# NIThePmini_school_L4-barren_plateaus

September 29, 2020

## 1 Barren plateaus in quantum neural networks

This notebook implements code from a PennyLane tutorial illustrating the barren plateau phenomenon found in quantum neural networks. The original tutorial can be found here: https://pennylane.ai/qml/demos/tutorial_barren_plateaus.html, author: Shahnawaz Ahmed (shahnawaz.ahmed95@gmail.com)

In [1]:
```
import pennylane as qml
from pennylane import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
np.random.seed(42)

num_qubits = 4
dev = qml.device("default.qubit", wires=num_qubits)
gate_set = [qml.RX, qml.RY, qml.RZ]
```

In [3]:
```
def rand_circuit(params, random_gate_sequence=None, num_qubits=None):
    for i in range(num_qubits):
        qml.RY(np.pi / 4, wires=i)

    for i in range(num_qubits):
        random_gate_sequence[i](params[i], wires=i)

    for i in range(num_qubits - 1):
        qml.CZ(wires=[i, i + 1])

    H = np.zeros((2 ** num_qubits, 2 ** num_qubits))
    H[0, 0] = 1
    wirelist = [i for i in range(num_qubits)]

    return qml.expval(qml.Hermitian(H, wirelist))
```

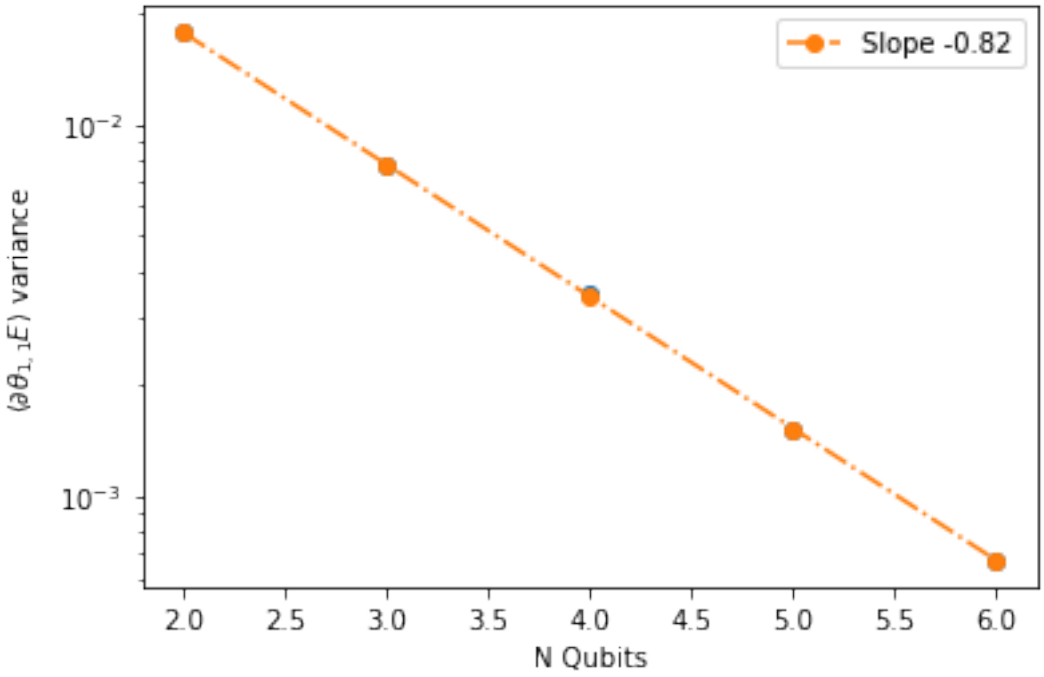
In [4]:
```
grad_vals = []
num_samples = 200
```

```
        for i in range(num_samples):
            gate_sequence = {n: np.random.choice(gate_set) for n in range(num_qubits)}
            qcircuit = qml.QNode(rand_circuit, dev)
            grad = qml.grad(qcircuit, argnum=0)
            params = np.random.uniform(0, 2 * np.pi, size=num_qubits)
            gradient = grad(params, random_gate_sequence=gate_sequence, num_qubits=num_qubits)
            grad_vals.append(gradient[-1])

        print("Variance of the gradients for {} random circuits: {}".format(num_samples, np.va
        print("Mean of the gradients for {} random circuits: {}".format(num_samples, np.mean(g
```

```
Variance of the gradients for 200 random circuits: 0.005681513693081505
Mean of the gradients for 200 random circuits: -0.001000226897652134
```

```
In [5]: qubits = [2, 3, 4, 5, 6]
        variances = []


        for num_qubits in qubits:
            grad_vals = []
            for i in range(num_samples):
                dev = qml.device("default.qubit", wires=num_qubits)
                qcircuit = qml.QNode(rand_circuit, dev)
                grad = qml.grad(qcircuit, argnum=0)

                gate_set = [qml.RX, qml.RY, qml.RZ]
                random_gate_sequence = {i: np.random.choice(gate_set) for i in range(num_qubit

                params = np.random.uniform(0, np.pi, size=num_qubits)
                gradient = grad(params, random_gate_sequence=random_gate_sequence, num_qubits=
                grad_vals.append(gradient[-1])
            variances.append(np.var(grad_vals))

        variances = np.array(variances)
        qubits = np.array(qubits)
```

```
In [6]: # Fit the semilog plot to a straight line
        p = np.polyfit(qubits, np.log(variances), 1)


        # Plot the straight line fit to the semilog
        plt.semilogy(qubits, variances, "o")
        plt.semilogy(qubits, np.exp(p[0] * qubits + p[1]), "o-.", label="Slope {:3.2f}".format
        plt.xlabel(r"N Qubits")
        plt.ylabel(r"$\langle \partial \theta_{1, 1} E\rangle$ variance")
        plt.legend()
        plt.show()
```


Slope -0.82
$10^{-2}$
$10^{-3}$
$(\partial\theta_{1,1}E)$ variance
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
N Qubits