Quantum Technology

Justin H. Wilson

2024-12-21

Table of contents

Preface

The goal for this text is to be a snapshot in time of quantum technologies: the good, the bad, and the ugly. As of 2024, there has been a large amount of industry interest and progress to develop quantum technologies, and a student approaching this industry should have the basic knowledge to understand *what* is trying to be achieved and *how* they are trying to achieve it. To cut through the PR of industry, this text offers the author's personal perspective on what has been achieved, where things need to go, and the challenges to get there. This is not meant to be an authoritative guide on any one of the technologies presented here, but a jumping off point for the interested student.

Finally, this is a Quarto book.

To learn more about Quarto books visit https://quarto.org/docs/books.

1 Introduction

In 2019 [1], Google reached a significant milestone in quantum computing when they achieved “Quantum Advantage”–demonstrating for the first time that a quantum computer could surpass the capabilities of classical computers, albeit for a specific, specialized task. Using their Sycamore processor with 53 qubits, they showed that sampling from a quantum circuit a million times took only 200 seconds, while the equivalent task would take approximately 10,000 years on a classical supercomputer. This breakthrough marked a turning point in quantum computing, igniting increased interest and investment in quantum technologies. Building on this success, in December 2024, Google announced their new Willow chip [2], representing their latest advancement in quantum hardware. Today, numerous companies and research institutions worldwide are racing to develop quantum hardware, each pursuing different technological approaches to challenge the computational limits of classical computers.

What is Quantum Advantage?

Quantum advantage (or quantum supremacy) refers to the demonstration that a quantum computer can solve a specific problem significantly faster than the best known classical algorithm. However, the problem doesn’t necessarily need to be useful–it just needs to be well-defined and verifiable.

This text aims to give undergraduate students a comprehensive introduction to quantum technology and computation. We will begin by exploring the fundamental mathematical framework that underlies quantum computing, building from basic principles to more advanced concepts. Crucially, we will see what specific things a quantum computer can achieve that a classical computer would struggle with. With this foundation, we will examine three of the most promising current quantum computing technologies: superconducting qubits, photonic quantum computing, and ion traps. Each of these approaches offers unique advantages and faces distinct challenges, which we will analyze in detail.

To bridge theory with practice, we will utilize IBM’s Qiskit software platform to implement basic quantum computations, providing hands-on experience with quantum programming. As we progress, we will explore critical practical considerations in quantum computing, including error mitigation and correction strategies. Time permitting, we will venture into the cutting-edge field of topological quantum computing, which offers a potentially more robust approach to quantum computation.

Throughout this text, we will maintain a balanced perspective, examining both the tremendous potential and significant challenges facing quantum computing technology. Our goal is to equip students with both theoretical understanding and practical insights into this rapidly evolving field.

1.1 The Quantum-Classical Arms Race

The story of Google's quantum supremacy claim illustrates a fascinating dynamic in the field of quantum computing–an ongoing arms race between quantum and classical algorithms. When Google first announced their achievement with the Sycamore processor [1], they estimated that their quantum sampling task would take a classical supercomputer approximately 10,000 years. However, within months, IBM researchers developed improved classical algorithms that could potentially perform the same calculation in just 2.5 days [3]. Further work even demonstrated that using tensor networks, the problem could be solved *faster* on a modern superconductor with ExaFLOPS performance [4].

This back-and-forth highlights several important lessons. First, it demonstrates the remarkable adaptability of classical computing. As quantum computers advance, classical algorithm developers find increasingly clever ways to simulate quantum systems or solve specific problems more efficiently. This competition drives innovation in both fields–quantum hardware must continually improve to maintain its advantage, while classical algorithms become more sophisticated in response.

Second, it serves as a cautionary tale about interpreting quantum computing announcements, particularly those aimed at the general public. While the achievement of quantum advantage represents a genuine milestone, the initial 10,000-year estimate proved overly optimistic. This pattern has repeated with various quantum computing companies, where marketing claims sometimes outpace peer-reviewed scientific validation. For students and researchers in the field, it's crucial to maintain a balanced perspective–acknowledging genuine breakthroughs while critically evaluating bold claims.

The recent announcement of Google's Willow chip [2] represents another step forward, but should be viewed within this context of ongoing competition and careful validation. This healthy tension between quantum and classical approaches ultimately benefits both fields, pushing the boundaries of what's computationally possible while maintaining rigorous scientific standards.

1.2 Early Computing: The Lesson of Transistors

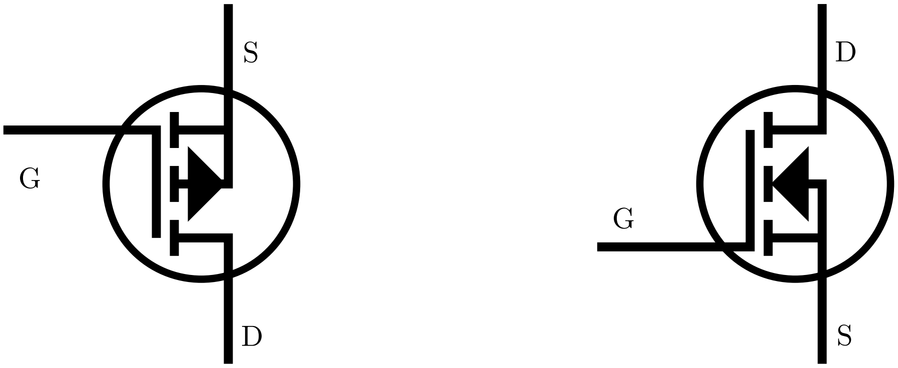
In Ref. [5] there are a few quotes about early computing

> *"Computers in the future may weigh no more than 1.5 tons."* –Popular Mechanics, forecasting the relentless marcho f science, 1949

> *"I think there is a world market for maybe five computers."* –Thomas Watson, chairman of IBM, 1943

These quotes raise important points about early technology. While the theory of modern computing really took off with Turing in 1937 [6], the technological advancement necessary for modern computurs would not occur until later: when the transistor came about.

1.2.1 The Dream: Field Effect Transistors



(a) P-channel

(b) N-channel

Fig. 1.1: Example Circuit diagrams for MOSFETs (*enh*)

To enable classical computing, we need something like a "switch" that can be on and off, keeping track of whether or not something like, current is flowing. This is hard to do with traditional circuit elements like resistors, capacitors, and inductors. It requires something more nonlinear: A switch that only allows current flow when a voltage is applied, a **transistor**.

Fig. 1.1 shows two of the types of circuit diagrams for a bipolar junction transistor (G = gate, S = source, and D = drain)[1]. A voltage applied at the gate enables a larger current to flow between collector and emitter. There are also *bipolar junction transistors* that use a smaller current to enable a larger current, in this case there is "base", "collector", and "emitter".

With these building blocks, logical gates can be created, but *how* to build these? Which device can be made small and in abundance? And what challenges were encountered on the way.

[1] Magic is a technical term in quantum computing, though we're using it in a colloquial sense here, see [7].

Scale of Modern Computing in your pocket

The iPhone A17 Pro chip's 19 billion transistors would cover an area of about 1 square centimeter. If each transistor were the size of a grain of rice, they would cover an area larger than 75 football fields! This incredible miniaturization is what enables modern computing.

It was recognized early on that semiconductors provided an ideal platform for these kinds of circuits, and much work was done to try and create the above "field effect transistors." Schematically, these take the form:

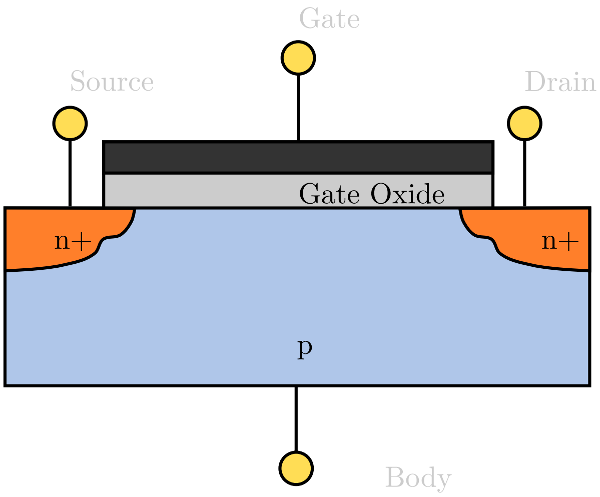
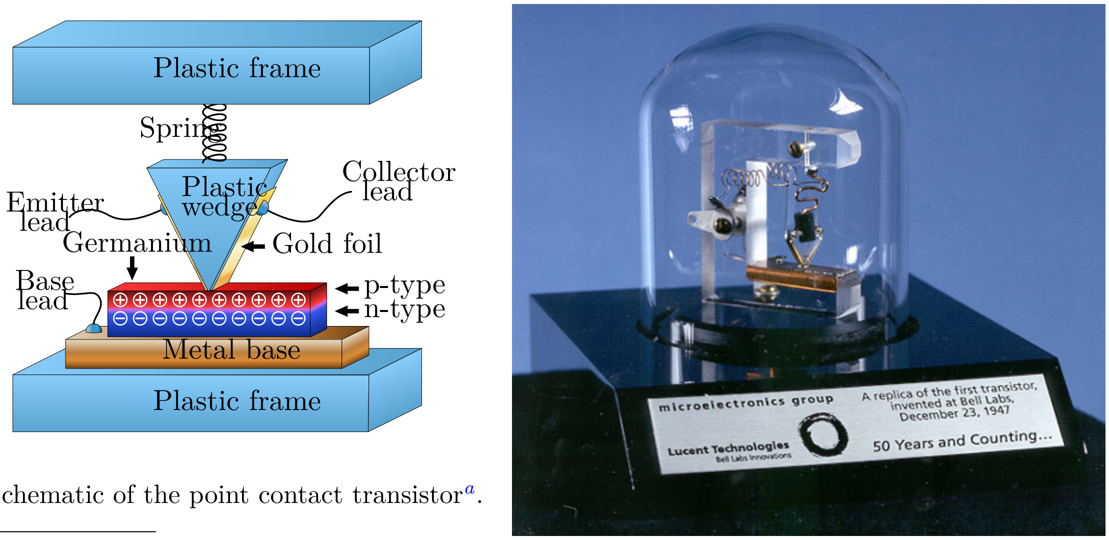


Fig. 1.2: Schematic of Field Effect Transistor[2]

In Fig. 1.2, electrons flow from source to drain, but only when the "gate" has an applied voltage to it (effectively "lowering the barrier" for electrons to get through). This theory was sound and based on the recently developed quantum electron theory of metals developed by Wolfgang Pauli, Werner Heisenberg, Arnold Sommerfeld, Felix Bloch, and Rudolf Peierls [8]. And indeed, the people at Bell labs worked on this problem theoretically and experimentally for the beginning in the 1930s (for a full history, see [8]). Despite the strong foundations though, creating a field effect transistor turned out to be difficult, and in the process Brattain and Schockley instead created the point-contact transistor.



(a) Schematic of the point contact transistor[a].

[a] Licensed under Creative Commons CC0 1.0 Universal Public Domain Dedication.

(b) Replica of first transistor

Fig. 1.3: The point contact transistor.

1.2.2 The Point Contact Transistor

The point-contact transistor, invented in 1947, worked quite differently from the field-effect design. Instead of using a voltage at a gate to control current flow, it used two very closely spaced metal contacts pressed against a semiconductor (typically germanium). One contact, called the emitter, would inject positive charge carriers (holes) into the semiconductor. The second contact, called the collector, would collect these carriers - but crucially, the amount of current that could flow through the collector could be controlled by small changes in the emitter current[3]. A schematic and image of a replica of the original device are illustrated in Fig. 1.3.

This amplification effect, where a small current controls a larger one, was revolutionary–though the exact physics behind it wasn't fully understood at the time. The key was that the metal contacts created special regions in the semiconductor where the positive carriers modified the barrier for current flow from the bulk material. While the detailed quantum mechanics is complex, you can think of it like creating "paths" that electrons prefer to take through the material, with the emitter current controlling how easily electrons can flow along these paths to the collector.

While point-contact transistors were eventually superseded by more reliable and easier-to-

[2] In some literature, these are matrices are denoted by σ_x, σ_y, and σ_z and related to spin operators via $S_i = \frac{1}{2}\sigma_i$. This insight can help bridge the idea of these operators and the Bloch sphere.

[3] For details on how this was made, see How the first transistor worked

manufacture designs, they represented a crucial breakthrough in electronics. They proved that solid-state devices could indeed amplify electrical signals. However, they were quite large. The original design for a field effect transistor would be needed, and they key resided in understanding and control the *surface physics* of semiconductors.

1.2.3 Surface physics and transistors

The key challenge in creating field effect transistors lay in understanding and controlling the surface properties of semiconductors. To understand why this was so difficult, let's break it down:

When a semiconductor crystal (like silicon) ends at a surface, something interesting happens. The regular pattern of atoms is suddenly interrupted - imagine a neat stack of blocks suddenly ending in mid-air. This interruption creates what we call "surface states" - special energy levels that electrons can occupy right at the surface of the material.

These surface states turned out to be extremely problematic for making transistors. Remember that in a field effect transistor, we want to control the flow of electrons using an electric field from the gate (see Fig. 1.2). However, these surface states acted like tiny electron traps, capturing and holding onto electrons. When electrons got stuck in these states, they effectively "screened" or blocked the electric field from the gate, preventing it from controlling the current flow through the semiconductor.

This screening effect was so strong that early attempts at field effect transistors simply didn't work–no matter how strong a voltage was applied to the gate, it couldn't effectively control the current flow. It was like trying to control a water flow with a valve, but having something constantly blocking the valve from moving.

The breakthrough came in the 1950s when researchers, particularly at Bell Labs, realized they needed to chemically "passivate" the semiconductor surface–essentially finding ways to neutralize these problematic surface states. The key discovery was that growing a thin layer of silicon dioxide (SiO) on silicon created a much more stable interface with far fewer problematic surface states. This oxide layer also served as an excellent insulator between the gate and the semiconductor.

This seemingly simple solution–growing an oxide layer–was actually a remarkable achievement that required precise control of material chemistry and manufacturing processes. It finally allowed the creation of practical field effect transistors, leading to the modern MOSFET (Metal-Oxide-Semiconductor Field Effect Transistor) that forms the backbone of today's electronics.

The success of this approach also highlights an important lesson in technology development: sometimes the biggest breakthroughs come not from changing the fundamental design, but from finding ways to control and manage the subtle physical effects that prevent a good design from working in practice.

1.2.4 Where are we with quantum computing?

Imagine that this course existed back in the 1950s and we called it “Computing Technology.” Transistors were still coming online to enable computation at scale and we had both the information science and material theory to achieve it:

1. We knew what was needed to do universal classical computation.
2. We had the quantum theory of metals to describe how to build components (transistors) to achieve classical computation.

However, the engineering challenges took decades to resolve. It was only when we resolved those that computation as we currently envision it took off and we could have classical computers.

For quantum computing, we are in a similar situation:

1. We know what is needed to do universal quantum computation.
2. We have the quantum theory of photons, atoms, and superconductivity to achieve quantum computation.

However, as we will see in what follows, we have significant engineering challenges to achieve these in practice. While we will be largely concerned with the physics that make #2 possible in this course (and we’ll touch on #1 for the first part of the course), we will pay attention to the strengths and weaknesses in the physics that lead to more pressing engineering challenges.

> ℹ Historical Parallel
>
> Just as the theory of classical computation [6] preceded practical computers by decades, we now have the theory of quantum computation [5] but face significant engineering challenges. The key difference is that we’re trying to control individual quantum systems rather than classical electrical currents.

1.3 Review of the Postulates of Quantum Mechanics

Quantum mechanics is built out of some basic postulates which are crucial to understand for quantum computation. When we talk about a “system” in these postulates, we will be thinking of two-level systems which we can label 0 or 1 (for our qubit). However, we will state them generally since many applications require some basic work to reduce the complicated system down to just the qubits we are interested in.

❗ Mathematical Notation Guide

Throughout this text, we'll use:

- $|\psi\rangle$ ("ket psi"): Represents a quantum state
- $\langle\phi|$ ("bra phi"): The dual vector to $|\phi\rangle$
- $\langle\phi|\psi\rangle$: The inner product between states
- $\otimes$: The tensor product operation

These notations provide a compact way to describe quantum systems.

1.3.1 Postulate I: The Hilbert space

> A state in an isolated physical system S can be described by a set of *normalized state vectors*–$|\psi\rangle$ and all vectors related by a phase $|\psi'\rangle = e^{i\phi}|\psi\rangle$–belonging in the Hilbert space $\mathcal{H}_S$.

importantly, a Hilbert space is equipped with an inner product (much like the dot product in three-dimensions) $\langle\alpha|\beta\rangle$.

We also need rules for attaching Hilbert spaces to one another. Afterall, we will need to use more than one qubit to do anything interesting.

> The Hilbert space of a composite system is the *tensor product* of the two individual systems $\mathcal{H}_{AB} = \mathcal{H}_A \otimes \mathcal{H}_B$.

1.3.2 Postulate II: Physical Observables and Measurements

This postulate is also sometimes called the **Born rule**.

In order to measure the system (e.g., "where is the particle?" or "Is the qubit a 0 or 1?"), we need to know what *physical observables* are

> (i) Every physical observable a can be described as a Hermitian operator A acting in the Hilbert Space.

Formally, a Hermitian operator has $A = A^\dagger$ where $A^\dagger$ is the *conjugate transpose* of A. These operators have a whole set of orthonormal eigenstates $A|a_n\rangle = a_n|a_n\rangle$ for a *real* number a_n (orthonormal means $\langle a_n|a_m\rangle = \delta_{nm}$). These mathematical details are important for how we will perform measurements

> (ii) When a physical observable with operator A is measured on a normalized eigenstate $|\psi\rangle$, the result is an eigenvalue a_n of that operator with probability $p_n = |\langle a_n|\psi\rangle|^2$ (or in the case of a degeneracy d, $p_n = \sum_{i=1}^{d} |\langle a_n, i|\psi\rangle|^2$).

If we measure A repeatedly, we are naturally lead to the expectation value $\langle\psi|A|\psi\rangle = a = \sum_n a_n p_n$, the average result of repeated quantum mechanical measurements. Once a measurement is performed, however, the state is changed, for this we need an operator P_n which projects onto the eigenspace of A associated with the eigenvalue a_n.

(iii) When a measurement of the observable A gives a results a_n, the state is changed to be the *normalized projection* of $|\psi\rangle$ to the eigenspace associated with a_n

$$|\psi\rangle \quad \Longrightarrow \quad \frac{P_n |\psi\rangle}{\sqrt{\langle\psi|P_n|\psi\rangle}}$$

If the eigenstates are not degenerate, then $|\psi\rangle \implies |a_n\rangle$. However, we will find that we will often have degenerate states, and in that case

$$|\psi\rangle \implies \frac{\sum_{i=1}^{d} \langle a_n, i|\psi\rangle \, |a_n, i\rangle}{\sqrt{\sum_{i=1}^{d} |\langle a_n, i|\psi\rangle|^2}}.$$

One comfortable with the braket notation might notice that within this, $P_n = \sum_{i=1}^{d} |a_n, i\rangle \langle a_n, i|$.

1.3.3 Postulate III: Time-evolution of a system

While we will talk about Hamiltonians, often in quantum computation we have gates that do not require these. In this case, we state this postulate as abstractly as possible

> The time evolution of a closed system from some time t_0 and state $|\psi_0\rangle$ to a final time t and state $|\psi\rangle$ can be described as a unitary transformation
>
> $$|\psi\rangle = U(t, t_0) |\psi_0\rangle .$$

This postulate is necessary for us to maintain probabilities. Unitary operators have the property that $UU^\dagger = U^\dagger U = \mathbb{1}$, and so

$$1 = \langle\psi_0|\psi_0\rangle = \langle\psi_0|U^\dagger U|\psi_0\rangle = \langle\psi|\psi\rangle .$$

In the case of time-independent *Hamiltonian dynamics,* the operator takes the form $U = e^{-iHt/\hbar}$ for a hermitian energy operator H called the *Hamiltonian.*

1.4 A Brief History of Computing: From Classical to Quantum

The concept that information is physical underlies both classical and quantum computation. Even in the earliest systems of record-keeping, we see physical objects encoding information:

- Kish Tablet (3500 BCE): A limestone tablet from Kish showing a record of pictographic writing.
- Quipu (2600–1900 BCE): A system of knotted ropes used by the Inca civilization for keeping records. The color, order, and number of knots all represented quantifiable or categorical data.

These examples highlight that from the very beginning, the act of storing and manipulating information has always had a physical basis—though the underlying physics often remained implicit for centuries.

1.4.1 Classical Computing Foundations

The paradigm of classical computing rests on a few fundamental ideas:

- **Boolean Logic & Universal Gates**: All classical computers can be built from a finite set of universal logical gates (e.g., {NAND}, {NOR}, or {AND, NOT}).

Most Boolean operations (AND, OR, NAND, etc.) are inherently irreversible: once a bit is erased or overwritten, the original state cannot be recovered from the output alone.

- **Extended (Physical) Church–Turing Thesis**: A probabilistic Turing machine can efficiently simulate any realistic physical model of computation with at most polynomial overhead.

This thesis, while unproven, underpins the expectation that classical computers (or at least classical models) are sufficient for simulating any physical system in principle. Quantum computation potentially challenges this with a polynomial in time algorithm (Shor's algorithm) that is exponential in time classically.

1.4.2 The Emergence of Quantum Information

While classical computing relies on bits that are strictly 0 or 1, quantum computing introduces powerful new concepts:

- **Superposition**: A quantum bit (qubit) can be in a linear combination of basis states (e.g., simultaneously "0" and "1" with certain complex amplitudes).
- **Entanglement**: Two or more qubits can become correlated in such a way that measuring one affects the outcomes for the others, even across vast distances.

The key question that launched the field of quantum information was whether these uniquely quantum properties—superposition and entanglement—could be exploited to perform computations more efficiently than any classical device.

In his seminal work, Simulating Physics with Computers [9], Richard Feynman observed that simulating quantum many-body systems on classical computers seems to require exponential resources. He posed the idea of harnessing genuine quantum systems themselves for simulation, planting the seeds for quantum computation as a research field.

Then, in a series of groundbreaking results between the 1980s and 1990s, researchers demonstrated that quantum computers could, in principle, outperform classical computers for certain tasks:

1. Deutsch (1985) [10]: Showed that it is possible to carry out a simple computational task on a quantum computer faster than any classical algorithm.
2. Deutsch–Jozsa (1992) [11]: Introduced a deterministic quantum algorithm that is exponentially faster (in the worst case) than any deterministic classical algorithm.
3. Bernstein–Vazirani (1992) [12]: Demonstrated a probabilistic quantum algorithm faster than any probabilistic classical algorithm.
4. Simon (1994) [13]: Provided a probabilistic quantum algorithm that is exponentially faster than any probabilistic classical algorithm for a specific promise problem.
5. Shor (1994) [14]: Showed how to factor integers efficiently, providing an exponential speedup over the best known classical methods. This result was particularly striking for cryptography, as factoring large numbers underpins many encryption schemes.

Alongside these algorithmic milestones, researchers began to delineate the limitations: not every problem can be exponentially sped up by quantum methods. For instance, Grover's algorithm (1996) [15] for unstructured search yields a quadratic speedup (from N to $\sqrt{N}$)—still better than classical, but not the exponential leap that Shor's algorithm provides for factoring.

1.4.3 Analog vs. Digital Quantum Simulation

As the field grew, quantum simulation branched into two approaches:

- Analog Quantum Simulation: Uses a controllable quantum system to mimic a target quantum system. The interactions in the simulator closely resemble the interactions in the system of interest.
- Digital Quantum Simulation: Decomposes a quantum evolution into a sequence of discrete gates (a "universal" set of quantum gates), akin to how classical digital computers function using logical gate operations.

Both approaches aim to exploit quantum mechanics to tackle problems in mathematics, physics, chemistry, and materials science that remain intractable for classical supercomputers.

Amid ongoing research, the interplay between classical and quantum paradigms remains a vibrant area of exploration. While classical computing infrastructure continues to be indispensable, quantum computing offers the promise of qualitatively new capabilities—provided we can tame the noise, errors, and fragilities inherent to quantum states.

1.5 Quantum Technologies

One major difference between the hindsight-history we have for classical computation and the current state of quantum computers is that are many platforms vying to enable quantum computation. There are arguments for and against each platform, and even arguments for using a combination of platforms. Here we give a list of some of the technologies and highlight the ones we will be surveying in this course.

Platforms covered in this course

- **Superconducting qubits**: Artificial atoms made from superconducting circuits that operate at ultra-low temperatures. Currently the most mature platform, used by companies like IBM and Google.
- **Trapped ions**: Individual atoms held in place by electromagnetic fields. Known for having very long coherence times and high-fidelity gates. Major players include IonQ and Quantinuum (formerly of Honeywell).
- **Photonic quantum computers**: Use particles of light (photons) as qubits. Can operate at room temperature and naturally interface with quantum communication systems. Being developed by companies like PsiQuantum and Xanadu.

Other platforms we will not cover

- **Silicon quantum dots**: Quantum bits made from individual electrons trapped in silicon, similar to classical semiconductor technology. Could potentially leverage existing manufacturing processes.
- **Neutral atoms and Rydberg arrays**: Individual neutral atoms arranged in arrays using laser beams. When excited to high-energy Rydberg states, atoms can interact strongly with their neighbors. Can create large numbers of identical qubits with programmable interactions. Companies like QuEra are pursuing this approach.
- **NV centers**: Quantum bits made from nitrogen-vacancy defects in diamond. Can operate at room temperature and have long coherence times, making them particularly promising for quantum sensing and networking applications.

A platform we will cover if time permits

- **Topological qubits**: A theoretical approach that would use special quantum states of matter to create error-protected qubits. Still in early research stages but could offer significant advantages if realized.

⚠ Current State of Quantum Computing

As of 2024, the largest quantum computers have around 50-100 physical qubits optimistically, but these are noisy and require error correction. For comparison, your smartphone has billions of classical bits. This highlights the early stage of quantum computing development and the engineering challenges ahead.

Each platform has its own advantages and challenges in terms of scalability, error rates, coherence times, and manufacturing complexity. The field is still evolving, and it's possible that different platforms may be optimal for different applications.

2 The Qubit

The fundamental building block of the classical computer was the bit: A 0 or 1 that could be manipulated by a classical computer (via transistors, see Section 1.2). In a similar manner, quantum computation has the "quantum bit" or just *qubit*, for short. This leads us to the linear algebra of 2 × 2 matrices, as we will see. Despite the apparent simplicity, we can already see many of the key features of quantum mechanics in this simple system.

2.1 The qubit Hilbert space

A qubit will be in two-dimensional complex vector space equipped with an inner product.

2.1.1 Qubit states

For the qubit, we associate two states with two different basis vectors: $|0\rangle$ and $|1\rangle$. This will be called the *computational basis*. The magic[1] of quantum mechanics is that a state need not be just one or the other, but could be *any* linear superposition of these

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,. \tag{2.1}$$

In this, we have adopted the *bra-ket* notation due to Dirac. While it can be quite useful, we can write this in terms of matrices and vectors

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

This makes it clear that these states are *orthogonal* $\langle 0|1\rangle = 0$.

In this case we have

$$|\psi\rangle = \begin{bmatrix}\alpha\\\beta\end{bmatrix}.$$

We also need the conjugate transpose, the Hermitian conjugate, of this vector, which will be a row-vector

$$\langle\psi| = \begin{bmatrix}\alpha^* & \beta^*\end{bmatrix}.$$

[1] Magic is a technical term in quantum computing, though we're using it in a colloquial sense here, see [7].

The key feature of quantum mechanics is that these states must be *normalized*, meaning that the probability of finding the system in any state must sum to 1. This means that

$$\langle\psi|\psi\rangle = \begin{bmatrix}\alpha^* & \beta^*\end{bmatrix}\begin{bmatrix}\alpha \\ \beta\end{bmatrix} = |\alpha|^2 + |\beta|^2 = 1.$$

In matrix notation, this is just the dot product of a vector with its complex conjugate.

ℹ Why Normalization Matters

The normalization condition $|\alpha|^2 + |\beta|^2 = 1$ isn't just mathematical convenience - it ensures probabilities add up to 100%! For example:

- $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ gives 50-50 chance of measuring 0 or 1
- $|\psi\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle$ gives 75% chance of 0 and 25% chance of 1

We can also write operators that act on these states. The simplest operator is the Pauli Z operator, which in matrix form is

$$Z = \begin{bmatrix}1 & 0 \\ 0 & -1\end{bmatrix}.$$

When this operator acts on our basis states, we find

$$Z|0\rangle = |0\rangle, \quad Z|1\rangle = -|1\rangle.$$

This means that $|0\rangle$ and $|1\rangle$ are *eigenstates* of Z with eigenvalues $+1$ and -1 respectively. For a general state $|\psi\rangle$, measuring Z will yield either $+1$ or -1, with probabilities determined by $|\alpha|^2$ and $|\beta|^2$ respectively.

Example: Measuring a superposition state

Consider the state $|\psi\rangle = \frac{3}{5}|0\rangle + \frac{4}{5}|1\rangle$. When we measure this state in the Z basis: Notice that this state is normalized since $(\frac{3}{5})^2 + (\frac{4}{5})^2 = 1$. If we measure this state in the computational basis:

- We'll get outcome $|0\rangle$ with probability $|\frac{3}{5}|^2 = 0.36$ (36%)
- We'll get outcome $|1\rangle$ with probability $|\frac{4}{5}|^2 = 0.64$ (64%)

After measurement, the state will collapse to either $|0\rangle$ or $|1\rangle$ with the above probabilities

⚠ Measurement Collapse in Practice

When we say a quantum state "collapses" upon measurement, what actually happens in the lab?

- For a superconducting qubit: We measure a voltage or current
- For an ion trap: We detect scattered photons
- For a photonic qubit: We count photons with a detector

Each technology has its own way of converting quantum information into classical signals!

2.1.2 Qubit operators

Since this is linear algebra, we can write a general operator $\mathcal{O}$ as a matrix

$$\mathcal{O} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

Often, we are interested in the eigenvalues and eigenstates of these operators $\mathcal{O}\,|\psi_i\rangle = \lambda_i\,|\psi_i\rangle$. Generically, we can find these by solving a polynomial equation

$$\det(\mathcal{O} - \lambda I) = 0.$$

Solving this step-by-step

$$\det(\mathcal{O} - \lambda I) = \begin{vmatrix} a-\lambda & b \\ c & d-\lambda \end{vmatrix} = 0,$$

which gives us

$$(a-\lambda)(d-\lambda) - bc = 0.$$

This is a quadratic equation that we can solve:

$$\lambda^2 - (a+d)\lambda + (ad - bc) = 0.$$

The eigenvalues are therefore

$$\lambda_\pm = \frac{a + d \pm \sqrt{(a-d)^2 + 4bc}}{2}. \tag{2.2}$$

For quantum mechanical **observables** (see Section 1.3.2), we are particularly interested in *Hermitian* operators where $\mathcal{O} = \mathcal{O}^\dagger$,

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a^* & c^* \\ b^* & d^* \end{bmatrix}$$

This means that a and d must be real and $c = b^*$. In this case, the eigenvalues are always real, as we can see from the Eq. 2.2.

A particularly important class of operators are **unitary operators**, where $U^\dagger U = UU^\dagger = I$. These are what we use for time-evolution, see Section 1.3.3.

These operators preserve the inner product between states:

$$\langle U\psi|U\phi\rangle = \langle\psi|\phi\rangle$$

For a 2×2 matrix

$$U = \begin{bmatrix} a & b \\ c & d \end{bmatrix},$$

the unitary condition means that

$$\begin{bmatrix} a^* & c^* \\ b^* & d^* \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

This gives us several conditions:

1. $|a|^2 + |c|^2 = 1$ (normalization of first column)
2. $|b|^2 + |d|^2 = 1$ (normalization of second column)
3. $ab^* + cd^* = 0$ (orthogonality of columns)

This immediately gives us some insight into these operators. If we define,

$$|\psi_1\rangle = \begin{bmatrix} a \\ c \end{bmatrix}, \quad |\psi_2\rangle = \begin{bmatrix} b \\ d \end{bmatrix},$$

then we have $\langle\psi_1|\psi_1\rangle = 1 = \langle\psi_2|\psi_2\rangle$ and $\langle\psi_1|\psi_2\rangle = 0$.

An important property of unitary operators is that their eigenvalues always have magnitude 1, meaning they can be written as $e^{i\theta}$ for some real θ. This makes them natural operators for describing quantum evolution.

Why Unitary?

Unitary operators are special because they:

1. Preserve the normalization of quantum states
2. Are reversible (have an inverse)
3. Represent physical operations that conserve probability

This is why quantum gates must be unitary - they represent real physical processes that can be undone!

2.1.3 The Pauli operators

A particularly important set of operators are the Pauli operators. We've already seen the Pauli Z operator. The other two are[2]

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}.$$

These operators satisfy some important algebraic relations:

$$X^2 = Y^2 = Z^2 = I, \quad XY = iZ, \quad YZ = iX, \quad ZX = iY.$$

We can additionally start to see some logical operations begin to appear; X operates on the computational basis as a **NOT** gate

$$X\left|0\right\rangle = \left|1\right\rangle, \quad X\left|1\right\rangle = \left|0\right\rangle.$$

Pauli Operators in Action

The Pauli operators represent quantum operations:

- X is like the classical NOT gate: flips between $\left|0\right\rangle$ and $\left|1\right\rangle$
- Z adds a phase: leaves $\left|0\right\rangle$ alone but negates $\left|1\right\rangle$
- $Y = iXZ$ combines both operations.

These simple operations are building blocks for more complex quantum algorithms!

Example: Applying operators

Let's apply the X (NOT) gate to our state $\left|\psi\right\rangle = (\frac{3}{5}\left|0\right\rangle + \frac{4}{5}\left|1\right\rangle)$:

$$\begin{aligned} X\left|\psi\right\rangle &= X(\tfrac{3}{5}\left|0\right\rangle + \tfrac{4}{5}\left|1\right\rangle) \\ &= \tfrac{3}{5}X\left|0\right\rangle + \tfrac{4}{5}X\left|1\right\rangle \\ &= \tfrac{3}{5}\left|1\right\rangle + \tfrac{4}{5}\left|0\right\rangle \\ &= \tfrac{4}{5}\left|0\right\rangle + \tfrac{3}{5}\left|1\right\rangle \end{aligned}$$

The full set of Pauli operators, along with the identity, form a complete basis for 2 × 2 matrices, meaning we can write any operator as

$$\mathcal{O} = aI + bX + cY + dZ,$$

[2] In some literature, these are matrices are denoted by σ_x, σ_y, and σ_z and related to spin operators via $S_i = \frac{1}{2}\sigma_i$. This insight can help bridge the idea of these operators and the Bloch sphere.

where a, b, c, and d are complex numbers. We can extract each of these numbers, mathematically, with a trace operation

$$\operatorname{tr}\mathcal{O} = 2a, \quad \operatorname{tr}\mathcal{O}X = 2b, \quad \operatorname{tr}\mathcal{O}Y = 2c, \quad \operatorname{tr}\mathcal{O}Z = 2d.$$

Note that separately, X, Y, and Z are Hermitian (and thus, observables). If $\mathcal{O}$ is an observable, then $\mathcal{O} = \mathcal{O}^\dagger$ immediately leads us to a, b, c, and d being all real.

We can put constraints on these coefficients for unitary operators as well, and we leave this as an exercise for the reader.

Finally, these operators have eigenstates as well, and we can define them as $X\left|\pm\right\rangle = \pm\left|\pm\right\rangle$ and $Y|\pm i\rangle = \pm\left|\pm i\right\rangle$, and they have the forms

$$\begin{aligned}\left|\pm\right\rangle &= \tfrac{1}{\sqrt{2}}(\left|0\right\rangle \pm \left|1\right\rangle),\\ \left|\pm i\right\rangle &= \tfrac{1}{\sqrt{2}}(\left|0\right\rangle \pm i\left|1\right\rangle).\end{aligned}$$

We will often want to change our basis from $\left|0\right\rangle$ and $\left|1\right\rangle$ to $\left|+\right\rangle$ and $\left|-\right\rangle$. This is accomplished with something called the *Hadamard gate* (we'll call it H, not to be confused with a Hamiltonian) and it is created specifically to change from computational basis to the X basis: $H\left|0\right\rangle = \left|+\right\rangle$ and $H\left|1\right\rangle = \left|-\right\rangle$. As a matrix it takes the form

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

! The Power of Hadamard

The Hadamard gate is one of the most important gates in quantum computing with useful properties:

1. It creates equal superpositions from computational basis states.
2. It's its own inverse ($H^2 = I$)
3. It's used in nearly every quantum algorithm
4. When applied to n qubits, it creates a superposition of all 2^n possible bit strings!

Example 3: The Hadamard Transform

The Hadamard gate is particularly important because it creates superposition states.

Let's see what happens when we apply it to $|0\rangle$:

$$\begin{aligned} H|0\rangle &= \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ &= \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ &= |+\rangle \end{aligned}$$

3 Summary

References

[1] F. Arute et al., Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).

[2] Google Quantum AI and Collaborators et al., Quantum error correction below the surface code threshold, Nature (2024).

[3] E. Pednault, J. A. Gunnels, G. Nannicini, L. Horesh, and R. Wisnieff, Leveraging Secondary Storage to Simulate Deep 54-qubit Sycamore Circuits, arXiv:1910.09534 (2019).

[4] F. Pan, K. Chen, and P. Zhang, Solving the Sampling Problem of the Sycamore Quantum Circuits, Physical Review Letters **129**, 090502 (2022).

[5] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*, Anniversary edition (Cambridge University Press, Cambridge ; New York, 2011).

[6] A. M. Turing, On Computable Numbers, with an Application to the Entscheidungsproblem, Proceedings of the London Mathematical Society **s2-42**, 230 (1937).

[7] S. Bravyi and A. Kitaev, Universal quantum computation with ideal Clifford gates and noisy ancillas, Physical Review A **71**, 022316 (2005).

[8] L. Hoddeson, The Discovery of the Point-Contact Transistor, Historical Studies in the Physical Sciences **12**, 41 (1981).

[9] R. P. Feynman, Simulating physics with computers, International Journal of Theoretical Physics **21**, 467 (1982).

[10] D. Deutsch, Quantum theory, the Church–Turing principle and the universal quantum computer, Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences **400**, 97 (1985).

[11] D. Deutsch and R. Jozsa, Rapid solution of problems by quantum computation, Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences **439**, 553 (1992).

[12] E. Bernstein and U. Vazirani, Quantum Complexity Theory, SIAM Journal on Computing **26**, 1411 (1997).

[13] D. R. Simon, On the Power of Quantum Computation, SIAM Journal on Computing **26**, 1474 (1997).

[14] P. W. Shor, *Algorithms for Quantum Computation: Discrete Logarithms and Factoring*, in *Proceedings 35th Annual Symposium on Foundations of Computer Science* (IEEE Comput. Soc. Press, Santa Fe, NM, USA, 1994), pp. 124–134.

[15] L. K. Grover, *A Fast Quantum Mechanical Algorithm for Database Search*, in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing - STOC '96* (ACM Press, Philadelphia, Pennsylvania, United States, 1996), pp. 212–219.