Dive into computational physical chemistry

Lecture 2: Data organization and analysis

Glen Hocky
September 12, 2024

Reminder: 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

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)

Data analysis

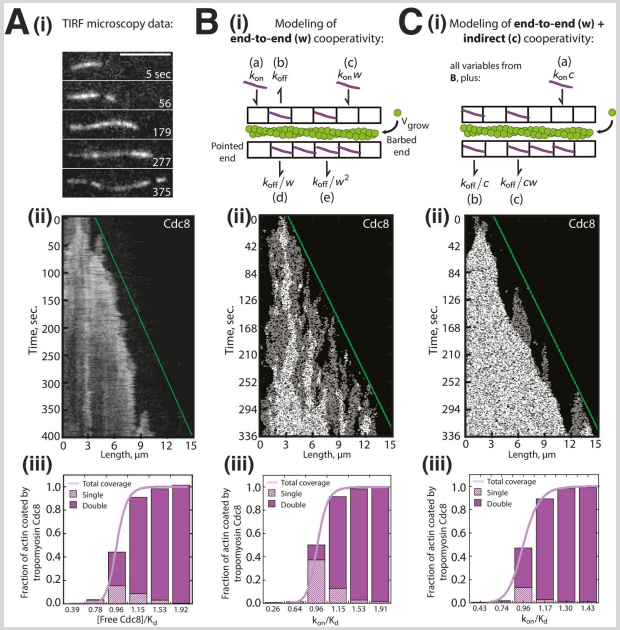
For many projects, generating data is easy, what is the hard part in that case?

- Finding something interesting in the noise
- Connecting results to underlying physical principles
- Presenting the data in a way that someone can rapidly understand your findings
- This procedure is also critical to finding mistakes!

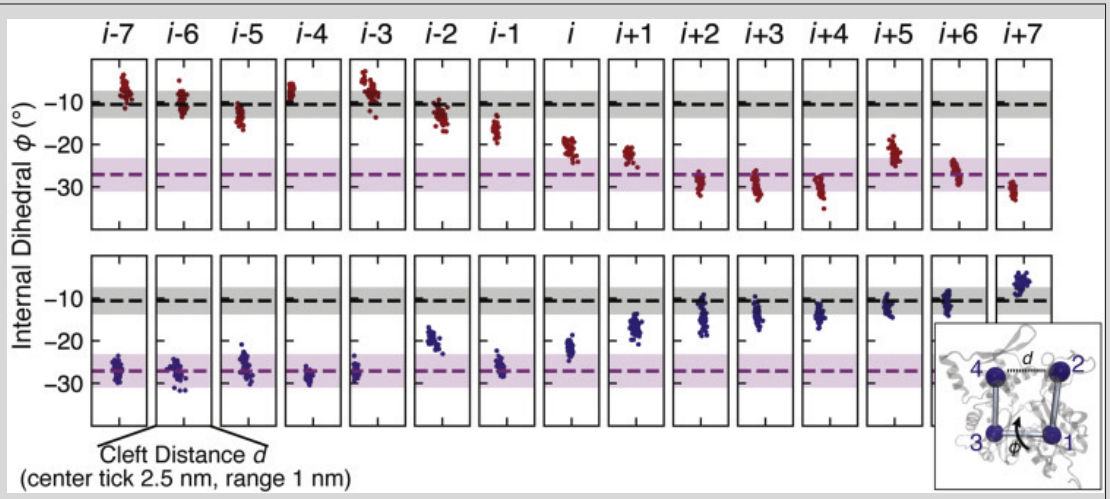
Plotting

- 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)

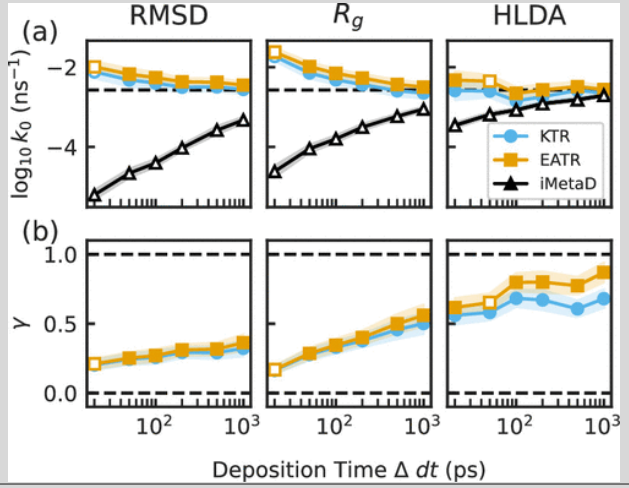
Examples from me



https://elifesciences.org/articles/23152



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



https://pubs.acs.org/doi/10.1021/acs.jctc.4c00425

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 results
  - Matplotlib, seaborn for plotting

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

Visualization of molecular data

- The output of molecular dynamics simulations or electronic structure calculations is a set of atomic positions with associated data (such as electron density in the case of ES)
- Making pictures of this data graphically is crucial to understanding your system
- We will first take the example of biomolecular data – biomolecular structures are 'solved' by experiment and then deposited in the 'Protein Data Bank', using the PDB file format.
  - This format gives the positions of the atoms, but also the type of amino acids or other molecules that could be included, and information about the experiment
  - PDB files can be parsed by software (e.g. in python) for analysis, and they can also be visualized by many softwares such as VMD, PyMol, Chimera, Molstar, etc.
- PDB file can also be interpreted to give the molecular topology (bonds)
- MD Data is in the form of 'trajectories' – this is xyz vs time, and has other 'binary' formats
- We will try using VMD to visualize MD output

Today

- Fork github if you didn't already, then clone your personal github
  - git clone https://github.com/YOURID/comp-lab-class-2024
- Try VMD as in the instructions
- If you need – do a python tutorial such as https://education.molssi.org/python_scripting_cms/
- Analyze energy data and structural data from a real MD simulation
- Have figures in jupyter notbook
- Push jupyter notebook to your cloned github

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