LTTC-QMC: Documentation

Matteo Walder

1 Introduction: Quantum Monte Carlo Method

This program makes use of the Quantum Monte Carlo (QMC) method to evaluate the energy of simple one- or two-electron chemical systems. The Monte Carlo method more generally speaking is a way of exploring a probability distribution or solving integrals based on random sampling rather than the more classical grid methods. Any mean value of a quantity f defined over a probability distribution $P(x)$ can be written as:

$$\langle f \rangle = \int_{-\infty}^{\infty} f(x)P(x)dx. \tag{1}$$

This integral can be solved numerically if one is able to make a random sampling of x_i configurations distributed according to $P(x)$:

$$\langle f \rangle \approx \frac{1}{N}\sum_{i}^{N} f(x_i) \tag{2}$$

where N is the number of generated configurations.
QMC exploits the fact that the expectation value of an operator $\mathcal{O}$ can be written as:

$$\frac{\langle \Psi|\mathcal{O}|\Psi\rangle}{\langle \Psi|\Psi\rangle} = \frac{\int \Psi^*(\mathbf{r})\mathcal{O}\Psi(\mathbf{r})d\mathbf{r}}{\int |\Psi(\mathbf{r})|^2 d\mathbf{r}} = \int \frac{\mathcal{O}\Psi(\mathbf{r})}{\Psi(\mathbf{r})}\frac{|\Psi(\mathbf{r})|^2}{\int |\Psi(\mathbf{r})|^2 d\mathbf{r}} d\mathbf{r}$$

Renaming the factors within the integral as:

$$o_L = \frac{\mathcal{O}\Psi(\mathbf{r})}{\Psi(\mathbf{r})} \tag{3}$$

$$\Pi(\mathbf{r}) = \frac{|\Psi(\mathbf{r})|^2}{\int |\Psi(\mathbf{r})|^2 d\mathbf{r}} \tag{4}$$

we obtain that the expectation value of the operator $\mathcal{O}$ can be written as the average over $\Pi(\mathbf{r})$ of the local operator o_L:

$$\frac{\langle \Psi|\mathcal{O}|\Psi\rangle}{\langle \Psi|\Psi\rangle} = \int o_L \Pi(\mathbf{r}) d\mathbf{r} \equiv \langle o_L \rangle \tag{5}$$

and then evaluated by the Monte Carlo method. The most common case is that of energy for which energy is evaluated by averaging over the local energy e_L.
There are several flavors of QMC, the two most famous being Variational Monte Carlo (VMC) in which the variational method is exploited and the second is Diffusion Monte Carlo (DMC) which is instead a projection method. In this particular code, the former is used and for the latter a variant called Pure Diffusion Monte Carlo (PDMC)is implemented. VMC evaluates the local energy of an appropriately parameterized $\tilde{\Psi}$ test wavefunction and attempts to minimize the energy by varying these parameters.

In contrast, the DMC method operates in a completely different way because it aims to find the true wave function of the system by considering the time-dependent Schrodinger equation:

$$i\frac{\partial\Psi(\mathbf{r},t)}{\partial t} = (\mathcal{H} - E_{\text{ref}})\Psi(\mathbf{r},t). \tag{6}$$

By expanding the wave function in the basis of eigenfunctions $\Phi_n(\mathbf{r})$ and changing the time variable to $\tau = it$ we can rewrite the wave function as:

$$\psi(\mathbf{r},\tau) = \sum_n c_n e^{-\tau(E_n - E_{\text{ref}})}\Phi_n(\mathbf{r}) \tag{7}$$

Simulating the time-dependent Scrodinger equation for sufficiently large τ will converge to exactly $\Phi_0(\mathbf{r})$. The simulation is done by considering eq.6 as a diffusion equation for the kinetic component that can be solved by the monte carlo method, while for the potential component one can operate with a branching process (DMC) or cumulate a statistical weight along the trajectory (PDMC).

In this code it allows testing on small “bosonic” systems the an ansatz of the ionic type, that is, in which the electrons are both considered to belong to one nucleus:

$$\tilde{\Psi}^a(\mathbf{r}_1,\mathbf{r}_2) = \sum_{\mu}^{Nn} \psi_\mu^a(\mathbf{r}_1)\psi_\mu^a(\mathbf{r}_2) \tag{8}$$

where Nn is the total number of nuclei, the function ψ_μ is an exponential parameterized with respect to a and the electron is centered on the nucleus μ:

$$\psi_\mu^a(\mathbf{r}) = e^{-a|\mathbf{r}-\mathbf{r}_\mu|}. \tag{9}$$

A single VMC cycle is inplemented in the code, for which the a parameter must be set manually, and the PDMC method built from the VMC code by adding the porcess of accumulating statistical weights during the projection time.

2 User Documentation

Here the different components of the code are explained, how the input file is constructed, and what kind of parameters can be passed.

2.1 Input File

Input data are read from the file **qmc.inp**, and parameters within it can be changed to do different simulations.

```
dif             # var (vmc) or dif (pdmc)
1.2             # Wavefunction Parameter ‘a’
2               # n of Electrons
[name].xyz      # Geometry file (.xyz)
0.001           # Time Step
10000           # n of Steps
500             # n of Walkers

Only for Pure Diffusion Monte Carlo:
-1.3            # Reference energy (a.u)
10000           # Projection time
```

The first entry is used to select the method and is read as a string of 3 characters corresponding to `var` or `dif` for VMC or PDMC methods, respectively. You can then change the parameter a, the number of electrons (1 or 2), and the file **.xyz** in which the coordinates of the nuclei in the system are written, from which the program will read them.

Next, you can change the simulation statistics by modifying parameters such as the time step, the number of steps and the number of walkers, that is, the number of times a simulation is run. Separately then you can change the reference energy and the projection time for a PDMC simulation.

2.2 The Code

The main code is contained in the file **QMC.f90** in which the input files **qmc.inp** and [**name**]**.xyz** are read by initializing the simulation. Here the nuclear coordinates are converted to atomic units and the center of charge of the system, equivalent to the center of mass for a system of hydrogen nuclei, is calculated, thus translating the nuclear coordinates with respect to it.
The simulation is then started by calling the subroutine `VMC` or `PDMC` depending on the choice. These are contained in the file **Var_Dif.f90**.
Finally, the average energy value and acceptance ratio value with their error bars are written into the standard output. The averages and error bars are calculated by the subroutine `ave_error` in the file **subroutines.F90**. In addition, the energy values calculated by each walker are written to the file **e_loc.dat** so that they can be plotted and the trend of the statistics can be visualized graphically.

The file **Var_Dif.f90**, as mentioned above, contains the core of the code i.e., the subroutines in which the VMC and PDMC methods are implemented.

In the file **psi_energies.F90** the local energy and wave function are implemented, complete with tests for kinetic and potential energy.

In the file **subroutines.F90** the subrotuines for the calculation of the drift vector, the Box-Muller method, and the calculation of the mean and error are implemented. Also present is the test on the subrotuine `drift`.