An Early Investigation of the HHL Quantum Linear Solver for Scientific Applications

MUQING ZHENG, Pacific Northwest National Laboratory, USA
CHENXU LIU, Pacific Northwest National Laboratory, USA
SAMUEL STEIN, Pacific Northwest National Laboratory, USA
XIANGYU LI, Pacific Northwest National Laboratory, USA
JOHANNES MÜLMENSTÄDT, Pacific Northwest National Laboratory, USA
YOUSU CHEN, Pacific Northwest National Laboratory, USA
ANG LI, Pacific Northwest National Laboratory, USA

In this paper, we explore using the Harrow-Hassidim-Lloyd (HHL) algorithm to address scientific and engineering problems through quantum computing utilizing the NWQSim simulation package on high-performance computing. Focusing on domains such as power-grid management and heat transfer problems, we demonstrate the correlations of the precision of quantum phase estimation, along with various properties of coefficient matrices, on the final solution and quantum resource cost in iterative and non-iterative numerical methods such as Newton-Raphson method and finite difference method, as well as their impacts on quantum error correction costs using Microsoft Azure Quantum resource estimator. We conclude the exponential resource cost from quantum phase estimation before and after quantum error correction and illustrate a potential way to reduce the demands on physical qubits. This work lays down a preliminary step for future investigations, urging a closer examination of quantum algorithms' scalability and efficiency in domain applications.

CCS Concepts: • **Computer systems organization → Quantum computing**; • **Applied computing → Physical sciences and engineering**.

Additional Key Words and Phrases: quantum computing, linear system solvers, scientific applications, resources estimation

ACM Reference Format:
Muqing Zheng, Chenxu Liu, Samuel Stein, Xiangyu Li, Johannes Mülmenstädt, Yousu Chen, and Ang Li. 2024. An Early Investigation of the HHL Quantum Linear Solver for Scientific Applications. 1, 1 (May 2024), 21 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Starting with the Deutsch–Jozsa algorithm and Shor's discrete logarithm algorithm [20, 62], the potential of quantum computing algorithms has extended beyond merely simulating quantum systems. The potential speedup of quantum algorithms over their classical counterparts has gathered tremendous attention, including a fundamental demand in

Authors' Contact Information: Muqing Zheng, muqing.zheng@pnnl.gov, Pacific Northwest National Laboratory, USA; Chenxu Liu, chenxu.liu@pnnl.gov, Pacific Northwest National Laboratory, USA; Samuel Stein, samuel.stein@pnnl.gov, Pacific Northwest National Laboratory, USA; Xiangyu Li, xiangyu.li@pnnl.gov, Pacific Northwest National Laboratory, USA; Johannes Mülmenstädt, johannes.muelmenstaedt@pnnl.gov, Pacific Northwest National Laboratory, USA; Yousu Chen, yousu.chen@pnnl.gov, Pacific Northwest National Laboratory, USA; Ang Li, ang.li@pnnl.gov, Pacific Northwest National Laboratory, Richland, WA, USA.

Manuscript submitted to ACM

science and engineering: solving linear systems. Harrow, Hassidim, and Lloyd (HHL) first developed a quantum linear solver with an exponential speedup in problem dimensions in [31]. Built upon the exponential speedup of quantum linear system algorithms (QLSAs), many works have explored theoretical quantum advantages in various applications. These fields include portfolio optimization [57], machine learning [21, 44], differential equation solving [45], linear optimization [10, 49, 50], and semidefinite optimization [5, 48].

However, the HHL algorithm proposed in [31] has a quadratic dependency on matrix condition number and matrix sparsity, worse than classical linear solvers such as factorization methods and conjugate gradient, where condition number is the product of the norm of the coefficient matrix and the norm of the inverse matrix. Several following works have been proposed to reduce the dependency on the condition number of coefficient matrices and the precision of the solution state [2, 4, 11, 14, 17, 18, 33, 63, 67]. Specifically, based on adiabatic theorems, the state of the art has a linear or quasi-linear dependency on the condition number and a logarithmic dependency on the inverse of the solution precision [4, 18, 33, 63].

The HHL algorithm has been demonstrated in experiments to solve linear algebra problems. The largest linear systems demonstrated on real gate-based quantum machines are up to 4×4 systems with variants of the HHL algorithm [52, 59, 70] and an 8×8 system with the linear solver based on adiabatic quantum computing [69]. However, testing QLSAs on real quantum devices to demonstrate a quantum advantage still suffers multiple obstacles, such as the large number of required quantum gates and the high noise level of current quantum devices [55].

With the current development of quantum hardware and exploration of quantum error correction (QEC) codes, a large-scale fault-tolerant quantum computer is expected to be demonstrated in the foreseeable future [7, 19, 27, 30, 37, 47, 68]. With the help of Quantum Error Correction, it is anticipated that QLSAs can be implemented in practical applications for speedup. Although the gap between algorithm requirements and hardware specifications is shrinking, the gap still exists, which necessitates the analysis of the resource costs involved [54]. Resource estimation for chemistry [22], for Grover's algorithm on the Advanced Encryption Standard [29], for Shor's discrete logarithm algorithm for the RSA cryptosystem [24], and the computation of elliptic curve discrete logarithms [58] have been performed. However, despite being essential for understanding the disparity between hardware capabilities and practical applications, there is limited work on non-asymptotic resource estimation for QLSAs [61].

In this paper, we focus on resource estimation and experiment with the HHL algorithm on several applications selected from domain science, such as power grid and thermal diffusion applications. Different from the previous works about asymptotic and non-asymptotic resource analysis [2, 4, 11, 14, 17, 18, 31, 33, 61, 63, 67], we investigate the factors affecting the final accuracy, resource cost, and fault-tolerant hardware requirements. Our experiments show the effectiveness of the HHL algorithm in scientific applications with a low precision in quantum phase estimation. Working with Microsoft Azure Quantum resource estimator, we summarize the exponential dependency of quantum resources on the number of clock qubits in HHL circuits and demonstrate a possible method to reduce the demands on physical qubits in fault-tolerant quantum computing.

The paper is organized as follows. Section 2 introduces the idea of quantum linear system solvers, with implementation-related details. Section 3 presents the simulator, NWQSim [38], and the resource estimation tool. Next, we explore the factors of interest in evaluating numerical experiments in Section 4 and perform those experiments in Section 5. Finally, we discuss the limitations in Section 6 and conclude the implications of our work on domain science applications in Section 7.

2 QUANTUM LINEAR SYSTEMS AND THE IMPLEMENTATION OF THE SOLVER

2.1 Overview of the Harrow-Hassidim-Lloyd (HHL) Algorithm

Quantum information is encoded into the state of quantum systems. Here, we assume all relevant quantum states can be represented as statevectors. An n_d-qubit statevector $|x\rangle = \sum_{j=0}^{2^{n_d}-1} \alpha_j |\vec{j}\rangle$ is a normalized complex vector, i.e., $\alpha_j \in \mathbb{C}$ for all j and $\sum_{j=0}^{2^{n_d}-1} |\alpha_j|^2 = 1$ while $\vec{j} \in \{0,1\}^{n_d}$ is the number j as a binary string. The set $\{|\vec{j}\rangle\}$ forms the basis set of $\mathbb{C}^{2^{n_d}}$, referred as the computational basis. Specifically, $|\vec{j}\rangle$ is the unit vector whose $(j+1)^{th}$ entry is 1 and other entries are 0. The notation $\langle\vec{j}|$ is the conjugate transpose of $|\vec{j}\rangle$.

Definition 2.1 (A quantum linear-system problem). *A quantum linear-system problem is to solve a system of linear equations with a normalized solution vector $|x\rangle = A^{-1}|b\rangle / \|A^{-1}|b\rangle\|_2$ where coefficient matrix $A \in \mathbb{C}^{N\times N}$ is Hermitian and $|x\rangle$ and $|b\rangle$ are both normalized vectors.*

Start with a classical complex linear system $A\vec{x} = \vec{b}$, $M \in \mathbb{C}^{N\times N}$, right-hand-side (RHS) vector $\vec{b}$ is normalized to obtain $|b\rangle := \vec{b}/\|\vec{b}\|_2$, then a quantum linear-system problem can be formed with A and $|b\rangle$ if A is Hermitian. Otherwise, a larger linear system can be constructed as the following [31],

$$\begin{bmatrix} \mathbf{0} & A \\ A^\dagger & \mathbf{0} \end{bmatrix} \begin{bmatrix} \vec{0} \\ |x\rangle \end{bmatrix} = \begin{bmatrix} |b\rangle \\ \vec{0} \end{bmatrix}, \tag{1}$$

where $\cdot^\dagger$ is the conjugate transpose. Therefore, we assume coefficient matrices are Hermitian in the rest of this paper. Since the data is encoded into qubits, if the dimensions of A and $\vec{b}$ are not in the power of 2, A and b must be expanded. Suppose there exists a quantum linear system solver that obtains $|x\rangle = A^{-1}|b\rangle / \|A^{-1}|b\rangle\|_2$ from the circuit, then the original solution of the system can be recovered by $\vec{x} = \|b\|_2 \|A^{-1}|b\rangle\|_2 |x\rangle$. While $\|b\|_2$ is known from the previous computation, the solver needs to provide the value of $\|A^{-1}|b\rangle\|_2$.

2.1.1 *Mathematical Foundation of HHL.*

In [31], Harrow, Hassidim, and Lloyd developed the HHL algorithm to solve the quantum linear-system problem. The fundamental idea behind the HHL algorithm is that the eigenstates of the Hermitian matrix A (noted as $\{|v_j\rangle\}$) form a complete orthonormal basis of $\mathbb{C}^N$ (i.e., $\langle v_j|v_k\rangle = \delta_{jk}$), and hence the state $|b\rangle$ can always be decomposed by this basis as $|b\rangle = \sum_{j=0}^{N-1} b_j |v_j\rangle$. Similarly,

$$|x\rangle = \frac{A^{-1}|b\rangle}{\|A^{-1}|b\rangle\|_2} \tag{2}$$

$$= \frac{1}{\|A^{-1}|b\rangle\|_2} \sum_{j=0}^{N-1} \frac{1}{\lambda_j} |v_j\rangle\langle v_j| \sum_{j=0}^{N-1} b_j |v_j\rangle \tag{3}$$

$$= \frac{1}{\sqrt{\sum_{j=0}^{N-1} \frac{|b_j|^2}{\lambda_j^2}}} \sum_{j=0}^{N-1} \frac{b_j}{\lambda_j} |v_j\rangle . \tag{4}$$

In other words, the HHL algorithm needs a quantum computer to perform eigen-decomposition of A and eigenvalue inversion. Figure 1 shows a general description of the circuit that exactly serves the purpose, with an additional n_d-qubit data-loading block to load $|b\rangle$ into the quantum computer and $n_d = \lceil \log(N) \rceil$.

2.1.2 *Quantum Phase Estimation.*

The eigen-decomposition requires a subroutine called quantum phase estimation (QPE), as illustrated in the sky blue part of Figure 1. Given a unitary matrix U has an eigenstate $|v_j\rangle$ with eigenvalue $e^{2\pi i\theta_j}$, QPE is a quantum algorithm to solve the phase of the eigenvalue (θ_j) [53]. After executing the QPE algorithm, the phase angle θ is stored as the qubit states in a binary representation. The qubits carrying the phase information are named "clock qubits". In the HHL algorithm, if $|v_j\rangle$ is an eigenstate of a Hermitian matrix A with eigenvalue λ_j, by constructing a unitary matrix $U = e^{itA}$ with a scale factor t, the state $|v_j\rangle$ becomes an eigenstate of U with eigenvalue $e^{it\lambda_j}$. Therefore, the eigenvalue λ_j can be estimated using the QPE algorithm.

Suppose we have access to the gate U, it is clear that

$$U^l |v_j\rangle = e^{2\pi i \theta_j l} |v_j\rangle \tag{5}$$

for some positive integer l. QPE requires a submodule called quantum Fourier transform (QFT). QFT maps

$$QFT |\vec{j}\rangle = \frac{1}{\sqrt{2^{n_c}}} \sum_{k=0}^{2^{n_c}-1} \omega^{jk} |\vec{k}\rangle = \frac{1}{\sqrt{2^{n_c}}} \left[\left(|0\rangle + e^{2\pi i 0.j_{n_c}} \right) \left(|0\rangle + e^{2\pi i 0.j_{n_c-1} j_{n_c}} \right) \cdots \left(|0\rangle + e^{2\pi i 0.j_1 j_2 \ldots j_{n_c}} \right) \right]$$

where $\vec{j}$ has n_c qubits and $\omega = e^{2\pi i/(2^{n_c})}$. Intuitively, $\vec{j}$ can be considered as binary number $\vec{j} = j_1 j_2 \ldots j_{n_c}$ such that $j_l \in \{0, 1\}$ and QFT transforms this binary number from a state to the phases of bases in different precision. So, on the contrary, if we apply the inverse of the QFT operator, denoted by $QFT^\dagger$, the phase value becomes a state, and we can measure the state to obtain the phase value in the binary representation.

To summarize the process of a standalone QPE routine, we have

$$\begin{aligned} |0\rangle^{\otimes n_c} |v_j\rangle &\xrightarrow{H^{\otimes n_c}} \frac{1}{\sqrt{2^{n_c}}} \sum_{k=0}^{2^{n_c}-1} |\vec{k}\rangle |v_j\rangle \\ &\xrightarrow{CU \text{ sequence}} \frac{1}{\sqrt{2^{n_c}}} \sum_{k=0}^{2^{n_c}-1} e^{2\pi i \theta_j k} |\vec{k}\rangle |v_j\rangle \\ &\xrightarrow{QFT^\dagger} |\tilde{\theta}_j\rangle |v_j\rangle \end{aligned}$$

where CU sequence is the controlled-U sequence in the sky blue part of Figure 1 and $\tilde{\theta}_j = \theta_j$ if θ_j can be perfectly represented in n_c bits; otherwise, $\tilde{\theta}_j \approx$ is an estimation of θ_j in a finite precision. In other words, the number of clock qubits, n_c, governs the precision of the estimated eigenvalue in QPE. To understand more details about QFT and QPE, we direct the interested reader to [53] and [71].

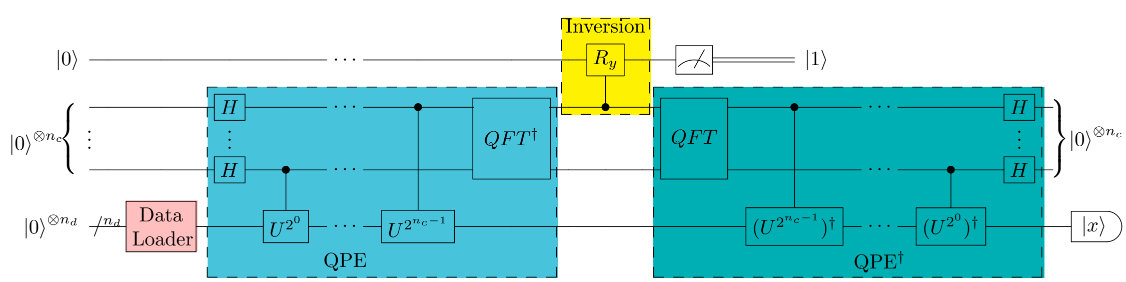

Fig. 1. HHL circuit. The unitary gates in quantum phase estimation (QPE) are $U = e^{itA}$ and $U^{2^j} = e^{i2^j tA}$ where $i^2 = -1$ and t is a scaling factor. The top qubit is referred to as the ancillary qubit, and it is the most significant qubit.

2.1.3 State Evolution in HHL. In general, the evolution of states in the HHL circuit is

$$
\begin{aligned}
|0\rangle\,|0\rangle^{\otimes n_c}\,|0^{n_d}\rangle &\xrightarrow{\text{Data loading}} |0\rangle\,|0\rangle^{\otimes n_c}\,|b\rangle \\
&\xrightarrow{\text{QPE}} \sum_{j=0}^{2^{n_d}-1} b_j\,|0\rangle\,|\tilde{\lambda}_j\rangle\,|v_j\rangle \\
&\xrightarrow[\text{Inversion}]{\text{Eigenvalue}} \sum_{j=0}^{2^{n_d}-1} b_j \left(\sqrt{1-\frac{C^2}{\tilde{\lambda}_j^2}}\,|0\rangle + \frac{C}{\tilde{\lambda}_j}\,|1\rangle\right)|\tilde{\lambda}_j\rangle\,|v_j\rangle \\
&\xrightarrow[\text{only keep } |1\rangle]{\text{Measure the ancillary}} D\sum_{j=0}^{2^{n_d}-1} \frac{b_j}{\tilde{\lambda}_j}\,|1\rangle\,|\tilde{\lambda}_j\rangle\,|v_j\rangle \\
&\xrightarrow{\text{QPE}^\dagger} D\,|1\rangle\,|0^{n_c}\rangle \sum_{j=0}^{2^{n_d}-1} \frac{b_j}{\tilde{\lambda}_j}\,|v_j\rangle \approx D\,|1\rangle\,|0^{n_c}\rangle\,|x\rangle
\end{aligned}
$$

where $\tilde{\lambda}_j \approx \lambda_j$ is the eigenvalue of A with a finite precision estimated in QPE, and C and D are both constant normalization factors

$$
D = \frac{C}{\sqrt{\sum_{j=0}^{2^{n_d}-1} C^2 \frac{|b_j|^2}{\tilde{\lambda}_j^2}}} \approx \frac{1}{\|A^{-1}\,|b\rangle\,\|_2}.
$$

Thus, the norm $\|A^{-1}\,|b\rangle\,\|_2$ can be estimated by the probability of measuring the ancillary qubit in state $|1\rangle$, i.e., $\|A^{-1}\,|b\rangle\,\|_2 \approx \sqrt{\Pr(\text{Measure 1 in ancillary})}$. This value can be obtained without extra cost as we need to run the circuit multiple times to get $|x\rangle$ or $\langle x|\,M\,|x\rangle$ for some observable M. The overall runtime complexity of HHL algorithm is $\tilde{O}(\log(N)s^2\kappa^2/\epsilon)$ where s is the sparsity of A, $\kappa = \|A\|\|A^{-1}\|$ is the condition number of A, and ϵ is the final additive error of the solution defined by the ideal state $|x\rangle$ and the result from HHL $|x_{HHL}\rangle$ through $\|\,|x\rangle - |x_{HHL}\rangle\,\| \le \epsilon$ [31].

2.1.4 Quantum-classical Data Exchange in HHL. There are two major input models for encoding both matrix A (or e^{itA}) and vector $|b\rangle$ into a quantum computer. One is the sparse-access model, used in the HHL algorithm [31]. Sparse-access model is a quantum version of classical sparse matrix computation, and we assume access to unitaries that calculate the index of the l^{th} non-zero element of the k^{th} row of a matrix A when given (k, l) as input. A different input model, now known as the quantum operator input model, is from Low and Chuang [46]. This method is based on the block-encoding of A to allow efficient access to entry values. Its circuit implementation can be found in [8, 9]. Meanwhile, this encoding scheme can also be achieved using quantum random access memory (QRAM) [25, 35, 36, 43]. It requires the complexity $O\left(\text{polylog}\left(N/\epsilon_{BE}\right)\right)$ for realizing an ϵ_{BE}-approximate block-encoding of $A \in \mathbb{C}^{N\times N}$ with QRAM [35].

Definition 2.2 (The block-encoding of a matrix). *The block-encoding of a matrix $A \in \mathbb{C}^{N\times N}$ is a unitary operator U such that*

$$
U = \begin{bmatrix} A/a & \cdot \\ \cdot & \cdot \end{bmatrix}
$$

where $a \ge \|A\|$ is a normalizing constant. In other words, U and A satisfies, for some constant a and n,

$$
a\left(\langle 0|^{\otimes n} \otimes I_N\right) U \left(I_N \otimes |0\rangle^{\otimes n}\right) = A
$$

Table 1. HHL circuit properties for four random examples

No. of Qubits n_d	n_c	CX gates	Depth	Total No. of Gates Before Fusion	After Fusion
4	6	116,535	248,084	325,189	70,804
5	7	1,111,178	2,373,842	3,106,244	665,921
6	8	9,335,345	19,969,964	26,117,061	5,557,777
7	9	78,420,632	167,816,254	219,386,270	46,631,320

where $I_N \in \mathbb{R}^{N\times N}$ is the identity matrix.

On the other hand, efficiency on reading $|x\rangle$ could be a potential threat to quantum speedup. The current state-of-the-art quantum state tomography algorithm is from Apeldoorn *et al.* [64]. For a state $|\psi\rangle = \sum_{j=0}^{N-1} \beta_j |\vec{j}\rangle \in \mathbb{C}^N$, with probability $1-\delta$, the pure-state tomography in [64] requires $O\left(\sqrt{N}/\sigma \cdot \log(N/\delta)\right)$ queries to the unitary oracle that prepares $|\psi\rangle$ from $|00\dots 0\rangle$ to output a vector $\vec{\beta}_{est} \in \mathbb{R}^N$ such that $\|\Re(\vec{\beta}) - \vec{\beta}_{est}\|_\infty \leq \sigma$. The same routine can be applied on $i|\psi\rangle$ to estimate the imaginary part.

2.2 Implementation of the Circuit Generation

In all experiments in this paper, the code for the HHL circuit generation comes from a Qiskit-based open-sourced package [66], which only produces the essential parts of the HHL circuit as colored in Figure 1. We made slight modifications to accommodate the changes in Qiskit 0.46. The state preparation for $|b\rangle$ uses the algorithm in [32] that decomposes an arbitrary isometry into the optimized number of single-qubit and CNOT gates, where isometry refers to the inner-product-preserving transformation that maps between two Hilbert spaces, i.e., the state preparation is a special case of isometries. For constructing the unitary operator e^{itA} in the QPE stage, the code directly accesses the entry data from the classical memory. In other words, the quantum memory structures that we discussed previously are not included in the circuits.

3 SIMULATOR AND RESOURCES ESTIMATION TOOL

The statevector simulator carries the simulations in the experiments, SV-Sim [39], in Northwest Quantum Circuit Simulation Environment (NWQSim) [38]. Compared to simulators in Aer from Qiskit [56] and qsim from Cirq [16], NWQSim provides specialized computation for a wide range of supported basis gates and architectures of CPUs and GPUs, such as gate fusion. In Table 1 and later in Section 5, we demonstrate that gate fusion strategy in NWQSim can reduce about 80% of gates in the circuits without sacrificing error rates. On the other hand, NWQSim utilizes a communication model called "PGAS-based SHMEM" that significantly reduces communication latency for intra-node CPUs/GPUs and inter-node CPU/GPU clusters. In this case, SV-Sim has an exceptional performance over other simulators in deep-circuit simulation [39]. Figure 2 shows the running time of the HHL circuit in the size of 11 qubits to 17 qubits on SV-Sim on four different GPUs.

The resources estimator in [6, 65] from Microsoft Azure Quantum establishes a systematical framework to access and model the resources necessary for implementing quantum algorithms on a user-specified fault-tolerant scenario. This tool enables detailed estimation of various computational resources, such as the number of physical qubits, the runtime, and other QEC-related properties to achieve a quantum advantage for certain applications. Specifically, the

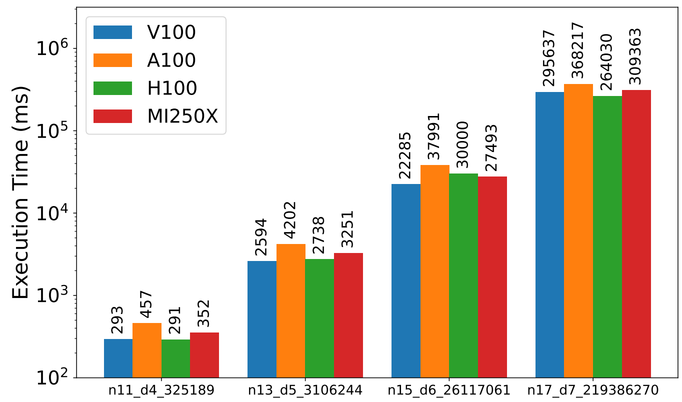

Fig. 2. NWQSim performance on different GPUs. The testing HHL circuits use randomly generated sparse matrices and random RHS vectors. The three numbers in the name of each testing circuit are the number of qubits in the circuit, the number of qubits for data loading, and the total number of gates in the circuit, respectively.

tool accepts a wide range of qubit and quantum error correction (QEC) code specifications and an error budget that allows different error rates to simulate a described fault-tolerant environment.

The tool is compatible with circuits generated from a high-level quantum computing language or package, including Qiskit and Q#. After a circuit is given, the input is compiled into Quantum Intermediate Representation through a unified processing program, and the estimator can examine the code and record qubit allocation, qubit release, gate operation, and measurement operation. Then, logical-level resources are estimated and used to compute the required physical-level resources further. The tool returns a thorough report on resources demanded to perform the given algorithm on fault-tolerant quantum computers, including the explanation and related mathematical equations of those estimates. A selected list of estimates is described in Section 4, and their values in conducted experiments are displayed in Section 5.

4 FACTORS OF INTEREST

As we focus on the linear system in scientific applications instead of random systems for benchmarking, we have less control over the specific values of matrix properties like condition numbers. Our interest is more on the number of clock qubits n_c in the HHL circuit, which controls the precision of estimated eigenvalues. The error in eigenvalue estimation affects the solution of the linear system through Eq. (4). From [53], to obtain an eigenvalue with 2^{-b} precision with at least $1 - p_{QPE,fail}$ success probability using QPE, we need

$$n_c = b + \left\lceil \log_2 \left(2 + \frac{1}{2p_{QPE,fail}} \right) \right\rceil.$$

In the Qiskit-based HHL implementation that we used [66], it is suggested that

$$n_c = \max\left(n_d + 1, \lceil \log_2(\kappa + 1) \rceil\right) + \mathbb{1}_{nv} \tag{6}$$

where $\mathbb{1}_{nv} = 1$ if the coefficient matrix has a negative eigenvalue, 0 otherwise. In this paper, we will adjust n_c to illustrate the influence of the QPE resources on the HHL circuit's total cost and the algorithm's precision in domain applications.

Table 2. Qubit parameter configurations

	Qubit Parameter Set	$(ns, 10^{-4})$	$(\mu s, 10^{-4})$
Operation Time	Measurement	100 ns	100 μs
	Single-qubit gate	50 ns	100 μs
	Two-qubit gate	50 ns	100 μs
	T gate	50 ns	100 μs
Error rate	Measurement	10^{-4}	10^{-4}
	Single-qubit gate	10^{-4}	10^{-4}
	Two-qubit gate	10^{-4}	10^{-4}
	T gate	10^{-4}	10^{-6}

Note that, without circuit optimization, the increase of n_c will exponentially increase the gate counts in HHL circuits. Recall the HHL circuit in Figure 1, an extra clock qubit leads an extra controlled $U^{2^{n_c-1}}$ and an extra controlled inverse $U^{2^{n_c-1}}$ in HHL circuit, where $U = e^{itA}$ and A is the coefficient matrix in a linear system. Generally, we should not explicitly compute the matrix $U^{2^{n_c-1}}$, but apply gate U for 2^{n_c-1} times in the circuit. Then, the QPE part of the circuit contains $\sum_{j=0}^{n_c-1} 2^j = 2^{n_c} - 1$ number of U, and when there are $n_c + 1$ number of clock qubits, an extra 2^{n_c} number of U is added into the QPE, which almost double the number of gates U in QPE. The same situation happens on the inverse QPE part of the HHL circuit.

When discussing resource estimation under a fault-tolerant setting, our primary concerns are the estimated runtime, the number of physical qubits, and extra resources required from the QEC code. We adopt a distance-7 surface code that encodes 98 physical qubits into a single logical qubit. The theoretical logical qubit error rate is 3×10^{-10}, and the error correction threshold is 0.01. Azure Quantum resource estimator provides several qubit parameter sets to simulate different qubit properties. The preset qubit settings we used in this paper are $(ns, 10^{-4})$ and $(\mu s, 10^{-4})$ from [6], where the former one is close to the specifications of superconducting transmon qubits or spin qubits, and the later one is more relevant for trapped-ion qubits [6]. A list of detailed configurations of qubit parameter set $(ns, 10^{-4})$ and $(\mu s, 10^{-4})$ is in Table 2. We enforce 2-D nearest-neighbor connectivity of the qubits to simulate the connectivity constraint on real quantum computers. So we also demonstrate the changes of some factors before this constraint is enforced ("pre-layout") and after this constraint is enforced ("after layout").

Another important tunable parameter is the overall allowed errors for the algorithm, namely error budget. Its parameter value is equally divided into three parts:

- logical error probability: the probability of at least one logical error
- T-distillation error probability: the probability of at least one faulty T-distillation
- rotation synthesis error probability: the probability of at least one failed rotation synthesis.

There are also specific breakdowns in the resource required by QEC that are of interest [6, 65]. We list them in Table 3.

5 SCIENTIFIC APPLICATIONS AND EVALUATION

This section examines the utilization of the HHL algorithm in the fields of power grids and heat transfer. We evaluate the performance of HHL in terms of solution accuracy, resource cost, and influence on convergence speed for applicable problems.

In addition to the hardware specifications in Section 3, all resource estimator jobs are run on the Azure Quantum cloud server. Due to the limitation on the cloud service usage, we cannot examine some of the deepest circuits in this

Table 3. Factors of Interest in QEC

Factors of Interest	Description
Number of logical qubits pre- and after layout	Under the nearest-neighbor constraint, extra logical qubits could be required to satisfy the connectivity needed in the algorithm (circuit); the relation is $n_{\text{after}} = 2n_{\text{alg}} + \lceil \sqrt{8n_{\text{alg}}} \rceil + 1$ where n_{alg} is the number of logical qubits pre-layout and n_{after} is the number of qubits after layout
Number of physical qubits for the algorithm	The product of the number of logical qubits after layout and the number of physical qubits needed to encode one logical qubit
Number of physical qubits for T factories	T factories produce T states to implement non-Clifford operations in a circuit
Number of physical qubits	The sum of the number of physical qubits for the algorithm and the number of physical qubits for T factories
Number of T states	The estimator requires one T state for each of the T gates in a circuit, four T states for each of the CCZ and $CCiX$ gates, and 18 T states for each of the arbitrary single-qubit rotation gates
Number of T factories	Determined from algorithm runtime, T state per T factory, the number of T states, and T factor duration through the equation $\left\lceil \frac{T\text{ state}\cdot T\text{ factor duration}}{T\text{ state per }T\text{ factory}\cdot\text{algorithm runtime}} \right\rceil$
Number of logical cycles for the algorithm	The logical depth of the algorithm
Min. logical qubit error rate required to run the algorithm within the error budget	$\frac{\text{logical error probability}}{\text{Number of logical qubits}\cdot\text{Number of logical cycles}}$
Min. T state error rate required for distilled T states	$\frac{T\text{ distillation error probability}}{\text{Total number of }T\text{ states}}$

section with the resources estimator, and all evaluated circuits are transpired with respect to a given basis gate set from the estimator using the transpiler in Qiskit. The optimization level of the transpiler is set to level 2. The Qiskit version is 0.46, and the Azure Quantum version is 0.30.0.

5.1 Power Grid

The use of quantum algorithms has drawn much attention in recent research on power system applications, especially the areas where quantum linear system solver can deploy, including power flow, contingency analysis, state estimation, and transient simulation [12, 13, 23, 26, 34, 73]. The specific problem type we illustrated in this section is an Alternating Current (AC) power flow problem.

The power flow equations are essential to analyzing the steady-state behavior of power systems by describing the relationship between bus voltages (magnitude and phase angles), currents, and power injections in a power system. The basic power flow equations are:

$$P_k = \sum_{j=1}^{n} \left(|V_k||V_j|\mathrm{Re}(Y_{kj}^*)\cos(\theta_{kj}) + |V_k||V_j|\mathrm{Im}(Y_{kj}^*)\sin(\theta_{kj}) \right) \quad (7)$$

$$Q_k = \sum_{j=1}^{n} \left(|V_k||V_j|\mathrm{Re}(Y_{kj}^*)\sin(\theta_{kj}) - |V_k||V_j|\mathrm{Im}(Y_{kj}^*)\cos(\theta_{kj}) \right) \quad (8)$$

Where:

- P_k: Real power injection at bus k.
- Q_k: Reactive power injection at bus k.
- $|V_k|$: Voltage magnitude at bus k.
- θ_{kj}: Phase angle difference between bus k and bus j.
- Y_{kj}: Admittance between bus k and bus j.

For a B buses and G generators power flow problem, there are $2(B-1)-(G-1)$ unknowns representing voltage magnitudes, $|V_k|$, and phase angles, θ_k , for load buses and voltage phase angles for generator buses. With the knowledge of the admittance matrix of the system that represents the nodal admittance of the buses, we can use Newton-Raphson (N-R) method to solve power flow equation iteratively: after an initial guess for the voltages at all buses, in each N-R iteration, we solve

$$\begin{bmatrix} \frac{\partial \Delta \vec{P}}{\partial \vec{\theta}} & \frac{\partial \Delta \vec{P}}{\partial |\vec{V}|} \\ \frac{\partial \Delta \vec{Q}}{\partial \vec{\theta}} & \frac{\partial \Delta \vec{Q}}{\partial |\vec{V}|} \end{bmatrix} \begin{bmatrix} \Delta \vec{\theta} \\ \Delta |\vec{V}| \end{bmatrix} = - \begin{bmatrix} \Delta \vec{P} \\ \Delta \vec{Q} \end{bmatrix} \tag{9}$$

where ΔP_k and ΔQ_k are computed using the admittance matrix, nodal power balance equation, and mismatch equations with the data from the last iteration or initial guess. Then, $\vec{\theta}$ and $|\vec{V}|$ are updated by $\Delta\vec{\theta}$ and $\Delta|\vec{V}|$, respectively. The algorithm is set to convergent when $\|\Delta\vec{P}\|$ and $\|\Delta\vec{Q}\|$ are smaller than a convergence tolerance.

It is worth noting that, while HHL can solve Eq. (9) for the normalized solution state $[\Delta\vec{\theta}^T \ \Delta|\vec{V}|^T]^T$ in a limited precision, the un-normalized vector could have a smaller norm than the precision of HHL. Thus, the final precision of voltage magnitude and phase angles is much higher than the precision used in HHL. This situation is similar to iterative refinement in semi-definite optimization in [48].

5.1.1 Settings of the Numerical Experiments. The test case is the four buses and two generators problem in [28, p. 377], coded in a MATLAB package called MATPOWER [74]. Based on the framework built in [72], we incorporate HHL circuits and quantum simulators into the solving process in MATPOWER. The linear systems of our interest are all 5×5 systems but not Hermitian. So, the actual inputted system is first expanded to 8×8 so the size of the RHS vector is the power of 2, then enlarged to 16×16 following Eq. (1). So, we eventually use 4 qubits to encode the vector $\vec{b}$. This process is illustrated in Figure 3(a).

The default value of n_c set by [53] using Eq. (6) is 6. To demonstrate how the precision of eigenvalues affects an iterative algorithm, we select n_c from 4 to 7. With 4 clock qubits in QPE and an ancillary qubit required by the HHL algorithm, the number of qubits in each HHL circuit ranges from 9 to 12. The N-R method converges when

$$\left\| \begin{bmatrix} \Delta\vec{P} \\ \Delta\vec{Q} \end{bmatrix} \right\|_\infty < 10^{-8}.$$

However, because the linear system formed in an N-R iteration depends on the solution from the previous N-R iteration, the linear systems at Iteration j with different n_c will differ. Our comparison focuses on the convergence speed and the final solution at the convergence instead of errors at each iteration across different n_c.

5.1.2 Performance Evaluations. The sparsity of all tested coefficient matrices is 84.375% after the expansion, with condition numbers in the range of $[5.950, 5.970]$. The minimums of the magnitude of eigenvalues are in the range of $[12.263, 12.506]$, and the maximums are $[73.209, 74.659]$. Figure 3(b) and (c) provide illustrative evidence of the use of a less precise linear solver in the iterative method like the N-R method. Although the N-R method with an HHL subroutine converges slower than a classical linear solver in MATLAB, all methods converge under the same criteria and obtain a similar solution. A trade-off between convergence speed and complexity of linear system solving exists in our experiments.

On the other hand, if we compare the values of normalized error $\| \, |x\rangle - |x\rangle_{HHL} \, \|_2$, when $n_c = 4, 5, 6, 7$, using more clock qubits indeed leads to lower error from the HHL algorithm itself. However, increasing n_c does not implies less error on the solution vectors, $\vec{x}_{HHL}$, nor faster converge by looking at the values of $\|\vec{x} - \vec{x}_{HHL}\|_2$ and $[\Delta\vec{P}^T \ \Delta\vec{Q}^T]^T$ in Figure 3. The HHL algorithm with $n_c = 5$ gives the fastest convergence, which is smaller than the default value, 6, from Eq. 6.

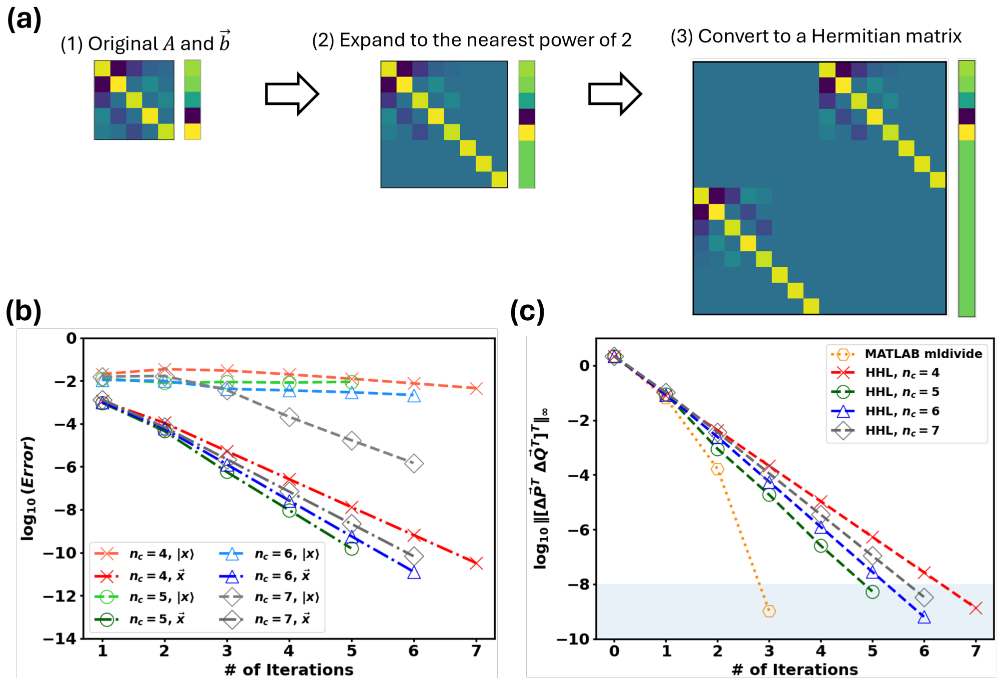

Fig. 3. Matrix expansion and error plot for the experiments of the power flow problem. (a) Expanding the coefficient matrices and the RHS vector in Eq. (9) to fulfill the requirements of the HHL algorithm. (b) The errors of the solution state in log base 10 (labeled "$|x\rangle$"), $\| \,|x\rangle - |x\rangle_{HHL} \,\|_2$, and the errors of the solutions in log base 10 (labeled "$\vec{x}$"), $\|\vec{x} - \vec{x}_{HHL}\|_2$, in each N-R iteration with different numbers of n_c. The symbols A, $\vec{b}$, $\vec{x}$, and $|x\rangle$ refer to the corresponding part in Eq. (9). (c) The infinity norms of $[\Delta\vec{P}^T\ \Delta\vec{Q}^T]^T$ in log base 10, i.e., the value governs the convergence of the N-R method. Iteration 0 represents the norm from the initial guess, and the gray-shaded area is where the convergence criteria are satisfied.

Table 4. Depths and gate counts of HHL circuits for power flow problems at Iteration 1

n_d	n_c	Depth	# of gates	# of 2-qubit gates	# of gates after fusion	Reduction from fusion
4	4	65,824	86,262	30,651	18,060	79.06%
4	5	135,986	178,180	63,315	37,283	79.08%
4	6	276,308	361,980	128,631	75,717	79.08%
4	7	556,950	729,534	259,247	152,570	79.09%

5.1.3 Gate Counts and Depths of HHL Circuits. Because the circuits from later iterations are in a similar resource demand, we only look at the circuits in the first iteration. The depths and gate counts of HHL circuits are the same across N-R iterations when n_c is fixed. While HHL with $n_c = 5, 6, 7$ gives similar convergence speed and accuracy, the required resources to run the circuits exponentially increase as n_c increases based on Table 4. On the other hand, although gate fusion employed in NWQSim does not mitigate these exponential trends, it maintains a constant proportional performance across various HHL circuits: a 79% reduction of gate counts on all tested circuits regardless of the value of n_c.

5.1.4 Resources Estimation in a Fault-Tolerant Scenario . Encoded by the surface code described in Section 4 along with a nearest-neighbor connectivity constraint, we estimate the runtime of HHL circuits by Azure Quantum resource

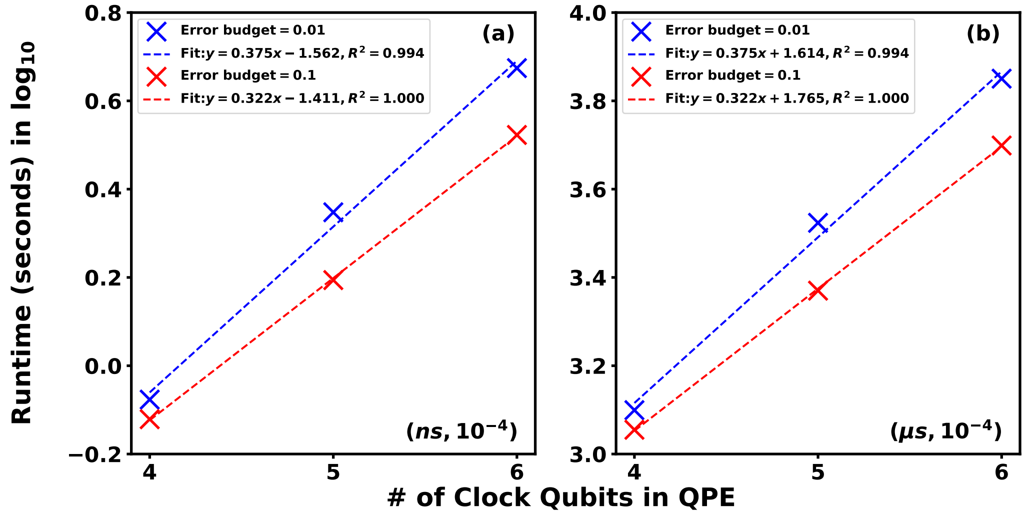

Fig. 4. The runtime in seconds as a function of the number of clock qubits in QPE under the qubit parameter set (a) $(ns, 10^{-4})$ and (b) $(us, 10^{-4})$. The estimated circuits are HHL circuits for power flow problems.

estimator and summarize the data in Figure 4. A strong and consistent linear correlation between the number of clock qubits in QPE, n_c, and the runtime in log base 10 is displayed across qubit parameter sets and error budgets. Every extra clock qubit brings $10^{0.322} \approx 2.099$ times longer runtime when the error budget is 0.1 and $10^{0.375} \approx 2.371$ times longer when the error budget is 0.01. This multiplier shows an increasing trend when the error budget decreases. The similar correlations are also demonstrated in Figure 5(a) and (b) when we further investigate how n_c affects the number of logical cycles for the circuit and the number of T states. Generally, the exponential dependencies of runtime, number of logical cycles, and number of T states on n_c match the relation between the number of gates in HHL circuits and n_c. Note that the slopes of the fitted line in Figure 5(a) and (b) are not sensitive to error budgets, different from the behavior in Figure 4. Error budgets affect the constant multiplier of the growth of logical cycles and the number of T states more.

Table 5 summarizes the other factors of our interest. Those factors have the same values in $(ns, 10^{-4})$ and $(\mu s, 10^{-4})$ settings. Note that there is a dramatic fall in the number of physical qubits when the error budget is 0.01 and n_c raises from 4 to 5. Combining with Figure 5(c) and (d), this reduction comes from a large drop in the number of physical qubits spent on T factories, a dominant demand on physical qubits instead of the quantum algorithm itself. The circuit requires 15 T factories when the error budget is 0.01 and $n_c = 4$, but this number is reduced to 12 when $n_c = 5$. Recall the definition of the number of T factories in Table 3, based on the fitted coefficients in Figure 4 and 5, we can see while the increase of n_c from 4 to 5 leads to $10^{0.322}$ times more T states, the runtime becomes $10^{0.375}$ times larger. Since T factory duration and T states per factory are kept constant, the faster-growing runtime reduces the number of T factories required, thus decreasing the overall number of physical qubits required. This phenomenon does not occur when the error budget is 0.1 because the growth of runtime and T-state count are at the same speed.

5.2 Heat transfer

Linear solvers are deeply embedded in differential equation solving through numerical methods such as the finite difference method. Such methods discretize the domain of the problems into grids, and the dimension of the formed

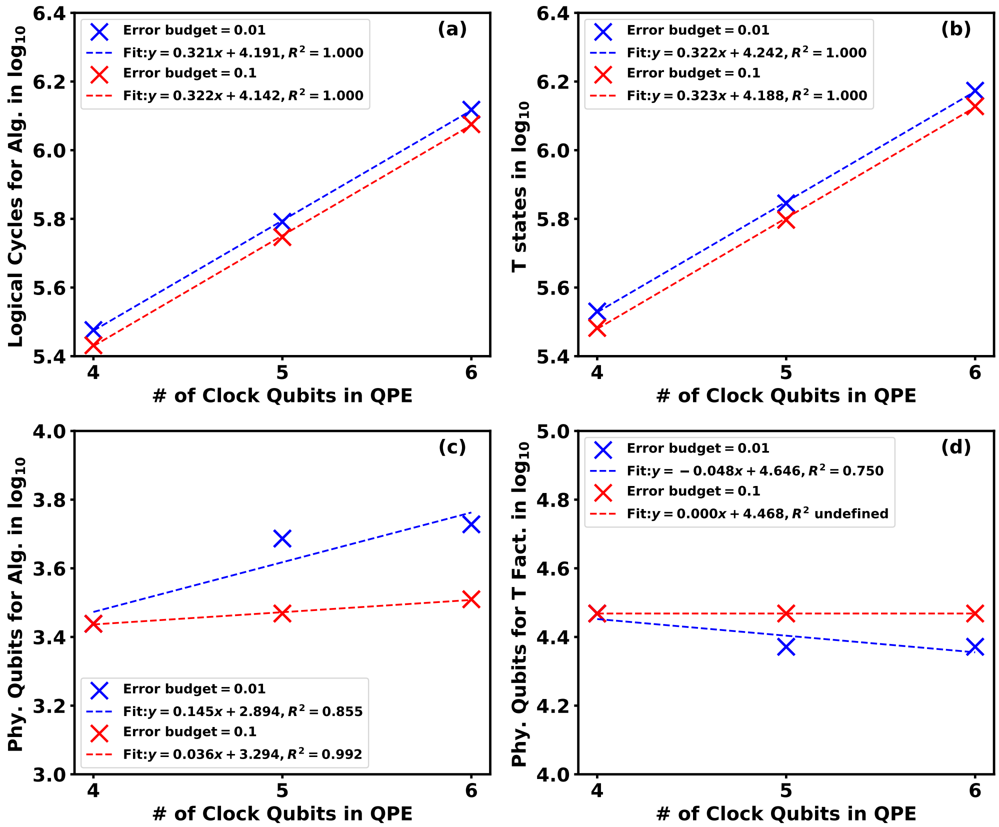

Fig. 5. The number of (a) logical cycles for the algorithm, (b) T states, (c) physical qubits for the algorithm after layout, and (d) physical qubits for the T factories as functions of the number of clock qubits in QPE, respectively. The estimated circuits are HHL circuits for power flow problems. Both qubit parameter sets $(ns, 10^{-4})$ and $(us, 10^{-4})$ have the same values under the same error budget for all four factors in the plots.

Table 5. Factors of interest for fault-tolerant HHL circuits in power flow problems

Error budget	n_c	Physical qubits after layout	Logical qubits pre- and after layout	Min. logical qubit error rate	Min. T state error rate
0.01	4	32,144	9 to 28	3.977×10^{-10}	9.831×10^{-9}
	5	28,380	10 to 30	1.797×10^{-10}	4.762×10^{-9}
	6	28,866	11 to 33	7.700×10^{-11}	2.236×10^{-9}
0.1	4	32,144	9 to 28	4.406×10^{-9}	1.098×10^{-7}
	5	32,340	10 to 30	1.990×10^{-9}	5.319×10^{-8}
	6	32,634	11 to 33	8.487×10^{-10}	2.483×10^{-8}

linear system scales as the size of discretization. The number of grid points scales polynomially with system size, while the demands for solving such differential equations (DEs) are ubiquitous in science and engineering. Due to the exponential speedup in problem dimension, the combination of quantum linear solvers and these numerical methods has become an attractive direction [15, 41, 51, 60].

Table 6. Depths and gate counts of HHL circuits for heat transfer problems

Dim.	n_d	n_c	Depth	# of gates	#of 2-qubit gates	# of gates after fusion	Reduction from fusion
9×9	4	3	30742	40290	14315	8445	79.04%
9×9	4	4	65824	86262	30651	18061	79.06%
9×9	4	5	135986	178180	63315	37284	79.08%
9×9	4	6	276308	361980	128631	75718	79.08%
25×25	5	3	133966	175253	62546	37147	78.80%
25×25	5	4	287134	375643	134046	79547	78.82%
25×25	5	5	593948	777069	277230	164338	78.85%

5.2.1 Settings of the Numerical Experiments. In this section, we examine the two-dimensional (2-D) heat diffusion equation in [42]

$$\frac{\partial T}{\partial t} = D\nabla^2 T + F \tag{10}$$

where T represents the temperature at a given 2-D point and time, D is the heat transfer coefficient, and F is the forcing term consisting of arbitrary boundary and initial conditions. Eq. (10) is a linear partial differential equation. We discretize Eq. (10) in space and time into a system of ordinary differential equations using the finite difference method,

$$AT = F, \tag{11}$$

where A is the resulting coefficient matrix. Take the square lattices with a lateral size of 3 grid points and 5 grid points, the resulting dimension of A is 9×9 and 25×25, respectively. Such configurations require 4 qubits and 5 qubits to represent the RHS vectors (F term in Eq. (11)) in both linear systems, respectively. Let $A^{(heat,l)}$ be the coefficient matrix generated from l number of grid points, the entry values are

$$A_{pq}^{(heat,l)} = \begin{cases} 1 + 4r, & p = q \\ -r & p = q + 1 \text{ or } p = q - 1 \text{ or } \text{ or } p = q - l \text{ or } p = q + l \\ 0, & \text{otherwise} \end{cases}$$

where p and q denotes the index of the entries of A and r is 0.00016 in 3-point case and 0.00064 in 5-point case.

5.2.2 Performance and Resources Evaluations. The coefficient matrices are Hermitian by design, so we only need to expand the dimension to the nearest power of 2, i.e., 16 and 32. After dimension expansion, the coefficient matrices have sparsity 82.813% and 88.281%, respectively. Both matrices have condition number 1, and all of their eigenvalues are around 1.

When $n_d = 4$, gate counts in Table 6 and 4 have almost the same numbers of circuit depths and gate counts. However, if we compare across different n_d in Table 6, significant increases appear in depths and all gate counts. This situation reflects one of Aaronson's concerns in [1] about the efficiency and the cost of data reading in quantum linear solvers. Furthermore, similar to the scenario in Section 5.1, the incremental of n_c, despite being very costly, has a limited contribution towards reducing errors, as shown in Figure 6.

5.2.3 Resources Estimation in a Fault-Tolerant Scenario. Most of the observations from Fig 7 and 8 and Table 7 for both problem sizes are isometric to the findings in Section 5.1.4, including the numerical values of the fitted-line coefficients related to runtime, logical cycles and the number of T states. The significant influence brought by deeper data loading modules for the 5-point problem is parallel shifts on longer runtime, more logical cycles, more T states, and more

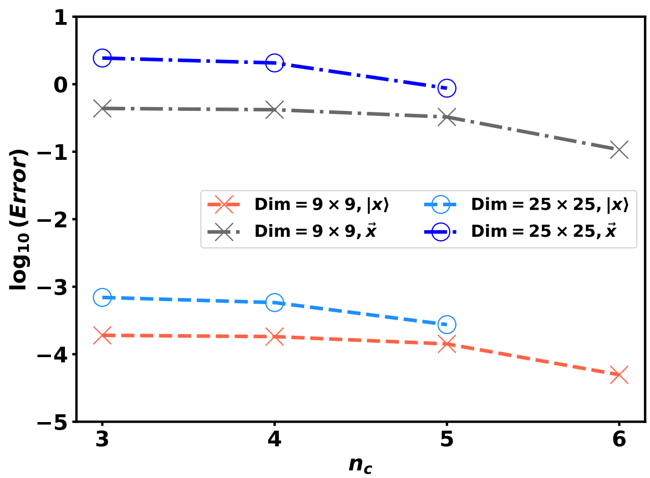

Fig. 6. The errors of the solution states in log base 10 (labeled as "$|x\rangle$"), $\| \, |x\rangle - |x\rangle_{HHL} \, \|_2$, and the errors of the solution vectors in log base 10 (labeled as "$\vec{x}$"), $\|\vec{x} - \vec{x}_{HHL}\|_2$, are presented as functions of n_c for two different numbers of grid points. The symbols $\vec{x}$ and $|x\rangle$ respectively refer to the solution vector and the normalized solution vector in the linear systems constructed by applying the finite difference method to Eq. (10).

strict requirements on logical qubit error rate and T state error rate. More data-loading qubits do not affect the growth speed of the logical cycle and the number of T states. Due to the limitation of computational time in Azure Quantum cloud service, we cannot collect more data points to understand this correlation better. However, from a theoretical perspective, this is expected because the QPE costs of HHL circuits are the same with the same n_c in the power flow and heat transfer problems.

Table 7. Factors of interest for fault-tolerant HHL circuits in heat transfer problems

Error budget	(n_d, n_c)	Physical qubits after layout	Logical qubits pre- and after layout	Min. logical qubit error rate	Min. T state error rate
0.01	(4,3)	31850	8 to 25	1.01×10^{-9}	2.22×10^{-8}
	(4,4)	32144	9 to 28	3.97×10^{-10}	9.81×10^{-9}
	(4,5)	28380	10 to 30	1.80×10^{-10}	4.77×10^{-9}
	(4,6)	28866	11 to 33	7.69×10^{-11}	2.23×10^{-9}
	(5,3)	28056	9 to 28	2.05×10^{-10}	5.11×10^{-9}
	(5,4)	28380	10 to 30	8.53×10^{-11}	2.27×10^{-9}
0.1	(4,3)	13450	8 to 25	1.12×10^{-8}	2.50×10^{-7}
	(4,4)	32144	9 to 28	4.40×10^{-9}	1.10×10^{-7}
	(4,5)	32340	10 to 30	2.00×10^{-9}	5.33×10^{-8}
	(4,6)	32634	11 to 33	8.47×10^{-10}	2.48×10^{-8}
	(5,3)	32144	9 to 28	2.27×10^{-9}	5.70×10^{-8}
	(5,4)	32340	10 to 30	9.39×10^{-10}	2.52×10^{-8}

6 DISCUSSION

The paper evaluates and analyzes the performance and resources required for the HHL algorithm in various scientific and engineering problems. There are still multiple points we need to address in future works. The foremost limitation

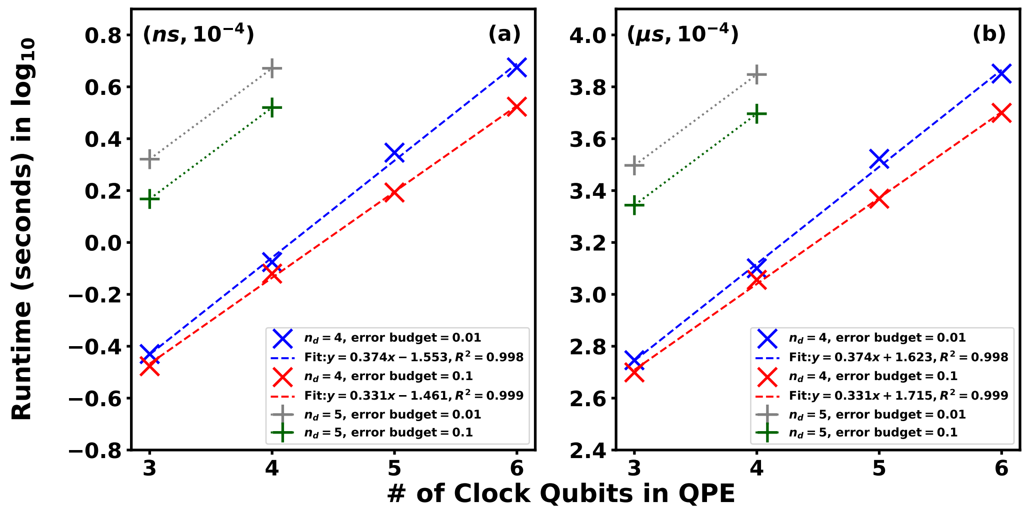

Fig. 7. The runtime in seconds as a function of the number of clock qubits in QPE under the qubit parameter set (a) $(ns, 10^{-4})$ and (b) $(us, 10^{-4})$. The estimated circuits are HHL circuits for the heat transfer problem.

in this work is the data loading module in the HHL circuit generation. While the data loading algorithm in [66] can encode an arbitrary vector into a quantum circuit, the circuit depth of this module is exponential to the number of qubits. Thus, this first part of the circuit severely damages the potential quantum speedup from HHL. We mitigate this drawback by comparing the outcomes from problems of different sizes to isolate the influence of the data loading module. An important future direction is incorporating an efficient data loading scheme into our analysis framework, like block encoding in [10]. A different data loading method could have a different precision, so it is necessary to investigate how data loading precision and condition number of coefficient matrices collectively affect the solution accuracy. This future direction illustrates the second drawback of this study. That is, our tested coefficient matrices are all well-conditioned. Because our experiments do not utilize randomly generated test cases, we have less control over the matrix properties, including condition number and sparsity. A potential source of ill-conditioned test cases is the methods that naturally have ill-conditioned matrices, such as the Newton systems produced by the interior-point method in optimization problems [48]. Thus, to solve those systems, a variant of the HHL algorithm in [17] accompanied by the sparse approximate inverse preconditioner is in our outlook. Limited by single-job running time in the Azure Quantum cloud server, we cannot process large HHL circuits. So, the number of data points in each plot in Section 5 is relatively tiny. This is why we only discuss the correlations whose coefficients of determination are almost 1. In future studies, we will dismantle the whole HHL circuit into different modules and evaluate the resource cost separately.

Some additional research can be conducted to further enhance our understanding of the application of quantum algorithms in scientific problems. An important direction is understanding the implication of various noise models on the HHL algorithm. We plan to conduct those experiments with the high-precision noise simulator in [40]. We can also include the quantum algorithms that address similar scientific applications into our resources analysis framework, such as the ordinary differential equation solvers in [3].

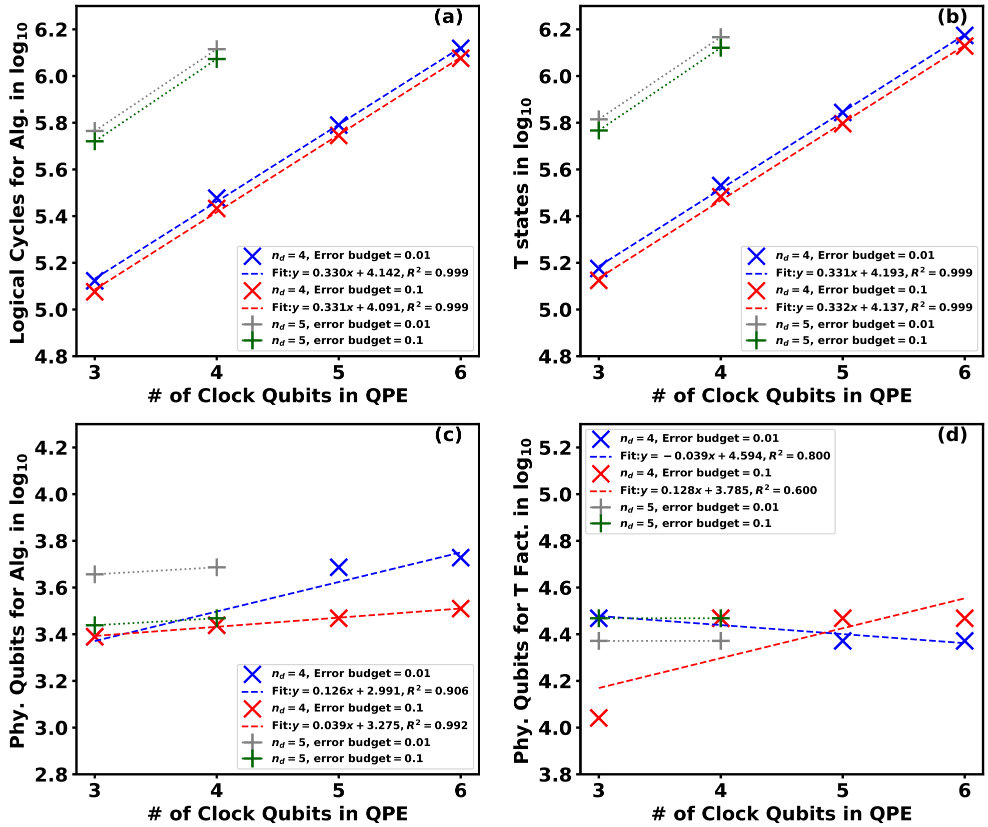

Fig. 8. The number of (a) logical cycles for the algorithm, (b) T states, (c) physical qubits for the algorithm after layout, and (d) physical qubits for the T factories as functions of the number of clock qubits in QPE, respectively. The estimated circuits are HHL circuits for the heat transfer problem. Both qubit parameter sets $(ns, 10^{-4})$ and $(us, 10^{-4})$ have the same values under the same error budget for all four factors in the plots.

7 CONCLUSION

In this paper, we investigate the practical applications and scalability of the HHL algorithm in solving quantum linear systems associated with scientific problems like power grids and heat transfer problems. Through the NWQSim package on high-performance computing platforms, we highlight the benefits of the utilization of low-precision QPE in HHL for both iterative and non-iterative methods in practice: low-precision QPE can exponentially reduce the gate counts and circuit depth in an HHL circuit, while keeping the same solution accuracy in iterative methods like Newton-Raphson method and maintain a similar level of accuracy in a non-iterative method like finite difference method.

Furthermore, with Azure Quantum resources estimator, we evaluate the resource requirements of HHL circuits in our experiments under two settings that simulate superconducting and trapped-ion qubits. The correlations between QEC-related criteria and the inputted HHL circuits have been thoroughly studied. The runtime, number of logical cycles, and number of T states have exponential decencies on the number of clock qubits in QPE. However, this relation is not necessarily inherited by the number of physical qubits demanded. If the runtime growth is faster than the required

T states, the circuit needs fewer T factories and fewer physical qubits to prepare T factories. Since the growth of runtime is sensitive to error budget, which is the allowance on the error occurrence on logical qubits, T distillation, and rotation synthesis, it is possible to reduce the physical qubit requirement if a low error budget is achievable on early fault-tolerant quantum devices. In other words, less but higher-fidelity logical qubits are as capable as more but lower-fidelity logical qubits in HHL circuits.

Our study provides pivotal insights into the operational requirements of quantum linear system algorithms, paving the way for further empirical studies. We propose future research on the applications of quantum linear system solvers and iterative refinement on high-fidelity quantum computers for small-scale experiments. For large-scale experiments, we suggest using noise-modelled simulators on high-performance platforms. In the context of QEC and early fault-tolerant quantum computing, we believe it is crucial to focus on controlling the resource cost of T factories by considering runtime and error budget. These research directions hold promise for bridging the gap between theoretical potential and practical usability in quantum computing.

ACKNOWLEDGMENTS

This material is based upon work supported by the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers, Quantum Science Center (QSC). X. Li was supported by the Co-design Center for Quantum Advantage (C2QA) under contract number DE-SC0012704 (PNNL FWP 76274). This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231. The Pacific Northwest National Laboratory is operated by Battelle for the U.S. Department of Energy under contract DE-AC05-76RL01830.

REFERENCES

[1] Scott Aaronson. 2015. Read the fine print. *Nature Physics* 11, 4 (April 2015), 291–293. https://doi.org/10.1038/nphys3272

[2] Andris Ambainis. 2012. Variable time amplitude amplification and quantum algorithms for linear algebra problems. In *STACS'12 (29th Symposium on Theoretical Aspects of Computer Science)*, Thomas Wilke Christoph Dürr (Ed.), Vol. 14. LIPIcs, Paris, France, 636–647. https://hal.science/hal-00678197

[3] Dong An, Andrew M Childs, and Lin Lin. 2023. Quantum algorithm for linear non-unitary dynamics with near-optimal dependence on all parameters. *arXiv preprint arXiv:2312.03916* (2023).

[4] Dong An and Lin Lin. 2022. Quantum Linear System Solver Based on Time-optimal Adiabatic Quantum Computing and Quantum Approximate Optimization Algorithm. *ACM Transactions on Quantum Computing* 3, 2, Article 5 (mar 2022), 28 pages. https://doi.org/10.1145/3498331

[5] Brandon Augustino, Giacomo Nannicini, Tamás Terlaky, and Luis F Zuluaga. 2023. Quantum interior point methods for semidefinite optimization. *Quantum* 7 (2023), 1110.

[6] Michael E Beverland, Prakash Murali, Matthias Troyer, Krysta M Svore, Torsten Hoeffler, Vadym Kliuchnikov, Guang Hao Low, Mathias Soeken, Aarthi Sundaram, and Alexander Vaschillo. 2022. Assessing requirements to scale to practical quantum advantage. *arXiv preprint arXiv:2211.07629* (2022).

[7] Dolev Bluvstein, Simon J Evered, Alexandra A Geim, Sophie H Li, Hengyun Zhou, Tom Manovitz, Sepehr Ebadi, Madelyn Cain, Marcin Kalinowski, Dominik Hangleiter, et al. 2024. Logical quantum processor based on reconfigurable atom arrays. *Nature* 626, 7997 (2024), 58–65.

[8] Daan Camps, Lin Lin, Roel Van Beeumen, and Chao Yang. 2024. Explicit quantum circuits for block encodings of certain sparse matrices. *SIAM J. Matrix Anal. Appl.* 45, 1 (2024), 801–827.

[9] Daan Camps and Roel Van Beeumen. 2022. Fable: Fast approximate quantum circuits for block-encodings. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 104–113.

[10] Pablo AM Casares and Miguel Angel Martin-Delgado. 2020. A quantum interior-point predictor–corrector algorithm for linear programming. *Journal of physics A: Mathematical and Theoretical* 53, 44 (2020), 445305. https://doi.org/10.1088/1751-8121/abb439

[11] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. 2018. The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation. *arXiv preprint arXiv:1804.01973* (2018). https://doi.org/10.4230/LIPIcs.ICALP.2019.33

[12] Yousu Chen, Zhenyu Huang, Shuangshuang Jin, and Ang Li. 2022. Computing for power system operation and planning: Then, now, and the future. *iEnergy* 1, 3 (2022), 315–324.

[13] Yousu Chen, Samuel Stein, Ang Li, and Zhenyu Henry Huang. 2023. Is It Coming Soon to Power Systems: Quantum Computing and Its Early Exploration. In *2023 IEEE Power & Energy Society General Meeting (PESGM)*. IEEE, 1–5.

[14] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. 2017. Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision. *SIAM J. Comput.* 46, 6 (2017), 1920–1950. https://doi.org/10.1137/16M1087072

[15] Andrew M Childs, Jin-Peng Liu, and Aaron Ostrander. 2021. High-precision quantum algorithms for partial differential equations. *Quantum* 5 (2021), 574.

[16] Cirq Developers. 2023. Cirq. https://doi.org/10.5281/ZENODO.4062499

[17] B David Clader, Bryan C Jacobs, and Chad R Sprouse. 2013. Preconditioned quantum linear system algorithm. *Physical review letters* 110, 25 (2013), 250504.

[18] Pedro CS Costa, Dong An, Yuval R Sanders, Yuan Su, Ryan Babbush, and Dominic W Berry. 2022. Optimal scaling quantum linear-systems solver via discrete adiabatic theorem. *PRX Quantum* 3, 4 (2022), 040303. https://doi.org/10.1103/PRXQuantum.3.040303

[19] MP da Silva, C Ryan-Anderson, JM Bello-Rivas, A Chernoguzov, JM Dreiling, C Foltz, JP Gaebler, TM Gatterman, D Hayes, N Hewitt, et al. 2024. Demonstration of logical qubits and repeated error correction with better-than-physical error rates. *arXiv preprint arXiv:2404.02280* (2024).

[20] David Deutsch and Richard Jozsa. 1992. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439, 1907 (1992), 553–558. https://doi.org/10.1098/rspa.1992.0167

[21] Bojia Duan, Jiabin Yuan, Chao-Hua Yu, Jianbang Huang, and Chang-Yu Hsieh. 2020. A survey on HHL algorithm: From theory to application in quantum machine learning. *Physics Letters A* 384, 24 (Aug. 2020), 126595. https://doi.org/10.1016/j.physleta.2020.126595

[22] Dmitry Fedorov, Matthew Otten, Byeol Kang, Anouar Benali, Salman Habib, Stephen Gray, and Yuri Alexeev. 2022. Quantum Resource Estimation for Quantum Chemistry Algorithms. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 859–861.

[23] Fei Feng, Yifan Zhou, and Peng Zhang. 2021. Quantum power flow. *IEEE Transactions on Power Systems* 36, 4 (2021), 3810–3812.

[24] Craig Gidney and Martin Ekerå. 2021. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum* 5 (2021), 433.

[25] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. 2008. Architectures for a quantum random access memory. *Physical Review A* 78, 5 (2008), 052310.

[26] Saeed Golestan, MR Habibi, SY Mousazadeh Mousavi, Josep M Guerrero, and Juan C Vasquez. 2023. Quantum computation in power systems: An overview of recent advances. *Energy Reports* 9 (2023), 584–596.

[27] Google Quantum AI. 2023. Suppressing quantum errors by scaling a surface code logical qubit. *Nature* 614, 7949 (2023), 676–681. https://doi.org/10.1038/s41586-022-05434-1

[28] John Grainger and William Stevenson. 1994. *Power System Analysis*. McGraw-Hill Professional, New York, NY.

[29] Markus Grassl, Brandon Langenberg, Martin Roetteler, and Rainer Steinwandt. 2016. Applying Grover's algorithm to AES: quantum resource estimates. In *International Workshop on Post-Quantum Cryptography*. Springer, 29–43.

[30] Riddhi S Gupta, Neereja Sundaresan, Thomas Alexander, Christopher J Wood, Seth T Merkel, Michael B Healy, Marius Hillenbrand, Tomas Jochym-O'Connor, James R Wootton, Theodore J Yoder, et al. 2024. Encoding a magic state with beyond break-even fidelity. *Nature* 625, 7994 (2024), 259–263.

[31] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. 2009. Quantum Algorithm for Linear Systems of Equations. *Phys. Rev. Lett.* 103 (Oct 2009), 150502. Issue 15. https://doi.org/10.1103/PhysRevLett.103.150502

[32] Raban Iten, Roger Colbeck, Ivan Kukuljan, Jonathan Home, and Matthias Christandl. 2016. Quantum circuits for isometries. *Physical Review A* 93, 3 (2016), 032318.

[33] David Jennings, Matteo Lostaglio, Sam Pallister, Andrew T Sornborger, and Yiğit Subaşı. 2023. Efficient quantum linear solver algorithm with detailed running costs. *arXiv preprint arXiv:2305.11352* (2023).

[34] Hang Jing, Yan Li, Matthew J Brandsema, Yousu Chen, and Meng Yue. 2024. HHL algorithm with mapping function and enhanced sampling for model predictive control in microgrids. *Applied Energy* 361 (2024), 122878.

[35] Iordanis Kerenidis and Anupam Prakash. 2016. Quantum recommendation systems. *arXiv preprint arXiv:1603.08675* (2016).

[36] Iordanis Kerenidis and Anupam Prakash. 2020. Quantum gradient descent for linear systems and least squares. *Physical Review A* 101, 2 (2020), 022316.

[37] Sebastian Krinner, Nathan Lacroix, Ants Remm, Agustin Di Paolo, Elie Genois, Catherine Leroux, Christoph Hellings, Stefania Lazar, Francois Swiadek, Johannes Herrmann, et al. 2022. Realizing repeated quantum error correction in a distance-three surface code. *Nature* 605, 7911 (2022), 669–674.

[38] Ang Li. 2024. NWQSim: Northwest Quantum Circuit Simulation Environment. https://github.com/pnnl/NWQ-Sim.

[39] Ang Li, Bo Fang, Christopher Granade, Guen Prawiroatmodjo, Bettina Heim, Martin Roetteler, and Sriram Krishnamoorthy. 2021. SV-sim: scalable PGAS-based state vector simulation of quantum circuits. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–14.

[40] Ang Li, Chenxu Liu, Samuel Stein, In-Saeng Suh, Muqing Zheng, Meng Wang, Yue Shi, Bo Fang, Martin Roetteler, and Travis Humble. 2024. TANQ-Sim: Tensorcore Accelerated Noisy Quantum System Simulation via QIR on Perlmutter HPC. *arXiv preprint arXiv:2404.13184* (2024).

[41] Xiangyu Li, Xiaolong Yin, Nathan Wiebe, Jaehun Chun, Gregory K Schenter, Margaret S Cheung, and Johannes Mülmenstädt. 2023. Potential quantum advantage for simulation of fluid dynamics. *arXiv preprint arXiv:2303.16550* (2023).

[42] Xiang-Yu Li, Gunilla Svensson, Axel Brandenburg, and Nils EL Haugen. 2019. Cloud-droplet growth due to supersaturation fluctuations in stratiform clouds. *Atmospheric Chemistry and Physics* 19, 1 (2019), 639–648.

[43] Chenxu Liu, Meng Wang, Samuel A Stein, Yufei Ding, and Ang Li. 2023. Quantum Memory: A Missing Piece in Quantum Computing Units. *arXiv preprint arXiv:2309.14432* (2023).

[44] Junyu Liu, Minzhao Liu, Jin-Peng Liu, Ziyu Ye, Yunfei Wang, Yuri Alexeev, Jens Eisert, and Liang Jiang. 2024. Towards provably efficient quantum algorithms for large-scale machine-learning models. *Nature Communications* 15, 1 (2024), 434. https://doi.org/10.1038/s41467-023-43957-x

[45] Jin-Peng Liu, Herman øie Kolden, Hari K. Krovi, Nuno F. Loureiro, Konstantina Trivisa, and Andrew M. Childs. 2021. Efficient quantum algorithm for dissipative nonlinear differential equations. *Proceedings of the National Academy of Sciences* 118, 35 (2021), e2026805118. https://doi.org/10.1073/pnas.2026805118

[46] Guang Hao Low and Isaac L Chuang. 2019. Hamiltonian simulation by qubitization. *Quantum* 3 (2019), 163.

[47] Karl Mayer, Ciarán Ryan-Anderson, Natalie Brown, Elijah Durso-Sabina, Charles H Baldwin, David Hayes, Joan M Dreiling, Cameron Foltz, John P Gaebler, Thomas M Gatterman, et al. 2024. Benchmarking logical three-qubit quantum Fourier transform encoded in the Steane code on a trapped-ion quantum computer. *arXiv preprint arXiv:2404.08616* (2024).

[48] Mohammadhossein Mohammadisiahroudi, Brandon Augustino, Pouya Sampourmahani, and Tamás Terlaky. 2023. Quantum Computing Inspired Iterative Refinement for Semidefinite Optimization. *arXiv preprint arXiv:2312.11253* (2023).

[49] Mohammadhossein Mohammadisiahroudi, Ramin Fakhimi, and Tamás Terlaky. 2022. Efficient use of quantum linear system algorithms in interior point methods for linear optimization. *arXiv preprint arXiv:2205.01220* (2022).

[50] Mohammadhossein Mohammadisiahroudi, Zeguan Wu, Brandon Augustino, Arriele Carr, and Tamás Terlaky. 2023. Improvements to quantum interior point method for linear optimization. *arXiv preprint arXiv:2310.07574* (2023).

[51] Ashley Montanaro and Sam Pallister. 2016. Quantum algorithms and the finite element method. *Physical Review A* 93, 3 (2016), 032324.

[52] Jack Morgan, Eric Ghysels, and Hamed Mohammadbagherpoor. 2024. An Enhanced Hybrid HHL Algorithm. *arXiv preprint arXiv:2404.10103* (2024).

[53] Michael A Nielsen and Isaac L Chuang. 2001. *Quantum computation and quantum information*. Vol. 2. Cambridge university press Cambridge.

[54] Vincent R Pascuzzi, Ning Bao, and Ang Li. 2022. On the importance of scalability and resource estimation of quantum algorithms for domain sciences. *arXiv preprint arXiv:2205.00585* (2022).

[55] John Preskill. 2018. Quantum Computing in the NISQ era and beyond. *Quantum* 2 (Aug. 2018), 79. https://doi.org/10.22331/q-2018-08-06-79

[56] Qiskit contributors. 2023. Qiskit: An Open-source Framework for Quantum Computing. https://doi.org/10.5281/zenodo.2573505

[57] Patrick Rebentrost and Seth Lloyd. 2018. Quantum computational finance: quantum algorithm for portfolio optimization. *arXiv preprint arXiv:1811.03975* (2018).

[58] Martin Roetteler, Michael Naehrig, Krysta M Svore, and Kristin Lauter. 2017. Quantum resource estimates for computing elliptic curve discrete logarithms. In *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II 23*. Springer, 241–270.

[59] Brynjar Sævarsson, Spyros Chatzivasileiadis, Hjörtur Jóhannsson, and Jacob Østergaard. 2024. Quantum Computing for Power Flow Algorithms: Testing on real Quantum Computers. In *Proceedings of 11th Bulk Power Systems Dynamics and Control Symposium (IREP 2022)*. 11th Bulk Power Systems Dynamics and Control Symposium, IREP'2022 ; Conference date: 25-07-2022 Through 30-07-2022.

[60] Kamal K Saha, Walter Robson, Connor Howington, In-Saeng Suh, Zhimin Wang, and Jaroslaw Nabrzyski. 2022. Advancing Algorithm to Scale and Accurately Solve Quantum Poisson Equation on Near-term Quantum Hardware. *arXiv preprint arXiv:2210.16668* (2022).

[61] Artur Scherer, Benoît Valiron, Siun-Chuon Mau, Scott Alexander, Eric Van den Berg, and Thomas E Chapuran. 2017. Concrete resource analysis of the quantum linear-system algorithm used to compute the electromagnetic scattering cross section of a 2D target. *Quantum Information Processing* 16 (2017), 1–65.

[62] P.W. Shor. 1994. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 124–134. https://doi.org/10.1109/SFCS.1994.365700

[63] Yiğit Subaşı, Rolando D. Somma, and Davide Orsucci. 2019. Quantum Algorithms for Systems of Linear Equations Inspired by Adiabatic Quantum Computing. *Phys. Rev. Lett.* 122 (Feb 2019), 060504. Issue 6. https://doi.org/10.1103/PhysRevLett.122.060504

[64] Joran van Apeldoorn, Arjan Cornelissen, András Gilyén, and Giacomo Nannicini. 2023. Quantum tomography using state-preparation unitaries. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 1265–1318.

[65] Wim van Dam, Mariia Mykhailova, and Mathias Soeken. 2023. Using Azure Quantum Resource Estimator for Assessing Performance of Fault Tolerant Quantum Computation. In *Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis*. 1414–1419.

[66] Almudena Carrera Vazquez. 2022. Quantum Linear Solvers. https://github.com/anedumla/quantum_linear_solvers.

[67] Almudena Carrera Vazquez, Ralf Hiptmair, and Stefan Woerner. 2022. Enhancing the quantum linear systems algorithm using Richardson extrapolation. *ACM Transactions on Quantum Computing* 3, 1 (2022), 1–37.

[68] Yang Wang, Selwyn Simsek, Thomas M Gatterman, Justin A Gerber, Kevin Gilmore, Dan Gresh, Nathan Hewitt, Chandler V Horst, Mitchell Matheny, Tanner Mengle, et al. 2023. Fault-tolerant one-bit addition with the smallest interesting colour code. *arXiv preprint arXiv:2309.09893* (2023).

[69] Jingwei Wen, Xiangyu Kong, Shijie Wei, Bixue Wang, Tao Xin, and Guilu Long. 2019. Experimental realization of quantum algorithms for a linear system inspired by adiabatic quantum computing. *Physical Review A* 99, 1 (2019), 012320.

[70] Romina Yalovetzky, Pierre Minssen, Dylan Herman, and Marco Pistoia. 2021. Hybrid HHL with Dynamic Quantum Circuits on Real Hardware. *arXiv preprint arXiv:2110.15958* (2021).

[71] Anika Zaman, Hector Jose Morrell, and Hiu Yung Wong. 2023. A Step-by-Step HHL Algorithm Walkthrough to Enhance Understanding of Critical Quantum Computing Concepts. *IEEE Access* (2023).

[72] Muqing Zheng, Yousu Chen, Xiu Yang, and Ang Li. 2024. Early Exploration of a Flexible Framework for Efficient Quantum Linear Solvers in Power Systems. *arXiv preprint arXiv:2402.08136* (2024).

[73] Yifan Zhou, Zefan Tang, Nima Nikmehr, Pouya Babahajiani, Fei Feng, Tzu-Chieh Wei, Honghao Zheng, and Peng Zhang. 2022. Quantum computing in power systems. *IEnergy* 1, 2 (2022), 170–187.

[74] Ray D. Zimmerman and Carlos E. Murillo-Sánchez. 2020. MATPOWER. https://doi.org/10.5281/ZENODO.4074135