Linear Algebra of Quantum Mechanics and the simulation of a Quantum Computer

Dhruva Sambrani - MS18163

June-August 2019

1 Abstract

This Summer project is an introductory reading to the Mathematical Structures that underlie Quantum Mechanics and then take it forward to a theoretical introduction to Quantum Information and Quantum Computing. Quantum Computers will bring about a paradigm shift in Cryptography and Scientific Computing[1]. Keeping this in mind, learning Quantum Computing is of paramount importance to help contribute to the field.

Once the basics of Quantum Computing are understood, we also attempt to simulate a Quantum Computer on a Classical computer and making a Julia[2] module for the same.

This paper serves as a Very Brief Introduction to Quantum Computing and documentation of the code in the form while publishing.

2 An Introduction to Quantum Mechanics [6]

We first explore certain very famous experiments which cannot be explained by Classical Mechanics and require a "new Physics" to explain them.

1. Young's Double Slit Experiment for Electrons
2. Crystal Diffraction
3. Existence of Finitely Sized Atoms

To explain these results, we introduce the concept of *Probability Amplitudes*, and model particles as also having a wave-like nature. Mathematically, we also introduce the Dirac Notation as a notation for the probability of an event happening and then explore the arithmetic of the Probability Densities. With this understanding in our hand, we attempt to explain the above experiments as a set of probabilistic events. We also visit the identical Particle Collision problem and are now able to explain the results, which was in disagreement with Classical predictions.

3 Mathematical Quantum Mechanics: The Linear Algebra of QM[3]

With the Schrödinger's Equation,

$$\imath\hbar\frac{\partial\Psi(\vec{r},t)}{\partial t} = \left[-\frac{\hbar^2}{2m}\nabla^2 + V(x)\right]\Psi(\vec{r},t) \tag{1}$$

taken ad-hoc, we now begin to define Quantum Mechanics as the study of the solutions of the Schrodinger Wave Equation , which we'll call *states*. Since the $\frac{\partial}{\partial t}$ and the ∇^2 operators are linear, the states of a system form a Vector Space $\mathscr{F}$ over the complex numbers, with the trivial solution as the identity. Physically however, the trivial solution is disregarded as it tells us that the particle doesn't exist, and we can ignore it. But we leave it in the set of solutions for a more generalised approach. To simplify calculations, we can look at only the variable separable solutions. Hence,

$$\Psi(\vec{r},t) \to \psi(x)\cdot\phi(t) \tag{2}$$

$$\Rightarrow \imath\hbar\psi(x)\frac{\partial\phi(t)}{\partial t} = \left[-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V(x)\right]\psi(x)\phi(t) \tag{3}$$

$$\Rightarrow \frac{\imath\hbar}{\phi(t)}\frac{\partial\phi(t)}{\partial t} = -\frac{\hbar^2}{2m\psi(x)}\frac{\partial^2\psi(x)}{\partial x^2} + V(x) \tag{4}$$

In (4), we see that the left side is dependant only on the time, and the right only on space, hence both must be a constant. Focusing only on space dependency,

$$-\frac{\hbar^2}{2m}\frac{\partial^2\psi(x)}{\partial x^2} + V(x)\psi(x) = E\psi(x) \tag{5}$$

which can be more compactly written as

$$\hat{\mathcal{H}}\psi = E\psi \tag{6}$$

The above equation is an eigenvalue equation which can be solved analytically for some contrived potentials, and computationally for more complex potentials.[5]

Since the integral form of the equation runs over all space, we need to look at solutions that go to zero at infinity. Of course, "a good mathematician can supply one with pathological counterexamples, but they do not arise in Physics; for us, the wave function *always* goes to zero at infinity."[4] The advantage of studying this equation becomes clear when we show that the solutions of this equation form a basis of $\mathscr{F}$. This analysis is, however, not necessary for any further insight. Some bases may not themselves exist in $\mathscr{F}$, but are useful as a tool to visualise the states.

With the mathematical basis set for further analysis, we explore certain bases such as

1. Plane Waves
2. Delta Function $\xi(r)$

3.1 Bras and Kets

We can now redefine the Dirac Notation with the kets ($|\psi\rangle$) being the state vectors and the bras ($\langle\phi|$) denoting the dual of the kets over an inner product. The inner product is defined as $(|\psi_1\rangle, |\psi_2\rangle) = \langle\psi_1|\psi_2\rangle = \int d^3r\ \psi_1^*(r)\ \psi_2(r)$ which is a complex number.

3.2 Operators

We also define an operator as a transformation over kets and give the Dirac notation as $|\psi_1\rangle\langle\psi_2|$. Clearly, from the Dirac notation, an operator left multiplied, takes one ket to another ket. It can also be shown that a right multiplication of an operator to a bra gives rise to another bra. Hence we can define a new linear functional $\langle\phi|(A|\psi\rangle) = (\langle\phi|A|)\psi\rangle = \langle\phi|A|\psi\rangle$ Hence we can also define $A^\dagger$ as the hermition conjugate of A as $(A|\psi\rangle)^\dagger = |A\psi\rangle^\dagger = \langle A\psi| = \langle\psi|A^\dagger$

We also study certain operators such as

1. Projectors

3.3 Matrix Notation

Generally, every element in a vector space can be denoted by a matrix in a given base. We can define kets as column matrices, bras as row matrices, and operators as square matrices. This process of denoting all the elements of quantum mechanics as a matrix plays an important role in computational quantum mechanics[5] and our simulation of a quantum circuit.

We further analyse operators and define commutator of [A, B] = AB - BA. We say that if [A,B] = 0, the operators *commute*. We define the eigenvectors and eigenvalues of an operator according to their usual definition, and prove that eigenvectors of A and B form a complete set $\iff$ $[A, B] = 0$. These are known as Complete Sets of Commuting Observables(C.S.C.O).

Now we can analyse commonly used representations such as:

1. $|\ \vec{r}\ \rangle$
2. $|\ \vec{p}\ \rangle$

3.4 Tensor Products

We define Tensor product between two different spaces such that it is linear in complex multiplication and distributive over addition. We then define all quantities for this new space as we had done before. This section plays a vital role later in writing an n-Qubit system simply as an n tensor product of the Qubit space. Along with the matrix representation, we can very easily extend operators for larger systems, and these two concepts(matrix representation and Tensor Products) form the basis of our approach to simulate a Quantum circuit.

4 Quantum Computing[1]

After a brief introduction to the concept of computing, we begin the study of Quantum Computing.

4.1 The Qubit

We first define a Qubit as a state in a two-dimensional Vector Space. We abstract the physical manifestation of such a state and deal only with the mathematical definition of a state.

As usual, there can be infinite bases, but the most commonly used bases are the 0,1 basis and the *Computational Basis.* A ket in the 0,1 basis is represented as

$\psi = \alpha|0\rangle + \beta|1\rangle = e^{\iota\gamma}(cos\frac{\theta}{2}|0\rangle + e^{\iota\phi}sin\frac{\theta}{2}|1\rangle$

The numbers θ and ϕ define a point on the unit three-dimensional sphere. This sphere is often called the Bloch sphere; it provides a useful means of visualising the state of a single Qubit and often serves as an excellent test-bed for ideas about quantum computation and quantum information.[1] It should also be noted that the $e^{\iota\gamma}$ makes no difference when measured, as the probability is dependant on the mod-square of coefficients.

A measurement of a Qubit will give either $|0\rangle$ or $|1\rangle$ with the probability $|\alpha|^2$ or $|\beta|^2$ respectively. This arises from the further analysis of the value of $\langle\phi|A|\phi\rangle$

The base vectors of the Computational Basis are defined as the following $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$, $|-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$. Clearly they are orthogonal and can form a basis.

4.1.1 Multiple Qubits

With the definition of a tensor product defined in §3.4, we can easily extend our definition of a Qubit to the definition of an n-Qubit state.

4.2 Quantum Computation

4.2.1 Single Bit Gates

Operators defined over the Qubit space are called “Gates”, and since they operate on Quantum Objects, there are called “Quantum Gates”. They are the Quantum equivalent of the Classical Boolean gates. Of course, infinite such gates exist, and they can all be written as

$\text{U} = e^{\iota\alpha}\begin{bmatrix} e^{\frac{-\iota\beta}{2}} & 0 \\ 0 & e^{\frac{\iota\beta}{2}} \end{bmatrix}\begin{bmatrix} cos\frac{\gamma}{2} & -sin\frac{\gamma}{2} \\ sin\frac{\gamma}{2} & cos\frac{\gamma}{2} \end{bmatrix}\begin{bmatrix} e^{\frac{-\iota\delta}{2}} & 0 \\ 0 & e^{\frac{\iota\delta}{2}} \end{bmatrix}$ where the matrices are rotations in the three planes on the Bloch sphere.

Despite the infinite possible gates, we focus only on some gates, which are of paramount importance.

1. NOT Gate ($0 \rightarrow 1$; $1 \rightarrow 0$)
2. Hadamard Gate ($0 \rightarrow +$; $1 \rightarrow -$)

3. Z Gate ($0 \rightarrow 0$; $1 \rightarrow -1$)

4.2.2 Multi-Qubit Gates

An n-bit Gates are those which operate on an n-bit system (or n bits of a k($> n$) bit system). A common multi-bit gate is the CNOT gate which does the operation $|A, B\rangle \rightarrow |A, A \oplus B\rangle$. It can be shown that the CNOT gate and the single bit gates together form the basis for all other gates. Another commonly used gate is the Toffoli gate which does $|A, B, C\rangle \rightarrow |A, B, C \oplus AB\rangle$. The Toffoli acts like a Classical NAND, and can hence be used to simulate any classical circuit quantum mechanically.

4.3 EPR states and Quantum "Teleportation"

EPR states are states which are of the form $\beta_{x,y} = \frac{|0,y\rangle+(-1)^x|1,\bar{y}\rangle}{\sqrt{2}}$

These states are exciting, as measuring one Qubit will give the state of the other immediately with certainty. This property gives rise to the paradox of whether separating the bits leads to an infinitely fast transfer of information. However, this is not the case, and Special Relativity is NOT violated as the measurement gives Classical information which can travel only at (or slower than) the speed of light.

We can exploit this to "Teleport" quantum information from A to B. We invite over Alice and Bob to demonstrate this. Suppose Alice and Bob had an EPR couple and are now physically separated from each other. Suppose Alice needs to send a Qubit in $|\psi\rangle$ state to Bob. She can perform a series of manipulations to the data bit and her EPR, such that Bob's EPR becomes a slight variation of the data bit. Once Alice measures her bits, she can share this information with Bob, and he can perform another set of operation on his EPR to get back the data bit. This procedure seems to offer the same paradox as above, but once Alice measures her bits, the information must travel to Bob over a Classical channel only, and it is bound by the speed limit of the universe.

4.4 Quantum Parallelism

The biggest advantage of a quantum computer is the fact that it can operate on many bits at once. This property is explored further with the Deutsch's and The Deutsch–Jozsa Algorithms.

5 Simulation of a Quantum Computer on a Classical Computer

It is first necessary to define what *simulation* means in this context. Without a doubt, we cannot wholly simulate a quantum computer as we do with other Classical Physical systems. Quantum Parallelism can never be accomplished on a Classical Computer (if we could why bother at all with Quantum Computers). However, we

CAN simulate a Qubit by “measuring” it with the probability of its coefficients. We make use of pseudo-random generators for this.

First, as all good programs should be, we modularise our program into two modules.

1. QuantumAlgebra will define the mathematical structures that we use in Quantum Computing. Kets, bras, bases are all represented in memory as matrices. Any operation is internally a matrix multiplication.

2. QuantumComputing will define a Qubit as a two-dimensional ket. We also define gates here, which are also represented as matrices. An n-Qubit system is represented as an n length array of Qubits. Multi-bits gates will operate on specified bits.

3. A Module can be added which will add GUI representations of Gates and the user will be able to make a Quantum Circuit without knowing the language.

Julia[2] was used due to its efficiency in Scientific Computing and Matrix Manipulation and ease of writing code.

5.1 QuantumAlgebra.jl

As we saw, Quantum Mechanics can be very elegantly studied as a Vector Space with defined Bases, Inner Products, Tensor Products and Transformations (Operators). To *simulate* a Quantum Computer, we must first define our Linear Algebra in the Hilbert space. Though not necessary, we make a Julia Module called Quantum Algebra, which handles a generalised Quantum Mechanic algebra of Bras, Kets, Operators and Bases. Then, QuantumComputing.jl a specific case of QuantumAlgebra.jl

5.1.1 Basis struct

We first define a Basis struct which is composed of a transformMatrix:: Array{Number,2} and a String name. transformMatrix is the Matrix form of the transformation which takes a Ket (or Bra) in the Identity Basis to a Ket (or Bra) in this basis.

The constructor takes both arguments and checks that transformMatrix is Unitary. If not, it throws a NotUnitaryError. We define Identity(N=2) function which returns the N $\times$ N Identity Basis.

5.1.2 Bra struct

We define a Bra struct which is composed of coefficients:: Array{Number, 2} and basis::Basis. There are two constructors, Bra(coefficients::Array{Number, 2}, basis::Basis) and Bra(coefficients::Array{Number, 2}) which defaults to the Identity Basis. coefficients is a Row vector which is the Matrix representation of the Bra in the given (or assumed) basis. Note - Bra is created in normalised state.

5.1.3 Ket struct

We define a Ket struct which is composed of coefficients::Array{Number, 1} and basis::Basis. There are two constructors, Ket(coefficients::Array{Number, 2}, basis::Basis) and Ket(coefficients::Array{Number, 2}) which defaults to the Identity Basis. coefficients is a Column vector which is the Matrix representation of the Ket in the given (or assumed) basis. Note - Ket is created in normalised state.

5.1.4 Associated Functions

Following functions are also defined-

- `transform!(`ψ`::Bra/Ket, newbasis::Basis) :: Bra/Ket = transform` ψ `from its basis to newbasis and also returns it`
- `measure!(`ψ`::Ket) :: Ket = "Measures" a Ket using StatsBase.sample and returns the collapsed, normalised` ψ`. Note - This function normalises the Ket. This is obvious as total probability needs to be 1.`
- `dual(`ψ`::Ket/Bra) = Returns the dual (Bra/Ket respectively)`
- `scalarProduct(`ψ`1::Bra/Ket,` ψ`2::Ket) :: Complex = returns the scalar product if both are in the same Basis, else a basisMismatch Error is thrown. Note - Though scalar product is invariant under Basis change, and one Ket/Bra could, in theory, be converted to the other's basis to calculate the scalar product, this has been avoided to prevent mistakes that may arise from incorrect scalar products.`
- `*(`ψ`::Bra/Ket,` ψ`::Ket) = Alias of scalarProduct`
- `*(b::Basis,` ψ`::Ket) = Alias of transform!. Note - This changes` ψ`.`
- `*(`ψ`::Bra, b::Basis) = Alias of transform!. Note - This changes` ψ`.`
- `==(k1::Bra/Ket/Basis, k2::Bra/Ket/Basis) = Checks if k1 = k2`
- `norm(`ψ`::UnionKet,Bra) :: Real = returns the norm (length) of the ket.`
- `normalise!(`ψ`::UnionKet,Bra) = returns normalised` ψ`.`

5.2 QuantumComputing.jl

The final wrapper module which is the most used module by the User.

5.2.1 Qubit struct

A Qubit is simply a wrapper around Ket which allows only 2 dimensional Kets and 2 × 2 Bases. Constructor is of the form Qubit(ψ::Array{Number,1}, basis::Basis) and falls back to Identity Basis if basis not provided. It throws IncorrectSizeError if length of ψ is not 2.

5.2.2 Operator struct

An Operator consists of an 2n × 2n Unitary matrix::Array{Number, 2} and number_of_bits :: Integer which is calculated from the matrix. Constructor is of form Operator(matrix :: Array{Number,2}, basis :: Basis) where matrix is the Matrix representation of the Operator in the given Basis (falls back to Identity basis if not provided). Note - Operator.matrix is ALWAYS stored in its Identity representation. It is converted to required basis according to the Qubit it operates on.

5.2.3 Associated Functions

- `transform(operator::Operator, newbasis::Basis) :: Operator = Overloaded from QuantumAlgebra. Transform an operator from its Basis to another and return the new operator. Note - The new operator is not technically` *`in the new basis`*`. It is essentially a new Operator that has the required matrix representation and hence NOT EXPORTED.`
- `operate!(operator::Operator, qubit::Qubit) = Operates an operator on a qubit and returns the qubit after the operation. Throws IncorrectInputNumberError if the operator is not a single bit operator`
- `operate!(operate!(operator::Operator, qubits::ArrayQubit, 1) :: ArrayQubit, 1 = Operates an operator on a set of Qubits and returns the Qubits after operation. Throws IncorrectInputNumberError if the number of Qubits is not equal to Operator size. WARNING - This function is still being built.`
- `Base.*(operator::Operator, qubit::Qubit) = Alias for operate!(). Note - Changes the qubit.`

5.2.4 Predefined operators

- `Hadamard, H`
- `PauliX, NOT`
- `PauliY`
- `PauliZ, Z`

5.2.5 Predefined Bases

- `Computational Basis (comp_basis)`

References

[1] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. 10th Anniversary. Cambridge University Press, 2010.

[2] J. Bezanson et al. "Julia: A Fresh Approach to Numerical Computing". In: *SIAM Review* 59.1 (2017), pp. 65–98. DOI: `10.1137/141000671`. eprint: `https://doi.org/10.1137/141000671`. URL: `https://doi.org/10.1137/141000671`.

[3] Claude Cohen-Tannoudji, Bernard Diu, and Frank Laloë. *Quantum Mechanics*. 2nd ed. Vol. 1. Wiley-Interscience, 1977.

[4] D.J. Griffiths. *Footnote 12, Introduction to Quantum Mechanics*. Pearson international edition. Pearson Prentice Hall, 2005. ISBN: 9780131118928. URL: `https://books.google.co.in/books?id=z4fwAAAAMAAJ`.

[5] H. Korsch and M Glück. "Computing quantum eigenvalues made easy". In: *European Journal of Physics* 23 (July 2002), p. 413. DOI: `10.1088/0143-0807/23/4/305`.

[6] Richard P. Feynman, Robert B. Leighton, and Matthew Sands. *The Feynman Lectures On Physics*. New Millenium. Vol. 3. Basic Books, 2011.