JOURNAL OF CHEMICAL EDUCATION

pubs.acs.org/jchemeduc Article

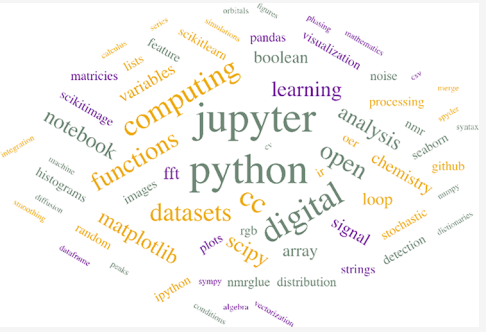
A Creative Commons Textbook for Teaching Scientific Computing to Chemistry Students with Python and Jupyter Notebooks

Charles J. Weiss*

Cite This: J. Chem. Educ. 2021, 98, 489–494 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: Scientific computing and computer literacy are increasingly important skills for chemistry students to learn, but despite this need, there is an absence of chemistry-specific texts available for teaching the subject. This article introduces a freely available textbook released under a Creative Commons license for use in an undergraduate scientific computing chemistry course teaching students basic Python programming; advanced skills in the processing, visualization, and analysis of data; and the coding of basic simulations. Subjects include basic programming, signal processing, machine learning, NMR data processing, and image analysis among others. The book is written to teach the subject using Python, Jupyter notebooks, and the SciPy stack which are all open source and cross-platform software; the text assumes no previous computer programming background from the students. This article provides descriptions and philosophy behind the structure and content of the book along with insights and advice based on its use in an undergraduate course.



KEYWORDS: *Textbooks/Reference Books, Upper-Division Undergraduate, Second-Year Undergraduate, Computer-Based Learning, Mathematics/Symbolic Mathematics, Chemoinformatics, Chemometrics*

INTRODUCTION

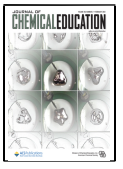
Computer programming is becoming an increasingly important skill for those in technical fields.[1] It allows individuals to automate repetitive tasks and perform data processing and analyses otherwise inaccessible through manual processing or spreadsheets. Examples include extracting information from images, processing large numbers of data files, machine learning, and working with memory-intensive data sets. To provide undergraduate chemistry students with these skills, a Scientific Computing for Chemists[2] course was developed and first taught in 2016 which teaches basic programming in the Python programming language using Jupyter notebooks and the SciPy stack to solve chemical problems. This course assumes no prior experience in computer programming from the students. Python,[3] Jupyter notebooks,[4] and the SciPy stack,[5] which together serve as a substitute for commercial software packages such as Matlab, were chosen for their ease of use and their suitability for solving chemical problems, and because they are all freely available, open source, and cross-platform pieces of software. This ensures that they are readily available to everyone regardless of budget and computer operating system (e.g., Windows, macOS, or Linux).

Despite there being numerous books on Python and the SciPy stack,[6] there is an absence of chemistry focused texts on this subject that apply these skills to solving chemistry related problems. In response, the textbook titled *Scientific Computing for Chemists* was drafted and first employed in the fall of 2017 for a scientific computing chemistry course.[2] The book has since been expanded and used again in the fall of 2019 for a scientific computing course cross-listed between the chemistry and computer science departments. Topics include basic programming, signal processing, machine learning, NMR processing, and image analysis, among others. The textbook is an open educational resource (OER)[7] released under the CC BY-NC-SA 4.0 Creative Commons license[12] and can be freely downloaded at https://github.com/weisscharlesj/SciCompforChemists. Herein, this book is introduced along with discussion on its structure and content, license, and usage in a scientific computing course.

TEXTBOOK OUTLINE

The text opens with a short chapter introducing Jupyter notebooks (Figure 1) as the environment in which all the code examples in the book, except Chapter 13, are executed. Jupyter notebooks, formerly known as IPython notebooks, were chosen for this text because they are quick to learn, are shareable documents, and provide cells for executable code, equations, and explanatory text to describe both the data and Python code. Finally, Jupyter notebooks utilize the IPython environment[8]

Received: August 11, 2020
Revised: November 19, 2020
Published: December 31, 2020



https://dx.doi.org/10.1021/acs.jchemed.0c01071
J. Chem. Educ. 2021, 98, 489–494

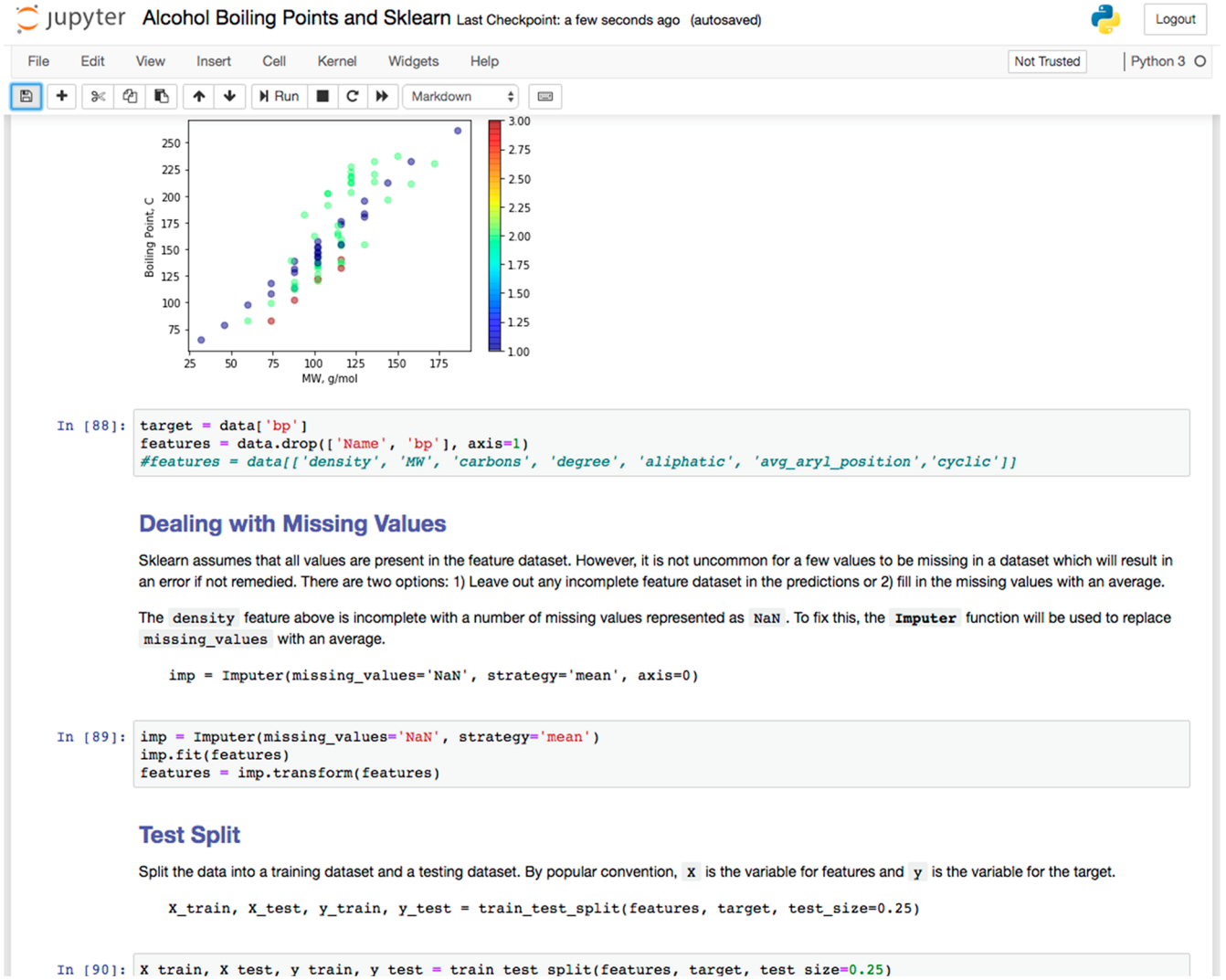


Figure 1. Screenshot of a Jupyter notebook containing code cells (gray background), markdown cells with explanatory text (white background), and a scatter plot visualizing the data.

which allows for images and plots to be displayed directly inside the notebook. These notebooks are well-suited for lectures, homework sets, and projects; Jupyter notebooks have demonstrated their utility by being employed in other science courses[9] and chemical research.[10]

The overarching philosophy of this text is to help students start solving chemical problems with Python as quickly as possible. To this end, the introduction to Python and computer programming is streamlined down to two chapters while providing sufficient programming skills for subsequent chapters. The coverage of programming concepts and Python is not as extensive as a student would likely receive from a traditional introductory course on computer programming using Python. Emphasis in this text is instead placed on teaching more science-specific tools such as NumPy, SciPy, matplotlib, seaborn, and pandas libraries and applying them to solving chemical problems. Libraries are collections of functions and/or data with a common theme and can be thought of as programming add-ons or tool packs. Matplotlib and seaborn are libraries containing functions for plotting and visualizing data while NumPy and pandas contain tools for storing, sorting, merging, and performing calculations on data sets. The SciPy library contains an assortment of functions and data for carrying about scientific data processing and analysis such as functions for signal processing and integrating data.

Table 1 provides a chapter outline of the book. The first six chapters (0−5) lay the foundation for programming with Python, data handling, and visualization while the subsequent chapters focus more on specific applications of these skills and more advanced Python libraries. The later chapters include a range of topics such as simulations, signal processing, machine learning, and image analysis. While the chapters are introduced in order of more fundamental topics first, not all chapters must be covered in this order. All application chapters require Chapters 0, 1, 3, 4 and parts of 2 as prerequisites, but only Chapters 10 and 12 rely on Chapter 5 (Figure 2). Chapters 10 and 12 utilize the pandas library from Chapter 5 because the seaborn and scikit-learn libraries are designed to work closely with pandas. However, seaborn and scikit-image do not strictly require pandas and could be introduced without it if desired. For this reason, the aforementioned scientific computing course[2a] routinely bypasses pandas to dive into early applications more quickly and then circles back around to Chapter 5 content later in the course before covering seaborn and scikit-learn. Because it is difficult to predict the variety of ways instructors may choose to incorporate this text into their curricula, all chapters after

https://dx.doi.org/10.1021/acs.jchemed.0c01071
J. Chem. Educ. 2021, 98, 489−494

Table 1. Chapter Outline

Chapter	Title
Foundational Topics	
0	Python and Jupyter Notebooks
1	Basic Python
2	Intermediate Python
3	Plotting with Matplotlib
4	Basic NumPy
5	Pandas for Data Management
Applications	
6	Signal & Noise
7	Image Processing & Analysis
8	Mathematics with SciPy & SymPy
9	Simulations
10	Plotting with Seaborn
11	NMR processing with NMRglue
12	Machine Learning Using Scikit-Learn
13	Command Line & Spyder

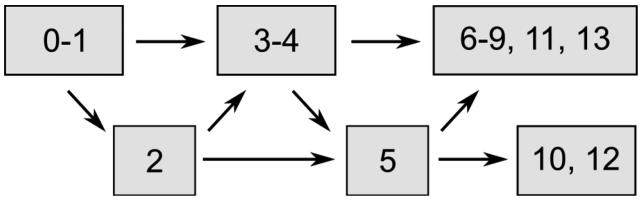


Figure 2. Map showing the orders in which the book chapters may be traversed.

Chapter 5 are designed to be traversed in any desired order to allow for versatility.

Each chapter also includes a Further Reading section which lists recommended websites, books, and articles with an emphasis on freely available resources when possible.

Descriptions of each chapter with select examples of chemistry applications from the chapter or chapter exercises are included below.

Chapter 0

This short chapter discusses the Python programming language and provides instructions on using Jupyter notebooks.

Chapter 1

This chapter introduces the reader to programming in the Python programming language and provides skills in using basic Python object types, employing loops, using built-in Python functions, and defining their own functions among others. Example applications including using loops to calculate pH during a titration and writing a function to calculate the distance between two atoms in 3D space.

Chapter 2

Chapter 2 covers intermediate Python topics including list comprehension, Python modules, extra Python object types, and more background in writing functions. This chapter can be bypassed by those in a hurry or short on time and returned to later as needed. Example applications include writing a function to calculate the molecular mass based on a molecular formula and calculating the probability of finding a particle of a particle-in-a-box in a subsection of the box.

Chapter 3

Chapter 3 introduces the reader to visualizing data using the matplotlib plotting library. Examples include plotting atomic wave functions in 2D and 3D and visualizing the distribution of specific heat capacities of common metals.

Chapter 4

In Chapter 4, the reader is introduced to the NumPy library for faster and more efficient handling of scientific data. This library is the foundation for most scientific Python libraries included in this book. Example applications include calculating the energies of eight energy levels in the Bohr model of the hydrogen atom *without* using a for loop and visualizing the relationship between ΔG and the equilibrium constant K.

Chapter 5

Chapter 5 introduces the pandas library for more advanced handing of scientific data with the major advantage of allowing users to include labels with their data. Pandas also provides a large selection of advanced tools for reading, merging, aligning, sorting, and cleaning data sets. An example application includes managing and merging data sets containing data on chemical elements and importing UV−vis data and selecting a data point on the basis of the wavelength.

Chapter 6

Signal process and the Fourier transform function are introduced in Chapter 6 that allow the reader to locate peaks and inflection points in data and to smooth and analyze noisy data. Example applications include examining IR and ^{1}H NMR spectral data.

Chapter 7

This chapter provides background on the structure of image data and introduces the scikit-image library for image processing and scientific image analysis. Example applications include using scikit-image to count objects in images and calculating the eccentricity of image objects.

Chapter 8

Chapter 8 demonstrates the use of the SymPy and SciPy libraries for performing symbolic mathematics and numerical calculations, respectively. Example applications include solving a quadratic equation to determine equilibrium concentrations and calculating the concentration of analytes in a three-component solution using Beer's law and matrices.

Chapter 9

Chapter 9 introduces basic stochastic and deterministic simulations using Python and NumPy. Example applications include simulating chemical kinetics and NMR splitting patterns.

Chapter 10

The seaborn plotting library is introduced in Chapter 10 for generating more complex plots in relatively few lines of code versus those of matplotlib. Example applications include visualizing boiling point data and generating a 2D heat map of bond enthalpy data.

Chapter 11

Chapter 11 focuses on processing and visualizing NMR data using the NMRglue library to read and process NMR data files.[13] In this chapter, a ^{1}H NMR spectrum is read, Fourier transformed, phased, referenced, and integrated in Python using NMRglue, NumPy, and matplotlib.

Chapter 12

In Chapter 12, the reader is introduced to basic supervised and unsupervised machine learning and the scikit-learn machine learning library. Example applications include predicting the

https://dx.doi.org/10.1021/acs.jchemed.0c01071
J. Chem. Educ. 2021, 98, 489−494

boiling points of alcohols and isolating the IR spectra of pure component compounds from the IR spectra of mixtures.

Chapter 13

While Jupyter notebooks are ideal for performing interactive data processing and analysis and sharing this analysis, data processing scripts that include hundreds of lines of code become cumbersome in Jupyter notebooks. Chapter 13 introduces an alternative coding environment called Spyder and demonstrates the running of Python scripts from the command line.

LICENSE

The book is released under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 (CC BY-NC-SA 4.0) license.[11] Creative Commons is a family of licenses commonly used to release creative works for free use and distribution including other free textbooks.[12] This specific version of the license permits the free use of this text with stipulations which include it be attributed to the author (BY), may only be used for NonCommercial purposes (NC), and must be released under the same license if redistributed (SA). The license defines NonCommercial as "not primarily intended for or directed towards commercial advantage or monetary compensation".[14] While this prohibits the selling of the book, it does not prevent commercial entities from using the book nor does it prevent someone from paying a commercial entity to have the book printed if the usage otherwise conforms to the license. For example, this license permits a school to pay to have copies of the book printed for their students and instructors.[15] The license does not restrict the modification of the text and conforms to the Open Educational Resource (OER) definition.[7] See the Creative Commons website[11] for the official license terms and further explanations of permitted uses.

PROBLEM SETS

Every chapter except the first concludes with practice problems to provide the reader with experience implementing the new topics. Practice is especially important in learning programming as most individuals cannot fully absorb these topics simply by reading or watching. In instances where problems reference a data file, these files are available on the website where the textbook is posted and are under the same license as the book. While these problems can be assigned directly from the textbook, they can also be distributed in Jupyter notebooks, as has been done in the above-mentioned courses. Multiple examples of homework Jupyter notebooks that have been assigned to students are included in the Supporting Information. The advantage of distributing problem sets in Jupyter notebooks is that they can contain hardcoded values or preliminary code from the instructor, and they are easier to grade as all student submitted notebooks will have identical formatting and headers. While not used in the above-mentioned courses, a Jupyter notebook can automate the grading of assignments using the nbgrader add-on.[16]

STUDENT PREREQUISITES

This text assumes that students have completed General Chemistry and have some understanding of basic spectroscopy commonly taught in undergraduate organic chemistry. The student prerequisites are intentionally kept to a minimum in an effort to make this text accessible to undergraduate chemistry students between their second and fourth years. As a result, book examples and problems do not engage in detailed analysis of data but rather focus on the technical aspects of processing, visualizing, and extracting information from the data. Any in-depth analysis of chemical data is left to individual instructors to incorporate at a level appropriate to his or her students.

Quantitative maturity is one of the most important prerequisites for students reading this book and entering into this subject. That is, students should be comfortable with plotting and interpreting data, and they should be competent in thinking about performing calculations on a series of data points even though the data set is too large for the student to view every value and calculation. Performing and keeping track of calculations on a series of data points can offer a challenge to some students when they cannot readily see all the data and observe the changes. These are skills students likely gain during their first year in undergraduate chemistry through the use of spreadsheets such as when students prepare first and second order integrated rate law plots from concentration−time series data. The inability of a student to think about these kinds of calculations can result in a student performing calculations without understanding why they are doing it and frequently needing guidance on what to do next.

EXAMPLE IMPLEMENTATION

As an example of this text being used in the classroom, this text was employed in a scientific computing course in the fall of 2019 at Augustana University. The course included 10 students and met during two 50 min blocks twice a week for a semester with the relevant sections of the book assigned to students as prelecture readings. The course schedule with reading assignments is provided in the Supporting Information. All chapters were covered in their entirety except Chapters 2, 8, 10, 11, and 13. Chapters 2 and 8, which cover intermediate Python topics and mathematics in Python, respectively, were only partially covered due to time constraints and because some of these topics are not strictly required for subsequent chapters. While Chapter 2 can be covered immediately after Chapter 1, the above-mentioned course introduces Chapter 2 topics throughout the course as needed in order to progress to more advanced chemistry applications more quickly. Chapter 10 covers the seaborn plotting library which was briefly introduced on a special topics day to make students aware of additional software. Chapters 11 and 13, which cover the NMR process using NMRglue and running Python scripts from the command line, were not covered in the course. Chapters 10, 11, and 13 were made available to students in this course merely as a resource for those who wish to read further.

While simulations can be covered immediately after Chapter 4, as was done in an earlier year, the course has been found to run more smoothly if this chapter is introduced at a later time. Simulations is one of the most challenging topics as it requires students to think more deeply about the mechanisms of physical and chemical processes. Waiting until students get further practice and proficiency with programming allows students to focus more on the simulations rather than how to implement their ideas in Python and NumPy.

Pandas is a library built on NumPy which provides both advanced data processing features and usability enhancements above the NumPy library. Early iterations of this course only dedicated one class day[2a] to this topic. Because of the historically strong student enthusiasm for the pandas library, the 2019 iteration expanded this introduction to three days. Even when NumPy could be used to solve homework and in-class problems, students often still gravitated toward pandas.

https://dx.doi.org/10.1021/acs.jchemed.0c01071
J. Chem. Educ. 2021, 98, 489−494

The seaborn plotting library is build on top of the classic matplotlib Python plotting library and allows the user to generate complex plots with relatively few lines of code. This plotting library was only briefly introduced in class on a special topics day, yet students often employed this plotting library in their homework and projects likely because of its ability to generate complex and eye-catching plots with relatively few lines of code. Future iterations of this course will likely dedicate more time to this library due to student interest.

To provide additional practice, problem sets were often assigned after each class period using problems from the back of the chapters. The problem sets were posted in the form of Jupyter notebooks which students downloaded, completed with their own code, and reuploaded for the instructor to grade. Because of the smaller class size, these homework notebooks were graded by hand, but if the course were significantly larger, nbgrader would likely have been used to assist in grading. In addition, many class periods included students coding either individually or in pairs. This was especially helpful in getting students to ask questions before leaving class as it is often not until students attempt a problem that they realize what they do and do not fully understand. Multiple activity days were also included in the schedule where students spent the entire class period coding in small groups with the instructor available to assist and answer questions. This is an effective way to get students asking questions and get help from the instructor and classmates before they work homework problems on their own. In-class coding is especially recommended for more challenging topics such as simulations and machine learning as students can easily get stuck in these topics.

Many students readily see the value of computer programming and have remarked on the utility of the skills learned in this course. While all students do not necessarily embrace computer programming, it is not unusual to hear about former students using skills from this course in other courses, and a few students have gone on to perform research that relies significantly on computer programming and other skills learned in this course.

SUMMARY

Scientific Computing for Chemists is a freely available textbook released under the CC BY-NC-SA 4.0 Creative Commons license for introducing chemistry students to computer programming and advanced digital data analysis techniques and tools. The text assumes no prior computer programming experience and only about a year of college-level chemistry background from students to allow for it to be incorporated into an undergraduate curriculum in a variety of locations. Focusing on teaching advanced computing tools for solving chemistry related problems, the book provides an abbreviated introduction to programming in Python in order to allow students to advance quickly to solving chemical problems such as the chemistry applications included in the chapters and postchapter exercises. The chapters are designed to be traversed in a variety of sequences to allow for flexibility in how the text is used in a course.

ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available at https://pubs.acs.org/doi/10.1021/acs.jchemed.0c01071.

Example course schedule used in the fall of 2019 (PDF)
Example homework problem sets (ZIP)

AUTHOR INFORMATION

Corresponding Author

Charles J. Weiss − *Department of Chemistry and Biochemistry, Augustana University, Sioux Falls, South Dakota 57197, United States;* orcid.org/0000-0003-4102-7683; Email: charles.weiss@augie.edu

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jchemed.0c01071

Notes

The author declares no competing financial interest.
The full textbook (PDF) can be freely downloaded at https://github.com/weisscharlesj/SciCompforChemists.

ACKNOWLEDGMENTS

Thank you to my friends and colleagues, current and former, at Augustana University and Wabash College for their support of my scientific computing course, and thank you to all my computing students and those who reported typos and provided input.

REFERENCES

(1) (a) Integrating Scientific Computing into Science Curricula. https://www.bnl.gov/newsroom/news.php?a=214374 (accessed November 2020). (b) Baker, M. Programming tools can speed up and strengthen analyses, but mastering the skills takes time and can be daunting. *Nature* **2017**, *541*, 563−566. (c) When All Science Becomes Data Science. https://www.sciencemag.org/careers/2013/05/when-all-science-becomes-data-science (accessed November 2020).

(2) (a) Weiss, C. J. Scientific Computing for Chemists: An Undergraduate Course in Simulations, Data Processing, and Visualization. *J. Chem. Educ.* **2017**, *94* (5), 592−597. (b) Weiss, C. J. Introduction to Stochastic Simulations for Chemical and Physical Processes: Principles and Applications. *J. Chem. Educ.* **2017**, *94* (12), 592−597.

(3) Python Software Foundation. https://www.python.org (accessed November 2020).

(4) (a) Jupyter Notebook Website. https://jupyter.org (accessed November 2020). (b) Perkel, J. M. Why Jupyter is Data Scientists' Computational Notebook of Choice. *Nature* **2018**, *563*, 145−146.

(5) SciPy Website. https://www.scipy.org (accessed November 2020).

(6) (a) Downey, A. B. *Think Python: How to Think Like a Computer Scientist*, 2nd ed.; O'Reilly Media: Sebastopol, CA, 2016. (b) Hill, C. *Learning Scientific Programming with Python*, 1st ed.; Cambridge University Press: Cambridge, U.K., 2016. (c) McKinney, W. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and Ipython*, 2nd ed.; O'Reilly: Sebastopol, CA, 2018. (d) VanderPlas, J. *Python Data Science Handbook: Essential Tools for Working with Data*, 1st ed.; O'Reilly: Sebastopol, CA, 2017. (e) Müller, A. C.; Guido, S. *Introduction to Machine Learning with Python*; O'Reilly: Sebastopol, CA, 2017.

(7) For the definition of an Open Educational Resource, see: Creative Commons Open Education. https://creativecommons.org/about/program-areas/education-oer/ (accessed November 2020). (b) H.R.2107 Affordable College Textbook Act. https://www.congress.gov/bill/116th-congress/house-bill/2107/text (accessed November 2020).

(8) (a) Peŕez, P.; Granger, B. E. Ipython:A System for Interactive Scientific Computing. *Comput. Sci. Eng.* **2007**, *9* (3), 21−29. (b) IPython Home Page. https://ipython.org (accessed November 2020).

(9) (a) Cruzeiro, C. W. D.; Gao, X.; Kleiman, V. D. Implementing New Educational Platforms in the Classroom: An Interactive Approach to the Particle in a Box Problem. *J. Chem. Educ.* **2019**, *96* (8), 1663−1670. (b) Srnec, M. N.; Upadhyay, S.; Madura, J. D. A Python Program

for Solving Schrödinger's Equation in Undergraduate Physical Chemistry. *J. Chem. Educ.* **2017**, *94* (6), 813−815. (c) Cruzeiro, V. W. D.; Gao, X.; Kleiman, V. D. Implementing New Educational Platforms in the Classroom: An Interactive Approach to the Particle in a Box Problem. *J. Chem. Educ.* **2019**, *96* (8), 1663−1670. (d) Srnec, M. N.; Upadhyay, S.; Madura, J. D. A Python Program for Solving Schrödinger's Equation in Undergraduate Physical Chemistry. *J. Chem. Educ.* **2017**, *94* (6), 813−815. (e) Srnec, M. N.; Upadhyay, S.; Madura, J. D. Teaching Reciprocal Space to Undergraduates via Theory and Code Components of an IPython Notebook. *J. Chem. Educ.* **2016**, *93* (12), 2106−2109. (f) Zúñiga-López, A.; Avilés-Cruz, C. Digital Signal Processing Course on Jupyter-Python Notebook for Electronics Undergraduates. *Comput. Appl. Eng. Educ.* **2020**, *28*, 1045−1057.

(10) (a) Srnec, M. N.; Upadhyay, S.; Madura, J. D. Teaching Reciprocal Space to Undergraduates via Theory and Code Components of an IPython Notebook. *J. Chem. Educ.* **2016**, *93* (12), 2106−2109. (b) Cruzeiro, V. W. D.; Gao, X.; Kleiman, V. D. Implementing New Educational Platforms in the Classroom: An Interactive Approach to the Particle in a Box Problem. *J. Chem. Educ.* **2019**, *96* (8), 1663−1670. (c) Schneider, N.; Stiefl, N.; Landrum, G. A. What's What: The (Nearly) Definitive Guide to Reaction Role Assignment. *J. Chem. Inf. Model.* **2016**, *56* (12), 2336−2346. (d) Willforss, J.; Chawade, A.; Levander, F. NormalyzerDE: Online Tool for Improved Normalization of Omics Expression Data and High-Sensitivity Differential Expression Analysis. *J. Proteome Res.* **2019**, *18* (2), 732−740. (e) Ou, L.; Luo, G.; Ray, A.; Li, C.; Hu, H.; Kelley, S.; Park, S. *ACS Sustainable Chem. Eng.* **2018**, *6* (8), 10851−10860. (f) da Silva, R. R.; Vargas, F.; Ernst, M.; Nguyen, N. H.; Bolleddu, S.; del Rosario, K. K.; Tsunoda, S. M.; Dorrestein, P. C.; Jarmusch, A. K. Computational Removal of Undesired Mass Spectral Features Possessing Repeat Units via a Kendrick Mass Filter. *J. Am. Soc. Mass Spectrom.* **2019**, *30* (2), 268−277.

(11) Creative Commons. https://creativecommons.org/licenses/by-nc-sa/4.0/ (accessed November 2020).

(12) Open Textbook Library. https://open.umn.edu/opentextbooks/ (accessed November 2020).

(13) (a) Helmus, J. J.; Jaroniec, C. P. Nmrglue: An open source Python package for the analysis of multidimensional NMR data. *J. Biomol. NMR* **2013**, *55*, 355−367. (b) NMRglue Documentation. https://nmrglue.readthedocs.io/en/latest/index.html (accessed December 2020).

(14) Creative Commons BY-NC-SA Legal Code. https://creativecommons.org/licenses/by-nc-sa/4.0/legalcode (accessed November 2020).

(15) For information on paying for NonCommercial items printed, see: (a) U.S. Appellate Court Enforces CC's Interpretation of NonCommercial https://creativecommons.org/2020/01/07/u-s-appellate-court-enforces-ccs-interpretation-of-noncommercial/ (accessed November 2020). (b) CC Legal Database. https://certificates.creativecommons.org/cccertedu/chapter/additional-resources-4/.

(16) (a) Project Jupyter; 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. nbgrader: A Tool for Creating and Grading Assignments in the Jupyter Notebook. *JOSE* **2019**, *2* (11), 32. (b) Nbgrader Documentation. https://nbgrader.readthedocs.io/en/stable/index.html (accessed November 2020).

https://dx.doi.org/10.1021/acs.jchemed.0c01071
J. Chem. Educ. 2021, 98, 489−494