COMPLETE COURSE

Quantum Computation with Applications in Finance

An Introduction

Oswaldo Zapata, PhD

www.ozatp.com

Part I - The State Vector Approach

1. Quantum Circuits
   - Classical Bits
   - Single Qubits
   - Multiple Qubits
   - Classical Gates
   - Single-Qubit Gates
   - Multiple Single-Qubit Gates
   - Multi-Qubit Gates
   - Measurements
2. Quantum Algorithms
   - Deutsch's Algorithm
   - The Quantum Fourier Transform
   - Shor's Factoring Algorithm
   - Grover's Search Algorithm

3. Quantum Error Correction
   - Entanglement with the Environment
   - Classical Error Correction
   - Generalities on QEC Codes
   - Single Qubit Error Correction

Part II - The Density Operator Approach

4. Open Quantum Circuits
   - Density Operators
   - Multipartite Systems
   - State Evolution
   - Single-Qubit Quantum Channels
   - Measurements
5. Information Theory
   - Classical Information
   - Quantum Information
6. NISQ Algorithms
   - Quantum Simulation
   - Hybrid Quantum-Classical Algorithms
   - The Variational Quantum Eigensolver
   - Introduction to the QAOA
   - Adiabatic Quantum Optimization

Part III - Application to Portfolio Optimization

7. Elements of Optimization Theory
   - Continuous Optimization
   - Dual Optimization Problems
   - Integer Programs
8. Classical Portfolio Optimization Theory
   - Mathematical Description
   - The Mean-Variance Model
   - Portfolio Optimization as a Quadratic Program
9. Quantum Portfolio Optimization
   - Portfolio Optimization via the VQE
   - Portfolio Optimization via the QAOA

Part IV - Quantum Machine Learning

10. Quantum Machine Learning
    - Machine Learning
    - Quantum Neural Networks
    - Quantum Kernels

1 Quantum Circuits

Quantum circuits are similar to classical circuits; they utilize qubits instead of electric currents, quantum gates in place of logic gates, and conclude with a measurement to obtain classical data.

1.1 Classical Bits

A *bit* $b \in \{0, 1\}$. All information, regardless of its form—be it discrete data like numbers and letters, or continuous media like audio and video—possesses a unique representation as a finite string of binary digits. A sequence of zeros and ones is called a *bit string*. For example

$$39 \longleftrightarrow 100111$$

$$39 = 2^5 b_1 + 2^4 b_2 + 2^3 b_3 + 2^2 b_4 + 2^1 b_5 + 2^0 b_6 = 2^5 1 + 2^4 0 + 2^3 0 + 2^2 1 + 2^1 1 + 2^0 1$$

In general

$$N = 2^{n-1} b_1 + 2^{n-2} b_2 + \ldots + 2^0 b_n \longleftrightarrow b_1\, b_2\, \ldots\, b_n$$

1.2 Single Qubits

A *single qubit*, or 1 *qubit*, is a two-level quantum system described by a two-dimensional complex unit vector

$$|\psi\rangle = a\,|\varphi_1\rangle + b\,|\varphi_2\rangle$$

We will use the notation

$$|q\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle = \sum_{i=0}^{1} \alpha_i\,|i\rangle$$

By the postulates of quantum mechanics $|\alpha_0|^2 + |\alpha_1|^2 = 1$ and we assume that $\langle i|j\rangle = \delta_{ij}$. $\{|0\rangle, |1\rangle\}$ is known as the *computational basis*. There are other basis; for example, the *Hadamard basis* $\{|+\rangle, |-\rangle\}$

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \qquad |-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

With inverse relations

$$|0\rangle = \frac{1}{\sqrt{2}}|+\rangle + \frac{1}{\sqrt{2}}|-\rangle \qquad |1\rangle = \frac{1}{\sqrt{2}}|+\rangle - \frac{1}{\sqrt{2}}|-\rangle$$

We can write the single qubit as

$$|q(\alpha_0, \alpha_1)\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$$

But we can also use two real parameters $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi)$ to describe it, where

$$|\alpha_0| = \cos(\theta/2) \qquad |\alpha_1| = \sin(\theta/2)$$

such that

$$|q(\theta, \phi)\rangle = \cos(\theta/2)|0\rangle + e^{i\phi}\sin(\theta/2)|1\rangle$$

Note that

$$|q(0, \phi)\rangle = |0\rangle \qquad |q(\pi, \phi)\rangle = |1\rangle$$

1.3 Multiple Qubits

Single qubit

$$|q_1\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle = \sum_{i=0,1} \alpha_i|i\rangle$$

2 qubit

$$|q_2\rangle = \alpha_{00}|0\,0\rangle + \alpha_{01}|0\,1\rangle + \alpha_{10}|1\,0\rangle + \alpha_{11}|1\,1\rangle = \sum_{i,j} \alpha_{ij}|ij\rangle$$

3 qubit

$$|q_3\rangle = \sum_{i,j,k} \alpha_{ijk}|i\,j\,k\rangle$$

But, what are $|i\,j\rangle$ and $|i\,j\,k\rangle$? In general, an *n qubit* is given by

$$|q_n\rangle \equiv |Q\rangle = \sum_{i_1,\ldots,i_n} \alpha_{i_1\ldots i_n}|i_1\ldots i_n\rangle$$

1.4 Classical (Logic) Gates

The *NOT gate*

$$\mathrm{NOT}\colon \{0,1\} \to \{0,1\} \qquad b \mapsto \mathrm{NOT}(b) = \bar{b}$$

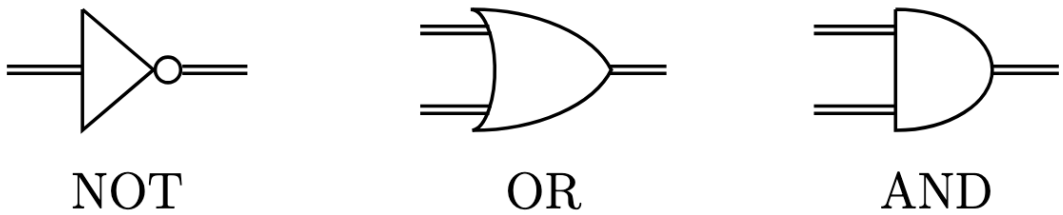
The *OR gate*

$$\mathrm{OR}\colon \{0,1\}^2 \to \{0,1\} \qquad b_1 b_2 \mapsto \mathrm{OR}(b_1 b_2)$$

The *AND gate*

$$\mathrm{AND}\colon \{0,1\}^2 \to \{0,1\} \qquad b_1 b_2 \mapsto \mathrm{AND}(b_1 b_2)$$

They form a *universal set of (classical) (logic) gates*.



1.5 Single-Qubit Gates

A *(quantum logic) gate* or *unitary* is a unitary transformation, $U^{-1} = U^{\dagger}$.
On a single qubit we write $|q\rangle \overset{U}{\longmapsto} U|q\rangle$

$$|q\rangle \text{ ——} \boxed{U} \text{——} U\,|q\rangle$$

Column vector notation,

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

For a single qubit

$$|q\rangle = \alpha_0 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \alpha_1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix}$$

A *single-qubit gate* will be represented by a 2×2 matrix

$$U = \begin{bmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{bmatrix}$$

and then

$$U|q\rangle = \begin{bmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix} = \begin{bmatrix} U_{00}\alpha_0 + U_{01}\alpha_1 \\ U_{10}\alpha_0 + U_{11}\alpha_1 \end{bmatrix}$$

In index notation

$$U|q\rangle = \sum_{i,j} \alpha_j U_{ij}|i\rangle$$

From here we see that

$$U = \sum_{i,j} U_{ij}|i\rangle\langle j|$$

that gives

$$U_{ij} = \langle i|U|j\rangle$$

For two consecutive single-qubit gates

$$|q\rangle \text{ —} \boxed{U_1} \text{—} \boxed{U_2} \text{—} U_2(U_1|q\rangle)$$

we have

$$|q\rangle \overset{U_1}{\longmapsto} U_1|q\rangle \overset{U_2}{\longmapsto} U_2U_1|q\rangle = \sum_{i,j,k} \alpha_j U_{2,ik} U_{1,kj} |i\rangle$$

The *Pauli matrices*

$$\sigma_X = X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad \sigma_Y = Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \qquad \sigma_Z = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

They are Hermitian, $\sigma_a^\dagger = \sigma_a$, where $a = X, Y, Z$. We will use

$$\sigma_A \quad \textit{for} \quad A = I, X, Y, Z$$

It is easy to show that

$$X|0\rangle = |1\rangle \qquad X|1\rangle = |0\rangle$$

Since

$$X|i\rangle = |\bar{i}\rangle = |1 - i\rangle$$

and for the classical NOT gate

$$\mathrm{NOT}(b) = \bar{b} = 1 - b$$

the X operator is also called the *bit-flit gate* and is usually denoted NOT. For the other two Pauli gates

$$Y|0\rangle = i|1\rangle \qquad Y|1\rangle = -i|0\rangle$$

$$Z|0\rangle = |0\rangle \qquad Z|1\rangle = -|1\rangle$$

The *Hadamard gate* is defined as follows

$$H|0\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle = |+\rangle \qquad H|1\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle = |-\rangle$$

From here it follows that

$$H|+\rangle = |0\rangle \qquad H|-\rangle = |1\rangle$$

and $H^2 = 1$. In the computational basis

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

The *relative phase gate*

$$P(\phi)|0\rangle = |0\rangle \qquad P(\phi)|1\rangle = e^{i\phi}|1\rangle$$

In matrix form,

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

When $\phi = \pi/2$, we call it the *S gate*

$$S|0\rangle = |0\rangle \qquad S|1\rangle = e^{i\pi/2}|1\rangle = i|1\rangle$$

and when $\phi = \pi/4$, it is *T gate*

$$T|0\rangle = |0\rangle \qquad T|1\rangle = e^{i\pi/4}|1\rangle = \frac{1}{\sqrt{2}}(1+i)|1\rangle$$

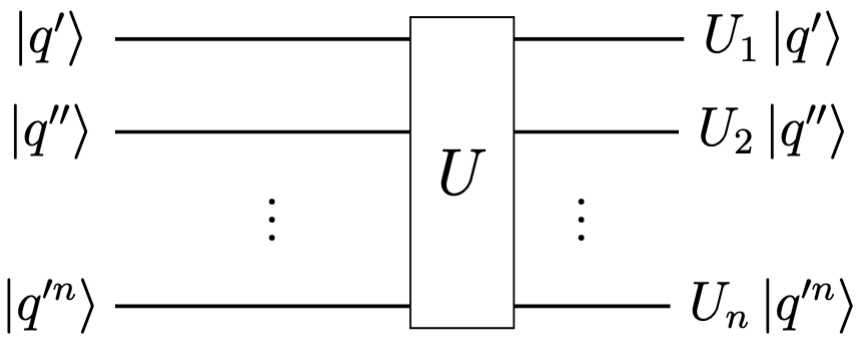
A rotation of θ_a radians about the a axis is given by

$$R_a(\theta_a) = e^{-i\sigma_a\theta_a/2} = \cos(\theta_a/2)I - i\sin(\theta_a/2)\sigma_a$$

with $\theta_a \in [0, 2\pi)$. In particular

$$R_x(\alpha) = e^{-iX\alpha/2} \quad R_y(\beta) = e^{-iY\beta/2} \quad R_z(\gamma) = e^{-iZ\gamma/2}$$

1.6 Multiple Single-Qubit Gates



$$U|q_n\rangle = U_1 \dots U_n\big(|q'\rangle \dots |q'^n\rangle\big) = U_1|q'\rangle \dots U_n|q'^n\rangle$$

For example, if $H_i = H$ for every $i = 1, \dots, n$

$$H \dots H\big(|q'\rangle \dots |q'^n\rangle\big) = H|q'\rangle \dots H|q'^n\rangle$$

Different notations for the action of the Hadamard gate

$$H\,|i\rangle = |(-1)^i\rangle$$

Also

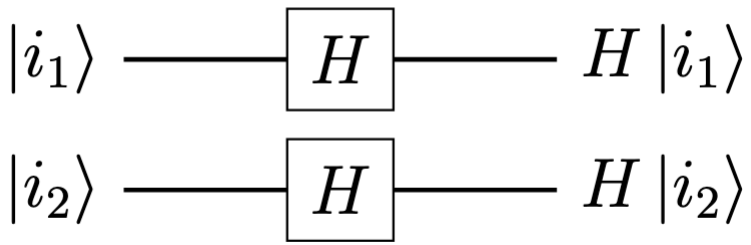
$$H|i\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}(-1)^i|1\rangle = \frac{1}{\sqrt{2}}\sum_j (-1)^{ij}|j\rangle$$

Using $e^{i\pi} = -1$,

$$H|k\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}e^{i\pi k}|1\rangle = \frac{1}{\sqrt{2}}\sum_j e^{i\pi kj}|j\rangle$$

Different expressions for the action of a Hadamard gate on a general single qubit

$$H|q\rangle = \sum_i \alpha_i|(-1)^i\rangle = \frac{1}{\sqrt{2}}\sum_{i,j}(-1)^{ij}\alpha_i|j\rangle = \frac{1}{\sqrt{2}}\sum_{k,j} e^{i\pi kj}\alpha_k|j\rangle$$



$$H|i_1\rangle H|i_2\rangle = \frac{1}{\sqrt{2^2}}\sum_{j_1,j_2}(-1)^{i_1j_1+i_2j_2}|j_1\rangle|j_2\rangle$$

We can write $H|i_1\rangle H|i_2\rangle = H^{\otimes 2}|i_1\ i_2\rangle$ and use the notation $|x\rangle = |i_1\ \ i_2\rangle$ to obtain

$$H^{\otimes 2}|x\rangle = \frac{1}{\sqrt{2^2}}\sum_y (-1)^{x\cdot y}|y\rangle$$

Note that $x \cdot y = i_1 j_1 + i_2 j_2$ and

$$\sum_y = \sum_{j_1, j_2}$$

For n Hadamard gates acting independently on n single qubits

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}}\sum_y (-1)^{x\cdot y}|y\rangle$$

where $x = i_1 \ldots i_n$, $y = j_1 \ldots j_n$ and $x \cdot y = i_1 j_1 + \ldots + i_n j_n$.

1.7 Multi-Qubit Gates

In this case $|Q\rangle \mapsto U|Q\rangle$

$$|Q\rangle \text{ —— } \boxed{U} \text{ —— } U\,|Q\rangle$$

where

$$|Q\rangle = \sum_{i_1,\ldots,i_n} \alpha_{i_1 \ldots i_n} |i_1 \ldots i_n\rangle$$

is a vector in a complex 2^n-dimensional vector space. Consider the case of a 2 qubit, i.e., $n = 2$,

$$|q_2\rangle = \sum_{i,j} \alpha_{ij} |i\ j\rangle$$

The four computational basis vectors are

$$|0\,0\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad |0\,1\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad |1\,0\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad |1\,1\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Every unitary on $|q_2\rangle$ will be given by

$$U = \begin{bmatrix} U_{11} & U_{12} & U_{13} & U_{14} \\ U_{21} & U_{22} & U_{23} & U_{24} \\ U_{31} & U_{32} & U_{33} & U_{34} \\ U_{41} & U_{42} & U_{43} & U_{44} \end{bmatrix}$$

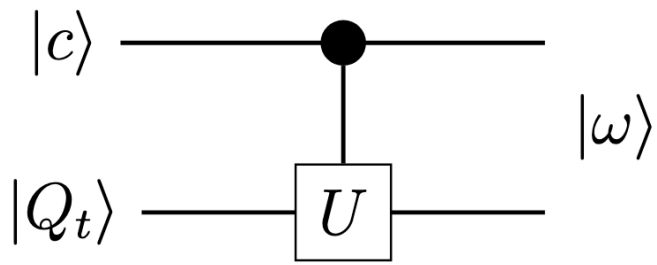
with $U_{rs} = U_{sr}^*$.

A *controlled gate* is a two-qubit operator that acts on a target qubit conditioned on the state of a control qubit. For $c \in \{0, 1\}$

$$|c\rangle|Q_t\rangle \longmapsto |c\rangle U(c)|Q_t\rangle$$

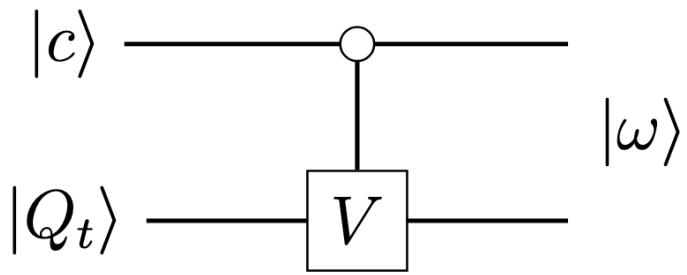
The *controlled-U gate* is defined as

$$\mathrm{C}U|0\rangle|Q_t\rangle = |0\rangle|Q_t\rangle \qquad \mathrm{C}U|1\rangle|Q_t\rangle = |1\rangle U|Q_t\rangle$$



The *controlled-V gate* is similar, but

$$\mathrm{C}V|0\rangle|Q_t\rangle = |0\rangle V|Q_t\rangle \qquad \mathrm{C}V|1\rangle|Q_t\rangle = |1\rangle|Q_t\rangle$$



We can rewrite

$$\mathrm{C}U|i\rangle|Q_t\rangle = |i\rangle U^i|Q_t\rangle$$

with

$$U^i = \begin{cases} 1 \text{ if } i = 0 \\ U \text{ if } i = 1 \end{cases}$$

For $|c\rangle = \sum_i c_i |i\rangle$ and $|t\rangle = \sum_j t_j |j\rangle$

$$\mathrm{C}U(|c\rangle|t\rangle) = \sum_{i,j} c_i t_j |i\rangle U^i |j\rangle$$

In matrix form

$$\mathrm{C}U = \begin{bmatrix} I & 0 \\ 0 & U \end{bmatrix}$$

The *CNOT gate* is defined as follows

$$|0\,0\rangle \xmapsto{\text{CNOT}} |0\,0\rangle \qquad |0\,1\rangle \xmapsto{\text{CNOT}} |0\,1\rangle$$

$$|1\,0\rangle \xmapsto{\text{CNOT}} |1\,1\rangle \qquad |1\,1\rangle \xmapsto{\text{CNOT}} |1\,0\rangle$$

In a more compact notation

$$CX|0\rangle|t\rangle = |0\rangle|t\rangle \quad CX|1\rangle|t\rangle = |1\rangle X|t\rangle$$

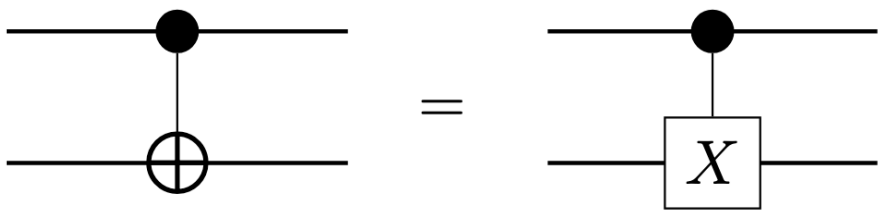
Alternatively

$$|i\rangle|j\rangle \xmapsto{\text{CNOT}} |i\rangle|j \oplus i\rangle$$

where $i \oplus j = (i + j) \bmod 2$. In the computational basis

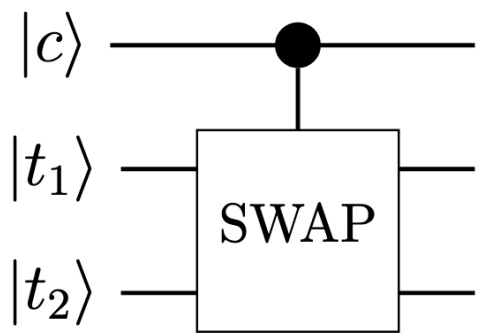
$$\text{CNOT} = \text{C}X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The gate is usually represented as follows



The CNOT gate is usually employed to create entanglement and build other unitary transformations.

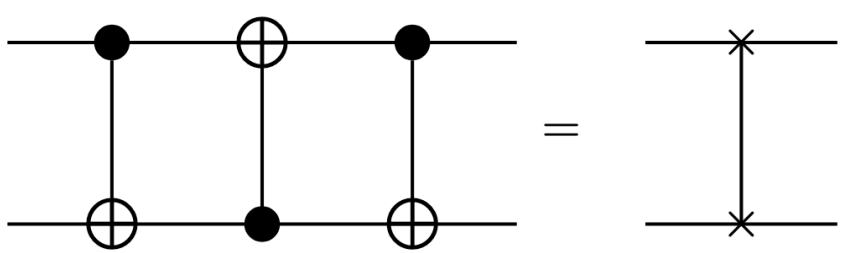
The *controlled-SWAP gate* (*CSWAP*) is another useful gate



where the *SWAP* gate interchanges the two target qubits

$$|t_1\rangle|t_2\rangle \xmapsto{\text{SWAP}} |t_2\rangle|t_1\rangle$$

It is easy to show that

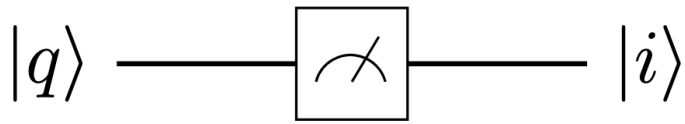


1.8 Measurements

For a single qubit $|q\rangle = \sum_i \alpha_i |i\rangle$, the probability of measuring the state $|i\rangle$ is $|\alpha_i|^2$

$$P(|i\rangle) = |\langle i|q\rangle|^2 = \left|\langle i| \sum_j \alpha_j |j\rangle\right|^2 = \left|\sum_j \alpha_j \langle i|j\rangle\right|^2 = \left|\sum_j \alpha_j \delta_{ij}\right|^2 = |\alpha_i|^2$$

It is represented by



For a 2 qubit $|q_2\rangle = \sum_{i,j} \alpha_{ij} |ij\rangle$

$$P(|ij\rangle) = |\alpha_{ij}|^2$$

and

$$P(|i\,\cdot\rangle) = P(|i\,0\rangle) + P(|i\,1\rangle) = |\alpha_{i0}|^2 + |\alpha_{i1}|^2 = \sum_j |\alpha_{ij}|^2$$

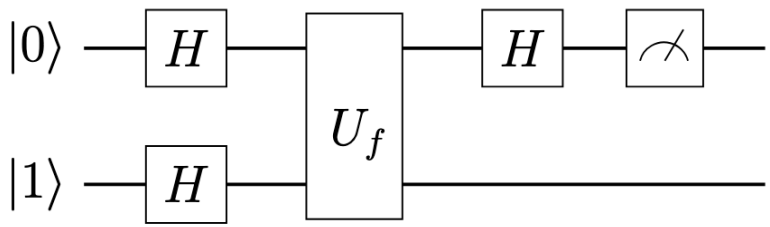
2 Quantum Algorithms

A quantum algorithm is a sequence of instructions that solves a specific problem more efficiently than the best known classical algorithms.

2.1 Deutsch's Algorithm

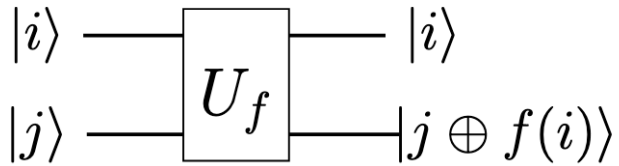
A Boolean function $f : \{0, 1\} \rightarrow \{0, 1\}$ is *constant* if $f(0) = f(1)$, otherwise it is *balanced*, $f(0) \neq f(1)$. To determined if the function is constant or balanced, it has to be called two time.

Deutsch discovered is that we can find out whether the function is constant or balanced by calling the function only once

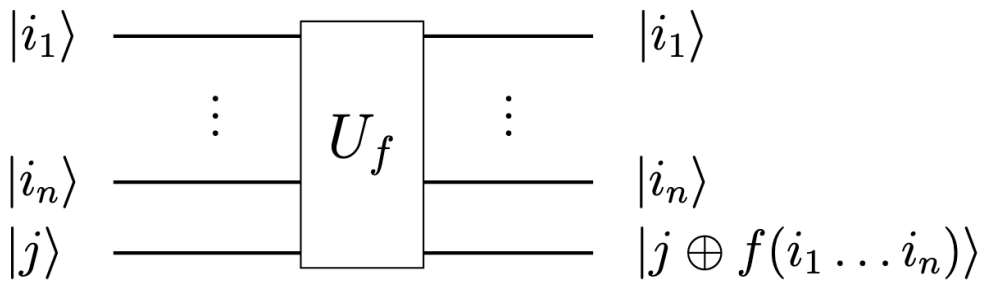


If the measurement gives 0 (1), the function is constant (balanced).

U_f it the *XOR oracle*



The *Deutsch-Jozsa algorithm* generalizes this to the Boolean function $f : \{0,1\}^n \to \{0,1\}$. It is *constant* if it takes the same value (0 or 1) for every $x \in \{0,1\}^n$ and it is *balanced* if half of the time it is 0 and the other half it is 1. The oracle is



All n qubits in the first register are measured. If the result is the string $00\ldots0$, the function is constant. Otherwise, the function is balanced.

2.2 The Quantum Fourier Transform

The *Quantum Fourier Transform* (*QFT*) is the "engine" behind almost all quantum algorithms that provide an exponential speedup

- Shor's algorithm (for breaking current encryption methods)
- Quantum Phase Estimation (used in the HHL algorithm for solving linear systems)
- Quantum Amplitude Estimation (fundamental for Monte Carlo simulations)

Requires fault-tolerant hardware for exponential speedup.

In decimal notation

$$|Q\rangle = \sum_{i_1,\ldots,i_n} \alpha_{i_1\ldots i_n} |i_1 \ \ldots \ i_n\rangle = \sum_{x=0}^{N-1} \alpha_x |x\rangle$$

where $N = 2^n$.

The *Fourier basis*

$$|x\rangle_{\mathrm{QFT}} = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{\frac{2\pi i}{N} xy} |y\rangle$$

$|x\rangle_{\mathrm{QFT}}$ is the *quantum Fourier transformed* of $|x\rangle$

$$|x\rangle_{\mathrm{QFT}} = \mathrm{QFT}_n |x\rangle$$

It can easily be shown that

$$\mathrm{QFT}_1 |i\rangle = H|i\rangle = |(-1)^i\rangle$$

and

$$\mathrm{QFT}_2 |j\, k\rangle = \mathrm{QFT}_2 |x = 2j + k\rangle = \frac{1}{2} \sum_{y=0}^{3} e^{\frac{\pi i}{2} xy} |y\rangle$$

The inverse QFT takes us from the Fourier basis to the computational basis

$$\mathrm{QFT}_n^{\dagger} |x\rangle_{\mathrm{QFT}} = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{-\frac{2\pi i}{N} xy} |y\rangle$$

2.3 Shor's Factoring Algorithm

It is not used to build financial models.

2.4 Grover's Search Algorithm

It provides a quadratic speedup $O(\sqrt{N})$ for searching unsorted databases.

- Considered less "revolutionary" than the exponential speedup offered by QFT-based algorithms.
- Enables a quadratic speedup in Monte Carlo sampling via Quantum Amplitude Estimation (QAE), which in turn utilizes the Inverse QFT for phase readout.
- Essential for calculating Value at Risk (VaR), Conditional Value at Risk (CVaR), and Option Pricing.

3 Quantum Error Correction

Quantum circuits are fragile systems and other qubits are needed to detect and correct their uncontrolled changes. That's the key message behind quantum error correction.

3.1 Entanglement with the Environment

The interaction of a qubit with its environment

$$|Q_0\rangle|e_0\rangle \overset{U}{\longmapsto} \sum |Q_t\rangle|e_t\rangle$$

For example, for a single qubit

$$U(|0\rangle|e\rangle) = |0\rangle|e_{00}\rangle + |1\rangle|e_{01}\rangle$$
$$U(|1\rangle|e\rangle) = |0\rangle|e_{10}\rangle + |1\rangle|e_{11}\rangle$$

giving

$$U(|q\rangle|e\rangle) = U\big((\alpha_0|0\rangle + \alpha_1|1\rangle)|e\rangle\big) = \sum_{i,j} \alpha_i|j\rangle|e_{ij}\rangle$$

3.2 Classical Error Correction

Suppose a bit b is sent through a noisy channel and $p \ll 1$ is the probability that the bit is flipped to $\bar{b}$

$$P(b) = (1-p) \qquad P(\bar{b}) = p$$

We assume $P(b) \gg P(\bar{b})$.

The *classical repetition code* prescribes that instead of sending a single bit b, one sends multiple redundant copies. For instance, bbb

$$P(3b, 0\bar{b}) = (1-p)^3 \qquad P(0b, 3\bar{b}) = p^3$$

Moreover

$$P(3b, 0\bar{b}) > P(2b, 1\bar{b}) > P(1b, 2\bar{b}) > P(0b, 3\bar{b})$$

To determine which qubit has flipped, we can perform a *parity check*. The whole process can be summarized as follows,

$$b \xrightarrow{encode} b\,b\,b \xrightarrow{send\ \ (error\ \ occurs)} b\,\bar{b}\,b \xrightarrow{detect} \xrightarrow{correct} b\,b\,b \xrightarrow{decode} b$$

3.3 Generalities on QEC Codes

1. The qubit prone to error is augmented with a set of *ancilla qubits* to

$$|Q\rangle \longrightarrow |Q\rangle|0\ldots0\rangle = |Q\rangle_{anc}$$

2. The information contained in the original qubit $|Q\rangle$ is then encoded into the extended state $|Q\rangle_{anc}$, providing the necessary redundancy for error detection and correction

$$|Q\rangle_{anc} \longrightarrow U_{enc}|Q\rangle_{anc} = |Q\rangle_{enc}$$

3. The error occurs

$$|Q\rangle_{enc} \longrightarrow E|Q\rangle_{enc} = |Q\rangle_{E}$$

4. A quantum circuit R is designed to detect and correct the error

$$|Q\rangle_{E} \longrightarrow R|Q\rangle_{E} = |Q\rangle_{R}$$

5 The encoded qubit is decoded

$$|Q\rangle_R \longrightarrow U_{dec}|Q\rangle_R = |Q\rangle_{anc}$$

where $U_{dec} = U_{enc}^{-1}$.

6 Finally, we decouple the ancillary qubits and recover the original state vector

$$|Q\rangle_{anc} \longrightarrow |Q\rangle$$

The no-cloning theorem asserts that it is impossible to create an identical copy of an arbitrary, unknown quantum state

$$|Q\rangle\,|0\rangle \nrightarrow |Q\rangle\,|Q\rangle$$

However, it is possible to create an arbitrary number of identical copies of the computational basis states

$$|0\rangle \longrightarrow |0\rangle \ldots |0\rangle \qquad |1\rangle \longrightarrow |1\rangle \ldots |1\rangle$$

3.4 Single Qubit Error Correction

The *bit flip error*

$$|i\rangle \longrightarrow |\bar{i}\rangle$$

According to the no-cloning theorem we cannot

$$|q\rangle \longrightarrow |q\rangle\,|q\rangle$$

but we can

$$|i\rangle \longrightarrow |i\rangle \ldots |i\rangle$$

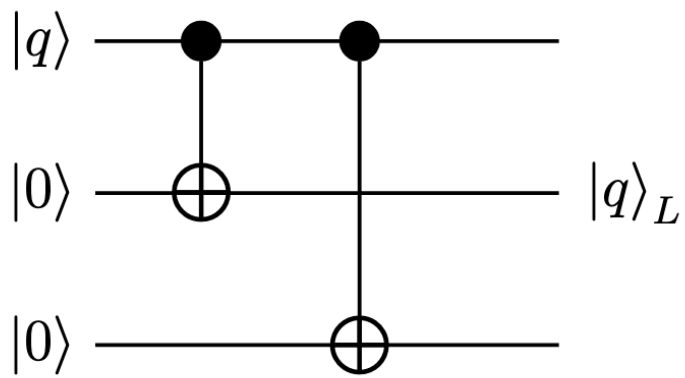
The *bit-flip quantum repetition code* is analogous to the classical repetition code

$$|q\rangle \longrightarrow |q\rangle_{anc} = |q\rangle\,|0\rangle\,|0\rangle = \alpha_0\,|000\rangle + \alpha_1\,|100\rangle$$

The *logical qubit* is

$$|q\rangle_L = \alpha_0\,|000\rangle + \alpha_1\,|111\rangle$$

The logical qubit can be created using the following circuit



$$|q\rangle_L = \sum_{i=0}^{1} \alpha_i |i\rangle|i\rangle|i\rangle$$

In summary

$$|q\rangle = \sum_{i=0}^{1} \alpha_i \, |i\rangle \xrightarrow{encode} |q\rangle_L = \sum_{i=0}^{1} \alpha_i \, |i \; i \; i\rangle \xrightarrow{send \;\; (error \;\; occurs)} \sum_{i=0}^{1} \alpha_i \, |i \; \bar{i} \; i\rangle$$

$$\xrightarrow{parity \;\; check} \xrightarrow{correct} |q\rangle_L = \sum_{i=0}^{1} \alpha_i \, |i \; i \; i\rangle \xrightarrow{decode} \sum_{i=0}^{1} \alpha_i \, |i\rangle = |q\rangle$$

The *phase flip error*

$$|q\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle \longrightarrow \alpha_0 |0\rangle - \alpha_1 |1\rangle$$

Any arbitrary single-qubit error can be treated as a linear combination of these two basic errors.

4 Open Quantum Circuits

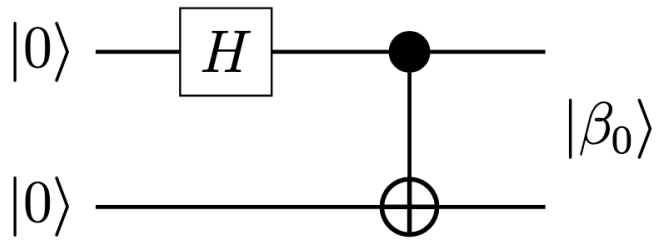
For entangled multipartite systems, the state vector is insufficient; the density operator is essential to fully characterize individual quantum interconnected systems.

4.1 Density Operators

Consider the 2 qubit (a *Bell state*)

$$|\beta_0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle|0\rangle + |1\rangle|1\rangle\right) = \frac{1}{\sqrt{2}}\sum_i |i\rangle|i\rangle$$

It is impossible to write it as $|q'\rangle\,|q''\rangle$. It is an *entangled state*.



A single qubit entangled with its environment

$$|\Psi\rangle = \sum_{i,j} \alpha_{ij} |i\rangle |e_i\rangle \neq |q\rangle |e\rangle$$

Consider a quantum system described by a statistical ensemble of identical copies. It is characterized by a probability distribution $\{|O_s\rangle, p_s\}$.

This contrasts with the state vector formalism, where the system is described by the superposition $\sum \alpha_{s'} |O_{s'}\rangle$.

In the *density operator formalism*, the quantum state is described by a *density operator* $\rho_Q : \mathcal{H}_Q \to \mathcal{H}_Q$ given by

$$\rho = \sum_s p_s |O_s\rangle\langle O_s|$$

Some properties of ρ:

$$\rho |Q\rangle = \sum_{s,s'} p_s \alpha_{s'} |O_s\rangle\langle O_s|Q_{s'}\rangle$$

When $|Q\rangle = |O_{s'}\rangle$

$$\rho|O_{s'}\rangle = \sum_s p_s|O_s\rangle\,\delta_{ss'} = p_{s'}|O_{s'}\rangle$$

It follows that the density operator has *unit trace*

$$\mathrm{Tr}\,\rho = \sum_{s'}\langle O_{s'}|\rho|O_{s'}\rangle = \sum_{s'} p_{s'} = 1$$

Moreover, among some other properties, it is *Hermitian*

$$\rho^\dagger = \sum_s p_s^*\left(|O_s\rangle\langle O_s|\right)^\dagger = \sum_s p_s|O_s\rangle\langle O_s| = \rho$$

Summary: In the density operator formalism, the operator ρ represents the state of the system, just as $|Q\rangle$ represents the state in the state vector formalism. We transition from describing the system as a state vector to describing it as a density operator.

For a single qubit

$$\rho_q = \sum_{s=1}^{2} p_s |O_s\rangle\langle O_s|$$

where $\{|O_1\rangle, |O_2\rangle\}$ is a basis for $\mathcal{H}_O$ (not necessarily orthogonal). The matrix elements are then given by

$$[\rho_q]_{s's''} = \langle O_{s'}|\rho_q|O_{s''}\rangle = \sum_s p_s \langle O_{s'}|O_s\rangle\langle O_s|O_{s''}\rangle$$

For example, if the single-qubit description and the observables are expressed in the computational basis

$$\rho_q = p_0|0\rangle\langle 0| + (1-p_0)|1\rangle\langle 1| = \begin{bmatrix} p_0 & 0 \\ 0 & 1-p_0 \end{bmatrix}$$

The off-diagonal elements carry the *quantum coherence* information.

Since any 2×2 complex matrix can be written as a linear combination of the Pauli matrices and the Identity

$$\rho_q = c_I I + \sum c_a \sigma_a$$

Choosing $c_I = 1/2$ and $c_a = B_a/2$

$$\rho_q = \frac{1}{2} I + \frac{1}{2} \sum_a B_a \sigma_a = \frac{1}{2}(I + \mathbf{B} \cdot \boldsymbol{\sigma})$$

$\mathbf{B} = [B_x \; B_y \; B_z]^T$ is the *Bloch vector*.

The matrix ρ_q has eigenvalues $(1 \pm \|\mathbf{B}\|)/2$. Since the eigenvalues of any density operator are always equal to or greater than zero (*positive semi-definite*), it follows that $\|\mathbf{B}\| \leq 1$.

Points on the Bloch sphere, $\|\mathbf{B}\| = 1$, are *pure states*, e.g.,

$$\rho_q(\mathbf{B} = [0 \; 0 \; 1]^T) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \implies [0 \; 0 \; 1]^T \leftrightarrow |0\rangle$$

Points in the Bloch ball, $\|\mathbf{B}\| < 1$, are *mixed states*, e.g.,

$$\rho_q(\mathbf{B} = [0 \; 0 \; 0]^T) = \frac{1}{2} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

4.2 Multipartite Systems

A *bipartite system* is a composite quantum system consisting of two physical subsystems. For example, a 2 qubit

$$|q_2\rangle = \sum_{i,i'} \alpha_{ii'} |i\ i'\rangle$$

The density operator of this pure state is given by

$$\rho_{q_2} = \rho_{qq'} = |q_2\rangle\langle q_2| = \Big(\sum_{i,i'} \alpha_{ii'} |i\ i'\rangle\Big)\Big(\sum_{j,j'} \alpha^*_{jj'} \langle j\ j'|\Big)$$

$$= \sum_{i,j,i',j'} \alpha_{ii'}\alpha^*_{jj'} |i\ i'\rangle\langle j\ j'|$$

Using

$$[\rho_{qq'}]_{ii',jj'} \equiv \alpha_{ii'}\alpha^*_{jj'} = \langle i\ i'|\rho_{qq'}|j\ j'\rangle$$

we arrive at

$$\rho_{qq'} = \sum_{i,j,i',j'} [\rho_{qq'}]_{ii',jj'} |i\ i'\rangle\langle j\ j'|$$

Or, in its full matrix form

$$\rho_{qq'} = \begin{bmatrix} \langle 0\ 0|\rho_{qq'}|0\ 0\rangle & \langle 0\ 0|\rho_{qq'}|0\ 1\rangle & \langle 0\ 0|\rho_{qq'}|1\ 0\rangle & \langle 0\ 0|\rho_{qq'}|1\ 1\rangle \\ \langle 0\ 1|\rho_{qq'}|0\ 0\rangle & \langle 0\ 1|\rho_{qq'}|0\ 1\rangle & \langle 0\ 1|\rho_{qq'}|1\ 0\rangle & \langle 0\ 1|\rho_{qq'}|1\ 1\rangle \\ \langle 1\ 0|\rho_{qq'}|0\ 0\rangle & \langle 1\ 0|\rho_{qq'}|0\ 1\rangle & \langle 1\ 0|\rho_{qq'}|1\ 0\rangle & \langle 1\ 0|\rho_{qq'}|1\ 1\rangle \\ \langle 1\ 1|\rho_{qq'}|0\ 0\rangle & \langle 1\ 1|\rho_{qq'}|0\ 1\rangle & \langle 1\ 1|\rho_{qq'}|1\ 0\rangle & \langle 1\ 1|\rho_{qq'}|1\ 1\rangle \end{bmatrix}$$

One of the qubits can then be described by a *reduced density operator*, obtained via the *partial trace* (an operation without a direct analog in the state vector formalism)

$$\rho_q = \mathrm{Tr}_{q'} \rho_{qq'} = \sum_{i'} \langle \cdot\ i'|\rho_{qq'}|\cdot\ i'\rangle$$

Similarly, for the other qubit

$$\rho_{q'} = \mathrm{Tr}_q \rho_{qq'} = \sum_i \langle\, i\, \cdot |\rho_{qq'}|\, i\, \cdot \rangle$$

The dots indicate that we leave the Hilbert space of one qubit untouched while tracing out over the other.

Using this definition, the elements of the reduced density matrix of the first qubit is

$$[\rho_q]_{ij} = \langle\, i\, \cdot |\rho_q|\, j\, \cdot \rangle = \langle\, i\, \cdot | \sum_{i'} \langle \cdot\, i'|\rho_{qq'}|\, \cdot\, i'\rangle |\, j\, \cdot \rangle$$

$$= \sum_{i'} \langle i\ i'|\rho_{qq'}|j\ i'\rangle = \langle i\ 0|\rho_{qq'}|j\ 0\rangle + \langle i\ 1|\rho_{qq'}|j\ 1\rangle$$

that is,

$$\rho_q = \frac{1}{2}\begin{bmatrix} \langle 0\ 0|\rho_{qq'}|0\ 0\rangle + \langle 0\ 1|\rho_{qq'}|0\ 1\rangle & \langle 0\ 0|\rho_{qq'}|1\ 0\rangle + \langle 0\ 1|\rho_{qq'}|1\ 1\rangle \\ \langle 1\ 0|\rho_{qq'}|0\ 0\rangle + \langle 1\ 1|\rho_{qq'}|0\ 1\rangle & \langle 1\ 0|\rho_{qq'}|1\ 0\rangle + \langle 1\ 1|\rho_{qq'}|1\ 1\rangle \end{bmatrix}$$

As an example, consider again the Bell state $|\beta_0\rangle$

$$\begin{aligned}\rho_{\beta_0} &= |\beta_0\rangle\langle\beta_0| \\ &= \frac{1}{2}\big(|0\rangle|0\rangle\langle 0|\langle 0| + |0\rangle|0\rangle\langle 1|\langle 1| + |1\rangle|1\rangle\langle 0|\langle 0| + |1\rangle|1\rangle\langle 1|\langle 1|\big) \\ &= \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}\end{aligned}$$

It is straightforward to check that

$$\rho_q = \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Note that this a mixed state rather than a pure state; thus, it cannot be represented using the state vector formalism. Its Bloch vector is the null vector, $\mathbf{B} = \mathbf{0}$, located at the center of the Bloch ball.

4.3 State Evolution

In the state vector formalism

$$|Q(t_0)\rangle \longmapsto |Q(t)\rangle = U(t, t_0)|Q(t_0)\rangle$$

where $U(t, t_0)\colon \mathcal{H}_{t_0} \to \mathcal{H}_t$ is the *time evolution operator*. In the density operator formalism, the system is initially in a pure state

$$\rho(t_0) = |Q(t_0)\rangle\langle Q(t_0)|$$

and evolves into another pure state

$$\begin{aligned}\rho(t) = |Q(t)\rangle\langle Q(t)| &= U(t, t_0)|Q(t_0)\rangle\langle Q(t_0)|U^\dagger(t, t_0) \\ &= U(t, t_0)\rho(t_0)U^\dagger(t, t_0)\end{aligned}$$

The initial and final states are related by a *superoperator* (also called a *quantum map* or *quantum channel*)

$$\Phi(t, t_0)\colon \mathcal{D}(t_0) \to \mathcal{D}(t)\,, \qquad \rho(t_0) \mapsto \rho(t) = \Phi(t, t_0)\rho(t_0)$$

4.4 Single-Qubit Quantum Channels

Suppose an n qubit is in a general state $\rho_Q(t_0)$, initially non-entangled with an environment in a pure state $|e_0\rangle\langle e_0|$. The total system is thus initially in the state

$$\rho_{Qe}(t_0) = \rho_Q(t_0) \, |e_0\rangle\langle e_0|$$

At time $t > t_0$

$$\rho_Q(t) = \mathrm{Tr}_{e_t}\big(\rho_{Qe}(t)\big) = \mathrm{Tr}_{e_t}\big(U(t,t_0)\rho_Q(t_0) \, |e_0\rangle\langle e_0| U^\dagger(t,t_0)\big)$$

Using the standard notation $|e_t\rangle = |k\rangle_e$

$$\begin{aligned}\rho_Q(t) &= \sum_k {}_e\langle k|U(t,t_0)|e_0\rangle \rho_Q(t_0) \big({}_e\langle k|U(t,t_0)|e_0\rangle\big)^\dagger \\ &= \sum_k E_k \rho_Q(t_0) E_k^\dagger\end{aligned}$$

where the *Kraus operators* E_k are physically interpreted as the *error channels* or *noise operators*.

Error operators satisfy the *completeness relation*

$$\sum_k E_k^\dagger E_k = 1$$

We have thus the quantum channel $\Phi(t, t_0)$ that takes

$$\rho_Q(t_0) \longmapsto \rho_Q(t) = \Phi(t, t_0)\rho_Q(t_0) = \sum_k E_k \rho_Q(t_0) E_k^\dagger$$

For example, for a single qubit, the *bit-flip channel* $\Phi_X(t, t_0)$ transforms

$$\rho_{t_0} \longmapsto \rho_t = \Phi_X(t, t_0)\rho_0 = (1-p)\rho_0 + pX\rho_0 X$$

4.5 Measurements

In the state vector formalism

$$|Q\rangle = \sum_{s'} \alpha_{s'} |Q_{s'}\rangle \xrightarrow{M_s} P(|Q_s\rangle) = p_s = \left|\langle Q_s| \sum_{s'} \alpha_{s'} |Q_{s'}\rangle\right|^2 = |\alpha_s|^2$$

In the density operator formalism, the situation is different. Given a basis $\{|O_s\rangle\}$ for the Hilbert space $\mathcal{H}_O$ of measurement outcomes, a *measurement operator* M_s is a map $M_s : \mathcal{H}_Q \to \mathcal{H}_Q$ that acts on a state $|Q\rangle$ such that the resulting outcome $|O_s\rangle$ is given by

$$|O_s\rangle = \frac{M_s|Q\rangle}{\sqrt{p_s}}$$

These operators must satisfy the *completeness relation*

$$\sum_s M_s^\dagger M_s = I$$

The probability of measuring the outcome $|O_s\rangle$ is given by

$$p_s = \langle Q|M_s^\dagger M_s|Q\rangle$$

After a measurement M_s, the initial state ρ becomes

$$\rho_s = \frac{M_s \rho M_s^\dagger}{\mathrm{Tr}(M_s^\dagger M_s \rho)}$$

5 Information Theory

Since classical physics is a limiting case of quantum physics, classical information theory is a special case of quantum information theory.

5.1 Classical Information

Let $X = \{x_s, P(x_s) = p_s\}_{s=1}^{s=S}$, with $0 \leq p_s \leq 1$ and $\sum_s p_s = 1$. The *information content of a single event* (also called the *Shannon entropy of a single event*) is given by

$$h(x_s) = h_s = \log_2 \frac{1}{p_s} = -\log p_s$$

The more probable an event, the less information it contains. That is, the information content of an event is greater the less probable it is to occur.

The *Shannon entropy of the probability distribution* X is

$$H(X) = -\sum_s p_s \log p_s$$

For example, for the probability distribution $X = \{x, p\,; \bar{x}, 1-p\}$

$$H(X) = -p \log p - (1-p) \log(1-p)$$

The Shannon entropy satisfies

$$0 \leq H(X) \leq \log N$$

where N is the number of possible outcomes. The highest value of $H(X)$ is obtained when all possible outcomes are equally likely; that is, when $p_s = 1/N$ for every s.

Now, suppose two probability distributions

$$X = \{x_s, p_s\}_{s=1}^{s=S} \qquad Y = \{y_r, p_r\}_{r=1}^{r=R}$$

The *joint information content of of* $x_s y_r$ (*Shannon joint entropy*)

$$h_{s,r} = -\log p_{s,r}$$

where $p_{s,r}$ is the *joint probability*.

If the two systems are independent of each other $p_{s,r} = p_s p_r$ and

$$h_{s,r} = -\log p_{s,r} = -\log(p_s p_r) = h_s + h_r$$

However, in general, the two systems are correlated and $p_{s,r} \neq p_s p_r$.

The *Shannon marginal joint entropy* of X is defined as

$$H(X, y_r) = -\sum_s p_{s,r} \log p_{s,r}$$

The *Shannon joint entropy of* XY is

$$H(X, Y) = \sum_r H(X, y_r) = -\sum_{s,r} p_{s,r} \log p_{s,r}$$

Since $p_{s,r} = p_{r,s}$

$$H(X, Y) = H(Y, X)$$

Moreover

$$H(X, Y) \geq H(X) \qquad H(X, Y) \geq H(Y)$$

The *conditional information content of* $y_r|x_s$ (*Shannon conditional entropy*) is given by

$$h_{r|s} = -\log p_{r|s}$$

The *Shannon marginal conditional entropy* $H(Y|x_s)$ is

$$H(Y|x_s) = -\sum_r p_{r|s} \log p_{r|s}$$

The *Shannon conditional entropy* is the average information content of the events of Y having a complete knowledge of X

$$H(Y|X) = \sum_s p_s H(Y|x_s) = -\sum_{s,r} p_s p_{r|s} \log p_{r|s}$$

According to *Bayes' theorem*, $p_{r|s} = p_{r,s}/p_s$, then

$$H(Y|X) = -\sum_{s,r} p_{s,r} \log p_{r|s}$$

When X and Y are independent ($p_{s,r} = p_s p_r$)

$$H(Y|X) = -\sum_{s,r} p_s p_r \log p_r = -\sum_{r} p_r \log p_r = H(Y)$$

More generally

$$0 \leq H(Y|X) \leq H(Y)$$

It can be proved that

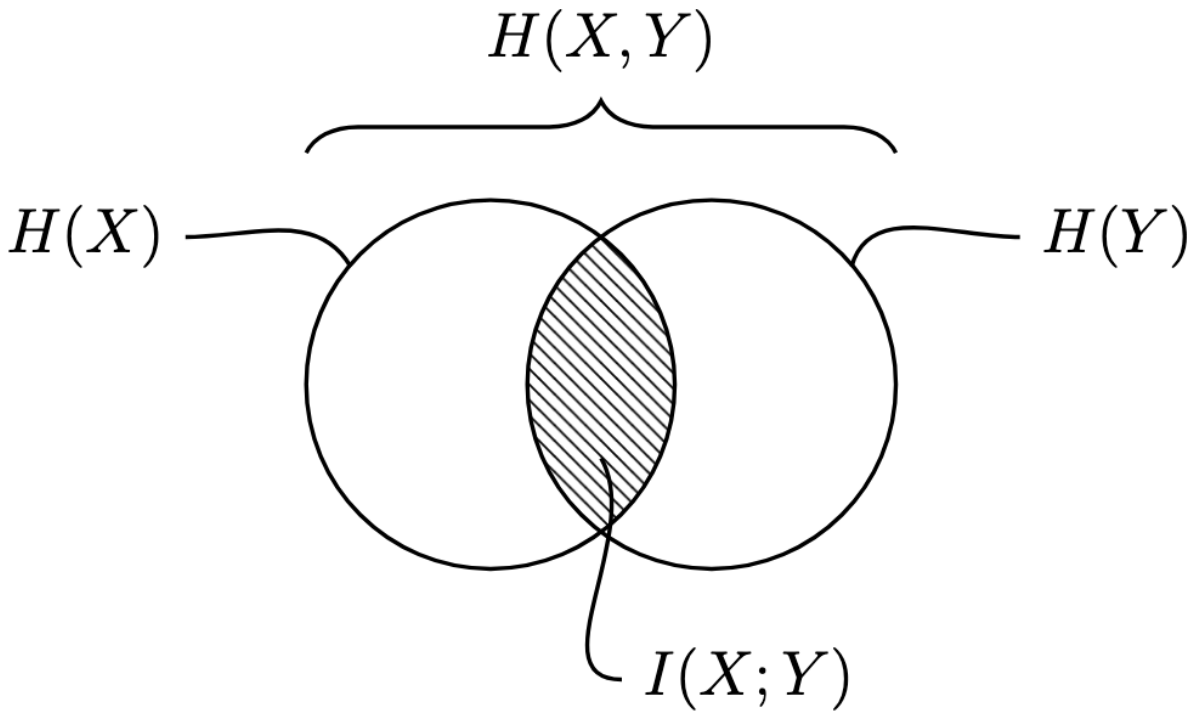
$$H(Y|X) \leq H(Y) \leq H(Y,X)$$

The *mutual information of XY*

$$I(X;Y) = \sum_{s,r} p_{s,r}(h_s - h_{s|r}) = -\sum_{s,r} p_{s,r} \log \frac{p_s p_r}{p_{s,r}} = I(Y;X)$$

It can be shown that

$$I(X;Y) = H(X) + H(Y) - H(X,Y) = H(X) - H(X|Y)$$



The *conditional mutual information* is defined as

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z)$$

It satisfies the *strong subadditivity relation*

$$I(X;Y|Z) \geq 0$$

Consider two apparently identical systems where $x_s = \tilde{x}_s$ for all s

$$X = \{x_s, p_s\} \qquad h(x_s) = -\log p_s \qquad H(X) = -\sum_s p_s \log p_s$$

$$\widetilde{X} = \{\tilde{x}_s, \tilde{p}_s\} \qquad h(\tilde{x}_s) = -\log \tilde{p}_s \qquad H(\widetilde{X}) = -\sum_s \tilde{p}_s \log \tilde{p}_s$$

Relative entropy between a pair of events $(x_s, \tilde{x}_s) \in (X, \widetilde{X})$

$$h(x_s \| \tilde{x}_s) = h(\tilde{x}_s) - h(x_s) = -\log \tilde{p}_s + \log p_s = -\log \frac{\tilde{p}_s}{p_s}$$

We define the *relative entropy between* X *and* $\widetilde{X}$ by

$$H(X \| \widetilde{X}) = -\sum_s p_s \log \frac{\tilde{p}_s}{p_s} = -\sum_s p_s (\log \tilde{p}_s - \log p_s)$$

Relative entropy is also known as *divergence* or *discrimination* (usually denoted with the letter D).

Since

$$H(X\|\widetilde{X}) \neq H(\widetilde{X}\|X)$$

it is more appropriate to say that $H(X\|\widetilde{X})$ is the entropy of X relative to $\widetilde{X}$, and $H(\widetilde{X}\|X)$ the entropy of $\widetilde{X}$ relative to X.

The relative entropy is a non-negative quantity

$$H(X\|\widetilde{X}) \geq 0$$

The equality holds iff $X = \widetilde{X}$.

We finally define the *joint relative entropy*

$$H(p_{s,r}\|\tilde{p}_{s,r}) = -\sum_{s,r} p_{s,r} \log \frac{\tilde{p}_{s,r}}{p_{s,r}}$$

and the *conditional relative entropy*

$$H(p_{s|r}\|\tilde{p}_{s|r}) = -\sum_{s,r} p_{s,r} \log \frac{\tilde{p}_{s|r}}{p_{s|r}}$$

5.2 Quantum Information

The *von Neumann entropy* of a quantum system Q is given by

$$S(\rho_Q) = -\mathrm{Tr}_Q\left[\rho_Q \log \rho_Q\right]$$

Since density matrices are Hermitian and Hermitian matrices are always diagonalizable, there is a basis of orthonormal eigenvectors $\{|e_s\rangle\}$ such that we can write

$$\rho_Q = \sum_s p_s |e_s\rangle\langle e_s|$$

This is called the *eigenvalue decomposition*. Substituting in the definition above

$$\begin{aligned} S(\rho_Q) &= -\mathrm{Tr}_Q\left[\rho_Q \log \rho_Q\right] \\ &= -\mathrm{Tr}_Q\Big[\sum_s p_s |e_s\rangle\langle e_s| \log \sum_{s'} p_{s'} |e_{s'}\rangle\langle e_{s'}|\Big] \\ &= -\mathrm{Tr}_Q\Big[\sum_{s,s'} p_s \log p_{s'} |e_s\rangle\langle e_s|e_{s'}\rangle\langle e_{s'}|\Big] \end{aligned}$$

$$= -\mathrm{Tr}_Q\Big[\sum_s p_s \log p_s |e_s\rangle\langle e_s|\Big]$$

$$= -\sum_s p_s \log p_s \mathrm{Tr}\Big[|e_s\rangle\langle e_s|\Big]$$

$$= -\sum_s p_s \log p_s \sum_s p_s$$

$$= -\sum_s p_s \log p_s$$

Thus, in quantum information theory, the von Neumann entropy $S(\rho_Q)$ is the direct quantum mechanical analogue of the classical Shannon entropy $H(X)$.

It is easy to show that the von Neumann entropy is independent of the orthonormal basis chosen for $\mathcal{H}_Q$. That is, given another orthonormal basis $\{|e'_s\rangle\}$ for $\mathcal{H}_Q$

$$S(\rho'_Q) = -\mathrm{Tr}_Q\big[\rho'_Q \log \rho'_Q\big] = -\mathrm{Tr}_Q\big[\rho_Q \log \rho_Q\big] = S(\rho_Q)$$

For example, the von Neumann entropy of a single-qubit state, when calculated using its eigenvalue decomposition, gives

$$
\begin{aligned}
S(\rho_q) &= -\mathrm{Tr}_q\left[\rho_q \log \rho_q\right] \\
&= -\mathrm{Tr}_q\left(\begin{bmatrix} p_0 & 0 \\ 0 & 1-p_0 \end{bmatrix}\begin{bmatrix} \log p_0 & 0 \\ 0 & \log(1-p_0) \end{bmatrix}\right) \\
&= -p_0 \log p_0 - (1-p_0)\log(1-p_0)
\end{aligned}
$$

We conclude that the von Neumann entropy of a mixed single-qubit state is equal to the Shannon entropy of a classical binary system.

They coincide because the two orthonormal eigenbasis states $\{|e_0\rangle, |e_1\rangle\}$ are perfectly distinguishable (no quantum coherence between them) much like the outcomes of a classical binary system, such as a bent coin.

Pure state (on the Bloch sphere): $S = 0$.

Maximally mixed state (center of the Bloch ball): $S = 1$.

Given a subsystem Q of QQ', we saw that $\rho_Q = \mathrm{Tr}_{Q'}\rho_{QQ'}$. We then define the *reduced von Neumann entropy* of Q

$$S(\rho_Q) = -\mathrm{Tr}_Q\left[\rho_Q \log \rho_Q\right] = -\mathrm{Tr}_Q\left[\mathrm{Tr}_{Q'}\rho_{QQ'} \log \mathrm{Tr}_{Q'}\rho_{QQ'}\right]$$

The *joint von Neumann entropy* is defined as

$$S(\rho_{QQ'}) = -\mathrm{Tr}_{QQ'}\left[\rho_{QQ'} \log \rho_{QQ'}\right]$$

When the subsystems are uncorrelated, $\rho_{QQ'} = \rho_Q\rho_{Q'}$, the joint von Neumann entropy satisfies the so-called *additivity property*

$$S(\rho_{QQ'}) = S(\rho_Q\rho_{Q'}) = S(\rho_Q) + S(\rho_{Q'})$$

However, in more general cases where $\rho_{QQ'} \neq \rho_Q\rho_{Q'}$, the joint entropy satisfies the *subadditivity property*

$$S(\rho_{QQ'}) \leq S(\rho_Q) + S(\rho_{Q'})$$

where the equality holds if and only if the subsystems are independent.

In classical information theory, we saw that $H(X, Y) \geq H(X), H(Y)$.

In quantum information theory, this is not true!

For any pure state $S(\rho_{QQ'}) = 0$, while $S(\rho_Q) = S(\rho_{Q'}) > 0$. For example, consider the Bell state $|\rho_{\beta_0}\rangle\langle\rho_{\beta_0}|$

$$\begin{aligned} S(q) &= -\mathrm{Tr}_q\left[\rho_q \log \rho_q\right] \\ &= -\mathrm{Tr}_q\left(\begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}\begin{bmatrix} \log\frac{1}{2} & 0 \\ 0 & \log\frac{1}{2} \end{bmatrix}\right) \\ &= -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2} = 1 \end{aligned}$$

Thus, in quantum information theory

$$S(q) = S(q') > S(\beta_0) = 0$$

This contrasts with classical information theory, where the joint entropy is always greater than or equal to the entropy of each individual subsystem.

Given a bipartite system QQ', the *conditional von Neumann entropy* is defined as

$$S(\rho_{Q|Q'}) = S(\rho_{QQ'}) - S(\rho_Q)$$

In analogy with the classical concept, the *mutual quantum information* is given by

$$I(\rho_{Q;Q'}) = S(\rho_Q) + S(\rho_{Q'}) - S(\rho_{QQ'})$$

It is semi-positive, $I(\rho_{Q;Q'}) \geq 0$, and is equal to zero iff $\rho_{QQ'} = \rho_Q \, \rho_{Q'}$.

Let $Q = \{|Q_s\rangle, p_s\}$ and $\widetilde{Q} = \{|Q_{\tilde{s}}\rangle, \tilde{p}_s\}$, where $|Q_s\rangle = |\widetilde{Q}\rangle_s$. The *relative von Neumann entropy* is

$$S(\rho_Q \| \rho_{\widetilde{Q}}) = \mathrm{Tr}\left[\rho_Q(\log \rho_Q - \log \rho_{\widetilde{Q}})\right]$$

Given that $S(Q\|Q) = 0$, a larger relative entropy $S(Q\|\widetilde{Q})$ implies a greater distinguishability between the two distributions Q and $\widetilde{Q}$.

6 NISQ Algorithms

Current quantum processors are characterized by significant noise levels. Modern algorithms are designed to account for these hardware limitations.

6.1 Quantum Simulation

Also known as *Hamiltonian simulation*: build a quantum circuit (built from elementary gates) that approximates as accurately as possible the *time-evolution operator* $U(t, t_0)$ of a real physical system

$$|\psi(t_0)\rangle \longmapsto |\psi(t)\rangle = U(t, t_0)|\psi(t_0)\rangle$$

For a time-independent Hamiltonian $\hat{H}$

$$U(t, t_0) = e^{-i\hat{H}(t-t_0)}$$

where $\hat{H}$ determines the time evolution according to the *Schrödinger equation*

$$\hat{H}|\psi(t)\rangle = i\frac{d}{dt}|\psi(t)\rangle$$

The matrix representation of the Hamiltonian of a two-level quantum system (qubit) in the computational basis is

$$\hat{H}_{q_1} = \begin{bmatrix} H_{00} & H_{01} \\ H_{10} & H_{11} \end{bmatrix}$$

But

$$\hat{H}_{q_1} = \sum_{A} h_A \, \sigma_A$$

Similarly, any complex 4×4 matrix

$$\hat{H}_{q_2} = \sum_{A,B} h_{AB} \, \sigma_A \otimes \sigma_B$$

In general, for an n qubit

$$\hat{H}_{q_n} = \sum_{A_1,\ldots,A_n} h_{A_1 \ldots A_n} \, \sigma_{A_1} \otimes \ldots \otimes \sigma_{A_n}$$

Suppose a single-qubit system governed by the Schrödinger equation

$$i\frac{d}{dt}|q(t)\rangle = Z|q(t)\rangle$$

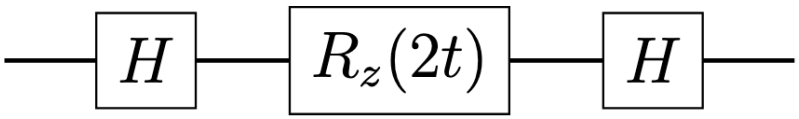
That is, $\hat{H} = Z$. Then $U(t) = e^{-iZt}$. But, we also know that

$$R_z(\gamma) = e^{-iZ\gamma/2}$$

thus

$$U(t) = R_z(2t)$$

Suppose now a single-qubit system with dynamics governed by the Hamiltonian $\hat{H} = X$. The quantum circuit that models its time evolution is



The quantum circuit that simulates the evolution of a generic multi-qubit system is significantly more complex.

Suppose a single-qubit system governed by the Schrödinger equation

$$i\frac{d}{dt}|q_2(t)\rangle = Z^{\otimes 2}|q_2(t)\rangle$$

The time evolution operator is then $U(t) = e^{-iZ^{\otimes 2}t}$. Using the Taylor expansion for the matrix exponential

$$e^{-i\sigma_A^{\otimes n}t} = \cos(t)I - i\sin(t)\sigma_A^{\otimes n}$$

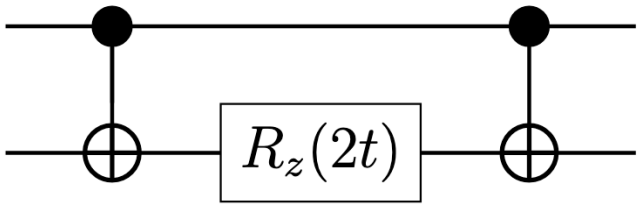
we have that

$$U(t) = \cos(t)I - i\sin(t)Z^{\otimes 2}$$

The two qubit thus evolves according to

$$\begin{aligned} U(t)|q_2\rangle &= e^{-iZ^{\otimes 2}t}\sum_{i,j}\alpha_{ij}|ij\rangle \\ &= \cos(t)\sum_{i,j}\alpha_{ij}|ij\rangle - i\sin(t)\sum_{i,j}\alpha_{ij}Z|i\rangle Z|j\rangle \end{aligned}$$

This time evolution operator can be implemented by the following circuit



6.2 Hybrid Quantum-Classical Algorithms

In the NISQ era, algorithms are typically not executed solely on quantum hardware. Instead, they function as an iterative feedback loop

- Quantum Subroutine: A *parameterized quantum circuit* (*PQC*) prepares a parameterized quantum state and performs measurements to evaluate a specific expectation value.
- Classical Processor: A classical optimization algorithm processes these results and updates the parameters to navigate the solution space.

This synergy defines the class of *hybrid quantum-classical algorithms*.

The most prominent examples include the *Variational Quantum Eigensolver* (*VQE*), the *Quantum Approximate Optimization Algorithm* (*QAOA*), and *Quantum Neural Networks* (*QNN*).

6.3 The Variational Quantum Eigensolver

The VQE is a hybrid (quantum-classical) algorithm.

Consider a quantum system in the state $|\psi\rangle$. If $|E_0\rangle$, $|E_1\rangle$, $|E_2\rangle$ ... denotes the set of orthonormal eigenstates of the Hamiltonian, with $E_0 < E_1 < E_2 < \ldots$ then

$$|\psi\rangle = \sum_{\nu} c_\nu |E_\nu\rangle$$

The *variational theorem*: if $|E_0\rangle$ is the state of minimum energy

$$\hat{H}|E_0\rangle = E_0|E_0\rangle$$

then

$$E_0 = \langle E_0|\hat{H}|E_0\rangle \leq \langle\psi|\hat{H}|\psi\rangle$$

Introduce the set of real parameters $\theta_1, \theta_2, \ldots, \theta_D$ such that

$$|\psi\rangle = |\psi(\theta_1, \theta_2, \ldots, \theta_D)\rangle$$

For example, a two-level quantum system in the state $|\psi(\theta, \phi)\rangle$, where θ and ϕ represent the spherical polar angles on the Bloch sphere.

For notational simplicity, let $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_D)$ denote the vector of parameters. Then, in general $|\psi\rangle = |\psi(\boldsymbol{\theta})\rangle$ and $|\psi(\boldsymbol{\theta}_0)\rangle = |E_0\rangle$. According to the variational theorem

$$E_0 = \langle\psi(\boldsymbol{\theta}_0)| \hat{H} |\psi(\boldsymbol{\theta}_0)\rangle \leq \langle\psi(\boldsymbol{\theta})| \hat{H} |\psi(\boldsymbol{\theta})\rangle = E(\boldsymbol{\theta})$$

The iteration begins with a *trial state* or *ansatz* $|\psi(\widetilde{\boldsymbol{\theta}})\rangle$, estimated using standard techniques. We then use the quantum processor to evaluate the expectation value $E(\widetilde{\boldsymbol{\theta}})$, repeating this procedure for neighboring points in the parameter space.

A classical optimizer compares these results to determine the optimal descent direction within the parameter space. This iterative process continues until the convergence criteria are satisfied

$$E_0 \lesssim E_* = \min_{\boldsymbol{\theta}} E(\boldsymbol{\theta})$$

The VQE is a hybrid quantum-classical algorithm: the quantum subroutine is responsible for 1) state preparation and 2) the measurement of Hamiltonian expectation values; a classical optimizer handles the parameter updates.

1) State preparation: Assume any state $|\psi(\boldsymbol{\theta})\rangle$ can be mapped to a multi-qubit state $|Q(\boldsymbol{\theta})\rangle$. Specifically, the ansatz state

$$|\psi(\widetilde{\boldsymbol{\theta}})\rangle \longleftrightarrow |Q(\widetilde{\boldsymbol{\theta}})\rangle = U(\widetilde{\boldsymbol{\theta}})\,|\mathbf{0}\rangle$$

The *parameterized quantum circuit* $U(\boldsymbol{\theta})$ is constructed from a parameterized sequence of Pauli gates, single-qubit rotations, CNOT gates, and so on.

2) Hamiltonian Expectation Value: The Hamiltonian operator $\hat{H}$ is expressed as a linear combination of Pauli strings (tensor products of Pauli operators)

$$\hat{H} = \sum h_{A_1 \ldots A_n}\, \sigma_{A_1} \otimes \ldots \otimes \sigma_{A_n}$$

The total expectation value $E(\widetilde{\boldsymbol{\theta}})$ is given by

$$E(\widetilde{\boldsymbol{\theta}}) = \sum h_{A_1 \ldots A_n} \langle \mathbf{0} | \, U^{\dagger}(\widetilde{\boldsymbol{\theta}}) \, \sigma_{A_1} \otimes \ldots \otimes \sigma_{A_n} U(\widetilde{\boldsymbol{\theta}}) \, | \mathbf{0} \rangle$$

The VQE framework distributes the workload by using the quantum hardware to measure each individual term

$$\langle \mathbf{0} | U^{\dagger}(\widetilde{\boldsymbol{\theta}}) \, \sigma_{A_1} \otimes \cdots \otimes \sigma_{A_n} \, U(\widetilde{\boldsymbol{\theta}}) | \mathbf{0} \rangle$$

The classical processor handles the final summation. This iterative sampling is conducted across the local parameter neighborhood of $\widetilde{\boldsymbol{\theta}}$, facilitating either gradient estimation or a direct search of the cost landscape. The process is repeated until convergence criteria are satisfied.

In summary

$$E_0 \lesssim E_* = \min_{\boldsymbol{\theta}} \sum h_{A_1 \ldots A_n} \langle \mathbf{0} | \, U^{\dagger}(\boldsymbol{\theta}) \, \sigma_{A_1} \otimes \ldots \otimes \sigma_{A_n} U(\boldsymbol{\theta}) \, | \mathbf{0} \rangle$$

6.4 Introduction to the QAOA

The *Quantum Approximate Optimization Algorithm* (*QAOA*) is a variational algorithm.

A *quadratic function* is given by

$$f(x_1, \dots x_i, \dots, x_n) = a \sum_{i,j=1}^{n} x_i A_{ij} x_j + \sum_{i=1}^{n} b_i x_i + c$$

where A is a symmetric matrix. Or, equivalently

$$f(x_1, \dots x_i, \dots, x_n) = \alpha \sum_{i,i'=1}^{n} x_i \Sigma_{ii'} x_{i'} - \lambda \sum_{i=1}^{n} \mu_i x_i$$

Because c is irrelevant for optimization problems. If the variables x_i are binary, i.e., $x_i = \{0, 1\}$, we denote them by b_s, and then

$$f(b_1, \dots b_s, \dots, b_S) = \alpha \sum_{s,s'=1}^{S} b_s \Sigma_{ss'} b_{s'} - \lambda \sum_{s=1}^{S} \mu_s b_s$$

The optimization of this class of functions is known as a *Quadratic Unconstrained Binary Optimization* (*QUBO*) problem.

The QAOA was specifically designed to address such QUBO formulations.

The Hamiltonian associated with the objective function is constructed as follows. First we use

$$Z\left|0\right\rangle = \left|0\right\rangle = \left(1 - 2 \cdot 0\right)\left|0\right\rangle$$

$$Z\left|1\right\rangle = -\left|1\right\rangle = \left(1 - 2 \cdot 1\right)\left|1\right\rangle$$

that is

$$Z\left|b\right\rangle = \left(1 - 2b\right)\left|b\right\rangle$$

Inverting this relation

$$b\left|b\right\rangle = \frac{1}{2}\left(I - Z\right)\left|b\right\rangle$$

Therefore, for every vector $\left|b_s\right\rangle = \left|\cdots\ b_s\ \cdots\right\rangle$

$$b_s\left|b_s\right\rangle = \frac{1}{2}\left(I - Z_s\right)\left|b_s\right\rangle$$

The operator corresponding to the linear term in the quadratic binary function is

$$\sum_{s=1}^{S} b_s \longmapsto \sum_{s=1}^{S} \frac{1}{2} I - \sum_{s=1}^{S} \frac{1}{2} Z_s$$

For the quadratic term

$$\sum_{s,s'=1}^{S} b_s \Sigma_{ss'} b_{s'} \longmapsto \sum_{s,s'=1}^{S} \frac{1}{2} (I - Z_s) \Sigma_{ss'} \frac{1}{2} (I - Z_{s'})$$

$$= \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} (I - Z_{s'} - Z_s + Z_s Z_{s'})$$

$$= \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} I - \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{2} Z_s + \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} Z_s Z_{s'}$$

Putting everything together

$$\hat{H} = \alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} Z_s Z_{s'} - \sum_{s=1}^{S} \frac{1}{2} \Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \lambda \mu_s \Big) Z_s$$

$$+ \sum_{s=1}^{S} \frac{1}{2} \Big(\alpha \sum_{s'=1}^{S} \frac{\Sigma_{ss'}}{2} - \lambda \mu_s \Big) I$$

To simplify the notation, we introduce

$$\hat{H}_{ZZ} = \alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} Z_s Z_{s'} \qquad \hat{H}_Z = - \sum_{s=1}^{S} \frac{1}{2} \Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \lambda \mu_s \Big) Z_s$$

giving the so-called *cost Hamiltonian*

$$\hat{H}_C = \hat{H}_{ZZ} + \hat{H}_Z$$

The last term was omitted because it is a constant.

The evolution operator associated with this Hamiltonian is

$$U_C(\gamma) = e^{-i\gamma\hat{H}_C} = e^{-i\gamma\hat{H}_{ZZ}} e^{-i\gamma\hat{H}_Z}$$

where γ is a real parameter.

The initial qubit $|Q\rangle_{in}$ that enters $U_C(\gamma)$ is prepared by applying a Hadamard gate to each individual single qubit in state $|0\rangle$

$$|Q\rangle_{in} = |+\rangle^{\otimes S} = H^{\otimes S} |0 \dots 0\rangle$$

Thus, we have

$$|\mathbf{0}\rangle \longmapsto |Q\rangle_{in} \longmapsto U_C(\gamma) |Q\rangle_{in}$$

Next, the *mixer Hamiltonian*

$$\hat{H}_M = \sum_{s=1}^{S} X_s$$

is used to construct the corresponding parameterized unitary

$$U_M(\beta) = e^{-i\beta\hat{H}_M}$$

The state $|Q(\gamma,\beta)\rangle$ that exits the quantum circuit—after the successive application of $U_C(\gamma)$ and $U_M(\beta)$—is now parameterized by the angles γ and β

$$|Q(\gamma,\beta)\rangle = U_M(\beta)\, U_C(\gamma)\, |Q\rangle_{in} = e^{-i\beta\hat{H}_M} e^{-i\gamma\hat{H}_C}\, |Q\rangle_{in}$$

This is the ansatz state used to compute the expectation value of the cost Hamiltonian

$$\langle Q(\gamma,\beta)|\, \hat{H}_C\, |Q(\gamma,\beta)\rangle$$

To simplify the notation, we define

$$F(\gamma,\beta) = \langle Q(\gamma,\beta)|\, \hat{H}_C\, |Q(\gamma,\beta)\rangle$$

The estimated expectation values are passed to a classical optimizer, which proposes refined values for γ and β. This iterative process continues until the parameters converge to an optimal set (γ^*, β^*) that minimizes the quadratic objective function—yielding the bitstring that best approximates the solution to the original QUBO problem.

The QAOA proposes that a better approximation may be found by applying a sequence of p layers of unitaries $U_C(\gamma_k)$ and $U_M(\beta_k)$, $k = 1, \ldots, p$, where each layer is defined by its own unique pair of variational parameters

$$|Q\rangle_{in} \rightarrow U_M(\beta_p)\, U_C(\gamma_p) \ldots U_M(\beta_k)\, U_C(\gamma_k) \ldots U_M(\beta_1)\, U_C(\gamma_1)\, |Q\rangle_{in}$$

$$= \prod_{k=1}^{p} U_M(\beta_k)\, U_C(\gamma_k)\, |Q\rangle_{in} \equiv |Q_p(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle$$

The objective function in $2p$ variables that the classical optimizer must minimize is defined as the expectation value of the cost Hamiltonian

$$F_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \langle Q_p(\boldsymbol{\gamma}, \boldsymbol{\beta})|\, \hat{H}_C\, |Q_p(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle$$

A better solution to the QUBO problem is then given by

$$\min_{\boldsymbol{\gamma}, \boldsymbol{\beta}} \; F_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \min_{\boldsymbol{\gamma}, \boldsymbol{\beta}} \langle Q_p(\boldsymbol{\gamma}, \boldsymbol{\beta})|\, \hat{H}_C\, |Q_p(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle$$

6.5 Adiabatic Quantum Optimization

Consider the instantaneous eigenvalue equation for a time-dependent Hamiltonian $\hat{H}(t)$

$$\hat{H}(t)|E_\nu(t)\rangle = E_\nu(t)|E_\nu(t)\rangle$$

The *quantum adiabatic theorem* states that if a system is prepared in its initial ground state $|\psi(t_0)\rangle = |E_0(t_0)\rangle$ and the Hamiltonian is evolved sufficiently slowly, the system will track the instantaneous ground state

$$|\psi(t)\rangle \approx e^{i\theta(t)}|E_0(t)\rangle$$

More specifically, provided the total evolution time T satisfies the *adiabatic condition*

$$T \gg \hbar/\Delta^2_{min}$$

the final state converges to the global minimum.

Note that, while the state $|\psi(t)\rangle$ remains in the ground level, the energy eigenvalue $E_0(t)$.

The transition from a simple physical system to a complex optimization is governed by a time-dependent interpolation:

1) We begin with an *initial Hamiltonian* H_i representing a simple quantum system with a known, easily prepared ground state.

2) The *final (target) Hamiltonian* H_f encodes the specific cost function of the optimization problem.

The system evolves according to the linear function

$$\hat{H}(s) = (1-s)\hat{H}_i + s\hat{H}_f$$

where $s = t/T \in [0,1]$ is the *normalized time* and T is the *total annealing duration*. In summary

$$|E_0(t_0)\rangle = |+\rangle^{\otimes n} \xrightarrow{\text{Adiabatic Evolution}} |E_0(T)\rangle \approx |x_*\rangle$$

7 Optimization Theory

Optimization—the search for function extrema—is a pillar of computer science and the mathematical bedrock of quantum computing, enabling the solution of classically intractable, high-dimensional problems.

7.1 Continuous Optimization

Let $f : I \subseteq \mathbb{R} \to \mathbb{R}$, $x \mapsto f(x)$, be a differentiable function. A *critical point* $x_c \in I$ of f satisfies

$$\frac{\partial f(x_c)}{\partial x} = 0$$

A critical point can be an optimal point x^* (a minimizer x_{min} or maximizer x_{max}) or a non-optimal point.

The *second derivative test*: we compute $\partial_x^2 f(x_c)$. If it is > 0 the point is a minimum, if it is < 0 it is a maximum, and if it is $= 0$ the test is inconclusive (it may be an inflection point).

Suppose now a differentiable real-valued function in three variables

$$f: U \subseteq \mathbb{R}^3 \to \mathbb{R} \qquad (x, y, z) \mapsto f(x, y, z)$$

A *critical point* $(x, y, z)_c$ satisfies

$$\partial_x f(x, y, z)_c = 0 \qquad \partial_y f(x, y, z)_c = 0 \qquad \partial_z f(x, y, z)_c = 0$$

The second derivative test is more complex. Define the *Hessian matrix*

$$Hf(x, y, z) = \begin{bmatrix} \frac{\partial^2 f(x,y,z)}{\partial x^2} & \frac{\partial^2 f(x,y,z)}{\partial x \partial y} & \frac{\partial^2 f(x,y,z)}{\partial x \partial z} \\ \frac{\partial^2 f(x,y,z)}{\partial y \partial x} & \frac{\partial^2 f(x,y,z)}{\partial y^2} & \frac{\partial^2 f(x,y,z)}{\partial y \partial z} \\ \frac{\partial^2 f(x,y,z)}{\partial z \partial x} & \frac{\partial^2 f(x,y,z)}{\partial z \partial y} & \frac{\partial^2 f(x,y,z)}{\partial z^2} \end{bmatrix}$$

It is symmetric if the partial derivatives commute.

We introduce the determinans

$$\Delta_1(x,y,z)_c = \det\left(\frac{\partial^2 f(x,y,z)_c}{\partial x^2}\right)$$

$$\Delta_2(x,y,z)_c = \det\begin{bmatrix} \frac{\partial^2 f(x,y,z)_c}{\partial x^2} & \frac{\partial^2 f(x,y,z)_c}{\partial x \partial y} \\ \frac{\partial^2 f(x,y,z)_c}{\partial y \partial x} & \frac{\partial^2 f(x,y,z)_c}{\partial y^2} \end{bmatrix}$$

$$\Delta_3(x,y,z)_c = \det\left(Hf(x,y,z)_c\right)$$

Minimizers $(x,y,z)_{\min}$ satisfy

$$\Delta_1(x,y,z)_{min} > 0 \qquad \Delta_2(x,y,z)_{min} > 0 \qquad \Delta_3(x,y,z)_{min} > 0$$

Maximizers $(x,y,z)_{\max}$ satisfy

$$\Delta_1(x,y,z)_{max} < 0 \qquad \Delta_2(x,y,z)_{max} > 0 \qquad \Delta_3(x,y,z)_{max} < 0$$

Any other configuration is a *saddle point*.

In general

$$f: U \subseteq \mathbb{R}^n \to \mathbb{R} \qquad \mathbf{x} \mapsto f(\mathbf{x})$$

Critical points satisfy

$$\partial_1 f(\mathbf{x}_c) = 0 \qquad \partial_2 f(\mathbf{x}_c) = 0 \qquad \dots \qquad \partial_n f(\mathbf{x}_c) = 0$$

In a more compact notation $\nabla f(\mathbf{x}_c) = \mathbf{0}$. We define the $n \times n$ *Hessian matrix*

$$Hf(\mathbf{x}) = \begin{bmatrix} \partial^2_{11} f(\mathbf{x}) & \cdots & \partial^2_{1n} f(\mathbf{x}) \\ \vdots & \ddots & \vdots \\ \partial^2_{n1} f(\mathbf{x}) & \dots & \partial^2_{nn} f(\mathbf{x}) \end{bmatrix}$$

The determinants $\Delta_1(\mathbf{x}_c)$, $\Delta_2(\mathbf{x}_c)$, $\dots$ $\Delta_n(\mathbf{x}_c)$ are defined as above.

Minimizers $\mathbf{x}_{\text{min}}$ satisfy $\Delta_1(\mathbf{x}_{\text{min}}) > 0, \Delta_2(\mathbf{x}_{\text{min}}) > 0, \dots \Delta_n(\mathbf{x}_{\text{min}}) > 0$.

Maximizers $\mathbf{x}_{\text{max}}$ satisfy $\Delta_1(\mathbf{x}_{\text{max}}) < 0, \Delta_2(\mathbf{x}_{\text{min}}) > 0, \Delta_3(\mathbf{x}_{\text{max}}) < 0 \dots$

Any other configuration is a *saddle point*.

The theory is simple but the practical application is complex. Experts rely on numerical approximation methods, such as *gradient descent*:

Choose an initial point $\mathbf{x}_0$ (ideally near the expected minimizer), then move a distance h (the *step size*) in the direction opposite to the gradient

$$\mathbf{x}_1 = \mathbf{x}_0 - h\nabla f(\mathbf{x}_0)$$

We use $\mathbf{x}_1$ to determine the next best approximation

$$\mathbf{x}_2 = \mathbf{x}_1 - h\nabla f(\mathbf{x}_1) = \big(\mathbf{x}_0 - h\nabla f(\mathbf{x}_0)\big) - h\nabla f\big(\mathbf{x}_0 - h\nabla f(\mathbf{x}_0)\big)$$

After k iterations, the minimizer is approximated by

$$\mathbf{x}_k = \mathbf{x}_{k-1} - h\nabla f(\mathbf{x}_{k-1}) = \big(\mathbf{x}_{k-2} - h\nabla f(\mathbf{x}_{k-2})\big) - h\nabla f\big(\mathbf{x}_{k-2} - h\nabla f(\mathbf{x}_{k-2})\big)$$

Every $\mathbf{x}_k$ is ultimately determined by the initial choice $\mathbf{x}_0$. The iterative process terminates once the convergence criteria are met.

Given the *objective function* $f: U \subseteq \mathbb{R}^n \to \mathbb{R}$, $\mathbf{x} \mapsto f(\mathbf{x})$, a *constrained continuous optimization problem*

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{or} \quad \max_{\mathbf{x}} f(\mathbf{x})$$

can include m *equality constraints* $(i = 1, 2, \ldots, m)$

$$h_i: U_i \subseteq \mathbb{R}^n \to \mathbb{R} \qquad \mathbf{x} \mapsto h_i(\mathbf{x}) = b_i$$

and r *inequality constraints* $(j = 1, 2, \ldots, r)$

$$g_j: U_j \subseteq \mathbb{R}^n \to \mathbb{R} \qquad \mathbf{x} \mapsto g_j(\mathbf{x}) \leq d_j$$

In vector notation

$$\mathbf{h}: U \subseteq \mathbb{R}^n \to \mathbb{R}^m \qquad \mathbf{x} \mapsto \mathbf{h}(\mathbf{x}) = \mathbf{b}$$

$$\mathbf{g}: U \subseteq \mathbb{R}^n \to \mathbb{R}^r \qquad \mathbf{x} \mapsto \mathbf{g}(\mathbf{x}) \leq \mathbf{d}$$

Thus, a constrained optimization problem is a minimization or maximization problem

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{or} \quad \max_{\mathbf{x}} f(\mathbf{x})$$

subject to

$$\mathbf{h}(\mathbf{x}) = \mathbf{b} \qquad \mathbf{g}(\mathbf{x}) \leq \mathbf{d}$$

Notice that $\max_{\mathbf{x}} f(\mathbf{x}) = \min_{\mathbf{x}} (-f(\mathbf{x}))$.

The *Lagrange multipliers method* transforms the problem of optimizing a function $f(\mathbf{x})$ subject to m equality constraints $\mathbf{h}$ and r inequality constraints $\mathbf{g}$ into a system of $n + m + r$ equations with $n + m + r$ unknowns. The method establishes that at a minimum

$$\begin{aligned} \nabla_{\mathbf{x}} f(\mathbf{x}^*) &= \sum_{i=1}^{m} \lambda_i^* \nabla_{\mathbf{x}} h_i(\mathbf{x}^*) + \sum_{j=1}^{r} \mu_j^* \nabla_{\mathbf{x}} g_j(\mathbf{x}^*) \\ &= \boldsymbol{\lambda}^* \cdot \nabla_{\mathbf{x}} \mathbf{h}(\mathbf{x}^*) + \boldsymbol{\mu}^* \cdot \nabla_{\mathbf{x}} \mathbf{g}(\mathbf{x}^*) \end{aligned}$$

$\boldsymbol{\lambda}^* = [\,\lambda_1^* \cdots \lambda_m^*\,]^T$ is the *Lagrange multiplier vector* and
$\boldsymbol{\mu}^* = [\,\mu_1^* \cdots \mu_r^*\,]^T$ is the *KKT multiplier vector*.

The solution to the minimization problem can be found by defining the *Lagrangian function*

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{x}) - \boldsymbol{\lambda} \cdot \big[\, \mathbf{h}(\mathbf{x}) - \mathbf{b} \,\big] - \boldsymbol{\mu} \cdot \big[\, \mathbf{g}(\mathbf{x}) - \mathbf{d} \,\big]$$

and imposing for equality constraints

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \mathbf{0} \qquad \nabla_{\boldsymbol{\lambda}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \mathbf{0}$$

and for inequality constraints

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \mathbf{0} \qquad \nabla_{\boldsymbol{\mu}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \leq \mathbf{0}$$

Additionally

$$\boldsymbol{\mu}^* \geq \mathbf{0} \qquad \boldsymbol{\mu}^* \cdot \big[\, \mathbf{g}(\mathbf{x}^*) - \mathbf{d} \,\big] = 0$$

7.2 Dual Optimization Problems

Lagrange Duality allows us to reformulate a *convex primal program* into a corresponding *dual program* that is often easier to solve. Specifically, we can solve a minimization problem in $\mathbf{x}$ by solving its associated maximization problem in the dual variables $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$.

Suppose we wish to solve the following linear program

$$\max_{\mathbf{x}} \mathbf{a}^T\mathbf{x} \quad \text{subject to} \quad H\mathbf{x} = \mathbf{b}$$

The corresponding Lagrangian function is

$$L(\mathbf{x}, \boldsymbol{\lambda}) = \mathbf{a}^T\mathbf{x} + \boldsymbol{\lambda}^T(H\mathbf{x} - \mathbf{b})$$

and the solutions is given by the system of equations

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*) = 0 \qquad \nabla_{\boldsymbol{\lambda}} L(\mathbf{x}^*, \boldsymbol{\mu}^*) = 0$$

Suppose we rewrite the Lagrangian function as follows

$$L(\mathbf{x}, \boldsymbol{\lambda}) = -\boldsymbol{\lambda}^T \mathbf{b} + (\mathbf{a}^T + \boldsymbol{\lambda}^T H)\,\mathbf{x}$$

$$= -\mathbf{b}^T \boldsymbol{\lambda} + \mathbf{x}^T(\mathbf{a} + H^T \boldsymbol{\lambda}) = -\big(\mathbf{b}^T \boldsymbol{\lambda} - \mathbf{x}^T(H^T \boldsymbol{\lambda} + \mathbf{a})\big)$$

Written in this form, the Lagrange multipliers $\boldsymbol{\lambda}$ emerge as the decision variables for the dual program

$$\min_{\boldsymbol{\lambda}} \mathbf{b}^T \boldsymbol{\lambda} \quad \text{subject to} \quad H^T \boldsymbol{\lambda} = -\mathbf{a}$$

while the original primal variables $\mathbf{x}$ now play the role of the new Lagrange multipliers for the dual constraints.

It can be shown that

$$\max_{\mathbf{x}} \mathbf{a}^T \mathbf{x} \quad \text{subject to} \quad G\mathbf{x} \leq \mathbf{d} \quad \text{and} \quad \mathbf{x} \geq \mathbf{0}$$

has as its dual program

$$\min_{\boldsymbol{\mu}} \mathbf{d}^T \boldsymbol{\mu} \quad \text{subject to} \quad G^T \boldsymbol{\mu} \geq \mathbf{a} \quad \text{and} \quad \boldsymbol{\mu} \geq \mathbf{0}$$

7.3 Integer Programs

Integer optimization problems (or programs) are optimization problems where all decision variables are required to be integers. These can be categorized into *constrained integer programs* and *unconstrained integer programs*.

For example, a general *linear integer program* is defined as

$$\min_{\mathbf{z}} \mathbf{a}^T \mathbf{z} \quad \text{or} \quad \max_{\mathbf{z}} \mathbf{a}^T \mathbf{z}$$

subject to

$$H\mathbf{z} = \mathbf{b}\,, \quad G\mathbf{z} \leq \mathbf{d} \quad \text{and} \quad \mathbf{z} \in (\mathbb{Z}_{\geq 0})^N$$

Binary programs are a special case of integer programs. In these problems, the objective function (and constraints) are defined over binary variables $\mathbf{b} = [b_1, \ldots, b_n]^T$ where each $b_i \in \{0, 1\}$ for $i = 1, 2, \ldots, n$

$$\mathbf{f} \colon \{0,1\}^n \to \mathbb{R} \qquad \mathbf{b} \mapsto f(\mathbf{b})$$

Consider a *binary quadratic function* $f(\mathbf{b})$

$$f(\mathbf{b}) = \alpha\, \mathbf{b}^T Q \mathbf{b} + \beta\, \mathbf{c}^T \mathbf{b}$$

where $\mathbf{b} \in \{0,1\}^n$, $Q \in \mathbb{R}^{n \times n}$ is a symmetric matrix, and $\alpha, \beta \in \mathbb{R}$. In index notation

$$f(b_1, \ldots, b_n) = \alpha \sum_{i,j=1}^{n} b_i Q_{ij} b_j + \beta \sum_{i=1}^{n} c_i b_i$$

A *Quadratic Unconstrained Binary Optimization* (*QUBO*) *problem* seeks to find

$$\min_{\mathbf{b} \in 0,1^n} f(\mathbf{b}) \quad \text{or} \quad \max_{\mathbf{b} \in 0,1^n} f(\mathbf{b})$$

where the binary requirement $\mathbf{b} \in \{0,1\}^n$ is the only constraint on the decision variables.

8 Classical Portfolio Optimization Theory

Formulated over seventy years ago, Markowitz's mean-variance model revolutionized the modern understanding of risk and return. Today, it provides the fundamental objective function for quantum portfolio optimization.

8.1 Mathematical Description

Suppose a portfolio consisting of S distinct stocks. The *initial portfolio price* is given by

$$p_P(0) = \sum_{s=1}^{S} p_s(0)$$

This can be expressed as

$$p_P(0) = p_P(0) \sum_{s=1}^{S} w_s(0)$$

The *stock weight* $w_s(0)$ for each asset s is defined as

$$w_s(0) = \frac{p_s(0)}{p_P(0)}$$

Note that, as expected, the sum of the weights is normalized to unity

$$\sum_{s=1}^{S} w_s(0) = 1$$

An (*investment*) *portfolio* is defined by a set of initial stock weights

$$\{w_1(0), w_2(0), \ldots, w_S(0)\}$$

For each stock s, the *stock return* from $t = 0$ to $t = T$ is defined as

$$r_s(T) = p_s(T) - p_s(0)$$

Consequently, the *portfolio return* is the aggregate change in total value

$$r_P(T) = \sum_{s=1}^{S} r_s(T) = p_P(T) - p_P(0)$$

The *return rate of stock* s from $t = 0$ to $t = T$ is defined as

$$R_s(T) = \frac{r_s(T)}{p_s(0)} = \frac{p_s(T) - p_s(0)}{p_s(0)}$$

and the *portfolio return rate* is given by

$$R_P(T) = \frac{r_P(T)}{p_P(0)} = \sum_{s=1}^{S} w_s(0) R_s(T)$$

Price vectors are collections of stock prices

$$\mathbf{p}(0) = \begin{bmatrix} p_1(0) & \dots & p_S(0) \end{bmatrix}^T \qquad \mathbf{p}(T) = \begin{bmatrix} p_1(T) & \dots & p_S(T) \end{bmatrix}^T$$

The *weight vector* at the initial time is defined as

$$\mathbf{w}(0) = \begin{bmatrix} w_1(0) & \dots & w_S(0) \end{bmatrix}^T$$

satisfying the constraint $\mathbf{1}^T \mathbf{w}(0) = 1$, where $\mathbf{1} = [1 \dots 1]^T$.

Similarly, we define the *return rate vector* as

$$\mathbf{R}(T) = \begin{bmatrix} R_1(T) & \dots & R_S(T) \end{bmatrix}^T$$

Using this notation, the *portfolio return rate* becomes

$$R_P(T) = \mathbf{w}^T(0)\,\mathbf{R}(T) = \mathbf{R}^T(T)\,\mathbf{w}(0)$$

For the sake of clarity, we will focus exclusively on the *single-period investment model*.

8.2 The Mean-Variance Model

Suppose the sample space $\Omega = \{\omega_i\}_{i=1}^{n}$ contains all possible market scenarios at time $T > 0$. The *price of the stock s* at time T is defined as the random variable

$$p_s(T)\colon \Omega \to \mathbb{R}^{\geq 0} \qquad \omega_i \mapsto \big(p_s(T)\big)(\omega_i) = p_s(T, \omega_i)$$

Since the stock price p_s is a random variable, we can define the *expected price of the stock s* at time T as

$$\mathbb{E}\big(p_s(T)\big) = \sum_{i=1}^{n} P\big(p_s(T, \omega_i)\big)\, p_s(T, \omega_i) = \sum_{i=1}^{n} p_i\, p_s(T, \omega_i)$$

Given that $p_s(T)$ is a random variable, and since $r_s(T)$ and $R_s(T)$ are defined in terms of it, they are also random variables. Specifically, we can define the *expected return rate of stock s* by

$$\mathbb{E}\big(R_s(T)\big) = \sum_{i=1}^{n} p_i R_s(T, \omega_i)$$

And the *expected portfolio return rate*

$$\mathbb{E}\big(R_P(T)\big) = \sum_{s=1}^{S} w_s(0)\,\mathbb{E}\big(R_s(T)\big)$$

To quantify the variability of the stock price, and ultimately the uncertainty of the investment, we employ the *variance of the return rate*

$$\mathrm{Var}\big(R_s(T)\big) = \sum_{i=1}^{n} p_i \left[R_s(T,\omega_i) - \mathbb{E}\big(R_s(T)\big)\right]^2$$

Suppose a portfolio P consisting of only two assets, s and s'. The properties of the covariance between two random variables imply

$$\mathrm{Var}\big(R_P(T)\big) = w_s^2(0)\,\mathrm{Var}\big(R_s(T)\big) + w_{s'}^2(0)\,\mathrm{Var}\big(R_{s'}(T)\big)$$

$$+ 2w_s(0)w_{s'}(0)\,\mathrm{Cov}\big(R_s(T), R_{s'}(T)\big)$$

Using the standard sigma notation for variances and covariances

$$\mathrm{Var}\big(R_P(T)\big) = \sigma_P^2(T) \qquad \mathrm{Var}\big(R_s(T)\big) = \sigma_{ss}(T) = \sigma_s^2(T)$$

$$\mathrm{Cov}\big(R_s(T), R_{s'}(T)\big) = \sigma_{ss'}(T)$$

We obtain the *variance of the portfolio return rate*

$$\sigma_P^2(T) = w_s^2(0)\,\sigma_s^2(T) + w_{s'}^2(0)\,\sigma_{s'}^2(T) + 2w_s(0)w_{s'}(0)\,\sigma_{ss'}(T)$$

For a portfolio of S stocks, the sum for all s and s' gives

$$\sigma_P^2(T) = \sum_{s=1}^{S} w_s^2(0)\sigma_s^2(T) + 2\sum_{\substack{s,s'=1 \\ s\neq s'}}^{S} w_s(0)w_{s'}(0)\sigma_{ss'}(T)$$

$$= \sum_{s=1}^{S} w_s(0)\sigma_{ss}(T)w_s(0)$$

$$+ \sum_{\substack{s,s'=1 \\ s<s'}}^{S} w_s(0)\sigma_{ss'}(T)w_{s'}(0) + \sum_{\substack{s,s'=1 \\ s>s'}}^{S} w_s(0)\sigma_{ss'}^2(T)w_{s'}(0)$$

We introduce the conventional notation

$$\mathbb{E}\big(R_P(T)\big) = \mu_P(T) \qquad \mathbb{E}\big(R_s(T)\big) = \mu_s(T)$$

giving

$$\mu_P(T) = \sum_{s=1}^{S} w_s(0)\,\mu_s(T)$$

The weight vector $\mathbf{w}(0) = [\,w_1(0)\ldots w_S(0)\,]^T$ and the vector of expected return rates $\boldsymbol{\mu}(T) = [\,\mu_1(T)\ldots\mu_S(T)\,]^T$ can be used to express the expected portfolio return in vector form

$$\mu_P(T) = \mathbf{w}^T(0)\,\boldsymbol{\mu}(T) = \boldsymbol{\mu}^T(T)\,\mathbf{w}(0)$$

Using the definition of the covariance matrix for S random variables—in this case, the stock return rates

$$\Sigma\big(R_1(T),\ldots,R_S(T)\big) = \Sigma\big(\mathbf{R}(T)\big) = \begin{bmatrix} \sigma_{11}(T) & \ldots & \sigma_{1S}(T) \\ \vdots & \ldots & \vdots \\ \sigma_{S1}(T) & \ldots & \sigma_{SS}(T) \end{bmatrix}$$

we rewrite the variance of the portfolio return rate using vector notation

$$\sigma_P^2(T) = \mathbf{w}^T(0)\,\Sigma\big(\mathbf{R}(T)\big)\,\mathbf{w}(0)$$

With this, the standard deviation becomes

$$\sigma_P(T) = \sqrt{\mathbf{w}^T(0)\,\Sigma\big(\mathbf{R}(T)\big)\,\mathbf{w}(0)}$$

In *modern portfolio theory*, also known as the *mean-variance model*, it is assumed that investment risk is represented by the variance of the portfolio return rate

$$\mathcal{R} = \alpha\,\mathbf{w}^T(0)\,\Sigma\big(\mathbf{R}(T)\big)\,\mathbf{w}(0)$$

where the positive quantity α is the *risk aversion coefficient*.

8.3 Portfolio Optimization as a Quadratic Program

The optimization problem an investor must solve is the following: given a certain risk level $\mathcal{R}$

$$\mathcal{R} = \alpha\, \mathbf{w}^T(0)\Sigma\big(\mathbf{R}(T)\big)\mathbf{w}(0)$$

what is the maximum expected return? Where

$$\max_{\mathbf{w}(0)} \boldsymbol{\mu}^T(T)\, \mathbf{w}(0)$$

and $\mathbf{1}^T\mathbf{w}(0) = 1$. In a simpler and more conventional notation

$$\max_{\mathbf{w}} \boldsymbol{\mu}^T\mathbf{w} \quad \text{subject to} \quad \alpha\, \mathbf{w}^T\Sigma\mathbf{w} = \mathcal{R} \quad \text{and} \quad \mathbf{1}^T\mathbf{w} = 1$$

The investor can equivalently choose a target expected return $\mathcal{M}$ and seek to minimize the risk. The optimization problem thus becomes

$$\min_{\mathbf{w}} \alpha\, \mathbf{w}^T\, \Sigma\, \mathbf{w} \quad \text{subject to} \quad \boldsymbol{\mu}^T\mathbf{w} = \mathcal{M} \quad \text{and} \quad \mathbf{1}^T\mathbf{w} = 1$$

For a *binary portfolio*, where the decision variables are restricted to $\{0,1\}$—meaning a stock is either included in or excluded from the portfolio—the optimization problem becomes

$$\min_{\mathbf{b}} \alpha\, \mathbf{b}^T \Sigma\, \mathbf{b} \quad \text{subject to} \quad \boldsymbol{\mu}^T \mathbf{b} = \mathcal{M}$$

where $\mathbf{b} \in \{0,1\}^S$ and Σ is the symmetric covariance matrix. In index notation

$$\min \alpha \sum_{s,s'=1}^{S} b_s\, \Sigma_{ss'}\, b_{s'} \quad \text{subject to} \quad \sum_{s=1}^{S} \mu_s b_s = \mathcal{M}$$

where $b_s \in \{0,1\}$ and $\Sigma_{ss'} = \Sigma_{s's}$. We can use the method of Lagrange multipliers to restate the problem as an unconstrained optimization

$$\min \alpha \sum_{s,s'=1}^{S} b_s\, \Sigma_{ss'}\, b_{s'} - \lambda\Big(\sum_{s=1}^{S} \mu_s b_s - \mathcal{M}\Big)$$

$$= \min \alpha \sum_{s,s'=1}^{S} b_s\, \Sigma_{ss'}\, b_{s'} - \lambda \sum_{s=1}^{S} \mu_s b_s + \lambda \mathcal{M}$$

Finally, since $\lambda\mathcal{M}$ is a constant that does not shift the location of the minimum, the optimization reduces to a QUBO problem

$$\min \alpha \sum_{s,s'=1}^{S} b_s \, \Sigma_{ss'} \, b_{s'} - \lambda \sum_{s=1}^{S} \mu_s b_s$$

9 Quantum Portfolio Optimization

In the NISQ era, hybrid algorithms like VQE and QAOA provide a robust path for achieving potential quantum speedups in asset selection.

9.1 Portfolio Optimization via the VQE

We translate the portfolio cost function into a *cost Hamiltonian*

$$f \longleftrightarrow \hat{H}_C$$

The goal is to map discrete asset selection to a quantum ground-state search

$$\min_{\mathbf{b}} f(\mathbf{b}) \longleftrightarrow \min_{\boldsymbol{\theta}} E(\boldsymbol{\theta})$$

where $E(\boldsymbol{\theta}) = \langle Q(\boldsymbol{\theta})| \hat{H}_C |Q(\boldsymbol{\theta})\rangle$. We prepare the trial (ansatz) state $|\widetilde{Q}\rangle$

$$|0\rangle^{\otimes n} = |\mathbf{0}\rangle \longmapsto |\widetilde{Q}\rangle = U(\widetilde{\boldsymbol{\theta}})\,|\mathbf{0}\rangle$$

In general

$$\hat{H} = \sum_{A_1,\ldots,A_n} h_{A_1\ldots A_n}\, \sigma_{A_1} \otimes \ldots \otimes \sigma_{A_n}$$

Thus, according to the VQE

$$E_{VQE} = \min_{\boldsymbol{\theta}} \sum_{A_1,\ldots,A_n} h_{A_1\ldots A_n} \langle \mathbf{0}| U^{\dagger}(\boldsymbol{\theta}) \sigma_{A_1} \otimes \ldots \otimes \sigma_{A_n} U(\boldsymbol{\theta}) |\mathbf{0}\rangle \gtrsim E_0$$

The QUBO for a binary portfolio of S assets is

$$\min f(b_1 \ldots b_s \ldots b_S) = \min \alpha \sum_{s,s'=1}^{S} b_s\, \Sigma_{ss'}\, b_{s'} - \lambda \sum_{s=1}^{S} \mu_s b_s$$

To construct the cost Hamiltonian, we map binary to Ising variables

$$b\,|b\rangle = \frac{1}{2}\,(I - Z)\,|b\rangle$$

that is

$$b_s\,|\cdots\, b_s\, \cdots\rangle = \frac{1}{2}\,(I - Z_s)\,|\cdots\, b_s\, \cdots\rangle$$

With the Ising mapping, $b_s \mapsto \frac{1}{2}(I - Z_s)$, the cost Hamiltonian becomes

$$\hat{H}_C = \hat{H}_{ZZ} + \hat{H}_Z$$

where the quadratic (risk) and linear (return) terms are defined as

$$\hat{H}_{ZZ} = \alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} Z_s Z_{s'} \qquad \hat{H}_Z = -\sum_{s=1}^{S} \frac{1}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \lambda \mu_s\Big) Z_s$$

A constant offset term has been omitted for clarity.

The portfolio optimization process then reduces to

$$\begin{aligned} \min_{\boldsymbol{\theta}} \langle Q(\boldsymbol{\theta})| \hat{H}_C |Q(\boldsymbol{\theta})\rangle &= \min_{\boldsymbol{\theta}} \langle Q(\boldsymbol{\theta})| (\hat{H}_{ZZ} + \hat{H}_Z) |Q(\boldsymbol{\theta})\rangle \\ &= \min_{\boldsymbol{\theta}} \langle 0 \ldots 0| U^\dagger(\boldsymbol{\theta})(\hat{H}_{ZZ} + \hat{H}_Z) U(\boldsymbol{\theta}) |0 \ldots 0\rangle \end{aligned}$$

9.2 Portfolio Optimization via the QAOA

The QAOA is a variational framework that builds upon the VQE by utilizing alternating operator layers.

Instead of a general heuristic ansatz, the state is evolved through p layers of alternating unitaries

$$U_M(\beta_k)U_C(\gamma_k)$$

where $k = 1, \ldots, p$ and the *cost*, and *mixer unitaries* are defined as

$$U_C(\gamma_k) = e^{-i\gamma_k \hat{H}_C} \qquad U_M(\beta_k) = e^{-i\beta_k \hat{H}_M}$$

To implement $U_C(\gamma)$ on hardware, we decompose the cost unitary. Since all terms in $\hat{H}_C$ (the ZZ and Z terms) commute, we can split the exponential exactly

$$U_C(\gamma) = e^{-i\gamma \hat{H}_C} = e^{-i\gamma \hat{H}_{ZZ}} e^{-i\gamma \hat{H}_Z}$$

The operator $e^{-i\,\gamma \hat{H}_Z}$ is equivalent to

$$e^{-i\,\gamma \hat{H}_Z} = \exp\Big[\sum_{s=1}^{S} i\,\frac{\gamma}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \lambda\mu_s\Big) Z_s\Big]$$

$$= \bigotimes_{s=1}^{S} \exp\Big[i\,\frac{\gamma}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \lambda\mu_s\Big) Z_s\Big] = \bigotimes_{s=1}^{S} e^{-i\gamma a_s Z_s}$$

where

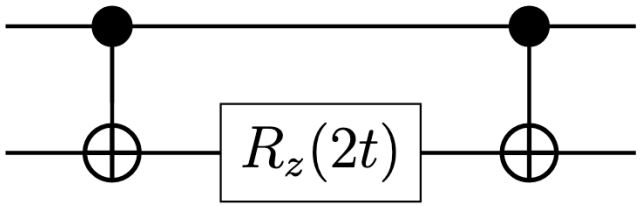
$$a_s = -\frac{1}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \lambda\mu_s\Big)$$

The unitary is, thus, equivalent to

$$e^{-i\gamma \hat{H}_Z} = \bigotimes_{s=1}^{S} e^{-i\gamma a_s Z_s} = \bigotimes_{s=1}^{S} R_z(2\gamma a_s)$$

Physically, this implements a rotation of each qubit s by an angle of $2\gamma a_s$ about the z-axis.

From our study of quantum simulations, the entangling operator $e^{-iZ^{\otimes 2}t}$ is implemented via the following circuit



$$e^{-i\delta Z_s Z_{s'}} \left|b_s\right\rangle \left|b_{s'}\right\rangle = \mathrm{CNOT}\left(I \otimes R_z(2\delta)\right) \mathrm{CNOT} \left|b_s\right\rangle \left|b_{s'}\right\rangle$$

Consequently, the risk-evolution factor is decomposed as

$$e^{-i\gamma \hat{H}_{ZZ}} = \exp\left[\sum_{s,s'=1}^{S} -i\gamma a_{ss'} Z_s Z_{s'}\right] = \bigotimes_{s,s'=1}^{S} e^{-i\gamma a_{ss'} Z_s Z_{s'}}$$

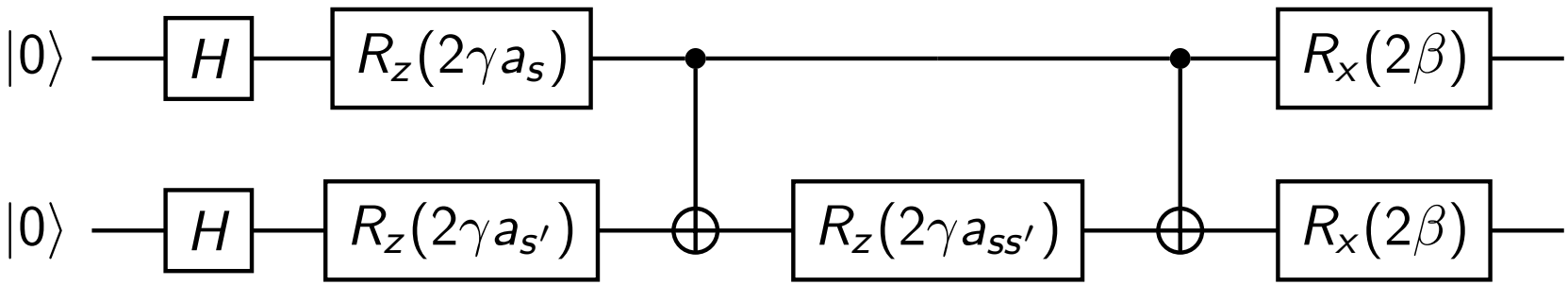
$$= \bigotimes_{\substack{s=1 \\ s'<s}}^{S} \mathrm{CNOT}_{ss'} \left(I \otimes R_z(2\gamma a_{ss'})\right) \mathrm{CNOT}_{ss'}$$

where $a_{ss'} = \alpha \Sigma_{ss'}/4$.

Finally, the mixer unitary $U_M(\beta)$ is generated by the mixer Hamiltonian $\hat{H}_M = \sum X_s$. The corresponding parameterized unitary is given by

$$U_M(\beta) = e^{-i\beta \hat{H}_M} = e^{-i\beta \sum_{s=1}^{S} X_s} = \bigotimes_{s=1}^{S} e^{-i\beta X_s} = \bigotimes_{s=1}^{S} R_x(2\beta)$$

In conclusion, the resulting quantum circuit for each qubit pair follows the operator sequence



This gate sequence is repeated for every pair of qubits $s, s' = 1, \ldots, S$, and across every layer $k = 1, \ldots, p$. In each layer, the unitaries are parameterized by a unique pair of variational angles (γ_k, β_k).

10 Quantum Machine Learning

QML leverages high-dimensional feature spaces to identify complex correlations and accelerate convergence beyond classical limits.

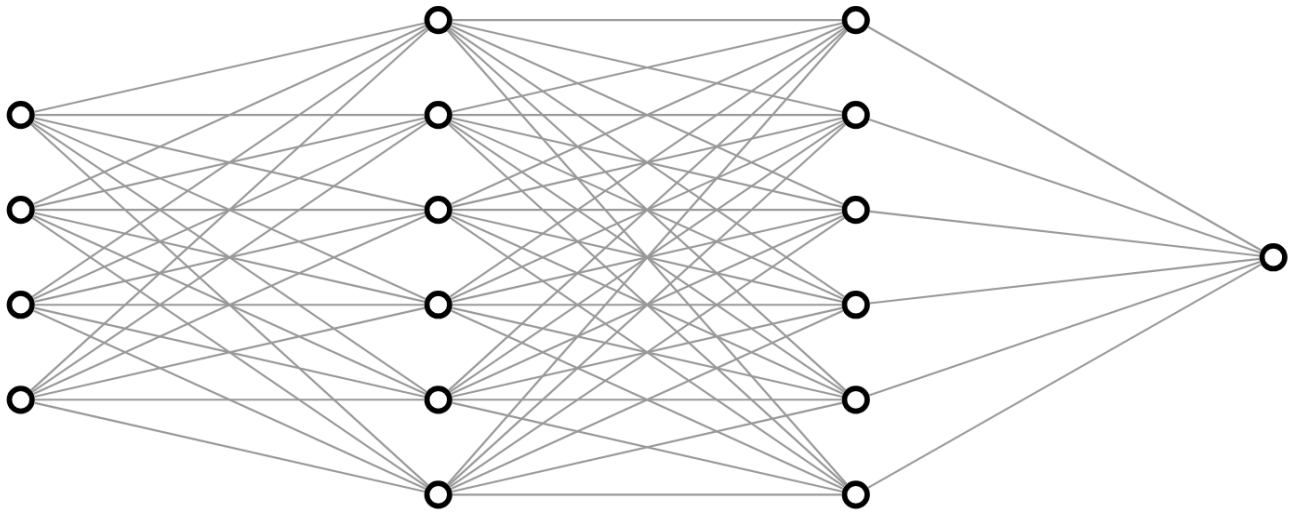
10.1 Machine Learning

Machine Learning (*ML*) is an advanced computational and statistical framework designed to identify patterns within data for predictive modeling and decision-making.

The architecture uses datasets to discover underlying structures, which are then refined and tested through systematic training and validation cycles.

- *Supervised learning*
- *Unsupervised learning*
- *Reinforcement learning*

1) *Neural Networks (NN)*

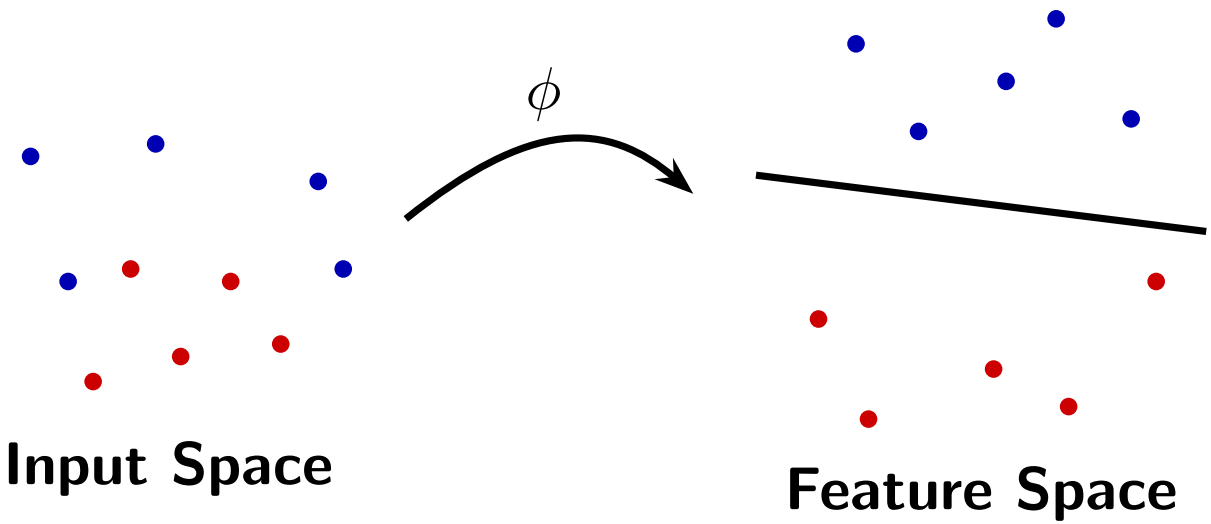


Universal Approximation Theorem: Any continuous function can be approximated to an arbitrary degree of precision by a neural network, given a sufficient density of hidden nodes and appropriate activation functions.

Activation functions (e.g., ReLU, Sigmoid): introduce the non-linearity required to capture complex, high-dimensional patterns.

Backpropagation: optimization procedure that iteratively updates model parameters, using gradient descent to converge toward the optimal fit.

2) *Support Vector Machines (SVM)*



Define the *feature map*

$$\phi : X \to F \qquad \mathbf{x} \mapsto \phi(\mathbf{x})$$

Inner products become

$$\langle \mathbf{x}^{(n)}, \mathbf{x}^{(n')} \rangle \qquad \longmapsto \qquad \langle \phi(\mathbf{x}^{(n)}), \phi(\mathbf{x}^{(n')}) \rangle = k(\mathbf{x}^{(n)}, \mathbf{x}^{(n')})$$

The *kernel trick*: inner products in the high-dimensional feature space can be computed directly from the original inputs via the *kernel function* k, without ever explicitly computing ϕ.

10.2 Data Encoding

The process of encoding classical information into a quantum system.

1) *Basis encoding*

Suppose $n = 1, \ldots, N$ samples, each defined by two features $x_1, x_2 \in \{0, 1, 2, 3\}$. In binary notation

$$0 = 00 \qquad 1 = 01 \qquad 2 = 10 \qquad 3 = 11$$

The *feature vector* of the n-th sample is denoted as

$$\mathbf{x}^{(n)} = \begin{bmatrix} x_1^{(n)} & x_2^{(n)} \end{bmatrix}^T \longleftrightarrow \mathbf{x}_b^{(n)} = \begin{bmatrix} x_{1b}^{(n)} & x_{2b}^{(n)} \end{bmatrix}^T$$

$$= \begin{bmatrix} b_{11}^{(n)} b_{12}^{(n)} & b_{21}^{(n)} b_{22}^{(n)} \end{bmatrix}^T$$

We map the binary feature vector $\mathbf{x}_b^{(n)}$ to a computational basis state within a 2^4-dimensional Hilbert space

$$\mathbf{x}_b^{(n)} \longmapsto |\mathbf{x}_b^{(n)}\rangle = |b_{11}^{(n)} b_{12}^{(n)} b_{21}^{(n)} b_{22}^{(n)}\rangle$$

Similarly, for any other sample n'

$$\mathbf{x}_b^{(n')} \longmapsto |\mathbf{x}_b^{(n')}\rangle = |b_{11}^{(n')} b_{12}^{(n')} b_{21}^{(n')} b_{22}^{(n')}\rangle$$

The entire dataset X can be represented as a state vector

$$X = \begin{bmatrix} x_1^{(n)} & x_2^{(n)} \\ x_1^{(n')} & x_2^{(n')} \end{bmatrix} \longmapsto |X_b\rangle = \frac{1}{\sqrt{2}} |\mathbf{x}_b^{(n)}\rangle + \frac{1}{\sqrt{2}} |\mathbf{x}_b^{(n')}\rangle$$

More generally, for a sample with I features

$$\mathbf{x}^{(n)} = \left[x_1^{(n)} \dots x_I^{(n)}\right]^T \longmapsto |\mathbf{x}_b^{(n)}\rangle = |b_{11}^{(n)} \dots b_{1n_1}^{(n)} \dots b_{I1}^{(n)} \dots b_{In_I}^{(n)}\rangle$$

The state vector associated with the full classical dataset $X = \{x_i^{(n)}\}$ is then defined as the normalized sum of all sample states

$$X \longmapsto |X\rangle = \frac{1}{\sqrt{N}} \sum_{n=1}^{N} |\mathbf{x}^{(n)}\rangle$$

2) *Angle (rotational) encoding*

A qubit in the Bloch sphere is described by the unit vector

$$|q(\vartheta,\phi)\rangle = \cos(\vartheta/2)|0\rangle + e^{i\phi}\sin(\vartheta/2)|1\rangle$$

where $\vartheta \in [0,\pi]$ and $\phi \in [0,2\pi)$. A rotation of θ radians about the y-axis is given by

$$R_y(\theta) = e^{-i\theta Y/2} = \cos(\theta/2)I - i\sin(\theta/2)Y$$

For example

$$R_y(\theta)\,|0\rangle = \cos(\theta/2)\,|0\rangle + \sin(\theta/2)\,|1\rangle$$

To encode a feature $x_i^{(n)}$ into a quantum state, we first rescale $x_i^{(n)}$ so that $x_i^{(n)} \mapsto \theta_i^{(n)} \in [0,\pi]$ and then

$$\begin{aligned}\theta_i^{(n)} \longmapsto |\theta_i^{(n)}\rangle &= R_y(\theta_i^{(n)})\,|0\rangle \\ &= \cos(\theta_i^{(n)}/2)\,|0\rangle + \sin(\theta_i^{(n)}/2)\,|1\rangle\end{aligned}$$

All features, $i = 1, \ldots, I$, are normalized into the $[0, \pi]$ range using the following transformation

$$\theta_i^{(n)} = \frac{x_i^{(n)} - x_{i,\min}}{x_{i,\max} - x_{i,\min}} \pi$$

We then map the *normalized feature vector* $\boldsymbol{\theta}^{(n)}$ to a quantum state via single-qubit rotations

$$\mathbf{x}^{(n)} \longmapsto \boldsymbol{\theta}^{(n)} = \left[\theta_1^{(n)} \ldots \theta_I^{(n)}\right]^T \longmapsto |\boldsymbol{\theta}^{(n)}\rangle = \bigotimes_{i=1}^{I} R_y(\theta_i^{(n)}) |0\rangle$$

A second piece of classical data can be encoded using a rotation about the z-axis

$$R_z(\phi) = e^{-i\phi Z/2} = \cos(\phi/2) I - i \sin(\phi/2) Z$$

The combined rotations yield the following state (up to a global phase)

$$R_z(\phi) R_y(\theta) |0\rangle = \cos(\theta/2) |0\rangle + e^{i\phi} \sin(\theta/2) |1\rangle$$

We can encode $2I$ features using the rotation angles of I qubits. For example, odd-indexed features can be mapped to rotations about the y-axis

$$x_{2i-1}^{(n)} \mapsto \bar{x}_{2i-1}^{(n)} \mapsto \theta_{2i-1}^{(n)}$$

while even-indexed features are mapped to rotations about the z-axis

$$x_{2i}^{(n)} \mapsto \bar{x}_{2i}^{(n)} \mapsto \phi_{2i}^{(n)}$$

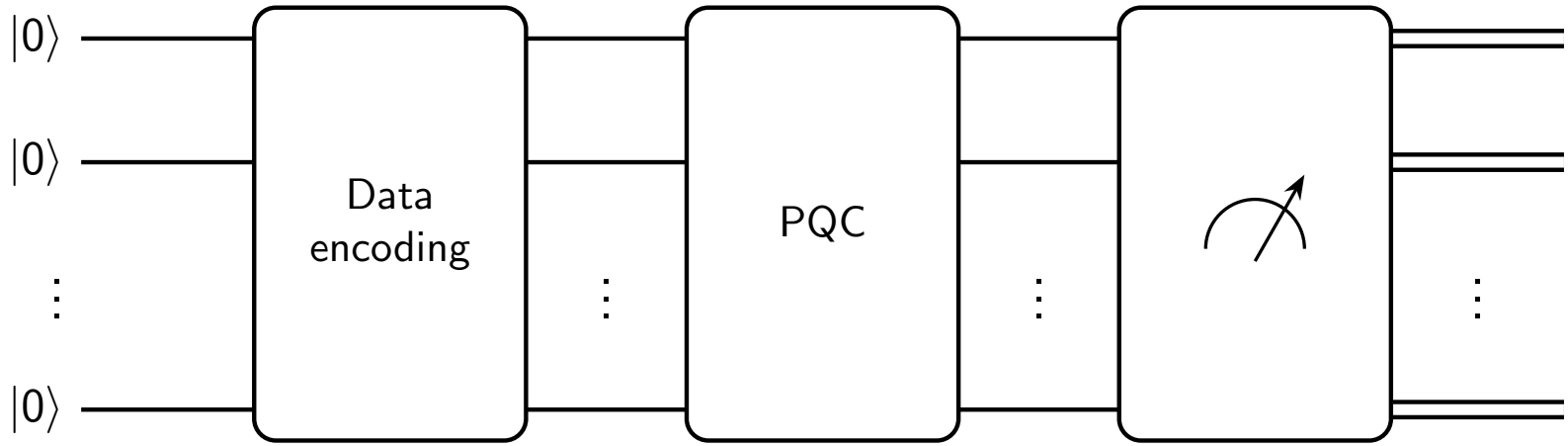
Ignoring the global phase, two classical data points are encoded into a single qubit as follows

$$\begin{aligned}\left[x_{2i-1}^{(n)} \; x_{2i}^{(n)}\right]^T \longmapsto & R_z(\phi_{2i}^{(n)}) R_y(\theta_{2i-1}^{(n)}) \left|0\right\rangle \\ & = \cos(\theta_{2i-1}^{(n)}/2) \left|0\right\rangle + e^{\phi_{2i}^{(n)}} \sin(\theta_{2i-1}^{(n)}/2) \left|1\right\rangle\end{aligned}$$

Finally, the complete feature vector for the n-th sample is encoded into the following state

$$\mathbf{x}^{(n)} = \left[\ldots x_{2i-1}^{(n)} \; x_{2i}^{(n)} \ldots\right]^T \longmapsto \left|\mathbf{x}_{\boldsymbol{\theta}}^{(n)}\right\rangle = \bigotimes_{i=1}^{I} R_z(\phi_{2i}^{(n)}) R_y(\theta_{2i-1}^{(n)}) \left|0\right\rangle$$

10.3 Quantum Neural Networks



Analogous to classical NN, a theorem for QNNs states that any continuous function can be approximated to arbitrary precision by a PQC, given sufficient circuit depth and a high enough density of trainable parameters.

The role of non-linear activation functions in classical NN is now played by the quantum encoding maps.

The optimization process is not performed by backpropagation, but by a hybrid quantum-classical loop using classical optimizers, as is standard in NISQ-era algorithms.

10.4 Quantum Kernels

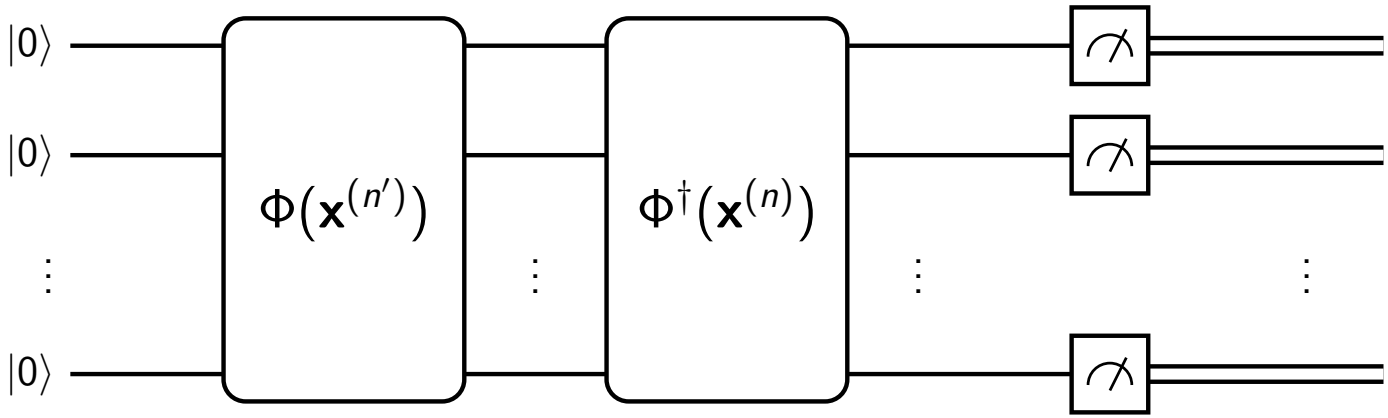
Let the feature map ϕ correspond to the quantum data encoding map Φ

$$\phi\colon X \to F \qquad \longleftrightarrow \qquad \Phi\colon X \to \mathcal{H}$$

The *quantum kernel function* evaluates the inner product between two encoded states

$$\kappa(\mathbf{x}^{(n)}, \mathbf{x}^{(n')}) = |\langle \mathbf{0}|\Phi^{\dagger}(\mathbf{x}^{(n)})\Phi(\mathbf{x}^{(n')})|\mathbf{0}\rangle|^2$$

In practice, this is implemented via the following circuit architecture



The probability of measuring the all-zero state yields the squared overlap

$$P(|\mathbf{0}\rangle) = |\langle \mathbf{0}|\Phi^{\dagger}(\mathbf{x}^{(n)})\Phi(\mathbf{x}^{(n')})|\mathbf{0}\rangle|^2$$

Do you need help with QC for Finance? Let me help you!

www.ozatp.com