复合材料结构适航验证与审定

课程设计报告

学　　号：011810716

姓　　名：吴哲铭

指导老师：李龙彪

2021 年 12 月

目录

1. 推导单层材料任意 θ 加载的应力应变关系

平面应力状态下的应力应变关系为：

$$\begin{bmatrix}\varepsilon_1\\ \varepsilon_2\\ \varepsilon_3\end{bmatrix}=\begin{bmatrix}S_{11} & S_{12} & 0\\ S_{12} & S_{22} & 0\\ 0 & 0 & S_{66}\end{bmatrix}\begin{bmatrix}\sigma_1\\ \sigma_2\\ \sigma_3\end{bmatrix}. \qquad (1-1)$$

其中：$S_{11}=\frac{1}{E_1}$, $S_{22}=\frac{1}{E_2}$, $S_{66}=\frac{1}{G_{12}}$, $S_{12}=\frac{-\nu_{21}}{E_1}=\frac{-\nu_{12}}{E_2}$

式(1-1)可以简写为：

$$\{\varepsilon\}=[S]\{\sigma\}. \qquad (1-2)$$

根据刚度矩阵和柔度矩阵的互逆关系$[Q]=[S]^{-1}$：

$$\{\sigma\}=[Q]\{\varepsilon\}. \qquad (1-3)$$

同时有应力旋转公式和应变旋转公式：

$$\begin{bmatrix}\sigma_x\\ \sigma_y\\ \tau_{xy}\end{bmatrix}=[T]^{-1}\begin{bmatrix}\sigma_1\\ \sigma_2\\ \tau_{12}\end{bmatrix}. \qquad (1-4)$$

$$\begin{bmatrix}\varepsilon_x\\ \varepsilon_y\\ \gamma_{xy}\end{bmatrix}=[T]^{T}\begin{bmatrix}\varepsilon_1\\ \varepsilon_2\\ \gamma_{12}\end{bmatrix}. \qquad (1-5)$$

其中：

$$T=\begin{bmatrix}cos^2\theta & sin^2\theta & 2sin\theta cos\theta\\ sin^2\theta & cos^2\theta & -2sin\theta cos\theta\\ -sin\theta cos\theta & sin\theta cos\theta & cos^2\theta-sin^2\theta\end{bmatrix}. \qquad (1-6)$$

式(1-4)(1-5)可简写为：$\{\sigma\}_\theta=[T]^{-1}\{\sigma\}$, $\{\varepsilon\}_\theta=[T]^{T}\{\varepsilon\}$

由式(1-1)(1-2)(1-3)(1-4)(1-5)(1-6)得：

$$\{\sigma\}_\theta=[T]^{-1}\{\sigma\}=[T]^{-1}[Q]\{\varepsilon\}=[T]^{-1}[Q][T^T]^{-1}\{\varepsilon\}_\theta=[Q]_\theta\{\varepsilon\}_\theta. \qquad (1-7)$$

$$\{\varepsilon\}_\theta=[[T]^{-1}[Q][T^T]^{-1}]^{-1}\{\sigma\}_\theta=[T]^{T}[Q]^{-1}[T]\{\sigma\}_\theta=[S]_\theta\{\varepsilon\}_\theta. \qquad (1-8)$$

2. 分析 X 方向加载复合材料弹性模量随 θ 变化曲线

沿 X 方向加载时，复合材料在 X 方向的弹性模量E_x即为柔度矩阵$[S]$中的S_{11}项的倒数$1/S_{11}$。由于纤维对复合材料单层的拉压性能具有加强作用，因此复合材料单层沿纤维方向的弹性模量远大于其垂直纤维方向的弹性模量，X 方向与纤维方向夹角越大，复合材料单层的弹性模量越小。

仿真结果如图 2-1 所示，图中弹性模量E_x随θ的增大单调递减，$E_x(\theta = 0^o)$远大于$E_x(\theta = 90^o)$，与理论分析结果相匹配。

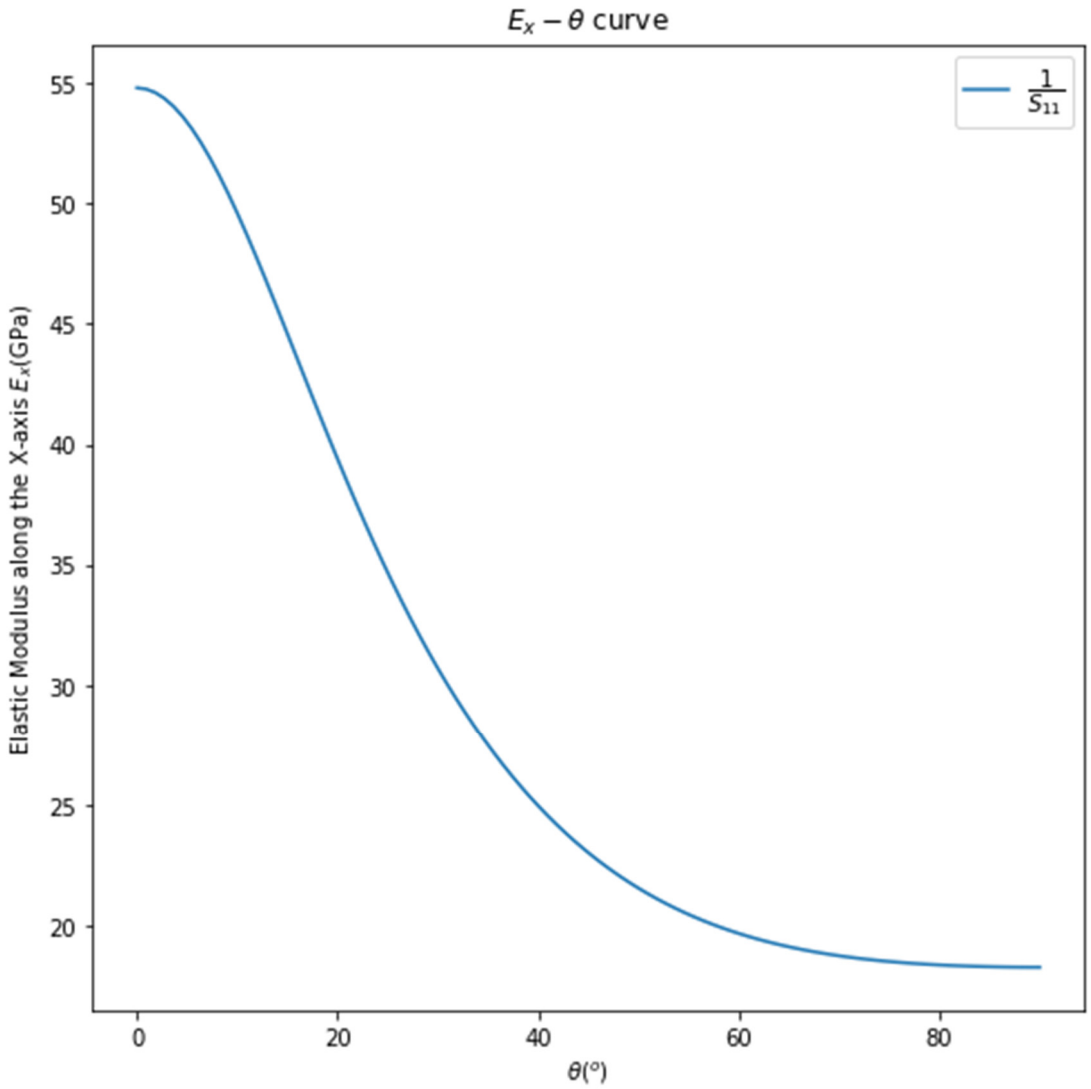

图 2-1 复合材料弹性模量随 θ 变化曲线

3. 给出 X 方向加载不同 θ 应力应变曲线

3.0 理论概述

仅 X 方向加载，即$\sigma_x = \sigma, \sigma_y = 0, \gamma_{xy} = 0$，根据式(1-8)有：

$$\begin{cases} \varepsilon_x = S_{\theta 11}\sigma_x \\ \varepsilon_y = S_{\theta 12}\sigma_x \\ \gamma_{xy} = S_{\theta 13}\sigma_x \end{cases} \quad (3-1)$$

基于式(3-1)可以画出 X 方向加载不同 θ 时的应力应变曲线。

3.1 不同$\boldsymbol{\theta}$下的$\boldsymbol{\varepsilon_x - \sigma_x}$曲线

如图 3-1 所示，随着θ的增大，$\varepsilon_x - \sigma_x$曲线的斜率也随之增大，即 X 方向上的单位载荷σ将在该方向造成更大的应变ε，这与偏轴加载时复合材料单层弹性模量随转角θ增大而增大的事实相匹配，以此证明图 3-1 的正确性。

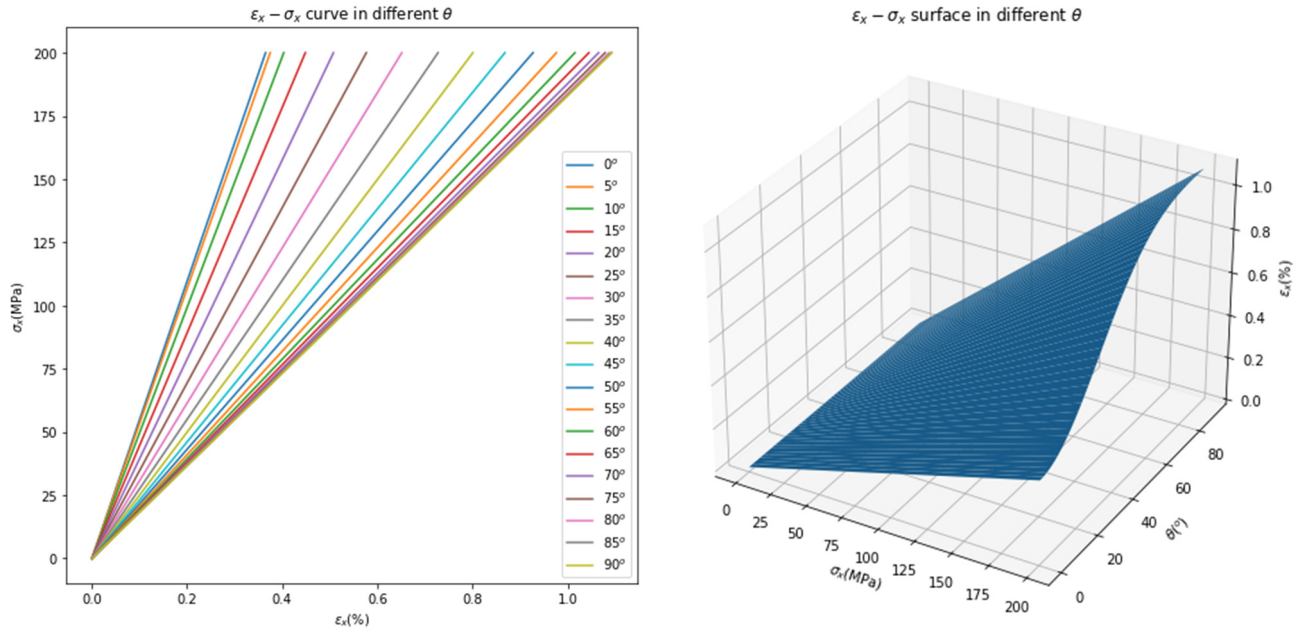

图 3-1 不同θ下的$\varepsilon_x - \sigma_x$曲线

3.2 不同$\boldsymbol{\theta}$下的$\boldsymbol{\varepsilon_y - \sigma_x}$曲线

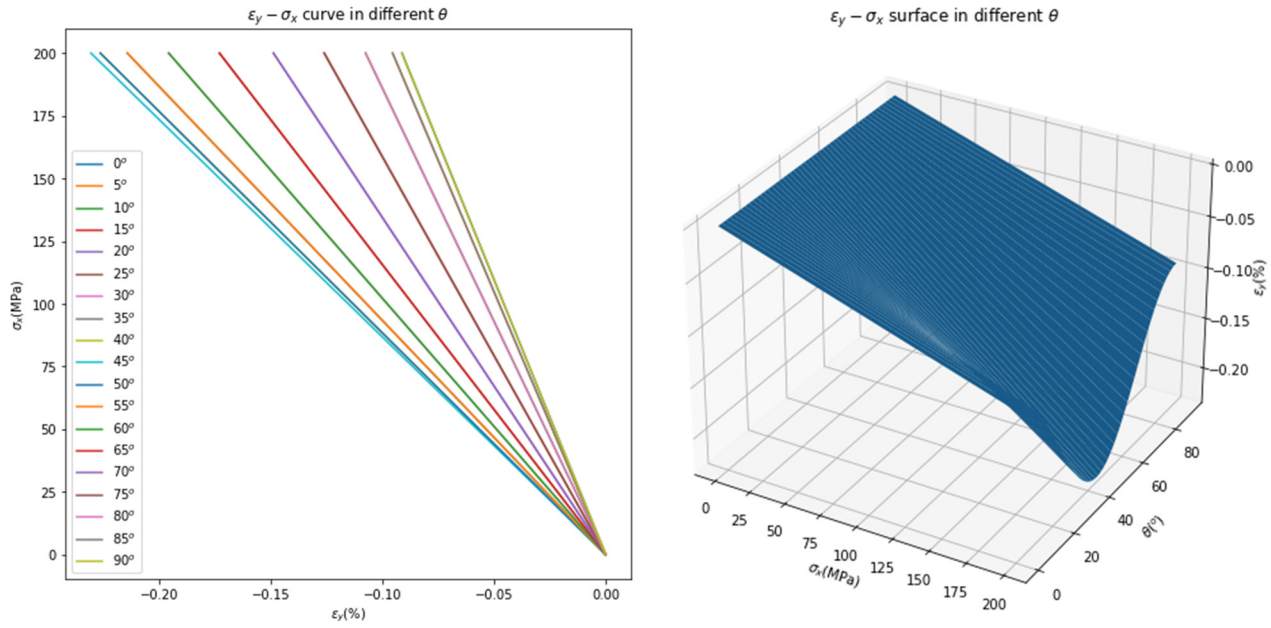

图 3-2 不同θ下的$\varepsilon_y - \sigma_x$曲线

3.3 不同$\boldsymbol{\theta}$下的$\boldsymbol{\gamma_{xy} - \sigma_x}$曲线

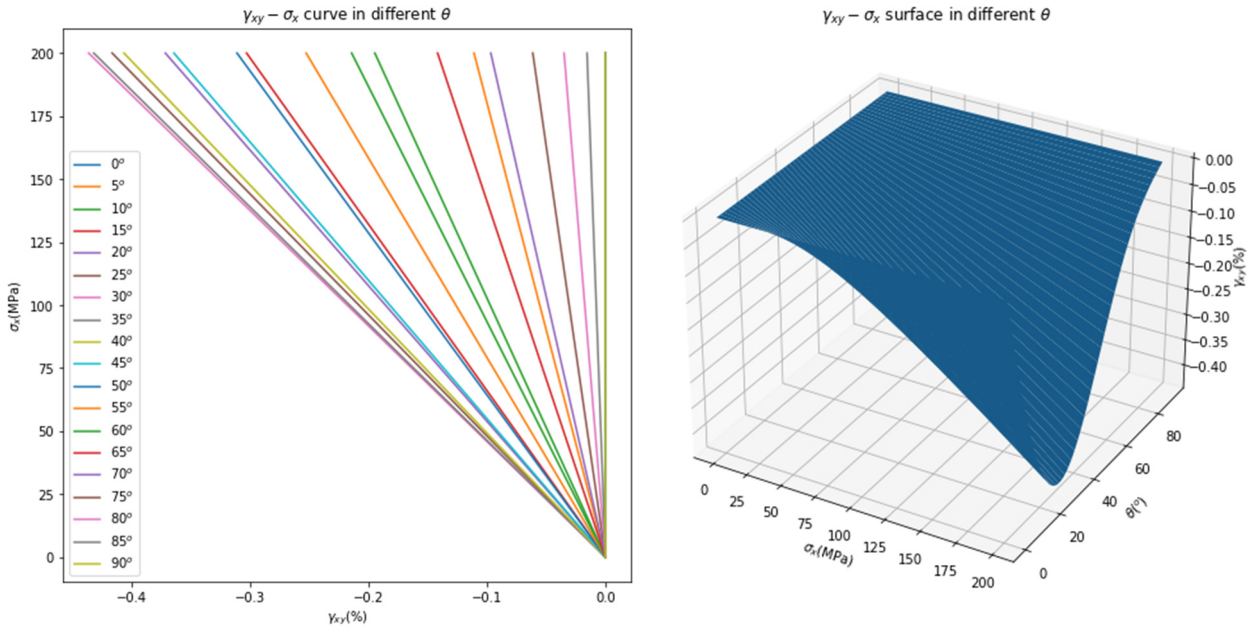

图 3-3 不同θ下的$\gamma_{xy} - \sigma_x$曲线

通过对比图 3-1、3-2 和 3-3，可以发现σ_x在 X 方向造成的变形远大于其在 Y 方向或剪切方向造成的变形，与实际情况相符合。

4. 给出 X 方向和 Y 方向同时加载不同 θ 应力应变曲线

4.0 理论概述

X 方向和 Y 方向同时加载，即$\sigma_x = \sigma_x, \sigma_y = \sigma_y, \gamma_{xy} = 0$，根据式(1-8)有：

$$\begin{cases} \varepsilon_x = S_{\theta 11}\sigma_x + S_{\theta 12}\sigma_y \\ \varepsilon_y = S_{\theta 12}\sigma_x + S_{\theta 22}\sigma_y \\ \gamma_{xy} = S_{\theta 13}\sigma_x + S_{\theta 23}\sigma_y \end{cases} \quad (4-1)$$

基于式(4-1)可以画出 X 方向加载不同 θ 时的应力应变曲线。

4.1 X 方向和 Y 方向同步加载

σ_x和σ_y同时由 0 上升到 200MPa，此时$\sigma - \varepsilon_x$曲线如图 4-1 所示：

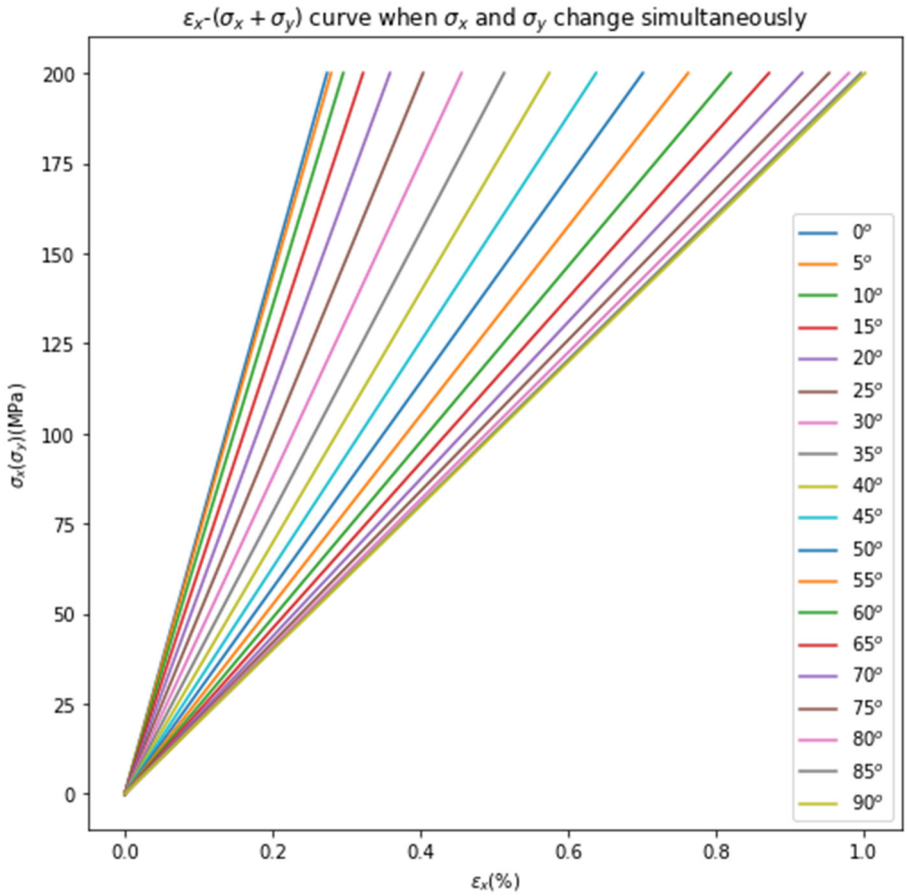

图 4-1 X 方向和 Y 方向同步加载时$\sigma - \varepsilon_x$曲线

4.2 特定角度($\boldsymbol{\theta = 45^o}$)X 方向和 Y 方向不同步加载

σ_x和σ_y各自由 0 上升到 200MPa，$\theta = 45^o$时$\sigma_x - \sigma_y - \varepsilon_x$曲面如图 4-2 所示：

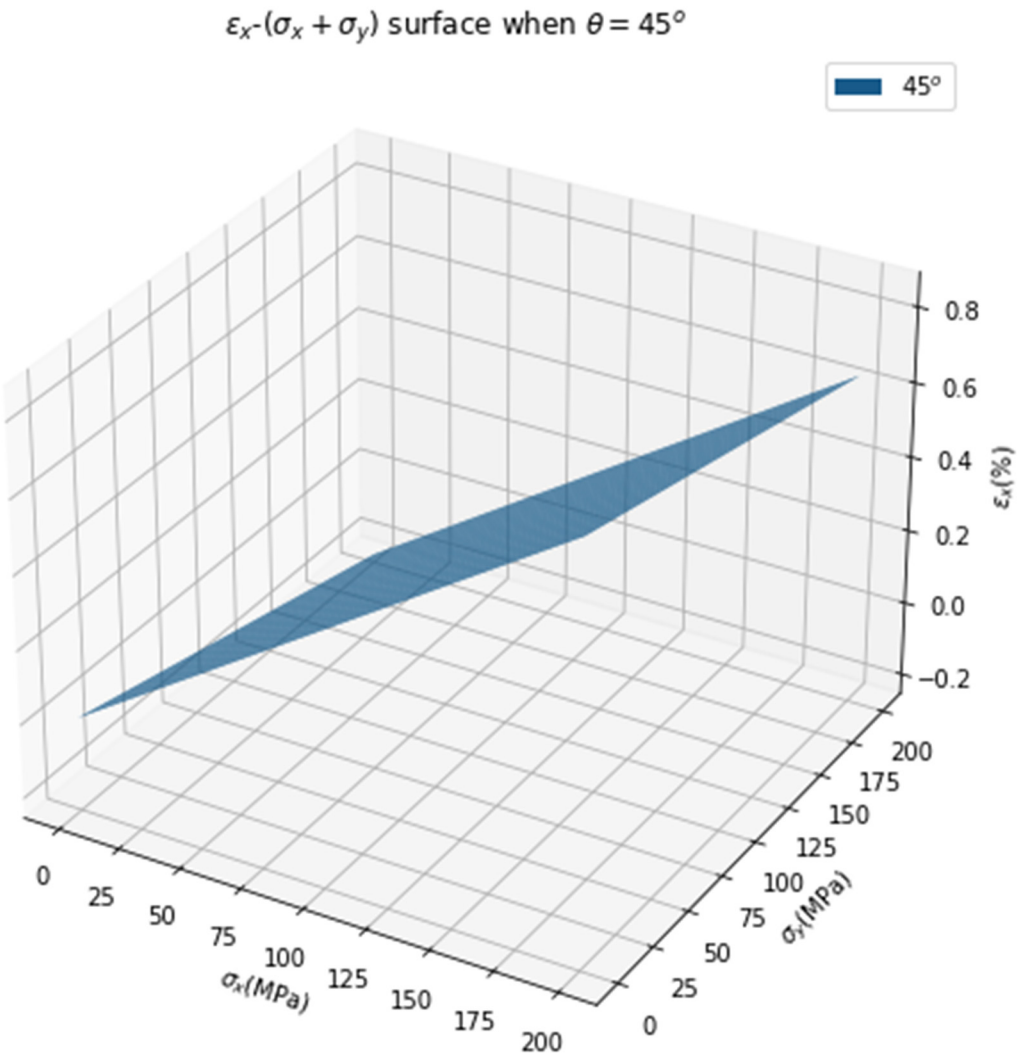

图 4-2 X 方向和 Y 方向不同步加载时$\sigma_x - \sigma_y - \varepsilon_x$曲面($\theta = 45^o$)

4.3 非特定角度 X 方向和 Y 方向不同步加载

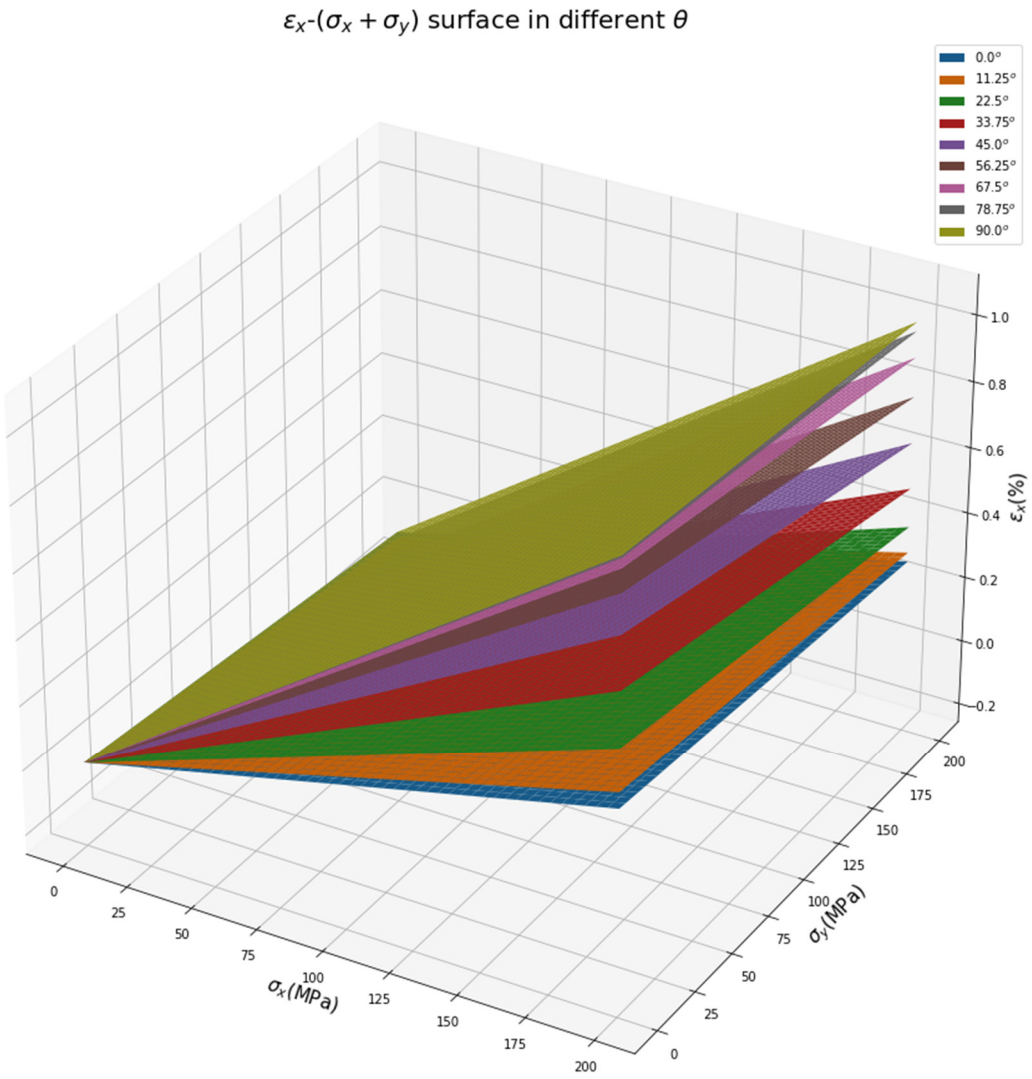

图 4-3 X 方向和 Y 方向不同步加载时$\sigma_x - \sigma_y - \varepsilon_x$曲面族($0^o \leq \theta \leq 90^o$)

5. 查阅相关资料分析偏轴加载下复合材料的损伤形式和损伤机理

5.0 概述

复合材料的破坏机理十分复杂，其强度受各种因素的影响，尤其对缺陷很

敏感。复合材料的强度不仅取决于两相材料的力学性质和体积分数，而且与纤维和基体的界面结合特性有关。各种工艺参数(如纤维表面处理状态、固化温度、压力大小、持续时间等)都会引起复合材料内部结构的变化(纤维分布不均匀、空穴或微裂纹、残余应力、界面层差异大等)，成为材料破坏的潜在因素，同时造成对复合材料强度预测的困难。

5.1 纵向拉伸

常用的纤维增强复合材料在拉断时表现为脆性破坏。当复合材料单向板受纵向拉伸时，在一些纤维的最弱处开始出现脆断，随着载荷的增加将继续有更多的纤维发生随机断裂。纤维断裂后，复合材料的承载能力降低，导致最终破坏。若基体是脆性的，且界面较强，纤维断裂处的裂纹将向基体扩展并穿过基体，引起临近纤维的断裂。裂纹扩展使得复合材料的有效承载面积减小，承受不了高的载荷，于是导致整个截面脆性断裂造成平断口，如图5-1(a)所示。如果界面较弱，纤维断裂处的界面将会发生脱黏，引起纤维从基体中拔出，最后使复合材料的横截面分离，如图5-1(b)所示；界面脱黏后的裂纹扩展也可能引起基体破坏，出现纵向开裂，如图 5-1(c)所示。实验表明，纤维体积分数φ_f低于某一定量φ_{fmin}($\varphi_f < \varphi_{fmin}$)时，复合材料表现为平断口的脆性破坏；中等纤维体积分数($\varphi_{fmin} < \varphi_f < \varphi_{fmax}$)的复合材料呈现出纤维拔出状的脆性破坏；纤维体积分数高($\varphi_f > \varphi_{fmax}$)的复合材料会发生基体开裂的脆性破坏。

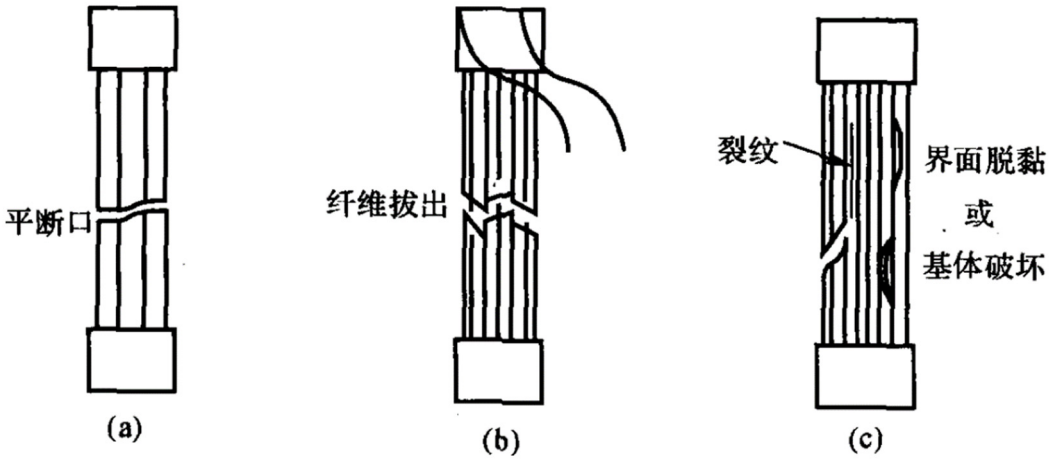

图 5-1 单向复合材料的拉伸破坏形式

5.2 纵向压缩

当复合材料单向板承受纵向压缩时，细长纤维容易发生微屈曲，见图 5-2(a)和(b)，纤维微屈曲可能出现两种特殊形式，反向屈曲和同向屈曲。当纤维间距相当大时，纤维反向屈曲(见图 5-2(a))就会发生，在基体中产生横向拉应变和压应变，因而这种屈曲形式又称为拉压型。纤维彼此同向屈曲时(见图 5-2(b))，基体以剪切应变为主，这种屈曲形式又称为剪切型，这也是最为常见的纤维微屈曲形式。实际上，复合材料中的纤维微屈曲是混合型的，拉压型和剪，切型同时存在，但总有某种形式占主导。复合材料单向板承受纵向压缩时也可能发生横向开裂破坏，如图 5-2(c)所示。在纵向压缩载荷下，泊松效应引起的横向伸长超过复合材料横向变形能力的极限值时，就会导致基体开裂或纤维脱黏，出现纵向破坏面。复合材料单向板承受纵向压缩载荷时也可能发生与加载线成45^o角的剪切破坏(见图 5-2(d))，在破坏之前纤维会出现弯折现象。

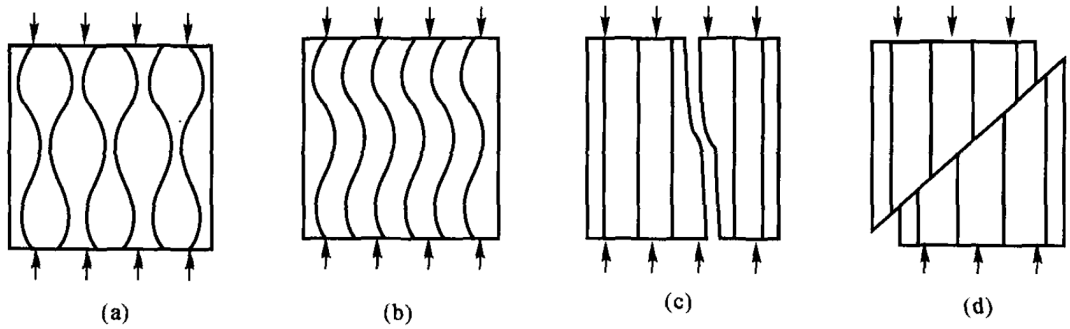

图 5-2 单向复合材料的压缩破坏形式

5.3 横向拉伸

复合材料单向板纵向强度主要是由纤维控制的，而横向强度和剪切强度则由基体或界面强度所控制，复合材料的破坏一般与纤维/基体间的界面状况密切相关。界面问题十分复杂，其力学性能表征的研究尚不充分。

复合材料单向板承受横向拉伸时，易出现基体或界面的拉伸破坏，如图 5-3 所示。因此，在横向拉伸载荷下复合材料的破坏模式就可以描述为：基体拉伸破坏、界面脱黏或纤维撕裂破坏。一般来说，破坏模式是联合作用的，复合材料破坏面的某些部分是由于基体拉伸破坏造成的，而另外部分是由于界面脱黏

或纤维撕裂引起的。在基体中，应力σ_{my}不是均匀分布的，而是在部分界面上达到最大值，即有应力集中区。当最大应力超过了基体拉伸强度σ_{mu}或界面拉伸强度σ_{iu}时，就会从界面处开始发生破坏。

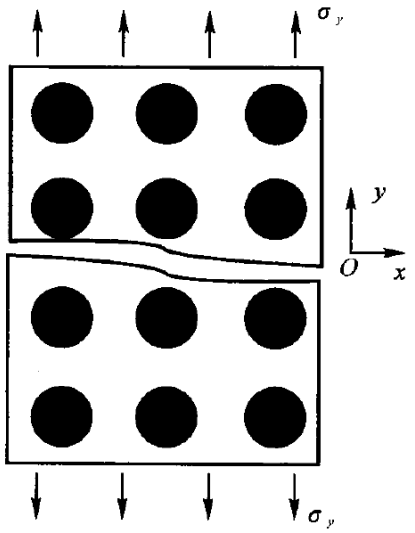

图 5-3 复合材料单向板承受横向拉伸破坏示意图

5.4 横向压缩

在横向压缩载荷作用下，复合材料单向板的破坏常常是由基体剪切破坏所致，如图 5-4 所示。大体上沿45^o斜面剪坏，有时还伴有界面破坏和纤维压碎。实验表明，横向压缩强度Y_c大约是横向拉伸强度Y_t的 3~7 倍。

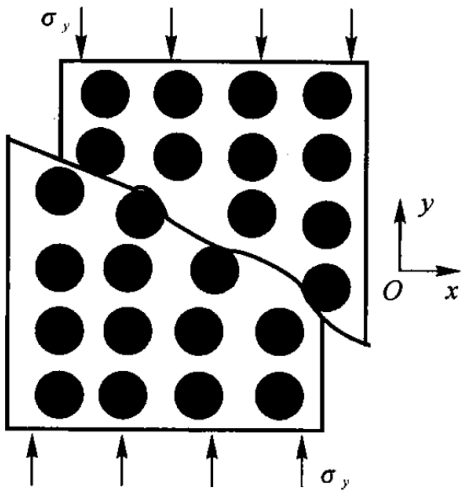

图 5-4 复合材料单向板承受横向压缩破坏示意图

5.5 剪切

在面内剪切载荷下，复合材料单向板的破坏是由基体剪切破坏、界面脱黏或者是两者联合作用所致，如图 5-5 所示。

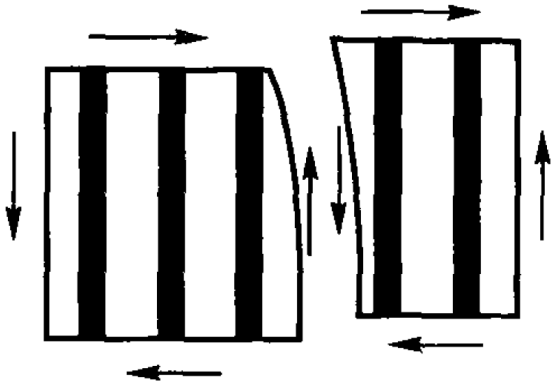

图 5-5 复合材料基体剪切破坏示意图

5.6 综合作用

复合材料的破坏机理十分复杂，不同复合材料在偏轴加载下的损伤形式和损伤机理存在较大差别，需要具体问题具体分析。

根据[3]中的试验及分析计算，对于 2D-SiC/SiC 复合材料，在45^o偏轴拉伸和压缩复合应力状态下材料损伤耦合力学行为的起始应力分别约为 40MPa 和-100MPa,复合应力状态下材料纤维束轴向方向上的拉伸损伤和面内剪切损伤进程间具有相互促进作用,面内剪切损伤对压缩损伤进程具有促进作用,但是压缩应力分量对面内剪切损伤进程具有明显的抑制作用;上述损伤耦合作用随着应力水平的增加而越发显著。由试件断口电镜扫描结果可知,复合应力状态下材料纤维束轴向方向上 3 个应力分量对材料内部0^o、90^o和45^o3 种取向基体裂纹开裂损伤进程的影响作用,是 2D-SiC/SiC 复合材料产生损伤耦合力学行为的主要细观损伤机制。

文献[4]以 T700-12K 纤维增强单向陶瓷基复合材料为例开展了偏轴加载试验。对于小角度偏轴加载（以2^o和3^o为例），其加载时的应力应变关系基本呈线性，且与沿纤维方向加载差异不大；而对于较大角度偏轴加载（以5^o和10^o为例），其在应变为 0.15%-0.20%区间内会突然失效，远没有达到纤维的强度水平。造成

这种现象的主要原因是单向陶瓷基复合材料在偏轴加载过程中，首先发生基体开裂，随着载荷的不断增大，基体裂纹不断萌生扩展，界面开始脱粘，并引发导致应力-应变曲线的切线模量逐渐减小。

6. 层板理论

6.0 层板理论简述

基于经典层合板理论，对于图 6-1 所示具有 n 的单层的层合板，其内力-变形关系如式(6-1)(6-2)所示：

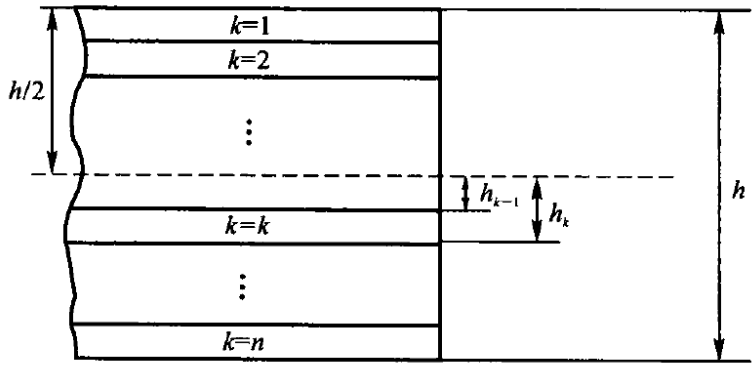

图 6-1 具有 n 个单层的层合板

$$\begin{bmatrix} N_x \\ N_y \\ N_{xy} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{16} \\ A_{12} & A_{22} & A_{26} \\ A_{16} & A_{26} & A_{66} \end{bmatrix} \begin{bmatrix} \varepsilon_x^0 \\ \varepsilon_y^0 \\ \gamma_{xy}^0 \end{bmatrix} + \begin{bmatrix} B_{11} & B_{12} & B_{16} \\ B_{12} & B_{22} & B_{26} \\ B_{16} & B_{26} & B_{66} \end{bmatrix} \begin{bmatrix} \kappa_x \\ \kappa_y \\ \kappa_{xy} \end{bmatrix}. \qquad (6-1)$$

$$\begin{bmatrix} M_x \\ M_y \\ M_{xy} \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} & B_{16} \\ B_{12} & B_{22} & B_{26} \\ B_{16} & B_{26} & B_{66} \end{bmatrix} \begin{bmatrix} \varepsilon_x^0 \\ \varepsilon_y^0 \\ \gamma_{xy}^0 \end{bmatrix} + \begin{bmatrix} D_{11} & D_{12} & D_{16} \\ D_{12} & D_{22} & D_{26} \\ D_{16} & D_{26} & D_{66} \end{bmatrix} \begin{bmatrix} \kappa_x \\ \kappa_y \\ \kappa_{xy} \end{bmatrix}. \qquad (6-2)$$

其中：

$$\begin{cases} A_{ij} = \sum_{k=1}^{n} \bar{Q}_{ij}^k (h_k - h_{k-1}) \\ B_{ij} = \frac{1}{2} \sum_{k=1}^{n} \bar{Q}_{ij}^k (h_k^2 - h_{k-1}^2) \ (i,j = 1,2,6) \\ D_{ij} = \frac{1}{3} \sum_{k=1}^{n} \bar{Q}_y^k (h_k^3 - h_{k-1}^3) \end{cases} \qquad (6-3)$$

将式(6-1)和(6-2)合成，得到一般层合板广义力和广义中面应变之间的关系，即：

$$\begin{bmatrix} N \\ M \end{bmatrix} = \begin{bmatrix} A & B \\ B & D \end{bmatrix} \begin{bmatrix} \varepsilon^0 \\ \kappa \end{bmatrix}. \tag{6-4}$$

对式(6-4)进行逆运算，得层合板的变形-内力关系：

$$\begin{bmatrix} \varepsilon^0 \\ \kappa \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} N \\ M \end{bmatrix}. \tag{6-5}$$

其中：

$$\begin{cases} a = A^{-1} - B^* D^{*-1} C^* \\ b = B^* D^{*-1} \\ c = -D^{*-1} C^* \\ d = D^{*-1} \end{cases} \tag{6-6}$$

$$\begin{cases} B^* = -A^{-1} B \\ C^* = B A^{-1} \\ D^* = D - B A^{-1} B \end{cases} \tag{6-7}$$

6.1 层合板性能分析

实例化除铺层顺序外完全相同的层合板$[\pm 45^o]_T$和$[0]_S$，均施加 Mx=20N 的力，绘制其 X 方向应变随θ变化的曲线如图 6-2 所示。

容易发现，$[\pm 45^o]_T$铺层的层合板各个方向性能更为均匀，而$[0]_S$铺层的层合板沿纤维方向性能优越，垂直纤维方向性能较差。

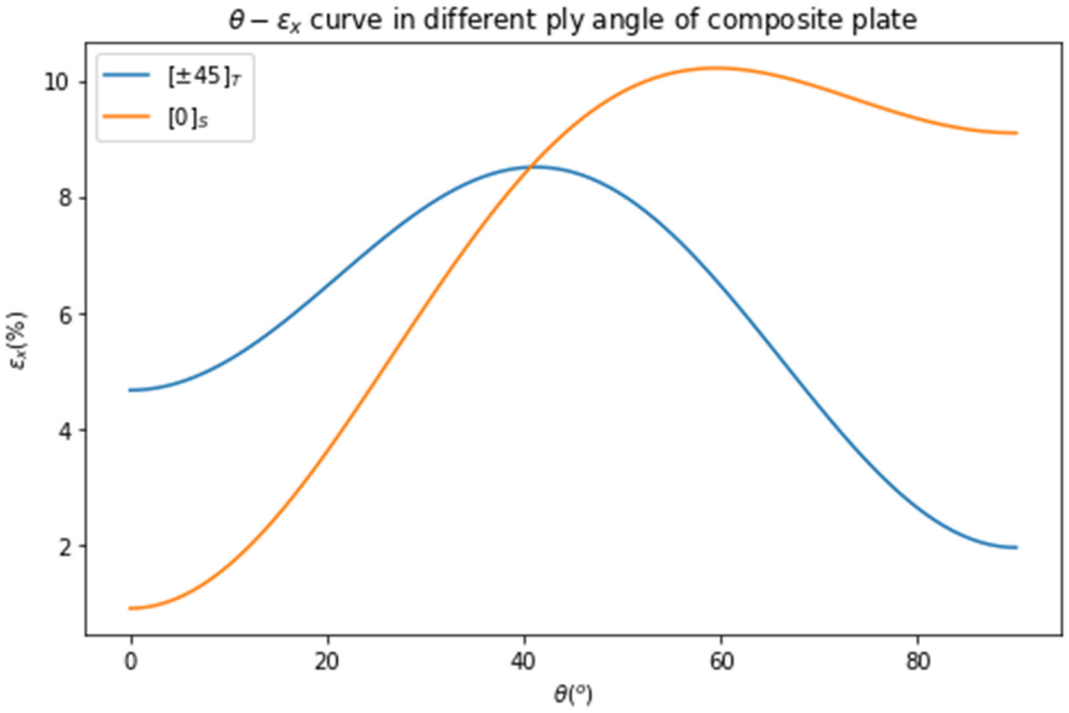

图 6-2 不同铺层顺序对层合板性能的影响

参考文献

[1] 李龙彪. 复合材料结构适航验证与审定. 北京: 北京航空航天大学出版社, 2019.

[2] 矫桂琼, 贾普荣. 复合材料力学. 西安: 西北工业大学出版社, 2008.

[3] 郭洪宝, 谢骏. 2D-SiC/SiC 复合材料损伤耦合力学行为[J]. 材料工程, 2019, 47(10): 160-165.

[4]牛序铭. 偏轴载荷下单向陶瓷基复合材料拉伸行为数值模拟及应用[D]. 南京航空航天大学, 2018.

附录

A.1 Code of composite.py

```
import numpy as np

class layer:
    def __init__(self,name='boron_epoxy'):
        """
        four kinds of material already in this package:
        'glass_epoxy', 'boron_epoxy',
        'graphite_epoxy' and 'aramid_epoxy'
        use layer('glass_epoxy') to create a layer made by glass_epoxy
        """
        if name=='glass_epoxy':
            EL,ET,vLT,GLT=54.8e9,18.3e9,0.25,9.1e9
            S11=1/EL
            S22=1/ET
            S12=-vLT/EL
            S66=1/GLT
            self.S=np.array([[S11,S12,0],[S12,S22,0],[0,0,S66]])
            self.Q=np.linalg.inv(self.S)
            self.S_=self.S
            self.Q_=self.Q
            pass
        elif name=='boron_epoxy':
            EL,ET,vLT,GLT=211e9,21.1e9,0.30,7.0e9
            S11=1/EL
            S22=1/ET
            S12=-vLT/EL
            S66=1/GLT
            self.S=np.array([[S11,S12,0],[S12,S22,0],[0,0,S66]])
            self.Q=np.linalg.inv(self.S)
            self.S_=self.S
            self.Q_=self.Q
            pass
        elif name == 'graphite_epoxy':
            EL,ET,vLT,GLT=211e9,5.3e9,0.25,2.6e9
            S11=1/EL
            S22=1/ET
```

```
            S12=-vLT/EL
            S66=1/GLT
            self.S=np.array([[S11,S12,0],[S12,S22,0],[0,0,S66]])
            self.Q=np.linalg.inv(self.S)
            self.S_=self.S
            self.Q_=self.Q
            pass
        elif name == 'aramid_epoxy':
            EL,ET,vLT,GLT=77e9,5.6e9,0.34,2.1e9
            S11=1/EL
            S22=1/ET
            S12=-vLT/EL
            S66=1/GLT
            self.S=np.array([[S11,S12,0],[S12,S22,0],[0,0,S66]])
            self.Q=np.linalg.inv(self.S)
            self.S_=self.S
            self.Q_=self.Q
            pass
        else:
            pass
        pass
        self.name=name
        self.angle=0  # default angle = 0
        self.t=0.125e-3  # default t = 0.125mm
        pass

    def set_by_E(self,EL,ET,vLT,GLT):
        """
        input: EL,ET,vLT,GLT  GPa
        """
        EL,ET,vLT,GLT=EL*1e9,ET*1e9,vLT,GLT*1e9
        S11=1/EL
        S22=1/ET
        S12=-vLT/EL
        S66=1/GLT
        self.S=np.array([[S11,S12,0],[S12,S22,0],[0,0,S66]])
        self.Q=np.linalg.inv(self.S)
        self.S_=self.S
        self.Q_=self.Q
        pass

    def set_by_Q(self,Q11,Q22,Q12,Q66):
        """
        input: Q11,Q22,Q12,Q66  GPa
```

```
        """
        Q11,Q22,Q12,Q66=Q11*1e9,Q22*1e9,Q12*1e9,Q66*1e9
        self.Q=np.array([[Q11,Q12,0],[Q12,Q22,0],[0,0,Q66]])
        self.S=np.linalg.inv(self.Q)
        self.Q_=self.Q
        self.S_=self.S
        pass

    def set_by_S(self,S11,S22,S12,S66):
        """
        input: Q11,Q22,Q12,Q66  GPa
        """
        S11,S22,S12,S66=S11*1e-9,S22*1e-9,S12*1e-9,S66*1e-9
        self.S=np.array([[S11,S12,0],[S12,S22,0],[0,0,S66]])
        self.Q=np.linalg.inv(self.S)
        self.S_=self.S
        self.Q_=self.Q
        pass

    def set_angle_degree(self,angle):
        self.angle=angle/180*np.pi
        self.__Q2Q_()
        self.__S2S_()
        pass

    def set_angle_rad(self,angle):
        self.angle=angle
        self.__Q2Q_()
        self.__S2S_()
        pass

    def __Q2Q_(self):
        angle=self.angle
        m,n=np.cos(angle),np.sin(angle)
        T=np.array([[m*m,n*n,2*m*n],[n*n,m*m,-2*m*n],[-m*n,m*n,m*m-
n*n]])
        T_inv=np.linalg.inv(T)
        T_inv_trans = np.transpose(T_inv)
        Q=self.Q
        Q_ = np.dot(np.dot(T_inv,Q),T_inv_trans)
        self.Q_= Q_
        pass

    def __S2S_(self):
```

```
        angle=self.angle
        m,n=np.cos(angle),np.sin(angle)
        T=np.array([[m*m,n*n,2*m*n],[n*n,m*m,-2*m*n],[-m*n,m*n,m*m-
n*n]])
        T_inv=np.linalg.inv(T)
        T_trans=np.transpose(T)
        S=self.S
        S_ = np.dot(np.dot(T_trans,S),T)
        self.S_=S_
        pass

    def set_t(self,t):
        self.t=t
        pass

    def set_name(self,name):
        self.name=name
        pass

    def get_status(self):
        print('name:\t'+self.name)
        print('t:\t',self.t)
        print('angle:\t',self.angle)
        print('Q:\t',self.Q)
        print('S:\t',self.S)
        print('Q_:\t',self.Q_)
        print('S_:\t',self.S_)
        pass

class compound:
    def __init__(self,layer_list,status='S'):
        if status=='S':
            layout=np.array(layer_list)
            reverse=layout[::-1]
            self.layout=np.append(layout,reverse)
            pass
        else:
            self.layout=np.array(layer_list)
            pass
        self.__get_t()
        self.__get_angle()
        self.h=np.sum(self.t)
        self.__ABDcal()
        self.__abdcal()
```

```
        pass

    def __get_t(self):
        t=np.array([])
        for item in self.layout:
            t=np.append(t,item.t)
            self.t=t
            pass
        pass

    def __get_angle(self):
        angle=np.array([])
        for item in self.layout:
            angle=np.append(angle,item.angle)
            self.angle=angle
            pass
        pass

    def __ABDcal(self):
        h_temp=0
        A,B,D=0,0,0
        for item in self.layout:
            if h_temp<self.h:
                bottom=h_temp-self.h/2
                head=bottom+item.t
                A=A+item.Q_*np.abs(head-bottom)
                B=B+item.Q_*(head**2-bottom**2)/2
                D=D+item.Q_*np.abs(head**3-bottom**3)/3
                h_temp=h_temp+item.t
                pass
            else:
                break
            pass
        self.A=A
        self.B=B
        self.D=D
        pass

    def __abdcal(self):
        A,B,D=self.A,self.B,self.D
        B_=-np.dot(np.linalg.inv(A),B)
        C_=np.dot(B,np.linalg.inv(A))
        D_=D-np.dot(np.dot(B,np.linalg.inv(A)),B)
        self.a=np.linalg.inv(A)-np.dot(np.dot(B_,np.linalg.inv(D_)),C_)
```

```
        self.b=np.dot(B_,np.linalg.inv(D_))
        self.d=np.linalg.inv(D_)
        pass

    def
strain_cal(self,Nx=0,Ny=0,Nxy=0,Mx=0,My=0,Mxy=0,outermost=False):
        f=np.array([[Nx,Ny,Nxy,Mx,My,Mxy]]).T
        a,b,d=self.a,self.b,self.d
        abd=np.r_[np.c_[a,b],np.c_[b,d]]
        eps_kappa_list=np.dot(abd,f)
        self.eps_kappa_list=eps_kappa_list
        self.eps0_x=eps_kappa_list[0]
        self.eps0_y=eps_kappa_list[1]
        self.eps0_xy=eps_kappa_list[2]
        self.kappa_x=eps_kappa_list[3]
        self.kappa_y=eps_kappa_list[4]
        self.kappa_xy=eps_kappa_list[5]
        if outermost==True:
            eps_x=self.eps0_x+self.h*self.kappa_x/2
            eps_y=self.eps0_y+self.h*self.kappa_y/2
            eps_xy=self.eps0_xy+self.h*self.kappa_xy/2
            return np.array([[eps_x,eps_y,eps_xy]]).T
        else:
            return eps_kappa_list
        pass
    pass
```

A.2 Code of answer.ipynb

A.2.0 头文件

```
import numpy as np
import matplotlib.pyplot as plt
import composite as cp
from mpl_toolkits.mplot3d import Axes3D
import warnings
warnings.filterwarnings('ignore')
```

A.2.1 X 方向加载复合材料弹性模量随θ变化曲线

```
layer1 = cp.layer(name = 'glass_epoxy')
Q_list = []
```

```
S_list = []
angle_list = np.linspace(0,180/2,100)
for i in angle_list:
    layer1.set_angle_degree(i)
    Q_list.append(layer1.Q_[0,0]/1e9)  # use Q11
    S_list.append(1/layer1.S_[0,0]/1e9)  # use 1/S11
    pass
# 下面是绘图部分
fig = plt.figure(figsize = (8,8))  # 设置画布大小

# plt.plot(angle_list,Q_list,label = '$Q_{11}$')  # use Q11
plt.plot(angle_list,S_list,label = '$\\dfrac{1}{S_{11}}$')  # use 1/S11
ax1 = plt.gca()
ax1.set_title('$E_x-\\theta$ curve')  # 设置标题
ax1.set_xlabel('$\\theta$($^o$)')  # 设置 x 轴
ax1.set_ylabel('Elastic Modulus along the X-axis $E_x$(GPa)')  # 设置 y
轴
plt.legend()  # 显示图例
```

A.2.2 X 方向加载不同θ应力应变曲线

A 2.2.1 二维

```
fig = plt.figure(figsize = (8,8))  # 设置画布大小

for angle in np.linspace(0,90,19):
    layer1 = cp.layer(name = 'glass_epoxy')
    layer1.set_angle_degree(angle)
    S_11,S_12,S_13 = layer1.S_[0,0],layer1.S_[1,0],layer1.S_[2,0]
    sigma_x = np.linspace(0,2e8,100)
    epsilon_x = S_11 * sigma_x

    # 下面是绘图部分
    plt.plot(epsilon_x*100, sigma_x/1e6, label =
str(int(angle))+'$^o$')


ax1 = plt.gca()
ax1.set_title('$\\varepsilon_x-\sigma_x$ curve')  # 设置标题
ax1.set_xlabel('$\\varepsilon_x$(%)')  # 设置 x 轴
ax1.set_ylabel('$\sigma_x$(MPa)')  # 设置 y 轴
plt.legend()  # 显示图例
```

A.2.2.2 三维

```
layer1 = cp.layer(name = 'glass_epoxy')
angle_list = np.linspace(0,90,99)
S_11_list = np.array([])
for angle in angle_list:
    layer1.set_angle_degree(angle)
    S_11_list = np.append(S_11_list, layer1.S_[0,0])


# 3d 坐标网格化部分
sigma_x = np.linspace(0,2e8,2)
_, angle_list1 = np.meshgrid(sigma_x, angle_list)
sigma_x1, S_11_list1 = np.meshgrid(sigma_x, S_11_list)
epsilon_x = S_11_list1 * sigma_x1


# 3d 绘图部分
fig = plt.figure(figsize = (8,8))
fig = plt.axes(projection = '3d')
fig.plot_surface(sigma_x1/1e6, angle_list1, epsilon_x*100)
fig.set_title('$\\varepsilon_x-\sigma_x$ curve in different $\\theta$')
fig.set_xlabel('$\sigma_x$(MPa)')
fig.set_ylabel('$\\theta$($^o$)')
fig.set_zlabel('$\\varepsilon_x$(%)')
```

A.2.3 X 方向和 Y 方向同时加载不同θ应力应变曲线

A.2.3.1 X 方向和 Y 方向同步加载

```
fig = plt.figure(figsize = (8,8))  # 设置画布大小

for angle in np.linspace(0,90,19):
    layer1 = cp.layer(name = 'glass_epoxy')
    layer1.set_angle_degree(angle)
    S_11,S_12,S_13 = layer1.S_[0,0],layer1.S_[1,0],layer1.S_[2,0]
    sigma_x, sigma_y = np.linspace(0,2e8,100), np.linspace(0,2e8,100)
    epsilon_x = S_11 * sigma_x + S_12 * sigma_y

    # 下面是绘图部分
    plt.plot(epsilon_x*100, sigma_x/1e6, label =
str(int(angle))+'$^o$')
```

```
ax1 = plt.gca()
ax1.set_title('$\\varepsilon_x$-($\sigma_x+\sigma_y$) curve when
$\sigma_x$ and $\sigma_y$ change simultaneously')  # 设置标题
ax1.set_xlabel('$\\varepsilon_x$(%)')  # 设置 x 轴
ax1.set_ylabel('$\sigma_x$($\sigma_y$)(MPa)')  # 设置 y 轴
plt.legend()  # 显示图例
```

A.2.3.2 特定角度($\boldsymbol{\theta = 45^o}$)X 方向和 Y 方向不同步加载

```
layer1 = cp.layer(name = 'glass_epoxy')
fig = plt.figure(figsize = (8,8))  # 设置画布大小
ax = plt.gca(projection = '3d')  # 绘制 3d 图

angle = 45
layer1 = cp.layer(name = 'glass_epoxy')
layer1.set_angle_degree(angle)
S_11,S_12 = layer1.S_[0,0],layer1.S_[1,0]
sigma_x = np.linspace(0,2e8,99)
sigma_y = np.linspace(0,2e9,99)

# 3d 绘图数据网格化
sigma_x, sigma_y = np.meshgrid(sigma_x, sigma_y)
epsilon_x = S_11 * sigma_x + S_12 * sigma_y
surf = ax.plot_surface(sigma_x/1e6, sigma_y/1e6, epsilon_x*100, label =
str(angle)+'$^o$')
surf._facecolors2d=surf._facecolor3d  # debug
surf._edgecolors2d=surf._edgecolor3d  # debug
# There is a bug in matplotlib.py, which can be solve on
"https://stackoverflow.com/questions/54994600/pyplot-legend-
poly3dcollection-object-has-no-attribute-edgecolors2d/54994985"

ax.set_title('$\\varepsilon_x-\sigma_x$ curve when $\\theta=45^o$')
ax.set_xlabel('$\sigma_x$(MPa)')
ax.set_ylabel('$\sigma_y$(MPa)')
ax.set_zlabel('$\\varepsilon_x$(%)')
ax.legend()
```

A.2.3.3 非特定角度 X 方向和 Y 方向不同步加载

```
layer1 = cp.layer(name = 'glass_epoxy')
fig = plt.figure(figsize = (16,16))  # 设置画布大小
```

```
ax = plt.gca(projection = '3d')  # 绘制 3d 图
font_title = {
    'size': 20
}
font_label = {
    'size': 15
}

for angle in np.linspace(0,90,9):
    layer1 = cp.layer(name = 'glass_epoxy')
    layer1.set_angle_degree(angle)
    S_11,S_12 = layer1.S_[0,0],layer1.S_[1,0]
    sigma_x = np.linspace(0,2e8,99)
    sigma_y = np.linspace(0,2e9,99)

    # 3d 绘图数据网格化
    sigma_x, sigma_y = np.meshgrid(sigma_x, sigma_y)
    epsilon_x = S_11 * sigma_x + S_12 * sigma_y
    surf = ax.plot_surface(sigma_x/1e6, sigma_y/1e6, epsilon_x*100,
label = str(angle)+'$^o$')
    surf._facecolors2d=surf._facecolor3d  # debug
    surf._edgecolors2d=surf._edgecolor3d  # debug
    # There is a bug in matplotlib.py, which can be solve on
"https://stackoverflow.com/questions/54994600/pyplot-legend-
poly3dcollection-object-has-no-attribute-edgecolors2d/54994985"

ax.set_title('$\\varepsilon_x-\sigma_x$ curve in different $\\theta$',
fontdict= font_title)
ax.set_xlabel('$\sigma_x$(MPa)', fontdict= font_label)
ax.set_ylabel('$\sigma_y$(MPa)', fontdict= font_label)
ax.set_zlabel('$\\varepsilon_x$(%)', fontdict= font_label)
ax.legend()
```

A.2.4 层板理论

```
fig = plt.figure(figsize = (8,5))  # 设置画布大小
Mx = 20
epsilon_x1, epsilon_x2 = np.array([]),np.array([])
for angle in np.linspace(0,90,99):
    layer1 = cp.layer(name='boron_epoxy')
    layer1.set_angle_degree(angle)
    layer2 = cp.layer(name='boron_epoxy')
    layer2.set_angle_degree(angle-90)
```

```
    comp1 = cp.compound([layer1,layer2],status='T')
    comp2 = cp.compound([layer1],status='S')

    epsilon_x1 = np.append(epsilon_x1,
comp1.strain_cal(Mx=Mx,outermost=True)[0,0,0])
    epsilon_x2 = np.append(epsilon_x2,
comp2.strain_cal(Mx=Mx,outermost=True)[0,0,0])

angle = np.linspace(0,90,99)

plt.plot(angle, epsilon_x1*100, label = '$[\pm 45]_T$')
plt.plot(angle, epsilon_x2*100, label = '$[0]_S$')

ax1 = plt.gca()
ax1.set_title('$\\theta-\\varepsilon_x$ curve in different ply angle of
composite plate')  # 设置标题
ax1.set_xlabel('$\\theta$($^o$)')  # 设置 x 轴
ax1.set_ylabel('$\\varepsilon_x$(%)')  # 设置 y 轴
plt.legend()  # 显示图例
```