The Power of Block-Encoded Matrix Powers: Improved Regression Techniques via Faster Hamiltonian Simulation

Shantanav Chakraborty
QuIC, Université libre de Bruxelles, Belgium
shchakra@ulb.ac.be

András Gilyén
QuSoft/CWI, The Netherlands
gilyen@cwi.nl

Stacey Jeffery
QuSoft/CWI, The Netherlands
jeffery@cwi.nl

Abstract

We apply the framework of block-encodings, introduced by Low and Chuang (under the name standard-form), to the study of quantum machine learning algorithms and derive general results that are applicable to a variety of input models, including sparse matrix oracles and matrices stored in a data structure. We develop several tools within the block-encoding framework, such as singular value estimation of a block-encoded matrix, and quantum linear system solvers using block-encodings. The presented results give new techniques for Hamiltonian simulation of non-sparse matrices, which could be relevant for certain quantum chemistry applications, and which in turn imply an exponential improvement in the dependence on precision in quantum linear systems solvers for non-sparse matrices.

In addition, we develop a technique of variable-time amplitude estimation, based on Ambainis’ variable-time amplitude amplification technique, which we are also able to apply within the framework.

As applications, we design the following algorithms: (1) a quantum algorithm for the quantum weighted least squares problem, exhibiting a 6-th power improvement in the dependence on the condition number and an exponential improvement in the dependence on the precision over the previous best algorithm of Kerenidis and Prakash; (2) the first quantum algorithm for the quantum generalized least squares problem; and (3) quantum algorithms for estimating electrical-network quantities, including effective resistance and dissipated power, improving upon previous work.

2012 ACM Subject Classification Theory of computation → Quantum computation theory

Keywords and phrases Quantum algorithms, Hamiltonian simulation, Quantum machine learning

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.33

Category Track A: Algorithms, Complexity and Games

Related Version Full version of this submission is available at: https://arxiv.org/abs/1804.01973.

Acknowledgements The authors are grateful to Iordanis Kerenidis, Anupam Prakash and Michael Walter for useful discussions. SC is supported by the Belgian Fonds de la Recherche Scientifique - FNRS under grants no F.4515.16 (QUICTIME) and R.50.05.18.F (QuantAlgo). AG is supported by ERC Consolidator Grant QPROGRESS. SJ is supported by an NWO WISE Grant and NWO Veni Innovative Research Grant under project number 639.021.752.
A rapidly growing and important class of quantum algorithms are those that use Hamiltonian simulation subroutines to solve linear algebraic problems, many with potential applications to machine learning. This subfield began with the HHL algorithm, due to Harrow, Hassidim and Lloyd [18], which solves the quantum linear system problem (QLS problem). In this problem, the input consists of a matrix $A \in \mathbb{R}^{N \times N}$ and a vector $\vec{b} \in \mathbb{R}^N$, in some specified format, and the algorithm should output a quantum state proportional to $\sum_{i=1}^{N} x_i |i\rangle$, where $\vec{x} = A^{-1}\vec{b}$.

The format in which the input is presented is of crucial importance. For a sparse $A$, given an efficient algorithm to query the $i$-th non-zero entry of the $j$-th row of $A$, the HHL algorithm and its subsequent improvements [2, 14] can solve the QLS problem in complexity that depends poly-logarithmically on $N$. Here, if $A$ were given naively as a list of all its entries, it would generally take time proportionally to $N^2$ just to read the input. We will refer to the model of accessing $A$, in which we can query the $i$-th non-zero entry of the $j$-th row, as the sparse-access input model.$^1$

In [19] and [20], Kerenidis and Prakash consider several linear algebraic problems in a different input model. They assume that data has been collected and stored in some carefully chosen data structure in advance. If the data is described by an arbitrary $N \times N$ matrix, then of course, this collection will take time at least $N^2$ (or, if the matrix is sparse, at least the number of non-zero entries). However, processing the data, given such a data structure, is significantly cheaper, depending only poly-logarithmically on $N$. Kerenidis and Prakash describe a data structure that, when stored in quantum-random-access read-only memory (QROM)$^2$, allows for the preparation of a superposition over $N$ data points in complexity poly-logarithmic in $N$. We call this the quantum data structure input model and discuss it more in Section 2.2. Although in some applications it might be too much to ask for the data to be presented in such a structure, one advantage of this input model is that it is not restricted to sparse matrices. This result can potentially also be useful for some quantum chemistry applications, since a recent proposal of Babbush et al. [4] uses a database of all Hamiltonian terms in order to simulate the electronic structure.

The HHL algorithm and its variants and several other applications are based on techniques from Hamiltonian simulation. Given a Hermitian matrix $H$ and an input state $|\psi\rangle$, the Hamiltonian simulation problem is to simulate the unitary $e^{iH}$ on $|\psi\rangle$ for some time $t$. Most work in this area has considered the sparse-access input model [22, 1, 6, 7, 5, 8, 12, 13, 15, 26, 31, 9, 25, 10], but recent work of Low and Chuang [24] has considered a different model, which we call the block-encoding framework.$^3$

### The block-encoding framework

A block-encoding of a matrix $A \in \mathbb{C}^{N \times N}$ is a unitary $U$ such that the top left block of $U$ is equal to $A/\alpha$ for some normalizing constant $\alpha \geq ||A||$:

$$U = \left( \begin{array}{ccc} A/\alpha & \cdots \\ \vdots & \ddots & \vdots \\ \alpha & \cdots & 1 \end{array} \right),$$

i.e. $(0^{\otimes a} \otimes I)U((0)^{\otimes a} \otimes |\psi\rangle) = A/\alpha$. In other words, for some $a$, for any state $|\psi\rangle$ of appropriate dimension, $\alpha((0^{\otimes a} \otimes I)U((0)^{\otimes a} \otimes |\psi\rangle) = A|\psi\rangle$.

---

$^1$ If the matrix is not symmetric (or Hermitian) we also assume access to its transpose in a similar fashion.

$^2$ This refers to memory that is only required to store classical (non-superposition) data, but can be addressed in superposition.

$^3$ Low and Chuang call this input model standard form.
Such an encoding is useful if $U$ can be implemented efficiently. In that case, $U$, combined with amplitude amplification, can be used to generate the state $A|\psi\rangle/\|A|\psi\rangle\|$ given a circuit for generating $|\psi\rangle$. The main motivation for using block-encodings is that Low and Chuang showed [24] how to perform optimal Hamiltonian simulation given a block-encoded Hamiltonian $A$.

In Ref. [19], Kerenidis and Prakash implicitly prove that if an $N \times N$ matrix $A$ is given as a quantum data structure, then there is an $\varepsilon$-approximate block-encoding of $A$ that can be implemented in complexity $\text{polylog}(N/\varepsilon)$. This implies that all results about block-encodings – including Low and Chuang’s Hamiltonian simulation when the input is given as a block-encoding [24], and other techniques we develop in this paper – also apply to input presented in the quantum data structure model. This observation is the essential idea behind our applications. Implicit in work by Childs [13] is the fact that, given $A$ in the sparse-access input model, there is an $\varepsilon$-approximate block-encoding of $A$ that can be implemented in complexity $\text{polylog}(N/\varepsilon)$, so our results also apply to the sparse-access input model. In fact, the block-encoding framework unifies a number of possible input models, and also enables one to work with hybrid input models, where some matrices may come from purifications of density operators, whereas other input matrices may be accessed through sparse oracles or a quantum data structure. For a very recent overview of these general techniques see e.g. [16].

We demonstrate the elegance of the block-encoding framework by showing how to combine and modify block-encodings to build up new block-encodings, similar to building new algorithms from existing subroutines. For example, given block-encodings of $A$ and $B$, their product yields a block-encoding of $AB$. Given a block-encoding of a Hermitian $A$, it is possible to construct a block-encoding of $e^{iA}$, using which one can implement a block-encoding of $A^{-1}$. We present these techniques in Section 3.

To illustrate the elegance of the block-encoding framework, consider one of our applications: generalized least squares. This problem, defined in Section 4, requires that given inputs $X \in \mathbb{R}^{M \times N}$, $\Omega \in \mathbb{R}^{M \times M}$ and $\vec{y} \in \mathbb{R}^M$, we output a quantum state proportional to

$$\vec{\beta} = (X^T \Omega^{-1} X)^{-1} X^T \Omega^{-1} \vec{y}.$$

Given block-encodings of $X$ and $\Omega$, it is simple to combine them to get a block-encoding of $(X^T \Omega^{-1} X)^{-1} X^T \Omega^{-1}$, which can then be applied to a quantum state proportional to $\vec{y}$.

**Variable-time amplitude estimation.** A variable-stopping-time quantum algorithm is a quantum algorithm $A$ consisting of $m$ stages $A = A_m \ldots A_1$, where $A_j A_{j+1} \ldots A_1$ has complexity $t_j$, for $t_m > \cdots > t_1 > 0$. At each stage, a certain flag register, which we can think of as being initialized to a neutral symbol, may be marked as “good” in some branches of the superposition, or “bad” in some branches of the superposition, or left neutral. Each subsequent stage only acts non-trivially on those branches of the superposition in which the flag is not yet set to “good” or “bad”.

At the end of the algorithm, we would like to project onto that part of the final state in which the flag register is set to “good”. This is straightforward using amplitude amplification, however this approach may be vastly sub-optimal. If the algorithm terminates with amplitude $\sqrt{p_{\text{succ}}}$ on the “good” part of the state, then standard amplitude amplification requires that we run $1/\sqrt{p_{\text{succ}}}$ rounds, each of which requires us to run the full algorithm $A$ to generate its final state, costing $t_m/\sqrt{p_{\text{succ}}}$.

To see why this might be sub-optimal, suppose that after $A_1$, the amplitude on the part of the state in which the flag register is set to “bad” is already very high. Using amplitude amplification at this stage is very cheap, because we only have to incur the cost
$t_1$ of $A_1$ at each round, rather than running all of $A$. In [2], Ambainis showed that given a variable-stopping-time quantum algorithm, there exists an algorithm that approximates the “good” part of the algorithm’s final state in cost $\tilde{O} \left(t_m + \sqrt{\sum_{j=1}^{m} \frac{p_j}{p_{\text{succ}}} t_j^2}\right)^4$, where $p_j$ is the amplitude on the part of the state that is moved from neutral to “good” or “bad” during application of $A_j$ (intuitively, the probability that the algorithm stops at stage $j$).

While amplitude amplification can easily be modified to not only project a state onto its “good” part, but also return an estimate of $p_{\text{succ}}$ (i.e. the probability of measuring “good” given the output of $A$), this is not immediate in variable-time amplitude amplification. The main difficulty is that a variable-time amplification algorithm applies a lot of subsequent amplification phases, where in each amplification phase the precise amount of amplification is a priori unknown. We overcome this difficulty by separately estimating the amount of amplification in each phase with some additional precision and finally combining the separate estimates in order to get a multiplicative estimate of $p_{\text{succ}}$.

We prove rigorously in the full version of this paper [11], how to estimate the success probability of a variable-stopping-time quantum algorithm to within a multiplicative error of $\varepsilon$ in complexity

$$\tilde{O}\left(\frac{1}{\varepsilon} \left(t_m + \sqrt{\sum_{j=1}^{m} \frac{p_j}{p_{\text{succ}}} t_j^2}\right)\right).$$

Meanwhile we also derive some logarithmic improvements to the complexity of variable-time amplitude amplification.

**Applications.** We give several applications of the block-encoding framework and variable-time amplitude estimation.

We first present a quantum weighted least squares solver (WLS solver), which outputs a quantum state proportional to the optimal solution to a weighted least squares problem, when the input is given either in the quantum data structure model of Kerenidis and Prakash, or the sparse-access input model. We remark that the sparse-access input model is perhaps less appropriate to the setting of data analysis, where we cannot usually assume any special structure on the input data, however, since our algorithm is designed in the block-encoding framework, it works for either input model. Our quantum WLS solver improves the dependence on the condition number from $\kappa^6$ in [20] to $\kappa$, and the dependence on $\varepsilon$ from $1/\varepsilon$ to $\text{polylog}(1/\varepsilon)$.

We next present the first quantum generalized least squares solver (GLS solver), which outputs a quantum state proportional to the optimal solution to a generalized least squares problem. We again assume that the input is given in either the quantum data structure model or the sparse-access model. The complexity is again polynomial in $\log(1/\varepsilon)$ and in the condition numbers of the input matrices. We describe our WLS and GLS solvers in Section 4.

We build on the algorithms of Wang [29] to estimate effective resistance between two nodes of an electrical network and the power dissipated across a network when the input is given as a quantum data structure or in the sparse-access model. We estimate the norm of the output state of a certain linear system by applying the variable-time amplitude estimation algorithm. In the sparse-access model, we find that our algorithm outperforms Wang’s linear-system-based algorithm. In the quantum data structure model, our algorithms offer

---

4 We use the notation $\tilde{O}(f(x))$ to indicate $O(f(x)\text{polylog}(f(x)))$.
5 In the paper of Kerenidis and Prakash their $\kappa$ corresponds to our $\kappa^2$. 
a speedup whenever the maximum degree of an electrical network of \(n\) nodes is \(\Omega(n^{1/3})\). Our algorithms also have a speedup over the quantum walk based algorithm by Wang in certain regimes.

Throughout the article, the theorems, lemmas, and corollaries that are provided without a reference, are all rigorously proven in the full version of this paper [11].

Related Work. Independently of this work, recently, Wang and Wossnig [28] have also considered Hamiltonian simulation of a Hamiltonian given in the quantum data structure model, using quantum-walk based techniques from earlier work on Hamiltonian simulation [9]. Their algorithm’s complexity scales as \(\|A\|_1\) (which they upper bound by \(\sqrt{N}\)); whereas our Hamiltonian simulation results (Theorem 8), which follow from Low and Chuang’s block-Hamiltonian simulation result, have a complexity that depends poly-logarithmically on the dimension, \(N\). Instead, our complexity depends on the parameter \(\mu\), described below, which is also at most \(\sqrt{N}\).

2 Preliminaries

For \(A \in \mathbb{C}^{M \times N}\), define \(\overline{A} \in \mathbb{C}^{(M+N) \times (M+N)}\) by

\[
\overline{A} = \begin{bmatrix}
0 & A \\
A^\dagger & 0
\end{bmatrix}.
\]  

(1)

For many applications where we want to simulate \(A\), or a function of \(A\), it suffices to simulate \(A\). For \(A \in \mathbb{C}^{N \times N}\), we will let \(\|A\|\) denote the spectral norm and \(\|A\|_F\) the Frobenius norm. For \(A \in \mathbb{C}^{M \times N}\), let \(A_i\), denote the \(i\)-th row of \(A\), and define the following:

\begin{itemize}
  \item For \(q \in [0,1]\), \(s_q(A) = \max_{i \in [M]} \|A_i\|_q\) (the largest \(q\)-th norm of \(A\))
  \item For \(p \in [0,1]\), \(\mu_p(A) = \sqrt{s_{2p}(A)s_{2(1-p)}(A^T)}\)
  \item \(\sigma_{\min}(A) = \min\{\|A|u\| : |u\| = 1\}\) (the smallest non-zero singular value)
  \item \(\sigma_{\max}(A) = \max\{\|A|u\| : |u\| = 1\}\) (the largest singular value)
  \item \(\|A\| = \|\overline{A}\| = \sigma_{\max}(A)\)
\end{itemize}

For \(A \in \mathbb{C}^{M \times N}\) with singular value decomposition \(A = \sum_i \sigma_i |u_i\rangle \langle u_i|\), we define the Moore-Penrose pseudoinverse of \(A\) by \(A^+ = \sum_i \sigma_i^{-1} |v_i\rangle \langle u_i|\). For a matrix \(A\), we let \(A^{(p)}\) be defined \(A^{(p)}_{ij} = (A_{ij})^p\).

2.1 Block-encodings

Following [16] we use the following definition:

\textbf{Definition 1 (Block-encoding).} Suppose that \(A\) is an \(s\)-qubit operator, \(\alpha, \varepsilon \in \mathbb{R}_+\) and \(a \in \mathbb{N}\). Then we say that the \((s+a)\)-qubit unitary \(U\) is an \((\alpha, a, \varepsilon)\)-block-encoding\(^6\) of \(A\), if

\[
\|A - \alpha (|0\rangle \otimes I) U (|0\rangle \otimes I)\| \leq \varepsilon.
\]

Block-encodings are really intuitive to work with. For example, one can easily take the product of two block-encoded matrices by keeping their ancilla qubits separately. The following lemma shows that the errors during such a multiplication simply add up as one would expect, and the block-encoding does not introduce any additional errors.

\(^6\) Note that since \(\|U\| = 1\) we necessarily have \(\|A\| \leq \alpha + \varepsilon\).
Lemma 2. If $U$ is an $(\alpha, a, \delta)$-block-encoding of an $s$-qubit operator $A$, and $V$ is a $(\beta, b, \varepsilon)$-block-encoding of an $s$-qubit operator $B$ then\(^7\) $(I_b \otimes U)(I_a \otimes V)$ is an $(\alpha \beta, a + b, \alpha \varepsilon + \beta \delta)$-block-encoding of $AB$.

Proof.

$$
\|AB - \alpha \beta \langle 0 | \otimes a + b \rangle I (I_b \otimes U)(I_a \otimes V) (\langle 0 | \otimes a + b \rangle I)\| \\
= \|AB - \alpha \beta \langle 0 | \otimes a \rangle I (I_b \otimes U) (\langle 0 | \otimes a \rangle I) \beta (\langle 0 | \otimes b \rangle V (\langle 0 | \otimes b \rangle I)\| \\
= \|AB - \tilde{A}B + \tilde{A}B - \tilde{A}B\| = \|A - \tilde{A}\| + \|B - \tilde{B}\| \\
\leq \alpha \varepsilon + \beta \delta.
$$

The above lemma can be made more efficient in some cases when both $A$ and $B$ are significantly subnormalized.

The following theorem about block-Hamiltonian simulation is a corollary of the results of [24, Theorem 1], which also includes bounds on the propagation of errors.

Theorem 3. Suppose that $U$ is an $(\alpha, a, \varepsilon/2|t|)$-block-encoding of the Hamiltonian $H$. Then we can implement an $\varepsilon$-precise Hamiltonian simulation unitary $V$ which is an $(1, a + 2, \varepsilon)$-block-encoding of $e^{tH}$, with $O(|\alpha t| + \log(1/\varepsilon))$ uses of controlled-$U$ or its inverse and with $O(a|\alpha t| + a\log(1/\varepsilon))$ two-qubit gates.

2.2 Data structures and sparse access

We will consider the following data structure, studied in [19]. We will refer to this data structure, because it is a classical data structure, which, if stored in QROM, is addressable in superposition, but needn’t be able to store a quantum state, facilitates the implementation of certain useful quantum operations. In our complexity analysis, we consider the cost of accessing a QROM of size $N$ to be $\text{polylog}(N)$. Although this operation requires order $N$ gates [17, 3], but the gates can be arranged in parallel such that the depth of the circuit indeed remains $\text{polylog}(N)$.

The following is proven in [19].

Theorem 4 (Implementing quantum operators using an efficient data structure [19]). Let $A \in \mathbb{R}^{M \times N}$ be a matrix with $A_{ij} \in \mathbb{R}$ being the entry of the $i$-th row and the $j$-th column. If $w$ is the number of non-zero entries of $A$, then there exists a data structure of size\(^8\) $O(w \log^2(MN))$ that, given the entries $(i, j, A_{ij})$ in an arbitrary order, stores them such that time\(^8\) taken to store each entry of $A$ is $O(\log(MN))$. Once this data structure has been initiated with all non-zero entries of $A$, there exists a quantum algorithm that can perform the following maps with $\varepsilon$-precision in $O(\text{polylog}(MN/\varepsilon))$ time:

$$
\tilde{U} : |i\rangle |0\rangle \mapsto |i\rangle \frac{1}{\|A_{ii}\|} \sum_{j=1}^{N} A_{i,j} |j\rangle = |i, A_i\rangle,
$$

\(^7\) In the expression $(I_b \otimes U)(I_a \otimes V)$, the identity operator $I_b$ should be seen as acting on the ancilla qubits of $V$, and $I_a$ on those of $U$.

\(^8\) Here, for simplicity we assume that we can store a real number in 1 data register, however more realistically we should actually count the number of bits, incurring logarithmic overheads. Also in this theorem we assign unit cost for classical arithmetic operations.
\[
\hat{V} : |0\rangle_j \mapsto \frac{1}{\|A\|_F} \sum_{i=1}^{M} \|[\tilde{A}_i]\|_i |i\rangle_j = |\tilde{A}, j\rangle,
\]
where \(|\tilde{A}_i\rangle\) is the normalized quantum state corresponding to the \(i\)-th row of \(A\) and \(\tilde{A}\) is a normalized quantum state such that \(\langle \tilde{A} | \tilde{A} \rangle = \|A_i\|\), i.e. the norm of the \(i\)-th row of \(A\).

In particular, given a vector \(\vec{v} \in \mathbb{R}^{M \times 1}\) stored in this data structure, we can generate an \(\varepsilon\)-approximation of the superposition \(\sum_{i=1}^{M} v_i |i\rangle/\|\vec{v}\|\) in complexity \(\text{polylog}(M/\varepsilon)\).

As a corollary, we have the following, which allows us to generate alternative quantum state representations of the rows of \(A\), as long as we have stored \(A\) appropriately beforehand:

**Corollary 5.** If \(A^{(p)}\) is stored in a quantum data structure, then there exists a quantum algorithm that can perform the following map with \(\varepsilon\)-precision in \(\text{polylog}(MN/\varepsilon)\) time:

\[
|i\rangle_0 \mapsto |i\rangle \frac{1}{s_2(A)} \sum_{j=1}^{N} A_{i,j}^p |j\rangle.
\]

The following was proven in [20], although not in the language of block-encodings.

**Lemma 6 (Implementing block-encodings from quantum data structures).** Let \(A \in \mathbb{C}^{M \times N}\).

1. Fix \(p \in [0, 1]\). If \(A \in \mathbb{C}^{M \times N}\), and \(A^{(p)}\) and \((A^{(1-p)})^\dagger\) are both stored in quantum-accessible data structures\(^9\), then there exist unitaries \(R_U\) and \(U_L\) that can be implemented in time \(O(\text{polylog}(MN/\varepsilon))\) such that \(U_R^\dagger U_L\) is a \((\mu_p(A), \lfloor \log(M + N + 1) \rfloor, \varepsilon)\)-block-encoding of \(\tilde{A}\).

2. On the other hand, if \(A\) is stored in a quantum-accessible data structure\(^9\), then there exist unitaries \(R_U\) and \(U_L\) that can be implemented in time \(O(\text{polylog}(MN/\varepsilon))\) such that \(U_R^\dagger U_L\) is a \((\|A\|_F, \lfloor \log(M + N) \rfloor, \varepsilon)\)-block-encoding of \(\tilde{A}\).

This allows us to apply our block-encoding results in the quantum data structure setting, including Hamiltonian simulation (Section 3.1), quantum linear system solvers (Section 3.3) and implementing negative powers of a Hamiltonian (Section 3.3).

In contrast, in the sparse-access model we assume that the input matrix \(A \in \mathbb{C}^{M \times N}\) has \(s_\tau\)-sparse rows and \(s_c\)-sparse columns, such that the matrix elements can be queried via an oracle

\[
O_A: |i\rangle |j\rangle |0\rangle^{\otimes b} \mapsto |i\rangle |j\rangle |a_{ij}\rangle \quad \forall i \in [M], j \in [N].
\]

Moreover, the indices of non-zero elements of each row can be queried via an oracle

\[
O_r: |i\rangle |k\rangle \mapsto |i\rangle |r_{ik}\rangle \quad \forall i \in [N], k \in [s_r],
\]

where \(r_{ij}\) is the index for the \(j\)-th non-zero entry of the \(i\)-th row of \(A\), or if there are less than \(i\) non-zero entries, then it is \(j + N\). If \(A\) is not symmetric (or Hermitian) then we also assume the analogous oracle for columns. It is not difficult to prove [13] that a block-encoding of \(A\) can be efficiently implemented in the sparse-access input model, see [16, Lemma 48] for a direct proof.

**Lemma 7 (Constructing block-encodings for sparse-access matrices [16, Lemma 48]).** Let \(A \in \mathbb{C}^{M \times N}\) be an \(s_r\), \(s_c\) row and column-sparse matrix given in the sparse-access input model. Then for any \(\varepsilon \in (0, 1)\), we can implement a \((\sqrt{s_r s_c}, \text{polylog}(MN/\varepsilon), \varepsilon)\)-block-encoding of \(A\) with \(O(1)\) queries and \(\text{polylog}(MN/\varepsilon)\) elementary gates.

Thus, our block-encoding results also apply to the sparse access model.

---

\(^9\) Here we assume that the data structure stores the matrices with sufficient precision.
3 Techniques for block-encodings

We develop several tools within the block-encoding framework that are crucial to our applications, but also likely of independent interest. Since an input given either in the sparse-access model or as a quantum data structure can be made into a block-encoding, our block-encoding results imply analogous results in each of the sparse-access and quantum data structure models.

Throughout this section, let $\mu(A)$, and “$\mu$-encoding of $A$” be one of:

1. $\mu(A) = \|A\|_F$, the Frobenius norm of $A$, in which case $\mu$-encoding refers to a quantum data structure encoding $A$;
2. for some $p \in [0, 1]$, $\mu(A) = \sqrt{s_p(A)s_{1-p}(A)}$, where $s_p(A) = \max_j \|A_j\|_p^p$, in which case $\mu$-encoding refers to quantum data structures encoding both $A^p$ and $(A^{1-p})^T$, defined by $A_{ij}^q := (A_{ij})^q$; or
3. $\mu(A) = \sqrt{s^r s^c}$, where $s^r$ and $s^c$ are the row and column sparsities of $A$, in which case, $\mu$-encoding refers to having sparse access to $A$.

3.1 Hamiltonian simulation from quantum data structure

We first have the following important building block for our other results:

▶ Theorem 8. For any $t \in \mathbb{R}$ and $\varepsilon \in (0, 1/2)$, let $H \in \mathbb{C}^{N \times N}$ be a Hermitian matrix that is $\mu$-encoded, with $\|H\| \leq 1$. Then we can implement a unitary $\hat{U}$ that is a $(1, n + 3, \varepsilon)$-block-encoding of $e^{itH}$ in time $\tilde{O}(t \mu(A) \text{polylog}(N/\varepsilon))$.

This follows from the quantum Hamiltonian simulation algorithm of Low and Chuang that expects the input as a block-encoding, and Lemmas 6 and 7. Independently, Wang and Wossnig have proven a similar result, with $\|A\|_1 \leq \sqrt{N}$ in place of $\mu(A)$ [28].

3.2 Quantum singular value estimation

Given access to a matrix $A \in \mathbb{R}^{M \times N}$ with singular value decomposition $A = \sum_j \sigma_j |u_j\rangle \langle v_j|$, and given some input state, the quantum singular value estimation (QSVE) problem requires estimating the singular values of $A$ up to some precision with a high probability. We present a quantum algorithm for singular value estimation of a matrix $A$ given as a block-encoding. In particular, using our algorithm it is possible to obtain an estimate of $\sigma_j$, $\tilde{\sigma}_j$ such that with probability $1 - \varepsilon$, $|\sigma_j - \tilde{\sigma}_j| \leq \Delta$. We give a precise description of quantum singular value estimation, and prove the following theorem:

▶ Theorem 9. Let $\varepsilon, \Delta \in (0, 1)$, and $\varepsilon' = \frac{\varepsilon \Delta}{4 \log^2(1/\Delta)}$. Let $U$ be an $(\alpha, a, \varepsilon')$-block-encoding of a matrix $A$ that can be implemented in cost $T_U$. Then we can implement a quantum algorithm that solves QSVE of $A$ in complexity

$$O\left(\frac{\alpha}{\Delta} (a + T_U) \text{polylog}(1/\varepsilon)\right).$$

In the special case when the block-encoding is implemented by a quantum data structure, we recover the complexity of the quantum algorithm for singular value estimation by Kerenidis and Prakash [19].
3.3 Quantum linear system solver

The quantum linear system problem (QLS problem) is the following. Given access to an $N \times N$ matrix $A$, and a procedure for computing a quantum state $|b\rangle$ in the image of $A$, prepare a state that is within $\varepsilon$ of $A^+|b\rangle/\|A^+|b\rangle\|$. Given a block-encoding of $A$, we can use Theorem 3 to get a block-encoding of $e^{i\mu A}$, from which we can implement a $(2\kappa, a, \varepsilon)$-block-encoding of $A^{-1}$ (for some $a$ and $\varepsilon$), where $\kappa$ is an upper bound on $\|A^{-1}\|$. Such a block-encoding can be applied to a state $|0\rangle$ to get $\frac{1}{\delta}\langle 0 | \otimes a (A^{-1}|b\rangle) + |0\rangle$ for some unnormalized state $|0\rangle$ orthogonal to every state with $|0\rangle$ in the first $a$ registers. Performing amplitude amplification on this procedure, we can approximate the state $A^{-1}|b\rangle/\|A^{-1}|b\rangle\|$ at a cost

$$\mathcal{O}\left(\kappa (\alpha (T_U + a) \log^2 \left(\frac{\kappa}{\varepsilon}\right) + T_V) \log(\kappa) \right).$$

From this we get the following theorem:

**Theorem 10.** Let $\kappa \geq 2$, and $A$ be an $N \times N$ Hermitian matrix such that the non-zero eigenvalues of $H$ lie in the range $[-1, -1/\varepsilon] \bigcup [1/\varepsilon, 1]$. Suppose that for $\delta = O(\varepsilon/(\kappa^2 \log^3 \frac{\kappa}{\varepsilon}))$ we have a unitary $U$ that is a $(\alpha, a, \delta)$-block-encoding of $A$ that can be implemented using $T_U$ elementary gates. Also suppose that we can prepare an input state $|\psi\rangle$ which spans the eigenvectors of $A$ in time $T_V$. Then there exists a variable time amplitude amplification based quantum algorithm that outputs a state that is $\varepsilon$-close to $A^{-1}|\psi\rangle/\|A^{-1}|\psi\rangle\|$ at a cost

$$\mathcal{O}\left(\kappa (\alpha (T_U + a) \log^2 \left(\frac{\kappa}{\varepsilon}\right) + T_V) \log(\kappa) \right).$$

For (ii), we use our new technique of variable time amplification estimation.

Finally, we generalize our QLS solver to apply $A^{-c}$ for any $c \in (0, \infty)$. Using variable-time amplification techniques we show the following:

**Theorem 12.** Let $\kappa \geq 2$, $c \in (0, \infty)$, $q = \max(1, c)$, and $A$ be an $N \times N$ Hermitian matrix such that the eigenvalues of $A$ lie in the range $[-1, -1/\varepsilon] \bigcup [1/\varepsilon, 1]$. Suppose that for $\delta = O(\varepsilon/(\kappa^2 \log^3 \frac{\kappa}{\varepsilon}))$ we have a unitary $U$ that is a $(\alpha, a, \delta)$-block-encoding of $A$ which can be implemented using $T_U$ elementary gates. Also suppose that we can prepare an input state $|\psi\rangle$ that is spanned by the eigenvectors of $A$ in time $T_V$. Then there exists a variable time amplitude amplification based quantum algorithm that outputs a state that is $\varepsilon$-close to $A^{-c}|\psi\rangle/\|A^{-c}|\psi\rangle\|$ at a cost of

$$\mathcal{O}\left(\alpha q^3 (T_U + a) \log^2 \left(\frac{\kappa^c}{\varepsilon}\right) + \kappa^c T_V \log(\kappa) \right).$$

---

10 In the special case when $\|A\| = 1$, $\kappa$ is an upper bound on the condition number of $A$, justifying the notation.

11 Since for any matrix $C \in \mathbb{C}^{M \times N'}$ we have that $\overline{C} \in \mathbb{C}^{(M+N) \times (M+N)}$ is Hermitian, and the eigenvalues of $\overline{C}$ are $\pm 1$ times the singular values of $C$, this statement and its corollaries also apply to non-symmetric matrices.
Also, there exists a variable time amplitude amplification based quantum algorithm that outputs a number $\Gamma$ such that

$$1 - \varepsilon \leq \frac{\Gamma}{\|A^\dagger|\psi\|} \leq 1 + \varepsilon,$$

with success probability at least $1 - \delta$, at a cost

$$O\left(\frac{1}{\varepsilon} \left( \alpha \kappa^3 (T_U + a) q \log^2 \left( \frac{\kappa q}{\varepsilon} \right) + \kappa^3 T \log \left( \frac{\log (\kappa q)}{\delta} \right) \right) \right).$$

### 4 Application to least squares

The problem of ordinary least squares (OLS) is the following. Given data points $\{(\vec{x}^{(i)}, y^{(i)})\}_{i=1}^M$ for $\vec{x}^{(1)}, \ldots, \vec{x}^{(M)} \in \mathbb{R}^N$ and $y^{(1)}, \ldots, y^{(M)} \in \mathbb{R}$, find $\vec{\beta} \in \mathbb{R}^N$ that minimizes:

$$\sum_{i=1}^M (y^{(i)} - \vec{\beta}^T \vec{x}^{(i)})^2.$$  

(2)

The motivation for this task is the assumption that the samples are obtained from some process such that at every sample $i$, $y^{(i)}$ depends linearly on $\vec{x}^{(i)}$, up to some random noise, so $y^{(i)}$ is drawn from a random variable $\vec{\beta}^T \vec{x}^{(i)} + E_i$, where $E_i$ is a random variable with mean 0, for example, a Gaussian. The vector $\vec{\beta}$ that minimizes (2) represents a good estimate of the underlying linear function. We assume $M \geq N$ so that it is feasible to recover this linear function.

We can generalize this task to settings in which certain samples are thought to be of higher quality than others, for example, because the random variables $E_i$ are not identical.

We express this belief by assigning a positive weight $w_i$ to each sample, and minimizing

$$\sum_{i=1}^M w_i (y^{(i)} - \vec{\beta}^T \vec{x}^{(i)})^2.$$  

(3)

Let $X \in \mathbb{R}^{M \times N}$ be the matrix such that its $i$th row is $\vec{x}^{(i)^T}$. Finding $\vec{\beta}$ given $X$, $\vec{w}$ and $\vec{y}$ is the problem of weighted least squares (WLS).

We can further generalize to settings in which the random variables $E_i$ for sample $i$ are correlated. In the problem of generalized least squares (GLS), the presumed correlations in error between pairs of samples are given in a symmetric non-singular covariance matrix $\Omega$.

We then want to find the vector $\vec{\beta}$ that minimizes

$$\sum_{i,j=1}^M \Omega^{-1}_{i,j} (y^{(i)} - \vec{\beta}^T \vec{x}^{(i)})(y^{(j)} - \vec{\beta}^T \vec{x}^{(j)}).$$  

(4)

We will consider solving quantum versions of these problems. Specifically, a quantum WLS solver (resp. quantum GLS solver) is given access to $\vec{y} \in \mathbb{R}^M$, $X \in \mathbb{R}^{M \times N}$, and positive weights $w_1, \ldots, w_M$ (resp. $\Omega$), in some specified manner, and outputs an $\varepsilon$-approximation of a quantum state $\sum_i \beta_i \hat{i} / \|\vec{\beta}\|$, where $\vec{\beta}$ minimizes the expression in (3) (resp. (4)).

Quantum algorithms for least squares fitting were first considered in [32]. They considered query access to $X$, and a procedure for outputting $|\vec{y}\rangle = \sum_i y_i |i\rangle / \|\vec{y}\|$, which we refer to as the sparse-access input model. They present a quantum OLS solver, outputting a state
We stress that our algorithms do not achieve the minimum possible
where κ is the condition number. To compute a state proportional to \( \vec{\beta} \),
they first apply \( X^T \) to \( | y \rangle \) to get a state proportional to \( X^T \vec{y} \), using techniques similar to
[18]. They then apply \( (X^T X)^{-1} \) using the quantum linear system solving algorithm of [18],
giving a final state proportional to \( (X^T X)^{-1}X^T \vec{y} = X^+ \vec{y} \).

The approach of [32] was later improved upon by [21], who also give a quantum OLS solver
in the sparse-access input model. Unlike [32], they apply \( X^+ \) directly, by using Hamiltonian
simulation of \( X \) and phase estimation to estimate the singular values of \( X \), and then apply a
rotation depending on the inverse singular value if it’s larger than 0, and using amplitude
amplification to de-amplify the singular-value-zero parts of the state. This results in an
algorithm with complexity \( \tilde{O}(s \kappa^3 \log(M + N)/\varepsilon^2) \).

Several works have also considered quantum algorithms for least squares problems with
a classical output. The first, due to Wang [30], outputs the vector \( \vec{\beta} \) in a classical form.
The input model should be compared with the sparse-access model – although \( \vec{y} \) is given in
classical random access memory, an assumption about the regularity of \( \vec{y} \) means the quantum
state \( | y \rangle \) can be efficiently prepared. The algorithm also requires a regularity condition on
the matrix \( X \). The algorithm’s complexity is \( \text{poly}(\log M, N, \kappa, \frac{1}{\varepsilon}) \). Like [21], Wang’s algorithm
uses techniques from quantum linear system solving to apply \( X^+ \) directly to \( | y \rangle \). To do this,
Hamiltonian simulation of \( X \) is accomplished via what we would call a block-encoding of \( X \).
This outputs a state proportional to \( X^+ \vec{y} \), whose amplitudes can be estimated one-by-one
to recover \( \vec{\beta} \).

A second algorithm to consider least squares with a classical output is [27], which does
not output \( \vec{\beta} \), but rather, given an input \( x \), outputs \( x^T \vec{\beta} \), thus predicting a new data point.
This algorithm requires that \( x, \vec{y} \), and even \( X \) be given as quantum states, and assumes
that \( X \) has low approximate rank. The algorithm uses techniques from quantum principal
component analysis [23], and runs in time \( \mathcal{O}(\log(N)\kappa^2/\varepsilon^3) \).

Recently, Kerenidis and Prakash introduced the quantum data structure input model
[19]. This input model fits data analysis tasks, because unlike in more abstract problems
such as Hamiltonian simulation, where the input matrix may be assumed to be sparse
and well-structured so that we can hope to have implemented efficient subroutines to find
the non-zero entries of the rows and columns, the input to least squares is generally noisy
data for which we may not assume any such structure. In Ref. [20], utilizing this data
structure, they solve the quantum version of the weighted least squares problem. Their
algorithm assumes access to quantum data structures storing \( X \), or some closely related
matrix (see Section 2.2), \( W = \text{diag}(\vec{w}) \), and \( \vec{y} \), and have running time \( \tilde{O}\left(\frac{\kappa^4\mu}{\varepsilon^2} \text{polylog}(MN)\right) \),
where \( \kappa \) is the condition number of \( X^T \sqrt{W} \), and \( \mu \) is some prior choice of \( \left\| X^T \sqrt{W} \right\|_F \) or
\( \sqrt{s_2 p (X^T \sqrt{W})s_2 (1-p)(X^T \sqrt{W})} \) for some \( p \in [0, 1] \)[12]. Note that the choice of \( \mu \) impacts
the way \( X \) must be encoded, leading to a family of algorithms requiring slightly different
encodings of the input.

\[12\] We stress that our algorithms do not achieve the minimum possible \( \mu \), but rather, we need to store the
input in QROM with a particular \( \mu \) in mind. We might more accurately describe the quantum data
structure input model as a family of input models, parametrized by \( \mu \).
4.1 Our results

We give quantum WLS and GLS solvers in the model where the input is given as a block-encoding. As a special case, we get quantum WLS and GLS solvers in the quantum data structure input model of Kerendis and Prakash. First, we give the following WLS solver:

**Theorem 13.** Let $A = \sqrt{W}X$ such that $\|A^\dagger\| \leq \kappa_A$. Suppose $\sqrt{W}y$ is stored in a quantum-accessible data structure, and $A$ is $\mu$-encoded. Suppose the data points have residual error $SS_W^W$ satisfying $SS_W^W \leq \eta$. Then we can implement a quantum WLS solver with error $\varepsilon$ in complexity:

$$\tilde{O} \left( \frac{\kappa_A \mu(A)}{\sqrt{1-\eta}} \text{polylog} (MN/\varepsilon) \right).$$

The residual error is a measure of how well the data can be linearly approximated. It is reasonable to assume that it’s close to 0, as otherwise, linear regression is inappropriate. Indeed, previous work seems to implicitly assume it’s bounded by a constant below 1.

Theorem 13 is a 6-th power improvement in the dependence on $\kappa$, and an exponential improvement in the dependent on $1/\varepsilon$ as compared with the quantum WLS solver of [20]. As a special case we get a quantum OLS solver, which compares favourably to previous quantum OLS solvers in the sparse-access model [32, 21] in having a linear dependence on $\kappa$, and a polylog$(1/\varepsilon)$ dependence on the precision. However, these previous results rely on QLS solver subroutines which have since been improved, so their complexity can also likely be improved.

In addition, we give the first quantum GLS solver. We first show how to implement a GLS solver when the inputs are given as block-encodings. We prove the following general Theorem:

**Theorem 14.** Suppose that we have a unitary $U_y$ preparing a quantum state proportional to $\vec{y}$ in complexity $T_y$. Suppose $X \in \mathbb{R}^{M \times N}$, $\Omega \in \mathbb{R}^{M \times M}$ are such that $\|X\| \leq 1$, $\|\Omega\| \leq 1$ and $\Omega > 0$ is positive definite. Suppose that we have access to $U_X$ that is an $(a_X, a_X, 0)$-block-encoding of $X$ which has complexity $T_X \geq a_X$, and similarly we have access $U_{\Omega}$ that is an $(a_\Omega, a_\Omega, 0)$-block-encoding of $\Omega^{-1/2}$ which has complexity $T_\Omega \geq a_\Omega$. Let $A := \Omega^{-1/2}X$, and suppose we have the following upper bounds: $\|A^\dagger\| \leq \kappa_A$, $\|\Omega^{-1}\| \leq \kappa_\Omega$, and $SS^\Omega \leq \eta$. Then we can implement a quantum GLS-solver with error $\varepsilon$ in complexity

$$O \left( \frac{\kappa_A \log(k_A)}{\sqrt{1-\eta}} \left( (\sqrt{\kappa_A a_X T_X} + a_\Omega T_\Omega) \log^3 \left( \frac{\kappa_A}{\varepsilon} \right) + \sqrt{\kappa_A T_y} \right) \right).$$

As a special case, we get the following:

**Corollary 15.** Suppose $\vec{y}$ is stored in a quantum-accessible data structure, and $X, \Omega$ are such that $\|X\| \leq 1$, $\|\Omega\| \leq 1$ and $\Omega$ is positive definite. Further assume they are $\mu$-encoded and $\|X^\dagger\| \leq \kappa_X$, $\|\Omega^\dagger\| \leq \kappa_\Omega$. Then we can implement a quantum GLS-solver with error $\varepsilon$ in complexity

$$\tilde{O} \left( \frac{\kappa_X \sqrt{\kappa_\Omega}}{\sqrt{1-\eta}} (\mu(X) + \mu(\Omega) \kappa_\Omega) \text{polylog} (MN/\varepsilon) \right).$$

Note that for Theorem 14 and Corollary 15, the parameter $\kappa_X$ is not exactly the condition number of $X$. In fact, what we require is that the product of the upper bounds on $\|X\|$ and $\|X^\dagger\|$ respectively, be upper bounded by $\kappa_X$. Thus without loss of generality it suffices to consider that $\|X\| \leq 1$ and $\|X^\dagger\| \leq \kappa_X$ (same holds for $\Omega$).
References


33:14  The Power of Block-Encoded Matrix Powers