Interactive Python Notebook Modules for Chemoinformatics

Babak Mahjour[1], Andrew McGrath[1], Andrew Outlaw[1], Ruheng Zhao[1], Charles Zhang[1], Tim Cernak[1,2*]

[1]Department of Medicinal Chemistry, University of Michigan
[2]Department of Chemistry, University of Michigan

ABSTRACT: Computational informatics are quickly becoming a mainstay and expected skill of researchers. Despite this, very few STEM graduates matriculate with even the most basic formal training in programming. This lesson plan was developed to introduce undergraduates studying chemistry or biology to chemoinformatics and data science in medicinal chemistry. The course is split into two class sessions, each with an introductory slide deck, python notebook protocol, and lab report template. Over the course of the lesson plan, students learned to parse datasets with python, perform machine learning analyses, and develop interactive graphs. During each session, students completed the Python notebook protocol and fill out a lab report template after a short lecture. By the end of the lesson plan, students reported to have increased confidence in their understanding of chemistry, Python, and data science. This lesson plan continues to be used as part of a chemoinformatics unit as part of an undergraduate degree.

Introduction

Chemoinformatics is the use of computational techniques to solve problems in chemistry. These in silico methods can be used to transform data into information and aid in the process of drug discovery. Recently, a rise in computational power and increased availability of developed tools have turned chemoinformatics into an invaluable tool for research. There has been a recent growing interest in teaching young scientists how to work at the interface of physical science and data science.[1-5] This lesson plan introduces students to the most popular scripting language, Python,[6] and guides them through the basics of importing data and generating plots.

Objectives

The purpose of this experiment is to introduce chemoinformatics using Python. The modules teach the student:

- about data structures in Python
- to load compiled data that is suitable for sharing and later use
- how to effectively parse through compiled data
- how to perform mathematical operation on compiled data
- how to plot data in a multitude of ways
- how to filter out unusable data
- to use data visualization to validate medicinal chemistry principles
- to perform basic statistical analyses
- and to simplify multidimensional data using principal component analyses

Structure and Content

This lesson plan is executed over two separate class sessions. Each class session consists of a brief slide deck and lecture introducing the very basics of python and its capabilities, presented to the students, followed by an interactive Python worksheet, composed of multiple modules. These worksheets are written in Google Colaboratory (Colab), an easily accessible online Python environment that executes code on the cloud for free. Its primary advantages here are allowing a fast and simple way for new students to get started coding instantly, as it is agnostic of computer and operating system and requires no technical setup. Code is separated into blocks called 'cells', which execute independently of each other. The two Colab notebooks, administered in order over two classes, walk the students through various exercises to meet the teaching objectives. Each notebook is to be completed alongside provided lab report templates consisting of module-specific questions and discussion items.

Preliminary Lecture Content

In the first session, an initial slide deck is presented to the class explaining the increasing popularity of scripting languages and their use in the industrial market and in academia, partially as a result to the accelerating accumulation of big data. Several examples of data visualizations generated by Python are shared. Python as a scripting language is then formally introduced, as well as Colab; students are directly shown how to execute "print("hello world")" and are encouraged to login to Colab from their computer and to attempt running a line of code. Lists and dictionaries are introduced as two basic data structures. If statements and for loops are introduced through their utility in filtering a list of dictionaries. Finally, it is shown how the Python package Pandas[7] can be used to load tabulated datasets from CSV files or JSON files. The student completes Notebook 1 and its corresponding lab report template after this lecture.

The second session begins with another short lecture. In this lecture, the concept of machine-readable molecular representations is introduced through SMILES. Box plots are shown as a way to organize molecular datasets into Lipinksi or non-Lipinski groups.[8] Finally, histograms and principal component analyses are touched on as other ways to analyze distributions of molecular data. Notebook 2 and its lab report are then completed after the lecture.

Notebook 1: Introduction to Colab, Python, and Chemoinformatics

This notebook introduces the student to Google Colab, plotting in Python, and basic chemoinformatic concepts. The learning objectives of this notebook are to learn basic python coding and to quickly load and plot chemoinformatic data from spreadsheets or other data formats. The module also exemplifies how to customize plots generated in Python.

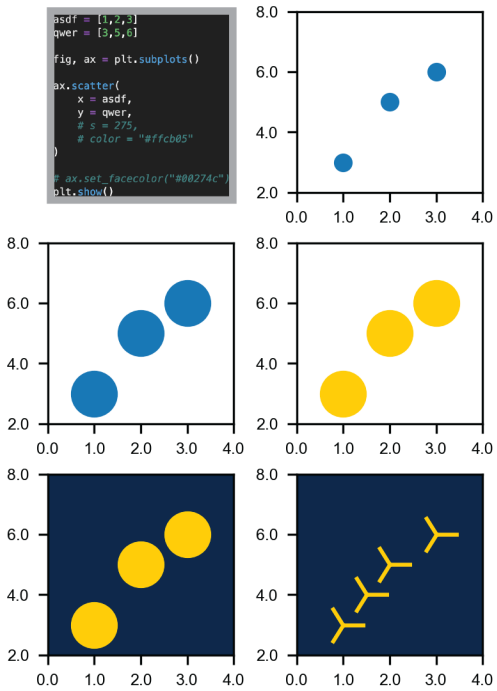

Figure 1. The code template generates several plots for the student. By commenting and uncommenting code, students can directly modify the plot without attention to syntax. This module focuses on teaching the usage of Colab through basic Python plotting.

The first module walkthrough basic plotting in Python using matplotlib. Students receive the entirety of the code written, but with several lines commented out. Students are instructed to run the script, note what happens, then uncomment a line of code and run the script again. The goal here is to teach the student to be familiar with using Colab and to show how including certain lines of code affects the scripts output.

The second module instructs the user to upload a datafile to colab, then load the data into Python with pandas. Inspecting the file reveals that it is a perfectly tabulatable JSON object, and thus can be directly read without issue as a Pandas DataFrame. A dataset of molecules with properties is provided from DrugBank.[9]

In the third module, the user is given templated plotting code to modify and generate plots. Here, physicochemical properties are introduced. Common parameters of druglikedness are included in the dataset such as LogP, polar surface area, number of aromatic rings, and hydrogen bond donors. The full list of included properties is included in the supporting information. The user is instructed to compare trends of physicochemical properties by modifying the attribute that is plotted on the x- and y- axes of the plot. The user also has the option to investigate in a third dimension by modifying the color of the plotted points.

INPUT

```
import pandas as pd

data = pd.read_json("alldrugsprops.json")
data.head()
```

OUTPUT

	SMILES
0	CC[C@H](C)[C@H](NC(=O)[C@H](CCC(O)=O)NC(=O)[C@...
1	CC(C)C[C@H](NC(=O)[C@@H](COC(C)(C)C)NC(=O)[C@H...

	LOGP	HBD	HBA	PSA	ROTB	AROM	FSP3	FC
0	-8.11643	28	29	901.57	66	3	0.540816	0
1	-3.10570	17	16	495.89	31	4	0.508475	0

Figure 2. Students are provided with the code to import any tabular JSON file. The utility of the package Pandas is used in reading tables programmatically.

Concepts such as variables and f-strings are introduced here. The user can set the color variable to a column header from the DataFrame. Through variables, the color of the plot and the title of the output figure is automatically updated.

This notebook is concluded with the following discussion questions:

1. What are the Lipinski rules?
2. Write code for filtering by drugs that pass all the Lipinski rules
3. Suggest a research question that you could ask of the DrugBank dataset

These discussion questions evaluate the student's understanding of basic druglike properties and their ability to incorporate these properties into basic Python plotting code. It is expected of the student to utilize search engines to assist in the writing of filter code to answer the second discussion question. Filters can be written manually with if statements and for loops, as explained in the preliminary presentation, or using a function included in the Pandas package.

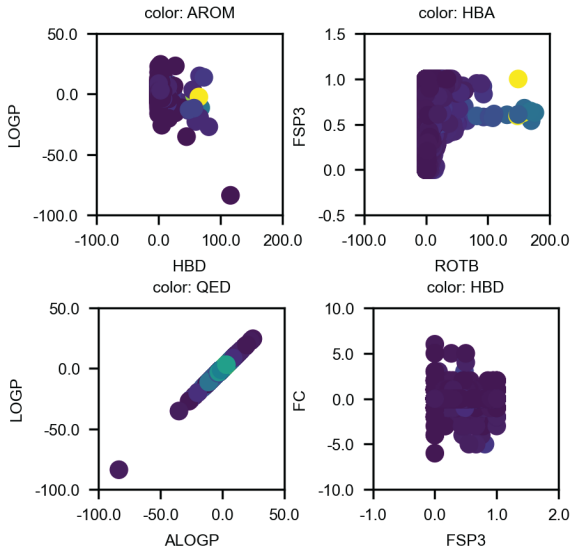

Figure 3. Four chemoinformatic experiments run by students during the first session of class. The developed script is usable with any tabulated dataset.

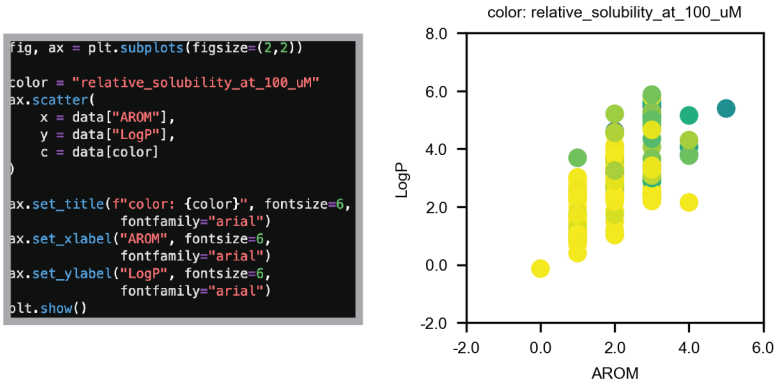

Figure 4. The template plotting code is incrementally improved until it can be used effectively to make manuscript-ready graphics. Validating GSK's Solubility Forecast Index is a simple experiment to allow the student to build confidence in their ability to manipulate and analyze datasets.

Notebook 2: Principal Component Analysis

This notebook utilizes a dataset from Diamond XChem's COVID Moonshot project.[10] Inhibition data against the SARS-COV-2 main protease alongside precalculated physicochemical properties and SMILES for various inhibitors are included in a CSV file provided to the students. The learning objectives of this notebook are to filter out unusable data, use data visualization to validate medicinal chemistry principles, perform basic statistical analyses, and simplify multidimensional data using principal component analysis.

In the first module, the student is instructed to load and inspect the csv using the Pandas package. Entries without IC_{50} values are then filtered, and the student is asked to record the number of remaining molecules in their report

In the second module, the students are asked to validate GSK's Solubility Forecast Index[11] using the filtered dataset from the previous module. The students are directed to use the template plotting code to show how solubility is affected by the number of aromatic rings and LogP. This requires the student to correctly plot certain properties from the dataset, which should be possible given an understanding of the components in the index and the script template.

In the third module, students are instructed to make box plots of various properties encoded in the datafile. Code template is provided, and students are asked to analyze and modify the code to reshape the grid of boxplots. Students are encouraged to improve the plots aesthetically and to practice modifying the data that is plotted by changing the variables.

In the fourth module, a principal component analysis is performed on the dataset. Students are instructed to create a matrix containing the Lipinski physicochemical properties for each entry in the dataset. Using template code, this matrix is fed into scikit-learn's principal component analysis decomposition algorithm, where the matrix is reduced to two dimensions. This is then plotted and colored by a property. Students are then instructed to filter out non-Lipinski compliant molecules, rerun the reduction and compare the final graphs.

In the final module, students use the filtered dataset and are introduced to a new package that allows for the creation of interactive plots. Using the template code, student use the package plotly to generate an interactive PCA that displays SMILES and other information for each plotted entry. The student is asked to record and evaluate several of these molecules from different clusters.

Participants

The participants in this study were students enrolled in a senior level undergraduate medicinal chemistry course. Nearly all participants had little to no previous coding experience at the time of the study, and these modules were their first introduction to a hands-on coding activity. This study was developed and conducted over several years of students; before, during, and after the SARS-COV-2 pandemic. Over 100 students have participated in the study.

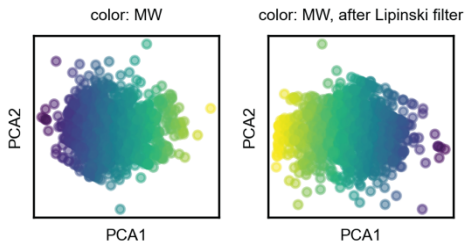

Figure 5. By the end of the modules, students have performed a filter and a principal component analysis on a dataset of SARS-COV-2 Main Protease inhibitors. This analysis is then plotted on an interactive graph, where the user can inspect datapoints with the mouse cursor.

Implementation

The initial implementation of this activity was done without Colab, with significant resources and tutorials provided to install Python on each student's personal computers. With the introduction of Colab to the academic community, the onboarding process for the activity was greatly simplified as it became guaranteed any student could complete the activities agnostic of personal hardware or software. This exercise is given to the class mid-semester, after several units introducing basic medicinal chemistry concepts are completed. The exercise is split into two lab sessions, one for each notebook. Students are provided with a short lecture and slide deck with basic coding and Python

concepts before the lab session. During the lab sessions, graduate student instructors are available for troubleshooting and questions either physically or virtually depending on the class format. Students are permitted to work in groups to complete the exercises. Generally, issues with the exercise have never been left unresolved, but special care is given to ensure answers are not provided to students.

Assessment of Effectiveness

An unrecorded survey is taken before the exercises to gauge the students' familiarity with Python and programming. As of 2023, nearly all senior undergraduates who have enrolled in the class have reported having little to no experience in any style of coding or programming at the time of the initial survey. After the lectures and lab sessions, students are asked to complete a five-question survey. Student answers are anonymized and plotted as bar charts to evaluate student sentiment regarding the learning objectives after the modules are completed and turned in.

1. This exercise improved my understanding of chemistry
2. This exercise improved my understanding of Python
3. This exercise improved my understanding of chemical space
4. This exercise improved my data science
5. I enjoyed this exercise

Based on the responses received from several years of students, we concluded that the current state of the module is effective in teaching young aspiring scientists the basics of data science and informatics in medicinal chemistry.

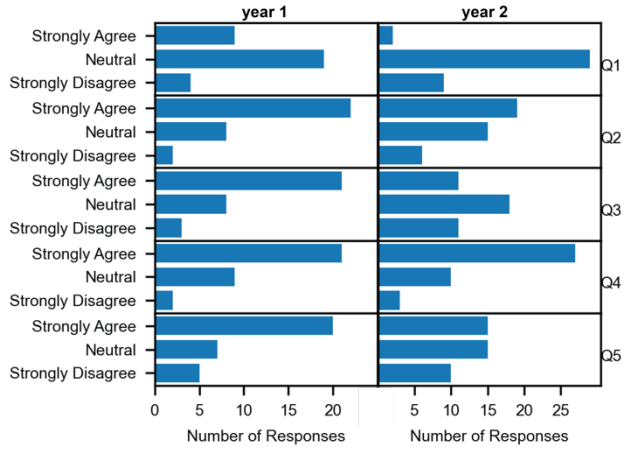

Figure 6. Student feedback to the lesson plan was positive, with improved confidence in coding, chemistry, and data science.

Summary

A lesson plan to teach undergraduates the basics of data science in medicinal chemistry was developed and validated over several semesters. Over the course of two to three class sessions, students are introduced to Python, Google Colaboratory, and several Python packages. Students learn these tools through guided, interactive modules that begin at learning how to function Colab and ends with developing a program that reads abstractable datasets and generates user-interactive data analytics through machine learning and Python. Through self-reported assessment, the lesson plan seems to be effective in improving student's familiarity with modern chemoinformatic tools and concepts. We postulate that the lesson plan may be effective for younger students as well and can be provided to anyone with a computer and internet access.

Associated Content

Supporting Information

The supporting information is available in the provided repository. All course details including the introductory slide decks, Colab notebooks, and lab report templates are provided.

Author Information

Corresponding Author

* Tim Cernak – Department of Medicinal Chemistry, College of Pharmacy, University of Michigan, Ann Arbor, MI, USA 48104; orcid.org/0000-0001-5407-0643.

Authors

Babak Mahjour – Department of Medicinal Chemistry, College of Pharmacy, University of Michigan, Ann Arbor, MI, USA 48104;

Andrew McGrath – Department of Medicinal Chemistry, College of Pharmacy, University of Michigan, Ann Arbor, MI, USA 48104;

Andrew Outlaw – Department of Medicinal Chemistry, College of Pharmacy, University of Michigan, Ann Arbor, MI, USA 48104;

Ruheng Zhao – Department of Medicinal Chemistry, College of Pharmacy, University of Michigan, Ann Arbor, MI, USA 48104;

Charles Zhang – Department of Medicinal Chemistry, College of Pharmacy, University of Michigan, Ann Arbor, MI, USA 48104;

Notes

The authors declare no competing interests.

Acknowledgements

The University of Michigan College of Pharmacy is thanked for instructional funds and guidance.

References

1 Bravenec, A. D. & Ward, K. D. (ACS Publications, 2022).

2 Lafuente, D. *et al.* A Gentle introduction to machine learning for chemists: an undergraduate workshop using python notebooks for visualization, data processing, analysis, and modeling. *Journal of Chemical Education* **98**, 2892-2898 (2021).

3 Menke, E. J. (ACS Publications, 2020).

4 van Staveren, M. Integrating Python into a Physical Chemistry Lab. *Journal of Chemical Education* **99**, 2604-2609 (2022).
5 Weiss, C. J. A creative commons textbook for teaching scientific computing to chemistry students with python and Jupyter notebooks. *Journal of Chemical Education* **98**, 489-494 (2020).
6 Virtanen, P. *et al.* SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature methods* **17**, 261-272 (2020).
7 McKinney, W. Pandas, python data analysis library. *URL http://pandas. pydata. org*, 3-15 (2015).
8 Lipinski, C. A. Lead- and drug-like compounds: the rule-of-five revolution. *Drug Discovery Today: Technologies* **1**, 337-341 (2004). https://doi.org:10.1016/j.ddtec.2004.11.007
9 Wishart, D. *et al.* Vol. 34 (Database issue):D668-72. 16381955. (Nucleic Acids Res., 2006).
10 Achdout, H. *et al.* Open Science Discovery of Oral Non-Covalent SARS-CoV-2 Main Protease Inhibitor Therapeutics (preprint). (2020).
11 Hill, A. P. & Young, R. J. Getting physical in drug discovery: a contemporary perspective on solubility and hydrophobicity. *Drug discovery today* **15**, 648-655 (2010).