Spacecraft Satellite Panel Composite Analysis

Luca Sacchetto, Mushashi Nozaki, Daniel Quarshie

[1] University of San Diego, CA 92110, USA

Abstract. NASA's CubeSat Satellites, compact size and cost-efficiency have transformed access to space exploration, supporting missions from Earth observation to interplanetary research. However, the development of lightweight yet resilient structures for CubeSats remains a critical challenge, particularly in optimizing composite materials to withstand extreme launch loads. A key research question is: how can the mechanical performance and failure resistance of CubeSat composite panels be maximized under complex loading conditions? To address this, we applied Classical Laminated Plate Theory (CLPT) to analyze stresses, strains, and failure indices for various stacking sequences of composite laminates. Using MATLAB, we modeled the mechanical behavior of three different laminates, considering symmetry, ply orientation, and material properties. We employed Tsai-Wu and Maximum Strain failure criterias to assess laminate performance and identify optimal configurations. Our results demonstrated that carbon/epoxy composites with the symmetric stacking sequence of [60 -60 -45 -30 30 -30 60]s provide the best combination of strength and durability, minimizing failure indices under launch-induced forces. These findings contribute to advancing the reliability and performance of CubeSat structures, paving the way for their expanded use in demanding aerospace applications.

Keywords: Classical Laminated Plate Theory (CLPT), Composite Materials, Laminated Composite Structures, Stress and Strain Analysis, Failure Criteria, Ply Stacking Sequence, Finite Element Analysis, Composite Laminate Design, Failure Modes in Composites, Numerical and Analytical Comparison, Engineering Applications of Composites

1 Introduction

This project focuses on the analysis and design of a laminated composite structure specifically tailored for CubeSat satellites **Figure 1**, which are compact spacecraft measuring 10x10x10 cm. These small spacecraft are designed to withstand the rigorous conditions of space travel, including gravitational loads experienced during launch to Low Earth Orbit (LEO), where they encounter microgravity conditions simulating weightlessness **[1]**.

The primary goal of this project is to evaluate the mechanical performance of CubeSat panels under various loading conditions, including longitudinal tension and compression, bending moments, and torsional loads. In this experiment, we will be looking at the gravitational forces acting upon the satellite during launch from earth up to reaching LEO, while ignoring external factors like radiation and pressure. Specific gravitational accelerations during launch were determined from a NASA research paper **[1]** to be 5G in the x-direction and 10G in the y-direction, where GGG is the gravitational acceleration. Utilizing Newton's second law of motion, the force in the x and y directions present during launch could then be found, given the mass of the CubeSat, which was found to be 10 kilograms.

To translate these forces into in-plane distributed forces (N), the forces were divided by the width of each individual solar panel cell (5 mm), resulting in distributed loads of 9.81 N/mm in the x-direction and 19.62 N/mm in the y-direction. The distributed moments M were calculated by multiplying these loads by the length of each panel, yielding 245.25 N*mm in the x-direction and 490.5 N*mm in the y-direction. The shear forces and moments (xy-direction) were negligible due to a lack of these forces and moments being present given the CubeSat application and research findings of NASA **[1]**.

Using Classical Laminated Plate Theory (CLPT), stress distribution, strain responses, and potential failure mechanisms to optimize structural integrity were analyzed. Carbon-epoxy composites were selected due to their exceptional strength-to-weight ratio and ability to endure the harsh thermal and mechanical environments of space **[2]**. This analysis considers critical factors such as stacking sequences, laminate thickness, and load scenarios to maximize performance while adhering to dimensional constraints. To achieve this, the project involves computing global and local stresses and strains, determining the optimal stacking sequence, and evaluating failure through criteria such as Tsai-Wu and maximum strain analysis.

Figure 1: A CubeSat satellite operating in Low Earth Orbit (LEO)

2 Methods

2.1 Material Selection Process:

The composite material selected for this study is Carbon/Epoxy IM6G/3501-6, as detailed in **Appendix A**. This material was chosen due to its exceptional strength-to-weight ratio, which is critical for CubeSat structures that endure extreme conditions during space launches and orbital operations. Compared to alternative materials, Carbon/Epoxy IM6G/3501-6 exhibits superior longitudinal modulus (169 GPa vs. 41 GPa for E-Glass/Epoxy, a 312% increase) and tensile strength (2240 MPa vs. 1140 MPa for E-Glass/Epoxy, a 96% increase). Similarly, it shows a 19% improvement in tensile strength over Kevlar/Epoxy (1400 MPa) and a 15% higher longitudinal modulus compared to Carbon/Epoxy AS4 (147 GPa). These significant improvements further justify its selection for the application. A visual comparison of these material properties is provided below in the following two figures.

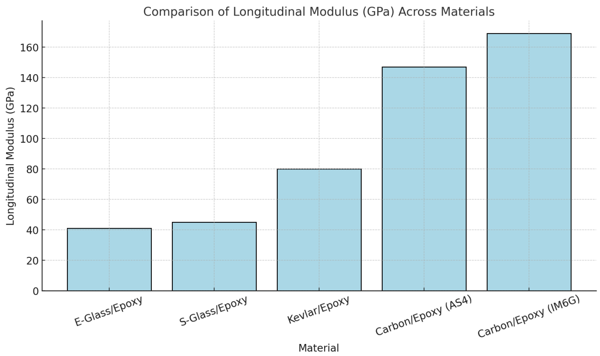

Figure 2: Material comparison of longitudinal modulus (GPa)

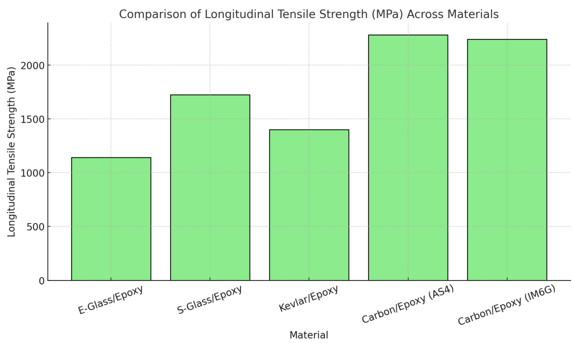

Figure 3: Material comparison of longitudinal tensile strength (MPa)

The figures above compare the longitudinal modulus and tensile strength of different composite materials. These charts illustrate the superior properties of Carbon/Epoxy IM6G/3501-6 over other materials like E-Glass/Epoxy and Kevlar/Epoxy. They highlight its significantly higher longitudinal modulus and tensile strength, supporting its selection for CubeSat applications.

2.2 Thickness and Layup Selection Process:

The laminate design adheres to a thickness limit of 3 mm, as required by the Classical Laminated Plate Theory (CLPT) for thin laminate analysis. A thinner laminate was prioritized to balance weight reduction with sufficient structural performance, as lower thickness minimizes mass which is critical for CubeSat applications where launch weight is constrained. Based on industry standards and prior studies on CubeSat panels, the average thickness of similar panels was found to be approximately 2 mm **[4]**. To ensure structural integrity and improve mechanical performance, a slightly increased thickness of 2.1 mm was selected. This thickness was determined based on the lamina thickness of 0.15 mm, allowing for 14 plies in the final layup, ensuring uniform distribution of strength and stiffness across the laminate. This lamina thickness enhances both tensile strength and stiffness, providing better resistance to the high stresses and moments experienced during launch, while still staying within the thin laminate assumptions of CLPT. This balances weight efficiency and performance, as supported by NASA's guidelines for small satellite structures **[1]**.

To optimize the stacking sequence for the complex loading case, a MATLab script was created in order to analyze 2,097,152 possible stacking sequences. This number was reached by assuming that our laminate will be symmetrical, as this simplifies the script and increases load-carrying capacity within the laminate. Taking the 8 possible stacking orientations: 0, +/- 30, +/- 45, +/- 60, 90 (degrees), we multiplied this number by 7 or half the plies to reach the number of possible stacking sequences. Once a large matrix with all the stacking sequence combinations was created, we ran a MATLAB algorithm, which is talked about in detail in **2.4 Code Structure and Functionality.** This found the most optimized stacking sequence given the Tsai-Wu and Max Strain failure criterion. The third stacking sequence analyzed within the following experiment was user defined using ply orientations of 0, +/- 45, and 90 only. This third stacking sequence will allow us to show the differences between the optimized stacking sequences and one where a human defined the stacking sequence. The specific tracking sequences used can be seen below in Figure 4.

1	[60 -60 -45 -30 30 -30 60 60 -30 30 -30 -45 -60 60]
2	[45 -60 -60 45 -45 -45 30 30 -45 -45 45 -60 -60 45]
3	[90 45 0 90 45 0 90 90 0 45 90 0 45 90]

Figure 4: Stacking Sequences Analyzed

2.3 CLPT Equations and Assumptions:

The Classical Laminated Plate Theory forms the foundation for the stress and strain analysis. The theory assumes a linear relationship between stress and strain and negligible shear deformation. The governing equations compute the global stresses and strains based on the laminate's stiffness matrices (A, B, and D) and the applied forces and moments. The material properties of each ply were input into the model, which then calculated the laminate-level response. Symmetry in the layup simplifies the plate theory equations due to the coupling matrix B becoming zero, and this further improves our laminate's performance. The assumptions of this theory include: each ply is thin compared to its in-plane dimensions; each ply behaves according to linear elasticity; the laminate is perfectly bonded; plane sections remain plane and perpendicular to the midplane after deformation; out-of-plane shear deformations are negligible. The specific equations used in CLPT analysis are presented in **Appendix E**.

The assumptions made for this project include the laminate being modeled using Classical Laminated Plate Theory (CLPT), assuming negligible shear deformation and perfectly bonded plies. Additionally, it was assumed that the CubeSat panels experience a uniform gravitational force during launch and that out-of-plane shear deformations are negligible. Finally, symmetrical stacking sequences were employed to simplify calculations when determining the laminate stacking sequence. This is due to the algorithm running the number of possible stacking orientations (8) squared to the number of unique plies in the laminate. If the laminate was not symmetrical we would have to run 8^14 or 4 trillion combinations. We could have removed the number of stacking orientations available but instead we exponentially cut down the computation time by making the ply symmetric so there would only be approximately 2 million combinations. By doing so the MATLab script was able to fully run in about 5 minutes.

2.4 Code Structure and Functionality:

The code used in the following experiment was broken up into two separate scripts for time efficiency purposes. The purpose of the first script was in order to find the two most optimal stacking sequences for the laminate given the Tsai-Wu and Max Strain failure criteria. Although the first script found the global and local stresses and determined the failure index of the plies given the two failure theories, a second MATLab script was created in order to efficiently calculate the max load, failure indices, and local stresses and strains for just the three stacking sequences we determined for the experiment. Both scripts run similarly, so the structure of the code is provided together in the following paragraph below. For a line-by-line visualization of the code, please refer to **Appendix B** and **Appendix C**.

The script begins by defining user inputs such as the material properties, engineering constants, normal and moment loads, and ply thickness, which were consistent throughout the entire laminate. Once this is accomplished, the script calculates the number of possible combinations for the stacking sequence by taking the power of the number of possible ply orientations (8) by the number of symmetric plies (7). A large matrix is created with the symmetric stacking sequences. Next, the ply location, S, and Q matrices were calculated on the laminate level. Initializing our stiffness matrices (A, B, and D) and running them through a for loop and storing the matrices was done in order to have the stiffness matrix for all possible stacking sequences. This was similarly done for the midplane and curvature matrices. Next, the global and local stresses and strains were found for all of the stacking sequences. Simultaneously, as the code found the local stresses and strains for each ply and each stacking sequence, a maximum failure index was keeping track of all plies and all stacking sequences to identify which laminate had the lowest maximum failure index. Looping over all possible combinations, this gave us the laminate with the lowest failure index, which allowed us to identify the two most optimal stacking sequences for the two failure criteria.

As previously mentioned, the second script followed the same structure as the one above but allowed us to single out and study the local stresses and strains and stiffness matrices of the three final laminates. Alongside this, the second script calculated the maximum loading reached by the three laminates prior to failure, which is when the failure index reaches 1.

2.5 Failure Theories Used:

This experiment utilized the Tsai-Wu and Maximum Strain Failure criteria to evaluate the performance of the three composite laminates. These criteria were chosen to provide a comparative perspective on failure prediction and to optimize stacking sequences for improved mechanical performance.

The Tsai-Wu criterion is a quadratic failure theory that considers interactions between different stress components. The advantage in this is that it provides a more complex prediction of failure. By accounting for both material strengths and the interaction of stresses, the Tsai-Wu criterion gives a more holistic result to the laminate performance; this also makes the failure criterion more conservative compared to other failure theories. In contrast, the Maximum Strain criterion evaluates failure based on the principal strains exceeding predefined material limits. This approach is computationally simpler and easier to interpret; however, it does not account for stress interactions, which can lead to less conservative or less accurate predictions under multi-axial loading. Using both criteria allowed for a comprehensive analysis of the laminates. For further detail on the two failure theories and their equations refer to the bottom of **Appendix B.**

3 Results and Discussions

Using the CLPT code, the local and global stresses and strains for each stacking sequence were calculated. The local stresses and strains for stacking sequence 1 can be seen below in Figure 5. All stacking sequences' local and global strains can be found in **Appendix D**.

	name	sigma1	sigma2	sigma12	epsilon1	epsilon2	gamma12
1	"ply1"	-1.0419e+03	-43.1304	-21.1824	-0.0061	-0.0029	-0.0033
2	"ply2"	-745.7487	-42.1053	24.3001	-0.0043	-0.0033	0.0037
3	"ply3"	-477.8251	-40.0242	19.9905	-0.0028	-0.0036	0.0031
4	"ply4"	-284.5388	-35.1544	11.3958	-0.0016	-0.0034	0.0018
5	"ply5"	-281.0320	-23.1940	-11.2994	-0.0016	-0.0021	-0.0017
6	"ply6"	-136.5873	-16.4994	4.8713	-7.7794e-04	-0.0016	7.4944e-04
7	"ply7"	-116.8582	-5.1451	-1.6091	-6.8203e-04	-3.5733e-04	-2.4756e-04
8	"ply8"	191.5011	7.5166	4.9153	0.0011	4.8391e-04	7.5620e-04
9	"ply9"	159.3158	20.8106	-8.1775	9.0452e-04	0.0020	-0.0013
10	"ply10"	305.9001	27.4252	14.7018	0.0018	0.0025	0.0023
11	"ply11"	307.2673	39.4655	-14.7020	0.0017	0.0038	-0.0023
12	"ply12"	525.2755	43.4116	-23.8637	0.0030	0.0039	-0.0037
13	"ply13"	818.2520	44.5567	-27.7024	0.0048	0.0034	-0.0043
14	"ply14"	1.1166e+03	45.5019	24.4887	0.0065	0.0030	0.0038

Figure 5: Stacking Sequence 1 local Stresses and Strains

In calculating the load for failure for each of the three chosen stacking sequences, the Tsai Wu criterion proved to be an overall more conservative failure criterion, as it generally predicted failure earlier than the maximum strain criterion. These loads for failure can be seen below in Figure 6. For the Tsai Wu criterion, stacking sequence 1 withheld 31% greater load than stacking sequence 2 and 84% greater than stacking sequence 3. For the Max Strain criterion, stacking sequence 2 performed the best, sustaining 47% greater load than stacking sequence 1 and 352% greater than stacking sequence 3.

Tsai Wu Criterion

Stacking Sequence	Load (N) case for Failure [1.001]
[60 -60 -45 -30 30 -30 60 60 -30 30 -30 -45 -60 60]	[18.15 36.3 0 453.75 907.5 0] (ply 14)
[45 -60 -60 45 -45 -45 30 30 -45 -45 45 -60 -60 45]	[13.89 27 78 0 347.25 694.5 0] (ply 14)
[90 45 0 90 45 0 90 90 0 45 90 0 45 90]	[9.86 19.72 0 246.5 493 0] (ply 13)

Max Strain Criterion

Stacking Sequence	Load (N) case for Failure [1.001]
[60 -60 -45 -30 30 -30 60 60 -30 30 -30 -45 -60 60]	[21.71 43.42 0 542.75 1085.5 0] (ply 1)
[45 -60 -60 45 -45 -45 30 30 -45 -45 45 -60 -60 45]	[31.92 63.84 0 798 1596 0] (ply 1)
[90 45 0 90 45 0 90 90 0 45 90 0 45 90]	[7.06 14.12 0 176.5 353 0] (ply 1)

Figure 6: Maximum Loads for Failure Results

From a mechanical standpoint, failure is most likely to occur at the outermost ply in composite laminates, especially under bending or high moments, as the outermost layers experience the largest tensile or compressive stresses due to their distance from the neutral axis. This makes them more susceptible to failure, such as delamination or matrix cracking. The type of loading significantly affects this behavior, as bending loads amplify stresses in the outer plies, while tensile or compressive loads can also lead to failure in these plies. To compare stiffness between the three layups, the ABD matrices can be compared. These respective matrices can be found in **Appendix F**. Looking at these matrices, it is clear that the stiffness layup is the third stacking sequence, as its A and D matrices hold higher values than the other sequences. The chosen final layup is not the stiffest. Although stiffness helps resist deformation, a stiffer laminate may be more prone to failure due to higher stress concentrations, leading to issues like delamination or cracking. Additionally, the stiffest layup may increase weight, which could be detrimental in applications where material efficiency is crucial, like in aerospace. Ultimately, the best layup balances stiffness, strength, durability, and weight for the specific loading conditions.

From these results, it is clear that stacking sequences 1 and 2 are superior to 3, as they both sustain a higher load for failure in each failure criterion. To determine the final laminate selection, it was determined to base the decision on the more conservative method, Tsai-Wu. By choosing this criterion, failure would be predicted earlier, thus preventing a premature failure of the CubeSat's panels while reaching LEO. In all three stacking sequences, the failure mode likely to occur is delamination. This is because the high moments create out-of-plane tensile stresses within the panel. These stresses act on the interlaminar regions of the layup, where the matrix holds the plies together, causing the layers to separate. This failure is critical near the edges and attachment points, as the stress concentrations will be the highest here and can propagate as the loading condition progresses.

4 Conclusions

This study demonstrates the Classical Laminated Plate Theory (CLPT) to optimize the laminate layup in order to improve the structural performance of the CubeSat composite panels under complex loading conditions. Carbon/epoxy was utilized in a 2.1 mm panel thickness, 0.15 mm ply thickness, and 14 ply setup. Although three stacking sequences were analyzed, [60 -60 -45 -30 30 -30 60]s was chosen as the final sequence as it performed the best under the conservative Tsai-Wu failure criterion. The specific loading case for failure was found to be [18.15 36.3 0 453.75 907.5 0]' causing failure in ply 14, as shown in Figure 2. The specific values for the global and local stresses and strains are presented in Appendix D. These findings contribute to the ongoing development of advanced composite structures for aerospace applications, offering a way to improve the accessibility and dependability of small satellite technologies. Future work could explore integrating environmental factors such as radiation and thermal cycling to improve the laminate design further.

5 References

1. National Aeronautics and Space Administration (NASA). Small Spacecraft Reliability Initiative Knowledge Base: Resources for Improving the Reliability of Small Spacecraft Systems. NASA, s3vi.ndc.nasa.gov/ssri-kb/static/resources/FULLTEXT01.pdf.
2. Daniel, Isaac M., and Ori Ishai. Engineering Mechanics of Composite Materials. 2nd ed., Oxford University Press, 2006.
3. National Aeronautics and Space Administration (NASA). "Chapter 5.1: Basics of Space Flight." Science NASA, science.nasa.gov/learn/basics-of-space-flight/chapter5-1/.
4. ISISpace. "ISIS CubeSat Solar Panels." ISISpace, www.isispace.nl/product/isis-cubesat-solar-panels/.

6 Appendix

6.1 **Appendix A:** Material Properties from Engineering Mechanics of Composite Materials by Daniel, Isaac M., and Ori Ishai

TABLE A.4 Properties of Typical Unidirectional Composites (Two-Dimensional)

Property	E-Glass/ Epoxy	S-Glass/ Epoxy	Kevlar/Epoxy (Aramid 49/ Epoxy)	Carbon/Epoxy (AS4/3501-6)	Carbon/Epoxy (IM6G/3501-6)
Fiber volume ratio, V_f	0.55	0.50	0.60	0.63	0.66
Density, ρ, g/cm^3 (lb/in^3)	1.97 (0.071)	2.00 (0.072)	1.38 (0.050)	1.60 (0.058)	1.62 (0.059)
Longitudinal modulus, E_1, GPa (Msi)	41 (6.0)	45 (6.5)	80 (11.6)	147 (21.3)	169 (24.5)
Transverse modulus, E_2, GPa (Msi)	10.4 (1.50)	11.0 (1.60)	5.5 (0.80)	10.3 (1.50)	9.0 (1.30)
In-plane shear modulus, G_{12}, GPa (Msi)	4.3 (0.62)	4.5 (0.66)	2.2 (0.31)	7.0 (1.00)	6.5 (0.94)
Major Poisson's ratio, ν_{12}	0.28	0.29	0.34	0.27	0.31
Minor Poisson's ratio, ν_{21}	0.06	0.06	0.02	0.02	0.02
Longitudinal tensile strength, F_{1t}, MPa (ksi)	1140 (165)	1725 (250)	1400 (205)	2280 (330)	2240 (325)
Transverse tensile strength, F_{2t}, MPa (ksi)	39 (5.7)	49 (7.1)	30 (4.2)	57 (8.3)	46 (6.7)
In-plane shear strength, F_6, MPa (ksi)	89 (12.9)	70 (10.0)	49 (7.1)	76 (11.0)	73 (10.6)
Ultimate longitudinal tensile strain, ε_{1t}^u	0.028	0.029	0.015	0.015	0.013
Ultimate transverse tensile strain, ε_{2t}^u	0.005	0.006	0.005	0.006	0.005
Longitudinal compressive strength, F_{1c}, MPa (ksi)	620 (90)	690 (100)	335 (49)	1725 (250)	1680 (245)
Transverse compressive strength, F_{2c}, MPa (ksi)	128 (18.6)	158 (22.9)	158 (22.9)	228 (33)	215 (31)
Longitudinal thermal expansion coefficient, α_1, 10^{-6}/°C (10^{-6}/°F)	7.0 (3.9)	7.1 (3.9)	−2.0 (−1.1)	−0.9 (−0.5)	−0.9 (−0.5)
Transverse thermal expansion coefficient, α_2, 10^{-6}/°C (10^{-6}/°F)	26 (14.4)	30 (16.7)	60 (33)	27 (15)	25 (13.9)
Longitudinal moisture expansion coefficient, β_1	0	0	0	0.01	0
Transverse moisture expansion coefficient, β_2	0.2	0.2	0.3	0.2	—

6.2 **Appendix B:** MATLAB Script for Identifying Optimal Stacking Sequences for Laminate Analysis

Composite Project - Script 1

```
%Material Properties: Carbon/Epoxy IM6G/3501-6
E1 = 169 * 10^3 %MPa
E2 = 9 * 10^3 %MPa
G12 = 6.5 * 10^3 %MPa
v12 = 0.31
v21 = 0.02

tply = 0.15 %mm
num_plys = 7
stacking_orientation = [0,45,90, 30, 60, -45, -30, -60]
%Testing with a smaller combination #
%stacking_orientation = [0,45,90]

%Generate all possible stacking sequences for the first 7 plies
num_combinations = length(stacking_orientation)^num_plys;
%Creates the empty matrix for all SS combinations
all_sequences = zeros(num_combinations, num_plys * 2);

%
for i = 0:num_combinations-1
    %Determines the stacking orientation and varies it for all thetas
    digits = dec2base(i, length(stacking_orientation), num_plys) - '0' + 1;

    %Map theta to the first 7 plies since the composite is symmetric
    first_half = stacking_orientation(digits);

    %Create the full symmetric sequence
    full_sequence = [first_half, fliplr(first_half)];

    %Store the full sequence for all combinations
    all_sequences(i+1, :) = full_sequence;
end

%All stacking sequences stored in the following:
all_sequences
```

```
%Location of the plys given the middle of the laminate is z = 0.
zk = [-7*tply -6*tply -5*tply -4*tply -3*tply -2*tply -tply 0 tply 2*tply 3*tply 4*tply 5*tply 6*tply 7*tply]

%Now compute S and Q matricies which are the same for all plys/composites.
S = [1/E1 -v12/E1 0;
    -v12/E1 1/E2 0;
    0 0 1/G12]
Q = inv(S)
```

```
% Initialize A, B, D, alpha, beta, del matrices for all sequences
num_sequences = size(all_sequences, 1);
A_all = cell(num_sequences, 1);
B_all = cell(num_sequences, 1);
D_all = cell(num_sequences, 1);
alpha_all = cell(num_sequences, 1);
beta_all = cell(num_sequences, 1);
delta_all = cell(num_sequences, 1);

% Loop through all stacking sequences
for seq_idx = 1:num_sequences
    % Current stacking sequence
    current_sequence = all_sequences(seq_idx, :);

    % Reset A, B, D matrices
    A = 0;
    B = 0;
    D = 0;

    % Loop over plies in the current sequence
    for ply_idx = 1:length(current_sequence)
        % Get ply angle from current sequence
        theta = current_sequence(ply_idx);

        % Strain Transformation Matrix
        T = [cosd(theta)^2, sind(theta)^2, sind(theta)*cosd(theta);
             sind(theta)^2, cosd(theta)^2, -sind(theta)*cosd(theta);
             -2*sind(theta)*cosd(theta), 2*sind(theta)*cosd(theta), cosd(theta)^2 - sind(theta)^2];

        % Calculate Q_bar for the ply
```

```
        % Calculate Q_bar for the ply
        Q_bar = transpose(T) * Q * T;

        % Update A, B, D matrices
        A = A + Q_bar * (zk(ply_idx+1) - zk(ply_idx));
        B = B + Q_bar * (zk(ply_idx+1)^2 - zk(ply_idx)^2);
        D = D + Q_bar * (zk(ply_idx+1)^3 - zk(ply_idx)^3);
    end

    % Normalize B and D matrices
    B = B / 2;
    D = D / 3;

    % Calculate alpha, beta, and delta
    delta = inv(D - (B * inv(A) * B));
    alpha = inv(A) + (inv(A) * B * delta * B * inv(A));
    beta = -(inv(A) * B * delta);

    % Store results for the current sequence
    A_all{seq_idx} = A;
    B_all{seq_idx} = B;
    D_all{seq_idx} = D;
    alpha_all{seq_idx} = alpha;
    beta_all{seq_idx} = beta;
    delta_all{seq_idx} = delta;
end
```

```
% Setting the load cases
Nx = 490.5 % N
Ny = 981  % N
Nxy = 0   % N
Mx = 73.58  % N*mm
My = 147.15    % N*mm
Mxy = 0   % N*mm

load_vector = [Nx; Ny; Nxy; Mx; My; Mxy];

% Initialize storage for midplane strain and curvature for all sequences
midplane_strains = cell(num_sequences, 1);
curvatures = cell(num_sequences, 1);

% Loop through all stacking sequences and calculate curvature and midplane
% strain
for seq_idx = 1:num_sequences

    alpha = alpha_all{seq_idx};
    beta = beta_all{seq_idx};
    delta = delta_all{seq_idx};

    % Midplane Strain and Curvature matrix
    msc_matrix = [alpha, beta; transpose(beta), delta] * load_vector;

    midplane_strains{seq_idx} = msc_matrix(1:3, 1);
    curvatures{seq_idx} = msc_matrix(4:6, 1);
end
```

TSAI-WU Failure Criterion Analysis:

```
% Adjust zk to be at the midplane of all plies
zk_mid = [(-7*tply)/2 (-6*tply)/2 (-5*tply)/2 (-4*tply)/2 (-3*tply)/2 (-2*tply)/2 (-tply)/2 tply/2 tply 3*tply/2 4*t

% Material Strengths
F1t = 2240; % Longitudinal tensile strength (example, in MPa)
F1c = 1680; % Longitudinal compressive strength (example, in MPa)
F2t = 46; % Transverse tensile strength (example, in MPa)
F2c = 215; % Transverse compressive strength (example, in MPa)
F12 = 73; % Shear strength (example, in MPa)

% Tsai-Wu Coefficients
F1 = 1/F1t - 1/F1c;
F2 = 1/F2t - 1/F2c;
F11 = 1/(F1t * F1c);
F22 = 1/(F2t * F2c);
F66 = 1/F12^2;
F12_coeff = -0.5 * sqrt(F11 * F22);

% Initialize storage for failure indices & check for best SS
failure_indices = cell(num_sequences, 1);
min_failure_index = Inf;
```

```matlab
best_sequence_idx = 0;

% Loop through all stacking sequences
for seq_idx = 1:num_sequences

    % Current SS
    current_sequence = all_sequences(seq_idx, :);
    midplane_strain = midplane_strains{seq_idx};
    K = curvatures{seq_idx};
    Qbar_seq = cell(1, length(current_sequence));

    for ply_idx = 1:length(current_sequence)
        theta = current_sequence(ply_idx);
        T = [cosd(theta)^2, sind(theta)^2, sind(theta)*cosd(theta);
             sind(theta)^2, cosd(theta)^2, -sind(theta)*cosd(theta);
             -2*sind(theta)*cosd(theta), 2*sind(theta)*cosd(theta), cosd(theta)^2 - sind(theta)^2];
        Qbar_seq{ply_idx} = transpose(T) * Q * T;
    end

    % Compute failure index for all plies in the sequence
    FI_seq = zeros(1, length(current_sequence));
    for ply_idx = 1:length(current_sequence)
        % Global stresses in the ply
        sigma_global = Qbar_seq{ply_idx} * midplane_strain + Qbar_seq{ply_idx} * (zk_mid(ply_idx) * K);

        % Transform to local stresses
        theta = current_sequence(ply_idx);
        T_stress = [cosd(theta)^2, sind(theta)^2, 2*sind(theta)*cosd(theta);
                    sind(theta)^2, cosd(theta)^2, -2*sind(theta)*cosd(theta);
                    -sind(theta)*cosd(theta), sind(theta)*cosd(theta), cosd(theta)^2 - sind(theta)^2];
        sigma_local = T_stress * sigma_global;

        % Extract local stress components
        sigma1 = sigma_local(1);
        sigma2 = sigma_local(2);
        tau12 = sigma_local(3);

        % Compute Tsai-Wu failure index
        FI = F1 * sigma1 + F2 * sigma2 + F11 * sigma1^2 + F22 * sigma2^2 + F66 * tau12^2 + 2 * F12_coeff * sigma1 *
        FI_seq(ply_idx) = FI;
    end

    % Store the failure indices for the current sequence
    failure_indices{seq_idx} = FI_seq;

    % Determine which failure index from the Tasi-Wu has the max value.
    % This will determine wether or not the stacking sequence is strongest.
    max_failure_index = max(FI_seq);

    % Check if this sequence has the lowest maximum failure index
    if max_failure_index < min_failure_index
        min_failure_index = max_failure_index;
        best_sequence_idx = seq_idx;
    end
end

best_stacking_sequence = all_sequences(best_sequence_idx, :);
yeah
% Display results
disp('Failure indices for all stacking sequences have been computed and stored.');
disp(['The stacking sequence with the best performance is at index ', num2str(best_sequence_idx)]);
disp(['Best stacking sequence: ', mat2str(best_stacking_sequence)]);
disp(['Lowest maximum failure index: ', num2str(min_failure_index)]);
```

Max Strain Failure Criterion Analysis:

```matlab
% Maximum Strain Failure Criterion Analysis
epsilon1_t = F1t / E1; % Longitudinal tensile strain
epsilon1_c = -F1c / E1; % Longitudinal compressive strain
epsilon2_t = F2t / E2; % Transverse tensile strain
epsilon2_c = -F2c / E2; % Transverse compressive strain
gamma12_max = F12 / G12; % Maximum shear strain

% Initialize storage for max strain failure indices for all sequences &
% vars to track best SS
strain_failure_indices = cell(num_sequences, 1);
min_strain_failure_index = Inf; % Set to a very high value initially
best_strain_sequence_idx = 0; % Index of the best stacking sequence

% Loop through all stacking sequences
for seq_idx = 1:num_sequences
```

```
for seq_idx = 1:num_sequences
    % Current stacking sequence
    current_sequence = all_sequences(seq_idx, :);

    % Retrieve midplane strain and curvature for the current sequence
    midplane_strain = midplane_strains{seq_idx};
    K = curvatures{seq_idx};

    % Retrieve Q_bar for the current sequence
    Qbar_seq = cell(1, length(current_sequence));
    for ply_idx = 1:length(current_sequence)
        % Ply angle and Q_bar
        theta = current_sequence(ply_idx);
        T = [cosd(theta)^2, sind(theta)^2, sind(theta)*cosd(theta);
             sind(theta)^2, cosd(theta)^2, -sind(theta)*cosd(theta);
             -2*sind(theta)*cosd(theta), 2*sind(theta)*cosd(theta), cosd(theta)^2 - sind(theta)^2];
        Qbar_seq{ply_idx} = transpose(T) * Q * T;
    end

    % Compute max strain failure index for all plies in the sequence
    strain_FI_seq = zeros(1, length(current_sequence));
    for ply_idx = 1:length(current_sequence)
        % Global strains in the ply
        strain_global = midplane_strain + zk_mid(ply_idx) * K;

        % Transform to local strains
        theta = current_sequence(ply_idx);
        T_strain = [cosd(theta)^2, sind(theta)^2, 2*sind(theta)*cosd(theta);
                    sind(theta)^2, cosd(theta)^2, -2*sind(theta)*cosd(theta);
                    -sind(theta)*cosd(theta), sind(theta)*cosd(theta), cosd(theta)^2 - sind(theta)^2];
        strain_local = T_strain * strain_global;

        % Extract local strain components
        epsilon1 = strain_local(1);
        epsilon2 = strain_local(2);
        gamma12 = strain_local(3);

        % Check strain criteria
        FI_epsilon1 = max([epsilon1 / epsilon1_t, epsilon1 / epsilon1_c]);
        FI_epsilon2 = max([epsilon2 / epsilon2_t, epsilon2 / epsilon2_c]);
        FI_gamma12 = abs(gamma12 / gamma12_max);

        % Overall failure index for the ply (max of all criteria)
        strain_FI_seq(ply_idx) = max([FI_epsilon1, FI_epsilon2, FI_gamma12]);
    end

    % Store the strain failure indices for the current sequence
    strain_failure_indices{seq_idx} = strain_FI_seq;

    % Compute the maximum strain failure index for this sequence
    max_strain_failure_index = max(strain_FI_seq);

    % Check if this sequence has the lowest maximum strain failure index
    if max_strain_failure_index < min_strain_failure_index
        min_strain_failure_index = max_strain_failure_index;
        best_strain_sequence_idx = seq_idx;
    end
end

best_strain_stacking_sequence = all_sequences(best_strain_sequence_idx, :);

% Display results
disp('Maximum strain failure indices for all stacking sequences have been computed and stored.');
disp(['The stacking sequence with the best performance (strain criterion) is at index ', num2str(best_strain_s
disp(['Best stacking sequence (strain criterion): ', mat2str(best_strain_stacking_sequence)]);
disp(['Lowest maximum strain failure index: ', num2str(min_strain_failure_index)]);
```

6.3 **Appendix C:** MATLAB Script for Maximum Load Determination, Local Stress-Strain Analysis, and Failure Evaluation Across All Three Stacking Sequences

```
%Material Properties: Carbon/Epoxy IM6G/3501-6
E1 = 169 * 10^3 %MPa
E2 = 9 * 10^3 %MPa
G12 = 6.5 * 10^3 %MPa
v12 = 0.31
v21 = 0.02

tply = 0.15 %mm
num_plys = 7
stacking_orientation = [0,45,90, 30, 60, -45, -30, -60]

%Testing with a smaller combination #
%stacking_orientation = [0,45,90]

%Generate all possible stacking sequences for the first 7 plies
num_combinations = length(stacking_orientation)^num_plys;
%Creates the empty matrix for all SS combinations
all_sequences = zeros(num_combinations, num_plys * 2);

%
for i = 0:num_combinations-1
    %Determines the stacking orientation and varies it for all thetas
    digits = dec2base(i, length(stacking_orientation), num_plys) - '0' + 1;

    %Map theta to the first 7 plies since the composite is symmetric
    first_half = stacking_orientation(digits);

    %Create the full symmetric sequence
    full_sequence = [first_half, fliplr(first_half)];

    %Store the full sequence for all combinations
    all_sequences(i+1, :) = full_sequence;
end


all_sequences=[60 -60 -45 -30 30 -30 60 60 -30 30 -30 -45 -60 60]
```

```
%Location of the plys given the middle of the laminate is z = 0.
zk = [-7*tply -6*tply -5*tply -4*tply -3*tply -2*tply -tply 0 tply 2*tply 3*tply 4*tply 5*tply 6*tply 7*tply]

%Now compute S and Q matricies which are the same for all plys/composites.
S = [1/E1 -v12/E1 0;
    -v12/E1 1/E2 0;
    0 0 1/G12]
Q = inv(S)
```

```
% Initialize A, B, D, alpha, beta, del matrices for all sequences
num_sequences = size(all_sequences, 1);
A_all = cell(num_sequences, 1);
B_all = cell(num_sequences, 1);
D_all = cell(num_sequences, 1);
alpha_all = cell(num_sequences, 1);
beta_all = cell(num_sequences, 1);
delta_all = cell(num_sequences, 1);

% Loop through all stacking sequences
for seq_idx = 1:num_sequences
    % Current stacking sequence
    current_sequence = all_sequences(seq_idx, :);

    % Reset A, B, D matrices
    A = 0;
    B = 0;
    D = 0;

    % Loop over plies in the current sequence
    for ply_idx = 1:length(current_sequence)
        % Get ply angle from current sequence
        theta = current_sequence(ply_idx);

        % Strain Transformation Matrix
        T = [cosd(theta)^2, sind(theta)^2, sind(theta)*cosd(theta);
             sind(theta)^2, cosd(theta)^2, -sind(theta)*cosd(theta);
             -2*sind(theta)*cosd(theta), 2*sind(theta)*cosd(theta), cosd(theta)^2 - sind(theta)^2];

        % Calculate Q_bar for the ply
        Q_bar = transpose(T) * Q * T;
```

```
        % Update A, B, D matrices
        A = A + Q_bar * (zk(ply_idx+1) - zk(ply_idx));
        B = B + Q_bar * (zk(ply_idx+1)^2 - zk(ply_idx)^2);
        D = D + Q_bar * (zk(ply_idx+1)^3 - zk(ply_idx)^3);
    end

    % Normalize B and D matrices
    B = B / 2;
    D = D / 3;

    % Calculate alpha, beta, and delta
    delta = inv(D - (B * inv(A) * B));
    alpha = inv(A) + (inv(A) * B * delta * B * inv(A));
    beta = -(inv(A) * B * delta);

    % Store results for the current sequence
    A_all{seq_idx} = A;
    B_all{seq_idx} = B;
    D_all{seq_idx} = D;
    alpha_all{seq_idx} = alpha;
    beta_all{seq_idx} = beta;
    delta_all{seq_idx} = delta;
end

% Adjust Loads until Failure is achieved
Nx = 18.15 % N/mm
Ny = Nx*2  % N/mm
Nxy = 0   % N/mm
Mx = Nx*5^2  % N*mm
My = Ny*5^2   % N*mm
Mxy = 0   % N*mm

load_vector = [Nx; Ny; Nxy; Mx; My; Mxy];

% Initialize storage for midplane strain and curvature for all sequences
midplane_strains = cell(num_sequences, 1);
curvatures = cell(num_sequences, 1);

% Loop through all stacking sequences and calculate curvature and midplane
% strain
for seq_idx = 1:num_sequences

    alpha = alpha_all{seq_idx};
```

```
    alpha = alpha_all{seq_idx};
    beta = beta_all{seq_idx};
    delta = delta_all{seq_idx};

    % Midplane Strain and Curvature matrix
    msc_matrix = [alpha, beta; transpose(beta), delta] * load_vector;

    midplane_strains{seq_idx} = msc_matrix(1:3, 1);
    curvatures{seq_idx} = msc_matrix(4:6, 1);
end
```

TSAI-WU Failure Criterion Analysis:

```
% Adjust zk to be at the midplane of all plies
zk_mid = [(-7*tply)/2 (-6*tply)/2 (-5*tply)/2 (-4*tply)/2 (-3*tply)/2 (-2*tply)/2
    (-tply)/2 tply/2 tply 3*tply/2 4*tply/2 5*tply/2 6*tply/2 7*tply/2];

% Material Strengths
F1t = 2240; % Longitudinal tensile strength (example, in MPa)
F1c = 1680; % Longitudinal compressive strength (example, in MPa)
F2t = 46; % Transverse tensile strength (example, in MPa)
F2c = 215; % Transverse compressive strength (example, in MPa)
F12 = 73; % Shear strength (example, in MPa)

% Tsai-Wu Coefficients
F1 = 1/F1t - 1/F1c;
F2 = 1/F2t - 1/F2c;
F11 = 1/(F1t * F1c);
F22 = 1/(F2t * F2c);
F66 = 1/F12^2;
F12_coeff = -0.5 * sqrt(F11 * F22);

% Initialize storage for failure indices & check for best SS
failure_indices = cell(num_sequences, 1);
min_failure_index = Inf;
best_sequence_idx = 0;

% Loop through all stacking sequences
for seq_idx = 1:num_sequences

```

```
    % Current SS
    current_sequence = all_sequences(seq_idx, :);
    midplane_strain = midplane_strains{seq_idx};
    K = curvatures{seq_idx};
    Qbar_seq = cell(1, length(current_sequence));

    for ply_idx = 1:length(current_sequence)
        theta = current_sequence(ply_idx);
        T = [cosd(theta)^2, sind(theta)^2, sind(theta)*cosd(theta);
             sind(theta)^2, cosd(theta)^2, -sind(theta)*cosd(theta);
             -2*sind(theta)*cosd(theta), 2*sind(theta)*cosd(theta), cosd(theta)^2 - sind(theta)^2];
        Qbar_seq{ply_idx} = transpose(T) * Q * T;
    end
    sigma_global=cell(1,14);
    strain_global=cell(1,14);
    sigma_local=cell(1,14)
    % Compute failure index for all plies in the sequence
    FI_seq = zeros(1, length(current_sequence));
    for ply_idx = 1:length(current_sequence)
        % Global stresses in the ply
        sigma_global{ply_idx} = Qbar_seq{ply_idx} * midplane_strain + Qbar_seq{ply_idx} * (zk_mid(ply_idx) * K);

        % Transform to local stresses
        theta = current_sequence(ply_idx);
        T_stress = [cosd(theta)^2, sind(theta)^2, 2*sind(theta)*cosd(theta);
                    sind(theta)^2, cosd(theta)^2, -2*sind(theta)*cosd(theta);
                    -sind(theta)*cosd(theta), sind(theta)*cosd(theta), cosd(theta)^2 - sind(theta)^2];
        sigma_local = T_stress * sigma_global{ply_idx};

        strain_global{ply_idx} = midplane_strain + zk_mid(ply_idx) * K;



        % Extract local stress components
        sigma1 = sigma_local(1);
        sigma2 = sigma_local(2);
        tau12 = sigma_local(3);

        % Compute Tsai-Wu failure index
        FI = F1 * sigma1 + F2 * sigma2 + F11 * sigma1^2 + F22 * sigma2^2 + F66 * tau12^2 + 2 * F12_coeff * sigma1 * sigma2;
        FI_seq(ply_idx) = FI;
```

```
    end
    FI_seq;

    % Store the failure indices for the current sequence
    failure_indices{seq_idx} = FI_seq;

    % Determine which failure index from the Tasi-Wu has the max value.
    % This will determine wether or not the stacking sequence is strongest.
    max_failure_index = max(FI_seq);
    for i = 1:length(FI_seq)
        if FI_seq(i) == max(FI_seq)
            index=i;
        end
    end

    % Check if this sequence has the lowest maximum failure index
    if max_failure_index > 1
        disp('failure has occured')
    end
    if max_failure_index < min_failure_index
        min_failure_index = max_failure_index;
        best_sequence_idx = seq_idx;
    end
end

best_stacking_sequence = all_sequences(best_sequence_idx, :);

% Display results
disp(['Maximum Tsai-Wu failure index: ', num2str(min_failure_index), '; failure in ply ', num2str(index)]);
```

Max Strain Failure Criterion Analysis:

```
% Maximum Strain Failure Criterion Analysis
epsilon1_t = F1t / E1; % Longitudinal tensile strain
epsilon1_c = -F1c / E1; % Longitudinal compressive strain
epsilon2_t = F2t / E2; % Transverse tensile strain
epsilon2_c = -F2c / E2; % Transverse compressive strain
gamma12_max = F12 / G12; % Maximum shear strain

% Initialize storage for max strain failure indices for all sequences &
% vars to track best SS
```

```
strain_failure_indices = cell(num_sequences, 1);
min_strain_failure_index = Inf; % Set to a very high value initially
best_strain_sequence_idx = 0; % Index of the best stacking sequence

% Loop through all stacking sequences
for seq_idx = 1:num_sequences
    % Current stacking sequence
    current_sequence = all_sequences(seq_idx, :);

    % Retrieve midplane strain and curvature for the current sequence
    midplane_strain = midplane_strains{seq_idx};
    K = curvatures{seq_idx};

    % Retrieve Q_bar for the current sequence
    Qbar_seq = cell(1, length(current_sequence));
    for ply_idx = 1:length(current_sequence)
        % Ply angle and Q_bar
        theta = current_sequence(ply_idx);
        T = [cosd(theta)^2, sind(theta)^2, sind(theta)*cosd(theta);
             sind(theta)^2, cosd(theta)^2, -sind(theta)*cosd(theta);
             -2*sind(theta)*cosd(theta), 2*sind(theta)*cosd(theta), cosd(theta)^2 - sind(theta)^2];
        Qbar_seq{ply_idx} = transpose(T) * Q * T;
    end

    % Compute max strain failure index for all plies in the sequence
    strain_FI_seq = zeros(1, length(current_sequence));
    for ply_idx = 1:length(current_sequence)
        % Global strains in the ply
        strain_global = midplane_strain + zk_mid(ply_idx) * K;

        % Transform to local strains
        theta = current_sequence(ply_idx);
        T_strain = [cosd(theta)^2, sind(theta)^2, 2*sind(theta)*cosd(theta);
                    sind(theta)^2, cosd(theta)^2, -2*sind(theta)*cosd(theta);
                    -sind(theta)*cosd(theta), sind(theta)*cosd(theta), cosd(theta)^2 - sind(theta)^2];
        strain_local = T_strain * strain_global;

        % Extract local strain components
        epsilon1 = strain_local(1);
        epsilon2 = strain_local(2);
        gamma12 = strain_local(3);

```

```
        % Check strain criteria
        FI_epsilon1 = max([epsilon1 / epsilon1_t, epsilon1 / epsilon1_c]);
        FI_epsilon2 = max([epsilon2 / epsilon2_t, epsilon2 / epsilon2_c]);
        FI_gamma12 = abs(gamma12 / gamma12_max);

        % Overall failure index for the ply (max of all criteria)
        strain_FI_seq(ply_idx) = max([FI_epsilon1, FI_epsilon2, FI_gamma12]);
    end

    % Store the strain failure indices for the current sequence
    sFI{seq_idx} = strain_FI_seq;

    % Compute the maximum strain failure index for this sequence
    max_strain_failure_index = max(strain_FI_seq);

    % Check if this sequence has the lowest maximum strain failure index
    if max_strain_failure_index < min_strain_failure_index
        min_strain_failure_index = max_strain_failure_index;
        best_strain_sequence_idx = seq_idx;
    end
end

best_strain_stacking_sequence = all_sequences(best_strain_sequence_idx, :);
% Display results
if min_strain_failure_index>1
    disp('failure has occured')
end
disp(['Maximum strain failure index: ', num2str(min_strain_failure_index), ' in ply ', num2str(best_strain_sequence_idx)]);
```

```
function [t] = SSF(E1, E2, G12, v12, tply, t1, NM, zm)
% t1 = stacking sequence, NM = forces and moment vector, zm = z distances
% Create compliance matrix, s using the material properties, in order to get stiffness, Q
s=[1/E1 -v12/E1 0; -v12/E1 1/E2 0; 0 0 1/G12];
% Convert Q by calculating th inverse of the S matrix
Q=inv(s);
% Create Array for to store all of the Te matricies
Te=cell(1,length(t1));
Ts=cell(1,length(t1));
% Create array to store all of the Qbar matrices for each ply
Qbar=cell(1,length(t1));
% Create an array that stores the z values for any even number of plies
```

```
% Create an array that stores the z values for any even number of plies
z=cell(1,length(t1));
% Set A, B, D matrcies to 0 so that the for loop can add each run up
A=0;
B=0;
D=0;
% Create array for z values
for i=-length(t1)/2:length(t1)/2 % run the for loop from the bottom ply (negative z value) to the top ply (positive z value)
    zo=tply*i;
    z{i+1+length(t1)/2}=zo; % Create an array filled with the z values
end
% Create for loop that creates every Te matrix for each ply angle, then gets the Qbar matrix from
% Te'*Q*Te. Then it creates the A, B, D matrices by their respective formulas
for i=1:length(z)-1
    % Strain transformation matrix
    T=[cosd(t1(i))^2 sind(t1(i))^2 cosd(t1(i))*sind(t1(i));
        sind(t1(i))^2 cosd(t1(i))^2 -cosd(t1(i))*sind(t1(i));
-2*cosd(t1(i))*sind(t1(i)) 2*cosd(t1(i))*sind(t1(i)) cosd(t1(i))^2-sind(t1(i))^2];
    Te{i}=T; % store transformation matrices in array Te
    Ts{i}=[cosd(t1(i))^2 sind(t1(i))^2 2*cosd(t1(i))*sind(t1(i));
        sind(t1(i))^2 cosd(t1(i))^2 -2*cosd(t1(i))*sind(t1(i));
-cosd(t1(i))*sind(t1(i)) cosd(t1(i))*sind(t1(i)) cosd(t1(i))^2-sind(t1(i))^2];;


    Qbar{i}=Te{i}'*Q*Te{i};
    x=Qbar{i}*(z{i+1}-z{i}); % Creates array for A Matrix
    A=A+x; % Sums values in A Matrix
    f=Qbar{i}*(z{i+1}^2-z{i}^2); % Creates array for B Matrix
    B=B+f; % Sums values in B Matrix
    g=Qbar{i}*(z{i+1}^3-z{i}^3); % Creates array for D Matrix
    D=D+g; % Sums values in D Matrix
end
% Still need to multiply by the factors outside of the sum
A=A;
B=B/2;
D=D/3;
% Get mid plane strains
C=zeros(6);
C([1:3],[1,2,3])=A;
C([4:6],[4,5,6])=D;
C([1,2,3],[4,5,6])=B;
C([4,5,6],[1,2,3])=B';
ek=inv(C)*NM;
```

```
ek=inv(C)*NM;
% get local stressses and strains
e=ek(1:3);
K=ek(4:6);
% Create transformation matrix to go from global strains to local.
% Create array to store transformation matrices for each ply angle
Te=cell(1,length(t1));
Qbar=cell(1,length(t1));
s=[1/E1 -v12/E1 0; -v12/E1 1/E2 0; 0 0 1/G12];
% Convert Q by calculating th inverse of the S matrix
Q=inv(s);
for i=1:length(t1)
    % Create transformation matrix
    T=[cosd(t1(i))^2 sind(t1(i))^2 cosd(t1(i))*sind(t1(i));
        sind(t1(i))^2 cosd(t1(i))^2 -cosd(t1(i))*sind(t1(i));
-2*cosd(t1(i))*sind(t1(i)) 2*cosd(t1(i))*sind(t1(i)) cosd(t1(i))^2-sind(t1(i))^2];
Te{i}=T;
% Calculates each ply's transformation matrix times mid plane strains and
% stores in seperate cell in array
Qbar{i}=Te{i}'*Q*Te{i};
end
% Create arrays to store all locl and global stress and strain values
sl=cell(1,length(t1));
sg=cell(1,length(t1));
eg=cell(1,length(t1));
el=cell(1,length(t1)); % Create matrix containing the mid plane z values
% Calculate all local and global midplane stress and strain values
for i=1:length(zm)
    sg{i}=Qbar{i}*e+Qbar{i}*zm(i)*K;
    sl{i}=Ts{i}*sg{i};
    eg{i}=inv(Qbar{i})*sg{i};
    el{i}=Te{i}*eg{i};
end
% Tabulate results
% Initialize arrays for 14 plies
name = ["ply1"; "ply2"; "ply3"; "ply4"; "ply5"; "ply6"; "ply7"; "ply8"; "ply9"; "ply10"; "ply11"; "ply12"; "ply13"; "ply14"];

% Assuming el and sl are cell arrays containing data for all 14 plies
epsilon1 = [eg{1}(1); eg{2}(1); eg{3}(1); eg{4}(1); eg{5}(1); eg{6}(1); eg{7}(1); eg{8}(1); eg{9}(1); eg{10}(1); eg{11}(1); eg{12}(1); eg{13}(1); eg{14}(1)];
epsilon2 = [eg{1}(2); eg{2}(2); eg{3}(2); eg{4}(2); eg{5}(2); eg{6}(2); eg{7}(2); eg{8}(2); eg{9}(2); eg{10}(2); eg{11}(2); eg{12}(2); eg{13}(2); eg{14}(2)];
gamma12  = [eg{1}(3); eg{2}(3); eg{3}(3); eg{4}(3); eg{5}(3); eg{6}(3); eg{7}(3); eg{8}(3); eg{9}(3); eg{10}(3); eg{11}(3); eg{12}(3); eg{13}(3); eg{14}(3)];
```

```
Qbar{i}=Te{i}'*Q*Te{i};
end
% Create arrays to store all locl and global stress and strain values
sl=cell(1,length(t1));
sg=cell(1,length(t1));
eg=cell(1,length(t1));
el=cell(1,length(t1)); % Create matrix containing the mid plane z values
% Calculate all local and global midplane stress and strain values
for i=1:length(zm)
    sg{i}=Qbar{i}*e+Qbar{i}*zm(i)*K;
    sl{i}=Ts{i}*sg{i};
    eg{i}=inv(Qbar{i})*sg{i};
    el{i}=Te{i}*eg{i};
end
% Tabulate results
% Initialize arrays for 14 plies
name = ["ply1"; "ply2"; "ply3"; "ply4"; "ply5"; "ply6"; "ply7"; "ply8"; "ply9"; "ply10"; "ply11"; "ply12"; "ply13"; "ply14"];

% Assuming el and sl are cell arrays containing data for all 14 plies
epsilon1 = [eg{1}(1); eg{2}(1); eg{3}(1); eg{4}(1); eg{5}(1); eg{6}(1); eg{7}(1); eg{8}(1); eg{9}(1); eg{10}(1); eg{11}(1); eg{12}(1); eg{13}(1); eg{14}(1)];
epsilon2 = [eg{1}(2); eg{2}(2); eg{3}(2); eg{4}(2); eg{5}(2); eg{6}(2); eg{7}(2); eg{8}(2); eg{9}(2); eg{10}(2); eg{11}(2); eg{12}(2); eg{13}(2); eg{14}(2)];
gamma12  = [eg{1}(3); eg{2}(3); eg{3}(3); eg{4}(3); eg{5}(3); eg{6}(3); eg{7}(3); eg{8}(3); eg{9}(3); eg{10}(3); eg{11}(3); eg{12}(3); eg{13}(3); eg{14}(3)];

sigma1 = [sg{1}(1); sg{2}(1); sg{3}(1); sg{4}(1); sg{5}(1); sg{6}(1); sg{7}(1); sg{8}(1); sg{9}(1); sg{10}(1); sg{11}(1); sg{12}(1); sg{13}(1); sg{14}(1)];
sigma2 = [sg{1}(2); sg{2}(2); sg{3}(2); sg{4}(2); sg{5}(2); sg{6}(2); sg{7}(2); sg{8}(2); sg{9}(2); sg{10}(2); sg{11}(2); sg{12}(2); sg{13}(2); sg{14}(2)];
sigma12 = [sg{1}(3); sg{2}(3); sg{3}(3); sg{4}(3); sg{5}(3); sg{6}(3); sg{7}(3); sg{8}(3); sg{9}(3); sg{10}(3); sg{11}(3); sg{12}(3); sg{13}(3); sg{14}(3)];


% Create table
t = table(name, sigma1, sigma2, sigma12, epsilon1, epsilon2, gamma12);

end

tply=0.15; zm = [(-7*tply)/2 (-6*tply)/2 (-5*tply)/2 (-4*tply)/2 (-3*tply)/2 (-2*tply)/2 (-tply)/2 tply/2 tply 3*tply/2 4*tply/2 5*tply/2 6*tply/2 7*tply/2];
E1 = 169 * 10^3 %MPa
E2 = 9 * 10^3 %MPa
G12 = 6.5 * 10^3 %MPa
v12 = 0.31
v21 = 0.02
t1=[60 -60 -45 -30 30 -30 60 60 -30 30 -30 -45 -60 60]
NM=[18.15 36.3 0 453.75 907.5 0]'
[t]=SSF(E1, E2, G12, v12, tply, t1, NM, zm)
```

6.4 **Appendix D:** Local and Global stress and strain results

	name	sigma1	sigma2	sigma12	epsilon1	epsilon2	gamma12
1	"ply1"	-1.0419e+03	-43.1304	-21.1824	-0.0061	-0.0029	-0.0033
2	"ply2"	-745.7487	-42.1053	24.3001	-0.0043	-0.0033	0.0037
3	"ply3"	-477.8251	-40.0242	19.9905	-0.0028	-0.0036	0.0031
4	"ply4"	-284.5388	-35.1544	11.3958	-0.0016	-0.0034	0.0018
5	"ply5"	-281.0320	-23.1940	-11.2994	-0.0016	-0.0021	-0.0017
6	"ply6"	-136.5873	-16.4994	4.8713	-7.7794e-04	-0.0016	7.4944e-04
7	"ply7"	-116.8582	-5.1451	-1.6091	-6.8203e-04	-3.5733e-04	-2.4756e-04
8	"ply8"	191.5011	7.5166	4.9153	0.0011	4.8391e-04	7.5620e-04
9	"ply9"	159.3158	20.8106	-8.1775	9.0452e-04	0.0020	-0.0013
10	"ply10"	305.9001	27.4252	14.7018	0.0018	0.0025	0.0023
11	"ply11"	307.2673	39.4655	-14.7020	0.0017	0.0038	-0.0023
12	"ply12"	525.2755	43.4116	-23.8637	0.0030	0.0039	-0.0037
13	"ply13"	818.2520	44.5567	-27.7024	0.0048	0.0034	-0.0043
14	"ply14"	1.1166e+03	45.5019	24.4887	0.0065	0.0030	0.0038

Stacking Sequence 1 local Stresses and Strains

	name	sigma1	sigma2	sigma12	epsilon1	epsilon2	gamma12
1	"ply1"	-805.5241	-44.2331	-52.3207	-0.0047	-0.0034	-0.0080
2	"ply2"	-807.2130	-33.2027	41.9375	-0.0047	-0.0022	0.0065
3	"ply3"	-667.2535	-27.4645	34.5077	-0.0039	-0.0018	0.0053
4	"ply4"	-446.6785	-24.7384	-28.7587	-0.0026	-0.0019	-0.0044
5	"ply5"	-247.6927	-21.2056	20.9047	-0.0014	-0.0019	0.0032
6	"ply6"	-160.3623	-13.5011	13.0508	-9.2412e-04	-0.0012	0.0020
7	"ply7"	-53.4493	-6.5283	-4.7885	-3.0429e-04	-6.2732e-04	-7.3669e-04
8	"ply8"	80.5230	10.4009	10.0711	4.5739e-04	0.0010	0.0015
9	"ply9"	188.9595	17.3169	-18.3652	0.0011	0.0016	-0.0028
10	"ply10"	276.2899	25.0213	-26.2192	0.0016	0.0023	-0.0040
11	"ply11"	510.2429	27.2477	34.0732	0.0030	0.0021	0.0052
12	"ply12"	732.3413	29.9169	-39.7903	0.0043	0.0020	-0.0061
13	"ply13"	872.3008	35.6550	-47.2201	0.0051	0.0024	-0.0073
14	"ply14"	869.0884	46.7424	57.6351	0.0051	0.0036	0.0089

Stacking Sequence 2 local Stresses and Strains

	name	sigma1	sigma2	sigma12	epsilon1	epsilon2	gamma12
1	"ply1"	-1.0694e+03	-70.1840	-55.1527	-0.0062	-0.0058	-0.0085
2	"ply2"	-281.7586	-83.4512	-1.9506	-0.0015	-0.0088	-3.0010e-04
3	"ply3"	-714.3226	-51.1283	38.9854	-0.0041	-0.0044	0.0060
4	"ply4"	-596.3357	-39.3754	-30.9017	-0.0035	-0.0033	-0.0048
5	"ply5"	-135.9855	-40.4139	-0.7331	-7.3052e-04	-0.0042	-1.1278e-04
6	"ply6"	-272.5442	-19.1505	14.7344	-0.0016	-0.0016	0.0023
7	"ply7"	-123.2630	-8.5667	-6.6508	-7.1365e-04	-7.2575e-04	-0.0010
8	"ply8"	192.1188	11.9724	9.5165	0.0011	9.7786e-04	0.0015
9	"ply9"	316.4937	23.4867	-17.6001	0.0018	0.0020	-0.0027
10	"ply10"	155.5608	45.6608	1.7021	8.3672e-04	0.0048	2.6186e-04
11	"ply11"	665.1915	42.7810	33.7674	0.0039	0.0035	0.0052
12	"ply12"	758.2721	55.4646	-41.8511	0.0044	0.0048	-0.0064
13	"ply13"	301.3339	88.6981	2.9197	0.0016	0.0093	4.4918e-04
14	"ply14"	1.1383e+03	73.5897	58.0184	0.0066	0.0061	0.0089

Stacking Sequence 2 local Stresses and Strains

	name	sigma1	sigma2	sigma12	epsilon1	epsilon2	gamma12
1	"ply1"	-274.4873	-810.5792	-421.9043	-0.0023	-0.0067	-0.0011
2	"ply2"	-196.9717	-590.8823	292.5365	-0.0019	-0.0057	-9.8151e-04
3	"ply3"	-238.9341	-278.9151	218.9005	-0.0016	-0.0047	-8.1670e-04
4	"ply4"	-212.3236	-107.3695	113.6845	-0.0013	-0.0037	-6.5188e-04
5	"ply5"	-206.7869	-97.4391	-117.2969	-9.7794e-04	-0.0027	-4.8706e-04
6	"ply6"	-102.3466	-50.7401	54.4352	-6.5462e-04	-0.0017	-3.2224e-04
7	"ply7"	-31.6799	-90.3235	-47.5686	-3.3131e-04	-7.0805e-04	-1.5742e-04
8	"ply8"	49.2559	149.7618	77.2099	3.1532e-04	0.0013	1.7221e-04
9	"ply9"	117.6075	62.5188	-64.0633	6.3864e-04	0.0023	3.3703e-04
10	"ply10"	223.5492	109.7760	127.9340	9.6195e-04	0.0033	5.0185e-04
11	"ply11"	227.5846	119.1483	-123.3126	0.0013	0.0043	6.6667e-04
12	"ply12"	260.4799	308.2072	-240.9319	0.0016	0.0053	8.3148e-04
13	"ply13"	213.9896	648.8192	-321.1687	0.0019	0.0063	9.9630e-04
14	"ply14"	292.0634	870.0175	451.5456	0.0023	0.0073	0.0012

Stacking Sequence 1 global Stresses and Strains

	name	sigma1	sigma2	sigma12	epsilon1	epsilon2	gamma12
1	"ply1"	-372.5579	-477.1993	-380.6455	-3.6573e-05	-0.0081	-0.0012
2	"ply2"	-190.3863	-650.0293	314.1876	-4.1486e-05	-0.0069	-0.0011
3	"ply3"	-157.5272	-537.1908	259.7829	-4.6400e-05	-0.0057	-8.6158e-04
4	"ply4"	-206.9497	-264.4672	-210.9701	-5.1314e-05	-0.0045	-6.6834e-04
5	"ply5"	-113.5444	-155.3539	113.2436	-5.6227e-05	-0.0033	-4.7509e-04
6	"ply6"	-73.8809	-99.9824	73.4306	-6.1141e-05	-0.0021	-2.8184e-04
7	"ply7"	-37.5721	-22.4055	-22.7116	-6.6055e-05	-8.6556e-04	-8.8595e-05
8	"ply8"	54.2706	36.6533	35.3993	-7.5882e-05	0.0015	2.9790e-04
9	"ply9"	84.7730	121.5033	-85.8213	-8.0796e-05	0.0027	4.9114e-04
10	"ply10"	124.4365	176.8748	-125.6343	-8.5709e-05	0.0039	6.8439e-04
11	"ply11"	234.6721	302.8184	241.4976	-9.0623e-05	0.0052	8.7764e-04
12	"ply12"	171.0636	591.1947	-284.2635	-9.5536e-05	0.0064	0.0011
13	"ply13"	203.9226	704.0332	-338.6682	-1.0045e-04	0.0076	0.0013
14	"ply14"	400.2803	515.5505	411.1730	-1.0536e-04	0.0088	0.0015

Stacking Sequence 2 global Stresses and Strains

	name	sigma1	sigma2	sigma12	epsilon1	epsilon2	gamma12
1	"ply1"	-70.1840	-1.0694e+03	55.1527	-0.0058	-0.0062	0.0085
2	"ply2"	-180.6543	-184.5555	-99.1537	-0.0050	-0.0053	0.0072
3	"ply3"	-714.3226	-51.1283	38.9854	-0.0041	-0.0044	0.0060
4	"ply4"	-39.3754	-596.3357	30.9017	-0.0033	-0.0035	0.0048
5	"ply5"	-87.4666	-88.9327	-47.7858	-0.0024	-0.0025	0.0035
6	"ply6"	-272.5442	-19.1505	14.7344	-0.0016	-0.0016	0.0023
7	"ply7"	-8.5667	-123.2630	6.6508	-7.2575e-04	-7.1365e-04	0.0010
8	"ply8"	11.9724	192.1188	-9.5165	9.7786e-04	0.0011	-0.0015
9	"ply9"	316.4937	23.4867	-17.6001	0.0018	0.0020	-0.0027
10	"ply10"	98.9087	102.3128	54.9500	0.0027	0.0029	-0.0040
11	"ply11"	42.7810	665.1915	-33.7674	0.0035	0.0039	-0.0052
12	"ply12"	758.2721	55.4646	-41.8511	0.0044	0.0048	-0.0064
13	"ply13"	192.0963	197.9356	106.3179	0.0052	0.0057	-0.0077
14	"ply14"	73.5897	1.1383e+03	-58.0184	0.0061	0.0066	-0.0089

Stacking Sequence 3 global Stresses and Strains

6.5 **Appendix E:** CLPT Equations

Laminate Stiffness Matrices:

$$A_{ij} = \sum_{k=1}^{n} (Q_{ij})_k (z_k - z_{k-1})$$

$$B_{ij} = \frac{1}{2} \sum_{k=1}^{n} (Q_{ij})_k (z_k^2 - z_{k-1}^2)$$

$$D_{ij} = \frac{1}{3} \sum_{k=1}^{n} (Q_{ij})_k (z_k^3 - z_{k-1}^3)$$

Force-strain and moment-curvature relationship:

$$\begin{bmatrix} N_x \\ N_y \\ N_{xy} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{16} \\ A_{12} & A_{22} & A_{26} \\ A_{16} & A_{26} & A_{66} \end{bmatrix} \begin{bmatrix} \varepsilon_{x0} \\ \varepsilon_{y0} \\ \gamma_{xy0} \end{bmatrix} + \begin{bmatrix} B_{11} & B_{12} & B_{16} \\ B_{12} & B_{22} & B_{26} \\ B_{16} & B_{26} & B_{66} \end{bmatrix} \begin{bmatrix} \kappa_x \\ \kappa_y \\ \kappa_{xy} \end{bmatrix}$$

$$\begin{bmatrix} M_x \\ M_y \\ M_{xy} \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} & B_{16} \\ B_{12} & B_{22} & B_{26} \\ B_{16} & B_{26} & B_{66} \end{bmatrix} \begin{bmatrix} \varepsilon_{x0} \\ \varepsilon_{y0} \\ \gamma_{xy0} \end{bmatrix} + \begin{bmatrix} D_{11} & D_{12} & D_{16} \\ D_{12} & D_{22} & D_{26} \\ D_{16} & D_{26} & D_{66} \end{bmatrix} \begin{bmatrix} \kappa_x \\ \kappa_y \\ \kappa_{xy} \end{bmatrix}$$

Material Stiffness:

$$Q = \begin{bmatrix} \frac{E_1}{1-\nu_{12}\nu_{21}} & \frac{\nu_{12}E_2}{1-\nu_{12}\nu_{21}} & 0 \\ \frac{\nu_{12}E_2}{1-\nu_{12}\nu_{21}} & \frac{E_2}{1-\nu_{12}\nu_{21}} & 0 \\ 0 & 0 & G_{12} \end{bmatrix}$$

Reduced Stiffness Matrix:

$$\bar{Q} = T_{\varepsilon}^{T} * Q * T_{\varepsilon}$$

Global and Local Stresses and Strains:

global stresses = $\bar{Q} * \varepsilon + \bar{Q} * z_K * K$

local stresses = T_s * global stresses

global strains = $\bar{Q}^{-1}$ * global stresses

local strains = T_{ε} * global strains

ε = midplane strains

K = curvatures

T_s = stress transformation matrix

T_{ε} = strain transformation matrix

6.6 **Appendix F:** Stiffness (ABD) Matrices

Stacking Sequence 1:

```
A = 3x3
10^5  x
     1.2710    0.6665   -0.2163
     0.6665    1.2710   -0.0249
    -0.2163   -0.0249    0.7441

B = 3x3
10^-11  x
    -0.1364   -0.3638   -0.0909
    -0.0909   -0.1819         0
    -0.0455         0   -0.1819

D = 3x3
10^4  x
     3.2347    2.4744   -0.6803
     2.4744    6.0572   -0.2354
    -0.6803   -0.2354    2.7596
```

Stacking Sequence 2:

```
A = 3x3
10^5  x
     1.0675    0.7494    0.0391
     0.7494    1.3087   -0.2480
     0.0391   -0.2480    0.8270

B = 3x3
10^-11  x
    -0.0909    0.0909    0.0909
     0.2728   -0.3638         0
          0         0   -0.0909

D = 3x3
10^4  x
     3.0128    2.7416    0.6144
     2.7416    5.7447   -0.4836
     0.6144   -0.4836    3.0268
```

Stacking Sequence 3:

```
A = 3×3
10^5 ×
      1.4164    0.2799    0.2412
      0.2799    1.8989    0.2412
      0.2412    0.2412    0.3575

B = 3×3
10^-11 ×
     -0.3183   -0.0455         0
     -0.0455         0         0
           0         0   -0.0227

D = 3×3
10^4 ×
      4.2375    1.1279    0.9951
      1.1279    7.7474    0.9951
      0.9951    0.9951    1.4131
```