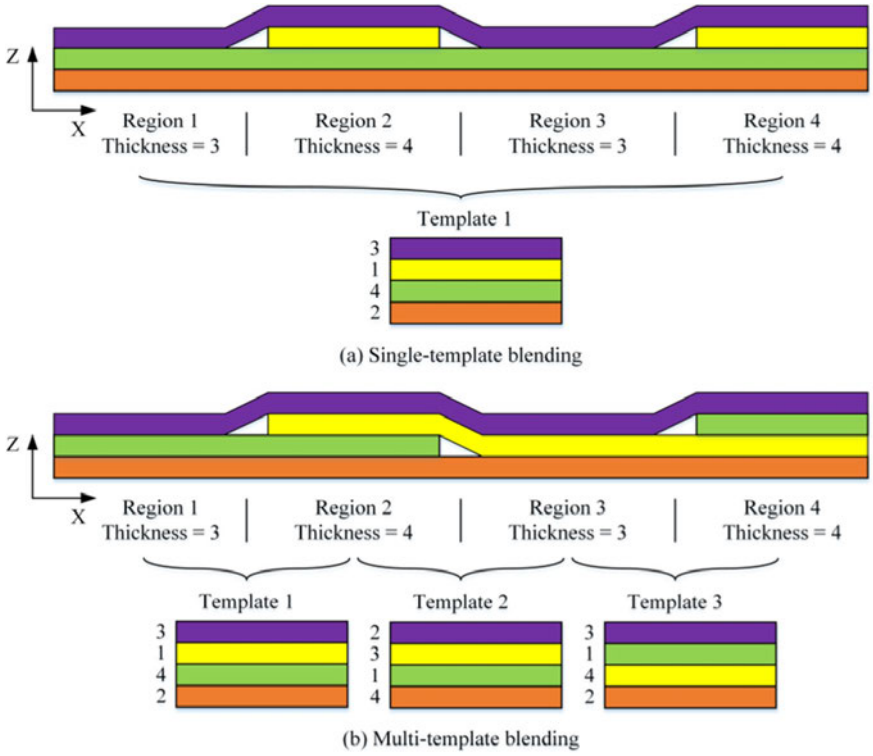


Figure 3: Multiple-template-blending [35]

Up to 2016, the most commonly used algorithm to solve the blending problem was the genetic algorithm as a representative for metaheuristic algorithms. In 2016, Macquart et al. [13] published a paper that featured a method for a gradient-based optimization using lamination parameters. In the classical laminate theory (CLT), the laminate stiffness is represented with the ABD-Matrix as the global stiffness of the laminate, which holds the extensional stiffness matrix A, the bending stiffness matrix D and the extension-bending matrix B. The matrix entries of the ABD-matrix are built upon the reduced stiffnesses of each single ply that are transformed from a local to a global coordinate system. The formulation of the first entry in the extensional matrix is given in equation 2 (assuming all plies have the same material and thicknesses). For each layer k, information about the ply angle and its thickness is combined with the material invariants U. The material invariants are calculated from the single, non-transformed material stiffnesses Q (equation 1).

$$
\begin{aligned}
U_1 &= \frac{1}{8}(3Q_{11} + 3Q_{22} + 2Q_{12} + 4Q_{66}) \\
U_2 &= \frac{1}{8}(Q_{11} + Q_{22}) \\
U_3 &= \frac{1}{8}(Q_{11} + Q_{22} + 2Q_{12} + 4Q_{66})
\end{aligned}
\tag{1}
$$

$$A_{11} = U_1 \sum_{k=1}^{N} (z_k - z_{k-1}) + U_2 \sum_{k=1}^{N} cos(2\theta_k)(z_k - z_{k-1}) + U_3 \sum_{k=1}^{N} cos(4\theta_k)(z_k - z_{k-1}) \quad (2)$$

A_{11} can also be defined with the lamination parameters V, which is presented in following equation.

$$A_{11} = U_1 T + U_2 V_1 + U_3 V_2 \quad (3)$$

This formulation smears the ply angle and the thickness for each individual ply to a laminate that is defined by its total thickness, the material invariants and the lamination parameters. The full relation between the ABD-Matrix entries and the lamination parameters can be found in the appendix C.1. The initially non-linear relationship between laminate stiffness and a discrete stacking sequence becomes linear when replacing the trigonometric functions with the lamination parameters so that the lamination parameters can be used as continuous and dimensionless design variables within a gradient-based optimization process [31]. This optimization process offers a less computationally costly approach than the heuristic and metaheuristic search algorithms. Until the publication of the paper from Macquart et al., lamination parameters had already been used in multi-step optimizations that split the blending problem into a continuous and discrete optimization step. However, there is a big design space discrepancy between the continuous and discrete optimization step, which results in a relevant performance loss [13]. Applying blending constraints during the continuous optimization instead, the method in the published paper aimed to reduce the performance loss observed between optimization levels. The blending constraints quantify the change in lamination parameters from one panel to another due to ply drops. The constraint formulation guarantees that no blended solution exists in case the constraint is not fulfilled.
A more recent research from Scardaoni et al. [21] and Panettieri et al. [17] focuses on the application of polar parameters instead of laminate parameters to define the blending constraints for an optimization process. In this approach, polar parameters are derived from the *First-order-shear-deformation-theory* (FSDT) that formulates the stiffness tensor of a laminate in the form of

$$K_{lam} = \begin{bmatrix} A & B & O \\ & D & O \\ sym & & H \end{bmatrix} \quad (4)$$

where A is the membrane stiffness tensor of the laminate, D is the bending stiffness tensor,

H is the out-of-plane shear stiffness tensor, and B is the membrane/bending coupling stiffness tensor. The FSDT is a generalisation of the CLT. It includes the transverse shear deformation for a plate.

While both laminate parameters and polar parameters highlight some physical aspects of the laminate that cannot be easily caught when using the Cartesian representation, the laminate parameters have one drawback over the polar parameter. Laminate parameters are not tensor invariants and do not have a simple and immediate physical meaning. Conversely, polar parameters are true tensor invariants and have an immediate physical meaning, which is linked to the different (elastic) symmetries of the stiffness tensors of the laminate [17]. It has been shown that the use of polar parameters eliminates redundant mechanical properties, and eases expressing changes in reference frames which leads to the design of laminates with enhanced mechanical responses beyond traditional approaches reliant on symmetric, balanced stacks and limited layer orientations [14].

3 Research gap

The following section summarizes some of the most relevant points from the literature review to then introduce the novelty approach of this work from which the objectives, scope and research question can be derived.

From reviewing the literature about blending implementations, it can be identified that for structural optimizations, a gradient-based approach for top-level optimizations is favoured due to the higher efficiency compared to non-gradient-based algorithms. But as stated, these optimization methods need continuous design variables which enforces a transformation of the stiffness description with the ABD matrix into laminate parameters. Lamination parameters can establish a non-linear relationship between the laminate stiffness and its discrete stacking that allows this kind of continuity.
Using them in the optimization, however, comes with the expense of information loss about the real stacking sequence of the laminate. It is therefore necessary to perform a retransformation of the lamination parameters back to a discrete stacking sequence. This retransformation is an ongoing challenge and researchers tested various methods that include genetic algorithms and particle swarm optimizations [27],[2].
While the blending constraints in the top-level optimization shall ensure that retransformed laminate stackings are blendable to a unified structure, a methodology needs to be devised to guarantee that the laminates, once retransformed, also demonstrate a certain blending with continuous plies.
Guide-based approaches offer the advantage that perfect blending is automatically generated when such a method is implemented. A perfect blending is reached if the structure is constructed with the least possible amount of plies. This feature is advantageous for transferring the loads acting on the structure between the individual sections. The resulting inflexibility for individual stacking sequences can be reduced with the multiple-template-blending approach from Zeng et al. [35], but the fundamental basis that layers must be taken from a guide remains.
To further reduce the inflexibility and to better match the stiffness requirements for the individual sections to avoid adding plies and therefore weight in a post-processing step, it would be beneficial to separate the retransformation of the lamination parameters and the blending procedure. Noval layup retrieval algorithms, like the one from Sprengholz et al. [27], are able to minimize the discrepancy between desired and retrieved stiffness values from laminates to a minimum. The hypothetical structure in figure 4 shall illustrate the shift from a perfectly blended structure to a non-perfectly blended one where the stacking sequence for sections one and four is modified in favour of their desired stiff-

ness values. However, it must be ensured that the structural integrity of the structure is maintained. This means that in the transition areas between the sections, enough layers must be continued to allow the transfer of loads and to avoid peak stresses.

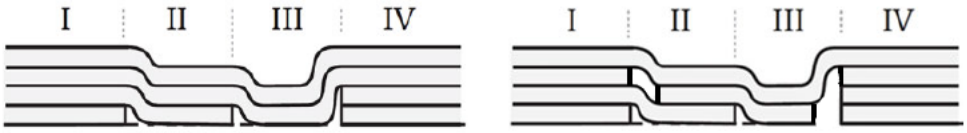


Figure 4: Perfect (left) and non-perfect (right) blending [13]

So the question arises of how to blend individual retransformed laminate sections. Scardaoni et al. [21] stated in their publication that blending has an intrinsic combinatorial nature, which makes computational costs rise rapidly and the solution search difficult, even with the most advanced numerical methods. This thesis aims to answer therefore the following research question

> Can individual laminate sections be blended independently of their creation from laminate parameters?

The objective is to develop a generic blending recovery method that can be flexibly used in stacking sequence retrieval problems. This is solved within four main parts comprising:

1. A representation of the layup data of adjacent laminate sections that can be used for
2. A search algorithm to find all possible solutions of ply continuity,
3. A criterion for the assessment of the degree of fulfilled blending within a composite structure and
4. A analysis of a benchmark optimization problem where the blending method and blending criterion can be applied within a stacking sequence retrieval process

4 Blending method

This section explains the main idea of the new implementation of blending, beginning with a little introduction dealing with the motivation for it. To better understand the method, a blending problem between only two adjacent sections is discussed.

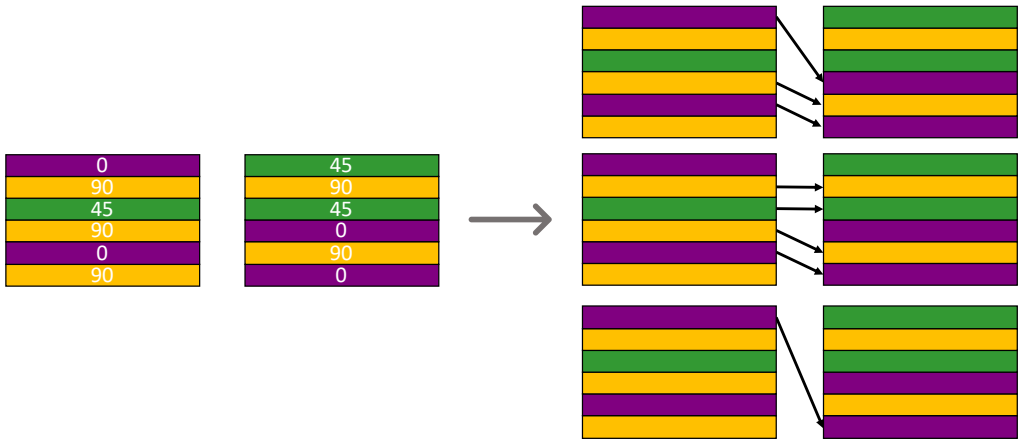


Figure 5: Blending two laminate sections

4.1 Combinatorial problem

A straightforward approach for blending the two layups in figure 5 could be established with combinatorial methods and filtering. For that, every possible combination of plies between the left and the right layup could be identified. Through permutations, these combinations could be varied in their order of appearance, which would create different ways of blending the two layups. However, by simply permutating different combinations, non-realisable blended sections would be created due to the crossing of plies or selecting plies twice for the same continuous ply. A filter process must therefore be performed after the permutation to filter out non-realisable blended sections.
Achieving blended sections with this method, however, has two main drawbacks. One, the permutative nature of the method causes a high number of solutions. Considering the problem in figure 5, one must find for each ply in the left layup, possible plies in the right layup, then select one of these plies and finally perform the permutations and filter the results. The resulting number of solutions, that must be filtered, can be calculated with

equation 5, where b is the number of solutions, n is the number of plies in the left layup and m_i is the number of possible in the right layup for each ply in the left layup.

$$b = n! \prod_{i=1}^{n} m_i \tag{5}$$

And secondly, only a small number of the solutions can actually be realised in a manufacturing process. For the above-mentioned blending problem, 720 solutions must be filtered to only 10 realisable solutions. For small laminate sizes, the found solutions are manageable to filter. Considering, however, a laminate with 12 plies, the factorial part of the equation already extends the number of solutions to 479001600.

4.2 Representing blending solutions in matrix

The new method shall avoid the two problems stated in the previous sections by establishing a representation of all possible continuous plies between two adjacent layups in the form of a matrix. Figure 6 represents the blending problem from figure 5.

	45	90	45	0	90	0
0	0	0	0	**1**	0	**1**
90	0	**1**	0	0	**1**	0
45	**1**	0	**1**	0	0	0
90	0	**1**	0	0	**1**	0
0	0	0	0	**1**	0	**1**
90	0	**1**	0	0	**1**	0

Figure 6: Blending representation

In figure 6, the numbers to the left of the matrix are the ply angles from the left layup from figure 5 from the top of the layup to the bottom. The numbers on top of the matrix from left to right equal the ply angles from the right layup. The matrix itself now displays every possible combination of plies between the left and the right layup, which is indicated by a one in the field. For example, the matrix element [1, 4] (zero-based numbering) indicates that the second ply from the left layup can be combined with the fifth ply from the right layup resulting in a continuous 90 degree ply from left to right.

4.2.1 The counting method

To extract various blending solutions from the matrix, the matrix must be traversed by selecting several 'ones' on the way. Figure 7 shows several paths through the matrix.

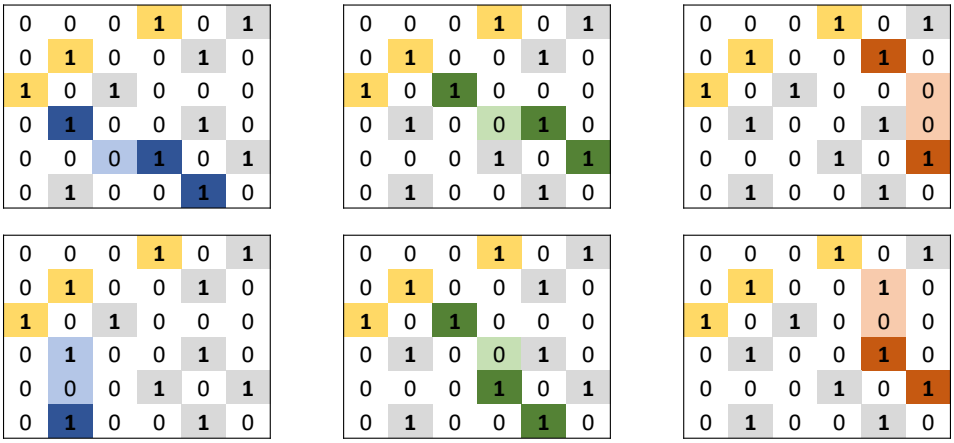


Figure 7: Different blending solutions

Fields in yellow represent possible starting points for traversing the matrix. The dark colour fields indicate that a combination is selected for the overall blending solution between the two layups. When selecting one of these fields, the search for other combinations must continue from the next row and the next column. In indices terms, that means: When finding a combination at $[i, j]$, the search continues from $[i+1, j+1]$. In principle, it would be allowed to then select one of all other combinations which fulfil $row >= (i+1)$ and $column >= (j+1)$. Though, it would be wise to select ones near $[i+1, j+1]$, as this increases the chances of selecting more combinations on the pass-through. Selecting combinations at i or j is not allowed, as this would mean that either the ply from the left layup or the ply from the right layup is used twice for a combination.
Figure 8 displays what each path would symbolize as a real blended solution with layups.

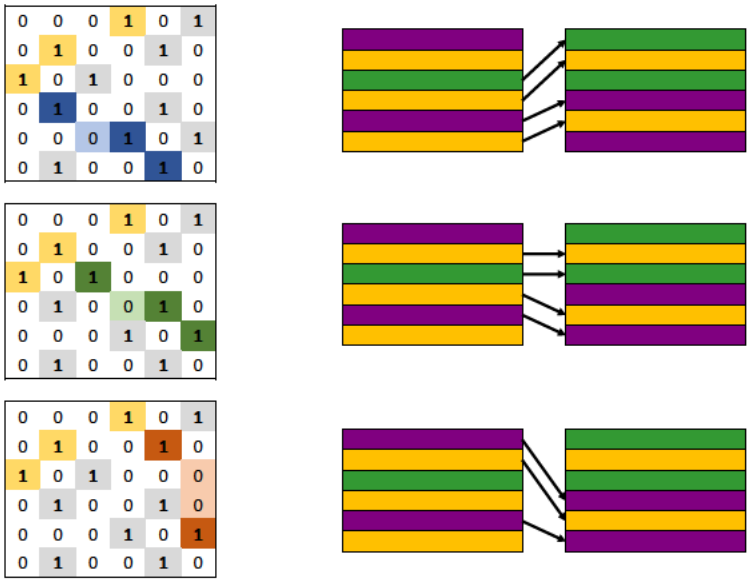


Figure 8: Different blending solutions with layups

Representing blending solutions in the form of this matrix offers combined information about the possible ply combination together with the position of the plies in the layup such that a possible blending solution does not need be investigated if it contains combinations that would lead to the crossing of plies. Every pass-through results in an admissible blending solution between the two layups.

4.2.2 Creating the blending matrix

The idea for the blending matrix derived from a method developed in Werthen and Dähne [31]. In the work, a method has been created that allows for an analysis of different layup combinations regarding the requirement that all plies of a thinner laminate exist in a thicker laminate. Sparse matrices were used to represent stackings (layups) for one specific section.

```
1       stackings = [[  0,45,90],
2                    [45,90,45],
3                    [ 0,  0,45]]
4
5                                 ____0_____45_____90____
6       sparseStackings = [[1,0,0,|0,1,0,|0,0,1],
7                          [0,0,0,|1,0,1,|0,1,0],
8                          [1,1,0,|0,0,1,|0,0,0],
```

Sparse matrices are characterized by having mostly zero-value elements. Substantial memory requirement reductions can be also realized by storing only the non-zero entries.
To convert multiple stackings to sparse stackings, all existent entries in the stackings must be identified. In case, a group of stackings differs in length, the longest array (layup with the most amount of plies) must also be identified. The shape of the sparse matrix is set to $n \times m$ where n is equal to the number of stackings and m is equal to the length of the longest array multiplied by the number of existent entries in the stackings. The sparse matrix can be visually grouped into several sections each representing the stackings where only one specific ply angle is marked. Essentially, the stackings are hereby divided into their constituent parts.
Now, it is possible to perform a matrix multiplication with two sparse vectors, taken from the sparse matrix, to find possible combinations of plies between two layups. In figure 9, a matrix multiplication with the sparse vectors that represent the layups from figure 5 is shown.

		0						45						90					
		0	0	0	1	0	1	1	0	1	0	0	0	0	1	0	0	1	0
	1	0	0	0	1	0	1	1	0	1	0	0	0	0	1	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	1	0	1	1	0	1	0	0	0	0	1	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	1	0	0	0	1	0	1	1	0	1	0	0	0	0	1	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	1	0	1	1	0	1	0	0	0	0	1	0	0	1	0
90	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	1	0	1	1	0	1	0	0	0	0	1	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	1	0	1	1	0	1	0	0	0	0	1	0	0	1	0

Figure 9: Multiplication of two sparse vectors

In each coloured section, a non-zero entry indicates that a ply from the left section can be continued to the right. The sub-row and sub-column show at which position the individual plies are located. To create the blending matrix from figure 6, the coloured submatrices

are summed up.

4.3 Finding all possible solutions

For the proposed method, it is essential to be able to find all possible ways of continuing plies from the left layup to the right. It was discovered that the blending matrix can be converted into a structural graph. In computer science, graphs are often used for algorithmic problems when searching for optimal paths in data structures. There are well-researched methods that can traverse through graphs very efficiently.

4.3.1 The graph representaion

In graph theory, a graph is an abstract structure that represents a set of objects together with the connections existing between these objects. The mathematical abstractions of the objects are called nodes or vertices of the graph. The pairwise connections between nodes are called edges. The edges can be directed or undirected. Often graphs are drawn descriptively by representing the nodes by points and the edges by lines. One classic example of a graph is the network map of the metro.
Graphs are generally categorized by their connections between the vertices. Figure 10 shows the different kinds of graph types.

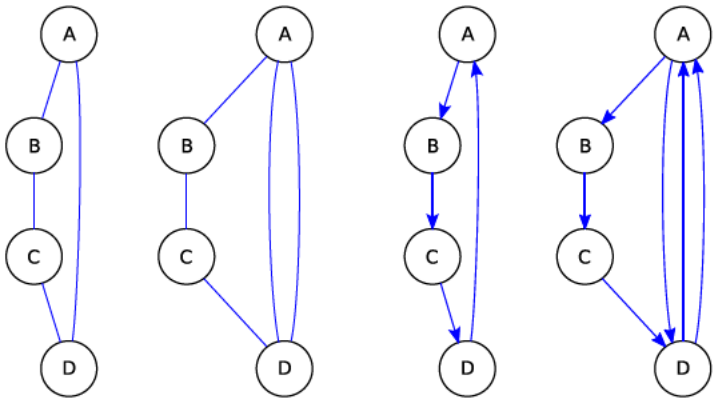


Figure 10: Types of graphs (from left to right: undirected graph, undirected multigraph, directed graph, directed multigraph) [32]

Undirected graphs represent networks where the network can be traversed without any restriction of the direction between the vertices. In the case of the outer left graph in figure 10, the edge between vertex A and B can be traversed in either way. Directional

graphs, however, restrict the traversing. There is only a connection from A to B but not vice versa. Multigraphs come in handy when weights are assigned to the edges of a graph. Weights are especially useful when there is more than one connection between the vertices but the connections differ. An example of weighted graphs could be networks representing the highway between cities. In some cases, there is more than one way to reach certain cities from other cities. The weights can then represent the length of the routes. The network map of a metro, however, is typically an undirected graph. Metros drive normally in two directions and the distances between the metro stations are the same, which makes assigning weights to the edges redundant.
A classic way of representing graphs for further use is through an adjacency list. An adjacency list collects for every node its possible neighbours. Depending on the connection, nodes can appear twice as a neighbouring node, indicating that there is more than one connection. For the undirected multigraph in figure 10, the adjacency list can be seen in the following listing. In this case, the 'list' is actually a Python dictionary.

Listing 1: Adjacency list

```
1       adjacency_list =
2                               {
3                                   A: [B, D, D],
4                                   B: [A, C],
5                                   C: [B, D],
6                                   D: [C, A, A]
7                               }
```

For directed graphs, only the nodes that can be reached from the respective node are listed in the list for the respective node.
The task at hand is now to convert the blending matrix into a graph. As discussed in section 4.2.1, the goal is to traverse the matrix by selecting non-zero entries on the pass-through. By defining the non-zero entries from the matrix as nodes in the graph, one can instead search through the graph to find a solution for the blending of two layups by travelling from node to node and storing the path on the way.
When traversing through the blending matrix, it is mandatory that when selecting a combination, the search must continue in the second row and second column. This means that for every node, i.e. every non-zero entry in the matrix, the neighbours for a node could be all non-zero entries which appear after the ith row and jth column. This, however, would result in unreasonable complex graphs because the graphs would contain paths that would skip certain combinations. The problem is illustrated in figure 11. All theoretical neighbouring nodes for the node (2,0) (coloured yellow) are coloured green and red. It

would be possible to select the node (5,4) as a neighbour for (2,0). But this would generate a structure in the graph which is displayed on the right side of the figure. Node (5,4) could also be reached by passing two other nodes on the way.

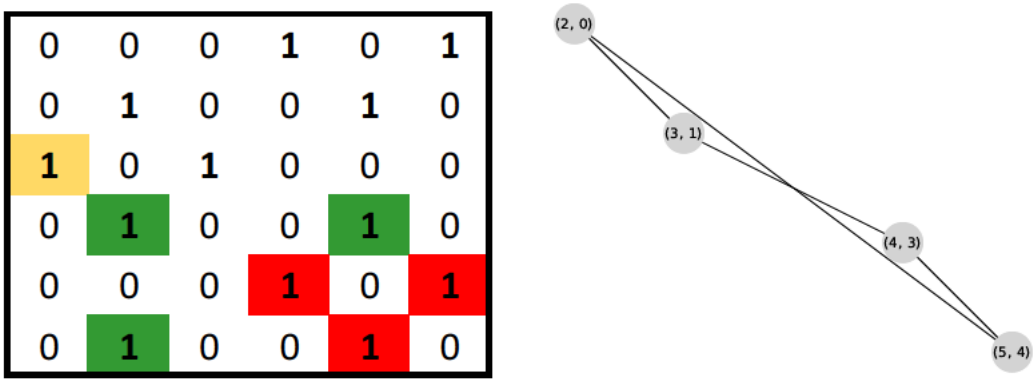


Figure 11: Select neighbouring nodes

Now, it is important to remember that the nodes represent possible continuous plies from the layups shown in 5. Skipping nodes would be the same as neglecting possible continuous plies in a solution to blend the two layups. In order to avoid unuseful edges, only the nodes marked in green are selected as neighbouring nodes for node (2,0). As the red-marked nodes can be neighbours of the green-marked nodes, they don't need to be listed as neighbours for the yellow-marked node.
The complete graph for the blending matrix from section 4.2 is displayed in figure 12. For reference, the yellow-marked non-zero entries from figure 7 are also marked yellow in the graph. It can be seen that the graph is a directed graph which is caused by the traversing rule.

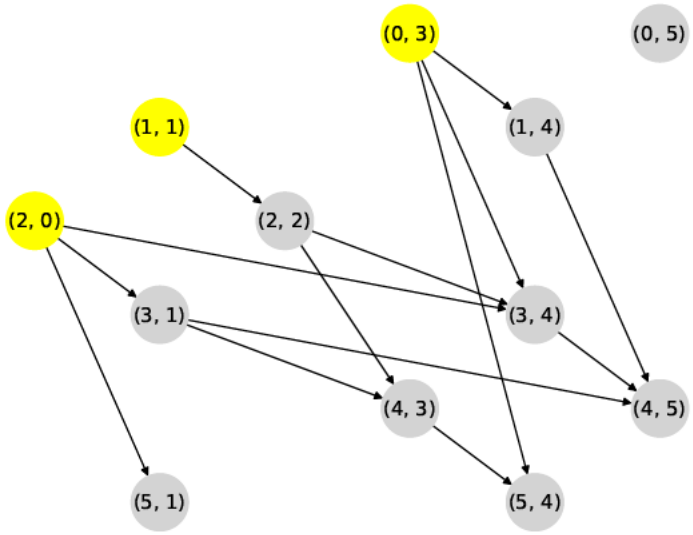


Figure 12: Graph for the blending matrix

4.3.2 The search algorithm

To find now multiple blending solutions in that graph, a graph traversal algorithm must be applied. These algorithms are essential tools in computer science for exploring the structure of graphs. Two of the more common ones are the *depth-first-search* (DFS) and the *breadth-first-search* (BFS). Both belong to the uninformed search algorithms which are characterised by avoiding any heuristics for finding a solution. Given a source and a target, the algorithms explore the graph, each in their own way.
In graph theory, nodes are often further described by the position inside the graph relative to a source or root node, known as the *level*. The level indicates, how far the node is from the source node. In figure 13, the nodes are positioned horizontally to indicate the different levels to the source node 0. It can be seen that the BFS and the DFS differ in reaching certain levels of the graph timewise.
The BFS starts from the source and investigates first every neighbour that the source has. Then, the search continues at the first neighbour by analyzing its neighbours. After that, the search returns to the second neighbour from the source and continues there. In other words, the BFS analyzes the nodes level by level.
On the other hand, the DFS tries to explore the depth of the graph by continuously reaching deeper into the graph until there are no neighbours for a certain node. The search then backtracks to the last node with an unvisited neighbour and continues to explore the graph from there.

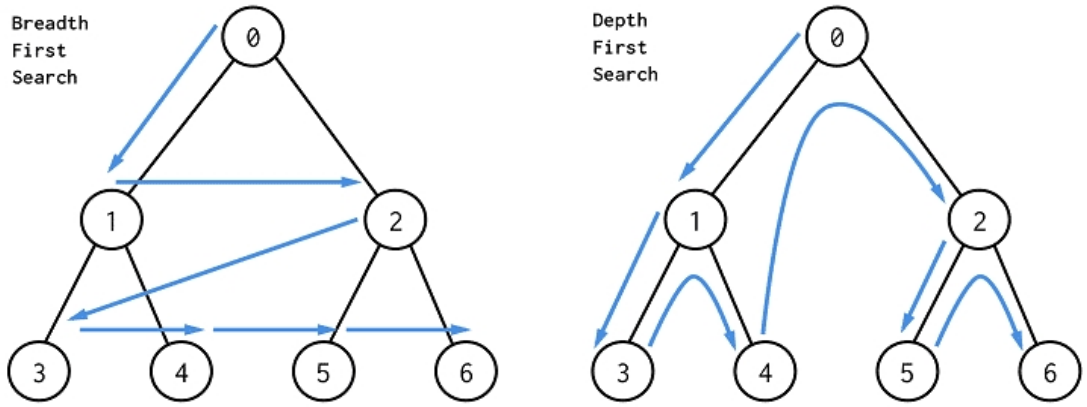


Figure 13: BFS and DFS [18]

In general, both search algorithms need the same amount of time to explore the graph, which is referred to as the time complexity of the search algorithm. Though, the BFS can be advantageous when the target node for a search is located closer to the source node, whereas the DFS is expected to find target notes faster if they lie deeper in the graph.

It must be also mentioned that the space complexity (required space on memory) for a BFS is greater due to its implementation. In most cases, the BFS uses a variable called *queue*. The queue determines which node shall be visited next after the current node. When reaching a new unvisited node, the queue is append by the neighbours of that node to the end of the queue. But the next node is always the first node in the queue. This ensures that each level is analyzed after each other. Storing the queue, however, is the drawback of the algorithm as complex graphs cause a long queue.
The DFS is often implemented as a recursive function. A recursive function calls itself inside its definition. A recursion happens every time, the search dives on level deeper into the graph. In contrast to the BFS, the DFS doesn't need a queue to proceed with the search. Below, a Python code snippet for a DFS function is given. The function searches for every path from a source to a destination.

Listing 2: DFS in Python

```
1 def DFS(adjacency_list, source, destination, path):
2     if (source == destination):
3         allpaths.append(copy.deepcopy(path))
4     else:
5         for adjacent_node in adjacency_list[source]:
6             path.append(adjacency_node)
7             DFS(adjacency_list, adjacency_node, destination
                  , path)
8             path.pop()
```

In the for-loop, which loops over every adjacent node (neighbours) of the current node, the DFS function is called within itself.

To search the graph created from the blending matrix, the DFS method is chosen. The smaller space complexity is important when analyzing big graphs. It is further advantageous that the method traverses the graph in a vertical direction which simplifies storing the different blending solutions. In the code snippet (listing 2), it can be seen that the presented DFS always holds a record of the current path. So when a valid blending solution is found, the current path, which represents this blending solution, can be easily stored. For example in figure 13, the DFS can find the path $[0, 1, 3]$. After that, the last entry is removed from the path and the search continues at node 1 to find node 4. Node 4 is appended to the path and a second solution path is found. The nodes 0 and 1 remain permanently in the list during this time.

Both methods need source- and destination nodes. Finding source nodes is comparable

to the *Selecting-neighbouring-nodes* problem. Source nodes are technically all nodes that directional wise are not adjacent to other nodes, i.e. only have adjacent nodes. Referring back to the base example from figure 5 and 12, the yellow mark nodes are source nodes and technically also the node $(0, 5)$. While it would be in theory correct to select these nodes as source nodes, this would also mean that the search method must be repeatedly initiated for every source. To initiate the search only once, an additional *master source node* is added above the source nodes (figure 14).

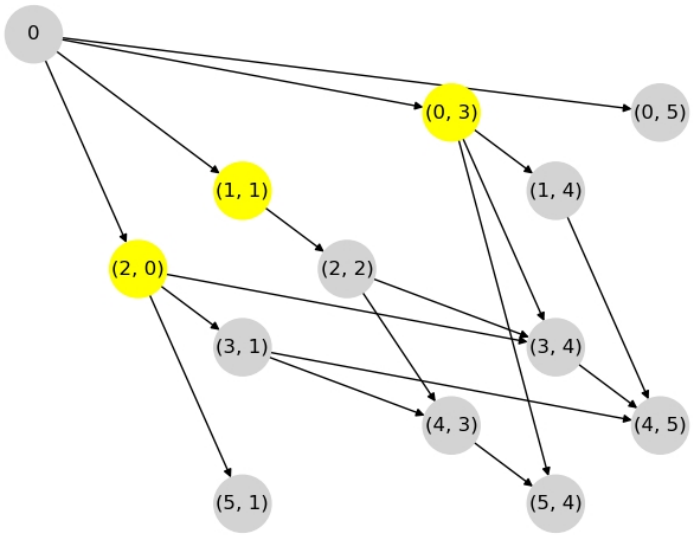


Figure 14: Graph with master source node

As the master source node is not part of the solution, it is not appended to the path variable.
The destination (target) nodes are all nodes which don't have adjacent nodes. These nodes can be easily identified from the adjacency list.

4.4 Directing the search for continuous plies

While the search method for finding continuous plies is able to find all possible blending solutions between two adjacent sections, there are applications where finding all solutions is not required and it is sufficient to find only blending solutions with the most number of continuous plies. Depending on the layups, layup combinations with up to 25 plies can generate around 3 million blending solutions and adding only a small amount of plies can easily double the number of solutions. When searching for continuous plies within a structure, the method therefore should prioritize blending solutions between sections with more continuous plies, gradually transitioning to those with fewer continuous plies.

Translating this to the graph object representing blending solutions, longer paths shall be found first and shorter paths later on in the search.
In general, the DFS code in listing 2 can not be modified to reliably find the paths in the order of their length. However, it is still possible to influence the search. In line 5 (listing 2), the for-loop loops over every adjacent node of the current node. By sorting the adjacent nodes before looping over them, one can choose, which adjacent node can be selected first. As already mentioned in section 4.2.1, when selecting a combination in the blending matrix, the next combination should be close to the current one, as this increases the chances of selecting more combinations in one path. By calculating the distance from every combination to every possible other combination, one can sort these combinations based on their relative distance. For the proposed method, the sorting part is integrated into the build-up of the adjacency list for the graph of the blending matrix. First, all possible adjacent nodes are identified and then sorted based on their column and row position in the blending matrix.

The distance is calculated with

$$distance = i_{current\_combination} - i_{possible\_next\_combination} + j_{current\_combination} - j_{possible\_next\_combination} \quad (6)$$

where i is the column index and r is the row index.
Figure 15, with figure 10 as the representative graph, displays for the base example the distance to the combination [2,0] with different shades of grey. A darker shade of grey resembles a greater distance to the combination.
This example demonstrates very well that the sorting method only increases the chance of finding longer paths first but does not guarantee it. Based on the grey scale, the sorting method would set the adjacency list for the graph node (2,0) with

```
1       adjacent_nodes = [(3, 1), (5, 1), (3, 4)]
```

Node (3,1) is clearly the best node to be selected next because the path can then be further append with the nodes (4,3) and (5, 4). Specifically for this example, however, it would be better if the DFS continues its path with the node (3,4) from the adjacency list. Selecting node (3, 4) allows additional selection of node (4, 5) whereas selecting node (5, 1) would end the search for a path with one selection less.

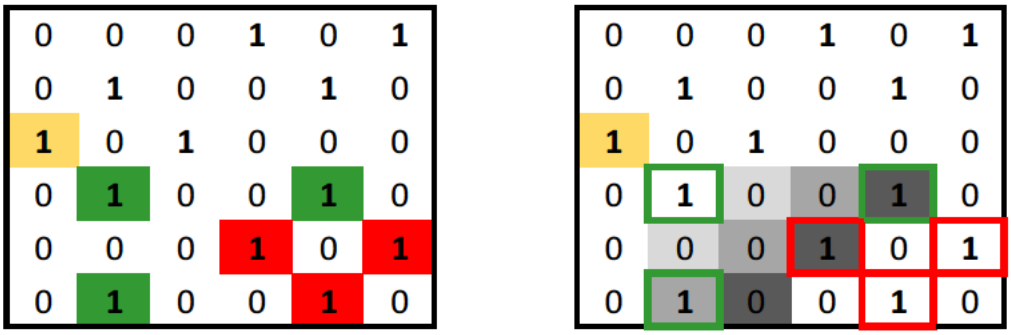


Figure 15: Distance to combination

Assuming the blending matrix has additional rows and columns, the node (5,1) would be however theoretically the better one to be selected next. One common characteristic of each node in the same grey shade is that all the nodes have the same area to be explored after their selection due to their diagonal orientation to each other. Each node has either one more row and one less column or one more column and one less row compared to its diagonal neighbour in the blending matrix that can be explored to find a next combination causing the area to be the same. Nodes in a darker grey shade have a smaller area that can be explored onwards.

To verify the assumption that blending solutions with a greater amount of continuous plies are more likely to be found first by sorting the adjacent nodes, a test run was conducted. A blending problem between two laminates, each having 12 plies, was set up. The plies in the laminate were allowed to have angles of 0, 45, -45 or 90 degrees but were stacked randomly in the laminate. With the blending method, blending solutions were found. Every solution was recorded with the number of continuous plies for the solution and the number, at which point in the search, the solution was found, e.g. 1 as the first solution and 5 as the fifth solution being found. The blending problem was repeated 1000 times to generate enough random stackings to verfiy the assumption. The results are displayed in figure 16.

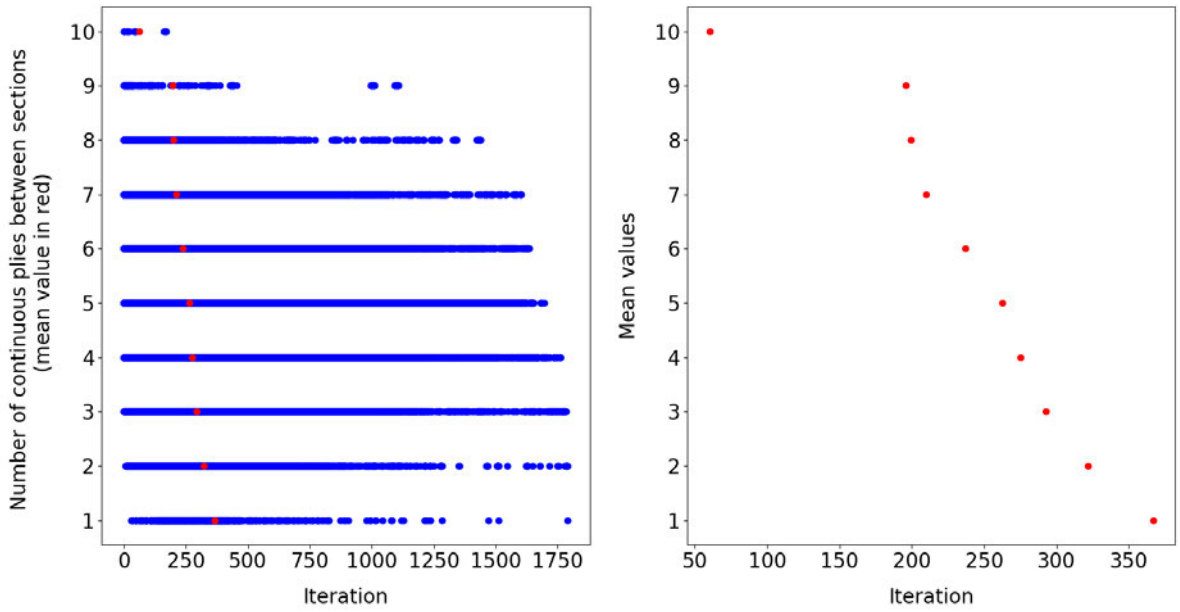


Figure 16: Verifying the directed search

The plot groups every blending solution from every blending problem into the number of continuous plies for the solution and the x-axis marks then the iterations number for every solution. The red dot indicates where the average for each group of numbers lies. The average values are plotted solely on the right to better identify the iteration number. It can be seen that a few blending solutions with 10 continuous plies are found near the beginning of the search.

In general, the averaged values indicate that the directed search finds, on average, solutions with a higher number of continuous plies first. Solutions with fewer continuous plies are found later on in relation to the search duration. Though, it must be mentioned that this behaviour is not guaranteed. Depending on the laminates of the blending problem, it is still possible to find a solution with a small amount of continuous plies before solutions with a greater amount of continuous plies.

4.5 Continuous plies over multiple sections

So far, the blending problem has only been discussed between the two adjacent laminate sections. Extending the problem to find continuous plies over multiple sections, one must search for possible continuous plies between all adjacent sections and combine the results. In figure 17, a blending problem with three sections is displayed.

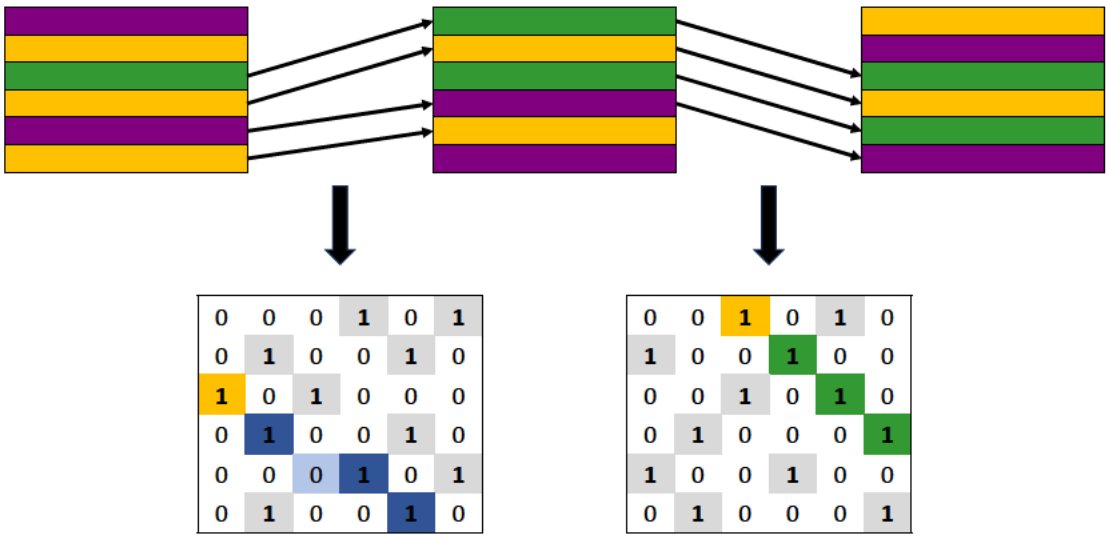


Figure 17: Blending three laminate sections

The task at hand is now to combine the results from the left and middle layup with the results from the middle and right layup. A graph construct can be used again to find these continuous plies. For this application, each ply in each section is represented by a vertex in the graph. The found blending solutions generate the edges for the graph. This representation allows to simply loop over the solutions and add them successively to the graph object. This results in multiple subgraphs, each representing one continuous ply.

5 A blending recovery criterion

As a post-processing step for the proposed blending method, blended laminates have to be assessed in order to compare different blended solutions. In previous work, especially with generic algorithms (see literature review), blending is implemented as an integral part of the laminate generation. For that, constraints are defined in the process that eliminate certain laminates if the constraints are not fulfilled. Usually, these constraints restrict the solution between two adjacent laminate sections [10],[3],[21].
For the proposed blending method, a new criterion for assessing blended laminates is developed, the *Blended Plies Ratio* (BPR). For the BPR, the number of existent plies n is related to the best and worst case of a blended laminate.
The best case can be equated to a fully blended laminate, i.e. inwardly or outwardly blended laminates (figure 2). The worst case reflects a laminate with no existent blending. Every section has its own plies and no plies continues from one section to another. In contrast to methods, where the degree of blending is defined before constructing the laminate structure with its discrete stacking sequence, assessing laminate structures with this criterion can be done independently from their generation. The ratio is calculated with

$$BPR = \frac{n_{no\ blending} - n_{laminate}}{n_{no\ blending} - n_{fully\ blended}} \tag{7}$$

In figure 18, BPRs of different blending solutions for a given laminate structure are presented.

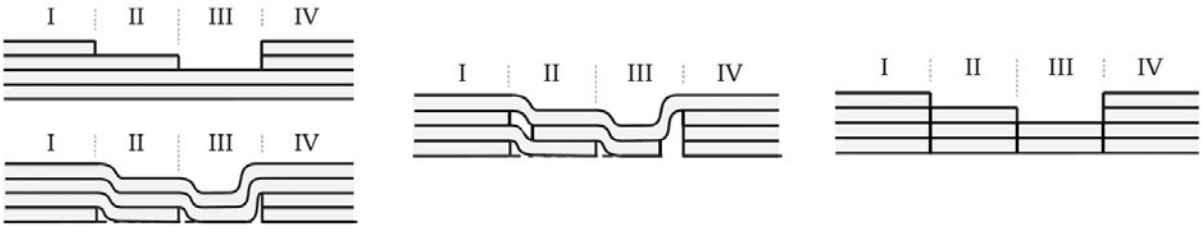


Figure 18: Laminate with different BPRs (left:1, right:0, middle:0,71) [13]

This method of assessing blended laminates with the new criterion broadens the scope from focusing solely on local blending degrees between adjacent sections to considering the entirety of the structure. It was important for the criterion to assess several solutions of a blended structure equally when the local blending degrees between the adjacent sections are the same but the order of appearance is changed for the solutions. Figure

19 shows an example of this requirement. The structure has three sections and for the top example, 4 out of 6 plies continue from the first to the second section, and 5 out of 6 plies continue from the second to the third section. For the bottom example, the number of continuous plies between the sections is swapped. However, both structures have a BPR of 0.75. It further doesn't matter, which of the plies are the continuous plies. This ultimately leads to the fact it is strictly not necessary to count the total number of plies when comparing to blending results for the same structure but rather count only the continuous plies at the intersections between sections. The number of continuous plies can easily be determined as the number equals the length of the path that represents a blending solution in the blending method. The computational effort for a comparison of different blending solutions is therefore reduced.

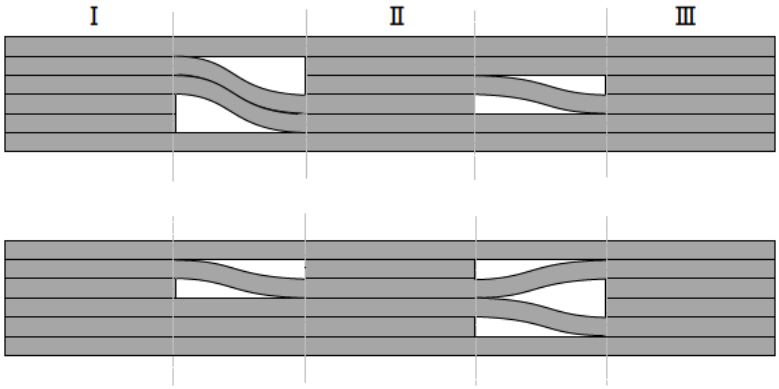


Figure 19: Different laminates with same BPR (0.75)

Another advantage of the method is that the BPR can be averaged. In some cases, it might be desirable to stack two blended designs on top of each other. The BPR for the resulting structure can determined by averaging the BPRs of the combined structures. At similar feature of the blending definition can be found in the publication of Campen et al. [3].
This is especially useful for symmetric laminates as only half of the laminates must be analyzed in terms of its blending degree which reduces the overall computation time for the evaluation.

6 A new layup retrieval method

The following section describes the integration of the blending method and criterion into an optimization process to find blended laminate structures from optimized lamination parameters. A general explanation of the optimizer itself and the optimization process is additionally given.
The main concept of the layup retrieval method is to vary the orientation angles of the plies for every laminate section in the structure and optimize them towards their optimal stiffness from optimized lamination parameters and blending.
In many studies about stacking sequence retrieval, an optimization process is part of the overall retrieval procedure. As mentioned in the literature review, early methods integrated genetic algorithms to find optimal blended laminates. In a more recent study from Sprengholz et al. [27], a rapid, universal layup retrieval algorithm was developed to transform one set of lamination parameters into one continuous stacking sequence. A continuous stacking sequence is here characterised by having continuous ply angles instead of discrete ones, i.e. 0, 45, -45 or 90 degrees.. The objective function of the optimization calculates the difference between optimal the lamination parameter set and the lamination parameter set from the current ply angles.
The objective function for this optimization is similar but instead uses discrete ply angles such that the blending method together with the blending criterion can be applied.

6.1 The optimization problem

For the present retransformation method, the objective function from Sprengholz et al. is adapted to calculate the error for multiple laminates concerning their optimal stiffness values instead of only one laminate. The deviations from the optimal stiffness value from the optimised lamination parameters are calculated with the Root Mean Square Error (RMSE). The effect of each error on RMSE is proportional to the size of the squared error which causes a greater distinction between optimal and less optimal ply orientations for the layup. A comparison of the RMSE and the Mean Absolute Error (MAE) is displayed in figure 20. At [45, 15], the RMSE optimization surface has a significant drop whereas the MAE optimization surface shows a similar decrease in value at [-45, 15]. The surface of the MAE also has more fluctuations compared to the RMSE. The RMSE, by squaring errors, tends to weigh larger errors more heavily, yet it can obscure the significance of improvements in individual lamination parameters. Squaring small errors further diminishes their impact on the overall error, making incremental improvements

seem less influential. Both features of the RMSE, i.e. a greater distinction to the optimal

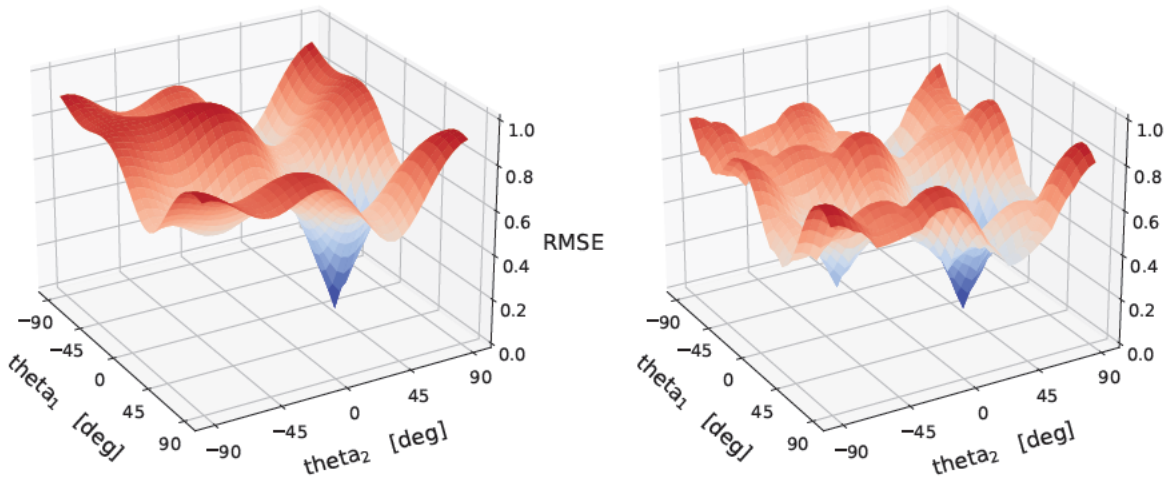


Figure 20: Plot of the RMSE (left) and MAE (right) optimization surface for a [45 degree; 15 degree] laminate

To receive a total error for multiple laminates, a total RMSE from each laminate RMSE is calculated (equation 8). The target lamination parameters V_t are compared to the current lamination parameters V_i for each laminate m for a total number of n laminates. In figure 21, the optimization surface for the total error for two single-ply laminates with 45 and 15 degree ply is plotted. The surface is similar in shape to that of the single laminate RMSE (figure 20, left). The values of the RMSE however differ for the ply orientations.

$$RMSE_{sprengholz} = \sqrt{\frac{(V_t - V_i)^2}{12}} \quad ; \quad RMSE_{method} = \sqrt{\frac{\left(\sum_{m=1}^{n} \sqrt{\frac{(V_{t_m} - V_{i_m})^2}{12}}\right)^2}{n}} \tag{8}$$

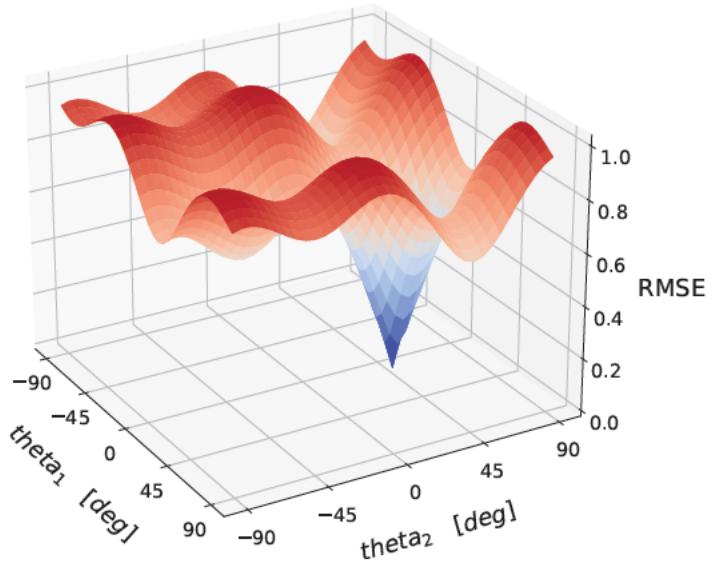


Figure 21: Plot of the RMSE optimization surface for the two laminates with 45 degree and 15 degree ply

The derivation of the lamination parameters is discussed in section 2. For the sake of completeness, the LPs are displayed in equation 9. In this work, the notation for the LPs from Tsai and Pagano [29] is used. If all single plies are of the same material and thickness, the twelve LPs can be defined without the total thickness of the laminate. The following definitions can also be found in the publications from Sprengholz et al. [27] and Macquart et al. [13] with the difference in notation.

$$
\begin{aligned}
V^{A}_{[1,2,3,4]} &= \frac{1}{N}\sum_{n=1}^{N}\left(z_n - z_{n-1}\right)\left[\cos\left(2\theta_n\right), \cos\left(4\theta_n\right), \sin\left(2\theta_n\right), \sin\left(4\theta_n\right)\right] \\
V^{B}_{[1,2,3,4]} &= \frac{1}{N^2}\sum_{n=1}^{N}\left(z_n^2 - z_{n-1}^2\right)\left[\cos\left(2\theta_n\right), \cos\left(4\theta_n\right), \sin\left(2\theta_n\right), \sin\left(4\theta_n\right)\right] \\
V^{D}_{[1,2,3,4]} &= \frac{4}{N^3}\sum_{n=1}^{N}\left(z_n^3 - z_{n-1}^3\right)\left[\cos\left(2\theta_n\right), \cos\left(4\theta_n\right), \sin\left(2\theta_n\right), \sin\left(4\theta_n\right)\right]
\end{aligned}
\tag{9}
$$

The distance from the top z_n and bottom z_{n-1} of each ply n (from a total number of N plies) to the laminate centre is defined by

$$z_n = -\frac{N}{2} + n \tag{10}$$

To evaluate the blending of the laminate with the BPR, it is beneficial to have a second objective function calculating the BPR, which makes the optimization a multi-objective optimization (MOO). Unlike traditional optimization problems that aim to optimize a single objective, MOO deals with situations where there are multiple goals to be achieved simultaneously, often with conflicting interests. These objectives could be maximizing or minimizing certain factors, and the challenge lies in finding the best possible compromise or trade-off between them.

In MOO, the solution space is complex because improving one objective might lead to the degradation of another. The goal is to find a set of solutions that represent the best trade-offs among these conflicting objectives, known as the Pareto optimal solutions. These solutions are considered optimal because no other solution can improve one objective without worsening at least one other objective. An example of a possible Pareto-Front is shown in figure 22, where objective 1 is subject to minimisation and objective 2 is subject to maximisation.

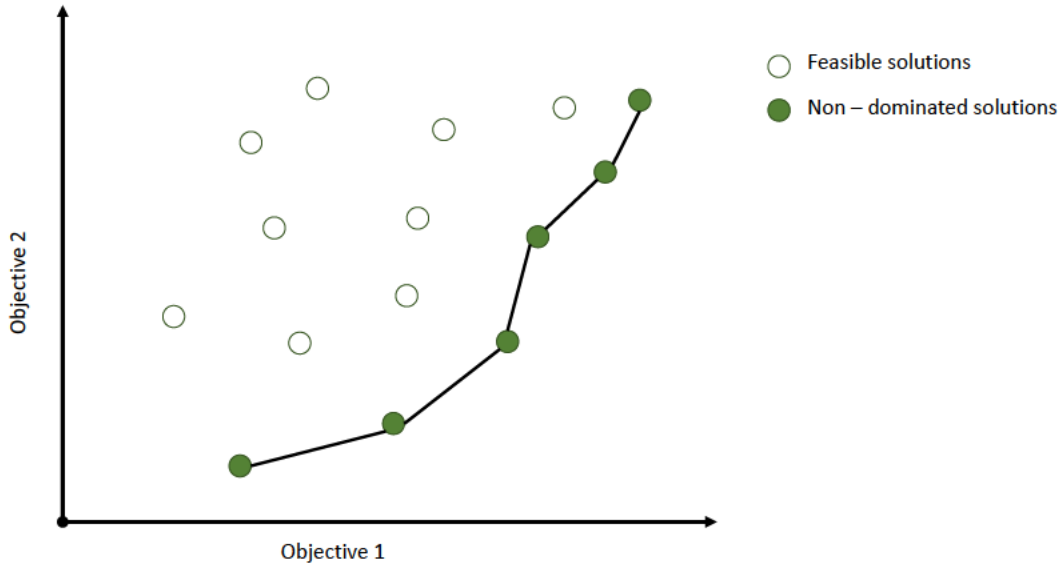


Figure 22: Example of Pareto Front

While it is possible to set the BPR as a constraint for the optimization, defining it as an objective is more suited to the goal of the optimization. Plotting the Pareto front can reveal improvements for the stiffness of the laminate via the lamination parameters for a compromise in blending if the lamination parameters of adjacent sections would dissolve in laminates with dissimilar plyshares or stackings. In these conditions, the blending method is most effective as the blending between the laminates can not easily be made just by looking at the stackings. Consequently, the guide-based optimization with perfect blending excels, when the laminates would differ less in the stacking sequence.
Based on the above-mentioned objectives, the formulation of the optimization problem can be stated as: find a set of design variables x that will

$$\begin{aligned} &minimize && f_1(x) \\ &maximize && f_2(x) \\ &subject\ to: && g_i(x) = 0, \quad i = 1, .., m_e \\ & && h_i(x) \geq 0, \quad i = 1, .., m_i \\ & && x = \{x_1, x_2, .., x_i\}, \quad i = 1, ..., n \\ & && x_i \in L, \quad L = \{\theta_1, \theta_2, ..., \theta_k\} \end{aligned}$$

where f_1 defines the RMSE for the whole structure (equation 8, right) and f_2 defines the BPR, which both are subject to possible equality constraints g and inequality constraints h. Each design variable represents one ply in one section with its ply angle θ.

6.2 The optimizer

Already in the year 2001, Kristinsdottir et al. [8] stated

> "It has been shown that optimal design of composite structures is a global optimization problem, with multiple local optima and complex design space"

This has been proven many times over past years, lately by Sprengholz et al. [27]. The optimization surface plots in the previous chapter further confirm this statement.
It is therefore a necessity to employ an appropriate optimizer that can handle those complex design spaces. Unlike conventional optimizers that need continuous design variables, the optimizer for the current problem must be able to work with discrete values for the design variables due to the implementation of the optimization problem.

One optimizer, that can be applied for the current problem, is MIDACO (Mixed Integer Distributed Ant Colony Optimization) [22]. MIDACO serves as a solver designed to tackle a wide array of numerical optimization problems. It's a versatile tool applicable to continuous, discrete/integer, and mixed integer problem types. Whether dealing with single- or multi-objective optimization, MIDACO handles problems constrained by equality and/or inequality limitations.
What sets MIDACO apart is its adaptability to handle substantial problem sizes, comfortably managing thousands of variables and hundreds of objectives. Its strength lies in implementing a derivative-free, evolutionary hybrid algorithm. This algorithm approaches problems as black-box scenarios, accommodating various critical function properties like non-convexity, discontinuities, or stochastic noise. Midaco is fundamentally based on an Ant Colony Optimization (ACO) algorithm. ACO mirrors how ants find food efficiently. Developed in 1990, this technique mimics how ants efficiently find the shortest path between their nest and food sources. This technique has evolved into a versatile problem-solving approach.
At its core, ACO revolves around the collaborative behaviour of ants. The insects communicate through pheromones, leaving chemical traces along their paths. When an ant discovers a food source, it returns to the nest, leaving behind a trail of pheromones. Other ants then follow these trails, with a preference for paths marked by a higher concentration of pheromones. Over time, shorter paths are favoured due to the faster accumulation of pheromones, leading to an emergent optimal route. This behaviour is displayed in figure 23. The source is equivalent to the nest of the ants and the target represents possible food. Initially, the ants are spread randomly on the paths towards the food. But over time, the concentration of ants on the shorter path is increased.

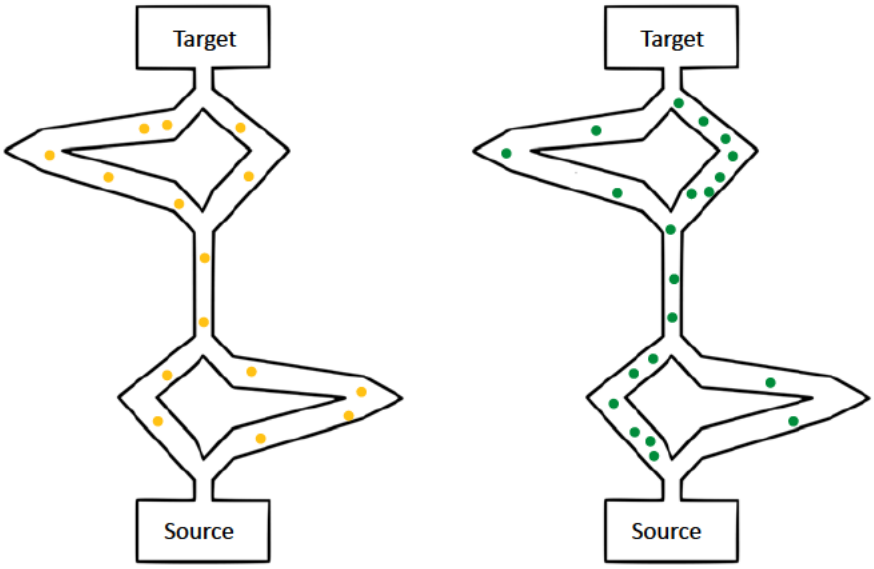


Figure 23: Ant colony behaviour when finding food (left: position of ants after initialisation, right: ants on preferred path

This algorithm works in two stages: solution construction and pheromone updates. During solution construction, artificial ants build potential solutions by probabilistically selecting paths based on pheromone levels and heuristic cues. Pheromone trails indicate path desirability, while heuristic information guides ants towards promising areas in the search space. This helps in gradually constructing solutions approaching optimal outcomes.
Concurrently, the pheromone update process mimics natural reinforcement. As ants use paths, pheromone levels adjust based on solution quality. Shorter paths receive higher pheromone deposits, becoming more attractive to subsequent ants. This iterative process drives the algorithm toward improved solutions.
ACO's adaptability finds use across various optimization problems, including the travelling salesman problem and route planning. It excels in handling intricate, combinatorial problems with vast solution spaces.
However, ACO faces challenges, particularly in parameter tuning. Achieving a balance between exploration and exploitation requires careful adjustment of parameters and understanding problem characteristics. Additionally, ACO's performance might fluctuate in dynamic or large-scale instances due to its reliance on pheromone trails.

The key features of MIDACO that are used for the optimization problem are the black box handling of the objectives functions and its ability to work with up to 10000 discrete variables. The documentation of MIDACO also states that the software excels in global optimization problems which is in line with the statement from Kristinsdottir et al. For reference, global optimization aims to find the best solution considering all possibilities, covering the entire feasible region. On the other hand, local optimization focuses on finding the best solution within a limited area, typically around a starting point. Global optimization explores the entire space, potentially taking longer, while local optimization, being faster, examines a smaller portion. However, local optimization runs the risk of getting trapped in local best solutions and missing the global optimum.

6.3 Optimization process

This chapter explains the implementation of the optimization process with the integration of the blending method. Figure 24 highlights the main process steps that are performed at every iteration.
First, the design variables are split into subarrays with the desired length of each section such that each subarray represents one section in the laminate with its size and ply angles. The values in the subarrays at this point are not the actual ply angles but only representative discrete integers as MIDACO is not able to select values for the design variables from a predefined list of ply angles. Instead, the range of possible values can only defined via box constraints that limit the variables to an upper and lower bound. The discrete integer values can then be transformed to ply angles with the use of a ply angle mapper that maps the box-constrained integers to any desired ply angles.
For each section, the lamination parameters are then calculated and compared to the desired lamination parameters. And second, the BPR is calculated for the evaluation of the blending. Here, the sign of the BPR must be changed because MIDACO demands all objectives to be minimized.

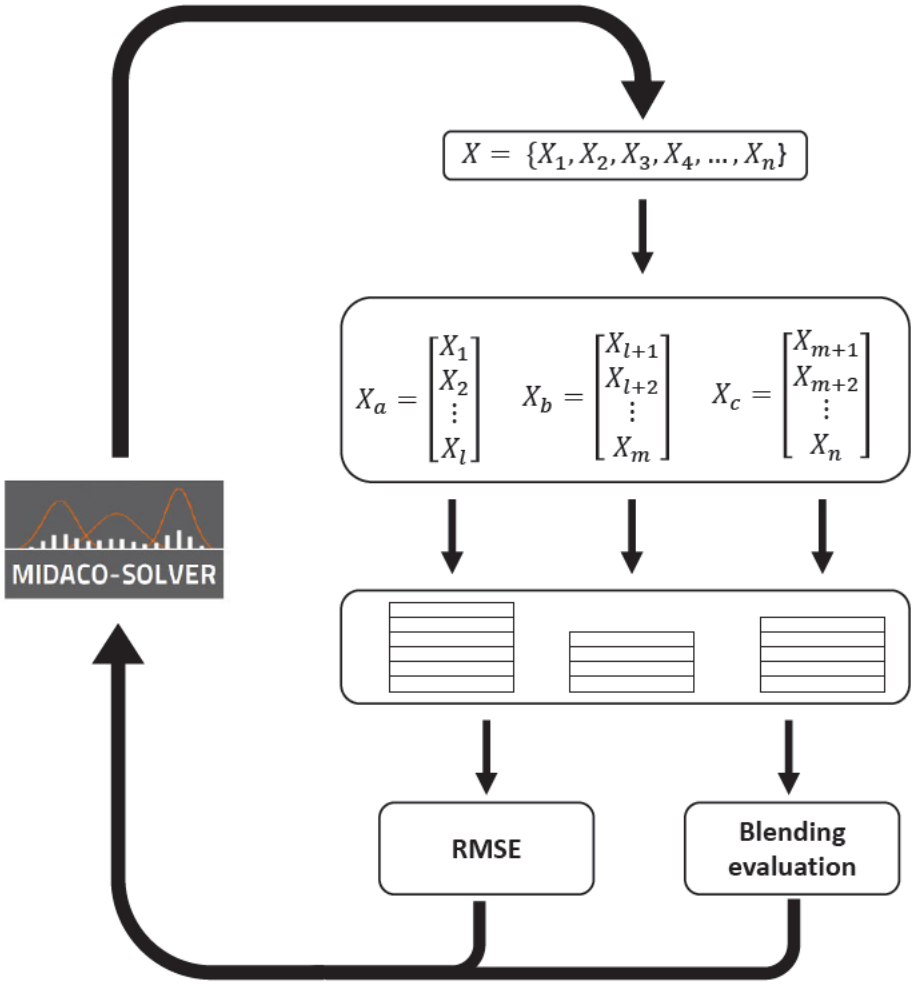


Figure 24: Optimization prcocess

6.3.1 Implementing design rules

In order to optimize the utilization of the inherent strengths of composite materials while effectively addressing their weaknesses, designers have developed specific design rules over the past few decades. These guidelines stem from a combination of prototype testing outcomes and practical industrial knowledge. The primary purpose of implementing these design rules is to prevent the emergence of critical failure modes, such as delamination, while ensuring that the design is feasible for manufacturing. Typically, these rules are formulated based on the ply level either in terms of thickness or in the longitudinal direction. For this optimization, certain design rules are selected based on their importance and realisability. Further design rules can be found in [11],[16].

Laminates with ply angles symmetrically aligned to the mid-plane are termed **symmetric laminates**. As highlighted in the reference [16],[11], these laminates offer a significant advantage: their uncoupling of membrane and bending behaviours within the structure results in a more predictable deformation behaviour. This predictability simplifies analysis and testing processes, facilitating the measurement of stiffness and strength values of the structure. Furthermore, managing tolerances during assembly becomes more straightforward. It is important to note that adhering strictly to this design rule might not always be feasible, particularly in areas where laminate thickness changes, such as tapering regions. In cases where laminates are locally non-symmetric, placing the asymmetric part as close as possible to the mid-surface helps minimize warping effects. To create symmetric laminates in the optimization process, the number of design variables is cut in half and the resulting subarrays (figure 24) are reversed and then concatenated with the original one, thus forcing the $V^B_{[1,2,3,4]}$ lamination parameters (equation 9) to be 0. This reduces the design space, making the search for optimal solutions more efficient. For the blending evaluation, half of the subarrays are used as the BPR doesn't change if the sub-laminates are mirrored. Consequently, symmetrical laminates can be analyzed with the computational effort for only half the laminate size.

Another design guideline specified in Niu [16] concerns the arrangement of **±45 degree plies on the outer surfaces** of the laminate in the thickness direction. This arrangement aims to enhance damage tolerance and increase resistance against impacts, such as tool drops. Simultaneously, it's advised to avoid placing the load-bearing 0 degree plies on the outer surfaces due to operational considerations. This recommended approach is particularly beneficial for potentially unstable laminates, aiming to optimize buckling resistance. To enforce ±45 degree plies on the outer surface of the laminate, the box constraints for the design variables representing the outer plies can be limited to only the desired integer according to the ply angle mapper.

Expanding on the concept of symmetric laminates, the application of **balanced laminates** becomes relevant. Balanced laminates entail ply angles arranged such that, aside from 0 degree and 90 degree, each angle is paired as $+\theta$ and $-\theta$ above and below the mid-surface. These laminates offer similar advantages to symmetric ones. Notably, they enable the decoupling of in-plane membrane and shear behaviour within the structure. This decoupling arises from specific characteristics in the stiffness terms of the plies, as elucidated by Jonas [7]. The coupling between membrane stiffness terms relies on the summation of ply stiffness terms containing odd powers of sine and cosine functions. Consequently,

0 degree and 90 degree angles do not contribute to these stiffness terms, and when $+\theta$ angles have a corresponding $-\theta$ angle, their summation results in zero, also contributing nothing. Another significant criterion favouring symmetric and balanced laminates involves maximizing buckling strength. As detailed in references like Niu [16] and Jonas [7], the coupling between membrane-bending and bending-twisting amplifies deformation while simultaneously reducing buckling resistance and vibration frequencies. This effect aligns with the expected behaviour of panels possessing significantly lower bending stiffness. This design rule must implemented by formulating an appropriate constraint on the optimization process. In theory, it would be possible to instead adjust the ply angle mapper to map the boxed integer values not to single ply angles but to ply pairs. One integer value could then represent the ± 45 degree ply pair. The drawback would be that other ply angle orientations must also be mapped as a pair of two ply angles to ensure that sub-laminates do not exceed or fail short of the specified laminate size.

7 Limitations of the blending method

In structural optimization, structures are usually subdivided into smaller subsections to be able to address the different load distributions within the structure more effectively. A great effort has been put into making the blending method available for all combinations of subsections that ultimately build up the laminate structure, which ultimately could not be achieved.

As mentioned in section 4.5, a completely blending solution for the whole structure is constructed from the blending solutions between each subsection. The rules for searching for solutions in the blending matrix permits solutions, where continuous plies would cross each other. A combination of those solutions would therefore also create laminates without crossed plies. However, if blending is only considered between individual sections and then arbitrarily combined with solutions from other sections, a different problem may arise for certain combinations of sections. It can occur that a continuous ply A is placed in one section below a continuous ply B, while in another section, ply B supports ply A. This is of course not possible in production processes where the plies have to be placed one after the other. A laminate structure where this scenario can possibly occur is displayed on the right in figure 25. When connecting the centre of individual sections with lines, these structures form a loop that allows plies to reach certain sections from different directions.

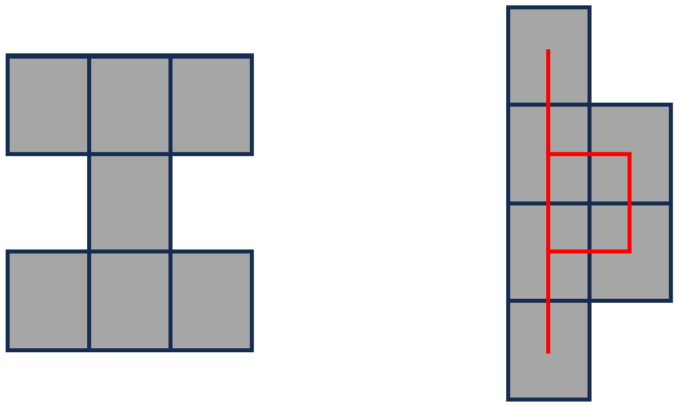


Figure 25: Possible (left) and non-possible (right) laminate structures

Many attempts have been made to avoid this problem. One idea in particular was to first allow these kinds of scenarios and apply a post-processing step that cuts the continuous

plies a certain positions such that the continuous ply A is always below the continuous ply B. This, however, would result in an optimization problem where the cut positions must be determined while still achieve an adequate blending. As this problem was discovered near the end of the project, the task of finding a lightweight optimizer that could easily be implemented was out of the scope of this project.

Another limitation of the blending method is related to the number of plies for the relevant sections to be blended. As the number of plies increases, the graph that represents the blending matrix grows simultaneously which leads obviously to more blending solutions. The number of solutions generated from the graph is, however, not linear but shows an exponential relation with the laminate size. Figure 26 shows the relation between the number of solutions for different laminate sizes. The plot displays the results of a blending problem analysis with two randomly generated single-section laminates. For each laminate size, 100 different blending problems are generated and with the blending method, the mean value for the recorded number of solutions and the mean time to find all solutions is plotted over the laminate size. The diagram highlights the exponential relation between the number of plies per laminate and the solution found between two laminates.

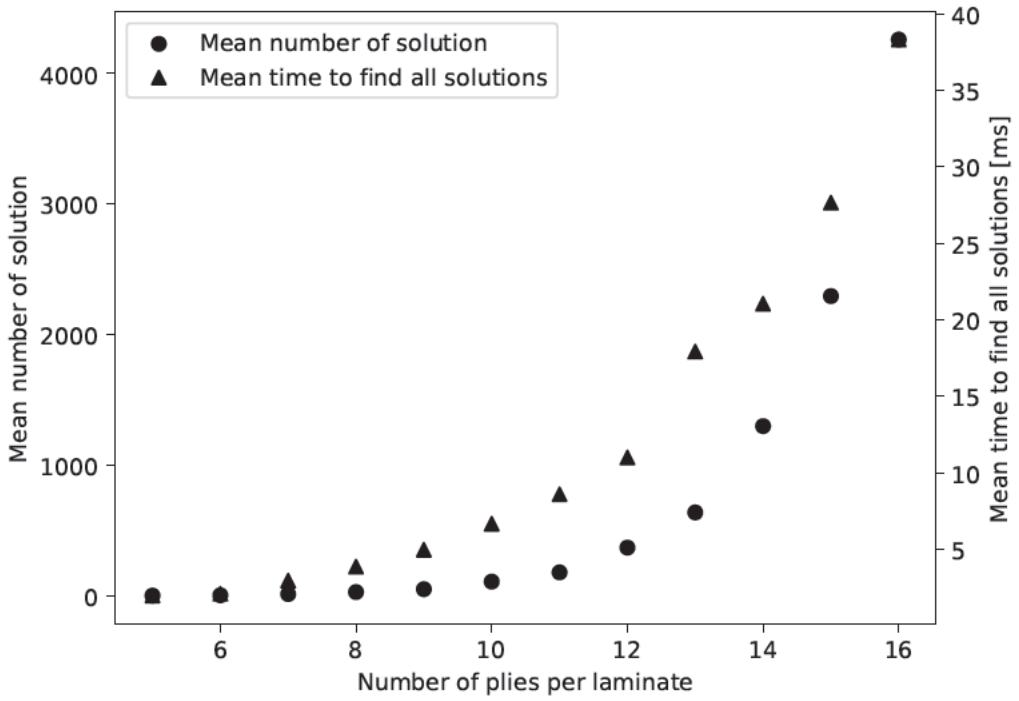


Figure 26: Number of solutions computation time with randomly generated laminates with different stacking sizes

The diagram demonstrates the exceptional performance for calculating the blending solutions for smaller laminate sizes. One must keep in mind, that these are randomly generated laminates. The computation time is linked to the generated number of solutions. The number of solutions is mainly dependent on the size of the laminate and the number of combinations that can be made between the left and the right layup in a two-laminate problem. Laminates with only one fibre orientation angle (extreme laminates) cause a high computation due to the many variations of continuous plies.
To counteract this exponential relationship between the laminate size and the number of solutions when searching for optimal blending solutions, the directed search for the DFS, described in section 4.4 was developed, so that the chance of finding better blending solutions first is increased and one doesn't analyse the whole solution graph.

8 Application examples

This chapter contains at the beginning applications of the blending method for two simple blending problems to showcase the capabilities and limitations of the method, which are followed by two self-created and one final benchmark optimization problems. The discussion of the results can be found in the following section.

8.1 Two-section-blending

The first blending problem is a simple two-section blending problem with four different ply orientations (0, 45, 135 as a representative for -45 and 90 degrees) and a laminate size of 26 plies. The two laminates with one of the best blending solutions with the highest number of continuous plies are displayed in figure 27. The method finds in 43 seconds 3702721 blending solutions. 50 blending solutions have the maximum of 21 continuous plies between the two laminates. The first solution with 21 continuous plies is available after 0.6 seconds at iteration step 35700 from 3702721. The last solution with 21 continuous plies is found after 15.8 seconds at iteration step 1607533 from 3702721.

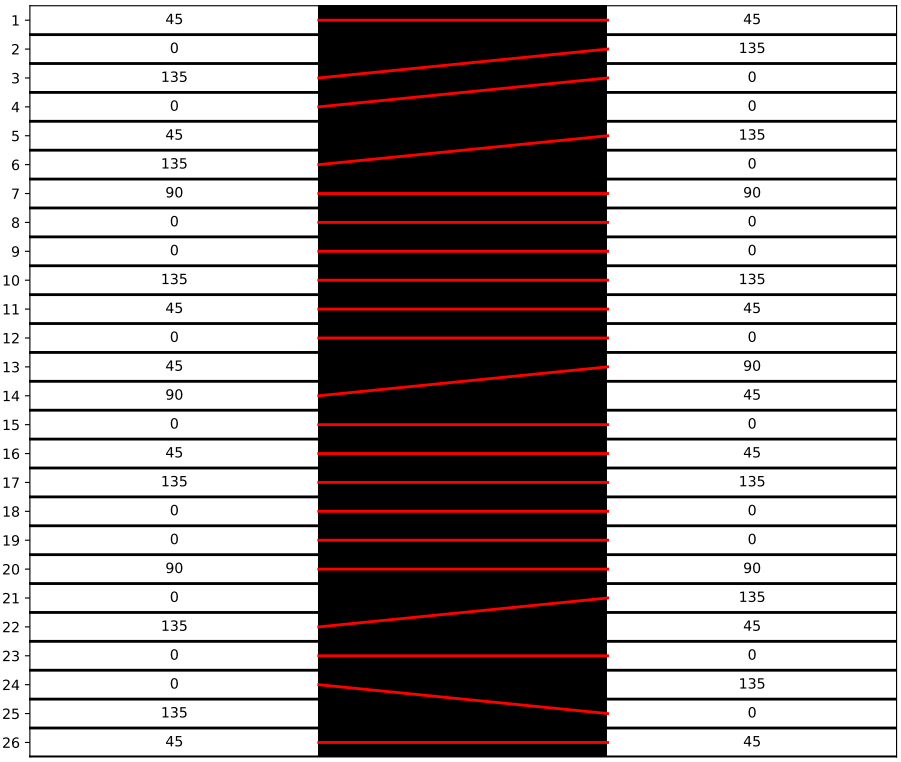


Figure 27: Two laminates with random plies

It was identified through multiple test runs with random plies that analyzing a blending problem with 25 plies using one thread on an Intel(R) Core(TM) i5-8265U takes around one minute, depending on the stacking of the two laminates. Adding plies from here one will have a noticeable increase in the computation time as the number of solutions significantly increases with increasing ply numbers as described in section 7. A two-section blending problem with 12 plies and only one ply angle takes about 2 seconds to run and the blending method finds 281287 solutions. Compared to a randomly generated two-section blending problem that is a 400 per cent increase in computation time.

8.2 Multiple-section-blending

An example of the multiple-section-blending problem is given in figure 28. The method takes about 2.1 seconds to calculate 166705 blending solutions between the sections. The best blending solutions result in a laminate structure with a total of 44 plies. This concludes to a BPR of 0.84.

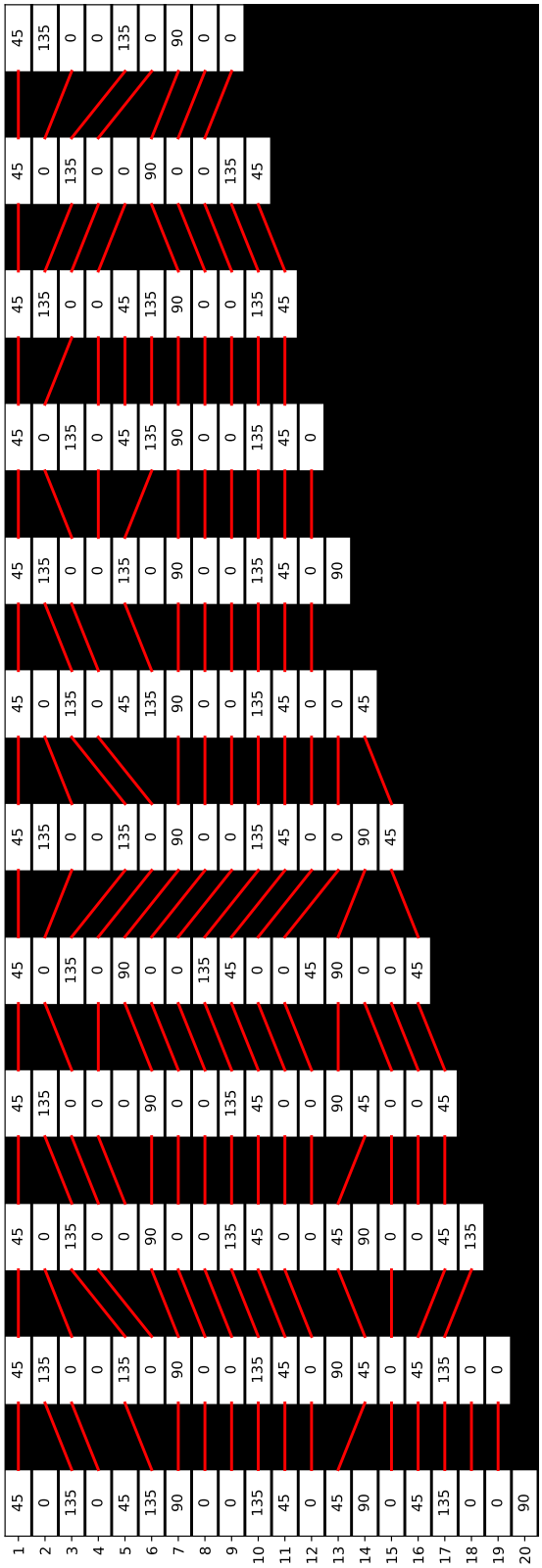


Figure 28: Multiple laminates with random plies

8.3 Optimization results

8.3.1 Verification of the optimization process

During the integration of the blending method, multiple tests were conducted to verify if MIDACO can find pre-known optimal solutions to a problem. More specifically, the task for these test runs is to find the stackings of each laminate such that the difference from the optimal lamination parameters is zero. The test problem is a three-section laminate structure, each section having 20 plies. The lamination parameters are built up such that they would represent a symmetric laminate with an equal amount of 0 and 90 degree plies, stacked collected together on top of each other, beginning with the 0 degree plies. The blending algorithm is disabled for this optimization as an optimum in the lamination parameter set would automatically cause a perfect blending. The LPs are shown in equation 11.

$$\begin{aligned} V^{A}_{[1,2,3,4]} &= [0,1,0,0] \\ V^{B}_{[1,2,3,4]} &= [0,0,0,0] \\ V^{D}_{[1,2,3,4]} &= [0.75,1,0,0] \end{aligned} \tag{11}$$

The design variables are allowed to take one of the four common 0, 45, -45 or 90 degree ply angle.
The optimization is split into two sub-optimizations to first find the region where the optimum could possibly ly and then secondly, with starting values for the design values from the previous run, to refine the solution and find the optimal solution. This has been proven to be a good strategy for finding optimal solutions. While MIDACO has its own restart mechanism to avoid getting stuck in certain local optimums, a manual restart of the optimization further minimises the time to find the optimal solution. Restarting the optimization to find faster better solutions is advised by the MIDACO manual and was also part of the optimization process from Sprengholz et al. [27].
The optimization takes in total 304000 iterations to find the optimal stacking sequence after 102 seconds. Interesting to note is that MIDACO can find a solution close to the optimum in just one second after 4000 iterations. Figure 29 shows the results after 4000 iterations and after 304000 iterations. It can be seen that, after 4000 iterations, the 90 degree plies are already more concentrated near the centre of the laminate and the 0 degree plies are located further away from the centre. Additionally, the plyshares are also close to the desired plyshares.

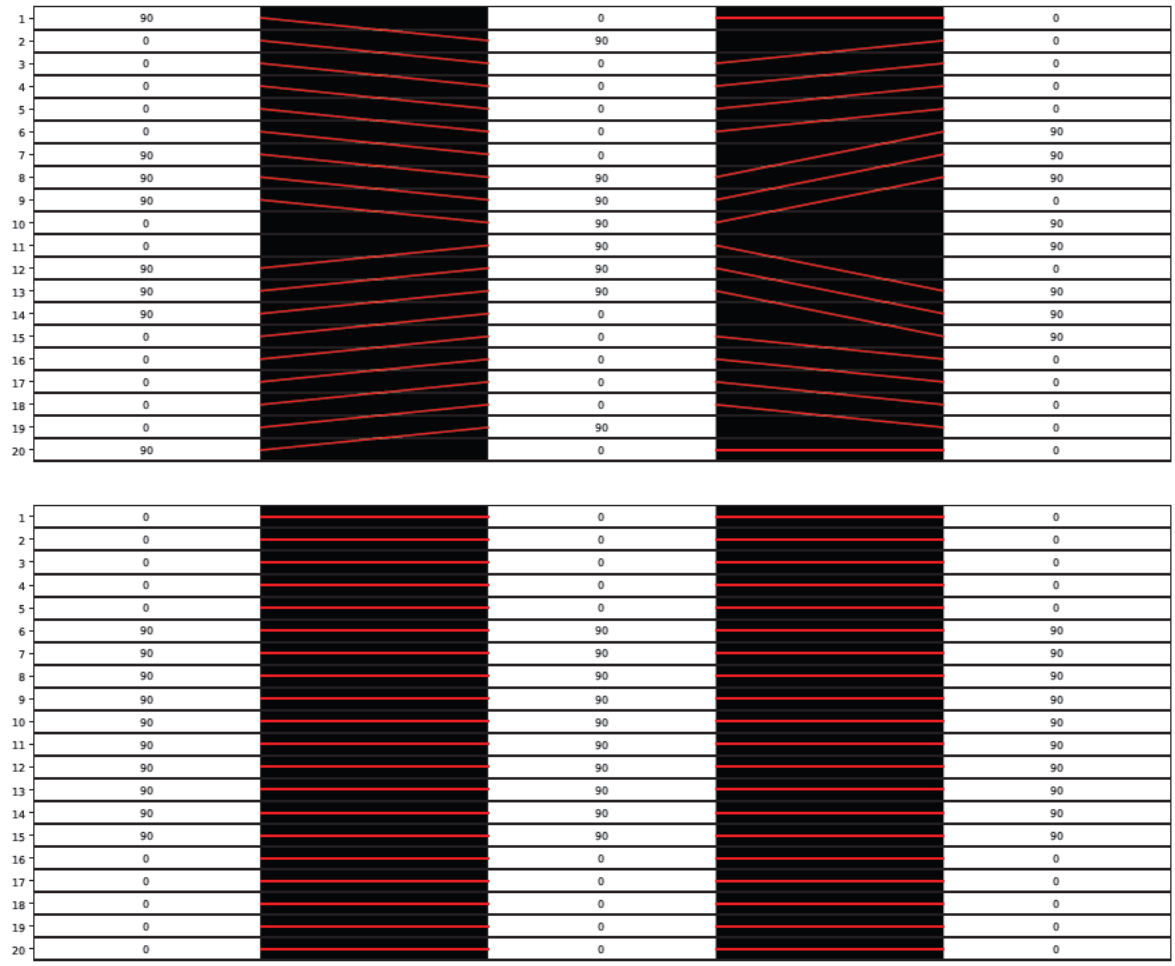


Figure 29: Finding the optimal stiffness (top: 4000 iterations, bottom: 304000 iterations)

8.3.2 Pareto-Front Example

This second example shows an extreme scenario with a three-section laminate, where the outer sections have LPs representing a laminate with only 0 and 90 degree plies and the middle section with only 45 and -45 plies. The individual laminates have again 20 plies and are symmetrical. The LPs for the laminate with only 45 and -45 degree plies are shown in equation 12.

$$\begin{aligned} V^{A}_{[1,2,3,4]} &= [0, -1, 0, 0] \\ V^{B}_{[1,2,3,4]} &= [0, 0, 0, 0] \\ V^{D}_{[1,2,3,4]} &= [0, -1, 0.75, 0] \end{aligned} \tag{12}$$

Figure 30 plots the Pareto front of the current problem with the two objective functions, the BPR for the blending evaluation and the RMSE for the stiffness evaluation. The BPR values are all negative as all objective functions are subject to minimization, which is mandatory for MIDACO. The plot is generated with the plotting tool from MIDACO. The optimization took in total 764 seconds for 44000 iterations. MIDACO found 11 non-dominated Pareto points during the optimization. The lowest RMSE of 0.154 can be achieved with a BPR of 0.3. The best blending with a BPR of 0.8 results in an RMSE of 0.346. The most balanced solution is marked with a green hexagon.
It is possible to set a balance parameter for MIDACO. This parameter defines on what part/area of the Pareto front MIDACO should focus its main search effort. By default and for this Pareto front, MIDACO focused most of its search effort on that part of the Pareto front, which offers the best equally balanced trade-off between all objectives. Using the BALANCE parameter, this focus can be shifted to any part of the Pareto front. Shifting the focus completely to BPR, it was identified that MIDACO is also able to achieve 100 per cent blending for certain blending problems.
The RMSE values from the Pareto Front for the three sections along with the corresponding MAE are listed in table 1. It can be seen that with an increase in the BPR, the deviation from the optimal stiffness values increases as well except for those of the left section. The deviation is minimized for a BPR of 0.6 compared to vales for a BPR of 0.3 and 0.8 which both have an RMSE close to 0.19.

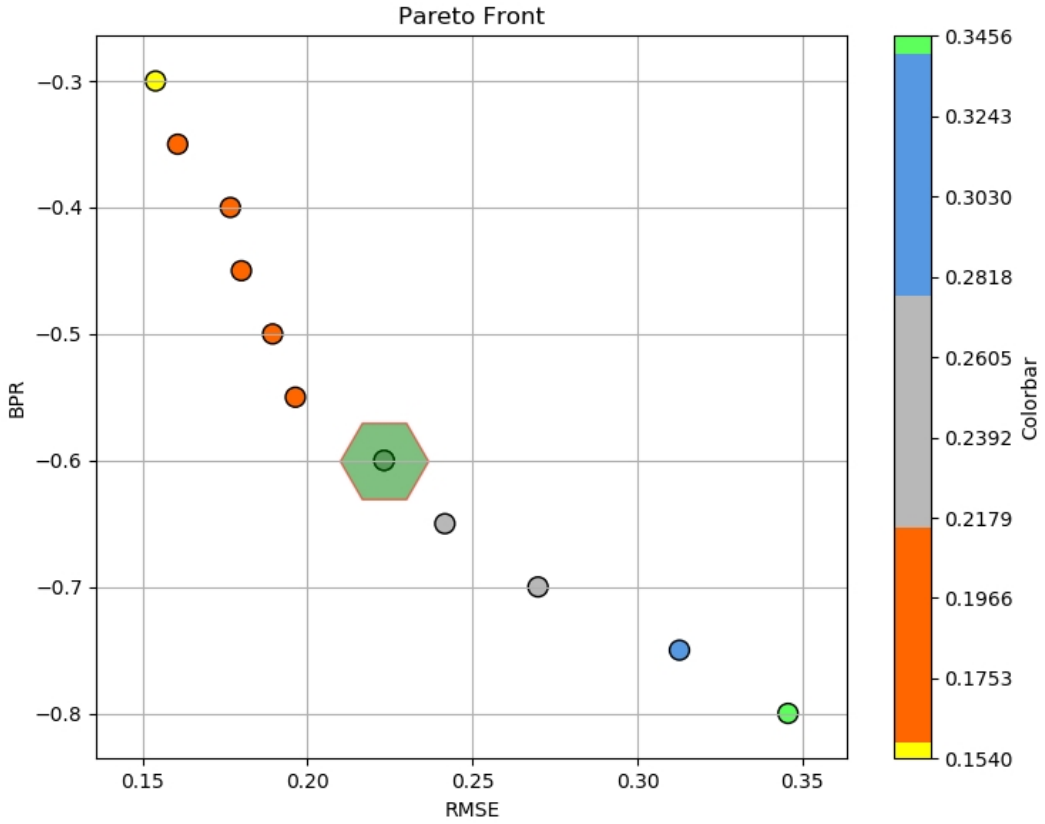


Figure 30: Pareto Front for non-matching ply sections

Table 1: RMSE and MAE for Pareto points

Section	BPR = 0.3		BPR = 0.6		BPR = 0.8	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
left	0.1954	0.1265	0.1651	0.0857	0.1916	0.1290
middle	0.1550	0.0747	0.3015	0.1688	0.4871	0.2685
right	0.1092	0.0550	0.1770	0.1032	0.2904	0.1567
combined	0.1540	0.0854	0.2232	0.1192	0.3456	0.1847

8.3.3 Wing Box Use Case

One common use case for composite optimization methods that is utilised in many publications ([12], [19], [17], [24]) is the wing box use case (figure 31). The wing box is fixed at its root section and subjected to four asymmetric transverse forces at the wing tip. Within this structure, there are nine distinct design regions in both the upper and lower covers, delineated by four spars and ribs, intended for optimization. For this analysis, the design regions are reduced such that there is only one region widthwise. The id for the new design region results from the highest id of the individual design regions. For example, regions 1, 2 and 3 are combined to the design region with the id 3.

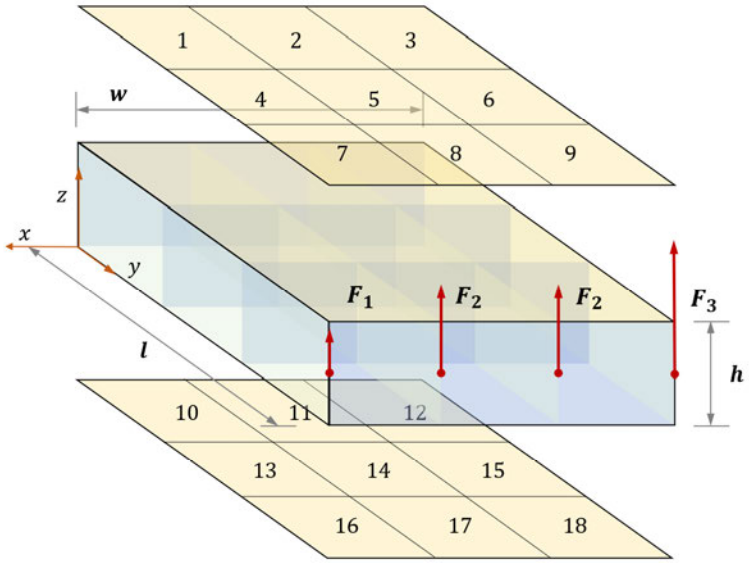


Figure 31: Geometry of the wing box use case

The same use case has been used in the publication of Liu et al. [12] about a bilevel optimization of blended composite wing panels. In the paper, two optimization approaches are used for the optimization of stacking sequences of laminated composite structures: a smeared-stiffness-based method and a lamination-parameter-based method. In the smeared-stiffness-based method, a gradient-based optimization is used to optimize the total volume of the structure at the top level subject to the buckling and strain constraints. A blending scheme and a ply shuffling code based on the layup design rules

are applied. In the lamination-parameter-based method, the total number of plies and the lamination parameters related to the out-of-plane stiffness matrix are treated as the design variables in the top-level optimization problem. Buckling and strain constraints are applied at this level and the total material volume is the objective function. Next, a permutation GA is used to shuffle the layers to minimize the difference between the values of computed lamination parameters for a current stacking sequence and those coming from the top level. This is embedded into a blending procedure applied at this level to achieve global ply continuity.

For this analysis, the genetic algorithm is replaced with the optimization procedure presented in this thesis. The data from the top-level optimization with the lamination-parameter-based method from Liu et al. is presented in table 2. The variable n represents half the number of the plies in the total stack that have the specified ply angle. For clarification, the D-Lamination parameters differ from those presented in the paper from Liu et al. A different notation other than the notation from Tsai and Pagano [29] was used. A transformation of the LPs was further performed such that all lamination parameters were above zero. Table 2 presents the back-transformed LPs with the notation from Tsai and Pagano.

Table 2: Data from Liu et al. [12]

Panel No.	n_0	n_{45}	n_{90}	V_1^D	V_2^D	V_3^D
12	5	1	1	0.5573	-0.3927	0.0869
15	4	1	2	0.1716	-0.3141	0.0864
18	8	2	3	-0.2352	-0.1038	0.0728

In the publication from Liu et al., five design rules are considered for the genetic algorithm

1. The laminate is symmetric
2. The laminate is balanced
3. Because of the damage-tolerance requirements, the outer plies for the skin should always contain at least one set of ±45 deg plies
4. The number of plies in any one direction placed sequentially in the stack is limited to four
5. A 90 deg change of angle between two adjacent plies is to be avoided, if possible

Applying some of the design rules, the set of individual LPs can be reduced. The lamination parameters resulting from the extension-bending coupling matrix $[B]$ are all zero. Is the laminate in addition balanced according to its mid-plane, the third and fourth LPs $V^A_{[3,4]}$ for the $[A]$-matrix are zero. 0 degree and 90 degree plies have no contribution to the sum of sine functions. Pairs of angled plies cancel out each other due to the antisymmetry of the sine function with respect to the ordinate. Is the set of allowable ply angles reduced to 0 degree, ±45 degree and 90 degree plies, the fourth LPs are zero, due to the factor 4 within the sine function. The lamination parameters for the A matrix are linear combinations of the ply share and can be computed with

$$\begin{aligned} V_1^A &= 2\left(\frac{2n_0}{n_{total}} - 0.5 + \frac{n_{45}}{n_{total}}\right) \\ V_2^A &= \frac{\frac{2n_{45}}{n_{total}} - 0.5}{-0.5} \end{aligned} \tag{13}$$

where n_{total} is the total number of plies in the stack. With this information, the target LPs for the optimization can be calculated (table 3).

Table 3: Target LPs for wing box use case

Panel No.	V_1^A	V_2^A	V_1^D	V_2^D	V_3^D
12	0.5714	0.7143	0.5573	-0.3927	0.0869
15	0.2857	0.7143	0.1716	-0.3141	0.0864
18	0.3846	0.6923	-0.2352	-0.1038	0.0728

The first design rules for symmetry and outer-ply-damage-tolerance from Liu et al. are implemented for the retransformation of the LPs to discrete stackings. Implementing the other design rules would result in less to no Pareto points in the Pareto front. The deviation from the target LPs in the form of the RMSE and MAE is presented in table 4. The difference in error between the several blending solutions for the three panels is relatively small except for panel 12 between the blending solution with a BPR of 0.9 and 1.0. The RMSE is increased by 65.7 per cent and the MAE is increased by 70.6 per cent.

Table 4: Solutions for wing box use case

Panel No.	Solution Liu (BPR = 0.9)		BPR = 0.8		BPR = 0.9		BPR = 1.0	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
12	0.1054	0.0575	0.1054	0.0575	0.1054	0.0575	0.1746	0.0981
15	0.1000	0.0526	0.0777	0.0356	0.0777	0.0356	0.0776	0.0356
18	0.1917	0.0956	0.0913	0.0502	0.1009	0.0675	0.0991	0.0628
combined	0.1389	0.0668	0.0921	0.0478	0.0955	0.0535	0.1243	0.0665

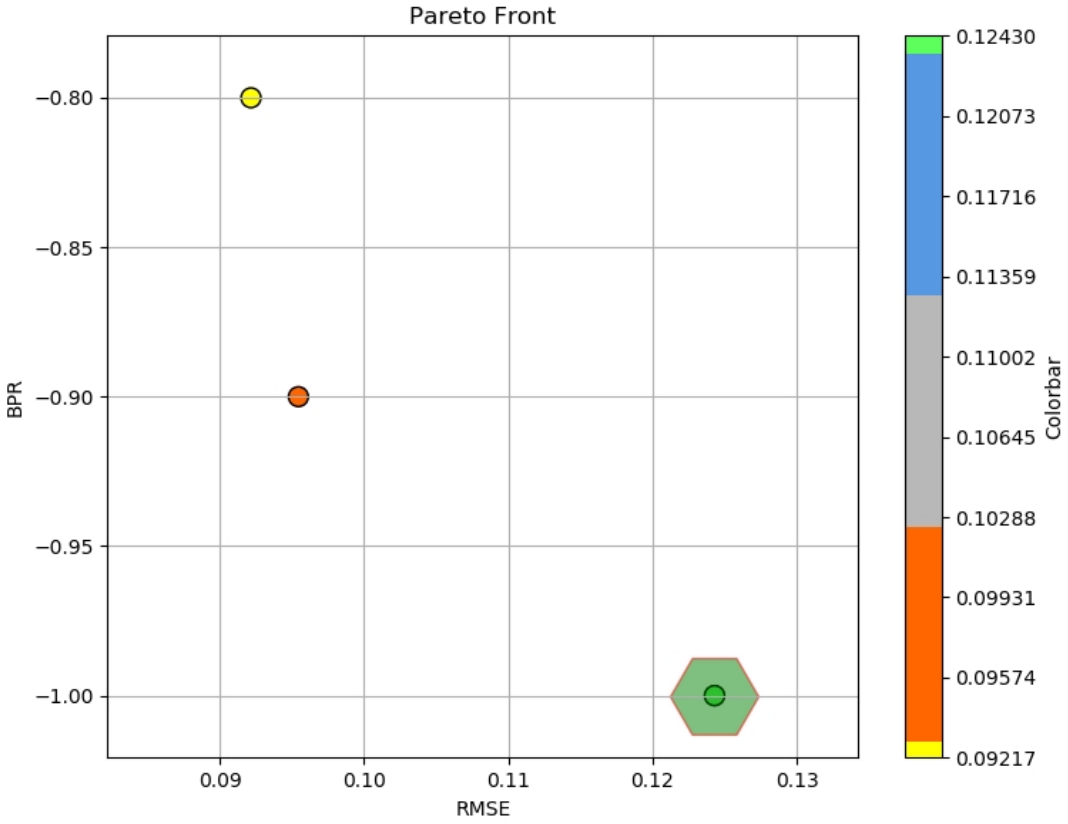


Figure 32: Pareto Front of wing box use case

RMSE influence on reserve factor for buckling load cases of plates

Until now, the RMSE values have allowed for an analysis of the disparity between desired and actual lamination parameters, enabling assessments of stiffness improvements for non-perfectly blended laminates. However, for engineering purposes, it would also be beneficial to know how the RMSE values correlate with enhancements in the reserve factor for load cases of laminate plates to better contextualise the RMSE values for laminate optimizations.
The structural optimization of the wing box use case includes a calculation of the critical buckling load for the panels on the top and bottom of the box which are then compared to the emerging loads for a given load case resulting in a reserve factor for buckling. For an orthotropic plate under longitudinal compression (figure 33), the critical buckling load can be calculated with

$$N_{x_{crit}} = k \cdot 2 \left(\frac{\pi}{b}\right)^2 \left[\sqrt{D_{11}D_{22}} + D_{12} + 2 \cdot D_{33}\right] \tag{14}$$

$$RF = \frac{N_{x_{crit}}}{N_{applied}} \tag{15}$$

where k is a knockdown factor for torsional coupling based on Rohwer from HSB-45114-01 [28] and can be calculated with

$$k = 1 - \frac{1.4 \cdot D_{13}D_{23}}{\sqrt{\left(D_{12} + 2 \cdot D_{33}\right)\left(D_{11}D_{22}\right)^{3/2}}} \tag{16}$$

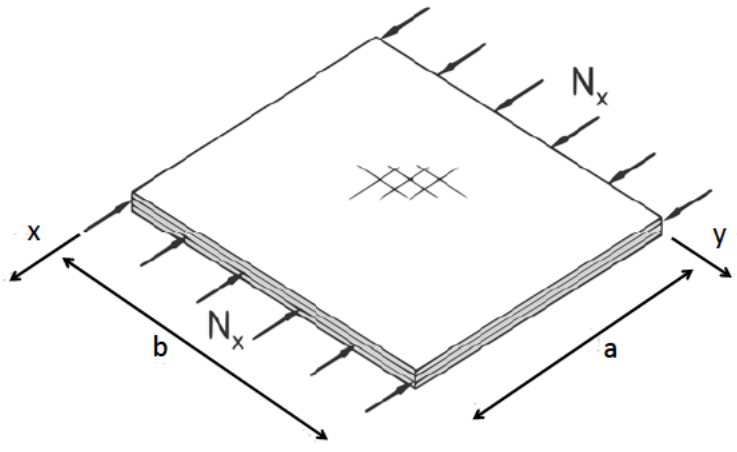


Figure 33: Orthotropic plate buckling under longitudinal compression

For the plots in figure 34, the reserve factor for buckling for a 12-ply panel from the wing box use case is calculated for every possible orthotropic stacking with ply orientation 0, 45, -45 and 90 degrees. The compression load is scaled such that the optimal stacking sequence would result in a reserve factor close to 1.5. For every other stacking, the RMSE between the optimal stacking and given stacking is calculated for the D-Matrix entries and the D-Lamination parameters and then plotted together with their correlating reserve factor.

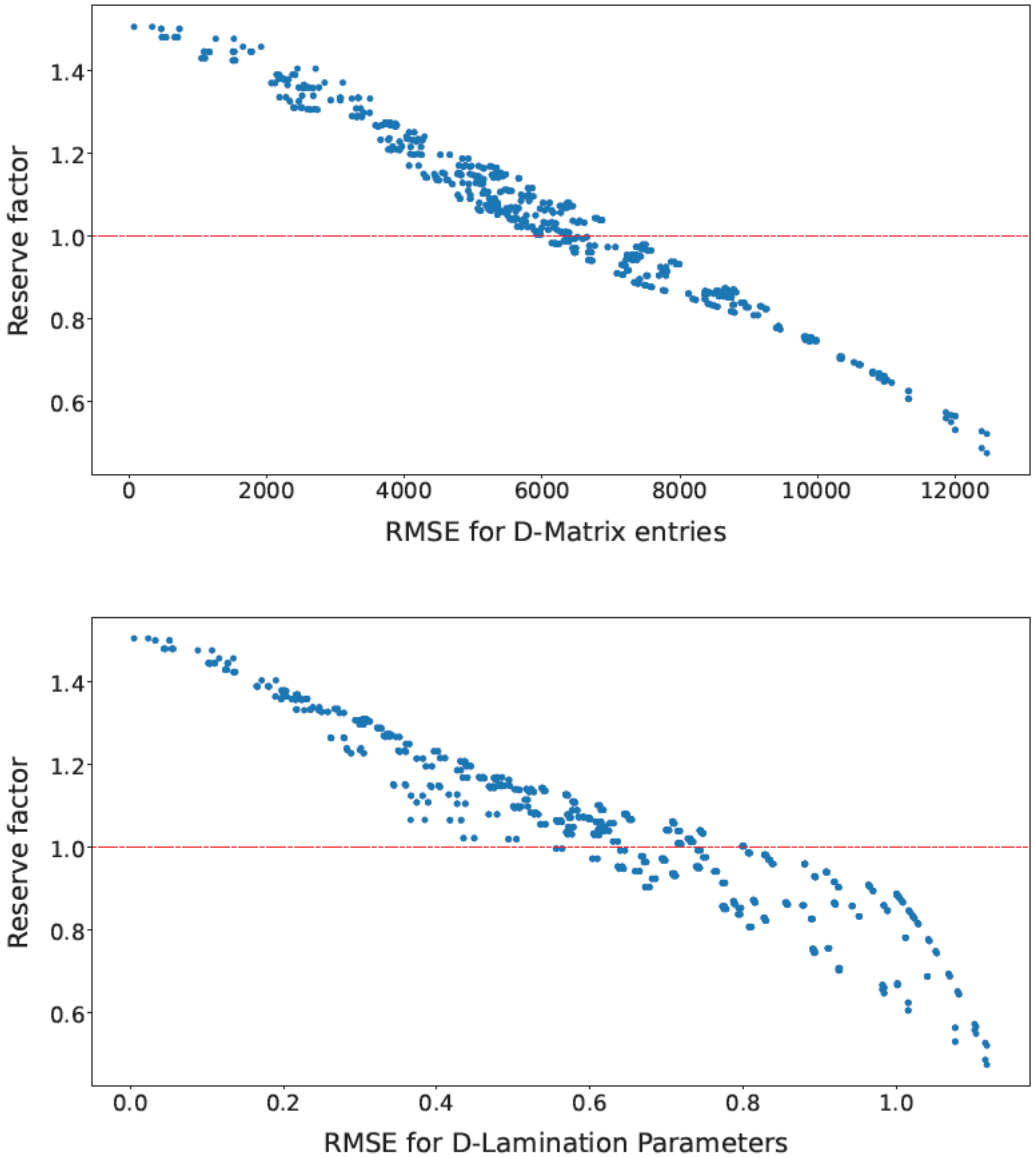


Figure 34: Reserve factor for buckling load case with different layups and their corresponding RMSE for the D-Matrix entries (top) and D-Matrix Lamination Parameters (bottom)

It can be seen that with increasing RMSE, the reserve factor is on average further min-

imized. However, the data points are not located on are linear line but are slightly scattered. The RMSE values from the D-Matrix entries are more scattered for low and middle RMSE values. The scattering is minimized for the higher RMSE values. The RMSE values for the D-Lamination parameters show a reversed behaviour. The scattering is minimized for for low RMSE values and increases with for higher RMSE values.
The RMSE values for the Pareto front example (table 1) and the wing box use cases example (table 6) are calculated with the full lamination parameter set (all 12 LPs). Using the full set for the buckling load case would cause an even greater scattering of the data points, which is shown in appendix B.1.
Consequently, the RMSE for the Pareto front example and the wing box use cases are determined again but this time only for the D-Lamination parameters. The results are listed in table 5 and 6. Looking at the results for the Pareto front example, the RMSE values for the lowest and highest BPR are further apart. The same applies to the RMSE values from the wing box use case with a BPR of 0.8 and 1.0.

Table 5: Pareto Points with D-Matrix LPs (RMSE values)

Section	BPR = 0.3	BPR = 0.6	BPR = 0.8
combined	0.1270	0.2507	0.4491

Table 6: Wing box solutions with D-Matrix LPs (RMSE values)

Panel No.	BPR = 0.8	BPR = 0.9	BPR = 1.0
12	0.1435	0.1435	0.2377
15	0.0793	0.0793	0.0793
18	0.0613	0.1544	0.1508
combined	0.1011	0.1300	0.1666

9 Discussion of the results

Blending Method

The practical applications have demonstrated that the blending method is effective in efficiently blending small to midsize laminates within a reasonable timeframe, thanks to its adept search algorithms. The analysis capability in terms of ply number can be doubled when dealing with symmetric laminates and further increased by having the blending method search for continuous ply packages rather than individual plies. These ply packages would consist of more than one ply stacked together with predefined ply angles. For instance, ply packages comprising two plies can enable the analysis of up to 100 plies in under a minute instead of laminates of 50 plies with individual plies. For comparison, laminates in wing structures can have a thickness of 40 layers near the tip to 250 layers close to the root of the wing [26]. However, it's important to note that opting for ply packages may limit the achievable stackings compared to using single plies.
While the addition of plies to a specific section results in an exponential increase in computation for the blending method, as discussed in chapter 7, incorporating more sections into the blending problem itself does not exhibit an exponential nature. This is because blending solutions between individual sections can be straightforwardly combined to yield a single blending solution for the entire structure. Consequently, the total computation time for the blending analysis is the sum of the computation times for blending between each section. This method's characteristic positions it favourably for structural optimization problems involving small laminate sizes but with numerous design regions.
The computational effort to find all blending solutions is further decreased if the number of ply orientations present in the stackings is increased. On the other hand, an analysis of extreme-like stackings (stackings with one ply orientation) leads to an increase in computing time. However, it can be argued that the trend moves towards generating laminate stackings with ply angles other than the classic 0, +- 45 and 90 ply angles to achieve better stiffness characteristics [27], [21]. This could therefore excuse the impaired performance of extreme-like laminates.
All blending solutions, if generated from independent laminate sections, should also be investigated for compliance with design rules. As the blending criterion focuses solely on the appearance of continuous plies, there is no influence of disregarding certain design rules on the BPR. For example, the multiple-section blending example 28 shows that there are some dropped edges from plies in contact with each other. Dropped edges in physical contact should generally be avoided because they are difficult to produce using tape-laying or tow-placement technology. Besides, local overlapping of plies due to

differences in orientation, makes these regions very unattractive [3].

Optimization

Even though MIDACO has a built-in backtracking line search for fast local convergence [22], Ant Colony Optimizer perform much better for global searches and has some shortcomings in the convergence speed and solution accuracy when dealing with a large amount of data [5],[34]. The results from the blending problem from figure 29 greatly support this. After a relatively small amount of iterations, MIDACO was able to identify that the stackings should not contain ±45-degree ply angles and the sequence of 0 and 90-degree plies matches also the desired sequence very closely. After that, it takes a long time to converge to the optimal solution.
Besides the problem of converging to the global optimum in complex problem spaces with multiple local optima, ACOs performance is sensitive to the appropriate tuning of its parameters. The choice of parameters, such as the pheromone evaporation rate and exploration-exploitation trade-off, can significantly impact the algorithm's convergence speed and solution quality [9],[18]. While some recommendations from the manual for MIDACO showed some improvements in convergence speed and solution quality, a significant increase could not be achieved. It is therefore possible that parameters for the blending problems are still not adequately tuned.
To achieve these significant improvements, one idea was to set the plyshares for the laminate section that can be calculated from the lamination parameter for the $[A]$ Matrix (if only 0, ±45 and 90 degre plies exist) as constraints for the optimization. To evaluate the violation of constraints, MIDACO uses a penalty method that transforms a constraint problem into a series of unconstrained problems by replacing the original objective function with a penalty function which is a weighted sum (or product) of the original objective function and the constraint violations [22]. Due to this evaluation method, it was expected that this would allow the best possible solution to stand out even more clearly. Unfortunately, the significant improvements could not be achieved. This observation can lead to the assumption that there are only a few or even no local optima in the vicinity of the optimal solution that do not fulfil the constraints, hence the problem of convergence is present.

The results from the Pareto Front example showed that the blending method is responsible for most of the computational effort of the optimization. Without the method, a laminate with three sections and a symmetric ply sequence with 20 plies could be analyzed in terms of LPs discrapency with 304000 iterations in 102 seconds. With the blending method, the performance is reduced to 44000 iterations in 764 seconds. From profiling the code,

it could be identified that in general, the search for solutions takes up the most amount of time. And as the blending evaluation occurs for every iteration step, this time adds up for the whole optimization. The directed search method can significantly minimize this time by stopping the search for blending solutions after a desired amount of solutions. However, it was observed that when reducing the number of desired solutions, most of the time from the blending method was used to create the graph object from the blending matrix. The time for creation is generally small but significant when the blending method is applied at every iteration in the optimization.
Regarding the Pareto front in 30, it is interesting to see that the non-dominated Pareto points are not on a straight line between the point with the lowest and highest BPR. They rather form a curve that is bent towards the lower left part of the diagram. This indicates that there is indeed an optimal solution if the BPR and RMSE are both of equal value.
Reviewing the stacking s of the Pareto front example of the left section for the three solutions in the appendix A.1, it can be identified that for every solution, each stacking has in total four plus or minus 45 degree plies. For the solutions with a BPR of 0.6, two of these plies are located right in the middle of the stacking whereas for the other two solutions, these two plies are located further outwards. The influence of the appearance of a 45 degree ply in stackings on the lamination parameters for the $[D]$ Matrix is maximized if the ply is located further outwards than inwards in stacking due to the exponential weighting term in the computation of the lamination parameters that weights the trigonometric functions. This explains why the RMSE is minimized for the middle section from the solution with BPR of 0.3 to a BPR of 0.6 and then again increased for the solution with a BPR of 0.8.

Compared to the results from the Pareto front example, the results from the wing box use case show a smaller improvement in the combined stiffness values for a reduction of blending. In addition, the location of the Pareto points on the Pareto front shows that the best tradeoff between blending and discrepancy to optimal stiffness values already lies close to a perfectly blended laminate.
Upon examining the RMSE values of the three panels individually, it becomes evident that panel number 12 primarily drives the enhancements in overall stiffness. Inspection of the stacking sequences in the appendix (see A.2) indicates that panel number 12 has preferably only two 90-degree plies near the centre. However, to achieve 100 per cent blending, two extra 90-degree plies need to be incorporated to match the stacking sequence from panel number 15. While the averaged improvement for all three panels is small, the individual improvement for panel three with 70.6 per cent MAE and 65.7 RMSE is remarkably high.

It is interesting to see that even though the blending solution from Liu et al. [12] for the bottom skin doesn't have a BPR of 1.0 , the combined RMSE is still greater than the RMSE for a BPR of 1.0 in this analysis. This is probably due to the fact that the laminates from Liu et al. are all balanced, limiting the possible stackings for the three panels.

The increased scattering of the datapoints of the reserve factor for buckling load case 34 makes it difficult to give an exact statement about the improvement of the reserve factor when the RMSE is reduced. The scattering itself is probably caused by the uneven influence of the D-Matrix entries on the reserve factor while the RMSE calculation treats all entries equally (14,16). Using the D-Lamination parameters instead of the D-Matrix entries directly for the determination of the RMSE further worsens the scattering effect.

10 Conclusion and Outlook

In conclusion, this thesis presents an innovative blending method designed to blend independent laminate sections effectively. Notably, this method stands out by performing blending as a post-processing step, eliminating the necessity for defining blending constraints in the retransformation of lamination parameters into discrete stackings. The core concept revolves around creating a blending matrix that encapsulates information about continuous plies between stackings. This matrix is then transformed into a graph object to explore diverse blending possibilities. The implementation of an enhanced depth-first search algorithm, tailored to prioritize solutions with a higher prevalence of continuous plies, optimizes the discovery of the most effective blending configurations. The application of the method on several blending problems revealed its limitation to blending layups with over 25 individual plies in under a minute but on the other hand showed that the method performs very well if instead the number of plies, the number of sections is increased.

The diversity introduced by the blending method necessitates the development of a blending criterion, allowing for the assessment of laminates based on their blending characteristics. The criterion classifies laminate structures based on the individual section size between a perfectly blended and non-blended structure.

The primary motivation behind postponing the blending process was to better address the stiffness requirements for individual design regions within a structural optimization framework by moving away from optimal blended structures. To demonstrate the possibility of utilizing the blending method in stacking sequence retrieval problems and to gauge the effectiveness of the postponing approach, the blending method was combined with the ant colony optimizer MIDACO. The synthesis of the blending method and MIDACO aimed to showcase the retransformation of lamination parameters into a blended laminate structure and to analyze the discrepancy between desired lamination parameters and those produced for both perfectly blended and non-perfectly blended laminates.

Indeed, the analysis indicated improvements for the stiffness requirements, which resulted from a reduced blending degree. However, the extent of these improvements varies depending on the specific problem at hand. As anticipated, problems where adjacent sections require dissimilar ply angles in their stackings exhibited significant enhancements in overall stiffness. Conversely, for real-world problems where adjacent sections had less disparity in their requirements for certain ply angles, these overall improvements were less pronounced. But, as seen in the benchmark problem, improvements made in specific sections may be overshadowed by the averaging process, underscoring the importance of closely examining the results.

This thesis concentrated on delivering a computationally efficient solution for the blending problem. A comprehensive exploration and implementation of the following ideas could not be performed within the timeframe of the master thesis and can therefore be reviewed as recommendations for future research.

1. Improvements for the search algorithm: The directed DFS method has been proven to be very useful for finding optimal blending solutions first. The search method could however be extended from an uninformed search method to an informed search method that includes a heuristic function to search in an ever more targeted manner and thus possibly reduce the runtime. It would be further beneficial to optimize the design of the graph object in a more efficient way or even dynamically while performing the search. This would ultimately extend the capabilities of the blending method to blended laminates with a greater amount of plies.

2. Laminate database: Another approach to tackle the stacking sequence retrieval problem from optimized lamination parameters is the usage of laminate databases. These databases have pre-designed laminate stacking sequences in-store that can already adhere to certain design rules. The blending method could be incorporated in the search for optimal stackings for a laminate.

References

[1] David B. Adams, Layne T. Watson, Zafer Gürdal, and Christine M. Anderson-Cook. "Genetic algorithm optimization and blending of composite laminates by locally reducing laminate thickness". In: *Advances in Engineering Software* 35.1 (2004), pp. 35–43. ISSN: 0965-9978. DOI: https://doi.org/10.1016/j.advengsoft.2003.09.001. URL: https://www.sciencedirect.com/science/article/pii/S0965997803001236.

[2] Mark Bloomfield, J. Herencia, and Paul Weaver. "Optimisation of Anisotropic Composite Plates Incorporating Non-Conventional Ply Orientations". In: Apr. 2008. ISBN: 978-1-60086-993-8. DOI: 10.2514/6.2008-1918.

[3] Julien van Campen, Omprakash Seresta, Mostafa Abdalla, and Zafer Gürdal. "General Blending Definitions for Stacking Sequence Design of Composite Laminate Structures". In: 2008. DOI: 10.2514/6.2008-1798. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2008-1798. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2008-1798.

[4] *CO2 EINGESPART MIT LEICHTIGKEIT*. URL: https://www.klimaschutz-portal.aero/verbrauch-senken/am-flugzeug/gewicht-einsparen/ (visited on 11/01/2024).

[5] Xiaolin Dai, Shuai Long, Zhiwen Zhang, and Dawei Gong. "Mobile Robot Path Planning Based on Ant Colony Algorithm With A* Heuristic Method". In: *Frontiers in Neurorobotics* 13 (2019). ISSN: 1662-5218. DOI: 10.3389/fnbot.2019.00015. URL: https://www.frontiersin.org/articles/10.3389/fnbot.2019.00015.

[6] Thanh N. Huynh and Jaehong Lee. "Optimal thickness distribution design for blending composite laminates using buckling factor prediction". In: *Composite Structures* 327 (2024), p. 117693. ISSN: 0263-8223. DOI: https://doi.org/10.1016/j.compstruct.2023.117693. URL: https://www.sciencedirect.com/science/article/pii/S0263822323010395.

[7] Robert M. Jonas. *Mechanics Of Composite Materials*. Taylor & Francis, 1999.

[8] Birna P. Kristinsdottir, Zelda B. Zabinsky, Mark E. Tuttle, and Sudipto Neogi. "Optimal design of large composite panels with varying loads". In: *Composite Structures* 51.1 (2001), pp. 93–102. ISSN: 0263-8223. DOI: https://doi.org/10.1016/S0263-8223(00)00128-8. URL: https://www.sciencedirect.com/science/article/pii/S0263822300001288.

[9] Sebastian Lague. *Coding Adventure: Ant and Slime Simulations*. Youtube. 2022. URL: https://www.youtube.com/watch?v=X-iSQQgOd1A&t=76s.

[10] Boyang Liu and Raphael Haftka. "Composite wing structural design optimization with continuity constraints". In: *19th AIAA Applied Aerodynamics Conference*. 2001. DOI: 10.2514/6.2001-1205. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2001-1205. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2001-1205.

[11] Dianzi Liu, Vassili Toropov, David Barton, and Osvaldo Querin. "Weight and mechanical performance optimization of blended composite wing panels using lamination parameters". In: *Structural and Multidisciplinary Optimization* 52 (May 2015). DOI: 10.1007/s00158-015-1244-x.

[12] Dianzi Liu, Vassili Toropov, Osvaldo Querin, and David Barton. "Bilevel Optimization of Blended Composite Wing Panels". In: vol. 48. May 2009. ISBN: 978-1-60086-975-4. DOI: 10.2514/6.2009-2182.

[13] Terence Macquart, Marco T. Bordogna, Paul Lancelot, and Roeland De Breuker. "Derivation and application of blending constraints in lamination parameter space for composite optimisation". In: *Composite Structures* 135 (2016), pp. 224–235. ISSN: 0263-8223. DOI: https://doi.org/10.1016/j.compstruct.2015.09.016. URL: https://www.sciencedirect.com/science/article/pii/S0263822315008521.

[14] Marco Montemurro. "An extension of the polar method to the First-order Shear Deformation Theory of laminates". In: *Composite Structures* 127 (2015), pp. 328–339. ISSN: 0263-8223. DOI: https://doi.org/10.1016/j.compstruct.2015.03.025. URL: https://www.sciencedirect.com/science/article/pii/S026382231500197X.

[15] Mike Müller and Dennis Freese. *Genetische Algorithmen*. Fachhochschule Köln. URL: https://www.gm.th-koeln.de/~hk/lehre/ala/ws0506/Praktikum/Projekt/B_rot/GenetischeAlgorithmen.pdf (visited on 07/09/2023).

[16] Michael Chun-Yung Niu. *AIRFRAME STRUCTURAL DESIGN*. Hong Kong Conmilit Press limited; 1rd edition, 1992. ISBN: 978-9627128069.

[17] Enrico Panettieri, Marco Montemurro, and Anita Catapano. "Blending constraints for composite laminates in polar parameters space". In: *Composites Part B: Engineering* 168 (2019), pp. 448–457. ISSN: 1359-8368. DOI: https://doi.org/10.1016/j.compositesb.2019.03.040. URL: https://www.sciencedirect.com/science/article/pii/S1359836819304494.

[18] Rakesh Patel. *What is the difference between BFS and DFS algorithms*. URL: https://www.upperinc.com/glossary/route-optimization/ant-colony-optimization/ (visited on 01/30/2024).

[19] Marco Picchi Scardaoni and Marco Montemurro. "A general global-local modelling framework for the deterministic optimisation of composite structures". In: *Structural and Multidisciplinary Optimization* 62 (Oct. 2020). DOI: 10.1007/s00158-020-02586-4.

[20] *Reducing emissions from aviation*. URL: https://climate.ec.europa.eu/eu-action/transport/reducing-emissions%20aviation_en#:~:text=The%20aviation%20sector%20creates%2013.9,in%20the%20top%2010%20emitters. (visited on 01/18/2024).

[21] Marco Picchi Scardaoni, Marco Montemurro, Enrico Panettieri, and Anita Catapano. "New blending constraints and a stack-recovery strategy for the multi-scale design of composite laminates". In: *Structural and Multidisciplinary Optimization* 63 (2021), pp. 741–766. DOI: https://doi.org/10.1007/s00158-020-02725-x. URL: https://link.springer.com/article/10.1007/s00158-020-02725-x#citeas.

[22] M. Schlueter, S. Erb, M. Gerdts, S. Kemble, and J.J. Ruckmann. "MIDACO on MINLP Space Applications". In: *Advances in Space Research* 51.7 (2013), pp. 1116–1131. DOI: 10.1016/j.asr.2012.11.006.

[23] Bianca Selzam. *Genetische Algorithmen*. TU Dortmund. URL: https://ls11-www.cs.tu-dortmund.de/lehre/SoSe03/PG431/Ausarbeitungen/GA_Selzam.pdf (visited on 07/09/2023).

[24] Omprakash Seresta, Zafer Gürdal, David B. Adams, and Layne T. Watson. "Optimal design of composite wing structures with blended laminates". In: *Composites Part B: Engineering* 38.4 (2007), pp. 469–480. ISSN: 1359-8368. DOI: https://doi.org/10.1016/j.compositesb.2006.08.005. URL: https://www.sciencedirect.com/science/article/pii/S1359836806001211.

[25] Grant Soremekun, Zafer Gürdal, Christos Kassapoglou, and Darryl Toni. "Stacking sequence blending of multiple composite laminates using genetic algorithms". In: *Composite Structures* 56.1 (2002), pp. 53–62. ISSN: 0263-8223. DOI: https://doi.org/10.1016/S0263-8223(01)00185-4. URL: https://www.sciencedirect.com/science/article/pii/S0263822301001854.

[26] Andreas Spaeth. *Flügelgebäck aus dem Riesen-Ofen*. 2016. URL: https://www.nzz.ch/mobilitaet/luftfahrt/leichtbau-in-der-aviatik-fluegelgebaeck-aus-dem-riesen-ofen-ld.122875.

[27] Moritz Sprengholz, Hendrik Traub, Michael Sinapius, Sascha Dähne, and Christian Hühne. “Rapid transformation of lamination parameters into stacking sequences”. In: *Composite Structures* 276 (2021), p. 114514. ISSN: 0263-8223. DOI: https://doi.org/10.1016/j.compstruct.2021.114514. URL: https://www.sciencedirect.com/science/article/pii/S0263822321009764.

[28] Working group Structural Analysis. *Handbook for structural analysis (HSB)*. 2009.

[29] Stephen Wei-Lun Tsai and N.J. Pagano. “Invariant Properties of Composite Materials”. In: *Composite Materials Workshop* (1968), pp. 223–253.

[30] *UltraFan The Ultimate TurboFan*. URL: https://www.rolls-royce.com/innovation/ultrafan.aspx (visited on 02/27/2024).

[31] Edgar Werthen and Sascha Dähne. “Design rules consideration within optimization of composite structures using lamination parameters”. In: (July 2016).

[32] Wikipedia. *Graph (Graphentheorie) — Wikipedia, The Free Encyclopedia*. http://de.wikipedia.org/w/index.php?title=Graph%20(Graphentheorie)&oldid=237321631. 2023. (Visited on 10/19/2023).

[33] Wikipedia. *Kohlenstofffaserverstärkter Kunststoff — Wikipedia, The Free Encyclopedia*. http://de.wikipedia.org/w/index.php?title=Kohlenstofffaserverst%C3%A4rkter%20Kunststoff&oldid=240810717. 2024. (Visited on 01/16/2024).

[34] Jin Yu, Xiaoming You, and Sheng Liu. “A heterogeneous guided ant colony algorithm based on space explosion and long–short memory”. In: *Applied Soft Computing* 113 (2021), p. 107991. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2021.107991. URL: https://www.sciencedirect.com/science/article/pii/S1568494621009133.

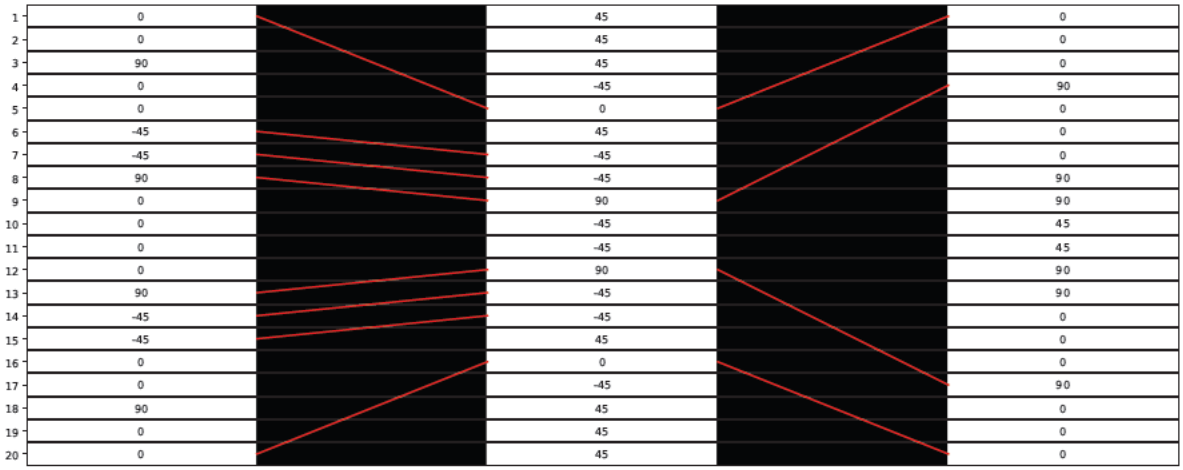
[35] Jiani Zeng, Zhengdong Huang, Yongfu Chen, Wei Liu, and Shanshan Chu. “A simulated annealing approach for optimizing composite structures blended with multiple stacking sequence tables”. In: *Structural and Multidisciplinary Optimization* 60 (Aug. 2019). DOI: 10.1007/s00158-019-02223-9.

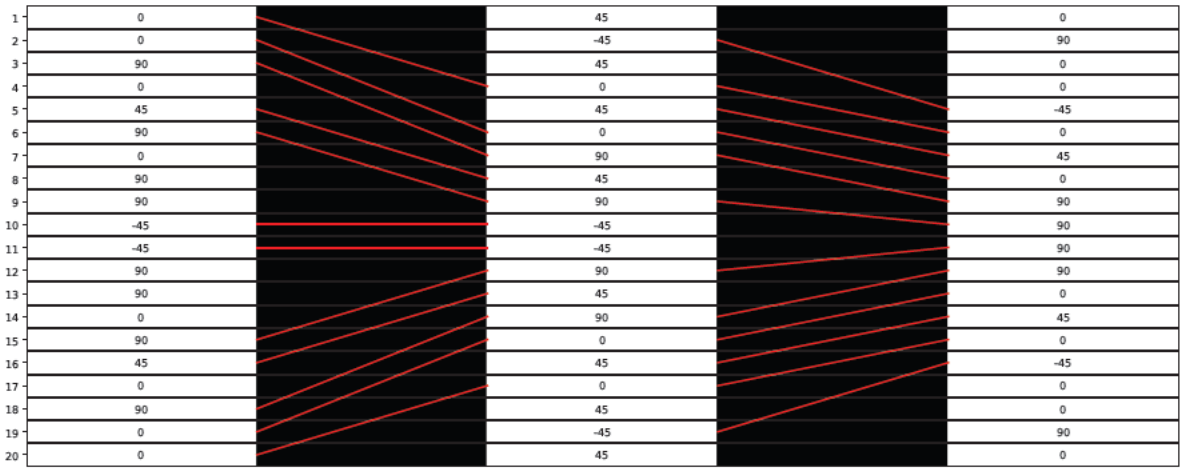
Appendices

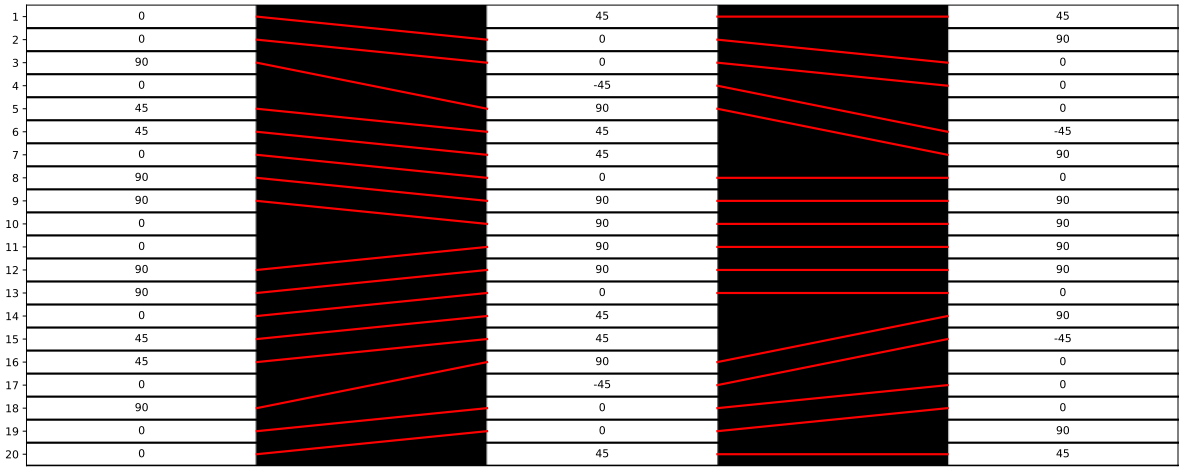
A Blending solutions

A.1 Pareto front example

Blending solutions for laminates with a BPR of 0.3 (top), 0.6 (middle) and 0.8 (bottom).

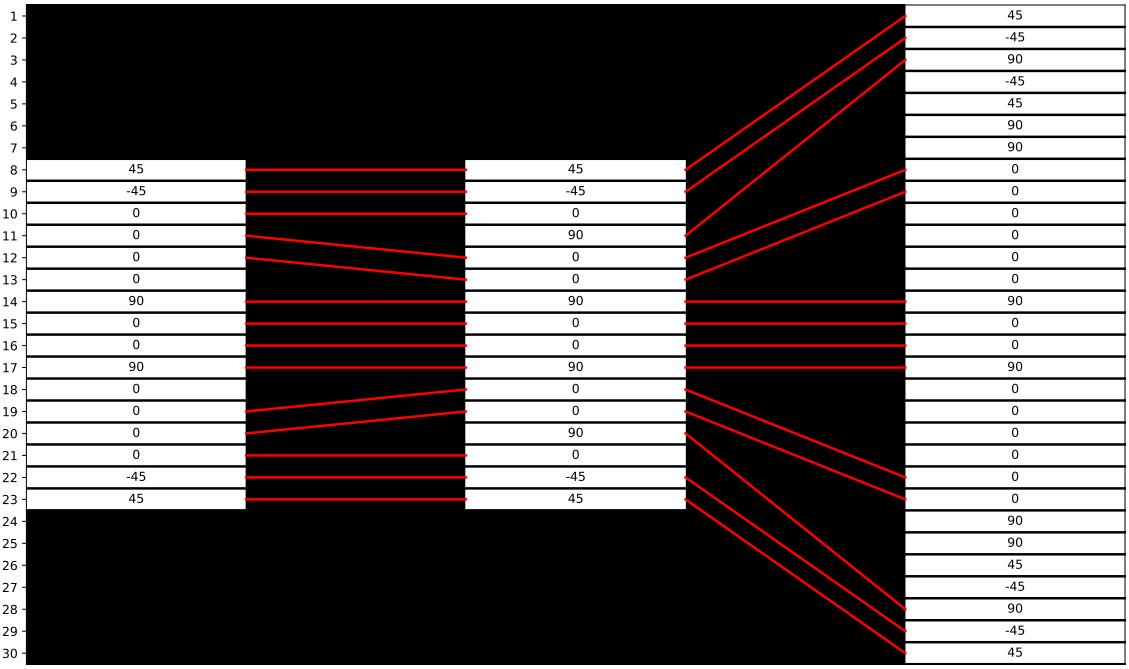


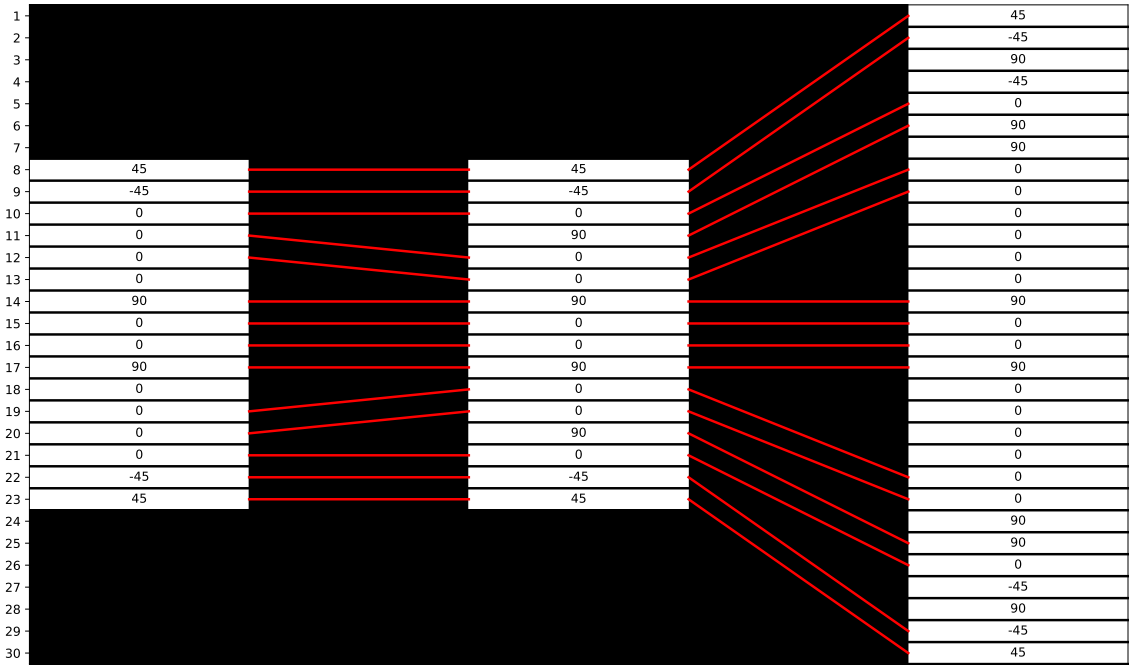


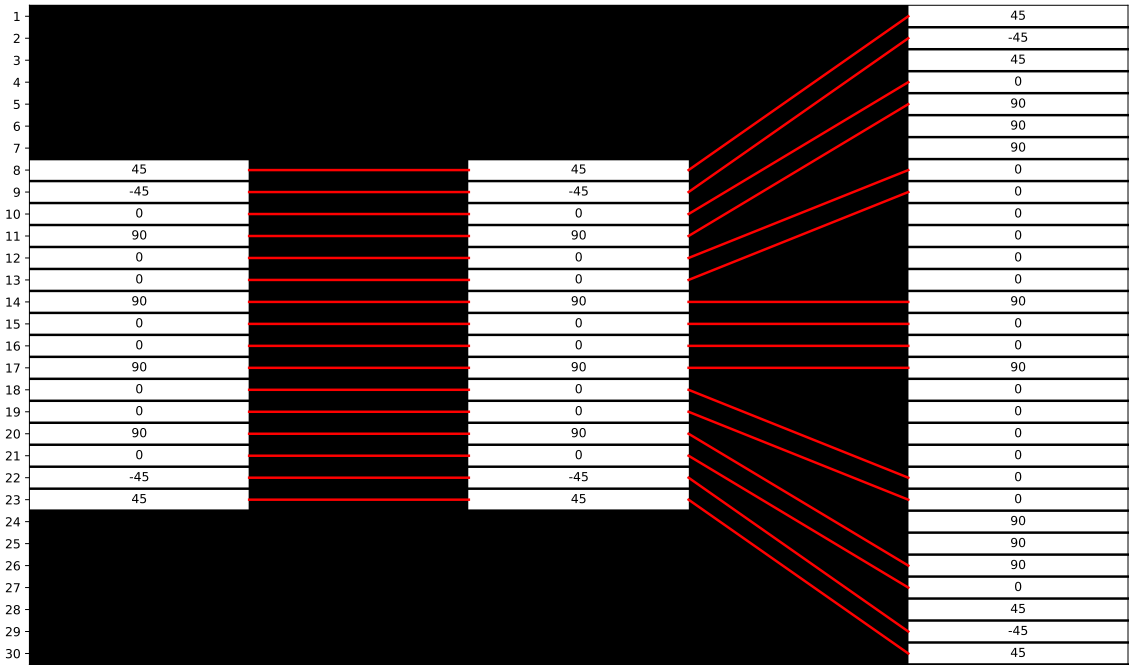


A.2 Wing box use case

Blending solutions for wing box use case with a BPR of 0.8 (top), 0.9 (middle) and 1.0 (bottom).



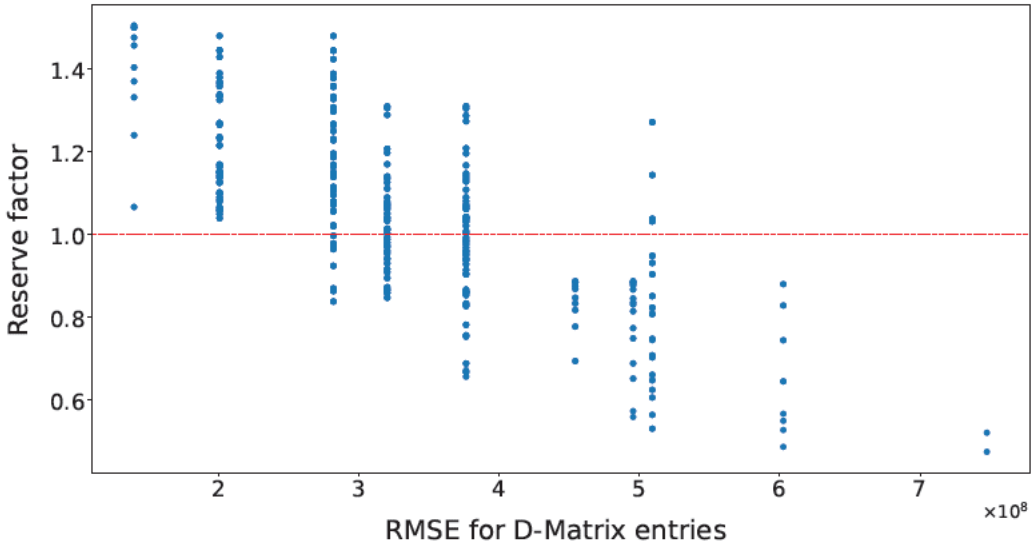

1 45
2 -45
3 90
4 -45
5 0
6 90
7 90
8 45 45 0
9 -45 -45 0
10 0 0 0
11 0 90 0
12 0 0 0
13 0 0 0
14 90 90 90
15 0 0 0
16 0 0 0
17 90 90 90
18 0 0 0
19 0 0 0
20 0 90 0
21 0 0 0
22 -45 -45 0
23 45 45 0
24 90
25 90
26 0
27 -45
28 90
29 -45
30 45

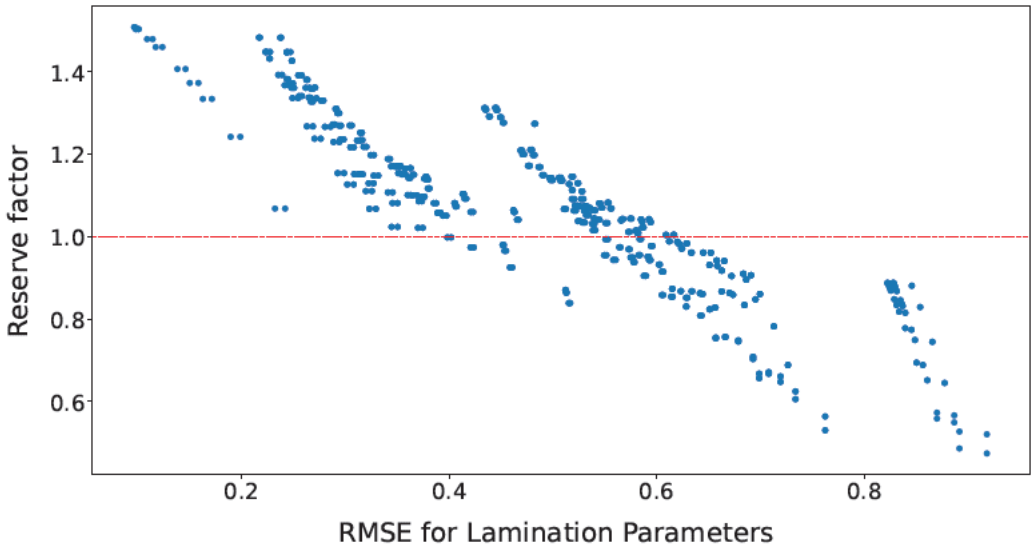

1 45
2 -45
3 45
4 0
5 90
6 90
7 90
8 45 45 0
9 -45 -45 0
10 0 0 0
11 90 90 0
12 0 0 0
13 0 0 0
14 90 90 90
15 0 0 0
16 0 0 0
17 90 90 90
18 0 0 0
19 0 0 0
20 90 90 0
21 0 0 0
22 -45 -45 0
23 45 45 0
24 90
25 90
26 90
27 0
28 45
29 -45
30 45

B Charts and Plots

B.1 Buckling Load Case

Reserve factor for buckling load case with different layups and their corresponding RMSE for the ABD-Matrix entries (top) and Lamination Parameters (bottom)





C Theoretical Background

C.1 Relation between ABD-Matrix entries and Lamination Parameters

$$\begin{Bmatrix} A_{11} \\ A_{22} \\ A_{12} \\ A_{66} \\ A_{16} \\ A_{26} \end{Bmatrix} = T \cdot \begin{bmatrix} 1 & V_1^A & V_2^A & 0 & 0 \\ 1 & -V_1^A & V_2^A & 0 & 0 \\ 0 & 0 & -V_2^A & 1 & 0 \\ 0 & 0 & -V_2^A & 0 & 1 \\ 0 & V_3^A/2 & V_4^A & 0 & 0 \\ 0 & V_3^A/2 & -V_4^A & 0 & 0 \end{bmatrix} \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{Bmatrix} \tag{17}$$

$$\begin{Bmatrix} B_{11} \\ B_{22} \\ B_{12} \\ B_{66} \\ B_{16} \\ B_{26} \end{Bmatrix} = \frac{T^2}{4} \cdot \begin{bmatrix} 1 & V_1^B & V_2^B & 0 & 0 \\ 1 & -V_1^B & V_2^B & 0 & 0 \\ 0 & 0 & -V_2^B & 1 & 0 \\ 0 & 0 & -V_2^B & 0 & 1 \\ 0 & V_3^B/2 & V_4^B & 0 & 0 \\ 0 & V_3^B/2 & -V_4^B & 0 & 0 \end{bmatrix} \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{Bmatrix} \tag{18}$$

$$\begin{Bmatrix} D_{11} \\ D_{22} \\ D_{12} \\ D_{66} \\ D_{16} \\ D_{26} \end{Bmatrix} = \frac{T^3}{12} \cdot \begin{bmatrix} 1 & V_1^D & V_2^D & 0 & 0 \\ 1 & -V_1^D & V_2^D & 0 & 0 \\ 0 & 0 & -V_2^D & 1 & 0 \\ 0 & 0 & -V_2^D & 0 & 1 \\ 0 & V_3^D/2 & V_4^D & 0 & 0 \\ 0 & V_3^D/2 & -V_4^D & 0 & 0 \end{bmatrix} \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{Bmatrix} \tag{19}$$

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Gemäß der Allgemeinen Prüfungs- und Studienordnung ist zusammen mit der Abschlussarbeit eine schriftliche Erklärung abzugeben, in der der Studierende bestätigt, dass die Abschlussarbeit „– bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit [(§ 18 Abs. 1 APSO-TI-BM bzw. § 21 Abs. 1 APSO-INGI)] – ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt wurden. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich zu machen."

Quelle: § 16 Abs. 5 APSO-TI-BM bzw. § 15 Abs. 6 APSO-INGI

Dieses Blatt, mit der folgenden Erklärung, ist nach Fertigstellung der Abschlussarbeit durch den Studierenden auszufüllen und jeweils mit Originalunterschrift als letztes Blatt in das Prüfungsexemplar der Abschlussarbeit einzubinden.
Eine unrichtig abgegebene Erklärung kann -auch nachträglich- zur Ungültigkeit des Studienabschlusses führen.

Erklärung zur selbstständigen Bearbeitung der Arbeit

Hiermit versichere ich,

Name: Westphal

Vorname: Kolja

dass ich die vorliegende Masterarbeit ☐ bzw. bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit – mit dem Thema:

Entwicklung einer Methode zur Generierung und Bewertung von geblendeten FVK-Lagenaufbauten

ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

- die folgende Aussage ist bei Gruppenarbeiten auszufüllen und entfällt bei Einzelarbeiten -

Die Kennzeichnung der von mir erstellten und verantworteten Teile der -bitte auswählen- ist erfolgt durch:

Ort	Datum	Unterschrift im Original