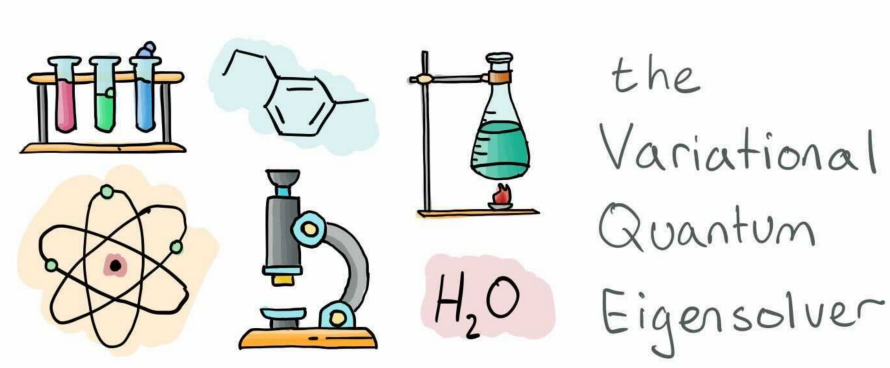

VQE: Ansatz Artistry [200 points]

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 [200 points]

The ansatz used in the 100-point question is a type of ansatz known as a *hardware-efficient* ansatz. These ansaetze use general single-qubit unitary rotations, and CNOT gates are performed only between adjacent qubits. Hardware-efficient ansaetze can be applied to most problems. For some problems, though, the ansatz design can be tailored to produce only quantum states of a prescribed type. In this problem, you'll design an ansatz for a class of Hamiltonians whose n-qubit eigenstates must have the form:

$$|\psi(\alpha)\rangle = \alpha_0|10\cdots0\rangle + \alpha_1|010\cdots0\rangle + \cdots + \alpha_{n-2}|0\cdots010\rangle + \alpha_{n-1}|0\cdots01\rangle,$$

where all the α_i are *real* numbers.

Your job is to:

- Design and implement the n-qubit ansatz
- Implement the classical optimization components of the VQE
- Put everything together and use the VQE to compute the ground state of these Hamiltonians!

Hint: A good way to start is by determining the number of parameters you will need, their properties, and what quantum operations produce them. Work iteratively, starting with the $n = 2$ case, then move to $n = 3$, to see how the ansatz generalizes. To help with checking your work, you can use the `qml.state()` functionality to view the output state (as demonstrated here). **Note: make sure to remove any print statements from your code before you submit it, or the submission will fail.**

Input

The input to the program is a particular class of Hamiltonian with a variable number of qubits. The $n = 2$ and $n = 3$ Hamiltonians have been provided for testing in the files `1.in` and `2.in` respectively. For visual ease, they are:

$$H_2 = 5.906709I + 0.218291Z_0 - 6.125Z_1 - 2.143304X_0X_1 - 2.143304Y_0Y_1$$

$$\begin{aligned} H_3 =& 15.531709I + 0.218291Z_0 - 6.125Z_1 - 2.143304X_0X_1 - 2.143304Y_0Y_1 \\ & - 9.625Z_2 - 3.913119X_1X_2 - 3.913119Y_1Y_2 \end{aligned}$$

They will be converted to a PennyLane Hamiltonian for you. Your solution will be graded using some larger values of n.

Output

The output of your program should be a single floating-point number which represents the ground state energy of the system.

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.
- 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_200_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: **0.01 (1%)**
Time limit: **60 s**

Version History

Version 1: Initial document.