An Open-Source Python Package for Tensile Testing Analysis

Paride Azzari[1,*], Alexander Mathys[1]

[1]Sustainable Food Processing Laboratory, Department of Health Sciences and Technology, ETH Zurich, Schmelzbergstrasse 9, 8092 Zürich, Switzerland

Abstract

Tensile testing is a crucial method in material science and engineering, providing insights into a sample's response to tension. However, the analysis of tensile test data can be time-consuming and repetitive, particularly when dealing with multiple datasets per sample. To address this issue, a Python package has been developed to streamline the data analysis process. By taking the sample size as input, this package significantly reduces analysis time, eliminates repetitive tasks, and automates the entire process.

1. Introduction

Tensile testing, a key method in material science and engineering, assesses a sample's response to controlled tension until failure, aiming to determine its tensile properties [1]. These properties, including tensile strength, Young's modulus, and Yield strength, offer insights into material behavior. The process of tensile testing involves placing the test specimen in the testing machine and slowly extending it until it fractures. The elongation of the gauge section is recorded against the applied force, and this data is used to calculate the strain and stress [2]. However, the sheer volume of data generated by tensile tests, especially when multiple datasets are produced for each sample, poses a significant challenge in terms of analysis efficiency and accuracy. Traditional manual analysis methods often entail tedious and time-consuming tasks, including data filtering, analysis, and parameter estimation, impeding the pace of research and development in materials science. To address this challenge, a novel Python package has been developed to streamline the analysis process, offering a solution to reduce analysis time, eliminate repetitive tasks, and automate the entire workflow. This article presents the documentation of this Python package [3], outlining its functionalities, capabilities, and contributions to enhancing the efficiency and effectiveness of tensile test data analysis.

2. Physics Fundamentals

In the domain of material physics, the strain (ϵ) experienced by a sample at a given time t can be determined using the formula:

$$\epsilon = \ln \frac{L(t)}{L_0} \tag{1}$$

*Corresponding author.

paride.azzari@hest.ethz.ch (P. Azzari); alexander.mathys@hest.ethz.ch (A. Mathys)

https://sfp.ethz.ch/ (A. Mathys)

where $L(t)$ represents the length of the sample at time t, and L_0 denotes its initial length.

The stress (σ) acting upon the sample is computed as the ratio of the applied force (F) to the cross-sectional area (A) of the specimen:

$$\sigma = \frac{F(t)}{A} \tag{2}$$

assuming constant volume of the specimen during measurement.

The Tensile Strength is derived from the maximum stress experienced by the sample:

$$\text{Tensile Strength} = \max(\sigma) \tag{3}$$

The Young's Modulus, a measure of stiffness, is obtained from the stress-strain curve as the slope of its linear regime:

$$\text{Young's Modulus} = \frac{\sigma}{\epsilon} \tag{4}$$

Lastly, the Yield Strength is defined as the point of intersection between the stress-strain curve and the extrapolated linear regime line shifted by 2% in strain.

A more detailed discussion of tensile testing quantities can be found in Ref.[1].

3. Tensile Analysis

TRA files are data files containing a time series of three quantities: force, depth, and gap width. Specifically, the force is quantified in Newtons (*N*), and both depth and gap width are measured in millimeters (*mm*).

Strain is calculated through equation (1), where L_0 as input. The evolving length, $L(t)$, is derived from the depth channel data, $D(t)$, through the simple addition of L_0 according to the relation:

$$L(t) = D(t) + L_0 \tag{5}$$

The stress is computed based on the sample area, A, which is also specified as input. The stress-strain relationship is obtained through the combined application of equations (1) and (2) to the collected data points.

To obtain the Young's Modulus, particularly in scenarios where nonlinearity prevails within the small strain regime, a systematic approach is adopted, where successive linear fits within the 0 to 10% strain range are performed. The optimal fit is obtained by evaluating their goodness-of-fit metric. This process effectively circumvents any non-linear effects stemming from the concave or convex behaviors that may manifest for small strains.

4. Available Functions

The Extrudion package comes with one public main class method:

```
import extrudion as ex
ex.start(folder_path)
```

This start an automated process where the folder given as a function argument gets analyzed file by file. The initial length and sample area are given as user input. Additional methods for analyzing a specific file or a specific folder can be called by importing the *extrudion.file* file: The *TRAFolder* class for analyzing a full folder, without using the command line input:

```
import extrudion.file as ex
ex.TRAFolder(folder_path).analyze( _
{'sample_area': sample_area, 'initial_length': initial_length})
```

The *TRAFolder* class for analyzing a single file:

```
import extrudion.file as ex
ex.TRAFile(file_path).analyze( _
{'sample_area': sample_area, 'initial_length': initial_length})
```

5. Results

The output of the analysis is as shown in Figure 1. The stress-strain curve, depicted in blue, features a blue dot denoting the maximum stress. A fitted line representing Young's modulus is depicted in orange dashed, alongside a 2% shifted line displayed in dotted green. The point of intersection between the stress-strain curve and the green dotted line shows the stress and strain yield.

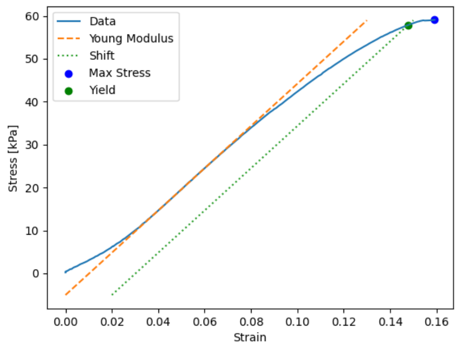

Figure 1: Plot output of the analyzed TRA file. The stress-strain curve is shown in blue, with a blue dot to represent the Max Stress. The Young's modulus fitted line is graphed as an orange dashed line, while the 2% shifted line is in dotted green. The intersection between the stress-strain curve and the green dotted line gives the Stress and Strain Yield.

The numerical analysis output is stored in a *csv* file. An example is shown in Table 1 For each file we have max stress and strain coordinates, Young's modulus, and the intercept of the fitting line in the linear regime. The intersection coordinates between the shifted line and the stress-strain curve are displayed as yield stress and yield strain.

Table 1
Summary of Tensile Test Results

File	Max Stress kPa	Max Strain	Young Modulus kPa	Intercept kPa	Yield Stress kPa	Yield Strain
File.TRA	61.73	0.14	692	0.80	50.29	0.092

6. Conclusion

The documentation provided in this article serves as a comprehensive guide for users. The Python Package[3] allows user to streamline data analysis and reduce repetitive tasks. The package is able to batch process files TRA in a single folder and output their analysis as a plot, as depicted in Figure 1 and as a csv datafile, as shown in Table 1. The output includes the calculation of critical mechanical properties such as the maximum stress, Young's modulus, yield stress, and strain [1].

Acknowledgments

Thanks to Corina Sägesser and Thomas Mair from ETH Zürich - Sustainable Food Processing Laboratory, for the help in developing this package.

A. Online Resources

The main source for the Python Package is the Github repository that is available at https://github.com/azzarip/extrudion_py. The package can also be installed directly using Pip as *pip install extrudion*.
For faster file analysis, a batch file can be downloaded from https://github.com/azzarip/extrudion. This file automatically updates Pip and Extrudion to the latest version and then runs the *start* method in the folder, where the .bat file is located.

References

[1] Joseph R Davis. *Tensile testing*. ASM international, 2004.

[2] Emmanuel Gdoutos and Maria Konsta-Gdoutos. "Tensile Testing". In: *Mechanical Testing of Materials*. Springer, 2024, pp. 1–34.

[3] Paride Azzari. *Extrudion: Python Package for Tensile Analysis*. Version 2.0.4. Mar. 2024. URL: https://github.com/azzarip/extrudion_py.