UNIVERSIDADE ESTADUAL DO MARANHÃO - UEMA

CENTRO DE CIÊNCIAS TECNOLÓGICAS – CCT CURSO DE ENGENHARIA MECÂNICA

Alan David Ferreira e Ferreira

Bruno Brito Santos

Helder Felipe Santana Nogueira

Joellisson Ribeiro Madeira

ANÁLISE DE PROPRIEDADES MECÂNICAS DE MATERIAS

São Luís

UNIVERSIDADE ESTADUAL DO MARANHÃO

2024

Alan David Ferreira e Ferreira

Bruno Brito Santos

Helder Felipe Santana Nogueira

Joellisson Ribeiro Madeira

ANÁLISE DE PROPRIEDADES MECÂNICAS DE MATERIAS

Desenvolvimento de um programa em Python para calcular e comparar propriedades mecânicas Básicas, como módulo de elasticidade e limite de escoamento, usando dados de tensão-Deformação. Com o intuito de simular testes de tração e representar graficamente as Curvas de tensão-deformação.

São Luís
2024

Lista de Ilustrações

Figure 1: Diagrama tensão–deformação típico.....5

Figure 2: Diagrama depois de atingido o limite elástico.....6

Figure 3: Diagrama modulo de elasticidade.....7

Figure 4: gráfico plotagem.....17

SUMÁRIO

1 INTRODUÇÃO.....5

2 REFERENCIAL TEÓRICO.....6

2.1 FERRAMENTAS COMPUTACIONAIS PARA ANÁLISE DE DADOS.....9

2.1.1 Pandas.....9

2.1.2 Matplotlib.pyplot.....9

2.1.3 NumPy.....9

3 DESENVOLVIMENTO DO CÓDIGO.....10

3.1 Importação das bibliotecas.....10

3.2 Modularização (Criação do módulo).....10

3.3 Leitura dos dados do corpo de prova.....12

3.4 Dimensionar o corpo de prova/calcular a tensão e a deformação.....14

3.5 Calcular os limites de resistência a tração e de elasticidade.....14

3.6 Criação das Regiões (classe).....15

3.7 Plotagem (Criação do Gráfico).....16

4 CONCLUSÃO.....18

REFERÊNCIAS.....19

1 INTRODUÇÃO

A precisão na análise das propriedades mecânicas contribui na melhoria da segurança e da durabilidade dos produtos finais. Essa prática possibilita a identificação de materiais adequados para cada aplicação específica, garantindo que os produtos possam suportar as condições de uso para as quais foram projetados. Além disso, a redução do desperdício de matéria-prima e a minimização dos custos de produção são benefícios adicionais que reforçam a importância desse tipo de análise no processo de desenvolvimento de produtos.

O seguinte trabalho aborda um estudo sobre as propriedades mecânicas dos materiais com o intuito de verificar a resistência para suportar os testes experimentais, no qual foi realizado uma análise computacional utilizando o programa Python para estudar o comportamento do corpo de prova cilíndrico de Alumínio com diâmetro de 12,80 mm e comprimento de 50,80 mm submetido em diferentes condições como tensão, deformação, elasticidade e limite de escoamento. Visto que, prever esses aspectos é crucial para entender as etapas necessárias no processamento industrial, pois garanti a integridade e eficiência de produtos e estruturas na engenharia. A importância deste estudo reside na capacidade prever o desempenho dos materiais sob diferentes ações, o que é fundamental para uma ótima otimização.

Portanto, o presente estudo tem como objetivo conhecer a resistência e deformabilidade dos componentes, de maneira que seja possível projetar produtos mais robustos, reduzir o desperdício de matéria prima e minimizar os custos de produção. De modo, que analisar de forma precisa usando linguagem de programação gerando gráficos pode-se obter uma ampliação sobre o funcionamento das propriedades mecânicas, o que ajuda diretamente no dimensionamento de materiais para atender o fluxo de demanda na fabricação dos componentes.

2 REFERENCIAL TEÓRICO

Tensão x Deformação

A resistência à tração refere-se à capacidade de um material de resistir a forças que tendem a alongá-lo. Este parâmetro é medido durante um ensaio de tração, onde uma amostra é esticada até a ruptura. De maneira, que se pode através do gráfico de diagrama de tensão e deformação analisar o comportamento do material, como mostra a figura:

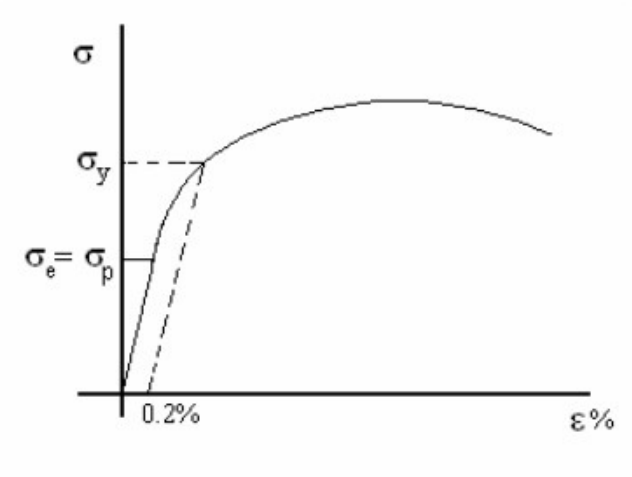

Figure 1: Diagrama tensão–deformação típico

Para a maioria dos materiais, o diagrama acima apresentado pode ser dividido em duas partes: na primeira que vai de zero até uma tensão σp, a tensão é linearmente proporcional à deformação e a partir de então esta proporcionalidade deixa de existir. Por causa disto, a tensão σp é chamada e limite de proporcionalidade e dentro da faixa p 0 − σ tem-se:

$$\sigma = E\varepsilon$$

Onde,

E – é o módulo de elasticidade do material e,

ε – é a deformação na direção de σ.

Esta relação é a largamente conhecida lei de Hooke para estado uniaxial de tensões. Observa-se também que para a maioria dos materiais, depois de excedido o limite de proporcionalidade e descarregado o material, ocorre deformação permanente. Assim, percebe-se que o limite elástico, ou seja, a máxima tensão que pode ser aplicada sem que ocorra deformação plástica é praticamente igual ao limite de proporcionalidade, de forma que $\sigma p \approx \sigma e$.

Limite de Elasticidade (σy): (E) = Tensão (σ) / Deformação (ε).

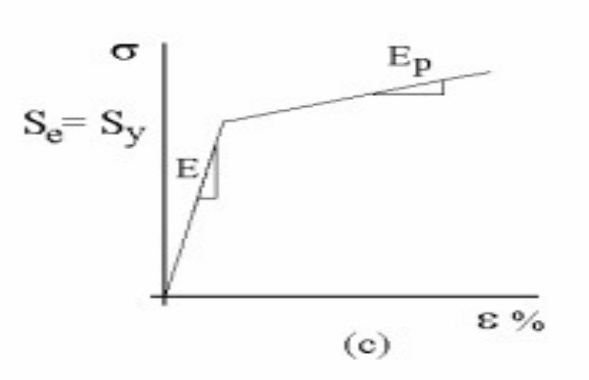

Figure 2: Diagrama depois de atingido o limite elástico

Nessa fórmula, a tensão é a carga aplicada ao corpo de prova dividida pela área transversal original, e a deformação é a variação do comprimento do corpo de prova dividida pelo comprimento original.

Resistência à Tração Máxima (σmax):

A resistência à tração máxima, também conhecida como tensão de ruptura (σmax), é a tensão máxima que um material pode suportar antes de falhar ou romper. Este parâmetro é obtido a partir de ensaios de tração e é fundamental para determinar a robustez de um material em aplicações estruturais. A resistência à tração máxima é crucial para garantir a integridade estrutural e a segurança de componentes submetidos a tensões elevadas.

Tenacidade

A tenacidade é a capacidade de um material de absorver energia e deformar-se sem fraturar. Este parâmetro é representado pela área sob a curva tensão-deformação em um ensaio de tração. Materiais com alta tenacidade são preferidos em aplicações onde a resistência ao impacto é crítica.

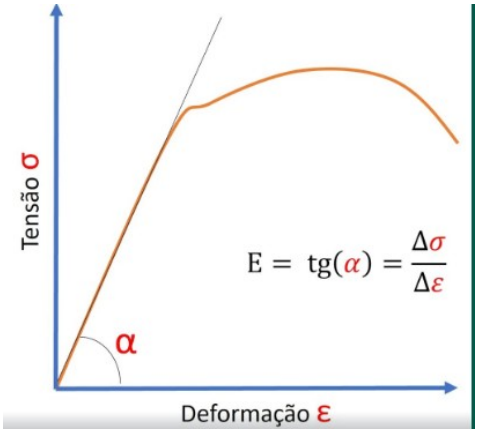

Figure 3: Diagrama modulo de elasticidade

Módulo de Elasticidade (Módulo de Young)

O módulo de elasticidade, ou módulo de Young, é uma medida da rigidez de um material. É definido como a razão entre a tensão e a deformação na região elástica (linha reta) da curva tensão-deformação. Este parâmetro é crucial para prever a deformação de um material sob uma carga específica. Módulo de Elasticidade (E) = Tensão (σ) / Deformação (ε).

Métodos de Ensaios

Os métodos de ensaio são fundamentais para a determinação das propriedades mecânicas dos materiais. Alguns dos principais métodos incluem:

- **Ensaios de Tração**: Para medir resistência à tração, módulo de elasticidade, limite de elasticidade e alongamento.

Aplicação Prática: Alumínio

O alumínio é um metal leve e versátil, amplamente utilizado em diversas indústrias. Suas principais características incluem:

- **Leveza**: Com uma densidade de aproximadamente 2.7 g/cm³, o alumínio é ideal para aplicações onde o peso é uma consideração importante, como na indústria aeroespacial e automotiva.

- **Resistência à Corrosão**: Forma uma camada de óxido na superfície que protege o material da corrosão, tornando-o adequado para uso em ambientes agressivos.
- **Ductilidade e Maleabilidade**: O alumínio pode ser facilmente trabalhado e moldado, permitindo a fabricação de componentes com formas complexas.

No experimento específico mencionado, a amostra de alumínio tinha um diâmetro inicial de 12.18 mm e um comprimento inicial de 50,8 mm, ilustrando a importância da precisão nas medições iniciais para garantir a reprodutibilidade dos resultados.

2.1 FERRAMENTAS COMPUTACIONAIS PARA ANÁLISE DE DADOS

Para esse estudo e ampliação do código foi usado as devidas bibliotecas do Python, que são algoritmos, no qual servem para facilitar as aplicações de comando de dados. No seguinte trabalho foi utilizado as seguintes bibliotecas:

2.1.1 Pandas

A biblioteca Pandas é uma das ferramentas mais populares para manipulação e análise de dados em Python. Ela fornece estruturas de dados rápidas, flexíveis e expressivas, como Series e DataFrame, facilitando o trabalho com dados tabulares.

2.1.2 Matplotlib.pyplot

Matplotlib.pyplot é um módulo da biblioteca Matplotlib que oferece uma interface estilo MATLAB para criação de gráficos e visualizações. É amplamente utilizado para plotar dados de forma visual, permitindo a personalização e criação de gráficos simples ou complexos.

2.1.3 NumPy

NumPy é fundamental para computação científica em Python, oferecendo suporte para arrays e matrizes multidimensionais, além de uma vasta coleção de funções matemáticas de alto nível. NumPy é essencial para a manipulação eficiente de grandes volumes de dados numéricos.

3 DESENVOLVIMENTO DO CÓDIGO

3.1 Importação das bibliotecas

O primeiro passo para a realização da criação do gráfico para análise das propriedades mecânicas é a importação das bibliotecas que serão uteis para leitura da tabela, plotagem de gráficos e manipulação de matrizes, a saber pandas, matplotlib e numpy, respectivamente, em que serão chamadas como variáveis simplificadas: *pd*, *plt*, *np*.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

3.2 Modularização (Criação do módulo)

Também será importado um módulo que foi criado contida a função *plotMosaico* para servi como subalgoritmo, reduzindo o código principal. O módulo está condito no diretório *plotagem* e está organizado da seguinte forma:

```
from plotagem import plotagem
```

```
!tree
├── codigoAvaliacao.py
├── dadosEnsaioTracao.xlsx
├── [figuras
│   ├── Figure_1.png
│   ├── Figure_2.png
│   ├── graficoVersaoRevisar.png
│   └── melhoraNoGrafico.png
├── plotagem
│   ├── __init__.py
│   ├── plotagem.py
│   └── [__pycache__
│       ├── __init__.cpython-312.pyc
│       └── plotagem.cpython-312.pyc
```

```
├── [__pycache__
│   └── codigoAvaliacao.cpython-312.pyc
├── README.md
└── RelatorioCodigo.ipynb

5 directories, 13 files
```

O conteúdo da função está exposto a baixo e consiste em plotar as diferentes regiões que estaram contidos numa lista em formato de objetos e são inseridas pelo parâmetro *regioes*. Outro parâmetro importante será o *textPlot* em que receberá um booleano que define se será adicionado uma indicação do ângulo e do modulo de elasticidade na plotagem da região. Vê-se que compete saber essas informações só na região elástica, logo será indicado como *True* quando for feito a chamada da função para ela.

A plotagem será feita em formato moisaco do matplotlib, por essa razão foi adicionado o parâmetro *pos*, que defini a posição do objeto axes, o qual também é um parâmetro no qual diz o plot que eu estou me referindo. As posições dos plots na figura são definidas na criação da variável *axs* que recebe o plot e são *upleft*, *upright*, *lowleft*, *lowright*, traduzindo, superior esquerdo, superior esquerdo, inferior esquerdo e inferior direito, respectivamente. Os demais parâmetros são para as funções de ajuste e plotagem do matplotlib.

```
def plotMosaico(fig, axs, title, labelx, labely, pos, regioes,
enquadro, posText=(0,0), legendaLoc="",textPlot=bool):
```

Esse laço abaixo é feito para navegar em cada região da lista *regioes*, em que será plotado o gráfico com base nas propriedades de cada objeto, como posição x, posição y, cor, label. Em seguida será plotado com a função *scatter* do matplotlib o ponto final de cada gráfico, e o par ordenado será composto pela última posição da propriedade x e y do objeto, que serão adquiridas por *[-1]*. Nesse ponto estará localizado os limites, por exemplo, a região elástica vai ter o limite de escoamento, a região plástica vai ter o limite de resistência a tração e a região da ruptura vai ter o limite que será a ruptura. Nesses limites será posto uma seta junto com a *string* que indentificadará cada um, para tanto será utilizado o parâmetro *posText* que somará com o par ordenado definindo a posição do texto e da seta.

```
for reg in regioes:
 axs[pos].plot(reg.x, reg.y, color=reg.cor, label=reg.label)
 axs[pos].scatter(reg.x[-1], reg.y[-1], s=20, facecolor='C0',
edgecolor='k')


axs[pos].annotate(reg.limite+f' (${reg.x[-1]:.3f}, ${reg.y[-
1]:.4f})', xy=(reg.x[-1], reg.y[-1]), xytext=(reg.x[-1] +
posText[0], reg.y[-1] + posText[1]),
                                arrowprops=dict(facecolor='black',
shrink=0.1))
```

Em segundo plano, tem-se a utilização do *textPlot* para definir a região e o gráfico que ficará indicado o ângulo e o texto contido o **módulo de elasticidade**. Essa indicação terá como referência o início do gráfico.

```
if textPlot:
        axs[pos].annotate(f'{reg.anguloGraus:.4f}°', xy=(reg.x[0],
reg.y[0]), xytext=(reg.x[0]+0.0001, reg.y[0]+0.001),
                          arrowprops=dict(facecolor='red', shrink=0))
        axs[pos].text(reg.x[0]+0.001, reg.y[0]+0.01, f'$\
epsilon={reg.variacao:.4f}$',fontsize=15)
```

Por fim, será chamado várias funções padrões do matplotlib para criar uma grade no gráfico, nomear os eixos, entitular o gráfico, definir os limites dos eixos e a localização da legenda.

```
axs[pos].grid(False)
axs[pos].set_xlabel(labelx)
axs[pos].set_ylabel(labely)
axs[pos].set_title(title)
axs[pos].axis(enquadro)
axs[pos].legend(loc=legendaLoc)
```

3.3 Leitura dos dados do corpo de prova

Em seguida será lido o arquivo .xlsx, no qual contém a tabela. Também será convertido essa tabela para um *array do numpy* para auxiliar na manipulação. Cada coluna será lida e recibida numa nos seus respectivos vetores que compreende cada uma delas.

```
# Recebendo os dados do corpo de prova
dados = pd.read_excel("dadosEnsaioTracao.xlsx").to_numpy()

print(dados)
```

```
[[ 0.    50.8  ]
 [ 7.33  50.851]
 [15.1   50.902]
 [23.1   50.952]
 [30.4   51.003]
 [34.4   51.054]
 [38.4   51.308]
 [41.3   51.816]
 [44.8   52.832]
 [46.2   53.848]
 [47.3   54.864]
 [47.5   55.88 ]
 [46.1   56.896]
 [44.8   57.658]
 [42.6   58.42 ]
 [36.4   59.182]]
```

```
Forca = dados[:,0] # Em Newtons (Primeira Coluna)
comprimentoFinal = dados[:,1] # Em Metro (Segunda Coluna)
```

7.29 A cylindrical specimen of aluminum having a diameter of 0.505 in. (12.8 mm) and a gauge length of 2.000 in. (50.800 mm) is pulled in tension. Use the load–elongation characteristics shown in the following table to complete parts (a) through (f).

Load		*Length*	
N	***lb$_f$***	***mm***	***in.***
0	0	50.800	2.000
7,330	1,650	50.851	2.002
15,100	3,400	50.902	2.004
23,100	5,200	50.952	2.006
30,400	6,850	51.003	2.008
34,400	7,750	51.054	2.010
38,400	8,650	51.308	2.020
41,300	9,300	51.816	2.040
44,800	10,100	52.832	2.080
46,200	10,400	53.848	2.120
47,300	10,650	54.864	2.160
47,500	10,700	55.880	2.200
46,100	10,400	56.896	2.240
44,800	10,100	57.658	2.270
42,600	9,600	58.420	2.300
36,400	8,200	59.182	2.330
Fracture			

(dados retirados do livro de Callister, W.D. e Rethwisch, D.G Fundamentals of Materials Science and Engineering, 4th Ed.)

3.4 Dimensionar o corpo de prova/calcular a tensão e a deformação

As medidas iniciais do corpo de prova foram tiradas da questão do *Callister* e para os calculos de área, força e deformação serão utilizados a seguintes fórmulas:

$$Area = \frac{\pi \cdot Do^2}{4} \qquad \sigma = \frac{F}{Area} \qquad \varepsilon = \frac{\Delta l}{lo}$$

```
# Definindo as dimenções inicias do corpo de prova. OBS: tudo em
milímetro
comprimentoInicial = 50.8
diametroInicial = 12.8
area = (np.pi*diametroInicial**2.0)/4


Tensao = Forca / area
e = (comprimentoFinal-comprimentoInicial)/comprimentoInicial
```

3.5 Calcular os limites de resistência a tração e de elasticidade

Primeiramente, tem-se que por definição que o limite de resistência a tração é o ponto máximo do gráfico de tensão-deformação logo para acha-ló será utilizado a função *max*.

```
# Cálculo do limite de resistência a tração
limRT = max(Tensao)
```

Em segundo lugar, para o limite de elasticidade se sabe que é o ponto em que os anteriores tem a mesma taxa de variação, ou seja, o ponto seguinte do gráfico terá um afastamento inicial da linearidade da curva, logo para saber esse ponto é definido um ângulo inicial e foi verificado qual ponto que será mudado esse angulo. Assim foi verificado o limite de elasticidade.

```
#Cálculo do limite de elasticidade
tamanho = Tensao.size
anguloInicial = round(np.tan(e[1] / Tensao[1]),8)
limElast = 0

for id in range(2,tamanho):
 angulo = round(np.tan(e[id]/Tensao[id]),8)
 if angulo != anguloInicial:
 limElast = Tensao[id-1]
 break
```

3.6 Criação das Regiões (classe)

Como cada região tem sua peculiaridade que serão expressas no gráfico, será criado um classe para desenvolver um objeto de cada região com sua respectivas propriedades, como inicio, fim e cor, por exemplo. Também aqui será caculado o *módulo de elasticidade* , quando, na *class*, ser definido a variação, pois o módulo de elasticidade é a taxa de variação da região elástica, que é dado pela fórmula abaixo.

$$E = \frac{\Delta \sigma}{\Delta \epsilon}$$

Já que a região elástica é uma função afim, pode-se determinar o ângulo entre a reta e o eixo horizontal pelo arco tangente dessa mesma variação.

$$\theta = \arctan(E)$$

```
#definindo a posição dos limites e as regiões
class Regiao:
 def __init__(self, inicio, fim, x, y, label, limite, cor):
 self.inicio = np.where(y == inicio)[0][0]
 self.fim = np.where(y == fim)[0][0] + 1
 self.x = x[self.inicio:self.fim]
 self.y = y[self.inicio:self.fim]
 self.limite = limite
 self.label = label
 self.cor = cor
 x0, y0 = self.x[0],self.y[0]
 x1, y1 = self.x[-1],self.y[-1]
 deltaX = x1-x0
 deltaY = y1-y0
 self.variacao = (deltaY)/(deltaX)
 self.angulo = np.arctan2(self.y,self.x)
 self.anguloGraus = (angulo*180)/np.pi


regElastica = Regiao(0, limElast, e, Tensao, "Região Elástica",
"Limite de Escoamento", "orange")
regPlastica = Regiao(limElast, limRT, e, Tensao, "Região Plástica",
"Limite de Resistência a Tração", "blue")
regRuptura = Regiao(limRT, Tensao[-1], e, Tensao, "Região da
Ruptura", "Ruptura", "red")
```

3.7 Plotagem (Criação do Gráfico)

Aqui será plotado o gráfico utilizando a função do *matplotlib*: *subplot_mosaic*, que serve para plotar múltiplos gráficos em uma figura, definindo sua posição, que é dita usando *upleft*,*uprigh*,*lowlef*,*lowright*. E para configurar o gráfico, definindo as limites dos eixos, as distâncias do texto para a curva, foi criado tuplas, que representam vetores nessa funcionalidade. Em seguida, com todas a variáveis e objetos criados foi chamado o módulo *plotagem* que contem a função que fará os ajustes para cada região e plot: *plotMosaico*. Por fim, será plotado a figura com todos os ajustes e gráficos nas suas posições definidas.

```
#Plotagem
fig, axs = plt.subplot_mosaic([['upleft', 'upright'],
 ['lowleft', 'lowright']], figsize=(15,10), gridspec_kw={'hspace':
0.5})

limInicial_plot0 = (0,0)
limFinal_plot0 = (max(e)+0.1, max(Tensao)+0.1)

limInicial_plot1 = (0,0)
limFinal_plot1 = (regPlastica.x[4], regPlastica.y[4])

limInicial_plot2 = (regPlastica.x[6], regPlastica.y[6])
limFinal_plot2 = (regRuptura.x[1]+0.05, max(regRuptura.y)+0.01)

limInicial_plot3 = (0,0)
limFinal_plot3 = (max(regElastica.x)+0.001, max(regElastica.y)+0.01)

enquadroPlot0 = [limInicial_plot0[0], limFinal_plot0[0],
limInicial_plot0[1], limFinal_plot0[1]]
enquadroPlot1 = [limInicial_plot1[0], limFinal_plot1[0],
limInicial_plot1[1], limFinal_plot1[1]]
enquadroPlot2 = [limInicial_plot2[0], limFinal_plot2[0],
limInicial_plot2[1], limFinal_plot2[1]]
enquadroPlot3 = [limInicial_plot3[0], limFinal_plot3[0],
limInicial_plot3[1], limFinal_plot3[1]]

distanciaTextPlot0 = (0.04,0.02)
distanciaTextPlot1 = (0.001,0)
distanciaTextPlot2= (-0.02,-0.02)
distanciaTextPlot3 = (0.0001,-0.02)

lengedaLocPlot0 = "lower right"
lengedaLocPlot1 = "lower right"
lengedaLocPlot2 = "upper right"
```

```
lengedaLocPlot3 = "upper right"


plotagem.plotMosaico(fig, axs, "Tensão X Deformação",  "ε deformação
[%]", "σ tensão [Pa]", "upleft",
 [regElastica, regPlastica, regRuptura], enquadroPlot0,
distanciaTextPlot0, lengedaLocPlot0,False)
plotagem.plotMosaico(fig, axs, "Limite de Escoamento",  "", "",
"upright",
 [regElastica, regPlastica], enquadroPlot1, distanciaTextPlot1,
lengedaLocPlot1, False)
plotagem.plotMosaico(fig, axs, "Limite de Resistência a Tração", "",
"", "lowright",
 [regPlastica, regRuptura], enquadroPlot2, distanciaTextPlot2,
lengedaLocPlot2, False)
plotagem.plotMosaico(fig, axs, "Módulo de Elasticidade",  "", "",
"lowleft",
 [regElastica], enquadroPlot3, distanciaTextPlot3,
lengedaLocPlot3,True)


plt.tight_layout
plt.show()
```

Para cada chamada da função do *plotMosaico* será plotado um gráfico em posições diferentes.

4 CONCLUSÃO

Verifica-se que é possível fazer uma análise clara das regiões para compreender o comportamento do material quando foi submetido sobre o ensaio de tração, levando a saber suas propriedades como ductibilidade, dureza, resistência e deformabilidade. Pela coloração da região, sabe-se o espaço de permancencia do corpo numa determinada região. sobre certa tensão. É possível observar os pontos dos limites de escoamento e de resistência a tração, sendo assim garante a saber a segurança e a qualidade do material. Sobre esse material em específico, nota-se que é dúctil e tenaz, pois deforma bastante plasticamente antes de fraturar.

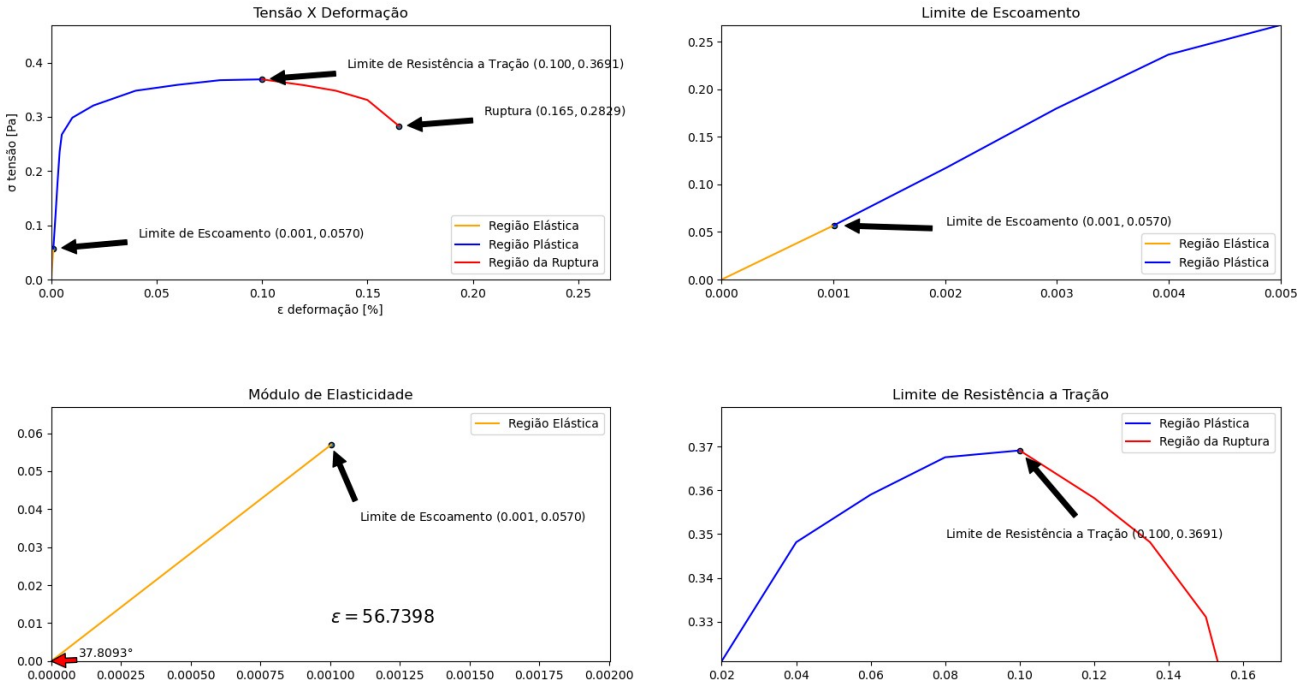

Figure 4: gráfico plotagem

REFERÊNCIAS

Callister, W. D., & Rethwisch, D. G. (2014). Ciência e Engenharia de Materiais: Uma Introdução (9ª ed.). John Wiley & Sons.

Callister, W. D. (2010). Materials Science and Engineering: An Introduction (8ª ed.). John Wiley & Sons.

Disponível em: Slideshare

Disponível em: Studocu

Ashby, M. F., & Jones, D. R. H. (2012). Engineering Materials 1: An Introduction to Properties, Applications, and Design (4ª ed.). Butterworth-Heinemann.