Variational Monte Carlo for the Hydrogen Atom

The Hydrogen atom is a system with two particles, electron and proton. The configuration space in which the system moves is therefore six dimensional. By moving to the center-of-mass system, the problem becomes effectively 3 dimensional, with Hamiltonian

$$H = -\frac{\hbar^2}{2m}\nabla^2 - \frac{e^2}{r} \, ,$$

where $\mathbf{r} = \mathbf{r}_{\rm e} - \mathbf{r}_{\rm p}$ is the relative coordinate of the electron with respect to the proton, e is the magnitude of the electron's charge, and $m = m_{\rm e}m_{\rm p}/(m_{\rm e} - m_{\rm p})$ is the reduced mass.

Reduction to a one-dimensional problem

By using conservation of angular motion and the fact that the ground state is spherically symmetric, i.e., it has zero orbital angular momentum, the problem can be reduced to one dimension with Hamiltonian operator

$$H = -\frac{\hbar^2}{2m}\left[\frac{{\rm d}^2}{{\rm d}r^2} + \frac{2}{r}\frac{\rm d}{{\rm d}r}\right] - \frac{e^2}{r} \, ,$$

which depends on on the radial coordinate r.

Exact solution for the ground state

The exact ground state energy and wavefunction are given by

$$E_0 = -\frac{e^2}{2a_0} \, , \qquad \psi_0(r) \sim e^{-r/a_0} \, .$$

where the *Bohr radius*

$$a_0 = \frac{\hbar^2}{me^2} \, .$$

It is convenient to use *atomic units* in which $\hbar = m = e = 1$ so

$$H = -\frac{1}{2}\left[\frac{{\rm d}^2}{{\rm d}r^2} + \frac{2}{r}\frac{\rm d}{{\rm d}r}\right] - \frac{1}{r} \, , \qquad E_0 = -\frac{1}{2} \, , \qquad \psi_0(r) \sim e^{-r} \, .$$

Variational trial wave function and local energy

A simple trial wave function for the Hydrogen atom ground state is

$$\psi_{\mathrm{T},\alpha}(r) = e^{-\alpha r} .$$

The local energy for this choice can easily be computed:

$$E_{\mathrm{L}}(r) = \frac{1}{\psi_{\mathrm{T},\alpha}} H \psi_{\mathrm{T},\alpha}(r) = -\frac{1}{2}\left[\alpha^2 - \frac{2\alpha}{r}\right] - \frac{1}{r} .$$

Note two important points about this local energy:

- It is minimum and also independent of r at $\alpha = 1$, which gives the exact ground state energy and eigenfunction.
- For $\alpha \neq 1$ it is *singular* at $r = 0$ where the potential diverges. For more complex problems, like the Helium atom to be considered next, these singularities can cause problems with the numerical calculation. To deal with these singularities, *cusp conditions* are used to restrict the variational parameters.

The textbook gives results for a VMC simulation of the ground state energy of Hydrogen: the harmonic oscillator program can be adapted to reproduce these results by changing the form of the trial wave function and local energy.

Two additional problems need to be addressed. Since $r \geq 0$, a one-dimensional Metropolis walker should not be allowed to cross the origin to $r < 0$. Also, the probability that the one-dimensional walker is found between r and $r + \mathrm{d}r$ must be proportional to $4\pi r^2$, which is the surface area of a sphere of radius r. The textbook suggests using a walker in 3 dimensional space. Given a walker position $\mathbf{r}$ and a maximum step size δ, the next trial step is chosen uniformly at random within a cube of side 2δ centered on the point $\mathbf{r}$ and aligned with the coordinate axes. This solves both of the problems above at the expense of making three calls to the random number generator for each trial move.

Variational Monte Carlo for the Helium Atom

The Helium atom is a 3-particle problem: two electrons orbit around a nucleus, which consists of two protons with charge e each and two neutral neutrons. The nucleus, which is $\sim 8,000$ times more massive than an electron, can be assumed to be at rest at the origin of the coordinate system. The electrons have positions $\mathbf{r}_1$ and $\mathbf{r}_2$. This is simpler than making a transformation to the center-of-mass system of the three particles, and it is sufficiently accurate.

If we use atomic units with $\hbar = m_e = e = 1$, the Hamiltonian for the motion of the two electrons can be written

$$H = -\frac{1}{2}\nabla_1^2 - \frac{1}{2}\nabla_2^2 - \frac{2}{r_1} - \frac{2}{r_2} + \frac{1}{r_{12}} ,$$

where $r_{12} = |\mathbf{r}_{12}| = |\mathbf{r}_1 - \mathbf{r}_2|$. The terms $-2/r_i$ represent the negative (attractive) potential energy between each electron with charge -1 and the Helium nucleus with charge $+2$, and the term $+1/r_{12}$ represents the positive (repulsize) potential energy between the two electrons.

A simple choice of variational trial wave function

If the repulsive term $1/r_{12}$ were not present, then the Hamiltonian would be that of two independent Hydrogen-like atoms. It can be shown that the energy and ground state wave function of a Hydrogen-like atom whose nucleus has charge Z are given by

$$E_0 = -\frac{Z^2}{2} , \qquad \psi_0 \sim e^{-Zr} .$$

The wave function of the combined atom with two non-interacting electrons would be the product of two such wave functions:

$$\psi(\mathbf{r}_1, \mathbf{r}_2) \sim e^{-2r_1} e^{-2r_2} .$$

This suggests a trial wave function of the form

$$\Psi_{\mathrm{T},\alpha} = e^{-\alpha r_1} e^{-\alpha r_2} ,$$

similar to what was done for the Hydrogen atom. If the electron-electron interaction is neglected, then the average energy with this wave function can be calculated

$$\left\langle -\frac{1}{2}\nabla_1^2 - \frac{1}{2}\nabla_2^2 - \frac{2}{r_1} - \frac{2}{r_2} \right\rangle = 2 \times \frac{\alpha^2}{2} - 2 \times \alpha ,$$

which has a minimum at $\alpha = 1$, which gives $\langle E \rangle = -1$. The experimentally measured ground state energy is $E_0 = -2.904$.

In fact, the average energy can be evaluated exactly for this trial wave function even if the electron-electron interaction is included:

$$\left\langle -\frac{1}{2}\nabla_1^2 - \frac{1}{2}\nabla_2^2 - \frac{2}{r_1} - \frac{2}{r_2} + \frac{1}{r_{12}} \right\rangle = \alpha^2 - \frac{27}{8}\alpha \, ,$$

which has a minimum at $\alpha = 27/16$, which gives $\langle E \rangle = -2.8477$. This shows that the repulsion between the electrons is important and lowers the energy.

Padé-Jastrow wave function

The textbook suggest using a trial wave function

$$\Psi(\mathbf{r}_1, \mathbf{r}_2) = e^{-2r_1} e^{-2r_2} e^{\frac{r_{12}}{2(1+\alpha r_{12})}} \, ,$$

with α as a variational parameter. The local energy with this wave function can be calculated

$$E_{\mathrm{L}}(\mathbf{r}_1, \mathbf{r}_2) = -4 + \frac{\alpha}{(1+\alpha r_{12})} + \frac{\alpha}{(1+\alpha r_{12})^2} + \frac{\alpha}{(1+\alpha r_{12})^3} - \frac{1}{4(1+\alpha r_{12})^4} + \frac{\hat{\mathbf{r}_{12}} \cdot (\hat{\mathbf{r}_1} - \hat{\mathbf{r}_2})}{(1+\alpha r_{12})^2} \, .$$

VMC program for the Helium Atom

The following program `vmc-he.cpp` implements this trial function choice.

vmc-he.cpp

```
// Variational Monte Carlo for the Helium Atom

#include <cmath>
```

```
#include <cstdlib>
#include <iostream>
#include "rng.h"

using namespace std;

const int NDIM = 3;         // dimensionality of space
const int NELE = 2;         // number of electrons
int N;                      // number of walkers
double (*r)[NELE][NDIM];    // walker coordinates in 6-D configuration space

double alpha;               // Pade-Jastrow variational parmeter
double delta;               // trial step size

void initialize() {
    r = new double [N][NELE][NDIM];
    for (int n = 0; n < N; n++)
    for (int e = 0; e < NELE; e++)
    for (int d = 0; d < NDIM; d++)
        r[n][e][d] = qadran() - 0.5;
    delta = 1;
}

double eSum;
double eSqdSum;

void zeroAccumulators() {
    eSum = eSqdSum = 0;
}
```

```
double Psi(double *rElectron1, double *rElectron2) {

    // value of trial wave function for walker n
    double r1 = 0, r2 = 0, r12 = 0;
    for (int d = 0; d < 3; d++) {
        r1 += rElectron1[d] * rElectron1[d];
        r2 += rElectron2[d] * rElectron2[d];
        r12 += (rElectron1[d] - rElectron2[d])
               * (rElectron1[d] - rElectron2[d]);
    }
    r1 = sqrt(r1);
    r2 = sqrt(r2);
    r12 = sqrt(r12);
    double Psi = - 2*r1 - 2*r2 + r12 / (2 * (1 + alpha*r12));
    return exp(Psi);

}

double eLocal(double *rElectron1, double *rElectron2) {

    // value of trial wave function for walker n
    double r1 = 0, r2 = 0, r12 = 0;
    for (int d = 0; d < 3; d++) {
        r1 += rElectron1[d] * rElectron1[d];
        r2 += rElectron2[d] * rElectron2[d];
        r12 += (rElectron1[d] - rElectron2[d]) *
               (rElectron1[d] - rElectron2[d]);
    }
```

```
        r1 = sqrt(r1);
        r2 = sqrt(r2);
        r12 = sqrt(r12);
        double dotProd = 0;
        for (int d = 0; d < 3; d++) {
            dotProd += (rElectron1[d] - rElectron2[d]) / r12 *
                       (rElectron1[d] / r1 - rElectron2[d] / r2);
        }
        double denom = 1 / (1 + alpha * r12);
        double denom2 = denom * denom;
        double denom3 = denom2 * denom;
        double denom4 = denom2 * denom2;
        double e = - 4 + alpha * (denom + denom2 + denom3)
                   - denom4 / 4 + dotProd * denom2;
        return e;
    }

    int nAccept;

    void MetropolisStep(int walker) {

        // make a trial move of each electron
        double rElectron1[3], rElectron2[3], rTrial1[3], rTrial2[3];
        for (int d = 0; d < 3; d++) {
            rElectron1[d] = r[walker][0][d];
            rTrial1[d] = rElectron1[d] + delta * (2 * qadran() - 1);
            rElectron2[d] = r[walker][1][d];
            rTrial2[d] = rElectron2[d] + delta * (2 * qadran() - 1);
        }
        // Metropolis test
        double w = Psi(rTrial1, rTrial2) / Psi(rElectron1, rElectron2);
        if (qadran() < w * w) {
            for (int d = 0; d < 3; d++) {
                r[walker][0][d] = rElectron1[d] = rTrial1[d];
                r[walker][1][d] = rElectron2[d] = rTrial2[d];
            }
            ++nAccept;
        }

        // accumulate local energy
        double e = eLocal(rElectron1, rElectron2);
        eSum += e;
        eSqdSum += e * e;
    }

    void oneMonteCarloStep() {

        // do Metropolis step for each walker
        for (int n = 0; n < N; n++)
            MetropolisStep(n);
    }

    int main() {

        cout << " Variational Monte Carlo for Helium Atom\n"
             << " ---------------------------------------\n";
        cout << " Enter number of walkers:  ";
```

```
cin >> N;
cout << " Enter parameter Pade-Jastrow parameter alpha:  ";
cin >> alpha;
cout << " Enter number of Monte Carlo steps:  ";
int MCSteps;
cin >> MCSteps;

initialize();

// perform 20% of MCSteps as thermalization steps
// and adjust step size so acceptance ratio ~50%
int thermSteps = int(0.2 * MCSteps);
int adjustInterval = int(0.1 * thermSteps) + 1;
nAccept = 0;
cout << " Performing " << thermSteps << " thermalization steps ..."
     << flush;
for (int i = 0; i < thermSteps; i++) {
    oneMonteCarloStep();
    if ((i+1) % adjustInterval == 0) {
        delta *= nAccept / (0.5 * N * adjustInterval);
        nAccept = 0;
    }
}
cout << "\n Adjusted step size delta = " << delta << endl;

// production steps
zeroAccumulators();
nAccept = 0;
cout << " Performing " << MCSteps << " production steps ..." << flush;
```

```
        for (int i = 0; i < MCSteps; i++)
            oneMonteCarloStep();

        // compute and print energy
        double eAve = eSum / double(N) / MCSteps;
        double eVar = eSqdSum / double(N) / MCSteps - eAve * eAve;
        double error = sqrt(eVar) / sqrt(double(N) * MCSteps);
        cout << "\n <Energy> = " << eAve << " +/- " << error
             << "\n Variance = " << eVar << endl;
    }
```