Acknowledgment

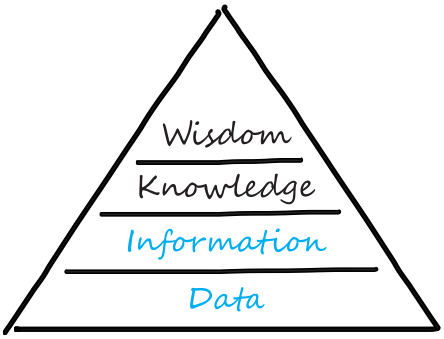
Writing this book would not have been possible without the help and support of the people I had the pleasure to work with along my journey. In particular, I want to thank Prof. Dr. Torben Gädt and Dr. Tobias Lange from TU Munich for critically reviewing the manuscript and their thoughts on the essential steps across the herein described data processing workflow.

Furthermore, I would like to thank the team at De Gruyter for giving me the opportunity to write this book as it turned out with the intention of serving practitioners in the natural sciences to advance their data-handling, analysis and visualisation capabilities.

Finally and endlessly, my gratitude goes to Katharina. After the initial *Why another book?*, you offered the continuous support and indulgence needed during the time of writing. Thank you.

Mission

The book is intended to provide a "hands-on" guide to lower-level data processing routines for data generated in the context of the natural sciences, such as chemistry, material science and the like. As in many practical use cases, raw data files obtained from an experimental device will serve as the starting point for the considerations herein. In most of the typical scenarios, these raw data files hold complex information and are best read by proprietary software provided by the respective manufacturer. According to the *data pyramid* scheme, *data—information—knowledge—wisdom*, the first step is to extract the *data* from the raw data file. Next, the extraction of *information*, i. e., the characteristic parameters from the collected *data*, will be demonstrated via the use of easily accessible tools. In particular, the focus is in on the high-level programming language Python for the critical transformation step from *data* to *information*.



https://doi.org/10.1515/9783110788433-201

Furthermore, the multiple options for storing the extracted data and information—from an easily structured file browser system across a local database to more advanced database systems—are treated, but the latter will not be the focus of the book. Finally, options for visualisation of the processed experimental data will be presented and possible methods for more detailed analyses will be explained. The process will be walked through on the basis of a simplified real-life example.

Audience

The book is intended to address graduate students of the natural sciences (probably excluding such "number-crunching" disciplines as genomics, etc., that use more advanced techniques) who seek to improve their data handling skills and capabilities for long-term accessibility of their experimental results. Likewise, the book's potential audience includes university chairs who want to ensure a sustainable long-term use of experimentally collected data across multiple "generations" of students. Another possible target audience could be smaller or midsize companies in the described fields not able or willing to employ a full-time developer for handling advanced database environments and scientific staff with a strong background in data management. In this sense, the book's goal is to promote a mutual understanding of the needs of both scientists and data engineers.

Assumptions

This book does not require in-depth previous knowledge on scripting in Python or data management, but familiarity with basic concepts such as the creation of folders and files is assumed. Some previous experience using scripting languages is certainly helpful but not mandatory for understanding the key ideas of this book and following the provided code snippets.

The examples presented herein were created using Python 3 on a Windows platform.

This book is not …

This book is not to be understood as a full-stack reference for all of the *packages* and functions used to organize the data related to the exemplary scientific question. Hence, there will be no in-depth explanations of *arguments* and *keyword arguments*, i. e., the arguments passed to a Python function. This is especially the case for those arguments that are not used in the scope of the presented examples. At this point, reference is made to the rich and current online documentation of the relevant packages.

For those sites, the documentation matching your installed version is readily available.[1,2,3] Furthermore, this book does not claim to achieve programming excellence or syntactic perfection in each line of code. Rather, it condenses the lessons obtained over the last few years as I worked with Python in a scientific context on a daily basis. In short, this is a book by one *practitioner* for the would-be practitioner.

Contents of this book

Chapter 1: Presenting the challenge

Experimental data comes in many forms. In order to ensure long-term sustainable access to experimental data generated and obtained with some effort, the following challenges, herein formulated as questions, have to be addressed by a anyone planning a well-defined data management system for natural scientific data:

- How to make the results accessible to yourself, your colleagues, your institute and/or your employer?
- How to make best use of the data? This includes drawing meaningful conclusions based on experimental evidence in order to allow for data-driven decision making.
- How to deal with new questions once the data collection has been completed?
- How to reuse experimental results?
- How to share *data* and *information*?

Chapter 2: Python quick start

Herein, the very basis of getting your Python installation up and running are addressed. Additionally, the key concepts of *scripting* used in the remainder of the book are introduced.

- How to install Python via distributions (such as WinPython or Anaconda)?
- Why is Python used herein?
- The *Zen of Python*.
- Key data structures.
- Key packages.

Essential examples of code are shown, and the basics of prototyping/scripting in the integrated development environment (IDE) *spyder* are explained.

---

1 https://pandas.pydata.org/docs/

2 https://seaborn.pydata.org/api.html

3 https://matplotlib.org/stable/api/index

Chapter 3: The steps of data processing

Probably, *the* challenge in a scientific, or likely in any other, field is to move up the *data pyramid* as defined within this chapter. According to this concept, *information* is defined in terms of *data*, *knowledge* in terms of *information* and *wisdom* in terms of *knowledge*. Critical for understanding is the notion that we cannot climb to the upper regions of this pyramid without having transversed the lower stages. In essence: There is no *wisdom* without *data*.

This problem is also linked to how we typically obtain results from experimental devices. Most of the time, they merely provide *data*, i. e., the lowest level of the *data pyramid,* in a more or less accessible way. The effort to master this *data*, i. e., to visualize the results, feels like a huge step. In reality, however, we haven't yet moved up to the next step of the pyramid. We just made the first step.

Developing a sound understanding of how to access the bottom levels according to this schema, *data* and *information*, will be the focus of the major part of this book.

Chapter 4: From experimental files to data

Getting access to the *data* part of an experimentally obtained results file is key to all of the downstream steps in data processing. So, let's start our Python scripting endeavour at the beginning and tackle the problem at the source. The following topics will be addressed:

- How to extract and separate *data*, *metadata* and *information* from an experimental results file?
- How can *regular expressions* assist us in this task?

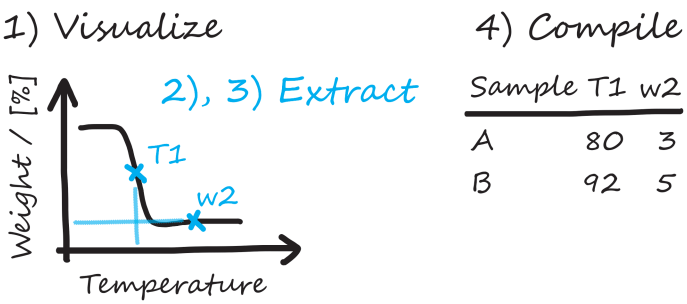
Chapter 5: From data to information

This can be justifiably regarded as the "fun part" for those experts in the respective fields having the storied *domain knowledge*. Long-standing hunches can be "translated" into quantifiable parameters based on experimental data. A four-step approach is suggested for extracting *parameters* (relevant and/or characteristic numbers), i. e., *information,* from the now available *data* (pure numbers without additional context):

- visualize collected *data*,
- extract parameters generally accepted in the field (state-of-the-art parameters),
- extract parameters that *might* be interesting for the purpose of a particular analysis but are not regularly considered in the field, and
- compile a set of parameters corresponding to a sample.

To illustrate the idea, let's consider some exemplary results from thermogravimetric analysis (TGA). In this method, the mass of a sample is measured upon heating. "Steps" in the derived relative mass versus temperature plots are associated with characteristic chemical decomposition processes. From those, we extract a temperature, *T1*, at which the maximum change in mass upon increasing temperature occurs. For

the sake of the example, this is to be understood a the *standard parameter*. Furthermore, we extract the remaining weight at a specific temperature, *w2*, as a *custom parameter*. This value might be relevant, e. g., for processual reasons such as ensuring minimum contents of a residual component given the defined process conditions restricted by the setup.



In the practical example presented later in this book, parameters are extracted using a Python script.

Chapter 6: Where to put data and information

Storing both *data* and *information* in one or the other way is a critical step to ensure long-term accessibility and thereby long-term use. Depending on the volume of data and possible restrictions in your environment (such as access and admin rights), there are approaches spanning several levels of complexity:

- Basic: using a folder structure populated via scripts using strict naming conventions. Files should not be opened and modified manually. A negative aspect of relying on this method is the requirement to maintain strict discipline.
- Intermediate: using local database files, such as *SQlite*, coming with Python or *Microsoft® Access®*.
- Advanced: hosted databases such as *MySQL* or *PostgreSQL*.

The general idea is merely to establish a “space”, where *data* and *information* are provided in a readily searchable, findable and accessible way. In this chapter, two examples are shown in great detail:

- Saving to an organized folder structure via a Python script.
- Saving to a *SQlite* database via `SQLAlchemy` (basic `SQLAlchemy` intro).

Chapter 7: How to visualize data and information

Once stored in the folder structure or database, there are multiple ways to visualize spanning diverse levels of user friendliness and complexity:

- Python script plus visualization libraries such as `matplotlib` and/or `seaborn`.
- Drag-and-drop solutions and dedicated software (Microsoft® Power BI Desktop®, Tableau, Origin).
- Custom Graphical User Interfaces (GUIs). Due to the limited scope of this book, this topic will not be covered in detail.

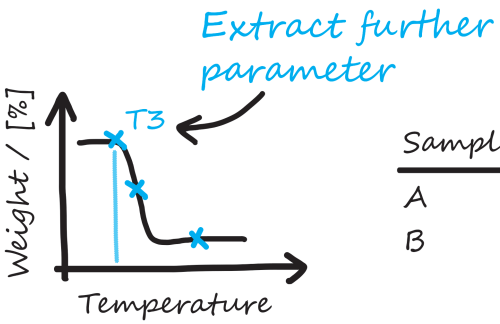
Chapter 8: Responding to lessons learned/posterior information

In most realistic cases, there's some insight generated in the course of conducting and analysing experiments. Ideally, you extracted the *right* parameters, i. e., pieces of *information* from collected *data*, right from the beginning of a series of experiments or a project.

Another common scenario is the situation in which you find at some midpoint or at the end that other parameters should have been collected—either for your own experiments or for those performed by colleagues in the past.

Having access to the *data*, extraction of further parameters is still possible in hindsight. Herein, a way to add further parameters to an already existing set of data and information is shown using a SQlite database via Python and `SQLAlchemy`.

Returning to the previously introduced TGA example, you might find that the *onset temperature* of the maximum change in observed mass, T3, is a highly relevant parameter. Accordingly, this value can be readily extracted from the *data* and included in the already existing list.



Sample	T1	w2	T3
A	80	3	66
B	92	5	71

Chapter 9: Where to go from here

So are you still waiting for *knowledge* or even *wisdom* according to the introduced concept of the *data pyramid*? Now, with your *data* and *information* sorted out, there's at least the chance to go there via techniques popularised by the huge field of data science.

The first steps could involve *linear modeling*. But also more complex approaches, such as *neural networks* or other *machine learning* techniques, can be applied (for larger data sets) to extract insights and conclusions from your experiments.

On a smaller scale, i. e., when one has no access to thousands or even more comparable experiments, scientists are typically attracted to linear models due to their simplicity and clear *cause–effect pattern*. For instance, if parameter A is increased, value B also increases (all other things being equal).

A widely neglected pathway in the field of natural sciences is probably rigorous *causal analysis*. Therein, the aim is to draw cause-and-effect relationships between observable variables and results via a previously defined hypothesis represented by a *graph*. In order to shed light on this highly interesting mode of operation, a small example coined to the world of natural sciences is given at the end of this chapter.

Chapter 10: Conclusion

The steps of the data processing routine described in the book are reiterated with a focus on the challenges associated with each of them.

Conventions used in this book

The following typographical conventions are used in this book:

Italic

for new terms, URLs, filenames, file extensions, pathnames and directories.

`Typewriter`

for commands, options, variables, attributes, keys, functions, types, classes, methods and modules.

SMALL CAPS

for Structured Query Language (SQL) keywords and queries.

This environment indicates some additional information.

This environment indicates attention to be directed to the described issue.

This environment indicates the option for some exercise based on the mentioned code snippet.

Concerning code snippets

Attribution of the code snippets considered useful for your project is appreciated but not necessarily required. An attribution regularly includes the title, author, publisher and ISBN. For the present case, this could be "*Data Management for Natural Scientists* by Matthias Hofmann, De Gruyter, 2023, ISBN 978-3-11-078840-2".

The exemplary experimental results files and Python scripts shown in the following are available free of charge on

- https://www.degruyter.com/document/isbn/9783110788433/html, and
- https://github.com/mj-hofmann/Data-Management-for-Natural-Scientists.

Microsoft® copyrighted content

The references to and screenshots of Microsoft products herein comply with the conditions of use of Microsoft copyrighted content (https://www.microsoft.com/en-us/legal/intellectualproperty/copyright/permissions). Accordingly, they are used with permission from Microsoft.