Intro to Quantum Monte Carlo

Nathan Apfel

1 Goal and Overview

In this project, I implement some basic, easy-to-follow Quantum Monte Carlo algorithms. First, I want to go over the basics of Monte Carlo (MC) methods in general by demonstrating a 3-D Metropolis Markov Chain Monte Carlo sampling algorithm that calculates the expectation value for the energy of wave function of a 2p orbital of a hydrogen atom. Second, I will attempt a Variational Monte Carlo method. Third, I will walk through the implementation of the interesting yet limited diffusion Monte Carlo.

Quantum Monte Carlo methods are a powerful family of numerical methods used across many physics and quantum chemistry applications. There isn't really one "Quantum Monte Carlo"—these methods are varied enough that one comprehensive definition cannot accurately tie them together. They all have one uniting feature—the use of random numbers to handle high dimensional integration or complicated combinatorix is what makes them so powerful. In deterministic numerical integration, error scales exponentially with the number of dimensions, which makes calculation of probability distributions (or pseudo-probability distributions, like quantum mechanical wave functions) across many interacting variables prohibitively expensive. Monte Carlo integration avoids this, with the error famously scaling like $1/\sqrt{M}$ no matter the dimension, where M is the number of evaluations.

This tutorial is meant for those who have some background (perhaps one grad-level semester) in quantum mechanics. Specifically, the two methods I describe are used to find ground-state energies of hydrogenic wave functions and quantum oscillators.

1.1 Metropolis Markov Chain Monte Carlo (MCMC) sampling and integration

1.1.1 Metropolis MCMC algorithm

The program takes a point and uses it to create another point nearby such that all of the points end up distributed with density determined by the probability distribution $P(R)$.

1. Given a starting point A, generate a new point B nearby according to a normal distribution centered at A
2. Calculate $P_{acc} = \min\left(1, \frac{P(B)}{P(A)}\right)$, the acceptance probability
3. Accept the point with probability P_{acc} – if the point is accepted, use it to generate the next point. If it's not, restart the procedure with the same A.

Beginning with a random point, let the program run a while to find the center of your probability distribution. Eventually, start collecting the points it generates and take these as your samples. I did this with a 2P orbital centered at (5,0,0).

For high dimensions (i.e. multi-particle systems), it's best to move one particle at a time (3 dimensions at a time).

1.1.2 Metropolis MCMC Expectation Values

As the points are accumulated, an expectation value can be calculated by the following sum:

$$\langle f(X) \rangle = \sum_{n}^{M} P_{acc} f(X_n) + (1 - P_{acc}) f(X_{n-1}) \tag{1}$$

I used this to calculate the expectation value $\langle X \rangle$ and $\left\langle -\frac{\hbar^2}{2m}\nabla^2 + \frac{ke^2}{r} \right\rangle$. To calculate the laplace operator, I used the autograd package.

1.2 Variational Monte Carlo

Variational Monte Carlo uses this MCMC method of finding expectation values to calculate the energy and energy variance for a trial wave function, then tweaks the wave function based on some parameters to minimize said energy (or, often the variance as energy eigenstates have no energy variance). The expectation of the energy can be given by:

$$E = \frac{\langle \psi | H | \psi \rangle}{\langle \psi | \psi \rangle} = \frac{1}{\int d^{3N} R \psi \dagger \psi} \int d^{3N} R \psi^\dagger \psi \frac{H\psi}{\psi} \tag{2}$$

We can interpret $\frac{\psi^\dagger \psi}{\int d^{3N} R \psi \dagger \psi}$ as a probability distribution and sample it, taking the rest to be the function of which we are finding the expectation value. This is often called the "local energy."

1.3 Diffusion Monte Carlo (DMC)

Rewriting the Schrödinger Equation with $\tau = -it$ (a cool trick known as Wick rotation), we get the following:

$$\frac{\partial}{\partial \tau}\Psi = \left(-\frac{\hbar^2}{2m}\nabla^2 + V(\mathbf{R}) \right) \Psi \tag{3}$$

Which has the exact form of a diffusion equation. As diffusion is the macroscopic limit of Brownian motion, it can be easily approximated by random walk Markov Chain Monte Carlo methods. If we write out an arbitrary solution to this new, imaginary time-dependent Schrödinger equation as a superposition of energy eigenstates, we get the this nice sum:

$$|\psi\rangle = \sum_n |E_n\rangle e^{-\frac{E_n}{\hbar}\tau}$$

This conveniently means that any energy eigenstate $|E_n\rangle$ will decay according to $e^{-\frac{E_n}{\hbar}\tau}$. Thus, if we adjust this equation by adding a constant $-E_0$ ground energy to the potential, we get the following behavior: any wave function, a superposition of eigenstates, will decay such that the only remaining contribution is that with $E_{adjusted} = 0$, i.e. $E = E_0$.

1.3.1 The algorithm

1. Given a well-chosen trial wave function Ψ_0, generate a configuration M points, called "walkers", using a MCMC method.
2. Calculate the energy of the configuration, using $\frac{\hbar^2}{2m}\nabla^2\Psi_0$ evaluated at each point in the configuration as the kinetic energy.

3. Move each point R according to a "quantum diffusion force" $d\tau F_q = d\tau \frac{\nabla \Psi_0}{\Psi 0}$ with an added random Gaussian drift to a point R'. This simulates the diffusion away from areas of high concentration over an imaginary time step $d\tau$
4. Accept or reject the move according to a weight function similar to that of Metropolis MCMC. Usually, this is related to a local Green's function solution of the Schrödinger Equation.
5. Create $m = \exp\left(-\left(\frac{1}{2}(E_l(R) + E_l(R')) - E_t\right) d\tau\right) + x$ copies of each walker, where E_l is the local energy calculated using the potential energy function and the kinetic energy of the guiding wave function, and x is a random number from 0-1.
6. Adjust the trial energy so that the number of walkers remains approximately constant. This will give an approximation for the ground state energy, assuming Ψ_0 and its derived potential energy is mostly correct.

1.4 Further reading

1. MUST READ!! This is Oak Ridge National Laboratory researcher Paul Kent's description of VMC and DMC, the best of it's kind on the internet. I encourage anyone interested to check it out. `https://web.ornl.gov/~kentpr/thesis/pkthnode21.html#SECTION00841000000000000000`
2. An early paper describing QMC by R.C Grimm and R.G Storer `https://www.sciencedirect.com/science/article/pii/0021999171900544`
3. An implementation of DMC that differs slightly from but has been the inspiration for this one by Ian Terrel `https://www.huy.dev/Terrel-thesis.pdf`
4. Helpful overview of Metropolis MCMC by Charles J. Geyer `https://www.mcmchandbook.net/HandbookChapter1.pdf`
5. An early paper by David Ceperley and Berni Alder going over DMC in an easy-to-follow way `https://www.science.org/doi/epdf/10.1126/science.231.4738.555?adobe_mc=MCMID%3D6816523141421699336120688`
   `7CMCORGID%3D242B6472541199F70A4C98A6%2540AdobeOrg%7CTS%3D1701480501`
6. MUST read for an intro to MC methods in physics by Stefan Weinzierl `https://arxiv.org/abs/hep-ph/0006269`
7. An AWS Jupyter notebook introducing QMC on a quantum computer via Amazon Braket by `https://github.com/amazon-braket/amazon-braket-examples/blob/feature/quantum-monte-carlo/examples/hybrid_quantum_algorithms/Quantum_Monte_Carlo_Chemistry/Quantum_Monte_Carlo_Chemistry.ipynb`

There are also many resources on YouTube, and searching 'quantum Monte Carlo' is always sure to be a lucrative experience.