Dive into computational physical chemistry

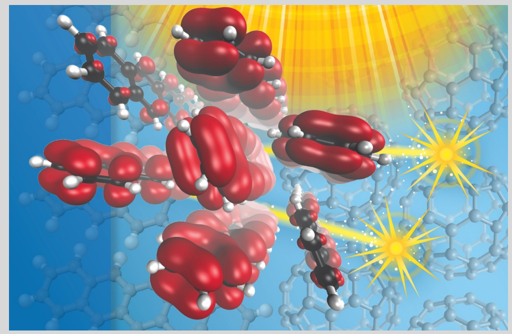
Lecture 1: Introduction

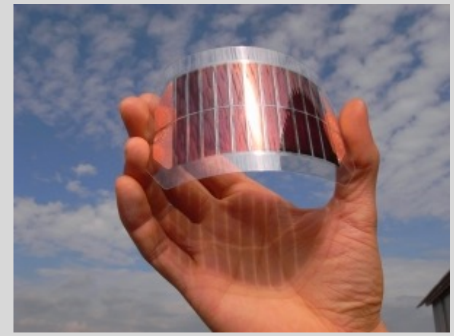
Glen Hocky
September 5, 2024

Two kinds of theoretical chemistry

Quantum Mechanics

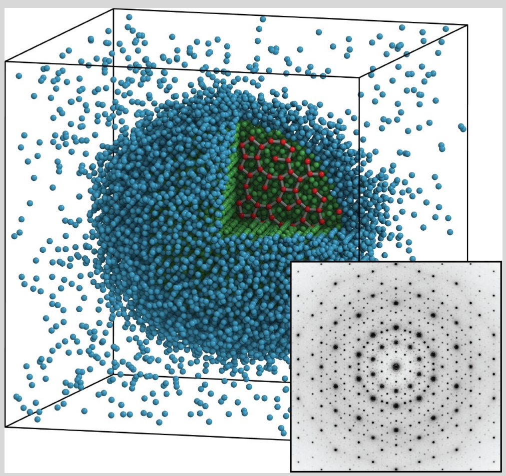
What is the behavior of the electrons?

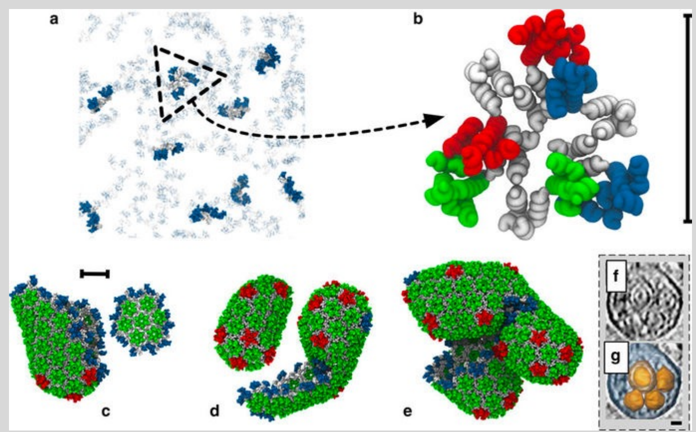




Statistical Mechanics

How do large collections of molecules behave?

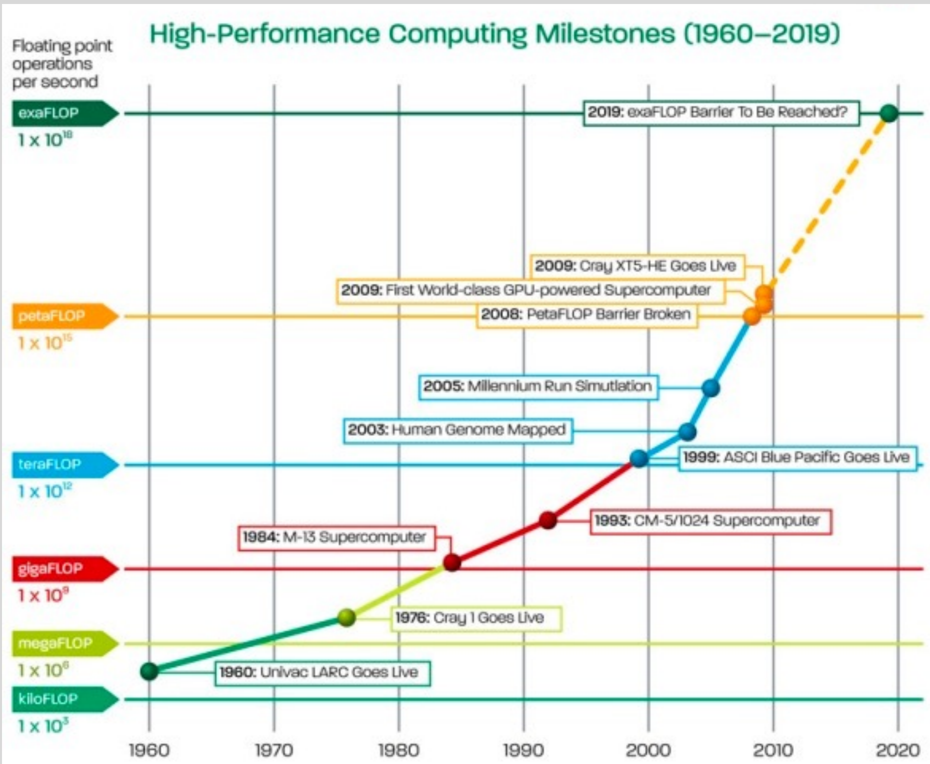


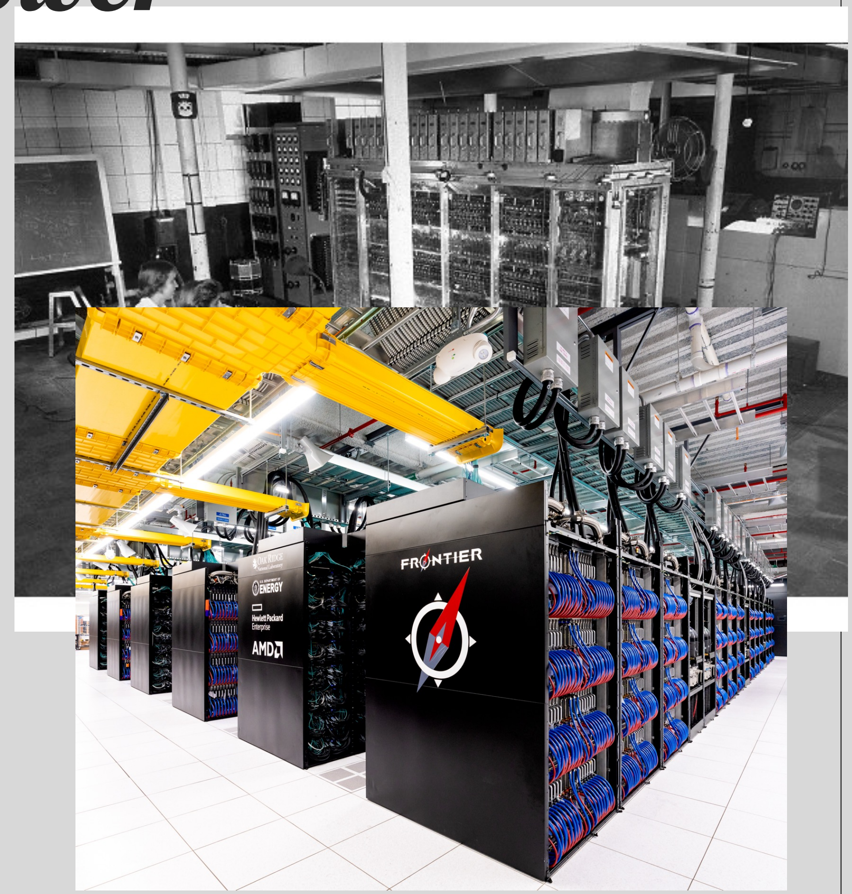


Need for computers

- Equations of quantum mechanics and of statistical mechanics are too complicated to solve by hand in most cases
- Used to make the most approximation that seemed reasonable, then sometimes use computers as calculators
- Computers first applied in chemistry during the Manhattan Project to predict nuclear properties

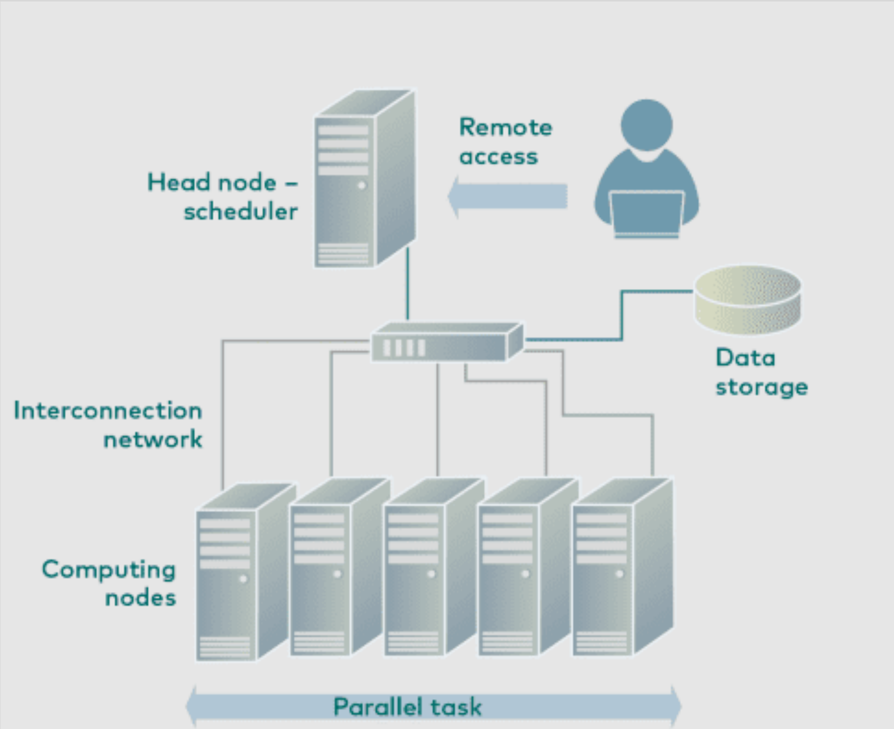
Advances in computing power





Frontier, ORNL, 1.01 exaflop. May 2022

High performance computing



NYU Greene (2024)

- The total number of nodes is 672
  - 6 login
  - 745 compute nodes
    - 524 Standard memory (180 GB)
    - 50 Medium memory (360 GB)
    - 4 Large memory (3,014 GB)
    - 73 GPU RTX8000
    - 11 GPU V100
    - 45 GPU A100
    - 15 GPU H100
  - 6 administrative
- The total number of CPU cores is 39256
- The total number of GPU cards is 768
- The total memory is 239 TB

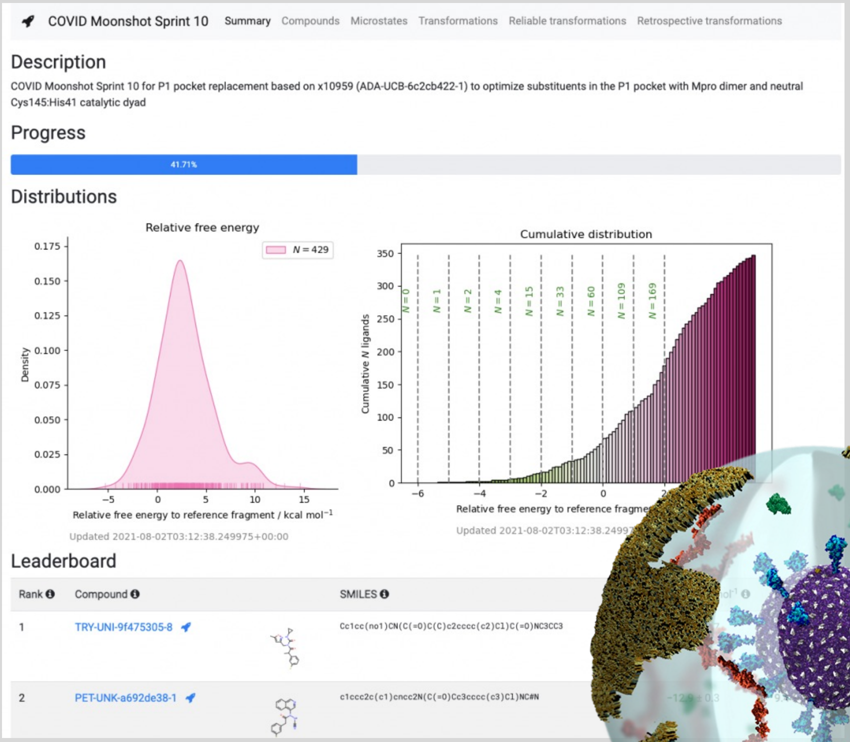
https://sites.google.com/nyu.edu/nyu-hpc/home

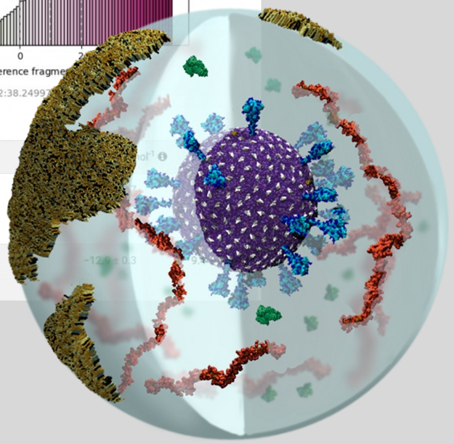
Different types of parallel computing

- Trivial/many task
- Tightly coupled, requires communication (e.g. MPI)
- Shared memory – OpenMP, GPU

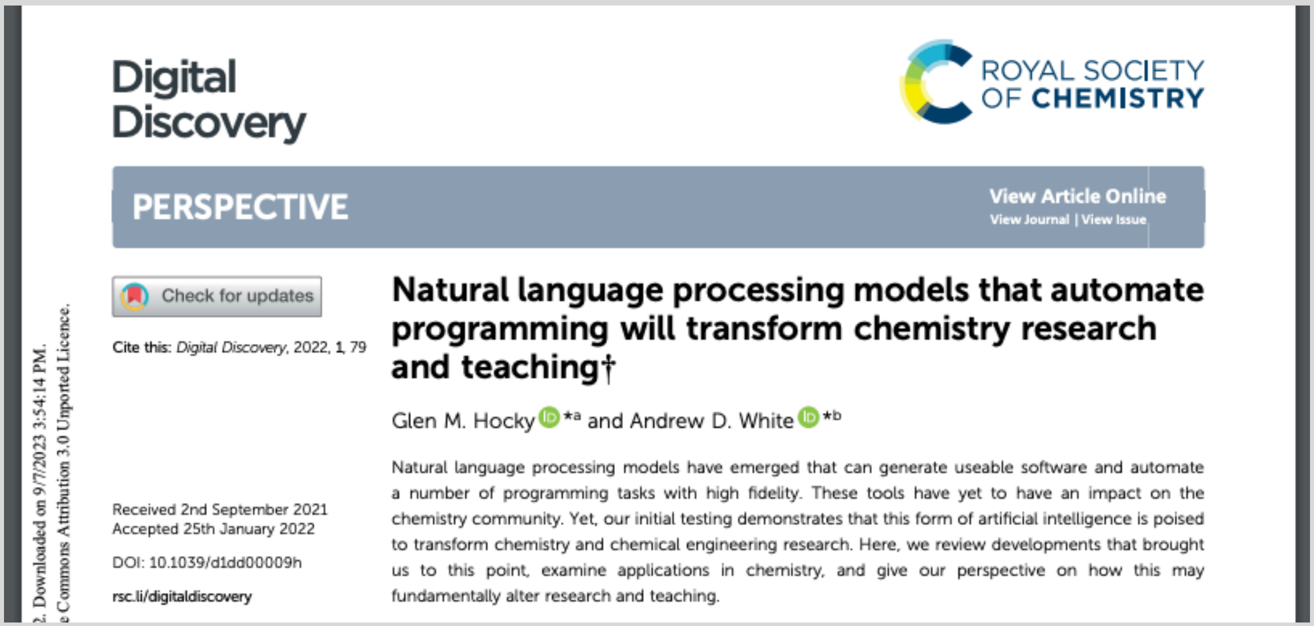
Typically we will do hybrid, many tasks each of which are accelerated through parallel process (discussed in a later lab)

Example use of computational chemistry





Solving programming problems with AI



Digital Discovery

ROYAL SOCIETY OF CHEMISTRY

PERSPECTIVE

View Article Online
View Journal | View Issue

Check for updates

Cite this: *Digital Discovery*, 2022, 1, 79

Natural language processing models that automate programming will transform chemistry research and teaching†

Glen M. Hocky *a and Andrew D. White *b

Natural language processing models have emerged that can generate useable software and automate a number of programming tasks with high fidelity. These tools have yet to have an impact on the chemistry community. Yet, our initial testing demonstrates that this form of artificial intelligence is poised to transform chemistry and chemical engineering research. Here, we review developments that brought us to this point, examine applications in chemistry, and give our perspective on how this may fundamentally alter research and teaching.

Received 2nd September 2021
Accepted 25th January 2022

DOI: 10.1039/d1dd00009h

rsc.li/digitaldiscovery

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 repository, 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-2024

Class logistics

Discussion of syllabus:

https://hockygroup.com/teaching/comp/syllabus/ComputationalLab_GA2671_Syllabus_2024_draft.pdf

Discussion of slack:

https://nyu-chem-ga-2671-2024.slack.com/

Today:

- The BASH shell and linux file system
  - Directory trees, relative directories, links [make your own directory in class /scratch/work space]
  - Moving and creating commands, eg cd, ls, pwd, mkdir
  - Man pages
  - Modules
- Secure shell (ssh) introduction, log in to greene on the command line
- https://ood.hpc.nyu.edu – running a jupyter notebook or interactive
- Text editing on the command line (VIM, emacs, nano)
- Taking a quick look at chatGPT etc
- What is git/github? A quick introduction to version control. A tour around an example project
- Example chemistry software - VMD