scientific reports

OPEN

Addressing quantum's "fine print" with efficient state preparation and information extraction for quantum algorithms and geologic fracture networks

Jessie M. Henderson[1,4]✉, John Kath[2,4], John K. Golden[1], Allon G. Percus[2] & Daniel O'Malley[3]

Quantum algorithms provide an exponential speedup for solving certain classes of linear systems, including those that model geologic fracture flow. However, this revolutionary gain in efficiency does not come without difficulty. Quantum algorithms require that problems satisfy not only algorithm-specific constraints, but also application-specific ones. Otherwise, the quantum advantage carefully attained through algorithmic ingenuity can be entirely negated. Previous work addressing quantum algorithms for geologic fracture flow has illustrated core algorithmic approaches while incrementally removing assumptions. This work addresses two further requirements for solving geologic fracture flow systems with quantum algorithms: efficient system state preparation and efficient information extraction. Our approach to addressing each is consistent with an overall exponential speed-up.

Quantum algorithms promise to revolutionize the solving of linear systems, which are essential components of problems in medicine, finance, urban development, and nearly any field that can be classified as a natural or applied science[1]. Several quantum algorithms[2–8] can provide a provable exponential speedup over classical linear solvers. However, remarkable though such gains are, they do not come without cost, nor without complication[9]. Problems of interest must be curated to satisfy algorithm-specified constraints. Moreover, the quantum algorithms themselves must account for the complexity of transporting information to and from the quantum computer via processes that bear little resemblance to classical counterparts[9]. Otherwise, all theoretical intrigue aside, quantum linear-systems algorithms become toothless: we know the algorithm could compute the solution exponentially faster than possible classically, but we can neither supply the problem nor extract the solution efficiently enough to benefit from this speedup[10]. These difficulties are further complicated by context-specific considerations; for example, it would be pointless to prepare and solve a linear system if we cannot extract information that is of practical relevance for the problem at hand.

Consequently, efficient approaches for both preparing the system to be solved and extracting useful information from quantum computers are as important as quantum algorithms themselves. This paper addresses these issues within the context of quantum algorithms for geologic fracture networks; while the approaches are fairly general and thus may have utility for applications beyond geologic linear systems, we explicitly addresses only the realm of geologic fracture problems. Linear systems representing fracture networks are too large to solve in their entirety with even the most sophisticated classical approaches[11,12], and reducing problem size requires methods such as upscaling, which supply only approximate solutions that may neglect important features of the network. For example, when small fractures are neglected, a network exhibiting percolation—complete connectivity of a fracture region—might no longer manifest that effect[13]. Such modelling issues make geologic fracture problems a prime candidate for benefiting from the speedup provided by quantum algorithms, so long as we can satisfy the algorithmic constraints and provide efficient state preparation and information extraction. Previous work has addressed solving fracture flow problems with quantum algorithms[14–18] while making assumptions about the auxiliary issues. Reference[19] then addressed the requirement of well-conditioned matrices by developing effective

[1]Los Alamos National Laboratory, CCS-3, Los Alamos, NM 87545, USA. [2]Institute of Mathematical Sciences, Claremont Graduate University, Claremont, CA 91711, USA. [3]Los Alamos National Laboratory, EES-16, Los Alamos, NM 87545, USA. [4]These authors contributed equally: Jessie M. Henderson and John Kath. ✉email: jessieh@lanl.gov

preconditioning permitting quadratic speedup for systems representing geologic fracture problems. Here, we address two further constraints of efficient state preparation and solution extraction.

The remainder of the paper proceeds as follows. We first provide an introduction to modelling geologic fracture networks with linear systems, including the relationship to quantum algorithms. We then present methods for state preparation and information extraction that have acceptable complexities and that are readily usable by human developers, including on noisy near-term quantum hardware. We include toy examples for both the state preparation and information extraction approaches to clarify method usage. Finally, we conclude with a brief discussion of future work, which includes empirical study of efficient, start-to-finish solution of varying-scale geologic fracture problems on newly-available, higher-qubit, less error-prone quantum hardware.

Background

Quantum algorithms for geologic fracture networks

Simulating geologic fracture networks is one of the most challenging problems in geophysics, in part because of the large range over which fractures exist[20–23]. Systems modelling fractures with sizes between 10^{-6} and 10^4 m cannot be solved accurately in their entirety on classical machines, and they sometimes cannot be accurately upscaled either. Specifically, information lost during upscaling pertains to small fractures that can have a critical effect on the fracture network; for example, the smallest fractures can determine whether the network crosses a percolation threshold, which has a substantial impact on fluid flow[13].

Quantum algorithms for solving linear systems are not burdened by the same constraints as their classical counterparts. The properties of quantum mechanics endow them with a fundamentally different physics—and thus a fundamentally different mathematics—that allows for efficiently solving problems that cannot be solved on classical computers using a reasonable amount of memory or time[10]. Previous work has both explained and demonstrated use of quantum algorithms for solving linear systems problems in the geologic fracture realm[14–17], but with the caveat that future work would need to explore efficient mechanisms of introducing the problem to the computer and extracting meaningful information from the solution. In this work, we consider those issues for pressure-identification problems of the form $\nabla \cdot (k\Delta h) = f$, where k is permeability, f is a fluid source or sink, and h is the pressure to be computed. These problems can be discretized and written as $A\mathbf{x} = \mathbf{b}$, where the pressure for each discretized node is stored in $\mathbf{x}$. Then, quantum algorithms for solving linear systems can compute a normalized vector that is proportional to that solution[2].

One such algorithm—which has been considered for geologic fracture network problems in Ref.[17]—is the Harrow-Hassidim-Lloyd (HHL) algorithm, which was the first for solving linear systems with quantum circuits. It provides an exponential speedup over classical algorithms under certain conditions, including constraints on matrix sparseness and condition number[2]. Because geologic fracture flow systems can be made to satisfy such conditioning requirements[19], and because they are unavoidably large in their complete form, such systems are ideal candidates for the HHL algorithm[17], assuming that we can efficiently specify the problem and extract the solution. Approaches to these information-transfer tasks are not algorithm-independent; the details can depend upon how a given quantum algorithm prepares the solution vector $\mathbf{x}$. However, there are often similarities in quantum linear systems algorithm structure that would make state-preparation and information-extraction approaches inter-algorithmically applicable. In this work, we will directly consider only the HHL algorithm while acknowledging that our methods for state preparation and information extraction may be more broadly applicable.

Brief introduction to HHL

The HHL algorithm prepares a solution proportional to that of the $N \times N$ system $A\mathbf{x} = \mathbf{b}$[2]. A single execution of the algorithm has a complexity of $O\left(\log(N)s^2\kappa^2/\epsilon\right)$, where N is the size of system, s is the sparseness of the matrix, κ is the condition number of the matrix, and ϵ is the additive error within which the system is solved[2,24]. As with most quantum algorithms, HHL is at once fairly simple in qualitative terms and quite subtle in quantitative ones. Given the algorithm's wide applicability and dramatic speedup, several works look to provide detailed—yet accessible—treatments of the algorithm; for more information, please see Refs.[25] or[26]. While this work is not intended as a detailed introduction to HHL, it is worth briefly describing the overall algorithm and highlighting a few relevant points.

HHL prepares a normalized solution to $\mathbf{x}$ by leveraging the fact that $\mathbf{x} = \sum_{i=1}^{N} \lambda_i^{-1} b_i |u_i\rangle$, where $|u_i\rangle$ is the ith eigenvector of A and λ_i is its associated eigenvalue. Specifically, as shown in the block diagram of Fig. 1, the algorithm requires two registers of qubits and an ancilla qubit. The b-register begins as storage for normalized values proportional to those in the right-hand side vector of $\mathbf{b}$, and if the ancilla is measured as 1, then it ends storing $|x\rangle$, which is a normalized vector proportional to $\mathbf{x}$. HHL therefore requires an efficient mechanism for converting the b-register (with n_b qubits) from the fiduciary state of $|0...0\rangle$ to a state in which each qubit holds two values of the normalized $\mathbf{b}$: $|bReg_0\rangle = b_0|0\rangle + b_1|1\rangle, |bReg_1\rangle = b_2|0\rangle + b_3|1\rangle, \ldots, |bReg_{n_b}\rangle = b_{2^{n_b}-2}|0\rangle + b_{2^{n_b}-1}|1\rangle$.

The w-register is a working register that is used to store intermediate values throughout the computation. Specifically, it is used during the subroutines of Quantum Phase Estimation (QPE) and Inverse QPE, which identify and isolate the eigenvalues λ_i of A alongside a normalization constant that makes the final state proportional to (rather than equal to) $\mathbf{x}$. Consequently, the information about A necessary to solve the linear system is encoded in the QPE subroutine, and HHL's efficiency requires that A be well-conditioned and sparse for this information-transfer process to avoid a complexity greater than that of the entire solve.

Finally, the ancilla qubit determines whether $|x\rangle$ is properly stored at the end of the algorithm; measuring it decouples the solution to the system ($|x\rangle = \gamma \sum_{i=1}^{N} \lambda_i^{-1} b_i |u_i\rangle$, where γ is a normalization constant) from

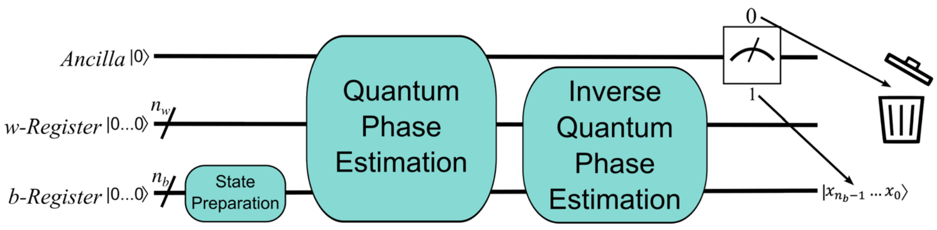

Figure 1. A block diagram of HHL. After a separate, *non-HHL* algorithm has prepared the right-hand-side vector, **b**, HHL applies QPE to extract the eigenvalues and eigenvectors from the matrix, A. Then, the algorithm prepares the eigenvalues for extraction from their entangled state using inverse QPE before measuring an ancilla qubit to determine if the operation was successful. A measurement of 1 indicates that $|x\rangle \sim \mathbf{x}$ is stored in the b-register, while a value of 0 indicates a probabilistic circuit failure, the likelihood of which is included in the complexity of HHL via ϵ.

associated 'garbage' information $\left(|x\rangle = \sum_{i=1}^{N} \sqrt{1 - \frac{\gamma^2}{\lambda_i^2}} b_i |u_i\rangle \right)$ that is added throughout the computation. Thus, HHL requires multiple circuit executions to account for when the ancilla is measured as a zero[24].

Before discussing HHL's utility for geologic fracture problems, it is worth noting that the HHL algorithm is known to be impractical for near-term quantum hardware. This is a consequence of the QPE and inverse QPE subroutines, both of which require more qubits that are less prone to decoherence than are generally available on today's machines.[27–33]. Nonetheless, the algorithm is of import, because quantum hardware is rapidly growing and becoming less error-prone[34–36], suggesting that the existing practical limitations of HHL are temporary, especially given variations on HHL that are tailored to problems with simpler QPE and inverse QPE subroutines that are consequently less resource intensive[37,38].

Figure 2 illustrates the relation between a geologic fracture flow problem and a circuit implementing HHL. First, the region of interest for a particular problem (say, the $\nabla \cdot (k\Delta h) = f$ of above) is discretized to form A and **b**. Then A determines the parameters in the fixed structure of a QPE subroutine. A state preparation procedure encodes the values of **b** in the b-register, and both the ancilla qubit and the qubits in the w-register begin as $|0\rangle$. If, at algorithm completion, the ancilla is measured as 1, then the problem solution—representing the pressure at each node of the discretization—is stored in what began as the b-register: $|bReg_0\rangle = x_0|0\rangle + x_1|1\rangle, |bReg_1\rangle = x_2|0\rangle + x_3|1\rangle, \dots, |bReg_{n_b}\rangle = x_{2^{n_b}-2}|0\rangle + x_{2^{n_b}-1}|1\rangle$.

Useful though such conceptual schematics are, they provide no guidance on efficiently encoding information into the b-register or extracting the eventual solution. As Scott Aaronson recognizes in his oft-cited remarks on the "fine print" of quantum algorithms[9], this is in part because such determinations are often very difficult problems in and of themselves, and indeed can be so difficult that the issue of transferring information to and

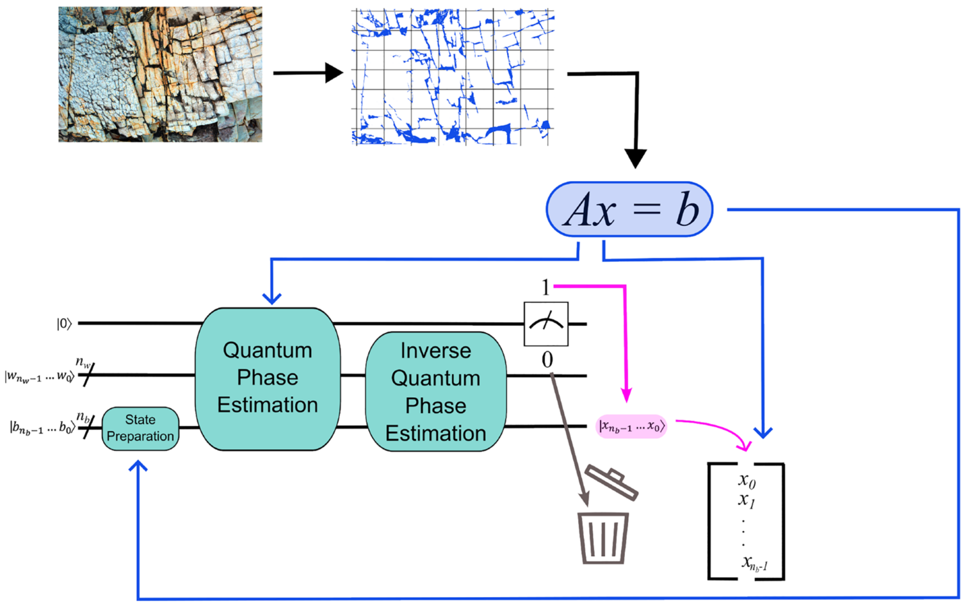
$Ax = b$
|0⟩
$|w_{n_w-1} \dots w_0\rangle$
$|b_{n_b-1} \dots b_0\rangle$
n_w
n_b
State Preparation
Quantum Phase Estimation
Inverse Quantum Phase Estimation
1
0
$|x_{n_b-1} \dots x_0\rangle$
x_0
x_1
x_{n_b-1}

Figure 2. A schematic illustrating how information is transferred from a geologic fracture flow problem into an HHL circuit that solves for the pressure at each discretized node.

from the quantum computer bars use of quantum approaches that would otherwise be preferable to classical variants. For example, obtaining the entirety of $|x\rangle$ requires $O(N)$ measurements, which requires running the entire HHL circuit $O(N)$ times, thus negating the complexity benefit provided by HHL in the first place. The next subsection introduces two subroutines that we will use to address this issue.

The Swap and Hadamard tests

The swap test is a straightforward quantum subroutine that obtains the overlap between two n-qubit quantum states using n controlled-swap gates, two Hadamard gates, and one measurement[39,40]. The circuit must be run $O(1/\epsilon^2)$ times to obtain a solution that is within a specified additive error, ϵ, of the true overlap. Subfigure a of Fig. 3 illustrates the circuit structure; by measuring the single qubit and obtaining the probability that it is 0, the inner product between the two registers is given as $|\langle\phi|\psi\rangle|^2 = 1 - 2p(0)$. It is thus worth noting that the swap test is capable only of determining the magnitude of the inner product and not its sign; the slightly more complicated Hadamard test addresses this, as illustrated in Subfigure b of Fig. 3. Specifically, the Hadamard test for computing inner products uses two Hadamard gates, two Pauli-$\mathbf{X}$ gates, and two controlled-U gates for unitaries U that prepare the states whose overlap is to be computed[41]. For two n-qubit registers, the complexity is still $O\left(1/\epsilon^2\right)$, and the inner product is given by $\mathrm{Re}[\langle\phi|\psi\rangle] = 2p(0) - 1$. (The imaginary component can be computed with a slight adjustment, but since geologic fracture network problems do not require complex values, we will not consider that here). Below, we will utilize these subroutines to efficiently extract the average pressure from a user-specified set of nodes after solving for the pressure in all nodes. But first, we describe efficiently preparing the b-register.

Efficient state preparation

One of HHL's critical assumptions is availability of an efficient method for preparing the quantum state of the b-register. The preparation of a general state $\mathbf{b}$ can require a computational effort of $\Omega(2^{n_b})$, negating the advantage of a quantum computation. Our approach, based on the algorithm of Gleinig and Hoefler[42], enables efficient generation of the quantum state $\mathbf{b}$ specifically tailored to subsurface flow problems.

For our fracture network problems, $\mathbf{b}$ encodes the boundary conditions. We will consider examples with a combination of Dirichlet boundary conditions, which specify pressure, and Neumann boundary conditions, which specify flux. Specifically, we present two scenarios that arguably represent the two most common scenarios considered when modelling subsurface flow.

Pressure gradient

In the first scenario, we consider Dirichlet boundary conditions where the left boundary experiences high pressure while the right boundary has low pressure, and we impose Neumann boundary conditions with zero flux at the top and bottom. In this case, preparing the state of $\mathbf{b}$ is straightforward. Without loss of generality, assume that n_b is even, and let $m = n_b/2$. The b-register starts in a state of $2m$ zero qubits ($|0^m\rangle|0^m\rangle$), and we apply Hadamard gates to the second m qubits to put them in a uniform superposition, giving

$$|0^m\rangle \frac{1}{\sqrt{2^m}} \sum_{i\in\{0,1\}^m} |i\rangle.$$

This produces a circuit with $n_b/2$ gates. Figure 4 depicts preparing $\mathbf{b}$ under a pressure gradient when $n_b = 8$.

Fluid injection

The remainder of this section discusses the second scenario. We consider Dirichlet boundary conditions where the left and right boundaries maintain zero pressure. We again impose Neumann boundary conditions with zero flux at the top and bottom. Additionally, we introduce the concept of injecting or extracting fluid at a small number of sites—*i.e.*, wells—in the middle of the domain, which is comparable to Neumann boundary conditions that specify a flux. It is worth emphasizing that, in practice, the number of wells considered in a given simulation is constant and typically small, meaning that the number of wells does not increase as the simulation mesh is refined.

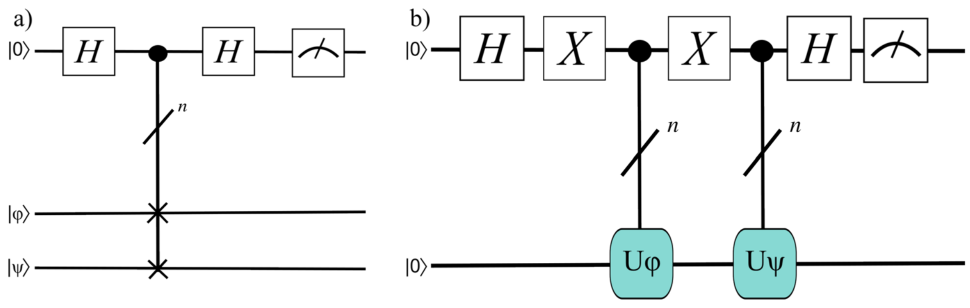

Figure 3. Subfigure (**a**) illustrates the swap test for two n-qubit registers, $|\phi\rangle$ and $|\psi\rangle$, while Subfigure (**b**) illustrates the Hadamard test for two n-qubit states that can be efficiently prepared via the gates U_ϕ and U_ψ.

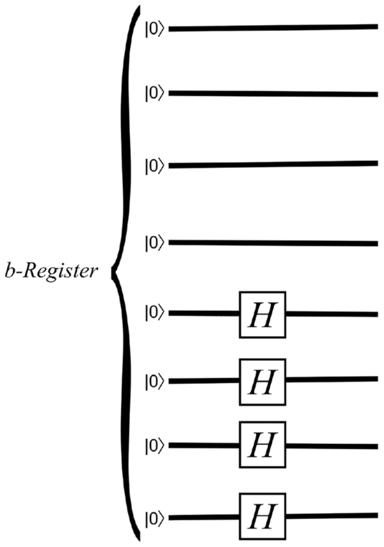

Figure 4. A quantum circuit that prepares the state of **b** for a fracture flow problem with a pressure gradient where $n_b = 8$. Note that **b** has $2^{n_b/2}$ nonzero entries.

Because the number of wells is small, the vector **b** is sparse, meaning it has few nonzero entries, with the total number of non-zero entries equal to the number of injection/extraction wells. In general, a circuit for preparing an n_b-qubit quantum state is a sequence of $O(2^{n_b})$ gates, where multicontrolled operations are used for readability. The method of Ref.[42] exploits the sparsity of **b**, taking a classical specification of a quantum state ϕ with W nonzero coefficients as input and producing, in polynomial time, a polynomial-size circuit C that maps the initial state $|0^{n_b}\rangle$ to ϕ. Note that W is both the number of nonzero coefficients and the number of injection/extraction wells.

The algorithm works by enabling efficient transformation of quantum states. These transformations range from basis states to zero states using NOT gates to more complex scenarios involving superpositions of basis states. The key challenge is preventing the "splitting" of basis states during transformation. Reference[42] relies on two crucial techniques:

1. Controlling the merging of basis states with no more than $O(\log W)$ control bits.
2. Using efficiently-implementable gate sequences for multicontrolled operations.

The circuit is generated from a gate library that includes controlled-NOT (CNOT) gates and single-qubit T gates. Using this method, the state can be prepared using only $O(Wn_b)$ CNOT gates and $O(W \log W + n_b)$ single qubit gates. Since $\log W$ is itself $O(n_b)$, this results in a total number of gates that is $O(Wn_b)$, rather than of exponential complexity.

Application to fracture networks

The polynomial-sized quantum circuit produced for preparing a sparse state illustrates this algorithm's practical relevance for solving geologic fracture flow problems with quantum algorithms. The following examples demonstrate efficient state preparation for fracture networks. First, we consider a simple case consisting of two intersecting fractures and two injection/extraction sites.

Consider $\mathbf{b} = (0, -164, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 113, 0, 0)$ where $n_b = 4$ and there are $W = 2$ nonzero coefficients. In this case, we have

$$|b\rangle = \frac{1}{\sqrt{39665}}(-164|0001\rangle + 113|1100\rangle).$$

By applying the algorithm of Ref.[42], we can generate this state using the circuit depicted in Fig. 5.

Figure 6 depicts a two-dimensional fracture network model involving two fractures intersecting in a †-configuration, with fractal-style recursion of the †-system to generate a more complicated pitchfork fracture network. The relative permeability of the fractures as compared to that of the underlying rock is a critical parameter in the analysis of fracture systems, and thus, low and high permeability contrast is represented by a gradient scale, where the smallest fractures have the least permeability contrast[19]. Fluid injection/extraction wells are located randomly at sites corresponding to red dots. And, as above, solving this fracture network problem with Neumann boundary conditions requires generating a quantum circuit which prepares the state of a sparse vector **b**.

We evaluate the performance of the method in Ref.[42] applied to this fracture problem. To generate random sparse states comprising n_b qubits with W nonzero coefficients, we select W distinct basis states $|x\rangle$, where $x \in \{0,1\}^{n_b}$, from a random, uniform distribution, and we form ϕ as the superposition of these selected states.

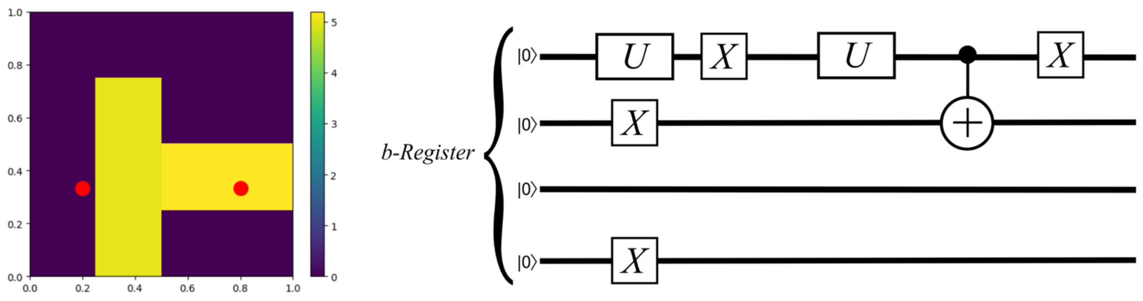

Figure 5. On the left, a simplified two-dimensional fracture network model with two fractures intersecting and two random fluid injection/extraction wells at the red dots. The color bar represents permeability on a logarithmic scale. **b** encodes boundary conditions for a 4×4 grid with $2^{n_b} = 16$ cells $\Rightarrow n_b = 4$. Nonzero entries in **b** correspond to well sites. On the right, a quantum circuit to prepare the state $|b\rangle = \frac{1}{\sqrt{39665}}(-164|0001\rangle + 113|1100\rangle)$ for this fracture flow problem with Neumann boundary conditions (sparse nonzero entries in **b**).

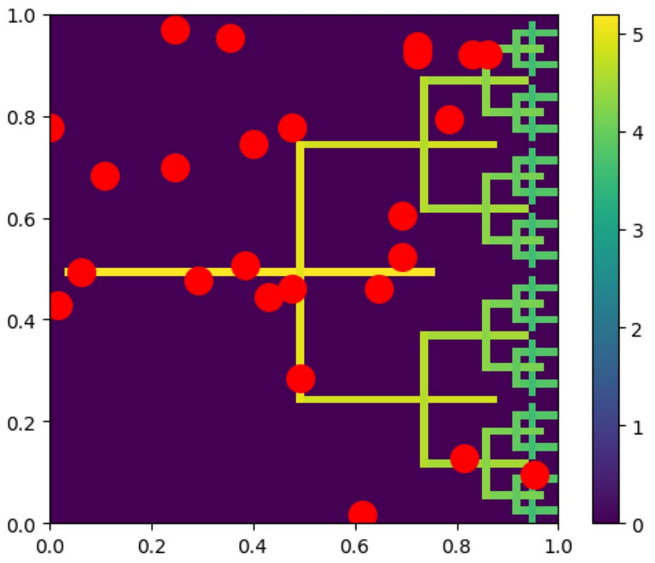

Figure 6. An illustration of a two-dimensional fracture network model with fractures that intersect in a pattern of fractal-style recursion to generate a pitchfork fracture network. Random fluid injection/extraction wells are located at the red dots. The color bar represents permeability on a logarithmic scale. **b** encodes Neumann boundary conditions (sparse nonzero entries) for a 64×64 grid with $2^{n_b} = 4096$ cells $\Rightarrow n_b = 12$. Nonzero entries in **b** correspond to well sites.

Coefficient values are chosen from a random normal distribution and subsequently normalized to represent the weights given to random injection/extraction wells in the problem domain. Note that in a uniform superposition, the coefficients of the W selected basis states are equal, and the size of the state-preparation circuit produced remains independent of the exact coefficient values, as long as they are nonzero.

In Fig. 7, we explore how the circuit size scales with an increase in W from 1 to 25 while keeping n_b fixed at $n_b = 12$. That is, the number of randomly-generated injection/extraction points increases from left to right along the horizontal axis. For each value of W, we conduct 5 random state samples. The average gate count is illustrated in the figure along with a bar indicating the smallest and largest counts. The quantum circuits produced of $O(Wn_b)$ size are asymptotically better than those produced by general state preparation methods of size $O(2^{n_b})$ for $W \ll 2^{n_b}$ (*i.e.*, when **b** is sparse).

Efficient information extraction

We now consider a second challenge in applying HHL to geologic fracture systems, namely extracting information upon solution completion. Because it is unrealistic to obtain all of the pressures from the quantum computer's solution, we must consider other quantities of interest[2,10], and in the realm of geologic fracture networks, one such quantity is the average pressure in a particular region. Average pressure is a sum of pressures in individual nodes divided by the total number of nodes considered, so we can obtain this using the swap test (or Hadamard test) with a register that prepares an appropriately complementary state to the $|x\rangle$ nodes whose average we seek.

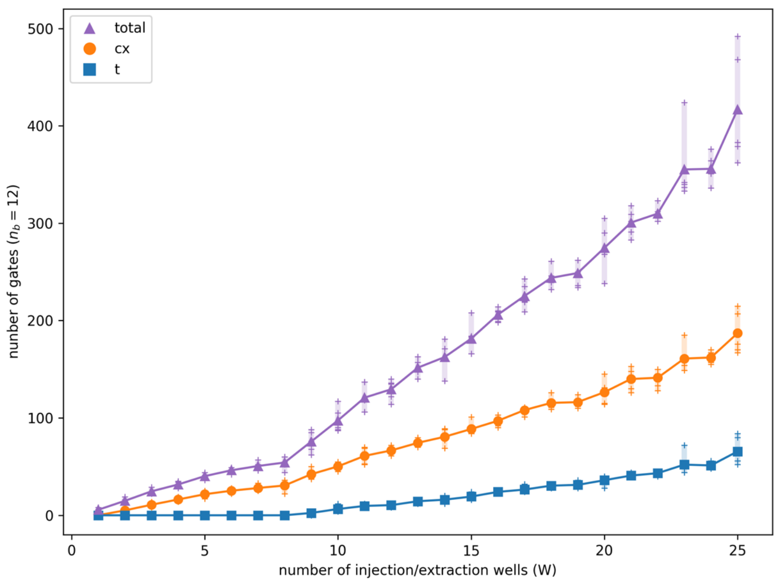

Figure 7. Total gate count (total), CNOT gate count (cx) and T gate count (t) of the quantum circuit preparing the state of a sparse vector **b**. In our system, the number of qubits is fixed at $n_b = 12$ while W, the number of nonzero entries in **b** corresponding to randomly-generated injection/extraction sites, increases from 1 to 25. For each value of W, gate counts from 5 random state samples are shown with + symbols, along with their average and bars extending from smallest to largest value. Theoretically, the state preparation method's total gate count scales as $O(Wn_b)$, further broken down as $O(Wn_b)$ CNOT gates and $O(W \log W + n_b)$ single-qubit gates.

Specifically, consider an r-register with at most n_b qubits, where n_b is—as above—the number of qubits in the b-register. If that r-register prepares a state such that $\langle r|x\rangle$ provides the sum of the pressures in a set of nodes, we can obtain $\langle r|x\rangle$—or at least $|\langle r|x\rangle|$—via either the Hadamard or swap tests and can then divide by the number of nodes that are in our desired region. Figures 8 and 9 illustrate the structure of circuits for extracting information with the swap and Hadamard tests, respectively. The remainder of this section describes swap test information extraction, Hadamard test information extraction, and a procedure for generating r states to obtain the average pressure for any user-specified region.

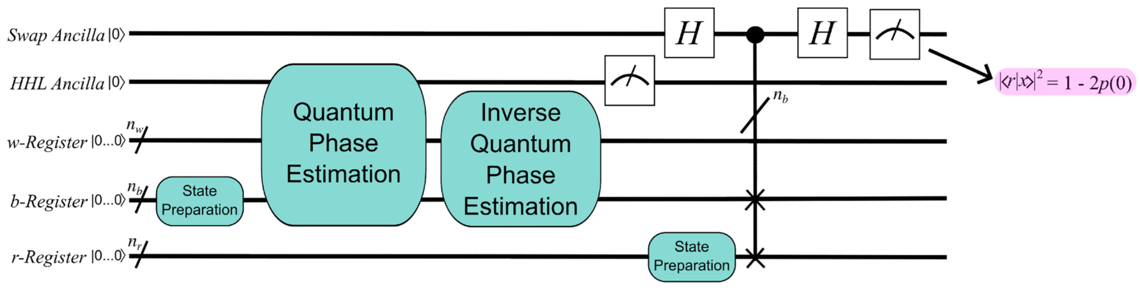

Figure 8. A schematic of the circuit structure for extracting average pressure using the swap test. Note that only the magnitude of the average pressure is extracted; sign is not.

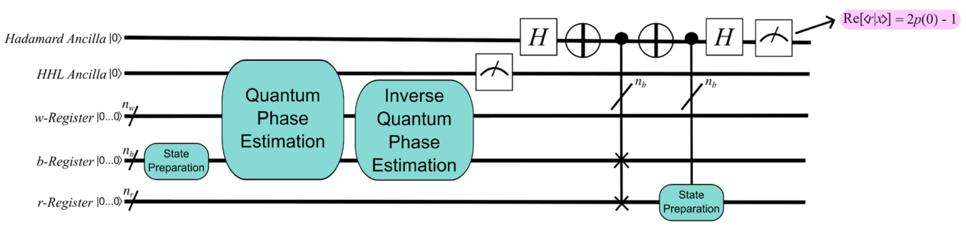

Figure 9. A schematic of the circuit structure for extracting average pressure using the Hadamard test. The circuit is more complicated than in Fig. 8, but it allows for extracting both the sign and magnitude of the average pressure.

First, the swap test approach. After preparing $|x\rangle$ using HHL, we apply the swap test using an additional ancilla qubit and an r-register that contains a state to provide $|\langle r|x\rangle| = \sum_{i\in \text{user\_elems}} x_i$, where `user_elems` is the set of node indices in the region for which average pressure should be computed. By determining the probability of measuring the swap test ancilla as zero, we can use a classical computer to obtain the unsigned average pressure as follows: Use $p(0)$ to obtain $|\langle r|x\rangle|^2 = 1 - 2p(0)$, take the square root of $|\langle r|x\rangle|^2$, and plug into $\text{avg\_press} = \frac{|\langle r|x\rangle|}{|\text{user\_elems}|} = \frac{\sum_{i\in \text{user\_elems}} x_i}{|\text{user\_elems}|}$. The swap test itself (not including the resources required for building the r-register, which are analyzed below) requires only one ancilla, two Hadamard gates, and n_r controlled-swap gates, where n_r is the number of qubits required for the r-register. Consequently, if we can build the r-register state efficiently, this is an acceptable approach to computing the average pressure in a specified region. It is also worth noting that all of these gates are applicable to near-term devices, and the swap test's complexity of $O\left(1/\epsilon^2\right)$ for a user-desired error, ϵ, is widely recognized and applied as reasonable for the near-term era, as well[8]. So, although HHL is not yet realistic for near-term devices, this swap test approach should be, and it would certainly be for fault-tolerant machines.

Second, the Hadamard test: we first prepare $|x\rangle$ using HHL, and then use an additional ancilla qubit and an r-register to prepare $|\langle r|x\rangle| = \sum_{i\in \text{user\_elems}} x_i$, where `user_elems` is as defined above. Not counting the resources required for building the r-register, this requires two Hadamard gates, two Pauli-$\mathbf{X}$ gates, and at most n_r controlled-swap gates. It is worth clarifying how the controlled gates work: the controlled-swap gates transfer the components of $|x\rangle$ that will be used in the inner product computation to the r-register, and these controlled-swaps thus replace the controlled-U_ϕ gate in Subfigure b of Fig. 3. The controlled-state preparation gate then prepares the r-register, as described below. So, to obtain the average pressure, we compute the probability of measuring the Hadamard test ancilla as zero, and use it to classically compute $\text{Re}[\langle r|x\rangle] = 2p(0) - 1$. We then divide by the number of desired nodes to find $\text{avg\_press} = \frac{\text{Re}[\langle r|x\rangle]}{|\text{user\_elems}|} = \frac{\sum_{i\in \text{user\_elems}} x_i}{|\text{user\_elems}|}$.

Building the *r* register

Using either the swap or Hadamard tests requires an r-register that properly extracts sums of equally-weighted node values. For the remainder of this section, we will illustrate the r-register-building process for the swap test, but using the Hadamard test would require only two modifications. First, instead of computing $|\langle r|x\rangle|^2$ using the probability that the ancilla is zero, we would compute $\text{Re}[\langle r|x\rangle]$, which does not require taking a square root. Second, while the r-register preparation for the swap test does not require controlled gates, each of the gates used to build the r-register for the Hadamard test would need to be controlled.

To build the r-register, we must determine both the number of qubits, n_r, and what gates need to be applied to these qubits such that $|\langle r|x\rangle|^2$ is a sum of the form $|\langle r|x\rangle| = \xi \sum_{i\in \text{user\_elems}} x_i$. If the quantum computer can provide a $|\langle r|x\rangle|^2$ with that form, then, as long as we know ξ, we can use the few classical computations described above to compute the average pressure in the set of nodes specified by `user_elems`. It is worth noting that our approach does not treat the initial state of the qubits in the r-register as a variable to be determined; the initial state of each r-register qubit is always the fiduciary state, $|0\rangle$.

While there is a general procedure for building r-registers that obtain the average pressure in an arbitrarily-specified set of discretized nodes, special cases offer simpler methods that have fewer steps and that aid understanding of the general process. Therefore, we first consider the special cases of an entire row, an entire column, and an entire diagonal of discretized nodes using the example region of Fig. 10, which is the simplest possible discretized region that satisfies the constraints of being square and having a number of nodes that is a power of two. It corresponds to a b-register of $n_b = 2$ qubits, and after applying the HHL algorithm, the final states of those qubits are $x_1|0\rangle + x_2|1\rangle$ and $x_3|0\rangle + x_4|1\rangle$, where $\mathbf{x} = \gamma \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$ with a normalization constant γ. Because two nodes' worth of information are stored in each qubit, we will term a 'pair' of nodes to be any whose information is stored on the same qubit. Thus, using the node-indexing scheme of Fig. 10, we have pairs x_1, x_2 and x_3, x_4. When both nodes in a pair are included in the desired average pressure region, we will term that group of nodes 'paired nodes.' When a node is included in the desired region without its paired node, we will term that node a single node. Nodes with even indices will be termed 'even-indexed,' and vice versa for nodes with odd indices.

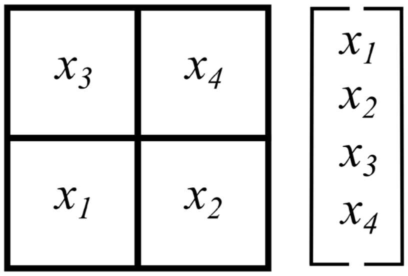

Figure 10. A schematic 2×2 grid illustrating a discretized region with four nodes, some combination of which holds the average pressure we seek to compute.

The discretized region of Fig. 10 has ten possible average pressures: the average pressure from a single node (either x_1, x_2, x_3, or x_4), from row $x_1 + x_2$, from row $x_3 + x_4$, from column $x_1 + x_3$, from column $x_2 + x_4$, from diagonal $x_1 + x_4$, and from diagonal $x_2 + x_3$. First, consider the case of a single node; the 'average' pressure of a single node is, of course, simply the pressure in that node. Consequently, an r-register-generation procedure for obtaining the value in a single node has little utility. Not only is the primary goal of extracting average pressure to avoid the costly procedure of sequentially extracting single elements from the solution, but it would also likely be simpler to just measure qubits of interest if the goal were to extract a few nodes' worth of individual pressures. Therefore, we will not further discuss the single node case.

Second, consider the rows; both cases require that the r-register prepare a state such that that $|\langle r|x\rangle| = r_i x_i + r_{i+1}x_{i+1}$ can be simplified to $\xi(x_i + x_{i+1})$, for $\xi = r_i = r_{i+1}$ and row nodes x_i and x_{i+1}. Note that rows are thus comprised of only paired nodes, and applying a Hadamard gate to each qubit in the r-register lends such an equal ξ coefficient to both nodes in a pair. So, to build the r-register for an entire row, add a qubit for each pair of nodes in the row, and apply a Hadamard gate to each qubit. This allows for computing the pressure in, say, the row with x_1 and x_2 as follows:

$$\begin{aligned} |\langle r|x\rangle|^2 &= \left(\frac{1}{\sqrt{2}}(x_1 + x_2)\right)^2 \\ \Rightarrow \text{avg\_pressure} &= \sqrt{2}\frac{|\langle r|x\rangle|}{2}. \end{aligned} \tag{1}$$

Consequently, the r-register preparation circuit for a row requires one qubit for every pair of nodes in the row and one Hadamard gate for every pair of nodes in the row.

Second, consider the case of a column, which has two subcases. The first subcase has columns with only odd-index nodes, and the second has columns with only even-index nodes; there will never be paired nodes in a column. For the even-index subcase, the r-register must use Pauli-**X** gates to flip the state of the r-register qubits from the initial state of $|0\rangle$ to a state of $|1\rangle$, such that a coefficient of 0 will be multiplied with each of the odd-indexed x values in a given qubit pair, and a coefficient of 1 will be multiplied with each of the even-indexed x values in the same. In the odd-index subcase, although we need one r-register qubit per desired node in the average pressure computation, that qubit need not have any gates on it. This is because every qubit in the r-register is initialized to the state $|0\rangle$, so a coefficient of 1 will already be multiplied with each of the odd-indexed x values whose average we seek. Consequently, in the column case, we need one qubit for every node in the desired region and one Pauli-**X** gate for every even-indexed node in the same. Then, the pressure in column x_1 and x_3 could be computed as

$$\begin{aligned} |\langle r|x\rangle|^2 &= ((1)x_1 + (1)x_3)^2 \\ \Rightarrow \text{avg\_pressure} &= \frac{|\langle r|x\rangle|}{2}. \end{aligned} \tag{2}$$

Third is the full diagonal case, which is functionally-equivalent to that of the column case, because a discretized diagonal will never involve paired nodes. Consequently, the r-register will again be comprised of one qubit per desired node in the average-pressure region, with one Pauli-**X** gate on each r-register qubit being swapped with an even node. Figure 11 summarizes the r-register generation procedures thus far.

We can extend these patterns to the more general case of arbitrarily-selected nodes in which to obtain the average pressure. We use the same patterns of applying a Hadamard gate on r-register qubits with paired node indices and a Pauli-**X** gate on r-register qubits with single, even-indexed nodes in the desired region. But now, obtaining average pressure in some sets of nodes will require more than one swap or Hadamard test. Specifically, in regions that contain both paired and single nodes, we will need two swap tests or two Hadamard tests: one test for paired nodes will result in a value with a coefficient of $\frac{1}{\sqrt{2}}$, while the other test for single nodes will produce a value with a coefficient of one. We can then compute average pressure by multiplying the result of the test for

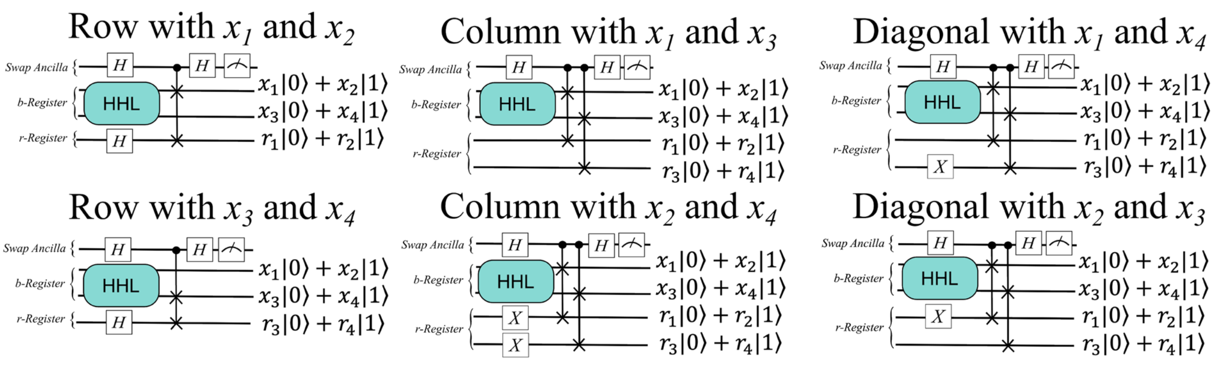
Row with x_1 and x_2
Swap Ancilla
b-Register
r-Register
HHL
H
$x_1|0\rangle + x_2|1\rangle$
$x_3|0\rangle + x_4|1\rangle$
$r_1|0\rangle + r_2|1\rangle$
Column with x_1 and x_3
$r_3|0\rangle + r_4|1\rangle$
Diagonal with x_1 and x_4
X
Row with x_3 and x_4
Column with x_2 and x_4
Diagonal with x_2 and x_3

Figure 11. Schematic of the HHL and swap test circuits for extracting the average pressure from a full row, column, and diagonal of a discretized two-by-two region.

paired nodes by $\sqrt{2}$ to remove the coefficient, adding the resulting sum with the sum that results from the second test for single nodes, and dividing by the total number of nodes in the desired region.

To clarify, consider a slightly more complicated example where we have regions that are not automatically an entire row, column, or diagonal, as they were in the case of Fig. 10. Specifically, consider the 4×4 discretized region in Subfigure a of Fig. 12, and suppose that the nodes circled in yellow represent nodes through which a two-pronged pitchfork fracture runs. Obtaining the average pressure in this fracture requires two circuits as illustrated in Subfigure b of Fig. 12. The first computes the sum of pressures in nodes that are single, while the second computes the sum of pressures in nodes that are pairs. Again, the difference between the circuit outputs is that paired nodes require multiplying by a known factor of $\sqrt{2}$ to remove the $\frac{1}{\sqrt{2}}$ term resulting from the Hadamard gates in the r-register. After obtaining the sums, we can classically compute the average by dividing by the known number of nodes in the desired region.

Because building r-registers for arbitrarily-specified regions uses the same procedures as for building the registers for entire rows, columns, or diagonals, the general r-register complexity also aligns with the complexities of those situations. Specifically, the circuit for determining the average pressure of single nodes requires an r-register with one qubit per desired node and one Pauli-$\mathbf{X}$ gate per even-indexed node, while the circuit for determining the average pressure of paired nodes requires an r-register with one qubit per pair of nodes and one Hadamard gate per pair of nodes.

Overall complexities: qubit count, gate count, and additive error for average pressure extraction

The complexity in terms of qubit and gate count for the HHL algorithm is well-understood[2,24,26], and is taken to be acceptable, if not for near-term machines, then for fault-tolerant or error-mitigated machines of the future[43]. Therefore, to assess whether our information extraction approach is appropriately efficient, we assess its worst-case complexity in terms of qubit and gate counts that are already required for the HHL portion of the circuit. We also comment on the relationship between the additive errors introduced during HHL and the information extraction portions of the computation.

First, qubit count: both the swap and Hadamard tests require only $n_r + 1$ additional qubits, so we need to determine how n_r scales with the size of the circuit required for HHL. Whether we have a region with all single nodes or all paired nodes, the maximum number of qubits in the r-register is n_b. Furthermore, the total number of r-register qubits cannot be greater than n_b even across two circuits for single and paired nodes, because if it were, at least one node would be double-counted in the resulting average. Therefore, n_r is $O(n_b)$, and the overall number of additional qubits for the swap/Hadamard test is $O(n_b) + 1 = O(n_b)$. As n_b qubits is considered reasonable in the HHL algorithm, an additional $O(n_b)$ is a reasonable worst-case-scenario for information extraction.

Second, gate count; both the swap and Hadamard tests add a few additional gates (two for the swap test and four for the Hadamard test) in addition to the controlled-swap gates and the gates required to construct the r-register. Because there can be at most n_b qubits in each r-register, there can be at most n_b controlled-swap gates, meaning the worst-case-scenario for controlled-swap gates is $O(n_b)$. Additionally, we found that the maximum number of gates required in the r-register is also $O(n_b)$ because at most one additional gate (either a Hadamard or a Pauli-$\mathbf{X}$) is required per qubit in the r-register. Consequently, the overall number of gates is $O(n_b)$ (either

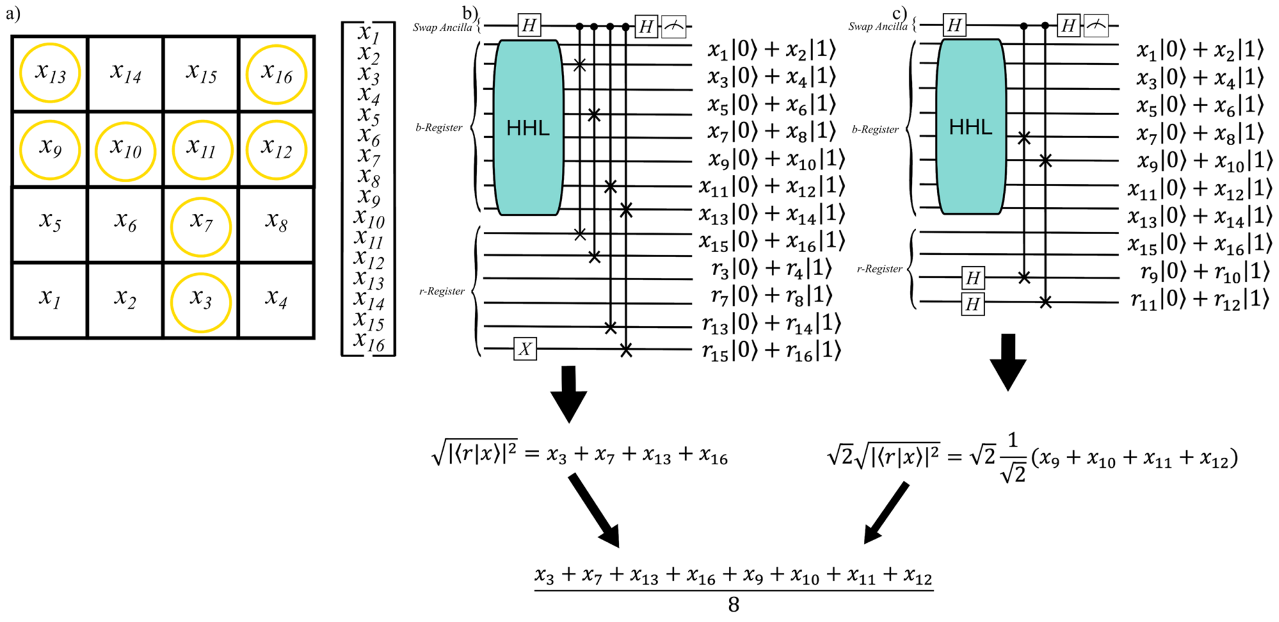
a)
b)
c)
Swap Ancilla
b-Register
r-Register
HHL
H
X
$x_1|0\rangle + x_2|1\rangle$
$x_3|0\rangle + x_4|1\rangle$
$x_5|0\rangle + x_6|1\rangle$
$x_7|0\rangle + x_8|1\rangle$
$x_9|0\rangle + x_{10}|1\rangle$
$x_{11}|0\rangle + x_{12}|1\rangle$
$x_{13}|0\rangle + x_{14}|1\rangle$
$x_{15}|0\rangle + x_{16}|1\rangle$
$r_3|0\rangle + r_4|1\rangle$
$r_7|0\rangle + r_8|1\rangle$
$r_{13}|0\rangle + r_{14}|1\rangle$
$r_{15}|0\rangle + r_{16}|1\rangle$
$r_9|0\rangle + r_{10}|1\rangle$
$r_{11}|0\rangle + r_{12}|1\rangle$
$\sqrt{|\langle r|x\rangle|^2} = x_3 + x_7 + x_{13} + x_{16}$
$\sqrt{2}\sqrt{|\langle r|x\rangle|^2} = \sqrt{2}\frac{1}{\sqrt{2}}(x_9 + x_{10} + x_{11} + x_{12})$
$\frac{x_3 + x_7 + x_{13} + x_{16} + x_9 + x_{10} + x_{11} + x_{12}}{8}$

Figure 12. Subfigure (**a**) is a 4×4 discretized region with a pitchfork fracture marked by yellow circles. Subfigure (**b**) illustrates the two circuits needed to compute the average pressure in a series of arbitrarily-selected nodes, by building r-registers as described above.

$2 + O(n_b) + O(n_b)$ or $4 + O(n_b) + O(n_b)$), which is—by the same reasoning as above—considered efficient for purposes of HHL and therefore also for purposes of information extraction.

We note that a more interesting complexity issue is raised by the two circuits required to obtain average pressure in a region with both single and paired nodes. This requires twice as many executions of the entire HHL procedure, which—particularly in the contemporary quantum hardware ecosystem—may add up to substantial runtime expense. Therefore, it is worth noting that we can avoid using two circuits to compute the average pressure in an arbitrary region, if we are willing to select only single nodes to represent our desired region. For example, consider again the fracture in Subfigure a of Fig. 12; if we removed nodes 10 and 12—or nodes 9 and 11 or 10 and 11 or 9 and 12—then we could use a single circuit with the structure of Fig. 12's Subfigure b to compute the average pressure in a region comprised of only single nodes. While this does not change the overall possible complexity of that circuit, it does reduce circuit executions to half of what would otherwise be required, at an 'accuracy cost' of removing just a few nodes from our desired average pressure region.

Third and finally, error propagation. The HHL algorithm prepares a solution $|x\rangle$ with an additive error of ϵ_{HHL}, where ϵ_{HHL} is a user-defined parameter that will affect the QPE and inverse QPE subroutines[2,24]. Therefore, upon completion of HHL, the b-register will be in a state with every element of $|x\rangle$ plus at most ϵ_{HHL}. (For simplicity, we assume that every element of the solution vector has the same error, which could be interpreted as the maximum of individual element errors, and which lends a state $|x + \epsilon_{HHL}\rangle$). The swap and Hadamard tests both introduce a second additive error, $\epsilon_{extract}$, such that the inner product measured has a value of at most $\epsilon_{extract}$ added to it.

Consider the structure of our r-registers, which weight the elements whose average we seek *equally*. This means that each node in the average will introduce an error of $1 * \epsilon_{HHL}$, for a total of $n * \epsilon_{HHL}$, where n is the number of desired nodes in the average computation. Then, an error of $\epsilon_{extract}$ will be added to the entire inner product—including the $n * \epsilon_{HHL}$ term—lending an overall error of $n * \epsilon_{HHL} + \epsilon_{extract}$.

Because both additive errors are specified by the user and relate to the number of circuit executions required, the user can determine the amount of error in the average pressure by choosing a number of circuit executions they want to perform and using the complexities of HHL and the swap test ($O(\log(N)\kappa s^2/\epsilon_{HHL})$ and $O(1/\epsilon_{extract}^2)$, respectively) to 'back out' approximate values of ϵ_{HHL} and $\epsilon_{extract}$. Then, the overall additive error in the average pressure solution is $n * \epsilon_{HHL} + \epsilon_{extract}$, which is $O(\epsilon)$, for $\epsilon = \max\{\epsilon_{HHL}, \epsilon_{extract}\}$. Assuming that ϵ_{HHL} and $\epsilon_{extract}$ are chosen to have similar magnitudes, our approach to efficient information extraction thus has an overall error that is the same order as the additive error of HHL itself, meaning our information extraction procedure is consistent with errors that are widely viewed as acceptable.

Conclusion

This work introduces efficient methods for both preparing the b-register for and extracting useful information from a quantum computer that solves a linear system representing a geologic fracture network problem. Both approaches have reasonable complexities, as they require $O(n_b)$ additional qubits and either $O(Wn_b)$ or $O(n_b)$ additional gates, where W is the number of injection/extraction wells. Furthermore, both approaches utilize only straightforward gate types that are applicable to near-term hardware[8], meaning they should be viable on improved hardware of the future. Consequently, this work—combined with the previous results in Refs.[17] and[19]—provides answers to three of the four "fine print" considerations for using quantum algorithms to solve geologic fracture network problems.

Of course, these works do not close inquiry into the issue of solving geologic fracture network problems with quantum algorithms, and there are several avenues for further work. First, there are likely other solutions to state preparation and information extraction considerations, some of which may lend themselves more readily to quantum linear solvers that are more applicable than HHL for today's NISQ-era devices. Second, empirical study of this work's approaches—particularly in concert with the core of HHL itself—remains to be done, particularly given advances in quantum hardware offering larger and more error-resistant machines[34–36]. Third and relatedly, error correction and/or mitigation techniques are likely required to make the core of HHL appropriate for geologic fracture network problems of any scale, making investigation of such techniques yet another avenue for further study. Nonetheless, this article and those it complements provide a foundational answer to the question of using quantum computing in the geologic fracture realm: it can be done and, especially with the rapid ascent of ever-improving quantum hardware, has the promise to revolutionize the modelling of geologic fracture networks.

Data availibility

The codes—which generate the considered data—are available at https://github.com/OrchardLANL/DPFEHM.jl/tree/master/examples/fracture_networks_for_qc.

Received: 3 October 2023; Accepted: 23 January 2024
Published online: 13 February 2024

References

1. Boyd, S. & Vandenberghe, L. *Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares* (Cambridge University Press, 2018).
2. Harrow, A. W., Hassidim, A. & Lloyd, S. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.* **103**, 150502. https://doi.org/10.1103/PhysRevLett.103.150502 (2009).
3. Ambainis, A. Variable time amplitude amplification and quantum algorithms for linear algebra problems. In *STACS'12 (29th Symposium on Theoretical Aspects of Computer Science), vol. 14*636–647 (LIPIcs, 2012).
4. Childs, A. M., Kothari, R. & Somma, R. D. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM J. Comput.* **46**, 1920–1950. https://doi.org/10.1137/16M1087072 (2017).
5. Wossnig, L., Zhao, Z. & Prakash, A. Quantum linear system algorithm for dense matrices. *Phys. Rev. Lett.* **120**, 050502 (2018).

6. Subaşı, Y., Somma, R. D. & Orsucci, D. Quantum algorithms for systems of linear equations inspired by adiabatic quantum computing. *Phys. Rev. Lett.* **122**, 060504 (2019).
7. Costa, P. *et al.* Optimal scaling quantum linear systems solver via discrete adiabatic theorem. arXiv:2111.08152 (2021).
8. O'Malley, D., Subasi, Y., Golden, J., Lowrie, R. & Eidenbenz, S. A near-term quantum algorithm for solving linear systems of equations based on the woodbury identity. arXiv:2205.00645. https://doi.org/10.48550/arXiv.2205.00645 (2022).
9. Aaronson, S. Read the fine print. *Nat. Phys.* **11**, 291–293. https://doi.org/10.1038/nphys3272 (2015).
10. Nielsen, M. A. & Chuang, I. L. *Quantum Computation and Quantum Information* (Cambridge University Press, 2010).
11. Hyman, J. D. *et al.* dfnworks: A discrete fracture network framework for modeling subsurface flow and transport. *Comput. Geosci.* **84**, 10–19 (2015).
12. Mills, R. T., Lu, C., Lichtner, P. C. & Hammond, G. E. Simulating subsurface flow and transport on ultrascale computers using pflotran. *J. Phys. Conf. Ser.* **78**, 012051 (2007).
13. O'Malley, D. *et al.* Where does water go during hydraulic fracturing?. *Groundwater* **54**, 488–497 (2016).
14. O'Malley, D. An approach to quantum-computational hydrologic inverse analysis. *Sci. Rep.* **8**, 6919 (2018).
15. Henderson, J. M., O'Malley, D. & Viswanathan, H. S. Interrogating the performance of quantum annealing for the solution of steady-state subsurface flow. In *2021 IEEE High Performance Extreme Computing Conference (HPEC)* 1–6 (IEEE, 2021).
16. Sahimi, M. & Tahmasebi, P. The potential of quantum computing for geoscience. *Transp. Porous Media* **145**, 367–387 (2022).
17. Henderson, J. M. *et al.* Quantum algorithms for geologic fracture networks. *Sci. Rep.* **13**, 2906 (2023).
18. Greer, S. & O'Malley, D. Early steps towards practical subsurface computations with quantum computing. *Front. Comput. Sci.* **5**, 1235784. https://doi.org/10.3389/fcomp.2023.1235784 (2023).
19. Golden, J., O'Malley, D. & Viswanathan, H. Quantum computing and preconditioners for hydrological linear systems. *Sci. Rep.* **12**, 145. https://doi.org/10.1038/s41598-022-25727-9 (2022).
20. Fountain, A. G., Jacobel, R. W., Schlichting, R. & Jansson, P. Fractures as the main pathways of water flow in temperate glaciers. *Nature* **433**, 618–621 (2005).
21. Davies, J. H. The role of hydraulic fractures and intermediate-depth earthquakes in generating subduction-zone magmatism. *Nature* **398**, 142–145 (1999).
22. Viswanathan, H. S. *et al.* From fluid flow to coupled processes in fractured rock: Recent advances and new frontiers. *Rev. Geophys.* **60**, e2021RG000744 (2022).
23. Laubach, S. E. *et al.* The role of chemistry in fracture pattern development and opportunities to advance interpretations of geological materials. *Rev. Geophys.* **57**, 1065–1111 (2019).
24. Chakrabarti, S. & Wu, X. *Quantum Algorithms for Solving Linear Systems of Equations* (Springer, 2018).
25. Morrell Jr, H. J., Zaman, A. & Wong, H. Y. Step-by-step hhl algorithm walkthrough to enhance the understanding of critical quantum computing concepts. arXiv:2108.09004. https://doi.org/10.48550/arXiv.2108.09004 (2021).
26. Duan, B., Yuan, J., Yu, C.-H., Huang, J. & Hsieh, C.-Y. A survey on hhl algorithm: From theory to application in quantum machine learning. *Phys. Lett. A* **384**, 126595 (2020).
27. Scherer, A. *et al.* Concrete resource analysis of the quantum linear-system algorithm used to compute the electromagnetic scattering cross section of a 2d target. *Quant. Inf. Process.* **16**, 1–65 (2017).
28. Zheng, Y. *et al.* Solving systems of linear equations with a superconducting quantum processor. *Phys. Rev. Lett.* **118**, 210504. https://doi.org/10.1103/PhysRevLett.118.210504 (2017).
29. Lee, Y., Joo, J. & Lee, S. Hybrid quantum linear equation algorithm and its experimental test on IBM quantum experience. *Sci. Rep.* **9**, 1–12. https://doi.org/10.1038/s41598-019-41324-9 (2019).
30. Pan, J. *et al.* Experimental realization of quantum algorithm for solving linear systems of equations. *Phys. Rev. A* **89**, 022313. https://doi.org/10.1103/PhysRevA.89.022313 (2014).
31. Cai, X.-D. *et al.* Experimental quantum computing to solve systems of linear equations. *Phys. Rev. Lett.* **110**, 230501. https://doi.org/10.1103/PhysRevLett.110.230501 (2013).
32. Barz, S. *et al.* A two-qubit photonic quantum processor and its application to solving systems of linear equations. *Sci. Rep.* **4**, 1–5. https://doi.org/10.1038/srep06115 (2014).
33. Wen, J. *et al.* Experimental realization of quantum algorithms for a linear system inspired by adiabatic quantum computing. *Phys. Rev. A* **99**, 012320. https://doi.org/10.1103/PhysRevA.99.012320 (2019).
34. Choi, C. An IBM quantum computer will soon pass the 1,000-qubit mark. *IEEE Spectr.* **24**, 14589 (2022).
35. Castelvecchi, D. IBM releases first-ever 1,000-qubit quantum chip. *Nature* **624**, 238–238 (2023).
36. Atom Computing. Quantum startup Atom Computing first to exceed 1,000 qubits. https://atom-computing.com/quantum-startup-atom-computing-first-to-exceed-1000-qubits/ (2023).
37. Perelshtein, M. *et al.* Solving large-scale linear systems of equations by a quantum hybrid algorithm. *Ann. Phys.* **534**, 2200082. https://doi.org/10.1002/andp.202200082 (2022).
38. Baskaran, N. *et al.* Adapting the harrow-hassidim-lloyd algorithm to quantum many-body theory. *Phys. Rev. Res.* **5**, 043113 (2023).
39. Barenco, A. *et al.* Stabilization of quantum computations by symmetrization. *SIAM J. Comput.* **26**, 1541–1557. https://doi.org/10.1137/S0097539796302452 (1997).
40. Buhrman, H., Cleve, R., Watrous, J. & De Wolf, R. Quantum fingerprinting. *Phys. Rev. Lett.* **87**, 167902. https://doi.org/10.1103/PhysRevLett.87.167902 (2001).
41. Cleve, R., Ekert, A., Macchiavello, C. & Mosca, M. Quantum algorithms revisited. *Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci.* **454**, 339–354. https://doi.org/10.1098/rspa.1998.0164 (1998).
42. Gleinig, N. & Hoefler, T. An efficient algorithm for sparse quantum state preparation. In *2021 58th ACM/IEEE Design Automation Conference (DAC)* 433–438. https://doi.org/10.1109/DAC18074.2021.9586240 (2021).
43. Kim, Y. *et al.* Evidence for the utility of quantum computing before fault tolerance. *Nature* **618**, 500–505. https://doi.org/10.1038/s41586-023-06096-3 (2023).

Acknowledgements
JMH and DO gratefully acknowledge support from the Department of Energy, Office of Science, Office of Basic Energy Sciences, Geoscience Research program under Award Number (LANLE3W1). JMH, JKG, and DO gratefully acknowledge support from Los Alamos National Laboratory's Laboratory Directed Research & Development project 20220077ER.

Author contributions
J.K.G. and D.O. designed the project. J.K., A.G.P., and D.O. designed the approach for efficient state preparation, and J.K. implemented the associated codes. J.M.H., J.K.G., and D.O. designed the approach for efficient information extraction. J.M.H. and J.K. prepared the first draft of the manuscript. All authors reviewed and revised the manuscript.

Competing Interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to J.M.H.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.