III
Quantum Fourier Transform

In classical term Discrete Fourier Transform (DFT), is a map from N complex numbers to N complex numbers (for a vector f):

$$\tilde{f}_j \;=\; \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{-jk}\, f_k. \tag{III.1}$$

where

$$\omega \;=\; \exp\!\left(\frac{2\pi i}{N}\right). \tag{III.2}$$

And, inverse DFT is defined as:

$$f_j \;=\; \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{jk}\, \tilde{f}_k. \tag{III.3}$$

Let $f_k^{(\ell)} = \delta_{k,\ell}$. Then:

$$\tilde{f}_j^{(\ell)} = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{-jk}\, \delta_{k,\ell} = \frac{1}{\sqrt{N}}\, \omega^{-j\ell}. \tag{III.4}$$

Orthonormality:

$$\sum_j \left(\tilde{f}_j^{(\ell)}\right)^* \tilde{f}_j^{(m)} = \left(\frac{1}{\sqrt{N}}\right)\left(\frac{1}{\sqrt{N}}\right)\sum_j \omega^{j\ell}\,\omega^{-jm} \tag{III.5}$$

$$= \frac{1}{N}\sum_j \omega^{j(\ell-m)} = \delta_{\ell m}. \tag{III.6}$$

Another important property is the convolution theorem:

$$(f * g)_k \;=\; \sum_{j=0}^{N-1} f_j\, g_{k-j} \qquad (\mathrm{mod}\, N), \tag{III.7}$$

where (circular convolution) $g_{-m} = g_{N-m}$.

Detour: what does $k - j$ mean? Let's understand it with an example, Assume:

$$f = [1, 2, 3, 4], \qquad g = [5, 6, 7, 8].$$

Then

$$(f * g)_0 = f_0\, g_{0-0} + f_1\, g_{0-1} + f_2\, g_{0-2} + f_3\, g_{0-3}. \tag{III.8}$$

Using indices modulo $N = 4$:

$$g_0 = 5, \qquad g_{0-1} = g_3 = 8, \qquad g_{0-2} = g_2 = 7, \qquad g_{0-3} = g_1 = 6.$$

So

$$f * g = \big[(f * g)_0,\, (f * g)_1,\, (f * g)_2,\, (f * g)_3\big].$$

Fast Fourier Transform (FFT) and applications

For the DFT we need to do N multiplications and then add them, why?

For a vector $x = [x_0, x_1, \ldots, x_{N-1}]$, the DFT is defined as

$$\hat{x}_k \;=\; \sum_{n=0}^{N-1} x_n \, e^{-2\pi i k n/N}, \qquad k = 0, 1, \ldots, N-1. \tag{III.9}$$

So, for each k you compute a sum of N terms:

- N complex multiplications and $\sim (N-1)$ additions,
- total work $\sim N(N-1) \approx N^2$,
- hence $\mathcal{O}(N^2)$ complexity.

Fast Fourier Transform (FFT)

Fast Fourier helps make the process faster. Here we assume $N = 2^n$ (the length is a power of 2).

To make the process fast, split the components of f into smaller vectors of size $N/2$:

$e_i = f_{2i}$ (even components), $\qquad o_i = f_{2i+1}$ (odd components).

Let $\omega_N = e^{2\pi i/N}$. Then:

$$\tilde{f}_j = \sum_{i=0}^{N-1} f_i \, \omega_N^{-ij} \tag{III.10}$$

$$= \sum_{i=0}^{N/2-1} f_{2i} \, \omega_N^{-j(2i)} \;+\; \sum_{i=0}^{N/2-1} f_{2i+1} \, \omega_N^{-j(2i+1)} \tag{III.11}$$

$$= \sum_{i=0}^{N/2-1} e_i \, \omega_N^{-2ji} \;+\; \sum_{i=0}^{N/2-1} o_i \, \omega_N^{-j} \, \omega_N^{-2ji}. \tag{III.12}$$

Now, let $\omega_N = e^{2\pi i/N}$ and assume N is even. After splitting

$$e_i := f_{2i} \quad \text{(even part)}, \qquad o_i := f_{2i+1} \quad \text{(odd part)}, \qquad i = 0, \ldots, \frac{N}{2} - 1,$$

the DFT can be written as

$$\tilde{f}_j = \sum_{i=0}^{\frac{N}{2}-1} f_{2i}\,\omega_N^{-j(2i)} \;+\; \sum_{i=0}^{\frac{N}{2}-1} f_{2i+1}\,\omega_N^{-j(2i+1)} \tag{III.13}$$

$$= \sum_{i=0}^{\frac{N}{2}-1} e_i\,\omega_N^{-2ji} \;+\; \omega_N^{-j} \sum_{i=0}^{\frac{N}{2}-1} o_i\,\omega_N^{-2ji}. \tag{III.14}$$

Note that

$$\omega_N^{-2j} = \left(e^{2\pi i/N}\right)^{-2j} = e^{-2\pi ij/(N/2)} = \omega_{N/2}^{-j}.$$

Hence $\omega_N^{-2ji} = \omega_{N/2}^{-ji}$ and therefore

$$\tilde{f}_j = \sum_{i=0}^{\frac{N}{2}-1} e_i\,\omega_{N/2}^{-ji} \;+\; \omega_N^{-j} \sum_{i=0}^{\frac{N}{2}-1} o_i\,\omega_{N/2}^{-ji}. \tag{III.15}$$

Define the length-$N/2$ DFTs

$$\tilde{e}_j := \sum_{i=0}^{\frac{N}{2}-1} e_i\,\omega_{N/2}^{-ji}, \qquad \tilde{o}_j := \sum_{i=0}^{\frac{N}{2}-1} o_i\,\omega_{N/2}^{-ji}, \qquad j = 0, \ldots, \frac{N}{2} - 1.$$

Then, for $j = 0, 1, \ldots, \frac{N}{2} - 1$,

$$\boxed{\tilde{f}_j = \tilde{e}_j + \omega_N^{-j}\,\tilde{o}_j} \tag{III.16}$$

The second half: $\tilde{f}_{j+N/2}$

Now compute for $j = \frac{N}{2} - 1$,...,N:

$$\tilde{f}_{j+\frac{N}{2}} = \tilde{e}_{j+\frac{N}{2}} + \omega_N^{-(j+\frac{N}{2})}\,\tilde{o}_{j+\frac{N}{2}}. \tag{III.17}$$

Because $\tilde{e}_j$ and $\tilde{o}_j$ are DFTs of length $N/2$, they are periodic with period $N/2$:

$$\tilde{e}_{j+\frac{N}{2}} = \tilde{e}_j, \qquad \tilde{o}_{j+\frac{N}{2}} = \tilde{o}_j.$$

Also,

$$\omega_N^{-(j+\frac{N}{2})} = \omega_N^{-j}\,\omega_N^{-\frac{N}{2}}, \qquad \omega_N^{-\frac{N}{2}} = e^{-2\pi i\frac{N/2}{N}} = e^{-\pi i} = -1.$$

So

$$\omega_N^{-(j+\frac{N}{2})} = -\,\omega_N^{-j},$$

and therefore

$$\boxed{\tilde{f}_{j+\frac{N}{2}} = \tilde{e}_j - \omega_N^{-j}\,\tilde{o}_j} \tag{III.18}$$

Cost of one split (vs. full DFT)

The full DFT is $\mathcal{O}(N^2)$. After one split, we compute two DFTs of size $N/2$:

$$2\left(\frac{N}{2}\right)^2 = \frac{N^2}{2}.$$

Then we combine them using

$$\tilde{f}_j = \tilde{e}_j + \omega_N^{-j}\tilde{o}_j, \qquad \tilde{f}_{j+\frac{N}{2}} = \tilde{e}_j - \omega_N^{-j}\tilde{o}_j,$$

which requires multiplying by ω_N^{-j} for $j = 0, \ldots, \frac{N}{2} - 1$, i.e. about $\frac{N}{2}$ extra multiplications.

So the (very rough) multiplication count becomes

$$\frac{N^2}{2} + \frac{N}{2},$$

which is about $\sim 50\%$ fewer multiplications than the original N^2 (for large N).

Recursive idea and the $N \log N$ bound

Instead of doing one split, recursively split each DFT of size $N/2$ into two DFTs of size $N/4$, and so on, until $N = 2$.

Let $N = 2^n$ and define T_n as the number of complex multiplications required for an input of size 2^n. At each stage:

- we compute two smaller FFTs: cost $2T_{n-1}$,
- we multiply by twiddle factors ω_N^{-j} for $j = 0, \ldots, 2^{n-1} - 1$: cost 2^{n-1}.

So the recurrence is

$$\boxed{T_n = 2T_{n-1} + 2^{n-1}} \tag{III.19}$$

A convenient bound/solution is:

$$T_n \leq n\, 2^{n-1}.$$

Since $N = 2^n$, this is

$$T_n = \mathcal{O}(n2^n) = \mathcal{O}(N \log N).$$

So the running time is bounded by $N \log N$.

Application of FFT: fast polynomial multiplication

Let

$$f(x) = a_0 + a_1 x + \cdots + a_{N-1} x^{N-1}, \qquad g(x) = b_0 + b_1 x + \cdots + b_{N-1} x^{N-1}.$$

Then

$$f(x)g(x) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} a_i b_j\, x^{i+j} = \sum_{k=0}^{2(N-1)} c_k x^k, \tag{III.20}$$

where the coefficients are the convolution

$$c_k = \sum_{i=0}^{k} a_i\, b_{k-i}. \tag{III.21}$$

Directly computing all c_k costs $\mathcal{O}(N^2)$.

Alternative via FFT (convolution theorem)

$$\mathrm{DFT}(f * g) = \mathrm{DFT}(f) \cdot \mathrm{DFT}(g) \quad \text{(pointwise product)}.$$

Practical steps:

1. Zero-pad the coefficient vectors so both have the same length (typically at least $2N$).
2. Compute FFTs:
$$\tilde{a} = \mathrm{FFT}(a), \qquad \tilde{b} = \mathrm{FFT}(b) \qquad \Rightarrow \qquad \mathcal{O}(N \log N).$$
3. Multiply pointwise:
$$\tilde{c}_u = \tilde{a}_u \, \tilde{b}_u \qquad \Rightarrow \qquad \mathcal{O}(N).$$
4. Inverse FFT to get c:
$$c = \mathrm{IFFT}(\tilde{c}) \qquad \Rightarrow \qquad \mathcal{O}(N \log N).$$

Quantum Fourier Transform

Now, we start exploring the quantum Fourier transform (QFT). In this case, lets assume a classical DFT for $N = 2$ and see how it is related to Hadamard:

Let
$$a = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix}, \qquad \tilde{a} = \begin{pmatrix} \tilde{a}_0 \\ \tilde{a}_1 \end{pmatrix}, \qquad (N = 2).$$

The (classical) DFT is

$$\tilde{a}_k = \sum_{n=0}^{N-1} a_n \, \omega_N^{kn}, \qquad \omega_N = e^{-2\pi i/N}.$$

For $N = 2$,
$$\omega_2 = e^{-2\pi i/2} = e^{-\pi i} = -1.$$

Hence

$$k = 0: \quad \tilde{a}_0 = a_0\omega_2^0 + a_1\omega_2^0 = a_0 + a_1,$$
$$k = 1: \quad \tilde{a}_1 = a_0\omega_2^0 + a_1\omega_2^1 = a_0 - a_1.$$

As a matrix multiplication,

$$\tilde{a} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \end{pmatrix}.$$

This looks like the Hadamard transform:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \qquad (N = 2).$$

So if we start with the state $a_0 |0\rangle + a_1 |1\rangle$, then applying H gives

$$H\big(a_0 |0\rangle + a_1 |1\rangle\big) = \frac{1}{\sqrt{2}}(a_0 + a_1) |0\rangle + \frac{1}{\sqrt{2}}(a_0 - a_1) |1\rangle = \tilde{a}_0 |0\rangle + \tilde{a}_1 |1\rangle,$$

where (with the QFT normalization)

$$\tilde{a}_0 = \frac{a_0 + a_1}{\sqrt{2}}, \qquad \tilde{a}_1 = \frac{a_0 - a_1}{\sqrt{2}}.$$

Definition of QFT

Write a state $\sum_{x=0}^{N-1} a_x |x\rangle$. The QFT produces amplitudes

$$\tilde{a}_x = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \omega_N^{-xy}\, a_y,$$

So the QFT is:

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

Thus the QFT operator can be written as

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

Unitarity check (sketch)

Compute:

$$\begin{aligned} U_{\mathrm{QFT}} U_{\mathrm{QFT}}^{\dagger} &= \frac{1}{N} \sum_{x,y} \sum_{x',y'} \omega_N^{-yx}\, \omega_N^{y'x'}\ |y\rangle \langle x|\, x' \langle y'| \\ &= \frac{1}{N} \sum_{x} \sum_{y,y'} \omega_N^{-yx}\, \omega_N^{y'x}\ |y\rangle \langle y'| \\ &= \frac{1}{N} \sum_{y,y'} \left(\sum_{x=0}^{N-1} \omega_N^{x(y'-y)} \right) |y\rangle \langle y'| \,. \end{aligned}$$

Using the standard identity

$$\sum_{x=0}^{N-1} \omega_N^{x(y'-y)} = N\, \delta_{y,y'},$$

we get

$$U_{\mathrm{QFT}} U_{\mathrm{QFT}}^{\dagger} = \sum_{y=0}^{N-1} |y\rangle \langle y| = I.$$

Deeper structure for $N = 2^n$

Assume $N = 2^n$ and $\omega_N = e^{-2\pi i/N}$. Then

$$|x\rangle \longmapsto \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} \omega_N^{-xy} |y\rangle \,, \qquad x \in \{0, 1, \ldots, 2^n - 1\}.$$

Write y in binary:

$$y \in \{0, 1, \ldots, 2^n - 1\}, \qquad y = \sum_{k=1}^{n} y_k\, 2^{\,n-k}, \qquad y_k \in \{0, 1\}.$$

Then

$$|x\rangle \longmapsto \frac{1}{\sqrt{2^n}} \sum_{y_1,\ldots,y_n \in \{0,1\}} \omega_N^{-x\left(\sum_{k=1}^{n} y_k 2^{n-k}\right)} |y_1 \cdots y_n\rangle \,.$$

Because exponents add, we can factor:

$$\omega_N^{-x\left(\sum_{k=1}^{n} y_k 2^{n-k}\right)} = \prod_{k=1}^{n} \omega_N^{-x\,2^{n-k}\,y_k}.$$

So the transform becomes

$$|x\rangle \longmapsto \frac{1}{\sqrt{2^n}} \sum_{y_1,\ldots,y_n \in \{0,1\}} \left(\prod_{k=1}^{n} \omega_N^{-x\,2^{n-k}\,y_k} \right) |y_1 \cdots y_n\rangle .$$

For each k, since $y_k \in \{0,1\}$,

$$\sum_{y_k \in \{0,1\}} \omega_N^{-x\,2^{n-k}\,y_k} |y_k\rangle = |0\rangle + \omega_N^{-x\,2^{n-k}} |1\rangle .$$

Therefore the whole QFT output factors as a tensor product:

$$\boxed{|x\rangle \longmapsto \frac{1}{\sqrt{2^n}} \bigotimes_{k=1}^{n} \left(|0\rangle + \omega_N^{-x\,2^{n-k}} |1\rangle \right).}$$

Writing phases using the bits of x (binary fraction)

Write x in binary:

$$|x\rangle = |x_1 x_2 \cdots x_n\rangle , \qquad x_j \in \{0,1\},$$

where x_1 is the most significant bit (MSB) and x_n is the least significant bit (LSB), so

$$x = x_1 2^{n-1} + x_2 2^{n-2} + \cdots + x_n 2^0.$$

One often rewrites the QFT result in the "gate-by-gate" form:

$$\boxed{|x\rangle \longmapsto \bigotimes_{k=1}^{n} \left(\frac{1}{\sqrt{2}} \left(|0\rangle + e^{2\pi i\,(0.x_k x_{k+1} \cdots x_n)} |1\rangle \right) \right)}$$

where $0.x_k x_{k+1} \cdots x_n$ denotes a *binary fraction*.

Binary fraction example

For example, if $|x\rangle = |101\rangle$ then $x_1 = 1,\ x_2 = 0,\ x_3 = 1$ and

$$0.x_1x_2x_3 = 0.101_2 = \frac{1}{2} + \frac{0}{4} + \frac{1}{8}.$$

The second qubit phase involves

$$0.x_2x_3 = 0.01_2 = \frac{0}{2} + \frac{1}{4} = \frac{1}{4}.$$

The third qubit phase involves

$$0.x_3 = 0.1_2 = \frac{1}{2}.$$

So each output qubit "sees" fewer and fewer of the input bits.

Circuit example: $n = 3$

Let

$$|x_1x_2x_3\rangle = |101\rangle\,, \qquad (x_1 = 1,\ x_2 = 0,\ x_3 = 1).$$

III.0.1 Hadamard on a single qubit.

If Hadamard acts on a qubit $|b\rangle$ with $b \in \{0, 1\}$, then

$$H\,|b\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + (-1)^b\,|1\rangle\right).$$

Step 1: act on qubit 1

Apply H to the first qubit ($x_1 = 1$):

$$H\,|1\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right).$$

(Equivalently, since $0.x_1 = 0.1_2 = \frac{1}{2}$, we have $e^{2\pi i(0.x_1)} = e^{2\pi i(1/2)} = e^{\pi i} = -1$.)

Controlled phase rotations on qubit 1

The controlled-R_k phase gates (acting on the target when the control is $|1\rangle$) are:

$$R_1 = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2} \end{pmatrix}, \qquad R_2 = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/4} \end{pmatrix}, \qquad R_3 = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/8} \end{pmatrix}.$$

For $|x\rangle = |1, 0, 1\rangle$:

$$x_2 = 0 \;\Rightarrow\; \text{control is } 0 \Rightarrow \text{no action (on qubit 1)},$$

$$x_3 = 1 \;\Rightarrow\; \text{control is } 1 \Rightarrow \text{apply } R_3 \text{ to qubit 1}.$$

Since

$$e^{2\pi i/8} = e^{i\pi/4},$$

the first-qubit state updates from

$$\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \quad \longmapsto \quad \frac{1}{\sqrt{2}}\left(|0\rangle - e^{i\pi/4}\,|1\rangle\right).$$

Step 2: move to qubit 2

The second qubit is initially $|0\rangle$, so after Hadamard:

$$H\,|0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right).$$

Because $x_3 = 1$, the controlled-R_2 acting on qubit 2 is activated, so

$$|1\rangle \;\mapsto\; e^{2\pi i/4}\,|1\rangle = e^{i\pi/2}\,|1\rangle = i\,|1\rangle.$$

Hence the second-qubit state becomes

$$\frac{1}{\sqrt{2}}\left(|0\rangle + i\,|1\rangle\right).$$

Step 3: move to qubit 3

Now go to the third qubit. Since $x_3 = 1$,

$$H|1\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right).$$

There are no controlled-R_k gates from qubits below qubit 3, so no further controlled phase gate is applied to qubit 3.