Acknowledgements

Writing this book would not have been possible without the help and support of people I had the pleasure to work with in both academia and the chemical industry.

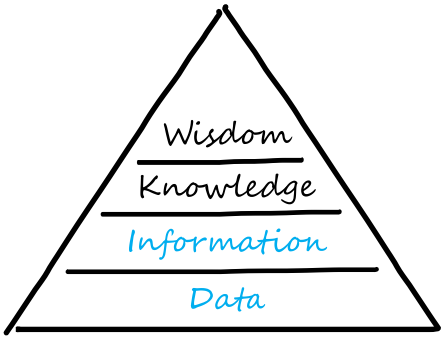
I would also like to express my gratitude to the team at De Gruyter for allowing me to write this book as an extension to *Data Management for Natural Scientists – A Practical Guide to Data Extraction and Storage Using Python* with the intention of serving R&D scientists improve their data handling, analysis and visualization capabilities. More than ever before, I am convinced that these skills need to be considered holistically, and are essential to the skill set of any modern era practitioner in the scientific field.

Finally and endlessly, my gratitude goes to Katharina.

Mission

The book is intended to serve as a "hands-on" guide to lower-level data processing routines for data generated in the context of natural sciences such as chemistry or materials science.

Referring to the Data – Information – Knowledge – Wisdom (DIKW) pyramid, the target here is to provide recipes for systematically ascending the steps of this model to speed up and improve the quality of your research. Everything shown here merely serves as a tool to support frequently encountered challenges in scientific workflows. Therefore, the focus is on the high-level programming language Python for all the concepts described here.



The hypothetical scenario of this book is setting up a research project from scratch. In addition to the physical preparations needed, the following concepts recommend steps to additionally set up your project for success from a data management, analysis and visualization point of view. The process is illustrated with the help of a simplified real-world example.

https://doi.org/10.1515/9783111334608-201

Scientific Data: A 50 Steps Guide using Python provides a compilation of the data processing steps that are essential for natural scientists. It aims to point out the dualism of classical natural sciences as taught in universities and the ever-growing need for technological/digital capabilities. The book covers a representative use case where both technical issues and scientific aspects are equally considered. After the initialization of a sustainable Python environment, the discussion shifts from these technical aspects to more scientific issues such as the extraction of meaningful characteristics and multi-objective optimization. The concise problem-solution-discussion structure throughout the chapters, supported by Python code snippets, emphasizes the book's ambition to serve as a practitioner's guide. It will give you a solid understanding of how to analyze experimental data in a common natural scientific context and how to ensure the sustainable use of your findings.

Audience

The book is addressed to graduate students of the natural sciences willing to improve their data handling skills and capabilities for long term accessibility of their experimental results. Another potential audience for the book are university professors who want to ensure a sustainable long-term use of experimentally collected data over several "generations" of students. An additional possible target audience could be small or medium-sized companies in the described field that are unable or unwilling to employ a full-time developer to handle advanced database environments and scientific staff with a strong background in data management. In this sense, the goal of the book is to promote a mutual understanding of the needs of both scientists and data engineers.

The greatest value of this book is probably inherent for (self-considered) *science dataists*. Although I could not find a strict definition of this term,[1] it refers to my understanding of a (trained) R&D scientist using *data science* tools as an "add-on" to improve the speed and quality of his or her natural scientific work.

Assumptions

This book does not require in-depth previous knowledge of Python or data management. Some experience with scripting languages is certainly helpful but not mandatory for understanding the key ideas of this book and along with the code snippets provided.

The examples presented here were created using Python 3 on a Windows machine.

1 ... to the best of my knowledge.

This book is not ...

This book is not intended as a full-stack reference for all of the *packages* and functions used for the purpose of organizing the data related to the exemplary scientific question. Hence, there will be no detailed explanations of *arguments* and *keyword arguments*, i. e., the arguments passed to a Python function. This is particularly the case for those arguments, which are not used in the context of the examples presented. At this point, reference is made to the extensive and up-to-date online documentation of the relevant packages. For these sites, the documentation appropriate for your installed version is readily available.[2,3,4] Furthermore, this book does not claim to show programming excellence in every line of code or syntactic perfection. Rather, it condenses the experience I have gained over the last few years working with Python in a natural scientific context on a daily basis. In short: This is a book by a so-called *practitioner* for aspiring practitioners.

Contents of this book

Basics

Summarizes the first steps in setting up your R&D project for success. In particular, it covers recommendations for getting Python up and running, managing Python packages via virtual environments, configuring the spyder integrated development environment (IDE) and the benefits of creating a GitHub account and repositories are covered therein. It also covers synchronization via *Github Desktop* and the basics of *Markdown*.

Organization

Before we begin the deep dive, we will first review the overall concept sketch that we need to keep in mind throughout the process in order to not lose sight of our target. Next, we will demonstrate initializing a project via the `poetry` package and tracking the environment created. In addition, `poetry` is also helpful for sharing projects once they are complete (or in an intermediate state). The remaining concepts in this section cover path handling on local drives, the benefits of writing convenience functions, the use of *toml* files and introduce the concept of *testing*.

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

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

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

Interfacing with common data formats

Handling experimental results or other source files is an integral part of hands-on scientific R&D work. Accordingly, reading the contents of Microsoft® Excel®, *txt*-, *Microsoft® Word®*- and *pdf*-files. Retrieving information from websites is also briefly covered. The section concludes with a short excursion to *regular expressions* and reading/writing to a SQLite database.

Planning experiments and/or building on legacy data/information

This is where the blank sheet approach meets the reality of legacy data. This section presents both concepts for proposing meaningful experiments from scratch and for leveraging already existing experimental results and conditions.

Collecting experimental data / lab work phase

Here we describe approaches for collecting experimental data in an efficient way during the lab work phase. In the case of the first approach of using dedicated Python packages that are already available, we move on to towards the building the missing parts[5] (in our opinion).

Visualization of experimental results

For the visualization of experimental results, plotting using the packages `matplotlib`, `seaborn` and `plotly` is demonstrated. Additionally, some concepts for visualizing multidimensional datasets are presented.

Approaching the scientific questions (modeling and recommendation)

This section covers the heart of the R&D process, which benefits most from a combination of in-depth scientific knowledge and advanced data processing and analysis skills. This is where the previously mentioned *science dataists* tend to shine.

A scientific background is certainly helpful for selecting *relevant* data and information. However, modeling the relations between defined inputs and obtained outputs is more in the realm of data science. With the models as – ideally – *digital twins* available

5 Ideally, those developments should be made available to a broader audience, if possible.

through different tools, R&D scientists can simulate and thereby validate system behavior. This clearly shows the close connection between *natural sciences* and *data science* at the very core of the *data scientific* process.

Additionally, a method is proposed for dealing with too few experiments.[6] The section concludes with the idea of numerically solving the "reverse design problem" applying multi-objective optimization and an introduction to formalized causal inference.

Sharing the project

Ideally, sharing the results of your work concludes a hopefully successful project. Once again, the `poetry` package is shown to be an effective tool for building files for distribution and publishing the package to package indices such as The Python Package Index (PyPI). Additionally, the option of sharing contents via `streamlit` applications is demonstrated.

Further reading

This concluding section addresses additional topics such as ensuring *code styling* using the `black` package, configuring `pre-commit` and a primer on building standalone solutions using the `PyQt` package.

Conventions used in this book

The following typographical conventions are used in this book:

Italic
for new terms, URLs, file names, file extensions, path names 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 additional information.

---

6 As many of experiments are costly and/or time consuming, handling with comparably small datasets is quite common.

This environment indicates that attention should be paid to the issue described.

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

Concerning code snippets

The attribution of 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 "*Scientific Data: A 50 Steps Guide using Python* by Matthias Hofmann, 2024, De Gruyter, ISBN-978-3-11-133457-8".

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

– https://www.degruyter.com/document/isbn/9783111334608/html, and
– https://github.com/mj-hofmann/Concepts-Use-Case.

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.