Implementation of HHL Algorithm for Solving a Linear Systems of Equations

Hassaan Ahmed (21060001)

28th April 2023

1 Abstract

Many natural systems of interest have non-linear dynamics, the dynamics are complex. Linearization reduces the complexity. To solve such linear systems, many classical methods exist. Harrow, Hassidim and Lloyd presented a quantum algorithm to solve such linear systems. In this project, HHL algorithm is used to solve a linear system of equations, the algorithm is simulated in python using qiskit library and then the program is run on a quantum computer. The results of quantum simulation and quantum computer are then compared.

2 Introduction

To study the evolution of systems with non-linear dynamics, the systems are linearized at their equilibrium points. Most data-processing techniques use linearized versions of such systems, but as the data becomes large, solving these systems requires more computational resources. The fastest known classical method solves linear systems in polynomial time. Quantum Computing promises an exponential speed over the classical methods, but it suffers from limitations in actual hardware implementations[11].

An algorithm to solve a linear system of equations was presented by Harrow, Hassidim and Lloyd [8]. The general form of a linear system of equations is shown in (1). There are M equations with M unknown variables. A is a $M \times M$ matrix and is assumed to be Hermitian i.e. it is the conjugate transpose of itself (2). A and $\vec{b}$ are known, while $\vec{x}$ is the unknown vector whose solution we desire. Dimensions of $\vec{x}$ and b are $M \times 1$. If A is not Hermitian then it can be converted into a Hermitian matrix A' as shown in (3), then the resulting system of equations is shown in (4,5,6)[8].

$$A\vec{x} = \vec{b} \tag{1}$$

$$A = (A^*)^T \tag{2}$$

$$A' = \begin{bmatrix} 0 & A \\ A^* & 0 \end{bmatrix} \tag{3}$$

$$\vec{x'} = \begin{bmatrix} 0 \\ \vec{x} \end{bmatrix} \tag{4}$$

$$\vec{b'} = \begin{bmatrix} \vec{b} \\ 0 \end{bmatrix} \tag{5}$$

$$A'\vec{x'} = \vec{b'} \tag{6}$$

To solve a linear system with M unknowns we need m qubits where

$$m = \log_2(M) \tag{7}$$

$$M = 2^m \tag{8}$$

To represent the vectors $\vec{x}$ and $\vec{b}$ as quantum states, we need to re-scale them into unit vectors. This is done by dividing the vectors with their norms as shown in (9,10) [2].

$$|b\rangle = \frac{\vec{b}}{||b||} \tag{9}$$

$$|x\rangle = \frac{A^{-1}\vec{b}}{||A^{-1}\vec{b}||} \tag{10}$$

2.1 Classical Methods for Solving Linear System of Equations

Gaussian Elimination and Conjugate Gradient Method are, traditionally, used to solve a linear system of equations.

In Gaussian Elimination, row reduction techniques applied on A are applied on $\vec{b}$ as well. A is transformed into identity and the resultant $\vec{b}$ vector, after the sequence of operations, is the solution vector $\vec{x}$ [6]. The Complexity of this method is $O(M^3)$. There are forms, other than identity, that matrix A can be reduced to but is not elaborated further in the report.

In Conjugate Gradient Method, an initial guess is used as a starting point, and then the direction of the steepest descent is determined. It is much faster than the Gaussian Elimination with complexity of $O(M)$ [11]. It is an iterative algorithm[4] that is applicable on sparse systems which are too large to be handled using direct methods.

2.2 Quantum Mechanical Concepts

2.2.1 Superposition

It is just a linear combination of 2 or more basis states as shown in (11). Where the coefficients c_o and c_1 are complex numbers. Typically, superposition can be created using a Hadamard Gate. It does not have a classical counterpart (unlike the Not gate). It creates equal superposition (equal probability) of the basis states.

$$|\psi\rangle = c_o |0\rangle + c_1 |1\rangle \tag{11}$$

2.2.2 Entanglement

When 2 or more states cannot be represented as a tensor product of the individual qubits, the states are said to be entangled. A completely entangled state is shown in (12)

$$|\psi\rangle = c_0 |00\rangle + c_1 |11\rangle \tag{12}$$

2.2.3 Eigenvalue and vectors

We can decompose every non-zero square matrix into a product of its eigenvectors and a diagonal matrix containing all the eigenvalues, this procedure is also called Eigenvalue Decomposition shown in (13).

$$A = \vec{V}^{-1} \Lambda \vec{V} \tag{13}$$

The eigenvalues are scalars and each eigenvalue has an eigenvector associated with it. If you pass an eigenvector of matrix A as an input to the matrix A then the output is a scaled version of the same eigenvector.

2.2.4 Controlled Operation

The controlled gate has a target qubit and a control qubit, the gate operates on the target qubit only when the control qubit is in the state $|1\rangle$, if the control qubit is in the state $|0\rangle$, then the target qubit passes through the gate as is.

2.3 Types of Encoding

2.3.1 Hamiltonian Encoding

The Hamiltonian represents the total energy of a system. It generates the time evolution of the quantum states. For a hermitian matrix A, which is encoded as the Hamiltonian of a unitary operator U, the operator U is defined as in (14). However, A does not have to be unitary in this definition.

$$U = e^{iAt} \tag{14}$$

This is just one type of Hamiltonian encoding, other forms also exist.

2.3.2 Amplitude Encoding

In (11) the amplitudes or coefficients of $|\psi\rangle$ basis vector $|0\rangle$ and $|1\rangle$ are c_0 and c_1, respectively. In amplitude encoding, this is represented as

$$\begin{bmatrix} c_0 \\ c_1 \end{bmatrix} \tag{15}$$

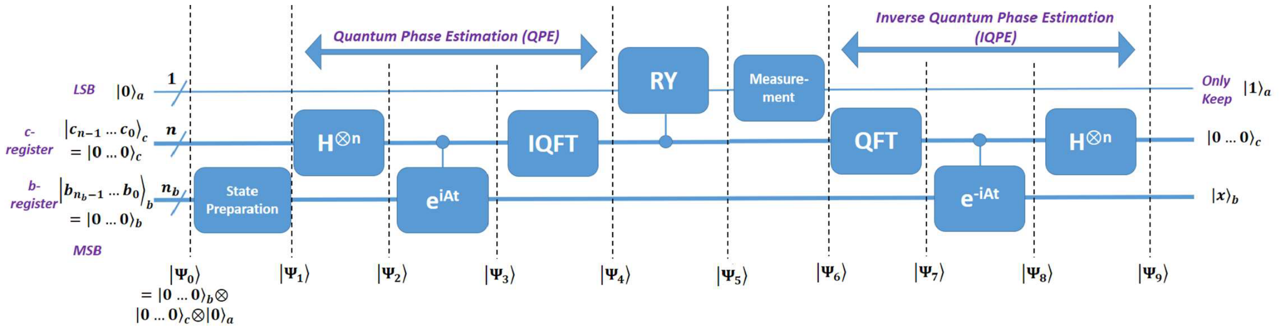


Figure 1: Schematic of HHL Algorithm [11]

where, the square of the coefficients are the probability amplitudes of the respective states. Consequently, sum of square all the coefficients should be unity.

$$\sum_i c_i^2 = 1, \forall i \tag{16}$$

2.3.3 Basis Encoding

In Basis encoding, decimal numbers are converted to their binary representation and then the binary representation is assigned respective quantum basis states $|0\rangle$ and $|1\rangle$. Example, for the decimal representation,

$$\begin{bmatrix} 0 \\ 3 \end{bmatrix} \tag{17}$$

The binary representation is

$$\begin{bmatrix} 00 \\ 11 \end{bmatrix} \tag{18}$$

Then its basis state representation is $|0011\rangle$.

3 Mathematical Formulation

3.1 Preliminaries

To begin, there are 3 main divisions of the total qubits required to implement the HHL algorithm. The b-register consists of m qubits, in this the information regarding $\vec{b}$ is encoded. The c-register consists of n qubits, it has information regarding the clock or timing of the controlled rotation part of the algorithm [11]. In addition to b-register and c-register, a single ancillary qubit is also a part of the algorithm. Total qubits required to implement HHL are $m + n + 1$.

HHL, itself, consists of basically three operations:

1. Quantum Phase Estimation (QPE)
2. Ancillary Bit Rotation
3. Inverse Quantum Phase Estimation (IQPE)

QPE itself consists of 3 operations:

1. Superposition via Hadamard Gates
2. Unitary Rotation
3. Quantum Fourier Transform (QFT)

A schematic of HHL algorithm is illustrated in Figure 1. QPE is carried out on the b-register and c-register, Hadamard gates create superposition of the c-register which then acts as control inputs for the unitary rotations applied to the b-register. Inverse Quantum Fourier Transform (IQFT) is applied to the c-register. After IQFT, The ancillary qubit is then rotated and measured, resulting in discarding of the ancillary qubit. Then the process of QPE is applied in reverse and we obtain a solution of the $\vec{x}$.

To represent the qubits, little-endian convention is used, in this the rightmost (ending) qubit represents the least significant bit (LSB). Ancillary qubit is the LSB. This convention is used in qiskit as well.

The Hamiltonian matrix A can be written as an weighted outer product of its basis vectors. The weights would be the eigenvalues of A and the basis vectors would be the eigenvectors of A.

$$A = \sum_{i=0}^{M-1} \lambda_i |u_i\rangle \langle u_i| \tag{19}$$

Similarly, $\vec{b}$ can also be represented as a weighted sum of the eigenvectors of A.

$$|b\rangle = \sum_{j=0}^{M-1} b_j |u_j\rangle \tag{20}$$

From Eigenvalue decomposition in Section 2.2.3, taking the inverse of A would result in

$$A^{-1} = (\vec{V}\Lambda\vec{V}^{-1})^{-1} = \vec{V}\Lambda^{-1}\vec{V}^{-1} \tag{21}$$

We can now simple write A^{-1} as

$$A = \sum_{i=0}^{M-1} \lambda_i^{-1} |u_i\rangle \langle u_i| \tag{22}$$

Therefore $\vec{x}$ can be written as

$$|x\rangle = A^{-1} |b\rangle = \sum_{i=0}^{M-1} \lambda_i^{-1} |u_i\rangle \langle u_i| \sum_{j=0}^{M-1} b_j |u_j\rangle \tag{23}$$

We know that $\langle u_i|u_j\rangle = 1$ only when $i = j$. Hence,

$$|x\rangle = \sum_{i=0}^{M-1} \lambda_i^{-1} b_i |u_i\rangle \tag{24}$$

This is the result that will be stored in the b-register but it will be encoded in the basis of $|0\rangle$ and $|1\rangle$. Here we do assume that the weights are normalized, for appropriate representation as unit vectors. Since, the square of the weights give us their respective probability amplitudes (sum of total probability can't be greater than 1), the squared sum of weights should be equal to 1.

All qubits are initialized at state $|0\rangle$. The b-register has m qubits, c-register has n qubits and there is one ancillary qubit which is the LSB, the respective subscripts are also shown in the initial state.

$$|\psi_0\rangle = |0\rangle_b^{\otimes m} |0\rangle_c^{\otimes n} |0\rangle_a \tag{25}$$

Before Quantum Phase Estimation, the values of the $\vec{b}$ are stored in the b-register, but these are not just the coefficients of the $\vec{b}$, but rather the probability amplitudes of the coefficients of $\vec{b}$ [7][8].

$$|\psi_1\rangle = |b\rangle_b |0\rangle_c^{\otimes n} |0\rangle_a \tag{26}$$

3.2 Quantum Phase Estimation

QPE is an eigenvalue phase estimation routine. The unitary operator (14) is part of a controlled gate in the QPE routine. The phase of the eigenvalue of U is proportional to the eigenvalue of the matrix A, this is because the eigenvalues of U are roots of unity. Hence, after OPE the eigenvalues of A are expected to be stored in the c-register [11].

Hadamard Gates are applied on the qubits of the c-register (clock qubits) which would serve as the control qubits in the next operation. This results in a superposition of the clock qubits.

$$|\psi_2\rangle = |b\rangle_b (\frac{1}{2^{\frac{n}{2}}}(|0\rangle + |1\rangle))_c^{\otimes n} |0\rangle_a \tag{27}$$

The number of qubits in the c-register, n, determine the number of times the controlled gate is applied on the b-register. If there are n qubits, these qubits can be represented as $|c_{n-1}c_{n-2}..c_1c_0\rangle$. If the qubit c_0 is in the state $|1\rangle$ then the U is applied onto the b-register 2^0 times, if the qubit c_{n-1} is in the state $|1\rangle$ then the operator U is applied onto the b-register 2^{n-1} times. Assume that U has an eigenvalue $e^{2\pi i\theta}$ and its associated eigenvector $|b\rangle$, then

$$U|b\rangle = e^{2\pi i\theta} |b\rangle \tag{28}$$

This results in the phase θ being encoded as the basis state in the c-register. Because the operation is only carried out when the clock qubit is $|1\rangle$ and that the operation can be represented as a multiplication factor of $e^{2\pi i\theta 2^j}$ with $|1\rangle$ of $|c_j\rangle$ [11]. Then the states of the c-register becomes

$$(|0\rangle + e^{2\pi i\theta 2^{n-1}}|1\rangle) \otimes (|0\rangle + e^{2\pi i\theta 2^{n-2}}|1\rangle) \otimes ... \otimes (|0\rangle + e^{2\pi i\theta 2^0}|1\rangle) \tag{29}$$

where the the last term is the LSB of the c-register. This (29) can be represented as a summation

$$\sum_{k=0}^{N-1} e^{2\pi i\theta k}|k\rangle \tag{30}$$

The State after the Unitary rotation now has the following expression

$$|\psi_3\rangle = |b\rangle_b \left(\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1} e^{2\pi i\theta k}|k\rangle\right)_c |0\rangle_a \tag{31}$$

where $N = 2^n$.

The IQFT ($U_Q^\dagger$) is applied to the c-register only. Note that QFT and IQFT are just rotations that result in a change of basis.

$$U_Q^\dagger |k\rangle = \frac{1}{\sqrt{N}}\sum_{y=0}^{N-1} e^{\frac{-2\pi iyk}{N}}|y\rangle \tag{32}$$

$$\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1} e^{2\pi i\theta k}U_Q^\dagger |k\rangle = \frac{1}{\sqrt{N}}\sum_{y=0}^{N-1}\sum_{k=0}^{N-1} e^{-2\pi ik(\theta - \frac{y}{N})}|y\rangle \tag{33}$$

LHS of (33) will be 1 only when $y = N\theta$ otherwise it will be 0. We can now rewrite the LHS as

$$\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1} e^0 |N\theta\rangle \tag{34}$$

Therefore the state now becomes

$$|\psi_4\rangle = |b\rangle_b |N\theta\rangle_c |0\rangle_a \tag{35}$$

Because the eigenvectors of U and A are related by (14), U is also diagonal in A's eigenvector, $|u_i\rangle$ basis. So, if $|b\rangle = |u_j\rangle$, then

$$U|b\rangle = e^{i\lambda_j t}|u_j\rangle \tag{36}$$

By Comparing (28) and (36), we conclude that

$$\theta = \frac{\lambda_j t}{2\pi} \tag{37}$$

We define a scaled version of the eigenvalue λ_j as (38) and using (20) we can rewrite $|\psi_4\rangle$. Note that λ is not usually an integer, so the value of t is chosen such that $\tilde{\lambda}$ is an integer [11].

$$\tilde{\lambda_j} = \frac{N\lambda_j t}{2\pi} \tag{38}$$

$$|\psi_4\rangle = \sum_{j=0}^{M-1} b_j |u_j\rangle |\tilde{\lambda_j}\rangle |0\rangle_a \tag{39}$$

This concludes the QPE routine of HHL algorithm.

3.3 Ancillary Qubit Rotation

A subroutine to invert the eigenvalues, a controlled rotation [2][9] of the ancillary qubit is implemented. In [3], A is chosen such that its eigenvalues are powers of 2 in order to simplify the inversion.

$$|\psi_5\rangle = \sum_{j=0}^{M-1} b_j |u_j\rangle |\tilde{\lambda_j}\rangle \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}}|0\rangle_a + \frac{C}{\tilde{\lambda_j}}|1\rangle_a\right) \tag{40}$$

Where C is a constant. When the ancillary qubit is measured, the measurement would be either $|0\rangle$ or $|1\rangle$. The required measurement is $|1\rangle$ and all the other results will be ignored until $|1\rangle$ is measured. This is because from (40), the desired result i.e. inverted eigenvalue, is a part of the coefficient of $|1\rangle$. This inversion can be implemented using $Ry(\theta)$, where θ is given by (67). Note that this is a non-unitary operation.

$$|\psi_6\rangle = \frac{1}{\sqrt{\sum_{j=0}^{M-1} |\frac{b_j C}{\tilde{\lambda_j}}|^2}} \sum_{j=0}^{M-1} b_j |u_j\rangle |\tilde{\lambda_j}\rangle \frac{C}{\tilde{\lambda_j}} |1\rangle_a \tag{41}$$

After the rotation the state would be as shown in (41). Here, it is clear that C should be as large as possible because it determines the probability of obtaining $|1\rangle$. We can measure the ancillary qubit before or after the inverse quantum phase estimation.

3.4 Inverse Quantum Phase Estimation

After the measurement of the ancillary qubit, the b-register and the c-register are in an entangled state. We need IQPE to de-entangle these 2 registers. The solution, so far, is encoded as the amplitudes of eigenvector basis vectors $|u_j\rangle$, if we use this as the measurement basis then the solution will be correct. But we don't have a way to measure in the eigenvector basis. So, only after de-entangling can we measure the b-register in $|0\rangle$ and $|1\rangle$ basis.

QFT is applied on the c-register

$$U_Q |\tilde{\lambda_j}\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{\frac{2\pi i y \tilde{\lambda_j}}{N}} |y\rangle \tag{42}$$

The state after QFT is

$$|\psi_7\rangle = \frac{1}{\sqrt{\sum_{j=0}^{M-1} |\frac{b_j C}{\tilde{\lambda_j}}|^2}} \sum_{j=0}^{M-1} b_j |u_j\rangle \frac{C}{\tilde{\lambda_j}} (\frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{\frac{2\pi i y \tilde{\lambda_j}}{N}} |y\rangle) |1\rangle_a \tag{43}$$

The Inverse of the controlled unitary Rotations is applied, the process is the same except that the QPE Unitary Rotation of (14) is now U^{-1} as shown in (44)

$$U^{-1} = e^{-iAt} \tag{44}$$

Using similar arguments made in QPE and taking into consideration the b-register only, for simplicity we obtain (45)

$$\frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{-i\lambda_j t y} e^{\frac{2\pi i y \tilde{\lambda_j}}{N}} |y\rangle \tag{45}$$

We know, $\lambda_j t = 2\pi\theta$. Therefore, the two exponential term cancel each other out and the b-register becomes

$$\frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} |y\rangle \tag{46}$$

The complete state at this point is

$$|\psi_8\rangle = \frac{1}{\sqrt{\sum_{j=0}^{M-1} |\frac{b_j C}{\tilde{\lambda_j}}|^2}} \frac{1}{\sqrt{N}} \sum_{j=0}^{M-1} \frac{b_j C |u_j\rangle}{\tilde{\lambda_j}} \sum_{y=0}^{N-1} |y\rangle |1\rangle_a \tag{47}$$

Substituting the result from (24), we get

$$|\psi_8\rangle = \frac{C}{\sqrt{\sum_{j=0}^{M-1} |\frac{b_j C}{\tilde{\lambda_j}}|^2}} \frac{1}{\sqrt{N}} |x\rangle_b \sum_{y=0}^{N-1} |y\rangle |1\rangle_a \tag{48}$$

It is clear that, b-register and c-register are no longer entangled. $|x\rangle$ is now stored in the b-register.

We complete the IQPE by applying Hadamard Gates on the c-register. Using the result in (49), to simplify (48) we get,

$$(U_H \left|0\right\rangle)^{\otimes n} = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \left|y\right\rangle \tag{49}$$

$$\left|\psi_9\right\rangle = \frac{C}{\sqrt{\sum_{j=0}^{M-1} |\frac{b_j C}{\tilde{\lambda_j}}|^2}} \left|x\right\rangle_b \left|0\right\rangle_c^{\otimes n} \left|1\right\rangle_a \tag{50}$$

From, the result of (50), it can be deduced the constant term should be equal to 1, because $\left|x\right\rangle$, $\left|0\right\rangle$ and $\left|1\right\rangle$ are unit vectors. Hence, the final result is

$$\left|\psi_9\right\rangle = \left|x\right\rangle_b \left|0\right\rangle_c^{\otimes n} \left|1\right\rangle_a \tag{51}$$

4 Implementation Methodology

The HHL algorithm is implemented in python using the QISKIT library. First, the system is solved classically, then a simulation is performed using QASM, finally, the algorithm is run on a Quantum Computer.

A 2×2 system of equations is considered. The test cases have three A matrices, each with slightly different off diagonal entries. The $\vec{b}$ is the same for all the cases. This gives a total of three test cases.

$$A_1 = \begin{bmatrix} 1 & -\frac{1}{3} \\ -\frac{1}{3} & 1 \end{bmatrix} \tag{52}$$

$$A_2 = \begin{bmatrix} 1 & -\frac{1}{4} \\ -\frac{1}{4} & 1 \end{bmatrix} \tag{53}$$

$$A_3 = \begin{bmatrix} 1 & -\frac{1}{5} \\ -\frac{1}{5} & 1 \end{bmatrix} \tag{54}$$

$$b = \frac{1}{\sqrt{2}} \begin{bmatrix} -1 \\ 1 \end{bmatrix} \tag{55}$$

The $\vec{b}$ is a special case, because it is an eigenvector of all the A matrices in the test cases. This is important in verifying the algorithm's processing error. If the input to a system is the eigenvector of it's system matrix, then the output of the system is the same eigenvector but it is scaled by the respective eigenvalue. In our implementation, this case would yield equal probability of measuring either states.

The Quantum circuit has 4 qubits and 2 classical bits for measurements. There is one ancillary qubit, two qubits for the c-register and one qubit for the b-register. The ancillary qubit is the least significant qubit. In the case where the $\vec{b}$ is not normalized, it should be first normalized before initializing the b-register otherwise an error will occur.

There is some pre-processing required before the algorithm can be run. The steps are listed below:

1. Compute eigenvalues (λ) and eigenvectors (V) of A.
2. Set $\tilde{\lambda_1} = 1$.
3. Calculate the value of t using (56).
4. Calculate $\tilde{\lambda_2}$ using (57).
5. Calculate U, U^{-1}, U^2, U^{-2} using (59) and (60).
6. Calculate the required phases.

$$t = \frac{2\pi\tilde{\lambda_1}}{N\lambda_{min}} \tag{56}$$

Where N is the number of qubits in the quantum circuit.

$$\tilde{\lambda_2} = \frac{Nt\lambda_{max}}{2\pi} \tag{57}$$

From (14) and (21), the diagonalized matrix U_{diag} would be the matrix exponential of Λ as shown in (58). U and U^2 are calculated using (59) and (60).

$$U_{diag} = \begin{bmatrix} e^{i\lambda_o t} & 0 \\ 0 & e^{i\lambda_1 t} \end{bmatrix} \tag{58}$$

$$U = VU_{diag}V^{\dagger} \tag{59}$$

$$U^2 = VU_{diag}^2V^{\dagger} \tag{60}$$

The inverse of U is simply calculated using the builtin function of python numpy library. U^{-2} would be the same as U^2 because it is unitary.

To calculate the phases $(\phi, \lambda, \theta, \gamma)$ of the controlled unitary gates, consider the general form of the 2 qubit unitary matrix in (61). Note that γ is the global phase, since this is a multi-qubit sytems, we can't ignore the global phase.

$$U_{GEN} = \begin{bmatrix} e^{i\gamma}\cos(\frac{\theta}{2}) & -e^{i(\gamma+\lambda)}\sin(\frac{\theta}{2}) \\ e^{i(\gamma+\phi)}\sin(\frac{\theta}{2}) & e^{i(\gamma+\lambda+\phi)}\cos(\frac{\theta}{2}) \end{bmatrix} \tag{61}$$

This general form can be used to calculate the phases. Because U is unitary and hermitian i.e. its on-diagonal entries are same as well its off-diagonal entries, it would have the form of (62).

$$U = \begin{bmatrix} a & b \\ b & a \end{bmatrix} \tag{62}$$

Using (61) and (62), the expressions obtained for calculation of phases are shown in (63, 64, 65, 66).

$$\phi = \frac{\pi}{2} \tag{63}$$

$$\lambda = \phi + \pi \tag{64}$$

$$\theta = \frac{2\tan^{-1}(\frac{b}{a})}{e^{i\phi}} \tag{65}$$

$$\gamma = \frac{\ln(a) - \ln(\cos(\frac{\theta}{2}))}{i} \tag{66}$$

The phases for rotation of the ancillary qubit were calculated using (67)

$$\theta_a = 2\sin^{-1}\frac{1}{\bar{\lambda}} \tag{67}$$

Once the pre-processing is done, the HHL algorithm can be run. Note that the pre-processing part has to be done for each and every case the algorithm is fed. Inside the HHL algorithm, after initializing the b-register, QPE is done by first applying Hadamard Gates on the c-register followed by controlled unitary rotations on the b-register (controlled by c-register) followed by IQFT, which is a combination of Hadamard Gates, Controlled Phase Gates and Swaps on the c-register. Ancillary Qubit Rotation is performed using controlled RY gates and then measured. Then the Inverse Quantum Phase Estimation is carried out to finally yield our quantum circuit. The Quantum Circuit for one of the cases is shown in Figure 2.

5 Results

5.1 Classical Solver

These system of equations are solved by using the python numpy library. The resultant $\vec{x}$ are listed below.

$$\vec{x_1} = \begin{bmatrix} -0.5303 \\ 0.5303 \end{bmatrix} \tag{68}$$

$$\vec{x_2} = \begin{bmatrix} -0.5656 \\ 0.5656 \end{bmatrix} \tag{69}$$

$$\vec{x_3} = \begin{bmatrix} -0.5892 \\ 0.5892 \end{bmatrix} \tag{70}$$

Since HHL does not give an exact solution of $\vec{x}$, we are instead interested in the probability ratio of the entries of $\vec{x}$. The probability ratio obtained is 1 for all the cases i.e. the outcomes are equally likely.

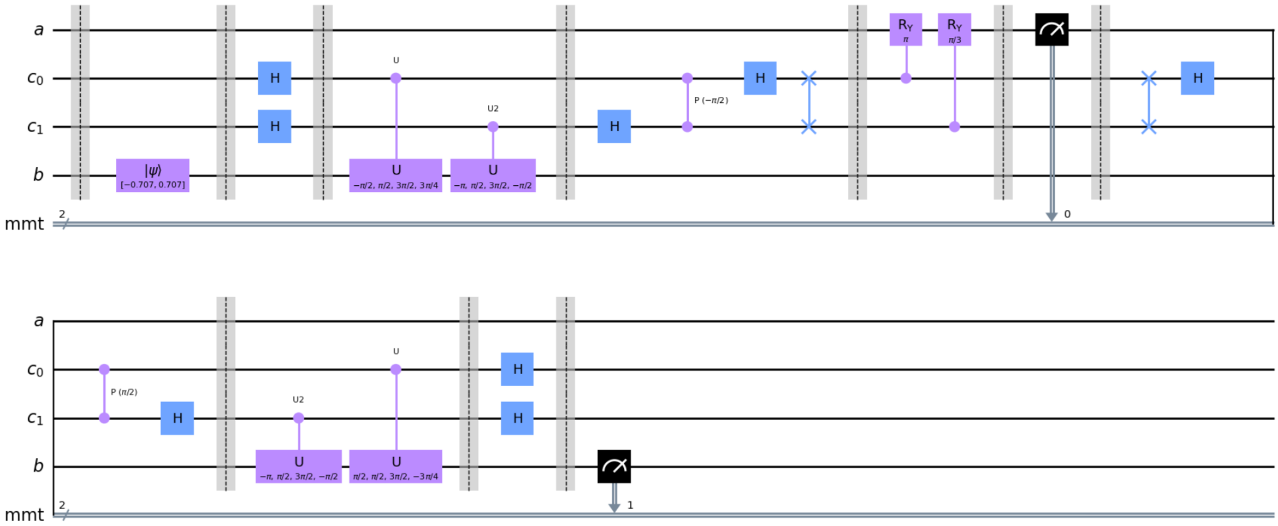
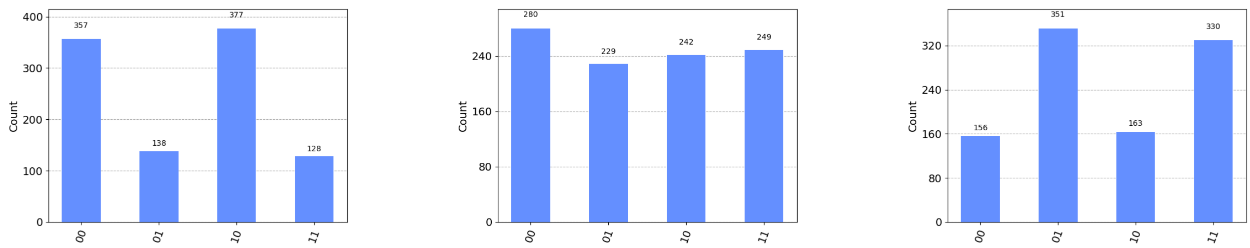


Figure 2: Quantum Circuit for Case 1



(a) Case 1

(b) Case 2

(c) Case 3

Figure 3: Quantum Simulator Histograms

5.2 Quantum Simulator

There are four possible outcomes, the b-register and the ancillary qubit states can be either $|0\rangle$ or $|1\rangle$. The four possible outcomes would be $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$. We consider the two outcomes where the ancillary qubit (LSB) is in $|1\rangle$ only i.e. the states $|01\rangle$ and $|11\rangle$. The relative magnitude of the possible solutions are reflected in the probability amplitudes of the outcomes obtained in the histogram. The histograms obtained from the simulator for each case are shown in Figure 3.

It is already established from the classical solution, the probability ratio of the 2 outcomes should be 1. The obtained probability ratios are 0.93, 1.09 and 0.94 for case 1,2 and 3, respectively.

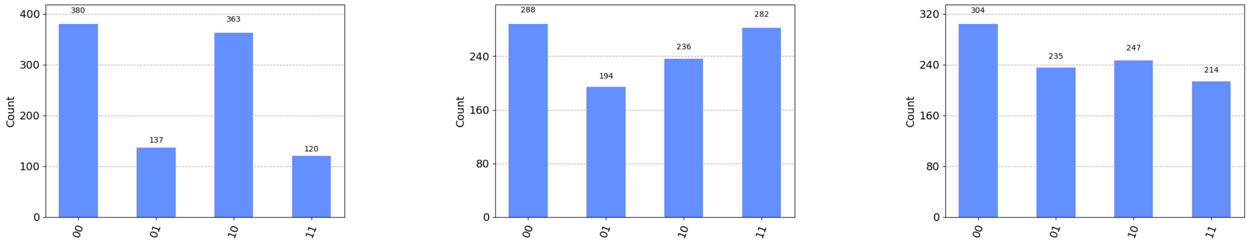
5.3 Quantum Computer

The histograms obtained for each case are shown in Figure 4. The probability ratios obtained are 0.88, 1.45 and 0.91 for case 1,2 and 3, respectively. The absolute error in obtained probability ratios is plotted against the ratio of eigenvalues (max over min) in Figure 5. The error is higher in the results obtained from a quantum computer. Since the hardware is imperfect and that some noise affects the hardware implementation, a higher error was expected.

5.4 Discussion

Figure 5 illustrates the eigenvalue ratio vs the absolute error (compared with the classical solution) in the two sets of results. The maximum percentage error for the simulator was 9%, this means the accuracy was above 90% which meets the requirements of solving linear system of equations [9]. But the percentage error in the results obtained from a quantum computer was between 9% and 45%, the performance was not satisfactory for all but 1 case.

If the histograms are observed case for case, the number of times the desired states ($|01\rangle$ and $|11\rangle$) are measured is either lower or comparable to the undesired states ($|00\rangle$ and $|10\rangle$). The probability of desired outcomes is lower than the undesired outcomes for all the results from the quantum computer whereas this is



(a) Case 1 (b) Case 2 (c) Case 3

Figure 4: Quantum Computer Histograms

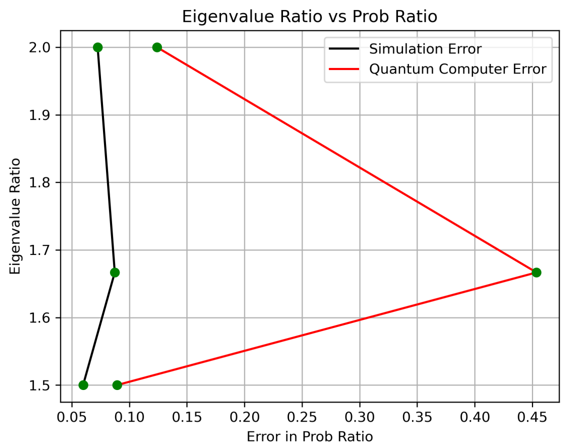


Figure 5: Eigenvalues Ratio vs Error in Probability

also untrue for Case 3 of the results from the simulator.

The error in the HHL algorithm stem from the Quantum Phase Estimation and the Eigenvalue Inversion Routine (Ancillary Qubit Rotation). Both of these routines are sensitive to small changes i.e. a small error in calculation of the phases can lead to a significant error in the outcome. The number of ancillary qubits used in implementing the algorithm, in this report, is 1, the precision of the eigenvalue inversion comes at a cost of increased number of ancillary qubits [12] [10]. However, increasing the number of ancillary qubits can introduce other sources of error. Other causes of errors can include quantum gate errors, decoherence of quantum states, state initialization errors and state measurement errors.

There are some methods to improve the accuracy in literature. In [13], an iterative algorithm is presented that can improve the accuracy of HHL algorithm as well as reduce computational complexity by making a series of simpler calculations. In [5], Quantum Singular Value Transformation is presented which can be used as an alternative to the eigenvalue inversion routine in HHL algorithm and it has been shown that, for certain type of matrices, the accuracy improves. The HHL requires postselection of the ancillary qubit, it has been shown in the work of [1] that the algorithm can work postselection-free, given that some conditions are satisfied, and reduces computational resource.

6 Conclusion

Linear system of equations is ubiquitous in data analysis. HHL allows us to explore the quantum mechanical advantages in solving such systems. In this report, the evolution of states is presented in mathematical fashion. The quantum circuit built using this algorithm is run on a simulator and a quantum computer. The results obtained from simulations were satisfactory while the results obtained from a quantum computer were not. Some possible sources of limitations in the algorithm along with some research work that has been done in improving the algorithm were also discussed.

References

[1] D. V. Babukhin. Harrow-hassidim-lloyd algorithm without ancilla postselection. *Physical Review A*, 107(4), apr 2023.

[2] Stefanie Barz, Ivan Kassal, Martin Ringbauer, Yannick Ole Lipp, Borivoje Dakic, Alán Aspuru-Guzik, and Philip Walther. Solving systems of linear equations on a quantum computer. *arXiv preprint arXiv:1302.1210*, 2013.

[3] Yudong Cao, Anmer Daskin, Steven Frankel, and Sabre Kais. Quantum circuit design for solving linear systems of equations. *Molecular Physics*, 110(15-16):1675–1680, aug 2012.

[4] Rati Chandra. *Conjugate gradient methods for partial differential equations.* Yale University, 1978.

[5] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. ACM, jun 2019.

[6] Joseph F. Grcar. How ordinary elimination became gaussian elimination. *Historia Mathematica*, 38(2):163–218, 2011.

[7] Lov Grover and Terry Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions. *arXiv preprint quant-ph/0208112*, 2002.

[8] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502, 2009.

[9] Xiaonan Liu, Lina Jing, Lin Han, and Jie Gao. Hhl analysis and simulation verification based on origin quantum platform. *Journal of Physics: Conference Series*, 2113(1):012083, nov 2021.

[10] Xiaonan Liu, Haoshan Xie, Zhengyu Liu, and Chenyan Zhao. Survey on the improvement and application of hhl algorithm. *Journal of Physics: Conference Series*, 2333(1):012023, aug 2022.

[11] Hector Jose Morrell Jr, Anika Zaman, and Hiu Yung Wong. Step-by-step hhl algorithm walkthrough to enhance the understanding of critical quantum computing concepts. *arXiv preprint arXiv:2108.09004*, 2021.

[12] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[13] Yoshiyuki Saito, Xinwei Lee, Dongsheng Cai, and Nobuyoshi Asai. An iterative improvement method for hhl algorithm for solving linear system of equations, 2021.