Contents

List of Tables

List of Figures

1 Problem Statement

The goal of this paper is to analyze the system found in Figure 1 by analyzing how the principal stresses at a point $K(x, y)$ vary in both magnitude and direction.

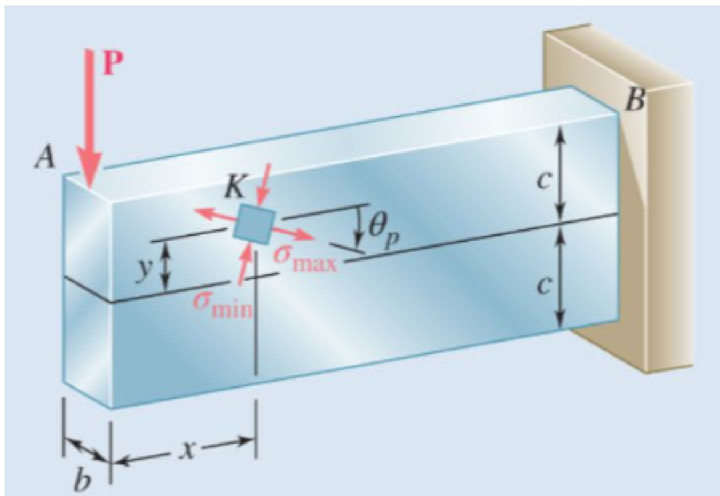


Figure 1: Problem Statement

The problem statement does not give any values for b, c, or P. Instead, it asks for σ_{max}/σ_m and σ_{min}/σ_m, along with θ_p, which are quantities that can be found independent of b, c, and P.

2 Theory Manual

When transforming stresses, we make use of the following equations:

$$\sigma_x' = \sigma_x cos^2(\theta) + \sigma_y sin^2(\theta) + 2\tau_{xy} sin(\theta) cos(\theta) \tag{1}$$

$$\sigma_y' = \sigma_x sin^2(\theta) + \sigma_y cos^2(\theta) - 2\tau_{xy} sin(\theta) cos(\theta) \tag{2}$$

$$\tau_{xy}' = (\sigma_y - \sigma_x) sin(\theta) cos(\theta) + \tau_{xy}(cos^2(\theta) - sin^2(\theta)) \tag{3}$$

Defining a value θ_p such that $\tau_{xy}'(\theta = \theta_p) = 0$, we get the principal angle from the following equation.

$$tan\,(2\theta_p) = \frac{2\tau_{xy}}{\sigma_x - \sigma_y} \tag{4}$$

Furthermore, substituting back in for θ_p gives an equation for σ_{min} and σ_{max}, which are the transformed stresses at the principal angle.

$$\sigma_{max}, \sigma_{min} = \frac{\sigma_x + \sigma_y}{2} \pm \sqrt{\left(\frac{\sigma_x - \sigma_y}{2}\right)^2 + \tau_{xy}^2} \tag{5}$$

Next, we must find the values of the various stresses. As there is no axial vertical stress, the value of σ_y is zero.

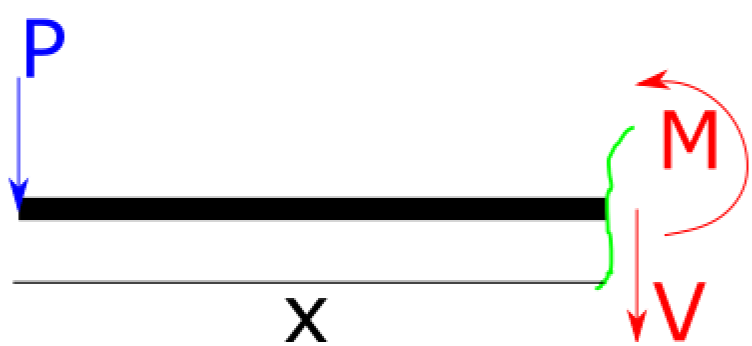


Figure 2: Shear and Moment FBD

We find from Figure 2 that the internal shear V has to be $-P$. This give the shear stress as follows:

$$\tau_{xy} = \frac{VQ}{It} = \frac{-3P\left[1 - \left(\frac{y}{c}\right)^2\right]}{4bc} \tag{6}$$

Likewise, from Figure 2 we find the internal moment to be $-Px$, giving the axial stress as follows:

$$\sigma_x = \frac{M}{I}y = \frac{-3P}{2bc}\left(\frac{x}{c}\right)\left(\frac{y}{c}\right) \tag{7}$$

The maximum value of stress, σ_m, is trivially $\frac{-3P}{2bc}\left(\frac{x}{c}\right)$. Combining this with Equations (5), (6), and (7), we get the following equation:

$$\frac{\sigma_{max}}{\sigma_m}, \frac{\sigma_{min}}{\sigma_m} = \frac{\left(\frac{x}{c}\right)\left(\frac{y}{x}\right) \pm \sqrt{\left[\left(\frac{x}{c}\right)\left(\frac{y}{c}\right)\right]^2 + \left[1 - \left(\frac{y}{c}\right)^2\right]^2}}{2\left(\frac{x}{c}\right)} \tag{8}$$

Furthermore, combining Equations (4), (6), and (7), we get the following equation for θ_p:

$$\theta_p = \frac{1}{2}arctan\left(\frac{1 - \left(\frac{y}{c}\right)^2}{\left(\frac{x}{c}\right)\left(\frac{y}{c}\right)}\right) \tag{9}$$

3 Programmer Manual

The variables used and their descriptions are below in Table 1.

Variable	Description
x	A 1D numpy array containing the values of x/c
y	A 1D numpy array containing the values of y/c
X	A 2D numpy array extending x to 2 dimensions
Y	A 2D numpy array extending y to 2 dimensions
SIG_min	A 2D numpy array containing the values of σ_{min}/σ_m
SIG_max	A 2D numpy array containing the values of σ_{max}/σ_m
index	The index of x used to plot SIG_min and SIG_max
y_index	The y index containing the maximum or minimum stress

Table 1: Table of Variables

The functions used and their descriptions are below in Table 2.

Function	Description
principal_angle	Returns θ_p in radians, given x/c and y/c
principal_angle_deg	Same as principal_angle, but returns in degrees
principal_stresses	Returns σ_{max}/σ_m and σ_{min}/σ_m given x/c and y/c

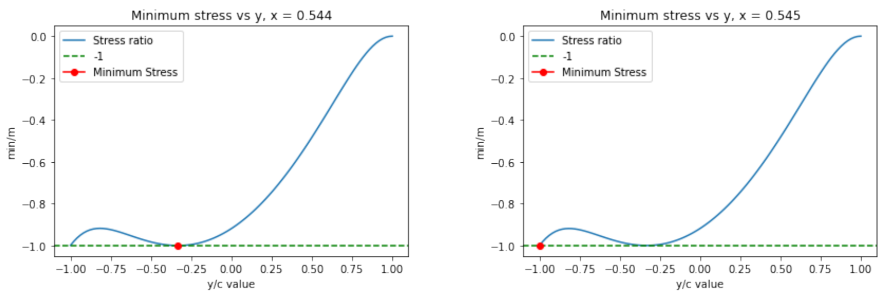
Table 2: Table of Functions

Steps taken in the code:

1. Import the numpy and matplotlib libraries to allow for easier analysis.
2. Define the functions in Table 2.
3. Create the numpy arrays to contain x/c and y/c.
4. Calculate σ_{min}/σ_m and σ_{max}/σ_m for every point defined in the previous step.
5. Define the x index that the calculations are being done on.
6. Find the minimum value of σ_{min}/σ_m and plot the whole curve.
7. Find the maximum value of σ_{max}/σ_m and plot the whole curve.
8. Plot the values of θ_p across the beam.
9. Compare the values of σ_{min}/σ_m and σ_{max}/σ_m in the problem statement to those obtained from the code.

4 Results and Analysis

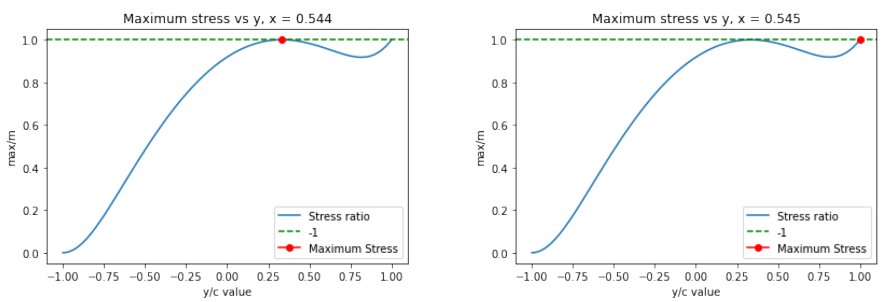
From the code, we get the following plots for σ_{min}/σ_m (Figure 3), σ_{max}/σ_m (Figure 4), and θ_p (Figure 5).



(a) x = .544c

(b) x = .545c

Figure 3: σ_{min}/σ_m at various x values



(a) x = .544c

(b) x = .545c

Figure 4: σ_{max}/σ_m at various x values

From the above figures, we can see that for $x \leq .544c$ we get $\sigma_{min}/\sigma_m < -1$ and $\sigma_{max}/\sigma_m > 1$.

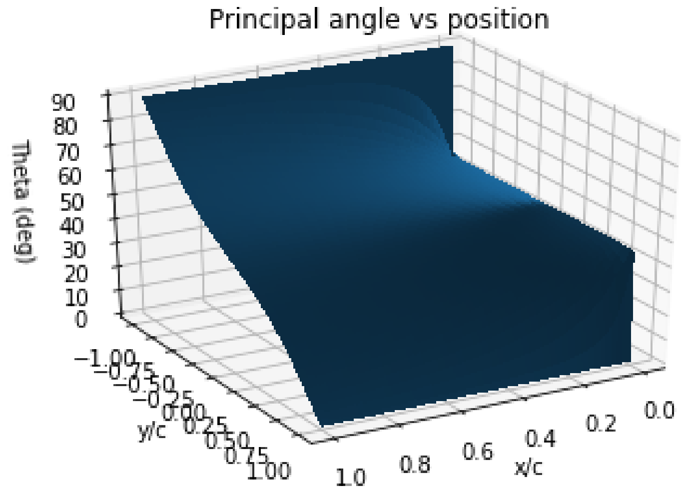


Figure 5: Principal angle at various coordinates

Figure 5 shows that the value of θ_p is almost constant as x/c approaches zero, and becomes approximately linear as x/c increases.

y/c	σ_{min}/σ_m	σ_{max}/σ_m
1.0	0.0	1.0
0.8	-0.01	0.81
0.6	-0.04	0.64
0.4	-0.09	0.49
0.2	-0.16	0.36
-0.0	-0.25	0.25
-0.2	-0.36	0.16
-0.4	-0.49	0.09
-0.6	-0.64	0.04
-0.8	-0.81	0.01
-1.0	-1.0	0.0

Table 3: Values for $x = 2c$

y/c	σ_{min}/σ_m	σ_{max}/σ_m
1.0	0.0	1.0
0.8	-0.001	0.801
0.6	-0.003	0.603
0.4	-0.007	0.407
0.2	-0.017	0.217
-0.0	-0.063	0.062
-0.2	-0.217	0.017
-0.4	-0.407	0.007
-0.6	-0.603	0.003
-0.8	-0.801	0.001
-1.0	-1.0	0.0

Table 4: Values for $x = 8c$

Tables 3 and 4 are both in line with the values given in the problem statement. As a result, we find that the code is accurate, meaning that the results are correct.

5 Appendix

```
import numpy as np

import matplotlib.pyplot as plt


def principal_stresses(xc, yc):

    left = yc

    right = np.sqrt((xc * yc) ** 2 + (1 - yc**2) **2 )/xc

    min = .5 * (left - right)

    max = .5 * (left + right)

    return((min, max))


def principal_angle(xc, yc):

    if(yc == 0):

        return(np.pi / 4)

    else:

        theta = 1/2 * np.arctan((1 - yc ** 2)/(xc * yc))

        if(theta < 0):

            theta += np.pi/2

        if(yc == -1):

            theta += np.pi/2

        return(theta)


def principal_angle_deg(xc, yc):

    return 180/np.pi * principal_angle(xc, yc)


x = np.linspace(0, 1, 1000)[1:]

y = np.linspace(-1, 1, 100)
```

```
X, Y = np.meshgrid(x, y)
SIG_min, SIG_max = principal_stresses(X, Y)

index = 543

# plot curve
plt.plot(y, SIG_min[:, index])
plt.axhline(-1, color='green', linestyle='dashed')

# plot minimum
y_index = np.argmin(SIG_min[:, index])
plt.plot(y[y_index], SIG_min[y_index, index], marker = 'o', color = 'red')

# add labels
plt.xlabel('y/c value')
plt.ylabel('min/m')
plt.title(f'Minimum stress vs y, x = {round(x[index], 3)}')
plt.legend(('Stress ratio', '-1', 'Minimum Stress'))

# plot curve
plt.plot(y, SIG_max[:, index])
plt.axhline(1, color='green', linestyle='dashed')

# plot minimum
y_index = np.argmax(SIG_max[:, index])
plt.plot(y[y_index], SIG_max[y_index, index], marker = 'o', color = 'red')
```

```
# add labels
plt.xlabel('y/c value')
plt.ylabel('max/m')
plt.title(f'Maximum stress vs y, x = {round(x[index], 3)}')
plt.legend(('Stress ratio', '-1', 'Maximum Stress'))

theta = np.empty_like(X)

for i in range(X.shape[0]):
    for j in range(X.shape[1]):
        theta[i, j] = principal_angle_deg(x[j],y[i])

fig = plt.figure()
ax = fig.add_subplot(projection='3d')

surf = ax.plot_surface(X, Y, theta,
                       linewidth=0, antialiased=False)
ax.set_xlabel('x/c')
ax.set_ylabel('y/c')
ax.set_zlabel('Theta (deg)')
ax.set_title('Principal angle vs position')

ax.view_init(30, 60)

y = np.arange(-1, 1.2, .2)[::-1]
x1 = 2
```

```
x2 = 8

# define lists
min_2c = np.empty_like(y)
max_2c = np.empty_like(y)
min_8c = np.empty_like(y)
max_8c = np.empty_like(y)

# generate values
min_2c, max_2c = principal_stresses(x1, y)
min_8c, max_8c = principal_stresses(x2, y)
```