Solution 12: mini IBM Q project- Theoretical part
Traitement Quantique de l'Information

1 Circuit output for diagonal D

a) For $n = 1, 2$, we have

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{1}$$

$$\begin{aligned}(H \otimes H)|00\rangle &= (H|0\rangle) \otimes (H|0\rangle) \\ &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ &= \frac{1}{2}\big(|00\rangle + |01\rangle + |10\rangle + |11\rangle\big)\end{aligned} \tag{2}$$

As we can see, $H^{\otimes 2}|00\rangle$ is an equal superposition of all computational basis states. For any n, we have

$$\begin{aligned}|\psi_1\rangle = H^{\otimes n}|0\rangle^{\otimes n} &= \underbrace{(H|0\rangle) \otimes \ldots \otimes (H|0\rangle)}_{\text{n times}} \\ &= \underbrace{\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \ldots \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)}_{\text{n times}} \\ &= \frac{1}{2^{n/2}} \sum_{b_1 \ldots b_n \in \{0,1\}^n} |b_1 b_2 \ldots b_n\rangle\end{aligned} \tag{3}$$

And, for $|\psi_2\rangle$ we obtain

$$\begin{aligned}|\psi_2\rangle &= D|\psi_1\rangle \\ &= D\Big\{\frac{1}{2^{n/2}} \sum_{b_1 \ldots b_n \in \{0,1\}^n} |b_1 b_2 \ldots b_n\rangle\Big\} \\ &= \frac{1}{2^{n/2}} \sum_{b_1 \ldots b_n \in \{0,1\}^n} D|b_1 b_2 \ldots b_n\rangle \\ &= \frac{1}{2^{n/2}} \sum_{b_1 \ldots b_n \in \{0,1\}^n} e^{i\varphi(b_1,\ldots,b_n)}|b_1 b_2 \ldots b_n\rangle\end{aligned} \tag{4}$$

In the second line, we use the linearity of D to get the third line, in which $D|b_1 b_2 \ldots b_n\rangle$ is by assumption $e^{i\varphi(b_1,\ldots,b_n)}|b_1 b_2 \ldots b_n\rangle$.

b) Generalizing the formula $H|b_i\rangle = \frac{1}{\sqrt{2}}\sum_{c_i=0,1}(-1)^{b_ic_i}|c_i\rangle$ for arbitrary n, we get

$$\begin{aligned} H^{\otimes n}|b_1\dots b_n\rangle &= (H|b_1\rangle)\otimes\dots\otimes(H|b_n\rangle) \\ &= \Big(\frac{1}{\sqrt{2}}\sum_{c_1=0,1}(-1)^{b_1c_1}|c_1\rangle\Big)\otimes\dots\otimes\Big(\frac{1}{\sqrt{2}}\sum_{c_n=0,1}(-1)^{b_nc_n}|c_n\rangle\Big) \\ &= \frac{1}{2^{n/2}}\sum_{c_1\dots c_n\in\{0,1\}^n}(-1)^{\sum_{i=1}^n b_ic_i}|c_1\dots c_n\rangle \end{aligned} \tag{5}$$

Using this formula, we have

$$\begin{aligned} |\psi_3\rangle &= H^{\otimes n}|\psi_2\rangle \\ &= H^{\otimes n}\Big\{\frac{1}{2^{n/2}}\sum_{b_1\dots b_n\in\{0,1\}^n} e^{i\varphi(b_1,\dots,b_n)}|b_1b_2\dots b_n\rangle\Big\} \\ &= \frac{1}{2^{n/2}}\sum_{b_1\dots b_n\in\{0,1\}^n} e^{i\varphi(b_1,\dots,b_n)}H^{\otimes n}|b_1b_2\dots b_n\rangle, \qquad \text{(linearity)} \\ &= \frac{1}{2^{n/2}}\sum_{b_1\dots b_n\in\{0,1\}^n} e^{i\varphi(b_1,\dots,b_n)}\Big\{\frac{1}{2^{n/2}}\sum_{c_1\dots c_n\in\{0,1\}^n}(-1)^{\sum_{i=1}^n b_ic_i}|c_1\dots c_n\rangle\Big\} \end{aligned} \tag{6}$$

Interchanging two summations, we get

$$|\psi_3\rangle = \frac{1}{2^n}\sum_{c_1\dots c_n\in\{0,1\}^n}\Big\{\sum_{b_1\dots b_n\in\{0,1\}^n}(-1)^{\sum_{i=1}^n b_ic_i}e^{i\varphi(b_1,\dots,b_n)}\Big\}|c_1\dots c_n\rangle \tag{7}$$

c) Measuring $|\psi_3\rangle$ in the computational basis, we obtain a distribution on bit strings of length n with probabilities

$$\begin{aligned} p(\bar{c}_1,\dots,\bar{c}_n) &= |\langle\bar{c}_1,\dots,\bar{c}_n|\psi_3\rangle|^2 \\ &= \left|\frac{1}{2^n}\sum_{c_1\dots c_n\in\{0,1\}^n}\Big\{\sum_{b_1\dots b_n\in\{0,1\}^n}(-1)^{\sum_{i=1}^n b_ic_i}e^{i\varphi(b_1,\dots,b_n)}\Big\}\langle\bar{c}_1,\dots,\bar{c}_n|c_1\dots c_n\rangle\right|^2 \end{aligned} \tag{8}$$

$\langle\bar{c}_1,\dots,\bar{c}_n|c_1\dots c_n\rangle$ is 1 only when $\bar{c}_1 = c_1,\dots,\bar{c}_n = c_n$, otherwise it is zero. Thus, we have

$$p(\bar{c}_1,\dots,\bar{c}_n) = \frac{1}{2^{2n}}\left|\sum_{b_1\dots b_n\in\{0,1\}^n}(-1)^{\sum_{i=1}^n b_i\bar{c}_i}e^{i\varphi(b_1,\dots,b_n)}\right|^2 \tag{9}$$

2 Finding the matrix D

d) We have

$$Z|0\rangle = |0\rangle, \quad Z|1\rangle = -|1\rangle \tag{10}$$

So, $|b_i\rangle$ is an eigenvector of Z with eigenvalue $(-1)^{b_i}$. So, $|b_1b_2\rangle$ is the eigenvector of $Z_1\otimes Z_2$ with eigenvalue $(-1)^{b_1}(-1)^{b_2}$.

$$Z_1\otimes Z_2|b_1b_2\rangle = Z_1|b_1\rangle\otimes Z_2|b_2\rangle = (-1)^{b_1}|b_1\rangle\otimes(-1)^{b_2}|b_2\rangle = (-1)^{b_1}(-1)^{b_2}|b_1b_2\rangle \tag{11}$$

Definition of matrix exponential implies that $|b_1b_2\rangle$ is the eigenvector of $e^{i\theta Z_1\otimes Z_2}$ with eigenvalue $e^{i\theta(-1)^{b_1}(-1)^{b_2}}$.
One may use the fact that $Z_1 \otimes Z_2$ is diagonal and deduce that

$$Z_1 \otimes Z_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \Rightarrow e^{i\theta Z_1\otimes Z_2} = \begin{pmatrix} e^{i\theta} & 0 & 0 & 0 \\ 0 & e^{-i\theta} & 0 & 0 \\ 0 & 0 & e^{-i\theta} & 0 \\ 0 & 0 & 0 & e^{i\theta} \end{pmatrix} \tag{12}$$

So eigenvectors of $e^{i\theta Z_1\otimes Z_2}$ are $|b_1b_2\rangle$ with eigenvalues $e^{i\theta(-1)^{b_1}(-1)^{b_2}}$.

We have that $\varphi(b_1,\dots,b_n) = \theta\big(\sum_{i=1}^{n-1}(-1)^{b_i}(-1)^{b_{i+1}} + (-1)^{b_n}(-1)^{b_1}\big)$, so we deduce that D is the equivalent to applying the $e^{i\theta Z\otimes Z}$ on every two consecutive qubits q_i, q_{i+1} and q_n, q_1. So, we can write $D \equiv e^{i\theta Z_1\otimes Z_2}e^{i\theta Z_2\otimes Z_3}\dots e^{i\theta Z_{n-1}\otimes Z_n}e^{i\theta Z_n\otimes Z_1}$, where by $e^{i\theta Z_i\otimes Z_{i+1}}$ we mean the gate applies on qubits $i, i+1$ and leaves the rest of the qubits unchanged. Note that, since the matrices are diagonal the order of matrices (or gates) does not matter.

$$\begin{aligned}
D|b_1\dots b_n\rangle &= e^{i\theta Z_1\otimes Z_2}e^{i\theta Z_2\otimes Z_3}\dots e^{i\theta Z_{n-1}\otimes Z_n}e^{i\theta Z_n\otimes Z_1}|b_1\dots b_n\rangle \\
&= e^{i\theta Z_1\otimes Z_2}e^{i\theta Z_2\otimes Z_3}\dots e^{i\theta Z_{n-1}\otimes Z_n}\left(e^{i\theta(-1)^{b_n}(-1)^{b_1}}|b_1\dots b_n\rangle\right) \\
&= e^{i\theta(-1)^{b_n}(-1)^{b_1}}e^{i\theta Z_1\otimes Z_2}e^{i\theta Z_2\otimes Z_3}\dots e^{i\theta Z_{n-1}\otimes Z_n}|b_1\dots b_n\rangle \\
&= e^{i\theta(-1)^{b_n}(-1)^{b_1}}e^{i\theta(-1)^{b_{n-1}}(-1)^{b_n}}e^{i\theta Z_1\otimes Z_2}e^{i\theta Z_2\otimes Z_3}\dots e^{i\theta Z_{n-2}\otimes Z_{n-1}}|b_1\dots b_n\rangle \\
&\vdots \\
&= e^{i\theta(-1)^{b_1}(-1)^{b_2}}e^{i\theta(-1)^{b_2}(-1)^{b_3}}\dots e^{i\theta(-1)^{b_{n-1}}(-1)^{b_n}}e^{i\theta(-1)^{b_n}(-1)^{b_1}}|b_1\dots b_n\rangle \\
&= e^{i\theta\left(\sum_{i=1}^{n-1}(-1)^{b_i}(-1)^{b_{i+1}}+(-1)^{b_n}(-1)^{b_1}\right)}|b_1\dots b_n\rangle \\
&= e^{i\varphi(b_1,\dots,b_n)}|b_1\dots b_n\rangle
\end{aligned} \tag{13}$$

e)

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \qquad I\otimes R(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & e^{-2i\theta} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{-2i\theta} \end{pmatrix}$$

$$\begin{aligned}
CNOT(I\otimes R(\theta))CNOT &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & e^{-2i\theta} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{-2i\theta} \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & e^{-2i\theta} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & e^{-2i\theta} & 0 \end{pmatrix} \\
&= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & e^{-2i\theta} & 0 & 0 \\ 0 & 0 & e^{-2i\theta} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
\end{aligned} \tag{14}$$

$$\Rightarrow e^{i\theta}CNOT(I \otimes R(\theta))CNOT = \begin{pmatrix} e^{i\theta} & 0 & 0 & 0 \\ 0 & e^{-i\theta} & 0 & 0 \\ 0 & 0 & e^{-i\theta} & 0 \\ 0 & 0 & 0 & e^{i\theta} \end{pmatrix} = e^{i\theta Z \otimes Z} \tag{15}$$

3 Implementation on the IBM Q experience

f) Circuit for $n = 3$ and $\theta = \pi/3$ is shown in figure (1).

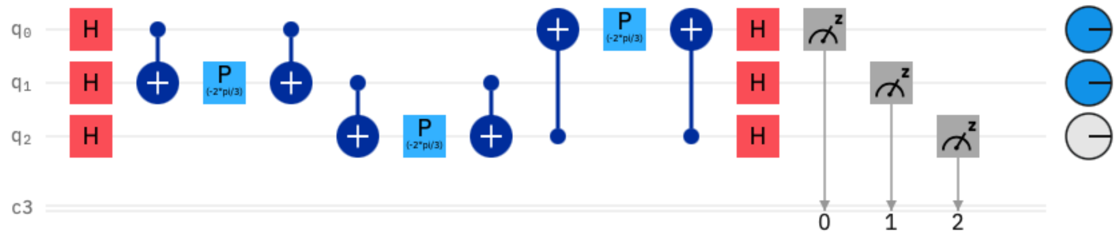


Figure 1: Circuit for $n = 3, \theta = \pi/3$

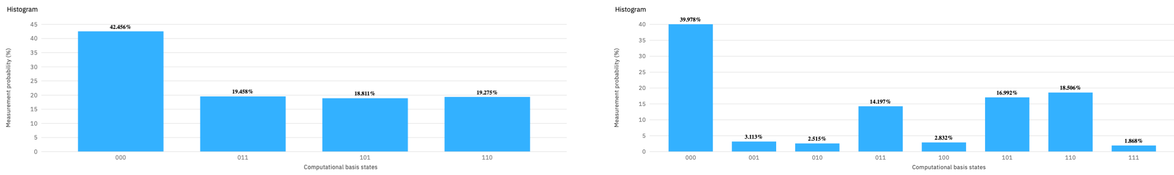
Simulation and experiment (from *ibmq_athens* machine) results for $n = 3, 4$ and $\theta = \pi/3, \pi/5$, and for 8192 shots, are shown in figure (2).

Comparing the plots, we see that in the experiment, the measured probabilities are quite different from simulations. Also, there are states such that in simulation, the probability of measuring them is zero. However, by running experiments on a real machine, these states have been measured. This observation indicates that the real quantum machines are noisy.

g)First, we briefly explain the Qiskit code to run the simulations and experiments.

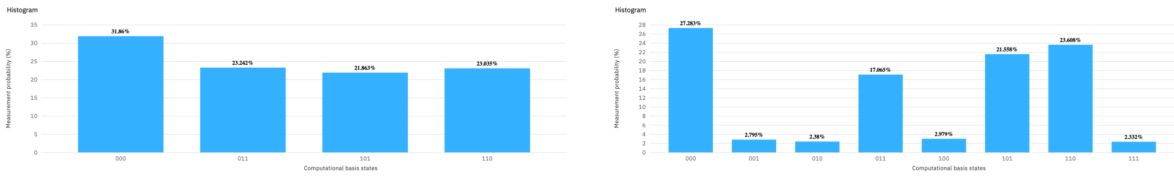
First part is to import necessary libraries.

```
import numpy as np
import matplotlib.pyplot as plt
from math import pi
from qiskit import QuantumCircuit, QuantumRegister, ClassicalRegister, execute,␣
↪BasicAer, IBMQ
from qiskit.providers.ibmq import least_busy
```



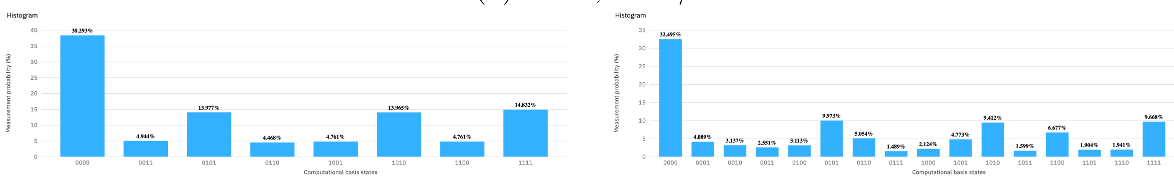
Simulation Experiment

(a) $n = 3, \theta = \pi/3$



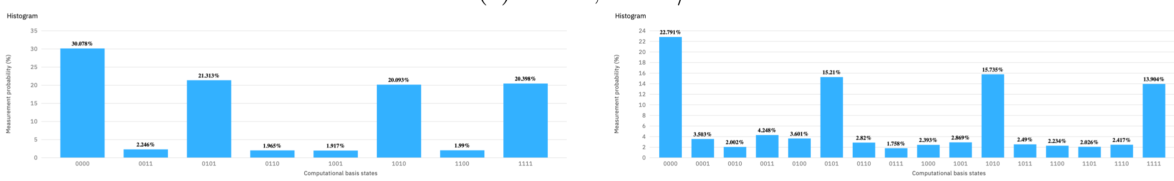
Simulation Experiment

(b) $n = 3, \theta = \pi/5$



Simulation Experiment

(c) $n = 4, \theta = \pi/3$



Simulation Experiment

(d) $n = 4, \theta = \pi/5$

Figure 2: Part **f**. Simulation and experiment results for $n = 3, 4$ and $\theta = \pi/3, \pi/5$

The function to construct a circuit for a given n and θ is as follows. The first and the third for loops place the series of Hadamard gates. The second loop puts the $CNOT(I \otimes R(\theta))CNOT$ for each of two consecutive qubits. The last for loop inserts the measurements gates for each of the qubits.

```
def sampling_circuit(n, Th):

    qreg_q = QuantumRegister(n, name='q')
    creg_c = ClassicalRegister(n, 'c')
    circuit = QuantumCircuit(qreg_q, creg_c)

    for i in range(n):
        circuit.h(qreg_q[i])

    for i in range(n-1):
        circuit.cx(qreg_q[i], qreg_q[i+1])
        circuit.p(-2*Th, qreg_q[i+1])
        circuit.cx(qreg_q[i], qreg_q[i+1])

    circuit.cx(qreg_q[n-1], qreg_q[0])
    circuit.p(-2*Th, qreg_q[0])
    circuit.cx(qreg_q[n-1], qreg_q[0])

    for i in range(n):
        circuit.h(qreg_q[i])

    for i in range(n):
        circuit.measure(qreg_q[i], creg_c[i])

    return circuit
```

Then, we construct a list of circuits for a given n, for various values of θ.

```
n = 4
circs = []
for i in range(33):
    circs.append(sampling_circuit(n,i*pi/32))
```

Once we obtain the list of circuits, we can run our simulations. For this, we use *qasm_simulator*. The simulation results for each circuit are a dictionary with bit strings of length n as keys (here $n = 4$). For each bit string, the value is the number of shots that it has been measured. Thus, by dividing by the number of shots, we get the probability.

```python
simulator_prob =  np.zeros(33)
backend = BasicAer.get_backend('qasm_simulator')
simul = execute(circs, backend, shots=8192)
results = simul.result()
for i in range(33):
    sim_measurement_result = results.get_counts(i)
    if '0000' in sim_measurement_result:
        simulator_prob[i] = sim_measurement_result['0000']/8192
    else:
        simulator_prob[i] = 0
```

To run the experiments on a real machine, we first need to load our accounts using the token we are given when creating our account. Then, we need to determine the machine we want to use. We can set it manually (the commented line) or use the least busy machine with as least n qubits.

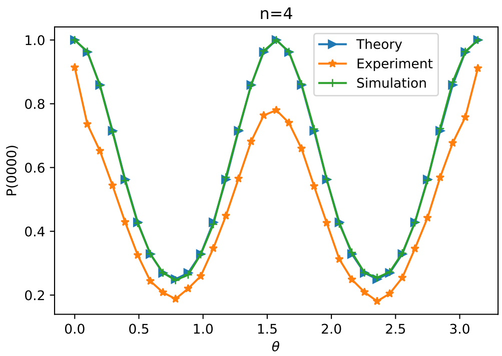
```python
IBMQ.load_account()
provider = IBMQ.get_provider(hub='ibm-q')
#backend = provider.backends.ibmq_16_melbourne
provider.backends()
backend = least_busy(provider.backends(filters=lambda b: b.configuration().
 ↪n_qubits >= n and not b.configuration().simulator and b.status().
 ↪operational==True))
job_set_exp = execute(circs, backend=backend, shots=8192)
results = job_set_exp.result()
```

Once we have our results, we can compute $P(0\dots0)$ the same way as simulation results.

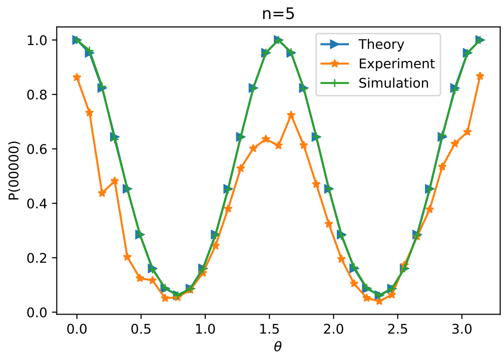
```python
exp_prob =  np.zeros(33)
for i in range(33):
    exp_measurement_result = results.get_counts(i)
    if '0000' in exp_measurement_result:
        exp_prob[i] = exp_measurement_result['0000']/8192
    else:
        exp_prob[i] = 0
```

Now, we have the simulation and experiment probabilities and we compare them with the theoretical probability $P(0\dots0) = |cos(\theta)^n + i^n sin(\theta)^n|^2$. Plots for different values of n are shown in figure 3.

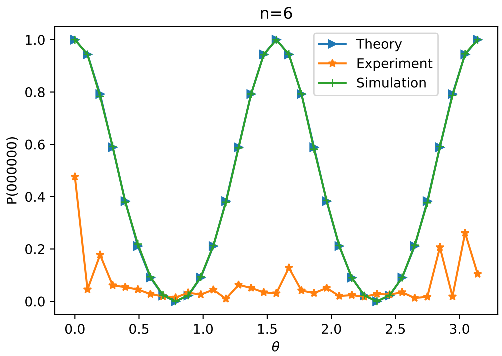
From the figures, we can see that the simulation probabilities perfectly match the theoretical one (since the number of shots is large enough). In experiment, for $n = 4, 5$ experiments are executed on *ibmq_athens*, and $P(0, \dots, 0)$ is close to theoretical and have similar curve. However, for $n \geq 6$, the results are too noisy. Experiments for $n \geq 6$ are executed on *ibmq_melbourne*, which is the only available machine for $n > 5$. To compare the quality of this machine, we run the experiment for $n = 4$ on this machine. From figure 4, we can see that this machine has poor performance comparing to other machines.



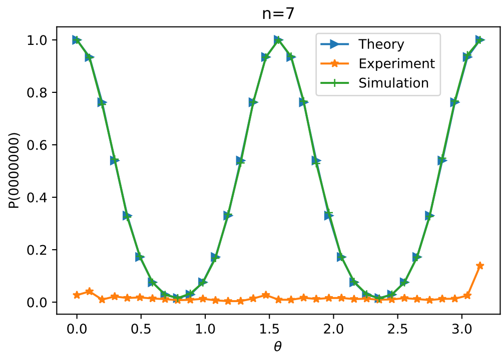
(a) $n = 4$



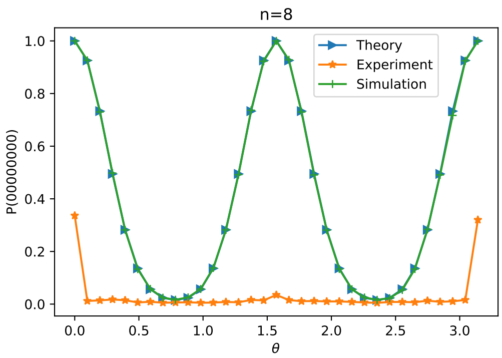
(b) $n = 5$



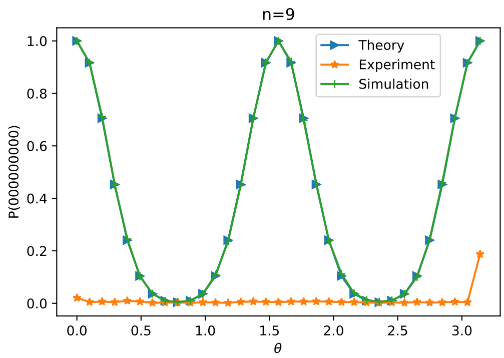
(c) $n = 6$



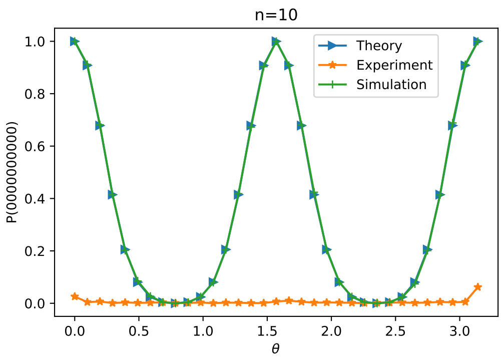
(d) $n = 7$



(e) $n = 8$



(f) $n = 9$



(g) $n = 10$

Figure 3: Part **g**. $P(0\ldots0)$ for $n = 4, \ldots, 10$, $\theta \in [0, \pi]$

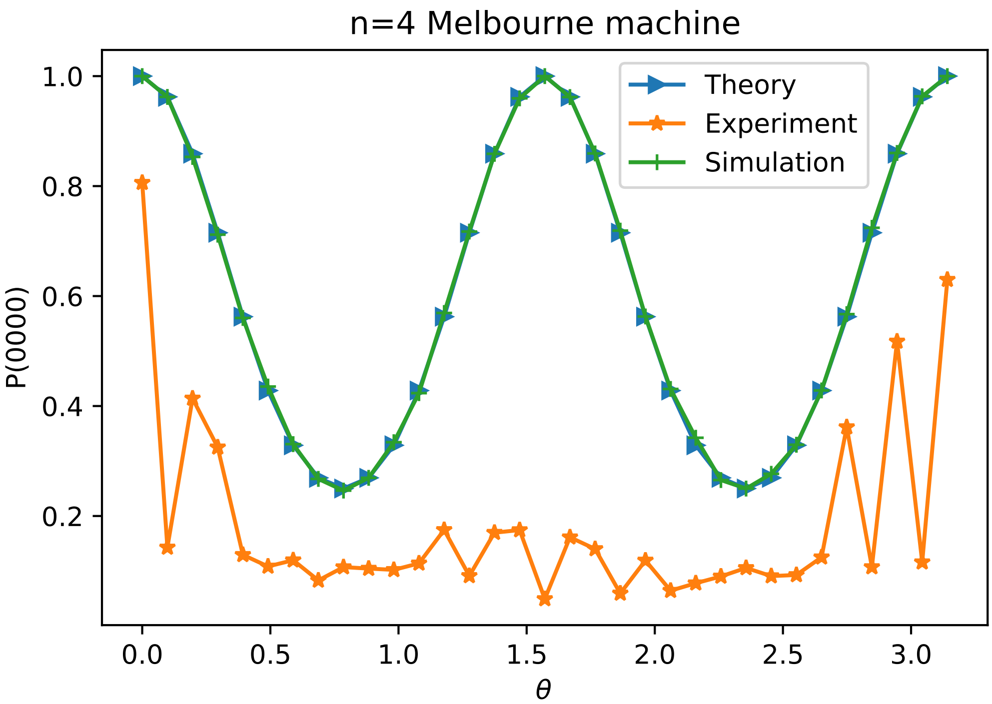


Figure 4: $P(0000)$ on *ibmq_melbourne*