

Implementing the Klein Gordon Equation on a Quantum Computer

Desiree Vogt-Lee, 44354471

10 June, 2019

All components of this project including code, Jupyter Notebooks, LaTeX files and images can be found in my [Github repository](#), please check this out for reference.

1 The Klein Gordon Equation

The Klein Gordon equation set out to unify quantum mechanics and special relativity and ultimately led to the creation of the Dirac equation from which quantum field theory was born.

The Klein Gordon (KG) equation is unable to describe a particle as the Schrodinger equation does due to there being no conserved probability.

$$\frac{1}{c^2} \frac{\partial^2}{\partial t^2} \psi - \nabla^2 \psi + \frac{m^2 c^2}{\hbar^2} \psi = 0 \quad (1)$$

The Schrodinger equation is a first order equation in the time derivative where its' wavefunction is able to predict its future evolution. The Klein Gordon equation on the other hand is second order in the time derivative and the boundary condition for the field must be given to work out the field evolution. The KG equation also allows energy to be a negative value which can't be eliminated by limiting to only positive initial conditions due to run off effects that result in particles with $E < 0$. The Dirac equation was Dirac's attempt to create an equation that had a first order time derivative and was relativistic [?].

The Klein paradox arises from using the Dirac equation to solve the problem of electron scattering from a barrier potential. Under non-relativistic circumstances, the electron would tunnel into the barrier. However if the barrier potential is greater than mc^2 , the particle is able to penetrate the barrier as if it were transparent [1].

This report will be based off a paper titled *Quantum Simulation of Klein Gordon Equation and Observation of Klein Paradox in IBM Quantum Computer* by Manik, K et al. (2018) [2] which aimed to simulate the time dependent Klein Gordon equation in a barrier potential on a quantum computer to see the tunneling of both the particle and anti-particle with the Klein Paradox.

The present report aims to replicate results, justify decisions made in the original publication and offer criticism and suggestions.

2 Quantum Computing

Quantum computing is the melding together of quantum mechanics and classical computation. The computer chips used in quantum computers use the same fundamental components as that of their classical counterparts, however are pieced together in a way which allow quantum-bits, or qubits, to be used instead of bits.

A fundamental flaw in the current design of quantum computers is the large amount of error that is created due to noise and decoherence, which increases with the number of qubits used. This severely limits the computation that can be performed with a negligible amount of error, as at some point, the current error correcting codes will have little effect.

Many believe that quantum computation is the future. Due to the superposition and entanglement properties these qubits can possess, all sorts of optimisation and new algorithms can be created, which would have significant impact on many fields of technology. For example, modern cryptography is centered around the assumption that creating incredibly long numbers take current computers an impossibly long time to factorise. Using Shor's algorithm, a quantum computer would be able to factorise a number in polynomial time, exponentially faster than the best classical algorithm, and thus being able to break modern cryptography. [3]

At present, this is all merely a premonition; as although there exists many quantum algorithms, the error is too great and the number of qubits too little to be able to use them. I believe that applying the Klein Gordon equation on a quantum computer is useful in showing the present capabilities of this new technology as well as just being a really interesting application.

2.1 Gates

To modify the state that a qubit is in, a gate or procession of gates can be applied to it. The quantum gates are similar to the logic gates present in classical computing, but work with the quantum nature of the qubit. To easily conceptualise the effect of a quantum gate, a geometric representation called the Bloch sphere is used, where the poles of the sphere represent the standard basis vectors $|0\rangle$ and $|1\rangle$.

In classical computation, the NOT gate simply flips a given bit state: a 1 will be transformed to a 0 and vice versa. In quantum computing there is an equivalent X gate which when applied to a qubit in the $|0\rangle$ state, performs a rotation about the x axis on a Bloch sphere by π radians resulting in the $|1\rangle$ state when measured (for more detailed explanation on the fundamental gates and terminology, please see my [blog post](#)).

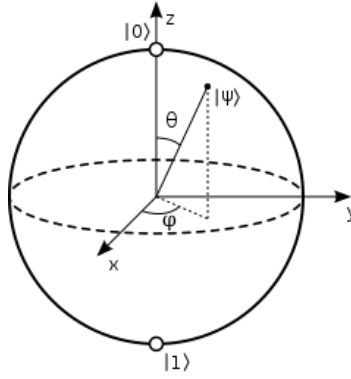


Figure 1: Bloch Sphere [4]

Here I will briefly explain the gates that will be used to create the circuit that implements the Klein Gordon equation (minus those explained in my blog post):

- **T** T: performs a $\frac{\pi}{4}$ rotation about the z axis, which is equivalent to a quarter of one Z gate operation. This gate acts on the phase of the qubit and is represented by the matrix:

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{pmatrix}$$

- **T[†]** T[†]: simply the opposite of the T gate and applies a $-\frac{\pi}{4}$ rotation about the z axis [5].
- **U1** U₁: performs a unitary rotation on the state vector where the continuous phase can be specified. It is represented by the following matrix and as can be seen, is of a similar to that of the T gate, suggesting that a U₁($\frac{\pi}{4}$) gate is equivalent to a T gate.

$$U_1(\lambda) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\lambda} \end{pmatrix}$$



- **U3** U₃: used to perform arbitrary unitary operation, where the θ , ψ and λ position of the vector on the Bloch sphere can be modified to any value. This is the most general form of a single qubit unitary operation and is represented in matrix form as [6]:

$$U = \begin{pmatrix} \cos(\theta/2) & -e^{i\lambda} \sin(\theta/2) \\ e^{i\phi} \sin(\theta/2) & e^{i\lambda+i\phi} \cos(\theta/2) \end{pmatrix}$$

- **R**

$$R = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{pmatrix}$$

- **SWAP** SWAP: operates on a pair of qubits and swaps their state.

-  Measurement: used at the end of the circuit to measure the outcome of the states in the z basis.
-  Barrier: directives which inform the compiler not to optimise across barriers. They serve the purpose of keeping gates, especially measurement gates, in their place and are as well used as a visualisation tool to separate parts of the circuit.

3 The Circuits

3.1 Klein Gordon Equation as Unitary Operators

To implement the KG equation on a quantum computer, it must be written in terms of unitary operators. To do this we start with the equation for a Hamiltonian in terms of mass of a particle (m), momentum operator (\hat{p}), potential (V), speed of light (c), identity matrix (I) and Pauli matrices ($\sigma_{1,2,3}$):

$$\hat{H} = \frac{\sigma_3 + i\sigma_2}{2m} \hat{p}^2 + \sigma_3 mc^2 + I\hat{V} \quad (2)$$

By using the vector:

$$\varphi = \begin{bmatrix} \phi \\ \chi \end{bmatrix}$$

We can solve two simultaneous equations, resulting in:

$$\begin{aligned} \hat{H}_1 \phi &= \frac{\hat{p}^2}{2m}(\phi + \chi) + (mc^2 + \hat{V})\phi \\ \hat{H}_2 \chi &= -\frac{\hat{p}^2}{2m}(\phi + \chi) + (-mc^2 + \hat{V})\chi \end{aligned}$$

These equations then need to be represented as a time evolution equation which can be done by taking the second order Trotter-Suzuki decomposition of the Hamiltonian:

$$|\psi(x, t + \delta)\rangle = e^{-i\hat{H}_i t} |\psi(x, t)\rangle$$

In the code there is an option to choose to run the circuit on a real IBM quantum computer or the simulator.

3.2 Fast Fourier Transform

The Fourier transform is able to decompose a function into its constituent functions, which has many practical applications including signal processing, analysis of differential equations and spectroscopy [7]. One particular variation of Fourier transform of interest is the discrete Fourier transform (DFT) as it is computed using the fast Fourier transform (FFT) algorithm. The DFT computes a finite sequence

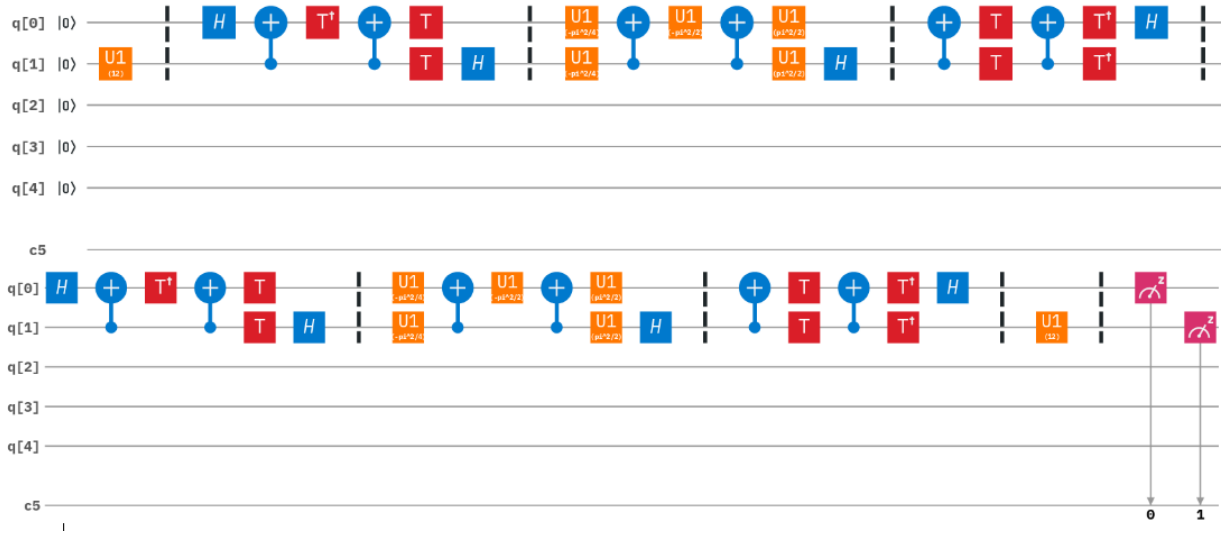


Figure 2: Circuit 1

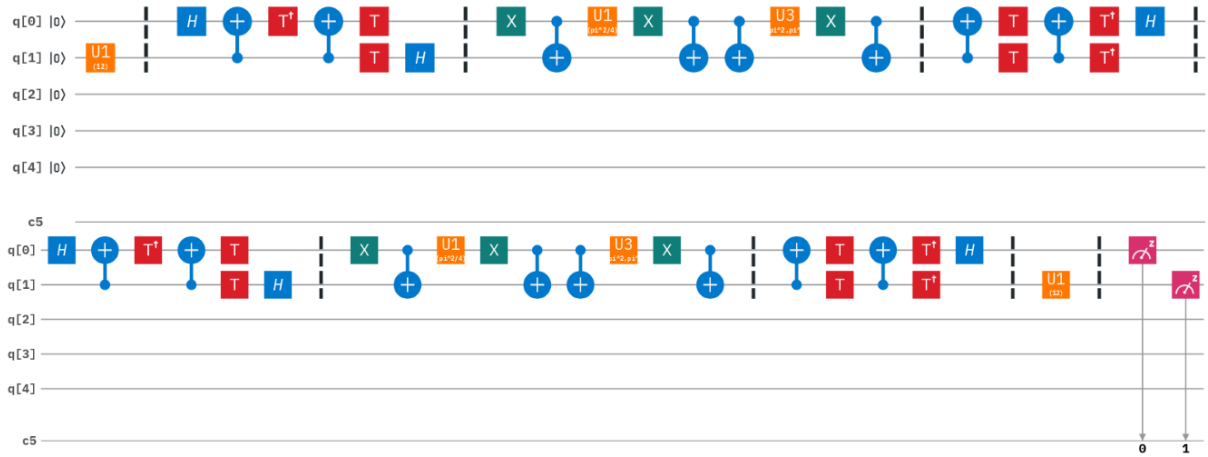


Figure 3: Circuit 2

of points of a function to its various frequency components, which although useful, is computationally expensive. The Cooley-Tukey [8] FFT algorithm accelerates this process by splitting up the DFT problem into smaller parts and recognising a symmetry that exists between each even term and each odd term:

$$\begin{aligned}
 y_k &= \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi kn/N} \\
 &= \sum_{m=0}^{N/2-1} x_{2m} \cdot e^{-i2\pi k(2m)/N} + \sum_{m=0}^{N/2-1} x_{2m+1} \cdot e^{-i2\pi k(2m+1)/N} \\
 &= \sum_{m=0}^{N/2-1} x_{2m} \cdot e^{-i2\pi km/(N/2)} + e^{-i2\pi k/N} \sum_{m=0}^{N/2-1} x_{2m+1} \cdot e^{-i2\pi km/(N/2)}
 \end{aligned} \tag{3}$$

Therefore instead of having to recompute the entirety of each term, only half of the subproblem needs to be computed. This method of dividing and computing half can be applied recursively until the vectors are small enough that the method no longer yields better results, at which point the computational complexity is $O(n \log n)$ (compared to the $O(n^2)$ of regular DFT) [9].

3.3 Quantum Fourier Transform

The quantum Fourier transform is very similar to the FFT mentioned above. The quantum version acts on amplitudes of a quantum state and maps them to new amplitudes. The mathematical representation of the QFT is:

$$\alpha_j |j\rangle \rightarrow \alpha_j \left(\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i \frac{kj}{N}} |k\rangle \right) \quad (4)$$

As in with standard DFT, a delta function will be transformed to a sine function using a QFT. For a two qubit system, with one qubit in the $|1\rangle$ state and the other in the $|0\rangle$ state, $|10\rangle$ will transform into the following superposition:

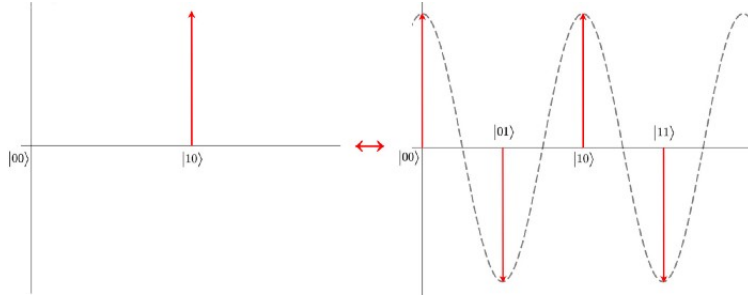


Figure 4: Delta function transformed to a sine function of two qubits: $|10\rangle \rightarrow |00\rangle - |01\rangle + |10\rangle - |11\rangle$ [10]

Just as the FFT is performed iteratively on subproblems, so is the QFT: Equation xxxxxxxxxxxxxxxx can be broken down into the following expression which allows for easier understanding of how the equation can be implemented as gates on a quantum circuit.

$$|j_1 \cdots j_n\rangle \rightarrow \frac{(|0\rangle + e^{2\pi i 0 \cdot j_n} |1\rangle) (|0\rangle + e^{2\pi i 0 \cdot j_{n-1} j_n} |1\rangle) \cdots (|0\rangle + e^{2\pi i 0 \cdot j_1 j_2 \cdots j_n} |1\rangle)}{2^{n/2}} \quad (5)$$

The first Hadamard in the following circuit representation of Equation xxxxxxxxxxxx puts the system into the superposition state required (the $1/\sqrt{2}$) and the following multiplications are implemented using the R controlled phase gates. The pattern apparent in the circuit indicates the recursive nature of the algorithm. The swap gate can be applied either at the beginning or end of the system as it is just required that the states of the two qubits are swapped prior to measurement and its position has no effect on the outcome.

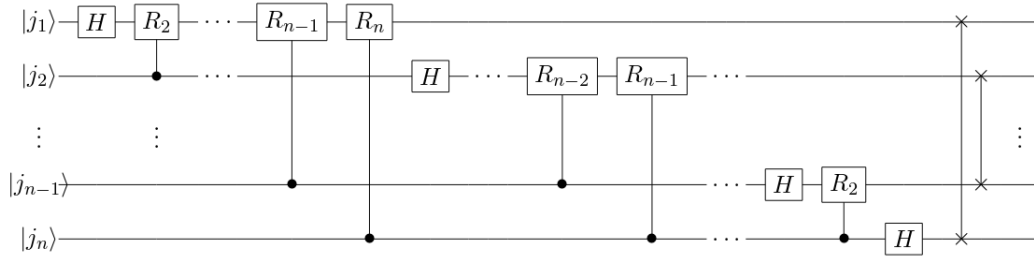


Figure 5: QFT circuit implemented on an n qubit system [11].

As our implementation of the Klein Gordon equation only uses two qubits, the quantum Fourier transform on two qubits is represented by the following circuit.

Like the classical Fourier transform, the quantum Fourier transform is reversible as well as unitary.

4 Exploration

The QFT in the circuit by Manik, K (2018) [2] is different to the aforementioned

We expect to see

5 Results

5.1 Kapil, M. et al. (2018) Results

5.2 My Results

References

- [1] O. Klein, “Die Reflexion von Elektronen an einem Potentialsprung nach der relativistischen Dynamik von Dirac,” *Zeitschrift fur Physik*, vol. 53, pp. 157–165, Mar. 1929.
- [2] M. Kapil, B. K. Behera, and P. K. Panigrahi, “Quantum Simulation of Klein Gordon Equation and Observation of Klein Paradox in IBM Quantum Computer,” *arXiv e-prints*, p. arXiv:1807.00521, Jul 2018.
- [3] H. Reich, “How quantum computers break encryption — shor’s algorithm explained.” <https://www.youtube.com/watch?v=lvTqbM5Dq4Q>, 2019.
- [4] S. Meister, “Bloch sphere.” https://en.wikipedia.org/wiki/File:Bloch_sphere.svg, 2009.

- [5] IBM, “Introducing qubit phase.” https://quantumexperience.ng.bluemix.net/proxy/tutorial/full-user-guide/002-The_Weird_and_Wonderful_World_of_the_Qubit/003-Introducing_qubit_phase.html, nd.
- [6] IBM, “Advanced single-qubit gates.” https://quantumexperience.ng.bluemix.net/proxy/tutorial/full-user-guide/002-The_Weird_and_Wonderful_World_of_the_Qubit/004-advanced_qubit_gates.html, nd.
- [7] P. Bevelacqua, “Fourier transforms, fourier transform, tutorial, fourier series, fourier.” <http://www.thefouriertransform.com/applications/fourier.php>, 2010.
- [8] J. Cooley and J. Tukey, “An algorithm for the machine calculation of complex fourier series,” *Mathematics of Computation*, vol. 19, no. 90, pp. 297–301, 1965.
- [9] J. VanderPlas, “Understanding the fft algorithm.” <https://jakevdp.github.io/blog/2013/08/28/understanding-the-fft/>, 2013.
- [10] J. Hui, “Qc — quantum fourier transform.” https://medium.com/@jonathan_hui/qc-quantum-fourier-transform-45436f90a43, 2018.
- [11] J. V. Gael, “The role of interference and entanglement in quantum computing.” <http://pages.cs.wisc.edu/~dieter/Papers/vangael-thesis.pdf>, ”May” 2005.