# Fault tolerant quantum computing:

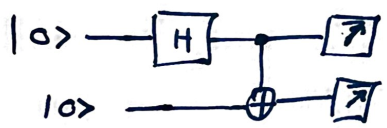


Another application of quantum error correction is not just providing a protection as the data is transmitted through a noisy channel but protect the information as it goes through some gates.

→ it turns out correct results can be obtained even if we we are using ~~a~~ faulty gates provided the error probability per gate is below a certain constant threshold.

The basic idea: Compute directly on the encoded data (quantum states) in such a way that decoding is never required.

Assuming a simple quantum circuit:

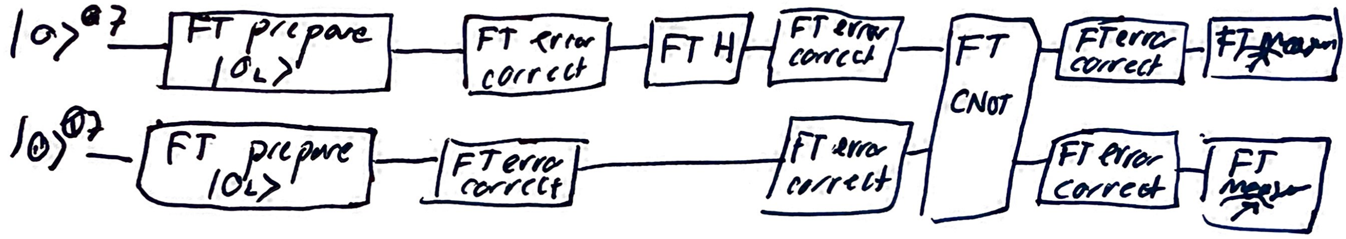



The error could be seen everywhere:

- state initialization
- logic gates
- transmission of qubits (wires)
- measurement

To overcome these errors, first we replace each qubit with an encoded block of qubits. This is done by an error correcting code, such as 7-qubit steane code, and replace the gates with the gates that work with the encoded data. However this not enough since the encoded gates are likely to produce errors that may not be correctable. Therefore encoded gates should be designed carefully so that they can only propagate errors to small number of qubits in the encoded data so that they can be corrected. Such gates are called fault tolerant procedures.

A second problem could be that error correction (circuit) can also introduce errors but they can be designed (similar to fault-tolerant gates) so that they do not introduce too many errors in the encoded data.

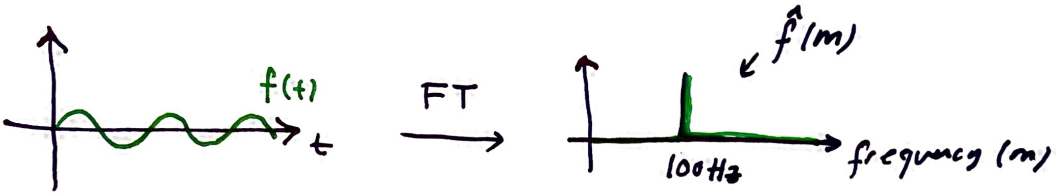


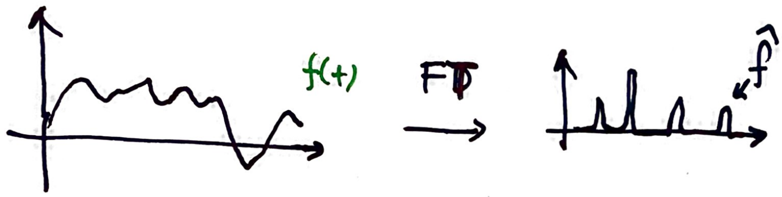

See the book (Nielsen and Chuang) pp. 478-491 for the design of fault tolerant gates and the threshold theorem.

See the book pp. 497-499 for further references on fault tolerant quantum computing.

# Quantum Fourier Transform

## 1) Continous Fourier Transform :




## 2) Discrete Fourier Transform :

A discrete time equivalent of continous Fourier transform

Data — patterns (Freqs.) in data

$$\begin{bmatrix} 1 & 1 & 1 & \cdots \\ 1 & w & w^2 & \cdots \\ \vdots & & \text{DFT matrix} & \\ \vdots & & & \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ \vdots \\ \vdots \end{bmatrix} = \begin{bmatrix} \hat{f}(0) \\ \hat{f}(1) \\ \vdots \\ \vdots \end{bmatrix}$$

(or mathematically
$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \underbrace{e^{2\pi i jk/N}}_{\omega_N^{jk}}$$
)

DFT ≡ a matrix-vector multiplication

**Cost** : Naive → $O(n^2)$
Fast (FFT) → $O(n \log n)$

A Quantum computer can do DFT ≈ $O(\log n)$ steps!

$|\text{state in computational basis}\rangle \xrightarrow{QFT} |\text{state in Fourier basis}\rangle$

$$QFT\,|x\rangle = |\tilde{x}\rangle$$

[redacted]

2

~~(qubit...)~~

## 1-qubit QFT:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \qquad \begin{matrix} x_0=\alpha \\ x_1=\beta \end{matrix} \quad \text{and } N=2$$

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

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

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

which means $U_{QFT} = H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

## n-bit QFT:

$(N=2^n)$

QFT on an orthonormal basis $|0\rangle, \dots, |N-1\rangle$ is defined to be,

**Notation**

$j = j_1 j_2 \cdots j_n$

i.e. $j = j_1 2^{n-1} + j_2 2^{n-2} + \dots + j_n 2^0$ ($j$ is a binary number)

and $0.j_\ell j_{\ell+1} \cdots j_m = \frac{j_\ell}{2} + \frac{j_{\ell+1}}{4} + \dots + \frac{j_m}{2^{m-\ell+1}}$

$$F_N|k\rangle = |j\rangle \longrightarrow \frac{1}{\sqrt{N}}\sum_{k=0}^{N-1} e^{2\pi i jk/N}|k\rangle$$

$$= \frac{1}{2^{n/2}}\sum_{k=0}^{2^n-1} e^{2\pi i jk/2^n}|k\rangle$$

$$= \frac{1}{2^{n/2}}\sum_{k_1=0}^{1}\cdots\sum_{k_n=0}^{1} e^{2\pi i j\left(\sum_{\ell=1}^{n} k_\ell 2^{-\ell}\right)}|k_1 \cdots k_n\rangle$$

$$= \frac{1}{2^{n/2}}\sum_{k_1=0}^{1}\cdots\sum_{k_n=0}^{1}\bigotimes_{\ell=1}^{n} e^{2\pi i j k_\ell 2^{-\ell}}|k_\ell\rangle$$

$$= \frac{1}{2^{n/2}}\bigotimes_{\ell=1}^{n}\left[\sum_{k_\ell=0}^{1} e^{2\pi i j k_\ell 2^{-\ell}}|k_\ell\rangle\right]$$

$$= \frac{1}{2^{n/2}}\bigotimes_{\ell=1}^{n}\left[|0\rangle + e^{2\pi i j 2^{-\ell}}|1\rangle\right]$$

$$= \frac{(|0\rangle + e^{2\pi i 0.j_n}|1\rangle)(|0\rangle + e^{2\pi i 0.j_{n-1}j_n}|1\rangle)\cdots\cdots(|0\rangle + e^{2\pi i 0.j_1 j_2 \cdots j_n}|1\rangle)}{2^{n/2}}$$

define: $R_s = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^s} \end{pmatrix}$, note that $R_1 = Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

$R_2 = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$, and for large $s$, $e^{2\pi i/2^s} \sim 1$ and $R_s \sim I$

example: $n=3$

$$F_8|k_1k_2k_3\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 0.k_3}|1\rangle\right)\otimes\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 0.k_2k_3}|1\rangle\right)$$
$$\otimes\frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 0.k_1k_2k_3}|1\rangle\right)$$

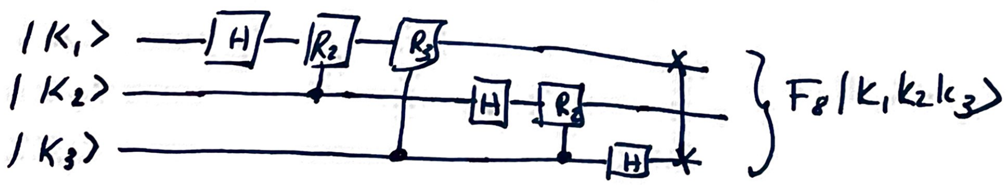
This suggests how the circuit should be.

For the first qubit, we can just apply a Hadamard gate to $|k_3\rangle$, which results in $\frac{1}{\sqrt{2}}(|0\rangle + (-1)^{k_3}|1\rangle)$ and observe that $(-1)^{k_3} = e^{2\pi i 0.k_3}$.

For the second qubits, apply Hadamard gate to $|k_2\rangle$, obtaining $\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 0.k_2}|1\rangle)$ then conditioned on $|k_3\rangle$ apply a rotation, $R_2$. This multiplies $|1\rangle$ with a phase $e^{2\pi i 0.0k_3}$, producing $\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 0.k_2k_3}|1\rangle)$

for the last qubit we apply Hadamard to $|k_1\rangle$ and apply $R_2$ conditioned on $k_2$ and $R_3$ conditioned on $k_3$ which produces $\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 0.k_1k_2k_3}|1\rangle)$, we now produced $F_8|k_1k_2k_3\rangle$ (but in wrong order)

So we swap qubits 1 and 3.




* at most $n$ gates are applied to each qubit
* total $O(n^2)$ gates
* most of the gates are phase gates that are very close to identity and doe not do much anyway ($R_s$ with $s \gg \log n$)
* As observed by Coppersmith, "An approximate Fourier transform useful in quantum factoring" (IBM research report) we can safely omit them keeping only $O(\log n)$ gates per qubit and $O(n \log n)$ gates overall.