Snowboard Body Design

Written By:

Sam Masten

Submitted as partial fulfillment of the requirements for Aerospace Structures II, MAE 472-001

North Carolina State University

Mechanical and Aerospace Engineering

Date Of Submission: 05/02/2023

We have neither given nor received unauthorized aid on this assignment.

SM

- **Problem Formulation:**

Composite structures are materials composed of two or more individual materials, or constituents. There are countless composites that are used every day in the aerospace industry. To name a few, fuselages, wings, body panels, pressure vessels, and motor casings are all composite structures. However, they are used in nearly all other industries as well. One common item that people might not realize is a composite structure is a snowboard.

Snowboards need to be both lightweight and slick, but also strong and durable, which makes them a perfect candidate for a composite structure. This is because composite materials are constructed using a combination of reinforcement and matrix materials that have differing but complementary characteristics, allowing a new material with desired properties to be produced. Today there are dozens of variations of composites in snowboard design.

This report will examine the composite structure, material properties, applied loading, and potential layups of a rocker-camber snowboard, pictured below. These snowboards are very common today and are good examples of composite materials in the real world.

Figure 1: Rocker-Camber Snowboard Model

To analyze the applied loading of this structure, the snowboard will be examined when laying flat on the snow at the bottom of a slope traveling at a constant velocity. This yields the following free body diagram, shown below in Figure 2.

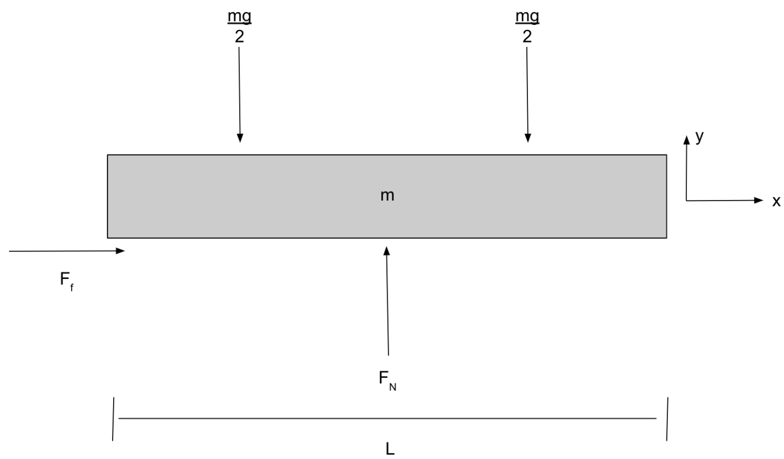

Figure 2: Free Body Diagram

The variables in the free body diagram above are as follows:

- m: Mass of snowboard
- mg/2: Weight of the rider on the board
- L: Total length of the snowboard
- F_N: Normal force applied by the ground
- F_f: Force of friction against the snow

The average weight of a human is 62 kg, the typical snowboard length is 1.5 meters, and the typical coefficient of static friction for a snowboard is around 0.1 (Lematta 2014). These values can be substituted and used to find the forces and moments applied to the snowboard.

The following are the equations of motion of the system:

$F_x = F_f = \mu_s F_N = \mu_s mg = 0.1(62)(9.81) = 60.822$ N
$F_y = 2(0.5mg) - F_N = 2(0.5)(9.81)(62) - (9.81)(62) = 0$ N
$F_z = 0$ N
$M_x = 0$ N/m
$M_y = 0$ N/m
$M_z = 0$ N/m

Each of the moments about the axes are equal to zero because the system is examined when the board is riding straight. If the snowboarder was twisting or rotating, initiating a turn, or applying a force while in the air, there would be resulting torsional, edge, or off-axis moments. However, in this case, there are only forces in the x and y-directions.

Snowboards need to be strong and durable, but also lightweight and flexible. For the design being described in this report, a composite material made of four layers will be created. This way, the board will be thick enough to provide the necessary strength but not so thick that it sacrifices the required flexibility. An example of a laminae similar to this is shown below in Figure 3.

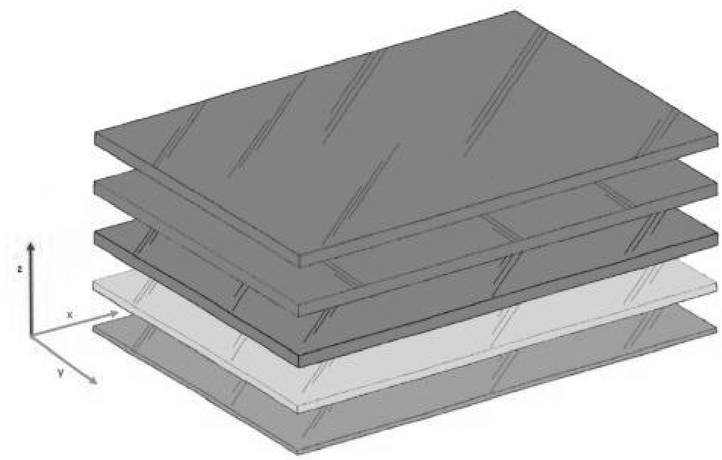

Figure 3: Five-Layer Laminae Example

Every composite material is made up of matrix and reinforcement, or fiber, materials. In the snowboard manufacturing industry, it is common for the fiber material to be carbon or glass. After careful cross examination of these materials, it is decided that AS-4 carbon fiber is the best choice for a lightweight snowboard. It provides the necessary strength and durability, but is not so strong that it becomes rigid. The matrix material of a board is typically some variation of epoxy resin. There are three types of epoxy, but this design will use Epoxy 3501-6, because it is highly adaptable and provides excellent bending capabilities.

The composite material will be assembled by placing alternating carbon and epoxy layers, for a total of four layers. The orientation of these layers varies by board manufacturer in the snowboarding industry, but for this report multiple cases with different layer orientations will be analyzed.

With this information in mind, the necessary material properties of the laminae can be found. For carbon AS-4, the E and G values are 235 GPa and 20 GPa and for epoxy 3501-6, the E and G values are 4.3 GPa and 1.6 GPa (Isaac et. al 2007). For a typical composite material, the fiber material makes up 50-70% of the laminae. For this design, median is used, and a V_f value of 0.6 is assigned. To make this laminae as efficient as possible, it will need to be an ideal composite, so the sum of V_f and V_m will equal 1. This yields a V_m value of 0.4. By substituting these values into the E_1, E_2, ν_{12}, and G_{12} equations found in the Appendix, the following material properties are yielded.

E_1 (GPa)	E_2 (GPa)	ν_{12}	G_{12} (GPa)
135.8	9.763	0.0816	3.26

Table 1: Material Properties of the Laminae

The C matrix for these material properties in GPa is as follows.

135.8	0.80	0.00
0.80	9.77	0.00
0.00	0.00	3.26

Table 2: C Matrix of Laminae

With the laminae designed and material properties calculated, different layups can be chosen. For this study, three different layups are analyzed: $[90\ 0]_2$, $[0\ 90]_S$, and [0 -45 45 90]. Each of these layups offer their own advantages for why they might yield the best possible laminate with the most deal properties. These advantages are described below.

A $[90\ 0]_2$, or [90 0 00 0] laminate is a cross-ply, unsymmetric laminate. In a cross-ply laminate the C_{16} and C_{26} C-bar values are equal to zero. This laminate sequence will have no shear-extension coupling, but will have in and out of plane coupling since neither the A or D matrices are isotropic. The A_{16}, A_{26}, D_{16}, and D_{26} values are all zero, but the rest of the matrix values are nonzero. Because there are no moments about the axis in the system, there is no twisting that needs to be counteracted. If there was an x-axis moment, then the ABD matrices would need to be manipulated such that the nonzero terms counteract the nonzero moment during the matrix multiplication used to generate the strain and curvature values.

A $[0\ 90]_S$, or [0 90 0 90] laminate is a cross-ply, symmetric laminate. In this laminate sequence, the C_{16} and C_{26} C-bar values are equal to zero and the B matrix is also equal to zero. The reason a second cross-ply sequence was chosen was that they have fewer coupling and loading issues to counteract, which is ideal for the snowboard system. This sequence will have extension coupling but no in or out of plane coupling. This case is very similar to case 1 but is symmetric, giving it a zero B matrix which could result in lower strain and stress values.

The third case chosen is a [0 -45 45 90] sequence, which is a quasi-isotropic, unsymmetric laminate. In quasi-isotropic laminates, the A_{11} and A_{22} values are equal to one another and thus the in-plane is isotropic. This results in no shear-extension coupling in the in-plane, but the non-isotropic D matrix will still yield shear-extension coupling in the out-of-plane. For these reasons, it is possible that this case could prove to be more favorable than the cross-ply sequences.

- **Simulations and Results:**

To determine which of these layups was the most optimal choice, the ABD matrices were calculated and used to plot the strains and stresses through the thicknesses of the laminates. Using the MATLAB code provided in the Appendix, the ABD matrices for each case were yielded and can be found at the end of the Appendix.

From these matrices, it can be determined that case 3 is likely the worst choice. Each case has zero values for A_{16}, A_{26}, D_{16}, and D_{26}, but in case 3 the rest of the ABD matrices are nonzero values, which results in more extension coupling and loading than in cases 2 and 3. Without the proper counteracting measures, these could lead to bends or breaks in the board. Cases 1 and 2 had very similar ABD matrices apart from minor value changes in the B matrix. From the ABD matrices alone, it is hard to tell which is the better choice; however, the ABD matrices were used to generate the following strain plots.

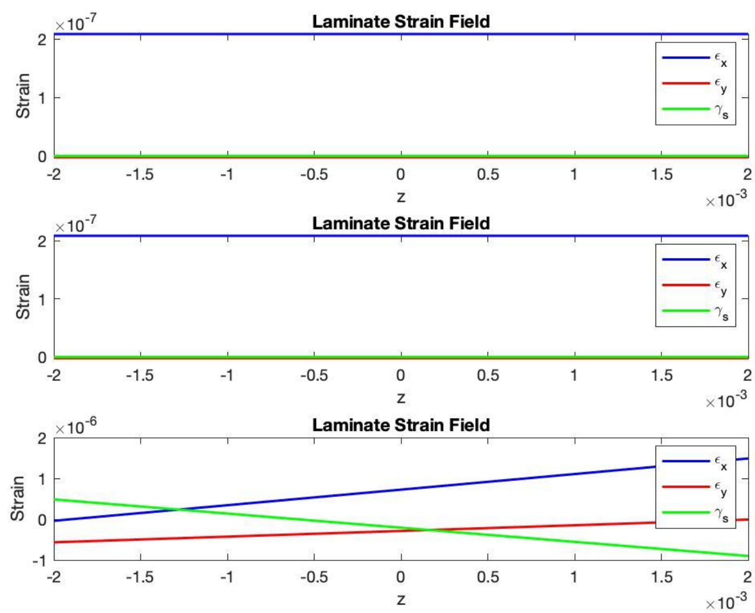

Figure 3: Strain Curves for Layups 1-3

From these strain plots, it can be determined that cases 1 and 2 have no strain in the y-direction, but case 3 has linearly varying strains in all directions. The plots also show that the strain values in cases 1 and 2 are constant and the same. Additionally, the strain values are very small, which is favorable for snowboards that are forced to experience a large amount of wear and tear. The stress plots are also generated, shown below.

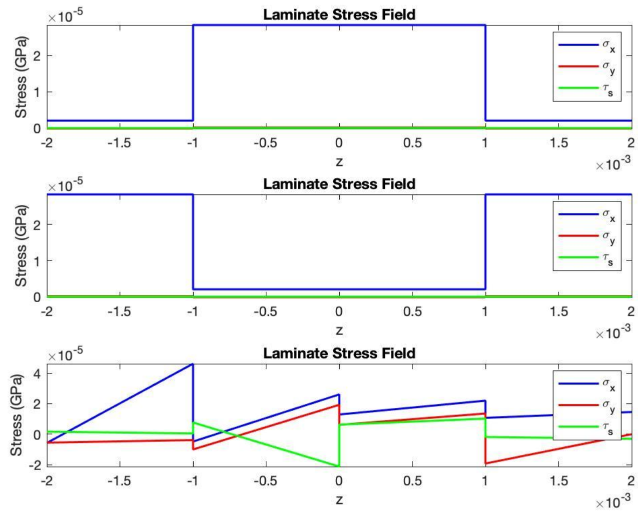

Figure 4: Stress Curves for Layups 1-3

From these stress plots, it can be determined that, similar to the strains, case 3 is the only choice with stresses in all three directions. It also has linearly varying strains whereas the strains in cases 1 and 2 stay constant between layers.The only difference between cases 1 and 2 is that case 2 has nonzero stress values on the ends of the laminate while case 1 has nonzero stress values in the middle of the laminate.

- **Conclusion:**

The purpose of this report was to design a laminae made up of a composite material and analyze different layups to determine the best laminate design for a snowboard. A multitude of techniques were used throughout this project such as applied loading and moments, calculation of material properties, layup analysis, and strain and stress plotting.

The analysis concludes that Case 2 is the best case and should be considered for a final design of the laminate. Although both case 1 and case 2 have favorable coupling and loading behaviors and fewer stresses and strains through the thicknesses, Case 2 has zero stress values in the middle of the laminate, which is favorable as the board wears down and the inner layers become exposed. It is important to note that case and case 2 are nearly identical in terms of stresses, strains, and ABD matrices. When choosing different scenarios, it was originally thought that symmetry would make more of a difference. It can be concluded that symmetry does not make a large difference, particularly when other properties are the same. In this case, they were both cross-ply laminates, so although only one was symmetric, they still had many similar values.

References

Daniel, Isaac, and Ori Ishai. *Engineering Mechanics of Composite Materials*. Oxford University Press, 2007.

Lematta, Craig. Physics of Snowboarding. 2014. http://ffden-2.phys.uaf.edu/webproj/212_spring_2014/Craig_Lematta/craig_lematta/page1.html

- **Appendix:**

```
%% MAE472 Final Project
% Sam Masten

%% Part 2

%Given Values

%d_m = 7; %Carbon AS-4 in um
%d_m2 = 10; %S-glass in um
%vol_m = 0; %total volume of matrix in material
%vol_r = 0; %total volume of reinforcement/fiber in material
%vol_tot = vol_m + vol_r; %total volume of material

%V_f = vol_r / vol_tot; %fiber volume fraction
%V_m = vol_m / vol_tot; %matrix volume fraction
V_f = 0.6;
V_m = 0.4;

vol_tot = 4*(.20^2); %example laminate volume (4r^2)
vol_f = vol_tot *V_f;
vol_m = vol_tot *V_m;

E_f = 235e9; %Carbon AS-4 in GPa
G_f = 20e9; %Carbon AS-4 in GPa
E_m = 4.3e9; %Epoxy 3501-6 in GPa
G_m = 1.6e9; %Epoxy 3501-6 in GPa

%E1, E2, v12, G12

E1 = E_f*V_f + E_m*V_m;
E2 = (E_f*E_m) / ( E_f*V_m + E_m*V_f );
v12 = vol_m*V_m + vol_f*V_f;
G12 = (G_f*G_m) / ( V_m*G_f + V_f*G_m );


%% Problem 1

%Given Values
E1 = E1; %Pa
E2 = E2; %Pa
G12 = G12; %Pa
v12 = v12;
v21 = v12 * (E2/E1);

%% Calculating the C Matrix

C11 = E1/(1-(v12*v21));
C12 = (v12*E2)/(1-(v12*v21));
C16= 0;
C22 = E2/(1-(v12*v21));
C26 = 0;
C66 = G12;
```

```
C = [C11 C12 C16; C12 C22 C26; C16 C26 C66];

fprintf('\nThe C matrix in GPa is\n');
fprintf('%14.4f %14.4f %14.4f\n', C/10^9)


%% Part 3

%Inputted Values
n = 4;

t = 0.001; %m, thickness of one ply
h = 0.004; %m, height
theta = [90 0 0 90];

%% Calculating the C Bar Matrix

Cbar = {};

for i = 1:length(theta)
   c = cosd(theta(i));
   s = sind(theta(i));

   C_11(i) = C11*c^4+C22*s^4+(2*C12+4*C66)*c^2*s^2;
   C_12(i) = C12*(c^4+s^4)+(C11+C22-4*C66)*c^2*s^2;
   C_22(i) = C11*s^4+C22*c^4+(2*C12+4*C66)*c^2*s^2;
   C_16(i) = (C11-C12-2*C66)*c^3*s-(C22-C12-2*C66)*c*s^3;
   C_26(i) = (C11-C12-2*C66)*c*s^3-(C22-C12-2*C66)*c^3*s;
   C_66(i) = (C11+C22-2*C12-2*C66)*c^2*s^2+C66*(c^4+s^4);

   Cbar{i} = [C_11(i) C_12(i) C_16(i); C_12(i) C_22(i) C_26(i); C_16(i) C_26(i) C_66(i)];

   fprintf('\nC Bar Matrix (%.1f degrees) in GPa\n',theta(i))
   fprintf('%14.4f %14.4f %14.4f\n', Cbar{i}/(10^9))

end



%% Calculating the z values

%Accounting for varying thickness
for i = 1:n+1
   z(i) = -h/2 + (t*(i-1));
end

%% Creating each Z vector

z1 = [];
z2 = [];
z3 = [];

for i = 1:n
   z1_loop = z(i+1)-z(i);
   z2_loop = z(i+1)^2 - z(i)^2;
   z3_loop = z(i+1)^3 - z(i)^3;
   z1 = [z1 z1_loop];
   z2 = [z2 z2_loop];
   z3 = [z3 z3_loop];
end
```

```
%% Calculating the A,B,D Matrices

for i = 1:n
   coeff = 1/3;
   A11(i) = C_11(i).*z1(i);
   A12(i) = C_12(i).*z1(i);
   A16(i) = C_16(i).*z1(i);
   A22(i) = C_22(i).*z1(i);
   A26(i) = C_26(i).*z1(i);
   A66(i) = C_66(i).*z1(i);
   B11(i) = 0.5*C_11(i)*z2(i);
   B12(i) = 0.5*C_12(i)*z2(i);
   B16(i) = 0.5*C_16(i)*z2(i);
   B22(i) = 0.5*C_22(i)*z2(i);
   B26(i) = 0.5*C_26(i)*z2(i);
   B66(i) = 0.5*C_66(i)*z2(i);
   D11(i) = coeff*C_11(i)*z3(i);
   D12(i) = coeff*C_12(i)*z3(i);
   D16(i) = coeff*C_16(i)*z3(i);
   D22(i) = coeff*C_22(i)*z3(i);
   D26(i) = coeff*C_26(i)*z3(i);
   D66(i) = coeff*C_66(i)*z3(i);
end

A = [sum(A11) sum(A12) sum(A16); sum(A12) sum(A22) sum(A26); sum(A16) sum(A26) sum(A66)];
B = [sum(B11) sum(B12) sum(B16); sum(B12) sum(B22) sum(B26); sum(B16) sum(B26) sum(B66)];
D = [sum(D11) sum(D12) sum(D16); sum(D12) sum(D22) sum(D26); sum(D16) sum(D26) sum(D66)];

%% Outputs

fprintf('A Matrix in N/m\n')
A

fprintf('B Matrix in N\n')
B

fprintf('D Matrix in N-m\n')
D

%% Applied Loads

Nx = .060822; %kN/m
Ny = 0; %kN/m
Ns = 0; %kN/m
Mx = 0; %kN
My = 0; %kN
Ms = 0; %kN

%Load Vector/Matrix
NM_mat = [Nx; Ny; Ns; Mx; My; Ms];
NM_mat = NM_mat * (10^3); %converting kN to N

%% Finding Strain Values

%Solving for Midplane Strain & Curvatures
F = [A B; B D]^(-1);
%changed N to kN

midplane_mat = F*NM_mat;

%Extracting each variable
```

```
epsilon_x0 = midplane_mat(1);
epsilon_y0 = midplane_mat(2);
epsilon_s0 = midplane_mat(3);
k_x = midplane_mat(4);
k_y = midplane_mat(5);
k_s = midplane_mat(6);

%Forming Vectors
strain0_mat = [epsilon_x0; epsilon_y0; epsilon_s0];
k_mat = [k_x; k_y; k_s]; %convert to /mm

%% Calculating Strains at Other Distances from Midplane

epsilon_x = strain0_mat(1) + z*k_mat(1);
epsilon_y = strain0_mat(2) + z*k_mat(2);
gamma_s = strain0_mat(3) + z*k_mat(3);


%% Calculating Stresses at Other Distances from Midplane

z_new = [z(1) z(2) z(2) z(3) z(3) z(4) z(4) z(5)];

for j = 1:n

    spot = 2*j-1;
    spot2 = 2*j;

    c11 = C_11(j)*(10^-9);
    c12 = C_12(j)*(10^-9);
    c16 = C_16(j)*(10^-9);
    c22 = C_22(j)*(10^-9);
    c26 = C_26(j)*(10^-9);
    c66 = C_66(j)*(10^-9);

    ex = epsilon_x(j);
    ey = epsilon_y(j);
    gs = gamma_s(j);

    ex2 = epsilon_x(j+1);
    ey2 = epsilon_y(j+1);
    gs2 = gamma_s(j+1);

    sigma_x(spot) = [c11*ex + c12*ey + c16*gs];
    sigma_y(spot) = [c12*ex + c22*ey + c26*gs];
    tau_s(spot) = [c16*ex + c26*ey + c66*gs];

    sigma_x(spot2) = [c11*ex2 + c12*ey2 + c16*gs2];
    sigma_y(spot2) = [c12*ex2 + c22*ey2 + c26*gs2];
    tau_s(spot2) = [c16*ex2 + c26*ey2 + c66*gs2];

end

%% Plots

%% Strain Field Plot

figure(1)
subplot(3,1,1)
plot(z,epsilon_x,'b','LineWidth',1.5)
hold on;
plot(z,epsilon_y,'r','LineWidth',1.5)
```

```
hold on;
plot(z,gamma_s,'g','LineWidth',1.5)
title('Laminate Strain Field')
xlabel('z')
ylabel('Strain')
legend('{\epsilon}_{x}','{\epsilon}_{y}','{\gamma}_{s}')


%% Stress Field Plot

figure(2)
subplot(3,1,1)
plot(z_new,sigma_x,'b','LineWidth',1.5)
hold on;
plot(z_new,sigma_y,'r','LineWidth',1.5)
hold on;
plot(z_new,tau_s,'g','LineWidth',1.5)
title('Laminate Stress Field')
xlabel('z')
ylabel('Stress (GPa)')
legend('{\sigma}_{x}','{\sigma}_{y}','{\tau}_{s}')


%% LAYUP TWO


%Inputted Values
n = 4;

t = 0.001; %m, thickness of one ply
h = 0.004; %m, height
theta = [0 90 90 0];

%% Calculating the C Bar Matrix

Cbar = {};

for i = 1:length(theta)
    c = cosd(theta(i));
    s = sind(theta(i));

    C_11(i) = C11*c^4+C22*s^4+(2*C12+4*C66)*c^2*s^2;
    C_12(i) = C12*(c^4+s^4)+(C11+C22-4*C66)*c^2*s^2;
    C_22(i) = C11*s^4+C22*c^4+(2*C12+4*C66)*c^2*s^2;
    C_16(i) = (C11-C12-2*C66)*c^3*s-(C22-C12-2*C66)*c*s^3;
    C_26(i) = (C11-C12-2*C66)*c*s^3-(C22-C12-2*C66)*c^3*s;
    C_66(i) = (C11+C22-2*C12-2*C66)*c^2*s^2+C66*(c^4+s^4);

    Cbar{i} = [C_11(i) C_12(i) C_16(i); C_12(i) C_22(i) C_26(i); C_16(i) C_26(i) C_66(i)];

    fprintf('\nC Bar Matrix (%.1f degrees) in GPa\n',theta(i))
    fprintf('%14.4f %14.4f %14.4f\n', Cbar{i}/(10^9))

end



%% Calculating the z values

%Accounting for varying thickness
for i = 1:n+1
```

```
    z(i) = -h/2 + (t*(i-1));
end

%% Creating each Z vector

z1 = [];
z2 = [];
z3 = [];

for i = 1:n
    z1_loop = z(i+1)-z(i);
    z2_loop = z(i+1)^2 - z(i)^2;
    z3_loop = z(i+1)^3 - z(i)^3;
    z1 = [z1 z1_loop];
    z2 = [z2 z2_loop];
    z3 = [z3 z3_loop];
end

%% Calculating the A,B,D Matrices

for i = 1:n
    coeff = 1/3;
    A11(i) = C_11(i).*z1(i);
    A12(i) = C_12(i).*z1(i);
    A16(i) = C_16(i).*z1(i);
    A22(i) = C_22(i).*z1(i);
    A26(i) = C_26(i).*z1(i);
    A66(i) = C_66(i).*z1(i);
    B11(i) = 0.5*C_11(i)*z2(i);
    B12(i) = 0.5*C_12(i)*z2(i);
    B16(i) = 0.5*C_16(i)*z2(i);
    B22(i) = 0.5*C_22(i)*z2(i);
    B26(i) = 0.5*C_26(i)*z2(i);
    B66(i) = 0.5*C_66(i)*z2(i);
    D11(i) = coeff*C_11(i)*z3(i);
    D12(i) = coeff*C_12(i)*z3(i);
    D16(i) = coeff*C_16(i)*z3(i);
    D22(i) = coeff*C_22(i)*z3(i);
    D26(i) = coeff*C_26(i)*z3(i);
    D66(i) = coeff*C_66(i)*z3(i);
end

A = [sum(A11) sum(A12) sum(A16); sum(A12) sum(A22) sum(A26); sum(A16) sum(A26) sum(A66)];
B = [sum(B11) sum(B12) sum(B16); sum(B12) sum(B22) sum(B26); sum(B16) sum(B26) sum(B66)];
D = [sum(D11) sum(D12) sum(D16); sum(D12) sum(D22) sum(D26); sum(D16) sum(D26) sum(D66)];

%% Outputs

fprintf('A Matrix in N/m\n')
A

fprintf('B Matrix in N\n')
B

fprintf('D Matrix in N-m\n')
D

%% Applied Loads

Nx = .060822; %kN/m
Ny = 0; %kN/m
```

```
Ns = 0; %kN/m
Mx = 0; %kN
My = 0; %kN
Ms = 0; %kN

%Load Vector/Matrix
NM_mat = [Nx; Ny; Ns; Mx; My; Ms];
NM_mat = NM_mat * (10^3); %converting kN to N

%% Finding Strain Values

%Solving for Midplane Strain & Curvatures
F = [A B; B D]^(-1);
%changed N to kN

midplane_mat = F*NM_mat;

%Extracting each variable
epsilon_x0 = midplane_mat(1);
epsilon_y0 = midplane_mat(2);
epsilon_s0 = midplane_mat(3);
k_x = midplane_mat(4);
k_y = midplane_mat(5);
k_s = midplane_mat(6);

%Forming Vectors
strain0_mat = [epsilon_x0; epsilon_y0; epsilon_s0];
k_mat = [k_x; k_y; k_s]; %convert to /mm

%% Calculating Strains at Other Distances from Midplane

epsilon_x = strain0_mat(1) + z*k_mat(1);
epsilon_y = strain0_mat(2) + z*k_mat(2);
gamma_s = strain0_mat(3) + z*k_mat(3);


%% Calculating Stresses at Other Distances from Midplane

z_new = [z(1) z(2) z(2) z(3) z(3) z(4) z(4) z(5)];

for j = 1:n

   spot = 2*j-1;
   spot2 = 2*j;

   c11 = C_11(j)*(10^-9);
   c12 = C_12(j)*(10^-9);
   c16 = C_16(j)*(10^-9);
   c22 = C_22(j)*(10^-9);
   c26 = C_26(j)*(10^-9);
   c66 = C_66(j)*(10^-9);

   ex = epsilon_x(j);
   ey = epsilon_y(j);
   gs = gamma_s(j);

   ex2 = epsilon_x(j+1);
   ey2 = epsilon_y(j+1);
   gs2 = gamma_s(j+1);

   sigma_x(spot) = [c11*ex + c12*ey + c16*gs];
```

```
    sigma_y(spot) = [c12*ex + c22*ey + c26*gs];
    tau_s(spot) = [c16*ex + c26*ey + c66*gs];

    sigma_x(spot2) = [c11*ex2 + c12*ey2 + c16*gs2];
    sigma_y(spot2) = [c12*ex2 + c22*ey2 + c26*gs2];
    tau_s(spot2) = [c16*ex2 + c26*ey2 + c66*gs2];

end

%% Plots

%% Strain Field Plot

figure(1)
subplot(3,1,2)
plot(z,epsilon_x,'b','LineWidth',1.5)
hold on;
plot(z,epsilon_y,'r','LineWidth',1.5)
hold on;
plot(z,gamma_s,'g','LineWidth',1.5)
title('Laminate Strain Field')
xlabel('z')
ylabel('Strain')
legend('{\epsilon}_{x}','{\epsilon}_{y}','{\gamma}_{s}')


%% Stress Field Plot

figure(2)
subplot(3,1,2)
plot(z_new,sigma_x,'b','LineWidth',1.5)
hold on;
plot(z_new,sigma_y,'r','LineWidth',1.5)
hold on;
plot(z_new,tau_s,'g','LineWidth',1.5)
title('Laminate Stress Field')
xlabel('z')
ylabel('Stress (GPa)')
legend('{\sigma}_{x}','{\sigma}_{y}','{\tau}_{s}')


%% LAYUP THREE

%Inputted Values
n = 4;

t = 0.001; %m, thickness of one ply
h = 0.004; %m, height
theta = [0 -45 45 90];

%% Calculating the C Bar Matrix

Cbar = {};

for i = 1:length(theta)
    c = cosd(theta(i));
    s = sind(theta(i));

    C_11(i) = C11*c^4+C22*s^4+(2*C12+4*C66)*c^2*s^2;
    C_12(i) = C12*(c^4+s^4)+(C11+C22-4*C66)*c^2*s^2;
    C_22(i) = C11*s^4+C22*c^4+(2*C12+4*C66)*c^2*s^2;
```

```
    C_16(i) = (C11-C12-2*C66)*c^3*s-(C22-C12-2*C66)*c*s^3;
    C_26(i) = (C11-C12-2*C66)*c*s^3-(C22-C12-2*C66)*c^3*s;
    C_66(i) = (C11+C22-2*C12-2*C66)*c^2*s^2+C66*(c^4+s^4);

    Cbar{i} = [C_11(i) C_12(i) C_16(i); C_12(i) C_22(i) C_26(i); C_16(i) C_26(i) C_66(i)];

    fprintf('\nC Bar Matrix (%.1f degrees) in GPa\n',theta(i))
    fprintf('%14.4f %14.4f %14.4f\n', Cbar{i}/(10^9))

end



%% Calculating the z values

%Accounting for varying thickness
for i = 1:n+1
    z(i) = -h/2 + (t*(i-1));
end

%% Creating each Z vector

z1 = [];
z2 = [];
z3 = [];

for i = 1:n
    z1_loop = z(i+1)-z(i);
    z2_loop = z(i+1)^2 - z(i)^2;
    z3_loop = z(i+1)^3 - z(i)^3;
    z1 = [z1 z1_loop];
    z2 = [z2 z2_loop];
    z3 = [z3 z3_loop];
end

%% Calculating the A,B,D Matrices

for i = 1:n
    coeff = 1/3;
    A11(i) = C_11(i).*z1(i);
    A12(i) = C_12(i).*z1(i);
    A16(i) = C_16(i).*z1(i);
    A22(i) = C_22(i).*z1(i);
    A26(i) = C_26(i).*z1(i);
    A66(i) = C_66(i).*z1(i);
    B11(i) = 0.5*C_11(i)*z2(i);
    B12(i) = 0.5*C_12(i)*z2(i);
    B16(i) = 0.5*C_16(i)*z2(i);
    B22(i) = 0.5*C_22(i)*z2(i);
    B26(i) = 0.5*C_26(i)*z2(i);
    B66(i) = 0.5*C_66(i)*z2(i);
    D11(i) = coeff*C_11(i)*z3(i);
    D12(i) = coeff*C_12(i)*z3(i);
    D16(i) = coeff*C_16(i)*z3(i);
    D22(i) = coeff*C_22(i)*z3(i);
    D26(i) = coeff*C_26(i)*z3(i);
    D66(i) = coeff*C_66(i)*z3(i);
end

A = [sum(A11) sum(A12) sum(A16); sum(A12) sum(A22) sum(A26); sum(A16) sum(A26) sum(A66)];
B = [sum(B11) sum(B12) sum(B16); sum(B12) sum(B22) sum(B26); sum(B16) sum(B26) sum(B66)];
```

```
D = [sum(D11) sum(D12) sum(D16); sum(D12) sum(D22) sum(D26); sum(D16) sum(D26) sum(D66)];

%% Outputs

fprintf('A Matrix in N/m\n')
A

fprintf('B Matrix in N\n')
B

fprintf('D Matrix in N-m\n')
D

%% Applied Loads

Nx = .060822; %kN/m
Ny = 0; %kN/m
Ns = 0; %kN/m
Mx = 0; %kN
My = 0; %kN
Ms = 0; %kN

%Load Vector/Matrix
NM_mat = [Nx; Ny; Ns; Mx; My; Ms];
NM_mat = NM_mat * (10^3); %converting kN to N

%% Finding Strain Values

%Solving for Midplane Strain & Curvatures
F = [A B; B D]^(-1);
%changed N to kN

midplane_mat = F*NM_mat;

%Extracting each variable
epsilon_x0 = midplane_mat(1);
epsilon_y0 = midplane_mat(2);
epsilon_s0 = midplane_mat(3);
k_x = midplane_mat(4);
k_y = midplane_mat(5);
k_s = midplane_mat(6);

%Forming Vectors
strain0_mat = [epsilon_x0; epsilon_y0; epsilon_s0];
k_mat = [k_x; k_y; k_s]; %convert to /mm

%% Calculating Strains at Other Distances from Midplane

epsilon_x = strain0_mat(1) + z*k_mat(1);
epsilon_y = strain0_mat(2) + z*k_mat(2);
gamma_s = strain0_mat(3) + z*k_mat(3);


%% Calculating Stresses at Other Distances from Midplane

z_new = [z(1) z(2) z(2) z(3) z(3) z(4) z(4) z(5)];

for j = 1:n

    spot = 2*j-1;
    spot2 = 2*j;
```

```
    c11 = C_11(j)*(10^-9);
    c12 = C_12(j)*(10^-9);
    c16 = C_16(j)*(10^-9);
    c22 = C_22(j)*(10^-9);
    c26 = C_26(j)*(10^-9);
    c66 = C_66(j)*(10^-9);

    ex = epsilon_x(j);
    ey = epsilon_y(j);
    gs = gamma_s(j);

    ex2 = epsilon_x(j+1);
    ey2 = epsilon_y(j+1);
    gs2 = gamma_s(j+1);

    sigma_x(spot) = [c11*ex + c12*ey + c16*gs];
    sigma_y(spot) = [c12*ex + c22*ey + c26*gs];
    tau_s(spot) = [c16*ex + c26*ey + c66*gs];

    sigma_x(spot2) = [c11*ex2 + c12*ey2 + c16*gs2];
    sigma_y(spot2) = [c12*ex2 + c22*ey2 + c26*gs2];
    tau_s(spot2) = [c16*ex2 + c26*ey2 + c66*gs2];

end

%% Plots

%% Strain Field Plot

figure(1)
subplot(3,1,3)
plot(z,epsilon_x,'b','LineWidth',1.5)
hold on;
plot(z,epsilon_y,'r','LineWidth',1.5)
hold on;
plot(z,gamma_s,'g','LineWidth',1.5)
title('Laminate Strain Field')
xlabel('z')
ylabel('Strain')
legend('{\epsilon}_{x}','{\epsilon}_{y}','{\gamma}_{s}')


%% Stress Field Plot

figure(2)
subplot(3,1,3)
plot(z_new,sigma_x,'b','LineWidth',1.5)
hold on;
plot(z_new,sigma_y,'r','LineWidth',1.5)
hold on;
plot(z_new,tau_s,'g','LineWidth',1.5)
title('Laminate Stress Field')
xlabel('z')
ylabel('Stress (GPa)')
legend('{\sigma}_{x}','{\sigma}_{y}','{\tau}_{s}')
```

Outputs

```
C Bar Matrix (90.0 degrees) in GPa
    10.4681      0.8709      0.0000
```

```
    0.8709    142.7925    0.0000
    0.0000      0.0000    3.5714

C Bar Matrix (0.0 degrees) in GPa
  142.7925      0.8709    0.0000
    0.8709     10.4681    0.0000
    0.0000      0.0000    3.5714

C Bar Matrix (0.0 degrees) in GPa
  142.7925      0.8709    0.0000
    0.8709     10.4681    0.0000
    0.0000      0.0000    3.5714

C Bar Matrix (90.0 degrees) in GPa
   10.4681      0.8709    0.0000
    0.8709    142.7925    0.0000
    0.0000      0.0000    3.5714
A Matrix in N/m

A =

   1.0e+08 *

    3.0652    0.0348         0
    0.0348    3.0652         0
         0         0    0.1429

B Matrix in N

B =

   1.0e-11 *

   -0.1819         0         0
         0         0         0
         0         0         0

D Matrix in N-m

D =

  144.0463    4.6451         0
    4.6451  673.3436         0
         0         0   19.0476


C Bar Matrix (0.0 degrees) in GPa
  142.7925      0.8709    0.0000
    0.8709     10.4681    0.0000
    0.0000      0.0000    3.5714

C Bar Matrix (90.0 degrees) in GPa
   10.4681      0.8709    0.0000
    0.8709    142.7925    0.0000
    0.0000      0.0000    3.5714

C Bar Matrix (90.0 degrees) in GPa
   10.4681      0.8709    0.0000
    0.8709    142.7925    0.0000
    0.0000      0.0000    3.5714
```

C Bar Matrix (0.0 degrees) in GPa

142.7925	0.8709	0.0000
0.8709	10.4681	0.0000
0.0000	0.0000	3.5714

A Matrix in N/m

A =

1.0e+08 *

3.0652	0.0348	0
0.0348	3.0652	0
0	0	0.1429

B Matrix in N

B =

1.0e-11 *

0	0	0
0	-0.1819	0
0	0	0

D Matrix in N-m

D =

673.3436	4.6451	0
4.6451	144.0463	0
0	0	19.0476

C Bar Matrix (0.0 degrees) in GPa

142.7925	0.8709	0.0000
0.8709	10.4681	0.0000
0.0000	0.0000	3.5714

C Bar Matrix (-45.0 degrees) in GPa

42.3221	35.1792	-33.0811
35.1792	42.3221	-33.0811
-33.0811	-33.0811	37.8797

C Bar Matrix (45.0 degrees) in GPa

42.3221	35.1792	33.0811
35.1792	42.3221	33.0811
33.0811	33.0811	37.8797

C Bar Matrix (90.0 degrees) in GPa

10.4681	0.8709	0.0000
0.8709	142.7925	0.0000
0.0000	0.0000	3.5714

A Matrix in N/m

A =

1.0e+08 *

2.3790	0.7210	0
0.7210	2.3790	0
0	0	0.8290

North Carolina State University
Department of Mechanical
& Aerospace Engineering

```
B Matrix in N

B =

   1.0e+05 *

   -1.9849    0.0000    0.3308
    0.0000    1.9849    0.3308
    0.3308    0.3308   -0.0000

D Matrix in N-m

D =

  385.8228   27.5172         0
   27.5172  385.8228         0
         0         0   41.9198

>>
```