CPEN 400Q Lecture 04
Basis rotations; entanglement and multi-qubit systems

Wednesday 15 January 2025

Announcements

- Assignment 1 due Sunday 26 Jan at 23:59
- Tutorial hands-on due Friday 17 Jan at 23:59
  - Bonus TA office hour Friday 2-3:30, KAIS 4037
- Quiz 2 on Monday (covers L3, L4)
- Will make Piazza thread for tutorial topic suggestions

We introduced the "bra" part of the "bra-ket notation"

The inner product between two states is defined as

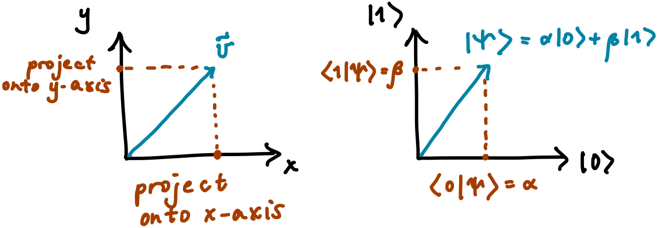
Inner product tells about the *overlap* (similarity) between states.

Last time

We introduced the concept of *orthonormal bases* for qubit states:

Examples:

We discussed *projective measurement* with respect to a basis.



When we measure state $|\varphi\rangle$ with respect to basis $\{|\psi_i\rangle\}$, the probability of obtaining outcome i is

Image credit: Xanadu Quantum Codebook I.9

Projective measurements can be performed with respect to any orthonormal basis. For example, $\{|+\rangle, |-\rangle\}$:

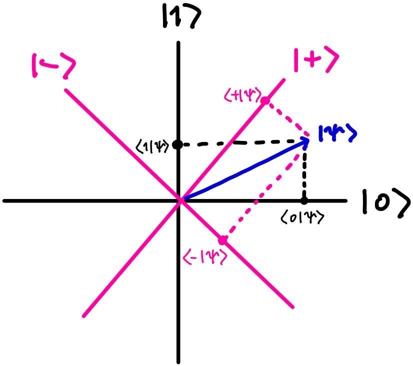


Image credit: Codebook node I.9

Learning outcomes

- Measure a qubit in different bases
- Mathematically describe a system of multiple qubits
- Describe the action of common multi-qubit gates
- Make any gate a controlled gate
- Perform measurements on multiple qubits

Basis rotations

So far we've seen 3 ways of extracting information out of a QNode:

1. `qml.state()`
2. `qml.probs(wires=x)`
3. `qml.sample()`

But these return results of measurements in the computational basis; and most hardware only allows for this.

How can we measure with respect to *different bases*?

Use a basis rotation to “trick” the quantum computer.

Suppose we want to measure in the “Y” basis:

$$|p\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + i|1\rangle\right), \quad |m\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - i|1\rangle\right).$$

Recall: unitary operations preserve length *and* angles between normalized quantum state vectors (prove on A1!)

There exists a unitary operation that will convert between this basis and the computational basis.

Basis rotations

Exercise: determine a quantum circuit that sends

$$\begin{aligned} |0\rangle \quad &\rightarrow \quad |p\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + i|1\rangle\right) \\ |1\rangle \quad &\rightarrow \quad |m\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - i|1\rangle\right) \end{aligned}$$

Basis rotations

At the end of our circuit, apply the reverse (adjoint) of this transformation to rotate *back* to the computational basis.

If we measure and observe $|0\rangle$, we know the qubit was previously $|p\rangle$ in the Y basis (analogous result for $|m\rangle$).

Adjoints

In PennyLane, we can compute adjoints of operations *and* entire quantum functions using `qml.adjoint`:

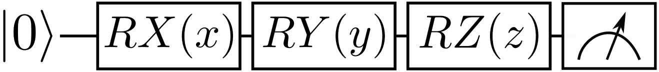
```
def some_function(x):
    qml.RZ(Z, wires=0)

def apply_adjoint(x):
    qml.adjoint(qml.S)(wires=0)
    qml.adjoint(some_function)(x)
```

`qml.adjoint` is a special type of function called a **transform**. We will cover transforms in more detail later in the course.

Basis rotations: hands-on

Let's run the following circuit, and measure in the Y basis



Hands-on time...

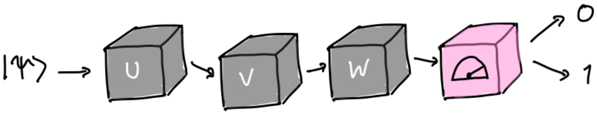
Multi-qubit systems

Recall this slide from lecture 1...

Quantum computing

Quantum computing is the act of manipulating the state of qubits in a way that represents solution of a computational problem:

1. **Prepare** qubits in a **superposition**
2. Apply **operations** that **entangle** the qubits and manipulate the amplitudes
3. **Measure** qubits to extract an answer
4. Profit



Let's simulate this using NumPy.

How do we express the mathematical space of multiple qubits?

Hilbert spaces compose under the *tensor product*.

Let

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}, \;\; B = \begin{pmatrix} e & f \\ g & h \end{pmatrix}.$$

The tensor product of A and B, $A \otimes B$ is

$$A \otimes B = \begin{pmatrix} a\begin{pmatrix} e & f \\ g & h \end{pmatrix} & b\begin{pmatrix} e & f \\ g & h \end{pmatrix} \\ c\begin{pmatrix} e & f \\ g & h \end{pmatrix} & d\begin{pmatrix} e & f \\ g & h \end{pmatrix} \end{pmatrix} = \begin{pmatrix} ae & af & be & bf \\ ag & ah & bg & bh \\ ce & cf & de & df \\ cg & ch & dg & dh \end{pmatrix}$$

Multi-qubit systems

Qubit state vectors also combine under the *tensor product*:

The states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ are the computational basis vectors for 2 qubits:

$$|00\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad |01\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \quad |10\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \quad |11\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

We can create arbitrary linear combinations of them (coefficients must still be normalized).

Same pattern for 3 qubits: $|000\rangle, |001\rangle, \ldots, |111\rangle$.

The tensor product is linear and distributive. Given

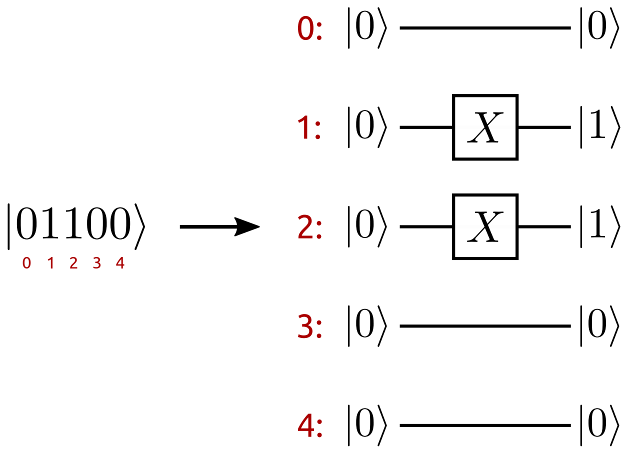
$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad |\varphi\rangle = \gamma|0\rangle + \delta|1\rangle,$$

Multi-qubit systems

Unitary operations also compose under tensor product.

For example, apply U_1 to qubit $|\psi\rangle$ and U_2 to qubit $|\varphi\rangle$:

In PennyLane:



(This is different in other frameworks!)

Exercise: determine the state of a 3-qubit system if H is applied to qubit 0, X and then S are applied to qubit 1.

Multi-qubit measurement outcome probabilities

$$|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$$

If we measure in the computational basis, the outcome probabilities are:

- $|\alpha|^2 = |\langle 00|\psi\rangle|^2$ for $|00\rangle$
- $|\beta|^2 = |\langle 01|\psi\rangle|^2$ for $|01\rangle$
- ...

Exercise: what is the probability of the qubits being in state $|110\rangle$ after measuring $(H \otimes SX \otimes I)|000\rangle$ in the computational basis?

$$|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$$

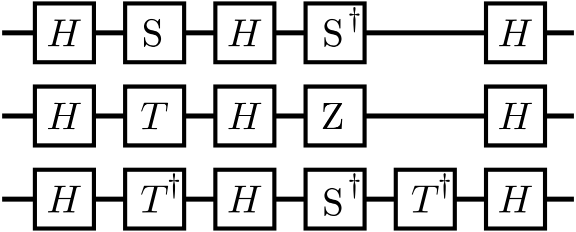
We can also measure *just one qubit*:

- The probability of the first qubit being in state $|0\rangle$ is $|\alpha|^2 + |\beta|^2$
- The probability of the second qubit being in state $|1\rangle$ is $|\beta|^2 + |\delta|^2$

Exercise: what is the probability of the second qubit being in state $|0\rangle$ after measuring $(H \otimes SX \otimes I)|000\rangle$ in the computational basis?

Multi-qubit gates

The circuits we've seen so far only involve single-qubit gates:



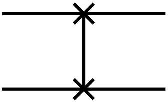
Surely this isn't all we can do...

Image credit: Xanadu Quantum Codebook I.11

SWAP

We can swap the state of two qubits using the SWAP operation.
First define what it does to the basis states...

Circuit element:

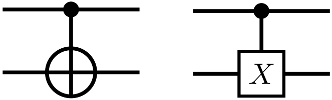


PennyLane: `qml.SWAP`

CNOT

CNOT = "controlled-NOT". A NOT (X) is applied to second qubit only if first qubit is in state $|1\rangle$.

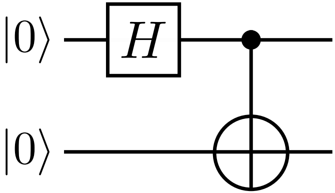
Circuit elements:



PennyLane: `qml.CNOT`

Image credit: Xanadu Quantum Codebook I.12

Exercise: What does CNOT do to qubits in a superposition? Determine the output state of this circuit, and the measurement outcome probabilities of the computational basis states.

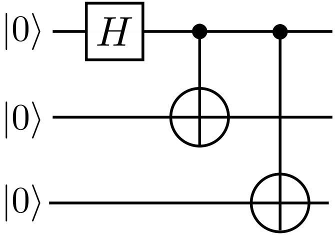


We *cannot express*

as a tensor product of two single-qubit states. Moreover, the measurement outcomes are correlated!

This state is **entangled**, and CNOT is an **entangling gate**!

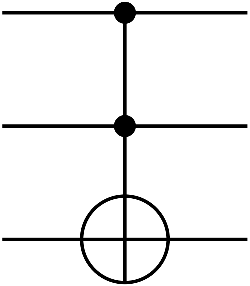
Entanglement generalizes to more than two qubits:



Exercise: Express the output state of this circuit in the computational basis.

Toffoli

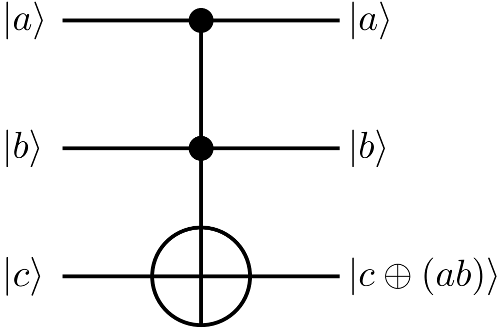
There are also gates on more than two qubits, like the Toffoli gate, which is a controlled-CNOT, or controlled-controlled-NOT.



$$TOF|000\rangle =$$
$$TOF|001\rangle =$$
$$TOF|010\rangle =$$
$$TOF|011\rangle =$$
$$TOF|100\rangle =$$
$$TOF|101\rangle =$$
$$TOF|110\rangle =$$
$$TOF|111\rangle =$$

PennyLane: `qml.Toffoli`

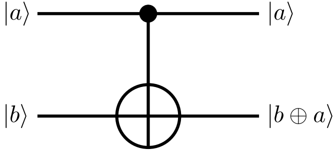
The Toffoli implements a reversible AND gate.



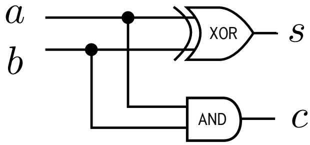
It is also universal for classical reversible computing.

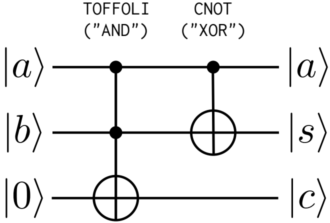
CNOT, Toffoli, and classical reversible circuits

CNOT also implements a reversible Boolean function.



X, CNOT, TOF can be used to create Boolean arithmetic circuits.

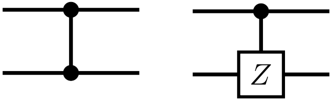




Fun for you: assignment problem 3 & 6.

Example: controlled-Z (CZ)

What does this operation do?



PennyLane: `qml.CZ`

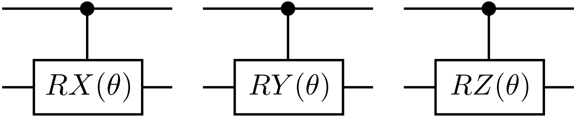
Image credit: Codebook node I.13

Example: controlled rotations (RX, RY, RZ)

Or this one?

$$CRY(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ 0 & 0 & \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix}$$

Circuit elements:



PennyLane: `qml.CRX`, `qml.CRY`, `qml.CRZ`

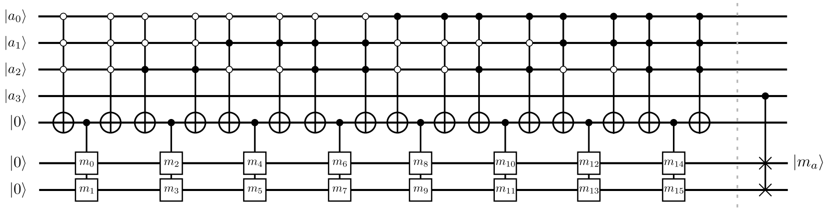
Image credit: Codebook node I.14

There is a pattern here:

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad CRY(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ 0 & 0 & \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix}$$

More generally,

Any unitary operation can be turned into a controlled operation, controlled on any state.



Most common controls are controlled-on-$|1\rangle$ (filled circle), and controlled-on-$|0\rangle$ (empty circle).

Hands-on: `qml.ctrl`

Remember `qml.adjoint` from last class:

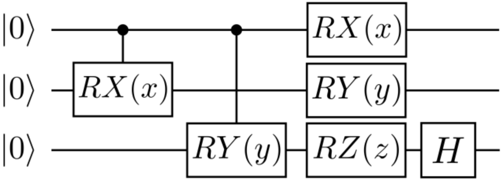
```
@qml.qnode(dev)
def my_circuit():
    qml.S(wires=0)
    qml.adjoint(qml.S)(wires=0)
    return qml.sample()
```

There is a similar *transform* that allows us to perform arbitrary controlled operations (or entire quantum functions)!

```
@qml.qnode(dev)
def my_circuit():
    qml.S(wires=0)
    qml.ctrl(qml.S, control=1)(wires=0)
    return qml.sample()
```

Hands-on: `qml.ctrl`

Let's go implement this circuit:



Next time

Content:

- Superdense coding
- The no-cloning theorem
- Quantum teleportation

Action items:

1. Work on Assignment 1 (can do 1 & 5, 2ab, 3 & 6 now)
2. Quiz 2 Monday about contents from this week

Recommended reading:

- For this week: Codebook IQC, SQ, MQ; Nielsen & Chuang 1.2, 1.3.1-1.3.5, 2.2.3-2.2.5, 4.3
- For next week: Codebook SQ (what did you expect?), MQ (all tied up); Nielsen & Chuang 1.3.6, 1.3.7, 2.3