Full CI starter project

This is a brief tutorial to write a full configuration interaction (FCI) code.

Originally written by Qiming Sun.

The FCI problem can be solved step by step:

1. Get familiar with Python, and the numpy/scipy libraries. If you have no experience, you can read this numpy doc `https://docs.scipy.org/doc/numpy/reference/`.

   If you need install Python and all the relevant libraries, we highly recommend the Anaconda package.

   Time: $\sim$ 1 week if you don't have any experience with Python

2. Use the Slater-Condon rule to evaluate the Hamiltonian in the CI basis, then directly diagonalize it using numpy. You can use the attached 1-electron and 2-electron integrals (for H6 chain, STO-3G basis, 6 orbital, 6 electrons) to do this step (the files are called h1e.npy and h2e.npy). The reference energy is -7.8399080148963369.

   Time: $\sim$ 2 weeks

3. Install PySCF and go through the tutorial `https://sunqm.github.io/pyscf/tutorial.html` to get familiar with the package, so that you can access the integrals for other systems as you want.

   Time: $<$ 1 week

4. For a faster and more technical implementation, you can move on to the so-called "string-based" determinant-CI (or direct-CI) algorithm. It is more of a challenge.

5. Read about the Davidson algorithm for matrix diagonalization and implement it for any symmetric matrix, e.g. the Hamiltonian you built up in step 2 for the H6 system.

   Time: this may take time, perhaps 1 to 2 weeks

6. Read Knowles and Handy's paper (1984). This is the most difficult step in this project. You may need to read a lot of relevant literature to get familiar with concepts like second quantization, the physical vacuum vs HF vacuum, etc.

   Time: perhaps 2 - 3 weeks or more

7. To program direct-CI, you should first implement functions to handle the FCI string, then the H*C operation. Your original implementation using numpy diagonalization will be a great aid in debugging the program.

   Time: $\sim$ 1 week once you fully understand step 6.

8. Use the FCI strings to evaluate the Hamiltonian matrix elements. This offers a good check for your step 2 and step 7.

   Time: 3 days

9. Put everything together: the Davidson diagonalization solver, preconditioner (can be taken from your program in the previsou step), H*C operation, and debug and tests.

   Time: 3 days.

Congratulations! The FCI project is finished! This project should take 5 to 10 weeks in total if progress moves smoothly as expected. If your programs are completely written in Python, it should be able to solve problem with a maximum system size of $\sim$ 12 orbitals with 12 electrons. If you have interest in extending the code to larger systems, you have some different possibilities.

One is to optimize the FCI code, e.g. using Cython to compile some piece of your code, or rewrite some code in C to get the best performance. If you just want to spend a little time to understand how this optimization can happen, you can consider reading the FCI module in PySCF which is a very efficient implementation.

The second choice is to read and implement some flavor of select-CI from recent work. This might be able to handle $\sim$ 5 more orbitals based on your Python FCI program.