Automated processing of Experimental data and reports using Free Software

Eliška Cézová

Department of Designing and Machine Components, Faculty of Mechanical Engineering, Czech Technical University in Prague; Technická 4, 166 07 Prague, Czech Republic;
eliska.cezova@fs.cvut.cz

Abstract This article describes how computers and free software tools can be used to process experimental data and prepare corresponding reports and protocols. The Python and LATEX are presented here as an example of suitable freely available open source software. They are a viable option offering numerous advantages over the widely used general purpose commercial software tools employing standard office spreadsheet and text editors. The use of this software is demonstrated here on a simple example experimental measurement processing. It can be shown that this approach leads to significant time savings during repeated measurements and important financial savings due to use of free software tools.

Keywords: data processing, visualization, statistics, measurement, experiment, Python, LaTeX

1 Introduction

1.1 Motivation

With the development of modern high accuracy and high resolution experimental techniques and measurement tools, the amount of experimental data produced is rapidly increasing. In order to process such voluminous and complex data sets, it is often necessary to use powerful computers and a variety of specialized software tools to gain the required information. This need for computer processing is even more accentuated when a large number of repeated measurements is performed. In such cases, the entire measurement procedure is often automated and controlled by software.

It is good to note, that what is described here for physical (real) experiments, also applies to mathematical models and their outputs. Although mathematical models and numerical (virtual) experiments can provide very useful and detailed insight into many real life phenomena, they still do relay on validation and calibration using physical measurements and experimentally obtained data sets.

An integral part of most experiments and measurements, is the preparation of a final report or protocol. The form and content of such protocol follows the requirements of the customer but often also has to respect the relevant legislation and technical standards. Given the complexity of the whole process, it is advantageous if the

M. Petr et al. (eds.), *62nd International Conference of Machine Design Departments (ICMD 2022)*,
Atlantis Highlights in Engineering 28,
https://doi.org/10.2991/978-94-6463-423-5_42

process of compiling the final report (or even its presentation) can be automated in some way.

The aim of this paper is to present an example of the use of freely available software in the processing and evaluation and presentation of measured data, representing an output of an experiment.

1.2 Typical experiment processing

In the process of designing an experiment, the following questions should be asked:

- What are the objectives of the measurement?
- What quantities will be measured?
- Will (can) the measurements be repeated?
- What quantities will be evaluated from the obtained data?
- What evaluation methods/statistics will be used?
- In which program will the data be processed and collected?
- What measurement tool and gauges will be used in the experiment?
- Are the measuring instruments calibrated?
- What will be the factors that can affect the quality of the experimental data?
- How the accuracy of the results will be verified?
- How will the measurement report be prepared?

This list shows how complex and time-consuming the preparation an experiment usually is. Therefore, all tools and methods that help to reduce the number of repetitive mechanical (routine) activities are important, allowing to concentrate on the substance of data evaluation and processing, leaving the mechanically performed tasks to automatic devices and computers. This not only allows to save time and reduce personal costs, but also helps to avoid human-induced errors in the experiments.

A typical processing of an experiment usually involves the following steps:

- specifying the goal of the experiment (measurement)
- preparation of experimental setup (samples, measuring equipment, etc.)
- performing the measurements
- data processing (calculations)
- outputs post-processing (visualization)
- preparation of the protocol (final report)
- interpretation and presentation of results

The whole process of conducting an experiment from its assignment to the presentation of the final results can be summarized into the diagram shown in Figure 1. The measured data, together with the input values and other parameters of the whole experiment, form the input for the next stage, which is the processing of all this data, often involving extensive computational operations. This step is included in the C*al-*

culation block in the figure and can be performed, for example, using Python. The output of this computational stage are the postprocessed data including the analysis of errors and uncertainties in the measurement process. These outputs then form the necessary input to the *Visualization* block, which again can be largely performed using Python. The output of the visualization step can for example consists of processed and formatted images, diagrams and tables. For the *Final Protocol* (report) are then used both, the parameters of the experimental setup and the images and text outputs from the previous processing stages. At this stage also the user input of additional data or parameters is possible. For editing and formatting of the final report, the LaTeX can be used, which allows to automatically load all the necessary data inputs and additional materials into a pre-prepared template and create a report describing the entire experimental measurement. The output of this report compilation can for example, be a file in PDF format, suitable both for printing and electronic communication. In specific cases, the final report can be further interpreted and presented based on the client's requirements.

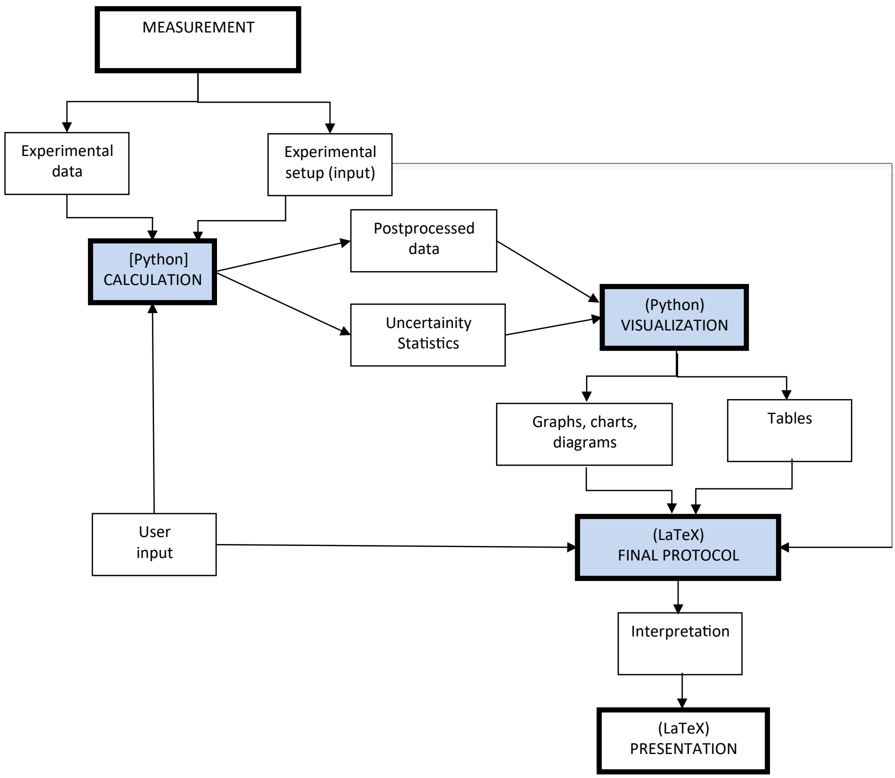

Fig. 1. Experiment processing procedure.

1.3 Goal and structure of the paper

In the following text the three main parts of the whole process (represented in the figure by three highlighted blocks), i.e. *Calculation*, *Visualization* and *Final Protocol* processing are presented and discussed in more detail.

The use of Python and LaTeX is shown as a viable combination of software tools allowing to handle the whole process. Python is a modern, versatile and high-level programming language. It can be learned very quickly. It has a simple and clear syntax. It can be used for all common platforms (Windows, Linux, Unix, MacOS) where it is available for free. Python can also generate images (graphs, charts and other plots) that can easily be embedded in LaTeX. In this way, the desired automation with all the cited advantages is very much facilitated. In the following, we describe in detail the general procedure of data evaluation in Python and LaTeX software. [2]

2 Calculation (using Python)

Typical measurement involves numerous data collection. In case the measurement is performed repeatedly, it is possible to prepare a template for the processing of experimental data. In some cases the evaluation is simple and it is often sufficient to calculate the mean and the standard deviation of the obtained data (like for example in the case of bolts investigation [4]). In some cases however, it is necessary to use more complex statistics, which can, for example, be the SPC, correlation, regression, etc.

Hereafter we will demonstrate the use of Python an LaTeX on a simple example of measured data representing a simple dependence of one output value on one input value. Such data can simply be written to a data file with two columns. The computational task of processing this data is then to determine the parameters of the linear dependence (perform a linear regression), including an evaluation of the associated uncertainties (standard deviation).

For the preparation of the experiment, the input parameters can, for example, be the material of the measured sample, the external temperature, etc. For the previously mentioned example with bolts and screws evaluation, it could be the friction, lubrication and its type according to the size of the screws, etc. The measurements in general produce different data files to work with. It is always necessary to think about what (or if) additional input parameters are needed for the calculation procedure.

Another important factor of the calculation procedure is in which program the calculations will be performed. Besides of the Python presented here, the Excel, Statistica, Mathematica or Minitab are often used. The choice of the processing software not only dictates the data format of the experimental input/output files, but also defines the type of outputs to be available from those chosen software tools.

Figures 2 - 5 show examples of simple input data and the source code for calculating linear regression in Python.

```
16.600000          508.296555
16.700000          507.169759
16.800000          505.880929
16.900000          504.294706
17.000000          504.481053
```

Fig. 2. Example of input values in .csv.

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

df = pd.read_csv('2022dat.csv', sep='\s+',header=None)
df.head()
```

Fig. 3. Example of loading an input data file into Python.

	0	1
0	16.6	508.296555
1	16.7	507.169759
2	16.8	505.880929
3	16.9	504.294706
4	17.0	504.481053

Fig. 4. Sample of loaded data in Python.

The resulting (post processed) data can be written to a file that will be used both for visualization and for the final report editing.

3 Visualization (using Python)

Visualization can be done in specialized software such as LabView, Grapher, etc.; but even Python, which was used to process the experimental data, allows basic visualization in the form of graphs, charts, and tables, see Figures 5-8. For the chosen example of linear regression, the visualization process using Python would display both the original measured data (individual data points) and the interpolated linear dependence in the form of a graph, see Figures 6 and 8.

```
import matplotlib.pyplot as plt
x = [16.6,16.7,16.8,16.9,17.0]
y = [508.296555, 507.169759, 505.880929,504.294706,504.481053]
plt.scatter(x, y)
plt.savefig('R1.pdf', bbox_inches='tight')
```

Fig. 5. Sample source code for calculating scatter plot in Python.

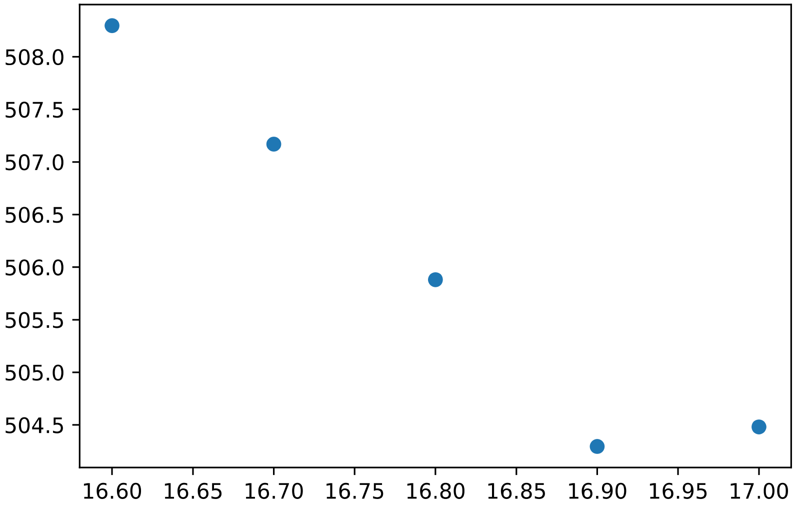

Fig. 6. Example of a graph result (scatter plot) in Python.

```
import matplotlib.pyplot as plt
from scipy import stats

x = [16.6,16.7,16.8,16.9,17.0]
y = [508.296555, 507.169759, 505.880929,504.294706,504.481053]

slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x, y)
plt.plot(x, mymodel)
plt.savefig('R2.pdf', bbox_inches='tight')
```

Fig. 7. Sample source code for calculating linear regression (linear interpolation) in Python.

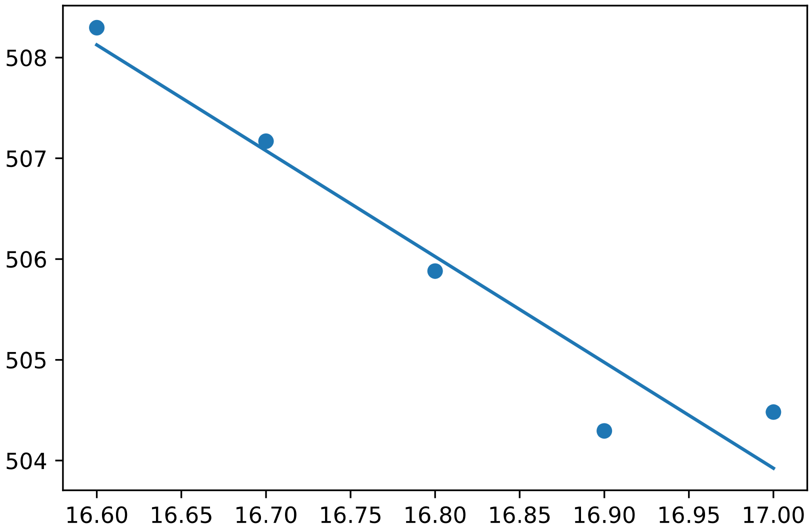

Fig. 8. Example of a graph result (linear regression interleaved with a straight line) in Python.

```python
import numpy as np
x = np.array([16.6,16.7,16.8,16.9,17.0]).reshape((-1, 1))
y = np.array([508.296555, 507.169759, 505.880929,504.294706,504.481053])
model = LinearRegression()
model.fit(x, y)
r_sq = model.score(x, y)
print(f"intercept: {model.intercept_}")
print(f"coefficients: {model.coef_}")
```

Fig. 9. Example of the source code of calculating the directive in Python.

```
intercept: 682.5263580000013
coefficients: [-10.506057]
```

Fig. 10. Sample directive output from Python.

Figure 6 and 8 can for example be saved in .jpg or .pdf a bitmap format. In case another method of data processing and visualization needs to be used, a corresponding Python script or its part can easily be obtained from various repositories that are freely available on the Internet.

Figure 9 shows a sample of the Python source code for the calculation of the linear regression slope, and its subsequent visualization in Python is shown in Figure 10.

4 Final protocol editing (using LaTeX)

The pre-prepared graphical and tabular outputs form part of the protocol, which contains other elements of this experiment, the experimental setup, and a description of the technical standards and regulations on which it is based. This output is in a structured form and is clearly arranged.

The following principles should be taken into account when developing protocols. Truthfulness and clarity of data are essential. Each coherent part of the protocol, consisting of tables and graphs, must have a separate reporting capability. It is therefore not possible to read the whole protocol to understand a graph or table. The measurement specification tells what will be measured and the conclusion contains the answers to the questions in the specification, including the measured values and the measurement errors (precision). The protocol usually contains basic relationships (formulas) with descriptions of quantities and units, tables, graphs and a conclusion. Tables must have a description that uniquely identifies the contents of the table. The measured values are recorded with the precision (number of decimal places) with which they were measured. Graphs shall be clearly described, not only the description but also the indicated axes including origin and units. The measured values shall be interleaved with a curve. The interleaved curve shall start on the ordinate of the first value plotted and end on the ordinate of the last value plotted. If there are multiple curves in the graph, it must be clear what each curve means. At the end there should be a clear summary of the measured data. Justification of measured and calculated values, agreement or disagreement with expectations. Discussion of possible sources of error, if any. The instruments used are also clearly identified (type, measuring instrument, accuracy, ranges used, serial number,...). A schematic of the experiment shall be included where appropriate. Comparisons with groups must be numerical and verbal. The verbal description should include a justification of the correspondences and differences.

The software used for the final protocol may be, for example, Word, LaTeX, or any other word processor or a text editor. In the samples below, the LaTeX is used as an example of free text processing system. It is suitable for technical writing including complex mathematical symbols and formulas, tables and figures. The LaTeX system is not only free, multiplatform, but also easy to interconnect with other data processing and visualization software tools with possible scripting and automation.

Figure 11 shows an example of the source code in LaTeX demonstrating the typesetting of a simple linear regression formula shown in Figure 12 as output after compiling the LaTeX source file. Figure 13 show an example of the source code in LaTeX with simple data table and in Figure 14 the compiled LaTeX source output.

```
$$
y = Ax + B
$$
$$
A = \frac{n \sum xy - \sum x \cdot \sum y}{n \sum x^2 - (\sum x)^2} =
\frac{5 \cdot 42505.0158279- 84 \cdot 2530.123002}{5 \cdot 1411.3 - 84^2} = - 10.5061
$$
$$
B = \frac{ \sum y - A \cdot \sum x}{n} =
\frac{2530.123002 - (-10.50605) \cdot 84}{5} = 682.52635
$$
$$
y = - 10.5061x + 682.5264\\
$$
```

Fig. 11. Source code sample for linear regression formulas typeset in LaTeX.

$$y = Ax + B$$

$$A = \frac{n \sum xy - \sum x \cdot \sum y}{n \sum x^2 - (\sum x)^2} = \frac{5 \cdot 42505.0158279 - 84 \cdot 2530.123002}{5 \cdot 1411.3 - 84^2} = -10.5061$$

$$B = \frac{\sum y - A \cdot \sum x}{n} = \frac{2530.123002 - (-10.50605) \cdot 84}{5} = 682.52635$$

$$y = -10.5061x + 682.5264$$

Fig. 12. Compiled sample output from LaTeX (linear regression) example.

```
\begin{center}
  \begin{tabular}{|c|c|@{}
    \hline
    $x$& $y$\\
    \hline
    16.6 & 508.29655\\
    16.7&507.169759\\
    16.8&505.880929\\
    16.9&504.294706\\
    17.0&504.481053\\
    \hline
  \end{tabular} \\
\end{center}
```

Fig. 13. Data table written using LaTeX – source code sample.

x	y
16.6	508.29655
16.7	507.169759
16.8	505.880929
16.9	504.294706
17.0	504.481053

Fig. 14. Compiled output from LaTeX – simple data table.

The output of this final stage is a text document (including also images, tables and other complimentary material) that summarizes the whole experiment and its initial parameters and resulting conclusions. The document must also comply with standards, regulations and customer requirements. The measurement report summarizes the demonstrable facts that need to be further interpreted and adapted for further use. For more information merging Python and LaTeX, the citation [2] can be used.

5 Presentation (using LaTeX)

Sometimes the formal protocol can be taken as the basis of a presentation that shows the whole experiment in a simplified form, including the results and their subsequent interpretation. The software used for the preparing a presentation is either Powerpoint (or similar), which can be replaced by LaTeX and suitable PDF viewer.

If LaTeX and Python are used to create the protocol, it is easy to modify the LaTeX source that will be the basis of the presentation. This greatly simplifies the creation of the presentation and eliminates errors from rewriting. Since the presentation is done in LaTeX, the fragments of the source from the final report can be copied and only the formatting is properly adjusted for presentation, see Figure 15 and 16.

```latex
\begin{slide}
  \begin{minipage}{0.22\textwidth}
    \begin{center}
      \begin{tabular}{|c|c|@}
        \hline
        $x$& $y$\\
        \hline
        16.6 & 508.29655\\
        16.7&507.169759\\
        16.8&505.880929\\
        16.9&504.294706\\
        17.0&504.481053\\
        \hline
      \end{tabular} \\
    \end{center}
  \end{minipage}
  \begin{minipage}{0.73\textwidth}
    \centering
    \includegraphics[width=0.7\linewidth]{R2.PNG}\\
  \end{minipage}
  $$
  y = Ax + B
  $$
  $$
  A = \frac{n \sum xy - \sum x \cdot \sum y}{n \sum x^2 - (\sum x)^2} =
  \frac{5 \cdot 42505.0158279- 84 \cdot 2530.123002}{5 \cdot 1411.3 - 84^2} = - 10.5061
  $$
  $$
  B = \frac{ \sum y - A \cdot \sum x}{n} =
  \frac{2530.123002 - (-10.50605) \cdot 84}{5} = 682.52635
  $$
  $$
  y = - 10.5061x + 682.5264\\
  $$
\end{slide}
```

Fig. 15. Sample LaTeX source code of presentation, using fragments from the protocol source.

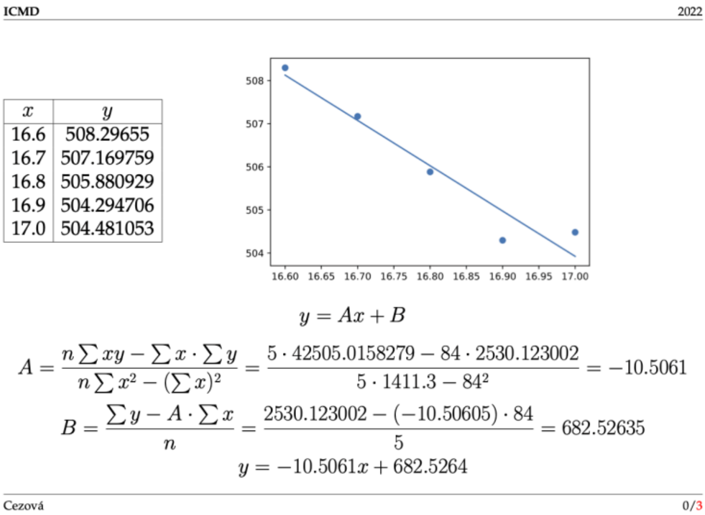

Fig. 16. Example of the compiled slide from the LaTeX source code.

6 Conclusion

The example presented in this paper clearly demonstrates how the Python and LaTeX can be used for experimental data processing, visualization and reporting. Figure 17 shows the types of possible output files covering the described sub-stages of the experiment (acquisition of measured data, calculations and presentation of results).

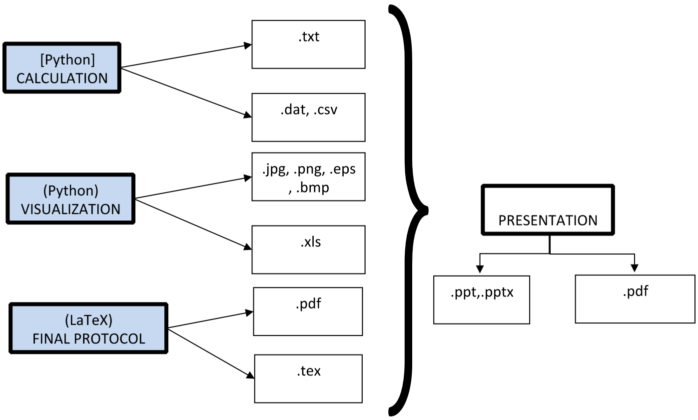

Fig. 17. Outputs of possible files.

Figure 17 shows how Python and LaTeX can be used throughout the processing of experimental data, which is in line with the aim of the paper and confirms the full applicability of these products in complex experimental data processing.

In addition, compared to classical, non-automated processing, the presented approach yields higher efficiency in the case of repetitive processing of typed tasks due to the fact that once a template has been processed, the repeated measurement only provides new data. These are uploaded to populate the template and repeat the calculations, including their documentation and visualization, are performed according to this prescription in an automated manner.

Acknowledgements This work was partly funded by the Czech Ministry of Education, under the Institutional support for the development of research organization, No.:RVO12000 for Faculty of Mechanical Engineering of the Czech Technical University in Prague

References

1. Goossens M., Rahtz S., Mittelbach F.: The LATEX Companion, Addison- Wesley, 1994
2. Cézová E.: Methods and Means in Teaching of Experimental Methods - 60th annual Conference on experimental stress analysis, EAN 2022 - Conference Proceedings, in printer
3. Pecinovský, R.: Python - Kompletní příručka jazyka pro verzi 3.10, Grada, Prague 2021, ISBN 978-80-271-3442-7
4. Kanaval, J., Cézová, E.: Strength calculation of prestressed bolted connections analysis using available computational software and fem method – 58th International Scientific Conference on Experimental Stress Analysis, EAN 2020 , pp. 195 - 203