QCE'24 Quantum Resource Estimation Educational Challenge Submission - Matrix Inversion by QSVT

Walden Killick

July 23, 2024

Abstract

We investigate the physical resources required to solve systems of linear equations using the quantum singular value transform algorithm. We find that empirically, both the qubit counts and runtimes scale favourably with the input size, and by extrapolating the data we predict that our implementation may be able to demonstrate quantum advantage for matrix inversion with achievable physical resources. We furthermore identify bottlenecks in the scaling and highlight opportunities for further optimisation.

1 Introduction

Since the breakthrough 2009 algorithm of Harrow, Hassidim, and Lloyd (HHL algorithm), it has been well-known that for sparse, well-conditioned matrices, quantum computers are capable of solving the system of linear equations (SLE) problem exponentially faster than the best possible classical algorithms [1]. Due to the complexities of practical implementation, the performance analyses of quantum algorithms for solving SLEs have thus far been primarily restricted to the asymptotic regime. In this project, we aim to answer the question of what physical quantum resources are required to solve SLEs of sizes at which the problem becomes classically intractable. Specifically, we consider the use of the quantum singular value transformation (QSVT) which achieves the best asymptotic complexity of all known quantum SLE algorithms [2, 3].

All code used to generate the results in this report can be found at https://github.com/Walden-Killick/QCE24-QRE-Challenge.

2 Preliminary

2.1 Problem definition

Given a quantum state $|b\rangle$ and an efficient classical description of a sparse matrix A, the quantum system of linear equations (QSLE) problem is the problem of preparing the state $|x\rangle = A^{-1}|b\rangle$. Broadly speaking, the primary challenges in any QSLE algorithm are (1) accessing A in a quantum circuit, and (2) performing (approximately) the transformation $x \mapsto 1/x$ on A's eigenvalues. The HHL algorithm [1] achieves these goals using a sparse Hamiltonian simulation subroutine [4] and quantum phase estimation [5], respectively. By contrast, QSVT utilises the fundamentally different techniques of block encoding [2] and quantum signal processing [6] and achieves an exponentially improved dependence on the target precision as well as a lesser improvement with respect to the condtion number. We briefly review these techniques in the following subsection.

2.2 Matrix inversion by QSVT

To access A in a quantum circuit, QSVT relies on quantum oracles which efficiently encode both the non-zero structure and non-zero elements of A. These are defined rigorously as follows.

Definition 2.1 (Sparse access oracles [7])**.** *Let A be a $2^n \times 2^n$ s-sparse matrix. Let $c(j,l)$ be a function which returns the row index of the l-th non-zero matrix element in the j-th column of A. The sparse access oracles O_c and O_A for A are the unitary operators defined as*

$$O_c|l\rangle|j\rangle = |l\rangle|c(j,l)\rangle$$

and

$$O_A|0\rangle|l\rangle|j\rangle = \left(A_{c(j,l),j}|0\rangle + \sqrt{1-\left|A_{c(j,l),j}\right|^2}|1\rangle\right)|l\rangle|j\rangle.$$

We should note that in this formalism, O_A also implicitly encodes the non-zero structure of A by calling $c(j,l)$; however, for certain sufficiently well-structured matrices such as those considered in this project, this is not prohibitive.

The following informal theorem states that one can construct a unitary matrix which directly contains (a scaled version of) A as a submatrix (block).

Theorem 2.2 (Block-encoding sparse matrices [2, 7])**.** *Let A, O_c, O_A be as in Definition 2.1. Using one call to each of O_c and O_A, we can construct a larger unitary matrix U_A for which the upper left $2^n \times 2^n$ block is equal to A/s.*

The final ingredient in matrix inversion by QSVT is then to transform the singular values of A/s as $x \mapsto 1/x$. QSVT cannot perform this transformation exactly, but can perform almost-arbitrary polynomial transformations [8] and thus approximate $1/x$ to arbitrary precision.

Theorem 2.3 (Matrix inversion by QSVT [2])**.** *Let A be a sparse matrix with condition number κ and let U_A block-encode A as in Theorem 2.2. Using $O(\kappa \log(\kappa/\varepsilon))$ calls to U_A and $U_A^\dagger$, we can construct a unitary which block-encodes A^{-1} to within accuracy ε.*

We refer the reader to [3] for a pedagogical introduction to block encodings and QSVT.

2.3 Banded circulant matrices

To construct polynomial-size oracles O_c and O_A, A must be well-structured in some way. For this project, we consider *banded circulant* matrices primarily due to the known explicit and simple construction of their sparse access oracles.

Definition 2.4 (Circulant matrix)**.** *A circulant matrix is a matrix in which each row is equal to the previous row but with each element shifted one place to the right.*

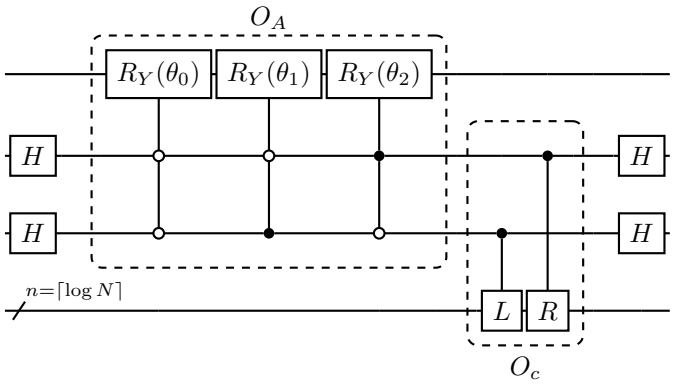

Figure 1: Quantum circuit for block-encoding an $N \times N$ banded circulant matrix, adapted from [7]. For $n = \lceil \log N \rceil$ qubits, the L and R gates each consist of n multi-controlled NOT gates with successively decreasing numbers of controls (see Figure 2).

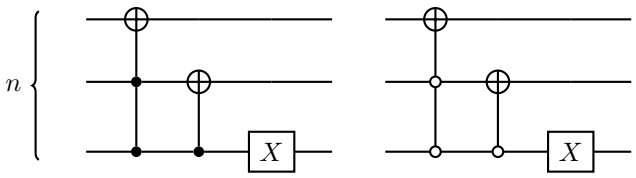

Figure 2: Quantum circuits for the L and R gates respectively, with $n = 3$ qubits.

Definition 2.5 (Banded circulant matrix)**.** *A circulant matrix is banded if only the first three entries of the second row are non-zero.*

The following is an example of a banded circulant matrix with diagonal entries α, sub-diagonal entries β, and super-diagonal entries γ:

$$\begin{bmatrix} \alpha & \gamma & 0 & \cdots & \beta \\ \beta & \alpha & \gamma & \cdots & 0 \\ 0 & \beta & \alpha & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma & 0 & 0 & \cdots & \alpha \end{bmatrix}.$$

Physically, a banded circulant matrix corresponds to the adjacency matrix of a cycle graph.

Reference [7] shows how to construct polynomial-size sparse-access oracles for this class of matrices, for which we show a high-level sketch in Figure 1. In particular, the authors show how to construct O_A in $O(1)$ time and O_c in $O(\text{poly}\log(N))$ time. Combining this with Theorems 2.2 and 2.3, we can see that using the aforementioned construction, we can perform matrix inversion by QSVT in time $O(\kappa\,\text{poly}\log(N)\log(\kappa/\varepsilon))$, retaining the exponential quantum speedup with respect to the matrix size. This furthermore outperforms the HHL algorithm with respect to both precision and the condition number, which has a complexity of $O(\kappa^2 \log(N)/\varepsilon)$ [1].

3 Results

The resource estimates were generated by exporting Qiskit QSVT circuits to OpenQASM 2 files and subsequently loading them into the Microsoft Azure Quantum Resource Estimator via the web portal. We are primarily concerned with observing how the resource estimates scale with the input size N, and thus arbitrarily assume condition number $\kappa = 3$ and target accuracy $\varepsilon = 0.1$ for all circuits.

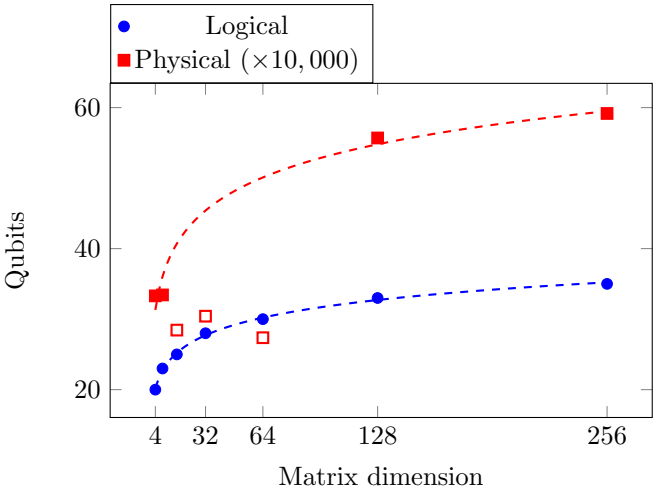

Figure 3: Resource estimates for the logical and physical qubit requirements for matrix inversion by QSVT.

3.1 Qubit counts

Figure 3 shows how the logical and physical resource estimates scale with the input size. Whilst the logical qubit estimates conform very naturally with the expected logarithmic growth, the physical estimates scale much more erratically, most likely due to the lack of a consistent structure between problem instances when transpiling error-corrected circuits to the physical layout. In the interest of extrapolating to worst-case resource estimates, we mark as outliers the instances which outperform an instance of smaller size (indicated with hollow markers in Figure 3). Promisingly, despite the somewhat irregular results, the physical resource estimates nonetheless likewise conform to a logarithmic growth with N.

The logarithmic best-fit functions, computed using NumPy's polyfit, are roughly:

$$y_{\text{logical}} \approx 3.61 \log N + 15.2,$$
$$y_{\text{physical}} \approx 67800 \log N + 219000,$$

where log denotes the natural logarithm.

Although highly impractical for small problem sizes due to the large constant factors, the data would seem to empirically imply that a realistically achievable number of physical qubits will be able to tackle problems of classically intractable sizes. For example, if the extrapolation of the physical qubit requirements remains accurate for large inputs, a 10^{100}-dimensional banded circulant matrix could be inverted using under 16 million physical qubits.

3.2 Runtimes

Figure 4 shows how the runtime estimates scale with the input size. In this case, the asymptotic growth of the runtime at first glance appears linear as opposed to the desired polylogarithmic growth.

To diagnose this issue, we first note that the only circuit elements that change size between instances are the L and

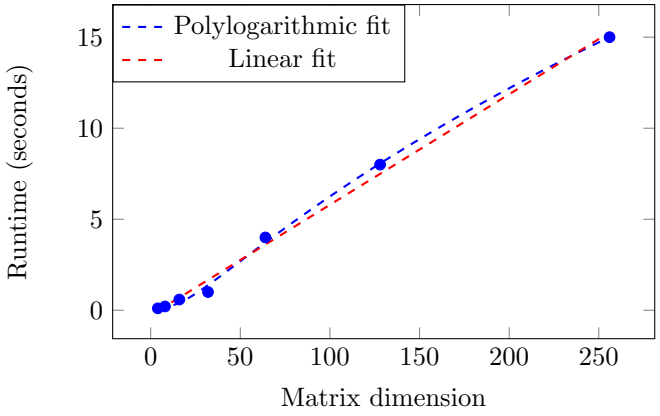

Figure 4: Resource estimates for the running time of matrix inversion by QSVT.

R gates (see Figure 1) which ultimately consist of several multi-controlled NOT gates (see Figure 2). In the exploratory notebook QISKIT_TRANSPILATION.IPYNB, we have empirically found the runtime of Qiskit's transpilation of multi-controlled gates to scale as $O(n^2)$, implying a 'naive' implementation such as the construction found in [9]. With the number of such gates present in the block-encoding circuit, we should then expect to see a runtime scaling of $O(n^3) = O(\log^3 N)$. Fitting the data to a degree-3 polylogarithmic function using SciPy's curve_fit yields

$$y_{\text{runtime}} = 0.308 \log^3 N - 1.77 \log^2 N + 3.42 \log N - 2.03$$

which indeed conforms to the dataset with a smaller average error than the linear fit. If the extrapolation to higher dimensional matrices remains accurate, with this implementation a 10^{10}-dimensional banded circulant matrix could be inverted on a quantum computer in about one hour.

4 Conclusion and discussion

Whilst of significant practical relevance, quantum algorithms for solving SLEs are still predominantly presented in a complexity-theoretic and oracular, rather than a practical, setting. In this project, we have built explicit quantum circuits for solving SLEs by QSVT for a specific class of sparse matrices, and evaluated the end-to-end physical resource requirements using Microsoft's Azure Quantum Resource Estimator.

Our findings show that both the logical and physical qubit count estimates adhere to the expected logarithmic growth, and that a realistically achievable number of physical qubits may suffice to demonstrate quantum advantage for matrix inversion.

The resource estimates for runtime seem to display a lesser degree of certainty in their asymptotic growth with matrix size, but we can deduce theoretically that we should expect to see a growth which scales as $O(\log^3 N)$. Fitting the data as such, if the behaviour remains consistent when extrapolated to large matrix sizes, we should likewise expect a reasonable amount of time to suffice to invert matrices of classically challenging sizes.

There is, however, further room for optimisation in these estimates. When scaling to large matrices, we are bottlenecked by the many multi-controlled NOT gates present in O_c (see Figure 2). Qiskit appears to transpile such gates into circuits of depth $O(n^2)$, resulting in an overall complexity for O_c of $O(\log^3 N)$. The best way to transpile multi-controlled gates is an area of active research, with recent theoretical results achieving depths as low as $O(\log^3 n)$ [10]. With the $n \sim \log N$ gates present in L and R, this would imply an overall runtime which scales with input size as $O(n \log^3 n) = O(\log N \log^3 \log N)$. If and by how much these results reduce resource requirements in practice remains an open question and a potential direction for future work.

References

[1] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. "Quantum algorithm for linear systems of equations". In: *Physical review letters* 103.15 (2009), p. 150502.

[2] András Gilyén et al. "Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics". In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. 2019, pp. 193–204.

[3] John M Martyn et al. "Grand unification of quantum algorithms". In: *PRX quantum* 2.4 (2021), p. 040203.

[4] Dominic W Berry et al. "Efficient quantum algorithms for simulating sparse Hamiltonians". In: *Communications in Mathematical Physics* 270 (2007), pp. 359–371.

[5] A Yu Kitaev. "Quantum measurements and the Abelian stabilizer problem". In: *arXiv preprint quant-ph/9511026* (1995).

[6] Guang Hao Low and Isaac L Chuang. "Optimal Hamiltonian simulation by quantum signal processing". In: *Physical review letters* 118.1 (2017), p. 010501.

[7] Daan Camps et al. "Explicit quantum circuits for block encodings of certain sparse matrices (2022)". In: *arXiv preprint arXiv:2203.10236* ().

[8] Christoph Sünderhauf. "Generalized Quantum Singular Value Transformation". In: *arXiv preprint arXiv:2312.00723* (2023).

[9] Adriano Barenco et al. "Elementary gates for quantum computation". In: *Physical review A* 52.5 (1995), p. 3457.

[10] Baptiste Claudon et al. "Polylogarithmic-depth controlled-NOT gates without ancilla qubits". In: *Nature Communications* 15.1 (2024), p. 5886.