From Qubits to Quantum Supremacy:

The Linear Algebra That Changed the World

Submitted by: Karan Singh

April 2025

1. Introduction

Every digital device you own operates on one foundational concept: the binary digit. A *bit* is either 0 or 1 — never both, never something in between. Yet in the world of quantum mechanics, this comforting discreteness dissolves. Quantum systems can exist in a superposition of states, allowing new forms of computation that defy our classical intuitions.

At the centre of this computational revolution is the **qubit** — a quantum version of the bit. Unlike its classical counterpart, a qubit can exist in any linear combination of the basis states $|0\rangle$ and $|1\rangle$. This is not just physics — it is mathematics. Deep, rich, and sometimes unsettling mathematics, built on complex vector spaces, unitary operators, and transformations that would feel just as comfortable in a graduate-level linear algebra course as in a physics lab.

This essay explores the mathematical foundations of quantum computing: vector spaces, inner products, tensor products, matrix algebra, the Discrete and Quantum Fourier Transforms, and ultimately, the structure of Shor's algorithm. The goal is to

follow the flow of ideas — from basic postulates to algorithmic supremacy — and demonstrate how quantum computers are, at their heart, mathematical machines.

2. Vector Spaces: Where It All Begins

Let us begin with the mathematical object that underpins everything in quantum theory: the vector space.

A **vector space** V over a field $\mathbb{F}$ is a set equipped with two operations:

1. Vector addition: $+ : V \times V \to V$
2. Scalar multiplication: $\cdot : \mathbb{F} \times V \to V$

These operations satisfy the following axioms (for all $\vec{u}, \vec{v}, \vec{w} \in V$, and all $a, b \in \mathbb{F}$):

(i) $\vec{u} + \vec{v} = \vec{v} + \vec{u}$ (Commutativity)

(ii) $(\vec{u} + \vec{v}) + \vec{w} = \vec{u} + (\vec{v} + \vec{w})$ (Associativity)

(iii) $\exists\, \vec{0} \in V$ such that $\vec{v} + \vec{0} = \vec{v}$ (Zero vector)

(iv) $\forall\, \vec{v} \in V$, $\exists\ -\vec{v} \in V$ such that $\vec{v} + (-\vec{v}) = \vec{0}$

(v) $a(\vec{v} + \vec{w}) = a\vec{v} + a\vec{w}$ (Distributivity I)

(vi) $(a + b)\vec{v} = a\vec{v} + b\vec{v}$ (Distributivity II)

(vii) $a(b\vec{v}) = (ab)\vec{v}$ (Associativity of scalar multiplication)

(viii) $1 \cdot \vec{v} = \vec{v}$ (Multiplicative identity)

In quantum computing, we work primarily in vector spaces over the field $\mathbb{C}$, the complex numbers. That's because quantum amplitudes — the numbers that describe probabilities — are complex by necessity.

The Qubit as a Vector

In the quantum world, the state of a qubit is a unit vector in a 2-dimensional complex vector space $\mathbb{C}^2$. A general state is written as:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad \text{where } \alpha, \beta \in \mathbb{C}, \quad |\alpha|^2 + |\beta|^2 = 1$$

Here, $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, and $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ are the computational basis vectors. The requirement $|\alpha|^2 + |\beta|^2 = 1$ ensures that the total probability of all outcomes sums to 1 — a reflection of the Born rule in quantum mechanics.

The space of all possible qubit states is thus the unit sphere in $\mathbb{C}^2$, modulo a global phase factor. This geometry gives rise to the Bloch sphere — a familiar visualization — but we will focus here on its algebraic structure.

Tensor Products: Scaling Up to Multiple Qubits

Quantum systems scale exponentially. Two qubits live in $\mathbb{C}^2 \otimes \mathbb{C}^2 = \mathbb{C}^4$, three qubits in $\mathbb{C}^8$, and so on. The tensor product of two states $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ and $|\phi\rangle = \gamma|0\rangle + \delta|1\rangle$ is:

$$|\psi\rangle \otimes |\phi\rangle = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle$$

Tensor products allow quantum computers to represent superpositions over many possible classical states simultaneously. This property — often mischaracterized as "parallelism" — is fundamental to quantum computational speedup.

In the next section, we'll introduce the mathematical refinement of vector spaces

that enables us to measure quantum states, calculate inner products, and define unitary evolution.

3. Hilbert Spaces: Where Geometry Meets Probability

To understand quantum mechanics — and by extension quantum computing — we need more than just any vector space. We need a *Hilbert space*, a complete inner product space that allows us to talk about angles, lengths, projections, and most importantly: measurements.

An **inner product** is a function $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{C}$ satisfying:

(i) $\langle \phi, \psi \rangle = \overline{\langle \psi, \phi \rangle}$

(ii) $\langle a\phi + b\chi, \psi \rangle = a\langle \phi, \psi \rangle + b\langle \chi, \psi \rangle$

(iii) $\langle \phi, \phi \rangle \geq 0$, with equality iff $\phi = 0$

The norm of a state $|\psi\rangle$ is defined as $\|\psi\| = \sqrt{\langle \psi | \psi \rangle}$. In quantum mechanics, all state vectors are normalized, meaning $\|\psi\| = 1$. This ensures that the total probability of all outcomes adds up to 1.

Bra-Ket Notation

We use Dirac's *bra-ket* notation:

$$\langle \phi | \psi \rangle \quad \text{is the inner product of } |\phi\rangle \text{ and } |\psi\rangle$$

Kets (like $|\psi\rangle$) are column vectors, while bras (like $\langle \psi|$) are the conjugate transpose

(row vectors):

$$|\psi\rangle = \begin{pmatrix} \alpha \\ \\ \beta \end{pmatrix}, \quad \langle\psi| = (\overline{\alpha}, \overline{\beta})$$

This formalism is not just aesthetic — it allows us to define projection operators, probabilities of outcomes, and transformations in a highly compact form.

Orthogonality and Measurement

Two quantum states $|\phi\rangle$ and $|\psi\rangle$ are orthogonal if $\langle\phi|\psi\rangle = 0$. In quantum computation, orthogonality means *perfect distinguishability*. That is, if two states are orthogonal, a measurement can tell them apart with certainty.

The probability of measuring state $|\psi\rangle$ and getting outcome associated with $|\phi\rangle$ is given by:

$$P = |\langle\phi|\psi\rangle|^2$$

This is a manifestation of the Born Rule — a purely mathematical law at the heart of quantum mechanics.

Completeness

A Hilbert space is *complete*, which means that every Cauchy sequence of vectors converges to a limit within the space. This property is important when considering infinite-dimensional spaces (such as those used in quantum field theory), but in quantum computing we usually work with finite-dimensional Hilbert spaces like $\mathbb{C}^{2^n}$.

Next, we'll look at how quantum gates are built from special types of matrices called **unitary operators**, and why these are essential for quantum evolution.

4. Quantum Gates: The Algebra of Evolution

In classical computing, gates like AND, OR, and NOT manipulate bits deterministically. In quantum computing, evolution is governed by **unitary matrices** — linear transformations that preserve the norm of state vectors and thus conserve probability.

Unitary Operators

A matrix $U \in \mathbb{C}^{n\times n}$ is unitary if:

$$U^\dagger U = UU^\dagger = I$$

where $U^\dagger$ is the conjugate transpose of U, and I is the identity matrix. This condition ensures that the transformation is reversible: no information is lost, and the vector norm is preserved.

Quantum evolution — whether by gate application or time evolution via Schrödinger's equation — is always represented by a unitary transformation. This is a direct consequence of the linear and norm-preserving structure of Hilbert space.

Important Single-Qubit Gates

Let us define some key gates mathematically:

- **Pauli-X (NOT gate)**:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

  This flips $|0\rangle \leftrightarrow |1\rangle$, analogous to classical NOT.

- **Hadamard (H) gate**:

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

  This creates superpositions: $H|0\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$

- **Pauli-Z (phase flip)**:

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

- **Phase Gate (S)**:

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$$

Each of these gates acts as a specific rotation or reflection on the Bloch sphere. The set $\{H, S, T, X, Z\}$ forms a basis from which universal computation can be built.

Multi-Qubit Gates

The Controlled-NOT (CNOT) gate is defined on two qubits and introduces entanglement — a uniquely quantum phenomenon where the state of one qubit cannot be described independently of the other.

The matrix form of CNOT (on computational basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$) is:

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

CNOT, combined with single-qubit gates, is sufficient for universal quantum compu-

tation.

5. The Discrete Fourier Transform (DFT)

The Discrete Fourier Transform is a linear transformation that takes a vector $x = (x_0, x_1, \ldots, x_{N-1}) \in \mathbb{C}^N$ and returns a vector $X = (X_0, X_1, \ldots, X_{N-1})$ defined by:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-2\pi ikn/N}$$

This transformation decomposes a signal into its frequency components. It is linear and invertible, with inverse:

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{2\pi ikn/N}$$

In matrix form, the DFT is represented as multiplication by the Fourier matrix $F \in \mathbb{C}^{N\times N}$, with entries:

$$F_{jk} = \frac{1}{\sqrt{N}} e^{-2\pi ijk/N}$$

The DFT has complexity $O(N^2)$, but with the Fast Fourier Transform (FFT), this is reduced to $O(N \log N)$.

6. The Quantum Fourier Transform (QFT)

The QFT is the quantum analogue of the DFT, acting on quantum superpositions of computational basis states. Given a state:

$$|\psi\rangle = \sum_{x=0}^{N-1} a_x |x\rangle$$

The QFT maps it to:

$$\text{QFT}|\psi\rangle = \sum_{k=0}^{N-1} \left(\frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} a_x e^{2\pi i x k/N} \right) |k\rangle$$

Efficiency of the QFT

The QFT can be implemented with only $O(n^2)$ quantum gates, where $n = \log_2 N$. This exponential speedup (compared to classical FFT's $O(N \log N)$) is one of the key reasons quantum computers outperform classical ones in certain domains.

Circuit Construction

The QFT circuit consists of:

- A sequence of Hadamard gates
- Controlled phase rotations $R_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{pmatrix}$
- A reversal of qubit order at the end

These gates are applied conditionally to different qubits, with phase information accumulating across the circuit. The resulting operation is both unitary and reversible — satisfying the physical requirements of quantum mechanics.

Next, we explore Shor's algorithm — the poster child of quantum computation — and show how the QFT is the engine that breaks modern encryption.

7. Shor's Algorithm: Factoring with Quantum Periodicity

Shor's algorithm (1994) is arguably the most famous quantum algorithm, celebrated for its ability to factor integers in polynomial time — something no known classical algorithm can do efficiently. At its heart lies the mathematics of periodicity, the structure of modular arithmetic, and the efficiency of the Quantum Fourier Transform.

Suppose we want to factor an integer N. Classically, we could try trial division up to $\sqrt{N}$, but this is slow for large N. Shor's algorithm, on the other hand, uses a quantum computer to solve the following problem:

The Order-Finding Problem

Given integers a and N, where $\gcd(a, N) = 1$, find the smallest $r \in \mathbb{N}$ such that:

$$a^r \equiv 1 \pmod{N}$$

This value r is called the *order* of $a \mod N$, and once found, factoring becomes surprisingly easy. Specifically, if r is even and $a^{r/2} \not\equiv -1 \mod N$, then:

$$\gcd(a^{r/2} \pm 1, N)$$

produces a non-trivial factor of N.

Quantum Subroutine: Period Finding

The key quantum subroutine is to use superposition to compute all values of $f(x) = a^x \mod N$ in parallel, and then use the QFT to extract the period r. Here's how:

1. Prepare a superposition of all integers x from 0 to $2^n - 1$

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$$

2. Compute $f(x) = a^x \mod N$ into a second register

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle|f(x)\rangle$$

3. Measure the second register — this collapses the state to a uniform superposition over all x that map to a specific value of $f(x)$.

4. Apply the QFT to the first register to reveal the period r via interference.

5. Use continued fractions to extract r from the measurement.

The algorithm runs in polynomial time: $O((\log N)^3)$. On a sufficiently large quantum computer, it could break RSA — the encryption system used in most internet communication — by factoring 2048-bit numbers in minutes.

8. Conclusion

Quantum computing is not built on science fiction or mysticism — it is built on pure mathematics. From complex vector spaces to unitary matrices, from inner products to Fourier transforms, every step of a quantum algorithm is grounded in rigorous mathematical logic.

We began with vector spaces, generalised them to Hilbert spaces, introduced unitary operators, and climbed through the algebra of quantum gates. We explored the Fourier Transform, showed how its quantum cousin powers exponential speedups, and culminated

with Shor's algorithm — a practical application that has real-world consequences for cybersecurity.

In essence, quantum computation is linear algebra on steroids. But it's not magic. It's just a different kind of machine, obeying different mathematical rules. And if we understand those rules, we can build — and break — the future.

References

1. Nielsen, M. A., & Chuang, I. L. (2010). *Quantum Computation and Quantum Information*. Cambridge University Press.

2. Preskill, J. (1998). *Lecture Notes for Physics 229: Quantum Information and Computation*. `http://theory.caltech.edu/~preskill/ph229/`

3. Aaronson, S. (2013). *Quantum Computing Since Democritus*. Cambridge University Press.

4. MIT OpenCourseWare. *Quantum Computation*. `https://ocw.mit.edu`

5. Shor, P. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing, 26*(5), 1484–1509.

6. Watrous, J. (2018). *The Theory of Quantum Information*. Cambridge University Press.

7. Lidar, D. A., & Brun, T. A. (Eds.). (2013). *Quantum Error Correction*. Cambridge University Press.