\\SESG6039 – Composites Engineering Design and Mechanics - Individual Assignment 1
CLPT CALCULATOR

Name: *Abhinandan Thour*
Student ID: *32453515*

Question 1)
The code matches the results of the worked example as shown:

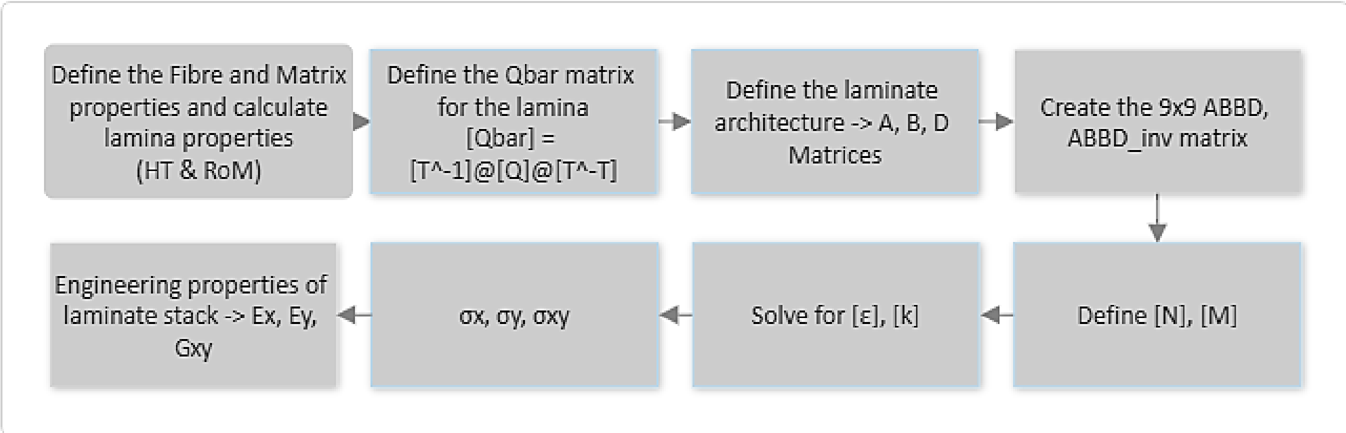

Figure 1 - Flow Chart of the CLPT Calculator

STEP 1 - Define the Fibre and Matrix properties and calculate lamina properties (HT & RoM)

Rule of Mixture:

```
------- Rule of Mixtures ---------
E11 = 153124000000.0 N/m^2
E22 = 4671136894.447 N/m^2
v12 = 0.281
v21 = 0.009
G12 = 1757041912.251 N/m^2
-------------------------------
```

Figure 2 - Results in the Worked example (RoM)

```
Using the RULE OF MIXTURE:
E11 is: 153.124 GN/m^2
E22 is: 4.671136894447375 GN/m^2
v12 is: 0.281
v21 is: 0.008572068828790474
G12 is: 1.7570419122510597 GN/m^2
```

Figure 2 - Results calculated (RoM)

```
# RULE OF MIXTURE
#Young Modulus Calculation
E1 = EMatrix*VMatrix + EFibre*VFibre  # Young's Modulus in the 1 direction
E2_RoM = (EFibre*EMatrix)/((VFibre*EMatrix)+(VMatrix*EFibre)) # Young's Modulus in the 2 direction

#Shear Modulus Calculation
GFibre = EFibre/(2*(1+vFibre)) # Shear Modulus of the Fibre
GMatrix = EMatrix/(2*(1+vMatrix)) # Shear Modulus of the Matrix
G12_RoM = (GFibre*GMatrix)/((GFibre*VMatrix) + (GMatrix*VFibre)) # Shear Modulus of the Laminate

#Major Poisson Ratio
v12 = vFibre*VFibre + vMatrix*VMatrix
#Minor Poisson Ratio
v21_RoM = v12 *(E2_RoM/E1)  # Minor Poisson Ration
```

Figure 3 - Code to calculate Properties (RoM)

Halpin-Tsai:

```
----- Halpin-Tsai  -----------
E11 = 153124000000.0 N/m^2
E22 = 5117175692.932 N/m^2
v12 = 0.281
v21 = 0.009
G12 = 2588319435.399 N/m^2
-------------------------------
```

Figure 3- Results on the Worked example (HT)

```
Using HALPIN-TSAI:
E11 is: 153.124 GN/m^2
E22 is: 5.117175692931722 GN/m^2
v12 is: 0.281
v21 is: 0.009390600883687822
G12 is: 2.588319435398565 GN/m^2
```

Figure 4 - Results calculated (HT)

```
# HALPIN TSAI
nE = ((EFibre/EMatrix)-1)/((EFibre/EMatrix) + S3)
E2 = EMatrix * ((1+(S3*nE*VFibre))/(1-(nE*VFibre))) # Young's Modulus in the 2 direction

nG = ((GFibre/GMatrix) - 1)/((GFibre/GMatrix) + 1)
G12 = GMatrix * ((1 + S12*nG*VFibre)/(1 - nG*VFibre)) # Shear Modulus of the Laminate

v21 = v12 *(E2/E1)  # Minor Poisson Ration
```

Figure 5 - Code to calculate Properties (HT)

STEP 2 - Define the Qbar matrix for the lamina -> [Qbar] = [T^-1]@[Q]@[T^-T]

- *Calculating Q matrix*

Q (GN/m^2)

$$\begin{bmatrix} 153.529 & 1.442 & 0 \\ 1.442 & 5.131 & 0 \\ 0 & 0 & 2.588 \end{bmatrix}$$

Figure 6 - Results of Q from worked Example.

```
 Q in GN/m^2:
[[153.529   1.442   0.    ]
 [  1.442   5.131   0.    ]
 [  0.      0.      2.588]]
```

Figure 7 - Results of Q calculated.

```python
# Qmatrix
Q11 = E1 / (1 - v12 * v21)
Q22 = E2 / (1 - v12 * v21)
Q12 = v12 * E2 / (1 - v12 * v21)
Q66 = G12
Q = np.array([[Q11, Q12, 0],
              [Q12, Q22, 0],
              [0, 0, Q66]])
Q = np.round(Q, 3)
print(f'\n Q in GN/m^2: \n{Q}')
```

Figure 8 - Code to calculate the Q matrix.

- *Calculating Qbar*

Qbar_0 (GN/m^2)

$$\begin{bmatrix} 153.529 & 1.442 & 0 \\ 1.442 & 5.131 & 0 \\ 0 & 0 & 2.588 \end{bmatrix}$$

Qbar_90 (GN/m^2)

$$\begin{bmatrix} 5.131 & 1.442 & 0 \\ 1.442 & 153.529 & 0 \\ 0 & 0 & 2.588 \end{bmatrix}$$

Figure 9 - QBar from worked example.

```
Qbar for angle 0 degrees in GN/m^2:
[153.529, 1.442, 0.0]
[1.442, 5.131, 0.0]
[0.0, 0.0, 2.588]

Qbar for angle 90 degrees in GN/m^2:
[5.131, 1.442, 0.0]
[1.442, 153.529, 0.0]
[0.0, 0.0, 2.588]
```

Figure 10 - QBar calculated.

```python
# Tmatrix
T11 = np.cos(theta) ** 2
T12 = np.sin(theta) ** 2
T13 = 2 * np.sin(theta) * np.cos(theta)
T21 = np.sin(theta) ** 2
T22 = np.cos(theta) ** 2
T23 = -2 * np.sin(theta) * np.cos(theta)
T31 = -np.sin(theta) * np.cos(theta)
T32 = np.sin(theta) * np.cos(theta)
T33 = np.cos(theta) ** 2 - np.sin(theta) ** 2

T = np.array([[T11, T12, T13],
              [T21, T22, T23],
              [T31, T32, T33]])

# Inverse of T matrix
T_inv = np.linalg.inv(T)

# Transpose of the inverse matrix
T_inv_transpose = np.transpose(T_inv)

# Qbar Matrix calculate dot product with T_Inv, Q, T_inv_t
Qbar = T_inv @ Q @ T_inv_transpose*1e9
Qbar = np.round(Qbar, 3)

# Converting the NumPy array to a regular Python list
Qbar_list.append(Qbar.tolist())
```

Figure 11 - Code to calculate the Qbar matrix.

To calculate various Qbar matrices, a for loop was employed to iterate through different angle orientations. Subsequently, the results were stored in separate lists defined prior to the start of the loop. This procedure was repeated for the [A], [B], and [D] matrices as well.

STEP 3 - Define the laminate architecture -> A, B, D Matrices

- *A Matrix Comparison*

A (N/m)

$$\begin{bmatrix} 317319681.503175 & 5766922.99794364 & 1.82185219030555 \cdot 10^{-10} \\ 5766922.99794364 & 317319681.503175 & 1.79913787977858 \cdot 10^{-8} \\ 1.82185219030555 \cdot 10^{-10} & 1.79913787977858 \cdot 10^{-8} & 10353277.7415943 \end{bmatrix}$$

```
The A Matrix in N/m is:
 [[3.17319682e+08 5.76692300e+06 0.00000000e+00]
 [5.76692300e+06 3.17319682e+08 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 1.03532777e+07]]
```

Figure 12 – A Matrix Comparison

- *B Matrix Comparison*

B (N)

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 4.54747350886464 \cdot 10^{-12} & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

```
The B Matrix in N is:
 [[0.00000000e+00 0.00000000e+00 0.00000000e+00]
 [0.00000000e+00 4.54747351e-12 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00]]
```

Figure 13 – B Matrix Comparison

- *D Matrix Comparison*

D (Nm)

$$\begin{bmatrix} 719.889732530147 & 7.68923066392485 & 6.07284063435182 \cdot 10^{-17} \\ 7.68923066392485 & 126.296084811653 & 5.99712626592862 \cdot 10^{-15} \\ 6.07284063435182 \cdot 10^{-17} & 5.99712626592862 \cdot 10^{-15} & 13.8043703221257 \end{bmatrix}$$

```
The D Matrix in Nm is:
 [[719.88973253   7.68923066   0.        ]
 [  7.68923066 126.29608481   0.        ]
 [  0.           0.          13.80437032]]
```

Figure 14 – D Matrix Comparison

```
# A Matrix
A1 = Qbar * (position[i + 1] - position[i])
A1_list.append(A1)

# B Matrix
B1 = (1/2)*Qbar * ((position[i + 1]**2) - (position[i]**2))
B1_list.append(B1)

# D Matrix
D1 = (1/3)*Qbar * ((position[i + 1]**3) - (position[i]**3))
D1_list.append(D1)
```

Figure 15 – Code used to generate [A], [B] and [D]

$$A_{ij} = \sum_{k=1}^{N} (Q_{ij})_k (Z_k - Z_{k-1})$$

$$B_{ij} = \frac{1}{2} \sum_{k=1}^{N} (Q_{ij})_k ({Z_k}^2 - {Z_{k-1}}^2)$$

$$D_{ij} = \frac{1}{3} \sum_{k=1}^{N} (Q_{ij})_k ({Z_k}^3 - {Z_{k-1}}^3)$$

Figure 16 – Formula provided to calculate [A], [B] and [D] Matrix Comparison

Step 4 - Create the 9x9 ABBD, ABBD_inv matrix

ABD

$$\begin{bmatrix} 317319681.503 & 5766922.998 & 0 & 0 & 0 & 0 \\ 5766922.998 & 317319681.503 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10353277.742 & 0 & 0 & 0 \\ 0 & 0 & 0 & 719.89 & 7.689 & 0 \\ 0 & 0 & 0 & 7.689 & 126.296 & 0 \\ 0 & 0 & 0 & 0 & 0 & 13.804 \end{bmatrix}$$

```
ABBD Matrix:
[[317319681.503 5766922.998 0 0 0 0]
 [5766922.998 317319681.503 0 0 0.000 0]
 [0 0 10353277.742 0 0 0]
 [0 0 0 719.890 7.689 0]
 [0 0.000 0 7.689 126.296 0]
 [0 0 0 0 0 13.804]]
```

Figure 17 – Comparison of the ABBD matrix

```
# Concatenating the A, B, and D matrices in the specified order
ABBD = np.block([[A, B], [B, D]])
print(f"ABBD Matrix: \n{ABBD} \n ") # Printing the ABBD matrix

# Inverse of ABBD Matrix
ABBD_inv = np.linalg.inv(ABBD)
print(f"ABBD_inv Matrix: \n{ABBD_inv}") # Printing the ABBD_inv matrix
print('\n ----------------- \n')
```

Figure 18 – Code to create ABBD and ABBD_inv Matrix

Step 5, 6 - Define [N], [M] and Solve for [ε], [k]

$$\begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \\ \kappa_x \\ \kappa_y \\ \kappa_{xy} \end{bmatrix} = \begin{bmatrix} A & B \\ B & D \end{bmatrix}^{-1} \begin{bmatrix} N_x \\ N_y \\ N_{xy} \\ M_x \\ M_y \\ M_{xy} \end{bmatrix}$$

Figure 19 – Correlation between strain, ABBD and NM Matrices

```
# Define the different directions
directions = ["Ex", "Ey", "Gxy"]
print("Engineering Properties:")
for direction in directions:
    NM = [0] * 6     # Initialize the NM Matrix

    if direction == "Ex":
        NM[0] = 1
    elif direction == "Ey":
        NM[1] = 1
    elif direction == "Gxy":
        NM[2] = 1

    NM = np.array(NM).reshape(6, 1)

    result = np.dot(ABBD_inv, NM) # Calculate the
    strain in different directions -> [εx, εy, γxy,
    κx, κy, κxy] vector
```

Figure 20 – Code to calculate strains in different directions

Step 7, 8 - σx, σy, σxy and Engineering properties of laminate stack -> Ex, Ey, Gxy

$$\sigma_x = \frac{A_{11}\epsilon_x + A_{12}\epsilon_y + A_{16}\gamma_{xy} + B_{11}\kappa_x + B_{12}\kappa_y + B_{16}\kappa_{xy}}{h}$$

$$E_x = \frac{\sigma_x}{\epsilon_x}$$

```
Young's modulus in the x-direction is   79303718565.13   N/m^2
Young's modulus in the y-direction is   79303718565.13   N/m^2
Shear modulus in the xy-plane is        2588319435.4     N/m^2
```

```
Engineering Properties:
Ex: [79303718565.126] N/m^2
Ey: [79303718565.126] N/m^2
Gxy: [2588319435.399] N/m^2
```

Figure 21 – Comparison of Engineering Properties

```
if direction == "Ex":
    Sx = ((A[0, 0] * ex + A[0, 1] * ey + A[0, 2]
    * yxy + B[0, 0] * kx + B[0, 1] * ky + B[0,
    2] * kxy)) / (nLayers * thickness)
    Ex = Sx / ex
    print(f"Ex: {Ex} N/m^2")
elif direction == "Ey":
    Sy = ((A[1, 0] * ex + A[1, 1] * ey + A[1, 2]
    * yxy + B[1, 0] * kx + B[1, 1] * ky + B[1,
    2] * kxy)) / (nLayers * thickness)
    Ey = Sy / ey
    print(f"Ey: {Ey} N/m^2")
elif direction == "Gxy":
    Sxy = ((A[2, 0] * ex + A[2, 1] * ey + A[2,
    2] * yxy + B[2, 0] * kx + B[2, 1] * ky + B
    [2, 2] * kxy)) / (nLayers * thickness)
    Gxy = Sxy / yxy
    print(f"Gxy: {Gxy} N/m^2 \n")
```

Figure 22 – Code to calculate Engineering Properties

Question 2)

Laminate configuration	*[-45, +45, 0, 90, 90, 0, 0, 90, 90, 0, +45, -45]*
Thickness:	*0.2mm*
Young's Modulus in 1 direction	*E1 = 54 GPa*
Young's Modulus in 1 direction	*E2 = 18 GPa*
Poisson Ratio of the laminate	*v12 = 0.28*
Shear Modulus of the laminate	*G12 = 6 GPa*

Table 1 - Configuration information for Q2, Q3, Q4

A Matrix:

```
The A Matrix in N/m is:
 [[80802190.581 20336911.281 -0.000]
 [20336911.281 80802190.581 -0.000]
 [-0.000 -0.000 22316319.825]]
```

Figure 23 – Calculate A matrix – Short Form

```
The A Matrix in N/m is:
 [[ 8.08021906e+07  2.03369113e+07 -1.86264515e-09]
 [ 2.03369113e+07  8.08021906e+07 -1.86264515e-09]
 [-1.86264515e-09 -1.86264515e-09  2.23163198e+07]]
```

Figure 24 – Calculate A matrix –Long Form

D Matrix:

```
The D Matrix in Nm is:
 [[35.746 13.984 -1.479]
 [13.984 33.380 -1.479]
 [-1.479 -1.479 14.934]]
```

Figure 25 – Calculate D matrix – Short Form

```
The D Matrix in Nm is:
 [[35.74592771 13.98375465 -1.47864184]
 [13.98375465 33.38010077 -1.47864184]
 [-1.47864184 -1.47864184 14.93387076]]
```

Figure 26 – Calculate D matrix –Long Form

Question 3)

B Matrix:

```
The B Matrix in N is:
 [[-0.000 0.000 0.000]
 [0.000 0.000 0.000]
 [0.000 0.000 0.000]]
```

Figure 27 – Calculate B matrix – Short Form

```
The B Matrix in N is:
 [[-2.72848411e-12  2.72848411e-12  1.36424205e-12]
 [ 2.72848411e-12  4.54747351e-12  1.36424205e-12]
 [ 1.36424205e-12  1.36424205e-12  9.09494702e-13]]
```

Figure 28 – Calculate B matrix –Long Form

The B matrix represents the coupling stiffness matrix and accounts for the coupling between in-plane and out-of-plane deformations. In a symmetric laminate, the properties on both sides of its midplane are balanced, resulting in no coupling between these deformations. This symmetry simplifies the analysis.

However, the computed B matrix may not be exactly zero due to various reasons of which:

- *Computational Error:* In numerical calculations, the precision of the computer or software being used can introduce small errors. Although in theory the B-matrix should be zero for a perfectly symmetric laminate, these small numerical errors or rounding issues can lead to non-zero values, typically in the order of very small numerical values, which are practically negligible.
- *Approximations:* Engineering calculations often involve certain approximations and simplifications to enhance computational efficiency. While the B-matrix should ideally be zero for a perfectly symmetric laminate, these approximations may lead to small non-zero values in the calculations.

In practice, the deviation from a completely zero B matrix is typically attributed to these numerical and approximation factors and is considered acceptable as long as the values remain sufficiently small and do not significantly impact the overall analysis.

Question 4)

```
Engineering Properties:
Ex: [31534850716.929] N/m^2
Ey: [31534850716.929] N/m^2
Gxy: [9298466593.647] N/m^2
```

Figure 29 – Calculated Engineering Properties

Both Ex and Ey are equal at [31.534] GN/m². This indicates that the material exhibits isotropic behaviour in the x-y plane. The high values of Ex and Ey indicate that the material is relatively stiff and resistant to deformation in both the x and y directions. This stiffness makes it suitable for applications where rigidity is essential. The shear modulus Gxy, measuring the material's resistance to shear deformation in the xy-plane, is [9.298] GN/m². This suggests that this combination of high Young's moduli and a substantial shear modulus is designed to provide both structural strength and stability against shear stresses, making it suitable for applications where stiffness and shear resistance are critical.

Question 5)

Laminate configuration	*[90, 45,-45, 0]*
Thickness:	*0.125mm*

Table 2 - Configuration information for Q5

The resulting matrices are:

```
The A Matrix in N/m is:
 [[16009173.056 5061473.165 0]
 [5061473.165 16009173.056 0]
 [0 0 5473849.945]]

The B Matrix in N is:
 [[866.392 0.000 -144.399]
 [0.000 -866.392 -144.399]
 [-144.399 -144.399 0]]

The D Matrix in Nm is:
 [[0.372 0.067 0]
 [0.067 0.372 0]
 [0 0 0.075]]
```

Figure 30 – Calculate A, B, D matrix – Short Form

```
The A Matrix in N/m is:
 [[16009173.05585988  5061473.165389         0.        ]
 [ 5061473.165389   16009173.05585987        0.        ]
 [       0.                0.          5473849.9452355 ]]

The B Matrix in N is:
 [[ 8.66391703e+02  1.42108547e-14 -1.44398617e+02]
 [ 1.42108547e-14 -8.66391703e+02 -1.44398617e+02]
 [-1.44398617e+02 -1.44398617e+02  0.00000000e+00]]

The D Matrix in Nm is:
 [[0.37217834 0.06679345 0.        ]
 [0.06679345 0.37217834 0.        ]
 [0.         0.         0.07538464]]
```

Figure 31 – Calculate A, B, D matrix –Long Form

The Young’s Modulus on the X and Y direction are:

```
Engineering Properties:
Ex: [24393282966.481] N/m^2
Ey: [24393282966.481] N/m^2
Gxy: [10722531353.114] N/m^2
```

Figure 32 – Calculated Engineering Properties for Config 3

The change in stacking sequence in Question 5 compared to Question 2, while keeping the material properties constant, has led to differences in the laminate's performance. Despite being thinner (0.5mm) and having fewer layers, the laminate in Question 5 excels in Gxy (shear resistance), with a 115% improvement over the laminate in Question 2.

However, this improvement in Gxy comes at the expense of in-plane stiffness (Ex and Ey), which are only 77% of their corresponded values in Question 2. The results indicate that the stacking sequence in Question 5 prioritizes shear resistance over in-plane stiffness.

Additionally, the reduced thickness in Question 5 demonstrates that the laminate achieves similar Ex and Ey values with just 1/5th of the thickness of Question 2. This suggests that for applications where thickness is a concern, the laminate in Question 5 may be more efficient.

The similarity in the strain results for the x and y directions shows that the Ex and Ey values are the same. Analysing the strain matrices, it is evident that the resulting matrices are perpendicular to each other. This perpendicular is confirmed by the dot product with the [NM] matrix, which yields the same value for their corresponding rows.

Strain values for the x direction: $\begin{bmatrix} \varepsilon_x = 81.989783900 \\ \varepsilon_y = -26.61715021 \\ \gamma_{xy} = -5.75451535 \end{bmatrix}$ Strain values for the y: $\begin{bmatrix} \varepsilon_x = -26.61715021 \\ \varepsilon_y = 81.989783900 \\ \gamma_{xy} = 5.75451535 \end{bmatrix}$

The asymmetry of the laminate in Question 5 results in a non-zero [B] matrix, highlighting the presence of coupling effects between in-plane and out-of-plane deformations.