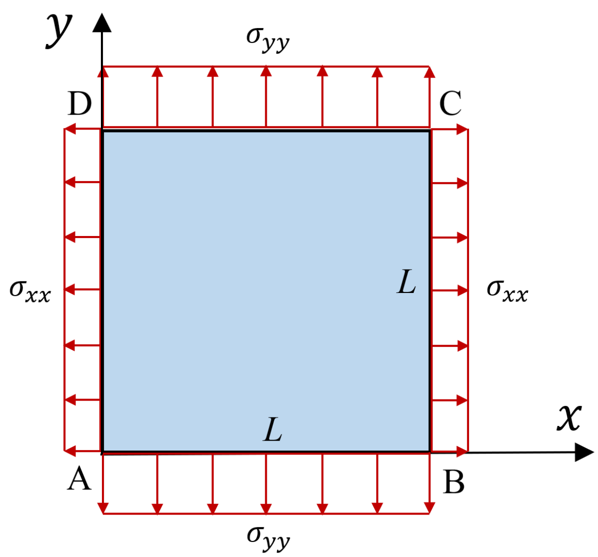
Homeworks for SM-I course

Homework 4: Understanding stresses, strains and constitutive relations

Consider a thin square wing spar made of aluminum alloy that can be treated as a homogeneous isotropic material with Young's modulus E and Poisson's ratio ν. It is subjected to uniform stress loading along all edges, as shown in the figure below. Because it is thin, and no force is applied in the z-direction, it can be treated a plane stress problem in x-y plane. The data are given as $E = 80GPa$, $\nu = 1/3$,$\sigma_{yy} = 150MPa$, and $L = 100mm$.



1. Derive the formulas for computing the strain components.
2. Determine the stress load σ_{xx} (in terms of σ_{yy}), under the condition that wing spar width remains unchanged. Compute the numerical value for given data.
3. Determine the strain and elongation in the y-direction, under the same condition given in 2). Compute the numerical value for given data.
4. Using the coordinate transformation rule, determine the elongation of the diagonal AC, under the same condition given in 2).
5. Determine the area change of the wing spar, under the same condition given in 2).

In [ ]:
```
# Place curse in this cell, and press Ctrl+Enter to import dependences.
import sys                              # for accessing the computer system
sys.path.append('../grbin/')   # Change to the directory in your system

from commonImports import *      # Import dependences from '../grbin/'
import grcodes as gr                 # Import the module of the author
#importlib.reload(gr)            # When grcodes is modified, reload it

from continuum_mechanics import vector
init_printing(use_unicode=True)       # For latex-like quality printing
np.set_printoptions(precision=4,suppress=True)  # Digits in print-outs
```

1.

No force in z means this can be treated as a plane-stress problem

$$\begin{bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \sigma_{xy}\end{bmatrix} = \begin{bmatrix}c_{xx} & c_{xy} & 0\\ c_{xy} & c_{yy} & 0\\ 0 & 0 & c_{zz}\end{bmatrix}\begin{bmatrix}\epsilon_{xx}\\ \epsilon_{yy}\\ 2\epsilon_{xy}\end{bmatrix}$$

$$\begin{bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \sigma_{xy}\end{bmatrix} = \frac{E}{(1-v^2)}\begin{bmatrix}1 & v & 0\\ v & 1 & 0\\ 0 & 0 & \frac{(1-v)}{2}\end{bmatrix}\begin{bmatrix}\epsilon_{xx}\\ \epsilon_{yy}\\ 2\epsilon_{xy}\end{bmatrix}$$

$$\begin{bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \sigma_{xy}\end{bmatrix} = \begin{bmatrix}\frac{E}{(1-v^2)} & \frac{Ev}{(1-v^2)} & 0\\ \frac{Ev}{(1-v^2)} & \frac{E}{(1-v^2)} & 0\\ 0 & 0 & \frac{E(\frac{1}{2}-\frac{v}{2})}{1-v^2}\end{bmatrix}\begin{bmatrix}\epsilon_{xx}\\ \epsilon_{yy}\\ 2\epsilon_{xy}\end{bmatrix}$$

$$\sigma_{xx} = \epsilon_{xx}\frac{E}{(1-v^2)} + \epsilon_{yy}\frac{Ev}{1-v^2}$$

$$\sigma_{yy} = \epsilon_{xx}\frac{Ev}{(1-v^2)} + \epsilon_{yy}\frac{E}{1-v^2}$$

$$\sigma_{xy} = 2\epsilon_{xy}\frac{E(\frac{1}{2}-\frac{v}{2})}{1-v^2}$$

Rearrainging and solving:

$$\epsilon_{xx} = \frac{1}{E}(\sigma_{xx} - v\sigma_{yy})$$

$$\epsilon_{yy} = \frac{1}{E}(\sigma_{yy} - v\sigma_{xx})$$

$$\epsilon_{xy} = \frac{\sigma_{xy}(1+v)}{E}$$

Additionally, strain in z is non-zero:

$$\epsilon_{zz} = -\frac{v}{E}(\sigma_{xx} + \sigma_{yy})$$

In [ ]:

```
sxx, syy, sxy, exx, eyy, exy, ezz = sp.symbols("sxx, syy, sxy, exx, eyy, exy, ezz")
E, v = sp.symbols("E, v")

C = (E/(1-(v**2)))*np.array([[1,v,0],[v,1,0],[0,0,((1-v)/2)]])

stress = C*np.array([[exx],[eyy],[2*exy]])

exx = (1/E)*(sxx-(v*syy))
eyy = (1/E)*(syy-(v*sxx))
exy = ((sxy*(1+v))/E)
ezz = -1*(v/E)*(sxx+syy)
```

2.

$$\epsilon_{xx} = 0 = \frac{1}{E}(\sigma_{xx} - v\sigma_{yy})$$
$$v\sigma_{yy} = \sigma_{xx}$$

In [ ]:

```
vnum = 1/3
Syy = 150 # MPa

Sxx = vnum*Syy

print("Stress in xx: %3.2f MPa" % (Sxx))
```

```
Stress in xx: 50.00 MPa
```

3.

In [ ]:
```
def transferM(theta, about = 'z'):
    '''Create a transformation matrix for coordinate transformation (numpy)\
    Input theta: rotation angle in degree \
          about: the axis of the rotation is about \
    Return: numpy array of transformation matrix of shape (3,3)'''
    from scipy.stats import ortho_group

    n = 3          # 3-dimensonal problem
    c, s = np.cos(np.deg2rad(theta)), np.sin(np.deg2rad(theta))
    #T = np.zeros((n,n))

    if about == 'z':
        # rotates about z by theta
        T = np.array([[ c, s, 0.],
                      [-s, c, 0.],
                      [0.,0., 1.]])
    elif about == 'y':
        # rotates about y by theta
        T = np.array([[ c, 0.,-s],
                      [0., 1.,0.],
                      [s, 0., c]])
    elif about == 'x':
        # rotates about x by theta
        T = np.array([[ 1.,0., 0.],
                      [ 0., c, s],
                      [ 0.,-s, c]])
    else: # randomly generated unitary matrix->transformation matrix, no theta
        T = ortho_group.rvs(dim=n)          # Generate a random matrix
        T[2,:] = np.cross(T[0,:], T[1,:])   # Enforce the righ-hand rule

    return T, about
```

In [ ]:
```
Enum = 80e3 # MPa

strainmatrix = np.array([[float(exx.subs({E:Enum,v:vnum,sxx:Sxx,syy:Syy})),0,0],[0,

strainmatrix
```

Out[ ]:
```
array([[ 0.    ,  0.    ,  0.    ],
       [ 0.    ,  0.0017,  0.    ],
       [ 0.    ,  0.    , -0.0008]])
```

In [ ]:
```
L = 100*(10**-3)

theta3 = 90

T3, about = transferM(theta3)

strain3 = T3@strainmatrix@T3.T

print("Strain: %3.6f" % (strain3[0,0]))
print("Elongation: %3.8f mm" % (L*strain3[0,0]*1000))
```

```
Strain: 0.001667
Elongation: 0.16666667 mm
```

4.

In [ ]:
```
theta4 = 45

T4, about = transferM(theta4)

strain4 = T4@strainmatrix@T4.T

print("Strain: %3.6f" % (strain4[0,0]))
print("Elongation: %3.8f mm" % (np.sqrt(2)*L*strain4[0,0]*1000))
```

```
Strain: 0.000833
Elongation: 0.11785113 mm
```

5.

In [ ]:
```
areachange = ((L*1000)*((L+(L*strain3[0,0]))*1000))-((L*1000)*(L*1000))

print("The change in area is %3.2f mm^2" % (areachange))
```

```
The change in area is 16.67 mm^2
```