Lab 0: Getting Started with Software and SeaWulf
CHE 525/PHY 567, Stony Brook University

1 Introduction

There are many software packages available for performing Quantum Chemistry calculations and visualizing the results. In this first lab, you will get familiar with running the software we will use in this course on Seawulf, Stony Brook's high-performance computing cluster. First let us introduce some terminology about different software you will use in this course:

1. **Python**: A powerful general-purpose program language with an emphasis on ease of use and readability. In recent years, the use of Python in scientific computing has increased dramatically due to the emergence of powerful and well supported modules, such as NumPy, SciPy, Matplotlib, and many more. Furthermore, the Anaconda distribution (https://www.anaconda.com/products/individual) enables fairly easy installation, and all of this software is available at the virtual SINC site.

2. **Psi4:** An open-source quantum chemistry software package that has evolved to include an elegant application programmer interface (API) to Python, such that running a quantum chemistry calculation and parsing the results can be done using simple commands from a Python script.

3. **Molden:** A computer program for assembling molecular structures and visualizing the results of quantum chemistry calculations.

4. **Git:** A version control system that also allows for easy distribution of version-controlled code to many users.

In this first lab, you will get familiar with getting code on the Seawulf system, running, it and visualizing the results.

2 Logging in to Seawulf and cloning the course Git repository

Many of the calculations in this class are not terribly computationally intensive and in principle can be done on your own computer. Since all of the software used in this course is open-source and freely available, you are welcome to try and install the software and run calculations on your own computer. For longer jobs, this can be helpful for quick testing of code before running the full calculation on the cluster. However, we will only be providing technical support for running calculations on Seawulf in the `psifour` environment we have established there. Furthermore, using the cluster will be essential for longer jobs. Any software you will need to use in this course has been installed either on Seawulf or is available on the virtual SINC site.

Here we give some step by step instructions, written as if you are logged in you are using the virtual SINC site. Much of the information in this section, with more detail, can also be found at the websites below:

```
https://it.stonybrook.edu/help/kb/logging-into-seawulf
https://it.stonybrook.edu/help/kb/getting-started-guide
```

If you are unfamiliar with working in a Linux environment, we encourage you to work through these two websites in detail.
To log in to the in to the Seawulf cluster from the virtual sinc site, first launch a new terminal with MobaXterm and then enter the command below at the terminal prompt:

```
ssh -X <your netID>@login.seawulf.stonybrook.edu
```

where you put your normal Stony Brook netID where it says `<your netID>`. You will be prompted for your password and may be asked to do further two-factor authentification. Once in the system, you will be on the login node, and your prompt should look like `<your netID>@login`. From the login node, you can do basic file i/o and other tasks that are not computationally intensive, but the login node should not be used for any calculations. We will discuss requesting a job on a "compute node" in the following section.

First, let's download some code to your home folder on Seawulf. The two commands below start the Git version control software and clone the sbu-che525-ss23 repository into your home folder

```
module load git
git clone https://github.com/blevine37/sbu-che525-ss23.git
```

If you type the unix command `ls`, you should now see a new folder called 'sbu-che525-ss23' in your home directory. Navigate to it with

```
cd sbu-che525-ss23
```

You can now see the full path of the directory you are in via the unix command `pwd`. Once you are successfully in the sbu-che525 directory, next navigate to the directory 'Lab0' with the command

```
cd Lab0
```

If you run the `ls` command in this directory, you should see a copy of the PDF you are reading now and also a file called 'Lab0starter.py.'

3 Running a calculation and visualizing the results

Next, we will walk through running a quantum chemistry calculation on Seawulf and interacting with the results. For this task, you will want to request a job from the slurm scheduler. To do this, enter the commands

```
module load slurm
salloc -N 1 -p short-28core
ssh -X $SLURM_NODELIST
```

The last command might take a few moments to execute. If it doesn't work try it again. Now you are working on a compute node you've been allocated by the slurm scheduler. You may have to navigate back to the `../sbu-che525-ss23/Lab0` directory, as you did before. Next let's load the Psi4 environment and Molden:

```
module load psifour
module load molden/6.2
```

The `/6.2` is important since simply `load molden` will initialize a version of molden that cannot read .molden files output by Psi4. Now you are ready to run a calculation. Cloning the sbu-che525-ss23 git repository should have downloaded several files into the newly created sbu-che525-ss23. You can see the files that are there via the `ls` command. Before running a calculation, let's first start by having a peek at the Python code, `Lab0starter.py`, that you will run. You can do this several ways. Here are two:

1. Open the code with the terminal-based text editor Emacs using the command `emacs Lab0starter.py`. Emacs takes a little getting used to, but is a powerful text editor that also runs very fast from the terminal. It pays to learn how to use it to quickly develop or modify code from the terminal. There are many online resources to learn how to navigate Emacs, save files, etc. To leave Emacs, use Ctrl-X followed by Ctrl-C.

2. Open the code with the GUI integrated development environment (IDE) Spyder using the command `spyder Lab0starter.py`. Spyder is a MATLAB-like IDE that lets you view plots, see the values of variables, and enter snippets of code into an IPython terminal. This can be very useful for debugging.

The code you are looking at runs a Hartree-Fock calculation for ethene at the geometry provided by the input Z matrix and then optimizes this geometry (i.e. finds a minimum in the potential energy surface), again at the Hartree-Fock level of theory. It saves the results in several output files. However you choose to look at the code, once you are done, close the editor so you are back at the terminal prompt. Then run the command

```
python Lab0starter.py
```

You should see some action in the terminal as Psi4 springs to life and starts doing your calculation. After the code is done running, the command prompt should return. Once you see this run the `ls` command to see the files in the directory. You should see three output files, `ethene-scf.dat`, `ethene-scf.molden`, and `timer.dat`. The `.dat` files contain details about the calculations that Psi4 has done and the time it took to do them. The `.molden` was generated by line 48 of the `Lab0starter.py` code to output some of the data in a format that Molden can read.

First have a look at the output file in a text editor, for example with `emacs ethene-scf.dat`. This file contains a log of *everything* that was done in the calculation, in gory detail. Notice that the in the Python code you looked at it, we boiled down the results of this calculation to just a few outputs of method calls, such as `E_0` and `wfn_opt`. This is the beauty of using Psi4's API for Python. You don't usually have to go in to the raw output file to extract the information you need from the calculation, since they are conveniently returned to you as Python objects. That said, the output file has *everything* in it. For example, it might be instructive to find the converged Hartree-Fock energy of the initially un-optimized geometry in the output file. This was returned to the variable `E_0` on line 45 of `Lab0starter.py` but can also be found in the output file. As you can see, the output file has much more information about this calculation. For example, can you find the energy (in Hartrees) calculated on the first iteration of the self-consistent field calculation?

Next let's look at the information in the `wfn_opt` object written to the `.molden` file. Close your text editor and open the `.molden` file with the command

```
molden ethene-scf.molden
```

The molden program should launch and you should see a stick representation of an ethene molecule. Explore the different ways you can visualize things in Molden. For example, you can explore the occupied and unoccupied orbitals with various visualizations. First click 'Dens. Mode.' For a three-dimensional visualization of the HOMO, first click 'Space.' Enter '0.05' and click 'Apply'. This sets the 'iso' value; the region in which the absolute value of the electronic wave function is greater than the iso value will be plotted. Now, click 'Homo.' You may rotate the molecule to see it from different angles by clicking in the main Molden window. Investigate how the choice of iso value changes the representation of the orbital by clicking 'Space' again and entering different numbers. Also, look at other orbitals by clicking 'orbital' and selecting different orbitals from the list. For example, you may view the LUMO by clicking the first orbital that has an 'Occupation' of 0.00 (that is, that contains no electrons). You can view a core orbital by clicking the orbital with the smallest 'Eigenvalue' (which corresponds to the orbital energy).

4 Getting files on and off the server

You've already downloaded some files to the server using git. You will also likely want to be able to transfer files directly from your computer (or the virtual SINC site) to the server or vice versa. There are several ways to do this. From MobaX, the easiest way is probably to open a new session and use sftp. Detailed instructions, including tips for avoiding problems with excessive two factor identification, can be found here:

`https://it.stonybrook.edu/help/kb/transferring-files-to-and-from-seawulf`

One way or another, download your output files to your local machine so you can see how it works. You may also want to download the `Lab0starter.py` if you want to see how you can open this with local editors that display Python code nicely. However, it will not run without Psi4 installed locally.

5 Deliverables

The goal of this first lab is to get you up and running on Seawulf. Thus, a formal lab report is not due for this exercise. Instead, turn in a PDF showing a screen shot of the HOMO of ethene visualized with Molden (with whatever scheme you like), and also the part of the output file generated by Psi4 that shows the self-consistent field calculation at the initial geometry.