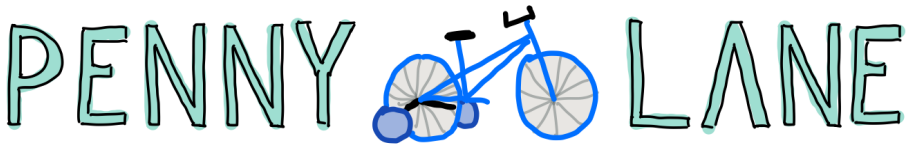

PennyLane 101: Know Your Devices [200 points]

Version: 1

PennyLane 101

The **PennyLane 101** challenges will introduce quantum computing concepts with PennyLane. Whether you're coming from an advanced quantum computing background, or you've never evaluated a quantum circuit before, these challenge questions will be a great start for you to learn how quantum computing works using PennyLane. Beyond these five questions in this category, there are well-developed demos and tutorials on the PennyLane website that are a good resource to fall back on if you are stuck. We also strongly recommend consulting the PennyLane documentation to see an exhaustive list of available gates and operations!

Problem statement [200 points]

In this challenge question, you need to implement two circuits that are effectively the same. The difference between both circuits will be the format of the output that is a consequence of the *device* that each circuit runs on. More on this in a bit. For now, the circuit you must implement is in Figure 1.

The provided template file called `know_your_devices_template.py` contains a function `compare_circuits` that you need to complete. Within this function are two functions you must complete called `pure_circuit` and `mixed_circuit`. Both functions will implement the circuit in Figure 1 with a different set of rotation angles and return the quantum state (`qml.state()`). You have to decorate each function correspondingly to return a pure state or a density matrix. In other words, you need to determine the quantum devices that store the output state as a density matrix (which is required for `mixed_circuit`) and as a state vector (which is required for `pure_circuit`).

After creating both circuits, their output will be compared by computing the matrix one-norm between both states:

$$\sum_{i,j} |\hat{\rho} - |\psi\rangle\langle\psi||_{i,j} \, .$$

To summarize, the completed `compare_circuits` function should:

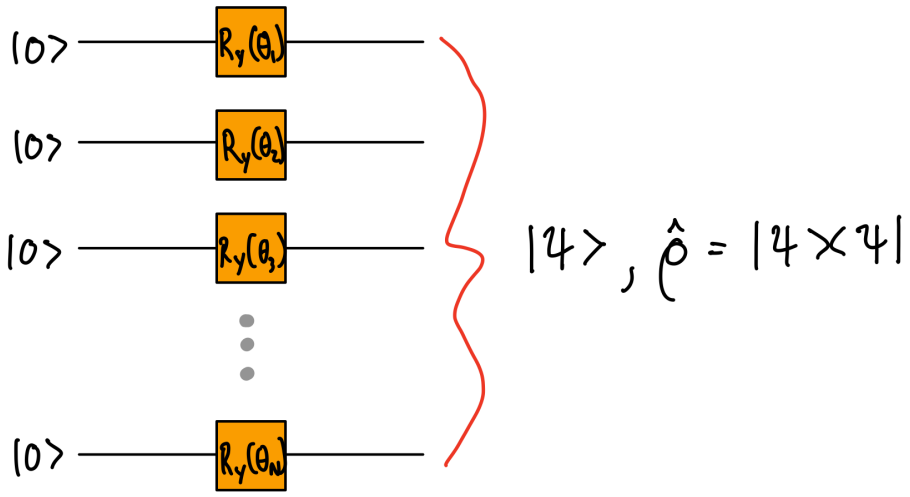

Figure 1: The circuit you must implement

- Define quantum devices for each circuit and decorate them correspondingly.
- Each circuit implements a y-rotation gate on each qubit/wire.
- Return the matrix one-norm as indicated above.

Input

- `list`: A list containing the number of qubits / wires and angles required to define y-rotation gates.

Output

- `float`: The corresponding matrix one-norm.

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 `0.0001` tolerance specified below. To clarify, your answer must satisfy

$$\text{tolerance} \geq \left| \frac{\text{your solution} - \text{correct answer}}{\text{correct answer}} \right|.$$

- Your solution must take no longer than the `60s` 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 {name_of_file}.py < 1.in
```

WARNING: Don't modify the code outside of the `# QHACK #` markers in the template file, 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: **0.0001**
Time limit: **60 s**

Version History

Version 1: Initial document.