

CPEN 400Q Lecture 04
**More on measurement; multi-qubit states
and gates**

Friday 20 January 2023

Announcements

- Literacy assignment 1 due Wednesday 25 Jan at 23:59
- Technical assignment 1 released over next few days (will be due in 2 weeks)
- Short class on Monday
 - Need to leave at 16:00
 - Quiz 2 will be at the *end* of class

We introduced the “bra” part of the “bra-ket notation”

The inner product between two states is defined as

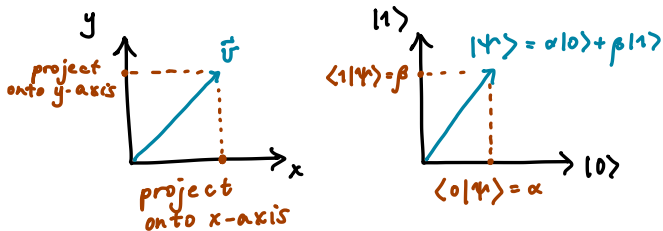
Inner product tells about the *overlap* (similarity) between states.

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

Examples:

Last time

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

Learning outcomes

- Measure a single 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

Note: moving expectation values (originally L3) to a future lecture.

Basis rotations

Exercise: consider the quantum state

$$|\psi\rangle = \frac{\sqrt{3}}{2}|0\rangle - \frac{1}{2}e^{i\frac{5}{4}}|1\rangle$$

If we measure in the $\{|p\rangle, |m\rangle\}$ basis, what are the measurement probabilities of the possible outcomes?

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

Basis rotations

Solution:

Basis rotations

Tedious... let's use software. But how?

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

Unitary operations preserve length *and* angles between normalized quantum state vectors; there exists some unitary transformation that will convert between any basis and the computational basis.

Basis rotations

Exercise: Suppose we want to measure in the “Y” basis. Determine a quantum circuit that sends

$$|0\rangle \rightarrow |p\rangle = \frac{1}{\sqrt{2}} (|0\rangle + i|1\rangle)$$

$$|1\rangle \rightarrow |m\rangle = \frac{1}{\sqrt{2}} (|0\rangle - i|1\rangle)$$

Basis rotations

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

Hands-on: 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.

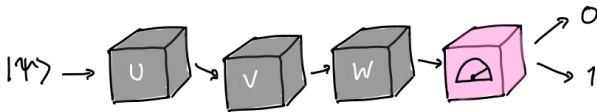
Multi-qubit systems

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?

Tensor products

Hilbert spaces compose under the *tensor product*.

Let

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

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

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

Qubit state vectors are also combined using the *tensor product*:

$$|01\rangle = |0\rangle \otimes |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ 0 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

An n -qubit state is therefore a vector of length 2^n .

Multi-qubit systems

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 as long as the normalization on the coefficients holds.

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

The tensor product is linear and distributive, so if we have

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

then they tensor together to form

Multi-qubit systems

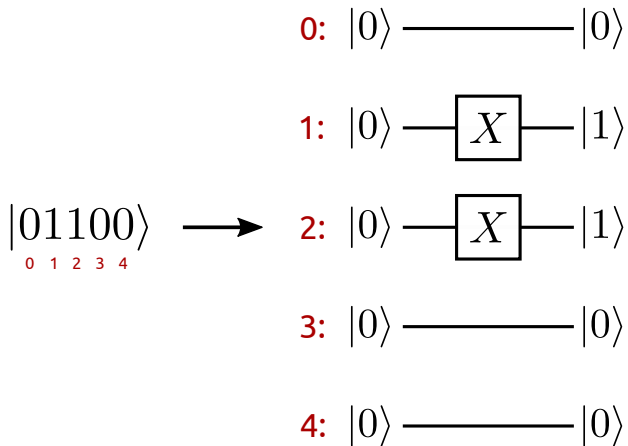
Single-qubit unitary operations also compose under tensor product.

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

If an n -qubit ket is a vector with length 2^n , then a unitary acting on n qubits has dimension $2^n \times 2^n$.

Qubit ordering (very important!)

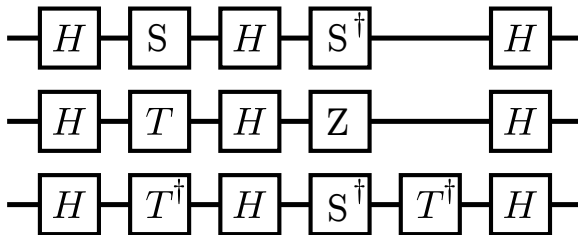
In PennyLane:



(This is different in other frameworks!)

Multi-qubit gates

The few small circuits we've seen so far only involve gates on single qubits:



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...

$$SWAP|00\rangle = |00\rangle$$

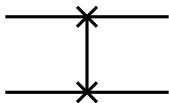
$$SWAP|01\rangle = |10\rangle$$

$$SWAP|10\rangle = |01\rangle$$

$$SWAP|11\rangle = |11\rangle$$

$$SWAP = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Circuit element:



PennyLane: `qml.SWAP`

SWAP

More generally,

$$SWAP(|\psi\rangle \otimes |\phi\rangle) = |\phi\rangle \otimes |\psi\rangle$$

Let's show this. Start by writing

$$\begin{aligned} |\psi\rangle \otimes |\phi\rangle &= (\alpha|0\rangle + \beta|1\rangle) \otimes (\gamma|0\rangle + \delta|1\rangle) \\ &= \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle \end{aligned}$$

Now apply the SWAP:

Consider a two-qubit operation U with the following action on the basis states:

$$U|00\rangle = |00\rangle$$

$$U|01\rangle = |01\rangle$$

$$U|10\rangle = |11\rangle$$

$$U|11\rangle = |10\rangle$$

CNOT

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

$$CNOT|00\rangle = |00\rangle$$

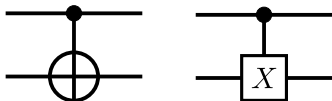
$$CNOT|01\rangle = |01\rangle$$

$$CNOT|10\rangle = |11\rangle$$

$$CNOT|11\rangle = |10\rangle$$

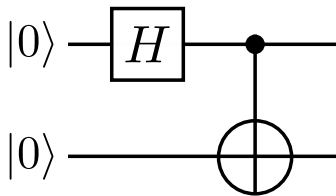
$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Circuit elements:

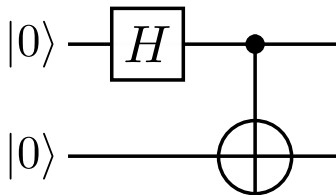


PennyLane: `qml.CNOT`

What does CNOT do with qubits in a superposition?



CNOT hands-on



The output state of this circuit is:

$$CNOT \cdot (H \otimes I) |00\rangle = \frac{1}{\sqrt{2}} (|00\rangle + |11\rangle)$$

This state is **entangled**!

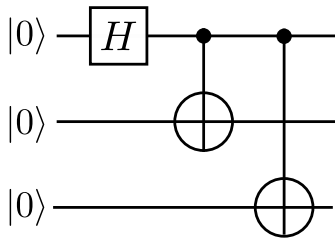
We *cannot* express

$$\frac{1}{\sqrt{2}} (|00\rangle + |11\rangle)$$

as a tensor product of two single-qubit states.

Entanglement

Entanglement generalizes to more than two qubits:



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

Consider the AND of two bits a and b :

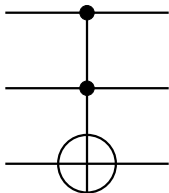
a	b	ab
0	0	0
0	1	0
1	0	0
1	1	1

This gate is *not* reversible: we cannot recover inputs from outputs.

But, we can make it reversible by adding one extra bit...

Toffoli

The **Toffoli** implements a reversible AND gate. (It is universal for classical reversible computing).



Controlled-CNOT, or controlled-controlled-NOT.

PennyLane: `qml.Toffoli`

What does it do to the basis states?

$$TOF|000\rangle =$$

$$TOF|001\rangle =$$

$$TOF|010\rangle =$$

$$TOF|011\rangle =$$

$$TOF|100\rangle =$$

$$TOF|101\rangle =$$

$$TOF|110\rangle =$$

$$TOF|111\rangle =$$

What is actually going on here?

$$TOF|000\rangle = |000\rangle$$

$$TOF|001\rangle = |001\rangle$$

$$TOF|010\rangle = |010\rangle$$

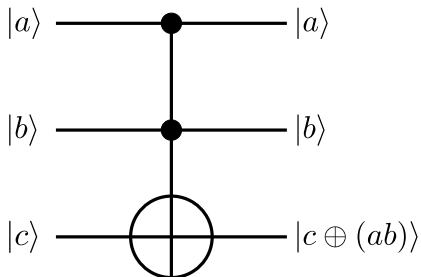
$$TOF|011\rangle = |011\rangle$$

$$TOF|100\rangle = |100\rangle$$

$$TOF|101\rangle = |101\rangle$$

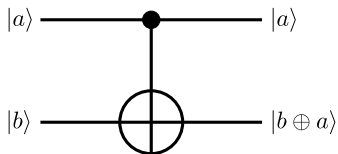
$$TOF|110\rangle = |111\rangle$$

$$TOF|111\rangle = |110\rangle$$

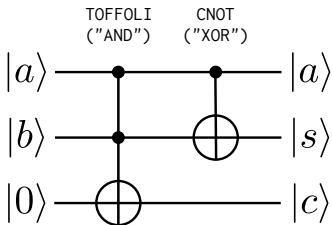
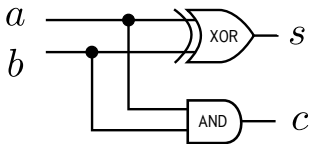


Hands-on: the half-adder

We can interpret CNOT in a similar way.

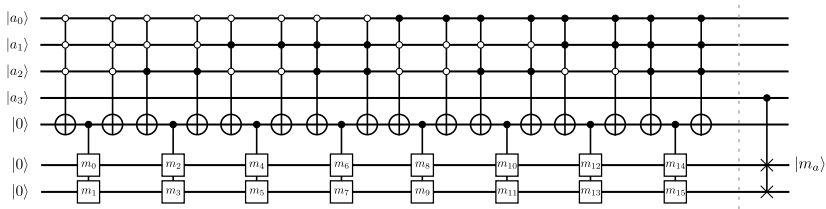


X, CNOT, TOF can be used to create Boolean arithmetic circuits.



Controlled unitary operations

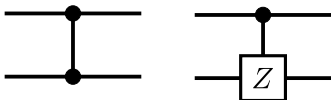
Any unitary operation can be turned into a controlled operation, controlled on any state.



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

Example: controlled- Z (CZ)

What does this operation do?



PennyLane: `qml.CZ`

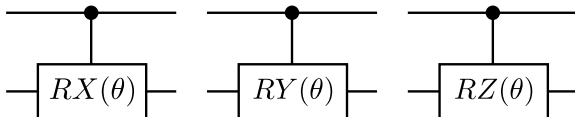
Image credit: Codebook node I.13

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`

Controlled- U

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,

$$CU = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{pmatrix} = \begin{pmatrix} I_2 & \mathbf{0}_2 \\ \mathbf{0}_2 & U \end{pmatrix}$$

... we don't want to be writing these matrices all the time.

Hands-on: qml.ctrl

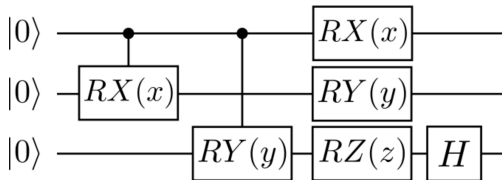
Remember from earlier, `qml.adjoint`:

```
@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()
```

Let's go implement this circuit:



Universal gate sets

Last class, we learned that with just

- H and T
- any two of RX , RY , and RZ ,

we can implement *any* single-qubit unitary operation up to arbitrary precision.

What about for two qubits?

What about for two qubits?

- H , T , and $CNOT$
- any two of RX , RY , RZ , and $CNOT$
- H and TOF

With just 2-3 gates, we can implement *any* two-qubit unitary operation up to arbitrary precision.

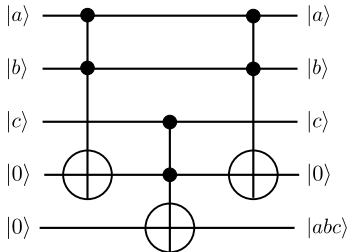
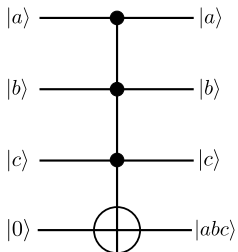
What about three or more qubits? (Same thing!)

In general, finding such an implementation (*quantum circuit synthesis*, part of the quantum compilation pipeline) is computationally hard.

- sometimes we can do so for small cases (PennyLane has many decompositions pre-programmed)
- sometimes having **auxiliary** qubits around can simplify the decomposition

Auxiliary qubits

Auxiliary qubits are like “scratch”, or “work” qubits. They start in state $|0\rangle$, and must be returned to state $|0\rangle$, but can be used to store intermediate results in a computation.



Review: single-qubit measurements

Given a state

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

- the probability of measuring and observing the qubit in state $|0\rangle$ is $|\alpha|^2 = \alpha\alpha^* = |\langle 0|\psi\rangle|^2$
- the probability of measuring and observing the qubit in state $|1\rangle$ is $|\beta|^2 = |\langle 1|\psi\rangle|^2$
- we can measure in different bases by “remapping” those basis states to the computational basis

We can do all this in the multi-qubit case as well.

Multi-qubit measurement outcome probabilities

Let

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

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 $|01\rangle$
- ...

Multi-qubit measurement outcome probabilities

Let

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

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 $|\beta|^2 + |\delta|^2$

Multi-qubit measurement outcome probabilities

Let

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

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 $|\beta|^2 + |\delta|^2$

Recap

- Measure a single 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

Next time

Content:

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

Action items:

1. Literacy assignment 1 due Wednesday
2. Technical assignment 1 posted soon
3. Quiz 2 Monday end of class on contents from this week

Recommended reading:

- From this class: Codebook nodes I.9, I.11-I.14
- For next class: Codebook nodes I.15