# Indian Institute of Technology, Guwahati

**A**

**PROJECT**

**REPORT**

**ON**

**<u>ANALYSIS OF LAMINATE</u>**


**Submitted to:**
Dr. Debabrata Chakraborty

**Submitted by:**
Manpreet Singh
204103313

**Department of Mechanical Engineering, IITG**

# (ANALYSIS OF LAMINATE)

## 1) Input Parameters:

Following two table combined together define the properties of material of laminate plies (from Excel file)

| Define Properties of Laminate | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| S. No | Ply Angle in Degree | Ply Thickness in meter | Engineering Constant (GPa) | | | | Co-efficient of Thermal Expansion (m/m/°C) | |
| | (Θ°) | t | $E_1$ | $E_2$ | $\nu_{12}$ | $G_{12}$ | $\alpha_1$ | $\alpha_2$ |
| 1 | 0 | 0.000125 | 3.86E+10 | 8.27E+09 | 0.28 | 4.14E+09 | 8.16E-06 | 2.21E-05 |
| 2 | 45 | 0.000125 | 3.86E+10 | 8.27E+09 | 0.28 | 4.14E+09 | 8.16E-06 | 2.21E-05 |
| 3 | -45 | 0.000125 | 3.86E+10 | 8.27E+09 | 0.28 | 4.14E+09 | 8.16E-06 | 2.21E-05 |
| 4 | 90 | 0.000125 | 3.86E+10 | 8.27E+09 | 0.28 | 4.14E+09 | 8.16E-06 | 2.21E-05 |
| 5 | 90 | 0.000125 | 3.86E+10 | 8.27E+09 | 0.28 | 4.14E+09 | 8.16E-06 | 2.21E-05 |
| 6 | -45 | 0.000125 | 3.86E+10 | 8.27E+09 | 0.28 | 4.14E+09 | 8.16E-06 | 2.21E-05 |
| 7 | 45 | 0.000125 | 3.86E+10 | 8.27E+09 | 0.28 | 4.14E+09 | 8.16E-06 | 2.21E-05 |
| 8 | 0 | 0.000125 | 3.86E+10 | 8.27E+09 | 0.28 | 4.14E+09 | 8.16E-06 | 2.21E-05 |

| Define Properties of Laminate | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| S. No | Ply Angle in Degree | Co-efficient of Moisture Expansion (m/m/kg/kg) | | Material Strength Parameters (MPa) | | | | |
| | (Θ°) | $\beta_1$ | $\beta_2$ | $(\sigma_1)^T_u$ | $(\sigma_1)^C_u$ | $(\sigma_2)^T_u$ | $(\sigma_2)^C_u$ | $(\tau_{12})_u$ |
| 1 | 0 | 0.09 | 0.3 | 1.06E+09 | 6.10E+08 | 3.10E+07 | 1.18E+08 | 7.20E+07 |
| 2 | 45 | 0.09 | 0.3 | 1.06E+09 | 6.10E+08 | 3.10E+07 | 1.18E+08 | 7.20E+07 |
| 3 | -45 | 0.09 | 0.3 | 1.06E+09 | 6.10E+08 | 3.10E+07 | 1.18E+08 | 7.20E+07 |
| 4 | 90 | 0.09 | 0.3 | 1.06E+09 | 6.10E+08 | 3.10E+07 | 1.18E+08 | 7.20E+07 |
| 5 | 90 | 0.09 | 0.3 | 1.06E+09 | 6.10E+08 | 3.10E+07 | 1.18E+08 | 7.20E+07 |
| 6 | -45 | 0.09 | 0.3 | 1.06E+09 | 6.10E+08 | 3.10E+07 | 1.18E+08 | 7.20E+07 |
| 7 | 45 | 0.09 | 0.3 | 1.06E+09 | 6.10E+08 | 3.10E+07 | 1.18E+08 | 7.20E+07 |
| 8 | 0 | 0.09 | 0.3 | 1.06E+09 | 6.10E+08 | 3.10E+07 | 1.18E+08 | 7.20E+07 |

## 2) Program Output :

| Angle of Ply | Failed Ply | Complete degradation Failure Load (N) | Partial degradation Failure Load (N) | Mode of failure |
|---|---|---|---|---|
| 0 | 4 | 75790 | 75790 | TT |
| 45 | 5 | 75790 | 75790 | TT |
| -45 | 2 | 175120 | 131360 | TT |
| 90 | 3 | 175120 | 131360 | TT |
| 90 | 6 | 175120 | 131360 | TT |
| -45 | 7 | 175120 | 131360 | TT |
| 45 | 1 | 265500 | 356670 | LT |
| 0 | 8 | 265500 | 356670 | LT |

## 3) MATAB Script:

Note: Program read data from excel sheet

Matlab code share link:
https://iitgoffice-my.sharepoint.com/:f:/g/personal/manpreetsingh_iitg_ac_in/EqQ5Ro9_GzVMhWpuYzK3ssgBgC7sGAISyafmYOZNoEDz0A

```
clearvars
clc
%--------------------------------------------------------------------------
% Failure Analysis of Laminate

% Assisgnment 6  ( Composite Materials ME 607 )

% By: Manpreet Singh
%     Roll No - 204103313
%--------------------------------------------------------------------------


% Input Parameters ->

del_T = 0; % Change in temperature in degree
del_C = 0; % Change in moisture content

force_vec = [100000; 0; 0; 0; 0; 0]; % Force vector N/m
```

```
% Reading input data file
data = readmatrix('Input_Data_File.xlsx');
data(:,1) = [];

%%  ------- OUTER LOOP OVER DEGRADATION CRITERIA -----------

for Degradation_criteria = 1: 2       % 1 -> complete Degradation
                                      % 2 -> partial Degradation

% Sorting given data ----->
    angle     = data(:,1);
    thickness = data(:,2);
    E_long    = data(:,3);
    E_tran    = data(:,4);
    nue12     = data(:,5);
    G         = data(:,6);
    alpha1    = data(:,7);
    alpha2    = data(:,8);
    beta1     = data(:,9);
    beta2     = data(:,10);
    nue21     = (nue12.*E_tran)./E_long;

    % Strength Paramters
    E1T       = data(:,11);
    E1C       = data(:,12);
    E2T       = data(:,13);
    E2C       = data(:,14);
    Tau12     = data(:,15);

    % additional paramters -->

    [No_of_plies, ~] = size(angle);       % Getting Number of plies in laminate
    Failure_Load = zeros(No_of_plies, 6); % to store failure load
  Failed_plies_order = zeros(3,No_of_plies); % Order of failure of plies
    Failed_plies_count = 0;                    % count of failed plies
    force_vec_update = 0;                      % to upadte failure load vector
    counter = 1;
    temporay = 1;


    %--------- Reference from mid plane or datum for each ply  ------------

    h_mid = -sum(thickness)/2;   % top surface height of laminate from mid plane
    h = zeros(No_of_plies+1,1);     % for calculation of ABD Matrix
    z = zeros(2*No_of_plies +1, 1);   % for stress and strain at mid of each ply
    h(1) = h_mid;
    z(1) = h_mid;

    for i = 1: No_of_plies
        z(2*i) = h(i) + 0.5*thickness(i);
        h_mid = thickness(i) + h_mid;
        h(1+i) = h_mid;
        z(2*i+1) = h(1+i);
    end
```

```
  %% ---------------------------------------------------------------------
  %  ----------------  LOOP OVER FAILURE OF EACH PLY --------------------
  % ----------------------------------------------------------------------

  while Failed_plies_count < No_of_plies  % <--  Main Loop for failure of
laminate


  %============ Calculating Q and Q-bar matrix for each ply =============

  Q     = zeros(3,3,No_of_plies);
  Q_bar = zeros(3,3,No_of_plies);

  for i = 1 : No_of_plies       % Q matrix for each ply

      % Q matrix
      Q11 =  E_long(i)/(1-nue12(i)*nue21(i));
      Q22 =  E_tran(i)/(1-nue12(i)*nue21(i));
      Q12 =  nue12(i)*E_tran(i)/ (1-nue12(i)*nue21(i));
      Q21 =  Q12;
      Q33 =  G(i);

      Q(:,:,i) = [Q11   Q12     0;
                  Q21   Q22     0;
                  0      0     Q33];

      % Q bar matrix
      theta = deg2rad(angle(i));

      Q_11 = Q11*cos(theta)^4 + Q22*sin(theta)^4 + 2*(Q12+2*Q33)*sin(theta)^2
*cos(theta)^2;
      Q_22 = Q11*sin(theta)^4 + Q22*cos(theta)^4 + 2*(Q12+2*Q33)*sin(theta)^2
*cos(theta)^2;

      Q_12 = (Q11 + Q22 - 4*Q33)*sin(theta)^2 *cos(theta)^2 + Q12*(cos(theta)^4
+ sin(theta)^4);
      Q_33 = (Q11 + Q22 -2*Q12 - 2*Q33)* sin(theta)^2 *cos(theta)^2 +
Q33*(sin(theta)^4 + cos(theta)^4);

      Q_13 = (Q11 - Q12 -2*Q33)*cos(theta)^3 *sin(theta) - (Q22 - Q12 -
2*Q33)*cos(theta)*sin(theta)^3;
      Q_23 = (Q11 - Q12 -2*Q33)*cos(theta)* sin(theta)^3 - (Q22 - Q12 -
2*Q33)*cos(theta)^3 *sin(theta);

      Q_bar(:,:,i) = [Q_11  Q_12  Q_13
                      Q_12  Q_22  Q_23
                      Q_13  Q_23  Q_33];


  end

      % ABD Matrix calculation -->
```

```
A = zeros(3,3);
B = zeros(3,3);
D = zeros(3,3);

for i = 1 : No_of_plies

    A = A + Q_bar(:,:,i) * (h(i+1) - h(i));
    B = B + Q_bar(:,:,i) * (h(i+1)^2 - h(i)^2);
    D = D + Q_bar(:,:,i) * (h(i+1)^3 - h(i)^3);

end

A;
B = B/2;
D = D/3;

% ABD matrix ----
ABD = [A, B
       B, D];

strain =   ABD\force_vec;      % mid plane strain and curvature vector

mid_strain = strain(1:3);      % mid plane strain
mid_curvature = strain(4:6);   % mid plane curvatures


%===== Global Stress - Strains at top-mid-bottom of each ply ======

Global_strain = zeros(3,3,No_of_plies);
Global_stress = zeros(3,3,No_of_plies);
j = 1;

for i =1: No_of_plies

    height = z(j:j+2); % distance of top-mid-bottom of ply from mid plane

    for k =1:3
        Global_strain(k,:,i) = mid_strain + height(k)*mid_curvature;
        Global_stress(k,:,i) = Q_bar(:,:,i) * Global_strain(k,:,i)';
    end
    j = j+2;

end


%====== Local Stress - Strains at top-mid-bottom of each ply ======

Local_strain = zeros(3,3,No_of_plies);
Local_stress = zeros(3,3,No_of_plies);

for i =1: No_of_plies
    a = angle(i);

    % Transfomation Matrix
```

```
        T = [[cosd(a)^2,          sind(a)^2,          2*sind(a)*cosd(a)    ]
            [sind(a)^2,           cosd(a)^2,         -2*sind(a)*cosd(a)    ]
            [-sind(a)*cosd(a),   sind(a)*cosd(a),    cosd(a)^2 - sind(a)^2]];

        for k = 1:3
            % Local Strain
            Global_strain(k,3,i) = Global_strain(k,3,i)/2;
            Local_strain(k,:,i) = T * Global_strain(k,:,i)';
            Local_strain(k,3,i) = 2*Local_strain(k,3,i);

            % Local Stress
            Local_stress(k,:,i) = T * Global_stress(k,:,i)';
        end

    end


    %=========== Thermal Stress - Strains for of each ply =============

    Global_thermal_coefficient = zeros(3, No_of_plies);
    Global_thermal_strain = zeros(3,No_of_plies);

    for i =1: No_of_plies
        a = angle(i);

        % Transfomation Matrix
        T = [[cosd(a)^2,          sind(a)^2,         -2*sind(a)*cosd(a)    ]
            [sind(a)^2,           cosd(a)^2,          2*sind(a)*cosd(a)    ]
            [sind(a)*cosd(a),   -sind(a)*cosd(a),    cosd(a)^2 - sind(a)^2]];

        Global_thermal_coefficient(:, i) = T * [alpha1(i); alpha2(i); 0];
        Global_thermal_coefficient(3, i)= 2* Global_thermal_coefficient(3, i);

        % Global Thermal Strain
        Global_thermal_strain(:,i) = del_T * Global_thermal_coefficient(:, i);

    end

    % ----------------------->
    % Equivalent thermal load

    Fx_thermal = zeros(3,1);    % Equivalent thermal Force
    Mx_thermal = zeros(3,1);    % Equivalent thermal Moment

    for i = 1 : No_of_plies

        Fx_thermal = Fx_thermal + Q_bar(:,:,i)*(h(i+1) -
h(i))*Global_thermal_coefficient(:, i);
        Mx_thermal = Mx_thermal + Q_bar(:,:,i)*(h(i+1)^2 -
h(i)^2)*Global_thermal_coefficient(:, i);

    end
    Fx_thermal = del_T * Fx_thermal;
    Mx_thermal = 0.5*del_T * Mx_thermal;
```

```
thermal_strain = ABD\[Fx_thermal; Mx_thermal];
mid_thermal_strain = thermal_strain(1:3);
mid_thermal_curvature = thermal_strain(4:6);


%=========== Moisture Stress - Strains for of each ply ============

Global_moisture_coefficient = zeros(3, No_of_plies);
Global_moisture_strain = zeros(3,No_of_plies);

for i =1: No_of_plies
    a = angle(i);

    % Transfomation Matrix
    T = [[cosd(a)^2,          sind(a)^2,         -2*sind(a)*cosd(a)    ]
         [sind(a)^2,          cosd(a)^2,          2*sind(a)*cosd(a)    ]
         [sind(a)*cosd(a),   -sind(a)*cosd(a),    cosd(a)^2 - sind(a)^2]];

    Global_moisture_coefficient(:, i)=T*[beta1(i); beta2(i); 0];
    Global_moisture_coefficient(3, i)=2*Global_moisture_coefficient(3, i);

    % Global moisture Strain
    Global_moisture_strain(:,i) = del_C*Global_moisture_coefficient(:, i);

end

% ------------------------>
% Equivalent moisture load

Fx_moisture = zeros(3,1);   % Equivalent moisture Force
Mx_moisture = zeros(3,1);   % Equivalent moisture Moment

for i = 1 : No_of_plies

    Fx_moisture = Fx_moisture + Q_bar(:,:,i)*(h(i+1) -
h(i))*Global_moisture_coefficient(:, i);
    Mx_moisture = Mx_moisture + Q_bar(:,:,i)*(h(i+1)^2 -
h(i)^2)*Global_moisture_coefficient(:, i);

end
Fx_moisture = del_C * Fx_moisture;
Mx_moisture = 0.5*del_C * Mx_moisture;

moisture_strain = ABD\[Fx_moisture; Mx_moisture];
mid_moisture_strain = moisture_strain(1:3);
mid_moisture_curvature = moisture_strain(4:6);


%================ Residual Strain and stresses ====================

Global_residual_strain = zeros(3, No_of_plies);
Global_residual_stress = zeros(3, No_of_plies);
Local_residual_stress = zeros(3, No_of_plies);
```

```
    for i =1: No_of_plies
        % Strain
        Global_residual_strain(:,i) = (mid_thermal_strain + (h(i+1) -
h(i))*mid_thermal_curvature) + (mid_moisture_strain + (h(i+1) -
h(i))*mid_moisture_curvature);
        Global_residual_strain(:,i) = Global_residual_strain(:,i) -
Global_thermal_strain(:,i) - Global_moisture_strain(:,i);
        % Stress
        Global_residual_stress(:,i) = Q_bar(:,:,i) *
Global_residual_strain(:,i);

        a = angle(i);

        % Transfomation Matrix
        T = [[cosd(a)^2,          sind(a)^2,          2*sind(a)*cosd(a)   ]
            [sind(a)^2,           cosd(a)^2,         -2*sind(a)*cosd(a)   ]
            [-sind(a)*cosd(a),   sind(a)*cosd(a),    cosd(a)^2 - sind(a)^2]];

        Local_residual_stress(:,i) = T*Global_residual_stress(:,i);

    end

    % Failure Strength considering residual stresses due to thermal and
    % moisture effects



    for i = 1: No_of_plies

        E1T(i)     = E1T(i) - Local_residual_stress(1,i);
        E1C(i)     = E1C(i) + Local_residual_stress(1,i);
        E2T(i)     = E2T(i) - Local_residual_stress(2,i);
        E2C(i)     = E2C(i) + Local_residual_stress(2,i);
        Tau12(i)   = Tau12(i) + Local_residual_stress(3,i);
    end

    %====================== Strength Ratio (SR) =======================

    SR = zeros(No_of_plies, 3);

    for i =1: No_of_plies

        for j = 1: 3

            if Local_stress(1, j, i)>0 && j == 1    % logitudinal tensile
                SR(i,j) = Local_stress(1, j, i) / E1T(i);

            elseif Local_stress(1, j, i)<0 && j == 1 % logitudinal compressive
                SR(i,j) = Local_stress(1, j, i) / E1C(i);

            elseif Local_stress(1, j, i)>0 && j == 2 % transverse tensile
                SR(i,j) = Local_stress(1, j, i) / E2T(i);
```

```
        elseif Local_stress(1, j, i)<0 && j == 2 % transverse compressive
            SR(i,j) = Local_stress(1, j, i) / E2C(i);

        else
            SR(i,j) = Local_stress(1, j, i) / Tau12(i); % in plane shear

        end
    end
end


%================ Failed plies and mode of failure ================

SR = round(SR, 10);
max_SR = max(max(abs(SR)));   % Max SR ratio
failed_ply = [];              % Plies failed
failure_mode = [];            % Mode of failure

for i = 1: No_of_plies

    for j = 1:3

        if abs(SR(i,j)) == max_SR

            failed_ply = [failed_ply; i];

            if SR(i,j) > 0 && j == 1
                failure_mode = [failure_mode; 1];  % longitudinal tensile

            elseif SR(i,j) < 0 && j ==1
                failure_mode=[failure_mode; 2]; % longitudinal compressive

            elseif SR(i,j) > 0 && j == 2
                failure_mode = [failure_mode; 3];  % transverse tensile

            elseif SR(i,j) < 0 && j == 2
                failure_mode=[failure_mode; 4];  % transverse compressive

            else
                failure_mode = [failure_mode; 5];  % Shear failure

            end
        end
    end
end

[s, ~] = size(failed_ply);

for i = 1 : s

    Failed_plies_order(1,temporay) = counter;  % -> order of failure
    Failed_plies_order(2,temporay) = failed_ply(i); % -> failed plies
    Failed_plies_order(3,temporay) =failure_mode(i);  % -> mode of failure
```

```matlab
            temporay = temporay + 1;

        end

        Failed_plies_count = Failed_plies_count + numel( failed_ply );    % Count
of no of failed plies


        %==================== Failure Load calculation ====================


        for i = 1: s    % s -> no of plies failed

            force_vec_update = force_vec_update + 1;
            Failure_Load(force_vec_update, :)=force_vec/max_SR; % Failure load
vector

        end

        if Degradation_criteria == 1
            % ================= For Complete degradation ==================

            [No_of_plies_failed, ~]  =  size(failed_ply);



            for i =1: No_of_plies_failed

                E_long(failed_ply(i))  = 0;
                E_tran(failed_ply(i))  = 0;
                G(failed_ply(i))  = 0;

            end


        else
            % ================== For Partial degradation ==================

            [No_of_plies_failed, ~]  =  size(failed_ply);

            for i =1: No_of_plies_failed

                if failure_mode(i)== 1 || failure_mode(i)==2 % ----> % logitudinal
                    E_long(failed_ply(i))  = 0;

                else %---------------------------------> % transverse
                    E_tran(failed_ply(i))  = 0;
                    G(failed_ply(i))       = 0;

                end
            end
```

```
        end
    end %-------< end of While loop over Failure of laminate

    if Degradation_criteria == 1
        disp('Failure load with complete degradation')
    else
        disp('Failure load with Partial degradation')
    end
    disp(Failure_Load(:,1));

end %----------< end of Degradation criteria Loop
```