Variable time amplitude amplification and quantum algorithms for linear algebra problems

Andris Ambainis

Variable time amplitude amplification and quantum algorithms for linear algebra problems*

Andris Ambainis[1]

1 **Faculty of Computing, University of Latvia, Raina bulv. 19, Riga, LV-1586, Latvia,**
ambainis@lu.lv

Abstract

Quantum amplitude amplification is a method of increasing a success probability of an algorithm from a small $\epsilon > 0$ to $\Theta(1)$ with less repetitions than classically. In this paper, we generalize quantum amplitude amplification to the case when parts of the algorithm that is being amplified stop at different times.

We then apply the new variable time amplitude amplification to give two new quantum algorithms for linear algebra problems. Our first algorithm is an improvement of Harrow et al. algorithm for solving systems of linear equations. We improve the running time of the algorithm from $O(\kappa^2 \log N)$ to $O(\kappa \log^3 \kappa \log N)$ where κ is the condition number of the system of equations. Our second algorithm tests whether a matrix A is singular or far-from-singular, faster then the previously known algorithms.

1998 ACM Subject Classification F.1.2 Modes of computation, F2.1 Numerical algorithms and problems

Keywords and phrases quantum computing, quantum algorithms, amplitude amplification, linear equations

Digital Object Identifier 10.4230/LIPIcs.STACS.2012.636

1 Introduction

Large systems of linear equations arise in many situations and faster algorithms for solving them are of great interest. For this reason, it would be very interesting to have a quantum algorithms for this problem.

However, there are some substantial difficulties with designing such an algorithm. First, if we have a system of linear equations $Ax = b$ with N equations and N unknowns, the coefficient matrix A is of size N^2. If a quantum algorithm accesses all or most of coefficients in A, it would require time $\Omega(N^2)$. We could allow query access to the coefficient matrix A (similarly to Grover's algorithm [12] and other quantum query algorithms) but then we run into a second problem. The quantum algorithm still has to output the solution vector x. Since the solution vector x consists of values for N variables, this requires time $\Omega(N)$.

This argument suggests that quantum speedup for this problem can be at most polynomial (because classical algorithms for systems of linear equations run in time $O(N^\omega)$) where $\omega = 2.37...$ is the matrix multiplication constant.

* Supported by ESF project 1DP/1.1.1.2.0/09/APIA/VIAA/044, FP7 Marie Curie Grant PIRG02-GA-2007-224886 and FP7 FET-Open project QCS.

29th Symposium on Theoretical Aspects of Computer Science (STACS'12).
Editors: Christoph Dürr, Thomas Wilke; pp. 636–647

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Recently, Harrow, Hassidim and Lloyd [13] discovered a surprising quantum algorithm that allows to bypass the limitations described above and to "solve" systems of linear equations in time $O(\log^c N)$ - in an unconventional sense. Instead of outputting the solution vector x in a classical form, their algorithm generates the quantum state $|x\rangle = \sum_{i=1}^{N} x_i|i\rangle$ with the coefficients x_i being equal to the values of variables in the solution $x = (x_1, x_2, \ldots, x_N)$ of the system $Ax = b$.

HHL algorithm has been quite controversial. On one hand, one cannot read the values $x_1, \ldots, x_N$ from the quantum state $|x\rangle = \sum_{i=1}^{N} x_i|i\rangle$. Even to estimate them, one would have to produce many copies of $|x\rangle$, increasing the running time of the quantum algorithm many times.

On the other hand, the state $|x\rangle = \sum_{i=1}^{N} x_i|i\rangle$ can be used to estimate expressions of the form $\sum_i c_i x_i$ which depend on all x_i simultaneously. Classically, it is intuitively unlikely that one would be able to estimate expressions of this form without solving the system and finding $x_1, \ldots, x_N$ - which requires time $\Theta(N^c)$. This intuition is substantiated by the fact that estimating such expressions is BQP-complete [13]. Thus a classical algorithm for evaluating expressions of the form $\sum_i c_i x_i$ in time $O(\log^c N)$ does not exist - unless P=BQP.

Another context in which the state $|x\rangle$ would be useful is if we wanted to test whether the solutions of two systems of linear equations $Ax = b$ and $A'x' = b'$ were close one to another. In this case, we could generate the solution states $|x\rangle$ and $|x'\rangle$ for both systems and then compare them using the SWAP-test [8]. (For example, we might be interested in testing whether the stationary distributions of two Markov chains are close one to another [13]. Then, each stationary distribution can be described as a solution to a system of linear equations.)

Besides providing the output as a quantum state $|x\rangle$, another weakness of the HHL algorithm is the dependence of its running time on parameters other than the size of the system of equations N. In particular, its running time depends on κ, the condition number of matrix A. The condition number is defined as the ratio between the largest and the smallest singular value of A: $\kappa = \max_{i,j} \frac{|\mu_i|}{|\mu_j|}$ where μ_i are the singular values of A.

When condition number κ is taken into account, the running time of the HHL quantum algorithm is $O(\kappa^2 \log N)$. Thus, the speedup achieved by the HHL algorithm is exponential, as long as $\kappa = O(\log^c N)$. However, systems of linear equations with a polylogarithmic condition number are quite rare. It is much more common for a system to have a condition number that scales as $\Theta(N)$ or $\Theta(N^c)$. (We present some examples in section 4.5.) For this reason, we think that it is important to improve the dependence of the HHL algorithm on κ.

In this paper, we present a better quantum algorithm of solving systems of linear equations in the sense of HHL, with the running time $O(\kappa \log^3 \kappa \log N)$. It would be desirable to have even better dependence on κ but our algorithm is probably close to being optimal. Harrow et al. [13] show that, unless $BQP = PSPACE$, time of $\Omega(\kappa^{1-o(1)})$ is necessary for generating the state $|x\rangle$ that describes the solution of the system.

Our second result is a quantum algorithm for testing whether a matrix A is singular, under a promise that A is either singular or far from being singular. (Here, "far from singular" means that all singular values are at least ϵ.) Under this assumption, we design a quantum algorithm that runs in time[1] $\tilde{O}(\frac{s(A)}{\max(\sqrt{k},1)})$ where k is the number of singular

[1] Here, $\tilde{O}$ notation ignores logarithmic factors.

values of A that are equal to 0 and

$$s(A) = \sqrt{\sum_{i=1}^{N} \frac{1}{\min^2(\rho_i, \epsilon)}}$$

where ρ_i are all the singular values of A.

Both of our results use a new tool, the *variable-time quantum amplitude amplification* which allows to amplify the success probability of quantum algorithms in which some branches of the computation stop earlier than other branches. The conventional amplitude amplification [7] would wait for all branches to stop - possibly resulting in a substantial inefficiency. Our new algorithm amplifies the success probability in multiple stages and takes advantage of the parts of computation which stop earlier.

The variable-time amplitude amplification is a generalization of variable time search of Ambainis [2]. Variable time search was a generalization of Grover's algorithm to the setting when queries to different items take different time. In this paper, we improve variable time search to deal with the more general setting of amplitude amplification.

We then apply the new variable-time amplitude amplification to design the quantum algorithms for solving systems of linear equations and testing singularity. We expect that the variable time amplitude amplification will be useful for building other quantum algorithms, as well.

Related work After this work was completed, Belovs [4] discovered another algorithm for testing the singularity. Belovs' algorithm achieves similar running time, using a different method (span programs) than our work (variable time eigenvalue estimation).

2 Methods and subroutines

Throughout the paper, we use two well known quantum algorithms: *eigenvalue estimation* and *amplitude amplification*.

Eigenvalue estimation. Quantum eigenvalue estimation [15] is a quantum algorithm that, given a Hamiltonian H (in form of a black box that allows by apply H for a time T that we choose) and its eigenstate $|\psi\rangle : H|\psi\rangle = \lambda|\psi\rangle$, outputs an unchanged eigenstate $|\psi\rangle$, together with an estimate $\tilde{\lambda}$ for the eigenvalue λ.

We assume that $0 \leq \lambda \leq 1$. The standard version of eigenvalue estimation [14, p. 118] performs the unitary $U = e^{-iH}$ up to 2^k times and outputs $x \in \{0, \frac{\pi}{2^k}, \frac{2\pi}{2^k}, \ldots, \frac{(2^k-1)\pi}{2^k}\}$ with probability

$$p(x) = \frac{1}{2^{2k}} \frac{\sin^2 2^k(\lambda - x)}{\sin^2(\lambda - x)} \tag{1}$$

(equation (7.1.30) from [14]).

According to Theorem 7.1.5 in [14], if $\lambda \in [\frac{m}{2^k}, \frac{m+1}{2^k}]$, then the probability of outputting one of the two closest estimates ($\frac{m}{2^k}$ and $\frac{m+1}{2^k}$)) is at least $\frac{8}{\pi^2}$. We can increase this probability to at least $1-\epsilon$ by repeating the eigenvalue estimation algorithm $O(\log \frac{1}{\epsilon})$ times and taking the majority of answers.

Amplitude amplification. Another tool that we repeatedly use is quantum amplitude amplification [7]. Quantum amplitude amplification takes an algorithm $\mathcal{A}$ that succeeds with a small probability ϵ and transforms it into an algorithm $\mathcal{A}'$ that succeeds a probability $2/3$ (or $1 - o(1)$).

Classically, increasing the success probability from ϵ to $2/3$ requires repeating $\mathcal{A}$ $\Theta(\frac{1}{\epsilon})$ times. Quantumly, amplitude amplification allows to do that with just $O(\frac{1}{\sqrt{\epsilon}})$ repetitions of

$\mathcal{A}$. The algorithm $\mathcal{A}$ whose success probability is being increased can be either a classical algorithm or a quantum algorithm.

3 Variable time amplitude amplification

Our first result is a generalization of the amplitude amplification. Consider a quantum algorithm $\mathcal{A}$ which may stop at one of several times $t_1, \ldots, t_m$. (In the case of singularity-testing or systems of linear equations, these times correspond to m runs of eigenvalue estimation with increasing precision and increasing number of steps.) To indicate the outcome, $\mathcal{A}$ has an extra register O with 3 possible values: 0, 1 and 2. 1 indicates the outcome that should be amplified. 0 indicates that the computation has stopped at this branch but did not result in the desired outcome 1. 2 indicates that the computation at this branch has not stopped yet.

Let p_i be the probability of the algorithm stopping at time t_i (with either the outcome 0 or outcome 1). The average stopping time of $\mathcal{A}$ (the l_2 average) is

$$T_{av} = \sqrt{\sum_i p_i t_i^2}.$$

T_{max} denotes the maximum possible running time of the algorithm (which is equal to t_m). Let

$$\alpha_{good}|1\rangle_O|\psi_{good}\rangle + \alpha_{bad}|0\rangle_O|\psi_{bad}\rangle$$

be the algorithm's output state after all branches of the computation have stopped. Our goal is to obtain $|\psi_{good}\rangle$ with a high probability. Let $p_{succ} = |\alpha_{good}|^2$ be the probability of obtaining this state via algorithm $\mathcal{A}$.

Our main result is

▶ Theorem 1. We can construct a quantum algorithm $\mathcal{A}'$ invoking $\mathcal{A}$ several times, for total time

$$O\left(T_{max}\sqrt{\log T_{max}} + \frac{T_{av}}{\sqrt{p_{succ}}}\log^{1.5} T_{max}\right)$$

that produces a state $\alpha|1\rangle \otimes |\psi_{good}\rangle + \beta|0\rangle \otimes |\psi'\rangle$ with probability $|\alpha|^2 \geq 1/2$ as the output[2].

Proof. The proof is given in the full version of the paper [3]. ◀

By repeating $\mathcal{A}'$ $O(\log \frac{1}{\epsilon})$ times, we can obtain $|\psi_{good}\rangle$ with a probability at least $1-\epsilon$.

In contrast to our algorithm, the usual amplitude amplification [7] would run for time $O(\frac{T_{max}}{\sqrt{p_{succ}}})$. Our algorithm $\mathcal{A}'$ provides an improvement whenever T_{av} is substantially smaller than T_{max}.

Our algorithm $\mathcal{A}'$ is optimal, up to the factor of $\log^c T_{max}$. If the algorithm $\mathcal{A}$ has just one stopping time $T = T_{av} = T_{max}$, then amplitude amplification cannot be performed with fewer than $O(\frac{T}{\sqrt{p_{succ}}})$ steps. Thus, the term of $\frac{T_{av}}{\sqrt{p_{succ}}}$ is necessary.

If we would like to algorithm $\mathcal{A}'$ to be exact (to produce an output state that is exactly $|\psi_{good}\rangle$, conditional on the first bit being $|1\rangle$), the term T_{max} is also necessary because, in some branch of computation, $\mathcal{A}$ can run for T_{max} steps and $\mathcal{A}'$ needs the part of $|\psi_{good}\rangle$ that comes from this branch. If $\mathcal{A}'$ only has to produce an approximation of $|\psi_{good}\rangle$, a better result is possible.

[2] The first bit of the output state indicates whether we have the desired state $|\psi_{good}\rangle$ or not. Since $|\alpha|^2 \geq 1/2$, we get $|\psi_{good}\rangle$ with probability at least 1/2.

▶ Theorem 2. Let $\epsilon > 0$ be a constant. We can construct a quantum algorithm $\mathcal{A}'$ invoking $\mathcal{A}$ several times, for total time

$$O\left(\frac{T_{av}}{\sqrt{p_{succ}}}\log^{1.5}\max\left(T_{av},\frac{1}{p_{succ}}\right)\right)$$

that produces a state $\alpha|1\rangle\otimes|\psi'_{good}\rangle+\beta|0\rangle\otimes|\psi'\rangle$ with $|\alpha|^2\geq 1/2$ and $\|\psi_{good}-\psi'_{good}\|\leq\epsilon$ as the output.

Theorem 2 is a straightforward consequence of Theorem 1. We observe that the probability of an algorithm $\mathcal{A}$ running for more than $T_0=\frac{T_{av}}{\sqrt{\delta p_{succ}}}$ steps is at most δp_{succ}. (Otherwise, we would have $T_{av}^2>\delta p_{succ}T_0^2=T_{av}$.)

We set $\delta=(\epsilon/2)^2$ and $T_0=\frac{T_{av}}{\epsilon\sqrt{p_{succ}}/2}$ and take a quantum algorithm $\mathcal{A}_1$ that runs $\mathcal{A}$ but stops after T_0 steps. Then, the output state of $\mathcal{A}_1$ is $|\psi'_{good}\rangle$ with[3] $\|\psi_{good}-\psi'_{good}\|\leq\epsilon$ and T_{max} for the new algorithm $\mathcal{A}_1$ is equal to T_0. Theorem 2 now follows by applying Theorem 1 to $\mathcal{A}_1$.

4 Quantum algorithms for linear algebra problems

4.1 Preliminaries

We will consider two problems: testing whether a matrix A is singular and "solving" systems of linear equations $Ax=b$ in the sense of [13].

Similarly to [13], we assume that the matrix A is Hermitian. This assumption is without a loss of generality. For singularity testing, if A is not Hermitian, we can replace it by

$$A'=\begin{pmatrix}0 & A\\ A^\dagger & 0\end{pmatrix}, \tag{2}$$

where 0 denotes the all-zero matrix of the appropriate size. Then, A' is singular if and only if A is singular.

For systems of linear equations, we can replace $Ax=b$ by $A'y=b'$ where $b'=\begin{pmatrix}0\\ b\end{pmatrix}$. The solution of this system is

$$y=\begin{pmatrix}x\\ 0\end{pmatrix}$$

which is essentially equivalent to x.

For both algorithms, the matrix A can be given in one of the following forms:

1. A black box implementing A (for Hermitian A) as a Hamiltonian;
2. A black box answering queries about the values of A, in one of the following two forms:
   a. (for dense matrices) given i,j, the black box returns a_{ij};
   b. (for sparse matrices) given i, the black box returns a list of all values in the i^{th} row (or i^{th} column) that are non-zero.

[3] Removing the part of $|\psi_{good}\rangle$ that corresponds to $\mathcal{A}$ running for more than T_0 steps results in an unnormalized state $|\psi''_{good}\rangle$ with $\|\psi_{good}-\psi''_{good}\|\leq\epsilon/2$. Normalizing $|\psi''_{good}\rangle$ results in a normalized state $|\psi'_{good}\rangle$ with $\|\psi''_{good}-\psi'_{good}\|\leq\epsilon/2$ and $\|\psi_{good}-\psi'_{good}\|\leq\epsilon$.

The second case reduces to the first one, because, given a black box that answers queries about values a_{ij}, we can build a black box implementing A, by using one of methods for simulating black-box Hamiltonians. In the sparse case, to simulate the Hamiltonian A for time T, it is sufficient to use the query black box for A $O((T\log N)^{1+o(1)})$ times [5, 10]. In the dense case, the quantum-walk based methods by Childs [9] give an $O(C(A)T)$ query simulation of the Hamiltonian A for time T, with a somewhat complicated dependence of $C(A)$ on the matrix A.

For the rest of this paper, we assume that A is given via a Hamiltonian. We assume that the evolution of the Hamiltonian A for time T can be simulated in time $C(A)\min(T,1)$, for some $C(A)$ (as in the simulation by [9]). (Using a simulation method that works in time $O(C(A)T^{1+o(1)})$ (as in [5, 10]) is also possible, with a corresponding increase in the running times of our algorithms.)

4.2 Singularity testing

We consider the problem of testing whether a matrix A is singular. It is known that testing the singularity of an $n\times n$ matrix requires $\Omega(n^2)$ queries in the quantum query model [11].

However, better quantum algorithms may be possible for restricted cases of the singularity problem. A natural restriction is to consider the case when the matrix A is either singular or far from being singular.

Namely, we consider the testing whether A is singular with a promise that $\|A\|\leq 1$ and one of the following two is true:

- A is singular;
- All singular values of A are at least ϵ.

We will refer to this problem as ϵ-*Singularity*.

Let $\rho_1(A),\ldots,\rho_N(A)$ be the singular values of a matrix A. Let

$$s(A)=\sqrt{\sum_{i=1}^{N}\frac{1}{\min^2(\rho_i,\epsilon)}}.$$

If A is not Hermitian, we replace it by a Hermitian, as described in section 4.1. If $\rho_1(A),\ldots,\rho_N(A)$ are the singular values of a matrix A, then the eigenvalues of A' are $\pm\rho_1(A),\ldots,\pm\rho_N(A)$.

▶ Theorem 3. There is an algorithm $\mathcal{A}$ for ϵ-singularity that runs in time $O(C(A)s(A)\log^{1.5}s(A)\log N)$ if A is non-singular and time

$$O\left(\frac{C(A)s(A)\log^{1.5}s(A)\log N}{\sqrt{k}}\right)$$

if A has $k>0$ singular values that are equal to 0.

Proof. We apply variable time amplitude amplification to Algorithm 1.

To analyze this algorithm, we first observe that receiving the second part of a completely mixed state as an input is equivalent to receiving the N-dimensional completely mixed state ρ_N as the input. The completely mixed state can be written as a mixture of eigenvectors $|v_i\rangle$ of A with equal coefficients:

$$\rho_N=\sum_{i=1}^{N}\frac{1}{N}|v_i\rangle\langle v_i|.$$

Input: an $N \times N$ matrix A.

1. With probability $\frac{1}{2N}$ output "non-singular" and stop.
2. Prepare a bipartite state

$$\sum_{i=1}^{N} \frac{1}{\sqrt{N}} |i\rangle \otimes |i\rangle.$$

3. Let $k = 1$.
4. While $k \leq \lceil \frac{2}{\epsilon} \rceil$, do:
   a. On the second register, apply eigenvalue estimation for A with parameters chosen so that, with probability at least $1 - \frac{1}{N^2}$, the estimate is within $\frac{1}{2^k}$ of being correct.
   b. If the obtained estimate is at least $\epsilon + \frac{1}{2^k}$, output "fail" and stop.
   c. If the obtained estimate is at most $\epsilon - \frac{1}{2^k}$, output "singular" and stop.

Algorithm 1 Algorithm for singularity testing.

If the input to eigenvalue estimation was $|v_i\rangle$, the eigenvalue estimation loop would stop after $O(\frac{1}{\rho_i} \log N)$ steps, with a high probability. Therefore, the l_2-average stopping time would be $T_{av} = O(\frac{s(A) \log N}{\sqrt{N}})$.

Let $\mathcal{A}$ be the algorithm obtained by applying Theorem 2 to Algorithm 1. If A has k singular values that are equal to 0, then the success probability of Algorithm 1 is $p_{succ} = \frac{k+0.5}{N}$ and the running time of $\mathcal{A}$ is

$$O\left(\frac{T_{av}}{\sqrt{p_{succ}}} \log^{1.5} \max(T_{av}, p_{succ})\right) = O\left(\frac{s(A) \log^{1.5} s(A) \log N}{\sqrt{k}}\right).$$

Conditional on the algorithm succeeding, the probability of the correct answer “singular" is $\frac{k}{k+0.5} \geq \frac{2}{3}$.

If A has no singular value equal to 0, then the success probability of Algorithm 1 is $p_{succ} = \frac{1}{2N} + O(\frac{1}{N^2})$, with the $O(\frac{1}{N^2})$ term coming from the possibility that eigenvalue estimation may output an incorrect estimate with probability (at most $\frac{1}{N^2}$). The running time of $\mathcal{A}$ is

$$O\left(\frac{T_{av}}{\sqrt{p_{succ}}} \log^{1.5} \max(T_{av}, p_{succ})\right) = O\left(s(A) \log^{1.5} s(A) \log N\right).$$

Conditional on the algorithm succeeding, the probability of the correct answer “non-singular" is $\frac{\frac{1}{2N}}{p_{succ}} = 1 - o(1)$. ◀

4.3 Systems of linear equations

We consider solving a system of linear equations $Ax = b$ where $A = (a_{ij})_{i,j \in [N]}$, $x = (x_i)_{i \in [N]}$, $b = (b_i)_{i \in [N]}$. As before, we assume that A is Hermitian.

Let $|v_i\rangle$ be the eigenvectors of A and λ_i be their eigenvalues. Similarly to [13], we assume that all λ_i satisfy $\frac{1}{\kappa} \leq |\lambda_i| \leq 1$ for some known κ. We can then transform the state $|b\rangle = \sum_{i=1}^{n} b_i |i\rangle$ into $|x\rangle = \sum_{i=1}^{n} x_i |i\rangle$ as follows:

1. If, in terms of eigenvectors $|v_i\rangle$ of A, we have $|b\rangle = \sum_i c_i |v_i\rangle$, then $|x\rangle = \sum_i \frac{c_i}{\lambda_i} |v_i\rangle$.
2. By eigenvalue estimation, we can create the state $|b'\rangle = \sum_i c_i |v_i\rangle |\tilde{\lambda}_i\rangle$ where $\tilde{\lambda}_i$ are the estimates of the true eigenvalues.

3. We then create the state

$$|b''\rangle = \sum_i c_i |v_i\rangle |\tilde{\lambda}_i\rangle \left(\frac{1}{\kappa\tilde{\lambda}_i}|1\rangle + \sqrt{1 - \frac{1}{\kappa^2\tilde{\lambda}^2}}|0\rangle \right). \quad (3)$$

   Conditional on the last bit being 1, the rest of state is $\sum_i \frac{c_i}{\tilde{\lambda}_i}|v_i\rangle|\tilde{\lambda}_i\rangle$ which can be turned into an approximation of $|x\rangle$ by running eigenvalue estimation in reverse and uncomputing $\tilde{\lambda}_i$.
4. We then amplify the part of state which has the last qubit equal to 1 (using amplitude amplification) and obtain a good approximation of $|x\rangle$ with a high probability.

▶ Theorem 4. [13] Let $Ax = b$ be a system of linear equations. Then, we can generate $|\psi\rangle$ satisfying $\||\psi\rangle - |x\rangle\| \leq \epsilon$ where $|x\rangle = \sum_{i=1}^n x_i|i\rangle$ in time $O(\frac{C(A)\kappa^2}{\epsilon}\log N)$.

The main term in the running time, κ^2 is generated as a product of two κ's. First, for $\||\psi\rangle - |x\rangle\| \leq \epsilon$, it suffices that the estimates $\tilde{\lambda}_i$ satisfy $|\lambda_i - \tilde{\lambda}_i| = O(\epsilon\tilde{\lambda}_i)$. Since $\lambda_i = \Omega(1/\kappa)$, this means $|\lambda_i - \tilde{\lambda}_i| = O(\frac{\epsilon}{\kappa})$. To estimate λ_i within error $O(\frac{\epsilon}{\kappa})$, we need to run H for time $O(\frac{\kappa}{\epsilon})$. Second, for amplitude amplification, we may need to repeat the algorithm generating $|b''\rangle$ $O(\kappa)$ times - resulting in the total running time $O(\kappa^2/\epsilon)$.

For eigenvalue estimation, the worst case is when all of most of λ_i are small (of order $\Theta(1/\kappa)$). Then, $|\lambda_i - \tilde{\lambda}_i| = \Theta(\frac{\epsilon}{\kappa})$ and eigenvalue estimation with the right precision indeed requires time $\Theta(\frac{\kappa}{\epsilon})$.

For amplitude amplification, the worst case is if most or all of λ_i are large (constant). Then, the coefficients $\frac{1}{\kappa\tilde{\lambda}_i}$ can be of order $\Theta(1/\kappa)$ and $\Theta(\kappa)$ repetitions are required for amplitude amplification.

We now observe that the two $\Theta(\kappa)$'s appear in the opposite cases. One of them appears when λ_i is small ($\lambda_i \approx \kappa$) but the other appears when λ_i is large ($\lambda_i \approx 1$).

If all eigenvalues are of roughly similar magnitude (e.g., $\lambda \in [a, 2a]$ for some a), the running time becomes $O(\kappa/\epsilon)$ because we can do eigenvalue estimation in time to error ϵa in $O(1/a\epsilon)$ and, for amplitude amplification, it suffices to repeat the generation of $|b''\rangle$ $O(\kappa a)$ times (since the amplitude of 1 in the last qubit of $|b'\rangle$ is at least $\frac{1}{\kappa a}$ for every v_i). Thus, the running time is

$$O\left(\frac{1}{a\epsilon}\right) \cdot O(\kappa a) = O\left(\frac{\kappa}{\epsilon}\right).$$

The problem is to achieve a similar running time in the general case (when the eigenvalues λ_i can range from κ to 1).

To do that, we run eigenvalue estimation several times. Each time, we double the precision and double the running time (as in Algorithm 1 for singularity testing). This gives a quantum algorithm in which different branches of computation stop at different times. By applying our variable-time amplitude amplification to this quantum algorithm, we get

▶ Theorem 5. Let $Ax = b$ be a system of linear equations. Then, we can generate $|\psi\rangle$ satisfying $\||\psi\rangle - |x\rangle\| \leq \epsilon$ in time

$$O\left(\frac{C(A)\kappa\log^3\frac{\kappa}{\epsilon}}{\epsilon^3}\log^2\frac{1}{\epsilon}\right).$$

For more details, we refer the reader to the full version of the paper [3].

4.4 Algorithm of Theorem 5

In this subsection, we describe the algorithm of Theorem 5. For its analysis, we refer the reader to the full version of this paper [3].

For our algorithm, we need a version of eigenvalue estimation that is guaranteed to output exactly the same estimate with a high probability. This can be achieved by running the standard eigenvalue estimation (described in section 2) k_{uniq} times and takes the most frequent answer x_{maj}.

▶ Lemma 1. For $k_{uniq} = O(\frac{1}{\epsilon^2} \log \frac{1}{\epsilon})$, we have

1. If $|\lambda - x| \leq \frac{1-\epsilon}{2^{n+1}}$, then $Pr[x_{maj} = x] \geq 1 - \epsilon$.
2. If $\lambda \in [x + \frac{1-\epsilon}{2^{n+1}}, x + \frac{1+\epsilon}{2^{n+1}}]$, then $Pr[x_{maj} \in \{x, x+1\}] \geq 1 - \epsilon$.

Proof. Omitted. ◀

We refer to this algorithm as **UniqueEst**$(H, 2^n, \epsilon)$.

When we use **UniqueEst** as a subroutine in algorithm 3, we need the answer to be unique (as in the first case) and not one of two high-probability answers (as in the second case). To deal with that, we will replace H with $H + \frac{\delta\pi}{2^n} I$ for a randomly chosen $\delta \in [0, 1]$. The eigenvalue becomes $\lambda' = \lambda + \frac{\delta\pi}{2^n}$ and, with probability $1 - \epsilon$,

$$\lambda' \in \left[\frac{x - \frac{1-\epsilon}{2}}{2^n}\pi, \frac{x + \frac{1-\epsilon}{2}}{2^n}\pi \right]$$

for some integer x. This allows to achieve the first case for all eigenvalues, except a small random fraction of them.

We now show that Theorem 1 implies our main result, Theorem 5. We start by describing a variable running time Algorithm 2. This algorithm uses the following registers:

- The input register I which holds the input state $|x\rangle$ (and is also used for the output state);
- The outcome register O, with basis states $|0\rangle, |1\rangle$ and $|2\rangle$ (as described in the setup for variable-time amplitude amplification);
- The step register S, with basis states $|1\rangle$, $|2\rangle$, ..., $|2m\rangle$ (to prevent interference between various branches of computation).
- The estimation register E, which is used for eigenvalue estimation (which is a subroutine for our algorithm).

$\mathcal{H}_I$, $\mathcal{H}_O$, $\mathcal{H}_S$ and $\mathcal{H}_E$ denote the Hilbert spaces of the respective registers.

From now on, we refer to ϵ appearing in Theorem 5 as ϵ_{final}. ϵ without a subscript is an error parameter for subroutines of algorithm 2 (which we will choose at the end of the proof so that the overall error in the output state is at most ϵ_{final}).

Our main algorithm is Algorithm 3 which consists of applying variable-time amplitude amplification to Algorithm 2.

We claim that, conditional on the output register being $|1\rangle_O$, the output state of Algorithm 2 is close to

$$|\psi_{ideal}\rangle = \sum_i \alpha_i |v_i\rangle_I \otimes \left(\frac{1}{\kappa\lambda_i} |1\rangle_O \otimes |2j_i\rangle_S \right). \tag{5}$$

Variable-time amplitude amplification then generates a state that is close to $\frac{|\psi_{ideal}\rangle}{\||\psi_{ideal}\rangle\|}$. Fourier transform in the last step of algorithm 3 then effectively erases the S register. Conditional

Input: parameters $x_1, \ldots, x_m \in [0,1]$, Hamiltonian H.

1. Initialize O to $|2\rangle$, S to $|1\rangle$ and E to $|0\rangle$. Set $j = 1$.
2. Let $m = \lceil \log_2 \frac{\kappa}{\epsilon} \rceil$.
3. Repeat until $j > m$:

   Stage j:

   a. Let $H' = H + \frac{x_j \pi}{2^j} I$. Using the registers I and S, run **UniqueEst**$(H', 2^j, \epsilon)$. Let λ' be the estimate output by **UniqueEst** and let $\lambda = \lambda' - \frac{x_j \pi}{2^j}$.

   b. If $\epsilon\lambda > \frac{1}{2^{j+1}}$, perform the transformation

$$|2\rangle_O \otimes |1\rangle_S \rightarrow \frac{1}{\kappa\lambda}|1\rangle_O \otimes |2j\rangle_S + \sqrt{1 - \frac{1}{(\kappa\lambda)^2}}|0\rangle_O \otimes |2j\rangle_S. \quad (4)$$

   c. Run **UniqueEst** in reverse, to erase the intermediate information.

   d. Check if the register E is in the correct initial state $|0\rangle_E$. If not, apply $|2\rangle_O \otimes |1\rangle_S \rightarrow |0\rangle_O \otimes |2j+1\rangle_S$ on the outcome register O.

   e. If the outcome register O is in the state $|2\rangle$, increase j by 1 and go to step 2.

Algorithm 2 State generation algorithm

Input: Hamiltonian H.

1. Generate uniformly random $x_1, \ldots, x_m \in [0,1]$.
2. Apply variable-time amplitude amplification to Algorithm 2, with H and $x_1, \ldots, x_m$ as the input.
3. Apply a transformation mapping $|2j\rangle_S \rightarrow |j\rangle_S$ to the S register. After that, apply Fourier transform F_m to the S register and measure. If the result is 0, output the state in the I register. Otherwise, stop without outputting a quantum state.

Algorithm 3 Main algorithm

on S being in $|0\rangle_S$ after the Fourier transform, the algorithm's output state is close to our desired output state $\frac{|x\rangle}{\||x\|\|}$, where

$$|x\rangle = \sum_i \alpha_i |v_i\rangle_I.$$

Finally, performing Fourier transform and measuring produces $|0\rangle_S$ with probability $1/m$. Because of that, the success probability of algorithm 3 needs to be amplified. This adds a factor of $O(\sqrt{m})$ to the running time, if we would like to obtain the result state with probability $\Omega(1)$ and a factor of $O(\sqrt{m}\log\frac{1}{\epsilon})$ if we would like to obtain it with probability at least $1-\epsilon$.

4.5 Examples of systems of linear equations

The Harrow-Hassidim-Lloyd algorithm achieves the biggest speedup when the condition number κ is small. If κ is polylogarithmic in N, then $O(\kappa^2 \log^c N) = O(\log^{c'} N)$. The Harrow-Hassidim-Lloyd algorithm then achieves an exponential speedup compared to the classical algorithms which run in time that is polynomial in N.

In this case, the additional advantage provided by our algorithm is small. However, systems of linear equations for which $\kappa = O(\log^c N)$ are quite rare. (We have looked at possible applications of the HHL algorithms and it was difficult to find natural examples of

systems where $\kappa = O(\log^c N)$.) We illustrate this with several natural examples of systems of linear equations.

Example 1: Assume that we have a system of equations $Ax = b$ in which A and b are random (for example, each entry is an i.i.d random variable which takes values +1 and -1 with probability 1/2 each).

With a high probability, the biggest singular value of A is of the order $\Theta(\sqrt{N})$ and the smallest singular value of A is of the order $\Theta(1/\sqrt{N})$ [16]. Hence, $\kappa = \Theta(N)$.

Thus, the running time of the HHL algorithm would be

$$O(\kappa^2 C(A) \log^c N) = O(N^2 C(A) \log^c N).$$

(For arbitrary A, with query access to A, $C(A) = O(N)$ [9]. This would give the overall running time of $O(N^3 \log^c N)$ - worse than classical algorithms for solving systems of linear equations.)

Theorem 5 provides an improvement of the running time to

$$O(\kappa \log^3 \kappa C(A) \log^c N) = O(NC(A) \log^c N).$$

Example 2: Consider a d-dimensional grid of size $\sqrt[d]{N} \times \sqrt[d]{N} \times \ldots \times \sqrt[d]{N}$, consisting of locations $(a_1, \ldots, a_d)$, $a_i \in \{1, 2, \ldots, \sqrt[d]{N}\}$. Let L be the Laplacian of this grid, defined by

$$L_{(a_1,\ldots,a_d),(b_1,\ldots,b_d)} = \begin{cases} 2d & \text{if } (a_1, \ldots, a_d) = (b_1, \ldots, b_d) \\ -1 & \text{if } a_i = b_i \pm 1 \text{ for one } i \text{ and } a_i = b_i \text{ for all other } i \\ 0 & \text{otherwise} \end{cases}$$

Consider a system of linear equations of the form $Lx = b$.

We can express $L = L_1 + L_2 + \ldots + L_d$ where

$$(L_i)_{(a_1,\ldots,a_d),(b_1,\ldots,b_d)} = \begin{cases} 2 & \text{if } (a_1, \ldots, a_d) = (b_1, \ldots, b_d) \\ -1 & \text{if } a_i = b_i \pm 1 \text{ and } a_j = b_j \text{ for all } j \neq i \\ 0 & \text{otherwise} \end{cases}$$

For simplicity, we assume that the grid has periodic boundary conditions (i.e., location $\sqrt[d]{N} + 1$ equals location 1). Then, the eigenvalues of L_i are $\lambda_j = 2 - 2\cos\frac{j\pi}{\sqrt[d]{N}}$, for $j = 0, 1, \ldots, \sqrt[d]{N} - 1$. Since $\cos x \approx 1 - \frac{x^2}{2}$ for small x, the smallest non-zero eigenvalue is

$$2 - 2\cos\frac{\pi}{\sqrt[d]{N}} \approx 2\left(\frac{\pi}{\sqrt[d]{N}}\right)^2 = \Theta\left(\frac{1}{N^{2/d}}\right).$$

The largest eigenvalue is upper bounded by 4.

The eigenvalues of L are of the form $\lambda_{j_1} + \lambda_{j_2} + \ldots + \lambda_{j_d}$ where λ_{j_i} are the eigenvalues of L_i. Hence, the smallest non-zero eigenvalue is $\Theta(\frac{1}{N^{2/d}})$ while the largest eigenvalue is $\Theta(d)$. The condition number is $O(dN^{2/d})$.

For $d = \log N$, the condition number would be of the order $O(\log N)$ and both HHL and our algorithm would run in polylogarithmic time. However, a more interesting case would be $d = 2$ or $d = 3$, since this would correspond to a discretization of actual physical processes in 2 or 3 dimensions. Then, $\kappa = O(N)$ (for $d = 2$) or $\kappa = O(N^{2/3})$ (for $d = 3$).

In this case, the HHL algorithm would run in time $\tilde{O}(N^2)$ or $\tilde{O}(N^{4/3})$. Our algorithm would improve this to $\tilde{O}(N)$ or $\tilde{O}(N^{2/3})$. (Since the Laplacian L is sparse, the overhead due to simulating Hamiltonian L is small.)

References

1 S. Aaronson, A. Ambainis, Quantum search of spatial regions. *Theory of Computing*, 1:47-79, 2005. Also quant-ph/0303041.

2 A. Ambainis. Quantum search with variable times. *Theory of Computing Systems*, 47(3): 786-807, 2010. Earlier version in STACS'08 and quant-ph/0609188.

3 A. Ambainis. Variable time amplitude amplification and quantum algorithms for linear algebra problems. arXiv:1010.4458.

4 A. Belovs. Span-program-based quantum algorithm for the rank problem. arXiv:1103.0842.

5 D.W. Berry, G. Ahokas, R. Cleve, and B.C. Sanders. Efficient Quantum Algorithms for Simulating Sparse Hamiltonians. *Communication in Mathematical Physics*, 270(2):359-371, 2007. Also arXiv:quant-ph/0508139

6 D. Berry. Quantum algorithms for solving linear differential equations. arXiv:1010.2745.

7 G. Brassard, P. Høyer, M. Mosca, A. Tapp. Quantum amplitude amplification and estimation. In *Quantum Computation and Quantum Information Science*, AMS Contemporary Mathematics Series, 305:53-74, 2002. Also quant-ph/0005055.

8 H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16):167902, 2001. Also quant-ph/0102001

9 A. M. Childs. On the relationship between continuous- and discrete-time quantum walk, *Communications in Mathematical Physics*, 294:581-603, 2010. Also arXiv:0810.0312.

10 A. M. Childs, R. Kothari. Simulating sparse Hamiltonians with star decompositions. *Proceedings of TQC 2010*, Lecture Notes in Computer Science 6519:94-103, 2011. Also arXiv:1003.3683.

11 S. Dörn, T. Thierauf. The quantum query complexity of the determinant. *Information Processing Letters*, 109 (6):325-328, 2009.

12 Lov K. Grover. A fast quantum mechanical algorithm for database search. *Proceedings of STOC'96*, pp. 212-219. Also quant-ph/9605043.

13 A. Harrow, A. Hassidim, S. Lloyd, Quantum algorithm for linear systems of equations. *Physical Review Letters*, 15(103):150502, 2009. Also arxiv:0811.3171.

14 P. Kaye, R. Laflamme, M. Mosca. *An Introduction to Quantum Computing.* Cambridge University Press, 2007.

15 M. Mosca, A. Ekert. The Hidden Subgroup Problem and Eigenvalue Estimation on a Quantum Computer. *Proceedings of QCQC'98*, Lecture Notes in Computer Science, 1509: 174-188, 1998. Also quant-ph/9903071.

16 M. Rudelson, R. Vershynin. The Littlewood - Offord problem and invertibility of random matrices. *Advances in Mathematics*, 218:600-633, 2008.

17 J. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical Report CMU-CS-94-125, School of Computer Science, Carnegie Mellon University, 1994.