Solving linear systems using quantum algorithms: Review of HHL algorithm

Adrien Poncet
Physics Department D-PHYS, ETHZ[*]

(Dated: January 9, 2023)

The Harrow-Hassidim-Llyod algorithm is the first attempt to solve a linear system $Ax = b$ on a quantum computer. This method provides a exponential speed compared to its classical equivalent. In this manuscript, this algorithm is reviewed, and its strengths and weaknesses are briefly discussed.

CONTENTS

Introduction 1

I. Theory and background 1
A. Mathematical preliminaries 1
B. Quantum formalism 2

II. Subroutines 2
A. Quantum Fourier transform 2
B. Encoding 3
1. Basis encoding 3
2. Amplitude encoding 3
3. Hamiltonian encoding 3
C. Quantum random access memory 3
D. Hamiltonian simulation 4
1. Trotter-Suzuki 4
E. Quantum phase estimation 4
1. Quantum phase estimation in Hamiltonian simulation 5

III. HHL algorithm 6
A. Algorithm overview 6
B. Error bounds 7
C. Constrains, caveat and ameliorations 7

IV. Conclusion 8

Acknowledgments 9

Appendix 9
A. Numerical application 9
1. Quantum phase estimation simulation 9
2. Non-exact quantum phase estimation 9

References 10

INTRODUCTION

In the 21st century, the development of numerical methods and the increasing computational power allowed the computation of datasets of larger and larger size. For example, the LHC experiment at CERN produces annually 90 Pb [1]. All this data must be filtered, analyzed and studied. Solving a linear system is an essential building block in the solution of more complex problems and a major breakthrough in these kind of algorithms could have numerous consequences in other problems.

Classically, the conjugate gradient (CG) is the standard method for some classes of matrices [2]. Since 1981 and the Feynmann's famous conference [3], quantum computing gives a new perspective in order to have exponentially faster algorithms than classical ones. In 2006, Harrow, Hassidim and Lloyd developed a quantum algorithm to solve linear systems with an exponential speedup compared to classical algorithm for some class of matrices [4].

The report will present the Hassidom-Harrow-Lloyd algorithm (HHL). The aim of this work is to explain this algorithm. In section I, the mathematical and physical background are presented. In section II, the subroutines of the HHL algorithm are studied; they constitute the building blocks of the method. In section III, the HHL algorithm is presented and the different constrains and caveats are analyzed.

I. THEORY AND BACKGROUND

In this section, the mathematical theory including linear algebra and physical formalism are briefly discussed.

A. Mathematical preliminaries

Let $A \in \mathbb{R}^{N\times N}$ be a matrix [5]. Moreover let $b \in \mathbb{R}^N$ and $x \in \mathbb{R}^N$ be vectors. Let us define the matrix equation, called *linear system* or *linear system problem* (LSP) as:

$$Ax = b. \tag{1}$$

Given A and b, the goal is to find x. The solution can be stated as $x = A^{-1}b$. Numerically, the goal is to find an approximation of the solution x, denoted by $\tilde{x}$, satisfying $\|x - \tilde{x}\| < \varepsilon$ for a fixed ε. (In this report, any numerical approximation of a quantity α will be denoted by a tilde

* adrien.poncet@student.ethz.ch

$\tilde{\alpha}$.) It is also possible to cast equation (1) into the form:

$$C\bar{x} = \bar{b} \Longleftrightarrow \begin{pmatrix} 0 & A \\ A^\dagger & 0 \end{pmatrix} \begin{pmatrix} 0 \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}. \quad (2)$$

where C is a Hermitian matrix. Solving equations (1) and (2) are mathematically equivalent.

The *condition number* κ of an invertible matrix A is defined as:

$$\kappa = \left| \frac{\lambda_{\max}}{\lambda_{\min}} \right|.$$

Let s be the maximal number of nonzero components per line of the matrix $A \in \mathbb{R}^{N\times N}$. Then A is called $s-$*sparse* if $s \ll N$ meaning that the nonzero entries are negligible compared to the total number of elements of the matrix.

B. Quantum formalism

In order to use quantum formalism, classical vectors must be transformed in quantum states. In theory, quantum states are expressed in term of the classical vectors as:

$$|b\rangle = \frac{b}{\|b\|}, \qquad |x\rangle = \frac{x}{\|x\|}.$$

In practise, this task is non-trivial and will be discussed later on section II B . Hence the classical system (1) is transformed into its quantum equivalent called *quantum linear solver problem* (QLSP):

$$A\,|x\rangle = |b\rangle\,. \quad (3)$$

Now let us suppose that A is a Hermitian matrix. Let $\{|u_1\rangle, \dots, |u_N\rangle\}$ be the eigenbasis of A and $\{\lambda_1, \dots, \lambda_N\}$ be the associated eigenvalues. By spectral theorem [6], A can be expressed as:

$$A = \sum_{j=0}^{N-1} \lambda_j \,|u_j\rangle \langle u_j|\,.$$

Moreover if A is invertible, then its inverse can be written as:

$$A^{-1} = \sum_{j=0}^{N-1} \frac{1}{\lambda_j} \,|u_j\rangle \langle u_j|\,.$$

Any integer number x expanded on m bits is represented by a *binary expansion* as:

$$x = \sum_{i=0}^{m-1} x_i 2^i + \delta,$$

where $x_i \in \{0,1\}$ for all $i \in \{1, \dots, m-1\}$ and $\delta \geq 0$ is an error term. In terms of quantum states, we use several notations:

$$|x\rangle = |x_{m-1}\rangle \otimes \cdots \otimes |x_0\rangle = |x_{m-1} \dots x_0\rangle = |x\rangle_m\,, \quad (4)$$

to express the binary expansion of x on the computational basis.

II. SUBROUTINES

In this section, the different subroutines needed for the HHL algorithms are briefly discussed. First, the quantum Fourier transform is reviewed. Second, the different encoding methods used in HHL are presented. Finally, the quantum phase estimation is described and used to explain the Hamiltonian simulations in different cases.

A. Quantum Fourier transform

Let $\omega_N = e^{\frac{2\pi}{N}i}$ be the primitive root of the unity. Let $g = (x_0, \dots, x_{N-1}) \in \mathbb{R}^N$ be a vector. The *discrete Fourier transform* (DFT) maps the vector x into the vector $\hat{g}$ whose components are:

$$\hat{g}_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \omega_N^{jk}. \quad (5)$$

The idea is to generalize the DFT for quantum states. To define the *Quantum Fourier transform* (QFT), one should apply the map in equation (5) to a quantum state $|x\rangle$. Let us suppose that the numerical result of the QFT is stored on m-bits. As notation, $|x_{m-1}\dots x_0\rangle$ (resp. $|k_{m-1}\dots k_0\rangle$) stands for the binary expansion of x (resp. k). Then, the number of states is $N = 2^m$ and it yields:

$$\begin{aligned}
\mathrm{QFT}\,|x\rangle &= \frac{1}{2^{\frac{m}{2}}} \sum_{k=0}^{N-1} \omega_N^{xk}\,|k\rangle = \frac{1}{2^{\frac{m}{2}}} \sum_{k=0}^{N-1} e^{2\pi i \frac{kx}{2^m}}\,|k\rangle\,, \\
&= \frac{1}{2^{\frac{m}{2}}} \sum_{k=0}^{N-1} e^{2\pi i x\left(\sum_{l=0}^{m-1} 2^{l-m} k_l\right)}\,|k\rangle\,, \\
&= \frac{1}{2^{\frac{m}{2}}} \sum_{k=0}^{N-1} \prod_{l=0}^{m-1} e^{2\pi i x k_l 2^{l-m}}\,|k\rangle\,, \\
&= \frac{1}{2^{\frac{m}{2}}} \sum_{k=0}^{N-1} \bigotimes_{l=0}^{m-1} e^{2\pi i x k_l 2^{l-m}}\,|k\rangle\,,
\end{aligned}$$

$$= \frac{1}{2^{\frac{m}{2}}} \sum_{k_{m-1}=0}^{1} \cdots \sum_{k_0=0}^{1} \prod_{l=0}^{m-1} e^{2\pi i x k_l 2^{l-m}} \left| k_{m-1} \ldots k_0 \right\rangle ,$$

$$= \frac{1}{2^{\frac{m}{2}}} \bigotimes_{l=0}^{m-1} \sum_{k_l=0}^{1} e^{2\pi i x k_l 2^{l-m}} \left| k_l \right\rangle ,$$

$$= \frac{1}{2^{\frac{m}{2}}} \bigotimes_{l=0}^{m-1} \left(|0\rangle + e^{2\pi i x 2^{l-m}} |1\rangle \right) .$$

In terms of matrix, the QFT matrix is explicitly defined as:

$$\mathrm{QFT} = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \sum_{i=0}^{N-1} \omega_N^{ik} |k\rangle \langle i| .$$

B. Encoding

In this section, the preparation of states is discussed, that is several manners on how classical information is embedded into quantum states. Note that encoding is an active field of research. For simplicity, it is assumed that the classical data are already encoded in binary. Indeed, there are three different common ways of encoding classical information that will be used in the HHL algorithm.

1. *Basis encoding*

The basis encoding consists of mapping a classical binary string into the computational basis of qubits. For example, consider $x = 5$ in its binary expansion: $x = 5 \to 101$. Then, the associated quantum state is $|101\rangle$. Moreover it is also possible to encode vectors. Let $y = (1, 2)$, then in binary it is: $y_{\text{bin}} = (01, 10)$. Then, the vector y_{bin} can be flattened and store into a single state $y_{\text{fin}} = |0110\rangle$. The advantage of this method is that it is straightforward to map classical to quantum. At the beginning, all registers of a quantum computer are in the zero state. Thus, in order to prepare a state to the desired configuration, NOT gates (Pauli matrix σ_x) are used. However, it is not efficient. As a "classical bit" is represented by a quantum single qubit, however; it needs a large number of qubits just to represent classical data [7]. There exist more efficient methods as amplitude encoding Table I.

2. *Amplitude encoding*

The amplitude consists of encoding the information in the amplitude of computational basis. For example a 2-dimensional array $x = (x_0, x_1)$ is encoded as

$$|x\rangle = \frac{1}{\sqrt{x_0^2 + x_1^2}} \left(x_0 |0\rangle + x_1 |1\rangle \right) . \qquad (6)$$

Note that the state is normalized and in theory the coefficient x_0 and x_1 are of infinite precision. The main advantage is that it is possible to store data in $\lceil \log_2(NM) \rceil$ where N is the size of the array and M is the number of arrays [8]. This applies also to any matrix $A \in \mathbb{R}^{N\times N}$ which fulfills $\sum_{i,j=0}^{N-1} |a_{ij}|^2 = 1$. That kind of matrix can be stored in a state $|\psi_A\rangle$:

$$|\psi_A\rangle = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} a_{ij} |i\rangle |j\rangle . \qquad (7)$$

In the QFT algorithm, the amplitude encoding allows us to store any function f in the amplitude of a m-qubits state $|\psi_f\rangle$ to compute its QFT:

$$|\psi_f\rangle = \sum_{x=0}^{2^m-1} f(x) |x\rangle ,$$

where x is encoded in binary on m-bits and $|x\rangle$ is expressed in the computational basis [9].

It is possible to store data in $\lceil \log_2(NM) \rceil$ [8], however, in order to rebuild entirely the initial vector, one needs to store one more bit of information which corresponds to the value of the norm of the vector before the normalisation. Thus, one needs $\lceil \log_2(NM) \rceil + 1$ (For a more complete discussion see [10]). Finally, compared with the basis encoding, it is an exponential improvement in terms of space density Table I. In practice, it is challenging to prepare such a state [11].

3. *Hamiltonian encoding*

The Hamiltonian encoding consists of encoding the data in a ground state of a physical model. For example, the Ising model [12] can be used and is given by:

$$H = -\sum_{\langle ij \rangle} J_{ij} \sigma_i \sigma_j - \sum_j h_j \sigma_j . \qquad (8)$$

The information is stored in the coupling constant J_{ij} and in the field h_j. Loading those classical data is equivalent to solving the Hamiltonian H. However, the Ising Hamiltonian (8) cannot be always used to store the classical information. Nevertheless, it remains useful in a large number of problems [13].

C. Quantum random access memory

In the previous section, the representation of data was discussed. In this section, the discussion is about how can one access or load these data. In a classical computer, the ability to access stored information is done through the *random access memory* (RAM). For quantum computers, one can use the same process but using qubits

Encoding	Number of qubits	Runtime
Basis	Nm	$\mathcal{O}(Nm)$
Amplitude	$\log_2(N)$	$\mathcal{O}(\log(N))$
Hamiltonian	$\log_2(N)$	$\mathcal{O}(\log(NM))$

Table I. *Qubits and runtime complexity for basis, amplitude and Hamiltonian encoding. N (resp. NM) denotes the size of the vector (resp. matrix), m the number of bits of the approximation. A single qubit should be added for every method to store the normalization constant in order to obtain the initial state [10].*

instead of bits. Such a device is called *quantum random access memory* (qRAM) . The aim of the qRAM is to return quantum states in superposition of the classical data stored in the memory to be run by a program or for a certain tasks [11]. The number of memory calls for qRAM is $\mathcal{O}(\log(N))$ where its classical equivalent needs $\mathcal{O}(N)$ calls. One major restriction in the use of qRAM is that in order to have a time complexity which is logarithmic, components of stored data must be relatively uniform meaning without values which are much more larger than the other. qRAM play an important role in the exponential speed of the HHL.

D. Hamiltonian simulation

The *Hamiltonian simulation* consists to find a best approximation of a state $|\psi(t)\rangle$ at a time t given a state with initial value $|\psi(t=0)\rangle$. In theory, the time evolution operator is given by:

$$U(t) = e^{-iHt}. \tag{9}$$

However, on a quantum computer, one needs to build an approximation of (9). More formally, let $\varepsilon > 0$ be the error at time t and H be a Hamiltonian. Then, the goal is to find an approximation unitary operator $\tilde{U}$ such that:

$$\|\tilde{U} - e^{-iHt}\| < \varepsilon. \tag{10}$$

Such an operation on a classical computer has a complexity of the order $\mathcal{O}(2^N)$ where N is the number of quantum variables. On the other hand, it has been shown that a quantum computer needs $\mathcal{O}(N)$ steps to achieve such a computation [14]. In particular, Hamiltonian simulation plays a central role in the exponential speed up achieved by HHL algorithm.

A Hamiltonian H represented by a matrix of size N is said to be *k-local* if it can be decomposed as:

$$H = \sum_{i=1}^{m} H_i, \tag{11}$$

where every matrix H_i acts at most on k qubits. In particular, this implies that H is sparse, there is at least $\mathcal{O}(mk)$ non zero entries. The idea is to break the Hamiltonian in some "easy" pieces to perform the simulation efficiently. It has be shown that k-local Hamiltonians are efficiently simulated by a quantum computer [15]. This idea with the QPE is the key of the exponential speedup of the HHL algorithm.

1. Trotter-Suzuki

Let us suppose, there are two k-local Hamiltonians H_1, H_2. As the two Hamiltonians do not necessarily commute (if the two Hamiltonians commute, then the formula is exact by Baker-Campbell-Hausdorff formula), one must use the Lie-Product formula [16]:

$$e^{-it(H_1+H_2)} = \lim_{n\to\infty}\left(e^{-it\frac{H_1}{n}}e^{-it\frac{H_2}{n}}\right)^n.$$

and:

$$\lim_{n\to\infty}\left(e^{-\frac{iH_1t}{n}}e^{-\frac{iH_2t}{n}}\right)^n + \mathcal{O}\left(\frac{t^2}{n}\right). \tag{12}$$

In order to have a maximal error of ε, one can choose $n = \mathcal{O}\left(\max(\|H_1\|, \|H_2\|)\, t^2/\varepsilon\right)$ [17]. One can generalize (12) to the sum (11), and obtain:

$$\begin{aligned} e^{-iHt} &= e^{-it\sum_{i=q}^{m} H_i} = \left(e^{-it\frac{H_1}{n}} \dots e^{-it\frac{H_m}{n}}\right)^n, \\ &= e^{-iH_1t} \dots e^{-iH_mt} + \mathcal{O}\left(\frac{t^2}{n}\right). \end{aligned}$$

The Hamiltonian simulation is the heart of the exponential speedup of the HHL algorithm. Today, there exists lots of algorithms such as quantum walk, product formulas of different orders, and others Table III.

E. Quantum phase estimation

Let U be a unitary operator and let λ be an eigenvalue associated to the eigenvector $|u\rangle$. Since U is unitary, λ is on the complex unit circle and can be written as $\lambda = e^{2\pi i\theta}$ for some $\theta \in [0,1)$. The goal of the *quantum phase estimation* (QPE) is to estimate the value of θ. The principle is the following: the value of the phase θ is stored in m qubits register called *counting register*. On the other side, the eigenvector is stored in a s qubit register. By applying control rotations, one can encode the value of the phase on m qubits. By applying an inverse QFT, one can extract the approximation of the phase $\tilde{\theta}$ and reading the result by measuring the states Fig. 1.

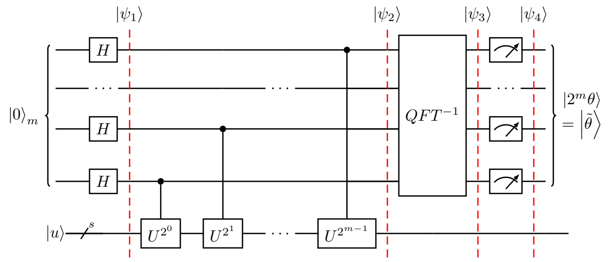

Fig. 1. *Quantum phase estimation algorithm (QPE): Inspired from a diagram of [18].*

More formally, the state is initially $|\psi_0\rangle = |0\rangle_m |u\rangle$ (see Fig. 1). Hadamard gates are applied to $|0\rangle_m$. Then, the state is placed on a superposition $|\psi_1\rangle = \frac{1}{2^{\frac{m}{2}}} (|0\rangle + |1\rangle)^{\otimes m} |u\rangle$. Then powers of the control unitary operator U are applied on the state $|u\rangle$ and yield:

$$\begin{aligned} U^{2^m} |u\rangle &= U^{2^{m-1}} U |u\rangle , \\ &= U^{2^{m-1}} \left(e^{2\pi i\theta} |u\rangle\right) , \\ &= \cdots = e^{2\pi i 2^m \theta} |u\rangle . \end{aligned}$$

Next, the control gate U^{2^j} applied on a state $\frac{1}{\sqrt{2}} (|0\rangle + |1\rangle)$ gives:

$$U^{2^j} \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}} \left(|0\rangle + e^{2\pi i\theta 2^j} |1\rangle\right) ,$$

for all $j \in \{0, \ldots, m-1\}$. Putting things together we obtain

$$|\psi_2\rangle = \frac{1}{2^{\frac{m}{2}}} \bigotimes_{j=0}^{m-1} \left(|0\rangle + e^{2\pi i\theta 2^j} |1\rangle\right) |u\rangle , \tag{13}$$

$$= \frac{1}{2^{\frac{m}{2}}} \sum_{k=0}^{2^m-1} e^{2\pi i\theta k} |k\rangle |u\rangle . \tag{14}$$

where $|k\rangle = |k_m \ldots k_0\rangle$. To manipulate the register, let us apply an inverse QFT:

$$\begin{aligned} |\psi_3\rangle &= \text{QFT}^{-1} |\psi_2\rangle , \\ &= \frac{1}{2^n} \sum_{x=0}^{2^m-1} \sum_{k=0}^{2^m-1} e^{-\frac{2\pi i k}{2^m}(x-2^n\theta)} |x\rangle |u\rangle , \end{aligned}$$

which peak at $x = 2^m\theta$. It can be shown that x will peaks with a probability of at least $\frac{4}{\pi^2} \approx 0.4$ [18]. Thus, by measuring the first register of the state $|\psi_4\rangle$:

$$|\psi_4\rangle = |2^m\theta\rangle |u\rangle \approx \left|\tilde{\theta}\right\rangle |u\rangle .$$

One obtains the value of x (x is $\tilde{\theta}$, which is the approximation on m-bits of θ). If the $\tilde{\theta}$ is an integer then the result is exact. Otherwise, $\tilde{\theta}$ is the best approximation of $2^n\theta$. (See section IV A 2)

One remaining question is how it is possible to have an eigenvector at the beginning of the process. It turns out that it is not necessary to start with an eigenstate. Let us suppose that $|\psi\rangle$ is not an eigenstate. But one can express any state in terms of an eigenbasis:

$$|\psi\rangle = \sum_{j=0}^{N-1} c_j |u_j\rangle .$$

And let θ_j be the phase of the eigenvalue associated to $|u_j\rangle$. Doing the QPE as before, it yields:

$$\begin{aligned} |\psi_4\rangle &= |2^m\theta\rangle \sum_{j=0}^{N-1} c_j |u_j\rangle , \\ &= \sum_{j=0}^{N-1} c_j |2^m\theta\rangle |u_j\rangle , \\ &= \sum_{j=0}^{N-1} c_j \left|\tilde{\theta}_j\right\rangle |u_j\rangle . \end{aligned}$$

When the state is measured, the eigenvalue $2^m\lambda_j$ is obtained and associated with $|u_j\rangle$. The associated probability is $|c_j|^2 \geq \frac{4}{\pi^2}$. It is not possible to select a single j-th eigenvalue during the measure. Hence, it is not necessary to start the QPE with an eigenvector with a single eigenvector.

1. *Quantum phase estimation in Hamiltonian simulation*

The Hamiltonian simulation allows us to approximate any time evolution of an Hamiltonian by a unitary matrix (9). But QPE allow us to find eigenvalues of any unitary matrix. Hence, it is possible to find eigenvalues of the Hamiltonian H. The QPE defined in the previous section must be slightly modified in order to achieve the goal.

In order to simplify computations, it is assumed that $N = 2^m$. Let $U \in \mathbb{C}^{2^m \times 2^m}$ be a unitary matrix and let $|\psi\rangle$ be an eigenvector associated to the eigenvalue $e^{2\pi i\theta}$. The goal is to estimate the eigenvalue λ_j of the Hamiltonian H on m-qubits. Let $U \in \mathbb{C}^{2^m \times 2^m}$ unitary be defined by $U = e^{-itH}$. Let $|u_j\rangle$ be an eigenstate of H associated to the eigenvalue $e^{2\pi i\lambda_j}$. Then:

$$e^{-iHt} |u_j\rangle = e^{-i\lambda_j t} |u_j\rangle . \tag{15}$$

But, comparing with:

$$U |\psi\rangle = e^{2\pi i\theta} |\psi\rangle , \tag{16}$$

after applying QPE, one obtains the following approximation:

$$\tilde{\theta} = 2^m\theta.$$

At the end comparing (15) and (16) yields [19]:

$$2\pi i\theta = i\lambda_j t \iff \theta = \frac{\lambda_j t}{2\pi}.$$

Thus:

$$\tilde{\theta} = \frac{\tilde{\lambda}_j t}{2\pi} 2^m. \tag{17}$$

III. HHL ALGORITHM

The HHL algorithm allows us to solve in a quantum manner the QLSP (3). In this section, the HHL will be studied in details. First, the algorithm is reviewed. Second, the complexity of the algorithm is studied and finally the feasability is studied.

The idea is to extract the eigenvalue using QPE and Hamiltonian simulation. In the eigenbasis, the matrix is diagonal and its inverse in this basis is just the inverse of each eigenvalue. The main difference with a classical algorithm is that this task can be achieved in a logarithmic number of operations.

The HHL requires some important hypothesis in order to achieve the exponential speedup. First the matrix A of the LSP (1) must be s-sparse. Without loss of generality, one can assume that A is Hermitian (otherwise the system is transformed into (2). In the following sections, the matrix is supposed Hermitian and sparse.

A. Algorithm overview

Three registers are needed to run the algorithm. One *ancilla register* of 1 qubit, a second register of n_l qubits to store the value of the approximation of the eigenvalue called *clock register* and a third register of $n_b = N$ qubits called *input register* to store the solution of the linear system. Initially, the system is prepared in the state Fig. 2 :

$$|\psi_0\rangle = |0\rangle |0\rangle_{n_l} |0\rangle_{n_b}.$$

Using amplitude encoding section II B 2, the vector b is cast into $|b\rangle$. The state $|b\rangle$ is decomposed in terms of $\{|u_j\rangle\}_{j=0}^{N}$, the eigenvectors of A:

$$|b\rangle = \sum_{i=0}^{N-1} \beta_j |u_j\rangle,$$

where $\beta_j = \langle u_j|b\rangle$ for all $j \in \{0, \dots, N-1\}$. After loading the vector b, the state is:

$$|\psi_1\rangle = |0\rangle |0\rangle_{n_l} |b\rangle_{n_b} = \sum_{j=0}^{N-1} |0\rangle |0\rangle_{n_l} \beta_j |u_j\rangle_{n_b}.$$

Then, using the method of QPE and Hamiltonian simulation section II E 1. More formally, applying QPE to the matrix e^{-iAt} yields:

$$|\psi_2\rangle = |0\rangle \left|\tilde{\lambda}_j\right\rangle_{n_l} |b\rangle_{n_b} = \sum_{j=0}^{N-1} \beta_j |0\rangle \left|\tilde{\lambda}_j\right\rangle_{n_l} |u_j\rangle_{n_b}.$$

A controlled rotation is then applied on the ancilla of an angle $\theta = 2\sin^{-1}\left(\frac{C}{\tilde{\lambda}_j}\right)$. This transfers the information of the eigenvalue in the ancilla. Depending on the method to approximate θ, it is possible to choose C such that $\frac{C}{\tilde{\lambda}_j} \in (-1, 1)$ see [20]. Then, the state $|\psi_2\rangle$ is transformed to the state $|\psi_3\rangle$:

$$|\psi_3\rangle = \sum_{j=0}^{N-1} \beta_j \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}} |0\rangle + \frac{C}{\tilde{\lambda}_j} |1\rangle\right) \left|\tilde{\lambda}_j\right\rangle_{n_l} |u_j\rangle_{n_b}.$$

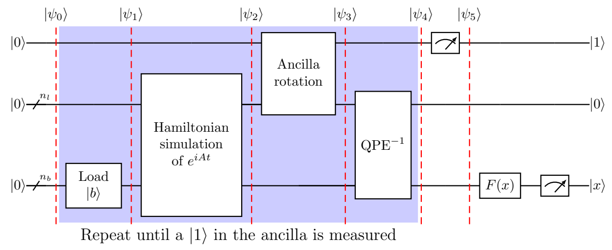

Fig. 2. *Schematic view of the main steps of the HHL algorithm.*

In order to avoid having a mixed state, the state $|\psi_3\rangle$ is uncomputed by applying an inverse QPE (This is also known as *uncompute trick*, see [21]). It yields:

$$|\psi_4\rangle = \sum_{j=0}^{N-1} \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}} |0\rangle + \frac{C}{\tilde{\lambda}_j} |1\rangle\right) |0\rangle_{n_l} \beta_j |u_j\rangle_{n_b}.$$

At this step, a measurement is performed. If a $|0\rangle$ is measured, the process is repeated until a $|1\rangle$ is measured. After the successful measurement, the state is:

$$\begin{aligned} |\psi_5\rangle &= \frac{1}{\mathcal{N}} \sum_{j=0}^{N-1} \frac{1}{\tilde{\lambda}_j} |1\rangle |0\rangle_{n_l} \beta_j |u_j\rangle_{n_b}, \\ &= \frac{1}{\mathcal{N}} \sum_{j=0}^{N-1} |1\rangle \left(\underbrace{\frac{1}{\tilde{\lambda}_j} |u_j\rangle \langle u_j|}_{\approx A^{-1}}\right) |b\rangle_{n_b} \\ &= \frac{1}{\mathcal{N}} \sum_{j=0}^{N-1} |1\rangle |0\rangle_{n_l} |\tilde{x}\rangle_{n_b}. \end{aligned}$$

Where $\mathcal{N} = \sqrt{\sum_{j=0}^{N-1} \frac{|\beta_j^2|}{|\lambda_j|^2}}$ is the normalisation constant. It possible to apply a function F in order to probe some properties of the state $|\tilde{x}\rangle$.

B. Error bounds

In order to obtain a bound for the Hamiltonian simulation it is possible to start with an different initial vector $|\psi_0\rangle$. Instead of initializing the clock register to $|0\rangle$ we initialize it at:

$$|\psi_0\rangle = |0\rangle \sqrt{\frac{2}{T}} \sum_{\tau=0}^{T-1} \sin\left(\frac{\pi\left(\tau+\frac{1}{2}\right)}{T}\right) |\tau\rangle_{n_l} |0\rangle_{n_b} .$$

where $T = \mathcal{O}\left(\log\left(N\right) s^2 t_0\right)$ is the number of computational steps to perform Hamiltonian simulation for some time $0 \le t \le t_0$ and t_0 is a parameter chosen at the end to minimize the error ε. The ratio $\frac{t_0}{T}$ is the step size of the simulation.

Then, the conditional Hamiltonian simulation $\mathbb{1} \otimes \sum_{\tau=0}^{T-1} |\tau\rangle\langle\tau| \otimes e^{iA\tau t_0/T}$ is applied to the input state $|0\rangle \otimes |\psi_0\rangle \otimes |b\rangle$ resulting:

$$|\psi_2\rangle = |0\rangle \left(\sum_{\tau=0}^{T-1} |\tau\rangle\langle\tau|\psi_0\rangle e^{iAt_0\tau/T} |0\rangle\right).$$

But:

$$\langle\tau|\psi_0\rangle = \sqrt{\frac{2}{T}} \sum_{\gamma=0}^{T-1} \sin\left(\frac{\pi\left(\gamma+\frac{1}{2}\right)}{T}\right) \underbrace{\langle\gamma|\tau\rangle}_{=\delta_{\gamma\tau}} .$$

Hence, the previous simplify in:

$$|\psi_2\rangle = |0\rangle \otimes \sum_{\tau=0}^{T-1} \sqrt{\frac{2}{T}} \sin\left(\frac{\pi\left(\tau+\frac{1}{2}\right)}{T}\right) |\tau\rangle ,$$
$$\otimes \sum_{j=0}^{N-1} \beta_j e^{-i\frac{t_0\tau}{T}\lambda_j} |u_j\rangle .$$

By applying a change of basis into the Fourier basis $|\tau\rangle = \sum_{k=0}^{N-1} \omega_N^{\tau k} |k\rangle$, it yields :

$$|\psi_2\rangle = \sum_{j=1}^{N} \beta_j \sum_{k,\tau=0}^{T-1} \left[\frac{\sqrt{2}}{T} \sin\left(\frac{\pi\left(\tau+\frac{1}{2}\right)}{T}\right) e^{i\frac{\tau}{T}(2\pi k - t_0)}\right]$$
$$\times |0\rangle |k\rangle |u_j\rangle$$
$$= \sum_{j=0}^{N} \beta_j \sum_{k=0}^{T-1} \alpha_{k|j} |k\rangle |u_j\rangle .$$

where:

$$a_{k|j} = \frac{\sqrt{2}}{T} \sum_{\tau=0}^{T-1} e^{i\frac{\tau}{T}(\lambda_j t_0 - 2\pi k)} \sin\left(\frac{\pi\left(\tau+\frac{1}{2}\right)}{T}\right).$$

Let us define $\delta = |\lambda_j t_0 - 2\pi k|$. Then, it is possible to find an upper bound for $a_{k|j}$ where:

$$|a_{k|j}| \le \frac{8\pi}{\delta}.$$

if $|k - \frac{\lambda_j t_0}{2\pi}| \ge 1$ (Full proof can be found [4]).
Then, if $|a_{k|j}| \gg 1$ is large, then $\delta \ll 1$. Hence $\tilde{\lambda}_j \approx \frac{2\pi k}{t_0}$:

$$|\psi_2\rangle = |0\rangle \sum_{j=0}^{N-1} \beta_j \sum_{k=0}^{T-1} \alpha_{k|j} \left|\tilde{\lambda}_j\right\rangle |u_j\rangle .$$

Note that if the QPE will be exact, then $|a_{k|j}| = 1$. This result is coherent with (13).

C. Constrains, caveat and ameliorations

Let us compute the time complexity of the HHL algorithm. The probability of success at each iteration is $p = \mathcal{O}\left(\frac{1}{\kappa^2}\right)$. It can be shown that one needs to perform the loop $\mathcal{O}\left(\kappa\right)$ times [21]. In order to load the vector b one needs $\mathcal{O}\left(T_b\right)$. The time duration of the Hamiltonian simulation is [22]:

$$T = \tilde{\mathcal{O}}\left(t_0 s^2 \log\left(N\right)\right). \quad (18)$$

Where $\tilde{\mathcal{O}}$ means that the slower growing terms are neglected. Hence, the total runtime is $\tilde{\mathcal{O}}\left(\kappa\left(T_b + t_0 s^2 \log\left(N\right)\right)\right) \sim \tilde{\mathcal{O}}\left(\kappa T_b + \kappa t_0 s^2 \log\left(N\right)\right)$. As $t_0 = \mathcal{O}\left(\frac{\kappa}{\varepsilon}\right)$ one can replace in (18). Which gives:

$$\tilde{\mathcal{O}}\left(\kappa T_b + \frac{\kappa^2 s^2}{\varepsilon} \log\left(N\right)\right).$$

One can see that the state preparation can be a important bottleneck. But using, qRAM one can load data in polynomial time. Thus, the total complexity is:

$$\tilde{\mathcal{O}}\left(\frac{\kappa^2 s^2}{\varepsilon} \log\left(N\right)\right).$$

A classical computer cannot solve the problem in less than $\mathcal{O}\left(N^2\right)$ because just reading the coefficient of a matrix A of size $N \times N$ needs $\mathcal{O}\left(N^2\right)$ operations. However, to obtain a speedup, strict conditions should be fulfilled [23]. Four main caveats are associated with the HHL algorithm and are:

1. The condition number κ of A.
2. The preparation of the state $|b\rangle$.
3. The Hamiltonian simulation e^{iAt}.
4. The solution $|\tilde{x}\rangle$ is a quantum state.

First, the matrix must be, as in any classical methods, robustly invertible, meaning that the condition number κ is not large. In other words that no eigenvalues are close to zero. Let $\lambda \ll 0$ be an eigenvalue of A. By computing, the inverse of the matrix, then diagonal elements of A^{-1} are $1/\lambda$. Thus a small change in λ leads to a large change in $1/\lambda$. Suppose, there is an eigenvalue $\lambda = \frac{\kappa}{\varepsilon_k}$ such that

$0 < \varepsilon \ll 1$. Then: $\frac{1}{\lambda} = \frac{\kappa}{\varepsilon_k} \gg 1$. Then, the inverse can be separated in two parts :

$$A = \sum_{j=0}^{N} \lambda_j \left|u_j\right\rangle \left\langle u_j\right| ,$$

$$\implies A^{-1} = \sum_{\substack{j=0 \\ \lambda \neq \lambda_j}}^{N} \frac{1}{\lambda_j} \left|u_j\right\rangle \left\langle u_j\right| + \frac{1}{\lambda} \left|u_\lambda\right\rangle \left\langle u_\lambda\right| .$$

Thus, small variations such a truncation can lead to a solution diverging from the "true" solution by several orders of magnitude. A solution to this problem is to use filter function to isolate the well-conditioned and ill-conditioned part of the matrix A to avoid these problems [4]. The complexity is linear in function of the condition number : If the condition number grows like $\mathcal{O}\left(2^N\right)$ then no exponential speedup can be achieved.

Second, the preparation of state $|b\rangle$ is a non-trivial step. It is possible to prepare the vector b in $\lceil \log_2(N) \rceil$ qubits using amplitude encoding. In theory, this can be done using qRAM. However, the vector b must be relatively uniform [24] On the other hand, if there is an explicit formula which defines coefficients of b. The vector b can be also loaded in logarithmic time Table I . Moreover, qRAM loads data in logarithmic time in theory. In practice, it is possible that the time complexity is $\mathcal{O}(N^c)$ for some $c \in \mathbb{R}_+$. Such a behaviour can typically occur due to memory latencies in the qRAM and should also be taken into account [23]. Another problem of the qRAM is its robustness to errors. Indeed, the need of error correction component can lead to suppress any exponential speedup [25].

Thirdly, the Hamiltonian simulation must be done with a logarithmic complexity. It is known that it can be done for a sparse matrix A [15] and more recently, it was showed that the method works also if A is low-rank [26]. However it is necessary that all its entries are efficiently available for example using qRAM with all its consequences.

Fourth, even if the result state $|\tilde{x}\rangle$ is stored on $\log_2(N)$ qubits reading out the solution $\tilde{x}$ take order $\mathcal{O}(N)$ which cancels out any previous speedup. Hence, strictly speaking it is not possible to solve the QLSP (ie: having the solution $\tilde{x}$) with an speedup compared to the classical algorithm. However, it is possible to probe the solution in order to extract information about the solution such as value of inner products, means or the location of large values in the sample [21]. Moreover, if the error of the final state is ε, then : $\| \, |\tilde{x}\rangle - |x\rangle \, \| < \varepsilon$. The error of the final classical solution is $\varepsilon \|x\|$. Hence, the error on the state must be extremely small in order for it to be acceptable. This increases the number of qubits needed for the computation [27]

Problem	Algorithm	Runtime complexity	Year
LSP	CG [a]	$\mathcal{O}\left(N\kappa s \log\left(1/\varepsilon\right)\right)$ [2]	-
QLSP	HHL	$\mathcal{O}\left(\log(N) s^2 \kappa^2/\varepsilon\right)$ [4]	2009
QLSP	VTAA-HHL	$\mathcal{O}\left(\log(N) s^2 \kappa/\varepsilon\right)$ [28]	2010
QLSP	Childs et. al.	$\mathcal{O}\left(s\kappa \text{ polylog}(s\kappa/\varepsilon)\right)$ [29]	2017
QLSP	QLSA	$\mathcal{O}\left(\kappa^2 \text{ polylog } (N)\|A\|/\varepsilon\right)$ [21]	2018

[a] This method only works for positive definite matrix.

Table II. *Different algorithms, their runtime complexities and year of publication.*

The HHL algorithm was the first algorithm to solve the LSP in a quantum manner and open a new way of approaching these problems. The recent papers made improvements with the aim to reduce the sensitivity to error and to condition number Table II.

IV. CONCLUSION

The aim of these report was to present in detail the HHL algorithm. The different building blocks were explained. The strengths and weaknesses of the HHL were studied. In conclusion, the HHL does not achieve a exponential speedup if the aim is to get the entire solution of the linear system. However, if the goal is to probe some properties of the solution, one gets an exponential speedup. Moreover, the number of hypothesis reduces the number of possibilities to apply this algorithm. However, in many problems one only wants some properties of the solution, in this case HHL is a major improvement. Numerous possibilities of applications are already under the scoop such as solving the heat equation [30] or quantum machine learning [31]. One should not forget that parallel computing can also achieve major speedup compared to standard classical algorithms. Indeed, solving a linear system with parallel computation solves the LSP in $\mathcal{O}\left(\log^2(N)\right)$ [23].

In terms of feasibility, on one side, qRAM has not given (state of 2022) promising results and is still far away from a mature device. Lots of difficulties arise such as noise effects or error corrections and no satisfying answers are already found. On the other hand, it is possible to estimate the number of qubits needed to run the HHL algorithm one a quantum computer.The HHL has a similar structure as the Shor algorithm. One can use it as a lower bound to compute the qubits needed to run this algorithm on a quantum computer. To factor a 2048 bit RSA one needs at least 4000 logical qubits which is equivalent to millions of physical qubits [32]. Today one of the most advanced quantum computers possesses 53 physical qubits [33].

ACKNOWLEDGMENTS

The author thanks Javad Komijani for his helpful comments and relevant commentary on my work. The author is also grateful to Michaela Macakova for her comments and corrections on the manuscript.

APPENDIX

A. Numerical application

In the following section, some simulations of QPE and HHL algorithms have been done in order to illustrate some of the aspects of each algorithm. The simulation have been performed on the IBM Q device [34]. These experiences are inspired from [35].

1. *Quantum phase estimation simulation*

To perform the QPE, $m = 3$ qubits are used to estimate the value of the phase and $s = 3$ qubits for the eigenvector Fig. 1 . The goal is to estimate the eigenvalues of an unitary operator section II E. Let consider the T-gate, which is explicitly given by:

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

Thus, the eigenvalues $\lambda_0 = 1$ (resp. $\lambda_1 = e^{i\frac{\pi}{4}}$) are associated to eigenvectors $|0\rangle$ (resp. $|1\rangle$). Their associated phases according to (16) are $\theta_0 = 0$ and $\theta_1 = \frac{1}{8}$. Hence θ_1 can be represented exactly in binary as $2^3\theta$, because it is an integer.

First, we start the QPE with the initial state $|\psi_0\rangle = |0\rangle$. Then, we obtained a probability distribution where the peaks represents the final result. In binary, it gives: $001 \to 1$. Hence:

$$\tilde{\theta}_1 = 2^8\theta_1 = 1 \Longrightarrow \tilde{\theta}_1 = \frac{1}{8}.$$

Then, using (17), we obtain the approximation $\tilde{\lambda}_1$

$$\tilde{\lambda}_1 = e^{2\pi\tilde{\theta}} = e^{2\pi i\frac{1}{8}} = e^{i\frac{\pi}{4}}.$$

One can perform the same calculation with an initial $|\psi_0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. Then, the approximation of the two phases of associated in eigenvalues are $\tilde{\theta}_0$ and $\tilde{\theta}_1$. According to Fig. 4, one have in binary for θ_0: $000 \longrightarrow 0$ and for θ_1 : $001 \longrightarrow 1$. Then, one obtained:

$$\theta_0 = 2^3\tilde{\theta}_0 \Longrightarrow \tilde{\theta}_0 = 0,$$
$$\theta_1 = 2^3\tilde{\theta}_1 \Longrightarrow \tilde{\theta}_1 = \frac{1}{8}.$$

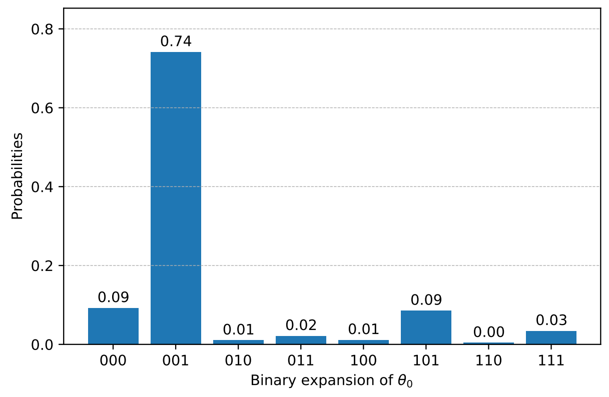

Fig. 3. Histogram of the QPE with initial state $|\psi_0\rangle = |0\rangle$. The experiment was performed with 2048 shots. The distribution peaks at $\theta_1 = 2^m \cdot \tilde{\theta}_1 = 1$ then the eigenvalue is $\lambda_1 = e^{i\frac{\pi}{4}}$.

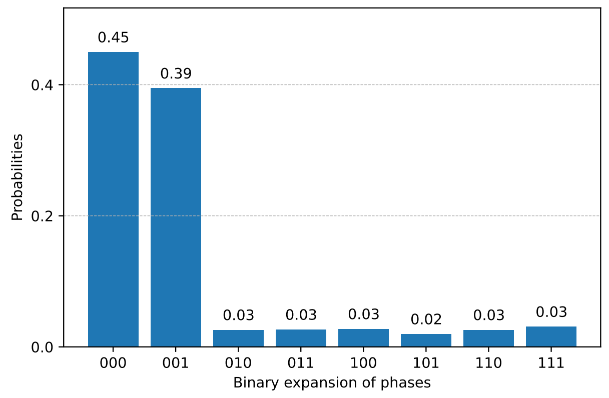

Fig. 4. Histogram of the QPE with initial state $|\psi_0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. The experiment was performed with 2048 shots. The distribution peaks at θ_0 and θ_1. Then, the eigenvalues are $\{1, e^{2\pi i\frac{1}{8}} = e^{i\frac{\pi}{4}}\}$

Then, eigenvalues are:

$$\tilde{\lambda}_0 = e^{2\pi i\cdot 0} = 1,$$

and

$$\tilde{\lambda}_1 = e^{2\pi i\frac{1}{8}} = e^{i\frac{\pi}{4}},$$

2. *Non-exact quantum phase estimation*

Let's study control rotations are studied $CR_{\frac{2\pi}{3}}$ of angle $\frac{2\pi}{3}$. This gate is explicitly given by:

$$CR_{\frac{2\pi}{3}} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i2\pi\frac{1}{3}} \end{pmatrix}.$$

Note that the eigenvalue associated to the eigenvector $|1\rangle$ is $\frac{1}{3}$ but on 3 qubits, it is not possible to obtain the exact value of the phase : $\frac{2^3}{3} = \frac{8}{3} \approx 2.6666$. This number is between $2 \to 010$ and is $3 \to 011$ but 011 is the nearest integer. It is expected that the most probable outcome of the computation is 011. By performing, the

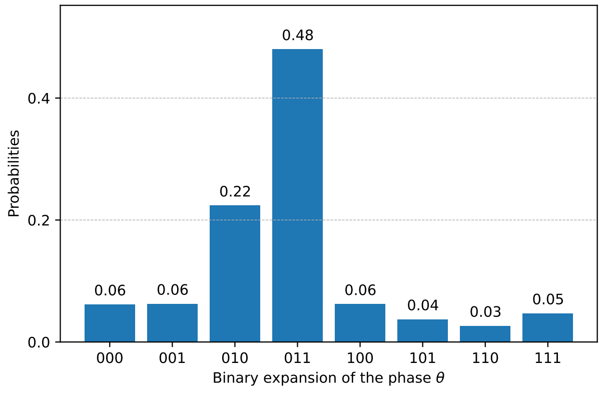

Fig. 5. Probability distribution of the quantum phase estimation of a control rotation $CR_{\frac{2\pi}{3}}$ on 3-qubits.

QPE estimation on a quantum computer, the most probable outcome is 011 as expected Fig. 5 and is bigger than $\frac{4}{\pi^2}$ as expected from section II E. Note also that the second most probable outcome is 010.

[1] "Storage-What data to record," (2022), retrieved from `www.home.cern/science/computing/storage`.

[2] J. R. Shewchuk, *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*, Tech. Rep. (USA, 1994).

[3] A. Hey and R. P. Feynman, *Feynman Lectures On Computation*, Frontiers in Physics (Westview Press, Philadelphia, PA, 2000).

[4] A. W. Harrow, A. Hassidim, and S. Lloyd, Phys. Rev. Lett. **103**, 150502 (2009).

[5] Complex matrix $A \in \mathbb{C}^{N\times N}$ can also be used without loss of generality.

[6] E. Kreyszig, *Advanced Engineering Mathematics 10E* (John Wiley & Sons, Chichester, England, 2010).

[7] J. A. Cortese and T. M. Braje, arXiv preprint arXiv:1803.01958 (2018).

[8] K. Mitarai, M. Kitagawa, and K. Fujii, Physical Review A **99** (2019), 10.1103/physreva.99.012301.

[9] J. Preskill, "Quantum computing 40 years later," (2021), arXiv:2106.10522 [quant-ph].

[10] M. Schuld, *Supervised learning with quantum computers* (Springer, Cham, 2018).

[11] V. Giovannetti, S. Lloyd, and L. Maccone, Physical Review Letters **100** (2008), 10.1103/physrevlett.100.160501.

[12] J. M. Yeomans, *Statistical mechanics of phase transitions* (Clarendon Press, Oxford, England, 1992).

[13] Z. Bian, F. Chudak, R. Israel, B. Lackey, W. G. Macready, and A. Roy, Frontiers in Physics **2** (2014), 10.3389/fphy.2014.00056.

[14] S. Lloyd, Science **273**, 1073 (1996), www.science.org/doi/pdf/10.1126/science.273.5278.1073.

[15] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders, Communications in Mathematical Physics **270**, 359–371 (2006).

[16] B. Hall, *Lie Groups, Lie Algebras, and Representations: An Elementary Introduction*, Graduate Texts in Mathematics (Springer International Publishing, 2015).

[17] M. Suzuki, Commun. Math. Phys. **51**, 183 (1976).

[18] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, England, 2010).

[19] Without loss of generality, the eigenvalue λ_j is chosen positively. If it is not the case, a transformation is applied : $H \to -H$.

[20] A. C. Vazquez, R. Hiptmair, and S. Woerner, ACM Transactions on Quantum Computing **3**, 1–37 (2022).

[21] L. Wossnig, Z. Zhao, and A. Prakash, Phys. Rev. Lett. **120**, 050502 (2018).

[22] A. Y. Kitaev, "Quantum measurements and the abelian stabilizer problem," (1995), arXiv:quant-ph/9511026 [quant-ph].

[23] S. Aaronson, Nature Physics **11**, 291 (2015).

[24] Otherwise, one can transform using amplitude amplification for example, the vector b such that $b_i = \delta_{ij}$ where j is the largest value of the vector. The associated quantum state would exhibit a delta function behaviour. This problem is equivalent to the Grover algorithm which has been proven that it can not be done with less than $\mathcal{O}\left(\sqrt{N}\right)$ [37]. Hence, one need a uniform vector to achieve a exponential speed up.

[25] S. Arunachalam, V. Gheorghiu, T. Jochym-O'Connor, M. Mosca, and P. V. Srinivasan, New Journal of Physics **17**, 123010 (2015).

[26] S. Lloyd, M. Mohseni, and P. Rebentrost, Nature Physics **10**, 631 (2014).

[27] A lower bound for the qubits can be obtained from the Shor algorithm which has the same structure as the HHL algorithm. This is given by $n = \log_2\left(\frac{1}{\varepsilon}\right)$ [18].

[28] A. Ambainis, (2010), arXiv:1010.4458 [quant-ph].

[29] A. M. Childs, R. Kothari, and R. D. Somma, SIAM Journal on Computing **46**, 1920 (2017), https://doi.org/10.1137/16M1087072.

[30] N. Linden, A. Montanaro, and C. Shao, (2020).

[31] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Nature **549**, 195–202 (2017).
[32] C. Gidney and M. Ekerå, Quantum **5**, 433 (2021).
[33] F. Arute, K. Arya, R. Babbush, D. Bacon, J. Bardin, R. Barends, R. Biswas, S. Boixo, F. Brandao, D. Buell, B. Burkett, Y. Chen, J. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. M. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. Harrigan, M. Hartmann, A. Ho, M. R. Hoffmann, T. Huang, T. Humble, S. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. Martinis, Nature **574**, 505–510 (2019).
[34] "Ibm quantum computer," (2022).
[35] "Qiskit: An open-source framework for quantum computing," (2021).
[36] B. Duan, J. Yuan, C.-H. Yu, J. Huang, and C.-Y. Hsieh, Physics Letters A **384**, 126595 (2020).
[37] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, SIAM Journal on Computing **26**, 1510–1523 (1997).

Algorithm	Query complexity	Time complexity	Gate complexity
Product formula	$\mathcal{O}\left(s^4t^2/\varepsilon\right)$	$\mathcal{O}\left(s^4t^2/\varepsilon\right)$	$\mathcal{O}\left(N^5\right)$
Product formula	$\mathcal{O}\left(5^{2k}s^3t(st/\varepsilon)^{1/2k}\right)$	$\mathcal{O}\left(5^{2k}s^3t(st/\varepsilon)^{1/2k}\right)$	$\mathcal{O}\left(5^{2k}n^{3+1/k}\right)$
Quantum walk	$\mathcal{O}\left(st/\sqrt{\varepsilon}\right)$	$\mathcal{O}\left(st/\sqrt{\varepsilon}\right)$	$\mathcal{O}\left(n^4\log n\right)$
Fractional-query simulation	$\mathcal{O}\left(s^2t\frac{\log(st/\varepsilon)}{\log\log(st/\varepsilon)}\right)$	$\mathcal{O}\left(s^2t\frac{\log^2(st/\varepsilon)}{\log\log(st/\varepsilon)}\right)$	$\mathcal{O}\left(n^4\frac{\log^2 n}{\log_{\log n}}\right)$
Taylor series	$\mathcal{O}\left(s^2t\frac{\log(st/\varepsilon)}{\log\log(st/\varepsilon)}\right)$	$\mathcal{O}\left(s^2t\frac{\log^2(st/\varepsilon)}{\log\log(st/\varepsilon)}\right)$	$\mathcal{O}\left(n^3\frac{\log^2 n}{\log_{\log n}}\right)$
Lin. combinaison of quantum walk steps	$\mathcal{O}\left(st\frac{\log(st/\varepsilon)}{\log\log(st/\varepsilon)}\right)$	$\mathcal{O}\left(st\frac{\log^{3.5}(st/\varepsilon)}{\log\log(st/\varepsilon)}\right)$	$\mathcal{O}\left(n^4\frac{\log^2 n}{\log\log n}\right)$
Quantum signal processing	$\mathcal{O}\left(st+\frac{\log(1/\varepsilon)}{\log\log(1/\varepsilon)}\right)$	$\mathcal{O}\left(st+\frac{\log(1/\varepsilon)}{\log\log(1/\varepsilon)}\right)$	$\mathcal{O}\left(N^3\right)$

Table III. List of different Hamiltonian simulation algorithms and their different complexities. Let t be the time of the simulation, ε the error on the solution, N denotes the size of the matrix and s is the sparsity of the matrix [36].