Quantum Fourier Transform

Jay Mehta

Department of Physics & Astronomy, McMaster University

1 Background

Fourier transform & the Fourier series:

Transforming a problem from its original domain into some other problem for which a solution is known is one of the most useful and clever ways of finding solutions in mathematics. One such transformation is known as the Fourier transform. It originates from the concept that Joseph Fourier introduced (Fourier, Darboux, et al., 1822), first in 1822, that any periodic function can be expressed as an infinite sum of sinusoidal waves, which is namely a Fourier series representation of that function.

Let $f : \Re \to \Re$ be a 2π periodic function. i.e, $f(t) = f(t + 2\pi)$ for all t. The Fourier series of f is given by

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cos(kt) + b_k \sin(kt) \tag{1a}$$

where,

$$a_k = \frac{1}{\pi} \int_0^{2\pi} f(t) \cos(kt) dt, \qquad k = 0, 1, \ldots, \tag{1b}$$

$$b_k = \frac{1}{\pi} \int_0^{2\pi} f(t) \sin(kt) dt, \qquad k = 1, 2, \ldots. \tag{1c}$$

Fourier Transform is essentially just an extension of this idea, by advancing the period of the function to infinity, making it possible to be decomposed in the corresponding Fourier domain as a superposition of complex-valued amplitudes.

The figure 1 shows a replica of an arbitrary function (signal) comprised of many constituent sinusoidal waves of varying frequencies. And as we can see that in the time domain, the function can be visualised as how it propagates with time; However, after the Fourier transform, we can observe all the corresponding frequencies that build-up to the full description of the wave signal. Also, as I mentioned earlier, these Fourier transformed coefficients in this frequency domain are complex in nature, meaning that the complex amplitudes indicate the amount of that particular frequency present in the wave and the complex angle is just the relative phase difference, which is insignificant here.

Mathematically, we may define the Fourier transform of an integrable function $f : \Re \to \mathbf{C}$ as follows:

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x)\, e^{-2\pi i x \xi}\, dx, \quad \forall\, \xi \in \Re \tag{2}$$

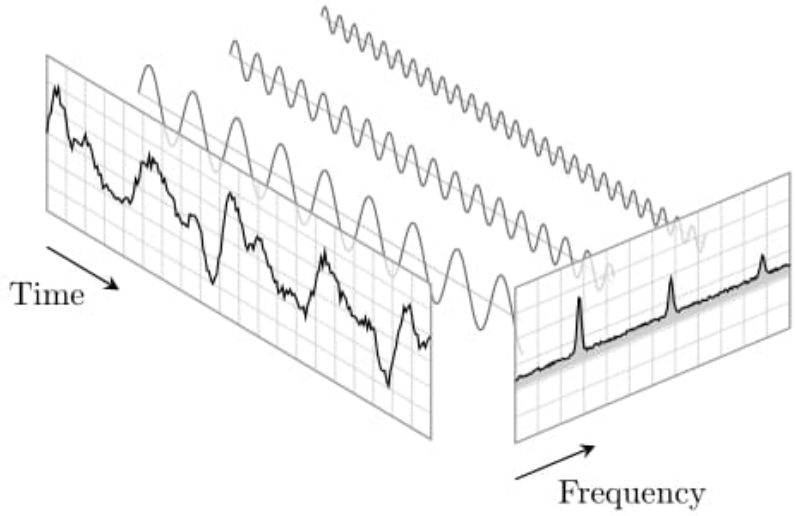


Figure 1: A schematic diagram of a signal, in both time- domain and the corresponding Fourier transformed frequency-domain.

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi)\, e^{2\pi i x \xi}\, d\xi, \quad \forall\, x \in \Re \tag{3}$$

These functions f and $\hat{f}$ in the equations 2 and 3 are typically referred to as a Fourier integral pair or Fourier transform pair. Thus,

$$f(x) \overset{\mathcal{F}}{\longleftrightarrow} \hat{f}(\xi) \tag{4}$$

In fact, let me also point out that, the notion of Fourier transform is not just limited to the functions of time. On the contrary, the domain of the original function is commonly referred to as the time domain. One such interesting example would be, the spatial Fourier transform pair of position and momentum, which arises naturally in the study of wavefunctions in quantum mechanics, and extremely useful to describe spatial waves as functions of either position or momentum or sometimes both.

In the jargon of data-driven science, the Fourier transform occurs in many different versions throughout classical computing, in areas ranging from signal processing to data compression to complexity theory.

Discrete Fourier transform:

The discrete Fourier transform acts on a vector $(x_0, ..., x_{N-1})$ and maps it to the vector $(y_0, ..., y_{N-1})$ according to the formula:

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \omega_N^{jk} \tag{5}$$

where, $\omega_N^{jk} = e^{2\pi i \frac{jk}{N}}$.

An N-point DFT is expressed as the multiplication $Y = W_N X$, where $X = (x_0, ..., x_{N-1})$ is the original input signal, W_N is the $N \times N$ square DFT matrix, and $Y = (y_0, ..., y_{N-1})$ is the DFT of the signal. And the matrix,

$$W_N = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_N^1 & \omega_N^2 & \dots & \omega_N^{N-1} \\ 1 & \omega_N^2 & \omega_N^4 & \dots & \omega_N^{2(N-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega_N^{N-1} & \omega_N^{2(N-1)} & \dots & \omega_N^{(N-1)^2} \end{pmatrix} \tag{6}$$

is called the *Fourier matrix*. This DFT matrix W_N is a unitary Vandermonde matrix.

As we have seen, given the nature of Fourier transforms, it is obvious that the analytical Fourier series and transforms are defined for continuous functions. Naturally, the discrete and continuous formulations should match the limit of inputs with infinitely fine resolution.

Fast Fourier transform

The DFT is tremendously useful for numerical approximation and computation, but it does not scale well to very large $N >> 1$, as the simple formulation involves multiplication by a dense $N \times N$ matrix, requiring $O(N^2)$ operations. In 1965, James W. Cooley and John W. Tukey developed the revolutionary fast Fourier transform (FFT) algorithm (Cooley and Tukey, 1965) that scales as $O(N\ log(N))$. As N becomes very large, the $log(N)$ component grows slowly, and the algorithm approaches a linear scaling. Their algorithm was based on a fractal symmetry in the Fourier transform that allows an N - dimensional DFT (as described in the equation 6) to be solved with a number of smaller dimensional DFT computations. Interestingly, similar techniques can be applied for multiplications by matrices such as Hadamard matrix and the Walsh matrix.

The quantum Fourier transform (QFT) is the quantum implementation of the discrete Fourier transform over the amplitudes of a wavefunction. It is part of many quantum algorithms, most notably Shor's factoring algorithm and quantum phase estimation.

2 Quantum Fourier Transform (QFT)

Introduction:

The quantum Fourier transform (QFT) transforms between two bases, the computational (Z) basis, and the Fourier basis. And its the quantum analogue of the discrete Fourier transform.

Similar to DFT, the quantum Fourier transform acts on a quantum state $|X\rangle = \sum_{j=0}^{N-1} x_j|j\rangle$ and maps it to the quantum state $|Y\rangle = \sum_{k=0}^{N-1} y_k|k\rangle$ according to the formula,

$$|j\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega_N^{jk}|k\rangle \tag{7}$$

Note that only the amplitudes of the state have been affected by this transformation. And $\omega_N^{jk} = e^{2\pi i \frac{jk}{N}}$ as above (Nielson and Chuang, 2000).

This mapping can be represented in a compact form using the following unitary matrix,

$$U_{QFT} = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} \omega_N^{jk}|k\rangle\langle j| \tag{8}$$

1-qubit QFT:

Consider a single qubit state vector $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. We can apply the QFT operator as defined above (equation 8) to act on the state $|\psi\rangle$ with $\mathbf{N = 2}$:

$$y_0 = \frac{1}{\sqrt{2}} \left(\alpha e^{\left(2\pi i \frac{0\times 0}{2}\right)} + \beta e^{\left(2\pi i \frac{1\times 0}{2}\right)} \right) = \frac{1}{\sqrt{2}} (\alpha + \beta)$$

And,

$$y_1 = \frac{1}{\sqrt{2}} \left(\alpha e^{\left(2\pi i \frac{0\times 1}{2}\right)} + \beta e^{\left(2\pi i \frac{1\times 1}{2}\right)} \right) = \frac{1}{\sqrt{2}} (\alpha - \beta)$$

The aforementioned operation is exactly the same result we may obtain by applying the Hadamard operator **(H)** on the single qubit state vector, as in the equation 9:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{9}$$

$$U_{QFT}|\psi\rangle = H(\alpha|0\rangle + \beta|1\rangle) = \frac{1}{\sqrt{2}}(\alpha+\beta)|0\rangle + \frac{1}{\sqrt{2}}(\alpha-\beta)|1\rangle$$

Thus, the Hadamard gate **(H)** performs the discrete Fourier transform for $\mathbf{N = 2}$ on the amplitudes of the state $|\psi\rangle$.

n-qubit QFT:

In case of $N = 2^n$, consider $|x\rangle = |x_1 \dots x_n\rangle$. Then, the quantum Fourier transform for the n-qubit state vector looks like the following (with usual notation);

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

$$U_{QFT}|x\rangle = \frac{1}{\sqrt{N}} \left(|0\rangle + e^{\frac{2\pi i}{2}x}|1\rangle\right) \otimes \left(|0\rangle + e^{\frac{2\pi i}{2^2}x}|1\rangle\right) \dots \left(|0\rangle + e^{\frac{2\pi i}{2^{n-1}}x}|1\rangle\right) \otimes \left(|0\rangle + e^{\frac{2\pi i}{2^n}x}|1\rangle\right)$$

3 Circuit Implementation

Here's an overview of a 3-qubit QFT circuit. Consider the following figure 2:

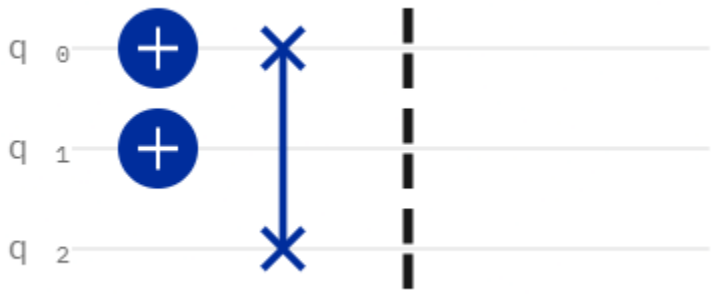
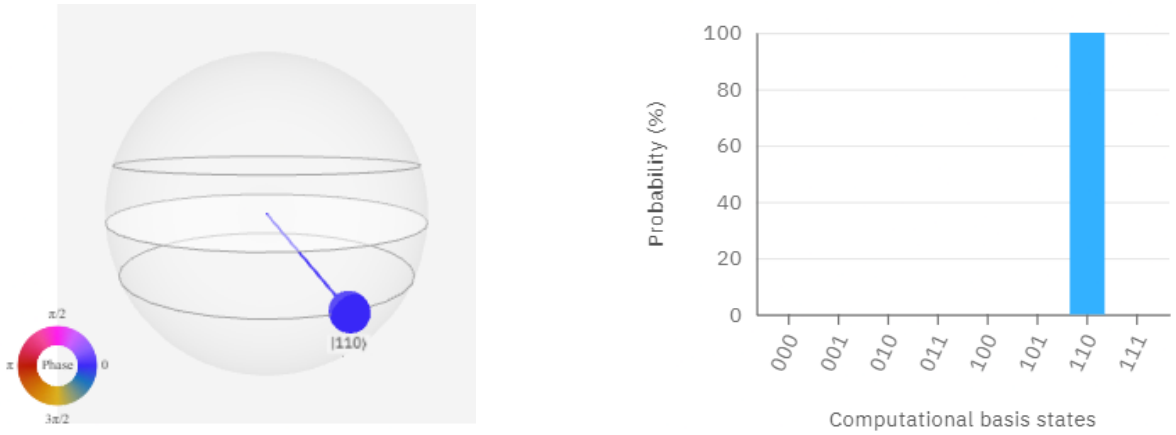


Figure 2: The initial state of the circuit has been set to $|110\rangle$.



(a) Q-sphere representation of $|110\rangle$

(b) Initial probabilities' histogram

Figure 3: Parameters of the initial state.

I have carried out these tasks on Qiskit (IBM Quantum Computing).

Now, in the following figure 4, the QFT algorithm has been applied on our initial choice of state vector.

Figure 4: The final state of the circuit after implementing QFT on the initial state vector.

The circuit that implements QFT makes use of two gates. The first one is a single-qubit Hadamard gate (**H**) as described in the equation 9. The second is a two-qubit controlled rotation $CROT_k$ given in block-diagonal form as,

$$CROT_k = \begin{bmatrix} I & 0 \\ 0 & UROT_k \end{bmatrix}$$

Where,

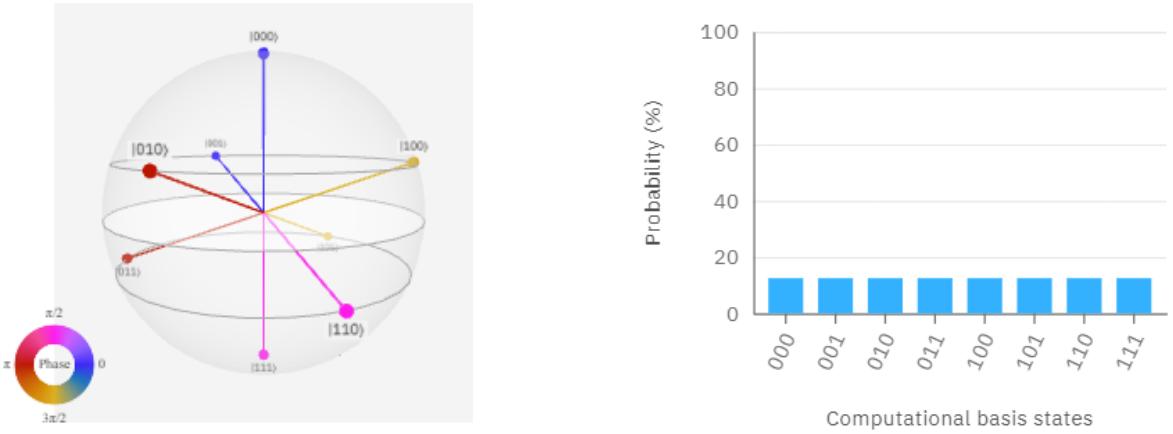
$$UROT_k = \begin{bmatrix} 1 & 0 \\ 0 & \exp\left(\frac{2\pi i}{2^k}\right) \end{bmatrix}$$

In Qiskit, the implementation of the $CROT$ gate (a controlled phase rotation gate) is defined as,

$$CP(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\theta} \end{bmatrix}$$

So, for a 3-qubit input state, $|y_1 y_2 y_3\rangle = U_{QFT}|x_1 x_2 x_3\rangle$ and we get the following final state:

$$\frac{1}{\sqrt{2}}\left[|0\rangle + e^{\left(\frac{2\pi i}{2}x_3\right)}|1\rangle\right] \otimes \frac{1}{\sqrt{2}}\left[|0\rangle + e^{\left(\frac{2\pi i}{2^2}x_3 + \frac{2\pi i}{2}x_2\right)}|1\rangle\right] \otimes \frac{1}{\sqrt{2}}\left[|0\rangle + e^{\left(\frac{2\pi i}{2^3}x_3 + \frac{2\pi i}{2^2}x_2 + \frac{2\pi i}{2}x_1\right)}|1\rangle\right]$$



(a) Q-sphere representation of the final state

(b) Final probabilities' histogram

Figure 5: Parameters of the final state.

4 Applications & Insights

The example above demonstrates a very useful form of the QFT for $N = 2^n$ ($n = 3$ in our case!). It's worth noting that only the last qubit is affected by the values of all the other input qubits, with each subsequent bit becoming less and less dependent on the input qubits. This is significant in practical implementations of the QFT, since nearest-neighbor couplings between qubits are easier to produce than distant couplings.

Furthermore, as the QFT circuit grows in size, more time is spent performing progressively minor rotations. It turns out that if we ignore rotations below a particular threshold, we can still obtain

decent results; this is referred to as the approximate QFT. In physical implementations, this is especially significant because lowering the number of processes reduces decoherence and associated gate errors.

Probably the most evident application of the quantum Fourier transform is in the quantum phase estimation algorithm and Peter Shor's algorithm (Beauregard, 2002), where the inverse QFT gate is applied on the period-finding quantum subroutine.

References

Beauregard, Stephane (2002). "Circuit for Shor's algorithm using 2n+3 qubits". In: DOI: 10.48550/ARXIV.QUANT-PH/0205095. URL: https://arxiv.org/abs/quant-ph/0205095.

Cooley, James W and John W Tukey (1965). "An algorithm for the machine calculation of complex Fourier series". In: *Mathematics of computation* 19.90, pp. 297–301.

Fourier, Jean Baptiste Joseph, Gaston Darboux, et al. (1822). *Théorie analytique de la chaleur*. Vol. 504. Didot Paris.

Nielson, Michael A and Isaac L Chuang (2000). *Quantum computation and quantum information*.