Dive into computational physical chemistry

Lecture 2: Data organization and analysis

Glen Hocky
September 13, 2022

Data is your most important resource

Some best practices

1. Keep your files organized
2. Label files (and inside of files) well – don’t use default generic names
3. Have a strategy for backups
4. Track changes (see next)
5. *Take notes (some ideas)
    1. Me: make very good scripts (not a great strategy, but okay)
    2. Lab notebook? Electronic notes?
    3. Another strategy - send self information in slack
    4. gist.github.com

What is more valuable than data?

Everything you need to generate the data

- Code/software (what version if software?)
- Input data (e.g. protein structure)
- Parameter files (how should the software run)

Key questions to ask yourself every day:

- If I came back to this in a week/month/year could I repeat it?
- Could someone else in my lab repeat it?
- Could a random stranger on the internet repeat it?

Strategies for replicable *research*

- Replicable != reproducible : replicable means you can repeat it, reproducible means you can arrive to the same conclusions possibly in your own way
- Write and share open-source code as part of your project
- Publish all the inputs and outputs (sometimes ‘downsampled’) and code you use to make figures
- Use version control systems to track your work, and collaborate!

- Bonus: Check out this paper: Promoting transparency and reproducibility in enhanced molecular simulations. Nature Methods 2019. https://doi.org/10.1038/s41592-019-0506-8

Version control

- Version control systems track the evolution of your project
- You *pull* changes from a *repository, commit* your updates, and *push* them back
- Early version control systems include CVS and SVN (subversion), which are *centralized* version control systems. This means a central server has to be running
- *Git* is a decentralized version control system created in 2005 by Linus Torvalds for Linux
  - *Decentralized* means that you can push and pull from many different copies of the respository, resolving conflicts
- *Github* is a popular website with a lot of services on top of Git, which serves as a semi-centralized place (bitbucket is another)
  - However, you can still *fork* these repositories and have your own copies, and contribute back by *pull-requests*
- *Branches* let you make changes and upload them without affecting the main code

Example: https://github.com/hockyg/comp-lab-class

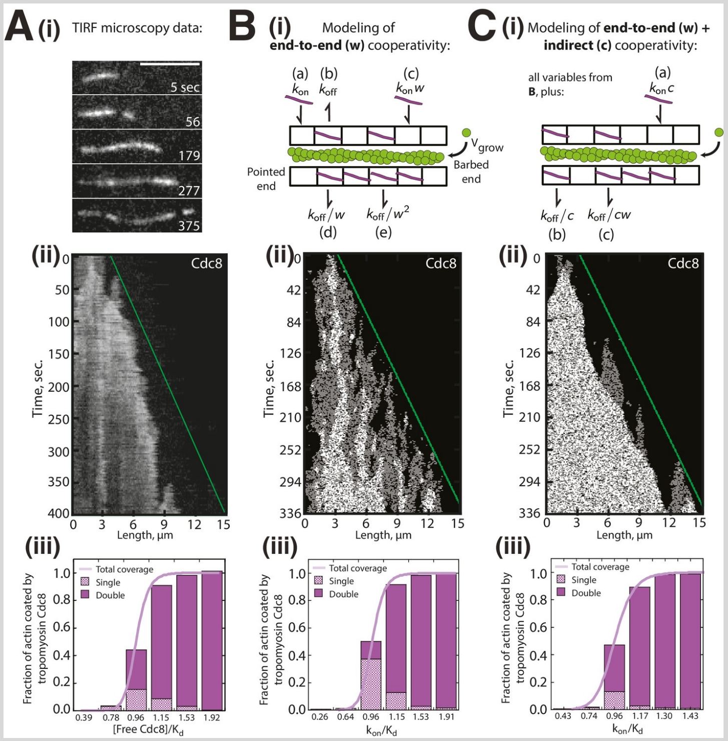
Python and libraries

- *Python* is a computer language developed by Guido van Rossum in 1991. Python 3 was released in 2008.
- Python is the most popular programming language for general computational chemistry software and data analysis
  - Python is an *interpreted* programming language, meaning it runs basically line by line
  - *Compiled* programming languages take your whole code and turn it into an executable before running – these are generally faster and used where high performance is needed (C,C++,fortran)
- Python is especially popular due to its libraries (which can be easily installed from people's individual codes or from a centralized server, e.g. Pypi.org)
- Example libraries:
  - Numpy for numerical tasks
  - Scipy, scikitlearn for scientific tasks like fitting, machine learning
  - Mdtraj and MDAnalysis for general analyzing of MD code
  - Matplotlib, seaborn for plotting

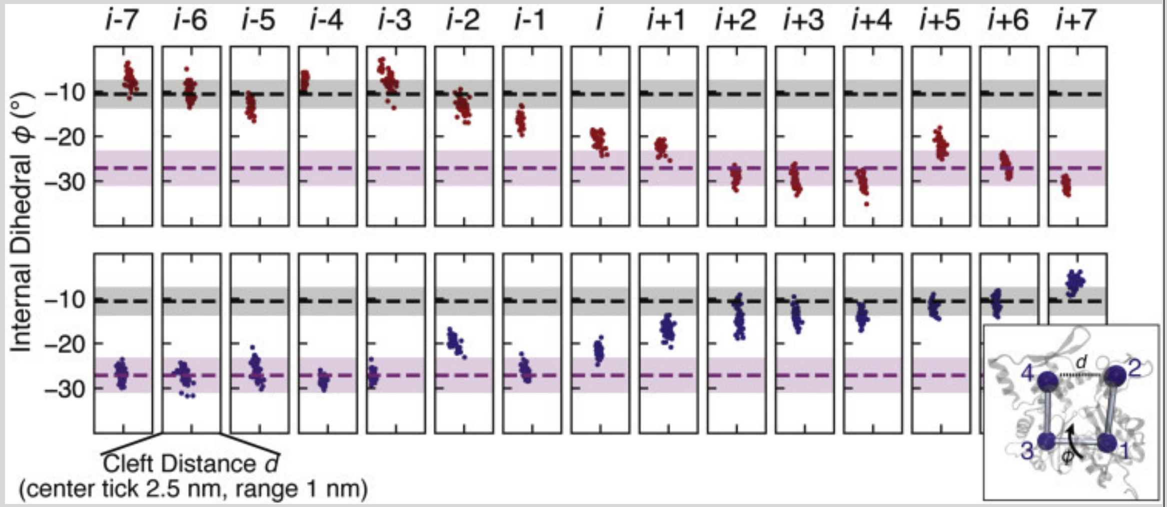
Plotting

- Making plots of data is pretty much your job
- There is a place for quick and dirty plots, which is to tell you if the simulation is running/working etc (see e.g. gnuplot)
- In this class, we will emphasize data aesthetics
    - Every axis should be labeled, including units!
    - Consider your ticks and tick-spacing
    - Consider colors, and color palette. Mix symbols with colors to help with colorblind
    - Are lines labeled, and are there legends?
    - Should the plot have a title?
        - Yes for your ongoing work and figures
        - Generally no for a paper
    - Scientifically, does your plot convey a message?
    - Are the font, size ratio appropriate for your current task (paper, poster, talk)

Examples that I made



https://elifesciences.org/articles/23152



https://www.jbc.org/article/S0021-9258(21)00108-3/fulltext

Jupyter notebooks

- Jupyter notebooks are an interactive environment for running python code
- They are especially nice for making plots, because the plots are embedded in the notebooks
  - Results of analysis can be directly visualized
- There are downsides!
  - See e.g. this article, although these can be avoided https://towardsdatascience.com/5-reasons-why-jupyter-notebooks-suck-4dc201e27086
- The primary problem is the nonlinear order of running code, so you might think something is working but it would not run if you did it again (Restart and run all is best practice to make sure)
- Best for exploration, in production, best to produce analysis scripts that produce data files (txt, database, pickle) that can be loaded and plotted with another script or notebook
- Also very good for creating tutorials and walk throughs of what you did

Combining copilot with running on greene

- This may take some setup to get working. However, when you get it working, it will greatly accelerate your work
- I will show an example now
- Here are instructions for doing remote vscode on greene:
  - https://sites.google.com/nyu.edu/nyu-hpc/training-support/general-hpc-topics/vs-code
- For today, maybe use vscode + codex to help write scripts, run them on command line to produce text files, and use ood jupyter to run the actual commands
  - I tried to set up a uniform environment for us. To install, try this:
  - cd /scratch/work/courses/CHEM-GA-2671-2022fa/jupyter-setup; bash install_kernel.sh

Today

- Do git crash course
- Clone class github
    - git clone https://github.com/hockyg/comp-lab-class
- Analyze energy data and structural data from a real MD simulation
- Have figures in jupyter notbook
- Push jupyter notebook to your cloned github
- Put your github url for this class in a spreadsheet

Next time

- What are molecular dynamics simulations
- Running batch jobs and interactive jobs on HPC
- Activity: setting up and running your own MD simulations in Gromacs