QuIC Seminar 13

Shor's Algorithm Part 2: The Quantum Fourier Transform

Contents

In the first seminar on Shor's algorithm, we explained how the problem of factoring $n = pq$ (where p, q are prime) can be reduced to finding the period of the modular exponential function

$$f_x(r) := x^r \mod n \tag{13.1}$$

where $f : \mathbb{Z}_+ \to \mathbb{Z}_n$ and $x \in \mathbb{Z}_n$. (We also spent some time on cryptography, but this was mainly to motivate why factoring is an interesting problem.)

In this seminar, we'll explain a key subroutine in Shor's algorithm—the Quantum Fourier Transform (QFT). The QFT is used in many other algorithms—we've already used it in quantum phase estimation—and is one of the most important subroutines in all of quantum computing. Mastering the QFT is critical.

In a more general sense than quantum computing, the QFT is interesting because it's, well, a Fourier transform—potentially the most useful transformation in applied mathematics. It's uses range from differential equations to signal processing to harmonic analysis and beyond. After my own brief foray into quantum field theory (another QFT), I'm also willing to say the Fourier transform is essentially everything there.

The history of algorithms for Fourier transforms is highly interesting as well, with the Fast Fourier Transform (FFT) providing a significant speedup over the "classic" FT algorithm. We'll see how the QFT takes the cake in this seminar—albeit under a crucially important caveat.

Question: Why would something like a Fourier transform be needed in Shor's algorithm? What's the intuition?

Answer: Shor's algorithm for factoring is all about period-finding, which is what Fourier transforms are all about! A FT maps from the time domain to the frequency (Fourier) domain. Frequency is *exactly* what we want to know about this modular exponential function guy (13.1).

13.1 Fourier transforms

Let's start with the definition of a classical Fourier transform, first in the continuous case and then in the discrete case. This will motivate the definition of the QFT and help us understand what's going on.

Definition 13.1 (Continuous Fourier transform). Let $f(t)$ be a real-valued function of "time" t[a]. Then, the **Fourier transform** of f is a function of "frequency" ω defined by

$$\mathcal{F}\{f(t)\}(\omega) \equiv \hat{f}(\omega) := \int_{-\infty}^{\infty} f(t)e^{-2\pi i\omega t}\,dt \tag{13.2}$$

The FT of a function f is defined whenever this integral converges for all ω. The **inverse Fourier transform** is defined by

$$f(t) \equiv \mathcal{F}^{-1}\{\hat{f}(\omega)\}(t) := \frac{1}{2\pi}\int_{-\infty}^{\infty} \hat{f}(\omega)e^{2\pi i\omega t}\,d\omega \tag{13.3}$$

[a]The function can also be complex-valued, but intuition is better in the real case with time and frequency, so we'll stick to this without any issues.

⚠ Caution! Almost everyone has their own unique definition of the Fourier transform. They're all essentially the same, but it varies where the constants go and the scaling in the exponential and even which is the forward/reverse transform. The standard definition of the QFT, which we'll shortly see, actually corresponds to the inverse FT[1]!

Exercise 92: Verify that $\mathcal{F}^{-1}\{\mathcal{F}\{f(t)\}\} = f(t)$ for all $f(t)$.

There's a bunch of examples you usually do with the FT, such as computing it for sine/cosine and getting a Dirac-delta function, showing the FT of a Gaussian is a Gaussion, etc. We won't be concerned with these examples here, but rather the discrete version of this continuous transformation.

The discrete Fourier transform (DFT) arises when f is not a continuous "signal," but rather a discrete one. Here, we can represent f by a vector of N discrete values where it is defined. In particular, let

$$\boldsymbol{f} := (f_0, f_1, ..., f_{N-1}). \tag{13.4}$$

The DFT can then be obtained from (13.2) by using the trapezoidal rule on the integral. The result, which we'll just define below, is the DFT. Here, we'll say the values f_i are complex to anticipate the definition of the QFT.

Definition 13.2 (Discrete Fourier transform). Let $\boldsymbol{f} \in \mathbb{C}^N$. The **discrete Fourier transform** of $\boldsymbol{f}$ is a vector $\hat{\boldsymbol{f}} \in \mathbb{C}_N$ whose kth element is defined by

$$\hat{f}_k := \frac{1}{\sqrt{N}}\sum_{j=0}^{N-1} \omega^{kj} f_j \tag{13.5}$$

where

$$\omega := e^{2\pi i/N} \tag{13.6}$$

is the Nth root of unity.

The entire action of the FT on the vector $\boldsymbol{f}$ can be described by a matrix transformation. (Equation (13.5) above is really the equation for the kth component in a matrix-vector multiplication.) We can see from (13.5) that this matrix is

$$\Omega := \frac{1}{\sqrt{N}}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^N \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2N} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \omega^N & \omega^{2N} & \cdots & \omega^{N^2} \end{bmatrix} \tag{13.7}$$

[1]I went to a talk by Peter Shor at Los Alamos National Lab where he said something like, "I got rid of the minus sign (in the forward QFT definition) because it was easier to think about without the minus sign. This is confusing now, and I learned from this that you should always follow the standard convention!" I don't think this was recorded anywhere, but if I ever find it I'll link it here. This was the gist of Peter's point, though.

You can convince yourself this is true by writing out the action of the DFT matrix Ω on the vector $\boldsymbol{f}$ and examining the kth component. In fact, this is a good exercise:

Exercise 93: Perform the matrix-vector multiplication $\Omega\boldsymbol{f}$ and inspect the kth component. Verify this is equal to the kth component in (13.5).

Exercise 94: Write out the DFT matrix (13.7) for $N = 2$. Does this look like a quantum gate?

Exercise 95: Write out the DFT matrix (13.7) for $N = 4$.

As you can see, this matrix has a lot of structure. This structure is exploited in deriving the quantum Fourier transform which implements this matrix, which we'll see below.

Hold the phone! Re-read that last sentence! If the quantum Fourier transform is going to implement this matrix, we better verify that it's unitary!

Theorem 13.1. The DFT matrix Ω in (13.7) is unitary.

Proof. There's multiple ways to show this. One way is to prove that the columns of Ω are orthonormal.

Let Ω_j and Ω_k denote the jth and kth columns of Ω, respectively. Then,

$$\Omega_j = \frac{1}{\sqrt{N}} \left[1, \omega^j, \omega^{2j}, ..., \omega^{Nj}\right]^T$$

$$\Omega_k = \frac{1}{\sqrt{N}} \left[1, \omega^k, \omega^{2k}, ..., \omega^{Nk}\right]^T .$$

The inner product is thus

$$\Omega_j^\dagger \Omega_k = \frac{1}{N} \sum_{i=0}^{N-1} \omega^{i(k-j)}. \tag{13.8}$$

If $k = j$, we see immediately that $\Omega_k^\dagger \Omega_k = 1$, which proves the normalization.

Suppose now $k \neq j$. We can use the geometric series formula to write

$$\Omega_j^\dagger \Omega_k = \frac{1 - \omega^{N(k-j)}}{1 - \omega^{k-j}} \tag{13.9}$$

Since $k \neq j$, we have $\omega^{N(k-j)} = e^{2\pi i(k-j)} = 1$. (Recall the definition of ω and Euler's identity $e^{2\pi i} = 1$.)

Another way to prove this is to construct the inverse

$$\Omega^\dagger = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \bar{\omega} & \bar{\omega}^2 & \cdots & \bar{\omega}^N \\ 1 & \bar{\omega}^2 & \bar{\omega}^4 & \cdots & \bar{\omega}^{2N} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \bar{\omega}^N & \bar{\omega}^{2N} & \cdots & \bar{\omega}^{N^2} \end{bmatrix} \tag{13.10}$$

where $\bar{\omega} = e^{-2\pi i/N}$ is the complex conjugate of ω.

□

Exercise 96: Verify that $\Omega^\dagger \Omega = I_N$ and $\Omega\Omega^\dagger = I_N$.

13.1.1 Asymptotic scalings

We're talking about algorithms, after all, so we care about how fast these transformations can be implemented. We're going to do them on a computer, where everything is discrete, so we're concerned with the DFT. The first guess is that the runtime of the DFT goes by N^2, since this is the number of elements in the matrix Ω. (And, roughly the number of floating point operations (FLOPs) needed to do the matrix-vector multiplication.) This is the runtime of the "basic" DFT algorithm.

There's a faster DFT algorithm, known as the *fast Fourier transform* (FFT). The FFT algorithm exploits the structure of the matrix Ω to get the scaling down to $O(N \log N)$, a huge improvement! The details of how this works are quite similar to how the QFT works. In the future, I may include a description of the FFT algorithm, but for now we'll see how the matrix factorization works in the QFT.

Impressed by the FFT? Wait until you see the runtime of the QFT:

$$O(\log(N)^2) \qquad \text{(scaling of the QFT)} \tag{13.11}$$

An exponential improvement! We'll prove this asymptotic scaling when we build up the QFT in the next section, showing that the number of gates needed is $O(n^2)$ where $n = \log(N)$ is the number of qubits. For a summary, we present the scalings of each algorithm discussed in this section in the table below.

Algorithm	**Scaling**
DFT	$O(N \cdot N)$
FFT	$O(N \cdot \log(N))$
QFT	$O(\log(N) \cdot \log(N))$

Table 13.1: Asymptotic scalings of three algorithms for the discrete Fourier transform, written stylistically to see the improvements. The *approximate quantum Fourier transform* (AQFT) achieves $O(\log(N) \log\log(N))$ scaling. We can cover this in a future seminar.

13.2 The quantum Fourier Transform

All of the preface for the quantum Fourier transform (QFT) was built up in the previous section. The definition of the QFT is identical to the DFT, but stated in slightly different notation. We include these definitions for completeness, but remark again the QFT is exactly the same transformation of the DFT.

Definition 13.3 (Quantum Fourier transformation). The **Quantum Fourier transformation** is a basis transformation described by the following unitary evolution. Let $N = 2^n$ and $|k\rangle$, $|j\rangle$ denote computational basis vectors. Then,

$$|k\rangle \xmapsto{\text{QFT}} \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \omega^{jk} |j\rangle \tag{13.12}$$

where

$$\omega := e^{2\pi i/2^n} \tag{13.13}$$

is the Nth root of unity. We may write QFT_N to denote the dimension.

The action of the QFT on an arbitrary quantum state is obtained by using linearity. The result is stated below, and the proof is left as an exercise.

Theorem 13.2 (Quantum Fourier transformation on an arbitrary state). Let

$$|\psi\rangle = \sum_k \alpha_k |k\rangle \tag{13.14}$$

be a quantum state. Then,

$$\text{QFT}|\psi\rangle = \sum_k \beta_k |k\rangle \tag{13.15}$$

where the coefficients are given by

$$\beta_k := \frac{1}{\sqrt{2^n}} \sum_j \omega^{jk} \alpha_k. \tag{13.16}$$

Exercise 97: Proof Theorem 13.2 by using Definition 13.3 and linearity.

If you were doing all the exercises, you'd agree that

$$\text{QFT}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} =: H \tag{13.17}$$

where H is the fan favorite Hadamard gate. Additionally, you would agree that

$$\text{QFT}_4 = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix}, \tag{13.18}$$

although maybe you didn't color the matrix elements this way. (If you didn't do these exercises, do them now to convince yourself of these facts.)

13.2.1 Seeing structure in a small example

Why color the matrix elements this way? Watch what happens when we *swap* the inner columns of the QFT_4 matrix, keeping the colors the same for clarity:

$$\widetilde{\text{QFT}}_4 = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & i & -i \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -i & i \end{bmatrix}, \tag{13.19}$$

Do you see any structure here? Any block-diagonal structure? Let's color the matrix elements in blocks for clarity:

$$\widetilde{\text{QFT}}_4 = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & i & -i \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -i & i \end{bmatrix}, \tag{13.20}$$

The top-left block is the Hadamard matrix, as is the bottom-left block. The top-right and bottom-right blocks look are related by a minus sign, and they can be written SH where S is the phase gate

$$S := \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}. \tag{13.21}$$

Thus the matrix can be written in terms of blocks as

$$\widetilde{\text{QFT}}_4 = \frac{1}{\sqrt{2}} \begin{bmatrix} H & SH \\ H & -SH \end{bmatrix} \tag{13.22}$$

This "modified QFT matrix" has a lot of structure—in particular, tensor product structure. I claim that this matrix can be factorized as follows:

$$\widetilde{\text{QFT}}_4 = (H \otimes I)(\Pi_0 \otimes I + \Pi_1 \otimes S)(I \otimes H). \tag{13.23}$$

Recall the notation for projectors: $\Pi_0 := |0\rangle\langle 0|$ and similarly or Π_1.

Exercise 98: Prove (13.23) by expanding the RHS and comparing it to (13.20).

Question: Looking at (13.23), what is a quantum circuit that implements the QFT on two qubits?
Answer: You just have to look at the equation and do what it says. Remember, though, that the *first* gate in the quantum circuit is the *last* gate in the equation. (And the rest continue in reverse order.) The circuit is shown in Figure 13.3 below.

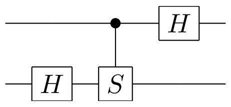

Figure 13.1: Quantum circuit for the $\widetilde{\text{QFT}}_4$ matrix (13.20).

Recall that this is the *modified* QFT_4 gate—it has the order of the inner columns swapped. How can we get the actual QFT_4? Just swap the qubits! This is shown in the circuit in Figure 13.2 below.

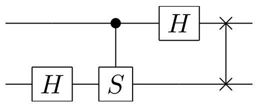

Figure 13.2: Quantum circuit for the QFT_4 matrix.

Exercise 99: Prove that swapping the qubits in Figure (13.2) is equivalent to swapping the inner columns of the $\widetilde{\text{QFT}}_4$ matrix to regain the original QFT_4 matrix.

Finally, we note that the SWAP gate doesn't need to be *physically* implemented here. (It's extremely rare you ever need to do a SWAP gate in practice unless you want to do a two-qubit gate between unconnected qubits—more on this to come later.) We can always "untangle" the qubit wires (i.e., relabel the qubits) to get rid of the SWAP gate. In our case, this amounts to swapping the top and bottom qubit wires, which results in the circuit shown in Figure 13.3 below.

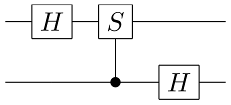

Figure 13.3: Quantum circuit for the QFT_4 matrix without a SWAP gate.

13.2.2 The general recursive pattern

The structure revealed in the QFT_4 case is enlightening. Since we know that the Hadamard gate is the QFT on one qubit (13.17), we can write

$$\widetilde{\text{QFT}}_4 = \begin{bmatrix} \text{QFT}_2 & S\ \text{QFT}_2 \\ \text{QFT}_2 & -S\ \text{QFT}_2 \end{bmatrix} \tag{13.24}$$

In general, it turns out that there is a recursive pattern:

$$\widetilde{\text{QFT}}_N = \begin{bmatrix} \text{QFT}_{N/2} & A_{N/2}\ \text{QFT}_{N/2} \\ \text{QFT}_{N/2} & -A_{N/2}\ \text{QFT}_{N/2} \end{bmatrix} \tag{13.25}$$

This recursive relationship leads directly to the $O(\log^2 N)$ scaling.

What is this $A_{N/2}$ matrix? It turns out to be a series of controlled Z-rotations. For $N = 2$, we know that $A_2 = S$, which is indeed a Z rotation.

Exercise 100: What is the angle θ such that $R_Z(\theta) = S$?

For the general case, define the gate

$$R_k := \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{bmatrix} \tag{13.26}$$

Exercise 101: Express the R_k gate as a Z rotation.

Exercise 102: Verify that $R_k = S$ for $k = 1$.

We now state the QFT circuit in the "general" case. Here, general means on four qubits because TEX'ing QFT circuits is a pain. it is easy to generalize this structure to an arbitrary n qubits, however.

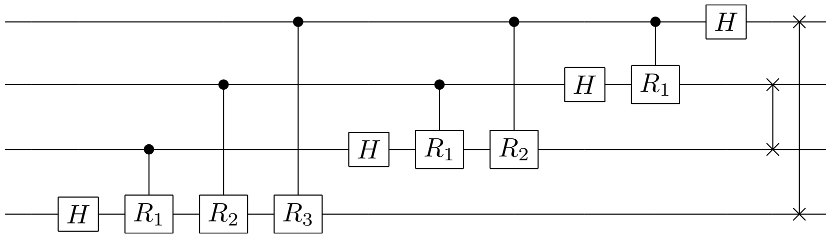

Figure 13.4: The quantum circuit for a QFT on four qubits.

We note again that the SWAP gates can be removed by reordering/relabeling qubits. It is this general structure that the QFT circuit follows for n qubits.

Exercise 103: Draw the QFT circuit on three qubits. *Hint:* You need to swap the "outer" qubits here. When you have an odd number, there's always one qubit in the middle that doesn't need to be swapped.

Exercise 104: Draw the QFT circuit on five qubits.

Exercise 105: For the QFT circuit on four qubits in Figure 13.4, draw the equivalent circuit with no SWAP gates.

⚠ Warning! Remember above we mentioned that the QFT has exponentially better asymptotic scaling than the FFT *but with an important caveat.* It's time we mention that caveat. The QFT is performed with a quantum circuit. As such, Fourier coefficients become *amplitudes in the wavefunction.* This means we can't access them directly! Any users wishing to use the QFT to actually compute Fourier coefficients are vastly misled, and will be persecuted without trial! State tomography is exponential in the number of qubits, so any advantage is instantly lost.

What's the purpose of the QFT then, if we can't even know what we're computing? As is common in quantum algorithms, the utility is different, or more subtle, than you originally think. The point of doing a QFT is to change bases and "see the problem in a new way." We'll see an example of this with Shor's algorithm with the ability to compute periods. Below, we'll highlight a property of the QFT in that it maps frequency information to basis information, and vice-versa.

13.3 Properties of the QFT

The first thing we should verify is the number of single qubit and two-qubit gates in the QFT circuit for n qubits. Since there is a Hadamard gate on each qubit, there will be a total of n single qubit gates. How about the controlled-Z rotations? There are $n-1$ to start, then $n-2$, then $n-3$ and so on until just 1. In total, then, the number of gates (single qubit and two qubit) is

$$1+2+\cdots+(n-2)+(n-1)+n=\frac{n(n+1)}{2}. \tag{13.27}$$

This proves that the asymptotic scaling is $O(\log^2(N))$ where $N=2^n$.

As a note, some of these Z-rotations have *exponentially* small angles. Indeed, by definition of the R_k gate, the angle is of order 2^{-k}. For large k, this is close to zero, so the gate is approximately the identity gate. This is the idea of the approximate QFT: ignore all R_k gates for k greater than some specified cutoff value. Although we won't divulge here, the AQFT can get the scaling down to $O(\log(N)\log\log(N))$.

Now that we have shown how to efficiently construct a QFT circuit, we mention one more property of the QFT which is useful in Shor's algorithm and elsewhere. In fact, this is probably the *most useful* property of the QFT to both know and conceptually understand. First, we'll state the conceptual understanding:

The QFT maps between basis information and frequency information.

This may be unclear now, but it should become clear after the following discussion. After this discussion, re-read this point—it's a good way to understand what the QFT is doing without digging into the mathematical details.

Consider a product state on n qubits

$$|\boldsymbol{j}\rangle := |j_1\rangle|j_2\rangle\cdots|j_n\rangle. \tag{13.28}$$

In the following theorem, we'll use the binary decimal notation

$$0.j_1j_2\cdots j_n := \sum_{k=1}^{n} j_k 2^{-k}. \tag{13.29}$$

Theorem 13.3 (Action of QFT: Basis information $\leftrightarrow$ frequency information). Let $|\boldsymbol{j}\rangle$ be as above. Then,

$$\text{QFT}|j_1\rangle|j_2\rangle\cdots|j_n\rangle=\frac{|0\rangle+e^{2\pi i 0.j_1j_2\cdots j_n}|1\rangle}{\sqrt{2}}\otimes\frac{|0\rangle+e^{2\pi i 0.j_2\cdots j_n}|1\rangle}{\sqrt{2}}\otimes\cdots\otimes\frac{|0\rangle+e^{2\pi i 0.j_n}|1\rangle}{\sqrt{2}}. \tag{13.30}$$

It is useful to highlight the inverse transform as well:

$$\text{QFT}^{-1}\left[\frac{|0\rangle+e^{2\pi i 0.j_1j_2\cdots j_n}|1\rangle}{\sqrt{2}}\otimes\frac{|0\rangle+e^{2\pi i 0.j_2\cdots j_n}|1\rangle}{\sqrt{2}}\otimes\cdots\otimes\frac{|0\rangle+e^{2\pi i 0.j_n}|1\rangle}{\sqrt{2}}\right]=|j_1\rangle|j_2\rangle\cdots|j_n\rangle. \tag{13.31}$$

The current proof I have for this is technical and not very enlightening (the one from Nielsen and Chuang). I'll omit the proof for now until I think of a better explanation.

For now, just note that Theorem 13.3 is potentially the most useful property and the one we'll use in the quantum part of Shor's algorithm, which we'll get to next seminar! Don't let this final fact outweigh the importance of our efficient construction of the QFT, however. Both properties are equally important.

Exercise 106: Construct a quantum circuit for the *inverse* QFT on, say, four qubits. *Hint:* You could proceed from first principles like how we built up the forward QFT, but we already did this once! Try to leverage the knowledge we already have, in particular Figure 13.4.