CAD project (Group 8)
Mohr’s Circle GUI

Abdulrahman Mohamed Abou El Fotouh	Mechatronics (Sec 1)
Ali Khalid Ali Ibrahim	Mechatronics (Sec 1)
Mohammed Mohsen Abdulrahman Mohammed	Mechatronics (Sec 2)
Mahmoud Morsy Abdullah Abdulrahman	Mechatronics (Sec 2)
Sameh Reda Mahrous Al-Sayed	Production

Supervision

Dr. Mohamed Shehata

Contents:

Table of symbol:

σ …… Normal stress

τ …… Shear stress

θ …… Angle of plan from vertical axis

σ_{avg} …… Normal stress

σ_{p_1} …… Maximum Normal Stress

σ_{p_2} …… Minimum Normal Stress

τ_{max} …… Maximum Shear Stress

R…… Radius of Mohr's circle

C…… center of Mohr's circle

Introduction:

The transformation equations for plane stress can be represented in graphical form by a plot known as Mohr's Circle.

What is Mohr's circle?

The Mohr circle is used to find the stress components and, i.e., coordinates of any point on the circle, acting on any other plane passing through making an angle with the plane

Why Mohr's circle?

because it enables you to visualize the relationships between the normal and shear stresses acting on various inclined planes at a point in a stressed body.

- principal stresses
- maximum shear stresses.
- stresses on inclined planes.

Applications of Mohr's circle:

Mohr's circle is often used in calculations relating to mechanical engineering for materials' strength, geotechnical engineering for the strength of soils, and structural engineering for strength of built structures. It is also used for calculating stresses in many planes by reducing them to vertical and horizontal components. These are called principal planes in which principal stresses are calculated; Mohr's circle can also be used to find the principal planes and the principal stresses in a graphical representation

Project objective:

Visual tool used to determine the stresses that exist at a given point about the angle of orientation of the stress element.

It is important in many mechanical engineering sciences like metallurgy and material science and metal forming and design

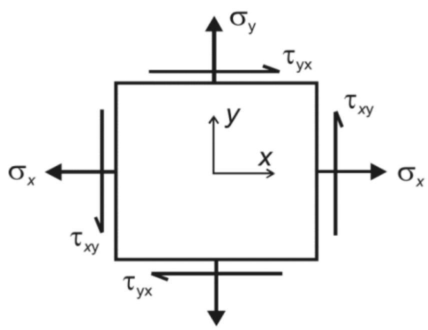
Eliminate engineering student efforts and help them to solve Mohr's circle problem

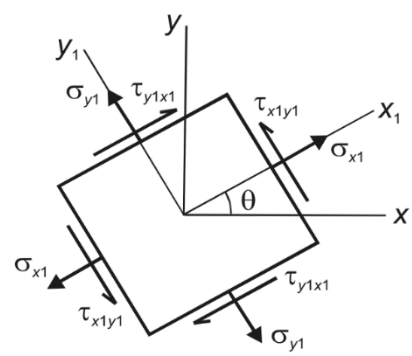
Stress Transformation:

Analytical Solutions:

$$\sigma_{x_1} - \frac{\sigma_x+\sigma_y}{2} = \frac{\sigma_x-\sigma_y}{2} cos\ 2\theta + \tau_{xy}\ sin\ 2\theta$$

$$\tau_{x_1y_1} = -\frac{\sigma_x+\sigma_y}{2} sin\ 2\theta + \tau_{xy}\ cos\ 2\theta$$





Mohr's circle can be used, along with geometry and trigonometry, to derive equations to solve for angles and values of stresses at selected points. Either the double angle method or the pole method can be used for the analytical solutions.

It is important to note that general equations for solving stress transformation problems provided in reference books are based on knowing the state of stress on two planes that are perpendicular to each other. If the problem you are trying to solve does not meet this criterion, then none of the equations can be used.

However, the necessary equations can still be derived using geometry, trigonometry, and either the double angle method or the pole method. Either the double angle method or the pole method can be used to solve any problem.

The choice of which method to use should be based on which method can achieve the desired results with the least amount of effort for that problem.

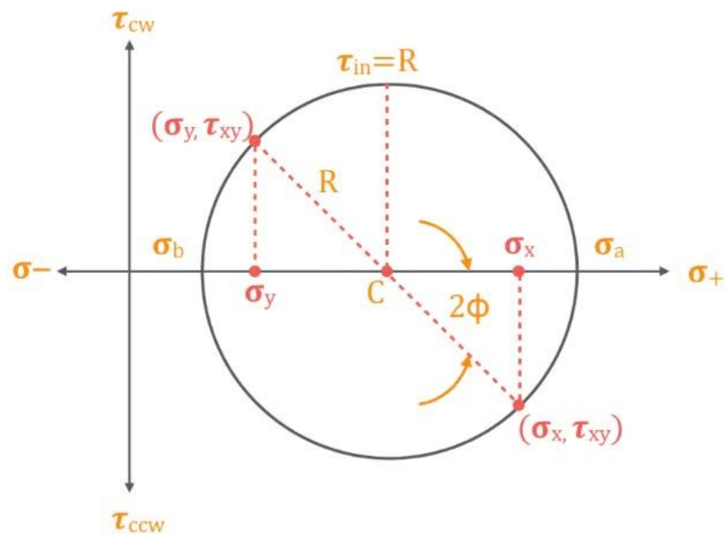
Graphical Solution:

C= $\sigma_{avg} = \frac{\sigma_x+\sigma_y}{2}$

R= $\sqrt{\left(\frac{\sigma_x-\sigma_y}{2}\right)^2+\tau_{xy}^2}$

$\sigma_{p1,p2} = c \pm R$

$\tau_{max} = R$



Variations of Normal stress and Shear stress value concerning angle Θ represent a form of a circle, which is known as Mohr's Circle.

Mohr's circle is the circle in which each point represents a plane in a stressed body in which the x-coordinate of the point represents the Normal stress, and the y-coordinate represents the Shear stress acting on the plane.

Mohr's Circle diagram can be used for creating various questions in the GATE question paper, and the use of this has been seen over the years.

Problem statement:

EX: Draw the Mohr's Circle of the stress element shown below. Determine the principal stresses and the maximum shear stresses.

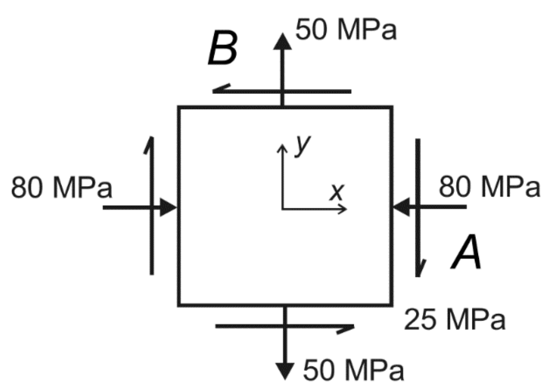
σ_x = -80 MPa

σ_y = +50 MPa

τ_{xy} = 25 MPa

Coordinates of Points:

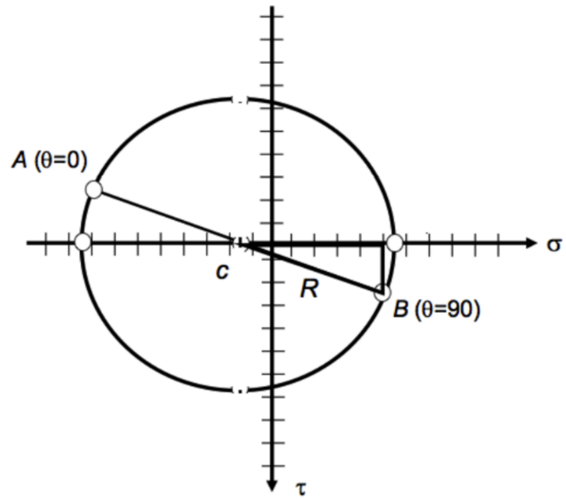
A: (-80,25) B: (50, -25)



<u>Sol</u>

C= $\sigma_{\text{avg}} = \frac{\sigma_x+\sigma_y}{2} = \frac{-80+15}{2}$ = -15

R= $\sqrt{\left(\frac{\sigma_x-\sigma_y}{2}\right)^2 + \tau_{xy}^2}$ = $\sqrt{(65)^2+(25)^2}$ =69.6

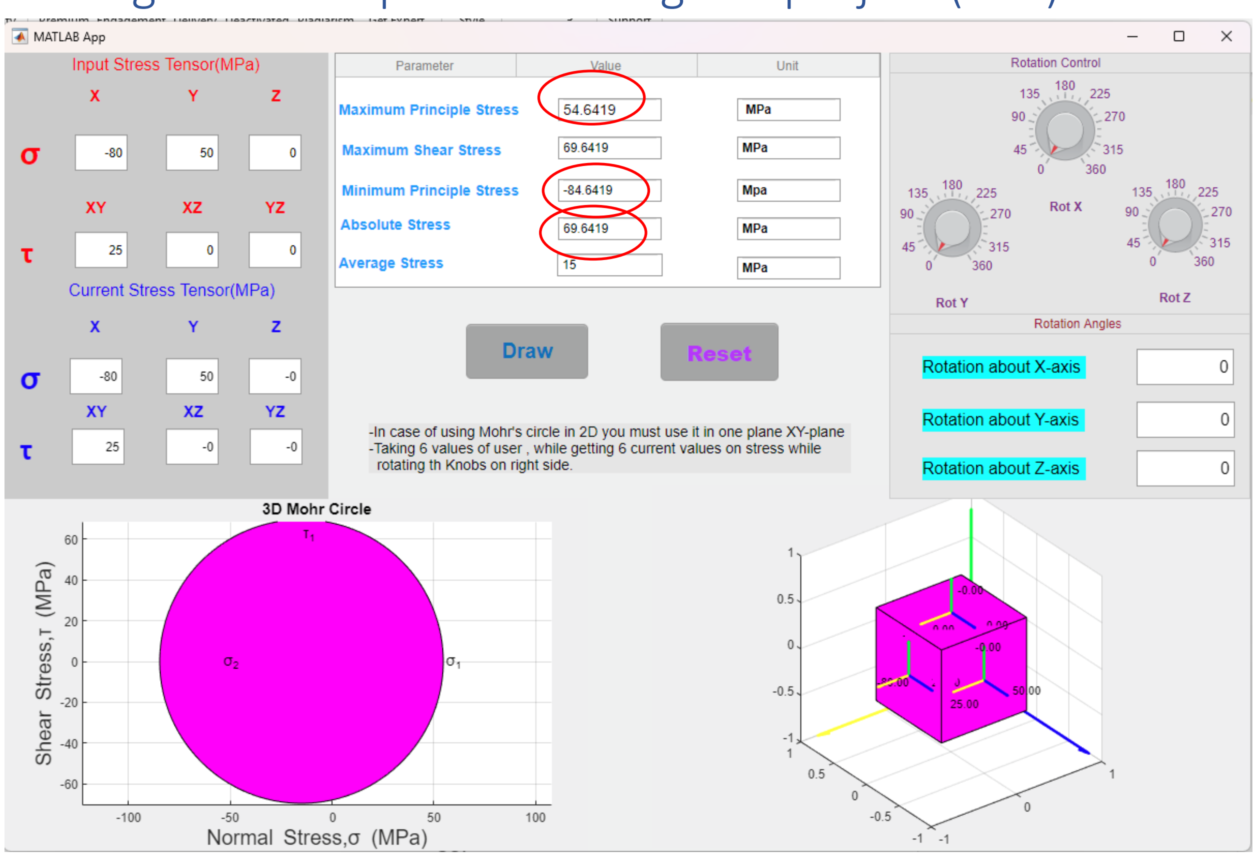


$$\sigma_{p1,p2} = c \pm R = -15 \pm 69.6$$

σ_{p1} = 54.6 MPa , σ_{p2} = -84.6 MPa

$$\tau_{max} = R = \ 69.6 \ \text{MPA}$$

Solving the same problem using our project (GUI):



Flow Chart:


Start
Get input stress from user
Draw the Mohr's circle
using stress equations
Get values , Max , min stress to show for user
Get the rotating values from the Knobs
put the values on the sress cube
Get the rotating values from the Knobs
put the values on the sress cube
Show the result of rotating on current stress
End

Code:

Our code can be divided into main Function

First function we use is "Rotate "

This function is used for rotating element in 3D in three angles, we use it for rotating the stress Cube.

```
function rotation = rotate(app,matrix,rotx,roty,rotz)

                %%about x-axis

        w=[1      0              0            0;
           0      cosd(rotx)    -sind(rotx)       0;
           0      sind(rotx)    cosd(rotx)        0;
           0      0               0           1];

                %%about y-axis

        r=[cosd(roty)    ,  0,    sind(roty)      0;
            0             1 ,      0           0;
            -sind(roty)    0      cosd(roty)      0;
            0             0        0           1];
                %%about z-axis

        t=[cosd(rotz)  -sind(rotz)  0  0;
           sind(rotz)  cosd(rotz)   0  0;
            0       0               1  0;
            0       0               0  1];

            rotation =w*r*t*matrix;
        end
```

second Function is "cube_3D"

this function is used for drawing a the stress cube and rotating it in the 3D while rotating vectors and variables of current stress in 3D too.

```
  function cube_3D(app,rotx,roty,rotz)

         cla(app.UIAxes,'reset');

        sigma_x = -(app.EditField_8.Value);
        sigma_y = -(app.EditField_9.Value);
        sigma_z = -(app.EditField_13.Value);

        tau_xy = -(app.EditField_10.Value);
        tau_yz = -(app.EditField_11.Value);
        tau_zx = -(app.EditField_12.Value);
```

```
        sigma_x_dash=-((sigma_x+sigma_y)/2 + ((sigma_x-
sigma_y)/2)*cos(2*rotx)+tau_xy*sin(2*rotx));
        sigma_y_dash=-((sigma_y+sigma_z)/2 + ((sigma_y-
sigma_z)/2)*cos(2*roty)+tau_yz*sin(2*roty));
        sigma_z_dash=-((sigma_z+sigma_x)/2 + ((sigma_z-
sigma_x)/2)*cos(2*rotz)+tau_zx*sin(2*rotz));

        tau_xy_dash=-((-(sigma_x-sigma_y)/2)*sin(2*rotx)+tau_xy*cos(2*rotx));
        tau_yz_dash=-((-(sigma_y-sigma_z)/2)*sin(2*roty)+tau_yz*cos(2*roty));
        tau_zx_dash=-((-(sigma_z-sigma_x)/2)*sin(2*rotz)+tau_zx*cos(2*rotz));


        tau_yx_dash=tau_xy_dash;
        tau_zy_dash=tau_yz_dash;
        tau_xz_dash=tau_zx_dash;



         view(app.UIAxes,3);


        c=[-.5 -.5 -.5 ;
           -.5 .5 -.5 ;
           .5 .5 -.5 ;
           .5 -.5 -.5 ;
           -.5 -.5 .5 ;
           -.5 .5 .5 ;
           .5 .5 .5 ;
           .5 -.5 .5 ];
        c=c';
        a=[c; ones(1,8)];

        faces = [1 2 3 4; 2 6 7 3; 4 3 7 8; 1 5 8 4; 1 2 6 5; 5 6 7 8];

        r=[0   0    0   0    0  -0.4;
           0   0    0  -0.4  0   0  ;
           0.5 0.4  0.5 0    0.5 0  ];
        mt1=[r; 1 1 1 1 1 1];
        rt1 = rotate(app,mt1,rotx,roty,rotz);

        q=[ 0    0     0   0     0   -0.4;
           -0.5 -0.4  -0.5 0    -0.5  0  ;
            0    0     0   0.4   0    0  ];
        mt2=[q ; 1 1 1 1 1 1];
        rt2 = rotate(app,mt2,rotx,roty,rotz);

        k=[-0.5 -0.4 -0.5  0   -0.5  0  ;
            0    0    0    0    0   -0.4;
            0    0    0    0.4  0    0  ];
        mt3=[k ; 1 1 1 1 1 1];
```

```
        rt3 = rotate(app,mt3,rotx,roty,rotz);

        axis(app.UIAxes, 'equal');
        set(app.UIAxes, 'View', [45 45]);%axis equal and look good



quiver3(app.UIAxes,rt1(1,1),rt1(2,1),rt1(3,1),rt1(1,2),rt1(2,2),rt1(3,2),'linewid
th',2,'color','g');
        hold (app.UIAxes,"on");

quiver3(app.UIAxes,rt1(1,3),rt1(2,3),rt1(3,3),rt1(1,4),rt1(2,4),rt1(3,4),'linewid
th',2,'color','b');

quiver3(app.UIAxes,rt1(1,5),rt1(2,5),rt1(3,5),rt1(1,6),rt1(2,6),rt1(3,6),'linewid
th',2,'color','y');

quiver3(app.UIAxes,rt2(1,1),rt2(2,1),rt2(3,1),rt2(1,2),rt2(2,2),rt2(3,2),'linewid
th',2,'color','b');

quiver3(app.UIAxes,rt2(1,3),rt2(2,3),rt2(3,3),rt2(1,4),rt2(2,4),rt2(3,4),'linewid
th',2,'color','g');

quiver3(app.UIAxes,rt2(1,5),rt2(2,5),rt2(3,5),rt2(1,6),rt2(2,6),rt2(3,6),'linewid
th',2,'color','y');


quiver3(app.UIAxes,rt3(1,1),rt3(2,1),rt3(3,1),rt3(1,2),rt3(2,2),rt3(3,2),'linewid
th',2,'color','y');

quiver3(app.UIAxes,rt3(1,3),rt3(2,3),rt3(3,3),rt3(1,4),rt3(2,4),rt3(3,4),'linewid
th',2,'color','g');

quiver3(app.UIAxes,rt3(1,5),rt3(2,5),rt3(3,5),rt3(1,6),rt3(2,6),rt3(3,6),'linewid
th',2,'color','b');

        quiver3(app.UIAxes,1,1,-1,-2,0,0,'linewidth',2,'color','y');
        quiver3(app.UIAxes,1,1,-1,0,-2,0,'linewidth',2,'color','b');
        quiver3(app.UIAxes,1,1,-1,0,0,2,'linewidth',2,'color','g');
        %patch('Vertices',a,'Faces',faces,'FaceColor','r','facealpha',0.3);

        h = rotate(app,a,rotx,roty,rotz);
        h=h';
        results=h(:,(1:3));
        hold (app.UIAxes,"on");

patch(app.UIAxes,'Vertices',results,'Faces',faces,'FaceColor','m','facealpha',1);

        p1=[-0.8 0.1 0;
            0.1 -0.8 -0.1;
```

```
            0 0 0.8];
        p2=[p1;1 1 1];
        p3=rotate(app,p2,rotx,roty,rotz);


text(app.UIAxes,p3(1,1),p3(2,1),p3(3,1),sprintf('%0.2f',sigma_x_dash),'color','k'
,'fontsize',11);

text(app.UIAxes,p3(1,2),p3(2,2),p3(3,2),sprintf('%0.2f',sigma_y_dash),'color','k'
,'fontsize',11);

text(app.UIAxes,p3(1,3),p3(2,3),p3(3,3),sprintf('%0.2f',sigma_z_dash),'color','k'
,'fontsize',11);

        p4=[-0.5  -0.1 -0.35;
            -0.35 -0.5 -0.1;
             0.1   0.4  0.5];
        p5=[p4;1 1 1];
        p6=rotate(app,p5,rotx,roty,rotz);


text(app.UIAxes,p6(1,1),p6(2,1),p6(3,1),sprintf('%0.2f',tau_xy_dash),'color','k',
'fontsize',11);

text(app.UIAxes,p6(1,2),p6(2,2),p6(3,2),sprintf('%0.2f',tau_yz_dash),'color','k',
'fontsize',11);

text(app.UIAxes,p6(1,3),p6(2,3),p6(3,3),sprintf('%0.2f',tau_zx_dash),'color','k',
'fontsize',11);

        p7=[-0.4  0.1  -0.5;
            -0.5 -0.35  0.1;
            -0.1  0.5   0.4];
        p8=[p7;1 1 1];
        p9=rotate(app,p8,rotx,roty,rotz);


text(app.UIAxes,p9(1,1),p9(2,1),p9(3,1),sprintf('%0.2f',tau_yx_dash),'color','k',
'fontsize',11);

text(app.UIAxes,p9(1,2),p9(2,2),p9(3,2),sprintf('%0.2f',tau_zy_dash),'color','k',
'fontsize',11);

text(app.UIAxes,p9(1,3),p9(2,3),p9(3,3),sprintf('%0.2f',tau_xz_dash),'color','k',
'fontsize',11);

        xlim(app.UIAxes,[-1 1]);
        ylim(app.UIAxes,[-1 1]);
        zlim(app.UIAxes,[-1 1]);
         hold (app.UIAxes,"on");
```

```
        grid (app.UIAxes,"on");

        end
```

third function is “Tensor”

this function is used to show and solve the equation of current stress tensor , and applying them on field to show them while rotating the knobs

```
  function  Tensor(app,rotx,roty,rotz)
       %get from user
        sigma_x = -(app.EditField_8.Value);
        sigma_y = -(app.EditField_9.Value);
        sigma_z = -(app.EditField_13.Value);

        tau_xy = -(app.EditField_10.Value);
        tau_yz = -(app.EditField_11.Value);
        tau_zx = -(app.EditField_12.Value);
       %current tensor equation
        sigma_x_dash=-((sigma_x+sigma_y)/2 + ((sigma_x-
sigma_y)/2)*cos(2*rotx)+tau_xy*sin(2*rotx));
        sigma_y_dash=-((sigma_y+sigma_z)/2 + ((sigma_y-
sigma_z)/2)*cos(2*roty)+tau_yz*sin(2*roty));
        sigma_z_dash=-((sigma_z+sigma_x)/2 + ((sigma_z-
sigma_x)/2)*cos(2*rotz)+tau_zx*sin(2*rotz));

        tau_xy_dash=-((-(sigma_x-sigma_y)/2)*sin(2*rotx)+tau_xy*cos(2*rotx));
        tau_yz_dash=-((-(sigma_y-sigma_z)/2)*sin(2*roty)+tau_yz*cos(2*roty));
        tau_zx_dash=-((-(sigma_z-sigma_x)/2)*sin(2*rotz)+tau_zx*cos(2*rotz));


        tau_yx_dash=tau_xy_dash;
        tau_zy_dash=tau_yz_dash;
        tau_xz_dash=tau_zx_dash;
       %show the results on field ...
        app.EditField_14.Value=double(sigma_x_dash);
        app.EditField_15.Value=double(sigma_y_dash);
        app.EditField_16.Value=double(sigma_z_dash);

        app.EditField_17.Value=double(tau_xy_dash);
        app.EditField_18.Value=double(tau_yz_dash);
        app.EditField_19.Value=double(tau_zx_dash);

        end
```

Fourth function is “mohrs_3d_2d”
This function is used in calculating the values of maximum, minimum, average, absolute and drawing the Mohr’s circle in a 2D case and 3D case also with special condition of drawing 2D in XY-plan only.

```
function Mohr_3d_2d(app)

        cla(app.UIAxes2, "reset");
       %values from user...
        sigma_x = -(app.EditField_8.Value);
        sigma_y = -(app.EditField_9.Value);
        sigma_z = -(app.EditField_13.Value);

        tau_xy = -(app.EditField_10.Value);
        tau_yz = -(app.EditField_11.Value);
        tau_zx = -(app.EditField_12.Value);

        tau_yx=tau_xy;
        tau_zy=tau_yz;
        tau_xz=tau_zx;

        % Coefficients for Mohr's Ciecle
        c3 = 1;
        c2 = sigma_x+sigma_y+sigma_z;
        c1 = sigma_x*sigma_y+sigma_y*sigma_z+sigma_z*sigma_x -tau_xy^2 -tau_yz^2
-tau_zx^2;
        c0 = sigma_x*sigma_y*sigma_z + 2*tau_xy*tau_yz*tau_zx -
(sigma_x*tau_yz^2+sigma_y*tau_zx^2+sigma_z*tau_xy^2);

        % Principal stresses
        normal_stresses = roots([c3 c2 c1 c0]);
        A = sort(normal_stresses,'descend');

        sigma_1 = A(1);
        sigma_2 = A(2);
        sigma_3 = A(3);
        sigma = [sigma_1;sigma_2;sigma_3];

        sigma_max = max(sigma);
        sigma_min = min(sigma);

        tau_1 = (sigma_1-sigma_3)/2;
        tau_2 = (sigma_1-sigma_2)/2;
        tau_3 = (sigma_2-sigma_3)/2;
        tau = [tau_1;tau_2;tau_3];

        tau_max = max(tau);

        absolute_stress=abs((sigma_max-sigma_min)/2);
```

```
        app.MaximumPrincipleStressEditField.Value= num2str(sigma_max);
        app.MinimumPrincipleStressEditField.Value= num2str(sigma_min);
        app.MaximumShearStressEditField.Value=num2str(tau_max);

        app.AbsoluteStressEditField.Value=num2str(absolute_stress);

        % Plotting
        theta = 0:0.01:2*pi;

        C1 = [(sigma_1+sigma_3)/2 0];
        C2 = [(sigma_1+sigma_2)/2 0];
        C3 = [(sigma_2+sigma_3)/2 0];

        %cirlce1=[C1(1)+tau_1*cos(theta') C1(2)+tau_1*sin(theta')];
        %cirlce2=[C2(1)+tau_2*cos(theta') C2(2)+tau_2*sin(theta')];
        %cirlce3=[C3(1)+tau_3*cos(theta') C3(2)+tau_3*sin(theta')];

        %% for the 2d condiong
        if sigma_z==0 && tau_yz==0 && tau_xz==0

          % sigma_max_2D=(sigma_x+sigma_y)/2+(((sigma_x-sigma_y)/2)^(2) +
(tau_xy)^(2))^(1/2)

          %  sigma_max_2D = max(sigma_max_2D);
          %  sigma_min_2D = min(MAt_MAX_MIN_2D);

          %   app.MaximumPrincipleStressEditField.Value= num2str(sigma_max_2D);
          %  app.MinimumPrincipleStressEditField.Value= num2str(sigma_min_2D);


%              Cemter_2d=(sigma_x+sigma_y)/2;
%              raduis_2d=(((sigma_x+sigma_y)/2)^(2)+(tau_xy)^(2))^(1/2);
%              sigma_max_2D = Cemter_2d+raduis_2d;
%              sigma_min_2D = Cemter_2d-raduis_2d;

                    %plotting
            plot(app.UIAxes2,C1(1)+tau_1*cos(theta),C1(2)+tau_1*sin(theta),'m');
            axis(app.UIAxes2,"equal")
            grid (app.UIAxes2,"on")
            fill(app.UIAxes2,C1(1)+tau_1*cos(theta),C1(2)+tau_1*sin(theta),"m");
             %labels
            if sigma_1>0

                text(app.UIAxes2,sigma_1*1.01,0,'\sigma_1','fontsize',15);
            else

                text(app.UIAxes2,sigma_1*0.95,0,'\sigma_1','fontsize',15);
            end

            if sigma_2>0
```

```
                text(app.UIAxes2,-sigma_1*1.0,0,'\sigma_2','fontsize',15);
            else

                text(app.UIAxes2,-sigma_1*0.99,0,'\sigma_2','fontsize',15);
            end

            text(app.UIAxes2,C1(1),tau_1*0.9,'\tau_1','fontsize',15);

            %average for the 2d
            average_stress_2D=(sigma_x+sigma_y)/2;
            app.AverageStressEditField.Value=num2str(average_stress_2D);

            %for the 3d condisions the rest of them
        elseif (sigma_x~=0 && sigma_y~=0 && sigma_z~=0) && (tau_xy~=0 ||
tau_yz~=0 || tau_xz~=0)

             sigma_1 = A(1);
             sigma_2 = A(2);
             sigma_3 = A(3);
             sigma = [sigma_1;sigma_2;sigma_3];

             sigma_max = max(sigma);
             sigma_min = min(sigma);
             app.MaximumPrincipleStressEditField.Value= num2str(sigma_max);
             app.MinimumPrincipleStressEditField.Value= num2str(sigma_min);
                %plotting
            plot(app.UIAxes2,C1(1)+tau_1*cos(theta),C1(2)+tau_1*sin(theta),'m');
            hold (app.UIAxes2,"on")
            plot(app.UIAxes2,C2(1)+tau_2*cos(theta),C2(2)+tau_2*sin(theta),'g');
            hold (app.UIAxes2,"on")
            plot(app.UIAxes2,C3(1)+tau_3*cos(theta),C3(2)+tau_3*sin(theta),'r');
            hold (app.UIAxes2,"on")
            axis(app.UIAxes2,"equal")
            grid (app.UIAxes2,"on")
            %labels
            if sigma_1>0
                text(app.UIAxes2,sigma_1*1.01,0,'\sigma_1','fontsize',15);
            else
                text(app.UIAxes2,sigma_1*0.95,0,'\sigma_1','fontsize',15);
            end

            if sigma_2>0
                text(app.UIAxes2,sigma_2*1.0,0,'\sigma_2','fontsize',15);
            else
                text(app.UIAxes2,sigma_2*0.99,0,'\sigma_2','fontsize',15);
            end

            if sigma_3>0
                text(app.UIAxes2,sigma_3*1.1,0,'\sigma_3','fontsize',15);
```

```
            else
                text(app.UIAxes2,sigma_3*0.99,0,'\sigma_3','fontsize',15);
            end

            text(app.UIAxes2,C1(1),tau_1*0.9,'\tau_1','fontsize',15);
            text(app.UIAxes2,C2(1),tau_2*0.9,'\tau_2','fontsize',15);
            text(app.UIAxes2,C3(1),tau_3*0.9,'\tau_3','fontsize',15);

             %avergae for the 3d
            average_stress_3D=(sigma_x+sigma_y+sigma_z)/3;
            app.AverageStressEditField.Value=num2str(average_stress_3D);

        elseif (tau_xy~=0 && tau_yz~=0 && tau_xz~=0) && (sigma_x~=0 || sigma_y~=0
|| sigma_z~=0)

            sigma_1 = A(1);
             sigma_2 = A(2);
             sigma_3 = A(3);
             sigma = [sigma_1;sigma_2;sigma_3];

             sigma_max = max(sigma);
             sigma_min = min(sigma);
             app.MaximumPrincipleStressEditField.Value= num2str(sigma_max);
             app.MinimumPrincipleStressEditField.Value= num2str(sigma_min);
            %plotting
            plot(app.UIAxes2,C1(1)+tau_1*cos(theta),C1(2)+tau_1*sin(theta),'m');
            hold (app.UIAxes2,"on")
            plot(app.UIAxes2,C2(1)+tau_2*cos(theta),C2(2)+tau_2*sin(theta),'g');
            hold (app.UIAxes2,"on")
            plot(app.UIAxes2,C3(1)+tau_3*cos(theta),C3(2)+tau_3*sin(theta),'r');
            hold (app.UIAxes2,"on")
            axis(app.UIAxes2,"equal")
            grid (app.UIAxes2,"on")



             %labels
            if sigma_1>0
                text(app.UIAxes2,sigma_1*1.01,0,'\sigma_1','fontsize',15);
            else
                text(app.UIAxes2,sigma_1*0.95,0,'\sigma_1','fontsize',15);
            end

            if sigma_2>0
                text(app.UIAxes2,sigma_2*1.0,0,'\sigma_2','fontsize',15);
            else
                text(app.UIAxes2,sigma_2*0.99,0,'\sigma_2','fontsize',15);
            end

            if sigma_3>0
```

```
        text(app.UIAxes2,sigma_3*1.1,0,'\sigma_3','fontsize',15);
    else
        text(app.UIAxes2,sigma_3*0.99,0,'\sigma_3','fontsize',15);
    end

    text(app.UIAxes2,C1(1),tau_1*0.9,'\tau_1','fontsize',15);
    text(app.UIAxes2,C2(1),tau_2*0.9,'\tau_2','fontsize',15);
    text(app.UIAxes2,C3(1),tau_3*0.9,'\tau_3','fontsize',15);

    %avergae for the 3d
    average_stress_3D=(sigma_x+sigma_y+sigma_z)/3;
    app.AverageStressEditField.Value=num2str(average_stress_3D);

else

    sigma_1 = A(1);
    sigma_2 = A(2);
    sigma_3 = A(3);
    sigma = [sigma_1;sigma_2;sigma_3];

    sigma_max = max(sigma);
    sigma_min = min(sigma);
    app.MaximumPrincipleStressEditField.Value= num2str(sigma_max);
    app.MinimumPrincipleStressEditField.Value= num2str(sigma_min);
          %plotting
    plot(app.UIAxes2,C1(1)+tau_1*cos(theta),C1(2)+tau_1*sin(theta),'m');
    hold (app.UIAxes2,"on")
    plot(app.UIAxes2,C2(1)+tau_2*cos(theta),C2(2)+tau_2*sin(theta),'g');
    hold (app.UIAxes2,"on")
    plot(app.UIAxes2,C3(1)+tau_3*cos(theta),C3(2)+tau_3*sin(theta),'r');
    hold (app.UIAxes2,"on")
    axis(app.UIAxes2,"equal")
    grid (app.UIAxes2,"on")

       %labels
    if sigma_1>0
        text(app.UIAxes2,sigma_1*1.01,0,'\sigma_1','fontsize',15);
    else
        text(app.UIAxes2,sigma_1*0.95,0,'\sigma_1','fontsize',15);
    end

    if sigma_2>0
        text(app.UIAxes2,sigma_2*1.0,0,'\sigma_2','fontsize',15);
    else
        text(app.UIAxes2,sigma_2*0.99,0,'\sigma_2','fontsize',15);
    end

    if sigma_3>0
        text(app.UIAxes2,sigma_3*1.1,0,'\sigma_3','fontsize',15);
    else
```

```
            text(app.UIAxes2,sigma_3*0.99,0,'\sigma_3','fontsize',15);
        end

        text(app.UIAxes2,C1(1),tau_1*0.9,'\tau_1','fontsize',15);
        text(app.UIAxes2,C2(1),tau_2*0.9,'\tau_2','fontsize',15);
        text(app.UIAxes2,C3(1),tau_3*0.9,'\tau_3','fontsize',15);

        %avergae for the 3d
        average_stress_3D=(sigma_x+sigma_y+sigma_z)/3;
        app.AverageStressEditField.Value=num2str(average_stress_3D);

    end

    xlabel(app.UIAxes2,'Normal Stress,\sigma (MPa)','fontsize',15);
    ylabel(app.UIAxes2,'Shear Stress,\tau (MPa)','fontsize',15);
    title(app.UIAxes2,'3D Mohr Circle','fontsize',15);
    end
```

the fifth function is "Restart"
this function is used to restart all the program to reinput the values and draw again and get outputs.

```
function restart(app)

    cla(app.UIAxes,'reset');
    cla(app.UIAxes2,'reset');

    Tensor(app,0,0,0);
    %restrt zero values in all fields
    app.MaximumPrincipleStressEditField.Value= num2str(0);
    app.MinimumPrincipleStressEditField.Value= num2str(0);
    app.MaximumShearStressEditField.Value=num2str(0);
    app.AbsoluteStressEditField.Value=num2str(0);
    app.AverageStressEditField.Value=num2str(0);

    app.EditField_8.Value=0;
    app.EditField_9.Value=0;
    app.EditField_13.Value=0;

    app.EditField_10.Value=0;
    app.EditField_11.Value=0;
    app.EditField_12.Value=0;

    app.EditField_14.Value=double(0);
    app.EditField_15.Value=double(0);
    app.EditField_16.Value=double(0);

    app.EditField_17.Value=double(0);
    app.EditField_18.Value=double(0);
    app.EditField_19.Value=double(0);
```

```
app.EditField_23.Value=0;
app.EditField_24.Value=0;
app.EditField_22.Value=0;

app.RotZKnob.Value=0;
app.RotYKnob.Value=0;
app.RotXKnob.Value=0;

end
```

the whole code is written by us only, no use of out functions, by the use of Doctor and teacher assistant and with help of another teacher assistant in production department to finish this project.

Conclusion:

The conclusion of this project can be that mohr's circle is a good way to get easy outputs for stress analysis, and hard to be in the real world because it takes much time to be drawn and getting values, but with coding we can get it with just klick on a button. So we finish this report with best wish to use our project.

Refrences:

- Mechanics of Materials for Dummies by James H. Allen III
- Metal forming third Edition Mechanics and Metallurgy
- https://pantelisliolios.com/mohr-circle-3d/
- https://byjusexamprep.com/mohrs-circle-i
- https://www.roymech.co.uk/Useful_Tables/Mechanics/Mohrs_circle.html