In [1]:
```
import numpy as np
from IPython.display import Image, display, HTML, Latex as lt
```

In [2]:
```
%%javascript
MathJax.Extension["TeX/cancel"]={version:"2.4.0",ALLOWED:{color:1,mathcolor:1,bac
```

In [3]:
```
def get_sigma(N, A, M, Iz0, y):
    sigma = (N/A) + (M/Iz0)*y
    return sigma

def get_tau(V, Ms, b, Iz0):
    tau = (V*Ms)/(b*Iz0)
    return tau

def get_sigma_c(sigma_x, sigma_y):
    sigma_c = (sigma_x + sigma_y)/2
    return sigma_c

def get_mohr_radius(sigma_x, sigma_y, tau_xy):
    radius = np.sqrt(((sigma_x - sigma_y)/(2))**2 + (tau_xy)**2)
    return radius

def get_sigma_1(sigma_c, radius):
    sigma_1 = sigma_c + radius
    return sigma_1

def get_sigma_2(sigma_c, radius):
    sigma_2 = sigma_c - radius
    return sigma_2

def get_tau_max(sigma_c, radius):
    tau_max = radius
    return tau_max

def get_tau_min(sigma_c, radius):
    tau_min = -radius
    return tau_min

def get_alpha_point(alpha, sigma_x, sigma_y, tau_xy):
    alpha = np.deg2rad(alpha)
    sigma_alpha = (sigma_x + sigma_y)/2 + \
                  ((sigma_x - sigma_y)/2)*np.cos(2*alpha) + \
                  tau_xy*np.sin(2*alpha)
    tau_alpha = -((sigma_x - sigma_y)/2)*np.sin(2*alpha) + \
                tau_xy*np.cos(2*alpha)
    return sigma_alpha, tau_alpha

def get_angle(sigma_p, tau_p, sigma_a, tau_a):
    '''
    funcao que determina o angulo entre uma reta vertical passando pelo polo P(si
    e a reta que liga P a um ponto A(sigma_a, tau_a)
    '''
    tan_angle = (sigma_p - sigma_a)/(tau_p - tau_a)
    angle = np.arctan(tan_angle)
    angle = np.rad2deg(angle)
    return angle
```

MANUAL DE EXERCÍCIO

Um exercício de Estado Duplo costuma ser uma viga com cargas (normalmente pontuais) e uma seção.

1. Encontrar os valores de N, V e M no ponto requisitado;

2. Na seção transversal dada, definir o sinal de y e M segundo o seguinte critério:
   A. Verificar o sentido do eixo da seção transversal (por exemplo, na Figura 1 o eixo z aponta para cima);
   B. Verificar qual o lado que o M traciona. Definimos positivo o M que traciona o lado que o eixo aponta, no caso, M é positivo quando tracionamos a parte de cima da viga;

3. Calcular M_s, através da fórmula $M_s^* = z_G . A^*$, onde A^* denota a área da região que o ponto de interesse divide da seção. No exemplo da Figura 1 é a área do retângulo acima de C, ou seja 3.3, neste mesmo exemplo $M_s^* = \frac{4+1}{2} . ((4-1).3) = \frac{45}{2}$;

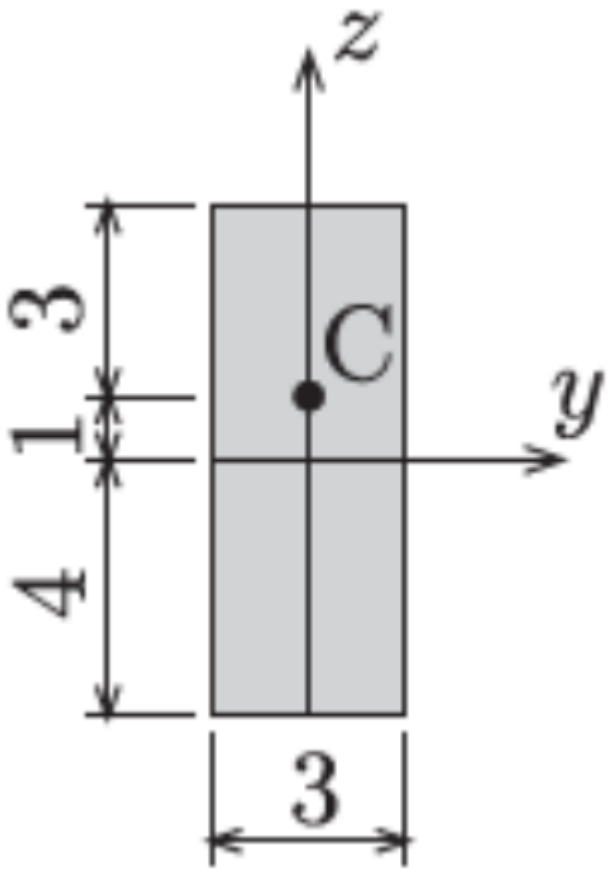

Figura 1 - Exemplo de seção transversal cobrado em exercício

4. Se necessário, calcular I_{y0} e A, mas estes normalmente são dados no enunciado;

5. Com esses dados, calcular σ_x e τ_{xy} através das funções get_sigma(N, A, M, Iz0, y) e get_tal(V, Ms, b, Iz0);
   A. A função get_sigma nada mais faz do que calcular a seguinte fórmula:
   $$\sigma = \frac{N}{A} + \frac{M}{I_{z0}}.y$$
   lembrando que no caso do exemplo da Figura 1, temos que usar I_{y0} e z no lugar de Y_{z0} e y, respectivamente;
   B. Analogamente, a função get_tau calcula a seguinte fórmula:
   $$\tau = \frac{V.M_s^*}{b^*.I_{z0}}$$

6. Se o enunciado não fornecer o valor de σ_y, este provavelmente é nulo (propriedade de vigas horizontais). Logo o triângulo de tensões está definido;

Triângulo de uma seção:

$\tau_{xy} \equiv$ tensão de cisalhamento da face vertical do triângulo

$\sigma_x \equiv$ tensão normal direcionada ao eixo x, ou seja, tensão normal do plano vertical

$\sigma_y \equiv$ tensão normal direcionada ao eixo y, ou seja, tensão normal do plano horizontal

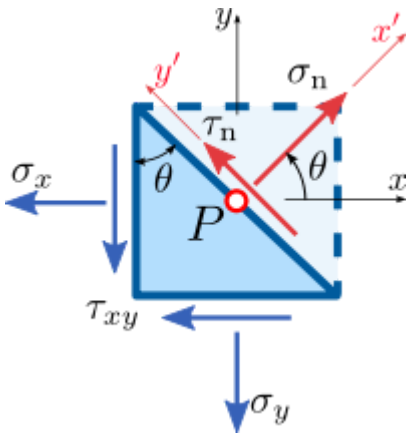

Figura 2 - Triângulo de Tensões do Estado Duplo

7. Desenhar o plano (σ, τ) e localizar nele os pontos $V = (\sigma_x, \tau_x)$ e $H = (\sigma_y, \tau_y)$, lembrando que é provável que σ_y seja 0;

8. Desenhar uma reta horizontal a partir do ponto (σ_y, τ_y) e uma reta vertical a partir de (σ_x, τ_x). O encontro dessas duas retas gera um ponto $P = (\sigma_P, \tau_P)$, o pólo do diagrama (ou seja, $\sigma_P = \sigma_x$ e $\tau_P = \tau_{yx}$);

9. Encontrar o centro $C = (\sigma_C, 0)$ do Círculo de Mohr através da relação $\sigma_C = \frac{\sigma_x + \sigma_y}{2}$;

10. Traçar o Círculo em si, como na Figura 3;

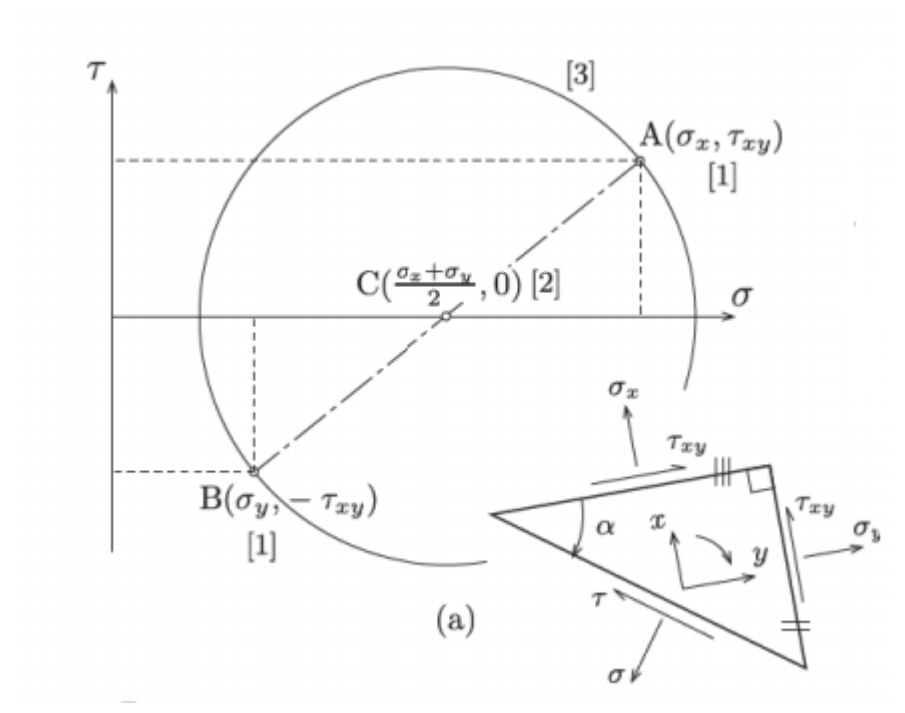

Figura 3 - Círculo de Mohr traçado

11. Transferir no pólo, tendo uma reta vertical como referência, o ângulo α e traçar uma reta com essa inclinação que encontre o Círculo;

12. A partir do pólo P, traçar reta para os 4 pontos "extremos" do Círculo: os dois pontos de τ extremos ($\tau_{máx}$ e $\tau_{mín}$) e os dois pontos de σ extremos ($\sigma_1 > \sigma_2$) e encontrar os ângulos que estas retas fazem com a vertical traçada no pólo;

13. Estes valores podem ser encontrados algébricamente através das seguintes relações:

$$\begin{cases} R & = & \sqrt{\left(\frac{\sigma_x - \sigma_y}{2}\right)^2 + \tau_{xy}^2} \\ \sigma_1 & = & \sigma_C + R \\ \sigma_2 & = & \sigma_C - R \\ \tau_{max} & = & R \\ \tau_{min} & = & -\tau_{max} = -R \end{cases}$$

14. Traçar retas ligando o pólo P aos pontos extremos. Em cada reta, desenhar as tensões extremas, como por exemplo na Figura 4, notando os seguintes detalhes:
    A. Na reta, trace um "chão" (hachurado) a ser usado de referência para o sinal das tensões;
    B. σ positivo aponta pra fora deste hachurado e τ positivo aponta no sentido horário dentro do triângulo $\triangle\ P\tau_{max}\tau_{min}$;

Um adendo sobre a Figura 4: ela trata de um exercício em particular onde o triângulo de tensões não tem suas bases ortogonais alinhadas com a base, ou seja, um triângulo tombada. Por este motivo que $\sigma_P \neq \sigma_x$ e $\tau_P \neq \tau_{xy}$;

15. Para traçar um prisma de tensões, copie triângulo $\triangle P\sigma_1\sigma 2$ e transporte as tensões de cada lado para o lado correspondente. Em seguida, transfira as retas $\overline{P\tau_{max}}$ e $\overline{P\tau_{min}}$ para os vértices σ_1 e σ_2 de forma que as duas retas consigam se encontrar e transporte as trações destes lados analogamente ao que foi feito com os primeiros lados;

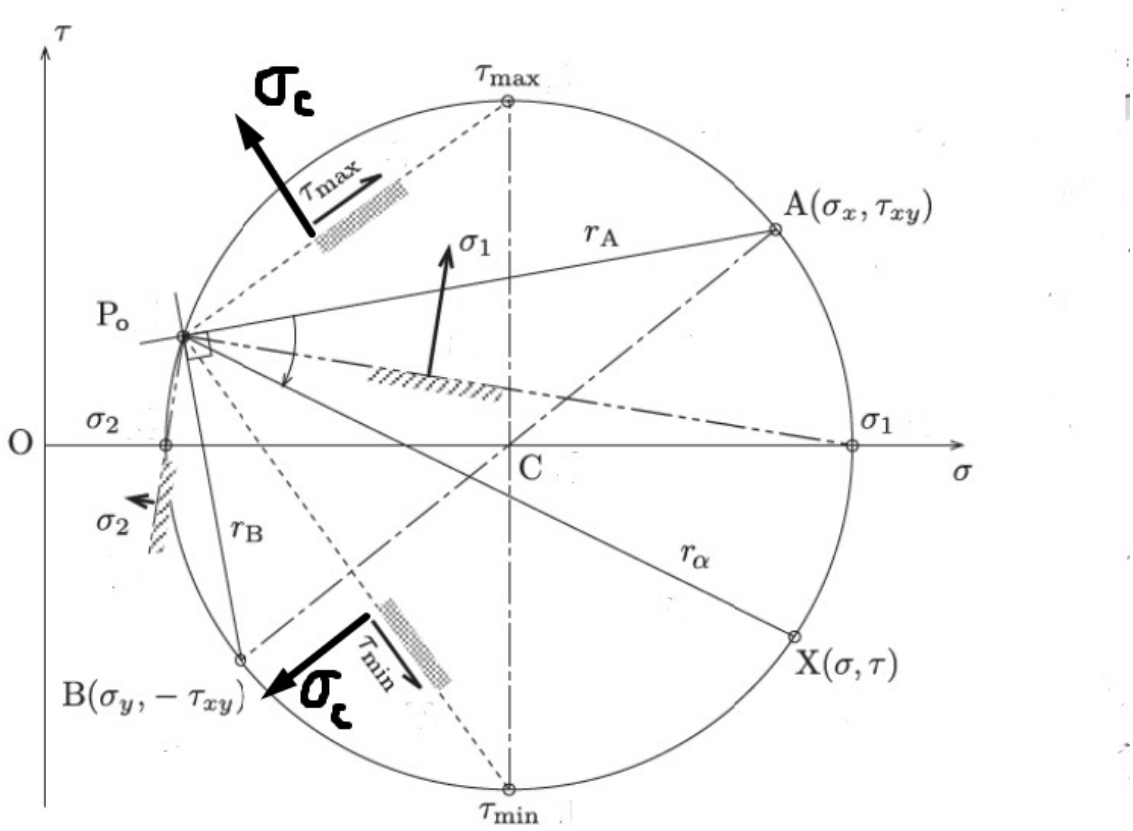

Figura 4 - Círculo de Mohr com os 4 planos extremos

16. Se requisitado, calcular os ângulos que definem os planos extremos através da fórmula

$$\tan\theta = \frac{\sigma_P - \sigma_A}{\tau_P - \tau_A} \Leftrightarrow \theta = \tan^{-1}\frac{\sigma_P - \sigma_A}{\tau_P - \tau_A}$$

17. Se requisitado, calcular o valor numérico das tensões no plano correspondente ao ângulo α através das equações (1) e (2):

$$\sigma(\alpha) = \frac{\sigma_x + \sigma_y}{2} + \frac{\sigma_x - \sigma_y}{2}\cos 2\alpha + \tau_{xy}\sin 2\alpha \quad (1)$$

$$\tau(\alpha) = -\frac{\sigma_x - \sigma_y}{2}\sin 2\alpha + \tau_{xy}\cos 2\alpha \quad (2)$$

Resolução exercício 1 (P1-2018)

1ª Questão (4,0 pontos)

A viga prismática da figura está submetida a duas cargas transversais além das forças axiais de tração. Para a seção transversal d-d, pedem-se:

a) os esforços solicitantes;

b) as tensões normal e tangencial no ponto Q atuantes no plano da seção (não deixar de indicar as direções e sentidos das tensões);

c) o círculo de Mohr representando o estado duplo de tensão em Q;

d) os valores das tensões normais e tangenciais extremas em Q e as direções dos planos em que ocorrem (indicar os resultados em prismas de tensões);

e) as componentes normal e tangencial do vetor tensão em Q atuante no plano inclinado de 30° indicado na figura.

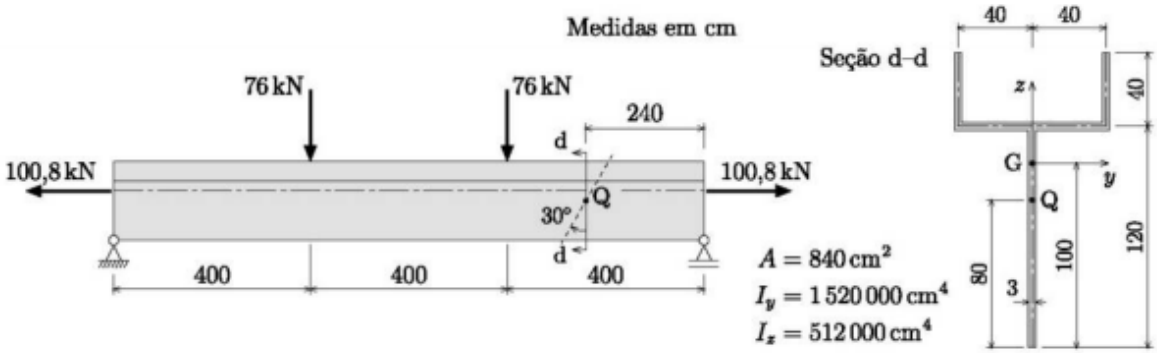

Figura 5 - Exercício 1 (P1-2018)

1. Calcular as reações de apoio através das equações de equilíbrio ($\sum F_x = 0$, $\sum F_y = 0$ e $\sum M_A = 0$)

   Neste caso, definindo o apoio da esquerda como A e o da direita como B, encontra-se que $H_A = 0$ e $V_A = V_B = 76\ kN.cm^2$;

2. Encontrar o valor de N, V e M em Q:

   $N = 100.8\ kN$ (positivo, tracionando a barra);

   $V = -76\ kN$ (negativo, pois gira a barra no sentido anti-horário);

   $M = 76.240\ kN.cm$ (tracionando em baixo da barra);

   Como M traciona o sentido contrário ao indicado pelo eixo z na seção transversal, consideraremos-o como negativo na fórmula;

3. Pela fórmula, $\sigma = \frac{N}{A} + \frac{M}{I_{y0}} z = \frac{100.8}{840} + \frac{(-76.240).(-20)}{1520000} = 0.36\ kN/cm^2$;

4. Calcula-se $M_s^* = y_G . A^* = \frac{100+20}{2} . (100 - 20).3 = 240.60$;

5. Calcula-se, pela fórmula, $\tau = \frac{V.M_s^*}{b^*.I_{y0}} = \frac{-76.240.60}{3.1520000} = -0.24\ kN/cm^2$;

6. Estes valores de tensão representam o plano vertical do triângulo. No plano horizontal $\tau_{yx} = -\tau$ e $\sigma_y = 0$ (propriedade de vigas horizontais);

7. Localizam-se estes pontos no plano (σ, τ) e encontra-se o ponto $P = (\sigma_x, \tau_{yx})$

8. Executam-se os passos 9−17

A seguir, através das funções:

In [4]:
```
N = 100.8
V = -76
M = -76*240
A = 840
Iy0 = 1520000
z = -20
b = 3
Ms = 240*60
alpha = 30
```

In [5]:
```
sigma = get_sigma(N, A, M, Iy0, z)
print("Sigma_x: ", sigma)
tau = get_tau(V, Ms, b, Iy0)
print("Tau_xy: ", tau)
print("O pólo fica no ponto (%f, %f)"%(sigma, -tau))

sigma_x = sigma
tau_xy = tau
sigma_y = 0

radius = get_mohr_radius(sigma_x, sigma_y, tau_xy)
print("Raio do círculo de mohr: ", radius)

sigma_c = get_sigma_c(sigma_x, sigma_y)
print("Sigma_c: ", sigma_c)

sigma_1 = get_sigma_1(sigma_c, radius)
sigma_2 = get_sigma_2(sigma_c, radius)
tau_max = get_tau_max(sigma_c, radius)
tau_min = get_tau_min(sigma_c, radius)
print("Sigma_max: ", sigma_1)
print("Sigma_min: ", sigma_2)
print("Tau_max: ", tau_max)
print("Tau_min: ", tau_min)

sigma_alpha, tau_alpha = get_alpha_point(alpha, sigma_x, sigma_y, tau_xy)
print("Sigma em alpha: ", sigma_alpha)
print("Tau em alpha: ", tau_alpha)

angulo_sigma_1 = get_angle(sigma_x, -tau_xy, sigma_1, 0)
print("Ângulo de sigma_1: ", angulo_sigma_1)
```

```
Sigma_x:  0.36
Tau_xy:  -0.24
O pólo fica no ponto (0.360000, 0.240000)
Raio do círculo de mohr:  0.3
Sigma_c:  0.18
Sigma_max:  0.48
Sigma_min:  -0.12
Tau_max:  0.3
Tau_min:  -0.3
Sigma em alpha:  0.06215390309173477
Tau em alpha:  -0.2758845726811989
Ângulo de sigma_1:  -26.56505117707799
```

In [6]:
```
def get_tensor(sigma_x, sigma_y, sigma_z, tau_xy, tau_xz, tau_yz):
    tensor = np.array([[sigma_x, tau_xy, tau_xz], [tau_xy, sigma_y, tau_yz], [tau
    return tensor

def get_normal_vector(n_x, n_y, n_z):
    n = np.array([n_x, n_y, n_z])
    return n

def get_t_vector(tensor, normal_vector):
    t_vector = tensor.dot(normal_vector)
    return t_vector

def get_sigma_vector(t_vector, normal_vector):
    sigma = np.inner(t_vector, normal_vector)
    sigma_vector = sigma * normal_vector
    return sigma_vector

def get_tau_vector(t_vector, sigma_vector):
    tau_vector = t_vector - sigma_vector
    return tau_vector

def get_principal_stresses(tensor):
    nested_values =  np.linalg.eig(tensor)
    eigen_dic = {}
    for i in range(len(nested_values[0])):
        eigen_dic[nested_values[0][i]] = nested_values[1][:, i]
    return eigen_dic

#modulo_de_x = np.linalg.norm(x)
```

Um exercício de Estado Triplo está muito relacionado a álgebra linear. Pode-se dizer que ele exige menos conhecimento sobre mecânica dos sólidos do que um exercício de Estado Duplo. Ele inicia, normalmente, inicia definindo um paralelepípedo de tensões, como na Figura 6:

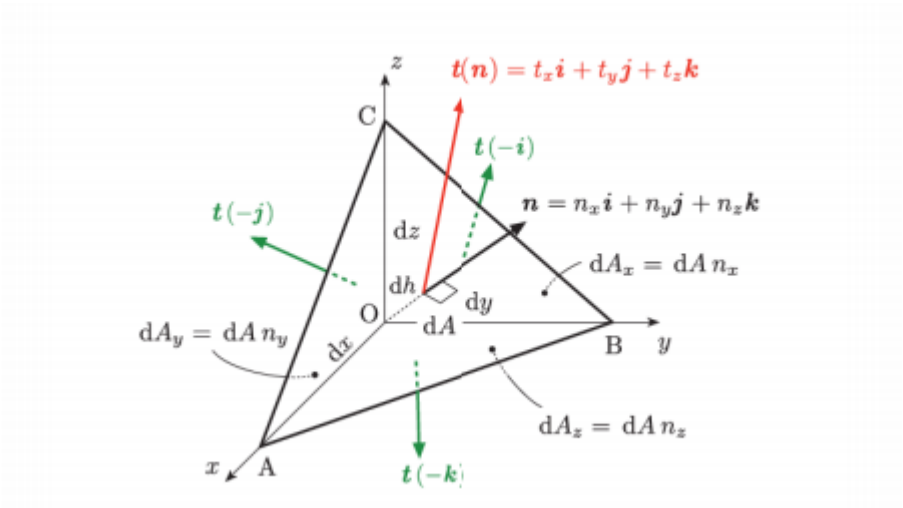

Figura 6 - Paralelepípedo de tensões aberto

1. Encontrar $\sigma_x, \sigma_y, \sigma_z$ que apontam nas direções x, y, z respectivamente. Encontrar

$\tau_{xy}, \tau_{xz}, \tau_{yz}$;

A. τ_{xy} é definido como sendo a tensão com direção a y no plano cuja normal tem direção x;
B. Lembrar de conferir a base que define o sistema coordenado (ex. 2, P1-2018);
    a. A face que precisamos escolher é aquela cuja **normal** tem mesmo sentido que o eixo coordenado;
    b. Se uma das faces do tetraedro estiver invertida (ou seja, se a normal **da face** do tetraedro não concordar com o sentido do eixo coordenado), basta considerar todas as tensões nesta face com seus sinais trocados, pois estas tensões representaram os valores da face oposta (esta sim concordando com o sentido do eixo);

---

2. Encontrar a tensão no plano requisitado a partir do tensor $\underline{\mathbf{T}}$, montado a partir das tensões encontradas, e o vetor $\underline{\mathbf{n}}$, dado do enunciado a partir da relação $\underline{\mathbf{t}} = \underline{\mathbf{T}}.\,\underline{\mathbf{n}}$;

---

3. Encontrar a norma de $\underline{\mathbf{t}}$

---

4. Encontrar a norma de σ através da relação $\sigma = \underline{\mathbf{t}}.\,\underline{\mathbf{n}}$

---

5. Encontrar o vetor $\underline{\sigma} = \sigma.\,\underline{\mathbf{n}} = (\underline{\mathbf{t}}.\,\underline{\mathbf{n}})\underline{\mathbf{n}}$

---

6. Encontrar o vetor tensão tangencial $\underline{\tau} = \underline{\mathbf{t}} - \underline{\sigma}$

---

7. Encontrar a norma de $|\underline{\tau}| = \sqrt{\underline{\tau}.\,\underline{\tau}}$

---

8. Encontrar as direções principais calculando o resultado de

$$\begin{bmatrix} (\sigma_x - \lambda) & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & (\sigma_y - \lambda) & \tau_{yz} \\ \tau_{xy} & \tau_{yz} & (\sigma_z - \lambda) \end{bmatrix} . \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

A. Cada λ solução da equação de determinante nulo define um plano $[h_1, h_2, h_3]$, denominado principal;
B. As três respostas da equação de determinante, correspondem às tensões dos planos planos principais, $\sigma_1, \sigma_2, \sigma_3$ são tais que $\sigma_3 < \sigma_2 < \sigma_1$;
C. Tendo posse de duas tensões principais, digamos n_1 e n_2, é possível encontrar $n_3 = n_1 \times n_2$, pois os versores dos planos principais formam base ortonormal;
D. Planos principais são tais que suas tensões tangenciais são nulas (ou seja, encontram-se no eixo σ no plano (σ, τ));

Exemplo de exercício: Exemplo 10, apostila do Prof. Edgard

Exemplo 10

Determine a máxima tensão de cisalhamento para o paralelepípedo de tensões da figura e indique o plano em que ela ocorre.

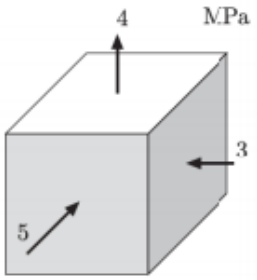

Figura 7 - Exercício de Tensão Tripla

1. Primeiramente, nota-se que $\tau_{xy} = \tau_{xz} = \tau_{yz} = 0$, o que implica matriz diagonal;

2. Não fornecendo o exercício base de referência, adota-se a base canônica xyz positiva, com z indo pra cima, y indo pra direita e x saindo da folha;

3. Calcula-se a matriz tensora $\underline{\mathbf{T}}$:

$$\underline{\mathbf{T}} = \begin{bmatrix} -5 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & 4 \end{bmatrix}$$

4. Calculam-se os auto-valores e auto-vetores do tensor através do método `get-principal-stresses(tensor)`:

$$\begin{aligned} \texttt{eigen-dic} &= \texttt{get-principal-stresses(tensor)} \\ \texttt{eigenvalues} &= \texttt{list(eigen-dic.keys())} \\ \texttt{eigenvectors} &= \texttt{list(eigen-dic.values())} \end{aligned}$$

In [7]:
```
tensor = get_tensor(-5, -3, 4, 0,  0, 0)
eigen_dic = get_principal_stresses(tensor)
eigenvalues = list(eigen_dic.keys())
eigenvectors= list(eigen_dic.values())
print(eigen_dic)
```

```
{-5.0: array([1., 0., 0.]), -3.0: array([0., 1., 0.]), 4.0: array([0., 0.,
1.])}
```

É possível verificar que os autovetores do tensor formam uma base orto-normal, como esperado dos vetores que formam os planos principais. Assim, constrói-se um Círculo de Mohr do Estado Triplo de tensões:

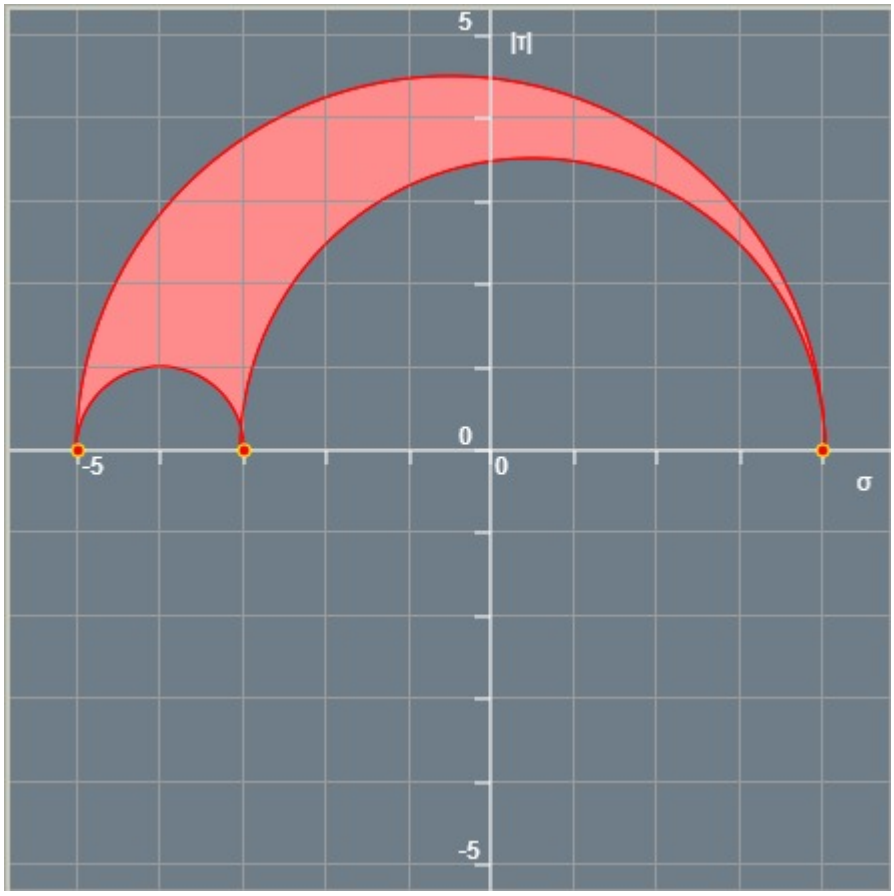

Figura 8 - Círculo de Mohr resultante

A Figura 8 ajuda a ver que o R, raio do círculo vale a diferença das tensões extremas dividido por 2, ou seja, $R = \frac{\sigma_1 - \sigma_3}{2} = \frac{4-(-5)}{2} = 4.5$ e, como os valores possíveis de τ são limitados a área interna do círculo, $\tau_{máx} = R = 4.5$

Resolução - P1 2017

Questão 1)

1ª Questão (3,0 pontos)

Para o estado de tensão representado no paralelepípedo ao lado, determine:

a) as componentes do tensor das tensões na base $(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k})$;

b) as tensões normal e tangencial no plano normal ao vetor $\boldsymbol{n} = \frac{1}{\sqrt{6}}(\boldsymbol{i} + \boldsymbol{j} + 2\boldsymbol{k})$;

c) as tensões e as direções principais (indicar as intersecções dos planos principais com as faces do paralelepípedo da figura).

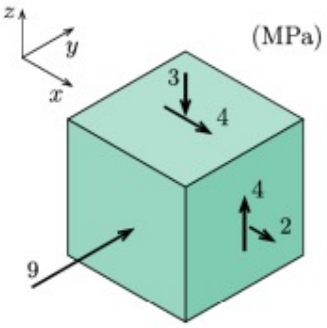

Figura 9 - Questão 1, P1-2017

Passos:

1. Encontrar o tensor $\underline{\mathbf{T}}$;
   A. $\sigma_x = 2, \sigma_y = -9^*, \sigma_z = -3$;

      *Negativo, pois a face que vemos com $\sigma = +9$ é tal que sua normal **NÃO** concordam com o sentido do eixo coordenado
   B. $\tau_{xy} = 0, \tau_{xz} = 4, \tau_{yz} = 0;$

$$\underline{\mathbf{T}} = \begin{bmatrix} 2 & 0 & 4 \\ 0 & -9 & 0 \\ 4 & 0 & -3 \end{bmatrix}$$

---

2. A partir de $\underline{\mathbf{T}}$ e tomando $\underline{\mathbf{n}} = \frac{1}{\sqrt{6}}(1, 1, 2)$, encontrar $\underline{\mathbf{t}} = \underline{\mathbf{T}}.\underline{\mathbf{n}}$;

---

3. Com $\underline{\mathbf{t}}$ e $\underline{\mathbf{n}}$ encontrar σ e com este vetor encontrar τ;

In [8]:
```
tensor = get_tensor(2, -9, -3, 0,  4, 0)
normal_vector = 1/np.sqrt(6) * get_normal_vector(1, 1, 2)
t_vector = get_t_vector(tensor, normal_vector)
sigma_vector = get_sigma_vector(t_vector, normal_vector)
tau_vector = get_tau_vector(t_vector, sigma_vector)
print("Sigma: ", sigma_vector,"   Tau:   ", tau_vector)
```

```
Sigma:  [-0.20412415 -0.20412415 -0.40824829]    Tau:    [ 4.28660705 -3.470110
47 -0.40824829]
```

4. A partir de $\underline{\mathbf{T}}$, encontrar os planos e tensões principais através de seus auto-valores e auto-vetores (na prova será necessário calcular o sistema linear 3×3, use o programa apenas para verificar seus resultados

In [9]:
```
eigen_dic = get_principal_stresses(tensor)
eigenvalues = list(eigen_dic.keys())
eigenvectors= list(eigen_dic.values())
print(eigen_dic)
```

```
{4.216990566028302: array([0.87464248, 0.        , 0.48476853]), -5.21699056602
83025: array([-0.48476853,  0.        ,  0.87464248]), -9.0: array([0., 1.,
0.])}
```

5. Enfim é possível desenhar os planos principais no paralelepípedo

Questão 3)

3ª Questão (3,5 pontos)

O arco da figura abaixo possui seção transversal constante e está submetido a duas forças concentradas. Para o ponto Q da seção transversal c-c, pedem-se:

a) as tensões normal e tangencial que atuam no plano da seção (não deixar de indicar as direções e sentidos das tensões);

b) o círculo de Mohr representando o estado de tensão no ponto;

c) os valores das tensões normais e tangenciais extremas no ponto e as direções dos planos em que ocorrem (indicar os resultados em prismas de tensões).

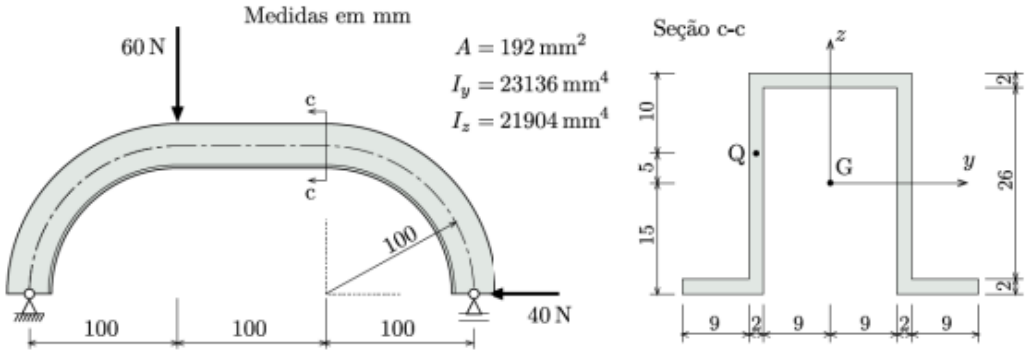

Figura 10 - Questão 3, P1-2017

1. Calcular as reações de apoio nas extremidades;

2. Transferir as forças e momentos para a seção C;

3. Calcular os valores de N, V e M na seção $c-c$ transferindo as forças e momentos de uma das extremidades para a seção, chegando nos resultados $N = -40\ kN$, $V = -20\ kN$ e $M = +2000 kN.mm$ (traciona a parte de cima, e portanto concorda com a direção do eixo z);

4. Calcular $M_s^* = y_G.A^*$:
   A. $y_G = \frac{18.14+2.10.10}{2.A_{vert}+A_{hor}}$
   B. $A^* = 2.A_{vert} + A_{hor}$
   C. $M_s^* = 18.14 + 2.10.10 = 452\ mm^2$

5. Colocar os parâmetros no código abaixo e rodar o programa.

In [10]:
```
N = -40
V = -20
M = +2000
A = 192
Iy0 = 23136
z = 5
b = 2
Ms = 452
sigma = get_sigma(N, A, M, Iy0, z)
print("Sigma_x: ", sigma)
tau = get_tau(V, Ms, b, Iy0)
print("Tau_xy: ", tau)
print("O pólo fica no ponto (%f, %f)"%(sigma, -tau))

sigma_x = sigma
tau_xy = tau
sigma_y = 0

radius = get_mohr_radius(sigma_x, sigma_y, tau_xy)
print("Raio do círculo de mohr: ", radius)

sigma_c = get_sigma_c(sigma_x, sigma_y)
print("Sigma_c: ", sigma_c)

sigma_1 = get_sigma_1(sigma_c, radius)
sigma_2 = get_sigma_2(sigma_c, radius)
tau_max = get_tau_max(sigma_c, radius)
tau_min = get_tau_min(sigma_c, radius)
print("Sigma_max: ", sigma_1)
print("Sigma_min: ", sigma_2)
print("Tau_max: ", tau_max)
print("Tau_min: ", tau_min)

alpha = 30
sigma_alpha, tau_alpha = get_alpha_point(alpha, sigma_x, sigma_y, tau_xy)
print("Sigma em alpha: ", sigma_alpha)
print("Tau em alpha: ", tau_alpha)

angulo_sigma_1 = get_angle(sigma_x, -tau_xy, sigma_1, 0)
print("Ângulo de sigma_1: ", angulo_sigma_1)
```

```
Sigma_x:  0.22389349930843708
Tau_xy:  -0.19536652835408022
O pólo fica no ponto (0.223893, 0.195367)
Raio do círculo de mohr:  0.22516694952694527
Sigma_c:  0.11194674965421854
Sigma_max:  0.3371136991811638
Sigma_min:  -0.11322019987272673
Tau_max:  0.22516694952694527
Tau_min:  -0.22516694952694527
Sigma em alpha:  -0.0012722521224784733
Tau em alpha:  -0.1946319932486902
Ângulo de sigma_1:  -30.093454736977737
```

Resolução P1 2020

A P1 de 2020 foi feita durante época de coronavírus, dada pelo Prof. Britto (diferente das provas de 2019 ou anteriores, que foram elaboradas pelo Prof. Edgard) e estava consideravelmente mais fácil. Segue uma resolução da mesma:

Exercício 1

Questão 1 (*4 pontos*): Para o ponto C da viga abaixo, situado na seção imediatamente à direita da carga aplicada, determinar:

a) as tensões normal e tangencial (em $\mathrm{kgf/cm^2}$), no plano da seção transversal

b) o círculo de *Mohr*, determinando o polo, as tensões principais e a tensão tangencial máxima.

A seção é de parede fina, com espessura constante e igual a 3 cm. O ponto C é o ponto médio do lado inferior esquerdo do losango. Suponha que o observador enxerga o elemento infinitesimal, que representa o ponto C, numa vista lateral da viga.

São dados os *momentos de inércia* da seção: $\mathrm{I_y = 320\,000\ cm^4}$ e $\mathrm{I_z = 180\,000\ cm^4}$.

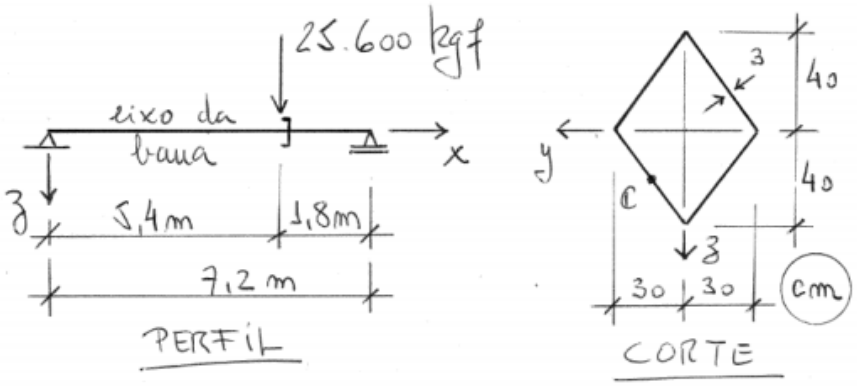

Figura 11 - Questão 1, P1-2020

1. Calcular as reações de apoio em A (esquerda) e B (direita):
   A. $H_A = 0\ kgf$;
   B. $V_A = 6\ 400\ kgf$;
   C. $V_B = 19\ 200\ kgf$;

2. Calcular N, V e M na seção requisitada:
   A. $N = 0\ kgf$;
   B. $V = -19\ 200\ kgf$;
   C. $M = 3\ 456\ 000\ kgf.cm$ (tracionando em baixo e positivo, pois z aponta pra baixo);

3. Calcular $\sigma = \cancel{\frac{N}{A}}^{0} + \frac{M}{I_{y0}} z = 216\ kgf/cm^2$

4. Calcular $M_s^* = \underbrace{30}_{y_G}.\underbrace{25.3}_{A^*} = 2250cm^2$

5. Calcular $\tau = \frac{V.M_s^*}{b^*.I_{y0}} = -45\ kgf/cm^2$

6. Calcular os dados do círculo:

In [11]:
```
N = 0
V = -19200
M = 3456000
A = 1 #pode ser qualquer valor pois a normal é 0
Iy0 = 320000
z = 20
b = 3
Ms = 2250
sigma = get_sigma(N, A, M, Iy0, z)
print("Sigma_x: ", sigma)
tau = get_tau(V, Ms, b, Iy0)
print("Tau_xy: ", tau)
print("O pólo fica no ponto (%f, %f)"%(sigma, -tau))

sigma_x = sigma
tau_xy = tau
sigma_y = 0

radius = get_mohr_radius(sigma_x, sigma_y, tau_xy)
print("Raio do círculo de mohr: ", radius)

sigma_c = get_sigma_c(sigma_x, sigma_y)
print("Sigma_c: ", sigma_c)

sigma_1 = get_sigma_1(sigma_c, radius)
sigma_2 = get_sigma_2(sigma_c, radius)
tau_max = get_tau_max(sigma_c, radius)
tau_min = get_tau_min(sigma_c, radius)
print("Sigma_max: ", sigma_1)
print("Sigma_min: ", sigma_2)
print("Tau_max: ", tau_max)
print("Tau_min: ", tau_min)
```

```
Sigma_x:  216.0
Tau_xy:  -45.0
O pólo fica no ponto (216.000000, 45.000000)
Raio do círculo de mohr:  117.0
Sigma_c:  108.0
Sigma_max:  225.0
Sigma_min:  -9.0
Tau_max:  117.0
Tau_min:  -117.0
```

7. Desenhar o círculo a partir do pontos extremos/sigma_c.

Exercício 2

Questão 2 (*3 pontos*): Para o prisma triangular infinitesimal, em estado duplo de tensão, mostrado na figura da esquerda (em que as tensões são dadas em $\mathrm{kN/cm^2}$), determinar, escrevendo as equações de equilíbrio, as tensões τ', τ'' e τ'''. As dimensões são dadas, em centímetros, na figura da direita. Sabe-se ainda que a espessura do prisma é *unitária* ($\mathrm{e} = 1$ cm).

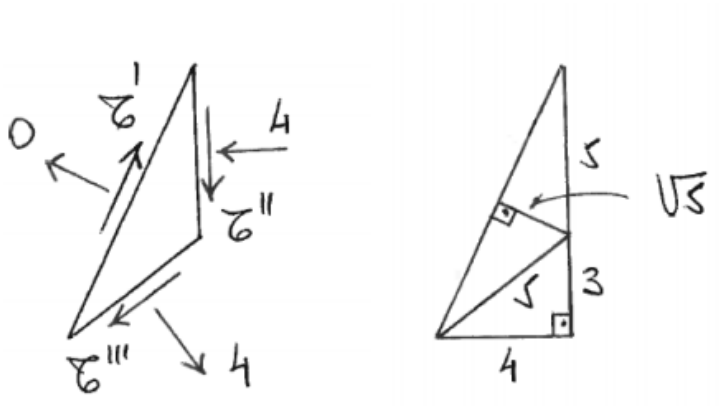

Figura 11 - Questão 2, P1-2020

1. Definir os ângulos α e β no triângulo da direita, conforme mostra a Figura 12:

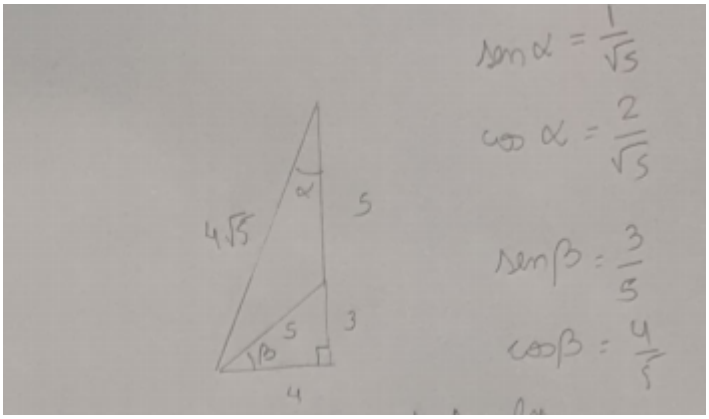

Figura 12 - Triângulo da direita com os ângulos

2. Tomando a medida do maior lado do triângulo da esquerda como ds, escrever as medidas dos outros dois lados em função deste ds:
   A. Lado vertical: $\frac{\sqrt{5}}{4}ds$
   B. Lado oblíquo: $\frac{\sqrt{5}}{4}ds$

3. Calcular o somatório das forças e momentos e igualá-las a 0:
   A. $\sum F_x = 0 \Rightarrow \cancel{e}.\cancel{ds}\,(\tau' \sin\alpha - \frac{\sqrt{5}}{4}4 - \frac{\sqrt{5}}{4}\tau''' \cos\beta + \frac{\sqrt{5}}{4}4\sin\beta) = 0$

B. $\sum F_y = 0 \Rightarrow \cancel{\rho}.\cancel{ds}\,(\tau' \cos\alpha - \frac{\sqrt{5}}{4}\tau'' - \frac{\sqrt{5}}{4}\tau''' \sin\beta + \frac{\sqrt{5}}{4}4\cos\beta) = 0$

C. Sendo C definido pelo vértice onde τ'' e τ''' se encontram,

$\sum M_C = 0 \Rightarrow \cancel{\rho}.\cancel{ds}\,(\frac{\sqrt{5}}{4}4.\frac{5}{2} + \frac{\sqrt{5}}{4}4.\frac{5}{2} - \sqrt{5}\tau') = 0$

---

4. Resolver este sistema linear (começando por C) e encontrar que
$\tau' = 5\ kN/cm^2, \tau'' = 3\ kN/cm^2, \tau''' = 3\ kN/cm^2$

Exercício 3

Questão 3 (*3 pontos*): Para um certo ponto M de uma estrutura é dada a matriz do tensor das tensões, numa base ortonormal $(\vec{i}, \vec{j}, \vec{k})$:

$$[T] = \begin{bmatrix} 2 & -2 & -1 \\ -2 & 2 & 1 \\ -1 & 1 & 5 \end{bmatrix} \qquad (kN/cm^2)$$

Determinar:

a) O vetor tensão $\vec{\rho} = X\vec{i} + Y\vec{j} + Z\vec{k}$ no plano definido pela normal externa:

$$\vec{n} = \frac{2}{3}\vec{i} + \frac{2}{3}\vec{j} - \frac{1}{3}\vec{k}$$

b) A tensão normal escalar σ e o módulo da tensão tangencial $|\vec{\tau}|$ no plano do ítem a)

c) As tensões principais (σ_1, σ_2 e σ_3) no ponto M, bem como a direção principal $\vec{h}_1$, associada a σ_1

Figura 13 - Questão 3, P1-2020

1. A partir do tensor $\underline{\mathbf{T}}$, aplicar $\underline{\rho} = \underline{\mathbf{T}}.\underline{\mathbf{n}}$

2. Calcular $\sigma = \underline{\rho}.\underline{\mathbf{n}}$, módulo da tensão normal

3. Calcular $\underline{\sigma} = \sigma.\underline{\mathbf{n}}$

4. Encontrar $\underline{\tau} = \sigma.\underline{\mathbf{n}} - \underline{\sigma}$

5. Encontrar o módulo de $\underline{\tau}$, $\tau = \sqrt{\underline{\tau}^{\mathbf{T}}.\underline{\tau}}$

6. Encontrar os autovalores do tensor

7. Selecionar o maior autovalor ($\sigma_1 = 6\ kN/cm^2$) e calcular sua direção principal $h_1 = \frac{1}{\sqrt{6}}[-1, 1, 2]$

In [12]:
```
tensor = get_tensor(2, 2, 5, -2,  -1, 1)
normal_vector = 1/3 * get_normal_vector(2, 2, -1)
t_vector = get_t_vector(tensor, normal_vector)
sigma_vector = get_sigma_vector(t_vector, normal_vector)
tau_vector = get_tau_vector(t_vector, sigma_vector)
print("Vetor rho:  ", t_vector)
print("Sigma: ", sigma_vector,"\nTau:   ", tau_vector)
eigen_dic = get_principal_stresses(tensor)
eigenvalues = list(eigen_dic.keys())
eigenvectors= list(eigen_dic.values())
print(eigen_dic)
```

```
Vetor rho:   [ 0.33333333 -0.33333333 -1.66666667]
Sigma:  [ 0.37037037  0.37037037 -0.18518519]
Tau:    [-0.03703704 -0.7037037  -1.48148148]
{0.0: array([ 7.07106781e-01,  7.07106781e-01, -1.06685494e-16]), 2.99999999999
9999: array([ 0.57735027, -0.57735027,  0.57735027]), 6.000000000000002: array
([-0.40824829,  0.40824829,  0.81649658])}
```