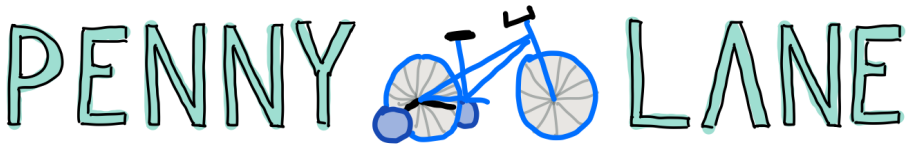

PennyLane 101: Finite-Difference Gradient [400 points]

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 [400 points]

In this challenge, you will be dealing with taking gradients of variational circuit outputs with respect to their tunable parameters. In PennyLane, there are several ways to calculate gradients, all of which are accessed through a keyword argument `"diff_method"` in a `qml.qnode` decorator (e.g., `@qml.qnode(dev, diff_method="<method>")`). Accessible methods include backpropagation (`"backprop"`), the adjoint method (`"adjoint"`), the parameter-shift rule (`"parameter-shift"`), and finite-difference method (`"finite-diff"`). You will be tasked with making a homemade finite-difference gradient calculator without using `@qml.qnode(dev, diff_method="finite-diff")` in your solution.

Finite-difference differentiation originates from the familiar expression that you may have seen in introductory calculus,

$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x_i + \delta/2) - f(x_i - \delta/2)}{\delta},$$

where δ is a very small positive number. The provided template file `finite_difference_template.py` contains a function called `my_finite_diff_grad` that you need to complete. The `variational_circuit` function implements a variational circuit that needs 6 parameters. The function that you take the gradient of is the `cost` function, which outputs a measurement from the

given variational circuit. The `my_finite_diff_grad` function will output an `np.ndarray` representing the gradient of the `cost` function with respect to each of the 6 variational circuit parameters.

Input

- `list(float)`: A list containing the 6 variational circuit parameters.

Output

- `list(float)`: A list representing the gradient of the cost function.

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 `0.005` 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: **0.005**
Time limit: **60 s**

Version History

Version 1: Initial document.