The Journal of Open Source Education

Course Materials for an Introduction to Data-Driven Chemistry

James Cumby[1], **Matteo T. Degiacomi**[2], **Valentina Erastova**[1], **J. Jasmin Güven**[1], **Claire L. Hobday**[1], **Antonia S. J. S. Mey**[1], **Hannah Pollak**[1], **and Rafal Szabla**[3]

1 EaStCHEM School of Chemistry, University of Edinburgh, Joseph Black Building, David Brewster Road, Edinburgh, EH9 3FJ, United Kingdom **2** Department of Physics, Durham University, South Road, Durham, DH1 3LE, United Kingdom **3** Department of Physical and Quantum Chemistry, Faculty of Chemistry, Wrocław University of Science and Technology, Wrocław, Poland

DOI: 10.21105/jose.00192

Software

- Review
- Repository
- Archive

Submitted: 14 January 2023
Published: 20 May 2023

Summary

Data-Driven Chemistry is a course aimed at undergraduate students in chemistry with no prior knowledge of programming and programmatic data analysis. It is designed as a 10-week-long course,[1] introducing Python programming and its usage in data analysis typically required for a chemistry degree. The course consists of 10 units designed to be used in a blended learning environment of live coding and explanations, followed by a set of in-course tasks to be solved individually or through pair programming.

In general, we aimed to follow the teaching philosophy of Software Carpentry (Wilson, 2006) as closely as possible for each unit. While this course is designed to be taught as a whole, each unit will cover a set of self-contained topics. The modular design is intended to make it easy to reuse and mix-and-match with other courses.

The original material is hosted online using the Noteable service provided by the University of Edinburgh, but to make it available outside of the University of Edinburgh, we have migrated the content to Google CoLab. The online hosting on CoLab will enable self-guided learning, as well as classroom-based learning, ensuring that the learning is not only limited to a university classroom setting.

Statement of Need

The modern world is digital, allowing for upscaling and automation of chemical processes through robots, but also enabling the fast production of large-scale datasets. Data analyses carried out with Graphical User Interfaces (GUIs) or spreadsheet-based tools are often limited in robustness, speed, and reproducibility. Programmatic solutions fare much better in this context, but programming is not typically taught as a skill across chemistry degrees, unlike in physics, mathematics, or even biology (White et al., 2022). Both the Royal Society of Chemistry ("Employability Skills," n.d.) and the American Chemical Society (Neiles & Mertz, 2020) have identified good computational skills as key for graduate employability (Hill et al., 2019). Our course is designed to address this gap in the undergraduate chemistry curriculum at the University of Edinburgh and to ensure that chemistry graduates remain competitive with other STEM graduates. The material

[1] This is a 20 SCQF credit course including assessments, roughly equivalent to 10 ECTS or 5 US credits

is made available as open source, with the hope that it may be used in other educational settings.

In recent years, programming has been integrated into chemistry degrees as a course on *Mathematics for Chemists* (Hutchison, 2021). While this approach provides a good foundation of programming, students are often left with few applications or examples relevant to their specific degree. There are excellent resources for self-study through books (Tanemura et al., 2022) as well as more general material for self-study of Python programming. Some material exists for a general introduction to programming and data analysis with a focus on, for instance, physical chemistry (Baptista, 2021), analytical chemistry (Menke, 2020), or machine learning for chemistry (Lafuente et al., 2021). However, little material is available for complete novices that combines teaching the basics of Python programming with how it can be applied to data in physical, inorganic, analytical, and even organic chemistry. The presented course fills this gap.

Overview, Content, and Structure

Target Audience

The course is aimed at early-year undergraduate students in chemistry, either first or second year, with little or no programming background in Python or other languages. The cohort size is typically around 100 students. During the first lecture, the 2022/23 cohort was asked the question: "Do you know how to code?" Overwhelmingly (62%), students replied with "I have no prior coding experience," while an additional 30% replied - "I only have some basic Python or coding experience." Only one respondent answered that they were confident in the use of Python.

By the end of the course, students should be proficient in using Python to:

- Break a problem into logical steps and use loops and decision operations to solve tasks;
- Perform numerical operations such as vector algebra and calculate simple statistics on data sets;
- Read and clean experimental data, visualize the data, and draw appropriate conclusions from the data through simple statistical analysis;
- Fit models to numerical data and present results in a clear and well-documented manner;
- Write readable, well-documented short snippets of code for data analysis, making use of functions, loops, and conditionals.

Content

The course is structured similarly to the PCP Notebooks of Müller & Rosenzweig (2022). Data-Driven Chemistry consists of 10 Units, each designed as a 3-hour workshop session, either in-person or online. Additional tasks are provided for completion after the workshop sessions. A summary of each unit can be found in Table 1 below:

Table 1: Summary of course material.

Unit	Content Summary	Materials
01	An Introduction to algorithmic thinking and using Jupyter notebooks	Unit 1 Notebook

Unit	Content Summary	Materials
02	Variables (`int`, `float`, `string`), lists, dictionaries and tuples in Python	Unit 2 Notebook
03	Loops and conditional statements	Unit 3 Notebook
04	Functions and basic input/output	Unit 4 Notebook
05	An introduction to plotting, using units and statistical analysis	Unit 5 Notebook
06	Comparison of distributions, t-tests and working with molecular geometries	Unit 6 Notebook
07	Correlations in data and model fitting	Unit 7 Notebook
08	Applications I: Finding peaks in mass spectrometry data, fitting radioactive decay pathways and writing a chemistry quiz	Unit 8 Notebook
09	Applications II: Working with UV-Vis and small angle X-ray scattering (SAXS) data	Unit 9 Notebook
10	Applications III: Nuclear magnetic resonance (NMR) data	Unit 10 Notebook

The content is grouped into three main parts. `Unit_01` to `Unit_04` introduce concepts around algorithmic thinking and Python syntax, including variables, loops, functions, libraries, documentation, how to get help, and how to read files. These were largely adapted from Plotting and Programming in Python. `Unit_05` to `Unit_07` introduce concepts from SciPy (Virtanen et al., 2020), NumPy (Harris et al., 2020), Matplotlib (Hunter, 2007) and Pandas (The pandas development team, 2022) to carry out basic statistical analysis and plotting of chemistry-related data. Our strategy was to incorporate as many chemistry topics or techniques already familiar to students while teaching new Python content. For example, we assume that students have already studied mathematical concepts such as fitting data and comparing distributions, but now are presented with a dataset relevant to their degree. The domain-specific twist aims to boost student motivation to engage with these mathematical concepts. Therefore, `Unit_08` to `Unit_10` cover specific application examples from different areas of chemistry, and some of the applications directly tie into the students' lab experiments (e.g., UV-Vis spectroscopy and NMR data). To ensure sufficient student support, ten teaching assistants are available at each 3-hour long workshop attended by around 100 learners. Furthermore, 1-hour long Q&A sessions with the teaching assistants were organised on alternate weeks.

Assessment and feedback

The course was formally assessed at the University of Edinburgh using nbgrader (Jupyter Project et al., 2019). It was important to initially test the students formatively with weekly online quizzes, which could be completed multiple times. This gave students instant feedback on their performance, and allowed them to improve. In later weeks, the course was assessed summatively. However, we still made use of informal feedback within the sessions with built-in quizzes in the Jupyter Notebooks using Mentimeter and an associated Python Class. We polled students to test their understanding of the material, to promote critical thinking, and check their background knowledge. We also used Mentimeter to gather feedback after each session, which helped us to improve the material further. Generally, the usage of embedded quizzes helped with engagement from students. Figure 1 shows an example of a Mentimeter quiz.

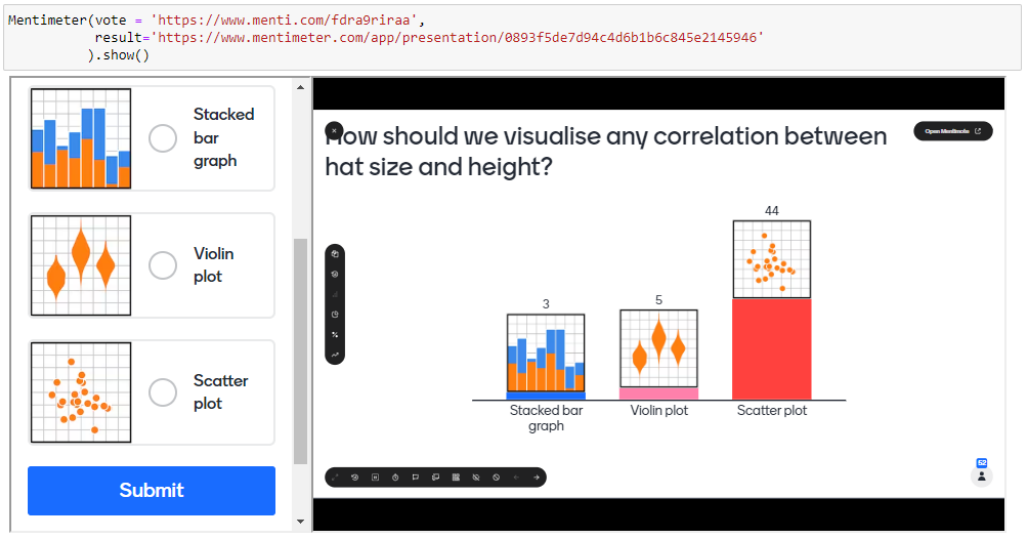

Figure 1: Example of how a Mentimeter poll can be directly embedded into a Jupyter notebook using the provided `Mentimeter` class.

Conclusion

We present a modular course to teach Python for chemistry undergraduate students, targeted at complete novices. We hope it is of value to other chemistry students and educators. Running the material through CoLab removes all installation requirements, making the course more easily accessible to novices, from students in guided university settings to other chemistry enthusiasts.

Author's Contribution

Authors are listed in alphabetical order. James Cumby (JC), Valentina Erastova (VE), Claire L. Hobday (CLH), and Antonia Mey (ASJSM) have been teaching this course at the University of Edinburgh since the academic year 2021/22. Jasmin Güven (JJG) and Hannah Pollak (HP) have been course demonstrators. Rafał Szabla (RS) taught one unit and created content for it in 2020/21, when the course was run in a shortened form as a replacement for physical chemistry laboratory practicals during the pandemic. Material was adapted from Matteo T. Degiacomi (MTD), who shared content developed in 2018 for his course at Durham University aimed at chemistry research students. He made some additional contributions beyond his original material. Specific contributions by each author are as follows:

JC: Created material for `Unit_01`, `Unit_10`, and the `helper_functions`, gave feedback on other materials, and helped edit the manuscript.
MTD: Contributed material for `Unit_03`, `Unit_05`, `Unit_07`, and `Unit_08`, and helped edit the manuscript.
VE: Created material for `Unit_05` and `Unit_08`, contributed to `Unit_06`, and helped edit the manuscript.
JJG: Contributed material to `Unit_09`, and helped edit the manuscript.
CLH: Created material for `Unit_03` and `Unit_04`, and helped edit the manuscript.
ASJSM: Created material for `Unit_02`, `Unit_06`, `Unit_07`, and `Unit_09`, provided feedback and small contributions to most other units, and wrote the manuscript.

HP: Contributed material to `Unit_05` and `Unit_08`, and helped edit the manuscript.
RS: Created the material for molecular geometries forming part of `Unit_06` and gave feedback on the manuscript.

Acknowledgements

We would like to acknowledge The Carpentries for material and inspiration, and all PhD student demonstrators who helped with teaching the material, giving feedback and creating some of the assignments.

References

Baptista, L. (2021). *Using Python and Google Colab to Teach Physical Chemistry During Pandemic*. https://doi.org/10.26434/chemrxiv.13656665.v1

Employability skills. (n.d.). In *RSC Education*. https://edu.rsc.org/future-in-chemistry/career-options/employability-skills.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hill, M. A., Overton, T. L., Thompson, C. D., Kitson, R. R. A., & Coppo, P. (2019). Undergraduate recognition of curriculum-related skill development and the skills employers are seeking. *Chem. Educ. Res. Pract.*, *20*(1), 68–84. https://doi.org/10.1039/C8RP00105G

Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Hutchison, G. R. (2021). Integrating Python into an Undergraduate Mathematics for Chemists Course. In *Teaching Programming across the Chemistry Curriculum* (Vol. 1387, pp. 123–134). American Chemical Society. https://doi.org/10.1021/bk-2021-1387.ch009

Jupyter Project, Blank, D., Bourgin, D., Brown, A., Bussonnier, M., Frederic, J., Granger, B., Griffiths, T. L., Hamrick, J., Kelley, K., Pacer, M., Page, L., Pérez, F., Ragan-Kelley, B., Suchow, J. W., & Willing, C. (2019). Nbgrader: A Tool for Creating and Grading Assignments in the Jupyter Notebook. *Journal of Open Source Education*, *2*(16), 32. https://doi.org/10.21105/jose.00032

Lafuente, D., Cohen, B., Fiorini, G., García, A. A., Bringas, M., Morzan, E., & Onna, D. (2021). A Gentle Introduction to Machine Learning for Chemists: An Undergraduate Workshop Using Python Notebooks for Visualization, Data Processing, Analysis, and Modeling. *J. Chem. Educ.*, *98*(9), 2892–2898. https://doi.org/10.1021/acs.jchemed.1c00142

Menke, E. J. (2020). Series of Jupyter Notebooks Using Python for an Analytical Chemistry Course. *J. Chem. Educ.*, *97*(10), 3899–3903. https://doi.org/10.1021/acs.jchemed.9b01131

Müller, M., & Rosenzweig, S. (2022). PCP Notebooks: A Preparation Course for Python with a Focus on Signal Processing. *Journal of Open Source Education*, *5*(47), 148. https://doi.org/10.21105/jose.00148

Neiles, K. Y., & Mertz, P. S. (2020). Professional Skills in Chemistry and Biochemistry Curricula: A Call to Action. In *Integrating Professional Skills into Undergraduate Chemistry Curricula* (Vol. 1365, pp. 3–15). American Chemical Society. https://doi.org/10.1021/bk-2020-1365.ch001

Tanemura, K. A., Sierra-Costa, D., & Merz, K. M. (2022). *Python for Chemists*. American Chemical Society. https://doi.org/10.1021/acsinfocus.7e5030

The pandas development team. (2022). *Pandas-dev/pandas: pandas* (Version v1.5.2) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.7344967

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … van Mulbregt, P. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nat Methods*, *17*(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2

White, E. P., Brym, Z. T., Marx, A. J., Riemer, K., Marconi, S., Harris, D. J., Cruz, V., & Ernest, S. K. M. (2022). Data Carpentry for Biologists: A semester long Data Carpentry course using ecological and other biological examples. *Journal of Open Source Education*, *5*(50), 139. https://doi.org/10.21105/jose.00139

Wilson, G. (2006). Software carpentry: Getting scientists to write better code by making them more productive. *Computing in Science & Engineering*. https://doi.org/10.1109/mcse.2006.122