Phase estimation procedure

Next, we'll discuss the *phase-estimation procedure*, which is a quantum algorithm for solving the phase estimation problem.

We'll begin with a low-precision warm-up, which explains some of the basic intuition behind the method. We'll then talk about the *quantum Fourier transform*, which is an important quantum operation used in the phase-estimation procedure, as well as its quantum circuit implementation. Once we have the quantum Fourier transform in hand, we'll describe the phase-estimation procedure in full generality and analyze its performance.

Warm-up: approximating phases with low precision

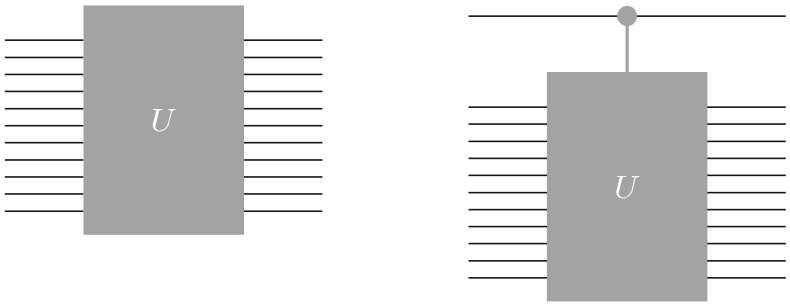
We'll begin with a couple of simple versions of the phase-estimation procedure that provide low-precision solutions to the phase-estimation problem. This is helpful for explaining the intuition behind the general procedure that we'll see a bit later in the lesson.

Using the phase kickback

A simple approach to the phase-estimation problem, which allows us to learn something about the value θ we seek, is based on the *phase kick-back* phenomenon. As we'll see, this is essentially a single-qubit version of the general phase-estimation procedure to be discussed later in the lesson.

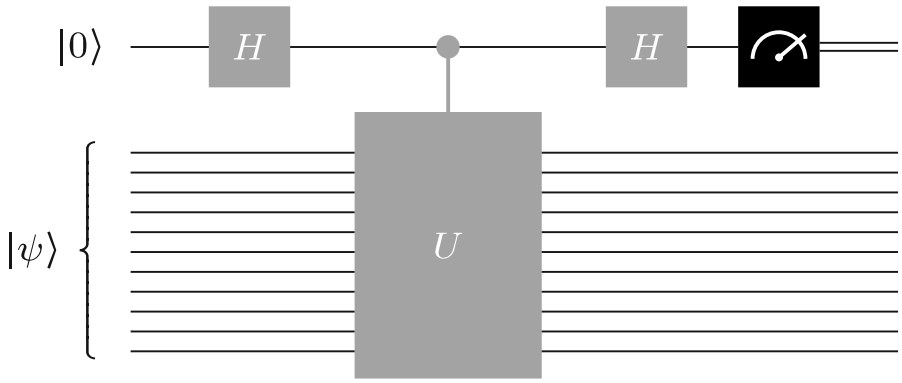
As part of the input to the phase estimation problem, we have a unitary quantum circuit for the operation U. We can use the description of this circuit to create a circuit for a *controlled-U* operation, which can be

depicted as this figure suggests (with the operation U, viewed as a quantum gate, on the left and a controlled-U operation on the right).

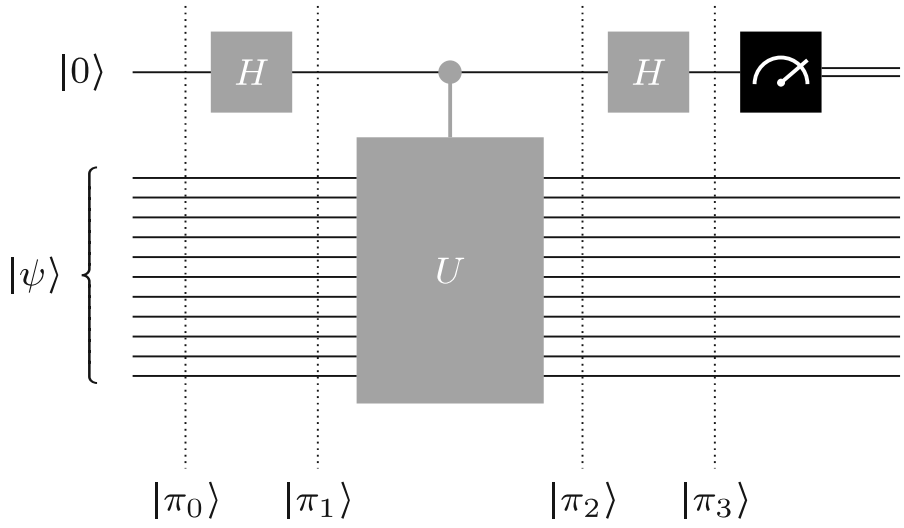


We can create a quantum circuit for a controlled-U operation by first adding a control qubit to the circuit for U, and then replacing every gate in the circuit for U with a controlled version of that gate — so our one new control qubit effectively controls every single gate in the circuit for U. This requires that we have a controlled version of every gate in our circuit, but we can always build circuits for these controlled operations in case they're not included in our gate set.

Now consider the following circuit, where the input state $|\psi\rangle$ of all of the qubits except the top one is the quantum state eigenvector of U.



The measurement outcome probabilities for this circuit depend on the eigenvalue of U corresponding to the eigenvector $|\psi\rangle$. Let's analyze the circuit in detail to determine exactly how.



The initial state of the circuit is

$$|\pi_0\rangle = |\psi\rangle|0\rangle$$

and the first Hadamard gate transforms this state to

$$|\pi_1\rangle = |\psi\rangle|+\rangle = \frac{1}{\sqrt{2}}|\psi\rangle|0\rangle + \frac{1}{\sqrt{2}}|\psi\rangle|1\rangle.$$

Next, the controlled-U operation is performed, which results in the state

$$|\pi_2\rangle = \frac{1}{\sqrt{2}}|\psi\rangle|0\rangle + \frac{1}{\sqrt{2}}\big(U|\psi\rangle\big)|1\rangle.$$

Using the assumption that $|\psi\rangle$ is an eigenvector of U having eigenvalue $\lambda = e^{2\pi i\theta}$, we can alternatively express this state as follows.

$$|\pi_2\rangle = \frac{1}{\sqrt{2}}|\psi\rangle|0\rangle + \frac{e^{2\pi i\theta}}{\sqrt{2}}|\psi\rangle|1\rangle = |\psi\rangle \otimes \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{e^{2\pi i\theta}}{\sqrt{2}}|1\rangle\right)$$

Here we observe the phase kickback phenomenon. It is slightly different this time than it was for Deutsch's algorithm and the Deutsch-Jozsa algorithm because we're not working with a query gate — but the idea is similar.

Finally, the second Hadamard gate is performed. After just a bit of simplification, we obtain this expression for this state.

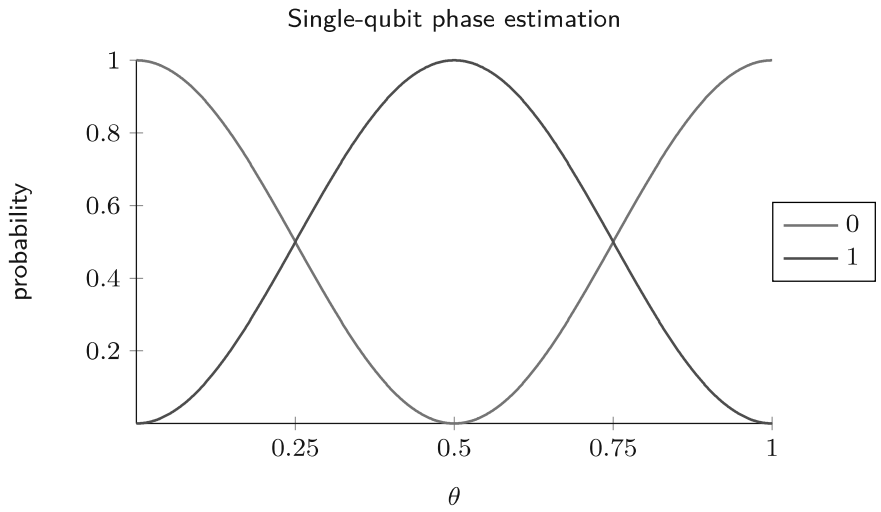
$$|\pi_3\rangle = |\psi\rangle \otimes \left(\frac{1 + e^{2\pi i\theta}}{2}|0\rangle + \frac{1 - e^{2\pi i\theta}}{2}|1\rangle\right)$$

The measurement therefore yields the outcomes 0 and 1 with these probabilities:

$$p_0 = \left| \frac{1 + e^{2\pi i \theta}}{2} \right|^2 = \cos^2(\pi\theta)$$

$$p_1 = \left| \frac{1 - e^{2\pi i \theta}}{2} \right|^2 = \sin^2(\pi\theta).$$

Here's a plot of the probabilities for the two possible outcomes, 0 and 1, as functions of θ.



Naturally, the two probabilities always sum to 1. Notice that when $\theta = 0$, the measurement outcome is always 0, and when $\theta = 1/2$, the measurement outcome is always 1. So, although the measurement result doesn't reveal exactly what θ is, it does provide us with some information about it — and if we were promised that either $\theta = 0$ or $\theta = 1/2$, we could learn from the circuit which one is correct without error.

Intuitively speaking, we can think of the circuit's measurement outcome as being a guess for θ to "one bit of accuracy." In other words, if we were to write θ in binary notation and round it off to one bit, we'd have a number like this:

$$0.a = \begin{cases} 0 & a = 0 \\ \frac{1}{2} & a = 1. \end{cases}$$

The measurement outcome can be viewed as a guess for the bit a. When θ is neither 0 nor $1/2$, there's a nonzero probability that the guess will be wrong — but the probability of making an error becomes smaller and smaller as we get closer to 0 or $1/2$.

It's natural to ask what role the two Hadamard gates play in this procedure:

- The first Hadamard gate sets the control qubit to a uniform superposition of $|0\rangle$ and $|1\rangle$, so that when the phase kickback occurs, it happens for the $|1\rangle$ state and not the $|0\rangle$ state, creating a *relative* phase difference that affects the measurement outcomes. If we didn't do this and the phase kickback produced a *global* phase, it would have no effect on the probabilities of obtaining different measurement outcomes.
- The second Hadamard gate allows us to learn something about the number θ through the phenomenon of *interference*. Prior to the second Hadamard gate, the state of the top qubit is

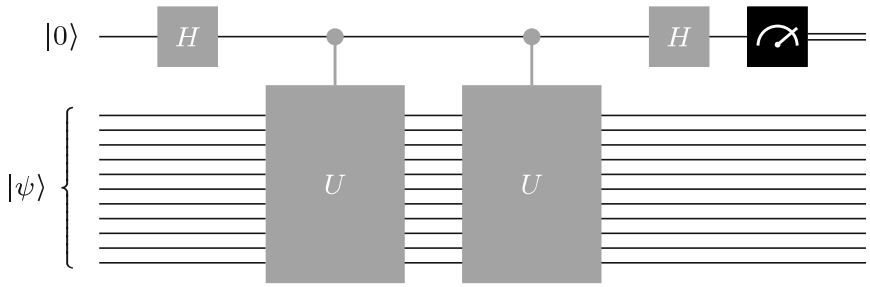
  $$\frac{1}{\sqrt{2}}|0\rangle + \frac{e^{2\pi i\theta}}{\sqrt{2}}|1\rangle,$$

  and if we were to measure this state, we would obtain 0 and 1 each with probability $1/2$, telling us nothing about θ. By performing the second Hadamard gate, however, we cause the number θ to affect the output probabilities.

Doubling the phase

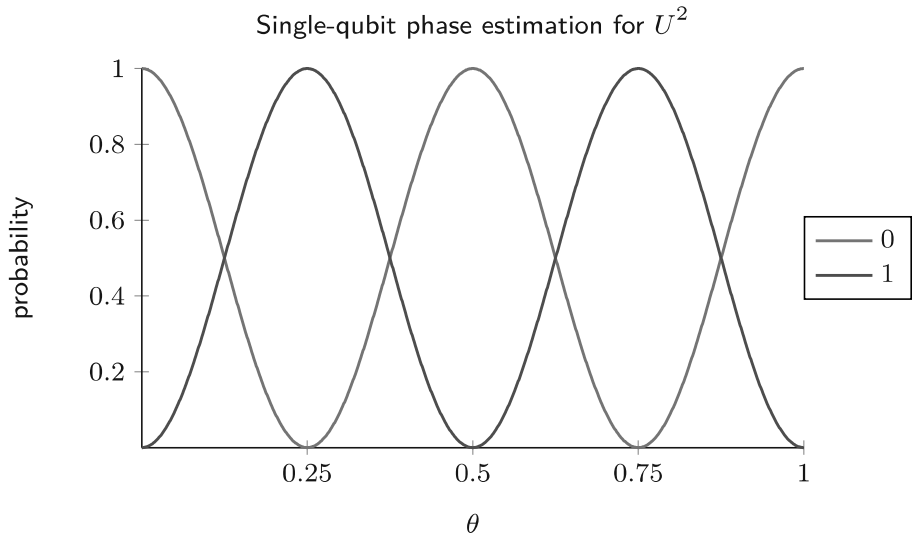
The circuit above uses the phase kickback phenomenon to approximate θ to a single bit of accuracy. One bit of accuracy may be all we need in some situations — but for factoring we're going to need a lot more accuracy than that. A natural question is, how can we learn more about θ?

One very simple thing we can do is to replace the controlled-U operation in our circuit with *two copies* of this operation, like in this circuit:



Two copies of a controlled-U operation is equivalent to a controlled-U^2 operation. If $|\psi\rangle$ is an eigenvector of U having eigenvalue $\lambda = e^{2\pi i\theta}$, then this state is also an eigenvector of U^2, this time having eigenvalue $\lambda^2 = e^{2\pi i(2\theta)}$.

So, if we run this version of the circuit, we're effectively performing the same computation as before, except that the number θ is replaced by 2θ. Here's a plot illustrating the output probabilities as θ ranges from 0 to 1.



Doing this can indeed provide us with some additional information about θ. If the binary representation of θ is

$$\theta = 0.a_1a_2a_3\cdots$$

then doubling θ effectively shifts the binary point one position to the right:

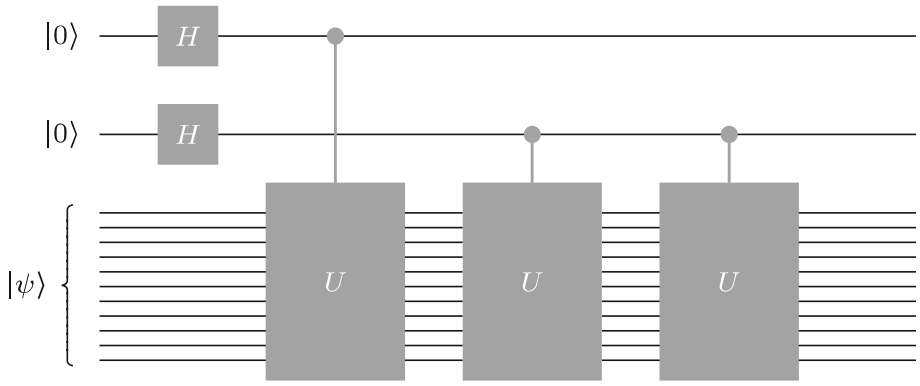
$$2\theta = a_1.a_2a_3\cdots$$

And because we're equating $\theta = 1$ with $\theta = 0$ as we move around the unit circle, we see that the bit a_1 has no influence on our probabilities, and we're effectively obtaining a guess for the *second* bit after the binary point if we round θ to two bits. For instance, if we knew in advance that θ was either 0 or $1/4$, then we could fully trust the measurement outcome to tell us which.

It's not immediately clear, though, how this estimation should be reconciled with what we learned from the original (non-doubled) phase kickback circuit to give us the most accurate information possible about θ. So let's take a step back and consider how to proceed.

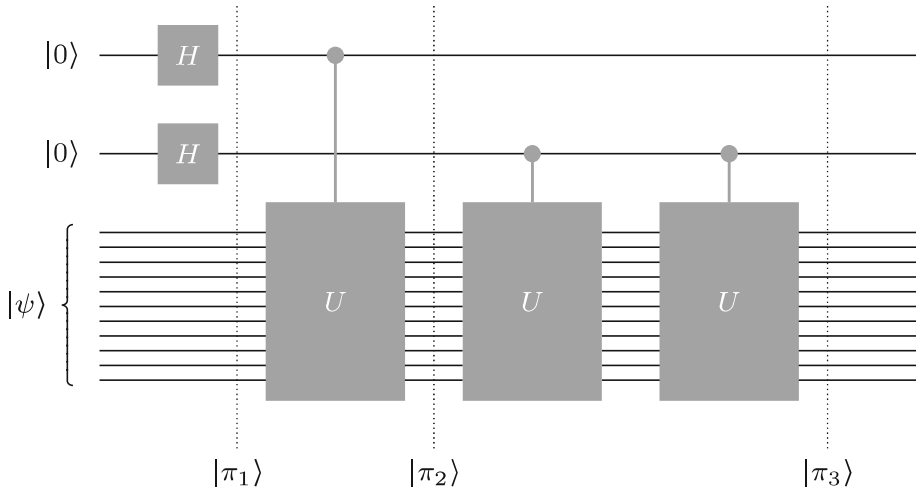
Two-qubit phase estimation

Rather than considering the two options described above separately, let's combine them into a single circuit like so.



The Hadamard gates after the controlled operations have been removed and there are no measurements here yet. We'll add more to the circuit as we consider our options for learning as much as we can about θ.

If we run this circuit when $|\psi\rangle$ is an eigenvector of U, the state of the bottom qubits will remain $|\psi\rangle$ throughout the entire circuit, and phases will be "kicked" into the state of the top two qubits. Let's analyze the circuit carefully, by means of the following figure.



We can write the state $|\pi_1\rangle$ like this:

$$|\pi_1\rangle = |\psi\rangle \otimes \frac{1}{2} \sum_{a_0=0}^{1} \sum_{a_1=0}^{1} |a_1 a_0\rangle.$$

When the first controlled-U operation is performed, the eigenvalue $\lambda = e^{2\pi i \theta}$ gets kicked into the phase when a_0 (the top qubit) is equal to 1, but not when it's 0. So, we can express the resulting state like this:

$$|\pi_2\rangle = |\psi\rangle \otimes \frac{1}{2} \sum_{a_0=0}^{1} \sum_{a_1=0}^{1} e^{2\pi i a_0 \theta} |a_1 a_0\rangle.$$

The second and third controlled-U gates do something similar, except for a_1 rather than a_0, and with θ replaced by 2θ. We can express the resulting state like this:

$$|\pi_3\rangle = |\psi\rangle \otimes \frac{1}{2} \sum_{a_0=0}^{1} \sum_{a_1=0}^{1} e^{2\pi i (2a_1 + a_0)\theta} |a_1 a_0\rangle.$$

If we think about the binary string $a_1 a_0$ as representing an integer $x \in \{0, 1, 2, 3\}$ in binary notation, which is $x = 2a_1 + a_0$, we can alternatively express this state as follows.

$$|\pi_3\rangle = |\psi\rangle \otimes \frac{1}{2} \sum_{x=0}^{3} e^{2\pi i x \theta} |x\rangle$$

Our goal is to extract as much information about θ as we can from this state.

At this point we'll consider a special case, where we're promised that $\theta = \frac{y}{4}$ for some integer $y \in \{0, 1, 2, 3\}$. In other words, we have $\theta \in \{0, 1/4, 1/2, 3/4\}$, so we can express this number exactly using binary notation with two bits, as $.00$, $.01$, $.10$, or $.11$. In general, θ might not be one of these four values, but thinking about this special case will help us to figure out how to most effectively extract information about θ in general.

First we'll define a two-qubit state vector for each possible value $y \in \{0, 1, 2, 3\}$.

$$|\phi_y\rangle = \frac{1}{2} \sum_{x=0}^{3} e^{2\pi i x (\frac{y}{4})} |x\rangle = \frac{1}{2} \sum_{x=0}^{3} e^{2\pi i \frac{xy}{4}} |x\rangle$$

After simplifying the exponentials, we can write these vectors as follows.

$$|\phi_0\rangle = \frac{1}{2}|0\rangle + \frac{1}{2}|1\rangle + \frac{1}{2}|2\rangle + \frac{1}{2}|3\rangle$$

$$|\phi_1\rangle = \frac{1}{2}|0\rangle + \frac{i}{2}|1\rangle - \frac{1}{2}|2\rangle - \frac{i}{2}|3\rangle$$

$$|\phi_2\rangle = \frac{1}{2}|0\rangle - \frac{1}{2}|1\rangle + \frac{1}{2}|2\rangle - \frac{1}{2}|3\rangle$$

$$|\phi_3\rangle = \frac{1}{2}|0\rangle - \frac{i}{2}|1\rangle - \frac{1}{2}|2\rangle + \frac{i}{2}|3\rangle$$

These vectors are orthogonal: if we choose any pair of them and compute their inner product, we get 0. Each one is also a unit vector, so

$\{|\phi_0\rangle, |\phi_1\rangle, |\phi_2\rangle, |\phi_3\rangle\}$ is an orthonormal basis. We therefore know right away that there is a measurement that can discriminate them perfectly — meaning that, if we're given one of them but we don't know which, then we can figure out which one it is without error.

To perform such a discrimination with a quantum circuit, we can first define a unitary operation V that transforms standard basis states into the four states listed above.

$$\begin{aligned} V|00\rangle &= |\phi_0\rangle \\ V|01\rangle &= |\phi_1\rangle \\ V|10\rangle &= |\phi_2\rangle \\ V|11\rangle &= |\phi_3\rangle \end{aligned}$$

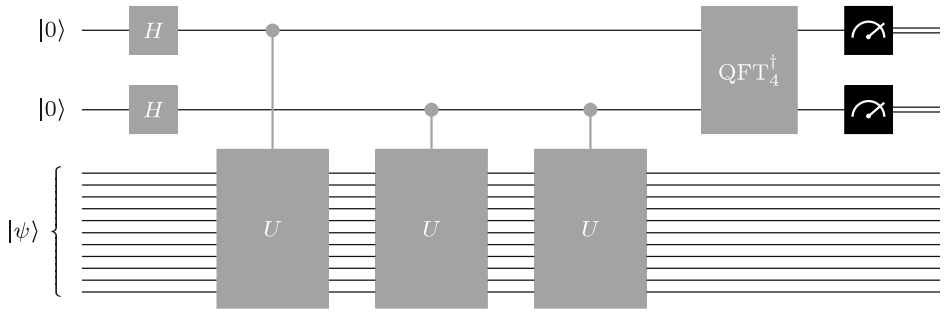
To write down V as a 4×4 matrix, it's just a matter of taking the columns of V to be the states $|\phi_0\rangle, \ldots, |\phi_3\rangle$.

$$V = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix}$$

This is a special matrix, and it's likely that some readers will have encountered it before: it's the matrix associated with the 4-dimensional *discrete Fourier transform*. In light of this fact, let us call it by the name QFT_4 rather than V. The name QFT is short for *quantum Fourier transform* — which is essentially just the discrete Fourier transform, viewed as a unitary operation. We'll discuss the quantum Fourier transform in greater detail and generality shortly.

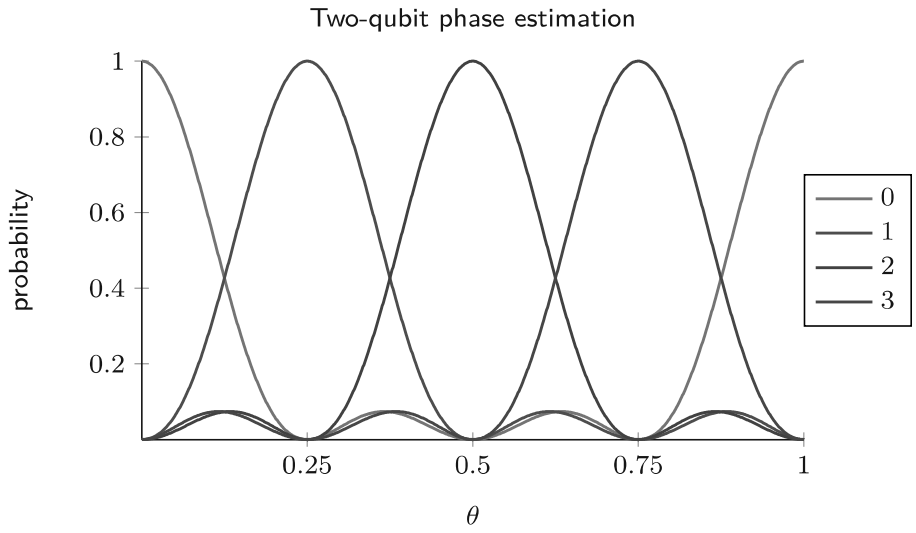
$$\mathrm{QFT}_4 = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix}$$

We can perform the inverse of this operation to go the other way, to transform the states $|\phi_0\rangle, \ldots, |\phi_3\rangle$ into the standard basis states $|0\rangle, \ldots, |3\rangle$. If we do this, then we can measure to learn which value $y \in \{0, 1, 2, 3\}$ describes θ as $\theta = y/4$. Here's a diagram of a quantum circuit that does this.



To summarize, if we run this circuit when $\theta = y/4$ for $y \in \{0, 1, 2, 3\}$, the state immediately before the measurements take place will be $|\psi\rangle|y\rangle$ (for y encoded as a two-bit binary string), so the measurements will reveal the value y without error.

This circuit is motivated by the special case that $\theta \in \{0, 1/4, 1/2, 3/4\}$ — but we can run it for any choice of U and $|\psi\rangle$, and hence any value of θ, that we wish. Here's a plot of the output probabilities the circuit produces for arbitrary choices of θ.



This is a clear improvement over the single-qubit variant described earlier in the lesson. It's not perfect — it can give us the wrong answer — but the answer is heavily skewed toward values of y for which $y/4$ is close to θ. In particular, the most likely outcome always corresponds to the closest value of $y/4$ to θ (equating $\theta = 0$ and $\theta = 1$ as before), and from the plot it looks like this closest value for x always appears with probability just above 40%. When θ is exactly halfway between two such values, like $\theta = 0.375$ for instance, the two equally close values of y are equally likely.

Preparing to generalize to many qubits

Given the improvement we've just obtained by using two control qubits rather than one, in conjunction with the inverse of the 4-dimensional quantum Fourier transform, it's natural to consider generalizing it further — by adding more control qubits. When we do this, we obtain the general *phase estimation procedure*. We'll see how this works shortly, but in order to describe it precisely we're going to need to discuss the quantum Fourier transform in greater generality, to see how it's defined for other dimensions and to see how we can implement it (or its inverse) with a quantum circuit.

Quantum Fourier transform

The quantum Fourier transform is a unitary operation that can be defined for any positive integer dimension N. In this section, we'll see how this operation is defined and how it can be implemented with a quantum circuit on m qubits with cost $O(m^2)$ when $N = 2^m$.

The matrices that describe the quantum Fourier transform are derived from an analogous operation on N-dimensional vectors known as the *discrete Fourier transform.* This operation can be thought about in different ways. For instance, we can think about the discrete Fourier transform in purely abstract, mathematical terms as a linear mapping. Or we can think about it in computational terms, where we're given an N-dimensional vector of complex numbers (using binary notation to encode the real and imaginary parts of the entries, let us suppose) and the goal is to calculate the N-dimensional vector obtained by applying the discrete Fourier transform. Our focus will be on third way, which is viewing this transformation as a unitary operation that can be performed on a quantum system.

There's an efficient algorithm for computing the discrete Fourier transform on a given input vector known as the *fast Fourier transform.* It has applications in signal processing and many other areas, and is considered by many to be one of the most important algorithms ever discovered. As it turns out, the implementation of the quantum Fourier transform when N is a power of 2 that we'll study is based on precisely the same underlying structure that make the fast Fourier transform possible.

Definition of the quantum Fourier transform

To define the quantum Fourier transform, we'll first define a complex number ω_N, for each positive integer N, like this:

$$\omega_N = e^{\frac{2\pi i}{N}} = \cos\left(\frac{2\pi}{N}\right) + i \sin\left(\frac{2\pi}{N}\right).$$

This is the number on the complex unit circle we obtain if we start at 1 and move counter-clockwise by an angle of $2\pi/N$ radians, or a fraction of $1/N$ of the circumference of the circle. Here are a few examples:

$$\omega_1 = 1$$

$$\omega_2 = -1$$

$$\omega_3 = -\frac{1}{2} + \frac{\sqrt{3}}{2} i$$

$$\omega_4 = i$$

$$\omega_8 = \frac{1+i}{\sqrt{2}}$$

$$\omega_{16} = \frac{\sqrt{2+\sqrt{2}}}{2} + \frac{\sqrt{2-\sqrt{2}}}{2} i$$

$$\omega_{100} \approx 0.998 + 0.063i$$

Now we can define the N-dimensional quantum Fourier transform, which is described by an $N \times N$ matrix whose rows and columns are associated with the standard basis states $|0\rangle, \ldots, |N-1\rangle$. We're only going to need this operation for when $N = 2^m$ is a power of 2 for phase estimation, but the operation can be defined for any positive integer N.

$$\mathrm{QFT}_N = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \omega_N^{xy} |x\rangle\langle y|$$

As was already stated, this is the matrix associated with the N-dimensional *discrete Fourier transform*. Often the leading factor of $1/\sqrt{N}$ is not included in the definition of this matrix, but we need to include it to obtain a unitary matrix.

Here's the quantum Fourier transform, written as a matrix, for some small values of N.

$$\mathrm{QFT}_1 = \begin{pmatrix} 1 \end{pmatrix}$$

$$\mathrm{QFT}_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$\mathrm{QFT}_3 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & \frac{-1+i\sqrt{3}}{2} & \frac{-1-i\sqrt{3}}{2} \\ 1 & \frac{-1-i\sqrt{3}}{2} & \frac{-1+i\sqrt{3}}{2} \end{pmatrix}$$

$$\mathrm{QFT}_4 = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix}$$

$$\mathrm{QFT}_8 = \frac{1}{2\sqrt{2}} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \frac{1+i}{\sqrt{2}} & i & \frac{-1+i}{\sqrt{2}} & -1 & \frac{-1-i}{\sqrt{2}} & -i & \frac{1-i}{\sqrt{2}} \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & \frac{-1+i}{\sqrt{2}} & -i & \frac{1+i}{\sqrt{2}} & -1 & \frac{1-i}{\sqrt{2}} & i & \frac{-1-i}{\sqrt{2}} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & \frac{-1-i}{\sqrt{2}} & i & \frac{1-i}{\sqrt{2}} & -1 & \frac{1+i}{\sqrt{2}} & -i & \frac{-1+i}{\sqrt{2}} \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & \frac{1-i}{\sqrt{2}} & -i & \frac{-1-i}{\sqrt{2}} & -1 & \frac{-1+i}{\sqrt{2}} & i & \frac{1+i}{\sqrt{2}} \end{pmatrix}$$

Notice, in particular, that QFT_2 is another name for a Hadamard operation.

Unitarity

Let's check that QFT_N is unitary, for any selection of N. One way to do this is to show that its columns form an orthonormal basis. We can define a vector corresponding to column number y, starting from $y = 0$ and going up to $y = N - 1$, like this:

$$|\phi_y\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \omega_N^{xy} |x\rangle.$$

Taking the inner product between any two of these vectors gives us this expression:

$$\langle \phi_z | \phi_y \rangle = \frac{1}{N} \sum_{x=0}^{N-1} \omega_N^{x(y-z)}$$

We can evaluate sums like this using the following formula for the sum of the first N terms of a geometric series.

$$1 + \alpha + \alpha^2 + \cdots + \alpha^{N-1} = \begin{cases} \frac{\alpha^N - 1}{\alpha - 1} & \text{if } \alpha \neq 1 \\ N & \text{if } \alpha = 1 \end{cases}$$

Specifically, we can use this formula when $\alpha = \omega_N^{y-z}$. When $y = z$, we have $\alpha = 1$, so using the formula and dividing by N gives

$$\langle \phi_y | \phi_y \rangle = 1.$$

When $y \neq z$, we have $\alpha \neq 1$, so the formula reveals this:

$$\langle \phi_z | \phi_y \rangle = \frac{1}{N} \frac{\omega_N^{N(y-z)} - 1}{\omega_N^{y-z} - 1} = \frac{1}{N} \frac{1 - 1}{\omega_N^{y-z} - 1} = 0.$$

This happens because $\omega_N^N = e^{2\pi i} = 1$, so $\omega_N^{N(y-z)} = 1^{y-z} = 1$, making numerator zero, while the denominator is nonzero because $\omega_N^{y-z} \neq 1$. Intuitively speaking, what we're doing is summing a bunch of points that are distributed around the unit circle, and they cancel out and leave 0 when summed.

We have therefore established that $\{|\phi_0\rangle, \ldots, |\phi_{N-1}\rangle\}$ is an orthonormal set,

$$\langle \phi_z | \phi_y \rangle = \begin{cases} 1 & y = z \\ 0 & y \neq z, \end{cases}$$

which reveals that QFT_N is unitary.

Controlled-phase gates

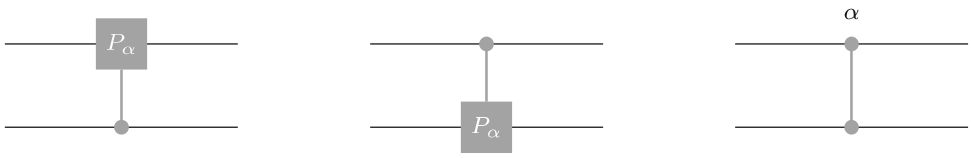
To implement the quantum Fourier transform with a quantum circuit, we'll need to make use of *controlled-phase* gates. Recall that a *phase operation* is a single-qubit unitary operation of the form

$$P_\alpha = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix}$$

for any real number α. A controlled version of this gate has the following matrix:

$$CP_\alpha = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\alpha} \end{pmatrix}$$

For this controlled gate, it doesn't actually matter which qubit is the control and which is the target because the two possibilities are equivalent. We can use any of the following symbols to represent this gate in quantum circuit diagrams.

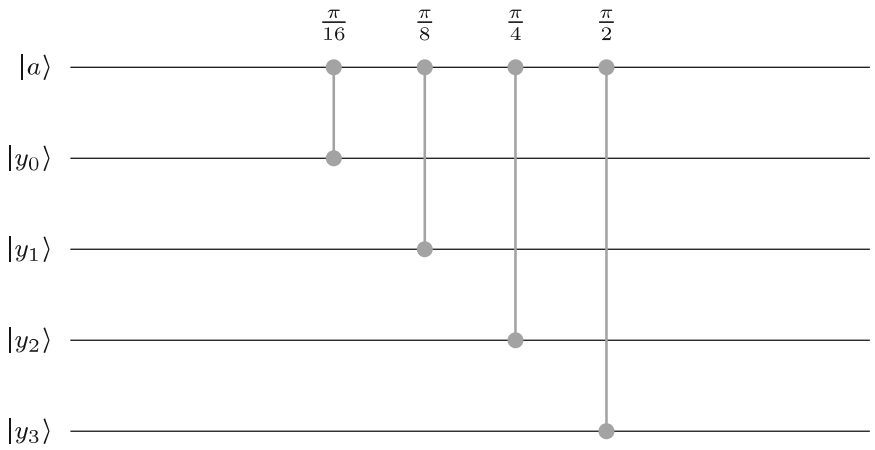


For the third form, the label α is also sometimes placed on the side of the control line or under the lower control when that's convenient.

To perform the quantum Fourier transform when $N = 2^m$ and $m \geq 2$, we're going to need to perform an operation on m qubits whose action on standard basis states can be described as

$$|y\rangle|a\rangle \mapsto \omega_{2^m}^{ay}|y\rangle|a\rangle,$$

where a is a bit and $y \in \{0, \ldots, 2^{m-1} - 1\}$ is a number encoded in binary notation as a string of $m - 1$ bits. This can be done using controlled-phase gates by generalizing the following example, for which $m = 5$.



In general, for an arbitrary choice of $m \geq 2$, the top qubit corresponding to the bit a can be viewed as the control, with the phase gates P_α ranging from $\alpha = \pi/2^{m-1}$ on the qubit corresponding to the least significant bit of y to $\alpha = \frac{\pi}{2}$ on the qubit corresponding to the most

significant bit of y. These controlled-phase gates all commute with one another and could be performed in any order.

Circuit implementation of the QFT

Now we'll see how we can implement the quantum Fourier transform with a circuit when the dimension $N = 2^m$ is a power of 2. There are, in fact, multiple ways to implement the quantum Fourier transform, but this is arguably the simplest method known. Once we know how to implement the quantum Fourier transform with a quantum circuit, it's straightforward to implement its inverse: we can replace each gate with its inverse (or, equivalently, conjugate transpose) and apply the gates in the reverse order. Every quantum circuit composed of unitary gates alone can be inverted in this way.

The implementation is recursive in nature, so that's how it's most naturally described. The base case is $m = 1$, in which case the quantum Fourier transform is a Hadamard operation.

To perform the quantum Fourier transform on m qubits when $m \geq 2$, we can perform the following steps, whose actions we'll describe for standard basis states of the form $|x\rangle|a\rangle$, where $x \in \{0, \ldots, 2^{m-1} - 1\}$ is an integer encoded as $m - 1$ bits using binary notation and a is a single bit.

1. First apply the 2^{m-1}-dimensional quantum Fourier transform to the bottom/leftmost $m - 1$ qubits to obtain this state:

   $$\Big(\mathrm{QFT}_{2^{m-1}}|x\rangle\Big)|a\rangle = \frac{1}{\sqrt{2^{m-1}}} \sum_{y=0}^{2^{m-1}-1} \omega_{2^{m-1}}^{xy} |y\rangle|a\rangle.$$

   This is done by recursively applying the method being described for one fewer qubit, using the Hadamard operation on a single qubit as the base case.

2. Use the top/rightmost qubit as a control to inject the phase $\omega_{2^m}^y$ for each standard basis state $|y\rangle$ of the remaining $m - 1$ qubits (as is described above) to obtain this state:

   $$\frac{1}{\sqrt{2^{m-1}}} \sum_{y=0}^{2^{m-1}-1} \omega_{2^{m-1}}^{xy} \omega_{2^m}^{ay} |y\rangle|a\rangle.$$

3. Perform a Hadamard gate on the top/rightmost qubit to obtain this state:

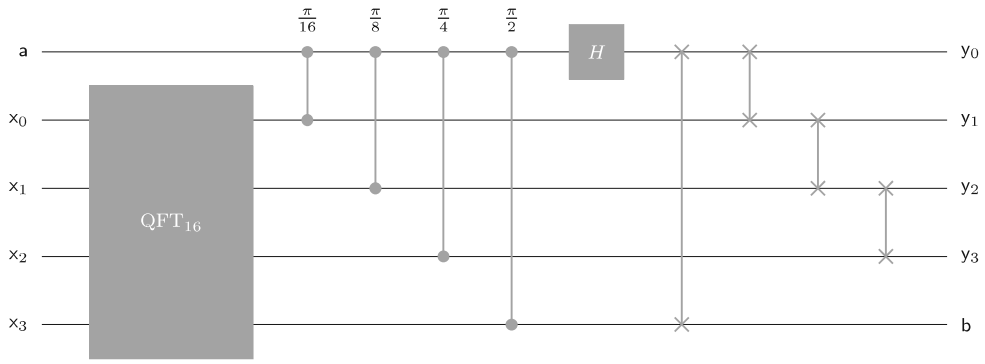
$$\frac{1}{\sqrt{2^m}} \sum_{y=0}^{2^{m-1}-1} \sum_{b=0}^{1} (-1)^{ab} \omega_{2^{m-1}}^{xy} \omega_{2^m}^{ay} |y\rangle |b\rangle.$$

4. Permute the order of the qubits so that the least significant bit becomes the most significant bit, with all others shifted up/right:

$$\frac{1}{\sqrt{2^m}} \sum_{y=0}^{2^{m-1}-1} \sum_{b=0}^{1} (-1)^{ab} \omega_{2^{m-1}}^{xy} \omega_{2^m}^{ay} |b\rangle |y\rangle.$$

For example, here's the circuit we obtain for $N = 32 = 2^5$. In this diagram, the qubits are given names that correspond to the standard basis vectors $|x\rangle|a\rangle$ (for the input) and $|b\rangle|y\rangle$ (for the output) for clarity.

32-dimensional quantum Fourier transform



Analysis

The key formula we need to verify that the circuit just described implements the 2^m-dimensional quantum Fourier transform is this one:

$$(-1)^{ab} \omega_{2^{m-1}}^{xy} \omega_{2^m}^{ay} = \omega_{2^m}^{(2x+a)(2^{m-1}b+y)}.$$

This formula works for any choice of integers a, b, x, and y, but we'll only need it for $a, b \in \{0, 1\}$ and $x, y \in \{0, \ldots, 2^{m-1} - 1\}$. It can be checked by expanding the product in the exponent on the right-hand side,

$$\omega_{2^m}^{(2x+a)(2^{m-1}b+y)} = \omega_{2^m}^{2^m xb} \omega_{2^m}^{2xy} \omega_{2^m}^{2^{m-1}ab} \omega_{2^m}^{ay} = (-1)^{ab} \omega_{2^{m-1}}^{xy} \omega_{2^m}^{ay},$$

where the second equality makes use of the observation that

$$\omega_{2^m}^{2^m xb} = \left(\omega_{2^m}^{2^m}\right)^{xb} = 1^{xb} = 1.$$

The 2^m-dimensional quantum Fourier transform is defined as follows for every $u \in \{0, \ldots, 2^m - 1\}$.

$$\mathrm{QFT}_{2^m} |u\rangle = \frac{1}{\sqrt{2^m}} \sum_{v=0}^{2^m - 1} \omega_{2^m}^{uv} |v\rangle$$

If we write u and v as

$$\begin{aligned} u &= 2x + a \\ v &= 2^{m-1} b + y \end{aligned}$$

for $a, b \in \{0, 1\}$ and $x, y \in \{0, \ldots, 2^{m-1} - 1\}$, we obtain

$$\begin{aligned} \mathrm{QFT}_{2^m} |2x + a\rangle &= \frac{1}{\sqrt{2^m}} \sum_{y=0}^{2^{m-1}-1} \sum_{b=0}^{1} \omega_{2^m}^{(2x+a)(2^{m-1}b + y)} |b 2^{m-1} + y\rangle \\ &= \frac{1}{\sqrt{2^m}} \sum_{y=0}^{2^{m-1}-1} \sum_{b=0}^{1} (-1)^{ab} \omega_{2^{m-1}}^{xy} \omega_{2^m}^{ay} |b 2^{m-1} + y\rangle. \end{aligned}$$

Finally, by thinking about the standard basis states $|x\rangle|a\rangle$ and $|b\rangle|y\rangle$ as binary encodings of integers in the range $\{0, \ldots, 2^m - 1\}$,

$$\begin{aligned} |x\rangle|a\rangle &= |2x + a\rangle \\ |b\rangle|y\rangle &= |2^{m-1} b + y\rangle, \end{aligned}$$

we see that the circuit above implements the required operation. If this method for performing the quantum Fourier transform seems remarkable, it's because it is: it's essentially the fast Fourier transform in the form of a quantum circuit.

Finally, let's count how many gates are used in the circuit just described. The controlled-phase gates aren't in the standard gate set that we discussed in the previous lesson, but to begin we'll ignore this and count each of them as a single gate.

Let's let s_m denote the number of gates we need for each possible choice of m. If $m = 1$, the quantum Fourier transform is just a Hadamard operation, so

$$s_1 = 1.$$

If $m \geq 2$, then in the circuit above we need s_{m-1} gates for the quantum Fourier transform on $m - 1$ qubits, plus $m - 1$ controlled-phase gates, plus a Hadamard gate, plus $m - 1$ swap gates, so

$$s_m = s_{m-1} + (2m - 1).$$

We can obtain a closed-form expression by summing:

$$s_m = \sum_{k=1}^{m} (2k - 1) = m^2.$$

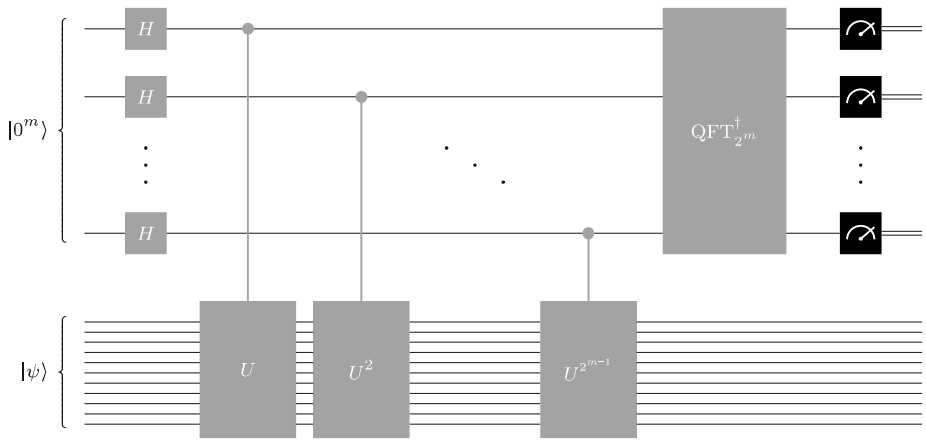
We don't actually need as many swap gates as the method describes. If we rearrange the gates just a bit, we can push all of the swap gates out to the right and reduce the number of swap gates required to $\lfloor m/2 \rfloor$. Asymptotically speaking this isn't a major improvement: we still obtain circuits with size $O(m^2)$ for performing QFT_{2^m}.

If we wish to implement the quantum Fourier transform using only gates from our standard gate set, we need to either build or approximate each of the controlled-phase gates with gates from our set. The number required depends on how much accuracy we require, but as a function of m the total cost remains quadratic.

It is, in fact, possible to approximate the quantum Fourier transform quite closely with a sub-quadratic number of gates by using the fact that P_α is very close to the identity operation when α is very small — which means that we can simply leave out most of the controlled-phase gates without suffering too much of a loss in terms of accuracy.

General procedure and analysis

Now we'll examine the phase-estimation procedure in general. The idea is to extend the two-qubit version of phase estimation that we considered above in the natural way suggested by the following diagram.



Notice that, for each new control qubit added on the top, we *double* the number of times the unitary operation U is performed. This is indicated in the diagram by the powers on U for each of the controlled-unitary operations.

The most straightforward way to implement a controlled-U^k operation for some choice of k is simply to repeat a controlled-U operation k times. If this is indeed the methodology that is used, it must be recognized that the addition of control qubits contributes significantly to the size of the circuit: if we have m control qubits, like the diagram depicts, a total of $2^m - 1$ copies of the controlled-U operation are required. This means that a significant computational cost is incurred as m is increased — but as we will see, it also leads to a significantly more accurate approximation of θ.

It is important to note, however, that for *some* choices of U it may be possible to create a circuit that implements the operation U^k for large values of k in a more efficient way than simply repeating k times the circuit for U. We'll see a specific example of this in the context of integer factorization later in the lesson, where the efficient algorithm for *modular exponentiation* discussed in the previous lesson comes to the rescue.

Now let us analyze the circuit just described. The state immediately prior to the inverse quantum Fourier transform looks like this:

$$\frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} \bigl(U^x|\psi\rangle\bigr)|x\rangle = |\psi\rangle \otimes \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} e^{2\pi i x\theta}|x\rangle.$$

A special case

Along similar lines to what we did in the $m = 2$ case, we'll first consider the special case that $\theta = y/2^m$ for $y \in \{0, \ldots, 2^m - 1\}$. In this case the state prior to the inverse quantum Fourier transform can alternatively be written like this:

$$|\psi\rangle \otimes \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} e^{2\pi i \frac{xy}{2^m}}|x\rangle = |\psi\rangle \otimes \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} \omega_{2^m}^{xy}|x\rangle = |\psi\rangle \otimes \mathrm{QFT}$$

So, when the inverse quantum Fourier transform is applied, the state becomes

$$|\psi\rangle|y\rangle$$

and the measurements reveal y (encoded in binary).

Bounding the probabilities

For other values of θ, meaning ones that don't take the form $y/2^m$ for an integer y, the measurement outcomes won't be certain, but we can prove bounds on the probabilities for different outcomes. Going forward, let's consider an arbitrary choice of θ satisfying $0 \leq \theta < 1$.

After the inverse quantum Fourier transform is performed, the state of the circuit is this:

$$|\psi\rangle \otimes \frac{1}{2^m} \sum_{y=0}^{2^m-1} \sum_{x=0}^{2^m-1} e^{2\pi i x(\theta - y/2^m)} |y\rangle.$$

So, when the measurements on the top m qubits are performed, we see each outcome y with probability

$$p_y = \left| \frac{1}{2^m} \sum_{x=0}^{2^m-1} e^{2\pi i x(\theta - y/2^m)} \right|^2.$$

To get a better handle on these probabilities, we'll make use of the same formula that we saw before, for the sum of the initial portion of a geometric series.

$$1 + \alpha + \alpha^2 + \cdots + \alpha^{N-1} = \begin{cases} \frac{\alpha^N - 1}{\alpha - 1} & \text{if } \alpha \neq 1 \\ N & \text{if } \alpha = 1 \end{cases}$$

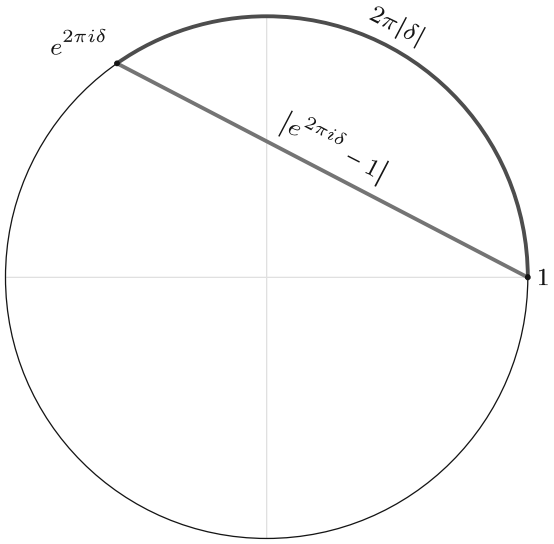
We can simplify the sum appearing in the formula for p_y by taking $\alpha = e^{2\pi i(\theta - y/2^m)}$. Here's what we obtain.

$$\sum_{x=0}^{2^m-1} e^{2\pi i x(\theta - y/2^m)} = \begin{cases} 2^m & \theta = y/2^m \\ \frac{e^{2\pi i(2^m\theta - y)} - 1}{e^{2\pi i(\theta - y/2^m)} - 1} & \theta \neq y/2^m \end{cases}$$

So, in the case that $\theta = y/2^m$, we find that $p_y = 1$ (as we already knew from considering this special case), and in the case that $\theta \neq y/2^m$, we find that

$$p_y = \frac{1}{2^{2m}} \left| \frac{e^{2\pi i(2^m\theta - y)} - 1}{e^{2\pi i(\theta - y/2^m)} - 1} \right|^2.$$

We can learn more about these probabilities by thinking about how arc lengths and chord lengths on the unit circle are related. Here's a figure that illustrates the relationships we need for any real number $\delta \in \left[-\frac{1}{2}, \frac{1}{2}\right]$.



First, the chord length (drawn in blue) can't possibly be larger than the arc length (drawn in purple):

$$\left|e^{2\pi i\delta} - 1\right| \leq 2\pi|\delta|.$$

Relating these lengths in the other direction, we see that the ratio of the arc length to the chord length is greatest when $\delta = \pm 1/2$, and in this case the ratio is half the circumference of the circle divided by the diameter, which is $\pi/2$. Thus, we have

$$\frac{2\pi|\delta|}{\left|e^{2\pi i\delta} - 1\right|} \leq \frac{\pi}{2},$$

and so

$$\left|e^{2\pi i\delta} - 1\right| \geq 4|\delta|.$$

An analysis based on these relations reveals the following two facts.

1. Suppose that θ is a real number and $y \in \{0, \ldots, 2^m - 1\}$ satisfies

$$\left|\theta - \frac{y}{2^m}\right| \leq 2^{-(m+1)}.$$

   This means that $y/2^m$ is either the best m-bit approximation to θ, or it's exactly halfway between $y/2^m$ and either $(y-1)/2^m$ or $(y+1)/2^m$, so it's one of the two best approximations to θ.

We'll prove that p_y has to be pretty large in this case. By the assumption we're considering, it follows that $|2^m\theta - y| \leq 1/2$, so we can use the second observation above relating arc and chord lengths to conclude that

$$\left|e^{2\pi i(2^m\theta - y)} - 1\right| \geq 4|2^m\theta - y| = 4 \cdot 2^m \cdot \left|\theta - \frac{y}{2^m}\right|.$$

We can also use the first observation about arc and chord lengths to conclude that

$$\left|e^{2\pi i(\theta - y/2^m)} - 1\right| \leq 2\pi\left|\theta - \frac{y}{2^m}\right|.$$

Putting these two inequalities to use on p_y reveals

$$p_y \geq \frac{1}{2^{2m}}\frac{16 \cdot 2^{2m}}{4\pi^2} = \frac{4}{\pi^2} \approx 0.405.$$

This explains our observation that the best outcome occurs with probability greater than 40% in the $m = 2$ version of phase estimation discussed earlier. It's not really 40%, it's $4/\pi^2$, and in fact this bound holds for every choice of m.

2. Now suppose that $y \in \{0, \ldots, 2^m - 1\}$ satisfies

$$2^{-m} \leq \left|\theta - \frac{y}{2^m}\right| \leq \frac{1}{2}.$$

This means that there's a better approximation $z/2^m$ to θ in between θ and $y/2^m$.

This time we'll prove that p_y can't be too big. We can start with the simple observation that

$$\left|e^{2\pi i(2^m\theta - y)} - 1\right| \leq 2,$$

which follows from the fact that any two points on the unit circle can differ in absolute value by at most 2.

We can also use the second observation about arc and chord lengths from above, this time working with the denominator of p_y rather than the numerator, to conclude

$$\left|e^{2\pi i(\theta - y/2^m)} - 1\right| \geq 4\left|\theta - \frac{y}{2^m}\right| \geq 4 \cdot 2^{-m}.$$

Putting the two inequalities together reveals

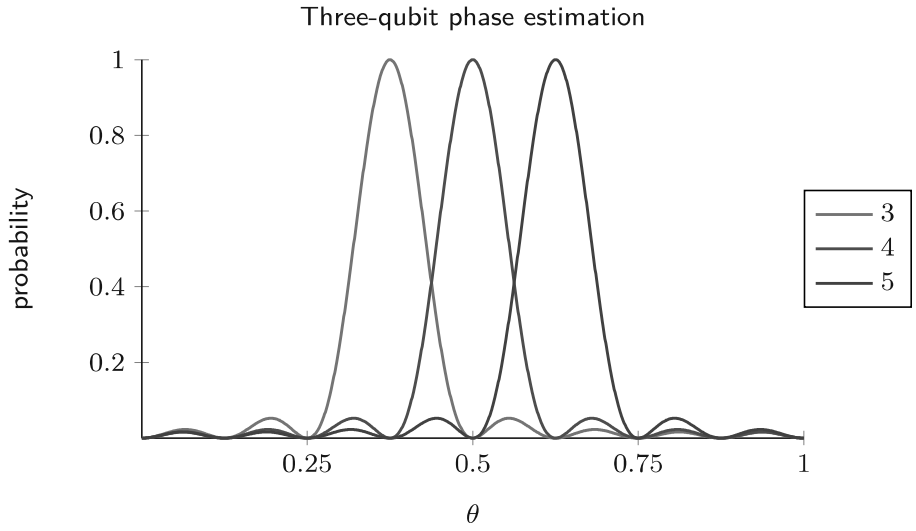
$$p_y \leq \frac{1}{2^{2m}}\frac{4}{16 \cdot 2^{-2m}} = \frac{1}{4}.$$

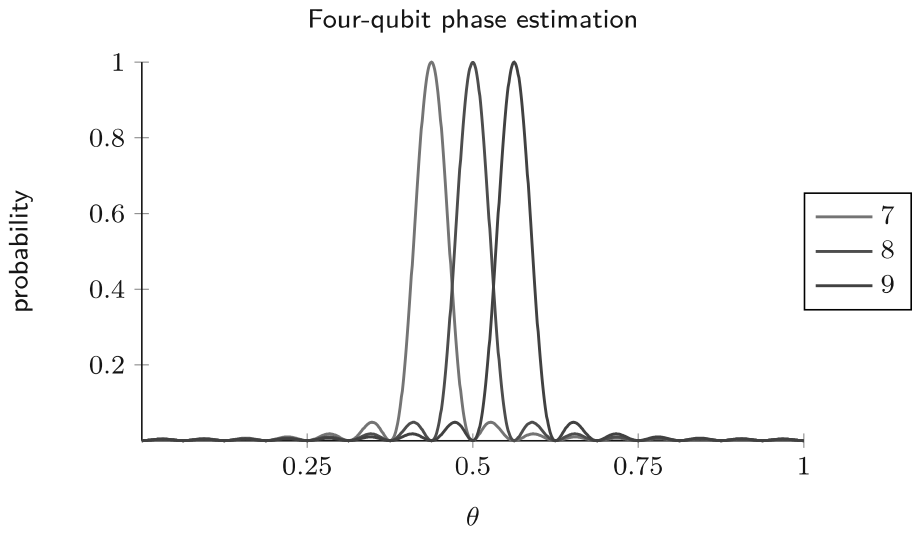
Note that, while this bound is good enough for our purposes, it is fairly crude — the probability is usually much lower than $1/4$.

The important take-away from this analysis is that very close approximations to θ are likely to occur — we'll get a best m-bit approximation with probability greater than 40% — whereas approximations off by more than 2^{-m} are less likely to occur, with probability upper bounded by 25%.

Given these guarantees, it is possible to boost our confidence by repeating the phase estimation procedure several times, to gather statistical evidence about θ. It is important to note that the state $|\psi\rangle$ of the bottom collection of qubits is unchanged by the phase estimation procedure, so it can be used to run the procedure as many times as we like. In particular, each time we run the circuit, we get a best m-bit approximation to θ with probability greater than 40%, while the probability of being off by more than 2^{-m} is bounded by 25%. If we run the circuit several times and take the most commonly appearing outcome of the runs, it's therefore exceedingly likely that the outcome that appears most commonly will not be one that occurs at most 25% of the time. As a result, we'll be very likely to obtain an approximation $y/2^m$ that's within $1/2^m$ of the value θ. Indeed, the unlikely chance that we're off by more than $1/2^m$ decreases exponentially in the number of times the procedure is run.

Here are two plots showing the probabilities for three consecutive values for y when $m = 3$ and $m = 4$ as functions of θ. (Only three outcomes are shown for clarity. Probabilities for other outcomes are obtained by cyclically shifting the same underlying function.)





Was this page helpful?

Yes No

Report a bug, typo, or request content on GitHub ↗.

Previous page Next page