Perelshtein, M. R.; Pakhomchik, A. I.; Melnikov, A. A.; Novikov, A. A.; Glatz, A.; Paraoanu, G. S.; Vinokur, V. M.; Lesovik, G. B.

Solving Large-Scale Linear Systems of Equations by a Quantum Hybrid Algorithm

Published in:
Annalen der Physik

DOI:
10.1002/andp.202200082

Published: 01/07/2022

Published under the following license:
CC BY

Please cite the original version:
Perelshtein, M. R., Pakhomchik, A. I., Melnikov, A. A., Novikov, A. A., Glatz, A., Paraoanu, G. S., Vinokur, V. M., & Lesovik, G. B. (2022). Solving Large-Scale Linear Systems of Equations by a Quantum Hybrid Algorithm. Annalen der Physik, 534(7), 1-10. Article 2200082. https://doi.org/10.1002/andp.202200082

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.
Solving Large-Scale Linear Systems of Equations by a Quantum Hybrid Algorithm

M. R. Perelshtein, A. I. Pakhomchik, A. A. Melnikov, A. A. Novikov, A. Glatz, G. S. Paraoanu, V. M. Vinokur,* and G. B. Lesovik

Today’s intermediate-scale quantum computers, although imperfect, already perform computational tasks that are manifestly beyond the capabilities of modern classical supercomputers. However, so far, quantum-enabled large-scale solutions have been realized only for limited set of problems. Here a hybrid algorithm based on phase estimation and classical optimization of the circuit width and depth is employed for solving a specific class of large linear systems of equations ubiquitous to many areas of science and engineering. A classification of linear systems based on the entanglement properties of the associated phase-estimation unitary operation is introduced, enabling a highly efficient search for solutions that is facilitated by a straightforward matrix-to-circuit map. A 2^{17}-dimensional problem is implemented on several IBM quantum computer superconducting quantum processors, a record-breaking result for a linear system solved by a quantum computer. Demonstrated realisation sets a clear benchmark in the quest for the future quantum speedup in the linear systems of equations solution.

1. Introduction

Since Peter Shor’s discovery of the factorization algorithm in 1994,[1] a search for quantum algorithms for solving classically intractable problems has become one of the mainstreams in quantum computing science. Often referred to as quantum computational advantage,[2] the enhancement in the processing capabilities requires nearly-ideal unitary operations, with fidelities that are beyond the capabilities of existing experimental setups. Despite these obstacles, state-of-the-art noisy intermediate-scale quantum devices,[3] although imperfect, have still the potential to go beyond the capabilities of modern classical supercomputers. Yet, there are only a few quantum algorithms that have been tested on a large scale up to date. The examples of such algorithms were implemented on superconducting quantum processing units (QPUs) for the specific problems of sampling a random circuit.[4–6]

Here we focus on a widely demanded problem of the engineering and science, finding the solutions of linear systems of equations, and compare the standard classical methods for solving this problem with the quantum protocol based on the Harrow–Hassidim–Lloyd algorithm[7] (HHL). At the core of this algorithm is the phase estimation (QPE) protocol,[8] which has been extensively studied and applied as a subroutine in a variety of quantum algorithms (factorization, quantum counting, calculation of the eigenvalues of unitary matrices). By virtue of the QPE, a quantum computer gains an exponential speedup in the context of the matrix inversion problem.[7]
This algorithm is further optimized by a classical analysis,[9] allowing for the elimination of some of the phase qubits, which is referred to as the hybrid HHL (h-HHL) algorithm.

In this work, we put forth the solution based on the h-HHL algorithm to which we add classical optimization of the corresponding circuit. This enables us to classify the linear systems into quantum complexity classes characterized by different entanglement cluster structures that can be effectively encoded and solved using the quantum hardware. Such classes, although limited by certain matrix structures, allow us to simplify the corresponding quantum circuits and reduce the influence of the noise existing in current quantum computers.

Our experiments, carried out on the high-end simulator and on IBMQ superconducting processors,[10] demonstrate the quantum algorithm’s capabilities in comparison with standard conventional classical solutions for linear systems. Our results evidence and highlight the main advantage of the HHL approach, the exponential compression of data, providing a notable improvement of the computational resource allocation. A thorough experimental analysis enables us to spot the shortcomings of the modern QPUs and outline solutions for surpassing them and boosting large-scale quantum data processing.[11] To the best of our knowledge, our work presents the first successful attempt for certain types of matrices this problem has been solved.[16,17] the general solutions being presented in ref. [18]. One of the most promising approaches is based on quantum walks, for example, the simulation for d-sparse A matrix is performed using less than \( O((d\|A\|_{\infty})^2) \) queries. Here, to focus on the phase estimation part of the quantum algorithm, we consider only particular matrices A for which the exponentiation can be efficiently performed, as discussed further. Furthermore, the HHL algorithm works with Hermitian matrices, therefore, if the matrix A is not Hermitian we use A and \( A^\dagger \) as the off-diagonal elements in the block 2 x 2 matrix so that the resulting matrix becomes Hermitian.[7]

2.2. Structure of the Algorithm

Let us briefly discuss the structure of the quantum algorithm for linear systems. The HHL algorithm that inverts a \( N \times N \) matrix exploits three groups of qubits: the vector register that consists of \( \log N \) qubits, \( p \) qubits for phase register, and a single qubit ancilla register, \( n = \log N + p + 1 \) qubits in total. The whole computation, in turn, consists of the QPE algorithm, the ancilla quantum encoding (AQE) part, in which the single ancillary qubit conditionally operates on the state of the phase registers, and the inverse QPE. Here, we omit the quantum amplitude amplification step[7] that improves the complexity in terms of the condition number, which is fixed in the considered problems. The phase estimation protocol exploits controlled unitary operations \( C\hat{U}, C\hat{U}^2, \ldots, C\hat{U}^{p-1} \) over phase and vector qubits, followed by the quantum Fourier transform, in order to process the eigenvalues and eigenvectors of the matrix. Throughout this paper, we consider particular linear systems, which requires \( p = 3 \) phase qubits to encode all matrix eigenvalues. The quantum circuit of the HHL algorithm that utilizes three phase registers is depicted on Figure 1a.

In order to implement the HHL algorithm we use the spectral decomposition of the \( N \times N \) matrix

\[
\hat{U} = \sum_{j=1}^{N} e^{x_j \hat{b}^\dagger \hat{b}} |u_j\rangle \langle u_j|
\]

and encode the vector \( \vec{b} \) into the qubit’s amplitudes \( |b\rangle = \sum_{j=1}^{N} b_j |j\rangle \).

Once we run the algorithm, the solution is encoded into the quantum state

\[
|x\rangle = \frac{(\log \hat{U})^{-1} |b\rangle}{\mathcal{N}_x} = \frac{1}{\mathcal{N}_x} \sum_{j=1}^{N} \frac{1}{\lambda_j} |u_j\rangle
\]

where \( \mathcal{N}_x \) is the normalization coefficient. Using projective measurement, we obtain the expectation value \( \langle x \rangle \) for some operator \( \hat{M} \) in exponentially shorter time than that allowed by the classical algorithms.[7]

2.3. Classical Component of the Algorithm

The HHL algorithm is improved by using some prior knowledge about the matrix A: this classical information allows us to refine
Figure 1. a) Quantum scheme of the original HHL algorithm with three phase qubits that employs the quantum phase estimation protocol (QPE) and the ancilla quantum encoding step (AQE) in order to solve Equation (3) with a unitary matrix $\hat{U}$. While the QPE part exploits controlled unitary operations $C\hat{U}$, $C\hat{U}^2$, $C\hat{U}^4$, and quantum Fourier transform in order to process eigenvalues and eigenvectors of $\hat{U}$, the AQE algorithm assists the matrix inversion. b) Quantum scheme of the simplified hybrid HHL algorithm, which provides the solution for the same matrix as the circuit from (a). The algorithm involves only two phase registers, and the QPE part is significantly reduced. c) The circuit depth as function of problem size for TP1 (circles), TP2 (triangles) and NTP (crosses) matrix types processing by the original HHL and by the simplified h-HHL. d–f) Quantum circuits that generate three types of $2^3 \times 2^3$ $\hat{U}$: $U_{TP1}$ that involves only single-qubit gates $U_i$; $U_{TP2}$ that localizes two-qubit clusters with a single layer of CNOTs; $U_{NTP}$ that entangles all qubits except the first. All types also contain the $U_i$ operator, which is used for the setting of the matrix spectrum. g) The scheme of a controlled-unitary gate $C\hat{U}_i$ of the single-qubit operation $U_i$ that involves single two-qubit interaction.

the quantum algorithm in order to downsize the noise-sensitive quantum part. We implement a hybrid HHL algorithm (h-HHL) presented in ref. [9] that takes advantage of the fact that some bits of $A^{-1}$ eigenvalues could be the same for any eigenvector. Since the QPE part encodes eigenvalues into the phase register qubits, we determine the identical bits of eigenvalues and, as a consequence, qubits corresponding to those bits. We treat such phase qubits as classical bits and exclude them from the computational scheme yielding a reduction in the width and in the depth of a circuit. In general, one can apply an iterative quantum phase estimation algorithm\textsuperscript{(19)} in order to determine the identical bits of the eigenvalues. Using the iterative algorithm, we can find a certain phase with precision $1/2^{m + 2}$ and error probability less than $\varepsilon$ via $O(m \log(m/\varepsilon))$ experiments, resulting in additional $O(\log(N))$ complexity. In contrast to previous works, here we focus on the large-scale implementation. For this purpose, we theoretically introduce and experimentally realize special classes of linear systems that can be solved using the h-HHL algorithm without limitation on the size of the system.
2.4. Benchmarking Problem

Here we discuss the class of matrices that we consider in this work. As an exemplary task, we choose an efficient solution of the following system of linear equations

\[
\begin{align*}
\frac{1}{2\pi i} \log \hat{U} x &= \hat{b} \\
\phi \left( \frac{1}{2\pi i} \log \hat{U} \right) &< 1
\end{align*}
\]

(3)

where \( \hat{U} \) and \( \hat{b} \) are a given matrix and vector, respectively, and \( \phi(\log \hat{U}) \) is the spectral radius of \( \log \hat{U} \). Since the logarithmic function is ambiguous, we fix the resulting matrix spectrum such that the largest absolute value of eigenvalues is less than 1.

To construct the quantum circuit that realizes the operator \( \hat{U} = e^{iA} \), we consider the matrix \( \hat{U} \) comprising quantum gates instead of decomposing the exponent of arbitrary matrix \( e^{iA} \) into the single-qubit and control-NOT gates. Furthermore, we choose the computational basis \( | b \rangle = |0 \rangle \). The entire family of the gate-based matrices is expressed as a tensor product of local operators \( \hat{U}_i \) that act only on the i-dimension subset of the computational circuit

\[
\hat{U} = \bigotimes_{i=1}^{M} \hat{U}_i
\]

(4)

To illustrate the capability of the quantum algorithm, we consider three types of such \( \hat{U} \) operators. First of all, we choose the block structure that consists of 2 × 2 blocks, 4 × 4 blocks and \( \dim(U) \) blocks. We divide the physical operators \( \hat{U} \) that correspond to this structure into three groups:

\[
\begin{align*}
\text{TP}_1: & \quad \hat{U}_{\text{TP}_1} \text{ is the Tensor product of single-qubit gates } \hat{U}_i, \text{ resulting in } \dim(U_i) = 2: \text{this operator does not entangle qubits. See Figure 1d.} \\
\text{TP}_2: & \quad \hat{U}_{\text{TP}_2} \text{ is the Tensor product of two-qubit operators, } \dim(U_i) = 2^2 \text{ that leads to the emergence of two-qubit clusters within which qubits are entangled. See Figure 1e.} \\
\text{NTP:} & \quad \hat{U}_{\text{NTP}} \text{ is not a tensor product of single- or two-qubit gates that leads to } \dim(U_i) = \dim(A): \text{this operator entangles all qubits. See Figure 1f.}
\end{align*}
\]

This hierarchy of unitaries, classified on the basis of entangle- ment properties, may be used as different levels of matrix approximation: while \( \hat{U}_{\text{TP}_1} \) provides access to simple block-structured matrices, \( \hat{U}_{\text{NTP}} \) is able to capture more correlations and approximate matrices with higher accuracy. Here, we set the structure of linear systems according to the capabilities of current quantum computers: solution of more complex systems, where the matrix can not be represented as a simple circuit, provides diminutive experimental fidelity and the proper comparison between classical and quantum solvers would be impossible.

The QPE protocol, which is a vital subroutine of the studied algorithm, involves controlled unitary operations and, therefore, is the most complicated part. Indeed, the realization of an arbitrary control single-qubit gate in \( \hat{U} \) requires two CNOTs, \( [8] \) which leads to a highly complex quantum circuit, for example \( [\log \hat{N}] \)-qubit \( \hat{U}_{\text{NTP}} \) comprises up to \( 2 \times [\log \hat{N}] \) single-qubit gates. Thus, we consider the continuous subset of single-qubit gates \( \hat{U}_i \) in a way that the control- \( \hat{U}_i \) is implemented via a single CNOT resulting in a dramatic simplification of the algorithm circuit, while this gate can be an arbitrary unitary and, simultaneously, Hermitian, so that \( U_i^\dagger = 1 \). The circuit for the control- \( \hat{U}_i \) is depicted in Figure 1g.

The simplification of controlled unitaries presented here is not unique. It is known that the quantum Fourier transform requires \( O(n^2) \) two-qubit operations, however it can be simplified for Ising-type multi-qubit systems \( [20] \) comprising \( O(n) \) operations. The further simplification can be made by discarding rotations with small angles. The latter allows for the complexity reduction to \( O(n \log(n)) \) operations maintaining decent accuracy. \( [21] \) In the considered benchmarking problem we initially reduced the circuit width, therefore eliminating the need to use such methods.

In order to demonstrate the classical part of the quantum algorithm, we manually control the spectrum \( \{ \lambda \} \) of \( A \). For that purpose, we introduce a correcting single-qubit gate \( \hat{U}_s \) in the quantum circuit; this assumption significantly simplifies the analysis of a quantum solution and, at the same time, does not lead to loss of generality.

Leveraging the full control over the matrix spectrum via the correcting gate \( \hat{U}_s \), we immediately eliminate the unnecessary phase qubits without the iterative phase estimation algorithm.

In this work, we adjust the correcting gate in such a way that \( \lambda \in \{ \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}} \} \). Such a correction allows us to utilize \( p = 3 \) phase qubits to encode the whole spectrum of \( N \times N \) \( \hat{U}_{\text{TP}_1}, \hat{U}_{\text{TP}_2} \), or \( \hat{U}_{\text{NTP}} \) matrices. One can introduce more sophisticated correcting gates that complicate the matrix spectrum, but, hereinafter, we focus on a given example. Leveraging the fact that the last bit of each eigenvalue equals 1 in the binary representation: \( \lambda = \frac{[0,1]}{\frac{1}{2}}, \frac{1}{\sqrt{2}}, \alpha, \beta \), where the different combinations of \{0,1\} correspond to different eigenvalues, we immediately eliminate one phase qubit without conducting the iterative phase estimation procedure resulting in \( p = 2 \) phase qubits instead of \( p = 3 \) in order to solve the linear system; \( n = [\log N] + 3 \) qubits in total. The scheme of the h-HHL algorithm is shown in Figure 1b. Such a technique allows us to significantly reduce the depth of the circuit, by an order of magnitude for the 50-qubit scheme. The circuit depth of the simplified h-HHL (red) and the original HHL (blue) as function of a problem size is shown in Figure 1c for the discussed matrix types. The circuit depth is calculated directly by evaluating the number of two-qubit gates that can not be realized simultaneously for each matrix type (for more details see Section 5).

2.5. Small-Scale Quantum Algorithm Implementation

Here, we demonstrate the performance of the h-HHL algorithm, which employs 2 phase and a single ancillary qubit, using real QPs provided by IBMQ.\(^{[10]} \)

Primarily, we consider a low number of qubits case where \( n \in [4, 7] \), three qubits that are assigned to the phase and ancillary registers. Under these conditions we employ the full state tomography analysis, which requires \( 3^{n-1} \) experiments with one circuit; each experiment consists of 8912 runs. We investigate all matrix types TP1, TP2, and NTP by analyzing 140 transpiled random quantum circuits (RQCs) for every type for every matrix size.

---

\(^{[8]}\) H. K. Krieger, Quantum Information Processing 18, 279 (2019).

\(^{[10]}\) A. N. Akhshabi et al., Nature 590, 557 (2021).
Quantum volumes \cite{22} for different h-HHL circuit widths. Since the measured fidelity averaged over 70 random terms of complexity analysis. The fidelity of the full state to the ideal solution, solid squares depict the density matrix. While blue color indicates positive values, red color indicates negative values; the size of a square shows the absolute value. b) The fidelity \( F_{\text{tom}} \) of the algorithm circuit that was averaged over 70 NTP RQCs. Since the quantum volume of Burlington, Yorktown, and Melbourne processor is the same \( V_Q \), the fidelity behavior is similar. However, Johannesburg possess slightly higher \( V_Q \) = 16, which is reflected in the fidelity.

Figure 2. Results of the full state tomography at the end of the h-HHL algorithm for different QPUs: IBMQ Burlington, Yorktown (both 5 qubits), Melbourne (15 qubits) and Johannesburg (20 qubits). a) The Hinton plot of an example of a density matrix, which corresponds to the vector register state—ancilla and phase registers were filtered out—that contains the solution of Equation (5): dashed squares depict the density matrix that corresponds to the ideal solution, solid squares depict the measured density matrix. While blue color indicates positive values, red color indicates negative values; the size of a square shows the absolute value. b) The fidelity \( F_{\text{tom}} \) of the algorithm circuit that was averaged over 70 NTP RQCs. Since the quantum volume of Burlington, Yorktown, and Melbourne processor is the same \( V_Q \) = 8, the fidelity level is similar. However, Johannesburg possess slightly higher \( V_Q \) = 16, which is reflected in the fidelity.

2.6. Large-Scale Quantum Algorithm Implementation

Here we consider large circuits performance, but, at first, let us elaborate on a suitable performance metric. Since the full state tomography requires \( 3^n \) experiments for \( n \)-qubit circuit, it is not possible to evaluate the fidelity by tomography in a reasonable time. Thus, we employ a similar cross entropy benchmarking approach as in ref. \cite{4}, which allows us to estimate the algorithm fidelity with only one experiment and a single Z-projective measurement that gives us the probability distribution of outcomes. Let us define the fidelity as follows

\[
F_{\text{XEB}} = \frac{\sum_{j=0}^{M} (p_j^c - \tilde{p}_j^c) (\tilde{p}_j^c)}{\sum_{j=0}^{M} (p_j^c - \tilde{p}_j^c)}
\]

where \( \langle \cdot, \cdot \rangle \) is a scalar product. The \( \tilde{p}_j^c \) and \( p_j^c \) are probability distribution of outcomes, which correspond to the noiseless implementation and experimental run of jth circuit, respectively; \( p_j^c \) is the uniform probability distribution. In the above formula, we average the probability distribution over M different jth RQCs. It is clear that the introduced metric shows averaged proximity of the obtained vector projection to the ideal solution rather than to a chaotic state—such a definition matches the cross entropy fidelity\cite{4}. More details on the algorithm characterization can be found in the Supporting Information.

However, we find that the \( V_Q \) of the Johannesburg QPU is twice as large, resulting in a slightly higher fidelity level.
Melbourne simulator

Figure 3. Cross entropy fidelity $F_{XEB}$ of the h-HHL algorithm as a function of a total number of qubits characterized for the IBMQ 15-qubit Melbourne and 20-qubit Johannesburg processors. 

a) The fidelity obtained on a simulator with an embedded depolarizing noise model based on the gate and readout errors measured for the IBMQ Melbourne device (filled crosses) for TP$_1$ (blue), TP$_2$ (orange), and NTP (red) matrices. The solid lines indicate the digital error model. It is clear that the simulator, which is devoid of any spatial or temporal correlations of gate errors, is described by the DEM. 

b) The experimental results measured on the Melbourne QPU (empty crosses); colormap matches the matrix types. The same type of measurements was performed for the 20-qubit Johannesburg processor c) on the simulator with a noise model and d) on the real device. The experimental results are not fitted by DEM by reason of correlations of control errors, which are specific to IBMQ devices.

The noise model includes the gate and readout errors that were measured beforehand. Each point is the fidelity that is averaged over 140 RQCs, which were topologically optimized in order to get the minimal depth, and each RQC experiment consists of $10^3$ runs on the simulator as well as on the QPU.

Besides, we consider the digital error model (DEM), which takes into account gates and readout errors and characterizes circuit performance by a set of localized Pauli errors; measured errors are presented in Supporting Information. For the estimation of the algorithm performance on a large QPU via DEM, we assume that only the fidelity of readout $F_r$, single-qubit gates $F_{1QG}$, and two-qubit gates $F_{2QG}$ contribute to the final fidelity resulting in

$$F_{XEB} = F_r \times F_{1QG} \times F_{2QG}$$  \hspace{1cm} (7)

The digital error approach can be used to predict the fidelity behavior when there is no space or time correlation between gate errors, which is the case in the simulator with an embedded depolarizing noise model, see Figure 3a,c. However, for the experiment conducted on IBMQ QPUs, we find that the DEM fails in predictions of the algorithm performance. It becomes clear from Figure 3b,d that the measured fidelity is almost independent on the matrix type despite the significant changes in corresponding circuits. Indeed, it was confirmed earlier that gate errors are correlated in some of the IBM superconducting processors. Nonetheless, we expect that an advanced equipment combined with the necessary precautions against the correlated errors, for example, Purcell filters, will provide the fidelity behavior according to the digital error model as was shown in ref. [4].

In this work, we experimentally find the quantum state projection that corresponds to the solution of a $2^{17} \times 2^{17}$ system of linear equations for each considered matrix type, which appears to be a record in matrix inversion on a gate-based QPU. Previously, the $8 \times 8$ problem was solved with an adiabatic-inspired quantum algorithm on a four-qubit nuclear magnetic resonance device and the $32 \times 32$ problem was tackled with variational quantum linear solver using Rigetti’s 16-qubit superconducting chip. Introduced circuit simplifications allow us to solve and analyze larger problem.

Examples of linear systems that were solved on a QPU in the scope of this work can be found in a GitHub repository. The solution for all types of linear systems was tested using the quantum circuit simulation software in order to verify the algorithm accuracy. As a result, we observe that the projection found by the
h-HHL algorithm corresponds to the actual solution with the machine precision indicating clear scalability of the quantum solution.

3. Discussion

3.1. Classical Approaches

Here we discuss classical algorithms for solving the linear system $A\vec{x} = \vec{b}$, where $A$ has a fixed spectrum and is dense, which is the case of Equation (3). Our analysis is carried out on NTP-type of system, as the most interesting in terms of complexity.

Analyzing the computational complexity, we omit the condition number scaling since in the scope of this work this number is fixed for all the considered problems. Such fixed spectrum can be exploited in the iteration eigenvalue algorithms. However, such an algorithm involves multiplications of matrices that lead to polynomial scaling in time and memory, resulting in $O(N^2)$ scaling with the system size. On the other hand, in order to compute $\hat{U}$ one can utilize matrix diagonalization or the inverse scaling and squaring method. Such techniques require at least an exponential number of operations or addresses to matrix elements.

The most popular solver for linear system of equation with Hermitian and positive-definite matrix is the conjugate gradient method that has a scaling $O(N^2)$ for dense matrices since it requires matrix-by-vector multiplication at each iteration step, thus, even a single step of iterative methods is computationally expensive. One can also apply LU-decomposition and Gaussian elimination methods, but the scaling is worse in that case: $O(N^3)$. Therefore, the classical computational cost of solving the system with $N \times N$ matrix using advanced numerical methods is no less than $O(N^2)$ regardless of mentioned assumptions.

3.2. Comparison with Classical Algorithms

As we have shown before, while the runtime of the classical algorithms that solves $N \times N$ linear system is $O(N^2)$, the runtime of the hybrid quantum algorithm has at max $O(\text{poly}(\log N))$ scaling. We expect that future improvements in classical algorithms and hardware will provide a considerable reduction in runtime and computational resources, however, persistent enhancement of quantum software allows hybrid quantum approaches to consistently outperform classical solutions. Besides, we find that the high-performance circuit emulation is considerably faster than the direct solution: $O(N^2)$ versus $O(N \log N)$, indicating that simulation of a quantum circuit is essentially an efficient tool for the solution and a proper reference for the quantum hardware to compare. Here, we consider the straightforward Schrödinger–Feynman simulation presented in ref. [4].

For algorithmic benchmarking, let us compare the efficiency of the $2^{17} \times 2^{17}$ NTP problem solution implemented on a QPU and a Schrödinger–Feynman simulation implemented on the most significant supercomputers in history. In order to properly compare classical approaches and quantum devices, we must take into account the final fidelity, since the equal-fidelity computation time scales linearly with $F_{\text{XEB}}$ as was shown in ref. [4].

While the net calculation time on a quantum device with 0.04% fidelity does not exceed 1 ms (assuming several runs to ensure the ancillary qubit is measured in [1]), the equal-fidelity classical solution would require seven orders of magnitude higher time for the supercomputers from 1960, five orders for devices from 1970 and few times higher for computers from 1990; the detailed comparison is shown in Figure 4. Note that the estimations for the simulation performance are over evaluated for supercomputers that do not have enough memory to process the whole circuit. We thus omit the final vector recreation runtime since the estimation of such an operation is a highly complicated task. Such a dramatic increase in speed is an experimental evidence that the hybrid quantum computing will surpass the classical computing industry that reigned in the XXth century.

For clarity, we specify that the comparison above was made with the simulation employing the standard matrix multiplication. However, we expect that more advanced quantum simulation approaches, such as tensor networks, may allow for a solution in poly-logarithmic time and provide several advantages. Interestingly, since the classical computer is not restricted to unitary operations, one gets rid of the ancillary qubit by performing...
a necessary non-unitary transformation via the tensor network. While general gate decomposition of multi-qubit AEQ is challenging, tensor networks avoid such an issue. Besides, classical simulation allows us to extract the whole solution vector rather than a projection that, on the other hand, requires a substantial memory. Nevertheless, this kind of quantum-inspired solution, which is devoid of many hardware-related problems, would show the power of quantum algorithms even more vastly. The investigation of these methods is a subject of the forthcoming works. The clear disadvantage of classical simulation methods originates in the exponential complexity growth with the increase in phase qubits number $p$.

For linear systems with a rich matrix spectrum, a hybrid algorithm implemented via advanced hardware would be a key component for solving large-scale problems. Since required quantum resources are not immediately available on near-term quantum devices, hybrid quantum approach can be improved by using variational algorithms. Such methods are based on circuits with changing parameters, whose optimization landscape may be very complex and non convex resulting in barren plateaus. Similarly to our work, a certain class of linear systems still can be solved, for example, where the matrix is represented as small linear combination of known unitaries or even a tensor product of Pauli operators. The serious drawback of such methods is that they are equivalent to the gate-based optimization and may not provide a solution in polynomial time being run on a perfect quantum device, while HH algorithm always finds the solution, sometime at the cost of deep circuit for a complex linear system.

3.3. Near-Term Applications

Finally, we discuss the possible practical applications of the H-HHL algorithm in the framework of the posed problem. Despite the fact that the fidelity level is low, the quantum algorithm still provides the solution with a tremendously higher probability than a random guess. For instance, for a given $2^{17}$-dimensional problem, the fidelity of a random guess is $1/2^{17} \approx 7.6 \cdot 10^{-6}$, while the QPU-run algorithm provides two orders of magnitude better fidelity $4 \times 10^{-4}$.

The exact realization of our algorithm allows for addressing some tangible problems, for instance, in the context of Markov processes, one could obtain the generator of a known stochast-ic $P$ matrix—a transition rate matrix $Q = \log P$. Then, we can solve the linear equation $Q \hat{f} = \Delta \hat{f}$ at some point of the process. Here, $\hat{f}$ is the distribution of state probabilities and $\Delta \hat{f} / \Delta t$ is a probability current, which is supposed to be known. Upon obtaining $\hat{f}$ and measuring $\langle f | M | f \rangle$ we check that the probability of a specific state is non-zero.

An analogous problem can be addressed in control theory. Let us suppose that a discrete-time process with time step $\Delta t$: $\hat{x}_{n+1} = A \hat{x}_n$ is modeled by a continuous one: $\dot{\hat{x}} = B \hat{x}$. Since $A = \exp(B \Delta t)$, one could obtain the vector $\hat{x}$ by solving $\dot{\hat{x}} = B \hat{x}$ at some point of the process if $\hat{x}$ is provided. By averaging proper Hermitian operators $\langle x | M | x \rangle$ we determine whether some components of $\hat{x}$ are non-zero.

The algorithm can be modified by applying matrix simulation techniques for $A$ in order to solve $A \hat{x} = \hat{b}$ as it was originally proposed in ref. [7]. In that case, a wider range of problems can be approached. For instance, it was proposed to use the SWAP test in order to determine whether solutions of different systems coincide. Such linear systems are also used to solve partial differential equations, for example, one can find the electromagnetic field energy in some region using the Poisson equation. One of the promising applications related to deep neural network training was discussed in ref. [37]: since the extension of the Bayesian approach to deep architectures is a serious challenge, one can exploit the hybrid quantum HHL algorithm developed for Gaussian processes in order calculate a model’s predictor.

Finally, our classification of unitaries applies to optimizing metrological tasks on superconducting-circuit platforms, which also utilize phase estimation as the core subroutine.

4. Conclusion

In this work, we implemented the quantum hybrid HHL algorithm solving a system of linear equation by fast matrix inversion. The matrix, in turn, is approximated by a unitary transformation, which was dictated by the sequences of single-qubit rotations and CNOT gates organized in a shallow circuit. We introduced classes of matrices that allow us to significantly reduce the complexity of the circuit, while the size of the linear system grows exponentially with the increasing number of qubits.

We probed the algorithm on the simulator with an embedded noise model, which does not include correlations in gate errors, and on the real IBMQ QPUs with 5, 15, and 20 qubits. The implemented quantum solution of a $2^{17}$-dimensional problem is a record for a linear system solution on quantum computers.

The main limitation of our work is the considered matrix structures. In case of the unstructured matrix, whose exponent is hard to represent as a circuit (Hamiltonian simulation), the direct decomposition produces enormously deep circuit, realization of which is not practical. Our simple example sets a clear benchmark that can be used during the evaluation of more sophisticated algorithms for more complex matrices. We envision that the demonstrated experiments will stimulate major players in the quantum computing industry to investigate the quantum hardware and software aiming at solution of linear systems of equations.

5. Experimental Section

Linear System Construction: Here, the implementation of $T P_1$, $T P_2$, and NTP matrix types are considered in more detail. In order to construct quantum circuits the Qiskit open-source framework, which allows to generate $U$ from single-qubit and two-qubit CNOT gates was used. An arbitrary single-qubit gate was decomposed by Qiskit into physical gates: the $U_1$ operator, which is the most general gate, is defined as follows

$$U_1(\theta, \phi, \lambda) = \begin{pmatrix} \cos \frac{\theta}{2} & -e^{i\phi} \sin \frac{\theta}{2} \\ e^{i\phi} \sin \frac{\theta}{2} & e^{i\lambda} \cos \frac{\theta}{2} \end{pmatrix} \tag{8}$$

Since the QPE protocol, which is a vital part of the H-HHL algorithm, involved controlled unitary operations, single-qubit gates $U_i$ and CNOTs in $U$ were extended to a control $C U_i$ and Toffoli gates, respectively. However, the realization of an arbitrary control single-qubit gate $C U_i$ demanded...
two CNOTs, which led to a highly complex quantum circuit. Thus, the subclass of single-qubit \( \hat{U} \), for which the \( C\hat{U} \) requires only one CNOT, resulting in dramatic simplification of the h-HHL circuit was considered.

Let us define the operator \( \hat{U}_j = \hat{U}_1(X, Y, Z) \hat{U}_1^j \), where any of the Pauli matrices \([X, Y, Z]\) can be chosen. Such a form led to the following \( C\hat{U}_j \) representation

\[
C\hat{U}_j = |0\rangle \langle 0| \otimes \hat{U}_j |1\rangle \langle 1| \otimes \hat{U}_j (X, Y, Z) \hat{U}_j^\dagger =
= |0\rangle \langle 0| \otimes I + |1\rangle \langle 1| \otimes \hat{U}_j
\]

where the Pauli gate is applied only when the control qubit is in the excited state, see Figure 1g. It was clear that the \( \hat{U}_j \) gate was essentially an arbitrary Hermitian \( \hat{U}_j \) gate, which can be build according to definition Equation (8) with the following condition

\[
\lambda + \phi = (2k - 1)\pi, \text{ where } k \in \mathbb{Z}
\]

Such a definition ensured that all controlled single-qubit gates contained only one two-qubit operation and, since \( \hat{U}_j \) is a Hermitian gate, it had only two eigenvalues \( \{1, -1\} \). Unfortunately, it led to another problem: studied \( \log(U_{TP1}) \), \( \log(U_{TP2}) \), \( \log(U_{XEB}) \) matrices had zero eigenvalues, which was unacceptable for the HHL algorithm. However, some advanced techniques including linear combinations of unitaries or block-encoding worked even when matrix was not invertible as long \( \tilde{b} \) is known to lie outside the null space of \( \mathcal{A} \). In order to avoid such an issue in the context of HHL algorithm, the following correcting gate was added to the first register of \( \bar{U} \) was added

\[
\hat{U}_j_x(\pi/2, 0, 0) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}
\]

The introduction of a such correcting gate led to two important and useful features: i) the spectrum of all matrix types is \( \exp(2\pi i k) \cdot \{\pm 1, \pm 3, \pm 7\} \); ii) \( U_{TP1, XEB} = U_{TP2} = U_{XEB} \) which greatly simplified the QPE part.

The knowledge about the matrix spectrum in the classical part of the h-HHL algorithm was exploited. Since the binary form of \( \log(\hat{U}) \) eigenvalues contained three bits, one needs three phase qubits to store whole spectrum. It was easy to show that the last bit of all eigenvalue is \( 1 \) if the four eigenvalues in binary representation were written down: \( \lambda = (0.1) + (0.1) + 1 \), where the different combinations of \( 0, 1 \) corresponded to different eigenvalues. Therefore, one phase qubit without conducting the iterative phase estimation procedure resulting in \( p = 2 \) was immediately eliminated. Hence, \( n = \log(N) + 3 \) qubits were utilized overall for each matrix type. The 5-qubit examples of \( \hat{U}_{TP1}, \hat{U}_{TP2} \) and \( \hat{U}_{XEB} \) operators are depicted on Figure 1d–f, respectively.

Circuit Emulation: Since the classical computational cost of solving Equation (3) with \( N \times N \) matrix was no less than \( \mathcal{O}(N^2) \), an alternative method, the high-performance simulation of the h-HHL quantum circuit, should be considered. The simulation using the Schrödinger algorithm (SA)[41] on one hand, was straightforward and provided a great speed in processing low-width circuits in comparison to the other simulations. On the other hand, the Schrödinger algorithm required a significant amount of RAM when processing many-qubits circuits. Such a requirement is difficult to meet since the memory of modern supercomputers is substantially limited. However, the Schrödinger–Feynman algorithm (SFA) solved the memory issues inherent in a SA, which presumably made it the fastest simulation of high-depth high-width quantum circuits.

Primarily, in order to estimate the classical simulation runtime, Schrödinger simulations, which are essential building blocks of SFA; the runtime scaling of a single n-qubit SA is \( T_{SA} = C \cdot n^2 \cdot 2^n \) were performed. A POWER8 processor with 160 cores and 312 Gb of RAM[41] was exploited in order to estimate the time constant \( C \)—evaluating the runtime for each circuit width the dependence was fitted and found that \( C_{TP1} = 5 \times 10^{-9} \) s, \( C_{TP2} \approx 14 \times 10^{-9} \) s, and \( C_{XEB} \approx 17 \times 10^{-9} \) s, which were assumed to scale linearly in the number of cores. Since modern supercomputers had roughly 100K cores, the constants were expected to be \( C_{TP1} \approx 10^{-12} \) s, \( C_{TP2} \approx 2.8 \times 10^{-12} \) s, and \( C_{XEB} \approx 3.4 \times 10^{-12} \) s in the best scenario.

For memory estimates, while the state-of-the-art supercomputers had 3 PB of RAM, it was supposed that one can store a \( 2^{12} \)-dimensional vector using 8 bytes to encode a simple complex number. Bore in mind that storing of 48-qubit state took twice RAM. Hence, SFA allowed to split the quantum circuit in the ratio \( (n - \tilde{n}) : \tilde{n} \), where \( \tilde{n} \leq n \leq 47 \). The execution time of the Schrödinger–Feynman method is \( T_{SFA} = C (n - \tilde{n}) \cdot 2^{n-\tilde{n}} \times N_p \), where \( N_p \) is the number of paths in final state vector calculation.[46] The scaling constant \( C \) is the same as in the Schrödinger algorithm, since almost all RAM was forced to use and algorithm parallelization became an elusive task.

In order to compare the 20-qubit QPUs with classical supercomputers, the constant \( C \) was recalculated in seconds per Flops; for the \( T_{NTP} \) circuit it was found that \( T_{SA} = 1.5 \times 10^{11} \times n^2 \) Flops s. For the characterization of the equal-fidelity calculation, it was assumed that the simulation runtime scales linearly with fidelity, therefore the equal-fidelity runtime was \( T_{XEB} \approx T_{SA} \). However, when it was impossible to store the whole vector in the memory the circuit was divided into \( 16 : 4 \) ratio. In that case, a single Toffoli gate was obtained for the cut for the single QPE step, which had two control qubits above the cut and a controlled qubit below the cut. This gate should be decomposed in order to simulate the two parts independently and to combine them afterward to get a solution. There are \( 2^p \) paths for such Toffoli gate that gave \( N_p = 2^p \) in total (due to QPE and inverse QPE).

Using the calculated runtime for \( 2^1 \) 16-qubit circuit emulations, the equal-fidelity runtime for the supercomputers was estimated and the data was plotted in Figure 4.

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

The authors thank Alexey A. Melnikov for reviewing the manuscript and providing valuable comments. M.R.P. and G.S.P. acknowledge support from the Academy of Finland through the Finnish Center of Excellence in Quantum Technology QTF (projects 312296, 336810, 312295, 336813). The work of A.G. at Argonne was supported by the U.S. Department of Energy, Office of Science, Basic Energy Sciences, Materials Sciences and Engineering Division. The research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725. The work of V.M.V., M.R.P., A.I.P., A.A.M., and G.B.L. was supported by Terra Quantum AG.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

A.I.P., M.R.P., and A.A.M. devised the algorithm. A.I.P., M.R.P., and A.G. wrote the code and performed the experiment. M.R.P., A.A.M., A.I.P., and A.A.N. analyzed the algorithm performance. M.R.P., A.I.P., A.A.M., G.S.P., and V.M.V. wrote the manuscript with inputs from all the authors. G.S.P., V.M.V., and G.B.L. initiated and supervised the project. All authors discussed the results and contributed to the work.

Data Availability Statement

The data that support the findings of this study are available in the supplementary material of this article.
