FYS-4096 Computational Physics: Exercise 10

Ilkka Kylänpää

Week 13

General:

- You should have access to `ex10_help` on teacher's repo.
- There you will find, e.g., files `pimc_simple.py`, and `graphene.py` that will most likely be useful in this exercise.
- For general future use the QMCPACK website might become useful for QMCPACK and Nexus related details.
- If you have not successfully compiled the Quantum Espresso and QMCPACK codes yet, the teacher should be able to help. Contact either via email or in Teams.
- **Due to holidays the deadline will be before noon on Monday April 12**

Problem 1: Introduction to Path integral Monte Carlo (PIMC) code and theory

a Run the simple PIMC code: `python3 pimc_simple.py`. You should get an energy somewhat close to 3 (in Hartree units). What system did you solve? What is the temperature? ($k_\mathrm{B} = 3.16681 \times 10^{-6}$ Hartree/Kelvin)

b Look into the code `pimc_simple.py` in more detail and make comments to identify details appearing on the lecture notes. Try to understand the details!

Problem 2: Applying and modifying a PIMC code

a Use the PIMC code `pimc_simple.py` to solve

- two electron harmonic quantum dot with $m = \omega = 1$.
- the Hydrogen molecule at internuclear distance of $1.4a_0$. For the electron-nuclei interaction use a "cumulant" approximation, i.e., $-Z\mathrm{erf}(r/\sqrt{2}\sigma)/r$, where r is the electron-ion distance, Z is the nuclear charge and σ is a smoothing parameter, e.g., $\sigma = 0.05$.
- **For one of the above cases (at least)** perform a not-too-thorough time step extrapolation (two to three points). Notice that you need to keep the temperature fixed! Also, feel free to use rather high temperatures in order to run smoothly.

b Make the bisection move / sampling part of the code as a function. Then add a function that moves the particle using a uniform distribution function. Compare how the two different ways to sample affect the simulation. Also, test that the kinetic part becomes sampled exactly with the bisection moves

Problem 3: PIMC for 2D many-electron semiconductor quantum dot

- In the help files there is a folder called `rpimc_2d_dot`.
- Copy all the files (and the test case folder) to Puhti at CSC.
- Compile the code to `/projappl/project.../student.../rpimc_2d_dot` by moving the *.f90 files and the Makefile to the specific folder.
- If you have the default modules loaded (no own module related modifications in .bashrc etc.) then you should be good to go and type `make`. This will create an executable called pimcf90.
- Put the path to pimcf90 in your .bashrc
- Move the `test_case` folder into your working directory.
- Run the test case be the command `sbatch parallel_run.scr`
  - You might need to add something into the run script.
  - After that it will run a short PIMC run for 2d semiconductor quantum dot with six electrons in state $S = 0$.
  - The semiconductor dot parameters are $m_e = 0.067$, $\epsilon = 12.4$, and $\hbar\omega = 11.857$ meV.
- Now, make a new directory for calculating the $S = 3$ state. Look into the init.f90 file and the `read_INPUT` function in it. Try to see how it sets up particle species. The other parameters in the INPUT file can remain the same, but instead of one species of 6 electrons you should end up with two species with 3 electrons in both, right. Notice that the spin variable actually does nothing, belonging to the same species group defines whether particles are identical or not.
- The electron densities are given as outputs in the `ag_density.dat` files (gnuplot format), and in general, the *.dat* files consist (block averages of) different observables.
- Plot the electron densities using Python for both of the cases. Also, calculate the total energy (remember the errorbar and equilibration).

Problem 4: Graphene with variational and diffusion Monte Carlo

- Do this also at CSC in your work directory.
- Try to get the graphene example working at Puhti. The example workflow can be found at your qmcpack folder qmcpack/nexus/examples/qmcpack/rsqmc_misch/
- Instead of copying the graphene.py from there, use the one at teacher's repo.
- You also need to load the python-env in order to use python.
- First, set the generate_only to 1 in the graphene.py file, if not set already. After that you need to set/change account, machine, jobs, and location of pseudopotentials. You should copy the pseudopotentials to some location at your work directory. You can find them based on the example file.
- Once you are confident the graphene.py should be fine, test the workflow (be sure to have generate_only=1). If the workflow goes through fine, run `./graphene.py --status_only` and look into the output.
- After that submit the runs!
- Report the energy that should be given after the workflow has been completed.
- Include the python files to your repository, that is, do not include all the outputs from the runs.

Returning your exercise

1. Make a new folder “exercise10” to your existing Computational Physics repo.
2. Create a file solvedProblems.txt in the “exercise10” folder. Inside it, write a comma separated list of problems you have solved, e.g., 1,2,3.
3. Make sure all your source files **and problem related figures** are under version control and push them to GitLab.
4. Push your commits to GitLab **before noon on Monday April 12**:
   ```
   git push --all && git push --tags
   ```