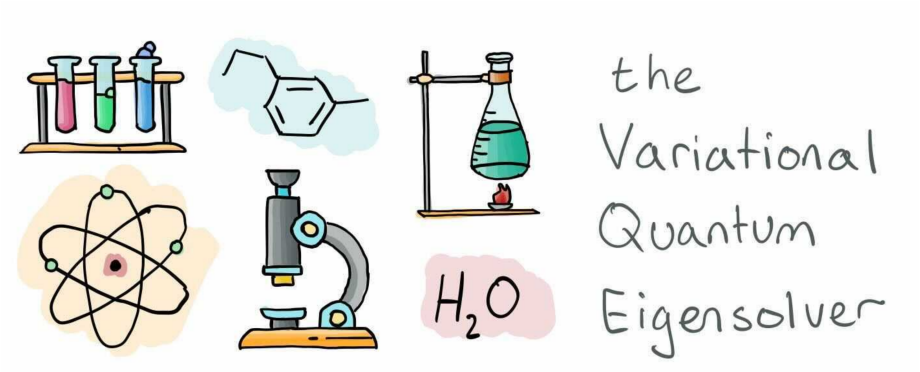

VQE: Moving On Up [500 points]

Version: 1

NOTE: Coding templates are provided for all challenge problems at this link. You are strongly encouraged to base your submission off the provided templates.

Overview: VQE challenges

Many problems in the physical sciences involve computation of the energy levels of quantum systems. The evolution of quantum systems is governed by a special type of matrix operator called a *Hamiltonian*, usually denoted H. The energies of a system, and the states that have them, can be obtained by finding the eigenvalues (or *eigenenergies*), and eigenvectors (*eigenstates*) of H:

$$H|\psi_i\rangle = E_i|\psi_i\rangle$$

where the $\{|\psi_i\rangle\}$ are the eigenstates, and $\{E_i\}$ are the real-valued eigenenergies.

Of particular interest is usually the ground-state energy:

$$H|\psi_g\rangle = E_g|\psi_g\rangle, \quad E_g = \min\{E_i\}.$$

One algorithm for finding this energy using a quantum computer is the variational quantum eigensolver (VQE). The VQE works by parameterizing the space of possible quantum states, and optimizing to find the set of parameters that yield the lowest energy. Formally, the optimization problem is

$$\min_{\alpha} \quad \langle 0\cdots 0|U^\dagger(\alpha)HU(\alpha)|0\cdots 0\rangle,$$

where α represents a set of parameters. The operation $U(\alpha)$ is a special type of quantum circuit called an ansatz. Ansaetze (the plural form of ansatz) are specially created in that we expect there is an $\tilde{\alpha}$ such that $U(\tilde{\alpha})|0\cdots 0\rangle = |\psi_g\rangle$.

The VQE consists of both quantum (Q) and classical (C) computations:

1. (C) Choose a suitable ansatz circuit $U(\alpha)$
2. (C) Choose a starting set of parameters α
3. (Q) Apply $U(\alpha)$ and measure the output state
4. (C) Use measurement results to compute numerical value of $\langle 0|U^\dagger(\alpha)HU(\alpha)|0\rangle$ (the *energy*)
5. (C) Use some optimization routine to choose a new α that should bring us to a state closer to the ground state.
6. Repeat steps 3-5 until the optimizer converges to a minimum value, or the number of iterations has exceeded a specified maximum.

The portion that is done on the quantum computer involves simulation of the quantum system—this is what quantum computers do best, and what gives VQE an edge over just solving this problem classically. In this set of challenges, you'll explore how to implement and extend the VQE to calculate the energies of quantum systems.

Problem statement [500 points]

While finding the ground states of Hamiltonians is important work, there are many applications in which *excited states* (the eigenstates with larger eigenvalues) are also important. The VQE algorithm as detailed above finds only the ground state. However, with a little bit of tweaking, a similar approach can be used to find the higher levels too.

In this problem, your job is to implement a variational method that will find the ground state, as well as the first two excited states of the provided Hamiltonian.

Input

The input to the program is a Hamiltonian of a prescribed type, with a varying number of qubits. It will be converted to a PennyLane Hamiltonian for you.

Output

The output of your program should be a series of floating point numbers separated by commas that represent the lowest three energies of the system. They should be ordered from smallest to largest.

Acceptance Criteria

In order for your submission to be judged as "correct":

- The outputs generated by your solution when run with a given `.in` file must match those in the corresponding `.ans` file to within the `Tolerance`

specified below. Additional outputs not found in the `.ans` file but that are generated by the provided template must match those expected outputs exactly.

- Your solution must take no longer than the `Time limit` specified below to produce its outputs.

You can test your solution by passing the `#.in` input data to your program as stdin and comparing the output to the corresponding `#.ans` file:

```
python3 vqe_500_template.py < 1.in
```

WARNING: Don't modify any of the code in the template outside of the `# QHACK #` markers, as this code is needed to test your solution. Do not add any print statements to your solution, as this will cause your submission to fail.

Specs
Tolerance: **0.03 (3%)**
Time limit: **180 s**

Version History

Version 1: Initial document.